In [ ]:
```
# module definition

from tensorflow.keras.models import Model
from tensorflow.keras.utils import plot_model
from tensorflow.keras.preprocessing.sequence import pad_sequences
from tensorflow.keras.layers import Input, Embedding, LSTM, Flatten, Dense, Ac
tivation, Dropout, Concatenate, BatchNormalization, Conv1D
from keras.preprocessing.text import Tokenizer
from tensorflow.keras.utils import to_categorical
from tensorflow.keras.preprocessing.text import one_hot
from tensorflow import py_function
from tensorflow.keras.optimizers import Adadelta, Adagrad, Adamax, Adam, Adama
x, RMSprop, SGD
from keras.constraints import unit_norm
from tensorflow.keras.callbacks import EarlyStopping, ModelCheckpoint, TensorB
oard
import tensorflow
from sklearn.preprocessing import LabelEncoder
from sklearn.preprocessing import OneHotEncoder
from sklearn.model_selection import train_test_split
from sklearn.preprocessing import MinMaxScaler
from sklearn.metrics import roc_auc_score, accuracy_score, log_loss, roc_curve
, auc
from sklearn.feature_extraction.text import TfidfVectorizer
import seaborn as sns
import matplotlib.pyplot as plt
import datetime, pandas as pd, numpy as np, re, pickle, os, datetime
from collections import Counter
from pylab import rcParams

from google.colab import drive
drive.mount('/content/drive/', force_remount=True)
```

```
Mounted at /content/drive/
```

utility functions

In [ ]:
```
###############################################################################
##############################
# utility functions
###############################################################################
##############################

ddir='/content/drive/My Drive/aaic/case_studies/donors_choose/'

def roc_auc_scr(y_true, y_pred):
  '''
  compute roc_auc_score given actual and predicted output labels

  '''
  return py_function(roc_auc_score, (y_true, y_pred), tensorflow.double)


def plot_graphs(history):
  '''
  Create graphs for train and validation losses, train and validation accuraci
es, train and validation roc_auc_scr,

  '''
  rcParams['figure.figsize'] = 5, 10
  fig, axs = plt.subplots(3)

  axs[0].plot(history.epoch, history.history['loss'], label='train_loss')
  axs[0].plot(history.epoch, history.history['val_loss'], label='val_loss')
  axs[0].legend()
  axs[0].grid()


  axs[1].plot(history.epoch, history.history['accuracy'] , label='train_accura
cy')
  axs[1].plot(history.epoch, history.history['val_accuracy'], label='val_accur
acy')
  axs[1].legend()
  axs[1].grid()


  axs[2].plot(history.epoch, history.history['roc_auc_scr'], label='train_roc_
auc')
  axs[2].plot(history.epoch, history.history['val_roc_auc_scr'], label='val_ro
c_auc')
  axs[2].legend()
  axs[2].grid()



def create_embedding_dict():
  '''
  Create embedding dictionary using glove 300 dimesional vectors

  '''
  embeddings_index = {}
  f = open(ddir+'glove.42B.300d.txt','r')
  for line in f:
```

```
    values = line.split()
    word = values[0]
    coefs = np.asarray(values[1:], dtype='float32')
    embeddings_index[word] = coefs
  f.close()
  return embeddings_index

embeddings_index = create_embedding_dict()
print('Found %s word vectors.' % len(embeddings_index))

# f=open(ddir+'embeddings.txt','wb')
# pickle.dump(embeddings_index, f)
# f.close()

# f=open(ddir+'embeddings.txt','rb')
# embeddings_index = pickle.load(f)
# f.close()
# print('Found %s word vectors.' % len(embeddings_index))
```

```
Found 1917495 word vectors.
```

Load data and create resource_summary_contains_numeric_digits feature

In [ ]:
```
################################################################################
##############################
# Load data and create resource_summary_contains_numeric_digits feature
################################################################################
##############################

ddir='/content/drive/My Drive/aaic/case_studies/donors_choose/'
data=pd.read_csv(ddir+'preprocessed_data.csv')
summaries = pd.read_csv(ddir+'train_data.csv')['project_resource_summary'].val
ues

# https://stackoverflow.com/questions/19859282/check-if-a-string-contains-a-nu
mber
# https://stackoverflow.com/questions/20840803/how-to-convert-false-to-0-and-t
rue-to-1-in-python

data['resource_summary_contains_numeric_digits'] = list(map(lambda x: 1*bool(r
e.search(r'\d', x)), summaries))

print(data.shape)
print(data.project_is_approved.value_counts())
```

```
(109248, 10)
1    92706
0    16542
Name: project_is_approved, dtype: int64
```

Note: Since there is high imbalance in the data, so we will upsample the data to create more minority class samples.

Upsample the monitory class to handle data imbalance

In [ ]:
```
# https://www.tensorflow.org/tutorials/structured_data/imbalanced_data


not_approved = data.groupby('project_is_approved').get_group(0)
max_size = data['project_is_approved'].value_counts().max()
data_for_oversampling = not_approved.sample(max_size-len(not_approved), replac
e=True)
data = data.append(data_for_oversampling)

print("Number of data points in class 0 and class 1")
print(data.project_is_approved.value_counts())

Y = data['project_is_approved']
X = data.drop('project_is_approved',axis=1)
X_train, X_test, Y_train, Y_test = train_test_split(X,Y, test_size=0.2, strati
fy=Y)

# one hot encode Y_train and Y_test

Y_train_cat = to_categorical(Y_train,num_classes=2)
Y_test_cat = to_categorical(Y_test,num_classes=2)


print('X.shape:',X.shape)
print('X_train.shape:',X_train.shape)
print('X_test.shape',X_test.shape)
print('Y_train_cat.shape',Y_train_cat.shape)
print('Y_test_cat.shape',Y_test_cat.shape)
```

```
Number of data points in class 0 and class 1
1    92706
0    92706
Name: project_is_approved, dtype: int64
X.shape: (185412, 9)
X_train.shape: (148329, 9)
X_test.shape (37083, 9)
Y_train_cat.shape (148329, 2)
Y_test_cat.shape (37083, 2)
```

Model 1 : DATA Preprocessing

In [ ]:
```
################################## LABEL ENCODING ################################
##


######### essay DATA Preprocessing ###########

MAX_NUM_WORDS = 16000
MAX_LENGTH = 100
EMBEDDING_DIM = 300

tokenizer = Tokenizer(MAX_NUM_WORDS)
tokenizer.fit_on_texts(X_train.essay)

train_essay = tokenizer.texts_to_sequences(X_train.essay)
test_essay = tokenizer.texts_to_sequences(X_test.essay)

train_essay_padded = pad_sequences(train_essay,maxlen=MAX_LENGTH)
test_essay_padded = pad_sequences(test_essay,maxlen=MAX_LENGTH)

word_index = tokenizer.word_index
embedding_matrix1 = np.zeros((len(word_index) + 1, EMBEDDING_DIM))

for word, i in word_index.items():
  embedding_vector = embeddings_index.get(word)
  if embedding_vector is not None:
    embedding_matrix1[i] = embedding_vector


######### school_state DATA Preprocessing ###########
le = LabelEncoder()
le.fit(X_train.school_state)
train_ss = le.transform(X_train.school_state)
test_ss = le.transform(X_test.school_state)

n_classes_ss = len(np.unique(X.school_state))
train_ss = train_ss.reshape(-1,1)/n_classes_ss
test_ss = test_ss.reshape(-1,1)/n_classes_ss


######### teacher_prefix DATA Preprocessing ###########

le = LabelEncoder()
le.fit(X_train.teacher_prefix)
train_tp = le.transform(X_train.teacher_prefix)
test_tp = le.transform(X_test.teacher_prefix)

n_classes_tp = len(np.unique(X.teacher_prefix))
train_tp = train_tp.reshape(-1,1)/n_classes_tp
test_tp = test_tp.reshape(-1,1)/n_classes_tp


#########project grade category DATA Preprocessing ###########

le = LabelEncoder()
le.fit(X_train.project_grade_category)
train_pgc = le.transform(X_train.project_grade_category)
```

```
test_pgc = le.transform(X_test.project_grade_category)

n_classes_pgc = len(np.unique(X.project_grade_category))
train_pgc = train_pgc.reshape(-1,1)/n_classes_pgc
test_pgc = test_pgc.reshape(-1,1)/n_classes_pgc

######### clean_categories DATA Preprocessing ###########
# we are using X to fit the model instead of X_train since categories are diff
erent for train and test data

le = LabelEncoder()
le.fit(X.clean_categories)
train_cc = le.transform(X_train.clean_categories)
test_cc = le.transform(X_test.clean_categories)

n_classes_cc = len(np.unique(X.clean_categories))
train_cc = train_cc.reshape(-1,1)/n_classes_cc
test_cc = test_cc.reshape(-1,1)/n_classes_cc

######### clean_subcategories DATA Preprocessing ###########
# we are using X to fit the model instead of X_train since categories are diff
erent for train and test data

le = LabelEncoder()
le.fit(X.clean_subcategories)
train_cs = le.transform(X.clean_subcategories)
test_cs = le.transform(X_test.clean_subcategories)

n_classes_cs = len(np.unique(X.clean_subcategories))
train_cs = train_cc.reshape(-1,1)/n_classes_cs
test_cs = test_cc.reshape(-1,1)/n_classes_cs

# ######### teacher_prefix DATA Preprocessing ###########
mms = MinMaxScaler()
train_num = mms.fit_transform(X_train[['teacher_number_of_previously_posted_pr
ojects','price','resource_summary_contains_numeric_digits']])
test_num = mms.transform(X_test[['teacher_number_of_previously_posted_project
s','price','resource_summary_contains_numeric_digits']])


print('embedding_matrix1.shape:',embedding_matrix1.shape)
print('\n')
print('train_essay_padded.shape',train_essay_padded.shape)
print('train_ss.shape',train_ss.shape)
print('train_tp.shape',train_tp.shape)
print('train_pgc.shape', train_pgc.shape)
print('train_cc.shape', train_cc.shape)
print('train_cs.shape', train_cs.shape)
print('train_num.shape', train_num.shape)

print('\n')
print('test_essay_padded.shape',test_essay_padded.shape)
print('test_ss.shape',test_ss.shape)
print('test_tp.shape',test_tp.shape)
print('test_pgc.shape', test_pgc.shape)
print('test_cc.shape', test_cc.shape)
```

```
print('test_cs.shape', test_cs.shape)
print('test_num.shape', test_num.shape)
```

```
embedding_matrix1.shape: (52603, 300)


train_essay_padded.shape (148329, 100)
train_ss.shape (148329, 1)
train_tp.shape (148329, 1)
train_pgc.shape (148329, 1)
train_cc.shape (148329, 1)
train_cs.shape (148329, 1)
train_num.shape (148329, 3)


test_essay_padded.shape (37083, 100)
test_ss.shape (37083, 1)
test_tp.shape (37083, 1)
test_pgc.shape (37083, 1)
test_cc.shape (37083, 1)
test_cs.shape (37083, 1)
test_num.shape (37083, 3)
```

