ML_with_TF_in_VertexAI

April 25, 2023

0.1 Create minimal training and validation data

[1]:
```
import os, json, math, shutil
import numpy as np
import tensorflow as tf
#set environment variables to use bash cells
PROJECT = !(gcloud config get-value project)
# print(PROJECT)
PROJECT = PROJECT[0]
REGION = 'us-central1'
BUCKET = "{}-dsongcp".format(PROJECT)
os.environ["ENDPOINT_NAME"] = "flights"
os.environ["BUCKET"] = BUCKET
os.environ["REGION"] = REGION
os.environ["TF_VERSION"] = "2-" + tf.__version__[2:3]
```

[2]:
```
%%bigquery
# Export files that contain training, validation data
CREATE OR REPLACE TABLE dsongcp.flights_train_data AS
SELECT
  IF(arr_delay < 15, 1.0, 0.0) AS ontime,
  dep_delay,
  taxi_out,
  distance,
  origin,
  dest,
  EXTRACT(hour FROM dep_time) AS dep_hour,
  IF (EXTRACT(dayofweek FROM dep_time) BETWEEN 2 AND 6, 1, 0) AS is_weekday,
  UNIQUE_CARRIER AS carrier,
  dep_airport_lat,
  dep_airport_lon,
  arr_airport_lat,
  arr_airport_lon
FROM dsongcp.flights_tzcorr f
JOIN dsongcp.trainday t
ON f.FL_DATE = t.FL_DATE
WHERE
  f.CANCELLED = False AND
```

```
  f.DIVERTED = False AND
  is_train_day = 'True'
```

```
Query is running:   0%|          |
```

[2]:
```
Empty DataFrame
Columns: []
Index: []
```

[3]:
```
%%bigquery
# Create the evaluation dataset flights_eval_data for model evaluation:
CREATE OR REPLACE TABLE dsongcp.flights_eval_data AS
SELECT
  IF(arr_delay < 15, 1.0, 0.0) AS ontime,
  dep_delay,
  taxi_out,
  distance,
  origin,
  dest,
  EXTRACT(hour FROM dep_time) AS dep_hour,
  IF (EXTRACT(dayofweek FROM dep_time) BETWEEN 2 AND 6, 1, 0) AS is_weekday,
  UNIQUE_CARRIER AS carrier,
  dep_airport_lat,
  dep_airport_lon,
  arr_airport_lat,
  arr_airport_lon
FROM dsongcp.flights_tzcorr f
JOIN dsongcp.trainday t
ON f.FL_DATE = t.FL_DATE
WHERE
  f.CANCELLED = False AND
  f.DIVERTED = False AND
  is_train_day = 'False'
```

```
Query is running:   0%|          |
```

[3]:
```
Empty DataFrame
Columns: []
Index: []
```

[4]:
```
%%bigquery
# Create the full dataset flights_all_data using the following code:
CREATE OR REPLACE TABLE dsongcp.flights_all_data AS
SELECT
  IF(arr_delay < 15, 1.0, 0.0) AS ontime,
  dep_delay,
```

```
  taxi_out,
  distance,
  origin,
  dest,
  EXTRACT(hour FROM dep_time) AS dep_hour,
  IF (EXTRACT(dayofweek FROM dep_time) BETWEEN 2 AND 6, 1, 0) AS is_weekday,
  UNIQUE_CARRIER AS carrier,
  dep_airport_lat,
  dep_airport_lon,
  arr_airport_lat,
  arr_airport_lon,
  IF (is_train_day = 'True',
      IF(ABS(MOD(FARM_FINGERPRINT(CAST(f.FL_DATE AS STRING)), 100)) < 60,␣
 ↪'TRAIN', 'VALIDATE'),
      'TEST') AS data_split
FROM dsongcp.flights_tzcorr f
JOIN dsongcp.trainday t
ON f.FL_DATE = t.FL_DATE
WHERE
  f.CANCELLED = False AND
  f.DIVERTED = False
```

```
Query is running:   0%|          |
```

[4]:
```
Empty DataFrame
Columns: []
Index: []
```

[5]:
```
%%bash
PROJECT=$(gcloud config get-value project)
for dataset in "train" "eval" "all"; do
  TABLE=dsongcp.flights_${dataset}_data
  CSV=gs://${BUCKET}/ch9/data/${dataset}.csv
  echo "Exporting ${TABLE} to ${CSV} and deleting table"
  bq --project_id=${PROJECT} extract --destination_format=CSV $TABLE $CSV
  bq --project_id=${PROJECT} rm -f $TABLE
done
```

```
Exporting dsongcp.flights_train_data to gs://qwiklabs-
gcp-01-c3764cc81f6b-dsongcp/ch9/data/train.csv and deleting table
Exporting dsongcp.flights_eval_data to gs://qwiklabs-
gcp-01-c3764cc81f6b-dsongcp/ch9/data/eval.csv and deleting table
Exporting dsongcp.flights_all_data to gs://qwiklabs-
gcp-01-c3764cc81f6b-dsongcp/ch9/data/all.csv and deleting table

Waiting on bqjob_r60ae91da6b8e0a89_00000187b9b16e17_1 … (65s) Current status:
DONE
Waiting on bqjob_r7f616cecbe84f130_00000187b9b27de1_1 … (23s) Current status:
```

```
DONE
Waiting on bqjob_r3b286050cf254292_00000187b9b2e85b_1 … (85s) Current status:
DONE
```

[6]:
```
!gsutil ls -lh gs://{BUCKET}/ch9/data
```

```
445.01 MiB  2023-04-25T18:36:38Z  gs://qwiklabs-
gcp-01-c3764cc81f6b-dsongcp/ch9/data/all.csv
115.19 MiB  2023-04-25T18:35:15Z  gs://qwiklabs-
gcp-01-c3764cc81f6b-dsongcp/ch9/data/eval.csv
296.96 MiB  2023-04-25T18:34:37Z  gs://qwiklabs-
gcp-01-c3764cc81f6b-dsongcp/ch9/data/train.csv
TOTAL: 3 objects, 898801258 bytes (857.16 MiB)
```

0.2 Create the input data

[7]:
```
DEVELOP_MODE = True
NUM_EXAMPLES = 5000*1000
```

[8]:
```
training_data_uri = 'gs://{}/ch9/data/train*'.format(BUCKET)
validation_data_uri = 'gs://{}/ch9/data/eval*'.format(BUCKET)
```

