SEP 720 – Cloud Computing: Assignment 4

End-to-end Machine Learning with TensorFlow on GCP

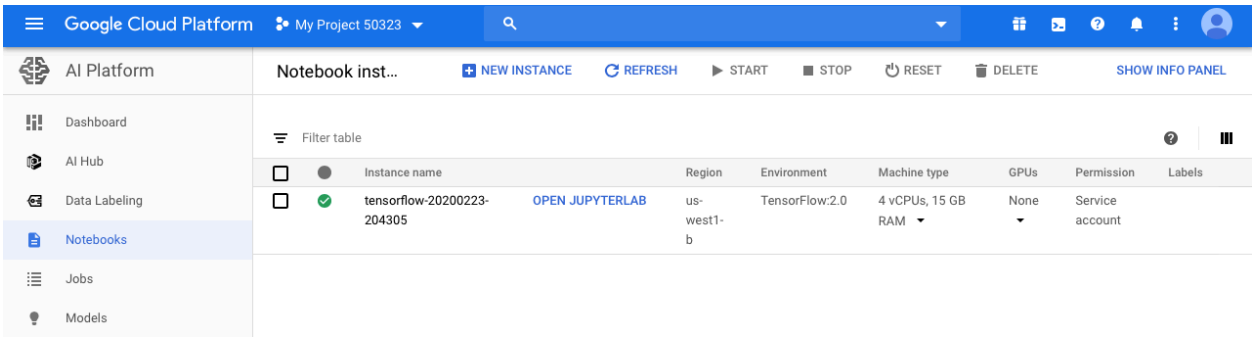
Submitted by,

Greeshma Gopal(gopalg)

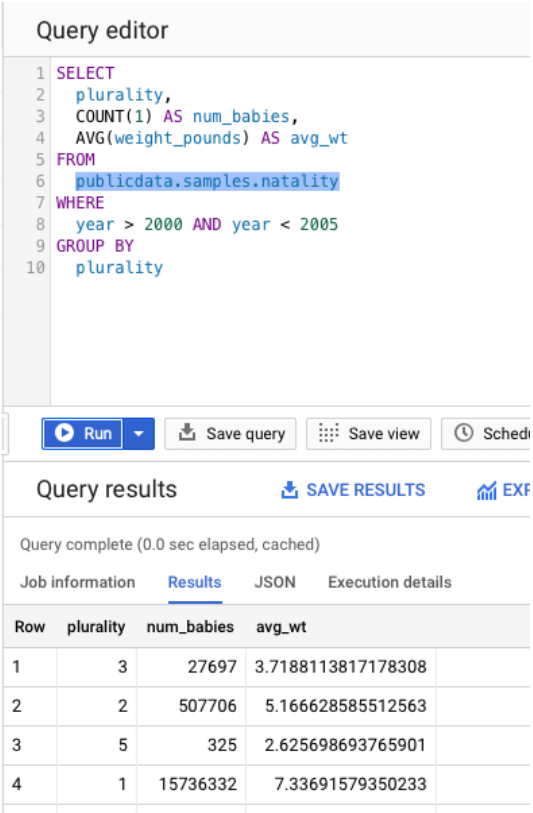
ID- 400245291

STEP 1: DATA EXPLORATION

- Creating a new notebook instance with TensorFlow 2.1 without GPU's



- Running query to explore the data in big query



How many triplets were born in the US between 2000 and 2005? **27697**

- Exploring the data in jupyter by loading the first 100 rows. The results are stored as a dataframe.

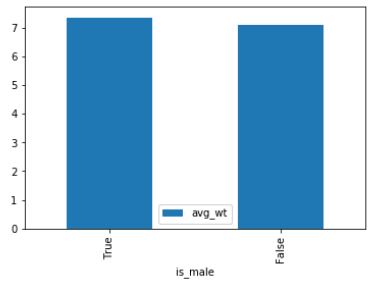
[1]:
```
query="""
SELECT
  weight_pounds,
  is_male,
  mother_age,
  plurality,
  gestation_weeks
FROM
  publicdata.samples.natality
WHERE year > 2000
"""
from google.cloud import bigquery
df = bigquery.Client().query(query + " LIMIT 100").to_dataframe()
df.head()
```

[1]:

	weight_pounds	is_male	mother_age	plurality	gestation_weeks
0	8.818490	False	17	1	42
1	8.141671	False	29	1	38
2	5.948072	True	38	1	38
3	8.838332	True	27	1	39
4	9.259415	True	28	1	38

- Fetching the average weight and the number of babies. In this step we have a function get_distinct_values within which the query for average weight has been defined. The values which are fetched are being plotted as a bar chart.

```
[2]: def get_distinct_values(column_name):
       sql = """
SELECT
  {0},
  COUNT(1) AS num_babies,
  AVG(weight_pounds) AS avg_wt
FROM
  publicdata.samples.natality
WHERE
  year > 2000
GROUP BY
  {0}
  """.format(column_name)
  return bigquery.Client().query(sql).to_dataframe()

df = get_distinct_values('is_male')
df.plot(x='is_male', y='avg_wt', kind='bar');
```

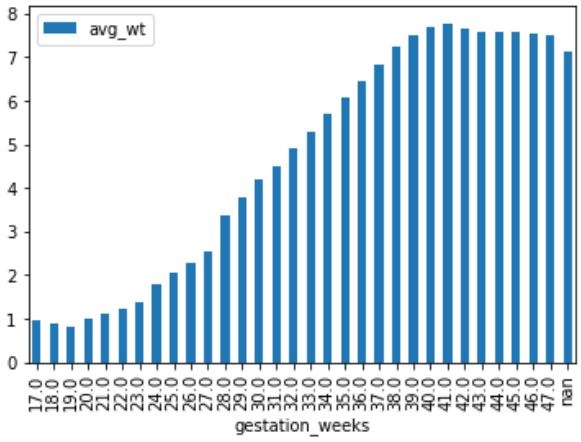


Are male babies heavier or lighter than female babies? Did you know this? **Yes!**

Is the sex of the baby a good feature to use in our machine learning model? **Yes. This is a good feature for the model**

- Fetching the gestation period using the method defined earlier. The data is sorted and used for plotting the graph against average weight of the babies.

```
df = get_distinct_values('gestation_weeks')
df = df.sort_values('gestation_weeks')
df.plot(x='gestation_weeks', y='avg_wt', kind='bar');
```



Is `gestation_weeks` a good feature to use in our machine learning model? **Yes**

Is gestation_weeks always available? **It is available in most of the cases**

Compare the variability of birth weight due to sex of baby and due to gestation weeks. Which factor do you think is more important for accurate weight prediction? **Birth weight due to sex seems to be an important parameter when compared to the weight due to gestation weeks since this might not be available in all cases.**

STEP 2: CREATING A SAMPLED DATASET

- Cloning the GITHUB dataset by giving the below path

  https://github.com/GoogleCloudPlatform/training-data-analyst/

- Changing the bucket to desired one.

```
BUCKET = 'greeshmagopal'
PROJECT = 'crack-map-265614'
REGION = 'us-west1-b'
```

```
import os
os.environ['BUCKET'] = BUCKET
os.environ['PROJECT'] = PROJECT
os.environ['REGION'] = REGION
```

```
%%bash
if ! gsutil ls | grep -q gs://${BUCKET}/; then
  gsutil mb -l ${REGION} gs://${BUCKET}
fi
```

- Fetching the data which is after the year 2000. BigQuery but GROUP BY the hashmonth and see number of records for each group to enable us to get the correct train and evaluation percentages

```
df = bigquery.Client().query("SELE
print("There are {} unique hashmon
df.head()
```

```
There are 96 unique hashmonths.
```

	hashmonth	num_babies
0	8904940584331855459	344191
1	-2126480030009879160	344357
2	-1525201076796226340	303664
3	6691862025345277042	338820
4	5934265245228309013	324598

