aws

Developer Guide

Amazon Forecast

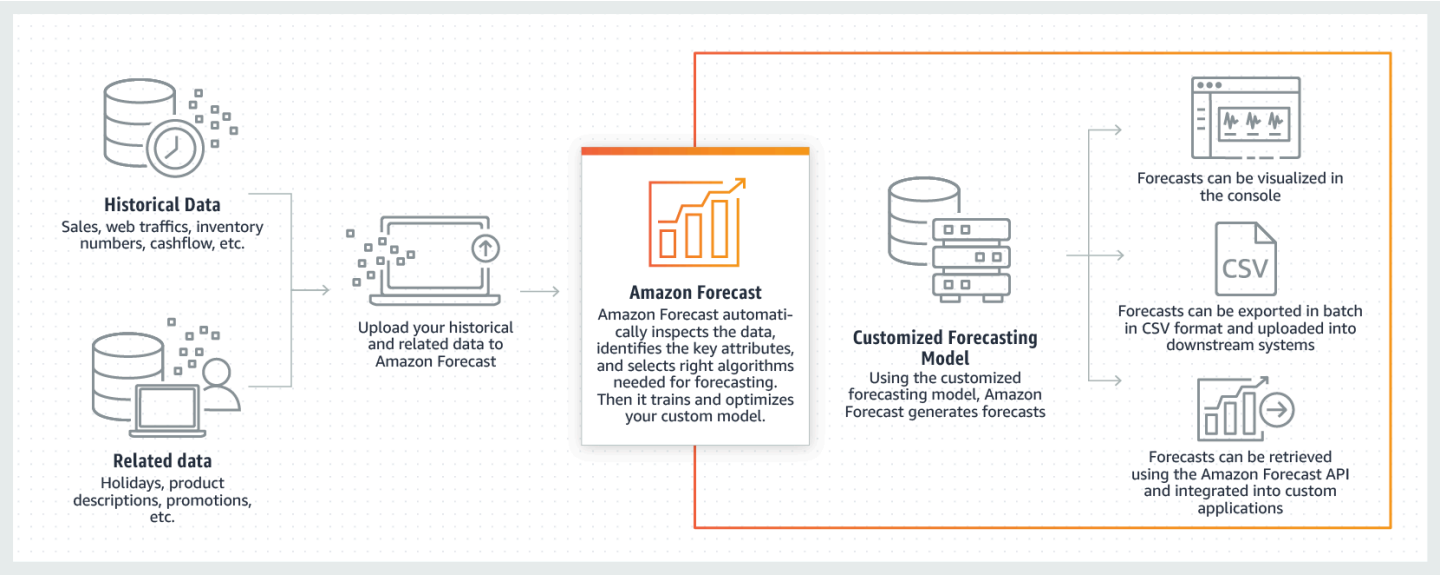
Amazon Forecast: Developer Guide

Table of Contents

What Is Amazon Forecast?

Amazon Forecast is a fully managed service that uses statistical and machine learning algorithms to deliver highly accurate time-series forecasts. Based on the same technology used for time-series forecasting at Amazon.com, Forecast provides state-of-the-art algorithms to predict future time-series data based on historical data, and requires no machine learning experience.

Time-series forecasting is useful in multiple fields, including retail, finance, logistics, and healthcare. You can also use Forecast to predict domain-specific metrics for your inventory, workforce, web traffic, server capacity, and finances.



For more information about the technical aspects of Amazon Forecast, see Time Series Forecasting Principles with Amazon Forecast.

Topics

- Using Amazon Forecast
- Features of Amazon Forecast
- Pricing for Amazon Forecast
- Are You a First-Time User of Amazon Forecast?
- Using Forecast with an AWS SDK

Using Amazon Forecast

You can use the APIs, AWS Command Line Interface (AWS CLI), Python Software Development Kit (SDK), and Amazon Forecast console to import time series datasets, train predictors, and generate forecasts.

Here are some common use cases for Amazon Forecast:

- **Retail demand planning** – Predict product demand, allowing you to more accurately vary inventory and pricing at different store locations.
- **Supply chain planning** – Predict the quantity of raw goods, services, or other inputs required by manufacturing.
- **Resource planning** – Predict requirements for staffing, advertising, energy consumption, and server capacity.
- **Operational planning** – Predict levels of web traffic, AWS usage, and IoT sensor usage.

Features of Amazon Forecast

Amazon Forecast automates much of the time-series forecasting process, enabling you to focus on preparing your datasets and interpreting your predictions.

Forecast provides the following features:

- **Automated machine learning** – Forecast automates complex machine learning tasks by finding the optimal combination of machine learning algorithms for your datasets.
- **State-of-the-art algorithms** – Apply a combination of machine learning algorithms that are based on the same technology used at Amazon.com. Forecast offers a wide range of training algorithms, from commonly used statistical methods to complex neural networks.
- **Missing value support** – Forecast provides several filling methods to automatically handle missing values in your datasets.
- **Additional built-in datasets** – Forecast can automatically incorporate built-in datasets to improve your model. These datasets are already feature engineered and do not require additional configuration.

Pricing for Amazon Forecast

With Amazon Forecast, you pay only for what you use. There are no minimum fees and no upfront commitments. The costs of Amazon Forecast depend on the number generated forecasts, data storage, and training hours.

The AWS Free Tier allows you a monthly limit of up to 10,000 time series forecasts, up to 10GB of storage, and up to 10 hours of training time. The Amazon Forecast free tier is valid for the first two months of usage.

For a complete list of charges and prices, see Amazon Forecast pricing.

Are You a First-Time User of Amazon Forecast?

If you are a first-time user of Amazon Forecast, we recommend that you start with the following pages:

1. How Amazon Forecast Works – Learn about the key concepts and the process of importing datasets, creating predictors, and generating forecasts.
2. Getting Started – Follow one of the tutorials to create your first Amazon Forecast forecasting predictor.
3. API Reference – Familiarize yourself with the Amazon Forecast API actions and data types.

Using Forecast with an AWS SDK

AWS software development kits (SDKs) are available for many popular programming languages. Each SDK provides an API, code examples, and documentation that make it easier for developers to build applications in their preferred language.

SDK documentation	Code examples
AWS SDK for C++	AWS SDK for C++ code examples
AWS CLI	AWS CLI code examples
AWS SDK for Go	AWS SDK for Go code examples

SDK documentation	Code examples
AWS SDK for Java	AWS SDK for Java code examples
AWS SDK for JavaScript	AWS SDK for JavaScript code examples
AWS SDK for Kotlin	AWS SDK for Kotlin code examples
AWS SDK for .NET	AWS SDK for .NET code examples
AWS SDK for PHP	AWS SDK for PHP code examples
AWS Tools for PowerShell	Tools for PowerShell code examples
AWS SDK for Python (Boto3)	AWS SDK for Python (Boto3) code examples
AWS SDK for Ruby	AWS SDK for Ruby code examples
AWS SDK for Rust	AWS SDK for Rust code examples
AWS SDK for SAP ABAP	AWS SDK for SAP ABAP code examples
AWS SDK for Swift	AWS SDK for Swift code examples

Example availability

Can't find what you need? Request a code example by using the **Provide feedback** link at the bottom of this page.

How Amazon Forecast Works

When creating forecasting projects in Amazon Forecast, you work with the following resources:

- **Importing Datasets** – *Datasets* are collections of your input data. Dataset groups are collections of datasets that contain complimentary information. Forecast algorithms use your dataset groups to train custom forecasting models, called predictors.
- **Training Predictors** – *Predictors* are custom models trained on your data. You can train a predictor by choosing a prebuilt algorithm,or by choosing the AutoML option to have Amazon Forecast pick the best algorithm for you.
- **Generating Forecasts** – You can generate forecasts for your time-series data, query them using the QueryForecast API, or visualize them in the console.

Setting Up

Before using Amazon Forecast to evaluate or forecast time-series data, create an AWS account, configure access permissions, and set up the AWS Command Line Interface (AWS CLI).

Topics

Sign Up for AWS

When you sign up for Amazon Web Services (AWS), your AWS account is automatically signed up for all services in AWS, including Amazon Forecast. You are charged only for the services that you use.

Set Up the AWS CLI

The AWS Command Line Interface (AWS CLI) is a unified developer tool for managing AWS services, including Amazon Forecast. We recommend that you install and use it.

1. To install the AWS CLI, follow the instructions in Installing the AWS Command Line Interface in the *AWS Command Line Interface User Guide*.
2. To configure the AWS CLI and set up a profile to call it, follow the instructions in Configuring the AWS CLI in the *AWS Command Line Interface User Guide*.
3. To confirm that the AWS CLI profile is configured correctly, run the following command in a command window:

   ```
   aws configure --profile default
   ```

   If your profile has been configured correctly, you should see output similar to the following:

   ```
   AWS Access Key ID [****************52FQ]:
   AWS Secret Access Key [****************xgyZ]:
   Default region name [us-west-2]:
   ```

```
Default output format [json]:
```

4. To verify that the AWS CLI is configured for use with Amazon Forecast, run the following commands.

```
aws forecast help
```

```
aws forecastquery help
```

If the AWS CLI is configured correctly, you will see a list of the supported AWS CLI commands for Amazon Forecast or Amazon Forecast Query.

Set Up Permissions for Amazon Forecast

Amazon Forecast uses Amazon Simple Storage Service (Amazon S3) to store the target time-series data that are used to train predictors that can generate forecasts. To access Amazon S3 on your behalf, Amazon Forecast needs your permission.

To grant Amazon Forecast permission to use Amazon S3 on your behalf, you must have an AWS Identity and Access Management (IAM) role and IAM policy in your account. The IAM policy specifies the required permissions, and must be attached to the IAM role.

To create the IAM role and policy and to attach the policy to the role, you can use the IAM console or the AWS Command Line Interface (AWS CLI).

> **Note**
>
> Forecast does not communicate with Amazon Virtual Private Cloud and is unable to support the Amazon S3 VPCE gateway. Using S3 buckets that only allow VPC access will result in an `AccessDenied` error.

Topics

Create an IAM Role for Amazon Forecast (IAM Console)

You can use the AWS IAM console to do the following:

- Create an IAM role with Amazon Forecast as a trusted entity
- Create an IAM policy with permissions that allows Amazon Forecast to show, read, and write data in an Amazon S3 bucket
- Attach the IAM policy to the IAM role

To create an IAM role and policy that allows Amazon Forecast to access Amazon S3 (IAM console)

1. Sign in to the IAM console (https://console.aws.amazon.com/iam).
2. Choose **Policies** and do the following to create the required policy:

   a. Click **Create policy**.

   b. On the **Create policy** page, in the policy editor, choose the **JSON** tab.

   c. Copy the following policy and replace the text in the editor by pasting the this policy over it. Be sure to replace *bucket-name* with the name of your S3 bucket, then choose **Review policy**.

```
{
   "Version":"2012-10-17",
   "Statement":[
      {
         "Effect":"Allow",
         "Action":[
            "s3:Get*",
            "s3:List*",
            "s3:PutObject"
         ],
         "Resource":[
            "arn:aws:s3:::bucket-name",
            "arn:aws:s3:::bucket-name/*"
         ]
      }
   ]
}
```

Click **Next: Tags**

d. Optionally, you can assign tags to this policy. Click **Next: Review**.

e. In **Review policy**, for **Name**, enter a name for the policy. For example, `AWSS3BucketAccess`. Optionally, provide a description for this policy, then choose **Create policy**.

3. In the navigation pane, choose **Roles**. Then do the following to create the IAM role:

a. Choose **Create role**.

b. For **Trusted entity type**, choose **AWS service**.

For **Use case**, select **Forecast** from either the **Common use cases** section or the **Use cases for other AWS services** drop-down list. If you cannot find **Forecast**, choose **EC2**.

Click **Next**.

c. In the **Add permissions** section, click **Next**.

d. In the **Name, review, and create** section, for **Role name**, enter a name for the role (for example, `ForecastRole`). Update the description for the role in **Role description**, then choose **Create role**.

e. You should now be back at the **Roles** page. Choose the new role to open the role's details page.

f. In the **Summary**, copy the **Role ARN** value and save it. You need it to import a dataset into Amazon Forecast.

g. If you didn't choose **Amazon Forecast** as the service that will use this role, choose **Trust relationships**, and then choose **Edit trust relationship** to update the trust policy as follows.

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Sid": "",
      "Effect": "Allow",
      "Principal": {
        "Service": "forecast.amazonaws.com"
      },
      "Action": "sts:AssumeRole",
      "Condition": {
```

```
        "StringEquals": {
          "aws:SourceAccount": "account-id"
        },
        "ArnLike": {
          "aws:SourceArn": "arn:aws:forecast:region:account-id:*"
        }
      }
    }
  ]
}
```

h. **[OPTIONAL]** When using a KMS key to enable encryption, attach the KMS key and ARN:

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Sid": "ForecastKMS",
            "Effect": "Allow",
            "Action": "kms:*",
            "Resource": "arn:aws:kms:region:account-id:key/KMS-key-id"
        }
    ]
}
```

Create an IAM Role for Amazon Forecast (AWS CLI)

You can use the AWS CLI to do the following:

- Create an IAM role with Amazon Forecast as a trusted entity
- Create an IAM policy with permissions that allows Amazon Forecast to show, read, and write data in an Amazon S3 bucket
- Attach the IAM policy to the IAM role

To create an IAM role and policy that allows Amazon Forecast to access Amazon S3 (AWS CLI)

1. Create an IAM role with Amazon Forecast as a trusted entity that can assume the role for you:

```
aws iam create-role \
 --role-name ForecastRole \
```

```
 --assume-role-policy-document '{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Sid": "",
      "Effect": "Allow",
      "Principal": {
        "Service": "forecast.amazonaws.com"
      },
      "Action": "sts:AssumeRole",
      "Condition": {
        "StringEquals": {
          "aws:SourceAccount": "account-id"
        },
        "ArnLike": {
          "aws:SourceArn": "arn:aws:forecast:region:account-id:*"
        }
      }
    }
  ]
}'
```

This command assumes that the default AWS configuration profile is targeted for an AWS Region supported by Amazon Forecast. If you have configured another profile (for example, `aws-forecast`) to target an AWS Region that is not supported by Amazon Forecast, you must explicitly specify that configuration by including the `profile` parameter in the command, for example, `--profile aws-forecast`. For more information about setting up an AWS CLI configuration profile, see the AWS CLI configure command.

If the command successfully creates the role, it returns it as output, which should look similar to the following:

```
{
    "Role": {
        "Path": "/",
        "RoleName": "ForecastRole",
        "RoleId": your-role-ID,
        "Arn": "arn:aws:iam::your-acct-ID:role/ForecastRole",
        "CreateDate": "creation-date",
        "AssumeRolePolicyDocument": {
            "Version": "2012-10-17",
            "Statement": [
```

```
                {
                    "Sid": "",
                    "Effect": "Allow",
                    "Principal": {
                        "Service": "forecast.amazonaws.com"
                    },
                    "Action": "sts:AssumeRole",
                    "Condition": {
                        "StringEquals": {
                            "aws:SourceAccount": "your-acct-ID"
                        },
                        "ArnLike": {
                            "aws:SourceArn": "arn:aws:forecast:region:your-acct-
ID:*"
                        }
                    }
                }
            ]
        }
    }
}
```

Record the role's ARN. You need it when you import a dataset to train an Amazon Forecast predictor.

2. Create an IAM policy with permissions to list, read, and write data in Amazon S3, and attach it to the IAM role that you created in Step 1:

```
aws iam put-role-policy \
  --role-name ForecastRole \
  --policy-name ForecastBucketAccessPolicy \
  --policy-document '{
   "Version":"2012-10-17",
   "Statement":[
      {
         "Effect":"Allow",
         "Action":[
            "s3:Get*",
            "s3:List*",
            "s3:PutObject"
         ],
         "Resource":[
            "arn:aws:s3:::bucket-name",
```

```
            "arn:aws:s3:::bucket-name/*"
         ]
      }
   ]
}'
```

3. **[OPTIONAL]** When using a KMS key to enable encryption, attach the KMS key and ARN:

```
aws iam put-role-policy \
  --role-name ForecastRole \
  --policy-name ForecastBucketAccessPolicy \
  --policy-document '{
   "Version":"2012-10-17",
   "Statement":[
      {
         "Effect":"Allow",
         "Action":[
            "s3:Get*",
            "s3:List*",
            "s3:PutObject"
         ],
         "Resource":[
            "arn:aws:s3:::bucket-name",
            "arn:aws:s3:::bucket-name/*"
         ]
      }
   ]
}'aws iam put-role-policy \
  --role-name ForecastRole \
  --policy-name ForecastKMSAccessPolicy \
  --policy-document '{
   "Version":"2012-10-17",
   "Statement":[
      {
         "Effect":"Allow",
         "Action":[
        "kms:DescribeKey",
        "kms:CreateGrant",
        "kms:RetireGrant"
         ],
         "Resource":[
         "arn:aws:kms:region:account-id:key/KMS-key-id"
         ]
```

```
      }
   ]
}'
```

Cross-service confused deputy prevention

The confused deputy problem is a security issue where an entity that doesn't have permission to perform an action can coerce a more-privileged entity to perform the action. In AWS, cross-service impersonation can result in the confused deputy problem. Cross-service impersonation can occur when one service (the calling service) calls another service (the called service). The calling service can be manipulated to use its permissions to act on another customer's resources in a way it should not otherwise have permission to access. To prevent this, AWS provides tools that help you protect your data for all services with service principals that have been given access to resources in your account.

We recommend using the `aws:SourceArn` and `aws:SourceAccount` global condition context keys in resource policies to limit the permissions that Identity and Access Management (IAM) gives Amazon Forecast access to your resources. If you use both global condition context keys, the `aws:SourceAccount` value and the account in the `aws:SourceArn` value must use the same account id when used in the same policy statement.

Getting Started

To get started using Amazon Forecast, you do the following.

- Create a Forecast dataset and import training data.
- Create a Forecast predictor, which you use generate forecasts based on your time-series data. Forecast applies the optimal combination of algorithms to each time series in your datasets.
- Generate a forecast.

In this exercise, you use a modified version of a publicly available electricity usage dataset to train a predictor. For more information, see ElectricityLoadDiagrams20112014 Data Set. The following are sample rows from the dataset:

```
2014-01-01 01:00:00,   2.53807106598985, client_0
2014-01-01 01:00:00, 23.648648648648624, client_1
2014-01-01 02:00:00,  9.648648648612345, client_0
```

For this exercise, you use the dataset to train a predictor, and then predict the hourly electricity usage by client.

You can use either the Forecast console or the AWS Command Line Interface (AWS CLI) for this exercise. Pay attention to the default regions of the Amazon Forecast console, the AWS CLI, and the Amazon Forecast SDKs, as Amazon Forecast resources are not shared across regions.

Important

Before you begin, make sure that you have an AWS account and have installed the AWS CLI. For more information, see Setting Up. We also recommend that you review How Amazon Forecast Works.

Topics

- Getting Started (Python Notebooks)
- Clean Up Resources

Prepare Input Data

Regardless of whether you use the Amazon Forecast console or the AWS Command Line Interface (AWS CLI) to set up a forecasting project, you need to set up your input data. To prepare your data, you do the following:

- Download training data to your computer and upload it to an Amazon Simple Storage Service (Amazon S3) bucket in your AWS account. To import your data to an Amazon Forecast dataset, you must store it in an Amazon S3 bucket.
- Create an AWS Identity and Access Management (IAM) role. You give Amazon Forecast permission to access your S3 bucket with the IAM role. For more information about IAM roles, see IAM Roles in the *IAM User Guide*.

To prepare training data

1. Download the zip file, electricityusagedata.zip.

   For this exercise, you use a modified version of the individual household electric power consumption dataset. (Dua, D. and Karra Taniskidou, E. (2017). UCI Machine Learning Repository [http://archive.ics.uci.edu/ml]. Irvine, CA: University of California, School of Information and Computer Science.) We aggregate the usage data hourly.
2. Unzip the content and save it locally as `electricityusagedata.csv`.
3. Upload the data file to an S3 bucket.

   For step-by-step instructions, see Uploading Files and Folders by Using Drag and Drop in the *Amazon Simple Storage Service User Guide.*
4. Create an IAM role.

   If you want to use the AWS CLI for the Getting Started exercise, you must create an IAM role. If you use the console, you can have it create the role for you. For step-by-step instructions, see Set Up Permissions for Amazon Forecast.

After you finish uploading the data to Amazon S3, you are ready to use the Amazon Forecast console or the AWS CLI to import training data, create a predictor, generate a forecast, and see the forecast.

- Getting Started (Console)
- Getting Started (AWS CLI)

Getting Started (Console)

In this exercise, you use the Amazon Forecast console to import time-series data of electricity usage, create an predictor based on the input dataset, and make predictions of future electricity usage based on the forecast horizon.

For this exercise, you use a modified version the individual household electric power consumption dataset. (Dua, D. and Karra Taniskidou, E. (2017). UCI Machine Learning Repository [http://archive.ics.uci.edu/ml]. Irvine, CA: University of California, School of Information and Computer Science.) We aggregate the usage data hourly. The modified data is available as zip file, electricityusagedata.zip.

Prerequisites

- An AWS account. If you don't already have an AWS account, create one as described in Sign Up for AWS.
- Training data in your Amazon Simple Storage Service (Amazon S3) bucket. For more information, see Prepare Input Data.
- An AWS Identity and Access Management (IAM) role that allows Amazon Forecast to read and write to your S3 buckets. For more information, see Create an IAM Role for Amazon Forecast (IAM Console).

Be aware that there are several steps in this exercise that require several minutes to a few hours to complete.

Step 1: Import Training Data

To import time-series data into Amazon Forecast, create a dataset group, choose a domain for your dataset group, specify the details of your data, and point Amazon Forecast to the S3 location of your data. The target time series used in this example is historical electricity usage data.

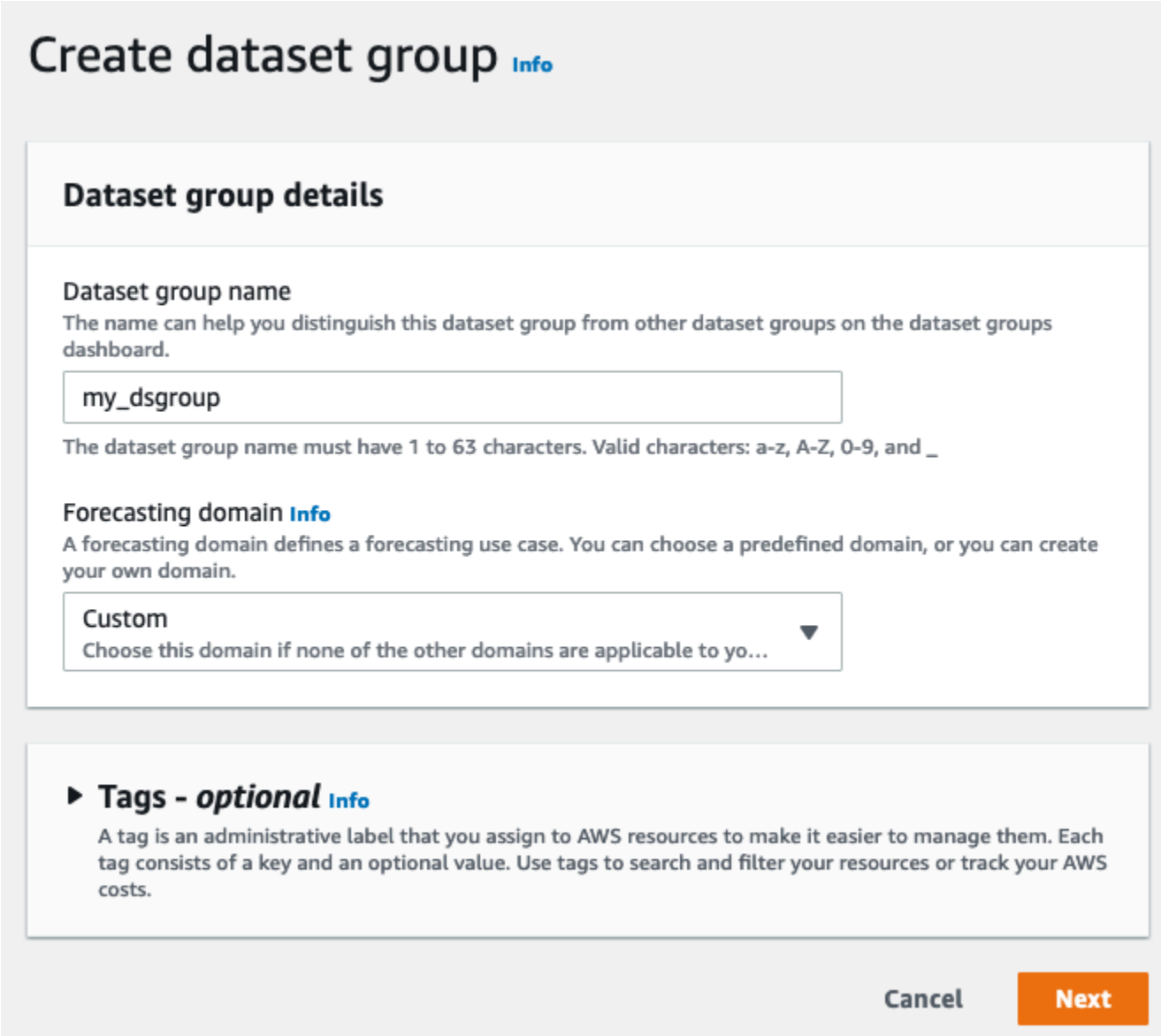
> **Note**
>
> This exercise assumes that you have not created any dataset groups. If you previously created a dataset group, what you see will vary slightly from the following screenshots and instructions.

To import time-series data for forecasting

1. Open the Amazon Forecast console at https://console.aws.amazon.com/forecast/.
2. On the Amazon Forecast home page, choose **Create dataset group**.
3. On the **Create dataset group** page, for **Dataset group details**, provide the following information:

   - **Dataset group name** – Enter a name for your dataset group.
   - **Forecasting domain** – From the drop-down menu, choose **Custom**. For more information about how to choose a forecasting domain, see dataset domains and types.

   Leave the **Tags** section unchanged. Your screen should look similar to the following:



4. Choose **Next**.
5. On the **Create target time series dataset** page, for **Dataset details**, provide the following information:

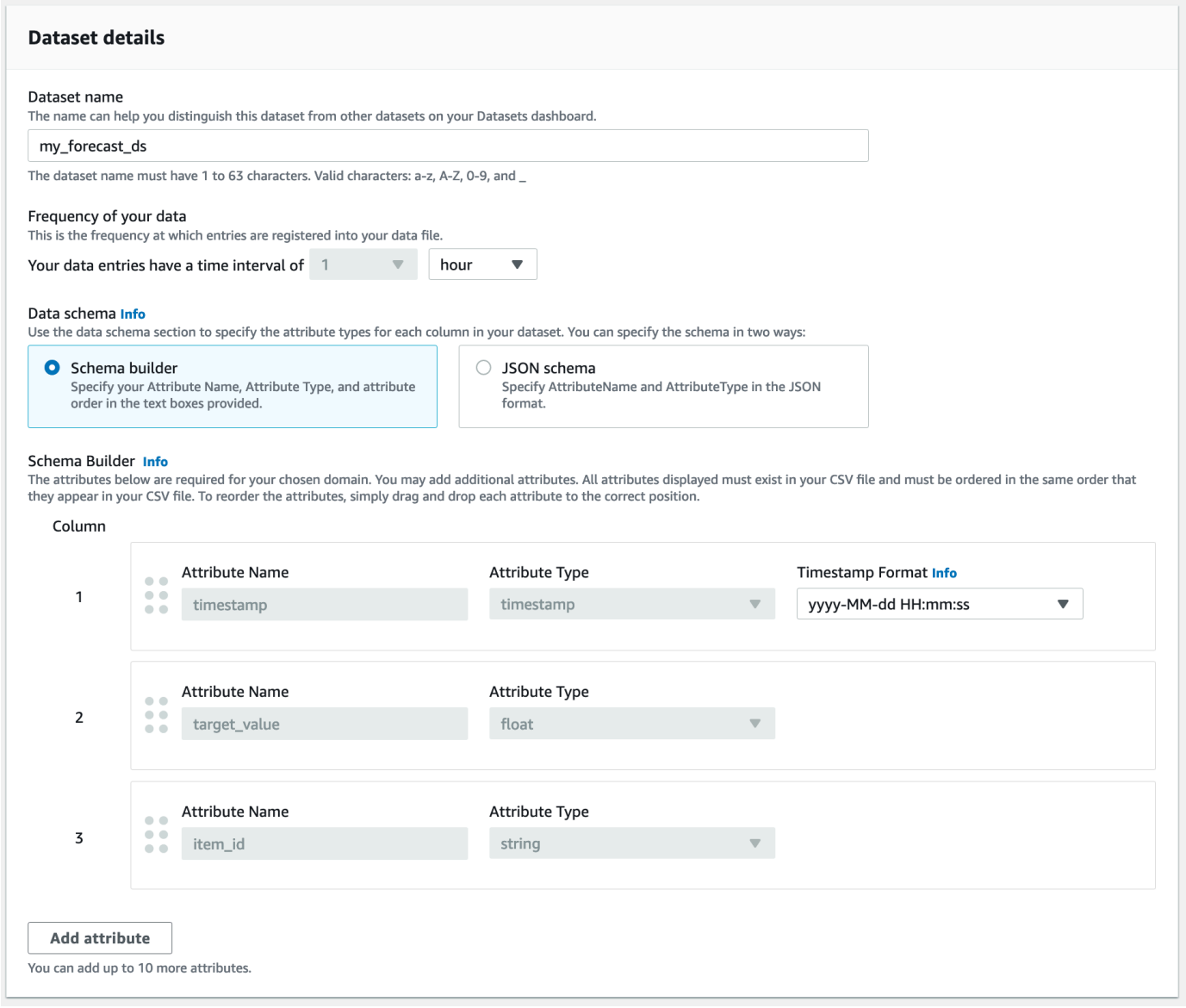
   - **Dataset name** – Enter a name for your dataset.
   - **Frequency of your data** – Keep the default value of **1**, and choose **hour** from the drop-down menu. This setting must be consistent with the input time series data. The time interval in the sample electricity-usage data is an hour.
   - **Data schema** – Choose **Schema builder** and drag the column components to match the time-series data order from top to bottom.

     1. timestamp - Use the default **Timestamp Format** of **`yyyy-MM-dd HH:mm:ss`**.

2. target_value
3. item_id

For the electricity usage input data, the columns correspond to: a timestamp, the electricity usage at the specified time (target_value), and the ID of the customer charged for the electricity usage (string). The order of the columns and the timestamp format specified here must be consistent with the input time series data.

The **Dataset details** panel should look similar to the following:



6. For **Dataset import details**, provide the following information:

- **Dataset import name** – Enter a name for your dataset.
- **Select time zone** – Leave the default selected (**Do not use time zone**).
- **Data location** – Use the following format to enter the location of your .csv file on Amazon S3:

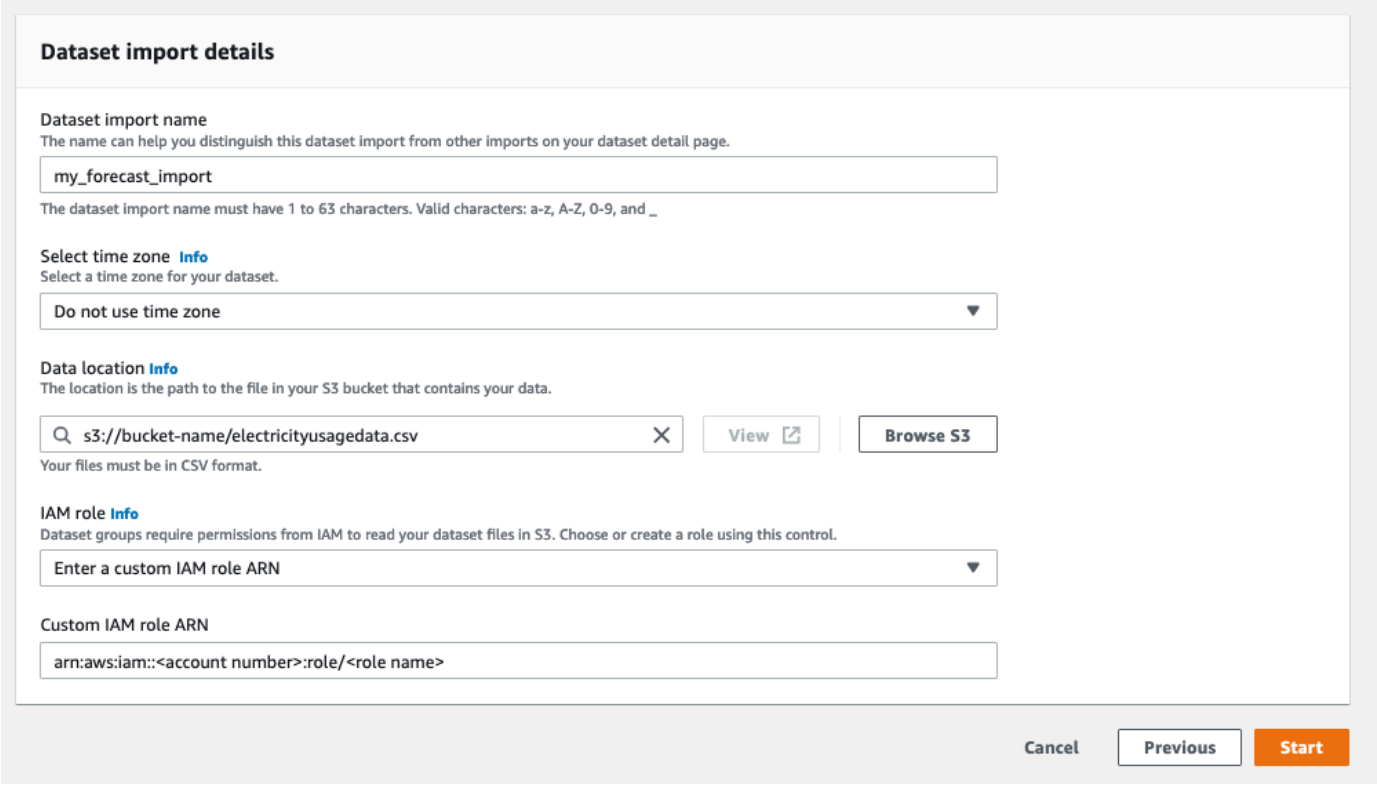
  ```
  s3://<name of your S3 bucket>/<folder path>/<filename.csv>
  ```

- **IAM role** – Keep the default **Enter a custom IAM role ARN**.

  Alternatively, you can have Amazon Forecast create the required IAM role for you by choosing **Create a new role** from the drop-down menu and following the on-screen instructions.

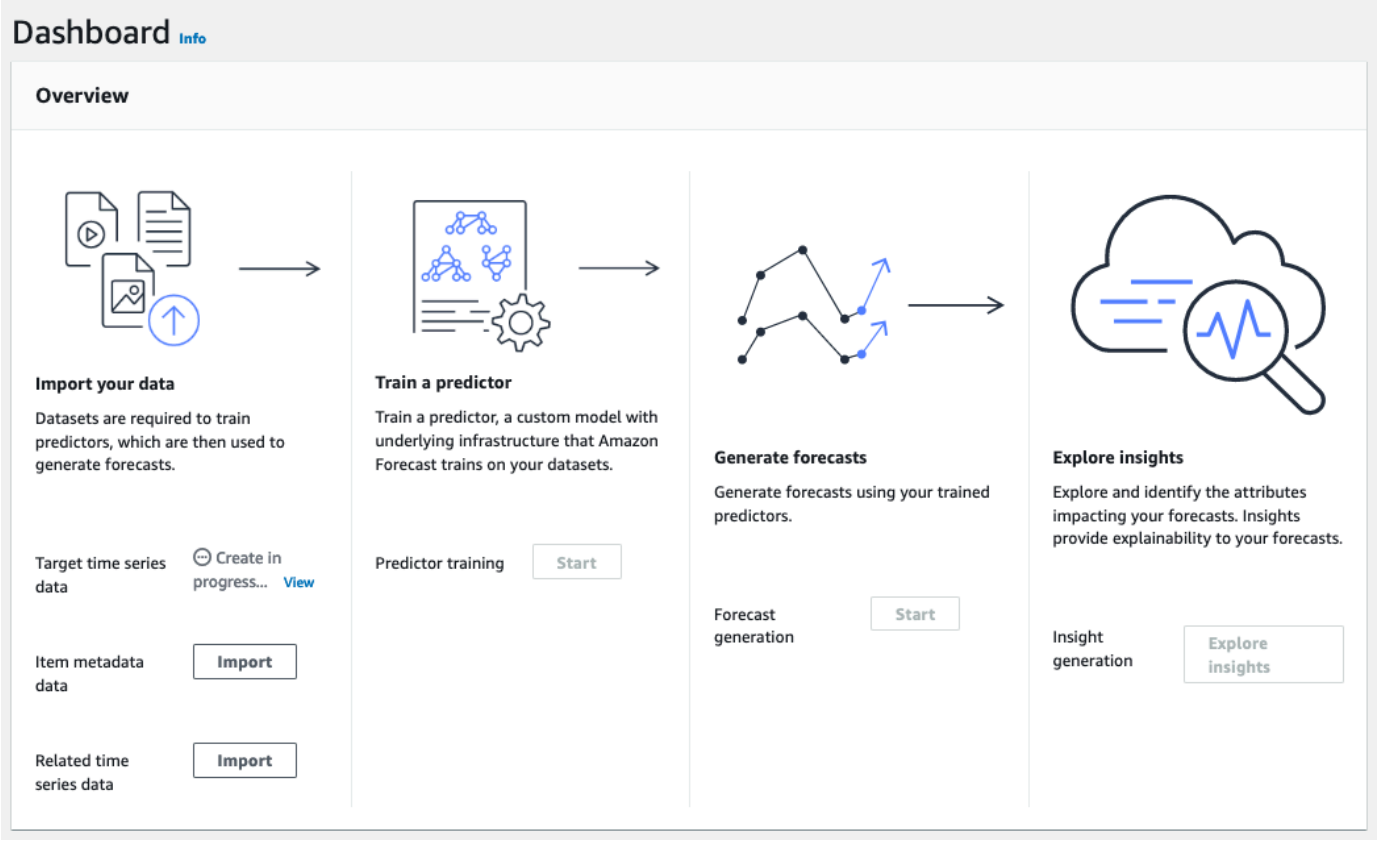
- **Custom IAM role ARN** – Enter the Amazon Resource Name (ARN) of the IAM role that you created in Create an IAM Role for Amazon Forecast (IAM Console).

The **Dataset import details** panel should look similar to the following:



7. Choose **Start**. If you are returned to the Amazon Forecast home page, choose **View dataset group**.

8. Click the name of the dataset group that you just created. The dataset group's **Dashboard** page is displayed. Your screen should look similar to the following:



Next to **Target time series data**, you will see the status of the import job. Wait for Amazon Forecast to finish importing your time series data. The process can take several minutes or longer. When your dataset has been imported, the status transitions to **Active** and the banner at the top of the dashboard notifies you that you have successfully imported your data.

Now that your target time series dataset has been imported, you can create a predictor.

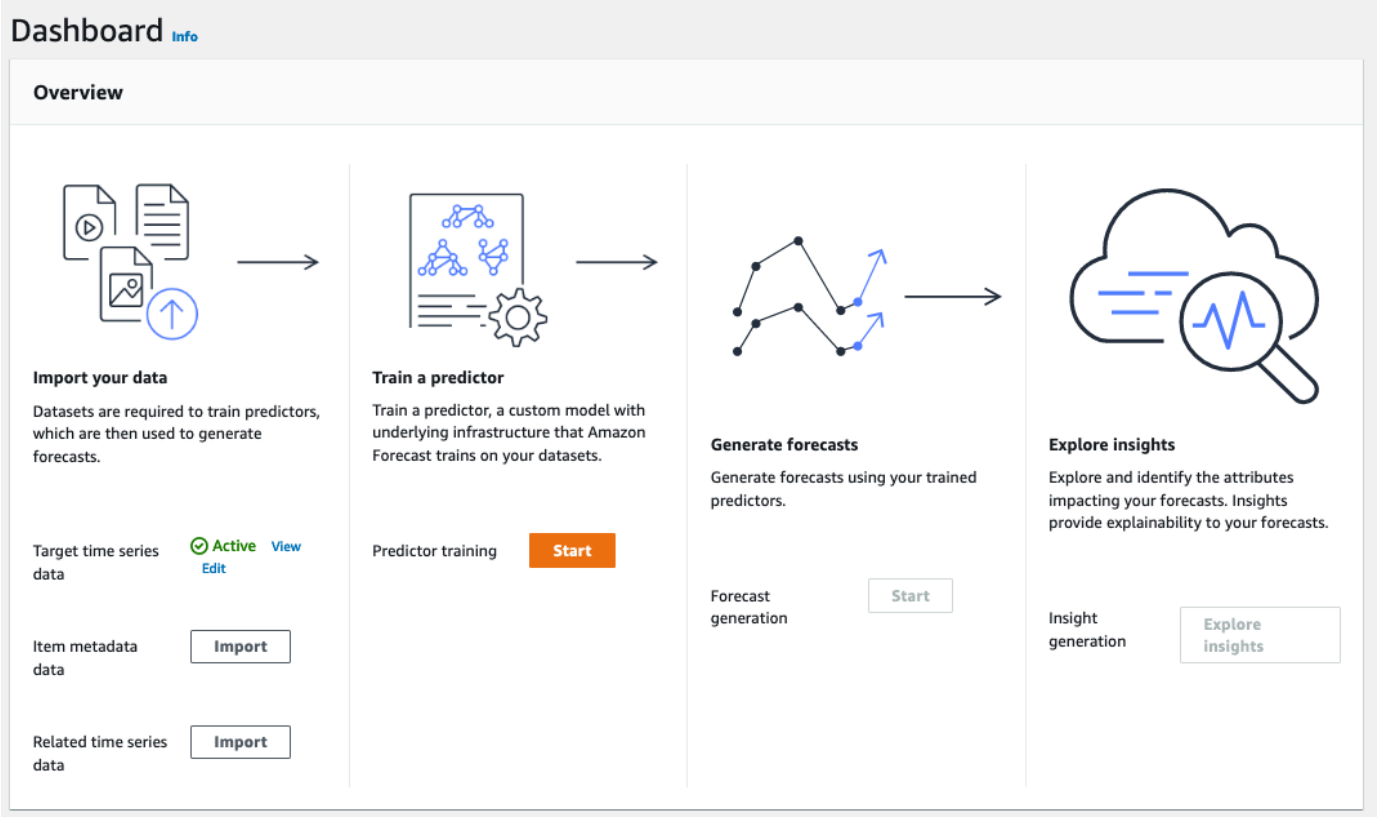
Step 2: Create a predictor

Next you create a predictor, which you use to generate forecasts based on your time series data. Forecast applies the optimal combination of algorithms to each time series in your datasets

To create a predictor with the Forecast console, you specify a predictor name, a forecast frequency, and define a forecast horizon. For more information about the additional fields you can configure, see Training Predictors.

To create a predictor

1. After your target time series dataset has finished importing, your dataset group's **Dashboard** should look similar to the following:



   Under **Train a predictor**, choose **Start**. The **Train predictor** page is displayed.

> **Note**
>
> The `Status` of the **Target time series data** must be `Active`, which signifies that the import successfully finished, before you can train the predictor.

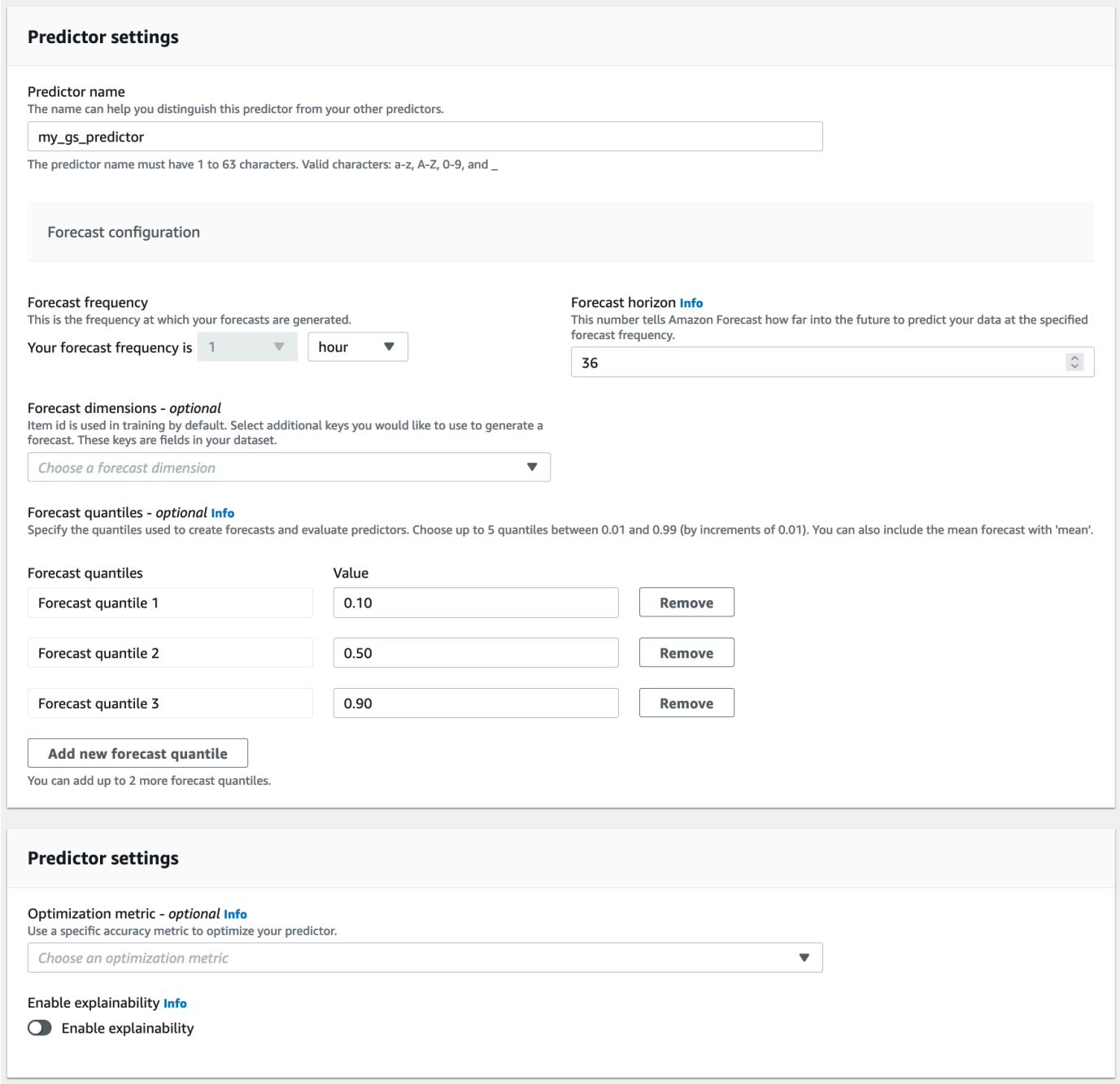
2. On the **Train predictor** page, for **Predictor settings**, provide the following information:

   - **Predictor name** – Enter a name for your predictor.
   - **Forecast frequency** – Keep the default value of **1**. From the drop-down menu, choose **hour**. This setting must be consistent with the input time series data. The time interval in the sample electricity-usage data is an hour.
   - **Forecast horizon** – Choose how far into the future to make predictions. This number multiplied by the data entry frequency (`hourly`) that you specified in `Step 1: Import`

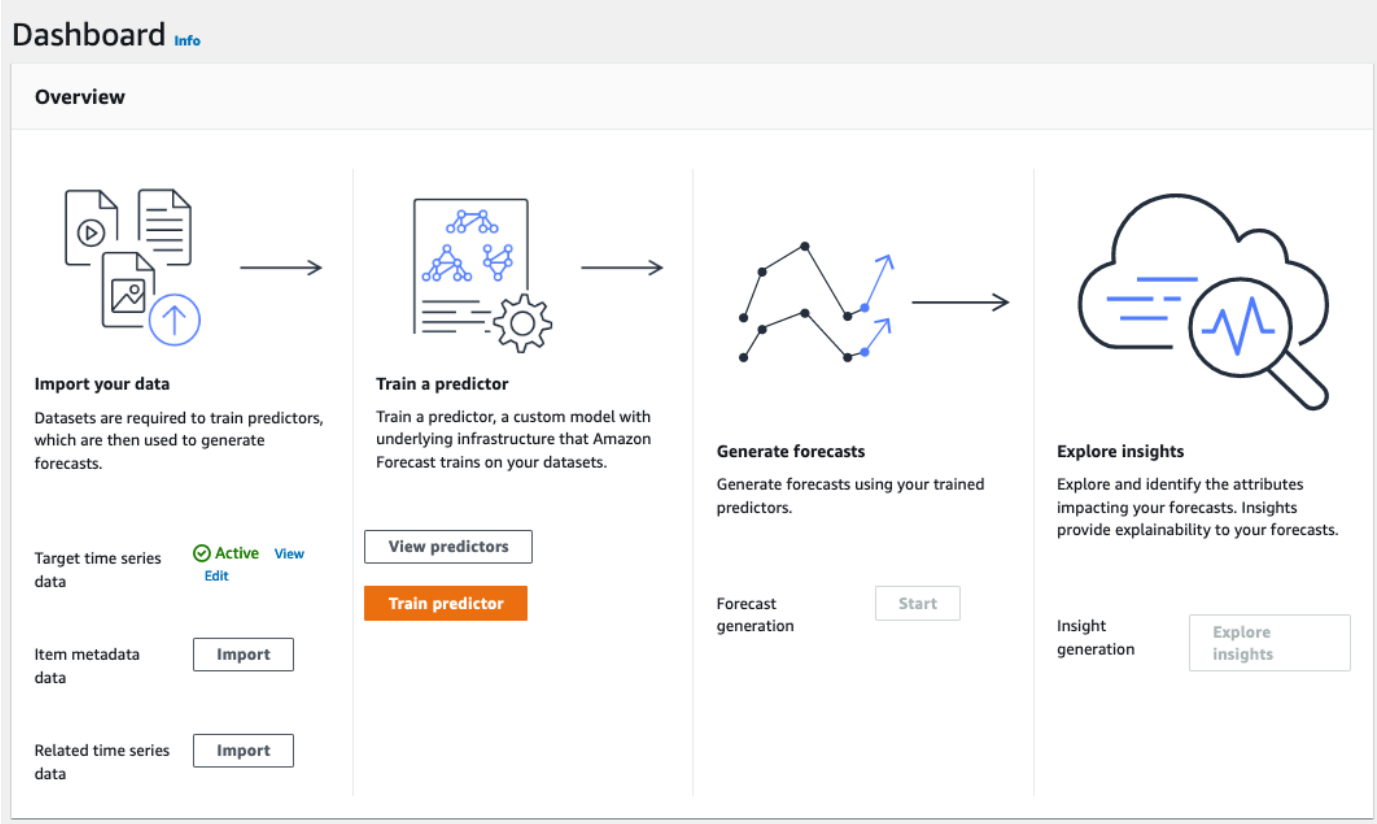
`the Training Data` determines how far into the future to make predictions. For this exercise, set the number to 36, to provide predictions for 36 hours.

- **Forecast dimensions** and **Forecast quantiles** – Leave the default values for these fields.

The remaining **Input data configuration** and **Tags** sections are optional, so leave the default values. The **Predictor settings** sections should look similar to the following:

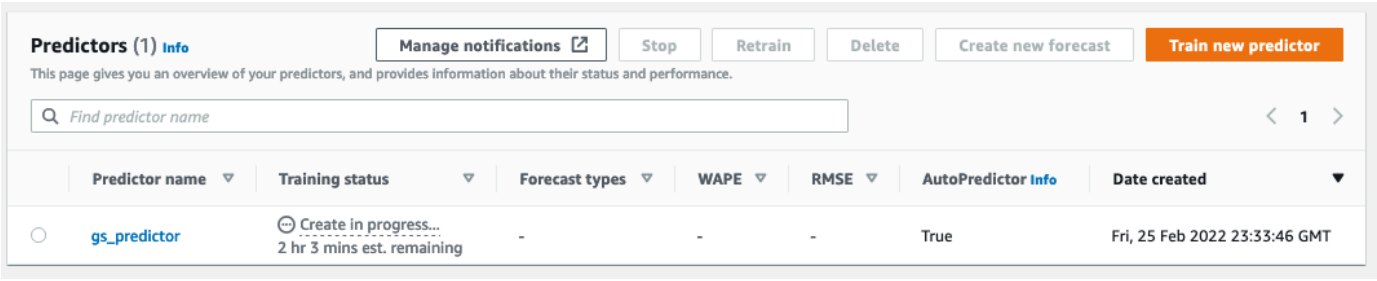


3. Choose **Create**. Your dataset group's **Dashboard** page is displayed. Your screen should look similar to the following:



4. To find the status of your predictor, choose **View predictors**.

5. On the **Predictors** page find the status of your predictor in the **Training status** column. Your screen should look similar to the following:



Wait for Amazon Forecast to finish training the predictor. The process can take several minutes or longer. When your predictor has been trained, the status transitions to **Active** and a banner displays notifying you that you can start generating forecasts.

Step 3: Create a Forecast

After your predictor is Active, you can create a forecast. A forecast is a group of predictions, one for every item in the target dataset. To retrieve the complete forecast, you create an export job.

To get and view your forecast

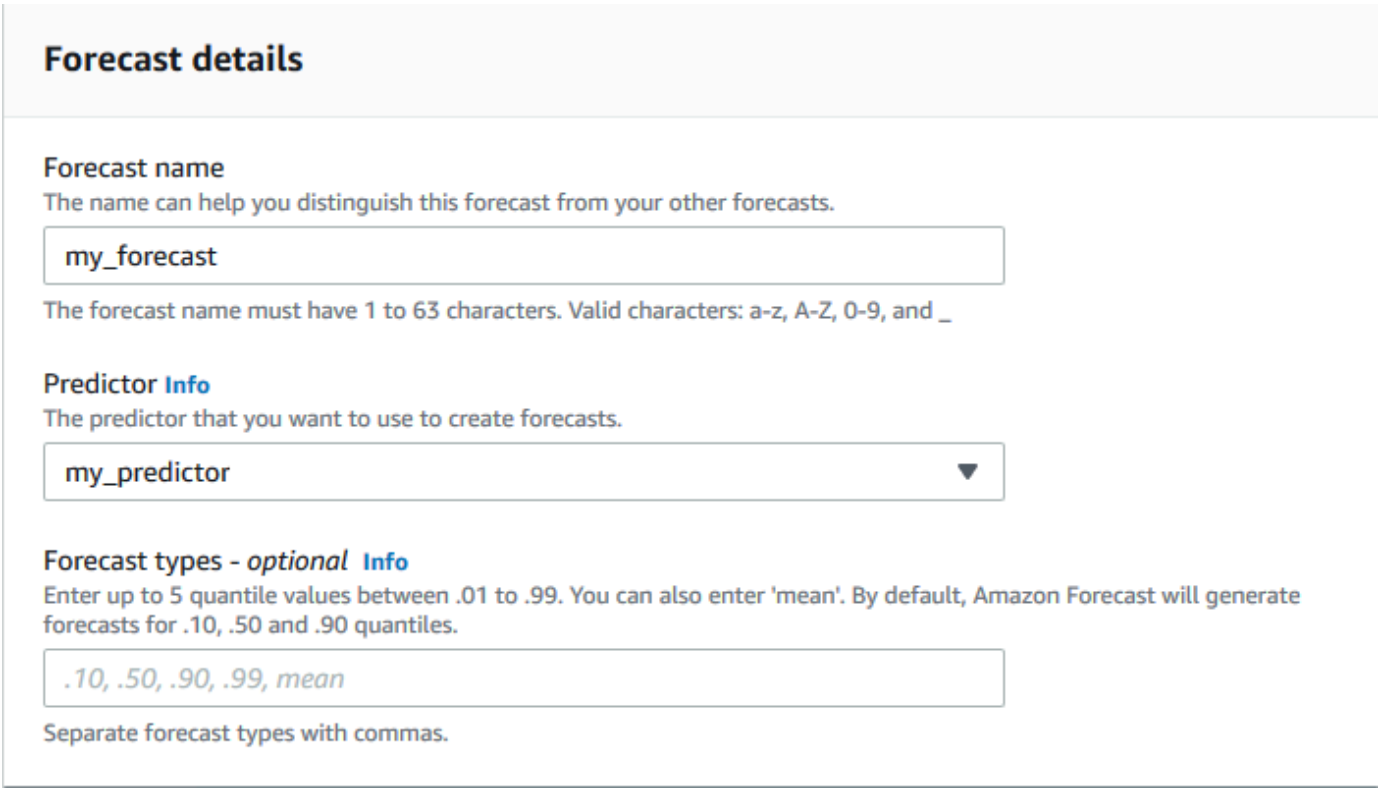
1. On your dataset group's **Dashboard**, under **Forecast generation**, choose **Start**. The **Create a forecast** page is displayed.

   > **Note**
   >
   > The `Status` of **Predictor training** must be `Active` before you can generate a forecast.

2. On the **Create a forecast** page, for **Forecast details**, provide the following information:

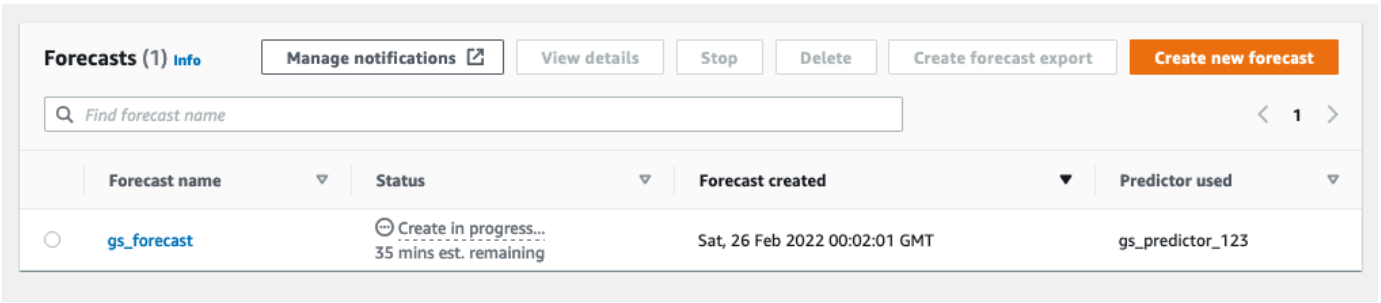
   - **Forecast name** – Enter a name for your forecast.
   - **Predictor** – From the drop-down menu, choose the predictor that you created in `Step 2: Train a Predictor`.

   The **Forecast quantiles** and **Tags** fields are optional, so leave the default value. Your screen should look similar to the following:



Click **Start**.

3. The **Forecasts** page is displayed. Your screen should look similar to the following:



The **Status** column lists the status of your forecast.. Wait for Amazon Forecast to finish creating the forecast. The process can take several minutes or longer. When your forecast has been created, the status transitions to **Active**.

Now that your forecast has been created, you can export the forecast.

Step 4: Exporting a Forecast

After the forecast has been created, you can export the complete forecast.

To export the complete forecast

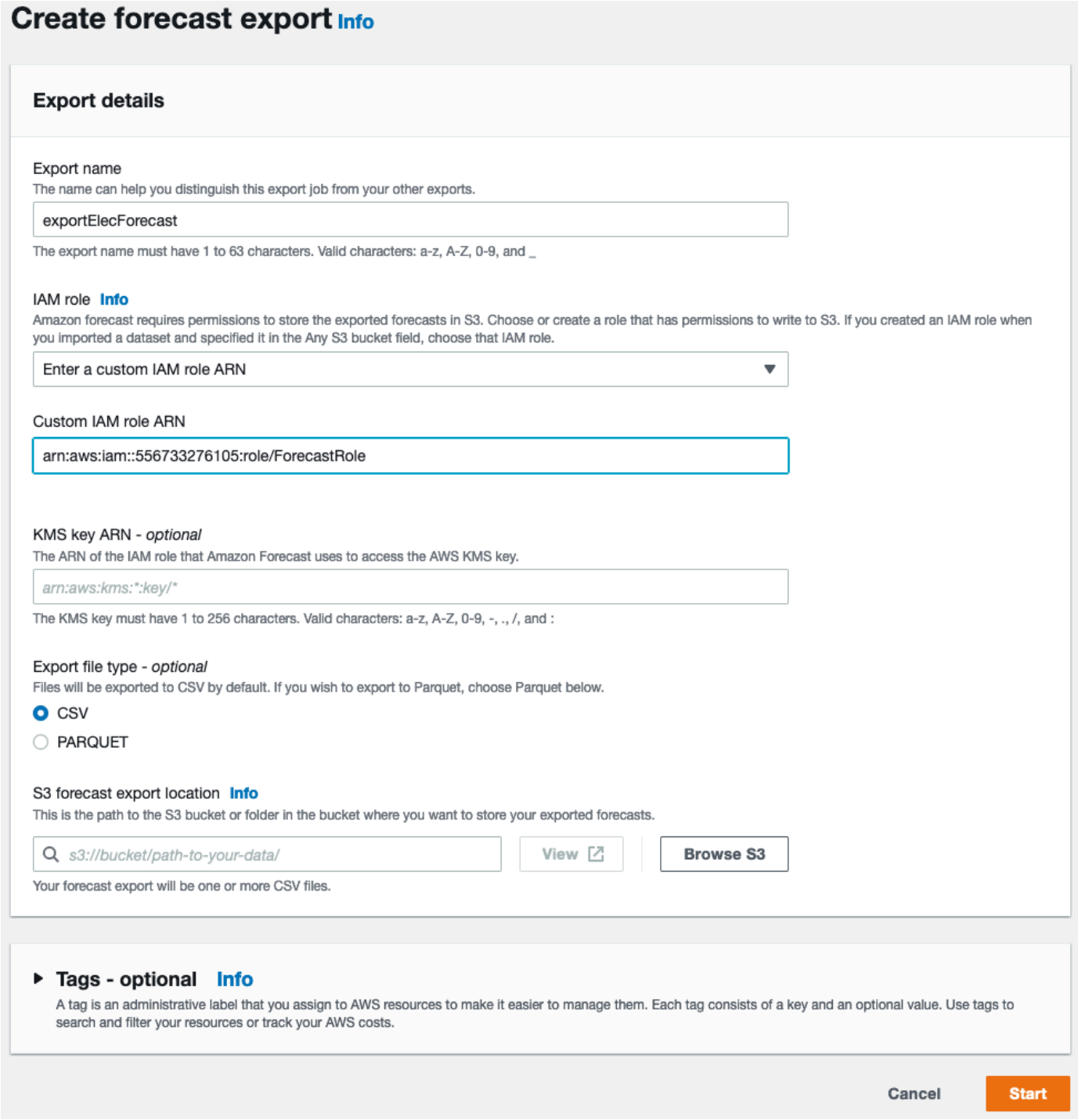
1. On the dataset groups page, click the dataset group that you created in `Step 1: Import Training Data`.
2. Click
   ☰
   in the upper left corner of the screen to open the navigation pane. Under your dataset group, click **Forecasts**.
3. Choose the radio button next to the forecast that you created in `Step 3: Create a Forecast`.
4. Choose **Create forecast export**. The **Create forecast export** page is displayed.
5. On the **Create forecast export** page, for **Export details**, provide the following information.

   - **Export name** – Enter a name for your forecast export job.
   - **IAM role** – Keep the default **Enter a custom IAM role ARN**.

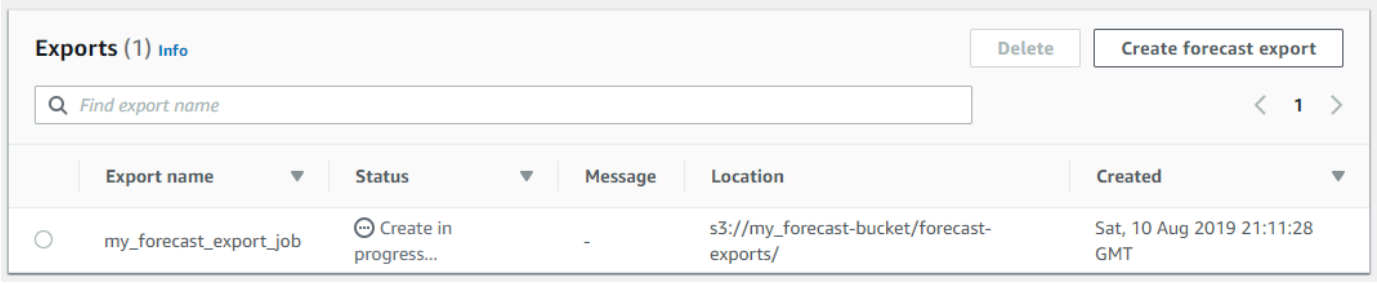
     Alternatively, you can have Amazon Forecast create the required IAM role for you by choosing **Create a new role** from the drop-down menu and following the on-screen instructions.
   - **Custom IAM role ARN** – Enter the Amazon Resource Name (ARN) of the IAM role that you created in Create an IAM Role for Amazon Forecast (IAM Console).
   - **S3 forecast export location** – Use the following format to enter the location of your Amazon Simple Storage Service (Amazon S3) bucket or folder in the bucket:

     `s3://<name of your S3 bucket>/<folder path>/`

   Your screen should look similar to the following:



6. Click **Start**. The **Forecasts** page is displayed.
7. Click the forecast that you created in `Step 3: Create a Forecast`. Find the **Exports** section. Your screen should look similar to the following:



You should see the status progress. Wait for Amazon Forecast to finish exporting the forecast. The process can take several minutes or longer. When your forecast has been exported, the status transitions to **Active** and you can find the forecast files in your S3 bucket.

Getting Started (AWS CLI)

In this exercise, you use the AWS Command Line Interface (AWS CLI) to explore Amazon Forecast. You create an Amazon Forecast dataset, train an predictor, and use the resulting predictor to generate a forecast. Before you begin, make sure that you have an AWS account and that you've set up the AWS CLI. For more information, see Setting Up.

> **Note**
>
> The AWS CLI commands in this exercise were tested on Linux. For information about using the AWS CLI commands on Windows, see Specifying Parameter Values for the AWS Command Line Interface in the *AWS Command Line Interface User Guide*.

Step 1: Import Training Data

Begin by creating a dataset and importing the electricity usage data into it.

To create an Amazon Forecast dataset

1. Decide which domain and dataset type is appropriate.

   The training data that you will import into the dataset influences your choice of dataset domain and type. So, let's review a few sample rows of the electricity usage data.

```
2014-01-01 01:00:00,   2.53807106598985, client_0
2014-01-01 01:00:00, 23.648648648648624, client_1
```

```
2014-01-01 02:00:00,  9.648648648612345, client_0
```

The data format is CSV (comma-separated values), and it's collected hourly (as shown by the timestamps). It includes these columns:

- Column 1 – Timestamps that show when electricity usage was recorded.
- Column 2 – Hourly electricity usage values (note how the timestamp values increase by hour).
- Column 3 – Client ID values that identify the customers using the electricity.

For this data, choose the following predefined dataset domain and dataset type:

- Custom domain – None of the dataset domains, such as METRICS, RETAIL, or WEB_TRAFFIC, applies to this data, so choose the Custom domain.
- Target time series type – The data is a time series because it tracks electricity usage over time. It also includes the *target* that we want to forecast (Column 2, electricity usage). Therefore, choose the target time series dataset type.

  To understand why you choose this type, see Predefined Dataset Domains and Dataset Types.

2. Decide on a dataset schema.

The target time series type for the CUSTOM Domain requires these fields; `timestamp`, `target_value`, and `item_id`. The `target_value` field is the target. Amazon Forecast generates the forecast for this field.

To map the required fields to columns in your data, you create a schema. Each *attribute* in the schema maps to a field in the data.

> **Important**
>
> The order of attributes in the schema must match the order of fields in the training data.

```
{
  "Attributes":[
```

```
    {
       "AttributeName": "timestamp",
       "AttributeType": "timestamp"
    },
    {
       "AttributeName": "target_value",
       "AttributeType": "float"
    },
    {
       "AttributeName": "item_id",
       "AttributeType": "string"
    }
  ]
}
```

You now have the information necessary to create a dataset and import data into it.

3. Create the dataset.

```
aws forecast create-dataset \
--dataset-name electricity_demand_ds \
--domain CUSTOM \
--dataset-type TARGET_TIME_SERIES \
--data-frequency H \
--schema '{
  "Attributes": [
    {
      "AttributeName": "timestamp",
      "AttributeType": "timestamp"
    },
    {
      "AttributeName": "target_value",
      "AttributeType": "float"
    },
    {
      "AttributeName": "item_id",
      "AttributeType": "string"
    }
  ]
}'
```

In the request, the `data-frequency` value `H` represents a data collection frequency of hourly. The following is an example response.

```
{
    "DatasetArn": "arn:aws:forecast:us-west-2:acct-id:dataset/
electricity_demand_ds"
}
```

For more information about this operation, see CreateDataset.

4. (Optional) Get the description of the dataset.

```
aws forecast describe-dataset \
--dataset-arn arn:aws:forecast:us-west-2:acct-id:dataset/electricity_demand_ds
```

The following is an example response.

```
{
    "DatasetName": "electricity_demand_ds",
    "DatasetArn": "arn:aws:forecast:us-west-2:acct-id:dataset/
electricity_demand_ds",
    "CreationTime": 1564533087.907,
    "LastModificationTime": 1564533087.907,
    "Domain": "CUSTOM",
    "DatasetType": "TARGET_TIME_SERIES",
    "DataFrequency": "H",
    "Schema": { ... },
    "EncryptionConfig": {},
    "Status": "ACTIVE"
}
```

> **Note**
>
> The order of the key-value pairs in the response is arbitrary.

5. Create a dataset group and add the dataset to it. The value of the `domain` parameter must match the `domain` of the dataset.

```
aws forecast create-dataset-group \
--dataset-group-name electricity_ds_group \
--dataset-arns arn:aws:forecast:us-west-2:acct-id:dataset/electricity_demand_ds \
--domain CUSTOM
```

The following is an example response.

```
{
    "DatasetGroupArn": "arn:aws:forecast:us-west-2:acct-id:dataset-group/
electricity_ds_group"
}
```

For more information about this operation, see CreateDatasetGroup.

6. (Optional) Get the description of the dataset group.

```
aws forecast describe-dataset-group \
--dataset-group-arn arn:aws:forecast:us-west-2:acct-id:dataset-group/
electricity_ds_group
```

The following is an example response.

```
{
    "DatasetGroupName": "electricity_ds_group",
    "DatasetGroupArn": "arn:aws:forecast:us-west-2:acct-id:dataset-group/
electricity_ds_group",
    "DatasetArns": [
        "arn:aws:forecast:us-west-2:acct-id:dataset-group/electricity_ds_group"
    ],
    "Domain": "CUSTOM",
    "CreationTime": 1564533719.852,
    "LastModificationTime": 1564533719.852,
    "Status": "ACTIVE"
}
```

7. Import the electricity usage training data from your Amazon S3 bucket to the dataset. The IAM role that you provide must have permission to read data from your S3 bucket. For information on how to create an IAM role, see Create an IAM Role for Amazon Forecast (AWS CLI).

```
aws forecast create-dataset-import-job \
--dataset-arn arn:aws:forecast:us-west-2:acct-id:dataset/electricity_demand_ds \
--dataset-import-job-name electricity_ds_import_job \
--data-source '{
    "S3Config": {
      "Path": "s3://bucket/electricityusagedata.csv",
      "RoleArn": "arn:aws:iam::acct-id:role/Role"
```

```
    }
  }'
```

The following is the shorthand syntax for the `data-source` parameter.

```
--data-source S3Config="{Path='s3://bucket/
electricityusagedata.csv',RoleArn='arn:aws:iam::acct-id:role/Role'}"
```

The following is an example response.

```
{
    "DatasetImportJobArn": "arn:aws:forecast:us-west-2:acct-id:dataset-import-job/
electricity_demand_ds/electricity_ds_import_job"
}
```

For more information about this operation, see CreateDatasetImportJob.

8. Check the import status.

```
aws forecast describe-dataset-import-job \
--dataset-import-job-arn arn:aws:forecast:us-west-2:acct-id:dataset-import-job/
electricity_demand_ds/electricity_ds_import_job
```

The following is an example response.

```
{
    "DatasetImportJobName": "electricity_ds_import_job",
    "DatasetImportJobArn": "arn:aws:forecast:us-west-2:acct-id:dataset-import-job/
electricity_demand_ds/electricity_ds_import_job",
    "DatasetArn": "arn:aws:forecast:us-west-2:acct-id:dataset/
electricity_demand_ds",
    "DataSource": {
        "S3Config": {
            "Path": "s3://bucket/electricityusagedata.csv",
            "RoleArn": "arn:aws:iam::acct-id:role/ForecastRole"
        }
    },
    "DataSize": 0.14639010466635227,
    "TimeStampFormat": "yyyy-MM-dd HH:mm:ss",
    "CreationTime":  1564537011.114,
    "LastModificationTime": 1564537028.223,
```

```
    "Status": "CREATE_IN_PROGRESS"
}
```

When all of the data has been imported, the status changes to ACTIVE and the response includes statistics for the data, as shown in the following example.

```
{
    "DatasetArn": "arn:aws:forecast:us-west-2:acct-id:dataset/
electricity_demand_ds",
    "Status": "ACTIVE",
    "FieldStatistics": {
        "date": {
            "Min": "2014-01-01T01:00:00Z",
            "Max": "2015-01-01T00:00:00Z",
            "Count": 3241200,
            "CountDistinct": 8760,
            "CountNull": 0
        },
        "target": {
            "Min": "0.0",
            "Max": "168200.0",
            "Avg": 606.5167610461679,
            "Stddev": 3518.405223972031,
            "Count": 3241200,
            "CountDistinct": 1196961,
            "CountNull": 0,
            "CountNan": 0
        },
        "item": {
            "Count": 3241200,
            "CountDistinct": 370,
            "CountNull": 0
        }
    },
    ...
}
```

Important

You must wait until the status is ACTIVE before creating a predictor with the dataset group.

For more information about this operation, see DescribeDatasetImportJob.

Step 2: Create a Predictor

To create a predictor, you use the CreateAutoPredictor operation and provide the following information.

- **Predictor name** – Give the predictor a name so you can distinguish it from your other predictors
- **Dataset group** – You created the dataset group in the preceding step.
- **Forecast frequency** – The granularity of your forecasts (hourly, daily, weekly, etc).
- **Forecast horizon** – The number of time steps being forecasted.

After the predictor is created, you review the accuracy metrics generated by Amazon Forecast. The metrics help you decide whether to use the predictor for generating a forecast. For more information about predictors, see Training Predictors.

To create a predictor and review the accuracy metrics

1. Create the predictor.

```
aws forecast create-predictor \
--predictor-name electricitypredictor \
--input-data-config DatasetGroupArn="arn:aws:forecast:us-west-2:acct-id:dsgroup/
electricity_ds_group" \
--forecast-horizon 36 \
--forecast-frequency D
```

   The following is an example response.

```
{
    "PredictorArn": "arn:aws:forecast:us-west-2:acct-id:predictor/
electricitypredictor"
}
```

2. Get the predictor's status.

```
aws forecast describe-predictor \
```

```
--predictor-arn arn:aws:forecast:us-west-2:acct-id:predictor/electricitypredictor
```

The following is an example response.

```
{
    "PredictorArn": "arn:aws:forecast:<region>:<acct-num>:predictor/
electricitypredictor",
    "PredictorName": "electricitypredictor",
    "ForecastHorizon": 36,
    "ForecastTypes": [
        "0.1",
        "0.5",
        "0.9"
    ],
    "ForecastFrequency": "D",
    "DatasetImportJobArns": [
        "arn:aws:forecast:<region>:<acct-num>:dataset-import-job/
getting_started_dataset/gs_import"
    ],
    "DataConfig": {
        "DatasetGroupArn": "arn:aws:forecast:<region>:<acct-num>:dataset-group/
getting_started",
        "AttributeConfigs": [
            {
                "AttributeName": "target_value",
                "Transformations": {
                    "aggregation": "sum",
                    "backfill": "zero",
                    "frontfill": "none",
                    "middlefill": "zero"
                }
            }
        ]
    },
    "EstimatedTimeRemainingInMinutes": 97,
    "Status": "CREATE_IN_PROGRESS",
    "CreationTime": "2022-02-23T09:26:24.643000-08:00",
    "LastModificationTime": "2022-02-23T09:49:26.899000-08:00",
    "ExplainabilityInfo": {
        "Status": "NOT_AVAILABLE"
    }
}
```

> **Important**
>
> Model training takes time. Don't proceed until training has completed and the status of the predictor is ACTIVE.

3. Get the accuracy metrics for the predictor.

```
aws forecast get-accuracy-metrics \
--predictor-arn arn:aws:forecast:us-west-2:acct-id:predictor/electricitypredictor
```

The following is an example response.

```
{
    "PredictorEvaluationResults": [
        {
            "TestWindows": [
                {
                    "EvaluationType": "SUMMARY",
                    "Metrics": {
                        "RMSE": 448.19602551622864,
                        "WeightedQuantileLosses": [
                            {
                                "Quantile": 0.9,
                                "LossValue": 0.11574311406253326
                            },
                            {
                                "Quantile": 0.5,
                                "LossValue": 0.1706269067283527
                            },
                            {
                                "Quantile": 0.1,
                                "LossValue": 0.11724164222477837
                            }
                        ]
                    }
                },
                {
                    "EvaluationType": "COMPUTED",
                    "Metrics": {
                        "RMSE": 448.19602551622864,
                        "WeightedQuantileLosses": [
```

```
                            {
                                "Quantile": 0.9,
                                "LossValue": 0.11574311406253326
                            },
                            {
                                "Quantile": 0.5,
                                "LossValue": 0.1706269067283527
                            },
                            {
                                "Quantile": 0.1,
                                "LossValue": 0.11724164222477837
                            }
                        ]
                    },
                    "TestWindowEnd":   1420070400.0,
                    "TestWindowStart": 1420002000.0
                }
            ]
        }
    ]
}
```

The metrics show the error loss for each quantile. For example, there was an 11.7% error for the first quantile. The metrics also show the root-mean-square error (RMSE).

The summary metrics show the average of the computed metrics over all test windows. Because there was only one test window, the summary and computed metrics are equal.

For more information about this operation, see GetAccuracyMetrics.

Step 3: Create a Forecast

Amazon Forecast creates a forecast for the `target_value` field (as determined by the dataset domain and type) for each unique `item_id` in the dataset. In this exercise, the `target_value` field provides electricity usage and the `item_id` provides client IDs. You get a forecast for the hourly electricity usage by customer.

After the forecast has been created, you can query for a single item or export the complete forecast.

To create, retrieve, and export a forecast

1. Create the forecast.

```
aws forecast create-forecast \
--forecast-name electricityforecast \
--predictor-arn arn:aws:forecast:us-west-2:acct-id:predictor/electricitypredictor
```

The operation uses the predictor to create a forecast. In the response, you get the Amazon Resource Name (ARN) of the forecast. You use this ARN to retrieve and export the forecast. The following is an example response.

```
{
    "ForecastArn": "arn:aws:forecast:us-west-2:acct-id:forecast/
electricityforecast"
}
```

For more information about this operation, see CreateForecast.

2. Retrieve the first two hours of the forecast for `client_1`.

> **Note**
>
> The service name, `forecastquery`, is different then the service name used elsewhere.

```
aws forecastquery query-forecast \
--forecast-arn arn:aws:forecast:us-west-2:acct-id:forecast/electricityforecast \
--start-date 2015-01-01T00:00:00 \
--end-date   2015-01-01T02:00:00 \
--filters '{"item_id":"client_1"}'
```

The operation includes the following parameters.

- `start-date` and `end-date` – Specifies an optional date range to retrieve the forecast for. If you don't specify these parameters, the operation returns the entire forecast for `client_1`.
- `filters` – Specifies the `item_id` filter to retrieve the electricity forecast for `client_1`.

  The following is the shorthand syntax for the `filters` parameter.

```
--filters item_id="client_1"
```

The following is an example response.


```
{
    "Forecast": {
        "Predictions": {
            "mean": [
                {
                    "Timestamp": "2015-01-01T01:00:00",
                    "Value": 20.952411651611328
                },
                {
                    "Timestamp": "2015-01-01T02:00:00",
                    "Value": 19.11078453063965
                }
            ],
            "p90": [
                {
                    "Timestamp": "2015-01-01T01:00:00",
                    "Value": 24.524038314819336
                },
                {
                    "Timestamp": "2015-01-01T02:00:00",
                    "Value": 22.319091796875
                }
            ],
            "p50": [
                {
                    "Timestamp": "2015-01-01T01:00:00",
                    "Value": 20.7841739654541
                },
                {
                    "Timestamp": "2015-01-01T02:00:00",
                    "Value": 19.237524032592773
                }
            ],
            "p10": [
                {
                    "Timestamp": "2015-01-01T01:00:00",
                    "Value": 18.507278442382812
```

```
                },
                {
                    "Timestamp": "2015-01-01T02:00:00",
                    "Value": 16.15062141418457
                }
            ]
        }
    }
}
```

Because this is an hourly forecast, the response shows hourly forecast values. In the response, note the following:

- `mean` – For the specific date and time, the mean is the predicted mean electricity usage value for the customer.
- `p90`, `p50`, and `p10` – Specify the confidence level that the actual value will be below the listed value at the specified date and time. For example, at 2015-01-01T01:00:00, Amazon Forecast is 90% confident that the electric usage will be below 24.5. Amazon Forecast is 50% confident that usage will be below 20.8, and 10% confident that usage will be below 18.5.

For more information about this operation, see QueryForecast.

3. Export the complete forecast to your Amazon S3 bucket. The IAM role that you provide must have permission to write data to your S3 bucket. For information on how to create an IAM role, see Create an IAM Role for Amazon Forecast (AWS CLI).

   Create a forecast export job.

```
aws forecast create-forecast-export-job \
--forecast-export-job-name electricityforecast_exportjob \
--forecast-arn arn:aws:forecast:us-west-2:acct-id:forecast/electricityforecast \
--destination S3Config="{Path='s3://bucket',RoleArn='arn:aws:iam::acct-
id:role/Role'}"
```

   The following is an example response.

```
{
   "ForecastExportJobArn": "arn:aws:forecast::us-west-2:acct-id:forecast-
export/64bbc087"
```

```
}
```

For more information about this operation, see CreateForecastExportJob.

4. Get the status of the export job.

```
aws forecast describe-forecast-export-job \
--forecast-export-job-arn arn:aws:forecast:us-west-2:acct-id:forecast/
electricityforecast
```

The following is an example response.

```
{
    "ForecastExportJobArn": "arn:aws:forecast::us-west-2:acct-id:forecast-
export/64bbc087",
    "ForecastExportJobName": "electricityforecast_exportjob",
    "Status": "CREATE_IN_PROGRESS"
}
```

When the status is ACTIVE, you can find the forecast files in the specified S3 bucket.

Getting Started (Python Notebooks)

> **Note**
>
> For a complete list of tutorials using Python notebooks, see the Amazon Forecast Github Samples page.

To get started using Amazon Forecast APIs with Python notebooks, see the Getting Started Tutorial. The tutorial guides you through the core steps of Forecast from start to finish.

For basic tutorials for specific processes, refer to the following Python notebooks:

1. Preparing data - Prepare a dataset, create a dataset group, define the schema, and import the dataset group.
2. Building your predictor - Train a predictor on the data you imported into your Forecast dataset.
3. Evaluating predictors - Obtain predictions, visualize predictions, and compare results.

4. Retraining predictors - Retrain an existing predictor with updated data.
5. Upgrade to AutoPredictor - Upgrade legacy predictors to AutoPredictor.
6. Clean Up - Delete the dataset groups, predictors, forecasts created during the tutorials.

To repeat the Getting Started tutorial with AutoML, see Getting Started with AutoML.

Advanced Tutorials

For more advanced tutorials, refer to the following Python notebooks:

- Item-level Explainability - Understand how dataset attributes impact forecasts for specific time series and time points.
- Comparing multiple models - Create predictors using Prophet, ETS, and DeepAR+, and compare their performances by visualizing the results.
- Cold start forecasting - Use item metadata and the DeepAR+ algorithm to forecast for cold-start scenarios (when there is little to no historical data).
- Incorporating related time-series datasets - Use related time-series datasets to improve the accuracy of your model.
- Incorporating item metadata - Use item metadata to improve the accuracy of your model.
- Using the Weather Index - Use the Weather Index to incorporate historical and projected weather information when training your predictors.
- Performing What-if analysis - Explore different pricing scenarios and evaluate how it impacts demand.
- Evaluate item-level accuracy - Export backtest metrics and forecasts, and evaluate the item-level performance of your predictor.

Clean Up Resources

To avoid incurring unnecessary charges, delete the resources you created after you're done with the getting started exercise. To delete the resources, use either the Amazon Forecast console or the `Delete` APIs from the SDKs or the AWS Command Line Interface (AWS CLI). For example, use the DeleteDataset API to delete a dataset.

To delete a resource, its status must be `ACTIVE`, `CREATE_FAILED`, or `UPDATE_FAILED`. Check the status using the `Describe` APIs, for example, DescribeDataset.

Some resources must be deleted before others, as shown in the following table. This process can take some time.

To delete the training data you uploaded, `electricityusagedata.csv`, see How Do I Delete Objects from an S3 Bucket?.

Resource to Delete	Delete This First	Notes
`ForecastExportJob`		
`Forecast`		You can't delete a forecast while it is being exported. After a forecast is deleted, you can no longer query the forecast.
`Predictor`	All associated forecasts.	
`DatasetImportJob`		Can not be deleted.
`Dataset`		All `DatasetImportJob` s that target the dataset are also deleted. You can't delete a `Dataset` that is used by a predictor.
`DatasetSchema`	All datasets that reference the schema.	
`DatasetGroup`	All associated predictors All associated forecasts. All datasets in the dataset group.	You can't delete a `DatasetGroup` that contains a `Dataset` used by a predictor.

Tutorials

The following tutorial shows you how to perform common tasks in Amazon Forecast and provide ready-made solutions for common use cases. For a complete list of tutorials using Python notebooks, see the Amazon Forecast GitHub Samples page.

Tutorials

- Automating Forecast with CloudFormation - Use an AWS CloudFormation stack to automatically deploy datasets to an S3 bucket and trigger a Forecast pipeline.

Automating with AWS CloudFormation

In this tutorial, you use an AWS CloudFormation automation stack to launch an Amazon Forecast pipeline and generate forecasts using a demonstration dataset.

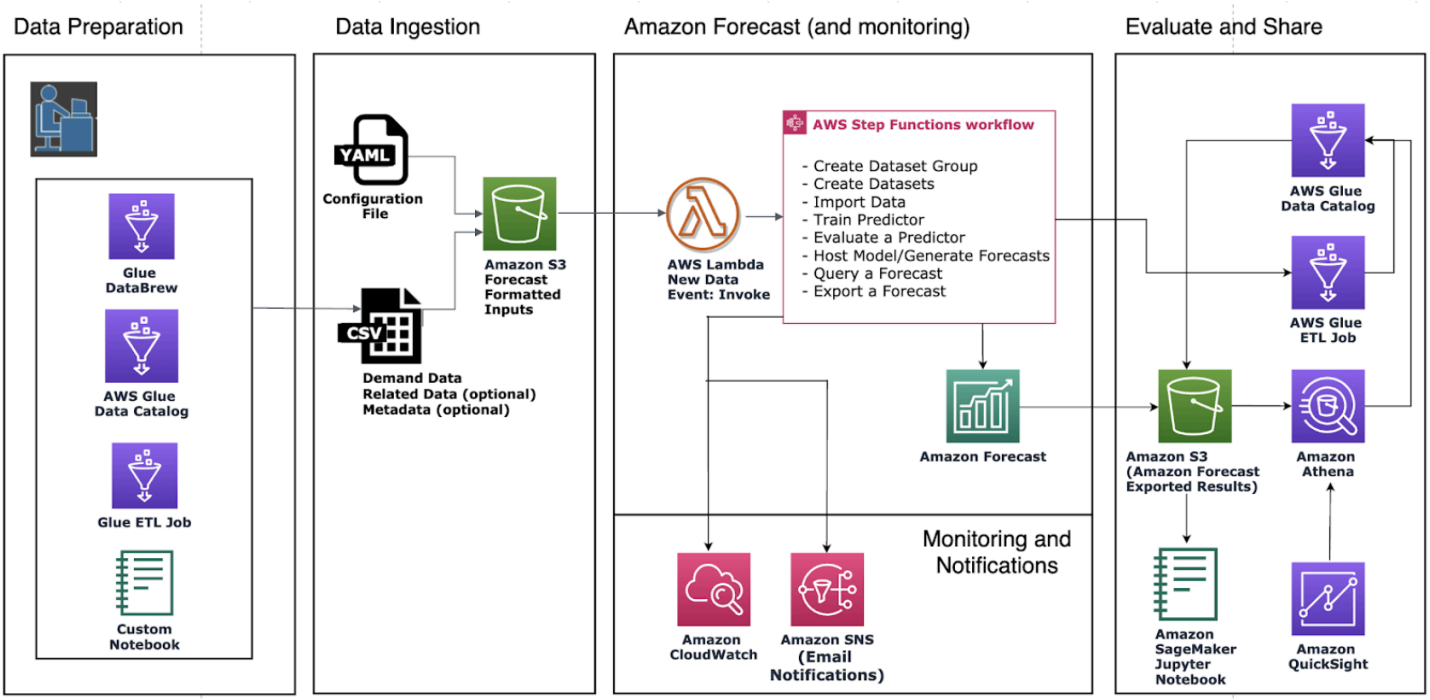
The AWS Forecast AWS CloudFormation stack:

- Deploys the Improving Forecast Accuracy with Machine Learning Solution AWS CloudFormation template.
- Deploys the NYC Taxi Datasets to the Forecast Data Amazon S3 bucket.
- Automatically starts the demo NYC taxi forecast pipeline in Forecast.

The AWS CloudFormation template is preloaded with target time-series, related time-series, and item metadata demonstration datasets. Relevant fields in the console are pre-filled with their respective S3 locations.

After completing this tutorial using the demonstration datasets, you can use the same automation stack to generate forecasts with your own datasets.

The following diagram shows the components used in this tutorial.



Prerequisites

Before starting the tutorial, make sure you have logged into your AWS account and installed the AWS CloudFormation template:

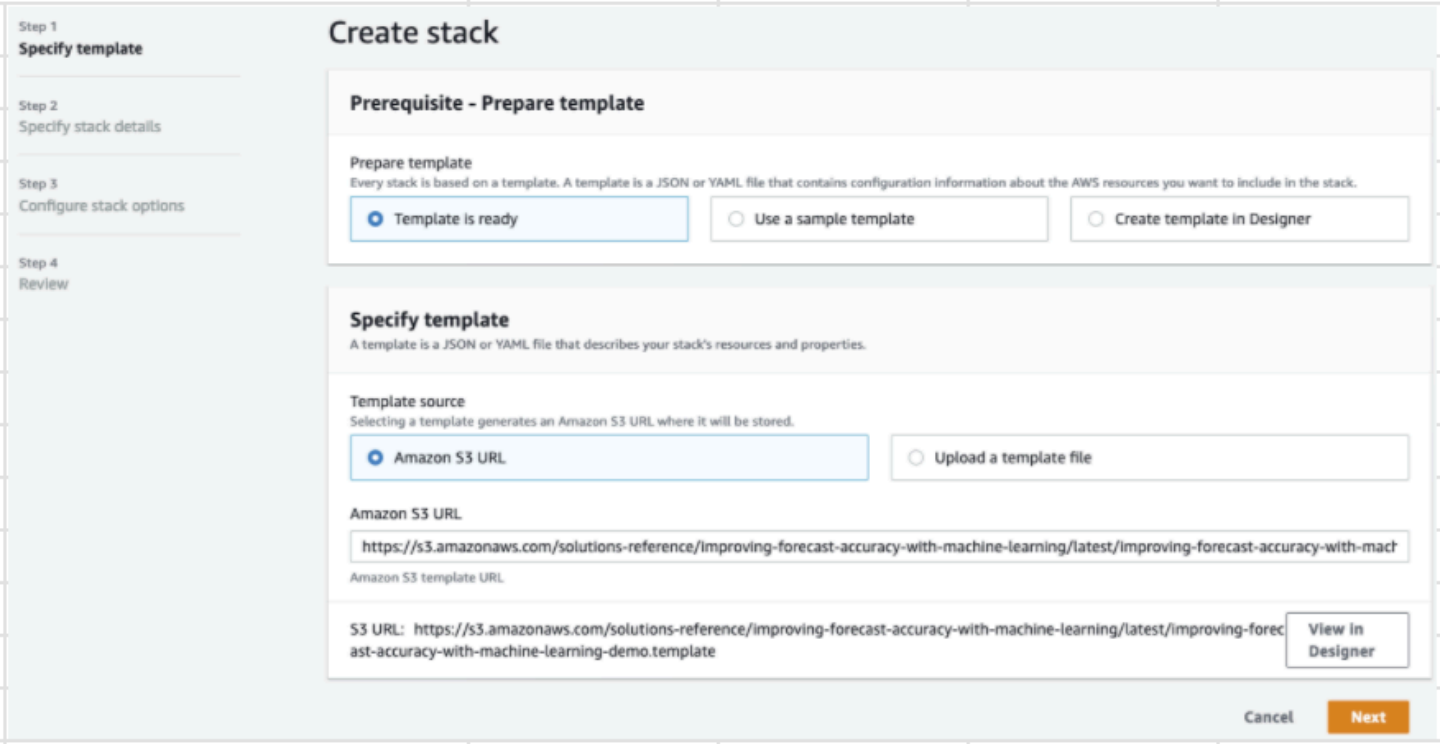
1. Log in to your AWS account. If you do not already have one, create an AWS account.
2. Install the AWS CloudFormation template. Choose the Region closest to you:
   - Tokyo: ap-northeast-1
   - Seoul: ap-northeast-2
   - Mumbai: ap-south-1
   - Singapore: ap-southeast-1
   - Sydney: ap-southeast-2
   - Frankfurt: eu-cental-1
   - Ireland: eu-west-1
   - N. Virginia: us-east-1
   - Ohio: us-east-2
   - Oregon: us-west-2

This deploys a demonstration stack using the NYC Taxi Dataset.

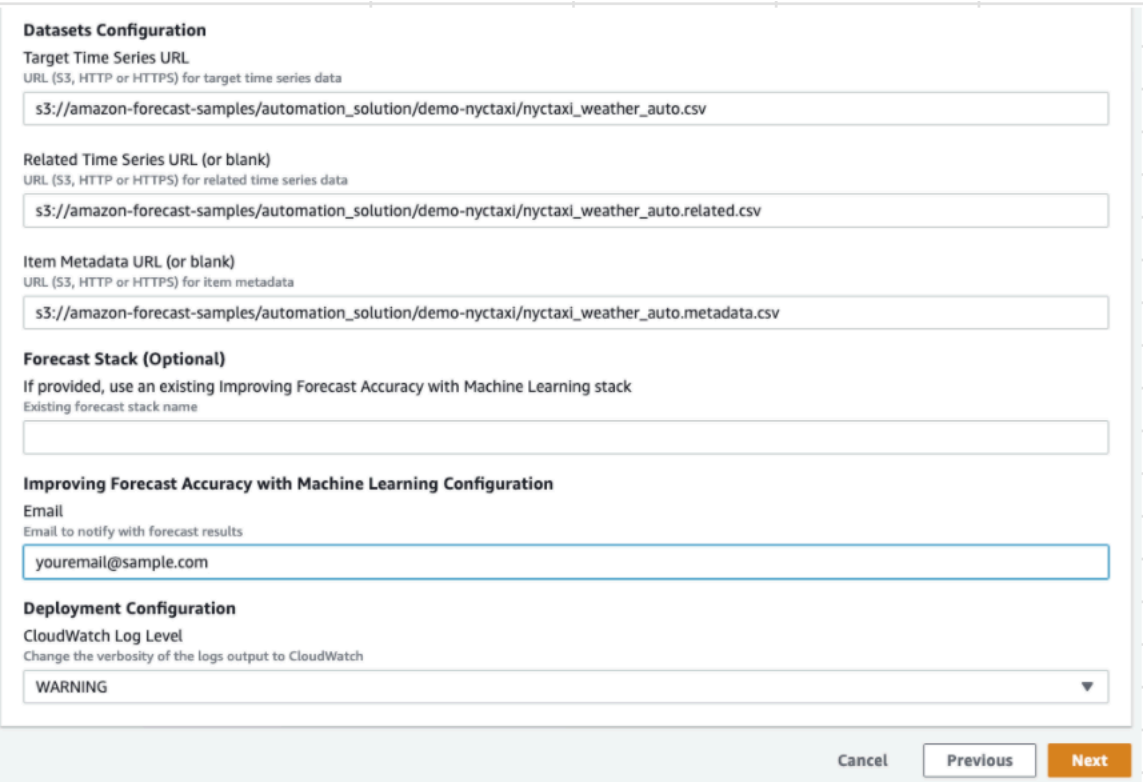
Deploying an AWS CloudFormation Template for Forecast automation

To deploy the CloudFormation template using the NYC Taxi Dataset

Step 1: Accept the defaults and choose **Next**.

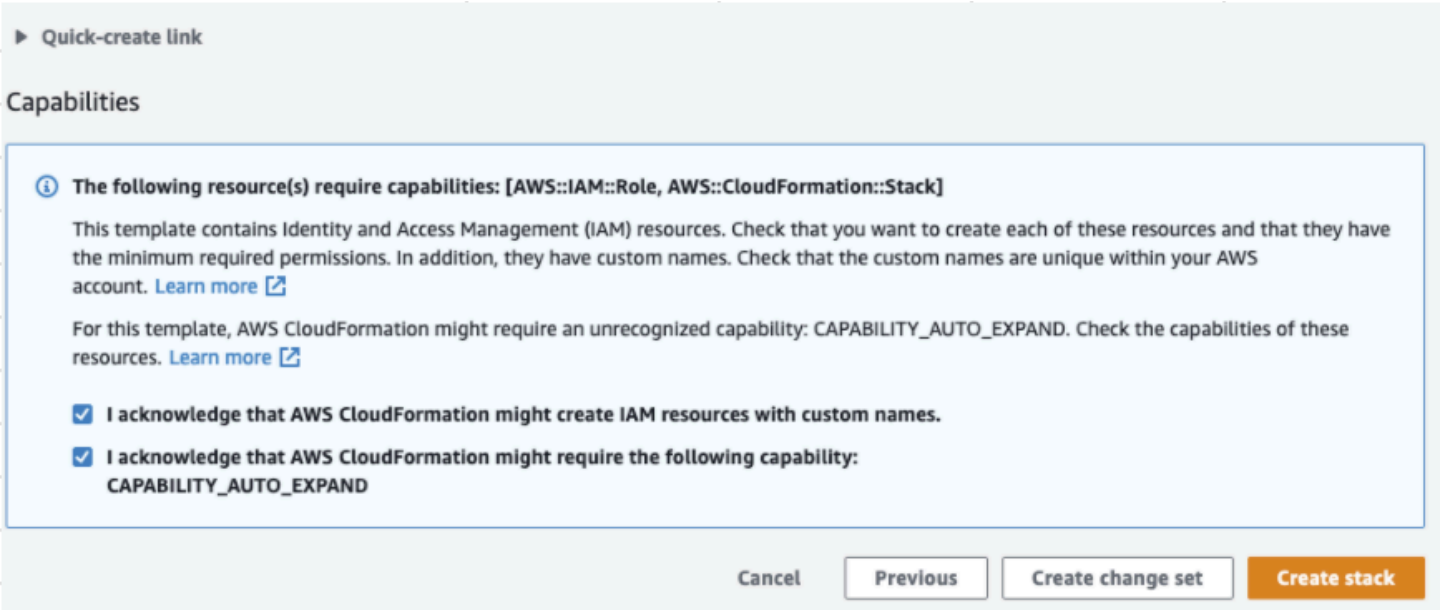


Step 2: Provide an email address for notifications and choose **Next**.



Step 3: Accept defaults and choose **Next**.

Step 4: For Capabilities, select both check boxes to allow AWS CloudFormation to create AWS Identity and Access Management (IAM) resources and nested stacks. Choose **Create stack**.



You have deployed an AWS CloudFormation template in Forecast.

Clean Up

After deploying this AWS CloudFormation template, you can clean up newly created resources, deploy the AWS CloudFormation stack using your own datasets, and explore other deployment options.

- **Cleaning up**: Deleting the demo stack retains the "Improving Forecast Accuracy with Machine Learning" stack. Deleting the "Improving Forecast Accuracy with Machine Learning" stack retains all S3, Athena, QuickSight, and Forecast data.
- **Using your own datasets**: To deploy this AWS CloudFormation template with your own time-series data, enter the S3 locations of your datasets in the Datasets Configuration section in **Step 2**.
- **Other deployment options**: For more deployment options, see Automated Deployment. If data is already available, you can deploy the stack without the demo data.

Importing Datasets

Datasets contain the data used to train a predictor. You create one or more Amazon Forecast datasets and import your training data into them. A *dataset group* is a collection of complementary datasets that detail a set of changing parameters over a series of time. After creating a dataset group, you use it to train a predictor.

Each dataset group can have up to three datasets, one of each dataset type: target time series, related time series, and item metadata.

To create and manage Forecast datasets and dataset groups, you can use the Forecast console, AWS Command Line Interface (AWS CLI), or AWS SDK.

For example Forecast datasets, see the Amazon Forecast Sample GitHub repository.

Topics

Datasets

To create and manage Forecast datasets, you can use the Forecast APIs, including the CreateDataset and DescribeDataset operations. For a complete list of Forecast APIs, see API Reference.