Model1 : Architecture

In [ ]:
```
 #### Essay Input ####
essay_i = Input(shape=(MAX_LENGTH,), name='inp1')
essay_eo = Embedding(len(word_index) + 1, EMBEDDING_DIM, weights=[embedding_ma
trix1], input_length=(MAX_LENGTH,), trainable=False)(essay_i)
essay_lo = LSTM(32)(essay_eo)
essay_fo = Flatten()(essay_lo)

#### School state Input ####
ss_i = Input(shape=(train_ss.shape[1],), name='inp2')
ss_eo = Embedding(n_classes_ss+1, 3, input_length=(train_ss.shape[1],))(ss_i)
ss_fo = Flatten()(ss_eo)

#### Teacher prefix Input ####
tp_i = Input(shape=(train_tp.shape[1],), name='inp3')
tp_eo = Embedding(n_classes_tp+1, 2, input_length=(train_tp.shape[1],))(tp_i)
tp_fo = Flatten()(tp_eo)

#### Project Grade category Input ####
pgc_i = Input(shape=(train_pgc.shape[1],), name='inp4')
pgc_eo = Embedding(n_classes_pgc+1, 2, input_length=(train_pgc.shape[1],))(pgc
_i)
pgc_fo = Flatten()(pgc_eo)

#### Clean Categories Input ####
cc_i = Input(shape=(train_cc.shape[1],), name='inp5')
cc_eo = Embedding(n_classes_cc+1, int((n_classes_cc+1)**0.25)+1, input_length=
(train_cc.shape[1],))(cc_i)
cc_fo = Flatten()(cc_eo)

#### Clean sub Categories Input ####
cs_i = Input(shape=(train_cs.shape[1],), name='inp6')
cs_eo = Embedding(n_classes_cs+1 , int((n_classes_cs+1)**0.25)+1, input_length
=(train_cs.shape[1],))(cs_i)
cs_fo = Flatten()(cs_eo)

#### Numeric Input ####
num_i = Input(shape=(train_num.shape[1],), name='inp7')
num_fo = Dense(16, activation='relu')(num_i) #

#### Concatenate all the inputs
concatenated_input = Concatenate()([essay_fo, ss_fo, tp_fo, pgc_fo, cc_fo, cs_
fo, num_fo])

#### Define intermediate dense layers and output layer
x = Dense(64, activation='relu', kernel_initializer='he_normal', bias_initiali
zer='random_normal')(concatenated_input)
x = Dropout(0.5)(x)
x = Dense(32, activation='relu', kernel_initializer='he_normal', bias_initiali
zer='random_normal')(x)
x = Dropout(0.5)(x)
x = Dense(16, activation='relu', kernel_initializer='he_normal', bias_initiali
zer='random_normal')(x)
x = Dropout(0.5)(x)
output = Dense(2, activation='softmax')(x)

# Define model
```

```
M1 = Model(inputs=[essay_i, ss_i, tp_i, pgc_i, cc_i, cs_i, num_i], outputs=out
put)
adm = Adam(learning_rate=0.001)
M1.compile(loss='categorical_crossentropy', metrics=['accuracy',roc_auc_scr],
optimizer=adm)
```

Model1: Training

In [ ]:
```
checkpoint_callback1 = ModelCheckpoint("donors_choose/tensorboard_checkpoints/
Model1", monitor='val_roc_auc_scr', verbose=1, save_best_only=True, mode='max'
)
earlystop_callback1  = EarlyStopping(monitor='val_roc_auc_scr', verbose=1,  mo
de='max', patience=3)
tensorboard_callback1= TensorBoard("donors_choose/tensorboard_logs/Model1", hi
stogram_freq=1)

history = M1.fit({'inp1': train_essay_padded, 'inp2': train_ss,'inp3': train_t
p, 'inp4':train_pgc, 'inp5':train_cc, 'inp6':train_cs, 'inp7':train_num},Y_tra
in_cat,validation_split=0.2 , epochs=7, batch_size=256, callbacks=[tensorboard
_callback1, checkpoint_callback1, earlystop_callback1],  shuffle=False , verbo
se=1)
```

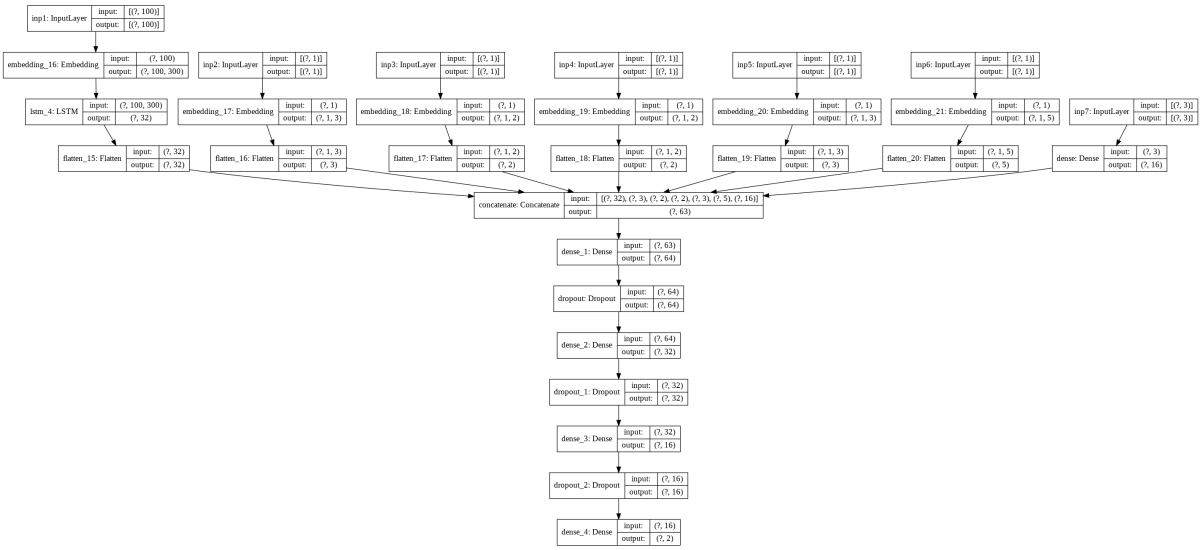
```
Epoch 1/7
  1/464 [..............................] - ETA: 0s - loss: 0.6981 - accuracy:
0.5430 - roc_auc_scr: 0.5784WARNING:tensorflow:From /usr/local/lib/python3.6/
dist-packages/tensorflow/python/ops/summary_ops_v2.py:1277: stop (from tensor
flow.python.eager.profiler) is deprecated and will be removed after 2020-07-0
1.
Instructions for updating:
use `tf.profiler.experimental.stop` instead.
  2/464 [..............................] - ETA: 31s - loss: 0.7178 - accurac
y: 0.5117 - roc_auc_scr: 0.5459WARNING:tensorflow:Callbacks method `on_train_
batch_end` is slow compared to the batch time (batch time: 0.0336s vs `on_tra
in_batch_end` time: 0.1021s). Check your callbacks.
464/464 [==============================] - ETA: 0s - loss: 0.6934 - accuracy:
0.5105 - roc_auc_scr: 0.5154
Epoch 00001: val_roc_auc_scr improved from -inf to 0.66247, saving model to d
onors_choose/tensorboard_checkpoints/Model1
WARNING:tensorflow:From /usr/local/lib/python3.6/dist-packages/tensorflow/pyt
hon/training/tracking/tracking.py:111: Model.state_updates (from tensorflow.p
ython.keras.engine.training) is deprecated and will be removed in a future ve
rsion.
Instructions for updating:
This property should not be used in TensorFlow 2.0, as updates are applied au
tomatically.
WARNING:tensorflow:From /usr/local/lib/python3.6/dist-packages/tensorflow/pyt
hon/training/tracking/tracking.py:111: Layer.updates (from tensorflow.python.
keras.engine.base_layer) is deprecated and will be removed in a future versio
n.
Instructions for updating:
This property should not be used in TensorFlow 2.0, as updates are applied au
tomatically.
INFO:tensorflow:Assets written to: donors_choose/tensorboard_checkpoints/Mode
l1/assets
464/464 [==============================] - 18s 38ms/step - loss: 0.6934 - acc
uracy: 0.5105 - roc_auc_scr: 0.5154 - val_loss: 0.6710 - val_accuracy: 0.6182
- val_roc_auc_scr: 0.6625
Epoch 2/7
464/464 [==============================] - ETA: 0s - loss: 0.6777 - accuracy:
0.5599 - roc_auc_scr: 0.5788
Epoch 00002: val_roc_auc_scr improved from 0.66247 to 0.72818, saving model t
o donors_choose/tensorboard_checkpoints/Model1
INFO:tensorflow:Assets written to: donors_choose/tensorboard_checkpoints/Mode
l1/assets
464/464 [==============================] - 17s 37ms/step - loss: 0.6777 - acc
uracy: 0.5599 - roc_auc_scr: 0.5788 - val_loss: 0.6211 - val_accuracy: 0.6705
- val_roc_auc_scr: 0.7282
Epoch 3/7
462/464 [============================>.] - ETA: 0s - loss: 0.6156 - accuracy:
0.6837 - roc_auc_scr: 0.7293
Epoch 00003: val_roc_auc_scr improved from 0.72818 to 0.76193, saving model t
o donors_choose/tensorboard_checkpoints/Model1
INFO:tensorflow:Assets written to: donors_choose/tensorboard_checkpoints/Mode
l1/assets
464/464 [==============================] - 17s 37ms/step - loss: 0.6157 - acc
uracy: 0.6836 - roc_auc_scr: 0.7292 - val_loss: 0.5929 - val_accuracy: 0.6972
- val_roc_auc_scr: 0.7619
Epoch 4/7
464/464 [==============================] - ETA: 0s - loss: 0.5944 - accuracy:
```

```
0.7045 - roc_auc_scr: 0.7544
Epoch 00004: val_roc_auc_scr improved from 0.76193 to 0.77375, saving model t
o donors_choose/tensorboard_checkpoints/Model1
INFO:tensorflow:Assets written to: donors_choose/tensorboard_checkpoints/Mode
l1/assets
464/464 [==============================] - 17s 36ms/step - loss: 0.5944 - acc
uracy: 0.7045 - roc_auc_scr: 0.7544 - val_loss: 0.5758 - val_accuracy: 0.7092
- val_roc_auc_scr: 0.7737
Epoch 5/7
464/464 [==============================] - ETA: 0s - loss: 0.5801 - accuracy:
0.7169 - roc_auc_scr: 0.7702
Epoch 00005: val_roc_auc_scr improved from 0.77375 to 0.78255, saving model t
o donors_choose/tensorboard_checkpoints/Model1
INFO:tensorflow:Assets written to: donors_choose/tensorboard_checkpoints/Mode
l1/assets
464/464 [==============================] - 18s 38ms/step - loss: 0.5801 - acc
uracy: 0.7169 - roc_auc_scr: 0.7702 - val_loss: 0.5629 - val_accuracy: 0.7221
- val_roc_auc_scr: 0.7826
Epoch 6/7
462/464 [============================>.] - ETA: 0s - loss: 0.5653 - accuracy:
0.7313 - roc_auc_scr: 0.7859
Epoch 00006: val_roc_auc_scr improved from 0.78255 to 0.79262, saving model t
o donors_choose/tensorboard_checkpoints/Model1
INFO:tensorflow:Assets written to: donors_choose/tensorboard_checkpoints/Mode
l1/assets
464/464 [==============================] - 17s 37ms/step - loss: 0.5654 - acc
uracy: 0.7311 - roc_auc_scr: 0.7857 - val_loss: 0.5531 - val_accuracy: 0.7282
- val_roc_auc_scr: 0.7926
Epoch 7/7
462/464 [============================>.] - ETA: 0s - loss: 0.5501 - accuracy:
0.7432 - roc_auc_scr: 0.8006
Epoch 00007: val_roc_auc_scr improved from 0.79262 to 0.80338, saving model t
o donors_choose/tensorboard_checkpoints/Model1
INFO:tensorflow:Assets written to: donors_choose/tensorboard_checkpoints/Mode
l1/assets
464/464 [==============================] - 17s 38ms/step - loss: 0.5501 - acc
uracy: 0.7430 - roc_auc_scr: 0.8005 - val_loss: 0.5407 - val_accuracy: 0.7382
- val_roc_auc_scr: 0.8034
```

