Customer Propensity- Google Analytics Sample Data - Assignment Documentation – Kanwalpreet Kaur

Objective - Using the google analytics 360 sample dataset on BigQuery, develop and deploy a model in VertexAI to predict when a 'hit' will have an event action of "Add To Cart". To start with, a 1- month period of your choice should be used to train the model, with the goal of being able to easily substitute in the 12-month period (full dataset) after validating the model.

Creating a VertexAI Instance

1. As a Pre-requisite, create a google cloud account first.
2. Enable the Notebooks API.
3. In the Google Cloud console, go to the Instances page.

   Go to Instances
4. Click + sign to Create new.
5. In the New instance dialog, click Advanced options.
6. In the Create instance dialog, in the Details section, provide the following information for your new instance:
   - Name: Name your new instance as Kaur-Capstone.
   - Region and Zone: Select a region as US-West1 and zone US-West1-B for the new instance.
7. In the Environment section, provide the following:
   - Version: Use the "M115" option of Vertex AI Workbench instances.
8. In the Machine type section, provide the following:
   - Machine type: Select the machine type as e2-standard-4 with 4 vCPU's and 16 GB of RAM for our new instance.

     Note: You can modify the machine type and GPU configuration for your instance after it is created.
   - Shielded VM: Optional. Select or clear the following checkboxes:
     - Secure Boot - Disabled
     - Virtual Trusted Platform Module (vTPM) - Enabled
     - Integrity monitoring - Enabled
   - Idle shutdown: - Enabled with Time of inactivity before shutdown (180 Minutes)
9. In the Disks section, provide the following:

- Disks: Select Data disk size as 100 GB.
- Delete to trash: Not selected
- Backup: Not selected.

10. In the Networking section, provide the following:
    a. In the Network field, select the “Default Network”.
    b. In the Subnetwork field, select the “Default” subnetwork.
11. Use the auto-generated service account for Your instance to interact with Google Cloud services and APIs.
12. Security options: Select or clear the following checkboxes:
    - Root access to the instance - Enabled
    - Nbconvert - Enabled
    - File downloading - Enabled
    - Terminal access - Enabled
13. In the System health section, provide the following:
    - In Reporting, select or clear the following checkboxes:
      - Report system health - Enabled
      - Report custom metrics to Cloud Monitoring - Disabled
      - Install Cloud Monitoring – Not installed
      - Report DNS status for required Google domains - Enabled
14. Click Create.

    Vertex AI Workbench creates an instance and automatically starts it. When the instance is ready to use, Vertex AI Workbench activates an Open JupyterLab link.

Unnesting the “Data”

1. Use the commands below to install the necessary libraries.

```
!pip3 install google-cloud-aiplatform --user
!pip3 install pyarrow==11.0.0 --user
!pip3 install --upgrade google-cloud-bigquery --user
!pip3 install --upgrade google-cloud-bigquery-storage --user
!pip3 install --upgrade google-cloud-storage --user
!pip install db-dtypes
```

2. Set the Project ID and Region by using the below commands

```
# Retrieve and set PROJECT_ID and REGION environment variables.
PROJECT_ID = !(gcloud config get-value core/project)
PROJECT_ID = PROJECT_ID[0]
BQ_LOCATION = 'US'
REGION = 'us-west1'
```

3. Create a Storage Bucket using the below commands

```
# Create a GC Storage bucket for later artifact storage
GCS_BUCKET = f"{PROJECT_ID}-k_capbucket"
!gsutil mb -l $REGION gs://$GCS_BUCKET
```

4. Create a dataset using the following commands

```
#bq --location=us mk --dataset <project_id>:my_dataset
BQ_DATASET = f"{PROJECT_ID}:p_bq"
!bq mk --location={BQ_LOCATION} --dataset {BQ_DATASET}
```

5. Import Pandas as PD
6. From SKLearn Model selection, import Train_test_split
7. From SKLearn Metrics, import accuracy score classification report.
8. From Google.Cloud.BigQuery, import client, queryjob configuration.
9. Create an object for the BigQuery Client.
10. Initialize the VertexAI using the below command

```
vertexai.init(project=PROJECT_ID, location=REGION,
staging_bucket=f"gs://{GCS_BUCKET}")
```

11. Using the BIGQuery, Run a query to see the data for the month of March 2017 as that is the month which we selected to work on.
12. Use the **dot notations** to access the child columns. See the example below

```
#standardsql
select
  visitId
  , totals.hits
  , totals.pageviews
  , totals.timeOnScreen
```

```
  , trafficSource.campaign
  , trafficSource.isTrueDirect
  , trafficSource.adwordsClickInfo.adGroupId
  , trafficSource.adwordsClickInfo.campaignId
from test.ga_sessions_20170801
```

You can chain your dot notation to query deeper field values, for example trafficSource.adwordsClickInfo.adGroupId.