[9]:
```
NBUCKETS = 5
NEMBEDS = 3
TRAIN_BATCH_SIZE = 64
DNN_HIDDEN_UNITS = '64,32'
```

[10]:
```
if DEVELOP_MODE:
    train_df = tf.data.experimental.make_csv_dataset(training_data_uri,␣
 ↪batch_size=5)
    for n, data in enumerate(train_df):
        numpy_data = {k: v.numpy() for k, v in data.items()}
        print(n, numpy_data)
        if n==1: break
```

```
2023-04-25 18:36:53.734706: W
tensorflow/stream_executor/platform/default/dso_loader.cc:64] Could not load
dynamic library 'libcuda.so.1'; dlerror: libcuda.so.1: cannot open shared object
file: No such file or directory; LD_LIBRARY_PATH:
/usr/local/cuda/lib64:/usr/local/nccl2/lib:/usr/local/cuda/extras/CUPTI/lib64
2023-04-25 18:36:53.734757: W
tensorflow/stream_executor/cuda/cuda_driver.cc:269] failed call to cuInit:
UNKNOWN ERROR (303)
2023-04-25 18:36:53.734785: I
tensorflow/stream_executor/cuda/cuda_diagnostics.cc:156] kernel driver does not
appear to be running on this host (tensorflow): /proc/driver/nvidia/version does
not exist
2023-04-25 18:36:53.738700: I tensorflow/core/platform/cpu_feature_guard.cc:142]
```

```
This TensorFlow binary is optimized with oneAPI Deep Neural Network Library
(oneDNN) to use the following CPU instructions in performance-critical
operations:  AVX2 FMA
To enable them in other operations, rebuild TensorFlow with the appropriate
compiler flags.
2023-04-25 18:36:54.857494: I
tensorflow/compiler/mlir/mlir_graph_optimization_pass.cc:185] None of the MLIR
Optimization Passes are enabled (registered 2)

0 {'ontime': array([1, 1, 1, 1, 1], dtype=int32), 'dep_delay': array([ 3, 15,
-6,  6, -1], dtype=int32), 'taxi_out': array([10, 13, 10, 13, 47], dtype=int32),
'distance': array([ 216,  967,  727, 1020,  337], dtype=int32), 'origin':
array([b'HNL', b'DEN', b'TPA', b'PHX', b'LAX'], dtype=object), 'dest':
array([b'ITO', b'SFO', b'SDF', b'GEG', b'SFO'], dtype=object), 'dep_hour':
array([21,  2, 22, 20, 17], dtype=int32), 'is_weekday': array([1, 0, 1, 1, 1],
dtype=int32), 'carrier': array([b'HA', b'UA', b'WN', b'WN', b'UA'],
dtype=object), 'dep_airport_lat': array([21.317778, 39.861668, 27.975555,
33.434166, 33.9425  ],
      dtype=float32), 'dep_airport_lon': array([-157.92027 , -104.67306 ,
-82.53333 , -112.011665, -118.40806 ],
      dtype=float32), 'arr_airport_lat': array([19.720278, 37.61889 , 38.174168,
47.619167, 37.61889 ],
      dtype=float32), 'arr_airport_lon': array([-155.04834, -122.375  ,
-85.73639, -117.53528, -122.375  ],
      dtype=float32)}
1 {'ontime': array([1, 0, 1, 1, 1], dtype=int32), 'dep_delay': array([18, 29,
-3,  1, -6], dtype=int32), 'taxi_out': array([11, 17, 12, 12, 14], dtype=int32),
'distance': array([ 550, 1020,  284,  500,  224], dtype=int32), 'origin':
array([b'PDX', b'PHX', b'DTW', b'ATL', b'SEA'], dtype=object), 'dest':
array([b'SFO', b'GEG', b'CIU', b'AEX', b'GEG'], dtype=object), 'dep_hour':
array([ 1,  4, 15, 20,  5], dtype=int32), 'is_weekday': array([1, 1, 0, 1, 1],
dtype=int32), 'carrier': array([b'OO', b'AA', b'OO', b'EV', b'AS'],
dtype=object), 'dep_airport_lat': array([45.58861 , 33.434166, 42.2125  ,
33.636665, 47.45    ],
      dtype=float32), 'dep_airport_lon': array([-122.59695 , -112.011665,
-83.35333 ,  -84.42778 , -122.31167 ],
      dtype=float32), 'arr_airport_lat': array([37.61889 , 47.619167, 46.25083 ,
31.3275  , 47.619167],
      dtype=float32), 'arr_airport_lon': array([-122.375   , -117.53528 ,
-84.4725  ,  -92.548615, -117.53528 ],
      dtype=float32)}
```

[11]:
```
def features_and_labels(features):
  label = features.pop('ontime')
  return features, label

def read_dataset(pattern, batch_size, mode=tf.estimator.ModeKeys.TRAIN,␣
 ↪truncate=None):
```

```
  dataset = tf.data.experimental.make_csv_dataset(pattern, batch_size,␣
 ↪num_epochs=1)
  dataset = dataset.map(features_and_labels)
  if mode == tf.estimator.ModeKeys.TRAIN:
    dataset = dataset.shuffle(batch_size*10)
    dataset = dataset.repeat()
  dataset = dataset.prefetch(1)
  if truncate is not None:
    dataset = dataset.take(truncate)
  return dataset

if DEVELOP_MODE:
    print("Checking input pipeline")
    one_item = read_dataset(training_data_uri, batch_size=2, truncate=1)
    print(list(one_item)) # should print one batch of 2 items
```

```
Checking input pipeline
[(OrderedDict([('dep_delay', <tf.Tensor: shape=(2,), dtype=int32,
numpy=array([54,  1], dtype=int32)>), ('taxi_out', <tf.Tensor: shape=(2,),
dtype=int32, numpy=array([11, 22], dtype=int32)>), ('distance', <tf.Tensor:
shape=(2,), dtype=int32, numpy=array([1846,  262], dtype=int32)>), ('origin',
<tf.Tensor: shape=(2,), dtype=string, numpy=array([b'ORD', b'SBA'],
dtype=object)>), ('dest', <tf.Tensor: shape=(2,), dtype=string,
numpy=array([b'SFO', b'SFO'], dtype=object)>), ('dep_hour', <tf.Tensor:
shape=(2,), dtype=int32, numpy=array([15,  1], dtype=int32)>), ('is_weekday',
<tf.Tensor: shape=(2,), dtype=int32, numpy=array([1, 1], dtype=int32)>),
('carrier', <tf.Tensor: shape=(2,), dtype=string, numpy=array([b'AA', b'OO'],
dtype=object)>), ('dep_airport_lat', <tf.Tensor: shape=(2,), dtype=float32,
numpy=array([41.979443, 34.42611 ], dtype=float32)>), ('dep_airport_lon',
<tf.Tensor: shape=(2,), dtype=float32, numpy=array([ -87.9075 , -119.84139],
dtype=float32)>), ('arr_airport_lat', <tf.Tensor: shape=(2,), dtype=float32,
numpy=array([37.61889, 37.61889], dtype=float32)>), ('arr_airport_lon',
<tf.Tensor: shape=(2,), dtype=float32, numpy=array([-122.375, -122.375],
dtype=float32)>)]), <tf.Tensor: shape=(2,), dtype=int32, numpy=array([0, 1],
dtype=int32)>)]
```

Create, train and evaluate TensorFlow model

[12]:
```
real = {
    colname : tf.feature_column.numeric_column(colname)
          for colname in
            (
                'dep_delay,taxi_out,distance,dep_hour,is_weekday,' +
                'dep_airport_lat,dep_airport_lon,' +
                'arr_airport_lat,arr_airport_lon'
            ).split(',')
}
sparse = {
```

```
    'carrier': tf.feature_column.
↪categorical_column_with_vocabulary_list('carrier',
                vocabulary_list='AS,VX,F9,UA,US,WN,HA,EV,MQ,DL,OO,B6,NK,AA'.
↪split(',')),
    'origin' : tf.feature_column.
↪categorical_column_with_hash_bucket('origin', hash_bucket_size=1000),
    'dest'   : tf.feature_column.categorical_column_with_hash_bucket('dest',␣
↪hash_bucket_size=1000),
}
```

[13]:
```
inputs = {
    colname : tf.keras.layers.Input(name=colname, shape=(), dtype='float32')
          for colname in real.keys()
}
inputs.update({
    colname : tf.keras.layers.Input(name=colname, shape=(), dtype='string')
          for colname in sparse.keys()
})
```

