Machine Learning for Telecommunication

AWS Implementation Guide

Vijay Sathish

November 2018

Last updated: December 2019 (see revisions)

Contents

Overview 3
Cost 3
Architecture Overview 4
Implementation Considerations 5
The Jupyter Notebook 5
AWS Glue 6
Demo Data 7
Amazon SageMaker Instance Types 7
AWS CloudFormation Template 8
Automated Deployment 8
What We'll Cover 8
Step 1. Launch the Stack 8
Step 2. Run the Notebooks 10
Security 11
Additional Resources 12
Appendix A: Using Your Own Data 13
Appendix B: Collection of Operational Metrics 14
Source Code 15
Document Revisions 15

About This Guide

This implementation guide discusses architectural considerations and configuration steps for deploying the Machine Learning for Telecommunication solution on the Amazon Web Services (AWS) Cloud. It includes links to an AWS CloudFormation template that launches, configures, and runs the AWS services required to deploy this solution on AWS using AWS best practices for security and availability.

The guide is intended for data scientists, chief data officers, and data engineers who have practical experience architecting on the AWS Cloud.

Overview

Machine learning (ML) helps Amazon Web Services (AWS) customers use historical data to predict future outcomes, which can lead to better business decisions. ML techniques are core to the communications service provider (CSP) industry. CSPs can use ML algorithms to construct and refine mathematical models from their business data, and then use those models to help identify fraudulent use of network services, automate network functions (zero touch), and reduce customer churn.

AWS offers several ML services and tools tailored for a variety of use cases and levels of expertise. However, it can be a challenge to understand the mechanics of model training and tuning, identify relevant data features, design a workflow that can perform complex extraction, transformation, load (ETL) activities, and scale to accommodate large datasets.

To help customers get started with a machine learning workflow for CSP use cases, AWS offers the Machine Learning for Telecommunication solution. This solution uses AWS CloudFormation to deploy a scalable, customizable ML architecture that leverages Amazon SageMaker, a fully managed ML service, and The Jupyter Notebook, an open source web application for creating and sharing live code, equations, visualizations, and narrative text.

Additionally, the solution provides a sample demo of ad-hoc data exploration, data processing and feature engineering, and model training and evaluation. It also includes a synthetic telecom IP Data Record (IPDR) dataset to demonstrate how to use ML algorithms to test and train models for predictive analysis in telecommunication. Customers can use the included notebooks as a starting point to develop their own custom ML models, and customize the included notebooks for their own use case and datasets.

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 **$1.21 per hour**. This includes charges for Amazon SageMaker, Amazon Simple Storage Service (Amazon S3), and AWS Glue. This estimate does not include variable charges for data processing and data transfer costs.

Prices are subject to change. For full details, see the pricing webpage for each AWS service you will use in this solution.

Architecture Overview

Deploying this solution builds the following environment in the AWS Cloud.

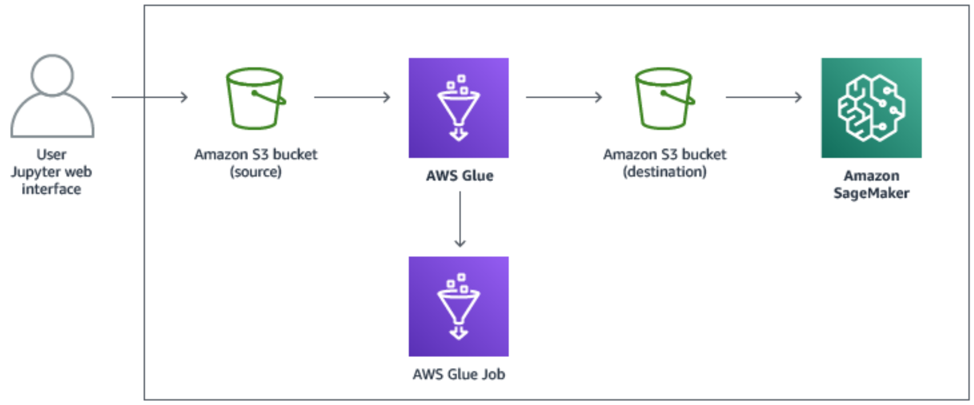

Figure 1: Machine Learning for Telecommunication architecture on AWS

The AWS CloudFormation template deploys an Amazon Simple Storage Service (Amazon S3) bucket that includes synthetic IP Data Record (IPDR) datasets in Abstract Syntax Notation One (ASN.1) format or call detail record (CDR) format. The template also deploys an AWS Glue job to convert the dataset from CSV to Parquet compressed format, and an Amazon SageMaker instance with Machine Learning (ML) Jupyter notebooks. The solution's Jupyter notebooks include instructions that guide the user through the process of constructing additional data featues for future models.

