Objective: To train the model using Truncated_SVD.

In [0]:

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

```
Go to this URL in a browser: https://accounts.
google.com/o/oauth2/auth?client_id=94731898980
3-6bn6qk8qdgf4n4g3pfee6491hc0brc4i.apps.google
usercontent.com&redirect_uri=urn%3Aietf%3Awg%3
Aoauth%3A2.0%3Aoob&scope=email%20https%3A%2F%2
Fwww.googleapis.com%2Fauth%2Fdocs.test%20https
%3A%2F%2Fwww.googleapis.com%2Fauth%2Fdrive%20h
ttps%3A%2F%2Fwww.googleapis.com%2Fauth%2Fdrive
.photos.readonly%20https%3A%2F%2Fwww.googleapi
s.com%2Fauth%2Fpeopleapi.readonly&response_typ
e=code

Enter your authorization code:
··********··
Mounted at /content/drive
```

In [0]:

```
%cd /content/drive/My Drive
```

```
/content/drive/My Drive
```

In [0]:

```
%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.c
om/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
```

Reading Data

In [0]:

```
project_data = pd.read_csv('train_data.csv')
resource_data = pd.read_csv('resources.csv')
```

In [0]:

```
print("Number of data points in train data", project_data.sha
pe)
print(project_data.columns)
```

```
Number of data points in train data (109248, 1
7)
Index(['Unnamed: 0', 'id', 'teacher_id', 'teac
her_prefix', 'school_state',
       'project_submitted_datetime', 'project_
grade_category',
       'project_subject_categories', 'project_
subject_subcategories',
       'project_title', 'project_essay_1', 'pr
oject_essay_2',
       'project_essay_3', 'project_essay_4', '
project_resource_summary',
       'teacher_number_of_previously_posted_pr
ojects', 'project_is_approved'],
      dtype='object')
```

In [0]:

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

```
#sort dataframe based on time pandas python: https://stackove
rflow.com/a/49702492/4084039
project_data['Date'] = pd.to_datetime(project_data['project_s
ubmitted_datetime'])
project_data.drop('project_submitted_datetime', axis=1, inpla
ce=True)
project_data.sort_values(by=['Date'], inplace=True)


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


project_data.head(2)
```

Out[0]:

	Unnamed: 0	id	teacher_id	teacher_prefix	sch
55660	8393	p205479	2bf07ba08945e5d8b2a3f269b2b3cfe5	Mrs.	
76127	37728	p043609	3f60494c61921b3b43ab61bdde2904df	Ms.	

In [0]:

```
print("Number of data points in resource data", resource_data
.shape)
print("Number of data points in resource data", resource_data
.columns)
resource_data.head(2)
```

```
Number of data points in resource data (154127
2, 4)
Number of data points in resource data Index([
'id', 'description', 'quantity', 'price'], dty
pe='object')
```

Out[0]:

	id	description	quantity	price
0	p233245	LC652 - Lakeshore Double-Space Mobile Drying Rack	1	149.00
1	p069063	Bouncy Bands for Desks (Blue support pipes)	3	14.95

In [0]:

```
# https://stackoverflow.com/questions/22407798/how-to-reset-a
-dataframes-indexes-for-all-groups-in-one-step
price_data = resource_data.groupby('id').agg({'price':'sum',
'quantity':'sum'}).reset_index()
price_data.head(2)
```

Out[0]:

	id	price	quantity
0	p000001	459.56	7
1	p000002	515.89	21

In [0]:

```
# join two dataframes in python:
project_data = pd.merge(project_data, price_data, on='id', ho
w='left')
```

In [0]:

```
project_data.head(2)
```

Out[0]:

	Unnamed: 0	id	teacher_id	teacher_prefix	school_
0	8393	p205479	2bf07ba08945e5d8b2a3f269b2b3cfe5	Mrs.	
1	37728	p043609	3f60494c61921b3b43ab61bdde2904df	Ms.	

In [0]:

```
project_data.shape
```

Out[0]:

```
(109248, 19)
```

In [0]:

```
final_appr = project_data[project_data['project_is_approved']
 == 1]
final_appr = final_appr.sample(n=21000)
final_appr.shape
```

Out[0]:

```
(21000, 19)
```

In [0]:

```
final_rej =  project_data[project_data['project_is_approved']
 == 0]
final_rej = final_rej.sample(n=4000)
final_rej.shape
```

Out[0]:

```
(4000, 19)
```

In [0]:

```
final=pd.concat([final_appr,final_rej])
final=final.sort_values('Date', axis=0, ascending=True, inpla
ce=False, kind='quicksort', na_position='last')
final.shape
```

Out[0]:

```
(25000, 19)
```

In [0]:

```
project_data = final
```

In [0]:

```
project_data.shape
```

Out[0]:

```
(25000, 19)
```

1.1 Preprocessing of project_subject_categories

In [0]:

```
catogories = list(project_data['project_subject_categories'].
values)
# remove special characters from list of strings python: http
s://stackoverflow.com/a/47301924/4084039

# https://www.geeksforgeeks.org/removing-stop-words-nltk-pyth
on/
# https://stackoverflow.com/questions/23669024/how-to-strip-a
-specific-word-from-a-string
```

```
# https://stackoverflow.com/questions/8270092/remove-all-whit
espace-in-a-string-in-python
cat_list = []
for i in 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", "Care & Hunger"]
        if 'The' in j.split(): # this will split each of the
catogory based on space "Math & Science"=> "Math","&", "Scien
ce"
            j=j.replace('The','') # if we have the words "The
" we are going to replace it with ''(i.e removing 'The')
        j = j.replace(' ','') # we are placeing all the ' '(s
pace) with ''(empty) ex:"Math & Science"=>"Math&Science"
        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())

project_data['clean_categories'] = cat_list
project_data.drop(['project_subject_categories'], axis=1, inp
lace=True)

from collections import Counter
my_counter = Counter()
for word in project_data['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]))
```

1.2 Preprocessing of project_subject_subcategories

In [0]:

```
sub_catogories = list(project_data['project_subject_subcatego
ries'].values)
# remove special characters from list of strings python: http
s://stackoverflow.com/a/47301924/4084039

# https://www.geeksforgeeks.org/removing-stop-words-nltk-pyth
on/
# https://stackoverflow.com/questions/23669024/how-to-strip-a
-specific-word-from-a-string
# https://stackoverflow.com/questions/8270092/remove-all-whit
espace-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", "Care & Hunger"]
        if 'The' in j.split(): # this will split each of the
catogory based on space "Math & Science"=> "Math","&", "Scien
ce"
            j=j.replace('The','') # if we have the words "The
" we are going to replace it with ''(i.e removing 'The')
        j = j.replace(' ','') # we are placeing all the ' '(s
pace) with ''(empty) ex:"Math & Science"=>"Math&Science"
        temp +=j.strip()+" "#" abc ".strip() will return "abc
", remove the trailing spaces
        temp = temp.replace('&','_')
    sub_cat_list.append(temp.strip())
```

