

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

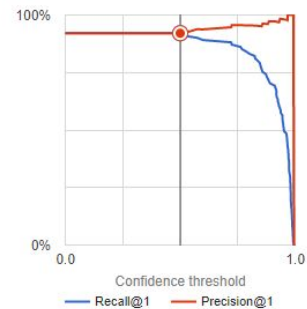
Yes

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	<input type="text" value="NA"/>	0
Cloud9am	Text	Required	<input type="text" value="NA"/>	0
Date	Text	Required	<input type="text" value="2010-04-21"/>	0
Evaporation	Text	Required	<input type="text" value="NA"/>	0

Predict label

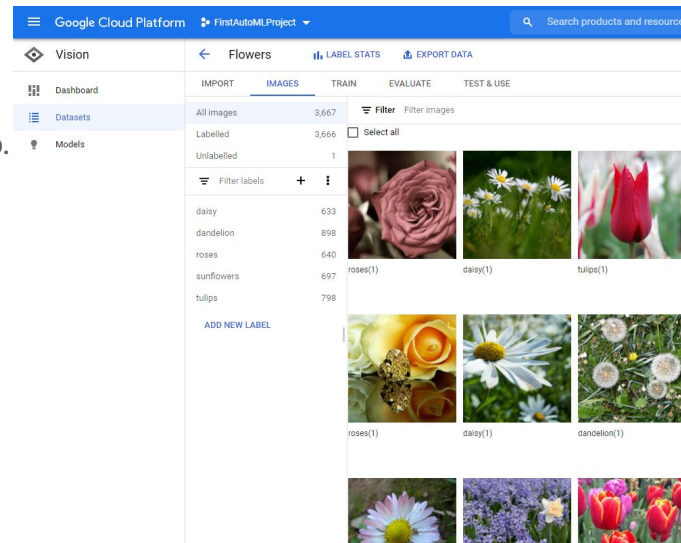
Prediction result

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


 Vision


 Dashboard

 Datasets

 Models

← Flowers

 LABEL STATS

 EXPORT DATA

IMPORT

IMAGES

TRAIN


EVALUATE


TEST & USE

Models


TRAIN NEW MODEL


Flowers_20210221093522




Average precision 

0.987

Precision*  96.12%

Recall*  94.55%

* Using a score threshold of 0.5

Model ID 

ICN4003112929523138560

Created

21 Feb 2021, 09:36:53

Base model

None

Data

3,666 images

Model type

Mobile Best Trade-Off

Train cost

1.731 node hours

Deployment state

Deployed

Part 2(a): Model Evaluation

Model
Flowers_20210221093522

Confidence threshold



0.5

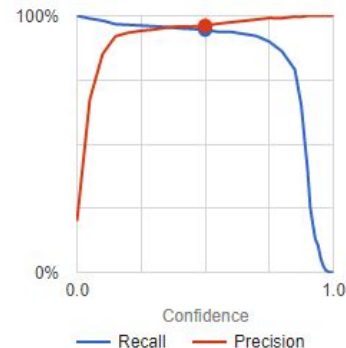
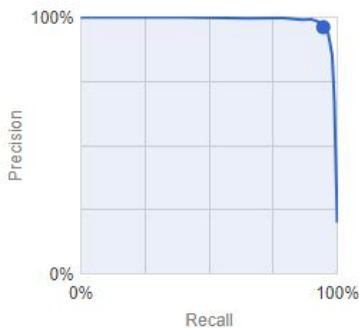
Filter labels

