```
#importing libraries

import os
import regex as re
import numpy as np
import pandas as pd
from tqdm import tqdm
import numpy as np
from sklearn.model_selection import train_test_split
import matplotlib.pyplot as plt
import seaborn as sns

import nltk
from nltk.corpus import stopwords
from nltk.tokenize import word_tokenize
nltk.download('punkt')
nltk.download('stopwords')

from sklearn.metrics import confusion_matrix
from sklearn import metrics
from sklearn.metrics import roc_curve, auc,roc_auc_score

import tensorflow as tf
from tensorflow.keras.preprocessing.text import Tokenizer
from tensorflow.keras.preprocessing.sequence import pad_sequences
```

```
[nltk_data] Downloading package punkt to /root/nltk_data...
[nltk_data]   Package punkt is already up-to-date!
[nltk_data] Downloading package stopwords to /root/nltk_data...
[nltk_data]   Package stopwords is already up-to-date!
```

Loading Data

```
file = r'/content/drive/MyDrive/Colab Notebooks/Datasets/Donor_Choose_LSTM/preprocessed_data.csv'

# df = pd.read_csv(file,nrows=1000)
df = pd.read_csv(file)
df.head(3)
```

```
  school_state teacher_prefix project_grade_category  \
0           ca            mrs          grades_prek_2
1           ut             ms             grades_3_5
2           ca            mrs          grades_prek_2

   teacher_number_of_previously_posted_projects
project_is_approved  \
0                                            53                    1

1                                             4                    1
```

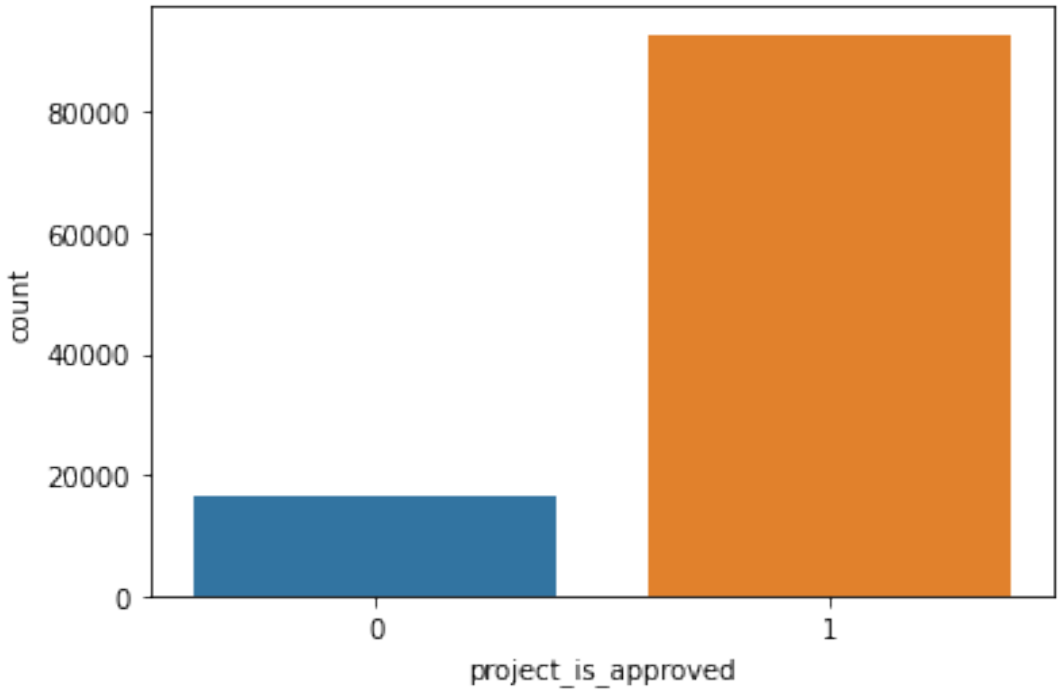
```
2                                          10                    1


    clean_categories                clean_subcategories  \
0       math_science  appliedsciences health_lifescience
1       specialneeds                     specialneeds
2  literacy_language                         literacy

                                               essay   price
0  i fortunate enough use fairy tale stem kits cl...  725.05
1  imagine 8 9 years old you third grade classroo...  213.03
2  having class 24 students comes diverse learner...  329.00
```

```
#checking for null values
df.isnull().sum()
```

```
school_state                                    0
teacher_prefix                                  0
project_grade_category                          0
teacher_number_of_previously_posted_projects    0
project_is_approved                             0
clean_categories                                0
clean_subcategories                             0
essay                                           0
price                                           0
dtype: int64
```

```
print(df.project_is_approved.value_counts())
sns.countplot(x=df.project_is_approved)
plt.show()
```

```
1    92706
0    16542
Name: project_is_approved, dtype: int64
```



Preprocessing some data

```
#removing stopwords

stop_words = set(stopwords.words('english'))
stop_words.add('due')

def min_len_stopword_remover(sent):

  # tokenizing each essay
  word_tokens = word_tokenize(sent)

  filtered_sentence = [w for w in word_tokens if not w.lower() in
stop_words]
  final_sent = ' '.join(filtered_sentence)

  return final_sent


#applying on dataframe
df['essay'] = df['essay'].apply(lambda x :
min_len_stopword_remover(x))

#Removing words having len()<=2 and numbers which are followed by
words like '7995d' or 'dasdas8'

def number_char_remover(sent):

  pattern = '[A-Za-z]+\d+|\d+[A-Za-z]+'
```

```
    pattern = re.compile(pattern)
    sent = re.sub(pattern,' ',sent)

    #removing numbers
    pattern = '\d+'
    sent = re.sub(re.compile(pattern),' ',sent)
    sent = ' '.join([i for i in sent.split() if len(i.strip())>=2])

    return sent

df['essay'] = df['essay'].apply(lambda x : number_char_remover(x))

#removing _ from categories
df['project_grade_category'] =
df['project_grade_category'].apply(lambda x:''.join(x.split('_')))
```

Splitting Data

```
train_df, test_df = train_test_split(df,
test_size=0.3,random_state=2,stratify=df['project_is_approved'],shuffl
e=True)

print('Shape of train and test df = ',train_df.shape, test_df.shape)
```

```
Shape of train and test df =  (76473, 9) (32775, 9)
```

Class Labels

```
#OHE Encoding class labels

from tensorflow.keras.utils import to_categorical

y_train = to_categorical(train_df['project_is_approved'])
y_test = to_categorical(test_df['project_is_approved'])

#printing shape
print('Train = ',y_train.shape)
print('Test = ',y_test.shape)
```

```
Train =  (76473, 2)
Test =  (32775, 2)
```

Text Vectorizing Data

1. School State

```
#checking total states
print('Total School states in Train
=',len(train_df['school_state'].unique()))
```

```
print('Total School states in Test =
',len(test_df['school_state'].unique()))
```

```
Total School states in Train = 51
Total School states in Test =  51
```

```
#Creating school tokenizer
school_token = Tokenizer()

#fiitng on train data
school_token.fit_on_texts(train_df['school_state'])
print(school_token.word_index)

#creating tokenized sequence
train_school_state =
school_token.texts_to_sequences(train_df['school_state'])
test_school_state =
school_token.texts_to_sequences(test_df['school_state'])
```

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

```
#Train data
x_train_school_state = pad_sequences(train_school_state,maxlen=1)
print(x_train_school_state.shape)

#Test data
x_test_school_state = pad_sequences(test_school_state,maxlen=1)
print(x_test_school_state.shape)
```

```
(76473, 1)
(32775, 1)
```

```
x_train_school_state[5]
```

```
array([18], dtype=int32)
```

```
#checking for Nan values
np.isnan(x_test_school_state).sum()
```

```
0
```

2. Project Grade

```
train_df['project_grade_category'].unique()
```

```
array(['grades35', 'gradesprek2', 'grades68', 'grades912'],
dtype=object)
```

```
#checking total states
print('Total Project_Grades in Train
=',len(train_df['project_grade_category'].unique()))
print('Total Project_Grades in Test =
',len(test_df['project_grade_category'].unique()))
```

```
Total Project_Grades in Train = 4
Total Project_Grades in Test =  4
```

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

#fiitng on train data
project_grade_token.fit_on_texts(train_df['project_grade_category'])
print(project_grade_token.word_index)

#creating tokenized sequence
train_project_grade =
project_grade_token.texts_to_sequences(train_df['project_grade_categor
y'])
test_project_grade =
project_grade_token.texts_to_sequences(test_df['project_grade_category
'])
```

```
{'gradesprek2': 1, 'grades35': 2, 'grades68': 3, 'grades912': 4}
```

```
#Train data
x_train_project_grade = pad_sequences(train_project_grade,maxlen=1)
print(x_train_project_grade.shape)

#Test data
x_test_project_grade = pad_sequences(test_project_grade,maxlen=1)
print(x_test_project_grade.shape)
```

```
(76473, 1)
(32775, 1)
```

```
#checking for Nan values
np.isnan(x_test_project_grade).sum()
```

```
0
```

3. clean_categories

```
#checking total states
print('Total clean_categories in Train
=',len(train_df['clean_categories'].unique()))
print('Total clean_categories in Test =
',len(test_df['clean_categories'].unique()))
```

```
Total clean_categories in Train = 51
Total clean_categories in Test =  50
```

```
train_df['clean_categories'].value_counts()
```

```
literacy_language                          16643
math_science                               12022
literacy_language math_science             10188
health_sports                               7050
music_arts                                  3630
specialneeds                                2960
literacy_language specialneeds              2792
appliedlearning                             2668
math_science literacy_language              1590
appliedlearning literacy_language           1511
math_science specialneeds                   1286
history_civics                              1281
literacy_language music_arts                1211
math_science music_arts                     1142
appliedlearning specialneeds                1022
history_civics literacy_language            1010
health_sports specialneeds                   977
warmth care_hunger                           935
math_science appliedlearning                 826
appliedlearning math_science                 739
literacy_language history_civics             577
health_sports literacy_language              564
appliedlearning music_arts                   538
math_science history_civics                  436
literacy_language appliedlearning            430
appliedlearning health_sports                428
math_science health_sports                   301
history_civics math_science                  223
specialneeds music_arts                      213
history_civics music_arts                    209
health_sports math_science                   181
history_civics specialneeds                  179
health_sports appliedlearning                142
health_sports music_arts                     118
appliedlearning history_civics               114
music_arts specialneeds                       98
literacy_language health_sports               50
health_sports history_civics                  34
specialneeds health_sports                    30
history_civics appliedlearning                27
health_sports warmth care_hunger              17
music_arts health_sports                      15
specialneeds warmth care_hunger               14
music_arts history_civics                     10
appliedlearning warmth care_hunger             9
math_science warmth care_hunger                9
```

```
history_civics health_sports                9
music_arts appliedlearning                  7
literacy_language warmth care_hunger        6
history_civics warmth care_hunger           1
music_arts warmth care_hunger               1
Name: clean_categories, dtype: int64
```

```
train_df.tail()
```

```
      school_state teacher_prefix project_grade_category  \
55558           la             ms                grades35
35962           nc            mrs             gradesprek2
86373           sc             mr               grades912
46563           il             ms             gradesprek2
30742           tn            mrs                grades68

       teacher_number_of_previously_posted_projects
project_is_approved  \
55558                                             0
1
35962                                            14
1
86373                                             1
1
46563                                             3
1
30742                                             0
0

                    clean_categories          clean_subcategories  \
55558                    specialneeds                  specialneeds
35962  health_sports appliedlearning        health_wellness other
86373                    math_science              appliedsciences
46563                   health_sports                  gym_fitness
30742                    math_science  appliedsciences mathematics

                                                   essay    price
55558  classroom consists children significant disabi...  1511.09
35962  person person matter small dr seuss classroom ...   120.07
86373  students starting path become certified pc tec...    49.37
46563  school year new different makes teaching speci...   259.99
30742  roughly students still access internet home li...   799.45
```

```
#Creating school tokenizer
clean_categories_token = Tokenizer(filters='!"#$%&()*+,-.;<=>?
@^`{|}~',lower=False,split='  ')

#fiitng on train data
clean_categories_token.fit_on_texts(train_df['clean_categories'])
print(clean_categories_token.word_index)
```

```
#creating tokenized sequence
train_clean_categories =
clean_categories_token.texts_to_sequences(train_df['clean_categories']
)
test_clean_categories =
clean_categories_token.texts_to_sequences(test_df['clean_categories'])
```

```
{'literacy_language': 1, 'math_science': 2, 'literacy_language
math_science': 3, 'health_sports': 4, 'music_arts': 5, 'specialneeds':
6, 'literacy_language specialneeds': 7, 'appliedlearning': 8,
'math_science literacy_language': 9, 'appliedlearning
literacy_language': 10, 'math_science specialneeds': 11,
'history_civics': 12, 'literacy_language music_arts': 13,
'math_science music_arts': 14, 'appliedlearning specialneeds': 15,
'history_civics literacy_language': 16, 'health_sports specialneeds':
17, 'warmth care_hunger': 18, 'math_science appliedlearning': 19,
'appliedlearning math_science': 20, 'literacy_language
history_civics': 21, 'health_sports literacy_language': 22,
'appliedlearning music_arts': 23, 'math_science history_civics': 24,
'literacy_language appliedlearning': 25, 'appliedlearning
health_sports': 26, 'math_science health_sports': 27, 'history_civics
math_science': 28, 'specialneeds music_arts': 29, 'history_civics
music_arts': 30, 'health_sports math_science': 31, 'history_civics
specialneeds': 32, 'health_sports appliedlearning': 33, 'health_sports
music_arts': 34, 'appliedlearning history_civics': 35, 'music_arts
specialneeds': 36, 'literacy_language health_sports': 37,
'health_sports history_civics': 38, 'specialneeds health_sports': 39,
'history_civics appliedlearning': 40, 'health_sports warmth
care_hunger': 41, 'music_arts health_sports': 42, 'specialneeds warmth
care_hunger': 43, 'music_arts history_civics': 44, 'appliedlearning
warmth care_hunger': 45, 'math_science warmth care_hunger': 46,
'history_civics health_sports': 47, 'music_arts appliedlearning': 48,
'literacy_language warmth care_hunger': 49, 'history_civics warmth
care_hunger': 50, 'music_arts warmth care_hunger': 51}
```

```
#Train data
x_train_clean_categories =
pad_sequences(train_clean_categories,maxlen=1)
print(x_train_clean_categories.shape)

#Test data
x_test_clean_categories =
pad_sequences(test_clean_categories,maxlen=1)
print(x_test_clean_categories.shape)
```

```
(76473, 1)
(32775, 1)
```

```
#checking for Nan values
np.isnan(x_test_clean_categories).sum()
```

```
0
```

4. clean_subcategories

```
#checking total states
print('Total clean_subcategories in Train
=',len(train_df['clean_subcategories'].unique()))
print('Total clean_subcategories in Test =
',len(test_df['clean_subcategories'].unique()))
```

```
Total clean_subcategories in Train = 398
Total clean_subcategories in Test =  360
```

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

#fiitng on train data
clean_subcategories_token.fit_on_texts(train_df['clean_subcategories']
)
print(clean_subcategories_token.word_index)

#creating tokenized sequence
train_clean_subcategories =
clean_subcategories_token.texts_to_sequences(train_df['clean_subcatego
ries'])
test_clean_subcategories =
clean_subcategories_token.texts_to_sequences(test_df['clean_subcategor
ies'])
```

```
{'literacy': 1, 'literacy mathematics': 2, 'literature_writing
mathematics': 3, 'literacy literature_writing': 4, 'mathematics': 5,
'literature_writing': 6, 'specialneeds': 7, 'health_wellness': 8,
'appliedsciences mathematics': 9, 'appliedsciences': 10, 'literacy
specialneeds': 11, 'esl literacy': 12, 'visualarts': 13, 'gym_fitness
health_wellness': 14, 'music': 15, 'literature_writing specialneeds':
16, 'warmth care_hunger': 17, 'mathematics specialneeds': 18,
'gym_fitness': 19, 'health_wellness specialneeds': 20,
'environmentalscience': 21, 'teamsports': 22, 'environmentalscience
health_lifescience': 23, 'appliedsciences environmentalscience': 24,
'music performingarts': 25, 'earlydevelopment': 26, 'other': 27,
'environmentalscience mathematics': 28, 'health_lifescience': 29,
'earlydevelopment specialneeds': 30, 'health_wellness
nutritioneducation': 31, 'esl literature_writing': 32,
'earlydevelopment literacy': 33, 'literature_writing visualarts': 34,
'history_geography literature_writing': 35, 'gym_fitness teamsports':
36, 'appliedsciences visualarts': 37, 'appliedsciences
health_lifescience': 38, 'history_geography': 39, 'appliedsciences
literacy': 40, 'health_lifescience mathematics': 41,
'history_geography literacy': 42, 'literacy visualarts': 43,
'mathematics visualarts': 44, 'health_wellness literacy': 45,
'college_careerprep': 46, 'environmentalscience literacy': 47,
```

