In [1]:
```
%matplotlib inline
import warnings
warnings.filterwarnings("ignore")

import pandas as pd
import numpy as np
import nltk
import matplotlib.pyplot as plt
import seaborn as sns
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

import re
from nltk.stem.wordnet import WordNetLemmatizer

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

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

1 PreProcessing Data

1.1 Reading Data

In [14]:
```
project_data=pd.read_csv("C:/Users/91888/Desktop/Assignment/DecisionTree Assignment/train_data.csv")
resource_data=pd.read_csv("C:/Users/91888/Desktop/Assignment/DecisionTree Assignment/resources.csv")
```

In [15]:
```
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 [16]:
```
print("Number of data points in resource data",resource_data.shape)
print('_'*50)
print("The attributes of data :",resource_data.columns.values)
```

```
Number of data points in resource data (1541272, 4)
__________________________________________________
The attributes of data : ['id' 'description' 'quantity' 'price']
```

1.2 Preporcessing Categorical Data

teacher_prefix

In [17]:
```
project_data['teacher_prefix'].value_counts()
```

Out[17]:
```
Mrs.       57269
Ms.        38955
Mr.        10648
Teacher     2360
Dr.           13
Name: teacher_prefix, dtype: int64
```

In [18]:
```
print(project_data['teacher_prefix'].isnull().values.any())
print("Number of nan values", project_data['teacher_prefix'].isnull().values.sum())
```

```
True
Number of nan values 3
```

In [19]:
```
#replace msiing values with Mrs
project_data['teacher_prefix']=project_data['teacher_prefix'].fillna('Mrs.')
```

In [20]:
```
project_data['teacher_prefix'].value_counts()
```

Out[20]:
```
Mrs.       57272
Ms.        38955
Mr.        10648
Teacher     2360
Dr.           13
Name: teacher_prefix, dtype: int64
```

In [21]:
```
project_data['teacher_prefix']=project_data['teacher_prefix'].str.replace('.','')
project_data['teacher_prefix']=project_data['teacher_prefix'].str.lower()
project_data['teacher_prefix'].value_counts()
```

Out[21]:
```
mrs        57272
ms         38955
mr         10648
teacher     2360
dr            13
Name: teacher_prefix, dtype: int64
```

project_grade_category

In [22]:
```
project_data['project_grade_category'].value_counts()
```

Out[22]:
```
Grades PreK-2    44225
Grades 3-5       37137
Grades 6-8       16923
Grades 9-12      10963
Name: project_grade_category, dtype: int64
```

In [23]:
```
print(project_data['project_grade_category'].isnull().values.any())
print("Number of nan values", project_data['project_grade_category'].isnull().values.sum())
```

```
False
Number of nan values 0
```

In [24]:
```
project_data['project_grade_category']=project_data['project_grade_category'].str.replace(' ','_')
project_data['project_grade_category']=project_data['project_grade_category'].str.replace('-','_')
project_data['project_grade_category']=project_data['project_grade_category'].str.lower()
project_data['project_grade_category'].value_counts()
```

Out[24]:
```
grades_prek_2    44225
grades_3_5       37137
grades_6_8       16923
grades_9_12      10963
Name: project_grade_category, dtype: int64
```

school_state

In [25]:
```
project_data['school_state'].value_counts()
```

Out[25]:

CA	15388
TX	7396
NY	7318
FL	6185
NC	5091
IL	4350
GA	3963
SC	3936
MI	3161
PA	3109
IN	2620
MO	2576
OH	2467
LA	2394
MA	2389
WA	2334
OK	2276
NJ	2237
AZ	2147
VA	2045
WI	1827
AL	1762
UT	1731
TN	1688
CT	1663
MD	1514
NV	1367
MS	1323
KY	1304
OR	1242
MN	1208
CO	1111
AR	1049
ID	693
IA	666
KS	634
NM	557
DC	516
HI	507
ME	505
WV	503

```
NH      348
AK      345
DE      343
NE      309
SD      300
RI      285
MT      245
ND      143
WY       98
VT       80
Name: school_state, dtype: int64
```

In [26]:
```
project_data['school_state'].isnull().values.any()
```

Out[26]:
```
False
```

In [27]:
```
project_data['school_state']=project_data['school_state'].str.lower()
project_data['school_state'].value_counts()
```

Out[27]:

ca	15388
tx	7396
ny	7318
fl	6185
nc	5091
il	4350
ga	3963
sc	3936
mi	3161
pa	3109
in	2620
mo	2576
oh	2467
la	2394
ma	2389
wa	2334
ok	2276
nj	2237
az	2147
va	2045
wi	1827
al	1762
ut	1731
tn	1688
ct	1663
md	1514
nv	1367
ms	1323
ky	1304
or	1242
mn	1208
co	1111
ar	1049
id	693
ia	666
ks	634
nm	557
dc	516
hi	507
me	505
wv	503

```
nh      348
ak      345
de      343
ne      309
sd      300
ri      285
mt      245
nd      143
wy       98
vt       80
Name: school_state, dtype: int64
```

project_subject_categories

In [28]:
```
project_data['project_subject_categories'].value_counts()
```

Out[28]:

Literacy & Language	23655
Math & Science	17072
Literacy & Language, Math & Science	14636
Health & Sports	10177
Music & The Arts	5180
Special Needs	4226
Literacy & Language, Special Needs	3961
Applied Learning	3771
Math & Science, Literacy & Language	2289
Applied Learning, Literacy & Language	2191
History & Civics	1851
Math & Science, Special Needs	1840
Literacy & Language, Music & The Arts	1757
Math & Science, Music & The Arts	1642
Applied Learning, Special Needs	1467
History & Civics, Literacy & Language	1421
Health & Sports, Special Needs	1391
Warmth, Care & Hunger	1309
Math & Science, Applied Learning	1220
Applied Learning, Math & Science	1052
Literacy & Language, History & Civics	809
Health & Sports, Literacy & Language	803
Applied Learning, Music & The Arts	758
Math & Science, History & Civics	652
Literacy & Language, Applied Learning	636
Applied Learning, Health & Sports	608
Math & Science, Health & Sports	414
History & Civics, Math & Science	322
History & Civics, Music & The Arts	312
Special Needs, Music & The Arts	302
Health & Sports, Math & Science	271
History & Civics, Special Needs	252
Health & Sports, Applied Learning	192
Applied Learning, History & Civics	178
Health & Sports, Music & The Arts	155
Music & The Arts, Special Needs	138
Literacy & Language, Health & Sports	72
Health & Sports, History & Civics	43
History & Civics, Applied Learning	42
Special Needs, Health & Sports	42
Health & Sports, Warmth, Care & Hunger	23

```
Special Needs, Warmth, Care & Hunger            23
Music & The Arts, Health & Sports               19
Music & The Arts, History & Civics              18
History & Civics, Health & Sports               13
Math & Science, Warmth, Care & Hunger           11
Applied Learning, Warmth, Care & Hunger         10
Music & The Arts, Applied Learning              10
Literacy & Language, Warmth, Care & Hunger       9
Music & The Arts, Warmth, Care & Hunger          2
History & Civics, Warmth, Care & Hunger          1
Name: project_subject_categories, dtype: int64
```

In [29]:
```
print(project_data['project_subject_categories'].isnull().values.any())
print("Number of nan values", project_data['project_subject_categories'].isnull().values.sum())
```

```
False
Number of nan values 0
```

In [30]:
```
project_data['project_subject_categories'] = project_data['project_subject_categories'].str.replace(' The ','')
project_data['project_subject_categories'] = project_data['project_subject_categories'].str.replace(' ','')
project_data['project_subject_categories'] = project_data['project_subject_categories'].str.replace('&','_')
project_data['project_subject_categories'] = project_data['project_subject_categories'].str.replace(',','_')
project_data['project_subject_categories'] = project_data['project_subject_categories'].str.lower()
project_data['project_subject_categories'].value_counts()
```

Out[30]:

literacy_language	23655
math_science	17072
literacy_language_math_science	14636
health_sports	10177
music_arts	5180
specialneeds	4226
literacy_language_specialneeds	3961
appliedlearning	3771
math_science_literacy_language	2289
appliedlearning_literacy_language	2191
history_civics	1851
math_science_specialneeds	1840
literacy_language_music_arts	1757
math_science_music_arts	1642
appliedlearning_specialneeds	1467
history_civics_literacy_language	1421
health_sports_specialneeds	1391
warmth_care_hunger	1309
math_science_appliedlearning	1220
appliedlearning_math_science	1052
literacy_language_history_civics	809
health_sports_literacy_language	803
appliedlearning_music_arts	758
math_science_history_civics	652
literacy_language_appliedlearning	636
appliedlearning_health_sports	608
math_science_health_sports	414
history_civics_math_science	322
history_civics_music_arts	312
specialneeds_music_arts	302
health_sports_math_science	271
history_civics_specialneeds	252
health_sports_appliedlearning	192
appliedlearning_history_civics	178
health_sports_music_arts	155
music_arts_specialneeds	138
literacy_language_health_sports	72
health_sports_history_civics	43
specialneeds_health_sports	42
history_civics_appliedlearning	42
health_sports_warmth_care_hunger	23

```
specialneeds_warmth_care_hunger            23
music_arts_health_sports                   19
music_arts_history_civics                  18
history_civics_health_sports               13
math_science_warmth_care_hunger            11
music_arts_appliedlearning                 10
appliedlearning_warmth_care_hunger         10
literacy_language_warmth_care_hunger        9
music_arts_warmth_care_hunger               2
history_civics_warmth_care_hunger           1
Name: project_subject_categories, dtype: int64
```

project_subject_subcategories

In [31]:
```
project_data['project_subject_subcategories'].value_counts()
```

Out[31]:

Literacy	9486
Literacy, Mathematics	8325
Literature & Writing, Mathematics	5923
Literacy, Literature & Writing	5571
Mathematics	5379
Literature & Writing	4501
Special Needs	4226
Health & Wellness	3583
Applied Sciences, Mathematics	3399
Applied Sciences	2492
Literacy, Special Needs	2440
Gym & Fitness, Health & Wellness	2264
ESL, Literacy	2234
Visual Arts	2217
Music	1472
Warmth, Care & Hunger	1309
Literature & Writing, Special Needs	1306
Gym & Fitness	1195
Health & Wellness, Special Needs	1189
Mathematics, Special Needs	1187
Environmental Science	1079
Team Sports	1061
Applied Sciences, Environmental Science	984
Environmental Science, Health & Life Science	964
Music, Performing Arts	948
Early Development	905
Environmental Science, Mathematics	838
Other	831
Health & Life Science	827
Health & Wellness, Nutrition Education	797
...	...
Environmental Science, Team Sports	2
College & Career Prep, Team Sports	2
Early Development, Economics	2
Foreign Languages, Gym & Fitness	2
Character Education, Nutrition Education	2
Financial Literacy, Parent Involvement	2
Community Service, Financial Literacy	1
ESL, Team Sports	1
Community Service, Gym & Fitness	1
Civics & Government, Parent Involvement	1

```
Literature & Writing, Nutrition Education          1
Economics, Nutrition Education                    1
Economics, Foreign Languages                      1
History & Geography, Warmth, Care & Hunger        1
Parent Involvement, Team Sports                   1
Gym & Fitness, Warmth, Care & Hunger              1
ESL, Economics                                    1
Parent Involvement, Warmth, Care & Hunger         1
Economics, Other                                  1
Financial Literacy, Performing Arts               1
Gym & Fitness, Parent Involvement                 1
Civics & Government, Nutrition Education          1
Community Service, Music                          1
Gym & Fitness, Social Sciences                    1
College & Career Prep, Warmth, Care & Hunger      1
Financial Literacy, Foreign Languages             1
Economics, Music                                  1
Other, Warmth, Care & Hunger                      1
Civics & Government, Foreign Languages            1
Extracurricular, Financial Literacy               1
Name: project_subject_subcategories, Length: 401, dtype: int64
```

In [32]:
```
print(project_data['project_subject_subcategories'].isnull().values.any())
print("Number of nan values", project_data['project_subject_subcategories'].isnull().values.sum())
```

```
False
Number of nan values 0
```

In [33]:
```
project_data['project_subject_subcategories'] = project_data['project_subject_subcategories'].str.replace(' The ','')
project_data['project_subject_subcategories'] = project_data['project_subject_subcategories'].str.replace(' ','')
project_data['project_subject_subcategories'] = project_data['project_subject_subcategories'].str.replace('&','_')
project_data['project_subject_subcategories'] = project_data['project_subject_subcategories'].str.replace(',','_')
project_data['project_subject_subcategories'] = project_data['project_subject_subcategories'].str.lower()
project_data['project_subject_subcategories'].value_counts()
```

Out[33]:

literacy	9486
literacy_mathematics	8325
literature_writing_mathematics	5923
literacy_literature_writing	5571
mathematics	5379
literature_writing	4501
specialneeds	4226
health_wellness	3583
appliedsciences_mathematics	3399
appliedsciences	2492
literacy_specialneeds	2440
gym_fitness_health_wellness	2264
esl_literacy	2234
visualarts	2217
music	1472
warmth_care_hunger	1309
literature_writing_specialneeds	1306
gym_fitness	1195
health_wellness_specialneeds	1189
mathematics_specialneeds	1187
environmentalscience	1079
teamsports	1061
appliedsciences_environmentalscience	984
environmentalscience_health_lifescience	964
music_performingarts	948
earlydevelopment	905
environmentalscience_mathematics	838
other	831
health_lifescience	827
health_wellness_nutritioneducation	797
...	...
visualarts_warmth_care_hunger	2
economics_literature_writing	2
financialliteracy_health_wellness	2
environmentalscience_teamsports	2
civics_government_teamsports	2
civics_government_health_wellness	2
literature_writing_nutritioneducation	1
other_warmth_care_hunger	1
parentinvolvement_warmth_care_hunger	1
college_careerprep_warmth_care_hunger	1

```
esl_teamsports                              1
civics_government_nutritioneducation        1
esl_economics                               1
financialliteracy_performingarts            1
economics_nutritioneducation                1
parentinvolvement_teamsports                1
communityservice_gym_fitness                1
gym_fitness_socialsciences                  1
communityservice_music                      1
extracurricular_financialliteracy           1
civics_government_parentinvolvement         1
economics_foreignlanguages                  1
financialliteracy_foreignlanguages          1
economics_music                             1
gym_fitness_warmth_care_hunger              1
civics_government_foreignlanguages          1
history_geography_warmth_care_hunger        1
economics_other                             1
gym_fitness_parentinvolvement               1
communityservice_financialliteracy          1
Name: project_subject_subcategories, Length: 401, dtype: int64
```

1.3 Preporcessing Text Data

project_essay

In [37]:
```
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 [38]:
```
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', 'furthe
r',\
            'then', 'once', 'here', 'there', 'when', 'where', 'why', 'how', 'all', 'any', 'both', 'each', 'few', 'mor
e',\
            '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', "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', "were
n't", \
            'won', "won't", 'wouldn', "wouldn't"]
```

In [39]:
```
from tqdm import tqdm
def preprocess_text(text_data):
    preprocessed_text = []
    # tqdm is for printing the status bar
    for sentance in tqdm(text_data):
        sent = decontracted(sentance)
        sent = sent.replace('\\r', ' ')
        sent = sent.replace('\\n', ' ')
        sent = sent.replace('\\"', ' ')
        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_text.append(sent.lower().strip())
    return preprocessed_text
```

In [40]:
```
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 [41]:
```
print("Printing some random essays")
print(9,project_data["essay"].values[9])
print('-'*100)
print(34,project_data["essay"].values[34])
print('-'*100)
print(147,project_data["essay"].values[147])
```

Printing some random essays
9 Over 95% of my students are on free or reduced lunch. I have a few who are homeless, but despite that, they come t
o school with an eagerness to learn. My students are inquisitive eager learners who embrace the challenge of not ha
ving great books and other resources every day. Many of them are not afforded the opportunity to engage with these
big colorful pages of a book on a regular basis at home and they don't travel to the public library. \r\nIt is my du
ty as a teacher to do all I can to provide each student an opportunity to succeed in every aspect of life. \r\nReadin
g is Fundamental! My students will read these books over and over again while boosting their comprehension skills. Th
ese books will be used for read alouds, partner reading and for Independent reading. \r\nThey will engage in reading
to build their \"Love for Reading\" by reading for pure enjoyment. They will be introduced to some new authors as wel
l as some old favorites. I want my students to be ready for the 21st Century and know the pleasure of holding a good
hard back book in hand. There's nothing like a good book to read! \r\nMy students will soar in Reading, and more bec
ause of your consideration and generous funding contribution. This will help build stamina and prepare for 3rd grade.
Thank you so much for reading our proposal!nannan
---
34 My students mainly come from extremely low-income families, and the majority of them come from homes where both pa
rents work full time. Most of my students are at school from 7:30 am to 6:00 pm (2:30 to 6:00 pm in the after-school
program), and they all receive free and reduced meals for breakfast and lunch. \r\n\r\n\r\nI want my students to feel
as comfortable in my classroom as they do at home. Many of my students take on multiple roles both at home as well as
in school. They are sometimes the caretakers of younger siblings, cooks, babysitters, academics, friends, and most of
all, they are developing who they are going to become as adults. I consider it an essential part of my job to model
helping others gain knowledge in a positive manner. As a result, I have a community of students who love helping each
other in and outside of the classroom. They consistently look for opportunities to support each other's learning in a
kind and helpful way.I am excited to be experimenting with alternative seating in my classroom this school year. Stud
ies have shown that giving students the option of where they sit in a classroom increases focus as well as motivatio
n. \r\n\r\nBy allowing students choice in the classroom, they are able to explore and create in a welcoming environm
ent. Alternative classroom seating has been experimented with more frequently in recent years. I believe (along with
many others), that every child learns differently. This does not only apply to how multiplication is memorized, or a
paper is written, but applies to the space in which they are asked to work. I have had students in the past ask \"Can
I work in the library? Can I work on the carpet?\" My answer was always, \"As long as you're learning, you can work w
herever you want!\" \r\n\r\nWith the yoga balls and the lap-desks, I will be able to increase the options for seating
in my classroom and expand its imaginable space.nannan
---
147 My students are eager to learn and make their mark on the world.\r\n\r\nThey come from a Title 1 school and need
extra love.\r\n\r\nMy fourth grade students are in a high poverty area and still come to school every day to get thei
r education. I am trying to make it fun and educational for them so they can get the most out of their schooling. I c
reated a caring environment for the students to bloom! They deserve the best.\r\nThank you!\r\nI am requesting 1 Chro
mebook to access online interventions, differentiate instruction, and get extra practice. The Chromebook will be used
to supplement ELA and math instruction. Students will play ELA and math games that are engaging and fun, as well as p
articipate in assignments online. This in turn will help my students improve their skills. Having a Chromebook in the
classroom would not only allow students to use the programs at their own pace, but would ensure more students are get
ting adequate time to use the programs. The online programs have been especially beneficial to my students with speci

```
al needs. They are able to work at their level as well as be challenged with some different materials. This is making
these students more confident in their abilities.\r\n\r\nThe Chromebook would allow my students to have daily access
to computers and increase their computing skills.\r\nThis will change their lives for the better as they become more
successful in school. Having access to technology in the classroom would help bridge the achievement gap.nannan
```

In [42]:
```
preprocessed_essays = preprocess_text(project_data['essay'].values)
```

```
100%|██████████| 109248/109248 [01:13<00:00, 1486.19it/s]
```

In [43]:
```
print("printing some random essay")
print(9, preprocessed_essays[9])
print('-'*50)
print(34, preprocessed_essays[34])
print('-'*50)
print(147, preprocessed_essays[147])
```

```
printing some random essay
9 95 students free reduced lunch homeless despite come school eagerness learn students inquisitive eager learners emb
race challenge not great books resources every day many not afforded opportunity engage big colorful pages book regul
ar basis home not travel public library duty teacher provide student opportunity succeed every aspect life reading fu
ndamental students read books boosting comprehension skills books used read alouds partner reading independent readin
g engage reading build love reading reading pure enjoyment introduced new authors well old favorites want students re
ady 21st century know pleasure holding good hard back book hand nothing like good book read students soar reading con
sideration generous funding contribution help build stamina prepare 3rd grade thank much reading proposal nannan
--------------------------------------------------
34 students mainly come extremely low income families majority come homes parents work full time students school 7 30
6 00 pm 2 30 6 00 pm school program receive free reduced meals breakfast lunch want students feel comfortable classro
om home many students take multiple roles home well school sometimes caretakers younger siblings cooks babysitters ac
ademics friends developing going become adults consider essential part job model helping others gain knowledge positi
ve manner result community students love helping outside classroom consistently look opportunities support learning k
ind helpful way excited experimenting alternative seating classroom school year studies shown giving students option
sit classroom increases focus well motivation allowing students choice classroom able explore create welcoming enviro
nment alternative classroom seating experimented frequently recent years believe along many others every child learns
differently not apply multiplication memorized paper written applies space asked work students past ask work library
work carpet answer always long learning work wherever want yoga balls lap desks able increase options seating classro
om expand imaginable space nannan
--------------------------------------------------
147 students eager learn make mark world come title 1 school need extra love fourth grade students high poverty area
still come school every day get education trying make fun educational get schooling created caring environment studen
ts bloom deserve best thank requesting 1 chromebook access online interventions differentiate instruction get extra p
ractice chromebook used supplement ela math instruction students play ela math games engaging fun well participate as
signments online turn help students improve skills chromebook classroom would not allow students use programs pace wo
uld ensure students getting adequate time use programs online programs especially beneficial students special needs a
ble work level well challenged different materials making students confident abilities chromebook would allow student
s daily access computers increase computing skills change lives better become successful school access technology cla
ssroom would help bridge achievement gap nannan
```

In [44]:
```
#adding processed essays to project_data
project_data['processed_essay']=preprocessed_essays
```

1.4 Preprocessing Numerical Features

In [45]:
```
price_data = resource_data.groupby('id').agg({'price':'sum', 'quantity':'sum'}).reset_index()
price_data.head(2)
```

Out[45]:

	id	price	quantity
0	p000001	459.56	7
1	p000002	515.89	21

In [46]:
```
# join two dataframes in python:
project_data = pd.merge(project_data, price_data, on='id', how='left')
```

In [47]:
```
project_data['price'].head()
```

Out[47]:
```
0    154.60
1    299.00
2    516.85
3    232.90
4     67.98
Name: price, dtype: float64
```

In [48]:
```
project_data.columns.values
```

Out[48]:
```
array(['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', 'essay', 'processed_essay', 'price',
       'quantity'], dtype=object)
```

removing unnecessary columns

In [49]:
```
project_data = project_data.drop(project_data.columns[[0,1,2,5,9,10,11,12,13,14,17,20]], axis=1)
```

In [50]:
```
project_data.head()
```

Out[50]:

	teacher_prefix	school_state	project_grade_category	project_subject_categories	project_subject_subcategories	teacher_numb
0	mrs	in	grades_prek_2	literacy_language	esl_literacy	0
1	mr	fl	grades_6_8	history_civics_health_sports	civics_government_teamsports	7
2	ms	az	grades_6_8	health_sports	health_wellness_teamsports	1
3	mrs	ky	grades_prek_2	literacy_language_math_science	literacy_mathematics	4
4	mrs	tx	grades_prek_2	math_science	mathematics	1

In [51]:
```
#renaming some columns
project_data = project_data.rename(index=str, columns=
                                   {'project_subject_categories':'clean_categories',
                                    'project_subject_subcategories':'clean_subcategories'})
```

In [52]:
```
#changing position of columns : https://stackoverflow.com/questions/41968732/set-order-of-columns-in-pandas-dataframe
project_data = project_data[['processed_essay', 'teacher_prefix','project_grade_category',
                             'school_state','clean_categories','clean_subcategories',
                             'teacher_number_of_previously_posted_projects','price','project_is_approved']]
```

converting dataframe to csv

In [55]:
```
project_data.to_csv(r'C:/Users/91888/Desktop/Assignment/DecisionTree Assignment/preprocessed_data.csv', index = False)
```

2. Decision Tree

2.1 Loading Data

In [2]:
```
data=pd.read_csv("C:/Users/91888/Desktop/Assignment/DecisionTree Assignment/preprocessed_data.csv",nrows=100000)
```

In [3]:
```
data.head()
```

Out[3]:

	processed_essay	teacher_prefix	project_grade_category	school_state	clean_categories	clean_subcategorie
0	students english learners working english seco...	mrs	grades_prek_2	in	literacy_language	esl_literacy
1	students arrive school eager learn polite gene...	mr	grades_6_8	fl	history_civics_health_sports	civics_government_teamspor
2	true champions not always ones win guts mia ha...	ms	grades_6_8	az	health_sports	health_wellness_teamsports
3	work unique school filled esl english second l...	mrs	grades_prek_2	ky	literacy_language_math_science	literacy_mathematics
4	second grade classroom next year made around 2...	mrs	grades_prek_2	tx	math_science	mathematics

In [4]:
```
data.shape
```

Out[4]: (100000, 9)

2.2 Splitting data into Train and Cross Validation

In [5]:
```
y = data['project_is_approved'].values
X = data.drop(['project_is_approved'], axis=1)
```

In [6]:
```
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, stratify=y)
```

In [7]:
```
X_train.shape
```

Out[7]: (67000, 8)

In [8]:
```
X_test.shape
```

Out[8]: (33000, 8)

2.3 Vectorizing Text Data

TFIDF

In [9]:
```
print(X_train.shape, y_train.shape)
print(X_test.shape, y_test.shape)

print("="*100)


vectorizer1 = TfidfVectorizer(min_df=5, max_features=5000)
vectorizer1.fit(X_train['processed_essay'].values) # fit has to happen only on train data


X_train_essay_tfidf = vectorizer1.transform(X_train['processed_essay'].values)
X_test_essay_tfidf = vectorizer1.transform(X_test['processed_essay'].values)

print("After vectorizations")
print(X_train_essay_tfidf.shape, y_train.shape)
print(X_test_essay_tfidf.shape, y_test.shape)
print("="*100)
```

```
(67000, 8) (67000,)
(33000, 8) (33000,)
====================================================================================================
After vectorizations
(67000, 5000) (67000,)
(33000, 5000) (33000,)
====================================================================================================
```

TFIDF W2V

In [10]:
```
with open("C:/Users/91888/Desktop/Assignment/DecisionTree Assignment/glove_vectors", 'rb') as f:
    model = pickle.load(f)
    glove_words =  set(model.keys())
```

In [11]:
```
tfidf_model = TfidfVectorizer(min_df=5, max_features=5000)
tfidf_model.fit(X_train['processed_essay'].values)
dictionary = dict(zip(tfidf_model.get_feature_names(), list(tfidf_model.idf_)))
tfidf_words = set(tfidf_model.get_feature_names())
```

TFIDF W2V X_train

In [12]:
```
train_tfidf_w2v_vectors = [];
for sentence in tqdm(X_train['processed_essay']):
    vector = np.zeros(300)
    tf_idf_weight =0;
    for word in sentence.split():
        if (word in glove_words) and (word in tfidf_words):
            vec = model[word]
            tf_idf = dictionary[word]*(sentence.count(word)/len(sentence.split()))
            vector += (vec * tf_idf)
            tf_idf_weight += tf_idf
    if tf_idf_weight != 0:
        vector /= tf_idf_weight
    train_tfidf_w2v_vectors.append(vector)

print(len(train_tfidf_w2v_vectors))
print(len(train_tfidf_w2v_vectors[0]))
```

```
100%|██████████| 67000/67000 [04:09<00:00, 268.23it/s]

67000
300
```

