Assignment : 14

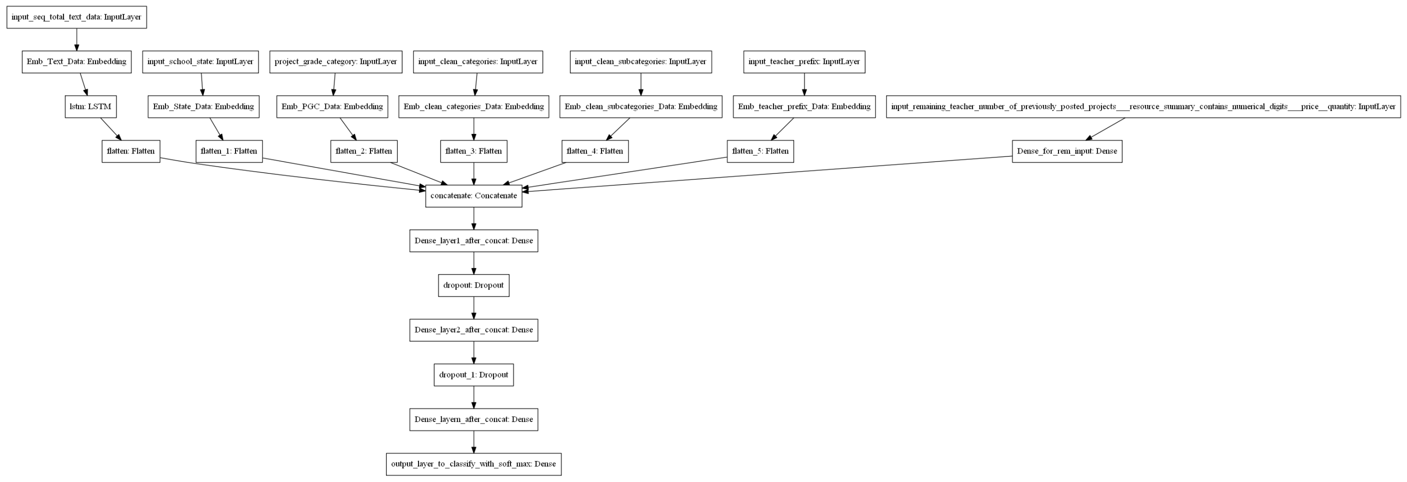
In [0]:
```
from google.colab import drive
drive.mount('/content/drive')
```

```
Go to this URL in a browser: https://accounts.google.com/o/oauth2/auth?client
_id=947318989803-6bn6qk8qdgf4n4g3pfee6491hc0brc4i.apps.googleusercontent.com&
redirect_uri=urn%3Aietf%3Awg%3Aoauth%3A2.0%3Aoob&scope=email%20https%3A%2F%2F
www.googleapis.com%2Fauth%2Fdocs.test%20https%3A%2F%2Fwww.googleapis.com%2Fau
th%2Fdrive%20https%3A%2F%2Fwww.googleapis.com%2Fauth%2Fdrive.photos.readonly%
20https%3A%2F%2Fwww.googleapis.com%2Fauth%2Fpeopleapi.readonly&response_type=
code

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

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_nun** ---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-la
yer-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.

In [0]:
```
# importing required libraries
import warnings
warnings.filterwarnings("ignore")
import pandas as pd
import numpy as np
from keras.layers import Input, Embedding, LSTM, Dropout, BatchNormalization,
Dense, concatenate, Flatten, Conv1D, MaxPool1D, LeakyReLU, ELU, SpatialDropout
1D, MaxPooling1D, GlobalAveragePooling1D, GlobalMaxPooling1D
from keras.preprocessing.text import Tokenizer, one_hot
from keras.preprocessing.sequence import pad_sequences
from keras.models import Model, load_model
from keras import regularizers
from keras.optimizers import *
from keras.callbacks import ModelCheckpoint, EarlyStopping, TensorBoard, Reduc
eLROnPlateau
from sklearn.feature_extraction.text import TfidfVectorizer, CountVectorizer
from sklearn.metrics import roc_auc_score
import tensorflow as tf
import matplotlib.pyplot as plt
%matplotlib inline
import re
from tqdm import tqdm
from sklearn.preprocessing import LabelEncoder
import seaborn as sns
import pickle
```

```
Using TensorFlow backend.
```

In [0]:
```
%matplotlib inline
import warnings
warnings.filterwarnings("ignore")

import sqlite3
import pandas as pd
import numpy as np
import nltk
import string
import matplotlib.pyplot as plt
import seaborn as sns
from sklearn.feature_extraction.text import TfidfTransformer
from sklearn.feature_extraction.text import TfidfVectorizer

from sklearn.feature_extraction.text import CountVectorizer
from sklearn.metrics import confusion_matrix
from sklearn import metrics
from sklearn.metrics import roc_curve, auc
from nltk.stem.porter import PorterStemmer

import re
# Tutorial about Python regular expressions: https://pymotw.com/2/re/
import string
from nltk.corpus import stopwords
from nltk.stem import PorterStemmer
from nltk.stem.wordnet import WordNetLemmatizer

from gensim.models import Word2Vec
from gensim.models import KeyedVectors
import pickle

from tqdm import tqdm_notebook as tqdm1
from tqdm import tqdm
import time
import os

from plotly import plotly
import plotly.offline as offline
import plotly.graph_objs as go
offline.init_notebook_mode()
from collections import Counter

from sklearn.model_selection import train_test_split
```

In [0]:
```
# project_data = pd.read_csv('preprocessed_data.csv')
# project_data.head()
```

In [0]:
```
# project_data = pd.read_csv('train_data.csv', nrows=50000)
project_data = pd.read_csv('/content/drive/My Drive/AAIC/Assignments/train_dat
a.csv')
resource_data = pd.read_csv('/content/drive/My Drive/AAIC/Assignments/resource
s.csv')
```

In [0]:
```
print("Number of data points in train data", project_data.shape)
print('-'*50)
print("The attributes of data :", project_data.columns.values)
```

```
Number of data points in train data (109248, 17)
--------------------------------------------------
The attributes of data : ['Unnamed: 0' 'id' 'teacher_id' 'teacher_prefix' 'sc
hool_state'
 'project_submitted_datetime' 'project_grade_category'
 'project_subject_categories' 'project_subject_subcategories'
 'project_title' 'project_essay_1' 'project_essay_2' 'project_essay_3'
 'project_essay_4' 'project_resource_summary'
 'teacher_number_of_previously_posted_projects' 'project_is_approved']
```

Text preprocessing(1)

In [0]:
```
catogories = list(project_data['project_subject_categories'].values)
# remove special characters from list of strings python: https://stackoverflo
w.com/a/47301924/4084039

# https://www.geeksforgeeks.org/removing-stop-words-nltk-python/
# https://stackoverflow.com/questions/23669024/how-to-strip-a-specific-word-fr
om-a-string
# https://stackoverflow.com/questions/8270092/remove-all-whitespace-in-a-strin
g-in-python
cat_list = []
for i in catogories:
    temp = ""
    # consider we have text like this "Math & Science, Warmth, Care & Hunger"
    for j in i.split(','): # it will split it in three parts ["Math & Scienc
e", "Warmth", "Care & Hunger"]
        if 'The' in j.split(): # this will split each of the catogory based on
space "Math & Science"=> "Math","&", "Science"
            j=j.replace('The','') # if we have the words "The" we are going to
replace it with ''(i.e removing 'The')
        j = j.replace(' ','') # we are placeing all the ' '(space) with ''(emp
ty) ex:"Math & Science"=>"Math&Science"
        temp+=j.strip()+" " #" abc ".strip() will return "abc", remove the tra
iling spaces
        temp = temp.replace('&','_') # we are replacing the & value into
    cat_list.append(temp.strip())
```

In [0]:
```
project_data['clean_categories'] = cat_list
project_data.drop(['project_subject_categories'], axis=1, inplace=True)
project_data.head(5)
```

Out[0]:

	Unnamed: 0	id	teacher_id	teacher_prefix	school_state	project_
0	160221	p253737	c90749f5d961ff158d4b4d1e7dc665fc	Mrs.	IN	
1	140945	p258326	897464ce9ddc600bced1151f324dd63a	Mr.	FL	
2	21895	p182444	3465aaf82da834c0582ebd0ef8040ca0	Ms.	AZ	
3	45	p246581	f3cb9bffbba169bef1a77b243e620b60	Mrs.	KY	
4	172407	p104768	be1f7507a41f8479dc06f047086a39ec	Mrs.	TX	

In [0]:
```
# count of all the words in corpus python: https://stackoverflow.com/a/2289859
5/4084039
from collections import Counter
my_counter = Counter()
for word in project_data['clean_categories'].values:
    my_counter.update(word.split())
my_counter
```

Out[0]:
```
Counter({'AppliedLearning': 12135,
         'Care_Hunger': 1388,
         'Health_Sports': 14223,
         'History_Civics': 5914,
         'Literacy_Language': 52239,
         'Math_Science': 41421,
         'Music_Arts': 10293,
         'SpecialNeeds': 13642,
         'Warmth': 1388})
```

In [0]:
```
# dict sort by value python: https://stackoverflow.com/a/613218/4084039
cat_dict = dict(my_counter)
sorted_cat_dict = dict(sorted(cat_dict.items(), key=lambda kv: kv[1]))


# ind = np.arange(len(sorted_cat_dict))
# plt.figure(figsize=(20,5))
# p1 = plt.bar(ind, list(sorted_cat_dict.values()))

# plt.ylabel('Projects')
# plt.title('% of projects aproved category wise')
# plt.xticks(ind, list(sorted_cat_dict.keys()))
# plt.show()
# print(sorted_cat_dict)
```

In [0]:
```
sub_catogories = list(project_data['project_subject_subcategories'].values)
# remove special characters from list of strings python: https://stackoverflo
w.com/a/47301924/4084039

# https://www.geeksforgeeks.org/removing-stop-words-nltk-python/
# https://stackoverflow.com/questions/23669024/how-to-strip-a-specific-word-fr
om-a-string
# https://stackoverflow.com/questions/8270092/remove-all-whitespace-in-a-strin
g-in-python

sub_cat_list = []
for i in sub_catogories:
    temp = ""
    # consider we have text like this "Math & Science, Warmth, Care & Hunger"
    for j in i.split(','): # it will split it in three parts ["Math & Scienc
e", "Warmth", "Care & Hunger"]
        if 'The' in j.split(): # this will split each of the catogory based on
space "Math & Science"=> "Math","&", "Science"
            j=j.replace('The','') # if we have the words "The" we are going to
replace it with ''(i.e removing 'The')
        j = j.replace(' ','') # we are placeing all the ' '(space) with ''(emp
ty) ex:"Math & Science"=>"Math&Science"
        temp +=j.strip()+" "#" abc ".strip() will return "abc", remove the tra
iling spaces
        temp = temp.replace('&','_')
    sub_cat_list.append(temp.strip())
```

In [0]:
```
project_data['clean_subcategories'] = sub_cat_list
project_data.drop(['project_subject_subcategories'], axis=1, inplace=True)
project_data.head(2)
```

Out[0]:

	Unnamed: 0	id	teacher_id	teacher_prefix	school_state	project_
0	160221	p253737	c90749f5d961ff158d4b4d1e7dc665fc	Mrs.	IN	
1	140945	p258326	897464ce9ddc600bced1151f324dd63a	Mr.	FL	

In [0]:
```
# count of all the words in corpus python: https://stackoverflow.com/a/2289859
5/4084039
from collections import Counter
my_counter = Counter()
for word in project_data['clean_subcategories'].values:
    my_counter.update(word.split())
```

In [0]:
```
# dict sort by value python: https://stackoverflow.com/a/613218/4084039
sub_cat_dict = dict(my_counter)
sorted_sub_cat_dict = dict(sorted(sub_cat_dict.items(), key=lambda kv: kv[1]))


# ind = np.arange(len(sorted_sub_cat_dict))
# plt.figure(figsize=(20,5))
# p1 = plt.bar(ind, list(sorted_sub_cat_dict.values()))

# plt.ylabel('Projects')
# plt.title('% of projects aproved state wise')
# plt.xticks(ind, list(sorted_sub_cat_dict.keys()))
# plt.show()
```

In [0]:
```
# merge two column text dataframe:
project_data["essay"] = project_data["project_essay_1"].map(str) +\
                        project_data["project_essay_2"].map(str) + \
                        project_data["project_essay_3"].map(str) + \
                        project_data["project_essay_4"].map(str)
```

In [0]:
```
# https://stackoverflow.com/questions/22407798/how-to-reset-a-dataframes-index
es-for-all-groups-in-one-step
price_data = resource_data.groupby('id').agg({'price':'sum', 'quantity':'sum'
}).reset_index()
price_data.head(2)
```

Out[0]:

	id	price	quantity
0	p000001	459.56	7
1	p000002	515.89	21

In [0]:
```
# join two dataframes in python:
project_data = pd.merge(project_data, price_data, on='id', how='left')
```

In [0]:
```
#presence of the numerical digits in a strings with numeric : https://stackove
rflow.com/a/19859308/8089731
def hasNumbers(inputString):
    return any(i.isdigit() for i in inputString)
p1 = project_data[['id','project_resource_summary']]
p1 = pd.DataFrame(data=p1)
p1.columns = ['id','digits_in_summary']
p1['digits_in_summary'] =  p1['digits_in_summary'].map(hasNumbers)
# https://stackoverflow.com/a/17383325/8089731
p1['digits_in_summary'] = p1['digits_in_summary'].astype(int)
project_data = pd.merge(project_data, p1, on='id', how='left')
project_data.head(5)
```

Out[0]:

	Unnamed: 0	id	teacher_id	teacher_prefix	school_state	project_
0	160221	p253737	c90749f5d961ff158d4b4d1e7dc665fc	Mrs.	IN	
1	140945	p258326	897464ce9ddc600bced1151f324dd63a	Mr.	FL	
2	21895	p182444	3465aaf82da834c0582ebd0ef8040ca0	Ms.	AZ	
3	45	p246581	f3cb9bffbba169bef1a77b243e620b60	Mrs.	KY	
4	172407	p104768	be1f7507a41f8479dc06f047086a39ec	Mrs.	TX	

Text preprocessing(2)

In [0]:
```
# https://stackoverflow.com/a/47091490/4084039
import re

def decontracted(phrase):
    # specific
    phrase = re.sub(r"won't", "will not", phrase)
    phrase = re.sub(r"can\'t", "can not", phrase)

    # general
    phrase = re.sub(r"n\'t", " not", phrase)
    phrase = re.sub(r"\'re", " are", phrase)
    phrase = re.sub(r"\'s", " is", phrase)
    phrase = re.sub(r"\'d", " would", phrase)
    phrase = re.sub(r"\'ll", " will", phrase)
    phrase = re.sub(r"\'t", " not", phrase)
    phrase = re.sub(r"\'ve", " have", phrase)
    phrase = re.sub(r"\'m", " am", phrase)
    return phrase
```

In [0]:
```
# https://gist.github.com/sebleier/554280
# we are removing the words from the stop words list: 'no', 'nor', 'not'
stopwords= ['i', 'me', 'my', 'myself', 'we', 'our', 'ours', 'ourselves', 'you'
, "you're", "you've",\
            "you'll", "you'd", 'your', 'yours', 'yourself', 'yourselves', 'he'
, 'him', 'his', 'himself', \
            'she', "she's", 'her', 'hers', 'herself', 'it', "it's", 'its', 'it
self', 'they', 'them', 'their',\
            'theirs', 'themselves', 'what', 'which', 'who', 'whom', 'this', 't
hat', "that'll", 'these', 'those', \
            'am', 'is', 'are', 'was', 'were', 'be', 'been', 'being', 'have',
'has', 'had', 'having', 'do', 'does', \
            'did', 'doing', 'a', 'an', 'the', 'and', 'but', 'if', 'or', 'becau
se', 'as', 'until', 'while', 'of', \
            'at', 'by', 'for', 'with', 'about', 'against', 'between', 'into',
'through', 'during', 'before', 'after',\
            'above', 'below', 'to', 'from', 'up', 'down', 'in', 'out', 'on',
'off', 'over', 'under', 'again', 'further',\
            'then', 'once', 'here', 'there', 'when', 'where', 'why', 'how', 'a
ll', 'any', 'both', 'each', 'few', 'more',\
            'most', 'other', 'some', 'such', 'only', 'own', 'same', 'so', 'tha
n', 'too', 'very', \
            's', 't', 'can', 'will', 'just', 'don', "don't", 'should', "shoul
d've", 'now', 'd', 'll', 'm', 'o', 're', \
            've', 'y', 'ain', 'aren', "aren't", 'couldn', "couldn't", 'didn',
"didn't", 'doesn', "doesn't", 'hadn',\
            "hadn't", 'hasn', "hasn't", 'haven', "haven't", 'isn', "isn't", 'm
a', 'mightn', "mightn't", 'mustn',\
            "mustn't", 'needn', "needn't", 'shan', "shan't", 'shouldn', "shoul
dn't", 'wasn', "wasn't", 'weren', "weren't", \
            'won', "won't", 'wouldn', "wouldn't"]
```

In [0]:
```
# Combining all the above statemennts
from tqdm import tqdm
preprocessed_essays = []
# tqdm is for printing the status bar
for sentance in tqdm1(project_data['essay'].values):
    sent = decontracted(sentance)
    sent = sent.replace('\\r', ' ')
    sent = sent.replace('\\"', ' ')
    sent = sent.replace('\\n', ' ')
    sent = re.sub('[^A-Za-z0-9]+', ' ', sent)
    sent = re.sub('nannan', '', sent)
    # https://gist.github.com/sebleier/554280
    sent = ' '.join(e for e in sent.split() if e not in stopwords)
    preprocessed_essays.append(sent.lower().strip())
```

In [0]:
```
from tqdm import tqdm
preprocessed_titles = []
# tqdm is for printing the status bar
for title in tqdm1(project_data['project_title'].values):
    _title = decontracted(title)
    _title = _title.replace('\\r', ' ')
    _title = _title.replace('\\"', ' ')
    _title = _title.replace('\\n', ' ')
    _title = re.sub('[^A-Za-z0-9]+', ' ', _title)
    # https://gist.github.com/sebleier/554280
    _title = ' '.join(e for e in _title.split() if e not in stopwords)
    preprocessed_titles.append(_title.lower().strip())
```

In [0]:
```
preprocessed_titles[1000]
```

Out[0]:
```
'sailing into super 4th grade year'
```

In [0]:
```
project_grade_catogories = list(project_data['project_grade_category'].values)
# remove special characters from list of strings python: https://stackoverflo
w.com/a/47301924/4084039

# https://www.geeksforgeeks.org/removing-stop-words-nltk-python/
# https://stackoverflow.com/questions/23669024/how-to-strip-a-specific-word-fr
om-a-string
# https://stackoverflow.com/questions/8270092/remove-all-whitespace-in-a-strin
g-in-python

project_grade_cat_list = []
for i in tqdm1(project_grade_catogories):
    temp = ""
    # consider we have text like this "Math & Science, Warmth, Care & Hunger"
    for j in i.split(','): # it will split it in three parts ["Math & Scienc
e", "Warmth", "Care & Hunger"]
        if 'The' in j.split(): # this will split each of the catogory based on
space "Math & Science"=> "Math","&", "Science"
            j=j.replace('The','') # if we have the words "The" we are going to
replace it with ''(i.e removing 'The')
        j = j.replace(' ','') # we are placeing all the ' '(space) with ''(emp
ty) ex:"Math & Science"=>"Math&Science"
        temp +=j.strip()+" "#" abc ".strip() will return "abc", remove the tra
iling spaces
        temp = temp.replace('&','_')
    project_grade_cat_list.append(temp.strip())
```

In [0]:
```
project_data['clean_project_grade_category'] = project_grade_cat_list
project_data.drop(['project_grade_category'], axis=1, inplace=True)
project_data.head(2)
```

Out[0]:

	Unnamed: 0	id	teacher_id	teacher_prefix	school_state	project_
0	160221	p253737	c90749f5d961ff158d4b4d1e7dc665fc	Mrs.	IN	
1	140945	p258326	897464ce9ddc600bced1151f324dd63a	Mr.	FL	

In [0]:
```
project_data.drop(['project_essay_1','project_essay_2','project_essay_3','proj
ect_essay_4'], axis=1, inplace=True)
project_data.head(2)
```

Out[0]:

	Unnamed: 0	id	teacher_id	teacher_prefix	school_state	project_
0	160221	p253737	c90749f5d961ff158d4b4d1e7dc665fc	Mrs.	IN	
1	140945	p258326	897464ce9ddc600bced1151f324dd63a	Mr.	FL	

In [0]:
```
#Replacing Nan's with maximum occured value: https://stackoverflow.com/a/51053
916/8089731
project_data['teacher_prefix'].value_counts().argmax()
project_data.fillna(value=project_data['teacher_prefix'].value_counts().argmax
(),axis=1,inplace=True)
```

In [0]:
```
project_data['preprocessed_essays'] = preprocessed_essays
project_data['preprocessed_titles'] = preprocessed_titles
```

In [0]:
```
X_train, X_test, y_train, y_test = train_test_split(project_data,project_data[
'project_is_approved'], test_size=0.33, stratify = project_data['project_is_ap
proved'])
# X_train, X_cv, y_train, y_cv = train_test_split(X_train, y_train, test_size=
0.33, stratify=y_train)

X_train.drop(['project_is_approved'], axis=1, inplace=True)
X_test.drop(['project_is_approved'], axis=1, inplace=True)
# X_cv.drop(['project_is_approved'], axis=1, inplace=True)
print(X_train.shape)
print(X_test.shape)
```

