In [0]:

```
%matplotlib inline
import warnings
warnings.filterwarnings("ignore")
import pandas as pd
import numpy as np
import nltk
import string
import matplotlib.pyplot as plt
%matplotlib inline
import seaborn as sns
import matplotlib.pyplot as plt
from sklearn import model_selection
from sklearn.model_selection import train_test_split
```

In [72]:

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

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

In [0]:

```
# Getting data into a dataframe
path="/content/drive/My Drive/Colab_Notebooks/ass14/preprocessed_data.csv"
df = pd.read_csv(path)
```

In [49]:

```
df.head(5)
```

Out[49]:

	school_state	teacher_prefix	project_grade_category	teacher_number_of_previously_posted_pro
0	ca	mrs	grades_prek_2	
1	ut	ms	grades_3_5	
2	ca	mrs	grades_prek_2	
3	ga	mrs	grades_prek_2	
4	wa	mrs	grades_3_5	

In [50]:

```
df.columns
```

Out[50]:

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

In [51]:

```
df.shape
```

Out[51]:

```
(109248, 9)
```

In [0]:

```
#df = df.sample(n=40000)
#project_data=project_data.tail(1000)
#project_data.shape
```

Spliting dataset

In [52]:
```
y=df['project_is_approved']
y.shape
```

Out[52]:
```
(109248,)
```

In [53]:
```
features = df.drop(["project_is_approved"],axis=1)
features.shape
```

Out[53]:
```
(109248, 8)
```

In [0]:
```
#https://scikit-learn.org/stable/modules/generated/sklearn.model_selection.tra
in_test_split.html

#split the data into train and test fo bag of words

x_train,x_test,y_train,y_test=model_selection.train_test_split(features,y,test
_size=0.33,stratify=y,random_state=0)
#split train into cross val train and cross val test
#x_train,x_cv,y_train,y_cv=model_selection.train_test_split(x_t,y_t,test_size=
0.3,random_state=0)
```

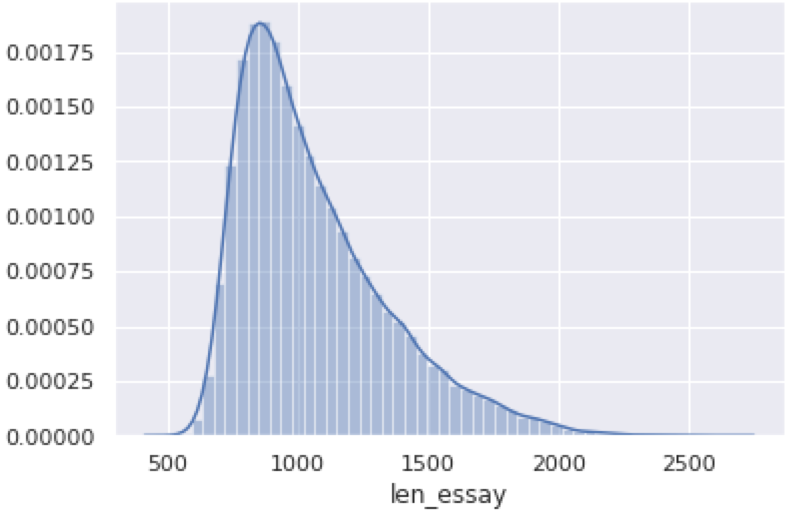
In [55]:
```
print(x_train.shape)
print("+++++++++++++")
print(x_test.shape)
```

```
(73196, 8)
+++++++++++++
(36052, 8)
```

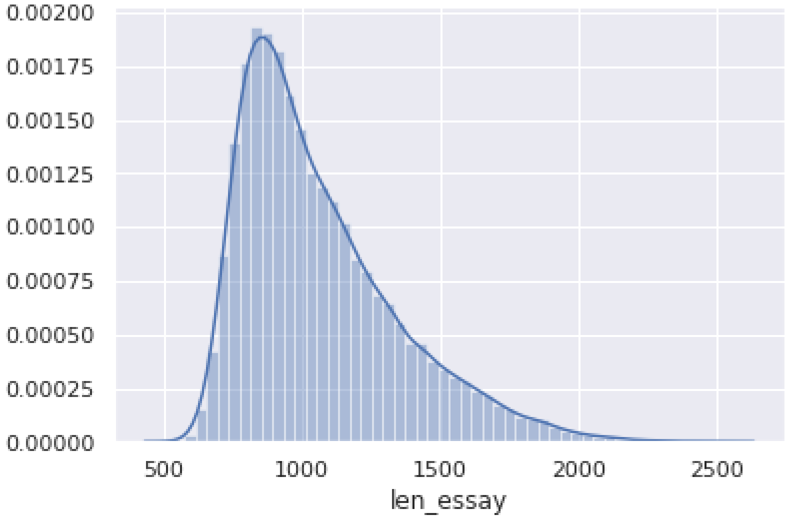
In [0]:
```
# Preparing Text Data As per Our Model
x_train["len_essay"] = x_train["essay"].apply(len)
x_test["len_essay"] = x_test["essay"].apply(len)
```

Distribution plot of essay dataset

In [57]:
```
sns.set()
ax = sns.distplot(x_train["len_essay"])
```



In [59]:
```
ax = sns.distplot(x_test["len_essay"])
```



In [0]:
```
from sklearn.feature_extraction.text import CountVectorizer
from nltk.stem.porter import PorterStemmer
import 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
```