TFIDF W2V X_test

In [13]:
```
test_tfidf_w2v_vectors = [];
for sentence in tqdm(X_test['processed_essay']):
    vector = np.zeros(300)
    tf_idf_weight =0;
    for word in sentence.split():
        if (word in glove_words) and (word in tfidf_words):
            vec = model[word]
            tf_idf = dictionary[word]*(sentence.count(word)/len(sentence.split()))
            vector += (vec * tf_idf)
            tf_idf_weight += tf_idf
    if tf_idf_weight != 0:
        vector /= tf_idf_weight
    test_tfidf_w2v_vectors.append(vector)

print(len(test_tfidf_w2v_vectors))
print(len(test_tfidf_w2v_vectors[0]))
```

```
100%|██████████| 33000/33000 [02:03<00:00, 268.24it/s]

33000
300
```

2.4 Encoding Categorical Features

teacher_prefix

In [14]:
```
vectorizer3 = CountVectorizer()
vectorizer3.fit(X_train['teacher_prefix'].values)


X_train_teacher_ohe = vectorizer3.transform(X_train['teacher_prefix'].values)
X_test_teacher_ohe = vectorizer3.transform(X_test['teacher_prefix'].values)

print("After vectorizations")
print(X_train_teacher_ohe.shape, y_train.shape)
print(X_test_teacher_ohe.shape, y_test.shape)
print(vectorizer3.get_feature_names())
print("="*100)
```

```
After vectorizations
(67000, 5) (67000,)
(33000, 5) (33000,)
['dr', 'mr', 'mrs', 'ms', 'teacher']
====================================================================================================
```

project_grade_category

In [15]:
```
vectorizer4 = CountVectorizer()
vectorizer4.fit(X_train['project_grade_category'].values)


X_train_grade_ohe = vectorizer4.transform(X_train['project_grade_category'].values)
X_test_grade_ohe = vectorizer4.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(vectorizer4.get_feature_names())
print("="*100)
```

```
After vectorizations
(67000, 4) (67000,)
(33000, 4) (33000,)
['grades_3_5', 'grades_6_8', 'grades_9_12', 'grades_prek_2']
====================================================================================================
```

school_state

In [16]:
```
vectorizer5 = CountVectorizer()
vectorizer5.fit(X_train['school_state'].values)


X_train_state_ohe = vectorizer5.transform(X_train['school_state'].values)
X_test_state_ohe = vectorizer5.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(vectorizer5.get_feature_names())
print("="*100)
```

```
After vectorizations
(67000, 51) (67000,)
(33000, 51) (33000,)
['ak', 'al', 'ar', 'az', 'ca', 'co', 'ct', 'dc', 'de', 'fl', 'ga', 'hi', 'ia', 'id', 'il', 'in', 'ks', 'ky', 'la', 'm
a', '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']
====================================================================================================
```

clean_categories

In [17]:
```
vectorizer6 = CountVectorizer()
vectorizer6.fit(X_train['clean_categories'].values)


X_train_category_ohe = vectorizer6.transform(X_train['clean_categories'].values)
X_test_category_ohe = vectorizer6.transform(X_test['clean_categories'].values)

print("After vectorizations")
print(X_train_category_ohe.shape, y_train.shape)
print(X_test_category_ohe.shape, y_test.shape)
print(vectorizer6.get_feature_names())
print("="*100)
```

```
After vectorizations
(67000, 51) (67000,)
(33000, 51) (33000,)
['appliedlearning', 'appliedlearning_health_sports', 'appliedlearning_history_civics', 'appliedlearning_literacy_lang
uage', 'appliedlearning_math_science', 'appliedlearning_music_arts', 'appliedlearning_specialneeds', 'appliedlearning
_warmth_care_hunger', 'health_sports', 'health_sports_appliedlearning', 'health_sports_history_civics', 'health_sport
s_literacy_language', 'health_sports_math_science', 'health_sports_music_arts', 'health_sports_specialneeds', 'health
_sports_warmth_care_hunger', 'history_civics', 'history_civics_appliedlearning', 'history_civics_health_sports', 'his
tory_civics_literacy_language', 'history_civics_math_science', 'history_civics_music_arts', 'history_civics_specialne
eds', 'history_civics_warmth_care_hunger', 'literacy_language', 'literacy_language_appliedlearning', 'literacy_langua
ge_health_sports', 'literacy_language_history_civics', 'literacy_language_math_science', 'literacy_language_music_art
s', 'literacy_language_specialneeds', 'literacy_language_warmth_care_hunger', 'math_science', 'math_science_appliedle
arning', 'math_science_health_sports', 'math_science_history_civics', 'math_science_literacy_language', 'math_science
_music_arts', 'math_science_specialneeds', 'math_science_warmth_care_hunger', 'music_arts', 'music_arts_appliedlearni
ng', 'music_arts_health_sports', 'music_arts_history_civics', 'music_arts_specialneeds', 'music_arts_warmth_care_hung
er', 'specialneeds', 'specialneeds_health_sports', 'specialneeds_music_arts', 'specialneeds_warmth_care_hunger', 'war
mth_care_hunger']
====================================================================================================
```

clean_subcategories

In [18]:
```
vectorizer7 = CountVectorizer()
vectorizer7.fit(X_train['clean_subcategories'].values)


X_train_subcategory_ohe = vectorizer7.transform(X_train['clean_subcategories'].values)
X_test_subcategory_ohe = vectorizer7.transform(X_test['clean_subcategories'].values)

print("After vectorizations")
print(X_train_subcategory_ohe.shape, y_train.shape)
print(X_test_subcategory_ohe.shape, y_test.shape)
print(vectorizer7.get_feature_names())
print("="*100)
```

```
After vectorizations
(67000, 388) (67000,)
(33000, 388) (33000,)
['appliedsciences', 'appliedsciences_charactereducation', 'appliedsciences_civics_government', 'appliedsciences_colle
ge_careerprep', 'appliedsciences_communityservice', 'appliedsciences_earlydevelopment', 'appliedsciences_economics',
'appliedsciences_environmentalscience', 'appliedsciences_esl', 'appliedsciences_extracurricular', 'appliedsciences_fi
nancialliteracy', 'appliedsciences_foreignlanguages', 'appliedsciences_gym_fitness', 'appliedsciences_health_lifescie
nce', 'appliedsciences_health_wellness', 'appliedsciences_history_geography', 'appliedsciences_literacy', 'appliedsci
ences_literature_writing', 'appliedsciences_mathematics', 'appliedsciences_music', 'appliedsciences_nutritioneducatio
n', 'appliedsciences_other', 'appliedsciences_parentinvolvement', 'appliedsciences_performingarts', 'appliedsciences_
socialsciences', 'appliedsciences_specialneeds', 'appliedsciences_teamsports', 'appliedsciences_visualarts', 'applied
sciences_warmth_care_hunger', 'charactereducation', 'charactereducation_civics_government', 'charactereducation_colle
ge_careerprep', 'charactereducation_communityservice', 'charactereducation_earlydevelopment', 'charactereducation_eco
nomics', 'charactereducation_environmentalscience', 'charactereducation_esl', 'charactereducation_extracurricular',
'charactereducation_financialliteracy', 'charactereducation_foreignlanguages', 'charactereducation_gym_fitness', 'cha
ractereducation_health_lifescience', 'charactereducation_health_wellness', 'charactereducation_history_geography', 'c
haractereducation_literacy', 'charactereducation_literature_writing', 'charactereducation_mathematics', 'characteredu
cation_music', 'charactereducation_other', 'charactereducation_parentinvolvement', 'charactereducation_performingart
s', 'charactereducation_socialsciences', 'charactereducation_specialneeds', 'charactereducation_teamsports', 'charact
ereducation_visualarts', 'charactereducation_warmth_care_hunger', 'civics_government', 'civics_government_college_car
eerprep', 'civics_government_communityservice', 'civics_government_economics', 'civics_government_environmentalscienc
e', 'civics_government_esl', 'civics_government_financialliteracy', 'civics_government_health_lifescience', 'civics_g
overnment_health_wellness', 'civics_government_history_geography', 'civics_government_literacy', 'civics_government_l
iterature_writing', 'civics_government_mathematics', 'civics_government_parentinvolvement', 'civics_government_perfor
mingarts', 'civics_government_socialsciences', 'civics_government_specialneeds', 'civics_government_teamsports', 'civ
ics_government_visualarts', 'college_careerprep', 'college_careerprep_communityservice', 'college_careerprep_earlydev
elopment', 'college_careerprep_economics', 'college_careerprep_environmentalscience', 'college_careerprep_esl', 'coll
ege_careerprep_extracurricular', 'college_careerprep_financialliteracy', 'college_careerprep_foreignlanguages', 'coll
ege_careerprep_health_lifescience', 'college_careerprep_health_wellness', 'college_careerprep_history_geography', 'co
llege_careerprep_literacy', 'college_careerprep_literature_writing', 'college_careerprep_mathematics', 'college_caree
rprep_music', 'college_careerprep_nutritioneducation', 'college_careerprep_other', 'college_careerprep_parentinvolvem
ent', 'college_careerprep_performingarts', 'college_careerprep_socialsciences', 'college_careerprep_specialneeds', 'c
ollege_careerprep_teamsports', 'college_careerprep_visualarts', 'college_careerprep_warmth_care_hunger', 'communityse
rvice', 'communityservice_earlydevelopment', 'communityservice_economics', 'communityservice_environmentalscience',
'communityservice_esl', 'communityservice_extracurricular', 'communityservice_financialliteracy', 'communityservice_g
ym_fitness', 'communityservice_health_lifescience', 'communityservice_health_wellness', 'communityservice_history_geo
graphy', 'communityservice_literacy', 'communityservice_literature_writing', 'communityservice_mathematics', 'communi
tyservice_music', 'communityservice_nutritioneducation', 'communityservice_other', 'communityservice_parentinvolvemen
t', 'communityservice_performingarts', 'communityservice_socialsciences', 'communityservice_specialneeds', 'community
service_visualarts', 'earlydevelopment', 'earlydevelopment_economics', 'earlydevelopment_environmentalscience', 'earl
ydevelopment_extracurricular', 'earlydevelopment_financialliteracy', 'earlydevelopment_foreignlanguages', 'earlydevel
```

```
opment_gym_fitness', 'earlydevelopment_health_lifescience', 'earlydevelopment_health_wellness', 'earlydevelopment_his
tory_geography', 'earlydevelopment_literacy', 'earlydevelopment_literature_writing', 'earlydevelopment_mathematics',
'earlydevelopment_music', 'earlydevelopment_nutritioneducation', 'earlydevelopment_other', 'earlydevelopment_parentin
volvement', 'earlydevelopment_performingarts', 'earlydevelopment_socialsciences', 'earlydevelopment_specialneeds', 'e
arlydevelopment_teamsports', 'earlydevelopment_visualarts', 'earlydevelopment_warmth_care_hunger', 'economics', 'econ
omics_environmentalscience', 'economics_financialliteracy', 'economics_health_lifescience', 'economics_history_geogra
phy', 'economics_literacy', 'economics_literature_writing', 'economics_mathematics', 'economics_music', 'economics_nu
tritioneducation', 'economics_socialsciences', 'economics_specialneeds', 'economics_visualarts', 'environmentalscienc
e', 'environmentalscience_extracurricular', 'environmentalscience_financialliteracy', 'environmentalscience_foreignla
nguages', 'environmentalscience_gym_fitness', 'environmentalscience_health_lifescience', 'environmentalscience_health
_wellness', 'environmentalscience_history_geography', 'environmentalscience_literacy', 'environmentalscience_literatu
re_writing', 'environmentalscience_mathematics', 'environmentalscience_music', 'environmentalscience_nutritioneducati
on', 'environmentalscience_other', 'environmentalscience_parentinvolvement', 'environmentalscience_performingarts',
'environmentalscience_socialsciences', 'environmentalscience_specialneeds', 'environmentalscience_teamsports', 'envir
onmentalscience_visualarts', 'environmentalscience_warmth_care_hunger', 'esl', 'esl_earlydevelopment', 'esl_economic
s', 'esl_environmentalscience', 'esl_extracurricular', 'esl_financialliteracy', 'esl_foreignlanguages', 'esl_gym_fitn
ess', 'esl_health_lifescience', 'esl_health_wellness', 'esl_history_geography', 'esl_literacy', 'esl_literature_writi
ng', 'esl_mathematics', 'esl_music', 'esl_nutritioneducation', 'esl_other', 'esl_parentinvolvement', 'esl_performinga
rts', 'esl_socialsciences', 'esl_specialneeds', 'esl_teamsports', 'esl_visualarts', 'extracurricular', 'extracurricul
ar_financialliteracy', 'extracurricular_foreignlanguages', 'extracurricular_gym_fitness', 'extracurricular_health_lif
escience', 'extracurricular_health_wellness', 'extracurricular_history_geography', 'extracurricular_literacy', 'extra
curricular_literature_writing', 'extracurricular_mathematics', 'extracurricular_music', 'extracurricular_nutritionedu
cation', 'extracurricular_other', 'extracurricular_parentinvolvement', 'extracurricular_performingarts', 'extracurric
ular_specialneeds', 'extracurricular_teamsports', 'extracurricular_visualarts', 'financialliteracy', 'financiallitera
cy_foreignlanguages', 'financialliteracy_health_lifescience', 'financialliteracy_health_wellness', 'financialliteracy
_history_geography', 'financialliteracy_literacy', 'financialliteracy_literature_writing', 'financialliteracy_mathema
tics', 'financialliteracy_other', 'financialliteracy_parentinvolvement', 'financialliteracy_socialsciences', 'financi
alliteracy_specialneeds', 'financialliteracy_visualarts', 'foreignlanguages', 'foreignlanguages_gym_fitness', 'foreig
nlanguages_health_lifescience', 'foreignlanguages_health_wellness', 'foreignlanguages_history_geography', 'foreignlan
guages_literacy', 'foreignlanguages_literature_writing', 'foreignlanguages_mathematics', 'foreignlanguages_music', 'f
oreignlanguages_other', 'foreignlanguages_performingarts', 'foreignlanguages_socialsciences', 'foreignlanguages_speci
alneeds', 'foreignlanguages_visualarts', 'gym_fitness', 'gym_fitness_health_lifescience', 'gym_fitness_health_wellnes
s', 'gym_fitness_history_geography', 'gym_fitness_literacy', 'gym_fitness_literature_writing', 'gym_fitness_mathemati
cs', 'gym_fitness_music', 'gym_fitness_nutritioneducation', 'gym_fitness_other', 'gym_fitness_performingarts', 'gym_f
itness_socialsciences', 'gym_fitness_specialneeds', 'gym_fitness_teamsports', 'gym_fitness_visualarts', 'gym_fitness_
warmth_care_hunger', 'health_lifescience', 'health_lifescience_health_wellness', 'health_lifescience_history_geograph
y', 'health_lifescience_literacy', 'health_lifescience_literature_writing', 'health_lifescience_mathematics', 'health
_lifescience_music', 'health_lifescience_nutritioneducation', 'health_lifescience_other', 'health_lifescience_parenti
nvolvement', 'health_lifescience_performingarts', 'health_lifescience_socialsciences', 'health_lifescience_specialnee
ds', 'health_lifescience_teamsports', 'health_lifescience_visualarts', 'health_lifescience_warmth_care_hunger', 'heal
th_wellness', 'health_wellness_history_geography', 'health_wellness_literacy', 'health_wellness_literature_writing',
'health_wellness_mathematics', 'health_wellness_music', 'health_wellness_nutritioneducation', 'health_wellness_othe
```

```
r', 'health_wellness_parentinvolvement', 'health_wellness_performingarts', 'health_wellness_socialsciences', 'health_
wellness_specialneeds', 'health_wellness_teamsports', 'health_wellness_visualarts', 'health_wellness_warmth_care_hung
er', 'history_geography', 'history_geography_literacy', 'history_geography_literature_writing', 'history_geography_ma
thematics', 'history_geography_music', 'history_geography_other', 'history_geography_parentinvolvement', 'history_geo
graphy_performingarts', 'history_geography_socialsciences', 'history_geography_specialneeds', 'history_geography_visu
alarts', 'history_geography_warmth_care_hunger', 'literacy', 'literacy_literature_writing', 'literacy_mathematics',
'literacy_music', 'literacy_nutritioneducation', 'literacy_other', 'literacy_parentinvolvement', 'literacy_performing
arts', 'literacy_socialsciences', 'literacy_specialneeds', 'literacy_teamsports', 'literacy_visualarts', 'literacy_wa
rmth_care_hunger', 'literature_writing', 'literature_writing_mathematics', 'literature_writing_music', 'literature_wr
iting_other', 'literature_writing_parentinvolvement', 'literature_writing_performingarts', 'literature_writing_social
sciences', 'literature_writing_specialneeds', 'literature_writing_teamsports', 'literature_writing_visualarts', 'lite
rature_writing_warmth_care_hunger', 'mathematics', 'mathematics_music', 'mathematics_nutritioneducation', 'mathematic
s_other', 'mathematics_parentinvolvement', 'mathematics_performingarts', 'mathematics_socialsciences', 'mathematics_s
pecialneeds', 'mathematics_teamsports', 'mathematics_visualarts', 'mathematics_warmth_care_hunger', 'music', 'music_o
ther', 'music_parentinvolvement', 'music_performingarts', 'music_socialsciences', 'music_specialneeds', 'music_teamsp
orts', 'music_visualarts', 'nutritioneducation', 'nutritioneducation_other', 'nutritioneducation_specialneeds', 'nutr
itioneducation_teamsports', 'nutritioneducation_visualarts', 'nutritioneducation_warmth_care_hunger', 'other', 'other
_parentinvolvement', 'other_performingarts', 'other_socialsciences', 'other_specialneeds', 'other_teamsports', 'other
_visualarts', 'other_warmth_care_hunger', 'parentinvolvement', 'parentinvolvement_performingarts', 'parentinvolvement
_socialsciences', 'parentinvolvement_specialneeds', 'parentinvolvement_teamsports', 'parentinvolvement_visualarts',
'parentinvolvement_warmth_care_hunger', 'performingarts', 'performingarts_socialsciences', 'performingarts_specialnee
ds', 'performingarts_teamsports', 'performingarts_visualarts', 'socialsciences', 'socialsciences_specialneeds', 'soci
alsciences_teamsports', 'socialsciences_visualarts', 'specialneeds', 'specialneeds_teamsports', 'specialneeds_visuala
rts', 'specialneeds_warmth_care_hunger', 'teamsports', 'teamsports_visualarts', 'visualarts', 'visualarts_warmth_care
_hunger', 'warmth_care_hunger']
==========================================================================================
```