Model1: show structure

In [ ]:
```
plot_model(M1, show_shapes=True)
```

Out[ ]:



Model1: Test scores

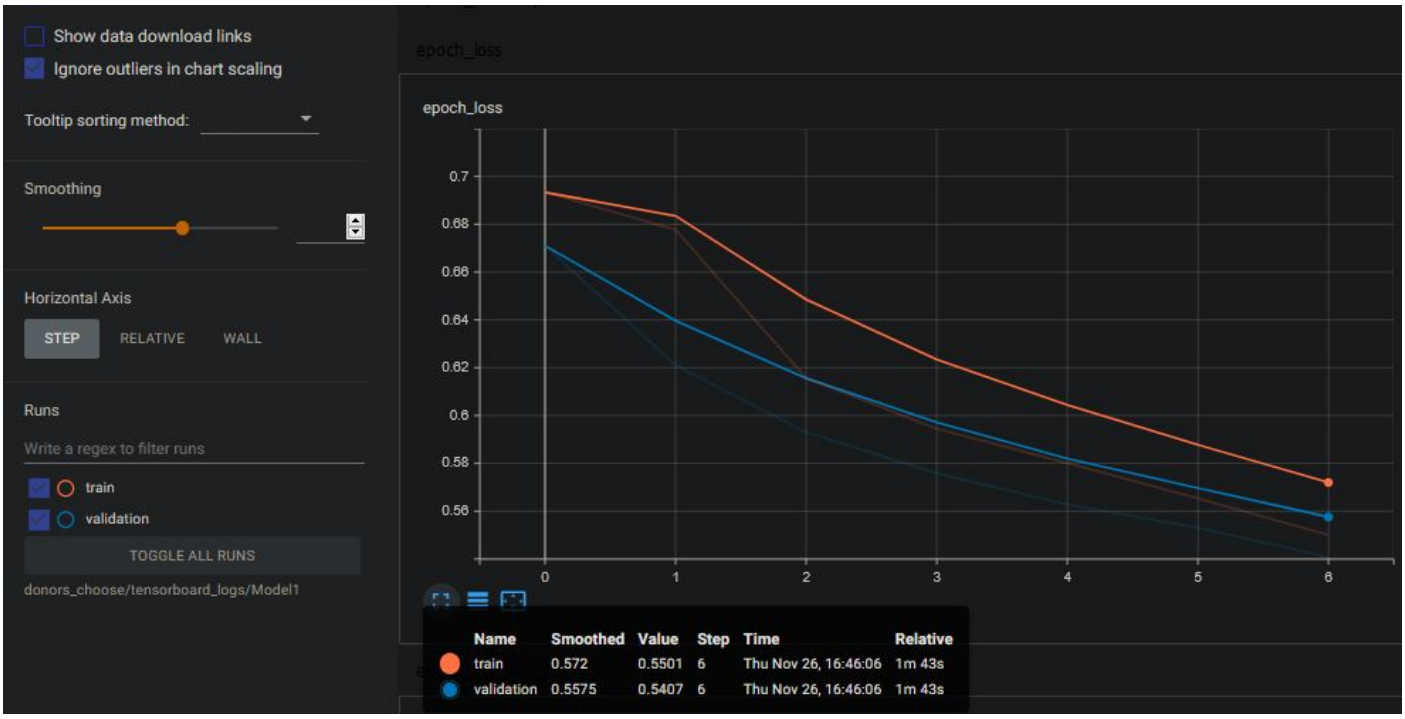
In [ ]:
```
M1.evaluate({'inp1': test_essay_padded, 'inp2': test_ss,'inp3': test_tp, 'inp
4':test_pgc, 'inp5':test_cc, 'inp6':test_cs, 'inp7':test_num}, Y_test_cat, bat
ch_size=256)
```

```
145/145 [==============================] - 2s 16ms/step - loss: 0.5374 - accu
racy: 0.7396 - roc_auc_scr: 0.8065
```

Out[ ]:
```
[0.5374088287353516, 0.7395572066307068, 0.8064515590667725]
```

Model1 : Train and Validation metrics plots Tensorboard

In [ ]:
```
%load_ext tensorboard
%tensorboard --logdir 'donors_choose/tensorboard_logs/Model1'
```



Model2: DATA Preprocessing

In model2 we are choosing important idf values. Features which are in the middle 50 percentile range i.e. features whose idf values are between 25th to 75th percentile of idf values when sorted numerically. Rest other features are same as model1.

In [ ]:
```
# essay features : Select Important features

tfv = TfidfVectorizer()
train_idf_matrix = tfv.fit_transform(X_train.essay)
test_idf_matrix = tfv.transform(X_test.essay)

features = tfv.get_feature_names()
scores = tfv.idf_

indices = tfv.idf_.argsort()
total = len(indices)

# find important tfidf features (Those which are in between 25th and 75th perc
entile.)

important_indices = np.argsort(tfv.idf_)[int(total*0.25):int(total*0.75)]
idf_scores = sorted(scores)[int(total*0.25):int(total*0.75)]

important_features = []
for ind in important_indices:
  important_features.append(features[ind])

######### essay DATA Preprocessing ############

MAX_NUM_WORDS = len(important_indices)  #diff
EMBEDDING_DIM = 300
MAX_LENGTH = 100

embedding_matrix2 = np.zeros((MAX_NUM_WORDS + 1, EMBEDDING_DIM))

for i, f in enumerate(important_features):
  embedding_vector = embeddings_index.get(f)
  if embedding_vector is not None:
    embedding_matrix2[i] = embedding_vector


tokenizer = Tokenizer(MAX_NUM_WORDS)
tokenizer.fit_on_texts(important_features)

train_essay2 = tokenizer.texts_to_sequences(X_train.essay)
test_essay2 = tokenizer.texts_to_sequences(X_test.essay)

train_essay_padded2 = pad_sequences(train_essay2, maxlen=MAX_LENGTH)
test_essay_padded2 = pad_sequences(test_essay2, maxlen=MAX_LENGTH)

print('embedding_matrix2.shape:',embedding_matrix2.shape)
print('train_essay_padded2.shape:',train_essay_padded2.shape)
print('test_essay_padded2.shape:',test_essay_padded2.shape)
```

```
embedding_matrix2.shape: (26284, 300)
train_essay_padded2.shape: (148329, 100)
test_essay_padded2.shape: (37083, 100)
```

Model2: Architecture

In [ ]:
```
#### Essay Input ####
essay_i2 = Input(shape=(MAX_LENGTH,), name='inp1')
essay_eo2 = Embedding(len(important_indices) + 1, EMBEDDING_DIM, weights=[embe
dding_matrix2], input_length=(MAX_LENGTH,), trainable=False)(essay_i2)
essay_lo2 = LSTM(32)(essay_eo2)
essay_fo2 = Flatten()(essay_lo2)

#### School state Input ####
ss_i = Input(shape=(train_ss.shape[1],), name='inp2')
ss_eo = Embedding(n_classes_ss+1, 3, input_length=(train_ss.shape[1],))(ss_i)
ss_fo = Flatten()(ss_eo)

#### Teacher prefix Input ####
tp_i = Input(shape=(train_tp.shape[1],), name='inp3')
tp_eo = Embedding(n_classes_tp+1, 2, input_length=(train_tp.shape[1],))(tp_i)
tp_fo = Flatten()(tp_eo)

#### Project Grade category Input ####
pgc_i = Input(shape=(train_pgc.shape[1],), name='inp4')
pgc_eo = Embedding(n_classes_pgc+1, 2, input_length=(train_pgc.shape[1],))(pgc
_i)
pgc_fo = Flatten()(pgc_eo)

#### Clean Categories Input ####
cc_i = Input(shape=(train_cc.shape[1],), name='inp5')
cc_eo = Embedding(n_classes_cc+1, int((n_classes_cc+1)**0.25)+1, input_length=
(train_cc.shape[1],))(cc_i)
cc_fo = Flatten()(cc_eo)

#### Clean sub Categories Input ####
cs_i = Input(shape=(train_cs.shape[1],), name='inp6')
cs_eo = Embedding(n_classes_cs+1 , int((n_classes_cs+1)**0.25)+1, input_length
=(train_cs.shape[1],))(cs_i)
cs_fo = Flatten()(cs_eo)

#### Numeric Input ####
num_i = Input(shape=(train_num.shape[1],), name='inp7')
num_fo = Dense(16, activation='relu')(num_i) #

#### Concatenate all the inputs
concatenated_input = Concatenate()([essay_fo2, ss_fo, tp_fo, pgc_fo, cc_fo, cs
_fo, num_fo])

#### Define intermediate dense layers and output layer
x = Dense(64, activation='relu', kernel_initializer='he_normal', bias_initiali
zer='random_normal')(concatenated_input)
x = Dropout(0.5)(x)
x = Dense(32, activation='relu', kernel_initializer='he_normal', bias_initiali
zer='random_normal')(x)
x = Dropout(0.5)(x)
x = Dense(16, activation='relu', kernel_initializer='he_normal', bias_initiali
zer='random_normal')(x)
x = Dropout(0.5)(x)
output = Dense(2, activation='softmax')(x)

# Define model
```

```
M2 = Model(inputs=[essay_i2, ss_i, tp_i, pgc_i, cc_i, cs_i, num_i], outputs=ou
tput)
adm = Adam(learning_rate=0.001)
M2.compile(loss='categorical_crossentropy', metrics=['accuracy',roc_auc_scr],
optimizer=adm)
```

