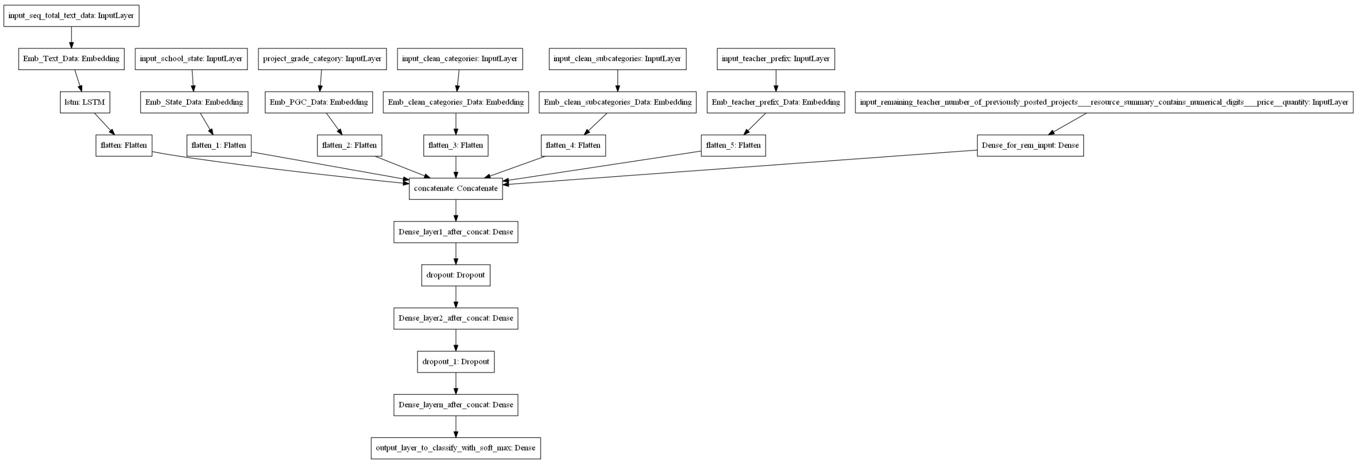
Assignment : 14

1. Download the preprocessed DonorsChoose data from here Dataset (https://drive.google.com/file/d/1GU3LIJJ3zS1xLXXe-sdItSJHtI5txjVO/view?usp=sharing)
2. Split the data into train, cv, and test
3. After step 2 you have to train 3 types of models as discussed below.
4. For all the model use 'auc' (https://scikit-learn.org/stable/modules/model_evaluation.html#roc-metrics) as a metric. check this (https://datascience.stackexchange.com/a/20192) for using auc as a metric. you need to print the AUC value for each epoch. Note: you should NOT use the tf.metric.auc
5. You are free to choose any number of layers/hiddden units but you have to use same type of architectures shown below.
6. You can use any one of the optimizers and choice of Learning rate and momentum, resources: cs231n class notes (http://cs231n.github.io/neural-networks-3/), cs231n class video (https://www.youtube.com/watch?v=hd_KFJ5ktUc).
7. You should Save the best model weights.
8. For all the model's use TensorBoard (https://www.youtube.com/watch?v=2U6Jl7oqRkM) and plot the Metric value and Loss with epoch. While submitting, take a screenshot of plots and include those images in .ipynb notebook and PDF.
9. Use Categorical Cross Entropy as Loss to minimize.
10. try to get AUC more than 0.8 for atleast one model

Model-1

Build and Train deep neural network as shown below


input_seq_total_text_data: InputLayer
Emb_Text_Data: Embedding
input_school_state: InputLayer
project_grade_category: InputLayer
input_clean_categories: InputLayer
input_clean_subcategories: InputLayer
input_teacher_prefix: InputLayer
lstm: LSTM
Emb_State_Data: Embedding
Emb_PGC_Data: Embedding
Emb_clean_categories_Data: Embedding
Emb_clean_subcategories_Data: Embedding
Emb_teacher_prefix_Data: Embedding
input_remaining_teacher_number_of_previously_posted_projects___resource_summary_contains_numerical_digits___price__quantity: InputLayer
flatten: Flatten
flatten_1: Flatten
flatten_2: Flatten
flatten_3: Flatten
flatten_4: Flatten
flatten_5: Flatten
Dense_for_rem_input: Dense
concatenate: Concatenate
Dense_layer1_after_concat: Dense
dropout: Dropout
Dense_layer2_after_concat: Dense
dropout_1: Dropout
Dense_layern_after_concat: Dense
output_layer_to_classify_with_soft_max: Dense

ref: https://i.imgur.com/w395Yk9.png (https://i.imgur.com/w395Yk9.png)

- **Input_seq_total_text_data** --- You have to give Total text data columns. After this use the Embedding layer to get word vectors. Use given predefined glove word vectors, don't train any word vectors. After this use LSTM and get the LSTM output and Flatten that output.
- **Input_school_state** --- Give 'school_state' column as input to embedding layer and Train the Keras Embedding layer.
- **Project_grade_category** --- Give 'project_grade_category' column as input to embedding layer and Train the Keras Embedding layer.
- **Input_clean_categories** --- Give 'input_clean_categories' column as input to embedding layer and Train the Keras Embedding layer.
- **Input_clean_subcategories** --- Give 'input_clean_subcategories' column as input to embedding layer and Train the Keras Embedding layer.
- **Input_clean_subcategories** --- Give 'input_teacher_prefix' column as input to embedding layer and Train the Keras Embedding layer.
- **Input_remaining_teacher_number_of_previously_posted_projects._resource_summary_contains_** ---concatenate remaining columns and add a Dense layer after that.

- For LSTM, you can choose your sequence padding methods on your own or you can train your LSTM without padding, there is no restriction on that.

Below is an example of embedding layer for a categorical columns. In below code all are dummy values, we gave only for referance.

In [0]:

```
# https://stats.stackexchange.com/questions/270546/how-does-keras-embedding-layer-work
input_layer = Input(shape=(n,))
embedding = Embedding(no_1, no_2, input_length=n)(input_layer)
flatten = Flatten()(embedding)
```

1. Go through this blog, if you have any doubt on using predefined Embedding values in Embedding layer - https://machinelearningmastery.com/use-word-embedding-layers-deep-learning-keras/ (https://machinelearningmastery.com/use-word-embedding-layers-deep-learning-keras/)

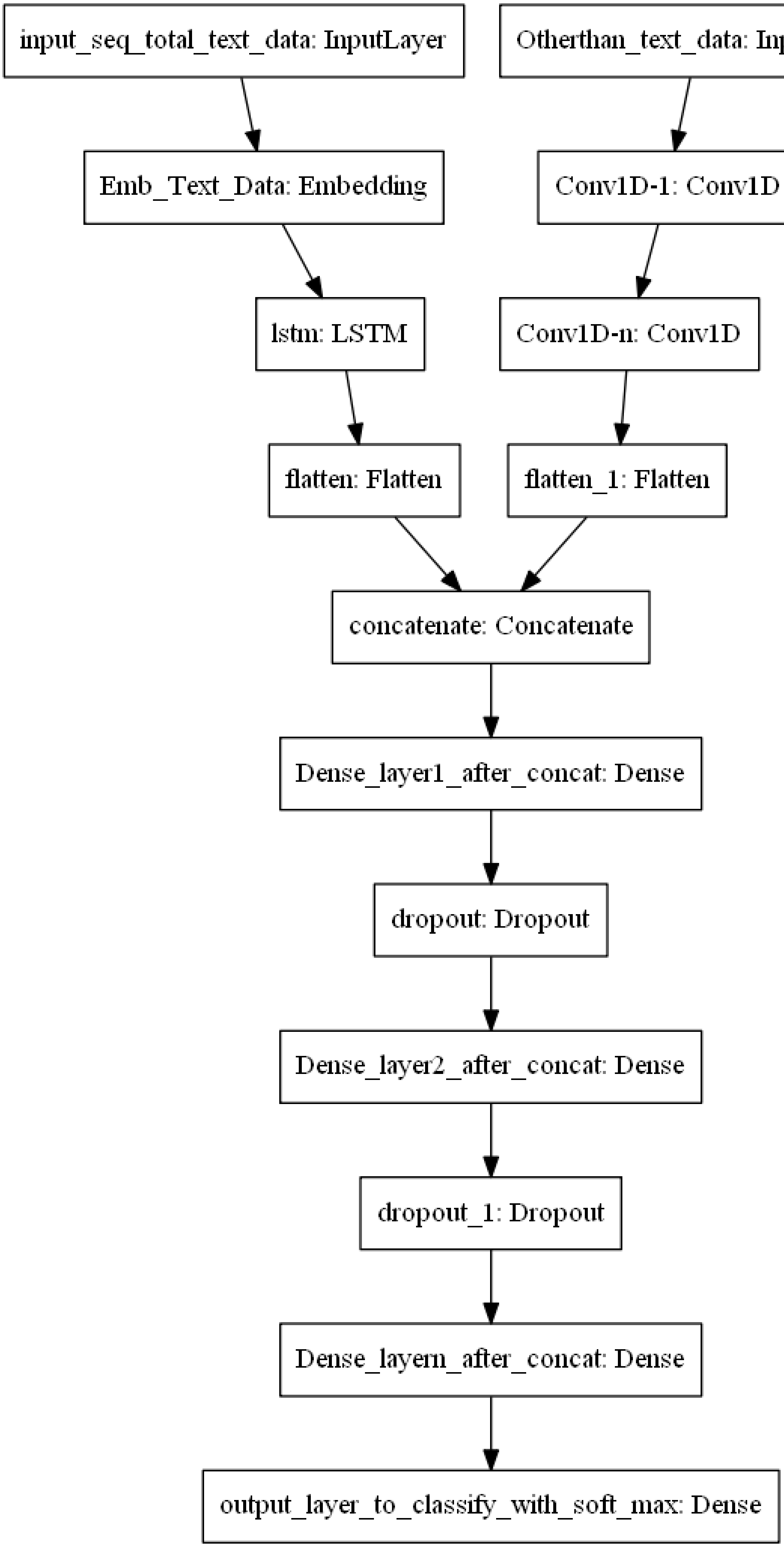
2. Please go through this link https://keras.io/getting-started/functional-api-guide/ (https://keras.io/getting-started/functional-api-guide/) and check the 'Multi-input and multi-output models' then you will get to know how to give multiple inputs.

Model-2

Use the same model as above but for 'input_seq_total_text_data' give only some words in the sentance not all the words. Filter the words as below.

```
1. Train the TF-IDF on the Train data feature 'essay'

2. Get the idf value for each word we have in the train data.

3. Remove the low idf value and high idf value words from our data. Do some anal
ysis on the Idf values and based on those values choose the low and high thresho
ld value. Because very frequent words and very very rare words don't give much i
nformation. (you can plot a box plots and take only the idf scores within IQR ra
nge and corresponding words)

4. Train the LSTM after removing the Low and High idf value words. (In model-1 T
rain on total data but in Model-2 train on data after removing some words based
 on IDF values)
```

Model-3


input_seq_total_text_data: InputLayer
Otherthan_text_data: InputLayer
Emb_Text_Data: Embedding
Conv1D-1: Conv1D
lstm: LSTM
Conv1D-n: Conv1D
flatten: Flatten
flatten_1: Flatten
concatenate: Concatenate
Dense_layer1_after_concat: Dense
dropout: Dropout
Dense_layer2_after_concat: Dense
dropout_1: Dropout
Dense_layern_after_concat: Dense
output_layer_to_classify_with_soft_max: Dense

ref: https://i.imgur.com/fkQ8nGo.png (https://i.imgur.com/fkQ8nGo.png)

- **input_seq_total_text_data**:

```
  . Use text column('essay'), and use the Embedding layer to get word vector
s.

  . Use given predefined glove word vectors, don't train any word vectors.

  . Use LSTM that is given above, get the LSTM output and Flatten that outpu
t.

  . You are free to preprocess the input text as you needed.
```

- **Other_than_text_data**:

```
  . Convert all your Categorical values to onehot coded and then concatenate
all these onehot vectors

  . Neumerical values and use CNN1D (https://keras.io/getting-started/sequen
tial-model-guide/#sequence-classification-with-1d-convolutions) as shown in
 above figure.

  . You are free to choose all CNN parameters like kernel sizes, stride.
```

</pre>

In [1]:

```
import numpy as np
import pandas as pd
from keras.preprocessing.sequence import pad_sequences
from keras.models import Sequential
from keras.layers import Dense, Input , Dropout
from keras.layers import Flatten
from keras.layers import concatenate
from keras.layers.embeddings import Embedding
from keras.models import Model
from keras.utils import to_categorical
from sklearn.model_selection import train_test_split
from keras.preprocessing.text import Tokenizer
import matplotlib.pyplot as plt
import pickle
from keras.layers import LSTM
from keras.preprocessing.text import text_to_word_sequence
import tensorflow as tf
from keras.callbacks import ModelCheckpoint,TensorBoard,ReduceLROnPlateau, EarlyStopping
from keras.layers.normalization import BatchNormalization
from sklearn.feature_extraction.text import TfidfVectorizer
import seaborn as sns
from keras.regularizers import l2
from sklearn.metrics import roc_auc_score
from keras.models import load_model
from IPython.display import Image
from scipy.sparse import hstack
from keras.layers import Conv1D
from sklearn.feature_extraction.text import CountVectorizer
from prettytable import PrettyTable
```

```
Using TensorFlow backend.
```

The default version of TensorFlow in Colab will soon switch to TensorFlow 2.x.
We recommend you upgrade (https://www.tensorflow.org/guide/migrate) now or ensure your notebook will continue to use TensorFlow 1.x via the `%tensorflow_version 1.x` magic: more info (https://colab.research.google.com/notebooks/tensorflow_version.ipynb).