2.5 Encoding Numerical Feature

Price

In [19]:
```
from sklearn.preprocessing import Normalizer
normalizer = Normalizer()
normalizer.fit(X_train['price'].values.reshape(1,-1))

X_train_price_norm = normalizer.transform(X_train['price'].values.reshape(1,-1))
X_test_price_norm = normalizer.transform(X_test['price'].values.reshape(1,-1))

X_train_price_norm=X_train_price_norm.reshape(-1,1)
X_test_price_norm=X_test_price_norm.reshape(-1,1)

print("After vectorizations")
print(X_train_price_norm.shape, y_train.shape)
print(X_test_price_norm.shape, y_test.shape)
print("="*100)
```

```
After vectorizations
(67000, 1) (67000,)
(33000, 1) (33000,)
====================================================================================================
```

teacher_number_of_previously_posted_projects

In [20]:
```
from sklearn.preprocessing import Normalizer
normalizer = Normalizer()
normalizer.fit(X_train['teacher_number_of_previously_posted_projects'].values.reshape(1,-1))

X_train_teachernumber_norm = normalizer.transform(X_train['teacher_number_of_previously_posted_projects'].values.reshape(1,-1))
X_test_teachernumber_norm = normalizer.transform(X_test['teacher_number_of_previously_posted_projects'].values.reshape(1,-1))

X_train_teachernumber_norm=X_train_teachernumber_norm.reshape(-1,1)
X_test_teachernumber_norm=X_test_teachernumber_norm.reshape(-1,1)

print("After vectorizations")
print(X_train_teachernumber_norm.shape, y_train.shape)
print(X_test_teachernumber_norm.shape, y_test.shape)
print("="*100)
```

```
After vectorizations
(67000, 1) (67000,)
(33000, 1) (33000,)
====================================================================================================
```

2.6 Sentiment Scores

In [21]:
```
import nltk
nltk.download('vader_lexicon')
```

```
[nltk_data] Downloading package vader_lexicon to
[nltk_data]     C:\Users\91888\AppData\Roaming\nltk_data...
[nltk_data]   Package vader_lexicon is already up-to-date!
```

Out[21]: True

In [22]:
```
from nltk.sentiment.vader import SentimentIntensityAnalyzer
```

In [23]:
```
sid = SentimentIntensityAnalyzer()
essay=X_train['processed_essay']
essay_sentiment1=[]
essay_sentiment2=[]
essay_sentiment3=[]
essay_sentiment4=[]
for i in tqdm(essay):
    score = sid.polarity_scores(i)
    essay_sentiment1.append(score['neg'])
    essay_sentiment2.append(score['neu'])
    essay_sentiment3.append(score['pos'])
    essay_sentiment4.append(score['compound'])
X_train['neg_sentiment_train'] = essay_sentiment1
X_train['neu_sentiment_train'] = essay_sentiment2
X_train['pos_sentiment_train'] = essay_sentiment3
X_train['compound_sentiment_train'] = essay_sentiment4
```

```
100%|██████████| 67000/67000 [03:27<00:00, 322.21it/s]
```

In [25]:
```
neg_sentiment_train=X_train['neg_sentiment_train'].values.reshape(-1,1)
neu_sentiment_train=X_train['neu_sentiment_train'].values.reshape(-1,1)
pos_sentiment_train=X_train['pos_sentiment_train'].values.reshape(-1,1)
compound_sentiment_train=X_train['compound_sentiment_train'].values.reshape(-1,1)
```

In [26]:
```
sid = SentimentIntensityAnalyzer()
essay=X_test['processed_essay']
essay_sentiment1=[]
essay_sentiment2=[]
essay_sentiment3=[]
essay_sentiment4=[]
for i in tqdm(essay):
    score = sid.polarity_scores(i)
    essay_sentiment1.append(score['neg'])
    essay_sentiment2.append(score['neu'])
    essay_sentiment3.append(score['pos'])
    essay_sentiment4.append(score['compound'])
X_test['neg_sentiment_test'] = essay_sentiment1
X_test['neu_sentiment_test'] = essay_sentiment2
X_test['pos_sentiment_test'] = essay_sentiment3
X_test['compound_sentiment_test'] = essay_sentiment4
```

```
100%|██████████| 33000/33000 [01:41<00:00, 326.22it/s]
```

In [27]:
```
neg_sentiment_test=X_test['neg_sentiment_test'].values.reshape(-1,1)
neu_sentiment_test=X_test['neu_sentiment_test'].values.reshape(-1,1)
pos_sentiment_test=X_test['pos_sentiment_test'].values.reshape(-1,1)
compound_sentiment_test=X_test['compound_sentiment_test'].values.reshape(-1,1)
```

3. Merging all features

set 1 tfidf

In [28]:
```
from scipy.sparse import hstack
X_tr = hstack((X_train_essay_tfidf,X_train_teacher_ohe,X_train_grade_ohe,X_train_state_ohe,X_train_category_ohe,X_train_subcategory_ohe,X_train_price_norm,X_train_teachernumber_norm,neg_sentiment_train,neu_sentiment_train,pos_sentiment_train,compound_sentiment_train)).tocsr()
X_te = hstack((X_test_essay_tfidf,X_test_teacher_ohe,X_test_grade_ohe,X_test_state_ohe,X_test_category_ohe,X_test_subcategory_ohe,X_test_price_norm,X_test_teachernumber_norm,neg_sentiment_test,neu_sentiment_test,pos_sentiment_test,compound_sentiment_test)).tocsr()
print("Final Data matrix")
print(X_tr.shape, y_train.shape)
print(X_te.shape, y_test.shape)
print("="*100)
```

```
Final Data matrix
(67000, 5505) (67000,)
(33000, 5505) (33000,)
====================================================================================================
```

Task - 1

3.1 Apply Decision Tree Classifier on SET 1

3.1.1 Hyper parameter Tuning

In [140]:
```
from sklearn.model_selection import GridSearchCV
from sklearn.tree import DecisionTreeClassifier

dc=DecisionTreeClassifier()
parameters={'max_depth':[1, 5, 10, 50],'min_samples_split':[5, 10, 100, 500]}

clf = GridSearchCV(dc, parameters, cv= 3, scoring='roc_auc',verbose=1,return_train_score=True,n_jobs=-1)
clf.fit(X_tr,y_train)

train_auc= clf.cv_results_['mean_train_score']
train_auc_std= clf.cv_results_['std_train_score']
cv_auc = clf.cv_results_['mean_test_score']
cv_auc_std= clf.cv_results_['std_test_score']
bestMaxDepth=clf.best_params_['max_depth']
bestMinSampleSplit=clf.best_params_['min_samples_split']
bestScore=clf.best_score_

print("best Max depth",bestMaxDepth)
print("best min sample split",bestMinSampleSplit)
print("best score",bestScore)
```

```
Fitting 3 folds for each of 16 candidates, totalling 48 fits

[Parallel(n_jobs=-1)]: Using backend LokyBackend with 8 concurrent workers.
[Parallel(n_jobs=-1)]: Done  48 out of  48 | elapsed:  4.3min finished

best Max depth 10
best min sample split 500
best score 0.6458170607104589
```

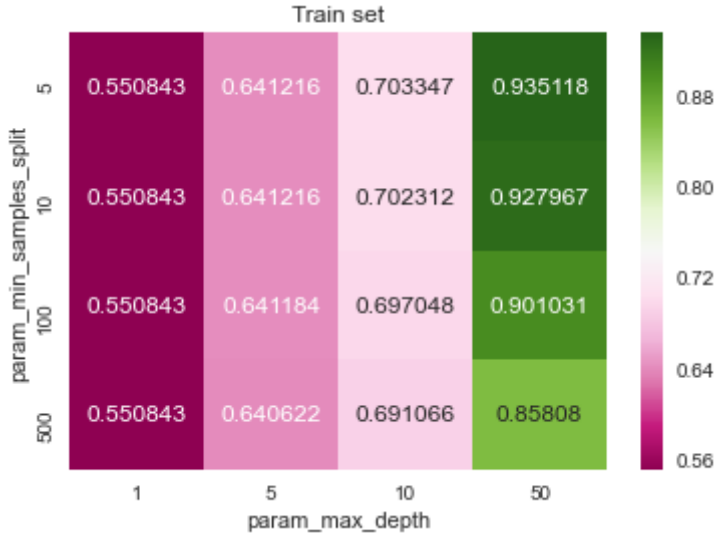
3.1.2 Representation of results

In [159]:
```
#converting results to list
```

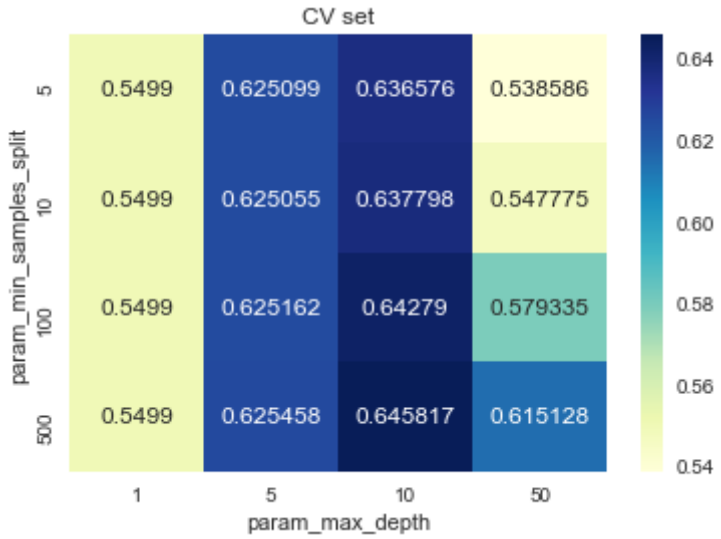
In [141]:
```
trscore=clf.cv_results_['mean_train_score']
trscore.tolist()
testscore=clf.cv_results_['mean_test_score']
testscore.tolist()
sample=clf.cv_results_['param_min_samples_split']
sample.tolist()
depth=clf.cv_results_['param_max_depth']
print(depth.tolist())
```