When creating a dataset, you provide information, such as the following:

- The frequency/interval at which you recorded your data. For example, you might aggregate and record retail item sales every week. In the Getting Started exercise, you use the average electricity used per hour.
- The prediction format (the *domain*) and dataset type (within the domain). A dataset domain specifies which type of forecast you'd like to perform, while a dataset type helps you organize your training data into Forecast-friendly categories.
- The dataset *schema*. A schema maps the column headers of your dataset. For instance, when monitoring demand, you might have collected hourly data on the sales of an item at multiple stores. In this case, your schema would define the order, from left to right, in which timestamp, location, and hourly sales appear in your training data file. Schemas also define each column's data type, such as `string` or `integer`.
- Geolocation and time zone information. The geolocation attribute is defined within the schema with the attribute type `geolocation`. Time zone information is defined with the CreateDatasetImportJob operation. Both geolocation and time zone data must be included to enable the Weather Index.

Each column in your Forecast dataset represents either a forecast *dimension* or *feature*. Forecast dimensions describe the aspects of your data that do not change over time, such a `store` or `location`. Forecast features include any parameters in your data that vary across time, such as `price` or `promotion`. Some dimensions, like `timestamp` or `itemId`, are required in target time series and related time series datasets.

Dataset Domains and Dataset Types

When you create a Forecast dataset, you choose a domain and a dataset type. Forecast provides domains for a number of use cases, such as forecasting retail demand or web traffic. You can also create a custom domain. For a complete list of Forecast domains, see Predefined Dataset Domains and Dataset Types.

Within each domain, Forecast users can specify the following types of datasets:

- Target time series dataset (required) – Use this dataset type when your training data is a time series *and* it includes the field that you want to generate a forecast for. This field is called the *target field*.
- Related time series dataset (optional) – Choose this dataset type when your training data is a time series, but it *doesn't* include the target field. For instance, if you're forecasting item demand, a related time series dataset might have `price` as a field, but not `demand`.

- Item metadata dataset (optional) – Choose this dataset type when your training data *isn't* time-series data, but includes metadata information about the items in the target time series or related time series datasets. For instance, if you're forecasting item demand, an item metadata dataset might have `color` or `brand` as dimensions.

  Forecast only considers the data provided by an item metadata dataset type when you use the CNN-QR or DeepAR+ algorithm.

  Item metadata is especially useful in coldstart forecasting scenarios, in which you have little direct historical data with which to make predictions, but do have historical data on items with similar metadata attributes. When you include item metadata, Forecast creates coldstart forecasts based on similar time series, which can create a more accurate forecast.

Depending on the information in your training data and what you want to forecast, you might create more than one dataset.

For example, suppose that you want to generate a forecast for the demand of retail items, such as shoes and socks. You might create the following datasets in the RETAIL domain:

- Target time series dataset – Includes the historical time-series demand data for the retail items (`item_id`, `timestamp`, and the target field `demand`). Because it designates the target field that you want to forecast, you must have at least one target time series dataset in a dataset group.

  You can also add up to ten other dimensions to a target time series dataset. If you include only a target time series dataset in your dataset group, you can create forecasts at either the item level or the forecast dimension level of granularity only. For more information, see CreatePredictor.
- Related time series dataset – Includes historical time-series data other than the target field, such as `price` or `revenue`. Because related time series data must be mappable to target time series data, each related time series dataset must contain the same identifying fields. In the RETAIL domain, these would be `item_id` and `timestamp`.

  A related time series dataset might contain data that refines the forecasts made off of your target time series dataset. For example, you might include `price` data in your related time series dataset on the future dates that you want to generate a forecast for. This way, Forecast can make predictions with an additional dimension of context. For more information, see Using Related Time Series Datasets.
- Item metadata dataset – Includes metadata for the retail items. Examples of metadata include `brand`, `category`,`color`, and `genre`.

Example Dataset with a Forecast Dimension

Continuing with the preceding example, imagine that you want to forecast the demand for shoes and socks based on a store's previous sales. In the following target time series dataset, `store` is a time-series forecast dimension, while `demand` is the target field. Socks are sold in two store locations (NYC and SFO), and shoes are sold only in ORD.

The first three rows of this table contain the first available sales data for the NYC, SFO, and ORD stores. The last three rows contain the last recorded sales data for each store. The `...` row represents all of the item sales data recorded between the first and last entries.

timestamp	item_id	store	demand
2019-01-01	socks	NYC	25
2019-01-05	socks	SFO	45
2019-02-01	shoes	ORD	10
...			
2019-06-01	socks	NYC	100
2019-06-05	socks	SFO	5
2019-07-01	shoes	ORD	50

Dataset Schema

Each dataset requires a schema, a user-provided JSON mapping of the fields in your training data. This is where you list both the required and optional dimensions and features that you want to include in your dataset.

If your dataset includes a geolocation attribute, define the attribute within the schema with the attribute type `geolocation`. For more information, see Adding Geolocation information. In order to apply the Weather Index, you must include a geolocation attribute in your target time series and any related time series datasets.

Some domains have optional dimensions that we recommend including. Optional dimensions are listed in the descriptions of each domain later in this guide. For an example, see RETAIL Domain. All optional dimensions take the data type `string`.

A schema is required for every dataset. The following is the accompanying schema for the example target time series dataset above.

```
{
     "attributes": [
        {
           "AttributeName": "timestamp",
           "AttributeType": "timestamp"
        },
        {
           "AttributeName": "item_id",
           "AttributeType": "string"
        },
        {
           "AttributeName": "store",
           "AttributeType": "string"
        },
        {
           "AttributeName": "demand",
           "AttributeType": "float"
        }
    ]
}
```

When you upload your training data to the dataset that uses this schema, Forecast assumes that the `timestamp` field is column 1, the `item_id` field is column 2, the `store` field is column 3, and the demand field, the *target* field, is column 4.

For the related time series dataset type, all related features must have a float or integer attribute type. For the item metadata dataset type, all features must have a string attribute type. For more information, see SchemaAttribute.

Note

An `attributeName` and `attributeType` pair is required for every column in the dataset. Forecast reserves a number of names that can't be used as the name of a schema attribute. For the list of reserved names, see Reserved Field Names.

Dataset Groups

A *dataset group* is a collection of one to three complimentary datasets, one of each dataset type. You import datasets to a dataset group, then use the dataset group to train a predictor.

Forecast includes the following operations to create dataset groups and add datasets to them:

- CreateDatasetGroup
- UpdateDatasetGroup

Resolving Conflicts in Data Collection Frequency

Forecast can train predictors with data that doesn't align with the data frequency you specify in the CreateDataset operation. For example, you can import data in recorded in hourly intervals even though some of the data isn't timestamped at the top of the hour (02:20, 02:45). Forecast uses the data frequency you specify to learn about your data. Then Forecast aggregates the data during predictor training. For more information see Data aggregation for different forecast frequencies.

Using Related Time Series Datasets

A related time series dataset includes time-series data that isn't included in a target time series dataset and might improve the accuracy of your predictor.

For example, in the demand forecasting domain, a target time series dataset would contain `timestamp` and `item_id` dimensions, while a complementary related time series dataset also includes the following supplementary features: `item price`, `promotion`, and `weather`.

A related time series dataset can contain up to 10 forecast dimensions (the same ones in your target time series dataset) and up to 13 related time-series features.

> **Python notebooks**
>
> For a step-by-step guide on using related time-series datasets, see Incorporating Related Time Series.

Topics

- Historical and Forward-looking Related Time Series

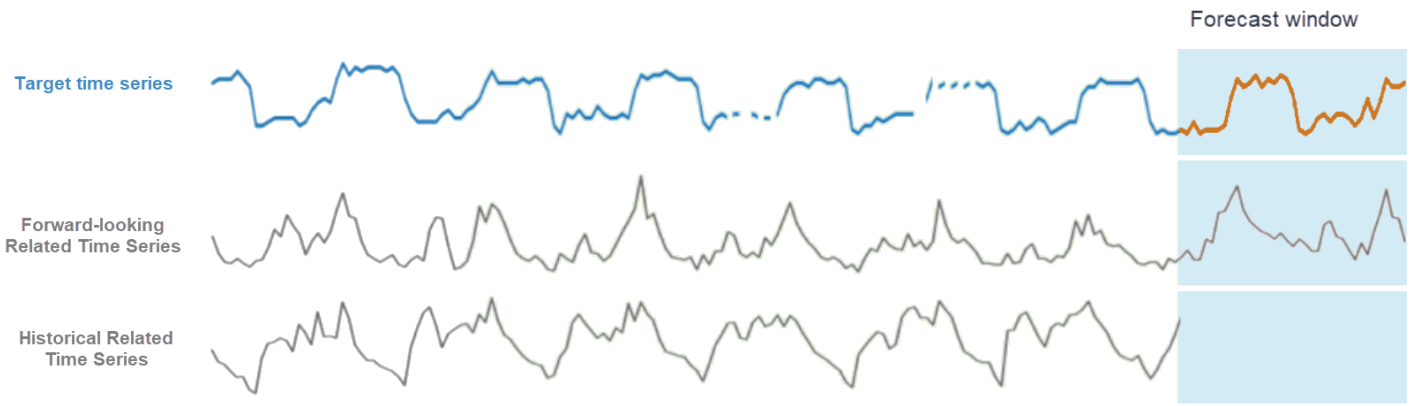
Historical and Forward-looking Related Time Series

> **Note**
>
> A related time series that contains any values within the forecast horizon is treated as a forward-looking time series.

Related time series come in two forms:

- **Historical time series**: time series *without* data points within the forecast horizon.
- **Forward-looking time series**: time series *with* data points within the forecast horizon.

Historical related time series contain data points up to the forecast horizon, and do not contain any data points within the forecast horizon. Forward-looking related time series contain data points up to *and* within the forecast horizon.



Related Time Series Dataset Validation

A related time series dataset has the following restrictions:

- It can't include the target value from the target time series.

- It must include `item_id` and `timestamp` dimensions, and at least one related feature (such as `price`).
- Related time series feature data must be of the `int` or `float` datatypes.
- In order to use the entire target time series, all items from the target time series dataset must also be included in the related time series dataset. If a related time series only contains a subset of items from the target time series, then the model creation and forecast generation will be limited to that specific subset of items.

  For example, if the target time series contains 1000 items and the related time series dataset only contains 100 items, then the model and forecasts will be based on only those 100 items.
- The frequency at which data is recorded in the related time series dataset must match the interval at which you want to generate forecasts (the forecasting *granularity*).

  For example, if you want to generate forecasts at a weekly granularity, the frequency at which data is recorded in the related time series must also be weekly, even if the frequency at which data is recorded in the target time series is daily.
- The data for each item in the related time series dataset must start on or before the beginning `timestamp` of the corresponding `item_id` in the target time series dataset.

  For example, if the target time series data for `socks` starts at 2019-01-01 and the target time series data for `shoes` starts at 2019-02-01, the related time series data for `socks` must begin on or before 2019-01-01 and the data for `shoes` must begin on or before 2019-02-01.
- For forward-looking related time series datasets, the last timestamp for every item must be on the last timestamp in the user-designated forecast window (called the *forecast horizon*).

  In the example related time series file below, the `timestamp` data for both socks and shoes must end on or after 2019-07-01 (the last recorded timestamp) *plus* the forecast horizon. If data frequency in the target time series is daily and the forecast horizon is 10 days, daily data points must be provided in the forward-looking related time series file until 2019-07-11.
- For historical related time series datasets, the last timestamp for every item must match the last timestamp in the target time series.

  In the example related time series file below, the `timestamp` data for both socks and shoes must end on 2019-07-01 (the last recorded timestamp).
- The Forecast dimensions provided in the related time series dataset must be either equal to or a subset of the dimensions designated in the target time series dataset.

- Related time series cannot have missing values. For information on missing values in a related time series dataset, see Handling Missing Values.

Example: Forward-looking Related Time Series File

The following table shows a correctly configured related time series dataset file. For this example, assume the following:

- The last data point was recorded in the target time series dataset on 2019-07-01.
- The forecast horizon is 10 days.
- The forecast granularity is daily (D).

A "..." row indicates all of the data points in between the previous and succeeding rows.

timestamp	item_id	store	price
2019-01-01	socks	NYC	10
2019-01-02	socks	NYC	10
2019-01-03	socks	NYC	15
. . .			
2019-06-01	socks	NYC	10
. . .			
2019-07-01	socks	NYC	10
. . .			
2019-07-11	socks	NYC	20
2019-01-05	socks	SFO	45
. . .			
2019-06-05	socks	SFO	10

timestamp	item_id	store	price
...			
2019-07-01	socks	SFO	10
...			
2019-07-11	socks	SFO	30
2019-02-01	shoes	ORD	50
...			
2019-07-01	shoes	ORD	75
...			
2019-07-11	shoes	ORD	60

Example: Forecasting Granularity

The following table shows compatible data recording frequencies for target time series and related time series to forecast at a weekly granularity. Because data in a related time series dataset can't be aggregated, Forecast accepts only a related time series data frequency that is the same as the chosen forecasting granularity.

Target Input Data Frequency	Related Time Series Frequency	Forecasting Granularity	Supported by Forecast?
Daily	Weekly	Weekly	Yes
Weekly	Weekly	Weekly	Yes
N/A	Weekly	Weekly	Yes
Daily	Daily	Weekly	No

Legacy Predictors and Related Time Series

> **Note**
>
> To upgrade an existing predictor to AutoPredictor, see the section called "Upgrading to AutoPredictor"

When using a legacy predictor, you can use a related time series dataset when training a predictor with the CNN-QR, DeepAR+, and Prophet algorithms. NPTS, ARIMA, and ETS do not accept related time series data.

The following table shows the types of related time series each Amazon Forecast algorithm accepts.

	CNN-QR	DeepAR+	Prophet	NPTS	ARIMA	ETS
Historical related time series						
Forward-looking related time series						

When using AutoML, you can provide both historical and forward-looking related time series data, and Forecast will only use those time series where applicable.

If you provide *forward-looking* related time series data, Forecast will use the related data with CNN-QR, DeepAR+, and Prophet, and will not use the related data with NPTS, ARIMA and ETS. If provided *historical* related time series data, Forecast will use the related data with CNN-QR, and will not use the related data with DeepAR+, Prophet, NPTS, ARIMA, and ETS.

Using Item Metadata Datasets

An *item metadata dataset* contains categorical data that provides valuable context for the items in a target time-series dataset. Unlike related time-series datasets, item metadata datasets provide

information that is static. That is, the data values remain constant over time, like an item's color or brand. Item metadata datasets are optional additions to your dataset groups. You can use an item metadata only if every item in your target time-series dataset is present in the corresponding item metadata dataset.

Item metadata might include the brand, color, model, category, place of origin, or other supplemental feature of a particular item. For example, an item metadata dataset might provide context for some of the demand data found in a target time-series dataset that represents the sales of black Amazon e-readers with 32 GB of storage. Because these characteristics don't change from day-to-day or hour-to-hour, they belong in an item metadata dataset.

Item metadata is useful for discovering and tracking descriptive patterns across your time-series data. If you include an item metadata dataset in your dataset group, Forecast can train the model to make more accurate predictions based on similarities across items. For example, you might find that virtual assistant products made by Amazon are more likely to sell out than those created by other companies, and then plan your supply chain accordingly.

Item metadata is especially useful in coldstart forecasting scenarios, in which you have no historical data with which to make predictions, but do have historical data on items with similar metadata attributes. The item metadata enables Forecast to leverage similar items to your coldstart items to produce a forecast.

When you include item metadata, Forecast creates coldstart forecasts based on similar time series, which can create a more accurate forecast. Coldstart forecasts are generated for items that are in the item metadata dataset but not in the trailing time series. First, Forecast generates forecasts for the non-coldstart items, which are items with historical data in the trailing time series. Next, for each coldstart item, its nearest neighbors are found using the item metadata dataset. Then, these nearest neighbors are used to create a coldstart forecast.

Each row in an item metadata dataset can contain up to 10 metadata fields, one of which must be an identification field to match the metadata to an item in the target time series. As with all dataset types, the values of each field are designated by a dataset schema.

Python notebooks

For a step-by-step guide on using item metadata, see Incorporating Item Metadata.

Topics

Example: Item Metadata File and Schema

The following table shows a section of a correctly configured item metadata dataset file that describes Amazon e-readers. For this example, assume that the header row represents the dataset's schema, and that each listed item is in a corresponding target time-series dataset.

item_id	brand	model	color	waterproof
1	amazon	paperwhite	black	yes
2	amazon	paperwhite	blue	yes
3	amazon	base_model	black	no
4	amazon	base_model	white	no
...				

The following is the same information represented in CSV format.

```
1,amazon,paperwhite,black,yes
2,amazon,paperwhite,blue,yes
3,amazon,base_model,black,no
4,amazon,base_model,white,no
...
```

The following is the schema for this example dataset.

```
{
     "attributes": [
        {
           "AttributeName": "item_id",
           "AttributeType": "string"
        },
```

```
        {
           "AttributeName": "brand",
           "AttributeType": "string"
        },
        {
           "AttributeName": "model",
           "AttributeType": "string"
        },
        {
           "AttributeName": "color",
           "AttributeType": "string"
        },
        {
           "AttributeName": "waterproof",
           "AttributeType": "string"
        }
    ]
}
```

Legacy Predictors and Item Metadata

> **Note**
>
> To upgrade an existing predictor to AutoPredictor, see the section called "Upgrading to AutoPredictor"

When using a legacy predictor, you can use item metadata when training a predictor with the CNN-QR or DeepAR+ algorithms. When using AutoML, you can provide Item metadata and Forecast will only use those time series where applicable

See Also

For an in-depth walkthrough on using item metadata datasets, see Incorporating Item Metadata Datasets into Your Predictor in the Amazon Forecast Samples GitHub Repository.

Predefined Dataset Domains and Dataset Types

To train a predictor, you create one or more datasets, add them to a dataset group, and provide the dataset group for training.

For each dataset that you create, you associate a dataset domain and a dataset type. A *dataset domain* specifies a pre-defined dataset schema for a common use case, and does not impact model algorithms or hyperparameters.

Amazon Forecast supports the following dataset domains:

- RETAIL Domain – For retail demand forecasting
- INVENTORY_PLANNING Domain – For supply chain and inventory planning
- EC2 CAPACITY Domain – For forecasting Amazon Elastic Compute Cloud (Amazon EC2) capacity
- WORK_FORCE Domain – For work force planning
- WEB_TRAFFIC Domain – For estimating future web traffic
- METRICS Domain – For forecasting metrics, such as revenue and cash flow
- CUSTOM Domain – For all other types of time-series forecasting

Each domain can have one to three *dataset types*. The dataset types that you create for a domain are based on the type of data that you have and what you want to include in training.

Each domain requires a target time series dataset, and optionally supports the related time series and item metadata dataset types.

The dataset types are:

- Target time series – The only required dataset type. This type defines the *target* field that you want to generate forecasts for. For example, if you want to forecast the sales for a set of products, then you must create a dataset of historical time-series data for each of the products that you want to forecast. Similarly, you can create a target time series dataset for metrics—such as revenue, cash flow, and sales—that you might want to forecast.
- Related time series – Time-series data that is related to the target time series data. For example, price is related to product sales data, so you might provide it as a related time series.
- Item metadata – Metadata that is applicable to the target time-series data. For example, if you are forecasting sales for a particular product, attributes of the product—such as brand, color, and genre—will be part of item metadata. When predicting EC2 capacity for EC2 instances, metadata might include the CPU and memory of the instance types.

For each dataset type, your input data must contain certain required fields. You can also include optional fields that Amazon Forecast suggests that you include.

The following examples show how to choose a dataset domain and corresponding dataset types.

Example Example 1: Dataset Types in the RETAIL Domain

If you are a retailer interested in forecasting demand for items, you might create the following datasets in the RETAIL domain:

- Target time series is the required dataset of historical time-series demand (sales) data for each item (each product a retailer sells). In the RETAIL domain, this dataset type requires that the dataset includes the `item_id`, `timestamp`, and the `demand` fields. The `demand` field is the forecast target, and is typically the number of items sold by the retailer in a particular week or day.
- Optionally, a dataset of the related time series type. In the RETAIL domain, this type can include optional, but suggested, time-series information such as `price`, `inventory_onhand`, and `webpage_hits`.
- Optionally, a dataset of the item metadata type. In the RETAIL domain, Amazon Forecast suggests providing metadata information related to the items that you provided in target time series, such as `brand`, `color`, `category`, and `genre`.

Example Example 2: Dataset Types in the METRICS Domain

If you want to forecast key metrics for your organization—such as revenue, sales and cash flow—you can provide Amazon Forecast with the following datasets:

- The target time series dataset that provides historical time-series data for the metric that you want to forecast. If your interest is to forecast the revenue of all of the business units in your organization, you can create a `target time series` dataset with the `metric`, `business unit`, and `metric_value` fields.
- If you have any metadata for each metric that isn't required, such as `category` or `location`, you might provide datasets of the related time series and item metadata type.

At a minimum, you must provide a target time series dataset for Forecast to generate forecasts for your target metrics.

Example Example 3: Dataset Types in the CUSTOM Domain

The training data for your forecasting application might not fit into any of the Amazon Forecast domains. If that's the case, choose the CUSTOM domain. You must provide the target time series dataset, but you can add your own custom fields.

The Getting Started exercise forecasts electricity usage for a client. The electricity usage training data doesn't fit into any of the dataset domains, so we used the CUSTOM domain. In the exercise, we use only one dataset type, the target time series type. We map the data fields to the minimum fields required by the dataset type.

RETAIL Domain

The RETAIL domain supports the following dataset types. For each dataset type, we list required and optional fields. For information on how to map the fields to columns in your training data, see Dataset Domains and Dataset Types.

Topics

- Target Time Series Dataset Type
- Related Time Series Dataset Type
- Item Metadata Dataset Type

Target Time Series Dataset Type

The target time series is the historical time series data for each item or product sold by the retail organization. The following fields are required:

- `item_id` (string) – A unique identifier for the item or product that you want to predict the demand for.
- `timestamp` (timestamp)
- `demand` (float) – The number of sales for that item at the timestamp. This is also the *target* field for which Amazon Forecast generates a forecast.

The following dimension is optional and can be used to change forecasting granularity:

- `location` (string) – The location of the store that the item got sold at. This should only be used if you have multiple stores/locations.

Ideally, only these required fields and optional dimensions should be included. Other additional time series information should be included in a related time series dataset.

Related Time Series Dataset Type

You can provide Amazon Forecast with related time series datasets, such as the price or the number of web hits the item received on a particular date. The more information that you provide, the more accurate the forecast. The following fields are required:

- `item_id` (string)
- `timestamp` (timestamp)

The following fields are optional and might be useful in improving forecast results:

- `price` (float) – The price of the item at the time of the timestamp.
- `promotion_applied` (integer; 1=true, 0=false) – A flag that specifies whether there was a marketing promotion for that item at the timestamp.

In addition to the required and suggested optional fields, your training data can include other fields. To include other fields in the dataset, provide the fields in a schema when you create the dataset.

Item Metadata Dataset Type

This dataset provides Amazon Forecast with information about metadata (attributes) of the items whose demand is being forecast. The following fields are required:

- `item_id` (string)

The following fields are optional and might be useful in improving forecast results:

- `category` (string)
- `brand` (string)
- `color` (string)
- `genre` (string)

In addition to the required and suggested optional fields, your training data can include other fields. To include other fields in the dataset, provide the fields in a schema when you create the dataset.

CUSTOM Domain

The CUSTOM domain supports the following dataset types. For each dataset type, we list required and optional fields. For information on how to map the fields to columns in your training data, see Dataset Domains and Dataset Types.

Topics

Target Time Series Dataset Type

The following fields are required:

- `item_id` (string)
- `timestamp` (timestamp)
- `target_value` (floating-point integer) – This is the `target` field for which Amazon Forecast generates a forecast.

Ideally, only these required fields should be included. Other additional time series information should be included in a related time series dataset.

Related Time Series Dataset Type

The following fields are required:

- `item_id` (string)
- `timestamp` (timestamp)

In addition to the required fields, your training data can include other fields. To include other fields in the dataset, provide the fields in a schema when you create the dataset.

Item Metadata Dataset Type

The following field is required:

- `item_id` (string)

The following field is optional and might be useful in improving forecast results:

- `category` (string)

In addition to the required and suggested optional fields, your training data can include other fields. To include other fields in the dataset, provide the fields in a schema when you create the dataset.

INVENTORY_PLANNING Domain

Use the INVENTORY_PLANNING domain for forecasting demand for raw materials and determining how much inventory of a particular item to stock. It supports the following dataset types. For each dataset type, we list required and optional fields. For information on how to map the fields to columns in your training data, see Dataset Domains and Dataset Types.

Topics

Target Time Series Dataset Type

The following fields are required:

- `item_id` (string)
- `timestamp` (timestamp)
- `demand` (float) – This is the `target` field for which Amazon Forecast generates a forecast.

The following dimension is optional and can be used to change forecasting granularity:

- `location` (string) – The location of the distribution center where the item is stocked. This should only be used if you have multiple stores/locations.

Ideally, only these required fields and optional dimensions should be included. Other additional time series information should be included in a related time series dataset.

Related Time Series Dataset Type

The following fields are required:

- `item_id` (string)
- `timestamp` (timestamp)

The following fields are optional and might be useful in improving forecast results:

- `price` (float) – The price of the item

In addition to the required and suggested optional fields, your training data can include other fields. To include other fields in the dataset, provide the fields in a schema when you create the dataset.

Item Metadata Dataset Type

The following fields are required:

- `item_id` (string)

The following fields are optional and might be useful in improving forecast results:

- `category` (string) – The category of the item.
- `brand` (string) – The brand of the item.
- `lead_time` (string) – The lead time, in days, to manufacture the item.
- `order_cycle` (string) – The order cycle starts when work begins and ends when the item is ready for delivery.
- `safety_stock` (string) – The minimum amount of stock to keep on hand for that item.

In addition to the required and suggested optional fields, your training data can include other fields. To include other fields in the dataset, provide the fields in a schema when you create the dataset.

EC2 CAPACITY Domain

Use the EC2 CAPACITY domain for forecasting Amazon EC2 capacity. It supports the following dataset types. For each dataset type, we list required and optional fields. For information on how to map the fields to columns in your training data, see Dataset Domains and Dataset Types.

Target Time Series Dataset Type

The following fields are required:

- `instance_type` (string) – The type of instance (for example, c5.xlarge).
- `timestamp` (timestamp)
- `number_of_instances` (integer) – The number of instances of that particular instance type that was consumed at the timestamp. This is the `target` field for which Amazon Forecast generates a forecast.

The following dimension is optional and can be used to change forecasting granularity:

- `location` (string) – You can provide an AWS Region, such as us-west-2 or us-east-1. This should only be used if you're modeling multiple Regions.

Ideally, only these required and suggested optional fields should be included. Other additional time series information should be included in a related time series dataset.

Related Time Series Dataset Type

The following fields are required:

- `instance_type` (string)
- `timestamp` (timestamp)

In addition to the required fields, your training data can include other fields. To include other fields in the dataset, provide the fields in a schema when you create the dataset.

WORK_FORCE Domain

Use the WORK_FORCE domain to forecast workforce demand. It supports the following dataset types. For each dataset type, we list required and optional fields. For information on how to map the fields to columns in your training data, see Dataset Domains and Dataset Types.

Topics

Target Time Series Dataset Type

The following fields are required:

- `workforce_type` (string) – The type of work force labor being forecast. For example, call center demand or fulfillment center labor demand.
- `timestamp` (timestamp)
- `workforce_demand` (floating-point integer) – This is the `target` field for which Amazon Forecast generates a forecast.

The following dimension is optional and can be used to change forecasting granularity:

- `location` (string) – The location where the work force resources are sought. This should be used if you have multiple stores/locations.

Ideally, only these required fields and optional dimensions should be included. Other additional time series information should be included in a related time series dataset.

Related Time Series Dataset Type

The following fields are required:

- `workforce_type` (string)
- `timestamp` (timestamp)

In addition to the required fields, your training data can include other fields. To include other fields in the dataset, provide the fields in a schema when you create the dataset.

Item Metadata Dataset Type

The following field is required:

- `workforce_type` (string)

The following fields are optional and might be useful in improving forecast results:

- `wages` (float) – The average wages for that particular workforce type.
- `shift_length` (string) – The length of the shift.
- `location` (string) – The location of the workforce.

In addition to the required and suggested optional fields, your training data can include other fields. To include other fields in the dataset, provide the fields in a schema when you create the dataset.

WEB_TRAFFIC Domain

Use the WEB_TRAFFIC domain to forecast web traffic to a web property or a set of web properties. It supports the following dataset types. The relevant topics describe required and optional fields the dataset type supports. For information about how to map these fields to columns in your training data see Dataset Domains and Dataset Types.

Topics

Target Time Series Dataset Type

The following fields are required:

- `item_id` (string) – A unique identifier for each web property being forecast.
- `timestamp` (timestamp)
- `value` (float) – This is the `target` field for which Amazon Forecast generates a forecast.

Ideally, only these required fields should be included. Other additional time series information should be included in a related time series dataset.

Related Time Series Dataset Type

The following fields are required:

- `item_id` (string)
- `timestamp` (timestamp)

In addition to the required fields, your training data can include other fields. To include other fields in the dataset, provide the fields in a schema when you create the dataset.

Item Metadata Dataset Type

The following field is required:

- `item_id` (string)

The following field is optional and might be useful in improving forecast results:

- `category` (string)

In addition to the required and suggested optional fields, your training data can include other fields. To include other fields in the dataset, provide the fields in a schema when you create the dataset.

METRICS Domain

Use the METRICS domain for forecasting metrics, such as revenue, sales, and cash flow. It supports the following dataset types. For each dataset type, we list required and optional fields. For information on how to map the fields to columns in your training data, see Dataset Domains and Dataset Types.

Topics

- Item Metadata Dataset Type

Target Time Series Dataset Type

The following fields are required:

- `metric_name` (string)
- `timestamp` (timestamp)
- `metric_value` (floating-point integer) – This is the `target` field for which Amazon Forecast generates a forecast (for example, the amount of revenue generated on a particular day).

Ideally, only these required fields should be included. Other additional time series information should be included in a related time series dataset.

Related Time Series Dataset Type

The following fields are required:

- `metric_name` (string)
- `timestamp` (timestamp)

In addition to the required fields, your training data can include other fields. To include other fields in the dataset, provide the fields in a schema when you create the dataset.

Item Metadata Dataset Type

The following field is required:

- `metric_name` (string)

The following field is optional and might be useful in improving forecast results:

- `category` (string)

In addition to the required and suggested optional fields, your training data can include other fields. To include other fields in the dataset, provide the fields in a schema when you create the dataset.

Updating Data

As you collect new data, you will want to import that data into Forecast. To do so, you have two options, replacement and incremental updates. A replacement dataset import job will overwrite all existing data with the newly imported data. An incremental update will append the newly imported data to the dataset.

After importing your new data, you can use an existing predictor to generate a forecast for that data.

Topics

Import modes

To configure how Amazon Forecast adds new data to existing dataset, you specify the import mode for your dataset import job. The default import mode is `FULL`. You can only configure the import mode by using the Amazon Forecast API.

- To overwrite all existing data in your dataset, specify `FULL` in the CreateDatasetImportJob API operation.
- To append the records to the existing data in your dataset, specify `INCREMENTAL` in the CreateDatasetImportJob API operation. If an existing record and an imported record have the same timeseries ID (item ID, dimension, and timestamp), then the existing record is replaced with the newly imported record. Amazon Forecast always uses the record with the most recent timestamp.

If you have not imported a dataset, the incremental option is not available. The default import mode is a full replacement.

Incremental import mode guidelines

When you perform an incremental dataset import, you cannot change the timestamp format, data format, or geolocation data. To change any of these items, you need to perform a full data dataset import.

Updating existing datasets

> **Important**
>
> By default, a dataset import job replaces any existing data in the dataset that you imported into. You can change this by specifying the dataset import job's Import modes.

To update a dataset, create a dataset import job for the dataset and specify the import mode.

CLI

To update a dataset, use the `create-dataset-import-job` command. For the `import-mode`, specify `FULL`, to replace existing data or `INCREMENTAL` to add to it. For more information, see Import modes.

The following code shows how to create a dataset import job that incrementally imports new data into a dataset.

```
aws forecast create-dataset-import-job \
                        --dataset-import-job-name dataset import job name \
                        --dataset-arn dataset arn \
                        --data-source "S3Config":{"KMSKeyArn":"string",
 "Path":"string", "RoleArn":"string"} \
                        --import-mode INCREMENTAL
```

Python

To update a dataset, use the `create_dataset_import_job` method. For the `import-mode`, specify `FULL`, to replace existing data or `INCREMENTAL` to add to it. For more information, see Import modes.

```
import boto3

forecast = boto3.client('forecast')

response = forecast.create_dataset_import_job(
    datasetImportJobName = 'YourImportJob',
    datasetArn = 'dataset_arn',
    dataSource = {"S3Config":{"KMSKeyArn":"string", "Path":"string",
 "RoleArn":"string"}},
```

```
    importMode = 'INCREMENTAL'
)
```

Updating forecasts

As you collect new data, you might want to use it to generate new forecasts. Forecast does not automatically retrain a predictor when you import an updated dataset, but you can manually retrain a predictor to generate a new forecast with the updated data. For instance, if you collect daily sales data and want to include new data points in your forecast, you could import the updated data and use it to generate a forecast without training a new predictor. For newly imported data to have an impact on your forecasts, you must retrain the predictor.

To generate a forecast from the new data:

1. Upload the new data to an Amazon S3 bucket. Your new data should contain only the data added since your last data set import.
2. Create an **Incremental** dataset import job with the new data. The new data is appended to the existing data and the forecast is generated from the updated data. If your new data file contains both previously-imported data and new data, create a **Full** dataset import job.
3. Create a new forecast using the existing predictor.
4. Retrieve the forecast as usual.

Handling Missing Values

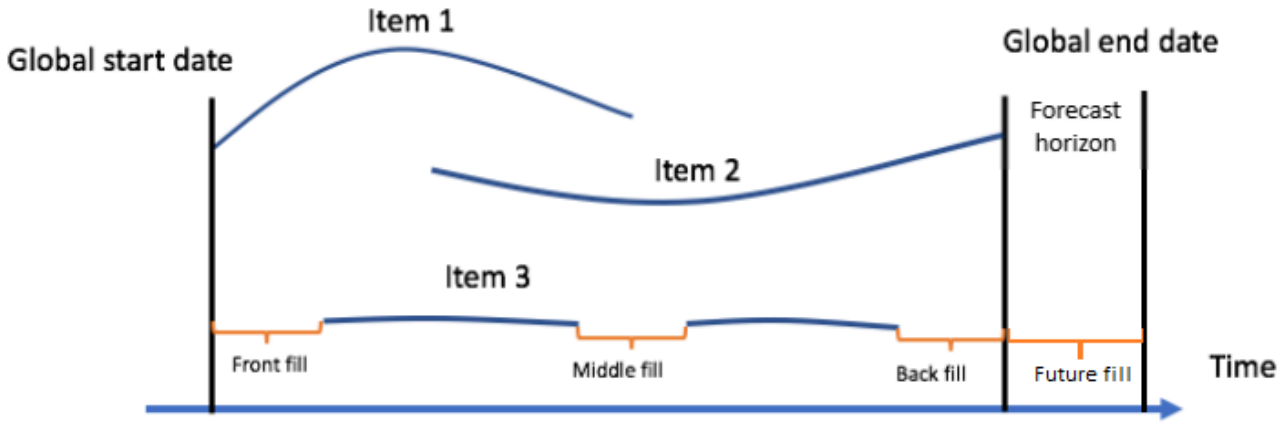
A common issue in time-series forecasting data is the presence of missing values. Your data might contain missing values for a number of reasons, including measurement failures, formatting problems, human errors, or a lack of information to record. For instance, if you're forecasting product demand for a retail store and an item is sold out or unavailable, there would be no sales data to record while that item is out of stock. If prevalent enough, missing values can significantly impact a model's accuracy.

Amazon Forecast provides a number of filling methods to handle missing values in your target time series and related time series datasets. Filling is the process of adding standardized values to missing entries in your dataset.

Forecast supports the following filling methods:

- **Middle filling** – Fills any missing values between the item start and item end date of a data set.
- **Back filling** – Fills any missing values between the last recorded data point and global end date of a dataset.
- **Future filling (related time series only)** – Fills any missing values between the global end date and the end of the forecast horizon.

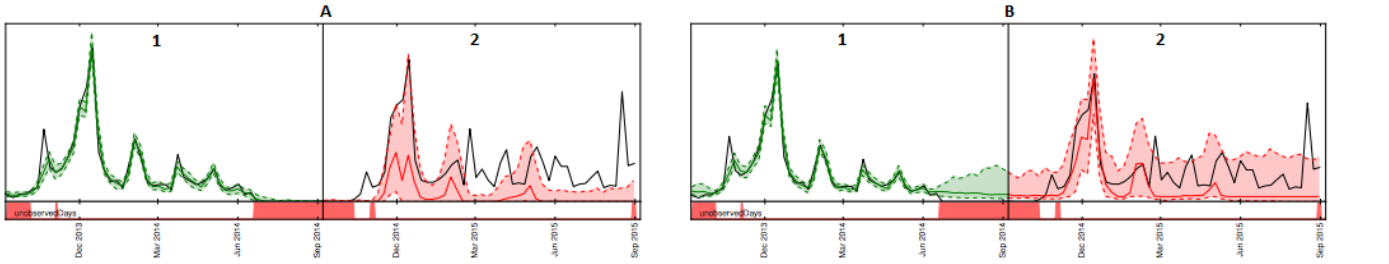
The following image provides a visual representation of different filling methods.



Choosing Filling Logic

When choosing a filling logic, you should consider how the logic will be interpreted by your model. For instance, in a retail scenario, recording 0 sales of an available item is different from recording 0 sales of an unavailable item, as the latter does not imply a lack of customer interest in the item. Because of this, `0` filling in the target time series might cause the predictor to be under-biased in its predictions, while `NaN` filling might ignore actual occurrences of 0 available items being sold and cause the predictor to be over-biased.

The following time-series graphs illustrate how choosing the wrong filling value can significantly affect the accuracy of your model. Graphs A and B plot the demand for an item that is partially out-of-stock, with the black lines representing actual sales data. Missing values in A1 are filled with `0`, leading to relatively under-biased predictions (represented by the dotted lines) in A2. Similarly, missing values in B1 are filled with `NaN`, which leads to predictions that are more exact in B2.



For a list of supported filling logic, see the following section.

Target Time Series and Related Time Series Filling Logic

You can perform filling on both target time series and related time series datasets. Each dataset type has different filling guidelines and restrictions.

Filling Guidelines

Dataset type	Filling by default?	Supported filling methods	Default filling logic	Accepted filling logic
Target time series	Yes	Middle and back filling	0	• `zero` - 0 filling. • `value` - an integer or float number. • `nan` - not a number. • `mean` - the mean value from the data series. • `median` - the median value from the data series. • `min` - the minimum

Dataset type	Filling by default?	Supported filling methods	Default filling logic	Accepted filling logic
				value from the data series. • `max` - the maximum value from the data series.
Related time series	No	Middle, back, and future filling	No default	• `zero` - 0 filling. • `value` - an integer or float value. • `mean` - the mean value from the data series. • `median` - the median value from the data series. • `min` - the minimum value from the data series. • `max` - the maximum value from the data series.

> **Important**
>
> For both target and related time series datasets, `mean`, `median`, `min`, and `max` are calculated based on a rolling window of the 64 most recent data entries before the missing values.

Missing Value Syntax

To perform missing value filling, specify the types of filling to implement when you call the CreatePredictor operation. Filling logic is specified in FeaturizationMethod objects.

The following excerpt demonstrates a correctly formatted `FeaturizationMethod` object for a target time series attribute and related time series attribute (`target_value` and `price` respectively).

To set a filling method to a specific value, set the fill parameter to `value` and define the value in a corresponding `_value` parameter. As shown below, backfilling for the related time series is set to a value of 2 with the following: `"backfill": "value"` and `"backfill_value":"2"`.

```
[
    {
        "AttributeName": "target_value",
        "FeaturizationPipeline": [
            {
                "FeaturizationMethodName": "filling",
                "FeaturizationMethodParameters": {
                    "aggregation": "sum",
                    "middlefill": "zero",
                    "backfill": "zero"
                }
            }
        ]
    },
    {
        "AttributeName": "price",
        "FeaturizationPipeline": [
            {
                "FeaturizationMethodName": "filling",
                "FeaturizationMethodParameters": {
                    "middlefill": "median",
```

```
                    "backfill": "value",
                    "backfill_value": "2",
                    "futurefill": "max"
                    }
            }
        ]
    }
]
```

Dataset Guidelines for Forecast

Consult to the following guidelines if Amazon Forecast fails to import your dataset, or if your dataset doesn't function as expected.

Timestamp Format

For Year (`Y`), Month (`M`), Week (`W`), and Day (`D`) collection frequencies, Forecast supports the `yyyy-MM-dd` timestamp format (for example, `2019-08-21`) and, optionally, the `HH:mm:ss` format (for example, `2019-08-21 15:00:00`).

For Hour (`H`) and Minute (`M`) frequencies, Forecast supports only the `yyyy-MM-dd HH:mm:ss` format (for example `2019-08-21 15:00:00`).

Guideline: Change the timestamp format for the collection frequency of your dataset to the supported format.

Amazon S3 File or Bucket

When you import a dataset, you can specify either the path to a CSV or Parquet file in your Amazon Simple Storage Service (Amazon S3) bucket that contains your data or the name of the S3 bucket that contains your data. If you specify a CSV or Parquet file, Forecast imports just that file. If you specify an S3 bucket, Forecast imports all of the CSV or Parquet files in the bucket up to 10,000 files. If you import multiple files by specifying a bucket name, all CSV or Parquet files must conform to the specified schema.

Guideline: Specify a specific file or an S3 bucket using the following syntax:

`s3://bucket-name/example-object.csv`

`s3://bucket-name/example-object.parquet`

`s3://bucket-name/prefix/`

`s3://bucket-name`

Parquet files can have the extension .parquet, .parq, .pqt, or no extension at all.

Full Dataset Updates

Your first dataset import is always a full import, subsequent imports can either be full or incremental updates. You must use the Forecast API to specify the import mode.

With a full update, all existing data is replaced with the newly imported data. Because full dataset import jobs are not aggregated, your most recent dataset import is the one that is used when training a predictor or generating a forecast.

Guideline: Create an incremental dataset update to append your new data to the existing data. Otherwise, ensure that your most recent dataset import contains all of the data you want to model, and not just the new data collected since the previous import.

Incremental Dataset Updates

Fields such as timestamp, data format, geolocation, etc. are read from the currently active dataset. You do not need to include this information with an incremental dataset import. If they are included, they must match the originally provided values.

Guideline: Perform a full dataset import to change any of these values.

Attribute Order

The order of attributes specified in the schema definition must match the column order in the CSV or Parquet file that you are importing. For example, if you defined `timestamp` as the first attribute, then `timestamp` must also be the first column in the input file.

Guideline: Verify that the columns in the input file are in the same order as the schema attributes that you created.

Weather Index

In order to apply the Weather Index, you must include a geolocation attribute in your target time series and any related time series datasets. You also need to specify time zones for your target time series timestamps.

Guideline: Ensure that your datasets include a geolocation attribute and that your timestamps have an assigned time zone. For more information, refer to the Weather Index Conditions and Restrictions.

Dataset Header

A dataset header in your input CSV may cause a validation error. We recommend omitting a header for CSV files.

Guideline: Delete the dataset header and try the import again.

A dataset header is required for Parquet files.

Dataset Status

Before you can import training data with the the section called “CreateDatasetImportJob” operation, the `Status` of the dataset must be `ACTIVE`.

Guideline: Use the DescribeDataset operation to get the dataset's status. If the creation or update of the dataset failed, check the formatting of your dataset file and attempt to create it again.

Default File Format

The default file format is CSV.

File Format and Delimiter

Forecast supports only the comma-separated values (CSV) file format and Parquet format. You can't separate values using tabs, spaces, colons, or any other characters.

Guideline: Convert your dataset to CSV format (using only commas as your delimiter) or Parquet format and try importing the file again.

File Name

File names must contain at least one alphabetic character. Files with names that are only numeric can't be imported.

Guideline: Rename your input data file to include at least one alphabetic character and try importing the file again.

Partitioned Parquet Data

Forecast does not read partitioned Parquet files.

What-if analysis Dataset Requirements

What-if analyses require CSV datasets. The TimeSeriesSelector operation of the CreateWhatIfAnalysis action and the TimeSeriesReplacementDataSource operation of the CreateWhatIfForecast do not accept Parquet files.

Training Predictors

A predictor is an Amazon Forecast model that is trained using your target time series, related time series, item metadata, and any additional datasets you include. You can use predictors to generate forecasts based on your time-series data.

By default, Amazon Forecast creates an AutoPredictor, where Forecast applies the optimal combination of algorithms to each time series in your datasets.

Topics

Creating a Predictor

Amazon Forecast requires the following inputs to train a predictor:

- **Dataset group** – A dataset group that must include a target time series dataset. The target time series dataset includes the target attribute (`item_id`) and timestamp attribute, as well as any dimensions. Related time series and Item metadata is optional. For more information, see *Importing Datasets*.
- **Forecast frequency** – The granularity of your forecasts (hourly, daily, weekly, etc). Amazon Forecast lets you determine the exact granularity of your forecasts when you provide the frequency unit and value. Only integer values are allowed

Frequency unit	Allowed values
Minutely	1-59
Hourly	1-23
Daily	1-6
Weekly	1-4
Monthly	1-11
Yearly	1

For example, if you want every other week forecasts, your frequency unit is weekly and the value is 2. Or, if you want quarterly forecasts, your frequency unit is monthly and the value is 3.

When your data is collected at a greater frequency than the forecast frequency, it is aggregated to the forecast frequency. This includes the trailing time series and related time series data. For more information on aggregation see Data aggregation for different forecast frequencies.

- **Forecast horizon** – The number of time steps being forecasted.

You can also set values for the following optional inputs:

- **Time alignment boundary** – The time boundary Forecast uses to aggregate your data and generate forecasts that align with the forecast frequency you specify. For more information on aggregation see Data aggregation for different forecast frequencies. For information on specifying a time boundary see Time Boundaries.
- **Forecast dimensions** – Dimensions are optional attributes in your target time series dataset that can be used in combination with the target value (`item_id`) to create separate time series.
- **Forecast types** – The quantiles used to evaluate your predictor.
- **Optimization metric** – The accuracy metric used to optimize your predictor.
- **Additional datasets** – Built-in Amazon Forecast datasets like the Weather Index and Holidays.

You can create a predictor using the Software Development Kit (SDK) or the Amazon Forecast console.

Console

To create a predictor

1. Sign in to the AWS Management Console and open the Amazon Forecast console at https://console.aws.amazon.com/forecast/.
2. From **Dataset groups**, choose your dataset group.
3. In the navigation pane, choose **Predictors**.
4. Choose **Train new predictor**.
5. Provide values for the following mandatory fields:
   - **Name** - a unique predictor name.
   - **Forecast frequency** - the granularity of your forecasts.
   - **Forecast horizon** - The number of time steps to forecast.
6. Choose **Start**.

For information on additional datasets, see the section called " Weather Index" and the section called "Holidays Featurization". To learn more about customizing forecast types and optimization metrics, see the section called "Predictor Metrics".

AWS CLI

To create an auto predictor with the AWS CLI, use the `create-predictor` command. The following code creates an auto predictor that makes predictions for 14 days in the future.

Provide a name for the predictor and the Amazon Resource Name (ARN) of the dataset group that includes your training data. Optionally modify the forecast horizon and forecast frequency. Optionally add any tags for the predictor. For more information see Tagging Amazon Forecast Resources.

For information on required and optional parameters see CreateAutoPredictor.

```
aws forecast create-predictor \
--predictor-name predictor_name \
--data-config DatasetGroupArn="arn:aws:forecast:region:account:dataset-
group/datasetGroupName" \
```

```
--forecast-horizon 14 \
--forecast-frequency D \
--tags Key=key1,Value=value1 Key=key2,Value=value2
```

To learn more about customizing forecast types and optimization metrics, see the section called "Predictor Metrics" The Weather Index and Holidays additional datasets are defined within the `DataConfig` datatype. For information on additional datasets, see the section called " Weather Index" and the section called "Holidays Featurization".

Python

To create an auto predictor with the SDK for Python (Boto3), use the `create_auto_predictor` method. The following code creates an auto predictor that makes predictions for 14 days in the future.

Provide a name for the predictor and the Amazon Resource Name (ARN) of the dataset group that includes your training data. Optionally modify the forecast horizon and forecast frequency. Optionally add any tags for the predictor. For more information see Tagging Amazon Forecast Resources.

For information on required and optional parameters see CreateAutoPredictor.

```
import boto3

forecast = boto3.client('forecast')

create_predictor_response = forecast.create_auto_predictor(
    PredictorName = 'predictor_name',
    ForecastHorizon = 14,
    ForecastFrequency = 'D',
    DataConfig = {
      "DatasetGroupArn": "arn:aws:forecast:region:account:dataset-
group/datasetGroupName"
    },
    Tags = [
      {
         "Key": "key1",
         "Value": "value1"
      },
      {
         "Key": "key2",
         "Value": "value2"
```

```
        }
    ]
)
print(create_predictor_response['PredictorArn'])
```

To learn more about customizing forecast types and optimization metrics, see the section called "Predictor Metrics" The Weather Index and Holidays additional datasets are defined within the `DataConfig` datatype. For information on additional datasets, see the section called " Weather Index" and the section called "Holidays Featurization".

Upgrading to AutoPredictor

> **Python notebooks**
>
> For a step-by-step guide on upgrading predictors to AutoPredictor, see Upgrading a predictor to AutoPredictor.

Predictors created with AutoML or manual selection (CreatePredictor) can be upgraded to an AutoPredictor. Upgrading an existing to AutoPredictor will transfer all the relevant predictor configuration settings.

After Upgrading to AutoPredictor, the original predictor will remain active and the upgraded predictor will have a separate Predictor ARN. This enables you to compare accuracy metrics between the two predictors, and you can still generate forecasts with the original predictor.

You can upgrade a predictor using the Software Development Kit (SDK) or the Amazon Forecast console.

Console

To upgrade a predictor

1. Sign in to the AWS Management Console and open the Amazon Forecast console at https://console.aws.amazon.com/forecast/.
2. In the navigation pane, choose **Predictors**.
3. Choose the predictor to upgrade, and choose **Upgrade**.
4. Set a unique name for the upgraded predictor.

5. Choose **Upgrade to AutoPredictor**.

CLI

To upgrade a predictor with the AWS CLI, use the `create-predictor` method, but specify *only* the predictor name and the value of `reference-predictor-arn` (the ARN of the predictor you want to upgrade).

```
aws forecast create-predictor \
--predictor-name predictor_name \
--reference-predictor-arn arn:aws:forecast:region:account:predictor/predictorName
```

Python

To upgrade a predictor with the SDK for Python (Boto3), use the `create_auto_predictor` method, but specify *only* the predictor name and the value of `ReferencePredictorArn` (the ARN of the predictor you want to upgrade).

```
import boto3

forecast = boto3.client('forecast')

create_predictor_response = forecast.create_auto_predictor(
    PredictorName = 'predictor_name',
    ReferencePredictorArn =
 'arn:aws:forecast:region:account:predictor/predictorName'
)
print(create_predictor_response['PredictorArn'])
```

Data aggregation for different forecast frequencies

When you create a predictor, you must specify a forecast frequency. The forecast frequency determines the frequency of predictions in your forecasts. For example, monthly sales forecasts. Amazon Forecast predictors can generate forecasts for data frequencies that are higher than the forecast frequency you specify. For example, you can generate weekly forecasts even if your data is recorded daily. During training, Forecast aggregates the daily data to generate forecasts at the weekly forecast frequency.

Topics

How Aggregation Works

During training, Amazon Forecast aggregates any data that does not align with the forecast frequency you specify. For example, you might have some daily data but specify a weekly forecast frequency. Forecast aligns the daily data based on the week that it belongs in. Forecast then combines it into single record for each week. Forecast determines what week (or month or day and so on) data belongs in based on its relation to a time boundary. Time boundaries specify the beginning of a unit of time, such as what hour a day begins or what day a week begins.

For hourly and minutely forecasts, or unspecified time boundaries, Forecast uses a default time boundary based on your frequency's unit of time. For auto predictors with daily, weekly, monthly, or yearly forecast frequencies, you can specify a custom time boundary. For more information about time boundaries, see Time Boundaries.

During aggregation, the default transformation method is to sum the data. You can configure the transformation when you create your predictor. You do this in the **Input data configuration** section on the **Create predictor** page in the Forecast console. Or you can set the transformation method in the `Transformations` parameter in the AttributeConfig of the CreateAutoPredictor operation.

The following tables show an example aggregation for an hourly forecast frequency using the default time boundary: Each hour begins at the top of the hour.

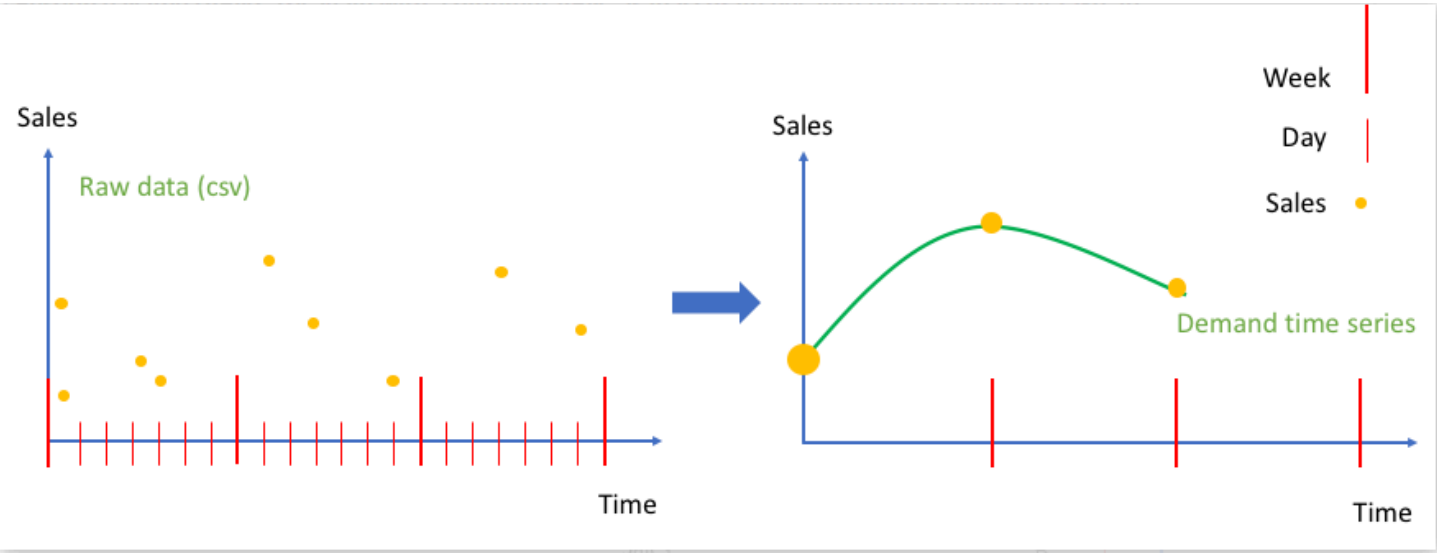
Pre-transformation

Time	Data	At Top of the Hour
2018-03-03 01:00:00	100	Yes
2018-03-03 02:20:00	50	No
2018-03-03 02:45:00	20	No
2018-03-03 04:00:00	120	Yes

Post-transformation

Time	Data	Notes
2018-03-03 01:00:00	100	
2018-03-03 02:00:00	70	Sum of the values between 02:00:00-02:59:59 (50 + 20)
2018-03-03 03:00:00	Empty	No values between 03:00:00-03:59:59
2018-03-03 04:00:00	120	

The following figure shows how Forecast transforms data to fit the default weekly time boundary.



Time Boundaries

Time boundaries specify the beginning of a unit of time, such what day a week begins. Before aggregating your data, Amazon Forecast aligns the data the based on the unit of time of your forecast frequency. It does this based on the data's relation to a time boundary.

For example, if you specify a daily forecast frequency but not your own time boundary, Forecast aligns each hourly record based on the day it belongs in. Each day starts at 0 hours. The definition of when the day starts, 0 hours, is the time boundary. Then Forecast aggregates the hourly records to a single record for that day.

Forecast uses a default time boundary based on your forecast frequency's unit of time. If you create an auto predictor, you can specify a custom time boundary.

If you specify both a custom time boundary and a custom forecast frequency, Forecast aggregates your data within the forecast frequency and aligns it to the custom time boundary. The forecast frequency determines how often the data is aggregated while the custom time boundary determines where the alignment is located. For example, assume your data is collected daily and you want Amazon Forecast to generate quarterly forecasts on the 15th of the month for one year. To do so, set the forecast frequency to every 3 months and the custom time boundary to 15. See the following AWS Command Line Interface example.

```
aws forecast create-predictor \
--predictor-name predictor_name \
--data-config DatasetGroupArn="arn:aws:forecast:region:account:dataset-
group/datasetGroupName" \
--forecast-horizon 4 \
--forecast-frequency 3M \
--time-alignment-boundary DayOfMonth=15
```

In this example, all of the daily data is summed (the default aggregation) to the 15th of every third month.

Note that this aggregation does not require daily data, just that the data is collected monthly or more frequently.

Topics

Default Time Boundaries

The following table lists the default time alignment boundaries that Forecast uses when aggregating data.

Frequency	Boundary
Minute	Last top of the minute (45:00, 06:00)

Frequency	Boundary
Hour	Last top of the hour (09:00:00, 13:00:00)
Day	First hour of the day (hour 0)
Week	Most recent Monday
Month	First day of the month
Year	First day of the year (January 1)

Specifying a Time Boundary

> **Note**
>
> You can only specify a time boundary for an auto predictor.

When you create an auto predictor with a daily, weekly, monthly, or yearly forecast frequency, you can specify the time boundary that Forecast uses to aggregate data. You might specify a time boundary if your business calendar doesn't align with the default time boundaries. For example, you might want to generate monthly forecasts where each month begins on the third day of the month. If you don't specify a time boundary, Forecast uses a set of Default Time Boundaries.

The time boundary unit that you specify must be one unit finer than your forecast frequency. The following table lists the time boundary unit and values that you can specify, organized by forecast frequency.

You can only specify a `Monthly` time boundary with a boundary value of 28 or less.

Forecast frequency unit	Boundary unit	Boundary values
Daily	Hour	0–23
Weekly	Day of week	Monday through Sunday
Monthly	Day of month	1 through 28

Forecast frequency unit	Boundary unit	Boundary values
Yearly	Month	January through December

You specify a time alignment boundary when you create a predictor as follows. For information on the different time boundary units and boundary values you can specify programmatically, see TimeAlignmentBoundary.

Console

To specify a time alignment boundary for a predictor

1. Sign in to the AWS Management Console and open the Amazon Forecast console at https://console.aws.amazon.com/forecast/.
2. From **Dataset groups**, choose your dataset group.
3. In the navigation pane, choose **Predictors**.
4. Choose **Train new predictor**.
5. Provide values for the mandatory **Name**, **Forecast frequency**, and **Forecast horizon** fields.
6. For **Time alignment boundary**, specify the time boundary the predictor will use when aggregating your data. The values in this list depend on the **Forecast frequency** you choose.
7. Choose **Start**. Forecast will aggregate data using the time alignment boundary you specify as it creates your predictor.

AWS CLI

To specify a time alignment boundary for a predictor with the AWS CLI, use the `create-predictor` command. For the `time-alignment-boundary` parameter, provide the unit of time and boundary value. The following code creates an auto predictor that makes predictions for 5 weeks in the future, where each week starts on a Tuesday.

`DayOfWeek` and `DayOfMonth` values must be in all uppercase. For information on the different time boundary units and boundary values you can specify, see TimeAlignmentBoundary. For information on required and optional parameters, see CreateAutoPredictor.

```
aws forecast create-predictor \
```

```
--predictor-name predictor_name \
--data-config DatasetGroupArn="arn:aws:forecast:region:account:dataset-
group/datasetGroupName" \
--forecast-horizon 5 \
--forecast-frequency W \
--time-alignment-boundary DayOfWeek=TUESDAY
```

Python

To specify a time alignment boundary for a predictor with the SDK for Python (Boto3), use the `create_auto_predictor` method. For the `TimeAlignmentBoundary` parameter, provide a dictionary with the unit of time as the key and boundary value as the value. The following code creates an auto predictor that makes predictions for 5 weeks in the future, where each week starts on a Tuesday.

`DayOfWeek` and `DayOfMonth` values must be in all uppercase. For information on the different time boundary units and boundary values you can specify, see TimeAlignmentBoundary. For information on required and optional parameters, see CreateAutoPredictor.

```
import boto3

forecast = boto3.client('forecast')

create_predictor_response = forecast.create_auto_predictor(
    PredictorName = 'predictor_name',
    ForecastHorizon = 5,
    ForecastFrequency = 'W',
    DataConfig = {
      "DatasetGroupArn": "arn:aws:forecast:region:account:dataset-
group/datasetGroupName"
    },
    TimeAlignmentBoundary = {
      "DayOfWeek": "TUESDAY"
    }
)
print(create_predictor_response['PredictorArn'])
```

Data Aggregation Assumptions

Forecast doesn't assume that your data is from any specific time zone. However, it makes the following assumptions when aggregating time series data:

- All data is from the same time zone.
- All forecasts are in the same time zone as the data in the dataset.
- If you specify the the section called “SupplementaryFeature” holiday feature in the the section called “InputDataConfig” parameter for the the section called “CreatePredictor” operation, the input data is from the same country.

Using additional datasets

Amazon Forecast can include the Weather Index and Holidays when creating your predictor. The Weather Index incorporates meteorological information into your model and Holidays incorporates information regarding national holidays.

The Weather Index requires a ‘geolocation’ attribute in your target time series dataset and information regarding time zones for your timestamps. For more information, see the section called “ Weather Index”.

Holidays includes holiday information on over 250 countries. For more information, see the section called “Holidays Featurization”.

Working with legacy predictors

> **Note**
>
> To upgrade an existing predictor to AutoPredictor, see the section called “Upgrading to AutoPredictor”

AutoPredictor is the default and preferred method to create a predictor with Amazon Forecast. AutoPredictor creates predictors by applying the optimal combination of algorithms for each time series in your dataset.

Predictors created with AutoPredictor are generally more accurate than predictors created with AutoML or manual selection. The Forecast Explainability and predictor retraining features are only available for predictors created with AutoPredictor.

Amazon Forecast can also create legacy predictors in the following ways:

1. **AutoML** - Forecast finds the best-performing algorithm and applies it to your entire dataset.

2. **Manual selection** - Manually choose a single algorithm that is applied to your entire dataset.

You might be able to create a legacy predictor using the Software Development Kit (SDK).

SDK

To use AutoML

Using the `CreatePredictor` operation, set the value of `PerformAutoML` to `"true"`.

```
{
    ...
    "PerformAutoML": "true",
}
```

If you use AutoML, you cannot set a value for the following CreatePredictor parameters: `AlgorithmArn`, `HPOConfig`, `TrainingParameters`.

Evaluating Predictor Accuracy

Amazon Forecast produces accuracy metrics to evaluate predictors and help you choose which to use to generate forecasts. Forecast evaluates predictors using Root Mean Square Error (RMSE), Weighted Quantile Loss (wQL), Mean Absolute Percentage Error (MAPE), Mean Absolute Scaled Error (MASE), and Weighted Absolute Percentage Error (WAPE) metrics.

Amazon Forecast uses backtesting to tune parameters and produce accuracy metrics. During backtesting, Forecast automatically splits your time-series data into two sets: a training set and a testing set. The training set is used to train a model and generate forecasts for data points within the testing set. Forecast evaluates the model's accuracy by comparing forecasted values with observed values in the testing set.

Forecast enables you to evaluate predictors using different forecast types, which can be a set of quantile forecasts and the mean forecast. The mean forecast provides a point estimate, whereas quantile forecasts typically provide a range of possible outcomes.

Python notebooks

For a step-by-step guide on evaluating predictor metrics, see Computing Metrics Using Item-level Backtests..

Topics

Interpreting Accuracy Metrics

Amazon Forecast provides Root Mean Square Error (RMSE), Weighted Quantile Loss (wQL), Average Weighted Quantile Loss (Average wQL), Mean Absolute Scaled Error (MASE), Mean Absolute Percentage Error (MAPE), and Weighted Absolute Percentage Error (WAPE) metrics to evaluate your predictors. Along with metrics for the overall predictor, Forecast calculates metrics for each backtest window.

You can view accuracy metrics for your predictors using the Amazon Forecast Software Development Kit (SDK) and the Amazon Forecast console.

Forecast SDK

Using the GetAccuracyMetrics Operation, specif y your `PredictorArn` to view the RMSE, MASE, MAPE, WAPE, Average wQL, and wQL metrics for each backtest.

```
{
    "PredictorArn": "arn:aws:forecast:region:acct-id:predictor/example-id"
}
```

Forecast Console

Choose your predictor on the **Predictors** page. Accuracy metrics for the predictor are shown in the **Predictor metrics** section.

> **Note**
>
> For Average wQL, wQL, RMSE, MASE, MAPE, and WAPE metrics, a lower value indicates a superior model.

Topics

Weighted Quantile Loss (wQL)

The Weighted Quantile Loss (wQL) metric measures the accuracy of a model at a specified quantile. It is particularly useful when there are different costs for underpredicting and overpredicting. By setting the weight (τ) of the wQL function, you can automatically incorporate differing penalties for underpredicting and overpredicting.

The loss function is calculated as follows.

$$\mathrm{wQL}[\tau] = 2\frac{\sum_{i,t}[\tau \max(y_{i,t} - q_{i,t}^{(\tau)}, 0) + (1-\tau)\max(q_{i,t}^{(\tau)} - y_{i,t}, 0)]}{\sum_{i,t}|y_{i,t}|}$$

Where:

τ - a quantile in the set {0.01, 0.02, ..., 0.99}

$q_{i,t}^{(\tau)}$ - the τ-quantile that the model predicts.

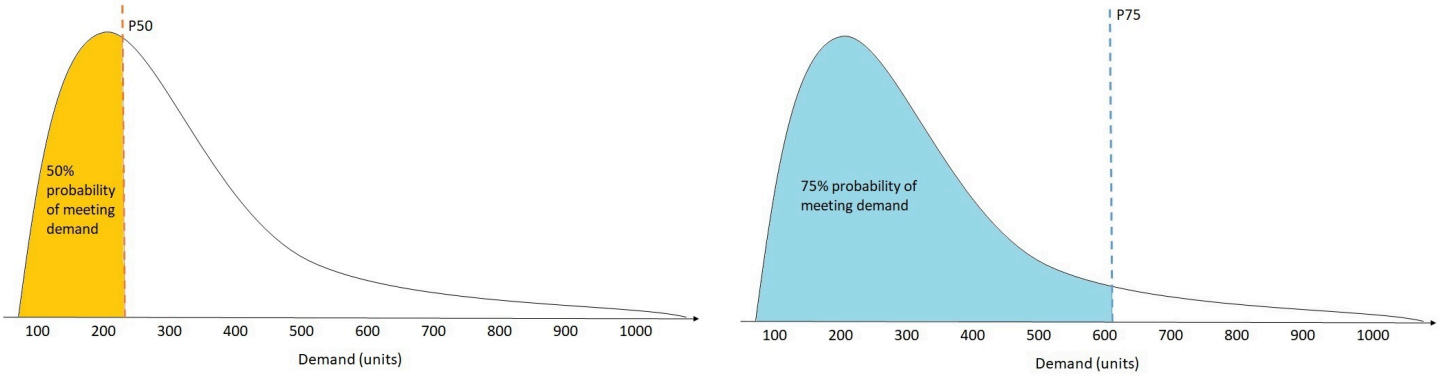
$y_{i,t}$ - the observed value at point (i,t)

The quantiles (τ) for wQL can range from 0.01 (P1) to 0.99 (P99). The wQL metric cannot be calculated for the mean forecast.

By default, Forecast computes wQL at `0.1` (P10), `0.5` (P50), and `0.9` (P90).

- **P10 (0.1)** - The true value is expected to be lower than the predicted value 10% of the time.
- **P50 (0.5)** - The true value is expected to be lower than the predicted value 50% of the time. This is also known as the median forecast.
- **P90 (0.9)** - The true value is expected to be lower than the predicted value 90% of the time.

In retail, the cost of being understocked is often higher than the cost of being overstocked, and so forecasting at P75 ($\tau = 0.75$) can be more informative than forecasting at the median quantile (P50). In these cases, wQL[0.75] assigns a larger penalty weight to underforecasting (0.75) and a smaller penalty weight to overforecasting (0.25).



The figure above shows the differing demand forecasts at wQL[0.50] and wQL[0.75]. The forecasted value at P75 is significantly higher than the forecasted value at P50 because the P75 forecast is expected to meet demand 75% of the time, whereas the P50 forecast is only expected to meet demand 50% of the time.

When the sum of observed values over all items and time points is approximately zero in a given backtest window, the weighted quantile loss expression is undefined. In these cases, Forecast outputs the unweighted quantile loss, which is the numerator in the wQL expression.

Forecast also calculates the average wQL, which is the mean value of weighted quantile losses over all specified quantiles. By default, this will be the average of wQL[0.10], wQL[0.50], and wQL[0.90].

Weighted Absolute Percentage Error (WAPE)

The Weighted Absolute Percentage Error (WAPE) measures the overall deviation of forecasted values from observed values. WAPE is calculated by taking the sum of observed values and the sum of predicted values, and calculating the error between those two values. A lower value indicates a more accurate model.

When the sum of observed values for all time points and all items is approximately zero in a given backtest window, the weighted absolute percentage error expression is undefined. In these cases, Forecast outputs the unweighted absolute error sum, which is the numerator in the WAPE expression.

$$\text{WAPE} = \frac{\sum_{i,t} |y_{i,t} - \hat{y}_{i,t}|}{\sum_{i,t} |y_{i,t}|}$$

Where:

- $y_{i,t}$ - the observed value at point (i,t)
- $\hat{y}_{i,t}$ - the predicted value at point (i,t)

Forecast uses the mean forecast as the predicted value, $\hat{y}_{i,t}$.

WAPE is more robust to outliers than Root Mean Square Error (RMSE) because it uses the absolute error instead of the squared error.

Amazon Forecast previously referred to the WAPE metric as the Mean Absolute Percentage Error (MAPE) and used the median forecast (P50) as the predicted value. Forecast now uses the mean forecast to calculate WAPE. The wQL[0.5] metric is equivalent to the WAPE[median] metric, as shown below:

$$\text{wQL}[0.5] = 2\frac{\sum_{i,t} 0.5[\max(y_{i,t} - q_{i,t}^{(0.5)}, 0) + \max(q_{i,t}^{(0.5)} - y_{i,t}), 0]}{\sum_{i,t} |y_{i,t}|} = \frac{\sum_{i,t} |y_{i,t} - q_{i,t}^{(0.5)}|}{\sum_{i,t} |y_{i,t}|}$$

Root Mean Square Error (RMSE)

Root Mean Square Error (RMSE) is the square root of the average of squared errors, and is therefore more sensitive to outliers than other accuracy metrics. A lower value indicates a more accurate model.

$$\mathrm{RMSE} = \sqrt{\frac{1}{nT}\sum_{i,t}(\hat{y}_{i,t} - y_{i,t})^2},$$

$$i = 1, \ldots, n$$

$$t = 1, \ldots T$$

Where:

- $y_{i,t}$ - the observed value at point (i,t)
- $\hat{y}_{i,t}$ - the predicted value at point (i,t)
- nT - the number of data points in a testing set

Forecast uses the mean forecast as the predicted value, $\hat{y}_{i,t}$. When calculating predictor metrics, nT is the number of data points in a backtest window.

RMSE uses the squared value of the residuals, which amplifies the impact of outliers. In use cases where only a few large mispredictions can be very costly, the RMSE is the more relevant metric.

Predictors created before November 11, 2020 calculated RMSE using the 0.5 quantile (P50) by default. Forecast now uses the mean forecast.

Mean Absolute Percentage Error (MAPE)

Mean Absolute Percentage Error (MAPE) takes the absolute value of the percentage error between observed and predicted values for each unit of time, then averages those values. A lower value indicates a more accurate model.

$$MAPE = \frac{1}{n}\sum_{t=1}^{n}\left|\frac{A_t - F_t}{A_t}\right|$$

Where:

A_t - the observed value at point *t*

F_t - the predicted value at point *t*

n - the number of data points in the time series

Forecast uses the mean forecast as the predicted value, F_t.

MAPE is useful for cases where values differ significantly between time points and outliers have a significant impact.

Mean Absolute Scaled Error (MASE)

Mean Absolute Scaled Error (MASE) is calculated by dividing the average error by a scaling factor. This scaling factor is dependent on the seasonality value, *m*, which is selected based on the forecast frequency. A lower value indicates a more accurate model.

$$\text{MASE} = \text{mean}\left(\frac{|e_j|}{\frac{1}{T-m}\sum_{t=m+1}^{T}|Y_t - Y_{t-m}|}\right) = \frac{\frac{1}{J}\sum_j |e_j|}{\frac{1}{T-m}\sum_{t=m+1}^{T}|Y_t - Y_{t-m}|}$$

Where:

Y_t - the observed value at point *t*

Y_{t-m} - the observed value at point *t-m*

e_j - the error at point *j* (observed value - predicted value)

m - the seasonality value

Forecast uses the mean forecast as the predicted value.

MASE is ideal for datasets that are cyclical in nature or have seasonal properties. For example, forecasting for items that are in high demand during summers and in low demand during winters can benefit from taking into account the seasonal impact.

Exporting Accuracy Metrics

> **Note**
>
> Export files can directly return information from the Dataset Import. This makes the files vulnerable to CSV injection if the imported data contains formulas or commands. For this reason, exported files can prompt security warnings. To avoid malicious activity, disable links and macros when reading exported files.

Forecast enables you to export forecasted values and accuracy metrics generated during backtesting.

You can use these exports to evaluate specific items at specific time points and quantiles, and better understand your predictor. The backtest exports are sent to a specified S3 location and contains two folders:

- **forecasted-values**: Contains CSV or Parquet files with forecasted values at each forecast type for each backtest.
- **accuracy-metrics-values**: Contains CSV or Parquet files with metrics for each backtest, along with the average across all backtests. These metrics include wQL for each quantile, Average wQL,RMSE, MASE, MAPE, and WAPE.

The `forecasted-values` folder contains forecasted values at each forecast type for each backtest window. It also includes information on item IDs, dimensions, timestamps, target values, and backtest window start and end times.

The `accuracy-metrics-values` folder contains accuracy metrics for each backtest window, as well as the average metrics across all backtest windows. It contains wQL metrics for each specified quantile, as well as Average wQL, RMSE, MASE, MAPE, and WAPE metrics.

Files within both folders follow the naming convention: `<ExportJobName>_<ExportTimestamp>_<PartNumber>.csv`.

You can export accuracy metrics using the Amazon Forecast Software Development Kit (SDK) and the Amazon Forecast console.

Forecast SDK

Using the `CreatePredictorBacktestExportJob` operation, specify your S3 location and IAM role in the `DataDestination` object, along with the `PredictorArn` and `PredictorBacktestExportJobName`.

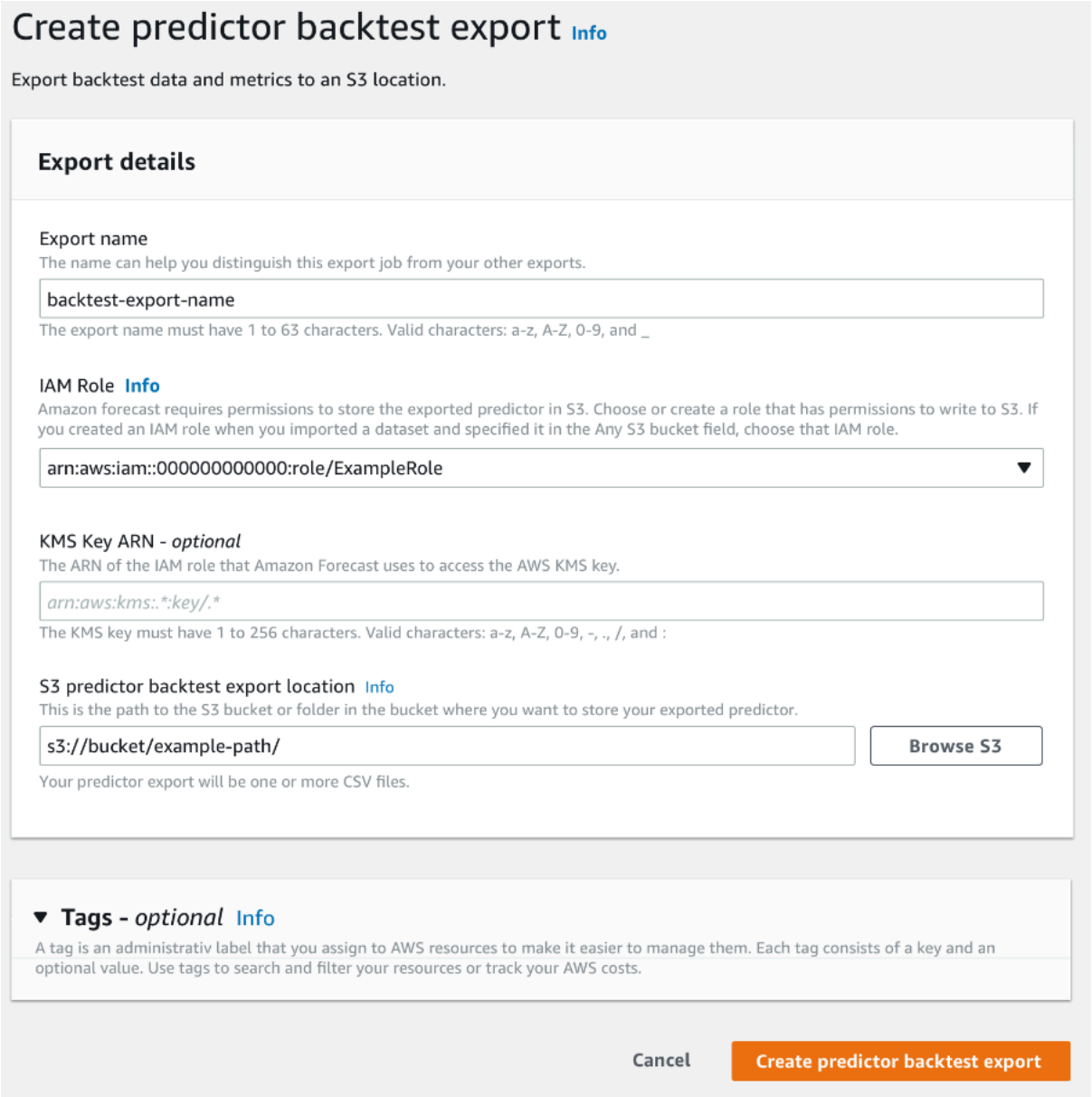
For example:

```
{
   "Destination": {
      "S3Config": {
         "Path": "s3://bucket/example-path/",
         "RoleArn": "arn:aws:iam::000000000000:role/ExampleRole"
      }
   },
   "Format": PARQUET;
   "PredictorArn": "arn:aws:forecast:region:predictor/example",
   "PredictorBacktestExportJobName": "backtest-export-name",
}
```

Forecast Console

Choose your predictor on the **Predictors** page. In the **Predictor metrics** section, choose **Export backtest results**.

During the **Create predictor backtest export** stage, set the **Export name**, **IAM Role**, and **S3 predictor backtest export location** fields.



Choosing Forecast Types

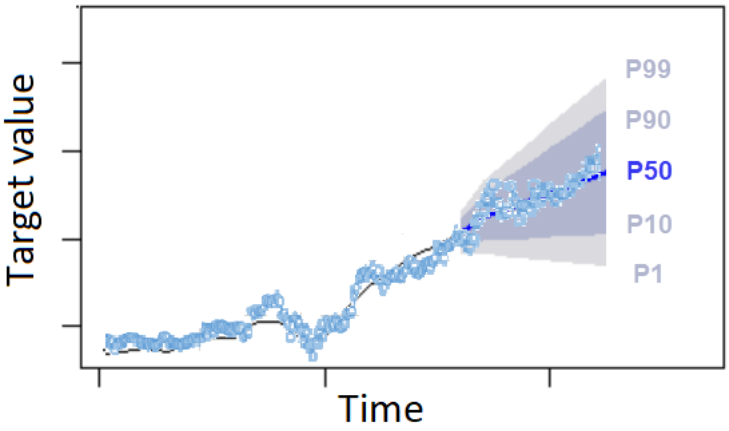
Amazon Forecast uses forecast types to create predictions and evaluate predictors. Forecast types come in two forms:

- **Mean forecast type** - A forecast using the mean as the expected value. Typically used as point forecasts for a given time point.
- **Quantile forecast type** - A forecast at a specified quantile. Typically used to provide a prediction interval, which is a range of possible values to account for forecast uncertainty. For example, a

forecast at the `0.65` quantile will estimate a value that is lower than the observed value 65% of the time.

By default, Forecast uses the following values for the predictor forecast types: `0.1` (P10), `0.5` (P50), and `0.9` (P90). You can choose up to five custom forecast types, including `mean` and quantiles ranging from `0.01` (P1) to `0.99` (P99).

Quantiles can provide an upper and lower bound for forecasts. For example, using the forecast types `0.1` (P10) and `0.9` (P90) provides a range of values known as an 80% confidence interval. The observed value is expected to be lower than the P10 value 10% of the time, and the P90 value is expected to be higher than the observed value 90% of the time. By generating forecasts at p10 and P90, you can expect the true value to fall between those bounds 80% of the time. This range of values is depicted by the shaded region between P10 and P90 in the figure below.



You can also use a quantile forecast as a point forecast when the cost of underpredicting differs from the cost of overpredicting. For example, in some retail cases the cost of being understocked is higher than the cost of being overstocked. In these cases, the forecast at 0.65 (P65) is more informative than the median (P50) or mean forecast.

When training a predictor, you can choose custom forecast types using the Amazon Forecast Software Development Kit (SDK) and Amazon Forecast console.

Forecast SDK

Using the `CreateAutoPredictor` operation, specify the custom forecast types in the `ForecastTypes` parameter. Format the parameter as an array of strings.

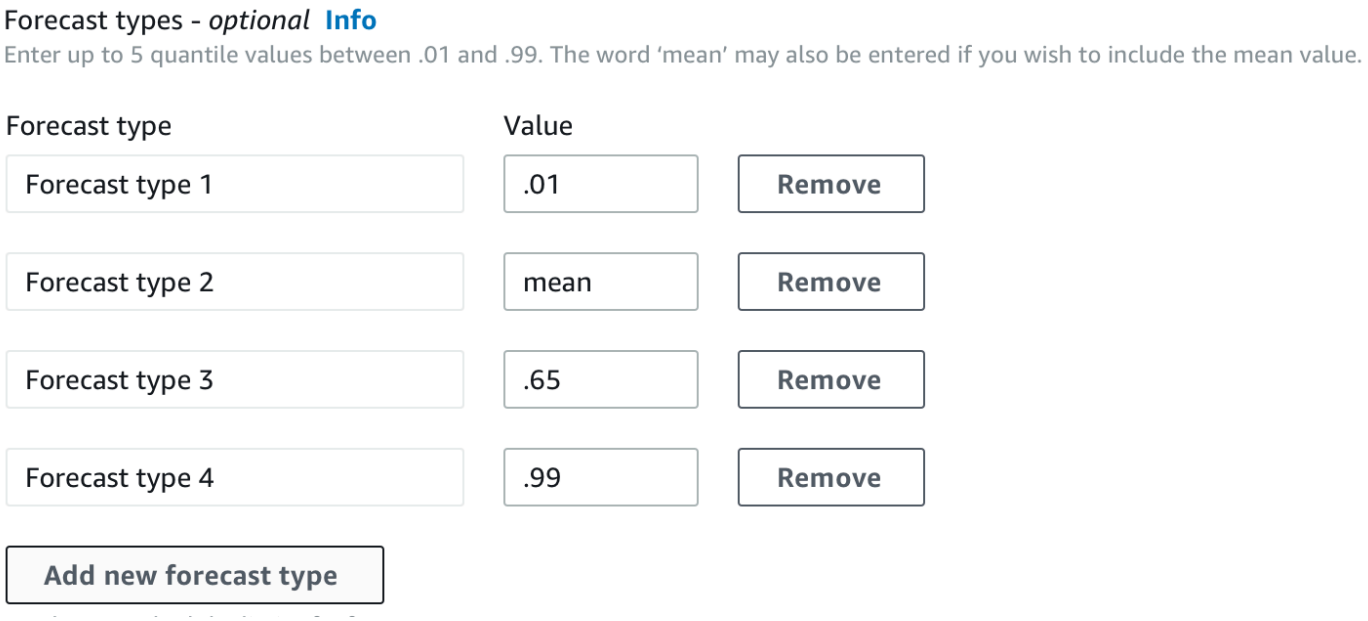
For example, to create a predictor at the `0.01`, `mean`, `0.65`, and `0.99` forecast types, use the following code.

```
{
    "ForecastTypes": [ "0.01", "mean", "0.65", "0.99" ],
},
```

Forecast Console

During the **Train Predictor** stage, specify the custom forecast types in the **Forecast types** field. Choose **Add new forecast type** and enter a forecast type value.

For example, to create a predictor using the `0.01`, `mean`, `0.65`, and `0.99` forecast types, enter the following values in the **Forecast types** fields shown below.



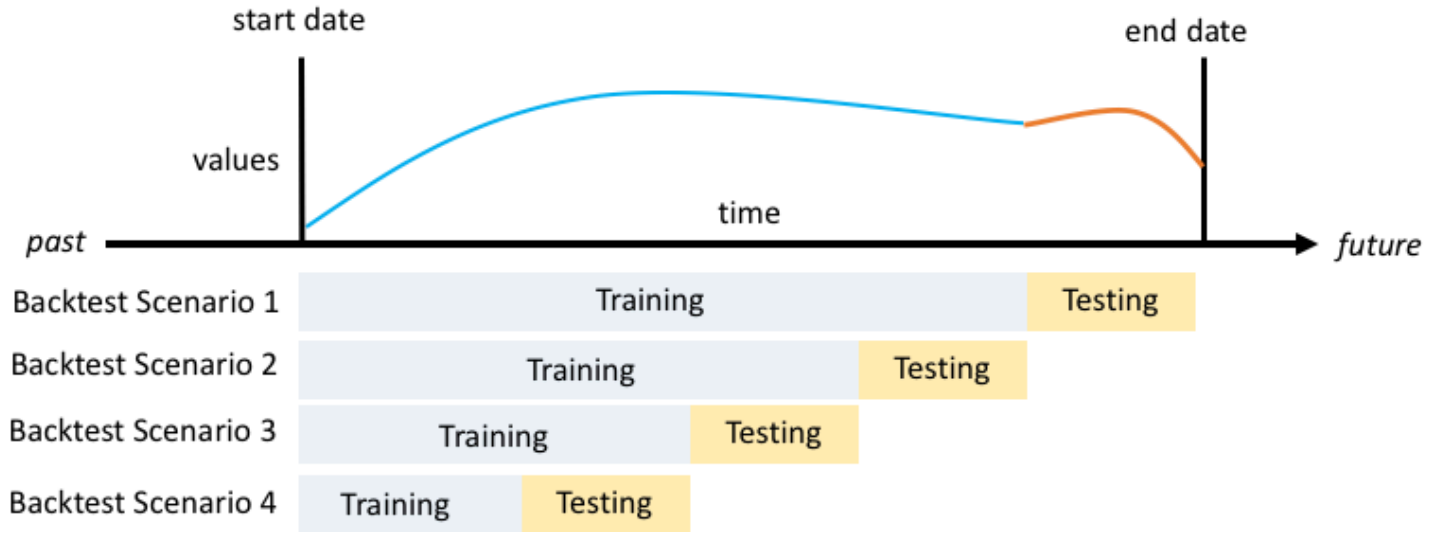
Working With Legacy Predictors

Setting Backtesting Parameters

Forecast uses backtesting to calculate accuracy metrics. If you run multiple backtests, Forecast averages each metric over all backtest windows. By default, Forecast computes one backtest, with the size of the backtest window (testing set) equal to the length of the forecast horizon (prediction window). You can set both the *backtest window length* and the *number of backtest scenarios* when training a predictor.

Forecast omits filled values from the backtesting process, and any item with filled values within a given backtest window will be excluded from that backtest. This is because Forecast only compares forecasted values with observed values during backtesting, and filled values are not observed values.

The backtest window must be at least as large as the forecast horizon, and smaller than half the length of the entire target time-series dataset. You can choose from between 1 and 5 backtests.



Generally, increasing the number of backtests produces more reliable accuracy metrics, since a larger portion of the time series is used during testing and Forecast is able to take an average of metrics across all backtests.

You can set the backtesting parameters using the Amazon Forecast Software Development Kit (SDK) and the Amazon Forecast console.

Forecast SDK

Using the CreatePredictor operation, set the backtest parameters in the EvaluationParameters datatype. Specify the length of the testing set during backtesting with the `BackTestWindowOffset` parameter, and the number of backtest windows with the `NumberOfBacktestWindows` parameter.

For example, to run 2 backtests with a testing set of 10 time points, use the follow code.

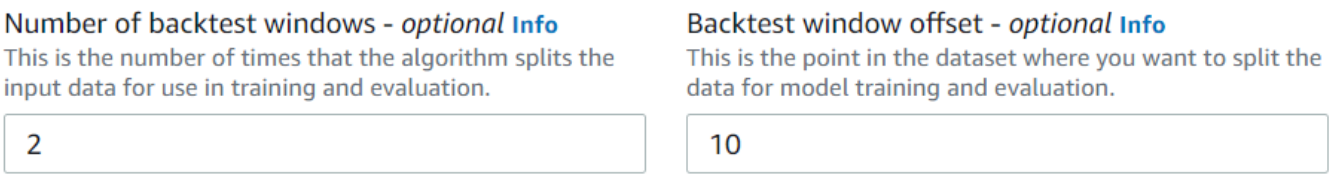
```
"EvaluationParameters": {
    "BackTestWindowOffset": 10,
    "NumberOfBacktestWindows": 2
```

```
}
```

Forecast Console

During the **Train Predictor** stage, set the length of the testing set during backtesting with the **Backtest window offset** field, and the number of backtest windows with the **Number of backtest windows** field.

For example, to run 2 backtests with a testing set of 10 time points, set the following values.



HPO and AutoML

By default, Amazon Forecast uses the `0.1` (P10), `0.5` (P50), and `0.9` (P90) quantiles for hyperparameter tuning during hyperparameter optimization (HPO) and for model selection during AutoML. If you specify custom forecast types when creating a predictor, Forecast uses those forecast types during HPO and AutoML.

If custom forecast types are specified, Forecast uses those specified forecast types to determine the optimal outcomes during HPO and AutoML. During HPO, Forecast uses the first backtest window to find the optimal hyperparameter values. During AutoML, Forecast uses the averages across all backtest windows and the optimal hyperparameters values from HPO to find the optimal algorithm.

For both AutoML and HPO, Forecast chooses the option that minimizes the average losses over the forecast types. You can also optimize your predictor during AutoML and HPO with one of the following accuracy metrics: Average Weighted Quantile loss (Average wQL), Weighted Absolute Percentage Error (WAPE), Root Mean Squared Error (RMSE), Mean Absolute Percentage Error (MAPE), or Mean Absolute Scaled Error (MASE).

You can choose an optimization metric using the Amazon Forecast Software Development Kit (SDK) and Amazon Forecast console.

Forecast SDK

Using the `CreatePredictor` operation, specify the custom forecast types in the `ObjectiveMetric` parameter.

The `ObjectiveMetric` parameter accepts the following values:

- `AverageWeightedQuantileLoss` - Average Weighted Quantile loss
- `WAPE` - Weighted Absolute Percentage Error
- `RMSE` - Root Mean Squared Error
- `MAPE` - Mean Absolute Percentage Error
- `MASE` - Mean Absolute Scaled Error

For example, to create a predictor with AutoML and optimize using the Mean Absolute Scaled Error (MASE) accuracy metric, use the following code.

```
{
    ...
    "PerformAutoML": "true",
    ...
    "ObjectiveMetric": "MASE",
},
```

Forecast Console

During the **Train Predictor** stage, choose **Automatic (AutoML)**. In the **Objective metric** section, choose the accuracy metric to use to optimize your predictor.

For example, the following image shows a predictor created with AutoML and optimized using the Mean Absolute Scaled Error (MASE) accuracy metric.

When using the console, you can only specify the Objective metric when you create a predictor using AutoML. If you manually select an algorithm, you cannot specify the Objective metric for HPO.

Retraining Predictors

> **Note**
>
> Retraining is only available for predictors created with AutoPredictor (`CreateAutoPredictor`). You can upgrade existing legacy predictors to AutoPredictor. See the section called "Upgrading to AutoPredictor".

Predictors can be retained with updated datasets to keep your predictors up to date. When retraining a predictor, Amazon Forecast maintains the same predictor configuration settings. After retraining, the original predictor will remain active and the retrained predictor will have a separate Predictor ARN.

Retraining a predictor can improve forecasting accuracy in two ways:

1. **More current data**: Your retrained predictor will incorporate more up-to-date data when training a model.
2. **Predictor improvements**: Your retrained predictor will incorporate any updates and improvements in the Amazon Forecast algorithms and additional datasets.

Retraining a predictor can be up to 50% faster than creating a new predictor from scratch. Predictor training times are faster and Forecast automatically uses your existing configuration settings.

> **Python notebooks**
>
> For a step-by-step guide on retraining predictors, see Retraining a predictor.

You can retrain a predictor using the Software Development Kit (SDK) or the Amazon Forecast console.

Console

To retrain a predictor

1. Sign in to the AWS Management Console and open the Amazon Forecast console at https://console.aws.amazon.com/forecast/.

2. In the navigation pane, choose **Predictors**.
3. Choose the predictor to retrain.
4. In the **Predictor actions** drop-down, choose **Retrain**.
5. Set a unique name for the upgraded predictor.
6. Choose **Retrain predictor**.

SDK

To retrain a predictor

Using the `CreateAutoPredictor` operation, assign the predictor a unique name and set the value of `ReferencePredictorArn` to the predictor you wish to retrain.

```
{
  "PredictorName": "RetrainedPredictor",
  "ReferencePredictorArn": "arn:aws:forecast:us-west-2:938097332257:predictor/
OriginalPredictor"
}
```

When retraining a predictor, assign values to only the `PredictorName` and `ReferencePredictorArn` parameters.

Weather Index

The Amazon Forecast Weather Index is a built-in featurization that incorporates historical and projected weather information into your model. It is especially useful for retail use cases, where temperature and precipitation can significantly affect product demand.

When the Weather Index is enabled, Forecast applies the weather featurization only to time series where it finds improvements in accuracy during predictor training. If supplementing a time series with weather information does not improve its predictive accuracy during backtesting, Forecast does not apply the Weather Index to that particular time series.

To apply the Weather Index, you must include a geolocation attribute in your target time series dataset and any related time series datasets. You also need to specify time zones for your target time-series timestamps. For more information regarding dataset requirements, see Conditions and Restrictions.

Python notebooks

For a step-by-step guide on using the Weather Index, see NY Taxi: Amazon Forecast with Weather Index.

Topics

Enabling the Weather Index

The Weather Index is enabled during the predictor training stage. When using the `CreateAutoPredictor` operation, the Weather Index is included in the AdditionalDataset data type.

Before enabling the Weather Index, you must include a geolocation attribute in your target time series and related timeseries datasets, and define the time zones for your timestamps. For more information, see Adding Geolocation Information and Specifying Time Zones.

You can enable the Weather Index using the Forecast console or the Forecast Software Development Kit (SDK) .

Console

To enable the Weather Index

1. Sign in to the AWS Management Console and open the Amazon Forecast console at https://console.aws.amazon.com/forecast/.
2. From **Dataset groups**, choose your dataset group.
3. In the navigation pane, choose **Predictors**.
4. Choose **Train new predictor**.
5. Choose **Enable Weather Index**.

SDK

To enable the Weather Index

Using the `CreateAutoPredictor` operation, enable the Weather Index by adding `"Name": "weather"` and `"Value": "true"` in the AdditionalDataset data type.

```
"DataConfig": {
    ...
    "AdditionalDatasets": [
        ...
        {
            "Name": "weather",
        }
        ]
    },
```

Adding Geolocation Information to Datasets

To use the Weather Index, you must include a geolocation attribute for each item in your target time series and related time series datasets. The attribute is defined with the `geolocation` attribute type within the dataset schemas.

All geolocation values in a dataset must be exclusively within a single region. The regions are: US (excluding Hawaii and Alaska), Canada, South America, Central America, Asia Pacific, Europe, and Africa & Middle East.

Specify the geolocation attribute in one of two formats:

- **Latitude & Longitude** (All regions) - Specify the latitude and longitude in decimal format (Example: 47.61_-122.33)
- **Postal code** (US only) - Specify the country code (US), followed by the 5-digit ZIP code (Example: US_98121)

The Latitude & Longitude format is supported for all regions. The Postal code format is only supported for the US region.

Topics

- Latitude & Longitude Bounds

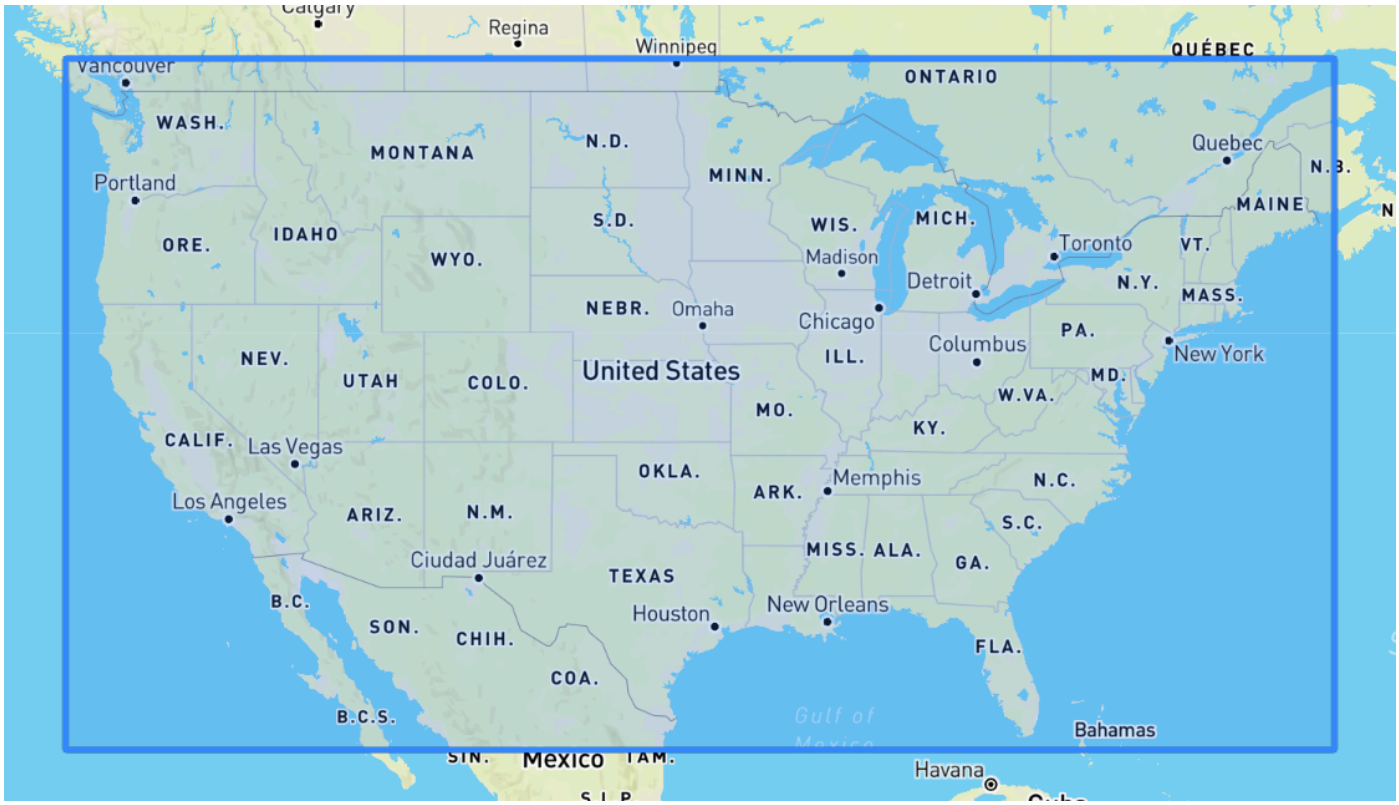
- Including Geolocation in the Dataset Schema
- Setting the Geolocation Format

Latitude & Longitude Bounds

The following are the latitudinal and longitudinal bounds for the accepted regions:

US Region

Bounds: latitude (24.6, 50.0), longitude (-126.0, -66.4).



Canada Region

Bounds: latitude (41.0, 75.0), longitude (-142.0, -52.0).

Europe Region

Bounds: latitude (34.8, 71.8), longitude (-12.6, 44.8).

