Capstone Analysis Report
Luke Sands
York Solutions B2E

Introduction

Our project was to predict event “Add To Cart” probability for hits in the google analytics 360 sample dataset on BigQuery. I chose the month of february 2017 for my analysis, with the intention that the query would later be expanded to an entire year.

EDA

The data set was initially divided into smaller tables for more target investigation. Features for each table were examined for value counts, percentage of missing values, and other characteristics. A systematic process of elimination was used to remove most columns in the dataset due to high null value percentage, high cardinality, or a perceived lack of importance to the model.

Data Pre-processing

All categorical variables were cast to strings in the main SQL query and null values were set to “Unknown”. All null integer values were set to 0. This data had been previously investigated during EDA so the number of null values should have been low or 0. Categorical feature encoding was handled by the model automatically, and features that were encoded were later mapped to user input data for online predictions.

Feature Selection and Engineering

All features were selected initially for having a low or zero percentage of null values and a manageable cardinality. Three variables were concatenated to be used as a unique user-session-hit identifier for both query investigations and data splitting.

Model Choice

* Note: To bring in entire year of data, change above FROM statement #1 (current in notebook) to FROM statement #2 and select desired date range:

Model Data

Statement #1

```
FROM `bigquery-public-data.google_analytics_sample.ga_sessions_201702*`,
UNNEST(hits) AS hits LEFT JOIN UNNEST(hits.product) AS product
WHERE hits.eCommerceAction.action_type != '0'
```

Statement #2

```
FROM `bigquery-public-data.google_analytics_sample.ga_sessions_*`
UNNEST(hits) AS hits LEFT JOIN UNNEST(hits.product) AS product
WHERE hits.eCommerceAction.action_type != '0'
AND
WHERE _TABLE_SUFFIX BETWEEN '20160801' AND '20170831'
```

MODEL_TYPE="BOOSTED_TREE_CLASSIFIER",
INPUT_LABEL_COLS=["Target"],
AUTO_CLASS_WEIGHTS = True,
ENABLE_GLOBAL_EXPLAIN=True,

Model Evaluation Metrics: ROC AUC curve, Recall, Precision

Model Tuning

Hyperparameter space
num_trials=8, max_parallel_trials=4,
HPARAM_TUNING_OBJECTIVES=["roc_auc"], EARLY_STOP=True,
LEARN_RATE=HPARAM_RANGE(0.01, 0.1),
MIN_TREE_CHILD_WEIGHT = HPARAM_RANGE(0, 5),
COLSAMPLE_BYTREE = HPARAM_RANGE(0.7, 1.0),
MAX_TREE_DEPTH=HPARAM_CANDIDATES([4,5,6,7])

Deployment Documentation

1). Set environment variables, create a storage bucket, and create a dataset:

Environment variables:

```
PROJECT_ID = !(gcloud config get-value core/project)
PROJECT_ID = PROJECT_ID[0]
BQ_LOCATION = 'US'
REGION = 'us-west3'
```

Storage bucket:

```
GCS_BUCKET = f"{PROJECT_ID}-lukes_capstone"
!gsutil mb -l $REGION gs://$GCS_BUCKET
```

Dataset:

```
BQ_DATASET = f"{PROJECT_ID}:luke_cap_data"
!bq mk --location={BQ_LOCATION} --dataset {BQ_DATASET}
```

2). Create variables for model name, big query model name, and big query export path to export directory to previously created storage bucket:

```
MODEL_NAME = "Luke_cap"
BQ_MODEL = BQ_DATASET.MODEL_NAME
BQ_EXPORT_DIR = f”g://GCS_BUCKET/MODEL_NAME
```

3). Create location and export model to storage bucket call above-assigned variables:

```
!bq –location=$BQ_location extract \
–destination_format ML_XGBOOST_BOOSTER \
–model $BQ_MODEL \
$BQ_MODEL_EXPORT_DIR
```

4). Create docker container with most recent pre-built XGBOOST prediction configuration:

```
IMAGE_URI='us-docker.pkg.dev/vertex-ai/prediction/xgboost-cpu.1-4:latest'
```

5). Send model resource from cloud storage to Vertex AI via docker image container:

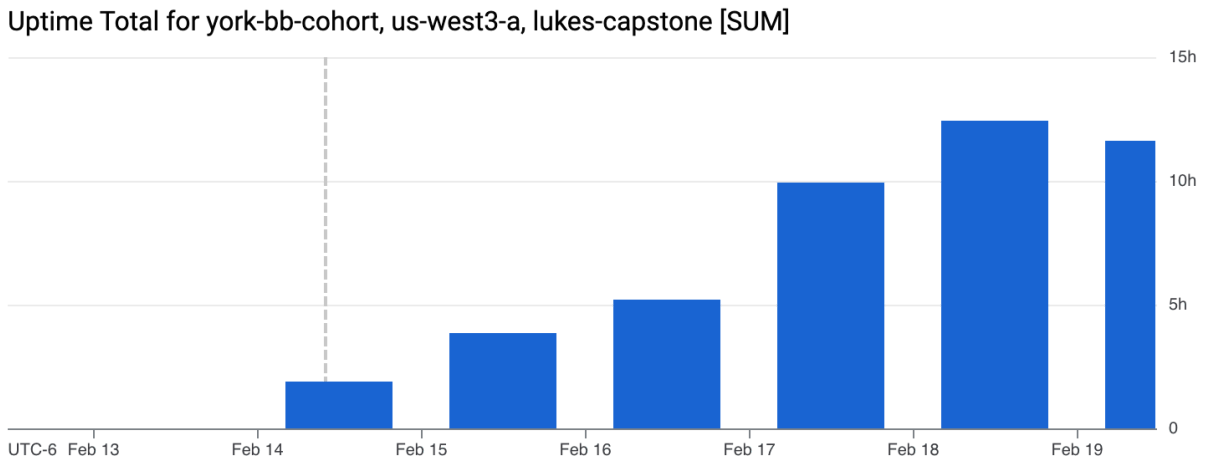
```
model = vertexai.Model.upload(
   display_name=MODEL_NAME,
   artifact_uri=BQ_MODEL_EXPORT_DIR,
   serving_container_image_uri=IMAGE_URI,
)
```

6). Create Endpoint resource to which model resource is deployed

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

Project Cost Report

Cost is difficult to calculate because I deleted my instance initially created and then created a new one in a different zone. Below is a summary of my instance uptime and the time charge it incurred. Further below is a cost projection of the entire project's cost consumption, but perhaps inaccurate due to the deletion and re-creation of my instance. Since I deleted my instance after 2 days, I will conservatively double the estimate base on the total below, which is $47.50: $47.50 x 2 = $95.00. Adding in the cpu uptime cost of $9.66 I project a total of $104.66 as my best estimate to bring this project to deployment.



Visual for total cpu instance uptime above, same numbers in chart below.

Day	Month	Date	Year	Seconds	Hours
Thu	Feb	15	2024	6839.017391	1.899727053
Fri	Feb	16	2024	13948.49106	3.874580851
Sat	Feb	17	2024	18824.28064	5.228966845
Sun	Feb	18	2024	35744.29042	9.928969561
Mon	Feb	19	2024	44851.78022	12.45882784
Tue	Feb	20	2024	41880.52825	11.63348007
				162088.388	45.02455222
				Total Seconds	Total Hours
				Total Cost	$9.66
				CPU rate	$0.19
				VAI workbench	$0.03

CPU Instance rate = 0.$19 / hr + Vertex AI workbench rate = $0.03 / hr
Overall uptime ~45 hours

Overall Project Cost Estimate: