# Genomics Tertiary Analysis and Machine Learning Using Amazon SageMaker

## Implementation Guide

# Genomics Tertiary Analysis and Machine Learning Using Amazon SageMaker: Implementation Guide

# Table of Contents

# Build machine learning models on genomic datasets using AWS

Publication date: *August 2020*

This implementation guide discusses architectural considerations and configuration steps for deploying the Genomics Tertiary Analysis and Machine Learning Using Amazon SageMaker solution in the Amazon Web Services (AWS) Cloud. It includes a link to an AWS CloudFormation template that launches and configures the AWS services required to deploy this solution using AWS best practices for security and availability.

The guide is intended for IT infrastructure architects, administrators, bioinformatics scientists, data engineers, and DevOps professionals who have practical experience architecting in the AWS Cloud.

# Overview

The Genomics Tertiary Analysis and Machine Learning Using Amazon SageMaker solution sets up a platform in AWS that allows users to build machine learning models on genomic datasets using AWS managed services. We define *tertiary analysis* to be the interpretation of genomic variants and assigning meaning to them. In this solution, we address the specific challenge of competing clinical definitions when examining genomic variants. Our intention is to provide a broad platform for genomic machine learning in AWS, using variant classification as an example of a scientifically meaningful problem that can be solved using this platform.

This solution sets up a pipeline that consists of a data processing layer for feature engineering, a notebook instance for interactive code development, and a model training and deployment setup for training models and generating inferences respectively. Once deployed, customers can walk through a built-in example that demonstrates the working of this pipeline using an example use case.

Our example is based on the following Kaggle challenge. We create a model to predict if a variant annotated in ClinVar has a conflicting classification or not. ClinVar variants are manually classified by clinical laboratories on a categorical spectrum ranging from benign, likely benign, uncertain significance, likely pathogenic, and pathogenic. Variants that have conflicting classifications (from laboratory to laboratory) can cause confusion when clinicians or researchers try to interpret whether the variant has an impact on the disease of a given patient. Hence, a model that can predict the existence of a conflicting classification for a variant can save valuable time that researchers have to spend looking for such conflicts.

This solution uses AWS Glue to convert a VCF file into parquet and to convert the ClinVar annotation file from TSV to parquet, a SageMaker notebook instance for model code development and an Amazon SageMaker Autopilot job for training and deploying the model. The steps can be altered or replicated by customers for their own use cases and datasets. For information about customizing the solution, refer to the Genomics Tertiary Analysis and Machine Learning Using Amazon SageMaker Developer Guide.

This solution demonstrates how to 1) automate the preparation of a genomics machine learning training dataset, 2) develop genomics machine learning model training and deployment pipelines and, 3) generate predictions and evaluate model performance using test data.

# Automate the preparation of the genomics machine learning training dataset

This solution starts a job in AWS Glue to create the dataset used to train machine learning models. The training dataset is based on the publicly available ClinVar dataset combined with Ensembl Variant Effect Predictor (VEP) annotations. You can learn more about adding or modifying jobs in AWS Glue to prepare machine learning training datasets in the Genomics Tertiary Analysis and Machine Learning Using Amazon SageMaker Developer Guide.

> **Note:**
> ClinVar is a publicly available dataset that aggregates information about genomic variation and its relationship to human health. VEP determines the effect of variants Single Nucleotide Polymorphisms (SNP)s, insertions, deletions, Copy Number Variations (CNVs) on genes, transcripts, protein sequence, and regulatory regions.

This solution is based on the Genetic Variant Classifications Kaggle project, which demonstrates how to predict whether a variant will have conflicting clinical classifications using machine learning. To learn more about how the solution uses the Kaggle project, refer to Appendix A (p. 14).

# Develop genomics machine learning model generation pipelines

This solution uses Amazon SageMaker Autopilot to create a binary classification model that predicts if the given variant has a conflicting variant. Sagemaker Autopilot runs a series of model training experiments in the background using the provided dataset. The experiments and the plan of execution are captured in the `AmazonSageMakerAutopilotCandidateDefinitionNotebook` Jupyter notebook to provide you the details of how the model was created, refine it as desired, and recreate it from the notebook at any point in the future. Sagemaker Autopilot also creates a data exploration notebook that provides details about the dataset. To learn more about adding new genomics model generation pipelines to the solution, refer to the Genomics Tertiary Analysis and Machine Learning Using Amazon SageMaker Developer Guide.

# Generate predictions and evaluate model performance using test data

This solution provides a `AmazonSageMakerAutopilotCandidateDefinitionNotebook` Jupyter notebook to enable you to generate predictions with test data using the model endpoint deployed. You can generate model predictions and evaluate model performance with a confusion matrix and a receiver operating characteristic (ROC) curve, which are example metrics packaged with the solution. You can modify the training dataset, experiments, and prediction techniques at any time.

# Cost

You are responsible for the cost of the AWS services used while running this solution. As of the date of publication, the cost for running this solution *with default settings* in the US East (N. Virginia) Region is approximately **$0.05 a month**, which includes running the Amazon SageMaker Jupyter notebook instance that runs all the solution notebooks. Prices are subject to change. For full details, see the pricing webpage for each AWS service used in this solution.

