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 t • `Art Wil` •
`project_grade_category`	Grade level of students for which the project is targete • • • •

Feature	
project_subject_categories	One or more (comma-separated) subject categories f following enur • • • • • Li • • • •
school_state	State where school is located (Two- (https://en.wikipedia.org/wiki/List_of_U.S._state_abbrevia
project_subject_subcategories	One or more (comma-separated) subject subcate • • Literature & Writing
project_resource_summary	An explanation of the resources needed for t • My students need hands on literacy mat 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. **Exa**
teacher_id	A unique identifier for the teacher of the propos bdf8baa8fedef6b
teacher_prefix	Teacher's title. One of the following • • • • • •
teacher_number_of_previously_posted_projects	Number of project applications previously submitted

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

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

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

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

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

from tqdm import tqdm
import os

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

1.1 Reading Data

In [2]:
```
project_data= pd.read_csv('train_new_data.csv')
resource_data= pd.read_csv('resources.csv')
```

In [3]:
```
print(project_data.shape)
print(resource_data.shape)
```

```
(109248, 17)
(1541272, 4)
```

In [4]:
```
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 [5]:
```
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']
```

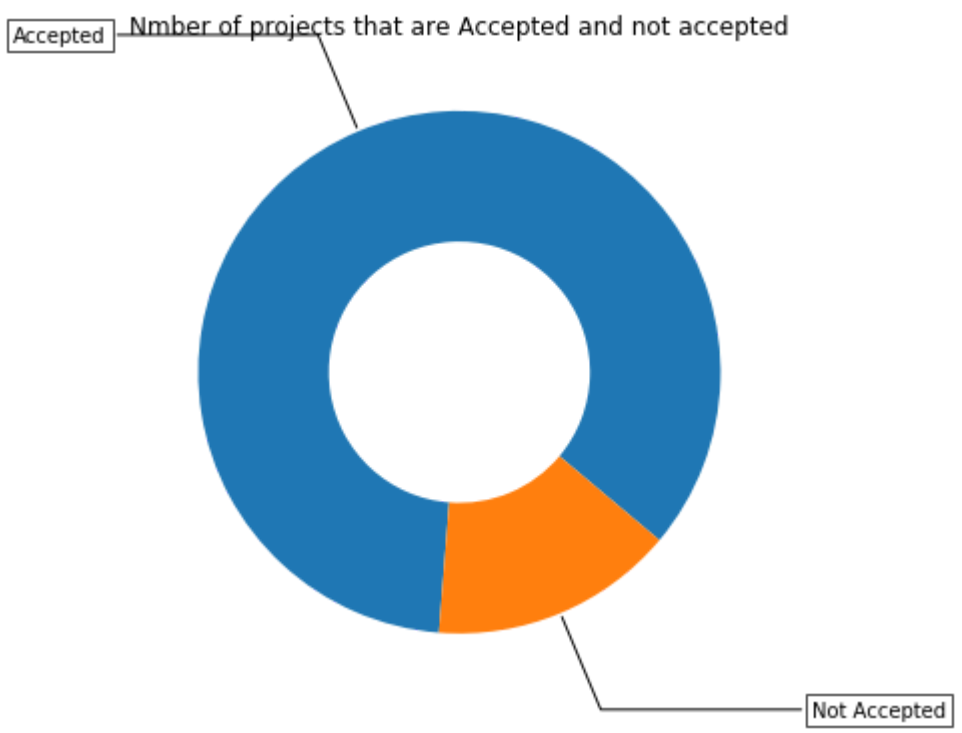
Out[5]:

	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 [6]:
```
# 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#s


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 %)
```



Observation: 1.Around 85% of projects are accepted 2.15% projects are not accepted

1.2.1 Univariate Analysis: School State

In [7]:
```
# Pandas dataframe groupby count, mean: https://stackoverflow.com/a/19385591/4084

temp = pd.DataFrame(project_data.groupby("school_state")["project_is_approved"].a
# 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(188,189,220
            [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[7]:
```
'# 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'
```

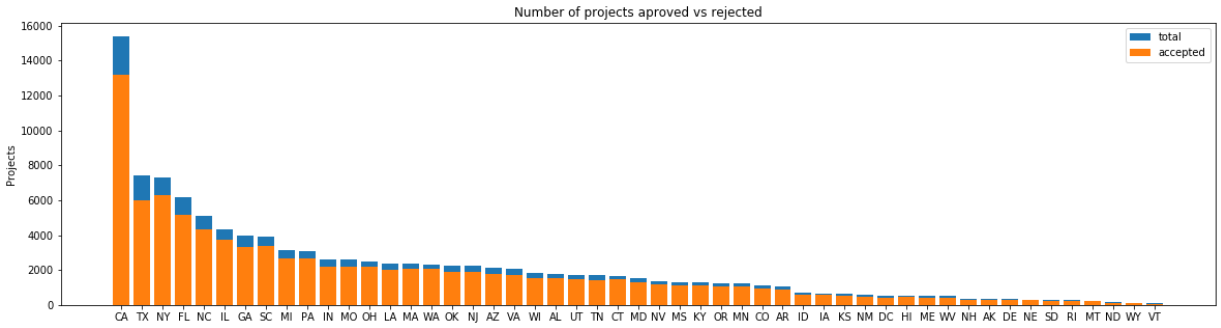
In [8]:
```
# https://www.csi.cuny.edu/sites/default/files/pdf/administration/ops/2letterstat
temp.sort_values(by=['num_proposals'], inplace=True)
print("States with lowest % approvals")
print(temp.head(5))
print('='*50)
print("States with highest % approvals")
print(temp.tail(5))
```

```
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 [9]:
```
#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 [10]:
```
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':'c
    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 [11]:
```
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

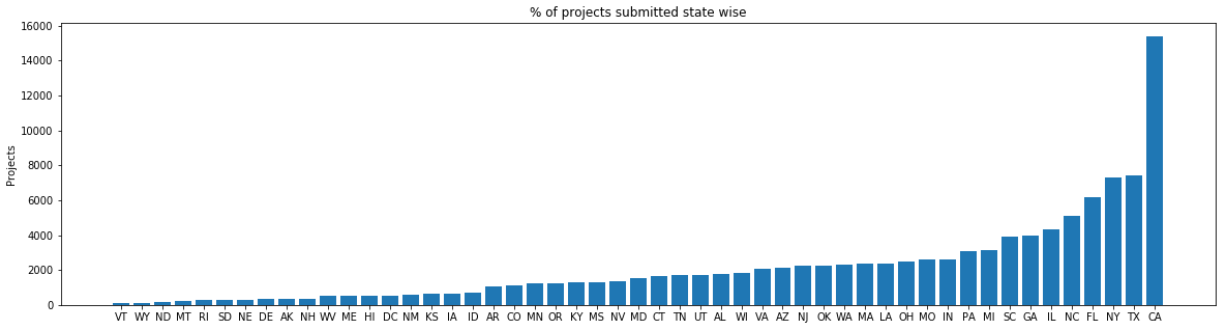
==========

	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: 1.Every state has greater than 80% success rate in approval 2.California Teachers have submitted most number of projects. 3.Vermont Teachers have least submission and approval rate.

In [12]:
```
# count of all the words in corpus python: https://stackoverflow.com/a/22898595/4
from collections import Counter
my_counter = Counter()
for word in project_data['school_state'].values:
    my_counter.update(word.split())
```

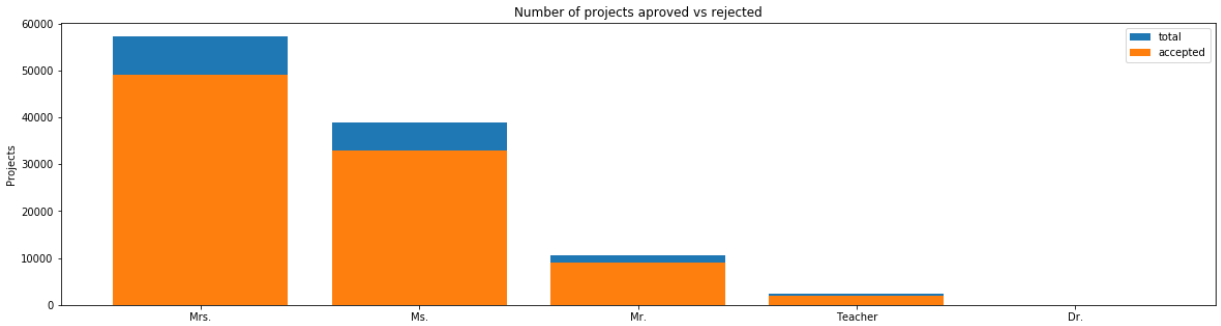
In [13]:
```
state_dict = dict(my_counter)
sorted_state_dict = dict(sorted(state_dict.items(), key=lambda kv: kv[1]))


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

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



Observation: 1.California has highest project submission rate. 2.least number of projects are submitted by vermont.

1.2.2 Univariate Analysis: teacher_prefix

In [14]:
```
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	1880	2363	0.795599
0	Dr.	9	13	0.692308

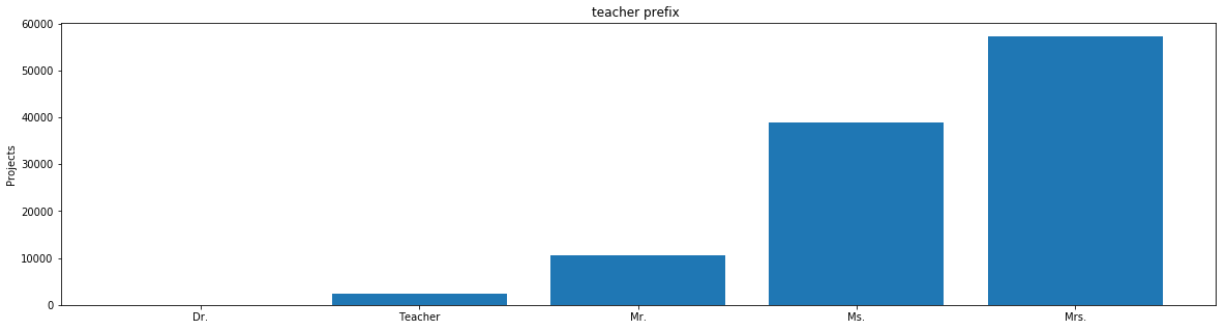
==========

	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	1880	2363	0.795599
0	Dr.	9	13	0.692308

Observation: 1.Most number of project submissions are done by female as compared to men teachers. 2.More number of project approvals are for married female teacher i.e who have a Teacher_prefix of Mrs. 3.Least number of projects are submitted by teachers who have a doctrate. 4.Majority of teachers dont have a doctrate degree.