```
(73196, 18)
(36052, 18)
```

In [0]:
```
X_train
```

Out[0]:

	Unnamed: 0	id	teacher_id	teacher_prefix	school_state	p
74816	66347	p096461	17b3e30cc4663cbf3e30096644e9e825	Mr.	CO	
99412	36847	p097485	cf7a3bbb1e42e478156b38ba970ff9a1	Mrs.	NC	
21814	65679	p173394	00c14809f1ee207e4e5c0ee26fc50357	Mr.	IL	
9966	130421	p214011	9c55cc54ff0b38ce69959b3d22f206ef	Ms.	CA	
72147	90182	p190966	1f8ad9e87a1f3fe9eb6315044005a5e4	Mr.	OR	
49595	129996	p231073	9a001dad8acf41cb0c8a6fef21725192	Mr.	PA	
18802	170442	p172041	1dd06f59bbcd099b23d6df4fbf7c9ce1	Mrs.	NY	
82561	53923	p197065	1cf9bf227b6219273db2ab8ef504ae96	Ms.	MO	
6617	113020	p124227	eca2679f485b276396e09a671d817965	Mrs.	FL	
71003	102357	p011167	b86b594e8cc5fe325ec88fad97722b32	Ms.	TX	
90237	134822	p181352	611f8e78cb2c1b24fd72897f219482b6	Mrs.	NJ	
80944	176806	p086471	6d63ffaa64f6573357c7647730bfd14f	Mrs.	NC	
81008	164566	p123478	ba4621880f1d3bcd4c9ca87172d6e7fa	Ms.	NM	

	Unnamed: 0	id	teacher_id	teacher_prefix	school_state
65299	98401	p080491	28d592a964b88e923aa66b5a1bf689cf	Ms.	IL
93480	44625	p064644	797eb20fd3b66a5850596287bc2255c8	Mrs.	CA
41039	20049	p033416	1fd54b6791c04742b7d1983aab4006c5	Ms.	PA
18770	1004	p182065	e29dfbf8afa034d39619151779a2ead2	Ms.	CA
47523	79581	p194575	665bc98ef043b13efbda994c7869f4b5	Mrs.	CA
1296	118333	p147248	ef68d389ba1872eac06843d08791bc16	Mrs.	OH
77910	93959	p101495	8e11df30154e52df2b7d60d3a06999a3	Ms.	WA
65932	112641	p107640	ac628b25f39c93ba2819d4835bbf11f3	Mrs.	TX
52597	128497	p247125	43c03cd752da0b04507fdcbdf9b1c164	Mrs.	KS
62361	132297	p177358	f023c40dd3fb3a409436a932c7dc3b20	Ms.	SC
61185	70129	p215004	2e2741155d934bbcb2baa5169c564b72	Ms.	NY
31258	61859	p257689	eba503a200788e08fdb696af0d667610	Ms.	OH
55137	87358	p137631	7e7b7c54fd1d5de212338f45542ddbe2	Ms.	TX

	Unnamed: 0	id	teacher_id	teacher_prefix	school_state	p
36693	40838	p204901	3c8586ea656cda4fcfcef1dd9a87ff52	Mrs.	OH	
65979	16374	p259819	a4c147e61bacf1866e214a4941fbdc83	Mrs.	CA	
31311	40488	p020050	3ac8970fd982b40003944f7e849520f3	Mrs.	MN	
33770	69112	p031868	45baac20a1b01c42ccbcaec1478482e8	Ms.	OH	
...	...	...	...	...	...	
76237	80564	p054639	4159df7029269734d5b06694f8b4b602	Mrs.	LA	
75170	34437	p157983	f9bb8990266fe4f92de0a83b699cdd6f	Mr.	AZ	
49722	106585	p039311	fb0cba59eb388ac3981cd388c7b1ca61	Ms.	MT	
76806	160135	p148955	e5d225972a16b4acef9b5d858085d128	Ms.	GA	
60301	108582	p169531	bb00430c6bca9c097953f3cedd7fab17	Mrs.	NJ	
73896	114819	p173422	00331380c8003e87a614155866dd0ab8	Mrs.	CA	
63703	75219	p063808	935eee7f43eaaf89e5596fe204fc4b69	Mrs.	CA	
16745	25961	p101388	cb50b26941f5f567960f2b0f7f5cb970	Ms.	CA	
72879	110805	p009004	b09629f0e223854f29c77d54fb42feae	Teacher	TX	

	Unnamed: 0	id	teacher_id	teacher_prefix	school_state	p
52588	86128	p225789	0477dc22f5e80229f9b9cc992f38bfee	Mrs.	MN	
104744	6521	p151386	d4db9a5f645b0092ea6bc46fed28122b	Mrs.	PA	
18331	115009	p201569	4c10c52573696cc9d8e7cea3aff1b19d	Ms.	OH	
103126	159981	p075162	8952ce00faa054def806d011964d31ae	Teacher	IN	
20788	32079	p020823	8f0c219c2ac67b0664f6481de112bbee	Mrs.	OK	
57904	53367	p127439	bee92bef407e4980eed78be3601298d0	Ms.	CA	
8685	125095	p183578	eaa75436c10eb7a5a92115fd891f4714	Ms.	LA	
27893	161071	p097496	ff33887927df0a37d64774a0ee239c53	Mrs.	FL	
99490	8339	p246152	0de5b9af44dc5768d3b16f8ef3e5d46a	Mrs.	CA	
97249	3404	p220690	05cccd8723961d9f24053948e57ae157	Ms.	CO	
59680	156078	p124460	315c99648e6a73ae29034fb10b7042a3	Ms.	PA	
105492	181763	p135663	58d56903889b54dc7764743896048b05	Mrs.	NC	

	Unnamed: 0	id	teacher_id	teacher_prefix	school_state	p
4734	70978	p062984	f18380d0403f7fed56a391e0a86300a6	Mrs.	AZ	
100545	28564	p072017	0b4bcc10792f33609c8888759137cf0c	Mrs.	FL	
30567	104083	p063325	7c90c73496a5af77fccc6c201bec7e7b	Ms.	FL	
20353	61180	p113832	fa4505dfa0ddede6c9f94b0297226709	Mrs.	GA	
17857	75759	p240184	1f5ea7faea916cc931e93cd1b289e1e1	Ms.	VA	
19379	48370	p049537	c54e9a993bd878e293c34f890aacb3cf	Ms.	FL	
69097	144181	p114032	b0365f10b815902280d79f2e500ca3fc	Teacher	TX	
76413	45323	p237849	ecd3ceba1b7ef03d28d7e74d6866f75c	Ms.	CA	
107640	49706	p057594	b8f0cd67fb9e0de9040f28eba83ffcfd	Mrs.	UT	

73196 rows × 18 columns

In [0]:
```
import dill
# dill.dump_session('notebook_envall.db')
dill.load_session('notebook_envall.db')
```

```
Using TensorFlow backend.
```

In [0]:
```
from numpy import array
from numpy import asarray
from numpy import zeros
from keras.preprocessing.text import Tokenizer
from keras.preprocessing.sequence import pad_sequences
from keras.models import Sequential
from keras.layers import Dense
from keras.layers import Flatten
from keras.layers import Embedding
```

In [0]:
```
tokenizer = Tokenizer()
tokenizer.fit_on_texts(X_train["essay"].tolist())
seq_train = tokenizer.texts_to_sequences(X_train["essay"])
seq_test = tokenizer.texts_to_sequences(X_test["essay"])
```

In [0]:
```
vocab_size = len(tokenizer.word_index) + 1
```

In [0]:
```
padded_train = pad_sequences(seq_train,maxlen=800,padding='post', truncating=
'post')
padded_test = pad_sequences(seq_test, maxlen=800,padding='post', truncating='p
ost')
```

In [0]:
```
pickle_in = open("/content/drive/My Drive/AAIC/Assignments/glove_vectors","rb"
)
glove_words = pickle.load(pickle_in)
```

In [0]:
```
embedding_matrix = zeros((vocab_size, 300))
for word, i in tokenizer.word_index.items():
    embedding_vector = glove_words.get(word)
    if embedding_vector is not None:
        embedding_matrix[i] = embedding_vector
```

In [0]:
```
# input_text = Input(shape=(800,),name="input_text")
embedding_layer = Embedding(vocab_size,300,weights=[embedding_matrix],input_le
ngth=800,trainable=False)
input_text = Input(shape=(800,),name="input_text")
x = embedding_layer(input_text)
x = LSTM(128,recurrent_dropout=0.5,kernel_regularizer=regularizers.l2(0.001),r
eturn_sequences=True)(x)
flat_1 = Flatten()(x)
```

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

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

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

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

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

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

In [0]:
```
# Teacher Prefix
no_of_unique_prefix  = X_train["teacher_prefix"].nunique()
embed_size_prefix = int(min(np.ceil((no_of_unique_prefix)/2), 50 ))

input_prefix = Input(shape=(1,),name="teacher_prefix")
embed_prefix = Embedding(no_of_unique_prefix,embed_size_prefix,trainable=True)
(input_prefix)
flat_2 = Flatten()(embed_prefix)
lab_enc = LabelEncoder()
enc_prefix_train = lab_enc.fit_transform(X_train["teacher_prefix"])
X_test["teacher_prefix"] = X_test["teacher_prefix"].map(lambda s: ' ' if s not
in lab_enc.classes_ else s)
lab_enc.classes_ = np.append(lab_enc.classes_, ' ')
enc_prefix_test = lab_enc.transform(X_test["teacher_prefix"])
```

In [0]:
```
# School State
no_of_unique_state  = X_train["school_state"].nunique()
embed_size_state= int(min(np.ceil((no_of_unique_state)/2), 50 ))

input_state = Input(shape=(1,),name="school_prefix")
embed_state = Embedding(no_of_unique_state,embed_size_state,name="emb_state",t
rainable=True)(input_state)
flat_3 = Flatten()(embed_state)
enc_state_train = lab_enc.fit_transform(X_train["school_state"])
enc_state_test = lab_enc.transform(X_test["school_state"])
```

In [0]:
```
X_train
```

Out[0]:

	Unnamed: 0	id	teacher_id	teacher_prefix	school_state	p
61692	9233	p134729	2fba990f30bae627a30c12abc14b4ef8	Mrs.	LA	
101374	30334	p190416	b62d6e4ad22c5fcc2063d3eb99713627	Ms.	NY	
8708	89772	p149606	b642d19bb72bc81afe90784121562e92	Ms.	NM	
70173	6607	p009634	c751cdc864bdc94a7c73fbc8d81f6260	Ms.	LA	
90839	144635	p000962	413dd95145e3bc9940e70f7dcf1c6304	Mr.	AZ	
62631	78012	p212397	53ccd40b8bd6f268ab7a0c73be3c0128	Mrs.	TX	
41361	50382	p237295	50e93b8c452382361a4847f4bb3acf24	Mrs.	MI	
97600	28582	p136590	c0432ba8cc235436596dd1e9184ca59d	Mr.	CO	
67695	85668	p204009	c9fd190fc4673e37fdfd919816f8fb34	Ms.	CA	
62517	177330	p039526	aa96ef933643e81d26495f0cbaea40aa	Ms.	MI	
79924	49997	p092797	028c1a9b97dbe4382bde1768309029a6	Mrs.	IN	
3293	88593	p253595	32be6df3fae18951952a8d734e2c5282	Mrs.	IL	

	Unnamed: 0	id	teacher_id	teacher_prefix	school_state	p
20959	85447	p202376	9126ed70645e56668963f82f65049246	Mrs.	MI	
46118	144888	p065891	3733f7f569d0b1f5366929e50134d147	Ms.	NC	
52590	60373	p249054	7cdf8261f3995678e92f10ca3e9e4154	Ms.	CA	
53712	118959	p222662	0de9f14da9526236c3bfe68d74942961	Ms.	MO	
71562	180573	p119887	12095a3ab253dc5d76cd30a35a68d7f5	Mrs.	IN	
49044	97203	p054181	e5df9f1df68eaf72b513d0d87ca7fabf	Mrs.	NV	
17597	143412	p233096	ae3d1fed5e7d36ec6507720d2a6e19c8	Ms.	DC	
35975	155318	p032324	3e23eef919ed0d7eaebe5f7da7395353	Mrs.	LA	
11802	164921	p213956	7bd92da677bb1fab60eae4780fab4018	Ms.	NC	
7818	31126	p086960	6add52d24845c544880cb30c8ba861df	Mr.	OR	
71082	2906	p213225	2d6594ae664616d2649fc6513427e905	Mr.	CO	
53187	166431	p237910	88d99ccd29c97bf2e91d244f745652c5	Ms.	NC	

	Unnamed: 0	id	teacher_id	teacher_prefix	school_state	pı
87936	105035	p134626	1107c3baabc13d506f332fc1c92f43e2	Ms.	NE	
38345	123888	p054357	061456f27cf575c6ce9d2c4bdb761bc6	Ms.	MO	
12250	40928	p247256	76cdb75d01b2e87341bc40ed83534397	Ms.	CA	
24951	152108	p001117	abb220c60ae2b56e773a01652d5452b5	Mrs.	NY	
81247	44203	p078243	b1cb571b0125b72f4db62fd4fb5b9607	Ms.	SC	
39315	8277	p123224	d2510a557eb8c903d4aae58c0d9dabda	Ms.	UT	
...	...	...	...	...	...	
73516	166116	p208444	d4a9803a0281bc39880f3d593ef0983a	Ms.	NJ	
63010	8201	p160652	fc1769db9f1982d0c5f4317065090d00	Mrs.	MO	
44561	59556	p194709	84074b3aa2e7046e7fb465b6b510efeb	Mrs.	LA	
69035	5025	p042180	130182667468b0ec6ccfd106359f6143	Mrs.	CA	
37302	110982	p156758	5d0caac278c76b0b3fb8a9d4d413d495	Mrs.	OH	
35529	123949	p014992	5c65c01fac4e8b11b233db944c6dde78	Ms.	MO	

	Unnamed: 0	id	teacher_id	teacher_prefix	school_state	p
82657	35180	p207628	997a37e2e303548d637fbe924c26a5b2	Ms.	TX	
57159	26239	p029264	93e12c086635fccef958a765f1ef9d82	Ms.	IL	
24278	3987	p010222	9efea81af5c1e406f178754028df762d	Mrs.	CA	
44039	59264	p099155	6ad46801a6ac3ff111eb48a29a563102	Mrs.	OH	
29633	97774	p081868	7d71578de985d546bdc1db7425dbf534	Mrs.	OH	
32458	158351	p065941	5b7d16fec33d61c38b3684cbfa7c000f	Mr.	TX	
54308	15779	p081553	ee2e204481999bd839cea5fc3d293ec8	Mrs.	NY	
8929	181159	p228458	8c9234d6fd08bc6d548649698d385f87	Ms.	AZ	
79971	176465	p020345	a238b2ea3c69831edd224620b21c1fce	Mrs.	VA	
72750	169104	p120275	577a5a08365015d2e444ab832214c046	Ms.	NY	
82091	40001	p078763	f8fc4b35b8f87bc601c3cc0c662c5bbe	Mrs.	CO	
76005	59031	p180583	6256f3cb5c44aa6ab76fbf3b0b37327d	Ms.	NY	

	Unnamed: 0	id	teacher_id	teacher_prefix	school_state	p
71420	68777	p203595	dc56b5f1a30ba8f1ec4bdd8e397307f8	Mrs.	SC	
75662	165959	p066542	7709b9dbc2bea6ee0e52266a8df8559c	Ms.	MI	
103851	150825	p225662	20dc6dacfef7de81d46927325a7562e5	Ms.	LA	
19769	85663	p241125	1f0e2a8ba9e02b9ead4293ec2be83185	Mrs.	GA	
33667	138647	p091149	d599114df256ae1998b53b5b725e016c	Ms.	IL	
103201	17999	p177022	2e9925990a881fd36a09da73e7388365	Mrs.	OK	
54380	26631	p246832	a70a978e33eb178017623ce48202c808	Mrs.	MI	
53764	150418	p228011	144a533f809ba503b9b9774d7b16cbeb	Mrs.	GA	
19038	157043	p108749	d70065ac07d35035a7a42bd14ee10413	Mrs.	UT	
40900	53017	p111028	6b9ce31517454b321e83408c03990595	Mr.	NJ	
27017	126271	p191642	2de954340b83289b322a80f2cd641cb8	Ms.	IL	
61986	61301	p080839	547945c09afffecffebda041fc437602	Mrs.	NC	

73196 rows × 18 columns

In [0]:
```
#project_grade_category
no_of_unique_grade  = X_train["clean_project_grade_category"].nunique()
embed_size_grade = int(min(np.ceil((no_of_unique_grade)/2), 50 ))

input_grade= Input(shape=(1,),name="grade_cat")
embed_grade = Embedding(no_of_unique_grade,embed_size_grade,name="emb_grade",t
rainable=True)(input_grade)
flat_4 = Flatten()(embed_grade)
enc_grade_train = lab_enc.fit_transform(X_train["clean_project_grade_category"
])
enc_grade_test = lab_enc.transform(X_test["clean_project_grade_category"])
```

In [0]:
```
#project_subject_categories
no_of_unique_subcat  = X_train["clean_categories"].nunique()
embed_size_subcat = int(min(np.ceil((no_of_unique_subcat)/2), 50 ))

input_subcat= Input(shape=(1,),name="sub_cat")
embed_subcat = Embedding(no_of_unique_subcat,embed_size_subcat,name="emb_subca
t",trainable=True)(input_subcat)
flat_5 = Flatten()(embed_subcat)
lab_enc = LabelEncoder()
enc_subcat_train = lab_enc.fit_transform(X_train["clean_categories"])
X_test["clean_categories"] = X_test["clean_categories"].map(lambda s: ' ' if s
not in lab_enc.classes_ else s)
lab_enc.classes_ = np.append(lab_enc.classes_, ' ')
enc_subcat_test= lab_enc.transform(X_test["clean_categories"])
```

In [0]:
```
#project_subject_subcategories
no_of_unique_subcat_1  = X_train["clean_subcategories"].nunique()
embed_size_subcat_1 = int(min(np.ceil((no_of_unique_subcat_1)/2), 50 ))

input_subcat_1= Input(shape=(1,),name="sub_cat_1")
embedding_subcat_1 = Embedding(no_of_unique_subcat_1,embed_size_subcat_1,name=
"emb_subcat_1",trainable=True)(input_subcat_1)
flat_6 = Flatten()(embedding_subcat_1)
lab_enc = LabelEncoder()
enc_subcat_1_train = lab_enc.fit_transform(X_train["clean_subcategories"])
X_test["clean_subcategories"] = X_test["clean_subcategories"].map(lambda s: '
 ' if s not in lab_enc.classes_ else s)
lab_enc.classes_ = np.append(lab_enc.classes_, ' ')
enc_subcat_1_test= lab_enc.transform(X_test["clean_subcategories"])
```

In [0]:
```
numerical_train_a=X_train['digits_in_summary'].values.reshape(-1, 1)
numerical_train_b=X_train['price'].values.reshape(-1, 1)
numerical_train_c=X_train['quantity'].values.reshape(-1, 1)
numerical_train_d=X_train['teacher_number_of_previously_posted_projects'].valu
es.reshape(-1, 1)
numerical_test_a=X_test['digits_in_summary'].values.reshape(-1, 1)
numerical_test_b=X_test['price'].values.reshape(-1, 1)
numerical_test_c=X_test['quantity'].values.reshape(-1, 1)
numerical_test_d=X_test['teacher_number_of_previously_posted_projects'].values
.reshape(-1, 1)
numerical_train=np.concatenate((numerical_train_a,numerical_train_b,numerical_
train_c,numerical_train_d),axis=1)
numerical_test=np.concatenate((numerical_test_a,numerical_test_b,numerical_tes
t_c,numerical_test_d),axis=1)

from sklearn.preprocessing import StandardScaler
normal=StandardScaler()
normal_train=normal.fit_transform(numerical_train)
normal_test=normal.transform(numerical_test)
```

In [0]:
```
numerical_feats = Input(shape=(4,),name="numerical_features")
numerical_featss = Dense(100,activation="relu",kernel_initializer="he_normal",
kernel_regularizer=regularizers.l2(0.001))(numerical_feats)
```

In [0]:
```
x = concatenate([flat_1,flat_2,flat_3,flat_4,flat_5,flat_6,numerical_featss])
x = Dense(128,activation="relu", kernel_initializer="he_normal",kernel_regular
izer=regularizers.l2(0.001))(x)
x=Dropout(0.5)(x)
x = Dense(256,activation="relu",kernel_initializer="he_normal",kernel_regulari
zer=regularizers.l2(0.001))(x)
x=Dropout(0.5)(x)
x = Dense(64,activation="relu", kernel_initializer="he_normal",kernel_regulari
zer=regularizers.l2(0.001))(x)
x = BatchNormalization()(x)
output = Dense(2, activation='softmax', name='output')(x)
model_1 = Model(inputs=[input_text,input_prefix,input_state,input_grade,input_
subcat,input_subcat_1,numerical_feats],outputs=[output])
```

In [0]:
```
train_data_1 = [padded_train,enc_prefix_train,enc_state_train,enc_grade_train,
enc_subcat_train,enc_subcat_1_train,normal_train]
test_data_1 = [padded_test,enc_prefix_test,enc_state_test,enc_grade_test,enc_s
ubcat_test,enc_subcat_1_test,normal_test]
```

In [0]:
```
from keras.utils import np_utils
Y_train = np_utils.to_categorical(y_train, 2)
Y_test = np_utils.to_categorical(y_test, 2)
```

In [0]:
```
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]:
```
model_1.compile(optimizer=Adam(lr=0.001), loss='categorical_crossentropy', met
rics=[auroc])
```

```
WARNING:tensorflow:From /usr/local/lib/python3.6/dist-packages/keras/optimize
rs.py:793: The name tf.train.Optimizer is deprecated. Please use tf.compat.v
1.train.Optimizer instead.

WARNING:tensorflow:From <ipython-input-20-4a25250c5bd7>:5: py_func (from tens
orflow.python.ops.script_ops) is deprecated and will be removed in a future v
ersion.
Instructions for updating:
tf.py_func is deprecated in TF V2. Instead, there are two
    options available in V2.
    - tf.py_function takes a python function which manipulates tf eager
    tensors instead of numpy arrays. It's easy to convert a tf eager tensor t
o
    an ndarray (just call tensor.numpy()) but having access to eager tensors
    means `tf.py_function`s can use accelerators such as GPUs as well as
    being differentiable using a gradient tape.
    - tf.numpy_function maintains the semantics of the deprecated tf.py_func
    (it is not differentiable, and manipulates numpy arrays). It drops the
    stateful argument making all functions stateful.
```

