DonorsChoose

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

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

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

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

About the DonorsChoose Data Set

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

Feature	Description
`project_id`	A unique identifier for the proposed project. **Example:** `p036502`
`project_title`	Title of the project. **Examples:** • `Art Will Make You Happy!` • `First Grade Fun`
`project_grade_category`	Grade level of students for which the project is targeted. One of the following enumerated values: • `Grades PreK-2` • `Grades 3-5` • `Grades 6-8` • `Grades 9-12`
`project_subject_categories`	One or more (comma-separated) subject categories for the project from the following enumerated list of values: • `Applied Learning` • `Care & Hunger` • `Health & Sports` • `History & Civics` • `Literacy & Language` • `Math & Science` • `Music & The Arts` • `Special Needs` • `Warmth` **Examples:** • `Music & The Arts` • `Literacy & Language, Math & Science`
`school_state`	State where school is located (Two-letter U.S. postal code). **Example:** `WY`
`project_subject_subcategories`	One or more (comma-separated) subject subcategories for the project. **Examples:** • `Literacy`

- Literature & Writing, Social Sciences

Feature	Description
project_resource_summary	An explanation of the resources needed for the project. **Example:** • `My students need hands on literacy materials to manage sensory needs!`
project_essay_1	First application essay*
project_essay_2	Second application essay*
project_essay_3	Third application essay*
project_essay_4	Fourth application essay*
project_submitted_datetime	Datetime when project application was submitted. **Example:** `2016-04-28 12:43:56.245`
teacher_id	A unique identifier for the teacher of the proposed project. **Example:** `bdf8baa8fedef6bfeec7ae4ff1c15c56`
teacher_prefix	Teacher's title. One of the following enumerated values: • `nan` • `Dr.` • `Mr.` • `Mrs.` • `Ms.` • `Teacher.`
teacher_number_of_previously_posted_projects	Number of project applications previously submitted by the same teacher. **Example:** `2`

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

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

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

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

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

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

Notes on the Essay Data

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

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

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

- __project_essay_1:__ "Describe your students: What makes your students special? Specific details about their background, your neighborhood, and your school are all helpful."

- __project_essay_2:__ "About your project: How will these materials make a difference in your students' learning and improve their school lives?"

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

In [47]:

```
%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
```

1.1 Reading Data

In [48]:

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

In [49]:

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

```
Number of data points in train data (109248, 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']
```

In [50]:

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

```
Number of data points in train data (1541272, 4)
['id' 'description' 'quantity' 'price']
```

Out[50]:

	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 [51]:

```
price_data = resource_data.groupby('id').agg({'price':'sum', 'quantity':'sum'}).reset_index()
project_data = pd.merge(project_data, price_data, on='id', how='left')
```

In [52]:

```
project_data=project_data.sample(n=40000)
```

In [53]:

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

```
Number of data points in train data (40000, 19)
--------------------------------------------------
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'
 'price' 'quantity']
```

1.2 preprocessing of `project_subject_categories`

In [54]:

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

# 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 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 & H
unger"]
        if 'The' in j.split(): # this will split each of the catogory based on space "Math & Scienc
e"=> "Math","&", "Science"
            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 ' '(space) 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, inplace=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]))
```

1.3 preprocessing of `project_subject_subcategories`

In [55]:

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

# 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", "Care & H
unger"]
        if 'The' in j.split(): # this will split each of the catogory based on space "Math & Scienc
e"=> "Math","&", "Science"
            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 ' '(space) 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://stackoverflow.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=lambda kv: kv[1]))
```

1.3 Text preprocessing

In [56]:

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

In [57]:

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

Out[57]:

	Unnamed: 0	id	teacher_id	teacher_prefix	school_state	project_submitted_datetime

	Unnamed: 0	id	teacher_id	teacher_prefix	school_state	project_submitted_datetime
61625	133125	p018173	09be9df34ac1b65fd781200175dd8ce2	Mrs.	ND	2016-05-29 10:53:48
72393	61495	p176237	043b4eafa170c7e0ada00f9fd03bc940	Mrs.	RI	2016-10-05 11:53:28