19]:
```
# Added the RAND() so that we can now subsample from each of the hashmonths to get approximately the record
trainQuery = "SELECT * FROM (" + query + ") WHERE ABS(MOD(hashmonth, 4)) < 3 AND RAND() < 0.0005"
evalQuery = "SELECT * FROM (" + query + ") WHERE ABS(MOD(hashmonth, 4)) = 3 AND RAND() < 0.0005"
traindf = bigquery.Client().query(trainQuery).to_dataframe()
evaldf = bigquery.Client().query(evalQuery).to_dataframe()
print("There are {} examples in the train dataset and {} in the eval dataset".format(len(traindf), len(evalc
```

```
There are 13190 examples in the train dataset and 3368 in the eval dataset
```

]:
```
traindf.head()
```

]:

	weight_pounds	is_male	mother_age	plurality	gestation_weeks	hashmonth
0	5.676903	False	34	1	42.0	3095933535584005890
1	8.624484	True	24	1	41.0	3095933535584005890
2	9.124933	True	25	1	36.0	3095933535584005890
3	8.126239	True	23	1	35.0	3095933535584005890
4	7.941051	True	27	1	39.0	3095933535584005890

```
traindf.describe()
```

	weight_pounds	mother_age	plurality	gestation_weeks	hashmonth
count	13280.000000	13292.000000	13292.000000	13194.000000	1.329200e+04
mean	7.248088	27.377219	1.033704	38.661513	3.152577e+17
std	1.304056	6.228287	0.190217	2.561727	5.204560e+18
min	0.500449	13.000000	1.000000	18.000000	-9.183606e+18
25%	6.573082	22.000000	1.000000	38.000000	-3.340563e+18
50%	7.312733	27.000000	1.000000	39.000000	-3.280124e+17
75%	8.062305	32.000000	1.000000	40.000000	4.331750e+18
max	12.125424	50.000000	5.000000	47.000000	8.599690e+18

```
traindf.head()
traindf = preprocess(traindf)
evaldf = preprocess(evaldf)
traindf.head()
```

	weight_pounds	is_male	mother_age	plurality	gestation_weeks	hashmonth
0	5.676903	False	34	Single(1)	42.0	3095933535584005890
1	8.624484	True	24	Single(1)	41.0	3095933535584005890
2	9.124933	True	25	Single(1)	36.0	3095933535584005890
3	8.126239	True	23	Single(1)	35.0	3095933535584005890
4	7.941051	True	27	Single(1)	39.0	3095933535584005890

```
traindf.tail()
```

	weight_pounds	is_male	mother_age	plurality	gestation_weeks	hashmonth
13287	10.937133	Unknown	33	Single(1)	40.0	-774501970389208065
13288	7.297301	Unknown	35	Single(1)	39.0	-774501970389208065
13289	2.248715	Unknown	42	Single(1)	31.0	-774501970389208065
13290	7.061406	Unknown	29	Single(1)	39.0	-774501970389208065
13291	7.749249	Unknown	36	Single(1)	38.0	-774501970389208065

```
traindf.to_csv('train.csv', index=False, header=False)
evaldf.to_csv('eval.csv', index=False, header=False)
```

```
%%bash
wc -l *.csv
head *.csv
tail *.csv
```

```
   6466 eval.csv
  26374 train.csv
  32840 total
==> eval.csv <==
6.9225150268,False,35,Single(1),42.0,6392072535155213407
8.18796841068,False,35,Single(1),37.0,-6244544205302024223
8.5208664263,True,18,Single(1),44.0,2246942437170405963
6.4815905028,False,36,Single(1),42.0,-6782146986770280327
7.06361087448,False,30,Single(1),42.0,1569531340167098963
7.81318256528,True,36,Single(1),38.0,-1866590652208008467
8.3555197298,True,20,Single(1),38.0,1569531340167098963
7.29950549482,True,30,Single(1),37.0,1088037545023002395
6.37576861704,True,30,Single(1),41.0,3182182455926341111
7.64783586878,True,23,Single(1),38.0,-7146494315947640619

==> train.csv <==
5.6769032465,False,34,Single(1),42.0,3095933535584005890
8.62448368944,True,24,Single(1),41.0,3095933535584005890
9.12493302418,True,25,Single(1),36.0,3095933535584005890
8.12623897732,True,23,Single(1),35.0,3095933535584005890
7.94105067724,True,27,Single(1),39.0,3095933535584005890
7.7492485093,False,23,Single(1),40.0,3095933535584005890
7.87491199864,True,35,Single(1),41.0,3095933535584005890
8.375361333379999,False,28,Single(1),40.0,30959335355840058 90
7.68751907594,False,35,Single(1),40.0,3095933535584005890
```

STEP 3: Create Keras DNN and wide-and-deep model

- Assigning labels and columns. Is_male and plurality would be strings

```
import shutil
import numpy as np
import tensorflow as tf
print(tf.__version__)
```

```
2.1.0
```

```
CSV_COLUMNS = 'weight_pounds,is_male,mother_age,plurality,gestation_weeks,key'.split(',')
LABEL_COLUMN = 'weight_pounds'
KEY_COLUMN = 'key'

DEFAULTS = [[0.0], ['null'], [0.0], ['null'], [0.0], ['nokey']]
```

- Defining two methods for labels/features and for loading the dataset

```
def features_and_labels(row_data):
    for unwanted_col in ['key']:
        row_data.pop(unwanted_col)
    label = row_data.pop(LABEL_COLUMN)
    return row_data, label

def load_dataset(pattern, batch_size=1, mode=tf.estimator.ModeKeys.EVAL):
  dataset = (tf.data.experimental.make_csv_dataset(pattern, batch_size, CSV_COLUMNS, DEFAULTS)
             .map(features_and_labels)
             )
  if mode == tf.estimator.ModeKeys.TRAIN:
        dataset = dataset.shuffle(1000).repeat()
  dataset = dataset.prefetch(1)
  return dataset
```

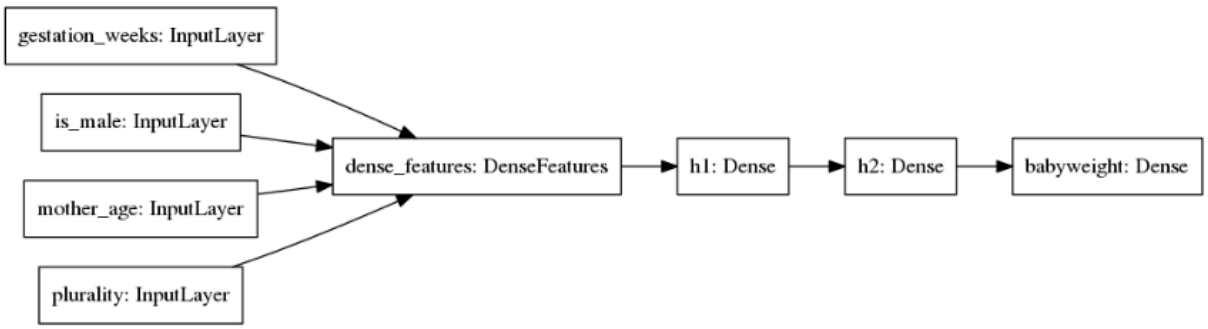
- Below are the details of the DNN architecture: -
    - We have three layers in total which includes the output layer
    - 64 neurons in the first layer, 32 neurons in the next layer and since it is a classification issue, we have only one in the last output layer
    - For the first two hidden layers we are using the activation layer relu, while the last one is linear.

```
Layer (type)                 Output Shape              Param #   Connected to
==================================================================================================
gestation_weeks (InputLayer) [(None,)]                 0
__________________________________________________________________________________________________
is_male (InputLayer)         [(None,)]                 0
__________________________________________________________________________________________________
mother_age (InputLayer)      [(None,)]                 0
__________________________________________________________________________________________________
plurality (InputLayer)       [(None,)]                 0
__________________________________________________________________________________________________
dense_features (DenseFeatures) (None, 2)               0         gestation_weeks[0][0]
                                                                 is_male[0][0]
                                                                 mother_age[0][0]
                                                                 plurality[0][0]
__________________________________________________________________________________________________
h1 (Dense)                   (None, 64)                192       dense_features[0][0]
__________________________________________________________________________________________________
h2 (Dense)                   (None, 32)                2080      h1[0][0]
__________________________________________________________________________________________________
babyweight (Dense)           (None, 1)                 33        h2[0][0]
==================================================================================================
Total params: 2,305
Trainable params: 2,305
Non-trainable params: 0
__________________________________________________________________________________________________
```