> **Note:**
> A copy of the solution training dataset that is produced by the `genomicslearning-create-trainingset` AWS Glue job is copied into the solution's data lake bucket. If you run the AWS Glue job to recreate the dataset, there is a charge.
> AWS Glue job execution: 2 DPUs * 1/6 hour at $0.44 per DPU-Hour or $0.15

If you customize this solution to create your own training dataset, the cost factors include the storage size of the data being analyzed, the number of Extract Transform and Load (ETL) jobs, compute resources required for each job, and the number of notebook instances provisioned. For a more accurate estimate of cost, we recommend working with a sample dataset as a benchmark. Prices are subject to change. For full details, refer to the pricing webpage for each AWS service used in this solution

# Architecture

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

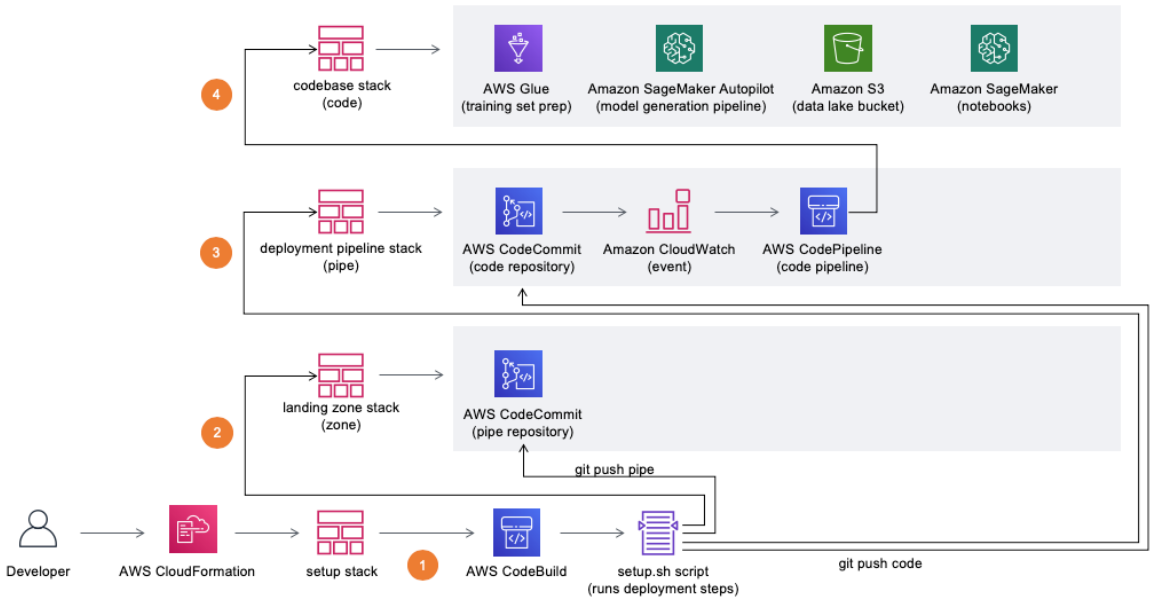


**Figure 1: Genomics Tertiary Analysis and Machine Learning Using Amazon SageMaker deployment architecture**

The AWS CloudFormation template creates four CloudFormation stacks in your AWS account including a `setup` stack to install the solution. The other stacks include a landing zone (`zone`) stack containing the common solution resources and artifacts; a deployment pipeline (`pipe`) stack defining the solution's continuous integration and continuous delivery (CI/CD) pipeline; and a codebase (`code`) stack providing the ETL scripts, jobs, crawlers, a data catalog, and notebook resources.

1. The solution's `setup` stack creates an AWS CodeBuild project containing the `setup.sh` script. This script creates the remaining CloudFormation stacks and provides the source code for both the AWS CodeCommit `pipe` repository and the `code` repository.
2. The landing zone (`zone`) stack creates the CodeCommit `pipe` repository. After the landing zone (`zone`) stack completes its setup, the `setup.sh` script pushes source code to the CodeCommit `pipe` repository.
3. The deployment pipeline (`pipe`) stack creates the CodeCommit `code` repository, an Amazon CloudWatch event, and the CodePipeline `code` pipeline. After the deployment pipeline (`pipe`) stack completes setup, the `setup.sh` script pushes source code to the CodeCommit `code` repository.
4. The CodePipeline (`code`) pipeline deploys the codebase (`code`) CloudFormation stack. After the AWS CodePipeline pipelines complete their setup, the resources deployed in your account include Amazon Simple Storage Service (Amazon S3) buckets for storing object access logs, build artifacts, and data; CodeCommit repositories for source code; an AWS CodeBuild project for building code artifacts (for example, third-party libraries used for data processing); an AWS CodePipeline pipeline for automating builds and deployment of resources; example AWS Glue jobs; and an Amazon SageMaker Jupyter notebook instance.

The example code includes the resources needed to quickly develop machine learning models using genomics data and generate predictions.

The following diagram describes the overall solution architecture, including the types of Amazon SageMaker Jupyter notebooks that are deployed.

