```
%matplotlib inline
import warnings
warnings.filterwarnings("ignore")

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

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

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

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

from tqdm import tqdm
import os

from plotly import plotly
import plotly.offline as offline
import plotly.graph_objs as go
offline.init_notebook_mode()
from collections import Counter
print('done')
```

```
done
```

Reading Data

```
dft = pd.read_csv('train_data.csv',nrows=50000)
```

```
dfr = pd.read_csv('resources.csv')


print("Number of data points in train data", dft.shape)
print('-'*50)
print("The attributes of data :", dft.columns.values)

print(dfr.shape)
print(dfr.columns.values)
```

```
Number of data points in train data (50000, 17)
--------------------------------------------------
The attributes of data : ['Unnamed: 0' 'id' 'teacher_id' 'teacher_prefix' 'school_state'
 'project_submitted_datetime' 'project_grade_category'
 'project_subject_categories' 'project_subject_subcategories'
 'project_title' 'project_essay_1' 'project_essay_2' 'project_essay_3'
 'project_essay_4' 'project_resource_summary'
 'teacher_number_of_previously_posted_projects' 'project_is_approved']
(1541272, 4)
['id' 'description' 'quantity' 'price']
```

```
#sort the datapoints by date  <-

# how to replace elements in list python: https://stackoverflow.com/a/2582163/4084039
cols = ['Date' if x=='project_submitted_datetime' else x for x in list(dft.columns)]


#sort dataframe based on time pandas python: https://stackoverflow.com/a/49702492/4084039
dft['Date'] = pd.to_datetime(dft['project_submitted_datetime'])
dft.drop('project_submitted_datetime', axis=1, inplace=True)# we drop the col
dft.sort_values(by=['Date'], inplace=True)# sort the values y date


# how to reorder columns pandas python: https://stackoverflow.com/a/13148611/4084039
dft = dft[cols]


dft.head(2)
```

	Unnamed: 0	id	teacher_id	teacher_prefix	school_state
473	100660	p234804	cbc0e38f522143b86d372f8b43d4cff3	Mrs.	GA
41558	33679	p137682	06f6e62e17de34fcf81020c77549e1d5	Mrs.	WA

Text preprocessing

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

```
dft.head(2)
```

	Unnamed: 0	id	teacher_id	teacher_prefix	school_state
473	100660	p234804	cbc0e38f522143b86d372f8b43d4cff3	Mrs.	GA
41558	33679	p137682	06f6e62e17de34fcf81020c77549e1d5	Mrs.	WA

```
# https://stackoverflow.com/a/47091490/4084039
import re

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

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


# https://gist.github.com/sebleier/554280
# we are removing the words from the stop words list: 'no', 'nor', 'not'
# we are removing the words from the stop words list: 'no', 'nor', 'not'
stopwords= ['i', 'me', 'my', 'myself', 'we', 'our', 'ours', 'ourselves', 'you', "you're", "yo
            "you'll", "you'd", 'your', 'yours', 'yourself', 'yourselves', 'he', 'him', 'his',
            'she', "she's", 'her', 'hers', 'herself', 'it', "it's", 'its', 'itself', 'they',
            'theirs', 'themselves', 'what', 'which', 'who', 'whom', 'this', 'that', "that'll"
            'am', 'is', 'are', 'was', 'were', 'be', 'been', 'being', 'have', 'has', 'had', 'h
            'did', 'doing', 'a', 'an', 'the', 'and', 'but', 'if', 'or', 'because', 'as', 'unt
            'at', 'by', 'for', 'with', 'about', 'against', 'between', 'into', 'through', 'dur
            'above', 'below', 'to', 'from', 'up', 'down', 'in', 'out', 'on', 'off', 'over', '
            'then', 'once', 'here', 'there', 'when', 'where', 'why', 'how', 'all', 'any', 'bo
            'most', 'other', 'some', 'such', 'only', 'own', 'same', 'so', 'than', 'too', 'ver
            's', 't', 'can', 'will', 'just', 'don', "don't", 'should', "should've", 'now', 'd
            've', 'y', 'ain', 'aren', "aren't", 'couldn', "couldn't", 'didn', "didn't", 'does
            "hadn't", 'hasn', "hasn't", 'haven', "haven't", 'isn', "isn't", 'ma', 'mightn', "
            "mustn't", 'needn', "needn't", 'shan', "shan't", 'shouldn', "shouldn't", 'wasn',
            'won', "won't", 'wouldn', "wouldn't"]
```

Preprocessing of project_subject_categories

```
categories = list(dft['project_subject_categories'].values)
# remove special characters from list of strings python: https://stackoverflow.com/a/47301924

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

dft['clean_categories'] = cat_list
dft.drop(['project_subject_categories'], axis=1, inplace=True)

from collections import Counter
my_counter = Counter()
for word in dft['clean_categories'].values:
    my_counter.update(word.split())

cat_dict = dict(my_counter)
sorted_cat_dict = dict(sorted(cat_dict.items(), key=lambda kv: kv[1]))
```

Preprocessing of project_subject_subcategories

```
sub_catogories = list(dft['project_subject_subcategories'].values)
# remove special characters from list of strings python: https://stackoverflow.com/a/47301924

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

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

dft['clean_subcategories'] = sub_cat_list
dft.drop(['project_subject_subcategories'], axis=1, inplace=True)

# count of all the words in corpus python: https://stackoverflow.com/a/22898595/4084039
my_counter = Counter()
for word in dft['clean_subcategories'].values:
    my_counter.update(word.split())

sub_cat_dict = dict(my_counter)
sorted_sub_cat_dict = dict(sorted(sub_cat_dict.items(), key=lambda kv: kv[1]))
```

Preprocessing of project_grade_category

```
print(dft['project_grade_category'][:20])#   we have to remove the graddes from every row
```

```
473      Grades PreK-2
41558       Grades 3-5
29891       Grades 3-5
23374    Grades PreK-2
49228    Grades PreK-2
7176     Grades PreK-2
35006       Grades 3-5
5145        Grades 3-5
48237      Grades 9-12
46375       Grades 3-5
36468    Grades PreK-2
36358    Grades PreK-2
39438    Grades PreK-2
2521     Grades PreK-2
40180    Grades PreK-2
25460       Grades 6-8
34399       Grades 3-5
5364        Grades 6-8
47478      Grades 9-12
45858       Grades 3-5
Name: project_grade_category, dtype: object
```

```
d= list(dft['project_grade_category'].values)
# remove special characters from list of strings python: https://stackoverflow.com/a/47301924

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

grade_cat_list = []
for i in d:
    # consider we have text like this:
    for j in i.split(' '): #    # split by spae
        j=j.replace('Grades','')# clean grades from the row
    grade_cat_list.append(j.strip())




dft['clean_grade'] = grade_cat_list
dft.drop(['project_grade_category'], axis=1, inplace=True)

# count of all the words in corpus python: https://stackoverflow.com/a/22898595/4084039
my_counter = Counter()
for word in dft['clean_grade'].values:
    my_counter.update(word.split())

project_grade_category_dict= dict(my_counter)
sorted_project_grade_category_dict = dict(sorted(project_grade_category_dict.items(), key=lam
```

Preparing data for the models

Test - Train Split

```
#Splitting Data into train and Test sklearn https://scikit-learn.org/stable/modules/generated
from sklearn.model_selection import train_test_split
X_train, X_test, y_train, y_test = train_test_split(dft,
                                                  dft['project_is_approved'],
                                                    stratify=  dft['project_is_approved'],
                                                    test_size = 0.33
                                                   )


X_train, X_cv, y_train, y_cv = train_test_split(X_train, y_train,  stratify= y_train,
                                                test_size = 0.33)


print(y_train.value_counts())
print(y_test.value_counts())
print(y_cv.value_counts())
# huge imbalance
```

```
1    18982
0     3463
Name: project_is_approved, dtype: int64
1    13954
0     2546
Name: project_is_approved, dtype: int64
1    9350
0    1705
Name: project_is_approved, dtype: int64
```

```
#droping the y labels
#https://stackoverflow.com/questions/13411544/delete-column-from-pandas-dataframe-by-column-n
#x_train =
X_train.drop(["project_is_approved"], axis = 1, inplace = True)
#x_test =
X_test.drop(["project_is_approved"], axis = 1, inplace = True)
#x_cv =
X_cv.drop(["project_is_approved"], axis = 1, inplace = True)
print(X_train)
```

```
       Unnamed: 0       id                        teacher_id teacher_prefix  \
19617      115789  p241258  1de26f587275b51c8113097c48d2191f           Mrs.
44542      155307  p212523  b8678a3f0938b1e1026fb0b7ce9e25bd            Ms.
8792        66079  p239681  62f7c40fbd176a2d9c3d93bb4b9c64ad           Mrs.
25974      172712  p223608  fd6df7d7e9374d98c5e32997f9af7ec6            Ms.
18120       59643  p095656  d23b27efc87b13832df0e2cd1107f768           Mrs.
17351       72489  p158391  c01e73f550c7a0a39fc8e7b074322a33           Mrs.
22925       77948  p124386  2de7f0b5ad7387bfbaf3a08dfd9866eb           Mrs.
19317       61244  p116830  1df3c8ea6a0f45314a5c922a2ce91d86            Ms.
10569        6582  p244766  c8b3f99581f468d0e8635cca0870a675           Mrs.
42779       79687  p258554  58d8ad4c9e0fa7129b8ff8d707aacde8           Mrs.
40456       73497  p220593  ce21478a570b082b251d5a3425091fd4            Ms.
30426       40474  p127767  fb5c8d34582c1307185e9266a3b480bc           Mrs.
35093       67475  p195646  1033f4a5a8fc318f17e08526eb94f1e9            Ms.
38144      120757  p136385  dc4b8bba53041d26a44a4c375b93e682           Mrs.
47871       51251  p164932  27eaaa612ac4b4711555e7e0b220f5a5            Mr.
39284       26685  p109140  ad3eae955a3cee708c82a0a11854d1b2            Ms.
11507      132632  p135642  c1dd0c3744029b837ef6e6f09099fd5d            Ms.
35564      102961  p059022  7213287aab328210611797e1c27bf48e           Mrs.
30320      159769  p066097  f9f8e9c3e599693bb3314ff645f4beec            Ms.
18699        8286  p053340  ea94d0d360113e756195aeea80dd36f9            Ms.
36429      144611  p102392  6c9dd23184c048df50a7388b5d61e8e4            Ms.
38984      124991  p134222  507494cdcac17ac7974cc21ab7b349a3           Mrs.
23309      120220  p042134  862de24454ad34fd8937f9dcb18abdd3           Mrs.
40353      155911  p253123  5ea8a6daa87740fa916d9c98415e9114            Ms.
27925      163527  p143021  42bc7030c016f625d8a0e28f69599752           Mrs.
16118      139696  p131898  6fefa33c27c375ddb39a7fafc5e6df76           Mrs.
216         50938  p027540  f356bc9063c1c1334cee2fcb6d0ddf57            Mr.
36537       88157  p244428  85f308713e285c327047d95e50d1554f            Ms.
8400       145981  p169535  a58e50a713edef56e6a136f421f4fa99            Ms.
39774       25240  p238128  11c9673d9c660e711579d8e6ff07260e            Ms.
...           ...      ...                               ...            ...
12783       39244  p176370  2d9af17605887d1048438175779c324e        Teacher
43739        3194  p033209  b852f9d7f183b4a6a41a6d2a695aa702            Ms.
3995        50128  p209904  4e7e14041d3f3a64e99f6e698ba122ea            Ms.
28114       54792  p069249  9df44cd9b4f57a102ae8f463e53f15a0           Mrs.
23615       56023  p038396  3b950f9d307bcafcfeda4f833f99e8de           Mrs.
45830      137680  p214261  c25c2b33471663158b4a1398dcdfc4f7           Mrs.
14894      103105  p172065  04dcdcf90807262e5cbe3a7a1435ca8b           Mrs.
38994      179539  p183680  4137d18d1b67c25b1ef331c039e7d643           Mrs.
28377      143315  p218796  6d13895525a0070ee4b1da65f32011bc            Ms.
49705       36786  p183090  b25b1a97c0e1bd8f8f0e67405861db20            Ms.
43128      146380  p214834  1a5b07c74d9ce2b245acc4e0af01f477           Mrs.
14958       93536  p151204  dabeb82dcbf80009104fbf085819cc1f            Ms.
22697       25398  p178741  cbe1077b38c3baed9b036e392f48817d            Ms.
36777       81712  p126609  555918822cf6000f32d8866f9ada2b30           Mrs.
7348         1302  p001014  49dbe5521b40c3f5e89ccfc29fb3fbb7           Mrs.
8509        62127  p137484  b4af7caa752f754cfcb3a1f9f1e06fc0           Mrs.
4665       143007  p187559  493c3219121650b2fafcecc27d837dc2            Ms.
34668       60163  p016176  881f55bb99d35f9b86bfc76f25baf3b1           Mrs.
22540       61530  p189297  db7f3336654f2f84cc86c6b14f3903b6            Mr.
27347      162636  p102923  28abf57a96070e068108255626395202            Mr.
32442       86102  p165188  05005648111c30c372cbb3aaa84b6b59           Mrs.
20527       14181  p162053  7ec5df5e932c6a5fee2ee5c294182f95            Ms.
41095      116776  p169026  5f3aded414868e29d90200eaa9251e25            Ms.
15558      175477  p046203  51376478c08a9a60b3e558b4e28b503c            Mr.
11860       81632  p113712  464db35aa7cc3915c2b3a4cf8cc6c61d           Mrs.
```

```
14697       45976  p143810  1837091274bd8a061b7ffd0cc6fb3338           Mr.
44715      132427  p198999  12ddc3da8d9ac763918aca0bfba3d7c0           Ms.
33670       78339  p179229  853c234b876d9bdfe586d7f10f5af870          Mrs.
48247      110601  p091962  5e9a1a690001f6d55919507bf4541399          Mrs.
447        115547  p031282  ecd5f5ea74067b600dac8f5954771ab2          Mrs.

      school_state                Date  \
19617           LA 2017-01-23 11:37:37
44542           MI 2016-08-18 20:25:18
8792            NC 2016-05-18 10:37:56
25974           DC 2016-12-22 23:37:30
18120           CA 2017-03-16 01:09:53
17351           AR 2016-11-29 08:21:15
22925           CA 2017-01-09 00:14:21
19317           VA 2016-12-31 12:33:55
10569           MS 2016-09-01 10:29:00
42779           MA 2016-11-22 13:25:56
40456           LA 2017-01-17 17:46:45
30426           OK 2016-08-19 19:14:37
35093           MN 2016-11-29 08:50:36
38144           FL 2016-10-02 16:41:25
47871           ME 2016-12-07 10:54:48
39284           LA 2016-09-01 03:53:57
11507           NC 2016-11-21 19:57:19
35564           WY 2016-10-07 14:10:29
30320           AR 2016-09-28 09:55:56
18699           IN 2017-03-09 15:27:29
36429           MN 2016-05-04 13:24:28
38984           NY 2016-06-02 23:19:28
23309           TX 2016-10-03 12:48:34
40353           MD 2017-02-08 11:27:27
27925           MA 2016-08-01 01:18:45
16118           NC 2016-08-20 19:39:56
216             CA 2016-07-31 21:46:25
36537           CA 2016-06-01 17:43:43
8400            UT 2016-09-02 20:28:00
39774           IA 2016-09-30 13:28:30
...            ...                 ...
12783           PA 2016-07-08 22:37:59
43739           NY 2017-03-19 22:48:14
3995            MN 2017-02-17 09:28:17
28114           KY 2017-01-17 14:41:35
23615           UT 2016-09-18 22:37:46
45830           SC 2016-11-16 12:11:43
14894           GA 2016-09-28 14:55:43
38994           FL 2016-07-19 19:25:29
28377           MN 2016-08-03 21:56:05
49705           NC 2017-03-30 12:53:16
43128           SC 2016-05-03 17:02:32
14958           VA 2016-08-15 18:58:09
22697           NY 2016-08-01 14:03:22
36777           UT 2016-09-01 17:23:21
7348            NJ 2017-03-28 18:40:54
8509            VA 2016-10-13 15:19:40
4665            HI 2016-08-31 17:07:55
34668           MO 2016-05-27 15:48:24
22540           CA 2017-01-14 17:37:48
27347           TN 2016-09-25 20:52:45
```

```
32442           IL 2016-08-08 08:18:01
20527           NY 2017-02-10 15:27:23
41095           LA 2016-10-05 23:53:14
15558           TX 2016-07-20 03:01:32
11860           FL 2016-08-18 19:12:15
14697           FL 2017-03-29 09:10:41
44715           CA 2016-05-10 21:54:37
33670           NC 2016-08-17 22:59:34
48247           CO 2016-08-16 13:48:53
447             PA 2016-08-02 18:14:56

                                             project_title  \
19617         Building Young Minds: STEM BUNDLE In PRE-K
44542                                Books For 5th Graders
8792         Beautiful, comfortable and educational rug!
25974   It's Time for Graffiti Art with 3Doodler! Part 2
18120                  Flexible Seating For Flexible Minds
17351                                DASHing Into Coding
22925                                 Can't Stop Moving!
19317         Stick it to Math the Magnetic          Way
10569                                   Chrome for class
42779           Music!  Demystify the Magic Behind Sound
40456                           Chromebooks for Chrome-kids
30426   HELP US SET OUR BRAIN TO WORK IN TWO LANGUAGES!!
35093                                Classroom Chromebooks!
38144                 IPads, Ebooks and Accelerated Reader
47871                                Keep Things Organzied
39284                           Growing With Music Movement
11507  Engaging Families at Home and School in Their ...
35564                                 Building Our Future!
30320                      Something Wonderful Has Sprouted
18699  Re-Imagining the Past...Classic Myths as Graph...
36429                           Teaching Young Minds in 2016
38984  Our kids want to STAND and LEARN in the classr...
23309                      Taking the Lexile Hop To The Top!
40353                           The Interscholastic Athlete!
27925                         Increasing App-titudes is Fun!
16118    Organization Leads to Successful First Graders!
216             SAVVY STEM START-UP USING ROBUST ROBOTICS
36537                             Space and Place for STEM
8400   Happy, Wiggly First Graders in Search of Rug a...
39774                      Chromebooks for Technology Class
...                                                    ...
12783  Making Math Accessible: The Supplies that Seco...
43739  The Heart and Soul of Our Classroom Needs an U...
3995                             Girl Runners! Girl Power!
28114                  A Nap a Day Keeps the Doctor Away!
23615                                  Education in Motion
45830      Technology in Kindergarten??? ... ABSOLUTELY!
14894               My Students Are on Fire! Kindle Fire!
38994                             Back(pack) to School II.
28377                           Where does food come from?
49705             Flexible Seating For My Energetic Class
43128             Deskless Classroom and Flexible Seating
14958  How Can We Play Volleyball Without A Net Or Sh...
22697                                   Technology is KEY!
36777                                 Sensory for Success!
7348             Play-doh in PreK: Help us Mold Our Future
```

```
7348          Play doh in PreK: Help us Mold Our Future
8509   Silly second graders wobble, but they don't fa...
4665                                       Taffy Techies
34668                   Supporting a Classroom Community
22540         Leveraging Learning with a Set of Laptops!
27347                                   Gym Class Heroes
32442          Flexible Seating for Kindergarten Wiggles
20527                                        Book Club !
41095                 Today a Reader, Tomorrow a Leader!
15558           Capturing Our Future, Changing The Past!
11860              Technology and Books for 1st Graders!
14697                             To Shot or Not to Shot
44715                               Spanish for Everyone
33670  Starting The Year Off Right With Data Tracking...
48247                  Wiggles and Jiggles Help Us Learn
447                         We Need to Move It, Move It!

                                         project_essay_1  \
19617  I am a preschool teacher at an elementary scho...
44542  We are a school that is welcoming and invites ...
8792   A typical day in our classroom consists of lot...
25974  Save our art class! We need your help to conti...
18120  As a teacher in a low-income/high poverty scho...
17351  My sixth grade students work hard, knowing tha...
22925  You may think that you know what goes on in ou...
19317  I work at a school where the children are grow...
10569  I have a typical seventh grade class. My stude...
42779  All my students do is want to learn.  That's i...
40456  We are fantastic Scientists and Mathematicians...
30426  My students have been enrolled in the Dual Lan...
35093  My second grade class is filled with students ...
38144  I teach at a Title 1 school. Many of them rece...
47871  My students are energetic, eager, and enthusia...
39284  My pre-kindergarten students would be best des...
11507  As a teacher in a Title I school, many of my s...
35564  I teach in a Title I School where the majority...
30320  Our school's upper elementary is comprised of ...
18699  As a teacher in a low-income school, I am alwa...
36429  Talk. Talk. Talk.. this is how students are le...
38984  We are a Title 1 school in the South Bronx. I ...
23309  I teach at a Title I school where most of our ...
40353  The students at Samuel Ogle Middle School are ...
27925  Located in the heart of Chinatown, our school ...
16118  Our classroom will soon be filled with a diver...
216    After walking my students to their junior high...
36537  My classroom community is made up of a diverse...
8400   I teach at a Title One school in a small town ...
39774  As the Technology and Project Lead the Way tea...
...                                                  ...
12783  As an incoming first year teacher, I haven't b...
43739  The school is located in an urban community, w...
3995   Minneapolis 3rd, 4th, and 5th graders are gett...
28114  Hunter Hills Elementary has the highest povert...
23615  My students are a group of top notch students....
45830  My classroom is filled with happy, diverse stu...
14894  My students are brave, creative, and intellige...
38994  Students at my school are awesome. There are a...
28377  My students are five year olds!  Most of my st...
```

```
49705  My students come from a variety of backgrounds...
43128  Where did you like to do your homework when yo...
14958  The Lady Vikings Volleyball team has never bee...
22697  I teach in a classroom where the students are ...
36777  As a teacher of students with disabilities, my...
7348   My students need the opportunity to fulfill th...
8509   My students are eager learners from diverse ba...
4665   My 5th graders can be quite a talkative class!...
34668  My students have become my family. This is my ...
22540  My students come from multicultural background...
27347  As a traveling physical education teacher in a...
32442  \"Tell me and I forgot. Teach me and I remembe...
20527  Our students are full of wonder about what the...
41095  If you walk into my classroom, you would be gr...
15558  I teach at a Junior High where 69.5% of the st...
11860  There can be infinite uses of the computer and...
14697  My students are Television Production vocation...
44715  My students need to feel incredible about thei...
33670  I teach at a low performing school where most ...
48247  Our wonderful school is nestled in the downtow...
447    My students are a great group of kids from a r...

                                          project_essay_2  \
19617  The stem bundle will allow my little preschool...
44542  The books that I am requesting for my classroo...
8792   My student spend so much time learning while s...
25974  My students love graffiti art, symbols, and em...
18120  My students are in great need of flexible seat...
17351  A Dash coding robot will make a difference in ...
22925  My students are always on the go.  A lot of th...
19317  With your help, I'll be able to easily reach m...
10569  Teachers and students will use these mobile de...
42779  My students will be exploring and experimentin...
40456  We all know that this generation of students t...
30426  The Dual Language program has one unique objec...
35093  My students will use these chromebooks during ...
38144  Thank you for your interest in my project.  Th...
47871  These various storage materials will be used t...
39284  Circle Time is an important and dynamic part o...
11507  Partnering with my parents is very important t...
35564  I teach in a Title I School where the majority...
30320  The lighted plant stand that I am requesting w...
18699  Every year, my students and I look forward to ...
36429  I teach active third graders in a northern sub...
38984  A lot of research supports giving students man...
23309  My 1st grade students are like little sponges,...
40353  When students participate in after-school acti...
27925  My students were born in the 21st century.  Th...
16118  How do you feel about organization? Imagine yo...
216    Presidential elections, no.  World Series game...
36537  When students are faced with a problem or task...
8400   Currently, my classroom doesn't have a big eno...
39774  Technology class is in great need of a few Chr...
...                                                 ...
12783  I want my math classroom to build true mathema...
43739  Our 4th grade classroom becomes a home away fr...
3995   Girl Runners! Girl POWER!\r\n\r\nMy students n...
28114  In preschool, students are between the ages of...
```

```
23615  Students are children, and should be given the...
45830  These tablets will be utilized each day within...
14894  Students in my class aspire to great things, b...
38994  Scores of students come to our school with sub...
28377  Where does your food come from?  \"Target!\"  ...
49705  My students tend to wander around the room whe...
43128  My group of students are first graders who lov...
14958  At our school we know in our hearts that SPORT...
22697  Throughout teaching, a copy machine is require...
36777  These products will enable my students to gain...
7348   My students love to work with play doh. It has...
8509   My students are eager to please and eager to l...
4665   I strongly believe that having an iPad Air in ...
34668  This project will help build on our classroom ...
22540  All over the country, more and more classrooms...
27347  In this first year of a physical education cla...
32442  We would love to be able to have more flexible...
20527  My students need these different books to join...
41095  As I mentioned, I am striving to provide a pos...
15558  Technology is a powerful tool in motivating yo...
11860  Currently we have 3 other Ipad Mini's in our c...
14697  It is difficult for school systems to stay in ...
44715  In my beautiful school, you will see students ...
33670  I have been researching ways that I can motiva...
48247  Students need the opportunity to move around w...
447    My students need opportunities to move! A few ...

                                          project_essay_3  \
19617                                                 NaN
44542                                                 NaN
8792                                                  NaN
25974                                                 NaN
18120                                                 NaN
17351                                                 NaN
22925                                                 NaN
19317                                                 NaN
10569                                                 NaN
42779                                                 NaN
40456                                                 NaN
30426                                                 NaN
35093                                                 NaN
38144                                                 NaN
47871                                                 NaN
39284                                                 NaN
11507                                                 NaN
35564                                                 NaN
30320                                                 NaN
18699                                                 NaN
36429  I am trying to design a 21st century classroom...
38984                                                 NaN
23309                                                 NaN
40353                                                 NaN
27925                                                 NaN
16118                                                 NaN
216                                                   NaN
36537                                                 NaN
8400                                                  NaN
```

```
39774                                                  NaN
...                                                    ...
12783                                                  NaN
43739                                                  NaN
3995                                                   NaN
28114                                                  NaN
23615                                                  NaN
45830                                                  NaN
14894                                                  NaN
38994                                                  NaN
28377                                                  NaN
49705                                                  NaN
43128  I am on the journey to making our classroom co...
14958                                                  NaN
22697                                                  NaN
36777                                                  NaN
7348                                                   NaN
8509                                                   NaN
4665                                                   NaN
34668                                                  NaN
22540                                                  NaN
27347                                                  NaN
32442                                                  NaN
20527                                                  NaN
41095                                                  NaN
15558                                                  NaN
11860                                                  NaN
14697                                                  NaN
44715  Spanish is everywhere, whether your are at the...
33670                                                  NaN
48247                                                  NaN
447                                                    NaN

                                           project_essay_4  \
19617                                                  NaN
44542                                                  NaN
8792                                                   NaN
25974                                                  NaN
18120                                                  NaN
17351                                                  NaN
22925                                                  NaN
19317                                                  NaN
10569                                                  NaN
42779                                                  NaN
40456                                                  NaN
30426                                                  NaN
35093                                                  NaN
38144                                                  NaN
47871                                                  NaN
39284                                                  NaN
11507                                                  NaN
35564                                                  NaN
30320                                                  NaN
18699                                                  NaN
36429  I believe that allowing children to move while...
38984                                                  NaN
23309                                                  NaN
40353                                                  NaN
```

```
27925                                                NaN
16118                                                NaN
216                                                  NaN
36537                                                NaN
8400                                                 NaN
39774                                                NaN
...                                                  ...
12783                                                NaN
43739                                                NaN
3995                                                 NaN
28114                                                NaN
23615                                                NaN
45830                                                NaN
14894                                                NaN
38994                                                NaN
28377                                                NaN
49705                                                NaN
43128  The deskless classroom will solve many issues ...
14958                                                NaN
22697                                                NaN
36777                                                NaN
7348                                                 NaN
8509                                                 NaN
4665                                                 NaN
34668                                                NaN
22540                                                NaN
27347                                                NaN
32442                                                NaN
20527                                                NaN
41095                                                NaN
15558                                                NaN
11860                                                NaN
14697                                                NaN
44715  My classroom will be a more pleasant place to ...
33670                                                NaN
48247                                                NaN
447                                                  NaN

                                project_resource_summary  \
19617  My students need the stem bundle to allow the ...
44542  My students need lots of books to help them ha...
8792   My students need a rug to sit on because we sp...
25974  My students need 3Doodler create education kit...
18120  My students need flexible seating to allow the...
17351  My students need a DASH coding robot for our S...
22925  My students need Bouncy Bands for Chairs to ke...
19317  My students need the magnetic materials and ma...
10569  My students need access to more technology. Th...
42779  My students need books about music and sound t...
40456  My students need Chromebooks to access technol...
30426  My students need a great amount of Spanish res...
35093  My students need chromebooks to provide them w...
38144  My students need an ipad to have access to ebo...
47871  My students need various types of storage cont...
39284  My students need an iPod nano and classroom CD...
11507  My students need books in their native languag...
35564  My students need HANDS ON Building Supplies to...
30320  My students need a lighted plant cart for grow...
```

```
18699  My students need a set of graphic novels that ...
36429  My students need an elementary classroom couch...
38984  My students need to have an exciting alternati...
23309  My students need a Leveled Books Classroom Lib...
40353  My students need a fast pitch softball bat.  T...
27925  My students need iPads to allow us to use educ...
16118  My students need individual seat sacks to hold...
216    My students need a VEX IQ superkit to begin th...
36537  My students need a STEM bundle appropriate for...
8400   My students need a classroom rug to sit on, an...
39774  My students need a set of 5 chromebooks that d...
...                                                 ...
12783  My students need effective, hands-on materials...
43739  My students need 4 bookshelves to create a new...
3995   My students need sf 20 yoga mats to prepare fo...
28114  My students need 20 sheets and blankets for th...
23615  My students need an option to move while learn...
45830  My students need 2 iPad minis and 2 otterboxes...
14894  My students need 3 Kindle Fire tablets so they...
38994  My students need durable backpacks to begin th...
28377  My students need the Miracle-Gro AeroGardens s...
49705  My students need flexible seating so that they...
43128  My students need the four stools to go at a sm...
14958  My students need the necessary equipment to pl...
22697  My students need these printing materials beca...
36777  My students need sensory supplies in order to ...
7348   My students need play doh and foam dough to cr...
8509   My students need stools that will allow them t...
4665   My students need an iPad Air to use in the cla...
34668  My students need a classroom where they can le...
22540  My students need access to technology so that ...
27347  My students need physical activity equipment t...
32442  My students need to have more flexible seating...
20527  My students need 5 copies of each book to star...
41095  My students need clipboards, dry erase dots, w...
15558  My students need a new Canon 70D DSLR camera t...
11860  My students need an Ipad Mini 2 with Otterbox ...
14697  My students need two DSLR cameras in order to ...
44715  My students need Spanish books.\r\nMy students...
33670  My students need one inch binders to track the...
48247  My students need the opportunity to move their...
447    My students need 8 Hokki stools to allow for c...

       teacher_number_of_previously_posted_projects  \
19617                                             3
44542                                             0
8792                                              3
25974                                           128
18120                                             0
17351                                             3
22925                                             2
19317                                             1
10569                                             1
42779                                            18
40456                                             2
30426                                             0
35093                                             0
```

```
38144                                               1
47871                                              18
39284                                               3
11507                                               2
35564                                               0
30320                                               1
18699                                               2
36429                                               1
38984                                               0
23309                                              34
40353                                               3
27925                                              17
16118                                               0
216                                                 7
36537                                               8
8400                                                1
39774                                               0
...                                               ...
12783                                               0
43739                                              94
3995                                                1
28114                                               1
23615                                               1
45830                                               3
14894                                              14
38994                                               1
28377                                               8
49705                                               0
43128                                               0
14958                                               5
22697                                               0
36777                                               4
7348                                               75
8509                                               18
4665                                                1
34668                                               1
22540                                               2
27347                                               0
32442                                               1
20527                                               1
41095                                               0
15558                                               0
11860                                               5
14697                                               1
44715                                               0
33670                                               0
48247                                               0
447                                                25

                                                   essay  \
19617  I am a preschool teacher at an elementary scho...
44542  We are a school that is welcoming and invites ...
8792   A typical day in our classroom consists of lot...
25974  Save our art class! We need your help to conti...
18120  As a teacher in a low-income/high poverty scho...
17351  My sixth grade students work hard, knowing tha...
22925  You may think that you know what goes on in ou...
19317  I work at a school where the children are grow...
```

```
        10569  I have a typical seventh grade class. My stude...
        42779  All my students do is want to learn.  That's i...
        40456  We are fantastic Scientists and Mathematicians...
        30426  My students have been enrolled in the Dual Lan...
        35093  My second grade class is filled with students ...
        38144  I teach at a Title 1 school. Many of them rece...
        47871  My students are energetic, eager, and enthusia...
        39284  My pre-kindergarten students would be best des...
        11507  As a teacher in a Title I school, many of my s...
        35564  I teach in a Title I School where the majority...
        30320  Our school's upper elementary is comprised of ...
        18699  As a teacher in a low-income school, I am alwa...
        36429  Talk. Talk. Talk.. this is how students are le...
        38984  We are a Title 1 school in the South Bronx. I ...
        23309  I teach at a Title I school where most of our ...
        40353  The students at Samuel Ogle Middle School are ...
        27925  Located in the heart of Chinatown, our school ...
        16118  Our classroom will soon be filled with a diver...
        216    After walking my students to their junior high...
        36537  My classroom community is made up of a diverse...
        8400   I teach at a Title One school in a small town ...
        39774  As the Technology and Project Lead the Way tea...
        ...                                                 ...
        12783  As an incoming first year teacher, I haven't b...
        43739  The school is located in an urban community, w...
        3995   Minneapolis 3rd, 4th, and 5th graders are gett...
        28114  Hunter Hills Elementary has the highest povert...
        23615  My students are a group of top notch students....
        45830  My classroom is filled with happy, diverse stu...
        14894  My students are brave, creative, and intellige...
        38994  Students at my school are awesome. There are a...
        28377  My students are five year olds!  Most of my st...
        49705  My students come from a variety of backgrounds...
        43128  Where did you like to do your homework when yo...
        14958  The Lady Vikings Volleyball team has never bee...
        22697  I teach in a classroom where the students are ...
        36777  As a teacher of students with disabilities, my...
        7348   My students need the opportunity to fulfill th...
        8509   My students are eager learners from diverse ba...
        4665   My 5th graders can be quite a talkative class!...
        34668  My students have become my family. This is my ...
        22540  My students come from multicultural background...
        27347  As a traveling physical education teacher in a...
        32442  \"Tell me and I forgot. Teach me and I remembe...
        20527  Our students are full of wonder about what the...
        41095  If you walk into my classroom, you would be gr...
        15558  I teach at a Junior High where 69.5% of the st...
        11860  There can be infinite uses of the computer and...
        14697  My students are Television Production vocation...
        44715  My students need to feel incredible about thei...
        33670  I teach at a low performing school where most ...
        48247  Our wonderful school is nestled in the downtow...
        447    My students are a great group of kids from a r...

                                clean_categories                  clean_subcategories  \
        19617       Math_Science AppliedLearning    AppliedSciences EarlyDevelopment
        44542                  Literacy_Language                             Literacy
        8792       Literacy_Language Math_Science      Literature_Writing Mathematics
```

```
25974          Math_Science Music_Arts          AppliedSciences VisualArts
18120                     Health_Sports                     Health_Wellness
17351                      Math_Science                     AppliedSciences
22925        Health_Sports SpecialNeeds         Health_Wellness SpecialNeeds
19317                      Math_Science                         Mathematics
10569                      Math_Science                   Health_LifeScience
42779          Math_Science Music_Arts                AppliedSciences Music
40456                      Math_Science    EnvironmentalScience Mathematics
30426                 Literacy_Language           ForeignLanguages Literacy
35093     Literacy_Language Math_Science      Literature_Writing Mathematics
38144     Literacy_Language Math_Science                Literacy Mathematics
47871                      Math_Science                     AppliedSciences
39284       Literacy_Language Music_Arts                      Literacy Music
11507  Literacy_Language AppliedLearning         Literacy ParentInvolvement
35564                      Math_Science         AppliedSciences Mathematics
30320                      Math_Science                EnvironmentalScience
18699                 Literacy_Language                            Literacy
36429     Literacy_Language Math_Science                Literacy Mathematics
38984         Math_Science SpecialNeeds            Mathematics SpecialNeeds
23309                 Literacy_Language                            Literacy
40353                     Health_Sports                          TeamSports
27925     Literacy_Language Math_Science                Literacy Mathematics
16118     Literacy_Language Math_Science                Literacy Mathematics
216                        Math_Science                     AppliedSciences
36537                      Math_Science                     AppliedSciences
8400     Health_Sports Literacy_Language            Health_Wellness Literacy
39774                      Math_Science                     AppliedSciences
...                                 ...                                 ...
12783                      Math_Science                         Mathematics
43739                 Literacy_Language                            Literacy
3995                      Health_Sports          Health_Wellness TeamSports
28114                   AppliedLearning                    EarlyDevelopment
23615    Health_Sports Literacy_Language            Health_Wellness Literacy
45830     Literacy_Language Math_Science                Literacy Mathematics
14894                 Literacy_Language                  Literature_Writing
38994                   AppliedLearning                               Other
28377                     Health_Sports  Health_Wellness NutritionEducation
49705                      Math_Science                     AppliedSciences
43128  AppliedLearning Literacy_Language           EarlyDevelopment Literacy
14958                     Health_Sports                          TeamSports
22697     Literacy_Language Math_Science      Literature_Writing Mathematics
36777      AppliedLearning SpecialNeeds                  Other SpecialNeeds
7348         AppliedLearning Music_Arts         EarlyDevelopment VisualArts
8509         Health_Sports SpecialNeeds         Health_Wellness SpecialNeeds
4665                    AppliedLearning                  CharacterEducation
34668                   AppliedLearning                  CharacterEducation
22540  Literacy_Language AppliedLearning                      Literacy Other
27347                     Health_Sports          Gym_Fitness Health_Wellness
32442                     Health_Sports                     Health_Wellness
20527                 Literacy_Language                            Literacy
41095                 Literacy_Language                            Literacy
15558        AppliedLearning Music_Arts        College_CareerPrep VisualArts
11860                 Literacy_Language                            Literacy
14697                        Music_Arts                          VisualArts
44715                 Literacy_Language                    ForeignLanguages
33670     Literacy_Language Math_Science      Literature_Writing Mathematics
48247     Literacy_Language Math_Science                Literacy Mathematics
```

```
447                           Health_Sports                     Health_Wellness

      clean_grade
19617      PreK-2
44542         3-5
8792       PreK-2
25974         3-5
18120         3-5
17351         6-8
22925      PreK-2
19317         3-5
10569         6-8
42779      PreK-2
40456         3-5
30426      PreK-2
35093      PreK-2
38144      PreK-2
47871         6-8
39284      PreK-2
11507      PreK-2
35564      PreK-2
30320         3-5
18699         3-5
36429         3-5
38984         3-5
23309      PreK-2
40353         6-8
27925      PreK-2
16118      PreK-2
216           6-8
36537      PreK-2
8400       PreK-2
39774         6-8
...           ...
12783      PreK-2
43739         3-5
3995          3-5
28114      PreK-2
23615         3-5
45830      PreK-2
14894         6-8
38994         6-8
28377      PreK-2
49705        9-12
43128      PreK-2
14958         6-8
22697         3-5
36777        9-12
7348       PreK-2
8509       PreK-2
4665          3-5
34668         3-5
22540         3-5
27347      PreK-2
32442      PreK-2
20527      PreK-2
41095      PreK-2
15558         6-8
```

```
11860      PreK-2
14697        9-12
44715         6-8
33670         3-5
48247         3-5
447           3-5

[22445 rows x 17 columns]
```