In [15]:
```
from collections import Counter
my_counter = Counter()
for word in project_data['teacher_prefix'].values:
    my_counter.update(word.split())
```

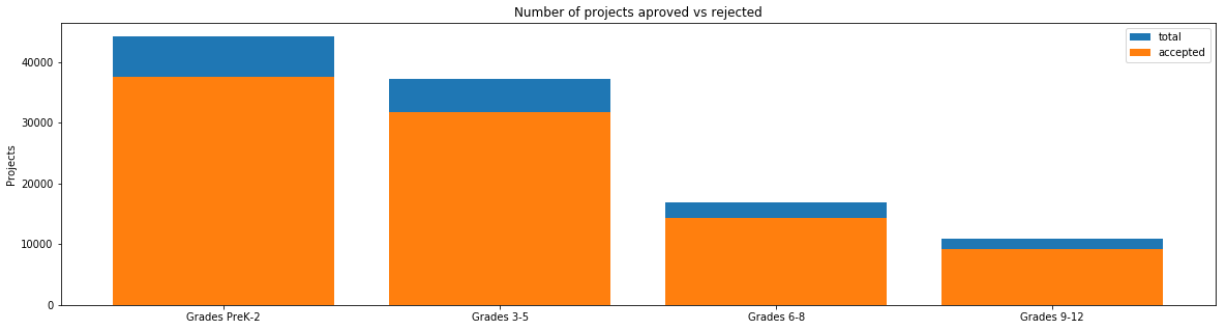
In [16]:
```
prefix_dict = dict(my_counter)
sorted_prefix_dict = dict(sorted(prefix_dict.items(), key=lambda kv: kv[1]))


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

plt.ylabel('Projects')
plt.title('teacher prefix ')
plt.xticks(ind, list(sorted_prefix_dict.keys()))
plt.show()
```



Observation: 1.Teachers with doctrates have submittedvery less projects 2.Married female teacher projects have high approval rates.

1.2.3 Univariate Analysis: project_grade_category

In [17]:
```
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
```

Observation: 1.Maximum number of projects are submitted for Grades PreK-2. 2.More number of

project approvals are for Grades 3-5. 3.Least number of projects are submitted for class 9-12.

In [18]:
```
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-
pgc_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
    pgc_list.append(temp.strip())
```

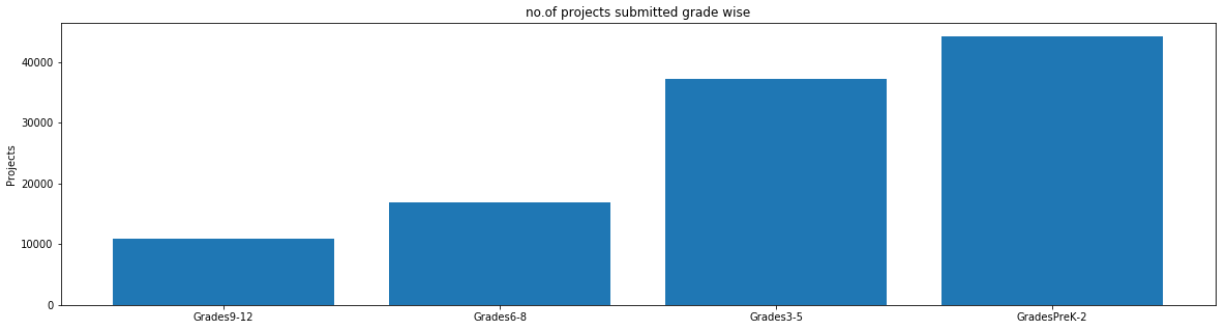
In [19]:
```
project_data['clean_project_grade_category'] = pgc_list
project_data.drop(['project_grade_category'], axis=1, inplace=True)
project_data.head(2)
```

Out[19]:

	Unnamed: 0	id	teacher_id	teacher_prefix	school_state	project_sub
0	0	p036502	484aaf11257089a66cfedc9461c6bd0a	Ms.	NV	
1	3	p185307	525fdbb6ec7f538a48beebaa0a51b24f	Mr.	NC	

In [20]:
```
from collections import Counter
my_counter = Counter()
for word in project_data['clean_project_grade_category'].values:
    my_counter.update(word.split())
```

In [21]:
```
project_grade_dict = dict(my_counter)
sorted_project_grade_dict = dict(sorted(project_grade_dict.items(), key=lambda k


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

plt.ylabel('Projects')
plt.title('no.of projects submitted grade wise')
plt.xticks(ind, list(sorted_project_grade_dict.keys()))
plt.show()
```



Observation: 1.Most of the projects are proposed for grade PreK-2. 2.least projects were proposed for Grade 9-12.

1.2.4 Univariate Analysis: project_subject_categories

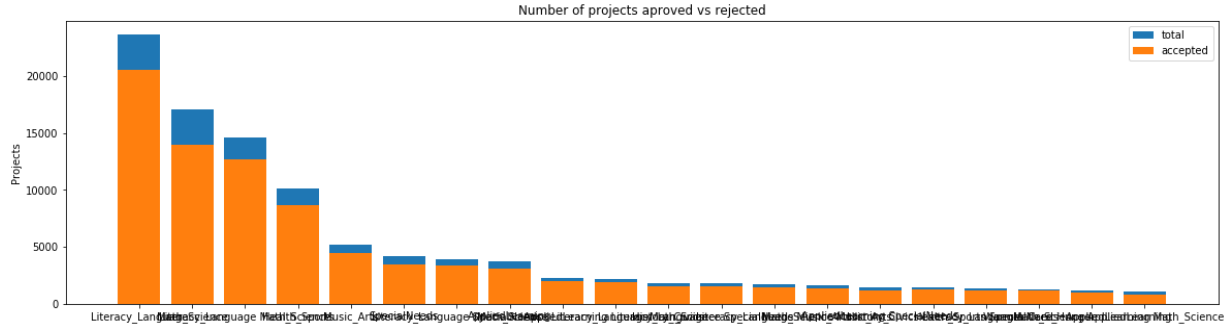
In [22]:
```
catogories = list(project_data['project_subject_categories'].values)
# remove special characters from list of strings python: https://stackoverflow.co

# 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 [23]:
```
project_data['clean_categories'] = cat_list
project_data.drop(['project_subject_categories'], axis=1, inplace=True)
project_data.head(2)
```

Out[23]:

	Unnamed: 0	id	teacher_id	teacher_prefix	school_state	project_sub
0	0	p036502	484aaf11257089a66cfedc9461c6bd0a	Ms.	NV	
1	3	p185307	525fdbb6ec7f538a48beebaa0a51b24f	Mr.	NC	

In [24]:
```
univariate_barplots(project_data, 'clean_categories', 'project_is_approved', top=
```

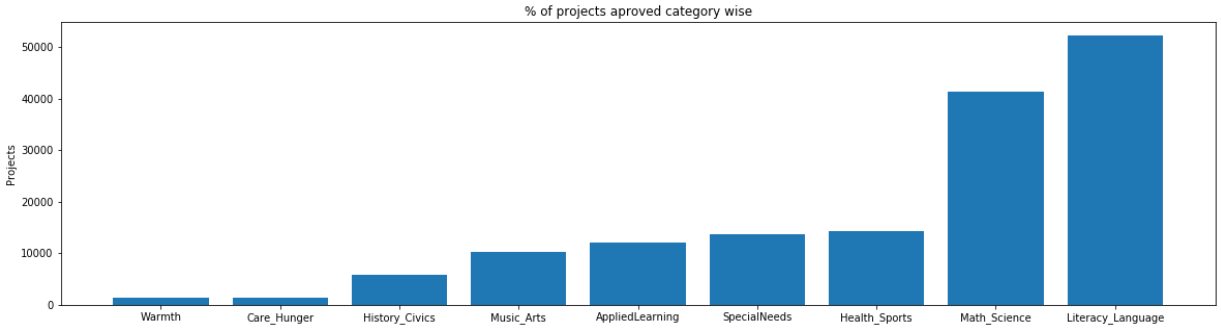



```
                clean_categories  project_is_approved  total       Avg
24              Literacy_Language                20520  23655  0.867470
32                   Math_Science                13991  17072  0.819529
28  Literacy_Language Math_Science                12725  14636  0.869432
8                   Health_Sports                 8640  10177  0.848973
40                    Music_Arts                 4429   5180  0.855019
==================================================
                   clean_categories  project_is_approved  total       Avg
19  History_Civics Literacy_Language                 1271   1421  0.894441
14        Health_Sports SpecialNeeds                 1215   1391  0.873472
50                Warmth Care_Hunger                 1212   1309  0.925898
33      Math_Science AppliedLearning                 1019   1220  0.835246
4       AppliedLearning Math_Science                  855   1052  0.812738
```

Observation: 1.Most number of projects are submitted in domain Literacy_language. 2.Most number of projects are approved in domain of warmth care_hunger. 3.Least number of projects are approved in domain of AppliedLearning Math_Science.

In [25]:
```
# count of all the words in corpus python: https://stackoverflow.com/a/22898595/4
from collections import Counter
my_counter = Counter()
for word in project_data['clean_categories'].values:
    my_counter.update(word.split())
```

In [26]:
```
# 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()
```



Observation: 1.Most number of projects are submitted in literacy language. 2.Least number of projects are submitted in area of Warmth.

In [27]:
```
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

