Maintaining Personalized Experiences with Machine Learning

Implementation guide

Maintaining Personalized Experiences with Machine Learning: Implementation guide

Table of Contents

Deploy a mechanism to keep personalized recommendations up-to-date

Publication date: September 2021 (last update (p. 28): November 2021)

The Maintaining Personalized Experiences with Machine Learning solution streamlines and accelerates development by providing automated pipeline construction; automated personalization model configuration, training, retraining, and deployment; improved visibility into model performance; and advanced error handling mechanisms.

This solution helps you build personalized experiences with Amazon Personalize for your product portfolio and provide real-time, curated experiences across digital channels. Implementing this solution aims to increase your user engagement metrics, clickthroughs, and conversion rates by providing up-to-date product recommendations, personalized product rerankings, and customized direct marketing.

Personalization opportunity exists in multiple areas along the customer journey including:

- **Discoverability** - Helping consumers easily and quickly discover products and content
- **Acquisition and retention** - Attracting and retaining consumers in a digital environment
- **Engagement** - Understanding, measuring, and increasing time spent engaging with products and content
- **Efficiencies and revenue** - Increasing average revenue per user

This AWS solution accelerates the development and deployment of personalization workloads by leveraging the functionalities of the Amazon Personalize service and streamlining productionization and maintenance of Amazon Personalize solutions. It provides end-to-end automation for Amazon Personalize through the entire lifecycle of a workload, which includes:

- Automating the creation of Amazon Personalize solutions and solution versions to baseline model performance (by comparing user-configured model offline metrics over time).
- Presenting the results in an Amazon CloudWatch dashboard.
- Automating the scheduled retraining and update of Amazon Personalize solution versions to assess their performance over time, as typical personalization workloads benefit from full model retraining every one to five days.

This implementation guide describes architectural considerations and configuration steps for deploying Maintaining Personalized Experiences with Machine Learning in the Amazon Web Services (AWS) Cloud. It includes links to an AWS CloudFormation template that launches and configures the AWS services required to deploy this AWS solution using AWS best practices for security and availability.

The guide is intended for IT architects, developers, DevOps, and data analysts who have practical experience architecting in the AWS Cloud.

Cost

You are responsible for the cost of the AWS services used while running this AWS solution.

As of November 2021, the cost for running this AWS solution with the default options for powering content discovery and recommendation through real-time profiling of user preferences and consumption behavior against a 200GB dataset, training daily, with each training taking 20 minutes to complete and consuming 10 training hours per training, while offering real-time inference (**at 10TPS for 24 hours per day**) and exporting a single batch inference for one million users at the end of the month in the US East (N. Virginia) is **$1602.25**.

This solution's cost is highly dependent on Amazon Personalize transactions per second (TPS). For more information, refer to Amazon Personalize Pricing.

This cost estimate does not account for Amazon S3 PUT and GET requests, which can vary depending on how frequently data is accessed in S3.

AWS service	Dimensions	Cost/ month
Amazon Personalize	300 training hours	$72.00
Amazon Personalize (Data Storage, 200 GB)	Data Storage, 200 GB	$10.00
Amazon Personalize	Real-Time Inference, 720 TPS-hours	$1440.00
Amazon Personalize	Batch inference, 1,000,000 recommendations	$67.00
Amazon S3	200 GB	$4.60
AWS Step Functions	100 state transitions for 100 workflow requests	$0.25
Amazon DynamoDB	2 items, 10KB storage, with PITR	$0.30
Amazon CloudWatch	21 metrics, 1GB data ingested, and 1 dashboard with 13 metrics	Metrics: $6.30 Log ingestion: $0.50 Dashboard: $1.30
		Total monthly cost: $1602.25

Prices are subject to change. For full details, refer to the pricing webpage for each AWS service used in this AWS solution.

Architecture overview

Deploying this AWS solution with the default parameters builds the following environment in the AWS Cloud.