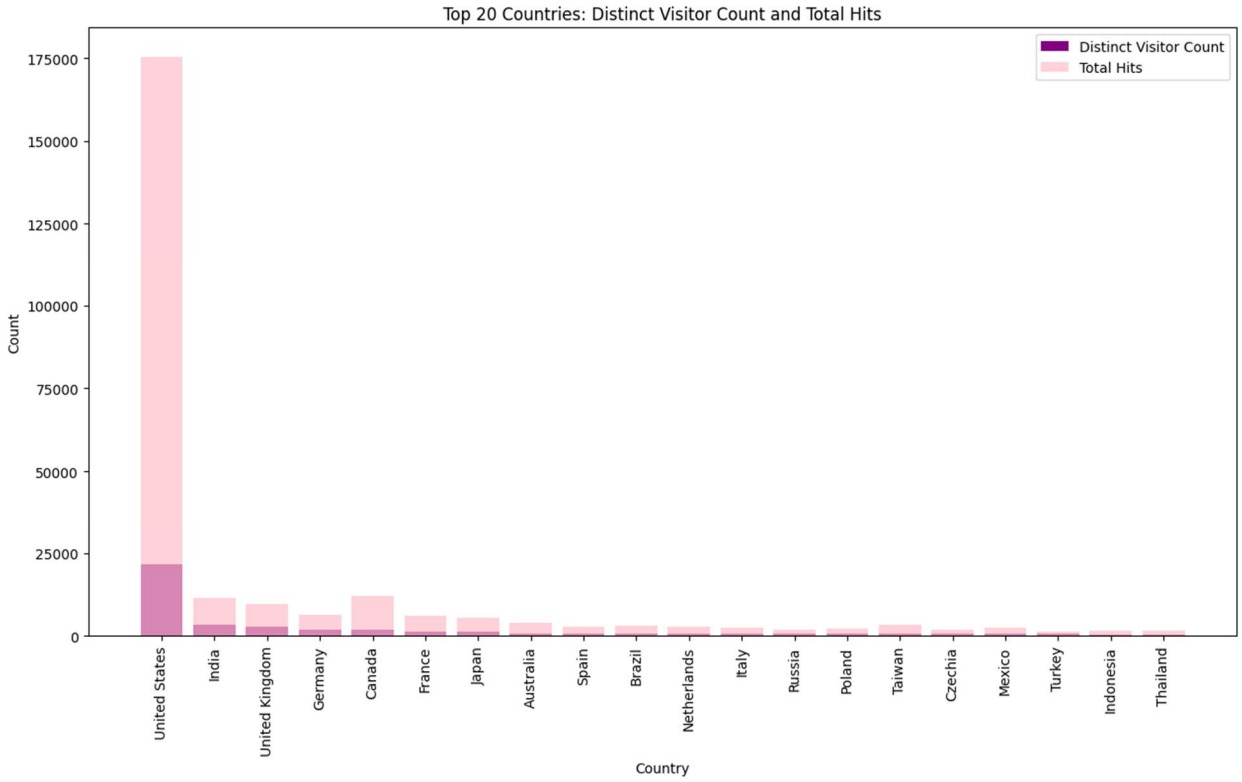
- Use UNNEST to flatten the field into a table where each row is an element of the array
- Join the flattened table back to the base table.

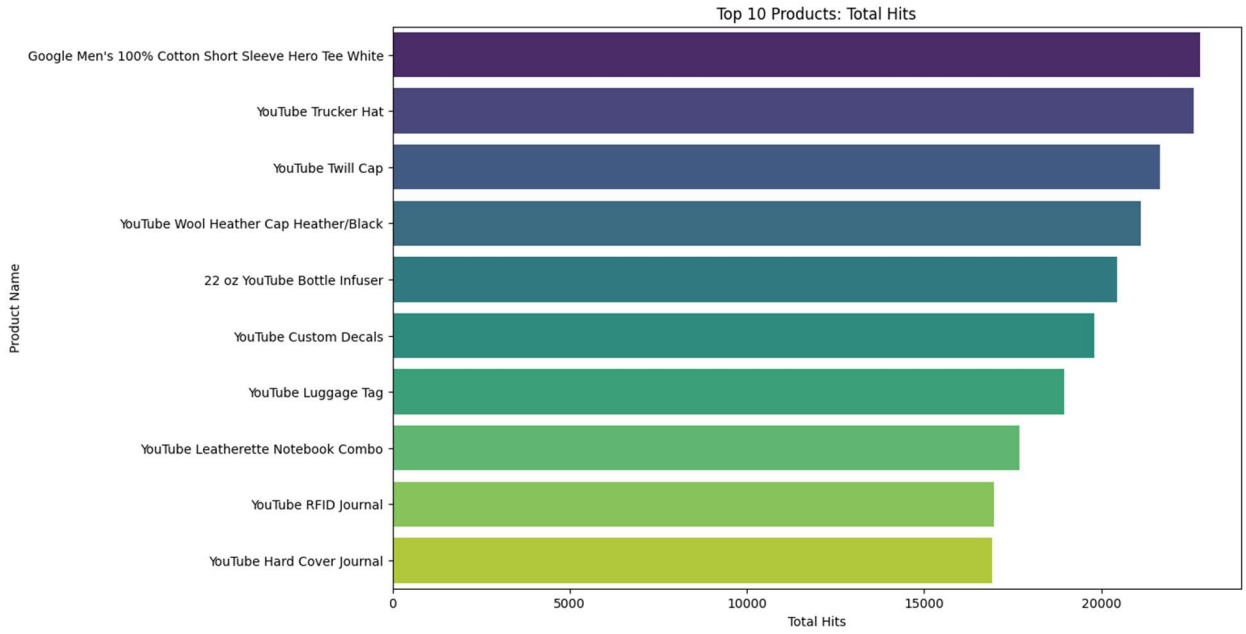
Sample code:

```
#standardsql
select
 visitId
 , h.hitNumber
 , h.time
 , h.hour
 , h.isInteraction
 , h.isEntrance
 , h.isExit
 , h.referer
 , h.page.pagePath
from test.ga_sessions_20170801
left join unnest(hits) as h -- No need to specify join key
```

