Importing Libraries

In [1]:

```
import pandas as pd
import numpy as np
from time import time
from tensorflow.python.keras.callbacks import TensorBoard
import re

#from google.colab import drive
#drive.mount('/content/gdrive')
# Dataset is now stored in a Pandas Dataframe
```

```
C:\ProgramData\Anaconda3\lib\site-packages\h5py\__init__.py:36: FutureWarning: Conversion of the s
econd argument of issubdtype from `float` to `np.floating` is deprecated. In future, it will be
treated as `np.float64 == np.dtype(float).type`.
  from ._conv import register_converters as _register_converters
```

Importing Data

In [2]:

```
data=pd.read_csv('DonorsChoose_LSTM.csv')
```

In [60]:

```
data[101926:101927]
```

Out[60]:

	teacher_prefix	school_state	project_grade_category	teacher_number_of_previously_posted_projects	project_is_
101926	Ms.	FL	Grades 9-12	2	1

In [11]:

```
data['total_text']=data['clean_essay'].map(str) + data['clean_title'].map(str)
```

In [12]:

```
from sklearn.model_selection import train_test_split
from sklearn.metrics import roc_auc_score
train, test = train_test_split(data, random_state=123, shuffle=True, test_size=0.1)
print("Training data shape:", train.shape)
print("Test data shape:", test.shape)
```

```
Training data shape: (98323, 13)
Test data shape: (10925, 13)
```

In [13]:

```
train=train.reset_index()
test=test.reset_index()
```

In [14]:

```
train.head(2)
```

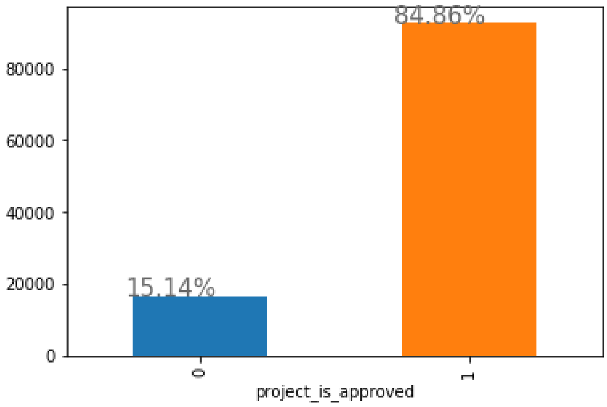
Out[14]:

	index	teacher_prefix	school_state	project_grade_category	teacher_number_of_previously_posted_projects	project_
0	101926	Ms.	FL	Grades 9-12	2	1
1	83948	Mrs.	CA	Grades 3-5	0	1

Distribution of positive and negative data points

In [65]:

```
ax = data.groupby("project_is_approved")["project_is_approved"].count().plot.bar()

# create a list to collect the plt.patches data
totals = []

# find the values and append to list
for i in ax.patches:
    totals.append(i.get_height())

# set individual bar lables using above list
total = sum(totals)

# set individual bar lables using above list
for i in ax.patches:
    # get_x pulls left or right; get_height pushes up or down
    ax.text(i.get_x()-.03, i.get_height()+.5, \
            str(round((i.get_height()/total)*100, 2))+'%', fontsize=15,
                color='dimgrey')
```



Importing pre trained Glove vectors

In [3]:

```
import io
import pickle
with io.open('glove_vectors.txt', 'rb') as f:
    glove_model = pickle.load(f)
    glove_words =  set(glove_model.keys())
```

Preprocessing with Keras tokenizer

In [8]:

```
from keras.preprocessing.text import Tokenizer
```

```
Using TensorFlow backend.
```

In [9]:

```
token = Tokenizer()

token.fit_on_texts(train['total_text'])
train['total_text']=token.texts_to_sequences(train['total_text'])
test['total_text']=token.texts_to_sequences(test['total_text'])
text_size = len(token.word_index) + 1

# create a weight matrix for words in training docs --code copied from
https://machinelearningmastery.com/use-word-embedding-layers-deep-learning-keras/
embedding_matrix = np.zeros((text_size, 300))
for word, i in token.word_index.items():
    embedding_vector = glove_model.get(word)
    if embedding_vector is not None:
        embedding_matrix[i] = embedding_vector

token = Tokenizer()

token.fit_on_texts(train['clean_categories'])
train['clean_categories']=token.texts_to_sequences(train['clean_categories'])
test['clean_categories']=token.texts_to_sequences(test['clean_categories'])
category_size = len(token.word_index) + 1

token = Tokenizer()

token.fit_on_texts(train['teacher_prefix'])
train['teacher_prefix']=token.texts_to_sequences(train['teacher_prefix'])
test['teacher_prefix']=token.texts_to_sequences(test['teacher_prefix'])
prefix_size = len(token.word_index) + 1

token = Tokenizer()

token.fit_on_texts(train['school_state'])
train['school_state']=token.texts_to_sequences(train['school_state'])
test['school_state']=token.texts_to_sequences(test['school_state'])
state_size = len(token.word_index) + 1

token = Tokenizer()

token.fit_on_texts(train['project_grade_category'])
train['project_grade_category']=token.texts_to_sequences(train['project_grade_category'])
test['project_grade_category']=token.texts_to_sequences(test['project_grade_category'])
grade_size = len(token.word_index) + 1

token = Tokenizer()

token.fit_on_texts(train['clean_subcategories'])
train['clean_subcategories']=token.texts_to_sequences(train['clean_subcategories'])
test['clean_subcategories']=token.texts_to_sequences(test['clean_subcategories'])
subcategory_size = len(token.word_index) + 1
```

In [74]:

```
embedding_matrix.shape
```

Out[74]:

```
(67062, 300)
```

In [75]:

```
train.drop(['clean_essay','clean_title','index'] , axis=1, inplace=True)
test.drop(['clean_essay','clean_title','index'] , axis=1, inplace=True)
```

In [10]:

```
train.tail(2)
```

Out[10]:

	index	teacher_prefix	school_state	project_grade_category	teacher_number_of_previously_posted_projects	proje
98321	28030	[2]	[2]	[1, 2, 3]	1	1
98322	15725	[2]	[18]	[1, 2, 3]	1	0

Getting data in the form of Dictionary which then will be given as a input to Deep Learning Models

In [13]:

```
from keras.preprocessing.sequence import pad_sequences
from numpy import array
```

In [78]:

```
X_train={}

X_train["posted_projects"]= array(train["teacher_number_of_previously_posted_projects"]).reshape(l
en(train),1)
X_train["price"]= array(train["price"]).reshape(len(train),1)
X_train["quantity"]= array(train["quantity"]).reshape(len(train),1)
X_train["Is_digit_present"]= array(train["Is_digit_present"]).reshape(len(train),1)

X_train["teacher_prefix"] = pad_sequences(train["teacher_prefix"], maxlen=1)
X_train["school_state"] = pad_sequences(train["school_state"], maxlen=1)
X_train["project_grade_category"] = pad_sequences(train["project_grade_category"], maxlen=3)

X_train["total_text"] = pad_sequences(train["total_text"], maxlen=300)
X_train["clean_categories"] = pad_sequences(train["clean_categories"], maxlen=4)
X_train["clean_subcategories"] = pad_sequences(train["clean_subcategories"], maxlen=4)

X_train["output"]= array(train["project_is_approved"])
```

In [79]:

```
X_test={}

X_test["posted_projects"]= array(test["teacher_number_of_previously_posted_projects"]).reshape(len
(test),1)
X_test["price"]= array(test["price"]).reshape(len(test),1)
X_test["quantity"]= array(test["quantity"]).reshape(len(test),1)
X_test["Is_digit_present"]= array(test["Is_digit_present"]).reshape(len(test),1)

X_test["teacher_prefix"]= pad_sequences(test["teacher_prefix"], maxlen=1)
X_test["school_state"]= pad_sequences(test["school_state"], maxlen=1)
X_test["project_grade_category"]= pad_sequences(test["project_grade_category"], maxlen=3)

X_test["total_text"]= pad_sequences(test["total_text"], maxlen=300)
X_test["clean_categories"]= pad_sequences(test["clean_categories"], maxlen=4)
X_test["clean_subcategories"]= pad_sequences(test["clean_subcategories"], maxlen=4)
```

```
X_test["output"]= array(test["project_is_approved"])
```

In [14]:

```
from keras.initializers import he_normal
from keras.models import Model
from keras.optimizers import Adam
from keras.layers import Input, BatchNormalization, Embedding, LSTM, Flatten, concatenate, Dense, D
ropout
#import tensorflow as tf
#sess = tf.Session()

#from keras import backend as K
#K.set_session(sess)
```

In [80]:

```
import platform
print(platform.python_version())
```

```
3.6.8
```

