DonorsChoose

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

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

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

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

About the DonorsChoose Data Set

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

Feature	Description
`project_id`	A unique identifier for the proposed project. **Example:** `p036502`
`project_title`	Title of the project. **Examples:** • `Art Will Make You Happy!` • `First Grade Fun`
`project_grade_category`	Grade level of students for which the project is targeted. One of the following enumerated values: • `Grades PreK-2` • `Grades 3-5` • `Grades 6-8` • `Grades 9-12`
`project_subject_categories`	One or more (comma-separated) subject categories for the project from the following enumerated list of values: • `Applied Learning` • `Care & Hunger` • `Health & Sports` • `History & Civics`

	• Literacy & Language • Math & Science • Music & The Arts • Special Needs • Warmth **Examples:** • Music & The Arts • Literacy & Language, Math & Science
school_state	State where school is located (Two-letter U.S. postal code). **Example:** WY
project_subject_subcategories	One or more (comma-separated) subject subcategories for the project. **Examples:** • Literacy • Literature & Writing, Social Sciences
project_resource_summary	An explanation of the resources needed for the project. **Example:** • My students need hands on literacy materials to manage sensory needs!
project_essay_1	First application essay*
project_essay_2	Second application essay*
project_essay_3	Third application essay*
project_essay_4	Fourth application essay*
project_submitted_datetime	Datetime when project application was submitted. **Example:** 2016-04-28 12:43:56.245
teacher_id	A unique identifier for the teacher of the proposed project. **Example:** bdf8baa8fedef6bfeec7ae4ff1c15c56
teacher_prefix	Teacher's title. One of the following enumerated values: • nan • Dr. • Mr. • Mrs. • Ms. • Teacher.
teacher_number_of_previously_posted_projects	Number of project applications previously submitted by the same teacher. **Example:** 2

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

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

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

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

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

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

Notes on the Essay Data

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

- __project_essay_1:__ "Introduce us to your classroom"
- __project_essay_2:__ "Tell us more about your students"
- __project_essay_3:__ "Describe how your students will use the materials you're requesting"
- __project_essay_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
```

```
F:\Anaconda3\lib\site-packages\gensim\utils.py:1197: UserWarning: detected
Windows; aliasing chunkize to chunkize_serial
  warnings.warn("detected Windows; aliasing chunkize to chunkize_serial")
```

1.1 Reading Data

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

In [3]:
```
print("Number of data points in train data", project_data.shape)
print('-'*50)
print("The attributes of data :", project_data.columns.values)
```

```
Number of data points in train data (109248, 17)
--------------------------------------------------
The attributes of data : ['Unnamed: 0' 'id' 'teacher_id' 'teacher_prefix'
'school_state'
 'project_submitted_datetime' 'project_grade_category'
 'project_subject_categories' 'project_subject_subcategories'
 'project_title' 'project_essay_1' 'project_essay_2' 'project_essay_3'
 'project_essay_4' 'project_resource_summary'
 'teacher_number_of_previously_posted_projects' 'project_is_approved']
```

In [4]:
```
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[4]:

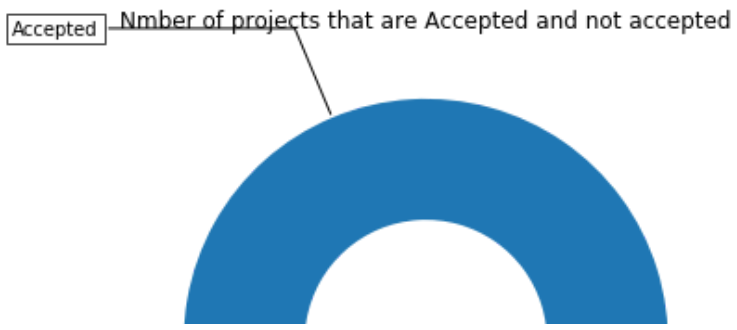
	id	description	quantity	price

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 [5]:
```
# 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#sphx-glr-gallery-pie-and-polar-charts-pie-and-donut-labels-py


y_value_counts = project_data['project_is_approved'].value_counts()
print("Number of projects thar are approved for funding ", y_value_counts[1], ", (", (y_value_counts[1]/(y_value_counts[1]+y_value_counts[0]))*100,"%)")
print("Number of projects thar are not approved for funding ", y_value_counts[0], ", (", (y_value_counts[0]/(y_value_counts[1]+y_value_counts[0]))*100, "%)")

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 , ( 15.141695957820739 %)
```

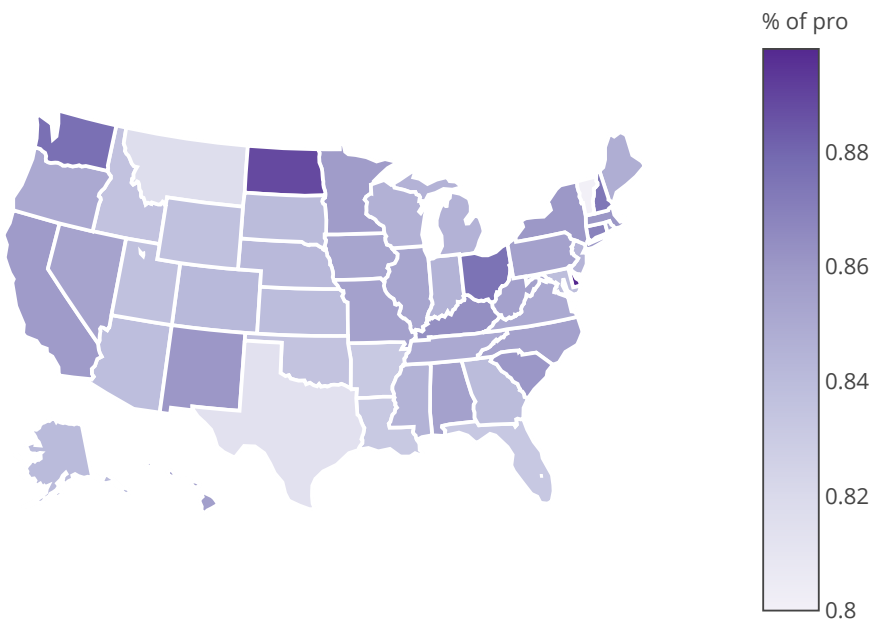


So we come to the conclusion that 84.85% of the projects are approved and the rest are unapproved

1.2.1 Univariate Analysis: School State

In [6]:
```
# Pandas dataframe groupby count, mean: https://stackoverflow.com/a/19385591/
4084039

temp = pd.DataFrame(project_data.groupby("school_state")["project_is_approve
d"].apply(np.mean)).reset_index()
# if you have data which contain only 0 and 1, then the mean = percentage (th
ink about it)
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(188,18
9,220)'],\
            [0.6, 'rgb(158,154,200)'],[0.8, 'rgb(117,107,177)'],[1.0, 'rgb(8
4,39,143)']]

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')
```

Project Proposals % of Acceptance Rate by US States



In [7]:
```
# https://www.csi.cuny.edu/sites/default/files/pdf/administration/ops/2letterstabbrev.pdf
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
```

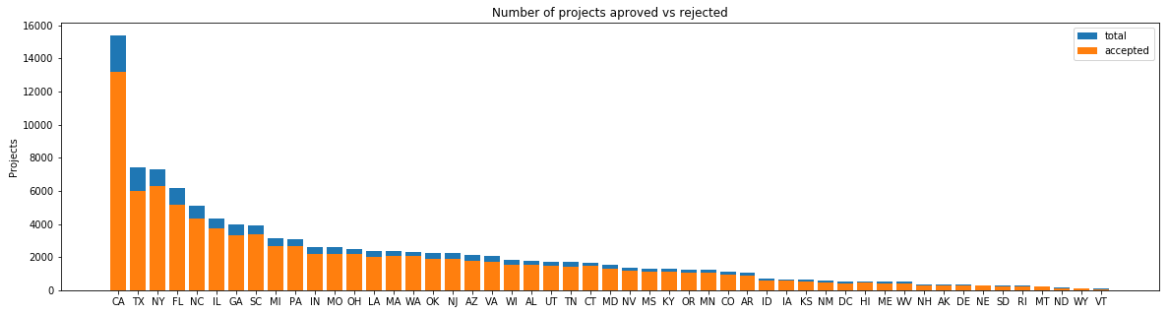
1) 'DE' state has the maximum approval rate of projects followed by 'ND' 2) 'VT' state has the lowest approval rate of projects followed by 'DC'

In [8]:
```
#stacked bar plots matplotlib: https://matplotlib.org/gallery/lines_bars_and_markers/bar_stacked.html
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 [9]:
```
def univariate_barplots(data, col1, col2='project_is_approved', top=False):
    # Count number of zeros in dataframe python: https://stackoverflow.com/a/51540521/4084039
    temp = pd.DataFrame(project_data.groupby(col1)[col2].agg(lambda x: x.eq(1).sum())).reset_index()

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

    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 [10]:
```
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.

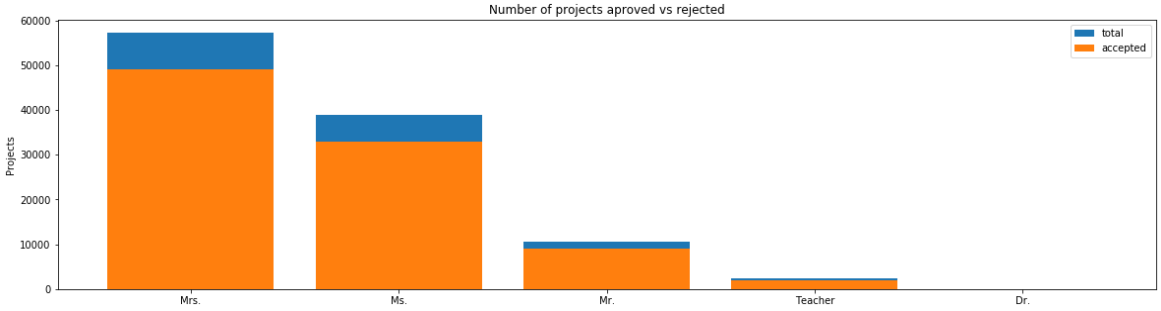
2. There is a lot of variability in the number of projects that have been submitted across the states.

3. CA has the highest number of project proposals when compared to the other states. Surprisingly, 85% of the projects gets approved on an average.

4. VT has the lowest number of project proposals submitted and almost 80% of the project proposal gets accepted.

1.2.2 Univariate Analysis: teacher_prefix

In [11]:
```
univariate_barplots(project_data, 'teacher_prefix', 'project_is_approved' , top=False)
```




```
  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
```

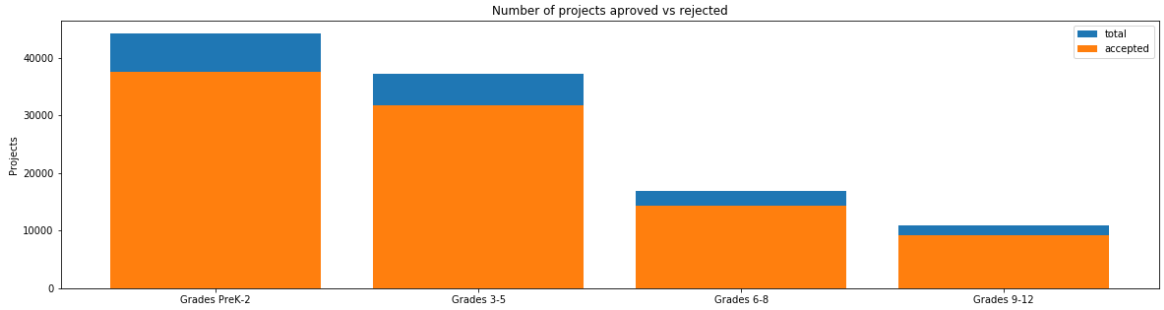
SUMMARY

1. Women teachers have the maximum number of projects proposed and accepted compared to the male teachers.

2. Teachers with prefixes Mrs. , which means Married Women as teachers have a higher number of projects Proposed as well as Accepted when compared to the younger Unmarried Women Teachers and other teachers too.

1.2.3 Univariate Analysis: project_grade_category

In [12]:
```
univariate_barplots(project_data, 'project_grade_category', 'project_is_approved', top=False)
```