Text preprocessing

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

```
100%|██████████████████████████████████████████████████████| 22445/2
```

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

```
100%|██████████████████████████████████████████████████████| 16500/1
```

```
#Proprocessing for essay
# Combining all the above stundents
from tqdm import tqdm
```

```
preprocessed_essays_cv = []
# tqdm is for printing the status bar
for sentance in tqdm(X_cv['essay'].values):
  sent = decontracted(sentance)
  sent = sent.replace('\\r', ' ')
  sent = sent.replace('\\"', ' ')
  sent = sent.replace('\\n', ' ')
  sent = re.sub('[^A-Za-z0-9]+', ' ', sent)
   # https://gist.github.com/sebleier/554280
  sent = ' '.join(e for e in sent.split() if e.lower() not in stopwords)
  preprocessed_essays_cv.append(sent.lower().strip())
```

```
100%|██████████████████████████████████████████████████| 11055/1
```

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

```
100%|█████████████████████████████████████████████████| 11055/11
```

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

```
100%|█████████████████████████████████████████████████| 22445/22
```

```
#Proprocessing for essay
# Combining all the above stundents
from tqdm import tqdm
```

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

```
100%|████████████████████████████████████████████████████████████| 16500/16
```

Encoding numerical, Categorical features

vectorize categorical data

```
#projectsubjectcategories convert categorical to vectors
# convert train,cv and test data of clean_categories into vectors
# we use count vectorizer to convert the values into one
from sklearn.feature_extraction.text import CountVectorizer
vectorizer1 = CountVectorizer(vocabulary=list(sorted_cat_dict.keys()), lowercase=False, binar
vectorizer1.fit(X_train['clean_categories'].values)

# firstly convert fit the train data into the vectoriaer then it learn hte vocablery

# we use the fitted CountVectorizer to convert the text to vector
X_train_cat = vectorizer1.transform(X_train['clean_categories'].values)
X_cv_cat = vectorizer1.transform(X_cv['clean_categories'].values)
X_test_cat = vectorizer1.transform(X_test['clean_categories'].values)

print(vectorizer1.get_feature_names())
```

```
['Warmth', 'Care_Hunger', 'History_Civics', 'Music_Arts', 'AppliedLearning', 'SpecialNee
```

```
print("After vectorizations")
print(X_train_cat.shape, y_train.shape)
print(X_cv_cat.shape, y_cv.shape)
print(X_test_cat.shape, y_test.shape)
print("="*100)
```

```
After vectorizations
(22445, 9) (22445,)
(11055, 9) (11055,)
(16500, 9) (16500,)
==========================================================================================
```

2.*project*_subject_subcategories convert categorical to vectors*

```
#projectsubject subcategories convert categorical to vectors
# convert train,cv and test data of clean_categories into vectors
# we use count vectorizer to convert the values into one
from sklearn.feature_extraction.text import CountVectorizer
vectorizer2 = CountVectorizer(vocabulary=list(sorted_sub_cat_dict.keys()), lowercase=False, b
vectorizer2.fit(X_train['clean_subcategories'].values)

# firstly convert fit the train data into the vectoriaer then it learn hte vocablery

# we use the fitted CountVectorizer to convert the text to vector
X_train_subcat = vectorizer2.transform(X_train['clean_subcategories'].values)
X_cv_subcat = vectorizer2.transform(X_cv['clean_subcategories'].values)
X_test_subcat = vectorizer2.transform(X_test['clean_subcategories'].values)

print(vectorizer2.get_feature_names())
```

```
['Economics', 'CommunityService', 'FinancialLiteracy', 'ParentInvolvement', 'Extracurric
```

```
print("After vectorizations")
print(X_train_subcat.shape, y_train.shape)
print(X_cv_subcat.shape, y_cv.shape)
print(X_test_subcat.shape, y_test.shape)
print("="*100)
```

```
After vectorizations
(22445, 30) (22445,)
(11055, 30) (11055,)
(16500, 30) (16500,)
==========================================================================================
```