Model2: Training

In [ ]:
```
checkpoint_callback2 = ModelCheckpoint("donors_choose/tensorboard_checkpoints/
Model2", monitor='val_roc_auc_scr', verbose=1, save_best_only=True, mode='max'
)
earlystop_callback2  = EarlyStopping(monitor='val_roc_auc_scr', verbose=1,  mo
de='max', patience=3)
tensorboard_callback2= TensorBoard("donors_choose/tensorboard_logs/Model2", hi
stogram_freq=1)

history = M2.fit({'inp1': train_essay_padded2, 'inp2': train_ss,'inp3': train_
tp, 'inp4':train_pgc, 'inp5':train_cc, 'inp6':train_cs, 'inp7':train_num},Y_tr
ain_cat,validation_split=0.2 , epochs=15, batch_size=256, callbacks=[tensorboa
rd_callback2, checkpoint_callback2, earlystop_callback2],  shuffle=False , ver
bose=1)
```

```
Epoch 1/15
  1/464 [..............................] - ETA: 0s - loss: 0.7471 - accuracy:
0.5195 - roc_auc_scr: 0.5194WARNING:tensorflow:From /usr/local/lib/python3.6/
dist-packages/tensorflow/python/ops/summary_ops_v2.py:1277: stop (from tensor
flow.python.eager.profiler) is deprecated and will be removed after 2020-07-0
1.
Instructions for updating:
use `tf.profiler.experimental.stop` instead.
  2/464 [..............................] - ETA: 30s - loss: 0.7586 - accurac
y: 0.4961 - roc_auc_scr: 0.4769WARNING:tensorflow:Callbacks method `on_train_
batch_end` is slow compared to the batch time (batch time: 0.0307s vs `on_tra
in_batch_end` time: 0.1024s). Check your callbacks.
463/464 [============================>.] - ETA: 0s - loss: 0.6949 - accuracy:
0.5020 - roc_auc_scr: 0.5025
Epoch 00001: val_roc_auc_scr improved from -inf to 0.49779, saving model to d
onors_choose/tensorboard_checkpoints/Model2
WARNING:tensorflow:From /usr/local/lib/python3.6/dist-packages/tensorflow/pyt
hon/training/tracking/tracking.py:111: Model.state_updates (from tensorflow.p
ython.keras.engine.training) is deprecated and will be removed in a future ve
rsion.
Instructions for updating:
This property should not be used in TensorFlow 2.0, as updates are applied au
tomatically.
WARNING:tensorflow:From /usr/local/lib/python3.6/dist-packages/tensorflow/pyt
hon/training/tracking/tracking.py:111: Layer.updates (from tensorflow.python.
keras.engine.base_layer) is deprecated and will be removed in a future versio
n.
Instructions for updating:
This property should not be used in TensorFlow 2.0, as updates are applied au
tomatically.
INFO:tensorflow:Assets written to: donors_choose/tensorboard_checkpoints/Mode
l2/assets
464/464 [==============================] - 17s 36ms/step - loss: 0.6949 - acc
uracy: 0.5019 - roc_auc_scr: 0.5024 - val_loss: 0.6931 - val_accuracy: 0.5027
- val_roc_auc_scr: 0.4978
Epoch 2/15
462/464 [============================>.] - ETA: 0s - loss: 0.6932 - accuracy:
0.4996 - roc_auc_scr: 0.5013
Epoch 00002: val_roc_auc_scr improved from 0.49779 to 0.50736, saving model t
o donors_choose/tensorboard_checkpoints/Model2
INFO:tensorflow:Assets written to: donors_choose/tensorboard_checkpoints/Mode
l2/assets
464/464 [==============================] - 16s 35ms/step - loss: 0.6932 - acc
uracy: 0.4997 - roc_auc_scr: 0.5013 - val_loss: 0.6930 - val_accuracy: 0.5049
- val_roc_auc_scr: 0.5074
Epoch 3/15
464/464 [==============================] - ETA: 0s - loss: 0.6922 - accuracy:
0.5099 - roc_auc_scr: 0.5134
Epoch 00003: val_roc_auc_scr improved from 0.50736 to 0.56167, saving model t
o donors_choose/tensorboard_checkpoints/Model2
INFO:tensorflow:Assets written to: donors_choose/tensorboard_checkpoints/Mode
l2/assets
464/464 [==============================] - 17s 36ms/step - loss: 0.6922 - acc
uracy: 0.5099 - roc_auc_scr: 0.5134 - val_loss: 0.6881 - val_accuracy: 0.5343
- val_roc_auc_scr: 0.5617
Epoch 4/15
463/464 [============================>.] - ETA: 0s - loss: 0.6877 - accuracy:
```

```
0.5364 - roc_auc_scr: 0.5472
Epoch 00004: val_roc_auc_scr improved from 0.56167 to 0.59391, saving model t
o donors_choose/tensorboard_checkpoints/Model2
INFO:tensorflow:Assets written to: donors_choose/tensorboard_checkpoints/Mode
l2/assets
464/464 [==============================] - 16s 34ms/step - loss: 0.6877 - acc
uracy: 0.5365 - roc_auc_scr: 0.5474 - val_loss: 0.6825 - val_accuracy: 0.5525
- val_roc_auc_scr: 0.5939
Epoch 5/15
462/464 [============================>.] - ETA: 0s - loss: 0.6798 - accuracy:
0.5606 - roc_auc_scr: 0.5821
Epoch 00005: val_roc_auc_scr improved from 0.59391 to 0.61736, saving model t
o donors_choose/tensorboard_checkpoints/Model2
INFO:tensorflow:Assets written to: donors_choose/tensorboard_checkpoints/Mode
l2/assets
464/464 [==============================] - 17s 36ms/step - loss: 0.6797 - acc
uracy: 0.5607 - roc_auc_scr: 0.5822 - val_loss: 0.6730 - val_accuracy: 0.5724
- val_roc_auc_scr: 0.6174
Epoch 6/15
462/464 [============================>.] - ETA: 0s - loss: 0.6671 - accuracy:
0.5843 - roc_auc_scr: 0.6155
Epoch 00006: val_roc_auc_scr improved from 0.61736 to 0.64968, saving model t
o donors_choose/tensorboard_checkpoints/Model2
INFO:tensorflow:Assets written to: donors_choose/tensorboard_checkpoints/Mode
l2/assets
464/464 [==============================] - 16s 34ms/step - loss: 0.6670 - acc
uracy: 0.5844 - roc_auc_scr: 0.6156 - val_loss: 0.6589 - val_accuracy: 0.5935
- val_roc_auc_scr: 0.6497
Epoch 7/15
464/464 [==============================] - ETA: 0s - loss: 0.6482 - accuracy:
0.6075 - roc_auc_scr: 0.6491
Epoch 00007: val_roc_auc_scr improved from 0.64968 to 0.67182, saving model t
o donors_choose/tensorboard_checkpoints/Model2
INFO:tensorflow:Assets written to: donors_choose/tensorboard_checkpoints/Mode
l2/assets
464/464 [==============================] - 16s 35ms/step - loss: 0.6482 - acc
uracy: 0.6075 - roc_auc_scr: 0.6491 - val_loss: 0.6432 - val_accuracy: 0.6099
- val_roc_auc_scr: 0.6718
Epoch 8/15
462/464 [============================>.] - ETA: 0s - loss: 0.6301 - accuracy:
0.6273 - roc_auc_scr: 0.6773
Epoch 00008: val_roc_auc_scr improved from 0.67182 to 0.68941, saving model t
o donors_choose/tensorboard_checkpoints/Model2
INFO:tensorflow:Assets written to: donors_choose/tensorboard_checkpoints/Mode
l2/assets
464/464 [==============================] - 16s 35ms/step - loss: 0.6301 - acc
uracy: 0.6274 - roc_auc_scr: 0.6774 - val_loss: 0.6300 - val_accuracy: 0.6309
- val_roc_auc_scr: 0.6894
Epoch 9/15
462/464 [============================>.] - ETA: 0s - loss: 0.6110 - accuracy:
0.6442 - roc_auc_scr: 0.7017
Epoch 00009: val_roc_auc_scr improved from 0.68941 to 0.71195, saving model t
o donors_choose/tensorboard_checkpoints/Model2
INFO:tensorflow:Assets written to: donors_choose/tensorboard_checkpoints/Mode
l2/assets
464/464 [==============================] - 16s 33ms/step - loss: 0.6111 - acc
uracy: 0.6442 - roc_auc_scr: 0.7016 - val_loss: 0.6157 - val_accuracy: 0.6483
```

```
- val_roc_auc_scr: 0.7120
Epoch 10/15
463/464 [============================>.] - ETA: 0s - loss: 0.5916 - accuracy:
0.6598 - roc_auc_scr: 0.7251
Epoch 00010: val_roc_auc_scr improved from 0.71195 to 0.72323, saving model t
o donors_choose/tensorboard_checkpoints/Model2
INFO:tensorflow:Assets written to: donors_choose/tensorboard_checkpoints/Mode
l2/assets
464/464 [==============================] - 16s 35ms/step - loss: 0.5916 - acc
uracy: 0.6598 - roc_auc_scr: 0.7251 - val_loss: 0.6098 - val_accuracy: 0.6497
- val_roc_auc_scr: 0.7232
Epoch 11/15
462/464 [============================>.] - ETA: 0s - loss: 0.5790 - accuracy:
0.6692 - roc_auc_scr: 0.7393
Epoch 00011: val_roc_auc_scr improved from 0.72323 to 0.73230, saving model t
o donors_choose/tensorboard_checkpoints/Model2
INFO:tensorflow:Assets written to: donors_choose/tensorboard_checkpoints/Mode
l2/assets
464/464 [==============================] - 15s 33ms/step - loss: 0.5790 - acc
uracy: 0.6692 - roc_auc_scr: 0.7393 - val_loss: 0.6069 - val_accuracy: 0.6464
- val_roc_auc_scr: 0.7323
Epoch 12/15
462/464 [============================>.] - ETA: 0s - loss: 0.5644 - accuracy:
0.6776 - roc_auc_scr: 0.7546
Epoch 00012: val_roc_auc_scr improved from 0.73230 to 0.74140, saving model t
o donors_choose/tensorboard_checkpoints/Model2
INFO:tensorflow:Assets written to: donors_choose/tensorboard_checkpoints/Mode
l2/assets
464/464 [==============================] - 16s 35ms/step - loss: 0.5644 - acc
uracy: 0.6777 - roc_auc_scr: 0.7544 - val_loss: 0.6077 - val_accuracy: 0.6479
- val_roc_auc_scr: 0.7414
Epoch 13/15
463/464 [============================>.] - ETA: 0s - loss: 0.5544 - accuracy:
0.6843 - roc_auc_scr: 0.7648
Epoch 00013: val_roc_auc_scr improved from 0.74140 to 0.75061, saving model t
o donors_choose/tensorboard_checkpoints/Model2
INFO:tensorflow:Assets written to: donors_choose/tensorboard_checkpoints/Mode
l2/assets
464/464 [==============================] - 16s 35ms/step - loss: 0.5543 - acc
uracy: 0.6843 - roc_auc_scr: 0.7649 - val_loss: 0.6006 - val_accuracy: 0.6615
- val_roc_auc_scr: 0.7506
Epoch 14/15
463/464 [============================>.] - ETA: 0s - loss: 0.5436 - accuracy:
0.6922 - roc_auc_scr: 0.7751
Epoch 00014: val_roc_auc_scr improved from 0.75061 to 0.75596, saving model t
o donors_choose/tensorboard_checkpoints/Model2
INFO:tensorflow:Assets written to: donors_choose/tensorboard_checkpoints/Mode
l2/assets
464/464 [==============================] - 16s 33ms/step - loss: 0.5435 - acc
uracy: 0.6923 - roc_auc_scr: 0.7752 - val_loss: 0.6053 - val_accuracy: 0.6598
- val_roc_auc_scr: 0.7560
Epoch 15/15
462/464 [============================>.] - ETA: 0s - loss: 0.5365 - accuracy:
0.6945 - roc_auc_scr: 0.7803
Epoch 00015: val_roc_auc_scr improved from 0.75596 to 0.76074, saving model t
o donors_choose/tensorboard_checkpoints/Model2
INFO:tensorflow:Assets written to: donors_choose/tensorboard_checkpoints/Mode
```

```
l2/assets
464/464 [==============================] - 16s 35ms/step - loss: 0.5365 - acc
uracy: 0.6945 - roc_auc_scr: 0.7803 - val_loss: 0.6075 - val_accuracy: 0.6735
- val_roc_auc_scr: 0.7607
```