```
  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
```

SUMMARY

1. There are alot of projects proposed for the students between Pre Kindergarden and 2nd Grade while for the rest it keeps decreasing.

2. The average Acceptance rate of the project in PerK-2 is 84.8% and for Grades 3-5 is 85.4% and for 6-8 and 9-12 it is 84.2% and 83.76% respectively

3. We also notice that Students between the 9th Grade and 12th Grade have the lowest number of projects proposed as well as accepted and the students between greades 3-5 has the maximum acceptance rate

1.2.4 Univariate Analysis: project_subject_categories

In [13]:

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

# https://www.geeksforgeeks.org/removing-stop-words-nltk-python/
# https://stackoverflow.com/questions/23669024/how-to-strip-a-specific-word-from-a-string
# https://stackoverflow.com/questions/8270092/remove-all-whitespace-in-a-string-in-python
cat_list = []
for i in catogories:
    temp = ""
    # consider we have text like this "Math & Science, Warmth, Care & Hunger"
    for j in i.split(','): # it will split it in three parts ["Math & Science", "Warmth", "Care & Hunger"]
        if 'The' in j.split(): # this will split each of the catogory based on space "Math & Science"=> "Math","&", "Science"
            j=j.replace('The','') # if we have the words "The" we are going to replace it with ''(i.e removing 'The')
        j = j.replace(' ','') # we are placeing all the ' '(space) with ''(empty) ex:"Math & Science"=>"Math&Science"
        temp+=j.strip()+" " #" abc ".strip() will return "abc", remove the trailing spaces
```

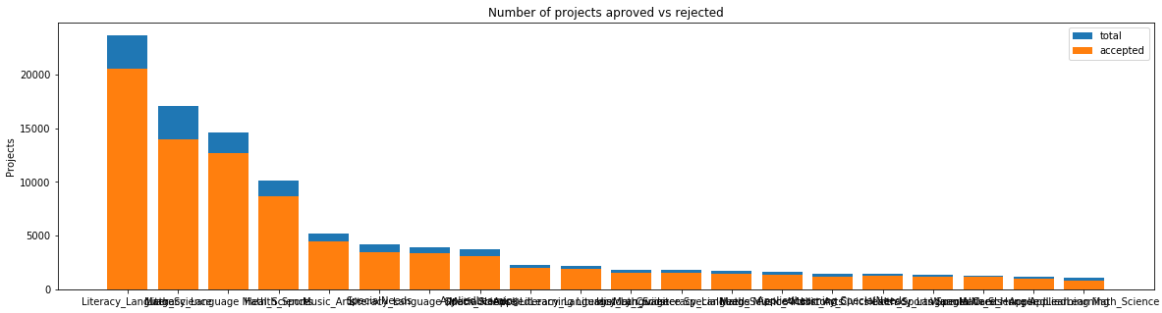
```
        temp = temp.replace('&','_') # we are replacing the & value into
    cat_list.append(temp.strip())
```

In [14]:
```
project_data['clean_categories'] = cat_list
project_data.drop(['project_subject_categories'], axis=1, inplace=True)
project_data.head(2)
```

Out[14]:

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

In [15]:
```
univariate_barplots(project_data, 'clean_categories', 'project_is_approved',
top=20)
```




```
                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
```

SUMMARY

_1.Projects belonging to the Literacy and Language categories have the highest number of projects proposed under. The maximum number of accepted projects also belong to HISTORY_CIVICS LITERACY_LANGUAGE category, having an acceptance rate of nearly 89.44% and Literacy*Language alone has the highest 86.7%*

2.Projects belonging to both Maths and Science have acceptance rate of nearly 82% while introducing the concept of Literacy and Language to this can increase its accpetance rate to nearly 87%

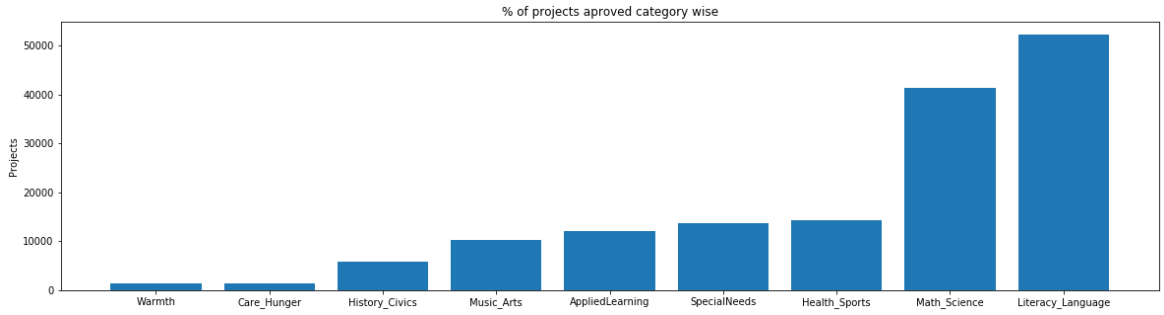
3.There is a lot of variablity in the total number of projects proposed per Category of the project.

4.Projects belonging to both Maths and Science Applied Learning has the least number of projects proposed as well approved.

5.There is also Variability in Acceptance rate, projects under the category Warmth, Care and Hunger have an acceptance rate of 93.5%

In [16]:
```
# count of all the words in corpus python: https://stackoverflow.com/a/22898595/4084039
from collections import Counter
my_counter = Counter()
for word in project_data['clean_categories'].values:
    my_counter.update(word.split())
```

In [17]:
```
# 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()
```



In [18]:
```
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
```

SUMMARY

1. The highest number of projects are registered under Literacy and Language with 52,239 projects, followed by Maths and Science having 41421 projects.

_2. There are only 1388 projects under the category of Warmth & Care*Hunger.*

1.2.5 Univariate Analysis: project_subject_subcategories

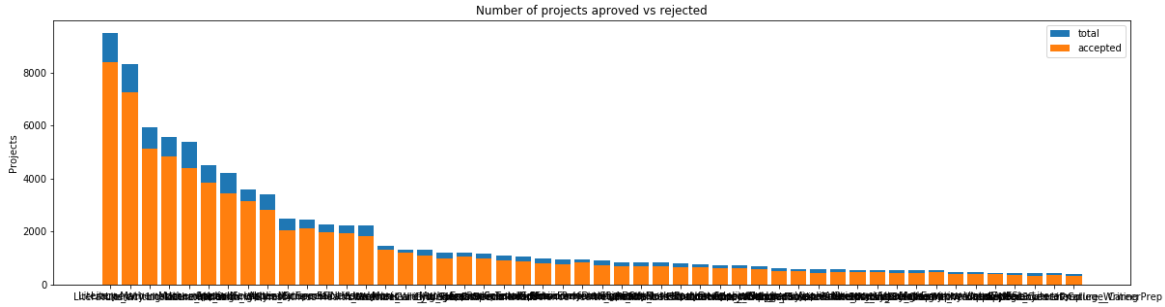
In [19]:
```
sub_catogories = list(project_data['project_subject_subcategories'].values)
# remove special characters from list of strings python: https://stackoverflow.com/a/47301924/4084039

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

sub_cat_list = []
for i in sub_catogories:
    temp = ""
    # consider we have text like this "Math & Science, Warmth, Care & Hunger"
    for j in i.split(','): # it will split it in three parts ["Math & Science", "Warmth", "Care & Hunger"]
        if 'The' in j.split(): # this will split each of the catogory based on space "Math & Science"=> "Math","&", "Science"
            j=j.replace('The','') # if we have the words "The" we are going to replace it with ''(i.e removing 'The')
        j = j.replace(' ','') # we are placeing all the ' '(space) with ''(empty) ex:"Math & Science"=>"Math&Science"
        temp +=j.strip()+" "#" abc ".strip() will return "abc", remove the trailing spaces
        temp = temp.replace('&','_')
    sub_cat_list.append(temp.strip())
```

In [20]:
```
project_data['clean_subcategories'] = sub_cat_list
project_data.drop(['project_subject_subcategories'], axis=1, inplace=True)
project_data.head(2)
```

Out[20]:

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

In [21]:
```
univariate_barplots(project_data, 'clean_subcategories', 'project_is_approved', top=50)
```



	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

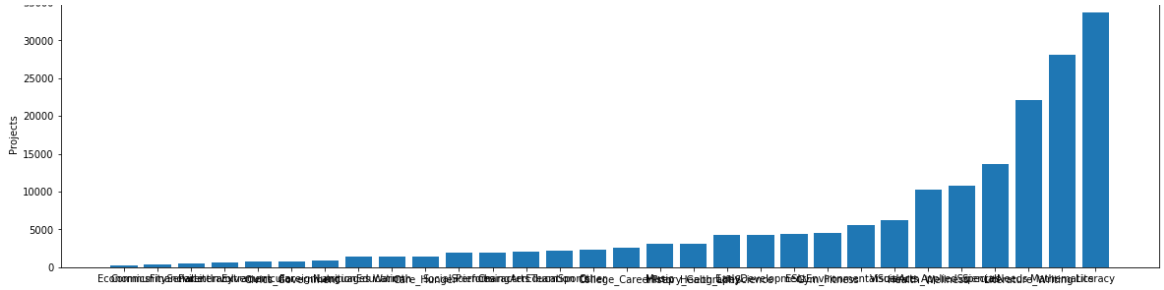
SUMMARY :

1.Literacy has the highest number of projects approved with 8371 projects AND the acceptance rate is 88%.

2.Health and Wellness have the lowest number of projects proposed with 3,583 projects only.

In [22]:
```
# count of all the words in corpus python: https://stackoverflow.com/a/22898595/4084039
from collections import Counter
my_counter = Counter()
for word in project_data['clean_subcategories'].values:
    my_counter.update(word.split())
```