In [0]:

```
# convert the sentences (strings) into integers
from keras.preprocessing.text import Tokenizer
tokenizer = Tokenizer(num_words=5000)
tokenizer.fit_on_texts(x_train["essay"].tolist())
sequences_train = tokenizer.texts_to_sequences(x_train["essay"])
sequences_test = tokenizer.texts_to_sequences(x_test["essay"])
```

In [61]:

```
sequences_train
```

Out[61]:
```
[[25,
  6,
  1759,
  30,
  1008,
  16,
  3225,
  12,
  1,
  24,
  802,
  1729,
  3,
  9,
  144,
  19,
  1196,
  1786,
  436,
  645,
  3126,
  46,
  1082,
  762,
  39,
  90,
  12,
  1,
  512,
  4302,
  807,
  191,
  1047,
  108,
  7,
  3786,
  1297,
  277,
  437,
  1,
  333,
  259,
  2707,
  205,
  44,
  36,
  59,
  597,
  568,
  2721,
  6,
  1698,
  129,
  1842,
  146,
  12,
  1,
  133,
  2505,
  322,
  3231,
  6,
  597,
  9,
  504,
  2118,
  3538,
  2201,
  1351,
  2235,
  1837,
  258,
  597,
  139,
  179,
  333,
  587,
  385,
  66,
  10,
  376,
  2,
  491,
  321,
  1,
  239,
  2235,
  1659,
  78,
  16,
  2,
  24,
  491,
  110,
  631,
  1766,
  76,
  635,
  2,
  950,
  11,
  779,
  1,
  33,
  137,
  826,
  398,
  587,
  4165,
  3181,
  232,
  51,
  547,
  1,
             100,
             19,
             3,
             2,
             301,
             34,
             737,
             518,
             61,
             981,
             631,
             2,
             15,
             117,
             980,
             31,
             148,
             117,
             360,
             483,
             9,
             100,
             247,
             936,
             2,
             34,
             83,
             313,
             58,
             2,
             146,
             3,
             556,
             748,
             158,
             2,
             19,
             432,
             2,
             19,
             1,
             199,
             214,
             182,
             45,
             262,
             37,
             58,
             2,
             8,
             723,
             1073,
             86,
             15,
             6,
             54,
             924,
             677,
             38,
             270,
             232,
             587,
             516,
             5,
             85,
             2,
             15,
             2806,
             942,
             583,
             1,
             8,
             40,
             4521,
             2011,
             1572,
             2,
             15,
             879,
             929,
             650,
             80,
             531,
             264,
             2,
             15,
             1056,
             2,
             603,
             173,
             453,
             1053,
             536,
             40,
             1,
             375,
             2,
             15,
             2868,
             825,
             117,
             1,
             360,
             1659,
             432,
             66,
             766,
             398,
             89,
             2589,
             2,
             16,
             50,
             131,
             3,
             1792,
             2360,
             1011,
             40,
             63,
             54,
             182,
             113,
             33,
             411,
             385,
             182,
             113,
             33,
             385,
             6,
             13],
            ...]
```

In [62]:

```
print("No. of datapoints in X_train :",len(x_train))
print("No. of datapoints in X_test :",len(x_test))
print("Shape of Y_train :",y_train.shape)
print("Shape of Y_test :",y_test.shape)
```

```
No. of datapoints in X_train : 73196
No. of datapoints in X_test : 36052
Shape of Y_train : (73196,)
Shape of Y_test : (36052,)
```

In [63]:

```
# get word -> integer mapping
word2idx = tokenizer.word_index
print('Found %s unique tokens.' % len(word2idx))
```

```
Found 48499 unique tokens.
```

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, SpatialDropout1D, 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, ReduceLROnPlateau
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
```

In [65]:

```
# truncate and/or pad input sequences
max_review_length = 800
encoded_train = pad_sequences(sequences_train,maxlen=max_review_length,padding
='post', truncating='post')
encoded_test = pad_sequences(sequences_test, maxlen=max_review_length,padding=
'post', truncating='post')
print('Shape of train data tensor:', encoded_train.shape)
print('Shape of test data tensor:', encoded_test.shape)