Out[141]:
```
[1, 1, 1, 1, 5, 5, 5, 5, 10, 10, 10, 10, 50, 50, 50, 50]
```

In [158]:
```
#https://stackoverflow.com/questions/37790429/seaborn-heatmap-using-pandas-dataframe
df = pd.DataFrame({'trscore': trscore, 'param_min_samples_split':sample, 'param_max_depth':depth})
result = df.pivot(index='param_min_samples_split', columns='param_max_depth', values='trscore')
sns.heatmap(result, annot=True, fmt="g",cmap='PiYG')
plt.title("Train set")
plt.show()
```

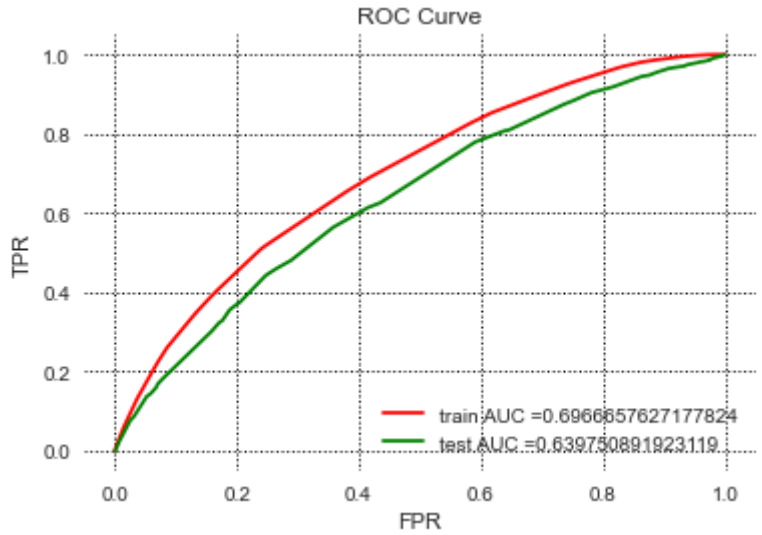


In [155]:
```
#test result
#https://stackoverflow.com/questions/37790429/seaborn-heatmap-using-pandas-dataframe
df = pd.DataFrame({'tescore': testscore, 'param_min_samples_split':sample, 'param_max_depth':depth})
result = df.pivot(index='param_min_samples_split', columns='param_max_depth', values='tescore')
sns.heatmap(result, annot=True, fmt="g",cmap="YlGnBu")
plt.title("CV set")
plt.show()
```



3.1.3 ROC CURVE

In [160]:
```
dc_test = DecisionTreeClassifier(min_samples_split=bestMinSampleSplit,max_depth=bestMaxDepth)
dc_test.fit(X_tr,y_train)

y_train_pred = dc_test.predict_proba( X_tr)[:, 1]
y_test_pred = dc_test.predict_proba(X_te)[:, 1]
train_fpr, train_tpr, tr_thresholds =  metrics.roc_curve(y_train, y_train_pred)
test_fpr, test_tpr, te_thresholds = metrics.roc_curve(y_test, y_test_pred)

#Plot curve :
ab=plt.subplot()
plt.plot(train_fpr, train_tpr,color='r',label="train AUC ="+str(metrics.auc(train_fpr, train_tpr)))
plt.plot(test_fpr, test_tpr,color='g', label="test AUC ="+str(metrics.auc(test_fpr, test_tpr)))
plt.legend(loc='lower right')
plt.xlabel("FPR")
plt.ylabel("TPR")
plt.title("ROC Curve")
plt.grid(b=True, which='major', color='k', linestyle=':')
ab.set_facecolor("white")
plt.show()
```



3.1.4 Confusion Matrix

In [161]:
```
#finding best threshold : https://stats.stackexchange.com/questions/123124/how-to-determine-the-optimal-threshold-for-a-classifier-and-generate-roc-curve
from sklearn.metrics import roc_curve, auc
fpr, tpr, thresholds = roc_curve(y_test, y_test_pred)
optimal_idx = np.argmax(tpr - fpr)
optimal_threshold = thresholds[optimal_idx]
print("Threshold value is:", np.round(optimal_threshold,3))
```

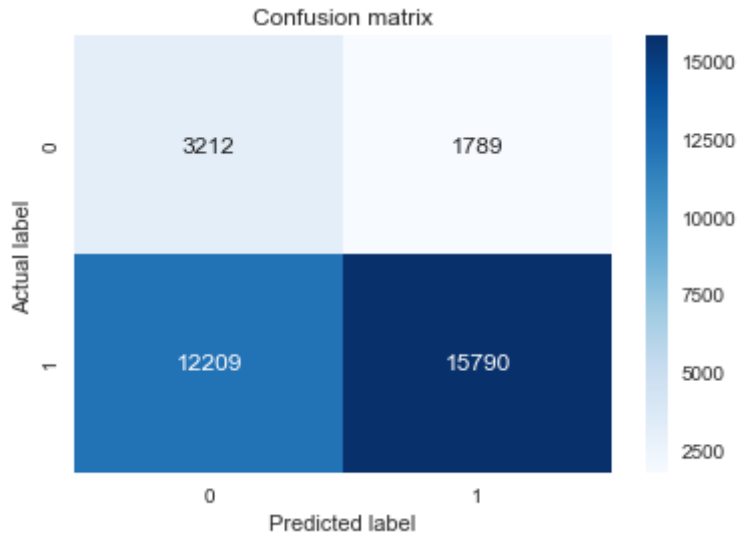
```
Threshold value is: 0.852
```

In [162]:
```
def predict(proba, threshould):
    predictions = []
    for i in proba:
        if i>=threshould:
            predictions.append(1)
        else:
            predictions.append(0)
    return predictions
prediction = predict(y_test_pred,optimal_threshold)
```

In [163]:
```
from sklearn.metrics import confusion_matrix
matrix = confusion_matrix(y_test,prediction)
print('Confusion matrix : \n',matrix)
```

```
Confusion matrix :
 [[ 3212  1789]
 [12209 15790]]
```

In [165]:
```
import seaborn as sns
import matplotlib.pyplot as plt

sns.heatmap(matrix, annot=True,cmap='Blues',fmt='g')
plt.xlabel('Predicted label')
plt.ylabel('Actual label')
plt.title('Confusion matrix')
plt.show()
```

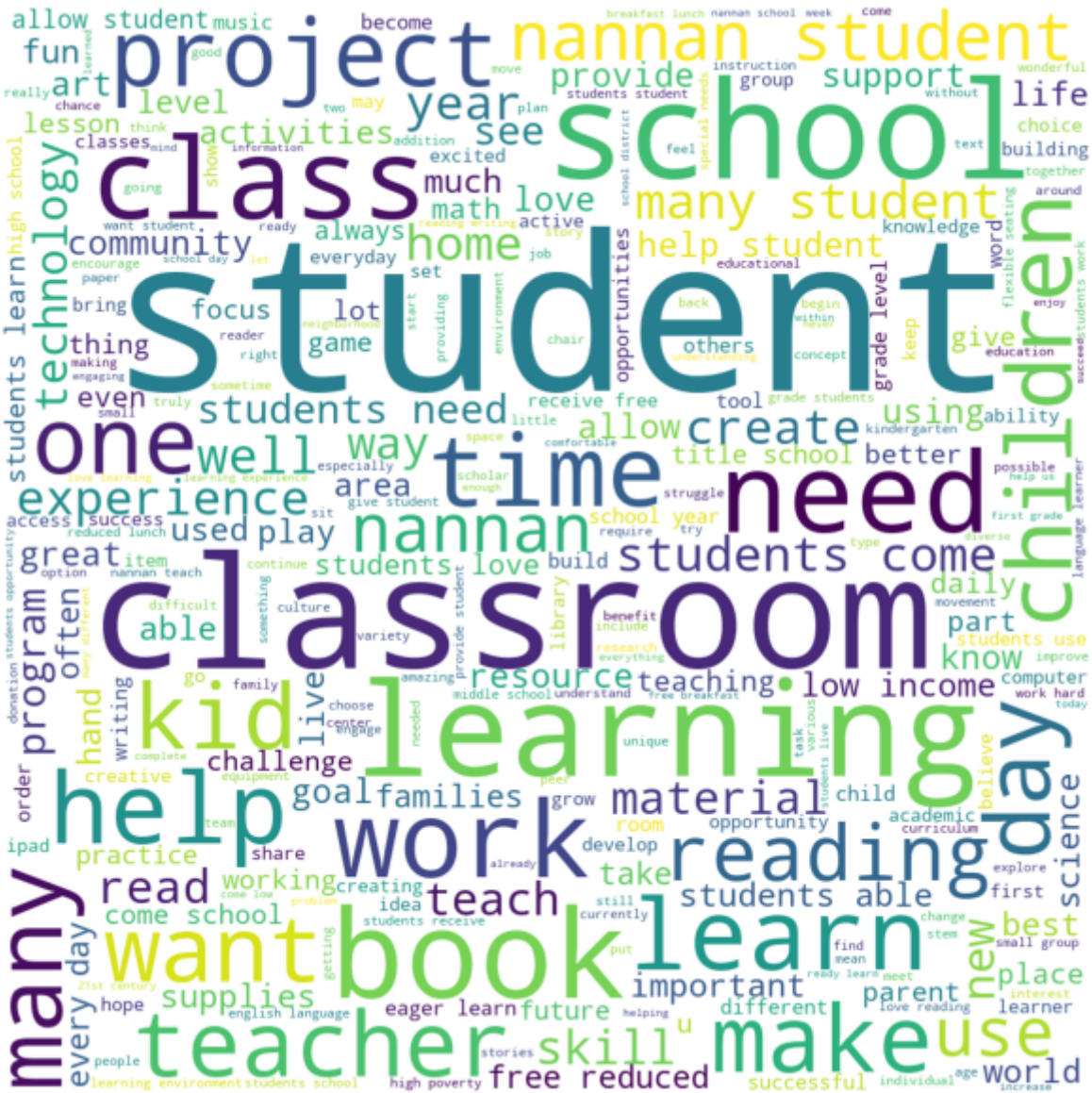


3.1.5 Word Cloud

get False Positive Data Points

In [167]:
```
data1={'Actual':y_test,'Predicted':prediction}
df=pd.DataFrame(data1)

indices=list(df[(df['Actual']==0) & (df['Predicted']==1)].index)
```

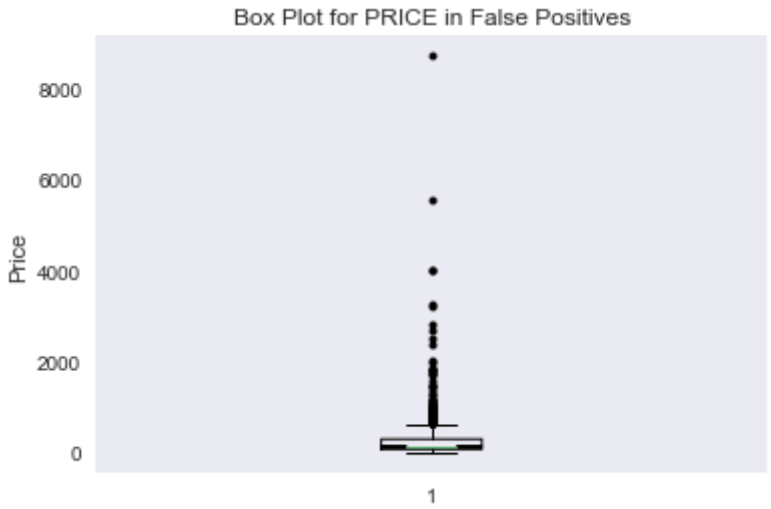
In [168]:
```
#https://stackoverflow.com/questions/45588724/generating-word-cloud-for-items-in-a-list-in-python
Essay_Tests=""
for i in indices:

    Essay_Tests+=" {}".format(X_test.iloc[i]['processed_essay'])

from wordcloud import WordCloud, STOPWORDS

wc = WordCloud(width = 1000, height = 1000,background_color="white", max_words=len(Essay_Tests), stopwords=STOPWORDS,m
in_font_size = 10)
wc.generate(Essay_Tests)
plt.figure(figsize = (8, 8), facecolor = None)
plt.imshow(wc)
plt.axis("off")
plt.tight_layout(pad = 0)
```



3.1.6 Box Plot with price of FP datapoints

In [169]:
```
price=[]
for i in indices:
    price.append(X_test.iloc[i]['price'])
```

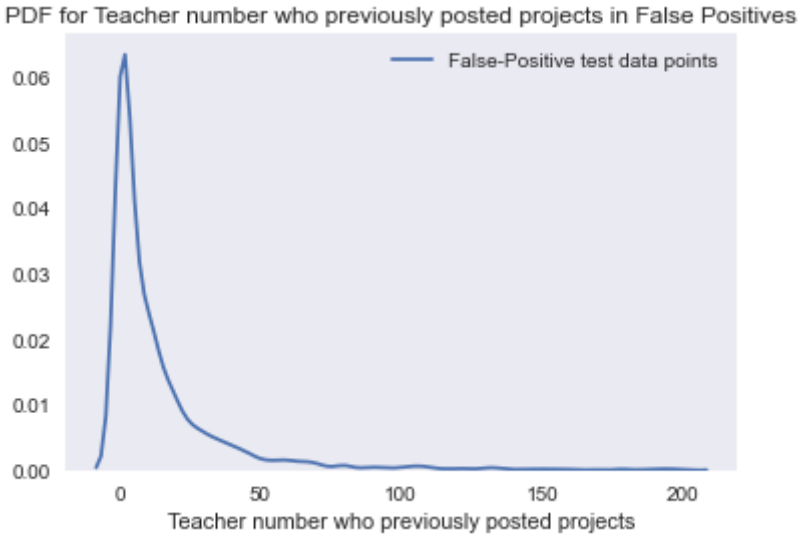
In [171]:
```
plt.boxplot(price,sym='k.',notch=True)
plt.title('Box Plot for PRICE in False Positives')
plt.ylabel('Price')
plt.grid()
plt.show()
```



