Using AutoMl tools on Google Cloud Platform

Submitted By:
Surabhi Govil

Part 1: Using the AI Platform (Unified) create and train a model

Dataset:

- Used Rain in Australia dataset publicly available on Kaggle

Steps involved in training a AutoML model on GCP:

- Create a bucket and data to it.
- Create a new dataset on AI Platform (Unified) and upload data from cloud bucket
- Train the model
- Deploy the model to an endpoint and test it.

Evaluation:

EVALUATE | DEPLOY AND TEST | BATCH PREDICTIONS | MODEL PROPERTIES

Filter labels

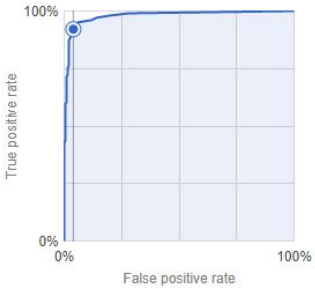
Confidence threshold 0.5

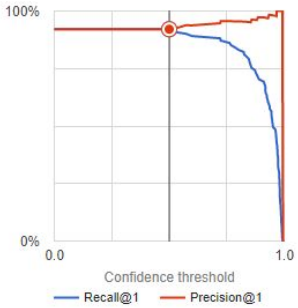
All labels	0
NA	1
No	0.9835
Yes	0.79633

All labels

PR AUC	0.966
ROC AUC	0.981
Log loss	0.217
F1 score	0.9207921
Precision	92.1%
Recall	92.1%
Created	20 Feb 2021, 13:43:46

Use the slider to see which confidence threshold works best for your model on the precision-recall trade-off curve. Learn more about these metrics and graphs





Test Results:

Test your model PREVIEW

Feature column name	Type	Required or optional	Value	Local feature importance
Cloud3pm	Text	Required	NA	0
Cloud9am	Text	Required	NA	0
Date	Text	Required	2010-04-21	0
Evaporation	Text	Required	NA	0

