In [0]:

```
#importing the necessary lib
import pandas as pd
import numpy as np
import os
import matplotlib.pyplot as plt
from sklearn.feature_extraction.text import TfidfVectorizer,CountVectorizer
from sklearn.model_selection import train_test_split
from keras.preprocessing.text import Tokenizer
from keras.preprocessing.sequence import pad_sequences
from keras.layers import LSTM, BatchNormalization,concatenate,Flatten,Embedding,Dense,Dropout,MaxPoolin
g2D,CuDNNLSTM,SpatialDropout1D
from keras.models import Sequential
from keras import Model,Input
from keras.layers.convolutional import Conv2D,Conv1D
import keras.backend as k
from sklearn.metrics import roc_auc_score
import tensorflow as tf
from keras.initializers import he_normal
from keras.callbacks import Callback, EarlyStopping
from time import time
from tensorflow.python.keras.callbacks import TensorBoard,ModelCheckpoint
import warnings
warnings.filterwarnings("ignore")
import keras
from keras.regularizers import l2
import pickle
```

In [0]:

```
from google.colab import drive
drive.mount('/content/drive')
```

```
Go to this URL in a browser: https://accounts.google.com/o/oauth2/auth?client_id=947318989803-6bn6qk8qd
gf4n4g3pfee6491hc0brc4i.apps.googleusercontent.com&redirect_uri=urn%3Aietf%3Awg%3Aoauth%3A2.0%3Aoob&sco
pe=email%20https%3A%2F%2Fwww.googleapis.com%2Fauth%2Fdocs.test%20https%3A%2F%2Fwww.googleapis.com%2Faut
h%2Fdrive%20https%3A%2F%2Fwww.googleapis.com%2Fauth%2Fdrive.photos.readonly%20https%3A%2F%2Fwww.googlea
pis.com%2Fauth%2Fpeopleapi.readonly&response_type=code

Enter your authorization code:
··········
Mounted at /content/drive
```

In [0]:

```
df = pd.read_csv('/content/drive/My Drive/Applied ML assignments/preprocessed_data.csv')
```

In [0]:

```
resource_data = pd.read_csv('/content/drive/My Drive/LSTM Assignment/resources.csv')
resource_data.columns
project_data = pd.read_csv('/content/drive/My Drive/LSTM Assignment/train_data.csv')
project_data.columns
```

Out[0]:

```
Index(['Unnamed: 0', 'id', 'teacher_id', 'teacher_prefix', 'school_state',
       'project_submitted_datetime', 'project_grade_category',
       'project_subject_categories', 'project_subject_subcategories',
       'project_title', 'project_essay_1', 'project_essay_2',
       'project_essay_3', 'project_essay_4', 'project_resource_summary',
       'teacher_number_of_previously_posted_projects', 'project_is_approved'],
      dtype='object')
```

In [0]:

```
price_data = resource_data.groupby('id').agg({'price':'sum', 'quantity':'sum'}).reset_index()
```

```
price_data.head(2)
```

Out[0]:

	id	price	quantity
0	p000001	459.56	7
1	p000002	515.89	21

In [0]:

```
project_data = pd.merge(project_data, price_data, on='id', how='left')
```

In [0]:

```
df['quantity'] = project_data['quantity']
#df1['columename']= df2['existing_colume_name']
```

In [0]:

```
y=df['project_is_approved']
df.drop(['project_is_approved'],axis=1, inplace=True)
x=df
```

In [0]:

```
#Splitting into train and test data
X_train, X_test, y_train, y_test = train_test_split(x, y, test_size=0.2)
print(X_train.shape, y_train.shape)
print(X_test.shape, y_test.shape)
#Splitting train data into train and cv(60:20)
X_tr, X_cv, y_tr, y_cv = train_test_split(X_train, y_train, test_size=0.2)
print(X_tr.shape, y_tr.shape)
print(X_cv.shape, y_cv.shape)
```

```
(87398, 9) (87398,)
(21850, 9) (21850,)
(69918, 9) (69918,)
(17480, 9) (17480,)
```