Model 1

In [107]:

```
    # Input layers
previously_posted_projects = Input(shape=(1,), name="posted_projects")
price = Input(shape=(1,), name="price")
digit_present = Input(shape=(1,), name="Is_digit_present")
quantity = Input(shape=(1,), name="quantity")


school_state = Input(shape=(1,), name="school_state")
teacher_prefix = Input(shape=(1,), name="teacher_prefix")
project_grade= Input(shape=(3,), name="project_grade_category")

total_text = Input(shape=(300,), name="total_text")
clean_categories = Input(shape=(4,), name="clean_categories")
clean_subcategories = Input(shape=(4,), name="clean_subcategories")

    # Batch normalization layer
#previously_posted_projects_bn = BatchNormalization(axis=1, momentum=0.99, epsilon=0.001, center=T
rue, scale=True, beta_initializer='zeros', gamma_initializer='ones',
moving_mean_initializer='zeros', moving_variance_initializer='ones')(previously_posted_projects)
#price_bn = BatchNormalization(axis=1, momentum=0.99, epsilon=0.001, center=True, scale=True, beta
_initializer='zeros', gamma_initializer='ones', moving_mean_initializer='zeros',
moving_variance_initializer='ones')(price)
#quantity_bn = BatchNormalization(axis=1, momentum=0.99, epsilon=0.001, center=True, scale=True, b
eta_initializer='zeros', gamma_initializer='ones', moving_mean_initializer='zeros',
moving_variance_initializer='ones')(quantity)



    # Embedding layers
emb_text_layer = Embedding(text_size, 300, weights=[embedding_matrix],  trainable=False)
emb_category_layer = Embedding(category_size, 4)
emb_subcategory_layer = Embedding(subcategory_size, 4)

emb_state_layer = Embedding(state_size, 8)
emb_prefix_layer = Embedding(prefix_size, 2)
emb_grade_layer = Embedding(grade_size, 2)

    # Giving Input to Embedding layers
emb_text = emb_text_layer(total_text)
emb_category =emb_category_layer(clean_categories)
emb_subcategory =emb_subcategory_layer(clean_subcategories)

emb_state =emb_state_layer(school_state)
emb_prefix =emb_prefix_layer(teacher_prefix)
emb_grade =emb_grade_layer(project_grade)
```

```
    # LSTM layers
lstm_text = LSTM(12, activation="relu", return_sequences=True)(emb_text)

    # Flatten layers
flatten_text =Flatten()(lstm_text)
flatten_category =Flatten()(emb_category)
flatten_subcategory =keras.layers.Flatten()(emb_subcategory)

flatten_state =Flatten()(emb_state)
flatten_prefix =Flatten()(emb_prefix)
flatten_grade =Flatten()(emb_grade)

    # concatenation of all numeric layers
numeric= concatenate([previously_posted_projects,
                                      price,
                                      digit_present,
                                      quantity])
    # Dense layer
dense_numeric =Dense(4, activation='relu',kernel_initializer=he_normal(seed=5))(numeric)

    # Merge all layers into one
x = concatenate([dense_numeric,
                                 flatten_text,
                                 flatten_category,
                                 flatten_subcategory,
                                 flatten_state,
                                 flatten_prefix,
                                 flatten_grade])

dense_x =Dense(8, activation='relu',kernel_initializer=he_normal(seed=3))(x)

dense_x_bn= BatchNormalization(axis=1, momentum=0.99, epsilon=0.001, center=True, scale=True, beta_
initializer='zeros', gamma_initializer='ones', moving_mean_initializer='zeros',
moving_variance_initializer='ones')(dense_x)

#drop_x =Dropout(0.5)(dense_x_bn)

dense2_x =Dense(4, activation='relu',kernel_initializer=he_normal(seed=1))(dense_x_bn)

#drop2_x =Dropout(0.5)(dense2_x)

dense3_x =Dense(2, activation='relu',kernel_initializer=he_normal(seed=2))(dense2_x)

    # Dense layers
    #x = keras.layers.Dense(128, activation="relu")(x)

    # Output layers
output = Dense(1, activation="sigmoid", name='final_output')(dense3_x)



model = Model(inputs=[previously_posted_projects,price,digit_present,quantity,school_state,teacher_
prefix,project_grade,total_text,clean_categories,clean_subcategories], outputs=[output])
model.summary()
```

```
__________________________________________________________________________________________________
Layer (type)                    Output Shape         Param #     Connected to
==================================================================================================
total_text (InputLayer)         (None, 300)          0
__________________________________________________________________________________________________
posted_projects (InputLayer)    (None, 1)            0
__________________________________________________________________________________________________
price (InputLayer)              (None, 1)            0
__________________________________________________________________________________________________
Is_digit_present (InputLayer)   (None, 1)            0
__________________________________________________________________________________________________
quantity (InputLayer)           (None, 1)            0
__________________________________________________________________________________________________
embedding_91 (Embedding)        (None, 300, 300)     20118600    total_text[0][0]
__________________________________________________________________________________________________
clean_categories (InputLayer)   (None, 4)            0
__________________________________________________________________________________________________
clean_subcategories (InputLayer (None, 4)            0
__________________________________________________________________________________________________
school_state (InputLayer)       (None, 1)            0
```

```
__________________________________________________________________________________________________
teacher_prefix (InputLayer)     (None, 1)            0                                            
__________________________________________________________________________________________________
project_grade_category (InputLa (None, 3)            0                                            
__________________________________________________________________________________________________
concatenate_31 (Concatenate)    (None, 4)            0           posted_projects[0][0]            
                                                                 price[0][0]                      
                                                                 Is_digit_present[0][0]           
                                                                 quantity[0][0]                   
__________________________________________________________________________________________________
lstm_16 (LSTM)                  (None, 300, 12)      15024       embedding_91[0][0]               
__________________________________________________________________________________________________
embedding_92 (Embedding)        (None, 4, 4)         64          clean_categories[0][0]           
__________________________________________________________________________________________________
embedding_93 (Embedding)        (None, 4, 4)         152         clean_subcategories[0][0]        
__________________________________________________________________________________________________
embedding_94 (Embedding)        (None, 1, 8)         416         school_state[0][0]               
__________________________________________________________________________________________________
embedding_95 (Embedding)        (None, 1, 2)         12          teacher_prefix[0][0]             
__________________________________________________________________________________________________
embedding_96 (Embedding)        (None, 3, 2)         20          project_grade_category[0][0]     
__________________________________________________________________________________________________
dense_61 (Dense)                (None, 4)            20          concatenate_31[0][0]             
__________________________________________________________________________________________________
flatten_91 (Flatten)            (None, 3600)         0           lstm_16[0][0]                    
__________________________________________________________________________________________________
flatten_92 (Flatten)            (None, 16)           0           embedding_92[0][0]               
__________________________________________________________________________________________________
flatten_93 (Flatten)            (None, 16)           0           embedding_93[0][0]               
__________________________________________________________________________________________________
flatten_94 (Flatten)            (None, 8)            0           embedding_94[0][0]               
__________________________________________________________________________________________________
flatten_95 (Flatten)            (None, 2)            0           embedding_95[0][0]               
__________________________________________________________________________________________________
flatten_96 (Flatten)            (None, 6)            0           embedding_96[0][0]               
__________________________________________________________________________________________________
concatenate_32 (Concatenate)    (None, 3652)         0           dense_61[0][0]                   
                                                                 flatten_91[0][0]                 
                                                                 flatten_92[0][0]                 
                                                                 flatten_93[0][0]                 
                                                                 flatten_94[0][0]                 
                                                                 flatten_95[0][0]                 
                                                                 flatten_96[0][0]                 
__________________________________________________________________________________________________
dense_62 (Dense)                (None, 8)            29224       concatenate_32[0][0]             
__________________________________________________________________________________________________
batch_normalization_44 (BatchNo (None, 8)            32          dense_62[0][0]                   
__________________________________________________________________________________________________
dense_63 (Dense)                (None, 4)            36          batch_normalization_44[0][0]     
__________________________________________________________________________________________________
dense_64 (Dense)                (None, 2)            10          dense_63[0][0]                   
__________________________________________________________________________________________________
final_output (Dense)            (None, 1)            3           dense_64[0][0]                   
==================================================================================================
Total params: 20,163,613
Trainable params: 44,997
Non-trainable params: 20,118,616
__________________________________________________________________________________________________
```

AUC metric Function

In [13]:

```
import tensorflow as tf


def roc_auc(y_true, y_pred):
    auc = tf.metrics.auc(y_true, y_pred,weights=None,num_thresholds=200)[1]
    keras.backend.get_session().run(tf.local_variables_initializer()) # use to reset the local
variables created by tf.metrics.auc
    return auc
```