- The entire flow can be viewed using Keras plot_model

```
]: tf.keras.utils.plot_model(model, 'dnn_model.png', show_shapes=False, rankdir='LR')
```

]:

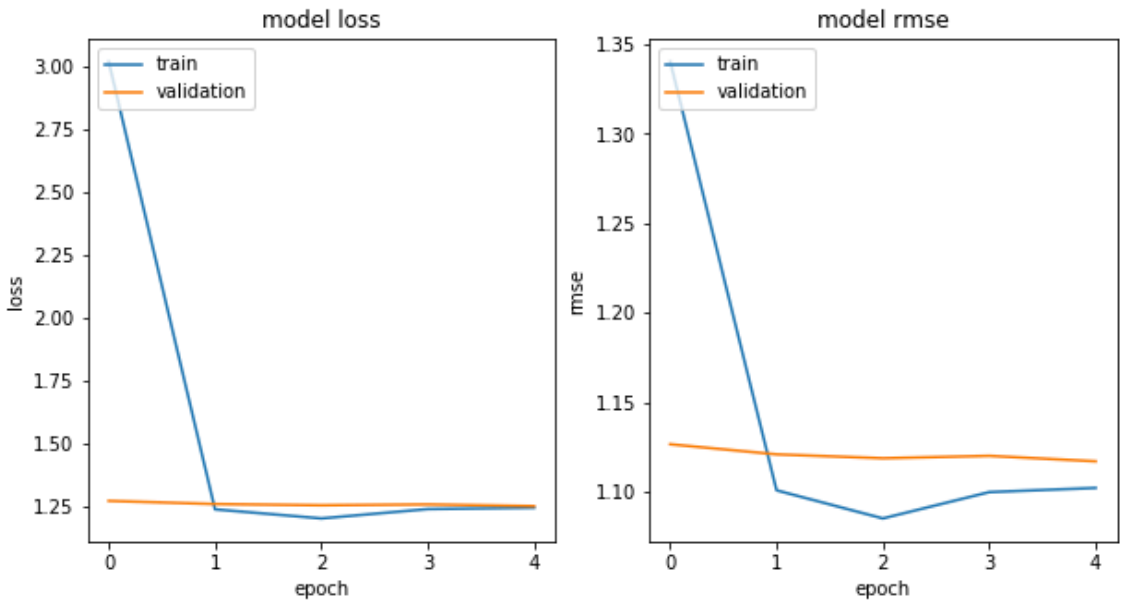


- We are running the training and validation data with 5 epochs. The metrics used include mse and rmse.

[10]:
```
TRAIN_BATCH_SIZE = 32
NUM_TRAIN_EXAMPLES = 10000 * 5 # training dataset repeats, so it will wrap around
NUM_EVALS = 5  # how many times to evaluate
NUM_EVAL_EXAMPLES = 10000 # enough to get a reasonable sample, but not so much that it slows down

trainds = load_dataset('train*', TRAIN_BATCH_SIZE, tf.estimator.ModeKeys.TRAIN)
evalds = load_dataset('eval*', 1000, tf.estimator.ModeKeys.EVAL).take(NUM_EVAL_EXAMPLES//1000)

steps_per_epoch = NUM_TRAIN_EXAMPLES // (TRAIN_BATCH_SIZE * NUM_EVALS)

history = model.fit(trainds,
                    validation_data=evalds,
                    epochs=NUM_EVALS,
                    steps_per_epoch=steps_per_epoch)
```

```
Train for 312 steps, validate for 10 steps
Epoch 1/5
312/312 [==============================] - 5s 17ms/step - loss: 3.0181 - rmse: 1.3404 - mse: 3.0181 - val_l
oss: 1.2697 - val_rmse: 1.1266 - val_mse: 1.2697
Epoch 2/5
312/312 [==============================] - 2s 8ms/step - loss: 1.2360 - rmse: 1.1009 - mse: 1.2360 - val_lo
ss: 1.2571 - val_rmse: 1.1210 - val_mse: 1.2571
Epoch 3/5
312/312 [==============================] - 3s 9ms/step - loss: 1.1998 - rmse: 1.0852 - mse: 1.1998 - val_lo
ss: 1.2523 - val_rmse: 1.1188 - val_mse: 1.2523
Epoch 4/5
312/312 [==============================] - 2s 6ms/step - loss: 1.2371 - rmse: 1.0999 - mse: 1.2371 - val_lo
ss: 1.2556 - val_rmse: 1.1201 - val_mse: 1.2556
Epoch 5/5
312/312 [==============================] - 3s 10ms/step - loss: 1.2424 - rmse: 1.1022 - mse: 1.2424 - val_l
oss: 1.2488 - val_rmse: 1.1171 - val_mse: 1.2488
```

- The model loss and mse VS epochs was plotted using matplotlib.



- Saving the data

```
import shutil, os, datetime
OUTPUT_DIR = './export/babyweight'
shutil.rmtree(OUTPUT_DIR, ignore_errors=True)
EXPORT_PATH = os.path.join(OUTPUT_DIR, datetime.datetime.now().strftime('%Y%m%d%H%M%S'))
tf.saved_model.save(model, EXPORT_PATH, signatures={'serving_default': my_serve})
print("Exported trained model to {}".format(EXPORT_PATH))
os.environ['EXPORT_PATH'] = EXPORT_PATH
```

```
WARNING:tensorflow:From /usr/local/lib/python3.5/dist-packages/tensorflow_core/python/ops/resource_variable
_ops.py:1786: calling BaseResourceVariable.__init__ (from tensorflow.python.ops.resource_variable_ops) with
constraint is deprecated and will be removed in a future version.
Instructions for updating:
If using Keras pass *_constraint arguments to layers.
INFO:tensorflow:Assets written to: ./export/babyweight/20200224033118/assets
Exported trained model to ./export/babyweight/20200224033118
```

```
!find $EXPORT_PATH
```

```
./export/babyweight/20200224033118
./export/babyweight/20200224033118/variables
./export/babyweight/20200224033118/variables/variables.data-00000-of-00001
./export/babyweight/20200224033118/variables/variables.index
./export/babyweight/20200224033118/assets
./export/babyweight/20200224033118/saved_model.pb
```

- Deploy trained model to Cloud AI Platform

```
[15]: !saved_model_cli show --tag_set serve --signature_def serving_default --dir {EXPORT_PATH}
```