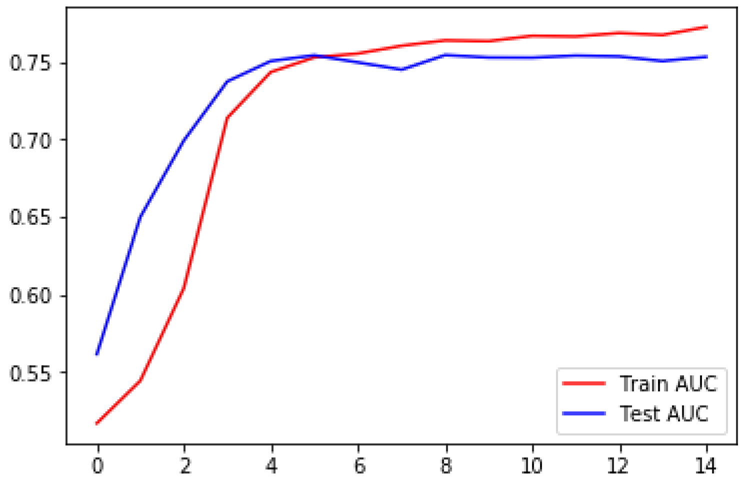
In [0]:
```
checkpoint_3 = ModelCheckpoint("model_1.h5",monitor="val_auroc",mode="max",sav
e_best_only = True,verbose=1)
NAME = 'model_1'
tensorboard_2 = TensorBoard(log_dir='logss\{}'.format(NAME),update_freq='epoc
h',batch_size=512)
callbacks_2 = [tensorboard_2,checkpoint_3]
```

In [0]:
```
history_1 = model_1.fit(train_data_1,Y_train,batch_size=512,
                        epochs=15,validation_data=(test_data_1,Y_test),verbose
=1,callbacks=callbacks_2)
```

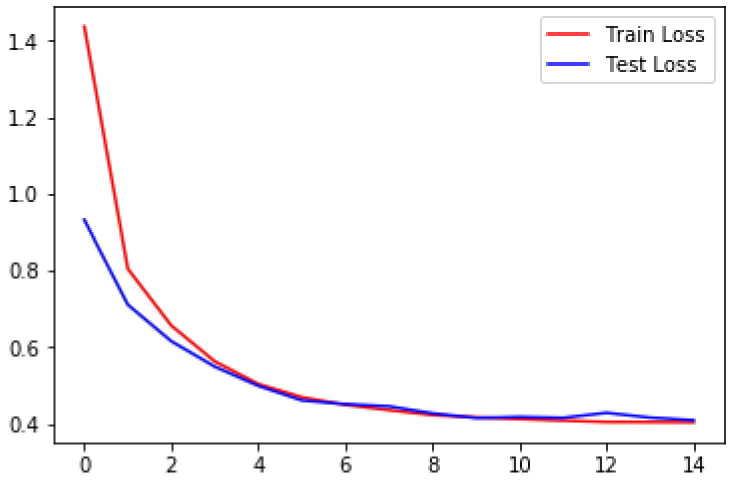
```
WARNING:tensorflow:From /usr/local/lib/python3.6/dist-packages/tensorflow/pyt
hon/ops/math_grad.py:1250: add_dispatch_support.<locals>.wrapper (from tensor
flow.python.ops.array_ops) is deprecated and will be removed in a future vers
ion.
Instructions for updating:
Use tf.where in 2.0, which has the same broadcast rule as np.where
Train on 73196 samples, validate on 36052 samples
WARNING:tensorflow:From /usr/local/lib/python3.6/dist-packages/keras/callback
s.py:1122: The name tf.summary.merge_all is deprecated. Please use tf.compat.
v1.summary.merge_all instead.

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

Epoch 1/15
73196/73196 [==============================] - 214s 3ms/step - loss: 1.4379 -
auroc: 0.5169 - val_loss: 0.9320 - val_auroc: 0.5615

Epoch 00001: val_auroc improved from -inf to 0.56152, saving model to model_
1.h5
Epoch 2/15
73196/73196 [==============================] - 208s 3ms/step - loss: 0.8051 -
auroc: 0.5444 - val_loss: 0.7111 - val_auroc: 0.6498

Epoch 00002: val_auroc improved from 0.56152 to 0.64984, saving model to mode
l_1.h5
Epoch 3/15
73196/73196 [==============================] - 207s 3ms/step - loss: 0.6565 -
auroc: 0.6042 - val_loss: 0.6155 - val_auroc: 0.6993

Epoch 00003: val_auroc improved from 0.64984 to 0.69925, saving model to mode
l_1.h5
Epoch 4/15
73196/73196 [==============================] - 206s 3ms/step - loss: 0.5627 -
auroc: 0.7139 - val_loss: 0.5490 - val_auroc: 0.7375

Epoch 00004: val_auroc improved from 0.69925 to 0.73749, saving model to mode
l_1.h5
Epoch 5/15
73196/73196 [==============================] - 206s 3ms/step - loss: 0.5035 -
auroc: 0.7435 - val_loss: 0.4988 - val_auroc: 0.7507

Epoch 00005: val_auroc improved from 0.73749 to 0.75065, saving model to mode
l_1.h5
Epoch 6/15
73196/73196 [==============================] - 206s 3ms/step - loss: 0.4690 -
auroc: 0.7530 - val_loss: 0.4607 - val_auroc: 0.7544

Epoch 00006: val_auroc improved from 0.75065 to 0.75439, saving model to mode
l_1.h5
Epoch 7/15
73196/73196 [==============================] - 205s 3ms/step - loss: 0.4491 -
auroc: 0.7555 - val_loss: 0.4509 - val_auroc: 0.7498

Epoch 00007: val_auroc did not improve from 0.75439
Epoch 8/15
```

```
73196/73196 [==============================] - 206s 3ms/step - loss: 0.4351 -
auroc: 0.7606 - val_loss: 0.4451 - val_auroc: 0.7451

Epoch 00008: val_auroc did not improve from 0.75439
Epoch 9/15
73196/73196 [==============================] - 206s 3ms/step - loss: 0.4222 -
auroc: 0.7640 - val_loss: 0.4266 - val_auroc: 0.7545

Epoch 00009: val_auroc improved from 0.75439 to 0.75454, saving model to mode
l_1.h5
Epoch 10/15
73196/73196 [==============================] - 207s 3ms/step - loss: 0.4165 -
auroc: 0.7635 - val_loss: 0.4146 - val_auroc: 0.7530

Epoch 00010: val_auroc did not improve from 0.75454
Epoch 11/15
73196/73196 [==============================] - 206s 3ms/step - loss: 0.4118 -
auroc: 0.7670 - val_loss: 0.4174 - val_auroc: 0.7529

Epoch 00011: val_auroc did not improve from 0.75454
Epoch 12/15
73196/73196 [==============================] - 207s 3ms/step - loss: 0.4078 -
auroc: 0.7666 - val_loss: 0.4150 - val_auroc: 0.7542

Epoch 00012: val_auroc did not improve from 0.75454
Epoch 13/15
73196/73196 [==============================] - 207s 3ms/step - loss: 0.4042 -
auroc: 0.7688 - val_loss: 0.4281 - val_auroc: 0.7536

Epoch 00013: val_auroc did not improve from 0.75454
Epoch 14/15
73196/73196 [==============================] - 206s 3ms/step - loss: 0.4039 -
auroc: 0.7675 - val_loss: 0.4156 - val_auroc: 0.7507

Epoch 00014: val_auroc did not improve from 0.75454
Epoch 15/15
73196/73196 [==============================] - 206s 3ms/step - loss: 0.4032 -
auroc: 0.7727 - val_loss: 0.4087 - val_auroc: 0.7534

Epoch 00015: val_auroc did not improve from 0.75454
```

In [0]:
```
plt.plot(history_1.history['auroc'], 'r')
plt.plot(history_1.history['val_auroc'], 'b')
plt.legend({'Train AUC': 'r', 'Test AUC':'b'})
plt.show()
```



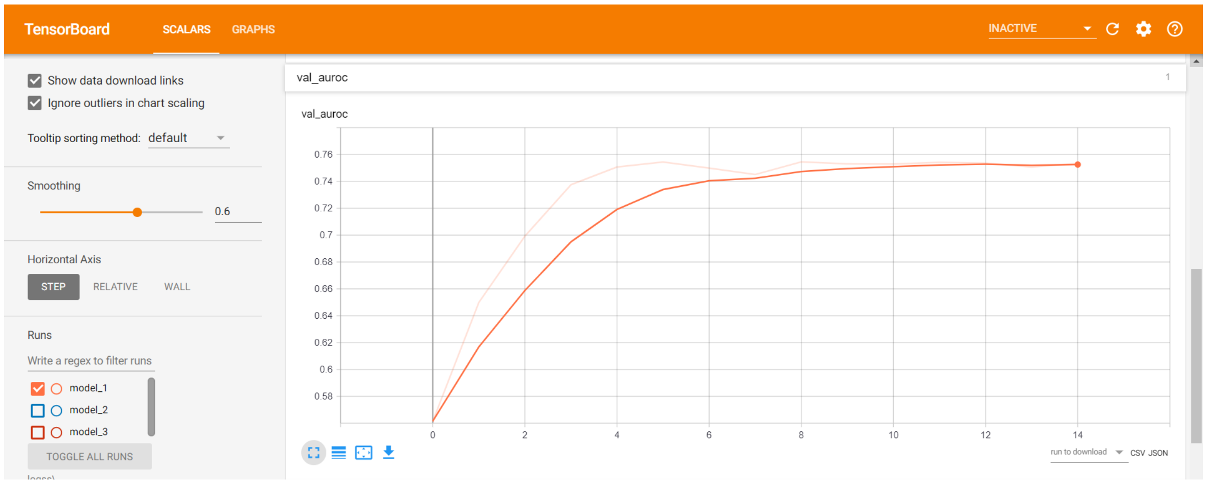
In [0]:
```
plt.plot(history_1.history['loss'], 'r')
plt.plot(history_1.history['val_loss'], 'b')
plt.legend({'Train Loss': 'r', 'Test Loss':'b'})
plt.show()
```



In [0]:

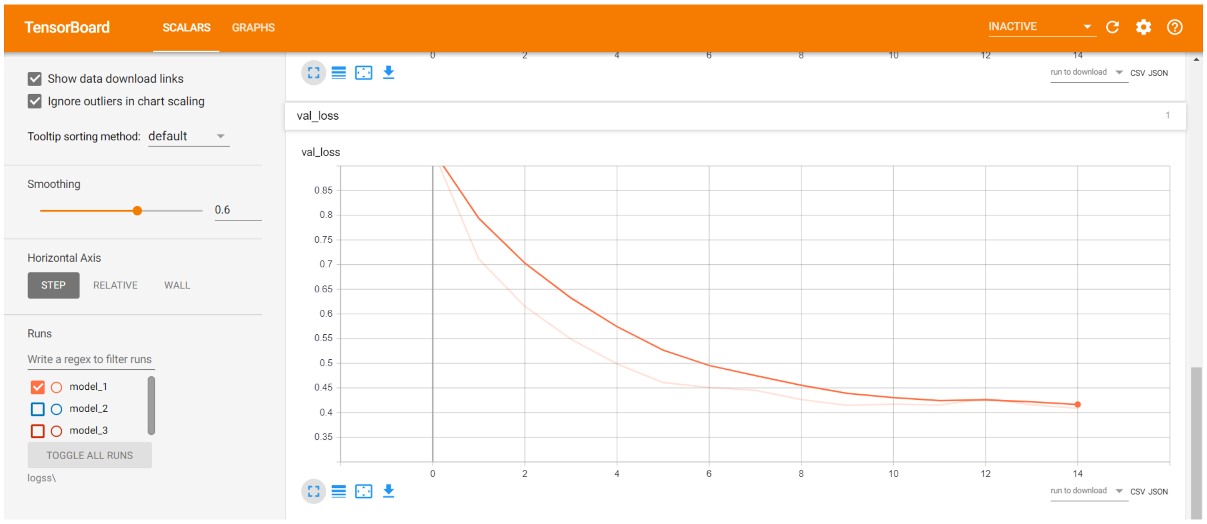
In [1]:
```
from IPython.display import Image
Image("model1auc.png")
```

Out[1]:



In [2]:
```
from IPython.display import Image
Image("model1loss.png")
```

Out[2]:



I'm able to get 0.75+ for Model-1 itself, lets see if we can improve for other models

In [0]:

In [0]:

In [0]:

In [0]:

In [0]:

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 analysi
s on the Idf values and based on those values choose the low and high threshold val
ue. Because very frequent words and very very rare words don't give much informatio
n. (you can plot a box plots and take only the idf scores within IQR range and corr
esponding words)

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

In [0]:

```
# importing required libraries
import warnings
warnings.filterwarnings("ignore")
import pandas as pd
import numpy as np
from keras.layers import Input, Embedding, LSTM, Dropout, BatchNormalization,
Dense, concatenate, Flatten, Conv1D, MaxPool1D, LeakyReLU, ELU, SpatialDropout
1D, MaxPooling1D, GlobalAveragePooling1D, GlobalMaxPooling1D
from keras.preprocessing.text import Tokenizer, one_hot
from keras.preprocessing.sequence import pad_sequences
from keras.models import Model, load_model
from keras import regularizers
from keras.optimizers import *
from keras.callbacks import ModelCheckpoint, EarlyStopping, TensorBoard, Reduc
eLROnPlateau
from sklearn.feature_extraction.text import TfidfVectorizer, CountVectorizer
from sklearn.metrics import roc_auc_score
import tensorflow as tf
import matplotlib.pyplot as plt
%matplotlib inline
import re
from tqdm import tqdm
from sklearn.preprocessing import LabelEncoder
import seaborn as sns
import pickle
```

```
Using TensorFlow backend.
```

In [0]:
```
%matplotlib inline
import warnings
warnings.filterwarnings("ignore")

import sqlite3
import pandas as pd
import numpy as np
import nltk
import string
import matplotlib.pyplot as plt
import seaborn as sns
from sklearn.feature_extraction.text import TfidfTransformer
from sklearn.feature_extraction.text import TfidfVectorizer

from sklearn.feature_extraction.text import CountVectorizer
from sklearn.metrics import confusion_matrix
from sklearn import metrics
from sklearn.metrics import roc_curve, auc
from nltk.stem.porter import PorterStemmer

import re
# Tutorial about Python regular expressions: https://pymotw.com/2/re/
import string
from nltk.corpus import stopwords
from nltk.stem import PorterStemmer
from nltk.stem.wordnet import WordNetLemmatizer

from gensim.models import Word2Vec
from gensim.models import KeyedVectors
import pickle

from tqdm import tqdm_notebook as tqdm1
from tqdm import tqdm
import time
import os

from plotly import plotly
import plotly.offline as offline
import plotly.graph_objs as go
offline.init_notebook_mode()
from collections import Counter

from sklearn.model_selection import train_test_split
```

In [0]:
```
# project_data = pd.read_csv('preprocessed_data.csv')
# project_data.head()
```

In [0]:
```
# project_data = pd.read_csv('train_data.csv', nrows=50000)
project_data = pd.read_csv('train_data.csv', nrows=60000)
resource_data = pd.read_csv('resources.csv')
```

In [0]:
```
print("Number of data points in train data", project_data.shape)
print('-'*50)
print("The attributes of data :", project_data.columns.values)
```

```
Number of data points in train data (60000, 17)
--------------------------------------------------
The attributes of data : ['Unnamed: 0' 'id' 'teacher_id' 'teacher_prefix' 'sc
hool_state'
 'project_submitted_datetime' 'project_grade_category'
 'project_subject_categories' 'project_subject_subcategories'
 'project_title' 'project_essay_1' 'project_essay_2' 'project_essay_3'
 'project_essay_4' 'project_resource_summary'
 'teacher_number_of_previously_posted_projects' 'project_is_approved']
```

Text preprocessing(1)

In [0]:
```
catogories = list(project_data['project_subject_categories'].values)
# remove special characters from list of strings python: https://stackoverflo
w.com/a/47301924/4084039

# https://www.geeksforgeeks.org/removing-stop-words-nltk-python/
# https://stackoverflow.com/questions/23669024/how-to-strip-a-specific-word-fr
om-a-string
# https://stackoverflow.com/questions/8270092/remove-all-whitespace-in-a-strin
g-in-python
cat_list = []
for i in catogories:
    temp = ""
    # consider we have text like this "Math & Science, Warmth, Care & Hunger"
    for j in i.split(','): # it will split it in three parts ["Math & Scienc
e", "Warmth", "Care & Hunger"]
        if 'The' in j.split(): # this will split each of the catogory based on
space "Math & Science"=> "Math","&", "Science"
            j=j.replace('The','') # if we have the words "The" we are going to
replace it with ''(i.e removing 'The')
        j = j.replace(' ','') # we are placeing all the ' '(space) with ''(emp
ty) ex:"Math & Science"=>"Math&Science"
        temp+=j.strip()+" " #" abc ".strip() will return "abc", remove the tra
iling spaces
        temp = temp.replace('&','_') # we are replacing the & value into
    cat_list.append(temp.strip())
```

In [0]:
```
project_data['clean_categories'] = cat_list
project_data.drop(['project_subject_categories'], axis=1, inplace=True)
project_data.head(5)
```

Out[0]:

	Unnamed: 0	id	teacher_id	teacher_prefix	school_state	project_
0	160221	p253737	c90749f5d961ff158d4b4d1e7dc665fc	Mrs.	IN	
1	140945	p258326	897464ce9ddc600bced1151f324dd63a	Mr.	FL	
2	21895	p182444	3465aaf82da834c0582ebd0ef8040ca0	Ms.	AZ	
3	45	p246581	f3cb9bffbba169bef1a77b243e620b60	Mrs.	KY	
4	172407	p104768	be1f7507a41f8479dc06f047086a39ec	Mrs.	TX	

In [0]:
```
# count of all the words in corpus python: https://stackoverflow.com/a/2289859
5/4084039
from collections import Counter
my_counter = Counter()
for word in project_data['clean_categories'].values:
    my_counter.update(word.split())
my_counter
```

Out[0]:
```
Counter({'Literacy_Language': 28785,
         'History_Civics': 3229,
         'Health_Sports': 7854,
         'Math_Science': 22654,
         'SpecialNeeds': 7431,
         'AppliedLearning': 6662,
         'Music_Arts': 5658,
         'Warmth': 781,
         'Care_Hunger': 781})
```

In [0]:
```
# dict sort by value python: https://stackoverflow.com/a/613218/4084039
cat_dict = dict(my_counter)
sorted_cat_dict = dict(sorted(cat_dict.items(), key=lambda kv: kv[1]))


# ind = np.arange(len(sorted_cat_dict))
# plt.figure(figsize=(20,5))
# p1 = plt.bar(ind, list(sorted_cat_dict.values()))

# plt.ylabel('Projects')
# plt.title('% of projects aproved category wise')
# plt.xticks(ind, list(sorted_cat_dict.keys()))
# plt.show()
# print(sorted_cat_dict)
```

In [0]:
```
sub_catogories = list(project_data['project_subject_subcategories'].values)
# remove special characters from list of strings python: https://stackoverflo
w.com/a/47301924/4084039

# https://www.geeksforgeeks.org/removing-stop-words-nltk-python/
# https://stackoverflow.com/questions/23669024/how-to-strip-a-specific-word-fr
om-a-string
# https://stackoverflow.com/questions/8270092/remove-all-whitespace-in-a-strin
g-in-python

sub_cat_list = []
for i in sub_catogories:
    temp = ""
    # consider we have text like this "Math & Science, Warmth, Care & Hunger"
    for j in i.split(','): # it will split it in three parts ["Math & Scienc
e", "Warmth", "Care & Hunger"]
        if 'The' in j.split(): # this will split each of the catogory based on
space "Math & Science"=> "Math","&", "Science"
            j=j.replace('The','') # if we have the words "The" we are going to
replace it with ''(i.e removing 'The')
        j = j.replace(' ','') # we are placeing all the ' '(space) with ''(emp
ty) ex:"Math & Science"=>"Math&Science"
        temp +=j.strip()+" "#" abc ".strip() will return "abc", remove the tra
iling spaces
        temp = temp.replace('&','_')
    sub_cat_list.append(temp.strip())
```

In [0]:
```
project_data['clean_subcategories'] = sub_cat_list
project_data.drop(['project_subject_subcategories'], axis=1, inplace=True)
project_data.head(2)
```

Out[0]:

	Unnamed: 0	id	teacher_id	teacher_prefix	school_state	project_
0	160221	p253737	c90749f5d961ff158d4b4d1e7dc665fc	Mrs.	IN	
1	140945	p258326	897464ce9ddc600bced1151f324dd63a	Mr.	FL	

In [0]:
```
# count of all the words in corpus python: https://stackoverflow.com/a/2289859
5/4084039
from collections import Counter
my_counter = Counter()
for word in project_data['clean_subcategories'].values:
    my_counter.update(word.split())
```

In [0]:
```
# dict sort by value python: https://stackoverflow.com/a/613218/4084039
sub_cat_dict = dict(my_counter)
sorted_sub_cat_dict = dict(sorted(sub_cat_dict.items(), key=lambda kv: kv[1]))


# ind = np.arange(len(sorted_sub_cat_dict))
# plt.figure(figsize=(20,5))
# p1 = plt.bar(ind, list(sorted_sub_cat_dict.values()))

# plt.ylabel('Projects')
# plt.title('% of projects aproved state wise')
# plt.xticks(ind, list(sorted_sub_cat_dict.keys()))
# plt.show()
```