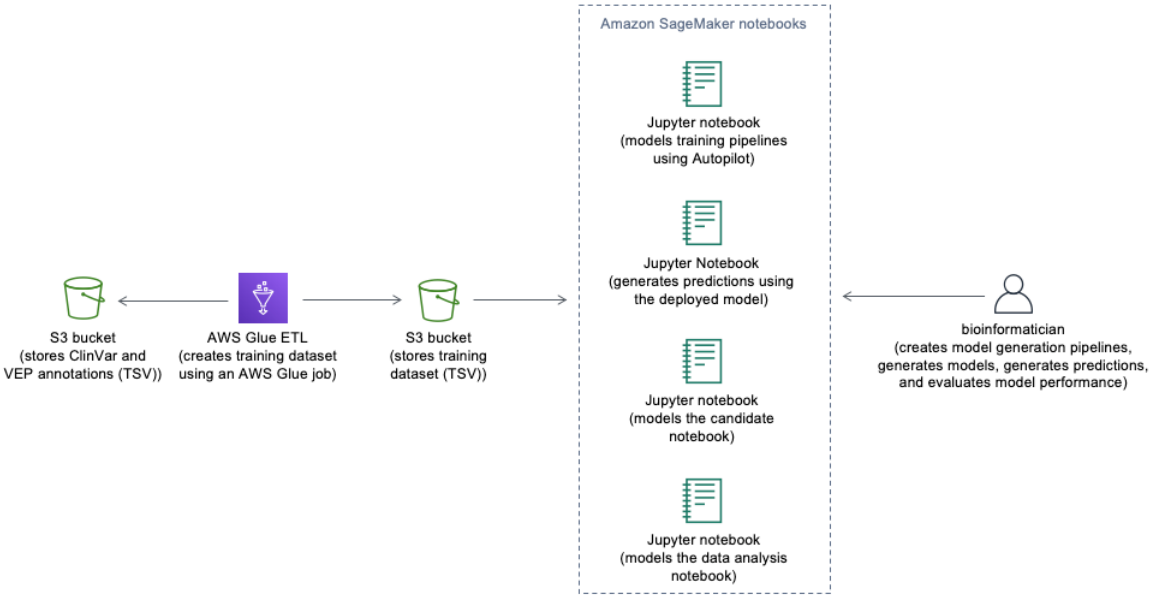


**Figure 2: Genomics Tertiary Analysis and Machine Learning Using Amazon SageMaker architecture**

**Note**

ClinVar is a publicly available dataset that aggregates information about genomic variation and its relationship to human health.

Ensemble Variant Effect Predictor (VEP) determines the effect of your variants (SNPs, insertions, deletions, CNVs or structural variants) on genes, transcripts, and protein sequence, as well as regulatory regions.

The `clinvar.annotation.vcf.gz` dataset was created to determine the effects of gene variants (such as SNPs, insertions, deletions, CNVs, or structural variants) on genes, transcripts, protein sequence, and regulatory regions.

To access the ClinVar dataset with VEP annotations used in this solution, refer to the clinvar-kaggle GitHub repository.

During deployment, a ClinVar tab separated values (TSV) file with VEP annotations is copied into the `DataLakeBucket` S3 bucket. An AWS Glue job creates the machine learning training dataset and copies the resulting file into the `DataLakeBucket` S3 bucket. The solution script `process_clinvar.py` is used to create the training set. For information about the SageMaker Jupyter notebooks that are used in this solution, refer to Appendix D (p. 24).

# Solution components

## CI/CD pipeline

A complete continuous integration and continuous deployment (CI/CD) pipeline is created when you launch the solution. This pipeline is built using AWS CodeCommit as the source code, AWS CodeBuild projects to copy resources to the `ResourcesBucket` Amazon Simple Storage Service (Amazon S3) bucket, and an AWS CodePipeline pipeline to run a build project and automate deployment (using AWS CloudFormation) after the updated source code is published.

The AWS resources that compose the CI/CD pipeline are defined in the `Pipe/template_cfn.yml` file. Changes to the solution requiring new artifacts to be built, for example, bioinformatics libraries like Hail or Adam, require an update to the CI/CD pipeline definition.

## Solution demonstration datasets

This solution copies the ClinVar with VEP annotations public dataset into the `DataLakeBucket` S3 bucket. This dataset aggregates information about genomic variation and its relationship to human health. Variant effect predictors are added to this dataset. The final training dataset is copied into the `DataLakeBucket` S3 bucket, which makes running the AWS Glue job optional.

> **Note**
> ClinVar is a publicly available dataset that aggregates information about genomic variation and its relationship to human health.
> Ensemble Variant Effect Predictor (VEP) determines the effect of your variants (SNPs, insertions, deletions, CNVs or structural variants) on genes, transcripts, and protein sequence, as well as regulatory regions.
> The `clinvar.annotation.vcf.gz` dataset was created to determine the effects of gene variants (such as SNPs, insertions, deletions, CNVs, or structural variants) on genes, transcripts, protein sequence, and regulatory regions.
> To access the ClinVar dataset with VEP annotations used in this solution, refer to the clinvar-kaggle GitHub repository.

Additionally, you can build a genomics data lake using AWS services. To learn more, refer to the Genomics Tertiary Analysis and Data Lakes Using AWS Glue and Amazon Athena solution.

## AWS Glue jobs

This solution creates the `create-training-set` job in AWS Glue, which transforms the ClinVar dataset with variant effect predictors. These predictors are added into ClinVar datasets and include features that are used for model training.

## SageMaker notebook instance

This solution creates an Amazon SageMaker notebook instance that demonstrates how to use AWS Glue and Amazon SageMaker Autopilot to create a machine learning model generation pipeline.

# Amazon S3 buckets

This solution creates the following Amazon S3 buckets. Each bucket has encryption and logging enabled.

- **DataLakeBucket** – Stores ClinVar variant annotation data and the solution training dataset.
- **ResourcesBucket** – Stores notebooks and shell scripts.
- **BuildBucket** – Stores build artifacts deployed through the pipeline.
- **LogsBucket** – Stores solution log files.

# Considerations

The Genomics Tertiary Analysis and Machine Learning Using Amazon SageMaker solution fully leverages *infrastructure as code* principles and best practices that enable you to rapidly evolve the solution. Storing your Extract Transform and Load (ETL) job and notebook deployment definitions as code makes them easier to share, inspect for compliance, and reproduce. Additionally, each change you make is tracked by the continuous integration and continuous delivery (CI/CD) pipeline, facilitating change control management, rollbacks, and auditing.

## Regional deployment

This solution uses the AWS CodePipeline service, which is currently available in specific AWS Regions only. Therefore, you must launch this solution in an AWS Region where this service is available. For the most current service availability by AWS Region, see AWS service offerings by Region.