Predict label

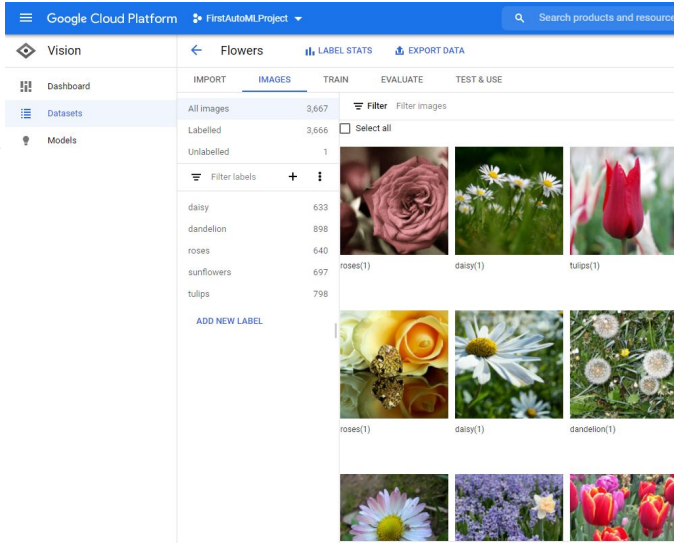
Prediction result

No

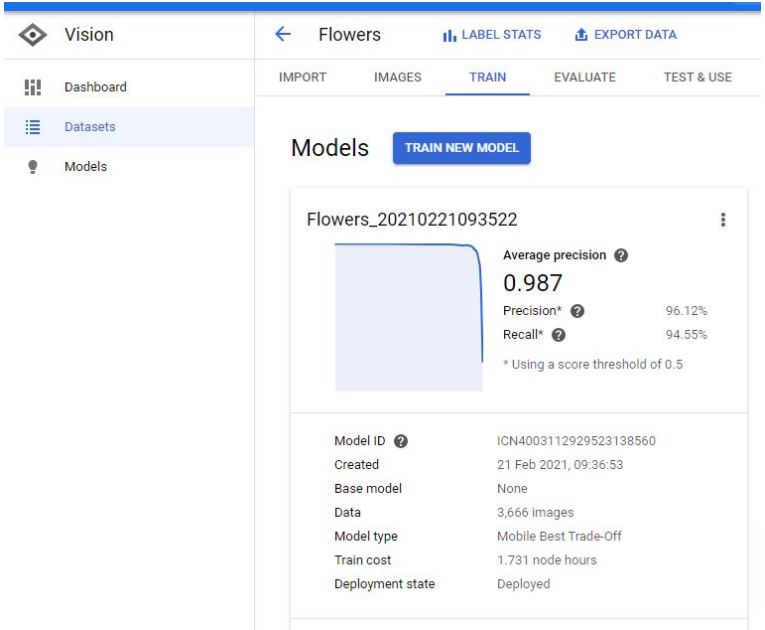
Confidence score: 0.9425028562545776

Part 2(a): Create an AutoMl model and deploy to edge device

- AutoMl Vision Edge and ML Kit are used here to develop and deploy model to iOS device
- Used the flowers dataset for classification.
- Steps:
  - Create a project on Firebase.
  - Add an iOS app to the project and download the .plist file for the app.
  - The .plist file contains properties and configuration for the app.
- Using the downloaded data folder create a dataset in AutoMl Vision.



Part 2(a): Model Training Results



Part 2(a): Model Evaluation

Model
Flowers_20210221093522

Confidence threshold 0.5

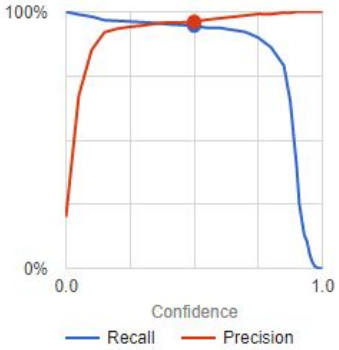
Filter labels

All labels	0.98728
daisy	0.99733
dandelion	0.99702
roses	0.97076
sunflowers	0.98309
tulips	0.99104