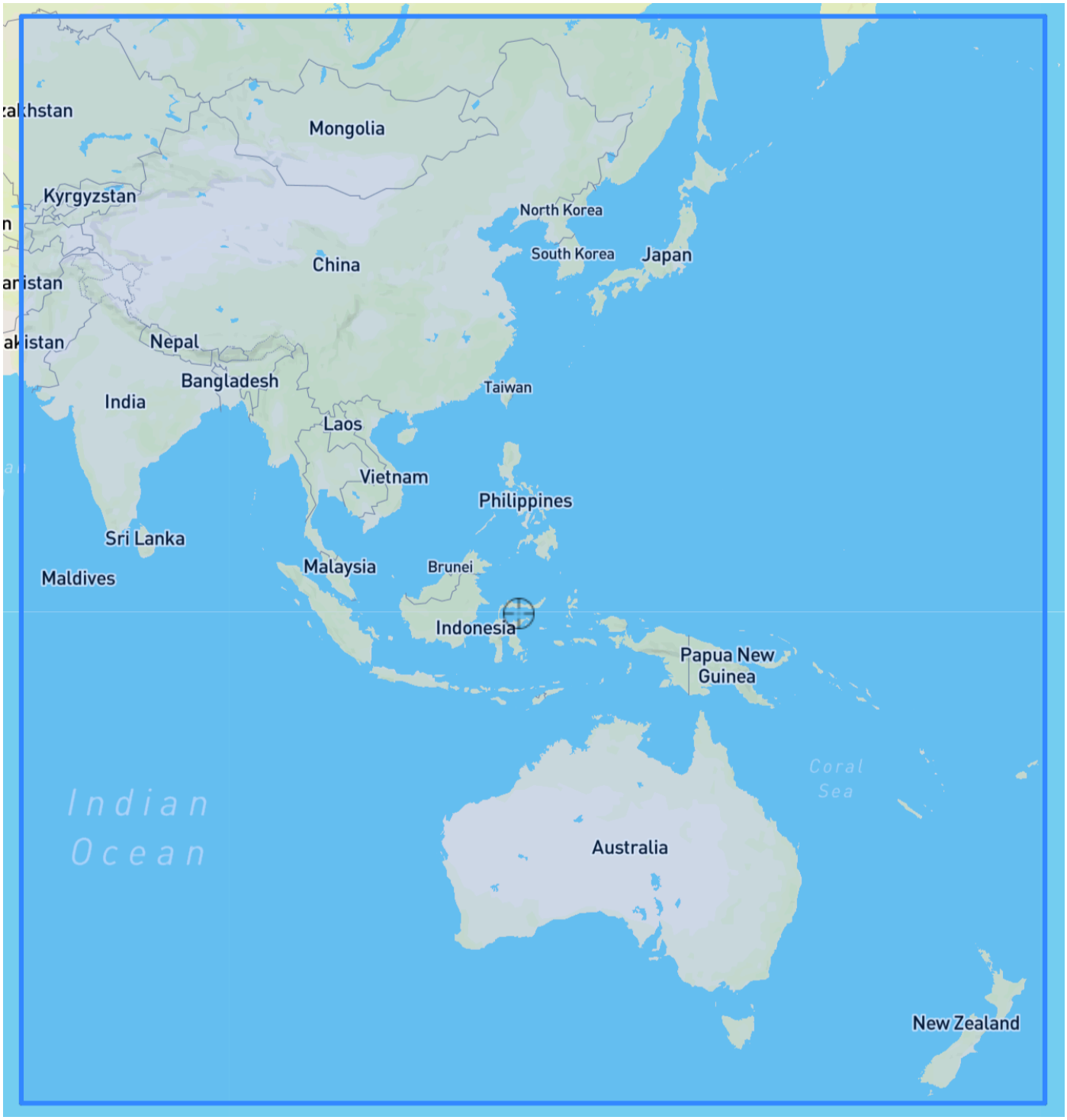
South America Region

Bounds: latitude (-56.6, 14.0), longitude (-82.4, -33.00).


Barbados
Cartagena
Caracas
Trinidad and Tobago
Panama
Venezuela
Medellín
Paramaribo
Guyana
Suriname
Colombia
RORAIMA
Macapa
Ecuador
São Luís
Manaus
AMAZONAS
PARÁ
MARANHÃO
CEARÁ
PERNAMBUCO
ACRE
Peru
RONDÔNIA
Brazil
SERGIPE
Lima
BAHIA
Cusco
Salvador
MATO GROSSO
Arequipa
GOIÁS
Bolivia
MINAS GERAIS
MATO GROSSO DO SUL
Iquique
Tarija
RIO DE JANEIRO
Paraguay
Antofagasta
PARANÁ
São Paulo
Chile
SANTA CATARINA
La Serena
Porto Alegre
Cordoba
Uruguay
Santiago
Buenos Aires
Talca
Montevideo
Argentina
Temuco
Puerto Montt
Comodoro Rivadavia
Falkland Islands (Islas Malvinas)
Punta Arenas
Grytviken
Scotia Sea

Asia Pacific Region

Bounds: latitude (-47.8, 55.0), longitude (67.0, 180.60).



Central America Region

Bounds: latitude (6.80, 33.20), longitude (-118.80, -58.20).

Africa & Middle East Region

Bounds: latitude (-35.60, 43.40), longitude (-18.80, -58.20).

Including Geolocation in the Dataset Schema

Using the console or CreateDataset operation, define the location attribute type as 'geolocation' within the JSON schema for the target time series and any related time series. The attributes in the schema must be ordered as they appear in the datasets.

```
{
 "Attributes":[
   {
```

```
        "AttributeName": "timestamp",
        "AttributeType": "timestamp"
    },
    {
        "AttributeName": "target_value",
        "AttributeType": "float"
    },
    {
        "AttributeName": "item_id",
        "AttributeType": "string"
    },
    {
        "AttributeName": "location",
        "AttributeType": "geolocation"
    }
  ]
}
```

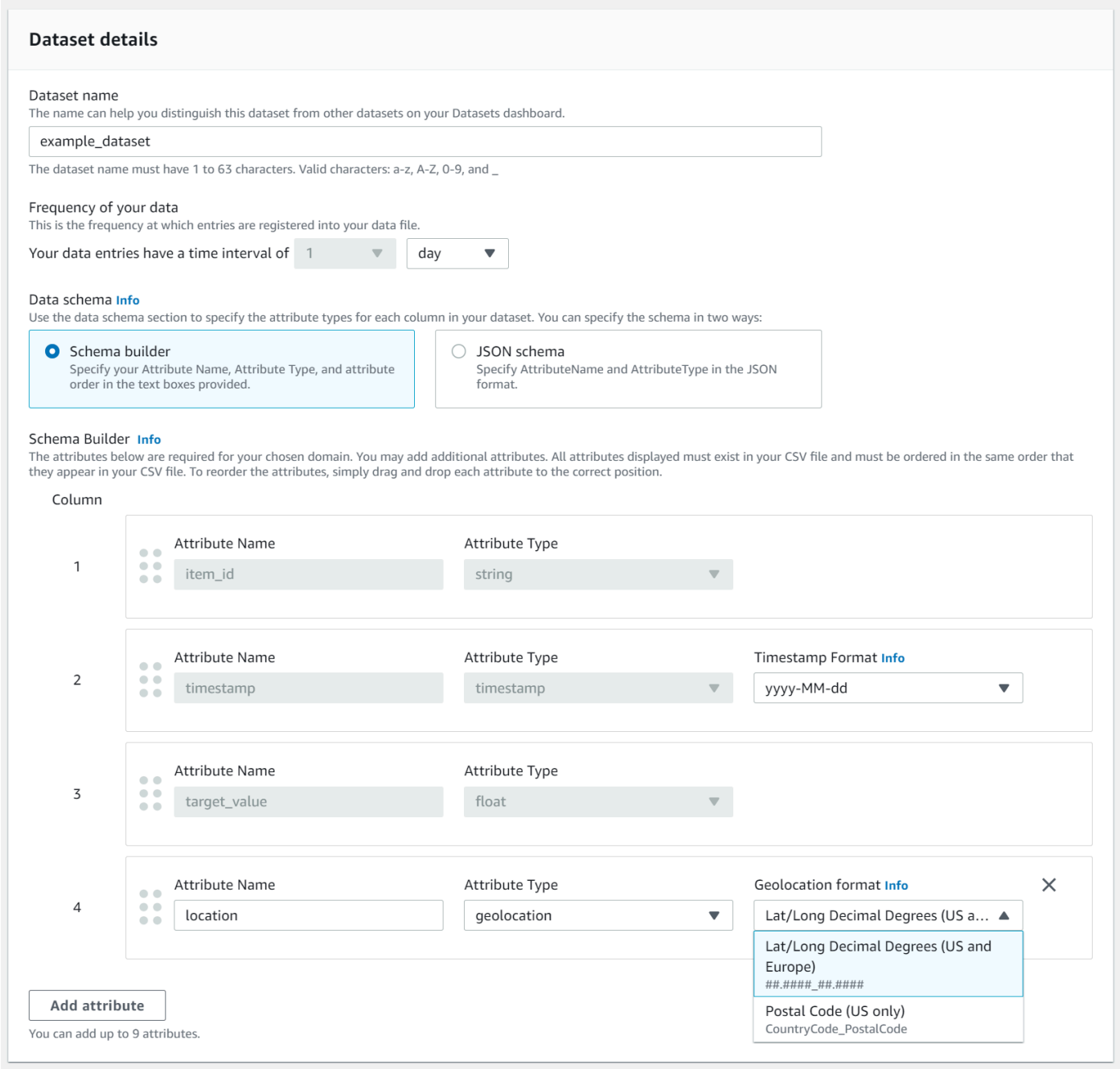
Setting the Geolocation Format

The format of the geolocation attribute can be in the **Postal Code** or **Latitude & Longitude** format. You can set the geolocation format using the Forecast console or the Forecast Software Development Kit (SDK).

Console

To add a geolocation attribute to a time series dataset

1. Sign in to the AWS Management Console and open the Amazon Forecast console at https://console.aws.amazon.com/forecast/.
2. Choose **Create dataset group**.
3. In the **Schema builder**, set your geolocation **Attribute type** to `geolocation`.
4. In the **Geolocation format** drop-down, choose your location format.



You can also define your attributes in JSON format and select a location format from the **Geolocation format** drop-down.

SDK

To add a geolocation attribute to a time series dataset

Using the CreateDatasetImportJob operation, set the value of `GeolocationFormat` to one of the following:

- **Latitude & longitude** (All regions): `"LAT_LONG"`
- **Postal code** (US Only): `"CC_POSTALCODE"`

For example, to specify the latitude & longitude format, include the following in `CreateDatasetImportJob` request:

```
{
    ...
    "GeolocationFormat": "LAT_LONG"
}
```

Specifying Time Zones

You can either let Amazon Forecast automatically synchronize your time zone information with your geolocation attribute, or you can manually assign a single time zone to your entire dataset.

Topics

Automatically Sync Time Zones with Geolocation

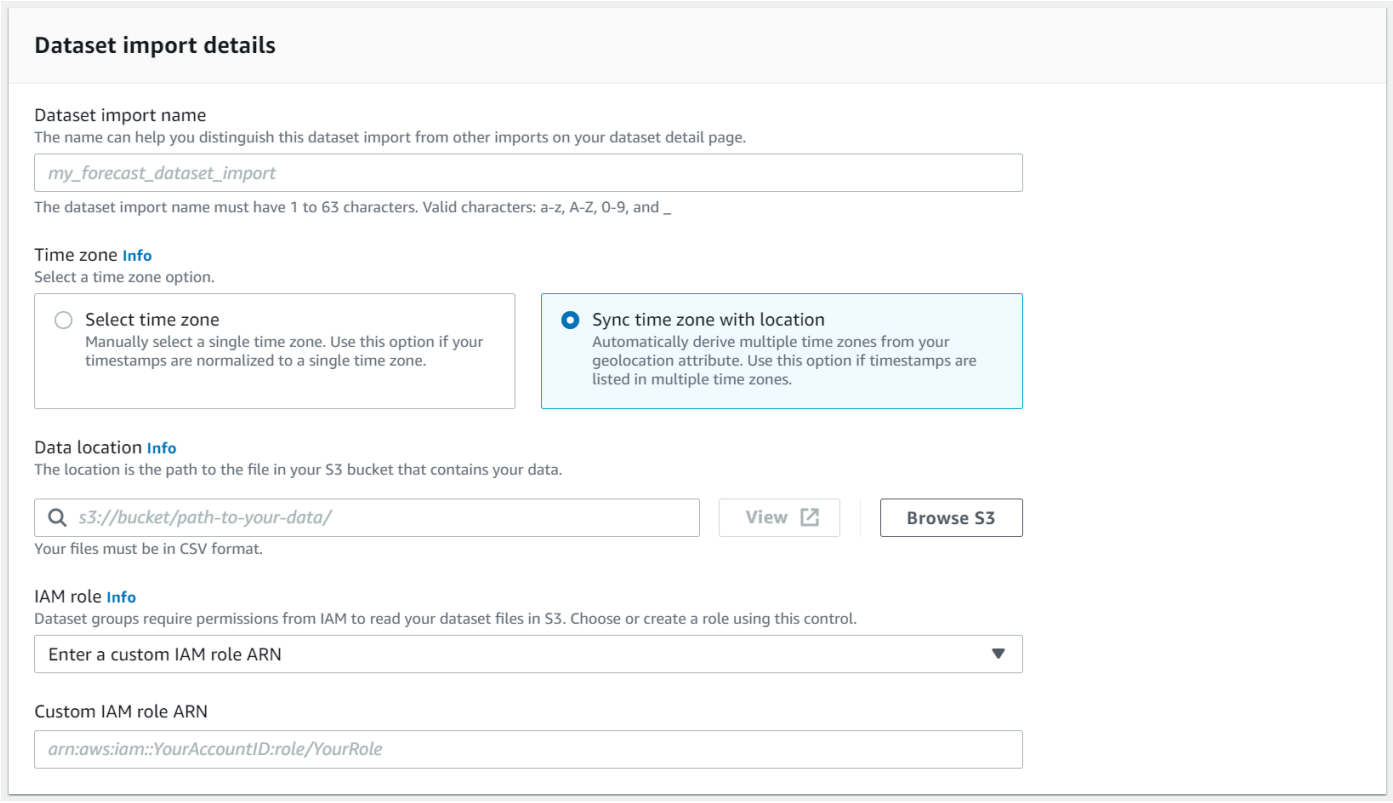
This option is ideal for datasets that contain timestamps in multiple time zones, and those timestamps are expressed in local time. Forecast assigns a time zone for every item in the target time series dataset based on the item's geolocation attribute.

You can automatically sync your timestamps with your geolocation attribute using the Forecast console or Forecast SDK.

Console

To sync time zones with the geolocation attribute

1. Sign in to the AWS Management Console and open the Amazon Forecast console at https://console.aws.amazon.com/forecast/.
2. In the navigation pane, choose **Create dataset group**.
3. In **Dataset import details**, choose **Sync time zone with location**.



SDK

To sync time zones with the geolocation attribute

Using the CreateDatasetImportJob operation, set `"UseGeolocationForTimeZone"` to `"true"`.

```
{
    ...
    "UseGeolocationForTimeZone": "true"
}
```

Manually Select a Single Time Zone

> **Note**
>
> You can manually select a time zone outside of the *US region*, *Canada region*, *South America region*, *Central America region*, *Asia Pacific region*, *Europe region*, and *Africa & Middle East region*. However, all geolocation values must still be within one of those regions.

This option is ideal for datasets with all timestamps within a single time zone, or if all timestamps are normalized to a single time zone. Using this option applies the same time zone to every item in the dataset.

The Weather Index accepts the following time zones:

US region

- America/Los_Angeles
- America/Phoenix
- America/Denver
- America/Chicago
- America/New_York

Canada region

- America/Vancouver
- America/Edmonton
- America/Regina
- America/Winnipeg
- America/Toronto
- America/Halifax
- America/St_Johns

Europe region

- Europe/London
- Europe/Paris
- Europe/Helsinki

South America region

- America/Buenos_Aires
- America/Noronha
- America/Caracas

Asia Pacific region

- Asia/Kabul
- Asia/Karachi
- Asia/Kolkata
- Asia/Kathmandu
- Asia/Dhaka
- Asia/Rangoon
- Asia/Bangkok
- Asia/Singapore
- Asia/Seoul
- Australia/Adelaide
- Australia/Melbourne
- Australia/Lord_Howe
- Australia/Eucla
- Pacific/Norfolk
- Pacific/Auckland

Central America

- America/Puerto_Rico

Africa & Middle East

- Africa/Nairobi
- Asia/Tehran
- Asia/Dubai

Other

- Pacific/Midway
- Pacific/Honolulu
- Pacific/Marquesas
- America/Anchorage
- Atlantic/Cape_Verde
- Asia/Anadyr
- Pacific/Chatham
- Pacific/Enderbury

- Pacific/Kiritimati

Select a time zone from the **Other** list if items in your dataset are located within one of the accepted region, but your timestamps are standardized to a time zone outside of that region.

For a complete list of valid time zone names, see Joda-Time library.

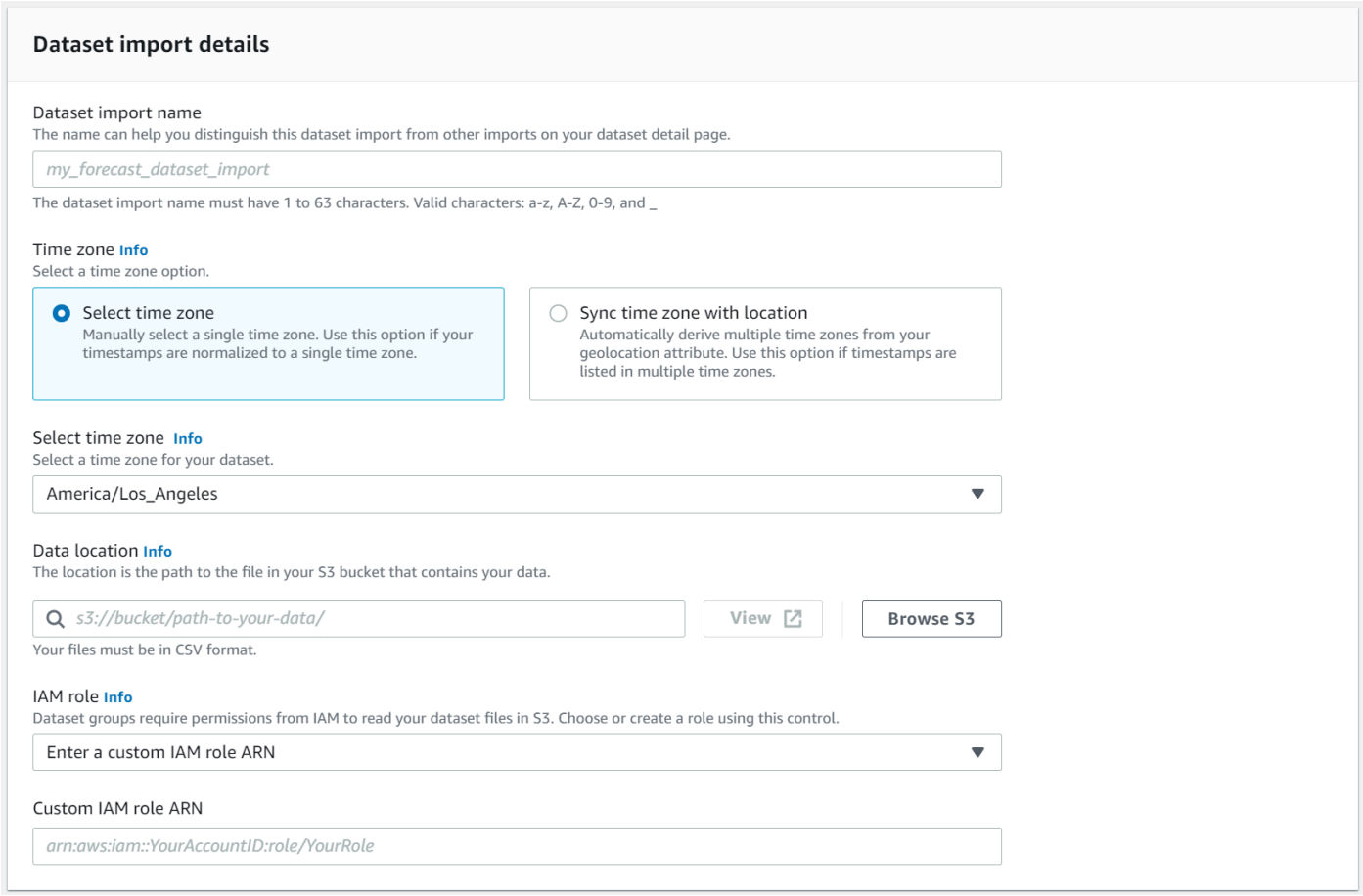
You can manually set a time zone for your datasets using the Forecast console or Forecast SDK.

Console

To select a single time zone for your dataset

1. Sign in to the AWS Management Console and open the Amazon Forecast console at https://console.aws.amazon.com/forecast/.
2. In the navigation pane, choose **Create dataset group**.
3. In **Dataset import details**, choose **Select time zone**.

For example, use the following to apply Los Angeles time (Pacific Standard Time) to your datasets.



SDK

To select a single time zone for your dataset

Using the CreateDatasetImportJob operation, set `"TimeZone"` to a valid time zone.

For example, use the following to apply Los Angeles time (Pacific Standard Time) to your datasets.

```
{
    ...
    "TimeZone": "America/Los_Angeles"
}
```

Conditions and Restrictions

The following conditions and restrictions apply when using the Weather Index:

- **Available algorithms**: If using a legacy predictor, the Weather Index can be enabled when you train a predictor with the CNN-QR, DeepAR+, and Prophet algorithms. The Weather Index is not applied to ARIMA, ETS, and NPTS.
- **Forecast frequency**: The valid forecast frequencies are `Minutely`, `Hourly`, and `Daily`.
- **Forecast horizon**: The forecast horizon cannot span further than 14 days into the future. For forecast horizon limits for each forecast frequency, refer to the list below:
  - `1 minute` - 500
  - `5 minutes` - 500
  - `10 minutes` - 500
  - `15 minutes` - 500
  - `Hourly` - 330
  - `Daily` - 14
- **Time series length**: When training a model with the Weather Index, Forecast truncates all time series datasets with timestamps before the start date of the Forecast weather dataset featurization. The Forecast weather dataset featurization contains the following start dates:
  - **US region**: July 2, 2018
  - **Europe region**: July 2, 2018
  - **Asia Pacific region**: July 2, 2018
  - **Canada region**: July 2, 2019
  - **South America region**: January 2, 2020
  - **Central America region**: September 2, 2020
  - **Africa & Middle East region**: March 25, 2021

  With the Weather Index enabled, data points with timestamps before the start date will not be used during predictor training.
- **Number of locations**: The target time series dataset cannot exceed 2000 unique locations.
- **Region bounds**: All items in your datasets must be located within a single region.
- **Minimum time series length**: Due to additional data requirements when testing the Weather Index, the minimum length for a time series dataset is:

  ```
  3 × ForecastHorizon + (BacktestWindows + 1) × BacktestWindowOffset
  ```

  If your time series datasets do not meet this requirement, consider decreasing the following:

- `ForecastHorizon` - Shorten your forecast horizon.
- `BacktestWindowOffset` - Shorten the length of the testing set during backtesting.
- `BacktestWindows` - Reduce the number of backtests.

Holidays Featurization

Holidays is a built-in featurization that incorporates a feature-engineered dataset of national holiday information into your model. It provides native support for the holiday calendars of over 250 countries. Amazon Forecast incorporates both the Holiday API library and Jollyday API to generate holiday calendars.

The Holidays featurization is especially useful in the retail domain, where public holidays can significantly affect demand.

The Holiday featurization supports a minimum forecast frequency of 5 minutes and a maximum of 1 month.

Topics

Enabling the Holidays Featurization

The Holidays featurization is included in Amazon Forecast as an Additional Dataset, and is enabled before training a predictor. It is recommended that your historical data contains at least two years of data. This allows Forecast to identify demand patters that are associated with specific holidays. After you choose a country, Holidays applies that country's holiday calendar to every item in your dataset during training.

You can enable Holidays using the Amazon Forecast console or the Forecast Software Development Kit (SDK).

Forecast SDK

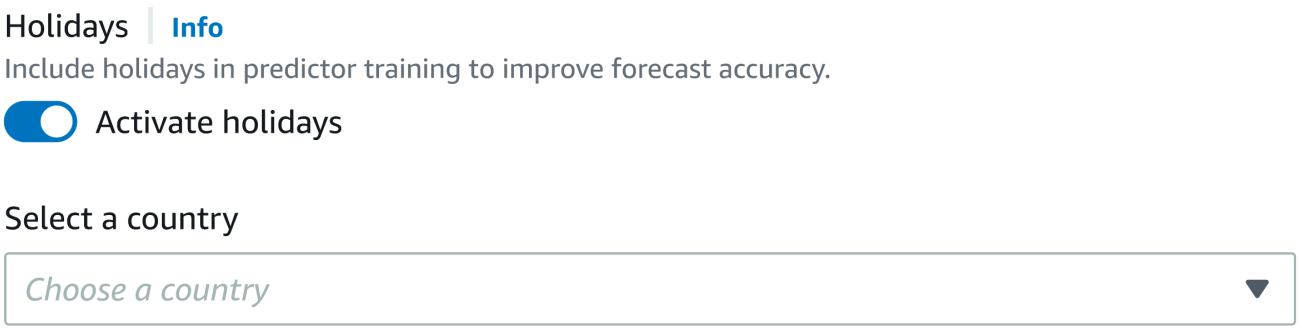
Using the `CreateAutoPredictor` operation, enable Holidays by adding `"Name": "holiday"` and setting `"Configuration"` to map `"CountryCode"` a two-letter country code. See the section called “Country Codes”.

For example, to include the USA holiday calendar, use the following code.

```
"DataConfig": {
  "AdditionalDatasets": [
      {
          "Name": "holiday",
          "Configuration": {
              "CountryCode" : ["US"]
          }
      },
    ]
  },
```

Forecast Console

Choose a country from the **Country for Holidays** drop-down during the **Train Predictor** stage.



Country Codes

Amazon Forecast provides native support for the public holiday calendars of the following countries. Use the **Country Code** when specifying a country with the API.

Supported Countries

Country	Country Code
Afghanistan	AF
Åland Islands	AX
Albania	AL
Algeria	DZ
American Samoa	AS
Andorra	AD
Angola	AO
Anguilla	AI
Antartica	AQ
Antigua and Barbuda	AG
Argentina	AR
Armenia	AM
Aruba	AW
Australia	AU
Austria	AT
Azerbaijan	AZ
Bahamas	BS
Bahrain	BH
Bangladesh	BD
Barbados	BB

Country	Country Code
Belarus	BY
Belgium	BE
Belize	BZ
Benin	BJ
Bermuda	BM
Bhutan	BT
Bolivia	BO
Bosnia and Herzegovina	BA
Botswana	BW
Bouvet Island	BV
Brazil	BR
British Indian Ocean Territory	IO
British Virgin Islands	VG
Brunei Darussalam	BN
Bulgaria	BG
Burkina Faso	BF
Burundi	BI
Cambodia	KH
Cameroon	CM
Canada	CA

Country	Country Code
Cape Verde	CV
Caribbean Netherlands	BQ
Cayman Islands	KY
Central African Republic	CF
Chad	TD
Chile	CL
China	CN
Christmas Island	CX
Cocos (Keeling) Islands	CC
Colombia	CO
Comoros	KM
Cook Islands	CK
Costa Rica	CR
Croatia	HR
Cuba	CU
Curaçao	CW
Cyprus	CY
Czechia	CZ
Democratic Republic of the Congo	CD
Denmark	DK

Country	Country Code
Djibouti	DJ
Dominica	DM
Dominican Republic	DO
Ecuador	EC
Egypt	EG
El Salvador	SV
Equatorial Guinea	GQ
Eritrea	ER
Estonia	EE
Eswatini	SZ
Ethiopia	ET
Falkland Islands	FK
Faroe Islands	FO
Fiji	FJ
Finland	FI
France	FR
French Guiana	GF
French Polynesia	PF
French Southern Territories	TF
Gabon	GA

Country	Country Code
Gambia	GM
Georgia	GE
Germany	DE
Ghana	GH
Gibraltar	GI
Greece	GR
Greenland	GL
Grenada	GD
Guadeloupe	GP
Guam	GU
Guatemala	GT
Guernsey	GG
Guinea	GN
Guinea-Bissau	GW
Guyana	GY
Haiti	HT
Heard Island and McDonald Islands	HM
Honduras	HN
Hong Kong	HK
Hungary	HU

Country	Country Code
Iceland	IS
India	IN
Indonesia	ID
Iran	IR
Iraq	IQ
Ireland	IE
Isle of Man	IM
Israel	IL
Italy	IT
Ivory Coast	CI
Jamaica	JM
Japan	JP
Jersey	JE
Jordan	JO
Kazakhstan	KZ
Kenya	KE
Kiribati	KI
Kosovo	XK
Kuwait	KW
Kyrgyzstan	KG

Country	Country Code
Laos	LA
Latvia	LV
Lebanon	LB
Lesotho	LS
Liberia	LR
Libya	LY
Liechtenstein	LI
Lithuania	LT
Luxembourg	LU
Macao	MO
Madagascar	MG
Malawi	MW
Malaysia	MY
Maldives	MV
Mali	ML
Malta	MT
Marshall Islands	MH
Martinique	MQ
Mauritania	MR
Mauritius	MU

Country	Country Code
Mayotte	YT
Mexico	MX
Micronesia	FM
Moldova	MD
Monaco	MC
Mongolia	MN
Montenegro	ME
Montserrat	MS
Morocco	MA
Mozambique	MZ
Myanmar	MM
Namibia	NA
Nauru	NR
Nepal	NP
Netherlands	NL
New Caledonia	NC
New Zealand	NZ
Nicaragua	NI
Niger	NE
Nigeria	NG

Country	Country Code
Niue	NU
Norfolk Island	NF
North Korea	KP
North Macedonia	MK
Northern Mariana Islands	MP
Norway	NO
Oman	OM
Pakistan	PK
Palau	PW
Palestine	PS
Panama	PA
Papua New Guinea	PG
Paraguay	PY
Peru	PE
Philippines	PH
Pitcairn Islands	PN
Poland	PL
Portugal	PT
Puerto Rico	PR
Qatar	QA

Country	Country Code
Republic of the Congo	CG
Réunion	RE
Romania	RO
Russian Federation	RU
Rwanda	RW
Saint Barthélemy	BL
"Saint Helena, Ascension and Tristan da Cunha "	SH
Saint Kitts and Nevis	KN
Saint Lucia	LC
Saint Martin	MF
Saint Pierre and Miquelon	PM
Saint Vincent and the Grenadines	VC
Samoa	WS
San Marino	SM
Sao Tome and Principe	ST
Saudi Arabia	SA
Senegal	SN
Serbia	RS
Seychelles	SC
Sierra Leone	SL

Country	Country Code
Singapore	SG
Sint Maarten	SX
Slovakia	SK
Slovenia	SI
Solomon Islands	SB
Somalia	SO
South Africa	ZA
South Georgia and the South Sandwich Islands	GS
South Korea	KR
South Sudan	SS
Spain	ES
Sri Lanka	LK
Sudan	SD
Suriname	SR
Svalbard and Jan Mayen	SJ
Sweden	SE
Switzerland	CH
Syrian Arab Republic	SY
Taiwan	TW
Tajikistan	TJ

Country	Country Code
Tanzania	TZ
Thailand	TH
Timor-Leste	TL
Togo	TG
Tokelau	TK
Tonga	TO
Trinidad and Tobago	TT
Tunisia	TN
Turkey	TR
Turkmenistan	TM
Turks and Caicos Islands	TC
Tuvalu	TV
Uganda	UG
Ukraine	UA
United Arab Emirates	AE
United Kingdom	GB
United Nations	UN
United States	US
United States Minor Outlying Islands	UM
United States Virgin Islands	VI

Country	Country Code
Uruguay	UY
Uzbekistan	UZ
Vanuatu	VU
Vatican City	VA
Venezuela	VE
Vietnam	VN
Wallis and Futuna	WF
Western Sahara	EH
Yemen	YE
Zambia	ZM
Zimbabwe	ZW

Additional Holiday Calendars

Amazon Forecast also supports holidays for India, Korea, and United Arab Emirates. Their holidays are listed below.

India - "IN"

- January 26 - Republic Day
- August 15 - Independence Day
- October 2 - Gandhi Jayanti

Korea - "KR"

- January 1 - New Year
- March 1 - Independence Movement Day

May 5 - Children's Day

June 6 - Memorial Day

August 15 - Liberation Day

October 3 - National Foundation Day

October 9 - Hangul Day

December 25 - Christmas Day

United Arab Emirates - "AE"

January 1 - New Year

December 1 - Commemoration Day

December 2-3 - National Day

Ramadan*

Eid al-Fitr*

Eid al-Adha*

Islamic New Year*

*Islamic Holidays are determined by lunar cycles.

Predictor Explainability

Predictor Explainability helps you better understand how the attributes in your datasets impact your target variable. Forecast uses a metric called Impact scores to quantify the relative impact of each attribute and determine whether they increase or decrease forecast values.

For example, consider a forecasting scenario where the target is `sales` and there are two related attributes: `price` and `color`. Forecast may find that an item's price significantly impacts sales (high Impact score), while the item's color has a negligible effect (low Impact score).

To enable Predictor Explainability, your predictor must include at least one of the following: related time series, item metadata, or additional datasets like Holidays and the Weather Index. See Restrictions and best practices for more information.

To create Impact scores for specific time series and time points, use Forecast Explainability instead of Predictor Explainability. See Forecast Explainability.

Topics

Interpreting Impact Scores

Impact scores measure the relative impact attributes have on forecast values. For example, if the 'price' attribute has an impact score that is twice as large as the 'store location' attribute, you can conclude that the price of an item has twice the impact on forecast values than the store location.

Impact scores also provide information on whether attributes increase or decrease forecast values. In the console, this is denoted by the two graphs. Attributes with blue bars increase forecast values, while attributes with red bars decrease forecast values.

In the console, Impact scores range from 0 to 1, where a score of 0 denotes no impact and a score close to 1 denotes a significant impact. In the SDKs, Impact scores range from -1 to 1, where the sign denotes the direction of the impact.

It is important to note that Impact scores measure the relative impact of attributes, not the absolute impact. Therefore, Impact scores cannot be used to determine whether particular attributes improve model accuracy. If an attribute has a low Impact score, that does not necessarily mean that it has a low impact on forecast values; it means that it has a lower impact on forecast values than other attributes used by the predictor.

Creating Predictor Explainability

> **Note**
>
> You can create a maximum of one Predictor Explainability per predictor

When you enable Predictor Explainability, Amazon Forecast calculates Impact scores for all attributes in your datasets. The Impact scores can be interpreted as the impact attributes have on overall forecast values. You can enable Predictor Explainability when you create a predictor, or you can enable the feature after creating the predictor.

Enabling Predictor Explainability for a new predictor

Enabling Predictor Explainability when creating a new predictor will create both a Predictor resource and Explainability resource. You can enable Predictor Explainability for a new predictor using the Software Development Kit (SDK) or the Amazon Forecast console.

Console

To enable Predictor Explainability

1. Sign in to the AWS Management Console and open the Amazon Forecast console at https://console.aws.amazon.com/forecast/.
2. From **Dataset groups**, choose your dataset group.
3. In the navigation pane, choose **Predictors**.
4. Choose **Train new predictor**.
5. In the **Predictor configuration** section, choose **Enable Explainability**.
6. Provide values for the following mandatory fields:
   - **Name** - a unique predictor name.
   - **Forecast frequency** - the granularity of your forecasts.
   - **Forecast horizon** - The number of time steps to forecast.

7. Choose **Start**

Python

To enable explainability for a new predictor with the SDK for Python (Boto3), use the `create_auto_predictor` method and set ExplainPredictor to true.

The following code creates an auto predictor that makes predictions for 24 (`ForecastHorizon`) days (`ForecastFrequency`) in the future, and has `ExplainPredictor` set to true. For information on required and optional parameters see CreateAutoPredictor.

```
import boto3

forecast = boto3.client('forecast')

create_predictor_response = forecast.create_auto_predictor(
    PredictorName = 'predictor_name',
    ForecastHorizon = 24,
    ForecastFrequency = 'D',
    DataConfig = {
        "DatasetGroupArn": "arn:aws:forecast:region:account:dataset-
group/datasetGroupName"
    },
    ExplainPredictor = True
)
```

Enabling Predictor Explainability for an existing predictor

Enabling Predictor Explainability for an existing predictor will create a Explainability resource for that resource. You can only create an Explainability resource for predictors that do not already contain an Explainability resource. To view Impact scores for an updated dataset, retrain or recreate the predictor with the updated data.

You can enable Predictor Explainability for a new predictor using the Software Development Kit (SDK) or the Amazon Forecast console.

Console

To enable Predictor Explainability

1. Sign in to the AWS Management Console and open the Amazon Forecast console at https://console.aws.amazon.com/forecast/.
2. From **Dataset groups**, choose your dataset group.
3. In the navigation pane, choose **Predictors**.
4. Choose your predictor.
5. In the **Predictor Explainability** section, choose **Enable Explainability**.
6. Provide a unique name for the Predictor Explainability.
7. Choose **Start**

Python

To enable Predictor Explainability for an existing predictor with the SDK for Python (Boto3), use the `create_explainability` method. Specify a name for the explainability, the ARN for the predictor, and for `ExplainabilityConfig`, set both `TimePointGranularity` and `TimeSeriesGranularity` to **ALL**. To create an Explainability visualization that is viewable within the console, set `EnableVisualization` to **True**.

For information on required and optional parameters see CreateExplainability.

```
import boto3

forecast = boto3.client('forecast')

create_explainability_response = forecast.create_explainability(
    ExplainabilityName = 'explainability_name',
    ResourceArn = 'arn:aws:forecast:region:accountNumber:predictor/predictorName',
    ExplainabilityConfig = {
      "TimePointGranularity": "ALL",
      "TimeSeriesGranularity": "ALL"
    },
    EnableVisualization = True
)
```

Exporting Predictor Explainability

> **Note**
>
> Export files can directly return information from the Dataset Import. This makes the files vulnerable to CSV injection if the imported data contains formulas or commands. For this reason, exported files can prompt security warnings. To avoid malicious activity, disable links and macros when reading exported files.

Forecast enables you to export a CSV or Parquet file of Impact scores to an S3 location. The Impact scores range from -1 to 1, where the sign denotes the direction of the impact. You can export Impact scores using the Amazon Forecast Software Development Kit (SDK) and the Amazon Forecast console.

	A	B	C	D
1	Price-NormalizedImpactScore	Promotion-NormalizedImpactScore	WeatherIndex-NormalizedImpactScore	Holiday_US-NormalizedImpactScore
2	-0.97	0.1	0.87	0.23

Console

To export Predictor Explainability

1. Sign in to the AWS Management Console and open the Amazon Forecast console at https://console.aws.amazon.com/forecast/.
2. From **Dataset groups**, choose your dataset group.
3. In the navigation pane, choose **Predictors**.
4. Choose your predictor.
5. In the **Predictor Explainability** section, choose **Export**.
6. For the **Export name** field, provide a unique name for the export.
7. For the **S3 explainability export location** field, provide an S3 location to export the CSV file.
8. For the **IAM Role** field, provide a role with access to the specified S3 location.
9. Choose **Create export.**

Python

To export a Predictor Explainability with the SDK for Python (Boto3), use the `create_explainability_export` method. Give the job a name, specify the ARN of the explainability, and, in the `Destination` object, specify your Amazon S3 destination location, and IAM service role.

For information on required and optional parameters see CreateExplainabilityExport.

```
import boto3

forecast = boto3.client('forecast')

export_response = forecast.create_explainability_export(
    Destination = {
        "S3Config": {
            "Path": "s3://bucketName/filename.csv",
            "RoleArn": "arn:aws:iam::accountNumber:role/roleName"
        }
    },
    ExplainabilityArn =
 'arn:aws:forecast:region:accountNumber:explainability/explainabilityName',
    ExplainabilityExportName = 'job_name'
)
```

Restrictions and best practices

Consider the following restrictions and best practices when working with Predictor Explainability.

- **Predictor Explainability is only available for some predictors created with AutoPredictor** - You cannot enable Explainability for legacy predictors that were created with AutoML or through manual selection. See Upgrading to AutoPredictor.
- **Predictor Explainability is not available for all models** - The ARIMA (AutoRegressive Integrated Moving Average), ETS (Exponential Smoothing State Space Model), and NPTS (Non-Parametric Time Series) models do not incorporate external time series data. Therefore, these models do not create an explainability report, even if you include the additional datasets.
- **Explainability requires attributes** - Your predictor must include at least one of the following: related time series, item metadata, Holidays, or the Weather Index.

- **Predictors are limited to one Explainability resource** - You cannot create multiple Explainability resources for a predictor. If you are interested in Impact scores for an updated dataset, retrain your predictor.
- **Impact scores of zero denote no impact**- If an attribute has an impact score of 0, then that attribute has no significant impact on forecast values.
- **Retrying failed Predictor Explainability jobs** - If Forecast successfully creates a Predictor but the Predictor Explainability job fails, you can retry creating Predictor Explainability in the console or with the CreateExplainability operation.
- **You cannot create Impact scores for specific time points and time series** - To view Impact scores for specific time points and time series, see Forecast Explainability.
- **Predictor Explainability visualizations are available for 90 days after creation** - To view the visualization after 90 days, retrain the predictor.

Predictor Monitoring

> **Note**
>
> If you enable predictor monitoring, Amazon Forecast will store data from each of your forecasts for predictor performance analysis, even after deleting forecast data. To delete this data, delete the monitoring resource.

Predictor monitoring allows you to see how your predictor's performance changes over time. A variety of factors can cause performance changes, such as economic developments or changes in your customer's behavior.

For example, consider a forecasting scenario where the target is `sales` and there are two related attributes: `price` and `color`. In the months after creating your first predictor, certain colors might unexpectedly become more popular with your customers. This might drive up sales for items with this attribute. This new data could impact your predictor's performance and the accuracy of the forecasts it generates.

With predictor monitoring enabled, Forecast analyzes your predictor's performance as you generate forecasts and import more data. Forecast compares the new data to the earlier forecasts to detect any changes in performance. You can view graphs of how different accuracy metrics

have changed over time in the Forecast console. Or you can get monitoring results with the ListMonitorEvaluations operation.

Predictor monitoring can help decide if it is time to retrain your predictor. If performance is degrading, you might want to retrain the predictor on more recent data. If you choose to retrain your predictor, the new predictor will include the monitoring data from the previous one. You might also use predictor monitoring to gather contextual data about your production environment, or to perform comparisons for different experiments.

Predictor monitoring is only available for AutoPredictors. You can upgrade existing legacy predictors to AutoPredictor. See Upgrading to AutoPredictor.

Topics

Predictor Monitoring Workflow

To get predictor monitoring results, you must first use your predictor to generate a forecast and then import more data. The monitoring workflow is as follows.

1. Enable predictor monitoring for an auto predictor:
   - Create a new predictor with monitoring enabled. See Enabling Predictor Monitoring for a New Predictor.
   - Or enable monitoring for an existing predictor. See Enabling Predictor Monitoring for an Existing Predictor.
2. Use the predictor to generate one or more forecasts.
3. Import more data. For information about importing data into Forecast, see Importing Datasets.
4. View predictor monitoring results:
   - You can view results on the **Monitoring** tab for your predictor.
   - Or you can get monitoring results with the ListMonitorEvaluations operation.

   For more information, see Viewing Monitoring Results.

Enabling Predictor Monitoring

You can enable predictor monitoring when you create the predictor, or you can enable it for an existing predictor.

> **Note**
>
> Predictor monitoring is only available for AutoPredictors. You can upgrade existing legacy predictors to AutoPredictor. See Upgrading to AutoPredictor.

Topics

Enabling Predictor Monitoring for a New Predictor

You can enable predictor monitoring for a new predictor with the console, AWS CLI, AWS SDKs, and the CreateAutoPredictor operation.

Console

To enable Predictor monitoring

1. Sign in to the AWS Management Console and open the Amazon Forecast console at https://console.aws.amazon.com/forecast/.
2. From **Dataset groups**, choose your dataset group.
3. In the navigation pane, choose **Predictors**.
4. Choose **Train new predictor**.
5. In the **Predictor configuration** section, choose **Enable monitoring**.
6. Provide values for the following mandatory fields:
   - **Name** - a unique predictor name.
   - **Forecast frequency** - the granularity of your forecasts.
   - **Forecast horizon** - The number of time steps to forecast.
7. Choose **Start** to create an auto predictor with monitoring enabled. You'll see monitoring results as you use the predictor to generate forecasts and then import more data.

Python

To enable predictor monitoring for a new predictor with the SDK for Python (Boto3), use the `create_auto_predictor` method and provide a monitor name in the `MonitoringConfig`.

The following code creates an auto predictor that makes predictions for 24 (`ForecastHorizon`) days (`ForecastFrequency`) in the future, and specifies `MyPredictorMonitor` as the `MonitorName`. After you generate a forecast and then import more data, you can view the results of predictor monitoring. For more information about retrieving results, see Viewing Monitoring Results.

For information on required and optional parameters for creating a predictor see CreateAutoPredictor.

```
import boto3

forecast = boto3.client('forecast')

create_predictor_response = forecast.create_auto_predictor(
    PredictorName = 'predictor_name',
    ForecastHorizon = 24,
    ForecastFrequency = 'D',
    DataConfig = {
        "DatasetGroupArn": "arn:aws:forecast:region:account:dataset-
group/datasetGroupName"
    },
    MonitorConifg = {
        "MonitorName": "MyMonitorName"
    }
)
```

Enabling Predictor Monitoring for an Existing Predictor

You can enable predictor monitoring for an existing predictor with the console, AWS CLI, and AWS SDKs.

Console

To enable predictor monitoring

1. Sign in to the AWS Management Console and open the Amazon Forecast console at https://console.aws.amazon.com/forecast/.
2. From **Dataset groups**, choose your dataset group.
3. In the navigation pane, choose **Predictors**.
4. Choose your predictor.
5. Navigate to the **Monitoring** tab.
6. In the **Monitoring details** section, choose **Start monitoring**

   When the **Monitoring status** is Active, predictor monitoring is enabled. After you generate a forecast and then import more data, you can view the results of predictor monitoring. For more information see Viewing Monitoring Results

Python

To enable predictor monitoring for an existing predictor with the SDK for Python (Boto3), use the `create_monitor` method. Specify a name for the monitoring, and for `ResourceArn` specify the Amazon Resource Name (ARN) for the predictor to monitor. Use the `describe_monitor` method and provide the monitor ARN to get the status of the monitor. After you generate a forecast and then import more data, you can view the results of predictor monitoring. For more information see Viewing Monitoring Results.

For information on required and optional parameters, see the CreateMonitor and DescribeMonitor.

```
import boto3

forecast = boto3.client('forecast')

create_monitor_response = forecast.create_monitor(
    MonitorName = 'monitor_name',
    ResourceArn = 'arn:aws:forecast:region:accountNumber:predictor/predictorName'
)

monitor_arn = create_monitor_response['MonitorArn']
```

```
describe_monitor_response = forecast.describe_monitor(
    MonitorArn = monitor_arn
)
print("Monitor status: " + describe_monitor_response['Status'])
```

Viewing Monitoring Results

After you generate a forecast and then import more data, you can view the results of predictor monitoring. You can see a visualization of results with the Forecast console or you can programmatically retrieve results with the ListMonitorEvaluations operation.

The Forecast console displays graphs of results for each predictor metric. Graphs include how each metric has changed over the lifetime of your predictor and predictor events, such as retraining.

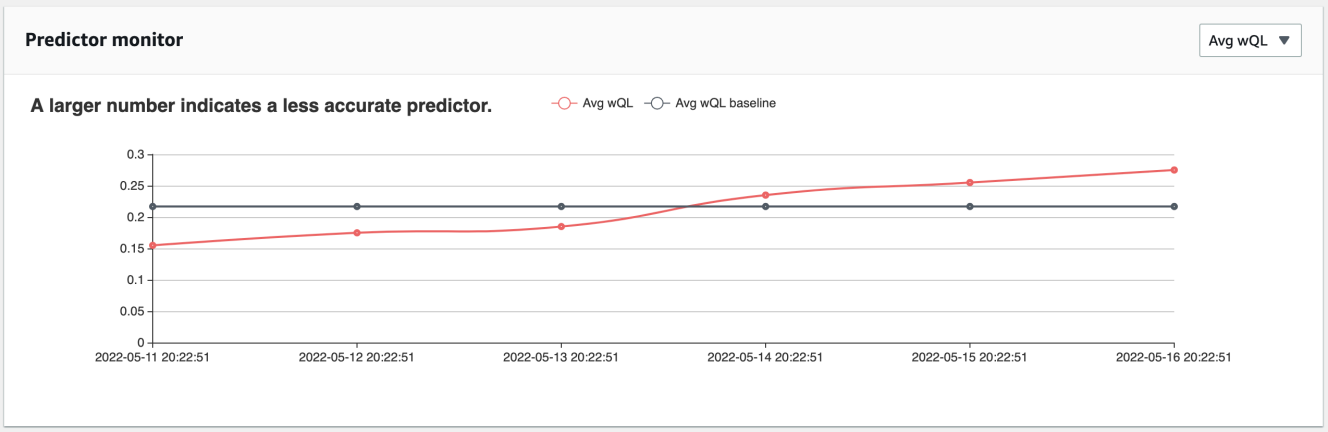
The ListMonitorEvaluations operation returns metric results and predictor events for different windows of time.

Console

To view predictor monitoring results

1. Sign in to the AWS Management Console and open the Amazon Forecast console at https://console.aws.amazon.com/forecast/.
2. From **Dataset groups**, choose your dataset group.
3. In the navigation pane, choose **Predictors**.
4. Choose the predictor and choose the **Monitoring** tab.
   - The **Monitoring results** section shows how different accuracy metrics have changed over time. Use the dropdown list to change which metric the graph tracks.
   - The **Monitoring history** section lists the details for the different events tracked in the results.

   The following is an example of a graph of how the `Avg wQL` score for a predictor has changed over time. In this graph, notice that the `Avg wQL` value is increasing over time. This increase indicates that the predictor accuracy is decreasing. Use this information to determine whether you need to revalidate the model and take action.



SDK for Python (Boto3)

To get monitoring results with the SDK for Python (Boto3), use the `list_monitor_evaluations` method. Provide the Amazon Resource Name (ARN) of the monitor, and optionally specify the maximum number of results to retrieve with the `MaxResults` parameter. Optionally specify a `Filter` to filter results. You can filter evaluations by a `EvaluationState` of either SUCCESS or `FAILURE`. The following code gets at maximum 20 successful monitoring evaluations.

```
import boto3

forecast = boto3.client('forecast')

monitor_results = forecast.list_monitor_evaluations(
    MonitorArn = 'monitor_arn',
    MaxResults = 20,
    Filters = [
      {
         "Condition": "IS",
         "Key": "EvaluationState",
         "Value": "SUCCESS"
      }
   ]
)
print(monitor_results)
```

The following is an example JSON response.

```
{
  "NextToken": "string",
  "PredictorMonitorEvaluations": [
    {
      "MonitorArn": "MonitorARN",
      "ResourceArn": "PredictorARN",
      "EvaluationTime": "2020-01-02T00:00:00Z",
      "EvaluationState": "SUCCESS",
      "WindowStartDatetime": "2019-01-01T00:00:00Z",
      "WindowEndDatetime": "2019-01-03T00:00:00Z",
      "PredictorEvent": {
        "Detail": "Retrain",
        "Datetime": "2020-01-01T00:00:00Z"
      },
      "MonitorDataSource": {
        "DatasetImportJobArn": "arn:aws:forecast:region:accountNumber:dataset-
import-job/*",
        "ForecastArn": "arn:aws:forecast:region:accountNumber:forecast/*",
        "PredictorArn": "arn:aws:forecast:region:accountNumber:predictor/*",

      },
      "MetricResults": [
        {
          "MetricName": "AverageWeightedQuantileLoss",
          "MetricValue": 0.17009070456599376
        },
        {
          "MetricName": "MAPE",
          "MetricValue": 0.250711322309796
        },
        {
          "MetricName": "MASE",
          "MetricValue": 1.6275608734888485
        },
        {
          "MetricName": "RMSE",
          "MetricValue": 3100.7125081405547
        },
        {
          "MetricName": "WAPE",
          "MetricValue": 0.17101159704738722}
      ]
```

```
    }
  ]
}
```

Restrictions and Best Practices

Consider the following restrictions and best practices when working with predictor monitoring.

- **Predictor monitoring is only available for auto predictors** – You cannot enable monitoring for legacy predictors that were created with AutoML or through manual selection. See Upgrading to AutoPredictor.
- **Predictor monitoring is unique per auto predictor** – You can only create one monitor per auto predictor.
- **Predictor monitoring requires new data and generating forecasts** – As you import new data that is used to generate new forecasts, predictor monitoring results become available. If you are not importing new data or the newly imported data does not cover a full forecast horizon, you will not see monitoring results.
- **Predictor monitoring requires new forecasts** – You must continuously generate new forecasts to generate monitoring results. If you are not generating new forecasts, you will not see monitoring results.
- **Amazon Forecast will store data from each of your forecasts for predictor performance analysis** – Forecast stores these data even if you delete forecasts. To delete these data, delete the associated monitor.
- The StopResource operation will stop all current evaluations and all future evaluations.
- The avgWQL metric is available only when you generate forecasts for quantiles other than the mean.
- In-progress monitor evaluations are not shown in the ListMonitorEvaluations operation.

Amazon Forecast Algorithms

An Amazon Forecast predictor uses an algorithm to train a model with your time series datasets. The trained model is then used to generate metrics and predictions.

If you are unsure of which algorithm to use to train your model, choose AutoML when creating a predictor and let Forecast train the optimal model for your datasets. Otherwise, you can manually select one of the Amazon Forecast algorithms.

Python notebooks

For a step-by-step guide on using AutoML, see Getting Started with AutoML.

Built-in Forecast Algorithms

Amazon Forecast provides six built-in algorithms for you to choose from. These range from commonly used statistical algorithms like Autoregressive Integrated Moving Average (ARIMA), to complex neural network algorithms like CNN-QR and DeepAR+.

CNN-QR

`arn:aws:forecast:::algorithm/CNN-QR`

Amazon Forecast CNN-QR, Convolutional Neural Network - Quantile Regression, is a proprietary machine learning algorithm for forecasting time series using causal convolutional neural networks (CNNs). CNN-QR works best with large datasets containing hundreds of time series. It accepts item metadata, and is the only Forecast algorithm that accepts related time series data without future values.

DeepAR+

`arn:aws:forecast:::algorithm/Deep_AR_Plus`

Amazon Forecast DeepAR+ is a proprietary machine learning algorithm for forecasting time series using recurrent neural networks (RNNs). DeepAR+ works best with large datasets containing hundreds of feature time series. The algorithm accepts forward-looking related time series and item metadata.

Prophet

`arn:aws:forecast:::algorithm/Prophet`

Prophet is a time series forecasting algorithm based on an additive model where non-linear trends are fit with yearly, weekly, and daily seasonality. It works best with time series with strong seasonal effects and several seasons of historical data.

NPTS

`arn:aws:forecast:::algorithm/NPTS`

The Amazon Forecast Non-Parametric Time Series (NPTS) proprietary algorithm is a scalable, probabilistic baseline forecaster. NPTS is especially useful when working with sparse or intermittent time series. Forecast provides four algorithm variants: Standard NPTS, Seasonal NPTS, Climatological Forecaster, and Seasonal Climatological Forecaster.

ARIMA

`arn:aws:forecast:::algorithm/ARIMA`

Autoregressive Integrated Moving Average (ARIMA) is a commonly used statistical algorithm for time-series forecasting. The algorithm is especially useful for simple datasets with under 100 time series.

ETS

`arn:aws:forecast:::algorithm/ETS`

Exponential Smoothing (ETS) is a commonly used statistical algorithm for time-series forecasting. The algorithm is especially useful for simple datasets with under 100 time series, and datasets with seasonality patterns. ETS computes a weighted average over all observations in the time series dataset as its prediction, with exponentially decreasing weights over time.

Comparing Forecast Algorithms

Use the following table to find the best option for your time series datasets.

	Neural Networks		Flexible Local Algorithms	Baseline Algorithms		
	CNN-QR	**DeepAR+**	**Prophet**	**NPTS**	**ARIMA**	**ETS**
Computationally intensive training process	High	High	Medium	Low	Low	Low

	Neural Networks		Flexible Local Algorithms	Baseline Algorithms		
	CNN-QR	**DeepAR+**	**Prophet**	**NPTS**	**ARIMA**	**ETS**
Accepts historical related time series*						
Accepts forward-looking related time series*						
Accepts item metadata (product color, brand, etc)						
Accepts the Weather Index built-in featurization						
Suitable for sparse datasets						
Performs Hyperparameter Optimization (HPO)						
Allows overriding default hyperparameter values						

*For more information on related time series, see Related Time Series.

Autoregressive Integrated Moving Average (ARIMA) Algorithm

Autoregressive Integrated Moving Average (ARIMA) is a commonly-used local statistical algorithm for time-series forecasting. ARIMA captures standard temporal structures (patterned organizations of time) in the input dataset. The Amazon Forecast ARIMA algorithm calls the Arima function in the `Package 'forecast'` of the Comprehensive R Archive Network (CRAN).

How ARIMA Works

The ARIMA algorithm is especially useful for datasets that can be mapped to stationary time series. The statistical properties of stationary time series, such as autocorrelations, are independent of time. Datasets with stationary time series usually contain a combination of signal and noise. The signal may exhibit a pattern of sinusoidal oscillation or have a seasonal component. ARIMA acts like a filter to separate the signal from the noise, and then extrapolates the signal in the future to make predictions.

ARIMA Hyperparameters and Tuning

For information about ARIMA hyperparameters and tuning, see the `Arima` function documentation in the Package 'forecast' of CRAN.

Amazon Forecast converts the `DataFrequency` parameter specified in the CreateDataset operation to the `frequency` parameter of the R ts function using the following table:

DataFrequency (string)	R ts frequency (integer)
Y	1
M	12
W	52
D	7
H	24
30min	2
15min	4
10min	6

DataFrequency (string)	R ts frequency (integer)
5min	12
1min	60

For frequencies less than 24 or short time series, the hyperparameters are set using the `auto.arima` function of the `Package 'forecast'` of CRAN. For frequencies greater than or equal to 24 and long time series, we use a Fourier series with K = 4, as described here, Forecasting with long seasonal periods.

Supported data frequencies that aren't in the table default to a `ts` frequency of 1.

CNN-QR Algorithm

Amazon Forecast CNN-QR, Convolutional Neural Network - Quantile Regression, is a proprietary machine learning algorithm for forecasting scalar (one-dimensional) time series using causal convolutional neural networks (CNNs). This supervised learning algorithm trains one global model from a large collection of time series and uses a quantile decoder to make probabilistic predictions.

Topics

- Getting Started with CNN-QR
- How CNN-QR Works
- Using Related Data with CNN-QR
- CNN-QR Hyperparameters
- Tips and Best Practices

Getting Started with CNN-QR

You can train a predictor with CNN-QR in two ways:

1. Manually selecting the CNN-QR algorithm.
2. Choosing AutoML (CNN-QR is part of AutoML).

If you are unsure of which algorithm to use, we recommend selecting AutoML, and Forecast will select CNN-QR if it is the most accurate algorithm for your data. To see if CNN-QR was selected as

the most accurate model, either use the DescribePredictor API or choose the predictor name in the console.

Here are some key use cases for CNN-QR:

- **Forecast with large and complex datasets** - CNN-QR works best when trained with large and complex datasets. The neural network can learn across many datasets, which is useful when you have related time series and item metadata.
- **Forecast with historical related time series** - CNN-QR does not require related time series to contain data points within the forecast horizon. This added flexibility allows you to include a broader range of related time series and item meta data, such as item price, events, web metrics, and product categories.

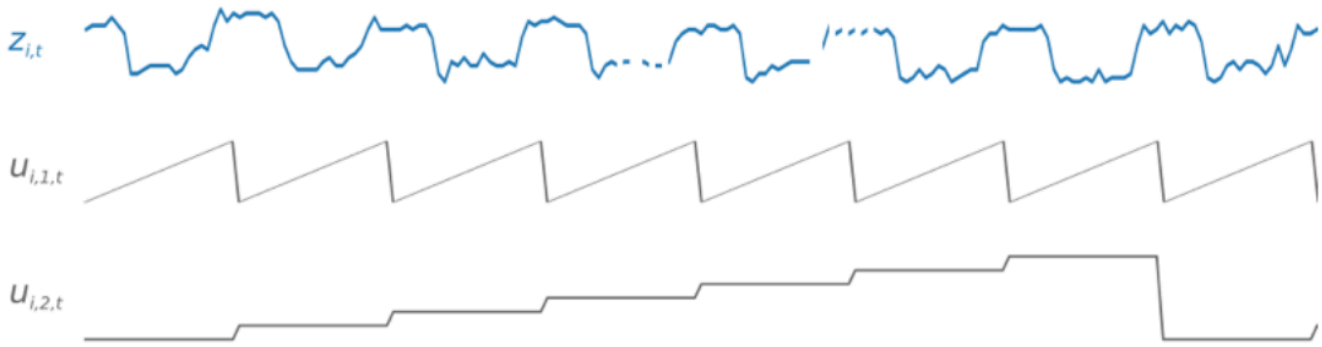
How CNN-QR Works

CNN-QR is a sequence-to-sequence (Seq2Seq) model for probabilistic forecasting that tests how well a prediction reconstructs the decoding sequence, conditioned on the encoding sequence.

The algorithm allows for different features in the encoding and the decoding sequences, so you can use a related time series in the encoder, and omit it from the decoder (and vice versa). By default, related time series with data points in the forecast horizon will be included in both the encoder and decoder. Related time series without data points in the forecast horizon will only be included in the encoder.

CNN-QR performs quantile regression with a hierarchical causal CNN serving as a learnable feature extractor.

To facilitate learning time-dependent patterns, such as spikes during weekends, CNN-QR automatically creates feature time series based on time-series granularity. For example, CNN-QR creates two feature time series (day-of-month and day-of-year) at a weekly time-series frequency. The algorithm uses these derived feature time series along with the custom feature time series provided during training and inference. The following example shows a target time series, $z_{i,t}$, and two derived time-series features: $u_{i,1,t}$ represents the hour of the day, and $u_{i,2,t}$ represents the day of the week.



CNN-QR automatically includes these feature time series based on the data frequency and the size of training data. The following table lists the features that can be derived for each supported basic time frequency.

Frequency of the Time Series	Derived Features
Minute	minute-of-hour, hour-of-day, day-of-week, day-of-month, day-of-year
Hour	hour-of-day, day-of-week, day-of-month, day-of-year
Day	day-of-week, day-of-month, day-of-year
Week	week-of-month, week-of-year
Month	month-of-year

During training, each time series in the training dataset consists of a pair of adjacent context and forecast windows with fixed predefined lengths. This is shown in the figure below, where the context window is represented in green, and the forecast window is represented in blue.

You can use a model trained on a given training set to generate predictions for time series in the training set, and for other time series. The training dataset consists of a target time series, which may be associated with a list of related time series and item metadata.

The figure below shows how this works for an element of a training dataset indexed by `i`. The training dataset consists of a target time series, $z_{i,t}$, and two associated related time series, $x_{i,1,t}$

and $x_{i,2,t}$. The first related time series, $x_{i,1,t}$, is a forward-looking time series, and the second, $x_{i,2,t}$, is a historical time series.

CNN-QR learns across the target time series, $z_{i,t}$, and the related time series, $x_{i,1,t}$ and $x_{i,2,t}$, to generate predictions in the forecast window, represented by the orange line.

Using Related Data with CNN-QR

CNNQR supports both historical and forward looking related time series datasets. If you provide a forward looking related time series dataset, any missing value will be filled using the future filling method. For more information on historical and forward-looking related time series, see Using Related Time Series Datasets.

You can also use item metadata datasets with CNN-QR. These are datasets with static information on the items in your target time series. Item metadata is especially useful for coldstart forecasting scenarios where there is little to no historical data. For more information on item metadata, see Item Metadata.

CNN-QR Hyperparameters

Amazon Forecast optimizes CNN-QR models on selected hyperparameters. When manually selecting CNN-QR, you have the option to pass in training parameters for these hyperparameters. The following table lists the tunable hyperparameters of the CNN-QR algorithm.

Parameter Name	Values	Description
`context_length`	Valid values Positive Integers	The number of time points that the model reads before making predictions. Typically

Parameter Name	Values	Description
	Valid range 10 to 500 Typical values 2 * ForecastHorizon to 12 * ForecastHorizon HPO tunable Yes	, CNN-QR has larger values for context_length than DeepAR+ because CNN-QR does not use lags to look at further historical data. If the value for context_length is outside of a predefined range, CNN-QR will automatically set the default context_length to an appropriate value.
use_related_data	Valid values ALL NONE HISTORICAL FORWARD_LOOKING Default value ALL HPO tunable Yes	Determines which kinds of related time series data to include in the model. Choose one of four options: • ALL: Include all provided related time series. • NONE: Exclude all provided related time series. • HISTORICAL : Include only related time series that *do not* extend into the forecast horizon. • FORWARD_LOOKING : Include only related time series that *do* extend into the forecast horizon. HISTORICAL includes all historical related time series, and FORWARD_LOOKING includes all forward-looking related time series. You cannot choose a subset of HISTORICAL or FORWARD_LOOKING related time series.

Parameter Name	Values	Description
use_item_metadata	Valid values ALL NONE Default value ALL HPO tunable Yes	Determines whether the model includes item metadata. Choose one of two options: • ALL: Include all provided item metadata. • NONE: Exlude all provided item metadata. use_item_metadata includes either all provided item metadata or none. You cannot choose a subset of item metadata.
epochs	Valid values Positive Integers Typical values 10 to 1000 Default value 100 HPO tunable No	The maximum number of complete passes through the training data. Smaller datasets require more epochs. For large values of ForecastHorizon and context_length , consider decreasing epochs to improve the training time.

Hyperparameter Optimization (HPO)

Hyperparameter optimization (HPO) is the task of selecting the optimal hyperparameter values for a specific learning objective. With Forecast, you can automate this process in two ways:

1. Choosing AutoML, and HPO will automatically run for CNN-QR.
2. Manually selecting CNN-QR and setting PerformHPO = TRUE.

Additional related time series and item metadata does not always improve the accuracy of your CNN-QR model. When you run AutoML or enable HPO, CNN-QR tests the accuracy of your model

with and without the provided related time series and item metadata, and selects the model with the highest accuracy.

Amazon Forecast automatically optimizes the following three hyperparameters during HPO and provides you with the final trained values:

- **context_length** - determines how far into the past the network can see. The HPO process automatically sets a value for `context_length` that maximizes model accuracy, while taking training time into account.
- **use_related_data** - determines which forms of related time series data to include in your model. The HPO process automatically checks whether your related time series data improves the model, and selects the optimal setting.
- **use_item_metadata** - determines whether to include item metadata in your model. The HPO process automatically checks whether your item metadata improves the model, and chooses the optimal setting.

> **Note**
>
> If `use_related_data` is set to `NONE` or `HISTORICAL` when the `Holiday` supplementary feature is selected, this means that including holiday data does not improve model accuracy.

You can set the HPO configuration for the `context_length` hyperparameter if you set `PerformHPO = TRUE` during manual selection. However, you cannot alter any aspect of the HPO configuration if you choose AutoML. For more information on HPO configuration, refer to the IntergerParameterRange API.

Tips and Best Practices

Avoid large values for ForecastHorizon - Using values over 100 for the `ForecastHorizon` will increase training time and can reduce model accuracy. If you want to forecast further into the future, consider aggregating to a higher frequency. For example, use `5min` instead of `1min`.

CNNs allow for a higher context length - With CNN-QR, you can set the `context_length` slightly higher than that for DeepAR+, as CNNs are generally more efficient than RNNs.

Feature engineering of related data - Experiment with different combinations of related time series and item metadata when training your model, and assess whether the additional information improves accuracy. Different combinations and transformations of related time series and item metadata will deliver different results.

CNN-QR does not forecast at the mean quantile – When you set `ForecastTypes` to `mean` with the CreateForecast API, forecasts will instead be generated at the median quantile (`0.5` or `P50`).

DeepAR+ Algorithm

Amazon Forecast DeepAR+ is a supervised learning algorithm for forecasting scalar (one-dimensional) time series using recurrent neural networks (RNNs). Classical forecasting methods, such as autoregressive integrated moving average (ARIMA) or exponential smoothing (ETS), fit a single model to each individual time series, and then use that model to extrapolate the time series into the future. In many applications, however, you have many similar time series across a set of cross-sectional units. These time-series groupings demand different products, server loads, and requests for web pages. In this case, it can be beneficial to train a single model jointly over all of the time series. DeepAR+ takes this approach. When your dataset contains hundreds of feature time series, the DeepAR+ algorithm outperforms the standard ARIMA and ETS methods. You can also use the trained model for generating forecasts for new time series that are similar to the ones it has been trained on.

> **Python notebooks**
>
> For a step-by-step guide on using the DeepAR+ algorithm, see Getting Started with DeepAR+.

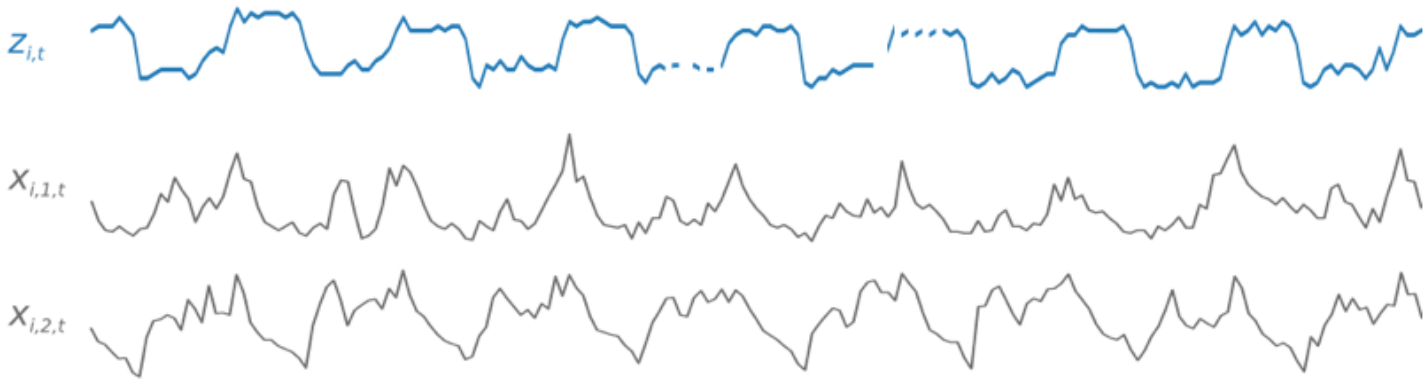
Topics

- How DeepAR+ Works
- DeepAR+ Hyperparameters
- Tune DeepAR+ Models

How DeepAR+ Works

During training, DeepAR+ uses a training dataset and an optional testing dataset. It uses the testing dataset to evaluate the trained model. In general, the training and testing datasets don't

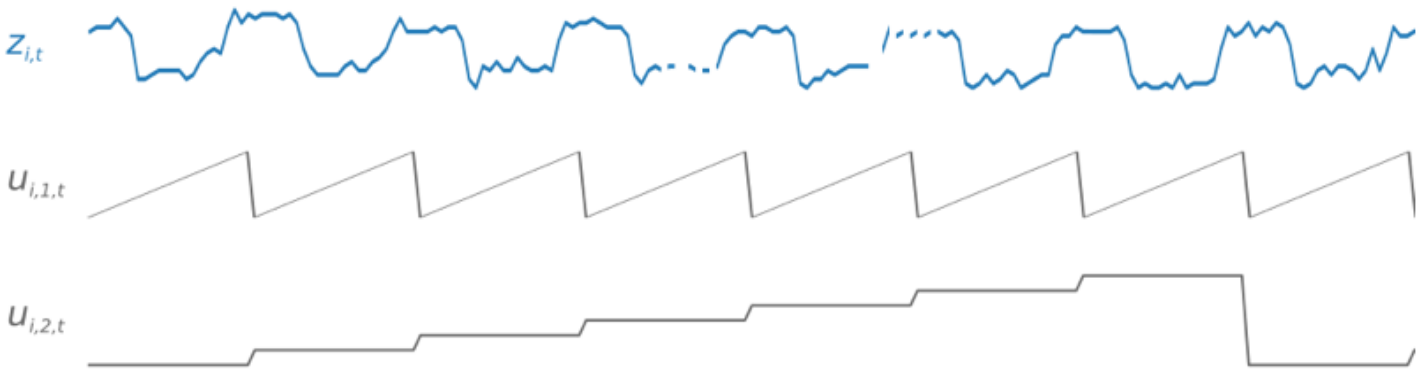
have to contain the same set of time series. You can use a model trained on a given training set to generate forecasts for the future of the time series in the training set, and for other time series. Both the training and the testing datasets consist of (preferably more than one) target time series. Optionally, they can be associated with a vector of feature time series and a vector of categorical features (for details, see DeepAR Input/Output Interface in the *SageMaker Developer Guide*). The following example shows how this works for an element of a training dataset indexed by `i`. The training dataset consists of a target time series, $z_{i,t}$, and two associated feature time series, $x_{i,1,t}$ and $x_{i,2,t}$.



The target time series might contain missing values (denoted in the graphs by breaks in the time series). DeepAR+ supports only feature time series that are known in the future. This allows you to run counterfactual "what-if" scenarios. For example, "What happens if I change the price of a product in some way?"

Each target time series can also be associated with a number of categorical features. You can use these to encode that a time series belongs to certain groupings. Using categorical features allows the model to learn typical behavior for those groupings, which can increase accuracy. A model implements this by learning an embedding vector for each group that captures the common properties of all time series in the group.

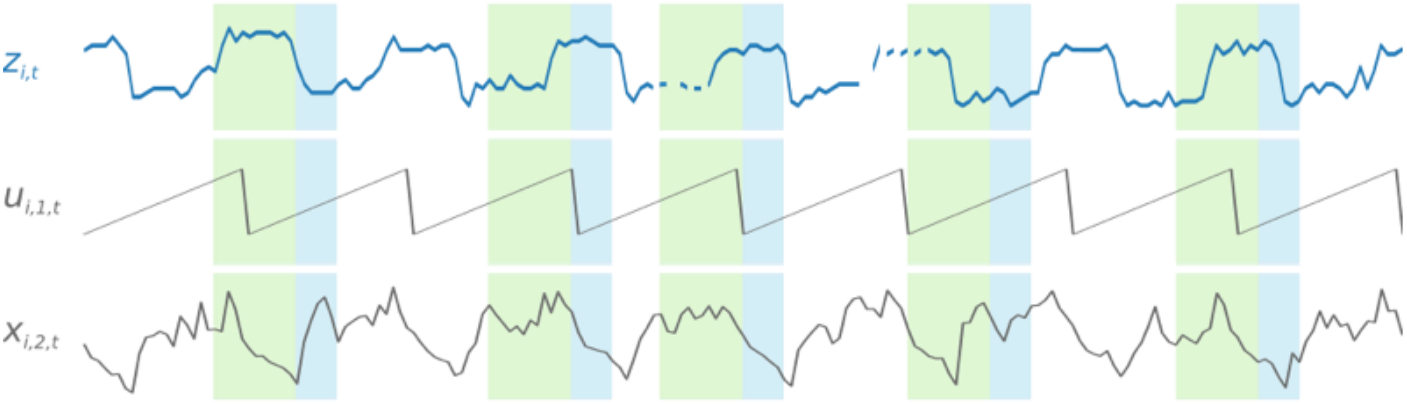
To facilitate learning time-dependent patterns, such as spikes during weekends, DeepAR+ automatically creates feature time series based on time-series granularity. For example, DeepAR + creates two feature time series (day of the month and day of the year) at a weekly time-series frequency. It uses these derived feature time series along with the custom feature time series that you provide during training and inference. The following example shows two derived time-series features: $u_{i,1,t}$ represents the hour of the day, and $u_{i,2,t}$ the day of the week.



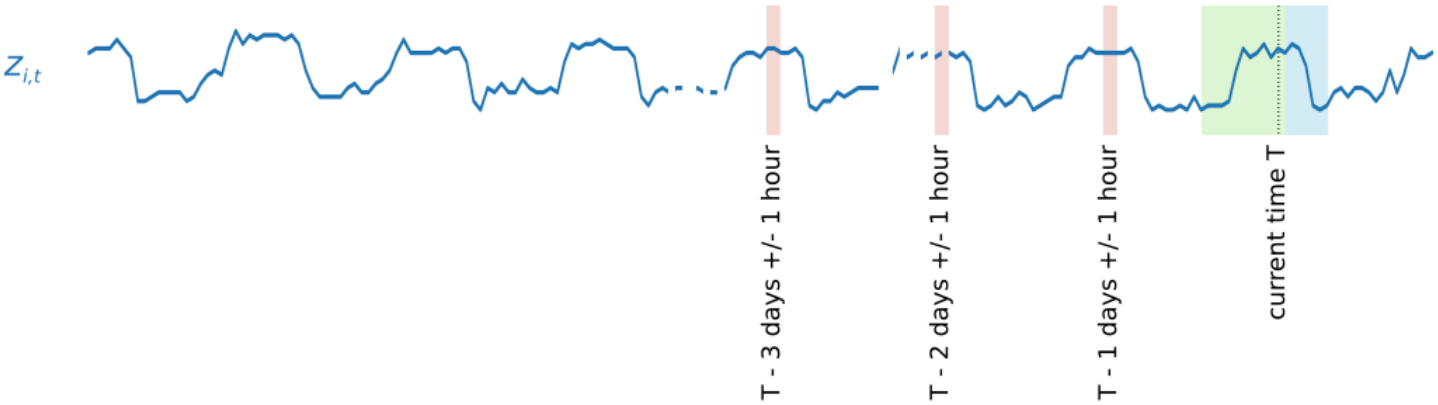
DeepAR+ automatically includes these feature time series based on the data frequency and the size of training data. The following table lists the features that can be derived for each supported basic time frequency.

Frequency of the Time Series	Derived Features
Minute	minute-of-hour, hour-of-day, day-of-week, day-of-month, day-of-year
Hour	hour-of-day, day-of-week, day-of-month, day-of-year
Day	day-of-week, day-of-month, day-of-year
Week	week-of-month, week-of-year
Month	month-of-year

A DeepAR+ model is trained by randomly sampling several training examples from each of the time series in the training dataset. Each training example consists of a pair of adjacent context and prediction windows with fixed predefined lengths. The `context_length` hyperparameter controls how far in the past the network can see, and the `ForecastHorizon` parameter controls how far in the future predictions can be made. During training, Amazon Forecast ignores elements in the training dataset with time series shorter than the specified prediction length. The following example shows five samples, with a context length (highlighted in green) of 12 hours and a prediction length (highlighted in blue) of 6 hours, drawn from element `i`. For the sake of brevity, we've excluded the feature time series $x_{i,1,t}$ and $u_{i,2,t}$.



To capture seasonality patterns, DeepAR+ also automatically feeds lagged (past period) values from the target time series. In our example with samples taken at an hourly frequency, for each time index `t = T`, the model exposes the $z_{i,t}$ values which occurred approximately one, two, and three days in the past (highlighted in pink).



For inference, the trained model takes as input the target time series, which might or might not have been used during training, and forecasts a probability distribution for the next `ForecastHorizon` values. Because DeepAR+ is trained on the entire dataset, the forecast takes into account learned patterns from similar time series.

For information on the mathematics behind DeepAR+, see DeepAR: Probabilistic Forecasting with Autoregressive Recurrent Networks on the Cornell University Library website.

DeepAR+ Hyperparameters

The following table lists the hyperparameters that you can use in the DeepAR+ algorithm. Parameters in bold participate in hyperparameter optimization (HPO).

Parameter Name	Description
context_length	The number of time points that the model reads in before making the prediction. The value for this parameter should be about the same as the ForecastHorizon . The model also receives lagged inputs from the target, so context_length can be much smaller than typical seasonalities. For example, a daily time series can have yearly seasonality. The model automatically includes a lag of one year, so the context length can be shorter than a year. The lag values that the model picks depend on the frequency of the time series. For example, lag values for daily frequency are: previous week, 2 weeks, 3 weeks, 4 weeks, and year. Valid values Positive integers Typical values ceil(0.1 * ForecastHorizon) to min(200, 10 * ForecastHorizon) Default value 2 * ForecastHorizon
epochs	The maximum number of passes to go over the training data. The optimal value depends on your data size and learning rate. Smaller datasets and lower learning rates both require more epochs, to achieve good results. Valid values Positive integers Typical values 10 to 1000 Default value 500
learning_rate	The learning rate used in training.

Parameter Name	Description
	Valid values Positive floating-point numbers Typical values 0.0001 to 0.1 Default value 0.001
`learning_rate_decay`	The rate at which the learning rate decreases. At most, the learning rate is reduced `max_learning_rate_decays` times, then training stops. This parameter will be used only if `max_learning_rate_decays` is greater than 0. Valid values Positive floating-point numbers Typical values 0.5 to 0.8 (inclusive) Default value 0.5

Parameter Name	Description
likelihood	The model generates a probabilistic forecast, and can provide quantiles of the distribution and return samples. Depending on your data, choose an appropriate likelihood (noise model) that is used for uncertainty estimates. Valid values • beta: Use for real-valued targets between 0 and 1, inclusively. • deterministic-L1 : A loss function that does not estimate uncertainty and only learns a point forecast. • gaussian: Use for real-valued data. • negative-binomial : Use for count data (non-negative integers). • piecewise-linear : Use for flexible distributions. • student-T : Use this alternative for real-valued data for bursty data. Default value student-T
max_learning_rate_decays	The maximum number of learning rate reductions that should occur. Valid values Positive integers Typical values 0 to 10 Default value 0

Parameter Name	Description
num_averaged_models	In DeepAR+, a training trajectory can encounter multiple models. Each model might have different forecasting strengths and weaknesses. DeepAR+ can average the model behaviors to take advantage of the strengths of all models. Valid values Positive integers Typical values 1 to 5 (inclusive) Default value 1
num_cells	The number of cells to use in each hidden layer of the RNN. Valid values Positive integers Typical values 30 to 100 Default value 40

Parameter Name	Description
num_layers	The number of hidden layers in the RNN. Valid values Positive integers Typical values 1 to 4 Default value 2

Tune DeepAR+ Models

To tune Amazon Forecast DeepAR+ models, follow these recommendations for optimizing the training process and hardware configuration.

Best Practices for Process Optimization

To achieve the best results, follow these recommendations:

- Except when splitting the training and testing datasets, always provide entire time series for training and testing, and when calling the model for inference. Regardless of how you set `context_length`, don't divide the time series or provide only a part of it. The model will use data points further back than `context_length` for the lagged values feature.
- For model tuning, you can split the dataset into training and testing datasets. In a typical evaluation scenario, you should test the model on the same time series used in training, but on the future `ForecastHorizon` time points immediately after the last time point visible during training. To create training and testing datasets that satisfy these criteria, use the entire dataset (all of the time series) as a testing dataset and remove the last `ForecastHorizon` points from each time series for training. This way, during training, the model doesn't see the target values for time points on which it is evaluated during testing. In the test phase, the last `ForecastHorizon` points of each time series in the testing dataset are withheld and a prediction is generated. The forecast is then compared with the actual values for the last `ForecastHorizon` points. You can create more complex evaluations by repeating time series

multiple times in the testing dataset, but cutting them off at different end points. This produces accuracy metrics that are averaged over multiple forecasts from different time points.

- Avoid using very large values (> 400) for the `ForecastHorizon` because this slows down the model and makes it less accurate. If you want to forecast further into the future, consider aggregating to a higher frequency. For example, use `5min` instead of `1min`.
- Because of lags, the model can look further back than `context_length`. Therefore, you don't have to set this parameter to a large value. A good starting point for this parameter is the same value as the `ForecastHorizon`.
- Train DeepAR+ models with as many time series as are available. Although a DeepAR+ model trained on a single time series might already work well, standard forecasting methods such as ARIMA or ETS might be more accurate and are more tailored to this use case. DeepAR+ starts to outperform the standard methods when your dataset contains hundreds of feature time series. Currently, DeepAR+ requires that the total number of observations available, across all training time series, is at least 300.

Exponential Smoothing (ETS) Algorithm

Exponential Smoothing (ETS) is a commonly-used local statistical algorithm for time-series forecasting. The Amazon Forecast ETS algorithm calls the ets function in the `Package 'forecast'` of the Comprehensive R Archive Network (CRAN).

How ETS Works

The ETS algorithm is especially useful for datasets with seasonality and other prior assumptions about the data. ETS computes a weighted average over all observations in the input time series dataset as its prediction. The weights are exponentially decreasing over time, rather than the constant weights in simple moving average methods. The weights are dependent on a constant parameter, which is known as the smoothing parameter.

ETS Hyperparameters and Tuning

For information about ETS hyperparameters and tuning, see the `ets` function documentation in the Package 'forecast' of CRAN.

Amazon Forecast converts the `DataFrequency` parameter specified in the CreateDataset operation to the `frequency` parameter of the R ts function using the following table:

DataFrequency (string)	R ts frequency (integer)
Y	1
M	12
W	52
D	7
H	24
30min	2
15min	4
10min	6
5min	12
1min	60

Supported data frequencies that aren't in the table default to a `ts` frequency of 1.

Non-Parametric Time Series (NPTS) Algorithm

The Amazon Forecast Non-Parametric Time Series (NPTS) algorithm is a scalable, probabilistic baseline forecaster. It predicts the future value distribution of a given time series by sampling from past observations. The predictions are bounded by the observed values. NPTS is especially useful when the time series is intermittent (or sparse, containing many 0s) and bursty. For example, forecasting demand for individual items where the time series has many low counts. Amazon Forecast provides variants of NPTS that differ in which of the past observations are sampled and how they are sampled. To use an NPTS variant, you choose a hyperparameter setting.

How NPTS Works

Similar to classical forecasting methods, such as exponential smoothing (ETS) and autoregressive integrated moving average (ARIMA), NPTS generates predictions for each time series individually. The time series in the dataset can have different lengths. The time points where the observations

are available are called the training range and the time points where the prediction is desired are called the prediction range.

Amazon Forecast NPTS forecasters have the following variants: NPTS, seasonal NPTS, climatological forecaster, and seasonal climatological forecaster.

Topics

NPTS

In this variant, predictions are generated by sampling from all observations in the training range of the time series. However, instead of uniformly sampling from all of the observations, this variant assigns weight to each of the past observations according to how far it is from the current time step where the prediction is needed. In particular, it uses weights that decay exponentially according to the distance of the past observations. In this way, the observations from the recent past are sampled with much higher probability than the observations from the distant past. This assumes that the near past is more indicative for the future than the distant past. You can control the amount of decay in the weights with the `exp_kernel_weights` hyperparameter.

To use this NPTS variant in Amazon Forecast, set the `use_seasonal_model` hyperparameter to `False` and accept all other default settings.

Seasonal NPTS

The seasonal NPTS variant is similar to NPTS except that instead of sampling from all of the observations, it uses only the observations from the past *seasons* . By default, the season is determined by the granularity of the time series. For example, for an hourly time series, to predict for hour *t*, this variant samples from the observations corresponding to the hour *t* on the previous days. Similar to NPTS, observation at hour *t* on the previous day is given more weight than the observations at hour *t* on earlier days. For more information about how to determine seasonality based on the granularity of the time series, see the section called “Seasonal Features”.

Climatological Forecaster

The climatological forecaster variant samples all of the past observations with uniform probability.

To use the climatological forecaster, set the `kernel_type` hyperparameter to `uniform` and the `use_seasonal_model` hyperparameter to `False`. Accept the default settings for all other hyperparameters.

Seasonal Climatological Forecaster

Similar to seasonal NPTS, the seasonal climatological forecaster samples the observations from past seasons, but samples them with uniform probability.

To use the seasonal climatological forecaster, set the `kernel_type` hyperparameter to `uniform`. Accept all other default settings for all of the other hyperparameters.

Seasonal Features

To determine what corresponds to a season for the seasonal NPTS and seasonal climatological forecaster, use the features listed in the following table. The table lists the derived features for the supported basic time frequencies, based on granularity. Amazon Forecast includes these feature time series, so you don't have to provide them.

Frequency of the Time Series	Feature to Determine Seasonality
Minute	minute-of-hour
Hour	hour-of-day
Day	day-of-week
Week	day-of-month
Month	month-of-year

Best Practices

When using the Amazon Forecast NPTS algorithms, consider the following best practices for preparing the data and achieving optimal results:

- Because NPTS generates predictions for each time series individually, provide the entire time series when calling the model for prediction. Also, accept the default value of the `context_length` hyperparameter. This causes the algorithm to use the entire time series.
- If you change the `context_length` (because the training data is too long), make sure it is large enough and covers multiple past seasons. For example, for a daily time series, this value must be at least 365 days (provided that you have that amount of data).

NPTS Hyperparameters

The following table lists the hyperparameters that you can use in the NPTS algorithm.

Parameter Name	Description
context_length	The number of time-points in the past that the model uses for making the prediction. By default, it uses all of the time points in the training range. Typically, the value for this hyperparameter should be large and should cover multiple past seasons. For example, for the daily time series this value must be at least 365 days. Valid values Positive integers Default value The length of the training time series

kernel_type	The kernel to use to define the weights used for sampling past observations.
	Valid values
	`exponential` or `uniform`
	Default values
	`exponential`

exp_kernel_weights	Valid only when `kernel_type` is `exponential` .

Parameter Name	Description
	The scaling parameter of the kernel. For faster (exponential) decay in the weights given to the observations in the distant past, use a large value. Valid values Positive floating-point numbers Default value `0.01`

use_seasonal_model	Whether to use a seasonal variant.
	Valid values
	`True` or `False`
	Default value
	`True`

use_default_time_features	Valid only for the *seasonal NPTS* and *seasonal climatological forecaster* variants.
	Whether to use seasonal features based on the granularity of the time series to determine seasonality.
	Valid values
	`True` or `False`
	Default value
	`True`

Prophet Algorithm

Prophet is a popular local Bayesian structural time series model. The Amazon Forecast Prophet algorithm uses the Prophet class of the Python implementation of Prophet.

How Prophet Works

Prophet is especially useful for datasets that:

- Contain an extended time period (months or years) of detailed historical observations (hourly, daily, or weekly)
- Have multiple strong seasonalities
- Include previously known important, but irregular, events
- Have missing data points or large outliers
- Have non-linear growth trends that are approaching a limit

Prophet is an additive regression model with a piecewise linear or logistic growth curve trend. It includes a yearly seasonal component modeled using Fourier series and a weekly seasonal component modeled using dummy variables.

For more information, see Prophet: forecasting at scale.

Prophet Hyperparameters and Related Time Series

Amazon Forecast uses the default Prophet hyperparameters. Prophet also supports related time-series as features, provided to Amazon Forecast in the related time-series CSV file.

Generating Forecasts

After you create an Amazon Forecast predictor, you are ready to create a forecast. By default, a forecast includes predictions for every item (`item_id`) in the dataset group that was used to train the predictor. However, you can specify a subset of items that are used to generate a forecast.

After you create a forecast, you can export it to your Amazon Simple Storage Service (Amazon S3) bucket.

Topics

Creating a forecast

You can create a forecast with the Forecast console, AWS CLI, or AWS SDKs. The status of your predictor must be **Active** before you can generate a forecast.

Console

To create a forecast

1. Sign in to the AWS Management Console and open the Amazon Forecast console at https://console.aws.amazon.com/forecast/.
2. From **Dataset groups**, choose your dataset group.
3. On your dataset group's **Dashboard**, under **Generate forecasts**, choose **Create a forecast**. The **Create a forecast** page appears.
4. On the **Create a forecast** page, for **Forecast details**, provide a name for your forecast and choose the predictor you want to use to create forecasts.
5. For **Forecast quantiles**, optionally specify the quantiles at which probabilistic forecasts are generated. The default quantiles are the quantiles you specified during predictor creation.

6. Optionally, choose the radio button for **Selected Items** to specify a subset of time series that are used for forecast generation.
7. Optionally, add any tags for the forecast. For more information see Tagging Amazon Forecast Resources.
8. Choose **Start**. The **Forecasts** page appears.

   The **Status** column lists the status of your forecast. Wait for Amazon Forecast to finish creating the forecast. The process can take several minutes or longer. When your forecast has been created, the status transitions to **Active**.

   Now that your forecast has been created, you can export the forecast. See Exporting a forecast.

CLI

To create a forecast with the AWS CLI, use the `create-forecast` command. Provide a name for the forecast and the Amazon Resource Name (ARN) of your predictor. For `forecast-types`, optionally specify the quantiles at which probabilistic forecasts are generated. The default values are the quantiles you specified when you created the predictor. Optionally add any tags for the forecast. For more information see Tagging Amazon Forecast Resources.

For information on required and optional parameters see CreateForecast.

```
aws forecast create-forecast \
--forecast-name forecast_name \
--forecast-types 0.1 0.5 0.9 \
--predictor-arn arn:aws:forecast:region:account_number:predictor/predictorName \
--tags Key=key1,Value=value1 Key=key2,Value=value2
```

Python

To create a forecast with the SDK for Python (Boto3), use the `create_forecast` method. Provide a name for the forecast and the Amazon Resource Name (ARN) of your predictor. For `ForecastTypes`, optionally specify the quantiles at which probabilistic forecasts are generated. The default values are the quantiles you specified when you created the predictor. Optionally add any tags for the forecast. For more information see Tagging Amazon Forecast Resources.

For information on required and optional parameters see CreateForecast.

```
import boto3

forecast = boto3.client('forecast')

create_forecast_response = forecast.create_forecast(
   ForecastName = "Forecast_Name",
   ForecastTypes = ["0.1", "0.5", "0.9"],        # optional, the default types/
quantiles are what you specified for the predictor
   PredictorArn = "arn:aws:forecast:region:accountNumber:predictor/predictorName",
   Tags = [
      {
         "Key": "key1",
         "Value": "value1"
      },
      {
         "Key": "key2",
         "Value": "value2"
      }
   ]
)
forecast_arn = create_forecast_response['ForecastArn']
print(forecast_arn)
```

Specifying time series

> **Note**
>
> A time series is a combination of the item (item_id) and all dimensions in your datasets.

To specify a list of time series, upload a CSV file identifying the time series by their item_id and dimension values to an S3 bucket. You must also define the attributes and attribute types of the time series in a schema.

For example, a retailer may want to know how an advertising campaign impacts sales for a specific item (`item_id`) at a specific store location (`store_location`). In this use case, you would specify the time series that is the combination of item_id and store_location.

The following CSV file selects the following five time series:

1. Item_id: 001, store_location: Seattle
2. Item_id: 001, store_location: New York
3. Item_id: 002, store_location: Seattle
4. Item_id: 002, store_location: New York
5. Item_id: 003, store_location: Denver

```
001, Seattle
001, New York
002, Seattle
002, New York
003, Denver
```

The schema defines the first column as `item_id` and the second column as `store_location`.

Forecast creation is skipped for any time series that you specify that are not in the input dataset. The forecast export file will not contain these time series or their forecasted values.

Exporting a forecast

After you create a forecast, you can export it to an Amazon S3 bucket. Exporting a forecast copies the forecast to your Amazon S3 bucket as a CSV file (by default), and the exported data includes all attributes of any item metadata dataset in addition to item predictions. You can specify the Parquet file format when you export a forecast.

The granularity of the exported forecasts (such as hourly, daily, or weekly) is the forecast frequency that you specified when you created the predictor. You can optionally specify an AWS Key Management Service key to encrypt the data before it's written to the bucket.

> **Note**
>
> Export files can directly return information from the Dataset Import. This makes the files vulnerable to CSV injection if the imported data contains formulas or commands. For this reason, exported files can prompt security warnings. To avoid malicious activity, disable links and macros when reading exported files.

Console

To export a forecast

1. In the navigation pane, under your dataset group, choose **Forecasts**.
2. Choose the radio button for your forecast and choose **Create forecast export**. The **Create forecast export** page is displayed.
3. On the **Create forecast export** page, for **Export details**, provide the following information.

   - **Export name** – Enter a name for your forecast export job.
   - **Generated forecast** – From the drop-down menu, choose the forecast that you created in `Step 3: Create a Forecast`.
   - **IAM role** – Either keep the default **Enter a custom IAM role ARN** or choose **Create a new role** to have Amazon Forecast create the role for you.
   - **Custom IAM role ARN** – If you are entering a custom IAM role, enter the Amazon Resource Name (ARN) of the IAM role that you created in Create an IAM Role for Amazon Forecast (IAM Console).
   - **KMS key ARN** – If you use AWS Key Management Service for bucket encryption, provide the Amazon Resource Name (ARN) of the AWS KMS key.
   - **S3 forecast export location** – Use the following format to enter the location of your Amazon Simple Storage Service (Amazon S3) bucket or folder in the bucket:

     `s3://<name of your S3 bucket>/<folder path>/`
4. Choose **Create forecast export**. The **my_forecast** page is displayed.

   Wait for Amazon Forecast to finish exporting the forecast. The process can take several minutes or longer. When your forecast has been exported, the status transitions to **Active** and you can find the forecast files in your Amazon S3 bucket.

CLI

To export a forecast with the AWS CLI you use the `export-forecast-job` command. Give the forecast export job a name, specify the ARN of the forecast to export, and optionally add any tags. For the `destination`, specify the path to your output Amazon S3 bucket, the ARN of the IAM role that you created in Create an IAM Role for Amazon Forecast (IAM Console), and if you use a AWS KMS key for bucket encryption, the ARN for your key.

For more information about required and optional parameters, see CreateForecastExportJob operation.

```
forecast create-forecast-export-job \
--forecast-export-job-name exportJobName \
--forecast-arn arn:aws:forecast:region:acctNumber:forecast/forecastName \
--destination
 S3Config="{Path='s3://bucket/folderName',RoleArn='arn:aws:iam::acctNumber:role/
Role, KMSKeyArn='arn:aws:kms:region:accountNumber:key/keyID'}"
--tags Key=key1,Value=value1 Key=key2,Value=value2
```

Python

To export a forecast with the SDK for Python (Boto3) you use the `export_forecast_job` method. Give the forecast export job a name, specify the ARN of the forecast to export, and optionally add any tags. For the `Destination`, specify the path to your output Amazon S3 bucket, the ARN of the IAM role that you created in Create an IAM Role for Amazon Forecast (IAM Console), and if you use a AWS KMS key for bucket encryption, the ARN for your key.

For more information about required and optional parameters, see CreateForecastExportJob operation.

```
import boto3

forecast = boto3.client('forecast')

export_forecast_response = forecast.create_forecast_export_job(
   Destination = {
      "S3Config": {
         "Path": "s3://bucketName/folderName/",
         "RoleArn": "arn:aws:iam::accountNumber:role/roleName",
         "KMSKeyArn": "arn:aws:kms:region:accountNumber:key/keyID"
      }
   },
   ForecastArn = "arn:aws:forecast:region:accountNumber:forecast/forecastName",
   ForecastExportJobName = "export_job_name",
   Tags = [
      {
         "Key": "key1",
         "Value": "value1"
      },
      {
```

```
        "Key": "key2",
        "Value": "value2"
      }
   ]
)
forecast_export_job_arn = export_forecast_response["ForecastExportJobArn"]
print(forecast_export_job_arn)
```

Querying a forecast

You can query a forecast using the QueryForecast operation. By default, the complete range of the forecast is returned. You can request a specific date range within the complete forecast.

When you query a forecast you must specify filtering criteria. A filter is a key-value pair. The key is one of the schema attribute names (including forecast dimensions) from one of the datasets used to create the forecast. The *value* is a valid values for the specified key. You can specify multiple key-value pairs. The returned forecast will only contain items that satisfy all the criteria.

Coldstart Forecasts

A common challenge faced by customers in industries such as retail, manufacturing, or consumer packaged goods is to generate forecasts for items with no historical data. This scenario is known as coldstart forecasting and is typically encountered when businesses introduce new products to market, on-board brands or catalogues, or cross-sell products in new regions.

Amazon Forecast requires item metadata to perform coldstart forecasting. Leveraging item characteristics found in the item metadata, Forecast explicitly identifies items in the item metadata that are similar to the item with no historical data. Forecast uses the demand characteristics of the existing items to generate a coldstart forecast for the new item.

Amazon Forecast identifies coldstart items as those items that are included in the item metadata file but are not included in the target time series file. To correctly identify a coldstart item, ensure that the item ID of the coldstart item is entered as a row in the item metadata file and that it is not entered in the target time series file. For multiple coldstart items, enter each item ID as a separate row in the item metadata file. If the coldstart item does not have an item ID, you can use any alphanumeric combination less than 64 characters and not already used by another item in dataset.

Coldstart forecasting requires both an item metadata dataset and an AutoPredictor.

Forecast Explainability

Forecast Explainability helps you better understand how the attributes in your datasets impact forecasts for specific time series (item and dimension combinations) and time points. Forecast uses a metric called Impact scores to quantify the relative impact of each attribute and determine whether they increase or decrease forecast values.

For example, consider a forecasting scenario where the target is `sales` and there are two related attributes: `price` and `color`. Forecast may find that the item's color has a high impact on sales for certain items, but a negligible effect for other items. It may also find that a promotion in the summer has a high impact on sales, but a promotion in the winter has little effect.

To enable Forecast Explainability, your predictor must include at least one of the following: related time series, item metadata, or additional datasets like Holidays and the Weather Index. See Restrictions and best practices for more information.

To view aggregated Impact scores for all time series and time points in your datasets, use Predictor Explainability instead of Forecast Explainability. See Predictor Explainability.