# AWS CloudFormation template

This solution uses AWS CloudFormation to automate the deployment of the Genomics Tertiary Analysis and Machine Learning Using Amazon SageMaker solution in the AWS Cloud. It includes the following AWS CloudFormation template, which you can download before deployment.

View Template

**genomics-tertiary-analysis-and-machine-learning-using-amazon-sagemaker.template:** Use this template to launch this solution and all associated components. The default configuration deploys an AWS CodePipeline deployment pipeline, an AWS CodeCommit repository for the pipeline code, an AWS CodeCommit repository for the solution code, an AWS Glue job, AWS Identity and Access Management (IAM) roles and policies, and an Amazon SageMaker notebook instance. You can also customize the template based on your specific needs.

# Automated deployment

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

**Time to deploy:** Approximately 30 minutes.

## Prerequisites

Before deploying this solution, verify that you have an administrator role in AWS Identity and Access Management (IAM) in your AWS account. Refer to Appendix B (p. 15) for details on permissions used by this solution. For information about setting up an administrator user, refer to Creating Your First IAM Admin User and Group in the *AWS Identity and Access Management User Guide*.

## Deployment overview

Use the following steps to deploy this solution on AWS. For detailed instructions, follow the link for each step.

Step 1. Launch the stack (p. 10)

- Launch the AWS CloudFormation template into your AWS account.

Step 2. Run the model generation pipeline (p. 11)

Step 3. Generate predictions and evaluate model performance (p. 11)

## Step 1. Launch the stack

This automated AWS CloudFormation template deploys the solution in the AWS Cloud. You must have the appropriate IAM permissions before launching the stack.

**Note:**
You are responsible for the cost of the AWS services used while running this solution. Refer to the Cost section for more details. For full details, see the pricing webpage for each AWS service used in this solution.

1. Sign in to the AWS Management Console and click the button below to launch the `genomics-tertiary-analysis-and-machine-learning-using-amazon-sagemaker.template` AWS CloudFormation template.


You can also 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 this solution in a different AWS Region, use the Region selector in the console navigation bar.

   **Note:**
   This solution uses the AWS CodePipeline service, which is currently available in specific AWS Regions only. Therefore, you must launch this solution in an AWS Region where this service is available. For the most current service availability by AWS Region, refer to AWS service offerings by Region.
3. On the **Create stack** page, verify that the correct template URL shows in the **Amazon S3 URL** text box and choose **Next**.
4. On the **Specify stack details** page, assign a name to your solution stack and provide a **Project** name for the solution installation.
5. Choose **Next.**
6. On the **Configure stack options** page, choose **Next**.
7. 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.
8. 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 see a status of CREATE_COMPLETE in approximately 30 minutes.

# Step 2. Run the model generation pipeline

Use the `variant_classifier-autopilot` Jupyter notebook to create and run the model generation pipeline. Use the following procedure to execute the notebook.

1. Sign in to the AWS Secrets Manager console.
2. Select **Notebook instances** under the **Notebook** category on the left pane.
3. Select your notebook instance and choose **Open Jupyter**.
4. Select **variant_classifier-autopilot.ipynb**.
5. Follow the **Setup** instructions in the notebook to run the model generation pipeline.

# Step 3. Generate predictions and evaluate model performance

Use the `variant_predictor` Jupyter notebook to generate model predictions and evaluate model performance using test data. Use the following procedure to execute the notebook.

1. Sign in to the Amazon SageMaker console.
2. Select **Notebook instances** under the **Notebook** category on the left pane.
3. Select your notebook instance and choose **Open Jupyter**.
4. Select **variant_predictor.ipynb**.
5. Follow the **Setup** instructions in the notebook to generate predictions and evaluate model performance.

# Security

When you build systems on AWS infrastructure, security responsibilities are shared between you and AWS. This shared model can reduce your operational burden as AWS operates, manages, and controls the components from the host operating system and virtualization layer down to the physical security of the facilities in which the services operate. For more information about security on AWS, visit the AWS Security Center.

This solution is preconfigured with all of the IAM policies and roles necessary to run the solution with least privileges.

## IAM roles

AWS Identity and Access Management (IAM) roles enable you to secure jobs and crawlers running in AWS Glue, and restrict access to the data catalog, the Amazon Simple Storage Service (Amazon S3) bucket, and the Amazon SageMaker notebook instance. All of the IAM roles in this solution have been defined with least privileges. Refer to Appendix B (p. 15) for details about roles and permissions used in this solution.

# Additional resources

- AWS CloudFormation
- AWS CodeCommit
- AWS CodePipeline
- AWS CodeBuild
- AWS Glue
- Amazon SageMaker
- AWS Lambda
- Amazon Simple Storage Service

# Appendix A: Open source licenses

This solution is based on the Genetic Variant Classifications Kaggle project to predict whether a variant will have conflicting clinical classifications. The script used to create the training dataset is based on the `process_clinvar.py` script available in the project GitHub repository. The input dataset used in this solution for creating the solution training dataset is a combination of the publicly available ClinVar dataset with Variant Effect Predictor (VEP) annotations added. This dataset is located in the `clinvar.annotation.vcf.gz` file in the project GitHub repository.

| Tool Name | Description | License |
|---|---|---|
| Genetic Variants Classification Kaggle Project | Project on Kaggle to Predict whether a variant will have conflicting clinical classifications. For more information, refer to the project documentation. | CCO 1.0 |

# Appendix B: Roles and permissions

This solution uses AWS CodeBuild, AWS CloudFormation, and AWS CodePipeline for continuous delivery (CD). Review the following CodeBuild, AWS CloudFormation, CodePipeline, and AWS Glue permissions to ensure that you have the appropriate permissions enabled.