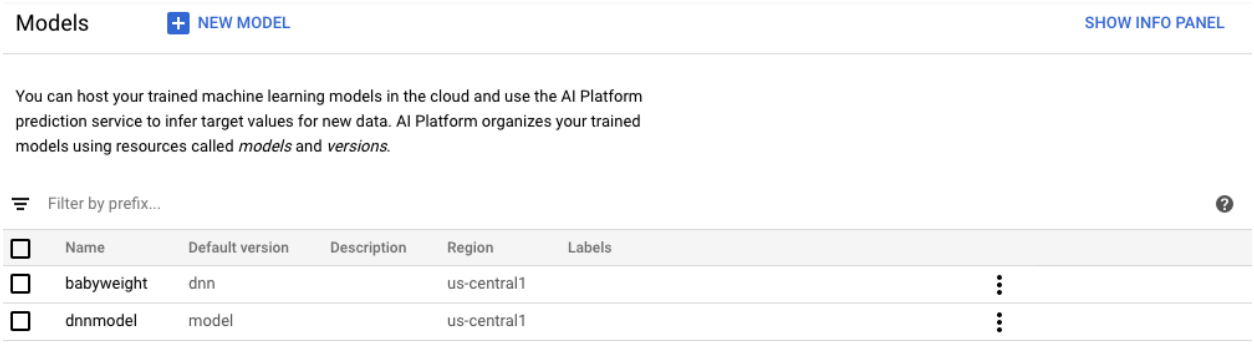
```
The given SavedModel SignatureDef contains the following input(s):
  inputs['gestation_weeks'] tensor_info:
      dtype: DT_FLOAT
      shape: (-1)
      name: serving_default_gestation_weeks:0
  inputs['is_male'] tensor_info:
      dtype: DT_STRING
      shape: (-1)
      name: serving_default_is_male:0
  inputs['key'] tensor_info:
      dtype: DT_STRING
      shape: (-1)
      name: serving_default_key:0
  inputs['mother_age'] tensor_info:
      dtype: DT_FLOAT
      shape: (-1)
      name: serving_default_mother_age:0
  inputs['plurality'] tensor_info:
      dtype: DT_STRING
      shape: (-1)
      name: serving_default_plurality:0
The given SavedModel SignatureDef contains the following output(s):
  outputs['babyweight'] tensor_info:
      dtype: DT_FLOAT
      shape: (-1, 1)
      name: StatefulPartitionedCall:0
  outputs['key'] tensor_info:
      dtype: DT_STRING
      shape: (-1)
      name: StatefulPartitionedCall:1
Method name is: tensorflow/serving/predict
```

- Creating a dnn model to deploy in the cloud platform

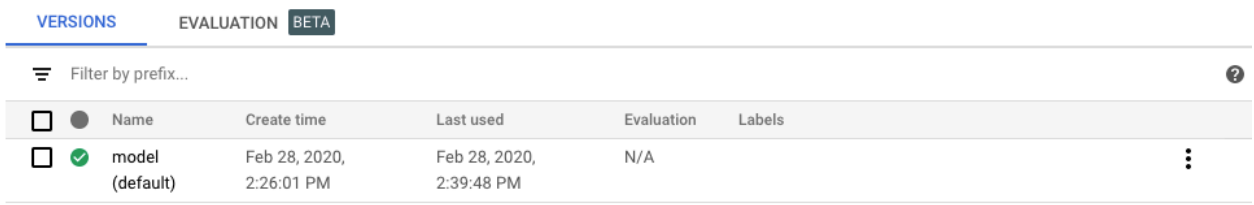
```
    echo "Please run this cell again if you don't see a Creating message ... "
    sleep 2
fi

# create model
echo "Creating $MODEL_NAME:$VERSION_NAME"
gcloud ai-platform versions create --model=$MODEL_NAME $VERSION_NAME --async \
       --framework=tensorflow --python-version=3.5 --runtime-version=1.14 \
       --origin=$MODEL_LOCATION --staging-bucket=gs://$BUCKET
```

```
Deleting and deploying dnnmodel  from ./export/babyweight/20200228192529 ... this will take
a few minutes
Creating dnnmodel model now.
Creating dnnmodel:model
Created ml engine model [projects/crack-map-265614/models/dnnmodel].
```

- The models which were created will be displayed in the model menu.



- A version of model was created which has the settings of which python version and tensorflow framework must be used.




```
!gcloud ai-platform predict --model babyweight --json-instances input.json --version dnn
```

```
BABYWEIGHT          KEY
[6.988781452178955] b1
[7.48026704788208]  b2
[6.988781452178955] g1
[7.48026704788208]  g2
```

```
parent = 'projects/%s/models/%s/versions/%s' % (project, model_name, version_name)
prediction = api.projects().predict(body=input_data, name=parent).execute()
print(prediction)
print(prediction['predictions'][0]['babyweight'][0])
```

```
{'predictions': [{'key': 'b1', 'babyweight': [7.436848163604736]}, {'key': 'g1', 'babyweigh
t': [7.348789215087891]}, {'key': 'b2', 'babyweight': [7.436848163604736]}, {'key': 'u1', 'b
abyweight': [7.348789215087891]}]}
7.436848163604736
```

Create Keras Wide-and-Deep model

[1]:
```
BUCKET = 'greeshmagopal'
PROJECT = 'crack-map-265614'
REGION = 'us-central1'
```

[2]:
```
import os
os.environ['BUCKET'] = BUCKET
os.environ['PROJECT'] = PROJECT
os.environ['REGION'] = REGION
```

[3]:
```
%%bash
if ! gsutil ls | grep -q gs://${BUCKET}/; then
  gsutil mb -l ${REGION} gs://${BUCKET}
fi
```

[4]:
```
%%bash
ls *.csv
```

```
eval.csv
train.csv
```

- Write an input_fn to read the data

[5]:
```
import shutil
import numpy as np
import tensorflow as tf
print(tf.__version__)
```

```
2.1.0
```

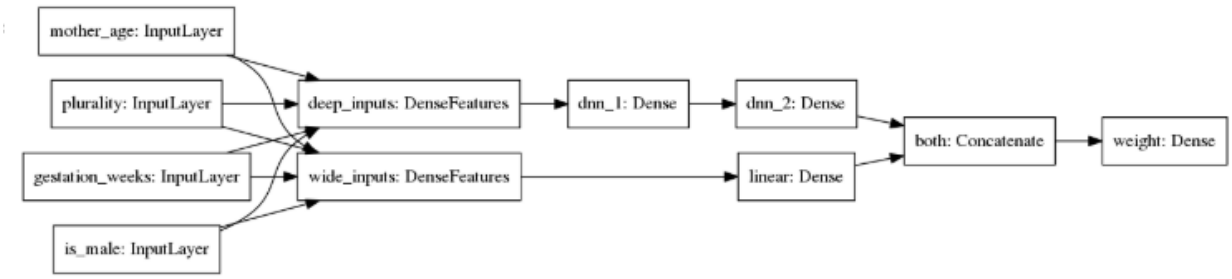
[6]:
```
# Determine CSV, label, and key columns
CSV_COLUMNS = 'weight_pounds,is_male,mother_age,plurality,gestation_weeks,key'.split(',')
LABEL_COLUMN = 'weight_pounds'
KEY_COLUMN = 'key'

# Set default values for each CSV column. Treat is_male and plurality as strings.
DEFAULTS = [[0.0], ['null'], [0.0], ['null'], [0.0], ['nokey']]
```

[7]:
```
def features_and_labels(row_data):
    for unwanted_col in ['key']:
        row_data.pop(unwanted_col)
    label = row_data.pop(LABEL_COLUMN)
    return row_data, label  # features, label

# load the training data
def load_dataset(pattern, batch_size=1, mode=tf.estimator.ModeKeys.EVAL):
  dataset = (tf.data.experimental.make_csv_dataset(pattern, batch_size, CSV_COLUMNS, DEFAULTS
             .map(features_and_labels) # features, label
             )
  if mode == tf.estimator.ModeKeys.TRAIN:
        dataset = dataset.shuffle(1000).repeat()
  dataset = dataset.prefetch(1) # take advantage of multi-threading; 1=AUTOTUNE
  return dataset
```