Python notebooks

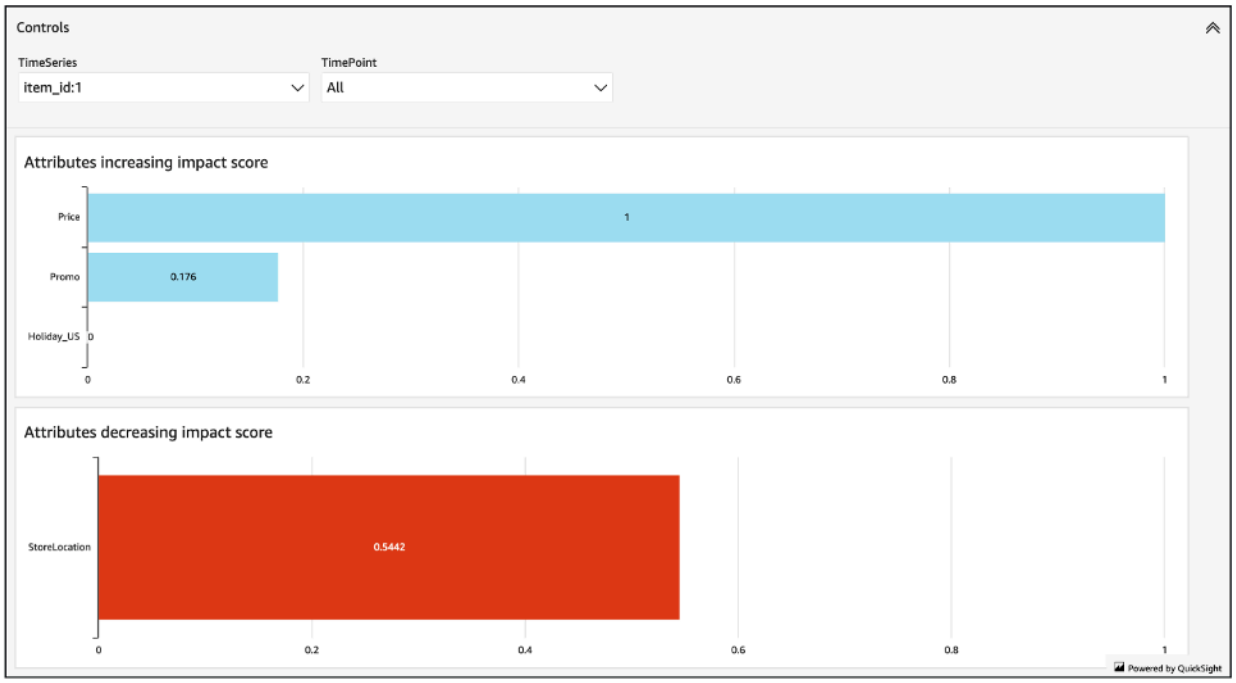
For a step-by-step guide on Forecast Explainability, see Item-Level Explainability.

Topics

Interpreting Impact Scores

Impact scores measure the relative impact attributes have on forecast values. For example, if the 'price' attribute has an impact score that is twice as large as the 'store location' attribute, you can conclude that the price of an item has twice the impact on forecast values than the store location.

Impact scores also provide information on whether attributes increase or decrease forecast values. In the console, this is denoted by the two graphs. Attributes with blue bars increase forecast values, while attributes with red bars decrease forecast values.



It is important to note that Impact scores measure the relative impact of attributes, not the absolute impact. Therefore, Impact scores cannot be used to determine whether particular attributes improve model accuracy. If an attribute has a low Impact score, that does not necessarily mean that it has a low impact on forecast values; it means that it has a lower impact on forecast values than other attributes used by the predictor.

It is possible for all or some impact scores to be zero. This can occur if the features have no impact on forecast values, the AutoPredictor used only a non-ML algorithm, or you did not provide related time series or item metadata.

For Forecast Explainability, Impact scores come in two forms: Normalized impact scores and Raw impact scores. Raw impact scores are based on Shapley values and are not scaled or bounded. Normalized impact scores scale the raw scores to a value between -1 and 1.

Raw impact scores are useful for combining and comparing scores across different Explainability resources. For example, if your predictor contains over 50 time series or over 500 time points, you can create multiple Forecast Explainability resources to cover a greater combined number of time series or time points, and directly compare raw impact scores for attributes. However, raw impact scores for Forecast Explainability resources from different forecasts are not directly comparable.

When viewing Impact scores in the console, you will only see Normalized impact scores. Exporting Explainability will provide you with both raw and normalized scores.

Creating Forecast Explainability

With Forecast Explainability, you can explore how attributes are impacting forecast values for specific time series at specific time points. After specifying time series and time points, Amazon Forecast calculates Impact scores for only those specific time series and time points.

You can enable Forecast Explainability for a predictor using the Software Development Kit (SDK) or the Amazon Forecast console. When using the SDK, use the CreateExplainability operation.

Topics

Specifying time series

> **Note**
>
> A time series is a combination of the item (item_id) and all dimensions in your datasets

When you specify time series (item and dimension combinations) for Forecast Explainability, Amazon Forecast calculates Impact scores for attributes for only those specific time series.

To specify a list of time series, upload a CSV file identifying the time series by their item_id and dimension values to an S3 bucket. You can specify up to 50 time series. You must also define the attributes and attribute types of the time series in a schema.

For example, a retailer may want to know how a promotion impacts sales for a specific item (`item_id`) at a specific store location (`store_location`). In this use case, you would specify the time series that is the combination of item_id and store_location.

The following CSV file selects the following five time series:

1. Item_id: 001, store_location: Seattle

2. Item_id: 001, store_location: New York
3. Item_id: 002, store_location: Seattle
4. Item_id: 002, store_location: New York
5. Item_id: 003, store_location: Denver

```
001, Seattle
001, New York
002, Seattle
002, New York
003, Denver
```

The schema defines the first column as `item_id` and the second column as `store_location`.

You can specify time series using the Forecast console or the Forecast Software Development Kit (SDK).

Console

To specify time series for Forecast Explainability

1. Sign in to the AWS Management Console and open the Amazon Forecast console at https://console.aws.amazon.com/forecast/.
2. From **Dataset groups**, choose your dataset group.
3. In the navigation pane, choose **Insights**.
4. Choose **Create Explainability**.
5. In the **Explainability name** field, provide a unique name for the Forecast Explainability.
6. In the **Select forecast** field, choose your forecast.
7. In the **S3 location** field, enter the location of the file with your time series.
8. In the **Data schema** field, set the **attribute name** and **attribute type** of the item ID and dimensions used in your time series.
9. Choose **Create Explainability.**

SDK

To specify time series for Forecast Explainability

Using the CreateExplainability operation, provide a unique name for ExplainabilityName and provide your forecast ARN for ResourceArn.

Configure the following datatypes:

- `ExplainabilityConfig` - set values for TimeSeriesGranularity to “SPECIFIC” and TimePointGranularity to “ALL”. (To specify time points, set TimePointGranularity to “SPECIFIC”. See Specifying time points)
- `S3Config` - set the values for “Path” to the S3 location of the time series file and “RoleArn” to a role with access to the S3 bucket.
- `Schema` - define the “AttributeName” and “AttributeType” for item_id and the dimensions in your time series.

The example below shows a schema for time series using a combination of “item_id” and the “store_location” dimension.

```
{
    "ExplainabilityName" : [unique_name],
    "ResourceArn" : [forecast_arn],
    "ExplainabilityConfig" {
        "TimeSeriesGranularity": "SPECIFIC",
        "TimePointGranularity": "ALL"
    },
    "DataSource": {
         "S3Config": {
            "Path": [S3_path_to_file],
            "RoleArn":[role-to-access-s3-bucket]
         }
      },
    "Schema": {
         "Attributes": [
            {
               "AttributeName": "item_id",
               "AttributeType": "string"
            },
                        {
               "AttributeName": "store_location",
               "AttributeType": "string"
            }
         ]
      },
```

```
}
```

Specifying time points

> **Note**
>
> If you do not specify time points (`"TimePointGranularity": "ALL"`), Amazon Forecast will consider the entire forecast horizon when calculating Impact scores.

When you specify time points for Forecast Explainability, Amazon Forecast calculates Impact scores for attributes for that specific time range. You can specify up to 500 consecutive time points within the forecast horizon.

For example, a retailer may want to know how their attributes impact sales during the winter. In this use case, they would specify the time points to span only the winter period in the forecast horizon.

You can specify time points using the Forecast console or the Forecast Software Development Kit (SDK).

Console

To specify time series for Forecast Explainability

1. Sign in to the AWS Management Console and open the Amazon Forecast console at https://console.aws.amazon.com/forecast/.
2. From **Dataset groups**, choose your dataset group.
3. In the navigation pane, choose **Insights**.
4. Choose **Create Explainability**.
5. In the **Explainability name** field, provide a unique name for the Forecast Explainability.
6. In the **Select forecast** field, choose your forecast.
7. In the **S3 location** field, enter the location of the file with your time series.
8. In the **Data schema** field, set the *attribute name* an *attribute type* of the item ID and dimensions used in your time series.
9. In the **Time duration** field, specify the start date and end date within the calendar.

10. Choose **Create Explainability.**

SDK

To specify time series for Forecast Explainability

Using the CreateExplainability operation, provide a unique name for ExplainabilityName and provide your forecast ARN for ResourceArn. Set the start date (`StartDateTime`) and end date (`EndDateTime`) using the following timestamp format: `yyyy-MM-ddTHH:mm:ss` (example: 2015-01-01T20:00:00).

Configure the following datatypes:

- `ExplainabilityConfig` - set values for TimeSeriesGranularity to “SPECIFIC” and TimePointGranularity to “SPECIFIC”.
- `S3Config` - set the values for “Path” to the S3 location of the time series file and “RoleArn” to a role with access to the S3 bucket.
- `Schema` - define the “AttributeName” and “AttributeType” for item_id and the dimensions in your time series.

The example below shows a schema for time series using a combination of “item_id” and the “store_location” dimension.

```
{
    "ExplainabilityName" : [unique_name],
    "ResourceArn" : [forecast_arn],
    "ExplainabilityConfig" {
        "TimeSeriesGranularity": "SPECIFIC",
        "TimePointGranularity": "SPECIFIC"
    },
    "DataSource": {
         "S3Config": {
            "Path": [S3_path_to_file],
            "RoleArn":[role-to-access-s3-bucket]
         }
      },
    "Schema": {
         "Attributes": [
            {
               "AttributeName": "item_id",
```

```
                "AttributeType": "string"
            },
                        {
                "AttributeName": "store_location",
                "AttributeType": "string"
            }
        ]
      },
    "StartDateTime": "string",
    "EndDateTime": "string",
}
```

Visualizing Forecast Explainability

When creating Forecast Explainability in the console, Forecast automatically visualizes your Impact scores. When creating Forecast Explainability with the CreateExplainability operation, set `EnableVisualization` to "true" and impact scores for that Explainability resource will be visualized within the console.

Impact score visualizations last for 30 days from the date of Explainability creation. To recreate the visualization, create a new Forecast Explainability.

Exporting Forecast Explainability

> **Note**
>
> Export files can directly return information from the Dataset Import. This makes the files vulnerable to CSV injection if the imported data contains formulas or commands. For this reason, exported files can prompt security warnings. To avoid malicious activity, disable links and macros when reading exported files.

Forecast enables you to export a CSV file of Impact scores to an S3 location.

The export contains raw and normalized impact scores for the specified time series, as well as normalized aggregated impact scores for all specified time series and all specified time points. If you didn't specify time points, the impact scores are already aggregated for all time points in your forecast horizon.

item_id	timestamp	Price-RawImpactScore	Promo-RawImpactScore	StoreLocation-RawImpactScore	Holiday_US-RawImpactScore	Price-NormalizedImpactScore	Promo-NormalizedImpactScore	StoreLocation-NormalizedImpactScore	Holiday_US-NormalizedImpactScore
Aggregate	Aggregate	NaN	NaN	NaN	NaN	-0.4967	0.6072	-0.2302	0
1	Aggregate	-0.0296	0.0572	-0.0797	0	1	0.176	-0.5442	0
2	Aggregate	57.5804	150.0358	4.0403	0	0.3838	1	-0.0263	0
3	Aggregate	-0.0751	0.025	-0.002	0	0.7174	0.3335	-1	0
1	2015-01-26T00:00:00Z	-6.8968	-12.9865	-0.2756	0	-0.1178	-0.2219	-0.0047	0
1	2016-05-09T00:00:00Z	-1.9732	-11.4329	-14.744	0	-0.0337	-0.1953	-0.2519	0
1	2015-03-09T00:00:00Z	-2.8406	-13.0931	-1.9269	0	-0.0485	-0.2237	-0.0329	0
1	2015-06-22T00:00:00Z	-2.3571	-8.5324	-14.4815	0	-0.0403	-0.1458	-0.2474	0
1	2016-08-29T00:00:00Z	-3.1274	-5.0817	-19.1643	0	-0.0534	-0.0868	-0.3274	0
1	2016-12-12T00:00:00Z	-11.4177	-12.4537	-4.4453	0	-0.1951	-0.2128	-0.076	0
1	2017-05-01T00:00:00Z	-2.4247	-7.2943	-18.8764	0	-0.0414	-0.1246	-0.3225	0
1	2017-07-03T00:00:00Z	-1.4287	-13.7995	-6.1356	0	-0.0244	-0.2358	-0.1048	0
1	2016-04-11T00:00:00Z	-6.00E-04	-0.0012	-0.0043	0	-0.0368	-0.0776	-0.2842	0
2	2015-03-16T00:00:00Z	-2.4852	-7.6812	-7.4003	0	-0.0425	-0.1312	-0.1264	0
2	2016-05-09T00:00:00Z	-0.0037	-1.00E-04	-5.00E-04	0	-0.2692	-0.0038	-0.0394	0
2	2015-05-04T00:00:00Z	10.4991	-10.0409	-2.1008	0	0.1794	-0.1716	-0.0359	0
2	2014-11-03T00:00:00Z	-6.00E-04	-0.0055	-0.0022	0	-0.0416	-0.3628	-0.1457	0
2	2015-02-09T00:00:00Z	-0.0024	-0.0011	-0.0025	0	-0.1599	-0.0726	-0.1632	0
2	2014-08-25T00:00:00Z	-3.7739	-2.4773	-2.4888	0	-0.0645	-0.0423	-0.0425	0
2	2017-12-18T00:00:00Z	-6.00E-04	-0.0047	-0.0041	0	-0.0424	-0.3063	-0.2683	0
2	2015-05-11T00:00:00Z	-3.00E-04	-2.00E-04	-3.00E-04	0	-0.0227	-0.0146	-0.0231	0
2	2014-12-08T00:00:00Z	-5.4927	-8.1132	-0.0168	0	-0.0939	-0.1386	-3.00E-04	0
2	2015-04-06T00:00:00Z	-0.0018	-0.0016	-3.00E-04	0	-0.1172	-0.1022	-0.0197	0
3	2014-10-20T00:00:00Z	-0.0025	-0.0027	-0.0034	0	-0.1639	-0.1804	-0.2227	0
3	2015-04-13T00:00:00Z	-21.7456	3.1561	-16.2541	0	-0.3716	0.0539	-0.2777	0
3	2018-04-23T00:00:00Z	-1.2579	-0.2137	-5.6459	0	-0.0215	-0.0037	-0.0965	0
3	2015-01-19T00:00:00Z	-0.0031	-9.00E-04	-0.0045	0	-0.2304	-0.0691	-0.3313	0
3	2017-01-30T00:00:00Z	-0.0036	-0.0034	-0.0023	0	-0.2414	-0.2229	-0.1542	0
3	2017-05-08T00:00:00Z	16.5512	-2.5499	-15.8288	0	0.2828	-0.0436	-0.2705	0
3	2016-05-30T00:00:00Z	-0.0015	-0.0027	-4.00E-04	0	-0.1078	-0.2004	-0.0276	0
3	2018-05-28T00:00:00Z	-15.528	-15.369	-0.4334	0	-0.2653	-0.2626	-0.0074	0
3	2017-06-19T00:00:00Z	-16.0061	-9.0946	10.3333	0	-0.2735	-0.1554	0.1766	0
3	2017-12-25T00:00:00Z	-8.5566	-1.8031	-2.7768	0	-0.1462	-0.0308	-0.0474	0

You can export Forecast Explainability using the Amazon Forecast Software Development Kit (SDK) and the Amazon Forecast console.

Console

To export Forecast Explainability

1. Sign in to the AWS Management Console and open the Amazon Forecast console at https://console.aws.amazon.com/forecast/.
2. From **Dataset groups**, choose your dataset group.
3. In the navigation pane, choose **Insights**.
4. Select your Explainability.
5. From the **Actions** drop-down, choose **Export**.
6. In the **Export name** field, provide a unique name for the Forecast Explainability export.
7. In the **S3 explainability export location**field, enter the S3 location to export the CSV file.
8. In the **IAM Role** field, choose a role with access to the chosen S3 location.
9. Choose **Create Explainability Export**.

SDK

To export Forecast Explainability

Using the CreateExplainabilityExport operation, specify your S3 location and IAM role in the `Destination` object, along with the `ExplainabilityArn` and `ExplainabilityExportName`.

For example:

```
{
   "Destination": {
      "S3Config": {
         "Path": "s3://bucket/example-path/",
         "RoleArn": "arn:aws:iam::000000000000:role/ExampleRole"
      }
   },
   "ExplainabilityArn": "arn:aws:forecast:region:explainability/example",
   "ExplainabilityName": "Explainability-export-name",
}
```

Restrictions and best practices

Consider the following restrictions and best practices when working with Forecast Explainability.

- **Forecast Explainability is only available for some Forecasts generated from AutoPredictor -** You cannot enable Forecast Explainability for Forecasts generated from legacy predictors (AutoML or manual selection). See Upgrading to AutoPredictor.
- **Forecast Explainability is not available for all models** - The ARIMA (AutoRegressive Integrated Moving Average), ETS (Exponential Smoothing State Space Model), and NPTS (Non-Parametric Time Series) models do not incorporate external time series data. Therefore, these models do not create an explainability report, even if you include the additional datasets.
- **Explainability requires attributes** - Your predictor must include at least one of the following: related time series, item metadata, Holidays, or the Weather Index.
- **Impact scores of zero denote no impact** - If one or more attribute has an impact score of zero, then these attributes have no significant impact on forecast values. Scores can also be zero if the AutoPredictor used only a non-ML algorithm, or you did not provide related time series or item metadata.
- **Specify a maximum of 50 time series** - You can specify up to 50 time series per Forecast Explainability.

- **Specify a maximum of 500 time points** - You can specify up to 500 consecutive time points per Forecast Explainability.
- **Forecast also calculates some aggregated Impact scores** - Forecast will also provide aggregated impact scores for the specified time series and time points.
- **Create multiple Forecast Explainability resources for a single Forecast** - If you want impact scores for more than 50 time series or 500 time points, you can create Explainability resources in batches to span a larger range.
- **Compare Raw impact scores across different Forecast Explainability resources** - Raw impact scores can be directly compared across Explainability resources from the same forecast.
- **Forecast Explainability visualizations are available for 30 days after creation** - To view the visualization after 30 days, create a new Forecast Explainability with the same configuration.

What-if analysis

A what-if analysis is a tool to help investigate and explain how different scenarios might affect the baseline forecast created by Amazon Forecast. The baseline forecast is the forecast that is created by Amazon Forecast based on the original related time series that you provide.

A what-if analysis creates a series of what-if forecasts based on how you chose to modify the related time series. Those what-if forecasts are compared and contrasted with the baseline forecast to help you understand how specific changes might impact your model.

There are two methods for creating a modified related time series. You can either provide a modified related time series in an Amazon S3 path or specify a set of transformations to the existing related time series. When you specify a set of transformations, a copy of the original related time series is created to contain these changes.

The transformations allow you to create a subset of the related time series and modify specific attributes of the related time series. For more information, see the section called “Replacement Dataset” and the section called “Transformation Functions”.

Topics

Creating a what-if analysis

A what-if analysis explores how changes to the baseline related time series can impact a forecast. You can only create a what-if analysis from a forecast that uses an AutoPredictor. After you create a what-if analysis, you create one or more what-if forecasts. Compare the what-if forecasts and the baseline forecast, and then export one or more what-if forecasts.

> **Note**
>
> Your data must be in comma-separated values (CSV) format to create a what-if analysis.

Topics

- Create a what-if analysis
- Create a what-if forecast
- Compare your what-if forecasts
- Export your what-if forecasts
- Query your what-if forecasts

Create a what-if analysis

You can create a what-if analysis using the Forecast console or the Forecast Software Development Kit (SDK).

Console

To create a what-if analysis, complete the following steps:

1. Create a forecast that is trained using an AutoPredictor.
2. Open the dataset group dashboard that contains the forecast you're interested in.
3. Choose **Explore what-if analysis**.
4. On the **What-if analysis** tab of the Insights page, choose **Create**.
5. Provide a unique name in the **What-if analysis name** field and choose the baseline forecast for this analysis.
6. In the **Item selection** area, select whether you want to automatically include all items in the analysis or specify the items to include with a file.

   If you choose **Select items with file**, you must provide a dataset that contains just the items that you want to modify in the what-if forecasts. For more information, see Specifying time series.
7. Choose **Create what-if analysis**. A banner at the top of the What-if Analysis page will display the status of the what-if analysis creation job.

SDK

Using the CreateWhatIfAnalysisoperation, provide a unique name for WhatIfAnalysisName and provide the forecast ARN of the baseline forecast for ForecastArn. The example below shows a schema for time series using a combination of “item_id” and the “store_location” dimension. For more information, see Specifying time series.

```
{
  "ForecastArn": "arn:aws:forecast:region:acctNumber:forecast/baselineForecast",
  "WhatIfAnalysisName": "unique_name",
  "TimeSeriesSelector": {
    "TimeSeriesIdentifiers": {
      "DataSource": {
        "S3Config": {
          "Path": "s3://bucket/example-path",
          "RoleArn": "arn:aws:iam::000000000000:role/ExampleRole"
        }
      },
      "Schema": {
        "Attributes": [
          {
            "AttributeName": "item_id",
            "AttributeType": "string"
          }
          {
            "AttributeName": "store_location",
            "AttributeType": "string"
          }
        ]
      }
    }
  }
}
```

Create a what-if forecast

You can create a what-if forecast using the Forecast console or the Forecast Software Development Kit (SDK).

Console

To create a what-if forecast, complete the following steps:

1. On the **What-if analysis** tab of the Insights page, choose the what-if analysis that you are interested in.
2. In the **What-if forecast** section, choose **Create**.

3. On the Create what-if forecast page, provide a unique **What-if forecast name** and choose either **Use transformation functions** or **Define the what-if forecast with a replacement dataset**. For more information, see the section called “Replacement Dataset” and the section called “Transformation Functions”.

   a. If you choose **Use transformation functions**, you must use the **Transformation function builder** to select and modify the rows that are included in the what-if forecast. All transformations are applied in the order they are specified. Conditions are applied in the order they are specified, and are joined with an AND operation. The transformation is applied only when all of the conditions are met.

   b. If you choose **Define the what-if forecast with a replacement dataset**, you must provide a replacement dataset that contains only the rows that you want to change for the what-if forecast.

4. Choose **Create**.

SDK - Transformation Function

Using the CreateWhatIfForecast operation, provide a unique name for WhatIfAnalysisName and provide your forecast ARN for ForecastArn. The example below shows a schema for a transformation to "price" when the "store_location" is not "tacoma".

```
{
  "WhatIfAnalysisArn": "arn:aws:forecast:us-west-2:666488130463:what-if-analysis/
jan2020forecast/PromotionAnalysis_01G8MB3PZM89J9V1VEXCC0BS63",
  "WhatIfForecastName": "unique_name",
  "TimeSeriesTransformations": [
    {
      "Action": {
        "AttributeName": "price",
        "Operation": "MULTIPLY",
        "Value": 0.85
      },
      "TimeSeriesConditions": [
        {
          "AttributeName": "store_location",
          "AttributeValue": "tacoma",
          "Condition": "NOT_EQUALS"
        }
      ]
    }
```

```
  ]
}
```

In this example, `jan2020forecast` is the baseline forecast and `PromotionAnalysis_01G8MB3PZM89J9V1VEXCC0BS63` is the what-if analysis name.

You can also specify a replacement dataset with the TimeSeriesReplacementsDataSource operation.

SDK - Replacement Dataset

Using the CreateWhatIfForecast operation, provide a unique name for WhatIfAnalysisName and provide your forecast ARN for ForecastArn. The example below shows a schema for a replacement datasource.

```
{
  "WhatIfAnalysisArn": "arn:aws:forecast:us-west-2:666488130463:what-if-analysis/
jan2020forecast/PromotionAnalysis_01G8MB3PZM89J9V1VEXCC0BS63",
  "WhatIfForecastName": "unique_name",
  "TimeSeriesReplacementsDataSource": {
    "S3config": {
      "Path" : "s3://bucket-name/replacementDatasource.csv",
      "RoleArn": "arn:aws:iam::acct-id:role/Role"
    },
    "Schema": {
      Attributes" : [
        {
          "AttributeName": "item_id",
          "AttributeType": "string"
        },
        {
          "AttributeName": "timestamp",
          "AttributeType": "timestamp"
        },
        {
          "AttributeName": "price",
          "AttributeType": "float"
        },
        {
          "AttributeName": "stock_count",
          "AttributeType": "integer"
        },
      ]
```

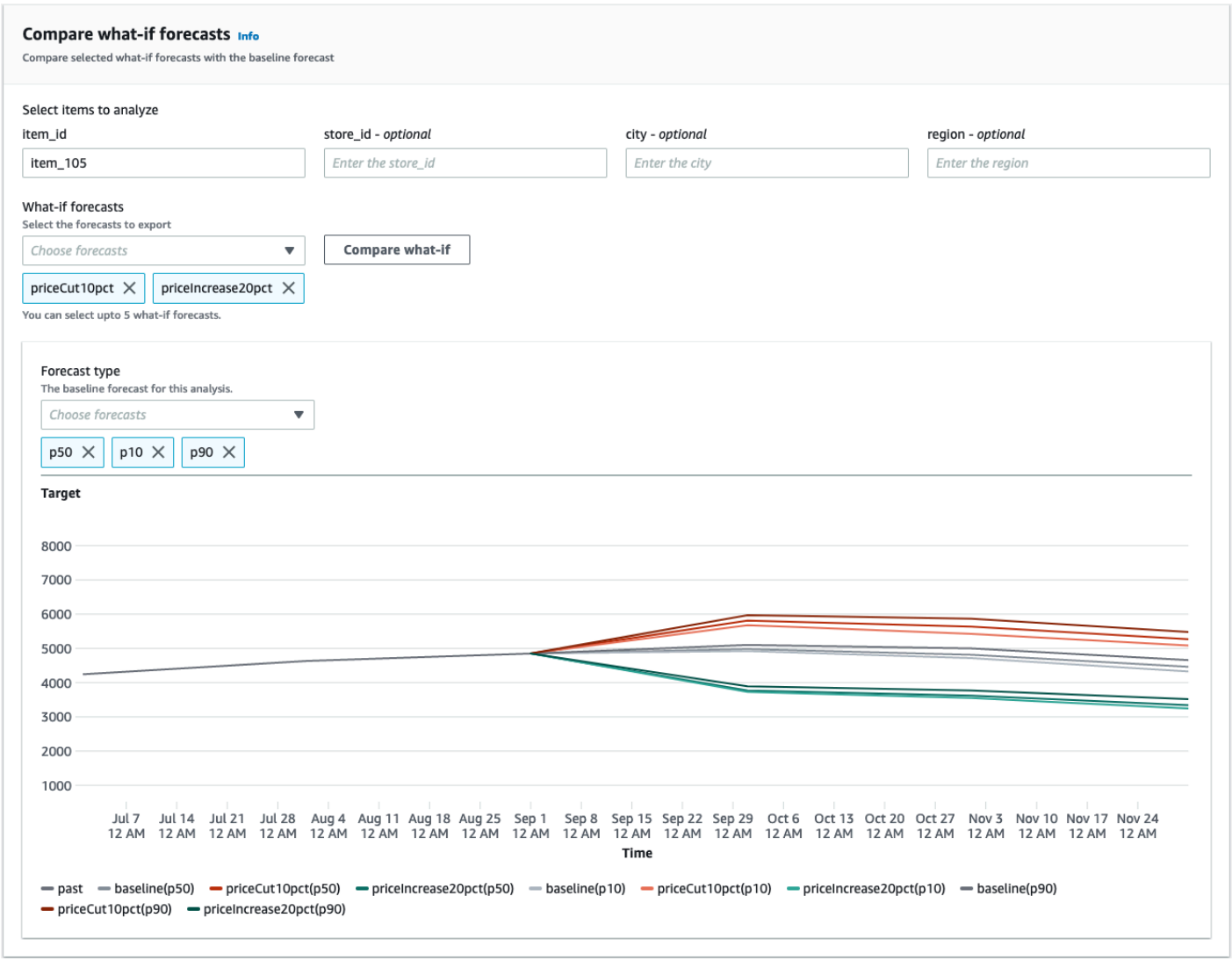
```
    }
  }
}
```

You can also specify changes to the related time series with the TimeSeriesTransformation operation.

Compare your what-if forecasts

To compare the what-if forecasts, complete the following steps in the Forecast console:

1. On the **What-if analysis** tab of the Insights page, choose the what-if analysis that you are interested in.
2. In the **Compare what-if forecasts** section, specify the item to analyze, one or more **What-if forecasts**, and at least one **Forecast type**.



In this example, there are two what-if forecasts, `priceCut10pct` and `priceIncrease20pct`, that are compared at the `p50`, `p10`, and `p90` forecast types for `item_105`. The graph allows you to see how these forecasts compare against the baseline time series.

3. Hover over the chart to investigate how the what-if forecasts compare to the baseline forecast.

Export your what-if forecasts

You can export a what-if forecast using the Forecast console or the Forecast Software Development Kit (SDK).

Console

To export the what-if forecasts, complete the following steps:

1. On the **What-if analysis** tab of the Insights page, choose the what-if analysis that you are interested in.
2. In the **What-if forecast export** section, choose **Create export**.
3. On the Create what-if forecast export page, provide a unique **What-if forecast export name**, specify the **What-if forecasts** to include, choose an **Export location**, and provide the **IAM role**.
4. Choose **Create export**.

SDK

Using the CreateWhatIfForecastExport operation, configure the "Destination" to point at the Amazon S3 bucket that will contain the export. Specify which what-if forecasts to export, and provide a unique name for the export.

```
{
  "WhatIfForecastArns": [ "arn:aws:forecast:region:acctNumber:what-if-forecast/
id1" , "arn:aws:forecast:region:acctNumber:what-if-forecast/id2" ],
  "WhatIfForecastExportName": "unique_export_name",
  "Destination": {
    "S3Config": {
      "Path": "s3://bucket/example-path",
      "RoleArn": "arn:aws:iam::000000000000:role/ExampleRole"
    }
  },
}
```

Query your what-if forecasts

You can query a what-if forecast using the QueryWhatIfForecast operation. By default, the complete range of the forecast is returned. You can request a specific date range within the complete forecast.

When you query a what-if forecast you must specify filtering criteria. A filter is a key-value pair. The key is one of the schema attribute names (including forecast dimensions) from one of the datasets used to create the forecast. The value is a valid values for the specified key. You can specify multiple key-value pairs. The returned what-if forecast will only contain items that satisfy all the criteria.

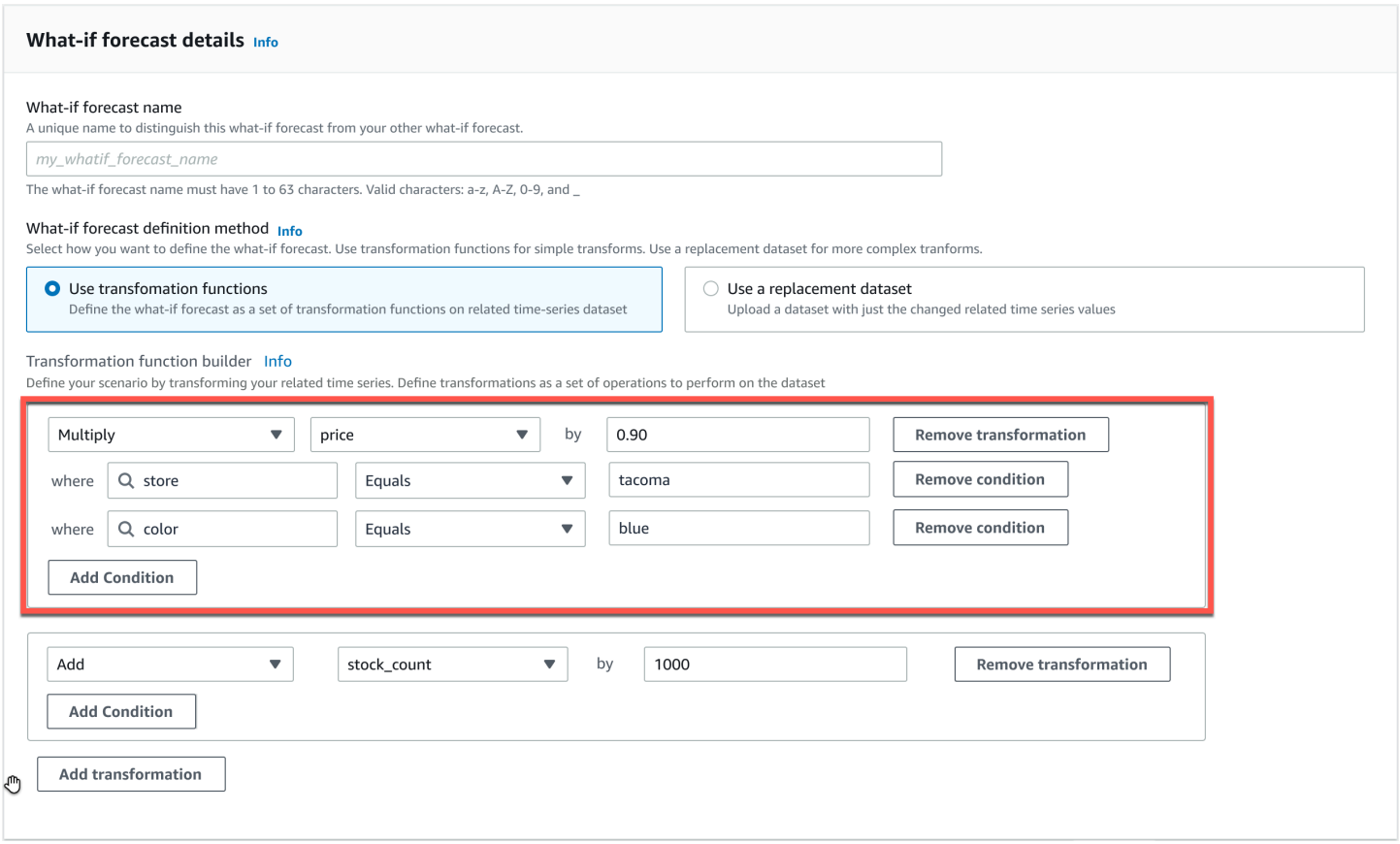
For example, use this code to get the what-if forecast for `product_42`.

```
{
          "Filters": {
          "item_id" : "product_42"
          },
          "WhatIfForecastArn": "arn:aws:forecast:region:acctNumber:what-if-forecast/
id1"
}
```

Transformation Functions

A transformation function is a set of operations that select and modify the rows in a related time series. You select the rows that you want with a condition operation. You then modify the rows with a transformation operation. All conditions are joined with an AND operation, meaning that all conditions must be true for the transformation to be applied. Transformations are applied in the order that they are listed.

When you create a what-if forecast, use the **Transformation function builder** to specify the conditions and transformations that you want to apply. The image below illustrates this functionality.



In the highlighted section, the `price` column is multiplied by 0.90 (i.e., a 10% discount) at the store in `tacoma` (i.e., Tacoma, Washington) for items that are colored `blue`. To do this, Amazon Forecast first creates a subset of the baseline related time series to contain only the rows of `store` that equal `tacoma`.

That subset is further pared down to include only the rows of `color` that equal `blue`. Finally, all values in the `price` column are multiplied by 0.90 to create a new related time series to use in the what-if forecast.

Amazon Forecast supports the following conditions:

- `EQUALS` - The value in the column is the same as the value that was provided in the condition.
- `NOT_EQUALS` - The value in the column isn't the same as the value that was provided in the condition.
- `LESS_THAN` - The value in the column is less than the value that was provided in the condition.
- `GREATER_THAN` - The value in the column is greater than the value that was provided in the condition.

Amazon Forecast supports the following actions:

- ADD - Adds the provided value to all rows in the column.
- SUBTRACT - Subtracts the provided value from all rows in the column.
- MULTIPLY - Multiplies all rows in the column by the value provided.
- DIVIDE - Divides all rows in the column by the value provided.

What follows are examples of how you can specify a time series transformation using the SDK.

Example 1

This example applies a 10% discount to all items in Seattle store. Note that "City" is a forecast dimension.

```
TimeSeriesTransformations=[
  {
    "Action": {
      "AttributeName": "price",
      "Operation": "MULTIPLY",
      "Value": 0.90
      },
    "TimeSeriesConditions": [
      {
        "AttributeName": "city",
        "AttributeValue": "seattle",
        "Condition": "EQUALS"
      }
    ]
  }
]
```

Example 2

This example applies a 10% discount on all items in the "electronics" category. Note that "product_category" is an item metadata.

```
TimeSeriesTransformations=[
  {
    "Action": {
      "AttributeName": "price",
      "Operation": "MULTIPLY",
```

```
      "Value": 0.90
      },
    "TimeSeriesConditions": [
      {
        "AttributeName": "product_category",
        "AttributeValue": "electronics",
        "Condition": "EQUALS"
      }
    ]
  }
]
```

Example 3

This example applies a 20% markup on the specific item_id BOA21314K.

```
TimeSeriesTransformations=[
  {
    "Action": {
      "AttributeName": "price",
      "Operation": "MULTIPLY",
      "Value": 1.20
      },
    "TimeSeriesConditions": [
      {
        "AttributeName": "item_id",
        "AttributeValue": "BOA21314K",
        "Condition": "EQUALS"
      }
    ]
  }
]
```

Example 4

This example adds $1 to all items in the Seattle and Bellevue stores.

```
TimeSeriesTransformations=[
  {
    "Action": {
      "AttributeName": "price",
      "Operation": "ADD",
      "Value": 1.0
```

```
      },
    "TimeSeriesConditions": [
      {
        "AttributeName": "city",
        "AttributeValue": "seattle",
        "Condition": "EQUALS"
      }
    ]
  },
  {
    "Action": {
      "AttributeName": "price",
      "Operation": "ADD",
      "Value": 1.0
      },
    "TimeSeriesConditions": [
      {
        "AttributeName": "city",
        "AttributeValue": "bellevue",
        "Condition": "EQUALS"
      }
    ]
  }
]
```

Example 5

This example subtracts $1 from all items in Seattle in the month of September, 2022.

```
TimeSeriesTransformations=[
  {
    "Action": {
      "AttributeName": "price",
      "Operation": "SUBTRACT",
      "Value": 1.0
      },
    "TimeSeriesConditions": [
      {
        "AttributeName": "city",
        "AttributeValue": "seattle",
        "Condition": "EQUALS"
      },
      {
        "AttributeName": "timestamp",
```

```
        "AttributeValue": "2022-08-31 00:00:00",
        "Condition": "GREATER_THAN"
      },
      {
        "AttributeName": "timestamp",
        "AttributeValue": "2022-10-01 00:00:00",
        "Condition": "LESS_THAN"
      }
    ]
  }
]
```

Example 6

In this example, the price is first multiplied by 10, then $5 is subtracted from the price. Note that actions are applied in the order that they are declared.

```
TimeSeriesTransformations=[
  {
    "Action": {
      "AttributeName": "price",
      "Operation": "MULTIPLY",
      "Value": 10.0
      },
    "TimeSeriesConditions": [
      {
        "AttributeName": "city",
        "AttributeValue": "seattle",
        "Condition": "EQUALS"
      }
    ]
    },
    {
    "Action": {
      "AttributeName": "price",
      "Operation": "SUBTRACT",
      "Value": 5.0
      },
    "TimeSeriesConditions": [
      {
        "AttributeName": "city",
        "AttributeValue": "seattle",
        "Condition": "EQUALS"
```

```
      }
    ]
   }
]
```

Example 7

This example creates an empty set, so the action is not applied to any time series. This code tries to modify the price of all items at the stores in Seattle and Bellevue. Because conditions are joined with the AND operation, and a store can exist in only one city, the results are an empty set. Therefore, the action is not applied.

```
TimeSeriesTransformations=[
  {
    "Action": {
      "AttributeName": "price",
      "Operation": "MULTIPLY",
      "Value": 10.0
      },
    "TimeSeriesConditions": [
      {
        "AttributeName": "city",
        "AttributeValue": "seattle",
        "Condition": "EQUALS"
      },
      {
        "AttributeName": "city",
        "AttributeValue": "bellevue",
        "Condition": "EQUALS"
      },
    ]
  }
]
```

For an example of how to apply a condition to multiple attributes, see Example 4.

Example 8

Transformation conditions that use a timestamp apply to the boundary-aligned data, not the raw data. For example, you input your data hourly and forecast daily. In this case, Forecast aligns timestamps to the day, so `2020-12-31 01:00:00` is aligned to `2020-12-31 00:00:00`. This code will create an empty set because it does not specify the timestamp at the boundary-aligned timestamp.

```
TimeSeriesTransformations=[
  {
    "Action": {
      "AttributeName": "price",
      "Operation": "MULTIPLY",
      "Value": 10.0
      },
    "TimeSeriesConditions": [
      {
        "AttributeName": "timestamp",
        "AttributeValue": "2020-12-31 01:00:00",
        "Condition": "EQUALS"
      },
    ]
  }
]
```

Replacement Dataset

A replacement dataset is a modified version of the baseline related time series that contains only the values that you want to change in a what-if forecast. The replacement dataset must contain the forecast dimensions, item identifiers, and timestamps in the baseline related time series, as well as at least 1 changed time series. This dataset is merged with the baseline related time series to create a transformed dataset that is used for the what-if forecast. The replacement dataset must be in CSV format.

This dataset should not contain duplicate timestamps for the same time series.

What follows are several examples of how you can specify a replacement time series and how those specifications are interpreted. Consider the case where you are forecasting daily and the forecast horizon is 2022-08-01 through 2022-08-03. The baseline related time series for all examples is given in the following table.

item_id	timestamp	price	stock_count
item_1	2022-08-01	100	50
item_1	2022-08-02	100	50

item_id	timestamp	price	stock_count
item_1	2022-08-03	100	50
item_2	2022-08-01	75	500
item_2	2022-08-02	75	500
item_2	2022-08-03	75	500

Unchanged values

To apply a 10% discount on item_1 for 2022-08-02 and 2022-08-03, it is sufficient to specify the following for the replacement dataset:

Replacement dataset

item_id	timestamp	price
item_1	2022-08-02	90
item_1	2022-08-03	90

However, it's also valid to specify unchanged values in the replacement dataset. When used as replacement datasets, each of the following three tables will yield the same results as the previously provided table.

Replacement dataset with an unchanged column

item_id	timestamp	price	stock_count
item_1	2022-08-02	90	50
item_1	2022-08-03	90	50

Replacement dataset with unchanged rows

item_id	timestamp	price
item_1	2022-08-01	100
item_1	2022-08-02	90
item_1	2022-08-03	90
item_2	2022-08-01	75
item_2	2022-08-02	75
item_2	2022-08-03	75

Replacement dataset with unchanged rows and columns

item_id	timestamp	price	stock_count
item_1	2022-08-01	100	50
item_1	2022-08-02	90	50
item_1	2022-08-03	90	50
item_2	2022-08-01	75	500
item_2	2022-08-02	75	500
item_2	2022-08-03	75	500

Missing values

Missing values in the replacement time series are replaced with values from the baseline related time series. Consider the scenario where you apply a 10% discount on item_1 for 2022-08-02 and 2022-08-03 and increase the stock of item_2 on 2022-08-01. This replacement dataset is sufficient:

Replacement dataset with missing values

item_id	timestamp	price	stock_count
item_1	2022-08-02	90	
item_1	2022-08-03	90	
item_2	2022-08-01		5000

The values missing from this table are imputed from the baseline related time series.

Extraneous values

Extraneous values in the replacement time series are ignored when creating a what-if forecast. That is, values in the replacement dataset that do not correspond to values in the baseline related time series are not modeled. Consider this replacement dataset:

Replacement dataset with extraneous values

item_id	timestamp	price	stock_count
item_1	2022-08-01	100	50
item_1	2022-08-02	100	50
item_1	2022-08-03	100	50
item_2	2022-08-01	75	500
item_2	2022-08-02	75	500
item_2	2022-08-03	75	500
item_3	2022-08-01	50	125
item_3	2022-08-02	50	125
item_3	2022-08-03	50	125

The rows containing item_3 are ignored and are not part of the what-if analysis.

Historical changes

Changes in the replacement dataset that are outside of the forecast horizon are ignored. Consider this replacement dataset:

Replacement dataset with values outside the forecast horizon

item_id	timestamp	price	stock_count
item_1	2022-07-31	100	50
item_1	2022-08-01	100	50
item_1	2022-08-02	100	50
item_1	2022-08-03	100	50
item_1	2022-08-04	100	50
item_2	2022-07-31	75	500
item_2	2022-08-01	75	500
item_2	2022-08-02	75	500
item_2	2022-08-03	75	500
item_3	2022-08-04	75	500

The rows containing 2022-07-31 and 2022-08-04 are ignored and are not part of the what-if analysis.

Forecast dimensions

If you include forecast dimensions in your dataset, then you must include them in the replacement dataset. Consider this baseline related time series:

item_id	store_id	timestamp	price	stock_count
item_1	store_1	2022-08-01	100	50

item_id	store_id	timestamp	price	stock_count
item_1	store_1	2022-08-02	100	50
item_1	store_1	2022-08-03	100	50
item_1	store_2	2022-08-01	75	500
item_1	store_2	2022-08-02	75	500
item_1	store_2	2022-08-03	75	500

Therefore, the replacement dataset for a 10% discount in all stores on 2022-08-02 would be the following:

item_id	store_id	timestamp	price
item_1	store_1	2022-08-02	90
item_1	store_2	2022-08-02	67.5

Managing Resources

You can manage your Amazon Forecast resources by stopping ongoing jobs, deleting completed or failed resources, tagging resources, and setting up event notifications through Amazon EventBridge and Amazon CloudWatch Events.

Topics

Stopping Resources

The Amazon Forecast `Stop Resource` (`StopResource`) operation stops a resource job that is in progress. You can stop the following resource jobs:

- Dataset group import (`CreateDatasetImportJob`)
- Predictor training (`CreateAutoPredictor` and `CreatePredictor`)
- Predictor backtest export (`CreatePredictorBacktestExportJob`)
- Forecast (`CreateForecast`)
- Forecast export (`CreateForecastExportJob`)
- What-if analysis (`CreateWhatIfAnalysis`)
- What-if forecast (`CreateWhatIfForecast`)
- What-if forecast export (`CreateWhatIfForecastExportJob`)

You can't resume a resource job after it has stopped.

Stopping a resource ends its workflow, but doesn't delete the resource. You can still preview the resource parameters in the console and with the `Describe` operation.

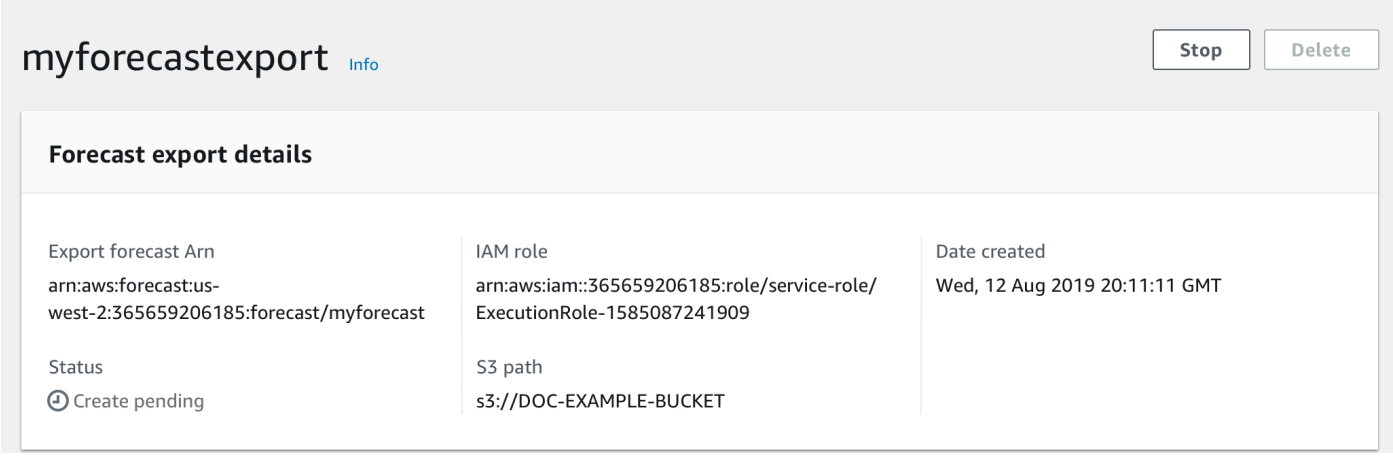
When you stop a predictor or forecast job, you are billed for resources used up to the point when the job stopped.

You can stop a resource job using the Forecast console or the AWS Software Development Kit (SDK).

Console

To stop a resource job

1. Sign in to the AWS Management Console and open the Amazon Forecast console at https://console.aws.amazon.com/forecast/.
2. In the navigation pane, choose the resource type.
3. Choose the resource job.
4. Choose **Stop**.



SDK

To stop a resource job

Using the StopResource operation, set the value of `ResourceArn` to the Amazon Resource Name (ARN) that identifies the resource job that you want to stop.

```
{
    "ResourceArn": "arn:partition:service:region:account-id:resource-id"
}
```

Deleting Resources

You can delete individual Amazon Forecast resources and entire resource trees with the Amazon Forecast console and AWS Software Development Kit (SDK).

A Forecast resource tree is a parent-child hierarchical structure. *Child resources* are resources created from other resources. For example, when you create a predictor using a dataset group, the dataset group is the parent resource and the predictor is the child resource. When deleting a Forecast resource, you must also delete its child resources.

Deleting a resource or resource tree is an irreversible action. It can't be stopped after it begins

Topics

Understanding Resource Trees

The Forecast resource tree is a parent-child hierarchical structure. *Child resources* are resources that were created from another resource. For example, when a forecast is generated from a predictor, the forecast is the child resource and the predictor is the parent resource.

To delete a Forecast resource, you must also delete its entire resource tree. This includes all child resources of the parent resource, and also the child resources of those child resources.

> **Note**
>
> Deleting a resource tree deletes only Amazon Forecast resources. It doesn't delete datasets or exported files stored in Amazon Simple Storage Service (Amazon S3).

Forecast resources have the following parent-child resource hierarchies.

For example, the resource tree of a *predictor* includes *predictor backtest jobs*, *forecasts*, and *forecast export jobs* as child resources. The resource tree of a *forecast* includes only *forecast export jobs* as child resources.

The *dataset* resource tree includes *dataset import jobs* as a child resource. Neither *datasets* or *dataset import jobs* are part of the *dataset group* resource tree.

Parent Resource	Child Resources
Dataset	Dataset import jobs
Dataset group	Predictors, predictor backtest export jobs, predictor explainabilities, predictor explainability exports, forecasts, forecast export jobs, forecast explainabilities, forecast explainability exports
Predictor	Predictor backtest export jobs, predictor explainabilities, predictor explainability exports, forecasts, forecast export jobs, forecast explainabilities, forecast explainability exports
Forecast	Forecast export jobs, forecast explainabilities, forecast explainability exports, what-if analyses, what-if forecasts, what-if forecast exports
Explainability	Explainability exports
What-if analysis	what-if forecasts, what-if forecast exports
What-if forecast	what-if forecast exports

If a resource doesn't have any child resources, you can delete it individually. If a resource has child resources, you must delete the entire resource tree.

When using the Forecast console, you are automatically prompted to delete the entire resource tree when you delete a resource with child resources. When using the AWS Software Development Kit (SDK), use the DeleteResourceTree operation to delete a resource tree.

Deleting Individual Resources

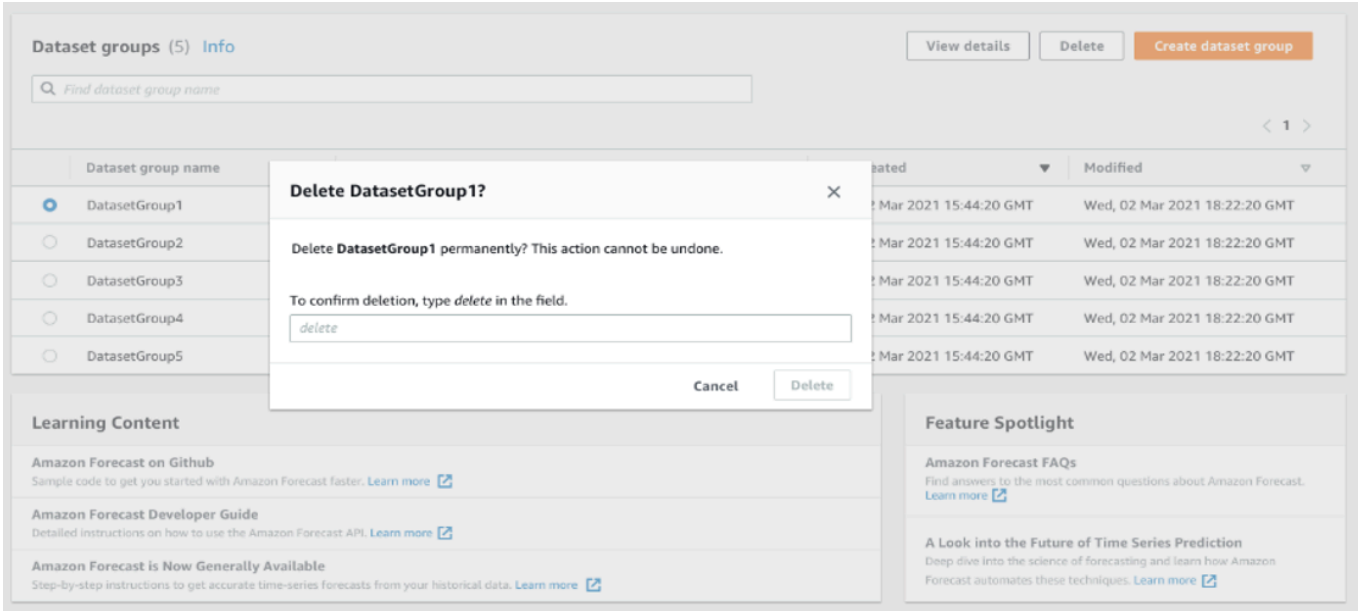
You can delete an individual resource if it's not associated with any child resources. For example, you can delete an individual predictor that has not been used to create any forecasts or export jobs.

You can delete resources using the Amazon Forecast console or the AWS Software Development Kit (SDK).

Console

To delete a resource

1. Sign in to the AWS Management Console and open the Amazon Forecast console at https://console.aws.amazon.com/forecast/.
2. In the navigation pane, choose the resource type of the resource that you want to delete.
3. Choose the resource and choose **Delete**.
4. In the confirmation field, type **`delete`**.
5. Choose **Delete**.



SDK

To delete a resource

The operation that you use to delete a resource depends on its resource type. Specify the resource Amazon Resource Name (ARN) in the operation for the resource type the you want to delete:

- DeleteDataset
- DeleteDatasetGroup
- DeleteDatasetImportJob
- DeletePredictor
- DeletePredictorBacktestExportJob
- DeleteForecast
- DeleteForecastExportJob
- DeleteExplainability

For example, to delete a predictor with the DeletePredictor operation, specify the value of `PredictorArn` to the ARN of the predictor that you want to delete.

```
{
    "PredictorArn": arn:partition:service:region:account-id:resource-id
}
```

Deleting Resource Trees

Deleting a resource tree deletes the parent resource and all associated child resources. For example, you can delete a predictor and all child resources — predictor backtest export jobs, forecasts, and forecast export jobs — associated with the predictor. You delete a resource tree by specifying the parent resource.

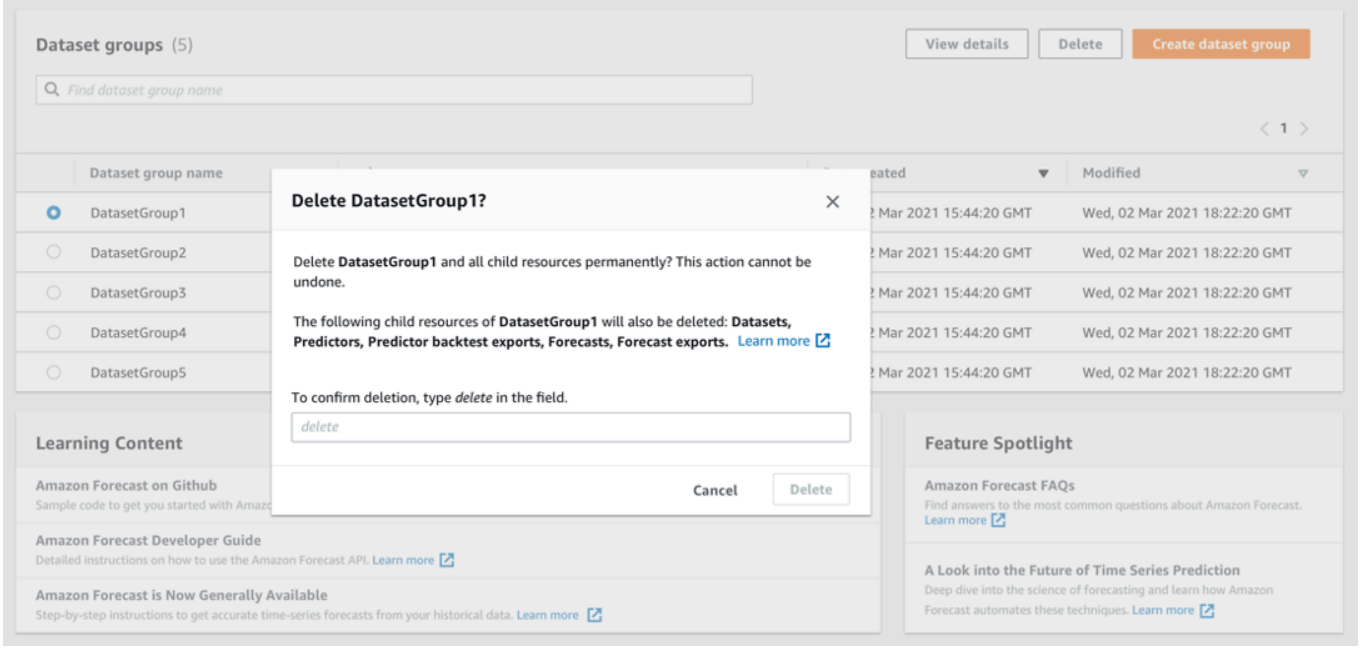
You can delete resource trees using the Amazon Forecast console or the AWS Software Development Kit (SDK).

Console

To delete a resource tree

1. Sign in to the AWS Management Console and open the Amazon Forecast console at https://console.aws.amazon.com/forecast/.

2. In the navigation pane, choose the resource type of the parent resource.
3. Choose the parent resource that you want to delete. and choose **Delete**.
4. In the confirmation field, type **`delete`**.
5. Choose **Delete**.



SDK

To delete a resource tree

To delete a resource tree, use the DeleteResourceTree operation. Set the value of `ResourceArn` to the Amazon Resource Name (ARN) of the parent resource.

```
{
    "ResourceArn": arn:partition:service:region:account-id:resource-id
}
```

Tagging Amazon Forecast Resources

A *tag* is a label that you optionally define and associate with AWS resources, including certain types of Amazon Forecast resources. Tags can help you categorize and manage resources in different ways, such as by purpose, owner, environment, or other criteria. For example, you can use tags

to apply policies or automation, or to identify resources that are subject to certain compliance requirements. You can add tags to the following types of Forecast resources:

- Dataset groups
- Datasets
- Dataset import jobs
- Predictors
- Predictor export jobs
- Forecasts
- Forecast export jobs
- What-if Analyses
- What-if Forecasts
- What-if Forecast export jobs

A resource can have as many as 50 tags.

Managing Tags

Each tag consists of a required tag key and an optional tag value, both of which you define. A tag key is a general label that acts as a category for more specific tag values. A tag value acts as a descriptor for a tag key. For example, if you have two versions of a Forecast dataset import job (one for internal testing and another for production), you might assign an `Environment` tag key to both projects. The value of the `Environment` tag key might be `Test` for one version of the dataset import job and `Production` for the other version.

A tag key can contain as many as 128 characters. A tag value can contain as many as 256 characters. The characters can be Unicode letters, numbers, white space, or one of the following symbols: _ . : / = + -. The following additional restrictions apply to tags:

- Tag keys and values are case sensitive.
- For each associated resource, each tag key must be unique and it can have only one value.
- Do not use `aws:`, `AWS:`, or any upper or lowercase combination of such as a prefix for keys, because it is reserved for AWS use. You cannot edit or delete tag keys with this prefix. Values can have this prefix. If a tag value has `aws` as its prefix but the key does not, then Forecast considers

it to be a user tag and will count against the limit of 50 tags. Tags with only the key prefix of `aws` do not count against your tags per resource limit.

- You can't update or delete a resource based only on its tags. You must also specify the Amazon Resource Name (ARN) or resource ID, depending on the operation that you use.
- You can associate tags with public or shared resources. However, the tags are available only for your AWS account, not any other accounts that share the resource. In addition, the tags are available only for resources that are located in the specified AWS Region for your AWS account.

To add, display, update, and remove tag keys and values from Forecast resources, you can use the AWS Command Line Interface (AWS CLI), the Forecast API, or an AWS SDK.

Using Tags in IAM Policies

After you start implementing tags, you can apply tag-based, resource-level permissions to AWS Identity and Access Management (IAM) policies and API operations. This includes operations that support adding tags to resources when resources are created. By using tags in this way, you can implement granular control of which groups and users in your AWS account have permission to create and tag resources, and which groups and users have permission to create, update, and remove tags more generally.

For example, you can create a policy that allows a user to have full access to all of the Forecast resources where their name is a value in the `Owner` tag for the resource.

```
{
   "Version": "2012-10-17",
   "Statement": [
      {
         "Sid": "ModifyResourceIfOwner",
         "Effect": "Allow",
         "Action": "forecast:*",
         "Resource": "*",
         "Condition": {
            "StringEqualsIgnoreCase": {
               "aws:ResourceTag/Owner": "${aws:username}"
            }
         }
      }
   ]
}
```

The following example shows how to create a policy to allow creating and deleting a dataset. These operations are allowed only if the user name is `johndoe`.

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": [
                "forecast:CreateDataset",
                "forecast:DeleteDataset"
            ],
            "Resource": "arn:aws:forecast:*:*:dataset/*",
            "Condition": {
                "StringEquals": {"aws:username" : "johndoe"}
            }
        },
        {
            "Effect": "Allow",
            "Action": "forecast:DescribeDataset",
            "Resource": "*"
        }
    ]
}
```

If you define tag-based, resource-level permissions, the permissions take effect immediately. This means that your resources are more secure as soon as they're created, and you can quickly start enforcing the use of tags for new resources. You can also use resource-level permissions to control which tag keys and values can be associated with new and existing resources. For more information, see Controlling Access Using Tags in the *AWS IAM User Guide*.

Adding Tags to Resources

The following examples show how to add a tag to Forecast resources by using the AWS CLI and the AWS Management Console.

AWS CLI

To add a tag when creating a new Forecast resource with the AWS CLI, use the appropriate `create` command for the resource and include the `tags` parameter and values. For example, the following command from the creates a new dataset group named `myDatasetGroup` for a

CUSTOM domain, and adds the following tags: An `Environment` tag key with a `Test` tag value, and a `Owner` tag key and a `xyzCorp` value.

```
aws forecast create-dataset-group \
--dataset-group-name myDatasetGroup \
--dataset-arns arn:aws:forecast:region:acct-id:dataset/dataset_name \
--domain CUSTOM \
--tags Key=Environment,Value=Test Key=Owner,Value=xyzCorp
```

For information about the commands that you can use to create a Forecast resource, see the Forecast AWS CLI Command Reference.

To add a tag to an existing resource, use the `tag-resource` command and specify the ARN of the resource and provide the tag key and value in the `tags-model` parameter.

```
aws forecast tag-resource \
--resource-arn resource ARN \
--tags Key=key,Value=value
```

AWS Management Console

When you create a resource in Forecast, you can add optional tags. The following example adds a tag to a dataset group. Adding tags to other resources follows a similar pattern.

To add tags to a new dataset group

1. Sign in to the AWS Management Console and open the Amazon Forecast console at https://console.aws.amazon.com/forecast/.
2. Choose **Create dataset group**.
3. For **Dataset group name**, enter a name.
4. For **Forecasting domain**, choose a domain.
5. Choose **Add new tag**.
6. For **Key** and **Value**, enter appropriate values.

   For example, **`Environment`** and **`Test`**, respectively.
7. To add more tags, choose **Add new tag**.

   You can add up to 50 tags to a resource.

8. Choose **Next** to continue creating your resource.

Additional Information

For more information about tagging, see the following resources.

- AWS Tagging Principles in the *AWS General Reference*
- AWS Tagging Strategies (downloadable PDF)
- AWS Access Control in the *AWS IAM User Guide*
- AWS Tagging Policies in the *AWS Organizations User Guide*

Receiving Job Status Notifications

You can have Amazon EventBridge or Amazon CloudWatch Events notify you with status updates for ongoing Amazon Forecast resource jobs, such as creating predictors or forecasts. EventBridge and CloudWatch Events deliver a near real-time stream of system events that describe changes in Amazon Web Services (AWS) resources. For example, you can set up an event to notify you when a Forecast predictor finishes training.

Events are emitted on a best-effort basis. For more information about events, see the Amazon EventBridge User Guide or the Amazon CloudWatch Events User Guide.

> **Note**
>
> We recommend using Amazon EventBridge to manage events. CloudWatch Events and EventBridge use the same API and provide the same functionality, but EventBridge provides more features. Changes that you make in either CloudWatch or EventBridge will appear in each console. For more information, see Amazon EventBridge.

Topics

- Monitoring Forecast Resource Jobs
- Creating an EventBridge Rule for Job Status Notifications
- Creating a CloudWatch Events Rule for Job Status Notifications

Monitoring Forecast Resource Jobs

An event indicates a change in your AWS environment, and a rule matches incoming events and routes them to targets for processing. You can set up rules to match Forecast events and route them to one or more target functions or streams. EventBridge and CloudWatch Events detect events as they occur and invoke the target in the matching rule.

The following table lists the Forecast resource jobs and their status change events, which you can monitor.

Resource Job	Status Change Event Name	Status
CreateDatasetImportJob	Forecast Dataset Import Job State Change	ACTIVE, CREATE_IN _PROGRESS, CREATE_FAILED, CREATE_STOPPED
CreatePredictor	Forecast Predictor Creation State Change	ACTIVE, CREATE_IN _PROGRESS, CREATE_FAILED, CREATE_STOPPED
CreateForecast	Forecast Forecast Creation State Change	ACTIVE, CREATE_IN _PROGRESS, CREATE_FAILED, CREATE_STOPPED
CreateExplainability	Forecast Explainability Creation State Change	ACTIVE, CREATE_IN _PROGRESS, CREATE_FAILED, CREATE_STOPPED
CreatePredictorBacktestExportJob	Forecast Predictor Backtest Export Job State Change	ACTIVE, CREATE_IN _PROGRESS, CREATE_FAILED, CREATE_STOPPED
CreateForecastExportJob	Forecast Forecast Export Job State Change	ACTIVE, CREATE_IN _PROGRESS, CREATE_FAILED, CREATE_STOPPED
CreateExplainabilityExport	Forecast Explainability Export Creation State Change	ACTIVE, CREATE_IN _PROGRESS, CREATE_FAILED, CREATE_STOPPED

Resource Job	Status Change Event Name	Status
CreateWhatIfAnalysis	Forecast What-if Analysis Creation State Change	ACTIVE, CREATE_IN _PROGRESS, CREATE_FAILED, CREATE_STOPPED
CreateWhatIfForecast	Forecast What-if Forecast Creation State Change	ACTIVE, CREATE_IN _PROGRESS, CREATE_FAILED, CREATE_STOPPED
CreateWhatIfForecastExport	Forecast What-if Forecast Export Creation State Change	ACTIVE, CREATE_IN _PROGRESS, CREATE_FAILED, CREATE_STOPPED
DeleteDataset	Forecast Dataset Deletion State Change	DELETE_IN_PROGRESS, DELETE_FAILED
DeleteDatasetImportJob	Forecast Dataset Import Job Deletion State Change	DELETE_IN_PROGRESS, DELETE_FAILED
DeletePredictor	Forecast Predictor Deletion State Change	DELETE_IN_PROGRESS, DELETE_FAILED
DeleteForecast	Forecast Forecast Deletion State Change	DELETE_IN_PROGRESS, DELETE_FAILED
DeleteExplainability	Forecast Explainability Deletion State Change	DELETE_IN_PROGRESS, DELETE_FAILED
DeleteExplainabilityExport	Forecast Explainability Export Deletion State Change	DELETE_IN_PROGRESS, DELETE_FAILED
DeleteWhatIfAnalysis	Forecast What-if Analysis Deletion State Change	DELETE_IN_PROGRESS, DELETE_FAILED
DeleteWhatIfForecast	Forecast What-if Forecast Deletion State Change	DELETE_IN_PROGRESS, DELETE_FAILED

Resource Job	Status Change Event Name	Status
DeleteWhatIfForecastExportJob	Forecast What-if Forecast Export Deletion State Change	DELETE_IN_PROGRESS, DELETE_FAILED

Notifications contain information about the resource, including the Amazon Resource Name (ARN), job status, job duration (in minutes), and, if the job failed, an error message. Delete event notifications don't include a `Duration` field. The following is an example notification:

```
{
    "version": "0",
    "id": "017fcb6d-7ca3-ebf8-819e-3e0fa956ee17",
    "detail-type": "Forecast Dataset Import Job State Change",
    "source": "aws.forecast",
    "account": "000000000001",
    "time": "2021-02-19T05:45:51Z",
    "region": "us-east-1",
    "resources": [
        "arn:aws:forecast:us-west-2:000000000001:dataset/example_data"
    ],
    "detail": {
        "Arn": "arn:aws:forecast:us-west-2:000000000001:dataset/example_data",
        "Duration": 60,
        "Status": "ACTIVE",
    }
}
```

Creating an EventBridge Rule for Job Status Notifications

To create an EventBridge rule to notify you of status changes for ongoing Forecast resource jobs, see Creating a rule for an AWS service in the *Amazon EventBridge User Guide*. In the procedure, for **Service name**, choose **Amazon Forecast**. For **Event type**, choose the Forecast event to monitor. See Monitoring Forecast Resource Jobs for the list of Forecast events.

Creating a CloudWatch Events Rule for Job Status Notifications

To create an CloudWatch Events rule to notify you of status changes for ongoing Forecast resource jobs, see Creating a CloudWatch Events Rule That Triggers on an Event in the *Amazon CloudWatch*

User Guide. In the procedure, for **Service name**, choose **Amazon Forecast**. For **Event type**, choose the Forecast event to monitor. See Monitoring Forecast Resource Jobs for a list of Forecast events.

Guidelines and Quotas

The following sections contain information about Amazon Forecast guidelines and quotas.

Topics

Supported AWS Regions

For a list of AWS Regions that support Forecast, see AWS Regions and Endpoints in the *Amazon Web Services General Reference*.

Compliance

For more information about Forecast compliance programs, see AWS Compliance, AWS Compliance Programs, and AWS Services in Scope by Compliance Program.

Service Quotas

> **Note**
>
> To request an increase for adjustable quotas, use the Service Quotas console and follow the steps in the Requesting a quota increase section of the *Service Quotas User Guide*.

Forecast has the following service quotas.

Quotas Imposed by the CreateDatasetImportJob API

Resource	Default Quota	Adjustable
Maximum number of files in your Amazon S3 bucket	10,000	No

Resource	Default Quota	Adjustable
Maximum cumulative size of all files in your Amazon S3 bucket	30 GB	Yes
Maximum number of datasets in a dataset group	3 (1 for each type)	No
Maximum number of rows in a dataset	3 billion Note: the quota for the *ap-south-1* region is 1 billion.	Yes
Maximum number of columns in a target time series dataset (required columns + additional forecast dimensions)	13 (3 + 10)	No
Maximum number of columns in a related time series dataset (required columns + additional forecast dimensions + related features)	25 (2 + 10 + 13)	No
Maximum number of columns in an item metadata dataset	10	No
Maximum number of columns in any other dataset	36	No

Quotas Imposed by the CreatePredictor API

Resource	Default Quota	Adjustable
Maximum `number of backtest windows` (EvaluationParameters)	5	No
Maximum number of time series per predictor (number of items X number of unique values across forecast dimensions in the target time series dataset)	5,000,000 across all target time series items and dimensions. Note: the quota for the *ap-south-1* region is 1,000,000 . If you exceed 100,000 items, Forecast supports yearly, monthly, weekly, and daily frequencies instead of more granular frequencies (such as hourly).	Yes
Maximum forecast horizon	CNN-QR, DeepAR+, AutoML: The lesser of 500 data points or 1/3 of the target time series dataset length ETS, NPTS, Prophet, ARIMA: The lesser of 500 data points or the length of the target time series dataset minus one.	No

General Resource Quotas

Resource	Default Quota	Adjustable
Maximum parallel running `CreateDatasetImportJob` tasks	3	Yes
Maximum parallel running `CreatePredictor` tasks	3	Yes
Maximum parallel running `CreatePredictor` tasks using AutoML	3	Yes
Maximum parallel running `CreateAutoPredictor` tasks	3	No
Maximum parallel running CreateExplainability tasks	3	No
Maximum parallel running CreateExplainabilityExport tasks	3	No
Maximum parallel running `CreatePredictorBacktestExportJob` tasks	3	Yes
Maximum parallel running `CreateForecast` tasks	3	Yes
Maximum parallel running `CreateForecastExportJob` tasks	3	Yes

Resource	Default Quota	Adjustable
Maximum parallel running `StopResource` tasks per resource type	3	Yes
Maximum number of datasets	1500	Yes
Maximum number of dataset groups	500	Yes
Maximum number of dataset import jobs	1000	Yes
Maximum number of predictors	500	Yes
Maximum number of AutoPredictors	500	No
Maximum number of predictor backtest export jobs	1000	Yes
Maximum number of forecasts	100	Yes
Maximum number of forecast export jobs	1000	Yes
Maximum time for which a forecast can be queried on console or QueryForecast API	30 days	No
Maximum number of tags you can add to a resource	50	No

Resource	Default Quota	Adjustable
Maximum parallel running `QueryForecast` API tasks	10 forecasts, including 5 created with large datasets (anything over 20GB or 100,000 items). If you have more than 5 forecasts created with large datasets, `QueryForecast` can access only the 5 most recent large dataset forecasts.	No
Maximum number of Explainabilities	1000	No
Maximum number of Explainability Export jobs	1000	No

What-if Analysis Quotas

Resource	Default Quota	Adjustable
Maximum parallel running `CreateWhatIfAnalysis` tasks	3	Yes
Maximum number of what-if analyses	500	Yes
Maximum parallel running `CreateWhatIfForecast` tasks	3	Yes
Maximum number of what-if forecasts	100	Yes

Resource	Default Quota	Adjustable
Maximum parallel running `CreateWhatIfForecastExport` tasks	3	Yes
Maximum number of what-if forecast exports	1000	Yes
Maximum number of what-if forecasts in an export job	3	No

Conditions and Restrictions

The following conditions and restrictions apply when using the Weather Index:

- **Available algorithms**: If using a legacy predictor, the Weather Index can be enabled when you train a predictor with the CNN-QR, DeepAR+, and Prophet algorithms. The Weather Index is not applied to ARIMA, ETS, and NPTS.
- **Forecast frequency**: The valid forecast frequencies are `Minutely`, `Hourly`, and `Daily`.
- **Forecast horizon**: The forecast horizon cannot span further than 14 days into the future. For forecast horizon limits for each forecast frequency, refer to the list below:
  - `1 minute` - 500
  - `5 minutes` - 500
  - `10 minutes` - 500
  - `15 minutes` - 500
  - `Hourly` - 330
  - `Daily` - 14
- **Time series length**: When training a model with the Weather Index, Forecast truncates all time series datasets with timestamps before the start date of the Forecast weather dataset featurization. The Forecast weather dataset featurization contains the following start dates:
  - **US region**: July 2, 2018
  - **Europe region**: July 2, 2018
  - **Asia Pacific region**: July 2, 2018

- **Canada region**: July 2, 2019
- **South America region**: January 2, 2020
- **Central America region**: September 2, 2020
- **Africa & Middle East region**: March 25, 2021

With the Weather Index enabled, data points with timestamps before the start date will not be used during predictor training.

- **Number of locations**: The target time series dataset cannot exceed 2000 unique locations.
- **Region bounds**: All items in your datasets must be located within a single region.
- **Minimum time series length**: Due to additional data requirements when testing the Weather Index, the minimum length for a time series dataset is:

  ```
  3 × ForecastHorizon + (BacktestWindows + 1) × BacktestWindowOffset
  ```

  If your time series datasets do not meet this requirement, consider decreasing the following:

  - `ForecastHorizon` - Shorten your forecast horizon.
  - `BacktestWindowOffset` - Shorten the length of the testing set during backtesting.
  - `BacktestWindows` - Reduce the number of backtests.

Reserved Field Names

Amazon Forecast reserves the following names. You can't use these names for your schema fields or dataset headers.

A

- A
- ABORT
- ABS
- ABSOLUTE
- ACCESS
- ACTION
- ADA
- ADD
- ADMIN
- AFTER
- AGGREGATE
- ALIAS
- ALL
- ALLOCATE
- ALSO
- ALTER
- ALWAYS
- ANALYSE
- ANALYZE
- AND
- ANY
- ARE
- ARRAY
- AS

- ASC
- ASENSITIVE
- ASSERTION
- ASSIGNMENT
- ASYMMETRIC
- AT
- ATOMIC
- ATTRIBUTE
- ATTRIBUTES
- AUDIT
- AUTHORIZATION
- AUTO_INCREMENT
- AVG
- AVG_ROW_LENGTH

B

- BACKUP
- BACKWARD
- BEFORE
- BEGIN
- BERNOULLI
- BETWEEN
- BIGINT
- BINARY
- BIT
- BIT_LENGTH
- BITVAR
- BLOB
- BOOL
- BOOLEAN

- BOTH
- BREADTH
- BREAK
- BROWSE
- BULK
- BY

C

- C
- CACHE
- CALL
- CALLED
- CARDINALITY
- CASCADE
- CASCADED
- CASE
- CAST
- CATALOG
- CATALOG_NAME
- CEIL
- CEILING
- CHAIN
- CHANGE
- CHAR
- CHAR_LENGTH
- CHARACTER
- CHARACTER_LENGTH
- CHARACTER_SET_CATALOG
- CHARACTER_SET_NAME
- CHARACTER_SET_SCHEMA

- CHARACTERISTICS
- CHARACTERS
- CHECK
- CHECKED
- CHECKPOINT
- CHECKSUM
- CLASS
- CLASS_ORIGIN
- CLOB
- CLOSE
- CLUSTER
- CLUSTERED
- COALESCE
- COBOL
- COLLATE
- COLLATION
- COLLATION_CATALOG
- COLLATION_NAME
- COLLATION_SCHEMA
- COLLECT
- COLUMN
- COLUMN_NAME
- COLUMNS
- COMMAND_FUNCTION
- COMMAND_FUNCTION_CODE
- COMMENT
- COMMIT
- COMMITTED
- COMPLETION
- COMPRESS

- COMPUTE
- CONDITION
- CONDITION_NUMBER
- CONNECT
- CONNECTION
- CONNECTION_NAME
- CONSTRAINT
- CONSTRAINT_CATALOG
- CONSTRAINT_NAME
- CONSTRAINT_SCHEMA
- CONSTRAINTS
- CONSTRUCTOR
- CONTAINS
- CONTAINSTABLE
- CONTINUE
- CONVERSION
- CONVERT
- COPY
- CORR
- CORRESPONDING
- COUNT
- COVAR_POP
- COVAR_SAMP
- CREATE
- CREATEDB
- CREATEROLE
- CREATEUSER
- CROSS
- CSV
- CUBE

- CUME_DIST
- CURRENT
- CURRENT_DATE
- CURRENT_DEFAULT_TRANSFORM_GROUP
- CURRENT_PATH
- CURRENT_ROLE
- CURRENT_TIME
- CURRENT_TIMESTAMP
- CURRENT_TRANSFORM_GROUP_FOR_TYPE
- CURRENT_USER
- CURSOR
- CURSOR_NAME
- CYCLE

D

- DATA
- DATABASE
- DATABASES
- DATETIME
- DATETIME_INTERVAL_CODE
- DATETIME_INTERVAL_PRECISION
- DAY
- DAY_HOUR
- DAY_MICROSECOND
- DAY_MINUTE
- DAY_SECOND
- DAYOFMONTH
- DAYOFWEEK
- DAYOFYEAR
- DBCC

- DEALLOCATE
- DEC
- DECIMAL
- DECLARE
- DEFAULT
- DEFAULTS
- DEFERRABLE
- DEFERRED
- DEFINED
- DEFINER
- DEGREE
- DELAY_KEY_WRITE
- DELAYED
- DELETE
- DELIMITER
- DELIMITERS
- DENSE_RANK
- DENY
- DEPTH
- DEREF
- DERIVED
- DESC
- DESCRIBE
- DESCRIPTOR
- DESTROY
- DESTRUCTOR
- DETERMINISTIC
- DIAGNOSTICS
- DICTIONARY
- DISABLE

- DISCONNECT
- DISK
- DISPATCH
- DISTINCT
- DISTINCTROW
- DISTRIBUTED
- DIV
- DO
- DOMAIN
- DOUBLE
- DROP
- DUAL
- DUMMY
- DUMP
- DYNAMIC
- DYNAMIC_FUNCTION
- DYNAMIC_FUNCTION_CODE

E

- EACH
- ELEMENT
- ELSE
- ELSEIF
- ENABLE
- ENCLOSED
- ENCODING
- ENCRYPTED
- END
- END-EXEC
- ENUM

- EQUALS
- ERRLVL
- ESCAPE
- ESCAPED
- EVERY
- EXCEPT
- EXCEPTION
- EXCLUDE
- EXCLUDING
- EXCLUSIVE
- EXEC
- EXECUTE
- EXISTING
- EXISTS
- EXIT
- EXP
- EXPLAIN
- EXTERNAL
- EXTRACT

F

- FALSE
- FETCH
- FIELDS
- FILE
- FILLFACTOR
- FILTER
- FINAL
- FIRST
- FLOAT

- FLOAT4
- FLOAT8
- FLOOR
- FLUSH
- FOLLOWING
- FOR
- FORCE
- FOREIGN
- FORTRAN
- FORWARD
- FOUND
- FREE
- FREETEXT
- FREETEXTTABLE
- FREEZE
- FROM
- FULL
- FULLTEXT
- FUNCTION
- FUSION

G

- G
- GENERAL
- GENERATED
- GET
- GLOBAL
- GO
- GOTO
- GRANT

- GRANTED
- GRANTS
- GREATEST
- GROUP
- GROUPING

H

- HANDLER
- HAVING
- HEADER
- HEAP
- HIERARCHY
- HIGH_PRIORITY
- HOLD
- HOLDLOCK
- HOST
- HOSTS
- HOUR
- HOUR_MICROSECOND
- HOUR_MINUTE
- HOUR_SECOND

I

- IDENTIFIED
- IDENTITY
- IDENTITY_INSERT
- IDENTITYCOL
- IF
- IGNORE
- ILIKE

- IMMEDIATE
- IMMUTABLE
- IMPLEMENTATION
- IMPLICIT
- IN
- INCLUDE
- INCLUDING
- INCREMENT
- INDEX
- INDICATOR
- INFILE
- INFIX
- INHERIT
- INHERITS
- INITIAL
- INITIALIZE
- INITIALLY
- INNER
- INOUT
- INPUT
- INSENSITIVE
- INSERT
- INSERT_ID
- INSTANCE
- INSTANTIABLE
- INSTEAD
- INT
- INT1
- INT2
- INT3

- INT4
- INT8
- INTEGER
- INTERSECT
- INTERSECTION
- INTERVAL
- INTO
- INVOKER
- IS
- ISAM
- ISNULL
- ISOLATION
- ITERATE

J

- JOIN

K

- K
- KEY
- KEY_MEMBER
- KEY_TYPE
- KEYS
- KILL

L

- LANCOMPILER
- LANGUAGE
- LARGE

- LAST
- LAST_INSERT_ID
- LATERAL
- LEADING
- LEAST
- LEAVE
- LEFT
- LENGTH
- LESS
- LEVEL
- LIKE
- LIMIT
- LINENO
- LINES
- LISTEN
- LN
- LOAD
- LOCAL
- LOCALTIME
- LOCALTIMESTAMP
- LOCATOR
- LOCK
- LOGIN
- LOGS
- LONG
- LONGBLOB
- LONGTEXT
- LOOP
- LOW_PRIORITY
- LOWER

M

- M
- MAP
- MATCH
- MATCHED
- MAX
- MAX_ROWS
- MAXEXTENTS
- MAXVALUE
- MEAN
- MEDIUMBLOB
- MEDIUMINT
- MEDIUMTEXT
- MEMBER
- MERGE
- MESSAGE_LENGTH
- MESSAGE_OCTET_LENGTH
- MESSAGE_TEXT
- METHOD
- MIDDLEINT
- MIN
- MIN_ROWS
- MINUS
- MINUTE
- MINUTE_MICROSECOND
- MINUTE_SECOND
- MINVALUE
- MLSLABEL
- MOD
- MODE

- MODIFIES
- MODIFY
- MODULE
- MONTH
- MONTHNAME
- MORE
- MOVE
- MULTISET
- MUMPS
- MYISAM

N

- NAME
- NAMES
- NATIONAL
- NATURAL
- NCHAR
- NCLOB
- NESTING
- NEW
- NEXT
- NO
- NO_WRITE_TO_BINLOG
- NOAUDIT
- NOCHECK
- NOCOMPRESS
- NOCREATEDB
- NOCREATEROLE
- NOCREATEUSER
- NOINHERIT

- NOLOGIN
- NONCLUSTERED
- NONE
- NORMALIZE
- NORMALIZED
- NOSUPERUSER
- NOT
- NOTHING
- NOTIFY
- NOTNULL
- NOWAIT
- NULL
- NULLABLE
- NULLIF
- NULLS
- NUMBER
- NUMERIC

O

- OBJECT
- OCTET_LENGTH
- OCTETS
- OF
- OFF
- OFFLINE
- OFFSET
- OFFSETS
- OIDS
- OLD
- ON

- ONLINE
- ONLY
- OPEN
- OPENDATASOURCE
- OPENQUERY
- OPENROWSET
- OPENXML
- OPERATION
- OPERATOR
- OPTIMIZE
- OPTION
- OPTIONALLY
- OPTIONS
- OR
- ORDER
- ORDERING
- ORDINALITY
- OTHERS
- OUT
- OUTER
- OUTFILE
- OUTPUT
- OVER
- OVERLAPS
- OVERLAY
- OVERRIDING
- OWNER

P

- PACK_KEYS

- PAD
- PARAMETER
- PARAMETER_MODE
- PARAMETER_NAME
- PARAMETER_ORDINAL_POSITION
- PARAMETER_SPECIFIC_CATALOG
- PARAMETER_SPECIFIC_NAME
- PARAMETER_SPECIFIC_SCHEMA
- PARAMETERS
- PARTIAL
- PARTITION
- PASCAL
- PASSWORD
- PATH
- PCTFREE
- PERCENT
- PERCENT_RANK
- PERCENTILE_CONT
- PERCENTILE_DISC
- PLACING
- PLAN
- PLI
- POSITION
- POSTFIX
- POWER
- PRECEDING
- PRECISION
- PREFIX
- PREORDER
- PREPARE

- PREPARED
- PRESERVE
- PRIMARY
- PRINT
- PRIOR
- PRIVILEGES
- PROC
- PROCEDURAL
- PROCEDURE
- PROCESS
- PROCESSLIST
- PUBLIC
- PURGE

Q

- QUOTE

R

- RAID0
- RAISERROR
- RANGE
- RANK
- RAW
- READ
- READS
- READTEXT
- REAL
- RECHECK
- RECONFIGURE
- RECURSIVE

- REF
- REFERENCES
- REFERENCING
- REGEXP
- REGR_AVGX
- REGR_AVGY
- REGR_COUNT
- REGR_INTERCEPT
- REGR_R2
- REGR_SLOPE
- REGR_SXX
- REGR_SXY
- REGR_SYY
- REINDEX
- RELATIVE
- RELEASE
- RELOAD
- RENAME
- REPEAT
- REPEATABLE
- REPLACE
- REPLICATION
- REQUIRE
- RESET
- RESIGNAL
- RESOURCE
- RESTART
- RESTORE
- RESTRICT
- RESULT

- RETURN
- RETURNED_CARDINALITY
- RETURNED_LENGTH
- RETURNED_OCTET_LENGTH
- RETURNED_SQLSTATE
- RETURNS
- REVOKE
- RIGHT
- RLIKE
- ROLE
- ROLLBACK
- ROLLUP
- ROUTINE
- ROUTINE_CATALOG
- ROUTINE_NAME
- ROUTINE_SCHEMA
- ROW
- ROW_COUNT
- ROW_NUMBER
- ROWCOUNT
- ROWGUIDCOL
- ROWID
- ROWNUM
- ROWS
- RULE

S

- SAVE
- SAVEPOINT
- SCALE

- SCHEMA
- SCHEMA_NAME
- SCHEMAS
- SCOPE
- SCOPE_CATALOG
- SCOPE_NAME
- SCOPE_SCHEMA
- SCROLL
- SEARCH
- SECOND
- SECOND_MICROSECOND
- SECTION
- SECURITY
- SELECT
- SELF
- SENSITIVE
- SEPARATOR
- SEQUENCE
- SERIALIZABLE
- SERVER_NAME
- SESSION
- SESSION_USER
- SET
- SETOF
- SETS
- SETUSER
- SHARE
- SHOW
- SHUTDOWN
- SIGNAL

- SIMILAR
- SIMPLE
- SIZE
- SMALLINT
- SOME
- SONAME
- SOURCE
- SPACE
- SPATIAL
- SPECIFIC
- SPECIFIC_NAME
- SPECIFICTYPE
- SQL
- SQL_BIG_RESULT
- SQL_BIG_SELECTS
- SQL_BIG_TABLES
- SQL_CALC_FOUND_ROWS
- SQL_LOG_OFF
- SQL_LOG_UPDATE
- SQL_LOW_PRIORITY_UPDATES
- SQL_SELECT_LIMIT
- SQL_SMALL_RESULT
- SQL_WARNINGS
- SQLCA
- SQLCODE
- SQLERROR
- SQLEXCEPTION
- SQLSTATE
- SQLWARNING
- SQRT

- SSL
- STABLE
- START
- STARTING
- STATE
- STATEMENT
- STATIC
- STATISTICS
- STATUS
- STDDEV_POP
- STDDEV_SAMP
- STDIN
- STDOUT
- STORAGE
- STRAIGHT_JOIN
- STRICT
- STRING
- STRUCTURE
- STYLE
- SUBCLASS_ORIGIN
- SUBLIST
- SUBMULTISET
- SUBSTRING
- SUCCESSFUL
- SUM
- SUPERUSER
- SYMMETRIC
- SYNONYM
- SYSDATE
- SYSID

- SYSTEM
- SYSTEM_USER

T

- TABLE
- TABLE_NAME
- TABLES
- TABLESAMPLE
- TABLESPACE
- TEMP
- TEMPLATE
- TEMPORARY
- TERMINATE
- TERMINATED
- TEXT
- TEXTSIZE
- THAN
- THEN
- TIES
- TIME
- TIMEZONE_HOUR
- TIMEZONE_MINUTE
- TINYBLOB
- TINYINT
- TINYTEXT
- TO
- TOAST
- TOP
- TOP_LEVEL_COUNT
- TRAILING

- TRAN
- TRANSACTION
- TRANSACTION_ACTIVE
- TRANSACTIONS_COMMITTED
- TRANSACTIONS_ROLLED_BACK
- TRANSFORM
- TRANSFORMS
- TRANSLATE
- TRANSLATION
- TREAT
- TRIGGER
- TRIGGER_CATALOG
- TRIGGER_NAME
- TRIGGER_SCHEMA
- TRIM
- TRUE
- TRUNCATE
- TRUSTED
- TSEQUAL
- TYPE

U

- UESCAPE
- UID
- UNBOUNDED
- UNCOMMITTED
- UNDER
- UNDO
- UNENCRYPTED
- UNION

- UNIQUE
- UNKNOWN
- UNLISTEN
- UNLOCK
- UNNAMED
- UNNEST
- UNSIGNED
- UNTIL
- UPDATE
- UPDATETEXT
- UPPER
- USAGE
- USE
- USER
- USER_DEFINED_TYPE_CATALOG
- USER_DEFINED_TYPE_CODE
- USER_DEFINED_TYPE_NAME
- USER_DEFINED_TYPE_SCHEMA
- USING
- UTC_DATE
- UTC_TIME
- UTC_TIMESTAMP

V

- VACUUM
- VALID
- VALIDATE
- VALIDATOR
- VALUE
- VALUES

- VAR_POP
- VAR_SAMP
- VARBINARY
- VARCHAR
- VARCHAR2
- VARCHARACTER
- VARIABLE
- VARIABLES
- VARYING
- VERBOSE
- VIEW
- VOLATILE

W

- WAITFOR
- WHEN
- WHENEVER
- WHERE
- WHILE
- WIDTH_BUCKET
- WINDOW
- WITH
- WITHIN
- WITHOUT
- WORK
- WRITE
- WRITETEXT

X

- X509

- XOR

Y

- YEAR
- YEAR_MONTH

Z

- ZEROFILL
- ZONE

Code examples for Forecast using AWS SDKs

The following code examples show how to use Forecast with an AWS software development kit (SDK).

Actions are code excerpts from larger programs and must be run in context. While actions show you how to call individual service functions, you can see actions in context in their related scenarios and cross-service examples.

For a complete list of AWS SDK developer guides and code examples, see Using Forecast with an AWS SDK. This topic also includes information about getting started and details about previous SDK versions.

Code examples

- Actions for Forecast using AWS SDKs
  - Use CreateDataset with an AWS SDK or CLI
  - Use CreateForecast with an AWS SDK or CLI
  - Use DeleteDataset with an AWS SDK or CLI
  - Use DeleteForecast with an AWS SDK or CLI
  - Use DescribeForecast with an AWS SDK or CLI
  - Use ListDatasetGroups with an AWS SDK or CLI
  - Use ListForecasts with an AWS SDK or CLI

Actions for Forecast using AWS SDKs

The following code examples demonstrate how to perform individual Forecast actions with AWS SDKs. These excerpts call the Forecast API and are code excerpts from larger programs that must be run in context. Each example includes a link to GitHub, where you can find instructions for setting up and running the code.

The following examples include only the most commonly used actions. For a complete list, see the Amazon Forecast API Reference.

Examples

- Use CreateDataset with an AWS SDK or CLI
- Use CreateForecast with an AWS SDK or CLI

- Use DeleteDataset with an AWS SDK or CLI
- Use DeleteForecast with an AWS SDK or CLI
- Use DescribeForecast with an AWS SDK or CLI
- Use ListDatasetGroups with an AWS SDK or CLI
- Use ListForecasts with an AWS SDK or CLI

Use `CreateDataset` with an AWS SDK or CLI

The following code example shows how to use `CreateDataset`.

Java

SDK for Java 2.x

> **Note**
>
> There's more on GitHub. Find the complete example and learn how to set up and run in the AWS Code Examples Repository.

```
import software.amazon.awssdk.regions.Region;
import software.amazon.awssdk.services.forecast.ForecastClient;
import software.amazon.awssdk.services.forecast.model.CreateDatasetRequest;
import software.amazon.awssdk.services.forecast.model.Schema;
import software.amazon.awssdk.services.forecast.model.SchemaAttribute;
import software.amazon.awssdk.services.forecast.model.CreateDatasetResponse;
import software.amazon.awssdk.services.forecast.model.ForecastException;
import java.util.ArrayList;
import java.util.List;

/**
 * Before running this Java V2 code example, set up your development
 * environment, including your credentials.
 *
 * For more information, see the following documentation topic:
 *
 * https://docs.aws.amazon.com/sdk-for-java/latest/developer-guide/get-
started.html
 */
public class CreateDataSet {
```

```
    public static void main(String[] args) {
        final String usage = """

                Usage:
                    <name>\s

                Where:
                    name - The name of the data set.\s
                """;

        if (args.length != 1) {
            System.out.println(usage);
            System.exit(1);
        }

        String name = args[0];
        Region region = Region.US_WEST_2;
        ForecastClient forecast = ForecastClient.builder()
                .region(region)
                .build();

        String myDataSetARN = createForecastDataSet(forecast, name);
        System.out.println("The ARN of the new data set is " + myDataSetARN);
        forecast.close();
    }

    public static String createForecastDataSet(ForecastClient forecast, String
name) {
        try {
            Schema schema = Schema.builder()
                    .attributes(getSchema())
                    .build();

            CreateDatasetRequest datasetRequest = CreateDatasetRequest.builder()
                    .datasetName(name)
                    .domain("CUSTOM")
                    .datasetType("RELATED_TIME_SERIES")
                    .dataFrequency("D")
                    .schema(schema)
                    .build();

            CreateDatasetResponse response =
forecast.createDataset(datasetRequest);
            return response.datasetArn();
```

```
        } catch (ForecastException e) {
            System.err.println(e.awsErrorDetails().errorMessage());
            System.exit(1);
        }

        return "";
    }

    // Create a SchemaAttribute list required to create a data set.
    private static List<SchemaAttribute> getSchema() {

        List<SchemaAttribute> schemaList = new ArrayList<>();
        SchemaAttribute att1 = SchemaAttribute.builder()
                .attributeName("item_id")
                .attributeType("string")
                .build();

        SchemaAttribute att2 = SchemaAttribute.builder()
                .attributeName("timestamp")
                .attributeType("timestamp")
                .build();

        SchemaAttribute att3 = SchemaAttribute.builder()
                .attributeName("target_value")
                .attributeType("float")
                .build();

        // Push the SchemaAttribute objects to the List.
        schemaList.add(att1);
        schemaList.add(att2);
        schemaList.add(att3);
        return schemaList;
    }
}
```

- For API details, see CreateDataset in *AWS SDK for Java 2.x API Reference*.

For a complete list of AWS SDK developer guides and code examples, see Using Forecast with an AWS SDK. This topic also includes information about getting started and details about previous SDK versions.

Use `CreateForecast` with an AWS SDK or CLI

The following code example shows how to use `CreateForecast`.

Java

SDK for Java 2.x

> **Note**
>
> There's more on GitHub. Find the complete example and learn how to set up and run in the AWS Code Examples Repository.

```
import software.amazon.awssdk.regions.Region;
import software.amazon.awssdk.services.forecast.ForecastClient;
import software.amazon.awssdk.services.forecast.model.CreateForecastRequest;
import software.amazon.awssdk.services.forecast.model.CreateForecastResponse;
import software.amazon.awssdk.services.forecast.model.ForecastException;

/**
 * Before running this Java V2 code example, set up your development
 * environment, including your credentials.
 *
 * For more information, see the following documentation topic:
 *
 * https://docs.aws.amazon.com/sdk-for-java/latest/developer-guide/get-
started.html
 */
public class CreateForecast {
    public static void main(String[] args) {
        final String usage = """

                Usage:
                    <name> <predictorArn>\s

                Where:
                    name - The name of the forecast.\s
                    predictorArn - The arn of the predictor to use.\s

                """;
```

```
        if (args.length != 2) {
            System.out.println(usage);
            System.exit(1);
        }

        String name = args[0];
        String predictorArn = args[1];
        Region region = Region.US_WEST_2;
        ForecastClient forecast = ForecastClient.builder()
                .region(region)
                .build();

        String forecastArn = createNewForecast(forecast, name, predictorArn);
        System.out.println("The ARN of the new forecast is " + forecastArn);
        forecast.close();
    }

    public static String createNewForecast(ForecastClient forecast, String name,
 String predictorArn) {
        try {
            CreateForecastRequest forecastRequest =
 CreateForecastRequest.builder()
                    .forecastName(name)
                    .predictorArn(predictorArn)
                    .build();

            CreateForecastResponse response =
 forecast.createForecast(forecastRequest);
            return response.forecastArn();

        } catch (ForecastException e) {
            System.err.println(e.awsErrorDetails().errorMessage());
            System.exit(1);
        }
        return "";
    }
}
```

- For API details, see CreateForecast in *AWS SDK for Java 2.x API Reference*.

For a complete list of AWS SDK developer guides and code examples, see Using Forecast with an AWS SDK. This topic also includes information about getting started and details about previous SDK versions.

Use `DeleteDataset` with an AWS SDK or CLI

The following code example shows how to use `DeleteDataset`.

Java

SDK for Java 2.x

> **Note**
>
> There's more on GitHub. Find the complete example and learn how to set up and run in the AWS Code Examples Repository.

```
import software.amazon.awssdk.regions.Region;
import software.amazon.awssdk.services.forecast.ForecastClient;
import software.amazon.awssdk.services.forecast.model.DeleteDatasetRequest;
import software.amazon.awssdk.services.forecast.model.ForecastException;

/**
 * Before running this Java V2 code example, set up your development
 * environment, including your credentials.
 *
 * For more information, see the following documentation topic:
 *
 * https://docs.aws.amazon.com/sdk-for-java/latest/developer-guide/get-
started.html
 */
public class DeleteDataset {

    public static void main(String[] args) {
        final String usage = """

                Usage:
                    <datasetARN>\s

                Where:
                    datasetARN - The ARN of the data set to delete.\s
```

```
                """;

        if (args.length != 1) {
            System.out.println(usage);
            System.exit(1);
        }

        String datasetARN = args[0];
        Region region = Region.US_WEST_2;
        ForecastClient forecast = ForecastClient.builder()
                .region(region)
                .build();

        deleteForecastDataSet(forecast, datasetARN);
        forecast.close();
    }

    public static void deleteForecastDataSet(ForecastClient forecast, String
 myDataSetARN) {
        try {
            DeleteDatasetRequest deleteRequest = DeleteDatasetRequest.builder()
                    .datasetArn(myDataSetARN)
                    .build();

            forecast.deleteDataset(deleteRequest);
            System.out.println("The Data Set was deleted");

        } catch (ForecastException e) {
            System.err.println(e.awsErrorDetails().errorMessage());
            System.exit(1);
        }
    }
}
```

- For API details, see DeleteDataset in *AWS SDK for Java 2.x API Reference*.

For a complete list of AWS SDK developer guides and code examples, see Using Forecast with an AWS SDK. This topic also includes information about getting started and details about previous SDK versions.

Use `DeleteForecast` with an AWS SDK or CLI

The following code example shows how to use `DeleteForecast`.

Java

SDK for Java 2.x

> **Note**
>
> There's more on GitHub. Find the complete example and learn how to set up and run in the AWS Code Examples Repository.

