06a_customer_churn_gcloud_execution.md

Customer Churn using Serverless Spark through Google Cloud Shell

Goal - Data Preparation and Model Training for Detecting Customer Churn.

Following are the lab modules:

1. Understanding Data

The dataset used for this project are customer churn data and customer test data.

The dataset contains the following features:

- Churn - Binary field which represents customers who left/were retained within the last month
- Services that each customer has signed up for – phone, multiple lines, internet, online security, online backup, device protection, tech support, and streaming TV and movies
- Customer account information – how long they've been a customer, contract, payment method, paperless billing, monthly charges, and total charges
- Demographic info about customers – gender, age range, and if they have partners and dependents

Note: The following features refer to these same-host connections.

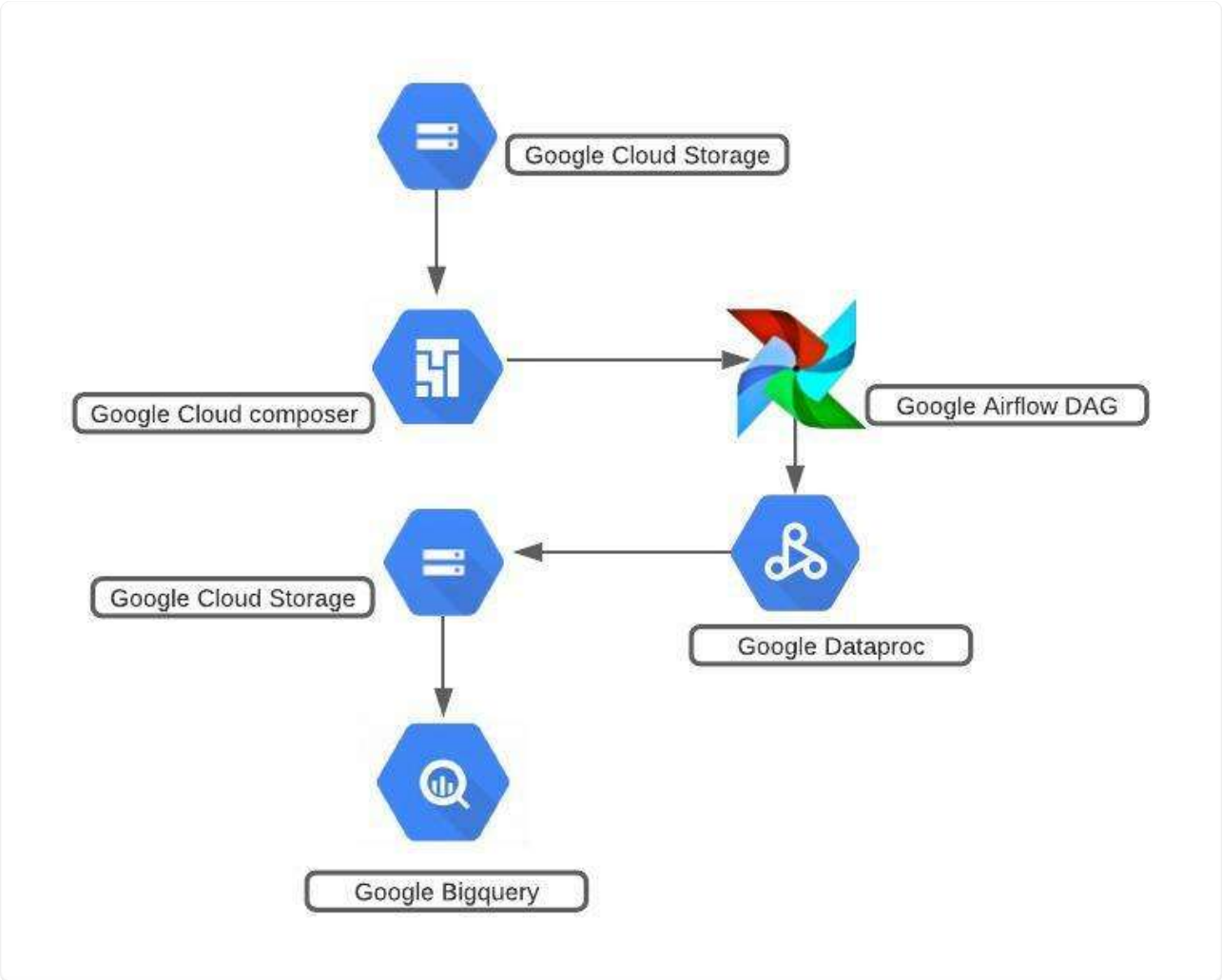
- serror_rate

- rerror_rate
- same_srv_rate
- diff_srv_rate
- srv_count

Note: The following features refer to these same-service connections.