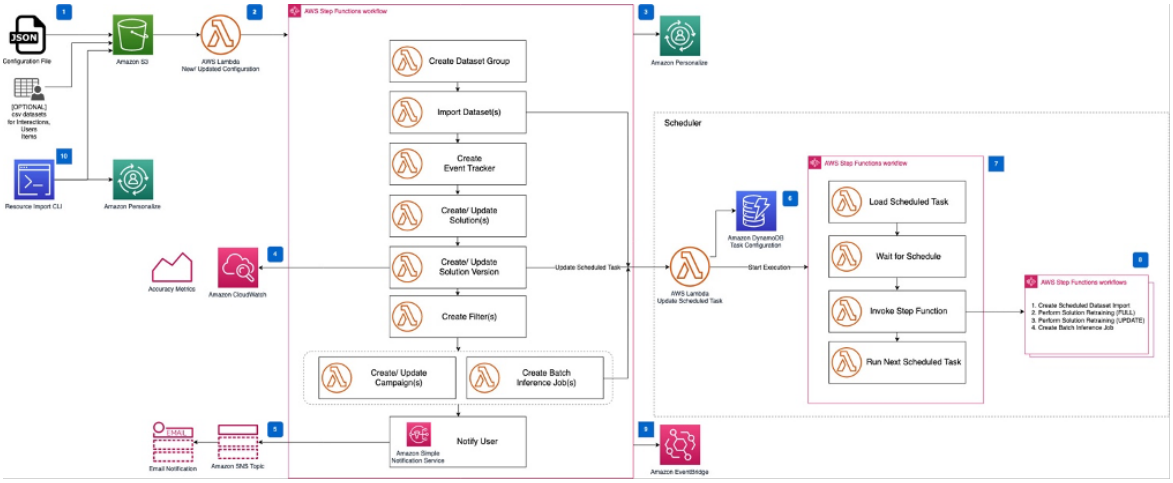

Figure 1 - Maintaining Personalized Experiences with Machine Learning architecture

The AWS CloudFormation template deploys the following infrastructure:

1. An Amazon S3 bucket used to store personalization data and configuration files.
2. An AWS Lambda function initiated when new or updated personalization configuration is uploaded to the personalization data bucket.
3. An AWS Step Functions workflow to manage all of the resources of an Amazon Personalize dataset group (including datasets, schemas, event tracker, filters, solutions, campaigns, and batch inference jobs).
4. Amazon CloudWatch metrics for Amazon Personalize for each new trained solution version are added to help you evaluate the performance of a model over time.
5. An Amazon Simple Notification Service (SNS) topic and subscription to notify an administrator when the maintenance workflow has completed via email.
6. Amazon DynamoDB tracks the scheduled events configured for Amazon Personalize to fully or partially retrain Amazon Personalize solutions, import or reimport datasets, and perform batch inference jobs.
7. An AWS Step Functions workflow tracks the current running scheduled events, and invoke step functions to perform Amazon Personalize solution maintenance (creating new solution versions, updating campaigns), import updated datasets, and perform batch inference.
8. A set of maintenance AWS Step Functions workflows are provided to:
   - Create new dataset import jobs on schedule.
   - Perform Amazon Personalize solution FULL retraining on schedule (and update associated campaigns).
   - Perform Amazon Personalize solution UPDATE retraining on schedule (and update associated campaigns).
   - Create batch inference jobs.
9. An Amazon EventBridge event bus, where resource status notification updates are posted throughout the AWS Step Functions workflow.

10A command line interface (CLI) allows you to import and establish schedules for resources that already exist in Amazon Personalize.

Note
AWS CloudFormation resources are created from AWS Cloud Development Kit (CDK) constructs.

AWS solution components

Personalization AWS Step Function workflows

There are three AWS Step Functions personalization workflows included with the AWS solution:

- **<stack_name>-personalize-workflow**

This is the main workflow, invoked when a valid new configuration file is uploaded to the Amazon S3 bucket for personalization data. This workflow creates all declared resources in the configuration file and any schedules associated with the maintenance of those resources.

- **<stack_name>-periodic-solution-maintenance**

This is a workflow that can be initiated on a schedule to perform Amazon Personalize solution maintenance. This maintenance can include (a) Amazon Personalize solution version creation and campaign update and/or (b) batch inference job creation. Amazon Personalize solution versions can be created as new versions (a FULL model retraining) or as updates (UPDATE model).

> **Note**
> The UPDATE retraining option in Amazon Personalize can only be used when there is an active solution version created from an input solution using the FULL option and the input solution was trained with the `User-Personalization` or `HRNN-Coldstart` recipe.

- **<stack_name>-periodic-dataset-import**

This is a nested workflow that can be initiated to import data on a schedule. Establishing schedules for dataset import is useful when creating batch inferences against data not being collected by an event tracker.

> **Warning**
> The dataset import job replaces any existing data in the dataset that was previously imported in bulk.

Scheduler

Maintaining Personalized Recommendations with Machine Learning deploys an AWS Step Function that acts as a scheduler in order to schedule periodic maintenance workflows. Each scheduled item consists of a name, cron-style schedule, step function to invoke, and step function input, and is managed by the main workflow. Each scheduled task configuration is versioned and stored in Amazon DynamoDB. For more information about cron-style schedules, refer to Schedule Expressions for Rules in the *Amazon CloudWatch Events User Guide.*

This solution provides scheduling support for:

- Dataset import (p. 13)
- Amazon Personalize solution maintenance (p. 15)
- Batch inference job creation (p. 17)

This solution provides a command line interface (CLI) that can be used to establish schedules for dataset import and solution maintenance for resources in Amazon Personalize that were not created by the solution.

Security

When you build systems on AWS infrastructure, security responsibilities are shared between you and AWS. This shared model reduces your operational burden because AWS operates, manages, and controls the components including the host operating system, the virtualization layer, and the physical security of the facilities in which the services operate. For more information about AWS security, visit AWS Cloud Security. For more information about Security in Amazon Personalize, refer to the Amazon Personalize Developer Guide.