*3 school_state convert categorical to vectors**

```
#school_state convert categorical to vectors
from collections import Counter
my_counter = Counter()
for word in dft['school_state'].values:
    my_counter.update(word.split())# count the words

school_state_dict = dict(my_counter)# store in dicionary
```

```
sorted_school_state_dict = dict(sorted(school_state_dict.items(), key=lambda kv: kv[1]))# sor
print(sorted_school_state_dict)
```

```
{'VT': 32, 'WY': 51, 'ND': 63, 'MT': 106, 'RI': 126, 'NH': 141, 'SD': 142, 'NE': 144, 'A
```

```
# convert train,cv and test data of clean_categories into vectors
# we use count vectorizer to convert the values into one
from sklearn.feature_extraction.text import CountVectorizer
vectorizer3 = CountVectorizer(vocabulary=list(sorted_school_state_dict.keys()), lowercase=Fal
vectorizer3.fit(dft['school_state'].values)

# firstly convert fit the train data into the vectoriaer then it learn hte vocabulary

# we use the fitted CountVectorizer to convert the text to vector
X_train_school_state = vectorizer3.transform(X_train['school_state'].values)
X_cv_school_state = vectorizer3.transform(X_cv['school_state'].values)
X_test_school_state = vectorizer3.transform(X_test['school_state'].values)

print(vectorizer3.get_feature_names())
```

```
['VT', 'WY', 'ND', 'MT', 'RI', 'NH', 'SD', 'NE', 'AK', 'DE', 'WV', 'ME', 'NM', 'HI', 'DC
```

```
print("After vectorizations")
print(X_train_school_state .shape, y_train.shape)
print(X_cv_school_state .shape, y_cv.shape)
print(X_test_school_state .shape, y_test.shape)
print("="*100)
```

```
After vectorizations
(22445, 51) (22445,)
(11055, 51) (11055,)
(16500, 51) (16500,)
====================================================================================
```

```
#project_grade_category categorical to vectors
#https://stackoverflow.com/questions/42224700/attributeerror-float-object-has-no-attribute-sp
dft['clean_grade']=dft['clean_grade'].fillna("")# fill the nulll values with space

# convert train,cv and test data of clean_categories into vectors


# we use count vectorizer to convert the values into one
from sklearn.feature_extraction.text import CountVectorizer
vectorizer4 = CountVectorizer(vocabulary=list(sorted_project_grade_category_dict.keys()), low
vectorizer4.fit(dft['clean_grade'].values)

# firstly convert fit the train data into the vectoriaer then it learn hte vocablery

# we use the fitted CountVectorizer to convert the text to vector
```

```
X_train_project_grade_category = vectorizer4.transform(X_train['clean_grade'].values)
X_cv_project_grade_category = vectorizer4.transform(X_cv['clean_grade'].values)
X_test_project_grade_category = vectorizer4.transform(X_test['clean_grade'].values)

print(vectorizer4.get_feature_names())
```

```
['9-12', '6-8', '3-5', 'PreK-2']
```

```
print("After vectorizations")
print(X_train_project_grade_category  .shape, y_train.shape)
print(X_cv_project_grade_category  .shape, y_cv.shape)
print(X_test_project_grade_category  .shape, y_test.shape)
print("="*100)
```

```
After vectorizations
(22445, 4) (22445,)
(11055, 4) (11055,)
(16500, 4) (16500,)
===================================================================================
```

```
#teacher_prefix categorical to vectors
##https://stackoverflow.com/questions/42224700/attributeerror-float-object-has-no-attribute-s
dft['teacher_prefix']=dft['teacher_prefix'].fillna(" ")# filll the null values with space
my_counter = Counter()
for word in dft['teacher_prefix'].values:
    my_counter.update(word.split())


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


# convert train,cv and test data of clean_categories into vectors
# we use count vectorizer to convert the values into one
from sklearn.feature_extraction.text import CountVectorizer
vectorizer5 = CountVectorizer(vocabulary=list(sorted_teacher_prefix_dict.keys()), lowercase=F
vectorizer5.fit(dft['teacher_prefix'].values.astype('U'))

# firstly convert fit the train data into the vectoriaer then it learn hte vocablery

# we use the fitted CountVectorizer to convert the text to vector
X_train_teacher_prefix = vectorizer5.transform(X_train['teacher_prefix'].values.astype('U'))
X_cv_teacher_prefix= vectorizer5.transform(X_cv['teacher_prefix'].values.astype('U'))
X_test_teacher_prefix = vectorizer5.transform(X_test['teacher_prefix'].values.astype('U'))

print(vectorizer5.get_feature_names())


# when i executeed this error comes
#np.nan is an invalid document, expected byte or unicode string.
```

```
# then iconvert to unicode just write .astype('U') after the .values in fit and transform
#https://stackoverflow.com/questions/39303912/tfidfvectorizer-in-scikit-learn-valueerror-np-n
```

```
['Dr.', 'Teacher', 'Mr.', 'Ms.', 'Mrs.']
```

```
print("After vectorizations")
print(X_train_teacher_prefix.shape, y_train.shape)
print(X_cv_teacher_prefix.shape, y_cv.shape)
print(X_test_teacher_prefix.shape, y_test.shape)
print("="*100)
```

```
After vectorizations
(22445, 5) (22445,)
(11055, 5) (11055,)
(16500, 5) (16500,)
====================================================================================================
```

Encoding essay, and Project_title

```
#Bow featurezation essay
X_train_essay=preprocessed_essays_train
X_cv_essay=preprocessed_essays_cv
X_test_essay=preprocessed_essays_test


X_train_title=preprocessed_titles_train
X_cv_title=preprocessed_titles_cv
X_test_title=preprocessed_titles_test

# We are considering only the words which appeared in at least 10 documents(rows or projects)
vectorizer6 = CountVectorizer(min_df=10,max_features=5000,ngram_range=(1, 2))# its a countvec
vectorizer6.fit(X_train_essay)# that is learned from trainned  data



# we use the fitted CountVectorizer to convert the text to vector
X_train_bow = vectorizer6.transform(X_train_essay)
X_cv_bow = vectorizer6.transform(X_cv_essay)
X_test_bow = vectorizer6.transform(X_test_essay)



print("After vectorizations")
print(X_train_bow.shape, y_train.shape)
print(X_cv_bow.shape, y_cv.shape)
print(X_test_bow.shape, y_test.shape)
print("="*100)
# so the dimension of alll are the same by using first fit and then transform
print(vectorizer6.get_feature_names())
```

```
After vectorizations
(22445, 5000) (22445,)
(11055, 5000) (11055,)
(16500, 5000) (16500,)
====================================================================================================
['000', '10', '100', '100 free', '100 percent', '100 students', '11', '12', '12th', '13'
```

```
#bow featurization title
vectorizer7 = CountVectorizer(min_df=10,max_features=5000,ngram_range=(1, 2))
vectorizer7.fit(X_train_title)# that is learned from trainned  data



# we use the fitted CountVectorizer to convert the text to vector
X_train_bow_title = vectorizer7.transform(X_train_title)
X_cv_bow_title= vectorizer7.transform(X_cv_title)
X_test_bow_title = vectorizer7.transform(X_test_title)



print("After vectorizations")
print(X_train_bow_title.shape, y_train.shape)
print(X_cv_bow_title.shape, y_cv.shape)
print(X_test_bow_title.shape, y_test.shape)
print("="*100)
# so the dimension of alll are the same by using first fit and then transform
```

```
After vectorizations
(22445, 1576) (22445,)
(11055, 1576) (11055,)
(16500, 1576) (16500,)
====================================================================================================
```

Tfidf featurization

```
#for titles
from sklearn.feature_extraction.text import TfidfVectorizer
# We are considering only the words which appeared in at least 10 documents(rows or projects)
vectorizer8 = TfidfVectorizer(min_df=10,max_features=5000,ngram_range=(1, 2))# its a countvec
vectorizer8.fit(X_train_title)# that is learned from trainned  data


# we use the fitted CountVectorizer to convert the text to vector
X_train_tf_title = vectorizer8.transform(X_train_title)
X_cv_tf_title= vectorizer8.transform(X_cv_title)
X_test_tf_title = vectorizer8.transform(X_test_title)
```

```
print("After vectorizations")
print(X_train_tf_title.shape, y_train.shape)
print(X_cv_tf_title.shape, y_cv.shape)
print(X_test_tf_title.shape, y_test.shape)
print("="*100)
# so the dimension of alll are the same by using first fit and then transform
```

```
After vectorizations
(22445, 1576) (22445,)
(11055, 1576) (11055,)
(16500, 1576) (16500,)
====================================================================================================
```

```
#for essay
from sklearn.feature_extraction.text import TfidfVectorizer
# We are considering only the words which appeared in at least 10 documents(rows or projects)
vectorizer9 = TfidfVectorizer(min_df=10,max_features=5000,ngram_range=(1, 2))# its a countvec
vectorizer9.fit(X_train_essay)# that is learned from trainned  data



# we use the fitted CountVectorizer to convert the text to vector
X_train_tf_essay = vectorizer9.transform(X_train_essay)
X_cv_tf_essay= vectorizer9.transform(X_cv_essay)
X_test_tf_essay = vectorizer9.transform(X_test_essay)



print("After vectorizations")
print(X_train_tf_essay.shape, y_train.shape)
print(X_cv_tf_essay.shape, y_cv.shape)
print(X_test_tf_essay.shape, y_test.shape)
print("="*100)
# so the dimension of alll are the same by using first fit and then transform
```

```
After vectorizations
(22445, 5000) (22445,)
(11055, 5000) (11055,)
(16500, 5000) (16500,)
====================================================================================================
```

Using Pretrained Models: Avg W2V

```
# Reading glove vectors in python: https://stackoverflow.com/a/38230349/4084039
```

```
def loadGloveModel(gloveFile):

    print ("Loading Glove Model")

    f = open(gloveFile,'r', encoding = 'utf8')

    model = {}

    for line in tqdm(f):
        splitLine = line.split()
        word = splitLine[0]
        embedding = np.array([float(val) for val in splitLine[1:]])
        model[word] = embedding

    print ("Done.",len(model)," words loaded!")

    return model



 model = loadGloveModel('glove.42B.300d.txt')
```

```
Loading Glove Model
1917495it [09:04, 3519.90it/s]
Done. 1917495  words loaded!
```

```
glove_words = set(model.keys())
```

```
#for essay
# average Word2Vec
# compute average word2vec for each review.
def func(wordlist):


  train_avg_w2v_vectors = []; # the avg-w2v for each sentence/review is stored in this list
  for sentence in tqdm(wordlist): # for each review/sentence
    vector = np.zeros(300) # as word vectors are of zero length    # we are taking the 300 di
    cnt_words =0; # num of words with a valid vector in the sentence/review
    for word in sentence.split(): # for each word in a review/sentence
        if word in glove_words:
            vector += model[word]
            cnt_words += 1
    if cnt_words != 0:
        vector /= cnt_words
    train_avg_w2v_vectors.append(vector)

  print(len(train_avg_w2v_vectors))
  print(len(train_avg_w2v_vectors[0]))
  return train_avg_w2v_vectors


train_avg_w2v_vectors=func(preprocessed_essays_train)
```

```
test_avg_w2v_vectors=func(preprocessed_essays_test)
cv_avg_w2v_vectors=func(preprocessed_essays_cv)
#for titles

cv_avg_w2v_vectors_title=func(preprocessed_titles_cv)
test_avg_w2v_vectors_title=func(preprocessed_titles_test)
train_avg_w2v_vectors_title=func(preprocessed_titles_train)
```

```
100%|██████████████████████████████████████████████████████████████████████████| 22445/2
22445
300
100%|██████████████████████████████████████████████████████████████████████████| 16500/1
16500
300
100%|██████████████████████████████████████████████████████████████████████████| 11055/1
11055
300
100%|█████████████████████████████████████████████████████████████████████████| 11055/11
11055
300
100%|█████████████████████████████████████████████████████████████████████████| 16500/16
16500
300
100%|█████████████████████████████████████████████████████████████████████████| 22445/22
22445
300
```

Using Pretrained Models: TFIDF weighted W2V

```
tfidf_model = TfidfVectorizer()
tfidf_model.fit(preprocessed_essays_train)
# we are converting a dictionary with word as a key, and the idf as a value
dictionary = dict(zip(tfidf_model.get_feature_names(), list(tfidf_model.idf_)))
tfidf_words = set(tfidf_model.get_feature_names())


# average Word2Vec
# compute average word2vec for each review.
def tf_idf_done(word_list):

  train_title_tfidf_w2v_vectors = []; # the avg-w2v for each sentence/review is stored in thi
  for sentence in tqdm(word_list): # for each review/sentence
    vector = np.zeros(300) # as word vectors are of zero length
    tf_idf_weight =0; # num of words with a valid vector in the sentence/review
    for word in sentence.split():#.split(): # for each word in a review/sentence
        if (word in glove_words) and (word in tfidf_words):
            vec = model[word] # getting the vector for each word
            # here we are multiplying idf value(dictionary[word]) and the tf value((sentence.
            tf_idf = dictionary[word]*(sentence.count(word)/len(sentence.split())) # getting
            vector += (vec * tf_idf) # calculating tfidf weighted w2v
            tf_idf_weight += tf_idf
    if tf_idf_weight != 0:
        vector /= tf_idf_weight
    train_title_tfidf_w2v_vectors.append(vector)

  print(len(train_title_tfidf_w2v_vectors))
  print(len(train_title_tfidf_w2v_vectors[0]))
  return train_title_tfidf_w2v_vectors


train_tfidf_w2v_vectors=tf_idf_done(preprocessed_essays_train)
test_tfidf_w2v_vectors=tf_idf_done(preprocessed_essays_test)
cv_tfidf_w2v_vectors=tf_idf_done(preprocessed_essays_cv)

train_title_tfidf_w2v_vectors=tf_idf_done(preprocessed_titles_train)
test_title_tfidf_w2v_vectors=tf_idf_done(preprocessed_titles_test)
cv_title_tfidf_w2v_vectors=tf_idf_done(preprocessed_titles_cv)
```

```
100%|██████████████████████████████████████| 22445/
22445
300
100%|██████████████████████████████████████| 16500/
16500
300
100%|██████████████████████████████████████| 11055/
11055
300
100%|█████████████████████████████████████| 22445/22
22445
300
100%|█████████████████████████████████████| 16500/16
16500
300
100%|█████████████████████████████████████| 11055/11
11055
300
```

Vectorizing Numerical features

```
price_data = dfr.groupby('id').agg({'price':'sum', 'quantity':'sum'}).reset_index()
dft = pd.merge(dft, price_data, on='id', how='left')
print(price_data.head(2))


# we also have to do this in tran,test and cv
# so also merge the resource data with the trian,cv and test

X_train = pd.merge(X_train, price_data, on = "id", how = "left")
#print(x_train.columns)
X_test = pd.merge(X_test, price_data, on = "id", how = "left")
X_cv = pd.merge(X_cv, price_data, on = "id", how = "left")
```

```
        id   price  quantity
0  p000001  459.56         7
1  p000002  515.89        21
```

```
#standardization
# check this one: https://www.youtube.com/watch?v=0HOqOcln3Z4&t=530s
# standardization sklearn: https://scikit-learn.org/stable/modules/generated/sklearn.preproce
from sklearn.preprocessing import StandardScaler
from sklearn.preprocessing import MinMaxScaler
from sklearn import preprocessing


price_scalar =  StandardScaler()

price_scalar.fit(X_train['price'].values.reshape(-1,1)) # finding the mean and standard devia
#print(f"Mean : {price_scalar.mean_[0]}, Standard deviation : {np.sqrt(price_scalar.var_[0])}

# Now standardize the data with above maen and variance.
train_price_standar = price_scalar.transform(X_train['price'].values.reshape(-1, 1))



# Now standardize the data with above maen and variance.
test_price_standar = price_scalar.transform(X_test['price'].values.reshape(-1, 1))



# Now standardize the data with above maen and variance.
cv_price_standar = price_scalar.transform(X_cv['price'].values.reshape(-1, 1))


# previous_year_projects
price_scalar.fit(X_train['teacher_number_of_previously_posted_projects'].values.reshape(-1,1)
#print(f"Mean : {price_scalar.mean_[0]}, Standard deviation : {np.sqrt(price_scalar.var_[0])}

# Now standardize the data with above maen and variance.
train_prev_proj_standar = price_scalar.transform(X_train['teacher_number_of_previously_posted
# Now standardize the data with above maen and variance.
test_prev_proj_standar = price_scalar.transform(X_test['teacher_number_of_previously_posted_p
# Now standardize the data with above maen and variance.
cv_prev_proj_standar = price_scalar.transform(X_cv['teacher_number_of_previously_posted_proje




price_scalar.fit(X_train['quantity'].values.reshape(-1,1)) # finding the mean and standard de
#print(f"Mean : {price_scalar.mean_[0]}, Standard deviation : {np.sqrt(price_scalar.var_[0])}

# Now standardize the data with above maen and variance.
train_qnty_standar = price_scalar.transform(X_train['quantity'].values.reshape(-1, 1))
```

```
# Now standardize the data with above maen and variance.
cv_qnty_standar = price_scalar.transform(X_cv['quantity'].values.reshape(-1, 1))


# Now standardize the data with above maen and variance.
test_qnty_standar = price_scalar.transform(X_test['quantity'].values.reshape(-1, 1))
```