In [0]:
```
# merge two column text dataframe:
project_data["essay"] = project_data["project_essay_1"].map(str) +\
                        project_data["project_essay_2"].map(str) + \
                        project_data["project_essay_3"].map(str) + \
                        project_data["project_essay_4"].map(str)
```

In [0]:
```
# https://stackoverflow.com/questions/22407798/how-to-reset-a-dataframes-indexes-for-all-groups-in-one-step
price_data = resource_data.groupby('id').agg({'price':'sum', 'quantity':'sum'}).reset_index()
price_data.head(2)
```

Out[0]:

	id	price	quantity
0	p000001	459.56	7
1	p000002	515.89	21

In [0]:
```
# join two dataframes in python:
project_data = pd.merge(project_data, price_data, on='id', how='left')
```

In [0]:
```
#presence of the numerical digits in a strings with numeric : https://stackove
rflow.com/a/19859308/8089731
def hasNumbers(inputString):
    return any(i.isdigit() for i in inputString)
p1 = project_data[['id','project_resource_summary']]
p1 = pd.DataFrame(data=p1)
p1.columns = ['id','digits_in_summary']
p1['digits_in_summary'] =  p1['digits_in_summary'].map(hasNumbers)
# https://stackoverflow.com/a/17383325/8089731
p1['digits_in_summary'] = p1['digits_in_summary'].astype(int)
project_data = pd.merge(project_data, p1, on='id', how='left')
project_data.head(5)
```

Out[0]:

	Unnamed: 0	id	teacher_id	teacher_prefix	school_state	project_
0	160221	p253737	c90749f5d961ff158d4b4d1e7dc665fc	Mrs.	IN	
1	140945	p258326	897464ce9ddc600bced1151f324dd63a	Mr.	FL	
2	21895	p182444	3465aaf82da834c0582ebd0ef8040ca0	Ms.	AZ	
3	45	p246581	f3cb9bffbba169bef1a77b243e620b60	Mrs.	KY	
4	172407	p104768	be1f7507a41f8479dc06f047086a39ec	Mrs.	TX	

5 rows × 21 columns

Text preprocessing(2)

In [0]:
```
# https://stackoverflow.com/a/47091490/4084039
import re

def decontracted(phrase):
    # specific
    phrase = re.sub(r"won't", "will not", phrase)
    phrase = re.sub(r"can\'t", "can not", phrase)

    # general
    phrase = re.sub(r"n\'t", " not", phrase)
    phrase = re.sub(r"\'re", " are", phrase)
    phrase = re.sub(r"\'s", " is", phrase)
    phrase = re.sub(r"\'d", " would", phrase)
    phrase = re.sub(r"\'ll", " will", phrase)
    phrase = re.sub(r"\'t", " not", phrase)
    phrase = re.sub(r"\'ve", " have", phrase)
    phrase = re.sub(r"\'m", " am", phrase)
    return phrase
```

In [0]:
```
# https://gist.github.com/sebleier/554280
# we are removing the words from the stop words list: 'no', 'nor', 'not'
stopwords= ['i', 'me', 'my', 'myself', 'we', 'our', 'ours', 'ourselves', 'you'
, "you're", "you've",\
            "you'll", "you'd", 'your', 'yours', 'yourself', 'yourselves', 'he'
, 'him', 'his', 'himself', \
            'she', "she's", 'her', 'hers', 'herself', 'it', "it's", 'its', 'it
self', 'they', 'them', 'their',\
            'theirs', 'themselves', 'what', 'which', 'who', 'whom', 'this', 't
hat', "that'll", 'these', 'those', \
            'am', 'is', 'are', 'was', 'were', 'be', 'been', 'being', 'have',
'has', 'had', 'having', 'do', 'does', \
            'did', 'doing', 'a', 'an', 'the', 'and', 'but', 'if', 'or', 'becau
se', 'as', 'until', 'while', 'of', \
            'at', 'by', 'for', 'with', 'about', 'against', 'between', 'into',
'through', 'during', 'before', 'after',\
            'above', 'below', 'to', 'from', 'up', 'down', 'in', 'out', 'on',
'off', 'over', 'under', 'again', 'further',\
            'then', 'once', 'here', 'there', 'when', 'where', 'why', 'how', 'a
ll', 'any', 'both', 'each', 'few', 'more',\
            'most', 'other', 'some', 'such', 'only', 'own', 'same', 'so', 'tha
n', 'too', 'very', \
            's', 't', 'can', 'will', 'just', 'don', "don't", 'should', "shoul
d've", 'now', 'd', 'll', 'm', 'o', 're', \
            've', 'y', 'ain', 'aren', "aren't", 'couldn', "couldn't", 'didn',
"didn't", 'doesn', "doesn't", 'hadn',\
            "hadn't", 'hasn', "hasn't", 'haven', "haven't", 'isn', "isn't", 'm
a', 'mightn', "mightn't", 'mustn',\
            "mustn't", 'needn', "needn't", 'shan', "shan't", 'shouldn', "shoul
dn't", 'wasn', "wasn't", 'weren', "weren't", \
            'won', "won't", 'wouldn', "wouldn't"]
```

In [0]:
```
# Combining all the above statemennts
from tqdm import tqdm
preprocessed_essays = []
# tqdm is for printing the status bar
for sentance in tqdm1(project_data['essay'].values):
    sent = decontracted(sentance)
    sent = sent.replace('\\r', ' ')
    sent = sent.replace('\\"', ' ')
    sent = sent.replace('\\n', ' ')
    sent = re.sub('[^A-Za-z0-9]+', ' ', sent)
    sent = re.sub('nannan', '', sent)
    # https://gist.github.com/sebleier/554280
    sent = ' '.join(e for e in sent.split() if e not in stopwords)
    preprocessed_essays.append(sent.lower().strip())
```

In [0]:
```
from tqdm import tqdm
preprocessed_titles = []
# tqdm is for printing the status bar
for title in tqdm1(project_data['project_title'].values):
    _title = decontracted(title)
    _title = _title.replace('\\r', ' ')
    _title = _title.replace('\\"', ' ')
    _title = _title.replace('\\n', ' ')
    _title = re.sub('[^A-Za-z0-9]+', ' ', _title)
    # https://gist.github.com/sebleier/554280
    _title = ' '.join(e for e in _title.split() if e not in stopwords)
    preprocessed_titles.append(_title.lower().strip())
```

In [0]:
```
preprocessed_titles[1000]
```

Out[0]:
```
'sailing into super 4th grade year'
```

In [0]:
```
project_grade_catogories = list(project_data['project_grade_category'].values)
# remove special characters from list of strings python: https://stackoverflo
w.com/a/47301924/4084039

# https://www.geeksforgeeks.org/removing-stop-words-nltk-python/
# https://stackoverflow.com/questions/23669024/how-to-strip-a-specific-word-fr
om-a-string
# https://stackoverflow.com/questions/8270092/remove-all-whitespace-in-a-strin
g-in-python

project_grade_cat_list = []
for i in tqdm1(project_grade_catogories):
    temp = ""
    # consider we have text like this "Math & Science, Warmth, Care & Hunger"
    for j in i.split(','): # it will split it in three parts ["Math & Scienc
e", "Warmth", "Care & Hunger"]
        if 'The' in j.split(): # this will split each of the catogory based on
space "Math & Science"=> "Math","&", "Science"
            j=j.replace('The','') # if we have the words "The" we are going to
replace it with ''(i.e removing 'The')
        j = j.replace(' ','') # we are placeing all the ' '(space) with ''(emp
ty) ex:"Math & Science"=>"Math&Science"
        temp +=j.strip()+" "#" abc ".strip() will return "abc", remove the tra
iling spaces
        temp = temp.replace('&','_')
    project_grade_cat_list.append(temp.strip())
```

In [0]:
```
project_data['clean_project_grade_category'] = project_grade_cat_list
project_data.drop(['project_grade_category'], axis=1, inplace=True)
project_data.head(2)
```

Out[0]:

	Unnamed: 0	id	teacher_id	teacher_prefix	school_state	project_
0	160221	p253737	c90749f5d961ff158d4b4d1e7dc665fc	Mrs.	IN	
1	140945	p258326	897464ce9ddc600bced1151f324dd63a	Mr.	FL	

2 rows × 21 columns

In [0]:
```
project_data.drop(['project_essay_1','project_essay_2','project_essay_3','proj
ect_essay_4'], axis=1, inplace=True)
project_data.head(2)
```

Out[0]:

	Unnamed: 0	id	teacher_id	teacher_prefix	school_state	project_
0	160221	p253737	c90749f5d961ff158d4b4d1e7dc665fc	Mrs.	IN	
1	140945	p258326	897464ce9ddc600bced1151f324dd63a	Mr.	FL	

In [0]:
```
#Replacing Nan's with maximum occured value: https://stackoverflow.com/a/51053
916/8089731
project_data['teacher_prefix'].value_counts().argmax()
project_data.fillna(value=project_data['teacher_prefix'].value_counts().argmax
(),axis=1,inplace=True)
```

In [0]:
```
project_data['preprocessed_essays'] = preprocessed_essays
project_data['preprocessed_titles'] = preprocessed_titles
```

In [0]:
```
X_train, X_test, y_train, y_test = train_test_split(project_data,project_data[
'project_is_approved'], test_size=0.33, stratify = project_data['project_is_ap
proved'])
# X_train, X_cv, y_train, y_cv = train_test_split(X_train, y_train, test_size=
0.33, stratify=y_train)

X_train.drop(['project_is_approved'], axis=1, inplace=True)
X_test.drop(['project_is_approved'], axis=1, inplace=True)
# X_cv.drop(['project_is_approved'], axis=1, inplace=True)
print(X_train.shape)
print(X_test.shape)
```

```
(40200, 18)
(19800, 18)
```

In [0]:
```
X_train
```

Out[0]:

	Unnamed: 0	id	teacher_id	teacher_prefix	school_state	pr
53990	73517	p081693	ba962cbafe31df98f6a222190ec180e1	Mr.	LA	
26175	68310	p169729	9c42729a83ab872a10224912a57074f1	Ms.	NJ	
28017	116471	p206399	eae27d7cfb841470d31a0caea6b3b288	Mrs.	OR	
45842	13608	p123373	50514d830baa865acd4cc57e23b344a5	Mrs.	GA	
6603	8248	p178167	f29ea1c7c7fbabe9dfc504cc55f7b26a	Mrs.	FL	
57387	80232	p021799	2b376452e1ef640d5ecde27641a3a7b4	Mrs.	KY	
16576	132341	p014368	493838a03fc6cda0229db2c9fb60597c	Mrs.	OK	
17698	104159	p248216	9db49183c0fdfba52fe5fb4d26824411	Mrs.	WI	
55821	166737	p199098	3aa717ec73f2e5b9b0bcb2c92c6dc62a	Ms.	GA	
42662	14486	p000320	c4a43211a120ec6bdf97555e2b4d6b17	Mrs.	MO	
22446	63973	p060210	6ea51fd46d48cfa1d3296f41fd6a4d4e	Ms.	AZ	
29217	66005	p035783	cbea53168bc28e8dcdc89e790d925cd3	Ms.	CA	

	Unnamed: 0	id	teacher_id	teacher_prefix	school_state	pr
32017	82902	p023753	1a042441d106790eb6be98bcd50c216f	Mrs.	PA	
2563	108646	p214882	c18f7d1783f94e595bda4451ee93a85e	Mr.	NC	
14083	127144	p050423	a6d785ad6fc63496eb42d175cbc5648e	Mrs.	NC	
5253	1119	p193991	be467845279126e1a5e1bff20e3f5ed1	Ms.	IL	
44197	180740	p015759	a2d7593291e5a8e31a841687a84281a9	Ms.	FL	
40203	153880	p077217	143edf787ce5aeb453868481f69fc71e	Mrs.	GA	
1583	155026	p010948	18a27ae523affb88b8e18fa389095a3f	Ms.	NC	
45386	149382	p021621	73cb3679eb11b9eda022b743341e31d9	Ms.	NY	
16525	104434	p097413	34ff20e99f8547311b7e58c80a600462	Mrs.	UT	
28032	81573	p095064	654020cb2af6e4fdfd1251387cf09d3e	Mrs.	MS	
1653	180225	p214501	732fc1bcaaae0e78f9de7ae2383b8cde	Mrs.	DE	
11563	67830	p230813	633d2678d047fa96568d1871084fe300	Mr.	IL	
55491	172354	p247965	f5fb3669a01bbc49a2e6b0e8cbae701b	Mrs.	NY	

	Unnamed: 0	id	teacher_id	teacher_prefix	school_state	pr
35773	79062	p077639	fe36c7e2d6e92aff697b97cc200b1ef2	Mrs.	AZ	
28219	105744	p161692	0079643d5a79ee7e6391a4ef42d49890	Mrs.	OH	
32970	94565	p134061	615a8e361894d6153ea00bf0b48d1581	Mr.	AZ	
3307	145118	p065447	84071283779dc10015dc7cd3055c936a	Mrs.	OK	
12542	21421	p138960	0959787558b987d9d8f19005a8feff93	Ms.	NJ	
...	...	...	...	...	...	
38195	51739	p226618	2d6bbbc65de2d690a64233fb2a5bd37b	Mrs.	OR	
55799	161735	p101257	3f8becf9819ec047b1e4a694988bcb61	Mrs.	OK	
25624	5210	p237385	a9c72e85d1fe0b347685727a4ef63dfc	Ms.	IA	
34260	152986	p092620	f0776e22056b14db338d7587606b9026	Mrs.	TN	
16118	139696	p131898	6fefa33c27c375ddb39a7fafc5e6df76	Mrs.	NC	
55492	128006	p252743	128db549aecbebca131f1f7d7c4f593f	Mrs.	CA	

	Unnamed: 0	id	teacher_id	teacher_prefix	school_state	pr
22464	154719	p170499	c64de8c19e38a0064a862a2bde5cf06c	Ms.	NY	
52971	50508	p052668	551ba584eb4ca683f003c523b84d3554	Ms.	NY	
58977	29083	p188716	f1ee7a5ea0c398c21c917668b083208a	Mrs.	NC	
1236	112871	p106819	fc0b0447c9f652605fd3f1a559f24436	Mrs.	MN	
3667	16911	p196780	b785332887095f0e4afcd30d4fa4471e	Mrs.	MN	
25577	136499	p217961	cad82511a2a07f2f190b4f2e1ee30b4d	Mrs.	SC	
28226	61498	p005886	18f8f6f2b8b2208339780a34dae71e57	Ms.	NJ	
30580	168646	p116383	6f6e951e435aa9dc966091945414bcc4	Ms.	NC	
37061	10845	p004401	4804248f73b437503eb78a2b2c074b4f	Mrs.	VA	
37250	150809	p211100	c87ab430581a9054a00742b29e4246b0	Ms.	WI	
47707	11235	p132518	8b2dce26ca10fe895bac9a969f595e15	Mrs.	OK	

	Unnamed: 0	id	teacher_id	teacher_prefix	school_state	pr
4818	150619	p103892	c635058b720a0cbeb9353ed60b0efddf	Ms.	NJ	
12264	173706	p220892	fbe978624c4a37c7ddd99a4c99c47334	Mrs.	SC	
7827	12491	p152778	42e63608cf0f42267cf06a0aa9db6644	Ms.	WI	
38415	122938	p113316	22de4717029313a7dd088e3292f3fe5c	Ms.	WA	
47238	13400	p114478	cc8f66588e8706550ec6a0ef8e89851d	Mrs.	PA	
19222	137516	p147331	f3b230a08ab24e16b37f38f458a78eb4	Ms.	MI	
33441	123259	p244352	3c05958502ef31d231b848af362ff25e	Mrs.	UT	
20100	90756	p180085	8090e7125d2473b7e34181f4fc89ce6a	Ms.	FL	
47847	62185	p096192	3d265d0eb9d440448656dbdb858b8ec2	Ms.	NY	
54512	133694	p197531	7c6d20c34157c0ce82f90857c99cde9d	Mrs.	MA	
45649	5653	p238937	b9fc2478661fe53047d5b4e6c1fa4abb	Mrs.	WY	
41507	115821	p193296	d9f01f40cb073653ba454c36e403e9ca	Ms.	CA	

	Unnamed: 0	id	teacher_id	teacher_prefix	school_state	pr
54838	55663	p074856	e8558a341b6319a1f6cfbbd7ca0750b1	Mrs.	OK	

40200 rows × 18 columns

In [0]:
```
import dill
# dill.dump_session('notebook_envall.db')
dill.load_session('notebook_envall.db')
```

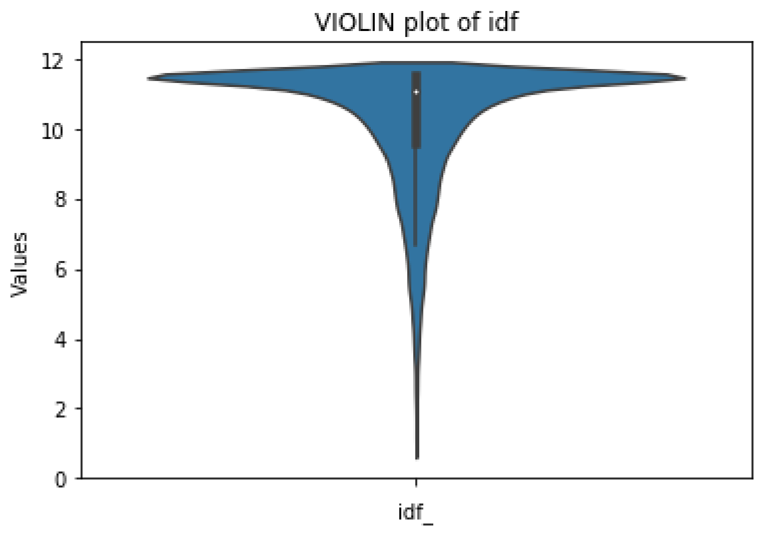
```
Using TensorFlow backend.
```

In [0]:
```
from numpy import array
from numpy import asarray
from numpy import zeros
from keras.preprocessing.text import Tokenizer
from keras.preprocessing.sequence import pad_sequences
from keras.models import Sequential
from keras.layers import Dense
from keras.layers import Flatten
from keras.layers import Embedding
```

In [0]:
```
tfidf = TfidfVectorizer()
_tfidf = tfidf.fit_transform(X_train["essay"])
_dict = dict(zip(tfidf.get_feature_names(),list(tfidf.idf_)))
tfidf_df = pd.DataFrame(list(_dict.items()), columns=['Words', 'Values'])
tfidf_df = tfidf_df.sort_values(by ='Values' )
```

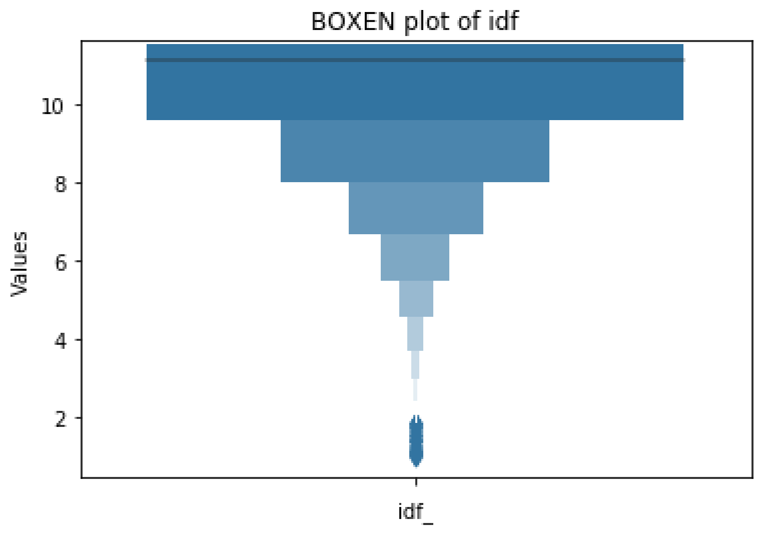
In [0]:
```
sns.violinplot(x = "Values",data=tfidf_df,orient="v")
plt.xlabel("idf_")
plt.title("VIOLIN plot of idf")
```

Out[0]:
```
Text(0.5, 1.0, 'VIOLIN plot of idf')
```



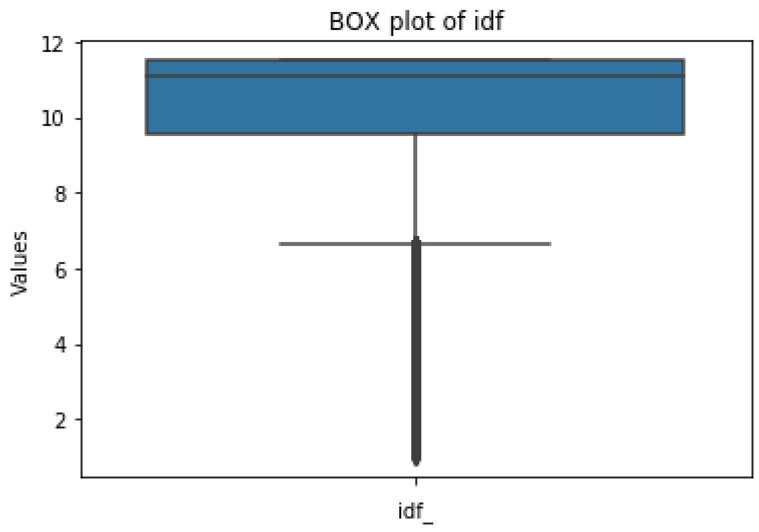
In [0]:
```
sns.boxenplot(x = "Values",data=tfidf_df,orient="v")
plt.xlabel("idf_")
plt.title("BOXEN plot of idf")
```

Out[0]:
```
Text(0.5, 1.0, 'BOXEN plot of idf')
```



In [0]:
```
sns.boxplot(x = "Values",data=tfidf_df,orient="v")
plt.xlabel("idf_")
plt.title("BOX plot of idf")
```

Out[0]:
```
Text(0.5, 1.0, 'BOX plot of idf')
```



In [0]:
```
print("0 - 100 Quantiles:")
print(np.percentile(tfidf_df['Values'],np.arange(0, 100, 25)))
```

```
0 - 100 Quantiles:
[ 1.0001093   9.56185239 11.10229743 11.50776253]
```