'performingarts': 48, 'esl': 49, 'appliedsciences literature_writing': 50, 'appliedsciences college_careerprep': 51, 'literacy socialsciences': 52, 'appliedsciences specialneeds': 53, 'health_wellness teamsports': 54, 'foreignlanguages': 55, 'literature_writing socialsciences': 56, 'college_careerprep literature_writing': 57, 'charactereducation literacy': 58, 'charactereducation': 59, 'health_lifescience literacy': 60, 'earlydevelopment health_wellness': 61, 'college_careerprep mathematics': 62, 'specialneeds visualarts': 63, 'history_geography socialsciences': 64, 'environmentalscience literature_writing': 65, 'health_wellness literature_writing': 66, 'earlydevelopment mathematics': 67, 'other specialneeds': 68, 'esl mathematics': 69, 'nutritioneducation': 70, 'civics_government history_geography': 71, 'college_careerprep literacy': 72, 'foreignlanguages literacy': 73, 'health_wellness mathematics': 74, 'environmentalscience visualarts': 75, 'earlydevelopment literature_writing': 76, 'esl specialneeds': 77, 'charactereducation specialneeds': 78, 'socialsciences': 79, 'health_lifescience literature_writing': 80, 'gym_fitness specialneeds': 81, 'health_wellness other': 82, 'charactereducation literature_writing': 83, 'literacy other': 84, 'health_lifescience health_wellness': 85, 'earlydevelopment visualarts': 86, 'charactereducation earlydevelopment': 87, 'history_geography visualarts': 88, 'appliedsciences earlydevelopment': 89, 'financialliteracy': 90, 'environmentalscience specialneeds': 91, 'environmentalscience history_geography': 92, 'literacy parentinvolvement': 93, 'financialliteracy mathematics': 94, 'earlydevelopment other': 95, 'appliedsciences extracurricular': 96, 'literacy music': 97, 'literature_writing other': 98, 'college_careerprep visualarts': 99, 'civics_government literacy': 100, 'extracurricular': 101, 'health_lifescience specialneeds': 102, 'literacy performingarts': 103, 'college_careerprep specialneeds': 104, 'history_geography specialneeds': 105, 'appliedsciences other': 106, 'music specialneeds': 107, 'charactereducation college_careerprep': 108, 'charactereducation health_wellness': 109, 'literature_writing performingarts': 110, 'performingarts visualarts': 111, 'charactereducation mathematics': 112, 'history_geography mathematics': 113, 'extracurricular visualarts': 114, 'charactereducation other': 115, 'health_lifescience visualarts': 116, 'mathematics parentinvolvement': 117, 'mathematics other': 118, 'civics_government literature_writing': 119, 'appliedsciences history_geography': 120, 'economics financialliteracy': 121, 'civics_government socialsciences': 122, 'college_careerprep other': 123, 'mathematics socialsciences': 124, 'foreignlanguages literature_writing': 125, 'civics_government': 126, 'environmentalscience socialsciences': 127, 'appliedsciences music': 128, 'appliedsciences esl': 129, 'health_lifescience history_geography': 130, 'esl earlydevelopment': 131, 'gym_fitness nutritioneducation': 132, 'appliedsciences socialsciences': 133, 'other visualarts': 134, 'literature_writing parentinvolvement': 135, 'earlydevelopment environmentalscience': 136, 'mathematics music':

137, 'health_lifescience socialsciences': 138, 'charactereducation visualarts': 139, 'socialsciences specialneeds': 140, 'communityservice': 141, 'charactereducation communityservice': 142, 'appliedsciences charactereducation': 143, 'socialsciences visualarts': 144, 'charactereducation extracurricular': 145, 'appliedsciences parentinvolvement': 146, 'health_lifescience nutritioneducation': 147, 'esl foreignlanguages': 148, 'environmentalscience health_wellness': 149, 'extracurricular mathematics': 150, 'appliedsciences health_wellness': 151, 'literature_writing music': 152, 'financialliteracy specialneeds': 153, 'health_wellness visualarts': 154, 'college_careerprep health_lifescience': 155, 'music visualarts': 156, 'communityservice environmentalscience': 157, 'health_wellness music': 158, 'parentinvolvement': 159, 'specialneeds teamsports': 160, 'college_careerprep extracurricular': 161, 'earlydevelopment parentinvolvement': 162, 'earlydevelopment health_lifescience': 163, 'environmentalscience nutritioneducation': 164, 'foreignlanguages mathematics': 165, 'esl environmentalscience': 166, 'esl visualarts': 167, 'esl health_lifescience': 168, 'extracurricular literacy': 169, 'earlydevelopment gym_fitness': 170, 'esl history_geography': 171, 'college_careerprep parentinvolvement': 172, 'parentinvolvement visualarts': 173, 'communityservice visualarts': 174, 'extracurricular other': 175, 'gym_fitness mathematics': 176, 'economics': 177, 'nutritioneducation specialneeds': 178, 'college_careerprep environmentalscience': 179, 'extracurricular literature_writing': 180, 'appliedsciences performingarts': 181, 'mathematics performingarts': 182, 'charactereducation parentinvolvement': 183, 'communityservice literature_writing': 184, 'financialliteracy literacy': 185, 'gym_fitness literacy': 186, 'earlydevelopment performingarts': 187, 'parentinvolvement specialneeds': 188, 'charactereducation environmentalscience': 189, 'college_careerprep socialsciences': 190, 'gym_fitness music': 191, 'extracurricular teamsports': 192, 'health_wellness history_geography': 193, 'charactereducation music': 194, 'history_geography music': 195, 'communityservice specialneeds': 196, 'college_careerprep history_geography': 197, 'college_careerprep communityservice': 198, 'civics_government specialneeds': 199, 'esl health_wellness': 200, 'extracurricular music': 201, 'charactereducation health_lifescience': 202, 'college_careerprep earlydevelopment': 203, 'economics history_geography': 204, 'performingarts specialneeds': 205, 'college_careerprep performingarts': 206, 'charactereducation teamsports': 207, 'college_careerprep health_wellness': 208, 'earlydevelopment music': 209, 'extracurricular performingarts': 210, 'foreignlanguages history_geography': 211, 'civics_government economics': 212, 'economics mathematics': 213, 'extracurricular specialneeds': 214, 'esl socialsciences': 215, 'foreignlanguages specialneeds': 216, 'specialneeds warmth care_hunger': 217, 'history_geography performingarts': 218, 'civics_government financialliteracy': 219, 'college_careerprep foreignlanguages': 220, 'charactereducation socialsciences': 221, 'health_wellness warmth care_hunger': 222,

'charactereducation esl': 223, 'other parentinvolvement': 224, 'environmentalscience performingarts': 225, 'health_wellness performingarts': 226, 'esl music': 227, 'civics_government environmentalscience': 228, 'financialliteracy literature_writing': 229, 'gym_fitness literature_writing': 230, 'communityservice extracurricular': 231, 'esl performingarts': 232, 'health_wellness socialsciences': 233, 'financialliteracy history_geography': 234, 'communityservice literacy': 235, 'civics_government health_lifescience': 236, 'appliedsciences gym_fitness': 237, 'earlydevelopment socialsciences': 238, 'communityservice mathematics': 239, 'esl other': 240, 'appliedsciences civics_government': 241, 'earlydevelopment extracurricular': 242, 'foreignlanguages visualarts': 243, 'communityservice health_lifescience': 244, 'gym_fitness other': 245, 'communityservice health_wellness': 246, 'environmentalscience other': 247, 'civics_government visualarts': 248, 'communityservice other': 249, 'charactereducation performingarts': 250, 'mathematics nutritioneducation': 251, 'college_careerprep nutritioneducation': 252, 'college_careerprep music': 253, 'gym_fitness performingarts': 254, 'economics literacy': 255, 'performingarts teamsports': 256, 'college_careerprep financialliteracy': 257, 'nutritioneducation other': 258, 'appliedsciences teamsports': 259, 'communityservice parentinvolvement': 260, 'civics_government college_careerprep': 261, 'foreignlanguages health_wellness': 262, 'earlydevelopment nutritioneducation': 263, 'college_careerprep esl': 264, 'health_lifescience teamsports': 265, 'charactereducation history_geography': 266, 'mathematics teamsports': 267, 'economics visualarts': 268, 'extracurricular parentinvolvement': 269, 'appliedsciences communityservice': 270, 'literacy teamsports': 271, 'music teamsports': 272, 'health_lifescience other': 273, 'gym_fitness visualarts': 274, 'health_lifescience parentinvolvement': 275, 'environmentalscience extracurricular': 276, 'economics socialsciences': 277, 'foreignlanguages socialsciences': 278, 'nutritioneducation teamsports': 279, 'literature_writing teamsports': 280, 'history_geography other': 281, 'health_lifescience music': 282, 'civics_government mathematics': 283, 'foreignlanguages music': 284, 'esl parentinvolvement': 285, 'charactereducation gym_fitness': 286, 'gym_fitness health_lifescience': 287, 'environmentalscience parentinvolvement': 288, 'extracurricular health_wellness': 289, 'appliedsciences financialliteracy': 290, 'parentinvolvement socialsciences': 291, 'communityservice socialsciences': 292, 'charactereducation foreignlanguages': 293, 'appliedsciences foreignlanguages': 294, 'teamsports visualarts': 295, 'charactereducation financialliteracy': 296, 'civics_government performingarts': 297, 'literacy nutritioneducation': 298, 'music socialsciences': 299, 'civics_government communityservice': 300, 'performingarts socialsciences': 301, 'health_lifescience performingarts': 302, 'charactereducation warmth care_hunger': 303, 'esl financialliteracy': 304, 'appliedsciences nutritioneducation': 305, 'gym_fitness history_geography': 306, 'financialliteracy

visualarts': 307, 'foreignlanguages other': 308, 'extracurricular health_lifescience': 309, 'music other': 310, 'earlydevelopment financialliteracy': 311, 'economics environmentalscience': 312, 'communityservice nutritioneducation': 313, 'nutritioneducation visualarts': 314, 'communityservice history_geography': 315, 'communityservice earlydevelopment': 316, 'environmentalscience foreignlanguages': 317, 'nutritioneducation warmth care_hunger': 318, 'esl extracurricular': 319, 'health_wellness parentinvolvement': 320, 'extracurricular socialsciences': 321, 'music parentinvolvement': 322, 'health_lifescience warmth care_hunger': 323, 'financialliteracy health_lifescience': 324, 'esl nutritioneducation': 325, 'appliedsciences economics': 326, 'communityservice performingarts': 327, 'literacy warmth care_hunger': 328, 'civics_government esl': 329, 'economics specialneeds': 330, 'communityservice esl': 331, 'environmentalscience gym_fitness': 332, 'esl gym_fitness': 333, 'literature_writing warmth care_hunger': 334, 'college_careerprep economics': 335, 'environmentalscience financialliteracy': 336, 'foreignlanguages performingarts': 337, 'environmentalscience music': 338, 'extracurricular foreignlanguages': 339, 'nutritioneducation socialsciences': 340, 'earlydevelopment warmth care_hunger': 341, 'environmentalscience warmth care_hunger': 342, 'civics_government extracurricular': 343, 'financialliteracy other': 344, 'earlydevelopment foreignlanguages': 345, 'mathematics warmth care_hunger': 346, 'extracurricular nutritioneducation': 347, 'history_geography parentinvolvement': 348, 'history_geography teamsports': 349, 'socialsciences teamsports': 350, 'extracurricular gym_fitness': 351, 'foreignlanguages health_lifescience': 352, 'environmentalscience teamsports': 353, 'other socialsciences': 354, 'charactereducation economics': 355, 'economics health_lifescience': 356, 'foreignlanguages gym_fitness': 357, 'college_careerprep gym_fitness': 358, 'appliedsciences warmth care_hunger': 359, 'other teamsports': 360, 'earlydevelopment history_geography': 361, 'communityservice economics': 362, 'college_careerprep teamsports': 363, 'economics foreignlanguages': 364, 'financialliteracy performingarts': 365, 'gym_fitness parentinvolvement': 366, 'financialliteracy foreignlanguages': 367, 'civics_government nutritioneducation': 368, 'communityservice music': 369, 'other warmth care_hunger': 370, 'earlydevelopment teamsports': 371, 'financialliteracy socialsciences': 372, 'esl economics': 373, 'parentinvolvement performingarts': 374, 'college_careerprep warmth care_hunger': 375, 'parentinvolvement warmth care_hunger': 376, 'economics literature_writing': 377, 'civics_government teamsports': 378, 'charactereducation nutritioneducation': 379, 'other performingarts': 380, 'economics nutritioneducation': 381, 'earlydevelopment economics': 382, 'civics_government parentinvolvement': 383, 'extracurricular financialliteracy': 384, 'history_geography warmth care_hunger': 385, 'communityservice financialliteracy': 386, 'economics music': 387, 'visualarts warmth care_hunger': 388, 'charactereducation civics_government': 389, 'financialliteracy parentinvolvement': 390, 'extracurricular

```
history_geography': 391, 'parentinvolvement teamsports': 392,
'gym_fitness warmth care_hunger': 393, 'gym_fitness socialsciences':
394, 'civics_government health_wellness': 395, 'financialliteracy
health_wellness': 396, 'esl teamsports': 397, 'economics other': 398}

#Train data
x_train_clean_subcategories =
pad_sequences(train_clean_subcategories,maxlen=1)
print(x_train_clean_subcategories.shape)

#Test data
x_test_clean_subcategories =
pad_sequences(test_clean_subcategories,maxlen=1)
print(x_test_clean_subcategories.shape)
```

```
(76473, 1)
(32775, 1)
```

```
#checking for Nan values
np.isnan(x_test_clean_subcategories).sum()
```

```
0
```

5. Teacher prefix

```
#checking total states
print('Total teacher_prefix in Train
=',len(train_df['teacher_prefix'].unique()))
print('Total teacher_prefix in Test =
',len(test_df['teacher_prefix'].unique()))
```

```
Total teacher_prefix in Train = 5
Total teacher_prefix in Test =  5
```

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

#fiitng on train data
teacher_prefix_token.fit_on_texts(train_df['teacher_prefix'])
print(teacher_prefix_token.word_index)

#creating tokenized sequence
train_teacher_prefix =
teacher_prefix_token.texts_to_sequences(train_df['teacher_prefix'])
test_teacher_prefix =
teacher_prefix_token.texts_to_sequences(test_df['teacher_prefix'])
```

```
{'mrs': 1, 'ms': 2, 'mr': 3, 'teacher': 4, 'dr': 5}
```

```
#Train data
x_train_teacher_prefix = pad_sequences(train_teacher_prefix,maxlen=1)
print(x_train_teacher_prefix.shape)
```

```
#Test data
x_test_teacher_prefix = pad_sequences(test_teacher_prefix,maxlen=1)
print(x_test_teacher_prefix.shape)
```

```
(76473, 1)
(32775, 1)
```

```
#checking for Nan values
np.isnan(x_test_teacher_prefix).sum()
```

```
0
```

6. Number of previously submitted projects

```
x_train_remaining_input =
train_df[['teacher_number_of_previously_posted_projects','price']]
x_test_remaining_input =
test_df[['teacher_number_of_previously_posted_projects','price']]
```

7. Essay

```
# https://machinelearningmastery.com/use-word-embedding-layers-deep-
learning-keras/
#Do Tokenizer i.e Assign token to each Number.

#loading tokenizer
essay_token = Tokenizer()

#fitiing on X_train
essay_token.fit_on_texts(train_df['essay'])

#creating word dictionary {word:token_number}
word_index = essay_token.word_index
print('Total words in X_train = ',len(word_index)+1)
```

```
Total words in X_train =  47297
```

```
#loading glove file
glove_file = r'/content/drive/MyDrive/Colab
Notebooks/Datasets/Donor_Choose_LSTM/glove.6B.100d.txt'
f = open(glove_file)

#creating dict {word:100_dim_vector}
glove_embeddings = dict()
for line in f:
  values = line.split()
  word = values[0]
  vector = np.asarray(values[1:], dtype='float32')
  glove_embeddings[word] = vector

f.close()
print(f'Loaded {len(glove_embeddings)} word vectors.')
```

```
Loaded 400000 word vectors.
```

```
#creating embedded matrix which contains GLOVE vector representation
of each word of tokenized words
vocab_size = len(word_index)+1

#each word will be 50 dim GLOVE Vector after loading
embedded_matrix = np.zeros(shape=(vocab_size,100))

#feeding glove vectors in embedding_matrix
for word,index in word_index.items():

  vector = glove_embeddings.get(word)

   #feed only if word is in GLOVE_words else dont feed
  if vector is not None:
    embedded_matrix[index] = vector

print('Shape of Embedded_matrix = ',embedded_matrix.shape)
```

```
Shape of Embedded_matrix =  (47297, 100)
```

```
#checking for Nan values
if np.any(np.isnan(embedded_matrix)):
  print('Nan values are present')
else:
  print('No Nan Values Found')
```

```
No Nan Values Found
```

```
#Padding sequence

# Encoding words of each document in X_train
train_essay = essay_token.texts_to_sequences(train_df['essay'])

#Using tokenizer fitted on X_train
test_essay = essay_token.texts_to_sequences(test_df['essay'])

max_len = len(max(train_essay,key=len))
print('Max length of Sentence is = ',len(max(train_essay,key=len)))
```

```
Max length of Sentence is =  310
```

```
x_train_essay =
pad_sequences(train_essay,maxlen=max_len,padding='post')
x_test_essay = pad_sequences(test_essay,maxlen=max_len,padding='post')

#Checking sample datapoint
i = np.random.randint(low=0,high=len(test_essay))

#if count of non_zeros after padding is equal or not to
```

```
len(train_essay datapoint)

#Train
print(np.count_nonzero(x_train_essay[i]) == len(train_essay[i]))

#Test
print(np.count_nonzero(x_test_essay[i]) == len(test_essay[i]))
```

```
True
True
```

Models

- List of Input features

1. x_train_essay
2. x_train_school_state
3. x_train_project_grade
4. x_train_clean_categories
5. x_train_clean_subcategories
6. x_train_teacher_prefix
7. x_train_remaining_input

```
from tensorflow.keras.layers import
Dense,Embedding,Conv1D,Input,concatenate,MaxPool1D,Dropout,Flatten,Bat
chNormalization
from keras.layers import LSTM
from tensorflow.keras.models import Model,load_model
from tensorflow.keras.utils import plot_model,to_categorical
from tensorflow.keras.initializers import
Constant,he_uniform,he_normal,glorot_normal,glorot_uniform
from tensorflow.keras.regularizers import L2

#Compiling
from tensorflow.keras.losses import categorical_crossentropy
from tensorflow.keras.optimizers import Adam
from tensorflow.keras.metrics import Accuracy
from sklearn.metrics import f1_score,accuracy_score,confusion_matrix
from sklearn.utils import class_weight
from tensorflow.keras.callbacks import
ModelCheckpoint,ReduceLROnPlateau,TerminateOnNaN,EarlyStopping,TensorB
oard

# Checking input shapes of above inputs

inputs_list =
[x_train_essay,x_train_school_state,x_train_project_grade,x_train_clea
n_categories,

x_train_clean_subcategories,x_train_teacher_prefix,x_train_remaining_i
```

```
nput]

inputs_str_list =
['x_train_essay','x_train_school_state','x_train_project_grade','x_tra
in_clean_categories',

'x_train_clean_subcategories','x_train_teacher_prefix','x_train_remain
ing_input']

for i,j in enumerate(inputs_list):
  print(f'{i+1}. {inputs_str_list[i]} shape is  = {j.shape}')

print('=='*80)
```

```
1. x_train_essay shape is  = (76473, 310)
2. x_train_school_state shape is  = (76473, 1)
3. x_train_project_grade shape is  = (76473, 1)
4. x_train_clean_categories shape is  = (76473, 1)
5. x_train_clean_subcategories shape is  = (76473, 1)
6. x_train_teacher_prefix shape is  = (76473, 1)
7. x_train_remaining_input shape is  = (76473, 2)
================================================================================
================================================================================
====================
```

```
#checking for NaN values

for i,j in enumerate(inputs_list):
  print(f'{inputs_str_list[i]} = ',np.where(np.isnan(j)))
```

```
x_train_essay =  (array([], dtype=int64), array([], dtype=int64))
x_train_school_state =  (array([], dtype=int64), array([],
dtype=int64))
x_train_project_grade =  (array([], dtype=int64), array([],
dtype=int64))
x_train_clean_categories =  (array([], dtype=int64), array([],
dtype=int64))
x_train_clean_subcategories =  (array([], dtype=int64), array([],
dtype=int64))
x_train_teacher_prefix =  (array([], dtype=int64), array([],
dtype=int64))
x_train_remaining_input =  (array([], dtype=int64), array([],
dtype=int64))
```

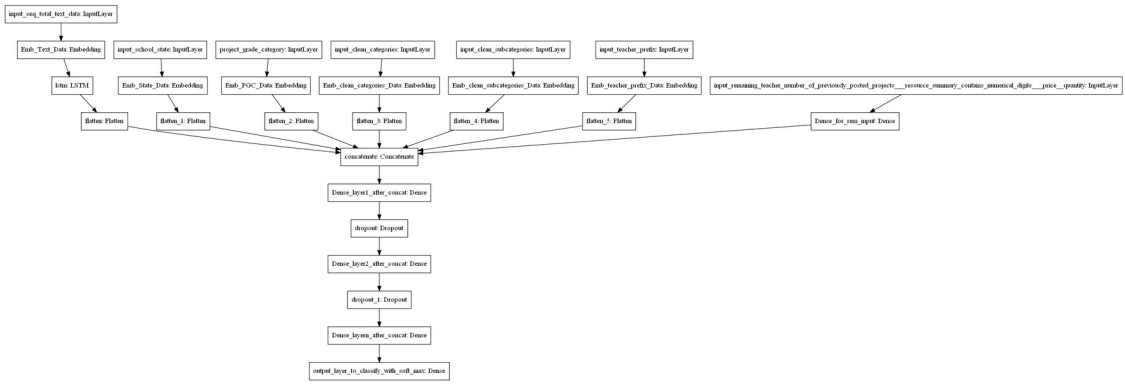
```
#preparing vocab size for embedding layers

token_list =
[essay_token,school_token,project_grade_token,clean_categories_token,
                  clean_subcategories_token,teacher_prefix_token]
token_str_list =
```

```
['essay_token','school_token','project_grade_token','clean_categories_
token',
                  'clean_subcategories_token','teacher_prefix_token']

for i,j in enumerate(token_list):
  print(f'{i+1}. {token_str_list[i]} vocab_size is  =
{len(j.word_index)+1}')

print('=='*80)
```

```
1. essay_token vocab_size is  = 47297
2. school_token vocab_size is  = 52
3. project_grade_token vocab_size is  = 5
4. clean_categories_token vocab_size is  = 52
5. clean_subcategories_token vocab_size is  = 399
6. teacher_prefix_token vocab_size is  = 6
================================================================================
================================================================================
====================
```

```
# https://stackoverflow.com/questions/41032551/how-to-compute-
receiving-operating-characteristic-roc-and-auc-in-keras

import tensorflow

def aucroc(y_true, y_pred):
    # print(y_true, y_pred)
    return tensorflow.py_function(roc_auc_score, (y_true, y_pred),
tf.double)
```

Model_1



Description of Layers

- **Input** (sent_lenght or category)
- **Embedding** (ip_dim=vocab_size, op_dim=100, ip_length=padding_length)

```
#input_1 Essay
ip_1 = Input(shape=(max_len,),name='essay_input')
```

```
l1 =
Embedding(input_dim=vocab_size,output_dim=100,input_length=max_len,
                        embeddings_initializer
=Constant(embedded_matrix),
                        trainable=False,name='Embed_layer')(ip_1)

l1 =
LSTM(128,activation='relu',return_sequences=True,name='LSTM_layer')
(l1)
l1 = Flatten(name='Flatten_essay')(l1)

#input_2 school_state
ip_2 = Input(shape=(1,),name='ip_school_state')
l2 = Embedding(input_dim=52,output_dim=1,name='school_emb')(ip_2)
l2 = Flatten()(l2)

#input project_grade
ip_3 = Input(shape=(1,),name='ip_project_grade')
l3 = Embedding(input_dim=5,output_dim=2,name='project_grade_emb')
(ip_3)
l3 = Flatten()(l3)


#input clean_cat
ip_4 = Input(shape=(1,),name='ip_clean_cat')
l4 = Embedding(input_dim=52,output_dim=2,name='clean_cat_emb')(ip_4)
l4 = Flatten()(l4)

#input clean_subcat
ip_5 = Input(shape=(1,),name='ip_clean_subcat')
l5 = Embedding(input_dim = 399,output_dim=64,name='clean_subcat_emb')
(ip_5)
l5 = Flatten()(l5)

#input teacher_prefix
ip_6 = Input(shape=(1,),name='ip_teacher_prefix')
l6 = Embedding(input_dim = 6,output_dim = 2,name =
'teacher_prefix_emb')(ip_6)
l6 = Flatten()(l6)

# input remaining input
ip_7 = Input(shape=(2,),name='remaining input')
l7 = Dense(16,activation='relu')(ip_7)

#Concatenate all inputs
concat = concatenate([l1,l2,l3,l4,l5,l5,l6,l7])

x = Dense(128, activation='relu',kernel_initializer=he_normal())
```

```
(concat)
x = Dense(64,activation='relu',kernel_initializer=he_normal())(x)
x = Dropout(0.5)(x)
x = Dense(32,activation='relu',kernel_initializer=he_normal())(x)
x = Dropout(0.4)(x)
output = Dense(2, activation = 'softmax')(x)

# model with all the inputs
model_1 = Model([ip_1,ip_2,ip_3,ip_4,ip_5,ip_6,ip_7], output)

#checking model
model_1.summary()
```

```
WARNING:tensorflow:Layer LSTM_layer will not use cuDNN kernels since
it doesn't meet the criteria. It will use a generic GPU kernel as
fallback when running on GPU.

Model: "model_5"
__________________________________________________________________________
_____________________________
 Layer (type)                Output Shape              Param #
Connected to
==========================================================================
============================
 essay_input (InputLayer)    [(None, 310)]             0           []



 Embed_layer (Embedding)     (None, 310, 100)          4729700
['essay_input[0][0]']

 ip_school_state (InputLayer)  [(None, 1)]             0           []



 ip_project_grade (InputLayer)  [(None, 1)]            0           []



 ip_clean_cat (InputLayer)   [(None, 1)]               0           []



 ip_clean_subcat (InputLayer)  [(None, 1)]             0           []



 ip_teacher_prefix (InputLayer)  [(None, 1)]           0           []
```

```
 LSTM_layer (LSTM)              (None, 310, 128)     117248
['Embed_layer[0][0]']


 school_emb (Embedding)         (None, 1, 1)         52
['ip_school_state[0][0]']


 project_grade_emb (Embedding)  (None, 1, 2)         10
['ip_project_grade[0][0]']


 clean_cat_emb (Embedding)      (None, 1, 2)         104
['ip_clean_cat[0][0]']


 clean_subcat_emb (Embedding)   (None, 1, 64)        25536
['ip_clean_subcat[0][0]']


 teacher_prefix_emb (Embedding)  (None, 1, 2)        12
['ip_teacher_prefix[0][0]']


 remaining input (InputLayer)   [(None, 2)]          0           []



 Flatten_essay (Flatten)        (None, 39680)        0
['LSTM_layer[0][0]']


 flatten_22 (Flatten)           (None, 1)            0
['school_emb[0][0]']


 flatten_23 (Flatten)           (None, 2)            0
['project_grade_emb[0][0]']


 flatten_24 (Flatten)           (None, 2)            0
['clean_cat_emb[0][0]']


 flatten_25 (Flatten)           (None, 64)           0
```

```
['clean_subcat_emb[0][0]']


 flatten_26 (Flatten)           (None, 2)           0
['teacher_prefix_emb[0][0]']


 dense_24 (Dense)               (None, 16)          48
['remaining input[0][0]']


 concatenate_5 (Concatenate)    (None, 39831)       0
['Flatten_essay[0][0]',

'flatten_22[0][0]',

'flatten_23[0][0]',

'flatten_24[0][0]',

'flatten_25[0][0]',

'flatten_25[0][0]',

'flatten_26[0][0]',

'dense_24[0][0]']


 dense_25 (Dense)               (None, 128)         5098496
['concatenate_5[0][0]']


 dense_26 (Dense)               (None, 64)          8256
['dense_25[0][0]']


 dropout_12 (Dropout)           (None, 64)          0
['dense_26[0][0]']


 dense_27 (Dense)               (None, 32)          2080
['dropout_12[0][0]']


 dropout_13 (Dropout)           (None, 32)          0
['dense_27[0][0]']
```

```
 dense_28 (Dense)               (None, 2)            66
['dropout_13[0][0]']


==================================================================================
==============================
Total params: 9,981,608
Trainable params: 5,251,908
Non-trainable params: 4,729,700
__________________________________________________________________________________
_______________________________
```

```
#plotting model
plot_model(model_1,show_layer_names=True,show_shapes=True,dpi=70)
```

```
x_train =
[x_train_essay,x_train_school_state,x_train_project_grade,x_train_clea
n_categories,

x_train_clean_subcategories,x_train_teacher_prefix,x_train_remaining_i
nput]

x_test =
[x_test_essay,x_test_school_state,x_test_project_grade,x_test_clean_ca
tegories,

x_test_clean_subcategories,x_test_teacher_prefix,x_test_remaining_inpu
t]

# Clear any logs from previous runs
%load_ext tensorboard
# !rm -rf ./logs/

log_dir = "logs/fit/" + "model_1"
tensorboard_callback =
tensorflow.keras.callbacks.TensorBoard(log_dir=log_dir,
histogram_freq=1)
```

```
The tensorboard extension is already loaded. To reload it, use:
  %reload_ext tensorboard
```

```
model_1.compile(loss='categorical_crossentropy', optimizer=Adam(),
metrics=[aucroc])
```

```
from sklearn.utils import class_weight

class_weights =
class_weight.compute_class_weight(class_weight='balanced',classes=np.u
nique(y_train[:,0]),y=train_df['project_is_approved'].tolist())
print(class_weights)
class_weights = {0:3.30679754,1:0.5890694}
```

```
[3.30222817 0.58921472]
```

```
#saving best model
# https://machinelearningmastery.com/check-point-deep-learning-models-
keras/

filepath="Model_1_{epoch:02d}_{val_aucroc:.2f}.hdf5"
checkpoint = ModelCheckpoint(filepath, monitor='val_aucroc',
verbose=1, save_best_only=True, mode='max')

#Callbacks List
callbacks_list = [checkpoint,tensorboard_callback]

#if you use random_state=5 class weights then at 4th epoch we are
getting 1class error for AUCROC. DOnt use batch normalization is
causes infinity, valueerror

# model_1.fit(x_train,
y_train,epochs=5,class_weight=class_weights,verbose=1,batch_size=2048,
validation_data=[x_test,y_test],
#
validation_batch_size=256,callbacks=callbacks_list,shuffle=True)

model_1.fit(x_train,
y_train,epochs=4,class_weight=class_weights,verbose=1,batch_size=256,v
alidation_data=[x_test,y_test],

validation_batch_size=256,callbacks=callbacks_list,shuffle=True)
```

```
Epoch 1/4
299/299 [==============================] - ETA: 0s - loss: 0.7055 -
aucroc: 0.6155
Epoch 1: val_aucroc improved from -inf to 0.72723, saving model to
Model_1_01_0.73.hdf5
299/299 [==============================] - 181s 601ms/step - loss:
0.7055 - aucroc: 0.6155 - val_loss: 0.6623 - val_aucroc: 0.7272
Epoch 2/4
```

```
299/299 [==============================] - ETA: 0s - loss: 0.6268 -
aucroc: 0.7193
Epoch 2: val_aucroc improved from 0.72723 to 0.73895, saving model to
Model_1_02_0.74.hdf5
299/299 [==============================] - 181s 607ms/step - loss:
0.6268 - aucroc: 0.7193 - val_loss: 0.5852 - val_aucroc: 0.7390
Epoch 3/4
299/299 [==============================] - ETA: 0s - loss: 0.6065 -
aucroc: 0.7433
Epoch 3: val_aucroc improved from 0.73895 to 0.74102, saving model to
Model_1_03_0.74.hdf5
299/299 [==============================] - 178s 594ms/step - loss:
0.6065 - aucroc: 0.7433 - val_loss: 0.6577 - val_aucroc: 0.7410
Epoch 4/4
299/299 [==============================] - ETA: 0s - loss: 0.5865 -
aucroc: 0.7672
Epoch 4: val_aucroc did not improve from 0.74102
299/299 [==============================] - 175s 586ms/step - loss:
0.5865 - aucroc: 0.7672 - val_loss: 0.5930 - val_aucroc: 0.7406

<keras.callbacks.History at 0x7fe95e070790>
```

```
# https://github.com/keras-team/keras/issues/10104
dependencies = {'aucroc': aucroc}

#loading best model
loaded_model_1 =
load_model('Model_1_03_0.74.hdf5',custom_objects=dependencies)

#checking loaded model
# loaded_model_1.summary()
```

```
WARNING:tensorflow:Layer LSTM_layer will not use cuDNN kernels since
it doesn't meet the criteria. It will use a generic GPU kernel as
fallback when running on GPU.
```

```
#predicting train dataset
train_y_pred = loaded_model_1.predict(x_train)
print('Train AUC_ROC_SCORE =',roc_auc_score(y_train,train_y_pred))
```

```
Train AUC_ROC_SCORE = 0.7788059742821933
```

```
#predicting test dataset
test_y_pred = loaded_model_1.predict(x_test)
print('Test AUC_ROC_SCORE =',roc_auc_score(y_test,test_y_pred))
```

```
Test AUC_ROC_SCORE = 0.7394602845466056
```

```
#checking tensorboard
%tensorboard --logdir logs/fit
```

```
Reusing TensorBoard on port 6006 (pid 515), started 0:19:37 ago. (Use
'!kill 515' to kill it.)
```

```
<IPython.core.display.Javascript object>
```

Model_2

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

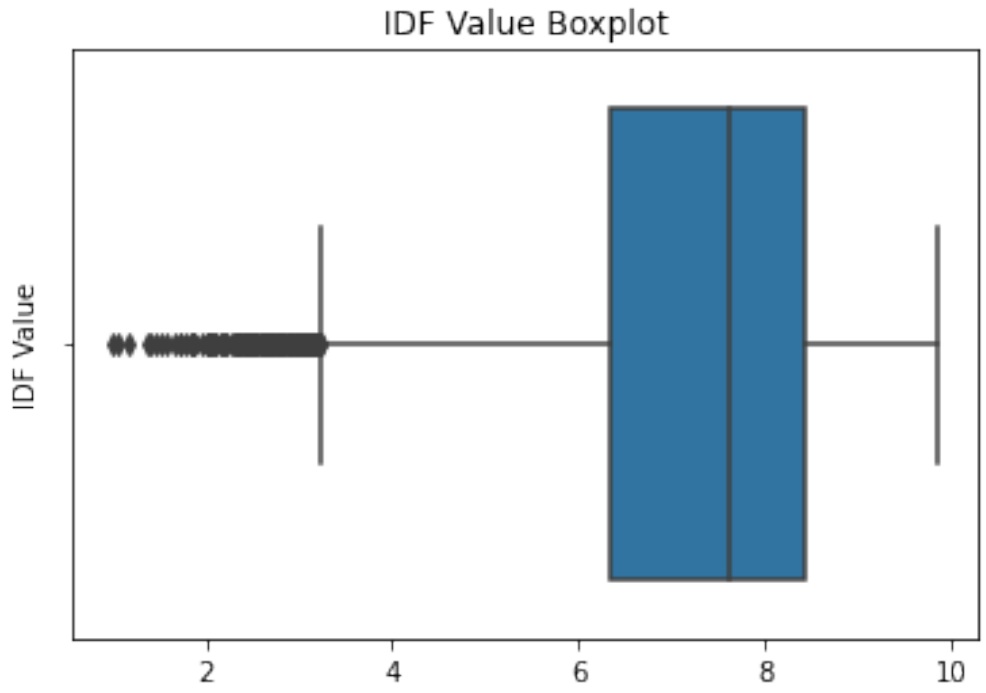
IDF value Analysis

```
from sklearn.feature_extraction.text import TfidfVectorizer
```

```
# train_df['essay']
train_df.head(5)
```

```
      school_state teacher_prefix project_grade_category  \
58689           ny            mrs               grades35
5809            ga            mrs            gradesprek2
15178           mo             ms            gradesprek2
76882           il            mrs               grades35
87515           ny            mrs               grades35

       teacher_number_of_previously_posted_projects
project_is_approved  \
58689                                             0
0
5809                                              0
1
15178                                             0
1
76882                                            20
1
87515                                             0
1

                    clean_categories    clean_subcategories  \
58689  literacy_language specialneeds  literacy specialneeds
5809                     specialneeds           specialneeds
15178                    math_science            mathematics
76882                    math_science        appliedsciences
87515                    math_science            mathematics