In [2]:

```
import pandas as pd
data=pd.read_csv('preprocessed_data.csv')
data.shape
```

Out[2]:

```
(109248, 9)
```

In [3]:

```
data.columns
```

Out[3]:

```
Index(['school_state', 'teacher_prefix', 'project_grade_category',
       'teacher_number_of_previously_posted_projects', 'project_is_approve
d',
       'clean_categories', 'clean_subcategories', 'essay', 'price'],
      dtype='object')
```

In [4]:

```
y=data['project_is_approved'].values
y.shape
```

Out[4]:

```
(109248,)
```

In [0]:

```
data.drop(['project_is_approved'],axis=1,inplace=True)
```

In [6]:

```
data.shape
```

Out[6]:

```
(109248, 8)
```

SPLITTING THE DATA

In [7]:

```
from sklearn.model_selection import train_test_split
X_train,X_test,y_train,y_test=train_test_split(data,y,test_size=0.30,stratify=y)
X_train,X_cv,y_train,y_cv=train_test_split(X_train,y_train,test_size=0.30,stratify=y_tr
ain)
print(X_train.shape,y_train.shape)
print(X_cv.shape,y_cv.shape)
print(X_test.shape,y_test.shape)
```

```
(53531, 8) (53531,)
(22942, 8) (22942,)
(32775, 8) (32775,)
```

In [0]:

```
y_train = to_categorical(y_train, num_classes=2)
y_cv =to_categorical(y_cv,num_classes=2)
y_test = to_categorical(y_test, num_classes=2)
```

TOKENIZING ESSAY WORDS

In [9]:

```
token = Tokenizer()
token.fit_on_texts(X_train['essay'])
vocab_size = len(token.word_index) +1
print('Total unique words in the x_train',vocab_size)
encoded_train = token.texts_to_sequences(X_train['essay'])
encoded_cv = token.texts_to_sequences(X_cv['essay'])
encoded_test = token.texts_to_sequences(X_test['essay'])
```

```
Total unique words in the x_train 42722
```

In [10]:

```
len(encoded_train)
```

Out[10]:

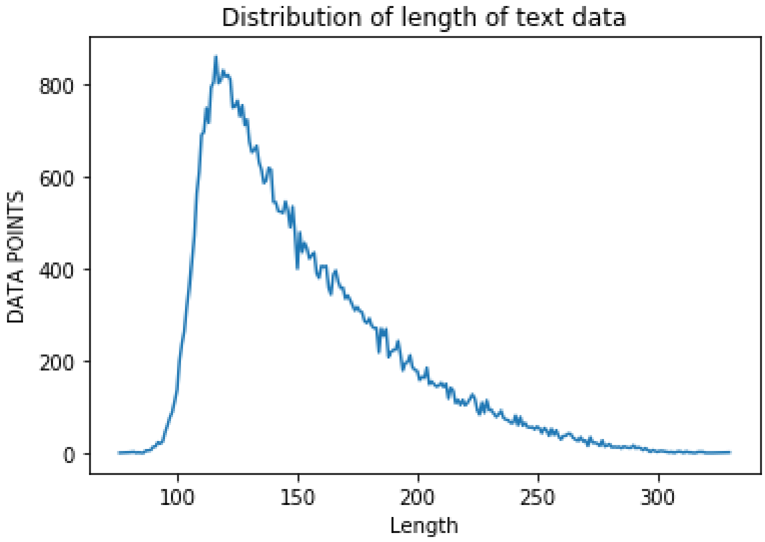
```
53531
```

In [0]:

```
length = []
for sentence in encoded_train:
    length.append(len(sentence))

s = list(set(length))
count = []
for i in s:
    count.append(length.count(i))
```

In [12]:

```
plt.plot(s,count)
plt.xlabel('Length')
plt.ylabel('DATA POINTS')
plt.title('Distribution of length of text data')
plt.show()
```



PADDING

In [13]:

```
max_length = 300
padded_train = pad_sequences(encoded_train, maxlen=max_length, padding='post')
padded_cv    =pad_sequences(encoded_cv,maxlen=max_length,padding='post')
padded_test = pad_sequences(encoded_test, maxlen=max_length, padding='post')
print("length of padded_train data",len(padded_train))
print("length of padded_cv data",len(padded_cv))
print("length of padded_test data",len(padded_test))
```

```
length of padded_train data 53531
length of padded_cv data 22942
length of padded_test data 32775
```

In [0]:

```

```

In [0]:

```
with open('glove_vectors', 'rb') as f:
    model = pickle.load(f)
    glove_words =  set(model.keys())
embedding_matrix_train = np.zeros((vocab_size, 300))
for word, i in token.word_index.items():
    if word in glove_words:
        embedding_vector = model[word]
        embedding_matrix_train[i] = embedding_vector
```

ENCODING CATEGORICAL VARIABLES

In [0]:

```
def label_encoder(column):
    unique = list(set(column))
    total = list(column)
    size = len(unique)
    count = []
    for category in unique:
        count.append([total.count(category),category])
    count.sort()
    rank = {}
    for i in range(1,len(count)+1):
        rank.update({count[i-1][1] : i})
    return (rank,unique,size)
```

In [17]:

```
category_rank,unique,size = label_encoder(X_train['clean_categories'])
print(category_rank)
categories_size = size
encoded_cat_train = []
encoded_cat_cv   =[]
encoded_cat_test = []
for category in X_train['clean_categories']:
    encoded_cat_train.append(category_rank[category])
for category in X_cv['clean_categories']:
    if category in unique:
        encoded_cat_cv.append(category_rank[category])
    else:
        encoded_cat_cv.append(0)

for category in X_test['clean_categories']:
    if category in unique:
        encoded_cat_test.append(category_rank[category])
    else:
        encoded_cat_test.append(0)

encoded_cat_train = np.asarray(encoded_cat_train)
encoded_cat_cv = np.asarray(encoded_cat_cv)
encoded_cat_test = np.asarray(encoded_cat_test)

print(encoded_cat_train.shape)
print(encoded_cat_cv.shape)
print(encoded_cat_test.shape)
```

```
{'literacy_language warmth care_hunger': 1, 'music_arts warmth care_hunge
r': 2, 'music_arts appliedlearning': 3, 'history_civics health_sports': 4,
'math_science warmth care_hunger': 5, 'appliedlearning warmth care_hunge
r': 6, 'music_arts health_sports': 7, 'music_arts history_civics': 8, 'spe
cialneeds warmth care_hunger': 9, 'health_sports warmth care_hunger': 10,
'health_sports history_civics': 11, 'specialneeds health_sports': 12, 'his
tory_civics appliedlearning': 13, 'literacy_language health_sports': 14,
'music_arts specialneeds': 15, 'health_sports music_arts': 16, 'appliedlea
rning history_civics': 17, 'health_sports appliedlearning': 18, 'history_c
ivics specialneeds': 19, 'health_sports math_science': 20, 'specialneeds m
usic_arts': 21, 'history_civics music_arts': 22, 'history_civics math_scie
nce': 23, 'math_science health_sports': 24, 'appliedlearning health_sport
s': 25, 'math_science history_civics': 26, 'literacy_language appliedlearn
ing': 27, 'appliedlearning music_arts': 28, 'health_sports literacy_langua
ge': 29, 'literacy_language history_civics': 30, 'appliedlearning math_sci
ence': 31, 'math_science appliedlearning': 32, 'warmth care_hunger': 33,
'health_sports specialneeds': 34, 'history_civics literacy_language': 35,
'appliedlearning specialneeds': 36, 'math_science music_arts': 37, 'litera
cy_language music_arts': 38, 'math_science specialneeds': 39, 'history_civ
ics': 40, 'appliedlearning literacy_language': 41, 'math_science literacy_
language': 42, 'appliedlearning': 43, 'literacy_language specialneeds': 4
4, 'specialneeds': 45, 'music_arts': 46, 'health_sports': 47, 'literacy_la
nguage math_science': 48, 'math_science': 49, 'literacy_language': 50}
(53531,)
(22942,)
(32775,)
```