- srv_serror_rate
- srv_rerror_rate
- srv_diff_host_rate

2. Solution Architecture



Model Pipeline

The model pipeline involves the following steps:

- Data cleanup and preparation
- Building and training two Machine Learning Models (Logistic Regression and Random Forest Classifier) before saving them into cloud storage
- Using the model built in above step to evaluate test data

3. Declaring cloud shell variables

3.1 Set the PROJECT_ID in Cloud Shell

Open Cloud shell or navigate to shell.cloud.google.com
Run the below

```
gcloud config set project $PROJECT_ID
```

3.2 Verify the PROJECT_ID in Cloud Shell

Next, run the following command in cloud shell to ensure that the current project is set correctly:

```
gcloud config get-value project
```

3.3 Declare the variables

Based on the prereqs and checklist, declare the following variables in cloud shell by replacing with your values:

```
PROJECT_ID=$(gcloud config get-value project)       #current GCP project where we ar
REGION=                                             #GCP region where all our resour
SUBNET=                                             #subnet which has private google
BUCKET_CODE=                                        #GCP bucket where our code, data
BUCKET_PHS=                                         #bucket where our application lc
HISTORY_SERVER_NAME=                                #name of the history server whic
BQ_DATASET_NAME=                                    #BigQuery dataset where all the
UMSA=serverless-spark                               #name of the user managed servic
SERVICE_ACCOUNT=$UMSA@$PROJECT_ID.iam.gserviceaccount.com
NAME=                                               #Your unique identifier
```

Note: For all the variables except 'NAME', please ensure to use the values provided by the admin team.

3.4 Update Cloud Shell SDK version

Run the below on cloud shell-

```
gcloud components update
```

4. Data Preparation

Based on EDA, the data preparation script has been created. Among the 21 columns, relevant features have been selected and stored in BQ for the next step of model training.

4.1. Run PySpark Serverless Batch for Data Preparation

Run the below on cloud shell -

```
gcloud dataproc batches submit \
  --project $PROJECT_ID \
  --region $REGION \
  pyspark --batch ${NAME}-batch-${RANDOM} \
  gs://$BUCKET_CODE/customer_churn/00-scripts/customer_churn_data_prep.py \
  --jars gs://spark-lib/bigquery/spark-bigquery-with-dependencies_2.12-0.22.2.jar \
  --subnet $SUBNET \
  --service-account $SERVICE_ACCOUNT \
  --history-server-cluster projects/$PROJECT_ID/regions/$REGION/clusters/$HISTORY_SE
  -- $PROJECT_ID $BQ_DATASET_NAME $BUCKET_CODE $NAME
```

4.2. Check the output table in BQ

Navigate to BigQuery Console, and check the **customer_churn_lab** dataset.
Once the data preparation batch is completed, a new table '<your_name_here>_training_data' and '<your_name_here>_test_data' will be created.