print(encoded_train[1])
```

```
Shape of train data tensor: (73196, 800)
Shape of test data tensor: (36052, 800)
[  25    3 4207 2571 1685 2196 2674   25    3 4755 1537 4485 1143 1569
 2196   25    1   21  133  174 4457  645  780  160  271  307  323   40
  160  271   14  968  298 2196    8 2124  294    3  968  368   25 1616
   15 1543  780 2166   14   93  780 2166 2172 1234   11   40 4457  190
 4423 4715  651    3  651  190 4356   25    3 4224 2571 1685 1650    3
  968  209  780  298   25    1   99  960  239    3   73 4976    9   28
  190 4423 4715  651  228  699 4976 1829  163    1   16 3293  170  190
   25    1 2975   16   40  271 4457  862  767   48    9 3907   15 2197
 2658 1543   37  968  996  581   50    3   13    0    0    0    0    0
    0    0    0    0    0    0    0    0    0    0    0    0    0    0
    0    0    0    0    0    0    0    0    0    0    0    0    0    0
    0    0    0    0    0    0    0    0    0    0    0    0    0    0
    0    0    0    0    0    0    0    0    0    0    0    0    0    0
    0    0    0    0    0    0    0    0    0    0    0    0    0    0
    0    0    0    0    0    0    0    0    0    0    0    0    0    0
    0    0    0    0    0    0    0    0    0    0    0    0    0    0
    0    0    0    0    0    0    0    0    0    0    0    0    0    0
    0    0    0    0    0    0    0    0    0    0    0    0    0    0
    0    0    0    0    0    0    0    0    0    0    0    0    0    0
    0    0    0    0    0    0    0    0    0    0    0    0    0    0
    0    0    0    0    0    0    0    0    0    0    0    0    0    0
    0    0    0    0    0    0    0    0    0    0    0    0    0    0
    0    0    0    0    0    0    0    0    0    0    0    0    0    0
    0    0    0    0    0    0    0    0    0    0    0    0    0    0
    0    0    0    0    0    0    0    0    0    0    0    0    0    0
    0    0    0    0    0    0    0    0    0    0    0    0    0    0
    0    0    0    0    0    0    0    0    0    0    0    0    0    0
    0    0    0    0    0    0    0    0    0    0    0    0    0    0
    0    0    0    0    0    0    0    0    0    0    0    0    0    0
    0    0    0    0    0    0    0    0    0    0    0    0    0    0
    0    0    0    0    0    0    0    0    0    0    0    0    0    0
    0    0    0    0    0    0    0    0    0    0    0    0    0    0
    0    0    0    0    0    0    0    0    0    0    0    0    0    0
    0    0    0    0    0    0    0    0    0    0    0    0    0    0
    0    0    0    0    0    0    0    0    0    0    0    0    0    0
    0    0    0    0    0    0    0    0    0    0    0    0    0    0
    0    0    0    0    0    0    0    0    0    0    0    0    0    0
    0    0    0    0    0    0    0    0    0    0    0    0    0    0
    0    0    0    0    0    0    0    0    0    0    0    0    0    0
    0    0    0    0    0    0    0    0    0    0    0    0    0    0
    0    0    0    0    0    0    0    0    0    0    0    0    0    0
    0    0    0    0    0    0    0    0    0    0    0    0    0    0
    0    0    0    0    0    0    0    0    0    0    0    0    0    0
    0    0    0    0    0    0    0    0    0    0    0    0    0    0
    0    0    0    0    0    0    0    0    0    0    0    0    0    0
    0    0    0    0    0    0    0    0    0    0    0    0    0    0
    0    0    0    0    0    0    0    0    0    0    0    0    0    0
    0    0    0    0    0    0    0    0    0    0    0    0    0    0
    0    0    0    0    0    0    0    0    0    0    0    0    0    0
    0    0    0    0    0    0    0    0    0    0    0    0    0    0
    0    0    0    0    0    0    0    0    0    0    0    0    0    0
    0    0    0    0    0    0    0    0    0    0    0    0    0    0
    0    0    0    0    0    0    0    0    0    0    0    0    0    0
    0    0    0    0    0    0    0    0    0    0    0    0    0    0
    0    0    0    0    0    0    0    0    0    0    0    0    0    0
    0    0    0    0    0    0    0    0    0    0    0    0    0    0
    0    0    0    0    0    0    0    0    0    0    0    0    0    0
    0    0]
```

In [0]:

```
# Loading Embedding File
pickle_in = open("glove_vectors","rb")
glove_words = pickle.load(pickle_in)
```

In [77]:

```
MAX_VOCAB_SIZE=5000
num_words = min(MAX_VOCAB_SIZE, len(word2idx) + 1)
embedding_matrix = np.zeros((num_words, 300))
for word, i in word2idx.items():
  if i < MAX_VOCAB_SIZE:
    embedding_vector = glove_words.get(word)
    if embedding_vector is not None:
      # words not found in embedding index will be all zeros.
      embedding_matrix[i] = embedding_vector


print(num_words)
print("+++++++++")
print(embedding_matrix.shape)
```

```
5000
+++++++++
(5000, 300)
```

In [78]:

```
# Load pre-trained word embeddings into an Embedding layer
# note that we set trainable = False so as to keep the embeddings fixed
MAX_SEQUENCE_LENGTH=800
embedding_layer = Embedding(
  num_words,
  300,
  weights=[embedding_matrix],
  input_length=MAX_SEQUENCE_LENGTH,
  trainable=False
)
input_text = Input(shape=(MAX_SEQUENCE_LENGTH,),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) # dropout=0.5
# x = SpatialDropout1D(0.5)(x)
flatten_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`.
```

caregorical variable

In [79]:

```
# Now will prepare all the remaining categorical features
# Teacher Prefix
no_of_unique_prefix  = x_train["teacher_prefix"].nunique()
embedding_size_prefix = int(min(np.ceil((no_of_unique_prefix)/2), 50 ))
print('Unique Categories:', no_of_unique_prefix,'Embedding Size:', embedding_size_prefix)


# Defining Input and Embedding Layer for the same

input_prefix = Input(shape=(1,),name="teacher_prefix")
embedding_prefix = Embedding(no_of_unique_prefix,embedding_size_prefix,name="emb_pre",trainable=True)(input_prefix)
flatten_2 = Flatten()(embedding_prefix)

lb = LabelEncoder()
encoder_prefix_train = lb.fit_transform(x_train["teacher_prefix"])
# encoder_prefix_cv = lb.transform(X_cv["teacher_prefix"])
encoder_prefix_test = lb.transform(x_test["teacher_prefix"])
```