                                                   essay   price
58689  students special needs various socioeconomic b...  876.21
5809   teach small school located rural georgia north...  299.99
15178  teacher low income high poverty school distric...  136.59
76882  bright bubbly batch third graders call classro...   47.29
87515  many students class live poverty line range ba...   79.14
```

```
#loading tfidf vectorizer
tfidf = TfidfVectorizer(min_df=5,max_features=10000)

##fitting and transforming on train_df['essay']
X = tfidf.fit_transform(train_df['essay'])

#storing idf values
idf = tfidf.idf_

#creating dictionary of word:idf_value
idf_dict = dict(zip(tfidf.get_feature_names_out(),idf))

#Analysis on idf_ values

sns.boxplot(x = tfidf.idf_)
plt.ylabel("IDF Value")
plt.title('IDF Value Boxplot')
plt.show()
```




```
#checking idf values as per percentile
for i in range (0,101,10):
    p = np.percentile(tfidf.idf_, i)
    print(str(i)+"th Percentile: "+ str(p))
```

```
0th Percentile: 1.0077581333878618
10th Percentile: 5.032928201970643
20th Percentile: 5.9549905218559305
30th Percentile: 6.664976266778706
40th Percentile: 7.182111059737961
```

```
50th Percentile: 7.609977104535292
60th Percentile: 7.968039973748873
70th Percentile: 8.274414179212805
80th Percentile: 8.581144446635282
90th Percentile: 8.843508711102771
100th Percentile: 9.846810819966556
```

```
#setting up lower and higher percentile values

min_thresh = np.percentile(tfidf.idf_,20)
max_thresh = np.percentile(tfidf.idf_,90)

print('Min IDF threshold is :',min_thresh)
print('Max IDF threshold is :',max_thresh)
```

```
Min IDF threshold is : 5.9549905218559305
Max IDF threshold is : 8.843508711102771
```

```
#Filtering words which are between IDF range within idf_dict

for i in list(idf_dict.keys()):

  if (idf_dict[i] <= min_thresh) or (idf_dict[i] >= max_thresh):
    idf_dict.pop(i)

print('No of words:',len(idf_dict))
```

```
No of words: 6920
```

Filtering words which are between IDF range from train_df and exporting pickle files

```
# #Filtering words which are between IDF range from train_df

# def idf_filter(txt):

#   #keeping those words which are in filtered idf_dict
#   txt = [w for w in txt.split() if w.lower() in
list(idf_dict.keys())]

#   filtered_txt = ' '.join(txt)

#   return filtered_txt

# #Filtering train data
# train_df['essay'] = train_df['essay'].apply(idf_filter)

# #Filtering test data
# test_df['essay'] = test_df['essay'].apply(idf_filter)

# train_df['essay'].iloc[569]
```

```
# # saving train_df and test_df to csv format
# train_df.to_pickle('model_2_train_df_new.pkl')
# test_df.to_pickle('model_2_test_df_new.pkl')

# # labels
# train_df.to_csv('model_2_train_df_new.csv')
# test_df.to_csv('model_2_test_df_new.csv')
```

Loading Filtered data

- **I saved csv files after filtering text data because filtering function was taking 53min to filter the words as per IDF threshold.**

```
# #loading new train_df and test_df
# train = r'/content/drive/MyDrive/Colab
Notebooks/Datasets/Donor_Choose_LSTM/model_2_train_df_new.pkl'
# test = r'/content/drive/MyDrive/Colab
Notebooks/Datasets/Donor_Choose_LSTM/model_2_test_df_new.pkl'

# train_df = pd.read_pickle(train)
# test_df = pd.read_pickle(test)
# # train_df.head()

#loading new train_df and test_df csv files
train = r'/content/drive/MyDrive/Colab
Notebooks/Datasets/Donor_Choose_LSTM/model_2_train_df_new.csv'
test = r'/content/drive/MyDrive/Colab
Notebooks/Datasets/Donor_Choose_LSTM/model_2_test_df_new.csv'

train_df = pd.read_csv(train)
test_df = pd.read_csv(test)
# train_df.head()

#checking per class datapoints
train_df['project_is_approved'].value_counts()
```

```
1    64894
0    11579
Name: project_is_approved, dtype: int64
```

```
#checking per class datapoints
test_df['project_is_approved'].value_counts()
```

```
1    27812
0     4963
Name: project_is_approved, dtype: int64
```

Text Vectorizing Data

1. School State

```
#checking total states
print('Total School states in Train
=',len(train_df['school_state'].unique()))
```

```
print('Total School states in Test =
',len(test_df['school_state'].unique()))
```

```
Total School states in Train = 51
Total School states in Test =  51
```

```
#Creating school tokenizer
school_token = Tokenizer()

#fiitng on train data
school_token.fit_on_texts(train_df['school_state'])
print(school_token.word_index)

#creating tokenized sequence
train_school_state =
school_token.texts_to_sequences(train_df['school_state'])
test_school_state =
school_token.texts_to_sequences(test_df['school_state'])
```

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

```
#Train data
x_train_school_state = pad_sequences(train_school_state,maxlen=1)
print(x_train_school_state.shape)

#Test data
x_test_school_state = pad_sequences(test_school_state,maxlen=1)
print(x_test_school_state.shape)
```

```
(76473, 1)
(32775, 1)
```

```
#checking for Nan values
np.isnan(x_test_school_state).sum()
```

```
0
```

2. Project Grade

```
train_df['project_grade_category'].unique()
```

```
array(['grades35', 'gradesprek2', 'grades68', 'grades912'],
dtype=object)
```

```
#checking total states
print('Total Project_Grades in Train
```

```
='',len(train_df['project_grade_category'].unique()))
print('Total Project_Grades in Test =
',len(test_df['project_grade_category'].unique()))
```

```
Total Project_Grades in Train = 4
Total Project_Grades in Test =  4
```

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

#fiitng on train data
project_grade_token.fit_on_texts(train_df['project_grade_category'])
print(project_grade_token.word_index)

#creating tokenized sequence
train_project_grade =
project_grade_token.texts_to_sequences(train_df['project_grade_categor
y'])
test_project_grade =
project_grade_token.texts_to_sequences(test_df['project_grade_category
'])
```

```
{'gradesprek2': 1, 'grades35': 2, 'grades68': 3, 'grades912': 4}
```

```
#Train data
x_train_project_grade = pad_sequences(train_project_grade,maxlen=1)
print(x_train_project_grade.shape)

#Test data
x_test_project_grade = pad_sequences(test_project_grade,maxlen=1)
print(x_test_project_grade.shape)
```

```
(76473, 1)
(32775, 1)
```

```
#checking for Nan values
np.isnan(x_test_project_grade).sum()
```

```
0
```

3. clean_categories

```
#checking total states
print('Total clean_categories in Train
=',len(train_df['clean_categories'].unique()))
print('Total clean_categories in Test =
',len(test_df['clean_categories'].unique()))
```

```
Total clean_categories in Train = 51
Total clean_categories in Test =  50
```

```
train_df['clean_categories'].value_counts()
```

literacy_language	16643
math_science	12022
literacy_language math_science	10188
health_sports	7050
music_arts	3630
specialneeds	2960
literacy_language specialneeds	2792
appliedlearning	2668
math_science literacy_language	1590
appliedlearning literacy_language	1511
math_science specialneeds	1286
history_civics	1281
literacy_language music_arts	1211
math_science music_arts	1142
appliedlearning specialneeds	1022
history_civics literacy_language	1010
health_sports specialneeds	977
warmth care_hunger	935
math_science appliedlearning	826
appliedlearning math_science	739
literacy_language history_civics	577
health_sports literacy_language	564
appliedlearning music_arts	538
math_science history_civics	436
literacy_language appliedlearning	430
appliedlearning health_sports	428
math_science health_sports	301
history_civics math_science	223
specialneeds music_arts	213
history_civics music_arts	209
health_sports math_science	181
history_civics specialneeds	179
health_sports appliedlearning	142
health_sports music_arts	118
appliedlearning history_civics	114
music_arts specialneeds	98
literacy_language health_sports	50
health_sports history_civics	34
specialneeds health_sports	30
history_civics appliedlearning	27
health_sports warmth care_hunger	17
music_arts health_sports	15
specialneeds warmth care_hunger	14
music_arts history_civics	10
appliedlearning warmth care_hunger	9
math_science warmth care_hunger	9
history_civics health_sports	9
music_arts appliedlearning	7
literacy_language warmth care_hunger	6
history_civics warmth care_hunger	1

```
music_arts warmth care_hunger               1
Name: clean_categories, dtype: int64
```

```
#Creating school tokenizer
clean_categories_token = Tokenizer(filters='!"#$%&()*+,-.;<=>?
@^`{|}~',lower=False,split='  ')

#fiitng on train data
clean_categories_token.fit_on_texts(train_df['clean_categories'])
print(clean_categories_token.word_index)

#creating tokenized sequence
train_clean_categories =
clean_categories_token.texts_to_sequences(train_df['clean_categories']
)
test_clean_categories =
clean_categories_token.texts_to_sequences(test_df['clean_categories'])
```

```
{'literacy_language': 1, 'math_science': 2, 'literacy_language
math_science': 3, 'health_sports': 4, 'music_arts': 5, 'specialneeds':
6, 'literacy_language specialneeds': 7, 'appliedlearning': 8,
'math_science literacy_language': 9, 'appliedlearning
literacy_language': 10, 'math_science specialneeds': 11,
'history_civics': 12, 'literacy_language music_arts': 13,
'math_science music_arts': 14, 'appliedlearning specialneeds': 15,
'history_civics literacy_language': 16, 'health_sports specialneeds':
17, 'warmth care_hunger': 18, 'math_science appliedlearning': 19,
'appliedlearning math_science': 20, 'literacy_language
history_civics': 21, 'health_sports literacy_language': 22,
'appliedlearning music_arts': 23, 'math_science history_civics': 24,
'literacy_language appliedlearning': 25, 'appliedlearning
health_sports': 26, 'math_science health_sports': 27, 'history_civics
math_science': 28, 'specialneeds music_arts': 29, 'history_civics
music_arts': 30, 'health_sports math_science': 31, 'history_civics
specialneeds': 32, 'health_sports appliedlearning': 33, 'health_sports
music_arts': 34, 'appliedlearning history_civics': 35, 'music_arts
specialneeds': 36, 'literacy_language health_sports': 37,
'health_sports history_civics': 38, 'specialneeds health_sports': 39,
'history_civics appliedlearning': 40, 'health_sports warmth
care_hunger': 41, 'music_arts health_sports': 42, 'specialneeds warmth
care_hunger': 43, 'music_arts history_civics': 44, 'appliedlearning
warmth care_hunger': 45, 'math_science warmth care_hunger': 46,
'history_civics health_sports': 47, 'music_arts appliedlearning': 48,
'literacy_language warmth care_hunger': 49, 'history_civics warmth
care_hunger': 50, 'music_arts warmth care_hunger': 51}
```

```
#Train data
x_train_clean_categories =
pad_sequences(train_clean_categories,maxlen=1)
print(x_train_clean_categories.shape)
```

```
#Test data
x_test_clean_categories =
pad_sequences(test_clean_categories,maxlen=1)
print(x_test_clean_categories.shape)
```

```
(76473, 1)
(32775, 1)
```

```
#checking for Nan values
np.isnan(x_test_clean_categories).sum()
```

```
0
```

4. clean_subcategories

```
#checking total states
print('Total clean_subcategories in Train
=',len(train_df['clean_subcategories'].unique()))
print('Total clean_subcategories in Test =
',len(test_df['clean_subcategories'].unique()))
```

```
Total clean_subcategories in Train = 398
Total clean_subcategories in Test =  360
```

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

#fiitng on train data
clean_subcategories_token.fit_on_texts(train_df['clean_subcategories']
)
print(clean_subcategories_token.word_index)

#creating tokenized sequence
train_clean_subcategories =
clean_subcategories_token.texts_to_sequences(train_df['clean_subcatego
ries'])
test_clean_subcategories =
clean_subcategories_token.texts_to_sequences(test_df['clean_subcategor
ies'])
```