In [108]:

```
tb =TensorBoard(log_dir="logs/{}".format(time()))

model.compile(optimizer = Adam(lr=1e-2),
                  loss = {'final_output': 'binary_crossentropy'},
                  metrics = [roc_auc])




model.fit({"posted_projects":X_train['posted_projects'], "price":X_train['price'],
              "Is_digit_present":X_train['Is_digit_present'], "quantity":X_train['quantity'],
              "school_state":X_train['school_state'], "teacher_prefix":X_train['teacher_prefix'],
        "project_grade_category":X_train['project_grade_category'], "total_text":X_train['total_tex
t'],
        "clean_categories":X_train['clean_categories'], "clean_subcategories":X_train['clean_subcat
egories']}
              ,{
              "final_output":X_train['output']},
                 batch_size=2500,
                 epochs=40,

              validation_data=({"posted_projects":X_test['posted_projects'], "price":X_test['price'
],
              "Is_digit_present":X_test['Is_digit_present'], "quantity":X_test['quantity'],
              "school_state":X_test['school_state'], "teacher_prefix":X_test['teacher_prefix'],
        "project_grade_category":X_test['project_grade_category'], "total_text":X_test['total_text'
],
        "clean_categories":X_test['clean_categories'], "clean_subcategories":X_test['clean_subcateg
ories']}
              ,{
              "final_output":X_test['output']}),
                callbacks=[tb])
```

```
Train on 98323 samples, validate on 10925 samples
Epoch 1/40
98323/98323 [==============================] - 46s 465us/step - loss: 0.5809 - roc_auc: 0.4886 - v
al_loss: 0.4358 - val_roc_auc: 0.4888
Epoch 2/40
98323/98323 [==============================] - 38s 386us/step - loss: 0.4177 - roc_auc: 0.5093 - v
al_loss: 0.4318 - val_roc_auc: 0.5271
Epoch 3/40
98323/98323 [==============================] - 38s 386us/step - loss: 0.4061 - roc_auc: 0.5437 - v
al_loss: 0.4327 - val_roc_auc: 0.5590
Epoch 4/40
98323/98323 [==============================] - 38s 387us/step - loss: 0.4004 - roc_auc: 0.5727 - v
al_loss: 0.4323 - val_roc_auc: 0.5832
Epoch 5/40
98323/98323 [==============================] - 38s 389us/step - loss: 0.3962 - roc_auc: 0.5928 - v
al_loss: 0.4339 - val_roc_auc: 0.6011
Epoch 6/40
98323/98323 [==============================] - 38s 387us/step - loss: 0.3941 - roc_auc: 0.6087 - v
al_loss: 0.4346 - val_roc_auc: 0.6142
Epoch 7/40
98323/98323 [==============================] - 38s 386us/step - loss: 0.3900 - roc_auc: 0.6202 - v
al_loss: 0.4356 - val_roc_auc: 0.6252
Epoch 8/40
98323/98323 [==============================] - 38s 383us/step - loss: 0.3875 - roc_auc: 0.6303 - v
al_loss: 0.4365 - val_roc_auc: 0.6344
Epoch 9/40
98323/98323 [==============================] - 38s 390us/step - loss: 0.3860 - roc_auc: 0.6386 - v
al_loss: 0.4400 - val_roc_auc: 0.6416
Epoch 10/40
98323/98323 [==============================] - 38s 386us/step - loss: 0.3837 - roc_auc: 0.6454 - v
al_loss: 0.4401 - val_roc_auc: 0.6480
Epoch 11/40
98323/98323 [==============================] - 38s 388us/step - loss: 0.3841 - roc_auc: 0.6508 - v
al_loss: 0.4418 - val_roc_auc: 0.6531
Epoch 12/40
98323/98323 [==============================] - 38s 388us/step - loss: 0.3827 - roc_auc: 0.6557 - v
al_loss: 0.4435 - val_roc_auc: 0.6574
Epoch 13/40
98323/98323 [==============================] - 38s 385us/step - loss: 0.3804 - roc_auc: 0.6596 - v
al_loss: 0.4436 - val_roc_auc: 0.6615
Epoch 14/40
98323/98323 [==============================] - 38s 386us/step - loss: 0.3779 - roc_auc: 0.6637 - v
al_loss: 0.4459 - val_roc_auc: 0.6655
```

```
al_loss: 0.4459 - val_roc_auc: 0.6655
Epoch 15/40
98323/98323 [==============================] - 38s 390us/step - loss: 0.3773 - roc_auc: 0.6674 - v
al_loss: 0.4506 - val_roc_auc: 0.6689
Epoch 16/40
98323/98323 [==============================] - 38s 386us/step - loss: 0.3783 - roc_auc: 0.6705 - v
al_loss: 0.4488 - val_roc_auc: 0.6718
Epoch 17/40
98323/98323 [==============================] - 38s 388us/step - loss: 0.3782 - roc_auc: 0.6732 - v
al_loss: 0.4467 - val_roc_auc: 0.6743
Epoch 18/40
98323/98323 [==============================] - 38s 387us/step - loss: 0.3760 - roc_auc: 0.6756 - v
al_loss: 0.4478 - val_roc_auc: 0.6768
Epoch 19/40
98323/98323 [==============================] - 38s 388us/step - loss: 0.3748 - roc_auc: 0.6782 - v
al_loss: 0.4462 - val_roc_auc: 0.6792
Epoch 20/40
98323/98323 [==============================] - 38s 385us/step - loss: 0.3743 - roc_auc: 0.6805 - v
al_loss: 0.4488 - val_roc_auc: 0.6814
Epoch 21/40
98323/98323 [==============================] - 38s 389us/step - loss: 0.3726 - roc_auc: 0.6826 - v
al_loss: 0.4508 - val_roc_auc: 0.6835
Epoch 22/40
98323/98323 [==============================] - 38s 384us/step - loss: 0.3737 - roc_auc: 0.6845 - v
al_loss: 0.4520 - val_roc_auc: 0.6853
Epoch 23/40
98323/98323 [==============================] - 38s 386us/step - loss: 0.3712 - roc_auc: 0.6864 - v
al_loss: 0.4598 - val_roc_auc: 0.6871
Epoch 24/40
98323/98323 [==============================] - 38s 384us/step - loss: 0.3727 - roc_auc: 0.6880 - v
al_loss: 0.4632 - val_roc_auc: 0.6886
Epoch 25/40
98323/98323 [==============================] - 38s 384us/step - loss: 0.3711 - roc_auc: 0.6894 - v
al_loss: 0.4626 - val_roc_auc: 0.6900
Epoch 26/40
98323/98323 [==============================] - 39s 399us/step - loss: 0.3697 - roc_auc: 0.6909 - v
al_loss: 0.4558 - val_roc_auc: 0.6915
Epoch 27/40
98323/98323 [==============================] - 43s 438us/step - loss: 0.3706 - roc_auc: 0.6923 - v
al_loss: 0.4581 - val_roc_auc: 0.6928
Epoch 28/40
98323/98323 [==============================] - 38s 387us/step - loss: 0.3681 - roc_auc: 0.6936 - v
al_loss: 0.4646 - val_roc_auc: 0.6942
Epoch 29/40
98323/98323 [==============================] - 39s 392us/step - loss: 0.3694 - roc_auc: 0.6948 - v
al_loss: 0.4679 - val_roc_auc: 0.6953
Epoch 30/40
98323/98323 [==============================] - 38s 386us/step - loss: 0.3663 - roc_auc: 0.6960 - v
al_loss: 0.4630 - val_roc_auc: 0.6966
Epoch 31/40
98323/98323 [==============================] - 38s 385us/step - loss: 0.3676 - roc_auc: 0.6973 - v
al_loss: 0.4686 - val_roc_auc: 0.6977
Epoch 32/40
98323/98323 [==============================] - 38s 386us/step - loss: 0.3661 - roc_auc: 0.6983 - v
al_loss: 0.4718 - val_roc_auc: 0.6988
Epoch 33/40
98323/98323 [==============================] - 38s 383us/step - loss: 0.3643 - roc_auc: 0.6995 - v
al_loss: 0.4767 - val_roc_auc: 0.6999
Epoch 34/40
98323/98323 [==============================] - 38s 386us/step - loss: 0.3663 - roc_auc: 0.7004 - v
al_loss: 0.4808 - val_roc_auc: 0.7007
Epoch 35/40
98323/98323 [==============================] - 38s 383us/step - loss: 0.3663 - roc_auc: 0.7012 - v
al_loss: 0.4775 - val_roc_auc: 0.7015
Epoch 36/40
98323/98323 [==============================] - 38s 391us/step - loss: 0.3663 - roc_auc: 0.7020 - v
al_loss: 0.4655 - val_roc_auc: 0.7023
Epoch 37/40
98323/98323 [==============================] - 38s 386us/step - loss: 0.3644 - roc_auc: 0.7029 - v
al_loss: 0.4799 - val_roc_auc: 0.7032
Epoch 38/40
98323/98323 [==============================] - 38s 389us/step - loss: 0.3646 - roc_auc: 0.7037 - v
al_loss: 0.4824 - val_roc_auc: 0.7039
Epoch 39/40
98323/98323 [==============================] - 38s 389us/step - loss: 0.3629 - roc_auc: 0.7044 - v
al_loss: 0.4767 - val_roc_auc: 0.7047
Epoch 40/40
98323/98323 [==============================] - 38s 384us/step - loss: 0.3632 - roc_auc: 0.7052 - v
```

```
98323/98323 [==============================] - 38s 384us/step - loss: 0.3632 - roc_auc: 0.7052 - v
al_loss: 0.4801 - val_roc_auc: 0.7055
```