0.3 Bucketing

[14]:
```
latbuckets = np.linspace(20.0, 50.0, NBUCKETS).tolist()  # USA
lonbuckets = np.linspace(-120.0, -70.0, NBUCKETS).tolist() # USA
disc = {}
disc.update({
       'd_{}'.format(key) : tf.feature_column.bucketized_column(real[key],␣
↪latbuckets)
          for key in ['dep_airport_lat', 'arr_airport_lat']
})
disc.update({
       'd_{}'.format(key) : tf.feature_column.bucketized_column(real[key],␣
↪lonbuckets)
          for key in ['dep_airport_lon', 'arr_airport_lon']
})
# cross columns that make sense in combination
sparse['dep_loc'] = tf.feature_column.crossed_column(
    [disc['d_dep_airport_lat'], disc['d_dep_airport_lon']], NBUCKETS*NBUCKETS)
sparse['arr_loc'] = tf.feature_column.crossed_column(
    [disc['d_arr_airport_lat'], disc['d_arr_airport_lon']], NBUCKETS*NBUCKETS)
sparse['dep_arr'] = tf.feature_column.crossed_column([sparse['dep_loc'],␣
↪sparse['arr_loc']], NBUCKETS ** 4)
# embed all the sparse columns
embed = {
       'embed_{}'.format(colname) : tf.feature_column.embedding_column(col,␣
↪NEMBEDS)
          for colname, col in sparse.items()
```

```
}
real.update(embed)
# one-hot encode the sparse columns
sparse = {
    colname : tf.feature_column.indicator_column(col)
          for colname, col in sparse.items()
}
if DEVELOP_MODE:
    print(sparse.keys())
    print(real.keys())
```

```
dict_keys(['carrier', 'origin', 'dest', 'dep_loc', 'arr_loc', 'dep_arr'])
dict_keys(['dep_delay', 'taxi_out', 'distance', 'dep_hour', 'is_weekday',
'dep_airport_lat', 'dep_airport_lon', 'arr_airport_lat', 'arr_airport_lon',
'embed_carrier', 'embed_origin', 'embed_dest', 'embed_dep_loc', 'embed_arr_loc',
'embed_dep_arr'])
```

0.4 Train and evaluate the model

[15]:
```
# Save the checkpoint:
output_dir='gs://{}/ch9/trained_model'.format(BUCKET)
os.environ['OUTDIR'] = output_dir  # needed for deployment
print('Writing trained model to {}'.format(output_dir))
```

```
Writing trained model to gs://qwiklabs-
gcp-01-c3764cc81f6b-dsongcp/ch9/trained_model
```

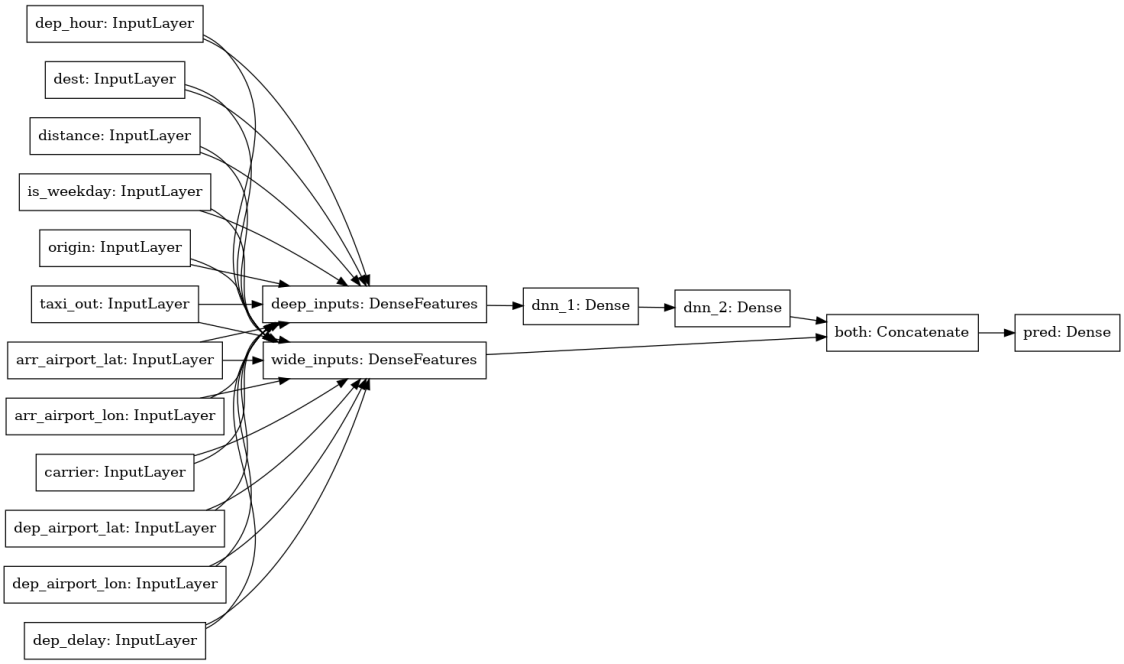
[16]:
```
# Delete the model checkpoints already present in the storage bucket:
!gsutil -m rm -rf $OUTDIR
```

```
CommandException: 1 files/objects could not be removed.
```

[17]:
```
# Build a wide-and-deep model.
def wide_and_deep_classifier(inputs, linear_feature_columns,␣
 ↪dnn_feature_columns, dnn_hidden_units):
    deep = tf.keras.layers.DenseFeatures(dnn_feature_columns,␣
 ↪name='deep_inputs')(inputs)
    layers = [int(x) for x in dnn_hidden_units.split(',')]
    for layerno, numnodes in enumerate(layers):
        deep = tf.keras.layers.Dense(numnodes, activation='relu', name='dnn_{}'.
 ↪format(layerno+1))(deep)
    wide = tf.keras.layers.DenseFeatures(linear_feature_columns,␣
 ↪name='wide_inputs')(inputs)
    both = tf.keras.layers.concatenate([deep, wide], name='both')
    output = tf.keras.layers.Dense(1, activation='sigmoid', name='pred')(both)
    model = tf.keras.Model(inputs, output)
    model.compile(optimizer='adam',
```

```
                  loss='binary_crossentropy',
                  metrics=['accuracy'])
    return model

model = wide_and_deep_classifier(
    inputs,
    linear_feature_columns = sparse.values(),
    dnn_feature_columns = real.values(),
    dnn_hidden_units = DNN_HIDDEN_UNITS)
tf.keras.utils.plot_model(model, 'flights_model.png', show_shapes=False,␣
 ↪rankdir='LR')
```

[17]:



[18]:

```
# training and evaluation dataset
train_batch_size = TRAIN_BATCH_SIZE
if DEVELOP_MODE:
    eval_batch_size = 100
    steps_per_epoch = 3
    epochs = 50
    num_eval_examples = eval_batch_size*10
else:
    eval_batch_size = 100
    steps_per_epoch = NUM_EXAMPLES // train_batch_size
    epochs = 10
    num_eval_examples = eval_batch_size * 100
train_dataset = read_dataset(training_data_uri, train_batch_size)
```

```
eval_dataset = read_dataset(validation_data_uri, eval_batch_size, tf.estimator.
  ↪ModeKeys.EVAL, num_eval_examples)
checkpoint_path = '{}/checkpoints/flights.cpt'.format(output_dir)
shutil.rmtree(checkpoint_path, ignore_errors=True)
cp_callback = tf.keras.callbacks.ModelCheckpoint(checkpoint_path,
                                                 save_weights_only=True,
                                                 verbose=1)
history = model.fit(train_dataset,
                    validation_data=eval_dataset,
                    epochs=epochs,
                    steps_per_epoch=steps_per_epoch,
                    callbacks=[cp_callback])
```

```
Epoch 1/50
3/3 [==============================] - 10s 3s/step - loss: 1.8575 - accuracy:
0.7656 - val_loss: 0.7295 - val_accuracy: 0.6385

Epoch 00001: saving model to gs://qwiklabs-
gcp-01-c3764cc81f6b-dsongcp/ch9/trained_model/checkpoints/flights.cpt
Epoch 2/50
3/3 [==============================] - 5s 3s/step - loss: 0.7876 - accuracy:
0.4844 - val_loss: 0.6990 - val_accuracy: 0.5778

Epoch 00002: saving model to gs://qwiklabs-
gcp-01-c3764cc81f6b-dsongcp/ch9/trained_model/checkpoints/flights.cpt
Epoch 3/50
3/3 [==============================] - 5s 2s/step - loss: 0.7202 - accuracy:
0.7240 - val_loss: 0.5791 - val_accuracy: 0.7945

Epoch 00003: saving model to gs://qwiklabs-
gcp-01-c3764cc81f6b-dsongcp/ch9/trained_model/checkpoints/flights.cpt
Epoch 4/50
3/3 [==============================] - 5s 2s/step - loss: 0.5792 - accuracy:
0.8021 - val_loss: 0.4814 - val_accuracy: 0.8128

Epoch 00004: saving model to gs://qwiklabs-
gcp-01-c3764cc81f6b-dsongcp/ch9/trained_model/checkpoints/flights.cpt
Epoch 5/50
3/3 [==============================] - 4s 2s/step - loss: 0.4880 - accuracy:
0.8125 - val_loss: 0.4504 - val_accuracy: 0.8547

Epoch 00005: saving model to gs://qwiklabs-
gcp-01-c3764cc81f6b-dsongcp/ch9/trained_model/checkpoints/flights.cpt
Epoch 6/50
3/3 [==============================] - 10s 5s/step - loss: 0.4705 - accuracy:
0.7917 - val_loss: 0.3994 - val_accuracy: 0.8598
```

```
Epoch 00006: saving model to gs://qwiklabs-
gcp-01-c3764cc81f6b-dsongcp/ch9/trained_model/checkpoints/flights.cpt
Epoch 7/50
3/3 [==============================] - 5s 2s/step - loss: 0.3749 - accuracy:
0.8854 - val_loss: 0.3681 - val_accuracy: 0.8582

Epoch 00007: saving model to gs://qwiklabs-
gcp-01-c3764cc81f6b-dsongcp/ch9/trained_model/checkpoints/flights.cpt
Epoch 8/50
3/3 [==============================] - 5s 2s/step - loss: 0.3520 - accuracy:
0.8542 - val_loss: 0.3544 - val_accuracy: 0.8698

Epoch 00008: saving model to gs://qwiklabs-
gcp-01-c3764cc81f6b-dsongcp/ch9/trained_model/checkpoints/flights.cpt
Epoch 9/50
3/3 [==============================] - 5s 2s/step - loss: 0.3478 - accuracy:
0.8646 - val_loss: 0.3459 - val_accuracy: 0.8763

Epoch 00009: saving model to gs://qwiklabs-
gcp-01-c3764cc81f6b-dsongcp/ch9/trained_model/checkpoints/flights.cpt
Epoch 10/50
3/3 [==============================] - 5s 3s/step - loss: 0.4070 - accuracy:
0.8229 - val_loss: 0.3329 - val_accuracy: 0.8755

Epoch 00010: saving model to gs://qwiklabs-
gcp-01-c3764cc81f6b-dsongcp/ch9/trained_model/checkpoints/flights.cpt
Epoch 11/50
3/3 [==============================] - 5s 2s/step - loss: 0.3577 - accuracy:
0.8490 - val_loss: 0.3635 - val_accuracy: 0.8582

Epoch 00011: saving model to gs://qwiklabs-
gcp-01-c3764cc81f6b-dsongcp/ch9/trained_model/checkpoints/flights.cpt
Epoch 12/50
3/3 [==============================] - 5s 2s/step - loss: 0.3561 - accuracy:
0.8750 - val_loss: 0.3186 - val_accuracy: 0.8702

Epoch 00012: saving model to gs://qwiklabs-
gcp-01-c3764cc81f6b-dsongcp/ch9/trained_model/checkpoints/flights.cpt
Epoch 13/50
3/3 [==============================] - 5s 2s/step - loss: 0.3860 - accuracy:
0.8125 - val_loss: 0.3099 - val_accuracy: 0.8751

Epoch 00013: saving model to gs://qwiklabs-
gcp-01-c3764cc81f6b-dsongcp/ch9/trained_model/checkpoints/flights.cpt
Epoch 14/50
3/3 [==============================] - 5s 2s/step - loss: 0.3037 - accuracy:
0.8646 - val_loss: 0.3639 - val_accuracy: 0.8570
```

```
Epoch 00014: saving model to gs://qwiklabs-
gcp-01-c3764cc81f6b-dsongcp/ch9/trained_model/checkpoints/flights.cpt
Epoch 15/50
3/3 [==============================] - 5s 2s/step - loss: 0.3168 - accuracy:
0.8854 - val_loss: 0.2979 - val_accuracy: 0.8973

Epoch 00015: saving model to gs://qwiklabs-
gcp-01-c3764cc81f6b-dsongcp/ch9/trained_model/checkpoints/flights.cpt
Epoch 16/50
3/3 [==============================] - 5s 3s/step - loss: 0.2573 - accuracy:
0.8958 - val_loss: 0.3173 - val_accuracy: 0.8734

Epoch 00016: saving model to gs://qwiklabs-
gcp-01-c3764cc81f6b-dsongcp/ch9/trained_model/checkpoints/flights.cpt
Epoch 17/50
3/3 [==============================] - 4s 2s/step - loss: 0.2906 - accuracy:
0.8854 - val_loss: 0.2796 - val_accuracy: 0.8964

Epoch 00017: saving model to gs://qwiklabs-
gcp-01-c3764cc81f6b-dsongcp/ch9/trained_model/checkpoints/flights.cpt
Epoch 18/50
3/3 [==============================] - 5s 2s/step - loss: 0.2949 - accuracy:
0.8906 - val_loss: 0.3319 - val_accuracy: 0.8863

Epoch 00018: saving model to gs://qwiklabs-
gcp-01-c3764cc81f6b-dsongcp/ch9/trained_model/checkpoints/flights.cpt
Epoch 19/50
3/3 [==============================] - 5s 2s/step - loss: 0.2946 - accuracy:
0.9010 - val_loss: 0.2827 - val_accuracy: 0.8897

Epoch 00019: saving model to gs://qwiklabs-
gcp-01-c3764cc81f6b-dsongcp/ch9/trained_model/checkpoints/flights.cpt
Epoch 20/50
3/3 [==============================] - 4s 2s/step - loss: 0.2519 - accuracy:
0.8958 - val_loss: 0.3133 - val_accuracy: 0.8816

Epoch 00020: saving model to gs://qwiklabs-
gcp-01-c3764cc81f6b-dsongcp/ch9/trained_model/checkpoints/flights.cpt
Epoch 21/50
3/3 [==============================] - 5s 3s/step - loss: 0.2486 - accuracy:
0.8906 - val_loss: 0.2614 - val_accuracy: 0.9092

Epoch 00021: saving model to gs://qwiklabs-
gcp-01-c3764cc81f6b-dsongcp/ch9/trained_model/checkpoints/flights.cpt
Epoch 22/50
3/3 [==============================] - 5s 2s/step - loss: 0.2917 - accuracy:
0.8958 - val_loss: 0.3318 - val_accuracy: 0.8902
```