3.1.7 PDF with teacher_number_of_previously_posted_projects of FP datapoints

In [172]:
```
teacher_project=[]
for i in indices:
    teacher_project.append(X_test.iloc[i]['teacher_number_of_previously_posted_projects'])
```

In [173]:
```
import seaborn as sns

legend= 'False-Positive test data points'
ax=sns.distplot(teacher_project,label=legend,hist=False)
plt.title('PDF for Teacher number who previously posted projects in False Positives')
plt.xlabel('Teacher number who previously posted projects')
plt.grid()
plt.show()
```



Set 2 tfidf w2v

In [174]:
```
from scipy.sparse import hstack
X_tr1 = hstack((train_tfidf_w2v_vectors,X_train_teacher_ohe,X_train_grade_ohe,X_train_state_ohe,X_train_category_ohe,X
_train_subcategory_ohe,X_train_price_norm,X_train_teachernumber_norm,neg_sentiment_train,neu_sentiment_train,pos_senti
ment_train,compound_sentiment_train)).tocsr()
X_te1 = hstack((test_tfidf_w2v_vectors,X_test_teacher_ohe,X_test_grade_ohe,X_test_state_ohe,X_test_category_ohe,X_test
_subcategory_ohe,X_test_price_norm,X_test_teachernumber_norm,neg_sentiment_test,neu_sentiment_test,pos_sentiment_test,
compound_sentiment_test)).tocsr()
print("Final Data matrix")
print(X_tr1.shape, y_train.shape)
print(X_te1.shape, y_test.shape)
print("="*100)
```

```
Final Data matrix
(67000, 805) (67000,)
(33000, 805) (33000,)
====================================================================================================
```

3.2 Apply Decision Tree Classifier on SET 2

3.2.1 Hyper parameter Tuning

In [175]:
```
from sklearn.model_selection import GridSearchCV
from sklearn.tree import DecisionTreeClassifier

dc=DecisionTreeClassifier()
parameters={'max_depth':[1, 5, 10, 50],'min_samples_split':[5, 10, 100, 500]}

clf1 = GridSearchCV(dc, parameters, cv= 3, scoring='roc_auc',verbose=1,return_train_score=True,n_jobs=-1)
clf1.fit(X_tr1,y_train)

train_auc= clf1.cv_results_['mean_train_score']
train_auc_std= clf1.cv_results_['std_train_score']
cv_auc = clf1.cv_results_['mean_test_score']
cv_auc_std= clf1.cv_results_['std_test_score']
bestMaxDepth=clf1.best_params_['max_depth']
bestMinSampleSplit=clf1.best_params_['min_samples_split']
bestScore=clf1.best_score_

print("best Max depth",bestMaxDepth)
print("best min sample split",bestMinSampleSplit)
print("best score",bestScore)
```

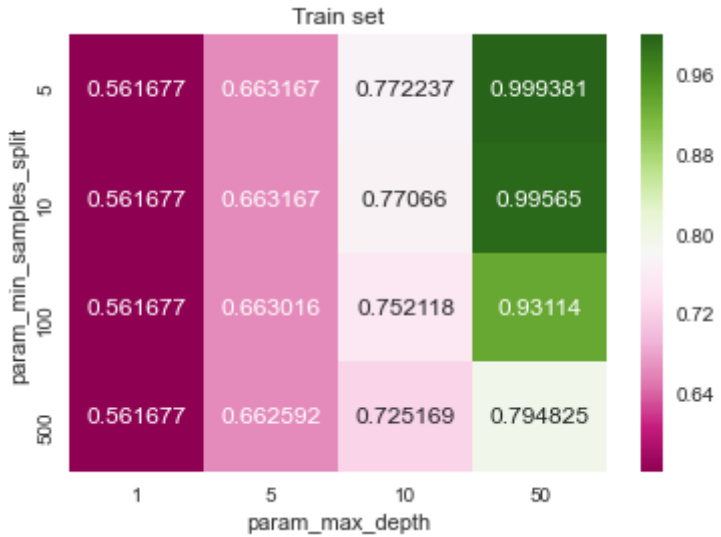
```
Fitting 3 folds for each of 16 candidates, totalling 48 fits

[Parallel(n_jobs=-1)]: Using backend LokyBackend with 8 concurrent workers.
[Parallel(n_jobs=-1)]: Done  48 out of  48 | elapsed: 11.1min finished

best Max depth 5
best min sample split 10
best score 0.6268988204922107
```

3.2.2 Representation of results

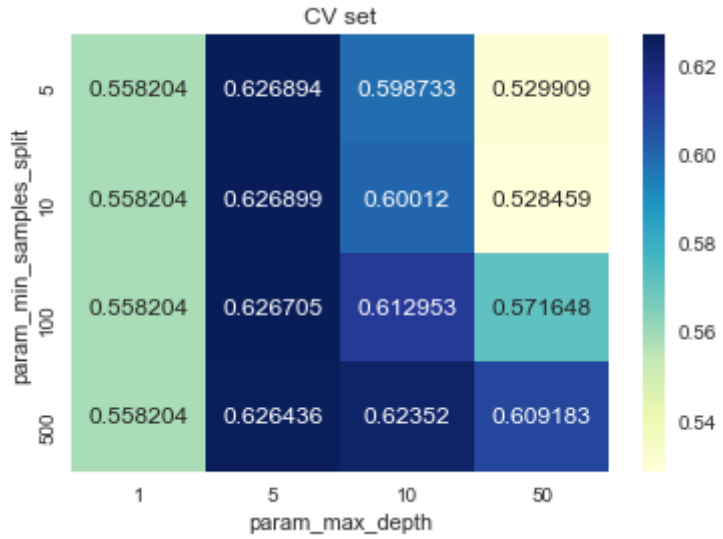
In [179]:
```
trscore=clf1.cv_results_['mean_train_score']
trscore.tolist()
testscore=clf1.cv_results_['mean_test_score']
testscore.tolist()
sample=clf1.cv_results_['param_min_samples_split']
sample.tolist()
depth=clf1.cv_results_['param_max_depth']
print(depth.tolist())
```

```
[1, 1, 1, 1, 5, 5, 5, 5, 10, 10, 10, 10, 50, 50, 50, 50]
```

In [178]:
```
#https://stackoverflow.com/questions/37790429/seaborn-heatmap-using-pandas-dataframe
df = pd.DataFrame({'trscore': trscore, 'param_min_samples_split':sample, 'param_max_depth':depth})
result = df.pivot(index='param_min_samples_split', columns='param_max_depth', values='trscore')
sns.heatmap(result, annot=True, fmt="g",cmap='PiYG')
plt.title("Train set")
plt.show()
```

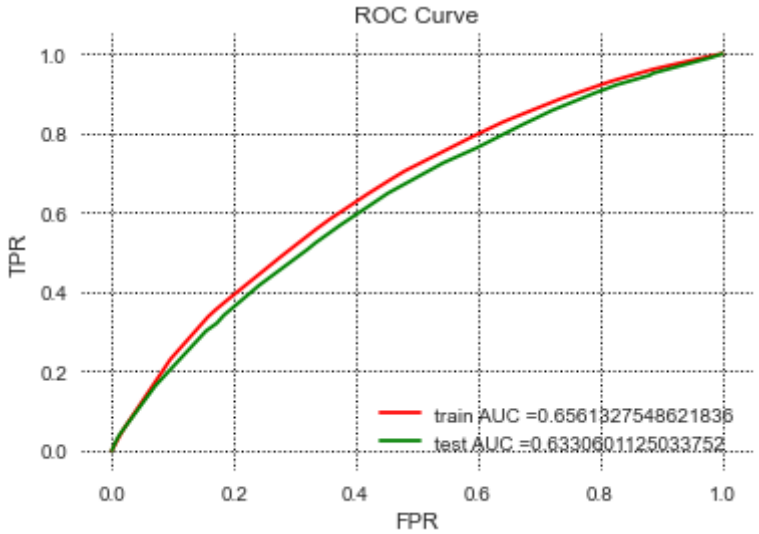


In [180]:
```
#test result
#https://stackoverflow.com/questions/37790429/seaborn-heatmap-using-pandas-dataframe
df = pd.DataFrame({'tescore': testscore, 'param_min_samples_split':sample, 'param_max_depth':depth})
result = df.pivot(index='param_min_samples_split', columns='param_max_depth', values='tescore')
sns.heatmap(result, annot=True, fmt="g",cmap="YlGnBu")
plt.title("CV set")
plt.show()
```



3.2.3 ROC CURVE

In [182]:
```
dc_test = DecisionTreeClassifier(min_samples_split=bestMinSampleSplit,max_depth=bestMaxDepth)
dc_test.fit(X_tr1,y_train)

y_train_pred = dc_test.predict_proba( X_tr1)[:, 1]
y_test_pred = dc_test.predict_proba(X_te1)[:, 1]
train_fpr, train_tpr, tr_thresholds =  metrics.roc_curve(y_train, y_train_pred)
test_fpr, test_tpr, te_thresholds = metrics.roc_curve(y_test, y_test_pred)

#Plot curve :
ab=plt.subplot()
plt.plot(train_fpr, train_tpr,color='r',label="train AUC ="+str(metrics.auc(train_fpr, train_tpr)))
plt.plot(test_fpr, test_tpr,color='g', label="test AUC ="+str(metrics.auc(test_fpr, test_tpr)))
plt.legend(loc='lower right')
plt.xlabel("FPR")
plt.ylabel("TPR")
plt.title("ROC Curve")
plt.grid(b=True, which='major', color='k', linestyle=':')
ab.set_facecolor("white")
plt.show()
```



3.2.4 Confusion Matrix

In [183]:
```
from sklearn.metrics import roc_curve, auc
fpr, tpr, thresholds = roc_curve(y_test, y_test_pred)
optimal_idx = np.argmax(tpr - fpr)
optimal_threshold = thresholds[optimal_idx]
print("Threshold value is:", np.round(optimal_threshold,3))
```

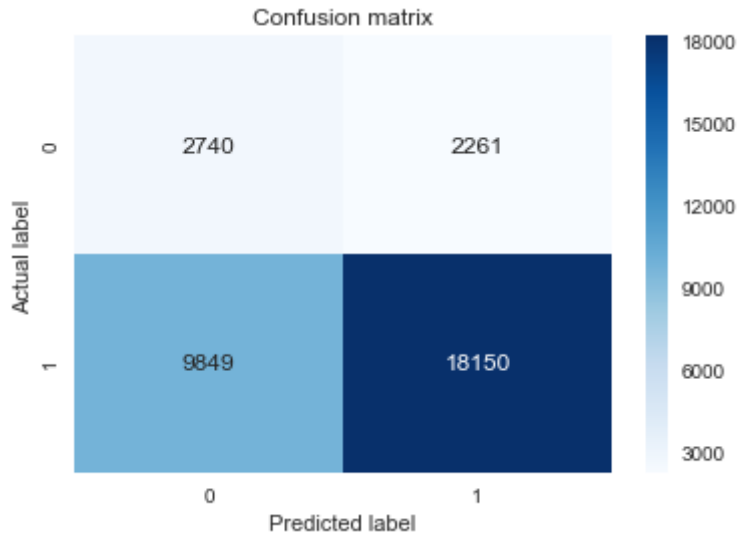
```
Threshold value is: 0.839
```

In [184]:
```
def predict(proba, threshould):
    predictions = []
    for i in proba:
        if i>=threshould:
            predictions.append(1)
        else:
            predictions.append(0)
    return predictions
prediction = predict(y_test_pred,optimal_threshold)
```

In [188]:
```
from sklearn.metrics import confusion_matrix
matrix = confusion_matrix(y_test,prediction)
print('Confusion matrix : \n',matrix)
```

```
Confusion matrix :
 [[ 2740  2261]
 [ 9849 18150]]
```