To view the data in these tables -

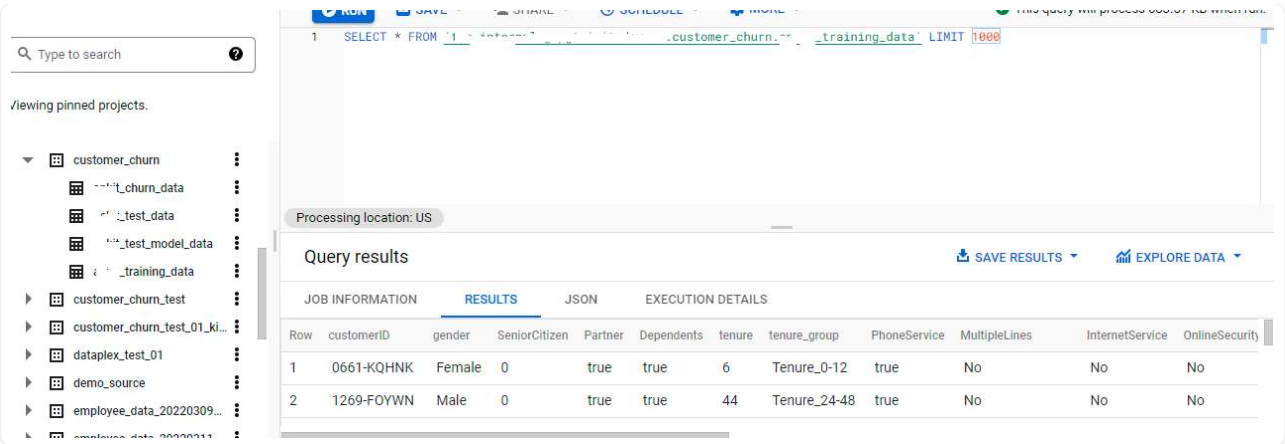
- Select the table from BigQuery Explorer by navigating 'project_id' > 'dataset' > 'table_name'
- Click on the **Preview** button to see the data in the table

Note: If the **Preview** button is not visible, run the below queries to view the data. However, these queries will be charged for the full table scan.

To query the table -

```
SELECT * FROM `<project_name>.<dataset_name>.<your_name_here>_training_data` LIMIT
SELECT * FROM `<project_name>.<dataset_name>.<your_name_here>_test_data` LIMIT 100
```

Note: Edit all occurrences of <project_name> and <dataset_name> to match the values of the variables PROJECT_ID, and BQ_DATASET_NAME respectively



5. Model Training and Testing

5.1. Run PySpark Serverless Batch for Model Training and Testing

The following script will train the model and save the model in the bucket.

Use the gcloud command below:

```
gcloud dataproc batches submit \
  --project $PROJECT_ID \
  --region $REGION \
  pyspark --batch ${NAME}-batch-${RANDOM} \
  gs://$BUCKET_CODE/customer_churn/00-scripts/customer_churn_model_building.py \
  --jars gs://spark-lib/bigquery/spark-bigquery-with-dependencies_2.12-0.22.2.jar  \
  --subnet $SUBNET \
  --service-account $SERVICE_ACCOUNT \
  --history-server-cluster projects/$PROJECT_ID/regions/$REGION/clusters/$HISTORY_SE
  -- $PROJECT_ID $BQ_DATASET_NAME $BUCKET_CODE $NAME
```

5.2. Query the model_test results BQ table

Navigate to BigQuery Console, and check the **customer_churn_lab** dataset.
Once the modelling batch is completed, a new table '<your_name_here>_predictions_data' will be created.

