# DonorsChoose

DonorsChoose.org receives hundreds of thousands of project proposals each year for classroom projects in need of funding. Right now, a large number of volunteers is needed to manually screen each submission before it's approved to be posted on the DonorsChoose.org website.

Next year, DonorsChoose.org expects to receive close to 500,000 project proposals. As a result, there are three main problems they need to solve:

- How to scale current manual processes and resources to screen 500,000 projects so that they can be posted as quickly and as efficiently as possible
- How to increase the consistency of project vetting across different volunteers to improve the experience for teachers
- How to focus volunteer time on the applications that need the most assistance

The goal of the competition is to predict whether or not a DonorsChoose.org project proposal submitted by a teacher will be approved, using the text of project descriptions as well as additional metadata about the project, teacher, and school. DonorsChoose.org can then use this information to identify projects most likely to need further review before approval.

## About the DonorsChoose Data Set

The `train.csv` data set provided by DonorsChoose contains the following features:

| Feature | Description |
|---:|---:|
| **`project_id`** | A unique identifier for the proposed project. **Example:** `p036502` |
| **`project_title`** | Title of the project. **Examples:**<br>• `Art Will Make You Happy!`<br>• `First Grade Fun` |
| **`project_grade_category`** | Grade level of students for which the project is targeted. One of the following enumerated values:<br>• `Grades PreK-2`<br>• `Grades 3-5`<br>• `Grades 6-8`<br>• `Grades 9-12` |
| **`project_subject_categories`** | One or more (comma-separated) subject categories for the project from the following enumerated list of values:<br>• `Applied Learning`<br>• `Care & Hunger`<br>• `Health & Sports`<br>• `History & Civics`<br>• `Literacy & Language`<br>• `Math & Science`<br>• `Music & The Arts`<br>• `Special Needs`<br>• `Warmth`<br>**Examples:**<br>• `Music & The Arts`<br>• `Literacy & Language, Math & Science` |
| **`school_state`** | State where school is located (Two-letter U.S. postal code). **Example:** `WY` |
| **`project_subject_subcategories`** | One or more (comma-separated) subject subcategories for the project. **Examples:**<br>• `Literacy`<br>• `Literature & Writing, Social Sciences` |
| **`project_resource_summary`** | An explanation of the resources needed for the project. **Example:**<br>• `My students need hands on literacy materials to manage sensory needs!` |
| **`project_essay_1`** | First application essay[*] |
| **`project_essay_2`** | Second application essay[*] |
| **`project_essay_3`** | Third application essay[*] |

| Feature | Description |
|---|---|
| `project_essay_4` | Fourth application essay |
| `project_submitted_datetime` | Datetime when project application was submitted. **Example:** `2016-04-28 12:43:56.245` |
| `teacher_id` | A unique identifier for the teacher of the proposed project. **Example:** `bdf8baa8fedef6bfeec7ae4ff1c15c56` |
| `teacher_prefix` | Teacher's title. One of the following enumerated values:<br>• `nan`<br>• `Dr.`<br>• `Mr.`<br>• `Mrs.`<br>• `Ms.`<br>• `Teacher.` |
| `teacher_number_of_previously_posted_projects` | Number of project applications previously submitted by the same teacher. **Example:** `2` |

* See the section **Notes on the Essay Data** for more details about these features.

Additionally, the `resources.csv` data set provides more data about the resources required for each project. Each line in this file represents a resource required by a project:

| Feature | Description |
|---|---|
| `id` | A `project_id` value from the `train.csv` file. **Example:** `p036502` |
| `description` | Desciption of the resource. **Example:** `Tenor Saxophone Reeds, Box of 25` |
| `quantity` | Quantity of the resource required. **Example:** `3` |
| `price` | Price of the resource required. **Example:** `9.95` |

**Note:** Many projects require multiple resources. The `id` value corresponds to a `project_id` in train.csv, so you use it as a key to retrieve all resources needed for a project:

The data set contains the following label (the value you will attempt to predict):

| Label | Description |
|---|---|
| `project_is_approved` | A binary flag indicating whether DonorsChoose approved the project. A value of `0` indicates the project was not approved, and a value of `1` indicates the project was approved. |

## Notes on the Essay Data

Prior to May 17, 2016, the prompts for the essays were as follows:

- __project_essay_1:__ "Introduce us to your classroom"
- __project_essay_2:__ "Tell us more about your students"
- __project_essay_3:__ "Describe how your students will use the materials you're requesting"
- __project_essay_3:__ "Close by sharing why your project will make a difference"

Starting on May 17, 2016, the number of essays was reduced from 4 to 2, and the prompts for the first 2 essays were changed to the following:

- __project_essay_1:__ "Describe your students: What makes your students special? Specific details about their background, your neighborhood, and your school are all helpful."
- __project_essay_2:__ "About your project: How will these materials make a difference in your students' learning and improve their school lives?"

For all projects with project_submitted_datetime of 2016-05-17 and later, the values of project_essay_3 and project_essay_4 will be NaN.

In [0]:

```
%matplotlib inline
import warnings
warnings.filterwarnings("ignore")
warnings.simplefilter("ignore")
warnings.warn("ignore")
import sqlite3
import pandas as pd
import numpy as np
import nltk
import string
import matplotlib.pyplot as plt
import matplotlib.pyplot as plt
import seaborn as sns
from sklearn.feature_extraction.text import TfidfTransformer
from sklearn.feature_extraction.text import TfidfVectorizer

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

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

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

from tqdm import tqdm
import os
```

In [0]:

```
from sklearn.metrics import accuracy_score
from sklearn.metrics import roc_auc_score
from sklearn.metrics import roc_curve, auc
from sklearn import preprocessing
```

In [3]:

```
from keras.preprocessing.text import one_hot
from keras.preprocessing.sequence import pad_sequences
from keras.models import Sequential
from keras.layers.core import Activation, Dropout, Dense ,Reshape
from keras.layers import Flatten, LSTM,Lambda
from keras.models import Model
from keras.layers.embeddings import Embedding
from keras.preprocessing.text import Tokenizer
from keras.layers import Input
from keras.layers import Concatenate
from keras.utils import to_categorical
from keras.layers import Conv1D, MaxPooling1D
from keras import regularizers
from keras.layers.normalization import BatchNormalization
from keras.utils import plot_model
from keras import optimizers
```

```
Using TensorFlow backend.
```

The default version of TensorFlow in Colab will soon switch to TensorFlow 2.x.
We recommend you upgrade now or ensure your notebook will continue to use TensorFlow 1.x via the `%tensorflow_version 1.x` magic: more info.

## 1.1 Reading Data

In [0]:

```
#https://stackabuse.com/python-for-nlp-creating-multi-data-type-classification-models-with-keras/
#https://www.pyimagesearch.com/2019/01/21/regression-with-keras/
#https://github.com/mmortazavi/EntityEmbedding-Working_Example/blob/master/EntityEmbedding.ipynb
#https://www.pyimagesearch.com/2019/02/04/keras-multiple-inputs-and-mixed-data/
#https://machinelearningmastery.com/cnn-models-for-human-activity-recognition-time-series-classifi
cation/
```

In [0]:

In [0]:

```
preprocessed_data = pd.read_csv('preprocessed_data.csv')
```

In [5]:

```
print("Number of data points in preprocessed data", preprocessed_data.shape)
```

```
Number of data points in preprocessed data (109248, 9)
```

In [6]:

```
preprocessed_data=preprocessed_data.sample(n=100000)
preprocessed_data.head()
```

Out[6]:

| | school_state | teacher_prefix | project_grade_category | teacher_number_of_previously_posted_projects | project_is_approved | clean_ |
|---|---|---|---|---|---|---|
| **69638** | in | mrs | grades_prek_2 | 1 | 1 | he |
| **7741** | wa | mrs | grades_3_5 | 5 | 1 | appl<br>literacy |
| **45671** | ak | mrs | grades_3_5 | 2 | 0 | appl |
| **52272** | wi | mrs | grades_prek_2 | 48 | 1 | appl |
| **23037** | wa | mrs | grades_prek_2 | 2 | 1 | literacy |

In [0]:

```
X=preprocessed_data.drop(columns=['project_is_approved'],axis=1)
y=preprocessed_data['project_is_approved']
```

In [0]:

```
label_encoder = preprocessing.LabelEncoder()
y = label_encoder.fit_transform(y)
```

In [0]:

```
X_1, X_test, y_1, y_test = model_selection.train_test_split(X, y, test_size=0.2, random_state=0,str
atify=y)

# split the train data set into cross validation train and cross validation test
X_train, X_cv, y_train, y_cv = model_selection.train_test_split(X_1, y_1, test_size=0.2, random_sta
te=0,stratify=y_1)
```

In [0]:

```
y_train = to_categorical(y_train)
y_cv    = to_categorical(y_cv)
y_test = to_categorical(y_test)
```

In [0]:

```
tokenizer = Tokenizer()
tokenizer.fit_on_texts(X_train['essay'].values)

X1_tr = np.array(tokenizer.texts_to_sequences(X_train['essay'].values))
X1_cv = np.array(tokenizer.texts_to_sequences(X_cv['essay'].values))
X1_test = np.array(tokenizer.texts_to_sequences(X_test['essay'].values))
```