In [0]:

```
#Converting categorical features to One hot encoded features
#clean_categories
vectorizer = CountVectorizer()
vectorizer.fit(X_tr['clean_categories'].values)
categories_one_hot_train = vectorizer.transform(X_tr['clean_categories'].values)
categories_one_hot_cv = vectorizer.transform(X_cv['clean_categories'].values)
categories_one_hot_test = vectorizer.transform(X_test['clean_categories'].values)
print(categories_one_hot_train.shape)
print(categories_one_hot_test.shape)
print(categories_one_hot_cv.shape)

#clean_subcategories
vectorizer = CountVectorizer()
vectorizer.fit(X_tr['clean_subcategories'].values)
subcategories_one_hot_train = vectorizer.transform(X_tr['clean_subcategories'].values)
subcategories_one_hot_cv = vectorizer.transform(X_cv['clean_subcategories'].values)
subcategories_one_hot_test = vectorizer.transform(X_test['clean_subcategories'].values)
print(subcategories_one_hot_train.shape)
print(subcategories_one_hot_test.shape)
print(subcategories_one_hot_cv.shape)

#school_state
vectorizer = CountVectorizer()
vectorizer.fit(X_tr['school_state'].values)
schoolstate_one_hot_train = vectorizer.transform(X_tr['school_state'].values)
schoolstate_one_hot_cv = vectorizer.transform(X_cv['school_state'].values)
```

```
schoolstate_one_hot_test = vectorizer.transform(X_test['school_state'].values)
print(schoolstate_one_hot_train.shape)
print(schoolstate_one_hot_test.shape)
print(schoolstate_one_hot_cv.shape)

#project_grade_category
vectorizer = CountVectorizer()
vectorizer.fit(X_tr['project_grade_category'].values)
project_grade_category_one_hot_train = vectorizer.transform(X_tr['project_grade_category'].values)
project_grade_category_one_hot_cv = vectorizer.transform(X_cv['project_grade_category'].values)
project_grade_category_one_hot_test = vectorizer.transform(X_test['project_grade_category'].values)
print(project_grade_category_one_hot_train.shape)
print(project_grade_category_one_hot_test.shape)
print(project_grade_category_one_hot_cv.shape)



#teacher_prefix
vectorizer = CountVectorizer()
vectorizer.fit(X_tr['teacher_prefix'].values)
teacherprefix_ohe_train = vectorizer.transform(X_tr['teacher_prefix'].values)
teacherprefix_ohe_cv = vectorizer.transform(X_cv['teacher_prefix'].values)
teacherprefix_ohe_test = vectorizer.transform(X_test['teacher_prefix'].values)
print(teacherprefix_ohe_cv.shape)
print(teacherprefix_ohe_train.shape)
print(teacherprefix_ohe_test.shape)
```

```
(69918, 9)
(21850, 9)
(17480, 9)
(69918, 30)
(21850, 30)
(17480, 30)
(69918, 51)
(21850, 51)
(17480, 51)
(69918, 4)
(21850, 4)
(17480, 4)
(17480, 5)
(69918, 5)
(21850, 5)
```

In [0]:

```
from sklearn.preprocessing import Normalizer
normalizer = Normalizer()
normalizer.fit(X_tr['price'].values.reshape(-1,1))

x_train_price_norm = normalizer.transform(X_tr['price'].values.reshape(-1,1))
x_cv_price_norm = normalizer.transform(X_cv['price'].values.reshape(-1,1))
x_test_price_norm = normalizer.transform(X_test['price'].values.reshape(-1,1))

print("After normalizing price")
print(x_train_price_norm.shape, y_tr.shape)
print(x_cv_price_norm.shape, y_cv.shape)
print(x_test_price_norm.shape, y_test.shape)

print("==========================================================")

normalizer = Normalizer()
normalizer.fit(X_tr['quantity'].values.reshape(-1,1))

x_train_qty_norm = normalizer.transform(X_tr['quantity'].values.reshape(-1,1))
x_cv_qty_norm = normalizer.transform(X_cv['quantity'].values.reshape(-1,1))
x_test_qty_norm = normalizer.transform(X_test['quantity'].values.reshape(-1,1))
print("After normalizing the quantity")
print(x_train_qty_norm.shape, y_tr.shape)
print(x_cv_qty_norm.shape, y_cv.shape)
print(x_test_qty_norm.shape, y_test.shape)
print("==========================================================")

normalizer = Normalizer()
normalizer.fit(X_tr['teacher_number_of_previously_posted_projects'].values.reshape(-1,1))

x_train_tpp_norm = normalizer.transform(X_tr['teacher_number_of_previously_posted_projects'].values.res
```

```
hape(-1,1))
x_cv_tpp_norm = normalizer.transform(X_cv['teacher_number_of_previously_posted_projects'].values.reshap
e(-1,1))
x_test_tpp_norm = normalizer.transform(X_test['teacher_number_of_previously_posted_projects'].values.re
shape(-1,1))
print("After normalizing the teacher_number_of_previously_posted_projects")
print(x_train_qty_norm.shape, y_tr.shape)
print(x_cv_qty_norm.shape, y_cv.shape)
print(x_test_qty_norm.shape, y_test.shape)
```

```
After normalizing price
(69918, 1) (69918,)
(17480, 1) (17480,)
(21850, 1) (21850,)
==========================================================
After normalizing the quantity
(69918, 1) (69918,)
(17480, 1) (17480,)
(21850, 1) (21850,)
==========================================================
After normalizing the teacher_number_of_previously_posted_projects
(69918, 1) (69918,)
(17480, 1) (17480,)
(21850, 1) (21850,)
```

In [0]:

```
#print(schoolstate_one_hot_train.shape)
from scipy import sparse
from numpy import hstack
x_tr_rem = sparse.hstack((schoolstate_one_hot_train, teacherprefix_ohe_train, project_grade_category_on
e_hot_train,subcategories_one_hot_train,categories_one_hot_train, x_train_price_norm,x_train_qty_norm,x
_train_tpp_norm)).todense()
x_cv_rem = sparse.hstack(( schoolstate_one_hot_cv, teacherprefix_ohe_cv, project_grade_category_one_hot
_cv,subcategories_one_hot_cv,categories_one_hot_cv, x_cv_price_norm,x_cv_qty_norm,x_cv_tpp_norm)).toden
se()
x_te_rem = sparse.hstack((schoolstate_one_hot_test, teacherprefix_ohe_test, project_grade_category_one_
hot_test,subcategories_one_hot_test,categories_one_hot_test, x_test_price_norm,x_test_qty_norm,x_test_t
pp_norm)).todense()
print("Final Data matrix")
print(x_tr_rem.shape, y_tr.shape)
print(x_cv_rem.shape, y_cv.shape)
print(x_te_rem.shape, y_test.shape)
print("="*100)
```

```
Final Data matrix
(69918, 102) (69918,)
(17480, 102) (17480,)
(21850, 102) (21850,)
====================================================================================================
```

In [0]:

```
from sklearn.preprocessing import StandardScaler
mms = StandardScaler().fit(x_tr_rem)
x_tr_rem_norm = mms.transform(x_tr_rem)
x_cv_rem_norm = mms.transform(x_cv_rem)
x_te_rem_norm = mms.transform(x_te_rem)
```

In [0]:

```
x_tr_rem_reshape = np.array(x_tr_rem).reshape(69918,102,1)
x_cv_rem_reshape = np.array(x_cv_rem).reshape(17480, 102,1)
x_test_rem_reshape = np.array(x_te_rem).reshape(21850, 102,1)
```

In [0]:

```
max_length=400
#https://machinelearningmastery.com/use-word-embedding-layers-deep-learning-keras/
def padded(encoded_docs):
  max_length = 400
```

```
  padded_docs = pad_sequences(encoded_docs, maxlen=max_length, padding='post')
  return padded_docs


#https://stackoverflow.com/posts/51956230/revisions
t = Tokenizer()
t.fit_on_texts(X_tr['essay'].values)
vocab_size = len(t.word_index) + 1
# integer encode the documents
encoded_docs = t.texts_to_sequences(X_tr['essay'].values)
essay_padded_train = padded(encoded_docs)


# integer encode the documents
encoded_docs = t.texts_to_sequences(X_cv['essay'].values)
essay_padded_cv = padded(encoded_docs)


encoded_docs = t.texts_to_sequences(X_test['essay'].values)
essay_padded_test = padded(encoded_docs)



print("encoded train data shape",essay_padded_train.shape)
print("encoded cv data shape",essay_padded_cv.shape)
print("encoded cv data shape",essay_padded_test.shape)
```

```
encoded train data shape (69918, 400)
encoded cv data shape (17480, 400)
encoded cv data shape (21850, 400)
```