In [18]:

```
subcategory_rank,unique,size = label_encoder(X_train['clean_subcategories'])
print(subcategory_rank)
subcategories_size = size
encoded_subcat_train = []
encoded_subcat_cv   =[]
encoded_subcat_test = []
for category in X_train['clean_subcategories']:
    encoded_subcat_train.append(subcategory_rank[category])
for category in X_cv['clean_subcategories']:
    if category in unique:
        encoded_subcat_cv.append(subcategory_rank[category])
    else:
        encoded_subcat_cv.append(0)

for category in X_test['clean_subcategories']:
    if category in unique:
        encoded_subcat_test.append(subcategory_rank[category])
    else:
        encoded_subcat_test.append(0)

encoded_subcat_train = np.asarray(encoded_subcat_train)
encoded_subcat_cv = np.asarray(encoded_subcat_cv)
encoded_subcat_test = np.asarray(encoded_subcat_test)

print(encoded_subcat_train.shape)
print(encoded_subcat_cv.shape)
print(encoded_subcat_test.shape)
```

```
{'appliedsciences economics': 1, 'appliedsciences foreignlanguages': 2, 'a
ppliedsciences nutritioneducation': 3, 'charactereducation nutritioneducat
ion': 4, 'civics_government foreignlanguages': 5, 'civics_government paren
tinvolvement': 6, 'civics_government teamsports': 7, 'college_careerprep g
ym_fitness': 8, 'college_careerprep teamsports': 9, 'college_careerprep wa
rmth care_hunger': 10, 'communityservice esl': 11, 'communityservice musi
c': 12, 'earlydevelopment economics': 13, 'economics foreignlanguages': 1
4, 'economics health_lifescience': 15, 'economics literature_writing': 16,
'economics music': 17, 'economics nutritioneducation': 18, 'esl economic
s': 19, 'esl financialliteracy': 20, 'extracurricular financialliteracy':
21, 'extracurricular foreignlanguages': 22, 'financialliteracy foreignlang
uages': 23, 'financialliteracy health_wellness': 24, 'financialliteracy pa
rentinvolvement': 25, 'financialliteracy performingarts': 26, 'financialli
teracy socialsciences': 27, 'gym_fitness parentinvolvement': 28, 'gym_fitn
ess socialsciences': 29, 'gym_fitness warmth care_hunger': 30, 'history_ge
ography teamsports': 31, 'literature_writing nutritioneducation': 32, 'nut
ritioneducation visualarts': 33, 'other socialsciences': 34, 'other warmth
care_hunger': 35, 'appliedsciences teamsports': 36, 'charactereducation wa
rmth care_hunger': 37, 'civics_government college_careerprep': 38, 'civics
_government health_wellness': 39, 'communityservice economics': 40, 'commu
nityservice nutritioneducation': 41, 'communityservice performingarts': 4
2, 'earlydevelopment financialliteracy': 43, 'earlydevelopment foreignlang
uages': 44, 'earlydevelopment teamsports': 45, 'economics specialneeds': 4
6, 'environmentalscience gym_fitness': 47, 'environmentalscience music': 4
8, 'environmentalscience warmth care_hunger': 49, 'esl extracurricular': 5
0, 'esl nutritioneducation': 51, 'extracurricular nutritioneducation': 52,
'extracurricular socialsciences': 53, 'foreignlanguages gym_fitness': 54,
'foreignlanguages health_lifescience': 55, 'foreignlanguages other': 56,
'gym_fitness history_geography': 57, 'health_lifescience warmth care_hunge
r': 58, 'health_wellness parentinvolvement': 59, 'history_geography parent
involvement': 60, 'literature_writing warmth care_hunger': 61, 'mathematic
s warmth care_hunger': 62, 'music parentinvolvement': 63, 'music socialsci
ences': 64, 'nutritioneducation warmth care_hunger': 65, 'visualarts warmt
h care_hunger': 66, 'charactereducation civics_government': 67, 'character
education financialliteracy': 68, 'civics_government esl': 69, 'civics_gov
ernment performingarts': 70, 'college_careerprep economics': 71, 'communit
yservice other': 72, 'earlydevelopment warmth care_hunger': 73, 'economics
socialsciences': 74, 'economics visualarts': 75, 'environmentalscience fin
ancialliteracy': 76, 'esl gym_fitness': 77, 'extracurricular gym_fitness':
78, 'extracurricular health_lifescience': 79, 'financialliteracy health_li
fescience': 80, 'financialliteracy other': 81, 'financialliteracy visualar
ts': 82, 'gym_fitness health_lifescience': 83, 'health_lifescience perform
ingarts': 84, 'literacy nutritioneducation': 85, 'mathematics teamsports':
86, 'music other': 87, 'nutritioneducation other': 88, 'other teamsports':
89, 'parentinvolvement performingarts': 90, 'performingarts teamsports': 9
1, 'teamsports visualarts': 92, 'charactereducation gym_fitness': 93, 'cha
ractereducation history_geography': 94, 'college_careerprep nutritioneduca
tion': 95, 'communityservice history_geography': 96, 'communityservice soc
ialsciences': 97, 'earlydevelopment socialsciences': 98, 'economics litera
cy': 99, 'extracurricular history_geography': 100, 'foreignlanguages socia
lsciences': 101, 'gym_fitness other': 102, 'gym_fitness performingarts': 1
03, 'health_lifescience music': 104, 'health_lifescience teamsports': 105,
'health_wellness socialsciences': 106, 'history_geography other': 107, 'li
terature_writing teamsports': 108, 'parentinvolvement socialsciences': 10
9, 'appliedsciences financialliteracy': 110, 'charactereducation foreignla
nguages': 111, 'charactereducation performingarts': 112, 'civics_governmen
t communityservice': 113, 'civics_government mathematics': 114, 'college_c
areerprep music': 115, 'communityservice parentinvolvement': 116, 'economi
cs environmentalscience': 117, 'environmentalscience extracurricular': 11
8, 'extracurricular health_wellness': 119, 'extracurricular parentinvolvem
ent': 120, 'foreignlanguages music': 121, 'gym_fitness visualarts': 122,
'music teamsports': 123, 'appliedsciences civics_government': 124, 'applie
dsciences gym_fitness': 125, 'civics_government health_lifescience': 126,
'communityservice earlydevelopment': 127, 'earlydevelopment extracurricula
r': 128, 'earlydevelopment nutritioneducation': 129, 'environmentalscience
foreignlanguages': 130, 'environmentalscience parentinvolvement': 131, 'ex
tracurricular specialneeds': 132, 'health_lifescience parentinvolvement':
133, 'charactereducation esl': 134, 'civics_government visualarts': 135,
'college_careerprep foreignlanguages': 136, 'communityservice extracurricu
lar': 137, 'esl performingarts': 138, 'financialliteracy history_geograph
y': 139, 'foreignlanguages visualarts': 140, 'history_geography performing
arts': 141, 'mathematics nutritioneducation': 142, 'nutritioneducation tea
msports': 143, 'other parentinvolvement': 144, 'performingarts socialscien
ces': 145, 'civics_government financialliteracy': 146, 'college_careerprep
esl': 147, 'college_careerprep financialliteracy': 148, 'college_careerpre
p health_wellness': 149, 'communityservice health_lifescience': 150, 'comm
unityservice health_wellness': 151, 'communityservice literacy': 152, 'com
munityservice mathematics': 153, 'environmentalscience other': 154, 'envir
onmentalscience performingarts': 155, 'esl music': 156, 'esl parentinvolve
ment': 157, 'financialliteracy literature_writing': 158, 'foreignlanguages
health_wellness': 159, 'health_lifescience other': 160, 'health_wellness h
istory_geography': 161, 'health_wellness warmth care_hunger': 162, 'litera
cy teamsports': 163, 'charactereducation music': 164, 'earlydevelopment pe
rformingarts': 165, 'esl socialsciences': 166, 'extracurricular music': 16
7, 'performingarts specialneeds': 168, 'charactereducation socialscience
s': 169, 'foreignlanguages history_geography': 170, 'gym_fitness literatur
e_writing': 171, 'specialneeds warmth care_hunger': 172, 'charactereducati
on health_lifescience': 173, 'civics_government specialneeds': 174, 'colle
ge_careerprep performingarts': 175, 'communityservice literature_writing':
176, 'communityservice specialneeds': 177, 'economics mathematics': 178,
'esl other': 179, 'extracurricular other': 180, 'extracurricular performin
garts': 181, 'financialliteracy literacy': 182, 'appliedsciences community
service': 183, 'civics_government environmentalscience': 184, 'college_car
eerprep environmentalscience': 185, 'college_careerprep history_geograph
y': 186, 'economics history_geography': 187, 'gym_fitness music': 188, 'he
alth_wellness performingarts': 189, 'parentinvolvement specialneeds': 190,
'charactereducation environmentalscience': 191, 'civics_government economi
cs': 192, 'college_careerprep socialsciences': 193, 'appliedsciences perfo
rmingarts': 194, 'charactereducation parentinvolvement': 195, 'charactered
ucation teamsports': 196, 'esl health_wellness': 197, 'extracurricular lit
erature_writing': 198, 'foreignlanguages specialneeds': 199, 'parentinvolv
ement visualarts': 200, 'foreignlanguages mathematics': 201, 'gym_fitness
literacy': 202, 'nutritioneducation specialneeds': 203, 'college_careerpre
p communityservice': 204, 'college_careerprep parentinvolvement': 205, 'co
mmunityservice visualarts': 206, 'earlydevelopment health_lifescience': 20
7, 'earlydevelopment music': 208, 'history_geography music': 209, 'special
needs teamsports': 210, 'college_careerprep extracurricular': 211, 'earlyd
evelopment parentinvolvement': 212, 'mathematics performingarts': 213, 'ap
pliedsciences health_wellness': 214, 'college_careerprep earlydevelopmen
t': 215, 'environmentalscience nutritioneducation': 216, 'extracurricular
teamsports': 217, 'college_careerprep health_lifescience': 218, 'esl visua
larts': 219, 'communityservice environmentalscience': 220, 'esl health_lif
escience': 221, 'esl history_geography': 222, 'gym_fitness mathematics': 2
23, 'esl environmentalscience': 224, 'parentinvolvement': 225, 'earlydevel
opment gym_fitness': 226, 'health_wellness visualarts': 227, 'characteredu
cation extracurricular': 228, 'economics': 229, 'literature_writing musi
c': 230, 'music visualarts': 231, 'health_wellness music': 232, 'extracurr
icular literacy': 233, 'health_lifescience nutritioneducation': 234, 'comm
unityservice': 235, 'financialliteracy specialneeds': 236, 'other visualar
ts': 237, 'socialsciences visualarts': 238, 'appliedsciences parentinvolve
ment': 239, 'environmentalscience health_wellness': 240, 'esl foreignlangu
ages': 241, 'gym_fitness nutritioneducation': 242, 'mathematics music': 24
```

```
3, 'extracurricular mathematics': 244, 'appliedsciences charactereducatio
n': 245, 'esl earlydevelopment': 246, 'health_lifescience history_geograph
y': 247, 'appliedsciences socialsciences': 248, 'socialsciences specialnee
ds': 249, 'health_lifescience socialsciences': 250, 'appliedsciences musi
c': 251, 'charactereducation visualarts': 252, 'earlydevelopment environme
ntalscience': 253, 'appliedsciences esl': 254, 'charactereducation communi
tyservice': 255, 'literature_writing parentinvolvement': 256, 'college_car
eerprep other': 257, 'environmentalscience socialsciences': 258, 'mathemat
ics parentinvolvement': 259, 'extracurricular visualarts': 260, 'foreignla
nguages literature_writing': 261, 'civics_government socialsciences': 262,
'civics_government literature_writing': 263, 'performingarts visualarts':
264, 'appliedsciences history_geography': 265, 'economics financialliterac
y': 266, 'charactereducation mathematics': 267, 'health_lifescience visual
arts': 268, 'charactereducation other': 269, 'mathematics socialsciences':
270, 'civics_government': 271, 'mathematics other': 272, 'charactereducati
on health_wellness': 273, 'literature_writing other': 274, 'history_geogra
phy mathematics': 275, 'literature_writing performingarts': 276, 'extracur
ricular': 277, 'appliedsciences other': 278, 'history_geography specialnee
ds': 279, 'literacy performingarts': 280, 'music specialneeds': 281, 'coll
ege_careerprep specialneeds': 282, 'charactereducation college_careerpre
p': 283, 'civics_government literacy': 284, 'environmentalscience history_
geography': 285, 'charactereducation earlydevelopment': 286, 'college_care
erprep visualarts': 287, 'earlydevelopment visualarts': 288, 'financiallit
eracy mathematics': 289, 'health_lifescience specialneeds': 290, 'health_l
ifescience health_wellness': 291, 'appliedsciences extracurricular': 292,
'earlydevelopment other': 293, 'literacy music': 294, 'literacy parentinvo
lvement': 295, 'charactereducation literature_writing': 296, 'environmenta
lscience specialneeds': 297, 'financialliteracy': 298, 'gym_fitness specia
lneeds': 299, 'health_wellness other': 300, 'history_geography visualart
s': 301, 'literacy other': 302, 'charactereducation specialneeds': 303, 'e
sl specialneeds': 304, 'appliedsciences earlydevelopment': 305, 'health_li
fescience literature_writing': 306, 'environmentalscience visualarts': 30
7, 'socialsciences': 308, 'earlydevelopment literature_writing': 309, 'civ
ics_government history_geography': 310, 'health_wellness mathematics': 31
1, 'foreignlanguages literacy': 312, 'college_careerprep literacy': 313,
'esl mathematics': 314, 'environmentalscience literature_writing': 315, 'h
ealth_wellness literature_writing': 316, 'nutritioneducation': 317, 'colle
ge_careerprep mathematics': 318, 'specialneeds visualarts': 319, 'earlydev
elopment health_wellness': 320, 'earlydevelopment mathematics': 321, 'hist
ory_geography socialsciences': 322, 'other specialneeds': 323, 'charactere
ducation': 324, 'college_careerprep literature_writing': 325, 'appliedscie
nces specialneeds': 326, 'charactereducation literacy': 327, 'health_lifes
cience literacy': 328, 'literature_writing socialsciences': 329, 'foreignl
anguages': 330, 'health_wellness teamsports': 331, 'performingarts': 332,
'appliedsciences college_careerprep': 333, 'esl': 334, 'literacy socialsci
ences': 335, 'appliedsciences literature_writing': 336, 'environmentalscie
nce literacy': 337, 'college_careerprep': 338, 'health_wellness literacy':
339, 'mathematics visualarts': 340, 'health_lifescience mathematics': 341,
'history_geography literacy': 342, 'history_geography': 343, 'appliedscien
ces health_lifescience': 344, 'appliedsciences literacy': 345, 'literacy v
isualarts': 346, 'gym_fitness teamsports': 347, 'appliedsciences visualart
s': 348, 'history_geography literature_writing': 349, 'literature_writing
visualarts': 350, 'earlydevelopment literacy': 351, 'esl literature_writin
g': 352, 'earlydevelopment specialneeds': 353, 'health_lifescience': 354,
'health_wellness nutritioneducation': 355, 'environmentalscience mathemati
cs': 356, 'other': 357, 'earlydevelopment': 358, 'music performingarts': 3
59, 'environmentalscience health_lifescience': 360, 'appliedsciences envir
onmentalscience': 361, 'environmentalscience': 362, 'teamsports': 363, 'he
alth_wellness specialneeds': 364, 'gym_fitness': 365, 'mathematics special
needs': 366, 'warmth care_hunger': 367, 'literature_writing specialneeds':
368, 'music': 369, 'visualarts': 370, 'gym_fitness health_wellness': 371,
'esl literacy': 372, 'literacy specialneeds': 373, 'appliedsciences': 374,
'appliedsciences mathematics': 375, 'health_wellness': 376, 'specialneed
s': 377, 'literature_writing': 378, 'mathematics': 379, 'literacy literatu
re_writing': 380, 'literature_writing mathematics': 381, 'literacy mathema
tics': 382, 'literacy': 383}
(53531,)
(22942,)
(32775,)
```