IAM roles

AWS Identity and Access Management (IAM) roles allow customers to assign granular access policies and permissions to services and users on the AWS Cloud. This AWS solution creates IAM roles that grant the solution's AWS Lambda functions access to create Regional resources.

Regional deployments

This AWS solution uses the Amazon Personalize service, which is not currently available in all AWS Regions. You must launch this solution in an AWS Region where Amazon Personalize is available. For the most current availability by Region, refer to the AWS Regionalal Services List.

Implementation considerations

Resource naming

In order to produce human-readable names in Amazon Personalize and in the AWS solution's Amazon S3 bucket, Maintaining Personalized Recommendations with Machine Learning follows a naming convention for batch inference jobs.

Resource	Naming Convention	Maximum Resource Name Length
Batch Inference Job	`batch_<solution_name>_%Y_%m_%d_%H_%M_%S`	`solution_name` must not exceed 37 characters in length

If these limits are exceeded, the AWS solution will notify the subscribed email address of the error in configuration.

AWS Lambda quotas

When managing large numbers of scheduled tasks using this AWS solution, it is possible that the account quota for concurrent tasks might be met. When scaling to hundreds of schedules, consider increasing the quota for concurrent tasks in AWS Lambda. For more information refer to Lambda quotas in the *AWS Lambda Developer Guide*.

AWS Step Functions quotas

When managing large numbers of scheduled tasks using this AWS solution, it possible that the account quotas for state throttling might be met. When scaling to hundreds of schedules, consider increasing the quota for bucket size and refill rate per second. For more information on these quotas, refer to Quotas in the *AWS Step Functions Developer Guide*.

AWS CloudFormation template

To automate deployment, this AWS solution uses the following AWS CloudFormation template, which you can download before deployment:

maintaining-personalized-experiences-with-machine-learning.template: Use this template to launch the AWS solution and all associated components. The default configuration deploys Amazon S3, AWS Lambda, AWS Step Functions, Amazon DynamoDB and Amazon SNS, but you can customize the template to meet your specific needs.

Note
AWS CloudFormation resources are created from AWS Cloud Development Kit (CDK) constructs.

Automated deployment

Before you launch the AWS solution, review the cost, architecture, network security, and other considerations discussed in this guide. Follow the step-by-step instructions in this section to configure and deploy the AWS solution into your account.

Time to deploy: Approximately five minutes

Deployment overview

Use the following steps to deploy Maintaining Personalized Recommendations with Machine Learning on AWS. For detailed instructions, follow the links for each step.

Step 1. Launch the stack (p. 10)

- Launch the AWS CloudFormation template into your AWS account.
- Review the templates parameters and enter or adjust the default values as needed.

Step 2. Use the solution to create and/or maintain resources in Amazon Personalize (p. 12)

- Create a configuration .json file for the Amazon Personalize resources to maintain and upload it to the Amazon S3 bucket deployed by the AWS solution.

Step 1. Launch the stack

> **Important**
> This AWS solution includes an option to send anonymous operational metrics to AWS. We use this data to better understand how customers use this AWS solution and related services and products. AWS owns the data gathered though this survey. Data collection is subject to the AWS Privacy Policy.

To opt out of this feature, download the template, modify the AWS CloudFormation mapping section, and then use the AWS CloudFormation console to upload your template and deploy the AWS solution. For more information, refer to the Collection of operational metrics (p. 26) section of this guide.

This automated AWS CloudFormation template deploys the Maintaining Personalized Experiences with Machine Learning solution in the AWS Cloud.

> **Note**
> You are responsible for the cost of the AWS services used while running this AWS solution. For more details, visit the Cost (p. 2) section in this guide, and refer to the pricing webpage for each AWS service used in this solution.

1. Sign in to the AWS Management Console and select the button to launch the `maintaining-personalized-experiences-with-machine-learning.template` AWS CloudFormation template.

Alternatively, you can download the template as a starting point for your own implementation.

2. The template launches in the US East (N. Virginia) Region by default. To launch the AWS solution in a different AWS Region, use the Region selector in the console navigation bar.

   Note
   This AWS solution uses the Amazon Personalize service, which is not currently available in all AWS Regions. You must launch this solution in an AWS Region where Amazon Personalize is available. For the most current availability by Region, refer to the AWS Regional Services List.

3. On the **Create stack** page, verify that the correct template URL is in the **Amazon S3 URL** text box and choose **Next**.
4. On the **Specify stack details** page, assign a name to your AWS solution stack. For information about naming character limitations, refer to IAM and STS Limits in the *AWS Identity and Access Management User Guide*.
5. Under **Parameters**, review the parameters for this AWS solution template and modify them as necessary. This AWS solution uses the following default values.