Out[108]:

```
<keras.callbacks.History at 0x198b5b67828>
```

Saving my neural network model to JSON

In [109]:

```
from keras.models import model_from_json

# serialize model to JSON
model_json = model.to_json()
with open("model.json", "w") as json_file:
    json_file.write(model_json)
# serialize weights to HDF5
model.save_weights("model.h5")
```

Getting TF-IDF values of words in text data

In [16]:

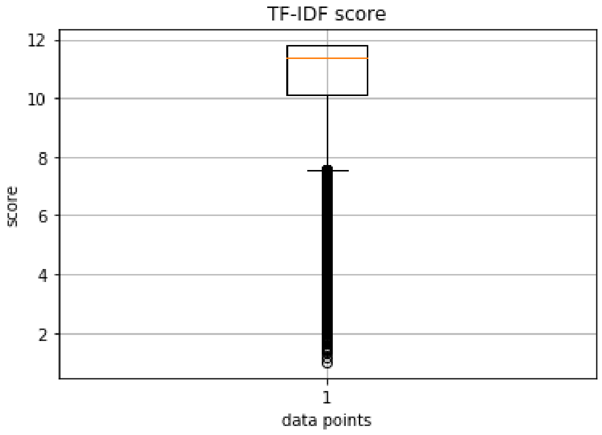
```
from sklearn.feature_extraction.text import TfidfVectorizer
from sklearn.feature_extraction.text import CountVectorizer
# merge texts
text = list(train['total_text'])

tfidf = TfidfVectorizer()
tfidf.fit_transform(text)

# dict key:word and value:tf-idf score
word2tfidf = dict(zip(tfidf.get_feature_names(), tfidf.idf_))
```

In [17]:

```
score=[]
for key in word2tfidf.keys():
    score.append(word2tfidf[key])
score=np.asarray(score).reshape(-1)
```

In [19]:

```
import matplotlib.pyplot as plt
plt.boxplot([score])
plt.title('TF-IDF score')
plt.xlabel('data points')
plt.ylabel('score')
plt.grid()
plt.show()
```



based on the above box plot low tf-idf value is 10 and high tf-idf value is 12

Removing words which have very high or very low TF-IDF values

In [20]:

```
remove_words=[]
for i in range(109248):
    total_words=(data['total_text'][i])
    wordList = re.sub("[^\w]", " ",  total_words).split()
    for j in wordList:
        try:
            idf = word2tfidf[j]
        except:
            idf = 0
        if(idf>10 and idf <12):
            continue;
        else:
            remove_words.append(j)
```

In [21]:

```
remove_words=list(set(remove_words))
```

In [27]:

```
#code copied from ---https://stackoverflow.com/questions/45447848/check-for-words-from-list-and-remove-those-words-in-pandas-dataframe-column
remove= r'\b(?:{})\b'.format('|'.join(remove_words))
train['total_text'] = train['total_text'].str.replace(remove, '')
test['total_text'] = test['total_text'].str.replace(remove, '')
```

In [35]:

```
token = Tokenizer()

token.fit_on_texts(train['total_text'])
train['total_text']=token.texts_to_sequences(train['total_text'])
test['total_text']=token.texts_to_sequences(test['total_text'])
text_size = len(token.word_index) + 1
```

Getting data in the form of Dictionary which then will be given as a input to Deep Learning Models

In [37]:

```
X_train={}

X_train["posted_projects"]= array(train["teacher_number_of_previously_posted_projects"]).reshape(len(train),1)
X_train["price"]= array(train["price"]).reshape(len(train),1)
X_train["quantity"]= array(train["quantity"]).reshape(len(train),1)
X_train["Is_digit_present"]= array(train["Is_digit_present"]).reshape(len(train),1)

X_train["teacher_prefix"]= pad_sequences(train["teacher_prefix"], maxlen=1)
X_train["school_state"]= pad_sequences(train["school_state"], maxlen=1)
X_train["project_grade_category"]= pad_sequences(train["project_grade_category"], maxlen=3)

X_train["total_text"]= pad_sequences(train["total_text"], maxlen=300)
X_train["clean_categories"]= pad_sequences(train["clean_categories"], maxlen=4)
X_train["clean_subcategories"]= pad_sequences(train["clean_subcategories"], maxlen=4)

X_train["output"]= array(train["project_is_approved"])
```

In [38]:

```
X_test={}
```

```
X_test={}

X_test["posted_projects"]= array(test["teacher_number_of_previously_posted_projects"]).reshape(len
(test),1)
X_test["price"]= array(test["price"]).reshape(len(test),1)
X_test["quantity"]= array(test["quantity"]).reshape(len(test),1)
X_test["Is_digit_present"]= array(test["Is_digit_present"]).reshape(len(test),1)

X_test["teacher_prefix"]= pad_sequences(test["teacher_prefix"], maxlen=1)
X_test["school_state"]= pad_sequences(test["school_state"], maxlen=1)
X_test["project_grade_category"]= pad_sequences(test["project_grade_category"], maxlen=3)

X_test["total_text"]= pad_sequences(test["total_text"], maxlen=300)
X_test["clean_categories"]= pad_sequences(test["clean_categories"], maxlen=4)
X_test["clean_subcategories"]= pad_sequences(test["clean_subcategories"], maxlen=4)

X_test["output"]= array(test["project_is_approved"])
```