When the workflow is triggered in the Jupyter web interface, the solution ingests data from the Amazon S3 bucket into the Amazon SageMaker cluster which enables the user to run Jupyter notebooks on the dataset. Amazon S3 Select reads the Parquet compressed data that was processed by the AWS Glue job. The notebooks preprocess the data, extract features, and divide the data into training and testing. ML algorithms process the training dataset to develop a model to identify anomalies and predict future anomalies. The solution then tests the results of the model's predictions against the testing dataset, identifies false positives and negatives, and retrains to fine-tune the model.

Implementation Considerations

The Jupyter Notebook

The Machine Learning for Telecommunication solution uses The Jupyter Notebook, an open source web application that allows you to create and share live code, equations, visualizations, and narrative text.

The custom Jupyter notebooks have machine learning (ML) models that can identify and predict insights such as call-characteristic anomalies. The notebooks use a default set of predefined, call detail record (CDR) related features to train the model, test the model, then compute the area under the curve (AUC) metric based on the model scores on the test data, plot the receiver operating characteristic (ROC) curve to better understand the model performance, and then report the false positives, false negatives, true positives, and true negatives using the confusion matrix. Users will need to modify the notebook code and features to create different ML models for their specific use cases and datasets.

The Jupyter notebooks that is included in the solution are:

`Ml-Telecom-NaiveBayes.ipynb`: This notebook demonstrates the exploration of CDR data and classification with an Apache Spark ML Naïve Bayes Algorithm. The notebook reads the Parquet data with Amazon S3 Select for performance optimization, and showcases visualization of data exploration with bar charts and box plots. Additionally, it classifies and predicts the probability of call disconnect reason code 16, and further displays the ML evaluation steps with the confusion matrix and cross validation.

`Ml-Telecom-PCA-KMeans.ipynb`: This notebook demonstrates unsupervised learning with Apache Spark ML on CDR data with principal component analysis (PCA) and k-means. The notebook reads the Parquet data with Amazon S3 Select for performance optimization, showcases feature extraction with PCA and an elbow graph for optimizing the choice of k for k-means clustering. Additionally, it further displays the exploration of the data features with a correlation matrix heat map, visualizes the k-means clustering data in 3D and 2D, and further evaluates the k-means cluster by creating centroids.

`Ml-Telecom-RandomForestClassifier.ipynb`: This notebook uses a supervised learning algorithm of the Apache Spark ML RandomForest Classifier for classifying the call disconnect reason in CDR Data. The notebook reads the Parquet data with Amazon S3 Select for performance optimization, showcases exploratory data analysis with the ML concepts of feature selection and correlation heat map matrix. Additionally, it uses ChiSqSelector for figuring feature importances, and further evaluates the model with a confusion matrix, AUC, ROC, and precision recall metrics.

`Ml-Telecom-TimeSeries-RandomForestClassifier-DeepAR.ipynb:`
This notebook demonstrates the DeepAR supervised learning algorithm for forecasting Scalar Time-Series with telecom data. The notebook uses a hybrid approach of using Apache Spark ML for classifying the call disconnect reason as an anomaly and Amazon DeepAR for the time series prediction of the call disconnect reason anomaly. Additionally, it further demonstrates how to prepare the dataset for time series training with DeepAR and how to use the trained model for inference, and uses the context length and prediction length to showcase the time series prediction graph.

`Ml-Telecom-RandomCutForest.ipynb`: This notebook demonstrates unsupervised anomaly detection on time series data with the Amazon SageMaker Random Cut Forest algorithm. The notebook uses the call service duration of CDR data for anomaly detection. Additionally, it further showcases data exploration by plotting the data, automatic model tuning and calculating anomaly scores on the data, and displays concepts of data shingling and prediction.

AWS Glue

The Machine Learning for Telecommunication solution invokes an AWS Glue job during the solution deployment to process the synthetic call detail record (CDR) data or the customer's data to convert from CSV to Parquet format. By default, the AWS Glue job deploys 10 data processing units (DPUs) for preprocessing and can be scheduled with a scheduler. Note that when you use your own dataset, you need to modify the schema definitions to meet your data attributes to enable the AWS Glue job to run successfully.