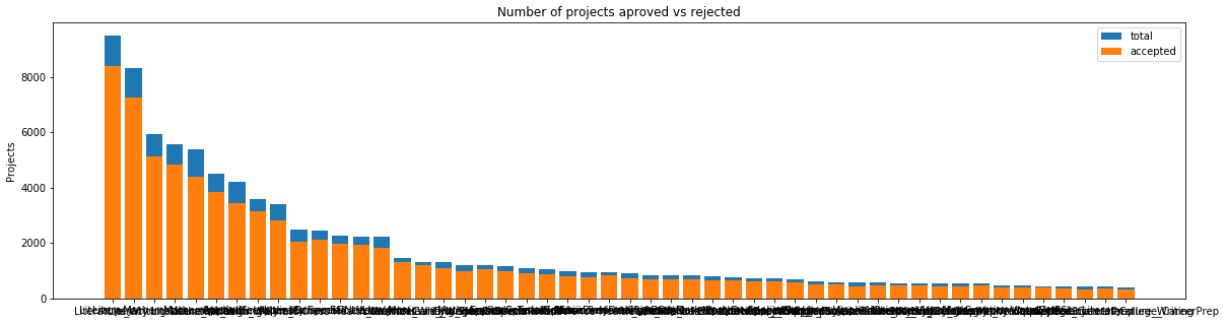
In [28]:
```
sub_catogories = list(project_data['project_subject_subcategories'].values)
# remove special characters from list of strings python: https://stackoverflow.co

# 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 [29]:
```
project_data['clean_subcategories'] = sub_cat_list
project_data.drop(['project_subject_subcategories'], axis=1, inplace=True)
project_data.head(2)
```

Out[29]:

	Unnamed: 0	id	teacher_id	teacher_prefix	school_state	project_sub
0	0	p036502	484aaf11257089a66cfedc9461c6bd0a	Ms.	NV	
1	3	p185307	525fdbb6ec7f538a48beebaa0a51b24f	Mr.	NC	

In [30]:
```
univariate_barplots(project_data, 'clean_subcategories', 'project_is_approved',
```



	clean_subcategories	project_is_approved	total	Avg
317	Literacy	8371	9486	0.882458
319	Literacy Mathematics	7260	8325	0.872072
331	Literature_Writing Mathematics	5140	5923	0.867803
318	Literacy Literature_Writing	4823	5571	0.865733
342	Mathematics	4385	5379	0.815207

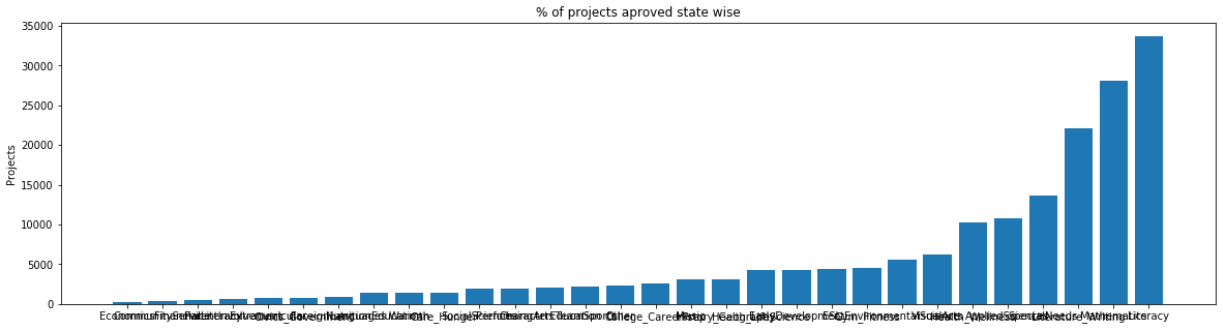
==========

	clean_subcategories	project_is_approved	total	Avg
196	EnvironmentalScience Literacy	389	444	0.876126
127	ESL	349	421	0.828979
79	College_CareerPrep	343	421	0.814727
17	AppliedSciences Literature_Writing	361	420	0.859524
3	AppliedSciences College_CareerPrep	330	405	0.814815

Observation: 1.The Literacy language has 5 subcategories in which the Literacy has the maximum number of projects and Mathematics has least number of projects submitted. 2.Least number of projects are submitted in area of AppliedSciences College_CareerPrep.

In [31]:
```
# count of all the words in corpus python: https://stackoverflow.com/a/22898595/4
from collections import Counter
my_counter = Counter()
for word in project_data['clean_subcategories'].values:
    my_counter.update(word.split())
```

In [32]:
```
# 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()
```



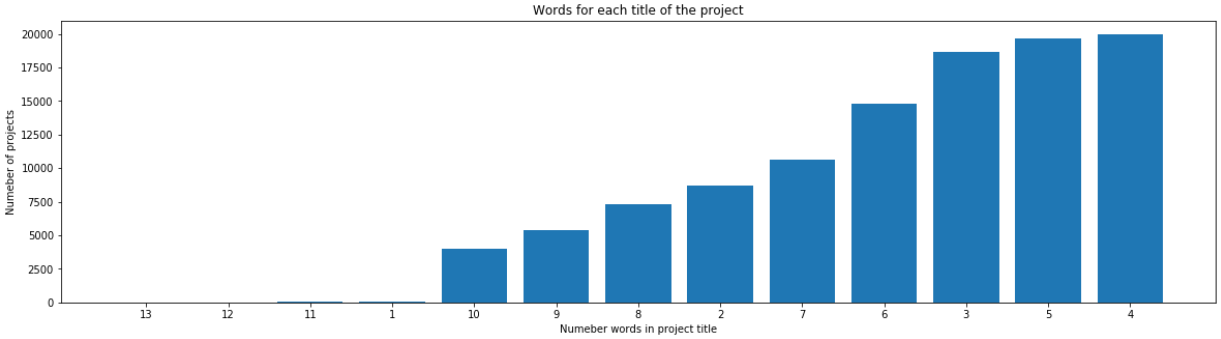
In [33]:
```
for i, j in sorted_sub_cat_dict.items():
    print("{:20} :{:10}".format(i,j))
```

```
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
```

Observation: 1.Minimum projects are in the area of economics. 2.Maximum projects are in the area of literacy.

1.2.6 Univariate Analysis: Text features (Title)

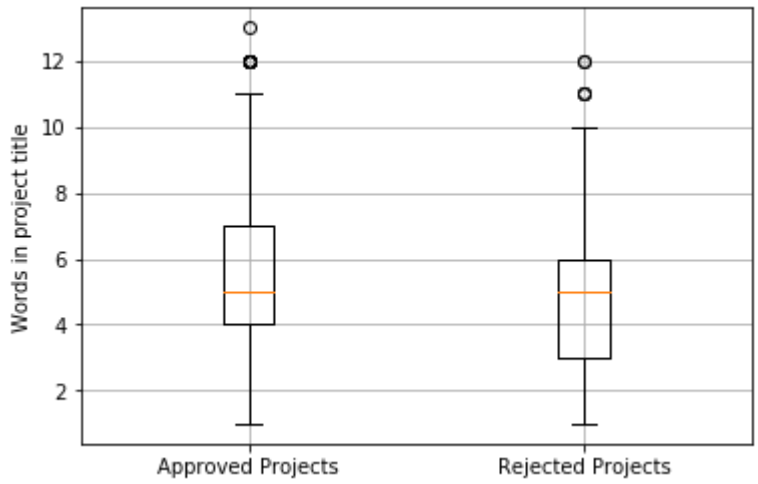
In [34]:
```
#How to calculate number of words in a string in DataFrame: https://stackoverflow
word_count = project_data['project_title'].str.split().apply(len).value_counts()
word_dict = dict(word_count)
word_dict = dict(sorted(word_dict.items(), key=lambda kv: kv[1]))


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()
```



Observation: 1.Most number of projects have 4-5 words in their project title. 2.least number of projects have 1 or 11-13 words in their project title.

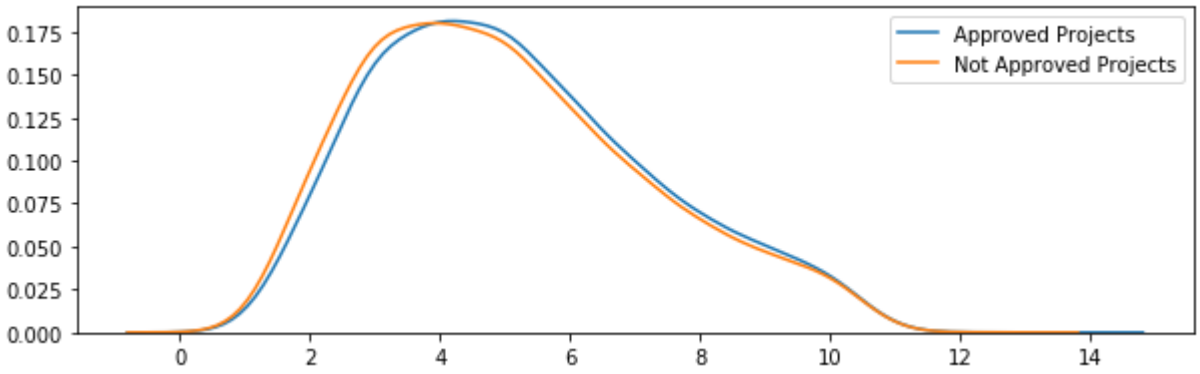
In [35]:
```
approved_title_word_count = project_data[project_data['project_is_approved']==1]
approved_title_word_count = approved_title_word_count.values

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

In [36]:
```
# 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.grid()
plt.show()
```



Observation: 1.Most of the Aprroved and Rejected projects have 4-5 words in their project titles. 2.No. of approved projects are more as compared to rejected projects 3.Approved projects have a maximum of 11 words and minimum of 1 word 4.Rejected projects have a maximum of 10 words and minimum of 1 word

In [37]:
```
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.legend()
plt.show()
```



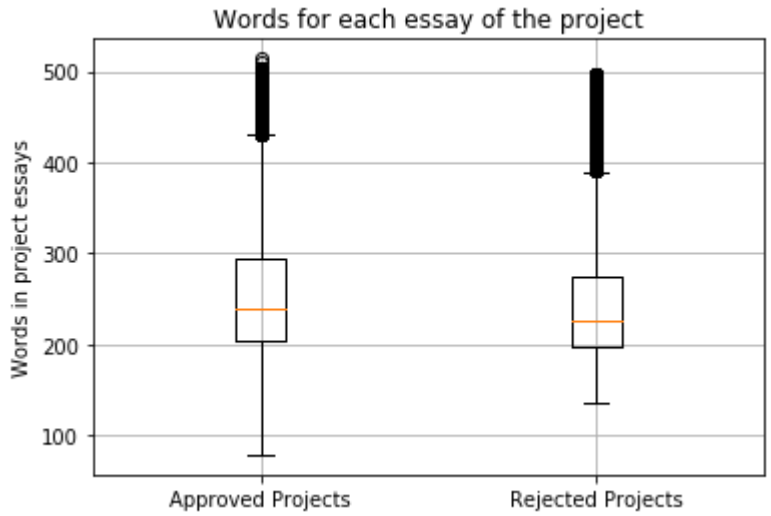
Observation: Approved projects have slightly more number of words in their project title.