MERGING

```
from scipy.sparse import hstack
# with the same hstack function we are concatinating a sparse matrix and a dense matirx :)
X_set1_train = hstack((X_train_bow_title,X_train_bow,# all bows
                      X_train_teacher_prefix,X_train_cat,X_train_subcat ,X_train_project_grad
                      train_qnty_standar,train_price_standar,train_prev_proj_standar))# all n


print(X_set1_train.shape, y_train.shape)
```

```
(22445, 6678) (22445,)
```

```
from scipy.sparse import hstack
# with the same hstack function we are concatinating a sparse matrix and a dense matirx :)
X_set1_cv = hstack((X_cv_bow_title,X_cv_bow,
                      X_cv_teacher_prefix,X_cv_cat,X_cv_subcat,
                      X_cv_project_grade_category,X_cv_school_state, cv_qnty_standar,cv_price_


print(X_set1_cv.shape, y_cv.shape)
```

```
(11055, 6678) (11055,)
```

```
from scipy.sparse import hstack
# with the same hstack function we are concatinating a sparse matrix and a dense matirx :)
X_set1_test = hstack((X_test_bow_title,X_test_bow,
                      X_test_teacher_prefix,X_test_cat,X_test_subcat,
                      X_test_project_grade_category,X_test_school_state,
                      test_qnty_standar,test_price_standar,test_prev_proj_standar))


print(X_set1_test.shape, y_test.shape)
```

```
(16500, 6678) (16500,)
```

```
from scipy.sparse import hstack
# with the same hstack function we are concatinating a sparse matrix and a dense matirx :)
X_set2_train = hstack((X_train_tf_essay,X_train_tf_title,
                      X_train_teacher_prefix,X_train_cat,X_train_subcat,
                      X_train_project_grade_category,X_train_school_state,
                       train_qnty_standar,train_price_standar,train_prev_proj_standar))


print(X_set2_train.shape, y_train.shape)
```

```
(22445, 6678) (22445,)
```

```
from scipy.sparse import hstack
# with the same hstack function we are concatinating a sparse matrix and a dense matirx :)
X_set2_cv = hstack((X_cv_tf_essay,X_cv_tf_title,
                      X_cv_teacher_prefix,X_cv_cat,X_cv_subcat,
                      X_cv_project_grade_category,X_cv_school_state,
                   cv_qnty_standar,cv_price_standar,cv_prev_proj_standar))


print(X_set2_cv.shape, y_cv.shape)
```

```
(11055, 6678) (11055,)
```

```
from scipy.sparse import hstack
# with the same hstack function we are concatinating a sparse matrix and a dense matirx :)
X_set2_test = hstack((X_test_tf_essay,X_test_tf_title,
                      X_test_teacher_prefix,X_test_cat,X_test_subcat,
                      X_test_project_grade_category,X_test_school_state,
                     test_qnty_standar,test_price_standar,test_prev_proj_standar))


print(X_set2_test.shape, y_test.shape)
```

```
(16500, 6678) (16500,)
```

```
from scipy.sparse import hstack
# with the same hstack function we are concatinating a sparse matrix and a dense matirx :)
X_set3_train = hstack((train_avg_w2v_vectors,train_avg_w2v_vectors_title,train_prev_proj_stan
                      X_train_teacher_prefix,X_train_cat,X_train_subcat,
                      X_train_project_grade_category,X_train_school_state))
```

```
print(X_set3_train.shape, y_train.shape)
```

```
(22445, 702) (22445,)
```

```
from scipy.sparse import hstack
# with the same hstack function we are concatinating a sparse matrix and a dense matirx :)
X_set3_cv = hstack((cv_avg_w2v_vectors,cv_avg_w2v_vectors_title,cv_prev_proj_standar,cv_price
                      X_cv_teacher_prefix,X_cv_cat,X_cv_subcat,
                      X_cv_project_grade_category,X_cv_school_state))
```

```
print(X_set3_cv.shape, y_cv.shape)
```

```
(11055, 702) (11055,)
```

```
from scipy.sparse import hstack
# with the same hstack function we are concatinating a sparse matrix and a dense matirx :)
X_set3_test = hstack((test_avg_w2v_vectors,test_avg_w2v_vectors_title,test_prev_proj_standar,
                      X_test_teacher_prefix,X_test_cat,X_test_subcat,
                      X_test_project_grade_category,X_test_school_state))
```

```
print(X_set3_test.shape, y_test.shape)
```

```
(16500, 702) (16500,)
```

```
from scipy.sparse import hstack
# with the same hstack function we are concatinating a sparse matrix and a dense matirx :)
X_set4_train = hstack((train_tfidf_w2v_vectors,train_title_tfidf_w2v_vectors,train_prev_proj_
                      X_train_teacher_prefix,X_train_cat,X_train_subcat,
                      X_train_project_grade_category,X_train_school_state))
```

```
print(X_set4_train.shape, y_train.shape)
```

```
(22445, 702) (22445,)
```

```
from scipy.sparse import hstack
# with the same hstack function we are concatinating a sparse matrix and a dense matirx :)
X_set4_cv = hstack((cv_tfidf_w2v_vectors,cv_title_tfidf_w2v_vectors,cv_prev_proj_standar,cv_p
                      X_cv_teacher_prefix,X_cv_cat,X_cv_subcat,
                      X_cv_project_grade_category,X_cv_school_state))


print(X_set4_cv.shape, y_cv.shape)
```

```
(11055, 702) (11055,)
```

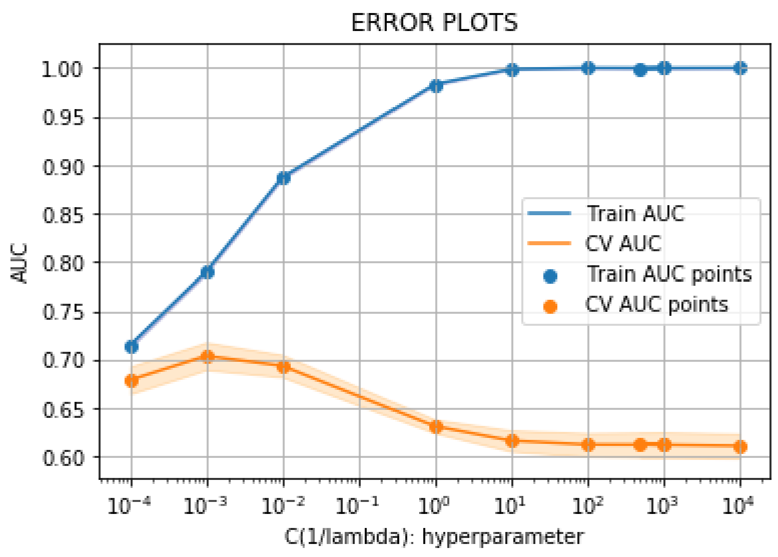
```
from scipy.sparse import hstack
# with the same hstack function we are concatinating a sparse matrix and a dense matirx :)
X_set4_test = hstack((test_title_tfidf_w2v_vectors,test_tfidf_w2v_vectors,test_prev_proj_stan
                      X_test_teacher_prefix,X_test_cat,X_test_subcat,
                      X_test_project_grade_category,X_test_school_state))


print(X_set4_test.shape, y_test.shape)
```

```
(16500, 702) (16500,)
```

Logistic Regression on BOW

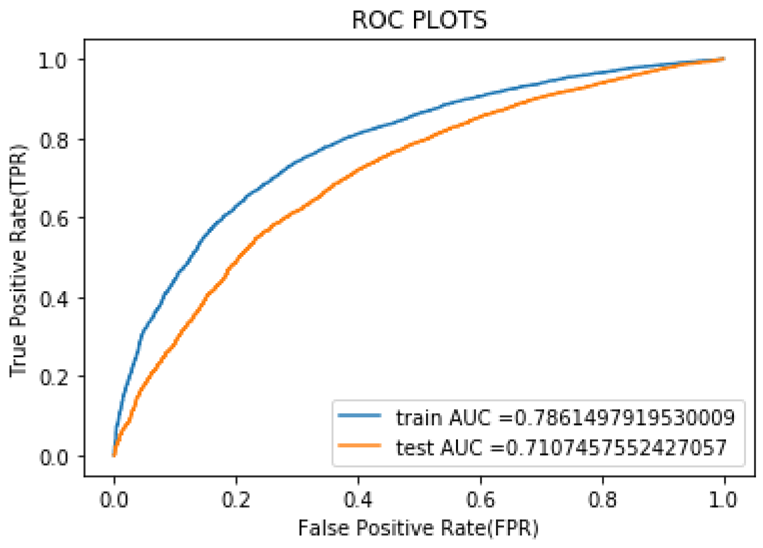
```
import warnings
warnings.filterwarnings('ignore')
from sklearn.metrics import roc_auc_score
import matplotlib.pyplot as plt
#from sklearn.grid_search import GridSearchCV
from sklearn.linear_model import LogisticRegression
from sklearn.model_selection import learning_curve, GridSearchCV

"""
y_true : array, shape = [n_samples] or [n_samples, n_classes]
True binary labels or binary label indicators.

y_score : array, shape = [n_samples] or [n_samples, n_classes]
Target scores, can either be probability estimates of the positive class, confidence values,
decisions (as returned by “decision_function” on some classifiers).
For binary y_true, y_score is supposed to be the score of the class with greater label.

"""



clf = LogisticRegression(class_weight='balanced');
parameters ={'C':[10**-4, 10**-3,10**-2,1,10,100,1000,500,1000,10000]}
sd=GridSearchCV(clf, parameters, cv=5, scoring='roc_auc',return_train_score=True)
sd.fit(X_set1_train, y_train);
```

```
train_auc= sd.cv_results_['mean_train_score']
train_auc_std= sd.cv_results_['std_train_score']
cv_auc = sd.cv_results_['mean_test_score']
cv_auc_std= sd.cv_results_['std_test_score']

plt.plot(parameters['C'], train_auc, label='Train AUC')
# this code is copied from here: https://stackoverflow.com/a/48803361/4084039
plt.gca().fill_between(parameters['C'],train_auc - train_auc_std,train_auc + train_auc_std,al

plt.plot(parameters['C'], cv_auc, label='CV AUC')
# this code is copied from here: https://stackoverflow.com/a/48803361/4084039
plt.gca().fill_between(parameters['C'],cv_auc - cv_auc_std,cv_auc + cv_auc_std,alpha=0.2,colo

plt.scatter(parameters['C'], train_auc, label='Train AUC points')
plt.scatter(parameters['C'], cv_auc, label='CV AUC points')
plt.xscale('log')

plt.legend()
plt.xlabel("C(1/lambda): hyperparameter")
plt.ylabel("AUC")
plt.title("ERROR PLOTS")
plt.grid()
plt.show()
```



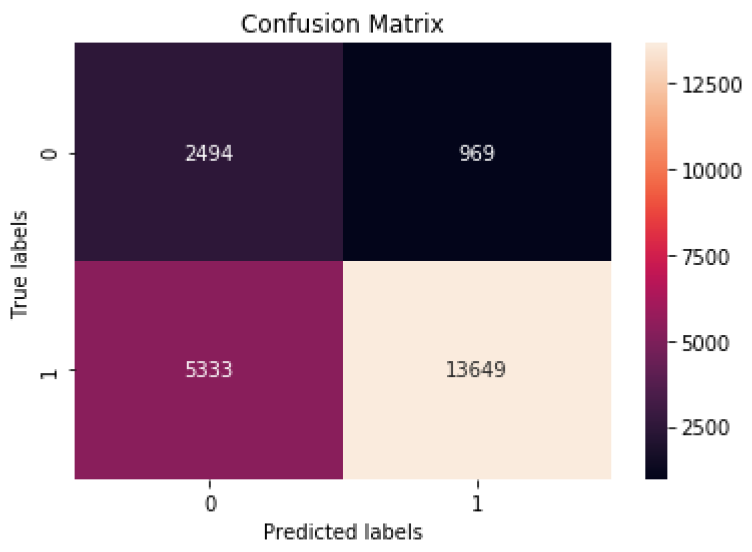
```
##Fitting Model to Hyper-Parameter Curve
# https://scikit-learn.org/stable/modules/generated/sklearn.metrics.roc_curve.html#sklearn.me
from sklearn.metrics import roc_curve, auc


neigh = LogisticRegression(C=10**-3,class_weight='balanced');
neigh.fit(X_set1_train ,y_train)
# roc_auc_score(y_true, y_score) the 2nd parameter should be probability estimates of the pos
# not the predicted outputs

train_fpr, train_tpr, thresholds = roc_curve(y_train, neigh.predict_proba(X_set1_train)[:,1])
```

```
test_fpr, test_tpr, thresholds = roc_curve(y_test, neigh.predict_proba(X_set1_test)[:,1])

plt.plot(train_fpr, train_tpr, label="train AUC ="+str(auc(train_fpr, train_tpr)))
plt.plot(test_fpr, test_tpr, label="test AUC ="+str(auc(test_fpr, test_tpr)))
plt.legend()
plt.ylabel("True Positive Rate(TPR)")
plt.xlabel("False Positive Rate(FPR)")
plt.title("ROC PLOTS")
plt.show()
```

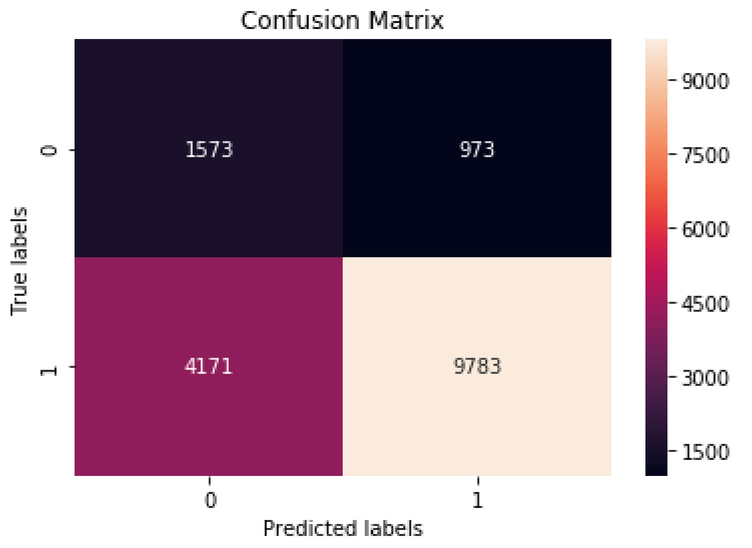


OBSERVATIONS: As we seen form the roc plot ,MOdel works well 71 auc score also good

```
#https://stackoverflow.com/questions/35572000/how-can-i-plot-a-confusion-matrix
import seaborn as sns
import matplotlib.pyplot as plt

ax= plt.subplot()
sns.heatmap(confusion_matrix(y_train, neigh.predict(X_set1_train )), annot=True, ax = ax,fmt=

# labels, title and ticks
ax.set_xlabel('Predicted labels');
ax.set_ylabel('True labels');
ax.set_title('Confusion Matrix');
#ax.xaxis.set_ticklabels(['business', 'health']); ax.yaxis.set_ticklabels(['health', 'busines
```