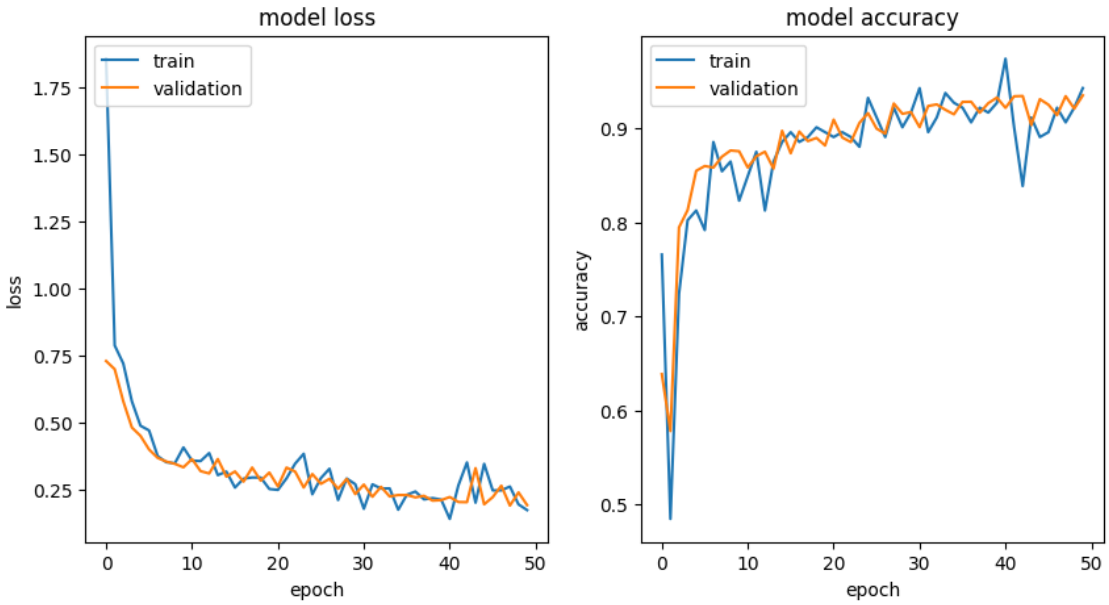
```
Epoch 00022: saving model to gs://qwiklabs-
gcp-01-c3764cc81f6b-dsongcp/ch9/trained_model/checkpoints/flights.cpt
Epoch 23/50
3/3 [==============================] - 5s 2s/step - loss: 0.3465 - accuracy:
0.8906 - val_loss: 0.3170 - val_accuracy: 0.8853

Epoch 00023: saving model to gs://qwiklabs-
gcp-01-c3764cc81f6b-dsongcp/ch9/trained_model/checkpoints/flights.cpt
Epoch 24/50
3/3 [==============================] - 5s 2s/step - loss: 0.3833 - accuracy:
0.8802 - val_loss: 0.2575 - val_accuracy: 0.9057

Epoch 00024: saving model to gs://qwiklabs-
gcp-01-c3764cc81f6b-dsongcp/ch9/trained_model/checkpoints/flights.cpt
Epoch 25/50
3/3 [==============================] - 5s 3s/step - loss: 0.2323 - accuracy:
0.9323 - val_loss: 0.3078 - val_accuracy: 0.9161

Epoch 00025: saving model to gs://qwiklabs-
gcp-01-c3764cc81f6b-dsongcp/ch9/trained_model/checkpoints/flights.cpt
Epoch 26/50
3/3 [==============================] - 5s 3s/step - loss: 0.2925 - accuracy:
0.9115 - val_loss: 0.2711 - val_accuracy: 0.8996

Epoch 00026: saving model to gs://qwiklabs-
gcp-01-c3764cc81f6b-dsongcp/ch9/trained_model/checkpoints/flights.cpt
Epoch 27/50
3/3 [==============================] - 5s 2s/step - loss: 0.3276 - accuracy:
0.8906 - val_loss: 0.2900 - val_accuracy: 0.8942

Epoch 00027: saving model to gs://qwiklabs-
gcp-01-c3764cc81f6b-dsongcp/ch9/trained_model/checkpoints/flights.cpt
Epoch 28/50
3/3 [==============================] - 5s 2s/step - loss: 0.2107 - accuracy:
0.9219 - val_loss: 0.2527 - val_accuracy: 0.9263

Epoch 00028: saving model to gs://qwiklabs-
gcp-01-c3764cc81f6b-dsongcp/ch9/trained_model/checkpoints/flights.cpt
Epoch 29/50
3/3 [==============================] - 5s 3s/step - loss: 0.2910 - accuracy:
0.9010 - val_loss: 0.2892 - val_accuracy: 0.9154

Epoch 00029: saving model to gs://qwiklabs-
gcp-01-c3764cc81f6b-dsongcp/ch9/trained_model/checkpoints/flights.cpt
Epoch 30/50
3/3 [==============================] - 5s 3s/step - loss: 0.2698 - accuracy:
0.9167 - val_loss: 0.2330 - val_accuracy: 0.9172
```

```
Epoch 00030: saving model to gs://qwiklabs-
gcp-01-c3764cc81f6b-dsongcp/ch9/trained_model/checkpoints/flights.cpt
Epoch 31/50
3/3 [==============================] - 5s 2s/step - loss: 0.1780 - accuracy:
0.9427 - val_loss: 0.2685 - val_accuracy: 0.9011

Epoch 00031: saving model to gs://qwiklabs-
gcp-01-c3764cc81f6b-dsongcp/ch9/trained_model/checkpoints/flights.cpt
Epoch 32/50
3/3 [==============================] - 5s 2s/step - loss: 0.2694 - accuracy:
0.8958 - val_loss: 0.2229 - val_accuracy: 0.9237

Epoch 00032: saving model to gs://qwiklabs-
gcp-01-c3764cc81f6b-dsongcp/ch9/trained_model/checkpoints/flights.cpt
Epoch 33/50
3/3 [==============================] - 10s 5s/step - loss: 0.2536 - accuracy:
0.9115 - val_loss: 0.2601 - val_accuracy: 0.9254

Epoch 00033: saving model to gs://qwiklabs-
gcp-01-c3764cc81f6b-dsongcp/ch9/trained_model/checkpoints/flights.cpt
Epoch 34/50
3/3 [==============================] - 5s 3s/step - loss: 0.2541 - accuracy:
0.9375 - val_loss: 0.2240 - val_accuracy: 0.9196

Epoch 00034: saving model to gs://qwiklabs-
gcp-01-c3764cc81f6b-dsongcp/ch9/trained_model/checkpoints/flights.cpt
Epoch 35/50
3/3 [==============================] - 5s 2s/step - loss: 0.1745 - accuracy:
0.9271 - val_loss: 0.2295 - val_accuracy: 0.9148

Epoch 00035: saving model to gs://qwiklabs-
gcp-01-c3764cc81f6b-dsongcp/ch9/trained_model/checkpoints/flights.cpt
Epoch 36/50
3/3 [==============================] - 5s 3s/step - loss: 0.2303 - accuracy:
0.9219 - val_loss: 0.2293 - val_accuracy: 0.9281

Epoch 00036: saving model to gs://qwiklabs-
gcp-01-c3764cc81f6b-dsongcp/ch9/trained_model/checkpoints/flights.cpt
Epoch 37/50
3/3 [==============================] - 4s 2s/step - loss: 0.2425 - accuracy:
0.9062 - val_loss: 0.2207 - val_accuracy: 0.9282

Epoch 00037: saving model to gs://qwiklabs-
gcp-01-c3764cc81f6b-dsongcp/ch9/trained_model/checkpoints/flights.cpt
Epoch 38/50
3/3 [==============================] - 4s 2s/step - loss: 0.2131 - accuracy:
0.9219 - val_loss: 0.2264 - val_accuracy: 0.9165
```

```
Epoch 00038: saving model to gs://qwiklabs-
gcp-01-c3764cc81f6b-dsongcp/ch9/trained_model/checkpoints/flights.cpt
Epoch 39/50
3/3 [==============================] - 5s 2s/step - loss: 0.2179 - accuracy:
0.9167 - val_loss: 0.2086 - val_accuracy: 0.9267

Epoch 00039: saving model to gs://qwiklabs-
gcp-01-c3764cc81f6b-dsongcp/ch9/trained_model/checkpoints/flights.cpt
Epoch 40/50
3/3 [==============================] - 5s 2s/step - loss: 0.2123 - accuracy:
0.9271 - val_loss: 0.2104 - val_accuracy: 0.9326

Epoch 00040: saving model to gs://qwiklabs-
gcp-01-c3764cc81f6b-dsongcp/ch9/trained_model/checkpoints/flights.cpt
Epoch 41/50
3/3 [==============================] - 5s 2s/step - loss: 0.1402 - accuracy:
0.9740 - val_loss: 0.2220 - val_accuracy: 0.9217

Epoch 00041: saving model to gs://qwiklabs-
gcp-01-c3764cc81f6b-dsongcp/ch9/trained_model/checkpoints/flights.cpt
Epoch 42/50
3/3 [==============================] - 5s 2s/step - loss: 0.2652 - accuracy:
0.9010 - val_loss: 0.2034 - val_accuracy: 0.9340

Epoch 00042: saving model to gs://qwiklabs-
gcp-01-c3764cc81f6b-dsongcp/ch9/trained_model/checkpoints/flights.cpt
Epoch 43/50
3/3 [==============================] - 5s 2s/step - loss: 0.3511 - accuracy:
0.8385 - val_loss: 0.2029 - val_accuracy: 0.9343

Epoch 00043: saving model to gs://qwiklabs-
gcp-01-c3764cc81f6b-dsongcp/ch9/trained_model/checkpoints/flights.cpt
Epoch 44/50
3/3 [==============================] - 4s 2s/step - loss: 0.2008 - accuracy:
0.9115 - val_loss: 0.3299 - val_accuracy: 0.9034

Epoch 00044: saving model to gs://qwiklabs-
gcp-01-c3764cc81f6b-dsongcp/ch9/trained_model/checkpoints/flights.cpt
Epoch 45/50
3/3 [==============================] - 5s 2s/step - loss: 0.3458 - accuracy:
0.8906 - val_loss: 0.1948 - val_accuracy: 0.9309

Epoch 00045: saving model to gs://qwiklabs-
gcp-01-c3764cc81f6b-dsongcp/ch9/trained_model/checkpoints/flights.cpt
Epoch 46/50
3/3 [==============================] - 4s 2s/step - loss: 0.2466 - accuracy:
0.8958 - val_loss: 0.2215 - val_accuracy: 0.9250
```

```
Epoch 00046: saving model to gs://qwiklabs-
gcp-01-c3764cc81f6b-dsongcp/ch9/trained_model/checkpoints/flights.cpt
Epoch 47/50
3/3 [==============================] - 4s 2s/step - loss: 0.2470 - accuracy:
0.9219 - val_loss: 0.2642 - val_accuracy: 0.9140

Epoch 00047: saving model to gs://qwiklabs-
gcp-01-c3764cc81f6b-dsongcp/ch9/trained_model/checkpoints/flights.cpt
Epoch 48/50
3/3 [==============================] - 5s 2s/step - loss: 0.2612 - accuracy:
0.9062 - val_loss: 0.1900 - val_accuracy: 0.9341

Epoch 00048: saving model to gs://qwiklabs-
gcp-01-c3764cc81f6b-dsongcp/ch9/trained_model/checkpoints/flights.cpt
Epoch 49/50
3/3 [==============================] - 5s 2s/step - loss: 0.1945 - accuracy:
0.9219 - val_loss: 0.2393 - val_accuracy: 0.9209

Epoch 00049: saving model to gs://qwiklabs-
gcp-01-c3764cc81f6b-dsongcp/ch9/trained_model/checkpoints/flights.cpt
Epoch 50/50
3/3 [==============================] - 5s 2s/step - loss: 0.1732 - accuracy:
0.9427 - val_loss: 0.1918 - val_accuracy: 0.9350

Epoch 00050: saving model to gs://qwiklabs-
gcp-01-c3764cc81f6b-dsongcp/ch9/trained_model/checkpoints/flights.cpt
```