In [40]:

```
import keras
    # Input layers
previously_posted_projects = Input(shape=(1,), name="posted_projects")
price = Input(shape=(1,), name="price")
digit_present = Input(shape=(1,), name="Is_digit_present")
quantity = Input(shape=(1,), name="quantity")


school_state = Input(shape=(1,), name="school_state")
teacher_prefix = Input(shape=(1,), name="teacher_prefix")
project_grade= Input(shape=(3,), name="project_grade_category")

total_text = Input(shape=(300,), name="total_text")
clean_categories = Input(shape=(4,), name="clean_categories")
clean_subcategories = Input(shape=(4,), name="clean_subcategories")

    # Batch normalization layer
#previously_posted_projects_bn = BatchNormalization(axis=1, momentum=0.99, epsilon=0.001, center=T
rue, scale=True, beta_initializer='zeros', gamma_initializer='ones',
moving_mean_initializer='zeros', moving_variance_initializer='ones')(previously_posted_projects)
#price_bn = BatchNormalization(axis=1, momentum=0.99, epsilon=0.001, center=True, scale=True, beta
_initializer='zeros', gamma_initializer='ones', moving_mean_initializer='zeros',
moving_variance_initializer='ones')(price)
#quantity_bn = BatchNormalization(axis=1, momentum=0.99, epsilon=0.001, center=True, scale=True, b
eta_initializer='zeros', gamma_initializer='ones', moving_mean_initializer='zeros',
moving_variance_initializer='ones')(quantity)



    # Embedding layers
emb_text_layer = Embedding(text_size, 300, weights=[embedding_matrix],  trainable=False)
emb_category_layer = Embedding(category_size, 4)
emb_subcategory_layer = Embedding(subcategory_size, 4)

emb_state_layer = Embedding(state_size, 8)
emb_prefix_layer = Embedding(prefix_size, 2)
emb_grade_layer = Embedding(grade_size, 2)

    # Giving Input to Embedding layers
emb_text = emb_text_layer(total_text)
emb_category =emb_category_layer(clean_categories)
emb_subcategory =emb_subcategory_layer(clean_subcategories)

emb_state =emb_state_layer(school_state)
emb_prefix =emb_prefix_layer(teacher_prefix)
emb_grade =emb_grade_layer(project_grade)


    # LSTM layers
lstm_text = LSTM(12, activation="relu", return_sequences=True)(emb_text)

    # Flatten layers
flatten_text =Flatten()(lstm_text)
flatten_category =Flatten()(emb_category)
flatten_subcategory =keras.layers.Flatten()(emb_subcategory)
```

```
flatten_state =Flatten()(emb_state)
flatten_prefix =Flatten()(emb_prefix)
flatten_grade =Flatten()(emb_grade)

    # concatenation of all numeric layers
numeric= concatenate([previously_posted_projects,
                                      price,
                                      digit_present,
                                      quantity])
    # Dense layer
dense_numeric =Dense(4, activation='relu',kernel_initializer=he_normal(seed=5))(numeric)

    # Merge all layers into one
x = concatenate([dense_numeric,
                                 flatten_text,
                                 flatten_category,
                                 flatten_subcategory,
                                 flatten_state,
                                 flatten_prefix,
                                 flatten_grade])

dense_x =Dense(8, activation='relu',kernel_initializer=he_normal(seed=3))(x)

dense_x_bn= BatchNormalization(axis=1, momentum=0.99, epsilon=0.001, center=True, scale=True, beta_
initializer='zeros', gamma_initializer='ones', moving_mean_initializer='zeros',
moving_variance_initializer='ones')(dense_x)

#drop_x =Dropout(0.5)(dense_x_bn)

dense2_x =Dense(4, activation='relu',kernel_initializer=he_normal(seed=1))(dense_x_bn)

#drop2_x =Dropout(0.5)(dense2_x)

dense3_x =Dense(2, activation='relu',kernel_initializer=he_normal(seed=2))(dense2_x)

    # Dense layers
    #x = keras.layers.Dense(128, activation="relu")(x)

    # Output layers
output = Dense(1, activation="sigmoid", name='final_output')(dense3_x)



model = Model(inputs=[previously_posted_projects,price,digit_present,quantity,school_state,teacher_
prefix,project_grade,total_text,clean_categories,clean_subcategories], outputs=[output])
model.summary()
```

```
__________________________________________________________________________________________________
Layer (type)                    Output Shape         Param #     Connected to
==================================================================================================
total_text (InputLayer)         (None, 300)          0
__________________________________________________________________________________________________
posted_projects (InputLayer)    (None, 1)            0
__________________________________________________________________________________________________
price (InputLayer)              (None, 1)            0
__________________________________________________________________________________________________
Is_digit_present (InputLayer)   (None, 1)            0
__________________________________________________________________________________________________
quantity (InputLayer)           (None, 1)            0
__________________________________________________________________________________________________
embedding_7 (Embedding)         (None, 300, 300)     18616800    total_text[0][0]
__________________________________________________________________________________________________
clean_categories (InputLayer)   (None, 4)            0
__________________________________________________________________________________________________
clean_subcategories (InputLayer (None, 4)            0
__________________________________________________________________________________________________
school_state (InputLayer)       (None, 1)            0
__________________________________________________________________________________________________
teacher_prefix (InputLayer)     (None, 1)            0
__________________________________________________________________________________________________
project_grade_category (InputLa (None, 3)            0
__________________________________________________________________________________________________
concatenate_1 (Concatenate)     (None, 4)            0           posted_projects[0][0]
                                                                 price[0][0]
                                                                 Is_digit_present[0][0]
                                                                 quantity[0][0]
```

```
__________________________________________________________________________________________________
lstm_2 (LSTM)                   (None, 300, 12)      15024       embedding_7[0][0]
__________________________________________________________________________________________________
embedding_8 (Embedding)         (None, 4, 4)         64          clean_categories[0][0]
__________________________________________________________________________________________________
embedding_9 (Embedding)         (None, 4, 4)         152         clean_subcategories[0][0]
__________________________________________________________________________________________________
embedding_10 (Embedding)        (None, 1, 8)         416         school_state[0][0]
__________________________________________________________________________________________________
embedding_11 (Embedding)        (None, 1, 2)         12          teacher_prefix[0][0]
__________________________________________________________________________________________________
embedding_12 (Embedding)        (None, 3, 2)         20          project_grade_category[0][0]
__________________________________________________________________________________________________
dense_1 (Dense)                 (None, 4)            20          concatenate_1[0][0]
__________________________________________________________________________________________________
flatten_3 (Flatten)             (None, 3600)         0           lstm_2[0][0]
__________________________________________________________________________________________________
flatten_4 (Flatten)             (None, 16)           0           embedding_8[0][0]
__________________________________________________________________________________________________
flatten_5 (Flatten)             (None, 16)           0           embedding_9[0][0]
__________________________________________________________________________________________________
flatten_6 (Flatten)             (None, 8)            0           embedding_10[0][0]
__________________________________________________________________________________________________
flatten_7 (Flatten)             (None, 2)            0           embedding_11[0][0]
__________________________________________________________________________________________________
flatten_8 (Flatten)             (None, 6)            0           embedding_12[0][0]
__________________________________________________________________________________________________
concatenate_2 (Concatenate)     (None, 3652)         0           dense_1[0][0]
                                                                 flatten_3[0][0]
                                                                 flatten_4[0][0]
                                                                 flatten_5[0][0]
                                                                 flatten_6[0][0]
                                                                 flatten_7[0][0]
                                                                 flatten_8[0][0]
__________________________________________________________________________________________________
dense_2 (Dense)                 (None, 8)            29224       concatenate_2[0][0]
__________________________________________________________________________________________________
batch_normalization_1 (BatchNor (None, 8)            32          dense_2[0][0]
__________________________________________________________________________________________________
dense_3 (Dense)                 (None, 4)            36          batch_normalization_1[0][0]
__________________________________________________________________________________________________
dense_4 (Dense)                 (None, 2)            10          dense_3[0][0]
__________________________________________________________________________________________________
final_output (Dense)            (None, 1)            3           dense_4[0][0]
==================================================================================================
Total params: 18,661,813
Trainable params: 44,997
Non-trainable params: 18,616,816
__________________________________________________________________________________________________
```