```
Unique Categories: 5 Embedding Size: 3
```

In [80]:

```
# School State
no_of_unique_state  = x_train["school_state"].nunique()
embedding_size_state= int(min(np.ceil((no_of_unique_state)/2), 50 ))
print('Unique Categories:', no_of_unique_state,'Embedding Size:', embedding_size_state)


# Defining Input and Embedding Layer for the same

input_state = Input(shape=(1,),name="school_prefix")
embedding_state = Embedding(no_of_unique_state,embedding_size_state,name="emb_state",trainable=True)(input_state)
flatten_3 = Flatten()(embedding_state)


encoder_state_train = lb.fit_transform(x_train["school_state"])
# encoder_state_cv = lb.transform(X_cv["school_state"])
encoder_state_test = lb.transform(x_test["school_state"])
```

```
Unique Categories: 51 Embedding Size: 26
```

In [81]:

```python
# For project_grade_category
no_of_unique_grade  = x_train["project_grade_category"].nunique()
embedding_size_grade = int(min(np.ceil((no_of_unique_grade)/2), 50 ))
print('Unique Categories:', no_of_unique_grade,'Embedding Size:', embedding_size_grade)


# Defining Input and Embedding Layer for the same

input_grade= Input(shape=(1,),name="grade_cat")
embedding_grade = Embedding(no_of_unique_grade,embedding_size_grade,name="emb_grade",trainable=True)(input_grade)
flatten_4 = Flatten()(embedding_grade)


encoder_grade_train = lb.fit_transform(x_train["project_grade_category"])
# encoder_grade_cv = lb.transform(X_cv["project_grade_category"])
encoder_grade_test = lb.transform(x_test["project_grade_category"])
```

```
Unique Categories: 4 Embedding Size: 2
```

In [82]:

```python
# For clean_categories
no_of_unique_subcat  = x_train["clean_categories"].nunique()
embedding_size_subcat = int(min(np.ceil((no_of_unique_subcat)/2), 50 ))
print('Unique Categories:', no_of_unique_subcat,'Embedding Size:', embedding_size_subcat)


# Defining Input and Embedding Layer for the same

input_subcat= Input(shape=(1,),name="sub_cat")
embedding_subcat = Embedding(no_of_unique_subcat,embedding_size_subcat,name="emb_subcat",trainable=True)(input_subcat)
flatten_5 = Flatten()(embedding_subcat)


# encoder_subcat_train = lb.fit_transform(x_train["clean_categories"])
# encoder_subcat_cv = lb.transform(X_cv["clean_categories"])
# encoder_subcat_test = lb.transform(x_test["clean_categories"])
le = LabelEncoder()
le.fit(x_train["clean_categories"])
x_test["clean_categories"] = x_test["clean_categories"].map(lambda s: '<unknown>' if s not in le.classes_ else s)
# X_cv["clean_categories"] = X_cv["clean_categories"].map(lambda s: '<unknown>' if s not in le.classes_ else s)
le.classes_ = np.append(le.classes_, '<unknown>')
encoder_subcat_train = le.transform(x_train["clean_categories"])
encoder_subcat_test= le.transform(x_test["clean_categories"])
# encoder_subcat_cv = le.transform(X_cv["clean_categories"])
```

```
Unique Categories: 51 Embedding Size: 26
```

In [83]:

```
# For clean_subcategories
no_of_unique_subcat_1  = x_train["clean_subcategories"].nunique()
embedding_size_subcat_1 = int(min(np.ceil((no_of_unique_subcat_1)/2), 50 ))
print('Unique Categories:', no_of_unique_subcat_1,'Embedding Size:', embedding_size_subcat_1)

# Defining Input and Embedding Layer for the same

input_subcat_1= Input(shape=(1,),name="sub_cat_1")
embedding_subcat_1 = Embedding(no_of_unique_subcat_1+1,embedding_size_subcat_1,name="emb_subcat_1",trainable=True)(input_subcat_1)#adding +1
flatten_6 = Flatten()(embedding_subcat_1)


le = LabelEncoder()
le.fit(x_train["clean_subcategories"])
x_test["clean_subcategories"] = x_test["clean_subcategories"].map(lambda s: '<unknown>' if s not in le.classes_ else s)
# X_cv["clean_subcategories"] = X_cv["clean_subcategories"].map(lambda s: '<unknown>' if s not in le.classes_ else s)
le.classes_ = np.append(le.classes_, '<unknown>')
encoder_subcat_1_train = le.transform(x_train["clean_subcategories"])
encoder_subcat_1_test= le.transform(x_test["clean_subcategories"])
# encoder_subcat_1_cv = le.transform(X_cv["clean_subcategories"])
```

```
Unique Categories: 390 Embedding Size: 50
```

numerical data

In [0]:

```
# Now we will prepare numerical features for our model
num_train_1=x_train['len_essay'].values.reshape(-1, 1)
num_train_2=x_train['price'].values.reshape(-1, 1)
num_train_3=x_train['teacher_number_of_previously_posted_projects'].values.reshape(-1, 1)

num_test_1=x_test['len_essay'].values.reshape(-1, 1)
num_test_2=x_test['price'].values.reshape(-1, 1)
num_test_3=x_test['teacher_number_of_previously_posted_projects'].values.reshape(-1, 1)


num_train=np.concatenate((num_train_1,num_train_2,num_train_3),axis=1)

num_test=np.concatenate((num_test_1,num_test_2,num_test_3),axis=1)
```

In [0]:

```
from sklearn.preprocessing import StandardScaler
norm=StandardScaler()
norm_train=norm.fit_transform(num_train)
norm_test=norm.transform(num_test)
```

In [0]:

```
# Defining the Input and Embedding Layer for the same
num_feats = Input(shape=(3,),name="numerical_features")
num_feats_ = Dense(100,activation="relu",kernel_initializer="he_normal",kernel
_regularizer=regularizers.l2(0.001))(num_feats)
```

In [0]:

```
x_concatenate = concatenate([flatten_1,flatten_2,flatten_3,flatten_4,flatten_5
,flatten_6,num_feats_])
```

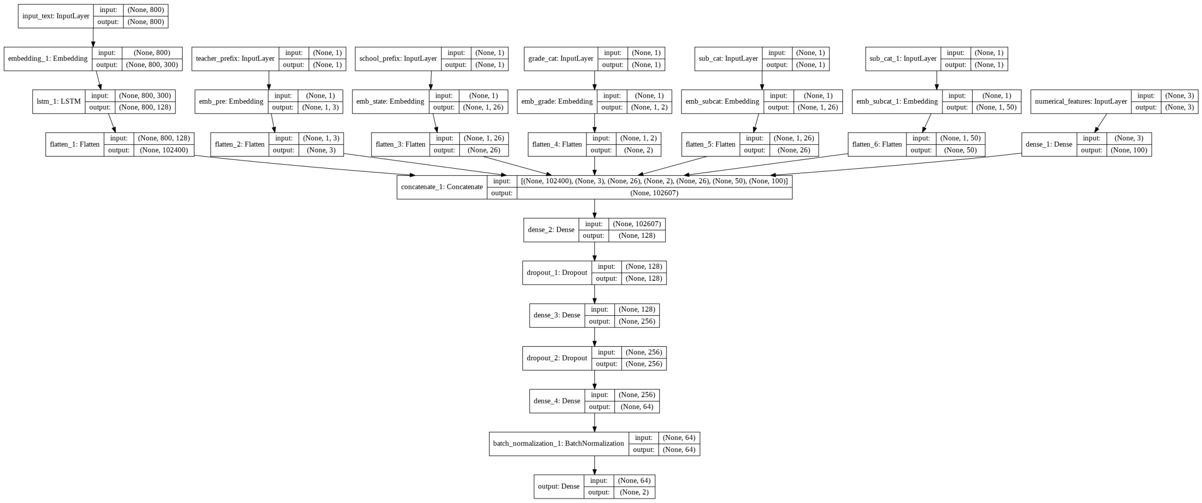
In [88]:

```
print("Building Model-1")

# x_concatenate = BatchNormalization()(x_concatenate)
x = Dense(128,activation="relu", kernel_initializer="he_normal",kernel_regular
izer=regularizers.l2(0.001))(x_concatenate)
# x=LeakyReLU(alpha=0.3)(x)
x=Dropout(0.5)(x)
x = Dense(256,activation="relu",kernel_initializer="he_normal",kernel_regulari
zer=regularizers.l2(0.001))(x)
# x=LeakyReLU(alpha=0.3)(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)
# x=LeakyReLU(alpha=0.3)(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,num_feats],outputs=[output
])
```

```
Building Model-1
```

In [89]:
```
# https://github.com/mmortazavi/EntityEmbedding-Working_Example/blob/master/EntityEmbedding.ipynb
#https://stackoverflow.com/questions/36886711/keras-runtimeerror-failed-to-import-pydot-after-installing-graphviz-and-pyd
from keras.utils import plot_model
import keras
import pydotplus
from keras.utils.vis_utils import model_to_dot
#keras.utils.vis_utils.pydot = pydot

