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	
`project_id`	A unique identifier for the proposed project
`project_title`	Title of tl • `Art Wil` •
`project_grade_category`	Grade level of students for which the project is targetec • • • •

Feature	
`project_subject_categories`	One or more (comma-separated) subject categories f following enur • • • • • `Li` • • • • • • `Literacy & Languag`
`school_state`	State where school is located ([Two- (https://en.wikipedia.org/wiki/List_of_U.S._state_abbrevia](https://en.wikipedia.org/wiki/List_of_U.S._state_abbrevia)
`project_subject_subcategories`	One or more (comma-separated) subject subcat • • `Literature & Writing`
`project_resource_summary`	An explanation of the resources needed for t • `My students need hands on literacy ma sen`
`project_essay_1`	F
`project_essay_2`	Sec
`project_essay_3`	Tl
`project_essay_4`	Fou
`project_submitted_datetime`	Datetime when project application was submitted. **Ex**
`teacher_id`	A unique identifier for the teacher of the propo `bdf8baa8fedef6b`
`teacher_prefix`	Teacher's title. One of the followin • • • • • •
`teacher_number_of_previously_posted_projects`	Number of project applications previously submittec

* 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_4:__ "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 [340]:

```
%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
from tqdm import tqdm_notebook as tqdm1
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 [341]:

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

In [342]:
```
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' 'scho
ol_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 [343]:
```
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[343]:

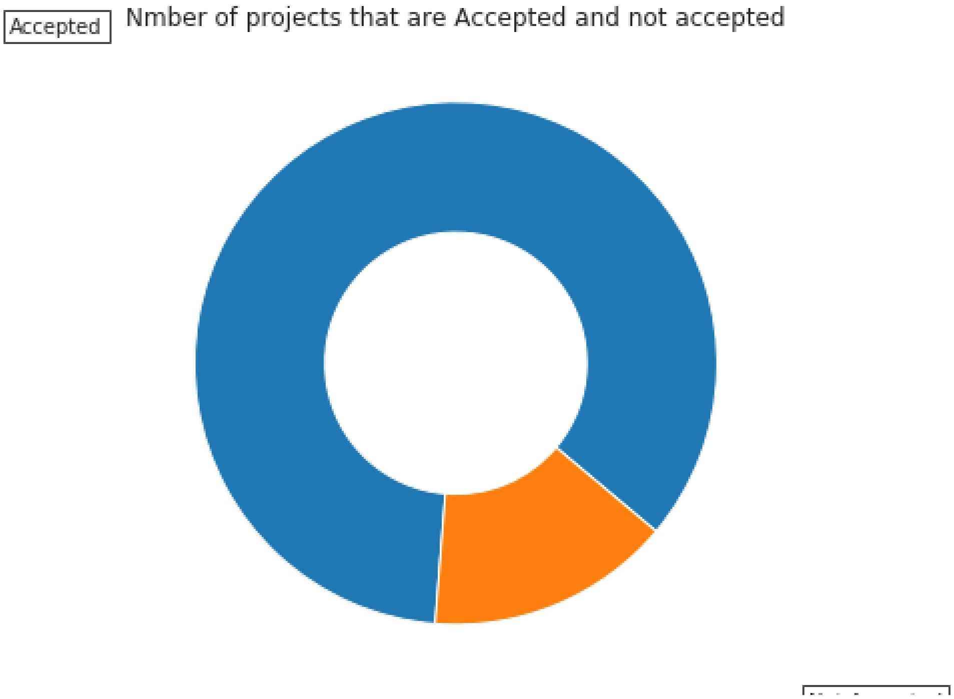
	id	description	quantity	price
0	p233245	LC652 - Lakeshore Double-Space Mobile Drying Rack	1	149.00
1	p069063	Bouncy Bands for Desks (Blue support pipes)	3	14.95

1.2 Data Analysis

In [344]:

```
# PROVIDE CITATIONS TO YOUR CODE IF YOU TAKE IT FROM ANOTHER WEBSITE.
# https://matplotlib.org/gallery/pie_and_polar_charts/pie_and_donut_labels.html#


y_value_counts = project_data['project_is_approved'].value_counts()
print("Number of projects thar are approved for funding ", y_value_counts[1], ",
print("Number of projects thar are not approved for funding ", y_value_counts[0]

fig, ax = plt.subplots(figsize=(6, 6), subplot_kw=dict(aspect="equal"))
recipe = ["Accepted", "Not Accepted"]

data = [y_value_counts[1], y_value_counts[0]]

wedges, texts = ax.pie(data, wedgeprops=dict(width=0.5), startangle=-40)

bbox_props = dict(boxstyle="square,pad=0.3", fc="w", ec="k", lw=0.72)
kw = dict(xycoords='data', textcoords='data', arrowprops=dict(arrowstyle="-"),
          bbox=bbox_props, zorder=0, va="center")

for i, p in enumerate(wedges):
    ang = (p.theta2 - p.theta1)/2. + p.theta1
    y = np.sin(np.deg2rad(ang))
    x = np.cos(np.deg2rad(ang))
    horizontalalignment = {-1: "right", 1: "left"}[int(np.sign(x))]
    connectionstyle = "angle,angleA=0,angleB={}".format(ang)
    kw["arrowprops"].update({"connectionstyle": connectionstyle})
    ax.annotate(recipe[i], xy=(x, y), xytext=(1.35*np.sign(x), 1.4*y),
                 horizontalalignment=horizontalalignment, **kw)

ax.set_title("Nmber of projects that are Accepted and not accepted")

plt.show()
```

```
Number of projects thar are approved for funding  92706 , ( 84.85830404217927
%)
Number of projects thar are not approved for funding  16542 , ( 15.141695957820
739 %)
```



1.2.1 Univariate Analysis: School State

In [346]:

```
# Pandas dataframe groupby count, mean: https://stackoverflow.com/a/19385591/408

temp = pd.DataFrame(project_data.groupby("school_state")["project_is_approved"].
# if you have data which contain only 0 and 1, then the mean = percentage (think
temp.columns = ['state_code', 'num_proposals']

'''# How to plot US state heatmap: https://datascience.stackexchange.com/a/9620

scl = [[0.0, 'rgb(242,240,247)'],[0.2, 'rgb(218,218,235)'],[0.4, 'rgb(188,189,22
            [0.6, 'rgb(158,154,200)'],[0.8, 'rgb(117,107,177)'],[1.0, 'rgb(84,39

data = [ dict(
        type='choropleth',
        colorscale = scl,
        autocolorscale = False,
        locations = temp['state_code'],
        z = temp['num_proposals'].astype(float),
        locationmode = 'USA-states',
        text = temp['state_code'],
        marker = dict(line = dict (color = 'rgb(255,255,255)',width = 2)),
        colorbar = dict(title = "% of pro")
    ) ]

layout = dict(
        title = 'Project Proposals % of Acceptance Rate by US States',
        geo = dict(
            scope='usa',
            projection=dict( type='albers usa' ),
            showlakes = True,
            lakecolor = 'rgb(255, 255, 255)',
        ),
    )

fig = go.Figure(data=data, layout=layout)
offline.iplot(fig, filename='us-map-heat-map')
'''
```

Out[346]:

```
'# How to plot US state heatmap: https://datascience.stackexchange.com/a/9620\n
\nscl (https://datascience.stackexchange.com/a/9620\n\nscl) = [[0.0, \'rgb(242,
240,247)\'],[0.2, \'rgb(218,218,235)\'],[0.4, \'rgb(188,189,220)\'],
[0.6, \'rgb(158,154,200)\'],[0.8, \'rgb(117,107,177)\'],[1.0, \'rgb(84,39,143)
\']]\n\ndata = [ dict(\n        type=\'choropleth\',\n        colorscale = sc
l,\n        autocolorscale = False,\n        locations = temp[\'state_code
\'],\n        z = temp[\'num_proposals\'].astype(float),\n        locationmode
 = \'USA-states\',\n        text = temp[\'state_code\'],\n        marker = dict
(line = dict (color = \'rgb(255,255,255)\',width = 2)),\n        colorbar = dic
t(title = "% of pro")\n    ) ]\n\nlayout = dict(\n        title = \'Project Pro
posals % of Acceptance Rate by US States\',\n        geo = dict(\n            s
cope=\'usa\',\n            projection=dict( type=\'albers usa\' ),\n
showlakes = True,\n            lakecolor = \'rgb(255, 255, 255)\',\n
 ),\n    )\n\nfig = go.Figure(data=data, layout=layout)\noffline.iplot(fig, fil
ename=\'us-map-heat-map\')\n'
```

In [347]:

```
# https://www.csi.cuny.edu/sites/default/files/pdf/administration/ops/2letterstal
temp.sort_values(by=['num_proposals'], inplace=True)
print("States with lowest % approvals")
print(temp.head(5))
print('='*50)
print("States with highest % approvals")
print(temp.tail(5))
```

```
States with lowest % approvals
   state_code  num_proposals
46         VT       0.800000
7          DC       0.802326
43         TX       0.813142
26         MT       0.816327
18         LA       0.831245
==================================================
States with highest % approvals
   state_code  num_proposals
30         NH       0.873563
35         OH       0.875152
47         WA       0.876178
28         ND       0.888112
8          DE       0.897959
```

In [348]:

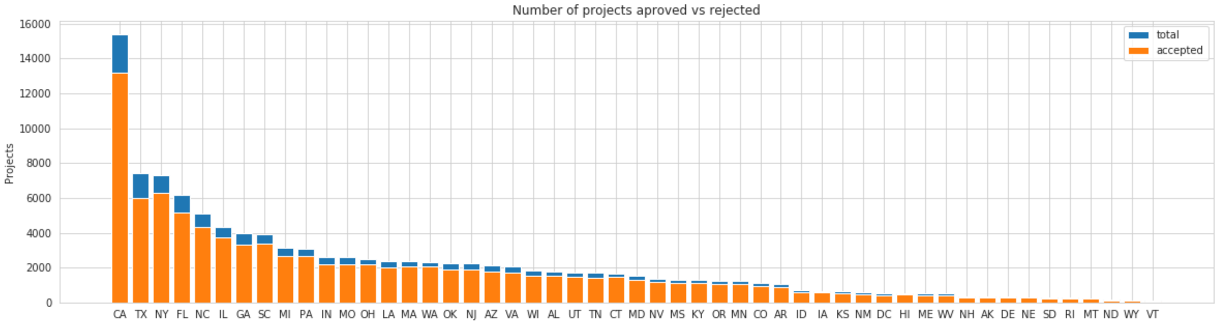
```
#stacked bar plots matplotlib: https://matplotlib.org/gallery/lines_bars_and_mark
def stack_plot(data, xtick, col2='project_is_approved', col3='total'):
    ind = np.arange(data.shape[0])

    plt.figure(figsize=(20,5))
    p1 = plt.bar(ind, data[col3].values)
    p2 = plt.bar(ind, data[col2].values)

    plt.ylabel('Projects')
    plt.title('Number of projects aproved vs rejected')
    plt.xticks(ind, list(data[xtick].values))
    plt.legend((p1[0], p2[0]), ('total', 'accepted'))
    plt.show()
```

In [349]:

```
def univariate_barplots(data, col1, col2='project_is_approved', top=False):
    # Count number of zeros in dataframe python: https://stackoverflow.com/a/5154
    temp = pd.DataFrame(project_data.groupby(col1)[col2].agg(lambda x: x.eq(1).su

    # Pandas dataframe grouby count: https://stackoverflow.com/a/19385591/4084039
    temp['total'] = pd.DataFrame(project_data.groupby(col1)[col2].agg({'total':'
    temp['Avg'] = pd.DataFrame(project_data.groupby(col1)[col2].agg({'Avg':'mean

    temp.sort_values(by=['total'],inplace=True, ascending=False)

    if top:
        temp = temp[0:top]

    stack_plot(temp, xtick=col1, col2=col2, col3='total')
    print(temp.head(5))
    print("="*50)
    print(temp.tail(5))
```

In [350]:

```
univariate_barplots(project_data, 'school_state', 'project_is_approved', False)
```

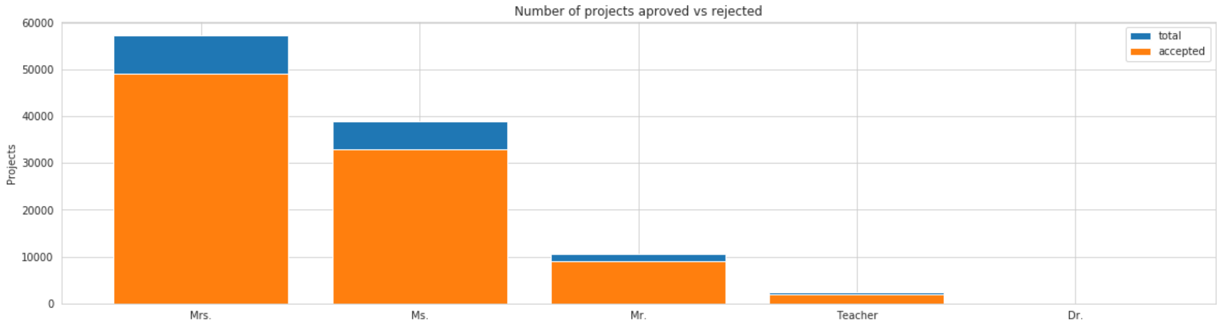


```
   school_state  project_is_approved  total       Avg
4            CA                13205  15388  0.858136
43           TX                 6014   7396  0.813142
34           NY                 6291   7318  0.859661
9            FL                 5144   6185  0.831690
27           NC                 4353   5091  0.855038
==================================================
   school_state  project_is_approved  total       Avg
39           RI                  243    285  0.852632
26           MT                  200    245  0.816327
28           ND                  127    143  0.888112
50           WY                   82     98  0.836735
46           VT                   64     80  0.800000
```

SUMMARY: Every state has greater than 80% success rate in approval

1.2.2 Univariate Analysis: teacher_prefix

In [351]:
```
univariate_barplots(project_data, 'teacher_prefix', 'project_is_approved' , top=
```

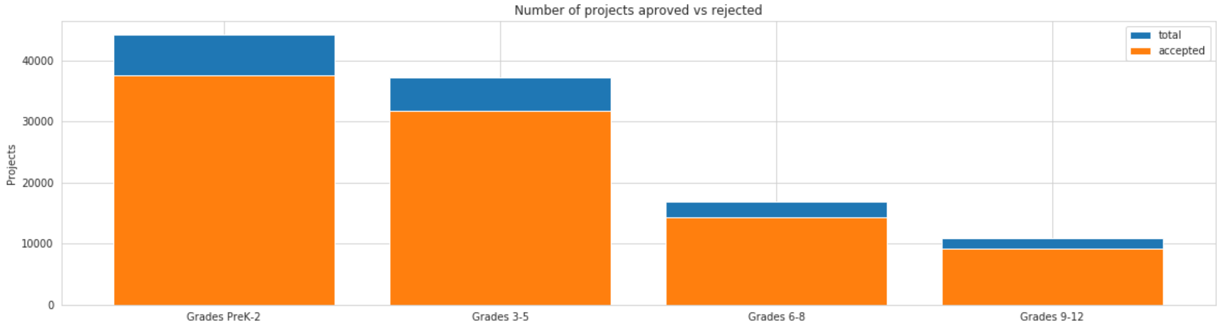