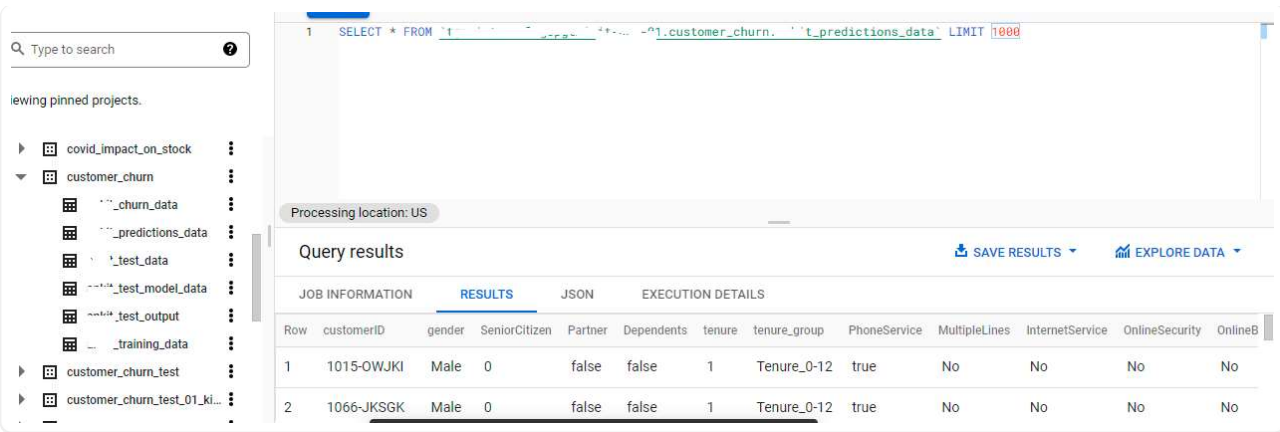
To view the data in this table -

- Select the table from BigQuery Explorer by navigating 'project_id' > 'dataset' > 'table_name'
- Click on the **Preview** button to see the data in the table

Note: If the **Preview** button is not visible, run the below queries to view the data. However, these queries will be charged for the full table scan.

```
SELECT * FROM `<project_name>.<dataset_name>.<your_name_here>_predictions_data` LI
```

Note: Edit all occurrences of <project_name> and <dataset_name> to match the values of the variables PROJECT_ID, and BQ_DATASET_NAME respectively



6. Model Evaluation

6.1. Run PySpark Serverless Batch for Model Evaluation

The following script will load the model and predict the new data.

Use the gcloud command below:

```
gcloud dataproc batches submit \
  --project $PROJECT_ID \
  --region $REGION \
  pyspark --batch ${NAME}-batch-${RANDOM} \
  gs://$BUCKET_CODE/customer_churn/00-scripts/customer_churn_model_testing.py \
  --jars gs://spark-lib/bigquery/spark-bigquery-with-dependencies_2.12-0.22.2.jar  \
  --subnet $SUBNET \
  --service-account $SERVICE_ACCOUNT \
  --history-server-cluster projects/$PROJECT_ID/regions/$REGION/clusters/$HISTORY_SE
  -- $PROJECT_ID $BQ_DATASET_NAME $BUCKET_CODE $NAME
```

6.2. Query the model_test results BQ table

Navigate to BigQuery Console, and check the **customer_churn_lab** dataset.
Once the model_testing batch is completed, a new table '<your_name_here>_test_output' will be created.

To view the data in this table -

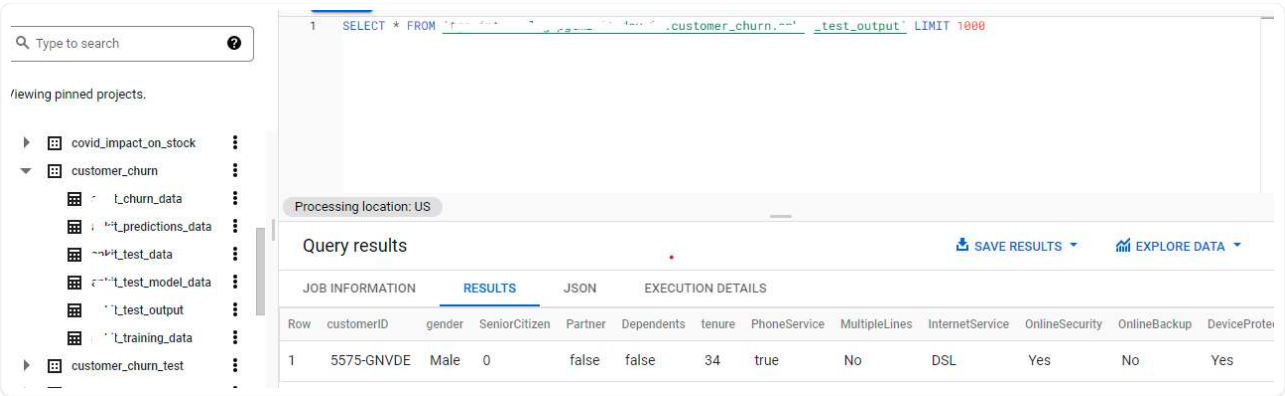
- Select the table from BigQuery Explorer by navigating 'project_id' > 'dataset' > 'table_name'