Model2: show structure

In [ ]:
```
plot_model(M2, show_shapes=True)
```

Out[ ]:

Model2: Test scores

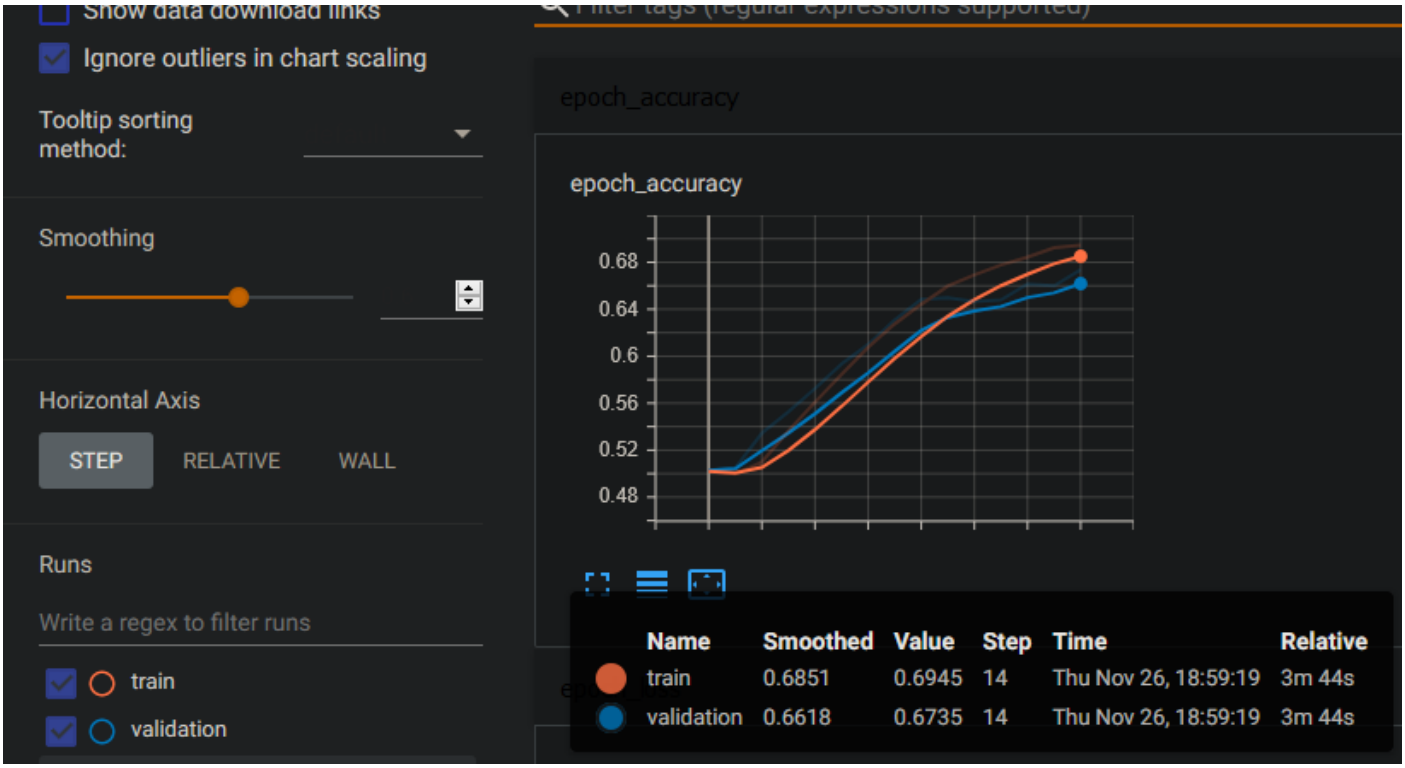
In [ ]:
```
M2.evaluate({'inp1': test_essay_padded2, 'inp2': test_ss,'inp3': test_tp, 'inp
4':test_pgc, 'inp5':test_cc, 'inp6':test_cs, 'inp7':test_num}, Y_test_cat, bat
ch_size=256)
```

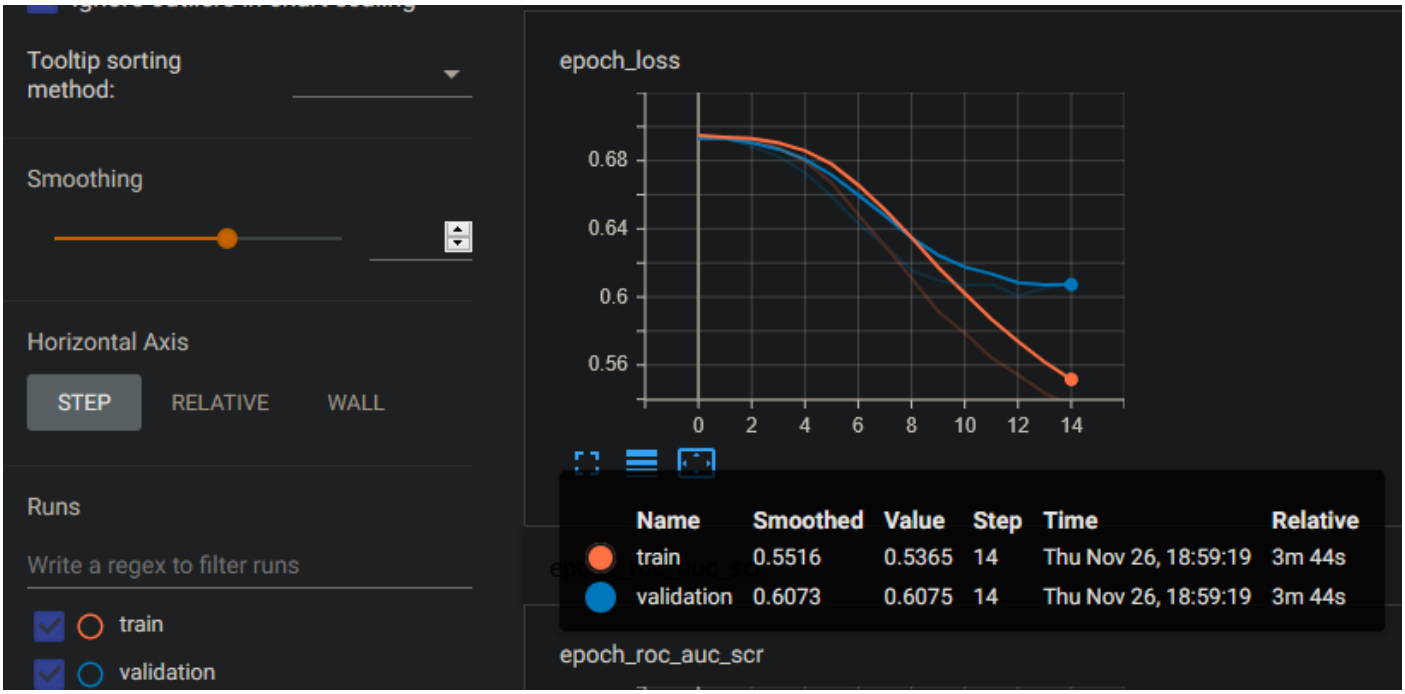
```
145/145 [==============================] - 2s 15ms/step - loss: 0.6021 - accu
racy: 0.6691 - roc_auc_scr: 0.7646
```

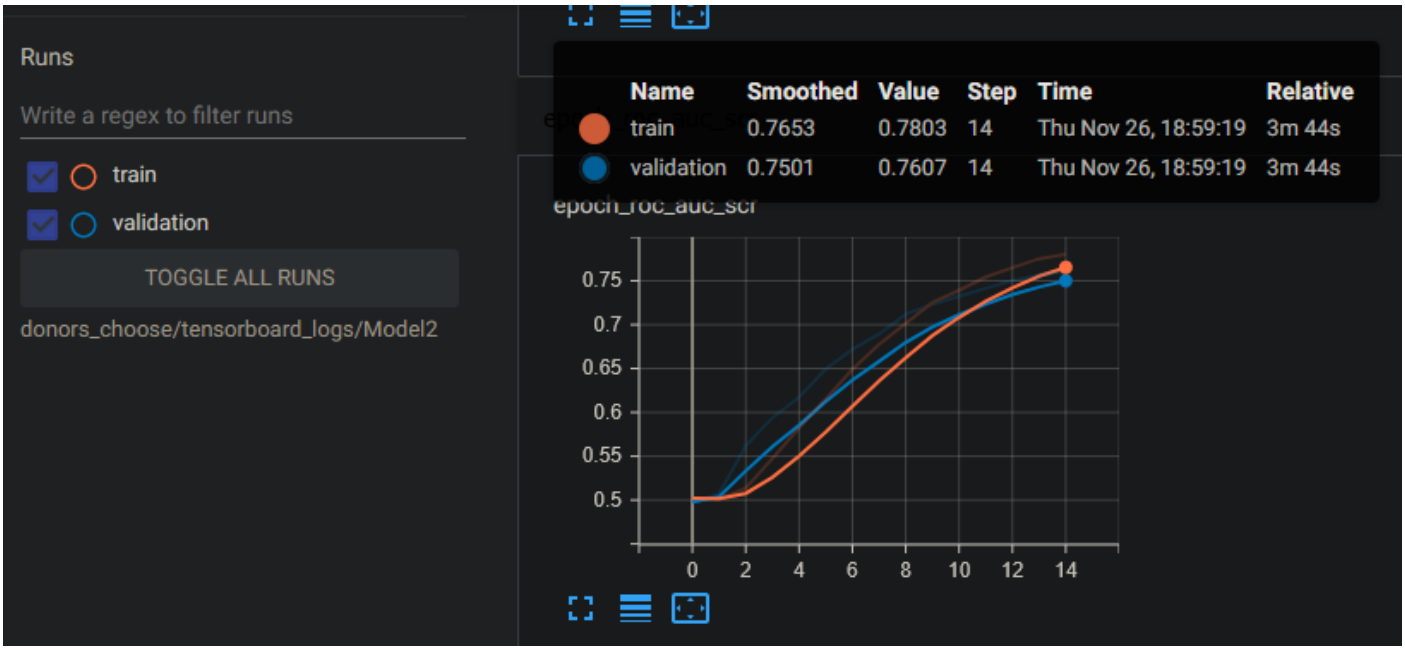
Out[ ]:
```
[0.6021195650100708, 0.6690666675567627, 0.7646275758743286]
```

Model2: Train and Validation metrics plots Tensorboard

In [ ]:
```
%load_ext tensorboard
%tensorboard --logdir 'donors_choose/tensorboard_logs/Model2'
```


Show data download links
Ignore outliers in chart scaling
Tooltip sorting method:
Smoothing
Horizontal Axis
STEP
RELATIVE
WALL
Runs
Write a regex to filter runs
train
validation
epoch_accuracy
epoch_accuracy
0.68
0.64
0.6
0.56
0.52
0.48
Name Smoothed Value Step Time Relative
train 0.6851 0.6945 14 Thu Nov 26, 18:59:19 3m 44s
validation 0.6618 0.6735 14 Thu Nov 26, 18:59:19 3m 44s