Parameter	Default	Description
Email	*<Optional input>*	The email that receives status notifications from the AWS Step Functions state machines deployed by this solution. If this parameter is blank, you are not notified with your personalization maintenance job results. You must accept the SNS subscription (through the email link that is sent after stack deployment) to activate notifications.
Personalize KMS Key ARN	*<Optional input>*	While Amazon Personalize will encrypt your data by default, you can monitor and restrict access to your data by specifying a KMS key in your account that can be used by the Amazon Personalize service. Specifying an AWS KMS key ARN in this parameter allows Amazon Personalize to use that key to protect your data. Revoking, turning off, or modifying the key policy associated with this key may render your data unusable by the Amazon Personalize service. To use your own key, specify the full key ARN, for example: `arn:aws:kms:<region>:<account_id>:ke`

Parameter	Default	Description
		`f8fed2cd-14ab-4ac4-a8a3-57975cbff81b`. The key policy must grant the personalize service Encrypt, Decrypt, GenerateDataKey, and DescribeKey permissions on the key, as well as turn on IAM user permissions for the key. An example key policy can be found in KMS customer-managed key policy (p. 22). Leave this parameter blank to have Amazon Personalize use its default encryption configuration for your data.

6. Choose **Next**.
7. On the **Configure stack options** page, choose **Next**.
8. On the **Review** page, review and confirm the settings. Check the box acknowledging that the template will create AWS Identity and Access Management (IAM) resources.
9. Choose **Create stack** to deploy the stack.

   You can view the status of the stack in the AWS CloudFormation Console in the **Status** column. You should receive a CREATE_COMPLETE status in approximately five minutes.

   Note
   In addition to the primary AWS Lambda functions used to create and maintain Amazon Personalize resources, this AWS solution includes additional Lambda functions, that run only during initial configuration or when resources are updated or deleted.
   When you run this AWS solution, you will notice Lambda functions with the stack name you used as a prefix in the AWS console. You must not delete any of these functions, as they are necessary to manage associated resources.

Step 2. Use the AWS solution to create and maintain resources in Amazon Personalize

Use the procedures in Solution configuration (p. 13) to create and maintan resources in Amazon Personalize. The solution configures resources in Amazon Personalize based on configuration .json files uploaded to the S3 bucket deployed by the solution. Resources are configured and maintained based on the contents of this file.

Solution configuration

To determine the name of the S3 bucket deployed by the AWS solution, note the `PersonalizeBucketName` value under the outputs tab of the CloudFormation stack deployed in Automated deployment - Step 1. Launch the stack (p. 10). You can use the AWS Management Console to copy configuration `.json` files to your `PersonalizeBucket` bucket.

By default, the AWS solution detects configuration files uploaded to subfolders under the **train** folder in the `PersonalizeBucket` deployed by the solution. Configuration files must be valid JSON and their file names must end in `.json`.

Note
When managing multiple dataset groups, you must use different subfolders for each dataset group's configuration. For example, for maintaining two dataset groups named **customer_1** and **customer_2**, you can create configuration files as follows:

- `s3://<personalize_bucket_name>/train/customer_1/customer_1_config.json`
- `s3://<personalize_bucket_name>/train/customer_2/customer_2_config.json`

This allows you to keep the input datasets for each dataset group separate.

Creating dataset groups

The top-level container for datasets in Amazon Personalize is the dataset group. To create a dataset group, upload the following configuration:

```
{
    "datasetGroup": {
        "serviceConfig": {
            "name": "customer_1"
        }
    }
}
```

Note
Do not provide a `kmsKeyArn` or `roleArn` to the `datasetGroup.serviceConfig` path – if you provided a Personalize Key ARN in Step 1, the solution will automatically use the key provided.

Creating and maintaining datasets

Amazon Personalize solutions cannot be configured without data being present. To supply data to the Amazon Personalize service, you can upload data directly (through a dataset import job) and/or through an Amazon Personalize event tracker.

This following configuration file is a sample dataset configuration – for more information on how to declare dataset schemas for Amazon Personalize, refer to Datasets and schemas.

To configure the dataset schema(s) required, you can update the configuration file:

```
{
  "datasetGroup": {
    "serviceConfig": {
      "name": "customer_1_datasetgroup"
    },
    "workflowConfig": {
      "schedules": {
        "import": "cron(0 */6 * * ? *)"
      }
    }
  },
  "datasets": {
    "interactions": {
      "dataset": {
        "serviceConfig": {
          "name": "customer_1_interactions"
        }
      },
      "schema": {
        "serviceConfig": {
          "name": "customer_1_interactions_schema",
          "schema": {
            "type": "record",
            "name": "interactions",
            "namespace": "com.amazonaws.personalize.schema",
            "fields": [
              {
                "name": "ITEM_ID",
                "type": "string"
              },
              {
                "name": "USER_ID",
                "type": "string"
              },
              {
                "name": "TIMESTAMP",
                "type": "long"
              },
              {
                "name": "EVENT_TYPE",
                "type": "string"
              },
              {
                "name": "EVENT_VALUE",
                "type": "float"
              }
            ]
          }
        }
      }
    }
  },
  "eventTracker": {
    "serviceConfig": {
      "name": "customer_1_event_tracker"
    }
  }
}
```

Maintaining Personalized Recommendations with Machine Learning inspects the paths `datasets.users.schema.serviceConfig`, `datasets.items.schema.serviceConfig` and `datasets.interactions.schema.serviceConfig` for dataset schema configuration. Schemas can be shared across dataset groups, but must be fully declared in the configuration file.