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

In [38]:
```
# 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)
```

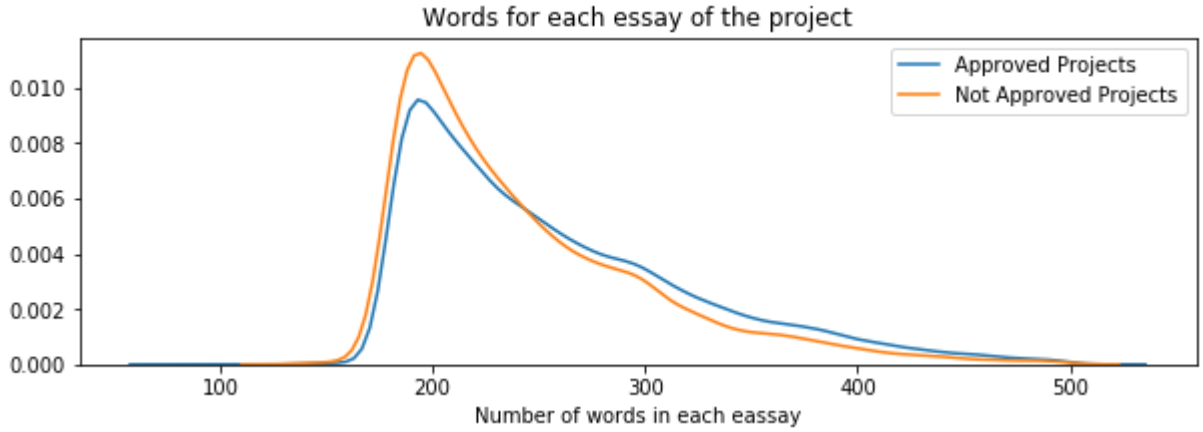
In [39]:
```
approved_word_count = project_data[project_data['project_is_approved']==1]['essay
approved_word_count = approved_word_count.values

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

In [40]:
```
# 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()
```



Observtion: 1.Approved Projects have maximum and least number of words in their project essays.
2.Rejected Projects have an average of 225 words in thir project essays.

In [41]:
```
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()
```



Observation: 1.Non Approved Projects have greater than 100 words in their project essays.
2.There are approved projects which have less than 100 words in their project essays.

1.2.8 Univariate Analysis: Cost per project

In [42]:
```
# we get the cost of the project using resource.csv file
resource_data.head(2)
```

Out[42]:

	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 [43]:
```
# 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)
```

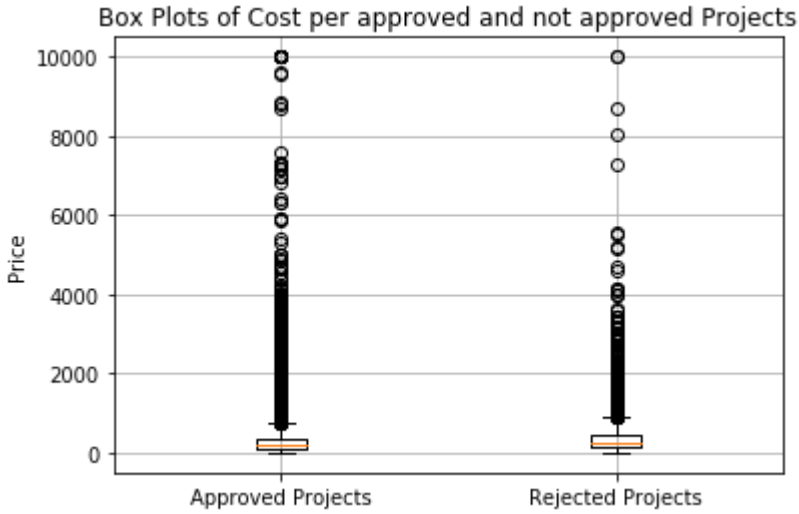
Out[43]:

	id	price	quantity
0	p000001	459.56	7
1	p000002	515.89	21

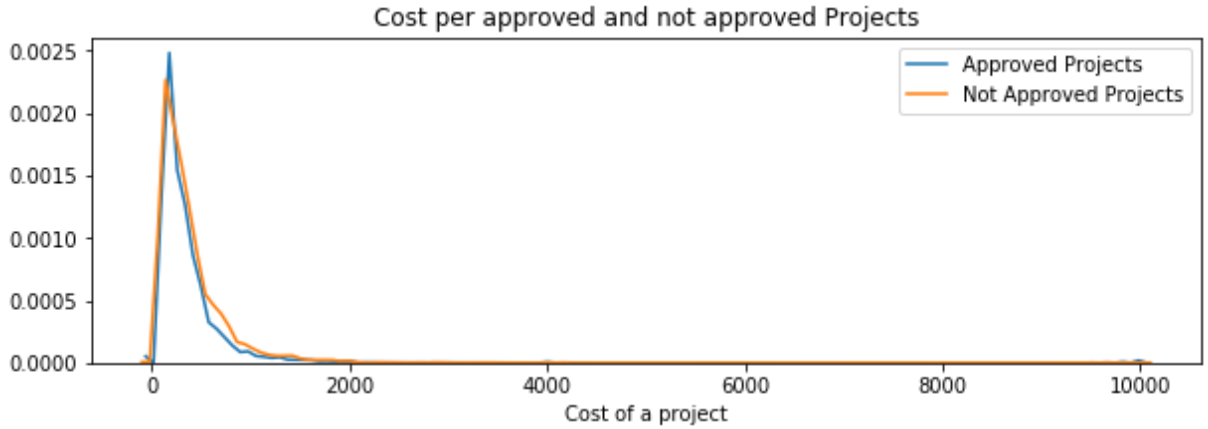
In [44]:
```
# join two dataframes in python:
project_data = pd.merge(project_data, price_data, on='id', how='left')
```

In [45]:
```
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 [46]:
```
# 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 [47]:
```
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()
```



Observation: Cost of approved projects are less as compared to non approved projects and maximum cost of a project is around 10,000 dollars.

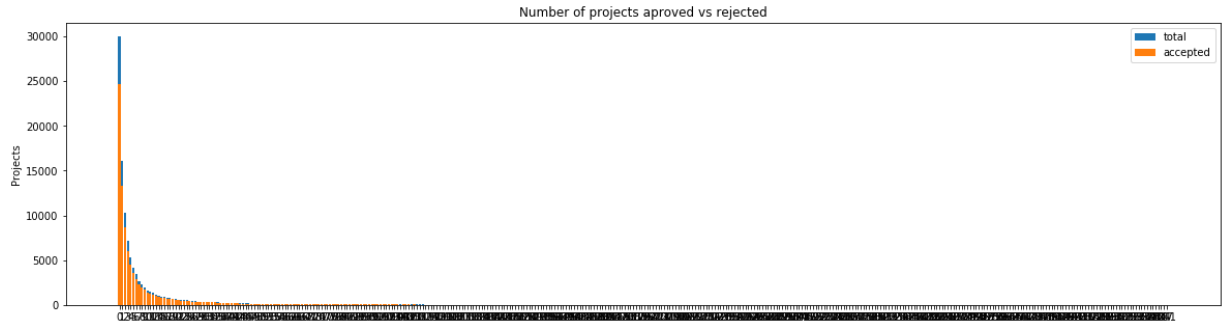
In [48]:
```
# http://zetcode.com/python/prettytable/
from prettytable import PrettyTable

#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_price,i), 3), np.round(np.percer
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 [49]:
```
univariate_barplots(project_data, 'teacher_number_of_previously_posted_projects'
```




```
   teacher_number_of_previously_posted_projects  project_is_approved  total  \
0                                             0                24652  30014
1                                             1                13329  16058
2                                             2                 8704  10349
3                                             3                 5997   7110
4                                             4                 4452   5266

        Avg
0  0.821350
1  0.830054
2  0.841047
3  0.843460
4  0.845423
==================================================
     teacher_number_of_previously_posted_projects  project_is_approved  total
\
242                                           242                   1      1
268                                           270                   1      1
234                                           234                   1      1
335                                           347                   1      1
373                                           451                   1      1

     Avg
242  1.0
268  1.0
234  1.0
335  1.0
373  1.0
```

Observation: 1.More number of participants are the teachers who didnot submit any project previously. 2.If the number of previously posted projets are high then the project acceptance rate is aslo high. 3.There is only one teacher with maximum project submission previously of 451.

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

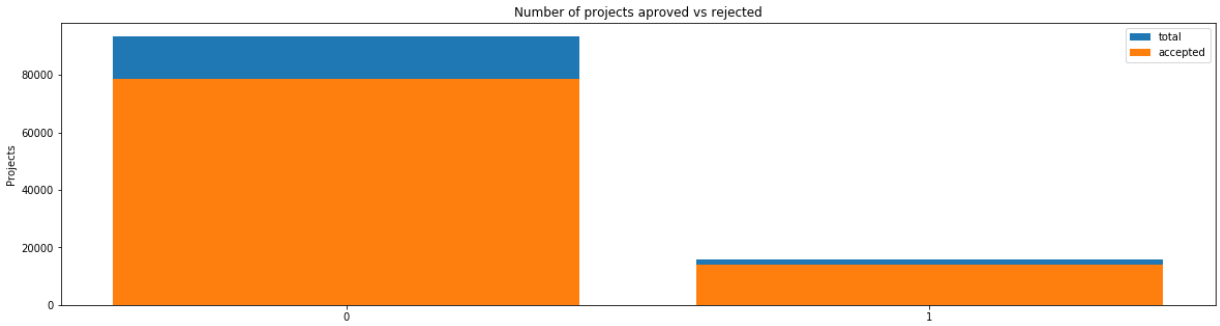
Check if the `presence of the numerical digits` in the `project_resource_summary` effects the acceptance of the project or not. If you observe that `presence of the numerical digits` is helpful in the classification, please include it for further process or you can ignore it.