In [19]:

```
state_rank,unique,size =label_encoder(X_train['school_state'])
print(state_rank)
state_size = size
encoded_state_train = []
encoded_state_cv=[]
encoded_state_test = []
for state in X_train['school_state']:
    encoded_state_train.append(state_rank[state])
for state in X_cv['school_state']:
    if state in unique:
        encoded_state_cv.append(state_rank[state])
    else:
        encoded_state_cv.append(0)
for state in X_test['school_state']:
    if state in unique:
        encoded_state_test.append(state_rank[state])
    else:
        encoded_state_test.append(0)

encoded_state_train = np.asarray(encoded_state_train)
encoded_state_cv=np.asarray(encoded_state_cv)
encoded_state_test = np.asarray(encoded_state_test)

print(encoded_state_train.shape)
print(encoded_state_cv.shape)
print(encoded_state_test.shape)
```

```
{'vt': 1, 'wy': 2, 'nd': 3, 'ri': 4, 'mt': 5, 'sd': 6, 'ne': 7, 'nh': 8,
'ak': 9, 'de': 10, 'wv': 11, 'me': 12, 'hi': 13, 'dc': 14, 'nm': 15, 'ks':
16, 'id': 17, 'ia': 18, 'ar': 19, 'co': 20, 'mn': 21, 'or': 22, 'ms': 23,
'ky': 24, 'nv': 25, 'md': 26, 'ct': 27, 'ut': 28, 'tn': 29, 'al': 30, 'w
i': 31, 'va': 32, 'az': 33, 'ma': 34, 'ok': 35, 'nj': 36, 'la': 37, 'wa':
38, 'oh': 39, 'mo': 40, 'in': 41, 'pa': 42, 'mi': 43, 'sc': 44, 'ga': 45,
'il': 46, 'nc': 47, 'fl': 48, 'ny': 49, 'tx': 50, 'ca': 51}
(53531,)
(22942,)
(32775,)
```

In [20]:

```
teacher_prefix_rank, unique,size = label_encoder(X_train['teacher_prefix'])
print(teacher_prefix_rank)
teacher_prefix_size =size
encoded_prefix_train = []
encoded_prefix_cv=[]
encoded_prefix_test = []
for prefix in X_train['teacher_prefix']:
    encoded_prefix_train.append(teacher_prefix_rank[prefix])
for prefix in X_cv['teacher_prefix']:
    if prefix in unique:
        encoded_prefix_cv.append(teacher_prefix_rank[prefix])
    else:
        encoded_prefix_cv.append(0)

for prefix in X_test['teacher_prefix']:
    if prefix in unique:
        encoded_prefix_test.append(teacher_prefix_rank[prefix])
    else:
        encoded_prefix_test.append(0)

encoded_prefix_train = np.asarray(encoded_prefix_train)
encoded_prefix_cv=np.asarray(encoded_prefix_cv)
encoded_prefix_test = np.asarray(encoded_prefix_test)

print(encoded_prefix_train.shape)
print(encoded_prefix_cv.shape)
print(encoded_prefix_test.shape)
```

```
{'dr': 1, 'teacher': 2, 'mr': 3, 'ms': 4, 'mrs': 5}
(53531,)
(22942,)
(32775,)
```

In [21]:

```
project_grade_rank,unique,size =label_encoder(X_train['project_grade_category'])
print(project_grade_rank)
project_grade_categories_size = size
encoded_grade_train = []
encoded_grade_cv=[]
encoded_grade_test = []
for grade in X_train['project_grade_category']:
    encoded_grade_train.append(project_grade_rank[grade])
for grade in X_cv['project_grade_category']:
    if grade in unique:
        encoded_grade_cv.append(project_grade_rank[grade])
    else:
        encoded_grade_cv.append(0)

for grade in X_test['project_grade_category']:
    if grade in unique:
        encoded_grade_test.append(project_grade_rank[grade])
    else:
        encoded_grade_test.append(0)

encoded_grade_train = np.asarray(encoded_grade_train)
encoded_grade_cv=np.asarray(encoded_grade_cv)
encoded_grade_test = np.asarray(encoded_grade_test)

print(encoded_grade_train.shape)
print(encoded_grade_cv.shape)
print(encoded_grade_test.shape)
```

```
{'grades_9_12': 1, 'grades_6_8': 2, 'grades_3_5': 3, 'grades_prek_2': 4}
(53531,)
(22942,)
(32775,)
```

In [22]:

```
encoded_grade_train[0:5]
```

Out[22]:

```
array([2, 4, 3, 4, 2])
```

STANDARDIZING NUMERICAL FEATURES

In [23]:

```
# price

from sklearn.preprocessing import StandardScaler
scaler = StandardScaler()

scaler.fit(X_train['price'].values.reshape(-1,1))

X_train_price_norm = scaler.transform(X_train['price'].values.reshape(-1,1))
X_cv_price_norm = scaler.transform(X_cv['price'].values.reshape(-1,1))
X_test_price_norm = scaler.transform(X_test['price'].values.reshape(-1,1))

print("After vectorizations")
print(X_train_price_norm.shape, y_train.shape)
print(X_cv_price_norm.shape,y_cv.shape)
print(X_test_price_norm.shape, y_test.shape)
```

```
After vectorizations
(53531, 1) (53531, 2)
(22942, 1) (22942, 2)
(32775, 1) (32775, 2)
```

In [24]:

```
scaler = StandardScaler()

scaler.fit(X_train['teacher_number_of_previously_posted_projects'].values.reshape(-1,1
))

X_train_projects_norm = scaler.transform(X_train['teacher_number_of_previously_posted_p
rojects'].values.reshape(-1,1))
X_cv_projects_norm = scaler.transform(X_cv['teacher_number_of_previously_posted_project
s'].values.reshape(-1,1))
X_test_projects_norm = scaler.transform(X_test['teacher_number_of_previously_posted_pro
jects'].values.reshape(-1,1))

print("After vectorizations")
print(X_train_projects_norm.shape, y_train.shape)
print(X_cv_projects_norm.shape,y_cv.shape)
print(X_test_projects_norm.shape, y_test.shape)
```

```
After vectorizations
(53531, 1) (53531, 2)
(22942, 1) (22942, 2)
(32775, 1) (32775, 2)
```

In [0]:

```
left_input_train = np.hstack((X_train_price_norm,X_train_projects_norm))
left_input_cv = np.hstack((X_cv_price_norm,X_cv_projects_norm))
left_input_test = np.hstack((X_test_price_norm,X_test_projects_norm))
```

MODEL 1

In [0]:

```
essay_text=Input(shape=(300,),name='essay_text')
x=Embedding(vocab_size,300,weights=[embedding_matrix_train],input_length=300)(essay_text)
lstm_1=LSTM(50,recurrent_dropout=0.5,return_sequences=True)(x)
flatten_1=Flatten()(lstm_1)

state=Input(shape=(1,),name="state")
x=Embedding(state_size+1,2,input_length=1)(state)
flatten_2=Flatten()(x)

project_grade_category=Input(shape=(1,),name='project_grade_category')
x=Embedding(project_grade_categories_size+1,2,input_length=1)(project_grade_category)
flatten_3=Flatten()(x)

clean_categories=Input(shape=(1,),name='clean_categories')
x=Embedding(categories_size+1,4,input_length=1)(clean_categories)
flatten_4=Flatten()(x)

subcategory=Input(shape=(1,),name='subcategory')
x=Embedding(subcategories_size+1,4,input_length=1)(subcategory)
flatten_5=Flatten()(x)

teacher_prefix=Input(shape=(1,),name='teacher_prefix')
x=Embedding(teacher_prefix_size+1,4,input_length=1)(teacher_prefix)
flatten_6=Flatten()(x)

left_input=Input(shape=(2,),name='left_input')
dense_1 = Dense(1, activation='relu',kernel_initializer="he_normal",kernel_regularizer=l2(0.001))(left_input)

x=concatenate([flatten_1,flatten_2,flatten_3,flatten_4,flatten_5,flatten_6,dense_1])

x = Dense(64, activation='relu',kernel_initializer="he_normal",kernel_regularizer=l2(0.001))(x)
x = Dropout(.5)(x)
x = Dense(128, activation='relu',kernel_initializer="he_normal",kernel_regularizer=l2(0.001))(x)
x = Dropout(.5)(x)
x = BatchNormalization()(x)

x = Dense(64, activation='relu',kernel_initializer="he_normal",kernel_regularizer=l2(0.001))(x)
final_output = Dense(2, activation='softmax')(x)

model = Model(inputs=[essay_text,state,project_grade_category,clean_categories,subcategory,teacher_prefix,left_input], outputs=[final_output])
print(model.summary())
```

```
Model: "model_2"
________________________________________________________________________
__________________________
Layer (type)                    Output Shape         Param #     Connect
ed to
========================================================================
==========================
essay_text (InputLayer)         (None, 300)          0
________________________________________________________________________
__________________________
embedding_7 (Embedding)         (None, 300, 300)     12768900    essay_t
ext[0][0]
________________________________________________________________________
__________________________
state (InputLayer)              (None, 1)            0
________________________________________________________________________
__________________________
project_grade_category (InputLa (None, 1)            0
________________________________________________________________________
__________________________
clean_categories (InputLayer)   (None, 1)            0
________________________________________________________________________
__________________________
subcategory (InputLayer)        (None, 1)            0
________________________________________________________________________
__________________________
teacher_prefix (InputLayer)     (None, 1)            0
________________________________________________________________________
__________________________
lstm_2 (LSTM)                   (None, 300, 50)      70200       embeddi
ng_7[0][0]
________________________________________________________________________
__________________________
embedding_8 (Embedding)         (None, 1, 2)         104         state
[0][0]
________________________________________________________________________
__________________________
embedding_9 (Embedding)         (None, 1, 2)         10          project
_grade_category[0][0]
________________________________________________________________________
__________________________
embedding_10 (Embedding)        (None, 1, 4)         204         clean_c
ategories[0][0]
________________________________________________________________________
__________________________
embedding_11 (Embedding)        (None, 1, 4)         1540        subcate
gory[0][0]
________________________________________________________________________
__________________________
embedding_12 (Embedding)        (None, 1, 4)         24          teacher
_prefix[0][0]
________________________________________________________________________
__________________________
left_input (InputLayer)         (None, 2)            0
________________________________________________________________________
__________________________
flatten_7 (Flatten)             (None, 15000)        0           lstm_2
[0][0]
________________________________________________________________________
__________________________
flatten_8 (Flatten)             (None, 2)            0           embeddi
```

```
ng_8[0][0]
________________________________________________________________________
__________________________
flatten_9 (Flatten)             (None, 2)            0           embeddi
ng_9[0][0]
________________________________________________________________________
__________________________
flatten_10 (Flatten)            (None, 4)            0           embeddi
ng_10[0][0]
________________________________________________________________________
__________________________
flatten_11 (Flatten)            (None, 4)            0           embeddi
ng_11[0][0]
________________________________________________________________________
__________________________
flatten_12 (Flatten)            (None, 4)            0           embeddi
ng_12[0][0]
________________________________________________________________________
__________________________
dense_6 (Dense)                 (None, 1)            3           left_in
put[0][0]
________________________________________________________________________
__________________________
concatenate_2 (Concatenate)     (None, 15017)        0           flatten
_7[0][0]
                                                                 flatten
_8[0][0]
                                                                 flatten
_9[0][0]
                                                                 flatten
_10[0][0]
                                                                 flatten
_11[0][0]
                                                                 flatten
_12[0][0]
                                                                 dense_6
[0][0]
________________________________________________________________________
__________________________
dense_7 (Dense)                 (None, 64)           961152      concate
nate_2[0][0]
________________________________________________________________________
__________________________
dropout_3 (Dropout)             (None, 64)           0           dense_7
[0][0]
________________________________________________________________________
__________________________
dense_8 (Dense)                 (None, 128)          8320        dropout
_3[0][0]
________________________________________________________________________
__________________________
dropout_4 (Dropout)             (None, 128)          0           dense_8
[0][0]
________________________________________________________________________
__________________________
batch_normalization_2 (BatchNor (None, 128)          512         dropout
_4[0][0]
________________________________________________________________________
__________________________
dense_9 (Dense)                 (None, 64)           8256        batch_n
ormalization_2[0][0]
```

```
________________________________________________________________________
__________________________
dense_10 (Dense)                (None, 2)            130         dense_9
[0][0]
========================================================================
==========================
Total params: 13,819,355
Trainable params: 13,819,099
Non-trainable params: 256
________________________________________________________________________
__________________________
None
```

In [0]:

```python
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]:

```python
train_data = [padded_train,encoded_state_train,encoded_grade_train,encoded_cat_train,encoded_subcat_train,encoded_prefix_train,left_input_train]
cv_data = [padded_cv,encoded_state_cv,encoded_grade_cv,encoded_cat_cv,encoded_subcat_cv,encoded_prefix_cv,left_input_cv]
test_data = [padded_test,encoded_state_test,encoded_grade_test,encoded_cat_test,encoded_subcat_test,encoded_prefix_test,left_input_test]
```

In [0]:

```python
from keras.callbacks import TensorBoard
checkpoint_1 = ModelCheckpoint("model_1.1",
                             monitor="val_auroc",
                             mode="max",
                             save_best_only = True,
                             verbose=1)
earlystop = EarlyStopping(monitor = 'val_auroc',
                            mode="max",
                            min_delta = 0,
                            patience = 2,
                            verbose = 1,)
tensorboard=TensorBoard(log_dir='lstm_1',batch_size=512)
callbacks_1=[checkpoint_1,earlystop,tensorboard]
```

In [0]:

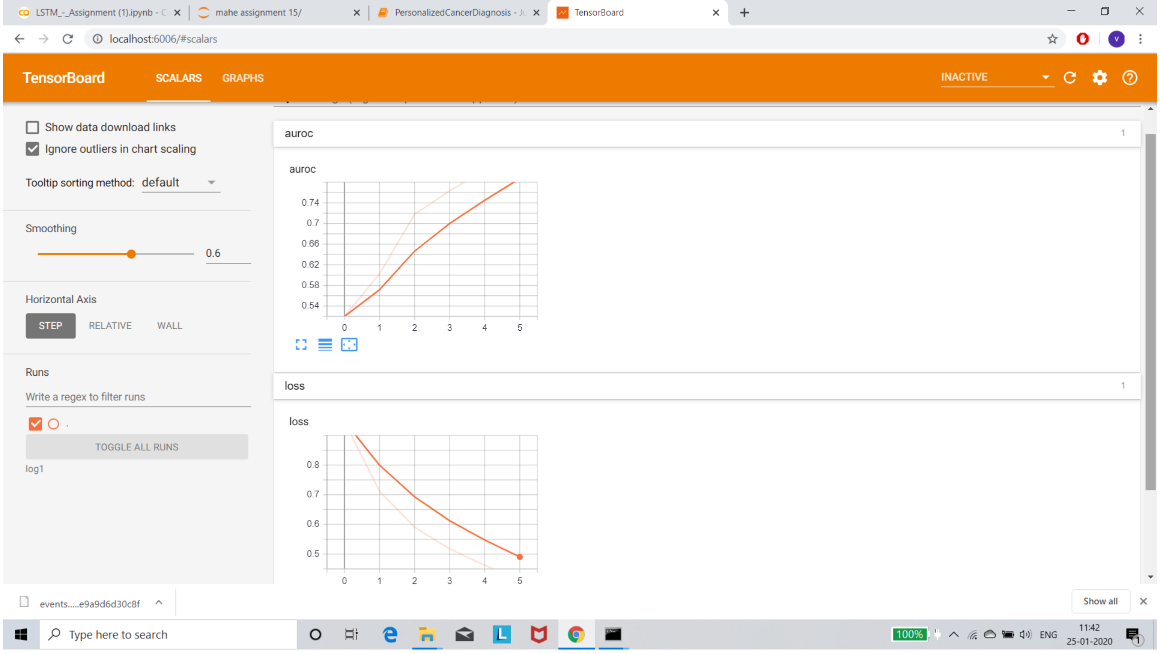
```
model.compile(optimizer='adam', loss='categorical_crossentropy', metrics=[auroc])
LSTM_1 = model.fit(train_data, y_train, batch_size=512, epochs=10, verbose=1,callbacks=
callbacks_1, validation_data=(cv_data, y_cv))
```

```
Train on 53531 samples, validate on 22942 samples
Epoch 1/10
53531/53531 [==============================] - 456s 9ms/step - loss: 0.947
4 - auroc: 0.5201 - val_loss: 0.7758 - val_auroc: 0.4897

Epoch 00001: val_auroc improved from -inf to 0.48968, saving model to mode
l_1.1
Epoch 2/10
53531/53531 [==============================] - 455s 9ms/step - loss: 0.711
2 - auroc: 0.6025 - val_loss: 0.7190 - val_auroc: 0.7081

Epoch 00002: val_auroc improved from 0.48968 to 0.70813, saving model to m
odel_1.1
Epoch 3/10
53531/53531 [==============================] - 450s 8ms/step - loss: 0.590
6 - auroc: 0.7181 - val_loss: 0.6186 - val_auroc: 0.7448

Epoch 00003: val_auroc improved from 0.70813 to 0.74482, saving model to m
odel_1.1
Epoch 4/10
53531/53531 [==============================] - 449s 8ms/step - loss: 0.517
5 - auroc: 0.7632 - val_loss: 0.5965 - val_auroc: 0.7520

Epoch 00004: val_auroc improved from 0.74482 to 0.75200, saving model to m
odel_1.1
Epoch 5/10
53531/53531 [==============================] - 448s 8ms/step - loss: 0.462
6 - auroc: 0.8038 - val_loss: 0.5980 - val_auroc: 0.7385

Epoch 00005: val_auroc did not improve from 0.75200
Epoch 6/10
53531/53531 [==============================] - 447s 8ms/step - loss: 0.411
1 - auroc: 0.8456 - val_loss: 0.5612 - val_auroc: 0.7269

Epoch 00006: val_auroc did not improve from 0.75200
Epoch 00006: early stopping
```

In [0]:

```
Image(retina=True, filename='/content/Screenshot (26).png')
```

Out[0]:



localhost:6006/#scalars

☐ Show data download links

☑ Ignore outliers in chart scaling

Tooltip sorting method: default

Smoothing 0.6

Horizontal Axis: STEP RELATIVE WALL

Runs

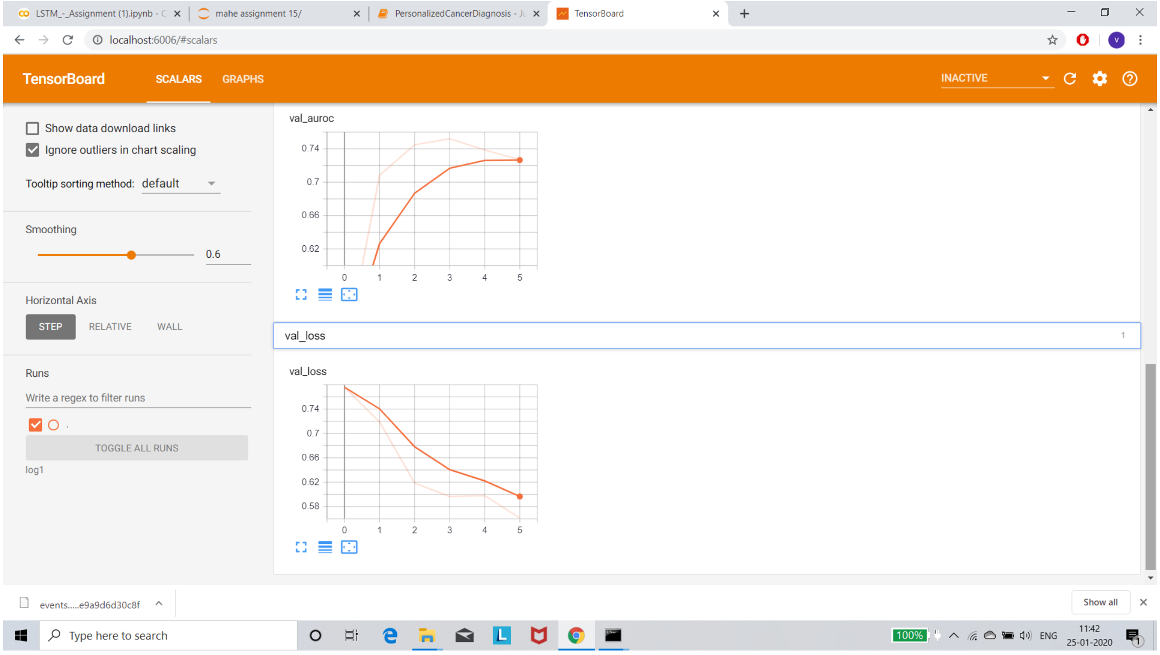
Write a regex to filter runs

log1

In [0]:

```
Image(retina=True, filename='/content/Screenshot (27).png')
```

Out[0]:



localhost:6006/#scalars

TensorBoard SCALARS GRAPHS INACTIVE

□ Show data download links
☑ Ignore outliers in chart scaling
Tooltip sorting method: default

Smoothing 0.6

Horizontal Axis
STEP RELATIVE WALL

Runs
Write a regex to filter runs
TOGGLE ALL RUNS
log1

val_loss

MODEL 2

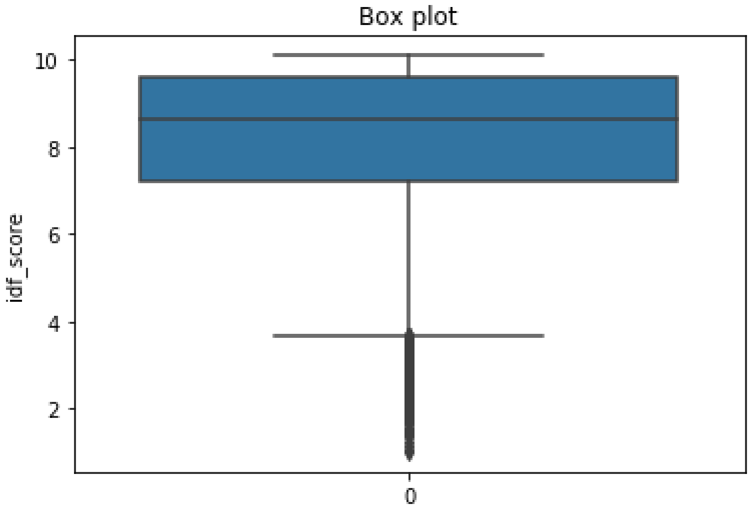
In [26]:

```
vectorizer = TfidfVectorizer(min_df=5)
X_train_essay_tfidf = vectorizer.fit_transform(X_train['essay'].values)
print("After vectorizations")
print(X_train_essay_tfidf.shape, y_train.shape)
```

```
After vectorizations
(53531, 16961) (53531, 2)
```

In [27]:

```
idf = vectorizer.idf_
sns.boxplot(data=idf)
plt.title('Box plot')
plt.ylabel('idf_score')
plt.show()
```



In [28]:

```
len(idf)
```

Out[28]:

```
16961
```

In [0]:

```
feature_names = np.asarray(vectorizer.get_feature_names())
index = []
for i in range(len(idf)):
    if idf[i] >= 2 and idf[i] <=10:
        index.append(i)
imp_words = []
for i in index:
    imp_words.append(feature_names[i])
```

In [30]:

```
print('total words=',len(feature_names))
print('important words=',len(imp_words))
```

```
total words= 16961
important words= 15665
```

In [31]:

```
# train_data
X_train_essay_imp = []
for essay in X_train['essay']:
    sentence = []
    for word in essay.split():
        if word in imp_words:
            sentence.append(word)
    X_train_essay_imp.append(' '.join(sentence))
print(len(X_train_essay_imp))
X_cv_essay_imp = []
for essay in X_cv['essay']:
    sentence = []
    for word in essay.split():
        if word in imp_words:
            sentence.append(word)
    X_cv_essay_imp.append(' '.join(sentence))
print(len(X_cv_essay_imp))
X_test_essay_imp = []
for essay in X_test['essay']:
    sentence = []
    for word in essay.split():
        if word in imp_words:
            sentence.append(word)
    X_test_essay_imp.append(' '.join(sentence))
print(len(X_test_essay_imp))
```

```
53531
22942
32775
```

In [32]:

```
 print(len(X_train_essay_imp))
print(len(X_cv_essay_imp))
print(len(X_test_essay_imp))
```

```
53531
22942
32775
```

In [33]:

```
token_2 = Tokenizer()
token_2.fit_on_texts(X_train_essay_imp)
imp_vocab = len(token_2.word_index) + 1
print('Total unique words in the important words',imp_vocab)
encoded_train_imp = token_2.texts_to_sequences(X_train_essay_imp)
encoded_cv_imp = token_2.texts_to_sequences(X_cv_essay_imp)
encoded_test_imp = token_2.texts_to_sequences(X_test_essay_imp)
print(len(encoded_train_imp))
print(len(encoded_cv_imp))
print(len(encoded_test_imp))
```

```
Total unique words in the important words 15666
53531
22942
32775
```

In [34]:

```
max_length = 300
padded_train_imp = pad_sequences(encoded_train_imp, maxlen=max_length, padding='post')
padded_cv_imp = pad_sequences(encoded_cv_imp, maxlen=max_length, padding='post')
padded_test_imp = pad_sequences(encoded_test_imp, maxlen=max_length, padding='post')
print("length of padded_train_new data",len(padded_train_imp))
print("length of padded_cv_new data",len(padded_cv_imp))
print("length of padded_test_new data",len(padded_test_imp))
```

```
length of padded_train_new data 53531
length of padded_cv_new data 22942
length of padded_test_new data 32775
```

In [0]:

```
with open('glove_vectors', 'rb') as f:
    model = pickle.load(f)
    glove_words =  set(model.keys())