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

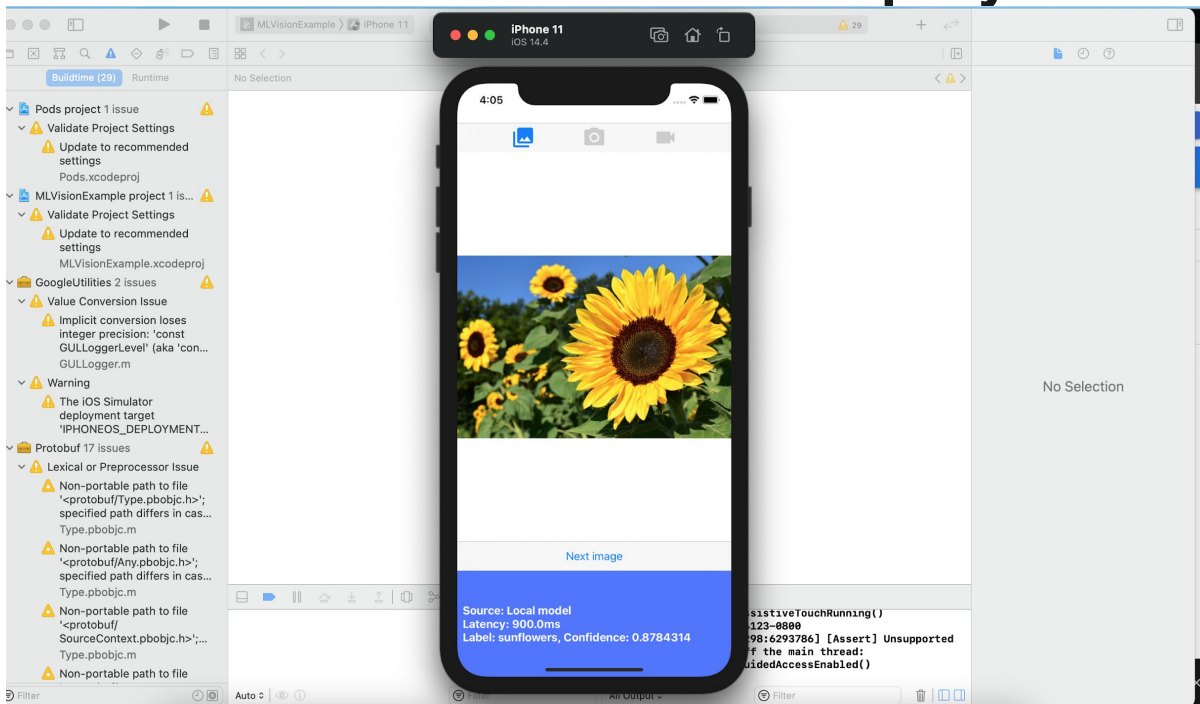
All labels

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

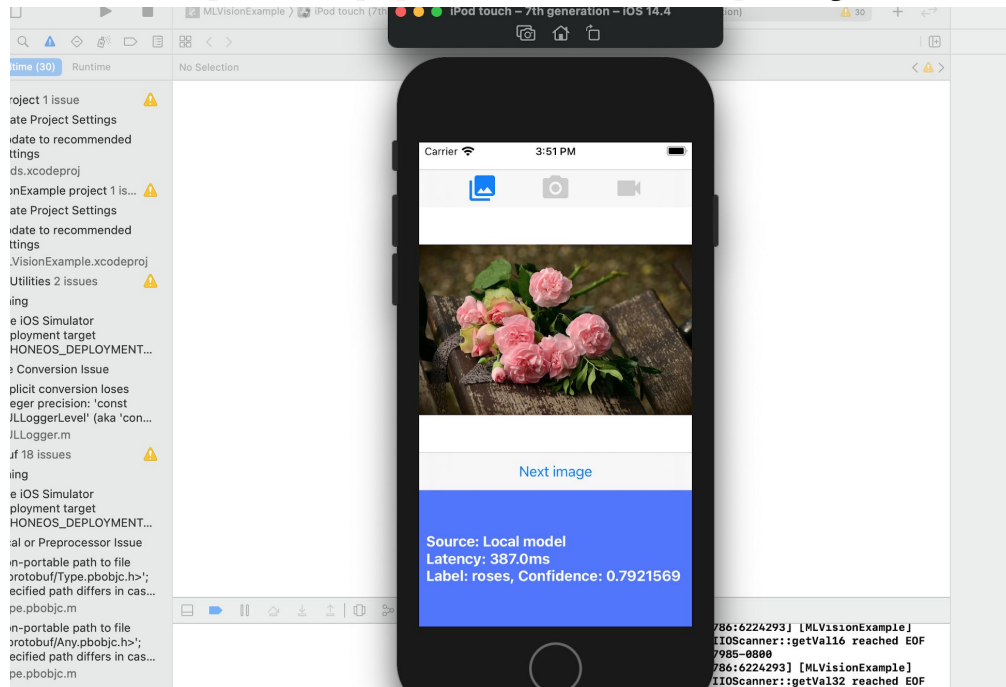
Use the slider to see which confidence threshold works best for your model on the precision-recall tradeoff curve.
[Learn more about these metrics and graphs.](#)



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.

