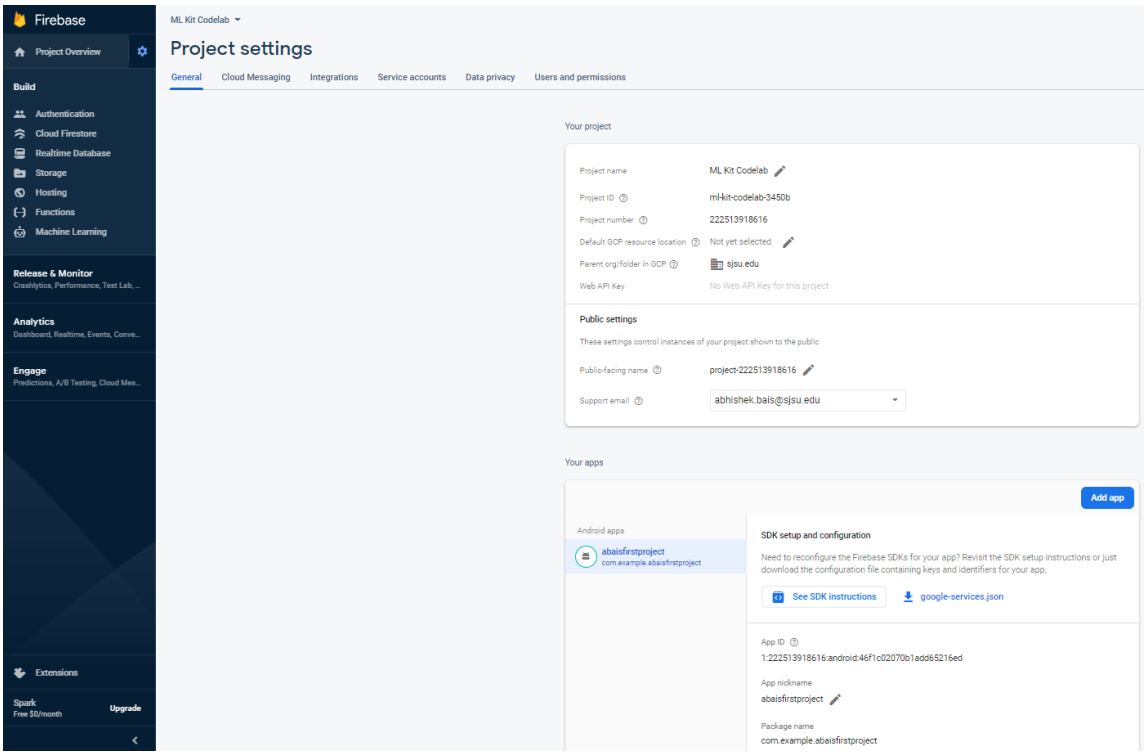
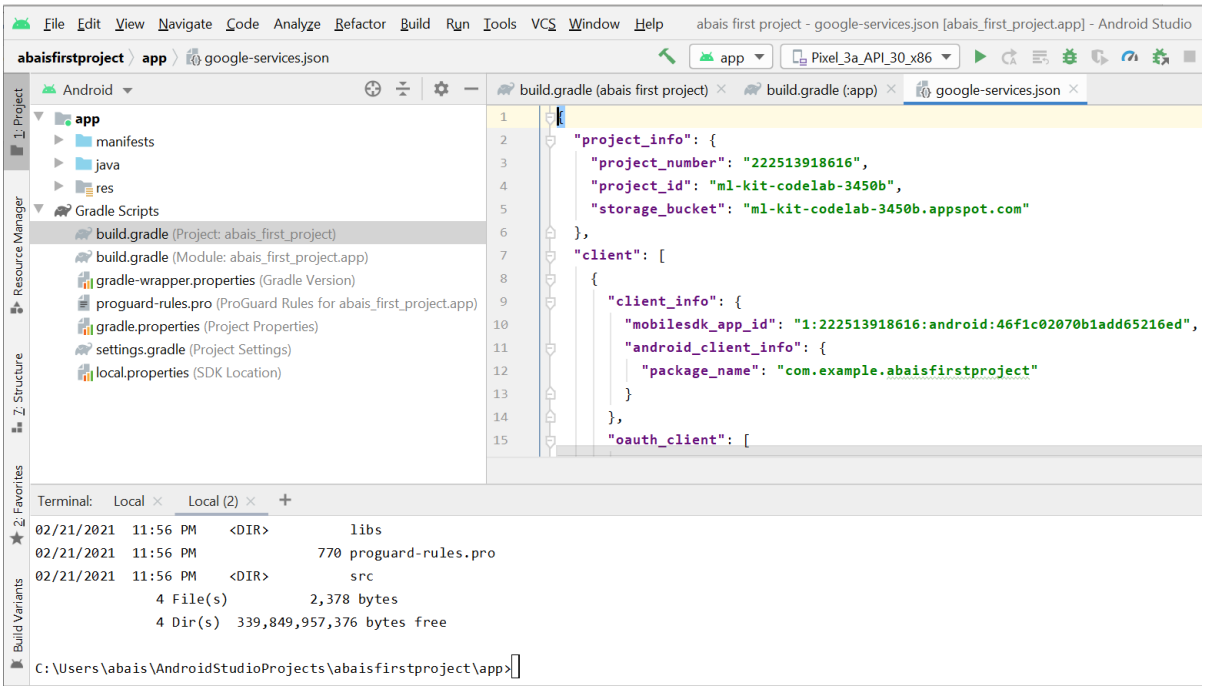
CMPE 258 – HW2 Abhishek Bais Part 2 Execute automl vision

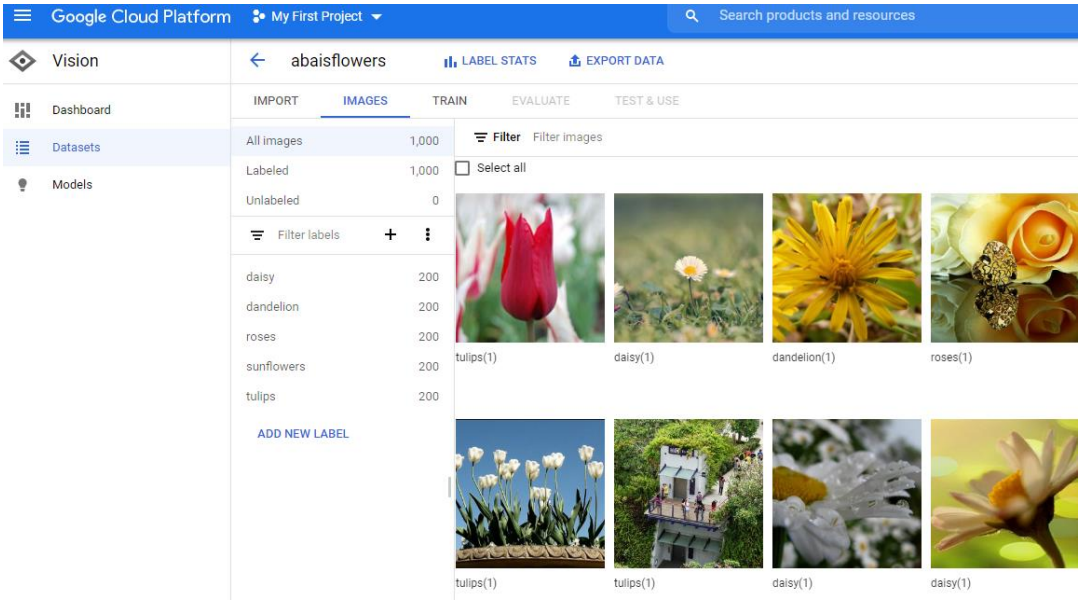
a) Firebase setup



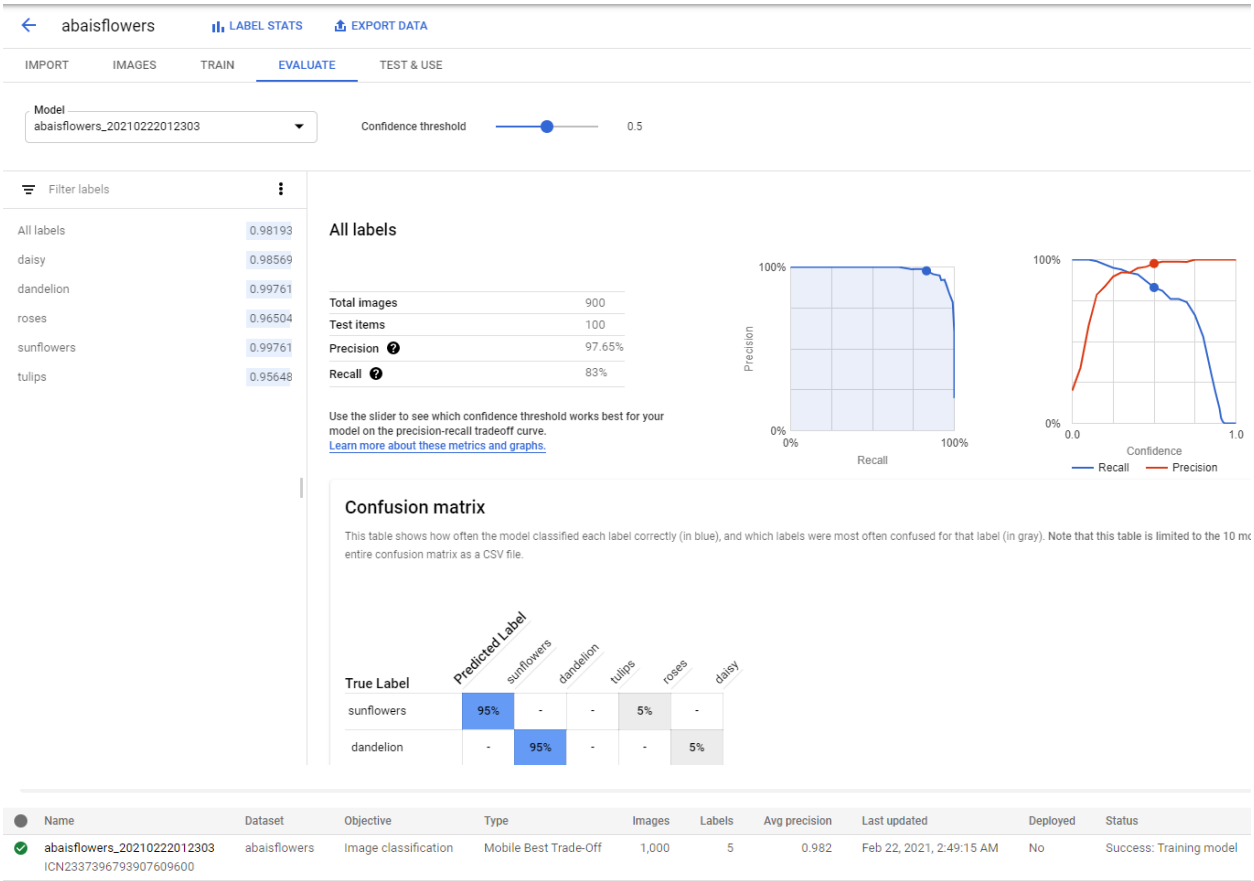
b) Copy over the google-services JSON file