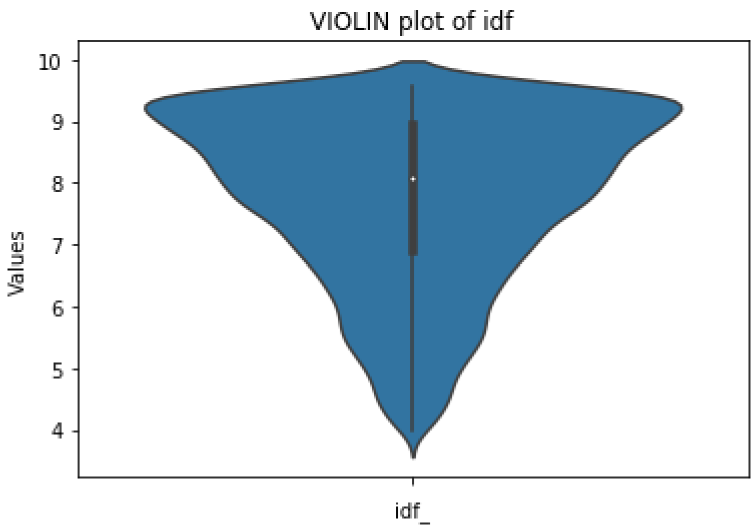
In [0]:
```
print("1th Percentile")
print(np.percentile(tfidf_df['Values'],1))
print("\n27th Percentiles")
print(np.percentile(tfidf_df['Values'],27))
```

```
1th Percentile
3.983908779165195

27th Percentiles
9.716003065714995
```

In [0]:
```
tfidf_fil = tfidf_df[(tfidf_df["Values"] >=  np.percentile(tfidf_df['Values'],
1)) & (tfidf_df["Values"] <=  np.percentile(tfidf_df['Values'],25))]
```

In [0]:
```
sns.violinplot(x = "Values",data=tfidf_fil,orient="v")
plt.xlabel("idf_")
plt.title("VIOLIN plot of idf")
```

Out[0]:
```
Text(0.5, 1.0, 'VIOLIN plot of idf')
```



In [0]:
```
fil_words = tfidf_fil["Words"].tolist()
```

In [0]:
```
tokenizer = Tokenizer()
tokenizer.fit_on_texts(fil_words)
seq_train = tokenizer.texts_to_sequences(X_train["essay"])
seq_test = tokenizer.texts_to_sequences(X_test["essay"])
```

In [0]:
```
vocab_size = len(tokenizer.word_index) + 1
```

In [0]:
```
padded_train = pad_sequences(seq_train,maxlen=800,padding='post', truncating=
'post')
padded_test = pad_sequences(seq_test, maxlen=800,padding='post', truncating='p
ost')
```

In [0]:
```
pickle_in = open("/content/drive/My Drive/AAIC/Assignments/glove_vectors","rb"
)
glove_words = pickle.load(pickle_in)
```

In [0]:
```
embedding_matrix = zeros((vocab_size, 300))
for word, i in tokenizer.word_index.items():
    embedding_vector = glove_words.get(word)
    if embedding_vector is not None:
        embedding_matrix[i] = embedding_vector
```

In [0]:
```
# input_text = Input(shape=(800,),name="input_text")
embedding_layer = Embedding(vocab_size,300,weights=[embedding_matrix],input_le
ngth=800,trainable=False)
input_text = Input(shape=(800,),name="input_text")
x = embedding_layer(input_text)
x = LSTM(128,recurrent_dropout=0.5,kernel_regularizer=regularizers.l2(0.001),r
eturn_sequences=True)(x)
flat_1 = Flatten()(x)
```

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

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

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

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

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

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

In [0]:
```
# Teacher Prefix
no_of_unique_prefix  = X_train["teacher_prefix"].nunique()
embed_size_prefix = int(min(np.ceil((no_of_unique_prefix)/2), 50 ))

input_prefix = Input(shape=(1,),name="teacher_prefix")
embed_prefix = Embedding(no_of_unique_prefix,embed_size_prefix,trainable=True)
(input_prefix)
flat_2 = Flatten()(embed_prefix)
lab_enc = LabelEncoder()
enc_prefix_train = lab_enc.fit_transform(X_train["teacher_prefix"])
X_test["teacher_prefix"] = X_test["teacher_prefix"].map(lambda s: ' ' if s not
in lab_enc.classes_ else s)
lab_enc.classes_ = np.append(lab_enc.classes_, ' ')
enc_prefix_test = lab_enc.transform(X_test["teacher_prefix"])
```

In [0]:
```
# School State
no_of_unique_state  = X_train["school_state"].nunique()
embed_size_state= int(min(np.ceil((no_of_unique_state)/2), 50 ))

input_state = Input(shape=(1,),name="school_prefix")
embed_state = Embedding(no_of_unique_state,embed_size_state,name="emb_state",t
rainable=True)(input_state)
flat_3 = Flatten()(embed_state)
enc_state_train = lab_enc.fit_transform(X_train["school_state"])
enc_state_test = lab_enc.transform(X_test["school_state"])
```

In [0]:
```
X_train
```

Out[0]:

	Unnamed: 0	id	teacher_id	teacher_prefix	school_state	p
61692	9233	p134729	2fba990f30bae627a30c12abc14b4ef8	Mrs.	LA	
101374	30334	p190416	b62d6e4ad22c5fcc2063d3eb99713627	Ms.	NY	
8708	89772	p149606	b642d19bb72bc81afe90784121562e92	Ms.	NM	
70173	6607	p009634	c751cdc864bdc94a7c73fbc8d81f6260	Ms.	LA	
90839	144635	p000962	413dd95145e3bc9940e70f7dcf1c6304	Mr.	AZ	
62631	78012	p212397	53ccd40b8bd6f268ab7a0c73be3c0128	Mrs.	TX	
41361	50382	p237295	50e93b8c452382361a4847f4bb3acf24	Mrs.	MI	
97600	28582	p136590	c0432ba8cc235436596dd1e9184ca59d	Mr.	CO	
67695	85668	p204009	c9fd190fc4673e37fdfd919816f8fb34	Ms.	CA	
62517	177330	p039526	aa96ef933643e81d26495f0cbaea40aa	Ms.	MI	
79924	49997	p092797	028c1a9b97dbe4382bde1768309029a6	Mrs.	IN	
3293	88593	p253595	32be6df3fae18951952a8d734e2c5282	Mrs.	IL	

	Unnamed: 0	id	teacher_id	teacher_prefix	school_state	p
20959	85447	p202376	9126ed70645e56668963f82f65049246	Mrs.	MI	
46118	144888	p065891	3733f7f569d0b1f5366929e50134d147	Ms.	NC	
52590	60373	p249054	7cdf8261f3995678e92f10ca3e9e4154	Ms.	CA	
53712	118959	p222662	0de9f14da9526236c3bfe68d74942961	Ms.	MO	
71562	180573	p119887	12095a3ab253dc5d76cd30a35a68d7f5	Mrs.	IN	
49044	97203	p054181	e5df9f1df68eaf72b513d0d87ca7fabf	Mrs.	NV	
17597	143412	p233096	ae3d1fed5e7d36ec6507720d2a6e19c8	Ms.	DC	
35975	155318	p032324	3e23eef919ed0d7eaebe5f7da7395353	Mrs.	LA	
11802	164921	p213956	7bd92da677bb1fab60eae4780fab4018	Ms.	NC	
7818	31126	p086960	6add52d24845c544880cb30c8ba861df	Mr.	OR	
71082	2906	p213225	2d6594ae664616d2649fc6513427e905	Mr.	CO	
53187	166431	p237910	88d99ccd29c97bf2e91d244f745652c5	Ms.	NC	

	Unnamed: 0	id	teacher_id	teacher_prefix	school_state	pı
87936	105035	p134626	1107c3baabc13d506f332fc1c92f43e2	Ms.	NE	
38345	123888	p054357	061456f27cf575c6ce9d2c4bdb761bc6	Ms.	MO	
12250	40928	p247256	76cdb75d01b2e87341bc40ed83534397	Ms.	CA	
24951	152108	p001117	abb220c60ae2b56e773a01652d5452b5	Mrs.	NY	
81247	44203	p078243	b1cb571b0125b72f4db62fd4fb5b9607	Ms.	SC	
39315	8277	p123224	d2510a557eb8c903d4aae58c0d9dabda	Ms.	UT	
...	...	...	...	...	...	
73516	166116	p208444	d4a9803a0281bc39880f3d593ef0983a	Ms.	NJ	
63010	8201	p160652	fc1769db9f1982d0c5f4317065090d00	Mrs.	MO	
44561	59556	p194709	84074b3aa2e7046e7fb465b6b510efeb	Mrs.	LA	
69035	5025	p042180	130182667468b0ec6ccfd106359f6143	Mrs.	CA	
37302	110982	p156758	5d0caac278c76b0b3fb8a9d4d413d495	Mrs.	OH	
35529	123949	p014992	5c65c01fac4e8b11b233db944c6dde78	Ms.	MO	

	Unnamed: 0	id	teacher_id	teacher_prefix	school_state	p
82657	35180	p207628	997a37e2e303548d637fbe924c26a5b2	Ms.	TX	
57159	26239	p029264	93e12c086635fccef958a765f1ef9d82	Ms.	IL	
24278	3987	p010222	9efea81af5c1e406f178754028df762d	Mrs.	CA	
44039	59264	p099155	6ad46801a6ac3ff111eb48a29a563102	Mrs.	OH	
29633	97774	p081868	7d71578de985d546bdc1db7425dbf534	Mrs.	OH	
32458	158351	p065941	5b7d16fec33d61c38b3684cbfa7c000f	Mr.	TX	
54308	15779	p081553	ee2e204481999bd839cea5fc3d293ec8	Mrs.	NY	
8929	181159	p228458	8c9234d6fd08bc6d548649698d385f87	Ms.	AZ	
79971	176465	p020345	a238b2ea3c69831edd224620b21c1fce	Mrs.	VA	
72750	169104	p120275	577a5a08365015d2e444ab832214c046	Ms.	NY	
82091	40001	p078763	f8fc4b35b8f87bc601c3cc0c662c5bbe	Mrs.	CO	
76005	59031	p180583	6256f3cb5c44aa6ab76fbf3b0b37327d	Ms.	NY	

	Unnamed: 0	id	teacher_id	teacher_prefix	school_state	p
71420	68777	p203595	dc56b5f1a30ba8f1ec4bdd8e397307f8	Mrs.	SC	
75662	165959	p066542	7709b9dbc2bea6ee0e52266a8df8559c	Ms.	MI	
103851	150825	p225662	20dc6dacfef7de81d46927325a7562e5	Ms.	LA	
19769	85663	p241125	1f0e2a8ba9e02b9ead4293ec2be83185	Mrs.	GA	
33667	138647	p091149	d599114df256ae1998b53b5b725e016c	Ms.	IL	
103201	17999	p177022	2e9925990a881fd36a09da73e7388365	Mrs.	OK	
54380	26631	p246832	a70a978e33eb178017623ce48202c808	Mrs.	MI	
53764	150418	p228011	144a533f809ba503b9b9774d7b16cbeb	Mrs.	GA	
19038	157043	p108749	d70065ac07d35035a7a42bd14ee10413	Mrs.	UT	
40900	53017	p111028	6b9ce31517454b321e83408c03990595	Mr.	NJ	
27017	126271	p191642	2de954340b83289b322a80f2cd641cb8	Ms.	IL	
61986	61301	p080839	547945c09afffecffebda041fc437602	Mrs.	NC	

73196 rows × 18 columns

In [0]:
```
#project_grade_category
no_of_unique_grade  = X_train["clean_project_grade_category"].nunique()
embed_size_grade = int(min(np.ceil((no_of_unique_grade)/2), 50 ))

input_grade= Input(shape=(1,),name="grade_cat")
embed_grade = Embedding(no_of_unique_grade,embed_size_grade,name="emb_grade",t
rainable=True)(input_grade)
flat_4 = Flatten()(embed_grade)
lab_enc = LabelEncoder()
enc_grade_train = lab_enc.fit_transform(X_train["clean_project_grade_category"
])
enc_grade_test = lab_enc.transform(X_test["clean_project_grade_category"])
```

In [0]:
```
#project_subject_categories
no_of_unique_subcat  = X_train["clean_categories"].nunique()
embed_size_subcat = int(min(np.ceil((no_of_unique_subcat)/2), 50 ))

input_subcat= Input(shape=(1,),name="sub_cat")
embed_subcat = Embedding(no_of_unique_subcat,embed_size_subcat,name="emb_subca
t",trainable=True)(input_subcat)
flat_5 = Flatten()(embed_subcat)
lab_enc = LabelEncoder()
enc_subcat_train = lab_enc.fit_transform(X_train["clean_categories"])
X_test["clean_categories"] = X_test["clean_categories"].map(lambda s: ' ' if s
not in lab_enc.classes_ else s)
lab_enc.classes_ = np.append(lab_enc.classes_, ' ')
enc_subcat_test= lab_enc.transform(X_test["clean_categories"])
```

In [0]:
```
#project_subject_subcategories
no_of_unique_subcat_1  = X_train["clean_subcategories"].nunique()
embed_size_subcat_1 = int(min(np.ceil((no_of_unique_subcat_1)/2), 50 ))

input_subcat_1= Input(shape=(1,),name="sub_cat_1")
embedding_subcat_1 = Embedding(no_of_unique_subcat_1,embed_size_subcat_1,name=
"emb_subcat_1",trainable=True)(input_subcat_1)
flat_6 = Flatten()(embedding_subcat_1)
lab_enc = LabelEncoder()
enc_subcat_1_train = lab_enc.fit_transform(X_train["clean_subcategories"])
X_test["clean_subcategories"] = X_test["clean_subcategories"].map(lambda s: '
 ' if s not in lab_enc.classes_ else s)
lab_enc.classes_ = np.append(lab_enc.classes_, ' ')
enc_subcat_1_test= lab_enc.transform(X_test["clean_subcategories"])
```

In [0]:
```
numerical_train_a=X_train['digits_in_summary'].values.reshape(-1, 1)
numerical_train_b=X_train['price'].values.reshape(-1, 1)
numerical_train_c=X_train['quantity'].values.reshape(-1, 1)
numerical_train_d=X_train['teacher_number_of_previously_posted_projects'].valu
es.reshape(-1, 1)
numerical_test_a=X_test['digits_in_summary'].values.reshape(-1, 1)
numerical_test_b=X_test['price'].values.reshape(-1, 1)
numerical_test_c=X_test['quantity'].values.reshape(-1, 1)
numerical_test_d=X_test['teacher_number_of_previously_posted_projects'].values
.reshape(-1, 1)
numerical_train=np.concatenate((numerical_train_a,numerical_train_b,numerical_
train_c,numerical_train_d),axis=1)
numerical_test=np.concatenate((numerical_test_a,numerical_test_b,numerical_tes
t_c,numerical_test_d),axis=1)

from sklearn.preprocessing import StandardScaler
normal=StandardScaler()
normal_train=normal.fit_transform(numerical_train)
normal_test=normal.transform(numerical_test)
```

In [0]:
```
numerical_feats = Input(shape=(4,),name="numerical_features")
numerical_featss = Dense(100,activation="relu",kernel_initializer="he_normal",
kernel_regularizer=regularizers.l2(0.001))(numerical_feats)
```

In [0]:
```
x = concatenate([flat_1,flat_2,flat_3,flat_4,flat_5,flat_6,numerical_featss])
x = Dense(128,activation="relu", kernel_initializer="he_normal",kernel_regular
izer=regularizers.l2(0.001))(x)
x=Dropout(0.5)(x)
x = Dense(256,activation="relu",kernel_initializer="he_normal",kernel_regulari
zer=regularizers.l2(0.001))(x)
x=Dropout(0.5)(x)
x = Dense(64,activation="relu", kernel_initializer="he_normal",kernel_regulari
zer=regularizers.l2(0.001))(x)
x = BatchNormalization()(x)
output = Dense(2, activation='softmax', name='output')(x)
model_1 = Model(inputs=[input_text,input_prefix,input_state,input_grade,input_
subcat,input_subcat_1,numerical_feats],outputs=[output])
```

In [0]:
```
train_data_1 = [padded_train,enc_prefix_train,enc_state_train,enc_grade_train,
enc_subcat_train,enc_subcat_1_train,normal_train]
test_data_1 = [padded_test,enc_prefix_test,enc_state_test,enc_grade_test,enc_s
ubcat_test,enc_subcat_1_test,normal_test]
```

In [0]:
```
from keras.utils import np_utils
Y_train = np_utils.to_categorical(y_train, 2)
Y_test = np_utils.to_categorical(y_test, 2)
```

In [0]:
```
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]:
```
model_1.compile(optimizer=Adam(lr=0.001), loss='categorical_crossentropy', met
rics=[auroc])
```

```
WARNING:tensorflow:From /usr/local/lib/python3.6/dist-packages/keras/optimize
rs.py:793: The name tf.train.Optimizer is deprecated. Please use tf.compat.v
1.train.Optimizer instead.

WARNING:tensorflow:From <ipython-input-29-4a25250c5bd7>:5: py_func (from tens
orflow.python.ops.script_ops) is deprecated and will be removed in a future v
ersion.
Instructions for updating:
tf.py_func is deprecated in TF V2. Instead, there are two
    options available in V2.
    - tf.py_function takes a python function which manipulates tf eager
    tensors instead of numpy arrays. It's easy to convert a tf eager tensor t
o
    an ndarray (just call tensor.numpy()) but having access to eager tensors
    means `tf.py_function`s can use accelerators such as GPUs as well as
    being differentiable using a gradient tape.
    - tf.numpy_function maintains the semantics of the deprecated tf.py_func
    (it is not differentiable, and manipulates numpy arrays). It drops the
    stateful argument making all functions stateful.
```

In [0]:
```
checkpoint_3 = ModelCheckpoint("model_2.h5",monitor="val_auroc",mode="max",sav
e_best_only = True,verbose=1)
NAME = 'model_2'
tensorboard_2 = TensorBoard(log_dir='logss\{}'.format(NAME),update_freq='epoc
h',batch_size=512)
callbacks_2 = [tensorboard_2,checkpoint_3]
```

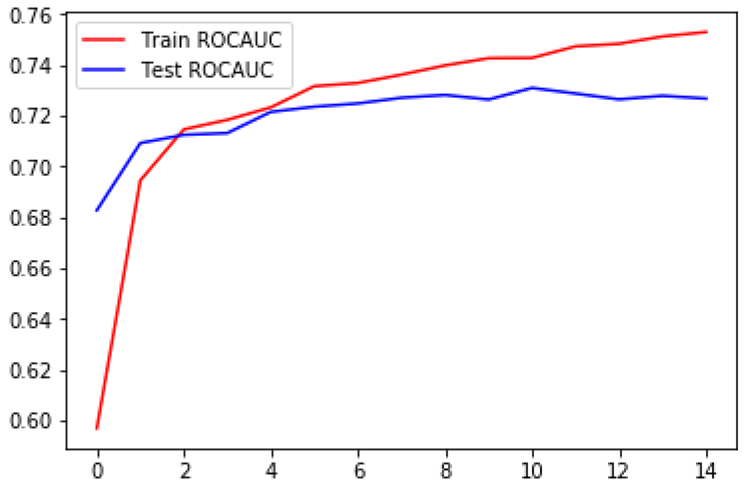
In [0]:
```
# from tensorflow.keras.callbacks import TensorBoard
# import time

# # NAME = 'cc-{}'.format(int(time.time()))
# # tensorboardd = TensorBoard(log_dir='C:\Users\LENOVO\Desktop\applidai\AAIC
\logss\\')
# NAME = 'model_2'
# tensorboardd = TensorBoard(log_dir='logss\{}'.format(NAME),update_freq='epoc
h',batch_size=320)
# # tensorboard_1 = TensorBoard(log_dir='logss\{}'.format(NAME), histogram_fre
q=0, batch_size=320, write_graph=True, write_grads=False, write_images=False,
 embeddings_freq=0, embeddings_layer_names=None, embeddings_metadata=None, emb
eddings_data=None, update_freq='epoch')
```

In [0]:
```
history_1 = model_1.fit(train_data_1,Y_train,batch_size=512,epochs=15,validati
on_data=(test_data_1,Y_test),verbose=1,callbacks=callbacks_2)
```

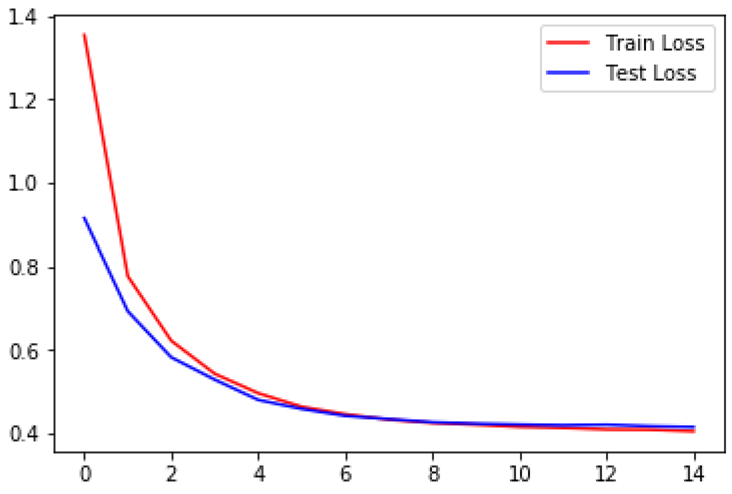
```
WARNING:tensorflow:From /usr/local/lib/python3.6/dist-packages/tensorflow/pyt
hon/ops/math_grad.py:1250: add_dispatch_support.<locals>.wrapper (from tensor
flow.python.ops.array_ops) is deprecated and will be removed in a future vers
ion.
Instructions for updating:
Use tf.where in 2.0, which has the same broadcast rule as np.where
Train on 73196 samples, validate on 36052 samples
WARNING:tensorflow:From /usr/local/lib/python3.6/dist-packages/keras/callback
s.py:1122: The name tf.summary.merge_all is deprecated. Please use tf.compat.
v1.summary.merge_all instead.