#import pydot_ng as pydot
plot_model(model_1, show_shapes=True, show_layer_names=True, to_file='model_1.png')
from IPython.display import Image
Image(retina=True, filename='model_1.png')
```

Out[89]:



input_text: InputLayer | input: (None, 800) | output: (None, 800)
embedding_1: Embedding | input: (None, 800) | output: (None, 800, 300)
teacher_prefix: InputLayer | input: (None, 1) | output: (None, 1)
school_prefix: InputLayer | input: (None, 1) | output: (None, 1)
grade_cat: InputLayer | input: (None, 1) | output: (None, 1)
sub_cat: InputLayer | input: (None, 1) | output: (None, 1)
sub_cat_1: InputLayer | input: (None, 1) | output: (None, 1)
lstm_1: LSTM | input: (None, 800, 300) | output: (None, 800, 128)
emb_pre: Embedding | input: (None, 1) | output: (None, 1, 3)
emb_state: Embedding | input: (None, 1) | output: (None, 1, 26)
emb_grade: Embedding | input: (None, 1) | output: (None, 1, 2)
emb_subcat: Embedding | input: (None, 1) | output: (None, 1, 26)
emb_subcat_1: Embedding | input: (None, 1) | output: (None, 1, 50)
numerical_features: InputLayer | input: (None, 3) | output: (None, 3)
flatten_1: Flatten | input: (None, 800, 128) | output: (None, 102400)
flatten_2: Flatten | input: (None, 1, 3) | output: (None, 3)
flatten_3: Flatten | input: (None, 1, 26) | output: (None, 26)
flatten_4: Flatten | input: (None, 1, 2) | output: (None, 2)
flatten_5: Flatten | input: (None, 1, 26) | output: (None, 26)
flatten_6: Flatten | input: (None, 1, 50) | output: (None, 50)
dense_1: Dense | input: (None, 3) | output: (None, 100)
concatenate_1: Concatenate | input: [(None, 102400), (None, 3), (None, 26), (None, 2), (None, 26), (None, 50), (None, 100)] | output: (None, 102607)
dense_2: Dense | input: (None, 102607) | output: (None, 128)
dropout_1: Dropout | input: (None, 128) | output: (None, 128)
dense_3: Dense | input: (None, 128) | output: (None, 256)
dropout_2: Dropout | input: (None, 256) | output: (None, 256)
dense_4: Dense | input: (None, 256) | output: (None, 64)
batch_normalization_1: BatchNormalization | input: (None, 64) | output: (None, 64)
output: Dense | input: (None, 64) | output: (None, 2)

In [0]:
```
train_data_1 = [encoded_train,encoder_prefix_train,encoder_state_train,
              encoder_grade_train,encoder_subcat_train,encoder_subcat_1_train,
norm_train]
test_data_1 = [encoded_test,encoder_prefix_test,encoder_state_test,encoder_grade_test,
             encoder_subcat_test,encoder_subcat_1_test,norm_test]

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]:

```
checkpoint_1 = ModelCheckpoint("model_1.h5",
                             monitor="val_loss",
                             mode="min",
                             save_best_only = True,
                             verbose=1)

earlystop_1 = EarlyStopping(monitor = 'val_loss',
                            mode="min",
                            min_delta = 0,
                            patience = 2,
                            verbose = 1,
                            restore_best_weights = True)

reduce_lr_1 = ReduceLROnPlateau(monitor = 'val_loss', factor = 0.2, patience =
1, verbose = 1, min_delta = 0.0001)

tensorboard_1 = TensorBoard(log_dir='graph_1', histogram_freq=0, batch_size=51
2, write_graph=True, write_grads=False, write_images=False, embeddings_freq=0,
embeddings_layer_names=None, embeddings_metadata=None, embeddings_data=None, u
pdate_freq='epoch')

callbacks_1 = [checkpoint_1,earlystop_1,tensorboard_1,reduce_lr_1]
```

In [0]:

```
# Defining Custom ROC-AUC Metrics
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]:

```
adam = Adam(lr=0.001, beta_1=0.9, beta_2=0.999, epsilon=None, decay=0.0, amsgr
ad=False)
```

In [94]:

```
model_1.compile(optimizer=adam, loss='categorical_crossentropy', metrics=[auro
c])
```