c) From firebase console – train a model -> import `flower_photos.zip`



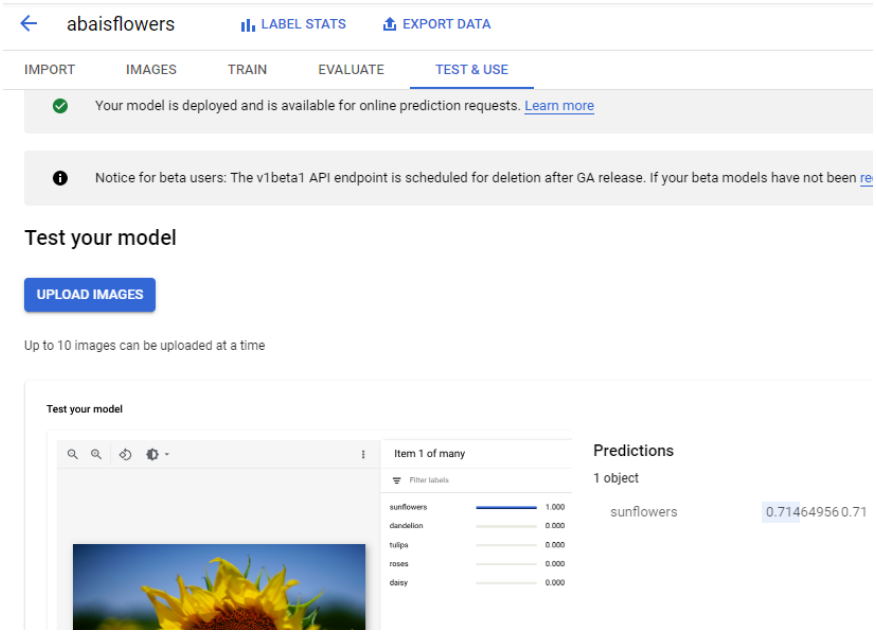
d) Evaluate model



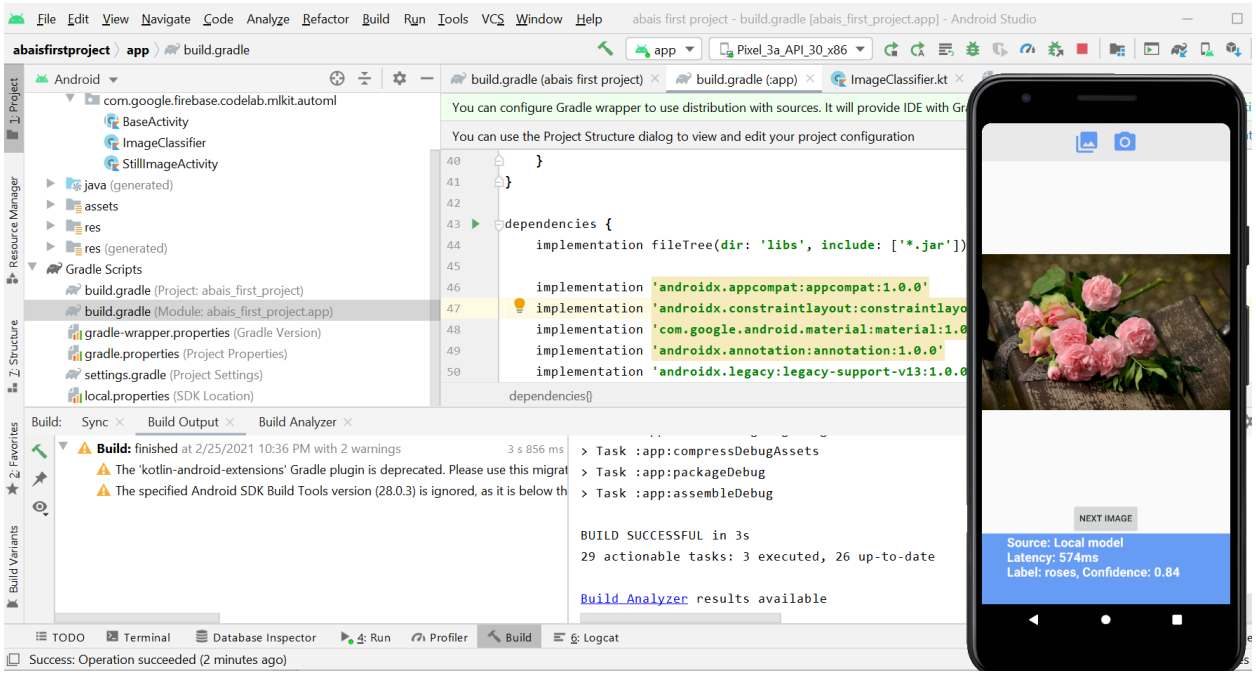
e) Deploy model

Name	Dataset	Objective	Type	Images	Labels	Avg precision	Last updated	Deployed	Status
abaisflowers_20210222012303 ICN2337396793907609600	abaisflowers	Image classification	Mobile Best Trade-Off	1,000	5	0.982	Feb 22, 2021, 9:15:36 AM	Yes 1 node	Success: Deploying model

f) Test model

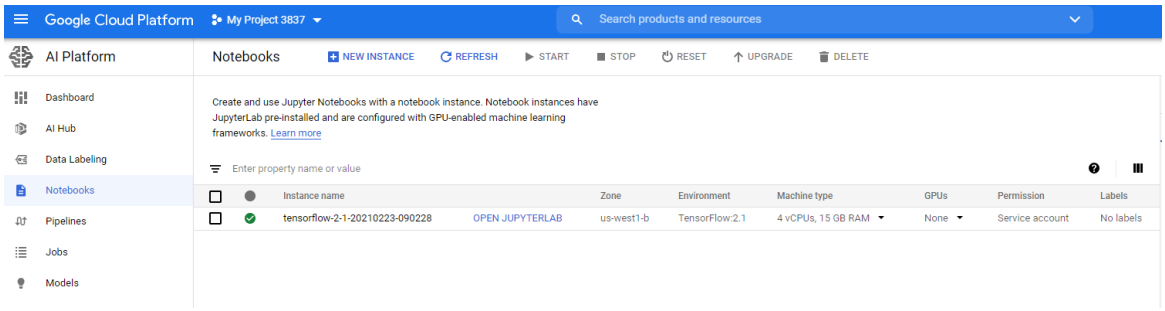


g) Add model to app, select image and check label and confidence

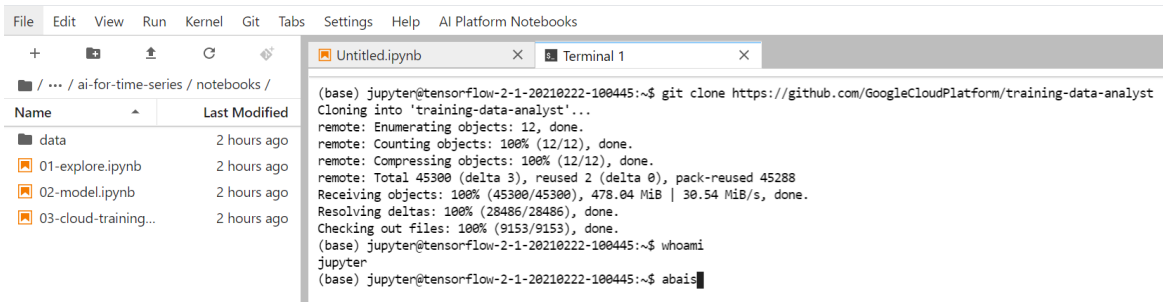


Part 2 Time Series forecasting

a) Create a notebook instance



b) Download lab materials



c) Explore and visualize data 01-explore.ipynb file screenshots

Create a project, load the data

In [10]:
```
# Enter your project and region. Then run the  cell to make sure the
# Cloud SDK uses the right project for all the commands in this notebook.

PROJECT = 'My First Project' # REPLACE WITH YOUR PROJECT NAME
REGION = 'us-central-1' # REPLACE WITH YOUR REGION e.g. us-central1

#Don't change the following command - this is to check if you have changed the project name above.
assert PROJECT != 'your-project-name', 'Don''t forget to change the project variables!'
```

In [11]:
```
target = 'total_rides' # The variable you are predicting
target_description = 'Total Rides' # A description of the target variable
features = {'day_type': 'Day Type'} # Weekday = W, Saturday = A, Sunday/Holiday = U
ts_col = 'service_date' # The name of the column with the date field

raw_data_file = 'https://data.cityofchicago.org/api/views/6iiy-9s97/rows.csv?accessType=DOWNLOAD'
processed_file = 'cta_ridership.csv' # Which file to save the results to
```

Load data

In [12]:
```
# Import CSV file

df = pd.read_csv(raw_data_file, index_col=[ts_col], parse_dates=[ts_col])
```