In [23]:
```
# 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()
```



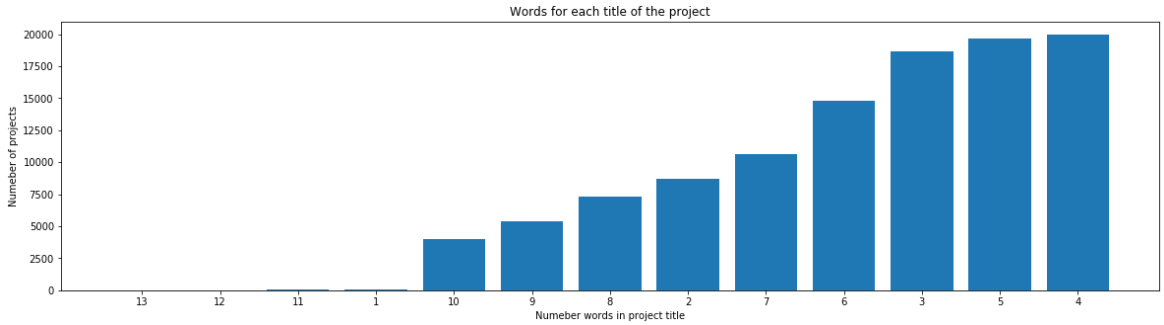
In [24]:
```
for i, j in sorted_sub_cat_dict.items():
    print("{:20} :{:10}".format(i,j))
```

```
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 [25]:
```
#How to calculate number of words in a string in DataFrame: https://stackover
flow.com/a/37483537/4084039
word_count = project_data['project_title'].str.split().apply(len).value_count
s()
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()
```



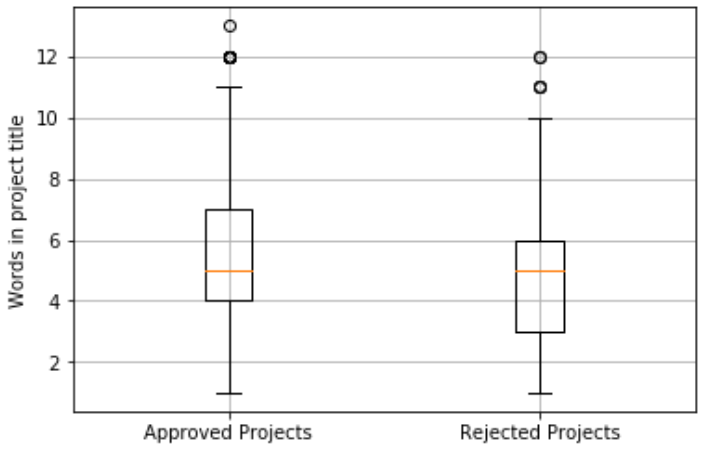
SUMMARY :

1. Most of the projects have 4 words in the title followed by 3,5,6 words.

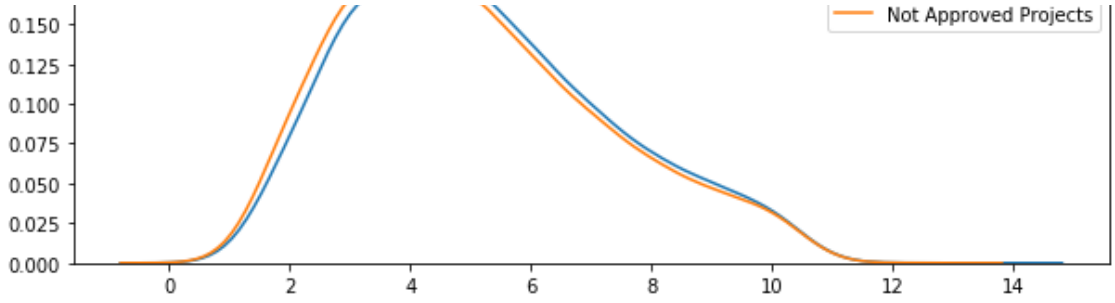
2. There are hardly any project titles containing greater than 10 words.

In [26]:
```
approved_title_word_count = project_data[project_data['project_is_approved']==1]['project_title'].str.split().apply(len)
approved_title_word_count = approved_title_word_count.values

rejected_title_word_count = project_data[project_data['project_is_approved']==0]['project_title'].str.split().apply(len)
rejected_title_word_count = rejected_title_word_count.values
```

In [27]:
```
# 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()
```



In [28]:
```
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()
```



SUMMARY

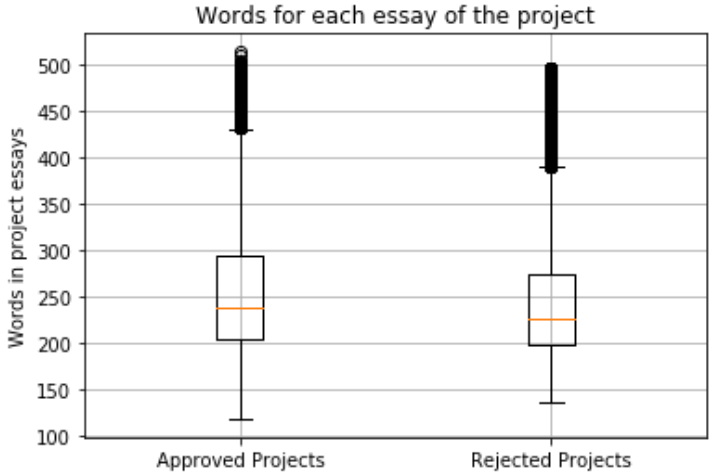
1) It has been observed slight more no of words increases the chance of being accepted.

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

In [29]:
```
# 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 [30]:
```
approved_word_count = project_data[project_data['project_is_approved']==1]['essay'].str.split().apply(len)
approved_word_count = approved_word_count.values

rejected_word_count = project_data[project_data['project_is_approved']==0]['essay'].str.split().apply(len)
rejected_word_count = rejected_word_count.values
```

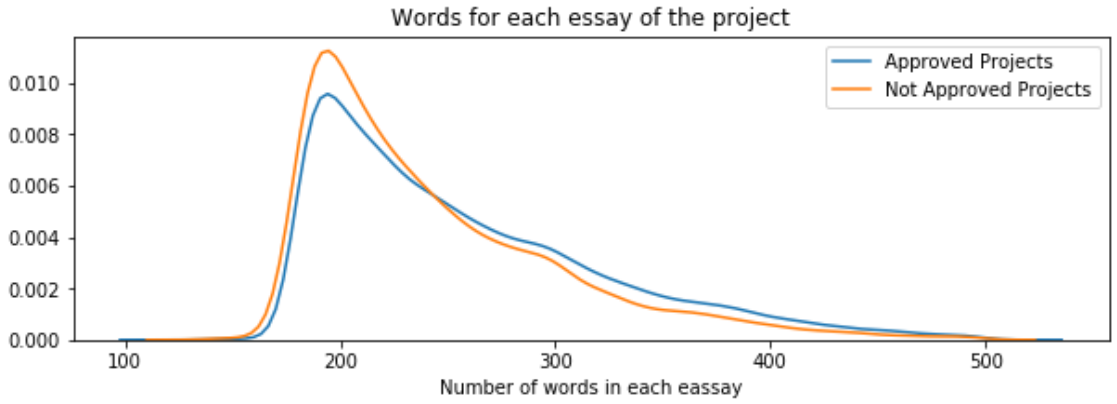
In [31]:
```
# https://glowingpython.blogspot.com/2012/09/boxplot-with-matplotlib.html
plt.boxplot([approved_word_count, rejected_word_count])
plt.title('Words for each essay of the project')
plt.xticks([1,2],('Approved Projects','Rejected Projects'))
plt.ylabel('Words in project essays')
plt.grid()
plt.show()
```



SUMMARY

Approved projects have a slightly more number of words in the project essays when compared to the projects that have not been approved.

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



SUMMARY:

The number of words in the Project Essays of Approved Projects are slightly more than the number of words in the Project Essays of the Rejected Projects. This can be noticed by looking at the Blue Line which is denser for words more than 250 to 500.

1.2.8 Univariate Analysis: Cost per project

In [33]:
```
# we get the cost of the project using resource.csv file
resource_data.head(2)
```

Out[33]:

	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 [34]:
```
# https://stackoverflow.com/questions/22407798/how-to-reset-a-dataframes-indexes-for-all-groups-in-one-step
price_data = resource_data.groupby('id').agg({'price':'sum', 'quantity':'sum'}).reset_index()
price_data.head(2)
```

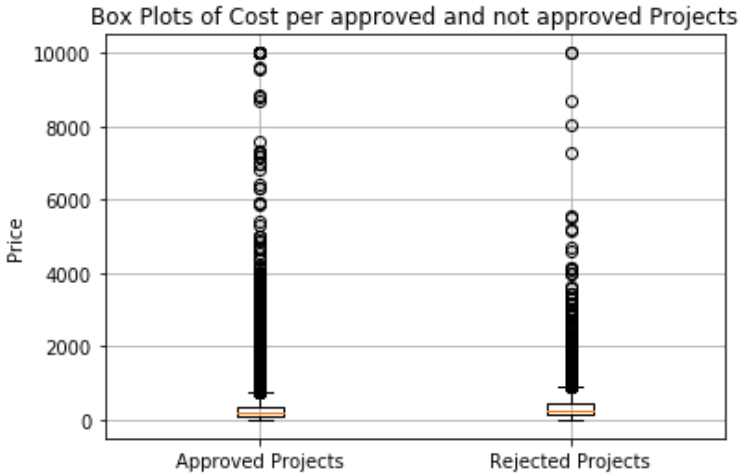
Out[34]:

	id	price	quantity
0	p000001	459.56	7
1	p000002	515.89	21

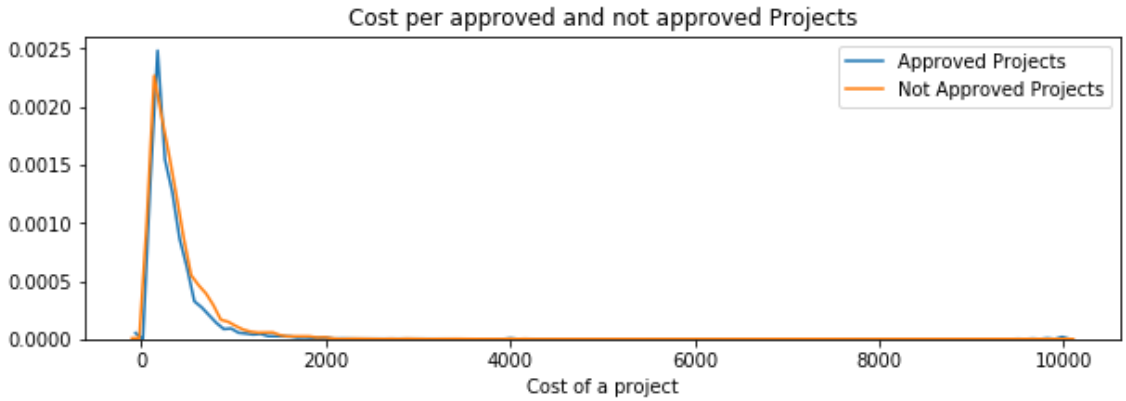
In [35]:
```
# join two dataframes in python:
project_data = pd.merge(project_data, price_data, on='id', how='left')
```

In [36]:
```
approved_price = project_data[project_data['project_is_approved']==1]['price'
].values

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

In [37]:
```
# 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 [38]:
```
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()
```



SUMMARY:

Nothing can be inferred from the box plot depicting the Cost involved per project. We can infer from the PDF curves that mostly Projects that are very costly are usually unapproved.

In [39]:
```
# http://zetcode.com/python/prettytable/
```

```
from prettytable import PrettyTable

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

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.pe
rcentile(rejected_price,i), 3)])
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        |
+------------+------------------+-----------------------+
```

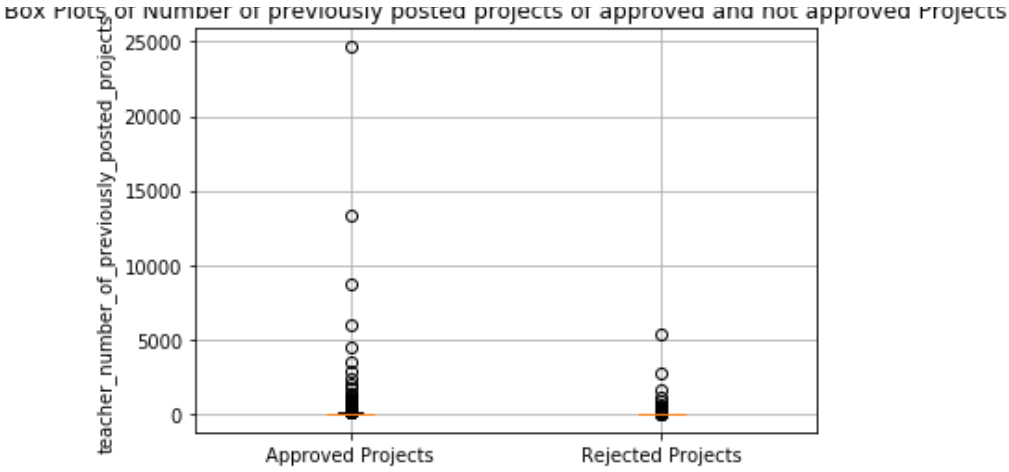
SUMMARY:

1) The approved projects tend to have lower cost when compared to the projects that have not been approved. This can be noticed by looking at the percentile values.