In [0]:

```
vocab_size = len(tokenizer.word_index) + 1

maxlen = 200

X1_tr = pad_sequences(X1_tr, padding='post', maxlen=maxlen)
X1_cv = pad_sequences(X1_cv, padding='post', maxlen=maxlen)
X1_test = pad_sequences(X1_test, padding='post', maxlen=maxlen)
```

In [13]:

```
print(X1_tr.shape)
print(X1_cv.shape)
print(X1_test.shape)
```

```
(64000, 200)
(16000, 200)
(20000, 200)
```

In [0]:

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

In [0]:

```
embeddings_dictionary = dict()
for word in glove_words:
    vector_dimensions = model[word]
    embeddings_dictionary [word] = vector_dimensions
```

In [0]:

```
embedding_matrix = np.zeros((vocab_size, 300))
for word, index in tokenizer.word_index.items():
    embedding_vector = embeddings_dictionary.get(word)
    if embedding_vector is not None:
        embedding_matrix[index] = embedding_vector
```

In [18]:

```
embedding_matrix.shape
```

Out[18]:

```
(45838, 300)
```

In [19]:

```
input_1 = Input(shape=(maxlen,),name='essay_input')
print(input_1.shape)
input_1_embedding = Embedding(vocab_size, 300, weights=[embedding_matrix], trainable=False )(input_
1)
print(input_1_embedding.shape)
input_1_lstm = LSTM(128,return_sequences=True)(input_1_embedding)
print(input_1_lstm.shape)
input_1_flatten=Flatten()(input_1_lstm)
print(input_1_flatten.shape)
```

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

(?, 200)
WARNING:tensorflow:From /usr/local/lib/python3.6/dist-
packages/keras/backend/tensorflow_backend.py:66: The name tf.get_default_graph is deprecated. Plea
se use tf.compat.v1.get_default_graph instead.

WARNING:tensorflow:From /usr/local/lib/python3.6/dist-
packages/keras/backend/tensorflow_backend.py:4432: The name tf.random_uniform is deprecated. Pleas
e 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. P
lease 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 us
e tf.compat.v1.ConfigProto instead.

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

WARNING:tensorflow:From /usr/local/lib/python3.6/dist-
packages/keras/backend/tensorflow_backend.py:207: The name tf.global_variables is deprecated. Plea
se use tf.compat.v1.global_variables instead.

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

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

(?, 200, 300)
(?, ?, 128)
(?, ?)
```

In [0]:

```
categoricals=['school_state','teacher_prefix','project_grade_category','clean_categories','clean_su
bcategories']
numericals=['teacher_number_of_previously_posted_projects','price']
```

In [0]:

```
embed_cols=[i for i in X_train[categoricals]]

for i in embed_cols:
    print(i,X_train[i].nunique())
```

In [22]:

```
tokenizer = Tokenizer()
tokenizer.fit_on_texts(X_train['school_state'].values)

X2_tr = np.array(tokenizer.texts_to_sequences(X_train['school_state'].values))
X2_cv =  np.array(tokenizer.texts_to_sequences(X_cv['school_state'].values))
X2_test =  np.array(tokenizer.texts_to_sequences(X_test['school_state'].values))
cat_emb_name= 'school_state_Embedding'
no_of_unique_cat  = X_train['school_state'].nunique()
embedding_size = int(min(np.ceil((no_of_unique_cat)/2), 50 ))
input_2 = Input(shape=(1,),name='school_state_input')
print(input_2.shape)
input_2_embedding = Embedding(no_of_unique_cat+1, embedding_size,input_length=1, name=cat_emb_name)
(input_2)
print(input_2_embedding.shape)
input_2_flatten=Flatten()(input_2_embedding)
print(input_2_flatten.shape)
```

```
(?, 1)
(?, 1, 26)
(?, ?)
```

In [23]:

```
tokenizer = Tokenizer()
tokenizer.fit_on_texts(X_train['teacher_prefix'].values)

X3_tr = np.array(tokenizer.texts_to_sequences(X_train['teacher_prefix'].values))
X3_cv = np.array(tokenizer.texts_to_sequences( X_cv['teacher_prefix'].values))
X3_test = np.array(tokenizer.texts_to_sequences(X_test['teacher_prefix'].values))
cat_emb_name= 'teacher_prefix_Embedding'
no_of_unique_cat  = X_train['teacher_prefix'].nunique()
embedding_size = int(min(np.ceil((no_of_unique_cat)/2), 50 ))
input_3 = Input(shape=(1,),name='teacher_prefix_input')
print(input_3.shape)
input_3_embedding = Embedding(no_of_unique_cat+1, embedding_size,input_length=1, name=cat_emb_name)
(input_3)
print(input_3_embedding.shape)
input_3_flatten=Flatten()(input_3_embedding)
print(input_3_flatten.shape)
```

```
(?, 1)
(?, 1, 3)
(?, ?)
```

In [0]:

```
#X_train['project_grade_category']=X_train['project_grade_category'].str.replace('_', '')
#X_cv['project_grade_category']=X_cv['project_grade_category'].str.replace('_', '')
#X_test['project_grade_category']=X_test['project_grade_category'].str.replace('_', '')
```

In [24]:

```
tokenizer = Tokenizer(filters='!"#$%&()*+,-/:;<=>?@[\\]^`{|}~\t\n')
tokenizer.fit_on_texts(X_train['project_grade_category'].values)

X4_tr = np.array(tokenizer.texts_to_sequences(X_train['project_grade_category'].values))
X4_cv = np.array(tokenizer.texts_to_sequences(X_cv['project_grade_category'].values))
X4_test = np.array(tokenizer.texts_to_sequences(X_test['project_grade_category'].values))
cat_emb_name= 'project_grade_category_Embedding'
no_of_unique_cat  = X_train['project_grade_category'].nunique()
embedding_size = int(min(np.ceil((no_of_unique_cat)/2), 50 ))
input_4 = Input(shape=(1,),name='project_grade_category_input')
print(input_4.shape)
input_4_embedding = Embedding(no_of_unique_cat+1, embedding_size,input_length=1,name=cat_emb_name)(
input_4)
print(input_4_embedding.shape)
input_4_flatten=Flatten()(input_4_embedding)
print(input_4_flatten.shape)
```

```
(?, 1)
(?, 1, 2)
(?, ?)
```

In [25]:

```
tokenizer = Tokenizer(filters='!"#$%&()*+-/:;<=>?@[\\]^`{|}~\t\n')
tokenizer.fit_on_texts(X_train['clean_categories'].values)

X5_tr = np.array(tokenizer.texts_to_sequences(X_train['clean_categories'].values))
X5_cv = np.array(tokenizer.texts_to_sequences(X_cv['clean_categories'].values))
X5_test = np.array(tokenizer.texts_to_sequences(X_test['clean_categories'].values))
maxlen=max(len(l) for l in X5_tr)
X5_tr = pad_sequences(X5_tr, padding='post', maxlen=maxlen)
X5_cv = pad_sequences(X5_cv, padding='post', maxlen=maxlen)
X5_test = pad_sequences(X5_test, padding='post', maxlen=maxlen)
cat_emb_name= 'clean_categories_Embedding'
no_of_unique_cat  = X_train['clean_categories'].nunique()
embedding_size = int(min(np.ceil((no_of_unique_cat)/2), 50 ))
input_5 = Input(shape=(maxlen,),name='clean_categories_input')
```

```
print(input_5.shape)
input_5_embedding = Embedding(no_of_unique_cat+1,
embedding_size,input_length=maxlen,name=cat_emb_name)(input_5)
print(input_5_embedding.shape)
input_5_flatten=Flatten()(input_5_embedding)
print(input_5_flatten.shape)
```

```
(?, 3)
(?, 3, 26)
(?, ?)
```

In [26]:

```
tokenizer = Tokenizer(filters='!"#$%&()*+,-/:;<=>?@[\\]^`{|}~\t\n')
tokenizer.fit_on_texts(X_train['clean_subcategories'].values)

X6_tr = np.array(tokenizer.texts_to_sequences(X_train['clean_subcategories'].values))
X6_cv = np.array(tokenizer.texts_to_sequences(X_cv['clean_subcategories'].values))
X6_test = np.array(tokenizer.texts_to_sequences(X_test['clean_subcategories'].values))
maxlen=max(len(l) for l in X6_tr)
X6_tr = pad_sequences(X6_tr, padding='post', maxlen=maxlen)
X6_cv = pad_sequences(X6_cv, padding='post', maxlen=maxlen)
X6_test = pad_sequences(X6_test, padding='post', maxlen=maxlen)
cat_emb_name= 'clean_subcategories_Embedding'
no_of_unique_cat  = X_train['clean_subcategories'].nunique()
embedding_size = int(min(np.ceil((no_of_unique_cat)/2), 50 ))
input_6 = Input(shape=(maxlen,),name='clean_subcategories_input')
print(input_6.shape)
input_6_embedding = Embedding(no_of_unique_cat+1, embedding_size,input_length=maxlen, name=cat_emb_
name)(input_6)
print(input_6_embedding.shape)
input_6_flatten=Flatten()(input_6_embedding)
print(input_6_flatten.shape)
```

```
(?, 3)
(?, 3, 50)
(?, ?)
```

In [27]:

```
X7_tr = preprocessing.normalize(X_train[['teacher_number_of_previously_posted_projects', 'price']]
)
X7_cv = preprocessing.normalize(X_cv[['teacher_number_of_previously_posted_projects', 'price']])
X7_test = preprocessing.normalize(X_test[['teacher_number_of_previously_posted_projects', 'price']
])
input_7 = Input(shape=(len(X_train[numericals].columns),),name='numerical_input')
print(input_7.shape)
input_7_dense = Dense(128)(input_7)
print(input_7_dense.shape)
```

```
(?, 2)
(?, 128)
```

In [28]:

```
print(X7_tr.shape)
print(X7_cv.shape)
print(X7_test.shape)
```

```
(64000, 2)
(16000, 2)
(20000, 2)
```

In [0]:

```
#At the end we concatenate altogther and add other Dense layers
output_1 = Concatenate()
([input_1_flatten,input_2_flatten,input_3_flatten,input_4_flatten,input_5_flatten,input_6_flatten,
input_7_dense])
output_1 = Dense(512,activation='relu',kernel_initializer='he_uniform')(output_1)
```

```
output_1= Dropout(0.2)(output_1)
output_1 = Dense(256,activation='relu',kernel_initializer='he_uniform')(output_1)
output_1= Dropout(0.3)(output_1)
output_1 = Dense(128,activation='relu',kernel_initializer='he_uniform')(output_1)
output_1= Dropout(0.4)(output_1)
output_1 = Dense(2, activation='softmax')(output_1)
```

In [0]:

```
#https://stackoverflow.com/questions/41032551/how-to-compute-receiving-operating-characteristic-ro
c-and-auc-in-keras
import tensorflow as tf
def auroc(y_true, y_pred):
    return tf.py_func(roc_auc_score, (y_true, y_pred), tf.double)
```

In [58]:

```
model = Model(inputs=[input_1,input_2,input_3,input_4,input_5,input_6,input_7], outputs=output_1)
model.compile(loss='binary_crossentropy', optimizer=optimizers.SGD(lr=0.001) ,metrics=[auroc])
model.summary()
```

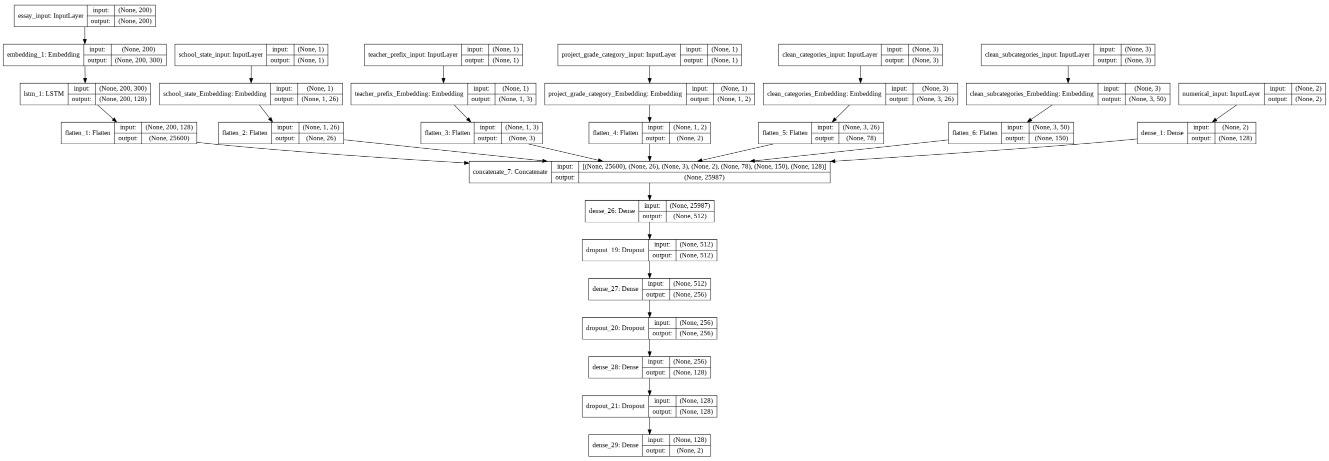
```
Model: "model_10"
__________________________________________________________________________________________________
Layer (type)                    Output Shape         Param #     Connected to                     
==================================================================================================
essay_input (InputLayer)        (None, 200)          0                                            
__________________________________________________________________________________________________
embedding_1 (Embedding)         (None, 200, 300)     13751400    essay_input[0][0]                
__________________________________________________________________________________________________
school_state_input (InputLayer) (None, 1)            0                                            
__________________________________________________________________________________________________
teacher_prefix_input (InputLaye (None, 1)            0                                            
__________________________________________________________________________________________________
project_grade_category_input (I (None, 1)            0                                            
__________________________________________________________________________________________________
clean_categories_input (InputLa (None, 3)            0                                            
__________________________________________________________________________________________________
clean_subcategories_input (Inpu (None, 3)            0                                            
__________________________________________________________________________________________________
lstm_1 (LSTM)                   (None, 200, 128)     219648      embedding_1[0][0]                
__________________________________________________________________________________________________
school_state_Embedding (Embeddi (None, 1, 26)        1352        school_state_input[0][0]         
__________________________________________________________________________________________________
teacher_prefix_Embedding (Embed (None, 1, 3)         18          teacher_prefix_input[0][0]       
__________________________________________________________________________________________________
project_grade_category_Embeddin (None, 1, 2)         10          project_grade_category_input[0][0
__________________________________________________________________________________________________
clean_categories_Embedding (Emb (None, 3, 26)        1352        clean_categories_input[0][0]     
__________________________________________________________________________________________________
clean_subcategories_Embedding ( (None, 3, 50)        19650       clean_subcategories_input[0][0]  
__________________________________________________________________________________________________
numerical_input (InputLayer)    (None, 2)            0                                            
__________________________________________________________________________________________________
flatten_1 (Flatten)             (None, 25600)        0           lstm_1[0][0]                     
__________________________________________________________________________________________________
flatten_2 (Flatten)             (None, 26)           0           school_state_Embedding[0][0]     
__________________________________________________________________________________________________
flatten_3 (Flatten)             (None, 3)            0           teacher_prefix_Embedding[0][0]   
__________________________________________________________________________________________________
flatten_4 (Flatten)             (None, 2)            0           project_grade_category_Embedding[
__________________________________________________________________________________________________
flatten_5 (Flatten)             (None, 78)           0           clean_categories_Embedding[0][0] 
__________________________________________________________________________________________________
flatten_6 (Flatten)             (None, 150)          0           clean_subcategories_Embedding[0][
__________________________________________________________________________________________________
dense_1 (Dense)                 (None, 128)          384         numerical_input[0][0]            
__________________________________________________________________________________________________
concatenate_7 (Concatenate)     (None, 25987)        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_26 (Dense)                (None, 512)          13305856    concatenate_7[0][0]
________________________________________________________________________________________________________________________
dropout_19 (Dropout)            (None, 512)          0           dense_26[0][0]
________________________________________________________________________________________________________________________
dense_27 (Dense)                (None, 256)          131328      dropout_19[0][0]
________________________________________________________________________________________________________________________
dropout_20 (Dropout)            (None, 256)          0           dense_27[0][0]
________________________________________________________________________________________________________________________
dense_28 (Dense)                (None, 128)          32896       dropout_20[0][0]
________________________________________________________________________________________________________________________
dropout_21 (Dropout)            (None, 128)          0           dense_28[0][0]
________________________________________________________________________________________________________________________
dense_29 (Dense)                (None, 2)            258         dropout_21[0][0]
========================================================================================================================
Total params: 27,464,152
Trainable params: 13,712,752
Non-trainable params: 13,751,400
________________________________________________________________________________________________________________________
```

In [76]:

```
plot_model(model, to_file='model_1.png', show_shapes=True, show_layer_names=True)
```

Out[76]:



essay_input: InputLayer | input: (None, 200) | output: (None, 200)
embedding_1: Embedding | input: (None, 200) | output: (None, 200, 300)
school_state_input: InputLayer | input: (None, 1) | output: (None, 1)
teacher_prefix_input: InputLayer | input: (None, 1) | output: (None, 1)
project_grade_category_input: InputLayer | input: (None, 1) | output: (None, 1)
clean_categories_input: InputLayer | input: (None, 3) | output: (None, 3)
clean_subcategories_input: InputLayer | input: (None, 3) | output: (None, 3)
lstm_1: LSTM | input: (None, 200, 300) | output: (None, 200, 128)
school_state_Embedding: Embedding | input: (None, 1) | output: (None, 1, 26)
teacher_prefix_Embedding: Embedding | input: (None, 1) | output: (None, 1, 3)
project_grade_category_Embedding: Embedding | input: (None, 1) | output: (None, 1, 2)
clean_categories_Embedding: Embedding | input: (None, 3) | output: (None, 3, 26)
clean_subcategories_Embedding: Embedding | input: (None, 3) | output: (None, 3, 50)
numerical_input: InputLayer | input: (None, 2) | output: (None, 2)
flatten_1: Flatten | input: (None, 200, 128) | output: (None, 25600)
flatten_2: Flatten | input: (None, 1, 26) | output: (None, 26)
flatten_3: Flatten | input: (None, 1, 3) | output: (None, 3)
flatten_4: Flatten | input: (None, 1, 2) | output: (None, 2)
flatten_5: Flatten | input: (None, 3, 26) | output: (None, 78)
flatten_6: Flatten | input: (None, 3, 50) | output: (None, 150)
dense_1: Dense | input: (None, 2) | output: (None, 128)
concatenate_7: Concatenate | input: [(None, 25600), (None, 26), (None, 3), (None, 2), (None, 78), (None, 150), (None, 128)] | output: (None, 25987)
dense_26: Dense | input: (None, 25987) | output: (None, 512)
dropout_19: Dropout | input: (None, 512) | output: (None, 512)
dense_27: Dense | input: (None, 512) | output: (None, 256)
dropout_20: Dropout | input: (None, 256) | output: (None, 256)
dense_28: Dense | input: (None, 256) | output: (None, 128)
dropout_21: Dropout | input: (None, 128) | output: (None, 128)
dense_29: Dense | input: (None, 128) | output: (None, 2)

In [71]:

```
history = model.fit(x=[X1_tr,X2_tr,X3_tr,X4_tr, X5_tr, X6_tr,X7_tr], y=y_train, validation_data=([X
1_cv,X2_cv,X3_cv,X4_cv, X5_cv, X6_cv,X7_cv],y_cv),epochs=10,batch_size=500,verbose=2)
```

```
Train on 64000 samples, validate on 16000 samples
Epoch 1/10
 - 65s - loss: 0.3529 - auroc: 0.7886 - val_loss: 0.3773 - val_auroc: 0.7444
Epoch 2/10
 - 64s - loss: 0.3517 - auroc: 0.7915 - val_loss: 0.3779 - val_auroc: 0.7443
Epoch 3/10
 - 65s - loss: 0.3528 - auroc: 0.7879 - val_loss: 0.3777 - val_auroc: 0.7443
Epoch 4/10
 - 65s - loss: 0.3505 - auroc: 0.7917 - val_loss: 0.3778 - val_auroc: 0.7446
Epoch 5/10
 - 64s - loss: 0.3519 - auroc: 0.7894 - val_loss: 0.3775 - val_auroc: 0.7449
Epoch 6/10
 - 65s - loss: 0.3518 - auroc: 0.7903 - val_loss: 0.3789 - val_auroc: 0.7447
Epoch 7/10
 - 64s - loss: 0.3501 - auroc: 0.7939 - val_loss: 0.3789 - val_auroc: 0.7448
Epoch 8/10
 - 64s - loss: 0.3506 - auroc: 0.7926 - val_loss: 0.3777 - val_auroc: 0.7451
Epoch 9/10
 - 64s - loss: 0.3504 - auroc: 0.7926 - val_loss: 0.3773 - val_auroc: 0.7451
Epoch 10/10
 - 65s - loss: 0.3492 - auroc: 0.7947 - val_loss: 0.3788 - val_auroc: 0.7450
```

In [0]:

```
score = model.evaluate(x=[X1_test,X2_test,X3_test,X4_test, X5_test, X6_test, X7_test], y=y_test, ve
rbose=2,batch_size=500)
```

In [73]:

```
print("Test Loss:", score[0])
print("Test AUC:", score[1])
```

```
Test Loss: 0.3780378520488739
Test AUC: 0.7460865303064443
```

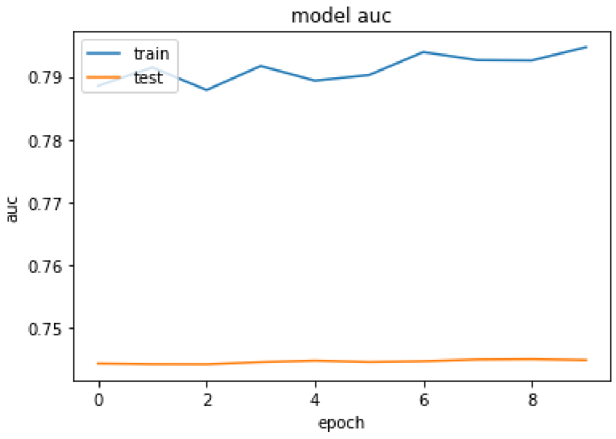
In [74]:

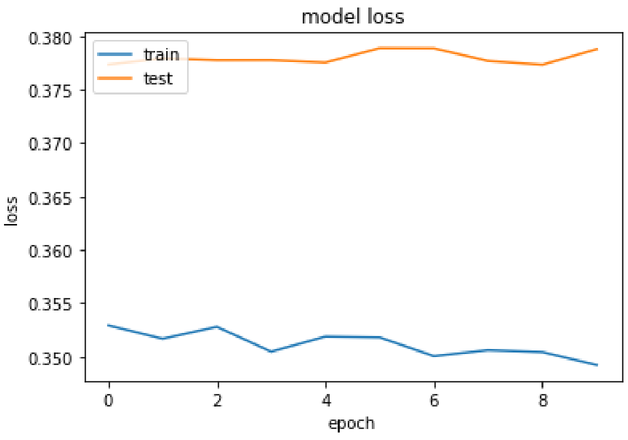
```
plt.plot(history.history['auroc'])
plt.plot(history.history['val_auroc'])

plt.title('model auc')
plt.ylabel('auc')
plt.xlabel('epoch')
plt.legend(['train','test'], loc='upper left')
plt.show()

plt.plot(history.history['loss'])
plt.plot(history.history['val_loss'])

plt.title('model loss')
plt.ylabel('loss')
plt.xlabel('epoch')
plt.legend(['train','test'], loc='upper left')
plt.show()
```





In [0]:

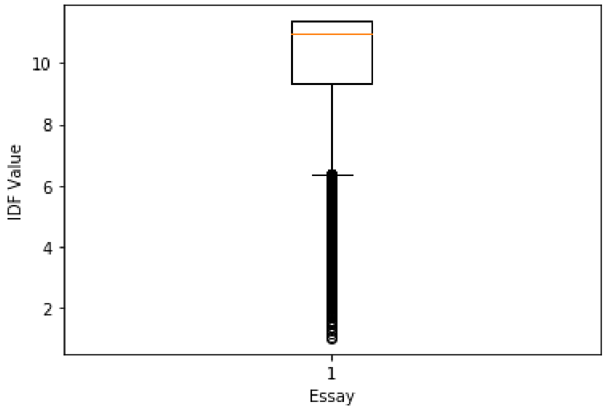
```
# serialize weights to HDF5
model.save_weights("model_1.h5")
print("Saved model to disk")
```

```
Saved model to disk
```

## Model 2

In [75]:

```
from sklearn.feature_extraction.text import TfidfVectorizer
import seaborn as sns
vectorizer = TfidfVectorizer()
vectorizer.fit(X_train['essay'].values)
plt.boxplot(list(vectorizer.idf_))
plt.xlabel('Essay')
plt.ylabel('IDF Value')
plt.show()
```



In [77]:

```
tenth_percentile=np.quantile((vectorizer.idf_),0.10)
ninty_percentile=np.quantile((vectorizer.idf_),0.90)
print(tenth_percentile)
print(ninty_percentile)
dictionary = dict(zip(vectorizer.get_feature_names(), list(vectorizer.idf_)))
filterred_words=[]
for k,v in dictionary.items():
    if v > 2.0 and v < 11.0:
        filterred_words.append(k)

len(filterred_words)
```

```
7.375306104990596
11.373506806659794
```

Out[77]:

```
27930
```

In [0]:

```
tokenizer = Tokenizer()
tokenizer.fit_on_texts(filterred_words)

X8_tr = np.array(tokenizer.texts_to_sequences(X_train['essay'].values))
X8_cv = np.array(tokenizer.texts_to_sequences(X_cv['essay'].values))
X8_test = np.array(tokenizer.texts_to_sequences(X_test['essay'].values))
```

In [0]:

```
vocab_size = len(tokenizer.word_index) + 1

maxlen = 200

X8_tr = pad_sequences(X8_tr, padding='post', maxlen=maxlen)
X8_cv = pad_sequences(X8_cv, padding='post', maxlen=maxlen)
X8_test = pad_sequences(X8_test, padding='post', maxlen=maxlen)
```

In [80]:

```
print(X8_tr.shape)
print(X8_cv.shape)
print(X8_test.shape)
```

```
(64000, 200)
(16000, 200)
(20000, 200)
```

In [0]:

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

In [0]:

```
embeddings_dictionary = dict()
for word in glove_words:
    vector_dimensions = model[word]
    embeddings_dictionary [word] = vector_dimensions
```

In [0]:

```
embedding_matrix = np.zeros((vocab_size, 300))
for word, index in tokenizer.word_index.items():
    embedding_vector = embeddings_dictionary.get(word)
    if embedding_vector is not None:
        embedding_matrix[index] = embedding_vector
```

In [84]:

```
embedding_matrix.shape
```

Out[84]:

```
(27931, 300)
```

In [85]:

```
input_8 = Input(shape=(maxlen,),name='essay_tfidf_input')
print(input_8.shape)
input_8_embedding = Embedding(vocab_size, 300, weights=[embedding_matrix], trainable=False )(input_
8)
print(input_8_embedding.shape)
input_8_lstm = LSTM(128,return_sequences=True)(input_8_embedding)
print(input_8_lstm.shape)
input_8_flatten=Flatten()(input_8_lstm)
print(input_8_flatten.shape)
```