```
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-92-a7e6cba44e56>:10: py_func (from ten
sorflow.python.ops.script_ops) is deprecated and will be removed in a future
version.
Instructions for updating:
tf.py_func is deprecated in TF V2. Instead, there are two
    options available in V2.
    - tf.py_function takes a python function which manipulates tf eager
    tensors instead of numpy arrays. It's easy to convert a tf eager tensor 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 [54]:

```
history_1 = model_1.fit(train_data_1,Y_train,batch_size=512,
                        epochs=20,validation_data=(test_data_1,Y_test),callbac
ks=callbacks_1)
```

```
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 1/20
73196/73196 [==============================] - 310s 4ms/step - loss: 1.3721 -
auroc: 0.5209 - val_loss: 0.9704 - val_auroc: 0.6374

Epoch 00001: val_loss improved from inf to 0.97041, saving model to model_1.h
5
Epoch 2/20
73196/73196 [==============================] - 308s 4ms/step - loss: 0.7985 -
auroc: 0.6546 - val_loss: 0.7230 - val_auroc: 0.7188

Epoch 00002: val_loss improved from 0.97041 to 0.72301, saving model to model
_1.h5
Epoch 3/20
73196/73196 [==============================] - 307s 4ms/step - loss: 0.6342 -
auroc: 0.7148 - val_loss: 0.5958 - val_auroc: 0.7400

Epoch 00003: val_loss improved from 0.72301 to 0.59585, saving model to model
_1.h5
Epoch 4/20
73196/73196 [==============================] - 307s 4ms/step - loss: 0.5477 -
auroc: 0.7338 - val_loss: 0.5348 - val_auroc: 0.7468

Epoch 00004: val_loss improved from 0.59585 to 0.53484, saving model to model
_1.h5
Epoch 5/20
73196/73196 [==============================] - 307s 4ms/step - loss: 0.4991 -
auroc: 0.7397 - val_loss: 0.4841 - val_auroc: 0.7482

Epoch 00005: val_loss improved from 0.53484 to 0.48410, saving model to model
_1.h5
Epoch 6/20
73196/73196 [==============================] - 307s 4ms/step - loss: 0.4659 -
auroc: 0.7465 - val_loss: 0.4550 - val_auroc: 0.7512

Epoch 00006: val_loss improved from 0.48410 to 0.45502, saving model to model
_1.h5
Epoch 7/20
73196/73196 [==============================] - 305s 4ms/step - loss: 0.4446 -
auroc: 0.7497 - val_loss: 0.4410 - val_auroc: 0.7508

Epoch 00007: val_loss improved from 0.45502 to 0.44101, saving model to model
_1.h5
```

```
Epoch 8/20
73196/73196 [==============================] - 305s 4ms/step - loss: 0.4304 -
auroc: 0.7486 - val_loss: 0.4317 - val_auroc: 0.7484

Epoch 00008: val_loss improved from 0.44101 to 0.43174, saving model to model
_1.h5
Epoch 9/20
73196/73196 [==============================] - 305s 4ms/step - loss: 0.4224 -
auroc: 0.7507 - val_loss: 0.4155 - val_auroc: 0.7521

Epoch 00009: val_loss improved from 0.43174 to 0.41553, saving model to model
_1.h5
Epoch 10/20
73196/73196 [==============================] - 305s 4ms/step - loss: 0.4146 -
auroc: 0.7540 - val_loss: 0.4182 - val_auroc: 0.7510

Epoch 00010: val_loss did not improve from 0.41553

Epoch 00010: ReduceLROnPlateau reducing learning rate to 0.000200000009499490
26.
Epoch 11/20
73196/73196 [==============================] - 304s 4ms/step - loss: 0.3966 -
auroc: 0.7656 - val_loss: 0.4018 - val_auroc: 0.7538

Epoch 00011: val_loss improved from 0.41553 to 0.40177, saving model to model
_1.h5
Epoch 12/20
73196/73196 [==============================] - 304s 4ms/step - loss: 0.3879 -
auroc: 0.7720 - val_loss: 0.4001 - val_auroc: 0.7539

Epoch 00012: val_loss improved from 0.40177 to 0.40010, saving model to model
_1.h5
Epoch 13/20
73196/73196 [==============================] - 304s 4ms/step - loss: 0.3861 -
auroc: 0.7751 - val_loss: 0.4014 - val_auroc: 0.7520

Epoch 00013: val_loss did not improve from 0.40010

Epoch 00013: ReduceLROnPlateau reducing learning rate to 4.0000001899898055e-
05.
Epoch 14/20
73196/73196 [==============================] - 303s 4ms/step - loss: 0.3778 -
auroc: 0.7859 - val_loss: 0.4003 - val_auroc: 0.7516

Epoch 00014: val_loss did not improve from 0.40010
Restoring model weights from the end of the best epoch

Epoch 00014: ReduceLROnPlateau reducing learning rate to 8.000000525498762e-0
6.
Epoch 00014: early stopping
```

In [55]:

```
print(model_1.summary())
```

```
Model: "model_1"
_____________________________________________________________________________
_____________________
Layer (type)                    Output Shape         Param #     Connected to
=============================================================================
=====================
input_text (InputLayer)         (None, 800)          0
_____________________________________________________________________________
_____________________
embedding_1 (Embedding)         (None, 800, 300)     1500000     input_text
[0][0]
_____________________________________________________________________________
_____________________
teacher_prefix (InputLayer)     (None, 1)            0
_____________________________________________________________________________
_____________________
school_prefix (InputLayer)      (None, 1)            0
_____________________________________________________________________________
_____________________
grade_cat (InputLayer)          (None, 1)            0
_____________________________________________________________________________
_____________________
sub_cat (InputLayer)            (None, 1)            0
_____________________________________________________________________________
_____________________
sub_cat_1 (InputLayer)          (None, 1)            0
_____________________________________________________________________________
_____________________
lstm_1 (LSTM)                   (None, 800, 128)     219648      embedding_1
[0][0]
_____________________________________________________________________________
_____________________
emb_pre (Embedding)             (None, 1, 3)         15          teacher_pref
ix[0][0]
_____________________________________________________________________________
_____________________
emb_state (Embedding)           (None, 1, 26)        1326        school_prefi
x[0][0]
_____________________________________________________________________________
_____________________
emb_grade (Embedding)           (None, 1, 2)         8           grade_cat[0]
[0]
_____________________________________________________________________________
_____________________
emb_subcat (Embedding)          (None, 1, 26)        1326        sub_cat[0]
[0]
_____________________________________________________________________________
_____________________
emb_subcat_1 (Embedding)        (None, 1, 50)        19550       sub_cat_1[0]
[0]
_____________________________________________________________________________
_____________________
numerical_features (InputLayer) (None, 3)            0
_____________________________________________________________________________
_____________________
flatten_1 (Flatten)             (None, 102400)       0           lstm_1[0][0]
_____________________________________________________________________________
```

```
_____________________
flatten_2 (Flatten)             (None, 3)           0           emb_pre[0]
[0]
_____________________________________________________________________________
_____________________
flatten_3 (Flatten)             (None, 26)          0           emb_state[0]
[0]
_____________________________________________________________________________
_____________________
flatten_4 (Flatten)             (None, 2)           0           emb_grade[0]
[0]
_____________________________________________________________________________
_____________________
flatten_5 (Flatten)             (None, 26)          0           emb_subcat
[0][0]
_____________________________________________________________________________
_____________________
flatten_6 (Flatten)             (None, 50)          0           emb_subcat_1
[0][0]
_____________________________________________________________________________
_____________________
dense_1 (Dense)                 (None, 100)         400         numerical_fe
atures[0][0]
_____________________________________________________________________________
_____________________
concatenate_1 (Concatenate)     (None, 102607)      0           flatten_1[0]
[0]
                                                                flatten_2[0]
[0]
                                                                flatten_3[0]
[0]
                                                                flatten_4[0]
[0]
                                                                flatten_5[0]
[0]
                                                                flatten_6[0]
[0]
                                                                dense_1[0]
[0]
_____________________________________________________________________________
_____________________
dense_2 (Dense)                 (None, 128)         13133824    concatenate_
1[0][0]
_____________________________________________________________________________
_____________________
dropout_1 (Dropout)             (None, 128)         0           dense_2[0]
[0]
_____________________________________________________________________________
_____________________
dense_3 (Dense)                 (None, 256)         33024       dropout_1[0]
[0]
_____________________________________________________________________________
_____________________
dropout_2 (Dropout)             (None, 256)         0           dense_3[0]
[0]
_____________________________________________________________________________
_____________________
```

```
dense_4 (Dense)                 (None, 64)           16448       dropout_2[0]
[0]
______________________________________________________________________________
____________________
batch_normalization_1 (BatchNor (None, 64)           256         dense_4[0]
[0]
______________________________________________________________________________
____________________
output (Dense)                  (None, 2)            130         batch_normal
ization_1[0][0]
==============================================================================
====================
Total params: 14,925,955
Trainable params: 13,425,827
Non-trainable params: 1,500,128
______________________________________________________________________________
____________________
None
```