- Defining the feature columns. mother_age and gestation_weeks as numeric. The others (is_male, plurality) should be categorical. Below is the screenshot of the mode outline.

```
_____
mother_age (InputLayer)         [(None,)]            0
_____
plurality (InputLayer)          [(None,)]            0
_____
deep_inputs (DenseFeatures)     (None, 5)            60000       gestation_weeks[0][0]
                                                                 is_male[0][0]
                                                                 mother_age[0][0]
                                                                 plurality[0][0]
_____
dnn_1 (Dense)                   (None, 64)           384         deep_inputs[0][0]
_____
wide_inputs (DenseFeatures)     (None, 71)           0           gestation_weeks[0][0]
                                                                 is_male[0][0]
                                                                 mother_age[0][0]
                                                                 plurality[0][0]
_____
dnn_2 (Dense)                   (None, 32)           2080        dnn_1[0][0]
_____
linear (Dense)                  (None, 10)           720         wide_inputs[0][0]
_____
both (Concatenate)              (None, 42)           0           dnn_2[0][0]
                                                                 linear[0][0]
_____
weight (Dense)                  (None, 1)            43          both[0][0]
```

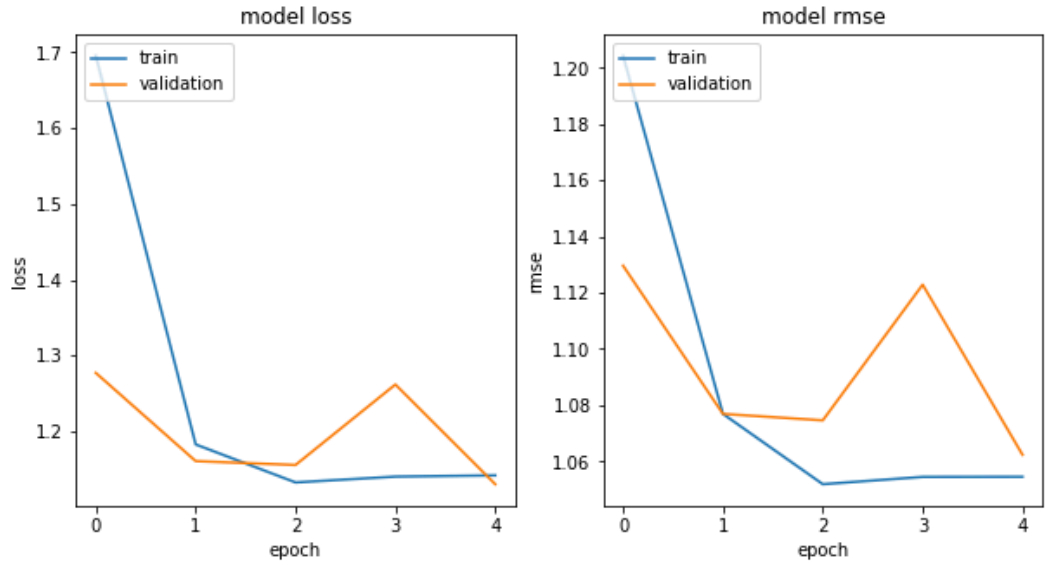
- Visualizing the DNN using the Keras plot_model utility.



- Training the model and evaluating with 5 epochs

```
Train for 312 steps, validate for 10 steps
Epoch 1/5
312/312 [==============================] - 6s 18ms/step - loss: 1.6956 - rmse: 1.2043 - mse:
1.6956 - val_loss: 1.2770 - val_rmse: 1.1296 - val_mse: 1.2770
Epoch 2/5
312/312 [==============================] - 3s 9ms/step - loss: 1.1826 - rmse: 1.0769 - mse:
1.1826 - val_loss: 1.1605 - val_rmse: 1.0769 - val_mse: 1.1605
Epoch 3/5
312/312 [==============================] - 4s 11ms/step - loss: 1.1324 - rmse: 1.0519 - mse:
1.1324 - val_loss: 1.1555 - val_rmse: 1.0746 - val_mse: 1.1555
Epoch 4/5
312/312 [==============================] - 3s 10ms/step - loss: 1.1401 - rmse: 1.0545 - mse:
1.1401 - val_loss: 1.2617 - val_rmse: 1.1229 - val_mse: 1.2617
Epoch 5/5
312/312 [==============================] - 3s 10ms/step - loss: 1.1417 - rmse: 1.0545 - mse:
1.1417 - val_loss: 1.1301 - val_rmse: 1.0624 - val_mse: 1.1301
```

- Plotting the loss curve of training and validation data.



- Saving the data

[12]:
```
import shutil, os, datetime
OUTPUT_DIR = 'babyweight_trained'
shutil.rmtree(OUTPUT_DIR, ignore_errors=True)
EXPORT_PATH = os.path.join(OUTPUT_DIR, datetime.datetime.now().strftime('%Y%m%d%H%M%S'))
tf.saved_model.save(model, EXPORT_PATH) # with default serving function
print("Exported trained model to {}".format(EXPORT_PATH))
```

```
WARNING:tensorflow:From /usr/local/lib/python3.5/dist-packages/tensorflow_core/python/ops/re
source_variable_ops.py:1786: calling BaseResourceVariable.__init__ (from tensorflow.python.o
ps.resource_variable_ops) with constraint is deprecated and will be removed in a future vers
ion.
Instructions for updating:
If using Keras pass *_constraint arguments to layers.
INFO:tensorflow:Assets written to: babyweight_trained/20200228211127/assets
Exported trained model to babyweight_trained/20200228211127
```

[13]:
```
!ls $EXPORT_PATH
```

```
assets  saved_model.pb  variables
```

STEP 4: Preprocessing using Cloud Dataflow

- Creating datasets for Machine Learning using Dataflow – Using Apache beam for better preprocessing

```
import apache_beam as beam
print(beam.__version__)
```

```
2.19.0
```

- Running to query to fetch the natality data after the year 2000

```
# Create SQL query using natality data after the year 2000
query = """
SELECT
  weight_pounds,
  is_male,
  mother_age,
  plurality,
  gestation_weeks,
  FARM_FINGERPRINT(CONCAT(CAST(YEAR AS STRING), CAST(month AS STRING))) AS hashmonth
FROM
  publicdata.samples.natality
WHERE year > 2000
"""
```

```
# Call BigQuery and examine in dataframe
from google.cloud import bigquery
df = bigquery.Client().query(query + " LIMIT 100").to_dataframe()
df.head()
```

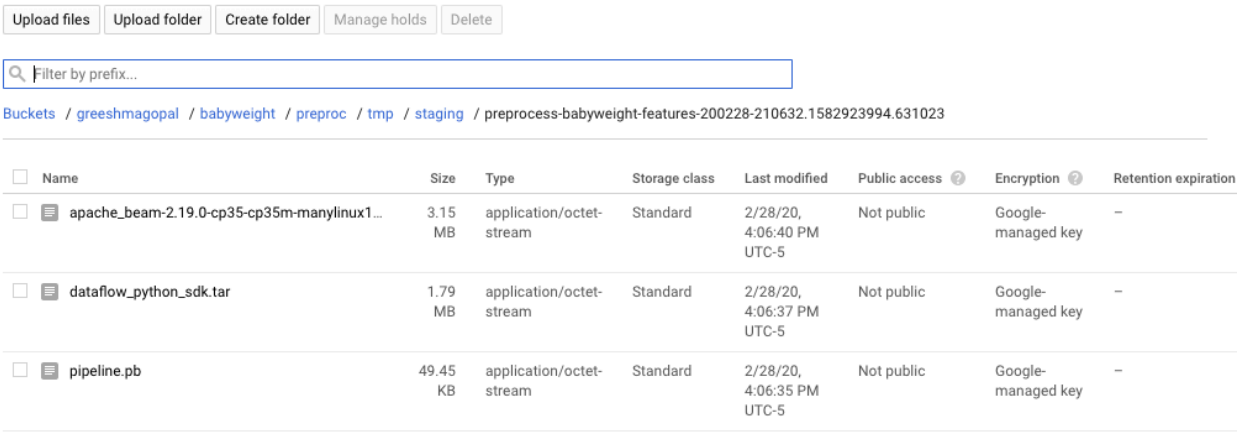
	weight_pounds	is_male	mother_age	plurality	gestation_weeks	hashmonth
0	7.063611	True	32	1	37.0	7108882242435606404
1	4.687028	True	30	3	33.0	-7170969733900686954
2	7.561856	True	20	1	39.0	6392072535155213407
3	7.561856	True	31	1	37.0	-2126480030009879160
4	7.312733	True	32	1	40.0	3408502330831153141