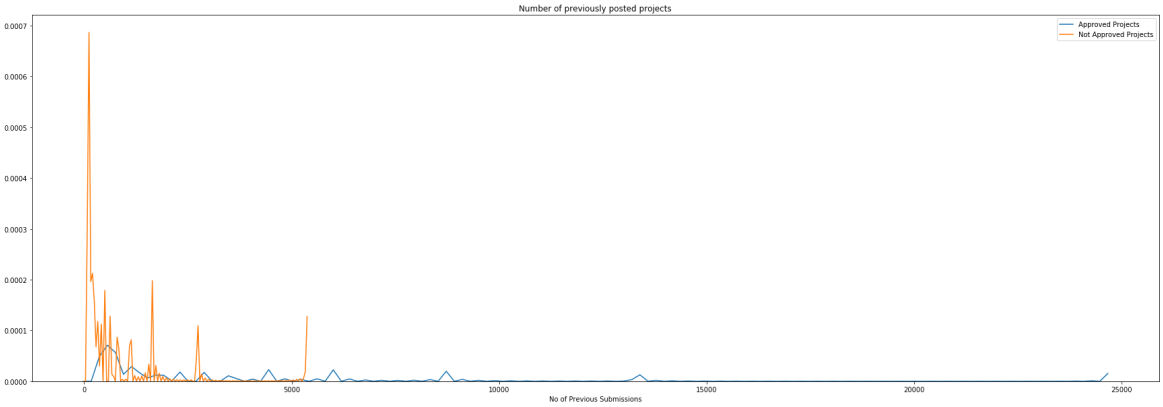
1.2.9 Univariate Analysis: teacher_number_of_previously_posted_projects

In [40]:
```
approved_noofprojects = project_data[project_data['project_is_approved']==1][
'teacher_number_of_previously_posted_projects'].value_counts()
rejected_noofprojects = project_data[project_data['project_is_approved']==0][
'teacher_number_of_previously_posted_projects'].value_counts()
```

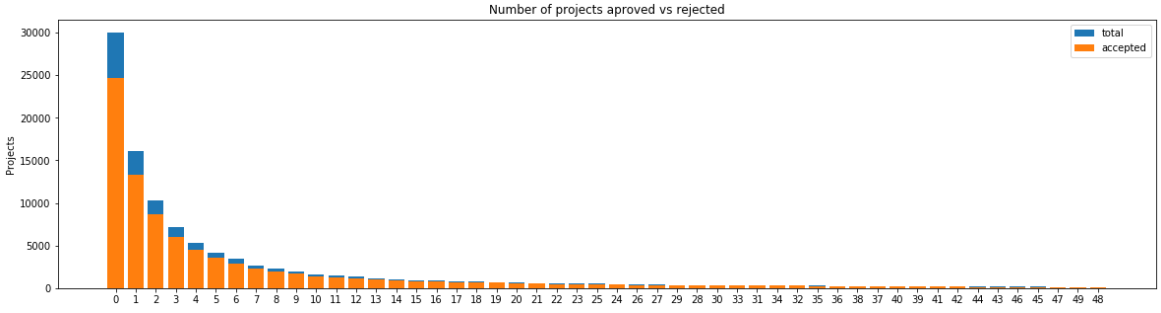
In [41]:
```
plt.boxplot([approved_noofprojects,rejected_noofprojects])
plt.title('Box Plots of Number of previously posted projects of approved and
 not approved Projects')
plt.xticks([1,2],('Approved Projects','Rejected Projects'))
plt.ylabel('teacher_number_of_previously_posted_projects')
plt.grid()
plt.show()
```



In [42]:
```
plt.figure(figsize=(30,10))
sns.distplot(approved_noofprojects, hist=False, label="Approved Projects")
sns.distplot(rejected_noofprojects, hist=False, label="Not Approved Projects"
)
plt.title('Number of previously posted projects')
plt.xlabel('No of Previous Submissions')
plt.legend()
plt.show()
```



In [43]:
```
univariate_barplots(project_data, 'teacher_number_of_previously_posted_projec
ts', 'project_is_approved', top=50)
```



	teacher_number_of_previously_posted_projects	project_is_approved	tota
1	\		
0	0	24652	3001
4			
1	1	13329	1605
8			
2	2	8705	1035
0			
3	3	5997	711

```
0
4                                                4                  4452   526
6

        Avg
0  0.821350
1  0.830054
2  0.841063
3  0.843460
4  0.845423
==================================================
    teacher_number_of_previously_posted_projects  project_is_approved  tot
al  \
46                                              46                  149    1
64
45                                              45                  141    1
53
47                                              47                  129    1
44
49                                              49                  128    1
43
48                                              48                  135    1
40

         Avg
46  0.908537
45  0.921569
47  0.895833
49  0.895105
48  0.964286
```

SUMMARY

1. There is alot of variability in the number of projects previously proposed by the teacher varying from 0 to more than 20.

2. We observe that it is not mandatory for a teacher to have proposed any project prior. Maximum number of teachers, nearly 82% of the approved projects have been submitted by teachers with no prior project proposals.

3. Very few teachers who have proposed more than 20 projects have got approval. But the rate of approval is Higher given the teacher has proposed atleast different projects.

1.2.10 Univariate Analysis: project_resource_summary

In [44]:
```
sub_catogories = list(project_data['project_resource_summary'])
sub_projectresource=[]
import re
def hasNumbers(inputString):
     return bool(re.search(r'\d', inputString))
for i in sub_catogories:
    if hasNumbers(i)==True:
        sub_projectresource.append(1)
    else:
        sub_projectresource.append(0)
```

In [45]:
```
project_data['Project_Resource'] = sub_projectresource
project_data.drop(['project_resource_summary'], axis=1, inplace=True)
```

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

Out[45]:

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

In [46]:
```
project_data['Project_Resource'].value_counts()
```

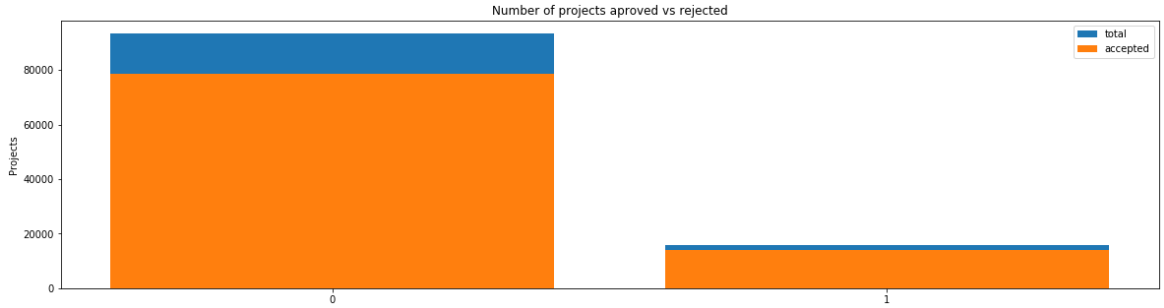
Out[46]:
```
0    93492
1    15756
Name: Project_Resource, dtype: int64
```

In [47]:
```
approved_noofprojects1 = project_data[project_data['project_is_approved']==1]
['Project_Resource'].value_counts()
rejected_noofprojects1 = project_data[project_data['project_is_approved']==0]
['Project_Resource'].value_counts()
```

In [48]:
```
print("Distribution of approved projects given 0 for no numbers present in th
e project resource summary and 1 vice versa",approved_noofprojects1)
print("Distribution of rejected projects given 0 for no numbers present in th
e project resource summary and 1 vice versa",rejected_noofprojects1)
print("Distribution of projects approved is",project_data['project_is_approve
d'].value_counts())
```

```
Distribution of approved projects given 0 for no numbers present in the pr
oject resource summary and 1 vice versa 0    78616
1    14090
Name: Project_Resource, dtype: int64
Distribution of rejected projects given 0 for no numbers present in the pr
oject resource summary and 1 vice versa 0    14876
1     1666
Name: Project_Resource, dtype: int64
Distribution of projects approved is 1    92706
0    16542
Name: project_is_approved, dtype: int64
```

In [49]:
```
univariate_barplots(project_data, 'Project_Resource', 'project_is_approved',
top=50)
```




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

SUMMARY

1. It is obvious from the graph that majority of the projects do not have numeric values stating the requirement of certain products.

2. The project summaries containing numeric values have a very high acceptance rate of 89%. Well, proper numbered requirements suggest clarity in the proposals and hence Alot of people tend to donate for a better cause, that is to help children.

1.3 Text preprocessing

1.3.1 Essay Text

In [50]:
```
project_data.head(2)
```

Out[50]:

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

In [51]:
```
# 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 seco
nd or third languages. We are a melting pot of refugees, immigrants, and n
ative-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 level of mastery.  We also have over 40 countries repres
ented with the families within our school.  Each student brings a wealth o
f knowledge and experiences to us that open our eyes to new cultures, beli
efs, and respect.\"The limits of your language are the limits of your worl
d.\"-Ludwig Wittgenstein  Our English learner's have a strong support syst
em at home that begs for more resources.  Many times our parents are learn
ing to read and speak English along side of their children.  Sometimes thi
s creates barriers for parents to be able to help their child learn phonet
ics, letter recognition, and other reading skills.\r\n\r\nBy providing the
se dvd's and players, students are able to continue their mastery of the E
nglish language even if no one at home is able to assist.  All families wi
th 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 are to help the child develop early reading skills.\r\n\r\nPare
nts that do not have access to a dvd player will have the opportunity to c
heck out a dvd player to use for the year.  The plan is to use these video
s and educational dvd's for the years to come for other EL students.\r\nna
nnan
==================================================
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% a
re minority students. \r\nThe school has a vibrant community that loves to
get together and celebrate. Around Halloween there is a whole school parad
e to show off the beautiful costumes that students wear. On Cinco de Mayo
we put on a big festival with crafts made by the students, dances, and gam
es. At the end of the year the school hosts a carnival to celebrate the ha
rd work put in during the school year, with a dunk tank being the most pop
ular activity.My students will use these five brightly colored Hokki stool
s in place of regular, stationary, 4-legged chairs. As I will only have a
total of ten in the classroom and not enough for each student to have an i
ndividual one, they will be used in a variety of ways. During independent
reading time they will be used as special chairs students will each use on
occasion. I will utilize them in place of chairs at my small group tables
during math and reading times. The rest of the day they will be used by th
e students who need the highest amount of movement in their life in order
to stay focused on school.\r\n\r\nWhenever asked what the classroom is mis
sing, my students always say more Hokki Stools. They can't get their fill
of the 5 stools we already have. When the students are sitting in group wi
th me on the Hokki Stools, they are always moving, but at the same time do
ing their work. Anytime the students get to pick where they can sit, the H
okki Stools are the first to be taken. There are always students who head
over to the kidney table to get one of the stools who are disappointed as
there are not enough of them. \r\n\r\nWe ask a lot of students to sit for
7 hours a day. The Hokki stools will be a compromise that allow my student
s to do desk work and move at the same time. These stools will help studen
ts to meet their 60 minutes a day of movement by allowing them to activate
```