In [13]:
```
# Model data prior to 2020

df = df[df.index < '2020-01-01']
```

Explore the data

Explore data

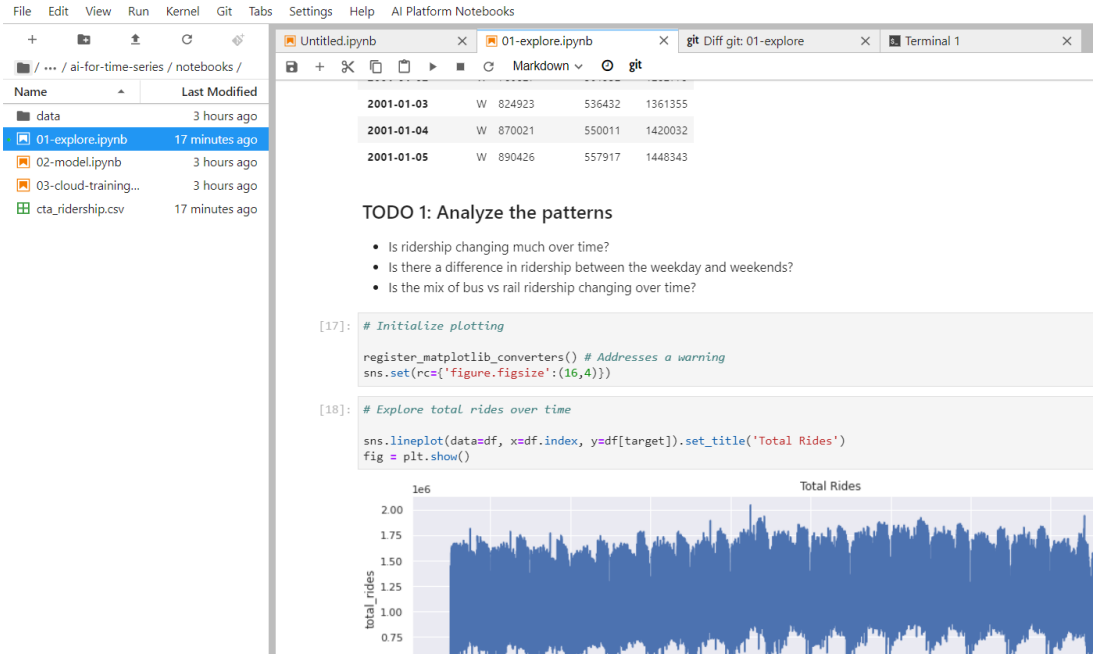
In [16]:
```
# Print the top 5 rows

df.head()
```

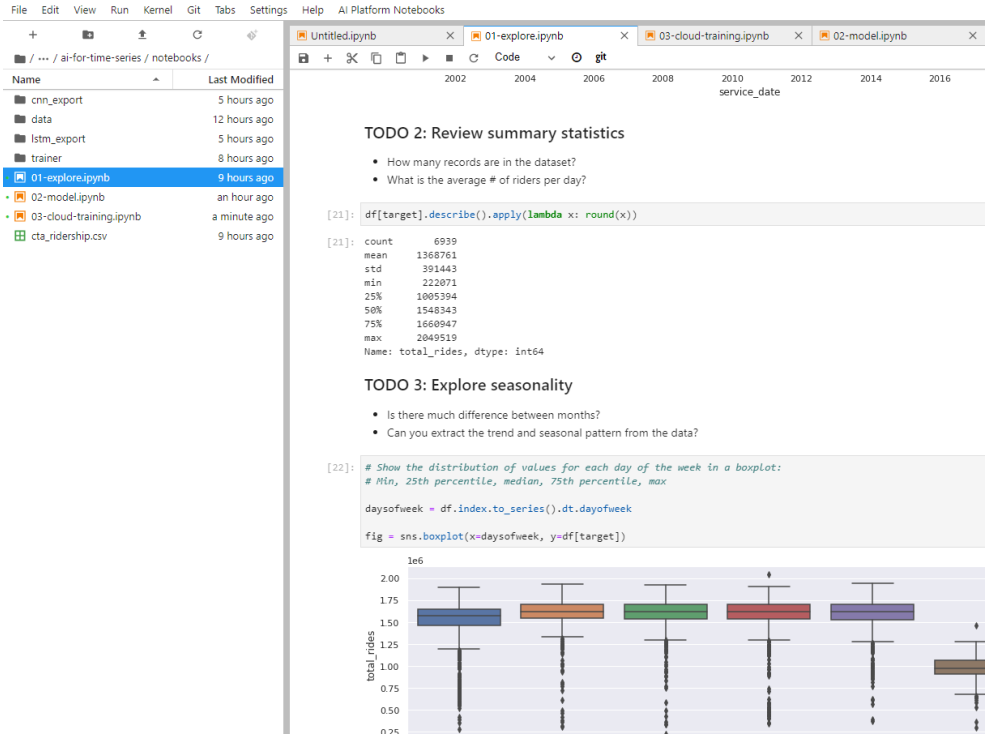
Out[16]:

	day_type	bus	rail_boardings	total_rides
service_date				
2001-01-01	U	297192	126455	423647
2001-01-02	W	780827	501952	1282779
2001-01-03	W	824923	536432	1361355
2001-01-04	W	870021	550011	1420032
2001-01-05	W	890426	557917	1448343