# for train
embedding_matrix_train2 = np.zeros((imp_vocab, 300))
for word, i in token_2.word_index.items():
    if word in glove_words:
        embedding_vector = model[word]
        embedding_matrix_train2[i] = embedding_vector
```

In [37]:

```
essay_text=Input(shape=(300,),name='essay_text')
x=Embedding(imp_vocab,300,weights=[embedding_matrix_train2],input_length=300)(essay_tex
t)
lstm_1=LSTM(50,recurrent_dropout=0.5,return_sequences=True)(x)
flatten_1=Flatten()(lstm_1)

state=Input(shape=(1,),name="state")
x=Embedding(state_size+1,2,input_length=1)(state)
flatten_2=Flatten()(x)

project_grade_category=Input(shape=(1,),name='project_grade_category')
x=Embedding(project_grade_categories_size+1,2,input_length=1)(project_grade_category)
flatten_3=Flatten()(x)

clean_categories=Input(shape=(1,),name='clean_categories')
x=Embedding(categories_size+1,4,input_length=1)(clean_categories)
flatten_4=Flatten()(x)

subcategory=Input(shape=(1,),name='subcategory')
x=Embedding(subcategories_size+1,4,input_length=1)(subcategory)
flatten_5=Flatten()(x)

teacher_prefix=Input(shape=(1,),name='teacher_prefix')
x=Embedding(teacher_prefix_size+1,4,input_length=1)(teacher_prefix)
flatten_6=Flatten()(x)

left_input=Input(shape=(2,),name='left_input')
dense_1 = Dense(1, activation='relu',kernel_initializer="he_normal",kernel_regularizer=
l2(0.001))(left_input)

x=concatenate([flatten_1,flatten_2,flatten_3,flatten_4,flatten_5,flatten_6,dense_1])

x = Dense(32, activation='relu',kernel_initializer="he_normal",kernel_regularizer=l2(0.
001))(x)
x = Dropout(.5)(x)
x = Dense(64, activation='relu',kernel_initializer="he_normal",kernel_regularizer=l2(0.
001))(x)
x = Dropout(.5)(x)
x = BatchNormalization()(x)

x = Dense(64, activation='relu',kernel_initializer="he_normal",kernel_regularizer=l2(0.
001))(x)
final_output = Dense(2, activation='softmax')(x)

model = Model(inputs=[essay_text,state,project_grade_category,clean_categories,subcateg
ory,teacher_prefix,left_input], outputs=[final_output])
print(model.summary())
```

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

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

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

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

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

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

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

WARNING:tensorflow:From /usr/local/lib/python3.6/dist-packages/keras/bac
kend/tensorflow_backend.py:216: The name tf.is_variable_initialized is d
eprecated. Please use tf.compat.v1.is_variable_initialized instead.

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

WARNING:tensorflow:From /usr/local/lib/python3.6/dist-packages/keras/bac
kend/tensorflow_backend.py:148: The name tf.placeholder_with_default is
deprecated. Please use tf.compat.v1.placeholder_with_default instead.

WARNING:tensorflow:From /usr/local/lib/python3.6/dist-packages/keras/bac
kend/tensorflow_backend.py:3733: calling dropout (from tensorflow.pytho
n.ops.nn_ops) with keep_prob is deprecated and will be removed in a futu
re 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/bac
kend/tensorflow_backend.py:4479: The name tf.truncated_normal is depreca
ted. Please use tf.random.truncated_normal instead.

Model: "model_1"
________________________________________________________________________
__________________________
Layer (type)                    Output Shape         Param #     Connect
ed to
========================================================================
==========================
essay_text (InputLayer)         (None, 300)          0
________________________________________________________________________
__________________________
```

```
embedding_1 (Embedding)         (None, 300, 300)     4699800     essay_t
ext[0][0]

________________________________________________________________________
__________________________
state (InputLayer)              (None, 1)            0

________________________________________________________________________
__________________________
project_grade_category (InputLa (None, 1)            0

________________________________________________________________________
__________________________
clean_categories (InputLayer)   (None, 1)            0

________________________________________________________________________
__________________________
subcategory (InputLayer)        (None, 1)            0

________________________________________________________________________
__________________________
teacher_prefix (InputLayer)     (None, 1)            0

________________________________________________________________________
__________________________
lstm_1 (LSTM)                   (None, 300, 50)      70200       embeddi
ng_1[0][0]

________________________________________________________________________
__________________________
embedding_2 (Embedding)         (None, 1, 2)         104         state
[0][0]

________________________________________________________________________
__________________________
embedding_3 (Embedding)         (None, 1, 2)         10          project
_grade_category[0][0]

________________________________________________________________________
__________________________
embedding_4 (Embedding)         (None, 1, 4)         204         clean_c
ategories[0][0]

________________________________________________________________________
__________________________
embedding_5 (Embedding)         (None, 1, 4)         1536        subcate
gory[0][0]

________________________________________________________________________
__________________________
embedding_6 (Embedding)         (None, 1, 4)         24          teacher
_prefix[0][0]

________________________________________________________________________
__________________________
left_input (InputLayer)         (None, 2)            0

________________________________________________________________________
__________________________
flatten_1 (Flatten)             (None, 15000)        0           lstm_1
[0][0]

________________________________________________________________________
__________________________
flatten_2 (Flatten)             (None, 2)            0           embeddi
ng_2[0][0]

________________________________________________________________________
__________________________
flatten_3 (Flatten)             (None, 2)            0           embeddi
ng_3[0][0]

________________________________________________________________________
__________________________
flatten_4 (Flatten)             (None, 4)            0           embeddi
ng_4[0][0]

________________________________________________________________________
```

```
__________________________
flatten_5 (Flatten)             (None, 4)            0           embeddi
ng_5[0][0]

________________________________________________________________________
__________________________
flatten_6 (Flatten)             (None, 4)            0           embeddi
ng_6[0][0]

________________________________________________________________________
__________________________
dense_1 (Dense)                 (None, 1)            3           left_in
put[0][0]

________________________________________________________________________
__________________________
concatenate_1 (Concatenate)     (None, 15017)        0           flatten
_1[0][0]
                                                                 flatten
_2[0][0]
                                                                 flatten
_3[0][0]
                                                                 flatten
_4[0][0]
                                                                 flatten
_5[0][0]
                                                                 flatten
_6[0][0]
                                                                 dense_1
[0][0]

________________________________________________________________________
__________________________
dense_2 (Dense)                 (None, 32)           480576      concate
nate_1[0][0]

________________________________________________________________________
__________________________
dropout_1 (Dropout)             (None, 32)           0           dense_2
[0][0]

________________________________________________________________________
__________________________
dense_3 (Dense)                 (None, 64)           2112        dropout
_1[0][0]

________________________________________________________________________
__________________________
dropout_2 (Dropout)             (None, 64)           0           dense_3
[0][0]

________________________________________________________________________
__________________________
batch_normalization_1 (BatchNor (None, 64)           256         dropout
_2[0][0]

________________________________________________________________________
__________________________
dense_4 (Dense)                 (None, 64)           4160        batch_n
ormalization_1[0][0]

________________________________________________________________________
__________________________
dense_5 (Dense)                 (None, 2)            130         dense_4
[0][0]
========================================================================
==========================
Total params: 5,259,115
Trainable params: 5,258,987
Non-trainable params: 128

________________________________________________________________________
```

```
None
```

In [0]:

```
from keras.callbacks import TensorBoard
checkpoint_2 = ModelCheckpoint("model_2.1",
                             monitor="val_auroc",
                             mode="max",
                             save_best_only = True,
                             verbose=1)
earlystop_2 = EarlyStopping(monitor = 'val_auroc',
                            mode="max",
                            min_delta = 0,
                            patience = 2,
                            verbose = 1,)
tensorboard_1=TensorBoard(log_dir='lstm_2',batch_size=512)
callbacks_2=[checkpoint_2,earlystop_2,tensorboard_1]
```

In [0]:

```
train_data_1 = [padded_train_imp,encoded_state_train,encoded_grade_train,encoded_cat_train,encoded_subcat_train,encoded_prefix_train,left_input_train]
cv_data_1 = [padded_cv_imp,encoded_state_cv,encoded_grade_cv,encoded_cat_cv,encoded_subcat_cv,encoded_prefix_cv,left_input_cv]
test_data_1 = [padded_test_imp,encoded_state_test,encoded_grade_test,encoded_cat_test,encoded_subcat_test,encoded_prefix_test,left_input_test]
```

In [42]:

```
model.compile(optimizer='adam', loss='categorical_crossentropy', metrics=[auroc])
LSTM_2 = model.fit(train_data_1, y_train, batch_size=512, epochs=10, verbose=1,callback
s=callbacks_2, validation_data=(cv_data_1, y_cv))
```

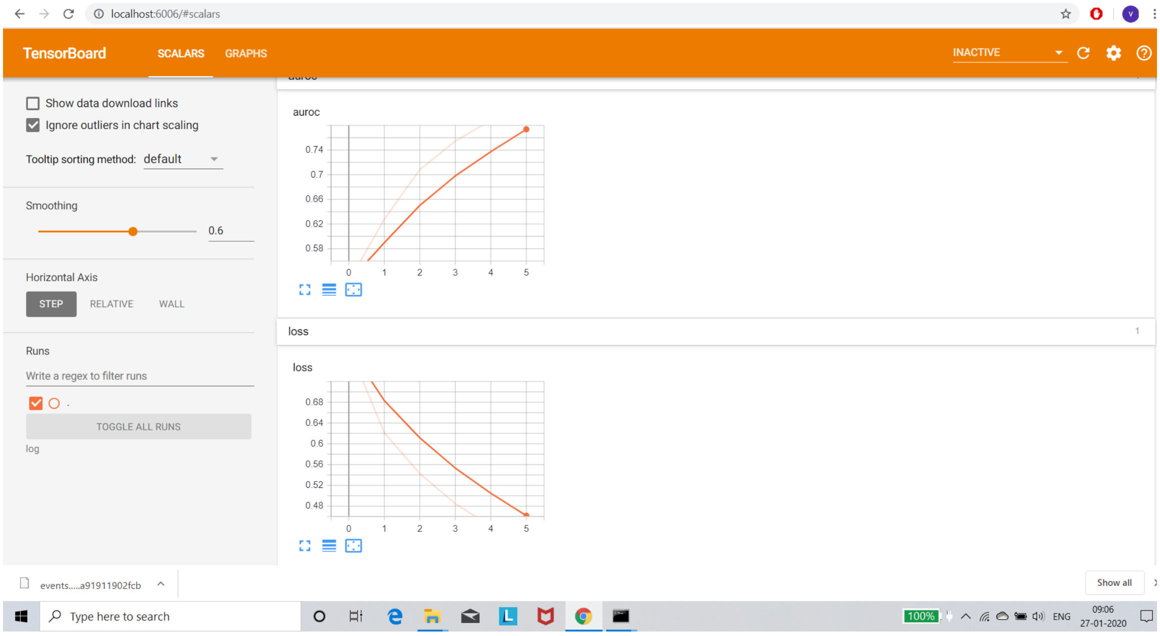
```
WARNING:tensorflow:From /usr/local/lib/python3.6/dist-packages/keras/optim
izers.py:793: The name tf.train.Optimizer is deprecated. Please use tf.com
pat.v1.train.Optimizer instead.

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

WARNING:tensorflow:From <ipython-input-41-4a25250c5bd7>:5: py_func (from t
ensorflow.python.ops.script_ops) is deprecated and will be removed in a fu
ture 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 tenso
r to
    an ndarray (just call tensor.numpy()) but having access to eager tenso
rs
    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_fu
nc
    (it is not differentiable, and manipulates numpy arrays). It drops the
    stateful argument making all functions stateful.

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 version.
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/backe
nd/tensorflow_backend.py:1033: The name tf.assign_add is deprecated. Pleas
e use tf.compat.v1.assign_add instead.

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

Train on 53531 samples, validate on 22942 samples
WARNING:tensorflow:From /usr/local/lib/python3.6/dist-packages/keras/callb
acks.py:1122: The name tf.summary.merge_all is deprecated. Please use tf.c
ompat.v1.summary.merge_all instead.

WARNING:tensorflow:From /usr/local/lib/python3.6/dist-packages/keras/callb
acks.py:1125: The name tf.summary.FileWriter is deprecated. Please use tf.
compat.v1.summary.FileWriter instead.

Epoch 1/10
53531/53531 [==============================] - 386s 7ms/step - loss: 0.786
5 - auroc: 0.5264 - val_loss: 0.6874 - val_auroc: 0.5568

Epoch 00001: val_auroc improved from -inf to 0.55678, saving model to mode
l_2.1
WARNING:tensorflow:From /usr/local/lib/python3.6/dist-packages/keras/callb
acks.py:1265: The name tf.Summary is deprecated. Please use tf.compat.v1.S
ummary instead.

Epoch 2/10
53531/53531 [==============================] - 385s 7ms/step - loss: 0.620
9 - auroc: 0.6278 - val_loss: 0.6117 - val_auroc: 0.7106
```

```
Epoch 00002: val_auroc improved from 0.55678 to 0.71064, saving model to m
odel_2.1
Epoch 3/10
53531/53531 [==============================] - 390s 7ms/step - loss: 0.542
5 - auroc: 0.7083 - val_loss: 0.6237 - val_auroc: 0.7147

Epoch 00003: val_auroc improved from 0.71064 to 0.71470, saving model to m
odel_2.1
Epoch 4/10
53531/53531 [==============================] - 391s 7ms/step - loss: 0.484
5 - auroc: 0.7545 - val_loss: 0.5351 - val_auroc: 0.7470

Epoch 00004: val_auroc improved from 0.71470 to 0.74695, saving model to m
odel_2.1
Epoch 5/10
53531/53531 [==============================] - 390s 7ms/step - loss: 0.441
9 - auroc: 0.7884 - val_loss: 0.5426 - val_auroc: 0.7382

Epoch 00005: val_auroc did not improve from 0.74695
Epoch 6/10
53531/53531 [==============================] - 389s 7ms/step - loss: 0.401
7 - auroc: 0.8239 - val_loss: 0.5385 - val_auroc: 0.7280

Epoch 00006: val_auroc did not improve from 0.74695
Epoch 00006: early stopping
```

In [44]:

```
Image(retina=True, filename='/content/Screenshot (32).png')
```

Out[44]:



localhost:6006/#scalars

TensorBoard SCALARS GRAPHS INACTIVE

□ Show data download links

☑ Ignore outliers in chart scaling

Tooltip sorting method: default

Smoothing 0.6

Horizontal Axis: STEP RELATIVE WALL

Runs

Write a regex to filter runs

☑ ○ .