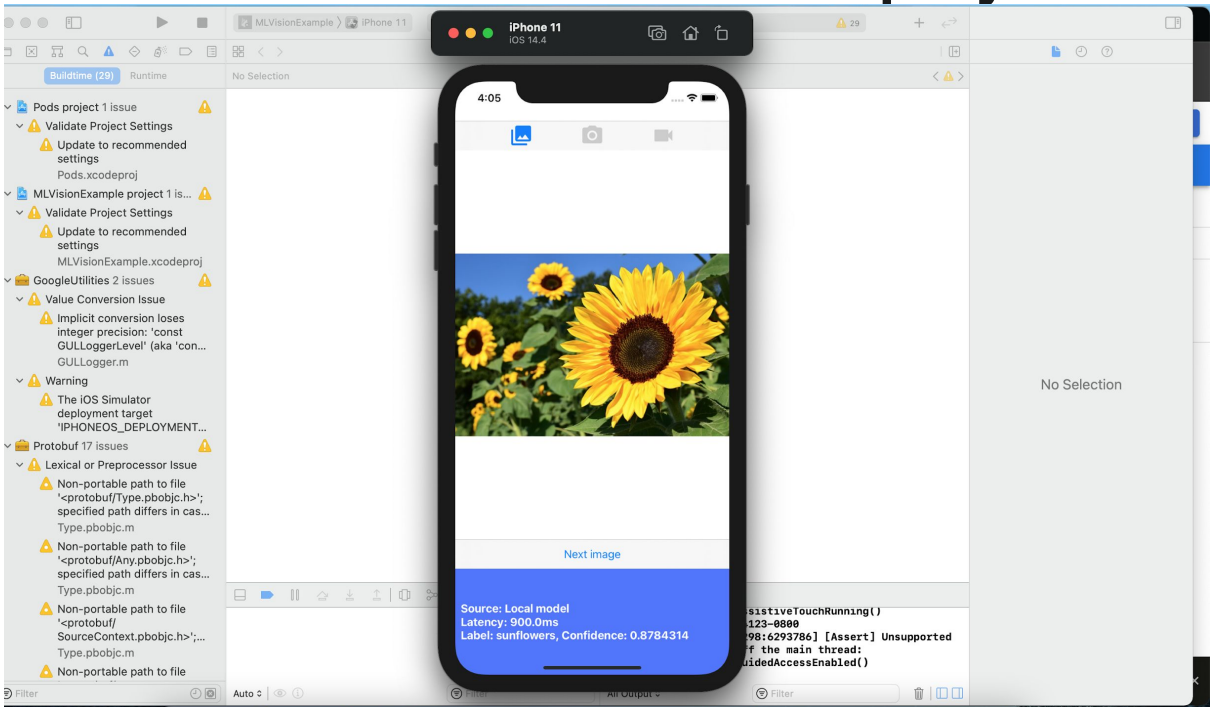
All labels

Total images	3,299
Test items	367
Precision	96.12%
Recall	94.55%

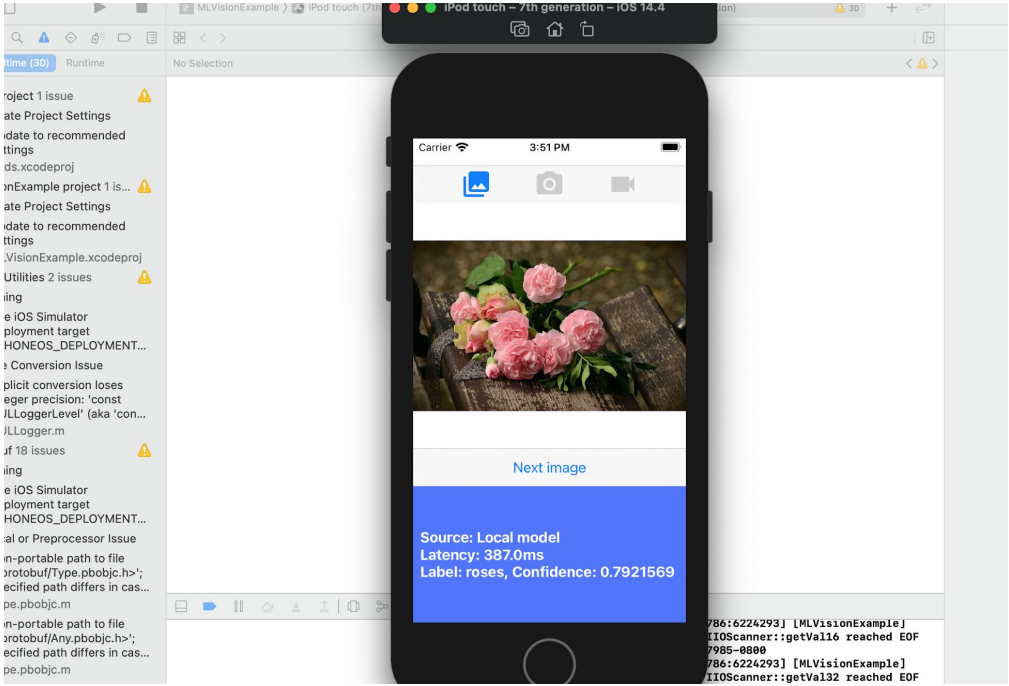
Use the slider to see which confidence threshold works best for your model on the precision-recall tradeoff curve.
Learn more about these metrics and graphs.