In [58]:

```
#### 1.4.2.3 Using Pretrained Models: TFIDF weighted W2V
```

In [59]:

```
# 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)
```

```
My students LOVE to read on a daily basis. However, they sometimes struggle to find books at their
individual reading and interest level in my classroom library. They get really bummed out if we ev
er have to cut our independent reading time shorter. \r\n\r\nOur small rural school provides stude
nts with a family feeling as our students work together daily to reach our monthly goal of one mil
lion words read. They get a big kick out of watching our word count grow daily.\r\nThis project wa
s requested by my 4-6 reading class. They have all read every Dan Gutman book in my class library.
They asked me to get more over the summer. How could I possibly return to school in the fall witho
ut their books? This project will show them how much I value their input in our class. They will l
earn they have a voice and they need to use it often in order to get their needs met.\r\n\r\nMy st
udents will develop their love of reading with Dan Gutman books they actually requested
themselves. The classroom library will reflect the students' interests if this project is fully fu
nded.nannan
==================================================
My classroom is an Inclusion setting of 3rd graders. 82 percent poverty, and high needs. Our stude
nts are from broken homes, living with foster and grandparents. I am looking to make my classroom
a safe, warm, loving environment. When a child feels safe and connected then they can learn. When
a child has choices that don't hurt them, they will build positive brain pathways and
responsibilities.  I look to build relationships and making them feel safe and loved so they will
want to be there and want to learn.Safe Outstanding Effort Always respectful and Responsible is wh
at we work toward in our class. I believe having the choice and comfort of seating will assist us
in reaching our goals. There are many research studies on having options helping studies. There is
also a sensory and movement side of the seats listed in this project.  These seats will be used in
my classroom this year, and many years to come. We also will use them during the small group time
when our students switch to different classrooms. Thank you for your consideration on this project
. Thank you very much.nannan
==================================================
My students are eager learners that connects them to the STEM world!  They take pride in who they
are, and are working hard to achieve their goals. They are full of character and curiosity. \r\n\r
\nThese students are resilient and deserve the support that they need to help them attain their go
als, regardless of their socioeconomic status. My biggest goal for them is to answer \"when will w
e use math in the real world\". My biggest challenge is that I do not have consistent availability
of technology or other resources, besides paper, pencil, textbook, to spark an interest in
mathematics.\r\nThe days of paper and pencil math are over. Students have become far less engaged
in old traditional methods of learning math. They are growing up in different world where
information is at a touch of a button. When trying to bridge the gap between learning in a digital
world versus learning with paper and pencil, it is best for each student to have a Chromebook of t
heir own in order to be fully engaged in the lesson. We share Chromebooks throughout our school, a
nd some days we have four or more students sharing one Chromebook. The lesson is not as effective
as it could be when sharing the Chromebooks, and I want my students to be fully engage in 21st cen
tury learning by bridging the gap between technology and math lessons.\r\nChromebooks in my classr
oom will make a significant difference in engaging my students in STEM education and exploration.
Modern mathematics is the foundation tool for advancements in engineering, technology and science.
To best prepare my students for their future careers, I must integrate technology into my daily ma
```

```
th curriculum through the use of Chromebooks. Students will be able to utilize this technology to
research, communicate, and educate others about STEM.\r\nnannan
==================================================
Benjamin Franklin once said, \"Tell me and I forget. Teach me and I remember.  Involve me and I le
arn.\"  Students need to be involved in their learning so that they can take ownership of what the
y are doing. Students need to get their hands \"dirty\" by using them to research, create and make
learning meaningful. I will teach about 22 eager, energetic 3rd grade students in my classroom thi
s year.  We are a Title I school with 100% participation in the reduced price/free lunch program.
Many of our students have never left the community nor experienced anything outside of the area. \
r\n\r\nAll of our students are eager to learn and need experiences that they do not get at home. W
e do not have much parental involvement at our school and we are looking to change that. We want p
arents to associate school with love, learning and caring. We would like to change parents'
points-of-view from the negative experiences they may have had as a child, to be able to see the p
ositive experiences we provide their children.\r\nThird grade can be very taxing on our students.
The children spend most of their time testing, to make a pathway that will allow them to go to
fourth grade. Students spend most the year stressed and anxious.\r\n\r\nI want to include more exc
iting, educational experiences for my students that will help them remember learning is fun.\r\nI
want to empower my students through art by having them create throughout the year about topics we
have learned. I also want to teach my students to create animations through computer programming,
using websites such as Scratch.\r\n\r\nOur school has a set number of computers, which 18 classes
have to share. We are only allotted a short amount of time on the computers each day, and with tha
t time we have to go on a remediation website.nannan
==================================================
```

In [60]:

```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 [61]:

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

```
Benjamin Franklin once said, \"Tell me and I forget. Teach me and I remember.  Involve me and I le
arn.\"  Students need to be involved in their learning so that they can take ownership of what the
y are doing. Students need to get their hands \"dirty\" by using them to research, create and make
learning meaningful. I will teach about 22 eager, energetic 3rd grade students in my classroom thi
s year.  We are a Title I school with 100% participation in the reduced price/free lunch program.
Many of our students have never left the community nor experienced anything outside of the area. \
r\n\r\nAll of our students are eager to learn and need experiences that they do not get at home. W
e do not have much parental involvement at our school and we are looking to change that. We want p
arents to associate school with love, learning and caring. We would like to change parents'
points-of-view from the negative experiences they may have had as a child, to be able to see the p
ositive experiences we provide their children.\r\nThird grade can be very taxing on our students.
The children spend most of their time testing, to make a pathway that will allow them to go to
fourth grade. Students spend most the year stressed and anxious.\r\n\r\nI want to include more exc
iting, educational experiences for my students that will help them remember learning is fun.\r\nI
want to empower my students through art by having them create throughout the year about topics we
have learned. I also want to teach my students to create animations through computer programming,
using websites such as Scratch.\r\n\r\nOur school has a set number of computers, which 18 classes
have to share. We are only allotted a short amount of time on the computers each day, and with tha
t time we have to go on a remediation website.nannan
==================================================
```

In [62]:

```
# \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)
```

```
Benjamin Franklin once said,  Tell me and I forget. Teach me and I remember.  Involve me and I lea
rn.   Students need to be involved in their learning so that they can take ownership of what they
are doing. Students need to get their hands  dirty  by using them to research, create and make
learning meaningful. I will teach about 22 eager, energetic 3rd grade students in my classroom thi
s year.  We are a Title I school with 100% participation in the reduced price/free lunch program.
Many of our students have never left the community nor experienced anything outside of the area.
All of our students are eager to learn and need experiences that they do not get at home. We do no
t have much parental involvement at our school and we are looking to change that. We want parents
to associate school with love, learning and caring. We would like to change parents' points-of-vie
w from the negative experiences they may have had as a child, to be able to see the positive exper
iences we provide their children.  Third grade can be very taxing on our students. The children sp
end most of their time testing, to make a pathway that will allow them to go to fourth grade.
Students spend most the year stressed and anxious.    I want to include more exciting, educational
experiences for my students that will help them remember learning is fun.  I want to empower my st
udents through art by having them create throughout the year about topics we have learned. I also
want to teach my students to create animations through computer programming, using websites such a
s Scratch.    Our school has a set number of computers, which 18 classes have to share. We are onl
y allotted a short amount of time on the computers each day, and with that time we have to go on a
remediation website.nannan
```

In [63]:

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

```
Benjamin Franklin once said Tell me and I forget Teach me and I remember Involve me and I learn St
udents need to be involved in their learning so that they can take ownership of what they are doin
g Students need to get their hands dirty by using them to research create and make learning
meaningful I will teach about 22 eager energetic 3rd grade students in my classroom this year We a
re a Title I school with 100 participation in the reduced price free lunch program Many of our stu
dents have never left the community nor experienced anything outside of the area All of our studen
ts are eager to learn and need experiences that they do not get at home We do not have much parent
al involvement at our school and we are looking to change that We want parents to associate school
with love learning and caring We would like to change parents points of view from the negative exp
eriences they may have had as a child to be able to see the positive experiences we provide their
children Third grade can be very taxing on our students The children spend most of their time test
ing to make a pathway that will allow them to go to fourth grade Students spend most the year
stressed and anxious I want to include more exciting educational experiences for my students that
will help them remember learning is fun I want to empower my students through art by having them c
reate throughout the year about topics we have learned I also want to teach my students to create
animations through computer programming using websites such as Scratch Our school has a set number
of computers which 18 classes have to share We are only allotted a short amount of time on the com
puters each day and with that time we have to go on a remediation website nannan
```

In [64]:

```
# 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', 'between', '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', 'e
ach', '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', "couldn't", 'didn', "didn't", 'doesn', "do
esn'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 [65]:

```
# 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 not in stopwords)
    preprocessed_essays.append(sent.lower().strip())
```

```
100%|████████████████████████████████████████████████████████████████████████| 40000/40000 [00:
53<00:00, 743.31it/s]
```

In [66]:

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

Out[66]:

```
'benjamin franklin said tell i forget teach i remember involve i learn students need involved lear
ning take ownership students need get hands dirty using research create make learning meaningful i
teach 22 eager energetic 3rd grade students classroom year we title i school 100 participation red
uced price free lunch program many students never left community nor experienced anything outside
area all students eager learn need experiences not get home we not much parental involvement schoo
l looking change we want parents associate school love learning caring we would like change
parents points view negative experiences may child able see positive experiences provide children
third grade taxing students the children spend time testing make pathway allow go fourth grade stu
dents spend year stressed anxious i want include exciting educational experiences students help re
member learning fun i want empower students art create throughout year topics learned i also want
teach students create animations computer programming using websites scratch our school set number
computers 18 classes share we allotted short amount time computers day time go remediation website
nannan'
```

In [67]:

```
project_data['clean_essays'] = preprocessed_essays
project_data.drop(['essay'], axis=1, inplace=True)
```

1.4 Preprocessing of `project_title`

In [68]:

```
# similarly you can preprocess the titles also
```

In [69]:

```
from tqdm import tqdm
preprocessed_titles = []
# 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_titles.append(sent.lower().strip())
```

```
100%|██████████████████████████████████████████████████████████████████████████| 40000/40000
[00:02<00:00, 15385.74it/s]
```

In [70]:

```
project_data['clean_project_titles'] = preprocessed_titles
project_data.drop(['project_title'], axis=1, inplace=True)
project_data.head(2)
```