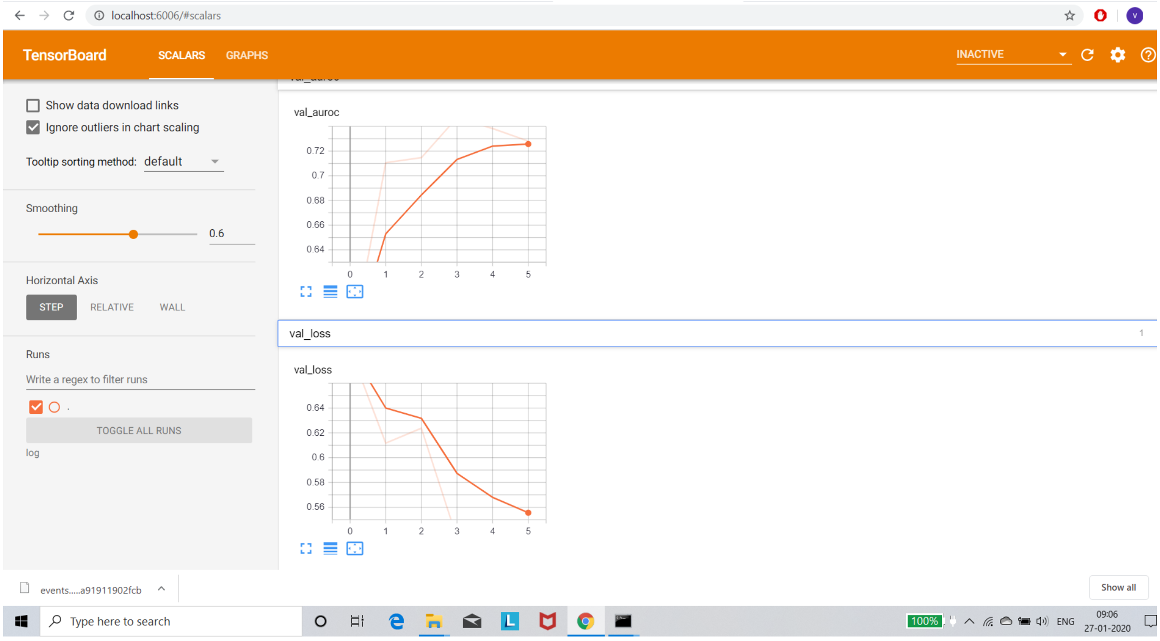
TOGGLE ALL RUNS

log

In [45]:

```
Image(retina=True, filename='/content/Screenshot (33).png')
```

Out[45]:



□ Show data download links

☑ Ignore outliers in chart scaling

Tooltip sorting method: default

Smoothing 0.6

Horizontal Axis: STEP RELATIVE WALL

Runs

Write a regex to filter runs

☑ ○ .

TOGGLE ALL RUNS

log

In [0]:

```

```

MODEL 3

ONE HOT ENCODING OF CATEGORICAL VARIABLES

In [0]:

```
vectorizer = CountVectorizer()
vectorizer.fit(X_train['clean_categories'].values)

X_train_categories_one_hot = vectorizer.transform(X_train['clean_categories'].values)
X_cv_categories_one_hot = vectorizer.transform(X_cv['clean_categories'].values)
X_test_categories_one_hot = vectorizer.transform(X_test['clean_categories'].values)
print(vectorizer.get_feature_names())
print("-"*100)
print("After vectorizations")
print(X_train_categories_one_hot.shape, y_train.shape)
print(X_cv_categories_one_hot.shape, y_cv.shape)
print(X_test_categories_one_hot.shape, y_test.shape)
```

```
['appliedlearning', 'care_hunger', 'health_sports', 'history_civics', 'lit
eracy_language', 'math_science', 'music_arts', 'specialneeds', 'warmth']
--------------------------------------------------------------------------
--------------------------
After vectorizations
(53531, 9) (53531, 2)
(22942, 9) (22942, 2)
(32775, 9) (32775, 2)
```

In [0]:

```
vectorizer = CountVectorizer()
vectorizer.fit(X_train['clean_subcategories'].values)

X_train_subcategories_one_hot = vectorizer.transform(X_train['clean_subcategories'].val
ues)
X_cv_subcategories_one_hot = vectorizer.transform(X_cv['clean_subcategories'].values)
X_test_subcategories_one_hot = vectorizer.transform(X_test['clean_subcategories'].value
s)
print(vectorizer.get_feature_names())
print("-"*100)
print("After vectorizations")
print(X_train_subcategories_one_hot.shape, y_train.shape)
print(X_cv_subcategories_one_hot.shape, y_cv.shape)
print(X_test_subcategories_one_hot.shape, y_test.shape)
```

```
['appliedsciences', 'care_hunger', 'charactereducation', 'civics_governmen
t', 'college_careerprep', 'communityservice', 'earlydevelopment', 'economi
cs', 'environmentalscience', 'esl', 'extracurricular', 'financialliterac
y', 'foreignlanguages', 'gym_fitness', 'health_lifescience', 'health_welln
ess', 'history_geography', 'literacy', 'literature_writing', 'mathematic
s', 'music', 'nutritioneducation', 'other', 'parentinvolvement', 'performi
ngarts', 'socialsciences', 'specialneeds', 'teamsports', 'visualarts', 'wa
rmth']
--------------------------------------------------------------------------
--------------------------
After vectorizations
(53531, 30) (53531, 2)
(22942, 30) (22942, 2)
(32775, 30) (32775, 2)
```

In [0]:

```
X_train.columns
```

Out[0]:

```
Index(['school_state', 'teacher_prefix', 'project_grade_category',
       'teacher_number_of_previously_posted_projects', 'clean_categories',
       'clean_subcategories', 'essay', 'price'],
      dtype='object')
```

In [0]:

```
vectorizer = CountVectorizer()
vectorizer.fit(X_train['teacher_prefix'].values)

X_train_prefix_one_hot = vectorizer.transform(X_train['teacher_prefix'].values)
X_cv_prefix_one_hot = vectorizer.transform(X_cv['teacher_prefix'].values)
X_test_prefix_one_hot = vectorizer.transform(X_test['teacher_prefix'].values)
print(vectorizer.get_feature_names())
print("-"*100)
print("After vectorizations")
print(X_train_prefix_one_hot.shape, y_train.shape)
print(X_cv_prefix_one_hot.shape, y_cv.shape)
print(X_test_prefix_one_hot.shape, y_test.shape)
```

```
['dr', 'mr', 'mrs', 'ms', 'teacher']
--------------------------------------------------------------------------
--------------------------
After vectorizations
(53531, 5) (53531, 2)
(22942, 5) (22942, 2)
(32775, 5) (32775, 2)
```

In [0]:

```
vectorizer = CountVectorizer()
vectorizer.fit(X_train['school_state'].values)

X_train_state_one_hot = vectorizer.transform(X_train['school_state'].values)
X_cv_state_one_hot = vectorizer.transform(X_cv['school_state'].values)
X_test_state_one_hot = vectorizer.transform(X_test['school_state'].values)
print(vectorizer.get_feature_names())
print("-"*100)
print("After vectorizations")
print(X_train_state_one_hot.shape, y_train.shape)
print(X_cv_state_one_hot.shape, y_cv.shape)
print(X_test_state_one_hot.shape, y_test.shape)
```

```
['ak', 'al', 'ar', 'az', 'ca', 'co', 'ct', 'dc', 'de', 'fl', 'ga', 'hi',
'ia', 'id', 'il', 'in', 'ks', 'ky', 'la', 'ma', 'md', 'me', 'mi', 'mn', 'm
o', 'ms', 'mt', 'nc', 'nd', 'ne', 'nh', 'nj', 'nm', 'nv', 'ny', 'oh', 'o
k', 'or', 'pa', 'ri', 'sc', 'sd', 'tn', 'tx', 'ut', 'va', 'vt', 'wa', 'w
i', 'wv', 'wy']
--------------------------------------------------------------------------
--------------------------
After vectorizations
(53531, 51) (53531, 2)
(22942, 51) (22942, 2)
(32775, 51) (32775, 2)
```

In [0]:

```
vectorizer = CountVectorizer()
vectorizer.fit(X_train['project_grade_category'].values)

X_train_grade_one_hot = vectorizer.transform(X_train['project_grade_category'].values)
X_cv_grade_one_hot = vectorizer.transform(X_cv['project_grade_category'].values)
X_test_grade_one_hot = vectorizer.transform(X_test['project_grade_category'].values)
print(vectorizer.get_feature_names())
print("-"*100)
print("After vectorizations")
print(X_train_grade_one_hot.shape, y_train.shape)
print(X_cv_grade_one_hot.shape, y_cv.shape)
print(X_test_grade_one_hot.shape, y_test.shape)
```

```
['grades_3_5', 'grades_6_8', 'grades_9_12', 'grades_prek_2']
--------------------------------------------------------------------------
--------------------------
After vectorizations
(53531, 4) (53531, 2)
(22942, 4) (22942, 2)
(32775, 4) (32775, 2)
```

CREATING INPUT DATA OTHER THAN TEXT

In [0]:

```
train_without_text=hstack((X_train_categories_one_hot,X_train_prefix_one_hot,X_train_st
ate_one_hot,X_train_subcategories_one_hot,X_train_grade_one_hot,X_train_price_norm,X_tr
ain_projects_norm)).todense()
cv_without_text=hstack((X_cv_categories_one_hot,X_cv_prefix_one_hot,X_cv_state_one_hot,
X_cv_subcategories_one_hot,X_cv_grade_one_hot,X_cv_price_norm,X_cv_projects_norm)).tode
nse()
test_without_text=hstack((X_test_categories_one_hot,X_test_prefix_one_hot,X_test_state_
one_hot,X_test_subcategories_one_hot,X_test_grade_one_hot,X_test_price_norm,X_test_proj
ects_norm)).todense()
print(train_without_text.shape)
print(cv_without_text.shape)
print(test_without_text.shape)
```

```
(53531, 101)
(22942, 101)
(32775, 101)
```

In [0]:

```
train_final = np.expand_dims(train_without_text,2)
cv_final = np.expand_dims(cv_without_text,2)
test_final=np.expand_dims(test_without_text,2)
print(train_final.shape)
print(cv_final.shape)
print(test_final.shape)
```

```
(53531, 101, 1)
(22942, 101, 1)
(32775, 101, 1)
```