Demo Data

This solution includes synthetic demo IP Data Record (IPDR) datasets in Abstract Syntax Notation One (ASN.1) format and call detail record (CDR) format. You can choose to copy these datasets into the source bucket to run the demo of AWS Glue transformations and ML model predictions, or use your own datasets.

The sample datasets descriptions in the Call Detail Record are as follows:

- **Call Detail Record – Start**: Indicates that the call has been successfully established/connected and a session has successfully started.
- **Call Detail Record – Stop**: Indicates that records that previously established a successful session are now terminated.
- **Call Detail Record – Start Sample**: A subset of the CDR start dataset
- **Call Detail Record – Stop Sample**: A subset of the CDR stop dataset

Amazon SageMaker Instance Types

This solution features three instance types for Amazon SageMaker (ml.t2.medium, ml.m4.xlarge, ml.p2.xlarge). By default, the solution deploys an ml.t2.medium instance. Note that you can provision an ml.p2.xlarge instance through the solution by changing the **Notebook Instance Type** template parameter. This instance allows Jupyter notebooks with large amounts of data to be processed faster, but will incur additional charges.

AWS CloudFormation Template

This solution uses AWS CloudFormation to automate the deployment of the Machine Learning for Telecommunication solution on the AWS Cloud. It includes the following CloudFormation template, which you can download before deployment:

View template

machine-learning-for-telecommunication.template: Use this template to launch the Machine Learning for Telecommunication solution and all associated components. The default configuration deploys an Amazon Simple Storage Service (Amazon S3) bucket, an AWS Glue job, and an Amazon SageMaker instance, but you can also customize the template based on your specific network needs.

Automated Deployment

Before you launch the automated deployment, please review the architecture, security, and other information discussed in this guide. Follow the step-by-step instructions in this section to configure and deploy the Machine Learning for Telecommunication solution into your account.

Time to deploy: Approximately five minutes

What We'll Cover

The procedure for deploying this architecture on AWS consists of the following steps. For detailed instructions, follow the links for each step.

Step 1. Launch the stack

- Launch the AWS CloudFormation template into your AWS account.
- Enter values for required parameters: **Source Bucket, Destination Bucket**
- Review the other template parameters and adjust if necessary.

Step 2. Run the Notebooks

- Test the machine learning algorithms and run the notebooks.

Step 1. Launch the Stack

This automated AWS CloudFormation template deploys the Machine Learning for Telecommunication solution on the AWS Cloud.

> **Note**: You are responsible for the cost of the AWS services used while running this solution. See the Cost section for more details. For full details, see the pricing webpage for each AWS service you will be using in this solution.

1. Sign in to the AWS Management Console and click the button to the right to launch the `machine-learning-for-telecommunication` AWS CloudFormation template.

   Launch Solution

   You can also download the template as a starting point for your own implementation.

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

> **Note**: This solution uses Amazon SageMaker, which is currently available in specific AWS Regions only. Therefore, you must launch this solution in an AWS Region where Amazon SageMaker is available. [1]

2. On the **Specify Details** page, assign a name to your solution stack.
3. Under **Parameters**, review the parameters for the template and modify them as necessary. This solution uses the following default values.

Parameter	Default	Description
Industry	`telecom`	The industry for which the solution is deployed
Source Bucket	*<Requires input>*	Unique name for the solution-created Amazon S3 bucket where your application artifacts will be stored.
Source Prefix	`machine-learning-for-all/V1.0.0/data`	Folder location of the dataset for the solution to process. By default, this is the location of synthetic dataset
Destination Bucket	*<Requires input>*	Unique name for the solution-created Amazon S3 bucket where your AWS Glue processed Parquet data will be stored
Destination Prefix	`machine-learning-for-all/V1.0.0/parq-conv`	Folder location of the AWS Glue processed dataset where glue job converts from CSV to Parquet
Synthetic Data	`Yes`	Select `Yes` to transfer the synthetic demo dataset to the solution created Amazon S3 bucket and use it for modeling in the notebook. Select `No` to use your own dataset for modeling. **Note:** You may need to update the schema. For more information, see Appendix A.

[1] For the most current service availability by AWS Region, see https://aws.amazon.com/about-aws/global-infrastructure/regional-product-services/

Parameter	Default	Description
Cron Job	`cron(0 0 1/1 * ? *)`	AWS Glue job scheduler for execution at a specific time. Use this parameter for long running jobs
Start Transformation	`Yes`	Choose whether to start an ETL job transformation for converting CSV to Parquet format
Deploy SageMaker	`Yes`	Choose whether to deploy an Amazon SageMaker instance for execution of machine learning Jupyter notebooks
Notebook Instance Type	`ml.t2.medium`	Select the instance type of the Amazon SageMaker instance **Note:** We recommend a ml.p2.xlarge instance for notebooks with processes large amounts of data. For example the `ML-telecom-TimeSeries-RandomForestClassifier-DeepAR` notebook.
KmsKeyId	<Optional Input>	AWS KMS key ID used to encrypt data at rest on the ML storage volume attached to the notebook instance.