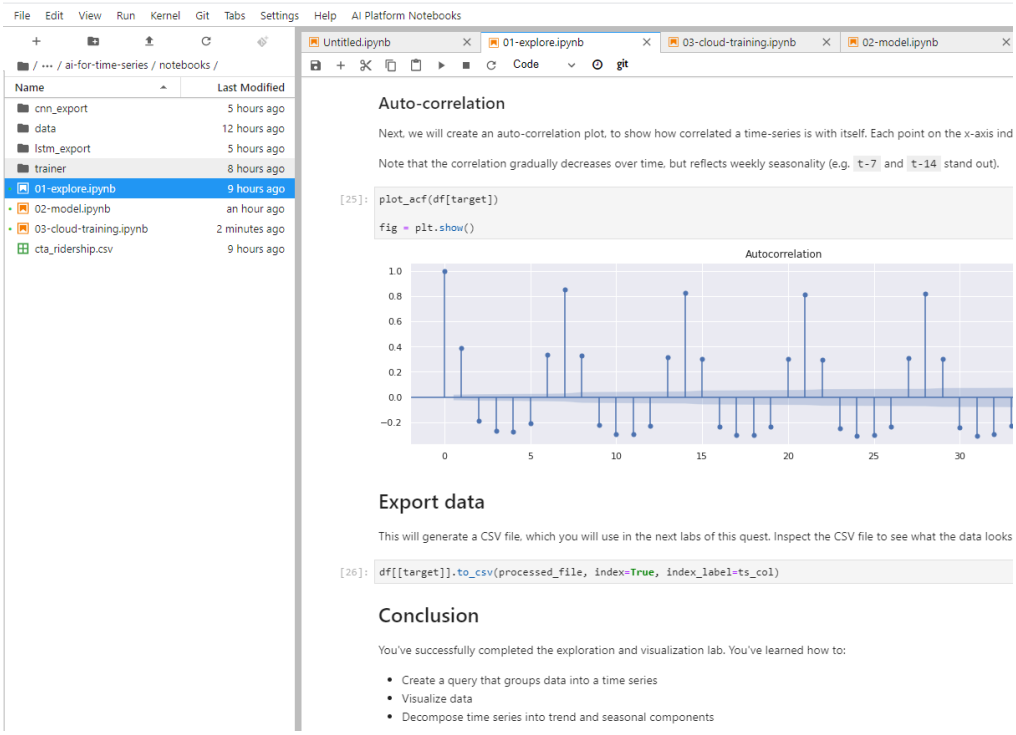
Analyze the patterns



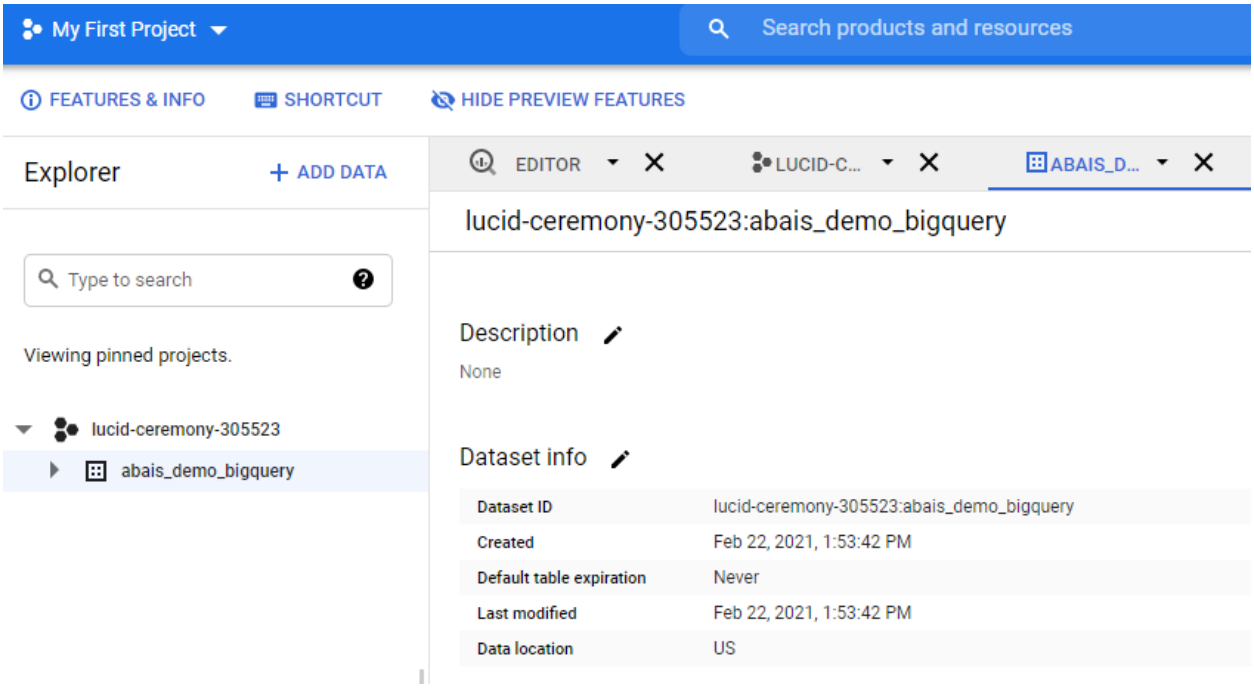
Review summary statistics



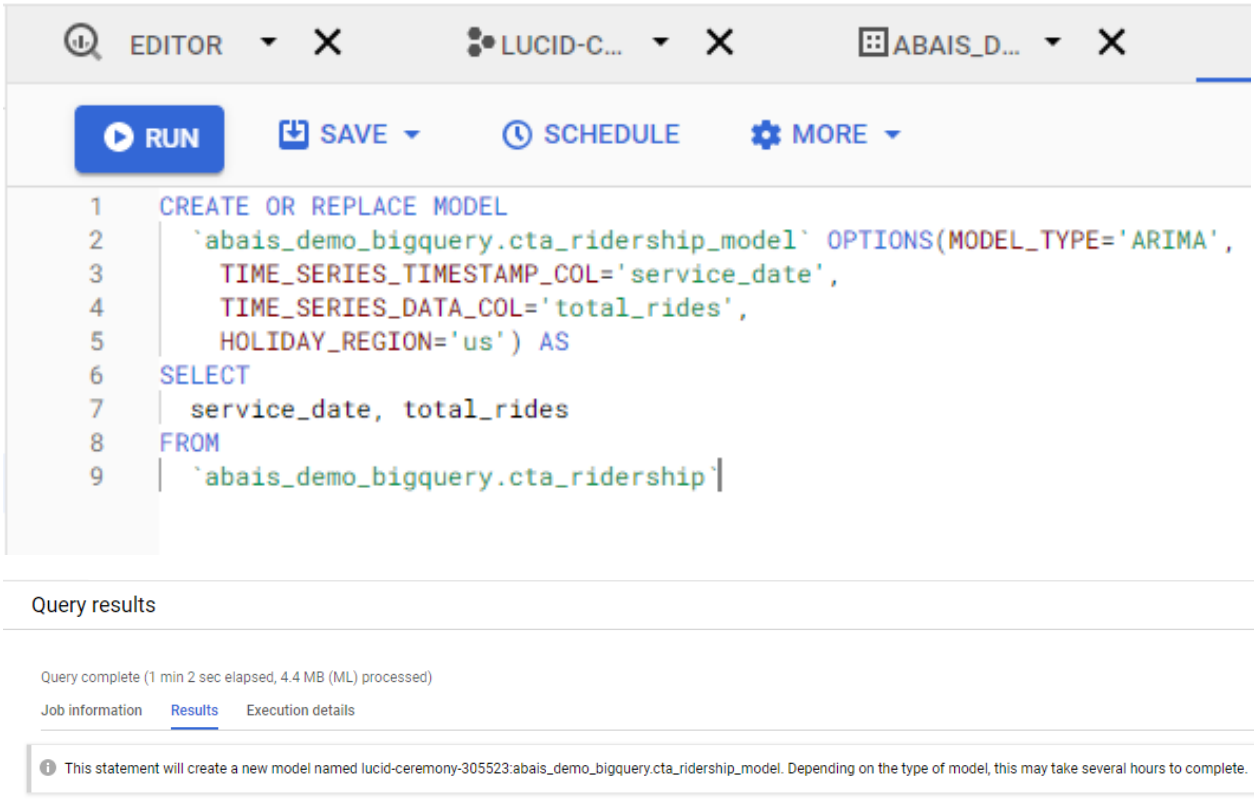
Perform auto correlation



d) Create a table for BigQuery with cta_ridership.csv data



e) Create a time-series query and run it



f) Query model with evaluation statistics in asc order of AIC

RUN SAVE SCHEDULE MORE

```
SELECT
  *
FROM
  ML.EVALUATE(MODEL `abais_demo_bigquery.cta_ridership_model`)

```

Query results SAVE RESULTS EXPLORE DATA

Query complete (0.5 sec elapsed, 0 B processed)

Job information | Results | JSON | Execution details

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	2.12328530813070 85E9	WEEKLY
								YEARLY

Rows per page: 100 1 - 42 of 42

g) Run a forecast query

RUN SAVE SCHEDULE MORE This query will process 23.4 KiB when r

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

Query results SAVE RESULTS EXPLORE DATA

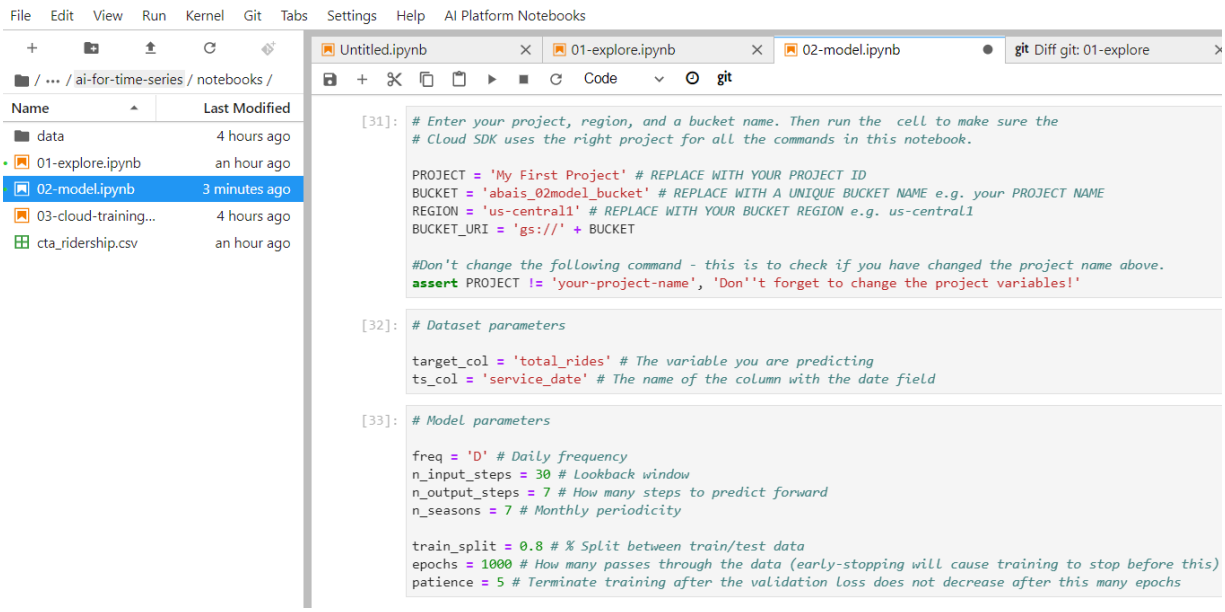
Query complete (0.3 sec elapsed, 23.4 KB processed)

Job information | Results | JSON | Execution details

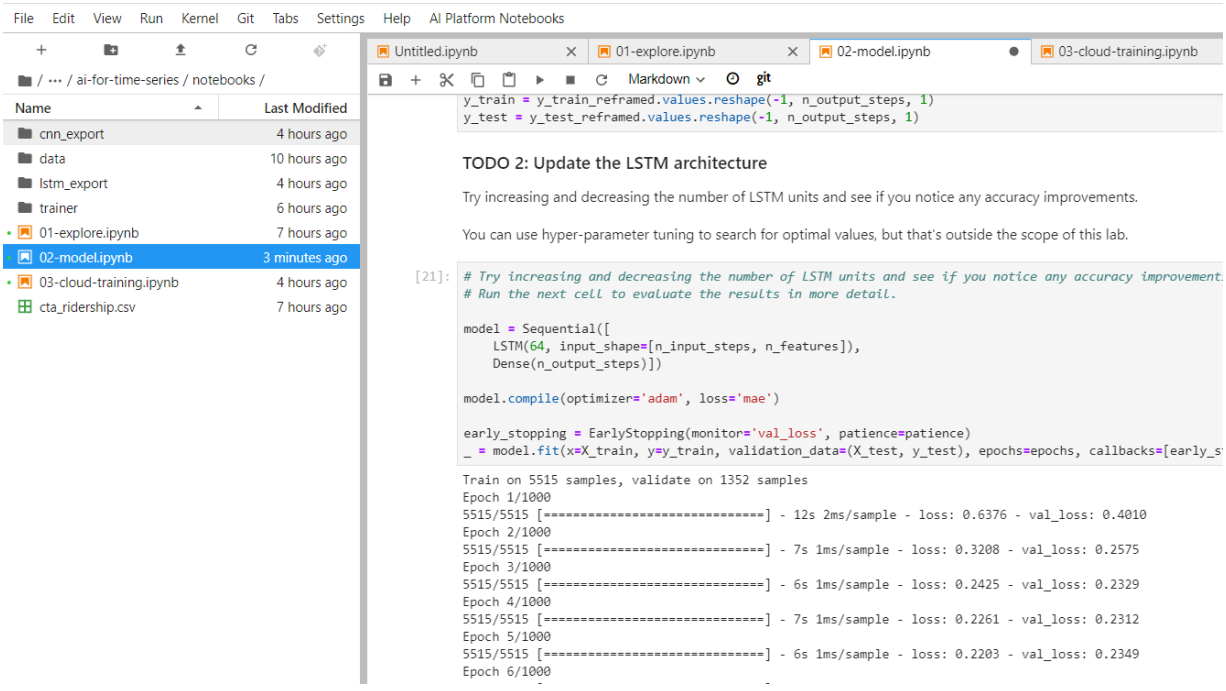
Row	forecast_timestamp	forecast_value	standard_error	confidence_level	prediction_interval_lower_bound	prediction_interval_upper_bound	confidence_interval_lower_bound	confidence_interval_upper_bou
1	2020-01-01 00:00:00 UTC	662436.4424369269	46059.49014554253	0.95	572322.980240453	752549.9046334007	572322.980240453	752549.90463340
2	2020-01-02 00:00:00 UTC	1029641.4669424891	46276.328347693256	0.95	939103.76989082	1120179.1639941582	939103.76989082	1120179.16399415
3	2020-01-03 00:00:00 UTC	1201660.2034356925	47233.43871922012	0.95	1109249.9600529654	1294070.4468184195	1109249.9600529654	1294070.44681841
4	2020-01-04 00:00:00 UTC	651095.9776391207	48157.99332862347	0.95	556876.8819095747	745315.0733686666	556876.8819095747	745315.07336866
5	2020-01-05 00:00:00 UTC	467394.91846646497	48621.50963880497	0.95	372268.97250121285	562520.8644317171	372268.97250121285	562520.86443171
6	2020-01-06 00:00:00 UTC	1158999.319539823	48869.23710364581	0.95	1063388.705171438	1254609.9339082083	1063388.705171438	1254609.93390820