Note
Do not provide a `datasetGroupArn` or `schemaArn` to any `datasets.*.serviceConfig` path as they will be inferred by the solution.

The AWS solution imports data provided at specific paths relative to the uploaded configuration file. For example, if the configuration file is uploaded to `s3://<personalize_bucket_name>/train/customer_1/customer_1_config.json`, the AWS solution attempts to import data from the following paths (and will complete the import after data is found at one of those paths).

Data Type	Path order to check

Interactions	`s3://<personalize_bucket_name>/train/customer_1/interactions/*.csv`
	
	`s3://<personalize_bucket_name>/train/customer_1/interactions.csv`

Users	`s3://<personalize_bucket_name>/train/customer_1/users/*.csv`
	
	`s3://<personalize_bucket_name>/train/customer_1/users.csv`

Items	`s3://<personalize_bucket_name>/train/customer_1/items/*.csv`
	
	`s3://<personalize_bucket_name>/train/customer_1/items.csv`

Note
If no import data is provided, the AWS solution continues to create resources. This allows you to create an event tracker to add items dynamically to your dataset group. Without data, the AWS solution fails to create an Amazon Personalize solution and solution version.
To create an Amazon Personalize solution and solution version once sufficient data has been added to the dataset group, re-run the step function with the same input, or upload the configuration file again.

In batch inference scenarios, it is useful to re-run a dataset import job (for new data that was uploaded to the Personalize bucket, for example) on a schedule. To define a dataset import job schedule, add a cron schedule `datasetGroup.workflowConfig.schedules.import` path.

Creating and maintaining solutions

An Amazon Personalize solution is the combination of an Amazon Personalize recipe, customized parameters and one or more solution versions (trained models). This AWS solution allows you to declare solution configuration using the same parameters as the Amazon Personalize API, and set schedules for solution version retraining. To configure Maintaining Personalized Recommendations with Machine Learning, you can update the configuration file (the datasets section from the above configuration has been omitted for brevity, but are required):

Note
All configuration parameters for solution creation are supported. For information on additional parameters to supply when configuring an Amazon Personalize solution, use the parameters from the CreateSolution API documentation in the *Amazon Personalize Developer Guide*. The `datasetGroupARN` is inferred by the solution and must not be provided.

```
{
  "datasetGroup": {
    "serviceConfig": {
      "name": "customer_1_datasetgroup"
    },
    "workflowConfig": {
      "schedules": {
        "import": "cron(0 */6 * * ? *)"
      }
    }
  },
  "datasets": {
    "see": "above"
  },
  "solutions": [
    {
      "serviceConfig": {
        "name": "customer_1_user_personalization",
        "recipeArn": "arn:aws:personalize:::recipe/aws-user-personalization"
      },
      "workflowConfig": {
        "schedules": {
          "full": "cron(30 0 * * ? *)",
          "update": "cron(0 * * * ? *)"
        }
      },
      "campaigns": [
        {
          "serviceConfig": {
            "name": "customer_1_user_personalization_cpn",
            "minProvisionedTPS": 1
          }
        }
      ]
    }
  ]
}
```

When using the `aws-user-personalization` recipe, the service automatically updates the solution version in the background every 2 hours (at no additional cost). This auto-update process brings in new items added since the last update so that they can start being recommended to users. If the 2 hour auto-update is not frequent enough for introducing new items, you can have the Maintaining Personalized Experiences with Machine Learning solution create a new solution version on a cron schedule specified at the path `solutions[idx].workflowConfig.schedules.update`. An update retraining does not fully retrain the model. To occasionally create a new solution version (a full retraining to recalculate weights across the model based on all data), specify a cron schedule at the path `solutions[idx].workflowConfig.schedules.full`. As of September 2021, Amazon Personalize recommends training a new model weekly. The schedules specified in the example configuration have FULL training performed daily at 00:30 UTC and update training performed hourly.

Note
When solution versions are updated (via FULL or UPDATE training) the campaigns associated with the solution are updated to use the new solution version. This will result in a training cost to create a new solution version.