- Click on the **Preview** button to see the data in the table

Note: If the **Preview** button is not visible, run the below queries to view the data. However, these queries will be charged for the full table scan.

```
SELECT * FROM `<project_name>.<dataset_name>.<your_name_here>_test_output` LIMIT 1
```

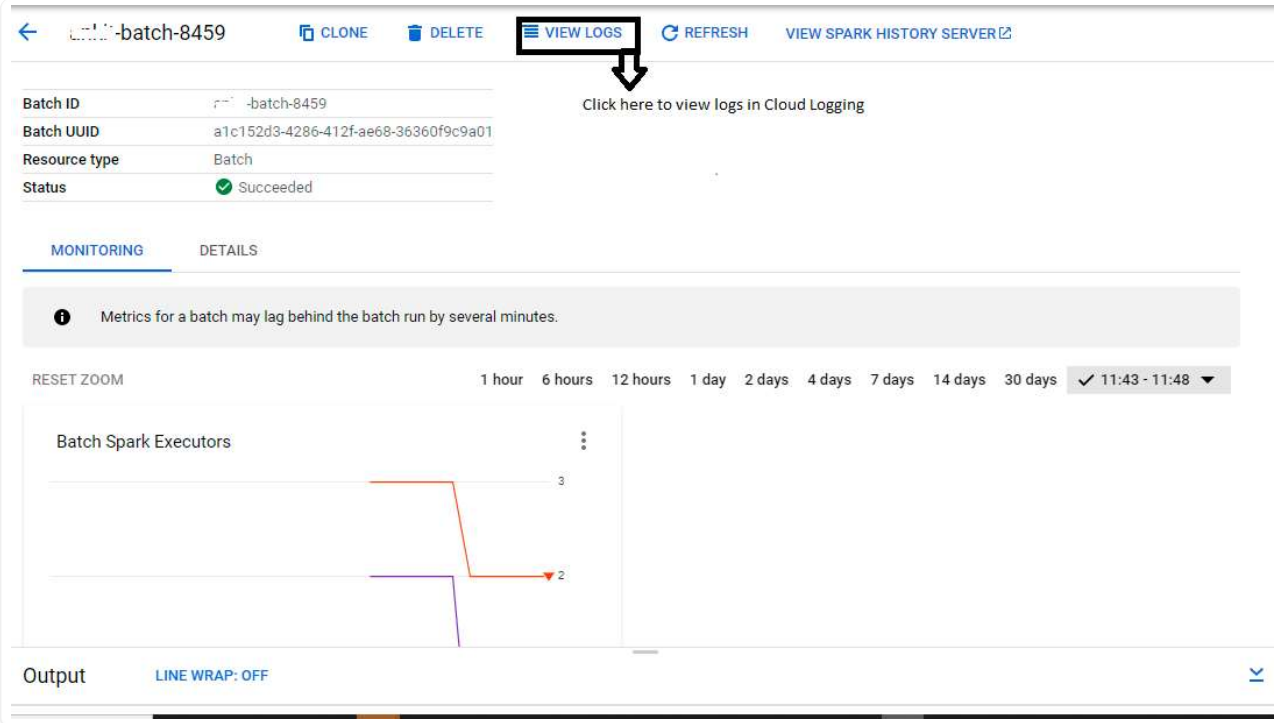
Note: Edit all occurrences of <project_name> and <dataset_name> to match the values of the variables PROJECT_ID, and BQ_DATASET_NAME respectively