```
import software.amazon.awssdk.regions.Region;
import software.amazon.awssdk.services.forecast.ForecastClient;
import software.amazon.awssdk.services.forecast.model.DeleteDatasetRequest;
import software.amazon.awssdk.services.forecast.model.ForecastException;

/**
 * Before running this Java V2 code example, set up your development
 * environment, including your credentials.
 *
 * For more information, see the following documentation topic:
 *
 * https://docs.aws.amazon.com/sdk-for-java/latest/developer-guide/get-
started.html
 */
public class DeleteDataset {

    public static void main(String[] args) {
        final String usage = """

                Usage:
                    <datasetARN>\s

                Where:
                    datasetARN - The ARN of the data set to delete.\s
                """;

        if (args.length != 1) {
            System.out.println(usage);
```

```
            System.exit(1);
        }

        String datasetARN = args[0];
        Region region = Region.US_WEST_2;
        ForecastClient forecast = ForecastClient.builder()
                .region(region)
                .build();

        deleteForecastDataSet(forecast, datasetARN);
        forecast.close();
    }

    public static void deleteForecastDataSet(ForecastClient forecast, String
 myDataSetARN) {
        try {
            DeleteDatasetRequest deleteRequest = DeleteDatasetRequest.builder()
                    .datasetArn(myDataSetARN)
                    .build();

            forecast.deleteDataset(deleteRequest);
            System.out.println("The Data Set was deleted");

        } catch (ForecastException e) {
            System.err.println(e.awsErrorDetails().errorMessage());
            System.exit(1);
        }
    }
}
```

- For API details, see DeleteForecast in *AWS SDK for Java 2.x API Reference*.

For a complete list of AWS SDK developer guides and code examples, see Using Forecast with an AWS SDK. This topic also includes information about getting started and details about previous SDK versions.

Use `DescribeForecast` with an AWS SDK or CLI

The following code example shows how to use `DescribeForecast`.

Java

SDK for Java 2.x

> **Note**
>
> There's more on GitHub. Find the complete example and learn how to set up and run in the AWS Code Examples Repository.

```
import software.amazon.awssdk.regions.Region;
import software.amazon.awssdk.services.forecast.ForecastClient;
import software.amazon.awssdk.services.forecast.model.DescribeForecastRequest;
import software.amazon.awssdk.services.forecast.model.DescribeForecastResponse;
import software.amazon.awssdk.services.forecast.model.ForecastException;

/**
 * Before running this Java V2 code example, set up your development
 * environment, including your credentials.
 *
 * For more information, see the following documentation topic:
 *
 * https://docs.aws.amazon.com/sdk-for-java/latest/developer-guide/get-
started.html
 */
public class DescribeForecast {
    public static void main(String[] args) {
        final String usage = """

                Usage:
                    <forecastarn>\s

                Where:
                    forecastarn - The arn of the forecast (for example,
 "arn:aws:forecast:us-west-2:xxxxx322:forecast/my_forecast)
                """;

        if (args.length != 1) {
            System.out.println(usage);
            System.exit(1);
        }
```

```
        String forecastarn = args[0];
        Region region = Region.US_WEST_2;
        ForecastClient forecast = ForecastClient.builder()
                .region(region)
                .build();

        describe(forecast, forecastarn);
        forecast.close();
    }

    public static void describe(ForecastClient forecast, String forecastarn) {
        try {
            DescribeForecastRequest request = DescribeForecastRequest.builder()
                    .forecastArn(forecastarn)
                    .build();

            DescribeForecastResponse response =
 forecast.describeForecast(request);
            System.out.println("The name of the forecast is " +
 response.forecastName());

        } catch (ForecastException e) {
            System.err.println(e.awsErrorDetails().errorMessage());
            System.exit(1);
        }
    }
}
```

- For API details, see DescribeForecast in *AWS SDK for Java 2.x API Reference*.

For a complete list of AWS SDK developer guides and code examples, see Using Forecast with an AWS SDK. This topic also includes information about getting started and details about previous SDK versions.

Use `ListDatasetGroups` with an AWS SDK or CLI

The following code example shows how to use `ListDatasetGroups`.

Java

SDK for Java 2.x

> **Note**
>
> There's more on GitHub. Find the complete example and learn how to set up and run in the AWS Code Examples Repository.

```
import software.amazon.awssdk.regions.Region;
import software.amazon.awssdk.services.forecast.ForecastClient;
import software.amazon.awssdk.services.forecast.model.DatasetGroupSummary;
import software.amazon.awssdk.services.forecast.model.ListDatasetGroupsRequest;
import software.amazon.awssdk.services.forecast.model.ListDatasetGroupsResponse;
import software.amazon.awssdk.services.forecast.model.ForecastException;
import java.util.List;

/**
 * Before running this Java V2 code example, set up your development
 * environment, including your credentials.
 *
 * For more information, see the following documentation topic:
 *
 * https://docs.aws.amazon.com/sdk-for-java/latest/developer-guide/get-
started.html
 */
public class ListDataSetGroups {
    public static void main(String[] args) {
        Region region = Region.US_WEST_2;
        ForecastClient forecast = ForecastClient.builder()
                .region(region)
                .build();

        listDataGroups(forecast);
        forecast.close();
    }

    public static void listDataGroups(ForecastClient forecast) {
        try {
            ListDatasetGroupsRequest group = ListDatasetGroupsRequest.builder()
                    .maxResults(10)
```

```
                    .build();

            ListDatasetGroupsResponse response =
 forecast.listDatasetGroups(group);
            List<DatasetGroupSummary> groups = response.datasetGroups();
            for (DatasetGroupSummary myGroup : groups) {
                System.out.println("The Data Set name is " +
 myGroup.datasetGroupName());
            }

        } catch (ForecastException e) {
            System.err.println(e.awsErrorDetails().errorMessage());
            System.exit(1);
        }
    }
}
```

- For API details, see ListDatasetGroups in *AWS SDK for Java 2.x API Reference*.

For a complete list of AWS SDK developer guides and code examples, see Using Forecast with an AWS SDK. This topic also includes information about getting started and details about previous SDK versions.

Use `ListForecasts` with an AWS SDK or CLI

The following code example shows how to use `ListForecasts`.

Java

SDK for Java 2.x

> **Note**
>
> There's more on GitHub. Find the complete example and learn how to set up and run in the AWS Code Examples Repository.

```
import software.amazon.awssdk.regions.Region;
import software.amazon.awssdk.services.forecast.ForecastClient;
import software.amazon.awssdk.services.forecast.model.ListForecastsResponse;
```

```
import software.amazon.awssdk.services.forecast.model.ListForecastsRequest;
import software.amazon.awssdk.services.forecast.model.ForecastSummary;
import software.amazon.awssdk.services.forecast.model.ForecastException;
import java.util.List;

/**
 * Before running this Java V2 code example, set up your development
 * environment, including your credentials.
 *
 * For more information, see the following documentation topic:
 *
 * https://docs.aws.amazon.com/sdk-for-java/latest/developer-guide/get-
started.html
 */
public class ListForecasts {

    public static void main(String[] args) {
        Region region = Region.US_WEST_2;
        ForecastClient forecast = ForecastClient.builder()
                .region(region)
                .build();

        listAllForeCasts(forecast);
        forecast.close();
    }

    public static void listAllForeCasts(ForecastClient forecast) {
        try {
            ListForecastsRequest request = ListForecastsRequest.builder()
                    .maxResults(10)
                    .build();

            ListForecastsResponse response = forecast.listForecasts(request);
            List<ForecastSummary> forecasts = response.forecasts();
            for (ForecastSummary forecastSummary : forecasts) {
                System.out.println("The name of the forecast is " +
 forecastSummary.forecastName());
            }

        } catch (ForecastException e) {
            System.err.println(e.awsErrorDetails().errorMessage());
            System.exit(1);
        }
    }
```

```
}
```

- For API details, see ListForecasts in *AWS SDK for Java 2.x API Reference*.

For a complete list of AWS SDK developer guides and code examples, see Using Forecast with an AWS SDK. This topic also includes information about getting started and details about previous SDK versions.

Security in Amazon Forecast

Cloud security at AWS is the highest priority. As an AWS customer, you benefit from data centers and network architectures that are built to meet the requirements of the most security-sensitive organizations.

Security is a shared responsibility between AWS and you. The shared responsibility model describes this as security *of* the cloud and security *in* the cloud:

- **Security of the cloud** – AWS is responsible for protecting the infrastructure that runs AWS services in the AWS Cloud. AWS also provides you with services that you can use securely. Third-party auditors regularly test and verify the effectiveness of our security as part of the AWS Compliance Programs. To learn about the compliance programs that apply to Amazon Forecast, see AWS Services in Scope by Compliance Program.
- **Security in the cloud** – Your responsibility is determined by the AWS service that you use. You are also responsible for other factors including the sensitivity of your data, your company's requirements, and applicable laws and regulations.

This documentation helps you understand how to apply the shared responsibility model when using Forecast. The following topics show you how to configure Forecast to meet your security and compliance objectives. You also learn how to use other AWS services that help you to monitor and secure your Forecast resources.

Topics

Data Protection in Amazon Forecast

The AWS shared responsibility model applies to data protection in Amazon Forecast. As described in this model, AWS is responsible for protecting the global infrastructure that runs all of the AWS Cloud. You are responsible for maintaining control over your content that is hosted on this infrastructure. You are also responsible for the security configuration and management tasks for the AWS services that you use. For more information about data privacy, see the Data Privacy FAQ. For information about data protection in Europe, see the AWS Shared Responsibility Model and GDPR blog post on the *AWS Security Blog*.

For data protection purposes, we recommend that you protect AWS account credentials and set up individual users with AWS IAM Identity Center or AWS Identity and Access Management (IAM). That way, each user is given only the permissions necessary to fulfill their job duties. We also recommend that you secure your data in the following ways:

- Use multi-factor authentication (MFA) with each account.
- Use SSL/TLS to communicate with AWS resources. We require TLS 1.2 and recommend TLS 1.3.
- Set up API and user activity logging with AWS CloudTrail.
- Use AWS encryption solutions, along with all default security controls within AWS services.
- Use advanced managed security services such as Amazon Macie, which assists in discovering and securing sensitive data that is stored in Amazon S3.
- If you require FIPS 140-2 validated cryptographic modules when accessing AWS through a command line interface or an API, use a FIPS endpoint. For more information about the available FIPS endpoints, see Federal Information Processing Standard (FIPS) 140-2.

We strongly recommend that you never put confidential or sensitive information, such as your customers' email addresses, into tags or free-form text fields such as a **Name** field. This includes when you work with Forecast or other AWS services using the console, API, AWS CLI, or AWS SDKs. Any data that you enter into tags or free-form text fields used for names may be used for billing or diagnostic logs. If you provide a URL to an external server, we strongly recommend that you do not include credentials information in the URL to validate your request to that server.

Encryption at Rest

In Amazon Forecast encryption configuration is provided during the CreateDataset and CreatePredictor operations. If encryption configuration is provided in the CreateDataset operation, your CMK and IAM Role for encryption at rest is used in the CreateDatasetImportJob operation.

For example, if you provide your key's KMSKeyArn and a RoleArn in the EncryptionConfig statement of the CreateDataset operation, Forecast will assume that role and use the key to encrypt the dataset. If no configuration is provided, then Forecast uses the default service keys for encryption. Additionally, if you provide the EncryptionConfig information for the CreatePredictor operation, then all subsequent operations, such as CreatePredictorExplanability, CreateForecast and CreatePredictorBacktestExportJob, will use the same configuration to perform encryption at rest. Again, if you do not provide an encryption configuration, then Forecast will use the default service encryption.

For any data stored in your Amazon S3 bucket, data is encrypted by the default Amazon S3 key. You can also use your own AWS KMS key to encrypt your data and give Forecast access to this key. For information about data encryption in Amazon S3 see Protecting data using encryption. For information about managing your own AWS KMS key, see Managing keys in the *AWS Key Management Service Developer Guide*.

Encryption in Transit and Processing

Amazon Forecast uses TLS with AWS certificates to encrypt any data sent to other AWS services. Any communication with other AWS services happens over HTTPS, and Forecast endpoints support only secure connections over HTTPS.

Amazon Forecast copies data out of your account and processes it in an internal AWS system. When processing data, Forecast encrypts data with either a Forecast AWS KMS key or any AWS KMS key you provide.

How Amazon Forecast uses grants in AWS KMS

Amazon Forecast requires a grant to use your customer managed key.

Forecast creates a grant using the IAM role that is passed during **EncryptionConfig** in the CreatePredictor or CreateDataset operation. Forecast assumes the role and does a create grant operation on your behalf. See Setup IAM role for more details.

However, when you create a predictor encrypted with a customer managed key, Amazon Forecast creates a grant on your behalf by sending a CreateGrant request to AWS KMS. Grants in AWS KMS are used to give Amazon Forecast access to an AWS KMS key in a customer account.

Amazon Forecast requires the grant so that it can use your customer managed key to send Decrypt requests to AWS KMS in order to read the encrypted dataset artifacts. Forecast also uses the grant

to send GenerateDataKey requests to AWS KMS in order to encrypt the training artifacts back to Amazon S3.

You can revoke access to the grant, or remove the service's access to the customer managed key at any time. If you do, Amazon Forecast won't be able to access any of the data encrypted by the customer managed key, which affects operations that are dependent on that data. For example, if you attempt to perform the CreateForecast operation on an encrypted predictor that Amazon Forecast can't access, then the operation will return an AccessDeniedException error.

Create a customer managed key

You can create a symmetric customer managed key by using the AWS Management Console or the AWS KMS API. To create a symmetric customer managed key, follow the steps for Creating symmetric customer managed key in the *AWS Key Management Service Developer Guide*.

Key policies control access to your customer managed key. Every customer managed key must have exactly one key policy, which contains statements that determine who can use the key and how they can use it. When you create your customer managed key, you can specify a key policy. For more information, see Managing access to customer managed keys in the *AWS Key Management Service Developer Guide*.

To use your customer managed key with Amazon Forecast resources, the following API operations must be permitted in the key policy:

- kms:DescribeKey – Provides the customer managed key details that allow Amazon Forecast to validate the key.
- kms:CreateGrant – Adds a grant to a customer managed key. Grants control access to a specified AWS KMS key, which allows access to grant operations that Amazon Forecast requires. This operation allows Amazon Forecast to call `GenerateDataKey` to generate an encrypted data key and store it, because the data key is not immediately used for encryption. Also, the operation allows Amazon Forecast to call `Decrypt` so that it can use the stored encrypted data key and access the encrypted data.
- kms:RetireGrant - Retire all the grants provided during `CreateGrant` operation after the operation is completed.

Note

Amazon Forecast performs `kms:Decrypt` and `kms:GenerateDataKey` validation on the caller's identity. You will receive an AccessDeniedException in the event that the caller doesn't have the relevant permissions. The key policy should also resemble the following code:

```
"Effect": "Allow",
"Principal": {
    "AWS": “AWS Invoking Identity”
},
"Action": [
    "kms:Decrypt",
    "kms:GenerateDataKey”
    ],
    "Resource": "*"
}
```

For more details, see IAM Policy.

The following are policy statement examples you can add for Amazon Forecast. These are the minimal permissions required, these can be added using IAM policies as well.

```
 "Statement" : [
   {"Sid" : "Allow access to principals authorized to use Amazon Forecast",
     "Effect" : "Allow",
     "Principal" : {"AWS" : "arn:aws:iam::111122223333:role/ROLE_PASSED_TO_FORECAST"
     },
     "Action" : [
       "kms:DescribeKey",
       "kms:CreateGrant",
       "kms:RetireGrant"
     ],
     "Resource" : "*",
     "Condition" : {"StringEquals" : {"kms:ViaService" :
"forecast.region.amazonaws.com",
         "kms:CallerAccount" : "111122223333"
       }
   },
   {"Sid": "Allow access for key administrators",
```

```
      "Effect": "Allow",
      "Principal": {"AWS": "arn:aws:iam::111122223333:root"
       },
      "Action" : [
        "kms:*"
       ],
      "Resource": "arn:aws:kms:region:111122223333:key/key_ID"
    }
  ]
```

See the *AWS Key Management Service Developer Guide* for more information about specifying permissions in a policy and troubleshooting key access.

Monitoring your encryption keys for Amazon Forecast Service

When you use an AWS KMS customer managed key with your Amazon Forecast Service resources, you can use AWS CloudTrail or Amazon CloudWatch Logs to track requests that Forecast sends to AWS KMS. The following examples are AWS CloudTrail events for `CreateGrant`, `RetireGrant`, and `DescribeKey` to monitor AWS KMS operations called by Amazon Forecast to access data encrypted by your customer managed key.

DescribeKey

```
{
    "eventVersion": "1.08",
    "userIdentity": {
        "type": "AssumedRole",
        "principalId": "AROAIGDTESTANDEXAMPLE:Sampleuser01",
        "arn": "arn:aws:sts::111122223333:assumed-role/Admin/Sampleuser01",
        "accountId": "111122223333",
        "accessKeyId": "AKIAIOSFODNN7EXAMPLE3",
        "sessionContext": {
            "sessionIssuer": {
                "type": "Role",
                "principalId": "AROAIGDTESTANDEXAMPLE:Sampleuser01",
                "arn": "arn:aws:sts::111122223333:assumed-role/Admin/Sampleuser01",
                "accountId": "111122223333",
                "userName": "Admin"
            },
            "webIdFederationData": {},
            "attributes": {
                "creationDate": "2022-10-05T21:16:23Z",
```

```
                "mfaAuthenticated": "false"
            }
        }
    },
    "eventTime": "2022-10-05T21:16:23Z",
    "eventSource": "kms.amazonaws.com",
    "eventName": "DescribeKey",
    "awsRegion": "region",
    "sourceIPAddress": "172.12.34.56",
    "userAgent": "ExampleDesktop/1.0 (V1; OS)",
    "requestParameters": {
        "keyId":
 "arn:aws:kms:region:111122223333:key/1234abcd-12ab-34cd-56ef-123456SAMPLE"
    },
    "responseElements": null,
    "requestID": "ff000af-00eb-00ce-0e00-ea000fb0fba0SAMPLE",
    "eventID": "ff000af-00eb-00ce-0e00-ea000fb0fba0SAMPLE",
    "readOnly": true,
    "resources": [
        {
            "accountId": "111122223333",
            "type": "AWS::KMS::Key",
            "ARN":
 "arn:aws:kms:region:111122223333:key/1234abcd-12ab-34cd-56ef-123456SAMPLE"
        }
    ],
    "eventType": "AwsApiCall",
    "managementEvent": true,
    "recipientAccountId": "111122223333",
    "eventCategory": "Management",
    "tlsDetails": {
        "tlsVersion": "TLSv1.2",
        "cipherSuite": "ECDHE-RSA-AES256-GCM-SHA384",
        "clientProvidedHostHeader": "kms.region.amazonaws.com"
    }
}
```

CreateGrant

```
{
    "eventVersion": "1.08",
    "userIdentity": {
        "type": "AssumedRole",
```

```
        "principalId": "AROAIGDTESTANDEXAMPLE:Sampleuser01",
        "arn": "arn:aws:sts::111122223333:assumed-role/Admin/Sampleuser01",
        "accountId": "111122223333",
        "accessKeyId": "AKIAIOSFODNN7EXAMPLE3",
        "sessionContext": {
            "sessionIssuer": {
                "type": "Role",
                "principalId": "AROAIGDTESTANDEXAMPLE:Sampleuser01",
                "arn": "arn:aws:sts::111122223333:assumed-role/Admin/Sampleuser01",
                "accountId": "111122223333",
                "userName": "Admin"
            },
            "webIdFederationData": {},
            "attributes": {
                "creationDate": "2022-10-05T23:10:27Z",
                "mfaAuthenticated": "false"
            }
        }
    },
    "eventTime": "2022-10-05T23:10:27Z",
    "eventSource": "kms.amazonaws.com",
    "eventName": "CreateGrant",
    "awsRegion": "region",
    "sourceIPAddress": "172.12.34.56",
    "userAgent": "ExampleDesktop/1.0 (V1; OS)",
    "requestParameters": {
        "operations": [
            "Decrypt",
            "GenerateDataKey"
        ],
        "granteePrincipal": "AWS Internal",
        "keyId":
"arn:aws:kms:region:111122223333:key/1234abcd-12ab-34cd-56ef-123456SAMPLE"
    },
    "responseElements": {
        "grantId":
"0ab0ac0d0b000f00ea00cc0a0e00fc00bce000c000f0000000c0bc0a0000aaafSAMPLE"
    },
    "requestID": "ff000af-00eb-00ce-0e00-ea000fb0fba0SAMPLE",
    "eventID": "ff000af-00eb-00ce-0e00-ea000fb0fba0SAMPLE",
    "readOnly": false,
    "resources": [
        {
            "accountId": "111122223333",
```

```
            "type": "AWS::KMS::Key",
            "ARN":
 "arn:aws:kms:region:111122223333:key/1234abcd-12ab-34cd-56ef-123456SAMPLE"
        }
    ],
    "eventType": "AwsApiCall",
    "managementEvent": true,
    "recipientAccountId": "111122223333",
    "eventCategory": "Management",
    "tlsDetails": {
        "tlsVersion": "TLSv1.2",
        "cipherSuite": "ECDHE-RSA-AES256-GCM-SHA384",
        "clientProvidedHostHeader": "kms.region.amazonaws.com"
    }
}
```

RetireGrant

```
{
    "eventVersion": "1.08",
    "userIdentity": {
        "type": "AssumedRole",
        "principalId": "AROAIGDTESTANDEXAMPLE:Sampleuser01",
        "arn": "arn:aws:sts::111122223333:assumed-role/Admin/Sampleuser01",
        "accountId": "111122223333",
        "accessKeyId": "AKIAIOSFODNN7EXAMPLE3",
        "sessionContext": {
            "sessionIssuer": {
                "type": "Role",
                "principalId": "AROAIGDTESTANDEXAMPLE:Sampleuser01",
                "arn": "arn:aws:sts::111122223333:assumed-role/Admin/Sampleuser01",
                "accountId": "111122223333",
                "userName": "Admin"
            },
            "webIdFederationData": {},
            "attributes": {
                "creationDate": "2022-10-06T04:56:14Z",
                "mfaAuthenticated": "false"
            }
        }
    },
    "eventTime": "2022-10-06T04:56:14Z",
    "eventSource": "kms.amazonaws.com",
```

```
    "eventName": "RetireGrant",
    "awsRegion": "region",
    "sourceIPAddress": "172.12.34.56",
    "userAgent": "ExampleDesktop/1.0 (V1; OS)",
    "requestParameters": null,
    "responseElements": null,
    "additionalEventData": {
        "grantId":
 "0ab0ac0d0b000f00ea00cc0a0e00fc00bce000c000f0000000c0bc0a0000aaafSAMPLE"
    },
    "requestID": "ff000af-00eb-00ce-0e00-ea000fb0fba0SAMPLE",
    "eventID": "ff000af-00eb-00ce-0e00-ea000fb0fba0SAMPLE",
    "readOnly": false,
    "resources": [
        {
            "accountId": "111122223333",
            "type": "AWS::KMS::Key",
            "ARN":
 "arn:aws:kms:region:111122223333:key/1234abcd-12ab-34cd-56ef-123456SAMPLE"
        }
    ],
    "eventType": "AwsApiCall",
    "managementEvent": true,
    "recipientAccountId": "111122223333",
    "eventCategory": "Management",
    "tlsDetails": {
        "tlsVersion": "TLSv1.2",
        "cipherSuite": "ECDHE-RSA-AES256-GCM-SHA384",
        "clientProvidedHostHeader": "kms.region.amazonaws.com"
    }
}
```

Identity and Access Management for Amazon Forecast

AWS Identity and Access Management (IAM) is an AWS service that helps an administrator securely control access to AWS resources. IAM administrators control who can be *authenticated* (signed in) and *authorized* (have permissions) to use Forecast resources. IAM is an AWS service that you can use with no additional charge.

Topics

- Audience

- Authenticating with identities
- Managing access using policies
- How Amazon Forecast works with IAM
- Identity-based policy examples for Amazon Forecast
- Troubleshooting Amazon Forecast identity and access

Audience

How you use AWS Identity and Access Management (IAM) differs, depending on the work that you do in Forecast.

Service user – If you use the Forecast service to do your job, then your administrator provides you with the credentials and permissions that you need. As you use more Forecast features to do your work, you might need additional permissions. Understanding how access is managed can help you request the right permissions from your administrator. If you cannot access a feature in Forecast, see Troubleshooting Amazon Forecast identity and access.

Service administrator – If you're in charge of Forecast resources at your company, you probably have full access to Forecast. It's your job to determine which Forecast features and resources your service users should access. You must then submit requests to your IAM administrator to change the permissions of your service users. Review the information on this page to understand the basic concepts of IAM. To learn more about how your company can use IAM with Forecast, see How Amazon Forecast works with IAM.

IAM administrator – If you're an IAM administrator, you might want to learn details about how you can write policies to manage access to Forecast. To view example Forecast identity-based policies that you can use in IAM, see Identity-based policy examples for Amazon Forecast.

Authenticating with identities

Authentication is how you sign in to AWS using your identity credentials. You must be *authenticated* (signed in to AWS) as the AWS account root user, as an IAM user, or by assuming an IAM role.

You can sign in to AWS as a federated identity by using credentials provided through an identity source. AWS IAM Identity Center (IAM Identity Center) users, your company's single sign-on authentication, and your Google or Facebook credentials are examples of federated identities.

When you sign in as a federated identity, your administrator previously set up identity federation using IAM roles. When you access AWS by using federation, you are indirectly assuming a role.

Depending on the type of user you are, you can sign in to the AWS Management Console or the AWS access portal. For more information about signing in to AWS, see How to sign in to your AWS account in the *AWS Sign-In User Guide*.

If you access AWS programmatically, AWS provides a software development kit (SDK) and a command line interface (CLI) to cryptographically sign your requests by using your credentials. If you don't use AWS tools, you must sign requests yourself. For more information about using the recommended method to sign requests yourself, see Signing AWS API requests in the *IAM User Guide*.

Regardless of the authentication method that you use, you might be required to provide additional security information. For example, AWS recommends that you use multi-factor authentication (MFA) to increase the security of your account. To learn more, see Multi-factor authentication in the *AWS IAM Identity Center User Guide* and Using multi-factor authentication (MFA) in AWS in the *IAM User Guide*.

AWS account root user

When you create an AWS account, you begin with one sign-in identity that has complete access to all AWS services and resources in the account. This identity is called the AWS account *root user* and is accessed by signing in with the email address and password that you used to create the account. We strongly recommend that you don't use the root user for your everyday tasks. Safeguard your root user credentials and use them to perform the tasks that only the root user can perform. For the complete list of tasks that require you to sign in as the root user, see Tasks that require root user credentials in the *IAM User Guide*.

Federated identity

As a best practice, require human users, including users that require administrator access, to use federation with an identity provider to access AWS services by using temporary credentials.

A *federated identity* is a user from your enterprise user directory, a web identity provider, the AWS Directory Service, the Identity Center directory, or any user that accesses AWS services by using credentials provided through an identity source. When federated identities access AWS accounts, they assume roles, and the roles provide temporary credentials.

For centralized access management, we recommend that you use AWS IAM Identity Center. You can create users and groups in IAM Identity Center, or you can connect and synchronize to a set of users and groups in your own identity source for use across all your AWS accounts and applications. For information about IAM Identity Center, see What is IAM Identity Center? in the *AWS IAM Identity Center User Guide*.

IAM users and groups

An *IAM user* is an identity within your AWS account that has specific permissions for a single person or application. Where possible, we recommend relying on temporary credentials instead of creating IAM users who have long-term credentials such as passwords and access keys. However, if you have specific use cases that require long-term credentials with IAM users, we recommend that you rotate access keys. For more information, see Rotate access keys regularly for use cases that require long-term credentials in the *IAM User Guide*.

An *IAM group* is an identity that specifies a collection of IAM users. You can't sign in as a group. You can use groups to specify permissions for multiple users at a time. Groups make permissions easier to manage for large sets of users. For example, you could have a group named *IAMAdmins* and give that group permissions to administer IAM resources.

Users are different from roles. A user is uniquely associated with one person or application, but a role is intended to be assumable by anyone who needs it. Users have permanent long-term credentials, but roles provide temporary credentials. To learn more, see When to create an IAM user (instead of a role) in the *IAM User Guide*.

IAM roles

An *IAM role* is an identity within your AWS account that has specific permissions. It is similar to an IAM user, but is not associated with a specific person. You can temporarily assume an IAM role in the AWS Management Console by switching roles. You can assume a role by calling an AWS CLI or AWS API operation or by using a custom URL. For more information about methods for using roles, see Using IAM roles in the *IAM User Guide*.

IAM roles with temporary credentials are useful in the following situations:

- **Federated user access** – To assign permissions to a federated identity, you create a role and define permissions for the role. When a federated identity authenticates, the identity is associated with the role and is granted the permissions that are defined by the role. For information about roles for federation, see Creating a role for a third-party Identity Provider

in the *IAM User Guide*. If you use IAM Identity Center, you configure a permission set. To control what your identities can access after they authenticate, IAM Identity Center correlates the permission set to a role in IAM. For information about permissions sets, see Permission sets in the *AWS IAM Identity Center User Guide*.

- **Temporary IAM user permissions** – An IAM user or role can assume an IAM role to temporarily take on different permissions for a specific task.
- **Cross-account access** – You can use an IAM role to allow someone (a trusted principal) in a different account to access resources in your account. Roles are the primary way to grant cross-account access. However, with some AWS services, you can attach a policy directly to a resource (instead of using a role as a proxy). To learn the difference between roles and resource-based policies for cross-account access, see How IAM roles differ from resource-based policies in the *IAM User Guide*.
- **Cross-service access** – Some AWS services use features in other AWS services. For example, when you make a call in a service, it's common for that service to run applications in Amazon EC2 or store objects in Amazon S3. A service might do this using the calling principal's permissions, using a service role, or using a service-linked role.
  - **Forward access sessions (FAS)** – When you use an IAM user or role to perform actions in AWS, you are considered a principal. When you use some services, you might perform an action that then initiates another action in a different service. FAS uses the permissions of the principal calling an AWS service, combined with the requesting AWS service to make requests to downstream services. FAS requests are only made when a service receives a request that requires interactions with other AWS services or resources to complete. In this case, you must have permissions to perform both actions. For policy details when making FAS requests, see Forward access sessions.
  - **Service role** – A service role is an IAM role that a service assumes to perform actions on your behalf. An IAM administrator can create, modify, and delete a service role from within IAM. For more information, see Creating a role to delegate permissions to an AWS service in the *IAM User Guide*.
  - **Service-linked role** – A service-linked role is a type of service role that is linked to an AWS service. The service can assume the role to perform an action on your behalf. Service-linked roles appear in your AWS account and are owned by the service. An IAM administrator can view, but not edit the permissions for service-linked roles.
- **Applications running on Amazon EC2** – You can use an IAM role to manage temporary credentials for applications that are running on an EC2 instance and making AWS CLI or AWS API requests. This is preferable to storing access keys within the EC2 instance. To assign an AWS role

to an EC2 instance and make it available to all of its applications, you create an instance profile that is attached to the instance. An instance profile contains the role and enables programs that are running on the EC2 instance to get temporary credentials. For more information, see Using an IAM role to grant permissions to applications running on Amazon EC2 instances in the *IAM User Guide*.

To learn whether to use IAM roles or IAM users, see When to create an IAM role (instead of a user) in the *IAM User Guide*.

Managing access using policies

You control access in AWS by creating policies and attaching them to AWS identities or resources. A policy is an object in AWS that, when associated with an identity or resource, defines their permissions. AWS evaluates these policies when a principal (user, root user, or role session) makes a request. Permissions in the policies determine whether the request is allowed or denied. Most policies are stored in AWS as JSON documents. For more information about the structure and contents of JSON policy documents, see Overview of JSON policies in the *IAM User Guide*.

Administrators can use AWS JSON policies to specify who has access to what. That is, which **principal** can perform **actions** on what **resources**, and under what **conditions**.

By default, users and roles have no permissions. To grant users permission to perform actions on the resources that they need, an IAM administrator can create IAM policies. The administrator can then add the IAM policies to roles, and users can assume the roles.

IAM policies define permissions for an action regardless of the method that you use to perform the operation. For example, suppose that you have a policy that allows the `iam:GetRole` action. A user with that policy can get role information from the AWS Management Console, the AWS CLI, or the AWS API.

Identity-based policies

Identity-based policies are JSON permissions policy documents that you can attach to an identity, such as an IAM user, group of users, or role. These policies control what actions users and roles can perform, on which resources, and under what conditions. To learn how to create an identity-based policy, see Creating IAM policies in the *IAM User Guide*.

Identity-based policies can be further categorized as *inline policies* or *managed policies*. Inline policies are embedded directly into a single user, group, or role. Managed policies are standalone

policies that you can attach to multiple users, groups, and roles in your AWS account. Managed policies include AWS managed policies and customer managed policies. To learn how to choose between a managed policy or an inline policy, see Choosing between managed policies and inline policies in the *IAM User Guide*.

Resource-based policies

Resource-based policies are JSON policy documents that you attach to a resource. Examples of resource-based policies are IAM *role trust policies* and Amazon S3 *bucket policies*. In services that support resource-based policies, service administrators can use them to control access to a specific resource. For the resource where the policy is attached, the policy defines what actions a specified principal can perform on that resource and under what conditions. You must specify a principal in a resource-based policy. Principals can include accounts, users, roles, federated users, or AWS services.

Resource-based policies are inline policies that are located in that service. You can't use AWS managed policies from IAM in a resource-based policy.

Access control lists (ACLs)

Access control lists (ACLs) control which principals (account members, users, or roles) have permissions to access a resource. ACLs are similar to resource-based policies, although they do not use the JSON policy document format.

Amazon S3, AWS WAF, and Amazon VPC are examples of services that support ACLs. To learn more about ACLs, see Access control list (ACL) overview in the *Amazon Simple Storage Service Developer Guide*.

Other policy types

AWS supports additional, less-common policy types. These policy types can set the maximum permissions granted to you by the more common policy types.

- **Permissions boundaries** – A permissions boundary is an advanced feature in which you set the maximum permissions that an identity-based policy can grant to an IAM entity (IAM user or role). You can set a permissions boundary for an entity. The resulting permissions are the intersection of an entity's identity-based policies and its permissions boundaries. Resource-based policies that specify the user or role in the `Principal` field are not limited by the permissions boundary. An explicit deny in any of these policies overrides the allow. For more information about permissions boundaries, see Permissions boundaries for IAM entities in the *IAM User Guide*.

- **Service control policies (SCPs)** – SCPs are JSON policies that specify the maximum permissions for an organization or organizational unit (OU) in AWS Organizations. AWS Organizations is a service for grouping and centrally managing multiple AWS accounts that your business owns. If you enable all features in an organization, then you can apply service control policies (SCPs) to any or all of your accounts. The SCP limits permissions for entities in member accounts, including each AWS account root user. For more information about Organizations and SCPs, see How SCPs work in the *AWS Organizations User Guide*.
- **Session policies** – Session policies are advanced policies that you pass as a parameter when you programmatically create a temporary session for a role or federated user. The resulting session's permissions are the intersection of the user or role's identity-based policies and the session policies. Permissions can also come from a resource-based policy. An explicit deny in any of these policies overrides the allow. For more information, see Session policies in the *IAM User Guide*.

Multiple policy types

When multiple types of policies apply to a request, the resulting permissions are more complicated to understand. To learn how AWS determines whether to allow a request when multiple policy types are involved, see Policy evaluation logic in the *IAM User Guide*.

How Amazon Forecast works with IAM

Before you use IAM to manage access to Forecast, learn what IAM features are available to use with Forecast.

IAM features you can use with Amazon Forecast

IAM feature	Forecast support
Identity-based policies	Yes
Resource-based policies	No
Policy actions	Yes
Policy resources	Yes
Policy condition keys (service-specific)	No

IAM feature	Forecast support
ACLs	No
ABAC (tags in policies)	Yes
Temporary credentials	Yes
Principal permissions	Yes
Service roles	Yes
Service-linked roles	No

To get a high-level view of how Forecast and other AWS services work with most IAM features, see AWS services that work with IAM in the *IAM User Guide*.

Identity-based policies for Forecast

Supports identity-based policies	Yes

Identity-based policies are JSON permissions policy documents that you can attach to an identity, such as an IAM user, group of users, or role. These policies control what actions users and roles can perform, on which resources, and under what conditions. To learn how to create an identity-based policy, see Creating IAM policies in the *IAM User Guide*.

With IAM identity-based policies, you can specify allowed or denied actions and resources as well as the conditions under which actions are allowed or denied. You can't specify the principal in an identity-based policy because it applies to the user or role to which it is attached. To learn about all of the elements that you can use in a JSON policy, see IAM JSON policy elements reference in the *IAM User Guide*.

Identity-based policy examples for Forecast

To view examples of Forecast identity-based policies, see Identity-based policy examples for Amazon Forecast.

Resource-based policies within Forecast

Supports resource-based policies	No

Resource-based policies are JSON policy documents that you attach to a resource. Examples of resource-based policies are IAM *role trust policies* and Amazon S3 *bucket policies*. In services that support resource-based policies, service administrators can use them to control access to a specific resource. For the resource where the policy is attached, the policy defines what actions a specified principal can perform on that resource and under what conditions. You must specify a principal in a resource-based policy. Principals can include accounts, users, roles, federated users, or AWS services.

To enable cross-account access, you can specify an entire account or IAM entities in another account as the principal in a resource-based policy. Adding a cross-account principal to a resource-based policy is only half of establishing the trust relationship. When the principal and the resource are in different AWS accounts, an IAM administrator in the trusted account must also grant the principal entity (user or role) permission to access the resource. They grant permission by attaching an identity-based policy to the entity. However, if a resource-based policy grants access to a principal in the same account, no additional identity-based policy is required. For more information, see How IAM roles differ from resource-based policies in the *IAM User Guide*.

Policy actions for Forecast

Supports policy actions	Yes

Administrators can use AWS JSON policies to specify who has access to what. That is, which **principal** can perform **actions** on what **resources**, and under what **conditions**.

The `Action` element of a JSON policy describes the actions that you can use to allow or deny access in a policy. Policy actions usually have the same name as the associated AWS API operation. There are some exceptions, such as *permission-only actions* that don't have a matching API operation. There are also some operations that require multiple actions in a policy. These additional actions are called *dependent actions*.

Include actions in a policy to grant permissions to perform the associated operation.

To see a list of Forecast actions, see Actions defined by Amazon Forecast in the *Service Authorization Reference*.

Policy actions in Forecast use the following prefix before the action:

```
forecast
```

To specify multiple actions in a single statement, separate them with commas.

```
"Action": [
      "forecast:action1",
      "forecast:action2"
         ]
```

Policy resources for Forecast

Supports policy resources	Yes

Administrators can use AWS JSON policies to specify who has access to what. That is, which **principal** can perform **actions** on what **resources**, and under what **conditions**.

The `Resource` JSON policy element specifies the object or objects to which the action applies. Statements must include either a `Resource` or a `NotResource` element. As a best practice, specify a resource using its Amazon Resource Name (ARN). You can do this for actions that support a specific resource type, known as *resource-level permissions*.

For actions that don't support resource-level permissions, such as listing operations, use a wildcard (*) to indicate that the statement applies to all resources.

```
"Resource": "*"
```

To see a list of Forecast resource types and their ARNs, see Resources defined by Amazon Forecast in the *Service Authorization Reference*. To learn with which actions you can specify the ARN of each resource, see Actions defined by Amazon Forecast.

To view examples of Forecast identity-based policies, see Identity-based policy examples for Amazon Forecast.

Policy condition keys for Forecast

Supports service-specific policy condition keys	No

Administrators can use AWS JSON policies to specify who has access to what. That is, which **principal** can perform **actions** on what **resources**, and under what **conditions**.

The `Condition` element (or `Condition` *block*) lets you specify conditions in which a statement is in effect. The `Condition` element is optional. You can create conditional expressions that use condition operators, such as equals or less than, to match the condition in the policy with values in the request.

If you specify multiple `Condition` elements in a statement, or multiple keys in a single `Condition` element, AWS evaluates them using a logical `AND` operation. If you specify multiple values for a single condition key, AWS evaluates the condition using a logical `OR` operation. All of the conditions must be met before the statement's permissions are granted.

You can also use placeholder variables when you specify conditions. For example, you can grant an IAM user permission to access a resource only if it is tagged with their IAM user name. For more information, see IAM policy elements: variables and tags in the *IAM User Guide*.

AWS supports global condition keys and service-specific condition keys. To see all AWS global condition keys, see AWS global condition context keys in the *IAM User Guide*.

To see a list of Forecast condition keys, see Condition keys for Amazon Forecast in the *Service Authorization Reference*. To learn with which actions and resources you can use a condition key, see Actions defined by Amazon Forecast.

To view examples of Forecast identity-based policies, see Identity-based policy examples for Amazon Forecast.

ACLs in Forecast

Supports ACLs	No

Access control lists (ACLs) control which principals (account members, users, or roles) have permissions to access a resource. ACLs are similar to resource-based policies, although they do not use the JSON policy document format.

ABAC with Forecast

Supports ABAC (tags in policies)	Yes

Attribute-based access control (ABAC) is an authorization strategy that defines permissions based on attributes. In AWS, these attributes are called *tags*. You can attach tags to IAM entities (users or roles) and to many AWS resources. Tagging entities and resources is the first step of ABAC. Then you design ABAC policies to allow operations when the principal's tag matches the tag on the resource that they are trying to access.

ABAC is helpful in environments that are growing rapidly and helps with situations where policy management becomes cumbersome.

To control access based on tags, you provide tag information in the condition element of a policy using the `aws:ResourceTag/key-name`, `aws:RequestTag/key-name`, or `aws:TagKeys` condition keys.

If a service supports all three condition keys for every resource type, then the value is **Yes** for the service. If a service supports all three condition keys for only some resource types, then the value is **Partial**.

For more information about ABAC, see What is ABAC? in the *IAM User Guide*. To view a tutorial with steps for setting up ABAC, see Use attribute-based access control (ABAC) in the *IAM User Guide*.

Using temporary credentials with Forecast

Supports temporary credentials	Yes

Some AWS services don't work when you sign in using temporary credentials. For additional information, including which AWS services work with temporary credentials, see AWS services that work with IAM in the *IAM User Guide*.

You are using temporary credentials if you sign in to the AWS Management Console using any method except a user name and password. For example, when you access AWS using your company's single sign-on (SSO) link, that process automatically creates temporary credentials. You also automatically create temporary credentials when you sign in to the console as a user and then

switch roles. For more information about switching roles, see Switching to a role (console) in the *IAM User Guide*.

You can manually create temporary credentials using the AWS CLI or AWS API. You can then use those temporary credentials to access AWS. AWS recommends that you dynamically generate temporary credentials instead of using long-term access keys. For more information, see Temporary security credentials in IAM.

Cross-service principal permissions for Forecast

Supports forward access sessions (FAS)	Yes

When you use an IAM user or role to perform actions in AWS, you are considered a principal. When you use some services, you might perform an action that then initiates another action in a different service. FAS uses the permissions of the principal calling an AWS service, combined with the requesting AWS service to make requests to downstream services. FAS requests are only made when a service receives a request that requires interactions with other AWS services or resources to complete. In this case, you must have permissions to perform both actions. For policy details when making FAS requests, see Forward access sessions.

Service roles for Forecast

Supports service roles	Yes

A service role is an IAM role that a service assumes to perform actions on your behalf. An IAM administrator can create, modify, and delete a service role from within IAM. For more information, see Creating a role to delegate permissions to an AWS service in the *IAM User Guide*.

> **Warning**
>
> Changing the permissions for a service role might break Forecast functionality. Edit service roles only when Forecast provides guidance to do so.

Service-linked roles for Forecast

Supports service-linked roles	No

A service-linked role is a type of service role that is linked to an AWS service. The service can assume the role to perform an action on your behalf. Service-linked roles appear in your AWS account and are owned by the service. An IAM administrator can view, but not edit the permissions for service-linked roles.

For details about creating or managing service-linked roles, see AWS services that work with IAM. Find a service in the table that includes a `Yes` in the **Service-linked role** column. Choose the **Yes** link to view the service-linked role documentation for that service.

Identity-based policy examples for Amazon Forecast

By default, users and roles don't have permission to create or modify Forecast resources. They also can't perform tasks by using the AWS Management Console, AWS Command Line Interface (AWS CLI), or AWS API. To grant users permission to perform actions on the resources that they need, an IAM administrator can create IAM policies. The administrator can then add the IAM policies to roles, and users can assume the roles.

To learn how to create an IAM identity-based policy by using these example JSON policy documents, see Creating IAM policies in the *IAM User Guide*.

For details about actions and resource types defined by Forecast, including the format of the ARNs for each of the resource types, see Actions, resources, and condition keys for Amazon Forecast in the *Service Authorization Reference*.

Whenever an operation is invoked, Amazon Forecast performs a set of authentication checks on the caller's permissions. These checks include the following:

- The caller's permission to invoke the operation is validated.
- If a role is provided within an operation, Amazon Forecast validates the PassRole permission for the role.
- If a KMS key is provided in the encryption configuration, then kms:Decrypt and kms:GenerateDataKey validation is performed on the caller's permissions. This key can differ for each operation performed in Amazon Forecast. You will receive an AccessDeniedException

in the event that you do not have the relevant permissions. The key policy should resemble the following code:

Example

```
"Effect": "Allow",
"Principal": {
    "AWS": “AWS Invoking Identity”
},
"Action": [
    "kms:Decrypt",
    "kms:GenerateDataKey”
    ],
    "Resource": "*"
}
```

Topics

- Policy best practices
- Using the Forecast console
- Allow users to view their own permissions
- AWS Managed (Predefined) Policies for Amazon Forecast
- Customer Managed Policy Examples

Policy best practices

Identity-based policies determine whether someone can create, access, or delete Forecast resources in your account. These actions can incur costs for your AWS account. When you create or edit identity-based policies, follow these guidelines and recommendations:

- **Get started with AWS managed policies and move toward least-privilege permissions** – To get started granting permissions to your users and workloads, use the *AWS managed policies* that grant permissions for many common use cases. They are available in your AWS account. We recommend that you reduce permissions further by defining AWS customer managed policies that are specific to your use cases. For more information, see AWS managed policies or AWS managed policies for job functions in the *IAM User Guide*.
- **Apply least-privilege permissions** – When you set permissions with IAM policies, grant only the permissions required to perform a task. You do this by defining the actions that can be taken on

specific resources under specific conditions, also known as *least-privilege permissions*. For more information about using IAM to apply permissions, see Policies and permissions in IAM in the *IAM User Guide*.

- **Use conditions in IAM policies to further restrict access** – You can add a condition to your policies to limit access to actions and resources. For example, you can write a policy condition to specify that all requests must be sent using SSL. You can also use conditions to grant access to service actions if they are used through a specific AWS service, such as AWS CloudFormation. For more information, see IAM JSON policy elements: Condition in the *IAM User Guide*.
- **Use IAM Access Analyzer to validate your IAM policies to ensure secure and functional permissions** – IAM Access Analyzer validates new and existing policies so that the policies adhere to the IAM policy language (JSON) and IAM best practices. IAM Access Analyzer provides more than 100 policy checks and actionable recommendations to help you author secure and functional policies. For more information, see IAM Access Analyzer policy validation in the *IAM User Guide*.
- **Require multi-factor authentication (MFA)** – If you have a scenario that requires IAM users or a root user in your AWS account, turn on MFA for additional security. To require MFA when API operations are called, add MFA conditions to your policies. For more information, see Configuring MFA-protected API access in the *IAM User Guide*.

For more information about best practices in IAM, see Security best practices in IAM in the *IAM User Guide*.

Using the Forecast console

To access the Amazon Forecast console, you must have a minimum set of permissions. These permissions must allow you to list and view details about the Forecast resources in your AWS account. If you create an identity-based policy that is more restrictive than the minimum required permissions, the console won't function as intended for entities (users or roles) with that policy.

You don't need to allow minimum console permissions for users that are making calls only to the AWS CLI or the AWS API. Instead, allow access to only the actions that match the API operation that they're trying to perform.

To ensure that users and roles can still use the Forecast console, also attach the following AWS managed policy to the entities. For more information, see Adding Permissions to a user in the *IAM User Guide*:

```
AWSForecastFullAccess
```

The following policy grants full access to all Amazon Forecast actions when using the console:

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": [
                "forecast:*"
            ],
            "Resource": "*"
        },
        {
            "Effect": "Allow",
            "Action": [
                "iam:PassRole"
            ],
            "Resource": "*",
            "Condition": {
                "StringEquals": {
                    "iam:PassedToService": "forecast.amazonaws.com"
                }
            }
        }
    ]
}
```

Allow users to view their own permissions

This example shows how you might create a policy that allows IAM users to view the inline and managed policies that are attached to their user identity. This policy includes permissions to complete this action on the console or programmatically using the AWS CLI or AWS API.

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Sid": "ViewOwnUserInfo",
            "Effect": "Allow",
```

```
            "Action": [
                "iam:GetUserPolicy",
                "iam:ListGroupsForUser",
                "iam:ListAttachedUserPolicies",
                "iam:ListUserPolicies",
                "iam:GetUser"
            ],
            "Resource": ["arn:aws:iam::*:user/${aws:username}"]
        },
        {
            "Sid": "NavigateInConsole",
            "Effect": "Allow",
            "Action": [
                "iam:GetGroupPolicy",
                "iam:GetPolicyVersion",
                "iam:GetPolicy",
                "iam:ListAttachedGroupPolicies",
                "iam:ListGroupPolicies",
                "iam:ListPolicyVersions",
                "iam:ListPolicies",
                "iam:ListUsers"
            ],
            "Resource": "*"
        }
    ]
}
```

AWS Managed (Predefined) Policies for Amazon Forecast

AWS addresses many common use cases by providing standalone IAM policies that are created and administered by AWS. These AWS managed policies grant necessary permissions for common use cases so that you can avoid having to investigate which permissions are needed. For more information, see AWS Managed Policies in the *IAM User Guide*.

The following AWS managed policies, which you can attach to users in your account, are specific to Amazon Forecast:

- **AmazonForecastFullAccess** – Grants full access to Amazon Forecast resources and all of the supported operations.

You can review these permissions policies by signing in to the IAM console and searching for them.

You can also create your own custom IAM policies to allow permissions for Amazon Forecast actions and resources. You can attach these custom policies to the IAM users or groups that require them.

Customer Managed Policy Examples

In this section, you can find example user policies that grant permissions for various Amazon Forecast actions. These policies work when you are using the AWS SDKs or the AWS CLI. When you are using the console, see Using the Forecast console.

Examples

- Example 1: Grant Account Administrator Permissions
- Example 2: Allow All Amazon Forecast and IAM PassRole Actions
- Example 3: Allow All Amazon Forecast actions while limiting IAM PassRole Actions
- Example 4: Action-based Policy: Amazon Forecast Read-Only Access
- Example 5: Allow all Amazon Forecast Actions with Pass Role and KMS Actions

Example 1: Grant Account Administrator Permissions

After you set up an account (see Sign Up for AWS), you create an administrator user to manage your account. The administrator user can create users and manage their permissions.

To grant the administrator user all of the permissions available for your account, attach the following permissions policy to that user:

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": "*",
            "Resource": "*"
        }
    ]
}
```

Example 2: Allow All Amazon Forecast and IAM PassRole Actions

You might choose to create a user who has permissions for all Amazon Forecast actions but not for any of your other services (think of this user as a service-specific administrator). Attach the following permissions policy to this user:

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": [
                "forecast:*"
            ],
            "Resource": "*"
        },
        {
            "Effect": "Allow",
            "Action": [
                "iam:PassRole"
            ],
            "Resource": "*",
            "Condition": {
                "StringEquals": {
                    "iam:PassedToService": "forecast.amazonaws.com"
                }
            }
        }
    ]
}
```

Example 3: Allow All Amazon Forecast actions while limiting IAM PassRole Actions

You might choose to create a user who has permissions for all Amazon Forecast actions while limiting their IAM PassRole actions. Attach the following permissions policy to this user:

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": [
                "forecast:*"
```

```
            ],
            "Resource": "*"
        },
        {
            "Effect": "Allow",
            "Action": [
                "iam:PassRole"
            ],
            "Resource": "arn:aws:iam::EXAMPLE_ACCOUNT_ID_12349858:role/
EXAMPLE_ROLE_TO_ALLOW_TO_PASS",
            "Condition": {
                "StringEquals": {
                    "iam:PassedToService": "forecast.amazonaws.com"
                }
            }
        }
    ]
}
```

Example 4: Action-based Policy: Amazon Forecast Read-Only Access

The following policy grants permissions to Amazon Forecast actions that allow a user to list and describe resources:

```
{
    "Version": "2012-10-17",
    "Statement": [
      {
        "Effect": "Allow",
        "Action": [
          "forecast:DescribeDataset",
          "forecast:DescribeDatasetGroup",
          "forecast:DescribeDatasetImportJob",
          "forecast:DescribeForecast",
          "forecast:DescribeForecastExportJob",
          "forecast:DescribePredictor",
          "forecast:ListDatasetGroups",
          "forecast:ListDatasetImportJobs",
          "forecast:ListDatasets",
          "forecast:ListDatasetExportJobs",
          "forecast:ListForecasts",
          "forecast:ListPredictors"
        ],
```

```
        "Resource": "*"
      }
    ]
}
```

Example 5: Allow all Amazon Forecast Actions with Pass Role and KMS Actions

You can create a user who has permissions for all Amazon Forecast actions, but does not have permissions for any other services, using a cross account Customer Managed Key for Encryption in Amazon Forecast. For more information, see AWS Cross Account Key policy in the AWS Key Management Service Developer Guide.

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": [
                "forecast:*"
            ],
            "Resource": "*"
        },
        {
            "Effect": "Allow",
            "Action": [
                "iam:PassRole"
            ],
            "Resource": "*",
            "Condition": {
                "StringEquals": {
                    "iam:PassedToService": "forecast.amazonaws.com"
                }
            }
        },
        {
            "Effect": "Allow",
            "Action": [
                "kms:Decrypt",
                "kms:GenerateDataKey"
            ],
            "Resource": "arn:aws:iam::1234567890:key/example_key"
        }
    ]
```

```
}
```

Troubleshooting Amazon Forecast identity and access

Use the following information to help you diagnose and fix common issues that you might encounter when working with Forecast and IAM.

Topics

I am not authorized to perform an action in Forecast

If you receive an error that you're not authorized to perform an action, your policies must be updated to allow you to perform the action.

The following example error occurs when the `mateojackson` IAM user tries to use the console to view details about a fictional *`my-example-widget`* resource but doesn't have the fictional `forecast:`*`GetWidget`* permissions.

```
User: arn:aws:iam::123456789012:user/mateojackson is not authorized to perform:
 forecast:GetWidget on resource: my-example-widget
```

In this case, the policy for the `mateojackson` user must be updated to allow access to the *`my-example-widget`* resource by using the `forecast:`*`GetWidget`* action.

If you need help, contact your AWS administrator. Your administrator is the person who provided you with your sign-in credentials.

I am not authorized to perform iam:PassRole

If you receive an error that you're not authorized to perform the `iam:PassRole` action, your policies must be updated to allow you to pass a role to Forecast.

Some AWS services allow you to pass an existing role to that service instead of creating a new service role or service-linked role. To do this, you must have permissions to pass the role to the service.

The following example error occurs when an IAM user named `marymajor` tries to use the console to perform an action in Forecast. However, the action requires the service to have permissions that are granted by a service role. Mary does not have permissions to pass the role to the service.

```
User: arn:aws:iam::123456789012:user/marymajor is not authorized to perform:
 iam:PassRole
```

In this case, Mary's policies must be updated to allow her to perform the `iam:PassRole` action.

If you need help, contact your AWS administrator. Your administrator is the person who provided you with your sign-in credentials.

I'm an Administrator and Want to Allow Others to Access Forecast

To allow others to access Forecast, you must create an IAM entity (user or role) for the person or application that needs access. They will use the credentials for that entity to access AWS. You must then attach a policy to the entity that grants them the correct permissions in Forecast.

To get started right away, see Creating your first IAM delegated user and group in the *IAM User Guide*.

I want to allow people outside of my AWS account to access my Forecast resources

You can create a role that users in other accounts or people outside of your organization can use to access your resources. You can specify who is trusted to assume the role. For services that support resource-based policies or access control lists (ACLs), you can use those policies to grant people access to your resources.

To learn more, consult the following:

- To learn whether Forecast supports these features, see How Amazon Forecast works with IAM.
- To learn how to provide access to your resources across AWS accounts that you own, see Providing access to an IAM user in another AWS account that you own in the *IAM User Guide*.
- To learn how to provide access to your resources to third-party AWS accounts, see Providing access to AWS accounts owned by third parties in the *IAM User Guide*.
- To learn how to provide access through identity federation, see Providing access to externally authenticated users (identity federation) in the *IAM User Guide*.
- To learn the difference between using roles and resource-based policies for cross-account access, see How IAM roles differ from resource-based policies in the *IAM User Guide*.

Logging and Monitoring in Amazon Forecast

Monitoring is an important part of maintaining the reliability, availability, and performance of your Amazon Forecast applications. To monitor Amazon Forecast API calls, you can use AWS CloudTrail. To monitor the status of your Forecast assets and processes, use Amazon CloudWatch.

Topics

Logging Forecast API Calls with AWS CloudTrail

Amazon Forecast is integrated with AWS CloudTrail, a service that provides a record of actions taken by a user, role, or an AWS service in Forecast. CloudTrail captures all API calls for Forecast as events. The calls captured include calls from the Forecast console and code calls to the Forecast API operations. If you create a trail, you can enable continuous delivery of CloudTrail events to an Amazon Simple Storage Service (Amazon S3) bucket, including events for Forecast. If you don't configure a trail, you can still view the most recent events in the CloudTrail console in **Event history**. Using the information collected by CloudTrail, you can determine the request that was made to Forecast, the IP address from which the request was made, who made the request, when it was made, and additional details.

To learn more about CloudTrail, see the AWS CloudTrail User Guide.

Forecast Information in CloudTrail

CloudTrail is enabled on your AWS account when you create the account. When activity occurs in Forecast, that activity is recorded in a CloudTrail event along with other AWS service events in **Event history**. You can view, search, and download recent events in your AWS account. For more information, see Viewing Events with CloudTrail Event History.

For an ongoing record of events in your AWS account, including events for Forecast, create a trail. A *trail* enables CloudTrail to deliver log files to an Amazon S3 bucket. By default, when you create a trail in the console, the trail applies to all AWS Regions. The trail logs events from all Regions in the AWS partition and delivers the log files to the Amazon S3 bucket that you specify. Additionally, you can configure other AWS services to further analyze and act upon the event data collected in CloudTrail logs. For more information, see the following:

- Overview for Creating a Trail
- CloudTrail Supported Services and Integrations
- Configuring Amazon SNS Notifications for CloudTrail
- Receiving CloudTrail Log Files from Multiple Regions and Receiving CloudTrail Log Files from Multiple Accounts

All Forecast actions are logged by CloudTrail and are documented in the Amazon Forecast Developer Guide. For example, calls to the `CreateDataset` and `CreateForecast` actions generate entries in the CloudTrail log files.

Every event or log entry contains information about who generated the request. The identity information helps you determine the following:

- Whether the request was made with root or AWS Identity and Access Management (IAM) user credentials.
- Whether the request was made with temporary security credentials for a role or federated user.
- Whether the request was made by another AWS service.

For more information, see the CloudTrail userIdentity Element.

Understanding Forecast Log File Entries

A trail is a configuration that enables delivery of events as log files to an Amazon S3 bucket that you specify. CloudTrail log files contain one or more log entries. An event represents a single request from any source and includes information about the requested action, the date and time of the action, request parameters, and so on. CloudTrail log files aren't an ordered stack trace of the public API calls, so they don't appear in any specific order.

The following example shows a CloudTrail log entry that demonstrates the `CreateDataset` action.

```
{
 "eventVersion": "1.05",
 "userIdentity": {
   "type": "IAMUser",
   "principalId": "AIDAIQ4PAJSMEEPNEXAMPLE",
   "arn": "arn:aws:iam::acct-id:user/userxyz",
```

```
    "accountId": "111111111111",
    "accessKeyId": "AKIAIOSFODNN7EXAMPLE",
    "userName": "userxyz"
  },
  "eventTime": "2018-11-21T23:53:06Z",
  "eventSource": "forecast.amazonaws.com",
  "eventName": "CreateDataset",
  "awsRegion": "us-west-2",
  "sourceIPAddress": "192.168.0.1",
  "userAgent": "Boto3/1.7.82 Python/3.6.5 Linux/4.14.72-68.55.amzn1.x86_64
Botocore/1.10.84",
  "requestParameters": {
    "domain": "CUSTOM",
    "datasetType": "TARGET_TIME_SERIES",
    "dataFormat": "CSV",
    "datasetName": "forecast_test_script_ds",
    "dataFrequency": "D",
    "timeStampFormat": "yyyy-MM-dd",
    "schema": {
      "attributes": [
        {
          "attributeName": "item_id",
          "attributeType": "string"
        },
        {
          "attributeName": "timestamp",
          "attributeType": "timestamp"
        },
        {
          "attributeName": "target_value",
          "attributeType": "float"
        },
        {
          "attributeName": "visits",
          "attributeType": "float"
        },
        {
          "attributeName": "was_open",
          "attributeType": "float"
        },
        {
          "attributeName": "promotion_applied",
          "attributeType": "float"
        }
```

```
      ]
    }
  },
  "responseElements": {
    "datasetName": "forecast_test_script_ds",
    "datasetArn": "arn:aws:forecast:us-west-2:acct-id:ds/forecast_test_script_ds"
  },
  "requestID": "EXAMPLE8-90ab-cdef-fedc-ba987EXAMPLE",
  "eventID": "EXAMPLE8-90ab-cdef-fedc-ba987EXAMPLE",
  "eventType": "AwsApiCall",
  "recipientAccountId": "111111111111"
}
```

CloudWatch Metrics for Amazon Forecast

This section contains information about the Amazon CloudWatch metrics available for Amazon Forecast.

The following table lists the Amazon Forecast metrics.

Metric	Dimension	Unit	Statistics	Description
DatasetSize		Kilobytes	Average, Sum, Min, Max	The total size of the datasets imported by Amazon Forecast into the customer's account.
DatasetSize	DatasetArn DatasetImportJobArn	Kilobytes	Average, Sum	The size of the dataset imported by the CreateDatasetImportJob operation.
CreatePredictorEvaluationTime	PredictorArn	Seconds	Average, Sum	The time taken for training, inference, and metrics for a specific predictor. Amazon Forecast normalizes the compute costs to a c5.xlarge instance to arrive at the number of hours consumed by the training job.

Metric	Dimension	Unit	Statistics	Description
CreateForecastEvaluationTime	ForecastArn	Seconds	Average, Sum	The time taken for training and inference during forecast generation. Amazon Forecast normalizes the compute costs to a c5.xlarge instance to arrive at the number of hours consumed by the training job.
TimeSeriesForecastsGenerated		Count	Average, Sum, Min, Max	The number of unique time series forecasts generated for each quantile across all predictors in the account. Forecasts are billed to the nearest 1000 and charged on a per 1,000 basis.
TimeSeriesForecastsGenerated	PredictorArn	Count	Average, Sum, Min, Max	The number of unique time series forecasts generated for each quantile across all predictors in the account. Forecasts are billed to the nearest 1,000 and charged on a per 1,000 basis.
TimeSeriesForecastsGenerated	PredictorArn ForecastArn	Count	Average, Sum, Min, Max	The number of unique time series forecasts generated for each quantile across all predictors in the account. Forecasts are billed to the nearest 1,000 and charged on a per 1,000 basis.
ForecastDataPointsGenerated	PredictorArn ForecastArn	Count	Average, Sum, Min, Max	The number of unique datapoints generated for each forecast across all predictors in the account. Forecasts are billed to the nearest 1,000 and charged on a per 1,000 basis.

Compliance Validation for Amazon Forecast

Third-party auditors assess the security and compliance of Amazon Forecast as part of multiple AWS compliance programs. These include SOC, PCI, HIPAA, and others.

For a list of AWS services in scope of specific compliance programs, see AWS Services in Scope by Compliance Program. For general information, see AWS Compliance Programs.

You can download third-party audit reports using AWS Artifact. For more information, see Downloading Reports in AWS Artifact.

Your compliance responsibility when using Forecast is determined by the sensitivity of your data, your company's compliance objectives, and applicable laws and regulations. AWS provides the following resources to help with compliance:

- Security and Compliance Quick Start Guides – These deployment guides discuss architectural considerations and provide steps for deploying security- and compliance-focused baseline environments on AWS.
- Architecting for HIPAA Security and Compliance Whitepaper – This whitepaper describes how companies can use AWS to create HIPAA-compliant applications.
- AWS Compliance Resources – This collection of workbooks and guides might apply to your industry and location.
- Evaluating Resources with Rules in the *AWS Config Developer Guide* – The AWS Config service assesses how well your resource configurations comply with internal practices, industry guidelines, and regulations.
- AWS Security Hub – This AWS service provides a comprehensive view of your security state within AWS that helps you check your compliance with security industry standards and best practices.

Resilience in Amazon Forecast

The AWS global infrastructure is built around AWS Regions and Availability Zones. AWS Regions provide multiple physically separated and isolated Availability Zones, which are connected with low-latency, high-throughput, and highly redundant networking. With Availability Zones, you can design and operate applications and databases that automatically fail over between zones without interruption. Availability Zones are more highly available, fault tolerant, and scalable than traditional single or multiple data center infrastructures.

For more information about AWS Regions and Availability Zones, see AWS Global Infrastructure.

Infrastructure Security in Amazon Forecast

As a managed service, Amazon Forecast is protected by AWS global network security. For information about AWS security services and how AWS protects infrastructure, see AWS Cloud Security. To design your AWS environment using the best practices for infrastructure security, see Infrastructure Protection in *Security Pillar AWS Well-Architected Framework*.

You use AWS published API calls to access Forecast through the network. Clients must support the following:

- Transport Layer Security (TLS). We require TLS 1.2 and recommend TLS 1.3.
- Cipher suites with perfect forward secrecy (PFS) such as DHE (Ephemeral Diffie-Hellman) or ECDHE (Elliptic Curve Ephemeral Diffie-Hellman). Most modern systems such as Java 7 and later support these modes.

Additionally, requests must be signed by using an access key ID and a secret access key that is associated with an IAM principal. Or you can use the AWS Security Token Service (AWS STS) to generate temporary security credentials to sign requests.

Forecast and interface VPC endpoints (AWS PrivateLink)

If you use Amazon Virtual Private Cloud (Amazon VPC) to host your AWS resources, you can establish a private connection between your VPC and Amazon Forecast. This connection allows Amazon Forecast to communicate with your resources on your VPC without going through the public internet.

Amazon VPC is an AWS service that you use to launch AWS resources in a virtual private cloud (VPC) or virtual network that you define. With a VPC, you have control over your network settings, such the IP address range, subnets, route tables, and network gateways. With VPC endpoints, the AWS network handles the routing between your VPC and AWS services.

To connect your VPC to Amazon Forecast, you define an interface VPC endpoint for Amazon Forecast. An interface endpoint is an elastic network interface with a private IP address that serves as an entry point for traffic destined to a supported AWS service. The endpoint provides reliable, scalable connectivity to Amazon Forecast—and it doesn't require an internet gateway, a

network address translation (NAT) instance, or a VPN connection. For more information, see What is Amazon VPC in the *Amazon VPC User Guide*.

Interface VPC endpoints are enabled by AWS PrivateLink. This AWS technology enables private communication between AWS services by using an elastic network interface with private IP addresses.

> **Note**
>
> All Amazon Forecast Federal Information Processing Standard (FIPS) endpoints are supported by AWS PrivateLink.

Considerations for Forecast VPC endpoints

Before you set up an interface VPC endpoint for Forecast, ensure that you review Interface endpoint properties and limitations in the *Amazon VPC User Guide*.

Forecast supports making calls to all of its API actions from your VPC.

Creating an interface VPC endpoint for Forecast

You can create a VPC endpoint for the Forecast service with either the Amazon VPC console or the AWS Command Line Interface (AWS CLI). For more information, see Creating an interface endpoint in the *Amazon VPC User Guide*.

You can create two types of VPC endpoints to use with Amazon Forecast:

- A VPC endpoint to use with Amazon Forecast operations. For most users, this is the most suitable type of VPC endpoint.
  - com.amazonaws.*region*.forecast
  - com.amazonaws.*region*.forecastquery
- A VPC endpoint for Amazon Forecast operations with endpoints that comply with the Federal Information Processing Standard (FIPS) Publication 140-2 US government standard (available in select regions, see Amazon Forecast endpoints and quotas).
  - com.amazonaws.*region*.forecast-fips
  - com.amazonaws.*region*.forecastquery-fips

If you enable private DNS for the endpoint, you can make API requests to Forecast using its default DNS name for the Region, for example, `forecast.us-east-1.amazonaws.com`.

For more information, see Accessing a service through an interface endpoint in the *Amazon VPC User Guide*.

Creating a VPC endpoint policy for Forecast

You can attach an endpoint policy to your VPC endpoint that controls access to Forecast. The policy specifies the following information:

- The principal that can perform actions.
- The actions that can be performed.
- The resources on which actions can be performed.

For more information, see Controlling access to services with VPC endpoints in the *Amazon VPC User Guide*.

Example: VPC endpoint policy allowing all Forecast actions and passRole actions

When attached to an endpoint, this policy grants access to all Forecast actions and passRole actions.

```
{
    "Statement": [
        {
            "Principal": "*",
            "Effect": "Allow",
            "Action": [
                "forecast:*",
                "iam:PassRole"
            ],
            "Resource": "*"
        }
    ]
}
```

Example: VPC endpoint policy allowing Forecast ListDatasets actions

When attached to an endpoint, this policy grants access to the listed Forecast ListDatasets actions.

```
{
    "Statement": [
        {
            "Principal": "*",
            "Effect": "Allow",
            "Action": [
                "forecast:ListDatasets"
            ],
            "Resource": "*"
        }
    ]
}
```

API Reference

This section provides documentation for the Amazon Forecast API operations.

Topics

Actions

The following actions are supported by Amazon Forecast Service:

- DeleteExplainabilityExport
- DeleteForecast
- DeleteForecastExportJob
- DeleteMonitor
- DeletePredictor
- DeletePredictorBacktestExportJob
- DeleteResourceTree
- DeleteWhatIfAnalysis
- DeleteWhatIfForecast
- DeleteWhatIfForecastExport
- DescribeAutoPredictor
- DescribeDataset
- DescribeDatasetGroup
- DescribeDatasetImportJob
- DescribeExplainability
- DescribeExplainabilityExport
- DescribeForecast
- DescribeForecastExportJob
- DescribeMonitor
- DescribePredictor
- DescribePredictorBacktestExportJob
- DescribeWhatIfAnalysis
- DescribeWhatIfForecast
- DescribeWhatIfForecastExport
- GetAccuracyMetrics
- ListDatasetGroups
- ListDatasetImportJobs
- ListDatasets
- ListExplainabilities
- ListExplainabilityExports

- ListForecastExportJobs
- ListForecasts
- ListMonitorEvaluations
- ListMonitors
- ListPredictorBacktestExportJobs
- ListPredictors
- ListTagsForResource
- ListWhatIfAnalyses
- ListWhatIfForecastExports
- ListWhatIfForecasts
- ResumeResource
- StopResource
- TagResource
- UntagResource
- UpdateDatasetGroup

The following actions are supported by Amazon Forecast Query Service:

- QueryForecast
- QueryWhatIfForecast

Amazon Forecast Service

The following actions are supported by Amazon Forecast Service:

- CreateAutoPredictor
- CreateDataset
- CreateDatasetGroup
- CreateDatasetImportJob
- CreateExplainability
- CreateExplainabilityExport
- CreateForecast

- CreateForecastExportJob
- CreateMonitor
- CreatePredictor
- CreatePredictorBacktestExportJob
- CreateWhatIfAnalysis
- CreateWhatIfForecast
- CreateWhatIfForecastExport
- DeleteDataset
- DeleteDatasetGroup
- DeleteDatasetImportJob
- DeleteExplainability
- DeleteExplainabilityExport
- DeleteForecast
- DeleteForecastExportJob
- DeleteMonitor
- DeletePredictor
- DeletePredictorBacktestExportJob
- DeleteResourceTree
- DeleteWhatIfAnalysis
- DeleteWhatIfForecast
- DeleteWhatIfForecastExport
- DescribeAutoPredictor
- DescribeDataset
- DescribeDatasetGroup
- DescribeDatasetImportJob
- DescribeExplainability
- DescribeExplainabilityExport
- DescribeForecast
- DescribeForecastExportJob
- DescribeMonitor

- DescribePredictor
- DescribePredictorBacktestExportJob
- DescribeWhatIfAnalysis
- DescribeWhatIfForecast
- DescribeWhatIfForecastExport
- GetAccuracyMetrics
- ListDatasetGroups
- ListDatasetImportJobs
- ListDatasets
- ListExplainabilities
- ListExplainabilityExports
- ListForecastExportJobs
- ListForecasts
- ListMonitorEvaluations
- ListMonitors
- ListPredictorBacktestExportJobs
- ListPredictors
- ListTagsForResource
- ListWhatIfAnalyses
- ListWhatIfForecastExports
- ListWhatIfForecasts
- ResumeResource
- StopResource
- TagResource
- UntagResource
- UpdateDatasetGroup

CreateAutoPredictor

Service: Amazon Forecast Service

Creates an Amazon Forecast predictor.

Amazon Forecast creates predictors with AutoPredictor, which involves applying the optimal combination of algorithms to each time series in your datasets. You can use CreateAutoPredictor to create new predictors or upgrade/retrain existing predictors.

Creating new predictors

The following parameters are required when creating a new predictor:

- `PredictorName` - A unique name for the predictor.
- `DatasetGroupArn` - The ARN of the dataset group used to train the predictor.
- `ForecastFrequency` - The granularity of your forecasts (hourly, daily, weekly, etc).
- `ForecastHorizon` - The number of time-steps that the model predicts. The forecast horizon is also called the prediction length.

When creating a new predictor, do not specify a value for `ReferencePredictorArn`.

Upgrading and retraining predictors

The following parameters are required when retraining or upgrading a predictor:

- `PredictorName` - A unique name for the predictor.
- `ReferencePredictorArn` - The ARN of the predictor to retrain or upgrade.

When upgrading or retraining a predictor, only specify values for the `ReferencePredictorArn` and `PredictorName`.

Request Syntax

```
{
   "DataConfig": {
      "AdditionalDatasets": [
         {
            "Configuration": {
               "string" : [ "string" ]
            },
```

```
            "Name": "string"
         }
      ],
      "AttributeConfigs": [
         {
            "AttributeName": "string",
            "Transformations": {
               "string" : "string"
            }
         }
      ],
      "DatasetGroupArn": "string"
   },
   "EncryptionConfig": {
      "KMSKeyArn": "string",
      "RoleArn": "string"
   },
   "ExplainPredictor": boolean,
   "ForecastDimensions": [ "string" ],
   "ForecastFrequency": "string",
   "ForecastHorizon": number,
   "ForecastTypes": [ "string" ],
   "MonitorConfig": {
      "MonitorName": "string"
   },
   "OptimizationMetric": "string",
   "PredictorName": "string",
   "ReferencePredictorArn": "string",
   "Tags": [
      {
         "Key": "string",
         "Value": "string"
      }
   ],
   "TimeAlignmentBoundary": {
      "DayOfMonth": number,
      "DayOfWeek": "string",
      "Hour": number,
      "Month": "string"
   }
}
```

Request Parameters

The request accepts the following data in JSON format.

DataConfig

The data configuration for your dataset group and any additional datasets.

Type: DataConfig object

Required: No

EncryptionConfig

An AWS Key Management Service (KMS) key and an AWS Identity and Access Management (IAM) role that Amazon Forecast can assume to access the key. You can specify this optional object in the CreateDataset and CreatePredictor requests.

Type: EncryptionConfig object

Required: No

ExplainPredictor

Create an Explainability resource for the predictor.

Type: Boolean

Required: No

ForecastDimensions

An array of dimension (field) names that specify how to group the generated forecast.

For example, if you are generating forecasts for item sales across all your stores, and your dataset contains a `store_id` field, you would specify `store_id` as a dimension to group sales forecasts for each store.

Type: Array of strings

Array Members: Minimum number of 1 item. Maximum number of 10 items.

Length Constraints: Minimum length of 1. Maximum length of 63.

Pattern: `^[a-zA-Z][a-zA-Z0-9_]*`

Required: No

ForecastFrequency

The frequency of predictions in a forecast.

Valid intervals are an integer followed by Y (Year), M (Month), W (Week), D (Day), H (Hour), and min (Minute). For example, "1D" indicates every day and "15min" indicates every 15 minutes. You cannot specify a value that would overlap with the next larger frequency. That means, for example, you cannot specify a frequency of 60 minutes, because that is equivalent to 1 hour. The valid values for each frequency are the following:

- Minute - 1-59
- Hour - 1-23
- Day - 1-6
- Week - 1-4
- Month - 1-11
- Year - 1

Thus, if you want every other week forecasts, specify "2W". Or, if you want quarterly forecasts, you specify "3M".

The frequency must be greater than or equal to the TARGET_TIME_SERIES dataset frequency.

When a RELATED_TIME_SERIES dataset is provided, the frequency must be equal to the RELATED_TIME_SERIES dataset frequency.

Type: String

Length Constraints: Minimum length of 1. Maximum length of 5.

Pattern: `^Y|M|W|D|H|30min|15min|10min|5min|1min$`

Required: No

ForecastHorizon

The number of time-steps that the model predicts. The forecast horizon is also called the prediction length.

The maximum forecast horizon is the lesser of 500 time-steps or 1/4 of the TARGET_TIME_SERIES dataset length. If you are retraining an existing AutoPredictor, then the

maximum forecast horizon is the lesser of 500 time-steps or 1/3 of the TARGET_TIME_SERIES dataset length.

If you are upgrading to an AutoPredictor or retraining an existing AutoPredictor, you cannot update the forecast horizon parameter. You can meet this requirement by providing longer time-series in the dataset.

Type: Integer

Required: No

ForecastTypes

The forecast types used to train a predictor. You can specify up to five forecast types. Forecast types can be quantiles from 0.01 to 0.99, by increments of 0.01 or higher. You can also specify the mean forecast with `mean`.

Type: Array of strings

Array Members: Minimum number of 1 item. Maximum number of 20 items.

Length Constraints: Minimum length of 2. Maximum length of 4.

Pattern: `(^0?\.\d\d?$|^mean$)`

Required: No

MonitorConfig

The configuration details for predictor monitoring. Provide a name for the monitor resource to enable predictor monitoring.

Predictor monitoring allows you to see how your predictor's performance changes over time. For more information, see Predictor Monitoring.

Type: MonitorConfig object

Required: No

OptimizationMetric

The accuracy metric used to optimize the predictor.

Type: String

Valid Values: `WAPE | RMSE | AverageWeightedQuantileLoss | MASE | MAPE`

Required: No

PredictorName

A unique name for the predictor

Type: String

Length Constraints: Minimum length of 1. Maximum length of 63.

Pattern: `^[a-zA-Z][a-zA-Z0-9_]*`

Required: Yes

ReferencePredictorArn

The ARN of the predictor to retrain or upgrade. This parameter is only used when retraining or upgrading a predictor. When creating a new predictor, do not specify a value for this parameter.

When upgrading or retraining a predictor, only specify values for the `ReferencePredictorArn` and `PredictorName`. The value for `PredictorName` must be a unique predictor name.

Type: String

Length Constraints: Maximum length of 256.

Pattern: `arn:([a-z\d-]+):forecast:.*:.*:.+`

Required: No

Tags

Optional metadata to help you categorize and organize your predictors. Each tag consists of a key and an optional value, both of which you define. Tag keys and values are case sensitive.

The following restrictions apply to tags:

- For each resource, each tag key must be unique and each tag key must have one value.
- Maximum number of tags per resource: 50.
- Maximum key length: 128 Unicode characters in UTF-8.
- Maximum value length: 256 Unicode characters in UTF-8.

- Accepted characters: all letters and numbers, spaces representable in UTF-8, and + - = . _ : / @. If your tagging schema is used across other services and resources, the character restrictions of those services also apply.
- Key prefixes cannot include any upper or lowercase combination of `aws:` or `AWS:`. Values can have this prefix. If a tag value has `aws` as its prefix but the key does not, Forecast considers it to be a user tag and will count against the limit of 50 tags. Tags with only the key prefix of `aws` do not count against your tags per resource limit. You cannot edit or delete tag keys with this prefix.

Type: Array of Tag objects

Array Members: Minimum number of 0 items. Maximum number of 200 items.

Required: No

TimeAlignmentBoundary

The time boundary Forecast uses to align and aggregate any data that doesn't align with your forecast frequency. Provide the unit of time and the time boundary as a key value pair. For more information on specifying a time boundary, see Specifying a Time Boundary. If you don't provide a time boundary, Forecast uses a set of Default Time Boundaries.

Type: TimeAlignmentBoundary object

Required: No

Response Syntax

```
{
   "PredictorArn": "string"
}
```

Response Elements

If the action is successful, the service sends back an HTTP 200 response.

The following data is returned in JSON format by the service.

PredictorArn

The Amazon Resource Name (ARN) of the predictor.

Type: String

Length Constraints: Maximum length of 256.

Pattern: `arn:([a-z\d-]+):forecast:.*:.*:.+`

Errors

InvalidInputException

We can't process the request because it includes an invalid value or a value that exceeds the valid range.

HTTP Status Code: 400

LimitExceededException

The limit on the number of resources per account has been exceeded.

HTTP Status Code: 400

ResourceAlreadyExistsException

There is already a resource with this name. Try again with a different name.

HTTP Status Code: 400

ResourceInUseException

The specified resource is in use.

HTTP Status Code: 400

ResourceNotFoundException

We can't find a resource with that Amazon Resource Name (ARN). Check the ARN and try again.

HTTP Status Code: 400

See Also

For more information about using this API in one of the language-specific AWS SDKs, see the following:

- AWS Command Line Interface

- AWS SDK for .NET
- AWS SDK for C++
- AWS SDK for Go v2
- AWS SDK for Java V2
- AWS SDK for JavaScript V3
- AWS SDK for PHP V3
- AWS SDK for Python
- AWS SDK for Ruby V3

CreateDataset

Service: Amazon Forecast Service

Creates an Amazon Forecast dataset. The information about the dataset that you provide helps Forecast understand how to consume the data for model training. This includes the following:

- *DataFrequency* - How frequently your historical time-series data is collected.
- *Domain* and *DatasetType* - Each dataset has an associated dataset domain and a type within the domain. Amazon Forecast provides a list of predefined domains and types within each domain. For each unique dataset domain and type within the domain, Amazon Forecast requires your data to include a minimum set of predefined fields.
- *Schema* - A schema specifies the fields in the dataset, including the field name and data type.

After creating a dataset, you import your training data into it and add the dataset to a dataset group. You use the dataset group to create a predictor. For more information, see Importing datasets.

To get a list of all your datasets, use the ListDatasets operation.

For example Forecast datasets, see the Amazon Forecast Sample GitHub repository.

> **Note**
>
> The `Status` of a dataset must be `ACTIVE` before you can import training data. Use the DescribeDataset operation to get the status.

Request Syntax

```
{
   "DataFrequency": "string",
   "DatasetName": "string",
   "DatasetType": "string",
   "Domain": "string",
   "EncryptionConfig": {
      "KMSKeyArn": "string",
      "RoleArn": "string"
   },
   "Schema": {
```

```
      "Attributes": [
         {
            "AttributeName": "string",
            "AttributeType": "string"
         }
      ]
   },
   "Tags": [
      {
         "Key": "string",
         "Value": "string"
      }
   ]
}
```

Request Parameters

The request accepts the following data in JSON format.

DataFrequency

The frequency of data collection. This parameter is required for RELATED_TIME_SERIES datasets.

Valid intervals are an integer followed by Y (Year), M (Month), W (Week), D (Day), H (Hour), and min (Minute). For example, "1D" indicates every day and "15min" indicates every 15 minutes. You cannot specify a value that would overlap with the next larger frequency. That means, for example, you cannot specify a frequency of 60 minutes, because that is equivalent to 1 hour. The valid values for each frequency are the following:

- Minute - 1-59
- Hour - 1-23
- Day - 1-6
- Week - 1-4
- Month - 1-11
- Year - 1

Thus, if you want every other week forecasts, specify "2W". Or, if you want quarterly forecasts, you specify "3M".

Type: String

Length Constraints: Minimum length of 1. Maximum length of 5.

Pattern: `^Y|M|W|D|H|30min|15min|10min|5min|1min$`

Required: No

DatasetName

A name for the dataset.

Type: String

Length Constraints: Minimum length of 1. Maximum length of 63.

Pattern: `^[a-zA-Z][a-zA-Z0-9_]*`

Required: Yes

DatasetType

The dataset type. Valid values depend on the chosen `Domain`.

Type: String

Valid Values: `TARGET_TIME_SERIES | RELATED_TIME_SERIES | ITEM_METADATA`

Required: Yes

Domain

The domain associated with the dataset. When you add a dataset to a dataset group, this value and the value specified for the `Domain` parameter of the CreateDatasetGroup operation must match.

The `Domain` and `DatasetType` that you choose determine the fields that must be present in the training data that you import to the dataset. For example, if you choose the `RETAIL` domain and `TARGET_TIME_SERIES` as the `DatasetType`, Amazon Forecast requires `item_id`, `timestamp`, and `demand` fields to be present in your data. For more information, see Importing datasets.

Type: String

Valid Values: `RETAIL | CUSTOM | INVENTORY_PLANNING | EC2_CAPACITY | WORK_FORCE | WEB_TRAFFIC | METRICS`

Required: Yes

EncryptionConfig

An AWS Key Management Service (KMS) key and the AWS Identity and Access Management (IAM) role that Amazon Forecast can assume to access the key.

Type: EncryptionConfig object

Required: No

Schema

The schema for the dataset. The schema attributes and their order must match the fields in your data. The dataset `Domain` and `DatasetType` that you choose determine the minimum required fields in your training data. For information about the required fields for a specific dataset domain and type, see Dataset Domains and Dataset Types.

Type: Schema object

Required: Yes

Tags

The optional metadata that you apply to the dataset to help you categorize and organize them. Each tag consists of a key and an optional value, both of which you define.

The following basic restrictions apply to tags:

- Maximum number of tags per resource - 50.
- For each resource, each tag key must be unique, and each tag key can have only one value.
- Maximum key length - 128 Unicode characters in UTF-8.
- Maximum value length - 256 Unicode characters in UTF-8.
- If your tagging schema is used across multiple services and resources, remember that other services may have restrictions on allowed characters. Generally allowed characters are: letters, numbers, and spaces representable in UTF-8, and the following characters: + - = . _ : / @.
- Tag keys and values are case sensitive.
- Do not use `aws:`, `AWS:`, or any upper or lowercase combination of such as a prefix for keys as it is reserved for AWS use. You cannot edit or delete tag keys with this prefix. Values can have this prefix. If a tag value has `aws` as its prefix but the key does not, then Forecast considers it to be a user tag and will count against the limit of 50 tags. Tags with only the key prefix of `aws` do not count against your tags per resource limit.

Type: Array of Tag objects

Array Members: Minimum number of 0 items. Maximum number of 200 items.

Required: No

Response Syntax

```
{
   "DatasetArn": "string"
}
```

Response Elements

If the action is successful, the service sends back an HTTP 200 response.

The following data is returned in JSON format by the service.

DatasetArn

The Amazon Resource Name (ARN) of the dataset.

Type: String

Length Constraints: Maximum length of 256.

Pattern: `arn:([a-z\d-]+):forecast:.*:.*:.+`

Errors

InvalidInputException

We can't process the request because it includes an invalid value or a value that exceeds the valid range.

HTTP Status Code: 400

LimitExceededException

The limit on the number of resources per account has been exceeded.

HTTP Status Code: 400

ResourceAlreadyExistsException

There is already a resource with this name. Try again with a different name.

HTTP Status Code: 400

See Also

For more information about using this API in one of the language-specific AWS SDKs, see the following:

- AWS Command Line Interface
- AWS SDK for .NET
- AWS SDK for C++
- AWS SDK for Go v2
- AWS SDK for Java V2
- AWS SDK for JavaScript V3
- AWS SDK for PHP V3
- AWS SDK for Python
- AWS SDK for Ruby V3

CreateDatasetGroup

Service: Amazon Forecast Service

Creates a dataset group, which holds a collection of related datasets. You can add datasets to the dataset group when you create the dataset group, or later by using the UpdateDatasetGroup operation.

After creating a dataset group and adding datasets, you use the dataset group when you create a predictor. For more information, see Dataset groups.

To get a list of all your datasets groups, use the ListDatasetGroups operation.

> **Note**
>
> The `Status` of a dataset group must be `ACTIVE` before you can use the dataset group to create a predictor. To get the status, use the DescribeDatasetGroup operation.

Request Syntax

```
{
   "DatasetArns": [ "string" ],
   "DatasetGroupName": "string",
   "Domain": "string",
   "Tags": [
      {
         "Key": "string",
         "Value": "string"
      }
   ]
}
```

Request Parameters

The request accepts the following data in JSON format.

DatasetArns

An array of Amazon Resource Names (ARNs) of the datasets that you want to include in the dataset group.

Type: Array of strings

Length Constraints: Maximum length of 256.

Pattern: `arn:([a-z\d-]+):forecast:.*:.*:.+`

Required: No

DatasetGroupName

A name for the dataset group.

Type: String

Length Constraints: Minimum length of 1. Maximum length of 63.

Pattern: `^[a-zA-Z][a-zA-Z0-9_]*`

Required: Yes

Domain

The domain associated with the dataset group. When you add a dataset to a dataset group, this value and the value specified for the `Domain` parameter of the CreateDataset operation must match.

The `Domain` and `DatasetType` that you choose determine the fields that must be present in training data that you import to a dataset. For example, if you choose the `RETAIL` domain and `TARGET_TIME_SERIES` as the `DatasetType`, Amazon Forecast requires that `item_id`, `timestamp`, and `demand` fields are present in your data. For more information, see Dataset groups.

Type: String

Valid Values: `RETAIL | CUSTOM | INVENTORY_PLANNING | EC2_CAPACITY | WORK_FORCE | WEB_TRAFFIC | METRICS`

Required: Yes

Tags

The optional metadata that you apply to the dataset group to help you categorize and organize them. Each tag consists of a key and an optional value, both of which you define.

The following basic restrictions apply to tags:

- Maximum number of tags per resource - 50.
- For each resource, each tag key must be unique, and each tag key can have only one value.
- Maximum key length - 128 Unicode characters in UTF-8.
- Maximum value length - 256 Unicode characters in UTF-8.
- If your tagging schema is used across multiple services and resources, remember that other services may have restrictions on allowed characters. Generally allowed characters are: letters, numbers, and spaces representable in UTF-8, and the following characters: + - = . _ : / @.
- Tag keys and values are case sensitive.
- Do not use `aws:`, `AWS:`, or any upper or lowercase combination of such as a prefix for keys as it is reserved for AWS use. You cannot edit or delete tag keys with this prefix. Values can have this prefix. If a tag value has `aws` as its prefix but the key does not, then Forecast considers it to be a user tag and will count against the limit of 50 tags. Tags with only the key prefix of `aws` do not count against your tags per resource limit.

Type: Array of Tag objects

Array Members: Minimum number of 0 items. Maximum number of 200 items.

Required: No

Response Syntax

```
{
   "DatasetGroupArn": "string"
}
```

Response Elements

If the action is successful, the service sends back an HTTP 200 response.

The following data is returned in JSON format by the service.

DatasetGroupArn

The Amazon Resource Name (ARN) of the dataset group.

Type: String

Length Constraints: Maximum length of 256.

Pattern: `arn:([a-z\d-]+):forecast:.*:.*:.+`

Errors

InvalidInputException

We can't process the request because it includes an invalid value or a value that exceeds the valid range.

HTTP Status Code: 400

LimitExceededException

The limit on the number of resources per account has been exceeded.

HTTP Status Code: 400

ResourceAlreadyExistsException

There is already a resource with this name. Try again with a different name.

HTTP Status Code: 400

ResourceInUseException

The specified resource is in use.

HTTP Status Code: 400

ResourceNotFoundException

We can't find a resource with that Amazon Resource Name (ARN). Check the ARN and try again.

HTTP Status Code: 400

See Also

For more information about using this API in one of the language-specific AWS SDKs, see the following:

- AWS Command Line Interface
- AWS SDK for .NET
- AWS SDK for C++

- AWS SDK for Go v2
- AWS SDK for Java V2
- AWS SDK for JavaScript V3
- AWS SDK for PHP V3
- AWS SDK for Python
- AWS SDK for Ruby V3

CreateDatasetImportJob

Service: Amazon Forecast Service

Imports your training data to an Amazon Forecast dataset. You provide the location of your training data in an Amazon Simple Storage Service (Amazon S3) bucket and the Amazon Resource Name (ARN) of the dataset that you want to import the data to.

You must specify a DataSource object that includes an AWS Identity and Access Management (IAM) role that Amazon Forecast can assume to access the data, as Amazon Forecast makes a copy of your data and processes it in an internal AWS system. For more information, see Set up permissions.

The training data must be in CSV or Parquet format. The delimiter must be a comma (,).

You can specify the path to a specific file, the S3 bucket, or to a folder in the S3 bucket. For the latter two cases, Amazon Forecast imports all files up to the limit of 10,000 files.

Because dataset imports are not aggregated, your most recent dataset import is the one that is used when training a predictor or generating a forecast. Make sure that your most recent dataset import contains all of the data you want to model off of, and not just the new data collected since the previous import.

To get a list of all your dataset import jobs, filtered by specified criteria, use the ListDatasetImportJobs operation.

Request Syntax

```
{
   "DatasetArn": "string",
   "DatasetImportJobName": "string",
   "DataSource": {
      "S3Config": {
         "KMSKeyArn": "string",
         "Path": "string",
         "RoleArn": "string"
      }
   },
   "Format": "string",
   "GeolocationFormat": "string",
   "ImportMode": "string",
   "Tags": [
      {
```

```
            "Key": "string",
            "Value": "string"
        }
    ],
    "TimestampFormat": "string",
    "TimeZone": "string",
    "UseGeolocationForTimeZone": boolean
}
```

Request Parameters

The request accepts the following data in JSON format.

DatasetArn

The Amazon Resource Name (ARN) of the Amazon Forecast dataset that you want to import data to.

Type: String

Length Constraints: Maximum length of 256.

Pattern: `arn:([a-z\d-]+):forecast:.*:.*:.+`

Required: Yes

DatasetImportJobName

The name for the dataset import job. We recommend including the current timestamp in the name, for example, `20190721DatasetImport`. This can help you avoid getting a `ResourceAlreadyExistsException` exception.

Type: String

Length Constraints: Minimum length of 1. Maximum length of 63.

Pattern: `^[a-zA-Z][a-zA-Z0-9_]*`

Required: Yes

DataSource

The location of the training data to import and an AWS Identity and Access Management (IAM) role that Amazon Forecast can assume to access the data. The training data must be stored in an Amazon S3 bucket.

If encryption is used, `DataSource` must include an AWS Key Management Service (KMS) key and the IAM role must allow Amazon Forecast permission to access the key. The KMS key and IAM role must match those specified in the `EncryptionConfig` parameter of the CreateDataset operation.

Type: DataSource object

Required: Yes

Format

The format of the imported data, CSV or PARQUET. The default value is CSV.

Type: String

Length Constraints: Maximum length of 7.

Pattern: `^CSV|PARQUET$`

Required: No

GeolocationFormat

The format of the geolocation attribute. The geolocation attribute can be formatted in one of two ways:

- `LAT_LONG` - the latitude and longitude in decimal format (Example: 47.61_-122.33).
- `CC_POSTALCODE` (US Only) - the country code (US), followed by the 5-digit ZIP code (Example: US_98121).

Type: String

Length Constraints: Maximum length of 256.

Pattern: `^[a-zA-Z0-9_]+$`

Required: No

ImportMode

Specifies whether the dataset import job is a `FULL` or `INCREMENTAL` import. A `FULL` dataset import replaces all of the existing data with the newly imported data. An `INCREMENTAL` import appends the imported data to the existing data.

Type: String

Valid Values: `FULL | INCREMENTAL`

Required: No

Tags

The optional metadata that you apply to the dataset import job to help you categorize and organize them. Each tag consists of a key and an optional value, both of which you define.

The following basic restrictions apply to tags:

- Maximum number of tags per resource - 50.
- For each resource, each tag key must be unique, and each tag key can have only one value.
- Maximum key length - 128 Unicode characters in UTF-8.
- Maximum value length - 256 Unicode characters in UTF-8.
- If your tagging schema is used across multiple services and resources, remember that other services may have restrictions on allowed characters. Generally allowed characters are: letters, numbers, and spaces representable in UTF-8, and the following characters: + - = . _ : / @.
- Tag keys and values are case sensitive.
- Do not use `aws:`, `AWS:`, or any upper or lowercase combination of such as a prefix for keys as it is reserved for AWS use. You cannot edit or delete tag keys with this prefix. Values can have this prefix. If a tag value has `aws` as its prefix but the key does not, then Forecast considers it to be a user tag and will count against the limit of 50 tags. Tags with only the key prefix of `aws` do not count against your tags per resource limit.

Type: Array of Tag objects

Array Members: Minimum number of 0 items. Maximum number of 200 items.

Required: No

TimestampFormat

The format of timestamps in the dataset. The format that you specify depends on the `DataFrequency` specified when the dataset was created. The following formats are supported

- "yyyy-MM-dd"

  For the following data frequencies: Y, M, W, and D

- "yyyy-MM-dd HH:mm:ss"

For the following data frequencies: H, 30min, 15min, and 1min; and optionally, for: Y, M, W, and D

If the format isn't specified, Amazon Forecast expects the format to be "yyyy-MM-dd HH:mm:ss".

Type: String

Length Constraints: Maximum length of 256.

Pattern: `^[a-zA-Z0-9\-\:\.\,\'\s]+$`

Required: No

TimeZone

A single time zone for every item in your dataset. This option is ideal for datasets with all timestamps within a single time zone, or if all timestamps are normalized to a single time zone.

Refer to the Joda-Time API for a complete list of valid time zone names.

Type: String

Length Constraints: Maximum length of 256.

Pattern: `^[a-zA-Z0-9\/\+\-\_]+$`

Required: No

UseGeolocationForTimeZone

Automatically derive time zone information from the geolocation attribute. This option is ideal for datasets that contain timestamps in multiple time zones and those timestamps are expressed in local time.

Type: Boolean

Required: No

Response Syntax

```
{
```

```
   "DatasetImportJobArn": "string"
}
```

Response Elements

If the action is successful, the service sends back an HTTP 200 response.

The following data is returned in JSON format by the service.

DatasetImportJobArn

The Amazon Resource Name (ARN) of the dataset import job.

Type: String

Length Constraints: Maximum length of 256.

Pattern: `arn:([a-z\d-]+):forecast:.*:.*:.+`

Errors

InvalidInputException

We can't process the request because it includes an invalid value or a value that exceeds the valid range.

HTTP Status Code: 400

LimitExceededException

The limit on the number of resources per account has been exceeded.

HTTP Status Code: 400

ResourceAlreadyExistsException

There is already a resource with this name. Try again with a different name.

HTTP Status Code: 400

ResourceInUseException

The specified resource is in use.

HTTP Status Code: 400

ResourceNotFoundException

We can't find a resource with that Amazon Resource Name (ARN). Check the ARN and try again.

HTTP Status Code: 400

See Also

For more information about using this API in one of the language-specific AWS SDKs, see the following:

- AWS Command Line Interface
- AWS SDK for .NET
- AWS SDK for C++
- AWS SDK for Go v2
- AWS SDK for Java V2
- AWS SDK for JavaScript V3
- AWS SDK for PHP V3
- AWS SDK for Python
- AWS SDK for Ruby V3

CreateExplainability

Service: Amazon Forecast Service

> **Note**
>
> Explainability is only available for Forecasts and Predictors generated from an AutoPredictor (CreateAutoPredictor)

Creates an Amazon Forecast Explainability.

Explainability helps you better understand how the attributes in your datasets impact forecast. Amazon Forecast uses a metric called Impact scores to quantify the relative impact of each attribute and determine whether they increase or decrease forecast values.

To enable Forecast Explainability, your predictor must include at least one of the following: related time series, item metadata, or additional datasets like Holidays and the Weather Index.

CreateExplainability accepts either a Predictor ARN or Forecast ARN. To receive aggregated Impact scores for all time series and time points in your datasets, provide a Predictor ARN. To receive Impact scores for specific time series and time points, provide a Forecast ARN.

CreateExplainability with a Predictor ARN

> **Note**
>
> You can only have one Explainability resource per predictor. If you already enabled `ExplainPredictor` in CreateAutoPredictor, that predictor already has an Explainability resource.

The following parameters are required when providing a Predictor ARN:

- `ExplainabilityName` - A unique name for the Explainability.
- `ResourceArn` - The Arn of the predictor.
- `TimePointGranularity` - Must be set to “ALL”.
- `TimeSeriesGranularity` - Must be set to “ALL”.

Do not specify a value for the following parameters:

- `DataSource` - Only valid when TimeSeriesGranularity is “SPECIFIC”.
- `Schema` - Only valid when TimeSeriesGranularity is “SPECIFIC”.
- `StartDateTime` - Only valid when TimePointGranularity is “SPECIFIC”.
- `EndDateTime` - Only valid when TimePointGranularity is “SPECIFIC”.

CreateExplainability with a Forecast ARN

> **Note**
>
> You can specify a maximum of 50 time series and 500 time points.

The following parameters are required when providing a Predictor ARN:

- `ExplainabilityName` - A unique name for the Explainability.
- `ResourceArn` - The Arn of the forecast.
- `TimePointGranularity` - Either “ALL” or “SPECIFIC”.
- `TimeSeriesGranularity` - Either “ALL” or “SPECIFIC”.

If you set TimeSeriesGranularity to “SPECIFIC”, you must also provide the following:

- `DataSource` - The S3 location of the CSV file specifying your time series.
- `Schema` - The Schema defines the attributes and attribute types listed in the Data Source.

If you set TimePointGranularity to “SPECIFIC”, you must also provide the following:

- `StartDateTime` - The first timestamp in the range of time points.
- `EndDateTime` - The last timestamp in the range of time points.

Request Syntax