Scheduling batch inference jobs

To schedule batch inference jobs for your solutions, you must specify the `batchInferenceJobs` key under the solution with which you want to perform the batch inference.

```
{
  "datasetGroup": {
    "serviceConfig": {
      "name": "customer_1_datasetgroup"
    },
    "workflowConfig": {
      "schedules": {
        "import": "cron(0 */6 * * ? *)"
      }
    }
  },
  "datasets": {"see": "above"},
  "solutions": [
    {
      "serviceConfig": {
        "name": "customer_1_user_personalization",
        "recipeArn": "arn:aws:personalize:::recipe/aws-user-personalization"
      },
      "workflowConfig": {
        "schedules": {
          "full": "cron(30 0 * * ? *)",
          "update": "cron(0 * * * ? *)"
        }
      },
      "campaigns": [
        {
          "serviceConfig": {
            "name": "customer_1_user_personalization_cpn",
            "minProvisionedTPS": 1
          }
        }
      ],
      "batchInferenceJobs": [
        {
          "serviceConfig": {},
          "workflowConfig": {
            "schedule": "cron(0 3 * * ? *)"
          }
        }
      ]
    }
  ]
}
```

Note
The batch inference jobs created use the most recent solution version discovered for their solution. Additional batch inference job parameters can be specified under the `serviceConfig` key (empty above). The `jobInput`, `jobName`, `jobOutput`, `roleArn`, and `solutionVersionArn` are inferred by the Maintaining Personalized Experiences with Machine Learning solution.

The solution expects batch inference job input .json file in an Amazon S3 location at the following path:

```
s3://<personalize_bucket_name>/batch/<dataset_group_name>/<solution_name>/job_config.json
```

The solution outputs batch inference job results in an Amazon S3 location at the following path:

```
s3://<personalize_bucket_name>/batch/<dataset_group_name>/<solution_name>/<job_name>/*
```

This example cron configuration performs a batch inference job nightly at 03:00 UTC.

Receiving job status notifications

You can configure Amazon EventBridge or CloudWatch Events to notify you with status updates for ongoing workflows, such as importing data, generating new solution versions, and creating batch inference jobs. EventBridge and CloudWatch Events deliver a near real-time stream of events that describe changes in AWS resources. For example, you can set up an event to notify you when an Amazon Personalize batch inference job finishes and notifies a third party system with the batch inference job results.

Events are emitted on a best-effort basis. For more information about events, refer to the *Amazon EventBridge User Guide* and 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, refer to the Amazon EventBridge documentation.

Monitoring personalize workflows

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 events generated by this solution 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 resources and status change events that you can monitor:

Resource	Status change event name	Status
Dataset Group	Personalize Dataset Group State Change	`ACTIVE, CREATE IN_PROGRESS`
Dataset	Personalize Dataset State Change	`ACTIVE, CREATE IN_PROGRESS`
Dataset Import Job	Personalize Dataset Import Job State Change	`ACTIVE, CREATE IN_PROGRESS`
Solution	Personalize Solution State Change	`ACTIVE, CREATE IN_PROGRESS`
Solution Version	Personalize Solution Version State Change	`ACTIVE, CREATE IN_PROGRESS`
Campaign	Personalize Campaign State Change	`ACTIVE, CREATE IN_PROGRESS, UPDATING`

Resource	Status change event name	Status
Batch Inference Job	Personalize Batch Inference Job State Change	`ACTIVE, CREATE IN_PROGRESS`

Notifications contain information about the resource, including the Amazon Resource Name (ARN), job status, job duration (in seconds), and, if the job failed, an error message. The following code snippet is an example notification:

```
{
    "version": "0",
    "id": "345a0b30-35d5-4c9f-89e4-1fc1200ee77e",
    "detail-type": "Personalize Solution Version State Change",
    "source": "solutions.aws.personalize",
    "account": "111122223333",
    "time": "2021-10-31T01:19:15Z",
    "region": "us-east-1",
    "resources": [
        "arn:aws:personalize:us-east-1:111122223333:solution/user_personalization_solution/
aaaaaaaa"
    ],
    "detail": {
        "Arn": "arn:aws:personalize:us-east-1:111122223333:solution/
user_personalization_solution/aaaaaaaa",
        "Status": "ACTIVE",
        "Duration": 1116
    }
}
```

Creating an EventBridge Rule for job status notifications

To create an EventBridge rule to notify you of status changes for ongoing workflows, refer to Creating Amazon EventBridge rules that react to events in the *Amazon EventBridge* User Guide.

In the sample procedure, for Service provider, select **Custom pattern**, and configure your event pattern as per the *Amazon EventBridge User Guide*.

For example, add the following parameters to start a rule when any batch inference job is completed:

```
{
    "source": ["solutions.aws.personalize"],
    "detail-type": ["Personalize Batch Inference Job State Change"],
    "detail": {
        "Status": ["ACTIVE"]
    }
}
```

Importing existing resources

It is useful to establish schedules for resources that already exist within Amazon Personalize (for example, to schedule FULL solution version retraining). To accomplish this, you can use the included scheduler CLI.