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

# count of all the words in corpus python: https://stackoverf
low.com/a/22898595/4084039
my_counter = Counter()
for word in project_data['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=l
ambda kv: kv[1]))
```

2.Text Preprocessing

2.1 Preprocessing of essay

In [0]:

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

In [0]:

```
project_data.head(2)
```

Out[0]:

	Unnamed: 0	id	teacher_id	teacher_prefix	school
0	8393	p205479	2bf07ba08945e5d8b2a3f269b2b3cfe5	Mrs.	
10	57854	p099430	4000cfe0c8b2df75a218347c1765e283	Ms.	

In [0]:

```
# printing some random reviews
print(project_data['essay'].values[0])
print("="*50)
print(project_data['essay'].values[150])
print("="*50)
print(project_data['essay'].values[1000])
print("="*50)
print(project_data['essay'].values[20000])
print("="*50)
```

```
I have been fortunate enough to use the Fairy
Tale STEM kits in my classroom as well as the
STEM journals, which my students really enjoye
d.  I would love to implement more of the Lake
shore STEM kits in my classroom for the next s
chool year as they provide excellent and engag
ing STEM lessons.My students come from a varie
ty of backgrounds, including language and soci
oeconomic status.  Many of them don't have a l
ot of experience in science and engineering an
d these kits give me the materials to provide
these exciting opportunities for my students.E
ach month I try to do several science or STEM/
```

STEAM projects. I would use the kits and robo
t to help guide my science instruction in enga
ging and meaningful ways. I can adapt the kit
s to my current language arts pacing guide whe
re we already teach some of the material in th
e kits like tall tales (Paul Bunyan) or Johnny
 Appleseed. The following units will be taugh
t in the next school year where I will impleme
nt these kits: magnets, motion, sink vs. float
, robots. I often get to these units and don'
t know If I am teaching the right way or using
 the right materials. The kits will give me
 additional ideas, strategies, and lessons to
prepare my students in science.It is challengi
ng to develop high quality science activities.
 These kits give me the materials I need to p
rovide my students with science activities tha
t will go along with the curriculum in my clas
sroom. Although I have some things (like magn
ets) in my classroom, I don't know how to use
them effectively. The kits will provide me wi
th the right amount of materials and show me h
ow to use them in an appropriate way.
==========
There is no such thing as a typical day in my
classroom. While we always follow a schedule
and have a daily routine, my students help gui
de the day because of their needs. My challen
ge in getting them to reach their potential an
d yearly goals is their sensory regulation and
 attention difficulties.I have 8 amazing child
ren, all on the Autism spectrum. All of my ch
ildren need help with sensory regulation every
 day. They also need help learning how to cal
m themselves, with support or independently.
Fine motor is also a challenge for many in my

class. We are one of 3 Special Education clas
srooms in a Child Development Center. The fac
ility and staff here is wonderful, but we ofte
n struggle with having the appropriate tools a
nd materials necessary for my kids.The pebble
crayons will help my students with proper gras
p for their pre-writing skills. The swing and
 bean bag chairs will help my children with th
e sensory regulation that they need. Some of
my students need swinging or rocking to help f
eel at ease. For others, the wrapping of the
bean bag around them and feel of the beads, pr
ovide extra sensory input to help calm them.By
 providing my students with these seats and fi
ne motor materials, you will be helping to pro
vide them with the tools necessary for self re
gulation. When they have better regulation, t
he students are better able to attend to instr
uctional times, and participate in social acti
vities more appropriately with their peers (in
stead of seeking out sensory input in less app
ropriate ways).
==========
====
The students at my school are curious and spon
ges that just want to learn. The school is th
eir constant safe place. They come from vast
backgrounds but school is one community. The
 children do not have an enormous amount of th
ings to bring to the classroom - but they do h
ave heart. \r\n\r\nThe community is in a low
 socio-economic neighborhood. Students live
in shelters, single parent homes, and foster
homes. We have children from Pre K to 5th
grade and a very transient population. Childr
en become homeless or move at the drop of a ha
t.These materials will bring creativity and im

agination to my classroom. The children can use these for all projects they need to complete. I want them available at all times. These items get used up so quickly. \r\nThink about how these items and how the children will have hours of enjoyment using them. \r\nRemember what it was like to get brand new supplies before school started - you can help my students have that feeling. A group of brand new crayons, sharpened colored pencils, liquid glue, and sparkly construction paper. Art cannot be taken out of the classrooms. Art supplies are not funded and given to teachers. I buy as much as I can but reach out for help when I need to. Please help me. \r\nnannan
==========
Our students represent a very diversified school. We have students from Israel, ex-Soviet Union countries many of us have never heard about, such as Georgia, Uzbekistan, Belorussia, in addition to students from all over Asia, Africa, and Central and South America. Our students come from low income families. Many of them live in the shelter, far away from the school and take a long journey to come to school. Many of them are a single parent families or live in the foster family. Despite the conditions our students live in, they show good attendance and are eager to learn.\r\n \r\nMy students are eager to become outstanding athletes so we would be proud of them.My students are highly involved in our School Team Sports. We have created a softball team to represent our school against other similar schools. Unfortunately, our small school has no budget for our students' Team Sports. Thus, as a coach looking at

```
 their eagerness to perform high, I need to he
lp them by getting exercise mats so they can g
et stronger.\r\n\r\nMy students are involved i
n the Softball Team to the degree that they ar
e eager to become the best Softball Team cityw
ide.\r\nAdditionally, they achieve high scores
 on the State Tests. For this reason, they nee
d to build their strength. I am asking for the
 exercise mats, so they can practice stretchin
g, push-ups, crunches, sit ups, and curl-ups i
n order to be ready to perform high.\r\n\r\nSi
nce my students snack before, during, and afte
r they play their favorite Softball game, I am
 concerned about their calories intake. Thus,
I need a scale so they would monitor their bod
y weight, fat, and calories during practicing
time for the Tournament and they would also us
e the scale to use at the end of the Tournamen
t. It is only thanks to Dick's Sporting Goods
who funded our equipment for the upcoming Tour
nament and with these supplies requested, they
 will succeed in March's Tournament.nannan
================================================
====
```

In [0]:

```python
# 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
```

In [0]:

```
sent = decontracted(project_data['essay'].values[20000])
print(sent)
print("="*50)
```