Out[70]:

	Unnamed: 0	id	teacher_id	teacher_prefix	school_state	project_submitted_datetime
61625	133125	p018173	09be9df34ac1b65fd781200175dd8ce2	Mrs.	ND	2016-05-29 10:53:48
72393	61495	p176237	043b4eafa170c7e0ada00f9fd03bc940	Mrs.	RI	2016-10-05 11:53:28

In [71]:

```
#https://planspace.org/20150607-textblob_sentiment/
#https://stackoverflow.com/questions/43485469/apply-textblob-in-for-each-row-of-a-dataframe
def cal_sentiment_polarity(inputString):
    try:
        return TextBlob(inputString).sentiment.polarity
    except:
        return 0
```

1.4.1 Calculating Sentiment score for essays`

In [72]:

```
project_data["essay_sentiment"]=project_data["clean_essays"].apply(cal_sentiment_polarity)
```

1.4.2 Calculating word counts

In [73]:

```
import re
def count_words(inputString):
    try:
        return len(re.findall(r'\w+', inputString))
    except:
        return None
```

In [74]:

```
project_data["title_word_count"]=project_data["clean_project_titles"].apply(count_words)
```

In [75]:

```
project_data["combine_essay_word_count"]=project_data["clean_essays"].apply(count_words)
```

1.5 Preparing data for models

In [76]:

```
project_data.columns
```

Out[76]:

```
Index(['Unnamed: 0', 'id', 'teacher_id', 'teacher_prefix', 'school_state',
       'project_submitted_datetime', 'project_grade_category',
       'project_essay_1', 'project_essay_2', 'project_essay_3',
       'project_essay_4', 'project_resource_summary',
       'teacher_number_of_previously_posted_projects', 'project_is_approved',
       'price', 'quantity', 'clean_categories', 'clean_subcategories',
       'clean_essays', 'clean_project_titles', 'essay_sentiment',
       'title_word_count', 'combine_essay_word_count'],
      dtype='object')
```

we are going to consider

```
- school_state : categorical data
- clean_categories : categorical data
- clean_subcategories : categorical data
- project_grade_category : categorical data
- teacher_prefix : categorical data

- project_title : text data
- text : text data
- project_resource_summary: text data (optinal)

- quantity : numerical (optinal)
- teacher_number_of_previously_posted_projects : numerical
- price : numerical
```

2. TruncatedSVD

Assignment 11: TruncatedSVD

- step 1 Select the top 2k words from essay text and project_title (concatinate essay text with project title and then find the top 2k words) based on their `idf_` values
- step 2 Compute the co-occurance matrix with these 2k words, with window size=5 (ref)

- step 3 Use TruncatedSVD on calculated co-occurance matrix and reduce its dimensions, choose the number of components (`n_components`) using elbow method

  > - The shape of the matrix after TruncatedSVD will be 2000*n, i.e. each row represents a vector form of the corresponding word.
  > - Vectorize the essay text and project titles using these word vectors. (while vectorizing, do ignore all the words which are not in top 2k words)

- step 4 Concatenate these truncatedSVD matrix, with the matrix with features
  - **school_state** : categorical data
  - **clean_categories** : categorical data
  - **clean_subcategories** : categorical data
  - **project_grade_category** :categorical data
  - **teacher_prefix** : categorical data
  - **quantity** : numerical data
  - **teacher_number_of_previously_posted_projects** : numerical data
  - **price** : numerical data

    - **sentiment score's of each of the essay** : numerical data
    - **number of words in the title** : numerical data
    - **number of words in the combine essays** : numerical data
    - **word vectors calculated in** step 3 : numerical data
- step 5: Apply GBDT on matrix that was formed in step 4 of this assignment, **DO REFER THIS BLOG: XGBOOST DMATRIX**
- **step 6:Hyper parameter tuning (Consider any two hyper parameters)**
    - **Find the best hyper parameter which will give the maximum AUC value**
    - **Find the best hyper paramter using k-fold cross validation or simple cross validation data**
    - **Use gridsearch cv or randomsearch cv or you can also write your own for loops to do this task of hyperparameter tuning**

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

In [77]:

```
# please write all the code with proper documentation, and proper titles for each subsection
# go through documentations and blogs before you start coding
# first figure out what to do, and then think about how to do.
# reading and understanding error messages will be very much helpfull in debugging your code
# when you plot any graph make sure you use
    # a. Title, that describes your plot, this will be very helpful to the reader
    # b. Legends if needed
    # c. X-axis label
    # d. Y-axis label
```

In [78]:

```
project_data['essay_and_title'] = project_data['clean_essays'] +' '+
project_data['clean_project_titles']
```