```
  teacher_prefix  project_is_approved  total       Avg
2           Mrs.                48997  57269  0.855559
3            Ms.                32860  38955  0.843537
1            Mr.                 8960  10648  0.841473
4        Teacher                 1877   2360  0.795339
0            Dr.                    9     13  0.692308
==================================================
```

```
  teacher_prefix  project_is_approved  total       Avg
2           Mrs.                48997  57269  0.855559
3            Ms.                32860  38955  0.843537
1            Mr.                 8960  10648  0.841473
4        Teacher                 1877   2360  0.795339
0            Dr.                    9     13  0.692308
```

1.2.3 Univariate Analysis: project_grade_category

In [352]:
```
univariate_barplots(project_data, 'project_grade_category', 'project_is_approved
```



```
  project_grade_category  project_is_approved  total       Avg
3          Grades PreK-2                37536  44225  0.848751
0             Grades 3-5                31729  37137  0.854377
1             Grades 6-8                14258  16923  0.842522
2            Grades 9-12                 9183  10963  0.837636
==================================================
```

```
  project_grade_category  project_is_approved  total       Avg
3          Grades PreK-2                37536  44225  0.848751
0             Grades 3-5                31729  37137  0.854377
1             Grades 6-8                14258  16923  0.842522
2            Grades 9-12                 9183  10963  0.837636
```

1.2.4 Univariate Analysis: project_subject_categories

In [353]:
```
catogories = list(project_data['project_subject_categories'].values)
# remove special characters from list of strings python: https://stackoverflow.co

# https://www.geeksforgeeks.org/removing-stop-words-nltk-python/
# https://stackoverflow.com/questions/23669024/how-to-strip-a-specific-word-from
# https://stackoverflow.com/questions/8270092/remove-all-whitespace-in-a-string-
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",
        if 'The' in j.split(): # this will split each of the catogory based on sp
            j=j.replace('The','') # if we have the words "The" we are going to re
        j = j.replace(' ','') # we are placeing all the ' '(space) with ''(empty
        temp+=j.strip()+" " #" abc ".strip() will return "abc", remove the trail
        temp = temp.replace('&','_') # we are replacing the & value into
    cat_list.append(temp.strip())
```

In [354]:
```
print(catogories[3])
cat_list[3]
```

```
Literacy & Language, Math & Science
```

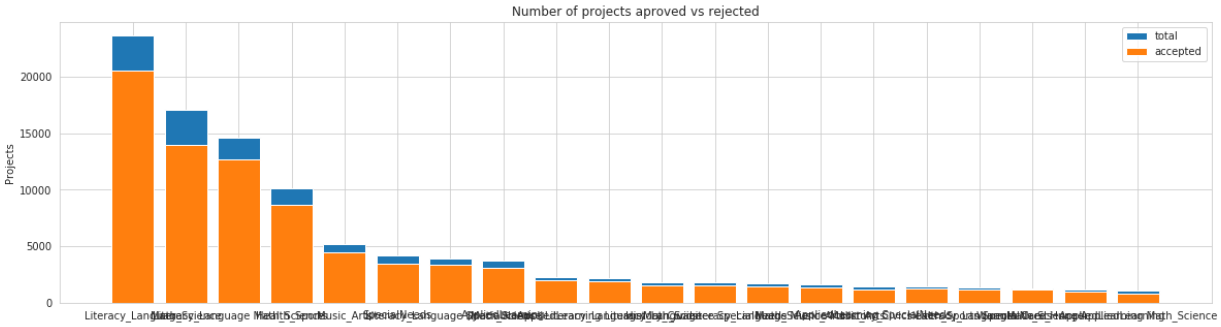
Out[354]:
```
'Literacy_Language Math_Science'
```

In [355]:
```
project_data['clean_categories'] = cat_list
project_data.drop(['project_subject_categories'], axis=1, inplace=True)
project_data.head(5)
```

Out[355]:

	Unnamed: 0	id	teacher_id	teacher_prefix	school_state	project_
0	160221	p253737	c90749f5d961ff158d4b4d1e7dc665fc	Mrs.	IN	
1	140945	p258326	897464ce9ddc600bced1151f324dd63a	Mr.	FL	
2	21895	p182444	3465aaf82da834c0582ebd0ef8040ca0	Ms.	AZ	
3	45	p246581	f3cb9bffbba169bef1a77b243e620b60	Mrs.	KY	

In [356]:
```
univariate_barplots(project_data, 'clean_categories', 'project_is_approved', top
```



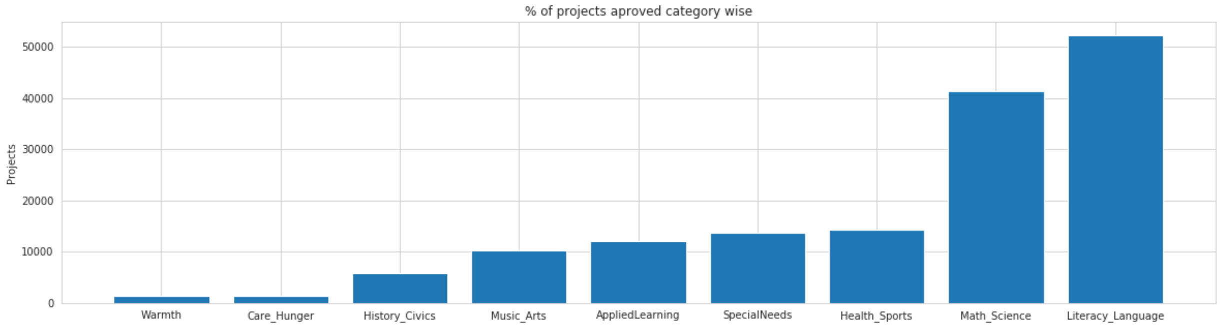
```
                    clean_categories    ...          Avg
24                 Literacy_Language    ...     0.867470
32                      Math_Science    ...     0.819529
28  Literacy_Language Math_Science    ...     0.869432
8                      Health_Sports    ...     0.848973
40                        Music_Arts    ...     0.855019

[5 rows x 4 columns]
==================================================
                    clean_categories    ...          Avg
19  History_Civics Literacy_Language    ...     0.894441
14        Health_Sports SpecialNeeds    ...     0.873472
50                Warmth Care_Hunger    ...     0.925898
33      Math_Science AppliedLearning    ...     0.835246
4       AppliedLearning Math_Science    ...     0.812738

[5 rows x 4 columns]
```

In [357]:
```

# count of all the words in corpus python: https://stackoverflow.com/a/22898595/
from collections import Counter
my_counter = Counter()
for word in project_data['clean_categories'].values:
    my_counter.update(word.split())
my_counter
```

Out[357]:
```
Counter({'Literacy_Language': 52239,
         'History_Civics': 5914,
         'Health_Sports': 14223,
         'Math_Science': 41421,
         'SpecialNeeds': 13642,
         'AppliedLearning': 12135,
         'Music_Arts': 10293,
         'Warmth': 1388,
         'Care_Hunger': 1388})
```

In [358]:

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


ind = np.arange(len(sorted_cat_dict))
plt.figure(figsize=(20,5))
p1 = plt.bar(ind, list(sorted_cat_dict.values()))

plt.ylabel('Projects')
plt.title('% of projects aproved category wise')
plt.xticks(ind, list(sorted_cat_dict.keys()))
plt.show()
print(sorted_cat_dict)
```



```
{'Warmth': 1388, 'Care_Hunger': 1388, 'History_Civics': 5914, 'Music_Arts': 102
93, 'AppliedLearning': 12135, 'SpecialNeeds': 13642, 'Health_Sports': 14223, 'M
ath_Science': 41421, 'Literacy_Language': 52239}
```

In [359]:

```
for i, j in sorted_cat_dict.items():
    print("{:20} :{:10}".format(i,j))
```

```
Warmth               :      1388
Care_Hunger          :      1388
History_Civics       :      5914
Music_Arts           :     10293
AppliedLearning      :     12135
SpecialNeeds         :     13642
Health_Sports        :     14223
Math_Science         :     41421
Literacy_Language    :     52239
```

1.2.5 Univariate Analysis: project_subject_subcategories

In [360]:
```
sub_catogories = list(project_data['project_subject_subcategories'].values)
# remove special characters from list of strings python: https://stackoverflow.co

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

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",
        if 'The' in j.split(): # this will split each of the catogory based on sp
            j=j.replace('The','') # if we have the words "The" we are going to re
        j = j.replace(' ','') # we are placeing all the ' '(space) with ''(empty
        temp +=j.strip()+" "#" abc ".strip() will return "abc", remove the trail
        temp = temp.replace('&','_')
    sub_cat_list.append(temp.strip())
```

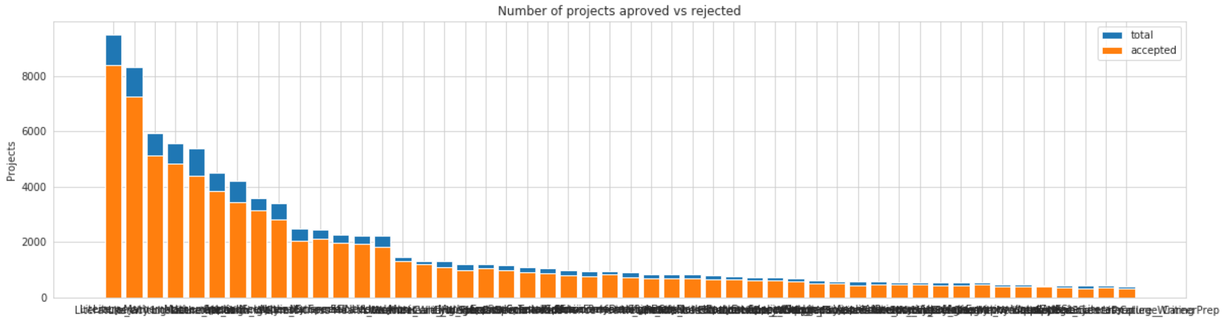
In [361]:
```
project_data['clean_subcategories'] = sub_cat_list
project_data.drop(['project_subject_subcategories'], axis=1, inplace=True)
project_data.head(2)
```

Out[361]:

	Unnamed: 0	id	teacher_id	teacher_prefix	school_state	project_sul
0	160221	p253737	c90749f5d961ff158d4b4d1e7dc665fc	Mrs.	IN	20
1	140945	p258326	897464ce9ddc600bced1151f324dd63a	Mr.	FL	20

In [362]:

```
univariate_barplots(project_data, 'clean_subcategories', 'project_is_approved',
```

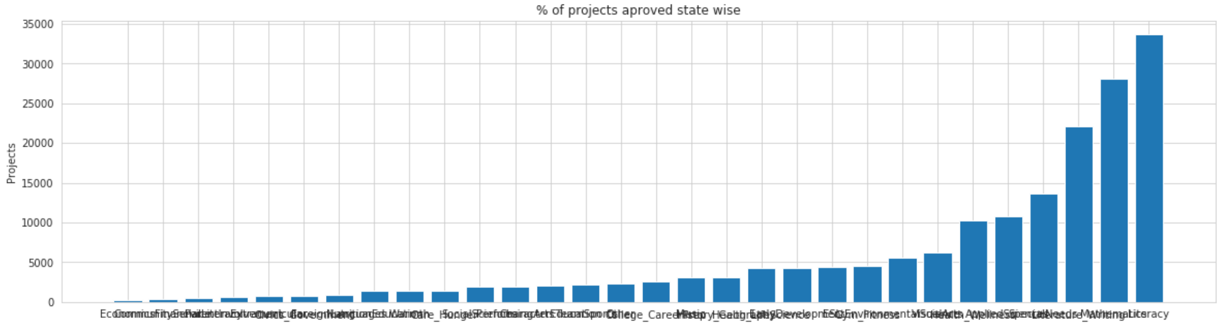


```
                   clean_subcategories    ...         Avg
317                           Literacy    ...    0.882458
319               Literacy Mathematics    ...    0.872072
331   Literature_Writing Mathematics    ...    0.867803
318      Literacy Literature_Writing    ...    0.865733
342                        Mathematics    ...    0.815207

[5 rows x 4 columns]
==================================================
                       clean_subcategories    ...         Avg
196        EnvironmentalScience Literacy    ...    0.876126
127                                   ESL    ...    0.828979
79                      College_CareerPrep    ...    0.814727
17    AppliedSciences Literature_Writing    ...    0.859524
3     AppliedSciences College_CareerPrep    ...    0.814815

[5 rows x 4 columns]
```

In [363]:

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

In [364]:
```
# dict sort by value python: https://stackoverflow.com/a/613218/4084039
sub_cat_dict = dict(my_counter)
sorted_sub_cat_dict = dict(sorted(sub_cat_dict.items(), key=lambda kv: kv[1]))


ind = np.arange(len(sorted_sub_cat_dict))
plt.figure(figsize=(20,5))
p1 = plt.bar(ind, list(sorted_sub_cat_dict.values()))

plt.ylabel('Projects')
plt.title('% of projects aproved state wise')
plt.xticks(ind, list(sorted_sub_cat_dict.keys()))
plt.show()
```



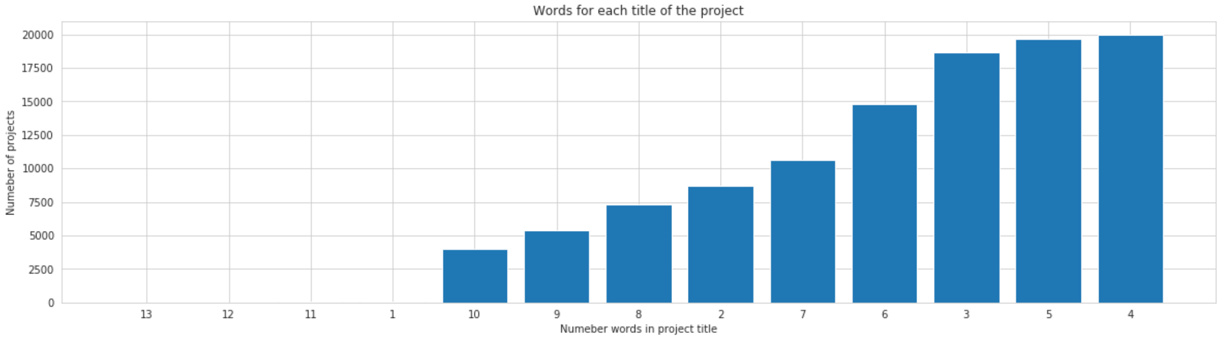
In [365]:
```
for i, j in sorted_sub_cat_dict.items():
    print("{:20} :{:10}".format(i,j))
```

```
Economics            :       269
CommunityService     :       441
FinancialLiteracy    :       568
ParentInvolvement    :       677
Extracurricular      :       810
Civics_Government    :       815
ForeignLanguages     :       890
NutritionEducation   :      1355
Warmth               :      1388
Care_Hunger          :      1388
SocialSciences       :      1920
PerformingArts       :      1961
CharacterEducation   :      2065
TeamSports           :      2192
Other                :      2372
College_CareerPrep   :      2568
Music                :      3145
History_Geography    :      3171
Health_LifeScience   :      4235
EarlyDevelopment     :      4254
ESL                  :      4367
Gym_Fitness          :      4509
EnvironmentalScience :      5591
VisualArts           :      6278
Health_Wellness      :     10234
AppliedSciences      :     10816
SpecialNeeds         :     13642
Literature_Writing   :     22179
Mathematics          :     28074
Literacy             :     33700
```