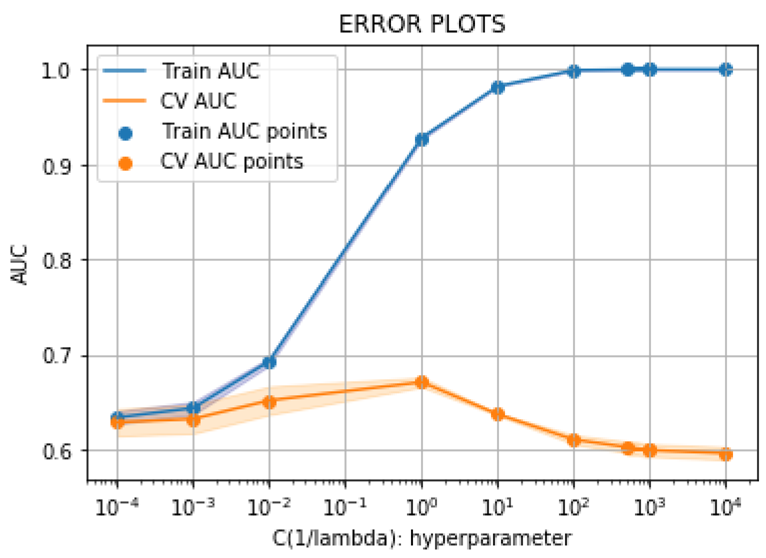
```
#https://stackoverflow.com/questions/35572000/how-can-i-plot-a-confusion-matrix
import seaborn as sns
import matplotlib.pyplot as plt

ax= plt.subplot()
sns.heatmap(confusion_matrix(y_test, neigh.predict(X_set1_test )), annot=True, ax = ax,fmt='g

# labels, title and ticks
ax.set_xlabel('Predicted labels');
ax.set_ylabel('True labels');
ax.set_title('Confusion Matrix');
#ax.xaxis.set_ticklabels(['business', 'health']); ax.yaxis.set_ticklabels(['health', 'busines
```



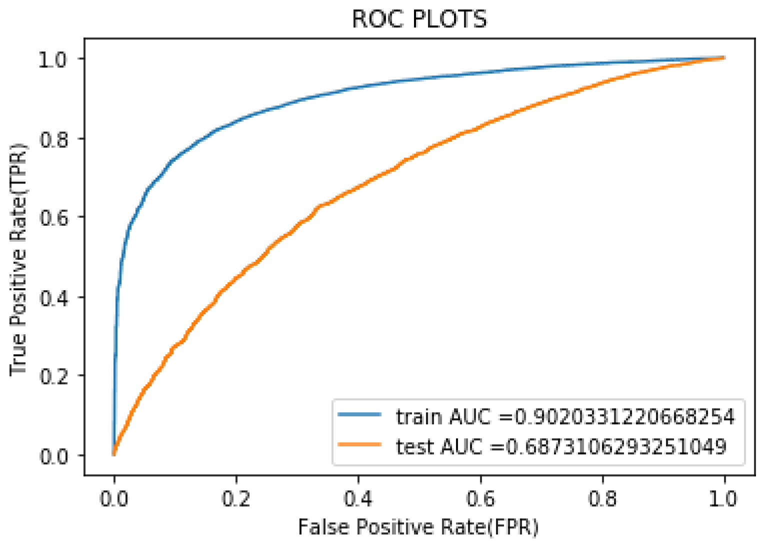
logistic regression on TFIDF

```
from sklearn.metrics import roc_auc_score
```

```
import matplotlib.pyplot as plt
#from sklearn.grid_search import GridSearchCV
from sklearn.linear_model import LogisticRegression
from sklearn.model_selection import learning_curve, GridSearchCV

"""
y_true : array, shape = [n_samples] or [n_samples, n_classes]
True binary labels or binary label indicators.

y_score : array, shape = [n_samples] or [n_samples, n_classes]
Target scores, can either be probability estimates of the positive class, confidence values,
decisions (as returned by “decision_function” on some classifiers).
For binary y_true, y_score is supposed to be the score of the class with greater label.

"""

clf = LogisticRegression(class_weight='balanced');
parameters ={'C':[10**-4, 10**-3,10**-2,1,10,100,1000,500,1000,10000]}
sd = GridSearchCV(clf, parameters, cv=3, scoring='roc_auc',return_train_score=True)
sd.fit(X_set2_train, y_train);

train_auc= sd.cv_results_['mean_train_score']
train_auc_std= sd.cv_results_['std_train_score']
cv_auc =sd.cv_results_['mean_test_score']
cv_auc_std=sd.cv_results_['std_test_score']

plt.plot(parameters['C'], train_auc, label='Train AUC')
# this code is copied from here: https://stackoverflow.com/a/48803361/4084039
plt.gca().fill_between(parameters['C'],train_auc - train_auc_std,train_auc + train_auc_std,al

plt.plot(parameters['C'], cv_auc, label='CV AUC')
# this code is copied from here: https://stackoverflow.com/a/48803361/4084039
plt.gca().fill_between(parameters['C'],cv_auc - cv_auc_std,cv_auc + cv_auc_std,alpha=0.2,colo

plt.scatter(parameters['C'], train_auc, label='Train AUC points')
plt.scatter(parameters['C'], cv_auc, label='CV AUC points')
plt.xscale('log')

plt.legend()
plt.xlabel("C(1/lambda): hyperparameter")
plt.ylabel("AUC")
plt.title("ERROR PLOTS")
plt.grid()
plt.show()
```



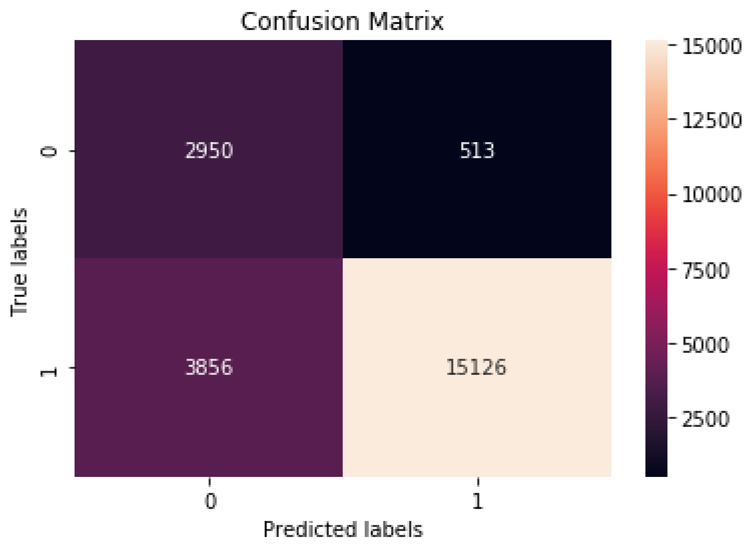
```
#Fitting Model to Hyper-Parameter Curve
# https://scikit-learn.org/stable/modules/generated/sklearn.metrics.roc_curve.html#sklearn.me
from sklearn.metrics import roc_curve, auc


neigh = LogisticRegression(C=1,class_weight='balanced');
neigh.fit(X_set2_train ,y_train)
# roc_auc_score(y_true, y_score) the 2nd parameter should be probability estimates of the pos
# not the predicted outputs

train_fpr, train_tpr, thresholds = roc_curve(y_train, neigh.predict_proba(X_set2_train)[:,1])
test_fpr, test_tpr, thresholds = roc_curve(y_test, neigh.predict_proba(X_set2_test)[:,1])

plt.plot(train_fpr, train_tpr, label="train AUC ="+str(auc(train_fpr, train_tpr)))
plt.plot(test_fpr, test_tpr, label="test AUC ="+str(auc(test_fpr, test_tpr)))
plt.legend()
plt.ylabel("True Positive Rate(TPR)")
plt.xlabel("False Positive Rate(FPR)")
plt.title("ROC PLOTS")
plt.show()
```



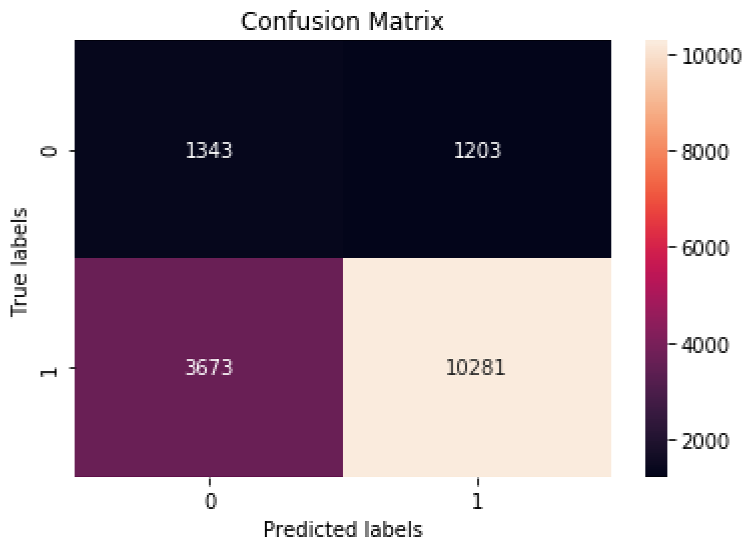
OBSERVATONS: So in trian data roc curve is good , but trian data curve is very much high from the tes

COnfusion matrix

```
#https://stackoverflow.com/questions/35572000/how-can-i-plot-a-confusion-matrix
import seaborn as sns
import matplotlib.pyplot as plt

ax= plt.subplot()
sns.heatmap(confusion_matrix(y_train, neigh.predict(X_set2_train )), annot=True, ax = ax,fmt=

# labels, title and ticks
ax.set_xlabel('Predicted labels');
ax.set_ylabel('True labels');
ax.set_title('Confusion Matrix');
#ax.xaxis.set_ticklabels(['business', 'health']); ax.yaxis.set_ticklabels(['health', 'busines
```



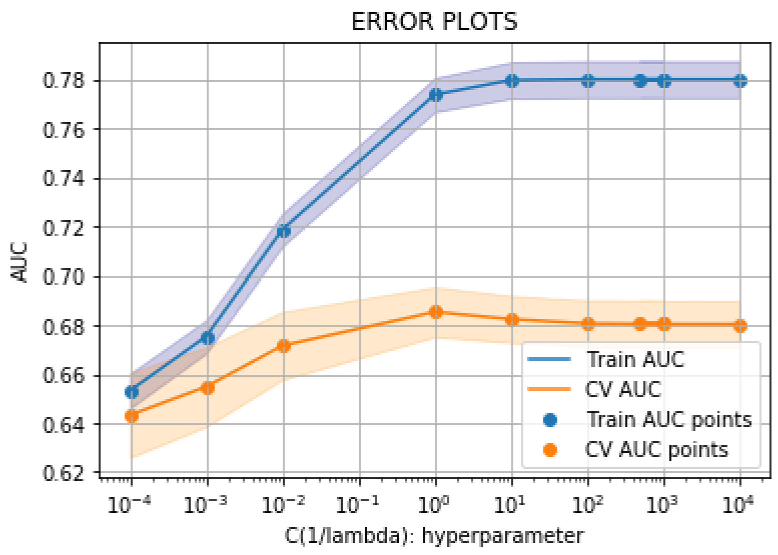
```
#https://stackoverflow.com/questions/35572000/how-can-i-plot-a-confusion-matrix
import seaborn as sns
import matplotlib.pyplot as plt

ax= plt.subplot()
sns.heatmap(confusion_matrix(y_test, neigh.predict(X_set2_test )), annot=True, ax = ax,fmt='g

# labels, title and ticks
ax.set_xlabel('Predicted labels');
ax.set_ylabel('True labels');
ax.set_title('Confusion Matrix');
#ax.xaxis.set_ticklabels(['business', 'health']); ax.yaxis.set_ticklabels(['health', 'busines
```



logistic regresion on AVG W2V

```
from sklearn.metrics import roc_auc_score
import matplotlib.pyplot as plt
#from sklearn.grid_search import GridSearchCV
from sklearn.linear_model import LogisticRegression
from sklearn.model_selection import learning_curve, GridSearchCV

"""
y_true : array, shape = [n_samples] or [n_samples, n_classes]
True binary labels or binary label indicators.

y_score : array, shape = [n_samples] or [n_samples, n_classes]
Target scores, can either be probability estimates of the positive class, confidence values,
decisions (as returned by “decision_function” on some classifiers).
For binary y_true, y_score is supposed to be the score of the class with greater label.

"""

clf = LogisticRegression(class_weight='balanced');
```

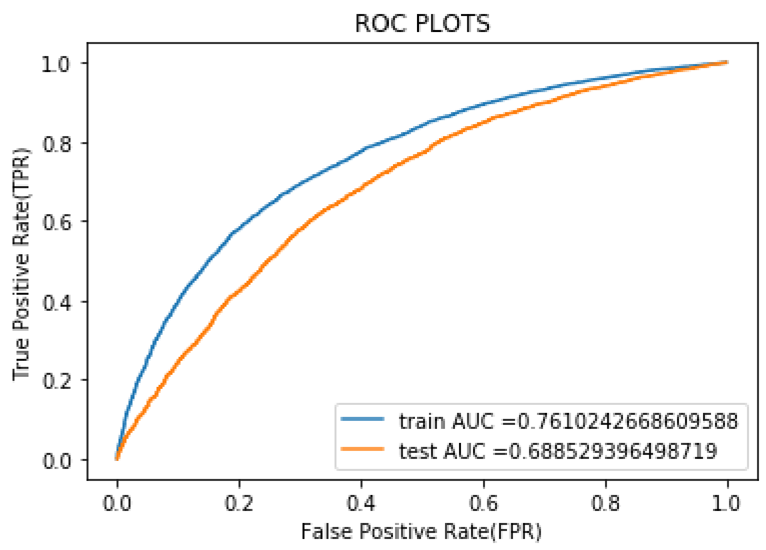
```
parameters ={'C':[10**-4, 10**-3,10**-2,1,10,100,1000,500,1000,10000]}
cl = GridSearchCV(clf , parameters, cv=3, scoring='roc_auc',return_train_score=True)
cl.fit(X_set3_train, y_train);

train_auc= cl.cv_results_['mean_train_score']
train_auc_std= cl.cv_results_['std_train_score']
cv_auc = cl.cv_results_['mean_test_score']
cv_auc_std= cl.cv_results_['std_test_score']

plt.plot(parameters['C'], train_auc, label='Train AUC')
# this code is copied from here: https://stackoverflow.com/a/48803361/4084039
plt.gca().fill_between(parameters['C'],train_auc - train_auc_std,train_auc + train_auc_std,al

plt.plot(parameters['C'], cv_auc, label='CV AUC')
# this code is copied from here: https://stackoverflow.com/a/48803361/4084039
plt.gca().fill_between(parameters['C'],cv_auc - cv_auc_std,cv_auc + cv_auc_std,alpha=0.2,colo

plt.scatter(parameters['C'], train_auc, label='Train AUC points')
plt.scatter(parameters['C'], cv_auc, label='CV AUC points')
plt.xscale('log')

plt.legend()
plt.xlabel("C(1/lambda): hyperparameter")
plt.ylabel("AUC")
plt.title("ERROR PLOTS")
plt.grid()
plt.show()
```