Running Model 1 but with words removed based on their TF-IDF values

In [41]:

```
tb =TensorBoard(log_dir="logs2/{}".format(time()))

model.compile(optimizer = Adam(lr=1e-3),
                  loss={'final_output': 'binary_crossentropy'},
                  metrics=[roc_auc])




model.fit({"posted_projects":X_train['posted_projects'], "price":X_train['price'],
              "Is_digit_present":X_train['Is_digit_present'], "quantity":X_train['quantity'],
              "school_state":X_train['school_state'], "teacher_prefix":X_train['teacher_prefix'],
        "project_grade_category":X_train['project_grade_category'], "total_text":X_train['total_tex
t'],
        "clean_categories":X_train['clean_categories'], "clean_subcategories":X_train['clean_subcat
egories']}
              ,{
              "final_output":X_train['output']},
                 batch_size=2500,
                epochs=40,
               validation_data=({"posted_projects":X_test['posted_projects'], "price":X_test['price
'],
              "Is_digit_present":X_test['Is_digit_present'], "quantity":X_test['quantity'],
              "school_state":X_test['school_state'], "teacher_prefix":X_test['teacher_prefix'],
        "project_grade_category":X_test['project_grade_category'], "total_text":X_test['total_text'
],
        "clean_categories":X_test['clean_categories'], "clean_subcategories":X_test['clean_subcateg
ories']}
              ,{
              "final_output":X_test['output']}),
                callbacks=[tb])
```

```
WARNING:tensorflow:From C:\ProgramData\Anaconda3\lib\site-
packages\tensorflow\python\ops\metrics_impl.py:526: to_float (from tensorflow.python.ops.math_ops)
is deprecated and will be removed in a future version.
Instructions for updating:
Use tf.cast instead.
WARNING:tensorflow:From C:\ProgramData\Anaconda3\lib\site-
packages\tensorflow\python\ops\metrics_impl.py:788: div (from tensorflow.python.ops.math_ops) is d
eprecated and will be removed in a future version.
Instructions for updating:
Deprecated in favor of operator or tf.math.divide.
WARNING:tensorflow:From C:\ProgramData\Anaconda3\lib\site-
packages\tensorflow\python\ops\math_ops.py:3066: to_int32 (from tensorflow.python.ops.math_ops) is
deprecated and will be removed in a future version.
Instructions for updating:
Use tf.cast instead.
Train on 98323 samples, validate on 10925 samples
Epoch 1/40
98323/98323 [==============================] - 36s 362us/step - loss: 0.6599 - roc_auc: 0.4565 - v
al_loss: 0.6201 - val_roc_auc: 0.4675
Epoch 2/40
98323/98323 [==============================] - 33s 333us/step - loss: 0.5860 - roc_auc: 0.4874 - v
al_loss: 0.5089 - val_roc_auc: 0.5061
Epoch 3/40
98323/98323 [==============================] - 33s 337us/step - loss: 0.5117 - roc_auc: 0.5199 - v
al_loss: 0.4726 - val_roc_auc: 0.5307
Epoch 4/40
98323/98323 [==============================] - 33s 336us/step - loss: 0.4589 - roc_auc: 0.5406 - v
al_loss: 0.4307 - val_roc_auc: 0.5484
Epoch 5/40
98323/98323 [==============================] - 33s 335us/step - loss: 0.4236 - roc_auc: 0.5563 - v
al_loss: 0.4505 - val_roc_auc: 0.5630
Epoch 6/40
98323/98323 [==============================] - 33s 332us/step - loss: 0.4017 - roc_auc: 0.5702 - v
al_loss: 0.4707 - val_roc_auc: 0.5766
Epoch 7/40
98323/98323 [==============================] - 33s 335us/step - loss: 0.3895 - roc_auc: 0.5835 - v
al_loss: 0.4460 - val_roc_auc: 0.5899
Epoch 8/40
98323/98323 [==============================] - 33s 333us/step - loss: 0.3817 - roc_auc: 0.5967 - v
al_loss: 0.4476 - val_roc_auc: 0.6026
Epoch 9/40
98323/98323 [==============================] - 33s 338us/step - loss: 0.3781 - roc_auc: 0.6085 - v
al_loss: 0.4495 - val_roc_auc: 0.6137
Epoch 10/40
98323/98323 [==============================] - 33s 338us/step - loss: 0.3729 - roc_auc: 0.6191 - v
al_loss: 0.4454 - val_roc_auc: 0.6238
Epoch 11/40
98323/98323 [==============================] - 33s 336us/step - loss: 0.3688 - roc_auc: 0.6287 - v
al_loss: 0.4161 - val_roc_auc: 0.6334
Epoch 12/40
98323/98323 [==============================] - 34s 344us/step - loss: 0.3661 - roc_auc: 0.6383 - v
al_loss: 0.4534 - val_roc_auc: 0.6421
Epoch 13/40
98323/98323 [==============================] - 33s 341us/step - loss: 0.3627 - roc_auc: 0.6461 - v
al_loss: 0.4164 - val_roc_auc: 0.6499
Epoch 14/40
98323/98323 [==============================] - 33s 333us/step - loss: 0.3596 - roc_auc: 0.6540 - v
al_loss: 0.4248 - val_roc_auc: 0.6575
Epoch 15/40
98323/98323 [==============================] - 33s 336us/step - loss: 0.3567 - roc_auc: 0.6612 - v
al_loss: 0.4345 - val_roc_auc: 0.6643
Epoch 16/40
98323/98323 [==============================] - 33s 331us/step - loss: 0.3539 - roc_auc: 0.6676 - v
al_loss: 0.4643 - val_roc_auc: 0.6705
Epoch 17/40
Epoch 17/40
98323/98323 [==============================] - 33s 335us/step - loss: 0.3529 - roc_auc: 0.6733 - v
al_loss: 0.4027 - val_roc_auc: 0.6763
Epoch 18/40
98323/98323 [==============================] - 32s 329us/step - loss: 0.3489 - roc_auc: 0.6795 - v
al_loss: 0.4042 - val_roc_auc: 0.6824
Epoch 19/40
98323/98323 [==============================] - 33s 334us/step - loss: 0.3462 - roc_auc: 0.6854 - v
al_loss: 0.4231 - val_roc_auc: 0.6879
Epoch 20/40
98323/98323 [==============================] - 33s 336us/step - loss: 0.3442 - roc_auc: 0.6906 - v
al_loss: 0.4086 - val_roc_auc: 0.6931
Epoch 21/40
98323/98323 [==============================] - 33s 333us/step - loss: 0.3409 - roc_auc: 0.6957 - v
al_loss: 0.4063 - val_roc_auc: 0.6981
Epoch 22/40
98323/98323 [==============================] - 33s 336us/step - loss: 0.3383 - roc_auc: 0.7006 - v
al_loss: 0.4158 - val_roc_auc: 0.7029
Epoch 23/40
98323/98323 [==============================] - 33s 336us/step - loss: 0.3356 - roc_auc: 0.7054 - v
al_loss: 0.4091 - val_roc_auc: 0.7075
Epoch 24/40
98323/98323 [==============================] - 33s 335us/step - loss: 0.3333 - roc_auc: 0.7098 - v
al_loss: 0.4304 - val_roc_auc: 0.7118
Epoch 25/40
98323/98323 [==============================] - 33s 337us/step - loss: 0.3309 - roc_auc: 0.7140 - v
al_loss: 0.4497 - val_roc_auc: 0.7158
Epoch 26/40
98323/98323 [==============================] - 33s 333us/step - loss: 0.3286 - roc_auc: 0.7178 - v
al_loss: 0.4439 - val_roc_auc: 0.7196
Epoch 27/40
98323/98323 [==============================] - 33s 334us/step - loss: 0.3259 - roc_auc: 0.7216 - v
al_loss: 0.4760 - val_roc_auc: 0.7232
Epoch 28/40
98323/98323 [==============================] - 33s 333us/step - loss: 0.3231 - roc_auc: 0.7251 - v
al_loss: 0.5100 - val_roc_auc: 0.7266
Epoch 29/40
98323/98323 [==============================] - 33s 331us/step - loss: 0.3220 - roc_auc: 0.7284 - v
al_loss: 0.4288 - val_roc_auc: 0.7301
Epoch 30/40
98323/98323 [==============================] - 33s 333us/step - loss: 0.3187 - roc_auc: 0.7319 - v
al_loss: 0.4312 - val_roc_auc: 0.7336
Epoch 31/40
98323/98323 [==============================] - 34s 351us/step - loss: 0.3149 - roc_auc: 0.7354 - v
al_loss: 0.4542 - val_roc_auc: 0.7369
Epoch 32/40
98323/98323 [==============================] - 33s 334us/step - loss: 0.3138 - roc_auc: 0.7386 - v
al_loss: 0.4366 - val_roc_auc: 0.7402
Epoch 33/40
98323/98323 [==============================] - 33s 334us/step - loss: 0.3122 - roc_auc: 0.7419 - v
al_loss: 0.4799 - val_roc_auc: 0.7433
Epoch 34/40
98323/98323 [==============================] - 33s 334us/step - loss: 0.3110 - roc_auc: 0.7448 - v
al_loss: 0.5137 - val_roc_auc: 0.7460
Epoch 35/40
98323/98323 [==============================] - 33s 334us/step - loss: 0.3075 - roc_auc: 0.7474 - v
al_loss: 0.5957 - val_roc_auc: 0.7483
Epoch 36/40
98323/98323 [==============================] - 33s 340us/step - loss: 0.3051 - roc_auc: 0.7496 - v
al_loss: 0.5717 - val_roc_auc: 0.7505
Epoch 37/40
98323/98323 [==============================] - 33s 335us/step - loss: 0.3031 - roc_auc: 0.7518 - v
al_loss: 0.4656 - val_roc_auc: 0.7531
Epoch 38/40
98323/98323 [==============================] - 33s 336us/step - loss: 0.3026 - roc_auc: 0.7545 - v
al_loss: 0.4533 - val_roc_auc: 0.7558
Epoch 39/40
98323/98323 [==============================] - 33s 336us/step - loss: 0.3004 - roc_auc: 0.7572 - v
al_loss: 0.4777 - val_roc_auc: 0.7585
Epoch 40/40
98323/98323 [==============================] - 33s 337us/step - loss: 0.2981 - roc_auc: 0.7599 - v
al_loss: 0.4897 - val_roc_auc: 0.7610
```

Out[41]:

```
<keras.callbacks.History at 0x2172b500f60>
```

Saving my neural network model to JSON

In [42]:

```
# serialize model to JSON
model_json = model.to_json()
with open("model_reduce_words.json", "w") as json_file:
    json_file.write(model_json)
# serialize weights to HDF5
model.save_weights("model_reduce_words.h5")
```

One Hot encoding categorical data using keras preprocessing

In [38]:

```
train['teacher_prefix'] = train['teacher_prefix'].map(lambda a:keras.preprocessing.text.one_hot(a,100))
train['school_state'] = train['school_state'].map(lambda a:keras.preprocessing.text.one_hot(a,100))
train['project_grade_category'] = train['project_grade_category'].map(lambda a:keras.preprocessing.text.one_hot(a,100))
train['clean_categories'] = train['clean_categories'].map(lambda a:keras.preprocessing.text.one_hot(a,100))
train['clean_subcategories'] = train['clean_subcategories'].map(lambda a:keras.preprocessing.text.one_hot(a,100))
```

In [35]:

```
test['teacher_prefix'] = test['teacher_prefix'].map(lambda a:keras.preprocessing.text.one_hot(a,100))
test['school_state'] = test['school_state'].map(lambda a:keras.preprocessing.text.one_hot(a,100))
test['project_grade_category'] = test['project_grade_category'].map(lambda a:keras.preprocessing.text.one_hot(a,100))
test['clean_categories'] = test['clean_categories'].map(lambda a:keras.preprocessing.text.one_hot(a,100))
test['clean_subcategories'] = test['clean_subcategories'].map(lambda a:keras.preprocessing.text.one_hot(a,100))
```

Getting data in the form of Dictionary which then will be given as a input to Deep Learning Models

In [16]:

```
X_train={}

X_train["posted_projects"]= array(train["teacher_number_of_previously_posted_projects"]).reshape(len(train),1)
X_train["price"]= array(train["price"]).reshape(len(train),1)
X_train["quantity"]= array(train["quantity"]).reshape(len(train),1)
X_train["Is_digit_present"]= array(train["Is_digit_present"]).reshape(len(train),1)

X_train["teacher_prefix"]= pad_sequences(train["teacher_prefix"], maxlen=4)
X_train["school_state"]= pad_sequences(train["school_state"], maxlen=4)
X_train["project_grade_category"]= pad_sequences(train["project_grade_category"], maxlen=4)

X_train["total_text"]= pad_sequences(train["total_text"], maxlen=300)
X_train["clean_categories"]= pad_sequences(train["clean_categories"], maxlen=4)
X_train["clean_subcategories"]= pad_sequences(train["clean_subcategories"], maxlen=4)

X_train["output"]= array(train["project_is_approved"])
```

In [17]:

```
X_test={}

X_test["posted_projects"]= array(test["teacher_number_of_previously_posted_projects"]).reshape(len(test),1)
X_test["price"]= array(test["price"]).reshape(len(test),1)
```

```
X_test["quantity"]= array(test["quantity"]).reshape(len(test),1)
X_test["Is_digit_present"]= array(test["Is_digit_present"]).reshape(len(test),1)

X_test["teacher_prefix"]= pad_sequences(test["teacher_prefix"], maxlen=4)
X_test["school_state"]= pad_sequences(test["school_state"], maxlen=4)
X_test["project_grade_category"]= pad_sequences(test["project_grade_category"], maxlen=4)

X_test["total_text"]= pad_sequences(test["total_text"], maxlen=300)
X_test["clean_categories"]= pad_sequences(test["clean_categories"], maxlen=4)
X_test["clean_subcategories"]= pad_sequences(test["clean_subcategories"], maxlen=4)

X_test["output"]= array(test["project_is_approved"])
```