```
{
   "DataSource": {
      "S3Config": {
         "KMSKeyArn": "string",
         "Path": "string",
```

```
         "RoleArn": "string"
      }
   },
   "EnableVisualization": boolean,
   "EndDateTime": "string",
   "ExplainabilityConfig": {
      "TimePointGranularity": "string",
      "TimeSeriesGranularity": "string"
   },
   "ExplainabilityName": "string",
   "ResourceArn": "string",
   "Schema": {
      "Attributes": [
         {
            "AttributeName": "string",
            "AttributeType": "string"
         }
      ]
   },
   "StartDateTime": "string",
   "Tags": [
      {
         "Key": "string",
         "Value": "string"
      }
   ]
}
```

Request Parameters

The request accepts the following data in JSON format.

DataSource

The source of your data, an AWS Identity and Access Management (IAM) role that allows Amazon Forecast to access the data and, optionally, an AWS Key Management Service (KMS) key.

Type: DataSource object

Required: No

EnableVisualization

Create an Explainability visualization that is viewable within the AWS console.

Type: Boolean

Required: No

EndDateTime

If `TimePointGranularity` is set to `SPECIFIC`, define the last time point for the Explainability.

Use the following timestamp format: yyyy-MM-ddTHH:mm:ss (example: 2015-01-01T20:00:00)

Type: String

Length Constraints: Maximum length of 19.

Pattern: `^\d{4}-\d{2}-\d{2}T\d{2}:\d{2}:\d{2}$`

Required: No

ExplainabilityConfig

The configuration settings that define the granularity of time series and time points for the Explainability.

Type: ExplainabilityConfig object

Required: Yes

ExplainabilityName

A unique name for the Explainability.

Type: String

Length Constraints: Minimum length of 1. Maximum length of 63.

Pattern: `^[a-zA-Z][a-zA-Z0-9_]*`

Required: Yes

ResourceArn

The Amazon Resource Name (ARN) of the Predictor or Forecast used to create the Explainability.

Type: String

Length Constraints: Maximum length of 256.

Pattern: `arn:([a-z\d-]+):forecast:.*:.*:.+`

Required: Yes

Schema

Defines the fields of a dataset.

Type: Schema object

Required: No

StartDateTime

If `TimePointGranularity` is set to `SPECIFIC`, define the first point for the Explainability.

Use the following timestamp format: yyyy-MM-ddTHH:mm:ss (example: 2015-01-01T20:00:00)

Type: String

Length Constraints: Maximum length of 19.

Pattern: `^\d{4}-\d{2}-\d{2}T\d{2}:\d{2}:\d{2}$`

Required: No

Tags

Optional metadata to help you categorize and organize your resources. Each tag consists of a key and an optional value, both of which you define. Tag keys and values are case sensitive.

The following restrictions apply to tags:

- For each resource, each tag key must be unique and each tag key must have one value.
- Maximum number of tags per resource: 50.
- Maximum key length: 128 Unicode characters in UTF-8.
- Maximum value length: 256 Unicode characters in UTF-8.
- Accepted characters: all letters and numbers, spaces representable in UTF-8, and + - = . _ : / @. If your tagging schema is used across other services and resources, the character restrictions of those services also apply.

- Key prefixes cannot include any upper or lowercase combination of `aws:` or `AWS:`. Values can have this prefix. If a tag value has `aws` as its prefix but the key does not, Forecast considers it to be a user tag and will count against the limit of 50 tags. Tags with only the key prefix of `aws` do not count against your tags per resource limit. You cannot edit or delete tag keys with this prefix.

Type: Array of Tag objects

Array Members: Minimum number of 0 items. Maximum number of 200 items.

Required: No

Response Syntax

```
{
   "ExplainabilityArn": "string"
}
```

Response Elements

If the action is successful, the service sends back an HTTP 200 response.

The following data is returned in JSON format by the service.

ExplainabilityArn

The Amazon Resource Name (ARN) of the Explainability.

Type: String

Length Constraints: Maximum length of 256.

Pattern: `arn:([a-z\d-]+):forecast:.*:.*:.+`

Errors

InvalidInputException

We can't process the request because it includes an invalid value or a value that exceeds the valid range.

HTTP Status Code: 400

LimitExceededException

The limit on the number of resources per account has been exceeded.

HTTP Status Code: 400

ResourceAlreadyExistsException

There is already a resource with this name. Try again with a different name.

HTTP Status Code: 400

ResourceInUseException

The specified resource is in use.

HTTP Status Code: 400

ResourceNotFoundException

We can't find a resource with that Amazon Resource Name (ARN). Check the ARN and try again.

HTTP Status Code: 400

See Also

For more information about using this API in one of the language-specific AWS SDKs, see the following:

- AWS Command Line Interface
- AWS SDK for .NET
- AWS SDK for C++
- AWS SDK for Go v2
- AWS SDK for Java V2
- AWS SDK for JavaScript V3
- AWS SDK for PHP V3
- AWS SDK for Python
- AWS SDK for Ruby V3

CreateExplainabilityExport

Service: Amazon Forecast Service

Exports an Explainability resource created by the CreateExplainability operation. Exported files are exported to an Amazon Simple Storage Service (Amazon S3) bucket.

You must specify a DataDestination object that includes an Amazon S3 bucket and an AWS Identity and Access Management (IAM) role that Amazon Forecast can assume to access the Amazon S3 bucket. For more information, see Set Up Permissions for Amazon Forecast.

> **Note**
>
> The `Status` of the export job must be `ACTIVE` before you can access the export in your Amazon S3 bucket. To get the status, use the DescribeExplainabilityExport operation.

Request Syntax

```
{
   "Destination": {
      "S3Config": {
         "KMSKeyArn": "string",
         "Path": "string",
         "RoleArn": "string"
      }
   },
   "ExplainabilityArn": "string",
   "ExplainabilityExportName": "string",
   "Format": "string",
   "Tags": [
      {
         "Key": "string",
         "Value": "string"
      }
   ]
}
```

Request Parameters

The request accepts the following data in JSON format.

Destination

The destination for an export job. Provide an S3 path, an AWS Identity and Access Management (IAM) role that allows Amazon Forecast to access the location, and an AWS Key Management Service (KMS) key (optional).

Type: DataDestination object

Required: Yes

ExplainabilityArn

The Amazon Resource Name (ARN) of the Explainability to export.

Type: String

Length Constraints: Maximum length of 256.

Pattern: `arn:([a-z\d-]+):forecast:.*:.*:.+`

Required: Yes

ExplainabilityExportName

A unique name for the Explainability export.

Type: String

Length Constraints: Minimum length of 1. Maximum length of 63.

Pattern: `^[a-zA-Z][a-zA-Z0-9_]*`

Required: Yes

Format

The format of the exported data, CSV or PARQUET.

Type: String

Length Constraints: Maximum length of 7.

Pattern: `^CSV|PARQUET$`

Required: No

Tags

Optional metadata to help you categorize and organize your resources. Each tag consists of a key and an optional value, both of which you define. Tag keys and values are case sensitive.

The following restrictions apply to tags:

- For each resource, each tag key must be unique and each tag key must have one value.
- Maximum number of tags per resource: 50.
- Maximum key length: 128 Unicode characters in UTF-8.
- Maximum value length: 256 Unicode characters in UTF-8.
- Accepted characters: all letters and numbers, spaces representable in UTF-8, and + - = . _ : / @. If your tagging schema is used across other services and resources, the character restrictions of those services also apply.
- Key prefixes cannot include any upper or lowercase combination of `aws:` or `AWS:`. Values can have this prefix. If a tag value has `aws` as its prefix but the key does not, Forecast considers it to be a user tag and will count against the limit of 50 tags. Tags with only the key prefix of `aws` do not count against your tags per resource limit. You cannot edit or delete tag keys with this prefix.

Type: Array of Tag objects

Array Members: Minimum number of 0 items. Maximum number of 200 items.

Required: No

Response Syntax

```
{
   "ExplainabilityExportArn": "string"
}
```

Response Elements

If the action is successful, the service sends back an HTTP 200 response.

The following data is returned in JSON format by the service.

ExplainabilityExportArn

The Amazon Resource Name (ARN) of the export.

Type: String

Length Constraints: Maximum length of 256.

Pattern: `arn:([a-z\d-]+):forecast:.*:.*:.+`

Errors

InvalidInputException

We can't process the request because it includes an invalid value or a value that exceeds the valid range.

HTTP Status Code: 400

LimitExceededException

The limit on the number of resources per account has been exceeded.

HTTP Status Code: 400

ResourceAlreadyExistsException

There is already a resource with this name. Try again with a different name.

HTTP Status Code: 400

ResourceInUseException

The specified resource is in use.

HTTP Status Code: 400

ResourceNotFoundException

We can't find a resource with that Amazon Resource Name (ARN). Check the ARN and try again.

HTTP Status Code: 400

See Also

For more information about using this API in one of the language-specific AWS SDKs, see the following:

- AWS Command Line Interface

- AWS SDK for .NET
- AWS SDK for C++
- AWS SDK for Go v2
- AWS SDK for Java V2
- AWS SDK for JavaScript V3
- AWS SDK for PHP V3
- AWS SDK for Python
- AWS SDK for Ruby V3

CreateForecast

Service: Amazon Forecast Service

Creates a forecast for each item in the `TARGET_TIME_SERIES` dataset that was used to train the predictor. This is known as inference. To retrieve the forecast for a single item at low latency, use the QueryForecast operation. To export the complete forecast into your Amazon Simple Storage Service (Amazon S3) bucket, use the CreateForecastExportJob operation.

The range of the forecast is determined by the `ForecastHorizon` value, which you specify in the CreatePredictor request. When you query a forecast, you can request a specific date range within the forecast.

To get a list of all your forecasts, use the ListForecasts operation.

> **Note**
>
> The forecasts generated by Amazon Forecast are in the same time zone as the dataset that was used to create the predictor.

For more information, see Generating Forecasts.

> **Note**
>
> The `Status` of the forecast must be `ACTIVE` before you can query or export the forecast. Use the DescribeForecast operation to get the status.

By default, a forecast includes predictions for every item (`item_id`) in the dataset group that was used to train the predictor. However, you can use the `TimeSeriesSelector` object to generate a forecast on a subset of time series. Forecast creation is skipped for any time series that you specify that are not in the input dataset. The forecast export file will not contain these time series or their forecasted values.

Request Syntax

```
{
   "ForecastName": "string",
   "ForecastTypes": [ "string" ],
```

```
   "PredictorArn": "string",
   "Tags": [
      {
         "Key": "string",
         "Value": "string"
      }
   ],
   "TimeSeriesSelector": {
      "TimeSeriesIdentifiers": {
         "DataSource": {
            "S3Config": {
               "KMSKeyArn": "string",
               "Path": "string",
               "RoleArn": "string"
            }
         },
         "Format": "string",
         "Schema": {
            "Attributes": [
               {
                  "AttributeName": "string",
                  "AttributeType": "string"
               }
            ]
         }
      }
   }
}
```

Request Parameters

The request accepts the following data in JSON format.

ForecastName

A name for the forecast.

Type: String

Length Constraints: Minimum length of 1. Maximum length of 63.

Pattern: `^[a-zA-Z][a-zA-Z0-9_]*`

Required: Yes

ForecastTypes

The quantiles at which probabilistic forecasts are generated. **You can currently specify up to 5 quantiles per forecast**. Accepted values include `0.01 to 0.99` (increments of .01 only) and `mean`. The mean forecast is different from the median (0.50) when the distribution is not symmetric (for example, Beta and Negative Binomial).

The default quantiles are the quantiles you specified during predictor creation. If you didn't specify quantiles, the default values are `["0.1", "0.5", "0.9"]`.

Type: Array of strings

Array Members: Minimum number of 1 item. Maximum number of 20 items.

Length Constraints: Minimum length of 2. Maximum length of 4.

Pattern: `(^0?\.\d\d?$|^mean$)`

Required: No

PredictorArn

The Amazon Resource Name (ARN) of the predictor to use to generate the forecast.

Type: String

Length Constraints: Maximum length of 256.

Pattern: `arn:([a-z\d-]+):forecast:.*:.*:.+`

Required: Yes

Tags

The optional metadata that you apply to the forecast to help you categorize and organize them. Each tag consists of a key and an optional value, both of which you define.

The following basic restrictions apply to tags:

- Maximum number of tags per resource - 50.
- For each resource, each tag key must be unique, and each tag key can have only one value.
- Maximum key length - 128 Unicode characters in UTF-8.
- Maximum value length - 256 Unicode characters in UTF-8.

- If your tagging schema is used across multiple services and resources, remember that other services may have restrictions on allowed characters. Generally allowed characters are: letters, numbers, and spaces representable in UTF-8, and the following characters: + - = . _ : / @.
- Tag keys and values are case sensitive.
- Do not use `aws:`, `AWS:`, or any upper or lowercase combination of such as a prefix for keys as it is reserved for AWS use. You cannot edit or delete tag keys with this prefix. Values can have this prefix. If a tag value has `aws` as its prefix but the key does not, then Forecast considers it to be a user tag and will count against the limit of 50 tags. Tags with only the key prefix of `aws` do not count against your tags per resource limit.

Type: Array of Tag objects

Array Members: Minimum number of 0 items. Maximum number of 200 items.

Required: No

TimeSeriesSelector

Defines the set of time series that are used to create the forecasts in a `TimeSeriesIdentifiers` object.

The `TimeSeriesIdentifiers` object needs the following information:

- `DataSource`
- `Format`
- `Schema`

Type: TimeSeriesSelector object

Required: No

Response Syntax

```
{
   "ForecastArn": "string"
}
```

Response Elements

If the action is successful, the service sends back an HTTP 200 response.

The following data is returned in JSON format by the service.

ForecastArn

The Amazon Resource Name (ARN) of the forecast.

Type: String

Length Constraints: Maximum length of 256.

Pattern: `arn:([a-z\d-]+):forecast:.*:.*:.+`

Errors

InvalidInputException

We can't process the request because it includes an invalid value or a value that exceeds the valid range.

HTTP Status Code: 400

LimitExceededException

The limit on the number of resources per account has been exceeded.

HTTP Status Code: 400

ResourceAlreadyExistsException

There is already a resource with this name. Try again with a different name.

HTTP Status Code: 400

ResourceInUseException

The specified resource is in use.

HTTP Status Code: 400

ResourceNotFoundException

We can't find a resource with that Amazon Resource Name (ARN). Check the ARN and try again.

HTTP Status Code: 400

See Also

For more information about using this API in one of the language-specific AWS SDKs, see the following:

- AWS Command Line Interface
- AWS SDK for .NET
- AWS SDK for C++
- AWS SDK for Go v2
- AWS SDK for Java V2
- AWS SDK for JavaScript V3
- AWS SDK for PHP V3
- AWS SDK for Python
- AWS SDK for Ruby V3

CreateForecastExportJob

Service: Amazon Forecast Service

Exports a forecast created by the CreateForecast operation to your Amazon Simple Storage Service (Amazon S3) bucket. The forecast file name will match the following conventions:

<ForecastExportJobName>_<ExportTimestamp>_<PartNumber>

where the <ExportTimestamp> component is in Java SimpleDateFormat (yyyy-MM-ddTHH-mm-ssZ).

You must specify a DataDestination object that includes an AWS Identity and Access Management (IAM) role that Amazon Forecast can assume to access the Amazon S3 bucket. For more information, see Set Up Permissions for Amazon Forecast.

For more information, see Generating Forecasts.

To get a list of all your forecast export jobs, use the ListForecastExportJobs operation.

> **Note**
>
> The `Status` of the forecast export job must be `ACTIVE` before you can access the forecast in your Amazon S3 bucket. To get the status, use the DescribeForecastExportJob operation.

Request Syntax

```
{
   "Destination": {
      "S3Config": {
         "KMSKeyArn": "string",
         "Path": "string",
         "RoleArn": "string"
      }
   },
   "ForecastArn": "string",
   "ForecastExportJobName": "string",
   "Format": "string",
   "Tags": [
      {
         "Key": "string",
```

```
            "Value": "string"
        }
    ]
}
```

Request Parameters

The request accepts the following data in JSON format.

Destination

The location where you want to save the forecast and an AWS Identity and Access Management (IAM) role that Amazon Forecast can assume to access the location. The forecast must be exported to an Amazon S3 bucket.

If encryption is used, `Destination` must include an AWS Key Management Service (KMS) key. The IAM role must allow Amazon Forecast permission to access the key.

Type: DataDestination object

Required: Yes

ForecastArn

The Amazon Resource Name (ARN) of the forecast that you want to export.

Type: String

Length Constraints: Maximum length of 256.

Pattern: `arn:([a-z\d-]+):forecast:.*:.*:.+`

Required: Yes

ForecastExportJobName

The name for the forecast export job.

Type: String

Length Constraints: Minimum length of 1. Maximum length of 63.

Pattern: `^[a-zA-Z][a-zA-Z0-9_]*`

Required: Yes

Format

The format of the exported data, CSV or PARQUET. The default value is CSV.

Type: String

Length Constraints: Maximum length of 7.

Pattern: `^CSV|PARQUET$`

Required: No

Tags

The optional metadata that you apply to the forecast export job to help you categorize and organize them. Each tag consists of a key and an optional value, both of which you define.

The following basic restrictions apply to tags:

- Maximum number of tags per resource - 50.
- For each resource, each tag key must be unique, and each tag key can have only one value.
- Maximum key length - 128 Unicode characters in UTF-8.
- Maximum value length - 256 Unicode characters in UTF-8.
- If your tagging schema is used across multiple services and resources, remember that other services may have restrictions on allowed characters. Generally allowed characters are: letters, numbers, and spaces representable in UTF-8, and the following characters: + - = . _ : / @.
- Tag keys and values are case sensitive.
- Do not use `aws:`, `AWS:`, or any upper or lowercase combination of such as a prefix for keys as it is reserved for AWS use. You cannot edit or delete tag keys with this prefix. Values can have this prefix. If a tag value has `aws` as its prefix but the key does not, then Forecast considers it to be a user tag and will count against the limit of 50 tags. Tags with only the key prefix of `aws` do not count against your tags per resource limit.

Type: Array of Tag objects

Array Members: Minimum number of 0 items. Maximum number of 200 items.

Required: No

Response Syntax

```
{
   "ForecastExportJobArn": "string"
}
```

Response Elements

If the action is successful, the service sends back an HTTP 200 response.

The following data is returned in JSON format by the service.

ForecastExportJobArn

The Amazon Resource Name (ARN) of the export job.

Type: String

Length Constraints: Maximum length of 256.

Pattern: `arn:([a-z\d-]+):forecast:.*:.*:.+`

Errors

InvalidInputException

We can't process the request because it includes an invalid value or a value that exceeds the valid range.

HTTP Status Code: 400

LimitExceededException

The limit on the number of resources per account has been exceeded.

HTTP Status Code: 400

ResourceAlreadyExistsException

There is already a resource with this name. Try again with a different name.

HTTP Status Code: 400

ResourceInUseException

The specified resource is in use.

HTTP Status Code: 400

ResourceNotFoundException

We can't find a resource with that Amazon Resource Name (ARN). Check the ARN and try again.

HTTP Status Code: 400

See Also

For more information about using this API in one of the language-specific AWS SDKs, see the following:

- AWS Command Line Interface
- AWS SDK for .NET
- AWS SDK for C++
- AWS SDK for Go v2
- AWS SDK for Java V2
- AWS SDK for JavaScript V3
- AWS SDK for PHP V3
- AWS SDK for Python
- AWS SDK for Ruby V3

CreateMonitor

Service: Amazon Forecast Service

Creates a predictor monitor resource for an existing auto predictor. Predictor monitoring allows you to see how your predictor's performance changes over time. For more information, see Predictor Monitoring.

Request Syntax

```
{
   "MonitorName": "string",
   "ResourceArn": "string",
   "Tags": [
      {
         "Key": "string",
         "Value": "string"
      }
   ]
}
```

Request Parameters

The request accepts the following data in JSON format.

MonitorName

The name of the monitor resource.

Type: String

Length Constraints: Minimum length of 1. Maximum length of 63.

Pattern: `^[a-zA-Z][a-zA-Z0-9_]*`

Required: Yes

ResourceArn

The Amazon Resource Name (ARN) of the predictor to monitor.

Type: String

Length Constraints: Maximum length of 256.

Pattern: `arn:([a-z\d-]+):forecast:.*:.*:.+`

Required: Yes

Tags

A list of tags to apply to the monitor resource.

Type: Array of Tag objects

Array Members: Minimum number of 0 items. Maximum number of 200 items.

Required: No

Response Syntax

```
{
   "MonitorArn": "string"
}
```

Response Elements

If the action is successful, the service sends back an HTTP 200 response.

The following data is returned in JSON format by the service.

MonitorArn

The Amazon Resource Name (ARN) of the monitor resource.

Type: String

Length Constraints: Maximum length of 256.

Pattern: `arn:([a-z\d-]+):forecast:.*:.*:.+`

Errors

InvalidInputException

We can't process the request because it includes an invalid value or a value that exceeds the valid range.

HTTP Status Code: 400

LimitExceededException

The limit on the number of resources per account has been exceeded.

HTTP Status Code: 400

ResourceAlreadyExistsException

There is already a resource with this name. Try again with a different name.

HTTP Status Code: 400

ResourceInUseException

The specified resource is in use.

HTTP Status Code: 400

ResourceNotFoundException

We can't find a resource with that Amazon Resource Name (ARN). Check the ARN and try again.

HTTP Status Code: 400

See Also

For more information about using this API in one of the language-specific AWS SDKs, see the following:

- AWS Command Line Interface
- AWS SDK for .NET
- AWS SDK for C++
- AWS SDK for Go v2
- AWS SDK for Java V2
- AWS SDK for JavaScript V3
- AWS SDK for PHP V3
- AWS SDK for Python
- AWS SDK for Ruby V3

CreatePredictor

Service: Amazon Forecast Service

> **Note**
>
> This operation creates a legacy predictor that does not include all the predictor functionalities provided by Amazon Forecast. To create a predictor that is compatible with all aspects of Forecast, use CreateAutoPredictor.

Creates an Amazon Forecast predictor.

In the request, provide a dataset group and either specify an algorithm or let Amazon Forecast choose an algorithm for you using AutoML. If you specify an algorithm, you also can override algorithm-specific hyperparameters.

Amazon Forecast uses the algorithm to train a predictor using the latest version of the datasets in the specified dataset group. You can then generate a forecast using the CreateForecast operation.

To see the evaluation metrics, use the GetAccuracyMetrics operation.

You can specify a featurization configuration to fill and aggregate the data fields in the `TARGET_TIME_SERIES` dataset to improve model training. For more information, see FeaturizationConfig.

For RELATED_TIME_SERIES datasets, `CreatePredictor` verifies that the `DataFrequency` specified when the dataset was created matches the `ForecastFrequency`. TARGET_TIME_SERIES datasets don't have this restriction. Amazon Forecast also verifies the delimiter and timestamp format. For more information, see Importing Datasets.

By default, predictors are trained and evaluated at the 0.1 (P10), 0.5 (P50), and 0.9 (P90) quantiles. You can choose custom forecast types to train and evaluate your predictor by setting the `ForecastTypes`.

AutoML

If you want Amazon Forecast to evaluate each algorithm and choose the one that minimizes the `objective function`, set `PerformAutoML` to `true`. The `objective function` is defined as the mean of the weighted losses over the forecast types. By default, these are the p10, p50, and p90 quantile losses. For more information, see EvaluationResult.

When AutoML is enabled, the following properties are disallowed:

- `AlgorithmArn`
- `HPOConfig`
- `PerformHPO`
- `TrainingParameters`

To get a list of all of your predictors, use the ListPredictors operation.

> **Note**
>
> Before you can use the predictor to create a forecast, the `Status` of the predictor must be `ACTIVE`, signifying that training has completed. To get the status, use the DescribePredictor operation.

Request Syntax

```
{
   "AlgorithmArn": "string",
   "AutoMLOverrideStrategy": "string",
   "EncryptionConfig": {
      "KMSKeyArn": "string",
      "RoleArn": "string"
   },
   "EvaluationParameters": {
      "BackTestWindowOffset": number,
      "NumberOfBacktestWindows": number
   },
   "FeaturizationConfig": {
      "Featurizations": [
         {
            "AttributeName": "string",
            "FeaturizationPipeline": [
               {
                  "FeaturizationMethodName": "string",
                  "FeaturizationMethodParameters": {
                     "string" : "string"
                  }
               }
```

```
            ]
         }
      ],
      "ForecastDimensions": [ "string" ],
      "ForecastFrequency": "string"
   },
   "ForecastHorizon": number,
   "ForecastTypes": [ "string" ],
   "HPOConfig": {
      "ParameterRanges": {
         "CategoricalParameterRanges": [
            {
               "Name": "string",
               "Values": [ "string" ]
            }
         ],
         "ContinuousParameterRanges": [
            {
               "MaxValue": number,
               "MinValue": number,
               "Name": "string",
               "ScalingType": "string"
            }
         ],
         "IntegerParameterRanges": [
            {
               "MaxValue": number,
               "MinValue": number,
               "Name": "string",
               "ScalingType": "string"
            }
         ]
      }
   },
   "InputDataConfig": {
      "DatasetGroupArn": "string",
      "SupplementaryFeatures": [
         {
            "Name": "string",
            "Value": "string"
         }
      ]
   },
   "OptimizationMetric": "string",
```

```
   "PerformAutoML": boolean,
   "PerformHPO": boolean,
   "PredictorName": "string",
   "Tags": [
      {
         "Key": "string",
         "Value": "string"
      }
   ],
   "TrainingParameters": {
      "string" : "string"
   }
}
```

Request Parameters

The request accepts the following data in JSON format.

AlgorithmArn

The Amazon Resource Name (ARN) of the algorithm to use for model training. Required if `PerformAutoML` is not set to `true`.

Supported algorithms:

- `arn:aws:forecast:::algorithm/ARIMA`
- `arn:aws:forecast:::algorithm/CNN-QR`
- `arn:aws:forecast:::algorithm/Deep_AR_Plus`
- `arn:aws:forecast:::algorithm/ETS`
- `arn:aws:forecast:::algorithm/NPTS`
- `arn:aws:forecast:::algorithm/Prophet`

Type: String

Length Constraints: Maximum length of 256.

Pattern: `arn:([a-z\d-]+):forecast:.*:.*:.+`

Required: No

AutoMLOverrideStrategy

> **Note**
>
> The `LatencyOptimized` AutoML override strategy is only available in private beta. Contact AWS Support or your account manager to learn more about access privileges.

Used to overide the default AutoML strategy, which is to optimize predictor accuracy. To apply an AutoML strategy that minimizes training time, use `LatencyOptimized`.

This parameter is only valid for predictors trained using AutoML.

Type: String

Valid Values: `LatencyOptimized | AccuracyOptimized`

Required: No

EncryptionConfig

An AWS Key Management Service (KMS) key and the AWS Identity and Access Management (IAM) role that Amazon Forecast can assume to access the key.

Type: EncryptionConfig object

Required: No

EvaluationParameters

Used to override the default evaluation parameters of the specified algorithm. Amazon Forecast evaluates a predictor by splitting a dataset into training data and testing data. The evaluation parameters define how to perform the split and the number of iterations.

Type: EvaluationParameters object

Required: No

FeaturizationConfig

The featurization configuration.

Type: FeaturizationConfig object

Required: Yes

ForecastHorizon

Specifies the number of time-steps that the model is trained to predict. The forecast horizon is also called the prediction length.

For example, if you configure a dataset for daily data collection (using the `DataFrequency` parameter of the CreateDataset operation) and set the forecast horizon to 10, the model returns predictions for 10 days.

The maximum forecast horizon is the lesser of 500 time-steps or 1/3 of the TARGET_TIME_SERIES dataset length.

Type: Integer

Required: Yes

ForecastTypes

Specifies the forecast types used to train a predictor. You can specify up to five forecast types. Forecast types can be quantiles from 0.01 to 0.99, by increments of 0.01 or higher. You can also specify the mean forecast with `mean`.

The default value is `["0.10", "0.50", "0.9"]`.

Type: Array of strings

Array Members: Minimum number of 1 item. Maximum number of 20 items.

Length Constraints: Minimum length of 2. Maximum length of 4.

Pattern: `(^0?\.\d\d?$|^mean$)`

Required: No

HPOConfig

Provides hyperparameter override values for the algorithm. If you don't provide this parameter, Amazon Forecast uses default values. The individual algorithms specify which hyperparameters support hyperparameter optimization (HPO). For more information, see Amazon Forecast Algorithms.

If you included the `HPOConfig` object, you must set `PerformHPO` to true.

Type: HyperParameterTuningJobConfig object

Required: No

InputDataConfig

Describes the dataset group that contains the data to use to train the predictor.

Type: InputDataConfig object

Required: Yes

OptimizationMetric

The accuracy metric used to optimize the predictor. The default value is `AverageWeightedQuantileLoss`.

Type: String

Valid Values: `WAPE | RMSE | AverageWeightedQuantileLoss | MASE | MAPE`

Required: No

PerformAutoML

Whether to perform AutoML. When Amazon Forecast performs AutoML, it evaluates the algorithms it provides and chooses the best algorithm and configuration for your training dataset.

The default value is `false`. In this case, you are required to specify an algorithm.

Set `PerformAutoML` to `true` to have Amazon Forecast perform AutoML. This is a good option if you aren't sure which algorithm is suitable for your training data. In this case, `PerformHPO` must be false.

Type: Boolean

Required: No

PerformHPO

Whether to perform hyperparameter optimization (HPO). HPO finds optimal hyperparameter values for your training data. The process of performing HPO is known as running a hyperparameter tuning job.

The default value is `false`. In this case, Amazon Forecast uses default hyperparameter values from the chosen algorithm.

To override the default values, set `PerformHPO` to `true` and, optionally, supply the HyperParameterTuningJobConfig object. The tuning job specifies a metric to optimize, which hyperparameters participate in tuning, and the valid range for each tunable hyperparameter. In this case, you are required to specify an algorithm and `PerformAutoML` must be false.

The following algorithms support HPO:

- DeepAR+
- CNN-QR

Type: Boolean

Required: No

PredictorName

A name for the predictor.

Type: String

Length Constraints: Minimum length of 1. Maximum length of 63.

Pattern: `^[a-zA-Z][a-zA-Z0-9_]*`

Required: Yes

Tags

The optional metadata that you apply to the predictor to help you categorize and organize them. Each tag consists of a key and an optional value, both of which you define.

The following basic restrictions apply to tags:

- Maximum number of tags per resource - 50.
- For each resource, each tag key must be unique, and each tag key can have only one value.
- Maximum key length - 128 Unicode characters in UTF-8.
- Maximum value length - 256 Unicode characters in UTF-8.
- If your tagging schema is used across multiple services and resources, remember that other services may have restrictions on allowed characters. Generally allowed characters are: letters, numbers, and spaces representable in UTF-8, and the following characters: + - = . _ : / @.

- Tag keys and values are case sensitive.
- Do not use `aws:`, `AWS:`, or any upper or lowercase combination of such as a prefix for keys as it is reserved for AWS use. You cannot edit or delete tag keys with this prefix. Values can have this prefix. If a tag value has `aws` as its prefix but the key does not, then Forecast considers it to be a user tag and will count against the limit of 50 tags. Tags with only the key prefix of `aws` do not count against your tags per resource limit.

Type: Array of Tag objects

Array Members: Minimum number of 0 items. Maximum number of 200 items.

Required: No

TrainingParameters

The hyperparameters to override for model training. The hyperparameters that you can override are listed in the individual algorithms. For the list of supported algorithms, see Amazon Forecast Algorithms.

Type: String to string map

Map Entries: Minimum number of 0 items. Maximum number of 100 items.

Key Length Constraints: Maximum length of 256.

Key Pattern: `^[a-zA-Z0-9\-\_\.\/\[\]\,\\]+$`

Value Length Constraints: Maximum length of 256.

Value Pattern: `^[a-zA-Z0-9\-\_\.\/\[\]\,\"\\\s]+$`

Required: No

Response Syntax

```
{
   "PredictorArn": "string"
}
```

Response Elements

If the action is successful, the service sends back an HTTP 200 response.

The following data is returned in JSON format by the service.

PredictorArn

The Amazon Resource Name (ARN) of the predictor.

Type: String

Length Constraints: Maximum length of 256.

Pattern: `arn:([a-z\d-]+):forecast:.*:.*:.+`

Errors

InvalidInputException

We can't process the request because it includes an invalid value or a value that exceeds the valid range.

HTTP Status Code: 400

LimitExceededException

The limit on the number of resources per account has been exceeded.

HTTP Status Code: 400

ResourceAlreadyExistsException

There is already a resource with this name. Try again with a different name.

HTTP Status Code: 400

ResourceInUseException

The specified resource is in use.

HTTP Status Code: 400

ResourceNotFoundException

We can't find a resource with that Amazon Resource Name (ARN). Check the ARN and try again.

HTTP Status Code: 400

See Also

For more information about using this API in one of the language-specific AWS SDKs, see the following:

- AWS Command Line Interface
- AWS SDK for .NET
- AWS SDK for C++
- AWS SDK for Go v2
- AWS SDK for Java V2
- AWS SDK for JavaScript V3
- AWS SDK for PHP V3
- AWS SDK for Python
- AWS SDK for Ruby V3

CreatePredictorBacktestExportJob

Service: Amazon Forecast Service

Exports backtest forecasts and accuracy metrics generated by the CreateAutoPredictor or CreatePredictor operations. Two folders containing CSV or Parquet files are exported to your specified S3 bucket.

The export file names will match the following conventions:

`<ExportJobName>_<ExportTimestamp>_<PartNumber>.csv`

The <ExportTimestamp> component is in Java SimpleDate format (yyyy-MM-ddTHH-mm-ssZ).

You must specify a DataDestination object that includes an Amazon S3 bucket and an AWS Identity and Access Management (IAM) role that Amazon Forecast can assume to access the Amazon S3 bucket. For more information, see Set Up Permissions for Amazon Forecast.

> **Note**
>
> The `Status` of the export job must be `ACTIVE` before you can access the export in your Amazon S3 bucket. To get the status, use the DescribePredictorBacktestExportJob operation.

Request Syntax

```
{
   "Destination": {
      "S3Config": {
         "KMSKeyArn": "string",
         "Path": "string",
         "RoleArn": "string"
      }
   },
   "Format": "string",
   "PredictorArn": "string",
   "PredictorBacktestExportJobName": "string",
   "Tags": [
      {
         "Key": "string",
         "Value": "string"
```

```
        }
    ]
}
```

Request Parameters

The request accepts the following data in JSON format.

Destination

The destination for an export job. Provide an S3 path, an AWS Identity and Access Management (IAM) role that allows Amazon Forecast to access the location, and an AWS Key Management Service (KMS) key (optional).

Type: DataDestination object

Required: Yes

Format

The format of the exported data, CSV or PARQUET. The default value is CSV.

Type: String

Length Constraints: Maximum length of 7.

Pattern: `^CSV|PARQUET$`

Required: No

PredictorArn

The Amazon Resource Name (ARN) of the predictor that you want to export.

Type: String

Length Constraints: Maximum length of 256.

Pattern: `arn:([a-z\d-]+):forecast:.*:.*:.+`

Required: Yes

PredictorBacktestExportJobName

The name for the backtest export job.

Type: String

Length Constraints: Minimum length of 1. Maximum length of 63.

Pattern: `^[a-zA-Z][a-zA-Z0-9_]*`

Required: Yes

Tags

Optional metadata to help you categorize and organize your backtests. Each tag consists of a key and an optional value, both of which you define. Tag keys and values are case sensitive.

The following restrictions apply to tags:

- For each resource, each tag key must be unique and each tag key must have one value.
- Maximum number of tags per resource: 50.
- Maximum key length: 128 Unicode characters in UTF-8.
- Maximum value length: 256 Unicode characters in UTF-8.
- Accepted characters: all letters and numbers, spaces representable in UTF-8, and + - = . _ : / @. If your tagging schema is used across other services and resources, the character restrictions of those services also apply.
- Key prefixes cannot include any upper or lowercase combination of `aws:` or `AWS:`. Values can have this prefix. If a tag value has `aws` as its prefix but the key does not, Forecast considers it to be a user tag and will count against the limit of 50 tags. Tags with only the key prefix of `aws` do not count against your tags per resource limit. You cannot edit or delete tag keys with this prefix.

Type: Array of Tag objects

Array Members: Minimum number of 0 items. Maximum number of 200 items.

Required: No

Response Syntax

```
{
   "PredictorBacktestExportJobArn": "string"
}
```

Response Elements

If the action is successful, the service sends back an HTTP 200 response.

The following data is returned in JSON format by the service.

PredictorBacktestExportJobArn

The Amazon Resource Name (ARN) of the predictor backtest export job that you want to export.

Type: String

Length Constraints: Maximum length of 256.

Pattern: `arn:([a-z\d-]+):forecast:.*:.*:.+`

Errors

InvalidInputException

We can't process the request because it includes an invalid value or a value that exceeds the valid range.

HTTP Status Code: 400

LimitExceededException

The limit on the number of resources per account has been exceeded.

HTTP Status Code: 400

ResourceAlreadyExistsException

There is already a resource with this name. Try again with a different name.

HTTP Status Code: 400

ResourceInUseException

The specified resource is in use.

HTTP Status Code: 400

ResourceNotFoundException

We can't find a resource with that Amazon Resource Name (ARN). Check the ARN and try again.

HTTP Status Code: 400

See Also

For more information about using this API in one of the language-specific AWS SDKs, see the following:

- AWS Command Line Interface
- AWS SDK for .NET
- AWS SDK for C++
- AWS SDK for Go v2
- AWS SDK for Java V2
- AWS SDK for JavaScript V3
- AWS SDK for PHP V3
- AWS SDK for Python
- AWS SDK for Ruby V3

CreateWhatIfAnalysis

Service: Amazon Forecast Service

What-if analysis is a scenario modeling technique where you make a hypothetical change to a time series and compare the forecasts generated by these changes against the baseline, unchanged time series. It is important to remember that the purpose of a what-if analysis is to understand how a forecast can change given different modifications to the baseline time series.

For example, imagine you are a clothing retailer who is considering an end of season sale to clear space for new styles. After creating a baseline forecast, you can use a what-if analysis to investigate how different sales tactics might affect your goals.

You could create a scenario where everything is given a 25% markdown, and another where everything is given a fixed dollar markdown. You could create a scenario where the sale lasts for one week and another where the sale lasts for one month. With a what-if analysis, you can compare many different scenarios against each other.

Note that a what-if analysis is meant to display what the forecasting model has learned and how it will behave in the scenarios that you are evaluating. Do not blindly use the results of the what-if analysis to make business decisions. For instance, forecasts might not be accurate for novel scenarios where there is no reference available to determine whether a forecast is good.

The TimeSeriesSelector object defines the items that you want in the what-if analysis.

> **Note**
>
> Your data must be in comma-separated values (CSV) format to create a what-if analysis.

Request Syntax

```
{
   "ForecastArn": "string",
   "Tags": [
      {
         "Key": "string",
         "Value": "string"
      }
   ],
   "TimeSeriesSelector": {
```

```
      "TimeSeriesIdentifiers": {
         "DataSource": {
            "S3Config": {
               "KMSKeyArn": "string",
               "Path": "string",
               "RoleArn": "string"
            }
         },
         "Format": "string",
         "Schema": {
            "Attributes": [
               {
                  "AttributeName": "string",
                  "AttributeType": "string"
               }
            ]
         }
      }
   },
   "WhatIfAnalysisName": "string"
}
```

Request Parameters

The request accepts the following data in JSON format.

ForecastArn

The Amazon Resource Name (ARN) of the baseline forecast.

Type: String

Length Constraints: Maximum length of 256.

Pattern: `arn:([a-z\d-]+):forecast:.*:.*:.+`

Required: Yes

Tags

A list of tags to apply to the what if forecast.

Type: Array of Tag objects

Array Members: Minimum number of 0 items. Maximum number of 200 items.

Required: No

TimeSeriesSelector

Defines the set of time series that are used in the what-if analysis with a `TimeSeriesIdentifiers` object. What-if analyses are performed only for the time series in this object.

The `TimeSeriesIdentifiers` object needs the following information:

- `DataSource`
- `Format`
- `Schema`

Type: TimeSeriesSelector object

Required: No

WhatIfAnalysisName

The name of the what-if analysis. Each name must be unique.

Type: String

Length Constraints: Minimum length of 1. Maximum length of 63.

Pattern: `^[a-zA-Z][a-zA-Z0-9_]*`

Required: Yes

Response Syntax

```
{
   "WhatIfAnalysisArn": "string"
}
```

Response Elements

If the action is successful, the service sends back an HTTP 200 response.

The following data is returned in JSON format by the service.

WhatIfAnalysisArn

The Amazon Resource Name (ARN) of the what-if analysis.

Type: String

Length Constraints: Maximum length of 256.

Pattern: `arn:([a-z\d-]+):forecast:.*:.*:.+`

Errors

InvalidInputException

We can't process the request because it includes an invalid value or a value that exceeds the valid range.

HTTP Status Code: 400

LimitExceededException

The limit on the number of resources per account has been exceeded.

HTTP Status Code: 400

ResourceAlreadyExistsException

There is already a resource with this name. Try again with a different name.

HTTP Status Code: 400

ResourceInUseException

The specified resource is in use.

HTTP Status Code: 400

ResourceNotFoundException

We can't find a resource with that Amazon Resource Name (ARN). Check the ARN and try again.

HTTP Status Code: 400

See Also

For more information about using this API in one of the language-specific AWS SDKs, see the following:

- AWS Command Line Interface
- AWS SDK for .NET
- AWS SDK for C++
- AWS SDK for Go v2
- AWS SDK for Java V2
- AWS SDK for JavaScript V3
- AWS SDK for PHP V3
- AWS SDK for Python
- AWS SDK for Ruby V3

CreateWhatIfForecast

Service: Amazon Forecast Service

A what-if forecast is a forecast that is created from a modified version of the baseline forecast. Each what-if forecast incorporates either a replacement dataset or a set of transformations to the original dataset.

Request Syntax

```
{
   "Tags": [
      {
         "Key": "string",
         "Value": "string"
      }
   ],
   "TimeSeriesReplacementsDataSource": {
      "Format": "string",
      "S3Config": {
         "KMSKeyArn": "string",
         "Path": "string",
         "RoleArn": "string"
      },
      "Schema": {
         "Attributes": [
            {
               "AttributeName": "string",
               "AttributeType": "string"
            }
         ]
      },
      "TimestampFormat": "string"
   },
   "TimeSeriesTransformations": [
      {
         "Action": {
            "AttributeName": "string",
            "Operation": "string",
            "Value": number
         },
         "TimeSeriesConditions": [
            {
               "AttributeName": "string",
```

```
                "AttributeValue": "string",
                "Condition": "string"
            }
        ]
    }
  ],
  "WhatIfAnalysisArn": "string",
  "WhatIfForecastName": "string"
}
```

Request Parameters

The request accepts the following data in JSON format.

Tags

A list of tags to apply to the what if forecast.

Type: Array of Tag objects

Array Members: Minimum number of 0 items. Maximum number of 200 items.

Required: No

TimeSeriesReplacementsDataSource

The replacement time series dataset, which contains the rows that you want to change in the related time series dataset. A replacement time series does not need to contain all rows that are in the baseline related time series. Include only the rows (measure-dimension combinations) that you want to include in the what-if forecast.

This dataset is merged with the original time series to create a transformed dataset that is used for the what-if analysis.

This dataset should contain the items to modify (such as item_id or workforce_type), any relevant dimensions, the timestamp column, and at least one of the related time series columns. This file should not contain duplicate timestamps for the same time series. This file must be in CSV format.

Timestamps and item_ids not included in this dataset are not included in the what-if analysis.

Type: TimeSeriesReplacementsDataSource object

Required: No

TimeSeriesTransformations

The transformations that are applied to the baseline time series. Each transformation contains an action and a set of conditions. An action is applied only when all conditions are met. If no conditions are provided, the action is applied to all items.

Type: Array of TimeSeriesTransformation objects

Array Members: Minimum number of 0 items. Maximum number of 30 items.

Required: No

WhatIfAnalysisArn

The Amazon Resource Name (ARN) of the what-if analysis.

Type: String

Length Constraints: Maximum length of 256.

Pattern: `arn:([a-z\d-]+):forecast:.*:.*:.+`

Required: Yes

WhatIfForecastName

The name of the what-if forecast. Names must be unique within each what-if analysis.

Type: String

Length Constraints: Minimum length of 1. Maximum length of 63.

Pattern: `^[a-zA-Z][a-zA-Z0-9_]*`

Required: Yes

Response Syntax

```
{
   "WhatIfForecastArn": "string"
}
```

Response Elements

If the action is successful, the service sends back an HTTP 200 response.

The following data is returned in JSON format by the service.

WhatIfForecastArn

The Amazon Resource Name (ARN) of the what-if forecast.

Type: String

Length Constraints: Maximum length of 300.

Pattern: `arn:([a-z\d-]+):forecast:.*:.*:.+`

Errors

InvalidInputException

We can't process the request because it includes an invalid value or a value that exceeds the valid range.

HTTP Status Code: 400

LimitExceededException

The limit on the number of resources per account has been exceeded.

HTTP Status Code: 400

ResourceAlreadyExistsException

There is already a resource with this name. Try again with a different name.

HTTP Status Code: 400

ResourceInUseException

The specified resource is in use.

HTTP Status Code: 400

ResourceNotFoundException

We can't find a resource with that Amazon Resource Name (ARN). Check the ARN and try again.

HTTP Status Code: 400

See Also

For more information about using this API in one of the language-specific AWS SDKs, see the following:

- AWS Command Line Interface
- AWS SDK for .NET
- AWS SDK for C++
- AWS SDK for Go v2
- AWS SDK for Java V2
- AWS SDK for JavaScript V3
- AWS SDK for PHP V3
- AWS SDK for Python
- AWS SDK for Ruby V3

CreateWhatIfForecastExport

Service: Amazon Forecast Service

Exports a forecast created by the CreateWhatIfForecast operation to your Amazon Simple Storage Service (Amazon S3) bucket. The forecast file name will match the following conventions:

≈<ForecastExportJobName>_<ExportTimestamp>_<PartNumber>

The <ExportTimestamp> component is in Java SimpleDateFormat (yyyy-MM-ddTHH-mm-ssZ).

You must specify a DataDestination object that includes an AWS Identity and Access Management (IAM) role that Amazon Forecast can assume to access the Amazon S3 bucket. For more information, see Set Up Permissions for Amazon Forecast.

For more information, see Generating Forecasts.

To get a list of all your what-if forecast export jobs, use the ListWhatIfForecastExports operation.

> **Note**
>
> The `Status` of the forecast export job must be `ACTIVE` before you can access the forecast in your Amazon S3 bucket. To get the status, use the DescribeWhatIfForecastExport operation.

Request Syntax

```
{
   "Destination": {
      "S3Config": {
         "KMSKeyArn": "string",
         "Path": "string",
         "RoleArn": "string"
      }
   },
   "Format": "string",
   "Tags": [
      {
         "Key": "string",
         "Value": "string"
      }
   ],
```

```
   "WhatIfForecastArns": [ "string" ],
   "WhatIfForecastExportName": "string"
}
```

Request Parameters

The request accepts the following data in JSON format.

Destination

The location where you want to save the forecast and an AWS Identity and Access Management (IAM) role that Amazon Forecast can assume to access the location. The forecast must be exported to an Amazon S3 bucket.

If encryption is used, `Destination` must include an AWS Key Management Service (KMS) key. The IAM role must allow Amazon Forecast permission to access the key.

Type: DataDestination object

Required: Yes

Format

The format of the exported data, CSV or PARQUET.

Type: String

Length Constraints: Maximum length of 7.

Pattern: `^CSV|PARQUET$`

Required: No

Tags

A list of tags to apply to the what if forecast.

Type: Array of Tag objects

Array Members: Minimum number of 0 items. Maximum number of 200 items.

Required: No

WhatIfForecastArns

The list of what-if forecast Amazon Resource Names (ARNs) to export.

Type: Array of strings

Array Members: Minimum number of 1 item. Maximum number of 50 items.

Length Constraints: Maximum length of 300.

Pattern: `arn:([a-z\d-]+):forecast:.*:.*:.+`

Required: Yes

WhatIfForecastExportName

The name of the what-if forecast to export.

Type: String

Length Constraints: Minimum length of 1. Maximum length of 63.

Pattern: `^[a-zA-Z][a-zA-Z0-9_]*`

Required: Yes

Response Syntax

```
{
   "WhatIfForecastExportArn": "string"
}
```

Response Elements

If the action is successful, the service sends back an HTTP 200 response.

The following data is returned in JSON format by the service.

WhatIfForecastExportArn

The Amazon Resource Name (ARN) of the what-if forecast.

Type: String

Length Constraints: Maximum length of 300.

Pattern: `arn:([a-z\d-]+):forecast:.*:.*:.+`

Errors

InvalidInputException

We can't process the request because it includes an invalid value or a value that exceeds the valid range.

HTTP Status Code: 400

LimitExceededException

The limit on the number of resources per account has been exceeded.

HTTP Status Code: 400

ResourceAlreadyExistsException

There is already a resource with this name. Try again with a different name.

HTTP Status Code: 400

ResourceInUseException

The specified resource is in use.

HTTP Status Code: 400

ResourceNotFoundException

We can't find a resource with that Amazon Resource Name (ARN). Check the ARN and try again.

HTTP Status Code: 400

See Also

For more information about using this API in one of the language-specific AWS SDKs, see the following:

- AWS Command Line Interface
- AWS SDK for .NET
- AWS SDK for C++
- AWS SDK for Go v2
- AWS SDK for Java V2

- AWS SDK for JavaScript V3
- AWS SDK for PHP V3
- AWS SDK for Python
- AWS SDK for Ruby V3

DeleteDataset

Service: Amazon Forecast Service

Deletes an Amazon Forecast dataset that was created using the CreateDataset operation. You can only delete datasets that have a status of `ACTIVE` or `CREATE_FAILED`. To get the status use the DescribeDataset operation.

> **Note**
>
> Forecast does not automatically update any dataset groups that contain the deleted dataset. In order to update the dataset group, use the UpdateDatasetGroup operation, omitting the deleted dataset's ARN.

Request Syntax

```
{
   "DatasetArn": "string"
}
```

Request Parameters

The request accepts the following data in JSON format.

DatasetArn

The Amazon Resource Name (ARN) of the dataset to delete.

Type: String

Length Constraints: Maximum length of 256.

Pattern: `arn:([a-z\d-]+):forecast:.*:.*:.+`

Required: Yes

Response Elements

If the action is successful, the service sends back an HTTP 200 response with an empty HTTP body.

Errors

InvalidInputException

We can't process the request because it includes an invalid value or a value that exceeds the valid range.

HTTP Status Code: 400

ResourceInUseException

The specified resource is in use.

HTTP Status Code: 400

ResourceNotFoundException

We can't find a resource with that Amazon Resource Name (ARN). Check the ARN and try again.

HTTP Status Code: 400

See Also

For more information about using this API in one of the language-specific AWS SDKs, see the following:

- AWS Command Line Interface
- AWS SDK for .NET
- AWS SDK for C++
- AWS SDK for Go v2
- AWS SDK for Java V2
- AWS SDK for JavaScript V3
- AWS SDK for PHP V3
- AWS SDK for Python
- AWS SDK for Ruby V3

DeleteDatasetGroup

Service: Amazon Forecast Service

Deletes a dataset group created using the CreateDatasetGroup operation. You can only delete dataset groups that have a status of ACTIVE, CREATE_FAILED, or UPDATE_FAILED. To get the status, use the DescribeDatasetGroup operation.

This operation deletes only the dataset group, not the datasets in the group.

Request Syntax

```
{
   "DatasetGroupArn": "string"
}
```

Request Parameters

The request accepts the following data in JSON format.

DatasetGroupArn

The Amazon Resource Name (ARN) of the dataset group to delete.

Type: String

Length Constraints: Maximum length of 256.

Pattern: `arn:([a-z\d-]+):forecast:.*:.*:.+`

Required: Yes

Response Elements

If the action is successful, the service sends back an HTTP 200 response with an empty HTTP body.

Errors

InvalidInputException

We can't process the request because it includes an invalid value or a value that exceeds the valid range.

HTTP Status Code: 400

ResourceInUseException

The specified resource is in use.

HTTP Status Code: 400

ResourceNotFoundException

We can't find a resource with that Amazon Resource Name (ARN). Check the ARN and try again.

HTTP Status Code: 400

See Also

For more information about using this API in one of the language-specific AWS SDKs, see the following:

- AWS Command Line Interface
- AWS SDK for .NET
- AWS SDK for C++
- AWS SDK for Go v2
- AWS SDK for Java V2
- AWS SDK for JavaScript V3
- AWS SDK for PHP V3
- AWS SDK for Python
- AWS SDK for Ruby V3

DeleteDatasetImportJob

Service: Amazon Forecast Service

Deletes a dataset import job created using the CreateDatasetImportJob operation. You can delete only dataset import jobs that have a status of `ACTIVE` or `CREATE_FAILED`. To get the status, use the DescribeDatasetImportJob operation.

Request Syntax

```
{
   "DatasetImportJobArn": "string"
}
```

Request Parameters

The request accepts the following data in JSON format.

DatasetImportJobArn

The Amazon Resource Name (ARN) of the dataset import job to delete.

Type: String

Length Constraints: Maximum length of 256.

Pattern: `arn:([a-z\d-]+):forecast:.*:.*:.+`

Required: Yes

Response Elements

If the action is successful, the service sends back an HTTP 200 response with an empty HTTP body.

Errors

InvalidInputException

We can't process the request because it includes an invalid value or a value that exceeds the valid range.

HTTP Status Code: 400

ResourceInUseException

The specified resource is in use.

HTTP Status Code: 400

ResourceNotFoundException

We can't find a resource with that Amazon Resource Name (ARN). Check the ARN and try again.

HTTP Status Code: 400

See Also

For more information about using this API in one of the language-specific AWS SDKs, see the following:

- AWS Command Line Interface
- AWS SDK for .NET
- AWS SDK for C++
- AWS SDK for Go v2
- AWS SDK for Java V2
- AWS SDK for JavaScript V3
- AWS SDK for PHP V3
- AWS SDK for Python
- AWS SDK for Ruby V3

DeleteExplainability

Service: Amazon Forecast Service

Deletes an Explainability resource.

You can delete only predictor that have a status of `ACTIVE` or `CREATE_FAILED`. To get the status, use the DescribeExplainability operation.

Request Syntax

```
{
   "ExplainabilityArn": "string"
}
```

Request Parameters

The request accepts the following data in JSON format.

ExplainabilityArn

The Amazon Resource Name (ARN) of the Explainability resource to delete.

Type: String

Length Constraints: Maximum length of 256.

Pattern: `arn:([a-z\d-]+):forecast:.*:.*:.+`

Required: Yes

Response Elements

If the action is successful, the service sends back an HTTP 200 response with an empty HTTP body.

Errors

InvalidInputException

We can't process the request because it includes an invalid value or a value that exceeds the valid range.

HTTP Status Code: 400

ResourceInUseException

The specified resource is in use.

HTTP Status Code: 400

ResourceNotFoundException

We can't find a resource with that Amazon Resource Name (ARN). Check the ARN and try again.

HTTP Status Code: 400

See Also

For more information about using this API in one of the language-specific AWS SDKs, see the following:

- AWS Command Line Interface
- AWS SDK for .NET
- AWS SDK for C++
- AWS SDK for Go v2
- AWS SDK for Java V2
- AWS SDK for JavaScript V3
- AWS SDK for PHP V3
- AWS SDK for Python
- AWS SDK for Ruby V3

DeleteExplainabilityExport

Service: Amazon Forecast Service

Deletes an Explainability export.

Request Syntax

```
{
   "ExplainabilityExportArn": "string"
}
```

Request Parameters

The request accepts the following data in JSON format.

ExplainabilityExportArn

The Amazon Resource Name (ARN) of the Explainability export to delete.

Type: String

Length Constraints: Maximum length of 256.

Pattern: `arn:([a-z\d-]+):forecast:.*:.*:.+`

Required: Yes

Response Elements

If the action is successful, the service sends back an HTTP 200 response with an empty HTTP body.

Errors

InvalidInputException

We can't process the request because it includes an invalid value or a value that exceeds the valid range.

HTTP Status Code: 400

ResourceInUseException

The specified resource is in use.

HTTP Status Code: 400

ResourceNotFoundException

We can't find a resource with that Amazon Resource Name (ARN). Check the ARN and try again.

HTTP Status Code: 400

See Also

For more information about using this API in one of the language-specific AWS SDKs, see the following:

- AWS Command Line Interface
- AWS SDK for .NET
- AWS SDK for C++
- AWS SDK for Go v2
- AWS SDK for Java V2
- AWS SDK for JavaScript V3
- AWS SDK for PHP V3
- AWS SDK for Python
- AWS SDK for Ruby V3

DeleteForecast

Service: Amazon Forecast Service

Deletes a forecast created using the CreateForecast operation. You can delete only forecasts that have a status of `ACTIVE` or `CREATE_FAILED`. To get the status, use the DescribeForecast operation.

You can't delete a forecast while it is being exported. After a forecast is deleted, you can no longer query the forecast.

Request Syntax

```
{
   "ForecastArn": "string"
}
```

Request Parameters

The request accepts the following data in JSON format.

ForecastArn

The Amazon Resource Name (ARN) of the forecast to delete.

Type: String

Length Constraints: Maximum length of 256.

Pattern: `arn:([a-z\d-]+):forecast:.*:.*:.+`

Required: Yes

Response Elements

If the action is successful, the service sends back an HTTP 200 response with an empty HTTP body.

Errors

InvalidInputException

We can't process the request because it includes an invalid value or a value that exceeds the valid range.

HTTP Status Code: 400

ResourceInUseException

The specified resource is in use.

HTTP Status Code: 400

ResourceNotFoundException

We can't find a resource with that Amazon Resource Name (ARN). Check the ARN and try again.

HTTP Status Code: 400

See Also

For more information about using this API in one of the language-specific AWS SDKs, see the following:

- AWS Command Line Interface
- AWS SDK for .NET
- AWS SDK for C++
- AWS SDK for Go v2
- AWS SDK for Java V2
- AWS SDK for JavaScript V3
- AWS SDK for PHP V3
- AWS SDK for Python
- AWS SDK for Ruby V3

DeleteForecastExportJob

Service: Amazon Forecast Service

Deletes a forecast export job created using the CreateForecastExportJob operation. You can delete only export jobs that have a status of `ACTIVE` or `CREATE_FAILED`. To get the status, use the DescribeForecastExportJob operation.

Request Syntax

```
{
   "ForecastExportJobArn": "string"
}
```

Request Parameters

The request accepts the following data in JSON format.

ForecastExportJobArn

The Amazon Resource Name (ARN) of the forecast export job to delete.

Type: String

Length Constraints: Maximum length of 256.

Pattern: `arn:([a-z\d-]+):forecast:.*:.*:.+`

Required: Yes

Response Elements

If the action is successful, the service sends back an HTTP 200 response with an empty HTTP body.

Errors

InvalidInputException

We can't process the request because it includes an invalid value or a value that exceeds the valid range.

HTTP Status Code: 400

ResourceInUseException

The specified resource is in use.

HTTP Status Code: 400

ResourceNotFoundException

We can't find a resource with that Amazon Resource Name (ARN). Check the ARN and try again.

HTTP Status Code: 400

See Also

For more information about using this API in one of the language-specific AWS SDKs, see the following:

- AWS Command Line Interface
- AWS SDK for .NET
- AWS SDK for C++
- AWS SDK for Go v2
- AWS SDK for Java V2
- AWS SDK for JavaScript V3
- AWS SDK for PHP V3
- AWS SDK for Python
- AWS SDK for Ruby V3

DeleteMonitor

Service: Amazon Forecast Service

Deletes a monitor resource. You can only delete a monitor resource with a status of `ACTIVE`, `ACTIVE_STOPPED`, `CREATE_FAILED`, or `CREATE_STOPPED`.

Request Syntax

```
{
   "MonitorArn": "string"
}
```

Request Parameters

The request accepts the following data in JSON format.

MonitorArn

The Amazon Resource Name (ARN) of the monitor resource to delete.

Type: String

Length Constraints: Maximum length of 256.

Pattern: `arn:([a-z\d-]+):forecast:.*:.*:.+`

Required: Yes

Response Elements

If the action is successful, the service sends back an HTTP 200 response with an empty HTTP body.

Errors

InvalidInputException

We can't process the request because it includes an invalid value or a value that exceeds the valid range.

HTTP Status Code: 400

ResourceInUseException

The specified resource is in use.

HTTP Status Code: 400

ResourceNotFoundException

We can't find a resource with that Amazon Resource Name (ARN). Check the ARN and try again.

HTTP Status Code: 400

See Also

For more information about using this API in one of the language-specific AWS SDKs, see the following:

- AWS Command Line Interface
- AWS SDK for .NET
- AWS SDK for C++
- AWS SDK for Go v2
- AWS SDK for Java V2
- AWS SDK for JavaScript V3
- AWS SDK for PHP V3
- AWS SDK for Python
- AWS SDK for Ruby V3

DeletePredictor

Service: Amazon Forecast Service

Deletes a predictor created using the DescribePredictor or CreatePredictor operations. You can delete only predictor that have a status of `ACTIVE` or `CREATE_FAILED`. To get the status, use the DescribePredictor operation.

Request Syntax

```
{
   "PredictorArn": "string"
}
```

Request Parameters

The request accepts the following data in JSON format.

PredictorArn

The Amazon Resource Name (ARN) of the predictor to delete.

Type: String

Length Constraints: Maximum length of 256.

Pattern: `arn:([a-z\d-]+):forecast:.*:.*:.+`

Required: Yes

Response Elements

If the action is successful, the service sends back an HTTP 200 response with an empty HTTP body.

Errors

InvalidInputException

We can't process the request because it includes an invalid value or a value that exceeds the valid range.

HTTP Status Code: 400

ResourceInUseException

The specified resource is in use.

HTTP Status Code: 400

ResourceNotFoundException

We can't find a resource with that Amazon Resource Name (ARN). Check the ARN and try again.

HTTP Status Code: 400

See Also

For more information about using this API in one of the language-specific AWS SDKs, see the following:

- AWS Command Line Interface
- AWS SDK for .NET
- AWS SDK for C++
- AWS SDK for Go v2
- AWS SDK for Java V2
- AWS SDK for JavaScript V3
- AWS SDK for PHP V3
- AWS SDK for Python
- AWS SDK for Ruby V3

DeletePredictorBacktestExportJob

Service: Amazon Forecast Service

Deletes a predictor backtest export job.

Request Syntax

```
{
   "PredictorBacktestExportJobArn": "string"
}
```

Request Parameters

The request accepts the following data in JSON format.

PredictorBacktestExportJobArn

The Amazon Resource Name (ARN) of the predictor backtest export job to delete.

Type: String

Length Constraints: Maximum length of 256.

Pattern: `arn:([a-z\d-]+):forecast:.*:.*:.+`

Required: Yes

Response Elements

If the action is successful, the service sends back an HTTP 200 response with an empty HTTP body.

Errors

InvalidInputException

We can't process the request because it includes an invalid value or a value that exceeds the valid range.

HTTP Status Code: 400

ResourceInUseException

The specified resource is in use.

HTTP Status Code: 400

ResourceNotFoundException

We can't find a resource with that Amazon Resource Name (ARN). Check the ARN and try again.

HTTP Status Code: 400

See Also

For more information about using this API in one of the language-specific AWS SDKs, see the following:

- AWS Command Line Interface
- AWS SDK for .NET
- AWS SDK for C++
- AWS SDK for Go v2
- AWS SDK for Java V2
- AWS SDK for JavaScript V3
- AWS SDK for PHP V3
- AWS SDK for Python
- AWS SDK for Ruby V3

DeleteResourceTree

Service: Amazon Forecast Service

Deletes an entire resource tree. This operation will delete the parent resource and its child resources.

Child resources are resources that were created from another resource. For example, when a forecast is generated from a predictor, the forecast is the child resource and the predictor is the parent resource.

Amazon Forecast resources possess the following parent-child resource hierarchies:

- **Dataset**: dataset import jobs
- **Dataset Group**: predictors, predictor backtest export jobs, forecasts, forecast export jobs
- **Predictor**: predictor backtest export jobs, forecasts, forecast export jobs
- **Forecast**: forecast export jobs

> **Note**
>
> `DeleteResourceTree` will only delete Amazon Forecast resources, and will not delete datasets or exported files stored in Amazon S3.

Request Syntax

```
{
   "ResourceArn": "string"
}
```

Request Parameters

The request accepts the following data in JSON format.

ResourceArn

The Amazon Resource Name (ARN) of the parent resource to delete. All child resources of the parent resource will also be deleted.

Type: String

Length Constraints: Maximum length of 256.

Pattern: `arn:([a-z\d-]+):forecast:.*:.*:.+`

Required: Yes

Response Elements

If the action is successful, the service sends back an HTTP 200 response with an empty HTTP body.

Errors

InvalidInputException

We can't process the request because it includes an invalid value or a value that exceeds the valid range.

HTTP Status Code: 400

ResourceInUseException

The specified resource is in use.

HTTP Status Code: 400

ResourceNotFoundException

We can't find a resource with that Amazon Resource Name (ARN). Check the ARN and try again.

HTTP Status Code: 400

See Also

For more information about using this API in one of the language-specific AWS SDKs, see the following:

- AWS Command Line Interface
- AWS SDK for .NET
- AWS SDK for C++
- AWS SDK for Go v2
- AWS SDK for Java V2

- AWS SDK for JavaScript V3
- AWS SDK for PHP V3
- AWS SDK for Python
- AWS SDK for Ruby V3

DeleteWhatIfAnalysis

Service: Amazon Forecast Service

Deletes a what-if analysis created using the CreateWhatIfAnalysis operation. You can delete only what-if analyses that have a status of `ACTIVE` or `CREATE_FAILED`. To get the status, use the DescribeWhatIfAnalysis operation.

You can't delete a what-if analysis while any of its forecasts are being exported.

Request Syntax

```
{
   "WhatIfAnalysisArn": "string"
}
```

Request Parameters

The request accepts the following data in JSON format.

WhatIfAnalysisArn

The Amazon Resource Name (ARN) of the what-if analysis that you want to delete.

Type: String

Length Constraints: Maximum length of 256.

Pattern: `arn:([a-z\d-]+):forecast:.*:.*:.+`

Required: Yes

Response Elements

If the action is successful, the service sends back an HTTP 200 response with an empty HTTP body.

Errors

InvalidInputException

We can't process the request because it includes an invalid value or a value that exceeds the valid range.

HTTP Status Code: 400

ResourceInUseException

The specified resource is in use.

HTTP Status Code: 400

ResourceNotFoundException

We can't find a resource with that Amazon Resource Name (ARN). Check the ARN and try again.

HTTP Status Code: 400

See Also

For more information about using this API in one of the language-specific AWS SDKs, see the following:

- AWS Command Line Interface
- AWS SDK for .NET
- AWS SDK for C++
- AWS SDK for Go v2
- AWS SDK for Java V2
- AWS SDK for JavaScript V3
- AWS SDK for PHP V3
- AWS SDK for Python
- AWS SDK for Ruby V3

DeleteWhatIfForecast

Service: Amazon Forecast Service

Deletes a what-if forecast created using the CreateWhatIfForecast operation. You can delete only what-if forecasts that have a status of `ACTIVE` or `CREATE_FAILED`. To get the status, use the DescribeWhatIfForecast operation.

You can't delete a what-if forecast while it is being exported. After a what-if forecast is deleted, you can no longer query the what-if analysis.

Request Syntax

```
{
   "WhatIfForecastArn": "string"
}
```

Request Parameters

The request accepts the following data in JSON format.

WhatIfForecastArn

The Amazon Resource Name (ARN) of the what-if forecast that you want to delete.

Type: String

Length Constraints: Maximum length of 300.

Pattern: `arn:([a-z\d-]+):forecast:.*:.*:.+`

Required: Yes

Response Elements

If the action is successful, the service sends back an HTTP 200 response with an empty HTTP body.

Errors

InvalidInputException

We can't process the request because it includes an invalid value or a value that exceeds the valid range.

HTTP Status Code: 400

ResourceInUseException

The specified resource is in use.

HTTP Status Code: 400

ResourceNotFoundException

We can't find a resource with that Amazon Resource Name (ARN). Check the ARN and try again.

HTTP Status Code: 400

See Also

For more information about using this API in one of the language-specific AWS SDKs, see the following:

- AWS Command Line Interface
- AWS SDK for .NET
- AWS SDK for C++
- AWS SDK for Go v2
- AWS SDK for Java V2
- AWS SDK for JavaScript V3
- AWS SDK for PHP V3
- AWS SDK for Python
- AWS SDK for Ruby V3

DeleteWhatIfForecastExport

Service: Amazon Forecast Service

Deletes a what-if forecast export created using the CreateWhatIfForecastExport operation. You can delete only what-if forecast exports that have a status of `ACTIVE` or `CREATE_FAILED`. To get the status, use the DescribeWhatIfForecastExport operation.

Request Syntax

```
{
   "WhatIfForecastExportArn": "string"
}
```

Request Parameters

The request accepts the following data in JSON format.

WhatIfForecastExportArn

The Amazon Resource Name (ARN) of the what-if forecast export that you want to delete.

Type: String

Length Constraints: Maximum length of 300.

Pattern: `arn:([a-z\d-]+):forecast:.*:.*:.+`

Required: Yes

Response Elements

If the action is successful, the service sends back an HTTP 200 response with an empty HTTP body.

Errors

InvalidInputException

We can't process the request because it includes an invalid value or a value that exceeds the valid range.

HTTP Status Code: 400

ResourceInUseException

The specified resource is in use.

HTTP Status Code: 400

ResourceNotFoundException

We can't find a resource with that Amazon Resource Name (ARN). Check the ARN and try again.

HTTP Status Code: 400

See Also

For more information about using this API in one of the language-specific AWS SDKs, see the following:

- AWS Command Line Interface
- AWS SDK for .NET
- AWS SDK for C++
- AWS SDK for Go v2
- AWS SDK for Java V2
- AWS SDK for JavaScript V3
- AWS SDK for PHP V3
- AWS SDK for Python
- AWS SDK for Ruby V3

DescribeAutoPredictor

Service: Amazon Forecast Service

Describes a predictor created using the CreateAutoPredictor operation.

Request Syntax

```
{
   "PredictorArn": "string"
}
```

Request Parameters

The request accepts the following data in JSON format.

PredictorArn

The Amazon Resource Name (ARN) of the predictor.

Type: String

Length Constraints: Maximum length of 256.

Pattern: `arn:([a-z\d-]+):forecast:.*:.*:.+`

Required: Yes

Response Syntax

```
{
   "CreationTime": number,
   "DataConfig": {
      "AdditionalDatasets": [
         {
            "Configuration": {
               "string" : [ "string" ]
            },
            "Name": "string"
         }
      ],
      "AttributeConfigs": [
         {
            "AttributeName": "string",
```

```
            "Transformations": {
               "string" : "string"
            }
         }
      ],
      "DatasetGroupArn": "string"
   },
   "DatasetImportJobArns": [ "string" ],
   "EncryptionConfig": {
      "KMSKeyArn": "string",
      "RoleArn": "string"
   },
   "EstimatedTimeRemainingInMinutes": number,
   "ExplainabilityInfo": {
      "ExplainabilityArn": "string",
      "Status": "string"
   },
   "ForecastDimensions": [ "string" ],
   "ForecastFrequency": "string",
   "ForecastHorizon": number,
   "ForecastTypes": [ "string" ],
   "LastModificationTime": number,
   "Message": "string",
   "MonitorInfo": {
      "MonitorArn": "string",
      "Status": "string"
   },
   "OptimizationMetric": "string",
   "PredictorArn": "string",
   "PredictorName": "string",
   "ReferencePredictorSummary": {
      "Arn": "string",
      "State": "string"
   },
   "Status": "string",
   "TimeAlignmentBoundary": {
      "DayOfMonth": number,
      "DayOfWeek": "string",
      "Hour": number,
      "Month": "string"
   }
}
```

Response Elements

If the action is successful, the service sends back an HTTP 200 response.

The following data is returned in JSON format by the service.

CreationTime

The timestamp of the CreateAutoPredictor request.

Type: Timestamp

DataConfig

The data configuration for your dataset group and any additional datasets.

Type: DataConfig object

DatasetImportJobArns

An array of the ARNs of the dataset import jobs used to import training data for the predictor.

Type: Array of strings

Length Constraints: Maximum length of 256.

Pattern: `arn:([a-z\d-]+):forecast:.*:.*:.+`

EncryptionConfig

An AWS Key Management Service (KMS) key and an AWS Identity and Access Management (IAM) role that Amazon Forecast can assume to access the key. You can specify this optional object in the CreateDataset and CreatePredictor requests.

Type: EncryptionConfig object

EstimatedTimeRemainingInMinutes

The estimated time remaining in minutes for the predictor training job to complete.

Type: Long

ExplainabilityInfo

Provides the status and ARN of the Predictor Explainability.

Type: ExplainabilityInfo object

ForecastDimensions

An array of dimension (field) names that specify the attributes used to group your time series.

Type: Array of strings

Array Members: Minimum number of 1 item. Maximum number of 10 items.

Length Constraints: Minimum length of 1. Maximum length of 63.

Pattern: `^[a-zA-Z][a-zA-Z0-9_]*`

ForecastFrequency

The frequency of predictions in a forecast.

Valid intervals are Y (Year), M (Month), W (Week), D (Day), H (Hour), 30min (30 minutes), 15min (15 minutes), 10min (10 minutes), 5min (5 minutes), and 1min (1 minute). For example, "Y" indicates every year and "5min" indicates every five minutes.

Type: String

Length Constraints: Minimum length of 1. Maximum length of 5.

Pattern: `^Y|M|W|D|H|30min|15min|10min|5min|1min$`

ForecastHorizon

The number of time-steps that the model predicts. The forecast horizon is also called the prediction length.

Type: Integer

ForecastTypes

The forecast types used during predictor training. Default value is ["0.1","0.5","0.9"].

Type: Array of strings

Array Members: Minimum number of 1 item. Maximum number of 20 items.

Length Constraints: Minimum length of 2. Maximum length of 4.

Pattern: `(^0?\.\d\d?$|^mean$)`

LastModificationTime

The last time the resource was modified. The timestamp depends on the status of the job:

- `CREATE_PENDING` - The `CreationTime`.
- `CREATE_IN_PROGRESS` - The current timestamp.
- `CREATE_STOPPING` - The current timestamp.
- `CREATE_STOPPED` - When the job stopped.
- `ACTIVE` or `CREATE_FAILED` - When the job finished or failed.

Type: Timestamp

Message

In the event of an error, a message detailing the cause of the error.

Type: String

MonitorInfo

A MonitorInfo object with the Amazon Resource Name (ARN) and status of the monitor resource.

Type: MonitorInfo object

OptimizationMetric

The accuracy metric used to optimize the predictor.

Type: String

Valid Values: `WAPE | RMSE | AverageWeightedQuantileLoss | MASE | MAPE`

PredictorArn

The Amazon Resource Name (ARN) of the predictor

Type: String

Length Constraints: Maximum length of 256.

Pattern: `arn:([a-z\d-]+):forecast:.*:.*:.+`

PredictorName

The name of the predictor.

Type: String

Length Constraints: Minimum length of 1. Maximum length of 63.

Pattern: `^[a-zA-Z][a-zA-Z0-9_]*`

ReferencePredictorSummary

The ARN and state of the reference predictor. This parameter is only valid for retrained or upgraded predictors.

Type: ReferencePredictorSummary object

Status

The status of the predictor. States include:

- `ACTIVE`
- `CREATE_PENDING`, `CREATE_IN_PROGRESS`, `CREATE_FAILED`
- `CREATE_STOPPING`, `CREATE_STOPPED`
- `DELETE_PENDING`, `DELETE_IN_PROGRESS`, `DELETE_FAILED`

Type: String

Length Constraints: Maximum length of 256.

TimeAlignmentBoundary

The time boundary Forecast uses when aggregating data.

Type: TimeAlignmentBoundary object

Errors

InvalidInputException

We can't process the request because it includes an invalid value or a value that exceeds the valid range.

HTTP Status Code: 400

ResourceNotFoundException

We can't find a resource with that Amazon Resource Name (ARN). Check the ARN and try again.

HTTP Status Code: 400

See Also

For more information about using this API in one of the language-specific AWS SDKs, see the following:

- AWS Command Line Interface
- AWS SDK for .NET
- AWS SDK for C++
- AWS SDK for Go v2
- AWS SDK for Java V2
- AWS SDK for JavaScript V3
- AWS SDK for PHP V3
- AWS SDK for Python
- AWS SDK for Ruby V3

DescribeDataset

Service: Amazon Forecast Service

Describes an Amazon Forecast dataset created using the CreateDataset operation.

In addition to listing the parameters specified in the `CreateDataset` request, this operation includes the following dataset properties:

- `CreationTime`
- `LastModificationTime`
- `Status`

Request Syntax

```
{
   "DatasetArn": "string"
}
```

Request Parameters

The request accepts the following data in JSON format.

DatasetArn

The Amazon Resource Name (ARN) of the dataset.

Type: String

Length Constraints: Maximum length of 256.

Pattern: `arn:([a-z\d-]+):forecast:.*:.*:.+`

Required: Yes

Response Syntax

```
{
   "CreationTime": number,
   "DataFrequency": "string",
   "DatasetArn": "string",
```

```
   "DatasetName": "string",
   "DatasetType": "string",
   "Domain": "string",
   "EncryptionConfig": {
      "KMSKeyArn": "string",
      "RoleArn": "string"
   },
   "LastModificationTime": number,
   "Schema": {
      "Attributes": [
         {
            "AttributeName": "string",
            "AttributeType": "string"
         }
      ]
   },
   "Status": "string"
}
```

Response Elements

If the action is successful, the service sends back an HTTP 200 response.

The following data is returned in JSON format by the service.

CreationTime

When the dataset was created.

Type: Timestamp

DataFrequency

The frequency of data collection.

Valid intervals are Y (Year), M (Month), W (Week), D (Day), H (Hour), 30min (30 minutes), 15min (15 minutes), 10min (10 minutes), 5min (5 minutes), and 1min (1 minute). For example, "M" indicates every month and "30min" indicates every 30 minutes.

Type: String

Length Constraints: Minimum length of 1. Maximum length of 5.

Pattern: `^Y|M|W|D|H|30min|15min|10min|5min|1min$`

DatasetArn

The Amazon Resource Name (ARN) of the dataset.

Type: String

Length Constraints: Maximum length of 256.

Pattern: `arn:([a-z\d-]+):forecast:.*:.*:.+`

DatasetName

The name of the dataset.

Type: String

Length Constraints: Minimum length of 1. Maximum length of 63.

Pattern: `^[a-zA-Z][a-zA-Z0-9_]*`

DatasetType

The dataset type.

Type: String

Valid Values: `TARGET_TIME_SERIES | RELATED_TIME_SERIES | ITEM_METADATA`

Domain

The domain associated with the dataset.

Type: String

Valid Values: `RETAIL | CUSTOM | INVENTORY_PLANNING | EC2_CAPACITY | WORK_FORCE | WEB_TRAFFIC | METRICS`

EncryptionConfig

The AWS Key Management Service (KMS) key and the AWS Identity and Access Management (IAM) role that Amazon Forecast can assume to access the key.

Type: EncryptionConfig object

LastModificationTime

When you create a dataset, `LastModificationTime` is the same as `CreationTime`. While data is being imported to the dataset, `LastModificationTime` is the current time

of the DescribeDataset call. After a CreateDatasetImportJob operation has finished, LastModificationTime is when the import job completed or failed.

Type: Timestamp

Schema

An array of SchemaAttribute objects that specify the dataset fields. Each SchemaAttribute specifies the name and data type of a field.

Type: Schema object

Status

The status of the dataset. States include:

- ACTIVE
- CREATE_PENDING, CREATE_IN_PROGRESS, CREATE_FAILED
- DELETE_PENDING, DELETE_IN_PROGRESS, DELETE_FAILED
- UPDATE_PENDING, UPDATE_IN_PROGRESS, UPDATE_FAILED

The UPDATE states apply while data is imported to the dataset from a call to the CreateDatasetImportJob operation and reflect the status of the dataset import job. For example, when the import job status is CREATE_IN_PROGRESS, the status of the dataset is UPDATE_IN_PROGRESS.

> **Note**
>
> The Status of the dataset must be ACTIVE before you can import training data.

Type: String

Length Constraints: Maximum length of 256.

Errors

InvalidInputException

We can't process the request because it includes an invalid value or a value that exceeds the valid range.

HTTP Status Code: 400

ResourceNotFoundException

We can't find a resource with that Amazon Resource Name (ARN). Check the ARN and try again.

HTTP Status Code: 400

See Also

For more information about using this API in one of the language-specific AWS SDKs, see the following:

- AWS Command Line Interface
- AWS SDK for .NET
- AWS SDK for C++
- AWS SDK for Go v2
- AWS SDK for Java V2
- AWS SDK for JavaScript V3
- AWS SDK for PHP V3
- AWS SDK for Python
- AWS SDK for Ruby V3

DescribeDatasetGroup

Service: Amazon Forecast Service

Describes a dataset group created using the CreateDatasetGroup operation.

In addition to listing the parameters provided in the `CreateDatasetGroup` request, this operation includes the following properties:

- `DatasetArns` - The datasets belonging to the group.
- `CreationTime`
- `LastModificationTime`
- `Status`

Request Syntax

```
{
   "DatasetGroupArn": "string"
}
```

Request Parameters

The request accepts the following data in JSON format.

DatasetGroupArn

The Amazon Resource Name (ARN) of the dataset group.

Type: String

Length Constraints: Maximum length of 256.

Pattern: `arn:([a-z\d-]+):forecast:.*:.*:.+`

Required: Yes

Response Syntax

```
{
   "CreationTime": number,
```

```
    "DatasetArns": [ "string" ],
    "DatasetGroupArn": "string",
    "DatasetGroupName": "string",
    "Domain": "string",
    "LastModificationTime": number,
    "Status": "string"
}
```

Response Elements

If the action is successful, the service sends back an HTTP 200 response.

The following data is returned in JSON format by the service.

CreationTime

When the dataset group was created.

Type: Timestamp

DatasetArns

An array of Amazon Resource Names (ARNs) of the datasets contained in the dataset group.

Type: Array of strings

Length Constraints: Maximum length of 256.

Pattern: `arn:([a-z\d-]+):forecast:.*:.*:.+`

DatasetGroupArn

The ARN of the dataset group.

Type: String

Length Constraints: Maximum length of 256.

Pattern: `arn:([a-z\d-]+):forecast:.*:.*:.+`

DatasetGroupName

The name of the dataset group.

Type: String

Length Constraints: Minimum length of 1. Maximum length of 63.

Pattern: `^[a-zA-Z][a-zA-Z0-9_]*`

Domain

The domain associated with the dataset group.

Type: String

Valid Values: `RETAIL | CUSTOM | INVENTORY_PLANNING | EC2_CAPACITY | WORK_FORCE | WEB_TRAFFIC | METRICS`

LastModificationTime

When the dataset group was created or last updated from a call to the UpdateDatasetGroup operation. While the dataset group is being updated, `LastModificationTime` is the current time of the `DescribeDatasetGroup` call.

Type: Timestamp

Status

The status of the dataset group. States include:

- `ACTIVE`
- `CREATE_PENDING`, `CREATE_IN_PROGRESS`, `CREATE_FAILED`
- `DELETE_PENDING`, `DELETE_IN_PROGRESS`, `DELETE_FAILED`
- `UPDATE_PENDING`, `UPDATE_IN_PROGRESS`, `UPDATE_FAILED`

The `UPDATE` states apply when you call the UpdateDatasetGroup operation.

> **Note**
>
> The `Status` of the dataset group must be `ACTIVE` before you can use the dataset group to create a predictor.

Type: String

Length Constraints: Maximum length of 256.

Errors

InvalidInputException

We can't process the request because it includes an invalid value or a value that exceeds the valid range.

HTTP Status Code: 400

ResourceNotFoundException

We can't find a resource with that Amazon Resource Name (ARN). Check the ARN and try again.

HTTP Status Code: 400

See Also

For more information about using this API in one of the language-specific AWS SDKs, see the following:

- AWS Command Line Interface
- AWS SDK for .NET
- AWS SDK for C++
- AWS SDK for Go v2
- AWS SDK for Java V2
- AWS SDK for JavaScript V3
- AWS SDK for PHP V3
- AWS SDK for Python
- AWS SDK for Ruby V3

DescribeDatasetImportJob

Service: Amazon Forecast Service

Describes a dataset import job created using the CreateDatasetImportJob operation.

In addition to listing the parameters provided in the `CreateDatasetImportJob` request, this operation includes the following properties:

- `CreationTime`
- `LastModificationTime`
- `DataSize`
- `FieldStatistics`
- `Status`
- `Message` - If an error occurred, information about the error.

Request Syntax

```
{
   "DatasetImportJobArn": "string"
}
```

Request Parameters

The request accepts the following data in JSON format.

DatasetImportJobArn

The Amazon Resource Name (ARN) of the dataset import job.

Type: String

Length Constraints: Maximum length of 256.

Pattern: `arn:([a-z\d-]+):forecast:.*:.*:.+`

Required: Yes

Response Syntax

```
{
```

```
    "CreationTime": number,
    "DatasetArn": "string",
    "DatasetImportJobArn": "string",
    "DatasetImportJobName": "string",
    "DataSize": number,
    "DataSource": {
       "S3Config": {
          "KMSKeyArn": "string",
          "Path": "string",
          "RoleArn": "string"
       }
    },
    "EstimatedTimeRemainingInMinutes": number,
    "FieldStatistics": {
       "string" : {
          "Avg": number,
          "Count": number,
          "CountDistinct": number,
          "CountDistinctLong": number,
          "CountLong": number,
          "CountNan": number,
          "CountNanLong": number,
          "CountNull": number,
          "CountNullLong": number,
          "Max": "string",
          "Min": "string",
          "Stddev": number
       }
    },
    "Format": "string",
    "GeolocationFormat": "string",
    "ImportMode": "string",
    "LastModificationTime": number,
    "Message": "string",
    "Status": "string",
    "TimestampFormat": "string",
    "TimeZone": "string",
    "UseGeolocationForTimeZone": boolean
}
```

Response Elements

If the action is successful, the service sends back an HTTP 200 response.

The following data is returned in JSON format by the service.

CreationTime

When the dataset import job was created.

Type: Timestamp

DatasetArn

The Amazon Resource Name (ARN) of the dataset that the training data was imported to.

Type: String

Length Constraints: Maximum length of 256.

Pattern: `arn:([a-z\d-]+):forecast:.*:.*:.+`

DatasetImportJobArn

The ARN of the dataset import job.

Type: String

Length Constraints: Maximum length of 256.

Pattern: `arn:([a-z\d-]+):forecast:.*:.*:.+`

DatasetImportJobName

The name of the dataset import job.

Type: String

Length Constraints: Minimum length of 1. Maximum length of 63.

Pattern: `^[a-zA-Z][a-zA-Z0-9_]*`

DataSize

The size of the dataset in gigabytes (GB) after the import job has finished.

Type: Double

DataSource

The location of the training data to import and an AWS Identity and Access Management (IAM) role that Amazon Forecast can assume to access the data.

If encryption is used, `DataSource` includes an AWS Key Management Service (KMS) key.

Type: DataSource object

EstimatedTimeRemainingInMinutes

The estimated time remaining in minutes for the dataset import job to complete.

Type: Long

FieldStatistics

Statistical information about each field in the input data.

Type: String to Statistics object map

Key Length Constraints: Maximum length of 256.

Key Pattern: `^[a-zA-Z0-9\_]+$`

Format

The format of the imported data, CSV or PARQUET.

Type: String

Length Constraints: Maximum length of 7.

Pattern: `^CSV|PARQUET$`

GeolocationFormat

The format of the geolocation attribute. Valid Values:`"LAT_LONG"` and `"CC_POSTALCODE"`.

Type: String

Length Constraints: Maximum length of 256.

Pattern: `^[a-zA-Z0-9_]+$`

ImportMode

The import mode of the dataset import job, FULL or INCREMENTAL.

Type: String

Valid Values: `FULL | INCREMENTAL`

LastModificationTime

The last time the resource was modified. The timestamp depends on the status of the job:

- `CREATE_PENDING` - The `CreationTime`.
- `CREATE_IN_PROGRESS` - The current timestamp.
- `CREATE_STOPPING` - The current timestamp.
- `CREATE_STOPPED` - When the job stopped.
- `ACTIVE` or `CREATE_FAILED` - When the job finished or failed.

Type: Timestamp

Message

If an error occurred, an informational message about the error.

Type: String

Status

The status of the dataset import job. States include:

- `ACTIVE`
- `CREATE_PENDING`, `CREATE_IN_PROGRESS`, `CREATE_FAILED`
- `DELETE_PENDING`, `DELETE_IN_PROGRESS`, `DELETE_FAILED`
- `CREATE_STOPPING`, `CREATE_STOPPED`

Type: String

Length Constraints: Maximum length of 256.

TimestampFormat

The format of timestamps in the dataset. The format that you specify depends on the `DataFrequency` specified when the dataset was created. The following formats are supported

- "yyyy-MM-dd"

  For the following data frequencies: Y, M, W, and D

- "yyyy-MM-dd HH:mm:ss"

  For the following data frequencies: H, 30min, 15min, and 1min; and optionally, for: Y, M, W, and D

Type: String

Length Constraints: Maximum length of 256.

Pattern: `^[a-zA-Z0-9\-\:\.\,\'\s]+$`

TimeZone

The single time zone applied to every item in the dataset

Type: String

Length Constraints: Maximum length of 256.

Pattern: `^[a-zA-Z0-9\/\+\-\_]+$`

UseGeolocationForTimeZone

Whether `TimeZone` is automatically derived from the geolocation attribute.

Type: Boolean

Errors

InvalidInputException

We can't process the request because it includes an invalid value or a value that exceeds the valid range.

HTTP Status Code: 400

ResourceNotFoundException

We can't find a resource with that Amazon Resource Name (ARN). Check the ARN and try again.

HTTP Status Code: 400

See Also

For more information about using this API in one of the language-specific AWS SDKs, see the following:

- AWS Command Line Interface
- AWS SDK for .NET

- AWS SDK for C++
- AWS SDK for Go v2
- AWS SDK for Java V2
- AWS SDK for JavaScript V3
- AWS SDK for PHP V3
- AWS SDK for Python
- AWS SDK for Ruby V3

DescribeExplainability

Service: Amazon Forecast Service

Describes an Explainability resource created using the CreateExplainability operation.

Request Syntax

```
{
   "ExplainabilityArn": "string"
}
```

Request Parameters

The request accepts the following data in JSON format.

ExplainabilityArn

The Amazon Resource Name (ARN) of the Explaianability to describe.

Type: String

Length Constraints: Maximum length of 256.

Pattern: `arn:([a-z\d-]+):forecast:.*:.*:.+`

Required: Yes

Response Syntax

```
{
   "CreationTime": number,
   "DataSource": {
      "S3Config": {
         "KMSKeyArn": "string",
         "Path": "string",
         "RoleArn": "string"
      }
   },
   "EnableVisualization": boolean,
   "EndDateTime": "string",
   "EstimatedTimeRemainingInMinutes": number,
   "ExplainabilityArn": "string",
```

```
   "ExplainabilityConfig": {
      "TimePointGranularity": "string",
      "TimeSeriesGranularity": "string"
   },
   "ExplainabilityName": "string",
   "LastModificationTime": number,
   "Message": "string",
   "ResourceArn": "string",
   "Schema": {
      "Attributes": [
         {
            "AttributeName": "string",
            "AttributeType": "string"
         }
      ]
   },
   "StartDateTime": "string",
   "Status": "string"
}
```

Response Elements

If the action is successful, the service sends back an HTTP 200 response.

The following data is returned in JSON format by the service.

CreationTime

When the Explainability resource was created.

Type: Timestamp

DataSource

The source of your data, an AWS Identity and Access Management (IAM) role that allows Amazon Forecast to access the data and, optionally, an AWS Key Management Service (KMS) key.

Type: DataSource object

EnableVisualization

Whether the visualization was enabled for the Explainability resource.

Type: Boolean

EndDateTime

If `TimePointGranularity` is set to `SPECIFIC`, the last time point in the Explainability.

Type: String

Length Constraints: Maximum length of 19.

Pattern: `^\d{4}-\d{2}-\d{2}T\d{2}:\d{2}:\d{2}$`

EstimatedTimeRemainingInMinutes

The estimated time remaining in minutes for the CreateExplainability job to complete.

Type: Long

ExplainabilityArn

The Amazon Resource Name (ARN) of the Explainability.

Type: String

Length Constraints: Maximum length of 256.

Pattern: `arn:([a-z\d-]+):forecast:.*:.*:.+`

ExplainabilityConfig

The configuration settings that define the granularity of time series and time points for the Explainability.

Type: ExplainabilityConfig object

ExplainabilityName

The name of the Explainability.

Type: String

Length Constraints: Minimum length of 1. Maximum length of 63.

Pattern: `^[a-zA-Z][a-zA-Z0-9_]*`

LastModificationTime

The last time the resource was modified. The timestamp depends on the status of the job:

- CREATE_PENDING - The CreationTime.
- CREATE_IN_PROGRESS - The current timestamp.
- CREATE_STOPPING - The current timestamp.
- CREATE_STOPPED - When the job stopped.
- ACTIVE or CREATE_FAILED - When the job finished or failed.

Type: Timestamp

Message

If an error occurred, a message about the error.

Type: String

ResourceArn

The Amazon Resource Name (ARN) of the Predictor or Forecast used to create the Explainability resource.

Type: String

Length Constraints: Maximum length of 256.

Pattern: arn:([a-z\d-]+):forecast:.*:.*:.+

Schema

Defines the fields of a dataset.

Type: Schema object

StartDateTime

If TimePointGranularity is set to SPECIFIC, the first time point in the Explainability.

Type: String

Length Constraints: Maximum length of 19.

Pattern: ^\d{4}-\d{2}-\d{2}T\d{2}:\d{2}:\d{2}$

Status

The status of the Explainability resource. States include:

- ACTIVE
- CREATE_PENDING, CREATE_IN_PROGRESS, CREATE_FAILED
- CREATE_STOPPING, CREATE_STOPPED
- DELETE_PENDING, DELETE_IN_PROGRESS, DELETE_FAILED

Type: String

Length Constraints: Maximum length of 256.

Errors

InvalidInputException

We can't process the request because it includes an invalid value or a value that exceeds the valid range.

HTTP Status Code: 400

ResourceNotFoundException

We can't find a resource with that Amazon Resource Name (ARN). Check the ARN and try again.

HTTP Status Code: 400

See Also

For more information about using this API in one of the language-specific AWS SDKs, see the following:

- AWS Command Line Interface
- AWS SDK for .NET
- AWS SDK for C++
- AWS SDK for Go v2
- AWS SDK for Java V2
- AWS SDK for JavaScript V3
- AWS SDK for PHP V3
- AWS SDK for Python
- AWS SDK for Ruby V3

DescribeExplainabilityExport

Service: Amazon Forecast Service

Describes an Explainability export created using the CreateExplainabilityExport operation.

Request Syntax

```
{
   "ExplainabilityExportArn": "string"
}
```

Request Parameters

The request accepts the following data in JSON format.

ExplainabilityExportArn

The Amazon Resource Name (ARN) of the Explainability export.

Type: String

Length Constraints: Maximum length of 256.

Pattern: `arn:([a-z\d-]+):forecast:.*:.*:.+`

Required: Yes

Response Syntax

```
{
   "CreationTime": number,
   "Destination": {
      "S3Config": {
         "KMSKeyArn": "string",
         "Path": "string",
         "RoleArn": "string"
      }
   },
   "ExplainabilityArn": "string",
   "ExplainabilityExportArn": "string",
   "ExplainabilityExportName": "string",
   "Format": "string",
```

```
   "LastModificationTime": number,
   "Message": "string",
   "Status": "string"
}
```

Response Elements

If the action is successful, the service sends back an HTTP 200 response.

The following data is returned in JSON format by the service.

CreationTime

When the Explainability export was created.

Type: Timestamp

Destination

The destination for an export job. Provide an S3 path, an AWS Identity and Access Management (IAM) role that allows Amazon Forecast to access the location, and an AWS Key Management Service (KMS) key (optional).

Type: DataDestination object

ExplainabilityArn

The Amazon Resource Name (ARN) of the Explainability export.

Type: String

Length Constraints: Maximum length of 256.

Pattern: `arn:([a-z\d-]+):forecast:.*:.*:.+`

ExplainabilityExportArn

The Amazon Resource Name (ARN) of the Explainability export.

Type: String

Length Constraints: Maximum length of 256.

Pattern: `arn:([a-z\d-]+):forecast:.*:.*:.+`

ExplainabilityExportName

The name of the Explainability export.

Type: String

Length Constraints: Minimum length of 1. Maximum length of 63.

Pattern: `^[a-zA-Z][a-zA-Z0-9_]*`

Format

The format of the exported data, CSV or PARQUET.

Type: String

Length Constraints: Maximum length of 7.

Pattern: `^CSV|PARQUET$`

LastModificationTime

The last time the resource was modified. The timestamp depends on the status of the job:

- `CREATE_PENDING` - The `CreationTime`.
- `CREATE_IN_PROGRESS` - The current timestamp.
- `CREATE_STOPPING` - The current timestamp.
- `CREATE_STOPPED` - When the job stopped.
- `ACTIVE` or `CREATE_FAILED` - When the job finished or failed.

Type: Timestamp

Message

Information about any errors that occurred during the export.

Type: String

Status

The status of the Explainability export. States include:

- `ACTIVE`
- `CREATE_PENDING`, `CREATE_IN_PROGRESS`, `CREATE_FAILED`
- `CREATE_STOPPING`, `CREATE_STOPPED`

- `DELETE_PENDING`, `DELETE_IN_PROGRESS`, `DELETE_FAILED`

Type: String

Length Constraints: Maximum length of 256.

Errors

InvalidInputException

We can't process the request because it includes an invalid value or a value that exceeds the valid range.

HTTP Status Code: 400

ResourceNotFoundException

We can't find a resource with that Amazon Resource Name (ARN). Check the ARN and try again.

HTTP Status Code: 400

See Also

For more information about using this API in one of the language-specific AWS SDKs, see the following:

- AWS Command Line Interface
- AWS SDK for .NET
- AWS SDK for C++
- AWS SDK for Go v2
- AWS SDK for Java V2
- AWS SDK for JavaScript V3
- AWS SDK for PHP V3
- AWS SDK for Python
- AWS SDK for Ruby V3

DescribeForecast

Service: Amazon Forecast Service

Describes a forecast created using the CreateForecast operation.

In addition to listing the properties provided in the `CreateForecast` request, this operation lists the following properties:

- `DatasetGroupArn` - The dataset group that provided the training data.
- `CreationTime`
- `LastModificationTime`
- `Status`
- `Message` - If an error occurred, information about the error.

Request Syntax

```
{
   "ForecastArn": "string"
}
```

Request Parameters

The request accepts the following data in JSON format.

ForecastArn

The Amazon Resource Name (ARN) of the forecast.

Type: String

Length Constraints: Maximum length of 256.

Pattern: `arn:([a-z\d-]+):forecast:.*:.*:.+`

Required: Yes

Response Syntax