```
Our students represent a very diversified scho
ol. We have students from Israel, ex-Soviet Un
ion countries many of us have never heard abou
t, such as Georgia, Uzbekistan, Belorussia, in
 addition to students from all over Asia, Afri
ca, and Central and South America. Our student
s come from low income families. Many of them
live in the shelter, far away from the school
and take a long journey to come to school. Man
y of them are a single parent families or live
 in the foster family. Despite the conditions
our students live in, they show good attendanc
e and are eager to learn.\r\n   \r\nMy student
s are eager to become outstanding athletes so
we would be proud of them.My students are high
ly involved in our School Team Sports. We have
 created a softball team to represent our scho
ol against other similar schools. Unfortunatel
y, our small school has no budget for our stud
ents' Team Sports. Thus, as a coach looking at
 their eagerness to perform high, I need to he
lp them by getting exercise mats so they can g
et stronger.\r\n\r\nMy students are involved i
```

```
n the Softball Team to the degree that they ar
e eager to become the best Softball Team cityw
ide.\r\nAdditionally, they achieve high scores
 on the State Tests. For this reason, they nee
d to build their strength. I am asking for the
 exercise mats, so they can practice stretchin
g, push-ups, crunches, sit ups, and curl-ups i
n order to be ready to perform high.\r\n\r\nSi
nce my students snack before, during, and afte
r they play their favorite Softball game, I am
 concerned about their calories intake. Thus,
I need a scale so they would monitor their bod
y weight, fat, and calories during practicing
time for the Tournament and they would also us
e the scale to use at the end of the Tournamen
t. It is only thanks to Dick is Sporting Goods
 who funded our equipment for the upcoming Tou
rnament and with these supplies requested, the
y will succeed in March is Tournament.nannan
==============================================
====
```

In [0]:

```
# \r \n \t remove from string python: http://texthandler.com/
info/remove-line-breaks-python/
sent = sent.replace('\\r', ' ')
sent = sent.replace('\\"', ' ')
sent = sent.replace('\\n', ' ')
print(sent)
```

```
Our students represent a very diversified scho
ol. We have students from Israel, ex-Soviet Un
ion countries many of us have never heard abou
t, such as Georgia, Uzbekistan, Belorussia, in
 addition to students from all over Asia, Afri
ca, and Central and South America. Our student
s come from low income families. Many of them
```

live in the shelter, far away from the school and take a long journey to come to school. Many of them are a single parent families or live in the foster family. Despite the conditions our students live in, they show good attendance and are eager to learn. My students are eager to become outstanding athletes so we would be proud of them.My students are highly involved in our School Team Sports. We have created a softball team to represent our school against other similar schools. Unfortunately, our small school has no budget for our students' Team Sports. Thus, as a coach looking at their eagerness to perform high, I need to help them by getting exercise mats so they can get stronger. My students are involved in the Softball Team to the degree that they are eager to become the best Softball Team citywide. Additionally, they achieve high scores on the State Tests. For this reason, they need to build their strength. I am asking for the exercise mats, so they can practice stretching, push-ups, crunches, sit ups, and curl-ups in order to be ready to perform high. Since my students snack before, during, and after they play their favorite Softball game, I am concerned about their calories intake. Thus, I need a scale so they would monitor their body weight, fat, and calories during practicing time for the Tournament and they would also use the scale to use at the end of the Tournament. It is only thanks to Dick is Sporting Goods who funded our equipment for the upcoming Tournament and with these supplies requested, they will succeed in March is Tournament.nannan

In [0]:

```
#remove spacial character: https://stackoverflow.com/a/5843547/4084039
sent = re.sub('[^A-Za-z0-9]+', ' ', sent)
print(sent)
```

```
Our students represent a very diversified scho
ol We have students from Israel ex Soviet Unio
n countries many of us have never heard about
such as Georgia Uzbekistan Belorussia in addit
ion to students from all over Asia Africa and
Central and South America Our students come fr
om low income families Many of them live in th
e shelter far away from the school and take a
long journey to come to school Many of them ar
e a single parent families or live in the fost
er family Despite the conditions our students
live in they show good attendance and are eage
r to learn My students are eager to become out
standing athletes so we would be proud of them
 My students are highly involved in our School
 Team Sports We have created a softball team t
o represent our school against other similar s
chools Unfortunately our small school has no b
udget for our students Team Sports Thus as a c
oach looking at their eagerness to perform hig
h I need to help them by getting exercise mats
 so they can get stronger My students are invo
lved in the Softball Team to the degree that t
hey are eager to become the best Softball Team
 citywide Additionally they achieve high score
s on the State Tests For this reason they need
 to build their strength I am asking for the e
xercise mats so they can practice stretching p
ush ups crunches sit ups and curl ups in order
 to be ready to perform high Since my students
 snack before during and after they play their
 favorite Softball game I am concerned about t
```

```
heir calories intake Thus I need a scale so th
ey would monitor their body weight fat and cal
ories during practicing time for the Tournamen
t and they would also use the scale to use at
the end of the Tournament It is only thanks to
 Dick is Sporting Goods who funded our equipme
nt for the upcoming Tournament and with these
supplies requested they will succeed in March
is Tournament nannan
```

In [0]:

```
# https://gist.github.com/sebleier/554280
# 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", "you've",\
            "you'll", "you'd", 'your', 'yours', 'yourself', '
yourselves', 'he', 'him', 'his', 'himself', \
            'she', "she's", 'her', 'hers', 'herself', 'it', "
it's", 'its', 'itself', 'they', 'them', 'their',\
            'theirs', 'themselves', 'what', 'which', 'who', '
whom', 'this', 'that', "that'll", 'these', 'those', \
            'am', 'is', 'are', 'was', 'were', 'be', 'been', '
being', 'have', 'has', 'had', 'having', 'do', 'does', \
            'did', 'doing', 'a', 'an', 'the', 'and', 'but', '
if', 'or', 'because', 'as', 'until', 'while', 'of', \
            'at', 'by', 'for', 'with', 'about', 'against', 'b
etween', 'into', 'through', 'during', 'before', 'after',\
            'above', 'below', 'to', 'from', 'up', 'down', 'in
', 'out', 'on', 'off', 'over', 'under', 'again', 'further',\
            'then', 'once', 'here', 'there', 'when', 'where',
 'why', 'how', 'all', 'any', 'both', 'each', 'few', 'more',\
            'most', 'other', 'some', 'such', 'only', 'own', '
same', 'so', 'than', 'too', 'very', \
            's', 't', 'can', 'will', 'just', 'don', "don't",
'should', "should've", 'now', 'd', 'll', 'm', 'o', 're', \
```

```
            've', 'y', 'ain', 'aren', "aren't", 'couldn', "co
uldn't", 'didn', "didn't", 'doesn', "doesn't", 'hadn',\
            "hadn't", 'hasn', "hasn't", 'haven', "haven't", '
isn', "isn't", 'ma', 'mightn', "mightn't", 'mustn',\
            "mustn't", 'needn', "needn't", 'shan', "shan't",
'shouldn', "shouldn't", 'wasn', "wasn't", 'weren', "weren't",
 \
            'won', "won't", 'wouldn', "wouldn't"]
```

In [0]:

```
# Combining all the above stundents
from tqdm import tqdm
preprocessed_essays = []
# tqdm is for printing the status bar
for sentance in tqdm(project_data['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.append(sent.lower().strip())
```

```
100%|██████████| 25000/25000 [00:12<00:00, 194
2.08it/s]
```

In [0]:

```
# after preprocesing
preprocessed_essays[20000]
```

Out[0]:

```
'students represent diversified school student
s israel ex soviet union countries many us nev
er heard georgia uzbekistan belorussia additio
```

```
n students asia africa central south america s
tudents come low income families many live she
lter far away school take long journey come sc
hool many single parent families live foster f
amily despite conditions students live show go
od attendance eager learn students eager becom
e outstanding athletes would proud students hi
ghly involved school team sports created softb
all team represent school similar schools unfo
rtunately small school no budget students team
 sports thus coach looking eagerness perform h
igh need help getting exercise mats get strong
er students involved softball team degree eage
r become best softball team citywide additiona
lly achieve high scores state tests reason nee
d build strength asking exercise mats practice
 stretching push ups crunches sit ups curl ups
 order ready perform high since students snack
 play favorite softball game concerned calorie
s intake thus need scale would monitor body we
ight fat calories practicing time tournament w
ould also use scale use end tournament thanks
dick sporting goods funded equipment upcoming
tournament supplies requested succeed march to
urnament nannan'
```

2.2 Preprocessing of project_title

In [0]:

```
# printing some random reviews
print(project_data['project_title'].values[0])
print("="*50)
print(project_data['project_title'].values[150])
print("="*50)
print(project_data['project_title'].values[1000])
```

```
print("="*50)
print(project_data['project_title'].values[2000])
print("="*50)
```

```
Engineering STEAM into the Primary Classroom
==================================================
====
Let's sit and chat about sensory!
==================================================
====
ART SUPPLIES MAKE CREATIVE CRITTERS
==================================================
====
Makers In The Class!
==================================================
====
```

In [0]:

```
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
```

In [0]:

```
sent = decontracted(project_data['project_title'].values[2000
])
print(sent)
print("="*50)
```

```
Makers In The Class!
==================================================
====
```

In [0]:

```
sent = sent.replace('\\r', ' ')
sent = sent.replace('\\"', ' ')
sent = sent.replace('\\n', ' ')
print(sent)
```

```
Makers In The Class!
```

In [0]:

```
sent = re.sub('[^A-Za-z0-9]+', ' ', sent)
print(sent)
```

```
Makers In The Class
```

In [0]:

```
# Combining all the above stundents
from tqdm import tqdm
preprocessed_essays = []
# tqdm is for printing the status bar
for sentance in tqdm(project_data['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_essays.append(sent.lower().strip())
```

```
100%|██████████| 25000/25000 [00:00<00:00, 389
78.49it/s]
```

In [0]:

```
# after preprocesing
preprocessed_essays[20000]
```

Out[0]:

```
'aiming achieve goal tournament 2017'
```

Sentiment scores

In [0]:

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

sid = SentimentIntensityAnalyzer()
S=project_data['essay']

compound = []
positive = []
neutral = []
negative = []
for x in S:
    x_comp = sid.polarity_scores(x)["compound"]
    x_pos = sid.polarity_scores(x)["pos"]
    x_neu = sid.polarity_scores(x)["neu"]
    x_neg = sid.polarity_scores(x)["neg"]
    compound.append(x_comp)
```

```
    positive.append(x_pos)
    neutral.append(x_neu)
    negative.append(x_neg)

# we can use these 4 things as features/attributes (neg, neu,
 pos, compound)
```

```
[nltk_data] Downloading package vader_lexicon
to /root/nltk_data...
```

In [0]:

```
project_data['positive']=positive
project_data['negative']=negative
project_data['neutral']=neutral
project_data['compound']=compound
```

In [0]:

```
project_data['essay_words'] = project_data['essay'].str.split
().str.len()
```

In [0]:

```
project_data['title_words'] = project_data['project_title'].s
tr.split().str.len()
```

In [0]:

```
project_data["title_essay"] = project_data["essay"].map(str)
+\
                        project_data["project_title"].map(str)
```

In [0]:

```
project_data.head(2)
```

Out[0]:

Unnamed:

	0	id	teacher_id	teacher_prefix	school
0	8393	p205479	2bf07ba08945e5d8b2a3f269b2b3cfe5	Mrs.	
10	57854	p099430	4000cfe0c8b2df75a218347c1765e283	Ms.	

In [0]:

```
print(project_data.shape)
print(project_data.columns)
```

```
(25000, 27)
Index(['Unnamed: 0', 'id', 'teacher_id', 'teac
her_prefix', 'school_state',
       'Date', 'project_grade_category', 'proj
ect_title', 'project_essay_1',
       'project_essay_2', 'project_essay_3', '
project_essay_4',
       'project_resource_summary',
       'teacher_number_of_previously_posted_pr
ojects', 'project_is_approved',
       'price', 'quantity', 'clean_categories'
, 'clean_subcategories', 'essay',
       'positive', 'negative', 'neutral', 'com
pound', 'essay_words',
       'title_words', 'title_essay'],
      dtype='object')
```

3.Splitting data into Train and cross validation(or test): Stratified Sampling

In [0]:

```
Y = project_data['project_is_approved'].values
project_data.drop(['project_is_approved'], axis=1, inplace=True)
X = project_data
```

In [0]:

```
project_data.shape
```

Out[0]:

```
(25000, 26)
```

In [0]:

```
# https://scikit-learn.org/stable/modules/generated/sklearn.model_selection.train_test_split.html
from sklearn.model_selection import train_test_split

# X_train, X_test, y_train, y_test = train_test_split(X, Y, test_size=0.33, shuffle=Flase)# this is for time series split
X_train, X_test, y_train, y_test = train_test_split(X, Y, test_size=0.33,stratify=Y) # this is random splitting
```

In [0]:

```
print(X_train.shape, y_train.shape)
print(X_test.shape, y_test.shape)
```

```
(16750, 26) (16750,)
(8250, 26) (8250,)
```

5.Catogorical features:one hot encoding

5.1 Clean_categories

In [0]:

```
vectorizer = CountVectorizer()
vectorizer.fit(X_train['clean_categories'].values) # fit has
to happen only on train data


# we use the fitted CountVectorizer to convert the text to ve
ctor
X_train_clean_category_ohe = vectorizer.transform(X_train['cl
ean_categories'].values)
X_test_clean_category_ohe = vectorizer.transform(X_test['clea
n_categories'].values)

print("After vectorizations")
print(X_train_clean_category_ohe.shape, y_train.shape)
print(X_test_clean_category_ohe.shape, y_test.shape)
print(vectorizer.get_feature_names())
```

```
After vectorizations
(16750, 9) (16750,)
(8250, 9) (8250,)
['appliedlearning', 'care_hunger', 'health_spo
rts', 'history_civics', 'literacy_language', '
math_science', 'music_arts', 'specialneeds', '
warmth']
```