## Code deployment pipeline permissions

Use AWS Identity and Access Management (IAM) to manage access to AWS CodeBuild jobs, AWS CloudFormation stacks, and the AWS CodePipeline `code` pipeline. CodeBuild jobs and the CodePipeline `code` pipeline have their own IAM roles and IAM policies.

The following code examples demonstrate the IAM roles and supporting IAM policies defined in the `GenomicsAnalysisPipe/pipe_cfn.yml` file; including `CodeBuildRole`, `CodePipelineRole`, `CloudFormationRole`, and `SourceEventRole`.

### CloudFormation Role

`CloudFormationRole` defines the permissions needed for AWS CloudFormation to provision IAM roles, Amazon Simple Storage Service (Amazon S3) buckets, an Amazon SageMaker notebook instance, and AWS Glue resources. AWS CloudFormation uses the CloudFormation action type in the CodePipeline.

```
CloudFormationRole:
    Type: AWS::IAM::Role
    Properties:
      Path: /
      AssumeRolePolicyDocument:
        Version: 2012-10-17
        Statement:
          - Effect: Allow
            Action:
              - sts:AssumeRole
            Principal:
              Service:
                - cloudformation.amazonaws.com
      Policies:
        - PolicyName: CloudFormationRolePolicy
          PolicyDocument:
            Version: 2012-10-17
            Statement:
              - Effect: Allow
                Action:
                  - iam:GetRolePolicy
                Resource: '*'
              - Effect: Allow
                Action:
                  - iam:CreateRole
                  - iam:DeleteRole
                  - iam:PutRolePolicy
                  - iam:GetRolePolicy
                  - iam:DeleteRolePolicy
                  - iam:AttachRolePolicy
                  - iam:DetachRolePolicy
                  - iam:UpdateAssumeRolePolicy
                  - iam:PassRole
```

```
                - iam:GetRole
              Resource:
                - !Sub arn:aws:iam::${AWS::AccountId}:role/${ResourcePrefix}*
            - Effect: Allow
              Action:
                - glue:CreateJob
                - glue:UpdateJob
                - glue:DeleteJob
                - glue:GetJob
              Resource: '*'
            - Effect: Allow
              Action:
                - s3:CreateBucket
                - s3:DeleteBucket
                - s3:GetObject
              Resource:
                - !Sub ${BuildBucket.Arn}
                - !Sub ${BuildBucket.Arn}/*
            - Effect: Allow
              Action:
                - sagemaker:CreateNotebookInstanceLifecycleConfig
                - sagemaker:DescribeNotebookInstanceLifecycleConfig
                - sagemaker:UpdateNotebookInstanceLifecycleConfig
                - sagemaker:DeleteNotebookInstanceLifecycleConfig
                - sagemaker:CreateNotebookInstance
                - sagemaker:UpdateNotebookInstance
                - sagemaker:StartNotebookInstance
                - sagemaker:DescribeNotebookInstance
                - sagemaker:DeleteNotebookInstance
                - sagemaker:StopNotebookInstance
              Resource:
                - !Sub arn:aws:sagemaker:${AWS::Region}:${AWS::AccountId}:notebook-
instance-lifecycle-config/${ResourcePrefixLowercase}*
                - !Sub arn:aws:sagemaker:${AWS::Region}:${AWS::AccountId}:notebook-
instance/${ResourcePrefixLowercase}*
    Metadata:
      cfn_nag:
        rules_to_suppress:
          - id: W11
            reason: AWS Glue requires * resources for the spedified actions. Same for get
 role policy.
```

# CodeBuild Role

`CodeBuildRole` defines the permissions needed for CodeBuild to run a code build job that copies the resources needed to the `ResourcesBucket` S3 bucket. The CodeBuild job is run using the CodeBuild action type in the CodePipeline.

```
CodeBuildRole:
    Type: AWS::IAM::Role
    DependsOn: ResourcesBucket
    Properties:
      AssumeRolePolicyDocument:
        Version: 2012-10-17
        Statement:
          - Action:
              - sts:AssumeRole
            Effect: Allow
            Principal:
              Service:
                - codebuild.amazonaws.com
      Path: /
      Policies:
```

```
        - PolicyName: CodeBuildAccess
          PolicyDocument:
            Version: 2012-10-17
            Statement:
              - Effect: Allow
                Resource:
                  - !Sub arn:aws:logs:${AWS::Region}:${AWS::AccountId}:log-group:/aws/
codebuild/${ResourcePrefix}*
                Action:
                  - logs:CreateLogGroup
                  - logs:CreateLogStream
                  - logs:PutLogEvents
              - Effect: Allow
                Action:
                  - s3:GetObject
                  - s3:GetObjectVersion
                  - s3:PutObject
                Resource: !Sub ${BuildBucket.Arn}/*
              - Effect: Allow
                Action:
                  - s3:ListBucket
                Resource:
                  - !Sub ${ResourcesBucket.Arn}
                  - !Sub ${DataLakeBucket.Arn}
              - Effect: Allow
                Action:
                  - s3:PutObject
                  - s3:PutObjectAcl
                Resource:
                  - !Sub ${ResourcesBucket.Arn}
                  - !Sub ${ResourcesBucket.Arn}/*
                  - !Sub ${DataLakeBucket.Arn}
                  - !Sub ${DataLakeBucket.Arn}/*
```