```
(?, 200)
(?, 200, 300)
(?, ?, 128)
(?, ?)
```

In [0]:

```
#At the end we concatenate altogther and add other Dense layers
output_2 = Concatenate()
([input_8_flatten,input_2_flatten,input_3_flatten,input_4_flatten,input_5_flatten,input_6_flatten,
input_7_dense])
output_2 = BatchNormalization()(output_2)
output_2 = Dense(512, kernel_initializer="he_uniform",activation='relu')(output_2)
output_2 = BatchNormalization()(output_2)
output_2= Dropout(0.5)(output_2)
output_2 = Dense(256, kernel_initializer="he_uniform",activation='relu')(output_2)
output_2 = BatchNormalization()(output_2)
```

```
output_2 = BatchNormalization()(output_2)
output_2= Dropout(0.5)(output_2)
output_2 = Dense(128, kernel_initializer="he_uniform", activation='relu')(output_2)
output_2 = BatchNormalization()(output_2)
output_2= Dropout(0.5)(output_2)
output_2 = Dense(2, activation='softmax')(output_2)
```

In [95]:

```
model_2 = Model(inputs=[input_8,input_2,input_3,input_4,input_5,input_6,input_7], outputs=output_2
)
model_2.compile(loss='binary_crossentropy', optimizer=optimizers.SGD(lr=0.001) ,metrics=[auroc])
model_2.summary()
```

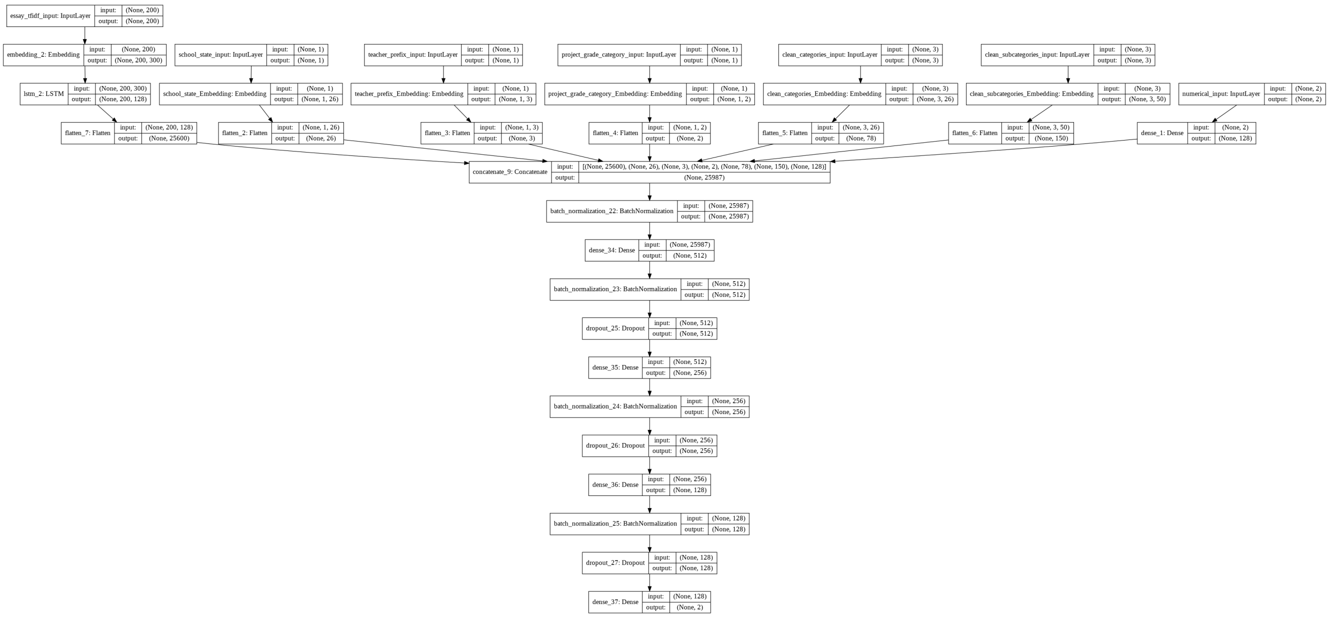
```
Model: "model_13"
__________________________________________________________________________________________________
Layer (type)                    Output Shape         Param #     Connected to
==================================================================================================
essay_tfidf_input (InputLayer)  (None, 200)          0
__________________________________________________________________________________________________
embedding_2 (Embedding)         (None, 200, 300)     8379300     essay_tfidf_input[0][0]
__________________________________________________________________________________________________
school_state_input (InputLayer) (None, 1)            0
__________________________________________________________________________________________________
teacher_prefix_input (InputLaye (None, 1)            0
__________________________________________________________________________________________________
project_grade_category_input (I (None, 1)            0
__________________________________________________________________________________________________
clean_categories_input (InputLa (None, 3)            0
__________________________________________________________________________________________________
clean_subcategories_input (Inpu (None, 3)            0
__________________________________________________________________________________________________
lstm_2 (LSTM)                   (None, 200, 128)     219648      embedding_2[0][0]
__________________________________________________________________________________________________
school_state_Embedding (Embeddi (None, 1, 26)        1352        school_state_input[0][0]
__________________________________________________________________________________________________
teacher_prefix_Embedding (Embed (None, 1, 3)         18          teacher_prefix_input[0][0]
__________________________________________________________________________________________________
project_grade_category_Embeddin (None, 1, 2)         10          project_grade_category_input[0][0
__________________________________________________________________________________________________
clean_categories_Embedding (Emb (None, 3, 26)        1352        clean_categories_input[0][0]
__________________________________________________________________________________________________
clean_subcategories_Embedding ( (None, 3, 50)        19650       clean_subcategories_input[0][0]
__________________________________________________________________________________________________
numerical_input (InputLayer)    (None, 2)            0
__________________________________________________________________________________________________
flatten_7 (Flatten)             (None, 25600)        0           lstm_2[0][0]
__________________________________________________________________________________________________
flatten_2 (Flatten)             (None, 26)           0           school_state_Embedding[0][0]
__________________________________________________________________________________________________
flatten_3 (Flatten)             (None, 3)            0           teacher_prefix_Embedding[0][0]
__________________________________________________________________________________________________
flatten_4 (Flatten)             (None, 2)            0           project_grade_category_Embedding[
__________________________________________________________________________________________________
flatten_5 (Flatten)             (None, 78)           0           clean_categories_Embedding[0][0]
__________________________________________________________________________________________________
flatten_6 (Flatten)             (None, 150)          0           clean_subcategories_Embedding[0][
__________________________________________________________________________________________________
dense_1 (Dense)                 (None, 128)          384         numerical_input[0][0]
__________________________________________________________________________________________________
concatenate_9 (Concatenate)     (None, 25987)        0           flatten_7[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]
__________________________________________________________________________________________________
batch_normalization_22 (BatchNo (None, 25987)        103948      concatenate_9[0][0]
__________________________________________________________________________________________________
dense_34 (Dense)                (None, 512)          13305856    batch_normalization_22[0][0]
__________________________________________________________________________________________________
batch_normalization_23 (BatchNo (None, 512)          2048        dense_34[0][0]
__________________________________________________________________________________________________
dropout_25 (Dropout)            (None, 512)          0           batch_normalization_23[0][0]
__________________________________________________________________________________________________
```

```
dense_35 (Dense)                (None, 256)          131328      dropout_25[0][0]
__________________________________________________________________________________________________
batch_normalization_24 (BatchNo (None, 256)          1024        dense_35[0][0]
__________________________________________________________________________________________________
dropout_26 (Dropout)            (None, 256)          0           batch_normalization_24[0][0]
__________________________________________________________________________________________________
dense_36 (Dense)                (None, 128)          32896       dropout_26[0][0]
__________________________________________________________________________________________________
batch_normalization_25 (BatchNo (None, 128)          512         dense_36[0][0]
__________________________________________________________________________________________________
dropout_27 (Dropout)            (None, 128)          0           batch_normalization_25[0][0]
__________________________________________________________________________________________________
dense_37 (Dense)                (None, 2)            258         dropout_27[0][0]
==================================================================================================
Total params: 22,199,584
Trainable params: 13,766,518
Non-trainable params: 8,433,066
__________________________________________________________________________________________________
```

In [97]:

```python
plot_model(model_2, to_file='model_2.png', show_shapes=True, show_layer_names=True)
```

Out[97]:



essay_tfidf_input: InputLayer | input: (None, 200) | output: (None, 200)
embedding_2: Embedding | input: (None, 200) | output: (None, 200, 300)
school_state_input: InputLayer | input: (None, 1) | output: (None, 1)
teacher_prefix_input: InputLayer | input: (None, 1) | output: (None, 1)
project_grade_category_input: InputLayer | input: (None, 1) | output: (None, 1)
clean_categories_input: InputLayer | input: (None, 3) | output: (None, 3)
clean_subcategories_input: InputLayer | input: (None, 3) | output: (None, 3)
lstm_2: LSTM | input: (None, 200, 300) | output: (None, 200, 128)
school_state_Embedding: Embedding | input: (None, 1) | output: (None, 1, 26)
teacher_prefix_Embedding: Embedding | input: (None, 1) | output: (None, 1, 3)
project_grade_category_Embedding: Embedding | input: (None, 1) | output: (None, 1, 2)
clean_categories_Embedding: Embedding | input: (None, 3) | output: (None, 3, 26)
clean_subcategories_Embedding: Embedding | input: (None, 3) | output: (None, 3, 50)
numerical_input: InputLayer | input: (None, 2) | output: (None, 2)
flatten_7: Flatten | input: (None, 200, 128) | output: (None, 25600)
flatten_2: Flatten | input: (None, 1, 26) | output: (None, 26)
flatten_3: Flatten | input: (None, 1, 3) | output: (None, 3)
flatten_4: Flatten | input: (None, 1, 2) | output: (None, 2)
flatten_5: Flatten | input: (None, 3, 26) | output: (None, 78)
flatten_6: Flatten | input: (None, 3, 50) | output: (None, 150)
dense_1: Dense | input: (None, 2) | output: (None, 128)
concatenate_9: Concatenate | input: [(None, 25600), (None, 26), (None, 3), (None, 2), (None, 78), (None, 150), (None, 128)] | output: (None, 25987)
batch_normalization_22: BatchNormalization | input: (None, 25987) | output: (None, 25987)
dense_34: Dense | input: (None, 25987) | output: (None, 512)
batch_normalization_23: BatchNormalization | input: (None, 512) | output: (None, 512)
dropout_25: Dropout | input: (None, 512) | output: (None, 512)
dense_35: Dense | input: (None, 512) | output: (None, 256)
batch_normalization_24: BatchNormalization | input: (None, 256) | output: (None, 256)
dropout_26: Dropout | input: (None, 256) | output: (None, 256)
dense_36: Dense | input: (None, 256) | output: (None, 128)
batch_normalization_25: BatchNormalization | input: (None, 128) | output: (None, 128)
dropout_27: Dropout | input: (None, 128) | output: (None, 128)
dense_37: Dense | input: (None, 128) | output: (None, 2)

In [102]:

```python
history = model_2.fit(x=[X8_tr,X2_tr,X3_tr,X4_tr, X5_tr, X6_tr,X7_tr], y=y_train, validation_data=(
[X8_cv,X2_cv,X3_cv,X4_cv, X5_cv, X6_cv,X7_cv],y_cv),epochs=30,batch_size=500,verbose=2)
```

```
Train on 64000 samples, validate on 16000 samples
Epoch 1/30
 - 67s - loss: 0.4107 - auroc: 0.6641 - val_loss: 0.4014 - val_auroc: 0.6906
Epoch 2/30
 - 68s - loss: 0.4110 - auroc: 0.6649 - val_loss: 0.4012 - val_auroc: 0.6908
Epoch 3/30
 - 67s - loss: 0.4113 - auroc: 0.6662 - val_loss: 0.4011 - val_auroc: 0.6911
Epoch 4/30
 - 67s - loss: 0.4104 - auroc: 0.6671 - val_loss: 0.4007 - val_auroc: 0.6916
Epoch 5/30
 - 67s - loss: 0.4098 - auroc: 0.6655 - val_loss: 0.4003 - val_auroc: 0.6920
Epoch 6/30
 - 67s - loss: 0.4099 - auroc: 0.6655 - val_loss: 0.4000 - val_auroc: 0.6924
Epoch 7/30
 - 67s - loss: 0.4094 - auroc: 0.6701 - val_loss: 0.3998 - val_auroc: 0.6927
Epoch 8/30
 - 67s - loss: 0.4076 - auroc: 0.6718 - val_loss: 0.3995 - val_auroc: 0.6928
Epoch 9/30
 - 67s - loss: 0.4078 - auroc: 0.6731 - val_loss: 0.3996 - val_auroc: 0.6930
Epoch 10/30
 - 67s - loss: 0.4085 - auroc: 0.6694 - val_loss: 0.3992 - val_auroc: 0.6931
 - 67s - loss: 0.4066 - auroc: 0.6694 - val_loss: 0.3992 - val_auroc: 0.6931
Epoch 11/30
 - 67s - loss: 0.4078 - auroc: 0.6706 - val_loss: 0.3992 - val_auroc: 0.6934
Epoch 12/30
 - 67s - loss: 0.4053 - auroc: 0.6772 - val_loss: 0.3989 - val_auroc: 0.6937
Epoch 13/30
 - 67s - loss: 0.4069 - auroc: 0.6733 - val_loss: 0.3988 - val_auroc: 0.6940
Epoch 14/30
 - 67s - loss: 0.4071 - auroc: 0.6723 - val_loss: 0.3984 - val_auroc: 0.6942
Epoch 15/30
 - 67s - loss: 0.4047 - auroc: 0.6793 - val_loss: 0.3983 - val_auroc: 0.6945
Epoch 16/30
 - 67s - loss: 0.4056 - auroc: 0.6762 - val_loss: 0.3981 - val_auroc: 0.6949
Epoch 17/30
 - 67s - loss: 0.4062 - auroc: 0.6740 - val_loss: 0.3981 - val_auroc: 0.6952
Epoch 18/30
 - 67s - loss: 0.4057 - auroc: 0.6758 - val_loss: 0.3979 - val_auroc: 0.6953
Epoch 19/30
 - 68s - loss: 0.4042 - auroc: 0.6798 - val_loss: 0.3978 - val_auroc: 0.6955
Epoch 20/30
 - 67s - loss: 0.4042 - auroc: 0.6794 - val_loss: 0.3976 - val_auroc: 0.6957
Epoch 21/30
 - 67s - loss: 0.4034 - auroc: 0.6799 - val_loss: 0.3973 - val_auroc: 0.6960
Epoch 22/30
 - 67s - loss: 0.4044 - auroc: 0.6786 - val_loss: 0.3972 - val_auroc: 0.6962
Epoch 23/30
 - 67s - loss: 0.4040 - auroc: 0.6784 - val_loss: 0.3971 - val_auroc: 0.6964
Epoch 24/30
 - 68s - loss: 0.4023 - auroc: 0.6848 - val_loss: 0.3970 - val_auroc: 0.6966
Epoch 25/30
 - 67s - loss: 0.4031 - auroc: 0.6815 - val_loss: 0.3967 - val_auroc: 0.6969
Epoch 26/30
 - 67s - loss: 0.4029 - auroc: 0.6819 - val_loss: 0.3963 - val_auroc: 0.6971
Epoch 27/30
 - 67s - loss: 0.4021 - auroc: 0.6843 - val_loss: 0.3960 - val_auroc: 0.6974
Epoch 28/30
 - 68s - loss: 0.4026 - auroc: 0.6824 - val_loss: 0.3962 - val_auroc: 0.6975
Epoch 29/30
 - 67s - loss: 0.4023 - auroc: 0.6829 - val_loss: 0.3961 - val_auroc: 0.6977
Epoch 30/30
 - 67s - loss: 0.4012 - auroc: 0.6866 - val_loss: 0.3959 - val_auroc: 0.6981
```

In [0]:

```
score = model_2.evaluate(x=[X8_test,X2_test,X3_test,X4_test, X5_test, X6_test, X7_test], y=y_test,
verbose=2,batch_size=500)
```

In [104]:

```
print("Test Loss:", score[0])
print("Test AUC:", score[1])
```

```
Test Loss: 0.39928831085562705
Test AUC: 0.6863849997066904
```

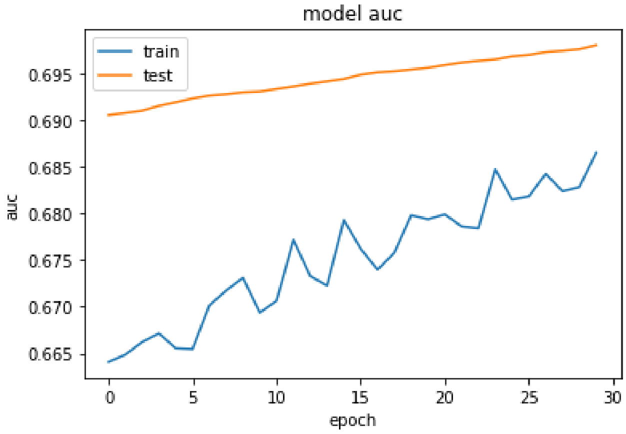
In [105]:

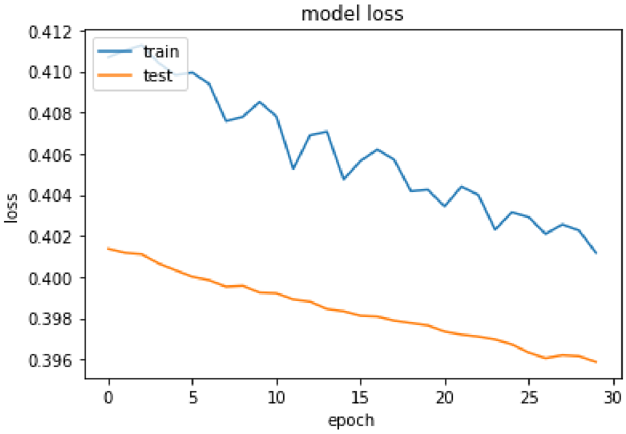
```
plt.plot(history.history['auroc'])
plt.plot(history.history['val_auroc'])

plt.title('model auc')
plt.ylabel('auc')
plt.xlabel('epoch')
plt.legend(['train','test'], loc='upper left')
plt.show()

plt.plot(history.history['loss'])
plt.plot(history.history['val_loss'])

plt.title('model loss')
plt.ylabel('loss')
plt.xlabel('epoch')
plt.legend(['train','test'], loc='upper left')
plt.show()
```





In [0]:

```
# serialize weights to HDF5
model_2.save_weights("model_2.h5")
print("Saved model to disk")
```

```
Saved model to disk
```

## Model 3

In [106]:

```
numeric_tr = np.hstack((X2_tr,X3_tr,X4_tr,X5_tr,X6_tr,X7_tr))
numeric_cv = np.hstack((X2_cv,X3_cv,X4_cv,X5_cv,X6_cv,X7_cv))
numeric_test = np.hstack((X2_test,X3_test,X4_test,X5_test,X6_test,X7_test))
print(numeric_tr.shape)
print(numeric_cv.shape)
print(numeric_test.shape)
```

```
(64000, 11)
(16000, 11)
(20000, 11)
```

In [107]:

```
numeric_tr=np.expand_dims(numeric_tr,axis=2)
numeric_cv=np.expand_dims(numeric_cv,axis=2)
numeric_test=np.expand_dims(numeric_test,axis=2)
print(numeric_tr.shape)
print(numeric_cv.shape)
print(numeric_test.shape)
```

```
(64000, 11, 1)
(16000, 11, 1)
```

```
(10000, 11, 1)
(20000, 11, 1)
```

In [108]:

```
input_9 = Input(shape=(numeric_tr.shape[1],numeric_tr.shape[2],),name='combined_input')
print(input_9.shape)
```

```
(?, 11, 1)
```

In [109]:

```
#At the end we concatenate altogther and add other Dense layers

#output_3=tf.reshape(output_3,[-1,output_3.shape[1],output_3.shape[1]])
#print(output_3.shape)
output_3 = Conv1D(128, 3, strides=1,activation="relu",kernel_initializer='he_uniform',padding='same
')(input_9)
output_3 = MaxPooling1D(pool_size=2)(output_3)
output_3= Dropout(0.2)(output_3)
output_3 = Conv1D(64, 3, activation="relu")(output_3)
output_3 = MaxPooling1D(pool_size=2)(output_3)
output_3= Dropout(0.4)(output_3)
output_3 = Flatten()(output_3)

output_4 = Concatenate()([input_1_flatten,output_3])
output_4 = Dense(256, kernel_initializer="he_uniform",activation='relu')(output_4)
output_4= Dropout(0.2)(output_4)
output_4 = Dense(128, kernel_initializer="he_uniform",activation='relu')(output_4)
output_4= Dropout(0.3)(output_4)
output_4 = Dense(64, kernel_initializer="he_uniform", activation='relu')(output_4)
output_4= Dropout(0.4)(output_4)
output_4 = Dense(2, activation='softmax')(output_4)
```

```
WARNING:tensorflow:From /usr/local/lib/python3.6/dist-
packages/keras/backend/tensorflow_backend.py:4267: The name tf.nn.max_pool is deprecated. Please u
se tf.nn.max_pool2d instead.
```

In [110]:

```
from keras import optimizers
model_3 = Model(inputs=[input_1,input_2,input_3,input_4,input_5,input_6,input_9], outputs=output_4
)
model_3.compile(loss='binary_crossentropy', optimizer=optimizers.SGD(lr=0.001, decay=1e-6, momentum
=0.9) ,metrics=[auroc])
model_3.summary()
```

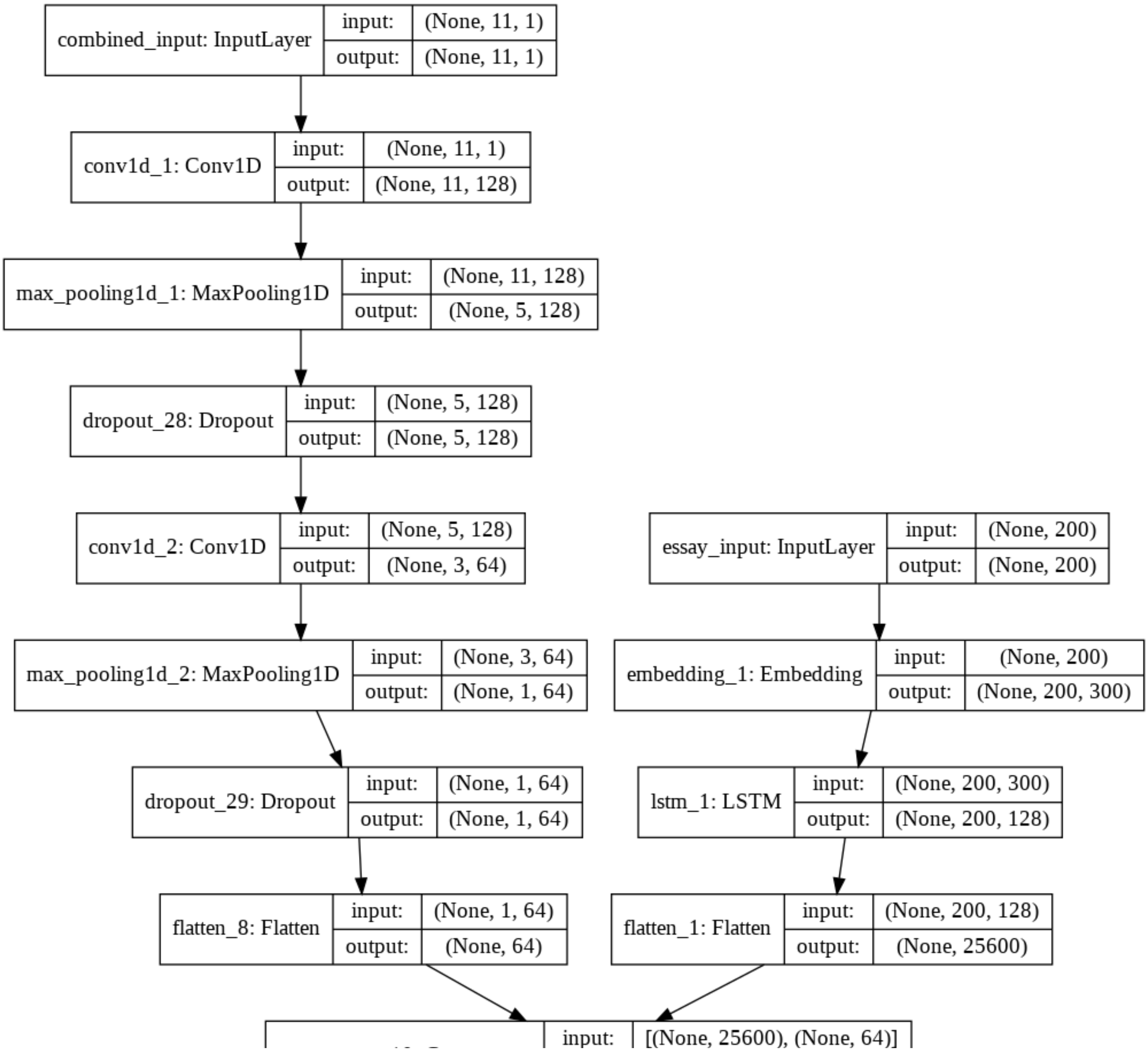
```
Model: "model_14"
__________________________________________________________________________________________________
Layer (type)                    Output Shape         Param #     Connected to                     
==================================================================================================
combined_input (InputLayer)     (None, 11, 1)        0                                            
__________________________________________________________________________________________________
conv1d_1 (Conv1D)               (None, 11, 128)      512         combined_input[0][0]             
__________________________________________________________________________________________________
max_pooling1d_1 (MaxPooling1D)  (None, 5, 128)       0           conv1d_1[0][0]                   
__________________________________________________________________________________________________
dropout_28 (Dropout)            (None, 5, 128)       0           max_pooling1d_1[0][0]            
__________________________________________________________________________________________________
essay_input (InputLayer)        (None, 200)          0                                            
__________________________________________________________________________________________________
conv1d_2 (Conv1D)               (None, 3, 64)        24640       dropout_28[0][0]                 
__________________________________________________________________________________________________
embedding_1 (Embedding)         (None, 200, 300)     13751400    essay_input[0][0]                
__________________________________________________________________________________________________
max_pooling1d_2 (MaxPooling1D)  (None, 1, 64)        0           conv1d_2[0][0]                   
__________________________________________________________________________________________________
lstm_1 (LSTM)                   (None, 200, 128)     219648      embedding_1[0][0]                
__________________________________________________________________________________________________
dropout_29 (Dropout)            (None, 1, 64)        0           max_pooling1d_2[0][0]            
__________________________________________________________________________________________________
flatten_1 (Flatten)             (None, 25600)        0           lstm_1[0][0]                     
```

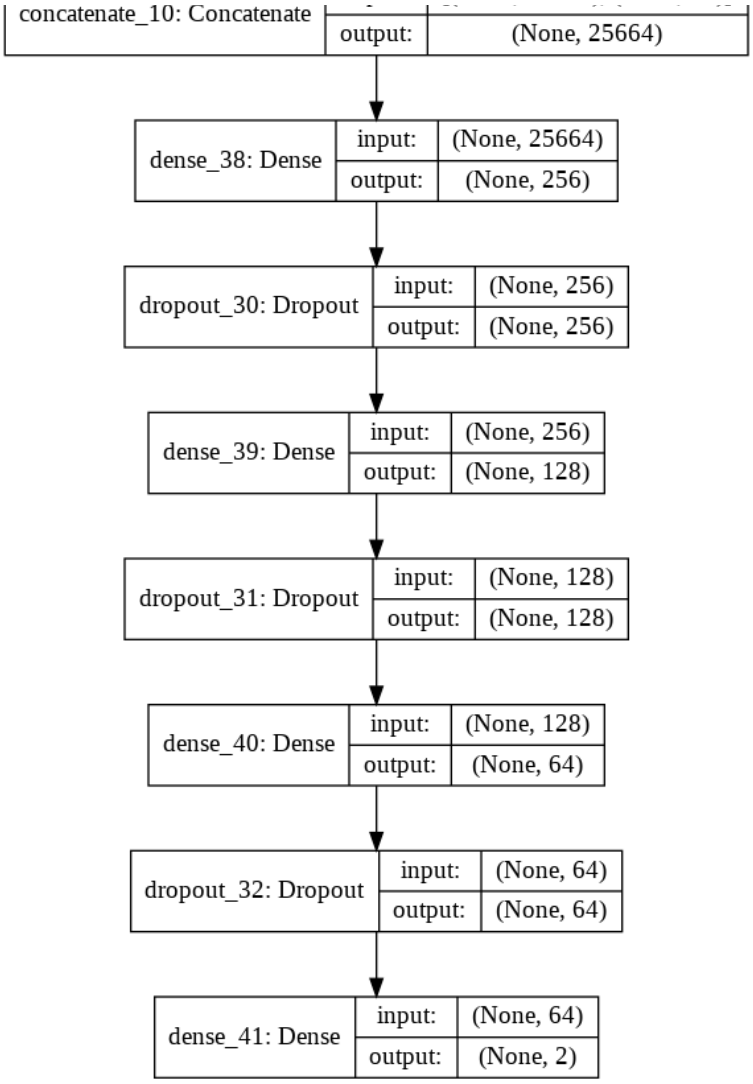
```
flatten_1 (Flatten)             (None, 25600)        0           lstm_1[0][0]
__________________________________________________________________________________________________
flatten_8 (Flatten)             (None, 64)           0           dropout_29[0][0]
__________________________________________________________________________________________________
concatenate_10 (Concatenate)    (None, 25664)        0           flatten_1[0][0]
                                                                 flatten_8[0][0]
__________________________________________________________________________________________________
dense_38 (Dense)                (None, 256)          6570240     concatenate_10[0][0]
__________________________________________________________________________________________________
dropout_30 (Dropout)            (None, 256)          0           dense_38[0][0]
__________________________________________________________________________________________________
dense_39 (Dense)                (None, 128)          32896       dropout_30[0][0]
__________________________________________________________________________________________________
dropout_31 (Dropout)            (None, 128)          0           dense_39[0][0]
__________________________________________________________________________________________________
dense_40 (Dense)                (None, 64)           8256        dropout_31[0][0]
__________________________________________________________________________________________________
dropout_32 (Dropout)            (None, 64)           0           dense_40[0][0]
__________________________________________________________________________________________________
dense_41 (Dense)                (None, 2)            130         dropout_32[0][0]
==================================================================================================
Total params: 20,607,722
Trainable params: 6,856,322
Non-trainable params: 13,751,400
__________________________________________________________________________________________________
```

In [111]:

```python
plot_model(model_3, to_file='model_3.png', show_shapes=True, show_layer_names=True)
```

Out[111]:





In [0]:

```
history = model_3.fit(x=[X1_tr,X2_tr,X3_tr,X4_tr, X5_tr, X6_tr,numeric_tr], y=y_train, validation_d
ata=([X1_cv,X2_cv,X3_cv,X4_cv, X5_cv, X6_cv,numeric_cv],y_cv),epochs=15,batch_size=300,verbose=2)
```

```
Train on 64000 samples, validate on 16000 samples
Epoch 1/15
 - 88s - loss: 0.3861 - auroc: 0.7274 - val_loss: 0.3936 - val_auroc: 0.7068
Epoch 2/15
 - 87s - loss: 0.3849 - auroc: 0.7293 - val_loss: 0.3943 - val_auroc: 0.7080
Epoch 3/15
 - 87s - loss: 0.3842 - auroc: 0.7312 - val_loss: 0.4000 - val_auroc: 0.7085
Epoch 4/15
 - 88s - loss: 0.3826 - auroc: 0.7349 - val_loss: 0.3901 - val_auroc: 0.7096
Epoch 5/15
 - 87s - loss: 0.3820 - auroc: 0.7371 - val_loss: 0.3935 - val_auroc: 0.7106
Epoch 6/15
 - 87s - loss: 0.3795 - auroc: 0.7418 - val_loss: 0.3895 - val_auroc: 0.7110
Epoch 7/15
 - 88s - loss: 0.3791 - auroc: 0.7444 - val_loss: 0.3893 - val_auroc: 0.7106
Epoch 8/15
 - 87s - loss: 0.3778 - auroc: 0.7474 - val_loss: 0.3908 - val_auroc: 0.7110
Epoch 9/15
 - 87s - loss: 0.3761 - auroc: 0.7490 - val_loss: 0.3902 - val_auroc: 0.7121
Epoch 10/15
 - 86s - loss: 0.3743 - auroc: 0.7549 - val_loss: 0.3898 - val_auroc: 0.7119
Epoch 11/15
 - 87s - loss: 0.3729 - auroc: 0.7576 - val_loss: 0.3910 - val_auroc: 0.7098
Epoch 12/15
 - 87s - loss: 0.3706 - auroc: 0.7621 - val_loss: 0.3947 - val_auroc: 0.7110
Epoch 13/15
 - 87s - loss: 0.3710 - auroc: 0.7638 - val_loss: 0.3911 - val_auroc: 0.7091
Epoch 14/15
 - 87s - loss: 0.3682 - auroc: 0.7679 - val_loss: 0.3907 - val_auroc: 0.7090
Epoch 15/15
Epoch 15/15
 - 87s - loss: 0.3657 - auroc: 0.7726 - val_loss: 0.3914 - val_auroc: 0.7092
```

In [0]:

```
score = model_3.evaluate(x=[X1_test,X2_test,X3_test,X4_test, X5_test, X6_test, numeric_test], y=y_t
est, verbose=2,batch_size=300)
```

In [0]:

```
print("Test Loss:", score[0])
print("Test AUC:", score[1])
```

```
Test Loss: 0.39427627846598623
Test AUC: 0.7000997626670504
```

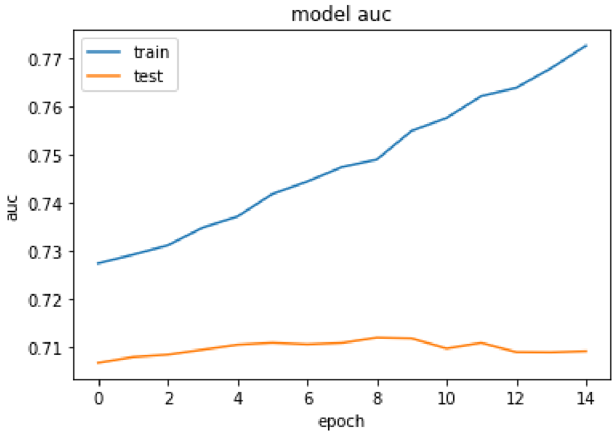
In [0]:

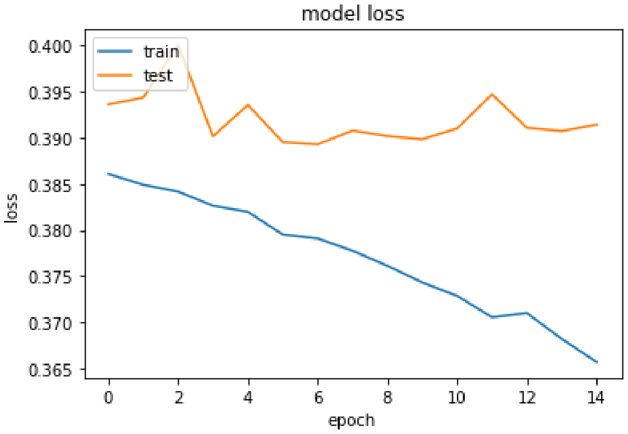
```
plt.plot(history.history['auroc'])
plt.plot(history.history['val_auroc'])

plt.title('model auc')
plt.ylabel('auc')
plt.xlabel('epoch')
plt.legend(['train','test'], loc='upper left')
plt.show()

plt.plot(history.history['loss'])
plt.plot(history.history['val_loss'])

plt.title('model loss')
plt.ylabel('loss')
plt.xlabel('epoch')
plt.legend(['train','test'], loc='upper left')
plt.show()
```





In [0]:

```
# serialize weights to HDF5
model_3.save_weights("model_3.h5")
print("Saved model to disk")
```

```
Saved model to disk
```