The image displays three screenshots from the Google Cloud Platform BigQuery interface, illustrating the process of creating a dataset and then a table within it.

Left Screenshot: Shows the BigQuery console with a search bar containing "CTA_RID...". Below the search bar, the "CREATE DATASET" button is highlighted with a red arrow.

Middle Screenshot: Shows the "Explorer" view. The "firstautomlproject-305202" project is expanded, and the "demo" dataset is selected. The "demo" dataset is expanded, showing a list of tables: "cta_ridership", "cta_ridership_model", "ridership_data", and "bigquery-public-data". A red arrow points to the "cta_ridership" table.

Right Screenshot: Shows the "Dataset info" view for the "firstautomlproject-305202:demo" dataset. The "CREATE TABLE" button is highlighted with a red arrow.

Dataset Info Table:

Property	Value
Dataset ID	firstautomlproject-305202:demo
Created	20 Feb 2021, 09:59:32
Default table expiry	Never
Last modified	20 Feb 2021, 09:59:32
Data location	US

- 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

Create table

Source

Create table from: Upload Select file: ? File format: CSV

Destination

☒ Search for a project ☐ Enter a project name

Project name: FirstAutoMLProject Dataset name: demo Table type: Native table

Table name

Letters, numbers and underscores allowed

Schema

Auto-detect

☒ Schema and input parameters

Schema will be automatically generated.

Partition and cluster settings

Partitioning: No partitioning

Clustering order (optional): ?

Clustering order determines the sort order of the data. Clustering can be used on both partitioned and non-partitioned tables.

Comma-separated list of fields to define clustering order (up to 4)

Advanced options

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

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

The screenshot displays the Google Cloud Platform (GCP) console interface. The top navigation bar shows 'Google Cloud Platform' and 'FirstAutoMLProject'. The left sidebar contains navigation options: 'BigQuery', 'SQL workspace', 'Data transfers', 'Scheduled queries', 'Reservations', and 'BI Engine'. The main area is divided into two panes. The left pane, titled 'Explorer', shows a tree view of the 'firstautomlproject-305202' project, with 'ridership_data' expanded to show 'cta_ridership' and 'cta_ridership_model'. The right pane, titled 'EDITOR', shows a SQL query in the 'SQL workspace' tab. The query is a CREATE OR REPLACE MODEL statement for 'ridership_data.cta_ridership_model' using the 'ARIMA' model type, with 'service_date' as the time series timestamp column and 'total_rides' as the time series data column. The query is executed, and the 'Query results' section shows two charts: 'Loss' and 'Duration (seconds)'. The 'Loss' chart shows a single bar at iteration 0 with a value of approximately 0.5. The 'Duration (seconds)' chart shows a single bar at iteration 0 with a value of approximately 1.5.

```
1 CREATE OR REPLACE MODEL
2   `ridership_data.cta_ridership_model` OPTIONS(MODEL_TYPE='ARIMA',
3   TIME_SERIES_TIMESTAMP_COL='service_date',
4   TIME_SERIES_DATA_COL='total_rides',
5   HOLIDAY_REGION='us') AS
6 SELECT
7   service_date, total_rides
8 FROM
9   `ridership_data.cta_ridership`
```

Query results

Loss

Duration (seconds)

Iteration

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

The screenshot displays the Google Cloud Platform BigQuery console interface. On the left, the 'Explorer' pane shows a project named 'firstautomproject-305202' with a dataset 'ridership_data' containing tables 'cta_ridership' and 'cta_ridership_model'. The main editor area shows a SQL query: `SELECT * FROM ML.EVALUATE(MODEL `ridership_data.cta_ridership_model`)`. Below the query, the 'Query results' section shows the query is complete. The results are displayed in a table with columns: Row, non_seasonal_p, non_seasonal_d, non_seasonal_q, has_drift, log_likelihood, AIC, variance, and seasonal_periods. The table contains 6 rows of data, showing evaluation results for different seasonal periods (WEEKLY and YEARLY) and drift status (true and false).