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

Epoch 1/15
73196/73196 [==============================] - 207s 3ms/step - loss: 1.3543 -
auroc: 0.5970 - val_loss: 0.9149 - val_auroc: 0.6827

Epoch 00001: val_auroc improved from -inf to 0.68273, saving model to model_
2.h5
Epoch 2/15
73196/73196 [==============================] - 204s 3ms/step - loss: 0.7764 -
auroc: 0.6945 - val_loss: 0.6924 - val_auroc: 0.7092

Epoch 00002: val_auroc improved from 0.68273 to 0.70924, saving model to mode
l_2.h5
Epoch 3/15
73196/73196 [==============================] - 203s 3ms/step - loss: 0.6215 -
auroc: 0.7146 - val_loss: 0.5818 - val_auroc: 0.7125

Epoch 00003: val_auroc improved from 0.70924 to 0.71249, saving model to mode
l_2.h5
Epoch 4/15
73196/73196 [==============================] - 203s 3ms/step - loss: 0.5423 -
auroc: 0.7184 - val_loss: 0.5282 - val_auroc: 0.7132

Epoch 00004: val_auroc improved from 0.71249 to 0.71318, saving model to mode
l_2.h5
Epoch 5/15
73196/73196 [==============================] - 202s 3ms/step - loss: 0.4957 -
auroc: 0.7233 - val_loss: 0.4796 - val_auroc: 0.7215

Epoch 00005: val_auroc improved from 0.71318 to 0.72152, saving model to mode
l_2.h5
Epoch 6/15
73196/73196 [==============================] - 203s 3ms/step - loss: 0.4631 -
auroc: 0.7317 - val_loss: 0.4580 - val_auroc: 0.7236

Epoch 00006: val_auroc improved from 0.72152 to 0.72358, saving model to mode
l_2.h5
Epoch 7/15
73196/73196 [==============================] - 205s 3ms/step - loss: 0.4454 -
auroc: 0.7329 - val_loss: 0.4414 - val_auroc: 0.7248

Epoch 00007: val_auroc improved from 0.72358 to 0.72485, saving model to mode
l_2.h5
```

```
Epoch 8/15
73196/73196 [==============================] - 203s 3ms/step - loss: 0.4321 -
auroc: 0.7362 - val_loss: 0.4335 - val_auroc: 0.7271

Epoch 00008: val_auroc improved from 0.72485 to 0.72712, saving model to mode
l_2.h5
Epoch 9/15
73196/73196 [==============================] - 204s 3ms/step - loss: 0.4243 -
auroc: 0.7399 - val_loss: 0.4259 - val_auroc: 0.7282

Epoch 00009: val_auroc improved from 0.72712 to 0.72820, saving model to mode
l_2.h5
Epoch 10/15
73196/73196 [==============================] - 203s 3ms/step - loss: 0.4201 -
auroc: 0.7427 - val_loss: 0.4226 - val_auroc: 0.7264

Epoch 00010: val_auroc did not improve from 0.72820
Epoch 11/15
73196/73196 [==============================] - 202s 3ms/step - loss: 0.4149 -
auroc: 0.7428 - val_loss: 0.4205 - val_auroc: 0.7310

Epoch 00011: val_auroc improved from 0.72820 to 0.73098, saving model to mode
l_2.h5
Epoch 12/15
73196/73196 [==============================] - 203s 3ms/step - loss: 0.4126 -
auroc: 0.7474 - val_loss: 0.4187 - val_auroc: 0.7288

Epoch 00012: val_auroc did not improve from 0.73098
Epoch 13/15
73196/73196 [==============================] - 202s 3ms/step - loss: 0.4088 -
auroc: 0.7484 - val_loss: 0.4200 - val_auroc: 0.7265

Epoch 00013: val_auroc did not improve from 0.73098
Epoch 14/15
73196/73196 [==============================] - 202s 3ms/step - loss: 0.4081 -
auroc: 0.7513 - val_loss: 0.4165 - val_auroc: 0.7279

Epoch 00014: val_auroc did not improve from 0.73098
Epoch 15/15
73196/73196 [==============================] - 203s 3ms/step - loss: 0.4043 -
auroc: 0.7530 - val_loss: 0.4144 - val_auroc: 0.7268

Epoch 00015: val_auroc did not improve from 0.73098
```

In [0]:
```
plt.plot(history_1.history['auroc'], 'r')
plt.plot(history_1.history['val_auroc'], 'b')
plt.legend({'Train ROCAUC': 'r', 'Test ROCAUC':'b'})
plt.show()
```

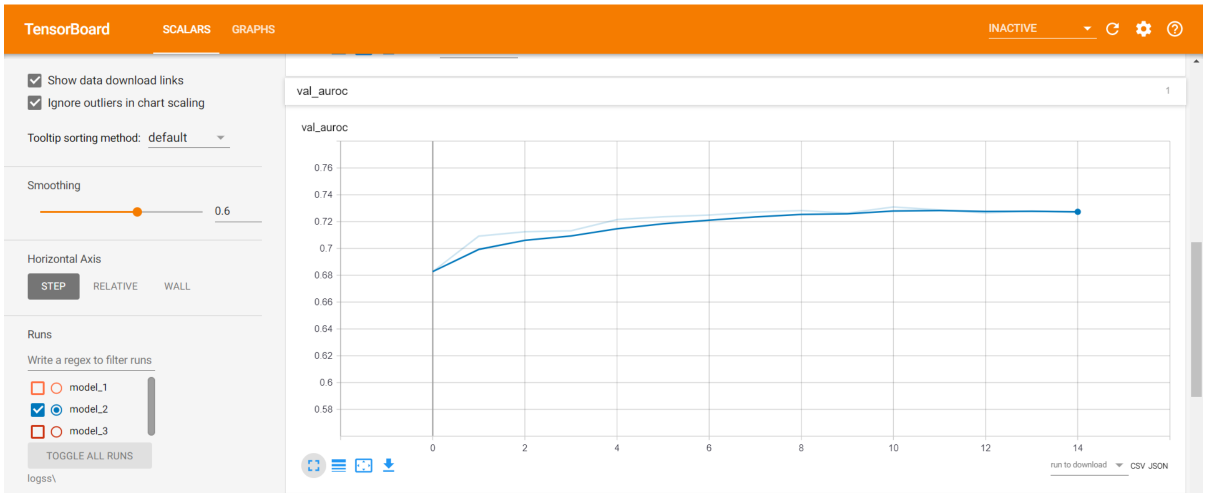


In [0]:
```
plt.plot(history_1.history['loss'], 'r')
plt.plot(history_1.history['val_loss'], 'b')
plt.legend({'Train Loss': 'r', 'Test Loss':'b'})
plt.show()
```



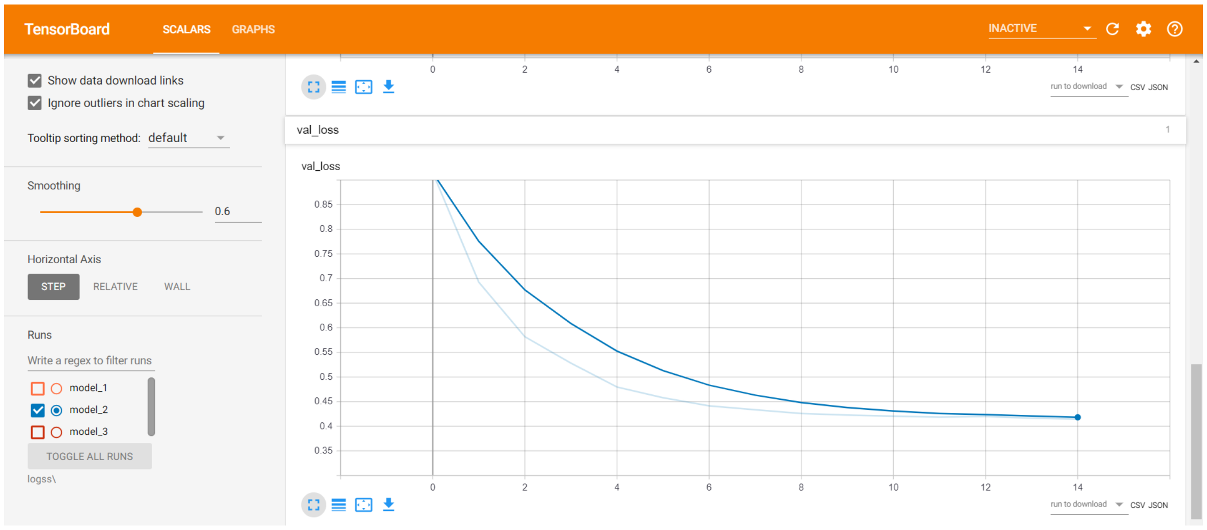
In [3]:
```python
from IPython.display import Image
Image("model2auc.png")
```

Out[3]:



In [4]:
```python
from IPython.display import Image
Image("model2loss.png")
```

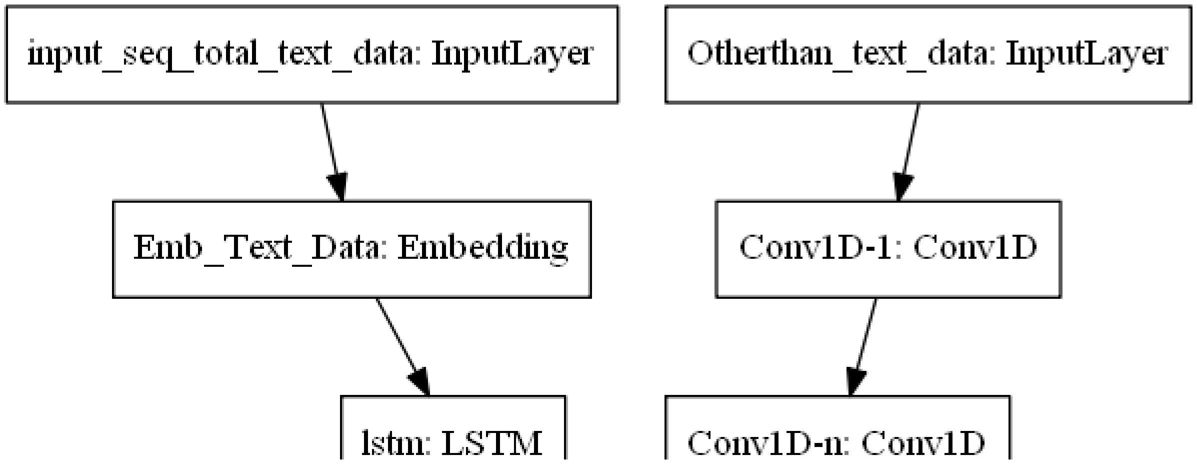
Out[4]:



Model-2(0.72) is not performing well when compared to Model-1

In [0]:
```python

```

Model-3


input_seq_total_text_data: InputLayer
Otherthan_text_data: InputLayer
Emb_Text_Data: Embedding
Conv1D-1: Conv1D
lstm: LSTM
Conv1D-n: Conv1D

- **input_seq_total_text_data**:

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

    . 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 output.

    . 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 al
  l these onehot vectors

    . Neumerical values and use CNN1D (https://keras.io/getting-started/sequentia
  l-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 [0]:

In [0]:
```
# importing required libraries
import warnings
warnings.filterwarnings("ignore")
import pandas as pd
import numpy as np
from keras.layers import Input, Embedding, LSTM, Dropout, BatchNormalization,
Dense, concatenate, Flatten, Conv1D, MaxPool1D, LeakyReLU, ELU, SpatialDropout
1D, MaxPooling1D, GlobalAveragePooling1D, GlobalMaxPooling1D
from keras.preprocessing.text import Tokenizer, one_hot
from keras.preprocessing.sequence import pad_sequences
from keras.models import Model, load_model
from keras import regularizers
from keras.optimizers import *
from keras.callbacks import ModelCheckpoint, EarlyStopping, TensorBoard, Reduc
eLROnPlateau
from sklearn.feature_extraction.text import TfidfVectorizer, CountVectorizer
from sklearn.metrics import roc_auc_score
import tensorflow as tf
import matplotlib.pyplot as plt
%matplotlib inline
import re
from tqdm import tqdm
from sklearn.preprocessing import LabelEncoder
import seaborn as sns
import pickle
```

```
Using TensorFlow backend.
```

In [0]:
```
%matplotlib inline
import warnings
warnings.filterwarnings("ignore")

import sqlite3
import pandas as pd
import numpy as np
import nltk
import string
import matplotlib.pyplot as plt
import seaborn as sns
from sklearn.feature_extraction.text import TfidfTransformer
from sklearn.feature_extraction.text import TfidfVectorizer

from sklearn.feature_extraction.text import CountVectorizer
from sklearn.metrics import confusion_matrix
from sklearn import metrics
from sklearn.metrics import roc_curve, auc
from nltk.stem.porter import PorterStemmer

import re
# Tutorial about Python regular expressions: https://pymotw.com/2/re/
import string
from nltk.corpus import stopwords
from nltk.stem import PorterStemmer
from nltk.stem.wordnet import WordNetLemmatizer

from gensim.models import Word2Vec
from gensim.models import KeyedVectors
import pickle

from tqdm import tqdm_notebook as tqdm1
from tqdm import tqdm
import time
import os

from plotly import plotly
import plotly.offline as offline
import plotly.graph_objs as go
offline.init_notebook_mode()
from collections import Counter

from sklearn.model_selection import train_test_split
```

In [0]:
```
# project_data = pd.read_csv('preprocessed_data.csv')
# project_data.head()
```

In [0]:
```
# project_data = pd.read_csv('train_data.csv', nrows=50000)
project_data = pd.read_csv('train_data.csv',nrows=60000)
resource_data = pd.read_csv('resources.csv')
```

In [0]:
```
print("Number of data points in train data", project_data.shape)
print('-'*50)
print("The attributes of data :", project_data.columns.values)
```

```
Number of data points in train data (109248, 17)
--------------------------------------------------
The attributes of data : ['Unnamed: 0' 'id' 'teacher_id' 'teacher_prefix' 'sc
hool_state'
 'project_submitted_datetime' 'project_grade_category'
 'project_subject_categories' 'project_subject_subcategories'
 'project_title' 'project_essay_1' 'project_essay_2' 'project_essay_3'
 'project_essay_4' 'project_resource_summary'
 'teacher_number_of_previously_posted_projects' 'project_is_approved']
```

Text preprocessing(1)

In [0]:
```
catogories = list(project_data['project_subject_categories'].values)
# remove special characters from list of strings python: https://stackoverflo
w.com/a/47301924/4084039

# https://www.geeksforgeeks.org/removing-stop-words-nltk-python/
# https://stackoverflow.com/questions/23669024/how-to-strip-a-specific-word-fr
om-a-string
# https://stackoverflow.com/questions/8270092/remove-all-whitespace-in-a-strin
g-in-python
cat_list = []
for i in catogories:
    temp = ""
    # consider we have text like this "Math & Science, Warmth, Care & Hunger"
    for j in i.split(','): # it will split it in three parts ["Math & Scienc
e", "Warmth", "Care & Hunger"]
        if 'The' in j.split(): # this will split each of the catogory based on
space "Math & Science"=> "Math","&", "Science"
            j=j.replace('The','') # if we have the words "The" we are going to
replace it with ''(i.e removing 'The')
        j = j.replace(' ','') # we are placeing all the ' '(space) with ''(emp
ty) ex:"Math & Science"=>"Math&Science"
        temp+=j.strip()+" " #" abc ".strip() will return "abc", remove the tra
iling spaces
        temp = temp.replace('&','_') # we are replacing the & value into
    cat_list.append(temp.strip())
```

In [0]:
```
project_data['clean_categories'] = cat_list
project_data.drop(['project_subject_categories'], axis=1, inplace=True)
project_data.head(5)
```

Out[0]:

	Unnamed: 0	id	teacher_id	teacher_prefix	school_state	project_
0	160221	p253737	c90749f5d961ff158d4b4d1e7dc665fc	Mrs.	IN	
1	140945	p258326	897464ce9ddc600bced1151f324dd63a	Mr.	FL	
2	21895	p182444	3465aaf82da834c0582ebd0ef8040ca0	Ms.	AZ	
3	45	p246581	f3cb9bffbba169bef1a77b243e620b60	Mrs.	KY	
4	172407	p104768	be1f7507a41f8479dc06f047086a39ec	Mrs.	TX	

In [0]:
```
# count of all the words in corpus python: https://stackoverflow.com/a/2289859
5/4084039
from collections import Counter
my_counter = Counter()
for word in project_data['clean_categories'].values:
    my_counter.update(word.split())
my_counter
```

Out[0]:
```
Counter({'Literacy_Language': 52239,
         'History_Civics': 5914,
         'Health_Sports': 14223,
         'Math_Science': 41421,
         'SpecialNeeds': 13642,
         'AppliedLearning': 12135,
         'Music_Arts': 10293,
         'Warmth': 1388,
         'Care_Hunger': 1388})
```

In [0]:
```
# dict sort by value python: https://stackoverflow.com/a/613218/4084039
cat_dict = dict(my_counter)
sorted_cat_dict = dict(sorted(cat_dict.items(), key=lambda kv: kv[1]))


# ind = np.arange(len(sorted_cat_dict))
# plt.figure(figsize=(20,5))
# p1 = plt.bar(ind, list(sorted_cat_dict.values()))

# plt.ylabel('Projects')
# plt.title('% of projects aproved category wise')
# plt.xticks(ind, list(sorted_cat_dict.keys()))
# plt.show()
# print(sorted_cat_dict)
```

In [0]:
```
sub_catogories = list(project_data['project_subject_subcategories'].values)
# remove special characters from list of strings python: https://stackoverflo
w.com/a/47301924/4084039

# https://www.geeksforgeeks.org/removing-stop-words-nltk-python/
# https://stackoverflow.com/questions/23669024/how-to-strip-a-specific-word-fr
om-a-string
# https://stackoverflow.com/questions/8270092/remove-all-whitespace-in-a-strin
g-in-python

sub_cat_list = []
for i in sub_catogories:
    temp = ""
    # consider we have text like this "Math & Science, Warmth, Care & Hunger"
    for j in i.split(','): # it will split it in three parts ["Math & Scienc
e", "Warmth", "Care & Hunger"]
        if 'The' in j.split(): # this will split each of the catogory based on
space "Math & Science"=> "Math","&", "Science"
            j=j.replace('The','') # if we have the words "The" we are going to
replace it with ''(i.e removing 'The')
        j = j.replace(' ','') # we are placeing all the ' '(space) with ''(emp
ty) ex:"Math & Science"=>"Math&Science"
        temp +=j.strip()+" "#" abc ".strip() will return "abc", remove the tra
iling spaces
        temp = temp.replace('&','_')
    sub_cat_list.append(temp.strip())
```

In [0]:
```
project_data['clean_subcategories'] = sub_cat_list
project_data.drop(['project_subject_subcategories'], axis=1, inplace=True)
project_data.head(2)
```

Out[0]:

	Unnamed: 0	id	teacher_id	teacher_prefix	school_state	project_
0	160221	p253737	c90749f5d961ff158d4b4d1e7dc665fc	Mrs.	IN	
1	140945	p258326	897464ce9ddc600bced1151f324dd63a	Mr.	FL	

In [0]:
```
# count of all the words in corpus python: https://stackoverflow.com/a/2289859
5/4084039
from collections import Counter
my_counter = Counter()
for word in project_data['clean_subcategories'].values:
    my_counter.update(word.split())
```

In [0]:
```
# dict sort by value python: https://stackoverflow.com/a/613218/4084039
sub_cat_dict = dict(my_counter)
sorted_sub_cat_dict = dict(sorted(sub_cat_dict.items(), key=lambda kv: kv[1]))


# ind = np.arange(len(sorted_sub_cat_dict))
# plt.figure(figsize=(20,5))
# p1 = plt.bar(ind, list(sorted_sub_cat_dict.values()))

# plt.ylabel('Projects')
# plt.title('% of projects aproved state wise')
# plt.xticks(ind, list(sorted_sub_cat_dict.keys()))
# plt.show()
```

In [0]:
```
# merge two column text dataframe:
project_data["essay"] = project_data["project_essay_1"].map(str) +\
                        project_data["project_essay_2"].map(str) + \
                        project_data["project_essay_3"].map(str) + \
                        project_data["project_essay_4"].map(str)
```

In [0]:
```
# https://stackoverflow.com/questions/22407798/how-to-reset-a-dataframes-index
es-for-all-groups-in-one-step
price_data = resource_data.groupby('id').agg({'price':'sum', 'quantity':'sum'
}).reset_index()
price_data.head(2)
```

Out[0]:

	id	price	quantity
0	p000001	459.56	7
1	p000002	515.89	21

In [0]:
```
# join two dataframes in python:
project_data = pd.merge(project_data, price_data, on='id', how='left')
```

In [0]:
```
#presence of the numerical digits in a strings with numeric : https://stackove
rflow.com/a/19859308/8089731
def hasNumbers(inputString):
    return any(i.isdigit() for i in inputString)
p1 = project_data[['id','project_resource_summary']]
p1 = pd.DataFrame(data=p1)
p1.columns = ['id','digits_in_summary']
p1['digits_in_summary'] =  p1['digits_in_summary'].map(hasNumbers)
# https://stackoverflow.com/a/17383325/8089731
p1['digits_in_summary'] = p1['digits_in_summary'].astype(int)
project_data = pd.merge(project_data, p1, on='id', how='left')
project_data.head(5)
```

Out[0]:

	Unnamed: 0	id	teacher_id	teacher_prefix	school_state	project_
0	160221	p253737	c90749f5d961ff158d4b4d1e7dc665fc	Mrs.	IN	
1	140945	p258326	897464ce9ddc600bced1151f324dd63a	Mr.	FL	
2	21895	p182444	3465aaf82da834c0582ebd0ef8040ca0	Ms.	AZ	
3	45	p246581	f3cb9bffbba169bef1a77b243e620b60	Mrs.	KY	
4	172407	p104768	be1f7507a41f8479dc06f047086a39ec	Mrs.	TX	

5 rows × 21 columns

Text preprocessing(2)

In [0]:
```
# https://stackoverflow.com/a/47091490/4084039
import re

def decontracted(phrase):
    # specific
    phrase = re.sub(r"won't", "will not", phrase)
    phrase = re.sub(r"can\'t", "can not", phrase)