es to meet their 60 minutes a day of movement by allowing them to activate their core muscles for balance while they sit. For many of my students, th ese chairs will take away the barrier that exists in schools for a child w ho can't sit still.nannan
==========
How do you remember your days of school? Was it in a sterile environment w ith plain walls, rows of desks, and a teacher in front of the room? A typi cal day in our room is nothing like that. I work hard to create a warm inv iting themed room for my students look forward to coming to each day.\r\n \r\nMy class is made up of 28 wonderfully unique boys and girls of mixed r aces in Arkansas.\r\nThey attend a Title I school, which means there is a high enough percentage of free and reduced-price lunch to qualify. Our sch ool is an \"open classroom\" concept, which is very unique as there are no walls separating the classrooms. These 9 and 10 year-old students are very eager learners; they are like sponges, absorbing all the information and e xperiences and keep on wanting more.With these resources such as the comfy red throw pillows and the whimsical nautical hanging decor and the blue fi sh nets, I will be able to help create the mood in our classroom setting t o be one of a themed nautical environment. Creating a classroom environmen t is very important in the success in each and every child's education. Th e 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 pi ctures 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 before even the first day of school! The nautical thank you c ards will be used throughout the year by the students as they create thank you cards to their team groups.\r\n\r\nYour generous donations will help m e to help make our classroom a fun, inviting, learning environment from da y one.\r\n\r\nIt costs lost of money out of my own pocket on resources to get our classroom ready. Please consider 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 language delays, cognitive delays, gross/fine motor delays, to autism. The y are eager beavers and always strive to work their hardest working past t heir limitations. \r\n\r\nThe materials we have are the ones I seek out fo r my students. I teach in a Title I school where most of the students rece ive free or reduced price lunch. Despite their disabilities and limitatio ns, 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 groov e and move as you were in a meeting? This is how my kids feel all the tim e. 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 enha nces gross motor and in Turn fine motor skills. \r\nThey also want to lear n through games, my kids don't want to sit and do worksheets. They want to learn to count by jumping and playing. Physical engagement is the key to o ur 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
==========
The mediocre teacher tells. The good teacher explains. The superior teache r demonstrates. The great teacher inspires. -William A. Ward\r\n\r\nMy sch ool has 803 students which is makeup is 97.6% African-American, making up the largest segment of the student body. A typical school in Dallas is mad e up of 23.2% African-American students. Most of the students are on free or reduced lunch. We aren't receiving doctors, lawyers, or engineers child ren from rich backgrounds or neighborhoods. As an educator I am inspiring minds of young children and we focus not only on academics but one smart, effective, efficient, and disciplined students with good character.In our classroom we can utilize the Bluetooth for swift transitions during class. I use a speaker which doesn't amplify the sound enough to receive the mess age. Due to the volume of my speaker my students can't hear videos or book s clearly and it isn't making the lessons as meaningful. But with the blue

```
tooth speaker my students will be able to hear and I can stop, pause and r
eplay it at any time.\r\nThe cart will allow me to have more room for stor
age of things that are needed for the day and has an extra part to it I ca
n use.  The table top chart has all of the letter, words and pictures for
students to learn about different letters and it is more accessible.nannan
==================================================
```

In [52]:
```
# 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 [53]:
```
sent = decontracted(project_data['essay'].values[20000])
print(sent)
print("="*50)
```

```
My kindergarten students have varied disabilities ranging from speech and
language delays, cognitive delays, gross/fine motor delays, to autism. The
y are eager beavers and always strive to work their hardest working past t
heir limitations. \r\n\r\nThe materials we have are the ones I seek out fo
r my students. I teach in a Title I school where most of the students rece
ive free or reduced price lunch.  Despite their disabilities and limitatio
ns, 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 groov
e and move as you were in a meeting? This is how my kids feel all the tim
e. 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 enha
nces gross motor and in Turn fine motor skills. \r\nThey also want to lear
n through games, my kids do not want to sit and do worksheets. They want t
o 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 happe
n. My students will forget they are doing work and just have the fun a 6 y
ear old deserves.nannan
==================================================
```

In [54]:
```
# \r \n \t remove from string python: http://texthandler.com/info/remove-line
-breaks-python/
sent = sent.replace('\\r', ' ')
sent = sent.replace('\\"', ' ')
sent = sent.replace('\\n', ' ')
print(sent)
```

```
My kindergarten students have varied disabilities ranging from speech and
language delays, cognitive delays, gross/fine motor delays, to autism. The
y are eager beavers and always strive to work their hardest working past t
heir limitations.     The 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 f
ree or reduced price lunch.  Despite their disabilities and limitations, m
y 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 their core, which enhances gro
ss 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 succes
s. The number toss and color and shape mats can make that happen. My stude
nts will forget they are doing work and just have the fun a 6 year old des
erves.nannan
```

In [55]:
```
#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
language delays cognitive delays gross fine motor delays to autism They ar
e eager beavers and always strive to work their hardest working past their
limitations The 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 red
uced price lunch Despite their disabilities and limitations my students lo
ve 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 w
ere 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 lo
ve then because they develop their core which enhances gross motor and in
Turn fine motor skills They also want to learn through games my kids do no
t 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 [56]:
```
# 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', 'yo
u', "you're", "you've",\
            "you'll", "you'd", 'your', 'yours', 'yourself', 'yourselves', 'h
e', 'him', 'his', 'himself', \
            'she', "she's", 'her', 'hers', 'herself', 'it', "it's", 'its', 'i
tself', 'they', 'them', 'their',\
            'theirs', 'themselves', 'what', 'which', 'who', 'whom', 'this',
'that', "that'll", 'these', 'those', \
            'am', 'is', 'are', 'was', 'were', 'be', 'been', 'being', 'have',
'has', 'had', 'having', 'do', 'does', \
            'did', 'doing', 'a', 'an', 'the', 'and', 'but', 'if', 'or', 'beca
use', 'as', 'until', 'while', 'of', \
            'at', 'by', 'for', 'with', 'about', 'against', 'between', 'into',
'through', 'during', 'before', 'after',\
            'above', 'below', 'to', 'from', 'up', 'down', 'in', 'out', 'on',
'off', 'over', 'under', 'again', 'further',\
            'then', 'once', 'here', 'there', 'when', 'where', 'why', 'how',
'all', 'any', 'both', 'each', 'few', 'more',\
            'most', 'other', 'some', 'such', 'only', 'own', 'same', 'so', 'th
an', 'too', 'very', \
            's', 't', 'can', 'will', 'just', 'don', "don't", 'should', "shoul
d've", 'now', 'd', 'll', 'm', 'o', 're', \
            've', 'y', 'ain', 'aren', "aren't", 'couldn', "couldn't", 'didn',
"didn't", 'doesn', "doesn't", 'hadn',\
```

```
            "hadn't", 'hasn', "hasn't", 'haven', "haven't", 'isn', "isn't",
'ma', 'mightn', "mightn't", 'mustn',\
            "mustn't", 'needn', "needn't", 'shan', "shan't", 'shouldn', "shou
ldn't", 'wasn', "wasn't", 'weren', "weren't", \
            'won', "won't", 'wouldn', "wouldn't"]
```

In [57]:
```
# 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 [00:51<00:00, 2131.64it/s]
```

In [58]:
```
# after preprocesing
preprocessed_essays[20000]
```

Out[58]:
```
'my kindergarten students varied disabilities ranging speech language dela
ys cognitive delays gross fine motor delays autism they eager beavers alwa
ys strive work hardest working past limitations the materials ones i seek
students i teach title i school students receive free reduced price lunch
despite disabilities limitations students love coming school come eager le
arn explore have ever felt like ants pants needed groove move meeting this
kids feel time the want able move learn say wobble chairs answer i love de
velop core enhances gross motor turn fine motor skills they also want lear
n games kids not want sit worksheets they want learn count jumping playing
physical engagement key success the number toss color shape mats make happ
en my students forget work fun 6 year old deserves nannan'
```

In [59]:
```
project_data['project_essayyys']=preprocessed_essays
project_data['project_essayyys']
```

Out[59]:
```
0         my students english learners working english s...
1         our students arrive school eager learn they po...
2         true champions not always ones win guts by mia...
3         i work unique school filled esl english second...
4         our second grade classroom next year made arou...
5         i moving 2nd grade 3rd grade beginning next sc...
6         my students dynamic energetic group middle sch...
7         not students struggle poverty also learning ma...
8         my students enthusiastic inquisitive learners ...
9         over 95 students free reduced lunch i homeless...
10        there many little ways enlarge world love book...
11        all students receive free breakfast lunch scho...
12        my students always working new projects time w...
13        i teach small school district central oklahoma...
14        my students babies i want world i teach three ...
15        located west dallas students face several chal...
16        my preschool children ages 3 5 years old autis...
17        my students special come variety backgrounds i...
18        i teach title i school low income area many st...
19        we apart urban district many students come fin...
20        the students school come diverse backgrounds v...
```

```
21        my students walk school every day full energy ...
22        every day english classroom work develop stude...
23        100 musical students eat free breakfast lunch ...
24        this year i teaching efl extended foreign lang...
25        my students highly motivated succeed unfortuna...
26        i teach 22 bright 5 6 year olds my students at...
27        my students spend day learning fourth grade su...
28        my students primary diagnosis autism secondary...
29        i awesome group 24 students teacher could ask ...
                                ...
109218    my students amazing group kind hearted loving ...
109219    creating interactive learning environment help...
109220    do remember middle school i sure tons adjectiv...
109221    most students enl students this first dual lan...
109222    for students first time school kindergarten mu...
109223    the students i serve low income community stud...
109224    my students amazing group kids these kids come...
109225    my students come ready rock roll every day we ...
109226    my students inquisitive engaging 4th graders t...
109227    my students attend title 1 school upstate new ...
109228    every day kindergarten class comes excited lea...
109229    each morning 21 loving smiling faces walk door...
109230    described diverse group characters despite siz...
109231    there nothing better seeing student succeed so...
109232    our class home diverse class students economic...
109233    my students hard workers strive best my studen...
109234    kind respectful eager learners they come schoo...
109235    as kindergarten teacher low income high povert...
109236    many students hard time making connection i te...
109237    you find magic wherever look sit back relax ne...
109238    we bucket fillers this means fill buckets ever...
109239    my students amazing motivated we inner city he...
109240    leaving family come school first time scary th...
109241    i wonderful group inquisitive enthusiastic lea...
109242    my students come rural community south carolin...
109243    welcome mr ramos 2nd grade classroom we title ...
109244    every morning start day core values lead solel...
109245    this great group sharing caring students it mu...
109246    our students live small rural community our cl...
109247    when last time used math probably within last ...
Name: project_essayyys, Length: 109248, dtype: object
```

1.3.2 Project title Text

In [60]:
```
# similarly you can preprocess the titles also
# 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 [61]:
```
sent1 = decontracted(project_data['project_title'].values[20000])
print(sent1)
print("="*50)
```

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

In [62]:
```
# \r \n \t remove from string python: http://texthandler.com/info/remove-line-breaks-python/
sent1 = sent1.replace('\\r', ' ')
sent1 = sent1.replace('\\"', ' ')
sent1 = sent1.replace('\\n', ' ')
print(sent1)
```

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

In [63]:
```
#remove spacial character: https://stackoverflow.com/a/5843547/4084039
sent1 = re.sub('[^A-Za-z0-9]+', ' ', sent1)
print(sent1)
```

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

In [64]:
```
# Combining all the above statemennts
from tqdm import tqdm
preprocessed_projectitle = []
# tqdm is for printing the status bar
for sentance in tqdm(project_data['project_title'].values):
    sent1 = decontracted(sentance)
    sent1 = sent1.replace('\\r', ' ')
    sent1 = sent1.replace('\\"', ' ')
    sent1 = sent1.replace('\\n', ' ')
    sent1 = re.sub('[^A-Za-z0-9]+', ' ', sent1)
    # https://gist.github.com/sebleier/554280
    sent1 = ' '.join(e for e in sent1.split() if e not in stopwords)
    preprocessed_projectitle.append(sent1.lower().strip())
```

```
100%|██████████| 109248/109248 [00:02<00:00, 47032.95it/s]
```

In [65]:
```
# after preprocesing
preprocessed_projectitle[19995:20000]
```

Out[65]:
```
['everyday pre k feels like day beach',
 'flexible seating fifth',
 'bouncy active learners',
 'paint brushes paint talented young artists',
 'help students learn love reading']
```

In [66]:
```
project_data.drop(['project_title'], axis=1, inplace=True)
project_data.head(2)
project_data['project_title']=preprocessed_projectitle
project_data['project_title']
```

Out[66]:
```
0                  educational support english learners home
1                          wanted projector hungry learners
```