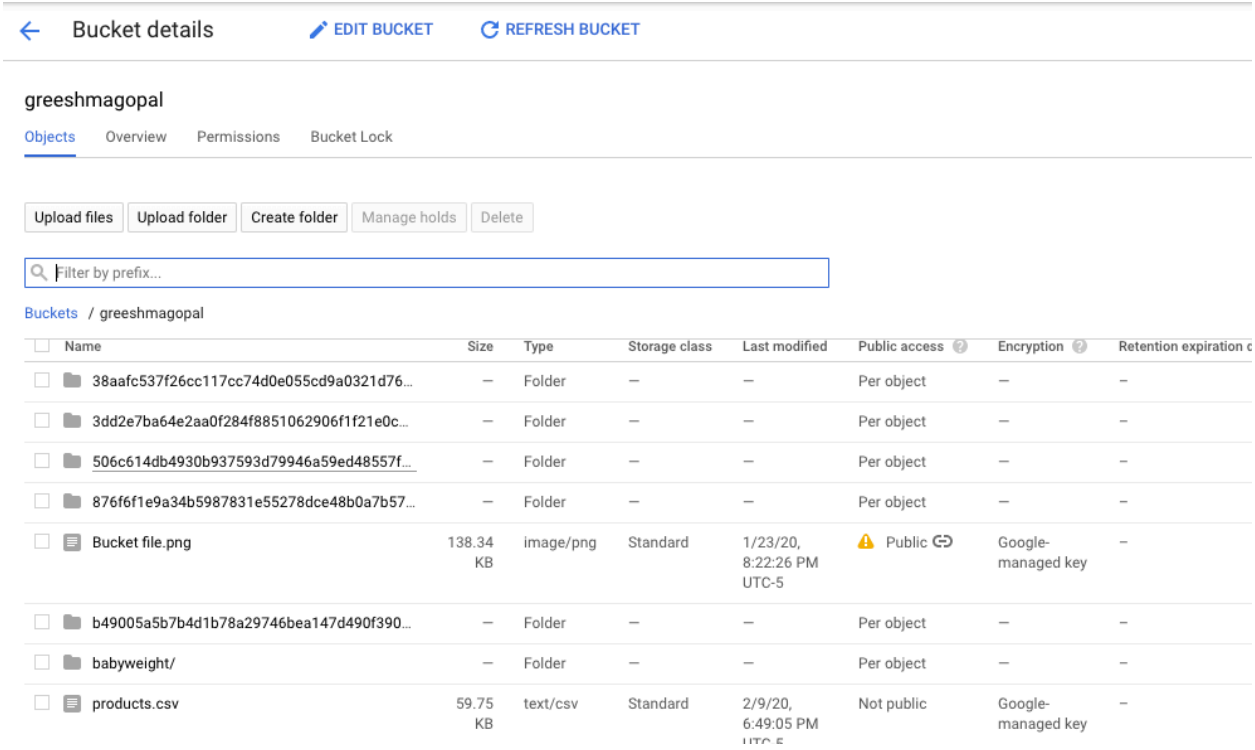
- Create ML dataset using Dataflow

```
    (p
     | '{}_read'.format(step) >> beam.io.Read(beam.io.BigQuerySource(query = selquery, use_standard_sql = Tr
     | '{}_csv'.format(step) >> beam.FlatMap(to_csv)
     | '{}_out'.format(step) >> beam.io.Write(beam.io.WriteToText(os.path.join(OUTPUT_DIR, '{}.csv'.format(s
    )

  job = p.run()
  if in_test_mode:
    job.wait_until_finish()
    print("Done!")

preprocess(in_test_mode = False)
```

```
Launching Dataflow job preprocess-babyweight-features-200229-001522 ... hang on
WARNING:apache_beam.runners.interactive.interactive_environment:Interactive Beam requires Python 3.5.3+.
WARNING:apache_beam.runners.interactive.interactive_environment:Dependencies required for Interactive Beam
PCollection visualization are not available, please use: `pip install apache-beam[interactive]` to install
necessary dependencies to enable all data visualization features.
```

- The dataflow has been created in the bucket which was provided.





[12]:
```
import shutil, os, datetime
OUTPUT_DIR = 'babyweight_trained'
shutil.rmtree(OUTPUT_DIR, ignore_errors=True)
EXPORT_PATH = os.path.join(OUTPUT_DIR, datetime.datetime.now().strftime('%Y%m%d%H%M%S'))
tf.saved_model.save(model, EXPORT_PATH) # with default serving function
print("Exported trained model to {}".format(EXPORT_PATH))
```

```
WARNING:tensorflow:From /usr/local/lib/python3.5/dist-packages/tensorflow_core/python/ops/resource_variable_ops.py:1786: calling BaseResourceVariable.__init__ (from tensorflow.python.ops.resource_variable_ops) with constraint is deprecated and will be removed in a future version.
Instructions for updating:
If using Keras pass *_constraint arguments to layers.
INFO:tensorflow:Assets written to: babyweight_trained/20200228211127/assets
Exported trained model to babyweight_trained/20200228211127
```

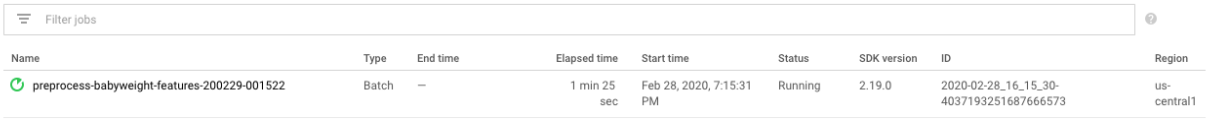
[13]:
```
!ls $EXPORT_PATH
```

```
assets  saved_model.pb  variables
```

[11]:
```
%%bash
gsutil ls gs://${greeshmagopal}/babyweight/preproc/*-00000*
```

```
gs://dataprep-staging-bf6301b0-fcba-486c-bbea-01975a37d697/
gs://greeshmagopal/
gs://testjan19/
```

- In the GCP console we can see the dataflow job running



STEP 5: Training Keras model on Cloud AI Platform

Training and hyperparameter tuning on Cloud AI Platform using Keras and requires TensorFlow 2.0

- Make Keras wide deep code work on a subset of data.

```
Copying gs://cloud-training-demos/babyweight/trained_model_tuned/model.ckpt-483973.index...
Copying gs://cloud-training-demos/babyweight/trained_model_tuned/model.ckpt-483973.meta...
Copying gs://cloud-training-demos/babyweight/trained_model_tuned/model.ckpt-571432.data-00000-of-00003...
Copying gs://cloud-training-demos/babyweight/trained_model_tuned/model.ckpt-571432.data-00001-of-00003...
Copying gs://cloud-training-demos/babyweight/trained_model_tuned/model.ckpt-571432.data-00002-of-00003...
Copying gs://cloud-training-demos/babyweight/trained_model_tuned/model.ckpt-571432.index...
Copying gs://cloud-training-demos/babyweight/trained_model_tuned/model.ckpt-571432.meta...
/ [573/573 files][  6.1 GiB/  6.1 GiB] 100% Done 955.1 MiB/s ETA 00:00:00
Operation completed over 573 objects/6.1 GiB.
```

[5]:
```
%%bash
gsutil ls gs://${BUCKET}/babyweight/preproc/*-00000*
```

```
gs://greeshmagopal/babyweight/preproc/eval.csv-00000-of-00012
gs://greeshmagopal/babyweight/preproc/train.csv-00000-of-00043
```

Now that we have the Keras wide-and-deep code working on a subset of the data, we can package the TensorFlow code up as a Python module and train it on Cloud AI Platform.