Tooltip sorting method:
Smoothing
Horizontal Axis
STEP
RELATIVE
WALL
Runs
Write a regex to filter runs
train
validation
epoch_loss
0.68
0.64
0.6
0.56
0 2 4 6 8 10 12 14
Name Smoothed Value Step Time Relative
train 0.5516 0.5365 14 Thu Nov 26, 18:59:19 3m 44s
validation 0.6073 0.6075 14 Thu Nov 26, 18:59:19 3m 44s
epoch_roc_auc_scr



Model3: DATA Preprocessing

In [ ]:
```
######### essay DATA Preprocessing ###########

MAX_NUM_WORDS = 16000
MAX_LENGTH = 100
EMBEDDING_DIM = 300

tokenizer = Tokenizer(MAX_NUM_WORDS)
tokenizer.fit_on_texts(X_train.essay)

train_essay = tokenizer.texts_to_sequences(X_train.essay)
test_essay = tokenizer.texts_to_sequences(X_test.essay)

train_essay_padded = pad_sequences(train_essay,maxlen=MAX_LENGTH)
test_essay_padded = pad_sequences(test_essay,maxlen=MAX_LENGTH)

word_index = tokenizer.word_index
embedding_matrix1 = np.zeros((len(word_index) + 1, EMBEDDING_DIM))

for word, i in word_index.items():
  embedding_vector = embeddings_index.get(word)
  if embedding_vector is not None:
    embedding_matrix1[i] = embedding_vector



######### school_state DATA Preprocessing ###########
ohe = OneHotEncoder()
ohe.fit(X_train.school_state.values.reshape(-1,1))
train_ss = le.transform(X_train.school_state.values.reshape(-1,1)).toarray()
test_ss = le.transform(X_test.school_state.values.reshape(-1,1)).toarray()


######### teacher_prefix DATA Preprocessing ###########
ohe = OneHotEncoder()
ohe.fit(X_train.teacher_prefix.values.reshape(-1,1))
train_tp = le.transform(X_train.teacher_prefix.values.reshape(-1,1)).toarray()
test_tp = le.transform(X_test.teacher_prefix.values.reshape(-1,1)).toarray()



#########project grade category DATA Preprocessing ###########
ohe = OneHotEncoder()
ohe.fit(X_train.project_grade_category.values.reshape(-1,1))
train_pgc = le.transform(X_train.project_grade_category.values.reshape(-1,1)).
toarray()
test_pgc = le.transform(X_test.project_grade_category.values.reshape(-1,1)).to
array()



######### clean_categories DATA Preprocessing ###########
# we are using X to fit the model instead of X_train since categories are diff
erent for train and test data
ohe = OneHotEncoder()
ohe.fit(X['clean_categories'].values.reshape(-1,1))
train_cc = ohe.transform(X_train['clean_categories'].values.reshape(-1,1)).toa
```

```
rray()
test_cc = ohe.transform(X_test['clean_categories'].values.reshape(-1,1)).toarr
ay()

#MAX_SEQUENCE_LENGTH_cc = X.clean_categories.apply(lambda x:len(x.split())).ma
x()
n_classes_cc = len(np.unique(X.clean_categories))



######### clean_subcategories DATA Preprocessing ###########
# we are using X to fit the model instead of X_train since categories are diff
erent for train and test data
ohe = OneHotEncoder()
ohe.fit(X['clean_subcategories'].values.reshape(-1,1))
train_cs = ohe.transform(X_train['clean_subcategories'].values.reshape(-1,1)).
toarray()
test_cs = ohe.transform(X_test['clean_subcategories'].values.reshape(-1,1)).to
array()
n_classes_cs = len(np.unique(X.clean_subcategories))

#MAX_SEQUENCE_LENGTH_cs = X.clean_subcategories.apply(lambda x:len(x.split
())).max()


######### train_num DATA Preprocessing ###########
mms = MinMaxScaler()
train_num = mms.fit_transform(X_train[['teacher_number_of_previously_posted_pr
ojects','price','resource_summary_contains_numeric_digits']])
test_num = mms.transform(X_test[['teacher_number_of_previously_posted_project
s','price','resource_summary_contains_numeric_digits']])


print('embedding_matrix1.shape:',embedding_matrix1.shape)
print('\n')
print('train_essay_padded.shape',train_essay_padded.shape)
print('train_ss.shape',train_ss.shape)
print('train_tp.shape',train_tp.shape)
print('train_pgc.shape', train_pgc.shape)
print('train_cc.shape', train_cc.shape)
print('train_cs.shape', train_cs.shape)
print('train_num.shape', train_num.shape)

print('\n')
print('test_essay_padded.shape',test_essay_padded.shape)
print('test_ss.shape',test_ss.shape)
print('test_tp.shape',test_tp.shape)
print('test_pgc.shape', test_pgc.shape)
print('test_cc.shape', test_cc.shape)
print('test_cs.shape', test_cs.shape)
print('test_num.shape', test_num.shape)
```

```
embedding_matrix1.shape: (52516, 300)


train_essay_padded.shape (148329, 100)
train_ss.shape (148329, 51)
train_tp.shape (148329, 5)
train_pgc.shape (148329, 4)
train_cc.shape (148329, 51)
train_cs.shape (148329, 401)
train_num.shape (148329, 3)


test_essay_padded.shape (37083, 100)
test_ss.shape (37083, 51)
test_tp.shape (37083, 5)
test_pgc.shape (37083, 4)
test_cc.shape (37083, 51)
test_cs.shape (37083, 401)
test_num.shape (37083, 3)
```

Model3: Architecture

In [ ]:
```
#### Essay Input ####
essay_i = Input(shape=(MAX_LENGTH,), name='inp1')
essay_eo = Embedding(len(word_index) + 1, EMBEDDING_DIM, weights=[embedding_ma
trix1], input_length=(MAX_LENGTH,), trainable=False)(essay_i)
essay_lo = LSTM(32)(essay_eo)
essay_fo = Flatten()(essay_lo)

#### School state Input ####
ss_i = Input(shape=(train_ss.shape[1],), name='inp2')
ss_eo = Embedding(train_ss.shape[1]+1, 3, input_length=(train_ss.shape[1],))(s
s_i)
ss_fo = Flatten()(ss_eo)

#### Teacher prefix Input ####
tp_i = Input(shape=(train_tp.shape[1],), name='inp3')
tp_eo = Embedding(train_tp.shape[1] , 2, input_length=(train_tp.shape[1],))(tp
_i)
tp_fo = Flatten()(tp_eo)

#### Project Grade category Input ####
pgc_i = Input(shape=(train_pgc.shape[1],), name='inp4')
pgc_eo = Embedding(train_pgc.shape[1], 2, input_length=(train_pgc.shape[1],))(
pgc_i)
pgc_fo = Flatten()(pgc_eo)

#### Clean Categories Input ####
cc_i = Input(shape=(train_cc.shape[1],), name='inp5')
cc_eo = Embedding(n_classes_cc , 3, input_length=(train_cc.shape[1],))(cc_i)
cc_fo = Flatten()(cc_eo)

#### Clean sub Categories Input ####
cs_i = Input(shape=(train_cs.shape[1],), name='inp6')
cs_eo = Embedding(n_classes_cs , 3, input_length=(train_cs.shape[1],))(cs_i)
cs_fo = Flatten()(cs_eo)

#### Numeric Input ####
num_i = Input(shape=(train_num.shape[1],), name='inp7')
num_fo = Dense(16, activation='relu')(num_i)


################################################################################
#############

ott_i = Concatenate()([ss_fo, tp_fo, pgc_fo, cc_fo, cs_fo, num_fo])
ott_i3 = tensorflow.expand_dims(ott_i, 1)
ott = Conv1D(64,7,padding='same', activation='relu', kernel_initializer='he_no
rmal', bias_initializer='random_normal',input_shape=((ott_i3.shape)))(ott_i3)
#
x = Conv1D(32,5, padding='same', activation='relu', kernel_initializer='he_nor
mal', bias_initializer='random_normal', )(ott)
x = Dropout(0.5)(x)
x = Conv1D(16,3, padding='same', activation='relu', kernel_initializer='he_nor
mal', bias_initializer='random_normal',)(x)

flatten_layer = Flatten()
ott_fo = flatten_layer(x)
```

```
##############################################################################
###########


Model31 = Model(inputs=essay_i, outputs=essay_fo)
Model32 = Model(inputs=[ss_i, tp_i, pgc_i, cc_i, cs_i, num_i], outputs=ott_fo)

total_concatenated_input = Concatenate()([Model31.output, Model32.output])

x = Dense(64, activation='relu', kernel_initializer='he_normal', bias_initiali
zer='random_normal')(total_concatenated_input)
x = Dropout(0.5)(x)
x = Dense(32, activation='relu', kernel_initializer='he_normal', bias_initiali
zer='random_normal')(x)
x = Dropout(0.5)(x)
x = Dense(16, activation='relu', kernel_initializer='he_normal', bias_initiali
zer='random_normal')(x)
x = Dropout(0.5)(x)
output = Dense(2, activation='softmax')(x)

M3 = Model(inputs=[Model31.input, Model32.input], outputs=output)
adm = Adam()
M3.compile(loss='categorical_crossentropy', metrics=['accuracy',roc_auc_scr],
optimizer=adm)
```