4. Choose **Next.**
5. On the **Options** page, choose **Next**.
6. On the **Review** page, review and confirm the settings. Be sure to check the box acknowledging that the template will create AWS Identity and Access Management (IAM) resources.
7. Choose **Create** 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 five minutes.

Note: In addition to the primary AWS Lambda functions, this solution includes the `SolutionHelper` Lambda function, which runs only during initial configuration or when resources are updated or deleted.

When running this solution, the `SolutionHelper` function is inactive. However, do not delete this function as it is necessary to manage associated resources.

Step 2. Run the Notebooks

1. In the AWS Management Console, navigate to the AWS CloudFormation stack **Outputs** tab.
2. Copy the **SageMakerInstance** name.
3. In the AWS Management Console, open the **SageMaker** instance.
4. Under files, choose **telecom**.

5. Open each of the notebooks: `Ml-Telecom-NaiveBayes.ipynb`, `Ml-Telecom-PCA-KMeans.ipynb`, and `Ml-Telecom-RandomForestClassifier.ipynb`.
6. Choose **Cell**, then select **Run all** in each of the opened notebooks.
7. Open the `Ml-Telecom-RandomCutForest.ipynp` notebook and update the **bucket_name** located in cell 2 with the value you provided in the **Source Bucket** template parameter.
8. Choose **Cell**, then select **Run all**.
9. Open the `Ml-Telecom-TimeSeries-RandomForestClassifier-DeepAR.ipynb` notebook, and update the **bucket_name** located in cell 18 with the value you provided in the **Source Bucket** template parameter.
10. Choose **Cell**, then select **Run all**.

Now you will be able to use the included Jupyter notebooks and the synthetic telecom dataset to demonstrate how to use machine learning algorithms to test and train models for time series predictive analysis in telecommunications.

Note: The AWS Glue job processes the data to convert from CSV to Parquet and make sure the conversion job is complete before execution of the notebook. The AWS Glue job for data conversion of synthetic telecommunication dataset takes approximately one minute.

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.

Additional Resources

AWS services documentation

- AWS CloudFormation
- AWS Glue
- Amazon SageMaker
- Amazon Simple Storage Service

Other resources

- Jupyter
- Apache Spark MLlib
- Call Detail Record

Appendix A: Using Your Own Data

This solution includes a synthetic dataset[2] for model-training purposes, and is intended for novice machine learning (ML) data scientists. If you choose to use the Jupyter notebooks against your own datasets, we recommend following the AWS best practices for uploading data into Amazon S3 to ensure that your data is uploaded quickly and securely.

Use the following steps to use your own datasets in the Jupyter notebooks:

1. Update the **Source Bucket** and **Source Prefix** location in AWS Cloudformation template to point to the location of your data in Amazon S3.
2. Transferring the synthetic demo data to your bucket to run the notebooks is optional. Modify the **Synthetic Data** to **No**, if the demo data transfer is not required.
3. Select a notebook to run.
4. Modify the value of the **bucket_name** variable to the Amazon S3 folder location of the Parquet files.
5. Choose **Cell**, then select **Run all**.

Note: AWS Glue charges by the amount of data that is scanned and processed. Customers who have large datasets can save on costs and achieve better performance if the data is partitioned, compressed, or converted into a columnar format such as, Apache Parquet format.

[2] We would like to thank Ribbon Communications for providing the synthetic dataset. The data was generated by test data generators, and is not customer or sensitive data.

Appendix B: 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 deployment
- **Timestamp:** Data-collection timestamp
- **Region:** The AWS Region where the solution is being deployed
- **DPU:** The number of DPUs used in the AWS Glue job
- **Instance Type**: The Amazon SageMaker instance type

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

a) Modify the AWS CloudFormation template mapping section as follows:

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

to

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

OR

b) After the solution has been launched, find the `SolutionHelper` function in the Lambda console and set the **SEND_ANONYMOUS_DATA** environment variable to `No`.

Source Code

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

Document Revisions

Date	Change
November 2018	Initial release
June 2019	Added information about demo data for modeling
December 2019	Upgraded the solution to Python 3.7