5.2 Clean_subcategories

In [0]:

```
vectorizer = CountVectorizer()
vectorizer.fit(X_train['clean_subcategories'].values) # fit h
```

```
as to happen only on train data


# we use the fitted CountVectorizer to convert the text to ve
ctor
X_train_clean_subcategory_ohe = vectorizer.transform(X_train[
'clean_subcategories'].values)
X_test_clean_subcategory_ohe = vectorizer.transform(X_test['c
lean_subcategories'].values)

print("After vectorizations")
print(X_train_clean_subcategory_ohe.shape, y_train.shape)
print(X_test_clean_subcategory_ohe.shape, y_test.shape)
print(vectorizer.get_feature_names())
```

```
After vectorizations
(16750, 30) (16750,)
(8250, 30) (8250,)
['appliedsciences', 'care_hunger', 'charactere
ducation', 'civics_government', 'college_caree
rprep', 'communityservice', 'earlydevelopment'
, 'economics', 'environmentalscience', 'esl',
'extracurricular', 'financialliteracy', 'forei
gnlanguages', 'gym_fitness', 'health_lifescien
ce', 'health_wellness', 'history_geography', '
literacy', 'literature_writing', 'mathematics'
, 'music', 'nutritioneducation', 'other', 'par
entinvolvement', 'performingarts', 'socialscie
nces', 'specialneeds', 'teamsports', 'visualar
ts', 'warmth']
```

5.3 Teacher_prefix

In [0]:

```
X_train.teacher_prefix = X_train.teacher_prefix.fillna('')
```

```
X_train['teacher_prefix'].value_counts()
```

Out[0]:

```
Mrs.       8691
Ms.        6030
Mr.        1661
Teacher     367
Dr.           1
Name: teacher_prefix, dtype: int64
```

In [0]:

```
X_test.teacher_prefix = X_test.teacher_prefix.fillna('')
X_test['teacher_prefix'].value_counts()
```

Out[0]:

```
Mrs.       4396
Ms.        2921
Mr.         750
Teacher     183
Name: teacher_prefix, dtype: int64
```

In [0]:

```
vectorizer = CountVectorizer()
vectorizer.fit(X_train['teacher_prefix'].values) # fit has to
 happen only on train data

# we use the fitted CountVectorizer to convert the text to ve
ctor
X_train_teacher_ohe = vectorizer.transform(X_train['teacher_p
refix'].values)
X_test_teacher_ohe = vectorizer.transform(X_test['teacher_pre
fix'].values)

print("After vectorizations")
print(X_train_teacher_ohe.shape, y_train.shape)
print(X_test_teacher_ohe.shape, y_test.shape)
```

```
print(vectorizer.get_feature_names())
```

```
After vectorizations
(16750, 5) (16750,)
(8250, 5) (8250,)
['dr', 'mr', 'mrs', 'ms', 'teacher']
```

5.4 School_state

In [0]:

```
vectorizer = CountVectorizer()
vectorizer.fit(X_train['school_state'].values) # fit has to h
appen only on train data

# we use the fitted CountVectorizer to convert the text to ve
ctor
X_train_state_ohe = vectorizer.transform(X_train['school_stat
e'].values)
X_test_state_ohe = vectorizer.transform(X_test['school_state'
].values)

print("After vectorizations")
print(X_train_state_ohe.shape, y_train.shape)
print(X_test_state_ohe.shape, y_test.shape)
print(vectorizer.get_feature_names())
```

```
After vectorizations
(16750, 51) (16750,)
(8250, 51) (8250,)
['ak', 'al', 'ar', 'az', 'ca', 'co', 'ct', 'dc
', 'de', 'fl', 'ga', 'hi', 'ia', 'id', 'il', '
in', 'ks', 'ky', 'la', 'ma', 'md', 'me', 'mi',
 'mn', 'mo', 'ms', 'mt', 'nc', 'nd', 'ne', 'nh
', 'nj', 'nm', 'nv', 'ny', 'oh', 'ok', 'or', '
pa', 'ri', 'sc', 'sd', 'tn', 'tx', 'ut', 'va',
```

```
 'vt', 'wa', 'wi', 'wv', 'wy']
```

5.5 Project_grade_category

In [0]:

```
X_train.project_grade_category = X_train.project_grade_catego
ry.str.replace('\s+', '_')
X_train.project_grade_category = X_train.project_grade_catego
ry.str.replace('-', '_')
X_train['project_grade_category'].value_counts()
```

Out[0]:

```
Grades_PreK_2    6798
Grades_3_5       5700
Grades_6_8       2588
Grades_9_12      1664
Name: project_grade_category, dtype: int64
```

In [0]:

```
vectorizer = CountVectorizer(lowercase=False, binary=True)
vectorizer.fit(X_train['project_grade_category'].values) # fi
t has to happen only on train data

# we use the fitted CountVectorizer to convert the text to ve
ctor
X_train_grade_ohe = vectorizer.transform(X_train['project_gra
de_category'].values)
X_test_grade_ohe = vectorizer.transform(X_test['project_grade
_category'].values)

print("After vectorizations")
print(X_train_grade_ohe.shape, y_train.shape)
print(X_test_grade_ohe.shape, y_test.shape)
print(vectorizer.get_feature_names())
```

```
After vectorizations
(16750, 4) (16750,)
(8250, 4) (8250,)
['Grades_3_5', 'Grades_6_8', 'Grades_9_12', 'G
rades_PreK_2']
```

6.Numerical features

6.1 Price

In [0]:

```
from sklearn.preprocessing import StandardScaler
standard_vec = StandardScaler(with_mean = False)
# this will rise an error Expected 2D array, got 1D array ins
tead:
# array=[105.22 215.96  96.01 ... 368.98  80.53 709.67].
# Reshape your data either using
# array.reshape(-1, 1) if your data has a single feature
# array.reshape(1, -1)  if it contains a single sample.
standard_vec.fit(X_train['price'].values.reshape(-1,1))

X_train_price_std = standard_vec.transform(X_train['price'].v
alues.reshape(-1,1))
X_test_price_std = standard_vec.transform(X_test['price'].val
ues.reshape(-1,1))

print("After vectorizations")
print(X_train_price_std.shape, y_train.shape)
print(X_test_price_std.shape, y_test.shape)
```

```
After vectorizations
(16750, 1) (16750,)
(8250, 1) (8250,)
```