In [0]:

```
embeddings_index = dict()
f = open('/content/drive/My Drive/Applied ML assignments/glove.6B.300d.txt')
for line in f:
 values = line.split()
 word = values[0]
 coefs = np.asarray(values[1:], dtype='float32')
 embeddings_index[word] = coefs
f.close()
```

In [0]:

```
embedding_matrix = np.zeros((vocab_size, 300))
for word, i in t.word_index.items():
 embedding_vector = embeddings_index.get(word)
 if embedding_vector is not None:
  embedding_matrix[i] = embedding_vector
```

In [0]:

```
print("embedding matrix shape",embedding_matrix.shape)
```

```
embedding matrix shape (47467, 300)
```

In [0]:

```
from keras.utils import to_categorical
y_binary_train = to_categorical(y_tr)
y_binary_cv = to_categorical(y_cv)
y_binary_test = to_categorical(y_test)
```

In [0]:

```
import keras
from tensorboardcolab import *
from keras.regularizers import l2
from keras.layers import Conv1D,MaxPooling1D,  LeakyReLU
import keras.backend as K
```

```
K.clear_session()
```

```
WARNING:tensorflow:From /usr/local/lib/python3.6/dist-packages/keras/backend/tensorflow_backend.py:107:
The name tf.reset_default_graph is deprecated. Please use tf.compat.v1.reset_default_graph instead.

WARNING:tensorflow:From /usr/local/lib/python3.6/dist-packages/keras/backend/tensorflow_backend.py:111:
The name tf.placeholder_with_default is deprecated. Please use tf.compat.v1.placeholder_with_default in
stead.

WARNING:tensorflow:From /usr/local/lib/python3.6/dist-packages/keras/backend/tensorflow_backend.py:66:
The name tf.get_default_graph is deprecated. Please use tf.compat.v1.get_default_graph instead.
```

In [0]:

```
text_input = Input(shape=(400,), name = "text_input")
# max_length = 400 ---->max length of sentence

e1 = Embedding(vocab_size, 300, weights=[embedding_matrix], input_length=400)(text_input)

l1= LSTM(128,activation = "relu",dropout=0.5,kernel_regularizer=l2(0.001),kernel_initializer='glorot_no
rmal',return_sequences=True,input_shape=(150,300))(e1)
#dout = Dropout(0.5)(l1)
f1= Flatten()(l1)

rem = Input(shape=(x_tr_rem.shape[1],1), name="rem")
rem_conv1 = Conv1D(128, 3,kernel_initializer='glorot_normal')(rem)

max_pool =MaxPooling1D(3)(rem_conv1)
#rem_conv3 =Conv1D(64, 3, activation='sigmoid')(max_pool)
#rem_conv4 =Conv1D(128, 3, activation='sigmoid')(rem_conv3)
f2= Flatten()(max_pool)
x = keras.layers.concatenate([f1,f2])

#x=BatchNormalization()(x)
x= Dense(32,kernel_regularizer=l2(0.001),kernel_initializer='glorot_normal')(x)
x= Dense(16, activation='relu')(x)
output=Dense(2, activation='softmax')(x)
model_3 = Model(inputs=[text_input,rem], outputs=output)
model_3.summary()
```

```
WARNING:tensorflow:From /usr/local/lib/python3.6/dist-packages/keras/backend/tensorflow_backend.py:541:
The name tf.placeholder is deprecated. Please use tf.compat.v1.placeholder instead.

WARNING:tensorflow:From /usr/local/lib/python3.6/dist-packages/keras/backend/tensorflow_backend.py:4432
: The name tf.random_uniform is deprecated. Please use tf.random.uniform instead.

WARNING:tensorflow:From /usr/local/lib/python3.6/dist-packages/keras/backend/tensorflow_backend.py:190:
The name tf.get_default_session is deprecated. Please use tf.compat.v1.get_default_session instead.

WARNING:tensorflow:From /usr/local/lib/python3.6/dist-packages/keras/backend/tensorflow_backend.py:197:
The name tf.ConfigProto is deprecated. Please use tf.compat.v1.ConfigProto instead.

WARNING:tensorflow:From /usr/local/lib/python3.6/dist-packages/keras/backend/tensorflow_backend.py:203:
The name tf.Session is deprecated. Please use tf.compat.v1.Session instead.

WARNING:tensorflow:From /usr/local/lib/python3.6/dist-packages/keras/backend/tensorflow_backend.py:207:
The name tf.global_variables is deprecated. Please use tf.compat.v1.global_variables instead.

WARNING:tensorflow:From /usr/local/lib/python3.6/dist-packages/keras/backend/tensorflow_backend.py:216:
The name tf.is_variable_initialized is deprecated. Please use tf.compat.v1.is_variable_initialized inst
ead.

WARNING:tensorflow:From /usr/local/lib/python3.6/dist-packages/keras/backend/tensorflow_backend.py:223:
The name tf.variables_initializer is deprecated. Please use tf.compat.v1.variables_initializer instead.

WARNING:tensorflow:From /usr/local/lib/python3.6/dist-packages/keras/backend/tensorflow_backend.py:4479
: The name tf.truncated_normal is deprecated. Please use tf.random.truncated_normal instead.

WARNING:tensorflow:From /usr/local/lib/python3.6/dist-packages/keras/backend/tensorflow_backend.py:3733
: calling dropout (from tensorflow.python.ops.nn_ops) with keep_prob is deprecated and will be removed
in a future version.
Instructions for updating:
Please use `rate` instead of `keep_prob`. Rate should be set to `rate = 1 - keep_prob`.
WARNING:tensorflow:From /usr/local/lib/python3.6/dist-packages/keras/backend/tensorflow_backend.py:4267
```

```
WARNING:tensorflow:From /usr/local/lib/python3.6/dist-packages/keras/backend/tensorflow_backend.py:4267
: The name tf.nn.max_pool is deprecated. Please use tf.nn.max_pool2d instead.

Model: "model_1"
```