[19]:
```
import matplotlib.pyplot as plt
nrows = 1
ncols = 2
fig = plt.figure(figsize=(10, 5))
for idx, key in enumerate(['loss', 'accuracy']):
    ax = fig.add_subplot(nrows, ncols, idx+1)
    plt.plot(history.history[key])
    plt.plot(history.history['val_{}'.format(key)])
    plt.title('model {}'.format(key))
    plt.ylabel(key)
    plt.xlabel('epoch')
    plt.legend(['train', 'validation'], loc='upper left');
```



0.5 Export the trained model

[20]:
```
import time
export_dir = '{}/export/flights_{}'.format(output_dir, time.
  ↪strftime("%Y%m%d-%H%M%S"))
print('Exporting to {}'.format(export_dir))
tf.saved_model.save(model, export_dir)
```

```
Exporting to gs://qwiklabs-
gcp-01-c3764cc81f6b-dsongcp/ch9/trained_model/export/flights_20230425-184304

2023-04-25 18:43:07.715159: W tensorflow/python/util/util.cc:348] Sets are not
currently considered sequences, but this may change in the future, so consider
avoiding using them.

INFO:tensorflow:Assets written to: gs://qwiklabs-gcp-01-c3764cc81f6b-dsongcp/ch9
/trained_model/export/flights_20230425-184304/assets
```

0.6 Deploy flights model to Vertex AI

[21]:
```
%%bash
# note TF_VERSION and ENDPOINT_NAME set in 1st cell
# TF_VERSION=2-6
# ENDPOINT_NAME=flights
TIMESTAMP=$(date +%Y%m%d-%H%M%S)
MODEL_NAME=${ENDPOINT_NAME}-${TIMESTAMP}
EXPORT_PATH=$(gsutil ls ${OUTDIR}/export | tail -1)
echo $EXPORT_PATH
```

```
# create the model endpoint for deploying the model
if [[ $(gcloud beta ai endpoints list --region=$REGION \
        --format='value(DISPLAY_NAME)' --filter=display_name=${ENDPOINT_NAME})␣
 ↪]]; then
    echo "Endpoint for $MODEL_NAME already exists"
else
    echo "Creating Endpoint for $MODEL_NAME"
    gcloud beta ai endpoints create --region=${REGION}␣
 ↪--display-name=${ENDPOINT_NAME}
fi
ENDPOINT_ID=$(gcloud beta ai endpoints list --region=$REGION \
              --format='value(ENDPOINT_ID)'␣
 ↪--filter=display_name=${ENDPOINT_NAME})
echo "ENDPOINT_ID=$ENDPOINT_ID"
# delete any existing models with this name
for MODEL_ID in $(gcloud beta ai models list --region=$REGION␣
 ↪--format='value(MODEL_ID)' --filter=display_name=${MODEL_NAME}); do
    echo "Deleting existing $MODEL_NAME ... $MODEL_ID "
    gcloud ai models delete --region=$REGION $MODEL_ID
done
# create the model using the parameters docker conatiner image and artifact uri
gcloud beta ai models upload --region=$REGION --display-name=$MODEL_NAME \
     --container-image-uri=us-docker.pkg.dev/vertex-ai/prediction/tf2-cpu.
 ↪${TF_VERSION}:latest \
     --artifact-uri=$EXPORT_PATH
MODEL_ID=$(gcloud beta ai models list --region=$REGION␣
 ↪--format='value(MODEL_ID)' --filter=display_name=${MODEL_NAME})
echo "MODEL_ID=$MODEL_ID"
# deploy the model to the endpoint
gcloud beta ai endpoints deploy-model $ENDPOINT_ID \
  --region=$REGION \
  --model=$MODEL_ID \
  --display-name=$MODEL_NAME \
  --machine-type=n1-standard-2 \
  --min-replica-count=1 \
  --max-replica-count=1 \
  --traffic-split=0=100
```

```
gs://qwiklabs-
gcp-01-c3764cc81f6b-dsongcp/ch9/trained_model/export/flights_20230425-184304/
Creating Endpoint for flights-20230425-184315
ENDPOINT_ID=366388060700540928
MODEL_ID=5491376684508643328
```

```
Using endpoint [https://us-central1-aiplatform.googleapis.com/]
WARNING: The following filter keys were not present in any resource :
display_name
```

```
Using endpoint [https://us-central1-aiplatform.googleapis.com/]
Waiting for operation [1087355221060878336]…
…
…
…
…
…
…
…
…
…
…
…
…
…
…
…
…
…
…
…
…
…
…
…
…
…
…done.
Created Vertex AI endpoint: projects/691390126128/locations/us-
central1/endpoints/366388060700540928.
Using endpoint [https://us-central1-aiplatform.googleapis.com/]
Using endpoint [https://us-central1-aiplatform.googleapis.com/]
WARNING: The following filter keys were not present in any resource :
display_name
Using endpoint [https://us-central1-aiplatform.googleapis.com/]
Waiting for operation [7298945007111634944]…
…
…done.
Using endpoint [https://us-central1-aiplatform.googleapis.com/]
Using endpoint [https://us-central1-aiplatform.googleapis.com/]
Waiting for operation [8668039293832265728]…
…
…
…
…
…
…
…
…
```

```
…
…
…
…
…
…
…
…
…
…
…
…
…
…
…
…
…
…
…
…
…
…
…
…
…
…
…
…
…
…
…
…
…
…
…
…
…
…
…
…done.
Deployed a model to the endpoint 366388060700540928. Id of the deployed model:
3791399766571614208.
```