```
{
```

```
   "CreationTime": number,
   "DatasetGroupArn": "string",
   "EstimatedTimeRemainingInMinutes": number,
   "ForecastArn": "string",
   "ForecastName": "string",
   "ForecastTypes": [ "string" ],
   "LastModificationTime": number,
   "Message": "string",
   "PredictorArn": "string",
   "Status": "string",
   "TimeSeriesSelector": {
      "TimeSeriesIdentifiers": {
         "DataSource": {
            "S3Config": {
               "KMSKeyArn": "string",
               "Path": "string",
               "RoleArn": "string"
            }
         },
         "Format": "string",
         "Schema": {
            "Attributes": [
               {
                  "AttributeName": "string",
                  "AttributeType": "string"
               }
            ]
         }
      }
   }
}
```

Response Elements

If the action is successful, the service sends back an HTTP 200 response.

The following data is returned in JSON format by the service.

CreationTime

When the forecast creation task was created.

Type: Timestamp

DatasetGroupArn

The ARN of the dataset group that provided the data used to train the predictor.

Type: String

Length Constraints: Maximum length of 256.

Pattern: `arn:([a-z\d-]+):forecast:.*:.*:.+`

EstimatedTimeRemainingInMinutes

The estimated time remaining in minutes for the forecast job to complete.

Type: Long

ForecastArn

The forecast ARN as specified in the request.

Type: String

Length Constraints: Maximum length of 256.

Pattern: `arn:([a-z\d-]+):forecast:.*:.*:.+`

ForecastName

The name of the forecast.

Type: String

Length Constraints: Minimum length of 1. Maximum length of 63.

Pattern: `^[a-zA-Z][a-zA-Z0-9_]*`

ForecastTypes

The quantiles at which probabilistic forecasts were generated.

Type: Array of strings

Array Members: Minimum number of 1 item. Maximum number of 20 items.

Length Constraints: Minimum length of 2. Maximum length of 4.

Pattern: `(^0?\.\d\d?$|^mean$)`

LastModificationTime

The last time the resource was modified. The timestamp depends on the status of the job:

- `CREATE_PENDING` - The `CreationTime`.
- `CREATE_IN_PROGRESS` - The current timestamp.
- `CREATE_STOPPING` - The current timestamp.
- `CREATE_STOPPED` - When the job stopped.
- `ACTIVE` or `CREATE_FAILED` - When the job finished or failed.

Type: Timestamp

Message

If an error occurred, an informational message about the error.

Type: String

PredictorArn

The ARN of the predictor used to generate the forecast.

Type: String

Length Constraints: Maximum length of 256.

Pattern: `arn:([a-z\d-]+):forecast:.*:.*:.+`

Status

The status of the forecast. States include:

- `ACTIVE`
- `CREATE_PENDING`, `CREATE_IN_PROGRESS`, `CREATE_FAILED`
- `CREATE_STOPPING`, `CREATE_STOPPED`
- `DELETE_PENDING`, `DELETE_IN_PROGRESS`, `DELETE_FAILED`

> **Note**
>
> The `Status` of the forecast must be `ACTIVE` before you can query or export the forecast.

Type: String

Length Constraints: Maximum length of 256.

Pattern: `^[a-zA-Z0-9\_]+$`

TimeSeriesSelector

The time series to include in the forecast.

Type: TimeSeriesSelector object

Errors

InvalidInputException

We can't process the request because it includes an invalid value or a value that exceeds the valid range.

HTTP Status Code: 400

ResourceNotFoundException

We can't find a resource with that Amazon Resource Name (ARN). Check the ARN and try again.

HTTP Status Code: 400

See Also

For more information about using this API in one of the language-specific AWS SDKs, see the following:

- AWS Command Line Interface
- AWS SDK for .NET
- AWS SDK for C++
- AWS SDK for Go v2
- AWS SDK for Java V2
- AWS SDK for JavaScript V3
- AWS SDK for PHP V3
- AWS SDK for Python

- AWS SDK for Ruby V3

DescribeForecastExportJob

Service: Amazon Forecast Service

Describes a forecast export job created using the CreateForecastExportJob operation.

In addition to listing the properties provided by the user in the `CreateForecastExportJob` request, this operation lists the following properties:

- `CreationTime`
- `LastModificationTime`
- `Status`
- `Message` - If an error occurred, information about the error.

Request Syntax

```
{
   "ForecastExportJobArn": "string"
}
```

Request Parameters

The request accepts the following data in JSON format.

ForecastExportJobArn

The Amazon Resource Name (ARN) of the forecast export job.

Type: String

Length Constraints: Maximum length of 256.

Pattern: `arn:([a-z\d-]+):forecast:.*:.*:.+`

Required: Yes

Response Syntax

```
{
   "CreationTime": number,
```

```
   "Destination": {
      "S3Config": {
         "KMSKeyArn": "string",
         "Path": "string",
         "RoleArn": "string"
      }
   },
   "ForecastArn": "string",
   "ForecastExportJobArn": "string",
   "ForecastExportJobName": "string",
   "Format": "string",
   "LastModificationTime": number,
   "Message": "string",
   "Status": "string"
}
```

Response Elements

If the action is successful, the service sends back an HTTP 200 response.

The following data is returned in JSON format by the service.

CreationTime

When the forecast export job was created.

Type: Timestamp

Destination

The path to the Amazon Simple Storage Service (Amazon S3) bucket where the forecast is exported.

Type: DataDestination object

ForecastArn

The Amazon Resource Name (ARN) of the exported forecast.

Type: String

Length Constraints: Maximum length of 256.

Pattern: `arn:([a-z\d-]+):forecast:.*:.*:.+`

ForecastExportJobArn

The ARN of the forecast export job.

Type: String

Length Constraints: Maximum length of 256.

Pattern: `arn:([a-z\d-]+):forecast:.*:.*:.+`

ForecastExportJobName

The name of the forecast export job.

Type: String

Length Constraints: Minimum length of 1. Maximum length of 63.

Pattern: `^[a-zA-Z][a-zA-Z0-9_]*`

Format

The format of the exported data, CSV or PARQUET.

Type: String

Length Constraints: Maximum length of 7.

Pattern: `^CSV|PARQUET$`

LastModificationTime

The last time the resource was modified. The timestamp depends on the status of the job:

- `CREATE_PENDING` - The `CreationTime`.
- `CREATE_IN_PROGRESS` - The current timestamp.
- `CREATE_STOPPING` - The current timestamp.
- `CREATE_STOPPED` - When the job stopped.
- `ACTIVE` or `CREATE_FAILED` - When the job finished or failed.

Type: Timestamp

Message

If an error occurred, an informational message about the error.

Type: String

Status

The status of the forecast export job. States include:

- `ACTIVE`
- `CREATE_PENDING`, `CREATE_IN_PROGRESS`, `CREATE_FAILED`
- `CREATE_STOPPING`, `CREATE_STOPPED`
- `DELETE_PENDING`, `DELETE_IN_PROGRESS`, `DELETE_FAILED`

> **Note**
>
> The `Status` of the forecast export job must be `ACTIVE` before you can access the forecast in your S3 bucket.

Type: String

Length Constraints: Maximum length of 256.

Errors

InvalidInputException

We can't process the request because it includes an invalid value or a value that exceeds the valid range.

HTTP Status Code: 400

ResourceNotFoundException

We can't find a resource with that Amazon Resource Name (ARN). Check the ARN and try again.

HTTP Status Code: 400

See Also

For more information about using this API in one of the language-specific AWS SDKs, see the following:

- AWS Command Line Interface
- AWS SDK for .NET
- AWS SDK for C++
- AWS SDK for Go v2
- AWS SDK for Java V2
- AWS SDK for JavaScript V3
- AWS SDK for PHP V3
- AWS SDK for Python
- AWS SDK for Ruby V3

DescribeMonitor

Service: Amazon Forecast Service

Describes a monitor resource. In addition to listing the properties provided in the CreateMonitor request, this operation lists the following properties:

- `Baseline`
- `CreationTime`
- `LastEvaluationTime`
- `LastEvaluationState`
- `LastModificationTime`
- `Message`
- `Status`

Request Syntax

```
{
   "MonitorArn": "string"
}
```

Request Parameters

The request accepts the following data in JSON format.

MonitorArn

The Amazon Resource Name (ARN) of the monitor resource to describe.

Type: String

Length Constraints: Maximum length of 256.

Pattern: `arn:([a-z\d-]+):forecast:.*:.*:.+`

Required: Yes

Response Syntax

```
{
```

```
   "Baseline": {
      "PredictorBaseline": {
         "BaselineMetrics": [
            {
               "Name": "string",
               "Value": number
            }
         ]
      }
   },
   "CreationTime": number,
   "EstimatedEvaluationTimeRemainingInMinutes": number,
   "LastEvaluationState": "string",
   "LastEvaluationTime": number,
   "LastModificationTime": number,
   "Message": "string",
   "MonitorArn": "string",
   "MonitorName": "string",
   "ResourceArn": "string",
   "Status": "string"
}
```

Response Elements

If the action is successful, the service sends back an HTTP 200 response.

The following data is returned in JSON format by the service.

Baseline

Metrics you can use as a baseline for comparison purposes. Use these values you interpret monitoring results for an auto predictor.

Type: Baseline object

CreationTime

The timestamp for when the monitor resource was created.

Type: Timestamp

EstimatedEvaluationTimeRemainingInMinutes

The estimated number of minutes remaining before the monitor resource finishes its current evaluation.

Type: Long

LastEvaluationState

The state of the monitor's latest evaluation.

Type: String

Length Constraints: Maximum length of 256.

LastEvaluationTime

The timestamp of the latest evaluation completed by the monitor.

Type: Timestamp

LastModificationTime

The timestamp of the latest modification to the monitor.

Type: Timestamp

Message

An error message, if any, for the monitor.

Type: String

MonitorArn

The Amazon Resource Name (ARN) of the monitor resource described.

Type: String

Length Constraints: Maximum length of 256.

Pattern: `arn:([a-z\d-]+):forecast:.*:.*:.+`

MonitorName

The name of the monitor.

Type: String

Length Constraints: Minimum length of 1. Maximum length of 63.

Pattern: `^[a-zA-Z][a-zA-Z0-9_]*`

ResourceArn

The Amazon Resource Name (ARN) of the auto predictor being monitored.

Type: String

Length Constraints: Maximum length of 256.

Pattern: `arn:([a-z\d-]+):forecast:.*:.*:.+`

Status

The status of the monitor resource.

Type: String

Length Constraints: Maximum length of 256.

Errors

InvalidInputException

We can't process the request because it includes an invalid value or a value that exceeds the valid range.

HTTP Status Code: 400

ResourceNotFoundException

We can't find a resource with that Amazon Resource Name (ARN). Check the ARN and try again.

HTTP Status Code: 400

See Also

For more information about using this API in one of the language-specific AWS SDKs, see the following:

- AWS Command Line Interface
- AWS SDK for .NET
- AWS SDK for C++
- AWS SDK for Go v2

- AWS SDK for Java V2
- AWS SDK for JavaScript V3
- AWS SDK for PHP V3
- AWS SDK for Python
- AWS SDK for Ruby V3

DescribePredictor

Service: Amazon Forecast Service

> **Note**
>
> This operation is only valid for legacy predictors created with CreatePredictor. If you are not using a legacy predictor, use DescribeAutoPredictor.

Describes a predictor created using the CreatePredictor operation.

In addition to listing the properties provided in the `CreatePredictor` request, this operation lists the following properties:

- `DatasetImportJobArns` - The dataset import jobs used to import training data.
- `AutoMLAlgorithmArns` - If AutoML is performed, the algorithms that were evaluated.
- `CreationTime`
- `LastModificationTime`
- `Status`
- `Message` - If an error occurred, information about the error.

Request Syntax

```
{
   "PredictorArn": "string"
}
```

Request Parameters

The request accepts the following data in JSON format.

PredictorArn

The Amazon Resource Name (ARN) of the predictor that you want information about.

Type: String

Length Constraints: Maximum length of 256.

Pattern: `arn:([a-z\d-]+):forecast:.*:.*:.+`

Required: Yes

Response Syntax

```
{
   "AlgorithmArn": "string",
   "AutoMLAlgorithmArns": [ "string" ],
   "AutoMLOverrideStrategy": "string",
   "CreationTime": number,
   "DatasetImportJobArns": [ "string" ],
   "EncryptionConfig": {
      "KMSKeyArn": "string",
      "RoleArn": "string"
   },
   "EstimatedTimeRemainingInMinutes": number,
   "EvaluationParameters": {
      "BackTestWindowOffset": number,
      "NumberOfBacktestWindows": number
   },
   "FeaturizationConfig": {
      "Featurizations": [
         {
            "AttributeName": "string",
            "FeaturizationPipeline": [
               {
                  "FeaturizationMethodName": "string",
                  "FeaturizationMethodParameters": {
                     "string" : "string"
                  }
               }
            ]
         }
      ],
      "ForecastDimensions": [ "string" ],
      "ForecastFrequency": "string"
   },
   "ForecastHorizon": number,
   "ForecastTypes": [ "string" ],
   "HPOConfig": {
      "ParameterRanges": {
         "CategoricalParameterRanges": [
```

```
            {
               "Name": "string",
               "Values": [ "string" ]
            }
         ],
         "ContinuousParameterRanges": [
            {
               "MaxValue": number,
               "MinValue": number,
               "Name": "string",
               "ScalingType": "string"
            }
         ],
         "IntegerParameterRanges": [
            {
               "MaxValue": number,
               "MinValue": number,
               "Name": "string",
               "ScalingType": "string"
            }
         ]
      }
   },
   "InputDataConfig": {
      "DatasetGroupArn": "string",
      "SupplementaryFeatures": [
         {
            "Name": "string",
            "Value": "string"
         }
      ]
   },
   "IsAutoPredictor": boolean,
   "LastModificationTime": number,
   "Message": "string",
   "OptimizationMetric": "string",
   "PerformAutoML": boolean,
   "PerformHPO": boolean,
   "PredictorArn": "string",
   "PredictorExecutionDetails": {
      "PredictorExecutions": [
         {
            "AlgorithmArn": "string",
            "TestWindows": [
```

```
                {
                   "Message": "string",
                   "Status": "string",
                   "TestWindowEnd": number,
                   "TestWindowStart": number
                }
             ]
          }
       ]
    },
    "PredictorName": "string",
    "Status": "string",
    "TrainingParameters": {
       "string" : "string"
    }
}
```

Response Elements

If the action is successful, the service sends back an HTTP 200 response.

The following data is returned in JSON format by the service.

AlgorithmArn

The Amazon Resource Name (ARN) of the algorithm used for model training.

Type: String

Length Constraints: Maximum length of 256.

Pattern: `arn:([a-z\d-]+):forecast:.*:.*:.+`

AutoMLAlgorithmArns

When `PerformAutoML` is specified, the ARN of the chosen algorithm.

Type: Array of strings

Length Constraints: Maximum length of 256.

Pattern: `arn:([a-z\d-]+):forecast:.*:.*:.+`

AutoMLOverrideStrategy

> **Note**
>
> The `LatencyOptimized` AutoML override strategy is only available in private beta. Contact AWS Support or your account manager to learn more about access privileges.

The AutoML strategy used to train the predictor. Unless `LatencyOptimized` is specified, the AutoML strategy optimizes predictor accuracy.

This parameter is only valid for predictors trained using AutoML.

Type: String

Valid Values: `LatencyOptimized | AccuracyOptimized`

CreationTime

When the model training task was created.

Type: Timestamp

DatasetImportJobArns

An array of the ARNs of the dataset import jobs used to import training data for the predictor.

Type: Array of strings

Length Constraints: Maximum length of 256.

Pattern: `arn:([a-z\d-]+):forecast:.*:.*:.+`

EncryptionConfig

An AWS Key Management Service (KMS) key and the AWS Identity and Access Management (IAM) role that Amazon Forecast can assume to access the key.

Type: EncryptionConfig object

EstimatedTimeRemainingInMinutes

The estimated time remaining in minutes for the predictor training job to complete.

Type: Long

EvaluationParameters

Used to override the default evaluation parameters of the specified algorithm. Amazon Forecast evaluates a predictor by splitting a dataset into training data and testing data. The evaluation parameters define how to perform the split and the number of iterations.

Type: EvaluationParameters object

FeaturizationConfig

The featurization configuration.

Type: FeaturizationConfig object

ForecastHorizon

The number of time-steps of the forecast. The forecast horizon is also called the prediction length.

Type: Integer

ForecastTypes

The forecast types used during predictor training. Default value is `["0.1","0.5","0.9"]`

Type: Array of strings

Array Members: Minimum number of 1 item. Maximum number of 20 items.

Length Constraints: Minimum length of 2. Maximum length of 4.

Pattern: `(^0?\.\d\d?$|^mean$)`

HPOConfig

The hyperparameter override values for the algorithm.

Type: HyperParameterTuningJobConfig object

InputDataConfig

Describes the dataset group that contains the data to use to train the predictor.

Type: InputDataConfig object

IsAutoPredictor

Whether the predictor was created with CreateAutoPredictor.

Type: Boolean

LastModificationTime

The last time the resource was modified. The timestamp depends on the status of the job:

- `CREATE_PENDING` - The `CreationTime`.
- `CREATE_IN_PROGRESS` - The current timestamp.
- `CREATE_STOPPING` - The current timestamp.
- `CREATE_STOPPED` - When the job stopped.
- `ACTIVE` or `CREATE_FAILED` - When the job finished or failed.

Type: Timestamp

Message

If an error occurred, an informational message about the error.

Type: String

OptimizationMetric

The accuracy metric used to optimize the predictor.

Type: String

Valid Values: `WAPE | RMSE | AverageWeightedQuantileLoss | MASE | MAPE`

PerformAutoML

Whether the predictor is set to perform AutoML.

Type: Boolean

PerformHPO

Whether the predictor is set to perform hyperparameter optimization (HPO).

Type: Boolean

PredictorArn

The ARN of the predictor.

Type: String

Length Constraints: Minimum length of 1. Maximum length of 63.

Pattern: `^[a-zA-Z][a-zA-Z0-9_]*`

PredictorExecutionDetails

Details on the the status and results of the backtests performed to evaluate the accuracy of the predictor. You specify the number of backtests to perform when you call the CreatePredictor operation.

Type: PredictorExecutionDetails object

PredictorName

The name of the predictor.

Type: String

Length Constraints: Minimum length of 1. Maximum length of 63.

Pattern: `^[a-zA-Z][a-zA-Z0-9_]*`

Status

The status of the predictor. States include:

- `ACTIVE`
- `CREATE_PENDING`, `CREATE_IN_PROGRESS`, `CREATE_FAILED`
- `DELETE_PENDING`, `DELETE_IN_PROGRESS`, `DELETE_FAILED`
- `CREATE_STOPPING`, `CREATE_STOPPED`

> **Note**
>
> The `Status` of the predictor must be `ACTIVE` before you can use the predictor to create a forecast.

Type: String

Length Constraints: Maximum length of 256.

TrainingParameters

The default training parameters or overrides selected during model training. When running AutoML or choosing HPO with CNN-QR or DeepAR+, the optimized values for the chosen hyperparameters are returned. For more information, see Amazon Forecast Algorithms.

Type: String to string map

Map Entries: Minimum number of 0 items. Maximum number of 100 items.

Key Length Constraints: Maximum length of 256.

Key Pattern: `^[a-zA-Z0-9\-\_\.\/\[\]\,\\]+$`

Value Length Constraints: Maximum length of 256.

Value Pattern: `^[a-zA-Z0-9\-\_\.\/\[\]\,\"\\\s]+$`

Errors

InvalidInputException

We can't process the request because it includes an invalid value or a value that exceeds the valid range.

HTTP Status Code: 400

ResourceNotFoundException

We can't find a resource with that Amazon Resource Name (ARN). Check the ARN and try again.

HTTP Status Code: 400

See Also

For more information about using this API in one of the language-specific AWS SDKs, see the following:

- AWS Command Line Interface
- AWS SDK for .NET
- AWS SDK for C++
- AWS SDK for Go v2
- AWS SDK for Java V2
- AWS SDK for JavaScript V3
- AWS SDK for PHP V3
- AWS SDK for Python

- AWS SDK for Ruby V3

DescribePredictorBacktestExportJob

Service: Amazon Forecast Service

Describes a predictor backtest export job created using the CreatePredictorBacktestExportJob operation.

In addition to listing the properties provided by the user in the `CreatePredictorBacktestExportJob` request, this operation lists the following properties:

- `CreationTime`
- `LastModificationTime`
- `Status`
- `Message` (if an error occurred)

Request Syntax

```
{
   "PredictorBacktestExportJobArn": "string"
}
```

Request Parameters

The request accepts the following data in JSON format.

PredictorBacktestExportJobArn

The Amazon Resource Name (ARN) of the predictor backtest export job.

Type: String

Length Constraints: Maximum length of 256.

Pattern: `arn:([a-z\d-]+):forecast:.*:.*:.+`

Required: Yes

Response Syntax

```
{
```

```
   "CreationTime": number,
   "Destination": {
      "S3Config": {
         "KMSKeyArn": "string",
         "Path": "string",
         "RoleArn": "string"
      }
   },
   "Format": "string",
   "LastModificationTime": number,
   "Message": "string",
   "PredictorArn": "string",
   "PredictorBacktestExportJobArn": "string",
   "PredictorBacktestExportJobName": "string",
   "Status": "string"
}
```

Response Elements

If the action is successful, the service sends back an HTTP 200 response.

The following data is returned in JSON format by the service.

CreationTime

When the predictor backtest export job was created.

Type: Timestamp

Destination

The destination for an export job. Provide an S3 path, an AWS Identity and Access Management (IAM) role that allows Amazon Forecast to access the location, and an AWS Key Management Service (KMS) key (optional).

Type: DataDestination object

Format

The format of the exported data, CSV or PARQUET.

Type: String

Length Constraints: Maximum length of 7.

Pattern: ^CSV|PARQUET$

LastModificationTime

The last time the resource was modified. The timestamp depends on the status of the job:

- CREATE_PENDING - The CreationTime.
- CREATE_IN_PROGRESS - The current timestamp.
- CREATE_STOPPING - The current timestamp.
- CREATE_STOPPED - When the job stopped.
- ACTIVE or CREATE_FAILED - When the job finished or failed.

Type: Timestamp

Message

Information about any errors that may have occurred during the backtest export.

Type: String

PredictorArn

The Amazon Resource Name (ARN) of the predictor.

Type: String

Length Constraints: Maximum length of 256.

Pattern: arn:([a-z\d-]+):forecast:.*:.*:.+

PredictorBacktestExportJobArn

The Amazon Resource Name (ARN) of the predictor backtest export job.

Type: String

Length Constraints: Maximum length of 256.

Pattern: arn:([a-z\d-]+):forecast:.*:.*:.+

PredictorBacktestExportJobName

The name of the predictor backtest export job.

Type: String

Length Constraints: Minimum length of 1. Maximum length of 63.

Pattern: `^[a-zA-Z][a-zA-Z0-9_]*`

Status

The status of the predictor backtest export job. States include:

- `ACTIVE`
- `CREATE_PENDING`, `CREATE_IN_PROGRESS`, `CREATE_FAILED`
- `CREATE_STOPPING`, `CREATE_STOPPED`
- `DELETE_PENDING`, `DELETE_IN_PROGRESS`, `DELETE_FAILED`

Type: String

Length Constraints: Maximum length of 256.

Errors

InvalidInputException

We can't process the request because it includes an invalid value or a value that exceeds the valid range.

HTTP Status Code: 400

ResourceNotFoundException

We can't find a resource with that Amazon Resource Name (ARN). Check the ARN and try again.

HTTP Status Code: 400

See Also

For more information about using this API in one of the language-specific AWS SDKs, see the following:

- AWS Command Line Interface
- AWS SDK for .NET
- AWS SDK for C++
- AWS SDK for Go v2

- AWS SDK for Java V2
- AWS SDK for JavaScript V3
- AWS SDK for PHP V3
- AWS SDK for Python
- AWS SDK for Ruby V3

DescribeWhatIfAnalysis

Service: Amazon Forecast Service

Describes the what-if analysis created using the CreateWhatIfAnalysis operation.

In addition to listing the properties provided in the `CreateWhatIfAnalysis` request, this operation lists the following properties:

- `CreationTime`
- `LastModificationTime`
- `Message` - If an error occurred, information about the error.
- `Status`

Request Syntax

```
{
   "WhatIfAnalysisArn": "string"
}
```

Request Parameters

The request accepts the following data in JSON format.

WhatIfAnalysisArn

The Amazon Resource Name (ARN) of the what-if analysis that you are interested in.

Type: String

Length Constraints: Maximum length of 256.

Pattern: `arn:([a-z\d-]+):forecast:.*:.*:.+`

Required: Yes

Response Syntax

```
{
   "CreationTime": number,
```

```
   "EstimatedTimeRemainingInMinutes": number,
   "ForecastArn": "string",
   "LastModificationTime": number,
   "Message": "string",
   "Status": "string",
   "TimeSeriesSelector": {
      "TimeSeriesIdentifiers": {
         "DataSource": {
            "S3Config": {
               "KMSKeyArn": "string",
               "Path": "string",
               "RoleArn": "string"
            }
         },
         "Format": "string",
         "Schema": {
            "Attributes": [
               {
                  "AttributeName": "string",
                  "AttributeType": "string"
               }
            ]
         }
      }
   },
   "WhatIfAnalysisArn": "string",
   "WhatIfAnalysisName": "string"
}
```

Response Elements

If the action is successful, the service sends back an HTTP 200 response.

The following data is returned in JSON format by the service.

CreationTime

When the what-if analysis was created.

Type: Timestamp

EstimatedTimeRemainingInMinutes

The approximate time remaining to complete the what-if analysis, in minutes.

Type: Long

ForecastArn

The Amazon Resource Name (ARN) of the what-if forecast.

Type: String

Length Constraints: Maximum length of 256.

Pattern: `arn:([a-z\d-]+):forecast:.*:.*:.+`

LastModificationTime

The last time the resource was modified. The timestamp depends on the status of the job:

- `CREATE_PENDING` - The `CreationTime`.
- `CREATE_IN_PROGRESS` - The current timestamp.
- `CREATE_STOPPING` - The current timestamp.
- `CREATE_STOPPED` - When the job stopped.
- `ACTIVE` or `CREATE_FAILED` - When the job finished or failed.

Type: Timestamp

Message

If an error occurred, an informational message about the error.

Type: String

Status

The status of the what-if analysis. States include:

- `ACTIVE`
- `CREATE_PENDING`, `CREATE_IN_PROGRESS`, `CREATE_FAILED`
- `CREATE_STOPPING`, `CREATE_STOPPED`
- `DELETE_PENDING`, `DELETE_IN_PROGRESS`, `DELETE_FAILED`

> **Note**
>
> The `Status` of the what-if analysis must be `ACTIVE` before you can access the analysis.

Type: String

Length Constraints: Maximum length of 256.

Pattern: `^[a-zA-Z0-9\_]+$`

TimeSeriesSelector

Defines the set of time series that are used to create the forecasts in a `TimeSeriesIdentifiers` object.

The `TimeSeriesIdentifiers` object needs the following information:

- `DataSource`
- `Format`
- `Schema`

Type: TimeSeriesSelector object

WhatIfAnalysisArn

The Amazon Resource Name (ARN) of the what-if analysis.

Type: String

Length Constraints: Maximum length of 256.

Pattern: `arn:([a-z\d-]+):forecast:.*:.*:.+`

WhatIfAnalysisName

The name of the what-if analysis.

Type: String

Length Constraints: Minimum length of 1. Maximum length of 63.

Pattern: `^[a-zA-Z][a-zA-Z0-9_]*`

Errors

InvalidInputException

We can't process the request because it includes an invalid value or a value that exceeds the valid range.

HTTP Status Code: 400

ResourceNotFoundException

We can't find a resource with that Amazon Resource Name (ARN). Check the ARN and try again.

HTTP Status Code: 400

See Also

For more information about using this API in one of the language-specific AWS SDKs, see the following:

- AWS Command Line Interface
- AWS SDK for .NET
- AWS SDK for C++
- AWS SDK for Go v2
- AWS SDK for Java V2
- AWS SDK for JavaScript V3
- AWS SDK for PHP V3
- AWS SDK for Python
- AWS SDK for Ruby V3

DescribeWhatIfForecast

Service: Amazon Forecast Service

Describes the what-if forecast created using the CreateWhatIfForecast operation.

In addition to listing the properties provided in the `CreateWhatIfForecast` request, this operation lists the following properties:

- `CreationTime`
- `LastModificationTime`
- `Message` - If an error occurred, information about the error.
- `Status`

Request Syntax

```
{
   "WhatIfForecastArn": "string"
}
```

Request Parameters

The request accepts the following data in JSON format.

WhatIfForecastArn

The Amazon Resource Name (ARN) of the what-if forecast that you are interested in.

Type: String

Length Constraints: Maximum length of 300.

Pattern: `arn:([a-z\d-]+):forecast:.*:.*:.+`

Required: Yes

Response Syntax

```
{
   "CreationTime": number,
   "EstimatedTimeRemainingInMinutes": number,
   "ForecastTypes": [ "string" ],
```

```
   "LastModificationTime": number,
   "Message": "string",
   "Status": "string",
   "TimeSeriesReplacementsDataSource": {
      "Format": "string",
      "S3Config": {
         "KMSKeyArn": "string",
         "Path": "string",
         "RoleArn": "string"
      },
      "Schema": {
         "Attributes": [
            {
               "AttributeName": "string",
               "AttributeType": "string"
            }
         ]
      },
      "TimestampFormat": "string"
   },
   "TimeSeriesTransformations": [
      {
         "Action": {
            "AttributeName": "string",
            "Operation": "string",
            "Value": number
         },
         "TimeSeriesConditions": [
            {
               "AttributeName": "string",
               "AttributeValue": "string",
               "Condition": "string"
            }
         ]
      }
   ],
   "WhatIfAnalysisArn": "string",
   "WhatIfForecastArn": "string",
   "WhatIfForecastName": "string"
}
```

Response Elements

If the action is successful, the service sends back an HTTP 200 response.

The following data is returned in JSON format by the service.

CreationTime

When the what-if forecast was created.

Type: Timestamp

EstimatedTimeRemainingInMinutes

The approximate time remaining to complete the what-if forecast, in minutes.

Type: Long

ForecastTypes

The quantiles at which probabilistic forecasts are generated. You can specify up to five quantiles per what-if forecast in the CreateWhatIfForecast operation. If you didn't specify quantiles, the default values are `["0.1", "0.5", "0.9"]`.

Type: Array of strings

Array Members: Minimum number of 1 item. Maximum number of 20 items.

Length Constraints: Minimum length of 2. Maximum length of 4.

Pattern: `(^0?\.\d\d?$|^mean$)`

LastModificationTime

The last time the resource was modified. The timestamp depends on the status of the job:

- `CREATE_PENDING` - The `CreationTime`.
- `CREATE_IN_PROGRESS` - The current timestamp.
- `CREATE_STOPPING` - The current timestamp.
- `CREATE_STOPPED` - When the job stopped.
- `ACTIVE` or `CREATE_FAILED` - When the job finished or failed.

Type: Timestamp

Message

If an error occurred, an informational message about the error.

Type: String

Status

The status of the what-if forecast. States include:

- ACTIVE
- CREATE_PENDING, CREATE_IN_PROGRESS, CREATE_FAILED
- CREATE_STOPPING, CREATE_STOPPED
- DELETE_PENDING, DELETE_IN_PROGRESS, DELETE_FAILED

> **Note**
>
> The Status of the what-if forecast must be ACTIVE before you can access the forecast.

Type: String

Length Constraints: Maximum length of 256.

Pattern: ^[a-zA-Z0-9_]+$

TimeSeriesReplacementsDataSource

An array of S3Config, Schema, and Format elements that describe the replacement time series.

Type: TimeSeriesReplacementsDataSource object

TimeSeriesTransformations

An array of Action and TimeSeriesConditions elements that describe what transformations were applied to which time series.

Type: Array of TimeSeriesTransformation objects

Array Members: Minimum number of 0 items. Maximum number of 30 items.

WhatIfAnalysisArn

The Amazon Resource Name (ARN) of the what-if analysis that contains this forecast.

Type: String

Length Constraints: Maximum length of 256.

Pattern: `arn:([a-z\d-]+):forecast:.*:.*:.+`

WhatIfForecastArn

The Amazon Resource Name (ARN) of the what-if forecast.

Type: String

Length Constraints: Maximum length of 300.

Pattern: `arn:([a-z\d-]+):forecast:.*:.*:.+`

WhatIfForecastName

The name of the what-if forecast.

Type: String

Length Constraints: Minimum length of 1. Maximum length of 63.

Pattern: `^[a-zA-Z][a-zA-Z0-9_]*`

Errors

InvalidInputException

We can't process the request because it includes an invalid value or a value that exceeds the valid range.

HTTP Status Code: 400

ResourceNotFoundException

We can't find a resource with that Amazon Resource Name (ARN). Check the ARN and try again.

HTTP Status Code: 400

See Also

For more information about using this API in one of the language-specific AWS SDKs, see the following:

- AWS Command Line Interface
- AWS SDK for .NET

- AWS SDK for C++
- AWS SDK for Go v2
- AWS SDK for Java V2
- AWS SDK for JavaScript V3
- AWS SDK for PHP V3
- AWS SDK for Python
- AWS SDK for Ruby V3

DescribeWhatIfForecastExport

Service: Amazon Forecast Service

Describes the what-if forecast export created using the CreateWhatIfForecastExport operation.

In addition to listing the properties provided in the `CreateWhatIfForecastExport` request, this operation lists the following properties:

- `CreationTime`
- `LastModificationTime`
- `Message` - If an error occurred, information about the error.
- `Status`

Request Syntax

```
{
   "WhatIfForecastExportArn": "string"
}
```

Request Parameters

The request accepts the following data in JSON format.

WhatIfForecastExportArn

The Amazon Resource Name (ARN) of the what-if forecast export that you are interested in.

Type: String

Length Constraints: Maximum length of 300.

Pattern: `arn:([a-z\d-]+):forecast:.*:.*:.+`

Required: Yes

Response Syntax

```
{
   "CreationTime": number,
   "Destination": {
```

```
      "S3Config": {
         "KMSKeyArn": "string",
         "Path": "string",
         "RoleArn": "string"
      }
   },
   "EstimatedTimeRemainingInMinutes": number,
   "Format": "string",
   "LastModificationTime": number,
   "Message": "string",
   "Status": "string",
   "WhatIfForecastArns": [ "string" ],
   "WhatIfForecastExportArn": "string",
   "WhatIfForecastExportName": "string"
}
```

Response Elements

If the action is successful, the service sends back an HTTP 200 response.

The following data is returned in JSON format by the service.

CreationTime

When the what-if forecast export was created.

Type: Timestamp

Destination

The destination for an export job. Provide an S3 path, an AWS Identity and Access Management (IAM) role that allows Amazon Forecast to access the location, and an AWS Key Management Service (KMS) key (optional).

Type: DataDestination object

EstimatedTimeRemainingInMinutes

The approximate time remaining to complete the what-if forecast export, in minutes.

Type: Long

Format

The format of the exported data, CSV or PARQUET.

Type: String

Length Constraints: Maximum length of 7.

Pattern: ^CSV|PARQUET$

LastModificationTime

The last time the resource was modified. The timestamp depends on the status of the job:

- CREATE_PENDING - The CreationTime.
- CREATE_IN_PROGRESS - The current timestamp.
- CREATE_STOPPING - The current timestamp.
- CREATE_STOPPED - When the job stopped.
- ACTIVE or CREATE_FAILED - When the job finished or failed.

Type: Timestamp

Message

If an error occurred, an informational message about the error.

Type: String

Status

The status of the what-if forecast. States include:

- ACTIVE
- CREATE_PENDING, CREATE_IN_PROGRESS, CREATE_FAILED
- CREATE_STOPPING, CREATE_STOPPED
- DELETE_PENDING, DELETE_IN_PROGRESS, DELETE_FAILED

> **Note**
>
> The Status of the what-if forecast export must be ACTIVE before you can access the forecast export.

Type: String

Length Constraints: Maximum length of 256.

WhatIfForecastArns

An array of Amazon Resource Names (ARNs) that represent all of the what-if forecasts exported in this resource.

Type: Array of strings

Length Constraints: Maximum length of 300.

Pattern: `arn:([a-z\d-]+):forecast:.*:.*:.+`

WhatIfForecastExportArn

The Amazon Resource Name (ARN) of the what-if forecast export.

Type: String

Length Constraints: Maximum length of 300.

Pattern: `arn:([a-z\d-]+):forecast:.*:.*:.+`

WhatIfForecastExportName

The name of the what-if forecast export.

Type: String

Length Constraints: Minimum length of 1. Maximum length of 63.

Pattern: `^[a-zA-Z][a-zA-Z0-9_]*`

Errors

InvalidInputException

We can't process the request because it includes an invalid value or a value that exceeds the valid range.

HTTP Status Code: 400

ResourceNotFoundException

We can't find a resource with that Amazon Resource Name (ARN). Check the ARN and try again.

HTTP Status Code: 400

See Also

For more information about using this API in one of the language-specific AWS SDKs, see the following:

- AWS Command Line Interface
- AWS SDK for .NET
- AWS SDK for C++
- AWS SDK for Go v2
- AWS SDK for Java V2
- AWS SDK for JavaScript V3
- AWS SDK for PHP V3
- AWS SDK for Python
- AWS SDK for Ruby V3

GetAccuracyMetrics

Service: Amazon Forecast Service

Provides metrics on the accuracy of the models that were trained by the CreatePredictor operation. Use metrics to see how well the model performed and to decide whether to use the predictor to generate a forecast. For more information, see Predictor Metrics.

This operation generates metrics for each backtest window that was evaluated. The number of backtest windows (`NumberOfBacktestWindows`) is specified using the EvaluationParameters object, which is optionally included in the `CreatePredictor` request. If `NumberOfBacktestWindows` isn't specified, the number defaults to one.

The parameters of the `filling` method determine which items contribute to the metrics. If you want all items to contribute, specify `zero`. If you want only those items that have complete data in the range being evaluated to contribute, specify `nan`. For more information, see FeaturizationMethod.

> **Note**
>
> Before you can get accuracy metrics, the `Status` of the predictor must be `ACTIVE`, signifying that training has completed. To get the status, use the DescribePredictor operation.

Request Syntax

```
{
   "PredictorArn": "string"
}
```

Request Parameters

The request accepts the following data in JSON format.

PredictorArn

The Amazon Resource Name (ARN) of the predictor to get metrics for.

Type: String

Length Constraints: Maximum length of 256.

Pattern: `arn:([a-z\d-]+):forecast:.*:.*:.+`

Required: Yes

Response Syntax

```
{
   "AutoMLOverrideStrategy": "string",
   "IsAutoPredictor": boolean,
   "OptimizationMetric": "string",
   "PredictorEvaluationResults": [
      {
         "AlgorithmArn": "string",
         "TestWindows": [
            {
               "EvaluationType": "string",
               "ItemCount": number,
               "Metrics": {
                  "AverageWeightedQuantileLoss": number,
                  "ErrorMetrics": [
                     {
                        "ForecastType": "string",
                        "MAPE": number,
                        "MASE": number,
                        "RMSE": number,
                        "WAPE": number
                     }
                  ],
                  "RMSE": number,
                  "WeightedQuantileLosses": [
                     {
                        "LossValue": number,
                        "Quantile": number
                     }
                  ]
               },
               "TestWindowEnd": number,
               "TestWindowStart": number
            }
         ]
      }
```

```
    ]
}
```

Response Elements

If the action is successful, the service sends back an HTTP 200 response.

The following data is returned in JSON format by the service.

AutoMLOverrideStrategy

> **Note**
>
> The `LatencyOptimized` AutoML override strategy is only available in private beta. Contact AWS Support or your account manager to learn more about access privileges.

The AutoML strategy used to train the predictor. Unless `LatencyOptimized` is specified, the AutoML strategy optimizes predictor accuracy.

This parameter is only valid for predictors trained using AutoML.

Type: String

Valid Values: `LatencyOptimized | AccuracyOptimized`

IsAutoPredictor

Whether the predictor was created with CreateAutoPredictor.

Type: Boolean

OptimizationMetric

The accuracy metric used to optimize the predictor.

Type: String

Valid Values: `WAPE | RMSE | AverageWeightedQuantileLoss | MASE | MAPE`

PredictorEvaluationResults

An array of results from evaluating the predictor.

Type: Array of EvaluationResult objects

Errors

InvalidInputException

We can't process the request because it includes an invalid value or a value that exceeds the valid range.

HTTP Status Code: 400

ResourceInUseException

The specified resource is in use.

HTTP Status Code: 400

ResourceNotFoundException

We can't find a resource with that Amazon Resource Name (ARN). Check the ARN and try again.

HTTP Status Code: 400

See Also

For more information about using this API in one of the language-specific AWS SDKs, see the following:

- AWS Command Line Interface
- AWS SDK for .NET
- AWS SDK for C++
- AWS SDK for Go v2
- AWS SDK for Java V2
- AWS SDK for JavaScript V3
- AWS SDK for PHP V3
- AWS SDK for Python
- AWS SDK for Ruby V3

ListDatasetGroups

Service: Amazon Forecast Service

Returns a list of dataset groups created using the CreateDatasetGroup operation. For each dataset group, this operation returns a summary of its properties, including its Amazon Resource Name (ARN). You can retrieve the complete set of properties by using the dataset group ARN with the DescribeDatasetGroup operation.

Request Syntax

```
{
   "MaxResults": number,
   "NextToken": "string"
}
```

Request Parameters

The request accepts the following data in JSON format.

MaxResults

The number of items to return in the response.

Type: Integer

Valid Range: Minimum value of 1. Maximum value of 100.

Required: No

NextToken

If the result of the previous request was truncated, the response includes a `NextToken`. To retrieve the next set of results, use the token in the next request. Tokens expire after 24 hours.

Type: String

Length Constraints: Minimum length of 1. Maximum length of 3000.

Pattern: `.+`

Required: No

Response Syntax

```
{
   "DatasetGroups": [
      {
         "CreationTime": number,
         "DatasetGroupArn": "string",
         "DatasetGroupName": "string",
         "LastModificationTime": number
      }
   ],
   "NextToken": "string"
}
```

Response Elements

If the action is successful, the service sends back an HTTP 200 response.

The following data is returned in JSON format by the service.

DatasetGroups

An array of objects that summarize each dataset group's properties.

Type: Array of DatasetGroupSummary objects

NextToken

If the response is truncated, Amazon Forecast returns this token. To retrieve the next set of results, use the token in the next request.

Type: String

Length Constraints: Minimum length of 1. Maximum length of 3000.

Pattern: `.+`

Errors

InvalidNextTokenException

The token is not valid. Tokens expire after 24 hours.

HTTP Status Code: 400

See Also

For more information about using this API in one of the language-specific AWS SDKs, see the following:

- AWS Command Line Interface
- AWS SDK for .NET
- AWS SDK for C++
- AWS SDK for Go v2
- AWS SDK for Java V2
- AWS SDK for JavaScript V3
- AWS SDK for PHP V3
- AWS SDK for Python
- AWS SDK for Ruby V3

ListDatasetImportJobs

Service: Amazon Forecast Service

Returns a list of dataset import jobs created using the CreateDatasetImportJob operation. For each import job, this operation returns a summary of its properties, including its Amazon Resource Name (ARN). You can retrieve the complete set of properties by using the ARN with the DescribeDatasetImportJob operation. You can filter the list by providing an array of Filter objects.

Request Syntax

```
{
   "Filters": [
      {
         "Condition": "string",
         "Key": "string",
         "Value": "string"
      }
   ],
   "MaxResults": number,
   "NextToken": "string"
}
```

Request Parameters

The request accepts the following data in JSON format.

Filters

An array of filters. For each filter, you provide a condition and a match statement. The condition is either `IS` or `IS_NOT`, which specifies whether to include or exclude the datasets that match the statement from the list, respectively. The match statement consists of a key and a value.

Filter properties

- `Condition` - The condition to apply. Valid values are `IS` and `IS_NOT`. To include the datasets that match the statement, specify `IS`. To exclude matching datasets, specify `IS_NOT`.
- `Key` - The name of the parameter to filter on. Valid values are `DatasetArn` and `Status`.
- `Value` - The value to match.

For example, to list all dataset import jobs whose status is ACTIVE, you specify the following filter:

`"Filters": [ { "Condition": "IS", "Key": "Status", "Value": "ACTIVE" } ]`

Type: Array of Filter objects

Required: No

MaxResults

The number of items to return in the response.

Type: Integer

Valid Range: Minimum value of 1. Maximum value of 100.

Required: No

NextToken

If the result of the previous request was truncated, the response includes a `NextToken`. To retrieve the next set of results, use the token in the next request. Tokens expire after 24 hours.

Type: String

Length Constraints: Minimum length of 1. Maximum length of 3000.

Pattern: `.+`

Required: No

Response Syntax

```
{
   "DatasetImportJobs": [
      {
         "CreationTime": number,
         "DatasetImportJobArn": "string",
         "DatasetImportJobName": "string",
         "DataSource": {
            "S3Config": {
               "KMSKeyArn": "string",
               "Path": "string",
               "RoleArn": "string"
            }
         },
```

```
            "ImportMode": "string",
            "LastModificationTime": number,
            "Message": "string",
            "Status": "string"
        }
    ],
    "NextToken": "string"
}
```

Response Elements

If the action is successful, the service sends back an HTTP 200 response.

The following data is returned in JSON format by the service.

DatasetImportJobs

An array of objects that summarize each dataset import job's properties.

Type: Array of DatasetImportJobSummary objects

NextToken

If the response is truncated, Amazon Forecast returns this token. To retrieve the next set of results, use the token in the next request.

Type: String

Length Constraints: Minimum length of 1. Maximum length of 3000.

Pattern: `.+`

Errors

InvalidInputException

We can't process the request because it includes an invalid value or a value that exceeds the valid range.

HTTP Status Code: 400

InvalidNextTokenException

The token is not valid. Tokens expire after 24 hours.

HTTP Status Code: 400

See Also

For more information about using this API in one of the language-specific AWS SDKs, see the following:

- AWS Command Line Interface
- AWS SDK for .NET
- AWS SDK for C++
- AWS SDK for Go v2
- AWS SDK for Java V2
- AWS SDK for JavaScript V3
- AWS SDK for PHP V3
- AWS SDK for Python
- AWS SDK for Ruby V3

ListDatasets

Service: Amazon Forecast Service

Returns a list of datasets created using the CreateDataset operation. For each dataset, a summary of its properties, including its Amazon Resource Name (ARN), is returned. To retrieve the complete set of properties, use the ARN with the DescribeDataset operation.

Request Syntax

```
{
   "MaxResults": number,
   "NextToken": "string"
}
```

Request Parameters

The request accepts the following data in JSON format.

MaxResults

The number of items to return in the response.

Type: Integer

Valid Range: Minimum value of 1. Maximum value of 100.

Required: No

NextToken

If the result of the previous request was truncated, the response includes a `NextToken`. To retrieve the next set of results, use the token in the next request. Tokens expire after 24 hours.

Type: String

Length Constraints: Minimum length of 1. Maximum length of 3000.

Pattern: `.+`

Required: No

Response Syntax

```
{
```

```
   "Datasets": [
      {
         "CreationTime": number,
         "DatasetArn": "string",
         "DatasetName": "string",
         "DatasetType": "string",
         "Domain": "string",
         "LastModificationTime": number
      }
   ],
   "NextToken": "string"
}
```

Response Elements

If the action is successful, the service sends back an HTTP 200 response.

The following data is returned in JSON format by the service.

Datasets

An array of objects that summarize each dataset's properties.

Type: Array of DatasetSummary objects

NextToken

If the response is truncated, Amazon Forecast returns this token. To retrieve the next set of results, use the token in the next request.

Type: String

Length Constraints: Minimum length of 1. Maximum length of 3000.

Pattern: `.+`

Errors

InvalidNextTokenException

The token is not valid. Tokens expire after 24 hours.

HTTP Status Code: 400

See Also

For more information about using this API in one of the language-specific AWS SDKs, see the following:

- AWS Command Line Interface
- AWS SDK for .NET
- AWS SDK for C++
- AWS SDK for Go v2
- AWS SDK for Java V2
- AWS SDK for JavaScript V3
- AWS SDK for PHP V3
- AWS SDK for Python
- AWS SDK for Ruby V3

ListExplainabilities

Service: Amazon Forecast Service

Returns a list of Explainability resources created using the CreateExplainability operation. This operation returns a summary for each Explainability. You can filter the list using an array of Filter objects.

To retrieve the complete set of properties for a particular Explainability resource, use the ARN with the DescribeExplainability operation.

Request Syntax

```
{
   "Filters": [
      {
         "Condition": "string",
         "Key": "string",
         "Value": "string"
      }
   ],
   "MaxResults": number,
   "NextToken": "string"
}
```

Request Parameters

The request accepts the following data in JSON format.

Filters

An array of filters. For each filter, provide a condition and a match statement. The condition is either `IS` or `IS_NOT`, which specifies whether to include or exclude the resources that match the statement from the list. The match statement consists of a key and a value.

Filter properties

- `Condition` - The condition to apply. Valid values are `IS` and `IS_NOT`.
- `Key` - The name of the parameter to filter on. Valid values are `ResourceArn` and `Status`.
- `Value` - The value to match.

Type: Array of Filter objects

Required: No

MaxResults

The number of items returned in the response.

Type: Integer

Valid Range: Minimum value of 1. Maximum value of 100.

Required: No

NextToken

If the result of the previous request was truncated, the response includes a NextToken. To retrieve the next set of results, use the token in the next request. Tokens expire after 24 hours.

Type: String

Length Constraints: Minimum length of 1. Maximum length of 3000.

Pattern: `.+`

Required: No

Response Syntax

```
{
   "Explainabilities": [
      {
         "CreationTime": number,
         "ExplainabilityArn": "string",
         "ExplainabilityConfig": {
            "TimePointGranularity": "string",
            "TimeSeriesGranularity": "string"
         },
         "ExplainabilityName": "string",
         "LastModificationTime": number,
         "Message": "string",
         "ResourceArn": "string",
         "Status": "string"
      }
   ],
   "NextToken": "string"
```

```
}
```

Response Elements

If the action is successful, the service sends back an HTTP 200 response.

The following data is returned in JSON format by the service.

Explainabilities

An array of objects that summarize the properties of each Explainability resource.

Type: Array of ExplainabilitySummary objects

NextToken

Returns this token if the response is truncated. To retrieve the next set of results, use the token in the next request.

Type: String

Length Constraints: Minimum length of 1. Maximum length of 3000.

Pattern: `.+`

Errors

InvalidInputException

We can't process the request because it includes an invalid value or a value that exceeds the valid range.

HTTP Status Code: 400

InvalidNextTokenException

The token is not valid. Tokens expire after 24 hours.

HTTP Status Code: 400

See Also

For more information about using this API in one of the language-specific AWS SDKs, see the following:

- AWS Command Line Interface
- AWS SDK for .NET
- AWS SDK for C++
- AWS SDK for Go v2
- AWS SDK for Java V2
- AWS SDK for JavaScript V3
- AWS SDK for PHP V3
- AWS SDK for Python
- AWS SDK for Ruby V3

ListExplainabilityExports

Service: Amazon Forecast Service

Returns a list of Explainability exports created using the CreateExplainabilityExport operation. This operation returns a summary for each Explainability export. You can filter the list using an array of Filter objects.

To retrieve the complete set of properties for a particular Explainability export, use the ARN with the DescribeExplainability operation.

Request Syntax

```
{
   "Filters": [
      {
         "Condition": "string",
         "Key": "string",
         "Value": "string"
      }
   ],
   "MaxResults": number,
   "NextToken": "string"
}
```

Request Parameters

The request accepts the following data in JSON format.

Filters

An array of filters. For each filter, provide a condition and a match statement. The condition is either `IS` or `IS_NOT`, which specifies whether to include or exclude resources that match the statement from the list. The match statement consists of a key and a value.

Filter properties

- `Condition` - The condition to apply. Valid values are `IS` and `IS_NOT`.
- `Key` - The name of the parameter to filter on. Valid values are `ResourceArn` and `Status`.
- `Value` - The value to match.

Type: Array of Filter objects

Required: No

MaxResults

The number of items to return in the response.

Type: Integer

Valid Range: Minimum value of 1. Maximum value of 100.

Required: No

NextToken

If the result of the previous request was truncated, the response includes a NextToken. To retrieve the next set of results, use the token in the next request. Tokens expire after 24 hours.

Type: String

Length Constraints: Minimum length of 1. Maximum length of 3000.

Pattern: `.+`

Required: No

Response Syntax

```
{
   "ExplainabilityExports": [
      {
         "CreationTime": number,
         "Destination": {
            "S3Config": {
               "KMSKeyArn": "string",
               "Path": "string",
               "RoleArn": "string"
            }
         },
         "ExplainabilityExportArn": "string",
         "ExplainabilityExportName": "string",
         "LastModificationTime": number,
         "Message": "string",
         "Status": "string"
```

```
      }
   ],
   "NextToken": "string"
}
```

Response Elements

If the action is successful, the service sends back an HTTP 200 response.

The following data is returned in JSON format by the service.

ExplainabilityExports

An array of objects that summarize the properties of each Explainability export.

Type: Array of ExplainabilityExportSummary objects

NextToken

Returns this token if the response is truncated. To retrieve the next set of results, use the token in the next request.

Type: String

Length Constraints: Minimum length of 1. Maximum length of 3000.

Pattern: `.+`

Errors

InvalidInputException

We can't process the request because it includes an invalid value or a value that exceeds the valid range.

HTTP Status Code: 400

InvalidNextTokenException

The token is not valid. Tokens expire after 24 hours.

HTTP Status Code: 400

See Also

For more information about using this API in one of the language-specific AWS SDKs, see the following:

- AWS Command Line Interface
- AWS SDK for .NET
- AWS SDK for C++
- AWS SDK for Go v2
- AWS SDK for Java V2
- AWS SDK for JavaScript V3
- AWS SDK for PHP V3
- AWS SDK for Python
- AWS SDK for Ruby V3

ListForecastExportJobs

Service: Amazon Forecast Service

Returns a list of forecast export jobs created using the CreateForecastExportJob operation. For each forecast export job, this operation returns a summary of its properties, including its Amazon Resource Name (ARN). To retrieve the complete set of properties, use the ARN with the DescribeForecastExportJob operation. You can filter the list using an array of Filter objects.

Request Syntax

```
{
   "Filters": [
      {
         "Condition": "string",
         "Key": "string",
         "Value": "string"
      }
   ],
   "MaxResults": number,
   "NextToken": "string"
}
```

Request Parameters

The request accepts the following data in JSON format.

Filters

An array of filters. For each filter, you provide a condition and a match statement. The condition is either `IS` or `IS_NOT`, which specifies whether to include or exclude the forecast export jobs that match the statement from the list, respectively. The match statement consists of a key and a value.

Filter properties

- `Condition` - The condition to apply. Valid values are `IS` and `IS_NOT`. To include the forecast export jobs that match the statement, specify `IS`. To exclude matching forecast export jobs, specify `IS_NOT`.
- `Key` - The name of the parameter to filter on. Valid values are `ForecastArn` and `Status`.
- `Value` - The value to match.

For example, to list all jobs that export a forecast named *electricityforecast*, specify the following filter:

`"Filters": [ { "Condition": "IS", "Key": "ForecastArn", "Value": "arn:aws:forecast:us-west-2:<acct-id>:forecast/electricityforecast" } ]`

Type: Array of Filter objects

Required: No

MaxResults

The number of items to return in the response.

Type: Integer

Valid Range: Minimum value of 1. Maximum value of 100.

Required: No

NextToken

If the result of the previous request was truncated, the response includes a `NextToken`. To retrieve the next set of results, use the token in the next request. Tokens expire after 24 hours.

Type: String

Length Constraints: Minimum length of 1. Maximum length of 3000.

Pattern: `.+`

Required: No

Response Syntax

```
{
   "ForecastExportJobs": [
      {
         "CreationTime": number,
         "Destination": {
            "S3Config": {
               "KMSKeyArn": "string",
               "Path": "string",
               "RoleArn": "string"
```

```
            }
         },
         "ForecastExportJobArn": "string",
         "ForecastExportJobName": "string",
         "LastModificationTime": number,
         "Message": "string",
         "Status": "string"
      }
   ],
   "NextToken": "string"
}
```

Response Elements

If the action is successful, the service sends back an HTTP 200 response.

The following data is returned in JSON format by the service.

ForecastExportJobs

An array of objects that summarize each export job's properties.

Type: Array of ForecastExportJobSummary objects

NextToken

If the response is truncated, Amazon Forecast returns this token. To retrieve the next set of results, use the token in the next request.

Type: String

Length Constraints: Minimum length of 1. Maximum length of 3000.

Pattern: `.+`

Errors

InvalidInputException

We can't process the request because it includes an invalid value or a value that exceeds the valid range.

HTTP Status Code: 400

InvalidNextTokenException

The token is not valid. Tokens expire after 24 hours.

HTTP Status Code: 400

See Also

For more information about using this API in one of the language-specific AWS SDKs, see the following:

- AWS Command Line Interface
- AWS SDK for .NET
- AWS SDK for C++
- AWS SDK for Go v2
- AWS SDK for Java V2
- AWS SDK for JavaScript V3
- AWS SDK for PHP V3
- AWS SDK for Python
- AWS SDK for Ruby V3

ListForecasts

Service: Amazon Forecast Service

Returns a list of forecasts created using the CreateForecast operation. For each forecast, this operation returns a summary of its properties, including its Amazon Resource Name (ARN). To retrieve the complete set of properties, specify the ARN with the DescribeForecast operation. You can filter the list using an array of Filter objects.

Request Syntax

```
{
   "Filters": [
      {
         "Condition": "string",
         "Key": "string",
         "Value": "string"
      }
   ],
   "MaxResults": number,
   "NextToken": "string"
}
```

Request Parameters

The request accepts the following data in JSON format.

Filters

An array of filters. For each filter, you provide a condition and a match statement. The condition is either `IS` or `IS_NOT`, which specifies whether to include or exclude the forecasts that match the statement from the list, respectively. The match statement consists of a key and a value.

Filter properties

- `Condition` - The condition to apply. Valid values are `IS` and `IS_NOT`. To include the forecasts that match the statement, specify `IS`. To exclude matching forecasts, specify `IS_NOT`.
- `Key` - The name of the parameter to filter on. Valid values are `DatasetGroupArn`, `PredictorArn`, and `Status`.
- `Value` - The value to match.

For example, to list all forecasts whose status is not ACTIVE, you would specify:

`"Filters": [ { "Condition": "IS_NOT", "Key": "Status", "Value": "ACTIVE" } ]`

Type: Array of Filter objects

Required: No

MaxResults

The number of items to return in the response.

Type: Integer

Valid Range: Minimum value of 1. Maximum value of 100.

Required: No

NextToken

If the result of the previous request was truncated, the response includes a `NextToken`. To retrieve the next set of results, use the token in the next request. Tokens expire after 24 hours.

Type: String

Length Constraints: Minimum length of 1. Maximum length of 3000.

Pattern: `.+`

Required: No

Response Syntax

```
{
   "Forecasts": [
      {
         "CreatedUsingAutoPredictor": boolean,
         "CreationTime": number,
         "DatasetGroupArn": "string",
         "ForecastArn": "string",
         "ForecastName": "string",
         "LastModificationTime": number,
         "Message": "string",
```

```
            "PredictorArn": "string",
            "Status": "string"
        }
    ],
    "NextToken": "string"
}
```

Response Elements

If the action is successful, the service sends back an HTTP 200 response.

The following data is returned in JSON format by the service.

Forecasts

An array of objects that summarize each forecast's properties.

Type: Array of ForecastSummary objects

NextToken

If the response is truncated, Amazon Forecast returns this token. To retrieve the next set of results, use the token in the next request.

Type: String

Length Constraints: Minimum length of 1. Maximum length of 3000.

Pattern: `.+`

Errors

InvalidInputException

We can't process the request because it includes an invalid value or a value that exceeds the valid range.

HTTP Status Code: 400

InvalidNextTokenException

The token is not valid. Tokens expire after 24 hours.

HTTP Status Code: 400

See Also

For more information about using this API in one of the language-specific AWS SDKs, see the following:

- AWS Command Line Interface
- AWS SDK for .NET
- AWS SDK for C++
- AWS SDK for Go v2
- AWS SDK for Java V2
- AWS SDK for JavaScript V3
- AWS SDK for PHP V3
- AWS SDK for Python
- AWS SDK for Ruby V3

ListMonitorEvaluations

Service: Amazon Forecast Service

Returns a list of the monitoring evaluation results and predictor events collected by the monitor resource during different windows of time.

For information about monitoring see Predictor Monitoring. For more information about retrieving monitoring results see Viewing Monitoring Results.

Request Syntax

```
{
   "Filters": [
      {
         "Condition": "string",
         "Key": "string",
         "Value": "string"
      }
   ],
   "MaxResults": number,
   "MonitorArn": "string",
   "NextToken": "string"
}
```

Request Parameters

The request accepts the following data in JSON format.

Filters

An array of filters. For each filter, provide a condition and a match statement. The condition is either `IS` or `IS_NOT`, which specifies whether to include or exclude the resources that match the statement from the list. The match statement consists of a key and a value.

Filter properties

- `Condition` - The condition to apply. Valid values are `IS` and `IS_NOT`.
- `Key` - The name of the parameter to filter on. The only valid value is `EvaluationState`.
- `Value` - The value to match. Valid values are only `SUCCESS` or `FAILURE`.

For example, to list only successful monitor evaluations, you would specify:

```
"Filters": [ { "Condition": "IS", "Key": "EvaluationState", "Value":
"SUCCESS" } ]
```

Type: Array of Filter objects

Required: No

MaxResults

The maximum number of monitoring results to return.

Type: Integer

Valid Range: Minimum value of 1. Maximum value of 100.

Required: No

MonitorArn

The Amazon Resource Name (ARN) of the monitor resource to get results from.

Type: String

Length Constraints: Maximum length of 256.

Pattern: `arn:([a-z\d-]+):forecast:.*:.*:.+`

Required: Yes

NextToken

If the result of the previous request was truncated, the response includes a `NextToken`. To retrieve the next set of results, use the token in the next request. Tokens expire after 24 hours.

Type: String

Length Constraints: Minimum length of 1. Maximum length of 3000.

Pattern: `.+`

Required: No

Response Syntax

```
{
```

```
   "NextToken": "string",
   "PredictorMonitorEvaluations": [
      {
         "EvaluationState": "string",
         "EvaluationTime": number,
         "Message": "string",
         "MetricResults": [
            {
               "MetricName": "string",
               "MetricValue": number
            }
         ],
         "MonitorArn": "string",
         "MonitorDataSource": {
            "DatasetImportJobArn": "string",
            "ForecastArn": "string",
            "PredictorArn": "string"
         },
         "NumItemsEvaluated": number,
         "PredictorEvent": {
            "Datetime": number,
            "Detail": "string"
         },
         "ResourceArn": "string",
         "WindowEndDatetime": number,
         "WindowStartDatetime": number
      }
   ]
}
```

Response Elements

If the action is successful, the service sends back an HTTP 200 response.

The following data is returned in JSON format by the service.

NextToken

If the response is truncated, Amazon Forecast returns this token. To retrieve the next set of results, use the token in the next request. Tokens expire after 24 hours.

Type: String

Length Constraints: Minimum length of 1. Maximum length of 3000.

Pattern: .+

PredictorMonitorEvaluations

The monitoring results and predictor events collected by the monitor resource during different windows of time.

For information about monitoring see Viewing Monitoring Results. For more information about retrieving monitoring results see Viewing Monitoring Results.

Type: Array of PredictorMonitorEvaluation objects

Errors

InvalidInputException

We can't process the request because it includes an invalid value or a value that exceeds the valid range.

HTTP Status Code: 400

InvalidNextTokenException

The token is not valid. Tokens expire after 24 hours.

HTTP Status Code: 400

ResourceNotFoundException

We can't find a resource with that Amazon Resource Name (ARN). Check the ARN and try again.

HTTP Status Code: 400

See Also

For more information about using this API in one of the language-specific AWS SDKs, see the following:

- AWS Command Line Interface
- AWS SDK for .NET
- AWS SDK for C++
- AWS SDK for Go v2

- AWS SDK for Java V2
- AWS SDK for JavaScript V3
- AWS SDK for PHP V3
- AWS SDK for Python
- AWS SDK for Ruby V3

ListMonitors

Service: Amazon Forecast Service

Returns a list of monitors created with the CreateMonitor operation and CreateAutoPredictor operation. For each monitor resource, this operation returns of a summary of its properties, including its Amazon Resource Name (ARN). You can retrieve a complete set of properties of a monitor resource by specify the monitor's ARN in the DescribeMonitor operation.

Request Syntax

```
{
   "Filters": [
      {
         "Condition": "string",
         "Key": "string",
         "Value": "string"
      }
   ],
   "MaxResults": number,
   "NextToken": "string"
}
```

Request Parameters

The request accepts the following data in JSON format.

Filters

An array of filters. For each filter, provide a condition and a match statement. The condition is either `IS` or `IS_NOT`, which specifies whether to include or exclude the resources that match the statement from the list. The match statement consists of a key and a value.

Filter properties

- `Condition` - The condition to apply. Valid values are `IS` and `IS_NOT`.
- `Key` - The name of the parameter to filter on. The only valid value is `Status`.
- `Value` - The value to match.

For example, to list all monitors who's status is ACTIVE, you would specify:

```
"Filters": [ { "Condition": "IS", "Key": "Status", "Value": "ACTIVE" } ]
```

Type: Array of Filter objects

Required: No

MaxResults

The maximum number of monitors to include in the response.

Type: Integer

Valid Range: Minimum value of 1. Maximum value of 100.

Required: No

NextToken

If the result of the previous request was truncated, the response includes a `NextToken`. To retrieve the next set of results, use the token in the next request. Tokens expire after 24 hours.

Type: String

Length Constraints: Minimum length of 1. Maximum length of 3000.

Pattern: `.+`

Required: No

Response Syntax

```
{
   "Monitors": [
      {
         "CreationTime": number,
         "LastModificationTime": number,
         "MonitorArn": "string",
         "MonitorName": "string",
         "ResourceArn": "string",
         "Status": "string"
      }
   ],
   "NextToken": "string"
}
```

Response Elements

If the action is successful, the service sends back an HTTP 200 response.

The following data is returned in JSON format by the service.

Monitors

An array of objects that summarize each monitor's properties.

Type: Array of MonitorSummary objects

NextToken

If the response is truncated, Amazon Forecast returns this token. To retrieve the next set of results, use the token in the next request.

Type: String

Length Constraints: Minimum length of 1. Maximum length of 3000.

Pattern: `.+`

Errors

InvalidInputException

We can't process the request because it includes an invalid value or a value that exceeds the valid range.

HTTP Status Code: 400

InvalidNextTokenException

The token is not valid. Tokens expire after 24 hours.

HTTP Status Code: 400

See Also

For more information about using this API in one of the language-specific AWS SDKs, see the following:

- AWS Command Line Interface
- AWS SDK for .NET
- AWS SDK for C++
- AWS SDK for Go v2
- AWS SDK for Java V2
- AWS SDK for JavaScript V3
- AWS SDK for PHP V3
- AWS SDK for Python
- AWS SDK for Ruby V3

ListPredictorBacktestExportJobs

Service: Amazon Forecast Service

Returns a list of predictor backtest export jobs created using the CreatePredictorBacktestExportJob operation. This operation returns a summary for each backtest export job. You can filter the list using an array of Filter objects.

To retrieve the complete set of properties for a particular backtest export job, use the ARN with the DescribePredictorBacktestExportJob operation.

Request Syntax

```
{
   "Filters": [
      {
         "Condition": "string",
         "Key": "string",
         "Value": "string"
      }
   ],
   "MaxResults": number,
   "NextToken": "string"
}
```

Request Parameters

The request accepts the following data in JSON format.

Filters

An array of filters. For each filter, provide a condition and a match statement. The condition is either `IS` or `IS_NOT`, which specifies whether to include or exclude the predictor backtest export jobs that match the statement from the list. The match statement consists of a key and a value.

Filter properties

- `Condition` - The condition to apply. Valid values are `IS` and `IS_NOT`. To include the predictor backtest export jobs that match the statement, specify `IS`. To exclude matching predictor backtest export jobs, specify `IS_NOT`.
- `Key` - The name of the parameter to filter on. Valid values are `PredictorArn` and `Status`.

- `Value` - The value to match.

Type: Array of Filter objects

Required: No

MaxResults

The number of items to return in the response.

Type: Integer

Valid Range: Minimum value of 1. Maximum value of 100.

Required: No

NextToken

If the result of the previous request was truncated, the response includes a NextToken. To retrieve the next set of results, use the token in the next request. Tokens expire after 24 hours.

Type: String

Length Constraints: Minimum length of 1. Maximum length of 3000.

Pattern: `.+`

Required: No

Response Syntax

```
{
   "NextToken": "string",
   "PredictorBacktestExportJobs": [
      {
         "CreationTime": number,
         "Destination": {
            "S3Config": {
               "KMSKeyArn": "string",
               "Path": "string",
               "RoleArn": "string"
            }
         },
         "LastModificationTime": number,
```

```
            "Message": "string",
            "PredictorBacktestExportJobArn": "string",
            "PredictorBacktestExportJobName": "string",
            "Status": "string"
        }
    ]
}
```

Response Elements

If the action is successful, the service sends back an HTTP 200 response.

The following data is returned in JSON format by the service.

NextToken

Returns this token if the response is truncated. To retrieve the next set of results, use the token in the next request.

Type: String

Length Constraints: Minimum length of 1. Maximum length of 3000.

Pattern: `.+`

PredictorBacktestExportJobs

An array of objects that summarize the properties of each predictor backtest export job.

Type: Array of PredictorBacktestExportJobSummary objects

Errors

InvalidInputException

We can't process the request because it includes an invalid value or a value that exceeds the valid range.

HTTP Status Code: 400

InvalidNextTokenException

The token is not valid. Tokens expire after 24 hours.

HTTP Status Code: 400

See Also

For more information about using this API in one of the language-specific AWS SDKs, see the following:

- AWS Command Line Interface
- AWS SDK for .NET
- AWS SDK for C++
- AWS SDK for Go v2
- AWS SDK for Java V2
- AWS SDK for JavaScript V3
- AWS SDK for PHP V3
- AWS SDK for Python
- AWS SDK for Ruby V3

ListPredictors

Service: Amazon Forecast Service

Returns a list of predictors created using the CreateAutoPredictor or CreatePredictor operations. For each predictor, this operation returns a summary of its properties, including its Amazon Resource Name (ARN).

You can retrieve the complete set of properties by using the ARN with the DescribeAutoPredictor and DescribePredictor operations. You can filter the list using an array of Filter objects.

Request Syntax

```
{
   "Filters": [
      {
         "Condition": "string",
         "Key": "string",
         "Value": "string"
      }
   ],
   "MaxResults": number,
   "NextToken": "string"
}
```

Request Parameters

The request accepts the following data in JSON format.

Filters

An array of filters. For each filter, you provide a condition and a match statement. The condition is either `IS` or `IS_NOT`, which specifies whether to include or exclude the predictors that match the statement from the list, respectively. The match statement consists of a key and a value.

Filter properties

- `Condition` - The condition to apply. Valid values are `IS` and `IS_NOT`. To include the predictors that match the statement, specify `IS`. To exclude matching predictors, specify `IS_NOT`.
- `Key` - The name of the parameter to filter on. Valid values are `DatasetGroupArn` and `Status`.

- `Value` - The value to match.

For example, to list all predictors whose status is ACTIVE, you would specify:

`"Filters": [ { "Condition": "IS", "Key": "Status", "Value": "ACTIVE" } ]`

Type: Array of Filter objects

Required: No

MaxResults

The number of items to return in the response.

Type: Integer

Valid Range: Minimum value of 1. Maximum value of 100.

Required: No

NextToken

If the result of the previous request was truncated, the response includes a `NextToken`. To retrieve the next set of results, use the token in the next request. Tokens expire after 24 hours.

Type: String

Length Constraints: Minimum length of 1. Maximum length of 3000.

Pattern: `.+`

Required: No

Response Syntax

```
{
   "NextToken": "string",
   "Predictors": [
      {
         "CreationTime": number,
         "DatasetGroupArn": "string",
         "IsAutoPredictor": boolean,
         "LastModificationTime": number,
```

```
         "Message": "string",
         "PredictorArn": "string",
         "PredictorName": "string",
         "ReferencePredictorSummary": {
            "Arn": "string",
            "State": "string"
         },
         "Status": "string"
      }
   ]
}
```

Response Elements

If the action is successful, the service sends back an HTTP 200 response.

The following data is returned in JSON format by the service.

NextToken

If the response is truncated, Amazon Forecast returns this token. To retrieve the next set of results, use the token in the next request.

Type: String

Length Constraints: Minimum length of 1. Maximum length of 3000.

Pattern: `.+`

Predictors

An array of objects that summarize each predictor's properties.

Type: Array of PredictorSummary objects

Errors

InvalidInputException

We can't process the request because it includes an invalid value or a value that exceeds the valid range.

HTTP Status Code: 400

InvalidNextTokenException

The token is not valid. Tokens expire after 24 hours.

HTTP Status Code: 400

See Also

For more information about using this API in one of the language-specific AWS SDKs, see the following:

- AWS Command Line Interface
- AWS SDK for .NET
- AWS SDK for C++
- AWS SDK for Go v2
- AWS SDK for Java V2
- AWS SDK for JavaScript V3
- AWS SDK for PHP V3
- AWS SDK for Python
- AWS SDK for Ruby V3

ListTagsForResource

Service: Amazon Forecast Service

Lists the tags for an Amazon Forecast resource.

Request Syntax

```
{
   "ResourceArn": "string"
}
```

Request Parameters

The request accepts the following data in JSON format.

ResourceArn

The Amazon Resource Name (ARN) that identifies the resource for which to list the tags.

Type: String

Length Constraints: Maximum length of 256.

Pattern: `arn:([a-z\d-]+):forecast:.*:.*:.+`

Required: Yes

Response Syntax

```
{
   "Tags": [
      {
         "Key": "string",
         "Value": "string"
      }
   ]
}
```

Response Elements

If the action is successful, the service sends back an HTTP 200 response.

The following data is returned in JSON format by the service.

Tags

The tags for the resource.

Type: Array of Tag objects

Array Members: Minimum number of 0 items. Maximum number of 200 items.

Errors

InvalidInputException

We can't process the request because it includes an invalid value or a value that exceeds the valid range.

HTTP Status Code: 400

ResourceNotFoundException

We can't find a resource with that Amazon Resource Name (ARN). Check the ARN and try again.

HTTP Status Code: 400

See Also

For more information about using this API in one of the language-specific AWS SDKs, see the following:

- AWS Command Line Interface
- AWS SDK for .NET
- AWS SDK for C++
- AWS SDK for Go v2
- AWS SDK for Java V2
- AWS SDK for JavaScript V3
- AWS SDK for PHP V3
- AWS SDK for Python
- AWS SDK for Ruby V3

ListWhatIfAnalyses

Service: Amazon Forecast Service

Returns a list of what-if analyses created using the CreateWhatIfAnalysis operation. For each what-if analysis, this operation returns a summary of its properties, including its Amazon Resource Name (ARN). You can retrieve the complete set of properties by using the what-if analysis ARN with the DescribeWhatIfAnalysis operation.

Request Syntax

```
{
   "Filters": [
      {
         "Condition": "string",
         "Key": "string",
         "Value": "string"
      }
   ],
   "MaxResults": number,
   "NextToken": "string"
}
```

Request Parameters

The request accepts the following data in JSON format.

Filters

An array of filters. For each filter, you provide a condition and a match statement. The condition is either `IS` or `IS_NOT`, which specifies whether to include or exclude the what-if analysis jobs that match the statement from the list, respectively. The match statement consists of a key and a value.

Filter properties

- `Condition` - The condition to apply. Valid values are `IS` and `IS_NOT`. To include the what-if analysis jobs that match the statement, specify `IS`. To exclude matching what-if analysis jobs, specify `IS_NOT`.
- `Key` - The name of the parameter to filter on. Valid values are `WhatIfAnalysisArn` and `Status`.
- `Value` - The value to match.

For example, to list all jobs that export a forecast named *electricityWhatIf*, specify the following filter:

```
"Filters": [ { "Condition": "IS", "Key": "WhatIfAnalysisArn", "Value":
"arn:aws:forecast:us-west-2:<acct-id>:forecast/electricityWhatIf" } ]
```

Type: Array of Filter objects

Required: No

MaxResults

The number of items to return in the response.

Type: Integer

Valid Range: Minimum value of 1. Maximum value of 100.

Required: No

NextToken

If the result of the previous request was truncated, the response includes a `NextToken`. To retrieve the next set of results, use the token in the next request. Tokens expire after 24 hours.

Type: String

Length Constraints: Minimum length of 1. Maximum length of 3000.

Pattern: `.+`

Required: No

Response Syntax

```
{
   "NextToken": "string",
   "WhatIfAnalyses": [
      {
         "CreationTime": number,
         "ForecastArn": "string",
         "LastModificationTime": number,
         "Message": "string",
```

```
            "Status": "string",
            "WhatIfAnalysisArn": "string",
            "WhatIfAnalysisName": "string"
        }
    ]
}
```

Response Elements

If the action is successful, the service sends back an HTTP 200 response.

The following data is returned in JSON format by the service.

NextToken

If the response is truncated, Forecast returns this token. To retrieve the next set of results, use the token in the next request.

Type: String

Length Constraints: Minimum length of 1. Maximum length of 3000.

Pattern: `.+`

WhatIfAnalyses

An array of `WhatIfAnalysisSummary` objects that describe the matched analyses.

Type: Array of WhatIfAnalysisSummary objects

Errors

InvalidInputException

We can't process the request because it includes an invalid value or a value that exceeds the valid range.

HTTP Status Code: 400

InvalidNextTokenException

The token is not valid. Tokens expire after 24 hours.

HTTP Status Code: 400

See Also

For more information about using this API in one of the language-specific AWS SDKs, see the following:

- AWS Command Line Interface
- AWS SDK for .NET
- AWS SDK for C++
- AWS SDK for Go v2
- AWS SDK for Java V2
- AWS SDK for JavaScript V3
- AWS SDK for PHP V3
- AWS SDK for Python
- AWS SDK for Ruby V3

ListWhatIfForecastExports

Service: Amazon Forecast Service

Returns a list of what-if forecast exports created using the CreateWhatIfForecastExport operation. For each what-if forecast export, this operation returns a summary of its properties, including its Amazon Resource Name (ARN). You can retrieve the complete set of properties by using the what-if forecast export ARN with the DescribeWhatIfForecastExport operation.

Request Syntax

```
{
   "Filters": [
      {
         "Condition": "string",
         "Key": "string",
         "Value": "string"
      }
   ],
   "MaxResults": number,
   "NextToken": "string"
}
```

Request Parameters

The request accepts the following data in JSON format.

Filters

An array of filters. For each filter, you provide a condition and a match statement. The condition is either `IS` or `IS_NOT`, which specifies whether to include or exclude the what-if forecast export jobs that match the statement from the list, respectively. The match statement consists of a key and a value.

Filter properties

- `Condition` - The condition to apply. Valid values are `IS` and `IS_NOT`. To include the forecast export jobs that match the statement, specify `IS`. To exclude matching forecast export jobs, specify `IS_NOT`.
- `Key` - The name of the parameter to filter on. Valid values are `WhatIfForecastExportArn` and `Status`.
- `Value` - The value to match.

For example, to list all jobs that export a forecast named *electricityWIFExport*, specify the following filter:

```
"Filters": [ { "Condition": "IS", "Key": "WhatIfForecastExportArn",
"Value": "arn:aws:forecast:us-west-2:<acct-id>:forecast/
electricityWIFExport" } ]
```

Type: Array of Filter objects

Required: No

MaxResults

The number of items to return in the response.

Type: Integer

Valid Range: Minimum value of 1. Maximum value of 100.

Required: No

NextToken

If the result of the previous request was truncated, the response includes a `NextToken`. To retrieve the next set of results, use the token in the next request. Tokens expire after 24 hours.

Type: String

Length Constraints: Minimum length of 1. Maximum length of 3000.

Pattern: `.+`

Required: No

Response Syntax

```
{
   "NextToken": "string",
   "WhatIfForecastExports": [
      {
         "CreationTime": number,
         "Destination": {
            "S3Config": {
```

```
                "KMSKeyArn": "string",
                "Path": "string",
                "RoleArn": "string"
            }
        },
        "LastModificationTime": number,
        "Message": "string",
        "Status": "string",
        "WhatIfForecastArns": [ "string" ],
        "WhatIfForecastExportArn": "string",
        "WhatIfForecastExportName": "string"
    }
  ]
}
```

Response Elements

If the action is successful, the service sends back an HTTP 200 response.

The following data is returned in JSON format by the service.

NextToken

If the response is truncated, Forecast returns this token. To retrieve the next set of results, use the token in the next request.

Type: String

Length Constraints: Minimum length of 1. Maximum length of 3000.

Pattern: `.+`

WhatIfForecastExports

An array of `WhatIfForecastExports` objects that describe the matched forecast exports.

Type: Array of WhatIfForecastExportSummary objects

Errors

InvalidInputException

We can't process the request because it includes an invalid value or a value that exceeds the valid range.

HTTP Status Code: 400

InvalidNextTokenException

The token is not valid. Tokens expire after 24 hours.

HTTP Status Code: 400

See Also

For more information about using this API in one of the language-specific AWS SDKs, see the following:

- AWS Command Line Interface
- AWS SDK for .NET
- AWS SDK for C++
- AWS SDK for Go v2
- AWS SDK for Java V2
- AWS SDK for JavaScript V3
- AWS SDK for PHP V3
- AWS SDK for Python
- AWS SDK for Ruby V3

ListWhatIfForecasts

Service: Amazon Forecast Service

Returns a list of what-if forecasts created using the CreateWhatIfForecast operation. For each what-if forecast, this operation returns a summary of its properties, including its Amazon Resource Name (ARN). You can retrieve the complete set of properties by using the what-if forecast ARN with the DescribeWhatIfForecast operation.

Request Syntax

```
{
   "Filters": [
      {
         "Condition": "string",
         "Key": "string",
         "Value": "string"
      }
   ],
   "MaxResults": number,
   "NextToken": "string"
}
```

Request Parameters

The request accepts the following data in JSON format.

Filters

An array of filters. For each filter, you provide a condition and a match statement. The condition is either `IS` or `IS_NOT`, which specifies whether to include or exclude the what-if forecast export jobs that match the statement from the list, respectively. The match statement consists of a key and a value.

Filter properties

- `Condition` - The condition to apply. Valid values are `IS` and `IS_NOT`. To include the forecast export jobs that match the statement, specify `IS`. To exclude matching forecast export jobs, specify `IS_NOT`.
- `Key` - The name of the parameter to filter on. Valid values are `WhatIfForecastArn` and `Status`.
- `Value` - The value to match.

For example, to list all jobs that export a forecast named *electricityWhatIfForecast*, specify the following filter:

```
"Filters": [ { "Condition": "IS", "Key": "WhatIfForecastArn",
"Value": "arn:aws:forecast:us-west-2:<acct-id>:forecast/
electricityWhatIfForecast" } ]
```

Type: Array of Filter objects

Required: No

MaxResults

The number of items to return in the response.

Type: Integer

Valid Range: Minimum value of 1. Maximum value of 100.

Required: No

NextToken

If the result of the previous request was truncated, the response includes a `NextToken`. To retrieve the next set of results, use the token in the next request. Tokens expire after 24 hours.

Type: String

Length Constraints: Minimum length of 1. Maximum length of 3000.

Pattern: `.+`

Required: No

Response Syntax

```
{
   "NextToken": "string",
   "WhatIfForecasts": [
      {
         "CreationTime": number,
         "LastModificationTime": number,
         "Message": "string",
```

```
            "Status": "string",
            "WhatIfAnalysisArn": "string",
            "WhatIfForecastArn": "string",
            "WhatIfForecastName": "string"
        }
    ]
}
```

Response Elements

If the action is successful, the service sends back an HTTP 200 response.

The following data is returned in JSON format by the service.

NextToken

If the result of the previous request was truncated, the response includes a `NextToken`. To retrieve the next set of results, use the token in the next request. Tokens expire after 24 hours.

Type: String

Length Constraints: Minimum length of 1. Maximum length of 3000.

Pattern: `.+`

WhatIfForecasts

An array of `WhatIfForecasts` objects that describe the matched forecasts.

Type: Array of WhatIfForecastSummary objects

Errors

InvalidInputException

We can't process the request because it includes an invalid value or a value that exceeds the valid range.

HTTP Status Code: 400

InvalidNextTokenException

The token is not valid. Tokens expire after 24 hours.

HTTP Status Code: 400

See Also

For more information about using this API in one of the language-specific AWS SDKs, see the following:

- AWS Command Line Interface
- AWS SDK for .NET
- AWS SDK for C++
- AWS SDK for Go v2
- AWS SDK for Java V2
- AWS SDK for JavaScript V3
- AWS SDK for PHP V3
- AWS SDK for Python
- AWS SDK for Ruby V3

ResumeResource

Service: Amazon Forecast Service

Resumes a stopped monitor resource.

Request Syntax

```
{
   "ResourceArn": "string"
}
```

Request Parameters

The request accepts the following data in JSON format.

ResourceArn

The Amazon Resource Name (ARN) of the monitor resource to resume.

Type: String

Length Constraints: Maximum length of 256.

Pattern: `arn:([a-z\d-]+):forecast:.*:.*:.+`

Required: Yes

Response Elements

If the action is successful, the service sends back an HTTP 200 response with an empty HTTP body.

Errors

InvalidInputException

We can't process the request because it includes an invalid value or a value that exceeds the valid range.

HTTP Status Code: 400

LimitExceededException

The limit on the number of resources per account has been exceeded.

HTTP Status Code: 400

ResourceInUseException

The specified resource is in use.

HTTP Status Code: 400

ResourceNotFoundException

We can't find a resource with that Amazon Resource Name (ARN). Check the ARN and try again.

HTTP Status Code: 400

See Also

For more information about using this API in one of the language-specific AWS SDKs, see the following:

- AWS Command Line Interface
- AWS SDK for .NET
- AWS SDK for C++
- AWS SDK for Go v2
- AWS SDK for Java V2
- AWS SDK for JavaScript V3
- AWS SDK for PHP V3
- AWS SDK for Python
- AWS SDK for Ruby V3

StopResource

Service: Amazon Forecast Service

Stops a resource.

The resource undergoes the following states: `CREATE_STOPPING` and `CREATE_STOPPED`. You cannot resume a resource once it has been stopped.

This operation can be applied to the following resources (and their corresponding child resources):

- Dataset Import Job
- Predictor Job
- Forecast Job
- Forecast Export Job
- Predictor Backtest Export Job
- Explainability Job
- Explainability Export Job

Request Syntax

```
{
   "ResourceArn": "string"
}
```

Request Parameters

The request accepts the following data in JSON format.

ResourceArn

The Amazon Resource Name (ARN) that identifies the resource to stop. The supported ARNs are `DatasetImportJobArn`, `PredictorArn`, `PredictorBacktestExportJobArn`, `ForecastArn`, `ForecastExportJobArn`, `ExplainabilityArn`, and `ExplainabilityExportArn`.

Type: String

Length Constraints: Maximum length of 256.

Pattern: `arn:([a-z\d-]+):forecast:.*:.*:.+`

Required: Yes

Response Elements

If the action is successful, the service sends back an HTTP 200 response with an empty HTTP body.

Errors

InvalidInputException

We can't process the request because it includes an invalid value or a value that exceeds the valid range.

HTTP Status Code: 400

LimitExceededException

The limit on the number of resources per account has been exceeded.

HTTP Status Code: 400

ResourceNotFoundException

We can't find a resource with that Amazon Resource Name (ARN). Check the ARN and try again.

HTTP Status Code: 400

See Also

For more information about using this API in one of the language-specific AWS SDKs, see the following:

- AWS Command Line Interface
- AWS SDK for .NET
- AWS SDK for C++
- AWS SDK for Go v2
- AWS SDK for Java V2
- AWS SDK for JavaScript V3
- AWS SDK for PHP V3

- AWS SDK for Python
- AWS SDK for Ruby V3

TagResource

Service: Amazon Forecast Service

Associates the specified tags to a resource with the specified `resourceArn`. If existing tags on a resource are not specified in the request parameters, they are not changed. When a resource is deleted, the tags associated with that resource are also deleted.

Request Syntax

```
{
   "ResourceArn": "string",
   "Tags": [
      {
         "Key": "string",
         "Value": "string"
      }
   ]
}
```

Request Parameters

The request accepts the following data in JSON format.

ResourceArn

The Amazon Resource Name (ARN) that identifies the resource for which to list the tags.

Type: String

Length Constraints: Maximum length of 256.

Pattern: `arn:([a-z\d-]+):forecast:.*:.*:.+`

Required: Yes

Tags

The tags to add to the resource. A tag is an array of key-value pairs.

The following basic restrictions apply to tags:

- Maximum number of tags per resource - 50.
- For each resource, each tag key must be unique, and each tag key can have only one value.

- Maximum key length - 128 Unicode characters in UTF-8.
- Maximum value length - 256 Unicode characters in UTF-8.
- If your tagging schema is used across multiple services and resources, remember that other services may have restrictions on allowed characters. Generally allowed characters are: letters, numbers, and spaces representable in UTF-8, and the following characters: + - = . _ : / @.
- Tag keys and values are case sensitive.
- Do not use `aws:`, `AWS:`, or any upper or lowercase combination of such as a prefix for keys as it is reserved for AWS use. You cannot edit or delete tag keys with this prefix. Values can have this prefix. If a tag value has `aws` as its prefix but the key does not, then Forecast considers it to be a user tag and will count against the limit of 50 tags. Tags with only the key prefix of `aws` do not count against your tags per resource limit.

Type: Array of Tag objects

Array Members: Minimum number of 0 items. Maximum number of 200 items.

Required: Yes

Response Elements

If the action is successful, the service sends back an HTTP 200 response with an empty HTTP body.

Errors

InvalidInputException

We can't process the request because it includes an invalid value or a value that exceeds the valid range.

HTTP Status Code: 400

LimitExceededException

The limit on the number of resources per account has been exceeded.

HTTP Status Code: 400

ResourceNotFoundException

We can't find a resource with that Amazon Resource Name (ARN). Check the ARN and try again.

HTTP Status Code: 400

See Also

For more information about using this API in one of the language-specific AWS SDKs, see the following:

- AWS Command Line Interface
- AWS SDK for .NET
- AWS SDK for C++
- AWS SDK for Go v2
- AWS SDK for Java V2
- AWS SDK for JavaScript V3
- AWS SDK for PHP V3
- AWS SDK for Python
- AWS SDK for Ruby V3

UntagResource

Service: Amazon Forecast Service

Deletes the specified tags from a resource.

Request Syntax

```
{
   "ResourceArn": "string",
   "TagKeys": [ "string" ]
}
```

Request Parameters

The request accepts the following data in JSON format.

ResourceArn

The Amazon Resource Name (ARN) that identifies the resource for which to list the tags.

Type: String

Length Constraints: Maximum length of 256.

Pattern: `arn:([a-z\d-]+):forecast:.*:.*:.+`

Required: Yes

TagKeys

The keys of the tags to be removed.

Type: Array of strings

Array Members: Minimum number of 0 items. Maximum number of 200 items.

Length Constraints: Minimum length of 1. Maximum length of 128.

Pattern: `^([\p{L}\p{Z}\p{N}_.:/=+\-@]*)$`

Required: Yes

Response Elements

If the action is successful, the service sends back an HTTP 200 response with an empty HTTP body.

Errors

InvalidInputException

We can't process the request because it includes an invalid value or a value that exceeds the valid range.

HTTP Status Code: 400

ResourceNotFoundException

We can't find a resource with that Amazon Resource Name (ARN). Check the ARN and try again.

HTTP Status Code: 400

See Also

For more information about using this API in one of the language-specific AWS SDKs, see the following:

- AWS Command Line Interface
- AWS SDK for .NET
- AWS SDK for C++
- AWS SDK for Go v2
- AWS SDK for Java V2
- AWS SDK for JavaScript V3
- AWS SDK for PHP V3
- AWS SDK for Python
- AWS SDK for Ruby V3

UpdateDatasetGroup

Service: Amazon Forecast Service

Replaces the datasets in a dataset group with the specified datasets.

> **Note**
>
> The `Status` of the dataset group must be `ACTIVE` before you can use the dataset group to create a predictor. Use the DescribeDatasetGroup operation to get the status.

Request Syntax

```
{
   "DatasetArns": [ "string" ],
   "DatasetGroupArn": "string"
}
```

Request Parameters

The request accepts the following data in JSON format.

DatasetArns

An array of the Amazon Resource Names (ARNs) of the datasets to add to the dataset group.

Type: Array of strings

Length Constraints: Maximum length of 256.

Pattern: `arn:([a-z\d-]+):forecast:.*:.*:.+`

Required: Yes

DatasetGroupArn

The ARN of the dataset group.

Type: String

Length Constraints: Maximum length of 256.

Pattern: `arn:([a-z\d-]+):forecast:.*:.*:.+`

Required: Yes

Response Elements

If the action is successful, the service sends back an HTTP 200 response with an empty HTTP body.

Errors

InvalidInputException

We can't process the request because it includes an invalid value or a value that exceeds the valid range.

HTTP Status Code: 400

ResourceInUseException

The specified resource is in use.

HTTP Status Code: 400

ResourceNotFoundException

We can't find a resource with that Amazon Resource Name (ARN). Check the ARN and try again.

HTTP Status Code: 400

See Also

For more information about using this API in one of the language-specific AWS SDKs, see the following:

- AWS Command Line Interface
- AWS SDK for .NET
- AWS SDK for C++
- AWS SDK for Go v2
- AWS SDK for Java V2
- AWS SDK for JavaScript V3
- AWS SDK for PHP V3
- AWS SDK for Python

- AWS SDK for Ruby V3

Amazon Forecast Query Service

The following actions are supported by Amazon Forecast Query Service:

- QueryForecast
- QueryWhatIfForecast

QueryForecast

Service: Amazon Forecast Query Service

Retrieves a forecast for a single item, filtered by the supplied criteria.

The criteria is a key-value pair. The key is either `item_id` (or the equivalent non-timestamp, non-target field) from the `TARGET_TIME_SERIES` dataset, or one of the forecast dimensions specified as part of the `FeaturizationConfig` object.

By default, `QueryForecast` returns the complete date range for the filtered forecast. You can request a specific date range.

To get the full forecast, use the CreateForecastExportJob operation.

> **Note**
>
> The forecasts generated by Amazon Forecast are in the same timezone as the dataset that was used to create the predictor.

Request Syntax

```
{
   "EndDate": "string",
   "Filters": {
      "string" : "string"
   },
   "ForecastArn": "string",
   "NextToken": "string",
   "StartDate": "string"
}
```

Request Parameters

The request accepts the following data in JSON format.

EndDate

The end date for the forecast. Specify the date using this format: yyyy-MM-dd'T'HH:mm:ss (ISO 8601 format). For example, 2015-01-01T20:00:00.

Type: String

Required: No

Filters

The filtering criteria to apply when retrieving the forecast. For example, to get the forecast for `client_21` in the electricity usage dataset, specify the following:

`{"item_id" : "client_21"}`

To get the full forecast, use the CreateForecastExportJob operation.

Type: String to string map

Map Entries: Maximum number of 50 items.

Key Length Constraints: Maximum length of 256.

Key Pattern: `^[a-zA-Z0-9\_\-]+$`

Value Length Constraints: Maximum length of 256.

Required: Yes

ForecastArn

The Amazon Resource Name (ARN) of the forecast to query.

Type: String

Length Constraints: Maximum length of 256.

Pattern: `arn:([a-z\d-]+):forecast:.*:.*:.+`

Required: Yes

NextToken

If the result of the previous request was truncated, the response includes a `NextToken`. To retrieve the next set of results, use the token in the next request. Tokens expire after 24 hours.

Type: String

Length Constraints: Minimum length of 1. Maximum length of 3000.

Required: No

StartDate

The start date for the forecast. Specify the date using this format: yyyy-MM-dd'T'HH:mm:ss (ISO 8601 format). For example, 2015-01-01T08:00:00.

Type: String

Required: No

Response Syntax

```
{
   "Forecast": {
      "Predictions": {
         "string" : [
            {
               "Timestamp": "string",
               "Value": number
            }
         ]
      }
   }
}
```

Response Elements

If the action is successful, the service sends back an HTTP 200 response.

The following data is returned in JSON format by the service.

Forecast

The forecast.

Type: Forecast object

Errors

InvalidInputException

The value is invalid or is too long.

HTTP Status Code: 400

InvalidNextTokenException

The token is not valid. Tokens expire after 24 hours.

HTTP Status Code: 400

LimitExceededException

The limit on the number of requests per second has been exceeded.

HTTP Status Code: 400

ResourceInUseException

The specified resource is in use.

HTTP Status Code: 400

ResourceNotFoundException

We can't find that resource. Check the information that you've provided and try again.

HTTP Status Code: 400

See Also

For more information about using this API in one of the language-specific AWS SDKs, see the following:

- AWS Command Line Interface
- AWS SDK for .NET
- AWS SDK for C++
- AWS SDK for Go v2
- AWS SDK for Java V2
- AWS SDK for JavaScript V3
- AWS SDK for PHP V3
- AWS SDK for Python
- AWS SDK for Ruby V3

QueryWhatIfForecast

Service: Amazon Forecast Query Service

Retrieves a what-if forecast.

Request Syntax

```
{
   "EndDate": "string",
   "Filters": {
      "string" : "string"
   },
   "NextToken": "string",
   "StartDate": "string",
   "WhatIfForecastArn": "string"
}
```

Request Parameters

The request accepts the following data in JSON format.

EndDate

The end date for the what-if forecast. Specify the date using this format: yyyy-MM-dd'T'HH:mm:ss (ISO 8601 format). For example, 2015-01-01T20:00:00.

Type: String

Required: No

Filters

The filtering criteria to apply when retrieving the forecast. For example, to get the forecast for `client_21` in the electricity usage dataset, specify the following:

`{"item_id" : "client_21"}`

To get the full what-if forecast, use the CreateForecastExportJob operation.

Type: String to string map

Map Entries: Maximum number of 50 items.

Key Length Constraints: Maximum length of 256.

Key Pattern: `^[a-zA-Z0-9\_\-]+$`

Value Length Constraints: Maximum length of 256.

Required: Yes

NextToken

If the result of the previous request was truncated, the response includes a `NextToken`. To retrieve the next set of results, use the token in the next request. Tokens expire after 24 hours.

Type: String

Length Constraints: Minimum length of 1. Maximum length of 3000.

Required: No

StartDate

The start date for the what-if forecast. Specify the date using this format: yyyy-MM-dd'T'HH:mm:ss (ISO 8601 format). For example, 2015-01-01T08:00:00.

Type: String

Required: No

WhatIfForecastArn

The Amazon Resource Name (ARN) of the what-if forecast to query.

Type: String

Length Constraints: Maximum length of 300.

Pattern: `arn:([a-z\d-]+):forecast:.*:.*:.+`

Required: Yes

Response Syntax