Model 2

In [18]:

```
from keras.layers.core import Reshape
from keras.layers import Conv1D
```

In [28]:

```
import keras
    # Input layers
previously_posted_projects = Input(shape=(1,), name="posted_projects")
price = Input(shape=(1,), name="price")
digit_present = Input(shape=(1,), name="Is_digit_present")
quantity = Input(shape=(1,), name="quantity")


school_state = Input(shape=(4,), name="school_state")
teacher_prefix = Input(shape=(4,), name="teacher_prefix")
project_grade= Input(shape=(4,), name="project_grade_category")

total_text = Input(shape=(300,), name="total_text")
clean_categories = Input(shape=(4,), name="clean_categories")
clean_subcategories = Input(shape=(4,), name="clean_subcategories")

    # Embedding layers
emb_text_layer = Embedding(text_size, 300, weights=[embedding_matrix], trainable=False)


    # Giving Input to Embedding layers
emb_text = emb_text_layer(total_text)

    # LSTM layers
lstm_text = LSTM(14, activation="relu", return_sequences=True)(emb_text)

    # Flatten layers
flatten_text = Flatten()(lstm_text)


    # concatenation of all numeric and categorical layers
other= concatenate([
                                      school_state,
                                      teacher_prefix,
                                      project_grade,
                                      clean_categories,
                                      clean_subcategories])
    # Dense layer
new= Reshape([4,-1])(other)

    # cnn layer
cnn_1= Conv1D(12,1, activation='relu')(new)

cnn_2= Conv1D(24, 3, activation='relu')(cnn_1)

    # Flatten layer
flatten_cnn_2= Flatten()(cnn_2)

     # Batch normalization layer
#previously_posted_projects_bn = BatchNormalization(axis=1, momentum=0.99, epsilon=0.001, center=T
rue, scale=True, beta_initializer='zeros', gamma_initializer='ones',
moving_mean_initializer='zeros', moving_variance_initializer='ones')(previously_posted_projects)
```

```
moving_mean_initializer='zeros', moving_variance_initializer='ones')(previously_posted_projects)
#price_bn= BatchNormalization(axis=1, momentum=0.99, epsilon=0.001, center=True, scale=True, beta_
initializer='zeros', gamma_initializer='ones', moving_mean_initializer='zeros',
moving_variance_initializer='ones')(price)
#quantity_bn = BatchNormalization(axis=1, momentum=0.99, epsilon=0.001, center=True, scale=True, b
eta_initializer='zeros', gamma_initializer='ones', moving_mean_initializer='zeros',
moving_variance_initializer='ones')(quantity)


    # Merge all layers into one
x = concatenate([flatten_text, flatten_cnn_2,previously_posted_projects,
                                 price,
                                 quantity,
                                 digit_present])

dense_x = Dense(8, activation='relu',kernel_initializer=he_normal(seed=None))(x)

dense_x_bn= keras.layers.BatchNormalization(axis=1, momentum=0.99, epsilon=0.001, center=True, scal
e=True, beta_initializer='zeros', gamma_initializer='ones', moving_mean_initializer='zeros', moving
_variance_initializer='ones')(dense_x)

#drop_x = Dropout(0.5)(dense_x_bn)

dense2_x = Dense(4, activation='relu',kernel_initializer=he_normal(seed=None))(dense_x_bn)

#drop2_x = Dropout(0.5)(dense2_x)

dense3_x = Dense(2, activation='relu',kernel_initializer=he_normal(seed=None))(dense2_x)

    # Dense layers
    #x = keras.layers.Dense(128, activation="relu")(x)

    # Output layers
output = Dense(1, activation="sigmoid", name="final_output")(dense3_x)



model = Model(inputs=[previously_posted_projects,price,digit_present,quantity,school_state,teacher_
prefix,project_grade,total_text,clean_categories,clean_subcategories], outputs=[output])
model.summary()
```

```
__________________________________________________________________________________________________
Layer (type)                    Output Shape         Param #     Connected to
==================================================================================================
school_state (InputLayer)       (None, 4)            0
__________________________________________________________________________________________________
teacher_prefix (InputLayer)     (None, 4)            0
__________________________________________________________________________________________________
project_grade_category (InputLa (None, 4)            0
__________________________________________________________________________________________________
clean_categories (InputLayer)   (None, 4)            0
__________________________________________________________________________________________________
clean_subcategories (InputLayer (None, 4)            0
__________________________________________________________________________________________________
concatenate_13 (Concatenate)    (None, 20)           0           school_state[0][0]
                                                                 teacher_prefix[0][0]
                                                                 project_grade_category[0][0]
                                                                 clean_categories[0][0]
                                                                 clean_subcategories[0][0]
__________________________________________________________________________________________________
total_text (InputLayer)         (None, 300)          0
__________________________________________________________________________________________________
reshape_7 (Reshape)             (None, 4, 5)         0           concatenate_13[0][0]
__________________________________________________________________________________________________
embedding_7 (Embedding)         (None, 300, 300)     18616800    total_text[0][0]
__________________________________________________________________________________________________
conv1d_13 (Conv1D)              (None, 4, 12)        72          reshape_7[0][0]
__________________________________________________________________________________________________
lstm_7 (LSTM)                   (None, 300, 14)      17640       embedding_7[0][0]
__________________________________________________________________________________________________
conv1d_14 (Conv1D)              (None, 2, 24)        888         conv1d_13[0][0]
__________________________________________________________________________________________________
flatten_13 (Flatten)            (None, 4200)         0           lstm_7[0][0]
__________________________________________________________________________________________________
flatten_14 (Flatten)            (None, 48)           0           conv1d_14[0][0]
__________________________________________________________________________________________________
posted_projects (InputLayer)    (None, 1)            0
```

```
      _
__________________________________________________________________________________________________
price (InputLayer)              (None, 1)            0
__________________________________________________________________________________________________
quantity (InputLayer)           (None, 1)            0
__________________________________________________________________________________________________
Is_digit_present (InputLayer)   (None, 1)            0
__________________________________________________________________________________________________
concatenate_14 (Concatenate)    (None, 4252)         0           flatten_13[0][0]
                                                                 flatten_14[0][0]
                                                                 posted_projects[0][0]
                                                                 price[0][0]
                                                                 quantity[0][0]
                                                                 Is_digit_present[0][0]
__________________________________________________________________________________________________
dense_17 (Dense)                (None, 8)            34024       concatenate_14[0][0]
__________________________________________________________________________________________________
batch_normalization_6 (BatchNor (None, 8)            32          dense_17[0][0]
__________________________________________________________________________________________________
dense_18 (Dense)                (None, 4)            36          batch_normalization_6[0][0]
__________________________________________________________________________________________________
dense_19 (Dense)                (None, 2)            10          dense_18[0][0]
__________________________________________________________________________________________________
final_output (Dense)            (None, 1)            3           dense_19[0][0]
==================================================================================================
Total params: 18,669,505
Trainable params: 52,689
Non-trainable params: 18,616,816
__________________________________________________________________________________________________
```