Create a test input file

[22]:
```
%%writefile example_input.json
{"instances": [
```

```
 {"dep_hour": 2, "is_weekday": 1, "dep_delay": 40, "taxi_out": 17, "distance":␣
↪41, "carrier": "AS", "dep_airport_lat": 58.42527778, "dep_airport_lon": -135.
↪7075, "arr_airport_lat": 58.35472222, "arr_airport_lon": -134.57472222,␣
↪"origin": "GST", "dest": "JNU"},
 {"dep_hour": 22, "is_weekday": 0, "dep_delay": -7, "taxi_out": 7, "distance":␣
↪201, "carrier": "HA", "dep_airport_lat": 21.97611111, "dep_airport_lon":␣
↪-159.33888889, "arr_airport_lat": 20.89861111, "arr_airport_lon": -156.
↪43055556, "origin": "LIH", "dest": "OGG"}
]}
```

```
Writing example_input.json
```

Make a prediction from the model endpoint.

[23]:
```
%%bash
ENDPOINT_ID=$(gcloud beta ai endpoints list --region=$REGION \
              --format='value(ENDPOINT_ID)'␣
 ↪--filter=display_name=${ENDPOINT_NAME})
echo $ENDPOINT_ID
gcloud beta ai endpoints predict $ENDPOINT_ID --region=$REGION␣
 ↪--json-request=example_input.json
```

```
3663880607005409 28
[[0.0247025788], [0.983607709]]

Using endpoint [https://us-central1-aiplatform.googleapis.com/]
Using endpoint [https://us-central1-prediction-aiplatform.googleapis.com/]
```

Send an HTTP POST request and you will get the result back as JSON:

[24]:
```
%%bash
PROJECT=$(gcloud config get-value project)
ENDPOINT_ID=$(gcloud beta ai endpoints list --region=$REGION \
              --format='value(ENDPOINT_ID)'␣
 ↪--filter=display_name=${ENDPOINT_NAME})
curl -X POST \
  -H "Authorization: Bearer "$(gcloud auth application-default␣
 ↪print-access-token) \
  -H "Content-Type: application/json; charset=utf-8" \
  -d @example_input.json \
  "https://${REGION}-aiplatform.googleapis.com/v1/projects/${PROJECT}/locations/
 ↪${REGION}/endpoints/${ENDPOINT_ID}:predict"
```

```
{
  "predictions": [
    [
      0.0247025788
    ],
    [
```

```
      0.983607709
    ]
  ],
  "deployedModelId": "3791399766571614208",
  "model": "projects/691390126128/locations/us-
central1/models/5491376684508643328",
  "modelDisplayName": "flights-20230425-184315",
  "modelVersionId": "1"
}
```

```
Using endpoint [https://us-central1-aiplatform.googleapis.com/]
  % Total    % Received % Xferd  Average Speed   Time    Time     Time  Current
                                 Dload  Upload   Total   Spent    Left  Speed
100   835    0   292  100   543   4709   8758 --:--:-- --:--:-- --:--:-- 13467
```

0.7 Model explainability

[25]:
```
%%bash
model_dir=$(gsutil ls ${OUTDIR}/export | tail -1)
echo $model_dir
saved_model_cli show --tag_set serve --signature_def serving_default --dir␣
 ↪$model_dir
```

```
gs://qwiklabs-
gcp-01-c3764cc81f6b-dsongcp/ch9/trained_model/export/flights_20230425-184304/
The given SavedModel SignatureDef contains the following input(s):
  inputs['arr_airport_lat'] tensor_info:
      dtype: DT_FLOAT
      shape: (-1)
      name: serving_default_arr_airport_lat:0
  inputs['arr_airport_lon'] tensor_info:
      dtype: DT_FLOAT
      shape: (-1)
      name: serving_default_arr_airport_lon:0
  inputs['carrier'] tensor_info:
      dtype: DT_STRING
      shape: (-1)
      name: serving_default_carrier:0
  inputs['dep_airport_lat'] tensor_info:
      dtype: DT_FLOAT
      shape: (-1)
      name: serving_default_dep_airport_lat:0
  inputs['dep_airport_lon'] tensor_info:
      dtype: DT_FLOAT
      shape: (-1)
      name: serving_default_dep_airport_lon:0
  inputs['dep_delay'] tensor_info:
      dtype: DT_FLOAT
```

```
      shape: (-1)
      name: serving_default_dep_delay:0
  inputs['dep_hour'] tensor_info:
      dtype: DT_FLOAT
      shape: (-1)
      name: serving_default_dep_hour:0
  inputs['dest'] tensor_info:
      dtype: DT_STRING
      shape: (-1)
      name: serving_default_dest:0
  inputs['distance'] tensor_info:
      dtype: DT_FLOAT
      shape: (-1)
      name: serving_default_distance:0
  inputs['is_weekday'] tensor_info:
      dtype: DT_FLOAT
      shape: (-1)
      name: serving_default_is_weekday:0
  inputs['origin'] tensor_info:
      dtype: DT_STRING
      shape: (-1)
      name: serving_default_origin:0
  inputs['taxi_out'] tensor_info:
      dtype: DT_FLOAT
      shape: (-1)
      name: serving_default_taxi_out:0
The given SavedModel SignatureDef contains the following output(s):
  outputs['pred'] tensor_info:
      dtype: DT_FLOAT
      shape: (-1, 1)
      name: StatefulPartitionedCall_2:0
Method name is: tensorflow/serving/predict
```

[26]:
```
cols = ('dep_delay,taxi_out,distance,dep_hour,is_weekday,' +
        'dep_airport_lat,dep_airport_lon,' +
        'arr_airport_lat,arr_airport_lon,' +
        'carrier,origin,dest')
inputs = {x: {"inputTensorName": "{}".format(x)}
        for x in cols.split(',')}
expl = {
    "inputs": inputs,
    "outputs": {
    "pred": {
      "outputTensorName": "pred"
    }
  }
}
```

```
print(expl)
with open('explanation-metadata.json', 'w') as ofp:
    json.dump(expl, ofp, indent=2)
```

```
{'inputs': {'dep_delay': {'inputTensorName': 'dep_delay'}, 'taxi_out':
{'inputTensorName': 'taxi_out'}, 'distance': {'inputTensorName': 'distance'},
'dep_hour': {'inputTensorName': 'dep_hour'}, 'is_weekday': {'inputTensorName':
'is_weekday'}, 'dep_airport_lat': {'inputTensorName': 'dep_airport_lat'},
'dep_airport_lon': {'inputTensorName': 'dep_airport_lon'}, 'arr_airport_lat':
{'inputTensorName': 'arr_airport_lat'}, 'arr_airport_lon': {'inputTensorName':
'arr_airport_lon'}, 'carrier': {'inputTensorName': 'carrier'}, 'origin':
{'inputTensorName': 'origin'}, 'dest': {'inputTensorName': 'dest'}}, 'outputs':
{'pred': {'outputTensorName': 'pred'}}}
```