Part 2(a): AutoMl trained Model Deploy in iOS app

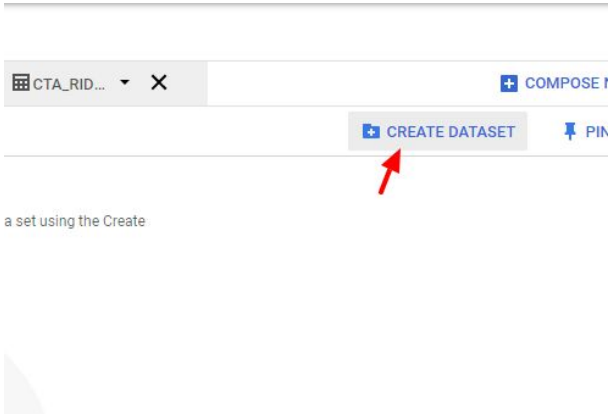


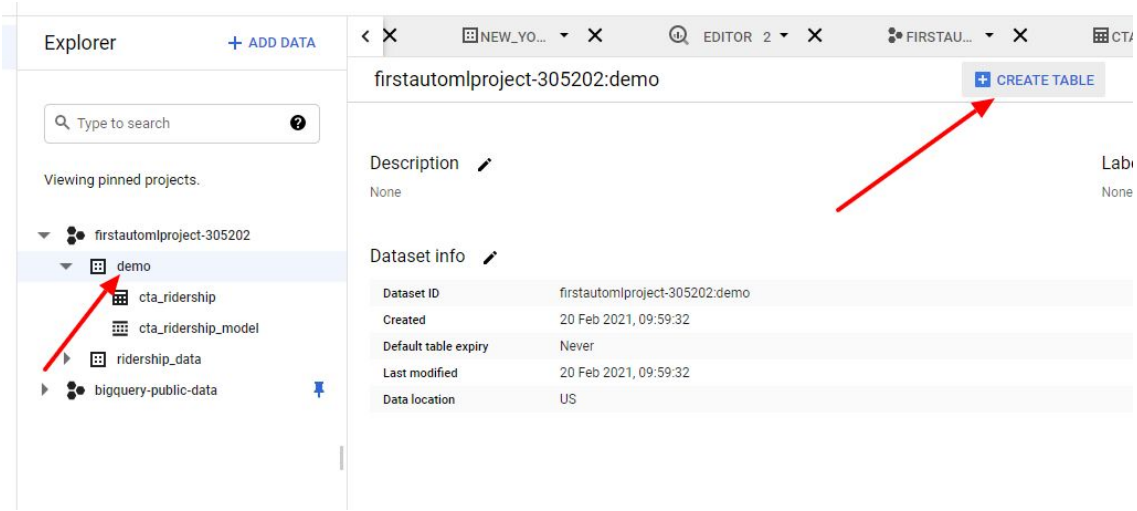
Part 2(a): Sample app Model Deploy



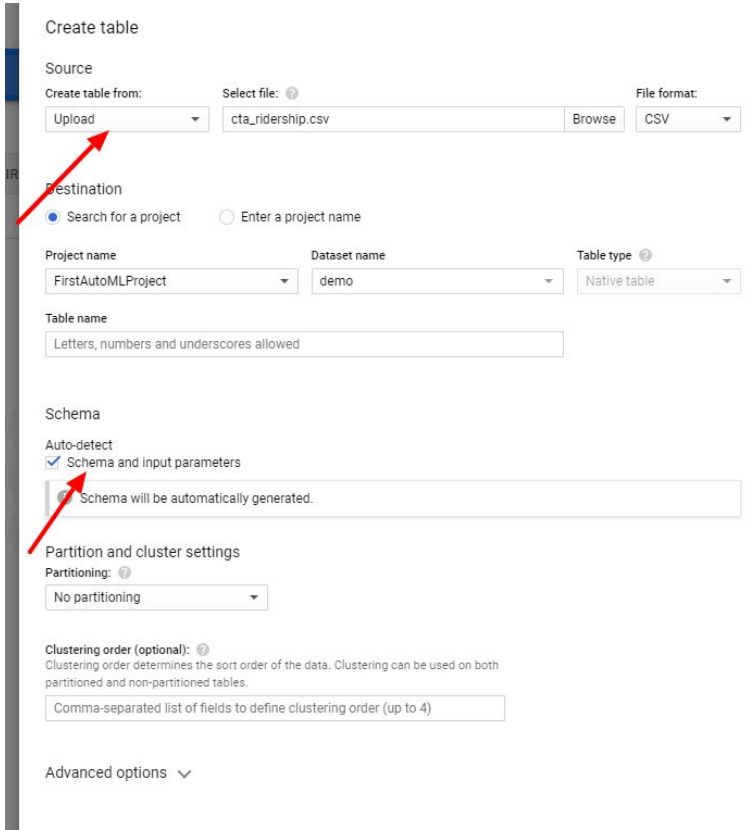
Part 2(b): Using BigQuery API on GCP:

- After enabling the Bigquery API we create a dataset.
- In the dataset created above we add a table.