```
{'literacy': 1, 'literacy mathematics': 2, 'literature_writing
mathematics': 3, 'literacy literature_writing': 4, 'mathematics': 5,
'literature_writing': 6, 'specialneeds': 7, 'health_wellness': 8,
'appliedsciences mathematics': 9, 'appliedsciences': 10, 'literacy
specialneeds': 11, 'esl literacy': 12, 'visualarts': 13, 'gym_fitness
health_wellness': 14, 'music': 15, 'literature_writing specialneeds':
16, 'warmth care_hunger': 17, 'mathematics specialneeds': 18,
'gym_fitness': 19, 'health_wellness specialneeds': 20,
'environmentalscience': 21, 'teamsports': 22, 'environmentalscience
health_lifescience': 23, 'appliedsciences environmentalscience': 24,
```

'music performingarts': 25, 'earlydevelopment': 26, 'other': 27,
'environmentalscience mathematics': 28, 'health_lifescience': 29,
'earlydevelopment specialneeds': 30, 'health_wellness
nutritioneducation': 31, 'esl literature_writing': 32,
'earlydevelopment literacy': 33, 'literature_writing visualarts': 34,
'history_geography literature_writing': 35, 'gym_fitness teamsports':
36, 'appliedsciences visualarts': 37, 'appliedsciences
health_lifescience': 38, 'history_geography': 39, 'appliedsciences
literacy': 40, 'health_lifescience mathematics': 41,
'history_geography literacy': 42, 'literacy visualarts': 43,
'mathematics visualarts': 44, 'health_wellness literacy': 45,
'college_careerprep': 46, 'environmentalscience literacy': 47,
'performingarts': 48, 'esl': 49, 'appliedsciences literature_writing':
50, 'appliedsciences college_careerprep': 51, 'literacy
socialsciences': 52, 'appliedsciences specialneeds': 53,
'health_wellness teamsports': 54, 'foreignlanguages': 55,
'literature_writing socialsciences': 56, 'college_careerprep
literature_writing': 57, 'charactereducation literacy': 58,
'charactereducation': 59, 'health_lifescience literacy': 60,
'earlydevelopment health_wellness': 61, 'college_careerprep
mathematics': 62, 'specialneeds visualarts': 63, 'history_geography
socialsciences': 64, 'environmentalscience literature_writing': 65,
'health_wellness literature_writing': 66, 'earlydevelopment
mathematics': 67, 'other specialneeds': 68, 'esl mathematics': 69,
'nutritioneducation': 70, 'civics_government history_geography': 71,
'college_careerprep literacy': 72, 'foreignlanguages literacy': 73,
'health_wellness mathematics': 74, 'environmentalscience visualarts':
75, 'earlydevelopment literature_writing': 76, 'esl specialneeds': 77,
'charactereducation specialneeds': 78, 'socialsciences': 79,
'health_lifescience literature_writing': 80, 'gym_fitness
specialneeds': 81, 'health_wellness other': 82, 'charactereducation
literature_writing': 83, 'literacy other': 84, 'health_lifescience
health_wellness': 85, 'earlydevelopment visualarts': 86,
'charactereducation earlydevelopment': 87, 'history_geography
visualarts': 88, 'appliedsciences earlydevelopment': 89,
'financialliteracy': 90, 'environmentalscience specialneeds': 91,
'environmentalscience history_geography': 92, 'literacy
parentinvolvement': 93, 'financialliteracy mathematics': 94,
'earlydevelopment other': 95, 'appliedsciences extracurricular': 96,
'literacy music': 97, 'literature_writing other': 98,
'college_careerprep visualarts': 99, 'civics_government literacy':
100, 'extracurricular': 101, 'health_lifescience specialneeds': 102,
'literacy performingarts': 103, 'college_careerprep specialneeds':
104, 'history_geography specialneeds': 105, 'appliedsciences other':
106, 'music specialneeds': 107, 'charactereducation
college_careerprep': 108, 'charactereducation health_wellness': 109,
'literature_writing performingarts': 110, 'performingarts visualarts':
111, 'charactereducation mathematics': 112, 'history_geography
mathematics': 113, 'extracurricular visualarts': 114,
'charactereducation other': 115, 'health_lifescience visualarts': 116,

'mathematics parentinvolvement': 117, 'mathematics other': 118,
'civics_government literature_writing': 119, 'appliedsciences
history_geography': 120, 'economics financialliteracy': 121,
'civics_government socialsciences': 122, 'college_careerprep other':
123, 'mathematics socialsciences': 124, 'foreignlanguages
literature_writing': 125, 'civics_government': 126,
'environmentalscience socialsciences': 127, 'appliedsciences music':
128, 'appliedsciences esl': 129, 'health_lifescience
history_geography': 130, 'esl earlydevelopment': 131, 'gym_fitness
nutritioneducation': 132, 'appliedsciences socialsciences': 133,
'other visualarts': 134, 'literature_writing parentinvolvement': 135,
'earlydevelopment environmentalscience': 136, 'mathematics music':
137, 'health_lifescience socialsciences': 138, 'charactereducation
visualarts': 139, 'socialsciences specialneeds': 140,
'communityservice': 141, 'charactereducation communityservice': 142,
'appliedsciences charactereducation': 143, 'socialsciences
visualarts': 144, 'charactereducation extracurricular': 145,
'appliedsciences parentinvolvement': 146, 'health_lifescience
nutritioneducation': 147, 'esl foreignlanguages': 148,
'environmentalscience health_wellness': 149, 'extracurricular
mathematics': 150, 'appliedsciences health_wellness': 151,
'literature_writing music': 152, 'financialliteracy specialneeds':
153, 'health_wellness visualarts': 154, 'college_careerprep
health_lifescience': 155, 'music visualarts': 156, 'communityservice
environmentalscience': 157, 'health_wellness music': 158,
'parentinvolvement': 159, 'specialneeds teamsports': 160,
'college_careerprep extracurricular': 161, 'earlydevelopment
parentinvolvement': 162, 'earlydevelopment health_lifescience': 163,
'environmentalscience nutritioneducation': 164, 'foreignlanguages
mathematics': 165, 'esl environmentalscience': 166, 'esl visualarts':
167, 'esl health_lifescience': 168, 'extracurricular literacy': 169,
'earlydevelopment gym_fitness': 170, 'esl history_geography': 171,
'college_careerprep parentinvolvement': 172, 'parentinvolvement
visualarts': 173, 'communityservice visualarts': 174, 'extracurricular
other': 175, 'gym_fitness mathematics': 176, 'economics': 177,
'nutritioneducation specialneeds': 178, 'college_careerprep
environmentalscience': 179, 'extracurricular literature_writing': 180,
'appliedsciences performingarts': 181, 'mathematics performingarts':
182, 'charactereducation parentinvolvement': 183, 'communityservice
literature_writing': 184, 'financialliteracy literacy': 185,
'gym_fitness literacy': 186, 'earlydevelopment performingarts': 187,
'parentinvolvement specialneeds': 188, 'charactereducation
environmentalscience': 189, 'college_careerprep socialsciences': 190,
'gym_fitness music': 191, 'extracurricular teamsports': 192,
'health_wellness history_geography': 193, 'charactereducation music':
194, 'history_geography music': 195, 'communityservice specialneeds':
196, 'college_careerprep history_geography': 197, 'college_careerprep
communityservice': 198, 'civics_government specialneeds': 199, 'esl
health_wellness': 200, 'extracurricular music': 201,
'charactereducation health_lifescience': 202, 'college_careerprep

earlydevelopment': 203, 'economics history_geography': 204,
'performingarts specialneeds': 205, 'college_careerprep
performingarts': 206, 'charactereducation teamsports': 207,
'college_careerprep health_wellness': 208, 'earlydevelopment music':
209, 'extracurricular performingarts': 210, 'foreignlanguages
history_geography': 211, 'civics_government economics': 212,
'economics mathematics': 213, 'extracurricular specialneeds': 214,
'esl socialsciences': 215, 'foreignlanguages specialneeds': 216,
'specialneeds warmth care_hunger': 217, 'history_geography
performingarts': 218, 'civics_government financialliteracy': 219,
'college_careerprep foreignlanguages': 220, 'charactereducation
socialsciences': 221, 'health_wellness warmth care_hunger': 222,
'charactereducation esl': 223, 'other parentinvolvement': 224,
'environmentalscience performingarts': 225, 'health_wellness
performingarts': 226, 'esl music': 227, 'civics_government
environmentalscience': 228, 'financialliteracy literature_writing':
229, 'gym_fitness literature_writing': 230, 'communityservice
extracurricular': 231, 'esl performingarts': 232, 'health_wellness
socialsciences': 233, 'financialliteracy history_geography': 234,
'communityservice literacy': 235, 'civics_government
health_lifescience': 236, 'appliedsciences gym_fitness': 237,
'earlydevelopment socialsciences': 238, 'communityservice
mathematics': 239, 'esl other': 240, 'appliedsciences
civics_government': 241, 'earlydevelopment extracurricular': 242,
'foreignlanguages visualarts': 243, 'communityservice
health_lifescience': 244, 'gym_fitness other': 245, 'communityservice
health_wellness': 246, 'environmentalscience other': 247,
'civics_government visualarts': 248, 'communityservice other': 249,
'charactereducation performingarts': 250, 'mathematics
nutritioneducation': 251, 'college_careerprep nutritioneducation':
252, 'college_careerprep music': 253, 'gym_fitness performingarts':
254, 'economics literacy': 255, 'performingarts teamsports': 256,
'college_careerprep financialliteracy': 257, 'nutritioneducation
other': 258, 'appliedsciences teamsports': 259, 'communityservice
parentinvolvement': 260, 'civics_government college_careerprep': 261,
'foreignlanguages health_wellness': 262, 'earlydevelopment
nutritioneducation': 263, 'college_careerprep esl': 264,
'health_lifescience teamsports': 265, 'charactereducation
history_geography': 266, 'mathematics teamsports': 267, 'economics
visualarts': 268, 'extracurricular parentinvolvement': 269,
'appliedsciences communityservice': 270, 'literacy teamsports': 271,
'music teamsports': 272, 'health_lifescience other': 273, 'gym_fitness
visualarts': 274, 'health_lifescience parentinvolvement': 275,
'environmentalscience extracurricular': 276, 'economics
socialsciences': 277, 'foreignlanguages socialsciences': 278,
'nutritioneducation teamsports': 279, 'literature_writing teamsports':
280, 'history_geography other': 281, 'health_lifescience music': 282,
'civics_government mathematics': 283, 'foreignlanguages music': 284,
'esl parentinvolvement': 285, 'charactereducation gym_fitness': 286,
'gym_fitness health_lifescience': 287, 'environmentalscience

parentinvolvement': 288, 'extracurricular health_wellness': 289, 'appliedsciences financialliteracy': 290, 'parentinvolvement socialsciences': 291, 'communityservice socialsciences': 292, 'charactereducation foreignlanguages': 293, 'appliedsciences foreignlanguages': 294, 'teamsports visualarts': 295, 'charactereducation financialliteracy': 296, 'civics_government performingarts': 297, 'literacy nutritioneducation': 298, 'music socialsciences': 299, 'civics_government communityservice': 300, 'performingarts socialsciences': 301, 'health_lifescience performingarts': 302, 'charactereducation warmth care_hunger': 303, 'esl financialliteracy': 304, 'appliedsciences nutritioneducation': 305, 'gym_fitness history_geography': 306, 'financialliteracy visualarts': 307, 'foreignlanguages other': 308, 'extracurricular health_lifescience': 309, 'music other': 310, 'earlydevelopment financialliteracy': 311, 'economics environmentalscience': 312, 'communityservice nutritioneducation': 313, 'nutritioneducation visualarts': 314, 'communityservice history_geography': 315, 'communityservice earlydevelopment': 316, 'environmentalscience foreignlanguages': 317, 'nutritioneducation warmth care_hunger': 318, 'esl extracurricular': 319, 'health_wellness parentinvolvement': 320, 'extracurricular socialsciences': 321, 'music parentinvolvement': 322, 'health_lifescience warmth care_hunger': 323, 'financialliteracy health_lifescience': 324, 'esl nutritioneducation': 325, 'appliedsciences economics': 326, 'communityservice performingarts': 327, 'literacy warmth care_hunger': 328, 'civics_government esl': 329, 'economics specialneeds': 330, 'communityservice esl': 331, 'environmentalscience gym_fitness': 332, 'esl gym_fitness': 333, 'literature_writing warmth care_hunger': 334, 'college_careerprep economics': 335, 'environmentalscience financialliteracy': 336, 'foreignlanguages performingarts': 337, 'environmentalscience music': 338, 'extracurricular foreignlanguages': 339, 'nutritioneducation socialsciences': 340, 'earlydevelopment warmth care_hunger': 341, 'environmentalscience warmth care_hunger': 342, 'civics_government extracurricular': 343, 'financialliteracy other': 344, 'earlydevelopment foreignlanguages': 345, 'mathematics warmth care_hunger': 346, 'extracurricular nutritioneducation': 347, 'history_geography parentinvolvement': 348, 'history_geography teamsports': 349, 'socialsciences teamsports': 350, 'extracurricular gym_fitness': 351, 'foreignlanguages health_lifescience': 352, 'environmentalscience teamsports': 353, 'other socialsciences': 354, 'charactereducation economics': 355, 'economics health_lifescience': 356, 'foreignlanguages gym_fitness': 357, 'college_careerprep gym_fitness': 358, 'appliedsciences warmth care_hunger': 359, 'other teamsports': 360, 'earlydevelopment history_geography': 361, 'communityservice economics': 362, 'college_careerprep teamsports': 363, 'economics foreignlanguages': 364, 'financialliteracy performingarts': 365, 'gym_fitness parentinvolvement': 366, 'financialliteracy foreignlanguages': 367, 'civics_government nutritioneducation': 368, 'communityservice music': 369, 'other warmth care_hunger': 370, 'earlydevelopment teamsports': 371,

```
'financialliteracy socialsciences': 372, 'esl economics': 373,
'parentinvolvement performingarts': 374, 'college_careerprep warmth
care_hunger': 375, 'parentinvolvement warmth care_hunger': 376,
'economics literature_writing': 377, 'civics_government teamsports':
378, 'charactereducation nutritioneducation': 379, 'other
performingarts': 380, 'economics nutritioneducation': 381,
'earlydevelopment economics': 382, 'civics_government
parentinvolvement': 383, 'extracurricular financialliteracy': 384,
'history_geography warmth care_hunger': 385, 'communityservice
financialliteracy': 386, 'economics music': 387, 'visualarts warmth
care_hunger': 388, 'charactereducation civics_government': 389,
'financialliteracy parentinvolvement': 390, 'extracurricular
history_geography': 391, 'parentinvolvement teamsports': 392,
'gym_fitness warmth care_hunger': 393, 'gym_fitness socialsciences':
394, 'civics_government health_wellness': 395, 'financialliteracy
health_wellness': 396, 'esl teamsports': 397, 'economics other': 398}

#Train data
x_train_clean_subcategories =
pad_sequences(train_clean_subcategories,maxlen=1)
print(x_train_clean_subcategories.shape)

#Test data
x_test_clean_subcategories =
pad_sequences(test_clean_subcategories,maxlen=1)
print(x_test_clean_subcategories.shape)

(76473, 1)
(32775, 1)

#checking for Nan values
np.isnan(x_test_clean_subcategories).sum()

0
```

5. Teacher prefix

```
#checking total states
print('Total teacher_prefix in Train
=',len(train_df['teacher_prefix'].unique()))
print('Total teacher_prefix in Test =
',len(test_df['teacher_prefix'].unique()))

Total teacher_prefix in Train = 5
Total teacher_prefix in Test =  5

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

#fiitng on train data
teacher_prefix_token.fit_on_texts(train_df['teacher_prefix'])
```

```
print(teacher_prefix_token.word_index)

#creating tokenized sequence
train_teacher_prefix =
teacher_prefix_token.texts_to_sequences(train_df['teacher_prefix'])
test_teacher_prefix =
teacher_prefix_token.texts_to_sequences(test_df['teacher_prefix'])
```

```
{'mrs': 1, 'ms': 2, 'mr': 3, 'teacher': 4, 'dr': 5}
```

```
#Train data
x_train_teacher_prefix = pad_sequences(train_teacher_prefix,maxlen=1)
print(x_train_teacher_prefix.shape)

#Test data
x_test_teacher_prefix = pad_sequences(test_teacher_prefix,maxlen=1)
print(x_test_teacher_prefix.shape)
```

```
(76473, 1)
(32775, 1)
```

```
#checking for Nan values
np.isnan(x_test_teacher_prefix).sum()
```

```
0
```

6. Number of previously submitted projects

```
x_train_remaining_input =
train_df[['teacher_number_of_previously_posted_projects','price']]
x_test_remaining_input =
test_df[['teacher_number_of_previously_posted_projects','price']]
```

7. Essay

```
train_df['essay'] = train_df['essay'].astype(str)
test_df['essay'] = test_df['essay'].astype(str)

# https://machinelearningmastery.com/use-word-embedding-layers-deep-
learning-keras/
#Do Tokenize i.e Assign token to each Number.

#loading tokenizer
essay_token = Tokenizer()

#fitiing on X_train
essay_token.fit_on_texts(train_df['essay'])

#creating word dictionary {word:token_number}
word_index = essay_token.word_index
print('Total words in X_train = ',len(word_index)+1)

#Total words will be equal to len(idf_dict)
```

```
Total words in X_train =  6922
```

```
#loading glove file
glove_file = r'/content/drive/MyDrive/Colab
Notebooks/Datasets/Donor_Choose_LSTM/glove.6B.100d.txt'
f = open(glove_file)

#creating dict {word:100_dim_vector}
glove_embeddings = dict()
for line in f:
  values = line.split()
  word = values[0]
  vector = np.asarray(values[1:], dtype='float32')
  glove_embeddings[word] = vector

f.close()
print(f'Loaded {len(glove_embeddings)} word vectors.')
```

```
Loaded 400000 word vectors.
```

```
#creating embedded matrix which contains GLOVE vector representation
of each word of tokenized words
vocab_size = len(word_index)+1

#each word will be 50 dim GLOVE Vector after loading
embedded_matrix = np.zeros(shape=(vocab_size,100))

#feeding glove vectors in embedding_matrix
for word,index in word_index.items():

  vector = glove_embeddings.get(word)

   #feed only if word is in GLOVE_words else dont feed
  if vector is not None:
    embedded_matrix[index] = vector

print('Shape of Embedded_matrix = ',embedded_matrix.shape)
```

```
Shape of Embedded_matrix =  (6922, 100)
```

```
#checking for Nan values
if np.any(np.isnan(embedded_matrix)):
  print('Nan values are present')
else:
  print('No Nan Values Found')
```

```
No Nan Values Found
```

```
#Padding sequence

# Encoding words of each document in X_train
```

```
train_essay = essay_token.texts_to_sequences(train_df['essay'])

#Using tokenizer fitted on X_train
test_essay = essay_token.texts_to_sequences(test_df['essay'])

max_len = len(max(train_essay,key=len))
print('Max length of Sentence is = ',len(max(train_essay,key=len)))
```

```
Max length of Sentence is =  84
```

```
x_train_essay =
pad_sequences(train_essay,maxlen=max_len,padding='post')
x_test_essay = pad_sequences(test_essay,maxlen=max_len,padding='post')

#Checking sample datapoint
i = np.random.randint(low=0,high=len(test_essay))

#if count of non_zeros after padding is equal or not to
len(train_essay datapoint)

#Train
print(np.count_nonzero(x_train_essay[i]) == len(train_essay[i]))

#Test
print(np.count_nonzero(x_test_essay[i]) == len(test_essay[i]))
```

```
True
True
```

Training Model_2

```
#preparing vocab size for embedding layers

token_list =
[essay_token,school_token,project_grade_token,clean_categories_token,
                  clean_subcategories_token,teacher_prefix_token]
token_str_list =
['essay_token','school_token','project_grade_token','clean_categories_
token',
                  'clean_subcategories_token','teacher_prefix_token']

for i,j in enumerate(token_list):
  print(f'{i+1}. {token_str_list[i]} vocab_size is  =
{len(j.word_index)+1}')

print('=='*80)
```

```
1. essay_token vocab_size is  = 6922
2. school_token vocab_size is  = 52
3. project_grade_token vocab_size is  = 5
4. clean_categories_token vocab_size is  = 52
5. clean_subcategories_token vocab_size is  = 399
```

```
6. teacher_prefix_token vocab_size is  = 6
==========================================================================
==========================================================================
====================

#Creating input of train and test dataset for model_2

x_train =
[x_train_essay,x_train_school_state,x_train_project_grade,x_train_clea
n_categories,

x_train_clean_subcategories,x_train_teacher_prefix,x_train_remaining_i
nput]

x_test =
[x_test_essay,x_test_school_state,x_test_project_grade,x_test_clean_ca
tegories,

x_test_clean_subcategories,x_test_teacher_prefix,x_test_remaining_inpu
t]

# Labels
y_train = to_categorical(train_df['project_is_approved'],)
y_test = to_categorical(test_df['project_is_approved'])

#printing shape
print('Train = ',y_train.shape)
print('Test = ',y_test.shape)

Train =  (76473, 2)
Test =  (32775, 2)

#Model_2

#input_1 Essay
ip_1 = Input(shape=(max_len,),name='essay_input')

l1 =
Embedding(input_dim=vocab_size,output_dim=100,input_length=max_len,
                      embeddings_initializer
=Constant(embedded_matrix),
                      trainable=False,name='Embed_layer')(ip_1)

l1 =
LSTM(128,activation='relu',return_sequences=True,name='LSTM_layer')
(l1)
l1 = Flatten(name='Flatten_essay')(l1)

#input_2 school_state
ip_2 = Input(shape=(1,),name='ip_school_state')
```

```
l2 = Embedding(input_dim=52,output_dim=1,name='school_emb')(ip_2)
l2 = Flatten()(l2)

#input project_grade
ip_3 = Input(shape=(1,),name='ip_project_grade')
l3 = Embedding(input_dim=5,output_dim=2,name='project_grade_emb')
(ip_3)
l3 = Flatten()(l3)

#input clean_category
ip_4 = Input(shape=(1,),name='ip_clean_cat')
l4 = Embedding(input_dim=52,output_dim=2,name='clean_cat_emb')(ip_4)
l4 = Flatten()(l4)

#input clean_subcategory
ip_5 = Input(shape=(1,),name='ip_clean_subcat')
l5 = Embedding(input_dim=393,output_dim=64,name='clean_subcat_emb')
(ip_5)
l5 = Flatten()(l5)

#input teacher_prefix
ip_6 = Input(shape=(1,),name='ip_teacher_prefix')
l6 = Embedding(input_dim = 6,output_dim = 2,name =
'teacher_prefix_emb')(ip_6)
l6 = Flatten()(l6)

# input remaining input
ip_7 = Input(shape=(2,),name='remaining input')
l7 = Dense(16,activation='relu')(ip_7)

#Concatenate all inputs
concat = concatenate([l1,l2,l3,l4,l5,l5,l6,l7])

x = Dense(64, activation='relu')(concat)
x = Dense(32,activation='relu')(x)
x = Dropout(0.5)(x)
# x = BatchNormalization()(x)
x = Dense(18,activation='relu')(x)
x = Dropout(0.6)(x)
output = Dense(2, activation = 'softmax')(x)

# model with all the inputs
model_2 = Model([ip_1,ip_2,ip_3,ip_4,ip_5,ip_6,ip_7], output)

#checking model
model_2.summary()
```

```
WARNING:tensorflow:Layer LSTM_layer will not use cuDNN kernels since
it doesn't meet the criteria. It will use a generic GPU kernel as
fallback when running on GPU.