# CodePipeline Role

`CodePipelineRole` defines the permissions needed for CodePipeline to run a deployment pipeline that copies resources to the `ResourcesBucket` S3 bucket, and provisions the AWS Glue resources needed to process the solution data.

```
CodePipelineRole:
    Type: AWS::IAM::Role
    Properties:
      AssumeRolePolicyDocument:
        Version: 2012-10-17
        Statement:
          - Action:
              - sts:AssumeRole
            Effect: Allow
            Principal:
              Service:
                - codepipeline.amazonaws.com
      Path: /
      Policies:
        - PolicyName: CloudFormationAccess
          PolicyDocument:
            Version: 2012-10-17
            Statement:
              - Action:
                  - cloudformation:CreateStack
                  - cloudformation:DescribeStacks
                  - cloudformation:UpdateStack
                Effect: Allow
```

```
                Resource: !Sub arn:aws:cloudformation:${AWS::Region}:
${AWS::AccountId}:stack/${ResourcePrefix}/*
        - PolicyName: S3Access
          PolicyDocument:
            Version: 2012-10-17
            Statement:
              - Effect: Allow
                Action:
                  - s3:GetObject
                  - s3:GetObjectVersion
                  - s3:GetBucketVersioning
                  - s3:DeleteObject
                  - s3:PutObject
                Resource:
                  - !Sub ${BuildBucket.Arn}
                  - !Sub ${BuildBucket.Arn}/*
        - PolicyName: CodeBuildAccess
          PolicyDocument:
            Version: 2012-10-17
            Statement:
              - Action:
                  - codebuild:StartBuild
                  - codebuild:BatchGetBuilds
                Effect: Allow
                Resource:
                  - !GetAtt CodeBuildCopyResourcesProject.Arn
        - PolicyName: IamAccess
          PolicyDocument:
            Version: 2012-10-17
            Statement:
              - Action:
                  - iam:PassRole
                Effect: Allow
                Resource: !GetAtt CodeBuildRole.Arn
        - PolicyName: IamAccessCF
          PolicyDocument:
            Version: 2012-10-17
            Statement:
              - Action:
                  - iam:PassRole
                Effect: Allow
                Resource: !Sub ${CloudFormationRole.Arn}
        - PolicyName: CodeCommitAccess
          PolicyDocument:
            Version: 2012-10-17
            Statement:
              - Effect: Allow
                Action:
                  - codecommit:UploadArchive
                  - codecommit:GetBranch
                  - codecommit:GetCommit
                  - codecommit:GetUploadArchiveStatus
                Resource: !GetAtt Repo.Arn
```

# Source Event Role

`SourceEventRole` defines the permissions needed for an Amazon CloudWatch event to trigger the deployment pipeline.

```
SourceEventRole:
    Type: AWS::IAM::Role
    DependsOn: CodePipeline
    Description: IAM role to allow Amazon CloudWatch Events to trigger AWS CodePipeline
```

```
      execution
    Properties:
      AssumeRolePolicyDocument:
        Statement:
          - Action: sts:AssumeRole
            Effect: Allow
            Principal:
              Service:
                - events.amazonaws.com
            Sid: 1
      Policies:
        - PolicyName: CloudWatchEventPolicy
          PolicyDocument:
            Statement:
              - Action:
                  - codepipeline:StartPipelineExecution
                Effect: Allow
                Resource:
                  - !Sub arn:aws:codepipeline:${AWS::Region}:${AWS::AccountId}:
${CodePipeline}*
```

# AWS Glue and Amazon SageMaker notebook permissions

Use IAM to manage access to the datasets and scripts in Amazon S3 using AWS Glue, and to define the permissions for your Amazon SageMaker Jupyter notebook instance. Adding new AWS Glue jobs and crawlers does not require any changes to the following roles or policies, but you must add those resources with the `${Project}` prefix.

The following code examples demonstrate the IAM roles and supporting IAM policies defined in the `GenomicsLearningCode/code_cfn.yml` file, including `JobRole`, and `RunbookRole`.

## Job role

`JobRole` defines the permissions needed for AWS Glue to run Extract, Transform, and Load (ETL).

This role must be updated to `get` or `put` objects in additional S3 buckets in your AWS account, or S3 buckets in other accounts.

```
JobRole:
    Type: AWS::IAM::Role
    Properties:
      AssumeRolePolicyDocument:
        Version: 2012-10-17
        Statement:
          - Effect: Allow
            Principal:
              Service:
                - glue.amazonaws.com
            Action:
              - sts:AssumeRole
      Path: /
      ManagedPolicyArns:
        - arn:aws:iam::aws:policy/service-role/AWSGlueServiceRole
      Policies:
        - PolicyName: s3_access
          PolicyDocument:
            Version: 2012-10-17
```

```
            Statement:
              - Effect: Allow
                Action:
                  - s3:GetObject
                  - s3:ListBucket
                Resource:
                  - !Sub arn:aws:s3:::${ResourcesBucket}
                  - !Sub arn:aws:s3:::${ResourcesBucket}/*
              - Effect: Allow
                Action:
                  - s3:PutObject
                  - s3:GetObject
                  - s3:ListBucket
                  - s3:DeleteObject
                Resource:
                  - !Sub arn:aws:s3:::${DataLakeBucket}
                  - !Sub arn:aws:s3:::${DataLakeBucket}/*
```