h) Build a custom forecasting model 02-model.ipynb screenshots

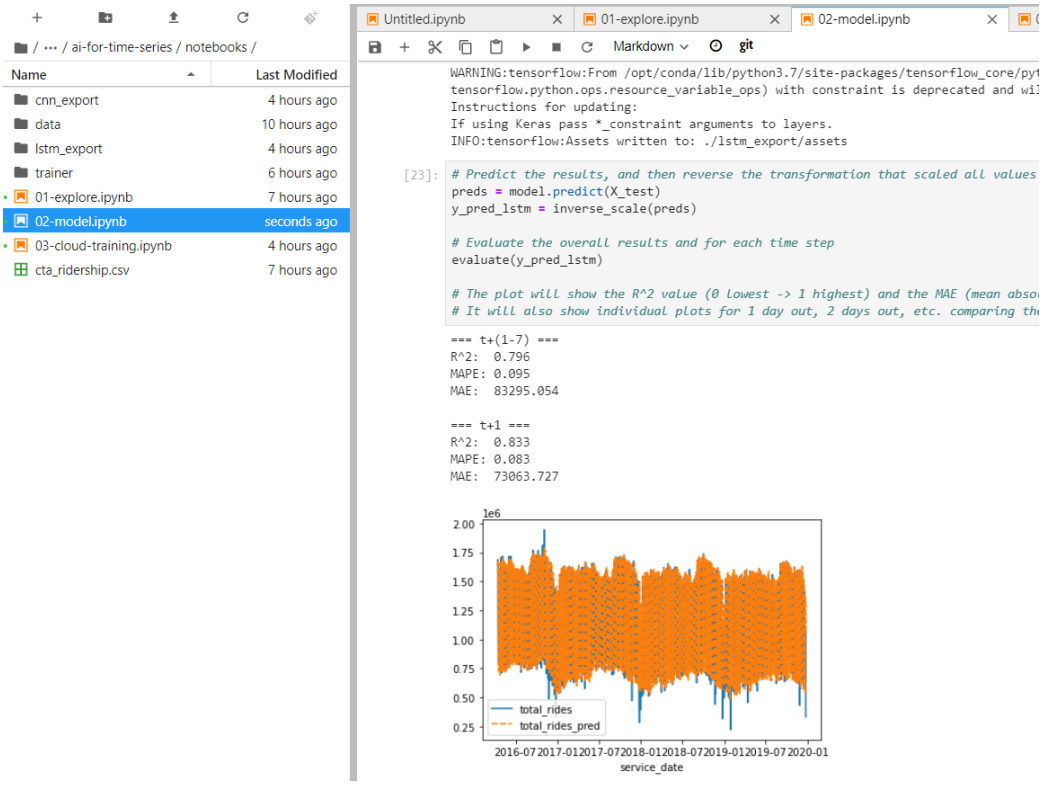
Setup PROJECT, BUCKET



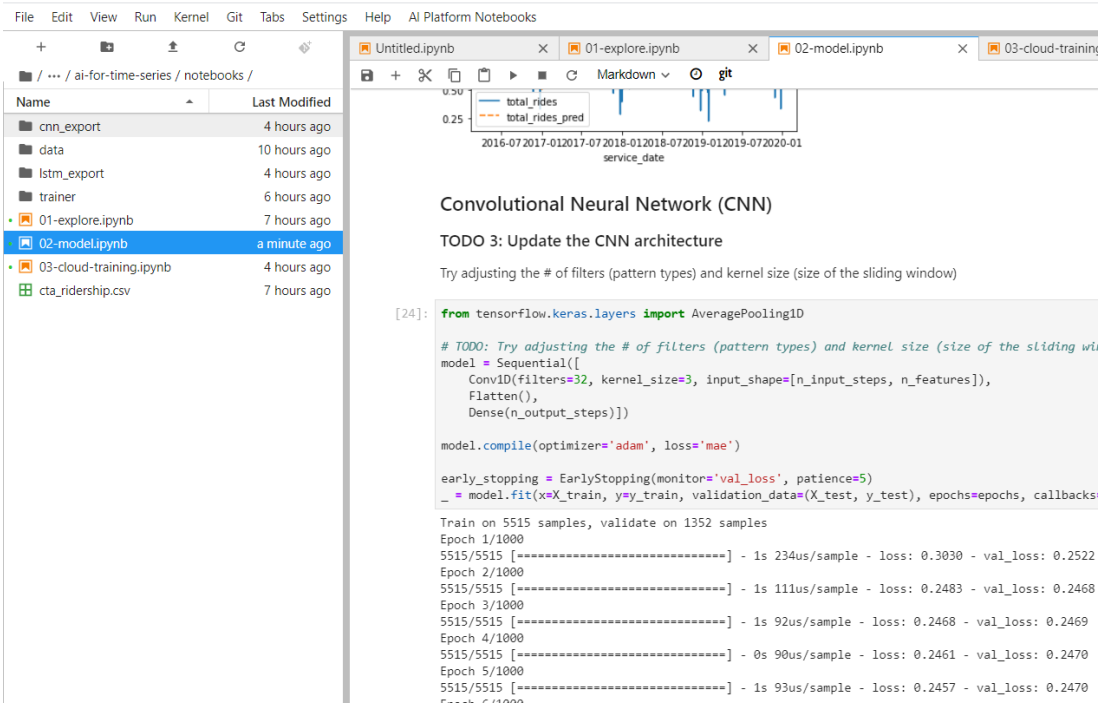
Update LSTM architecture



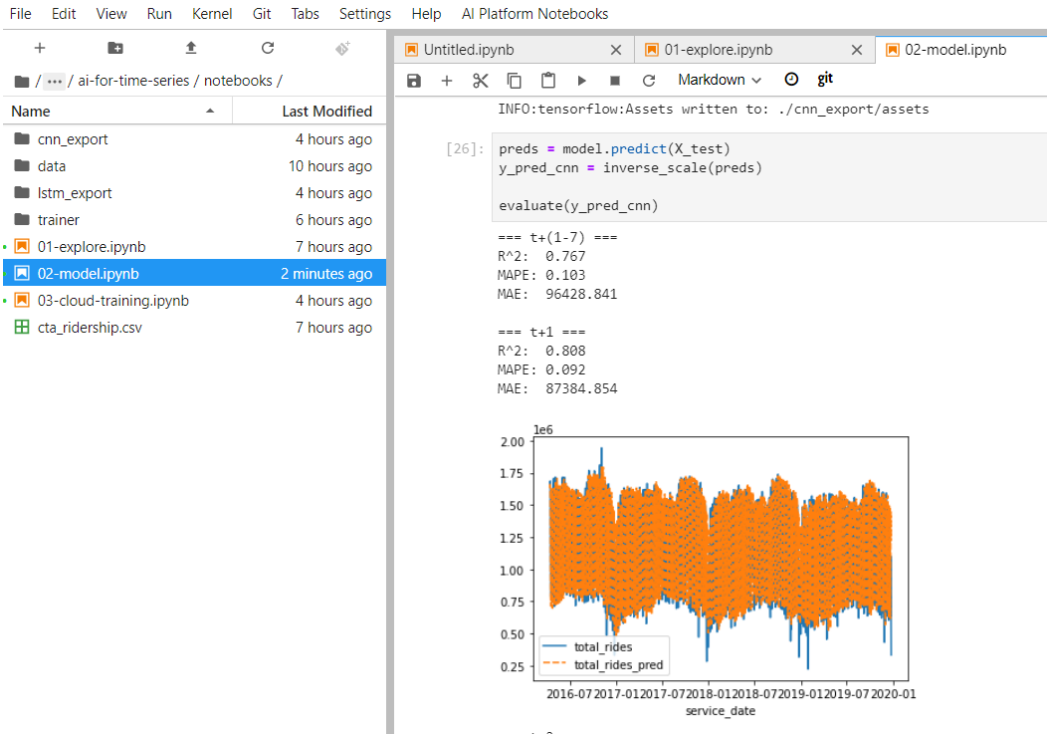
Evaluate results for each timestamp



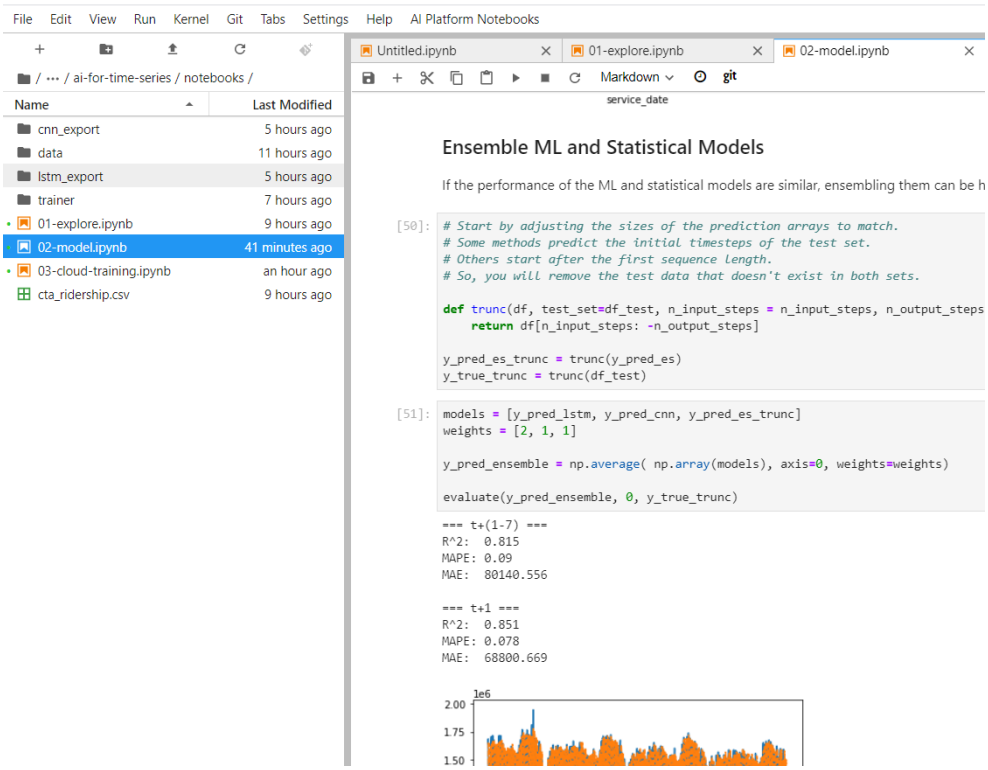
Update CNN architecture



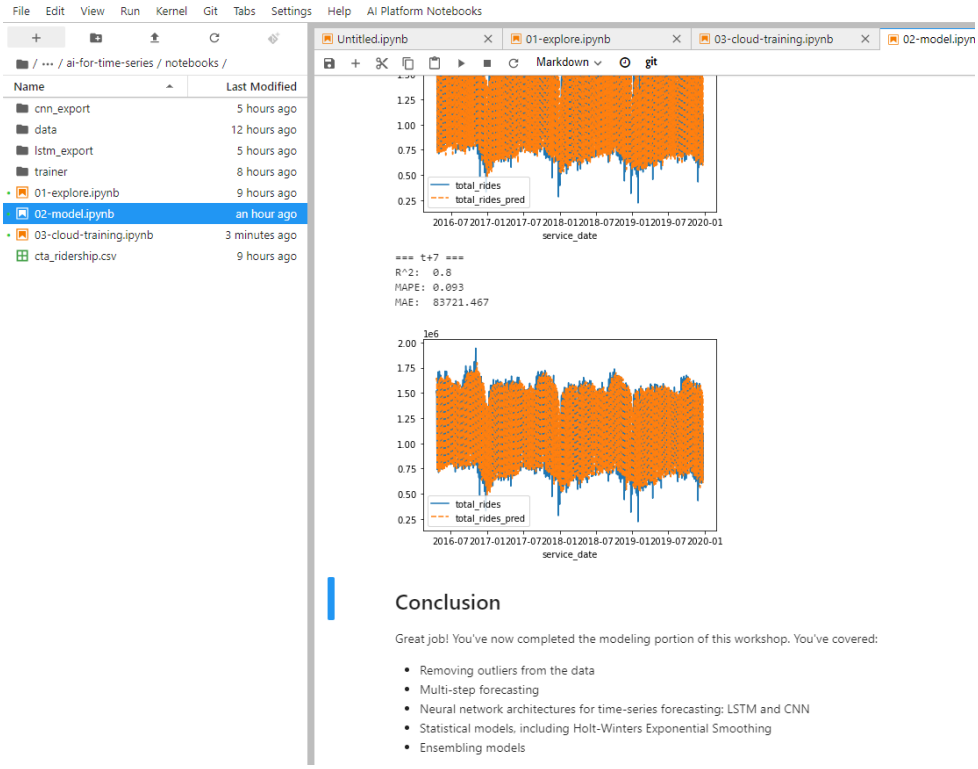
Evaluate the CNN model



Ensemble ML and statistical models

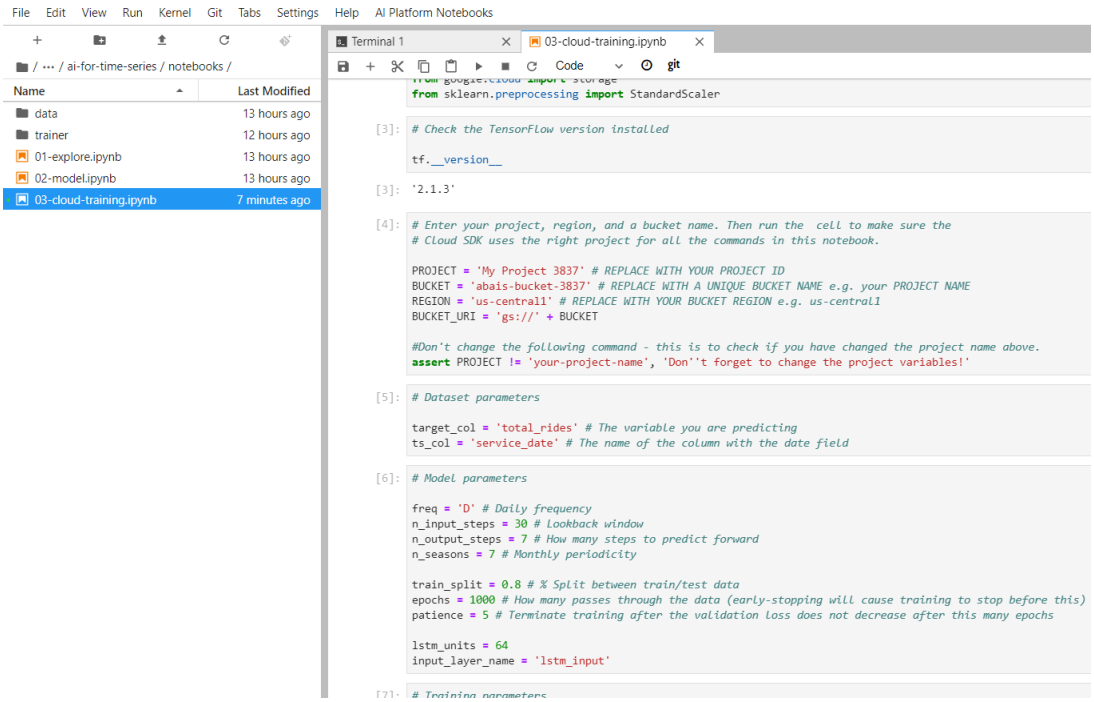


Study the conclusions

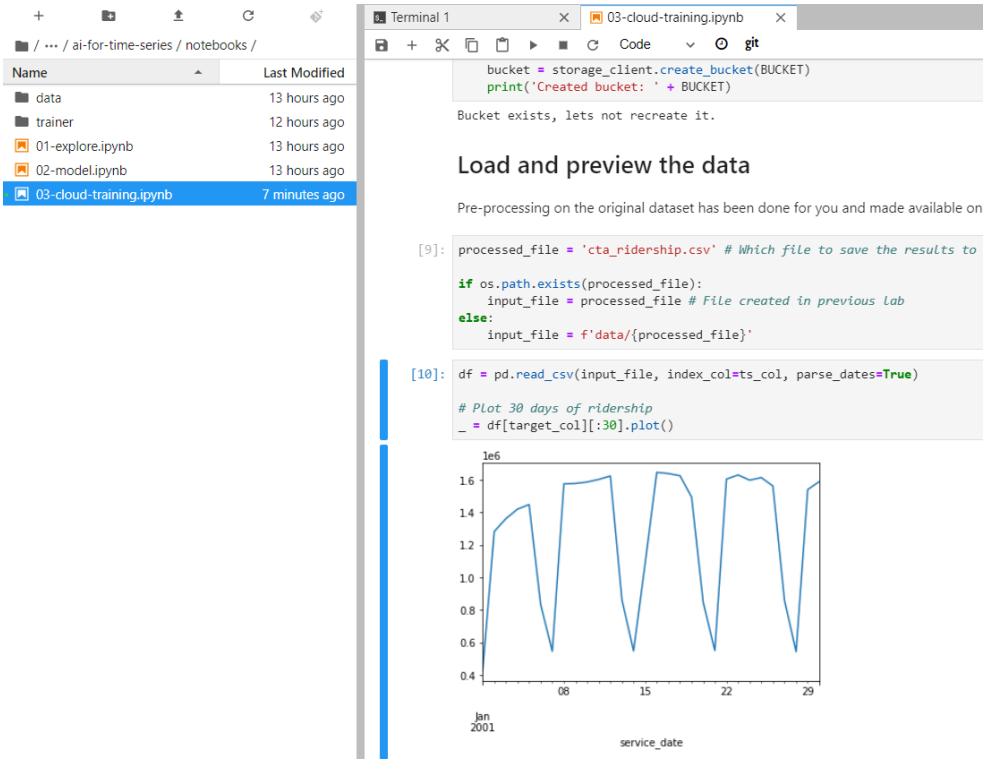


i) Train, predict in the cloud - 03-cloud-training.ipynb screenshots

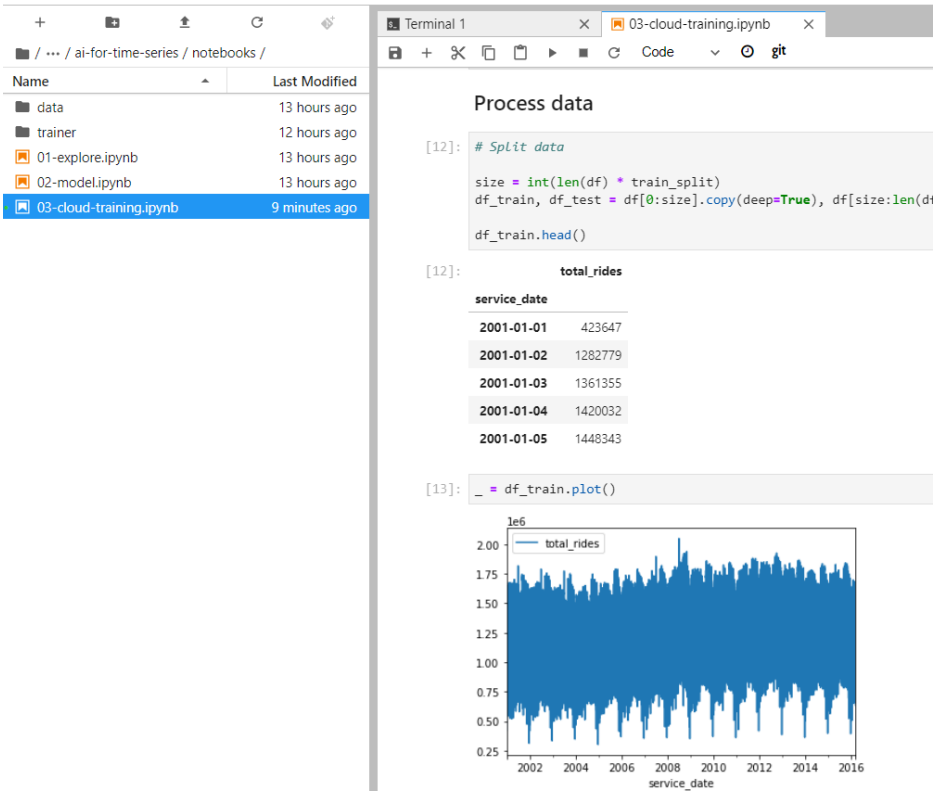
Setup PROJECT, BUCKET



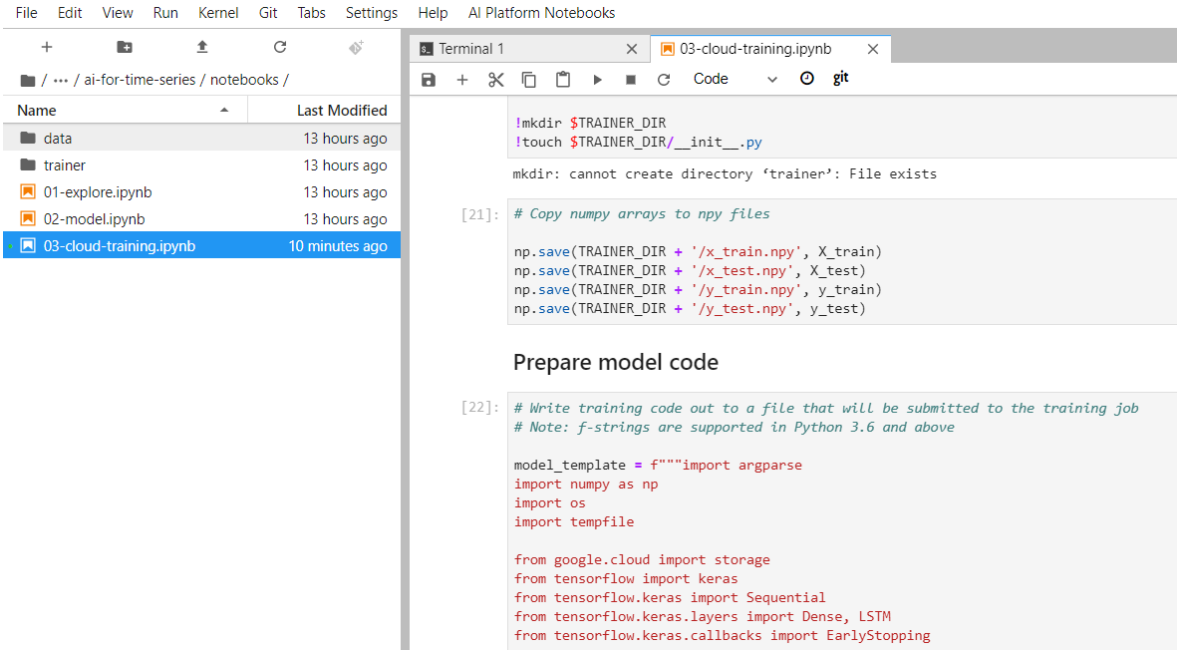
Load and Preview the data

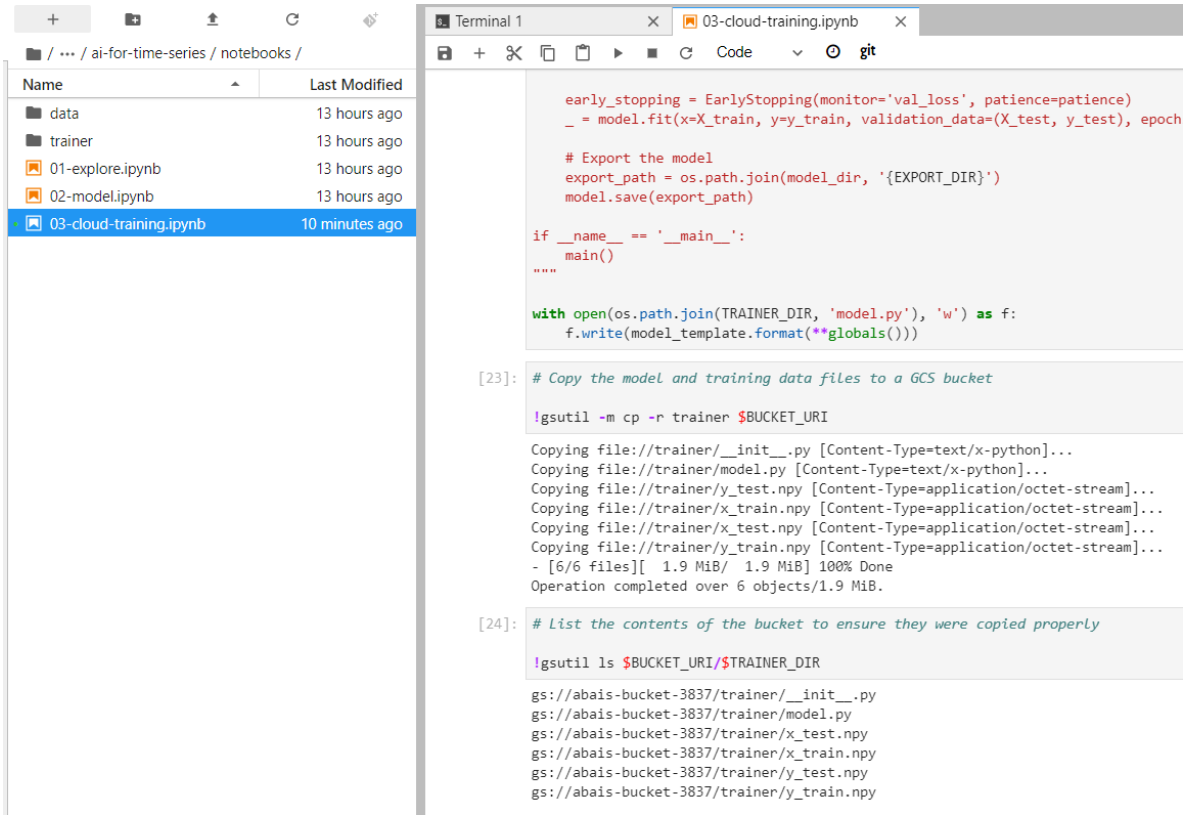


Process the data

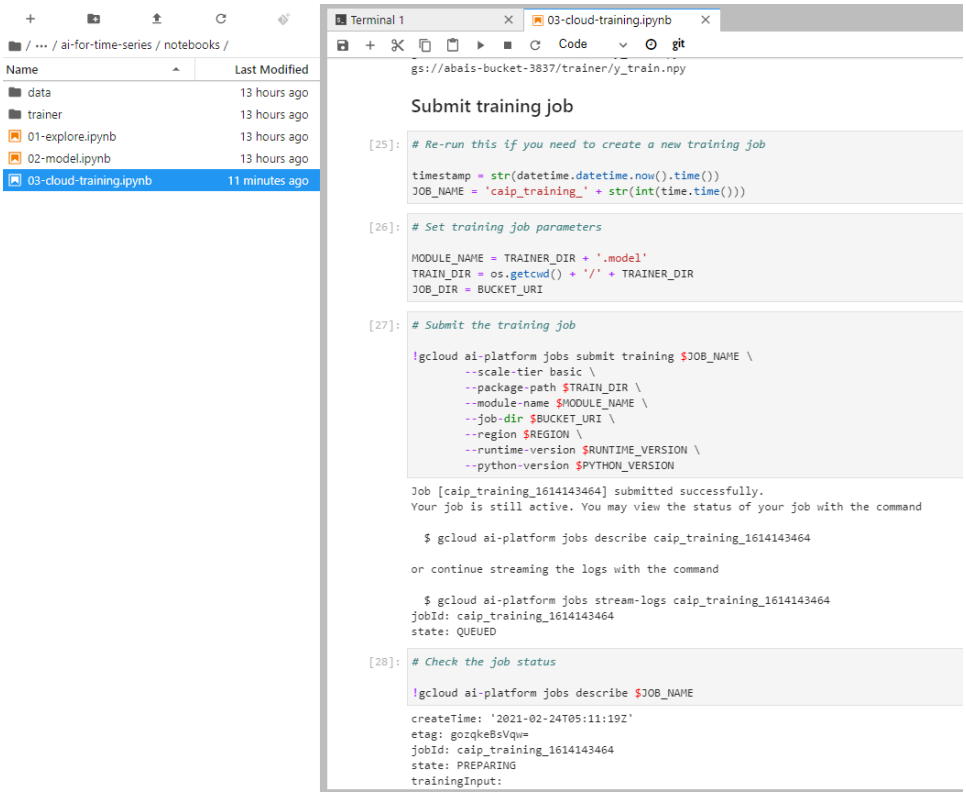


Prepare the model

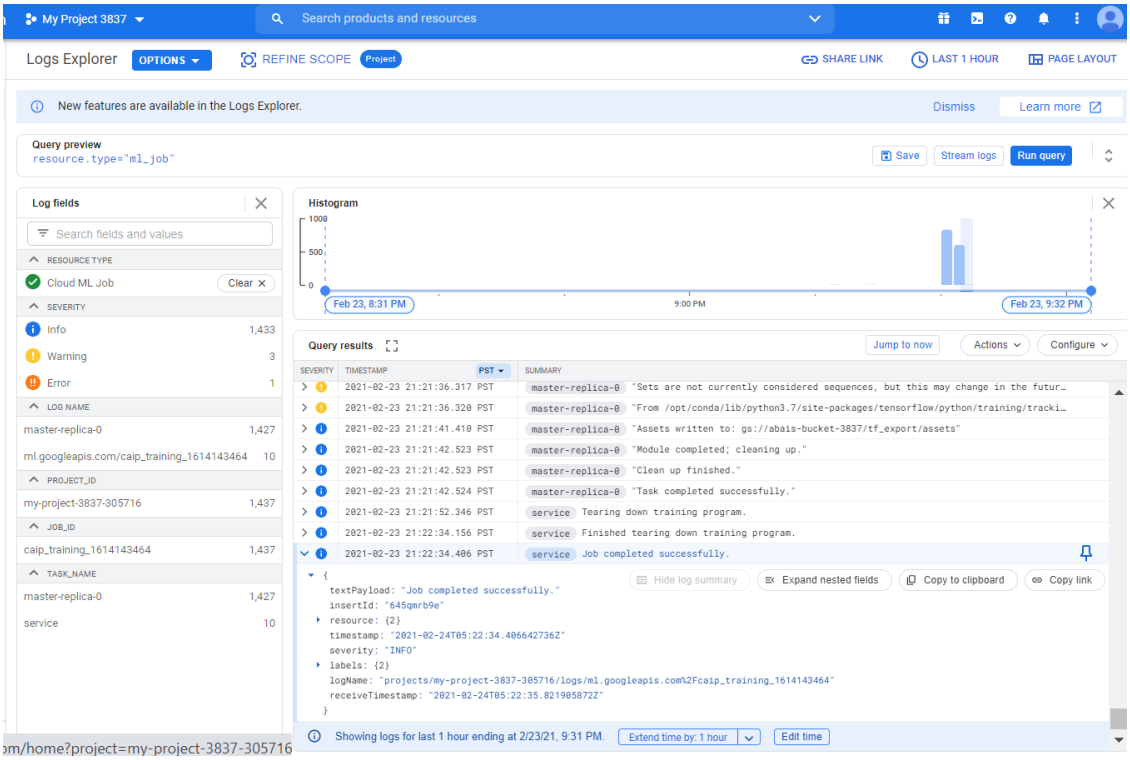




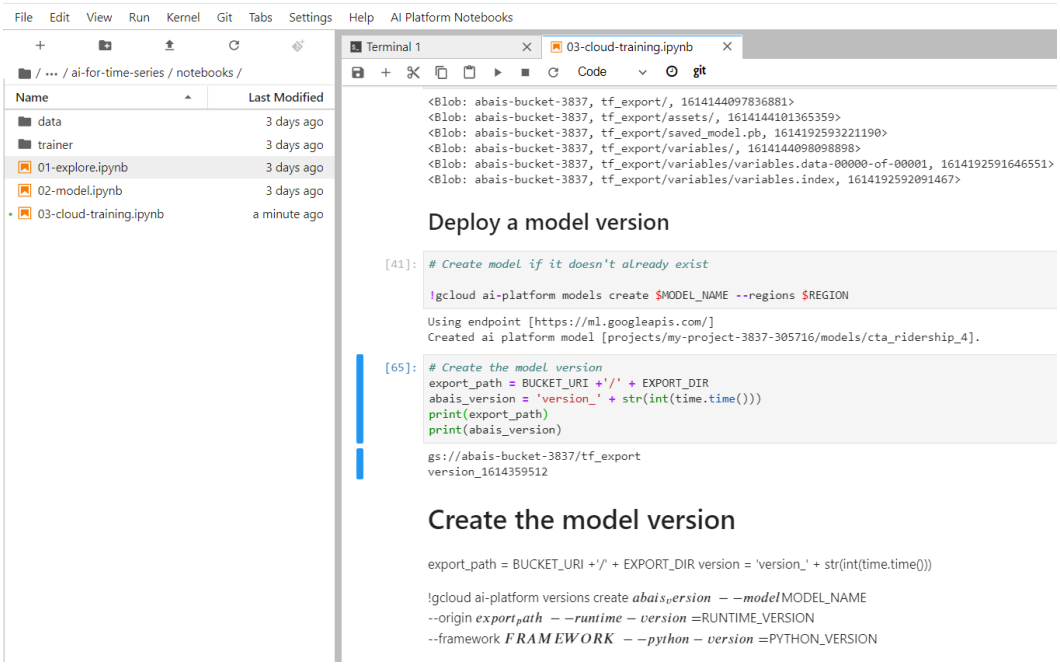
Submit a training job



Monitor the training job



Create a Model Version, Deploy the Model



Make Predictions on deployed model

```
print('Input:')
print(X_test.tolist()[:5])
pred_val = predict_json(PROJECT, REGION, MODEL_NAME, X_test.tolist(), version=abais_version)

print()

print('Predictions:')
print(pred_val[:5])
```

```
Input:
[[[0.6376744344043458], [0.6876651648609055], [0.6852415102977003], [0.7590728805100179], [0.648564383860 59], [-1.0852495684854537], [-1.