Model: "model_11"
__________________________________________________________________________
_______________________________
 Layer (type)                   Output Shape         Param #
Connected to
==========================================================================
==============================
 essay_input (InputLayer)       [(None, 84)]         0           []



 Embed_layer (Embedding)        (None, 84, 100)      692200
['essay_input[0][0]']

 ip_school_state (InputLayer)   [(None, 1)]          0           []



 ip_project_grade (InputLayer)  [(None, 1)]          0           []



 ip_clean_cat (InputLayer)      [(None, 1)]          0           []



 ip_clean_subcat (InputLayer)   [(None, 1)]          0           []



 ip_teacher_prefix (InputLayer)  [(None, 1)]         0           []



 LSTM_layer (LSTM)              (None, 84, 128)      117248
['Embed_layer[0][0]']

 school_emb (Embedding)         (None, 1, 1)         52
['ip_school_state[0][0]']

 project_grade_emb (Embedding)  (None, 1, 2)         10
```

```
['ip_project_grade[0][0]']


 clean_cat_emb (Embedding)      (None, 1, 2)         104
['ip_clean_cat[0][0]']


 clean_subcat_emb (Embedding)   (None, 1, 64)        25152
['ip_clean_subcat[0][0]']


 teacher_prefix_emb (Embedding)  (None, 1, 2)        12
['ip_teacher_prefix[0][0]']


 remaining input (InputLayer)   [(None, 2)]          0           []



 Flatten_essay (Flatten)        (None, 10752)        0
['LSTM_layer[0][0]']


 flatten_52 (Flatten)           (None, 1)            0
['school_emb[0][0]']


 flatten_53 (Flatten)           (None, 2)            0
['project_grade_emb[0][0]']


 flatten_54 (Flatten)           (None, 2)            0
['clean_cat_emb[0][0]']


 flatten_55 (Flatten)           (None, 64)           0
['clean_subcat_emb[0][0]']


 flatten_56 (Flatten)           (None, 2)            0
['teacher_prefix_emb[0][0]']


 dense_54 (Dense)               (None, 16)           48
['remaining input[0][0]']


 concatenate_11 (Concatenate)   (None, 10903)        0
```

```
['Flatten_essay[0][0]',

'flatten_52[0][0]',

'flatten_53[0][0]',

'flatten_54[0][0]',

'flatten_55[0][0]',

'flatten_55[0][0]',

'flatten_56[0][0]',

'dense_54[0][0]']


 dense_55 (Dense)               (None, 64)           697856
['concatenate_11[0][0]']


 dense_56 (Dense)               (None, 32)           2080
['dense_55[0][0]']


 dropout_24 (Dropout)           (None, 32)           0
['dense_56[0][0]']


 dense_57 (Dense)               (None, 18)           594
['dropout_24[0][0]']


 dropout_25 (Dropout)           (None, 18)           0
['dense_57[0][0]']


 dense_58 (Dense)               (None, 2)            38
['dropout_25[0][0]']


==================================================================
============================
Total params: 1,535,394
Trainable params: 843,194
Non-trainable params: 692,200
__________________________________________________________________
_____________________________
```

```
# Clear any logs from previous runs
%load_ext tensorboard
# !rm -rf ./logs/

log_dir = "logs/fit/" + "model_2"
tensorboard_callback =
tensorflow.keras.callbacks.TensorBoard(log_dir=log_dir,
histogram_freq=1)
```

```
The tensorboard extension is already loaded. To reload it, use:
  %reload_ext tensorboard
```

```
#compiling model
model_2.compile(loss='categorical_crossentropy',
optimizer=Adam(learning_rate=0.0006,), metrics=[aucroc])

#saving best model
# https://machinelearningmastery.com/check-point-deep-learning-models-
keras/

filepath="Model_2_{epoch:02d}_{val_aucroc:.2f}.hdf5"
checkpoint = ModelCheckpoint(filepath, monitor='val_aucroc',
verbose=1, save_best_only=True, mode='max')

#Callbacks List
callbacks_list = [checkpoint,tensorboard_callback]

#As data is heavily imbalanced calculating class weights
class_weights =
class_weight.compute_class_weight(class_weight='balanced',classes=np.u
nique(y_train[:,0]),y=train_df['project_is_approved'].tolist())
print(class_weights)
class_weights = {0:3.30687,1:0.58906}
```

```
[3.30222817 0.58921472]
```

```
model_2.fit(x_train,
y_train,epochs=10,verbose=1,batch_size=512,validation_data=[x_test,y_t
est],validation_batch_size=256,callbacks=callbacks_list,
            class_weight=class_weights,shuffle=True)
```

```
Epoch 1/10
150/150 [==============================] - ETA: 0s - loss: 0.7921 -
aucroc: 0.5466
Epoch 1: val_aucroc improved from -inf to 0.60183, saving model to
Model_2_01_0.60.hdf5
150/150 [==============================] - 32s 196ms/step - loss:
0.7921 - aucroc: 0.5466 - val_loss: 0.6746 - val_aucroc: 0.6018
Epoch 2/10
150/150 [==============================] - ETA: 0s - loss: 0.6875 -
aucroc: 0.5674
```

```
Epoch 2: val_aucroc improved from 0.60183 to 0.62663, saving model to
Model_2_02_0.63.hdf5
150/150 [==============================] - 29s 194ms/step - loss:
0.6875 - aucroc: 0.5674 - val_loss: 0.6543 - val_aucroc: 0.6266
Epoch 3/10
150/150 [==============================] - ETA: 0s - loss: 0.6829 -
aucroc: 0.5864
Epoch 3: val_aucroc improved from 0.62663 to 0.65206, saving model to
Model_2_03_0.65.hdf5
150/150 [==============================] - 28s 188ms/step - loss:
0.6829 - aucroc: 0.5864 - val_loss: 0.6255 - val_aucroc: 0.6521
Epoch 4/10
150/150 [==============================] - ETA: 0s - loss: 0.6807 -
aucroc: 0.5988
Epoch 4: val_aucroc did not improve from 0.65206
150/150 [==============================] - 27s 179ms/step - loss:
0.6807 - aucroc: 0.5988 - val_loss: 0.6677 - val_aucroc: 0.6510
Epoch 5/10
150/150 [==============================] - ETA: 0s - loss: 0.6750 -
aucroc: 0.6128
Epoch 5: val_aucroc improved from 0.65206 to 0.66103, saving model to
Model_2_05_0.66.hdf5
150/150 [==============================] - 28s 187ms/step - loss:
0.6750 - aucroc: 0.6128 - val_loss: 0.6334 - val_aucroc: 0.6610
Epoch 6/10
150/150 [==============================] - ETA: 0s - loss: 11.7343 -
aucroc: 0.6162
Epoch 6: val_aucroc did not improve from 0.66103
150/150 [==============================] - 31s 210ms/step - loss:
11.7343 - aucroc: 0.6162 - val_loss: 0.6640 - val_aucroc: 0.6498
Epoch 7/10
150/150 [==============================] - ETA: 0s - loss: 0.6760 -
aucroc: 0.6275
Epoch 7: val_aucroc did not improve from 0.66103
150/150 [==============================] - 27s 182ms/step - loss:
0.6760 - aucroc: 0.6275 - val_loss: 0.6482 - val_aucroc: 0.6587
Epoch 8/10
150/150 [==============================] - ETA: 0s - loss: 1.6687 -
aucroc: 0.6301
Epoch 8: val_aucroc did not improve from 0.66103
150/150 [==============================] - 27s 178ms/step - loss:
1.6687 - aucroc: 0.6301 - val_loss: 0.6743 - val_aucroc: 0.6574
Epoch 9/10
150/150 [==============================] - ETA: 0s - loss: 0.6663 -
aucroc: 0.6380
Epoch 9: val_aucroc did not improve from 0.66103
150/150 [==============================] - 29s 193ms/step - loss:
0.6663 - aucroc: 0.6380 - val_loss: 0.6418 - val_aucroc: 0.6600
Epoch 10/10
150/150 [==============================] - ETA: 0s - loss: 0.6658 -
```

```
aucroc: 0.6361
Epoch 10: val_aucroc did not improve from 0.66103
150/150 [==============================] - 27s 177ms/step - loss:
0.6658 - aucroc: 0.6361 - val_loss: 0.6642 - val_aucroc: 0.6606

<keras.callbacks.History at 0x7fe8926d4890>
```

```
#Loading Best_fit model
# https://github.com/keras-team/keras/issues/10104
dependencies = {'aucroc': aucroc}

#loading best model
loaded_model_2 =
load_model('Model_2_02_0.66.hdf5' ,custom_objects=dependencies)

#checking loaded model
# loaded_model_2.summary()
```

```
WARNING:tensorflow:Layer LSTM_layer will not use cuDNN kernels since
it doesn't meet the criteria. It will use a generic GPU kernel as
fallback when running on GPU.
```

```
#predicting train dataset
train_y_pred = loaded_model_2.predict(x_train)
print('Train AUC_ROC_SCORE =',roc_auc_score(y_train,train_y_pred))
```

```
Train AUC_ROC_SCORE = 0.6651130231542259
```

```
#predicting test dataset
test_y_pred = loaded_model_2.predict(x_test)
print('Test AUC_ROC_SCORE =',roc_auc_score(y_test,test_y_pred))
```

```
Test AUC_ROC_SCORE = 0.6563042731515965
```

```
#checking tensorboard
%tensorboard --logdir logs/fit
```

```
Reusing TensorBoard on port 6006 (pid 515), started 0:20:12 ago. (Use
'!kill 515' to kill it.)

<IPython.core.display.Javascript object>
```

Model_3

Text Vectorizing Data

1. School State

```
#checking total states
print('Total School states in Train
=',len(train_df['school_state'].unique()))
```

```
print('Total School states in Test =
',len(test_df['school_state'].unique()))
```

```
Total School states in Train = 51
Total School states in Test =  51
```

```
#Creating school tokenizer
school_token = Tokenizer()

#fiitng on train data
school_token.fit_on_texts(train_df['school_state'])
print(school_token.word_index)

#creating tokenized sequence
train_school_state =
school_token.texts_to_sequences(train_df['school_state'])
test_school_state =
school_token.texts_to_sequences(test_df['school_state'])
```

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

```
#Train data
x_train_school_state = pad_sequences(train_school_state,maxlen=1)
print(x_train_school_state.shape)

#Test data
x_test_school_state = pad_sequences(test_school_state,maxlen=1)
print(x_test_school_state.shape)
```

```
(76473, 1)
(32775, 1)
```

```
#checking for Nan values
np.isnan(x_test_school_state).sum()
```

```
0
```

2. Project Grade

```
train_df['project_grade_category'].unique()
```

```
array(['grades35', 'gradesprek2', 'grades68', 'grades912'],
dtype=object)
```

```
#checking total states
print('Total Project_Grades in Train
```

```
='',len(train_df['project_grade_category'].unique()))
print('Total Project_Grades in Test =
',len(test_df['project_grade_category'].unique()))
```

```
Total Project_Grades in Train = 4
Total Project_Grades in Test =  4
```

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

#fiitng on train data
project_grade_token.fit_on_texts(train_df['project_grade_category'])
print(project_grade_token.word_index)

#creating tokenized sequence
train_project_grade =
project_grade_token.texts_to_sequences(train_df['project_grade_categor
y'])
test_project_grade =
project_grade_token.texts_to_sequences(test_df['project_grade_category
'])
```

```
{'gradesprek2': 1, 'grades35': 2, 'grades68': 3, 'grades912': 4}
```

```
#Train data
x_train_project_grade = pad_sequences(train_project_grade,maxlen=1)
print(x_train_project_grade.shape)

#Test data
x_test_project_grade = pad_sequences(test_project_grade,maxlen=1)
print(x_test_project_grade.shape)
```

```
(76473, 1)
(32775, 1)
```

```
#checking for Nan values
np.isnan(x_test_project_grade).sum()
```

```
0
```

3. clean_categories

```
#checking total states
print('Total clean_categories in Train
=',len(train_df['clean_categories'].unique()))
print('Total clean_categories in Test =
',len(test_df['clean_categories'].unique()))
```

```
Total clean_categories in Train = 51
Total clean_categories in Test =  50
```

```
train_df['clean_categories'].value_counts()
```

literacy_language	16643
math_science	12022
literacy_language math_science	10188
health_sports	7050
music_arts	3630
specialneeds	2960
literacy_language specialneeds	2792
appliedlearning	2668
math_science literacy_language	1590
appliedlearning literacy_language	1511
math_science specialneeds	1286
history_civics	1281
literacy_language music_arts	1211
math_science music_arts	1142
appliedlearning specialneeds	1022
history_civics literacy_language	1010
health_sports specialneeds	977
warmth care_hunger	935
math_science appliedlearning	826
appliedlearning math_science	739
literacy_language history_civics	577
health_sports literacy_language	564
appliedlearning music_arts	538
math_science history_civics	436
literacy_language appliedlearning	430
appliedlearning health_sports	428
math_science health_sports	301
history_civics math_science	223
specialneeds music_arts	213
history_civics music_arts	209
health_sports math_science	181
history_civics specialneeds	179
health_sports appliedlearning	142
health_sports music_arts	118
appliedlearning history_civics	114
music_arts specialneeds	98
literacy_language health_sports	50
health_sports history_civics	34
specialneeds health_sports	30
history_civics appliedlearning	27
health_sports warmth care_hunger	17
music_arts health_sports	15
specialneeds warmth care_hunger	14
music_arts history_civics	10
appliedlearning warmth care_hunger	9
math_science warmth care_hunger	9
history_civics health_sports	9
music_arts appliedlearning	7
literacy_language warmth care_hunger	6
history_civics warmth care_hunger	1

```
music_arts warmth care_hunger              1
Name: clean_categories, dtype: int64
```

```
train_df.tail()
```

```
      school_state teacher_prefix project_grade_category  \
55558           la             ms               grades35
35962           nc            mrs            gradesprek2
86373           sc             mr              grades912
46563           il             ms            gradesprek2
30742           tn            mrs               grades68

       teacher_number_of_previously_posted_projects
project_is_approved  \
55558                                             0
1
35962                                            14
1
86373                                             1
1
46563                                             3
1
30742                                             0
0

                    clean_categories          clean_subcategories  \
55558                   specialneeds                  specialneeds
35962  health_sports appliedlearning       health_wellness other
86373                   math_science              appliedsciences
46563                  health_sports                  gym_fitness
30742                   math_science  appliedsciences mathematics

                                                   essay    price
55558  classroom consists children significant disabi...  1511.09
35962  person person matter small dr seuss classroom ...   120.07
86373  students starting path become certified pc tec...    49.37
46563  school year new different makes teaching speci...   259.99
30742  roughly students still access internet home li...   799.45
```

```
#Creating school tokenizer
clean_categories_token = Tokenizer(filters='!"#$%&()*+,-.;<=>?
@^`{|}~',lower=False,split='  ')

#fiitng on train data
clean_categories_token.fit_on_texts(train_df['clean_categories'])
print(clean_categories_token.word_index)

#creating tokenized sequence
train_clean_categories =
clean_categories_token.texts_to_sequences(train_df['clean_categories']
```

```
)
test_clean_categories =
clean_categories_token.texts_to_sequences(test_df['clean_categories'])
```

```
{'literacy_language': 1, 'math_science': 2, 'literacy_language
math_science': 3, 'health_sports': 4, 'music_arts': 5, 'specialneeds':
6, 'literacy_language specialneeds': 7, 'appliedlearning': 8,
'math_science literacy_language': 9, 'appliedlearning
literacy_language': 10, 'math_science specialneeds': 11,
'history_civics': 12, 'literacy_language music_arts': 13,
'math_science music_arts': 14, 'appliedlearning specialneeds': 15,
'history_civics literacy_language': 16, 'health_sports specialneeds':
17, 'warmth care_hunger': 18, 'math_science appliedlearning': 19,
'appliedlearning math_science': 20, 'literacy_language
history_civics': 21, 'health_sports literacy_language': 22,
'appliedlearning music_arts': 23, 'math_science history_civics': 24,
'literacy_language appliedlearning': 25, 'appliedlearning
health_sports': 26, 'math_science health_sports': 27, 'history_civics
math_science': 28, 'specialneeds music_arts': 29, 'history_civics
music_arts': 30, 'health_sports math_science': 31, 'history_civics
specialneeds': 32, 'health_sports appliedlearning': 33, 'health_sports
music_arts': 34, 'appliedlearning history_civics': 35, 'music_arts
specialneeds': 36, 'literacy_language health_sports': 37,
'health_sports history_civics': 38, 'specialneeds health_sports': 39,
'history_civics appliedlearning': 40, 'health_sports warmth
care_hunger': 41, 'music_arts health_sports': 42, 'specialneeds warmth
care_hunger': 43, 'music_arts history_civics': 44, 'appliedlearning
warmth care_hunger': 45, 'math_science warmth care_hunger': 46,
'history_civics health_sports': 47, 'music_arts appliedlearning': 48,
'literacy_language warmth care_hunger': 49, 'history_civics warmth
care_hunger': 50, 'music_arts warmth care_hunger': 51}
```

```
#Train data
x_train_clean_categories =
pad_sequences(train_clean_categories,maxlen=1)
print(x_train_clean_categories.shape)

#Test data
x_test_clean_categories =
pad_sequences(test_clean_categories,maxlen=1)
print(x_test_clean_categories.shape)
```

```
(76473, 1)
(32775, 1)
```

```
#checking for Nan values
np.isnan(x_test_clean_categories).sum()
```

```
0
```

4. clean_subcategories

```
#checking total states
print('Total clean_subcategories in Train
=',len(train_df['clean_subcategories'].unique()))
print('Total clean_subcategories in Test =
',len(test_df['clean_subcategories'].unique()))
```

```
Total clean_subcategories in Train = 398
Total clean_subcategories in Test =  360
```

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

#fiitng on train data
clean_subcategories_token.fit_on_texts(train_df['clean_subcategories']
)
print(clean_subcategories_token.word_index)

#creating tokenized sequence
train_clean_subcategories =
clean_subcategories_token.texts_to_sequences(train_df['clean_subcatego
ries'])
test_clean_subcategories =
clean_subcategories_token.texts_to_sequences(test_df['clean_subcategor
ies'])
```