6.2 Teacher_number_of_previously_posted_projects

In [0]:

```
from sklearn.preprocessing import StandardScaler
standard_vec = StandardScaler(with_mean = False)
# this will rise an error Expected 2D array, got 1D array ins
tead:
# array=[105.22 215.96  96.01 ... 368.98  80.53 709.67].
# Reshape your data either using
# array.reshape(-1, 1) if your data has a single feature
# array.reshape(1, -1)  if it contains a single sample.
standard_vec.fit(X_train['teacher_number_of_previously_posted
_projects'].values.reshape(-1,1))

X_train_prev_projects_std = standard_vec.transform(X_train['t
eacher_number_of_previously_posted_projects'].values.reshape(-
1,1))
X_test_prev_projects_std = standard_vec.transform(X_test['tea
cher_number_of_previously_posted_projects'].values.reshape(-1,
1))

print("After vectorizations")
print(X_train_prev_projects_std.shape, y_train.shape)
print(X_test_prev_projects_std.shape, y_test.shape)
```

```
After vectorizations
(16750, 1) (16750,)
(8250, 1) (8250,)
```

6.3 Quantity

In [0]:

```
from sklearn.preprocessing import StandardScaler
standard_vec = StandardScaler(with_mean = False)
# this will rise an error Expected 2D array, got 1D array ins
tead:
# array=[105.22 215.96  96.01 ... 368.98  80.53 709.67].
# Reshape your data either using
# array.reshape(-1, 1) if your data has a single feature
# array.reshape(1, -1)  if it contains a single sample.
standard_vec.fit(X_train['quantity'].values.reshape(-1,1))

X_train_quantity_std = standard_vec.transform(X_train['quanti
ty'].values.reshape(-1,1))
X_test_quantity_std = standard_vec.transform(X_test['quantity
'].values.reshape(-1,1))

print("After vectorizations")
print(X_train_quantity_std.shape, y_train.shape)
print(X_test_quantity_std.shape, y_test.shape)
```

```
After vectorizations
(16750, 1) (16750,)
(8250, 1) (8250,)
```

In [0]:

```
from sklearn.preprocessing import StandardScaler
standard_vec = StandardScaler(with_mean = False)
# this will rise an error Expected 2D array, got 1D array ins
tead:
# array=[105.22 215.96  96.01 ... 368.98  80.53 709.67].
# Reshape your data either using
# array.reshape(-1, 1) if your data has a single feature
# array.reshape(1, -1)  if it contains a single sample.
standard_vec.fit(X_train['essay_words'].values.reshape(-1,1))

X_train_essay_words = standard_vec.transform(X_train['essay_w
```

```
ords'].values.reshape(-1,1))
X_test_essay_words = standard_vec.transform(X_test['essay_wor
ds'].values.reshape(-1,1))

print("After vectorizations")
print(X_train_essay_words.shape, y_train.shape)
print(X_test_essay_words.shape, y_test.shape)
```

```
After vectorizations
(16750, 1) (16750,)
(8250, 1) (8250,)
```

In [0]:

```
from sklearn.preprocessing import StandardScaler
standard_vec = StandardScaler(with_mean = False)
# this will rise an error Expected 2D array, got 1D array ins
tead:
# array=[105.22 215.96  96.01 ... 368.98  80.53 709.67].
# Reshape your data either using
# array.reshape(-1, 1) if your data has a single feature
# array.reshape(1, -1)  if it contains a single sample.
standard_vec.fit(X_train['title_words'].values.reshape(-1,1))

X_train_title_words = standard_vec.transform(X_train['title_w
ords'].values.reshape(-1,1))
X_test_title_words = standard_vec.transform(X_test['title_wor
ds'].values.reshape(-1,1))

print("After vectorizations")
print(X_train_title_words.shape, y_train.shape)
print(X_test_title_words.shape, y_test.shape)
```

```
After vectorizations
(16750, 1) (16750,)
(8250, 1) (8250,)
```

In [0]:

```
from sklearn.preprocessing import StandardScaler
standard_vec = StandardScaler(with_mean = False)
# this will rise an error Expected 2D array, got 1D array ins
tead:
# array=[105.22 215.96  96.01 ... 368.98  80.53 709.67].
# Reshape your data either using
# array.reshape(-1, 1) if your data has a single feature
# array.reshape(1, -1)  if it contains a single sample.
standard_vec.fit(X_train['negative'].values.reshape(-1,1))

X_train_negative = standard_vec.transform(X_train['negative']
.values.reshape(-1,1))
X_test_negative = standard_vec.transform(X_test['negative'].v
alues.reshape(-1,1))

print("After vectorizations")
print(X_train_negative.shape, y_train.shape)
print(X_test_negative.shape, y_test.shape)
```

```
After vectorizations
(16750, 1) (16750,)
(8250, 1) (8250,)
```

In [0]:

```
from sklearn.preprocessing import StandardScaler
standard_vec = StandardScaler(with_mean = False)
# this will rise an error Expected 2D array, got 1D array ins
tead:
# array=[105.22 215.96  96.01 ... 368.98  80.53 709.67].
# Reshape your data either using
# array.reshape(-1, 1) if your data has a single feature
# array.reshape(1, -1)  if it contains a single sample.
standard_vec.fit(X_train['positive'].values.reshape(-1,1))

X_train_positive = standard_vec.transform(X_train['positive']
.values.reshape(-1,1))
X_test_positive = standard_vec.transform(X_test['positive'].v
```

```
alues.reshape(-1,1))

print("After vectorizations")
print(X_train_positive.shape, y_train.shape)
print(X_test_positive.shape, y_test.shape)
```

```
After vectorizations
(16750, 1) (16750,)
(8250, 1) (8250,)
```

In [0]:

```
from sklearn.preprocessing import StandardScaler
standard_vec = StandardScaler(with_mean = False)
# this will rise an error Expected 2D array, got 1D array ins
tead:
# array=[105.22 215.96  96.01 ... 368.98  80.53 709.67].
# Reshape your data either using
# array.reshape(-1, 1) if your data has a single feature
# array.reshape(1, -1)  if it contains a single sample.
standard_vec.fit(X_train['neutral'].values.reshape(-1,1))

X_train_neutral = standard_vec.transform(X_train['neutral'].v
alues.reshape(-1,1))
X_test_neutral = standard_vec.transform(X_test['neutral'].val
ues.reshape(-1,1))

print("After vectorizations")
print(X_train_neutral.shape, y_train.shape)
print(X_test_neutral.shape, y_test.shape)
```

```
After vectorizations
(16750, 1) (16750,)
(8250, 1) (8250,)
```