```
1                          wanted projector hungry learners
2            soccer equipment awesome middle school students
3                                    techie kindergarteners
4                                    interactive math tools
5          flexible seating mrs jarvis terrific third gra...
6              chromebooks special education reading program
7                                          it 21st century
8                               targeting more success class
9                        just for love reading pure pleasure
10                                    reading changes lives
11         elevating academics parent rapports through te...
12                       building life science experiences
13                                 everyone deserves heard
14                            tablets can show us the world
15                                      making recess active
16                          making great leap with leapfrog
17                 technology teaches tomorrow talents today
18                                                 test time
19                                  wiggling our way success
20                             magic carpet ride our library
21                            from sitting standing classroom
22                               books budding intellectuals
23                       instrumental power conquering steam
24         s t e a m challenges science technology engine...
25                                                 math masters
26                                               techy teaching
27                    4th grade french immersion class ipads
28                                 hands on language literacy
29                           basic classroom supplies needed
                                   ...
109218                           multi sensory classroom wish
109219                            make learning fun grade one
109220                  hooking young readers engaging books
109221                                    dual language class
109222     replenishing our supplies extend our learning ...
109223                                 hunger busters students
109224                                          stem 2nd grade
109225                                     together we learn
109226                                     stand up learning
109227                               grab stool fun about start
109228             technology for flooded kindergarten class
109229                      criss cross applesauce ready roll
109230          ipad minis special needs high school students
109231                     keeping students informed inspired
109232                                  everyone needs opinion
109233                                      engagement tablets
109234            developing a growth mindset school success
109235                                     let focus movement
109236                                     portable projector
109237                       choose kindness book club wonder
109238      we like move it move it flexible seating options
109239                                       integrating arts
109240                                  spread love literature
109241                                     read your heart out
109242                        stem learners need an ipad mini
109243     privacy shields help promote independent thinking
109244                                  technology our classroom
109245                        2016 2017 beginning year basics
109246                    flexible seating inclusive classroom


109247             classroom tech develop 21st century leaders
Name: project_title, Length: 109248, dtype: object
```

DATA PREPROCESSING OF TEACHER_PREFIX

In [67]:
```
from tqdm import tqdm
preprocessed_tf = []
# tqdm is for printing the status bar
for sentance in tqdm(project_data['teacher_prefix'].values):
    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_tf.append(sent.lower().strip())
```

```
100%|██████████| 109248/109248 [00:02<00:00, 38852.74it/s]
```

In [68]:
```
project_data['teacher_prefix'].fillna('',inplace=True)
```

1. 4 Preparing data for models

In [69]:
```
project_data.columns
```

Out[69]:
```
Index(['Unnamed: 0', 'id', 'teacher_id', 'teacher_prefix', 'school_state',
       'project_submitted_datetime', 'project_grade_category',
       'project_essay_1', 'project_essay_2', 'project_essay_3',
       'project_essay_4', 'teacher_number_of_previously_posted_projects',
       'project_is_approved', 'clean_categories', 'clean_subcategories',
       'essay', 'price', 'quantity', 'Project_Resource', 'project_essayyy
s',
       'project_title'],
      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: text data CONVERTED to numerical 0 & 1 0(numeri
   cal quantity not mentioned) & 1(numerical quantity mentioned)

       - 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/

In [70]:
```
# 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=False, binary=True)
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', 'SpecialNeeds', 'Health_Sports', 'Math_Science', 'Literacy_Language']
Shape of matrix after one hot encodig  (109248, 9)
```

In [71]:
```
# we use count vectorizer to convert the values into one hot encoded features
vectorizer = CountVectorizer(vocabulary=list(sorted_sub_cat_dict.keys()), lowercase=False, binary=True)
vectorizer.fit(project_data['clean_subcategories'].values)
print(vectorizer.get_feature_names())


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

```
['Economics', 'CommunityService', 'FinancialLiteracy', 'ParentInvolvement', 'Extracurricular', 'Civics_Government', 'ForeignLanguages', 'NutritionEducation', 'Warmth', 'Care_Hunger', 'SocialSciences', 'PerformingArts', 'CharacterEducation', 'TeamSports', 'Other', 'College_CareerPrep', 'Music', 'History_Geography', 'Health_LifeScience', 'EarlyDevelopment', 'ESL', 'Gym_Fitness', 'EnvironmentalScience', 'VisualArts', 'Health_Wellness', 'AppliedSciences', 'SpecialNeeds', 'Literature_Writing', 'Mathematics', 'Literacy']
Shape of matrix after one hot encoding  (109248, 30)
```

In [72]:
```
# STATE
from collections import Counter
my_counter = Counter()
for word in project_data['school_state'].values:
    my_counter.update(word.split())
# dict sort by value python: https://stackoverflow.com/a/613218/4084039
ss_dict = dict(my_counter)
sorted_ss_dict = dict(sorted(ss_dict.items(), key=lambda kv: kv[1]))
```

In [73]:
```
vectorizer = CountVectorizer(vocabulary=list(sorted_ss_dict.keys()), lowercase=False, binary=True)
vectorizer.fit(project_data['school_state'].values)
print(vectorizer.get_feature_names())


ss_one_hot = vectorizer.transform(project_data['school_state'].values)
print("Shape of matrix after one hot encoding ",ss_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', 'N
```

```
V', 'MD', 'CT', 'TN', 'UT', 'AL', 'WI', 'VA', 'AZ', 'NJ', 'OK', 'WA', 'M
A', 'LA', 'OH', 'MO', 'IN', 'PA', 'MI', 'SC', 'GA', 'IL', 'NC', 'FL', 'N
Y', 'TX', 'CA']
Shape of matrix after one hot encoding  (109248, 51)
```

In [74]:
```
# TEACHER PREFIX Please do the similar feature encoding with state, teacher_p
refix and project_grade_category also
```

In [75]:
```
vectorizer = CountVectorizer(binary=True)
tp_one_hot=vectorizer.fit_transform(project_data['teacher_prefix'])
print(vectorizer.get_feature_names())
print("Shape of matrix after one hot encoding ",tp_one_hot.shape)
```

```
['dr', 'mr', 'mrs', 'ms', 'teacher']
Shape of matrix after one hot encoding  (109248, 5)
```

In [76]:
```
#PROJECT GRADE CATAGORY from collections import Counter
my_counter = Counter()
for word in project_data['project_grade_category'].values:
    my_counter.update(word.split())
# dict sort by value python: https://stackoverflow.com/a/613218/4084039
pgc_dict = dict(my_counter)
sorted_pgc_dict = dict(sorted(pgc_dict.items(), key=lambda kv: kv[1]))
```

In [83]:
```
del sorted_pgc_dict["Grades"]
```

In [84]:
```
vectorizer = CountVectorizer(vocabulary=list(sorted_pgc_dict.keys()), lowerca
se=False, binary=True)
vectorizer.fit(project_data['project_grade_category'].values)
print(vectorizer.get_feature_names())


pgc_one_hot = vectorizer.transform(project_data['project_grade_category'].val
ues)
print("Shape of matrix after one hot encoding ",pgc_one_hot.shape)
```

```
['9-12', '6-8', '3-5', 'PreK-2']
Shape of matrix after one hot encoding  (109248, 4)
```

1.4.2 Vectorizing Text data

1.4.2.1 Bag of words

In [78]:
```
# We are considering only the words which appeared in at least 10 documents(r
ows or projects).
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 [79]:
```
# you can vectorize the title also
# before you vectorize the title make sure you preprocess it
vectorizer = CountVectorizer(min_df=3)
```

```
text_bowpt = vectorizer.fit_transform(preprocessed_projectitle)
print("Shape of matrix after one hot encodig ",text_bowpt.shape)
```

```
Shape of matrix after one hot encodig  (109248, 7013)
```

1.4.2.3 TFIDF vectorizer

In [80]:
```
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 [81]:
```
# Similarly you can vectorize for title also
vectorizer = TfidfVectorizer(min_df=3)
text_tfidfpt = vectorizer.fit_transform(preprocessed_projectitle)
print("Shape of matrix after one hot encodig ",text_tfidfpt.shape)
```

```
Shape of matrix after one hot encodig  (109248, 7013)
```

1.4.2.5 Using Pretrained Models: Avg W2V

In [82]:
```
# Reading glove vectors in python: https://stackoverflow.com/a/38230349/40840
39
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')
words = []
for i in preprocessed_essays:
    words.extend(i.split(' '))
for i in preprocessed_projectitle:
    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 cou
pus", \
      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/ho
```

```
w-to-use-pickle-to-save-and-load-variables-in-python/
import pickle
with open('glove_vectors', 'wb') as f:
    pickle.dump(words_courpus, f)
```

```
Loading Glove Model
```

```
1917495it [03:32, 9038.02it/s]
```

```
Done. 1917495  words loaded!
all the words in the coupus 17014413
the unique words in the coupus 58968
The number of words that are present in both glove vectors and our coupus
51503 ( 87.341 %)
word 2 vec length 51503
```

In [85]:
```
# stronging variables into pickle files python: http://www.jessicayung.com/ho
w-to-use-pickle-to-save-and-load-variables-in-python/
# 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 [86]:
```
# average Word2Vec
# compute average word2vec for each review.
avg_w2v_vectors = []; # the avg-w2v for each sentence/review is stored in thi
s list
for sentence in tqdm(preprocessed_essays): # for each essays
    vector = np.zeros(300) # as word vectors are of zero length
    cnt_words =0; # num of words with a valid vector in the essay
    for word in sentence.split(): # for each word in a essay
        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 [00:34<00:00, 3151.49it/s]
```

```
109248
300
```

1.4.2.6 Using Pretrained Models: AVG W2V on `project_title`

In [87]:
```
# Similarly you can vectorize for title also
avg_w2v_vectors2 = []; # the avg-w2v for each sentence/review is stored in th
is list
for sentence in tqdm(preprocessed_projectitle): # for each title
    vector = np.zeros(300) # as word vectors are of zero length
    cnt_words =0; # num of words with a valid vector in the essay
    for word in sentence.split(): # for each word in a essay
        if word in glove_words:
            vector += model[word]
            cnt_words += 1
    if cnt_words != 0:
        vector /= cnt_words
    avg_w2v_vectors2.append(vector)
```

```
print(len(avg_w2v_vectors2))
print(len(avg_w2v_vectors2[2]))
```

```
100%|██████████| 109248/109248 [00:02<00:00, 53478.17it/s]
```

```
109248
300
```

1.4.2.7 Using Pretrained Models: TFIDF weighted W2V

In [88]:
```
# 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 [89]:
```
# average Word2Vec
# compute average word2vec for each review.
tfidf_w2v_vectors = []; # the avg-w2v for each sentence/review is stored in t
his list
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/revi
ew
    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
 value((sentence.count(word)/len(sentence.split())))
            tf_idf = dictionary[word]*(sentence.count(word)/len(sentence.spli
t())) # getting the tfidf value for each word
            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 [03:19<00:00, 579.81it/s]
```

```
109248
300
```

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

In [90]:
```
# 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_projectitle)
# 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 [91]:
```
tfidf_w2v_vectors1 = []; # the avg-w2v for each sentence/review is stored in
 this list
for sentence in tqdm(preprocessed_projectitle): # 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/revi
ew
    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
 value((sentence.count(word)/len(sentence.split())))
            tf_idf = dictionary[word]*(sentence.count(word)/len(sentence.spli
t())) # getting the tfidf value for each word
            vector += (vec * tf_idf) # calculating tfidf weighted w2v
            tf_idf_weight += tf_idf
    if tf_idf_weight != 0:
        vector /= tf_idf_weight
    tfidf_w2v_vectors1.append(vector)

print(len(tfidf_w2v_vectors1))
print(len(tfidf_w2v_vectors1[0]))
```

```
100%|██████████| 109248/109248 [00:02<00:00, 37854.03it/s]
```

```
109248
300
```