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

The screenshot displays the Google Cloud BigQuery web interface. On the left, the 'Explorer' pane shows a project named 'firstautomlproject-305202' with a dataset 'ridership_data' containing tables 'cta_ridership' and 'cta_ridership_model'. The main editor shows a SQL query that uses the 'ML.FORECAST' function to generate a 7-day forecast from the 'ridership_data.cta_ridership_model' model. The query is executed, and the 'Query results' pane shows the output. The results indicate the query is complete, having processed 23.4 KB of data in 0.3 seconds. The results are displayed in a table with columns for 'Row', 'forecast_timestamp', 'forecast_value', 'standard_error', 'confidence_level', and 'pre'.

Explorer [+ ADD DATA](#)

Type to search ?

Viewing pinned projects.

- firstautomlproject-305202
 - ridership_data
 - cta_ridership
 - cta_ridership_model
 - [MORE RESULTS](#)
 - bigquery-public-data

[RUN](#) [SAVE](#) [SCHEDULE](#) [MORE](#)

```
1 SELECT
2   *
3 FROM
4   ML.FORECAST(MODEL `ridership_data.cta_ridership_model`,
5   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:

The screenshot displays the Google Cloud BigQuery interface. At the top, there's a toolbar with buttons for 'RUN', 'SAVE', 'SCHEDULE', and 'MORE'. A status message indicates 'This query will process 11.8 GiB when run.' Below the toolbar, the SQL query is shown: `1 SELECT * FROM `bigquery-public-data.new_york_311.311_service_requests` LIMIT 5`. The 'Query results' section shows 'Query complete (0.9 sec elapsed, 11.8 GB processed)'. Below this, there are tabs for 'Job information', 'Results' (selected), 'JSON', and 'Execution details'. The 'Results' tab displays a table with 12 columns: Row, unique_key, created_date, closed_date, agency, agency_name, complaint_type, descriptor, location_type, incident_zip, incident_address, and street. The table contains 5 rows of data.

Row	unique_key	created_date	closed_date	agency	agency_name	complaint_type	descriptor	location_type	incident_zip	incident_address	street
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	<i>null</i>	10455	<i>null</i>	<i>null</i>
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	<i>null</i>	<i>null</i>
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	<i>null</i>	<i>null</i>
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	<i>null</i>	<i>null</i>
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	<i>null</i>	<i>null</i>

Exploring 311 calls using Bigquery SQL:

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

The screenshot displays the Google Cloud BigQuery interface. On the left, the 'Explorer' pane shows a project named 'firstautomlproject-305202' with a dataset 'ridership_data' containing tables 'cta_ridership' and 'cta_ridership_model'. Below this, the 'bigquery-public-data' dataset is selected. The main editor shows a SQL query that counts unique keys by month. The 'Query results' pane at the bottom shows the query has completed, processing 381.3 MB in 1.8 seconds. A table of results is displayed with columns for row number, year, and date.

```
SELECT
  COUNT(unique_key) as y,
  DATE_TRUNC(DATE(created_date), month) as ds
FROM `bigquery-public-data.new_york_311.311_service_request`
GROUP BY ds ORDER BY ds asc
```

Query results

Query complete (1.8 sec elapsed, 381.3 MB processed)

Job information Results JSON Execution details

Row	y	ds
1	182117	2010-01-01
2	159489	2010-02-01
3	198639	2010-03-01
4	162854	2010-04-01
5	158039	2010-05-01
6	157840	2010-06-01
7	163614	2010-07-01
8	155877	2010-08-01

Using Bigquery reading data in jupyter notebook:

```
18]: 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()
```

```
18]:
```

	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

```
1 SELECT
2   extract(DAYOFWEEK
3   FROM
4     created_date) AS party_day,
5   borough,
6   COUNT(*) AS num_parties
7 FROM
8   `bigquery-public-data.new_york.311_service_requests`
9 WHERE
10  LOWER(descriptor) LIKE '%party%'
11 GROUP BY
12   party_day,
13   borough
14 ORDER BY
15   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