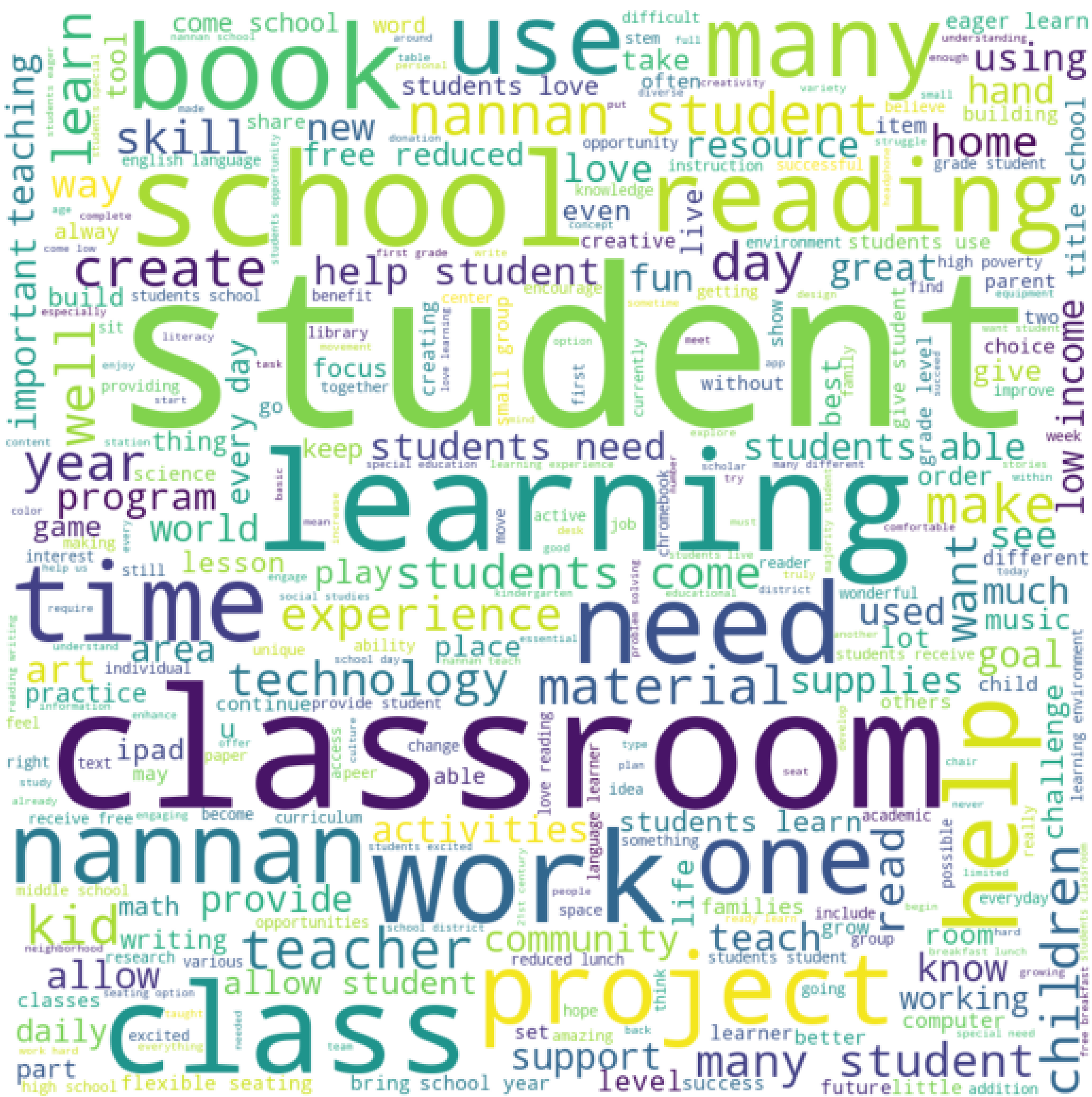
In [189]:
```
import seaborn as sns
import matplotlib.pyplot as plt

sns.heatmap(matrix, annot=True,cmap='Blues',fmt='g')
plt.xlabel('Predicted label')
plt.ylabel('Actual label')
plt.title('Confusion matrix')
plt.show()
```



3.2.5 Word Cloud

In [190]:
```
data1={'Actual':y_test,'Predicted':prediction}
df=pd.DataFrame(data1)

indices=list(df[(df['Actual']==0) & (df['Predicted']==1)].index)
```

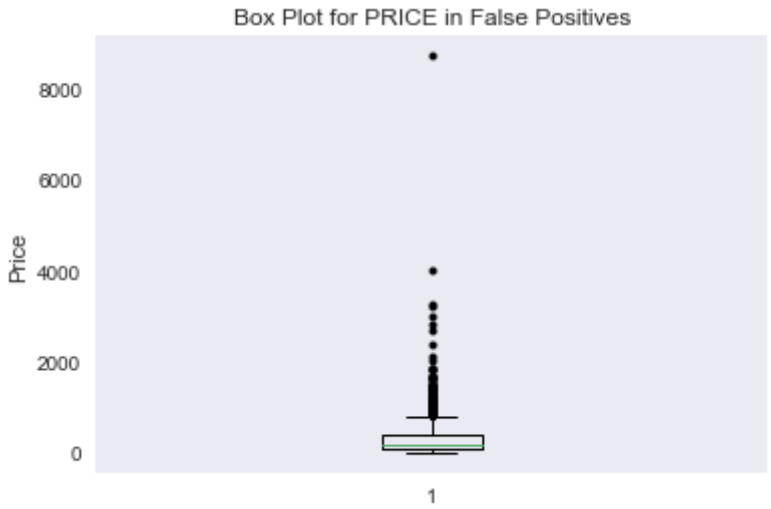
In [191]:
```
#https://stackoverflow.com/questions/45588724/generating-word-cloud-for-items-in-a-list-in-python
Essay_Tests=""
for i in indices:

    Essay_Tests+=" {}".format(X_test.iloc[i]['processed_essay'])

from wordcloud import WordCloud, STOPWORDS

wc = WordCloud(width = 1000, height = 1000,background_color="white", max_words=len(Essay_Tests), stopwords=STOPWORDS,m
in_font_size = 10)
wc.generate(Essay_Tests)
plt.figure(figsize = (8, 8), facecolor = None)
plt.imshow(wc)
plt.axis("off")
plt.tight_layout(pad = 0)
```



3.2.6 Box Plot with price of FP datapoints

In [192]:
```
price=[]
for i in indices:
    price.append(X_test.iloc[i]['price'])
```

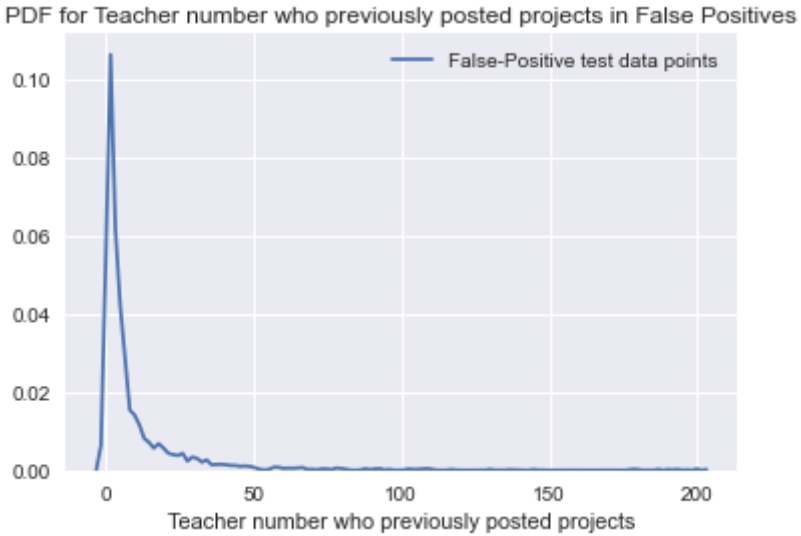
In [193]:
```
plt.boxplot(price,sym='k.')
plt.title('Box Plot for PRICE in False Positives')
plt.ylabel('Price')
plt.grid()
plt.show()
```



3.2.7 PDF with teacher_number_of_previously_posted_projects of FP datapoints

In [194]:
```
teacher_project=[]
for i in indices:
    teacher_project.append(X_test.iloc[i]['teacher_number_of_previously_posted_projects'])
```

In [195]:
```
import seaborn as sns

legend= 'False-Positive test data points'
ax=sns.distplot(teacher_project,label=legend,hist=False)
plt.title('PDF for Teacher number who previously posted projects in False Positives')
plt.xlabel('Teacher number who previously posted projects')
plt.show()
```



Task - 2

In [196]:
```
#finding feature importance for every feature
dc=DecisionTreeClassifier()
tree = dc.fit(X_tr,y_train)
tree.feature_importances_
```

Out[196]:
```
array([0.        , 0.00010716, 0.        , ..., 0.00581519, 0.00525188,
       0.00401562])
```

Creating dataset with non zero feature importance features

In [197]:
```
X_tr_feaimp=X_tr[:,tree.feature_importances_!=0]
```

In [198]:
```
#train data
X_tr_feaimp
```

Out[198]:
```
<67000x2730 sparse matrix of type '<class 'numpy.float64'>'
        with 6062178 stored elements in Compressed Sparse Row format>
```

In [199]:
```
X_test_feaimp=X_te[:,tree.feature_importances_!=0]
```

In [200]:
```
#test data
X_test_feaimp
```

Out[200]:
```
<33000x2730 sparse matrix of type '<class 'numpy.float64'>'
        with 2982769 stored elements in Compressed Sparse Row format>
```

In [201]:
```
#number of non zero features
X_tr_feaimp.shape
```

Out[201]:
```
(67000, 2730)
```

Apply model on feature having non zero feature importance features

Hyper parameter Tuning

In [202]:
```
from sklearn.model_selection import GridSearchCV
from sklearn.tree import DecisionTreeClassifier

dc=DecisionTreeClassifier()
parameters={'max_depth':[1, 5, 10, 50],'min_samples_split':[5, 10, 100, 500]}

clf2 = GridSearchCV(dc, parameters, cv= 3, scoring='roc_auc',verbose=1,return_train_score=True,n_jobs=-1)
clf2.fit(X_tr_feaimp,y_train)

train_auc= clf2.cv_results_['mean_train_score']
train_auc_std= clf2.cv_results_['std_train_score']
cv_auc = clf2.cv_results_['mean_test_score']
cv_auc_std= clf2.cv_results_['std_test_score']
bestMaxDepth=clf2.best_params_['max_depth']
bestMinSampleSplit=clf2.best_params_['min_samples_split']
bestScore=clf2.best_score_

print("best Max depth",bestMaxDepth)
print("best min sample split",bestMinSampleSplit)
print("best score",bestScore)
```

```
Fitting 3 folds for each of 16 candidates, totalling 48 fits

[Parallel(n_jobs=-1)]: Using backend LokyBackend with 8 concurrent workers.
[Parallel(n_jobs=-1)]: Done  48 out of  48 | elapsed:  3.2min finished

best Max depth 10
best min sample split 500
best score 0.6462414527602206
```

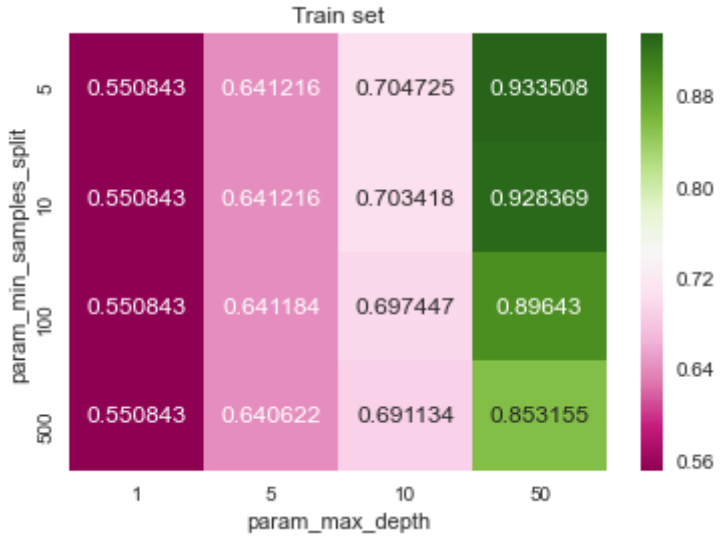
Representation of results

In [ ]:
```
#converting results to list
```

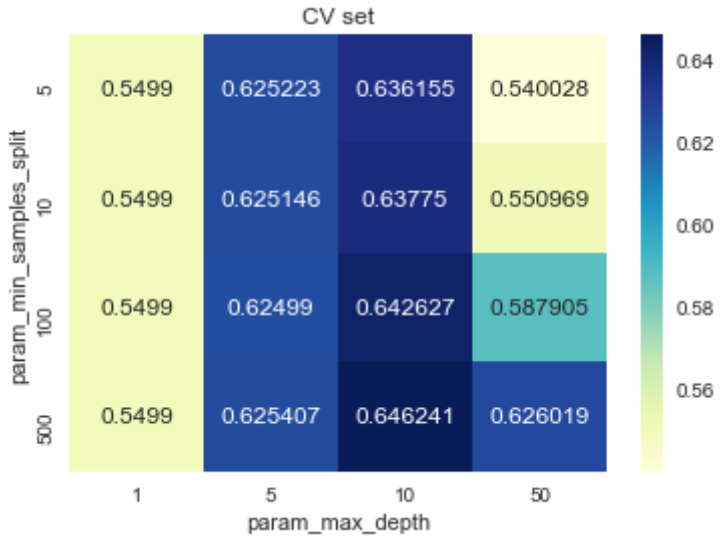
In [203]:
```
trscore=clf2.cv_results_['mean_train_score']
trscore.tolist()
testscore=clf2.cv_results_['mean_test_score']
testscore.tolist()
sample=clf2.cv_results_['param_min_samples_split']
sample.tolist()
depth=clf2.cv_results_['param_max_depth']
print(depth.tolist())
```

```
[1, 1, 1, 1, 5, 5, 5, 5, 10, 10, 10, 10, 50, 50, 50, 50]
```

In [204]:
```
#https://stackoverflow.com/questions/37790429/seaborn-heatmap-using-pandas-dataframe
df = pd.DataFrame({'trscore': trscore, 'param_min_samples_split':sample, 'param_max_depth':depth})
result = df.pivot(index='param_min_samples_split', columns='param_max_depth', values='trscore')
sns.heatmap(result, annot=True, fmt="g",cmap='PiYG')
plt.title("Train set")
plt.show()
```

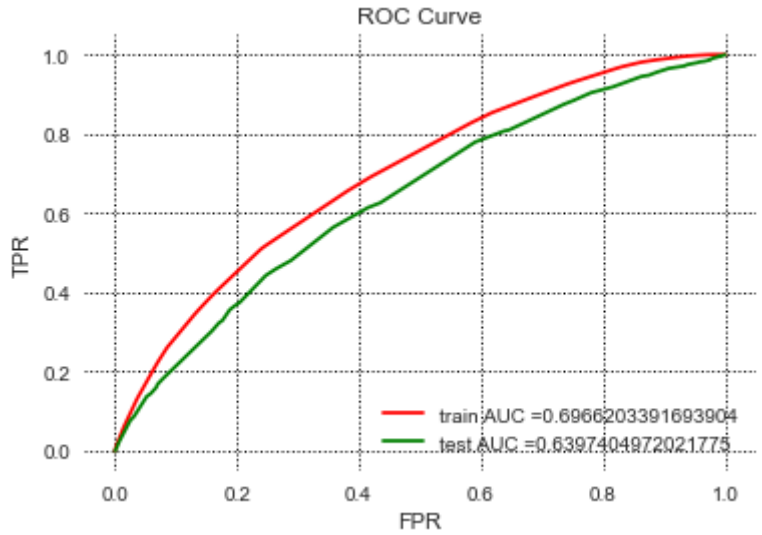


In [205]:
```
#test result
#https://stackoverflow.com/questions/37790429/seaborn-heatmap-using-pandas-dataframe
df = pd.DataFrame({'tescore': testscore, 'param_min_samples_split':sample, 'param_max_depth':depth})
result = df.pivot(index='param_min_samples_split', columns='param_max_depth', values='tescore')
sns.heatmap(result, annot=True, fmt="g",cmap="YlGnBu")
plt.title("CV set")
plt.show()
```