In [79]:

```
tfidf_model = TfidfVectorizer()
tfidf_model.fit_transform(project_data['essay_and_title'].values)
indices = np.argsort(tfidf_model.idf_)[::-1]
features = tfidf_model.get_feature_names()
top_n = 2000
top_words = [features[i] for i in indices[:top_n]]
```

2.2 Computing Co-occurance matrix

In [80]:

```
# please write all the code with proper documentation, and proper titles for each subsection
# go through documentations and blogs before you start coding
# first figure out what to do, and then think about how to do.
# reading and understanding error messages will be very much helpfull in debugging your code
# make sure you featurize train and test data separatly

# when you plot any graph make sure you use
    # a. Title, that describes your plot, this will be very helpful to the reader
    # b. Legends if needed
    # c. X-axis label
    # d. Y-axis label
```

In [81]:

```
#https://datascience.stackexchange.com/questions/40038/how-to-implement-word-to-word-co-occurence-
matrix-in-python
def Co_Occurance_Matrix(X_Text, Imp_Feat):

    print(" Co_Occurance Matrix ")
    # n X n matrix with initially value = 0.
    array = np.array([[0 for x in range(Imp_Feat)] for x in range(Imp_Feat)])

    df = pd.DataFrame(array, index=top_words, columns=top_words)

    for sent in tqdm(project_data['essay_and_title'].values):
        #Words splitting
        words = sent.split(" ")

        for word in range(len(words)):
            # neigh range (1 to 5)
            for neigh in range(1,6):

                if(word + neigh  < len(words) and words[word] != words[neigh]):

                    try:
                       #print("ram")
                        df.loc[words[word], words[neigh]] += 1


                        df.loc[words[neigh], words[word]] += 1

                    except:
                        pass


    print(df.shape)

    return df
```

In [82]:

```
import feather
if os.path.isfile('CoOccurance.feather'):
    df=pd.read_feather('CoOccurance.feather')
else:
    df=Co_Occurance_Matrix(project_data,top_n)
    feather.write_dataframe(df, 'CoOccurance.feather')
```

2.3 Applying TruncatedSVD and Calculating Vectors for `essay` and `project_title`

In [83]:

```
# please write all the code with proper documentation, and proper titles for each subsection
# go through documentations and blogs before you start coding
# first figure out what to do, and then think about how to do.
# reading and understanding error messages will be very much helpfull in debugging your code
# make sure you featurize train and test data separatly

# when you plot any graph make sure you use
    # a. Title, that describes your plot, this will be very helpful to the reader
    # b. Legends if needed
    # c. X-axis label
    # d. Y-axis label
```

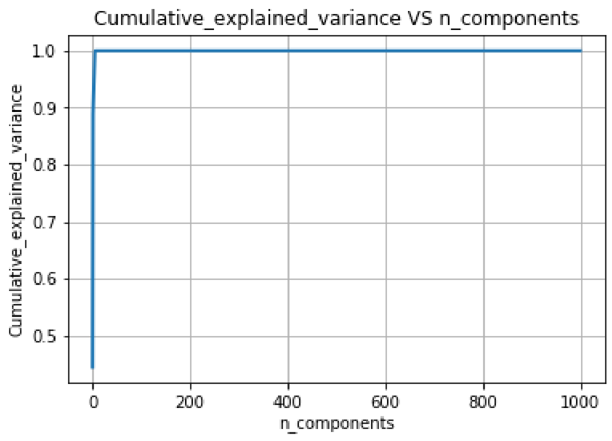
In [84]:

```
from sklearn.decomposition import TruncatedSVD
def SVD_Truncated(coo_matrix, no_of_comp):
    global Max_svd

    MaxVar = -1 # Max Explained varience

    Max_svd = 0 # initially 0

    #To get SVD with Max Explained varience

    for n_comp in no_of_comp:

        svd_matrix = TruncatedSVD(n_components=n_comp)
        svd=svd_matrix.fit(coo_matrix)
        exp_sum = svd.explained_variance_ratio_.sum()

        if exp_sum >  MaxVar :
            Max_svd = svd
            MaxVar  = exp_sum
    print("MaxExp==" ,MaxVar )
    percentage_var_explained =  Max_svd .explained_variance_ / np.sum( Max_svd .explained_variance_
)
    cum_var_explained = np.cumsum(percentage_var_explained)

    # Plotting for  MaxExp value in  list_component
    plt.plot(cum_var_explained , linewidth=2)
    plt.grid()
    plt.xlabel('n_components')
    plt.ylabel('Cumulative_explained_variance')
    plt.title("Cumulative_explained_variance VS n_components")
    plt.show()

    Max_svd=svd.transform(coo_matrix)
```

In [85]:

```
SVD_Truncated(df,[100,500,800,1000,1200,1500])
```

```
MaxExp== 1.0000000000000528
```