In [0]:

```
my_model = load_model("model_1.h5",custom_objects={"auroc":auroc})
```

In [0]:

```
project_status = {0:"Rejected",1:"Approved"}
```

In [0]:

```
Y_pred = my_model.predict(test_data_1,batch_size=512)
```

In [0]:

```
# took the function from https://nbviewer.jupyter.org/github/pranaya-mathur/Hu
man-Activity-Recognition/blob/master/Human_Activity_Recognition.ipynb
def confusion_matrix(Y_true, Y_pred):
    Y_true = pd.Series([project_status[y] for y in np.argmax(Y_test, axis=1)])
    Y_pred = pd.Series([project_status[y] for y in np.argmax(Y_pred, axis=1)])

    return pd.crosstab(Y_true, Y_pred, rownames=['True'], colnames=['Pred'])
```

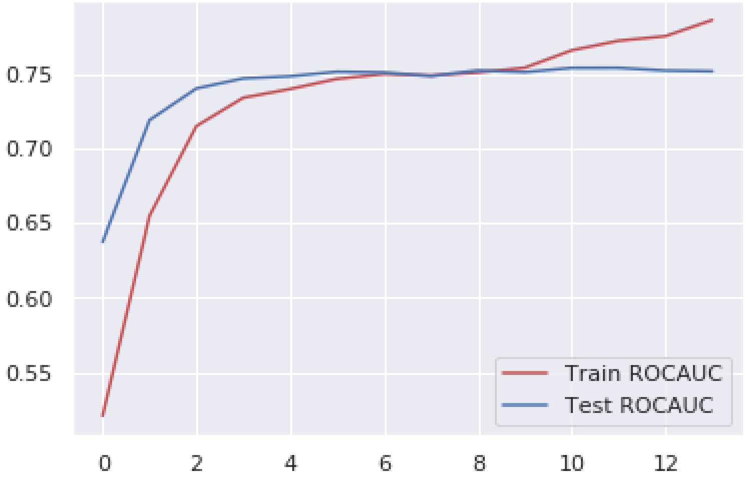
In [60]:

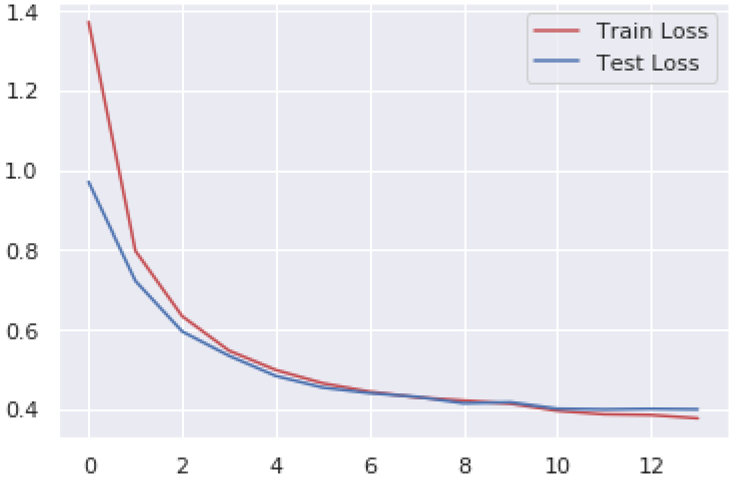
```
results = confusion_matrix(Y_test,Y_pred)
results
```

Out[60]:

Pred	Approved	Rejected
True		
Approved	30162	431
Rejected	4825	634

In [61]:
```
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()


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()
```





Model is overfitting because at end, train AUC(0.786) hs higher than test AUC(0.76)

In [0]:
```
```