    # general
    phrase = re.sub(r"n\'t", " not", phrase)
    phrase = re.sub(r"\'re", " are", phrase)
    phrase = re.sub(r"\'s", " is", phrase)
    phrase = re.sub(r"\'d", " would", phrase)
    phrase = re.sub(r"\'ll", " will", phrase)
    phrase = re.sub(r"\'t", " not", phrase)
    phrase = re.sub(r"\'ve", " have", phrase)
    phrase = re.sub(r"\'m", " am", phrase)
    return phrase
```

In [0]:
```
# https://gist.github.com/sebleier/554280
# we are removing the words from the stop words list: 'no', 'nor', 'not'
stopwords= ['i', 'me', 'my', 'myself', 'we', 'our', 'ours', 'ourselves', 'you'
, "you're", "you've",\
            "you'll", "you'd", 'your', 'yours', 'yourself', 'yourselves', 'he'
, 'him', 'his', 'himself', \
            'she', "she's", 'her', 'hers', 'herself', 'it', "it's", 'its', 'it
self', 'they', 'them', 'their',\
            'theirs', 'themselves', 'what', 'which', 'who', 'whom', 'this', 't
hat', "that'll", 'these', 'those', \
            'am', 'is', 'are', 'was', 'were', 'be', 'been', 'being', 'have',
'has', 'had', 'having', 'do', 'does', \
            'did', 'doing', 'a', 'an', 'the', 'and', 'but', 'if', 'or', 'becau
se', 'as', 'until', 'while', 'of', \
            'at', 'by', 'for', 'with', 'about', 'against', 'between', 'into',
'through', 'during', 'before', 'after',\
            'above', 'below', 'to', 'from', 'up', 'down', 'in', 'out', 'on',
'off', 'over', 'under', 'again', 'further',\
            'then', 'once', 'here', 'there', 'when', 'where', 'why', 'how', 'a
ll', 'any', 'both', 'each', 'few', 'more',\
            'most', 'other', 'some', 'such', 'only', 'own', 'same', 'so', 'tha
n', 'too', 'very', \
            's', 't', 'can', 'will', 'just', 'don', "don't", 'should', "shoul
d've", 'now', 'd', 'll', 'm', 'o', 're', \
            've', 'y', 'ain', 'aren', "aren't", 'couldn', "couldn't", 'didn',
"didn't", 'doesn', "doesn't", 'hadn',\
            "hadn't", 'hasn', "hasn't", 'haven', "haven't", 'isn', "isn't", 'm
a', 'mightn', "mightn't", 'mustn',\
            "mustn't", 'needn', "needn't", 'shan', "shan't", 'shouldn', "shoul
dn't", 'wasn', "wasn't", 'weren', "weren't", \
            'won', "won't", 'wouldn', "wouldn't"]
```

In [0]:
```
# Combining all the above statemennts
from tqdm import tqdm
preprocessed_essays = []
# tqdm is for printing the status bar
for sentance in tqdm1(project_data['essay'].values):
    sent = decontracted(sentance)
    sent = sent.replace('\\r', ' ')
    sent = sent.replace('\\"', ' ')
    sent = sent.replace('\\n', ' ')
    sent = re.sub('[^A-Za-z0-9]+', ' ', sent)
    sent = re.sub('nannan', '', sent)
    # https://gist.github.com/sebleier/554280
    sent = ' '.join(e for e in sent.split() if e not in stopwords)
    preprocessed_essays.append(sent.lower().strip())
```

In [0]:
```
from tqdm import tqdm
preprocessed_titles = []
# tqdm is for printing the status bar
for title in tqdm1(project_data['project_title'].values):
    _title = decontracted(title)
    _title = _title.replace('\\r', ' ')
    _title = _title.replace('\\"', ' ')
    _title = _title.replace('\\n', ' ')
    _title = re.sub('[^A-Za-z0-9]+', ' ', _title)
    # https://gist.github.com/sebleier/554280
    _title = ' '.join(e for e in _title.split() if e not in stopwords)
    preprocessed_titles.append(_title.lower().strip())
```

In [0]:
```
preprocessed_titles[1000]
```

Out[0]:
```
'sailing into super 4th grade year'
```

In [0]:
```
project_grade_catogories = list(project_data['project_grade_category'].values)
# remove special characters from list of strings python: https://stackoverflo
w.com/a/47301924/4084039

# https://www.geeksforgeeks.org/removing-stop-words-nltk-python/
# https://stackoverflow.com/questions/23669024/how-to-strip-a-specific-word-fr
om-a-string
# https://stackoverflow.com/questions/8270092/remove-all-whitespace-in-a-strin
g-in-python

project_grade_cat_list = []
for i in tqdm1(project_grade_catogories):
    temp = ""
    # consider we have text like this "Math & Science, Warmth, Care & Hunger"
    for j in i.split(','): # it will split it in three parts ["Math & Scienc
e", "Warmth", "Care & Hunger"]
        if 'The' in j.split(): # this will split each of the catogory based on
space "Math & Science"=> "Math","&", "Science"
            j=j.replace('The','') # if we have the words "The" we are going to
replace it with ''(i.e removing 'The')
        j = j.replace(' ','') # we are placeing all the ' '(space) with ''(emp
ty) ex:"Math & Science"=>"Math&Science"
        temp +=j.strip()+" "#" abc ".strip() will return "abc", remove the tra
iling spaces
        temp = temp.replace('&','_')
    project_grade_cat_list.append(temp.strip())
```

In [0]:
```
project_data['clean_project_grade_category'] = project_grade_cat_list
project_data.drop(['project_grade_category'], axis=1, inplace=True)
project_data.head(2)
```

Out[0]:

	Unnamed: 0	id	teacher_id	teacher_prefix	school_state	project_
0	160221	p253737	c90749f5d961ff158d4b4d1e7dc665fc	Mrs.	IN	
1	140945	p258326	897464ce9ddc600bced1151f324dd63a	Mr.	FL	

2 rows × 21 columns

In [0]:
```
project_data.drop(['project_essay_1','project_essay_2','project_essay_3','proj
ect_essay_4'], axis=1, inplace=True)
project_data.head(2)
```

Out[0]:

	Unnamed: 0	id	teacher_id	teacher_prefix	school_state	project_
0	160221	p253737	c90749f5d961ff158d4b4d1e7dc665fc	Mrs.	IN	
1	140945	p258326	897464ce9ddc600bced1151f324dd63a	Mr.	FL	

In [0]:
```
#Replacing Nan's with maximum occured value: https://stackoverflow.com/a/51053
916/8089731
project_data['teacher_prefix'].value_counts().argmax()
project_data.fillna(value=project_data['teacher_prefix'].value_counts().argmax
(),axis=1,inplace=True)
```

In [0]:
```
project_data['preprocessed_essays'] = preprocessed_essays
project_data['preprocessed_titles'] = preprocessed_titles
```

In [0]:
```
X_train, X_test, y_train, y_test = train_test_split(project_data,project_data[
'project_is_approved'], test_size=0.33, stratify = project_data['project_is_ap
proved'])
# X_train, X_cv, y_train, y_cv = train_test_split(X_train, y_train, test_size=
0.33, stratify=y_train)

X_train.drop(['project_is_approved'], axis=1, inplace=True)
X_test.drop(['project_is_approved'], axis=1, inplace=True)
# X_cv.drop(['project_is_approved'], axis=1, inplace=True)
print(X_train.shape)
print(X_test.shape)
```

```
(73196, 18)
(36052, 18)
```

In [0]:
```
X_train
```

Out[0]:

	Unnamed: 0	id	teacher_id	teacher_prefix	school_state	p
8612	151084	p053771	e5c38539378a091da8d9c9fbaf1922ec	Teacher	SC	
60457	67609	p168792	ad11c5c0ec1abbe7df7c6884d37e1d9f	Ms.	UT	
33028	161283	p168952	9343db5e432d93162ee648c648938383	Mrs.	VA	
97078	111495	p040276	0aa0c61573f4abe738b84c433293d98e	Ms.	MD	
46772	45037	p086290	95bddc931c75dfe44e7e374aee5d1221	Ms.	CA	
98772	179124	p204692	9e636ecd4e53a1233b5130739c977961	Mrs.	TX	
98476	74343	p075531	33e1ef4e5b7ca33b2bbf81d727bdcf76	Ms.	ME	
47492	70169	p238823	2235fc8fa05c8b6243fcea3d2a6a9738	Mrs.	MI	
43234	166460	p089073	6db62616b4ef6efc2310088f7ea0ae14	Ms.	GA	
105255	52719	p051505	d7dc008c1acf09c7a81efe3fd0d7d3bd	Ms.	WA	
66318	74532	p172031	5a29d0437db7a601684bf4eb5a8c5b1a	Ms.	OH	
48302	116592	p043295	560104957f16d35cb65a771d8c6a50b1	Mrs.	NJ	

	Unnamed: 0	id	teacher_id	teacher_prefix	school_state	p
93952	42548	p000485	1adcae303be41911b857b89a8b3f3ddf	Mrs.	MO	
75348	17887	p037097	f4639e654b1de86679645d6639d736d0	Ms.	PA	
42531	144794	p167181	66732a945fb4fef9eaffed7dc7ef9fc4	Mrs.	ME	
104750	74722	p167602	95cadb9b4c743cf674f04aa887552a42	Mrs.	CA	
30843	101532	p125865	af12d8d89f8256f53489774bb9eef223	Mrs.	TN	
23607	49275	p042949	4c820d5369b64a956eac1746d2396d48	Ms.	OH	
24361	30579	p212053	bb9d671c95cd89fd2ef1672c5faee4d8	Mrs.	UT	
98034	140800	p188478	e0b310802c30b715bb1221542e3c0c7f	Ms.	IA	
52250	139544	p143425	835976b3e023e1759a090d20fd382480	Teacher	CO	
10781	100444	p176819	ab15f85d0c9ae152f4713da9c5db0df5	Ms.	CA	
10369	109655	p044752	ddddf1665d2649a59daaf36364094997	Ms.	NJ	
95080	35674	p144456	d47c2072427005a90cee1d9069ad0e4d	Ms.	OK	

	Unnamed: 0	id	teacher_id	teacher_prefix	school_state	p
71448	164129	p212775	90c63f8eb69943f3904ffb605e3501ff	Mrs.	SC	
69953	90345	p032110	3a19dc26a8bf7fd2f67abfb27cb46ec8	Ms.	FL	
85493	65457	p145820	5c5022db019385ea645b9868bee7d8b7	Mrs.	NC	
74419	36135	p109237	38f79dfb031487edacd2f58f6a863168	Ms.	OH	
91531	115533	p100945	ebfc4fc9c7d4d046582feb5ef4d48d28	Mrs.	MD	
3554	110658	p047972	a23974b9fcb050f40e3d3c09bd505bd5	Ms.	OK	
...	...	...	...	...	...	
95293	92843	p081528	5ca5ffc3ffc68f06d689499098a81567	Mrs.	CA	
58979	59189	p011029	e1bee79bebb2e70b4a92b1b4f168ccfe	Mrs.	NC	
16649	33689	p130136	ff0b1090478595d0284efc36e0743365	Ms.	MA	
37315	141222	p213052	1418aceca52b15991c64d30587a4c73b	Mrs.	NE	
60046	22238	p223187	7944e6b6e33fe78bd5789110a4985ad8	Mrs.	NJ	
61173	87765	p242520	e2dda7b9e59c775784729c79fe732634	Mrs.	TX	

	Unnamed: 0	id	teacher_id	teacher_prefix	school_state	p
13818	179767	p112729	6c4ec85c817d1849275d55f4fe29a959	Mrs.	SC	
34019	161772	p115139	858d7d8d2a5cdf0d57a15243e4c9fb78	Ms.	ME	
67334	140854	p177768	2f86e316cd6b20ce94658487c28c5e61	Mrs.	NY	
61235	75398	p060766	7c601754709e9a3e4dbde8bf71e02867	Teacher	FL	
108232	180109	p111011	36d610674d982b8250635783766cb0c6	Mrs.	PA	
92455	3086	p237594	c9095bc282f075ca20225bce140f6728	Mrs.	FL	
8140	163662	p254904	20814335d4ab1373cd9e5977751736a4	Ms.	MA	
4553	160534	p115972	4072965f425d1665d31ef09002d377ca	Mrs.	CA	
26096	150454	p050936	af5de9eb7f32b2f49ec93e6629e7fccc	Ms.	FL	
25	20142	p009037	b8bf3507cee960d5fedcb27719df2d59	Mrs.	AL	
75516	109875	p016512	332eeffa6247676bc59f821f023f1ffb	Mr.	TX	
22412	41316	p179991	f8976d816853106ef73e56ea95f3185e	Mrs.	NC	
69059	98666	p097870	a1e3e74c2e8d949fddecf2ffbe35ecc3	Mr.	MI	

	Unnamed: 0	id	teacher_id	teacher_prefix	school_state	p
108307	177649	p050817	f6d705f4a12eaba8356b0a71e70cc4b1	Ms.	TX	
13457	2374	p193849	dda9818fcd6491ca3bcbadfe7b91438d	Ms.	NC	
23277	53819	p242981	29ff25816e2c9a3dad44a50bbd89c581	Mrs.	CA	
53410	39497	p013532	9c85244be0bda0958a8718effb3ae588	Mrs.	SC	
103537	172398	p130349	d2fee6790defdef81735437f9fcbd08b	Mrs.	FL	
53037	79660	p062747	20d0e267c9beb75e3135a14e7d819004	Ms.	AZ	
94535	16263	p209768	ce17d173096d6895ee828bc89bf0f67b	Mrs.	MA	
1311	126793	p102542	fec25f65a7dd7d21e92ae07448a66930	Mrs.	NC	
36516	42186	p213311	a44f86571a5ea934d4c47575c1967f2c	Ms.	CA	
49167	117825	p030150	39b22562dd7c4adefb3b8dbe7f2d20d1	Ms.	CA	
53188	106683	p035708	6edd0e97b7fd1628dfeb87bf85f96d58	Mrs.	IN	

73196 rows × 18 columns

In [0]:
```
import dill
# dill.dump_session('notebook_envall.db')
dill.load_session('notebook_envall.db')
```

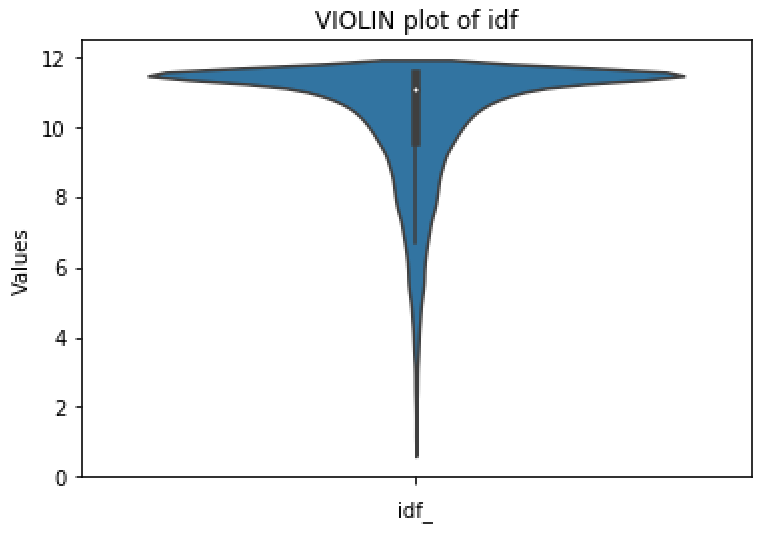
```
Using TensorFlow backend.
```

In [0]:
```
from numpy import array
from numpy import asarray
from numpy import zeros
from keras.preprocessing.text import Tokenizer
from keras.preprocessing.sequence import pad_sequences
from keras.models import Sequential
from keras.layers import Dense
from keras.layers import Flatten
from keras.layers import Embedding
```

In [0]:
```
tfidf = TfidfVectorizer()
_tfidf = tfidf.fit_transform(X_train["essay"])
_dict = dict(zip(tfidf.get_feature_names(),list(tfidf.idf_)))
tfidf_df = pd.DataFrame(list(_dict.items()), columns=['Words', 'Values'])
tfidf_df = tfidf_df.sort_values(by ='Values' )
```

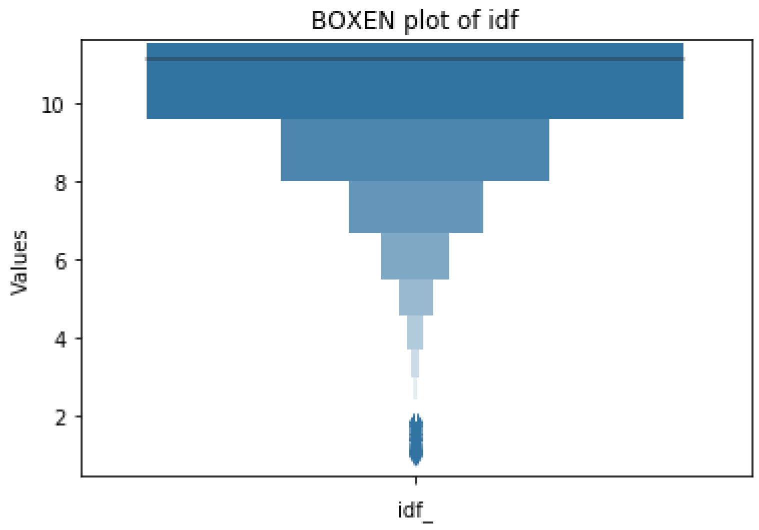
In [0]:
```
sns.violinplot(x = "Values",data=tfidf_df,orient="v")
plt.xlabel("idf_")
plt.title("VIOLIN plot of idf")
```

Out[0]:
```
Text(0.5, 1.0, 'VIOLIN plot of idf')
```



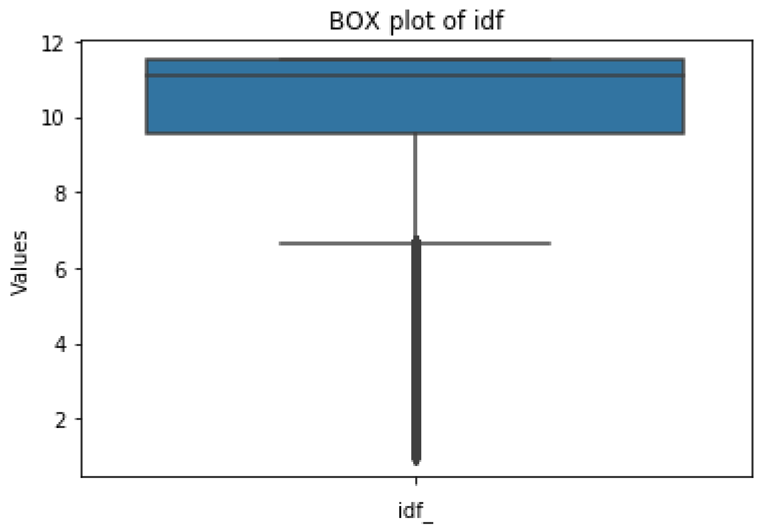
In [0]:
```
sns.boxenplot(x = "Values",data=tfidf_df,orient="v")
plt.xlabel("idf_")
plt.title("BOXEN plot of idf")
```

Out[0]:
```
Text(0.5, 1.0, 'BOXEN plot of idf')
```



In [0]:
```
sns.boxplot(x = "Values",data=tfidf_df,orient="v")
plt.xlabel("idf_")
plt.title("BOX plot of idf")
```

Out[0]:
```
Text(0.5, 1.0, 'BOX plot of idf')
```



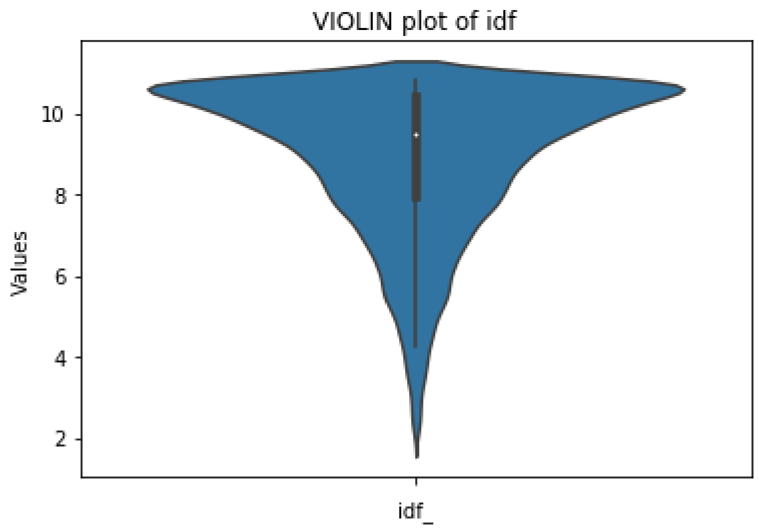
In [0]:
```
print("\n25th Percentile")
print(np.percentile(tfidf_df['Values'],1))
print("\n27th Percentiles")
print(np.percentile(tfidf_df['Values'],27))
```