Layer (type)	Output Shape	Param #	Connected to
text_input (InputLayer)	(None, 400)	0	
rem (InputLayer)	(None, 102, 1)	0	
embedding_1 (Embedding)	(None, 400, 300)	14240100	text_input[0][0]
conv1d_1 (Conv1D)	(None, 100, 128)	512	rem[0][0]
lstm_1 (LSTM)	(None, 400, 128)	219648	embedding_1[0][0]
max_pooling1d_1 (MaxPooling1D)	(None, 33, 128)	0	conv1d_1[0][0]
flatten_1 (Flatten)	(None, 51200)	0	lstm_1[0][0]
flatten_2 (Flatten)	(None, 4224)	0	max_pooling1d_1[0][0]
concatenate_1 (Concatenate)	(None, 55424)	0	flatten_1[0][0] flatten_2[0][0]
dense_1 (Dense)	(None, 32)	1773600	concatenate_1[0][0]
dense_2 (Dense)	(None, 16)	528	dense_1[0][0]
dense_3 (Dense)	(None, 2)	34	dense_2[0][0]

```
Total params: 16,234,422
Trainable params: 16,234,422
Non-trainable params: 0
```

In [0]:

```
#https://stackoverflow.com/posts/51734992/revisions
import tensorflow as tf
from sklearn.metrics import roc_auc_score

def auroc(y_true, y_pred):
    return tf.py_func(roc_auc_score, (y_true, y_pred), tf.double)
```

In [0]:

```
adam = keras.optimizers.Adam(lr=0.001)
model_3.compile(optimizer=adam, loss='categorical_crossentropy',metrics=[auroc])
```

```
WARNING:tensorflow:From /usr/local/lib/python3.6/dist-packages/keras/optimizers.py:793: The name tf.tra
in.Optimizer is deprecated. Please use tf.compat.v1.train.Optimizer instead.

WARNING:tensorflow:From /usr/local/lib/python3.6/dist-packages/keras/backend/tensorflow_backend.py:3576
: The name tf.log is deprecated. Please use tf.math.log instead.

WARNING:tensorflow:From <ipython-input-24-4a25250c5bd7>:5: py_func (from tensorflow.python.ops.script_o
ps) is deprecated and will be removed in a future version.
Instructions for updating:
tf.py_func is deprecated in TF V2. Instead, there are two
    options available in V2.
    - tf.py_function takes a python function which manipulates tf eager
    tensors instead of numpy arrays. It's easy to convert a tf eager tensor to
    an ndarray (just call tensor.numpy()) but having access to eager tensors
    means `tf.py_function`s can use accelerators such as GPUs as well as
    being differentiable using a gradient tape.
    - tf.numpy_function maintains the semantics of the deprecated tf.py_func
    (it is not differentiable, and manipulates numpy arrays). It drops the
    stateful argument making all functions stateful.
```