Fitting Model to Hyper-Parameter Curve:

```
#Fitting Model to Hyper-Parameter Curve:
# https://scikit-learn.org/stable/modules/generated/sklearn.metrics.roc_curve.html#sklearn.me
from sklearn.metrics import roc_curve, auc
```

```
neigh = LogisticRegression(C=1,class_weight='balanced');
neigh.fit(X_set3_train ,y_train)
# roc_auc_score(y_true, y_score) the 2nd parameter should be probability estimates of the pos
# not the predicted outputs

train_fpr, train_tpr, thresholds = roc_curve(y_train, neigh.predict_proba(X_set3_train)[:,1])
test_fpr, test_tpr, thresholds = roc_curve(y_test, neigh.predict_proba(X_set3_test)[:,1])

plt.plot(train_fpr, train_tpr, label="train AUC ="+str(auc(train_fpr, train_tpr)))
plt.plot(test_fpr, test_tpr, label="test AUC ="+str(auc(test_fpr, test_tpr)))
plt.legend()
plt.ylabel("True Positive Rate(TPR)")
plt.xlabel("False Positive Rate(FPR)")
plt.title("ROC PLOTS")
plt.show()
print("="*100)
```



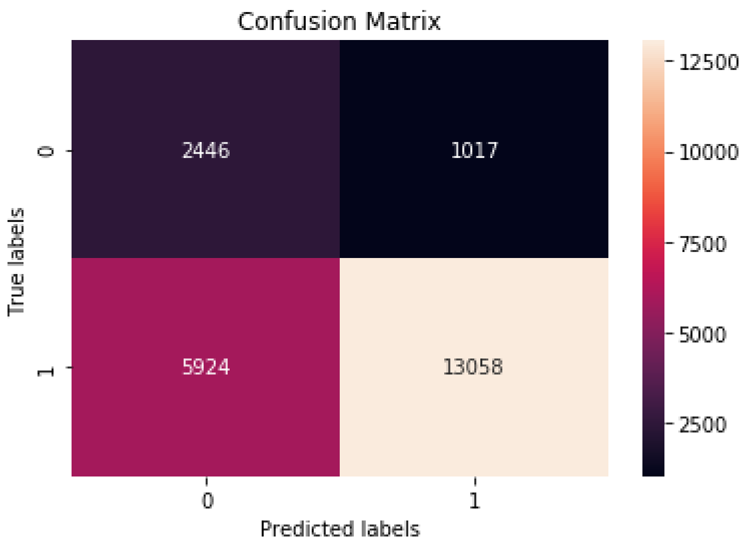
```
====================================================================================
```

Observations: So logistic regressoin with word2vec works prettywell , train and test roc curve very clo
better than LR with tf_idf of essay and titles

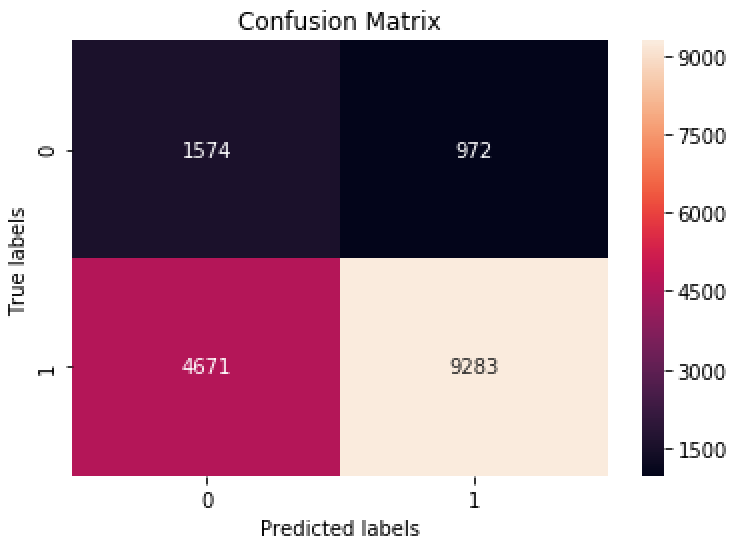
confusion matrix of train and test data

```
#https://stackoverflow.com/questions/35572000/how-can-i-plot-a-confusion-matrix
import seaborn as sns
import matplotlib.pyplot as plt

ax= plt.subplot()
sns.heatmap(confusion_matrix(y_train, neigh.predict(X_set3_train )), annot=True, ax = ax,fmt=

# labels, title and ticks
ax.set_xlabel('Predicted labels');
ax.set_ylabel('True labels');
ax.set_title('Confusion Matrix');
```

```
#ax.xaxis.set_ticklabels(['business', 'health']); ax.yaxis.set_ticklabels(['health', 'busines
```

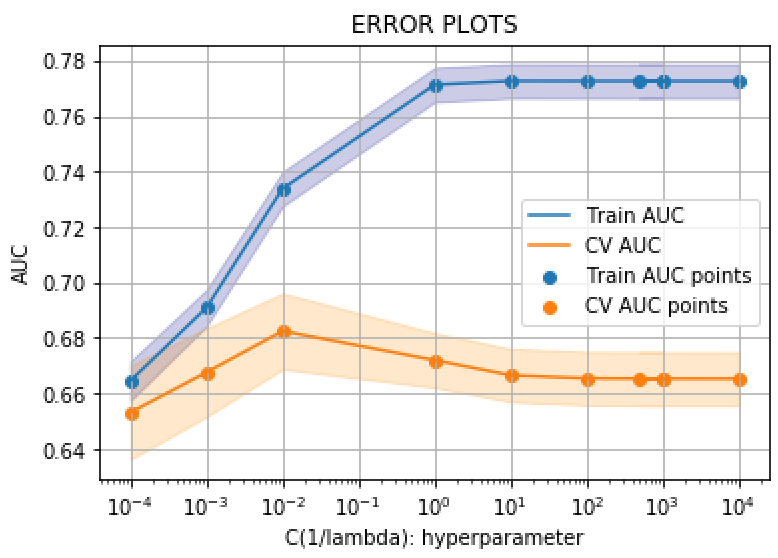


```
#for test data
#https://stackoverflow.com/questions/35572000/how-can-i-plot-a-confusion-matrix
import seaborn as sns
import matplotlib.pyplot as plt

ax= plt.subplot()
sns.heatmap(confusion_matrix(y_test, neigh.predict(X_set3_test )), annot=True, ax = ax,fmt='g

# labels, title and ticks
ax.set_xlabel('Predicted labels');
ax.set_ylabel('True labels');
ax.set_title('Confusion Matrix');
#ax.xaxis.set_ticklabels(['business', 'health']); ax.yaxis.set_ticklabels(['health', 'busines
```

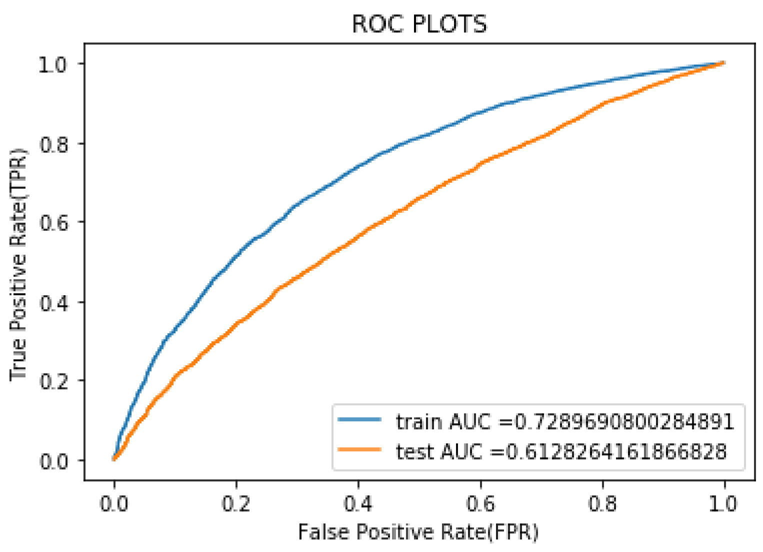


logistic regresion on td_idf W2V

```
from sklearn.metrics import roc_auc_score
import matplotlib.pyplot as plt
#from sklearn.grid_search import GridSearchCV
from sklearn.linear_model import LogisticRegression
from sklearn.model_selection import learning_curve, GridSearchCV

"""
y_true : array, shape = [n_samples] or [n_samples, n_classes]
True binary labels or binary label indicators.

y_score : array, shape = [n_samples] or [n_samples, n_classes]
Target scores, can either be probability estimates of the positive class, confidence values,
decisions (as returned by “decision_function” on some classifiers).
For binary y_true, y_score is supposed to be the score of the class with greater label.

"""

clf = LogisticRegression(class_weight='balanced');
parameters ={'C':[10**-4, 10**-3,10**-2,1,10,100,1000,500,1000,10000]}
cl = GridSearchCV(clf, parameters, cv=3, scoring='roc_auc',return_train_score=True)
cl.fit(X_set4_train, y_train);

train_auc= cl.cv_results_['mean_train_score']
train_auc_std= cl.cv_results_['std_train_score']
cv_auc = cl.cv_results_['mean_test_score']
cv_auc_std= cl.cv_results_['std_test_score']

plt.plot(parameters['C'], train_auc, label='Train AUC')
# this code is copied from here: https://stackoverflow.com/a/48803361/4084039
plt.gca().fill_between(parameters['C'],train_auc - train_auc_std,train_auc + train_auc_std,al

plt.plot(parameters['C'], cv_auc, label='CV AUC')
# this code is copied from here: https://stackoverflow.com/a/48803361/4084039
plt.gca().fill_between(parameters['C'],cv_auc - cv_auc_std,cv_auc + cv_auc_std,alpha=0.2,colo

plt.scatter(parameters['C'], train_auc, label='Train AUC points')
plt.scatter(parameters['C'], cv_auc, label='CV AUC points')
plt.xscale('log')

plt.legend()
plt.xlabel("C(1/lambda): hyperparameter")
plt.ylabel("AUC")
plt.title("ERROR PLOTS")
plt.grid()
plt.show()
```



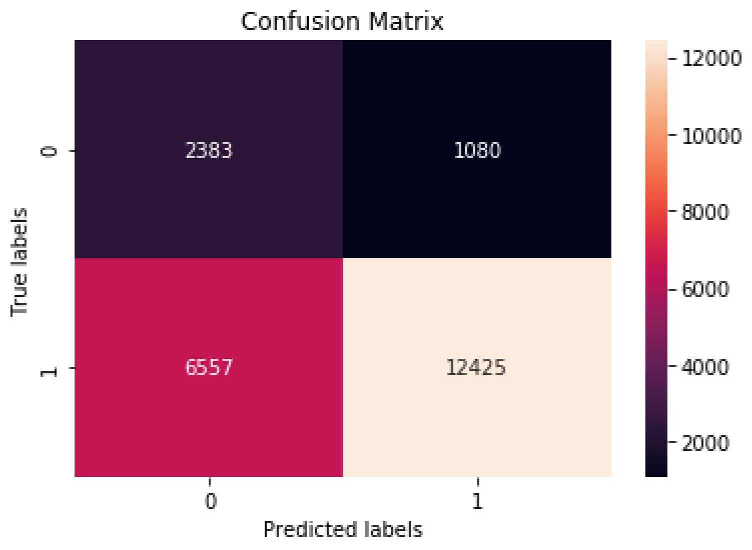
```
#Fitting Model to Hyper-Parameter Curve:
# https://scikit-learn.org/stable/modules/generated/sklearn.metrics.roc_curve.html#sklearn.me
from sklearn.metrics import roc_curve, auc


neigh = LogisticRegression(C=10**-2,class_weight='balanced');
neigh.fit(X_set4_train ,y_train)
# roc_auc_score(y_true, y_score) the 2nd parameter should be probability estimates of the pos
# not the predicted outputs

train_fpr, train_tpr, thresholds = roc_curve(y_train, neigh.predict_proba(X_set4_train)[:,1])
test_fpr, test_tpr, thresholds = roc_curve(y_test, neigh.predict_proba(X_set4_test)[:,1])

plt.plot(train_fpr, train_tpr, label="train AUC ="+str(auc(train_fpr, train_tpr)))
plt.plot(test_fpr, test_tpr, label="test AUC ="+str(auc(test_fpr, test_tpr)))
plt.legend()
plt.ylabel("True Positive Rate(TPR)")
plt.xlabel("False Positive Rate(FPR)")
plt.title("ROC PLOTS")
plt.show()
```



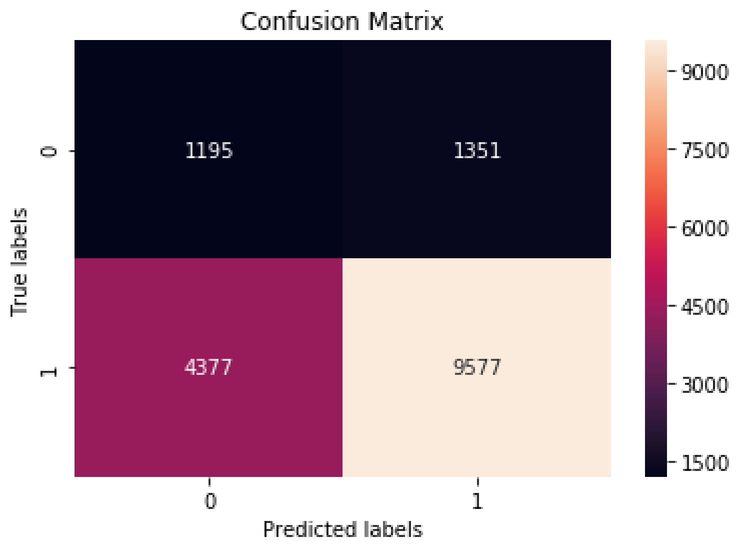
Observation: This is overfitting, in train data roc is good but in test data roc curve is only 61,so much l

Confusion matrix

```
#https://stackoverflow.com/questions/35572000/how-can-i-plot-a-confusion-matrix
import seaborn as sns
import matplotlib.pyplot as plt

ax= plt.subplot()
sns.heatmap(confusion_matrix(y_train, neigh.predict(X_set4_train )), annot=True, ax = ax,fmt=

# labels, title and ticks
ax.set_xlabel('Predicted labels');
ax.set_ylabel('True labels');
ax.set_title('Confusion Matrix');
#ax.xaxis.set_ticklabels(['business', 'health']); ax.yaxis.set_ticklabels(['health', 'busines
```



```
#https://stackoverflow.com/questions/35572000/how-can-i-plot-a-confusion-matrix
import seaborn as sns
import matplotlib.pyplot as plt

ax= plt.subplot()
sns.heatmap(confusion_matrix(y_test, neigh.predict(X_set4_test )), annot=True, ax = ax,fmt='g

# labels, title and ticks
ax.set_xlabel('Predicted labels');
ax.set_ylabel('True labels');
ax.set_title('Confusion Matrix');
#ax.xaxis.set_ticklabels(['business', 'health']); ax.yaxis.set_ticklabels(['health', 'busines
```



New feature(No. of words in title)

```
# For train data
title_length_train=[]
for i in range(0,22445):
  title_length_train.append(len(X_train["project_title"][i].split()))

title_length_train=np.array(title_length_train)

#for test data titles
title_length_test=[]
for i in range(0,16500):
  title_length_test.append(len(X_test["project_title"][i].split()))

title_length_test=np.array(title_length_test)

#for cv data titles

title_length_cv=[]
for i in range(0,11055):
  title_length_cv.append(len(X_cv["project_title"][i].split()))

title_length_cv=np.array(title_length_cv)
```