1.2.6 Univariate Analysis: Text features (Title)

In [366]:

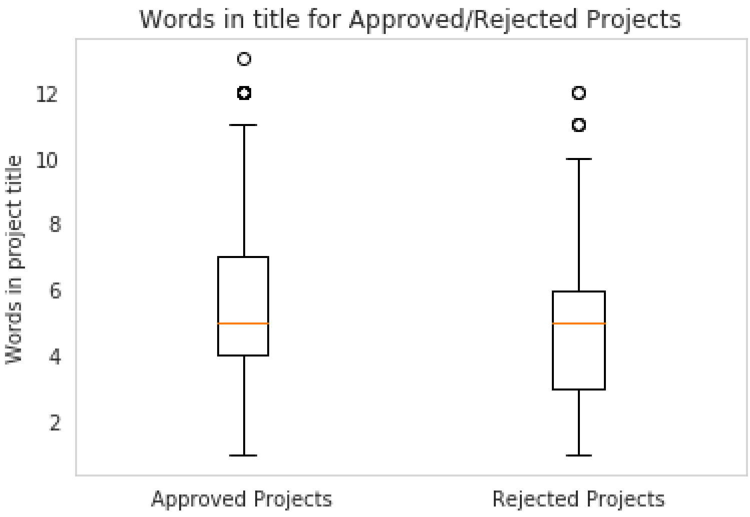
```
#How to calculate number of words in a string in DataFrame: https://stackoverflo
word_count = project_data['project_title'].str.split().apply(len).value_counts()
word_dict = dict(word_count)
word_dict = dict(sorted(word_dict.items(), key=lambda kv: kv[1]))


ind = np.arange(len(word_dict))
plt.figure(figsize=(20,5))
p1 = plt.bar(ind, list(word_dict.values()))

plt.ylabel('Numeber of projects')
plt.xlabel('Numeber words in project title')
plt.title('Words for each title of the project')
plt.xticks(ind, list(word_dict.keys()))
plt.show()
# w = project_data['project_title'][:2].str.split().apply(len).value_counts()
# w
```



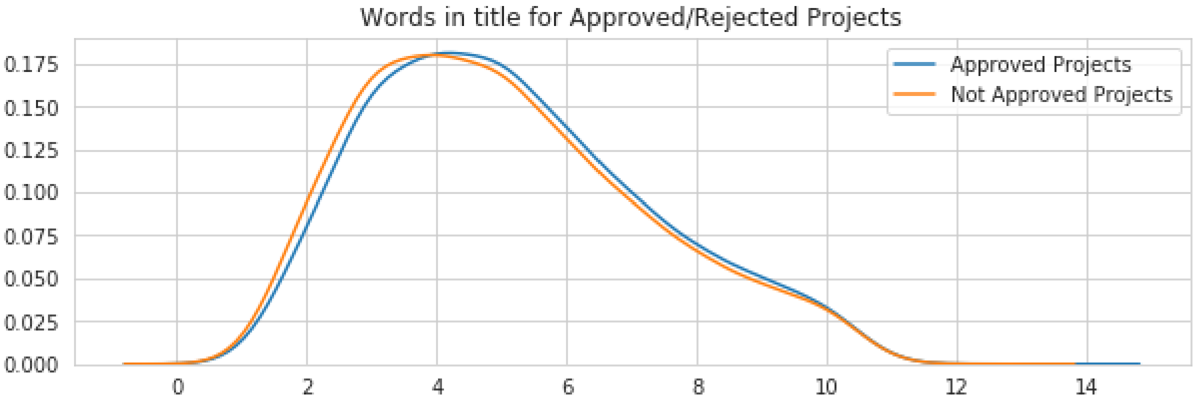
In [367]:

```
approved_title_word_count = project_data[project_data['project_is_approved']==1]
# print(approved_title_word_count[:5])
approved_title_word_count = approved_title_word_count.values
# print(approved_title_word_count[:5])

rejected_title_word_count = project_data[project_data['project_is_approved']==0]
rejected_title_word_count = rejected_title_word_count.values
```

In [368]:
```
# https://glowingpython.blogspot.com/2012/09/boxplot-with-matplotlib.html
plt.boxplot([approved_title_word_count, rejected_title_word_count])
plt.xticks([1,2],('Approved Projects','Rejected Projects'))
plt.ylabel('Words in project title')
plt.title('Words in title for Approved/Rejected Projects')
plt.grid()
plt.show()
```



In [450]:
```
plt.figure(figsize=(10,3))
sns.kdeplot(approved_title_word_count,label="Approved Projects", bw=0.6)
sns.kdeplot(rejected_title_word_count,label="Not Approved Projects", bw=0.6)
plt.title('Words in title for Approved/Rejected Projects')
plt.legend()
plt.show()
```

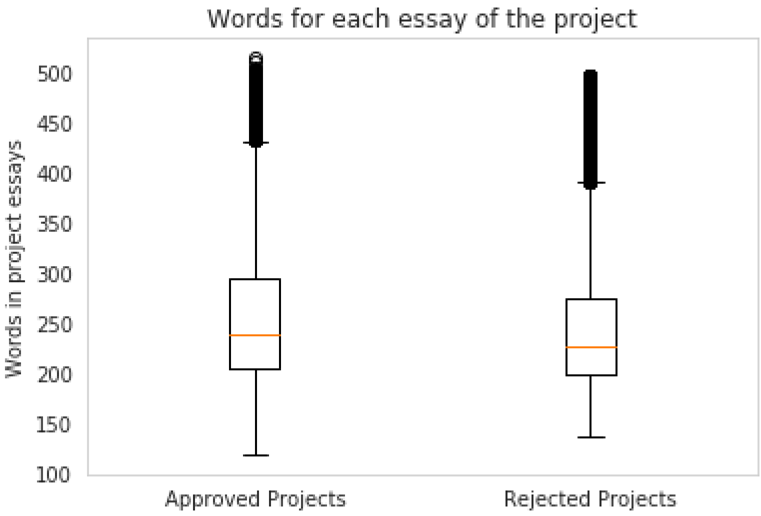


1.2.7 Univariate Analysis: Text features (Project Essay's)

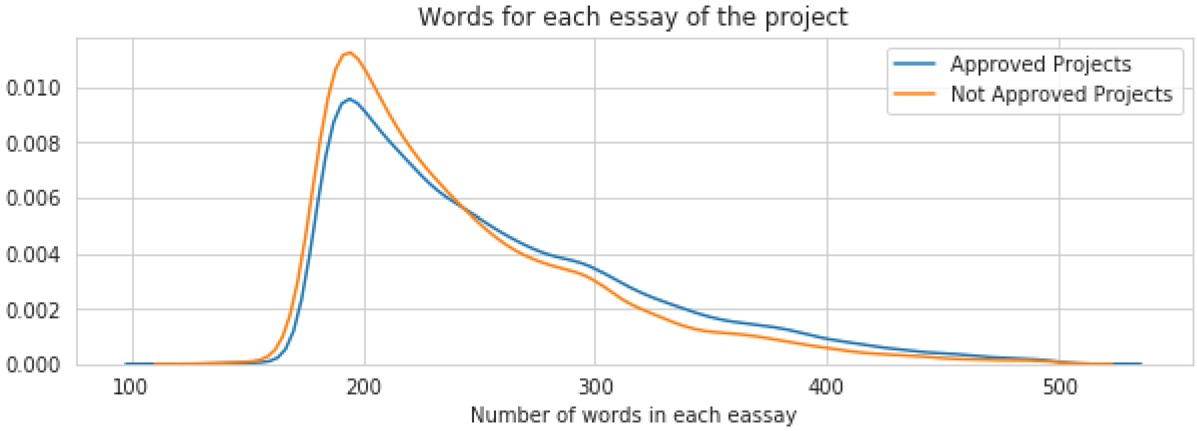
In [370]:
```
# merge two column text dataframe:
project_data["essay"] = project_data["project_essay_1"].map(str) +\
                        project_data["project_essay_2"].map(str) + \
                        project_data["project_essay_3"].map(str) + \
                        project_data["project_essay_4"].map(str)
```

In [371]:
```
approved_word_count = project_data[project_data['project_is_approved']==1]['essa
approved_word_count = approved_word_count.values

rejected_word_count = project_data[project_data['project_is_approved']==0]['essa
rejected_word_count = rejected_word_count.values
```

In [372]:
```
# https://glowingpython.blogspot.com/2012/09/boxplot-with-matplotlib.html
plt.boxplot([approved_word_count, rejected_word_count])
plt.title('Words for each essay of the project')
plt.xticks([1,2],('Approved Projects','Rejected Projects'))
plt.ylabel('Words in project essays')
plt.grid()
plt.show()
```



In [373]:
```
plt.figure(figsize=(10,3))
sns.distplot(approved_word_count, hist=False, label="Approved Projects")
sns.distplot(rejected_word_count, hist=False, label="Not Approved Projects")
plt.title('Words for each essay of the project')
plt.xlabel('Number of words in each eassay')
plt.legend()
plt.show()
```



1.2.8 Univariate Analysis: Cost per project

In [374]:

```
# we get the cost of the project using resource.csv file
resource_data.head(2)
```

Out[374]:

	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 [375]:

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

Out[375]:

	id	price	quantity
0	p000001	459.56	7
1	p000002	515.89	21

In [376]:

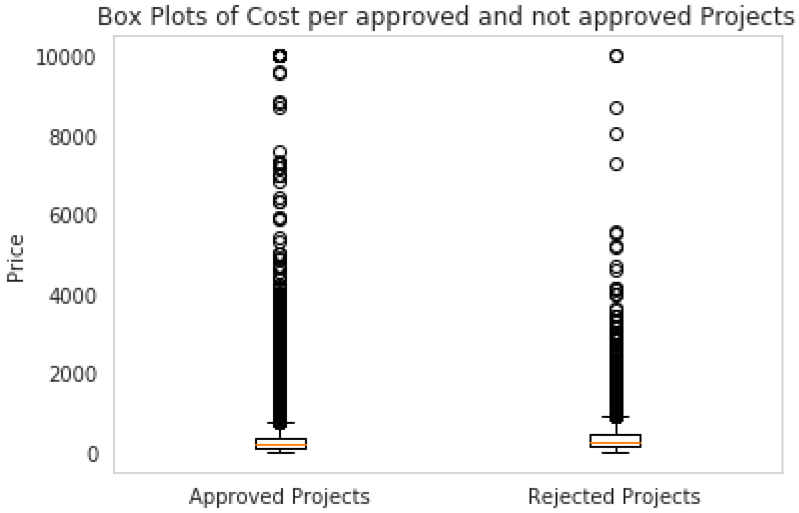
```
# join two dataframes in python:
project_data = pd.merge(project_data, price_data, on='id', how='left')
```

In [377]:

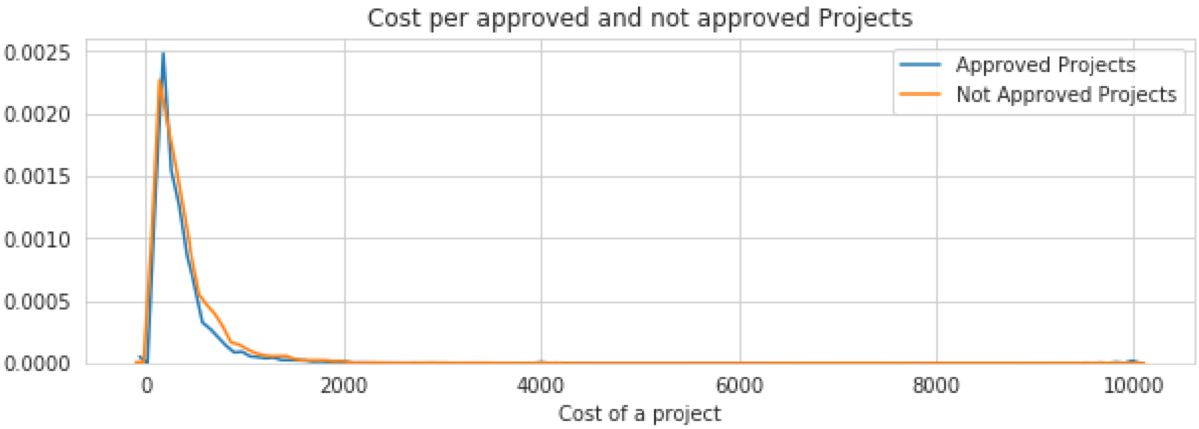
```
approved_price = project_data[project_data['project_is_approved']==1]['price'].va

rejected_price = project_data[project_data['project_is_approved']==0]['price'].va
```

In [378]:

```
# https://glowingpython.blogspot.com/2012/09/boxplot-with-matplotlib.html
plt.boxplot([approved_price, rejected_price])
plt.title('Box Plots of Cost per approved and not approved Projects')
plt.xticks([1,2],('Approved Projects','Rejected Projects'))
plt.ylabel('Price')
plt.grid()
plt.show()
```



In [379]:
```
plt.figure(figsize=(10,3))
sns.distplot(approved_price, hist=False, label="Approved Projects")
sns.distplot(rejected_price, hist=False, label="Not Approved Projects")
plt.title('Cost per approved and not approved Projects')
plt.xlabel('Cost of a project')
plt.legend()
plt.show()
```



In [380]:

```
# http://zetcode.com/python/prettytable/
from prettytable import PrettyTable

#If you get a ModuleNotFoundError error , install prettytable using: pip3 install

x = PrettyTable()
x.field_names = ["Percentile", "Approved Projects", "Not Approved Projects"]

for i in range(0,101,5):
    x.add_row([i,np.round(np.percentile(approved_price,i), 3), np.round(np.perce
print(x)
```

```
+------------+-------------------+-----------------------+
| Percentile | Approved Projects | Not Approved Projects |
+------------+-------------------+-----------------------+
|     0      |        0.66       |          1.97         |
|     5      |       13.59       |          41.9         |
|     10     |       33.88       |         73.67         |
|     15     |        58.0       |         99.109        |
|     20     |       77.38       |         118.56        |
|     25     |       99.95       |        140.892        |
|     30     |       116.68      |         162.23        |
|     35     |      137.232      |        184.014        |
|     40     |       157.0       |        208.632        |
|     45     |      178.265      |        235.106        |
|     50     |       198.99      |        263.145        |
|     55     |       223.99      |         292.61        |
|     60     |       255.63      |        325.144        |
|     65     |      285.412      |         362.39        |
|     70     |      321.225      |         399.99        |
|     75     |      366.075      |        449.945        |
|     80     |       411.67      |        519.282        |
|     85     |       479.0       |        618.276        |
|     90     |       593.11      |        739.356        |
|     95     |      801.598      |        992.486        |
|    100     |       9999.0      |         9999.0        |
+------------+-------------------+-----------------------+
```