ROC CURVE

In [206]:
```
dc_test = DecisionTreeClassifier(min_samples_split=bestMinSampleSplit,max_depth=bestMaxDepth)
dc_test.fit(X_tr_feaimp,y_train)

y_train_pred = dc_test.predict_proba(X_tr_feaimp)[:, 1]
y_test_pred = dc_test.predict_proba(X_test_feaimp)[:, 1]
train_fpr, train_tpr, tr_thresholds =  metrics.roc_curve(y_train, y_train_pred)
test_fpr, test_tpr, te_thresholds = metrics.roc_curve(y_test, y_test_pred)

#Plot curve :
ab=plt.subplot()
plt.plot(train_fpr, train_tpr,color='r',label="train AUC ="+str(metrics.auc(train_fpr, train_tpr)))
plt.plot(test_fpr, test_tpr,color='g', label="test AUC ="+str(metrics.auc(test_fpr, test_tpr)))
plt.legend(loc='lower right')
plt.xlabel("FPR")
plt.ylabel("TPR")
plt.title("ROC Curve")
plt.grid(b=True, which='major', color='k', linestyle=':')
ab.set_facecolor("white")
plt.show()
```



Confusion Matrix

In [207]:
```
from sklearn.metrics import roc_curve, auc
fpr, tpr, thresholds = roc_curve(y_test, y_test_pred)
optimal_idx = np.argmax(tpr - fpr)
optimal_threshold = thresholds[optimal_idx]
print("Threshold value is:", np.round(optimal_threshold,3))
```

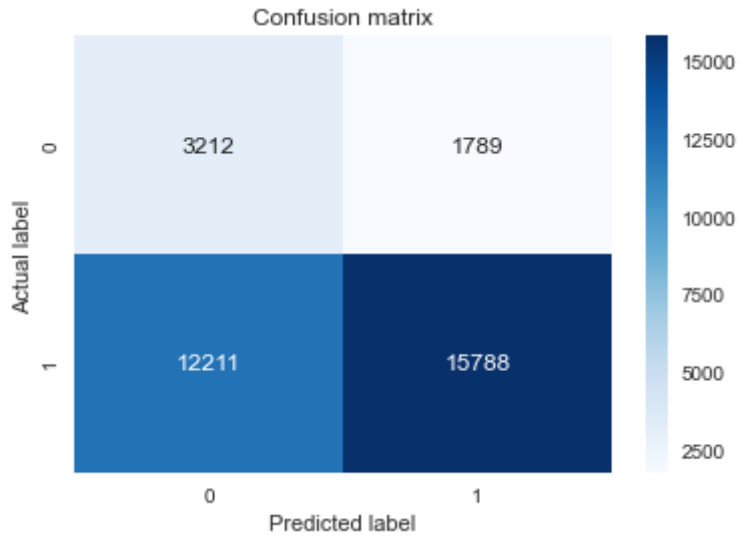
```
Threshold value is: 0.852
```

In [208]:
```
def predict(proba, threshould):
    predictions = []
    for i in proba:
        if i>=threshould:
            predictions.append(1)
        else:
            predictions.append(0)
    return predictions
prediction = predict(y_test_pred,optimal_threshold)
```

In [209]:
```
from sklearn.metrics import confusion_matrix
matrix = confusion_matrix(y_test,prediction)
print('Confusion matrix : \n',matrix)
```

```
Confusion matrix :
 [[ 3212  1789]
 [12211 15788]]
```

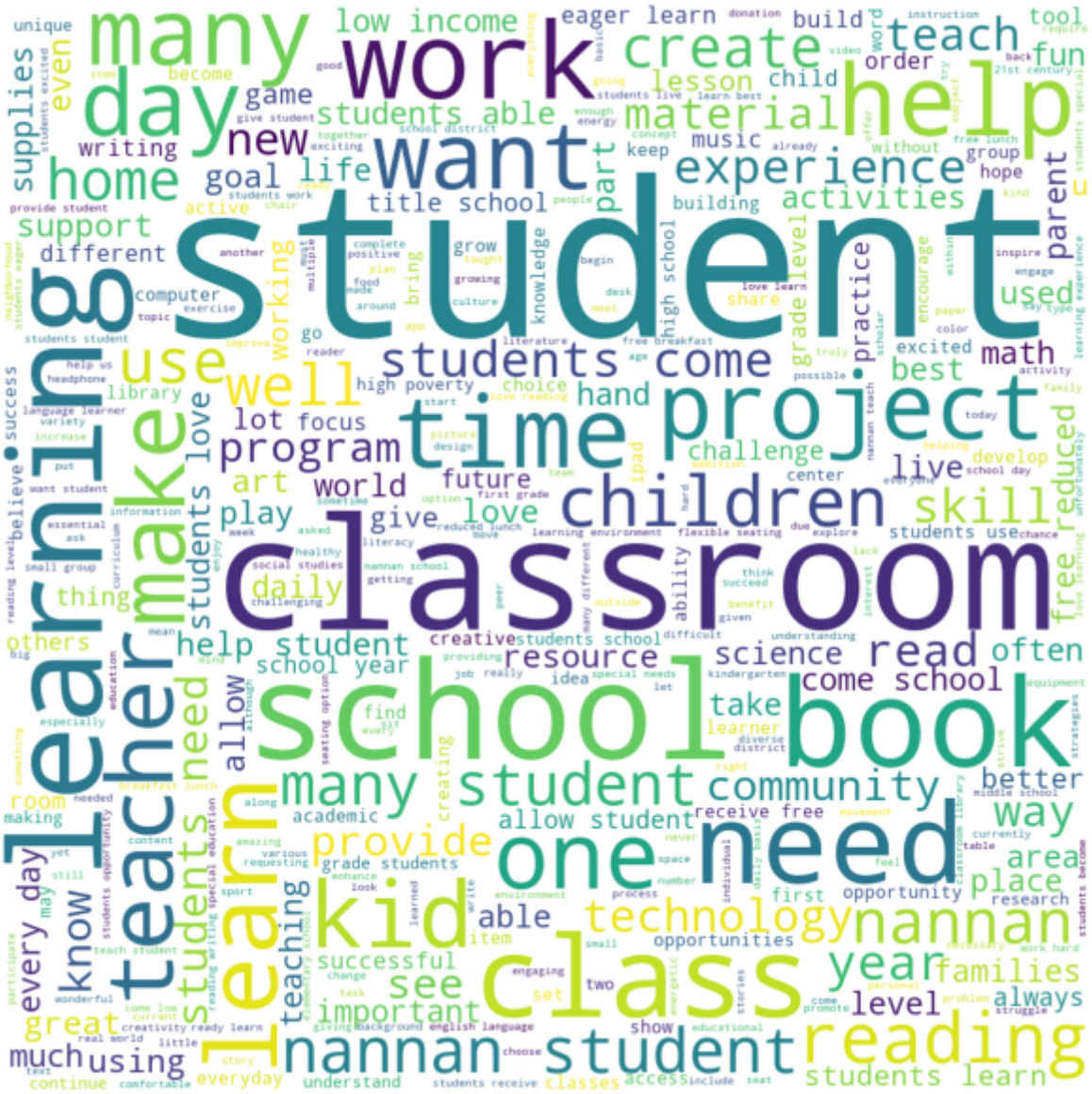
In [210]:
```
import seaborn as sns
import matplotlib.pyplot as plt

sns.heatmap(matrix, annot=True,cmap='Blues',fmt='g')
plt.xlabel('Predicted label')
plt.ylabel('Actual label')
plt.title('Confusion matrix')
plt.show()
```



Word Cloud

In [211]:
```
data1={'Actual':y_test,'Predicted':prediction}
df=pd.DataFrame(data1)

indices=list(df[(df['Actual']==0) & (df['Predicted']==1)].index)
```

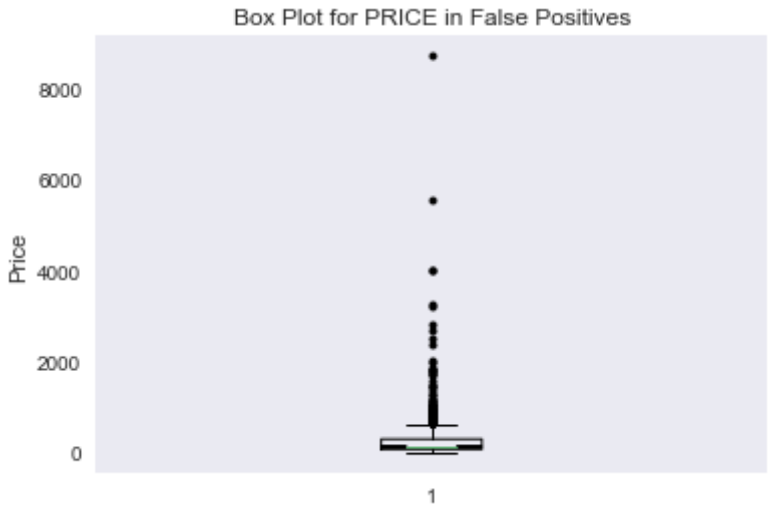
In [212]:
```
#https://stackoverflow.com/questions/45588724/generating-word-cloud-for-items-in-a-list-in-python
Essay_Tests=""
for i in indices:

    Essay_Tests+=" {}".format(X_test.iloc[i]['processed_essay'])

from wordcloud import WordCloud, STOPWORDS

wc = WordCloud(width = 1000, height = 1000,background_color="white", max_words=len(Essay_Tests), stopwords=STOPWORDS,m
in_font_size = 10)
wc.generate(Essay_Tests)
plt.figure(figsize = (8, 8), facecolor = None)
plt.imshow(wc)
plt.axis("off")
plt.tight_layout(pad = 0)
```



Box Plot with price of FP datapoints

In [213]:
```
price=[]
for i in indices:
    price.append(X_test.iloc[i]['price'])
```

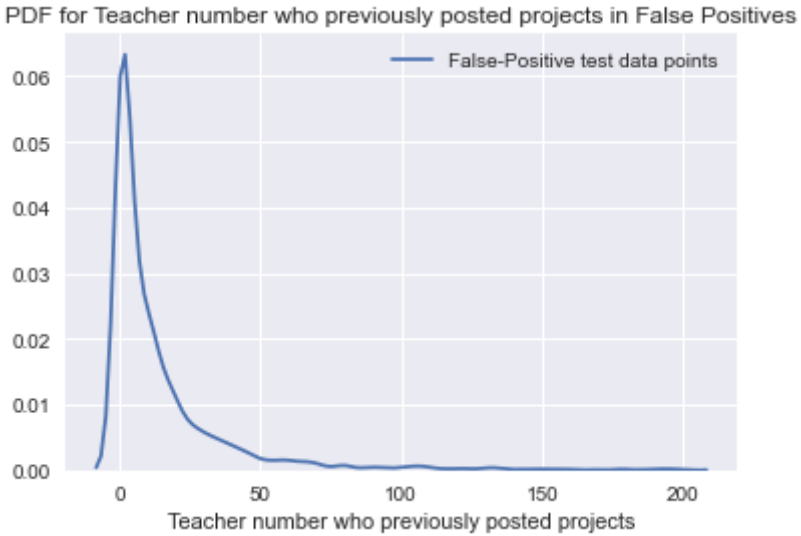
In [214]:
```
plt.boxplot(price,sym='k.',notch=True)
plt.title('Box Plot for PRICE in False Positives')
plt.ylabel('Price')
plt.grid()
plt.show()
```



PDF with teacher_number_of_previously_posted_projects of FP datapoints

In [215]:
```
teacher_project=[]
for i in indices:
    teacher_project.append(X_test.iloc[i]['teacher_number_of_previously_posted_projects'])
```

In [216]:
```
import seaborn as sns

legend= 'False-Positive test data points'
ax=sns.distplot(teacher_project,label=legend,hist=False)
plt.title('PDF for Teacher number who previously posted projects in False Positives')
plt.xlabel('Teacher number who previously posted projects')
plt.show()
```



conclusion

In [219]:
```
#http://zetcode.com/python/prettytable/

from prettytable import PrettyTable

x = PrettyTable()

x.field_names = ["Vectorizer","Model", "Min. Sample Split", "Max Dept", "Test AUC"]

x.add_row(["TF-IDF", "DecisionTreeClassifier", 500, 10,0.63975])
x.add_row(["TF-IDF W2V", "DecisionTreeClassifier",10,5,0.63306])
x.add_row(["TFIDF-FeatureImp", "DecisionTreeClassifier", 500,10, 0.639740])


print(x)
```

```
+------------------+-----------------------+-------------------+----------+----------+
|    Vectorizer    |         Model         | Min. Sample Split | Max Dept | Test AUC |
+------------------+-----------------------+-------------------+----------+----------+
|      TF-IDF      | DecisionTreeClassifier |        500        |    10    | 0.63975  |
|    TF-IDF W2V    | DecisionTreeClassifier |         10        |    5     | 0.63306  |
| TFIDF-FeatureImp | DecisionTreeClassifier |        500        |    10    | 0.63974  |
+------------------+-----------------------+-------------------+----------+----------+
```

In [ ]:
```
!jupyter nbconvert --to html DecisionTree_solve.ipynb
```