In [50]:
```
def digit_in_text(x):
    k=any(i.isdigit() for i in x)
    if k==True:
        return 1
    else:
        return 0
```

In [51]:
```
text = project_data['project_resource_summary']
v = text.map(digit_in_text)
project_data['digits_in_summary'] = v
print("Number of data points in our data", project_data.shape)
project_data.head(50)
```

```
Number of data points in our data (109248, 21)
```

In [52]:
```
univariate_barplots(project_data, 'digits_in_summary', 'project_is_approved', to
```




```
   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: 1.If digits are present in project resource summary then it has an higher acceptance rate of 89.4% 2.Most of the project resource summaries dont have digits present in it.

1.3 Text preprocessing

1.3.1 Essay Text

In [53]:
```
project_data.head(2)
```

Out[53]:

	Unnamed: 0	id	teacher_id	teacher_prefix	school_state	project_sub
0	0	p036502	484aaf11257089a66cfedc9461c6bd0a	Ms.	NV	
1	3	p185307	525fdbb6ec7f538a48beebaa0a51b24f	Mr.	NC	

2 rows × 21 columns

In [54]:
```
# 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)
```

```
Most of my kindergarten students come from low-income households and are cons
idered \"at-risk\". These kids walk to school alongside their parents and mos
t have never been further than walking distance from their house. For 80% of
my students, English is not their first language or the language spoken at ho
me. \r\n\r\nWhile my kindergarten kids have many obstacles in front of them,
they come to school each day excited and ready to learn. Most students starte
d the year out never being in a school setting. At the start of the year many
had never been exposed to letters. Each day they soak up more knowledge and t
ry their hardest to succeed. They are highly motivated to learn new things ev
ery day. We are halfway through the year and they are starting to take off. T
hey know know all letters, some sight words, numbers to 20, and a majority of
their letter sounds because of their hard work and determination. I am excite
d to see the places we will go from here!I currently have a differentiated si
ght word center that we do daily during our literacy stations. The students h
ave activities that relate to whatever sight word list they are on. This is o
ne of their favorite station activities. I want to continue to provide the st
udents with engaging ways to practice their sight words. \r\n\r\nI dream of h
aving the students use QR readers to scan the sight words that they are strug
gling with and the Ipods reading the sight words with them. This would help s
o many of my students by giving them multiple exposures to the words. My stud
ents need someone who can go over these sight words daily and I can't always
get around to everyone to practice their flashcards with them. With the Ipods
they would still have a way to practice their sight words on a daily basis.na
nnan
==================================================
Our school is located the second smallest city in Los Angeles County. Our ele
mentary school is 552 students strong. We have 1 percent African American, an
d 98 percent Latinos. We have a 90percent socioeconomically disadvantaged pop
ulation and 4 percent foster youth. 100% of our students get free lunch.\r\n
\r\n   Despite the many challenges they face, my students arrive every mornin
g full of life, ready to learn, and excited to get started on our day. I do m
y best to provide my students with creative and meaningful learning experienc
es.Every morning we begin our learning by coming to the rug and setting our g
oals for the day. We come together to begin our activities and we come togeth
er to end our activities. We also come to the carpet to just have independent
reading time. The carpet area is crucial part of our learning space.\r\n\r\n
\r\n        My students are currently, sitting in a torn, stained carpet that
continues to deteriorate every day.  Some of the strings have begun to run an
d the students can no longer just sit and focus. They have began to pull and
tug at the disintegrating carpet.\r\n\r\n\r\n      This carpet will allow my
students to have a nice, clean and soft place where we can meet and learn. Th
ey need a place where they can sit, focus and not worry about their seating c
oming apart.nannan
```

==========
Our Pre-K students come from very diverse backgrounds. Many come through our doors with developmental and communication delays and learn how to engage with the world around them through play and collaborate social-emotional skills. Our students also come to us from home environments that are identified as being \"at-risk\" due to family income, languages spoken at home, and other developmental and medical situations. Though they are diverse, they all come to us with the same excitement and desire to learn.The sandbox will provide our students will excellent opportunities for the development and practice of fine motor skills, social skills, and communication. By having a place where students can sit and play closely with their peers, we can effectively teach and work on the social skills that we actively teach in the classroom. \r\nThough we have a great outdoor space, we don't have many opportunities for our students to be close and interact cooperatively outside. With this sandbox and the play materials, our kids will be able to get valuable sensory input and tactile stimulation, all while learning through play!nannan
==========
Chicago schools, like many urban school districts across America, have been fighting against the challenges of the current state of education; severe budget cuts, lack of resources, increased classroom sizes, lead in the drinking water, and many others. When basic needs in school are not being met, the power of education to transform our young people is hindered. \r\n\r\nIn a few short weeks, I am proud to be joining the team of education warriors as I will be stepping into my own classroom as a first year teacher. My new school, being both 98% African American and 75% low income, faces many challenges similar to the other schools in Chicago. \r\n\r\nI am thrilled to be working with a group of about 90 eager 4th grade readers and writers. Like every child, regardless of race or socioeconomic background, they deserve the best teachers, education, and resources. It is the job of myself, my fellow teachers and staff of my school to make sure that happens. Despite these challenges, I am dedicated to teaching the strongest culturally relevant, identity confirming, social justice curriculum that I can!\"People don't realize how a man's whole life can be changed by one book.\" -Malcolm X\r\n\r\nDo you remember reading that one book in elementary school that changed your life? There's a good chance you were able to relate to the character in the book. But what happens in schools that are predominantly African American and Latino when students only have access to reading books about white characters and animals? These books are windows into other people's lives. Many classroom libraries are missing mirrors into their student's lives. Young people, like 11 year old Marley Dias, are bringing awareness to this issue. Dias launched a list of books, calling it 1,000 Black Girl Books. \r\n\r\nBeing in a school that is 98% African American, my goal is similar. I want my students to walk into their classroom library and find more than a bin of books labeled \"Multicultural Books.\" I want my students to see reflections of themselves in every genre. I want them to see people of color in positions of power and doing amazing things in the world. I want my students to hear some of the real stories about history and important people. I truly believe that having access to these books during read alouds, mini lessons, silent reading, and to check out will foster a love for reading. This love for reading will change lives.nannan
==========
Many of our students walk into their classrooms excited and always ready to tackle their work day! The students at this K-5 school are given opportunities to grow and are always encouraged to be themselves! Our students are comprised of many different backgrounds and cultures. Our teachers and staff always make our students their number one priority. \r\n\r\nThe students at our school are unique and amazing in their own way. Every day they take on their school challenges and try their best to succeed. No matter what our teachers embra

```
ce and support the students for their efforts. Our students know they can cou
nt on us as teachers and staff and we know that we can count on them to learn
and succeed.\r\n\r\n\r\nThese students participate in our Positive Behavior S
upport (PBS) program to increase academic performance, increase safety, decre
ase problem behavior and establish positive school outcomes. PBS is a researc
hed based positive intervention system that is used to create and support pos
itive school culture by increasing positive behavior, social competence and a
cademic performance. This support system is expected to help reinforce positi
ve conduct and reduce challenging behaviors. For example, when students demon
strate positive behaviors they will earn “Tiger Bucks”. Once they earn their
bucks they will be able to use them to shop at our PBS store and they may hav
e enough to participate in our monthly socials, which students have a privile
ge of attending themed parties. \r\n\r\nPBS will help our students stay focus
ed and help them show improvement! Essentially, by purchasing items for our P
BS project, such as Lego, markers, boards, kitchen set, toy cars, and many ot
her items listed in our cart will help students decrease problem behaviors an
d improve academic performance in school. Our program will help reinforce a d
esired positive school culture in turn rewarding students to make good decisi
ons. These supplies will help us encourage our students to be the best studen
ts they can be and teach them all that good that comes with being on their be
st behavior!\r\nnannan
==================================================
```

In [55]:

```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 [56]:
```
sent = decontracted(project_data['essay'].values[20000])
print(sent)
print("="*50)
```

```
Chicago schools, like many urban school districts across America, have been fig
hting against the challenges of the current state of education; severe budget c
uts, lack of resources, increased classroom sizes, lead in the drinking water,
and many others. When basic needs in school are not being met, the power of edu
cation to transform our young people is hindered.  \r\n\r\nIn a few short week
s, I am proud to be joining the team of education warriors as I will be steppin
g into my own classroom as a first year teacher. My new school, being both 98%
African American and 75% low income, faces many challenges similar to the other
schools in Chicago. \r\n\r\nI am thrilled to be working with a group of about 9
0 eager 4th grade readers and writers. Like every child, regardless of race or
socioeconomic background, they deserve the best teachers, education, and resour
ces. It is the job of myself, my fellow teachers and staff of my school to make
sure that happens. Despite these challenges, I am dedicated to teaching the str
ongest culturally relevant, identity confirming, social justice curriculum that
I can!\"People do not realize how a man is whole life can be changed by one boo
k.\" -Malcolm X\r\n\r\nDo you remember reading that one book in elementary scho
ol that changed your life? There is a good chance you were able to relate to th
e character in the book. But what happens in schools that are predominantly Afr
ican American and Latino when students only have access to reading books about
white characters and animals? These books are windows into other people is live
s. Many classroom libraries are missing mirrors into their student is lives.  Y
oung people, like 11 year old Marley Dias, are bringing awareness to this issu
e. Dias launched a list of books, calling it 1,000 Black Girl Books. \r\n\r\nBe
ing in a school that is 98% African American, my goal is similar. I want my stu
dents to walk into their classroom library and find more than a bin of books la
beled \"Multicultural Books.\" I want my students to see reflections of themsel
ves in every genre. I want them to see people of color in positions of power an
d doing amazing things in the world. I want my students to hear some of the rea
l stories about history and important people. I truly believe that having acces
s to these books during read alouds, mini lessons, silent reading, and to check
out will foster a love for reading. This love for reading will change lives.nan
nan
==================================================
```

In [57]:
```
# \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)
```

```
Chicago schools, like many urban school districts across America, have been fig
hting against the challenges of the current state of education; severe budget c
uts, lack of resources, increased classroom sizes, lead in the drinking water,
and many others. When basic needs in school are not being met, the power of edu
cation to transform our young people is hindered.      In a few short weeks, I
am proud to be joining the team of education warriors as I will be stepping int
o my own classroom as a first year teacher. My new school, being both 98% Afric
an American and 75% low income, faces many challenges similar to the other scho
ols in Chicago.     I am thrilled to be working with a group of about 90 eager
4th grade readers and writers. Like every child, regardless of race or socioeco
nomic background, they deserve the best teachers, education, and resources. It
is the job of myself, my fellow teachers and staff of my school to make sure th
at happens. Despite these challenges, I am dedicated to teaching the strongest
culturally relevant, identity confirming, social justice curriculum that I can!
People do not realize how a man is whole life can be changed by one book.  -Mal
colm X    Do you remember reading that one book in elementary school that chang
ed your life? There is a good chance you were able to relate to the character i
n the book. But what happens in schools that are predominantly African American
and Latino when students only have access to reading books about white characte
rs and animals? These books are windows into other people is lives. Many classr
oom libraries are missing mirrors into their student is lives.  Young people, l
ike 11 year old Marley Dias, are bringing awareness to this issue. Dias launche
d a list of books, calling it 1,000 Black Girl Books.     Being in a school tha
t is 98% African American, my goal is similar. I want my students to walk into
their classroom library and find more than a bin of books labeled  Multicultura
l Books.  I want my students to see reflections of themselves in every genre. I
want them to see people of color in positions of power and doing amazing things
in the world. I want my students to hear some of the real stories about history
and important people. I truly believe that having access to these books during
read alouds, mini lessons, silent reading, and to check out will foster a love
for reading. This love for reading will change lives.nannan
```