```
{'literacy': 1, 'literacy mathematics': 2, 'literature_writing
mathematics': 3, 'literacy literature_writing': 4, 'mathematics': 5,
'literature_writing': 6, 'specialneeds': 7, 'health_wellness': 8,
'appliedsciences mathematics': 9, 'appliedsciences': 10, 'literacy
specialneeds': 11, 'esl literacy': 12, 'visualarts': 13, 'gym_fitness
health_wellness': 14, 'music': 15, 'literature_writing specialneeds':
16, 'warmth care_hunger': 17, 'mathematics specialneeds': 18,
'gym_fitness': 19, 'health_wellness specialneeds': 20,
'environmentalscience': 21, 'teamsports': 22, 'environmentalscience
health_lifescience': 23, 'appliedsciences environmentalscience': 24,
'music performingarts': 25, 'earlydevelopment': 26, 'other': 27,
'environmentalscience mathematics': 28, 'health_lifescience': 29,
'earlydevelopment specialneeds': 30, 'health_wellness
nutritioneducation': 31, 'esl literature_writing': 32,
'earlydevelopment literacy': 33, 'literature_writing visualarts': 34,
'history_geography literature_writing': 35, 'gym_fitness teamsports':
36, 'appliedsciences visualarts': 37, 'appliedsciences
health_lifescience': 38, 'history_geography': 39, 'appliedsciences
literacy': 40, 'health_lifescience mathematics': 41,
'history_geography literacy': 42, 'literacy visualarts': 43,
'mathematics visualarts': 44, 'health_wellness literacy': 45,
'college_careerprep': 46, 'environmentalscience literacy': 47,
'performingarts': 48, 'esl': 49, 'appliedsciences literature_writing':
```

50, 'appliedsciences college_careerprep': 51, 'literacy socialsciences': 52, 'appliedsciences specialneeds': 53, 'health_wellness teamsports': 54, 'foreignlanguages': 55, 'literature_writing socialsciences': 56, 'college_careerprep literature_writing': 57, 'charactereducation literacy': 58, 'charactereducation': 59, 'health_lifescience literacy': 60, 'earlydevelopment health_wellness': 61, 'college_careerprep mathematics': 62, 'specialneeds visualarts': 63, 'history_geography socialsciences': 64, 'environmentalscience literature_writing': 65, 'health_wellness literature_writing': 66, 'earlydevelopment mathematics': 67, 'other specialneeds': 68, 'esl mathematics': 69, 'nutritioneducation': 70, 'civics_government history_geography': 71, 'college_careerprep literacy': 72, 'foreignlanguages literacy': 73, 'health_wellness mathematics': 74, 'environmentalscience visualarts': 75, 'earlydevelopment literature_writing': 76, 'esl specialneeds': 77, 'charactereducation specialneeds': 78, 'socialsciences': 79, 'health_lifescience literature_writing': 80, 'gym_fitness specialneeds': 81, 'health_wellness other': 82, 'charactereducation literature_writing': 83, 'literacy other': 84, 'health_lifescience health_wellness': 85, 'earlydevelopment visualarts': 86, 'charactereducation earlydevelopment': 87, 'history_geography visualarts': 88, 'appliedsciences earlydevelopment': 89, 'financialliteracy': 90, 'environmentalscience specialneeds': 91, 'environmentalscience history_geography': 92, 'literacy parentinvolvement': 93, 'financialliteracy mathematics': 94, 'earlydevelopment other': 95, 'appliedsciences extracurricular': 96, 'literacy music': 97, 'literature_writing other': 98, 'college_careerprep visualarts': 99, 'civics_government literacy': 100, 'extracurricular': 101, 'health_lifescience specialneeds': 102, 'literacy performingarts': 103, 'college_careerprep specialneeds': 104, 'history_geography specialneeds': 105, 'appliedsciences other': 106, 'music specialneeds': 107, 'charactereducation college_careerprep': 108, 'charactereducation health_wellness': 109, 'literature_writing performingarts': 110, 'performingarts visualarts': 111, 'charactereducation mathematics': 112, 'history_geography mathematics': 113, 'extracurricular visualarts': 114, 'charactereducation other': 115, 'health_lifescience visualarts': 116, 'mathematics parentinvolvement': 117, 'mathematics other': 118, 'civics_government literature_writing': 119, 'appliedsciences history_geography': 120, 'economics financialliteracy': 121, 'civics_government socialsciences': 122, 'college_careerprep other': 123, 'mathematics socialsciences': 124, 'foreignlanguages literature_writing': 125, 'civics_government': 126, 'environmentalscience socialsciences': 127, 'appliedsciences music': 128, 'appliedsciences esl': 129, 'health_lifescience history_geography': 130, 'esl earlydevelopment': 131, 'gym_fitness nutritioneducation': 132, 'appliedsciences socialsciences': 133, 'other visualarts': 134, 'literature_writing parentinvolvement': 135, 'earlydevelopment environmentalscience': 136, 'mathematics music': 137, 'health_lifescience socialsciences': 138, 'charactereducation

visualarts': 139, 'socialsciences specialneeds': 140,
'communityservice': 141, 'charactereducation communityservice': 142,
'appliedsciences charactereducation': 143, 'socialsciences
visualarts': 144, 'charactereducation extracurricular': 145,
'appliedsciences parentinvolvement': 146, 'health_lifescience
nutritioneducation': 147, 'esl foreignlanguages': 148,
'environmentalscience health_wellness': 149, 'extracurricular
mathematics': 150, 'appliedsciences health_wellness': 151,
'literature_writing music': 152, 'financialliteracy specialneeds':
153, 'health_wellness visualarts': 154, 'college_careerprep
health_lifescience': 155, 'music visualarts': 156, 'communityservice
environmentalscience': 157, 'health_wellness music': 158,
'parentinvolvement': 159, 'specialneeds teamsports': 160,
'college_careerprep extracurricular': 161, 'earlydevelopment
parentinvolvement': 162, 'earlydevelopment health_lifescience': 163,
'environmentalscience nutritioneducation': 164, 'foreignlanguages
mathematics': 165, 'esl environmentalscience': 166, 'esl visualarts':
167, 'esl health_lifescience': 168, 'extracurricular literacy': 169,
'earlydevelopment gym_fitness': 170, 'esl history_geography': 171,
'college_careerprep parentinvolvement': 172, 'parentinvolvement
visualarts': 173, 'communityservice visualarts': 174, 'extracurricular
other': 175, 'gym_fitness mathematics': 176, 'economics': 177,
'nutritioneducation specialneeds': 178, 'college_careerprep
environmentalscience': 179, 'extracurricular literature_writing': 180,
'appliedsciences performingarts': 181, 'mathematics performingarts':
182, 'charactereducation parentinvolvement': 183, 'communityservice
literature_writing': 184, 'financialliteracy literacy': 185,
'gym_fitness literacy': 186, 'earlydevelopment performingarts': 187,
'parentinvolvement specialneeds': 188, 'charactereducation
environmentalscience': 189, 'college_careerprep socialsciences': 190,
'gym_fitness music': 191, 'extracurricular teamsports': 192,
'health_wellness history_geography': 193, 'charactereducation music':
194, 'history_geography music': 195, 'communityservice specialneeds':
196, 'college_careerprep history_geography': 197, 'college_careerprep
communityservice': 198, 'civics_government specialneeds': 199, 'esl
health_wellness': 200, 'extracurricular music': 201,
'charactereducation health_lifescience': 202, 'college_careerprep
earlydevelopment': 203, 'economics history_geography': 204,
'performingarts specialneeds': 205, 'college_careerprep
performingarts': 206, 'charactereducation teamsports': 207,
'college_careerprep health_wellness': 208, 'earlydevelopment music':
209, 'extracurricular performingarts': 210, 'foreignlanguages
history_geography': 211, 'civics_government economics': 212,
'economics mathematics': 213, 'extracurricular specialneeds': 214,
'esl socialsciences': 215, 'foreignlanguages specialneeds': 216,
'specialneeds warmth care_hunger': 217, 'history_geography
performingarts': 218, 'civics_government financialliteracy': 219,
'college_careerprep foreignlanguages': 220, 'charactereducation
socialsciences': 221, 'health_wellness warmth care_hunger': 222,
'charactereducation esl': 223, 'other parentinvolvement': 224,

'environmentalscience performingarts': 225, 'health_wellness performingarts': 226, 'esl music': 227, 'civics_government environmentalscience': 228, 'financialliteracy literature_writing': 229, 'gym_fitness literature_writing': 230, 'communityservice extracurricular': 231, 'esl performingarts': 232, 'health_wellness socialsciences': 233, 'financialliteracy history_geography': 234, 'communityservice literacy': 235, 'civics_government health_lifescience': 236, 'appliedsciences gym_fitness': 237, 'earlydevelopment socialsciences': 238, 'communityservice mathematics': 239, 'esl other': 240, 'appliedsciences civics_government': 241, 'earlydevelopment extracurricular': 242, 'foreignlanguages visualarts': 243, 'communityservice health_lifescience': 244, 'gym_fitness other': 245, 'communityservice health_wellness': 246, 'environmentalscience other': 247, 'civics_government visualarts': 248, 'communityservice other': 249, 'charactereducation performingarts': 250, 'mathematics nutritioneducation': 251, 'college_careerprep nutritioneducation': 252, 'college_careerprep music': 253, 'gym_fitness performingarts': 254, 'economics literacy': 255, 'performingarts teamsports': 256, 'college_careerprep financialliteracy': 257, 'nutritioneducation other': 258, 'appliedsciences teamsports': 259, 'communityservice parentinvolvement': 260, 'civics_government college_careerprep': 261, 'foreignlanguages health_wellness': 262, 'earlydevelopment nutritioneducation': 263, 'college_careerprep esl': 264, 'health_lifescience teamsports': 265, 'charactereducation history_geography': 266, 'mathematics teamsports': 267, 'economics visualarts': 268, 'extracurricular parentinvolvement': 269, 'appliedsciences communityservice': 270, 'literacy teamsports': 271, 'music teamsports': 272, 'health_lifescience other': 273, 'gym_fitness visualarts': 274, 'health_lifescience parentinvolvement': 275, 'environmentalscience extracurricular': 276, 'economics socialsciences': 277, 'foreignlanguages socialsciences': 278, 'nutritioneducation teamsports': 279, 'literature_writing teamsports': 280, 'history_geography other': 281, 'health_lifescience music': 282, 'civics_government mathematics': 283, 'foreignlanguages music': 284, 'esl parentinvolvement': 285, 'charactereducation gym_fitness': 286, 'gym_fitness health_lifescience': 287, 'environmentalscience parentinvolvement': 288, 'extracurricular health_wellness': 289, 'appliedsciences financialliteracy': 290, 'parentinvolvement socialsciences': 291, 'communityservice socialsciences': 292, 'charactereducation foreignlanguages': 293, 'appliedsciences foreignlanguages': 294, 'teamsports visualarts': 295, 'charactereducation financialliteracy': 296, 'civics_government performingarts': 297, 'literacy nutritioneducation': 298, 'music socialsciences': 299, 'civics_government communityservice': 300, 'performingarts socialsciences': 301, 'health_lifescience performingarts': 302, 'charactereducation warmth care_hunger': 303, 'esl financialliteracy': 304, 'appliedsciences nutritioneducation': 305, 'gym_fitness history_geography': 306, 'financialliteracy visualarts': 307, 'foreignlanguages other': 308, 'extracurricular

health_lifescience': 309, 'music other': 310, 'earlydevelopment financialliteracy': 311, 'economics environmentalscience': 312, 'communityservice nutritioneducation': 313, 'nutritioneducation visualarts': 314, 'communityservice history_geography': 315, 'communityservice earlydevelopment': 316, 'environmentalscience foreignlanguages': 317, 'nutritioneducation warmth care_hunger': 318, 'esl extracurricular': 319, 'health_wellness parentinvolvement': 320, 'extracurricular socialsciences': 321, 'music parentinvolvement': 322, 'health_lifescience warmth care_hunger': 323, 'financialliteracy health_lifescience': 324, 'esl nutritioneducation': 325, 'appliedsciences economics': 326, 'communityservice performingarts': 327, 'literacy warmth care_hunger': 328, 'civics_government esl': 329, 'economics specialneeds': 330, 'communityservice esl': 331, 'environmentalscience gym_fitness': 332, 'esl gym_fitness': 333, 'literature_writing warmth care_hunger': 334, 'college_careerprep economics': 335, 'environmentalscience financialliteracy': 336, 'foreignlanguages performingarts': 337, 'environmentalscience music': 338, 'extracurricular foreignlanguages': 339, 'nutritioneducation socialsciences': 340, 'earlydevelopment warmth care_hunger': 341, 'environmentalscience warmth care_hunger': 342, 'civics_government extracurricular': 343, 'financialliteracy other': 344, 'earlydevelopment foreignlanguages': 345, 'mathematics warmth care_hunger': 346, 'extracurricular nutritioneducation': 347, 'history_geography parentinvolvement': 348, 'history_geography teamsports': 349, 'socialsciences teamsports': 350, 'extracurricular gym_fitness': 351, 'foreignlanguages health_lifescience': 352, 'environmentalscience teamsports': 353, 'other socialsciences': 354, 'charactereducation economics': 355, 'economics health_lifescience': 356, 'foreignlanguages gym_fitness': 357, 'college_careerprep gym_fitness': 358, 'appliedsciences warmth care_hunger': 359, 'other teamsports': 360, 'earlydevelopment history_geography': 361, 'communityservice economics': 362, 'college_careerprep teamsports': 363, 'economics foreignlanguages': 364, 'financialliteracy performingarts': 365, 'gym_fitness parentinvolvement': 366, 'financialliteracy foreignlanguages': 367, 'civics_government nutritioneducation': 368, 'communityservice music': 369, 'other warmth care_hunger': 370, 'earlydevelopment teamsports': 371, 'financialliteracy socialsciences': 372, 'esl economics': 373, 'parentinvolvement performingarts': 374, 'college_careerprep warmth care_hunger': 375, 'parentinvolvement warmth care_hunger': 376, 'economics literature_writing': 377, 'civics_government teamsports': 378, 'charactereducation nutritioneducation': 379, 'other performingarts': 380, 'economics nutritioneducation': 381, 'earlydevelopment economics': 382, 'civics_government parentinvolvement': 383, 'extracurricular financialliteracy': 384, 'history_geography warmth care_hunger': 385, 'communityservice financialliteracy': 386, 'economics music': 387, 'visualarts warmth care_hunger': 388, 'charactereducation civics_government': 389, 'financialliteracy parentinvolvement': 390, 'extracurricular history_geography': 391, 'parentinvolvement teamsports': 392,

```
'gym_fitness warmth care_hunger': 393, 'gym_fitness socialsciences':
394, 'civics_government health_wellness': 395, 'financialliteracy
health_wellness': 396, 'esl teamsports': 397, 'economics other': 398}

#Train data
x_train_clean_subcategories =
pad_sequences(train_clean_subcategories,maxlen=1)
print(x_train_clean_subcategories.shape)

#Test data
x_test_clean_subcategories =
pad_sequences(test_clean_subcategories,maxlen=1)
print(x_test_clean_subcategories.shape)

(76473, 1)
(32775, 1)

#checking for Nan values
np.isnan(x_test_clean_subcategories).sum()

0
```

5. Teacher prefix

```
#checking total states
print('Total teacher_prefix in Train
=',len(train_df['teacher_prefix'].unique()))
print('Total teacher_prefix in Test =
',len(test_df['teacher_prefix'].unique()))

Total teacher_prefix in Train = 5
Total teacher_prefix in Test =  5

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

#fiitng on train data
teacher_prefix_token.fit_on_texts(train_df['teacher_prefix'])
print(teacher_prefix_token.word_index)

#creating tokenized sequence
train_teacher_prefix =
teacher_prefix_token.texts_to_sequences(train_df['teacher_prefix'])
test_teacher_prefix =
teacher_prefix_token.texts_to_sequences(test_df['teacher_prefix'])

{'mrs': 1, 'ms': 2, 'mr': 3, 'teacher': 4, 'dr': 5}

#Train data
x_train_teacher_prefix = pad_sequences(train_teacher_prefix,maxlen=1)
print(x_train_teacher_prefix.shape)
```

```
#Test data
x_test_teacher_prefix = pad_sequences(test_teacher_prefix,maxlen=1)
print(x_test_teacher_prefix.shape)
```

```
(76473, 1)
(32775, 1)
```

```
#checking for Nan values
np.isnan(x_test_teacher_prefix).sum()
```

```
0
```

6. Number of previously submitted projects

```
df.columns
```

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

```
x_train_remaining_input =
train_df[['teacher_number_of_previously_posted_projects','price']]
x_test_remaining_input =
test_df[['teacher_number_of_previously_posted_projects','price']]
```

7. Essay

```
# https://machinelearningmastery.com/use-word-embedding-layers-deep-
learning-keras/
#Do Tokenizer i.e Assign token to each Number.