Analyzing the "Data"

1. Go through the Schema keeping in mind the goal of "Add to cart"
2. Select the features from the dataset like hits.hitnumber, hits.ecommerceaction.action_type etc.
3. Clean the data by removing the null columns and duplicate rows.
4. Change the column ecommerce.action_type (Target) to value 1 (add to cart) and 0 (rest) options
5. Create three feature groups named "Numerical" , "Boolean" and "Categorical"
6. Remove the Target Column present in any of the above feature groups.
7. Use the describe function for each of the above feature groups in Pandas Library to get the number of non-null values and maximum value etc.
8. Run two queries to find the relationship between different columns
9. Using Matlabplotlib, plot the graphs for the queries.





10. Drop the Target variable from the dataset
11. Create another variable for the Tagetset
12. Split the data into “Test” and “Train” sets
13. Using “onehotencoder”, encode the categorical variables
14. Create a Dataframe named “df_new” from it
15. Create a Table named “Data” in the bucket with few selected “Important” features such as device category, operating system etc.

Running a Model

1. Select a few features such as pageviews, timeonsite, browser, country, city, hitnumber from df_new
2. Create or replace view with these features
3. Specify the test evaluation and train dataset using farm_fingerprint method in the query
4. Run xgboost model named “ML_Model” using BigQuery for these features
5. Evaluate this model using the BigQuery
6. Create or replace another view from the features selected from the “Data” table created above
7. Run a query to see the number of rows in Test, Train and Evaluation set
8. Run a query to see the columns and rows in the feature set
9. Run another model named “Preet_Model” using these features.
10. Evaluate the models
11. Run a query to see the confusion metrics
12. Run a query to see the roc_curve

13. Run a query to explain the model
14. Run a query to predict the values on the test dataset
15. Create a table named “action_type_predictions”
16. Print actual and predicted values

Deploy the Model

1. Create an exporting directory using the below command

```
BQ_MODEL = f"{BQ_DATASET}.{MODEL_NAME}"
BQ_MODEL_EXPORT_DIR = f"gs://{GCS_BUCKET}/{MODEL_NAME}"
```

2. Export the model to the storage bucket

```
!bq --location=$BQ_LOCATION extract \
--destination_format ML_XGBOOST_BOOSTER \
--model $BQ_MODEL \
$BQ_MODEL_EXPORT_DIR
```

3. Name the image URI as IMAGE_URI='us-docker.pkg.dev/vertex-ai/prediction/xgboost-cpu.1-4:latest'

4. Upload BQML model to Vertex AI from GC Storage with the following command

```
model = vertexai.Model.upload(
   display_name=MODEL_NAME,
   artifact_uri=BQ_MODEL_EXPORT_DIR,
   serving_container_image_uri=IMAGE_URI,
)
```

5. Deploy model to the endpoint using the following command

```
        endpoint = model.deploy(
   traffic_split={"0": 100},
   machine_type="e2-standard-2",
)

%%bigquery df_cat_features --project $PROJECT_ID
```

6. Get the Datatypes of each of the features in Preet_Model
7. Make a list of the categorical features
8. Import “Ordinalencoder” from SKlearn preprocessing
9. Get the list of the encoded features
10. Preprocess the SGBoost model

11. Run a command to see the prediction in the deployed model

Total Cost of the Project

1. $506.54 for 11 colleagues
2. For my VertexAI instance in Workbench, it is

   $0.158 hourly
   $115.20 monthly