# Runbook role

`RunbookRole` provides the permissions needed for the Amazon SageMaker Jupyter notebook instance to access the Amazon S3 buckets, and to use Amazon SageMaker to create machine learning models and to generate predictions.

```
RunbookRole:
    Type: AWS::IAM::Role
    Properties:
      AssumeRolePolicyDocument:
        Version: 2012-10-17
        Statement:
          - Effect: Allow
            Principal:
              Service:
                - sagemaker.amazonaws.com
            Action:
              - sts:AssumeRole
      Path: /
      Policies:
        - PolicyName: logs_access
          PolicyDocument:
            Version: 2012-10-17
            Statement:
              - Effect: Allow
                Action:
                  - logs:CreateLogStream
                  - logs:DescribeLogStreams
                  - logs:CreateLogGroup
                  - logs:PutLogEvents
                  - logs:GetLogEvents
                Resource:
                  - !Sub arn:aws:logs:${AWS::Region}:${AWS::AccountId}:log-group:/aws/
sagemaker/*
        - PolicyName: s3_access
          PolicyDocument:
            Version: 2012-10-17
            Statement:
              - Effect: Allow
                Action:
                  - s3:CreateBucket
                  - s3:ListBucket
                Resource:
                  - !Sub arn:aws:s3:::sagemaker-${AWS::Region}-${AWS::AccountId}
              - Effect: Allow
```

```
                Action:
                  - iam:GetRole
                  - sagemaker:DescribeNotebookInstance
                Resource: '*'
              - Effect: Allow
                Action:
                  - s3:ListBucket
                  - s3:GetBucketLocation
                Resource:
                  - !Sub arn:aws:s3:::${DataLakeBucket}
                  - !Sub arn:aws:s3:::${ResourcesBucket}
              - Effect: Allow
                Action:
                  - s3:GetObject
                  - s3:PutObject
                  - s3:DeleteObject
                Resource:
                  - !Sub arn:aws:s3:::${DataLakeBucket}/*
                  - !Sub arn:aws:s3:::sagemaker-${AWS::Region}-${AWS::AccountId}/*
              - Effect: Allow
                Action:
                  - s3:GetObject
                Resource:
                  - !Sub arn:aws:s3:::${ResourcesBucket}/*
        - PolicyName: glue_access
          PolicyDocument:
            Version: 2012-10-17
            Statement:
              - Effect: Allow
                Action:
                  - glue:StartJobRun
                  - glue:StopJobRun
                Resource:
                  - !Sub arn:aws:glue:${AWS::Region}:${AWS::AccountId}:job/
${ResourcePrefix}*
        - PolicyName: cfn_access
          PolicyDocument:
            Version: 2012-10-17
            Statement:
              - Effect: Allow
                Action:
                  - cloudformation:DescribeStacks
                Resource:
                  - !Sub arn:aws:cloudformation:${AWS::Region}:${AWS::AccountId}:stack/
${ResourcePrefix}Pipe/*
                  - !Sub arn:aws:cloudformation:${AWS::Region}:${AWS::AccountId}:stack/
${ResourcePrefix}*
        - PolicyName: sagemaker_access
          PolicyDocument:
            Version: 2012-10-17
            Statement:
              - Effect: Allow
                Action:
                  - iam:CreateServiceLinkedRole
                Resource:
                  - !Sub arn:aws:iam::*:role/aws-service-role/sagemaker.application-
autoscaling.amazonaws.com/AWSServiceRoleForApplicationAutoScaling_SageMakerEndpoint
                Condition:
                  StringLike:
                    iam:AWSServiceName: sagemaker.application-autoscaling.amazonaws.com
              - Effect: Allow
                Action:
                  - iam:CreateServiceLinkedRole
                Resource: '*'
                Condition:
                  StringEquals:
```

```
                  iam:AWSServiceName: robomaker.amazonaws.com
              - Effect: Allow
                Action:
                  - iam:PassRole
                Resource:
                  - !Sub arn:aws:iam::*:role/*
                Condition:
                  StringEquals:
                    iam:PassedToService:
                      - sagemaker.amazonaws.com
                      - glue.amazonaws.com
                      - robomaker.amazonaws.com
                      - states.amazonaws.com
              - Effect: Allow
                Action:
                  - sagemaker:ListEndpoints
                Resource: '*'
              - Effect: Allow
                Action:
                  - sagemaker:DescribeTrainingJob
                  - sagemaker:DescribeTransformJob
                  - sagemaker:CreateTrainingJob
                  - sagemaker:CreateAutoMLJob
                  - sagemaker:CreateTransformJob
                  - sagemaker:StopTransformJob
                  - sagemaker:CreateHyperParameterTuningJob
                  - sagemaker:StopHyperParameterTuningJob
                  - sagemaker:DescribeHyperParameterTuningJob
                  - sagemaker:DescribeEndpoint
                  - sagemaker:DescribeEndpointConfig
                  - sagemaker:CreateEndpointConfig
                  - sagemaker:CreateEndpoint
                  - sagemaker:InvokeEndpoint
                  - sagemaker:ListTrainingJobsForHyperParameterTuningJob
                  - sagemaker:CreateModel
                  - sagemaker:ListTags
                  - logs:GetLogEvents
                  - sagemaker:DeleteModel
                  - sagemaker:StopAutoMLJob
                  - sagemaker:ListAutoMLJob
                  - sagemaker:DescribeAutoMLJob
                Resource:
                  - !Sub arn:aws:sagemaker:${AWS::Region}:${AWS::AccountId}:training-job*
                  - !Sub arn:aws:sagemaker:${AWS::Region}:${AWS::AccountId}:automl-job*
                  - !Sub arn:aws:sagemaker:${AWS::Region}:${AWS::AccountId}:transform-job*
                  - !Sub arn:aws:sagemaker:${AWS::Region}:${AWS::AccountId}:model*
                  - !Sub arn:aws:sagemaker:${AWS::Region}:${AWS::AccountId}:endpoint*
                  - !Sub arn:aws:sagemaker:${AWS::Region}:${AWS::AccountId}:hyper-
parameter-tuning-job*
              - Effect: Allow
                Action:
                  - iam:ListRoles
                Resource: '*'
    Metadata:
      cfn_nag:
        rules_to_suppress:
          - id: W11
            reason: GetRole, DescribeSageMakerRole, ListRoles and CreateServiceLinkedRole
 require star.
```

# Appendix C: Run the create-training-set AWS Glue job

This solution includes an example AWS Glue job that demonstrates how to create a machine learning training dataset using AWS Glue. You can run the job using either the AWS Command Line Interface (AWS CLI) or the AWS Glue console.