```
{
   "Forecast": {
      "Predictions": {
         "string" : [
```

```
            {
               "Timestamp": "string",
               "Value": number
            }
         ]
      }
   }
}
```

Response Elements

If the action is successful, the service sends back an HTTP 200 response.

The following data is returned in JSON format by the service.

Forecast

Provides information about a forecast. Returned as part of the QueryForecast response.

Type: Forecast object

Errors

InvalidInputException

The value is invalid or is too long.

HTTP Status Code: 400

InvalidNextTokenException

The token is not valid. Tokens expire after 24 hours.

HTTP Status Code: 400

LimitExceededException

The limit on the number of requests per second has been exceeded.

HTTP Status Code: 400

ResourceInUseException

The specified resource is in use.

HTTP Status Code: 400

ResourceNotFoundException

We can't find that resource. Check the information that you've provided and try again.

HTTP Status Code: 400

See Also

For more information about using this API in one of the language-specific AWS SDKs, see the following:

- AWS Command Line Interface
- AWS SDK for .NET
- AWS SDK for C++
- AWS SDK for Go v2
- AWS SDK for Java V2
- AWS SDK for JavaScript V3
- AWS SDK for PHP V3
- AWS SDK for Python
- AWS SDK for Ruby V3

Data Types

The following data types are supported by Amazon Forecast Service:

- Action
- AdditionalDataset
- AttributeConfig
- Baseline
- BaselineMetric
- CategoricalParameterRange
- ContinuousParameterRange
- DataConfig

- DataDestination
- DatasetGroupSummary
- DatasetImportJobSummary
- DatasetSummary
- DataSource
- EncryptionConfig
- ErrorMetric
- EvaluationParameters
- EvaluationResult
- ExplainabilityConfig
- ExplainabilityExportSummary
- ExplainabilityInfo
- ExplainabilitySummary
- Featurization
- FeaturizationConfig
- FeaturizationMethod
- Filter
- ForecastExportJobSummary
- ForecastSummary
- HyperParameterTuningJobConfig
- InputDataConfig
- IntegerParameterRange
- MetricResult
- Metrics
- MonitorConfig
- MonitorDataSource
- MonitorInfo
- MonitorSummary
- ParameterRanges
- PredictorBacktestExportJobSummary

- PredictorBaseline
- PredictorEvent
- PredictorExecution
- PredictorExecutionDetails
- PredictorMonitorEvaluation
- PredictorSummary
- ReferencePredictorSummary
- S3Config
- Schema
- SchemaAttribute
- Statistics
- SupplementaryFeature
- Tag
- TestWindowSummary
- TimeAlignmentBoundary
- TimeSeriesCondition
- TimeSeriesIdentifiers
- TimeSeriesReplacementsDataSource
- TimeSeriesSelector
- TimeSeriesTransformation
- WeightedQuantileLoss
- WhatIfAnalysisSummary
- WhatIfForecastExportSummary
- WhatIfForecastSummary
- WindowSummary

The following data types are supported by Amazon Forecast Query Service:

- DataPoint
- Forecast

Amazon Forecast Service

The following data types are supported by Amazon Forecast Service:

- Action
- AdditionalDataset
- AttributeConfig
- Baseline
- BaselineMetric
- CategoricalParameterRange
- ContinuousParameterRange
- DataConfig
- DataDestination
- DatasetGroupSummary
- DatasetImportJobSummary
- DatasetSummary
- DataSource
- EncryptionConfig
- ErrorMetric
- EvaluationParameters
- EvaluationResult
- ExplainabilityConfig
- ExplainabilityExportSummary
- ExplainabilityInfo
- ExplainabilitySummary
- Featurization
- FeaturizationConfig
- FeaturizationMethod
- Filter
- ForecastExportJobSummary
- ForecastSummary

- HyperParameterTuningJobConfig
- InputDataConfig
- IntegerParameterRange
- MetricResult
- Metrics
- MonitorConfig
- MonitorDataSource
- MonitorInfo
- MonitorSummary
- ParameterRanges
- PredictorBacktestExportJobSummary
- PredictorBaseline
- PredictorEvent
- PredictorExecution
- PredictorExecutionDetails
- PredictorMonitorEvaluation
- PredictorSummary
- ReferencePredictorSummary
- S3Config
- Schema
- SchemaAttribute
- Statistics
- SupplementaryFeature
- Tag
- TestWindowSummary
- TimeAlignmentBoundary
- TimeSeriesCondition
- TimeSeriesIdentifiers
- TimeSeriesReplacementsDataSource
- TimeSeriesSelector

- TimeSeriesTransformation
- WeightedQuantileLoss
- WhatIfAnalysisSummary
- WhatIfForecastExportSummary
- WhatIfForecastSummary
- WindowSummary

Action

Service: Amazon Forecast Service

Defines the modifications that you are making to an attribute for a what-if forecast. For example, you can use this operation to create a what-if forecast that investigates a 10% off sale on all shoes. To do this, you specify `"AttributeName": "shoes"`, `"Operation": "MULTIPLY"`, and `"Value": "0.90"`. Pair this operation with the TimeSeriesCondition operation within the CreateWhatIfForecast:TimeSeriesTransformations operation to define a subset of attribute items that are modified.

Contents

AttributeName

The related time series that you are modifying. This value is case insensitive.

Type: String

Length Constraints: Minimum length of 1. Maximum length of 63.

Pattern: `^[a-zA-Z][a-zA-Z0-9_]*`

Required: Yes

Operation

The operation that is applied to the provided attribute. Operations include:

- `ADD` - adds `Value` to all rows of `AttributeName`.
- `SUBTRACT` - subtracts `Value` from all rows of `AttributeName`.
- `MULTIPLY` - multiplies all rows of `AttributeName` by `Value`.
- `DIVIDE` - divides all rows of `AttributeName` by `Value`.

Type: String

Valid Values: `ADD | SUBTRACT | MULTIPLY | DIVIDE`

Required: Yes

Value

The value that is applied for the chosen `Operation`.

Type: Double

Required: Yes

See Also

For more information about using this API in one of the language-specific AWS SDKs, see the following:

- AWS SDK for C++
- AWS SDK for Java V2
- AWS SDK for Ruby V3

AdditionalDataset

Service: Amazon Forecast Service

Describes an additional dataset. This object is part of the DataConfig object. Forecast supports the Weather Index and Holidays additional datasets.

Weather Index

The Amazon Forecast Weather Index is a built-in dataset that incorporates historical and projected weather information into your model. The Weather Index supplements your datasets with over two years of historical weather data and up to 14 days of projected weather data. For more information, see Amazon Forecast Weather Index.

Holidays

Holidays is a built-in featurization that incorporates a feature-engineered dataset of national holiday information into your model. It provides native support for the holiday calendars of over 250 countries. Amazon Forecast incorporates both the Holiday API library and Jollyday API to generate holiday calendars. For more information, see Holidays Featurization.

Contents

Name

The name of the additional dataset. Valid names: `"holiday"` and `"weather"`.

Type: String

Length Constraints: Minimum length of 1. Maximum length of 63.

Pattern: `^[a-zA-Z][a-zA-Z0-9_]*`

Required: Yes

Configuration

Weather Index

To enable the Weather Index, do not specify a value for `Configuration`.

Holidays

Holidays

To enable Holidays, set `CountryCode` to one of the following two-letter country codes:

- Afghanistan - AF
- Åland Islands - AX
- Albania - AL
- Algeria - DZ
- American Samoa - AS
- Andorra - AD
- Angola - AO
- Anguilla - AI
- Antartica - AQ
- Antigua and Barbuda - AG
- Argentina - AR
- Armenia - AM
- Aruba - AW
- Australia - AU
- Austria - AT
- Azerbaijan - AZ
- Bahamas - BS
- Bahrain - BH
- Bangladesh - BD
- Barbados - BB
- Belarus - BY
- Belgium - BE
- Belize - BZ
- Benin - BJ
- Bermuda - BM
- Bhutan - BT
- Bolivia - BO
- Bosnia and Herzegovina - BA
- Botswana - BW

- Bouvet Island - BV
- Brazil - BR
- British Indian Ocean Territory - IO
- British Virgin Islands - VG
- Brunei Darussalam - BN
- Bulgaria - BG
- Burkina Faso - BF
- Burundi - BI
- Cambodia - KH
- Cameroon - CM
- Canada - CA
- Cape Verde - CV
- Caribbean Netherlands - BQ
- Cayman Islands - KY
- Central African Republic - CF
- Chad - TD
- Chile - CL
- China - CN
- Christmas Island - CX
- Cocos (Keeling) Islands - CC
- Colombia - CO
- Comoros - KM
- Cook Islands - CK
- Costa Rica - CR
- Croatia - HR
- Cuba - CU
- Curaçao - CW
- Cyprus - CY
- Czechia - CZ
- Democratic Republic of the Congo - CD

- Denmark - DK
- Djibouti - DJ
- Dominica - DM
- Dominican Republic - DO
- Ecuador - EC
- Egypt - EG
- El Salvador - SV
- Equatorial Guinea - GQ
- Eritrea - ER
- Estonia - EE
- Eswatini - SZ
- Ethiopia - ET
- Falkland Islands - FK
- Faroe Islands - FO
- Fiji - FJ
- Finland - FI
- France - FR
- French Guiana - GF
- French Polynesia - PF
- French Southern Territories - TF
- Gabon - GA
- Gambia - GM
- Georgia - GE
- Germany - DE
- Ghana - GH
- Gibraltar - GI
- Greece - GR
- Greenland - GL
- Grenada - GD
- Guadeloupe - GP

- Guam - GU
- Guatemala - GT
- Guernsey - GG
- Guinea - GN
- Guinea-Bissau - GW
- Guyana - GY
- Haiti - HT
- Heard Island and McDonald Islands - HM
- Honduras - HN
- Hong Kong - HK
- Hungary - HU
- Iceland - IS
- India - IN
- Indonesia - ID
- Iran - IR
- Iraq - IQ
- Ireland - IE
- Isle of Man - IM
- Israel - IL
- Italy - IT
- Ivory Coast - CI
- Jamaica - JM
- Japan - JP
- Jersey - JE
- Jordan - JO
- Kazakhstan - KZ
- Kenya - KE
- Kiribati - KI
- Kosovo - XK
- Kuwait - KW

- Kyrgyzstan - KG
- Laos - LA
- Latvia - LV
- Lebanon - LB
- Lesotho - LS
- Liberia - LR
- Libya - LY
- Liechtenstein - LI
- Lithuania - LT
- Luxembourg - LU
- Macao - MO
- Madagascar - MG
- Malawi - MW
- Malaysia - MY
- Maldives - MV
- Mali - ML
- Malta - MT
- Marshall Islands - MH
- Martinique - MQ
- Mauritania - MR
- Mauritius - MU
- Mayotte - YT
- Mexico - MX
- Micronesia - FM
- Moldova - MD
- Monaco - MC
- Mongolia - MN
- Montenegro - ME
- Montserrat - MS
- Morocco - MA

- Mozambique - MZ
- Myanmar - MM
- Namibia - NA
- Nauru - NR
- Nepal - NP
- Netherlands - NL
- New Caledonia - NC
- New Zealand - NZ
- Nicaragua - NI
- Niger - NE
- Nigeria - NG
- Niue - NU
- Norfolk Island - NF
- North Korea - KP
- North Macedonia - MK
- Northern Mariana Islands - MP
- Norway - NO
- Oman - OM
- Pakistan - PK
- Palau - PW
- Palestine - PS
- Panama - PA
- Papua New Guinea - PG
- Paraguay - PY
- Peru - PE
- Philippines - PH
- Pitcairn Islands - PN
- Poland - PL
- Portugal - PT
- Puerto Rico - PR

- Qatar - QA
- Republic of the Congo - CG
- Réunion - RE
- Romania - RO
- Russian Federation - RU
- Rwanda - RW
- Saint Barthélemy - BL
- "Saint Helena, Ascension and Tristan da Cunha " - SH
- Saint Kitts and Nevis - KN
- Saint Lucia - LC
- Saint Martin - MF
- Saint Pierre and Miquelon - PM
- Saint Vincent and the Grenadines - VC
- Samoa - WS
- San Marino - SM
- Sao Tome and Principe - ST
- Saudi Arabia - SA
- Senegal - SN
- Serbia - RS
- Seychelles - SC
- Sierra Leone - SL
- Singapore - SG
- Sint Maarten - SX
- Slovakia - SK
- Slovenia - SI
- Solomon Islands - SB
- Somalia - SO
- South Africa - ZA
- South Georgia and the South Sandwich Islands - GS
- South Korea - KR

- South Sudan - SS
- Spain - ES
- Sri Lanka - LK
- Sudan - SD
- Suriname - SR
- Svalbard and Jan Mayen - SJ
- Sweden - SE
- Switzerland - CH
- Syrian Arab Republic - SY
- Taiwan - TW
- Tajikistan - TJ
- Tanzania - TZ
- Thailand - TH
- Timor-Leste - TL
- Togo - TG
- Tokelau - TK
- Tonga - TO
- Trinidad and Tobago - TT
- Tunisia - TN
- Turkey - TR
- Turkmenistan - TM
- Turks and Caicos Islands - TC
- Tuvalu - TV
- Uganda - UG
- Ukraine - UA
- United Arab Emirates - AE
- United Kingdom - GB
- United Nations - UN
- United States - US
- United States Minor Outlying Islands - UM

- United States Virgin Islands - VI
- Uruguay - UY
- Uzbekistan - UZ
- Vanuatu - VU
- Vatican City - VA
- Venezuela - VE
- Vietnam - VN
- Wallis and Futuna - WF
- Western Sahara - EH
- Yemen - YE
- Zambia - ZM
- Zimbabwe - ZW

Type: String to array of strings map

Key Length Constraints: Minimum length of 1. Maximum length of 63.

Key Pattern: `^[a-zA-Z][a-zA-Z0-9_]*`

Array Members: Minimum number of 1 item. Maximum number of 20 items.

Length Constraints: Maximum length of 256.

Pattern: `^[a-zA-Z0-9\_\-]+$`

Required: No

See Also

For more information about using this API in one of the language-specific AWS SDKs, see the following:

- AWS SDK for C++
- AWS SDK for Java V2
- AWS SDK for Ruby V3

AttributeConfig

Service: Amazon Forecast Service

Provides information about the method used to transform attributes.

The following is an example using the RETAIL domain:

```
{

"AttributeName": "demand",

"Transformations": {"aggregation": "sum", "middlefill": "zero", "backfill":
"zero"}

}
```

Contents

AttributeName

The name of the attribute as specified in the schema. Amazon Forecast supports the target field of the target time series and the related time series datasets. For example, for the RETAIL domain, the target is `demand`.

Type: String

Length Constraints: Minimum length of 1. Maximum length of 63.

Pattern: `^[a-zA-Z][a-zA-Z0-9_]*`

Required: Yes

Transformations

The method parameters (key-value pairs), which are a map of override parameters. Specify these parameters to override the default values. Related Time Series attributes do not accept aggregation parameters.

The following list shows the parameters and their valid values for the "filling" featurization method for a **Target Time Series** dataset. Default values are bolded.

- `aggregation`: **sum**, `avg`, `first`, `min`, `max`
- `frontfill`: **none**

- `middlefill`: **zero**, `nan` (not a number), `value`, `median`, `mean`, `min`, `max`
- `backfill`: **zero**, `nan`, `value`, `median`, `mean`, `min`, `max`

The following list shows the parameters and their valid values for a **Related Time Series** featurization method (there are no defaults):

- `middlefill`: `zero`, `value`, `median`, `mean`, `min`, `max`
- `backfill`: `zero`, `value`, `median`, `mean`, `min`, `max`
- `futurefill`: `zero`, `value`, `median`, `mean`, `min`, `max`

To set a filling method to a specific value, set the fill parameter to `value` and define the value in a corresponding `_value` parameter. For example, to set backfilling to a value of 2, include the following: `"backfill": "value"` and `"backfill_value":"2"`.

Type: String to string map

Map Entries: Maximum number of 20 items.

Key Length Constraints: Minimum length of 1. Maximum length of 63.

Key Pattern: `^[a-zA-Z][a-zA-Z0-9_]*`

Value Length Constraints: Maximum length of 256.

Value Pattern: `^[a-zA-Z0-9\_\-]+$`

Required: Yes

See Also

For more information about using this API in one of the language-specific AWS SDKs, see the following:

- AWS SDK for C++
- AWS SDK for Java V2
- AWS SDK for Ruby V3

Baseline

Service: Amazon Forecast Service

Metrics you can use as a baseline for comparison purposes. Use these metrics when you interpret monitoring results for an auto predictor.

Contents

PredictorBaseline

The initial accuracy metrics for the predictor you are monitoring. Use these metrics as a baseline for comparison purposes as you use your predictor and the metrics change.

Type: PredictorBaseline object

Required: No

See Also

For more information about using this API in one of the language-specific AWS SDKs, see the following:

- AWS SDK for C++
- AWS SDK for Java V2
- AWS SDK for Ruby V3

BaselineMetric

Service: Amazon Forecast Service

An individual metric that you can use for comparison as you evaluate your monitoring results.

Contents

Name

The name of the metric.

Type: String

Length Constraints: Minimum length of 1. Maximum length of 63.

Pattern: `^[a-zA-Z][a-zA-Z0-9_]*`

Required: No

Value

The value for the metric.

Type: Double

Required: No

See Also

For more information about using this API in one of the language-specific AWS SDKs, see the following:

- AWS SDK for C++
- AWS SDK for Java V2
- AWS SDK for Ruby V3

CategoricalParameterRange

Service: Amazon Forecast Service

Specifies a categorical hyperparameter and it's range of tunable values. This object is part of the ParameterRanges object.

Contents

Name

The name of the categorical hyperparameter to tune.

Type: String

Length Constraints: Minimum length of 1. Maximum length of 63.

Pattern: `^[a-zA-Z][a-zA-Z0-9_]*`

Required: Yes

Values

A list of the tunable categories for the hyperparameter.

Type: Array of strings

Array Members: Minimum number of 1 item. Maximum number of 20 items.

Length Constraints: Maximum length of 256.

Pattern: `^[a-zA-Z0-9\_\-]+$`

Required: Yes

See Also

For more information about using this API in one of the language-specific AWS SDKs, see the following:

- AWS SDK for C++
- AWS SDK for Java V2
- AWS SDK for Ruby V3

ContinuousParameterRange

Service: Amazon Forecast Service

Specifies a continuous hyperparameter and it's range of tunable values. This object is part of the ParameterRanges object.

Contents

MaxValue

The maximum tunable value of the hyperparameter.

Type: Double

Required: Yes

MinValue

The minimum tunable value of the hyperparameter.

Type: Double

Required: Yes

Name

The name of the hyperparameter to tune.

Type: String

Length Constraints: Minimum length of 1. Maximum length of 63.

Pattern: `^[a-zA-Z][a-zA-Z0-9_]*`

Required: Yes

ScalingType

The scale that hyperparameter tuning uses to search the hyperparameter range. Valid values:

Auto

Amazon Forecast hyperparameter tuning chooses the best scale for the hyperparameter.

Linear

Hyperparameter tuning searches the values in the hyperparameter range by using a linear scale.

Logarithmic

Hyperparameter tuning searches the values in the hyperparameter range by using a logarithmic scale.

Logarithmic scaling works only for ranges that have values greater than 0.

ReverseLogarithmic

hyperparameter tuning searches the values in the hyperparameter range by using a reverse logarithmic scale.

Reverse logarithmic scaling works only for ranges that are entirely within the range 0 <= x < 1.0.

For information about choosing a hyperparameter scale, see Hyperparameter Scaling. One of the following values:

Type: String

Valid Values: `Auto | Linear | Logarithmic | ReverseLogarithmic`

Required: No

See Also

For more information about using this API in one of the language-specific AWS SDKs, see the following:

- AWS SDK for C++
- AWS SDK for Java V2
- AWS SDK for Ruby V3

DataConfig

Service: Amazon Forecast Service

The data configuration for your dataset group and any additional datasets.

Contents

DatasetGroupArn

The ARN of the dataset group used to train the predictor.

Type: String

Length Constraints: Maximum length of 256.

Pattern: `arn:([a-z\d-]+):forecast:.*:.*:.+`

Required: Yes

AdditionalDatasets

Additional built-in datasets like Holidays and the Weather Index.

Type: Array of AdditionalDataset objects

Array Members: Minimum number of 1 item. Maximum number of 2 items.

Required: No

AttributeConfigs

Aggregation and filling options for attributes in your dataset group.

Type: Array of AttributeConfig objects

Array Members: Minimum number of 1 item. Maximum number of 50 items.

Required: No

See Also

For more information about using this API in one of the language-specific AWS SDKs, see the following:

- AWS SDK for C++
- AWS SDK for Java V2
- AWS SDK for Ruby V3

DataDestination

Service: Amazon Forecast Service

The destination for an export job. Provide an S3 path, an AWS Identity and Access Management (IAM) role that allows Amazon Forecast to access the location, and an AWS Key Management Service (KMS) key (optional).

Contents

S3Config

The path to an Amazon Simple Storage Service (Amazon S3) bucket along with the credentials to access the bucket.

Type: S3Config object

Required: Yes

See Also

For more information about using this API in one of the language-specific AWS SDKs, see the following:

- AWS SDK for C++
- AWS SDK for Java V2
- AWS SDK for Ruby V3

DatasetGroupSummary

Service: Amazon Forecast Service

Provides a summary of the dataset group properties used in the ListDatasetGroups operation. To get the complete set of properties, call the DescribeDatasetGroup operation, and provide the `DatasetGroupArn`.

Contents

CreationTime

When the dataset group was created.

Type: Timestamp

Required: No

DatasetGroupArn

The Amazon Resource Name (ARN) of the dataset group.

Type: String

Length Constraints: Maximum length of 256.

Pattern: `arn:([a-z\d-]+):forecast:.*:.*:.+`

Required: No

DatasetGroupName

The name of the dataset group.

Type: String

Length Constraints: Minimum length of 1. Maximum length of 63.

Pattern: `^[a-zA-Z][a-zA-Z0-9_]*`

Required: No

LastModificationTime

When the dataset group was created or last updated from a call to the UpdateDatasetGroup operation. While the dataset group is being updated, `LastModificationTime` is the current time of the `ListDatasetGroups` call.

Type: Timestamp

Required: No

See Also

For more information about using this API in one of the language-specific AWS SDKs, see the following:

- AWS SDK for C++
- AWS SDK for Java V2
- AWS SDK for Ruby V3

DatasetImportJobSummary

Service: Amazon Forecast Service

Provides a summary of the dataset import job properties used in the ListDatasetImportJobs operation. To get the complete set of properties, call the DescribeDatasetImportJob operation, and provide the `DatasetImportJobArn`.

Contents

CreationTime

When the dataset import job was created.

Type: Timestamp

Required: No

DatasetImportJobArn

The Amazon Resource Name (ARN) of the dataset import job.

Type: String

Length Constraints: Maximum length of 256.

Pattern: `arn:([a-z\d-]+):forecast:.*:.*:.+`

Required: No

DatasetImportJobName

The name of the dataset import job.

Type: String

Length Constraints: Minimum length of 1. Maximum length of 63.

Pattern: `^[a-zA-Z][a-zA-Z0-9_]*`

Required: No

DataSource

The location of the training data to import and an AWS Identity and Access Management (IAM) role that Amazon Forecast can assume to access the data. The training data must be stored in an Amazon S3 bucket.

If encryption is used, `DataSource` includes an AWS Key Management Service (KMS) key.

Type: DataSource object

Required: No

ImportMode

The import mode of the dataset import job, FULL or INCREMENTAL.

Type: String

Valid Values: `FULL | INCREMENTAL`

Required: No

LastModificationTime

The last time the resource was modified. The timestamp depends on the status of the job:

- `CREATE_PENDING` - The `CreationTime`.
- `CREATE_IN_PROGRESS` - The current timestamp.
- `CREATE_STOPPING` - The current timestamp.
- `CREATE_STOPPED` - When the job stopped.
- `ACTIVE` or `CREATE_FAILED` - When the job finished or failed.

Type: Timestamp

Required: No

Message

If an error occurred, an informational message about the error.

Type: String

Required: No

Status

The status of the dataset import job. States include:

- `ACTIVE`
- `CREATE_PENDING`, `CREATE_IN_PROGRESS`, `CREATE_FAILED`
- `DELETE_PENDING`, `DELETE_IN_PROGRESS`, `DELETE_FAILED`

- `CREATE_STOPPING`, `CREATE_STOPPED`

Type: String

Length Constraints: Maximum length of 256.

Required: No

See Also

For more information about using this API in one of the language-specific AWS SDKs, see the following:

- AWS SDK for C++
- AWS SDK for Java V2
- AWS SDK for Ruby V3

DatasetSummary

Service: Amazon Forecast Service

Provides a summary of the dataset properties used in the ListDatasets operation. To get the complete set of properties, call the DescribeDataset operation, and provide the `DatasetArn`.

Contents

CreationTime

When the dataset was created.

Type: Timestamp

Required: No

DatasetArn

The Amazon Resource Name (ARN) of the dataset.

Type: String

Length Constraints: Maximum length of 256.

Pattern: `arn:([a-z\d-]+):forecast:.*:.*:.+`

Required: No

DatasetName

The name of the dataset.

Type: String

Length Constraints: Minimum length of 1. Maximum length of 63.

Pattern: `^[a-zA-Z][a-zA-Z0-9_]*`

Required: No

DatasetType

The dataset type.

Type: String

Valid Values: `TARGET_TIME_SERIES | RELATED_TIME_SERIES | ITEM_METADATA`

Required: No

Domain

The domain associated with the dataset.

Type: String

Valid Values: `RETAIL | CUSTOM | INVENTORY_PLANNING | EC2_CAPACITY | WORK_FORCE | WEB_TRAFFIC | METRICS`

Required: No

LastModificationTime

When you create a dataset, `LastModificationTime` is the same as `CreationTime`. While data is being imported to the dataset, `LastModificationTime` is the current time of the `ListDatasets` call. After a CreateDatasetImportJob operation has finished, `LastModificationTime` is when the import job completed or failed.

Type: Timestamp

Required: No

See Also

For more information about using this API in one of the language-specific AWS SDKs, see the following:

- AWS SDK for C++
- AWS SDK for Java V2
- AWS SDK for Ruby V3

DataSource

Service: Amazon Forecast Service

The source of your data, an AWS Identity and Access Management (IAM) role that allows Amazon Forecast to access the data and, optionally, an AWS Key Management Service (KMS) key.

Contents

S3Config

The path to the data stored in an Amazon Simple Storage Service (Amazon S3) bucket along with the credentials to access the data.

Type: S3Config object

Required: Yes

See Also

For more information about using this API in one of the language-specific AWS SDKs, see the following:

- AWS SDK for C++
- AWS SDK for Java V2
- AWS SDK for Ruby V3

EncryptionConfig

Service: Amazon Forecast Service

An AWS Key Management Service (KMS) key and an AWS Identity and Access Management (IAM) role that Amazon Forecast can assume to access the key. You can specify this optional object in the CreateDataset and CreatePredictor requests.

Contents

KMSKeyArn

The Amazon Resource Name (ARN) of the KMS key.

Type: String

Length Constraints: Maximum length of 256.

Pattern: `arn:aws:kms:.*:key/.*`

Required: Yes

RoleArn

The ARN of the IAM role that Amazon Forecast can assume to access the AWS KMS key.

Passing a role across AWS accounts is not allowed. If you pass a role that isn't in your account, you get an `InvalidInputException` error.

Type: String

Length Constraints: Maximum length of 256.

Pattern: `arn:([a-z\d-]+):forecast:.*:.*:.+`

Required: Yes

See Also

For more information about using this API in one of the language-specific AWS SDKs, see the following:

- AWS SDK for C++

- AWS SDK for Java V2
- AWS SDK for Ruby V3

ErrorMetric

Service: Amazon Forecast Service

Provides detailed error metrics to evaluate the performance of a predictor. This object is part of the Metrics object.

Contents

ForecastType

The Forecast type used to compute WAPE, MAPE, MASE, and RMSE.

Type: String

Length Constraints: Minimum length of 2. Maximum length of 4.

Pattern: `(^0?\.\d\d?$|^mean$)`

Required: No

MAPE

The Mean Absolute Percentage Error (MAPE)

Type: Double

Required: No

MASE

The Mean Absolute Scaled Error (MASE)

Type: Double

Required: No

RMSE

The root-mean-square error (RMSE).

Type: Double

Required: No

WAPE

The weighted absolute percentage error (WAPE).

Type: Double

Required: No

See Also

For more information about using this API in one of the language-specific AWS SDKs, see the following:

- AWS SDK for C++
- AWS SDK for Java V2
- AWS SDK for Ruby V3

EvaluationParameters

Service: Amazon Forecast Service

Parameters that define how to split a dataset into training data and testing data, and the number of iterations to perform. These parameters are specified in the predefined algorithms but you can override them in the CreatePredictor request.

Contents

BackTestWindowOffset

The point from the end of the dataset where you want to split the data for model training and testing (evaluation). Specify the value as the number of data points. The default is the value of the forecast horizon. `BackTestWindowOffset` can be used to mimic a past virtual forecast start date. This value must be greater than or equal to the forecast horizon and less than half of the TARGET_TIME_SERIES dataset length.

`ForecastHorizon` <= `BackTestWindowOffset` < 1/2 * TARGET_TIME_SERIES dataset length

Type: Integer

Required: No

NumberOfBacktestWindows

The number of times to split the input data. The default is 1. Valid values are 1 through 5.

Type: Integer

Required: No

See Also

For more information about using this API in one of the language-specific AWS SDKs, see the following:

- AWS SDK for C++
- AWS SDK for Java V2
- AWS SDK for Ruby V3

EvaluationResult

Service: Amazon Forecast Service

The results of evaluating an algorithm. Returned as part of the GetAccuracyMetrics response.

Contents

AlgorithmArn

The Amazon Resource Name (ARN) of the algorithm that was evaluated.

Type: String

Length Constraints: Maximum length of 256.

Pattern: `arn:([a-z\d-]+):forecast:.*:.*:.+`

Required: No

TestWindows

The array of test windows used for evaluating the algorithm. The `NumberOfBacktestWindows` from the EvaluationParameters object determines the number of windows in the array.

Type: Array of WindowSummary objects

Required: No

See Also

For more information about using this API in one of the language-specific AWS SDKs, see the following:

- AWS SDK for C++
- AWS SDK for Java V2
- AWS SDK for Ruby V3

ExplainabilityConfig

Service: Amazon Forecast Service

The ExplainabilityConfig data type defines the number of time series and time points included in CreateExplainability.

If you provide a predictor ARN for `ResourceArn`, you must set both `TimePointGranularity` and `TimeSeriesGranularity` to "ALL". When creating Predictor Explainability, Amazon Forecast considers all time series and time points.

If you provide a forecast ARN for `ResourceArn`, you can set `TimePointGranularity` and `TimeSeriesGranularity` to either "ALL" or "Specific".

Contents

TimePointGranularity

To create an Explainability for all time points in your forecast horizon, use `ALL`. To create an Explainability for specific time points in your forecast horizon, use `SPECIFIC`.

Specify time points with the `StartDateTime` and `EndDateTime` parameters within the CreateExplainability operation.

Type: String

Valid Values: `ALL | SPECIFIC`

Required: Yes

TimeSeriesGranularity

To create an Explainability for all time series in your datasets, use `ALL`. To create an Explainability for specific time series in your datasets, use `SPECIFIC`.

Specify time series by uploading a CSV or Parquet file to an Amazon S3 bucket and set the location within the DataDestination data type.

Type: String

Valid Values: `ALL | SPECIFIC`

Required: Yes

See Also

For more information about using this API in one of the language-specific AWS SDKs, see the following:

- AWS SDK for C++
- AWS SDK for Java V2
- AWS SDK for Ruby V3

ExplainabilityExportSummary

Service: Amazon Forecast Service

Provides a summary of the Explainability export properties used in the ListExplainabilityExports operation. To get a complete set of properties, call the DescribeExplainabilityExport operation, and provide the `ExplainabilityExportArn`.

Contents

CreationTime

When the Explainability was created.

Type: Timestamp

Required: No

Destination

The destination for an export job. Provide an S3 path, an AWS Identity and Access Management (IAM) role that allows Amazon Forecast to access the location, and an AWS Key Management Service (KMS) key (optional).

Type: DataDestination object

Required: No

ExplainabilityExportArn

The Amazon Resource Name (ARN) of the Explainability export.

Type: String

Length Constraints: Maximum length of 256.

Pattern: `arn:([a-z\d-]+):forecast:.*:.*:.+`

Required: No

ExplainabilityExportName

The name of the Explainability export

Type: String

Length Constraints: Minimum length of 1. Maximum length of 63.

Pattern: `^[a-zA-Z][a-zA-Z0-9_]*`

Required: No

LastModificationTime

The last time the resource was modified. The timestamp depends on the status of the job:

- `CREATE_PENDING` - The `CreationTime`.
- `CREATE_IN_PROGRESS` - The current timestamp.
- `CREATE_STOPPING` - The current timestamp.
- `CREATE_STOPPED` - When the job stopped.
- `ACTIVE` or `CREATE_FAILED` - When the job finished or failed.

Type: Timestamp

Required: No

Message

Information about any errors that may have occurred during the Explainability export.

Type: String

Required: No

Status

The status of the Explainability export. States include:

- `ACTIVE`
- `CREATE_PENDING`, `CREATE_IN_PROGRESS`, `CREATE_FAILED`
- `CREATE_STOPPING`, `CREATE_STOPPED`
- `DELETE_PENDING`, `DELETE_IN_PROGRESS`, `DELETE_FAILED`

Type: String

Length Constraints: Maximum length of 256.

Required: No

See Also

For more information about using this API in one of the language-specific AWS SDKs, see the following:

- AWS SDK for C++
- AWS SDK for Java V2
- AWS SDK for Ruby V3

ExplainabilityInfo

Service: Amazon Forecast Service

Provides information about the Explainability resource.

Contents

ExplainabilityArn

The Amazon Resource Name (ARN) of the Explainability.

Type: String

Length Constraints: Maximum length of 256.

Pattern: `arn:([a-z\d-]+):forecast:.*:.*:.+`

Required: No

Status

The status of the Explainability. States include:

- `ACTIVE`
- `CREATE_PENDING`, `CREATE_IN_PROGRESS`, `CREATE_FAILED`
- `CREATE_STOPPING`, `CREATE_STOPPED`
- `DELETE_PENDING`, `DELETE_IN_PROGRESS`, `DELETE_FAILED`

Type: String

Length Constraints: Maximum length of 256.

Required: No

See Also

For more information about using this API in one of the language-specific AWS SDKs, see the following:

- AWS SDK for C++
- AWS SDK for Java V2
- AWS SDK for Ruby V3

ExplainabilitySummary

Service: Amazon Forecast Service

Provides a summary of the Explainability properties used in the ListExplainabilities operation. To get a complete set of properties, call the DescribeExplainability operation, and provide the listed `ExplainabilityArn`.

Contents

CreationTime

When the Explainability was created.

Type: Timestamp

Required: No

ExplainabilityArn

The Amazon Resource Name (ARN) of the Explainability.

Type: String

Length Constraints: Maximum length of 256.

Pattern: `arn:([a-z\d-]+):forecast:.*:.*:.+`

Required: No

ExplainabilityConfig

The configuration settings that define the granularity of time series and time points for the Explainability.

Type: ExplainabilityConfig object

Required: No

ExplainabilityName

The name of the Explainability.

Type: String

Length Constraints: Minimum length of 1. Maximum length of 63.

Pattern: `^[a-zA-Z][a-zA-Z0-9_]*`

Required: No

LastModificationTime

The last time the resource was modified. The timestamp depends on the status of the job:

- `CREATE_PENDING` - The `CreationTime`.
- `CREATE_IN_PROGRESS` - The current timestamp.
- `CREATE_STOPPING` - The current timestamp.
- `CREATE_STOPPED` - When the job stopped.
- `ACTIVE` or `CREATE_FAILED` - When the job finished or failed.

Type: Timestamp

Required: No

Message

Information about any errors that may have occurred during the Explainability creation process.

Type: String

Required: No

ResourceArn

The Amazon Resource Name (ARN) of the Predictor or Forecast used to create the Explainability.

Type: String

Length Constraints: Maximum length of 256.

Pattern: `arn:([a-z\d-]+):forecast:.*:.*:.+`

Required: No

Status

The status of the Explainability. States include:

- `ACTIVE`
- `CREATE_PENDING`, `CREATE_IN_PROGRESS`, `CREATE_FAILED`
- `CREATE_STOPPING`, `CREATE_STOPPED`

- `DELETE_PENDING`, `DELETE_IN_PROGRESS`, `DELETE_FAILED`

Type: String

Length Constraints: Maximum length of 256.

Required: No

See Also

For more information about using this API in one of the language-specific AWS SDKs, see the following:

- AWS SDK for C++
- AWS SDK for Java V2
- AWS SDK for Ruby V3

Featurization

Service: Amazon Forecast Service

> **Note**
>
> This object belongs to the CreatePredictor operation. If you created your predictor with CreateAutoPredictor, see AttributeConfig.

Provides featurization (transformation) information for a dataset field. This object is part of the FeaturizationConfig object.

For example:

```
{

"AttributeName": "demand",

FeaturizationPipeline [ {

"FeaturizationMethodName": "filling",

"FeaturizationMethodParameters": {"aggregation": "avg", "backfill": "nan"}

} ]

}
```

Contents

AttributeName

The name of the schema attribute that specifies the data field to be featurized. Amazon Forecast supports the target field of the `TARGET_TIME_SERIES` and the `RELATED_TIME_SERIES` datasets. For example, for the `RETAIL` domain, the target is `demand`, and for the `CUSTOM` domain, the target is `target_value`. For more information, see Handling Missing Values.

Type: String

Length Constraints: Minimum length of 1. Maximum length of 63.

Pattern: `^[a-zA-Z][a-zA-Z0-9_]*`

Required: Yes

FeaturizationPipeline

An array of one `FeaturizationMethod` object that specifies the feature transformation method.

Type: Array of FeaturizationMethod objects

Array Members: Fixed number of 1 item.

Required: No

See Also

For more information about using this API in one of the language-specific AWS SDKs, see the following:

- AWS SDK for C++
- AWS SDK for Java V2
- AWS SDK for Ruby V3

FeaturizationConfig

Service: Amazon Forecast Service

> **Note**
>
> This object belongs to the CreatePredictor operation. If you created your predictor with CreateAutoPredictor, see AttributeConfig.

In a CreatePredictor operation, the specified algorithm trains a model using the specified dataset group. You can optionally tell the operation to modify data fields prior to training a model. These modifications are referred to as *featurization*.

You define featurization using the `FeaturizationConfig` object. You specify an array of transformations, one for each field that you want to featurize. You then include the `FeaturizationConfig` object in your `CreatePredictor` request. Amazon Forecast applies the featurization to the `TARGET_TIME_SERIES` and `RELATED_TIME_SERIES` datasets before model training.

You can create multiple featurization configurations. For example, you might call the `CreatePredictor` operation twice by specifying different featurization configurations.

Contents

ForecastFrequency

The frequency of predictions in a forecast.

Valid intervals are an integer followed by Y (Year), M (Month), W (Week), D (Day), H (Hour), and min (Minute). For example, "1D" indicates every day and "15min" indicates every 15 minutes. You cannot specify a value that would overlap with the next larger frequency. That means, for example, you cannot specify a frequency of 60 minutes, because that is equivalent to 1 hour. The valid values for each frequency are the following:

- Minute - 1-59
- Hour - 1-23
- Day - 1-6
- Week - 1-4
- Month - 1-11

- Year - 1

Thus, if you want every other week forecasts, specify "2W". Or, if you want quarterly forecasts, you specify "3M".

The frequency must be greater than or equal to the TARGET_TIME_SERIES dataset frequency.

When a RELATED_TIME_SERIES dataset is provided, the frequency must be equal to the TARGET_TIME_SERIES dataset frequency.

Type: String

Length Constraints: Minimum length of 1. Maximum length of 5.

Pattern: `^Y|M|W|D|H|30min|15min|10min|5min|1min$`

Required: Yes

Featurizations

An array of featurization (transformation) information for the fields of a dataset.

Type: Array of Featurization objects

Array Members: Minimum number of 1 item. Maximum number of 50 items.

Required: No

ForecastDimensions

An array of dimension (field) names that specify how to group the generated forecast.

For example, suppose that you are generating a forecast for item sales across all of your stores, and your dataset contains a `store_id` field. If you want the sales forecast for each item by store, you would specify `store_id` as the dimension.

All forecast dimensions specified in the `TARGET_TIME_SERIES` dataset don't need to be specified in the `CreatePredictor` request. All forecast dimensions specified in the `RELATED_TIME_SERIES` dataset must be specified in the `CreatePredictor` request.

Type: Array of strings

Array Members: Minimum number of 1 item. Maximum number of 10 items.

Length Constraints: Minimum length of 1. Maximum length of 63.

Pattern: `^[a-zA-Z][a-zA-Z0-9_]*`

Required: No

See Also

For more information about using this API in one of the language-specific AWS SDKs, see the following:

- AWS SDK for C++
- AWS SDK for Java V2
- AWS SDK for Ruby V3

FeaturizationMethod

Service: Amazon Forecast Service

Provides information about the method that featurizes (transforms) a dataset field. The method is part of the `FeaturizationPipeline` of the Featurization object.

The following is an example of how you specify a `FeaturizationMethod` object.

```
{

"FeaturizationMethodName": "filling",

"FeaturizationMethodParameters": {"aggregation": "sum", "middlefill":
"zero", "backfill": "zero"}

}
```

Contents

FeaturizationMethodName

The name of the method. The "filling" method is the only supported method.

Type: String

Valid Values: `filling`

Required: Yes

FeaturizationMethodParameters

The method parameters (key-value pairs), which are a map of override parameters. Specify these parameters to override the default values. Related Time Series attributes do not accept aggregation parameters.

The following list shows the parameters and their valid values for the "filling" featurization method for a **Target Time Series** dataset. Bold signifies the default value.

- `aggregation`: **sum**, `avg`, `first`, `min`, `max`
- `frontfill`: **none**
- `middlefill`: **zero**, `nan` (not a number), `value`, `median`, `mean`, `min`, `max`
- `backfill`: **zero**, `nan`, `value`, `median`, `mean`, `min`, `max`

The following list shows the parameters and their valid values for a **Related Time Series** featurization method (there are no defaults):

- `middlefill`: `zero`, `value`, `median`, `mean`, `min`, `max`
- `backfill`: `zero`, `value`, `median`, `mean`, `min`, `max`
- `futurefill`: `zero`, `value`, `median`, `mean`, `min`, `max`

To set a filling method to a specific value, set the fill parameter to `value` and define the value in a corresponding `_value` parameter. For example, to set backfilling to a value of 2, include the following: `"backfill": "value"` and `"backfill_value":"2"`.

Type: String to string map

Map Entries: Maximum number of 20 items.

Key Length Constraints: Maximum length of 256.

Key Pattern: `^[a-zA-Z0-9\-\_\.\/\[\]\,\\]+$`

Value Length Constraints: Maximum length of 256.

Value Pattern: `^[a-zA-Z0-9\-\_\.\/\[\]\,\"\\\s]+$`

Required: No

See Also

For more information about using this API in one of the language-specific AWS SDKs, see the following:

- AWS SDK for C++
- AWS SDK for Java V2
- AWS SDK for Ruby V3

Filter

Service: Amazon Forecast Service

Describes a filter for choosing a subset of objects. Each filter consists of a condition and a match statement. The condition is either `IS` or `IS_NOT`, which specifies whether to include or exclude the objects that match the statement, respectively. The match statement consists of a key and a value.

Contents

Condition

The condition to apply. To include the objects that match the statement, specify `IS`. To exclude matching objects, specify `IS_NOT`.

Type: String

Valid Values: `IS | IS_NOT`

Required: Yes

Key

The name of the parameter to filter on.

Type: String

Length Constraints: Maximum length of 256.

Pattern: `^[a-zA-Z0-9\_]+$`

Required: Yes

Value

The value to match.

Type: String

Length Constraints: Maximum length of 256.

Pattern: `arn:([a-z\d-]+):forecast:.*:.*:.+`

Required: Yes

See Also

For more information about using this API in one of the language-specific AWS SDKs, see the following:

- AWS SDK for C++
- AWS SDK for Java V2
- AWS SDK for Ruby V3

ForecastExportJobSummary

Service: Amazon Forecast Service

Provides a summary of the forecast export job properties used in the ListForecastExportJobs operation. To get the complete set of properties, call the DescribeForecastExportJob operation, and provide the listed `ForecastExportJobArn`.

Contents

CreationTime

When the forecast export job was created.

Type: Timestamp

Required: No

Destination

The path to the Amazon Simple Storage Service (Amazon S3) bucket where the forecast is exported.

Type: DataDestination object

Required: No

ForecastExportJobArn

The Amazon Resource Name (ARN) of the forecast export job.

Type: String

Length Constraints: Maximum length of 256.

Pattern: `arn:([a-z\d-]+):forecast:.*:.*:.+`

Required: No

ForecastExportJobName

The name of the forecast export job.

Type: String

Length Constraints: Minimum length of 1. Maximum length of 63.

Pattern: `^[a-zA-Z][a-zA-Z0-9_]*`

Required: No

LastModificationTime

The last time the resource was modified. The timestamp depends on the status of the job:

- `CREATE_PENDING` - The `CreationTime`.
- `CREATE_IN_PROGRESS` - The current timestamp.
- `CREATE_STOPPING` - The current timestamp.
- `CREATE_STOPPED` - When the job stopped.
- `ACTIVE` or `CREATE_FAILED` - When the job finished or failed.

Type: Timestamp

Required: No

Message

If an error occurred, an informational message about the error.

Type: String

Required: No

Status

The status of the forecast export job. States include:

- `ACTIVE`
- `CREATE_PENDING`, `CREATE_IN_PROGRESS`, `CREATE_FAILED`
- `CREATE_STOPPING`, `CREATE_STOPPED`
- `DELETE_PENDING`, `DELETE_IN_PROGRESS`, `DELETE_FAILED`

> **Note**
>
> The `Status` of the forecast export job must be `ACTIVE` before you can access the forecast in your S3 bucket.

Type: String

Length Constraints: Maximum length of 256.

Required: No

See Also

For more information about using this API in one of the language-specific AWS SDKs, see the following:

- AWS SDK for C++
- AWS SDK for Java V2
- AWS SDK for Ruby V3

ForecastSummary

Service: Amazon Forecast Service

Provides a summary of the forecast properties used in the ListForecasts operation. To get the complete set of properties, call the DescribeForecast operation, and provide the `ForecastArn` that is listed in the summary.

Contents

CreatedUsingAutoPredictor

Whether the Forecast was created from an AutoPredictor.

Type: Boolean

Required: No

CreationTime

When the forecast creation task was created.

Type: Timestamp

Required: No

DatasetGroupArn

The Amazon Resource Name (ARN) of the dataset group that provided the data used to train the predictor.

Type: String

Length Constraints: Maximum length of 256.

Pattern: `^[a-zA-Z0-9\_]+$`

Required: No

ForecastArn

The ARN of the forecast.

Type: String

Length Constraints: Maximum length of 256.

Pattern: `arn:([a-z\d-]+):forecast:.*:.*:.+`

Required: No

ForecastName

The name of the forecast.

Type: String

Length Constraints: Minimum length of 1. Maximum length of 63.

Pattern: `^[a-zA-Z][a-zA-Z0-9_]*`

Required: No

LastModificationTime

The last time the resource was modified. The timestamp depends on the status of the job:

- `CREATE_PENDING` - The `CreationTime`.
- `CREATE_IN_PROGRESS` - The current timestamp.
- `CREATE_STOPPING` - The current timestamp.
- `CREATE_STOPPED` - When the job stopped.
- `ACTIVE` or `CREATE_FAILED` - When the job finished or failed.

Type: Timestamp

Required: No

Message

If an error occurred, an informational message about the error.

Type: String

Required: No

PredictorArn

The ARN of the predictor used to generate the forecast.

Type: String

Length Constraints: Maximum length of 256.

Pattern: `^[a-zA-Z0-9\_]+$`

Required: No

Status

The status of the forecast. States include:

- `ACTIVE`
- `CREATE_PENDING`, `CREATE_IN_PROGRESS`, `CREATE_FAILED`
- `CREATE_STOPPING`, `CREATE_STOPPED`
- `DELETE_PENDING`, `DELETE_IN_PROGRESS`, `DELETE_FAILED`

> **Note**
>
> The `Status` of the forecast must be `ACTIVE` before you can query or export the forecast.

Type: String

Length Constraints: Maximum length of 256.

Required: No

See Also

For more information about using this API in one of the language-specific AWS SDKs, see the following:

- AWS SDK for C++
- AWS SDK for Java V2
- AWS SDK for Ruby V3

HyperParameterTuningJobConfig

Service: Amazon Forecast Service

Configuration information for a hyperparameter tuning job. You specify this object in the CreatePredictor request.

A *hyperparameter* is a parameter that governs the model training process. You set hyperparameters before training starts, unlike model parameters, which are determined during training. The values of the hyperparameters effect which values are chosen for the model parameters.

In a *hyperparameter tuning job*, Amazon Forecast chooses the set of hyperparameter values that optimize a specified metric. Forecast accomplishes this by running many training jobs over a range of hyperparameter values. The optimum set of values depends on the algorithm, the training data, and the specified metric objective.

Contents

ParameterRanges

Specifies the ranges of valid values for the hyperparameters.

Type: ParameterRanges object

Required: No

See Also

For more information about using this API in one of the language-specific AWS SDKs, see the following:

- AWS SDK for C++
- AWS SDK for Java V2
- AWS SDK for Ruby V3

InputDataConfig

Service: Amazon Forecast Service

> **Note**
>
> This object belongs to the CreatePredictor operation. If you created your predictor with CreateAutoPredictor, see DataConfig.

The data used to train a predictor. The data includes a dataset group and any supplementary features. You specify this object in the CreatePredictor request.

Contents

DatasetGroupArn

The Amazon Resource Name (ARN) of the dataset group.

Type: String

Length Constraints: Maximum length of 256.

Pattern: `arn:([a-z\d-]+):forecast:.*:.*:.+`

Required: Yes

SupplementaryFeatures

An array of supplementary features. The only supported feature is a holiday calendar.

Type: Array of SupplementaryFeature objects

Array Members: Minimum number of 1 item. Maximum number of 2 items.

Required: No

See Also

For more information about using this API in one of the language-specific AWS SDKs, see the following:

- AWS SDK for C++

- AWS SDK for Java V2
- AWS SDK for Ruby V3

IntegerParameterRange

Service: Amazon Forecast Service

Specifies an integer hyperparameter and it's range of tunable values. This object is part of the ParameterRanges object.

Contents

MaxValue

The maximum tunable value of the hyperparameter.

Type: Integer

Required: Yes

MinValue

The minimum tunable value of the hyperparameter.

Type: Integer

Required: Yes

Name

The name of the hyperparameter to tune.

Type: String

Length Constraints: Minimum length of 1. Maximum length of 63.

Pattern: `^[a-zA-Z][a-zA-Z0-9_]*`

Required: Yes

ScalingType

The scale that hyperparameter tuning uses to search the hyperparameter range. Valid values:

Auto

Amazon Forecast hyperparameter tuning chooses the best scale for the hyperparameter.

Linear

Hyperparameter tuning searches the values in the hyperparameter range by using a linear scale.

Logarithmic

Hyperparameter tuning searches the values in the hyperparameter range by using a logarithmic scale.

Logarithmic scaling works only for ranges that have values greater than 0.

ReverseLogarithmic

Not supported for `IntegerParameterRange`.

Reverse logarithmic scaling works only for ranges that are entirely within the range 0 <= x < 1.0.

For information about choosing a hyperparameter scale, see Hyperparameter Scaling. One of the following values:

Type: String

Valid Values: `Auto | Linear | Logarithmic | ReverseLogarithmic`

Required: No

See Also

For more information about using this API in one of the language-specific AWS SDKs, see the following:

- AWS SDK for C++
- AWS SDK for Java V2
- AWS SDK for Ruby V3

MetricResult

Service: Amazon Forecast Service

An individual metric Forecast calculated when monitoring predictor usage. You can compare the value for this metric to the metric's value in the Baseline to see how your predictor's performance is changing.

For more information about metrics generated by Forecast see Evaluating Predictor Accuracy

Contents

MetricName

The name of the metric.

Type: String

Length Constraints: Maximum length of 256.

Required: No

MetricValue

The value for the metric.

Type: Double

Required: No

See Also

For more information about using this API in one of the language-specific AWS SDKs, see the following:

- AWS SDK for C++
- AWS SDK for Java V2
- AWS SDK for Ruby V3

Metrics

Service: Amazon Forecast Service

Provides metrics that are used to evaluate the performance of a predictor. This object is part of the WindowSummary object.

Contents

AverageWeightedQuantileLoss

The average value of all weighted quantile losses.

Type: Double

Required: No

ErrorMetrics

Provides detailed error metrics for each forecast type. Metrics include root-mean square-error (RMSE), mean absolute percentage error (MAPE), mean absolute scaled error (MASE), and weighted average percentage error (WAPE).

Type: Array of ErrorMetric objects

Required: No

RMSE

This member has been deprecated.

The root-mean-square error (RMSE).

Type: Double

Required: No

WeightedQuantileLosses

An array of weighted quantile losses. Quantiles divide a probability distribution into regions of equal probability. The distribution in this case is the loss function.

Type: Array of WeightedQuantileLoss objects

Required: No

See Also

For more information about using this API in one of the language-specific AWS SDKs, see the following:

- AWS SDK for C++
- AWS SDK for Java V2
- AWS SDK for Ruby V3

MonitorConfig

Service: Amazon Forecast Service

The configuration details for the predictor monitor.

Contents

MonitorName

The name of the monitor resource.

Type: String

Length Constraints: Minimum length of 1. Maximum length of 63.

Pattern: `^[a-zA-Z][a-zA-Z0-9_]*`

Required: Yes

See Also

For more information about using this API in one of the language-specific AWS SDKs, see the following:

- AWS SDK for C++
- AWS SDK for Java V2
- AWS SDK for Ruby V3

MonitorDataSource

Service: Amazon Forecast Service

The source of the data the monitor used during the evaluation.

Contents

DatasetImportJobArn

The Amazon Resource Name (ARN) of the dataset import job used to import the data that initiated the monitor evaluation.

Type: String

Length Constraints: Maximum length of 256.

Pattern: `arn:([a-z\d-]+):forecast:.*:.*:.+`

Required: No

ForecastArn

The Amazon Resource Name (ARN) of the forecast the monitor used during the evaluation.

Type: String

Length Constraints: Maximum length of 256.

Pattern: `arn:([a-z\d-]+):forecast:.*:.*:.+`

Required: No

PredictorArn

The Amazon Resource Name (ARN) of the predictor resource you are monitoring.

Type: String

Length Constraints: Maximum length of 256.

Pattern: `arn:([a-z\d-]+):forecast:.*:.*:.+`

Required: No

See Also

For more information about using this API in one of the language-specific AWS SDKs, see the following:

- AWS SDK for C++
- AWS SDK for Java V2
- AWS SDK for Ruby V3

MonitorInfo

Service: Amazon Forecast Service

Provides information about the monitor resource.

Contents

MonitorArn

The Amazon Resource Name (ARN) of the monitor resource.

Type: String

Length Constraints: Maximum length of 256.

Pattern: `arn:([a-z\d-]+):forecast:.*:.*:.+`

Required: No

Status

The status of the monitor. States include:

- `ACTIVE`
- `ACTIVE_STOPPING`, `ACTIVE_STOPPED`
- `UPDATE_IN_PROGRESS`
- `CREATE_PENDING`, `CREATE_IN_PROGRESS`, `CREATE_FAILED`
- `DELETE_PENDING`, `DELETE_IN_PROGRESS`, `DELETE_FAILED`

Type: String

Length Constraints: Maximum length of 256.

Required: No

See Also

For more information about using this API in one of the language-specific AWS SDKs, see the following:

- AWS SDK for C++
- AWS SDK for Java V2

- AWS SDK for Ruby V3

MonitorSummary

Service: Amazon Forecast Service

Provides a summary of the monitor properties used in the ListMonitors operation. To get a complete set of properties, call the DescribeMonitor operation, and provide the listed `MonitorArn`.

Contents

CreationTime

When the monitor resource was created.

Type: Timestamp

Required: No

LastModificationTime

The last time the monitor resource was modified. The timestamp depends on the status of the job:

- `CREATE_PENDING` - The `CreationTime`.
- `CREATE_IN_PROGRESS` - The current timestamp.
- `STOPPED` - When the resource stopped.
- `ACTIVE` or `CREATE_FAILED` - When the monitor creation finished or failed.

Type: Timestamp

Required: No

MonitorArn

The Amazon Resource Name (ARN) of the monitor resource.

Type: String

Length Constraints: Maximum length of 256.

Pattern: `arn:([a-z\d-]+):forecast:.*:.*:.+`

Required: No

MonitorName

The name of the monitor resource.

Type: String

Length Constraints: Minimum length of 1. Maximum length of 63.

Pattern: `^[a-zA-Z][a-zA-Z0-9_]*`

Required: No

ResourceArn

The Amazon Resource Name (ARN) of the predictor being monitored.

Type: String

Length Constraints: Maximum length of 256.

Pattern: `arn:([a-z\d-]+):forecast:.*:.*:.+`

Required: No

Status

The status of the monitor. States include:

- `ACTIVE`
- `ACTIVE_STOPPING`, `ACTIVE_STOPPED`
- `UPDATE_IN_PROGRESS`
- `CREATE_PENDING`, `CREATE_IN_PROGRESS`, `CREATE_FAILED`
- `DELETE_PENDING`, `DELETE_IN_PROGRESS`, `DELETE_FAILED`

Type: String

Length Constraints: Maximum length of 256.

Required: No

See Also

For more information about using this API in one of the language-specific AWS SDKs, see the following:

- AWS SDK for C++
- AWS SDK for Java V2
- AWS SDK for Ruby V3

ParameterRanges

Service: Amazon Forecast Service

Specifies the categorical, continuous, and integer hyperparameters, and their ranges of tunable values. The range of tunable values determines which values that a hyperparameter tuning job can choose for the specified hyperparameter. This object is part of the HyperParameterTuningJobConfig object.

Contents

CategoricalParameterRanges

Specifies the tunable range for each categorical hyperparameter.

Type: Array of CategoricalParameterRange objects

Array Members: Minimum number of 1 item. Maximum number of 20 items.

Required: No

ContinuousParameterRanges

Specifies the tunable range for each continuous hyperparameter.

Type: Array of ContinuousParameterRange objects

Array Members: Minimum number of 1 item. Maximum number of 20 items.

Required: No

IntegerParameterRanges

Specifies the tunable range for each integer hyperparameter.

Type: Array of IntegerParameterRange objects

Array Members: Minimum number of 1 item. Maximum number of 20 items.

Required: No

See Also

For more information about using this API in one of the language-specific AWS SDKs, see the following:

- AWS SDK for C++
- AWS SDK for Java V2
- AWS SDK for Ruby V3

PredictorBacktestExportJobSummary

Service: Amazon Forecast Service

Provides a summary of the predictor backtest export job properties used in the ListPredictorBacktestExportJobs operation. To get a complete set of properties, call the DescribePredictorBacktestExportJob operation, and provide the listed `PredictorBacktestExportJobArn`.

Contents

CreationTime

When the predictor backtest export job was created.

Type: Timestamp

Required: No

Destination

The destination for an export job. Provide an S3 path, an AWS Identity and Access Management (IAM) role that allows Amazon Forecast to access the location, and an AWS Key Management Service (KMS) key (optional).

Type: DataDestination object

Required: No

LastModificationTime

The last time the resource was modified. The timestamp depends on the status of the job:

- `CREATE_PENDING` - The `CreationTime`.
- `CREATE_IN_PROGRESS` - The current timestamp.
- `CREATE_STOPPING` - The current timestamp.
- `CREATE_STOPPED` - When the job stopped.
- `ACTIVE` or `CREATE_FAILED` - When the job finished or failed.

Type: Timestamp

Required: No

Message

Information about any errors that may have occurred during the backtest export.

Type: String

Required: No

PredictorBacktestExportJobArn

The Amazon Resource Name (ARN) of the predictor backtest export job.

Type: String

Length Constraints: Maximum length of 256.

Pattern: `arn:([a-z\d-]+):forecast:.*:.*:.+`

Required: No

PredictorBacktestExportJobName

The name of the predictor backtest export job.

Type: String

Length Constraints: Minimum length of 1. Maximum length of 63.

Pattern: `^[a-zA-Z][a-zA-Z0-9_]*`

Required: No

Status

The status of the predictor backtest export job. States include:

- `ACTIVE`
- `CREATE_PENDING`, `CREATE_IN_PROGRESS`, `CREATE_FAILED`
- `CREATE_STOPPING`, `CREATE_STOPPED`
- `DELETE_PENDING`, `DELETE_IN_PROGRESS`, `DELETE_FAILED`

Type: String

Length Constraints: Maximum length of 256.

Required: No

See Also

For more information about using this API in one of the language-specific AWS SDKs, see the following:

- AWS SDK for C++
- AWS SDK for Java V2
- AWS SDK for Ruby V3

PredictorBaseline

Service: Amazon Forecast Service

Metrics you can use as a baseline for comparison purposes. Use these metrics when you interpret monitoring results for an auto predictor.

Contents

BaselineMetrics

The initial accuracy metrics for the predictor. Use these metrics as a baseline for comparison purposes as you use your predictor and the metrics change.

Type: Array of BaselineMetric objects

Required: No

See Also

For more information about using this API in one of the language-specific AWS SDKs, see the following:

- AWS SDK for C++
- AWS SDK for Java V2
- AWS SDK for Ruby V3

PredictorEvent

Service: Amazon Forecast Service

Provides details about a predictor event, such as a retraining.

Contents

Datetime

The timestamp for when the event occurred.

Type: Timestamp

Required: No

Detail

The type of event. For example, `Retrain`. A retraining event denotes the timepoint when a predictor was retrained. Any monitor results from before the `Datetime` are from the previous predictor. Any new metrics are for the newly retrained predictor.

Type: String

Length Constraints: Maximum length of 256.

Required: No

See Also

For more information about using this API in one of the language-specific AWS SDKs, see the following:

- AWS SDK for C++
- AWS SDK for Java V2
- AWS SDK for Ruby V3

PredictorExecution

Service: Amazon Forecast Service

The algorithm used to perform a backtest and the status of those tests.

Contents

AlgorithmArn

The ARN of the algorithm used to test the predictor.

Type: String

Length Constraints: Maximum length of 256.

Pattern: `arn:([a-z\d-]+):forecast:.*:.*:.+`

Required: No

TestWindows

An array of test windows used to evaluate the algorithm. The `NumberOfBacktestWindows` from the EvaluationParameters object determines the number of windows in the array.

Type: Array of TestWindowSummary objects

Required: No

See Also

For more information about using this API in one of the language-specific AWS SDKs, see the following:

- AWS SDK for C++
- AWS SDK for Java V2
- AWS SDK for Ruby V3

PredictorExecutionDetails

Service: Amazon Forecast Service

Contains details on the backtests performed to evaluate the accuracy of the predictor. The tests are returned in descending order of accuracy, with the most accurate backtest appearing first. You specify the number of backtests to perform when you call the CreatePredictor operation.

Contents

PredictorExecutions

An array of the backtests performed to evaluate the accuracy of the predictor against a particular algorithm. The `NumberOfBacktestWindows` from the EvaluationParameters object determines the number of windows in the array.

Type: Array of PredictorExecution objects

Array Members: Minimum number of 1 item. Maximum number of 5 items.

Required: No

See Also

For more information about using this API in one of the language-specific AWS SDKs, see the following:

- AWS SDK for C++
- AWS SDK for Java V2
- AWS SDK for Ruby V3

PredictorMonitorEvaluation

Service: Amazon Forecast Service

Describes the results of a monitor evaluation.

Contents

EvaluationState

The status of the monitor evaluation. The state can be `SUCCESS` or `FAILURE`.

Type: String

Length Constraints: Maximum length of 256.

Required: No

EvaluationTime

The timestamp that indicates when the monitor evaluation was started.

Type: Timestamp

Required: No

Message

Information about any errors that may have occurred during the monitor evaluation.

Type: String

Required: No

MetricResults

A list of metrics Forecast calculated when monitoring a predictor. You can compare the value for each metric in the list to the metric's value in the Baseline to see how your predictor's performance is changing.

Type: Array of MetricResult objects

Required: No

MonitorArn

The Amazon Resource Name (ARN) of the monitor resource.

Type: String

Length Constraints: Maximum length of 256.

Pattern: `arn:([a-z\d-]+):forecast:.*:.*:.+`

Required: No

MonitorDataSource

The source of the data the monitor resource used during the evaluation.

Type: MonitorDataSource object

Required: No

NumItemsEvaluated

The number of items considered during the evaluation.

Type: Long

Required: No

PredictorEvent

Provides details about a predictor event, such as a retraining.

Type: PredictorEvent object

Required: No

ResourceArn

The Amazon Resource Name (ARN) of the resource to monitor.

Type: String

Length Constraints: Maximum length of 256.

Pattern: `arn:([a-z\d-]+):forecast:.*:.*:.+`

Required: No

WindowEndDatetime

The timestamp that indicates the end of the window that is used for monitor evaluation.

Type: Timestamp

Required: No

WindowStartDatetime

The timestamp that indicates the start of the window that is used for monitor evaluation.

Type: Timestamp

Required: No

See Also

For more information about using this API in one of the language-specific AWS SDKs, see the following:

- AWS SDK for C++
- AWS SDK for Java V2
- AWS SDK for Ruby V3

PredictorSummary

Service: Amazon Forecast Service

Provides a summary of the predictor properties that are used in the ListPredictors operation. To get the complete set of properties, call the DescribePredictor operation, and provide the listed `PredictorArn`.

Contents

CreationTime

When the model training task was created.

Type: Timestamp

Required: No

DatasetGroupArn

The Amazon Resource Name (ARN) of the dataset group that contains the data used to train the predictor.

Type: String

Length Constraints: Maximum length of 256.

Pattern: `arn:([a-z\d-]+):forecast:.*:.*:.+`

Required: No

IsAutoPredictor

Whether AutoPredictor was used to create the predictor.

Type: Boolean

Required: No

LastModificationTime

The last time the resource was modified. The timestamp depends on the status of the job:

- `CREATE_PENDING` - The `CreationTime`.
- `CREATE_IN_PROGRESS` - The current timestamp.
- `CREATE_STOPPING` - The current timestamp.

- `CREATE_STOPPED` - When the job stopped.
- `ACTIVE` or `CREATE_FAILED` - When the job finished or failed.

Type: Timestamp

Required: No

Message

If an error occurred, an informational message about the error.

Type: String

Required: No

PredictorArn

The ARN of the predictor.

Type: String

Length Constraints: Maximum length of 256.

Pattern: `arn:([a-z\d-]+):forecast:.*:.*:.+`

Required: No

PredictorName

The name of the predictor.

Type: String

Length Constraints: Minimum length of 1. Maximum length of 63.

Pattern: `^[a-zA-Z][a-zA-Z0-9_]*`

Required: No

ReferencePredictorSummary

A summary of the reference predictor used if the predictor was retrained or upgraded.

Type: ReferencePredictorSummary object

Required: No

Status

The status of the predictor. States include:

- `ACTIVE`
- `CREATE_PENDING`, `CREATE_IN_PROGRESS`, `CREATE_FAILED`
- `DELETE_PENDING`, `DELETE_IN_PROGRESS`, `DELETE_FAILED`
- `CREATE_STOPPING`, `CREATE_STOPPED`

> **Note**
>
> The `Status` of the predictor must be `ACTIVE` before you can use the predictor to create a forecast.

Type: String

Length Constraints: Maximum length of 256.

Required: No

See Also

For more information about using this API in one of the language-specific AWS SDKs, see the following:

- AWS SDK for C++
- AWS SDK for Java V2
- AWS SDK for Ruby V3

ReferencePredictorSummary

Service: Amazon Forecast Service

Provides a summary of the reference predictor used when retraining or upgrading a predictor.

Contents

Arn

The ARN of the reference predictor.

Type: String

Length Constraints: Maximum length of 256.

Pattern: `arn:([a-z\d-]+):forecast:.*:.*:.+`

Required: No

State

Whether the reference predictor is `Active` or `Deleted`.

Type: String

Valid Values: `Active | Deleted`

Required: No

See Also

For more information about using this API in one of the language-specific AWS SDKs, see the following:

- AWS SDK for C++
- AWS SDK for Java V2
- AWS SDK for Ruby V3

S3Config

Service: Amazon Forecast Service

The path to the file(s) in an Amazon Simple Storage Service (Amazon S3) bucket, and an AWS Identity and Access Management (IAM) role that Amazon Forecast can assume to access the file(s). Optionally, includes an AWS Key Management Service (KMS) key. This object is part of the DataSource object that is submitted in the CreateDatasetImportJob request, and part of the DataDestination object.

Contents

Path

The path to an Amazon Simple Storage Service (Amazon S3) bucket or file(s) in an Amazon S3 bucket.

Type: String

Length Constraints: Minimum length of 7. Maximum length of 4096.

Pattern: `^s3://[a-z0-9].+$`

Required: Yes

RoleArn

The ARN of the AWS Identity and Access Management (IAM) role that Amazon Forecast can assume to access the Amazon S3 bucket or files. If you provide a value for the `KMSKeyArn` key, the role must allow access to the key.

Passing a role across AWS accounts is not allowed. If you pass a role that isn't in your account, you get an `InvalidInputException` error.

Type: String

Length Constraints: Maximum length of 256.

Pattern: `arn:([a-z\d-]+):forecast:.*:.*:.+`

Required: Yes

KMSKeyArn

The Amazon Resource Name (ARN) of an AWS Key Management Service (KMS) key.

Type: String

Length Constraints: Maximum length of 256.

Pattern: `arn:aws:kms:.*:key/.*`

Required: No

See Also

For more information about using this API in one of the language-specific AWS SDKs, see the following:

- AWS SDK for C++
- AWS SDK for Java V2
- AWS SDK for Ruby V3

Schema

Service: Amazon Forecast Service

Defines the fields of a dataset.

Contents

Attributes

An array of attributes specifying the name and type of each field in a dataset.

Type: Array of SchemaAttribute objects

Array Members: Minimum number of 1 item. Maximum number of 100 items.

Required: No

See Also

For more information about using this API in one of the language-specific AWS SDKs, see the following:

- AWS SDK for C++
- AWS SDK for Java V2
- AWS SDK for Ruby V3

SchemaAttribute

Service: Amazon Forecast Service

An attribute of a schema, which defines a dataset field. A schema attribute is required for every field in a dataset. The Schema object contains an array of `SchemaAttribute` objects.

Contents

AttributeName

The name of the dataset field.

Type: String

Length Constraints: Minimum length of 1. Maximum length of 63.

Pattern: `^[a-zA-Z][a-zA-Z0-9_]*`

Required: No

AttributeType

The data type of the field.

For a related time series dataset, other than date, item_id, and forecast dimensions attributes, all attributes should be of numerical type (integer/float).

Type: String

Valid Values: `string | integer | float | timestamp | geolocation`

Required: No

See Also

For more information about using this API in one of the language-specific AWS SDKs, see the following:

- AWS SDK for C++
- AWS SDK for Java V2
- AWS SDK for Ruby V3

Statistics

Service: Amazon Forecast Service

Provides statistics for each data field imported into to an Amazon Forecast dataset with the CreateDatasetImportJob operation.

Contents

Avg

For a numeric field, the average value in the field.

Type: Double

Required: No

Count

The number of values in the field. If the response value is -1, refer to `CountLong`.

Type: Integer

Required: No

CountDistinct

The number of distinct values in the field. If the response value is -1, refer to `CountDistinctLong`.

Type: Integer

Required: No

CountDistinctLong

The number of distinct values in the field. `CountDistinctLong` is used instead of `CountDistinct` if the value is greater than 2,147,483,647.

Type: Long

Required: No

CountLong

The number of values in the field. `CountLong` is used instead of `Count` if the value is greater than 2,147,483,647.

Type: Long

Required: No

CountNan

The number of NAN (not a number) values in the field. If the response value is -1, refer to `CountNanLong`.

Type: Integer

Required: No

CountNanLong

The number of NAN (not a number) values in the field. `CountNanLong` is used instead of `CountNan` if the value is greater than 2,147,483,647.

Type: Long

Required: No

CountNull

The number of null values in the field. If the response value is -1, refer to `CountNullLong`.

Type: Integer

Required: No

CountNullLong

The number of null values in the field. `CountNullLong` is used instead of `CountNull` if the value is greater than 2,147,483,647.

Type: Long

Required: No

Max

For a numeric field, the maximum value in the field.

Type: String

Length Constraints: Maximum length of 256.

Pattern: ^[a-zA-Z0-9_]+$

Required: No

Min

For a numeric field, the minimum value in the field.

Type: String

Length Constraints: Maximum length of 256.

Pattern: ^[a-zA-Z0-9_]+$

Required: No

Stddev

For a numeric field, the standard deviation.

Type: Double

Required: No

See Also

For more information about using this API in one of the language-specific AWS SDKs, see the following:

- AWS SDK for C++
- AWS SDK for Java V2
- AWS SDK for Ruby V3

SupplementaryFeature

Service: Amazon Forecast Service

> **Note**
>
> This object belongs to the CreatePredictor operation. If you created your predictor with CreateAutoPredictor, see AdditionalDataset.

Describes a supplementary feature of a dataset group. This object is part of the InputDataConfig object. Forecast supports the Weather Index and Holidays built-in featurizations.

Weather Index

The Amazon Forecast Weather Index is a built-in featurization that incorporates historical and projected weather information into your model. The Weather Index supplements your datasets with over two years of historical weather data and up to 14 days of projected weather data. For more information, see Amazon Forecast Weather Index.

Holidays

Holidays is a built-in featurization that incorporates a feature-engineered dataset of national holiday information into your model. It provides native support for the holiday calendars of over 250 countries. Amazon Forecast incorporates both the Holiday API library and Jollyday API to generate holiday calendars. For more information, see Holidays Featurization.

Contents

Name

The name of the feature. Valid values: `"holiday"` and `"weather"`.

Type: String

Length Constraints: Minimum length of 1. Maximum length of 63.

Pattern: `^[a-zA-Z][a-zA-Z0-9_]*`

Required: Yes

Value

Weather Index

To enable the Weather Index, set the value to `"true"`

Holidays

To enable Holidays, specify a country with one of the following two-letter country codes:

- Afghanistan - AF
- Åland Islands - AX
- Albania - AL
- Algeria - DZ
- American Samoa - AS
- Andorra - AD
- Angola - AO
- Anguilla - AI
- Antartica - AQ
- Antigua and Barbuda - AG
- Argentina - AR
- Armenia - AM
- Aruba - AW
- Australia - AU
- Austria - AT
- Azerbaijan - AZ
- Bahamas - BS
- Bahrain - BH
- Bangladesh - BD
- Barbados - BB
- Belarus - BY
- Belgium - BE
- Belize - BZ
- Benin - BJ
- Bermuda - BM
- Bhutan - BT
- Bolivia - BO

- Bosnia and Herzegovina - BA
- Botswana - BW
- Bouvet Island - BV
- Brazil - BR
- British Indian Ocean Territory - IO
- British Virgin Islands - VG
- Brunei Darussalam - BN
- Bulgaria - BG
- Burkina Faso - BF
- Burundi - BI
- Cambodia - KH
- Cameroon - CM
- Canada - CA
- Cape Verde - CV
- Caribbean Netherlands - BQ
- Cayman Islands - KY
- Central African Republic - CF
- Chad - TD
- Chile - CL
- China - CN
- Christmas Island - CX
- Cocos (Keeling) Islands - CC
- Colombia - CO
- Comoros - KM
- Cook Islands - CK
- Costa Rica - CR
- Croatia - HR
- Cuba - CU
- Curaçao - CW
- Cyprus - CY

- Czechia - CZ
- Democratic Republic of the Congo - CD
- Denmark - DK
- Djibouti - DJ
- Dominica - DM
- Dominican Republic - DO
- Ecuador - EC
- Egypt - EG
- El Salvador - SV
- Equatorial Guinea - GQ
- Eritrea - ER
- Estonia - EE
- Eswatini - SZ
- Ethiopia - ET
- Falkland Islands - FK
- Faroe Islands - FO
- Fiji - FJ
- Finland - FI
- France - FR
- French Guiana - GF
- French Polynesia - PF
- French Southern Territories - TF
- Gabon - GA
- Gambia - GM
- Georgia - GE
- Germany - DE
- Ghana - GH
- Gibraltar - GI
- Greece - GR
- Greenland - GL

- Grenada - GD
- Guadeloupe - GP
- Guam - GU
- Guatemala - GT
- Guernsey - GG
- Guinea - GN
- Guinea-Bissau - GW
- Guyana - GY
- Haiti - HT
- Heard Island and McDonald Islands - HM
- Honduras - HN
- Hong Kong - HK
- Hungary - HU
- Iceland - IS
- India - IN
- Indonesia - ID
- Iran - IR
- Iraq - IQ
- Ireland - IE
- Isle of Man - IM
- Israel - IL
- Italy - IT
- Ivory Coast - CI
- Jamaica - JM
- Japan - JP
- Jersey - JE
- Jordan - JO
- Kazakhstan - KZ
- Kenya - KE
- Kiribati - KI

- Kosovo - XK
- Kuwait - KW
- Kyrgyzstan - KG
- Laos - LA
- Latvia - LV
- Lebanon - LB
- Lesotho - LS
- Liberia - LR
- Libya - LY
- Liechtenstein - LI
- Lithuania - LT
- Luxembourg - LU
- Macao - MO
- Madagascar - MG
- Malawi - MW
- Malaysia - MY
- Maldives - MV
- Mali - ML
- Malta - MT
- Marshall Islands - MH
- Martinique - MQ
- Mauritania - MR
- Mauritius - MU
- Mayotte - YT
- Mexico - MX
- Micronesia - FM
- Moldova - MD
- Monaco - MC
- Mongolia - MN
- Montenegro - ME

- Montserrat - MS
- Morocco - MA
- Mozambique - MZ
- Myanmar - MM
- Namibia - NA
- Nauru - NR
- Nepal - NP
- Netherlands - NL
- New Caledonia - NC
- New Zealand - NZ
- Nicaragua - NI
- Niger - NE
- Nigeria - NG
- Niue - NU
- Norfolk Island - NF
- North Korea - KP
- North Macedonia - MK
- Northern Mariana Islands - MP
- Norway - NO
- Oman - OM
- Pakistan - PK
- Palau - PW
- Palestine - PS
- Panama - PA
- Papua New Guinea - PG
- Paraguay - PY
- Peru - PE
- Philippines - PH
- Pitcairn Islands - PN
- Poland - PL

- Portugal - PT
- Puerto Rico - PR
- Qatar - QA
- Republic of the Congo - CG
- Réunion - RE
- Romania - RO
- Russian Federation - RU
- Rwanda - RW
- Saint Barthélemy - BL
- "Saint Helena, Ascension and Tristan da Cunha " - SH
- Saint Kitts and Nevis - KN
- Saint Lucia - LC
- Saint Martin - MF
- Saint Pierre and Miquelon - PM
- Saint Vincent and the Grenadines - VC
- Samoa - WS
- San Marino - SM
- Sao Tome and Principe - ST
- Saudi Arabia - SA
- Senegal - SN
- Serbia - RS
- Seychelles - SC
- Sierra Leone - SL
- Singapore - SG
- Sint Maarten - SX
- Slovakia - SK
- Slovenia - SI
- Solomon Islands - SB
- Somalia - SO
- South Africa - ZA

- South Georgia and the South Sandwich Islands - GS
- South Korea - KR
- South Sudan - SS
- Spain - ES
- Sri Lanka - LK
- Sudan - SD
- Suriname - SR
- Svalbard and Jan Mayen - SJ
- Sweden - SE
- Switzerland - CH
- Syrian Arab Republic - SY
- Taiwan - TW
- Tajikistan - TJ
- Tanzania - TZ
- Thailand - TH
- Timor-Leste - TL
- Togo - TG
- Tokelau - TK
- Tonga - TO
- Trinidad and Tobago - TT
- Tunisia - TN
- Turkey - TR
- Turkmenistan - TM
- Turks and Caicos Islands - TC
- Tuvalu - TV
- Uganda - UG
- Ukraine - UA
- United Arab Emirates - AE
- United Kingdom - GB
- United Nations - UN

- United States - US
- United States Minor Outlying Islands - UM
- United States Virgin Islands - VI
- Uruguay - UY
- Uzbekistan - UZ
- Vanuatu - VU
- Vatican City - VA
- Venezuela - VE
- Vietnam - VN
- Wallis and Futuna - WF
- Western Sahara - EH
- Yemen - YE
- Zambia - ZM
- Zimbabwe - ZW

Type: String

Length Constraints: Maximum length of 256.

Pattern: `^[a-zA-Z0-9\_\-]+$`

Required: Yes

See Also

For more information about using this API in one of the language-specific AWS SDKs, see the following:

- AWS SDK for C++
- AWS SDK for Java V2
- AWS SDK for Ruby V3

Tag

Service: Amazon Forecast Service

The optional metadata that you apply to a resource to help you categorize and organize them. Each tag consists of a key and an optional value, both of which you define.

The following basic restrictions apply to tags:

- Maximum number of tags per resource - 50.
- For each resource, each tag key must be unique, and each tag key can have only one value.
- Maximum key length - 128 Unicode characters in UTF-8.
- Maximum value length - 256 Unicode characters in UTF-8.
- If your tagging schema is used across multiple services and resources, remember that other services may have restrictions on allowed characters. Generally allowed characters are: letters, numbers, and spaces representable in UTF-8, and the following characters: + - = . _ : / @.
- Tag keys and values are case sensitive.
- Do not use `aws:`, `AWS:`, or any upper or lowercase combination of such as a prefix for keys as it is reserved for AWS use. You cannot edit or delete tag keys with this prefix. Values can have this prefix. If a tag value has `aws` as its prefix but the key does not, then Forecast considers it to be a user tag and will count against the limit of 50 tags. Tags with only the key prefix of `aws` do not count against your tags per resource limit.

Contents

Key

One part of a key-value pair that makes up a tag. A `key` is a general label that acts like a category for more specific tag values.

Type: String

Length Constraints: Minimum length of 1. Maximum length of 128.

Pattern: `^([\p{L}\p{Z}\p{N}_.:/=+\-@]*)$`

Required: Yes

Value

The optional part of a key-value pair that makes up a tag. A `value` acts as a descriptor within a tag category (key).

Type: String

Length Constraints: Minimum length of 0. Maximum length of 256.

Pattern: `^([\p{L}\p{Z}\p{N}_.:/=+\-@]*)$`

Required: Yes

See Also

For more information about using this API in one of the language-specific AWS SDKs, see the following:

- AWS SDK for C++
- AWS SDK for Java V2
- AWS SDK for Ruby V3

TestWindowSummary

Service: Amazon Forecast Service

The status, start time, and end time of a backtest, as well as a failure reason if applicable.

Contents

Message

If the test failed, the reason why it failed.

Type: String

Required: No

Status

The status of the test. Possible status values are:

- `ACTIVE`
- `CREATE_IN_PROGRESS`
- `CREATE_FAILED`

Type: String

Length Constraints: Maximum length of 256.

Required: No

TestWindowEnd

The time at which the test ended.

Type: Timestamp

Required: No

TestWindowStart

The time at which the test began.

Type: Timestamp

Required: No

See Also

For more information about using this API in one of the language-specific AWS SDKs, see the following:

- AWS SDK for C++
- AWS SDK for Java V2
- AWS SDK for Ruby V3

TimeAlignmentBoundary

Service: Amazon Forecast Service

The time boundary Forecast uses to align and aggregate your data to match your forecast frequency. Provide the unit of time and the time boundary as a key value pair. If you don't provide a time boundary, Forecast uses a set of Default Time Boundaries.

For more information about aggregation, see Data Aggregation for Different Forecast Frequencies. For more information setting a custom time boundary, see Specifying a Time Boundary.

Contents

DayOfMonth

The day of the month to use for time alignment during aggregation.

Type: Integer

Valid Range: Minimum value of 1. Maximum value of 28.

Required: No

DayOfWeek

The day of week to use for time alignment during aggregation. The day must be in uppercase.

Type: String

Valid Values: `MONDAY | TUESDAY | WEDNESDAY | THURSDAY | FRIDAY | SATURDAY | SUNDAY`

Required: No

Hour

The hour of day to use for time alignment during aggregation.

Type: Integer

Valid Range: Minimum value of 0. Maximum value of 23.

Required: No

Month

The month to use for time alignment during aggregation. The month must be in uppercase.

Type: String

Valid Values: `JANUARY | FEBRUARY | MARCH | APRIL | MAY | JUNE | JULY | AUGUST | SEPTEMBER | OCTOBER | NOVEMBER | DECEMBER`

Required: No

See Also

For more information about using this API in one of the language-specific AWS SDKs, see the following:

- AWS SDK for C++
- AWS SDK for Java V2
- AWS SDK for Ruby V3

TimeSeriesCondition

Service: Amazon Forecast Service

Creates a subset of items within an attribute that are modified. For example, you can use this operation to create a subset of items that cost $5 or less. To do this, you specify `"AttributeName": "price"`, `"AttributeValue": "5"`, and `"Condition": "LESS_THAN"`. Pair this operation with the Action operation within the CreateWhatIfForecast:TimeSeriesTransformations operation to define how the attribute is modified.

Contents

AttributeName

The item_id, dimension name, IM name, or timestamp that you are modifying.

Type: String

Length Constraints: Minimum length of 1. Maximum length of 63.

Pattern: `^[a-zA-Z][a-zA-Z0-9_]*`

Required: Yes

AttributeValue

The value that is applied for the chosen `Condition`.

Type: String

Length Constraints: Maximum length of 256.

Pattern: `.+`

Required: Yes

Condition

The condition to apply. Valid values are `EQUALS`, `NOT_EQUALS`, `LESS_THAN` and `GREATER_THAN`.

Type: String

Valid Values: `EQUALS | NOT_EQUALS | LESS_THAN | GREATER_THAN`

Required: Yes

See Also

For more information about using this API in one of the language-specific AWS SDKs, see the following:

- AWS SDK for C++
- AWS SDK for Java V2
- AWS SDK for Ruby V3

TimeSeriesIdentifiers

Service: Amazon Forecast Service

Details about the import file that contains the time series for which you want to create forecasts.

Contents

DataSource

The source of your data, an AWS Identity and Access Management (IAM) role that allows Amazon Forecast to access the data and, optionally, an AWS Key Management Service (KMS) key.

Type: DataSource object

Required: No

Format

The format of the data, either CSV or PARQUET.

Type: String

Length Constraints: Maximum length of 7.

Pattern: `^CSV|PARQUET$`

Required: No

Schema

Defines the fields of a dataset.

Type: Schema object

Required: No

See Also

For more information about using this API in one of the language-specific AWS SDKs, see the following:

- AWS SDK for C++

- AWS SDK for Java V2
- AWS SDK for Ruby V3

TimeSeriesReplacementsDataSource

Service: Amazon Forecast Service

A replacement dataset is a modified version of the baseline related time series that contains only the values that you want to include in a what-if forecast. The replacement dataset must contain the forecast dimensions and item identifiers in the baseline related time series as well as at least 1 changed time series. This dataset is merged with the baseline related time series to create a transformed dataset that is used for the what-if forecast.

Contents

S3Config

The path to the file(s) in an Amazon Simple Storage Service (Amazon S3) bucket, and an AWS Identity and Access Management (IAM) role that Amazon Forecast can assume to access the file(s). Optionally, includes an AWS Key Management Service (KMS) key. This object is part of the DataSource object that is submitted in the CreateDatasetImportJob request, and part of the DataDestination object.

Type: S3Config object

Required: Yes

Schema

Defines the fields of a dataset.

Type: Schema object

Required: Yes

Format

The format of the replacement data, which must be CSV.

Type: String

Length Constraints: Maximum length of 7.

Pattern: `^CSV|PARQUET$`

Required: No

TimestampFormat

The timestamp format of the replacement data.

Type: String

Length Constraints: Maximum length of 256.

Pattern: `^[a-zA-Z0-9\-\:\.\,\'\s]+$`

Required: No

See Also

For more information about using this API in one of the language-specific AWS SDKs, see the following:

- AWS SDK for C++
- AWS SDK for Java V2
- AWS SDK for Ruby V3

TimeSeriesSelector

Service: Amazon Forecast Service

Defines the set of time series that are used to create the forecasts in a `TimeSeriesIdentifiers` object.

The `TimeSeriesIdentifiers` object needs the following information:

- `DataSource`
- `Format`
- `Schema`

Contents

TimeSeriesIdentifiers

Details about the import file that contains the time series for which you want to create forecasts.

Type: TimeSeriesIdentifiers object

Required: No

See Also

For more information about using this API in one of the language-specific AWS SDKs, see the following:

- AWS SDK for C++
- AWS SDK for Java V2
- AWS SDK for Ruby V3

TimeSeriesTransformation

Service: Amazon Forecast Service

A transformation function is a pair of operations that select and modify the rows in a related time series. You select the rows that you want with a condition operation and you modify the rows with a transformation operation. All conditions are joined with an AND operation, meaning that all conditions must be true for the transformation to be applied. Transformations are applied in the order that they are listed.

Contents

Action

An array of actions that define a time series and how it is transformed. These transformations create a new time series that is used for the what-if analysis.

Type: Action object

Required: No

TimeSeriesConditions

An array of conditions that define which members of the related time series are transformed.

Type: Array of TimeSeriesCondition objects

Array Members: Minimum number of 0 items. Maximum number of 10 items.

Required: No

See Also

For more information about using this API in one of the language-specific AWS SDKs, see the following:

- AWS SDK for C++
- AWS SDK for Java V2
- AWS SDK for Ruby V3

WeightedQuantileLoss

Service: Amazon Forecast Service

The weighted loss value for a quantile. This object is part of the Metrics object.

Contents

LossValue

The difference between the predicted value and the actual value over the quantile, weighted (normalized) by dividing by the sum over all quantiles.

Type: Double

Required: No

Quantile

The quantile. Quantiles divide a probability distribution into regions of equal probability. For example, if the distribution was divided into 5 regions of equal probability, the quantiles would be 0.2, 0.4, 0.6, and 0.8.

Type: Double

Required: No

See Also

For more information about using this API in one of the language-specific AWS SDKs, see the following:

- AWS SDK for C++
- AWS SDK for Java V2
- AWS SDK for Ruby V3

WhatIfAnalysisSummary

Service: Amazon Forecast Service

Provides a summary of the what-if analysis properties used in the ListWhatIfAnalyses operation. To get the complete set of properties, call the DescribeWhatIfAnalysis operation, and provide the `WhatIfAnalysisArn` that is listed in the summary.

Contents

CreationTime

When the what-if analysis was created.

Type: Timestamp

Required: No

ForecastArn

The Amazon Resource Name (ARN) of the baseline forecast that is being used in this what-if analysis.

Type: String

Length Constraints: Maximum length of 256.

Pattern: `arn:([a-z\d-]+):forecast:.*:.*:.+`

Required: No

LastModificationTime

The last time the resource was modified. The timestamp depends on the status of the job:

- `CREATE_PENDING` - The `CreationTime`.
- `CREATE_IN_PROGRESS` - The current timestamp.
- `CREATE_STOPPING` - The current timestamp.
- `CREATE_STOPPED` - When the job stopped.
- `ACTIVE` or `CREATE_FAILED` - When the job finished or failed.

Type: Timestamp

Required: No

Message

If an error occurred, an informational message about the error.

Type: String

Required: No

Status

The status of the what-if analysis. States include:

- `ACTIVE`
- `CREATE_PENDING`, `CREATE_IN_PROGRESS`, `CREATE_FAILED`
- `CREATE_STOPPING`, `CREATE_STOPPED`
- `DELETE_PENDING`, `DELETE_IN_PROGRESS`, `DELETE_FAILED`

> **Note**
>
> The `Status` of the what-if analysis must be `ACTIVE` before you can access the analysis.

Type: String

Length Constraints: Maximum length of 256.

Required: No

WhatIfAnalysisArn

The Amazon Resource Name (ARN) of the what-if analysis.

Type: String

Length Constraints: Maximum length of 256.

Pattern: `arn:([a-z\d-]+):forecast:.*:.*:.+`

Required: No

WhatIfAnalysisName

The name of the what-if analysis.

Type: String

Length Constraints: Minimum length of 1. Maximum length of 63.

Pattern: `^[a-zA-Z][a-zA-Z0-9_]*`

Required: No

See Also

For more information about using this API in one of the language-specific AWS SDKs, see the following:

- AWS SDK for C++
- AWS SDK for Java V2
- AWS SDK for Ruby V3

WhatIfForecastExportSummary

Service: Amazon Forecast Service

Provides a summary of the what-if forecast export properties used in the ListWhatIfForecastExports operation. To get the complete set of properties, call the DescribeWhatIfForecastExport operation, and provide the `WhatIfForecastExportArn` that is listed in the summary.

Contents

CreationTime

When the what-if forecast export was created.

Type: Timestamp

Required: No

Destination

The path to the Amazon Simple Storage Service (Amazon S3) bucket where the forecast is exported.

Type: DataDestination object

Required: No

LastModificationTime

The last time the resource was modified. The timestamp depends on the status of the job:

- `CREATE_PENDING` - The `CreationTime`.
- `CREATE_IN_PROGRESS` - The current timestamp.
- `CREATE_STOPPING` - The current timestamp.
- `CREATE_STOPPED` - When the job stopped.
- `ACTIVE` or `CREATE_FAILED` - When the job finished or failed.

Type: Timestamp

Required: No

Message

If an error occurred, an informational message about the error.

Type: String

Required: No

Status

The status of the what-if forecast export. States include:

- `ACTIVE`
- `CREATE_PENDING`, `CREATE_IN_PROGRESS`, `CREATE_FAILED`
- `CREATE_STOPPING`, `CREATE_STOPPED`
- `DELETE_PENDING`, `DELETE_IN_PROGRESS`, `DELETE_FAILED`

> **Note**
>
> The `Status` of the what-if analysis must be `ACTIVE` before you can access the analysis.

Type: String

Length Constraints: Maximum length of 256.

Required: No

WhatIfForecastArns

An array of Amazon Resource Names (ARNs) that define the what-if forecasts included in the export.

Type: Array of strings

Array Members: Minimum number of 1 item. Maximum number of 50 items.

Length Constraints: Maximum length of 300.

Pattern: `arn:([a-z\d-]+):forecast:.*:.*:.+`

Required: No

WhatIfForecastExportArn

The Amazon Resource Name (ARN) of the what-if forecast export.

Type: String

Length Constraints: Maximum length of 300.

Pattern: `arn:([a-z\d-]+):forecast:.*:.*:.+`

Required: No

WhatIfForecastExportName

The what-if forecast export name.

Type: String

Length Constraints: Minimum length of 1. Maximum length of 63.

Pattern: `^[a-zA-Z][a-zA-Z0-9_]*`

Required: No

See Also

For more information about using this API in one of the language-specific AWS SDKs, see the following:

- AWS SDK for C++
- AWS SDK for Java V2
- AWS SDK for Ruby V3

WhatIfForecastSummary

Service: Amazon Forecast Service

Provides a summary of the what-if forecast properties used in the ListWhatIfForecasts operation. To get the complete set of properties, call the DescribeWhatIfForecast operation, and provide the `WhatIfForecastArn` that is listed in the summary.

Contents

CreationTime

When the what-if forecast was created.

Type: Timestamp

Required: No

LastModificationTime

The last time the resource was modified. The timestamp depends on the status of the job:

- `CREATE_PENDING` - The `CreationTime`.
- `CREATE_IN_PROGRESS` - The current timestamp.
- `CREATE_STOPPING` - The current timestamp.
- `CREATE_STOPPED` - When the job stopped.
- `ACTIVE` or `CREATE_FAILED` - When the job finished or failed.

Type: Timestamp

Required: No

Message

If an error occurred, an informational message about the error.

Type: String

Required: No

Status

The status of the what-if forecast. States include:

- `ACTIVE`

- CREATE_PENDING, CREATE_IN_PROGRESS, CREATE_FAILED
- CREATE_STOPPING, CREATE_STOPPED
- DELETE_PENDING, DELETE_IN_PROGRESS, DELETE_FAILED

> **Note**
>
> The Status of the what-if analysis must be ACTIVE before you can access the analysis.

Type: String

Length Constraints: Maximum length of 256.

Required: No

WhatIfAnalysisArn

The Amazon Resource Name (ARN) of the what-if analysis that contains this what-if forecast.

Type: String

Length Constraints: Maximum length of 256.

Pattern: arn:([a-z\d-]+):forecast:.*:.*:.+

Required: No

WhatIfForecastArn

The Amazon Resource Name (ARN) of the what-if forecast.

Type: String

Length Constraints: Maximum length of 300.

Pattern: arn:([a-z\d-]+):forecast:.*:.*:.+

Required: No

WhatIfForecastName

The name of the what-if forecast.

Type: String

Length Constraints: Minimum length of 1. Maximum length of 63.

Pattern: `^[a-zA-Z][a-zA-Z0-9_]*`

Required: No

See Also

For more information about using this API in one of the language-specific AWS SDKs, see the following:

- AWS SDK for C++
- AWS SDK for Java V2
- AWS SDK for Ruby V3

WindowSummary

Service: Amazon Forecast Service

The metrics for a time range within the evaluation portion of a dataset. This object is part of the EvaluationResult object.

The `TestWindowStart` and `TestWindowEnd` parameters are determined by the `BackTestWindowOffset` parameter of the EvaluationParameters object.

Contents

EvaluationType

The type of evaluation.

- `SUMMARY` - The average metrics across all windows.
- `COMPUTED` - The metrics for the specified window.

Type: String

Valid Values: `SUMMARY | COMPUTED`

Required: No

ItemCount

The number of data points within the window.

Type: Integer

Required: No

Metrics

Provides metrics used to evaluate the performance of a predictor.

Type: Metrics object

Required: No

TestWindowEnd

The timestamp that defines the end of the window.

Type: Timestamp

Required: No

TestWindowStart

The timestamp that defines the start of the window.

Type: Timestamp

Required: No

See Also

For more information about using this API in one of the language-specific AWS SDKs, see the following:

- AWS SDK for C++
- AWS SDK for Java V2
- AWS SDK for Ruby V3

Amazon Forecast Query Service

The following data types are supported by Amazon Forecast Query Service:

- DataPoint
- Forecast

DataPoint

Service: Amazon Forecast Query Service

The forecast value for a specific date. Part of the Forecast object.

Contents

Timestamp

The timestamp of the specific forecast.

Type: String

Required: No

Value

The forecast value.

Type: Double

Required: No

See Also

For more information about using this API in one of the language-specific AWS SDKs, see the following:

- AWS SDK for C++
- AWS SDK for Java V2
- AWS SDK for Ruby V3

Forecast

Service: Amazon Forecast Query Service

Provides information about a forecast. Returned as part of the QueryForecast response.

Contents

Predictions

The forecast.

The *string* of the string-to-array map is one of the following values:

- p10
- p50
- p90

The default setting is `["0.1", "0.5", "0.9"]`. Use the optional `ForecastTypes` parameter of the CreateForecast operation to change the values. The values will vary depending on how this is set, with a minimum of `1` and a maximum of `5`.

Type: String to array of DataPoint objects map

Key Length Constraints: Maximum length of 4.

Required: No

See Also

For more information about using this API in one of the language-specific AWS SDKs, see the following:

- AWS SDK for C++
- AWS SDK for Java V2
- AWS SDK for Ruby V3

Common Errors

This section lists the errors common to the API actions of all AWS services. For errors specific to an API action for this service, see the topic for that API action.

AccessDeniedException

You do not have sufficient access to perform this action.

HTTP Status Code: 400

IncompleteSignature

The request signature does not conform to AWS standards.

HTTP Status Code: 400

InternalFailure

The request processing has failed because of an unknown error, exception or failure.

HTTP Status Code: 500

InvalidAction

The action or operation requested is invalid. Verify that the action is typed correctly.

HTTP Status Code: 400

InvalidClientTokenId

The X.509 certificate or AWS access key ID provided does not exist in our records.

HTTP Status Code: 403

NotAuthorized

You do not have permission to perform this action.

HTTP Status Code: 400

OptInRequired

The AWS access key ID needs a subscription for the service.

HTTP Status Code: 403

RequestExpired

The request reached the service more than 15 minutes after the date stamp on the request or more than 15 minutes after the request expiration date (such as for pre-signed URLs), or the date stamp on the request is more than 15 minutes in the future.

HTTP Status Code: 400

ServiceUnavailable

The request has failed due to a temporary failure of the server.

HTTP Status Code: 503

ThrottlingException

The request was denied due to request throttling.

HTTP Status Code: 400

ValidationError

The input fails to satisfy the constraints specified by an AWS service.

HTTP Status Code: 400

Common Parameters

The following list contains the parameters that all actions use for signing Signature Version 4 requests with a query string. Any action-specific parameters are listed in the topic for that action. For more information about Signature Version 4, see Signing AWS API requests in the *IAM User Guide*.

Action

The action to be performed.

Type: string

Required: Yes

Version

The API version that the request is written for, expressed in the format YYYY-MM-DD.

Type: string

Required: Yes

X-Amz-Algorithm

The hash algorithm that you used to create the request signature.

Condition: Specify this parameter when you include authentication information in a query string instead of in the HTTP authorization header.

Type: string

Valid Values: `AWS4-HMAC-SHA256`

Required: Conditional

X-Amz-Credential

The credential scope value, which is a string that includes your access key, the date, the region you are targeting, the service you are requesting, and a termination string ("aws4_request"). The value is expressed in the following format: *access_key*/*YYYYMMDD*/*region*/*service*/aws4_request.

For more information, see Create a signed AWS API request in the *IAM User Guide*.

Condition: Specify this parameter when you include authentication information in a query string instead of in the HTTP authorization header.

Type: string

Required: Conditional

X-Amz-Date

The date that is used to create the signature. The format must be ISO 8601 basic format (YYYYMMDD'T'HHMMSS'Z'). For example, the following date time is a valid X-Amz-Date value: `20120325T120000Z`.

Condition: X-Amz-Date is optional for all requests; it can be used to override the date used for signing requests. If the Date header is specified in the ISO 8601 basic format, X-Amz-Date is not required. When X-Amz-Date is used, it always overrides the value of the Date header. For more information, see Elements of an AWS API request signature in the *IAM User Guide*.

Type: string

Required: Conditional

X-Amz-Security-Token

The temporary security token that was obtained through a call to AWS Security Token Service (AWS STS). For a list of services that support temporary security credentials from AWS STS, see AWS services that work with IAM in the *IAM User Guide*.

Condition: If you're using temporary security credentials from AWS STS, you must include the security token.

Type: string

Required: Conditional

X-Amz-Signature

Specifies the hex-encoded signature that was calculated from the string to sign and the derived signing key.

Condition: Specify this parameter when you include authentication information in a query string instead of in the HTTP authorization header.

Type: string

Required: Conditional

X-Amz-SignedHeaders

Specifies all the HTTP headers that were included as part of the canonical request. For more information about specifying signed headers, see Create a signed AWS API request in the *IAM User Guide*.

Condition: Specify this parameter when you include authentication information in a query string instead of in the HTTP authorization header.

Type: string

Required: Conditional

Document History for Amazon Forecast

The following table describes important changes to the *Amazon Forecast Developer Guide*. For notifications about documentation updates, you can subscribe to the RSS feed.

- **Latest documentation update:** March 03, 2021

Change	Description	Date
New Feature	You can now specify a custom forecast frequency. For more information, see Creating a Predictor.	August 29, 2022
New Feature	You can now create a What-if Analysis to explore different scenarios. For more informati on, see What-if analysis.	August 22, 2022
New Feature	You can now have Amazon EventBridge or Amazon CloudWatch Events notify you with status updates for ongoing Amazon Forecast resource jobs. For more information see Setting Up Notifications.	March 15, 2021
New Feature	Amazon Forecast now supports manually stopping the following resources: dataset import jobs, predictor s, predictor backtest export jobs, forecasts, and forecast export jobs.	March 3, 2021

New Feature	Amazon Forecast now supports a built-in featurization that automatically incorporates historical and projected weather information into a model. For more information, see Amazon Forecast Weather Index.	December 8, 2020
New Feature	Amazon Forecast now supports the ability to export backtest forecasts and accuracy metrics for predictors. For more information, see Evaluating Predictor Accuracy.	November 23, 2020
New Feature	Amazon Forecast now supports the ability to specify predictor quantiles. For more information, see Evaluating Predictor Accuracy.	November 11, 2020
New Feature	Amazon Forecast now supports the CNN-QR algorithm. For more information, see CNN-QR.	August 10, 2020
New Feature	Amazon Forecast now supports tagging for the following resources: dataset groups, datasets, dataset import jobs, predictors, forecasts, and forecast export jobs. For more information, see Tagging Amazon Forecast Resources.	July 9, 2020

New Feature	Amazon Forecast now supports missing value filling for related time series datasets. For more informati on, see Handling Missing Values.	May 14, 2020
New Regions	Amazon Forecast adds support for the Asia Pacific (Seoul), Asia Pacific (Mumbai), and Europe (Frankfurt) Regions. For a complete list of the AWS Regions supported by Amazon Forecast, see the AWS Region Table or AWS Regions and Endpoints in the *AWS General Reference*.	March 17, 2020
New Region	Amazon Forecast adds support for the Asia Pacific (Seoul) Region. For a complete list of the AWS Regions supported by Amazon Forecast, see the AWS Region Table or AWS Regions and Endpoints in the *AWS General Reference*.	January 27, 2020
New Feature	Forecast now supports the ability to specify forecast quantiles. For more informati on, see CreateForecast in the Forecast API Guide.	November 22, 2019
Amazon Forecast general availability	Amazon Forecast is now available for general use.	August 21, 2019

Amazon Forecast preview release	This is the first preview release of the documentation for Amazon Forecast.	November 28, 2018

AWS Glossary

For the latest AWS terminology, see the AWS glossary in the *AWS Glossary Reference*.