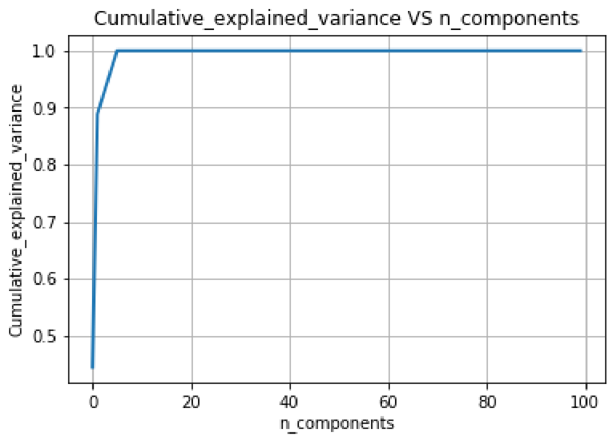


For n_components =100 the variance explained is maximum of 1.000%.So we can use n_components as 100 in further stages.

In [86]:

```
SVD_Truncated(df,[100])
```

```
MaxExp== 0.999999999999967
```



In [87]:

```
#https://stackoverflow.com/questions/50915223/how-to-get-the-vector-representation-of-a-word-using
-a-trained-svd-model
#https://www.analyticsvidhya.com/blog/2018/10/stepwise-guide-topic-modeling-latent-semantic-analys
```

```
is/
#https://www.kaggle.com/dex314/tfidf-truncatedsvd-and-light-gbm
```

In [88]:

```
from sklearn import model_selection
X=project_data.drop(['project_is_approved'],axis=1)
y=project_data[['project_is_approved']]
X_tr, X_test, y_tr, y_test = model_selection.train_test_split(X, y, test_size=0.2, random_state=0,s
tratify=y)
```

In [89]:

```
print(X_tr.shape, y_tr.shape)
print(X_test.shape, y_test.shape)
```

```
(32000, 23) (32000, 1)
(8000, 23) (8000, 1)
```

In [90]:

```
tfidf = TfidfVectorizer(min_df=5, max_features=top_n)
tfidf.fit(X_tr['clean_essays'].values)
svdT = TruncatedSVD(n_components=100)
text_tfidf_tr = svdT.fit_transform(tfidf.transform(X_tr['clean_essays'].values))
text_tfidf_test= svdT.fit_transform(tfidf.transform(X_test['clean_essays'].values))
print(text_tfidf_tr.shape,text_tfidf_test.shape)
```

```
(32000, 100) (8000, 100)
```

In [91]:

```
tfidf = TfidfVectorizer(min_df=5, max_features=top_n)
tfidf.fit(X_tr['clean_project_titles'].values)
svdT = TruncatedSVD(n_components=100)
title_tfidf_tr = svdT.fit_transform(tfidf.transform(X_tr['clean_project_titles'].values))
title_tfidf_test = svdT.fit_transform(tfidf.transform(X_test['clean_project_titles'].values))
print(title_tfidf_tr.shape,title_tfidf_test.shape)
```

```
(32000, 100) (8000, 100)
```

2.4 Merge the features from step 3 and step 4

In [92]:

```
# please write all the code with proper documentation, and proper titles for each subsection
# go through documentations and blogs before you start coding
# first figure out what to do, and then think about how to do.
# reading and understanding error messages will be very much helpfull in debugging your code
# when you plot any graph make sure you use
    # a. Title, that describes your plot, this will be very helpful to the reader
    # b. Legends if needed
    # c. X-axis label
    # d. Y-axis label
```

1.5.1 Vectorizing Categorical data

- https://www.appliedaicourse.com/course/applied-ai-course-online/lessons/handling-categorical-and-numerical-features/

In [93]:

```
from sklearn.feature_extraction.text import CountVectorizer
vectorizer = CountVectorizer(min_df=10)
categories_one_hot_tr = vectorizer.fit_transform(X_tr['clean_categories'].values)
categories_one_hot_test = vectorizer.transform(X_test['clean_categories'].values)
categories_one_hot_test = vectorizer.transform(X_test['clean_categories'].values)
```

In [94]:

```
school_state_one_hot_tr = vectorizer.fit_transform(X_tr['school_state'].values)
school_state_one_hot_test = vectorizer.transform(X_test['school_state'].values)
```

In [95]:

```
# you can do the similar thing with state, teacher_prefix and project_grade_category also
```

In [96]:

```
sub_categories_one_hot_tr = vectorizer.fit_transform(X_tr['clean_subcategories'].values)
sub_categories_one_hot_test = vectorizer.transform(X_test['clean_subcategories'].values)
```

In [97]:

```
teacher_prefix_one_hot_tr = vectorizer.fit_transform(X_tr['teacher_prefix'].values.astype('U'))
teacher_prefix_one_hot_test = vectorizer.transform(X_test['teacher_prefix'].values.astype('U'))
```

In [98]:

```
project_grade_category_one_hot_tr = vectorizer.fit_transform(X_tr['project_grade_category'].values
)
project_grade_category_one_hot_test = vectorizer.transform(X_test['project_grade_category'].values
)
```