To start the job using the AWS CLI, run the following command:

```
aws glue start-job-run --name create-training-set
```

**Note:**
The Amazon Simple Storage Service (Amazon S3) bucket name can be found as the `DataLakeBucket` output value in the `GenomicsLearningPipe` CloudFormation stack.

Use the following procedure to run the AWS Glue job in the AWS Glue console.

1. Sign in to the AWS Glue console.
2. Choose **Jobs** under the **ETL** category on the left pane. On the **Jobs** page, select **create-training-set**.
3. Choose **Actions** and select **Run Job**.
4. Choose **Run job**.

The job takes a few minutes to run, then the output is copied to the `DataLakeBucket` S3 bucket. You can learn more about the job input and output locations in the AWS Glue job details pane.

# Appendix D: Solution notebooks

The solution includes the following Amazon SageMaker Jupyter notebooks:

- **variant_classifier-autopilot –** This notebook reads the raw dataset from Amazon Simple Storage Service (Amazon S3) and divides the data into a train file and a test file. These files are saved locally in the notebook instance and in Amazon S3. This notebook then starts an Amazon SageMaker Autopilot job that produces two Jupyter notebooks:
  - **AmazonSageMakerAutopilotDataExploration –** This notebook is produced during the analysis phase of the SageMaker Autopilot `AutoML` job that helps you identify problems in your dataset. It identifies specific areas for investigation to help you identify upstream problems with your data that may result in a suboptimal model.
  - **AmazonSageMakerAutopilotCandidateDefinitionNotebook –** This notebook contains the suggested preprocessing steps, algorithms, and hyperparameter ranges. At the end of this notebook, you deploy a binary classification model on a SageMaker endpoint.
- **variant_predictor –** This notebook reads the test data from the CSV file that was created in the variant_classifier-autopilot notebook and generates predictions for them by invoking the SageMaker endpoint created in the `AmazonSagemakerCandidateDefinitionNotebook`. This notebook then creates evaluation metrics and plots showing the performance of the model.

The `AmazonSageMakerAutopilotCandidateDefinitionNotebook` and the `AmazonSageMakerAutopilotDataExploration` notebooks capture the data structures and statistics, feature engineering steps, algorithm selection, hyperparameter tuning steps, and the deployment of the model that provides the best accuracy metric based on the experiments conducted.

You can use the default options provided in Sagemaker Autopilot or customize these options by modifying the `AmazonSageMakerAutopilotCandidateDefinitionNotebook` Jupyter notebook to test your specific scenarios.

# Appendix E: Uninstall this solution

You can uninstall the Genomics Tertiary Analysis and Machine Learning Using Amazon SageMaker solution using the AWS Management Console, the AWS Command Line Interface (AWS CLI), or manually.

**Note:**
Uninstalling this solution deletes the Amazon Simple Storage Service (Amazon S3) solution buckets and the data in those buckets; AWS CodeCommit repositories and the code in those repositories; the AWS Glue jobs and data; and the Amazon SageMaker notebook instance.

## Using the AWS Management Console

1. Sign in to the AWS CloudFormation console.
2. Select the solution's installation stack that has a name ending in `-Setup`. All other solutions stacks will be deleted automatically.
3. Choose **Delete**.

## Using AWS CLI

Determine whether 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 the AWS CLI is available, run the following command.

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

## Uninstalling manually

To manually uninstall this solution, you must delete the related AWS CloudFormation stacks using the following procedure, in the specified order.

1. Delete all *<project-name>-*bucket* Amazon S3 bucket contents.
2. Delete the *<ProjectName>Code* stack.

   **Important**
   Wait for deletion to complete successfully before proceeding.
3. Delete the *<ProjectName>Pipe* stack.

   **Important:**
   Wait for deletion to complete successfully before proceeding.
4. Delete the *<ProjectName>Zone* stack.

   **Important**
   Wait for deletion to complete successfully before proceeding.
5. Delete the solution's installation stack.
6. Delete the *<ProjectName>Code* AWS CodeCommit repository.
7. Delete the *<ProjectName>Pipe* AWS CodeCommit repository.
8. Delete all *<project-name>-** image repositories in Amazon ECR.

# Appendix F: 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 enabled, the following information is collected and sent to AWS:

- **Solution ID:** The AWS solution identifier.
- **Unique ID (UUID):** Randomly generated, unique identifier for each solution deployment.
- **Timestamp:** Data-collection timestamp.
- **Instance Data:** Count of the state and type of instances that are managed by the EC2 Scheduler in each AWS Region.

Example data:

```
Running: {t2.micro: 2}, {m3.large:2}

Stopped: {t2.large: 1}, {m3.xlarge:3}
```

Note that AWS owns the data gathered via this survey. Data collection will be subject to the AWS Privacy Policy. To opt out of this feature, complete the following task.

Modify the AWS CloudFormation template mapping section as follows:

```
"Send" : {
"AnonymousUsage" : { "Data" : "Yes" }
},
```

to

```
"Send" : {
"AnonymousUsage" : { "Data" : "No" }
},
```

# Source code

You can visit our GitHub repository to download the templates and scripts for this solution, and to share your customizations with others.

# Document revisions

| Date | Change |
| --- | --- |
| August 2020 | *Initial release* |