In [0]:

```
from sklearn.preprocessing import StandardScaler
```

```
standard_vec = StandardScaler(with_mean = False)
# this will rise an error Expected 2D array, got 1D array ins
tead:
# array=[105.22 215.96  96.01 ... 368.98  80.53 709.67].
# Reshape your data either using
# array.reshape(-1, 1) if your data has a single feature
# array.reshape(1, -1)  if it contains a single sample.
standard_vec.fit(X_train['compound'].values.reshape(-1,1))

X_train_compound = standard_vec.transform(X_train['compound']
.values.reshape(-1,1))
X_test_compound = standard_vec.transform(X_test['compound'].v
alues.reshape(-1,1))

print("After vectorizations")
print(X_train_compound.shape, y_train.shape)
print(X_test_compound.shape, y_test.shape)
```

```
After vectorizations
(16750, 1) (16750,)
(8250, 1) (8250,)
```

Selecting top 2000 words from `essay` and `project_title`

In [0]:

```
from sklearn.feature_extraction.text import TfidfVectorizer
vect = TfidfVectorizer(ngram_range = (1,1) , max_features = 2
000)
tfidf = vect.fit_transform (X_train['title_essay'])

print("Shape of matrix after one hot encodig ",tfidf.shape)
```

```
Shape of matrix after one hot encodig  (16750,
 2000)
```

In [0]:

```
top_features= vect.get_feature_names()
```

In [0]:

```
print(len(top_features))
```

```
2000
```

Computing Co-occurance matrix

In [0]:

```
from tqdm import tqdm
n_neighbor = 5
occ_matrix = np.zeros((2000,2000))
for row in tqdm(X_train["title_essay"].values):
    words_in_row = row.split()
    for index,word in enumerate(words_in_row):
        if word in top_features:
            for j in range(max(index-n_neighbor,0),min(index+
n_neighbor,len(words_in_row)-1) + 1):
                if words_in_row[j] in top_features:
                    occ_matrix[top_features.index(word),top_f
eatures.index(words_in_row[j])] += 1
                else:
                    continue
        else:
            continue
```

```
100%|██████████| 16750/16750 [23:41<00:00, 11.
78it/s]
```

In [0]:

```
occ_matrix
```

Out[0]:

```
array([[3.550e+02, 1.000e+00, 1.000e+01, ...,
1.000e+00, 0.000e+00,
        0.000e+00],
       [1.000e+00, 2.510e+02, 0.000e+00, ...,
0.000e+00, 0.000e+00,
        0.000e+00],
       [1.000e+01, 0.000e+00, 8.800e+01, ...,
2.000e+00, 0.000e+00,
        0.000e+00],
       ...,
       [1.000e+00, 0.000e+00, 2.000e+00, ...,
1.834e+03, 0.000e+00,
        7.000e+00],
       [0.000e+00, 0.000e+00, 0.000e+00, ...,
0.000e+00, 2.210e+02,
        2.000e+00],
       [0.000e+00, 0.000e+00, 0.000e+00, ...,
7.000e+00, 2.000e+00,
        2.862e+03]])
```

In [0]:

```
print(occ_matrix.shape)
```

```
(2000, 2000)
```

Apply Truncated SVD

In [0]:

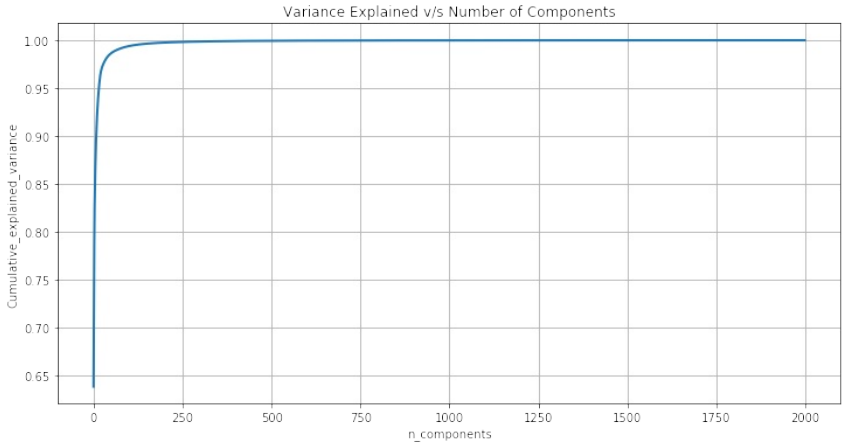
```
from sklearn.decomposition import TruncatedSVD
from sklearn.preprocessing import StandardScaler
svd = TruncatedSVD(n_components = 1999)
svd_2000 = svd.fit_transform(occ_matrix)
```

In [0]:

```
#cum=np.cumsum(svd.explained_variance_ratio_)
percentage_var_explained = svd.explained_variance_ / np.sum(s
vd.explained_variance_);
cum_var_explained = np.cumsum(percentage_var_explained)
```

In [0]:

```
plt.figure(figsize=(12,6))
plt.clf()
plt.plot(cum_var_explained, linewidth=2)
plt.axis('tight')
plt.grid()
plt.xlabel('n_components')
plt.ylabel('Cumulative_explained_variance')
plt.title("Variance Explained v/s Number of Components")
plt.show()
```



In [0]:

```
svd = TruncatedSVD(n_components = 150)
svd_2000 = svd.fit_transform(occ_matrix)
```

In [0]:

```
# Vectorizing the essay text and project titles using trucate
```

```
d svd.
essay_train = []; # the avg-w2v for each sentence/review is s
tored in this list
for sentence in tqdm(X_train['essay']): # for each review/sen
tence
    vector = np.zeros(150) # as word vectors are of zero leng
th
    cnt_words =0; # num of words with a valid vector in the s
entence/review
    for word in sentence.split(): # for each word in a review
/sentence
        if word in top_features:
            ind=top_features.index(word)
            vector += svd_2000[ind]
            cnt_words += 1
    if cnt_words != 0:
        vector /= cnt_words
    essay_train.append(vector)

print(len(essay_train))
print(len(essay_train[0]))
```

```
100%|██████████| 16750/16750 [02:00<00:00, 139
.44it/s]
```

```
16750
150
```

In [0]:

```
# Vectorizing the essay text and project titles using trucate
d svd.
essay_test = []; # the avg-w2v for each sentence/review is st
ored in this list
for sentence in tqdm(X_test['essay']): # for each review/sent
ence
    vector = np.zeros(150) # as word vectors are of zero leng
```

```
th
    cnt_words =0; # num of words with a valid vector in the s
entence/review
    for word in sentence.split(): # for each word in a review
/sentence
        if word in top_features:
            ind=top_features.index(word)
            vector += svd_2000[ind]
            cnt_words += 1
    if cnt_words != 0:
        vector /= cnt_words
    essay_test.append(vector)

print(len(essay_test))
print(len(essay_test[0]))
```

```
100%|██████████| 8250/8250 [00:58<00:00, 139.9
3it/s]
```

```
8250
150
```