Installing the scheduler CLI

The scheduler CLI is available with the solution source code, under the `source` directory. Use the following commands to install the CLI. It is recommended that you use a separate virtual environment for this installation.

```
cd source
pip install --upgrade pip
pip install cdk_solution_helper_py/helpers_common
pip install scheduler/common
```

Using the scheduler CLI

You can use the included help commands to list command options:

```
> aws-solutions-scheduler --help

Usage: aws-solutions-scheduler [OPTIONS] COMMAND [ARGS]...

  Schedluer CLI

Options:
  -s, --stack TEXT                [required]
  -r, --region TEXT               [required]
  --scheduler-table-name-output TEXT
  --scheduler-stepfunction-arn-output TEXT
  --help                          Show this message and exit.

Commands:
  activate              Activate a scheduled task
  deactivate            Deactivate a scheduled task
  describe              Describe a scheduled task
  import-dataset-group  Create a new configuration from an existing...
  list                  List all scheduled tasks
```

Listing active schedules

```
> aws-solutions-scheduler -s <stack_name> -r <region_name> list
{
  "tasks": [
    "personalize-dataset-import-item-recommender",
    "solution-maintenance-full-item-recommender-user-personalization"
  ]
}
```

Describing a scheduled task

Describing a scheduled task will show you its name, version, status (activated or deactivated), schedule, and target step function:

```
> aws-solutions-scheduler -s <stack_name> -r <region_name> describe -task
{
  "task": {
```

```
    "active": true,
    "name": "personalize-dataset-import-item-recommender",
    "schedule": "cron(*/15 * * * ? *)",
    "step_function": "arn:aws:states:us-east-1: 111122223333:stateMachine:personalizestack-
periodic-dataset-import-aaaaaaaaaaaa",
    "version": "v1"
  }
}
```

Deactivating a task

Deactivating a task will stop any active schedules but retain the scheduled task for future reactivation.

```
> aws-solutions-scheduler -s PersonalizeStack -r us-east-1 deactivate --task solution-
maintenance-full-item-recommender-user-personalization
{
  "task": {
    "active": false,
    "name": "solution-maintenance-full-item-recommender-user-personalization",
    "schedule": "cron(0 */6 * * ? *)",
    "step_function": "arn:aws:states:us-east-1: 111122223333:stateMachine:personalizestack-
periodic-solution-maintenance-aaaaaaaaaaaa",
    "version": "v1"
  }
}
```

Activating a task

Activating a task will ensure that the scheduler step function is running for the task on the specified schedule.

```
> aws-solutions-scheduler -s PersonalizeStack -r us-east-1 activate --task solution-
maintenance-item-recommender-user-personalization
{
  "task": {
    "active": true,
    "name": "solution-maintenance-full-item-recommender-user-personalization",
    "schedule": "cron(0 */6 * * ? *)",
    "step_function": "arn:aws:states:us-east-1: 111122223333:stateMachine:personalizestack-
periodic-solution-maintenance-aaaaaaaaaaaa",
    "version": "v1"
  }
}
```

Importing an existing resource and setting schedules

It can be useful to add scheduling to resources that already exist in Amazon Personalize but that are not managed by the scheduler or the solution. Use the following code to import and set schedules:

```
aws-solutions-scheduler -s <stack_name> -r <region_name> import-dataset-group
 -d <dataset_group_name> -i "cron(*/15 * * * ? *)" -f "item-recommender-user-
personalization@cron(0 */6 * * ? *)" -p train/item-recommender/config.json
```

The CLI supports cron-style expressions for import and solution version UPDATE and FULL training.

Note
While the solution supports batch inference job scheduling, the scheduler CLI cannot establish schedules for batch import. To establish a schedule for batch import, first import the dataset group using the CLI, then follow the instructions in Scheduling batch inference jobs (p. 17).

KMS customer-managed key policy

The following is a valid key policy for use with the solution when using the Personalize KMS Key ARN parameter.

Warning
If KMS is required, ensure the Personalize KMS Key ARN is provided when the stack is created. Do not update the stack to allow KMS. Do not remove the key after it is set. Doing either will result in resources being unable to update due to different security configurations.

```
{
    "Version": "2012-10-17",
    "Id": "PersonalizePolicy",
    "Statement": [
        {
            "Sid": "Allow use for the Personalize service",
            "Effect": "Allow",
            "Principal": {
                "Service": "personalize.amazonaws.com"
            },
            "Action": [
                "kms:Encrypt",
                "kms:Decrypt",
                "kms:GenerateDataKey",
                "kms:DescribeKey"
            ],
            "Resource": "*"
        },
        {
            "Sid": "Enable IAM User Permissions",
            "Effect": "Allow",
            "Principal": {
                "AWS": "arn:aws:iam::<account_id>:root"
            },
            "Action": "kms:*",
            "Resource": "*"
        }
    ]
}
```

Additional resources

AWS services

- Amazon S3
- AWS Lambda
- Amazon DynamoDB
- AWS Step Functions
- Amazon Simple Notification Service
- Amazon CloudWatch
- Amazon Personalize

Uninstall the solution

You can uninstall the Maintaining Personalized Recommendations with Machine Learning solution from the AWS Management Console or by using the AWS Command Line Interface. You must manually delete the Personalize bucket and scheduler AWS DynamoDB table created by this solution. AWS Solutions Implementations do not automatically delete buckets and DynamoDB databases in case you have stored data to retain.