1.4.3 Vectorizing Numerical features

In [92]:
```
# check this one: https://www.youtube.com/watch?v=0HOqOcln3Z4&t=530s
# standardization sklearn: https://scikit-learn.org/stable/modules/generated/
sklearn.preprocessing.StandardScaler.html
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 3
29.   ... 399.   287.73   5.5 ].
# Reshape your data either using array.reshape(-1, 1)

price_scalar = StandardScaler()
price_scalar.fit(project_data['price'].values.reshape(-1,1)) # finding the me
an and standard deviation of this data
print(f"Mean : {price_scalar.mean_[0]}, Standard deviation : {np.sqrt(price_s
calar.var_[0])}")

# Now standardize the data with above maen and variance.
price_standardized = price_scalar.transform(project_data['price'].values.resh
ape(-1, 1))
```

```
Mean : 298.1193425966608, Standard deviation : 367.49634838483496
```

In [93]:
```
price_standardized
```

Out[93]:
```
array([[-0.3905327 ],
       [ 0.00239637],
       [ 0.59519138],
       ...,
       [-0.15825829],
       [-0.61243967],
       [-0.51216657]])
```

In [94]:
```
teacher_number_of_previously_posted_projects_scalar = StandardScaler()
teacher_number_of_previously_posted_projects_scalar.fit(project_data['teacher
_number_of_previously_posted_projects'].values.reshape(-1,1)) # finding the m
ean and standard deviation of this data
print(f"Mean : {price_scalar.mean_[0]}, Standard deviation : {np.sqrt(price_s
calar.var_[0])}")

# Now standardize the data with above maen and variance.
teacher_number_of_previously_posted_projects_standardized = teacher_number_of
_previously_posted_projects_scalar.transform(project_data['price'].values.res
hape(-1, 1))
```

```
F:\Anaconda3\lib\site-packages\sklearn\utils\validation.py:475: DataConver
sionWarning:

Data with input dtype int64 was converted to float64 by StandardScaler.
```

```
Mean : 298.1193425966608, Standard deviation : 367.49634838483496
```

1.4.4 Merging all the above features

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

In [95]:
```
print(categories_one_hot.shape)
print(sub_categories_one_hot.shape)
print(text_bow.shape)
print(price_standardized.shape)
print(pgc_one_hot.shape)
print(tp_one_hot.shape)
print(text_bowpt.shape)
print(ss_one_hot.shape)
```

```
(109248, 9)
(109248, 30)
(109248, 16623)
(109248, 1)
(109248, 4)
(109248, 5)
(109248, 7013)
(109248, 51)
```

In [96]:
```
# 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 de
nse matirx :)
X = hstack((categories_one_hot,sub_categories_one_hot,price_standardized,text
_bowpt,tp_one_hot,ss_one_hot,pgc_one_hot,teacher_number_of_previously_posted_
projects_standardized))
X.shape
```

Out[96]:
```
(109248, 7114)
```

In [97]:
```
X1 = hstack((categories_one_hot,sub_categories_one_hot,price_standardized,tex
t_tfidfpt,tp_one_hot,ss_one_hot,pgc_one_hot,teacher_number_of_previously_post
ed_projects_standardized))
X1.shape
```

Out[97]:
```
(109248, 7114)
```

In [98]:
```
X2 = hstack((categories_one_hot,sub_categories_one_hot,price_standardized,avg_w2v_vectors2,tp_one_hot,ss_one_hot,pgc_one_hot,teacher_number_of_previously_posted_projects_standardized))
X2.shape
```

Out[98]: (109248, 401)

In [99]:
```
X3 = hstack((categories_one_hot,sub_categories_one_hot,price_standardized,tfidf_w2v_vectors1,tp_one_hot,ss_one_hot,pgc_one_hot,teacher_number_of_previously_posted_projects_standardized))
X3.shape
```

Out[99]: (109248, 401)

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

TAKING THE FIRST 5000 DATA POINTS AS I HAVE A RAM OF ONLY 6 GB :

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

In [100]:
```
import numpy as np
from sklearn.manifold import TSNE
```

```
import pandas as pd
import matplotlib.pyplot as plt
import seaborn as sns
```

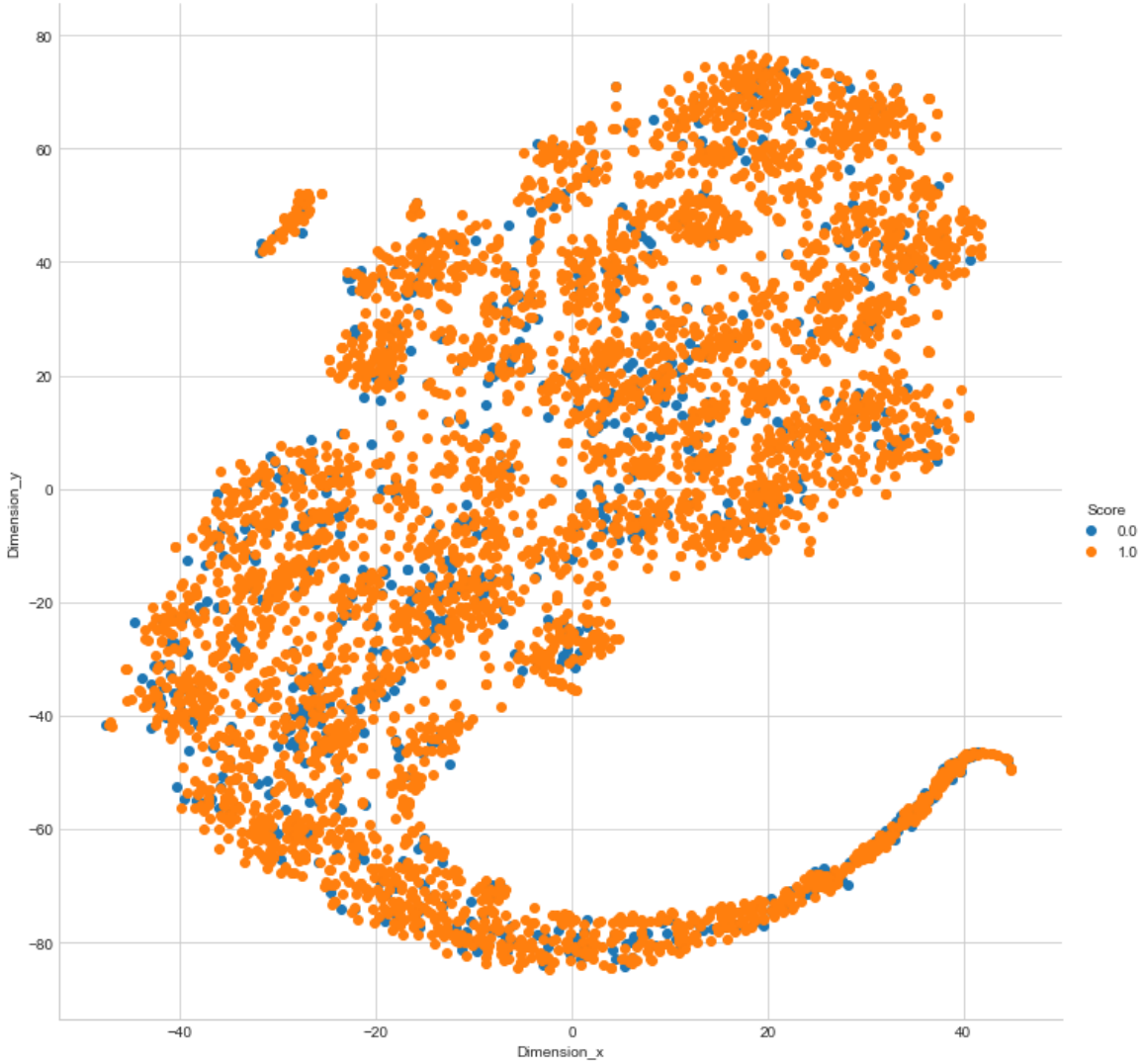
In [101]:
```
k = X.toarray()
x= k[:5000]
```

In [102]:
```
y = project_data['project_is_approved'][:5000]
y= y.as_matrix(columns=None)
```

In [106]:
```
tsne = TSNE(n_components=2, perplexity=40, learning_rate=200,n_iter=1500)
X_embedding = tsne.fit_transform(x)
```

In [107]:
```
for_tsne = np.hstack((X_embedding,y.reshape(-1,1)))
for_tsne_df = pd.DataFrame(data=for_tsne, columns=['Dimension_x','Dimension_y','Score'])
```

In [108]:
```
sns.set_style("whitegrid")
sns.FacetGrid(for_tsne_df,hue = "Score", size = 10).map(plt.scatter,"Dimension_x",'Dimension_y').add_legend()
plt.show()
```



SUMMARY:

1. We observe alot of overlapping in the datapoints.

2. The points are well scattered, unable to draw any proper conclusion.

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

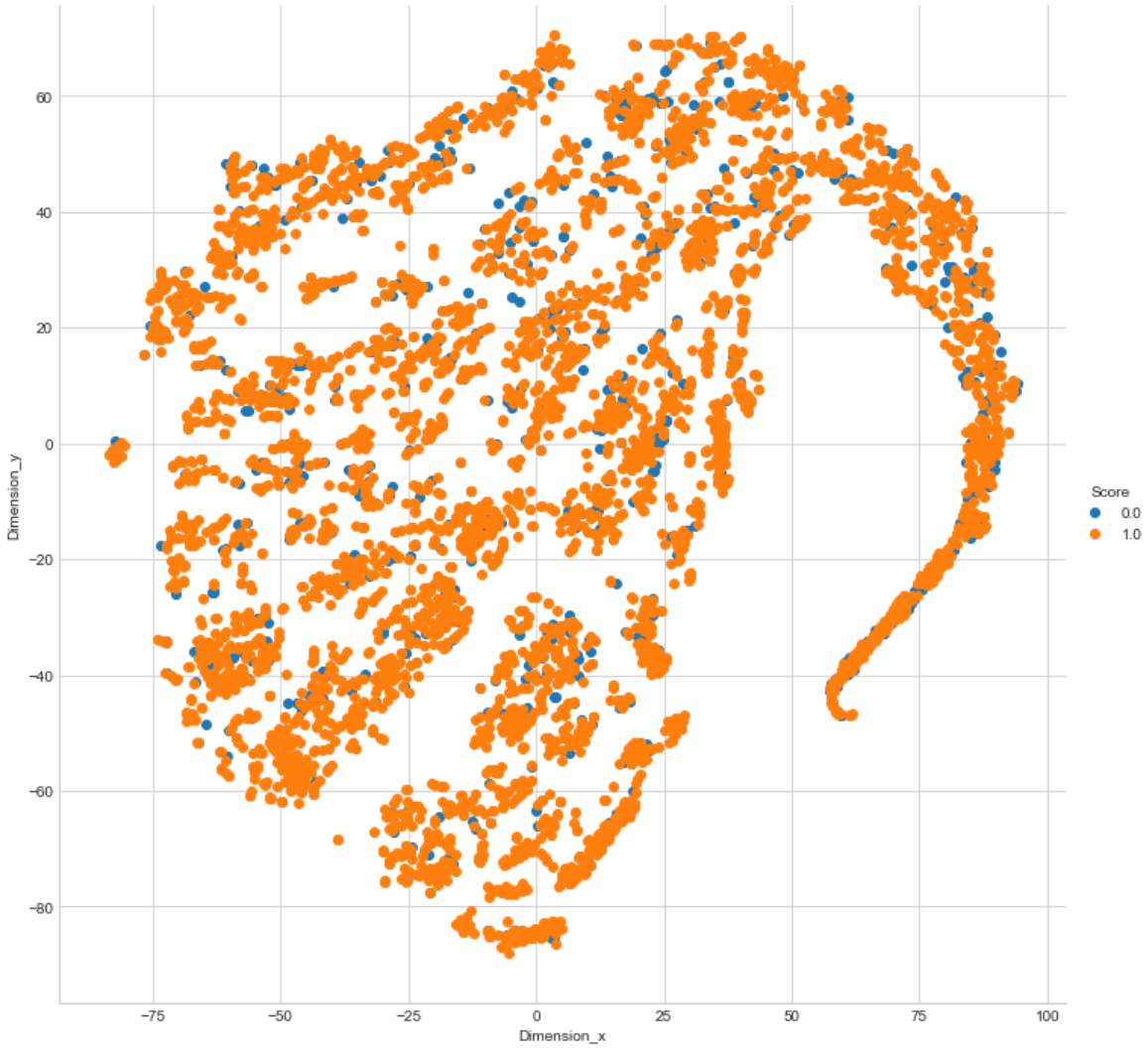
In [109]:
```
k1 = X1.toarray()
x1= k1[:5000]
```