7235056739500692], [0.4231264416097936], [0.5189255122400206], [0.42332185668975886], [0.18801672182980905], [-0.03106388567666394], [-1.
1461023319582624], [-1.9432461245836974], [0.47850758284357586], [0.6881042143262819], [0.5709744456816684], [0.5195472874944554], [0.584
9656578355427], [-1.23690689768756], [-1.635893733812931], [0.5416571079705214], [0.730131145809713], [0.5789052524984392], [0.6798434859
459335], [0.3810208365227401], [-1.0912998223898318], [-1.8741808520359864], [0.7131757153000026], [0.7051510077305214]], [[0.68766516486
09055], [0.6852415102977003], [0.7590728805100179], [0.648564383860 59], [-1.0852495684854537], [-1.7235056739500692], [0.423126441609793
6], [0.5189255122400206], [0.42332185668975886], [0.18801672182980905], [-0.03106388567666394], [-1.1461023319582624], [-1.943246124583697
4], [0.47850758284357586], [0.6881042143262819], [0.5709744456816684], [0.5195472874944554], [0.5849656578355427], [-1.23690689768756],
[-1.635893733812931], [0.5416571079705214], [0.730131145809713], [0.5789052524984392], [0.6798434859459335], [0.3810208365227401], [-1.09
12998223898318], [-1.8741808520359864], [0.7131757153000026], [0.7051510077305214], [0.6630174862034729]], [[0.6852415102977003], [0.7590
728805100179], [0.64856438386059], [-1.0852495684854537], [-1.7235056739500692], [0.4231264416097936], [0.5189255122400206], [0.423321856
68975886], [0.18801672182980905], [-0.03106388567666394], [-1.1461023319582624], [-1.9432461245836974], [0.47850758284357586], [0.6881042