1.2.9 Univariate Analysis: teacher_number_of_previously_posted_projects

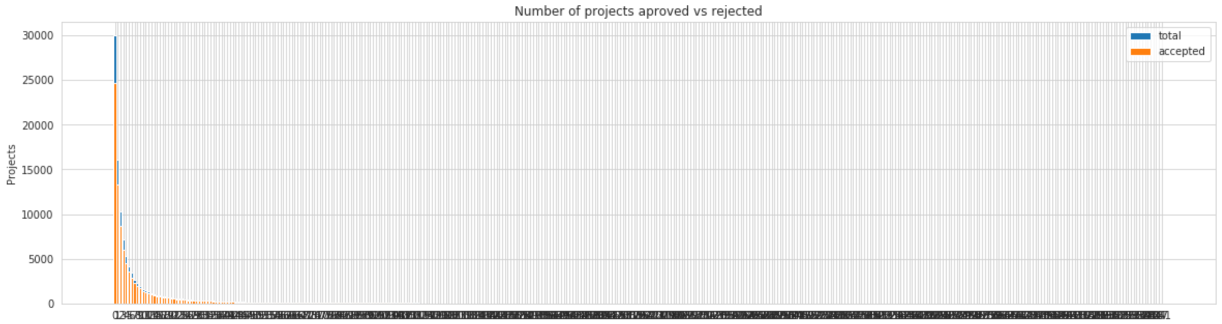
Please do this on your own based on the data analysis that was done in the above cells

In [381]:
```
project_data.head(5)
```

Out[381]:

	Unnamed: 0	id	teacher_id	teacher_prefix	school_state	project_sul
0	160221	p253737	c90749f5d961ff158d4b4d1e7dc665fc	Mrs.	IN	20
1	140945	p258326	897464ce9ddc600bced1151f324dd63a	Mr.	FL	20
2	21895	p182444	3465aaf82da834c0582ebd0ef8040ca0	Ms.	AZ	20
3	45	p246581	f3cb9bffbba169bef1a77b243e620b60	Mrs.	KY	20
4	172407	p104768	be1f7507a41f8479dc06f047086a39ec	Mrs.	TX	20

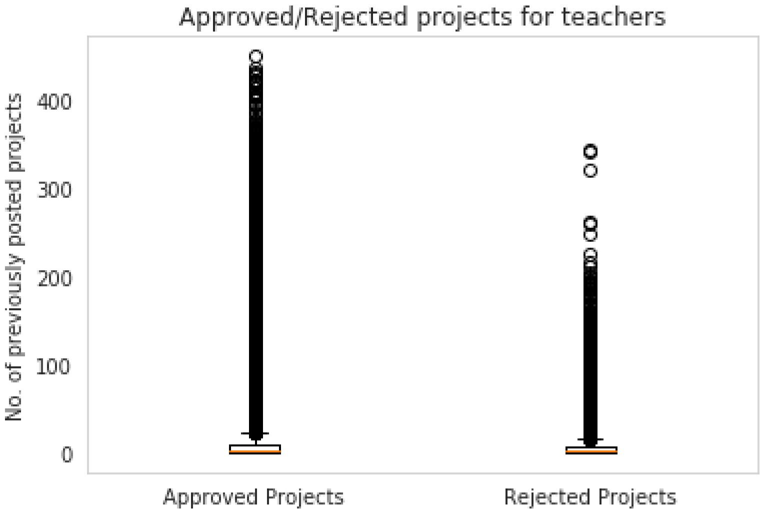
```
In [382]: univariate_barplots(project_data, 'teacher_number_of_previously_posted_projects'
```



```
   teacher_number_of_previously_posted_projects    ...         Avg
0                                              0    ...    0.821350
1                                              1    ...    0.830054
2                                              2    ...    0.841063
3                                              3    ...    0.843460
4                                              4    ...    0.845423

[5 rows x 4 columns]
==================================================
     teacher_number_of_previously_posted_projects ...   Avg
242                                           242 ...   1.0
268                                           270 ...   1.0
234                                           234 ...   1.0
335                                           347 ...   1.0
373                                           451 ...   1.0

[5 rows x 4 columns]
```

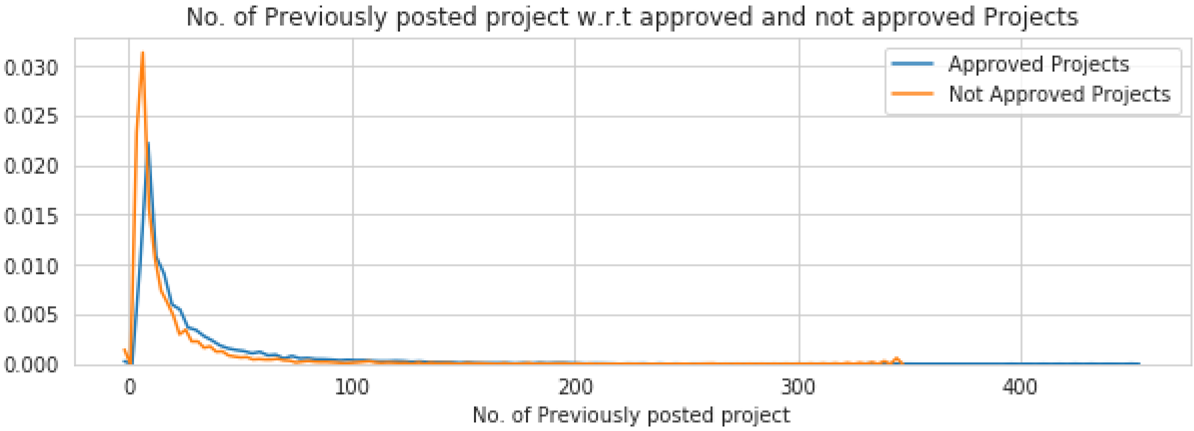
Observation: From Barplot, we can easily mis-interpret the data saying that if a teacher didn't post (or) posted fewer projects previously tend to have more approval chances than teachers who posted more previously but if we carefully see the numerical data, we observe that ,teachers who posted more projects previously, tend to post fewer projects but have high acceptance chance (and) teachers who posted fewer projects previously, tend to post more projects but have low acceptance chance.

In [383]:
```
approved_teacher_number_of_previously_posted_projects = project_data[project_dat
rejected_teacher_number_of_previously_posted_projects = project_data[project_dat

plt.boxplot([approved_teacher_number_of_previously_posted_projects, rejected_tea
plt.title('Approved/Rejected projects for teachers')
plt.xticks([1,2],('Approved Projects','Rejected Projects'))
plt.ylabel('No. of previously posted projects')
plt.grid()
plt.show()
```



Observation: We cannot infer much from Box plot,so let's try PDF

In [384]:
```
plt.figure(figsize=(10,3))
sns.distplot(approved_teacher_number_of_previously_posted_projects, hist=False, 
sns.distplot(rejected_teacher_number_of_previously_posted_projects, hist=False, 
plt.title('No. of Previously posted project w.r.t approved and not approved Proje
plt.xlabel('No. of Previously posted project')
plt.legend()
plt.show()
```



Observation:if a teacher didn't post (or) posted fewer projects previously tend to have more approval chances than teachers who posted more previously which is a wrong

interpretation.

In [385]:

```
from prettytable import PrettyTable

#If you get a ModuleNotFoundError error , install prettytable using: pip3 instal

x = PrettyTable()
x.field_names = ["Percentile", "Approved Projects", "Not Approved Projects"]

for i in range(0,101,5):
    x.add_row([i,np.round(np.percentile(approved_teacher_number_of_previously_pos
print(x)
```

```
+------------+-------------------+-----------------------+
| Percentile | Approved Projects | Not Approved Projects |
+------------+-------------------+-----------------------+
|     0      |        0.0        |          0.0          |
|     5      |        0.0        |          0.0          |
|     10     |        0.0        |          0.0          |
|     15     |        0.0        |          0.0          |
|     20     |        0.0        |          0.0          |
|     25     |        0.0        |          0.0          |
|     30     |        1.0        |          0.0          |
|     35     |        1.0        |          1.0          |
|     40     |        1.0        |          1.0          |
|     45     |        2.0        |          1.0          |
|     50     |        2.0        |          2.0          |
|     55     |        3.0        |          2.0          |
|     60     |        4.0        |          3.0          |
|     65     |        5.0        |          3.0          |
|     70     |        7.0        |          4.0          |
|     75     |        9.0        |          6.0          |
|     80     |        13.0       |          8.0          |
|     85     |        19.0       |          11.0         |
|     90     |        30.0       |          17.0         |
|     95     |        57.0       |          31.0         |
|    100     |       451.0       |         345.0         |
+------------+-------------------+-----------------------+
```

Observation: From the above pecentiles table, we can say that as the no. of previously posted projects increases, those teachers have more Approved chances.

1.2.10 Univariate Analysis: project_resource_summary

Please do this on your own based on the data analysis that was done in the above cells

Check if the `presence of the numerical digits` in the `project_resource_summary` effects the acceptance of the project or not. If you observe that `presence of the numerical digits` is helpful in the classification, please include it for further process or you can ignore it.

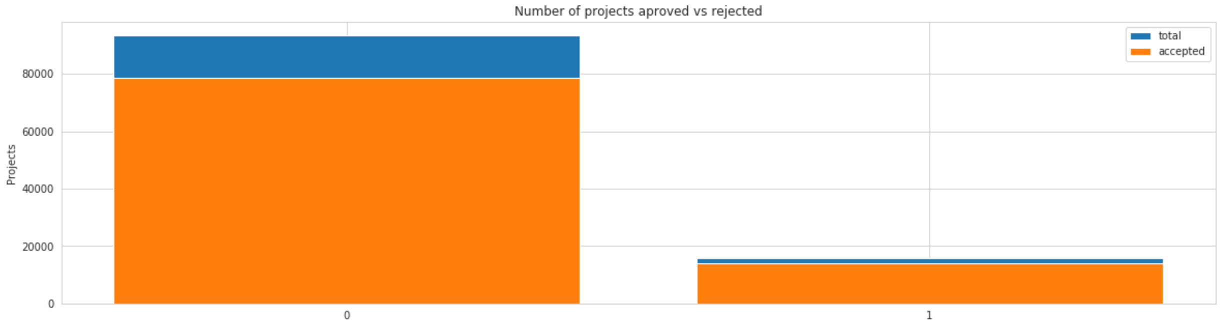
In [386]:

```
#presence of the numerical digits in a strings with numeric : https://stackoverf
def hasNumbers(inputString):
    return any(i.isdigit() for i in inputString)
p1 = project_data[['id','project_resource_summary']]
p1 = pd.DataFrame(data=p1)
p1.columns = ['id','digits_in_summary']
p1['digits_in_summary'] =  p1['digits_in_summary'].map(hasNumbers)
# https://stackoverflow.com/a/17383325/8089731
p1['digits_in_summary'] = p1['digits_in_summary'].astype(int)
project_data = pd.merge(project_data, p1, on='id', how='left')
project_data.head(5)
```

Out[386]:

	Unnamed: 0	id	teacher_id	teacher_prefix	school_state	project_sul
0	160221	p253737	c90749f5d961ff158d4b4d1e7dc665fc	Mrs.	IN	20
1	140945	p258326	897464ce9ddc600bced1151f324dd63a	Mr.	FL	20
2	21895	p182444	3465aaf82da834c0582ebd0ef8040ca0	Ms.	AZ	20
3	45	p246581	f3cb9bffbba169bef1a77b243e620b60	Mrs.	KY	20
4	172407	p104768	be1f7507a41f8479dc06f047086a39ec	Mrs.	TX	20

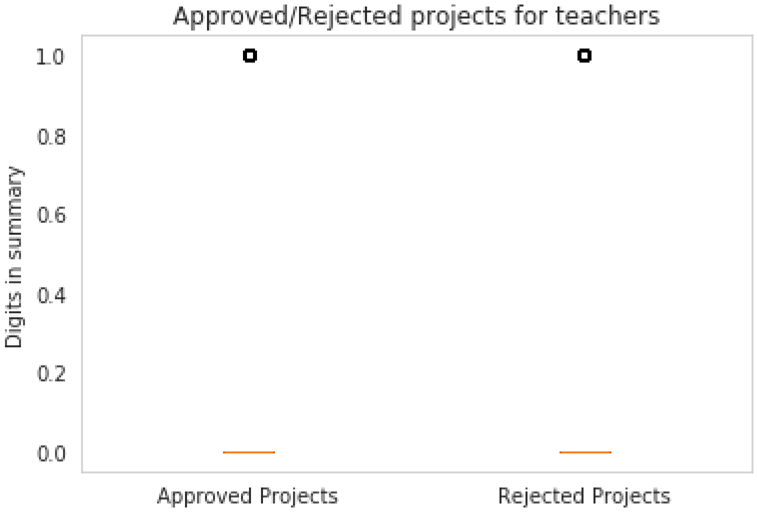
```
In [387]: univariate_barplots(project_data, 'digits_in_summary', 'project_is_approved', Fa
```



```
   digits_in_summary  project_is_approved  total       Avg
0                  0                78616  93492  0.840885
1                  1                14090  15756  0.894263
==================================================
   digits_in_summary  project_is_approved  total       Avg
0                  0                78616  93492  0.840885
1                  1                14090  15756  0.894263
```

Observation: From the numerical data we can say that, if project's summary have digits included, then they have high acceptance rate

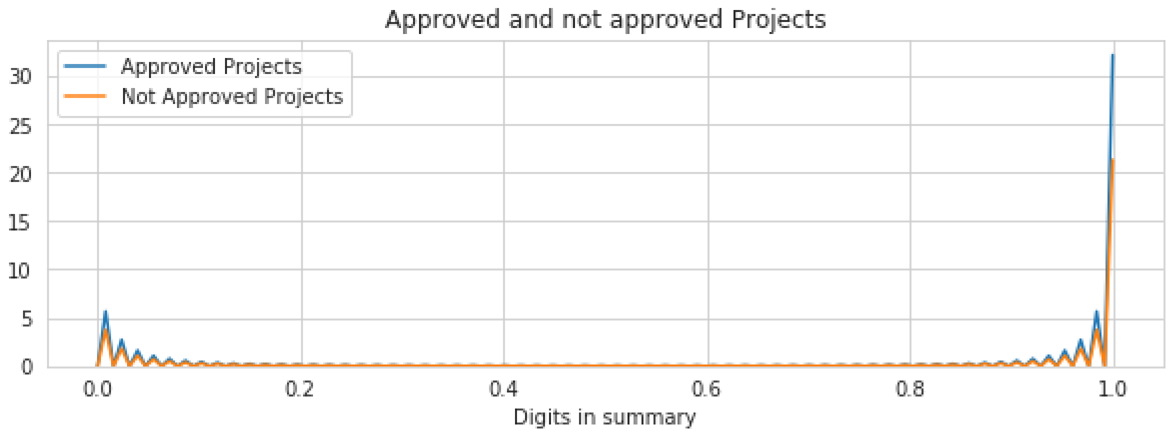
```
In [388]: approved_digits_in_summary = project_data[project_data['project_is_approved']==1
          rejected_digits_in_summary = project_data[project_data['project_is_approved']==0

          plt.boxplot([approved_digits_in_summary, rejected_digits_in_summary])
          plt.title('Approved/Rejected projects for teachers')
          plt.xticks([1,2],('Approved Projects','Rejected Projects'))
          plt.ylabel('Digits in summary')
          plt.grid()
          plt.show()
```



Observation: We cannot get enough information from the above boxplot

In [389]:

```
plt.figure(figsize=(10,3))
sns.distplot(approved_digits_in_summary, hist=False, label="Approved Projects")
sns.distplot(rejected_digits_in_summary, hist=False, label="Not Approved Projects
plt.title('Approved and not approved Projects')
plt.xlabel('Digits in summary')
plt.legend()
plt.show()
```



Observation: As you can see Projects having digits in their summary ,have more approval chances.