Model3: Training

In [ ]:
```
checkpoint_callback3  = ModelCheckpoint("donors_choose/tensorboard_checkpoint
s/Model3", monitor='val_roc_auc_scr', verbose=1, save_best_only=True, mode='ma
x')
earlystop_callback3  =  EarlyStopping(monitor='val_roc_auc_scr', verbose=1,  m
ode='max', patience=3)
tensorboard_callback3 = TensorBoard("donors_choose/tensorboard_logs/Model3", h
istogram_freq=1)

history = M3.fit({'inp1': train_essay_padded, 'inp2': train_ss,'inp3': train_t
p, 'inp4':train_pgc, 'inp5':train_cc, 'inp6':train_cs, 'inp7':train_num},Y_tra
in_cat,validation_split=0.2 , epochs=8, batch_size=256, callbacks=[tensorboard
_callback3, checkpoint_callback3, earlystop_callback3],  shuffle=False , verbo
se=1)
```

```
Epoch 1/8
  1/464 [..............................] - ETA: 0s - loss: 0.9785 - accuracy:
0.4648 - roc_auc_scr: 0.4498WARNING:tensorflow:From /usr/local/lib/python3.6/
dist-packages/tensorflow/python/ops/summary_ops_v2.py:1277: stop (from tensor
flow.python.eager.profiler) is deprecated and will be removed after 2020-07-0
1.
Instructions for updating:
use `tf.profiler.experimental.stop` instead.
  2/464 [..............................] - ETA: 29s - loss: 0.8803 - accurac
y: 0.4883 - roc_auc_scr: 0.4911WARNING:tensorflow:Callbacks method `on_train_
batch_end` is slow compared to the batch time (batch time: 0.0315s vs `on_tra
in_batch_end` time: 0.0949s). Check your callbacks.
464/464 [==============================] - ETA: 0s - loss: 0.6959 - accuracy:
0.5031 - roc_auc_scr: 0.5055
Epoch 00001: val_roc_auc_scr improved from -inf to 0.52847, saving model to d
onors_choose/tensorboard_checkpoints/Model3
WARNING:tensorflow:From /usr/local/lib/python3.6/dist-packages/tensorflow/pyt
hon/training/tracking/tracking.py:111: Model.state_updates (from tensorflow.p
ython.keras.engine.training) is deprecated and will be removed in a future ve
rsion.
Instructions for updating:
This property should not be used in TensorFlow 2.0, as updates are applied au
tomatically.
WARNING:tensorflow:From /usr/local/lib/python3.6/dist-packages/tensorflow/pyt
hon/training/tracking/tracking.py:111: Layer.updates (from tensorflow.python.
keras.engine.base_layer) is deprecated and will be removed in a future versio
n.
Instructions for updating:
This property should not be used in TensorFlow 2.0, as updates are applied au
tomatically.
INFO:tensorflow:Assets written to: donors_choose/tensorboard_checkpoints/Mode
l3/assets
464/464 [==============================] - 19s 42ms/step - loss: 0.6959 - acc
uracy: 0.5031 - roc_auc_scr: 0.5055 - val_loss: 0.6932 - val_accuracy: 0.4975
- val_roc_auc_scr: 0.5285
Epoch 2/8
464/464 [==============================] - ETA: 0s - loss: 0.6772 - accuracy:
0.5649 - roc_auc_scr: 0.5853
Epoch 00002: val_roc_auc_scr improved from 0.52847 to 0.72047, saving model t
o donors_choose/tensorboard_checkpoints/Model3
INFO:tensorflow:Assets written to: donors_choose/tensorboard_checkpoints/Mode
l3/assets
464/464 [==============================] - 18s 40ms/step - loss: 0.6772 - acc
uracy: 0.5649 - roc_auc_scr: 0.5853 - val_loss: 0.6340 - val_accuracy: 0.6676
- val_roc_auc_scr: 0.7205
Epoch 3/8
462/464 [============================>.] - ETA: 0s - loss: 0.6254 - accuracy:
0.6739 - roc_auc_scr: 0.7164
Epoch 00003: val_roc_auc_scr improved from 0.72047 to 0.75041, saving model t
o donors_choose/tensorboard_checkpoints/Model3
INFO:tensorflow:Assets written to: donors_choose/tensorboard_checkpoints/Mode
l3/assets
464/464 [==============================] - 19s 41ms/step - loss: 0.6254 - acc
uracy: 0.6738 - roc_auc_scr: 0.7164 - val_loss: 0.5976 - val_accuracy: 0.6919
- val_roc_auc_scr: 0.7504
Epoch 4/8
463/464 [============================>.] - ETA: 0s - loss: 0.6042 - accuracy:
```

```
0.6944 - roc_auc_scr: 0.7423
Epoch 00004: val_roc_auc_scr improved from 0.75041 to 0.76256, saving model t
o donors_choose/tensorboard_checkpoints/Model3
INFO:tensorflow:Assets written to: donors_choose/tensorboard_checkpoints/Mode
l3/assets
464/464 [==============================] - 19s 41ms/step - loss: 0.6043 - acc
uracy: 0.6943 - roc_auc_scr: 0.7422 - val_loss: 0.5897 - val_accuracy: 0.6990
- val_roc_auc_scr: 0.7626
Epoch 5/8
462/464 [============================>.] - ETA: 0s - loss: 0.5884 - accuracy:
0.7106 - roc_auc_scr: 0.7609
Epoch 00005: val_roc_auc_scr improved from 0.76256 to 0.77509, saving model t
o donors_choose/tensorboard_checkpoints/Model3
INFO:tensorflow:Assets written to: donors_choose/tensorboard_checkpoints/Mode
l3/assets
464/464 [==============================] - 19s 41ms/step - loss: 0.5884 - acc
uracy: 0.7105 - roc_auc_scr: 0.7608 - val_loss: 0.5710 - val_accuracy: 0.7151
- val_roc_auc_scr: 0.7751
Epoch 6/8
462/464 [============================>.] - ETA: 0s - loss: 0.5736 - accuracy:
0.7254 - roc_auc_scr: 0.7764
Epoch 00006: val_roc_auc_scr improved from 0.77509 to 0.78434, saving model t
o donors_choose/tensorboard_checkpoints/Model3
INFO:tensorflow:Assets written to: donors_choose/tensorboard_checkpoints/Mode
l3/assets
464/464 [==============================] - 19s 41ms/step - loss: 0.5736 - acc
uracy: 0.7253 - roc_auc_scr: 0.7763 - val_loss: 0.5660 - val_accuracy: 0.7228
- val_roc_auc_scr: 0.7843
Epoch 7/8
463/464 [============================>.] - ETA: 0s - loss: 0.5591 - accuracy:
0.7375 - roc_auc_scr: 0.7911
Epoch 00007: val_roc_auc_scr improved from 0.78434 to 0.79096, saving model t
o donors_choose/tensorboard_checkpoints/Model3
INFO:tensorflow:Assets written to: donors_choose/tensorboard_checkpoints/Mode
l3/assets
464/464 [==============================] - 19s 41ms/step - loss: 0.5591 - acc
uracy: 0.7375 - roc_auc_scr: 0.7911 - val_loss: 0.5549 - val_accuracy: 0.7289
- val_roc_auc_scr: 0.7910
Epoch 8/8
464/464 [==============================] - ETA: 0s - loss: 0.5422 - accuracy:
0.7521 - roc_auc_scr: 0.8067
Epoch 00008: val_roc_auc_scr improved from 0.79096 to 0.80345, saving model t
o donors_choose/tensorboard_checkpoints/Model3
INFO:tensorflow:Assets written to: donors_choose/tensorboard_checkpoints/Mode
l3/assets
464/464 [==============================] - 19s 41ms/step - loss: 0.5422 - acc
uracy: 0.7521 - roc_auc_scr: 0.8067 - val_loss: 0.5430 - val_accuracy: 0.7400
- val_roc_auc_scr: 0.8035
```

Model3: Test scores

In [ ]:
```
M3.evaluate({'inp1': test_essay_padded, 'inp2': test_ss,'inp3': test_tp, 'inp
4':test_pgc, 'inp5':test_cc, 'inp6':test_cs, 'inp7':test_num}, Y_test_cat, bat
ch_size=256)
```

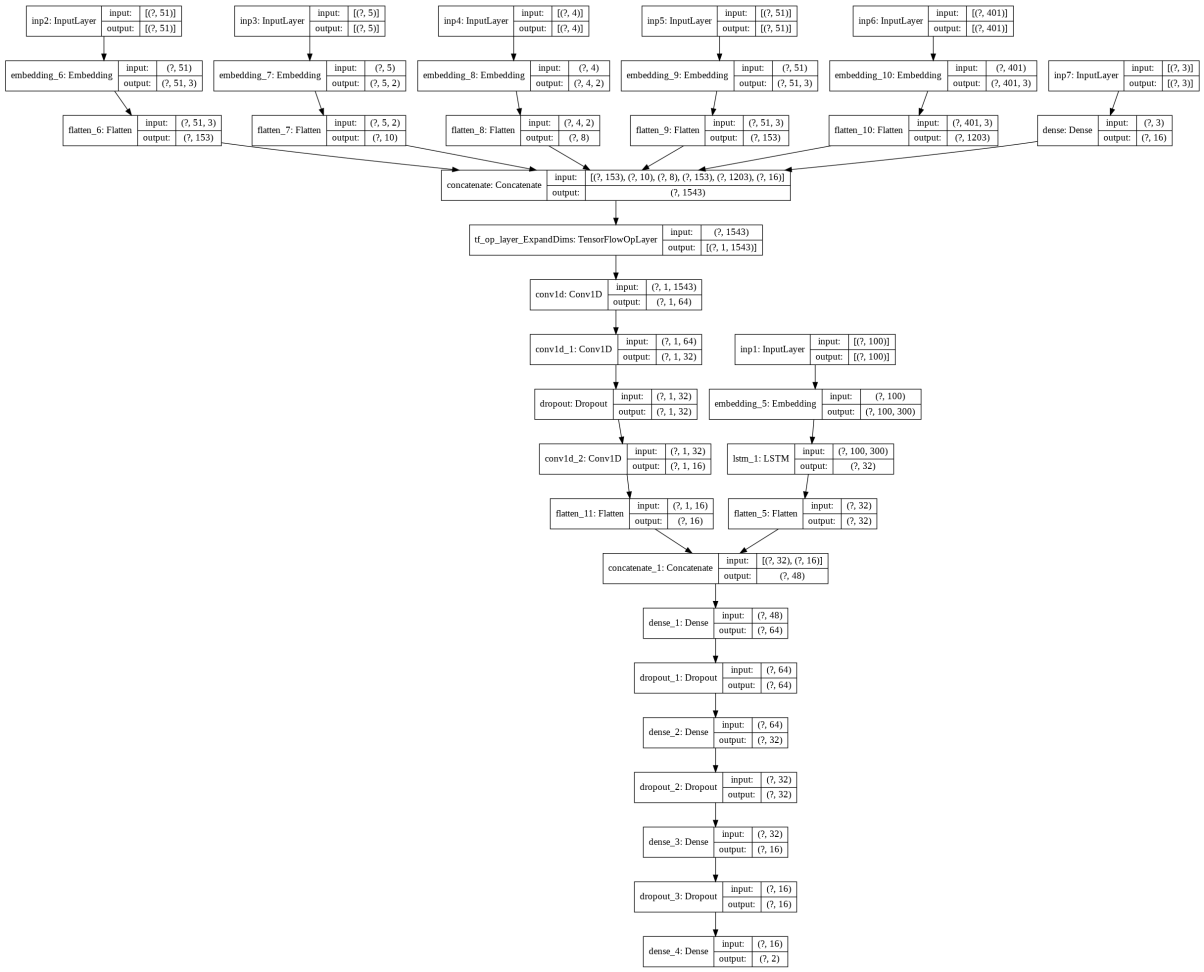
```
145/145 [==============================] - 3s 18ms/step - loss: 0.5425 - accu
racy: 0.7380 - roc_auc_scr: 0.8038
```