In [58]:
```
#remove spacial character: https://stackoverflow.com/a/5843547/4084039
sent = re.sub('[^A-Za-z0-9]+', ' ', sent)
print(sent)
```

```
Chicago schools like many urban school districts across America have been fight
ing against the challenges of the current state of education severe budget cuts
lack of resources increased classroom sizes lead in the drinking water and many
others When basic needs in school are not being met the power of education to t
ransform our young people is hindered In a few short weeks I am proud to be joi
ning the team of education warriors as I will be stepping into my own classroom
as a first year teacher My new school being both 98 African American and 75 low
income faces many challenges similar to the other schools in Chicago I am thril
led to be working with a group of about 90 eager 4th grade readers and writers
Like every child regardless of race or socioeconomic background they deserve th
e best teachers education and resources It is the job of myself my fellow teach
ers and staff of my school to make sure that happens Despite these challenges I
am dedicated to teaching the strongest culturally relevant identity confirming
social justice curriculum that I can People do not realize how a man is whole l
ife can be changed by one book Malcolm X Do you remember reading that one book
in elementary school that changed your life There is a good chance you were abl
e to relate to the character in the book But what happens in schools that are p
redominantly African American and Latino when students only have access to read
ing books about white characters and animals These books are windows into other
people is lives Many classroom libraries are missing mirrors into their student
is lives Young people like 11 year old Marley Dias are bringing awareness to th
is issue Dias launched a list of books calling it 1 000 Black Girl Books Being
in a school that is 98 African American my goal is similar I want my students t
o walk into their classroom library and find more than a bin of books labeled M
ulticultural Books I want my students to see reflections of themselves in every
genre I want them to see people of color in positions of power and doing amazin
g things in the world I want my students to hear some of the real stories about
history and important people I truly believe that having access to these books
during read alouds mini lessons silent reading and to check out will foster a l
ove for reading This love for reading will change lives nannan
```

In [59]:
```
# 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've
            've', 'y', 'ain', 'aren', "aren't", 'couldn', "couldn't", 'didn', "di
            "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 [60]:
```
# 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 [01:38<00:00, 1108.68it/s]
```

In [61]:
```
# after preprocesing
preprocessed_essays[20000]
```

Out[61]:
```
'chicago schools like many urban school districts across america fighting chall
enges current state education severe budget cuts lack resources increased class
room sizes lead drinking water many others when basic needs school not met powe
r education transform young people hindered in short weeks i proud joining team
education warriors i stepping classroom first year teacher my new school 98 afr
ican american 75 low income faces many challenges similar schools chicago i thr
illed working group 90 eager 4th grade readers writers like every child regardl
ess race socioeconomic background deserve best teachers education resources it
job fellow teachers staff school make sure happens despite challenges i dedicat
ed teaching strongest culturally relevant identity confirming social justice cu
rriculum i people not realize man whole life changed one book malcolm x do reme
mber reading one book elementary school changed life there good chance able rel
ate character book but happens schools predominantly african american latino st
udents access reading books white characters animals these books windows people
lives many classroom libraries missing mirrors student lives young people like
11 year old marley dias bringing awareness issue dias launched list books calli
ng 1 000 black girl books being school 98 african american goal similar i want
students walk classroom library find bin books labeled multicultural books i wa
nt students see reflections every genre i want see people color positions power
amazing things world i want students hear real stories history important people
i truly believe access books read alouds mini lessons silent reading check fost
er love reading this love reading change lives nannan'
```

1.3.2 Project title Text

In [62]:
```
# similarly you can preprocess the titles also
from tqdm import tqdm
preprocessed_project_titles = []
# tqdm is for printing the status bar
for sentence in tqdm(project_data['project_title'].values):
    sent = decontracted(sentence)
    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_project_titles.append(sent.lower().strip())
```

```
100%|██████████| 109248/109248 [00:04<00:00, 24388.96it/s]
```

1. 4 Preparing data for models

In [63]:
```
project_data.columns
```

Out[63]:
```
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_project_grade_category', 'clean_categories',
       'clean_subcategories', 'essay', 'price', 'quantity',
       'digits_in_summary'],
      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 [64]:
```
# 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=F
vectorizer.fit(project_data['clean_categories'].values)
print(vectorizer.get_feature_names())


categories_one_hot = vectorizer.transform(project_data['clean_categories'].values
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 [65]:
```
# we use count vectorizer to convert the values into one hot encoded features
vectorizer = CountVectorizer(vocabulary=list(sorted_sub_cat_dict.keys()), lowerca
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 [66]:
```
# Please do the similar feature encoding with state, teacher_prefix and project_g

from sklearn.feature_extraction.text import CountVectorizer
vectorizer = CountVectorizer(vocabulary=list(sorted_state_dict.keys()), lowercase
vectorizer.fit(project_data['school_state'].values)
print(vectorizer.get_feature_names())


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

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

In [67]:
```
vectorizer = CountVectorizer(vocabulary=list(sorted_project_grade_dict.keys()), 
vectorizer.fit(project_data['clean_project_grade_category'].values)
print(vectorizer.get_feature_names())
project_grade_category_one_hot = vectorizer.transform(project_data['clean_projec
print("Shape of matrix after one hot encoding ",project_grade_category_one_hot.sh
```

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

In [68]:
```
vectorizer = CountVectorizer(vocabulary=list(sorted_prefix_dict.keys()), lowercas
vectorizer.fit(project_data['teacher_prefix'].values)
print(vectorizer.get_feature_names())
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.', 'Teacher', 'Mr.', 'Ms.', 'Mrs.']
Shape of matrix after one hot encodig  (109248, 5)
```

1.4.2 Vectorizing Text data

1.4.2.1 Bag of words

In [69]:
```
# 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 [70]:
```
# you can vectorize the title also
# before you vectorize the title make sure you preprocess it
vectorizer = CountVectorizer(min_df=10)
text_bow_p_t= vectorizer.fit_transform(preprocessed_project_titles)
print("Shape of matrix after one hot encodig ",text_bow_p_t.shape)
```

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

1.4.2.3 TFIDF vectorizer

In [71]:
```
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 [72]:
```
# Similarly you can vectorize for title also
vectorizer = TfidfVectorizer(min_df=10)
text_tfidf_p_t = vectorizer.fit_transform(preprocessed_project_titles)
print("Shape of matrix after one hot encodig ",text_tfidf_p_t.shape)
```

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

1.4.2.5 Using Pretrained Models: Avg W2V

In [73]:

```
# 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 preprocessed_essays:
    words.extend(i.split(' '))

for i in preprocessed_project_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)
```

```
Loading Glove Model
```

```
279727it [01:37, 2855.88it/s]

Done. 279727  words loaded!
all the words in the coupus 17014183
the unique words in the coupus 58969
The number of words that are present in both glove vectors and our coupus 44769
( 75.92 %)
word 2 vec length 44769
```

In [74]:
```
# stronging variables into pickle files python: http://www.jessicayung.com/how-to
# make sure you have the glove_vectors file
with open('glove_vectors', 'rb') as f:
    model = pickle.load(f)
    glove_words =  set(model.keys())
```

In [75]:
```
# average Word2Vec
# compute average word2vec for each review.
avg_w2v_vectors = []; # the avg-w2v for each sentence/review is stored in this li
for sentence in tqdm(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_vectors.append(vector)

print(len(avg_w2v_vectors))
print(len(avg_w2v_vectors[0]))
```

```
100%|██████████| 109248/109248 [00:48<00:00, 2075.44it/s]

109248
300
```

1.4.2.6 Using Pretrained Models: AVG W2V on `project_title`

In [76]:
```
# Similarly you can vectorize for title also
avg_w2v_vectors_p_t = []; # the avg-w2v for each sentence/review is stored in th
for sentence in tqdm(preprocessed_project_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_vectors_p_t.append(vector)

print(len(avg_w2v_vectors_p_t))
print(len(avg_w2v_vectors_p_t[0]))
```

```
100%|██████████| 109248/109248 [00:02<00:00, 40028.56it/s]

109248
300
```

1.4.2.7 Using Pretrained Models: TFIDF weighted W2V

In [77]:
```
# 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 [78]:
```
# average Word2Vec
# compute average word2vec for each review.
tfidf_w2v_vectors = []; # the avg-w2v for each sentence/review is stored in this
for sentence in tqdm(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 valu
            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_vectors.append(vector)

print(len(tfidf_w2v_vectors))
print(len(tfidf_w2v_vectors[0]))
```

```
100%|██████████| 109248/109248 [06:18<00:00, 288.41it/s]

109248
300
```

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

In [79]:
```
# Similarly you can vectorize for title also
tfidf_model_pt = TfidfVectorizer()
tfidf_model_pt.fit(preprocessed_project_titles)
# we are converting a dictionary with word as a key, and the idf as a value
dictionary = dict(zip(tfidf_model_pt.get_feature_names(), list(tfidf_model_pt.id
tfidf_words_pt = set(tfidf_model_pt.get_feature_names())
```

In [80]:
```
tfidf_w2v_vectors_pt = []; # the avg-w2v for each sentence/review is stored in t
for sentence in tqdm(preprocessed_project_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_pt):
            vec = model[word] # getting the vector for each word
            # here we are multiplying idf value(dictionary[word]) and the tf valu
            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_vectors_pt.append(vector)

print(len(tfidf_w2v_vectors_pt))
print(len(tfidf_w2v_vectors_pt[0]))
```

```
100%|██████████| 109248/109248 [00:05<00:00, 18833.18it/s]

109248
300
```

1.4.3 Vectorizing Numerical features

In [81]:
```
# 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_scala

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

```
Mean : 298.11934259666083, Standard deviation : 367.49634838483496
```

In [82]:
```
price_standardized
```

Out[82]:
```
array([[ 0.00506306],
       [ 1.05130475],
       [ 0.15613939],
       ...,
       [ 0.6823487 ],
       [-0.12157765],
       [ 0.10851987]])
```

In [83]:
```
# check this one: https://www.youtube.com/watch?v=0HOqOcln3Z4&t=530s
# standardization sklearn: https://scikit-learn.org/stable/modules/generated/skle
# 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 = StandardScale
price_scalar.fit(project_data['teacher_number_of_previously_posted_projects'].va
print(f"Mean : {price_scalar.mean_[0]}, Standard deviation : {np.sqrt(price_scala

# Now standardize the data with above maen and variance.
teacher_number_of_previously_posted_projects_standardized = price_scalar.transfo
```

```
Mean : 11.153211042765085, Standard deviation : 27.777015452500134
```

In [84]:
```
teacher_number_of_previously_posted_projects_standardized
```

Out[84]:
```
array([[ 0.53449907],
       [ 0.17448919],
       [ 1.11051488],
       ...,
       [-0.36552563],
       [-0.36552563],
       [-0.36552563]])
```

1.4.4 Merging all the above features

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

In [85]:
```
print(categories_one_hot.shape)
print(sub_categories_one_hot.shape)
print(text_bow.shape)
print(price_standardized.shape)
```