In [390]:
```
from prettytable import PrettyTable

#If you get a ModuleNotFoundError error , install prettytable using: pip3 instal

x = PrettyTable()
x.field_names = ["Percentile", "Approved Projects", "Not Approved Projects"]

for i in range(0,101,5):
    x.add_row([i,np.round(np.percentile(approved_digits_in_summary,i), 3), np.ro
print(x)
```

```
+------------+-------------------+-----------------------+
| Percentile | Approved Projects | Not Approved Projects |
+------------+-------------------+-----------------------+
|     0      |        0.0        |          0.0          |
|     5      |        0.0        |          0.0          |
|     10     |        0.0        |          0.0          |
|     15     |        0.0        |          0.0          |
|     20     |        0.0        |          0.0          |
|     25     |        0.0        |          0.0          |
|     30     |        0.0        |          0.0          |
|     35     |        0.0        |          0.0          |
|     40     |        0.0        |          0.0          |
|     45     |        0.0        |          0.0          |
|     50     |        0.0        |          0.0          |
|     55     |        0.0        |          0.0          |
|     60     |        0.0        |          0.0          |
|     65     |        0.0        |          0.0          |
|     70     |        0.0        |          0.0          |
|     75     |        0.0        |          0.0          |
|     80     |        0.0        |          0.0          |
|     85     |        1.0        |          0.0          |
|     90     |        1.0        |          1.0          |
|     95     |        1.0        |          1.0          |
|    100     |        1.0        |          1.0          |
+------------+-------------------+-----------------------+
```

Observatio: From above table ,projects having digits in their summary have more Percentile i.e it starts from 85 Percentile,where as projects which doesn't have digits in their summary have less Percentile i.e it starts from 90th Percentile.

1.3 Text preprocessing

1.3.1 Essay Text

In [391]:

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

Out[391]:

	Unnamed: 0	id	teacher_id	teacher_prefix	school_state	project_sul
0	160221	p253737	c90749f5d961ff158d4b4d1e7dc665fc	Mrs.	IN	20
1	140945	p258326	897464ce9ddc600bced1151f324dd63a	Mr.	FL	20

In [392]:

```
# printing some random essays.
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)
print(project_data['essay'].values[99999])
print("="*50)
```

```
My students are English learners that are working on English as their second or
third languages. We are a melting pot of refugees, immigrants, and native-born
Americans bringing the gift of language to our school. \r\n\r\n We have over 24
languages represented in our English Learner program with students at every lev
el of mastery.  We also have over 40 countries represented with the families wi
thin our school.  Each student brings a wealth of knowledge and experiences to
us that open our eyes to new cultures, beliefs, and respect.\"The limits of you
r language are the limits of your world.\"-Ludwig Wittgenstein  Our English lea
rner's have a strong support system at home that begs for more resources.  Many
times our parents are learning to read and speak English along side of their ch
ildren.  Sometimes this creates barriers for parents to be able to help their c
hild learn phonetics, letter recognition, and other reading skills.\r\n\r\nBy p
roviding these dvd's and players, students are able to continue their mastery o
f the English language even if no one at home is able to assist.  All families
with students within the Level 1 proficiency status, will be a offered to be a
part of this program.  These educational videos will be specially chosen by the
English Learner Teacher and will be sent home regularly to watch.  The videos a
re to help the child develop early reading skills.\r\n\r\nParents that do not h
ave access to a dvd player will have the opportunity to check out a dvd player
to use for the year.  The plan is to use these videos and educational dvd's for
the years to come for other EL students.\r\nnannan
==================================================
The 51 fifth grade students that will cycle through my classroom this year all
love learning, at least most of the time. At our school, 97.3% of the students
receive free or reduced price lunch. Of the 560 students, 97.3% are minority st
udents. \r\nThe school has a vibrant community that loves to get together and c
elebrate. Around Halloween there is a whole school parade to show off the beaut
iful costumes that students wear. On Cinco de Mayo we put on a big festival wit
h crafts made by the students, dances, and games. At the end of the year the sc
hool hosts a carnival to celebrate the hard work put in during the school year,
with a dunk tank being the most popular activity.My students will use these fiv
e brightly colored Hokki stools in place of regular, stationary, 4-legged chair
s. As I will only have a total of ten in the classroom and not enough for each
student to have an individual one, they will be used in a variety of ways. Duri
ng independent reading time they will be used as special chairs students will e
ach use on occasion. I will utilize them in place of chairs at my small group t
ables during math and reading times. The rest of the day they will be used by t
he students who need the highest amount of movement in their life in order to s
tay focused on school.\r\n\r\nWhenever asked what the classroom is missing, my
students always say more Hokki Stools. They can't get their fill of the 5 stool
s we already have. When the students are sitting in group with me on the Hokki
Stools, they are always moving, but at the same time doing their work. Anytime
the students get to pick where they can sit, the Hokki Stools are the first to
```

```
be taken. There are always students who head over to the kidney table to get on
e of the stools who are disappointed as there are not enough of them. \r\n\r\nW
e ask a lot of students to sit for 7 hours a day. The Hokki stools will be a co
mpromise that allow my students to do desk work and move at the same time. Thes
e stools will help students to meet their 60 minutes a day of movement by allow
ing them to activate their core muscles for balance while they sit. For many of
my students, these chairs will take away the barrier that exists in schools for
a child who can't sit still.nannan
==================================================
How do you remember your days of school? Was it in a sterile environment with p
lain walls, rows of desks, and a teacher in front of the room? A typical day in
our room is nothing like that. I work hard to create a warm inviting themed roo
m for my students look forward to coming to each day.\r\n\r\nMy class is made u
p of 28 wonderfully unique boys and girls of mixed races in Arkansas.\r\nThey a
ttend a Title I school, which means there is a high enough percentage of free a
nd reduced-price lunch to qualify. Our school is an \"open classroom\" concept,
which is very unique as there are no walls separating the classrooms. These 9 a
nd 10 year-old students are very eager learners; they are like sponges, absorbi
ng all the information and experiences and keep on wanting more.With these reso
urces such as the comfy red throw pillows and the whimsical nautical hanging de
cor and the blue fish nets, I will be able to help create the mood in our class
room setting to be one of a themed nautical environment. Creating a classroom e
nvironment is very important in the success in each and every child's educatio
n. The nautical photo props will be used with each child as they step foot into
our classroom for the first time on Meet the Teacher evening. I'll take picture
s of each child with them, have them developed, and then hung in our classroom
ready for their first day of 4th grade.  This kind gesture will set the tone be
fore even the first day of school! The nautical thank you cards will be used th
roughout the year by the students as they create thank you cards to their team
groups.\r\n\r\nYour generous donations will help me to help make our classroom
a fun, inviting, learning environment from day one.\r\n\r\nIt costs lost of mon
ey out of my own pocket on resources to get our classroom ready. Please conside
r helping with this project to make our new school year a very successful one.
Thank you!nannan
==================================================
My kindergarten students have varied disabilities ranging from speech and langu
age delays, cognitive delays, gross/fine motor delays, to autism. They are eage
r beavers and always strive to work their hardest working past their limitation
s. \r\n\r\nThe materials we have are the ones I seek out for my students. I tea
ch in a Title I school where most of the students receive free or reduced price
lunch.  Despite their disabilities and limitations, my students love coming to
school and come eager to learn and explore.Have you ever felt like you had ants
in your pants and you needed to groove and move as you were in a meeting? This
is how my kids feel all the time. The want to be able to move as they learn or
so they say.Wobble chairs are the answer and I love then because they develop t
heir core, which enhances gross motor and in Turn fine motor skills. \r\nThey a
lso want to learn through games, my kids don't want to sit and do worksheets. T
hey want to learn to count by jumping and playing. Physical engagement is the k
ey to our success. The number toss and color and shape mats can make that happe
n. My students will forget they are doing work and just have the fun a 6 year o
ld deserves.nannan
==================================================
The mediocre teacher tells. The good teacher explains. The superior teacher dem
onstrates. The great teacher inspires. -William A. Ward\r\n\r\nMy school has 80
3 students which is makeup is 97.6% African-American, making up the largest seg
ment of the student body. A typical school in Dallas is made up of 23.2% Africa
n-American students. Most of the students are on free or reduced lunch. We are
```

```
n't receiving doctors, lawyers, or engineers children from rich backgrounds or
neighborhoods. As an educator I am inspiring minds of young children and we foc
us not only on academics but one smart, effective, efficient, and disciplined s
tudents with good character.In our classroom we can utilize the Bluetooth for s
wift transitions during class. I use a speaker which doesn't amplify the sound
enough to receive the message. Due to the volume of my speaker my students ca
n't hear videos or books clearly and it isn't making the lessons as meaningful.
But with the bluetooth speaker my students will be able to hear and I can stop,
pause and replay it at any time.\r\nThe cart will allow me to have more room fo
r storage of things that are needed for the day and has an extra part to it I c
an use.  The table top chart has all of the letter, words and pictures for stud
ents to learn about different letters and it is more accessible.nannan
==================================================
```

In [393]:

```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 [394]:
```
sent = decontracted(project_data['essay'].values[20000])
print(sent)
print("="*50)
```

```
My kindergarten students have varied disabilities ranging from speech and langu
age delays, cognitive delays, gross/fine motor delays, to autism. They are eage
r beavers and always strive to work their hardest working past their limitation
s. \r\n\r\nThe materials we have are the ones I seek out for my students. I tea
ch in a Title I school where most of the students receive free or reduced price
lunch.  Despite their disabilities and limitations, my students love coming to
school and come eager to learn and explore.Have you ever felt like you had ants
in your pants and you needed to groove and move as you were in a meeting? This
is how my kids feel all the time. The want to be able to move as they learn or
so they say.Wobble chairs are the answer and I love then because they develop t
heir core, which enhances gross motor and in Turn fine motor skills. \r\nThey a
lso want to learn through games, my kids do not want to sit and do worksheets.
They want to learn to count by jumping and playing. Physical engagement is the
key to our success. The number toss and color and shape mats can make that happ
en. My students will forget they are doing work and just have the fun a 6 year
old deserves.nannan
==================================================
```

In [395]:
```
# \r \n \t remove from string python: http://texthandler.com/info/remove-line-bre
sent = sent.replace('\\r', ' ')
sent = sent.replace('\\"', ' ')
sent = sent.replace('\\n', ' ')
print(sent)
```

```
My kindergarten students have varied disabilities ranging from speech and langu
age delays, cognitive delays, gross/fine motor delays, to autism. They are eage
r beavers and always strive to work their hardest working past their limitation
s.     The materials we have are the ones I seek out for my students. I teach i
n a Title I school where most of the students receive free or reduced price lun
ch.  Despite their disabilities and limitations, my students love coming to sch
ool and come eager to learn and explore.Have you ever felt like you had ants in
your pants and you needed to groove and move as you were in a meeting? This is
how my kids feel all the time. The want to be able to move as they learn or so
they say.Wobble chairs are the answer and I love then because they develop thei
r core, which enhances gross motor and in Turn fine motor skills.   They also w
ant to learn through games, my kids do not want to sit and do worksheets. They
want to learn to count by jumping and playing. Physical engagement is the key t
o our success. The number toss and color and shape mats can make that happen. M
y students will forget they are doing work and just have the fun a 6 year old d
eserves.nannan
```

In [396]:
```
#remove spacial character: https://stackoverflow.com/a/5843547/4084039
sent = re.sub('[^A-Za-z0-9]+', ' ', sent)
print(sent)
```

```
My kindergarten students have varied disabilities ranging from speech and langu
age delays cognitive delays gross fine motor delays to autism They are eager be
avers and always strive to work their hardest working past their limitations Th
e materials we have are the ones I seek out for my students I teach in a Title
I school where most of the students receive free or reduced price lunch Despite
their disabilities and limitations my students love coming to school and come e
ager to learn and explore Have you ever felt like you had ants in your pants an
d you needed to groove and move as you were in a meeting This is how my kids fe
el all the time The want to be able to move as they learn or so they say Wobble
chairs are the answer and I love then because they develop their core which enh
ances gross motor and in Turn fine motor skills They also want to learn through
games my kids do not want to sit and do worksheets They want to learn to count
by jumping and playing Physical engagement is the key to our success The number
toss and color and shape mats can make that happen My students will forget they
are doing work and just have the fun a 6 year old deserves nannan
```

In [397]:
```
# 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'll", "you'd", 'your', 'yours', 'yourself', 'yourselves', 'he',
            'she', "she's", 'her', 'hers', 'herself', 'it', "it's", 'its', 'itse
            'theirs', 'themselves', 'what', 'which', 'who', 'whom', 'this', 'tha
            'am', 'is', 'are', 'was', 'were', 'be', 'been', 'being', 'have', 'ha
            'did', 'doing', 'a', 'an', 'the', 'and', 'but', 'if', 'or', 'because
            'at', 'by', 'for', 'with', 'about', 'against', 'between', 'into', 't
            'above', 'below', 'to', 'from', 'up', 'down', 'in', 'out', 'on', 'of
            'then', 'once', 'here', 'there', 'when', 'where', 'why', 'how', 'all
            'most', 'other', 'some', 'such', 'only', 'own', 'same', 'so', 'than'
            's', 't', 'can', 'will', 'just', 'don', "don't", 'should', "should'v
            've', 'y', 'ain', 'aren', "aren't", 'couldn', "couldn't", 'didn', "d
            "hadn't", 'hasn', "hasn't", 'haven', "haven't", 'isn', "isn't", 'ma'
            "mustn't", 'needn', "needn't", 'shan', "shan't", 'shouldn', "shouldn
            'won', "won't", 'wouldn', "wouldn't"]
```

In [398]:
```
# Combining all the above statemennts
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%|██████████| 109248/109248 [01:18<00:00, 1394.57it/s]
```

In [399]:

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

Out[399]:

```
'my kindergarten students varied disabilities ranging speech language delays co
gnitive delays gross fine motor delays autism they eager beavers always strive
work hardest working past limitations the materials ones i seek students i teac
h title i school students receive free reduced price lunch despite disabilities
limitations students love coming school come eager learn explore have ever felt
like ants pants needed groove move meeting this kids feel time the want able mo
ve learn say wobble chairs answer i love develop core enhances gross motor turn
fine motor skills they also want learn games kids not want sit worksheets they
want learn count jumping playing physical engagement key success the number tos
s color shape mats make happen my students forget work fun 6 year old deserves
nannan'
```

1.3.2 Project title Text

In [400]:

```
# printing some random essays.
print(project_data['project_title'].values[0])
print("="*50)
print(project_data['project_title'].values[150])
print("="*50)
print(project_data['project_title'].values[1000])
print("="*50)
print(project_data['project_title'].values[20000])
print("="*50)
print(project_data['project_title'].values[99999])
print("="*50)
```

```
Educational Support for English Learners at Home
==================================================
More Movement with Hokki Stools
==================================================
Sailing Into a Super 4th Grade Year
==================================================
We Need To Move It While We Input It!
==================================================
Inspiring Minds by Enhancing the Educational Experience
==================================================
```

In [401]:

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