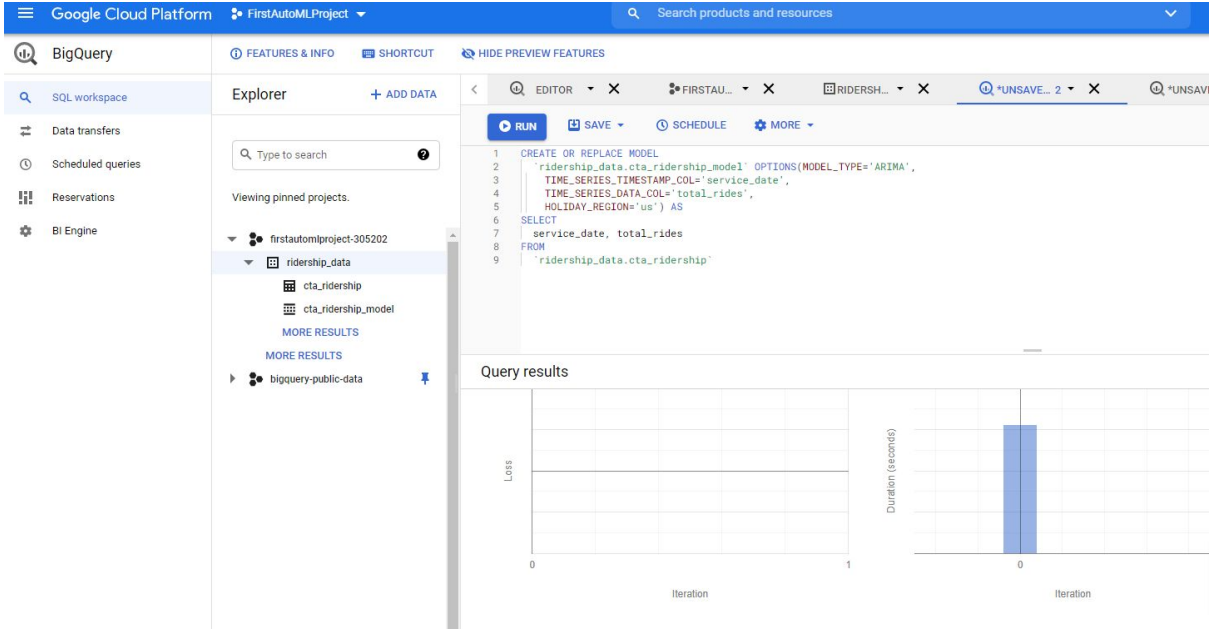


- Select the dataset from last step and click create table from the last step.
- Here we create table from a csv.
- We also let the schema be replicated from csv

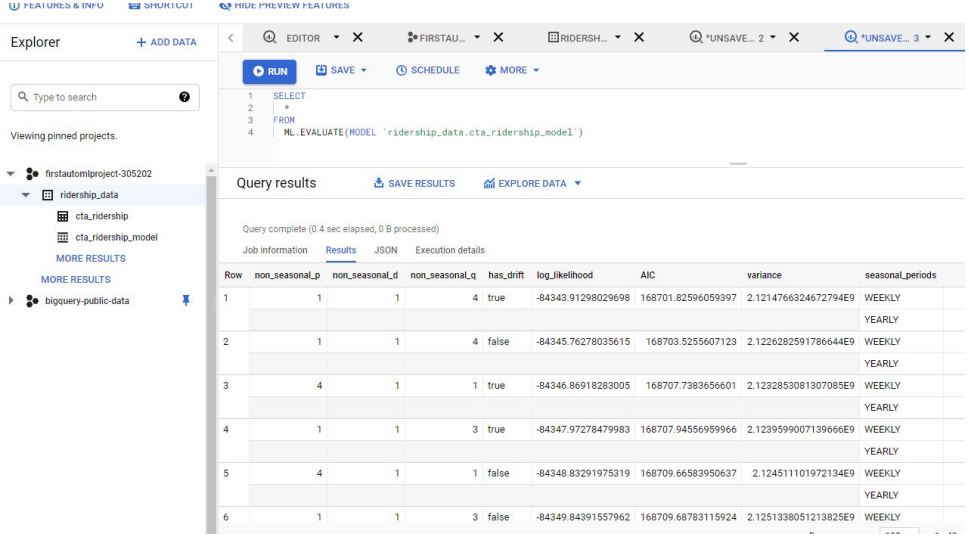


Part 2(b): Create model using BigQuery API on GCP:

- Dataset: Ride share data.
- Creating a model using Big Query API



Part 2(b): Evaluate model created using Bigquery on GCP:



Row	non_seasonal_p	non_seasonal_d	non_seasonal_q	has_drift	log_likelihood	AIC	variance	seasonal_periods
1	1	1	4	true	-84343.91298029698	168701.82596059397	2.1214766324672794E9	WEEKLY
								YEARLY
2	1	1	4	false	-84345.76278035615	168703.5255607123	2.1226282591786644E9	WEEKLY
								YEARLY
3	4	1	1	true	-84346.86918283005	168707.7383656601	2.1232853081307085E9	WEEKLY
								YEARLY
4	1	1	3	true	-84347.97278479983	168707.94556959966	2.1239599007139666E9	WEEKLY
								YEARLY
5	4	1	1	false	-84348.83291975319	168709.66583950637	2.124511101972134E9	WEEKLY
								YEARLY
6	1	1	3	false	-84349.84391557962	168709.68783115924	2.1251338051213825E9	WEEKLY

Part 2(b): Forecast on model using Bigquery

Explorer + ADD DATA

Type to search

Viewing pinned projects.

- firstautomlproject-305202
  - ridership_data
    - cta_ridership
    - cta_ridership_model
    - MORE RESULTS
  - MORE RESULTS
- bigquery-public-data

RIDERSH... | *UNSAVE... 2 | *UNSAVE... 3

RUN | SAVE | SCHEDULE | MORE

```
SELECT
  *
FROM
  ML.FORECAST(MODEL `ridership_data.cta_ridership_model`,
    STRUCT(7 AS horizon))
```

Query results | SAVE RESULTS | EXPLORE DATA

Query complete (0.3 sec elapsed, 23.4 KB processed)

Job information | Results | JSON | Execution details

Row	forecast_timestamp	forecast_value	standard_error	confidence_level	pre
1	2020-01-01 00:00:00 UTC	662436.4424369269	46059.49014554253	0.95	
2	2020-01-02 00:00:00 UTC	1029641.4669424891	46276.328347693256	0.95	
3	2020-01-03 00:00:00 UTC	1201660.2034356925	47233.43871922012	0.95	
4	2020-01-04 00:00:00 UTC	651095.9776391207	48157.99332862347	0.95	
5	2020-01-05 00:00:00 UTC	467394.91846646497	48621.50963880497	0.95	

Part 2(b): Exploring New York 311 service requests data:

NSAVE... 3 | CTA_RID... | *UNSAVE... 4 | BIGQUER... | *UNSAVE... 5 | *UNSAVE... 6 | COMPOSE NEW QUERY

RUN | SAVE | SCHEDULE | MORE

This query will process 11.8 GiB when run.

```
1  SELECT * FROM `bigquery-public-data.new_york_311.311_service_requests` LIMIT 5
```

Query results | SAVE RESULTS | EXPLORE DATA

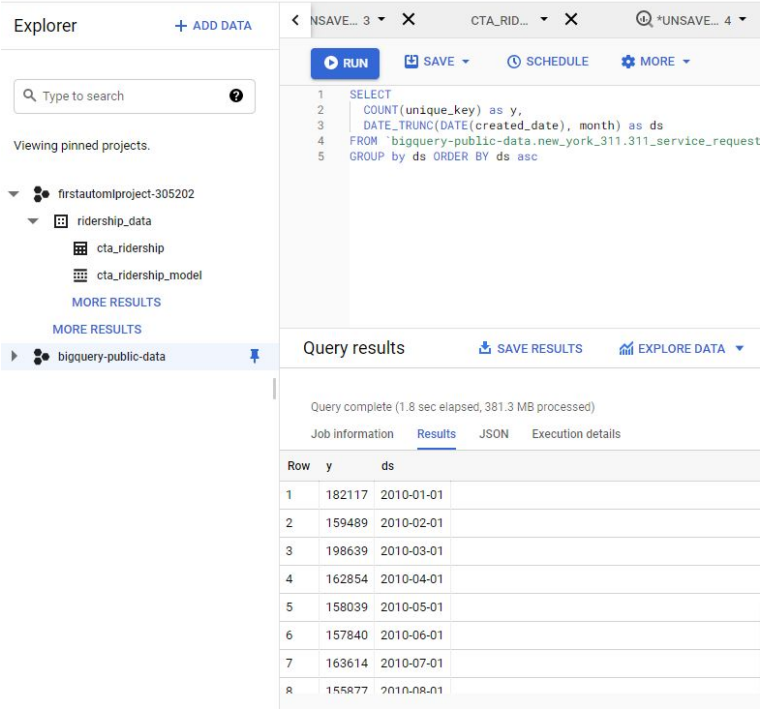
Query complete (0.9 sec elapsed, 11.8 GB processed)