```
(109248, 9)
(109248, 30)
(109248, 16623)
(109248, 1)
```

In [86]:
```
# 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
X = hstack((categories_one_hot, sub_categories_one_hot, text_bow, price_standard
X.shape
```

Out[86]: (109248, 16663)

TSNE PLOT COMBINING ALL FEATURES AND ALL ENCODINGS OF PROJECT TITLE:

NO. OF DATAPOINTS USED-5K.

In [87]:
```
S =hstack((categories_one_hot,sub_categories_one_hot,state_one_hot,project_grade_
S.shape
```

Out[87]: (109248, 7359)

In [88]:
```
from sklearn.manifold import TSNE
S = S.tocsr()
S_new = S[0:5000,:]
```

In [89]:
```
S_new = S_new.toarray()
model = TSNE(n_components = 2, perplexity = 50, random_state = 0)
tsne_data = model.fit_transform(S_new)
```

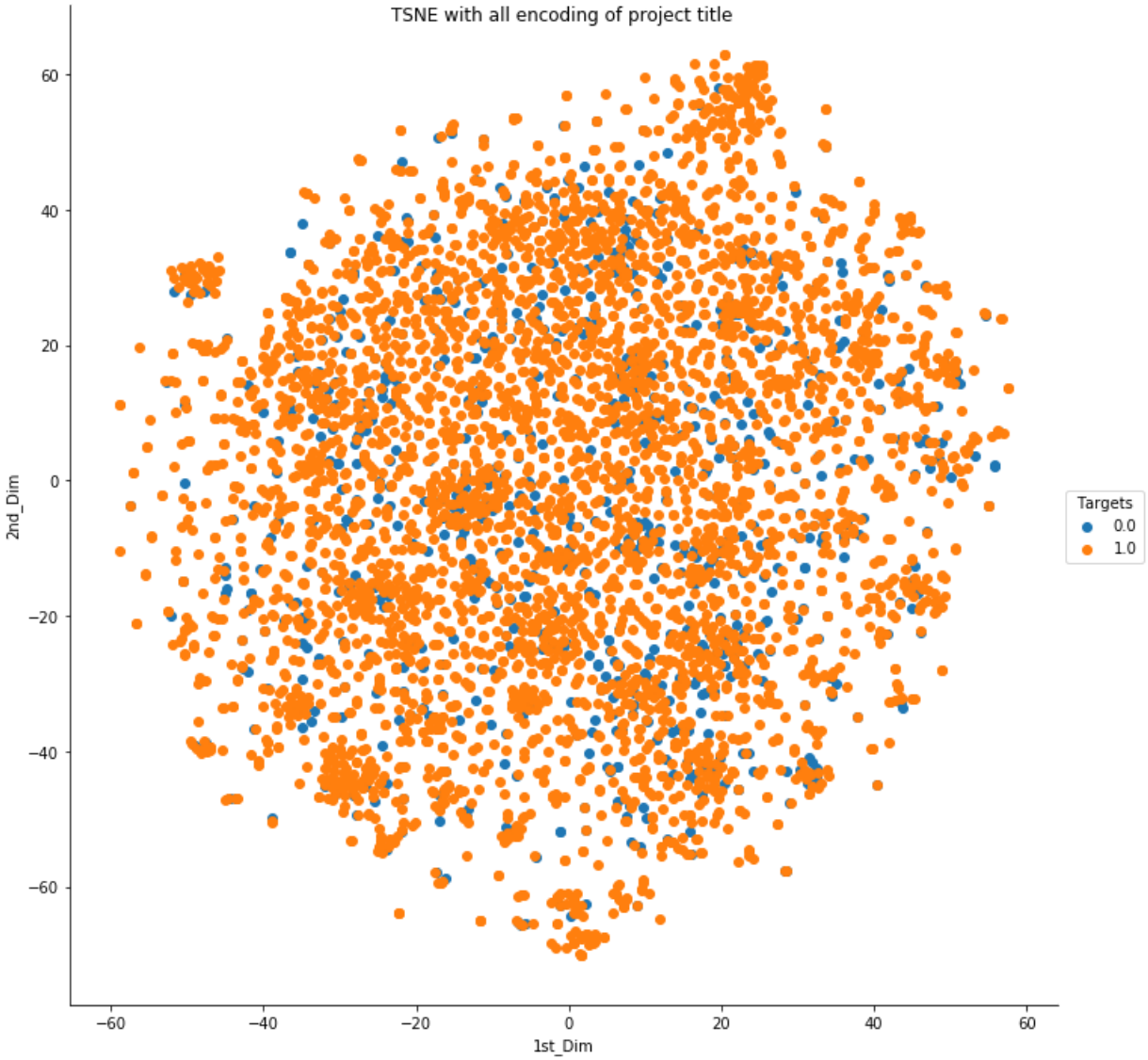
In [90]:
```
target = project_data["project_is_approved"]
target_new = target[0: 5000]
print(target_new.shape)
```

```
(5000,)
```

In [91]:
```
tsne_data = np.vstack((tsne_data.T, target_new)).T
tsne_data_frame = pd.DataFrame(tsne_data, columns = ("1st_Dim","2nd_Dim","Target
tsne_data_frame.shape
```

Out[91]: (5000, 3)

In [92]:
```
sns.FacetGrid(tsne_data_frame, hue = "Targets", size = 10).map(plt.scatter, "1st_
plt.show()
```



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.

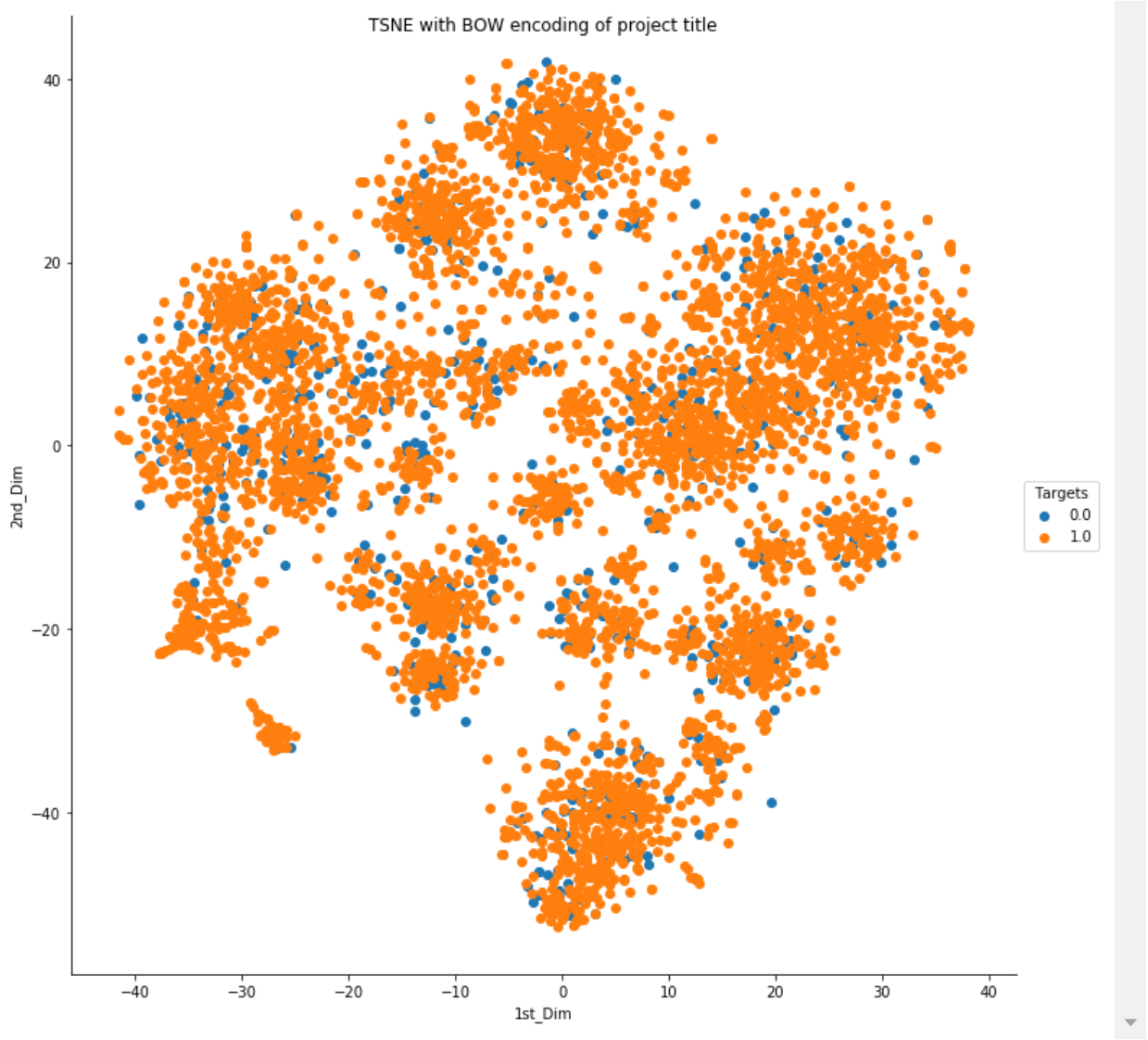
1. In the above cells we have plotted and analyzed many features. Please observe the plots and write the observations in markdown cells below every plot.
2. EDA: Please complete the analysis of the feature: teacher_number_of_previously_posted_projects
3. Build the data matrix using these features
   - 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

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

NO. OF DATAPOINTS USED-5K.

In [93]:
```
# 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
S =hstack((categories_one_hot,sub_categories_one_hot,state_one_hot,project_grade_
S.shape
S = S.tocsr()
S_new = S[0:5000,:]
S_new = S_new.toarray()
model = TSNE(n_components = 2, perplexity = 50, random_state = 0)
tsne_data = model.fit_transform(S_new)
target = project_data["project_is_approved"]
target_new = target[0: 5000]
print(target_new.shape)
tsne_data = np.vstack((tsne_data.T, target_new)).T
tsne_data_frame = pd.DataFrame(tsne_data, columns = ("1st_Dim","2nd_Dim","Targets
tsne_data_frame.shape
sns.FacetGrid(tsne_data_frame, hue = "Targets", size = 10).map(plt.scatter, "1st_
plt.show()
```