In [0]:

```
from keras.callbacks import *
from sklearn.metrics import roc_auc_score
checkpoint = EarlyStopping(monitor='val_loss', mode='min', verbose=1)
#callbacks_list = [checkpoint]
batch_size = 512
filepath = '/content/drive/My Drive/LSTM Assignment/epochs:{epoch:03d}-val_auc:{val_auroc:.3f}.hdf5'
#earlyStopping = EarlyStopping(monitor='val_loss', patience=10, verbose=0, mode='min')
mcp_save = ModelCheckpoint(filepath, save_best_only=True, monitor='val_auc', mode='max')
reduce_lr_loss = ReduceLROnPlateau(monitor='val_loss', factor=0.2, patience=1, verbose=1,min_lr=0.001,
mode='min')
callbacks=[checkpoint, mcp_save, reduce_lr_loss]

history_3= model_3.fit({'text_input': essay_padded_train, 'rem':x_tr_rem_reshape},y_binary_train,
          epochs=10, batch_size=512,verbose=1, validation_data=({'text_input': essay_padded_cv, 'rem':
x_cv_rem_reshape},y_binary_cv),callbacks=callbacks)
```

```
WARNING:tensorflow:From /usr/local/lib/python3.6/dist-packages/tensorflow_core/python/ops/math_grad.py:
1424: where (from tensorflow.python.ops.array_ops) is deprecated and will be removed in a future versio
n.
Instructions for updating:
Use tf.where in 2.0, which has the same broadcast rule as np.where
WARNING:tensorflow:From /usr/local/lib/python3.6/dist-packages/keras/backend/tensorflow_backend.py:1033
: The name tf.assign_add is deprecated. Please use tf.compat.v1.assign_add instead.

WARNING:tensorflow:From /usr/local/lib/python3.6/dist-packages/keras/backend/tensorflow_backend.py:1020
: The name tf.assign is deprecated. Please use tf.compat.v1.assign instead.

Train on 69918 samples, validate on 17480 samples
Epoch 1/10
69918/69918 [==============================] - 1266s 18ms/step - loss: 0.5759 - auroc: 0.6585 - val_los
s: 0.4355 - val_auroc: 0.7325
Epoch 2/10
69918/69918 [==============================] - 1251s 18ms/step - loss: 0.4099 - auroc: 0.7357 - val_los
s: 0.3943 - val_auroc: 0.7502
Epoch 3/10
69918/69918 [==============================] - 1256s 18ms/step - loss: 0.3819 - auroc: 0.7613 - val_los
s: 0.3805 - val_auroc: 0.7546
Epoch 4/10
69918/69918 [==============================] - 1246s 18ms/step - loss: 0.3683 - auroc: 0.7813 - val_los
s: 0.3802 - val_auroc: 0.7526
Epoch 5/10
69918/69918 [==============================] - 1251s 18ms/step - loss: 0.3543 - auroc: 0.8018 - val_los
s: 0.3821 - val_auroc: 0.7511

Epoch 00005: ReduceLROnPlateau reducing learning rate to 0.001.
Epoch 00005: early stopping
```

In [0]:

```
result = model_3.evaluate({'text_input': essay_padded_test, 'rem':x_test_rem_reshape},
          y_binary_test,batch_size=512)
```

```
21850/21850 [==============================] - 104s 5ms/step
```

In [0]:

```
print("{} of test data {}". format(model_3.metrics_names[0],result[0]))
print("{} of test data {}". format(model_3.metrics_names[1],result[1]))
```

```
loss of test data 0.38519651322670334
auroc of test data 0.7469911987786353
```

In [0]:

```
from prettytable import PrettyTable
Z=PrettyTable()
Z.field_names=["model","test_auc","test_loss"]
Z.add_row(["model_1","73.82","0.411"])
Z.add_row(["model_2","75.91","0.445"])
Z.add_row(["model_3","74.69","0.385"])
```

```
print(Z)
```

```
+---------+----------+-----------+
|  model  | test_auc | test_loss |
+---------+----------+-----------+
| model_1 |  73.82   |   0.411   |
| model_2 |  75.91   |   0.445   |
| model_3 |  74.69   |   0.385   |
+---------+----------+-----------+
```