Train on Cloud AI Platform

- Making the code a Python package

```
    # Append trial_id to path if we are doing hptuning
    # This code can be removed if you are not using hyperparameter tuning
    output_dir = os.path.join(
        output_dir,
        json.loads(
            os.environ.get('TF_CONFIG', '{}')
        ).get('task', {}).get('trial', '')
    )

    # Run the training job
    model.train_and_evaluate(output_dir)
```

```
Overwriting babyweight_tf2/trainer/task.py
```

- Using gcloud to submit the training code to Cloud AI Platform

```
                      steps_per_epoch=steps_per_epoch,
                      verbose=2, # 0=silent, 1=progress bar, 2=one line per epoch
                      callbacks=[cp_callback])

    EXPORT_PATH = os.path.join(output_dir, datetime.datetime.now().strftime('%Y%m%d%H%M%S'))
    tf.saved_model.save(model, EXPORT_PATH) # with default serving function
    print("Exported trained model to {}".format(EXPORT_PATH))
```

```
Overwriting babyweight_tf2/trainer/model.py
```

- Running the model

```
linear (Dense)                  (None, 10)           720         wide_inputs[0][0]
__________________________________________________________________________________________________
both (Concatenate)              (None, 14)           0           dnn_3[0][0]
                                                                 linear[0][0]
__________________________________________________________________________________________________
weight (Dense)                  (None, 1)            15          both[0][0]
==================================================================================================
Total params: 65,763
Trainable params: 65,763
Non-trainable params: 0
__________________________________________________________________________________________________
None
Train for 10 steps, validate for 1 steps
Epoch 1/10

Epoch 00001: saving model to babyweight_trained/checkpoints/babyweight
10/10 - 3s - loss: 54.4846 - rmse: 7.3638 - mse: 54.4846 - val_loss: 56.8280 - val_rmse: 7.5384 - val_mse:
56.8280
Epoch 2/10

Epoch 00002: saving model to babyweight_trained/checkpoints/babyweight
10/10 - 0s - loss: 53.6307 - rmse: 7.3189 - mse: 53.6307 - val_loss: 56.3361 - val_rmse: 7.5057 - val_mse:
56.3361
Epoch 3/10

Epoch 00003: saving model to babyweight_trained/checkpoints/babyweight
10/10 - 0s - loss: 54.3448 - rmse: 7.3636 - mse: 54.3448 - val_loss: 54.9866 - val_rmse: 7.4153 - val_mse:
54.9866
Epoch 4/10

Epoch 00004: saving model to babyweight_trained/checkpoints/babyweight
10/10 - 0s - loss: 54.3620 - rmse: 7.3596 - mse: 54.3620 - val_loss: 53.8337 - val_rmse: 7.3371 - val_mse:
```

- Updating the files

[10]:
```
%%writefile babyweight_tf2/Dockerfile
FROM gcr.io/deeplearning-platform-release/tf2-cpu
COPY trainer /babyweight_tf2/trainer
RUN apt update && \
    apt install --yes python3-pip && \
    pip3 install --upgrade --quiet tf-nightly-2.0-preview

ENV PYTHONPATH ${PYTHONPATH}:/babyweight_tf2
CMD ["python3", "-m", "trainer.task"]
```

```
Overwriting babyweight_tf2/Dockerfile
```

[11]:
```
%%writefile babyweight_tf2/push_docker.sh
export PROJECT_ID=$(gcloud config list project --format "value(core.project)")
export IMAGE_REPO_NAME=babyweight_training_container
#export IMAGE_TAG=$(date +%Y%m%d_%H%M%S)
#export IMAGE_URI=gcr.io/$PROJECT_ID/$IMAGE_REPO_NAME:$IMAGE_TAG
export IMAGE_URI=gcr.io/$PROJECT_ID/$IMAGE_REPO_NAME

echo "Building  $IMAGE_URI"
docker build -f Dockerfile -t $IMAGE_URI ./
echo "Pushing $IMAGE_URI"
docker push $IMAGE_URI
```

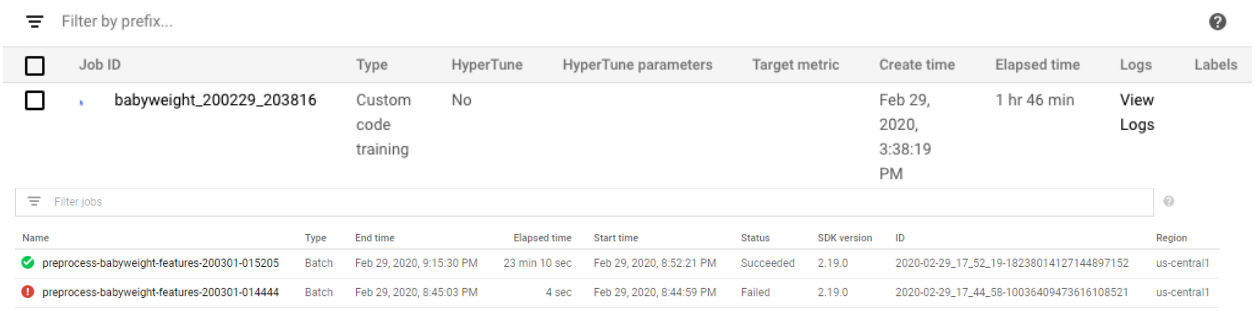
```
Overwriting babyweight_tf2/push_docker.sh
```

- Making the code stand alone to it on Cloud AI Platform.

[13]:
```
%%bash
OUTDIR=gs://${BUCKET}/babyweight/trained_model
JOBID=babyweight_$(date -u +%y%m%d_%H%M%S)
JOBNAME=$JOBID
echo $OUTDIR $REGION $JOBNAME
gsutil -m rm -rf $OUTDIR

#IMAGE=gcr.io/deeplearning-platform-release/tf2-cpu
#IMAGE=gcr.io/$PROJECT/serverlessml_training_container

gcloud ai-platform jobs submit training $JOBNAME \
  --region=$REGION \
  --module-name=trainer.task \
  --package-path=$(pwd)/babyweight/trainer \
  --job-dir=$OUTDIR \
  --staging-bucket=gs://$BUCKET \
  --scale-tier=BASIC_TPU \
  --runtime-version='1.15' \
  -- \
  --bucket=${BUCKET} \
  --output_dir=${OUTDIR} \
  --train_examples=200000
```

```
gs://greeshmagopal/babyweight/trained_model us-central1 babyweight_200229_203816
jobId: babyweight_200229_203816
state: QUEUED
CommandException: 1 files/objects could not be removed.
Job [babyweight_200229_203816] submitted successfully.
Your job is still active. You may view the status of your job with the command
```

- The job can also be monitored in GCP console.



Hyperparameter tuning

- For hyperparameter tuning, hyperparam.xml was created and passed as --configFile. The maxParallelTrials can be increased or maxTrials reduced to get it done faster.