```
(5000,)
```



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

NO. OF DATAPOINTS USED-5K.

In [94]:
```
# 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
S =hstack((categories_one_hot,sub_categories_one_hot,state_one_hot,project_grade_
S.shape
S = S.tocsr()
S_new = S[0:5000,:]
S_new = S_new.toarray()
model = TSNE(n_components = 2, perplexity = 50, random_state = 0)
tsne_data = model.fit_transform(S_new)
target = project_data["project_is_approved"]
target_new = target[0: 5000]
print(target_new.shape)
tsne_data = np.vstack((tsne_data.T, target_new)).T
tsne_data_frame = pd.DataFrame(tsne_data, columns = ("1st_Dim","2nd_Dim","Targets
tsne_data_frame.shape
sns.FacetGrid(tsne_data_frame, hue = "Targets", size = 10).map(plt.scatter, "1st_
plt.show()
```

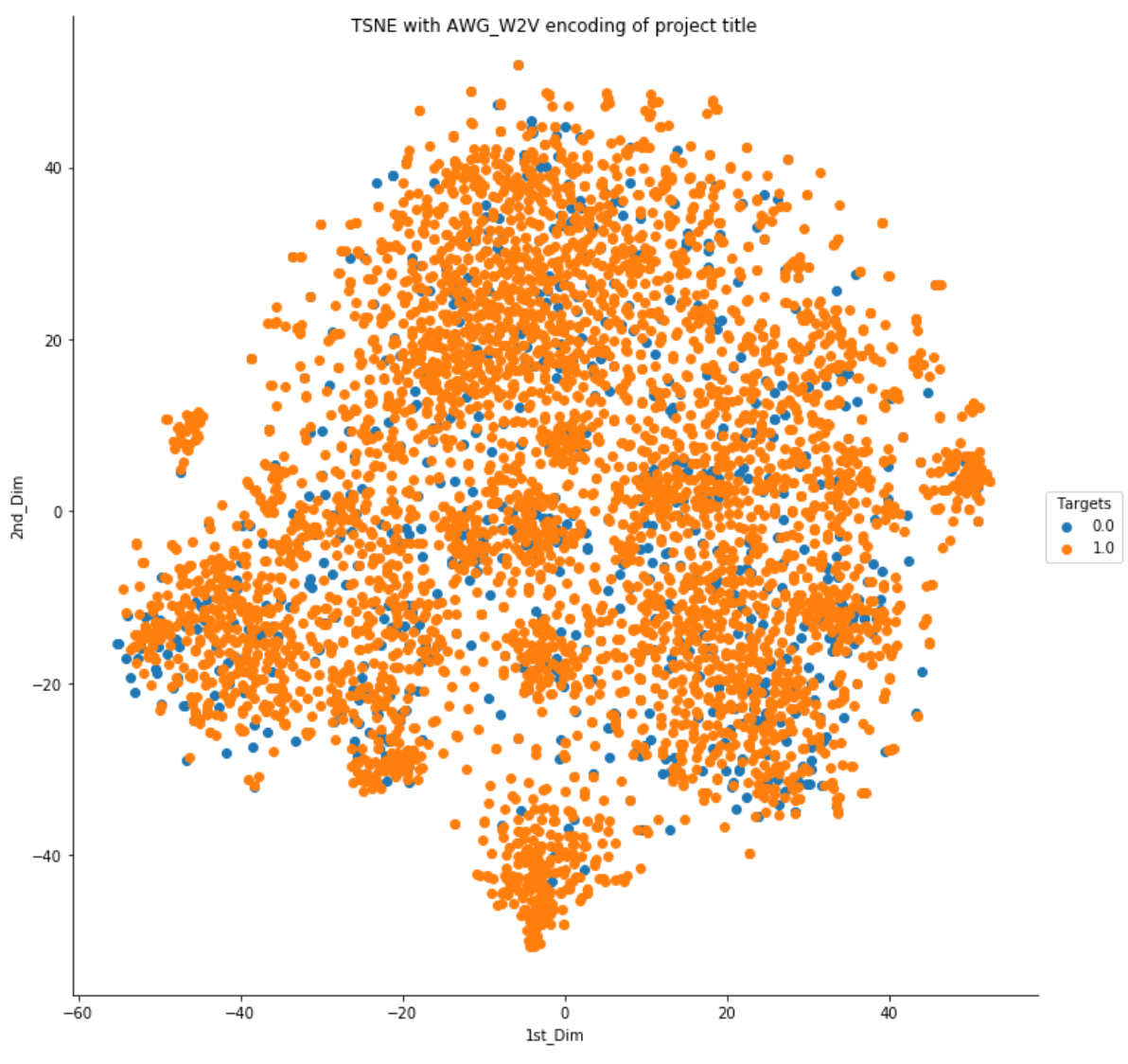
```
(5000,)
```

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

NO. OF DATAPOINTS USED-5K.

In [95]:
```
# 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
S =hstack((categories_one_hot,sub_categories_one_hot,state_one_hot,project_grade_
S.shape
S = S.tocsr()
S_new = S[0:5000,:]
S_new = S_new.toarray()
model = TSNE(n_components = 2, perplexity = 50, random_state = 0)
tsne_data = model.fit_transform(S_new)
target = project_data["project_is_approved"]
target_new = target[0: 5000]
print(target_new.shape)
tsne_data = np.vstack((tsne_data.T, target_new)).T
tsne_data_frame = pd.DataFrame(tsne_data, columns = ("1st_Dim","2nd_Dim","Targets
tsne_data_frame.shape
sns.FacetGrid(tsne_data_frame, hue = "Targets", size = 10).map(plt.scatter, "1st_
plt.show()
```

```
(5000,)
```

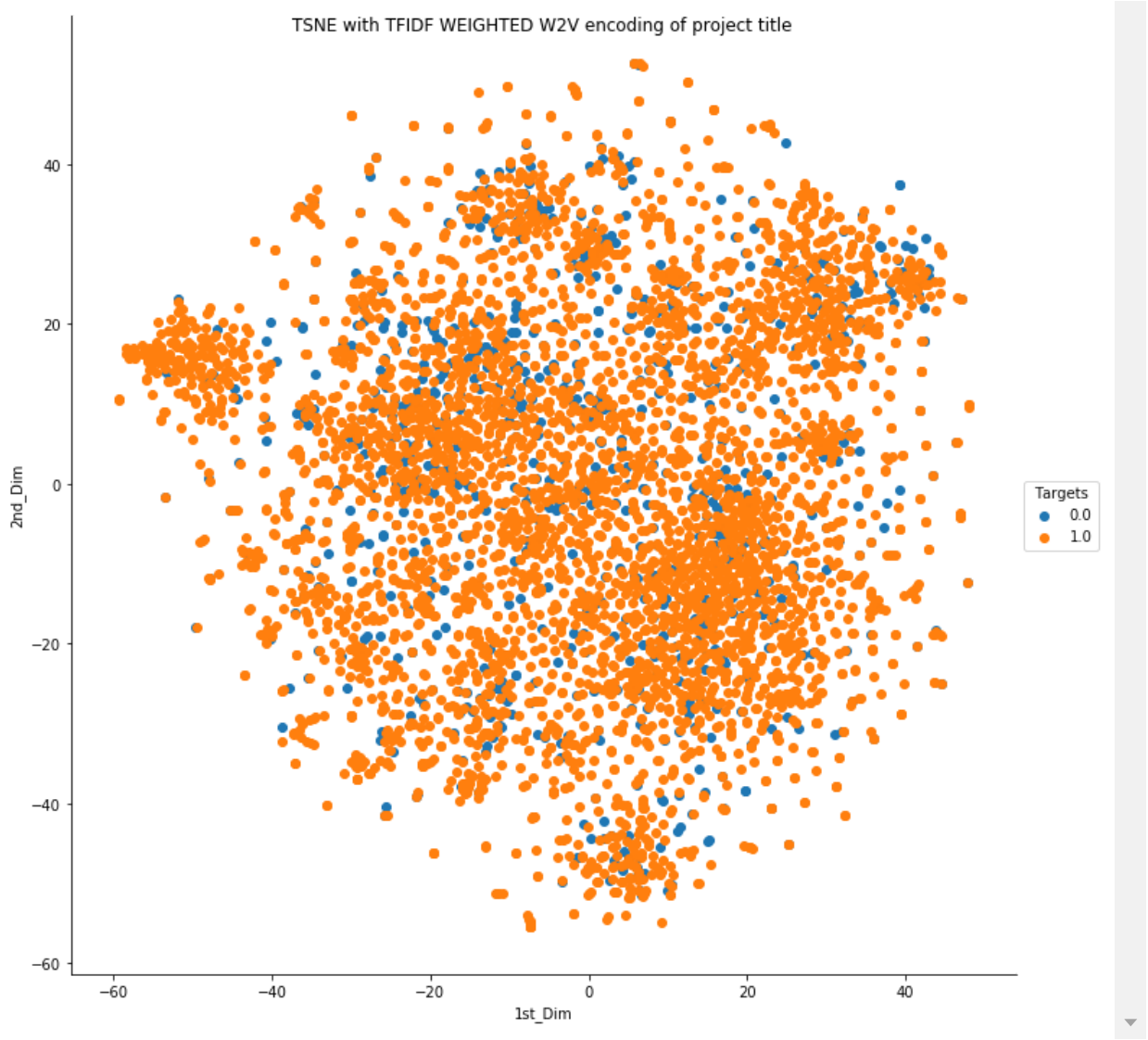


2.4 TSNE with `TFIDF Weighted W2V` encoding of `project_title` feature

NO. OF DATAPOINTS USED-5K.

In [96]:
```
# 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
S =hstack((categories_one_hot,sub_categories_one_hot,state_one_hot,project_grade_
S.shape
S = S.tocsr()
S_new = S[0:5000,:]
S_new = S_new.toarray()
model = TSNE(n_components = 2, perplexity = 50, random_state = 0)
tsne_data = model.fit_transform(S_new)
target = project_data["project_is_approved"]
target_new = target[0: 5000]
print(target_new.shape)
tsne_data = np.vstack((tsne_data.T, target_new)).T
tsne_data_frame = pd.DataFrame(tsne_data, columns = ("1st_Dim","2nd_Dim","Targets
tsne_data_frame.shape
sns.FacetGrid(tsne_data_frame, hue = "Targets", size = 10).map(plt.scatter, "1st_
plt.show()
```

```
(5000,)
```



2.5 Summary

Write few sentences about the results that you obtained and the observations you made.

All the TSNE plots have lots of overlapping of datapoints.So we are not able to make much sense out of plots as all points are well scattered.So to make sense out of data we to either increase number of datapoints or use any other method for vectorizing the text.