In [0]:

```
# Vectorizing the essay text and project titles using trucate
d svd.
title_train = []; # the avg-w2v for each sentence/review is s
tored in this list
for sentence in tqdm(X_train['project_title']): # for each re
view/sentence
    vector = np.zeros(150) # as word vectors are of zero leng
th
    cnt_words =0; # num of words with a valid vector in the s
entence/review
    for word in sentence.split(): # for each word in a review
/sentence
        if word in top_features:
```

```
            ind=top_features.index(word)
            vector += svd_2000[ind]
            cnt_words += 1
    if cnt_words != 0:
        vector /= cnt_words
    title_train.append(vector)

print(len(title_train))
print(len(title_train[0]))
```

```
100%|██████████| 16750/16750 [00:02<00:00, 707
7.56it/s]
```

```
16750
150
```

In [0]:

```
# Vectorizing the essay text and project titles using trucate
d svd.
title_test = []; # the avg-w2v for each sentence/review is st
ored in this list
for sentence in tqdm(X_test['project_title']): # for each rev
iew/sentence
    vector = np.zeros(150) # as word vectors are of zero leng
th
    cnt_words =0; # num of words with a valid vector in the s
entence/review
    for word in sentence.split(): # for each word in a review
/sentence
        if word in top_features:
            ind=top_features.index(word)
            vector += svd_2000[ind]
            cnt_words += 1
    if cnt_words != 0:
        vector /= cnt_words
    title_test.append(vector)
```

```
print(len(title_test))
print(len(title_test[0]))
```

```
100%|██████████| 8250/8250 [00:01<00:00, 7335.
84it/s]
```

```
8250
150
```

In [0]:

```
# merge two sparse matrices: https://stackoverflow.com/a/1971
0648/4084039
from scipy.sparse import hstack
X_tr = hstack((X_train_essay_words,X_train_title_words,X_trai
n_positive,X_train_neutral,X_train_compound,X_train_clean_cat
egory_ohe,X_train_clean_subcategory_ohe, X_train_state_ohe, X
_train_teacher_ohe, X_train_grade_ohe, X_train_price_std,X_tr
ain_prev_projects_std,X_train_quantity_std,essay_train,title_
train)).tocsr()
X_te = hstack((X_test_essay_words,X_test_title_words,X_test_p
ositive,X_test_neutral,X_test_compound,X_test_clean_category_
ohe,X_test_clean_subcategory_ohe, X_test_state_ohe, X_test_te
acher_ohe, X_test_grade_ohe, X_test_price_std,X_test_prev_pro
jects_std,X_test_quantity_std,essay_test,title_test)).tocsr()

print("Final Data matrix")
print(X_tr.shape, y_train.shape)
print(X_te.shape, y_test.shape)
```

```
Final Data matrix
(16750, 407) (16750,)
(8250, 407) (8250,)
```

In [0]:

```
def batch_predict(clf, data):
```

```
    # roc_auc_score(y_true, y_score) the 2nd parameter should
 be probability estimates of the positive class
    # not the predicted outputs

    y_data_pred = []
    tr_loop = data.shape[0] - data.shape[0]%1000
    # consider you X_tr shape is 49041, then your cr_loop wil
l be 49041 - 49041%1000 = 49000
    # in this for loop we will iterate unti the last 1000 mul
tiplier
    for i in range(0, tr_loop, 1000):
        y_data_pred.extend(clf.predict_proba(data[i:i+1000])[
:,1])
    # we will be predicting for the last data points
    y_data_pred.extend(clf.predict_proba(data[tr_loop:])[:,1]
)

    return y_data_pred
```

In [0]:

```
from xgboost import XGBClassifier
```

In [0]:

```
from sklearn.model_selection import RandomizedSearchCV

XG = XGBClassifier()
params = {'n_estimators': [50, 100, 200, 250], 'max_depth':[1
, 5, 10, 50, 100]}

grid = RandomizedSearchCV(XG , params, cv = 3, scoring = 'roc
_auc', random_state = 0)
grid.fit(X_tr,y_train)
print(grid.best_params_)
```

```
{'n_estimators': 200, 'max_depth': 1}
```

In [0]:

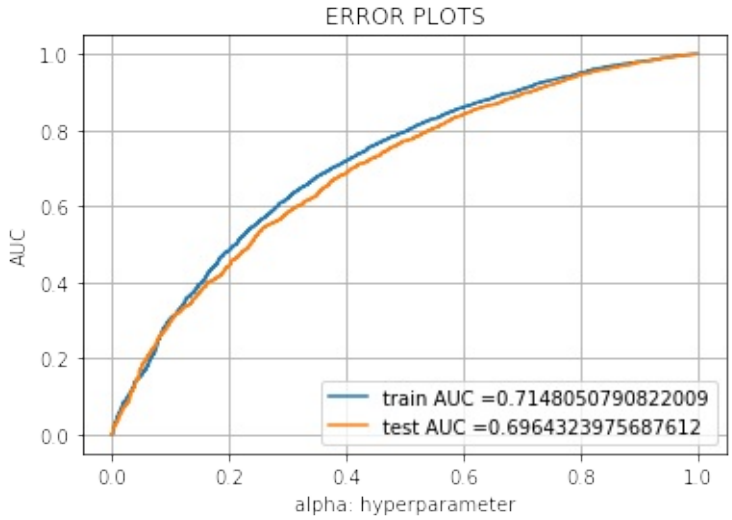
```
# https://scikit-learn.org/stable/modules/generated/sklearn.m
etrics.roc_curve.html#sklearn.metrics.roc_curve
from sklearn.metrics import roc_curve, auc


XG = XGBClassifier(max_depth = 1, n_estimators = 100)
XG.fit(X_tr, y_train)
# roc_auc_score(y_true, y_score) the 2nd parameter should be
probability estimates of the positive class
# not the predicted outputs


#https://datascience.stackexchange.com/questions/18374/predic
ting-probability-from-scikit-learn-svc-decision-function-with
-
#decision-fun/18375#18375
y_train_pred = batch_predict(XG,X_tr)
y_test_pred = batch_predict(XG,X_te)

train_fpr, train_tpr, tr_thresholds = roc_curve(y_train, y_tr
ain_pred)
test_fpr, test_tpr, te_thresholds = roc_curve(y_test, y_test_
pred)

plt.plot(train_fpr, train_tpr, label="train AUC ="+str(auc(tr
ain_fpr, train_tpr)))
plt.plot(test_fpr, test_tpr, label="test AUC ="+str(auc(test_
fpr, test_tpr)))
plt.legend()
plt.xlabel("alpha: hyperparameter")
plt.ylabel("AUC")
plt.title("ERROR PLOTS")
plt.grid()
plt.show()
```



Conclusion:

1.Formed data matrix with word vectors formed by truncated SVD (100 dimensional representation of each word).
2.Applied XGBoost on final data matrix and appied hyper parameter tuning on n-estimators and max depth giving max AUC value.