Job information | Results | JSON | Execution details

Row	unique_key	created_date	closed_date	agency	agency_name	complaint_type	descriptor	location_type	incident_zip	incident_address	stree
1	16105756	2010-02-28 21:48:00 UTC	2010-02-23 02:03:00 UTC	DOT	Department of Transportation	Street Light Condition	Glassware Broken	null	10455	null	null
2	16113406	2010-03-01 16:42:00 UTC	2010-03-02 12:00:00 UTC	DSNY	BCC - Bronx	Other Enforcement	E10 Obstruction (Street/Sidewalk)	Sidewalk	10455	null	null
3	16115797	2010-03-01 10:06:00 UTC	2010-03-04 12:00:00 UTC	DSNY	A - Bronx	Snow	E9 Snow / Icy Sidewalk	Sidewalk	10454	null	null
4	16115800	2010-02-28 15:57:00 UTC	2010-03-04 12:00:00 UTC	DSNY	A - Bronx	Snow	E9 Snow / Icy Sidewalk	Sidewalk	10454	null	null
5	16124382	2010-03-02 15:10:00 UTC	2010-03-03 12:00:00 UTC	DSNY	BCC - Bronx	Overflowing Litter Baskets	6 Overflowing Litter Baskets	Sidewalk	10451	null	null

Exploring 311 calls using Bigquery SQL:

- Visualizing the dataset for unique 311 calls.
- The target column is unique values.



Using Bigquery reading data in jupyter notebook:

```
[8]: from google.cloud import bigquery as bq

sql = """
SELECT * FROM `bigquery-public-data.new_york_311.311_service_requests` LIMIT 5
"""

client = bq.Client(project='firstautomlproject-305202')
df = client.query(sql).to_dataframe()

df.head()
```

[8]:

	unique_key	created_date	closed_date	agency	agency_name	complaint_type	descriptor	location_type	incident_zip	incident_address	...	vehicle_type	ta
0	49815570	2021-02-18 10:27:46+00:00	2021-02-19 09:11:05+00:00	DHS	Department of Homeless Services	Homeless Person Assistance	None	N/A	10169	230 PARK AVENUE	...	None	
1	49818016	2021-02-18 15:36:07+00:00	2021-02-19 09:11:04+00:00	DHS	Department of Homeless Services	Homeless Person Assistance	None	N/A	10169	230 PARK AVENUE	...	None	
2	49818144	2021-02-18 09:21:27+00:00	2021-02-19 09:11:03+00:00	DHS	Department of Homeless Services	Homeless Person Assistance	None	N/A	10169	230 PARK AVENUE	...	None	
3	49819440	2021-02-19 15:06:58+00:00	NaT	DHS	Department of Homeless Services	Homeless Person Assistance	None	N/A	10019	712 5 AVENUE	...	None	
4	49818978	2021-02-18 13:00:13+00:00	2021-02-19 09:11:03+00:00	DHS	Department of Homeless Services	Homeless Person Assistance	None	N/A	10169	230 PARK AVENUE	...	None	

5 rows × 41 columns

FEATURES & INFO | SHORTCUT | HIDE PREVIEW FEATURES

Explorer

Type to search

Viewing pinned projects.

- firstautomlproj...
- bigquery-public...

EDITOR | NYC 311 ...

RUN | SAVE | SCHEDULE | MORE

```
SELECT
  extract(DAYOFWEEK
  FROM
    created_date) AS party_day,
  borough,
  COUNT(*) AS num_parties
FROM
  `bigquery-public-data.new_york.311_service_requests`
WHERE
  LOWER(descriptor) LIKE '%party%'
GROUP BY
  party_day,
  borough
ORDER BY
  num_parties DESC
```

Query results | SAVE RESULTS | EXPLORE DATA

Query complete (0.0 sec elapsed, cached)

Job information | Results | JSON

Row	party_day	borough	num_parties
1	1	BROOKLYN	111114
2	7	BROOKLYN	110643
3	7	MANHATTAN	98680
4	1	MANHATTAN	88933
5	1	QUEENS	79136
6	1	BRONX	76331
7	7	BRONX	71942