143262819], [0.5709744456816684], [0.5195472874944554], [0.5849656578355427], [-1.23690689768756], [-1.635893733812931], [0.5416571079705
214], [0.730131145809713], [0.5789052524984392], [0.6798434859459335], [0.3810208365227401], [-1.0912998223898318], [-1.8741808520359864
4], [0.7131757153000026], [0.7051510077305214], [0.6630174862034729], [0.7169799647138712]], [[0.7590728805100179], [0.64856438386059],
[-1.0852495684854537], [-1.7235056739500692], [0.4231264416097936], [0.5189255122400206], [0.42332185668975886], [0.18801672182980905],
[-0.03106388567666394], [-1.1461023319582624], [-1.9432461245836974], [0.47850758284357586], [0.6881042143262819], [0.5709744456816684],
[0.5195472874944554], [0.5849656578355427], [-1.23690689768756], [-1.635893733812931], [0.5416571079705214], [0.730131145809713], [0.5789
052524984392], [0.6798434859459335], [0.3810208365227401], [-1.0912998223898318], [-1.8741808520359864], [0.7131757153000026], [0.7051510
077305214], [0.6630174862034729], [0.7169799647138712], [0.7652855573380033]], [[0.64856438386059], [-1.0852495684854537], [-1.7235056739
500692], [0.4231264416097936], [0.5189255122400206], [0.42332185668975886], [0.18801672182980905], [-0.03106388567666394], [-1.1461023319
582624], [-1.9432461245836974], [0.47850758284357586], [0.6881042143262819], [0.5709744456816684], [0.5195472874944554], [0.5849656578355
427], [-1.23690689768756], [-1.635893733812931], [0.5416571079705214], [0.730131145809713], [0.5789052524984392], [0.6798434859459335],