```
We Need To Move It While We Input It!
==================================================
```

In [402]:

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

```
We Need To Move It While We Input It!
```

In [403]:

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

```
We Need To Move It While We Input It
```

In [404]:

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

```
100%|██████████| 109248/109248 [00:03<00:00, 31776.54it/s]
```

In [405]:

```
preprocessed_titles[1000]
```

Out[405]:

```
'sailing into super 4th grade year'
```

1.3.3 Project grade

In [406]:

```
project_grade_catogories = list(project_data['project_grade_category'].values)
# remove special characters from list of strings python: https://stackoverflow.co

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

project_grade_cat_list = []
for i in tqdm1(project_grade_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",
        if 'The' in j.split(): # this will split each of the catogory based on sp
            j=j.replace('The','') # if we have the words "The" we are going to re
        j = j.replace(' ','') # we are placeing all the ' '(space) with ''(empty
        temp +=j.strip()+" "#" abc ".strip() will return "abc", remove the trail
        temp = temp.replace('&','_')
    project_grade_cat_list.append(temp.strip())
```

```
HBox(children=(IntProgress(value=0, max=109248), HTML(value='')))
```

In [407]:

```
project_data['clean_project_grade_category'] = project_grade_cat_list
project_data.drop(['project_grade_category'], axis=1, inplace=True)
project_data.head(2)
```

Out[407]:

	Unnamed: 0	id	teacher_id	teacher_prefix	school_state	project_sul
0	160221	p253737	c90749f5d961ff158d4b4d1e7dc665fc	Mrs.	IN	20
1	140945	p258326	897464ce9ddc600bced1151f324dd63a	Mr.	FL	20

1. 4 Preparing data for models

In [408]:
```
project_data.columns
```

Out[408]:
```
Index(['Unnamed: 0', 'id', 'teacher_id', 'teacher_prefix', 'school_state',
       'project_submitted_datetime', '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',
       'clean_categories', 'clean_subcategories', 'essay', 'price', 'quantity',
       'digits_in_summary', 'clean_project_grade_category'],
      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

- quantity : numerical
- teacher_number_of_previously_posted_projects : numerical
- price : numerical
```

1.4.1 Vectorizing Categorical data

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

In [409]:
```
# # we use count vectorizer to convert the values into one hot encoded features
# from sklearn.feature_extraction.text import CountVectorizer
# vectorizer1 = CountVectorizer(vocabulary=list(sorted_cat_dict.keys()), lowercas
# vectorizer1.fit(project_data['clean_categories'].values)
# print(vectorizer1.get_feature_names())

# vectorizer2 = CountVectorizer( lowercase=False, binary=True)
# vectorizer2.fit(project_data['clean_categories'].values)
# print(vectorizer2.get_feature_names())
# # categories_one_hot = vectorizer.transform(project_data['clean_categories'].va
# # print("Shape of matrix after one hot encodig ",categories_one_hot.shape)
```

In [410]:

```
# we use count vectorizer to convert the values into one hot encoded features
from sklearn.feature_extraction.text import CountVectorizer
vectorizer = CountVectorizer(vocabulary=list(sorted_cat_dict.keys()), lowercase=
vectorizer.fit(project_data['clean_categories'].values)
print(vectorizer.get_feature_names())


categories_one_hot = vectorizer.transform(project_data['clean_categories'].value
print("Shape of matrix after one hot encodig ",categories_one_hot.shape)
```

```
['Warmth', 'Care_Hunger', 'History_Civics', 'Music_Arts', 'AppliedLearning', 'S
pecialNeeds', 'Health_Sports', 'Math_Science', 'Literacy_Language']
Shape of matrix after one hot encodig  (109248, 9)
```

In [411]:

```
# we use count vectorizer to convert the values into one hot encoded features
vectorizer = CountVectorizer(vocabulary=list(sorted_sub_cat_dict.keys()), lowerc
vectorizer.fit(project_data['clean_subcategories'].values)
print(vectorizer.get_feature_names())


sub_categories_one_hot = vectorizer.transform(project_data['clean_subcategories'
print("Shape of matrix after one hot encodig ",sub_categories_one_hot.shape)
```

```
['Economics', 'CommunityService', 'FinancialLiteracy', 'ParentInvolvement', 'Ex
tracurricular', 'Civics_Government', 'ForeignLanguages', 'NutritionEducation',
'Warmth', 'Care_Hunger', 'SocialSciences', 'PerformingArts', 'CharacterEducatio
n', 'TeamSports', 'Other', 'College_CareerPrep', 'Music', 'History_Geography',
'Health_LifeScience', 'EarlyDevelopment', 'ESL', 'Gym_Fitness', 'EnvironmentalS
cience', 'VisualArts', 'Health_Wellness', 'AppliedSciences', 'SpecialNeeds', 'L
iterature_Writing', 'Mathematics', 'Literacy']
Shape of matrix after one hot encodig  (109248, 30)
```

In [412]:

```
# Please do the similar feature encoding with state, teacher_prefix and project_
```

In [413]:

```
# we use count vectorizer to convert the values into one hot encoded features
from sklearn.feature_extraction.text import CountVectorizer
vectorizer = CountVectorizer( lowercase=False, binary=True)
vectorizer.fit(project_data['school_state'].values)
print(vectorizer.get_feature_names())


school_state_one_hot = vectorizer.transform(project_data['school_state'].values)
print("Shape of matrix after one hot encodig ",school_state_one_hot.shape)
```

```
['AK', 'AL', 'AR', 'AZ', 'CA', 'CO', 'CT', 'DC', 'DE', 'FL', 'GA', 'HI', 'IA',
'ID', 'IL', 'IN', 'KS', 'KY', 'LA', 'MA', 'MD', 'ME', 'MI', 'MN', 'MO', 'MS',
'MT', 'NC', 'ND', 'NE', 'NH', 'NJ', 'NM', 'NV', 'NY', 'OH', 'OK', 'OR', 'PA',
'RI', 'SC', 'SD', 'TN', 'TX', 'UT', 'VA', 'VT', 'WA', 'WI', 'WV', 'WY']
Shape of matrix after one hot encodig  (109248, 51)
```

In [414]:
```
#Replacing Nan's with maximum occured value: https://stackoverflow.com/a/5105391
project_data['teacher_prefix'].value_counts().argmax()
project_data.fillna(value=project_data['teacher_prefix'].value_counts().argmax()
```

In [415]:
```
# we use count vectorizer to convert the values into one hot encoded features
from sklearn.feature_extraction.text import CountVectorizer
vectorizer = CountVectorizer( lowercase=False, binary=True)
vectorizer.fit(project_data['teacher_prefix'].values.astype('U'))
print(vectorizer.get_feature_names())

#https://stackoverflow.com/a/39308809/8089731
teacher_prefix_one_hot = vectorizer.transform(project_data['teacher_prefix'].valu
print("Shape of matrix after one hot encodig ",teacher_prefix_one_hot.shape)
```

```
['Dr', 'Mr', 'Mrs', 'Ms', 'Teacher']
Shape of matrix after one hot encodig  (109248, 5)
```

In [416]:
```
print(project_data['clean_project_grade_category'].unique())
```

```
['GradesPreK-2' 'Grades6-8' 'Grades3-5' 'Grades9-12']
```

In [417]:
```
# we use count vectorizer to convert the values into one hot encoded features
from sklearn.feature_extraction.text import CountVectorizer
# https://stackoverflow.com/a/38161028/8089731
pattern = "(?u)\\b[\\w-]+\\b"
vectorizer = CountVectorizer(token_pattern=pattern, lowercase=False, binary=True
vectorizer.fit(project_data['clean_project_grade_category'].values)
print(vectorizer.get_feature_names())

#https://stackoverflow.com/a/39308809/8089731
project_grade_category_one_hot = vectorizer.transform(project_data['clean_projec
print("Shape of matrix after one hot encodig ",project_grade_category_one_hot.sh
```

```
['Grades3-5', 'Grades6-8', 'Grades9-12', 'GradesPreK-2']
Shape of matrix after one hot encodig  (109248, 4)
```

1.4.2 Vectorizing Text data

1.4.2.1 Bag of words

In [418]:
```
# We are considering only the words which appeared in at least 10 documents(rows
vectorizer = CountVectorizer(min_df=10)
text_bow = vectorizer.fit_transform(preprocessed_essays)
print("Shape of matrix after one hot encodig ",text_bow.shape)
```

```
Shape of matrix after one hot encodig  (109248, 16623)
```

1.4.2.2 Bag of Words on `project_title`

In [419]:

```
# you can vectorize the title also
# before you vectorize the title make sure you preprocess it
```

In [420]:

```
# Similarly you can vectorize for title also
# We are considering only the words which appeared in at least 10 documents(rows
vectorizer = CountVectorizer(min_df=10)
title_bow = vectorizer.fit_transform(preprocessed_titles)
print("Shape of matrix after one hot encodig ",title_bow.shape)
```

```
Shape of matrix after one hot encodig  (109248, 3329)
```

1.4.2.3 TFIDF vectorizer

In [421]:

```
from sklearn.feature_extraction.text import TfidfVectorizer
vectorizer = TfidfVectorizer(min_df=10)
text_tfidf = vectorizer.fit_transform(preprocessed_essays)
print("Shape of matrix after one hot encodig ",text_tfidf.shape)
```

```
Shape of matrix after one hot encodig  (109248, 16623)
```

1.4.2.4 TFIDF Vectorizer on `project_title`

In [422]:

```
# Similarly you can vectorize for title also
from sklearn.feature_extraction.text import TfidfVectorizer
vectorizer = TfidfVectorizer(min_df=10)
title_tfidf = vectorizer.fit_transform(preprocessed_titles)
print("Shape of matrix after one hot encodig ",title_tfidf.shape)
```

```
Shape of matrix after one hot encodig  (109248, 3329)
```

1.4.2.5 Using Pretrained Models: Avg W2V

In [423]:

```
'''
# Reading glove vectors in python: https://stackoverflow.com/a/38230349/4084039
def loadGloveModel(gloveFile):
    print ("Loading Glove Model")
    f = open(gloveFile,'r', encoding="utf8")
    model = {}
    for line in tqdm(f):
        splitLine = line.split()
        word = splitLine[0]
        embedding = np.array([float(val) for val in splitLine[1:]])
        model[word] = embedding
    print ("Done.",len(model)," words loaded!")
    return model
model = loadGloveModel('glove.42B.300d.txt')

# ============================
Output:

Loading Glove Model
1917495it [06:32, 4879.69it/s]
Done. 1917495  words loaded!

# ============================

words = []
for i in preproced_texts:
    words.extend(i.split(' '))

for i in preproced_titles:
    words.extend(i.split(' '))
print("all the words in the coupus", len(words))
words = set(words)
print("the unique words in the coupus", len(words))

inter_words = set(model.keys()).intersection(words)
print("The number of words that are present in both glove vectors and our coupus
      len(inter_words),"(",np.round(len(inter_words)/len(words)*100,3),"%)")

words_courpus = {}
words_glove = set(model.keys())
for i in words:
    if i in words_glove:
        words_courpus[i] = model[i]
print("word 2 vec length", len(words_courpus))


# stronging variables into pickle files python: http://www.jessicayung.com/how-t

import pickle
with open('glove_vectors', 'wb') as f:
    pickle.dump(words_courpus, f)


'''
```

Out[423]:

```
'\n# Reading glove vectors in python: https://stackoverflow.com/a/38230349/4084
039\ndef (https://stackoverflow.com/a/38230349/4084039\ndef) loadGloveModel(glo
veFile):\n    print ("Loading Glove Model")\n    f = open(gloveFile,\'r\', enco
ding="utf8")\n    model = {}\n    for line in tqdm(f):\n        splitLine = lin
e.split()\n        word = splitLine[0]\n        embedding = np.array([float(va
l) for val in splitLine[1:]])\n        model[word] = embedding\n    print ("Don
e.",len(model)," words loaded!")\n    return model\nmodel = loadGloveModel(\'gl
ove.42B.300d.txt\')\n\n# ============================\nOutput:\n    \nLoading G
love Model\n1917495it [06:32, 4879.69it/s]\nDone. 1917495  words loaded!\n\n# =
===========================\n\nwords = []\nfor i in preproced_texts:\n    word
s.extend(i.split(\' \'))\n\nfor i in preproced_titles:\n    words.extend(i.spli
t(\' \'))\nprint("all the words in the coupus", len(words))\nwords = set(words)
\nprint("the unique words in the coupus", len(words))\n\ninter_words = set(mode
l.keys()).intersection(words)\nprint("The number of words that are present in b
oth glove vectors and our coupus",      len(inter_words),"(",np.round(len(inte
r_words)/len(words)*100,3),"%)")\n\nwords_courpus = {}\nwords_glove = set(mode
l.keys())\nfor i in words:\n    if i in words_glove:\n        words_courpus[i]
 = model[i]\nprint("word 2 vec length", len(words_courpus))\n\n\n# stronging va
riables into pickle files python: http://www.jessicayung.com/how-to-use-pickle-
to-save-and-load-variables-in-python/\n\nimport (http://www.jessicayung.com/how
-to-use-pickle-to-save-and-load-variables-in-python/\n\nimport) pickle\nwith op
en(\'glove_vectors\', \'wb\') as f:\n    pickle.dump(words_courpus, f)\n\n\n'
```

In [424]:

```python
# stronging variables into pickle files python: http://www.jessicayung.com/how-to
# make sure you have the glove_vectors file
with open('../input/glovee/glove_vectors', 'rb') as f:
    model = pickle.load(f)
    glove_words =  set(model.keys())
```

In [425]:

```python
# print(glove_words)
```

In [426]:

```
# average Word2Vec
# compute average word2vec for each review.
avg_w2v_preprocessed_essays_vectors = []; # the avg-w2v for each sentence/review
for sentence in tqdm1(preprocessed_essays): # for each review/sentence
    vector = np.zeros(300) # as word vectors are of zero length
    cnt_words =0; # num of words with a valid vector in the sentence/review
    for word in sentence.split(): # for each word in a review/sentence
        if word in glove_words:
            vector += model[word]
            cnt_words += 1
    if cnt_words != 0:
        vector /= cnt_words
    avg_w2v_preprocessed_essays_vectors.append(vector)

print(len(avg_w2v_preprocessed_essays_vectors))
print(len(avg_w2v_preprocessed_essays_vectors[0]))
```

```
HBox(children=(IntProgress(value=0, max=109248), HTML(value='')))


109248
300
```

1.4.2.6 Using Pretrained Models: AVG W2V on `project_title`

In [427]:

```
# Similarly you can vectorize for title also
# average Word2Vec
# compute average word2vec for each title.
avg_w2v_preprocessed_titles_vectors = []; # the avg-w2v for each sentence/review
for sentence in tqdm1(preprocessed_titles): # for each review/sentence
    vector = np.zeros(300) # as word vectors are of zero length
    cnt_words =0; # num of words with a valid vector in the sentence/review
    for word in sentence.split(): # for each word in a review/sentence
        if word in glove_words:
            vector += model[word]
            cnt_words += 1
    if cnt_words != 0:
        vector /= cnt_words
    avg_w2v_preprocessed_titles_vectors.append(vector)

print(len(avg_w2v_preprocessed_titles_vectors))
print(len(avg_w2v_preprocessed_titles_vectors[0]))
```

```
HBox(children=(IntProgress(value=0, max=109248), HTML(value='')))