1.5.3 Vectorizing Numerical features

In [99]:

```
from sklearn.preprocessing import StandardScaler
price_scalar = StandardScaler()
price_standardized_tr = price_scalar.fit_transform(X_tr['price'].values.reshape(-1,1)) # finding th
e mean and standard deviation of this data
price_standardized_test = price_scalar.transform(X_test['price'].values.reshape(-1,1))
```

In [100]:

```
previously_posted_scalar = StandardScaler()
previously_posted_standardized_tr =
previously_posted_scalar.fit_transform(X_tr['teacher_number_of_previously_posted_projects'].values
.reshape(-1, 1))
previously_posted_standardized_test =
previously_posted_scalar.transform(X_test['teacher_number_of_previously_posted_projects'].values.r
eshape(-1, 1))
```

In [101]:

```
quantity_scalar = StandardScaler()
quantity_standardized_tr = quantity_scalar.fit_transform(X_tr['quantity'].values.reshape(-1, 1))
quantity_standardized_test = quantity_scalar.transform(X_test['quantity'].values.reshape(-1, 1))
```

1.5.4 Merging all the above features

- **we need to merge all the numerical vectors i.e catogorical, text, numerical vectors**

In [104]:

```
from scipy.sparse import hstack
set1_tr =
hstack((categories_one_hot_tr,sub_categories_one_hot_tr,teacher_prefix_one_hot_tr,school_state_one_
hot_tr,project_grade_category_one_hot_tr,price_standardized_tr,previously_posted_standardized_tr,q
uantity_standardized_tr,title_tfidf_tr,text_tfidf_tr))
```

```
set1_test =
hstack((categories_one_hot_test,sub_categories_one_hot_test,teacher_prefix_one_hot_test,school_stat
e_one_hot_test,project_grade_category_one_hot_test,price_standardized_test,previously_posted_standa
rdized_test,quantity_standardized_test,title_tfidf_test,text_tfidf_test))
print(set1_tr.shape)
print(set1_test.shape)
```

```
(32000, 300)
(8000, 300)
```

In [0]:

```
import sys
import math

import numpy as np
from sklearn.grid_search import GridSearchCV
from sklearn.metrics import roc_auc_score

# you might need to install this one
import xgboost as xgb

class XGBoostClassifier():
    def __init__(self, num_boost_round=10, **params):
        self.clf = None
        self.num_boost_round = num_boost_round
        self.params = params
        self.params.update({'objective': 'multi:softprob'})

    def fit(self, X, y, num_boost_round=None):
        num_boost_round = num_boost_round or self.num_boost_round
        self.label2num = {label: i for i, label in enumerate(sorted(set(y)))}
        dtrain = xgb.DMatrix(X, label=[self.label2num[label] for label in y])
        self.clf = xgb.train(params=self.params, dtrain=dtrain, num_boost_round=num_boost_round, ve
rbose_eval=1)

    def predict(self, X):
        num2label = {i: label for label, i in self.label2num.items()}
        Y = self.predict_proba(X)
        y = np.argmax(Y, axis=1)
        return np.array([num2label[i] for i in y])

    def predict_proba(self, X):
        dtest = xgb.DMatrix(X)
        return self.clf.predict(dtest)

    def score(self, X, y):
        Y = self.predict_proba(X)[:,1]
        return roc_auc_score(y, Y)

    def get_params(self, deep=True):
        return self.params

    def set_params(self, **params):
        if 'num_boost_round' in params:
            self.num_boost_round = params.pop('num_boost_round')
        if 'objective' in params:
            del params['objective']
        self.params.update(params)
        return self


clf = XGBoostClassifier(eval_metric = 'auc', num_class = 2, nthread = 4,)
##################################################################
#               Change from here                                 #
##################################################################
parameters = {
    'num_boost_round': [100, 250, 500],
    'eta': [0.05, 0.1, 0.3],
    'max_depth': [6, 9, 12],
    'subsample': [0.9, 1.0],
    'colsample_bytree': [0.9, 1.0],
}

clf = GridSearchCV(clf, parameters)
X = np.array([[1,2], [3,4], [2,1], [4,3], [1,0], [4,5]])
```

```
X = np.array([[1,2], [3,4], [2,1], [4,3], [1,0], [4,5]])
Y = np.array([0, 1, 0, 1, 0, 1])
clf.fit(X, Y)

# print(clf.grid_scores_)
best_parameters, score, _ = max(clf.grid_scores_, key=lambda x: x[1])
print('score:', score)
for param_name in sorted(best_parameters.keys()):
    print("%s: %r" % (param_name, best_parameters[param_name]))
```

```
score: 0.8333333333333334
colsample_bytree: 0.9
eta: 0.05
max_depth: 6
num_boost_round: 100
subsample: 0.9
```