In [110]:
```
y = project_data['project_is_approved'][:5000]
y= y.as_matrix(columns=None)
```

In [111]:
```
tsne = TSNE(n_components=2, perplexity=30, learning_rate=250,n_iter=1500)
X1_embedding = tsne.fit_transform(x1)
```

In [112]:
```
for_tsne1 = np.hstack((X1_embedding,y.reshape(-1,1)))
for_tsne_df1 = pd.DataFrame(data=for_tsne1, columns=['Dimension_x','Dimensio
n_y','Score'])
```

In [113]:
```
sns.set_style("whitegrid")
sns.FacetGrid(for_tsne_df1,hue = "Score", size = 10).map(plt.scatter,"Dimens
ion_x",'Dimension_y').add_legend()
plt.show()
```



SUMMARY:

1. We observe alot of overlapping in the datapoints.

2. The points are well scattered, unable to draw any proper conclusion.

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

In [114]:
```
k2 = X2.toarray()
x2= k2[:5000]
```

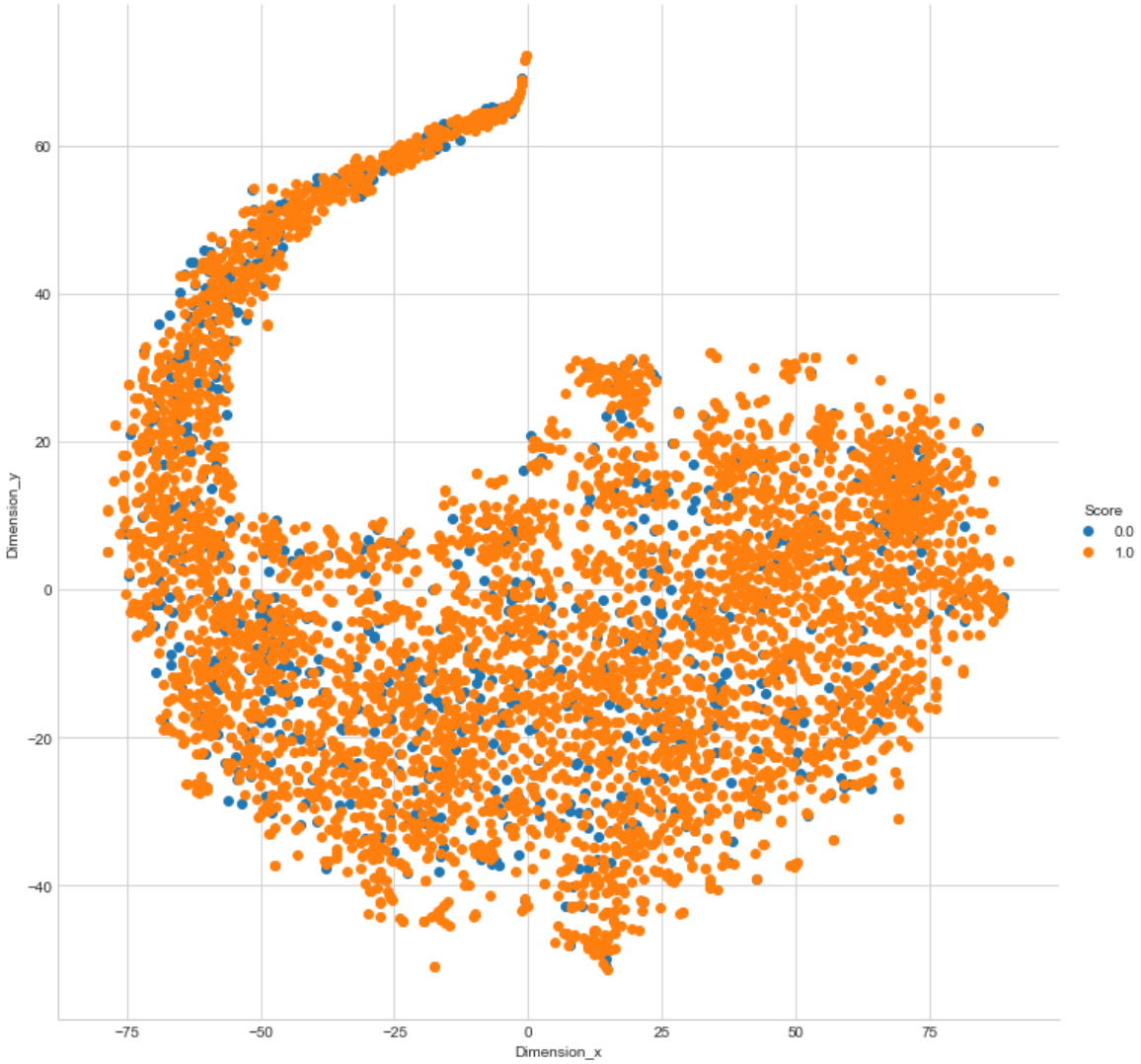
In [115]:
```
y = project_data['project_is_approved'][:5000]
y = y.as_matrix(columns=None)
```

In [116]:
```
tsne = TSNE(n_components=2, perplexity=35, learning_rate=200,n_iter=1500)
X2_embedding = tsne.fit_transform(x2)
```

In [117]:
```
for_tsne2 = np.hstack((X2_embedding,y.reshape(-1,1)))
for_tsne_df2 = pd.DataFrame(data=for_tsne2, columns=['Dimension_x','Dimensio
n_y','Score'])
```

In [118]:
```
sns.set_style("whitegrid")
sns.FacetGrid(for_tsne_df2,hue = "Score", size = 10).map(plt.scatter,"Dimens
```

```
ion_x",'Dimension_y').add_legend()
plt.show()
```



SUMMARY:

1. We observe alot of overlapping in the datapoints.

2. The points are well scattered, unable to draw any proper conclusion.

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

In [119]:
```
k3 = X3.toarray()
x3= k3[:5000]
```

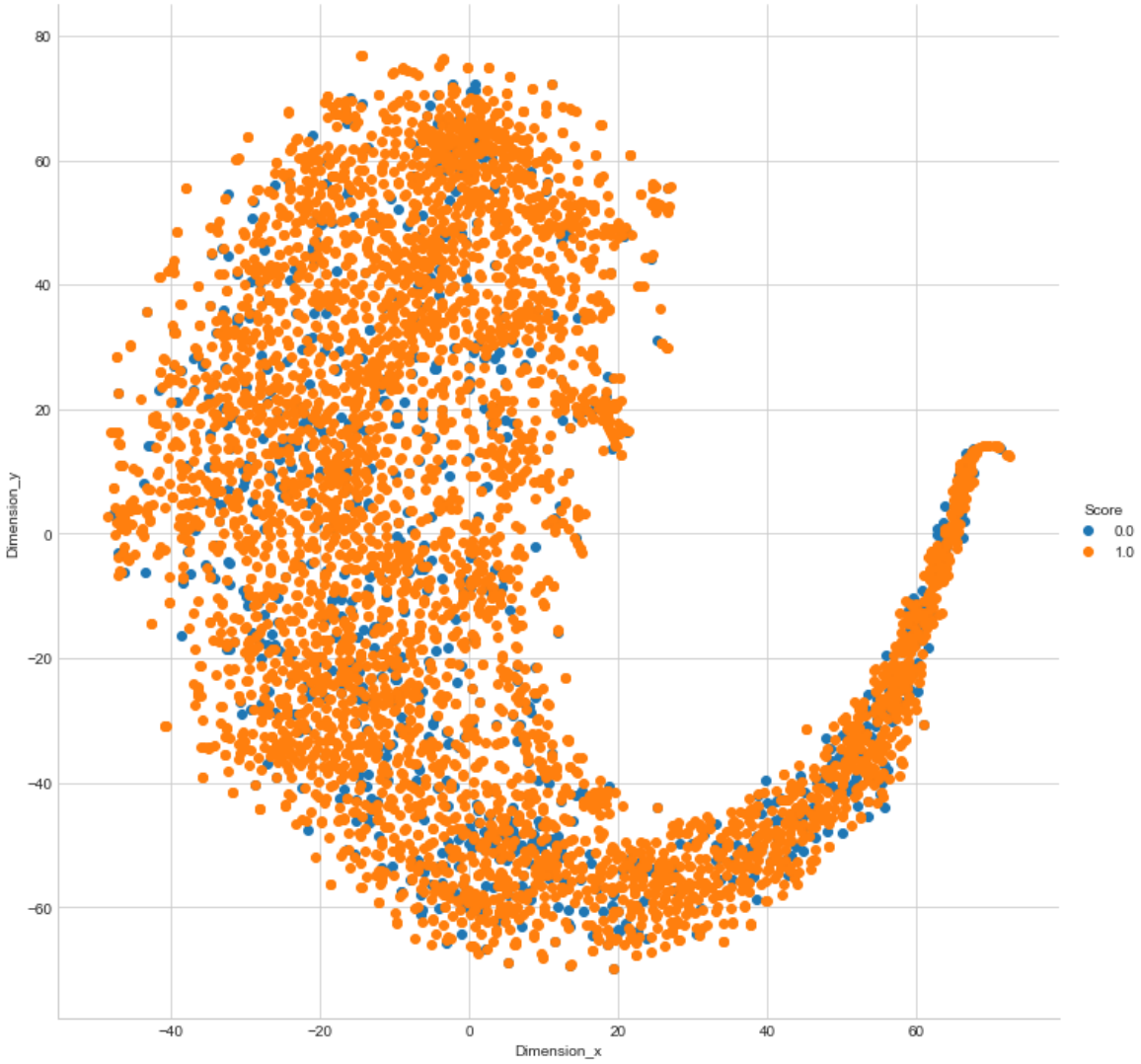
In [120]:
```
y = project_data['project_is_approved'][:5000]
y = y.as_matrix(columns=None)
```

In [121]:
```
tsne = TSNE(n_components=2,perplexity=40,learning_rate=150,n_iter=2000)
X3_embedding = tsne.fit_transform(x3)
```

In [122]:
```
for_tsne3 = np.hstack((X3_embedding,y.reshape(-1,1)))
```

```
for_tsne_df3 = pd.DataFrame(data=for_tsne3, columns=['Dimension_x','Dimensio
n_y','Score'])
```

In [123]:
```
sns.set_style("whitegrid")
sns.FacetGrid(for_tsne_df3,hue = "Score", size = 10).map(plt.scatter,"Dimens
ion_x",'Dimension_y').add_legend()
plt.show()
```



SUMMARY:

1. We observe alot of overlapping in the datapoints.

2. The points are well scattered, unable to draw any proper conclusion.

2.5 Summary

1)Visualisation of TSNE with Bag of Words, TF-IDF, Avg Word2Vec, TF-IDF Weighted Word2Vec does not seem to yield the expected result of clustering similar data points.

2) ALL THE ABOVE SUMMARY OR CONCLUSIONS ARE BASED ON PERSONAL ANALYIZATION! AND THE SAME ARE DRAFTED AFTER EACH AND EVERY PLOTS STATED OR SHOWN ABOVE

In [124]:
```
############################################## DONE ####################
############################
```