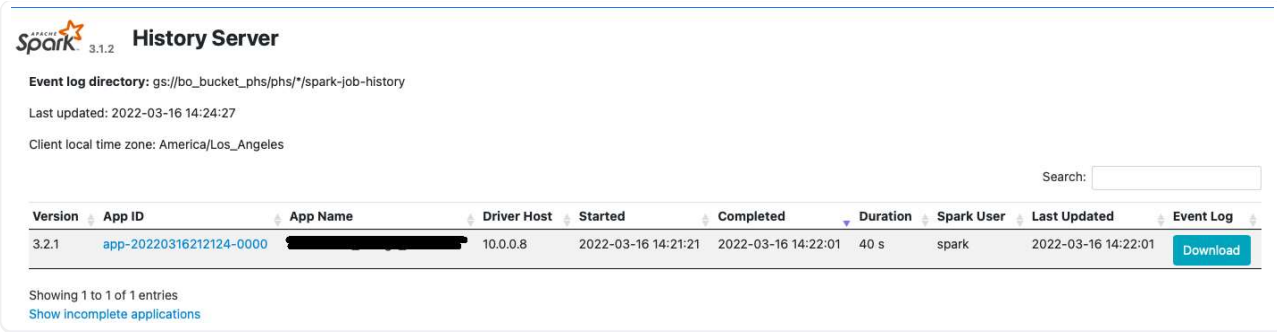


7. Logging

7.1 Serverless Batch logs

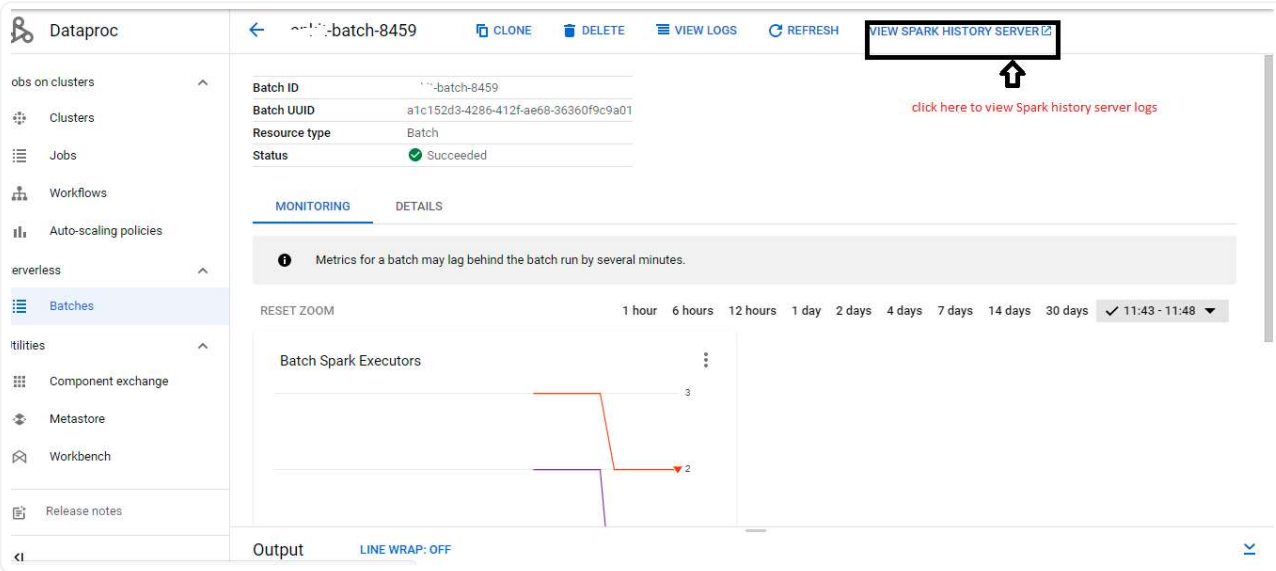
Logs associated with the application can be found in the logging console under **Dataproc > Serverless > Batches > <batch_name>**.
You can also click on "View Logs" button on the Dataproc batches monitoring page to get to the logging page for the specific Spark job.





7.2 Persistent History Server logs

To view the Persistent History server logs, click the 'View History Server' button on the Dataproc batches monitoring page and the logs will be shown as below:



History Server

Event log directory: gs://[illegible] /phs/*/spark-job-history

Last updated: 2022-03-30 13:32:41

Client local time zone: Asia/Calcutta

Search:

Version	App ID	App Name	Driver Host	Started	Completed	Duration	Spark User	Last Updated	Event Log
3.2.1	app-20220330061551-0000	churn model	10.202.0.56	2022-03-30 11:45:49	2022-03-30 11:46:37	47 s	spark	2022-03-30 11:46:38	Download

Showing 1 to 1 of 1 entries

Show incomplete applications