New feature(No. of words in combined essays)

```
#for test data esssay
essay_length_test=[]
for i in range(0,16500):
  essay_length_test.append(len(X_test["essay"][i].split()))

essay_length_test=np.array(essay_length_test)

#for cv data essay

essay_length_cv=[]
for i in range(0,11055):
  essay_length_cv.append(len(X_cv["essay"][i].split()))

essay_length_cv=np.array(essay_length_cv)


#for train data essay

essay_length_train=[]
for i in range(0,22445):
  essay_length_train.append(len(X_train["essay"][i].split()))

essay_length_train=np.array(essay_length_train)
```

New feature(Sentiment scores of each combined essay's)

```
import nltk
from nltk.sentiment.vader import SentimentIntensityAnalyzer
nltk.download('vader_lexicon')


#https://www.programcreek.com/python/example/100005/nltk.sentiment.vader.SentimentIntensityAn
def analyze_sentiment(df):
    sentiments = []
    sid = SentimentIntensityAnalyzer()
    for i in range(df.shape[0]):
        line = df['essay'][i]# take one essay
        sentiment = sid.polarity_scores(line)# calculate the sentiment
        sentiments.append([sentiment['neg'], sentiment['pos'],
                           sentiment['neu'], sentiment['compound']])# list of lists
    df[['neg', 'pos', 'neu', 'compound']] = pd.DataFrame(sentiments)
    df['Negative'] = df['compound'] < -0.1
    df['Positive'] = df['compound'] > 0.1
    return df
```

```
[nltk_data] Downloading package vader_lexicon to
[nltk_data]     C:\Users\Hp\AppData\Roaming\nltk_data...
[nltk_data]   Package vader_lexicon is already up-to-date!
```

```
X_train=analyze_sentiment(X_train)
X_test=analyze_sentiment(X_test)
X_cv=analyze_sentiment(X_cv)


#for train

pos=list(X_train['pos'])
pos=np.array(pos)
neg=list(X_train['neg'])
neg=np.array(neg)
com=list(X_train['compound'])
com=np.array(com)

# combine all
from scipy.sparse import hstack
# with the same hstack function we are concatinating a sparse matrix and a dense matirx :)
X_set5_train = hstack((
                      X_train_teacher_prefix,X_train_cat,X_train_subcat ,X_train_project_grad
                      train_qnty_standar,train_price_standar,train_prev_proj_standar,
                      essay_length_train.reshape(-1,1),title_length_train.reshape(-1,1),
                      pos.reshape(-1,1),neg.reshape(-1,1),com.reshape(-1,1),
                                                                                   ))# all numericals


print(X_set5_train.shape, y_train.shape)
```

```
(22445, 107) (22445,)
```

```
#For cv

pos=list(X_cv['pos'])
pos=np.array(pos)

neg=list(X_cv['neg'])
neg=np.array(neg)

com=list(X_cv['compound'])
com=np.array(com)

# combine all
```

```
from scipy.sparse import hstack
# with the same hstack function we are concatinating a sparse matrix and a dense matirx :)
X_set5_cv = hstack((
                      X_cv_teacher_prefix,X_cv_cat,X_cv_subcat ,X_cv_project_grade_category,X
                      cv_qnty_standar,cv_price_standar,cv_prev_proj_standar,
                      essay_length_cv.reshape(-1,1),title_length_cv.reshape(-1,1),
                      pos.reshape(-1,1),neg.reshape(-1,1),com.reshape(-1,1),
                                                                   ))# all numericals


print(X_set5_cv.shape, y_cv.shape)
```

```
(11055, 107) (11055,)
```

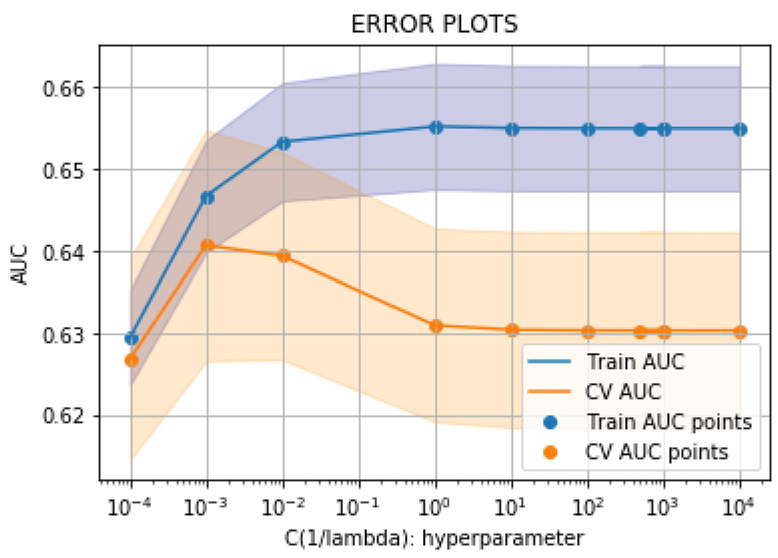
```
#for test
pos=list(X_test['pos'])
pos=np.array(pos)

neg=list(X_test['neg'])
neg=np.array(neg)

com=list(X_test['compound'])
com=np.array(com)

# combine all
from scipy.sparse import hstack
# with the same hstack function we are concatinating a sparse matrix and a dense matirx :)
X_set5_test = hstack((
                      X_test_teacher_prefix,X_test_cat,X_test_subcat ,X_test_project_grade_ca
                      test_qnty_standar,test_price_standar,test_prev_proj_standar,
                      essay_length_test.reshape(-1,1),title_length_test.reshape(-1,1),
                      pos.reshape(-1,1),neg.reshape(-1,1),com.reshape(-1,1),
                                                                   ))# all numericals


print(X_set5_test.shape, y_test.shape)
```

```
(16500, 107) (16500,)
```

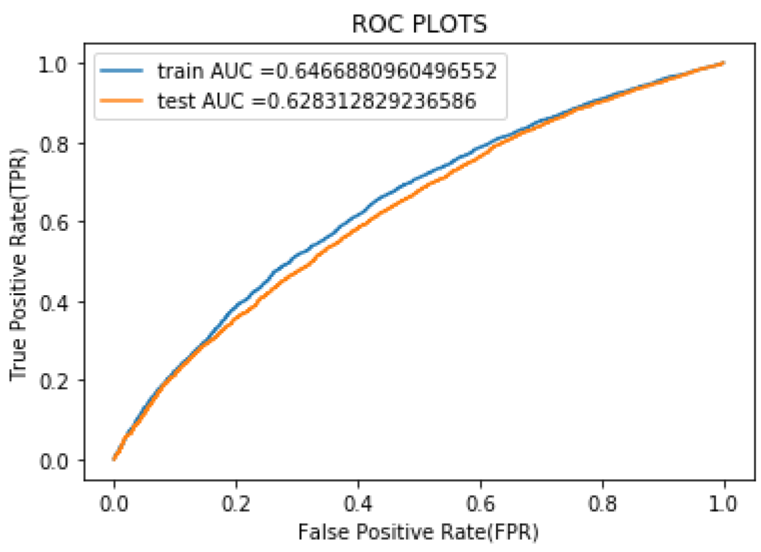
logistic regression on SET 5

```
from sklearn.metrics import roc_auc_score
import matplotlib.pyplot as plt
#from sklearn.grid_search import GridSearchCV
from sklearn.linear_model import LogisticRegression
from sklearn.model_selection import learning_curve, GridSearchCV

"""
```

```
y_true : array, shape = [n_samples] or [n_samples, n_classes]
True binary labels or binary label indicators.

y_score : array, shape = [n_samples] or [n_samples, n_classes]
Target scores, can either be probability estimates of the positive class, confidence values,
decisions (as returned by “decision_function” on some classifiers).
For binary y_true, y_score is supposed to be the score of the class with greater label.

"""

clf = LogisticRegression(class_weight='balanced');
parameters ={'C':[10**-4, 10**-3,10**-2,1,10,100,1000,500,1000,10000]}
cl = GridSearchCV(clf, parameters, cv=3, scoring='roc_auc',return_train_score=True)
cl.fit(X_set5_train, y_train);

train_auc= cl.cv_results_['mean_train_score']
train_auc_std= cl.cv_results_['std_train_score']
cv_auc = cl.cv_results_['mean_test_score']
cv_auc_std= cl.cv_results_['std_test_score']

plt.plot(parameters['C'], train_auc, label='Train AUC')
# this code is copied from here: https://stackoverflow.com/a/48803361/4084039
plt.gca().fill_between(parameters['C'],train_auc - train_auc_std,train_auc + train_auc_std,al

plt.plot(parameters['C'], cv_auc, label='CV AUC')
# this code is copied from here: https://stackoverflow.com/a/48803361/4084039
plt.gca().fill_between(parameters['C'],cv_auc - cv_auc_std,cv_auc + cv_auc_std,alpha=0.2,colo

plt.scatter(parameters['C'], train_auc, label='Train AUC points')
plt.scatter(parameters['C'], cv_auc, label='CV AUC points')
plt.xscale('log')

plt.legend()
plt.xlabel("C(1/lambda): hyperparameter")
plt.ylabel("AUC")
plt.title("ERROR PLOTS")
plt.grid()
plt.show()
```



```
# https://scikit-learn.org/stable/modules/generated/sklearn.metrics.roc_curve.html#sklearn.me
from sklearn.metrics import roc_curve, auc


neigh = LogisticRegression(C=10**-3,class_weight='balanced');
neigh.fit(X_set5_train ,y_train)
# roc_auc_score(y_true, y_score) the 2nd parameter should be probability estimates of the pos
# not the predicted outputs

train_fpr, train_tpr, thresholds = roc_curve(y_train, neigh.predict_proba(X_set5_train)[:,1])
test_fpr, test_tpr, thresholds = roc_curve(y_test, neigh.predict_proba(X_set5_test)[:,1])

plt.plot(train_fpr, train_tpr, label="train AUC ="+str(auc(train_fpr, train_tpr)))
plt.plot(test_fpr, test_tpr, label="test AUC ="+str(auc(test_fpr, test_tpr)))
plt.legend()
plt.ylabel("True Positive Rate(TPR)")
plt.xlabel("False Positive Rate(FPR)")
plt.title("ROC PLOTS")
plt.show()
```

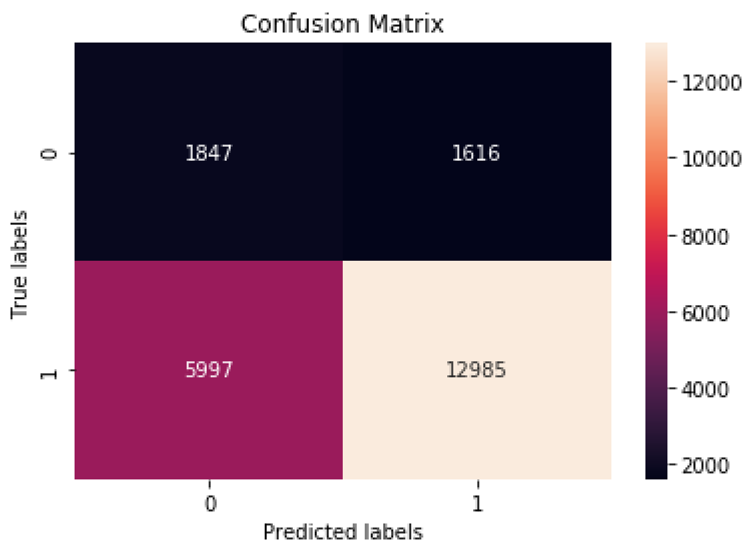


Observation:

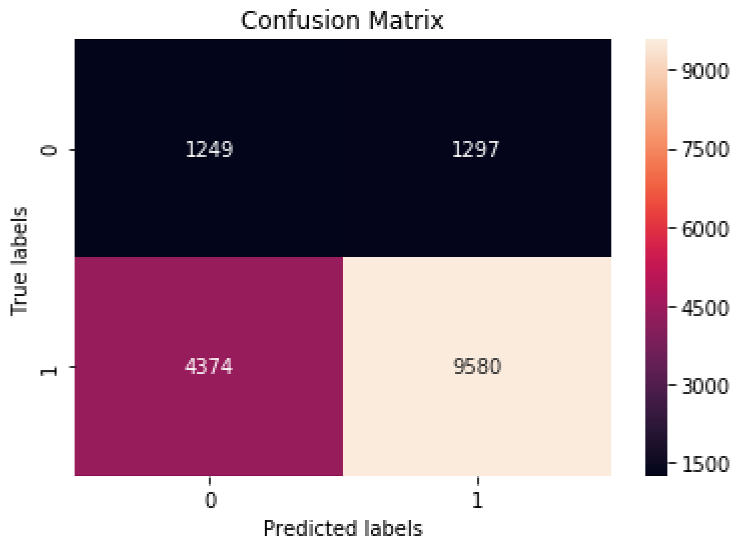
In this plot their is no overfitting so this roc curve is better than roc curves in which we used bow or tf
without feturizatoins our confusion matirx so bad, predicting negatives class wrong, also (model with
same.

Confusion matrix

```
#https://stackoverflow.com/questions/35572000/how-can-i-plot-a-confusion-matrix
import seaborn as sns
import matplotlib.pyplot as plt

ax= plt.subplot()
sns.heatmap(confusion_matrix(y_train, neigh.predict(X_set5_train )), annot=True, ax = ax,fmt=

# labels, title and ticks
ax.set_xlabel('Predicted labels');
ax.set_ylabel('True labels');
ax.set_title('Confusion Matrix');
#ax.xaxis.set_ticklabels(['business', 'health']); ax.yaxis.set_ticklabels(['health', 'busines
```



```
#https://stackoverflow.com/questions/35572000/how-can-i-plot-a-confusion-matrix
import seaborn as sns
import matplotlib.pyplot as plt

ax= plt.subplot()
sns.heatmap(confusion_matrix(y_test, neigh.predict(X_set5_test )), annot=True, ax = ax,fmt='g

# labels, title and ticks
ax.set_xlabel('Predicted labels');
ax.set_ylabel('True labels');
ax.set_title('Confusion Matrix');
#ax.xaxis.set_ticklabels(['business', 'health']); ax.yaxis.set_ticklabels(['health', 'busines
```



Observations:

1. If we compare the roc curves between model with featurizations and model without featurization, the model v
2. Confusion matrix is bad in both but in (with featurizatoin model) the confusion matrix is little bit good from th

3. Conclusions

```
# Please compare all your models using Prettytable library
# Please compare all your models using Prettytable library
#how to use pretty table http://zetcode.com/python/prettytable/
from prettytable import PrettyTable

tb = PrettyTable()
tb.field_names= ("Vectorizer", "Model", "HyperParameter", "AUC")
tb.add_row(["BOW", "Auto",10**-3, 71])
tb.add_row(["Tf-Idf", "Auto",1, 68])
tb.add_row(["AVGW2V", "Auto",1, 68])
tb.add_row(["Tf-Idf w2v", "Auto", 10**-2, 61])
tb.add_row(["Set 5", "Auto",10**-3, 62])
print(tb.get_string(titles = "Logistic Reg> - Observations"))
```

```
+------------+-------+----------------+-----+
| Vectorizer | Model | HyperParameter | AUC |
+------------+-------+----------------+-----+
|    BOW     |  Auto |     0.001      |  71 |
|   Tf-Idf   |  Auto |       1        |  68 |
|   AVGW2V   |  Auto |       1        |  68 |
| Tf-Idf w2v |  Auto |      0.01      |  61 |
|   Set 5    |  Auto |     0.001      |  62 |
+------------+-------+----------------+-----+
```