In [29]:

```
tb =TensorBoard(log_dir="logs3/{}".format(time()))

model.compile(optimizer= Adam(lr=1e-2),
                  loss={'final_output': 'binary_crossentropy'},
                  metrics=[roc_auc])




model.fit({"posted_projects":X_train['posted_projects'], "price":X_train['price'],
              "Is_digit_present":X_train['Is_digit_present'], "quantity":X_train['quantity'],
              "school_state":X_train['school_state'], "teacher_prefix":X_train['teacher_prefix'],
        "project_grade_category":X_train['project_grade_category'], "total_text":X_train['total_tex
t'],
        "clean_categories":X_train['clean_categories'], "clean_subcategories":X_train['clean_subcat
egories']}
              ,{
              "final_output":X_train['output']},
                 batch_size=2500,
                epochs=40,

               validation_data=({"posted_projects":X_test['posted_projects'], "price":X_test['price
'],
              "Is_digit_present":X_test['Is_digit_present'], "quantity":X_test['quantity'],
              "school_state":X_test['school_state'], "teacher_prefix":X_test['teacher_prefix'],
        "project_grade_category":X_test['project_grade_category'], "total_text":X_test['total_text'
],
        "clean_categories":X_test['clean_categories'], "clean_subcategories":X_test['clean_subcateg
ories']}
              ,{
              "final_output":X_test['output']}),
                callbacks=[tb])
```

```
Train on 98323 samples, validate on 10925 samples
Epoch 1/40
98323/98323 [==============================] - 36s 371us/step - loss: 0.4426 - roc_auc: 0.5899 - v
al_loss: 0.4213 - val_roc_auc: 0.6301
Epoch 2/40
98323/98323 [==============================] - 34s 341us/step - loss: 0.3742 - roc_auc: 0.6606 - v
al_loss: 0.3895 - val_roc_auc: 0.6835
Epoch 3/40
98323/98323 [==============================] - 34s 341us/step - loss: 0.3647 - roc_auc: 0.6986 - v
al_loss: 0.3843 - val_roc_auc: 0.7091
Epoch 4/40
```

```
98323/98323 [==============================] - 34s 342us/step - loss: 0.3570 - roc_auc: 0.7185 - v
al_loss: 0.4054 - val_roc_auc: 0.7250
Epoch 5/40
98323/98323 [==============================] - 34s 342us/step - loss: 0.3504 - roc_auc: 0.7317 - v
al_loss: 0.3895 - val_roc_auc: 0.7371
Epoch 6/40
98323/98323 [==============================] - 34s 341us/step - loss: 0.3401 - roc_auc: 0.7435 - v
al_loss: 0.3904 - val_roc_auc: 0.7484
Epoch 7/40
98323/98323 [==============================] - 34s 341us/step - loss: 0.3321 - roc_auc: 0.7542 - v
al_loss: 0.4048 - val_roc_auc: 0.7582
Epoch 8/40
98323/98323 [==============================] - 34s 343us/step - loss: 0.3228 - roc_auc: 0.7634 - v
al_loss: 0.4067 - val_roc_auc: 0.7672
Epoch 9/40
98323/98323 [==============================] - 33s 338us/step - loss: 0.3134 - roc_auc: 0.7720 - v
al_loss: 0.4245 - val_roc_auc: 0.7758
Epoch 10/40
98323/98323 [==============================] - 34s 343us/step - loss: 0.3072 - roc_auc: 0.7803 - v
al_loss: 0.4499 - val_roc_auc: 0.7831
Epoch 11/40
98323/98323 [==============================] - 34s 341us/step - loss: 0.2974 - roc_auc: 0.7874 - v
al_loss: 0.4687 - val_roc_auc: 0.7901
Epoch 12/40
98323/98323 [==============================] - 34s 341us/step - loss: 0.2916 - roc_auc: 0.7939 - v
al_loss: 0.4538 - val_roc_auc: 0.7965
Epoch 13/40
98323/98323 [==============================] - 34s 347us/step - loss: 0.2845 - roc_auc: 0.8000 - v
al_loss: 0.4703 - val_roc_auc: 0.8026
Epoch 14/40
98323/98323 [==============================] - 33s 340us/step - loss: 0.2759 - roc_auc: 0.8060 - v
al_loss: 0.4887 - val_roc_auc: 0.8084
Epoch 15/40
98323/98323 [==============================] - 33s 338us/step - loss: 0.2727 - roc_auc: 0.8115 - v
al_loss: 0.4735 - val_roc_auc: 0.8136
Epoch 16/40
98323/98323 [==============================] - 34s 343us/step - loss: 0.2742 - roc_auc: 0.8160 - v
al_loss: 0.5081 - val_roc_auc: 0.8178
Epoch 17/40
98323/98323 [==============================] - 33s 338us/step - loss: 0.2595 - roc_auc: 0.8206 - v
al_loss: 0.5350 - val_roc_auc: 0.8225
Epoch 18/40
98323/98323 [==============================] - 34s 342us/step - loss: 0.2728 - roc_auc: 0.8244 - v
al_loss: 0.5052 - val_roc_auc: 0.8257
Epoch 19/40
98323/98323 [==============================] - 33s 339us/step - loss: 0.2589 - roc_auc: 0.8278 - v
al_loss: 0.5571 - val_roc_auc: 0.8293
Epoch 20/40
98323/98323 [==============================] - 33s 340us/step - loss: 0.2495 - roc_auc: 0.8315 - v
al_loss: 0.5349 - val_roc_auc: 0.8331
Epoch 21/40
98323/98323 [==============================] - 34s 343us/step - loss: 0.2441 - roc_auc: 0.8353 - v
al_loss: 0.5689 - val_roc_auc: 0.8368
Epoch 22/40
98323/98323 [==============================] - 33s 339us/step - loss: 0.2405 - roc_auc: 0.8387 - v
al_loss: 0.5986 - val_roc_auc: 0.8401
Epoch 23/40
98323/98323 [==============================] - 33s 340us/step - loss: 0.2348 - roc_auc: 0.8420 - v
al_loss: 0.6061 - val_roc_auc: 0.8433
Epoch 24/40
98323/98323 [==============================] - 34s 341us/step - loss: 0.2321 - roc_auc: 0.8450 - v
al_loss: 0.5913 - val_roc_auc: 0.8463
Epoch 25/40
98323/98323 [==============================] - 34s 342us/step - loss: 0.2262 - roc_auc: 0.8480 - v
al_loss: 0.6160 - val_roc_auc: 0.8493
Epoch 26/40
98323/98323 [==============================] - 34s 343us/step - loss: 0.2252 - roc_auc: 0.8509 - v
al_loss: 0.6249 - val_roc_auc: 0.8520
Epoch 27/40
98323/98323 [==============================] - 34s 343us/step - loss: 0.2178 - roc_auc: 0.8536 - v
al_loss: 0.6534 - val_roc_auc: 0.8547
Epoch 28/40
98323/98323 [==============================] - 33s 340us/step - loss: 0.2179 - roc_auc: 0.8562 - v
al_loss: 0.6365 - val_roc_auc: 0.8572
Epoch 29/40
98323/98323 [==============================] - 33s 340us/step - loss: 0.2166 - roc_auc: 0.8586 - v
al_loss: 0.6717 - val_roc_auc: 0.8595
Epoch 30/40
Epoch 30/40
98323/98323 [==============================] - 34s 341us/step - loss: 0.2143 - roc_auc: 0.8608 - v
al_loss: 0.6637 - val_roc_auc: 0.8617
Epoch 31/40
98323/98323 [==============================] - 34s 341us/step - loss: 0.2105 - roc_auc: 0.8629 - v
al_loss: 0.6719 - val_roc_auc: 0.8638
Epoch 32/40
98323/98323 [==============================] - 34s 344us/step - loss: 0.2097 - roc_auc: 0.8649 - v
al_loss: 0.6991 - val_roc_auc: 0.8657
Epoch 33/40
98323/98323 [==============================] - 34s 342us/step - loss: 0.2027 - roc_auc: 0.8669 - v
al_loss: 0.7001 - val_roc_auc: 0.8677
Epoch 34/40
98323/98323 [==============================] - 34s 342us/step - loss: 0.2018 - roc_auc: 0.8689 - v
al_loss: 0.6934 - val_roc_auc: 0.8696
Epoch 35/40
98323/98323 [==============================] - 34s 343us/step - loss: 0.2016 - roc_auc: 0.8706 - v
al_loss: 0.7564 - val_roc_auc: 0.8713
Epoch 36/40
98323/98323 [==============================] - 34s 343us/step - loss: 0.1961 - roc_auc: 0.8723 - v
al_loss: 0.7229 - val_roc_auc: 0.8730
Epoch 37/40
98323/98323 [==============================] - 33s 341us/step - loss: 0.2060 - roc_auc: 0.8739 - v
al_loss: 0.6924 - val_roc_auc: 0.8745
Epoch 38/40
98323/98323 [==============================] - 33s 340us/step - loss: 0.1971 - roc_auc: 0.8754 - v
al_loss: 0.6980 - val_roc_auc: 0.8760
Epoch 39/40
98323/98323 [==============================] - 33s 339us/step - loss: 0.1936 - roc_auc: 0.8769 - v
al_loss: 0.7701 - val_roc_auc: 0.8775
Epoch 40/40
98323/98323 [==============================] - 34s 343us/step - loss: 0.1885 - roc_auc: 0.8784 - v
al_loss: 0.7549 - val_roc_auc: 0.8790
```

Out[29]:

```
<keras.callbacks.History at 0x2bff0e0e710>
```

Saving my neural network model to JSON

In [31]:

```python
# serialize model to JSON
model_json = model.to_json()
with open("model_conv1.json", "w") as json_file:
    json_file.write(model_json)
# serialize weights to HDF5
model.save_weights("model_conv1.h5")
```

Result

In [15]:

```python
#code copied from -http://zetcode.com/python/prettytable/
from prettytable import PrettyTable
x = PrettyTable()
x.field_names = ["Model no.", "train auc", "test auc"]
x.add_row(["Model 1", 0.7052 ,0.7055])
x.add_row(["Model 1 with reduced words", 0.7599, 0.7610])
x.add_row(["Model 2", 0.8784 ,0.8790])

print(x)
```

```
+----------------------------+-----------+----------+
|         Model no.          | train auc | test auc |
+----------------------------+-----------+----------+
|          Model 1           |   0.7052  |  0.7055  |
| Model 1 with reduced words |   0.7599  |  0.761   |
|          Model 2           |   0.8784  |  0.879   |
+----------------------------+-----------+----------+
```