Using the AWS Management Console

1. Sign in to the AWS CloudFormation console.
2. On the **Stacks** page, select this solution's installation stack.
3. Choose **Delete**.

Using AWS Command Line Interface

Determine whether the AWS Command Line Interface (AWS CLI) is available in your environment. For installation instructions, refer to What Is the AWS Command Line Interface in the *AWS CLI User Guide*. After confirming that the AWS CLI is available, run the following command.

```
$ aws cloudformation delete-stack --stack-name <installation-stack-name>
```

Deleting the Amazon S3 bucket

This solution is configured to retain the solution-created Amazon S3 bucket (for deploying in an opt-in Region) if you decide to delete the AWS CloudFormation stack to prevent accidental data loss. After uninstalling the solution, you can manually delete this S3 bucket if you do not need to retain the data. Follow these steps to delete the Amazon S3 bucket.

1. Sign in to the Amazon S3 console.
2. Choose **Buckets** from the left navigation pane.
3. Locate the bucket referred to in the outputs of the stack.
4. Select the S3 bucket and choose **Delete**.

To delete the S3 bucket using AWS CLI, run the following command:

```
$ aws s3 rb s3://<bucket-name> --force
```

Deleting the scheduler Amazon DynamoDB database

The solution is configured to retain the scheduled items configured in the DynamoDB database if you decide to delete the AWS CloudFormation stack to prevent accidental data loss. After uninstalling the solution, you can remove this database if it is not required. Follow these steps to delete it:

1. Sign in to the Amazon DynamoDB console.
2. Choose **Tables** from the left navigation pane.
3. Select the table referencing the stack that was deleted
4. Choose **Delete table**
5. Check **Delete all CloudWatch alarms for this table**
6. Type the word **delete** to confirm deletion of the table
7. Choose **Delete**

Collection of operational metrics

This solution includes an option to send anonymous operational metrics to AWS. We use this data to better understand how customers use this solution and related services and products. When invoked, the following information is collected and sent to AWS:

- **Solution ID:** The AWS solution identifier
- **Unique ID (UUID):** Randomly generated, unique identifier for each Maintaining Personalized Recommendations with Machine Learning deployment
- **Timestamp:** Data-collection timestamp

AWS owns the data gathered though this survey. Data collection is subject to the AWS Privacy Policy. To opt out of this feature, complete the following steps before launching the AWS CloudFormation template.

1. Download the AWS CloudFormation template to your local hard drive.
2. Open the AWS CloudFormation template with a text editor.
3. Modify the AWS CloudFormation template mapping section from:

```
AnonymousData:
    SendAnonymousData:
      Data: Yes
```

to:

```
AnonymousData:
    SendAnonymousData:
      Data: No
```

1. Sign in to the AWS CloudFormation console.
2. Select **Create stack**.
3. On the **Create stack** page, **Specify template** section, select **Upload a template file**.
4. Under **Upload a template file**, choose **Choose file** and select the edited template from your local drive.
5. Choose **Next** and follow the steps in Launch the stack (p. 10) in the Automated Deployment section of this guide.

Source code

Visit our GitHub repository to download the source files for this solution and to share your customizations with others. The Maintaining Personalized Experiences with Machine Learning templates are generated using the AWS Cloud Development Kit (CDK) (AWS CDK). Refer to the README.md file for additional information.

Revisions

Date	Change
September 2021	Initial release
October 2021	Release version 1.0.1 - Amazon SNS message formatting changes, and stack output changes. For more information about the changes, refer to the CHANGELOG.md file in the GitHub repository.
November 2021	Release version 1.1.0 - Adds Amazon EventBridge support and scheduler command line interface (CLI). For more information about the changes, refer to the CHANGELOG.md file in the GitHub repository.

Contributors

- Paul Shuparski-Miller

Notices

Customers are responsible for making their own independent assessment of the information in this document. This document: (a) is for informational purposes only, (b) represents AWS current product offerings and practices, which are subject to change without notice, and (c) does not create any commitments or assurances from AWS and its affiliates, suppliers or licensors. AWS products or services are provided “as is” without warranties, representations, or conditions of any kind, whether express or implied. AWS responsibilities and liabilities to its customers are controlled by AWS agreements, and this document is not part of, nor does it modify, any agreement between AWS and its customers.

AWS glossary

For the latest AWS terminology, see the AWS glossary in the *AWS General Reference*.