Out[ ]:
```
[0.5425145626068115, 0.7380470633506775, 0.8037858009338379]
```

Model3: show structure

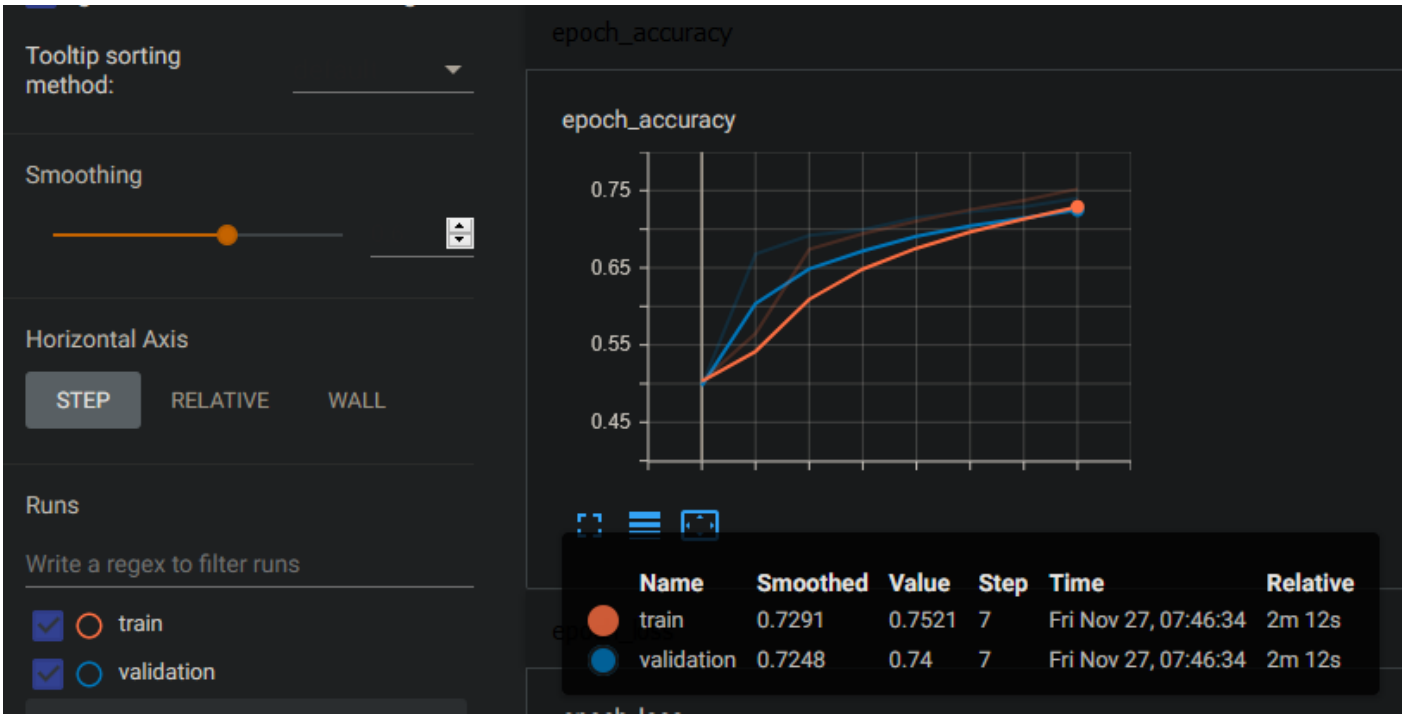
In [ ]:
```
plot_model(M3, show_shapes=True)
```

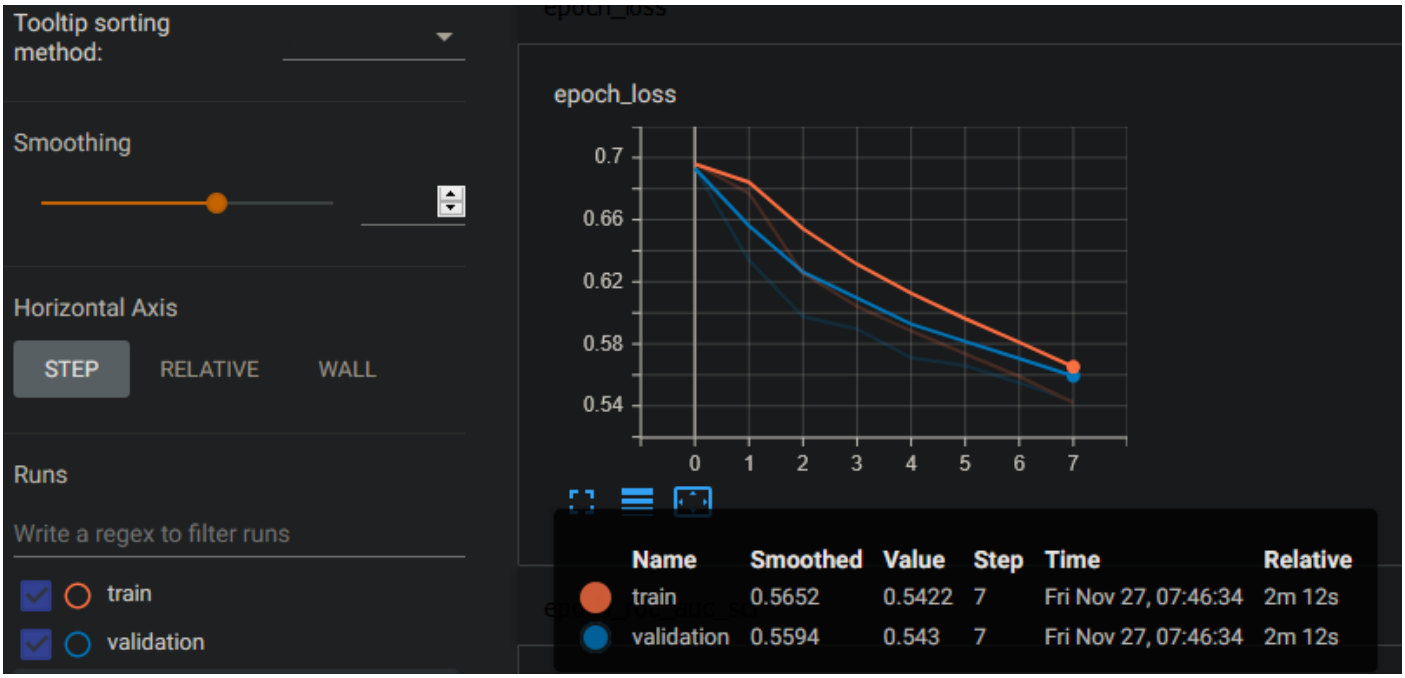
Out[ ]:

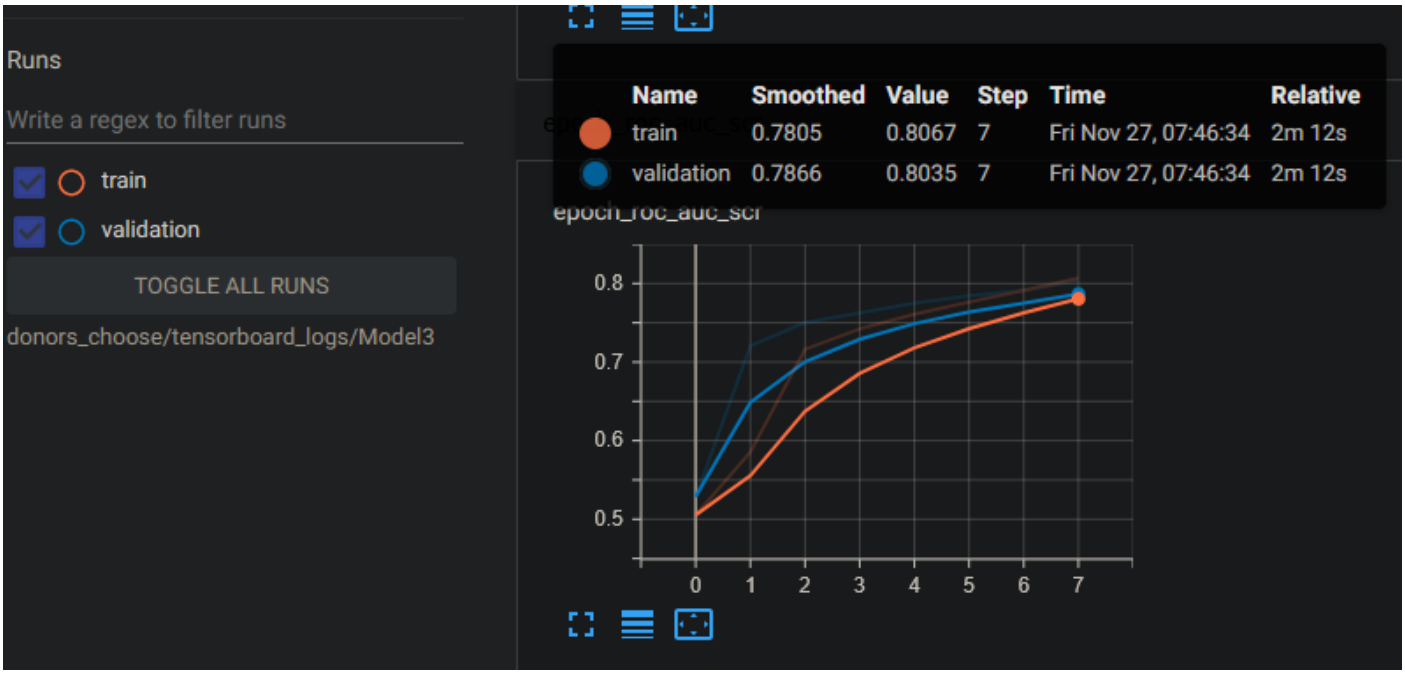


Model3: Train and Validation metrics plots Tensorboard

In [ ]:
```
%load_ext tensorboard
%tensorboard --logdir 'donors_choose/tensorboard_logs/Model3'
```


Tooltip sorting method:
Smoothing
Horizontal Axis
STEP
RELATIVE
WALL
Runs
Write a regex to filter runs
train
validation
epoch_accuracy
epoch_accuracy
0.75
0.65
0.55
0.45
Name Smoothed Value Step Time Relative
train 0.7291 0.7521 7 Fri Nov 27, 07:46:34 2m 12s
validation 0.7248 0.74 7 Fri Nov 27, 07:46:34 2m 12s


Tooltip sorting method:
Smoothing
Horizontal Axis
STEP
RELATIVE
WALL
Runs
Write a regex to filter runs
train
validation
epoch_loss
0.7
0.66
0.62
0.58
0.54
0 1 2 3 4 5 6 7
Name Smoothed Value Step Time Relative
train 0.5652 0.5422 7 Fri Nov 27, 07:46:34 2m 12s
validation 0.5594 0.543 7 Fri Nov 27, 07:46:34 2m 12s



Model3: Train and validation metrics plots

Conclusions

Here we have tried 3 types of models:

1. Model1: For Essay feature embedding features has been created using Glove model. All the words are considered while creating features. Since there are more features so model is converging very fast and hence within less number of epochs we are able to get the maximum roc_auc_score. Also here we have used label encoding for categorical features(Thanks for suggesting AAIC team)
2. Model2: It is similar to Model 2 except that instead of taking all the words as features we are choosing most important features. This we are doing through idf vectorizer. we are not choosing top 25% (These are words like The, a which are most frequent but actually don't contain any meaning. Also bottom idf features are rare features which also do not add any importance. With only 50% important and meaningful features we are able to get good score although in more number of epochs than Model 1 and Model 2. If we use more epochs probably we can get better results.
3. Model3: Here we are making use of convolution layer to learn any spatial feature in the categorical features. Both Model 1 and Model 3 are performing well, however Model 1 appears to converge faster. Instead of using label encoding here, We have used one hot encoding for categorical variables.

In [1]:
```
from prettytable import PrettyTable
t = PrettyTable()
t.field_names = ["Model","Train_roc_auc_score","Val_roc_auc_score", "Test_roc_
auc_score"]
t.add_row(["Model1",0.8005, 0.8034, 0.8064])
t.add_row(["Model2",0.7803 ,0.7607, 0.7646])
t.add_row(["Model3",0.8035, 0.8067, 0.8038])
print(t)
```

```
+--------+---------------------+-------------------+--------------------+
| Model  | Train_roc_auc_score | Val_roc_auc_score | Test_roc_auc_score |
+--------+---------------------+-------------------+--------------------+
| Model1 |        0.8005       |       0.8034      |       0.8064       |
| Model2 |        0.7803       |       0.7607      |       0.7646       |
| Model3 |        0.8035       |       0.8067      |       0.8038       |
+--------+---------------------+-------------------+--------------------+
```