MODEL 3

In [0]:

```
essay_text=Input(shape=(300,),name='essay_text')
x=Embedding(vocab_size,300,weights=[embedding_matrix_train],input_length=300)(essay_text)
lstm_1=LSTM(50,recurrent_dropout=0.5,return_sequences=True)(x)
flatten_1=Flatten()(lstm_1)

other_features=Input(shape=(101,1),name='other_features')
x = Conv1D(filters=128, kernel_size = 2, padding='valid', kernel_initializer='he_normal',)(other_features)
x = Conv1D(filters=128, kernel_size = 2, padding='valid', kernel_initializer='he_normal',)(x)
x = Flatten()(x)

con=concatenate([flatten_1,x])

x=Dense(128, activation='relu',kernel_initializer="he_normal",kernel_regularizer=l2(0.001))(con)
x=Dropout(0.5)(x)
x=Dense(128,activation='relu',kernel_initializer="he_normal",kernel_regularizer=l2(0.001))(x)
x=Dropout(0.5)(x)
x=BatchNormalization()(x)
x=Dense(128, activation='relu',kernel_initializer="he_normal",kernel_regularizer=l2(0.001))(x)

final_output = Dense(2, activation='softmax')(x)

model_3 = Model(inputs=[essay_text,other_features], outputs=[final_output])
print(model_3.summary())
```

```
Model: "model_3"
__________________________________________________________________________
________________________
Layer (type)                    Output Shape         Param #     Connected
to
==========================================================================
========================
essay_text (InputLayer)         (None, 300)          0
__________________________________________________________________________
________________________
other_features (InputLayer)     (None, 101, 1)       0
__________________________________________________________________________
________________________
embedding_13 (Embedding)        (None, 300, 300)     12768900    essay_tex
t[0][0]
__________________________________________________________________________
________________________
conv1d_1 (Conv1D)               (None, 100, 128)     384         other_fea
tures[0][0]
__________________________________________________________________________
________________________
lstm_3 (LSTM)                   (None, 300, 50)      70200       embedding
_13[0][0]
__________________________________________________________________________
________________________
conv1d_2 (Conv1D)               (None, 99, 128)      32896       conv1d_1
[0][0]
__________________________________________________________________________
________________________
flatten_13 (Flatten)            (None, 15000)        0           lstm_3[0]
[0]
__________________________________________________________________________
________________________
flatten_14 (Flatten)            (None, 12672)        0           conv1d_2
[0][0]
__________________________________________________________________________
________________________
concatenate_3 (Concatenate)     (None, 27672)        0           flatten_1
3[0][0]
                                                                 flatten_1
4[0][0]
__________________________________________________________________________
________________________
dense_11 (Dense)                (None, 128)          3542144     concatena
te_3[0][0]
__________________________________________________________________________
________________________
dropout_5 (Dropout)             (None, 128)          0           dense_11
[0][0]
__________________________________________________________________________
________________________
dense_12 (Dense)                (None, 128)          16512       dropout_5
[0][0]
__________________________________________________________________________
________________________
dropout_6 (Dropout)             (None, 128)          0           dense_12
[0][0]
__________________________________________________________________________
________________________
batch_normalization_3 (BatchNor (None, 128)          512         dropout_6
[0][0]
```

```
__________________________________________________________________________
________________________
dense_13 (Dense)                (None, 128)          16512       batch_nor
malization_3[0][0]
__________________________________________________________________________
________________________
dense_14 (Dense)                (None, 2)            258         dense_13
[0][0]
==========================================================================
========================
Total params: 16,448,318
Trainable params: 16,448,062
Non-trainable params: 256
__________________________________________________________________________
________________________
None
```

In [0]:

```python
from keras.callbacks import TensorBoard
checkpoint_3 = ModelCheckpoint("model_3.1",
                             monitor="val_auroc",
                             mode="max",
                             save_best_only = True,
                             verbose=1)
earlystop = EarlyStopping(monitor = 'val_auroc',
                            mode="max",
                            min_delta = 0,
                            patience = 3,
                            verbose = 1,)
tensorboard_2=TensorBoard(log_dir='lstm_3',batch_size=512)
callbacks_2=[checkpoint_3,earlystop,tensorboard_2]
```

In [0]:

```python
train_model_3=[padded_train,train_final]
cv_model_3=[padded_cv,cv_final]
test_model_3=[padded_test,test_final]
```

In [0]:

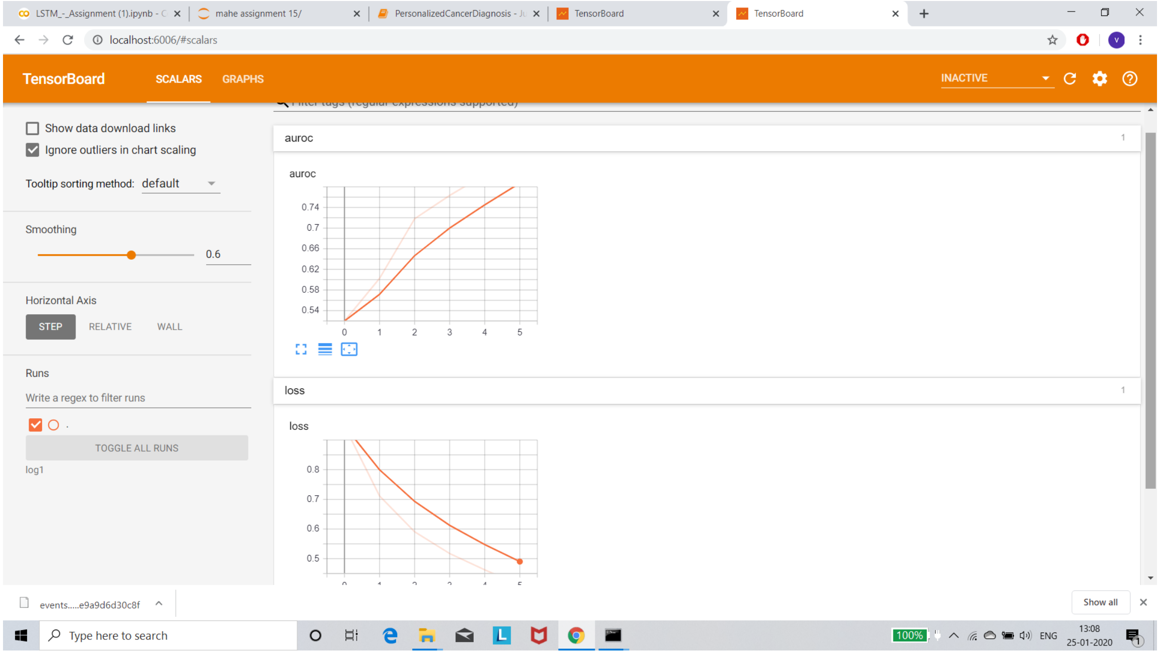
```
model_3.compile(optimizer='adam', loss='categorical_crossentropy', metrics=[auroc])
LSTM_3 = model_3.fit(train_model_3, y_train, batch_size=512, epochs=10, verbose=1,callb
acks=callbacks_2, validation_data=(cv_model_3, y_cv))
```

```
Train on 53531 samples, validate on 22942 samples
Epoch 1/10
53531/53531 [==============================] - 509s 10ms/step - loss: 1.19
05 - auroc: 0.5222 - val_loss: 0.9167 - val_auroc: 0.5946

Epoch 00001: val_auroc improved from -inf to 0.59462, saving model to mode
l_3.1
Epoch 2/10
53531/53531 [==============================] - 504s 9ms/step - loss: 0.850
1 - auroc: 0.5629 - val_loss: 0.7516 - val_auroc: 0.6469

Epoch 00002: val_auroc improved from 0.59462 to 0.64692, saving model to m
odel_3.1
Epoch 3/10
53531/53531 [==============================] - 496s 9ms/step - loss: 0.708
0 - auroc: 0.6658 - val_loss: 0.6449 - val_auroc: 0.7220

Epoch 00003: val_auroc improved from 0.64692 to 0.72202, saving model to m
odel_3.1
Epoch 4/10
53531/53531 [==============================] - 495s 9ms/step - loss: 0.609
6 - auroc: 0.7355 - val_loss: 0.6054 - val_auroc: 0.7544

Epoch 00004: val_auroc improved from 0.72202 to 0.75437, saving model to m
odel_3.1
Epoch 5/10
53531/53531 [==============================] - 495s 9ms/step - loss: 0.537
5 - auroc: 0.7747 - val_loss: 0.5510 - val_auroc: 0.7550

Epoch 00005: val_auroc improved from 0.75437 to 0.75499, saving model to m
odel_3.1
Epoch 6/10
53531/53531 [==============================] - 497s 9ms/step - loss: 0.487
6 - auroc: 0.8028 - val_loss: 0.5407 - val_auroc: 0.7480

Epoch 00006: val_auroc did not improve from 0.75499
Epoch 7/10
53531/53531 [==============================] - 492s 9ms/step - loss: 0.435
6 - auroc: 0.8382 - val_loss: 0.5047 - val_auroc: 0.7243

Epoch 00007: val_auroc did not improve from 0.75499
Epoch 8/10
53531/53531 [==============================] - 492s 9ms/step - loss: 0.383
0 - auroc: 0.8763 - val_loss: 0.5082 - val_auroc: 0.7154

Epoch 00008: val_auroc did not improve from 0.75499
Epoch 00008: early stopping
```

In [0]:

```
Image(retina=True, filename='/content/Screenshot (28).png')
```

Out[0]:

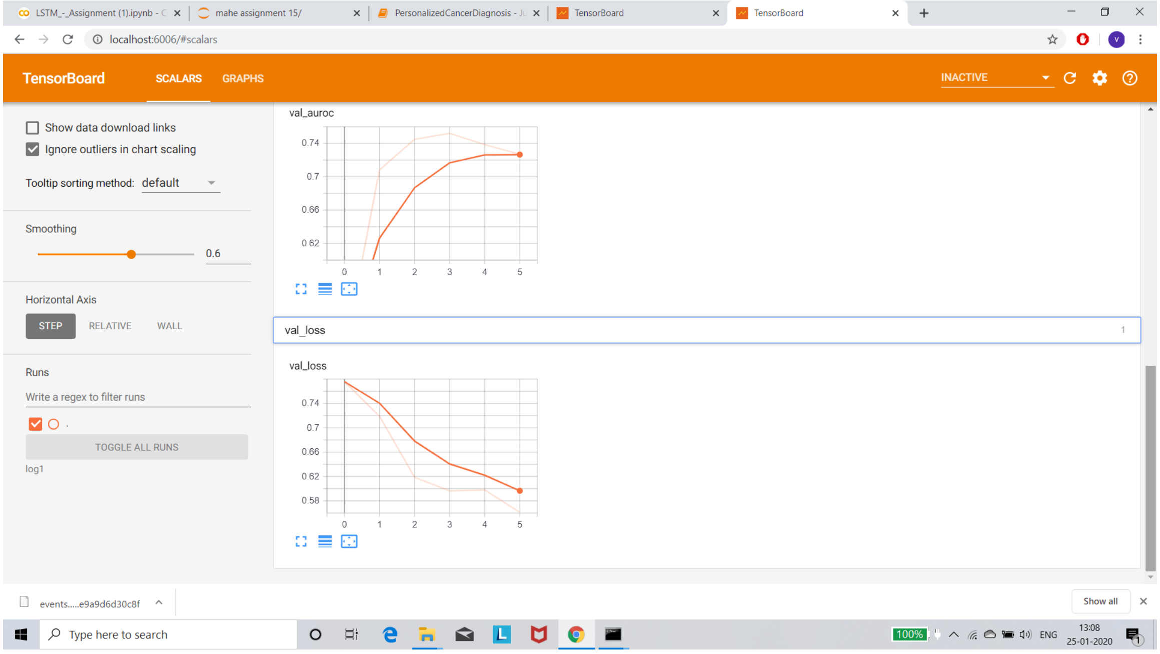


☐ Show data download links
☑ Ignore outliers in chart scaling
Tooltip sorting method: default
Smoothing 0.6
Horizontal Axis: STEP

In [0]:

```
Image(retina=True, filename='/content/Screenshot (29).png')
```

Out[0]:



☐ Show data download links
☑ Ignore outliers in chart scaling
Tooltip sorting method: default
Smoothing 0.6
Horizontal Axis: STEP

CONCLUSION

In [43]:

```
from prettytable import PrettyTable

conclusion=PrettyTable()
conclusion.field_names = ["ARCHITECTURE","EPOCHS", "TRAIN_auc", "VAL_auc",]
conclusion.add_row(["LSTM_1", 4, 0.7632,0.7520 ])
conclusion.add_row(["LSTM_2",5, 0.7545,0.7470])
conclusion.add_row(["LSTM_3", 5,0.7747,0.7550])

print(conclusion.get_string(start=0,end=7))
```

```
+--------------+--------+-----------+---------+
| ARCHITECTURE | EPOCHS | TRAIN_auc | VAL_auc |
+--------------+--------+-----------+---------+
|    LSTM_1    |   4    |   0.7632  |  0.752  |
|    LSTM_2    |   5    |   0.7545  |  0.747  |
|    LSTM_3    |   5    |   0.7747  |  0.755  |
+--------------+--------+-----------+---------+
```

In [0]:

```

```