[18]:
```
%%writefile hyperparam.yaml
trainingInput:
  scaleTier: STANDARD_1
  hyperparameters:
    hyperparameterMetricTag: rmse
    goal: MINIMIZE
    maxTrials: 20
    maxParallelTrials: 5
    enableTrialEarlyStopping: True
    params:
    - parameterName: batch_size
      type: INTEGER
      minValue: 8
      maxValue: 512
      scaleType: UNIT_LOG_SCALE
    - parameterName: nembeds
      type: INTEGER
      minValue: 3
      maxValue: 30
      scaleType: UNIT_LINEAR_SCALE
    - parameterName: nnsize
      type: INTEGER
      minValue: 64
      maxValue: 512
      scaleType: UNIT_LOG_SCALE
```

```
Writing hyperparam.yaml
```

```
  --job-dir=$OUTDIR \
  --staging-bucket=gs://$BUCKET \
  --scale-tier=STANDARD_1 \
  --runtime-version='1.15' \
  -- \
  --bucket=${BUCKET} \
  --output_dir=${OUTDIR} \
  --train_examples=20000 --batch_size=35 --nembeds=16 --nnsize=281
```

```
gs://greeshma/babyweight/trained_model_tuned us-central1 babyweight_200301_154303
jobId: babyweight_200301_154303
state: QUEUED
```

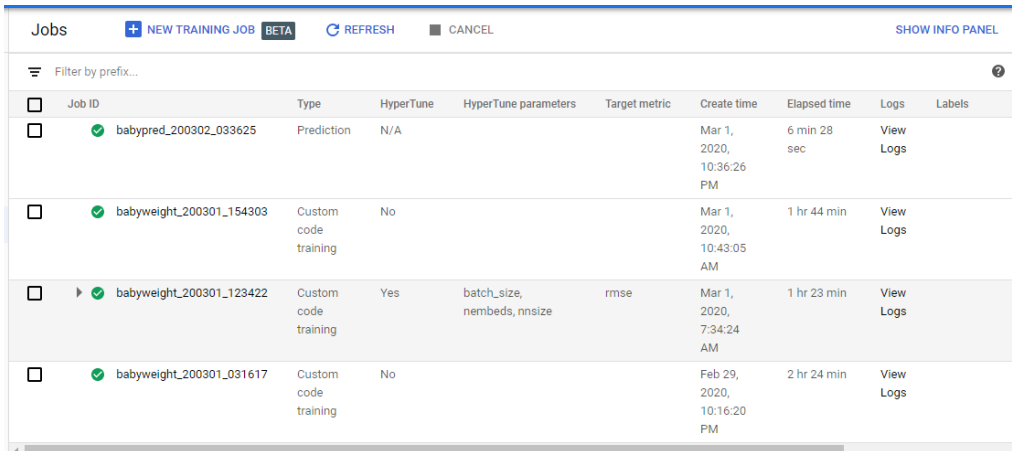
```
CommandException: 1 files/objects could not be removed.
Job [babyweight_200301_154303] submitted successfully.
Your job is still active. You may view the status of your job with the command

  $ gcloud ai-platform jobs describe babyweight_200301_154303

or continue streaming the logs with the command

  $ gcloud ai-platform jobs stream-logs babyweight_200301_154303
```

```
#IMAGE=gcr.io/deeplearning-platform-release/tf2-cpu
#IMAGE=gcr.io/$PROJECT/serverlessml_training_container

gcloud ai-platform jobs submit training $JOBNAME \
  --region=$REGION \
  --module-name=trainer.task \
  --package-path=$(pwd)/babyweight/trainer \
  --job-dir=$OUTDIR \
  --staging-bucket=gs://$BUCKET \
  --scale-tier=STANDARD_1 \
  --runtime-version='1.15' \
  -- \
  --bucket=${BUCKET} \
  --output_dir=${OUTDIR} \
  --train_examples=200000
```

```
gs://greeshma/babyweight/trained_model us-central1 babyweight_200301_031617
jobId: babyweight_200301_031617
state: QUEUED
```

```
Removing gs://greeshma/babyweight/trained_model/checkpoint#1583032346997118...
Removing gs://greeshma/babyweight/trained_model/eval/events.out.tfevents.1529348264.cmle-training-master-a137ac0fff-0-9q8r4#
1583032347221537...
Removing gs://greeshma/babyweight/trained_model/events.out.tfevents.1529347276.cmle-training-master-a137ac0fff-0-9q8r4#15830
32347402573...
```

- All the jobs have run successfully.



STEP 6: Deploy trained model

- Deploying the model

```
  gsutil mb -l ${REGION} gs://${BUCKET}
  # copy canonical model if you didn't do previous notebook
  gsutil -m cp -R gs://cloud-training-demos/babyweight/trained_model gs://${BUCKET}/babyweight/trained_model
fi
```

```
Creating gs://greeshma/...
ServiceException: 409 Bucket greeshma already exists.
Copying gs://cloud-training-demos/babyweight/trained_model/checkpoint...
Copying gs://cloud-training-demos/babyweight/trained_model/eval/events.out.tfevents.1529348264.cmle-training-master-a137ac0f
ff-0-9q8r4...
Copying gs://cloud-training-demos/babyweight/trained_model/events.out.tfevents.1529347276.cmle-training-master-a137ac0fff-0-
9q8r4...
Copying gs://cloud-training-demos/babyweight/trained_model/export/exporter/1529355466/variables/variables.data-00000-of-0000
1...
Copying gs://cloud-training-demos/babyweight/trained_model/export/exporter/1529355466/saved_model.pb...
Copying gs://cloud-training-demos/babyweight/trained_model/export/exporter/1529355466/variables/variables.index...
Copying gs://cloud-training-demos/babyweight/trained_model/model.ckpt-342784.data-00000-of-00003...
```

- Predictions of the model

```
  ]
}
response = requests.post(api, json=data, headers=headers)
print(response.content)
```

```
b'{"predictions": [{"predictions": [7.623125076293945], "key": ["b1"]}, {"predictions": [7.029077053070068], "key": ["g1"]},
{"predictions": [6.276741981506348], "key": ["b2"]}, {"predictions": [5.979225158691406], "key": ["u1"]}]}'
```

The predictions for the four instances were: 7.66, 7.22, 6.32 and 6.19 pounds respectively when I ran it (your results might be different).

- Creating a file with one instance per line and submitting using gcloud.

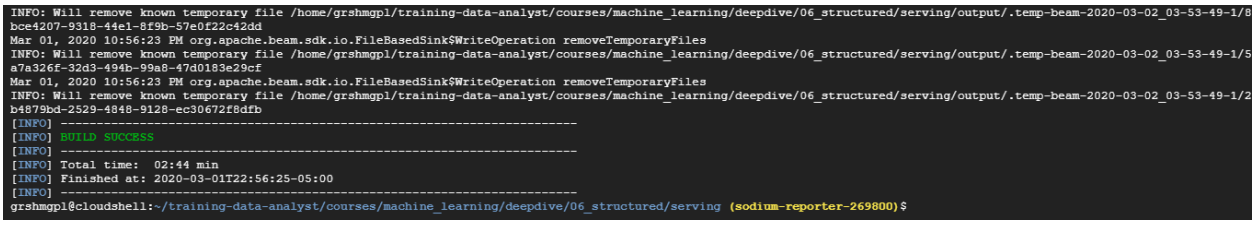
```
gcloud ai-platform jobs submit prediction babypred_$(date -u +%y%m%d_%H%M%S) \
  --data-format=TEXT --region ${REGION} \
  --input-paths=$INPUT \
  --output-path=$OUTPUT \
  --model=babyweight --version=ml_on_gcp
```

```
jobId: babypred_200302_033625
state: QUEUED
```

```
Copying file://inputs.json [Content-Type=application/json]...
/ [1 files][  205.0 B/  205.0 B]
Operation completed over 1 objects/205.0 B.
CommandException: 1 files/objects could not be removed.
Job [babypred_200302_033625] submitted successfully.
Your job is still active. You may view the status of your job with the command

  $ gcloud ai-platform jobs describe babypred_200302_033625

or continue streaming the logs with the command
```

STEP 7: Run client applications

- Deploying an AppEngine web application that consumes the machine learning service.
- Also, ML predictions are deployed to utilize in batch and real mode.



- The ML model is deployed in the model https://sodium-reporter-269800.appspot.com/form. This form can be used for model prediction.