2.5 Apply XGBoost on the Final Features from the above section

https://xgboost.readthedocs.io/en/latest/python/python_intro.html

In [105]:

```
import xgboost as xgb
from sklearn.metrics import roc_auc_score
from sklearn.model_selection import GridSearchCV
from sklearn.metrics import roc_curve, auc
from sklearn.metrics import confusion_matrix
import plotly.offline as offline
import plotly.graph_objs as go
offline.init_notebook_mode()
```

In [0]:

```
# No need to split the data into train and test(cv)
# use the Dmatrix and apply xgboost on the whole data
# please check the Quora case study notebook as reference

# please write all the code with proper documentation, and proper titles for each subsection
# go through documentations and blogs before you start coding
# first figure out what to do, and then think about how to do.
# reading and understanding error messages will be very much helpfull in debugging your code
# when you plot any graph make sure you use
    # a. Title, that describes your plot, this will be very helpful to the reader
    # b. Legends if needed
    # c. X-axis label
    # d. Y-axis label
```

In [106]:

```
xgb_model  = xgb.XGBClassifier(class_weight='balanced',n_jobs=-1)
parameters = [{'n_estimators': [10,20,50,70],'max_depth':   [4,5,6,7]}]
grid_search = GridSearchCV(xgb_model, parameters, cv=5, scoring='roc_auc')
grid_search.fit(set1_tr, y_tr.values.ravel())

scores_train = grid_search.cv_results_['mean_train_score'].reshape(len(parameters[0]['n_estimators'
]),len(parameters[0]['max_depth']))
scores_cv = grid_search.cv_results_['mean_test_score'].reshape(len(parameters[0]['n_estimators']),l
en(parameters[0]['max_depth']))

trace1 = go.Scatter3d(x=grid_search.cv_results_['param_n_estimators'],y=grid_search.cv_results_['pa
ram_max_depth'],z=scores_train.ravel(), name = 'train')
trace2 = go.Scatter3d(x=grid_search.cv_results_['param_n_estimators'],y=grid_search.cv_results_['pa
ram_max_depth'],z=scores_cv.ravel(), name = 'Cross validation')
data = [trace1, trace2]

layout = go.Layout(scene = dict(
        xaxis = dict(title='n_estimators'),
        yaxis = dict(title='max_depth'),
        zaxis = dict(title='AUC'),))
```

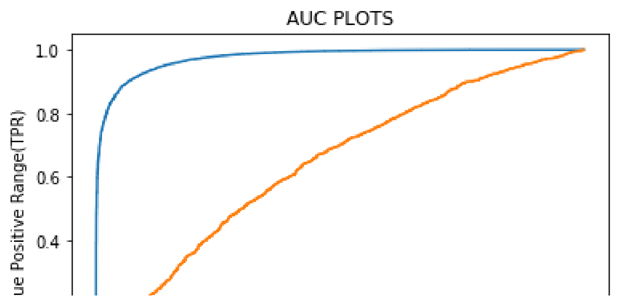
```
fig = go.Figure(data=data, layout=layout)
offline.iplot(fig, filename='3d-scatter-colorscale')
```

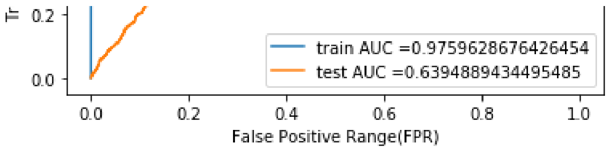
In [107]:

```
xg_best_params={'max_depth': 7, 'n_estimators':70}
```

In [108]:

```
xgb_model =
xgb.XGBClassifier(max_depth=xg_best_params['max_depth'],n_estimators=xg_best_params['n_estimators'
],class_weight='balanced',n_jobs=-1)
xgb_model.fit(set1_tr, np.ravel(y_tr,order='C'))
y_train_pred=[]
y_test_pred=[]
for j in range(0, set1_tr.shape[0], 1000):
    y_train_pred.extend(xgb_model.predict_proba(set1_tr.tocsr()[j:j+1000])[:,1])
for j in range(0, set1_test.shape[0], 1000):
    y_test_pred.extend(xgb_model.predict_proba(set1_test.tocsr()[j:j+1000])[:,1])
train_fpr, train_tpr, thresholds =  roc_curve(y_tr, y_train_pred)
test_fpr, test_tpr, thresholds = roc_curve(y_test, y_test_pred)

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)))
xg_train_auc=auc(train_fpr, train_tpr)
xg_test_auc=auc(test_fpr, test_tpr)
plt.legend()
plt.xlabel("False Positive Range(FPR)")
plt.ylabel("True Positive Range(TPR)")
plt.title("AUC PLOTS")
plt.show()
```




Tr
0.2
0.0
train AUC =0.9759628676426454
test AUC =0.6394889434495485
0.0
0.2
0.4
0.6
0.8
1.0
False Positive Range(FPR)