[0.3810208365227401], [-1.0912998223898318], [-1.8741808520359864], [0.7131757153000026], [0.7051510077305214], [0.6630174862034729], [0.
7169799647138712], [0.7652855573380033], [-0.8274107908331498]]]
projects/round-dreamer-304721/models/cta_ridership_2/versions/version_1613796790

Predictions:
[[0.659684241, 0.626064, 0.676753581, -1.0021131, -1.80011857, 0.605064034, 0.744683266], [0.608709455, 0.735271394, -1.0335294, -1.83254
254, 0.614551127, 0.688534737, 0.642241955], [0.710176051, -1.01311612, -1.82813942, 0.576377749, 0.706874132, 0.580318511, 0.614244103],
[-1.0662595, -1.86967057, 0.567883909, 0.750643671, 0.643298325, 0.624556184, 0.671228817], [-1.81234813, 0.597224957, 0.749779522, 0.7131
```

```
# Print prediction and compare to actual value

print('Predicted riders:', int(round(inverse_scale(np.array([pred_val[0][0]]).reshape(1,1))[0][0])))
print('Actual riders:   ', int(round(inverse_scale(np.array([y_test[0]]))[0][0])))
```

```
Predicted riders: 1646008
Actual riders:    1647321
```

Cleanup

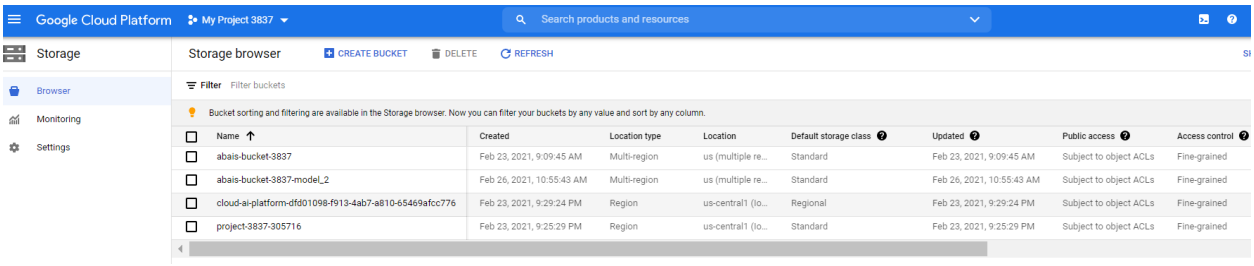
```
# Delete model version resource
!echo gcloud ai-platform versions delete {abais_version} --model {MODEL_NAME} --quiet

# Delete model resource
!echo gcloud ai-platform models delete {MODEL_NAME} --quiet
```

```
gcloud ai-platform versions delete version_1614359512 --model cta_ridership_4 --quiet
gcloud ai-platform models delete cta_ridership_4 --quiet
```

Storage Buckets

For My Project 3837



For My First Project