109248
300
```

1.4.2.7 Using Pretrained Models: TFIDF weighted W2V

In [428]:

```
# S = ["abc def pqr", "def def def abc", "pqr pqr def"]
tfidf_model = TfidfVectorizer()
tfidf_model.fit(preprocessed_essays)
# we are converting a dictionary with word as a key, and the idf as a value
dictionary = dict(zip(tfidf_model.get_feature_names(), list(tfidf_model.idf_)))
tfidf_words = set(tfidf_model.get_feature_names())
```

In [429]:

```
# average Word2Vec
# compute average word2vec for each review.
tfidf_w2v_preprocessed_essays_vectors = []; # the avg-w2v for each sentence/revi
for sentence in tqdm1(preprocessed_essays): # for each review/sentence
    vector = np.zeros(300) # as word vectors are of zero length
    tf_idf_weight =0; # num of words with a valid vector in the sentence/review
    for word in sentence.split(): # for each word in a review/sentence
        if (word in glove_words) and (word in tfidf_words):
            vec = model[word] # getting the vector for each word
            # here we are multiplying idf value(dictionary[word]) and the tf val
            tf_idf = dictionary[word]*(sentence.count(word)/len(sentence.split()
            vector += (vec * tf_idf) # calculating tfidf weighted w2v
            tf_idf_weight += tf_idf
    if tf_idf_weight != 0:
        vector /= tf_idf_weight
    tfidf_w2v_preprocessed_essays_vectors.append(vector)

print(len(tfidf_w2v_preprocessed_essays_vectors))
print(len(tfidf_w2v_preprocessed_essays_vectors[0]))
```

```
HBox(children=(IntProgress(value=0, max=109248), HTML(value='')))


109248
300
```

1.4.2.9 Using Pretrained Models: TFIDF weighted W2V on `project_title`

In [430]:

```
# Similarly you can vectorize for title also
# S = ["abc def pqr", "def def def abc", "pqr pqr def"]
tfidf_model = TfidfVectorizer()
tfidf_model.fit(preprocessed_titles)
# we are converting a dictionary with word as a key, and the idf as a value
dictionary = dict(zip(tfidf_model.get_feature_names(), list(tfidf_model.idf_)))
tfidf_words = set(tfidf_model.get_feature_names())
```

In [431]:

```
# average Word2Vec
# compute average word2vec for each review.
tfidf_w2v_preprocessed_titles_vectors = []; # the avg-w2v for each sentence/revi
for sentence in tqdm1(preprocessed_titles): # for each review/sentence
    vector = np.zeros(300) # as word vectors are of zero length
    tf_idf_weight =0; # num of words with a valid vector in the sentence/review
    for word in sentence.split(): # for each word in a review/sentence
        if (word in glove_words) and (word in tfidf_words):
            vec = model[word] # getting the vector for each word
            # here we are multiplying idf value(dictionary[word]) and the tf val
            tf_idf = dictionary[word]*(sentence.count(word)/len(sentence.split()
            vector += (vec * tf_idf) # calculating tfidf weighted w2v
            tf_idf_weight += tf_idf
    if tf_idf_weight != 0:
        vector /= tf_idf_weight
    tfidf_w2v_preprocessed_titles_vectors.append(vector)

print(len(tfidf_w2v_preprocessed_titles_vectors))
print(len(tfidf_w2v_preprocessed_titles_vectors[0]))
```

```
HBox(children=(IntProgress(value=0, max=109248), HTML(value='')))


109248
300
```

1.4.3 Vectorizing Numerical features

In [432]:

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

# price_standardized = standardScalar.fit(project_data['price'].values)
# this will rise the error
# ValueError: Expected 2D array, got 1D array instead: array=[725.05 213.03 329.
# Reshape your data either using array.reshape(-1, 1)

price_scalar = StandardScaler()
price_scalar.fit(project_data['price'].values.reshape(-1,1)) # finding the mean
print(f"Mean : {price_scalar.mean_[0]}, Standard deviation : {np.sqrt(price_scal

# Now standardize the data with above maen and variance.
price_standardized = price_scalar.transform(project_data['price'].values.reshape
print(price_standardized.shape)
```

```
Mean : 298.1193425966608, Standard deviation : 367.49634838483496
(109248, 1)
```

In [433]:

```
print(price_scalar.mean_[0])
print(price_scalar.mean_)
```

```
298.1193425966608
[298.1193426]
```

In [434]:
```
price_standardized
```

Out[434]:
```
array([[-0.3905327 ],
       [ 0.00239637],
       [ 0.59519138],
       ...,
       [-0.15825829],
       [-0.61243967],
       [-0.51216657]])
```

In [435]:
```
project_data.columns
```

Out[435]:
```
Index(['Unnamed: 0', 'id', 'teacher_id', 'teacher_prefix', 'school_state',
       'project_submitted_datetime', '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',
       'clean_categories', 'clean_subcategories', 'essay', 'price', 'quantity',
       'digits_in_summary', 'clean_project_grade_category'],
      dtype='object')
```

In [436]:
```
# check this one: https://www.youtube.com/watch?v=0HOqOcln3Z4&t=530s
# standardization sklearn: https://scikit-learn.org/stable/modules/generated/skle
from sklearn.preprocessing import StandardScaler

# price_standardized = standardScalar.fit(project_data['price'].values)
# this will rise the error
# ValueError: Expected 2D array, got 1D array instead: array=[725.05 213.03 329.
# Reshape your data either using array.reshape(-1, 1)

quantity_scalar = StandardScaler()
quantity_scalar.fit(project_data['quantity'].values.reshape(-1,1)) # finding the
print(f"Mean : {quantity_scalar.mean_[0]}, Standard deviation : {np.sqrt(quantity

# Now standardize the data with above maen and variance.
quantity_standardized = quantity_scalar.transform(project_data['quantity'].value
print(quantity_standardized.shape)
```

```
Mean : 16.965610354422964, Standard deviation : 26.182821919093175
(109248, 1)
```

In [437]:
```
quantity_standardized
```

Out[437]:
```
array([[ 0.23047132],
       [-0.60977424],
       [ 0.19227834],
       ...,
       [-0.4951953 ],
       [-0.03687954],
       [-0.45700232]])
```

In [438]:

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

# price_standardized = standardScalar.fit(project_data['price'].values)
# this will rise the error
# ValueError: Expected 2D array, got 1D array instead: array=[725.05 213.03 329.
# Reshape your data either using array.reshape(-1, 1)

teacher_number_of_previously_posted_projects_scalar = StandardScaler()
teacher_number_of_previously_posted_projects_scalar.fit(project_data['teacher_nu
print(f"Mean : {teacher_number_of_previously_posted_projects_scalar.mean_[0]}, S

# Now standardize the data with above maen and variance.
teacher_number_of_previously_posted_projects_standardized = teacher_number_of_pr
print(teacher_number_of_previously_posted_projects_standardized.shape)
```

```
Mean : 11.153165275336848, Standard deviation : 27.77702641477403
(109248, 1)
```

In [439]:

```
teacher_number_of_previously_posted_projects_standardized
```

Out[439]:

```
array([[-0.40152481],
       [-0.14951799],
       [-0.36552384],
       ...,
       [-0.29352189],
       [-0.40152481],
       [-0.40152481]])
```

1.4.4 Merging all the above features

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

In [440]:

```python
print(categories_one_hot.shape)
print(sub_categories_one_hot.shape)
print(text_bow.shape)
print(title_bow.shape)
print(school_state_one_hot.shape)
print(teacher_prefix_one_hot.shape)
print(project_grade_category_one_hot.shape)
print(text_tfidf.shape)
print(title_tfidf.shape)
# print(avg_w2v_preprocessed_essays_vectors.shape)
# print(avg_w2v_preprocessed_titles_vectors.shape)
# print(tfidf_w2v_preprocessed_essays_vectors.shape)
# print(tfidf_w2v_preprocessed_titles_vectors.shape)
print(price_standardized.shape)
print(quantity_standardized.shape)
print(teacher_number_of_previously_posted_projects_standardized.shape)
```

```
(109248, 9)
(109248, 30)
(109248, 16623)
(109248, 3329)
(109248, 51)
(109248, 5)
(109248, 4)
(109248, 16623)
(109248, 3329)
(109248, 1)
(109248, 1)
(109248, 1)
```

Assignment 2: Apply TSNE

If you are using any code snippet from the internet, you have to provide the reference/citations, as we did in the above cells. Otherwise, it will be treated as plagiarism without citations.

Type *Markdown* and LaTeX: α^2

1. In the above cells we have plotted and analyzed many features. Please observe the plots and write the observations in markdown cells below every plot.
2. EDA: Please complete the analysis of the feature: teacher_number_of_previously_posted_projects
3. Build the data matrix using these features
    - school_state : categorical data (one hot encoding)
    - clean_categories : categorical data (one hot encoding)
    - clean_subcategories : categorical data (one hot encoding)
    - teacher_prefix : categorical data (one hot encoding)
    - project_grade_category : categorical data (one hot encoding)
    - project_title : text data (BOW, TFIDF, AVG W2V, TFIDF W2V)
    - price : numerical
    - teacher_number_of_previously_posted_projects : numerical

4. Now, plot FOUR t-SNE plots with each of these feature sets.
    A. categorical, numerical features + project_title(BOW)
    B. categorical, numerical features + project_title(TFIDF)
    C. categorical, numerical features + project_title(AVG W2V)
    D. categorical, numerical features + project_title(TFIDF W2V)
5. Concatenate all the features and Apply TNSE on the final data matrix
6. Note 1: The TSNE accepts only dense matrices
7. Note 2: Consider only 5k to 6k data points to avoid memory issues. If you run into memory error issues, reduce the number of data points but clearly state the number of datat-poins you are using

In [441]:

```
# # this is the example code for TSNE
import numpy as np
from sklearn.manifold import TSNE
from sklearn import datasets
import pandas as pd
import matplotlib.pyplot as plt

# iris = datasets.load_iris()
# xex = iris['data']
# yex = iris['target']

# tsneex = TSNE(n_components=2, perplexity=30, learning_rate=200)

# X_embeddingex = tsneex.fit_transform(xex)
# # if x is a sparse matrix you need to pass it as X_embedding = tsne.fit_transf

# for_tsneex = np.hstack((X_embeddingex, yex.reshape(-1,1)))
# for_tsne_dfex = pd.DataFrame(data=for_tsneex, columns=['Dimension_x','Dimensio
# sns.set_style("whitegrid");
# sns.FacetGrid(for_tsne_dfex, hue="Score", size=4)\
#    .map(plt.scatter, "Dimension_x", "Dimension_y") \
#    .add_legend();
# plt.show();
# # print(X_embedding)
# # print(y.reshape(-1,1).shape)
```

2.1 TSNE with `BOW` encoding of `project_title` feature

In [442]:

```
# merge two sparse matrices: https://stackoverflow.com/a/19710648/4084039
from scipy.sparse import hstack
# with the same hstack function we are concatinating a sparse matrix and a dense
X1 = hstack((school_state_one_hot,categories_one_hot, sub_categories_one_hot,tea
X1.shape
```

Out[442]:

```
(109248, 3430)
```

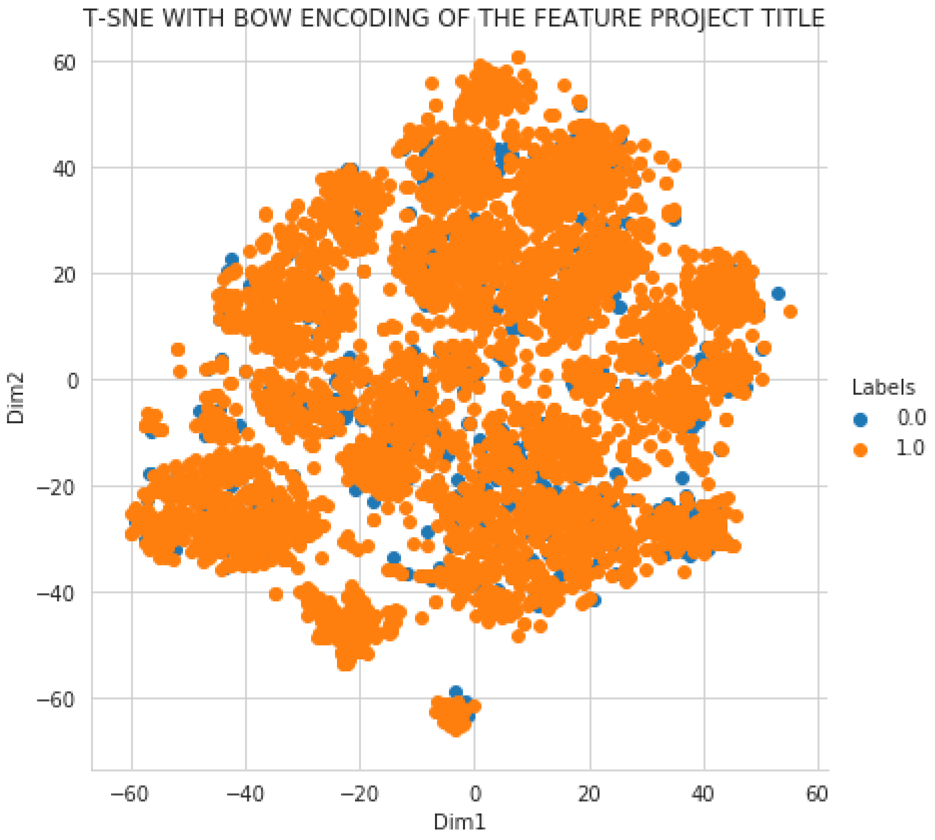
In [443]:

```
# Reducing to 5k data points for both X and y label
X1 = X1.tocsr() #https://stackoverflow.com/a/30175105/8089731
X1 = X1[0:5000]
print(X1.shape)

y = project_data['project_is_approved'][0:5000]
print(y.shape)
```

```
(5000, 3430)
(5000,)
```

In [444]:

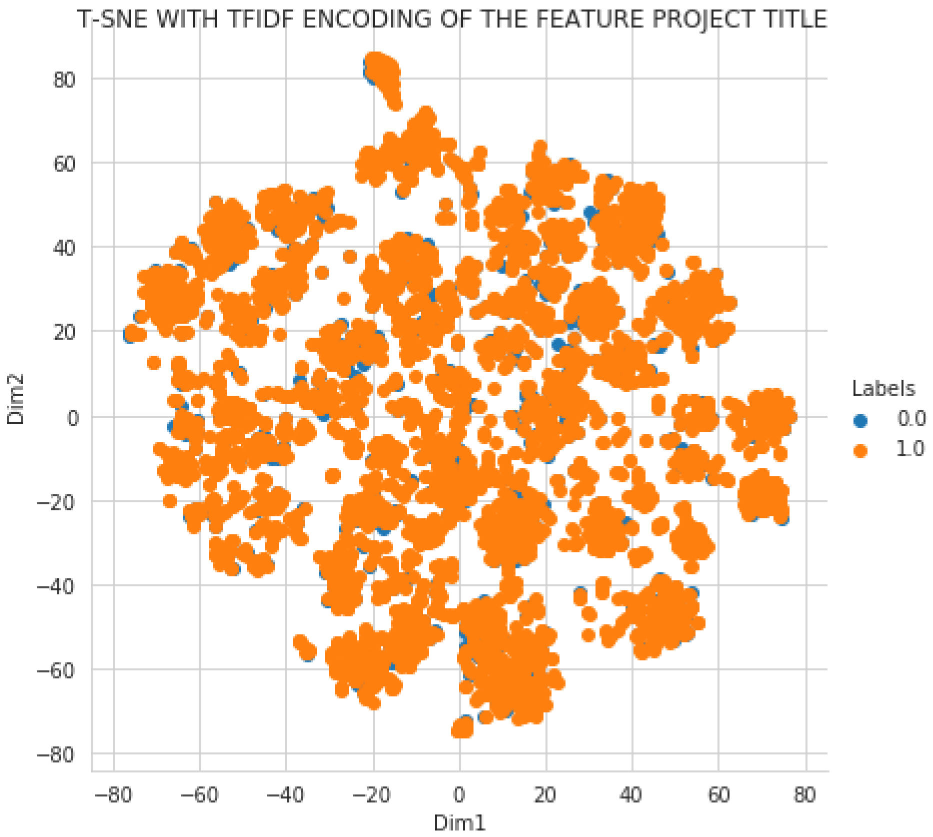
```
# please write all of the code with proper documentation and proper titles for ea
# when you plot any graph make sure you use
    # a. Title, that describes your plot, this will be very helpful to the reader
    # b. Legends if needed
    # c. X-axis label
    # d. Y-axis label

tsne = TSNE(n_components=2, perplexity=30, learning_rate=200)
X1_embedding = tsne.fit_transform(X1.toarray())
for_tsne1 = np.hstack((X1_embedding, y.values.reshape(-1,1)))
for_tsne_df1 = pd.DataFrame(data=for_tsne1, columns=['Dim1','Dim2','Labels'])
sns.set_style("whitegrid");
sns.FacetGrid(for_tsne_df1, hue="Labels", size=6)\
   .map(plt.scatter, "Dim1", "Dim2") \
   .add_legend()\
   .fig.suptitle("T-SNE WITH BOW ENCODING OF THE FEATURE PROJECT TITLE ");
plt.show();
```



Observation:We observe a lot of overlapping happening between approved and not approved projects.

2.2 TSNE with `TFIDF` encoding of `project_title` feature

In [445]:
```
# please write all the code with proper documentation, and proper titles for eacl
# 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
# Reducing to 5k data points for both X label
X2 = hstack((school_state_one_hot,categories_one_hot, sub_categories_one_hot,tea
X2 = X2.tocsr() #https://stackoverflow.com/a/30175105/8089731
X2 = X2[0:5000]

tsne2 = TSNE(n_components=2, perplexity=30, learning_rate=200)
X2_embedding = tsne2.fit_transform(X2.toarray())
for_tsne2 = np.hstack((X2_embedding, y.values.reshape(-1,1)))
for_tsne_df2 = pd.DataFrame(data=for_tsne2, columns=['Dim1','Dim2','Labels'])
sns.set_style("whitegrid");
sns.FacetGrid(for_tsne_df2, hue="Labels", size=6)\
   .map(plt.scatter, "Dim1", "Dim2") \
   .add_legend()\
   .fig.suptitle("T-SNE WITH TFIDF ENCODING OF THE FEATURE PROJECT TITLE ");
plt.show();
```

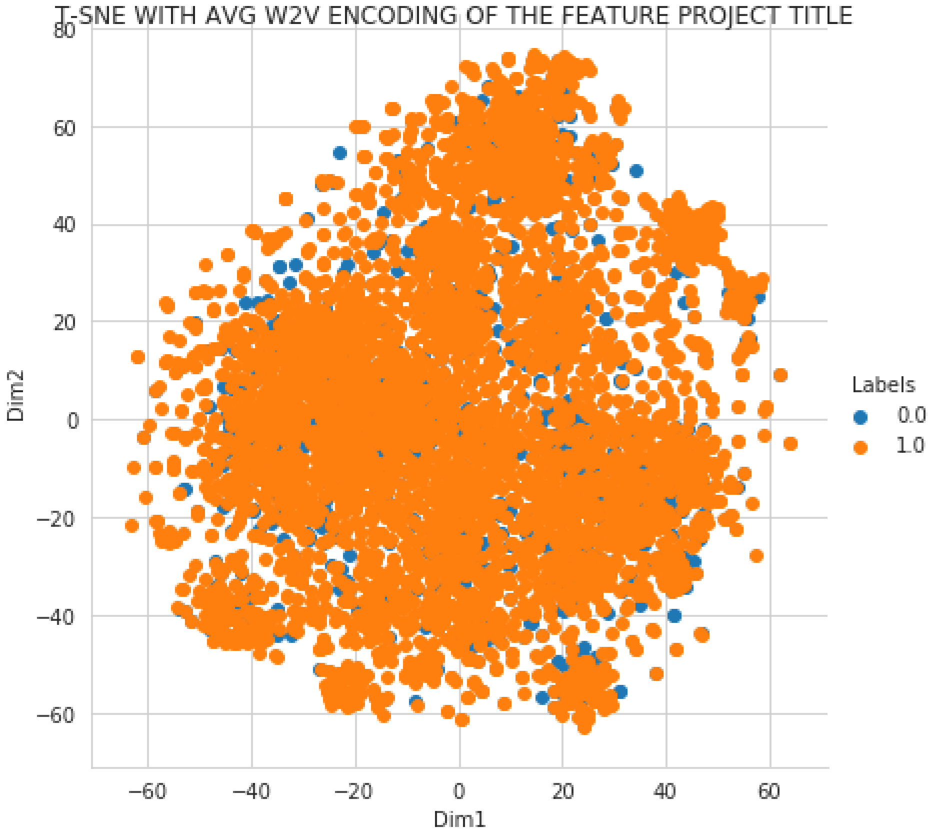


Observation:The blue and orange points seems to be overlapping which is difficult to classify a project is approved or not with TFIDF of project title

2.3 TSNE with `AVG W2V` encoding of `project_title` feature

In [446]:

```
# please write all the code with proper documentation, and proper titles for each
# 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
# Reducing to 5k data points for both X label
X3 = hstack((school_state_one_hot,categories_one_hot, sub_categories_one_hot,tea
X3 = X3.tocsr() #https://stackoverflow.com/a/30175105/8089731
X3 = X3[0:5000]

tsne3 = TSNE(n_components=2, perplexity=30, learning_rate=200)
X3_embedding = tsne3.fit_transform(X3.toarray())
for_tsne3 = np.hstack((X3_embedding, y.values.reshape(-1,1)))
for_tsne_df3 = pd.DataFrame(data=for_tsne3, columns=['Dim1','Dim2','Labels'])
sns.set_style("whitegrid");
sns.FacetGrid(for_tsne_df3, hue="Labels", size=6)\
   .map(plt.scatter, "Dim1", "Dim2") \
   .add_legend()\
   .fig.suptitle("T-SNE WITH AVG W2V ENCODING OF THE FEATURE PROJECT TITLE ");
plt.show();
```



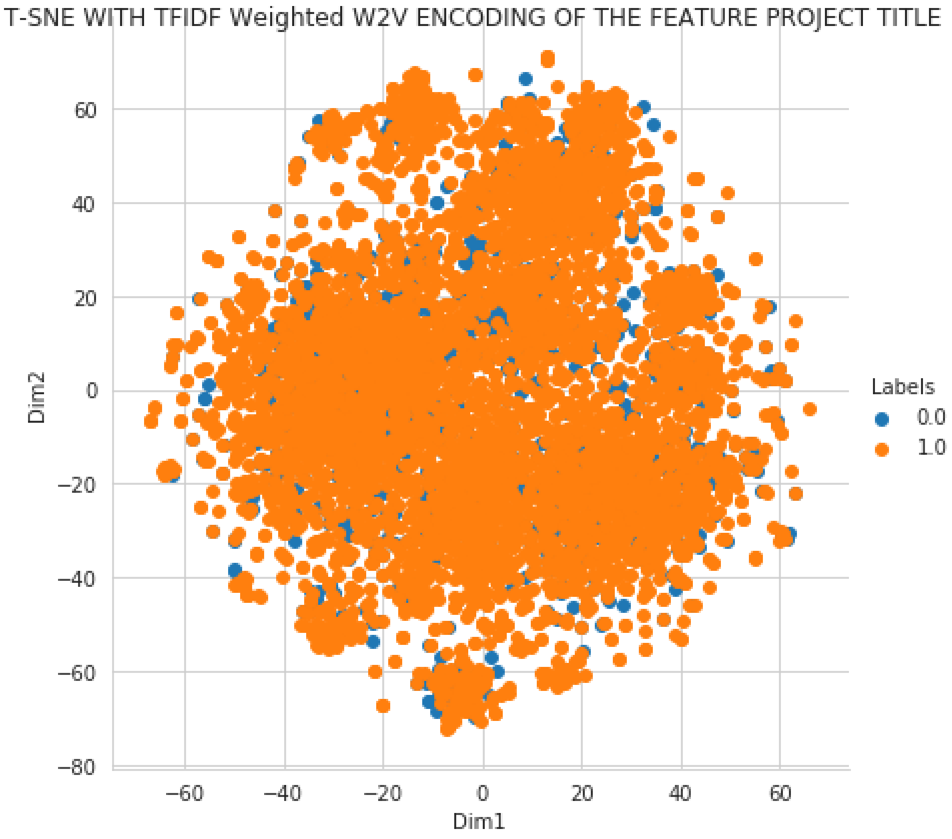
Observation:We cannot get any clusters to classify weather a project is approved or not ,so we cannot get the desired output from AVG W2V of

project title.

2.4 TSNE with `TFIDF Weighted W2V` encoding of `project_title` feature

In [447]:

```
# please write all the code with proper documentation, and proper titles for eac
# when you plot any graph make sure you use
    # a. Title, that describes your plot, this will be very helpful to the reade
    # b. Legends if needed
    # c. X-axis label
    # d. Y-axis label
# Reducing to 5k data points for both X label
X4 = hstack((school_state_one_hot,categories_one_hot, sub_categories_one_hot,tea
X4 = X4.tocsr() #https://stackoverflow.com/a/30175105/8089731
X4 = X4[0:5000]

tsne4 = TSNE(n_components=2, perplexity=30, learning_rate=200)
X4_embedding = tsne4.fit_transform(X4.toarray())
for_tsne4 = np.hstack((X4_embedding, y.values.reshape(-1,1)))
for_tsne_df4 = pd.DataFrame(data=for_tsne4, columns=['Dim1','Dim2','Labels'])
sns.set_style("whitegrid");
sns.FacetGrid(for_tsne_df4, hue="Labels", size=6)\
   .map(plt.scatter, "Dim1", "Dim2") \
   .add_legend()\
   .fig.suptitle("T-SNE WITH TFIDF Weighted W2V ENCODING OF THE FEATURE PROJECT 
plt.show();
```

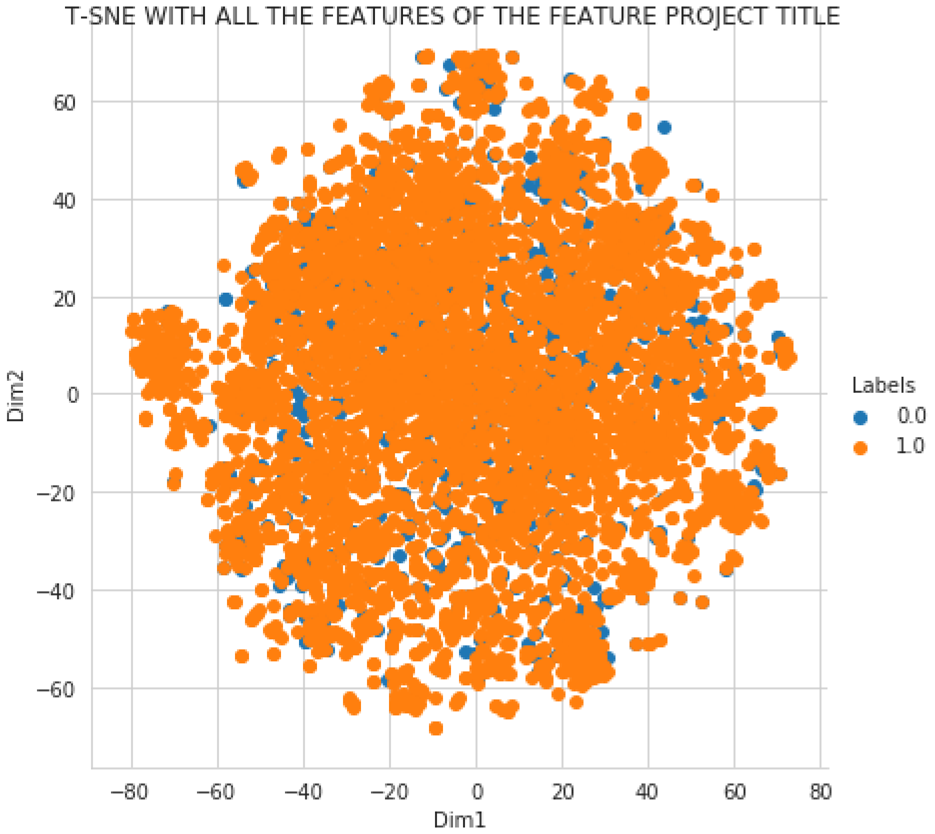


Observation:This visualisation of TSNE with TF-IDF Weighted Word2Vec does not seem to get the expected result of clusters of similar data points.

2.5 TSNE with `ALL THE FEATURES` of `project_title` feature

In [448]:

```
# Reducing to 5k data points for both X label
X5 = hstack((school_state_one_hot,categories_one_hot, sub_categories_one_hot,tea
X5 = X5.tocsr() #https://stackoverflow.com/a/30175105/8089731
X5 = X5[0:5000]

tsne5 = TSNE(n_components=2, perplexity=30, learning_rate=200)
X5_embedding = tsne5.fit_transform(X5.toarray())
for_tsne5 = np.hstack((X5_embedding, y.values.reshape(-1,1)))
for_tsne_df5 = pd.DataFrame(data=for_tsne5, columns=['Dim1','Dim2','Labels'])
sns.set_style("whitegrid");
sns.FacetGrid(for_tsne_df5, hue="Labels", size=6)\
   .map(plt.scatter, "Dim1", "Dim2") \
   .add_legend()\
   .fig.suptitle("T-SNE WITH ALL THE FEATURES OF THE FEATURE PROJECT TITLE ");
plt.show();
```



Observation:This visualisation of TSNE with Bag of Words, TF-IDF, Avg Word2Vec, TF-IDF Weighted Word2Vec does not seem to get the expected result of clustering similar data points. Hence we would have

to try any other method.

2.6 Summary

In [449]:

```
# Write few sentences about the results that you obtained and the observations yo
```

1. Female Teachers have mostly more no. of projects proposed and accepted than male teachers.
2. There are alot of projects proposed for the students between Pre Kg and 2nd standard,but it keeps decreasing as the Grades increase.
3. Students between the 9th Grade and 12th Grade have the lowest number of projects proposed as well as accepted.
4. Most of the projects have 3, 4 or 5 words in the title.
5. Visualisation with TSNE of Bow, TF-IDF, Avg Word2Vec,TF-IDF WeightedWord2Vec do not get the expected results of classifying similar data points.