[27]:
```
!cat explanation-metadata.json
```

```
{
  "inputs": {
    "dep_delay": {
      "inputTensorName": "dep_delay"
    },
    "taxi_out": {
      "inputTensorName": "taxi_out"
    },
    "distance": {
      "inputTensorName": "distance"
    },
    "dep_hour": {
      "inputTensorName": "dep_hour"
    },
    "is_weekday": {
      "inputTensorName": "is_weekday"
    },
    "dep_airport_lat": {
      "inputTensorName": "dep_airport_lat"
    },
    "dep_airport_lon": {
      "inputTensorName": "dep_airport_lon"
    },
    "arr_airport_lat": {
      "inputTensorName": "arr_airport_lat"
    },
    "arr_airport_lon": {
      "inputTensorName": "arr_airport_lon"
    },
    "carrier": {
      "inputTensorName": "carrier"
    },
```

```
    "origin": {
      "inputTensorName": "origin"
    },
    "dest": {
      "inputTensorName": "dest"
    }
  },
  "outputs": {
    "pred": {
      "outputTensorName": "pred"
    }
  }
}
```

0.8 Create and deploy another model flights_xai to Vertex AI

[28]:
```
%%bash
# note TF_VERSION set in 1st cell, but ENDPOINT_NAME is being changed
# TF_VERSION=2-6
ENDPOINT_NAME=flights_xai
TIMESTAMP=$(date +%Y%m%d-%H%M%S)
MODEL_NAME=${ENDPOINT_NAME}-${TIMESTAMP}
EXPORT_PATH=$(gsutil ls ${OUTDIR}/export | tail -1)
echo $EXPORT_PATH
# create the model endpoint for deploying the model
if [[ $(gcloud beta ai endpoints list --region=$REGION \
        --format='value(DISPLAY_NAME)' --filter=display_name=${ENDPOINT_NAME})␣
 ↪]]; then
    echo "Endpoint for $MODEL_NAME already exists"
else
    # create model endpoint
    echo "Creating Endpoint for $MODEL_NAME"
    gcloud beta ai endpoints create --region=${REGION}␣
 ↪--display-name=${ENDPOINT_NAME}
fi
ENDPOINT_ID=$(gcloud beta ai endpoints list --region=$REGION \
              --format='value(ENDPOINT_ID)'␣
 ↪--filter=display_name=${ENDPOINT_NAME})
echo "ENDPOINT_ID=$ENDPOINT_ID"
# delete any existing models with this name
for MODEL_ID in $(gcloud beta ai models list --region=$REGION␣
 ↪--format='value(MODEL_ID)' --filter=display_name=${MODEL_NAME}); do
    echo "Deleting existing $MODEL_NAME ... $MODEL_ID "
    gcloud ai models delete --region=$REGION $MODEL_ID
done
# upload the model using the parameters docker conatiner image, artifact URI,␣
 ↪explanation method,
```

```
# explanation path count and explanation metadata JSON file␣
  ↪`explanation-metadata.json`.
# Here, you keep number of feature permutations to `10` when approximating the␣
  ↪Shapley values for explanation.
gcloud beta ai models upload --region=$REGION --display-name=$MODEL_NAME \
     --container-image-uri=us-docker.pkg.dev/vertex-ai/prediction/tf2-cpu.
  ↪${TF_VERSION}:latest \
     --artifact-uri=$EXPORT_PATH \
     --explanation-method=sampled-shapley --explanation-path-count=10␣
  ↪--explanation-metadata-file=explanation-metadata.json
MODEL_ID=$(gcloud beta ai models list --region=$REGION␣
  ↪--format='value(MODEL_ID)' --filter=display_name=${MODEL_NAME})
echo "MODEL_ID=$MODEL_ID"
# deploy the model to the endpoint
gcloud beta ai endpoints deploy-model $ENDPOINT_ID \
  --region=$REGION \
  --model=$MODEL_ID \
  --display-name=$MODEL_NAME \
  --machine-type=n1-standard-2 \
  --min-replica-count=1 \
  --max-replica-count=1 \
  --traffic-split=0=100
```

```
gs://qwiklabs-
gcp-01-c3764cc81f6b-dsongcp/ch9/trained_model/export/flights_20230425-184304/
Creating Endpoint for flights_xai-20230425-190300
ENDPOINT_ID=419305356322144256
MODEL_ID=3185533675294949376
```

```
Using endpoint [https://us-central1-aiplatform.googleapis.com/]
Using endpoint [https://us-central1-aiplatform.googleapis.com/]
Waiting for operation [8182776433983094784]…
…done.
Created Vertex AI endpoint: projects/691390126128/locations/us-
central1/endpoints/419305356322144256.
Using endpoint [https://us-central1-aiplatform.googleapis.com/]
Using endpoint [https://us-central1-aiplatform.googleapis.com/]
Using endpoint [https://us-central1-aiplatform.googleapis.com/]
Waiting for operation [237300791394697216]…
…
…
…
…
…
…
…
…
```

```
…
…
…
…
…
…
…
…
…
…
…
…
…
…
…
…
…
…
…
…
…
…
…
…
…
…done.
Using endpoint [https://us-central1-aiplatform.googleapis.com/]
Using endpoint [https://us-central1-aiplatform.googleapis.com/]
Waiting for operation [3789514997483175936]…
…
…
…
…
…
…
…
…
…
…
…
…
…
…
…
…
…
…
…
…
…
…
…
…
…
```

```
…
…done.
Deployed a model to the endpoint 419305356322144256. Id of the deployed model:
4363356919247667200.
```

[29]:
```
%%bash
PROJECT=$(gcloud config get-value project)
ENDPOINT_NAME=flights_xai
ENDPOINT_ID=$(gcloud beta ai endpoints list --region=$REGION \
              --format='value(ENDPOINT_ID)'␣
 ↪--filter=display_name=${ENDPOINT_NAME})
curl -X POST \
  -H "Authorization: Bearer "$(gcloud auth application-default␣
 ↪print-access-token) \
  -H "Content-Type: application/json; charset=utf-8" \
  -d @example_input.json \
  "https://${REGION}-aiplatform.googleapis.com/v1/projects/${PROJECT}/locations/
 ↪${REGION}/endpoints/${ENDPOINT_ID}:explain"
```

```
{
  "error": {
    "code": 400,
    "message": "\"Explainability failed with exception: Exceeded max_retries (5)
while Explainer attempting to call Predictor. Error: \u003c_InactiveRpcError of
RPC that terminated with:\\n\\tstatus =
StatusCode.RESOURCE_EXHAUSTED\\n\\tdetails = \\\"\\\"\\n\\tdebug_error_string =
\\\"UNKNOWN:Error received from peer ipv4:10.60.9.2:8500
{created_time:\\\"2023-04-25T19:17:28.014529189+00:00\\\", grpc_status:8,
grpc_message:\\\"\\\"}\\\"\\n\u003e\"\n",
    "status": "FAILED_PRECONDITION"
  }
}
```

```
Using endpoint [https://us-central1-aiplatform.googleapis.com/]
  % Total    % Received % Xferd  Average Speed   Time    Time     Time  Current
                                 Dload  Upload   Total   Spent    Left  Speed
100  1071    0   528  100   543    213    219  0:00:02  0:00:02 --:--:--   432
```

[ ]:
```

```