#loading tokenizer
essay_token = Tokenizer()

#fitiing on X_train
essay_token.fit_on_texts(train_df['essay'])

#creating word dictionary {word:token_number}
word_index = essay_token.word_index
print('Total words in X_train = ',len(word_index)+1)
```

```
Total words in X_train =  47297
```

```
#loading glove file
glove_file = r'/content/drive/MyDrive/Colab
Notebooks/Datasets/Donor_Choose_LSTM/glove.6B.100d.txt'
f = open(glove_file)

#creating dict {word:100_dim_vector}
glove_embeddings = dict()
for line in f:
```

```
  values = line.split()
  word = values[0]
  vector = np.asarray(values[1:], dtype='float32')
  glove_embeddings[word] = vector

f.close()
print(f'Loaded {len(glove_embeddings)} word vectors.')
```

```
Loaded 400000 word vectors.
```

```
#creating embedded matrix which contains GLOVE vector representation
of each word of tokenized words
vocab_size = len(word_index)+1

#each word will be 50 dim GLOVE Vector after loading
embedded_matrix = np.zeros(shape=(vocab_size,100))

#feeding glove vectors in embedding_matrix
for word,index in word_index.items():

  vector = glove_embeddings.get(word)

   #feed only if word is in GLOVE_words else dont feed
  if vector is not None:
    embedded_matrix[index] = vector

print('Shape of Embedded_matrix = ',embedded_matrix.shape)
```

```
Shape of Embedded_matrix =  (47297, 100)
```

```
#checking for Nan values
if np.any(np.isnan(embedded_matrix)):
  print('Nan values are present')
else:
  print('No Nan Values Found')
```

```
No Nan Values Found
```

```
#Padding sequence

# Encoding words of each document in X_train
train_essay = essay_token.texts_to_sequences(train_df['essay'])

#Using tokenizer fitted on X_train
test_essay = essay_token.texts_to_sequences(test_df['essay'])

max_len = len(max(train_essay,key=len))
print('Max length of Sentence is = ',len(max(train_essay,key=len)))
```

```
Max length of Sentence is =  310
```

```
x_train_essay =
pad_sequences(train_essay,maxlen=max_len,padding='post')
x_test_essay = pad_sequences(test_essay,maxlen=max_len,padding='post')

#Checking sample datapoint
i = np.random.randint(low=0,high=len(test_essay))

#if count of non_zeros after padding is equal or not to
len(train_essay datapoint)

#Train
print(np.count_nonzero(x_train_essay[i]) == len(train_essay[i]))

#Test
print(np.count_nonzero(x_test_essay[i]) == len(test_essay[i]))
```

```
True
True
```

Model_3 Training

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

- input_seq_total_text_data:
  - Use text column('essay'), and use the Embedding layer to get word vectors.
  - Use given predefined glove word vectors, don't train any word vectors.
  - Use LSTM that is given above, get the LSTM output and Flatten that output.
  - You are free to preprocess the input text as you needed.
- Other_than_text_data:
  - Convert all your Categorical values to onehot coded and then concatenate all these onehot vectors
  - Neumerical values and use CNN1D as shown in above figure.
  - You are free to choose all CNN parameters like kernel sizes, stride.

Vectorizing categorical and text data

```
from sklearn.preprocessing import OneHotEncoder
from sklearn.feature_extraction.text import CountVectorizer

#school state
ohe_school = OneHotEncoder(sparse=False)
x_train_st = ohe_school.fit_transform(train_df[['school_state']])
x_test_st = ohe_school.transform(test_df[['school_state']])

print(x_test_st.shape)
```

```
(32775, 51)
```

```
#project grade
ohe_project_grade = OneHotEncoder(sparse=False)
x_train_project_grade =
ohe_school.fit_transform(train_df[['project_grade_category']])
x_test_project_grade =
ohe_school.transform(test_df[['project_grade_category']])

print(x_test_project_grade.shape)
```

```
(32775, 4)
```

```
#clean_category
ohe_clean_category = OneHotEncoder(sparse=False)
x_train_clean_category =
ohe_school.fit_transform(train_df[['clean_categories']])
x_test_clean_category =
ohe_school.transform(test_df[['clean_categories']])

#clean subcategory
ohe_clean_subcategory =
OneHotEncoder(handle_unknown='ignore',sparse=False)
x_train_clean_subcategory =
ohe_clean_subcategory.fit_transform(train_df[['clean_subcategories']])
x_test_clean_subcategory =
ohe_clean_subcategory.transform(test_df[['clean_subcategories']])

# teacher_prefix
ohe_teacher_prefix = OneHotEncoder(sparse=False)
x_train_teacher_prefix =
ohe_teacher_prefix.fit_transform(train_df[['teacher_prefix']])
x_test_teacher_prefix =
ohe_teacher_prefix.transform(test_df[['teacher_prefix']])

#number_of_previously_submitted_projects
x_train_submitted_projects =
train_df['teacher_number_of_previously_posted_projects'].values
x_train_submitted_projects = x_train_submitted_projects.reshape(76473,
1)

x_test_submitted_projects =
test_df['teacher_number_of_previously_posted_projects'].values
x_test_submitted_projects = x_test_submitted_projects.reshape(-1, 1)

#Encoding them
from tensorflow.keras.utils import to_categorical

y_train = to_categorical(train_df['project_is_approved'])
y_test = to_categorical(test_df['project_is_approved'])
```

```
#printing shape
print('Train = ',y_train.shape)
print('Test = ',y_test.shape)
```

```
Train =  (76473, 2)
Test =  (32775, 2)
```

```
for i in
[x_train_st,x_train_project_grade,x_train_clean_category,x_train_clean
_subcategory,x_train_teacher_prefix,x_train_submitted_projects]:
  print(i.shape)
```

```
(76473, 51)
(76473, 4)
(76473, 51)
(76473, 398)
(76473, 5)
(76473, 1)
```

```
#stacking all features
x_train_other = np.hstack([x_train_st,
                           x_train_project_grade,
                           x_train_clean_category,
                           x_train_clean_subcategory,
                           x_train_teacher_prefix,
                           x_train_submitted_projects])

x_train_other =
np.array(x_train_other).reshape(x_train_other.shape[0],x_train_other.s
hape[1],1)


x_test_other = np.hstack([x_test_st,
                           x_test_project_grade,
                           x_test_clean_category,
                           x_test_clean_subcategory,
                           x_test_teacher_prefix,
                           x_test_submitted_projects])
x_test_other =
np.array(x_test_other).reshape(x_test_other.shape[0],x_test_other.shap
e[1],1)

print('x_train_other shape = ', x_train_other.shape)
print('x_test_other shape = ', x_test_other.shape)
```

```
x_train_other shape =  (76473, 510, 1)
x_test_other shape =  (32775, 510, 1)
```

```
#creating list of data for inputs
x_train = [x_train_essay,x_train_other]
x_test  = [x_test_essay,x_test_other]
```

Model architecture

```
#Input shapes for text embedding layer and convolution other layer

ip_embed = x_train_essay.shape[1]
print('Input shapes for text embedding layer',x_train_essay.shape[1])

ip_conv = x_train_other.shape[1]
print('x_train_other shape = ', x_train_other.shape[1])
```

```
Input shapes for text embedding layer 310
x_train_other shape =  510
```

```
# input 1
input_1 = Input(shape=(ip_embed,))
x1 =Embedding(vocab_size, 100, weights=[embedded_matrix],
input_length=x_train_essay.shape[1], trainable=False)(input_1)
x1 = Dropout(0.3)(x1)
x1 = LSTM(128,return_sequences=True)(x1)
x1 = Flatten()(x1)

# input 2
input_2 = Input(shape=(ip_conv,1))
x2 = Conv1D(filters=128,kernel_size=3, strides=1)(input_2)
x2 = Conv1D(filters=64,kernel_size=3, strides=1)(x2)
x2 = Flatten()(x2)


# merging both the inputs
concat = concatenate([x1,x2])
x = Dense(128,activation='relu',kernel_initializer=he_normal())
(concat)
x = Dropout(0.5)(x)
x = Dense(64,activation='relu',kernel_initializer=he_normal())(x)
x = Dropout(0.5)(x)
x = BatchNormalization()(x)
x = Dense(32,activation='relu',kernel_initializer=he_normal())(x)
x = Dropout(0.6)(x)
output = Dense(2, activation = 'softmax')(x)

# create model with two inputs
model_3 = Model([input_1,input_2], output)
# tensorboard = TensorBoard(log_dir='logs/{}'.format(time()))
print(model_3.summary())
```

```
Model: "model_4"
__________________________________________________________________________________________
 Layer (type)                 Output Shape              Param #
Connected to
```

```
==================================================================================================
 input_1 (InputLayer)           [(None, 310)]        0           []


 embedding (Embedding)          (None, 310, 100)     4729700
['input_1[0][0]']

 input_2 (InputLayer)           [(None, 510, 1)]     0           []


 dropout_8 (Dropout)            (None, 310, 100)     0
['embedding[0][0]']

 conv1d (Conv1D)                (None, 508, 128)     512
['input_2[0][0]']

 lstm (LSTM)                    (None, 310, 128)     117248
['dropout_8[0][0]']

 conv1d_1 (Conv1D)              (None, 506, 64)      24640
['conv1d[0][0]']

 flatten_20 (Flatten)           (None, 39680)        0
['lstm[0][0]']

 flatten_21 (Flatten)           (None, 32384)        0
['conv1d_1[0][0]']

 concatenate_4 (Concatenate)    (None, 72064)        0
['flatten_20[0][0]',

'flatten_21[0][0]']

 dense_20 (Dense)               (None, 128)          9224320
['concatenate_4[0][0]']
```

```
 dropout_9 (Dropout)            (None, 128)          0
['dense_20[0][0]']


 dense_21 (Dense)               (None, 64)           8256
['dropout_9[0][0]']


 dropout_10 (Dropout)           (None, 64)           0
['dense_21[0][0]']


 batch_normalization (BatchNorm  (None, 64)          256
['dropout_10[0][0]']
 alization)



 dense_22 (Dense)               (None, 32)           2080
['batch_normalization[0][0]']


 dropout_11 (Dropout)           (None, 32)           0
['dense_22[0][0]']


 dense_23 (Dense)               (None, 2)            66
['dropout_11[0][0]']


==================================================================
============================
Total params: 14,107,078
Trainable params: 9,377,250
Non-trainable params: 4,729,828
__________________________________________________________________
_____________________________
None
```

```
# Clear any logs from previous runs
%load_ext tensorboard
# !rm -rf ./logs/

log_dir = "logs/fit/" + "model_3"
tensorboard_callback =
tensorflow.keras.callbacks.TensorBoard(log_dir=log_dir,
histogram_freq=1)
```

```
The tensorboard extension is already loaded. To reload it, use:
  %reload_ext tensorboard
```

```
model_3.compile(loss='categorical_crossentropy', optimizer=Adam(),
metrics=[aucroc])

#saving best model
# https://machinelearningmastery.com/check-point-deep-learning-models-
keras/

filepath="Model_3_{epoch:02d}_{val_aucroc:.2f}.hdf5"
checkpoint = ModelCheckpoint(filepath, monitor='val_aucroc',
verbose=1, save_best_only=True, mode='max')

#Callbacks List
callbacks_list = [checkpoint,tensorboard_callback]

from sklearn.utils import class_weight

class_weights =
class_weight.compute_class_weight(class_weight='balanced',classes=np.u
nique(y_train[:,0]),y=train_df['project_is_approved'].tolist())
print(class_weights)
class_weights = {0:3.30679754,1:0.5890694}
```

```
[3.30222817 0.58921472]
```

```
model_3.fit(x_train,
y_train,epochs=15,verbose=1,batch_size=512,validation_data=[x_test,y_t
est],validation_batch_size=512,callbacks=callbacks_list,
            class_weight=class_weights,shuffle=True)
```

```
Epoch 1/15
150/150 [==============================] - ETA: 0s - loss: 0.8363 -
aucroc: 0.5358
Epoch 1: val_aucroc improved from -inf to 0.62731, saving model to
Model_3_01_0.63.hdf5
150/150 [==============================] - 34s 144ms/step - loss:
0.8363 - aucroc: 0.5358 - val_loss: 0.7128 - val_aucroc: 0.6273
Epoch 2/15
149/150 [============================>.] - ETA: 0s - loss: 0.6989 -
aucroc: 0.6027
Epoch 2: val_aucroc improved from 0.62731 to 0.68932, saving model to
Model_3_02_0.69.hdf5
150/150 [==============================] - 21s 138ms/step - loss:
0.6994 - aucroc: 0.6031 - val_loss: 0.6458 - val_aucroc: 0.6893
Epoch 3/15
150/150 [==============================] - ETA: 0s - loss: 0.6680 -
aucroc: 0.6495
Epoch 3: val_aucroc improved from 0.68932 to 0.71123, saving model to
Model_3_03_0.71.hdf5
```

```
150/150 [==============================] - 23s 154ms/step - loss:
0.6680 - aucroc: 0.6495 - val_loss: 0.6359 - val_aucroc: 0.7112
Epoch 4/15
150/150 [==============================] - ETA: 0s - loss: 0.6523 -
aucroc: 0.6723
Epoch 4: val_aucroc improved from 0.71123 to 0.71784, saving model to
Model_3_04_0.72.hdf5
150/150 [==============================] - 20s 135ms/step - loss:
0.6523 - aucroc: 0.6723 - val_loss: 0.6380 - val_aucroc: 0.7178
Epoch 5/15
149/150 [============================>.] - ETA: 0s - loss: 0.6399 -
aucroc: 0.6920
Epoch 5: val_aucroc improved from 0.71784 to 0.72833, saving model to
Model_3_05_0.73.hdf5
150/150 [==============================] - 20s 136ms/step - loss:
0.6399 - aucroc: 0.6919 - val_loss: 0.6811 - val_aucroc: 0.7283
Epoch 6/15
149/150 [============================>.] - ETA: 0s - loss: 0.6292 -
aucroc: 0.7083
Epoch 6: val_aucroc improved from 0.72833 to 0.73565, saving model to
Model_3_06_0.74.hdf5
150/150 [==============================] - 20s 136ms/step - loss:
0.6292 - aucroc: 0.7085 - val_loss: 0.6351 - val_aucroc: 0.7357
Epoch 7/15
149/150 [============================>.] - ETA: 0s - loss: 0.6220 -
aucroc: 0.7172
Epoch 7: val_aucroc improved from 0.73565 to 0.74033, saving model to
Model_3_07_0.74.hdf5
150/150 [==============================] - 20s 134ms/step - loss:
0.6219 - aucroc: 0.7175 - val_loss: 0.6254 - val_aucroc: 0.7403
Epoch 8/15
150/150 [==============================] - ETA: 0s - loss: 0.6162 -
aucroc: 0.7256
Epoch 8: val_aucroc improved from 0.74033 to 0.74544, saving model to
Model_3_08_0.75.hdf5
150/150 [==============================] - 20s 134ms/step - loss:
0.6162 - aucroc: 0.7256 - val_loss: 0.7236 - val_aucroc: 0.7454
Epoch 9/15
149/150 [============================>.] - ETA: 0s - loss: 0.6098 -
aucroc: 0.7329
Epoch 9: val_aucroc improved from 0.74544 to 0.74682, saving model to
Model_3_09_0.75.hdf5
150/150 [==============================] - 20s 134ms/step - loss:
0.6101 - aucroc: 0.7324 - val_loss: 0.5704 - val_aucroc: 0.7468
Epoch 10/15
149/150 [============================>.] - ETA: 0s - loss: 0.6008 -
aucroc: 0.7424
Epoch 10: val_aucroc did not improve from 0.74682
150/150 [==============================] - 20s 131ms/step - loss:
0.6011 - aucroc: 0.7421 - val_loss: 0.5937 - val_aucroc: 0.7437
```

```
Epoch 11/15
150/150 [==============================] - ETA: 0s - loss: 0.5973 -
aucroc: 0.7501
Epoch 11: val_aucroc improved from 0.74682 to 0.74914, saving model to
Model_3_11_0.75.hdf5
150/150 [==============================] - 20s 134ms/step - loss:
0.5973 - aucroc: 0.7501 - val_loss: 0.6441 - val_aucroc: 0.7491
Epoch 12/15
149/150 [============================>.] - ETA: 0s - loss: 0.5876 -
aucroc: 0.7599
Epoch 12: val_aucroc did not improve from 0.74914
150/150 [==============================] - 20s 134ms/step - loss:
0.5875 - aucroc: 0.7597 - val_loss: 0.6167 - val_aucroc: 0.7457
Epoch 13/15
149/150 [============================>.] - ETA: 0s - loss: 0.5781 -
aucroc: 0.7704
Epoch 13: val_aucroc did not improve from 0.74914
150/150 [==============================] - 20s 131ms/step - loss:
0.5780 - aucroc: 0.7702 - val_loss: 0.6584 - val_aucroc: 0.7458
Epoch 14/15
150/150 [==============================] - ETA: 0s - loss: 0.5711 -
aucroc: 0.7799
Epoch 14: val_aucroc did not improve from 0.74914
150/150 [==============================] - 20s 132ms/step - loss:
0.5711 - aucroc: 0.7799 - val_loss: 0.8689 - val_aucroc: 0.7467
Epoch 15/15
149/150 [============================>.] - ETA: 0s - loss: 0.5585 -
aucroc: 0.7916
Epoch 15: val_aucroc did not improve from 0.74914
150/150 [==============================] - 20s 131ms/step - loss:
0.5585 - aucroc: 0.7916 - val_loss: 0.6182 - val_aucroc: 0.7431

<keras.callbacks.History at 0x7fe95fcc3290>

#Loading Best_fit model

# https://github.com/keras-team/keras/issues/10104
dependencies = {'aucroc': aucroc}

#loading best model
loaded_model_3 =
load_model('Model_3_11_0.75.hdf5',custom_objects=dependencies)

#checking loaded model
# loaded_model_3.summary()

#predicting train dataset
train_y_pred = loaded_model_3.predict(x_train)
print('Train AUC_ROC_SCORE =',roc_auc_score(y_train,train_y_pred))

Train AUC_ROC_SCORE = 0.7942496320898398
```

```
#predicting test dataset
test_y_pred = loaded_model_3.predict(x_test)
print('Test AUC_ROC_SCORE =',roc_auc_score(y_test,test_y_pred))
```

```
Test AUC_ROC_SCORE = 0.7451621576829476
```

```
#checking tensorboard
%tensorboard --logdir logs/fit
```

```
Reusing TensorBoard on port 6006 (pid 515), started 0:27:32 ago. (Use
'!kill 515' to kill it.)
```

```
<IPython.core.display.Javascript object>
```

Results

```
from prettytable import PrettyTable

table=PrettyTable()
table.field_names = ["Model", "Epochs", "val_accuracy"]
table.add_row(["Model_1", "4", 0.74])
table.add_row(["Model_2","5", 0.66])
table.add_row(["Model_3","15", 0.75])
print(table)
```

```
+---------+--------+--------------+
|  Model  | Epochs | val_accuracy |
+---------+--------+--------------+
| Model_1 |   4    |     0.74     |
| Model_2 |   5    |     0.66     |
| Model_3 |   15   |     0.75     |
+---------+--------+--------------+
```

```
#checking tensorboard
%tensorboard --logdir logs/fit
```

```
Reusing TensorBoard on port 6006 (pid 515), started 0:29:40 ago. (Use
'!kill 515' to kill it.)
```

```
<IPython.core.display.Javascript object>
```