```
25th Percentile
3.983908779165195

27th Percentiles
9.716003065714995
```

In [0]:
```
tfidf_fil = tfidf_df[(tfidf_df["Values"] >=  2) & (tfidf_df["Values"] <=  11)]
```

In [0]:
```
sns.violinplot(x = "Values",data=tfidf_fil,orient="v")
plt.xlabel("idf_")
plt.title("VIOLIN plot of idf")
```

Out[0]:
```
Text(0.5, 1.0, 'VIOLIN plot of idf')
```



In [0]:
```
fil_words = tfidf_fil["Words"].tolist()
```

In [0]:
```
tokenizer = Tokenizer()
tokenizer.fit_on_texts(fil_words)
seq_train = tokenizer.texts_to_sequences(X_train["essay"])
seq_test = tokenizer.texts_to_sequences(X_test["essay"])
```

In [0]:
```
vocab_size = len(tokenizer.word_index) + 1
```

In [0]:
```
padded_train = pad_sequences(seq_train,maxlen=100)
padded_test = pad_sequences(seq_test, maxlen=100)
```

In [0]:
```
pickle_in = open("/content/drive/My Drive/AAIC/Assignments/glove_vectors","rb"
)
glove_words = pickle.load(pickle_in)
```

In [0]:
```
embedding_matrix = zeros((vocab_size, 300))
for word, i in tokenizer.word_index.items():
    embedding_vector = glove_words.get(word)
    if embedding_vector is not None:
        embedding_matrix[i] = embedding_vector
```

In [0]:
```
print(embedding_matrix.shape)
```

```
(25199, 300)
```

In [0]:
```
# input_text = Input(shape=(800,),name="input_text")
embedding_layer = Embedding(vocab_size,300,weights=[embedding_matrix],input_le
ngth=100,trainable=False)
input_text = Input(shape=(100,),name="input_text")
x = embedding_layer(input_text)
x = LSTM(256,recurrent_dropout=0.5,kernel_regularizer=regularizers.l2(0.001),r
eturn_sequences=True)(x)
flat_1 = Flatten()(x)
```

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

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

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

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

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

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

In [0]:
```
X_train.columns
```

Out[0]:
```
Index(['Unnamed: 0', 'id', 'teacher_id', 'teacher_prefix', 'school_state',
       'project_submitted_datetime', 'project_title',
       'project_resource_summary',
       'teacher_number_of_previously_posted_projects', 'clean_categories',
       'clean_subcategories', 'essay', 'price', 'quantity',
       'digits_in_summary', 'clean_project_grade_category',
       'preprocessed_essays', 'preprocessed_titles'],
      dtype='object')
```

In [0]:
```
vect = CountVectorizer(binary=True)
vect.fit(X_train["teacher_prefix"])
train_prefix = vect.transform(X_train["teacher_prefix"])
test_prefix = vect.transform(X_test["teacher_prefix"])
```

In [0]:
```
vect = CountVectorizer(binary=True)
vect.fit(X_train["school_state"])
train_state = vect.transform(X_train["school_state"])
test_state = vect.transform(X_test["school_state"])
```

In [0]:
```
vect = CountVectorizer(binary=True)
vect.fit(X_train["clean_project_grade_category"])
train_grade = vect.transform(X_train["clean_project_grade_category"])
test_grade = vect.transform(X_test["clean_project_grade_category"])
```

In [0]:
```
vect = CountVectorizer(binary=True)
vect.fit(X_train["clean_categories"])
train_subcat = vect.transform(X_train["clean_categories"])
test_subcat = vect.transform(X_test["clean_categories"])
```

In [0]:
```
vect = CountVectorizer(binary=True)
vect.fit(X_train["clean_subcategories"])
train_subcat_1 = vect.transform(X_train["clean_subcategories"])
test_subcat_1 = vect.transform(X_test["clean_subcategories"])
```

In [0]:
```
numerical_train_a=X_train['digits_in_summary'].values.reshape(-1, 1)
numerical_train_b=X_train['price'].values.reshape(-1, 1)
numerical_train_c=X_train['quantity'].values.reshape(-1, 1)
numerical_train_d=X_train['teacher_number_of_previously_posted_projects'].valu
es.reshape(-1, 1)
numerical_test_a=X_test['digits_in_summary'].values.reshape(-1, 1)
numerical_test_b=X_test['price'].values.reshape(-1, 1)
numerical_test_c=X_test['quantity'].values.reshape(-1, 1)
numerical_test_d=X_test['teacher_number_of_previously_posted_projects'].values
.reshape(-1, 1)
numerical_train=np.concatenate((numerical_train_a,numerical_train_b,numerical_
train_c,numerical_train_d),axis=1)
numerical_test=np.concatenate((numerical_test_a,numerical_test_b,numerical_tes
t_c,numerical_test_d),axis=1)

from sklearn.preprocessing import StandardScaler
normal=StandardScaler()
normal_train=normal.fit_transform(numerical_train)
normal_test=normal.transform(numerical_test)
```

In [0]:
```
from scipy.sparse import hstack
ot_train = hstack([train_prefix,train_state,train_grade,train_subcat,train_sub
cat_1]).todense()
ot_test = hstack([test_prefix,test_state,test_grade,test_subcat,test_subcat_1
]).todense()
ot_all_train = np.hstack((ot_train,normal_train))
ot_all_test = np.hstack((ot_test,normal_test))
ot_train_all = np.expand_dims(ot_all_train,2)
ot_test_all = np.expand_dims(ot_all_test,2)
```

In [0]:
```
inp_conv=Input(shape=(104, 1))
x1 = Conv1D(filters=128, kernel_size=3, activation='relu',kernel_initializer=
"he_normal")(inp_conv)
x1 = Conv1D(filters=128, kernel_size=3, activation='relu',kernel_initializer=
"he_normal")(x1)
x1 = Flatten()(x1)
```

In [0]:
```
x_concatenate = concatenate([flat_1,x1])
x = Dense(128,activation="relu",kernel_initializer="he_normal",kernel_regulari
zer=regularizers.l2(0.001))(x_concatenate)
x=Dropout(0.5)(x)
x = Dense(64,activation="relu",kernel_initializer="he_normal",kernel_regulariz
er=regularizers.l2(0.001))(x)
x=Dropout(0.3)(x)
x = Dense(32,activation="relu",kernel_initializer="he_normal",kernel_regulariz
er=regularizers.l2(0.001))(x)
output = Dense(2, activation='softmax', name='output')(x)
model_1 = Model(inputs=[input_text,inp_conv],outputs=[output])
```

In [0]:
```
train_data_3 = [padded_train,ot_train_all]
test_data_3 = [padded_test,ot_test_all]
from keras.utils import np_utils
Y_train = np_utils.to_categorical(y_train, 2)
Y_test = np_utils.to_categorical(y_test, 2)
```

In [0]:
```
from sklearn.metrics import roc_auc_score
def auc1(y_true, y_pred):
    if len(np.unique(y_true[:,1])) == 1:
        return 0.5
    else:
        return roc_auc_score(y_true, y_pred)
def auroc(y_true, y_pred):
    return tf.py_func(auc1, (y_true, y_pred), tf.double)
```

In [0]:
```
model_1.compile(optimizer=Adam(lr=0.001, beta_1=0.9, beta_2=0.999, epsilon=Non
e, decay=0.0, amsgrad=False), loss='categorical_crossentropy', metrics=[auroc
])
```

```
WARNING:tensorflow:From /usr/local/lib/python3.6/dist-packages/keras/optimize
rs.py:793: The name tf.train.Optimizer is deprecated. Please use tf.compat.v
1.train.Optimizer instead.

WARNING:tensorflow:From <ipython-input-30-b3d027b8e5dd>:8: py_func (from tens
orflow.python.ops.script_ops) is deprecated and will be removed in a future v
ersion.
Instructions for updating:
tf.py_func is deprecated in TF V2. Instead, there are two
    options available in V2.
    - tf.py_function takes a python function which manipulates tf eager
    tensors instead of numpy arrays. It's easy to convert a tf eager tensor t
o
    an ndarray (just call tensor.numpy()) but having access to eager tensors
    means `tf.py_function`s can use accelerators such as GPUs as well as
    being differentiable using a gradient tape.
    - tf.numpy_function maintains the semantics of the deprecated tf.py_func
    (it is not differentiable, and manipulates numpy arrays). It drops the
    stateful argument making all functions stateful.
```

In [0]:
```
checkpoint_3 = ModelCheckpoint("model_3.h5",monitor="val_auroc",mode="max",sav
e_best_only = True,verbose=1)
NAME = 'model_3'
tensorboard_2 = TensorBoard(log_dir='logss\{}'.format(NAME),update_freq='epoc
h',batch_size=512)
callbacks_2 = [tensorboard_2,checkpoint_3]
```

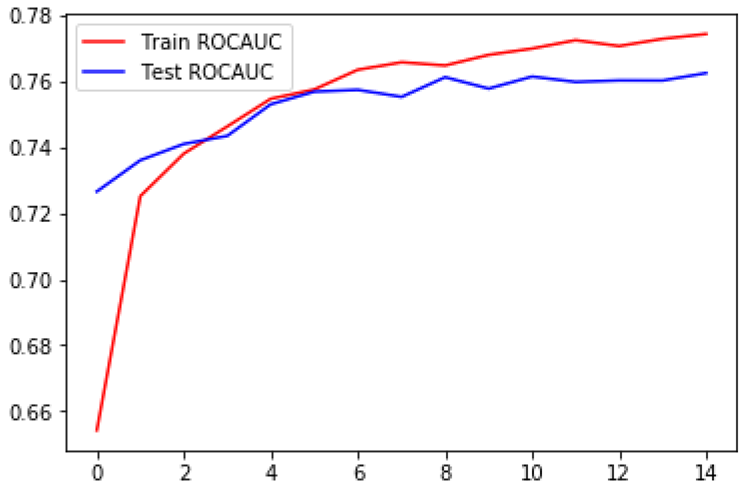
In [0]:
```
# from tensorflow.keras.callbacks import TensorBoard
# import time

# # NAME = 'cc-{}'.format(int(time.time()))
# # tensorboardd = TensorBoard(log_dir='C:\Users\LENOVO\Desktop\applidai\AAIC
\logss\\')
# NAME = 'model_2'
# tensorboardd = TensorBoard(log_dir='logss\{}'.format(NAME),update_freq='epoc
h',batch_size=320)
# # tensorboard_1 = TensorBoard(log_dir='logss\{}'.format(NAME), histogram_fre
q=0, batch_size=320, write_graph=True, write_grads=False, write_images=False,
 embeddings_freq=0, embeddings_layer_names=None, embeddings_metadata=None, emb
eddings_data=None, update_freq='epoch')
```

In [0]:
```
history_1 = model_1.fit(train_data_3,Y_train,batch_size=512,epochs=15,validati
on_data=(test_data_3,Y_test),verbose=1,callbacks=callbacks_2)
```

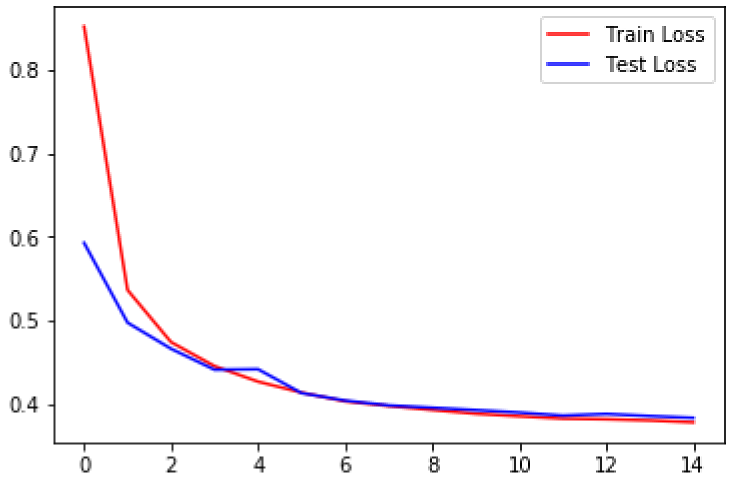
```
WARNING:tensorflow:From /usr/local/lib/python3.6/dist-packages/tensorflow/pyt
hon/ops/math_grad.py:1250: add_dispatch_support.<locals>.wrapper (from tensor
flow.python.ops.array_ops) is deprecated and will be removed in a future vers
ion.
Instructions for updating:
Use tf.where in 2.0, which has the same broadcast rule as np.where
Train on 73196 samples, validate on 36052 samples
WARNING:tensorflow:From /usr/local/lib/python3.6/dist-packages/keras/callback
s.py:1122: The name tf.summary.merge_all is deprecated. Please use tf.compat.
v1.summary.merge_all instead.

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

Epoch 1/15
73196/73196 [==============================] - 34s 465us/step - loss: 0.8525
- auroc: 0.6541 - val_loss: 0.5935 - val_auroc: 0.7265

Epoch 00001: val_auroc improved from -inf to 0.72653, saving model to model_
3.h5
Epoch 2/15
73196/73196 [==============================] - 29s 397us/step - loss: 0.5369
- auroc: 0.7251 - val_loss: 0.4977 - val_auroc: 0.7360

Epoch 00002: val_auroc improved from 0.72653 to 0.73605, saving model to mode
l_3.h5
Epoch 3/15
73196/73196 [==============================] - 29s 400us/step - loss: 0.4744
- auroc: 0.7381 - val_loss: 0.4664 - val_auroc: 0.7410

Epoch 00003: val_auroc improved from 0.73605 to 0.74100, saving model to mode
l_3.h5
Epoch 4/15
73196/73196 [==============================] - 29s 400us/step - loss: 0.4455
- auroc: 0.7463 - val_loss: 0.4413 - val_auroc: 0.7435

Epoch 00004: val_auroc improved from 0.74100 to 0.74345, saving model to mode
l_3.h5
Epoch 5/15
73196/73196 [==============================] - 29s 398us/step - loss: 0.4270
- auroc: 0.7547 - val_loss: 0.4419 - val_auroc: 0.7531

Epoch 00005: val_auroc improved from 0.74345 to 0.75306, saving model to mode
l_3.h5
Epoch 6/15
73196/73196 [==============================] - 29s 400us/step - loss: 0.4138
- auroc: 0.7576 - val_loss: 0.4132 - val_auroc: 0.7569

Epoch 00006: val_auroc improved from 0.75306 to 0.75690, saving model to mode
l_3.h5
Epoch 7/15
73196/73196 [==============================] - 29s 401us/step - loss: 0.4033
- auroc: 0.7636 - val_loss: 0.4043 - val_auroc: 0.7575

Epoch 00007: val_auroc improved from 0.75690 to 0.75746, saving model to mode
l_3.h5
```

```
Epoch 8/15
73196/73196 [==============================] - 29s 401us/step - loss: 0.3977
- auroc: 0.7658 - val_loss: 0.3984 - val_auroc: 0.7553

Epoch 00008: val_auroc did not improve from 0.75746
Epoch 9/15
73196/73196 [==============================] - 29s 396us/step - loss: 0.3929
- auroc: 0.7649 - val_loss: 0.3956 - val_auroc: 0.7613

Epoch 00009: val_auroc improved from 0.75746 to 0.76130, saving model to mode
l_3.h5
Epoch 10/15
73196/73196 [==============================] - 29s 397us/step - loss: 0.3884
- auroc: 0.7681 - val_loss: 0.3928 - val_auroc: 0.7578

Epoch 00010: val_auroc did not improve from 0.76130
Epoch 11/15
73196/73196 [==============================] - 29s 397us/step - loss: 0.3854
- auroc: 0.7700 - val_loss: 0.3898 - val_auroc: 0.7615

Epoch 00011: val_auroc improved from 0.76130 to 0.76149, saving model to mode
l_3.h5
Epoch 12/15
73196/73196 [==============================] - 29s 401us/step - loss: 0.3826
- auroc: 0.7726 - val_loss: 0.3862 - val_auroc: 0.7599

Epoch 00012: val_auroc did not improve from 0.76149
Epoch 13/15
73196/73196 [==============================] - 29s 395us/step - loss: 0.3818
- auroc: 0.7708 - val_loss: 0.3882 - val_auroc: 0.7603

Epoch 00013: val_auroc did not improve from 0.76149
Epoch 14/15
73196/73196 [==============================] - 29s 397us/step - loss: 0.3804
- auroc: 0.7730 - val_loss: 0.3856 - val_auroc: 0.7603

Epoch 00014: val_auroc did not improve from 0.76149
Epoch 15/15
73196/73196 [==============================] - 29s 398us/step - loss: 0.3781
- auroc: 0.7744 - val_loss: 0.3834 - val_auroc: 0.7625

Epoch 00015: val_auroc improved from 0.76149 to 0.76255, saving model to mode
l_3.h5
```

In [0]:
```
plt.plot(history_1.history['auroc'], 'r')
plt.plot(history_1.history['val_auroc'], 'b')
plt.legend({'Train ROCAUC': 'r', 'Test ROCAUC':'b'})
plt.show()
```

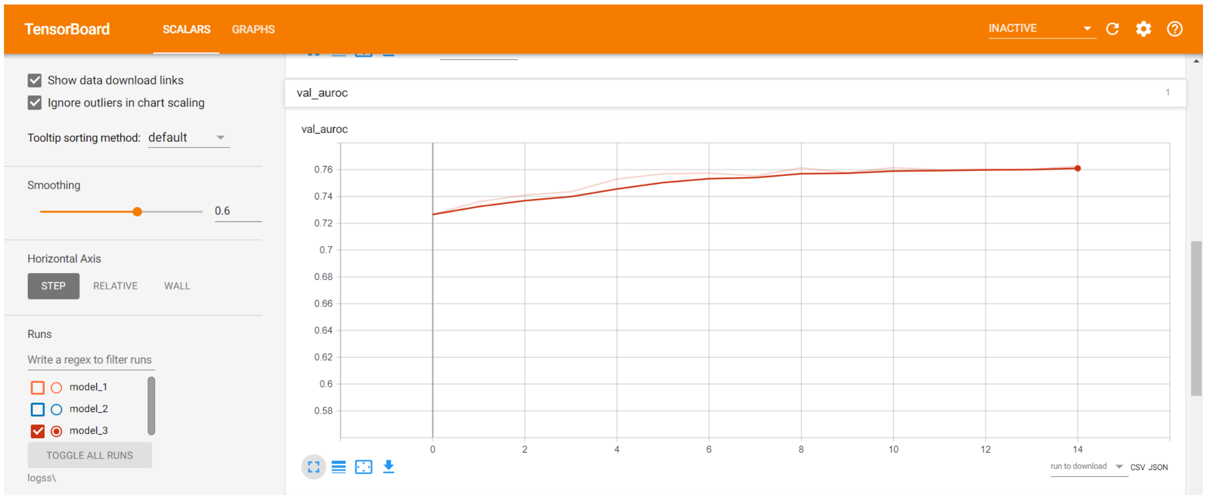


In [0]:
```
plt.plot(history_1.history['loss'], 'r')
plt.plot(history_1.history['val_loss'], 'b')
plt.legend({'Train Loss': 'r', 'Test Loss':'b'})
plt.show()
```



In [5]:
```python
from IPython.display import Image
Image("model3auc.png")
```

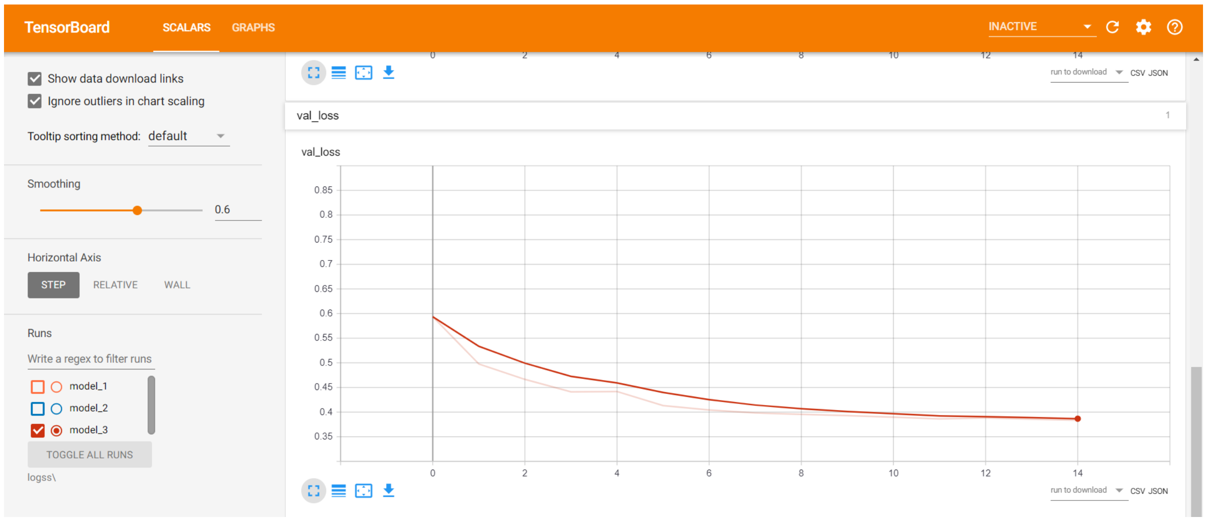
Out[5]:



☑ 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
☐ model_1
☐ model_2
☑ model_3
logss\

In [6]:
```python
from IPython.display import Image
Image("model3loss.png")
```

Out[6]:



☑ 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
☐ model_1
☐ model_2
☑ model_3
logss\

Finally For Model-3 , i'm able to get above 0.76 which is best among all the models

In [0]:

In [7]:
```
from prettytable import PrettyTable
x = PrettyTable()
x.field_names = ["Model","Description", "Test loss", "Test AUC"]
x.add_row(["1","On all text data","0.4266", "0.7545"])
x.add_row(["2","On filtered data by idf values","0.4205", "0.7310"])
x.add_row(["3","On filtered data by idf values with ConvNets","0.3834", "0.762
5"])
print(x)
```

```
+-------+----------------------------------------------+-----------+---------
-+
| Model |                 Description                  | Test loss | Test AUC
|
+-------+----------------------------------------------+-----------+---------
-+
|   1   |               On all text data               |   0.4266  |  0.7545
|
|   2   |        On filtered data by idf values        |   0.4205  |  0.7310
|
|   3   | On filtered data by idf values with ConvNets |   0.3834  |  0.7625
|
+-------+----------------------------------------------+-----------+---------
-+
```

In [0]: