DonorsChoose

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

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

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

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

About the DonorsChoose Data Set

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

Feature	
`project_id`	A unique identifier for the proposed project.
`project_title`	Title of th • `Art Will` •
`project_grade_category`	Grade level of students for which the project is targeted. • • • •

Feature	
project_subject_categories	One or more (comma-separated) subject categories fo following enum • • • • • Lit • • • • • • `Literacy & Language`
school_state	State where school is located ([Two-le (https://en.wikipedia.org/wiki/List_of_U.S._state_abbreviati](https://en.wikipedia.org/wiki/List_of_U.S._state_abbreviati)
project_subject_subcategories	One or more (comma-separated) subject subcateg • • `Literature & Writing,`
project_resource_summary	An explanation of the resources needed for th • `My students need hands on literacy mate sens`
project_essay_1	Fi
project_essay_2	Seco
project_essay_3	Thi
project_essay_4	Fou
project_submitted_datetime	Datetime when project application was submitted. **Exa**
teacher_id	A unique identifier for the teacher of the propose `bdf8baa8fedef6bf`
teacher_prefix	Teacher's title. One of the following • • • • • •
teacher_number_of_previously_posted_projects	Number of project applications previously submitted

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

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

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

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

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

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

Notes on the Essay Data

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

- **project_essay_1:** "Introduce us to your classroom"
- **project_essay_2:** "Tell us more about your students"
- **project_essay_3:** "Describe how your students will use the materials you're requesting"
- **project_essay_4:** "Close by sharing why your project will make a difference"

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

- **project_essay_1:** "Describe your students: What makes your students special? Specific details about their background, your neighborhood, and your school are all helpful."
- **project_essay_2:** "About your project: How will these materials make a difference in your students' learning and improve their school lives?"

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

In [1]:

```
%matplotlib inline
import warnings
warnings.filterwarnings("ignore")

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

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

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

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

from tqdm import tqdm
import os

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

```
:\Anaconda\lib\site-packages\gensim\utils.py:1197: UserWarning: detected Windo
s; 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)
```

```
 umber of data points in train data (109248, 17)
--------------------------------------------------
 he attributes of data : ['Unnamed: 0' 'id' 'teacher_id' 'teacher_prefix' 'scho
 l_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

Checking for nan Values

In [5]:

```
print('The Columns with their nan values counts are below ')
for col in project_data.columns:
    print('{col} '.format(col=col),project_data[col].isnull().sum())
```

```
The Columns with their nan values counts are below
Unnamed: 0  0
id  0
teacher_id  0
teacher_prefix  3
school_state  0
project_submitted_datetime  0
project_grade_category  0
project_subject_categories  0
project_subject_subcategories  0
project_title  0
project_essay_1  0
project_essay_2  0
project_essay_3  105490
project_essay_4  105490
project_resource_summary  0
teacher_number_of_previously_posted_projects  0
project_is_approved  0
```

The Variable teacher_prefix has 3 missing values and project essays 3 and 4 are almost in 105k range. However, for project essays it's justifyable as system got changed after few years but for teacher prefix i think they have been mishandled.

In [6]:

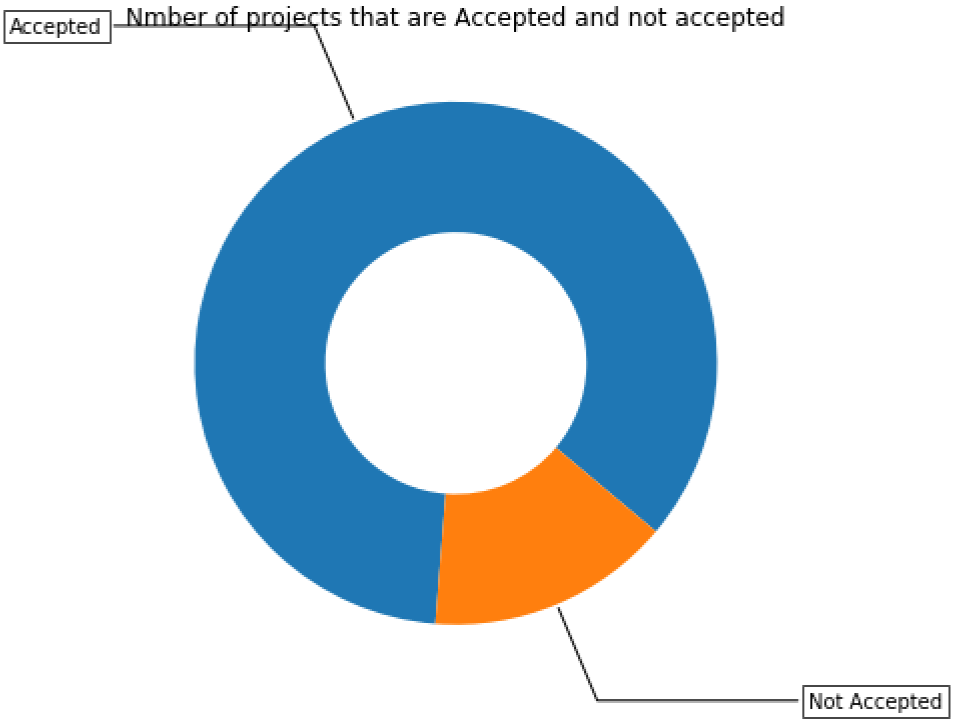
```
# removing 3 nan values from teacher prefix column as they seems to be outliers
# DataFrame.dropna(axis=0, how='any', thresh=None, subset=None, inplace=False)
project_data.dropna(subset=['teacher_prefix'],inplace=True)
```

1.2 Data Analysis

In [7]:

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


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

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

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

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

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

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

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

plt.show()
```

```
umber of projects thar are approved for funding  92703 , ( 84.85788823287108
)
umber of projects thar are not approved for funding  16542 , ( 15.142111767128
3 %)
```



Observation :

From Donut plot, we can see that the chances of project getting approved seems to be high as almost 85%. It means that If a teacher will post his project on this platform, he will have 85% chance for approval for his project by Donors.

1.2.1 Univariate Analysis: School State

In [8]:

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

temp = pd.DataFrame(project_data.groupby("school_state")["project_is_approved"].a
# if you have data which contain only 0 and 1, then the mean = percentage (think
temp.columns = ['state_code', 'num_proposals']
# How to plot US state heatmap: https://datascience.stackexchange.com/a/9620

scl = [[0.0, 'rgb(242,240,247)'],[0.2, 'rgb(218,218,235)'],[0.4, 'rgb(188,189,220
            [0.6, 'rgb(158,154,200)'],[0.8, 'rgb(117,107,177)'],[1.0, 'rgb(84,39,

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

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

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

In [9]:

```
# https://www.csi.cuny.edu/sites/default/files/pdf/administration/ops/2letterstab
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
```

Observations:

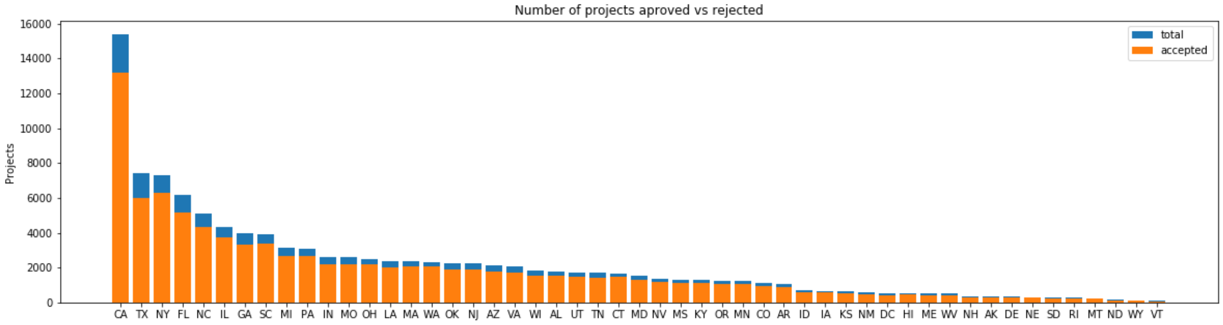
The state vermont has the least acceptance rate and Delaware has the highest acceptance rate.

In [10]:

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

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

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

```
In [11]: def univariate_barplots(data, col1, col2='project_is_approved', top=False):
             # Count number of zeros in dataframe python: https://stackoverflow.com/a/5154
             temp = pd.DataFrame(project_data.groupby(col1)[col2].agg(lambda x: x.eq(1).su

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

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

             if top:
                 temp = temp[0:top]

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

```
In [12]: univariate_barplots(project_data, 'school_state', 'project_is_approved', False)
```

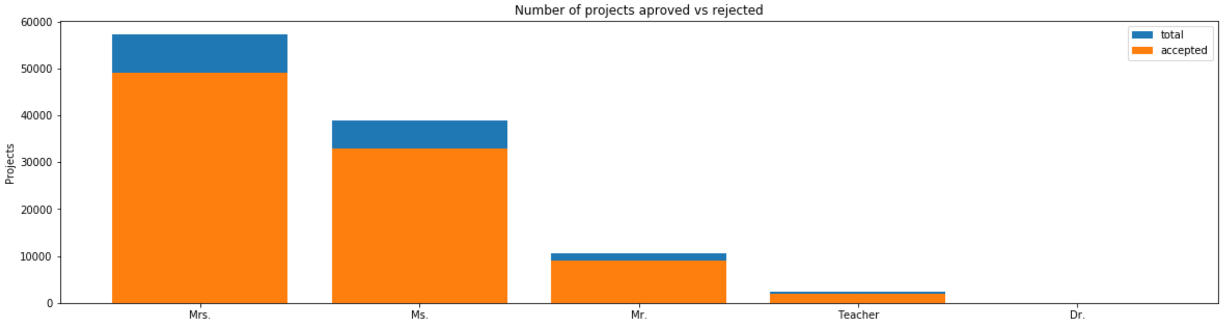


```
   school_state  project_is_approved  total       Avg
4            CA                13204  15387  0.858127
43           TX                 6014   7396  0.813142
34           NY                 6291   7318  0.859661
9            FL                 5144   6185  0.831690
27           NC                 4353   5091  0.855038
==================================================
   school_state  project_is_approved  total       Avg
39           RI                  243    285  0.852632
26           MT                  200    245  0.816327
28           ND                  127    143  0.888112
50           WY                   82     98  0.836735
46           VT                   64     80  0.800000
```

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

1.2.2 Univariate Analysis: teacher_prefix

```
In [13]: univariate_barplots(project_data, 'teacher_prefix', 'project_is_approved' , top=F
```



```
  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
==================================================
```

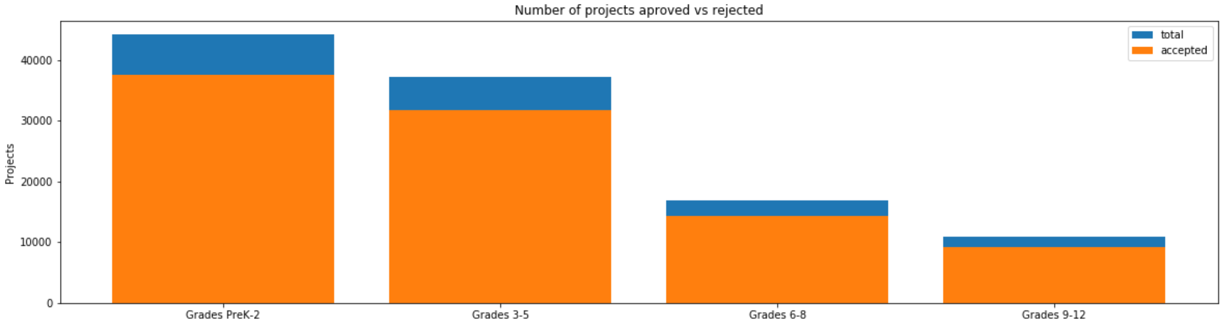
```
  teacher_prefix  project_is_approved  total       Avg
2           Mrs.                48997  57269  0.855559
3            Ms.                32860  38955  0.843537
1            Mr.                 8960  10648  0.841473
4        Teacher                 1877   2360  0.795339
0            Dr.     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

Summary:

The Project approval is high for titles Mrs, Ms, Mr almost 85%. The project approval is least for prefix having dr but for dr we don't have enough data to conclude anything as total number of submission is only 13 and out of 13, 9 got approved.
The teacher prefix Mrs has maximum number of submission and success rate.

1.2.3 Univariate Analysis: project_grade_category

In [14]:
```
univariate_barplots(project_data, 'project_grade_category', 'project_is_approved'
```



```
  project_grade_category  project_is_approved  total       Avg
3          Grades PreK-2                37536  44225  0.848751
0             Grades 3-5                31727  37135  0.854369
1             Grades 6-8                14258  16923  0.842522
2            Grades 9-12                 9182  10962  0.837621
==================================================
```

```
  project_grade_category  project_is_approved  total       Avg
3          Grades PreK-2                37536  44225  0.848751
0             Grades 3-5                31727  37135  0.854369
1             Grades 6-8                14258  16923  0.842522
2            Grades 9-12                 9182  10962  0.837621
```

Summary:

The Grades 3 to 5 has highest number of approval rate i.e. 85% and Grade 9 to 12 has lowest number of approval rate i.e. 83%. The project targeted towards kids are much likely to get approved.

1.2.4 Univariate Analysis: project_subject_categories

In [15]:
```
catogories = list(project_data['project_subject_categories'].values)
# remove special characters from list of strings python: https://stackoverflow.co

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

In [16]:

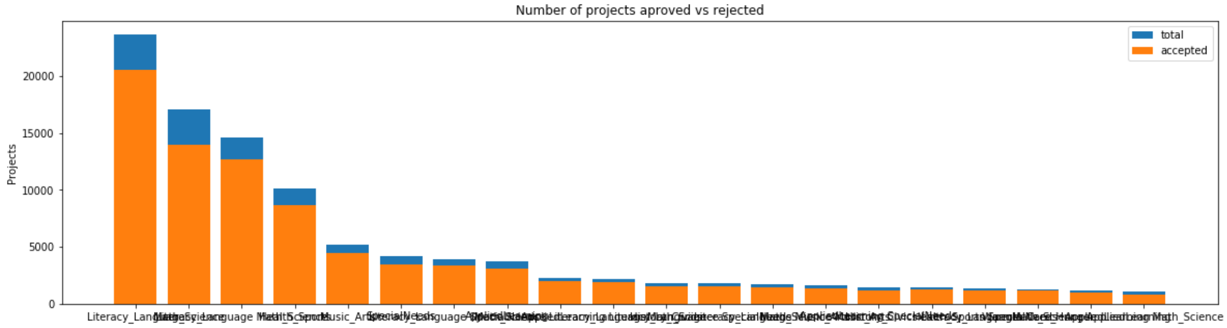
```
project_data['clean_categories'] = cat_list
project_data.drop(['project_subject_categories'], axis=1, inplace=True)
project_data.head(2)
```

Out[16]:

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

In [17]:

```
univariate_barplots(project_data, 'clean_categories', 'project_is_approved', top=
```



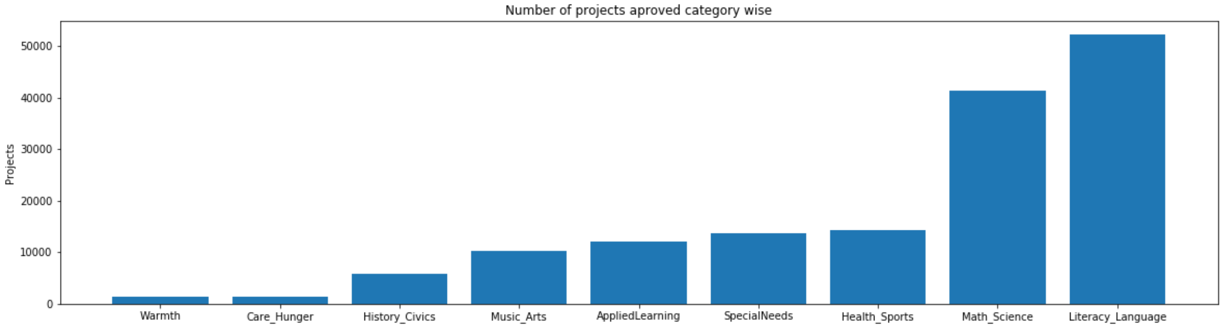
```
                  clean_categories  project_is_approved  total       Avg
24               Literacy_Language                20519  23654  0.867464
32                    Math_Science                13991  17072  0.819529
28  Literacy_Language Math_Science                12723  14634  0.869414
8                    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

The non stem Subject involving Music & Arts, Literacy & Languages has approval rate of greater than 85%, the standalone stem subjects like Math & Science has relatively lower approval rate. Also, The Project's subject involving social services keywords like care,Hunger etc has much more approval rate i.e. >90%. It can be infered that Donor's choose platform is more inclined towards arts,literature and Humanity.

In [18]:
```
# count of all the words in corpus python: https://stackoverflow.com/a/22898595/4
from collections import Counter
my_counter = Counter()
for word in project_data['clean_categories'].values:
    my_counter.update(word.split())
```

In [19]:
```
# 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('Number of projects aproved category wise') #Correction instead of % it
plt.xticks(ind, list(sorted_cat_dict.keys()))
plt.show()
```



Summary:

The most number of projects that has been submitted is from Literacy & Language and the least is from warmth category. The categories Music & Arts, Applied Learning, Special Needs and Health & sports belongs in the range of 10k to 15k sumbissions.
Clearly, we can see the below nummbers according to their project subject categories.

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

```
armth               :      1388
are_Hunger          :      1388
istory_Civics       :      5914
usic_Arts           :     10293
ppliedLearning      :     12135
pecialNeeds         :     13642
ealth_Sports        :     14223
ath_Science         :     41419
iteracy_Language    :     52236
```

1.2.5 Univariate Analysis: project_subject_subcategories

In [21]:

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

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

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

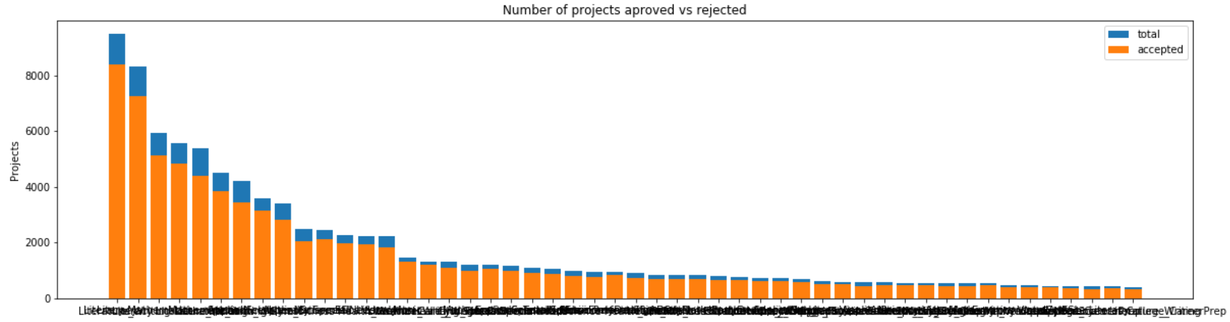
In [22]:

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

Out[22]:

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

In [23]:
```
univariate_barplots(project_data, 'clean_subcategories', 'project_is_approved', t
```



```
                     clean_subcategories  project_is_approved  total       Avg
317                             Literacy                 8371   9486  0.882458
319                 Literacy Mathematics                 7259   8324  0.872057
331  Literature_Writing Mathematics                     5139   5922  0.867781
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
```

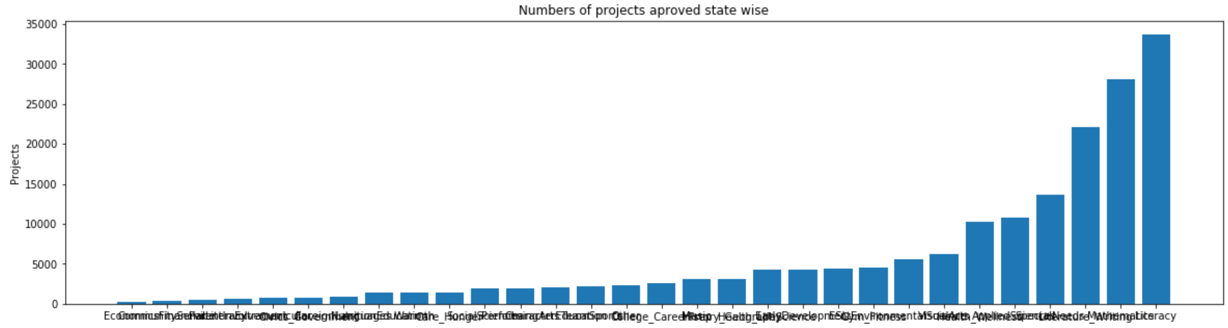
Summary

- The approval success rate is more than 80%.
- The highest approval rate is for Literacy. Also, If Literacy is tagged with another sub categories then it's approval rate is more than the standalone approval rate for sub categories. like for Literacy and Mathematics the approval rate is 87% but for mathematics it's 81%.

In [24]:
```
# count of all the words in corpus python: https://stackoverflow.com/a/22898595/4
from collections import Counter
my_counter = Counter()
for word in project_data['clean_subcategories'].values:
    my_counter.update(word.split())
```

In [25]:

```
# 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('Numbers of projects aproved state wise') #should be Number instead of
plt.xticks(ind, list(sorted_sub_cat_dict.keys()))
plt.show()
```



Summary

The number of project having subject sub category Literacy is maximum and their approval rate is also highest. The number of projects for economics is least. The Project's subject sub category involving the term Mathematics is second highest.

In [26]:
```
for i, j in sorted_sub_cat_dict.items():
    print("{:20} :{:10}".format(i,j))
```

```
conomics             :       269
ommunityService      :       441
inancialLiteracy     :       568
arentInvolvement     :       677
xtracurricular       :       810
ivics_Government     :       815
oreignLanguages      :       890
utritionEducation    :      1355
armth                :      1388
are_Hunger           :      1388
ocialSciences        :      1920
erformingArts        :      1961
haracterEducation    :      2065
eamSports            :      2192
ther                 :      2372
ollege_CareerPrep    :      2568
usic                 :      3145
istory_Geography     :      3171
ealth_LifeScience    :      4235
arlyDevelopment      :      4254
SL                   :      4367
ym_Fitness           :      4509
nvironmentalScience  :      5591
isualArts            :      6278
ealth_Wellness       :     10234
ppliedSciences       :     10816
pecialNeeds          :     13642
iterature_Writing    :     22177
athematics           :     28072
iteracy              :     33699
```

Since, Economics has least number of submission i.e. 269, But let's find out how many them have been approved

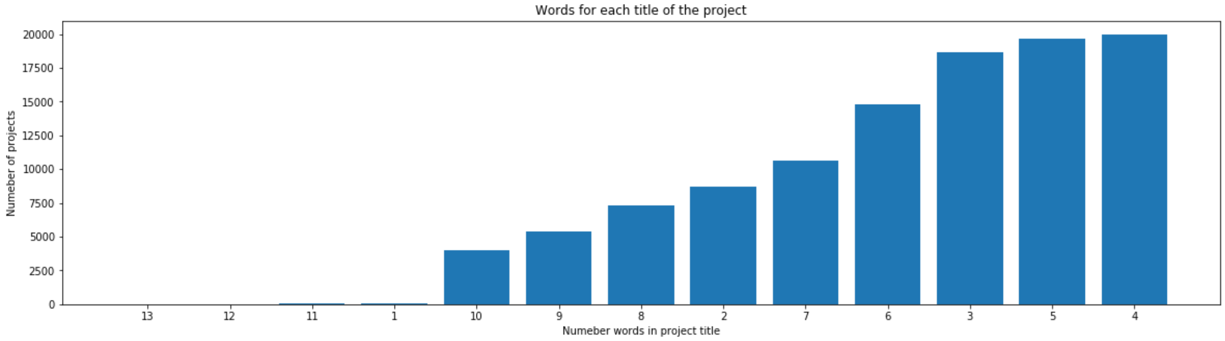
In [27]:
```
#https://stackoverflow.com/questions/8364674/how-to-count-the-number-of-true-elem
#https://stackoverflow.com/questions/19377969/combine-two-columns-of-text-in-data
np.sum(project_data[project_data['project_is_approved']==1]\
       ['clean_subcategories'].str.split().apply(lambda x : re.search(r"\bEconomi
```

Out[27]:
```
226
```

As we clearly see that the term involving Economics has 84% approval rate.

1.2.6 Univariate Analysis: Text features (Title)

In [28]:
```
#How to calculate number of words in a string in DataFrame: https://stackoverflow
word_count = project_data['project_title'].str.split().apply(len).value_counts()
word_dict = dict(word_count)
word_dict = dict(sorted(word_dict.items(), key=lambda kv: kv[1]))


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

plt.ylabel('Numeber of projects')
plt.xlabel('Numeber words in project title')
plt.title('Words for each title of the project')
plt.xticks(ind, list(word_dict.keys()))
plt.show()
```



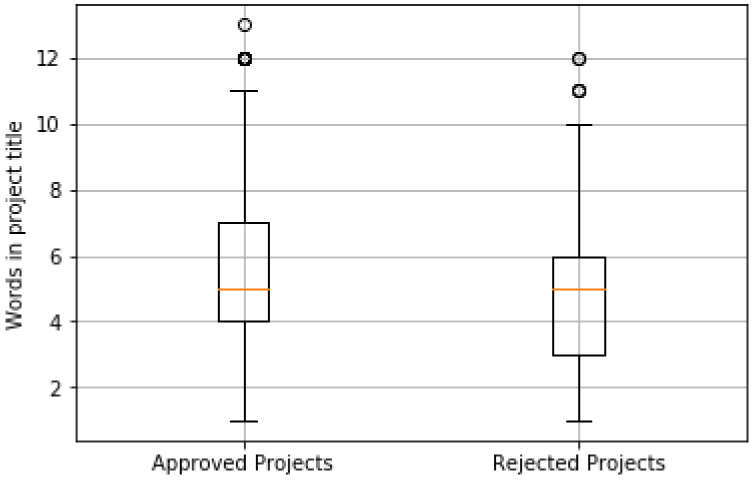
Summary

The most common length for project titles that has been submitted is 4 and least common is 10. The title having length 5 is 2nd most common among the others.

In [29]:
```
approved_title_word_count = project_data[project_data['project_is_approved']==1][
approved_title_word_count = approved_title_word_count.values

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

In [30]:

```
# 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()
```



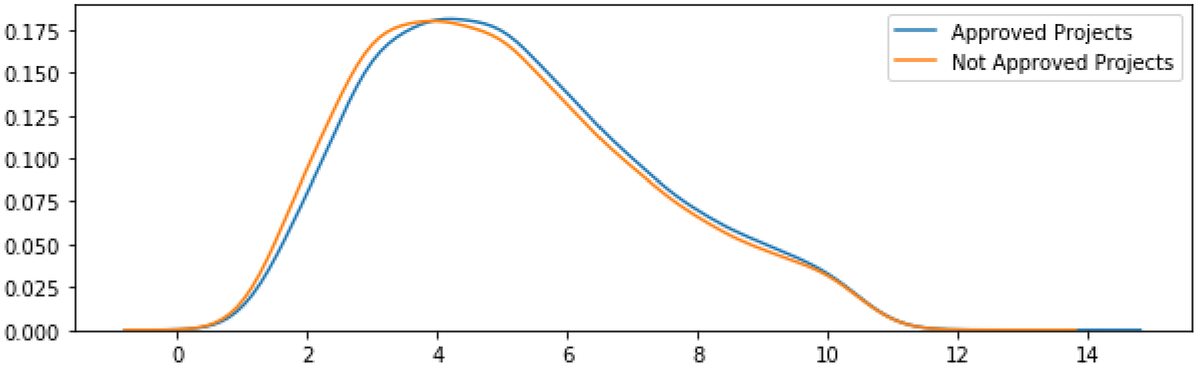
Summary

As we can from box plot that, 50th percentile line (known as median) for both decisions either approval or rejection is almost same i.e. 5.

- For Approved projects, the median (i.e. 5) is closer towards the 25th percentile that means there are few titles which has 5 words or less.
- For Approved projects, there are more titles that has words lie between lengths 5 to 7.
- For Rejected Projects, there are more titles that has words lie between lengths 3 to 5.
- For Rejected Projects, the median (i.e. 5) is closer towards the 75th percentile that means there are few titles which has 5 words or more.

In [31]:

```
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

The KDE plot for both classes is right tailed and they have same distribution. It's very hard to distinguish the class atrribute with title features

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

In [32]:

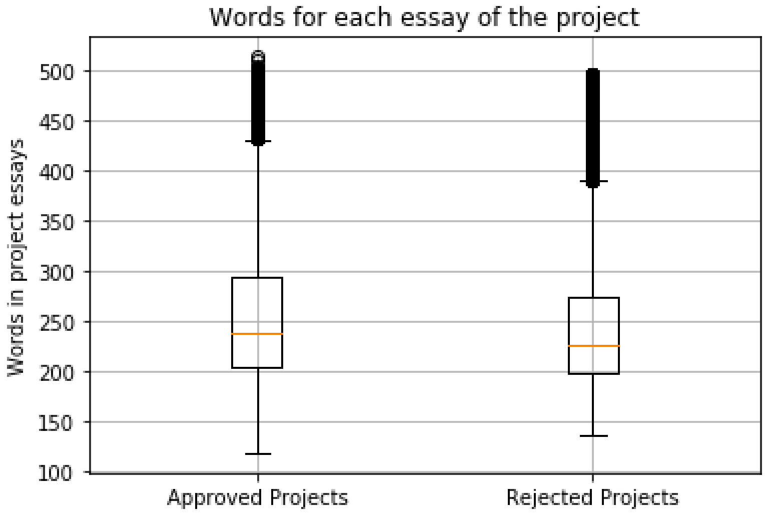
```
# merge two column text dataframe:
project_data["essay"] = project_data["project_essay_1"].map(str) +\
                        project_data["project_essay_2"].map(str) + \
                        project_data["project_essay_3"].map(str) + \
                        project_data["project_essay_4"].map(str)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

In [33]:

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

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

In [34]:

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



In [35]:

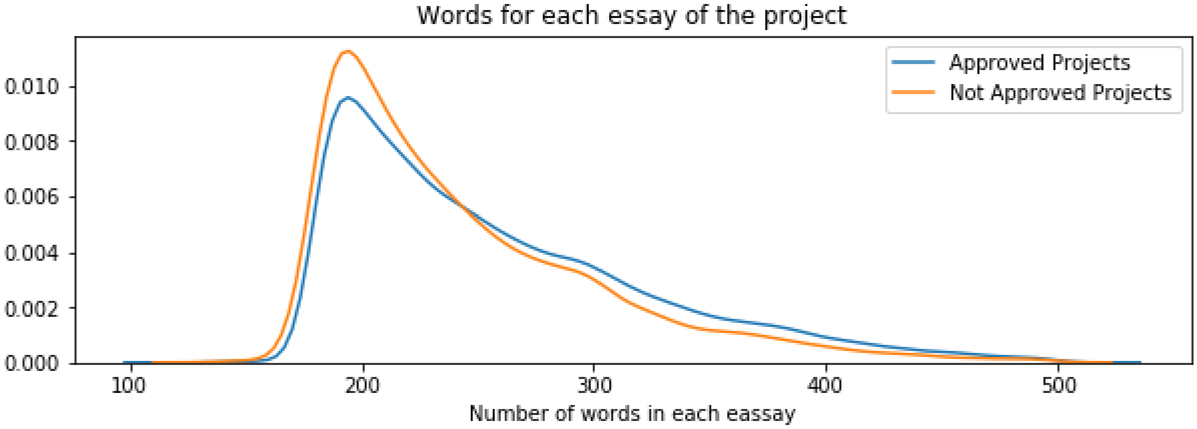
```
print("for approved ",np.median(approved_word_count))
print("for rejected",np.median(rejected_word_count))
```

```
for approved  239.0
for rejected 226.0
```

Summary

The median for approved project is 239 and for rejected project is 226. Also, We can see that there are maximum number of data points lie between 226 to near 300 for approved projects and for rejected projected it's 239 to approx 275. This IQR is also overlapping but not as of previous one and we can see this by plotting density plot as shown below.

In [36]:
```
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()
```



It's also right tailed and they are almost overlapping each other except for non approved project whose peak is taller at around 180-190 words count.

1.2.8 Univariate Analysis: Cost per project

In [37]:
```
# we get the cost of the project using resource.csv file
resource_data.head(2)
```

Out[37]:

	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 [38]:
```
# https://stackoverflow.com/questions/22407798/how-to-reset-a-dataframes-indexes-
price_data = resource_data.groupby('id').agg({'price':'sum', 'quantity':'sum'}).r
price_data.head(2)
```

Out[38]:

	id	price	quantity
0	p000001	459.56	7
1	p000002	515.89	21

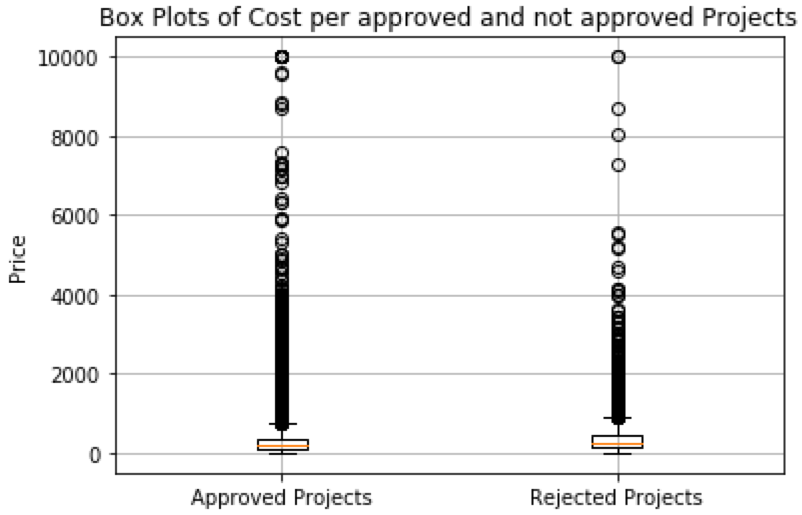
In [39]:
```
# join two dataframes in python:
project_data = pd.merge(project_data, price_data, on='id', how='left')
```

In [40]:

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

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

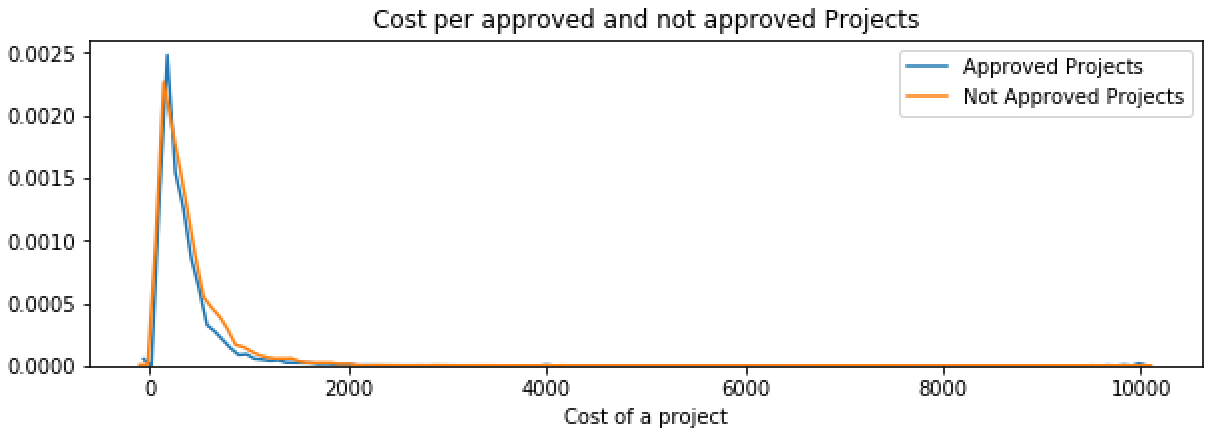
In [41]:

```
# 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 [42]:

```
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

From box plot, we can't conclude anything it's all messed up. However I can see the PDF above and can infer that blue line is almost vanishing when it's starts hitting the 10K mark for cost. It means that the higher cost is not supporting the project approval process.

In [43]:

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

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

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

for i in range(0,101,5):
    x.add_row([i,np.round(np.percentile(approved_price,i), 3), np.round(np.percen
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     |       77.374      |         118.56        |
|     25     |       99.95       |        140.892        |
|     30     |      116.672      |         162.23        |
|     35     |      137.207      |        184.014        |
|     40     |       157.0       |        208.632        |
|     45     |      178.259      |        235.106        |
|     50     |       198.99      |        263.145        |
|     55     |       223.99      |         292.61        |
|     60     |      255.598      |        325.144        |
|     65     |       285.41      |         362.39        |
|     70     |      321.222      |         399.99        |
|     75     |       366.07      |        449.945        |
|     80     |      411.666      |        519.282        |
|     85     |       479.0       |        618.276        |
|     90     |      593.082      |        739.356        |
|     95     |      801.494      |        992.486        |
|    100     |       9999.0      |         9999.0        |
+------------+-------------------+-----------------------+
```

Summary

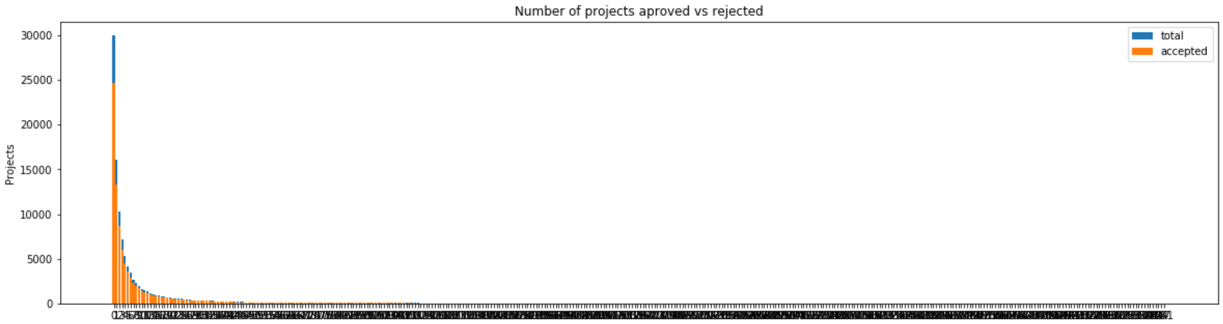
Now the percentile can easily uncover the secrets of cost vs project approval. If we are seeing the IQR range (where the majority of data lies) -

- At 25 %ile, Approved one has almost 40 units lesser than non approved projects.
- At 50 %ile, Approved one has almost 64 units lesser than non approved projects.
- At 75 %ile, Approved one has almost 83 units lesser than non approved projects.
  The above table can clearly indicate that for non approved projects costs are higher than approved projects.

1.2.9 Univariate Analysis: teacher_number_of_previously_posted_projects

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

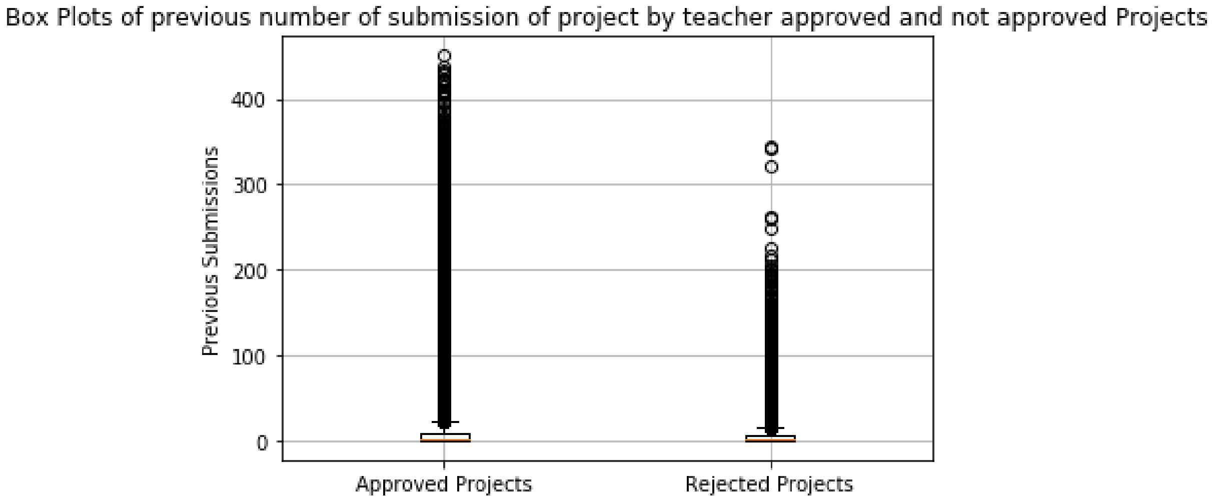
In [44]:
```
# Plotting the bar plot to to check for approval rates for this variable
univariate_barplots(project_data,'teacher_number_of_previously_posted_projects','
```



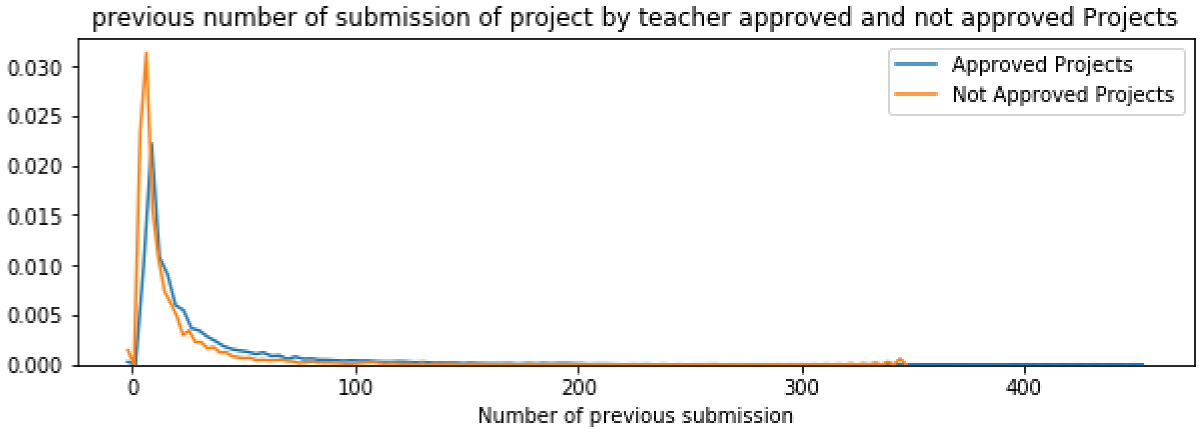
```
   teacher_number_of_previously_posted_projects  project_is_approved  total  \
0                                             0                24650  30012   
1                                             1                13328  16057   
2                                             2                 8705  10350   
3                                             3                 5997   7110   
4                                             4                 4452   5266   

        Avg  
0  0.821338  
1  0.830043  
2  0.841063  
3  0.843460  
4  0.845423  
==================================================
    teacher_number_of_previously_posted_projects  project_is_approved  total

42                                           242                    1      1
68                                           270                    1      1
34                                           234                    1      1
35                                           347                    1      1
73                                           451                    1      1

    Avg
42  1.0
68  1.0
34  1.0
35  1.0
73  1.0
```

In [45]:
```
approved_numbers = project_data[project_data['project_is_approved']==1]['teacher_

rejected_numbers = project_data[project_data['project_is_approved']==0]['teacher_
```

In [46]:
```
# https://glowingpython.blogspot.com/2012/09/boxplot-with-matplotlib.html
plt.boxplot([approved_numbers, rejected_numbers])
plt.title('Box Plots of previous number of submission of project by teacher appro
plt.xticks([1,2],('Approved Projects','Rejected Projects'))
plt.ylabel('Previous Submissions')
plt.grid()
plt.show()
```



In [47]:
```
plt.figure(figsize=(10,3))
sns.distplot(approved_numbers, hist=False, label="Approved Projects")
sns.distplot(rejected_numbers, hist=False, label="Not Approved Projects")
plt.title('previous number of submission of project by teacher approved and not a
plt.xlabel('Number of previous submission')
plt.legend()
plt.show()
```



In [48]:

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

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

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

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

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

Summary

The most frequent number of submission is the new submission i.e. approx 30K and 82% of them got approved. 450 is the maximum number of previous submission done on this platform by one teacher. The submission by same teacher for projects count above say 100 is rare. The box plot is not useful here as the data distribution is unreadable as well as unseperable for this features.As From, Percentile Table we can see that both are having same median and almost same distribution. The number of submission is not contributing anything towards the decision of approval of projects on Donors Choose platform. However, this platform looks more encouraging initially (upto 100 submissions) to teachers to post more projects.

1.2.10 Univariate Analysis: project_resource_summary

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

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

In [49]:
```
print(project_data['project_resource_summary'].values[0])
print("="*50)
print(project_data['project_resource_summary'].values[100])
print("="*50)
print(project_data['project_resource_summary'].values[150])
print("="*50)
```

```
 y students need opportunities to practice beginning reading skills in English
 t home.
==================================================
 y students need laptops that have printing abilities. I would like my students
 o have the ability to work on their projects and to print their works, researc
  and writings.
==================================================
 y students need 5 Hokki stools to increase their movement even while sitting.
==================================================
```

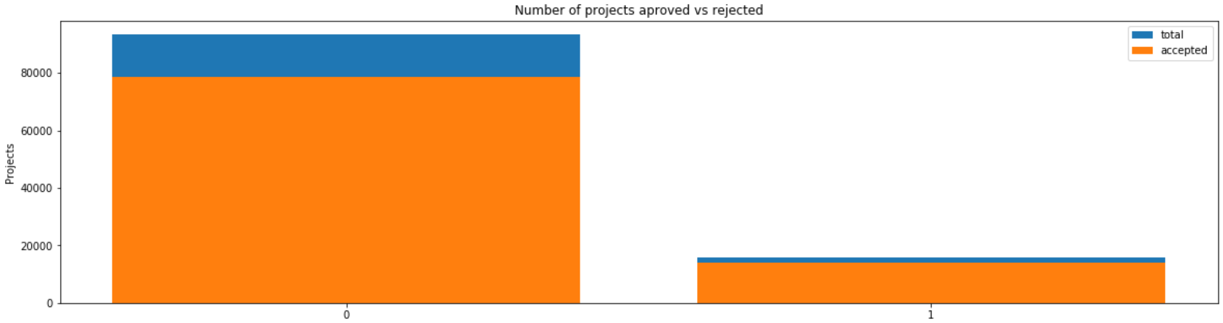
In [50]:

```
#https://stackoverflow.com/questions/34962104/pandas-how-can-i-use-the-apply-func
#https://stackoverflow.com/questions/19859282/check-if-a-string-contains-a-number
# considering digits to be non-negative
project_data["digits_in_resource_summary"] = project_data['project_resource_summa
project_data.iloc[150,]
```

Out[50]:

```
nnamed: 0
731
d
142819
eacher_id                                                       eafd3233848365
b7130b83e100434e7
eacher_prefix
s.
chool_state
O
roject_submitted_datetime                                                    2
16-09-27 23:30:32
roject_grade_category
rades 3-5
roject_title                                                     More Movement
ith Hokki Stools
roject_essay_1                                 The 51 fifth grade students tha
 will cycle th...
roject_essay_2                                 My students will use these five
rightly color...
roject_essay_3
aN
roject_essay_4
aN
roject_resource_summary                        My students need 5 Hokki stools
o increase th...
eacher_number_of_previously_posted_projects
6
roject_is_approved

lean_categories
ealth_Sports
lean_subcategories
ealth_Wellness
ssay                                           The 51 fifth grade students tha
 will cycle th...
rice
35.3
uantity

igits_in_resource_summary

ame: 150, dtype: object
```

```
In [51]: univariate_barplots(project_data,'digits_in_resource_summary','project_is_approve
```



```
   digits_in_resource_summary  project_is_approved  total       Avg
0                           0                78614  93490  0.840881
1                           1                14089  15755  0.894256
==================================================
```

```
   digits_in_resource_summary  project_is_approved  total       Avg
0                           0                78614  93490  0.840881
1                           1                14089  15755  0.894256
```

Summary

We can see that the presence of digits is merely affecting chances of approval of projects by 5%. Also, The volume of projects which doesn't have digits in their summary resource has got fair approval of almost 84%.

1.3 Text preprocessing

1.3.1 Essay Text

In [52]:

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

Out[52]:

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

2 rows × 21 columns

In [53]:
```
# 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)
```

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

```
  of the stools who are disappointed as there are not enough of them. \r\n\r\nW
  ask a lot of students to sit for 7 hours a day. The Hokki stools will be a co
 promise that allow my students to do desk work and move at the same time. Thes
  stools will help students to meet their 60 minutes a day of movement by allow
 ng them to activate their core muscles for balance while they sit. For many of
 y students, these chairs will take away the barrier that exists in schools for
  child who can't sit still.nannan
==================================================
 ow do you remember your days of school? Was it in a sterile environment with p
 ain walls, rows of desks, and a teacher in front of the room? A typical day in
 ur room is nothing like that. I work hard to create a warm inviting themed roo
  for my students look forward to coming to each day.\r\n\r\nMy class is made u
  of 28 wonderfully unique boys and girls of mixed races in Arkansas.\r\nThey a
 tend a Title I school, which means there is a high enough percentage of free a
 d reduced-price lunch to qualify. Our school is an \"open classroom\" concept,
 hich is very unique as there are no walls separating the classrooms. These 9 a
 d 10 year-old students are very eager learners; they are like sponges, absorbi
 g all the information and experiences and keep on wanting more.With these reso
 rces such as the comfy red throw pillows and the whimsical nautical hanging de
 or and the blue fish nets, I will be able to help create the mood in our class
 oom setting to be one of a themed nautical environment. Creating a classroom e
 vironment is very important in the success in each and every child's educatio
 . The nautical photo props will be used with each child as they step foot into
 ur classroom for the first time on Meet the Teacher evening. I'll take picture
  of each child with them, have them developed, and then hung in our classroom
 eady for their first day of 4th grade.  This kind gesture will set the tone be
 ore even the first day of school! The nautical thank you cards will be used th
 oughout the year by the students as they create thank you cards to their team
 roups.\r\n\r\nYour generous donations will help me to help make our classroom
  fun, inviting, learning environment from day one.\r\n\r\nIt costs lost of mon
 y out of my own pocket on resources to get our classroom ready. Please conside
  helping with this project to make our new school year a very successful one.
 hank you!nannan
==================================================
 y wonderful students are 3, 4, and 5 years old.  We are located in a small tow
  outside of Charlotte, NC.  All of my 22 students are children of school distr
 ct employees.\r\nMy students are bright, energetic, and they love to learn!  T
 ey love hands-on activities that get them moving.  Like most preschoolers, the
  enjoy music and creating different things. \r\nAll of my students come from w
 nderful families that are very supportive of our classroom.  Our parents enjoy
 atching their children's growth as much as we do!These materials will help me
 each my students all about the life cycle of a butterfly.  We will watch as th
  Painted Lady caterpillars grow bigger and build their chrysalis.  After a few
 eeks they will emerge from the chrysalis as beautiful butterflies!  We already
 ave a net for the chrysalises, but we still need the caterpillars and feeding
 tation.\r\nThis will be an unforgettable experience for my students.  My stude
 t absolutely love hands-on materials.  They learn so much from getting to touc
  and manipulate different things.  The supporting materials I have selected wi
 l help my students understand the life cycle through exploration.nannan
==================================================
 he students in my classroom are learners, readers, writers, explorers, scienti
 ts, and mathematicians! The potential in these first graders is endless! Each
 ay they come in grinning from ear-to-ear and ready to learn more. \r\nI choose
 urriculum that is real and relevant to the students, but it will also prepare
 hem for their futures. These kids are encouraged to investigate concepts that
 re exciting for them and I hope we can keep this momentum going! These kids de
 erve the best, please help me give that to them! Thank you! :)These kits inclu
```

```
e a wide variety of science, technology, engineering, and mechanics for my stu
ents to dive into at the beginning of the year. I want them to hit the ground
unning this upcoming year and these kits always encourage high interest.\r\nWh
 wouldn't want to build their own roller coaster, design a car, or even think
ritically to make a bean bag bounce as far as it can go?? These kits will also
hows students potential careers that they may have never heard of before!\r\nA
y donations would be greatly appreciated and my students will know exactly who
o thank for them!nannan
==================================================
```

In [54]:

```python
# https://stackoverflow.com/a/47091490/4084039
import re

def decontracted(phrase):
    # specific
    phrase = re.sub(r"won't", "will not", phrase)
    phrase = re.sub(r"can\'t", "can not", phrase)

    # general
    phrase = re.sub(r"n\'t", " not", phrase)
    phrase = re.sub(r"\'re", " are", phrase)
    phrase = re.sub(r"\'s", " is", phrase)
    phrase = re.sub(r"\'d", " would", phrase)
    phrase = re.sub(r"\'ll", " will", phrase)
    phrase = re.sub(r"\'t", " not", phrase)
    phrase = re.sub(r"\'ve", " have", phrase)
    phrase = re.sub(r"\'m", " am", phrase)
    return phrase
```

In [55]:

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

```
y wonderful students are 3, 4, and 5 years old.  We are located in a small tow
 outside of Charlotte, NC.  All of my 22 students are children of school distr
ct employees.\r\nMy students are bright, energetic, and they love to learn!  T
ey love hands-on activities that get them moving.  Like most preschoolers, the
 enjoy music and creating different things. \r\nAll of my students come from w
nderful families that are very supportive of our classroom.  Our parents enjoy
atching their children is growth as much as we do!These materials will help me
each my students all about the life cycle of a butterfly.  We will watch as th
 Painted Lady caterpillars grow bigger and build their chrysalis.  After a few
eeks they will emerge from the chrysalis as beautiful butterflies!  We already
ave a net for the chrysalises, but we still need the caterpillars and feeding
tation.\r\nThis will be an unforgettable experience for my students.  My stude
t absolutely love hands-on materials.  They learn so much from getting to touc
 and manipulate different things.  The supporting materials I have selected wi
l help my students understand the life cycle through exploration.nannan
==================================================
```

In [56]:

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

```
y wonderful students are 3, 4, and 5 years old.  We are located in a small tow
 outside of Charlotte, NC.  All of my 22 students are children of school distr
ct employees.  My students are bright, energetic, and they love to learn!  The
 love hands-on activities that get them moving.  Like most preschoolers, they
njoy music and creating different things.   All of my students come from wonde
ful families that are very supportive of our classroom.  Our parents enjoy wat
hing their children is growth as much as we do!These materials will help me te
ch my students all about the life cycle of a butterfly.  We will watch as the
ainted Lady caterpillars grow bigger and build their chrysalis.  After a few w
eks they will emerge from the chrysalis as beautiful butterflies!  We already
ave a net for the chrysalises, but we still need the caterpillars and feeding
tation.  This will be an unforgettable experience for my students.  My student
bsolutely love hands-on materials.  They learn so much from getting to touch a
d manipulate different things.  The supporting materials I have selected will
elp my students understand the life cycle through exploration.nannan
```

In [57]:

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

```
y wonderful students are 3 4 and 5 years old We are located in a small town ou
side of Charlotte NC All of my 22 students are children of school district emp
oyees My students are bright energetic and they love to learn They love hands
n activities that get them moving Like most preschoolers they enjoy music and
reating different things All of my students come from wonderful families that
re very supportive of our classroom Our parents enjoy watching their children
s growth as much as we do These materials will help me teach my students all a
out the life cycle of a butterfly We will watch as the Painted Lady caterpilla
s grow bigger and build their chrysalis After a few weeks they will emerge fro
 the chrysalis as beautiful butterflies We already have a net for the chrysali
es but we still need the caterpillars and feeding station This will be an unfo
gettable experience for my students My student absolutely love hands on materi
ls They learn so much from getting to touch and manipulate different things Th
 supporting materials I have selected will help my students understand the lif
 cycle through exploration nannan
```

In [58]:

```
# https://gist.github.com/sebleier/554280
# we are removing the words from the stop words list: 'no', 'nor', 'not'
stopwords= ['i', 'me', 'my', 'myself', 'we', 'our', 'ours', 'ourselves', 'you', "
            "you'll", "you'd", 'your', 'yours', 'yourself', 'yourselves', 'he', '
            'she', "she's", 'her', 'hers', 'herself', 'it', "it's", 'its', 'itsel
            'theirs', 'themselves', 'what', 'which', 'who', 'whom', 'this', 'that
            'am', 'is', 'are', 'was', 'were', 'be', 'been', 'being', 'have', 'has
            'did', 'doing', 'a', 'an', 'the', 'and', 'but', 'if', 'or', 'because'
            'at', 'by', 'for', 'with', 'about', 'against', 'between', 'into', 'th
            'above', 'below', 'to', 'from', 'up', 'down', 'in', 'out', 'on', 'off
            'then', 'once', 'here', 'there', 'when', 'where', 'why', 'how', 'all'
            'most', 'other', 'some', 'such', 'only', 'own', 'same', 'so', 'than',
            's', 't', 'can', 'will', 'just', 'don', "don't", 'should', "should've
            've', 'y', 'ain', 'aren', "aren't", 'couldn', "couldn't", 'didn', "di
            "hadn't", 'hasn', "hasn't", 'haven', "haven't", 'isn', "isn't", 'ma',
            "mustn't", 'needn', "needn't", 'shan', "shan't", 'shouldn', "shouldn'
            'won', "won't", 'wouldn', "wouldn't"]
```

In [59]:

```
# Combining all the above statemennts
# https://stackoverflow.com/questions/42212810/tqdm-in-jupyter-notebook
from tqdm import tqdm_notebook as 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())
```

```
HBox(children=(IntProgress(value=0, max=109245), HTML(value='')))
```

In [60]:

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

Out[60]:

```
'my wonderful students 3 4 5 years old we located small town outside charlotte
 c all 22 students children school district employees my students bright energe
 ic love learn they love hands activities get moving like preschoolers enjoy mu
 ic creating different things all students come wonderful families supportive c
 assroom our parents enjoy watching children growth much these materials help t
 ach students life cycle butterfly we watch painted lady caterpillars grow bigg
 r build chrysalis after weeks emerge chrysalis beautiful butterflies we alread
  net chrysalises still need caterpillars feeding station this unforgettable ex
 erience students my student absolutely love hands materials they learn much ge
 ting touch manipulate different things the supporting materials i selected hel
  students understand life cycle exploration nannan'
```

1.3.2 Project title Text

In [61]:

```
# similarly you can preprocess the titles also
print(project_data['project_title'].values[0])
print(project_data['project_title'].values[50])
print(project_data['project_title'].values[100])
print(project_data['project_title'].values[150])
```

```
Educational Support for English Learners at Home
Be Active! Be Energized!
21st Century learners, 21st century technology!
More Movement with Hokki Stools
```

In [62]:

```
# preprocessing the projec title
preprocessed_title = []
for sentance in tqdm(project_data['project_title'].values):
    sent = decontracted(sentance)
    sent = sent.replace('\\r', ' ')
    sent = sent.replace('\\"', ' ')
    sent = sent.replace('\\n', ' ')
    sent = re.sub('[^A-Za-z0-9]+', ' ', sent)
    # https://gist.github.com/sebleier/554280
    sent = ' '.join(e for e in sent.split() if e not in stopwords)
    preprocessed_title.append(sent.lower().strip())
```

```
HBox(children=(IntProgress(value=0, max=109245), HTML(value='')))
```

In [63]:

```
preprocessed_title[1000]
```

Out[63]:

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

In [64]:

```
project_data['project_title'].values[1000]
```

Out[64]:

```
'Sailing Into a Super 4th Grade Year'
```

1. 4 Preparing data for models

In [65]:

```
project_data.columns
```

Out[65]:

```
 ndex(['Unnamed: 0', 'id', 'teacher_id', 'teacher_prefix', 'school_state',
       'project_submitted_datetime', 'project_grade_category', 'project_title',
       'project_essay_1', 'project_essay_2', 'project_essay_3',
       'project_essay_4', 'project_resource_summary',
       'teacher_number_of_previously_posted_projects', 'project_is_approved',
       'clean_categories', 'clean_subcategories', 'essay', 'price', 'quantity',
       'digits_in_resource_summary'],
      dtype='object')
```

we are going to consider

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

- project_title : text data
- text : text data
- project_resource_summary: text data

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

1.4.1 Vectorizing Categorical data

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

In [66]:

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


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

```
'Warmth', 'Care_Hunger', 'History_Civics', 'Music_Arts', 'AppliedLearning', 'S
ecialNeeds', 'Health_Sports', 'Math_Science', 'Literacy_Language']
hape of matrix after one hot encodig  (109245, 9)
```

In [67]:

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


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

```
 'Economics', 'CommunityService', 'FinancialLiteracy', 'ParentInvolvement', 'Ex
 racurricular', 'Civics_Government', 'ForeignLanguages', 'NutritionEducation',
'Warmth', 'Care_Hunger', 'SocialSciences', 'PerformingArts', 'CharacterEducatio
 ', 'TeamSports', 'Other', 'College_CareerPrep', 'Music', 'History_Geography',
'Health_LifeScience', 'EarlyDevelopment', 'ESL', 'Gym_Fitness', 'EnvironmentalS
 ience', 'VisualArts', 'Health_Wellness', 'AppliedSciences', 'SpecialNeeds', 'L
 terature_Writing', 'Mathematics', 'Literacy']
 hape of matrix after one hot encodig  (109245, 30)
```

In [68]:

```
# Unique values in school state column i.e. total number of states
project_data.school_state.unique()
```

Out[68]:

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

In [69]:

```
# one hot encoding feature encoding for states
# # no need to pass the vocabulary as distinguish values are there for every cell
state_vec = CountVectorizer(lowercase=False,binary=True)
state_vec.fit(project_data['school_state'])
print(state_vec.get_feature_names())
state_one_hot = state_vec.transform(project_data['school_state'].values)
print("Shape of matrix after one hot encoding ",state_one_hot.shape)
```

```
['AK', 'AL', 'AR', 'AZ', 'CA', 'CO', 'CT', 'DC', 'DE', 'FL', 'GA', 'HI', 'IA',
'ID', 'IL', 'IN', 'KS', 'KY', 'LA', 'MA', 'MD', 'ME', 'MI', 'MN', 'MO', 'MS',
'MT', 'NC', 'ND', 'NE', 'NH', 'NJ', 'NM', 'NV', 'NY', 'OH', 'OK', 'OR', 'PA',
'RI', 'SC', 'SD', 'TN', 'TX', 'UT', 'VA', 'VT', 'WA', 'WI', 'WV', 'WY']
Shape of matrix after one hot encoding  (109245, 51)
```

In [70]:

```
# one hot encoding for teacher_prefix
# no need to pass the vocabulary as distinguish values are there for every cell
t_prefix_vec = CountVectorizer(lowercase=False,binary=True)
t_prefix_vec.fit(project_data['teacher_prefix'])
print(t_prefix_vec.get_feature_names())
t_prefix_one = t_prefix_vec.transform(project_data['teacher_prefix'].values)
print("Shape of matrix after one hot encoding is ",t_prefix_one.shape)
```

```
['Dr', 'Mr', 'Mrs', 'Ms', 'Teacher']
Shape of matrix after one hot encoding is  (109245, 5)
```

In [71]:
```
# One Hot encoding for Project Grade category
# Vocabulary need to passed otherwise It will be splitted into Grades, prek-2,num
# Hence vecotrizer needs to know what tokens to choose in vectorization
p_grade_vec = CountVectorizer(vocabulary=list(project_data.project_grade_category
p_grade_vec.fit(project_data.project_grade_category)
print(p_grade_vec.get_feature_names())
p_grade_one = p_grade_vec.transform(project_data.project_grade_category.values)
print("The shape of matrix after one hot encoding is ",p_grade_one.shape)
```

```
['Grades PreK-2', 'Grades 6-8', 'Grades 3-5', 'Grades 9-12']
The shape of matrix after one hot encoding is  (109245, 4)
```

1.4.2 Vectorizing Text data

1.4.2.1 Bag of words

In [72]:
```
# We are considering only the words which appeared in at least 10 documents(rows
vectorizer = CountVectorizer(min_df=10)
text_bow = vectorizer.fit_transform(preprocessed_essays)
print("Shape of matrix after one hot encodig ",text_bow.shape)
```

```
Shape of matrix after one hot encodig  (109245, 16623)
```

1.4.2.2 Bag of Words on `project_title`

In [73]:
```
# Vectorization of title
# We are considering only the words which appeared in at least 10 documents(rows
vectorizer = CountVectorizer(min_df=10) #min_df minimum document frequency
title_bow = vectorizer.fit_transform(preprocessed_title)
print("Shape of matrix after applying BOW on Project title ",title_bow.shape)
```

```
Shape of matrix after applying BOW on Project title  (109245, 3329)
```

1.4.2.3 TFIDF vectorizer

In [74]:
```
# # We are considering only the words which appeared in at least 10 documents(row
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  (109245, 16623)
```

1.4.2.4 TFIDF Vectorizer on `project_title`

In [75]:

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

```
Shape of matrix after one hot encodig  (109245, 3329)
```

1.4.2.5 Using Pretrained Models: Avg W2V

In [76]:

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

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

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

#  '''
```

```
Loading Glove Model

HBox(children=(IntProgress(value=1, bar_style='info', max=1), HTML(value='')))


Done. 1917495  words loaded!
```

Out[76]:

```
'# ============================\nOutput:\n    \nLoading Glove Model\n1917495it
[06:32, 4879.69it/s]\nDone. 1917495  words loaded!\n\n#  '
```

In [77]:

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

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

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

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


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

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

```
all the words in the coupus 17013963
the unique words in the coupus 58966
The number of words that are present in both glove vectors and our coupus 51501
( 87.34 %)
word 2 vec length 51501
```

In [78]:

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

In [79]:
```
# average Word2Vec
# compute average word2vec for each document in corpus.
avg_w2v_vectors = []; # the avg-w2v for each sentence/review is stored in this li
for sentence in tqdm(preprocessed_essays): # for each review/sentence
    vector = np.zeros(300) # as word vectors are of zero length
    cnt_words =0; # num of words with a valid vector in the sentence/review
    for word in sentence.split(): # for each word in a review/sentence
        if word in glove_words:
            vector += model[word]
            cnt_words += 1
    if cnt_words != 0:
        vector /= cnt_words
    avg_w2v_vectors.append(vector)

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

```
HBox(children=(IntProgress(value=0, max=109245), HTML(value='')))


109245
300
```

1.4.2.6 Using Pretrained Models: AVG W2V on `project_title`

In [80]:
```
# Similarly you can vectorize for title also
# average Word2Vec
# compute average word2vec for each document in corpus.
avg_w2v_vectors_title = []; # the avg-w2v for each sentence/review is stored in ti
for sentence in tqdm(preprocessed_title): # for each review/sentence
    vector = np.zeros(300) # as word vectors are of zero length
    cnt_words =0; # num of words with a valid vector in the sentence/review
    for word in sentence.split(): # for each word in a review/sentence
        if word in glove_words:
            vector += model[word]
            cnt_words += 1
    if cnt_words != 0:
        vector /= cnt_words
    avg_w2v_vectors_title.append(vector)

print(len(avg_w2v_vectors_title))
print(len(avg_w2v_vectors_title[0]))
```

```
HBox(children=(IntProgress(value=0, max=109245), HTML(value='')))


109245
300
```

1.4.2.7 Using Pretrained Models: TFIDF weighted W2V

In [81]:

```
# 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 [82]:

```
# average Word2Vec
# compute average word2vec for each review.
tfidf_w2v_vectors = []; # the avg-w2v for each sentence/review is stored in this
for sentence in tqdm(preprocessed_essays): # for each review/sentence
    vector = np.zeros(300) # as word vectors are of zero length
    tf_idf_weight =0; # num of words with a valid vector in the sentence/review
    for word in sentence.split(): # for each word in a review/sentence
        if (word in glove_words) and (word in tfidf_words):
            vec = model[word] # getting the vector for each word
            # here we are multiplying idf value(dictionary[word]) and the tf valu
            tf_idf = dictionary[word]*(sentence.count(word)/len(sentence.split()))
            vector += (vec * tf_idf) # calculating tfidf weighted w2v
            tf_idf_weight += tf_idf
    if tf_idf_weight != 0:
        vector /= tf_idf_weight
    tfidf_w2v_vectors.append(vector)

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

```
HBox(children=(IntProgress(value=0, max=109245), HTML(value='')))


109245
300
```

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

In [83]:

```
# tfid weighted word 2 vec for project_title column
tfidf_model_title = TfidfVectorizer()
tfidf_model_title.fit(preprocessed_title)
# we are converting a dictionary with word as a key, and the idf as a value
dictionary_title = dict(zip(tfidf_model_title.get_feature_names(), list(tfidf_mod
tfidf_words_title = set(tfidf_model_title.get_feature_names())
```

In [84]:

```
# average Word2Vec
# compute average word2vec for each review.
tfidf_w2v_vectors_title = []; # the avg-w2v for each sentence/review is stored in
for sentence in tqdm(preprocessed_title): # for each review/sentence
    vector = np.zeros(300) # as word vectors are of zero length
    tf_idf_weight =0; # num of words with a valid vector in the sentence/review
    for word in sentence.split(): # for each word in a review/sentence
        if (word in glove_words) and (word in tfidf_words_title):
            vec = model[word] # getting the vector for each word
            # here we are multiplying idf value(dictionary[word]) and the tf valu
            tf_idf = dictionary_title[word]*(sentence.count(word)/len(sentence.sp
            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_title.append(vector)

print(len(tfidf_w2v_vectors_title))
print(len(tfidf_w2v_vectors_title[0]))
```

```
HBox(children=(IntProgress(value=0, max=109245), HTML(value='')))


109245
300
```

1.4.3 Vectorizing Numerical features

In [85]:

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

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

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

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

```
Mean : 298.1152448166964, Standard deviation : 367.49642545627506
```

In [86]:

```
price_standardized
```

Out[86]:

```
array([[-0.39052147],
       [ 0.00240752],
       [ 0.5952024 ],
       ...,
       [-0.1582471 ],
       [-0.61242839],
       [-0.51215531]])
```

In [87]:

```
# we will be doing the standardization of teacher_number_of_previously_posted_pro
teacher_pp_count = StandardScaler()
teacher_pp_count.fit(project_data.teacher_number_of_previously_posted_projects.va
print(f"Mean : {teacher_pp_count.mean_[0]}, Standard deviation : {np.sqrt(teacher_

teacher_pp_count_std = teacher_pp_count.transform(project_data.teacher_number_of_
```

```
:\Anaconda\lib\site-packages\sklearn\utils\validation.py:475: DataConversionWa
ning:

ata with input dtype int64 was converted to float64 by StandardScaler.


Mean : 11.153462401025218, Standard deviation : 27.77734982798095

:\Anaconda\lib\site-packages\sklearn\utils\validation.py:475: DataConversionWa
ning:

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

In [88]:

```
teacher_pp_count_std
```

Out[88]:

```
array([[-0.40153083],
       [-0.14952695],
       [-0.36553028],
       ...,
       [-0.29352917],
       [-0.40153083],
       [-0.40153083]])
```

1.4.4 Merging all the above features

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

In [89]:

```
# shape of some encoded variables and featured vectors
print(categories_one_hot.shape)
print(sub_categories_one_hot.shape)
print(state_one_hot.shape)
print(t_prefix_one.shape)
print(text_bow.shape)
print(title_bow.shape)
print(price_standardized.shape)
print(teacher_pp_count_std.shape)
```

```
(109245, 9)
(109245, 30)
(109245, 51)
(109245, 5)
(109245, 16623)
(109245, 3329)
(109245, 1)
(109245, 1)
```

In [90]:

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

In [91]:

```
# https://stackoverflow.com/questions/11953111/numpy-how-to-filter-matrix-lines
# https://stackoverflow.com/questions/35646908/numpy-shuffle-multidimensional-arra
# https://stackoverflow.com/questions/21887754/concatenate-two-numpy-arrays-vertic
def extract_data(text_vec,n):
    x = hstack((X,text_vec))
    x = x.todense()
    x = np.c_[x,project_data.project_is_approved]
    x_pos = x[project_data.project_is_approved==1,:]
    x_neg = x[project_data.project_is_approved==0,:]
    x_pos = x_pos[:n,:] #filtering 3000 positive reviews
    x_neg = x_neg[:n,:] #filtering out 3000 negative reviews
    print("Shape of text vectors ",x.shape)
    print("Shape of positive reviews matrix ",x_pos.shape)
    print("Shape of negative reviews matrix ",x_neg.shape)
    concat_mat = np.vstack((x_pos,x_neg)) # concatenating postive and negative re
    print("Shape of concatenated matrix ",concat_mat.shape)
    # to shuffle the psoitive and negative reviews
    np.random.shuffle(concat_mat) #inplace operation return none
    return concat_mat
```

In [92]:
```
# for bag of words
# 6000 reviews
X_bow_title = extract_data(text_vec=title_bow,n=3000)
X_bow_title.shape
```

```
Shape of text vectors  (109245, 3427)
Shape of positive reviews matrix  (3000, 3427)
Shape of negative reviews matrix  (3000, 3427)
Shape of concatenated matrix  (6000, 3427)
```

Out[92]:
```
(6000, 3427)
```

In [93]:
```
# for tfidf
# 6000 reviews
X_tfidf_title = extract_data(text_vec=title_tfidf,n=3000)
X_tfidf_title.shape
```

```
Shape of text vectors  (109245, 3427)
Shape of positive reviews matrix  (3000, 3427)
Shape of negative reviews matrix  (3000, 3427)
Shape of concatenated matrix  (6000, 3427)
```

Out[93]:
```
(6000, 3427)
```

In [94]:
```
# for average word 2 vec
# 6000 reviews
X_avg_w2v_title = extract_data(text_vec=np.array(avg_w2v_vectors_title),n=3000)
X_avg_w2v_title.shape
```

```
Shape of text vectors  (109245, 398)
Shape of positive reviews matrix  (3000, 398)
Shape of negative reviews matrix  (3000, 398)
Shape of concatenated matrix  (6000, 398)
```

Out[94]:
```
(6000, 398)
```

In [95]:
```
# Need to convet to matrix first for Word 2 vec as we stored in 300 dimension lis
# for tfidf word 2 vec
# 6000 reviews
X_tfidf_w2v_title = extract_data(text_vec=np.array(tfidf_w2v_vectors_title),n=300
X_tfidf_w2v_title.shape
```

```
Shape of text vectors  (109245, 398)
Shape of positive reviews matrix  (3000, 398)
Shape of negative reviews matrix  (3000, 398)
Shape of concatenated matrix  (6000, 398)
```

Out[95]:
```
(6000, 398)
```

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.

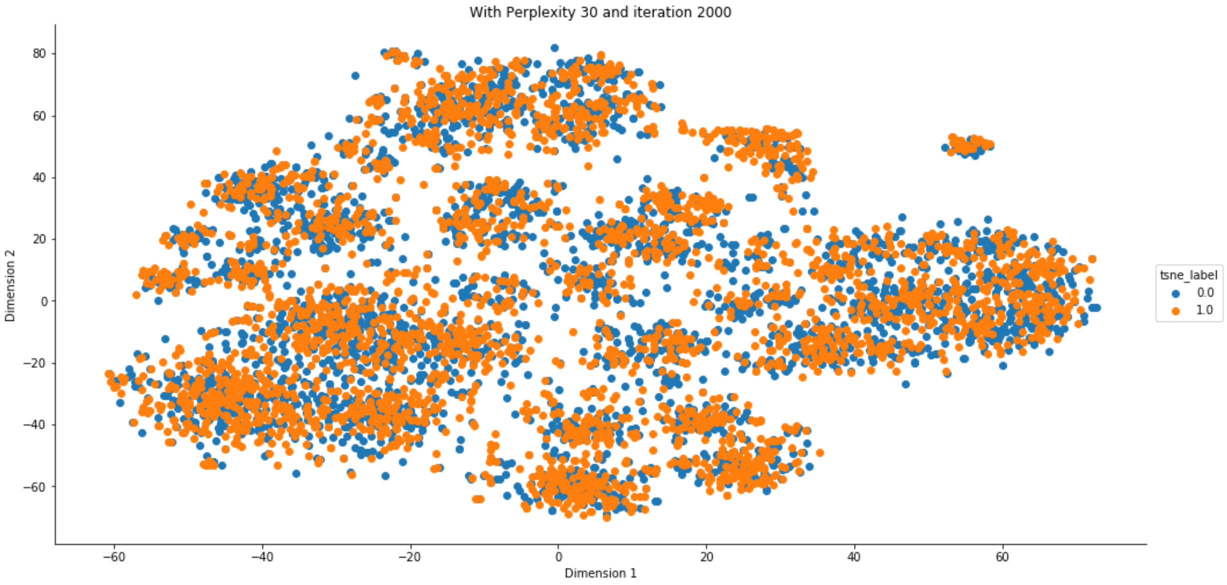
1. In the above cells we have plotted and analyzed many features. Please observe the plots and write the observations in markdown cells below every plot.
2. EDA: Please complete the analysis of the feature: teacher_number_of_previously_posted_projects
3. Build the data matrix using these features
   - 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_title : text data (BOW, TFIDF, AVG W2V, TFIDF W2V)
   - price : numerical
   - teacher_number_of_previously_posted_projects : numerical
4. Now, plot FOUR t-SNE plots with each of these feature sets.
   A. categorical, numerical features + project_title(BOW)
   B. categorical, numerical features + project_title(TFIDF)
   C. categorical, numerical features + project_title(AVG W2V)
   D. categorical, numerical features + project_title(TFIDF W2V)
5. Concatenate all the features and Apply TNSE on the final data matrix
6. Note 1: The TSNE accepts only dense matrices
7. Note 2: Consider only 5k to 6k data points to avoid memory issues. If you run into memory error issues, reduce the number of data points but clearly state the number of datat-poins you are using

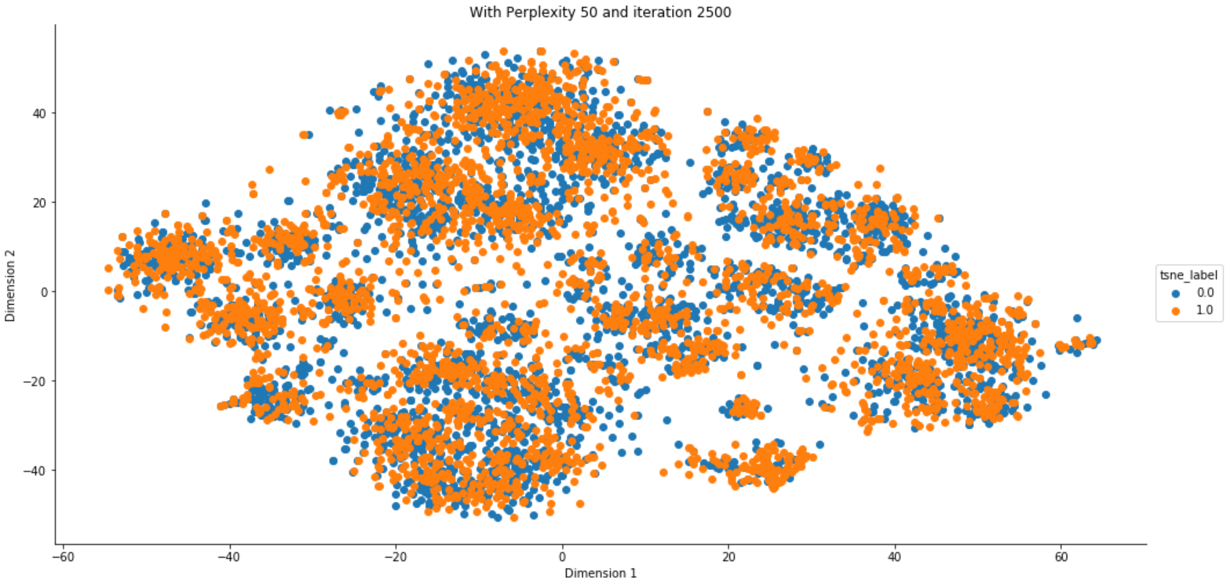
2.1 TSNE with `BOW` encoding of `project_title` feature

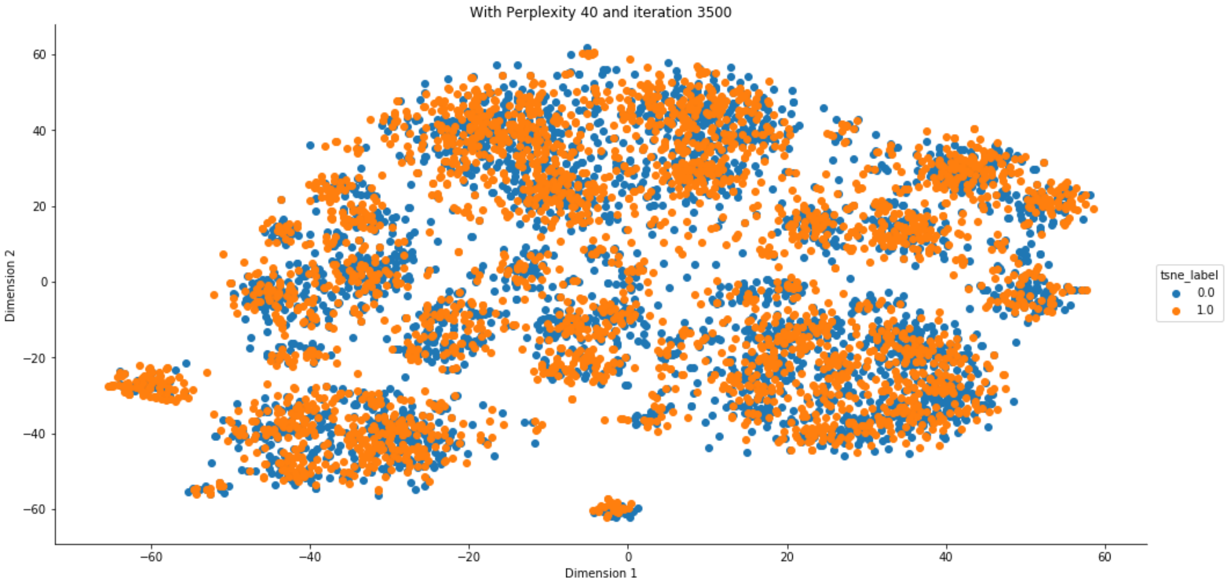
In [96]:

```
#from MulticoreTSNE import MulticoreTSNE as TSNE
from sklearn.manifold import TSNE
# https://github.com/DmitryUlyanov/Multicore-TSNE

def plot_tsne(param,tsne_data,tsne_label):
    """
    <Doc String>
    param will be perplexity and n_iter passed as list of tuple
    tsne_data is the input data matrix
    tsne_label is the class attribute

    """
    for p,i in param:
        if p is not None and i is not  None:
            model = TSNE(n_components=2, random_state =0, perplexity =p, n_iter=i
            #random state is 0 before tsne is an probablistic Algorithm

            # n_components is no of dimension it should be reduced to.
            # configuring the parameteres
            # the number of components = 2
            # default perplexity = 30
            # default learning rate = 200
            # default Maximum number of iterations for the optimization = 1000


            tsne_fitted_data  = model.fit_transform(tsne_data)
            # Visualization
            #concat the fitted data with their corresponding labels
            tsne_fitted_data = np.c_[tsne_fitted_data,tsne_label]
            tsne_df = pd.DataFrame(data = tsne_fitted_data,columns =["Dimension 1
            sns.FacetGrid(data = tsne_df,hue='tsne_label',size = 7,aspect=2)\
            .map(plt.scatter,"Dimension 1","Dimension 2").add_legend()
            plt.title("With Perplexity {0} and iteration {1}".format(p,i))
            plt.show()
```

```
In [97]: # Implementation of TSNE with BOW encoding of project_title feature

         ip_data = X_bow_title[:,:X_bow_title.shape[1]-1]
         label = X_bow_title[:,-1]
         trial_tsne_values = [(30,2000),(50,2500),(40,3500)] #(perplexity,n_iter)
         # executing TSNE and plotting with trial tsne values
         plot_tsne(param=trial_tsne_values,tsne_data=ip_data,tsne_label=label)
```


With Perplexity 30 and iteration 2000

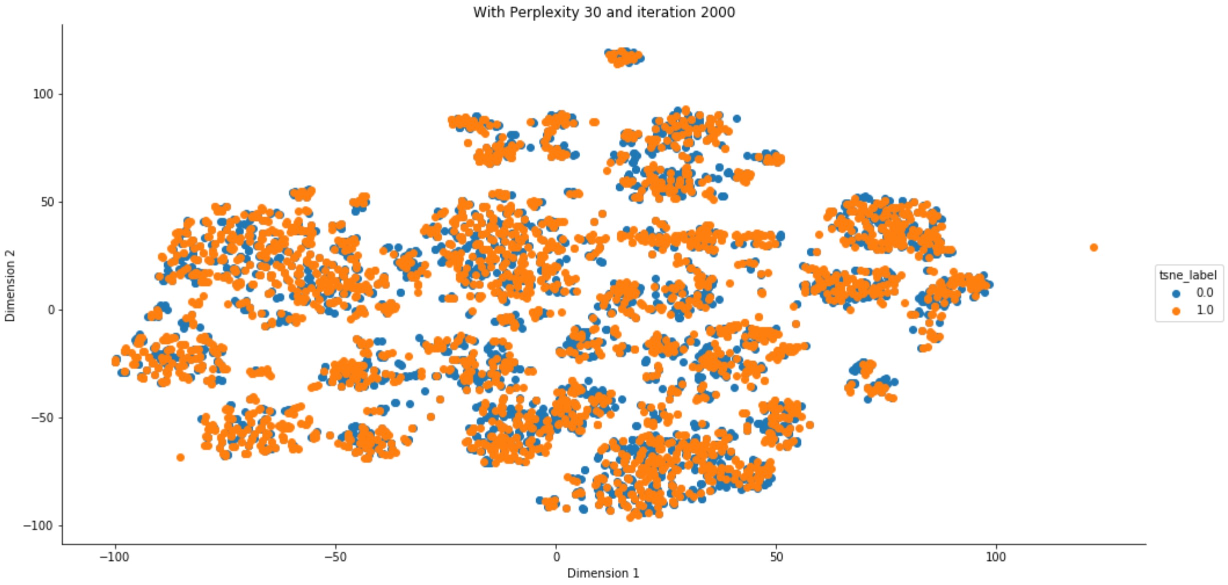

With Perplexity 50 and iteration 2500

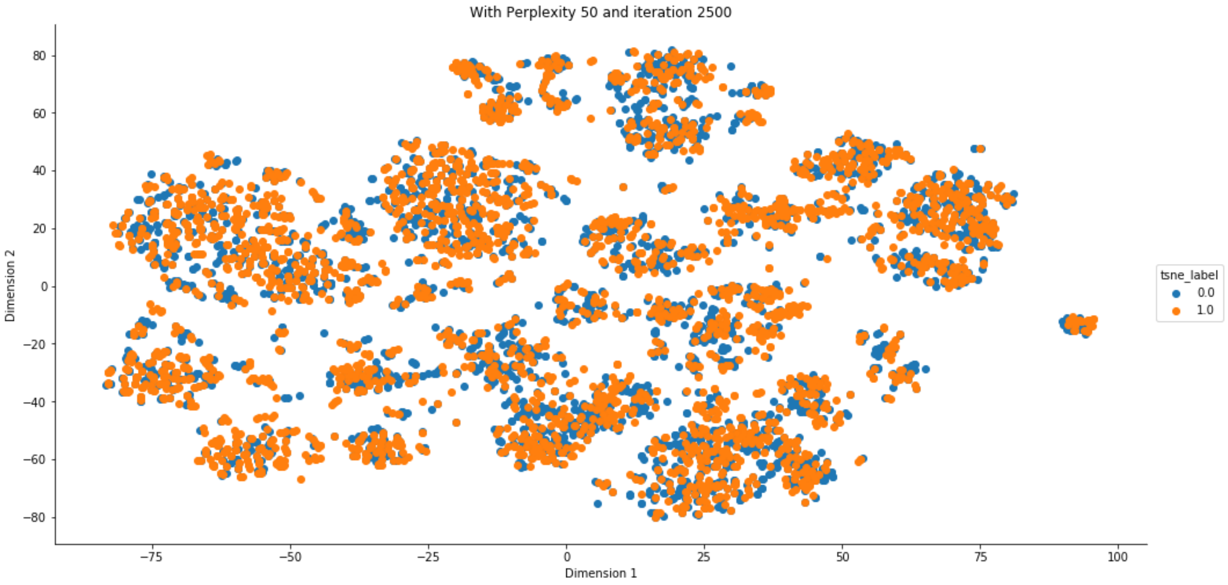


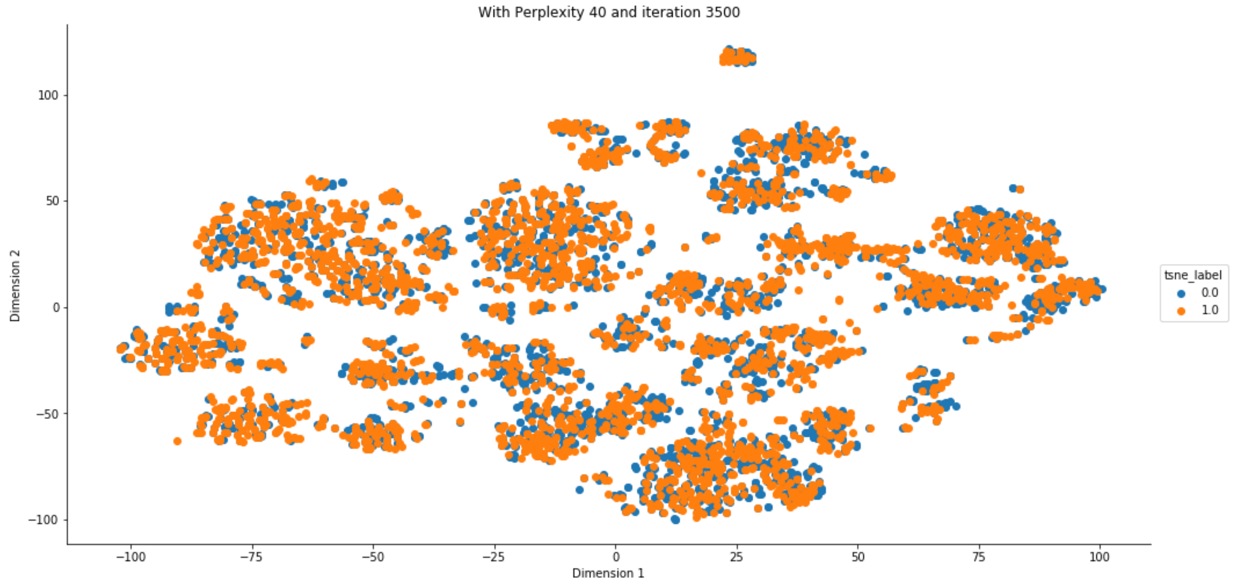
2.2 TSNE with `TFIDF` encoding of `project_title` feature

In [98]:

```
# Implementation of TSNE with TFIDF encoding of project_title feature

ip_data = X_tfidf_title[:,:X_tfidf_title.shape[1]-1]
label = X_tfidf_title[:,-1]
trial_tsne_values = [(30,2000),(50,2500),(40,3500)] #(perplexity,n_iter)
# executing TSNE and plotting with trial tsne values
plot_tsne(param=trial_tsne_values,tsne_data=ip_data,tsne_label=label)
```

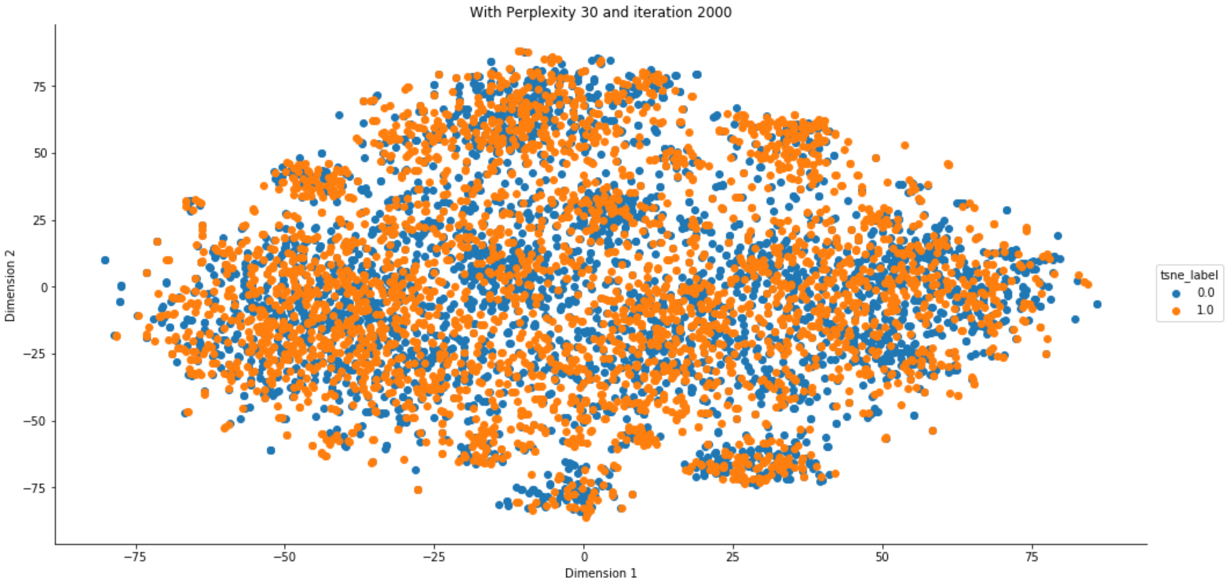


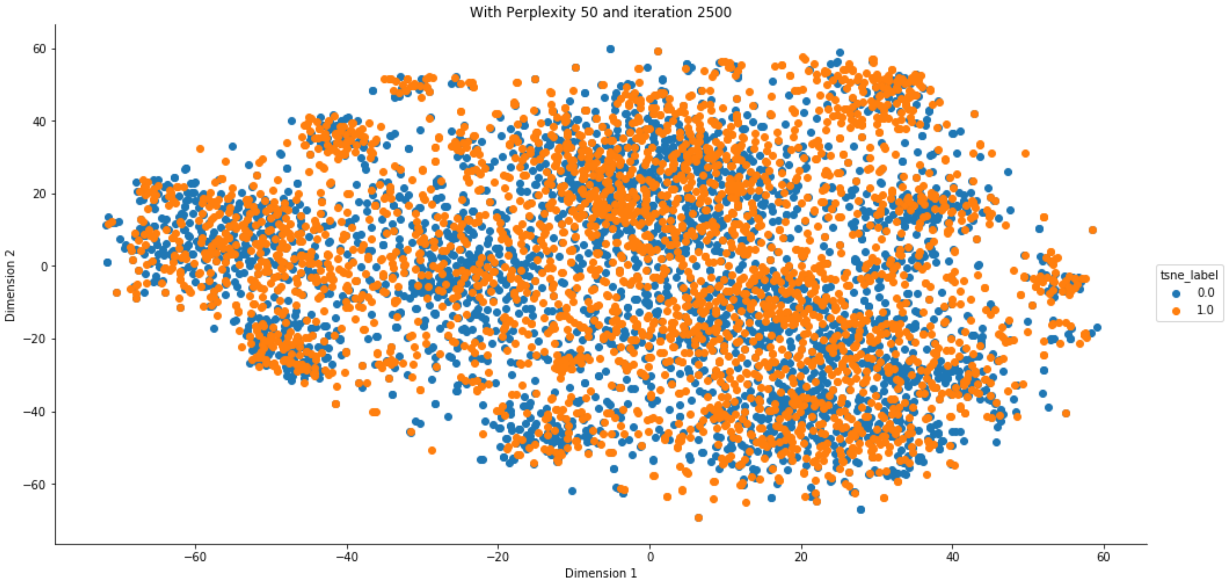


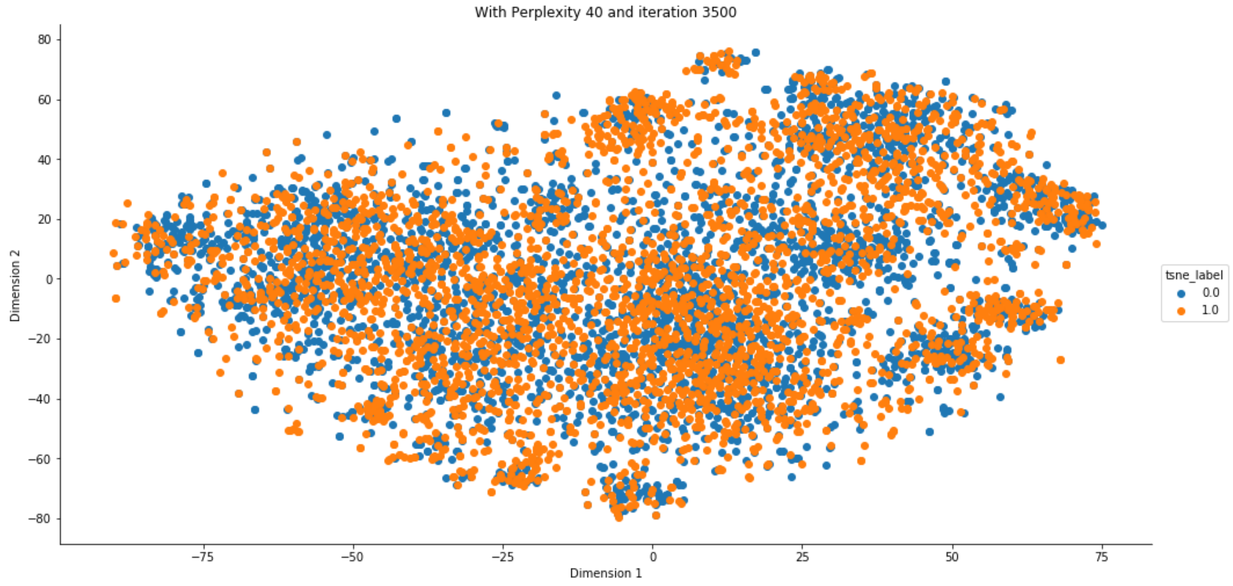


2.3 TSNE with `AVG W2V` encoding of `project_title` feature

```
In [99]: # Implementation of TSNE with Avg W2V encoding of project_title feature

         ip_data = X_avg_w2v_title[:,:X_avg_w2v_title.shape[1]-1]
         label = X_avg_w2v_title[:,-1]
         trial_tsne_values = [(30,2000),(50,2500),(40,3500)] #(perplexity,n_iter)
         # executing TSNE and plotting with trial tsne values
         plot_tsne(param=trial_tsne_values,tsne_data=ip_data,tsne_label=label)
```



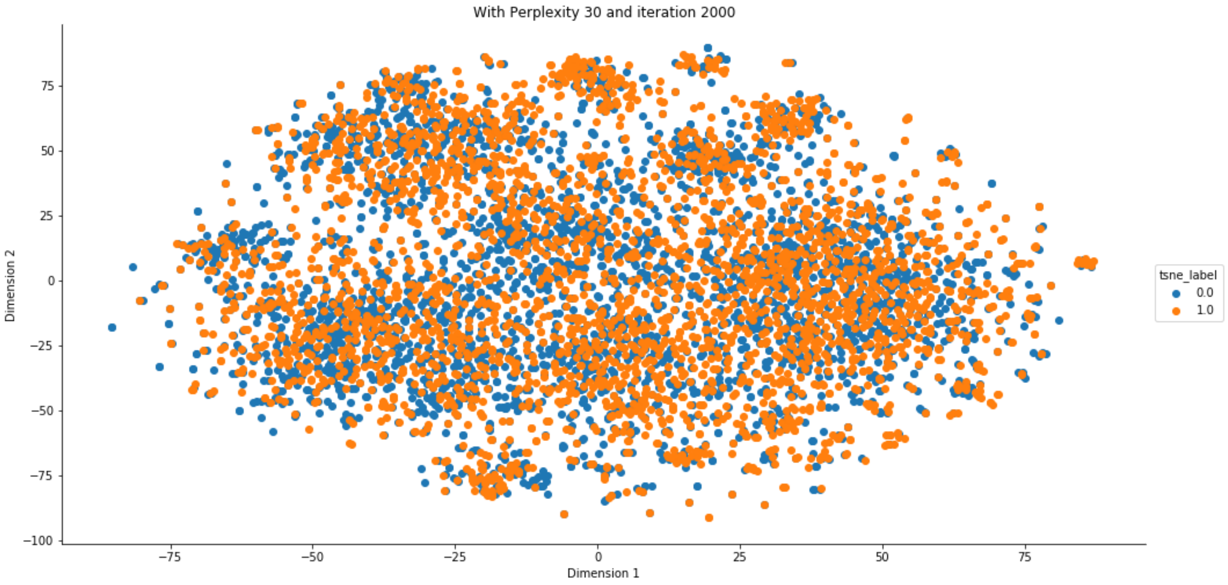


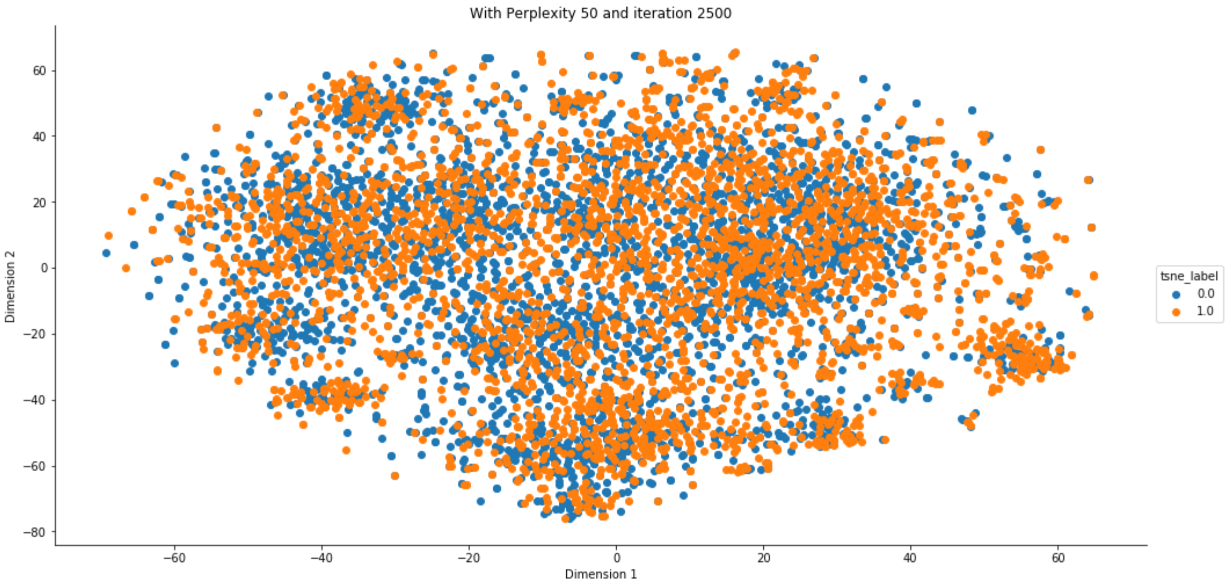


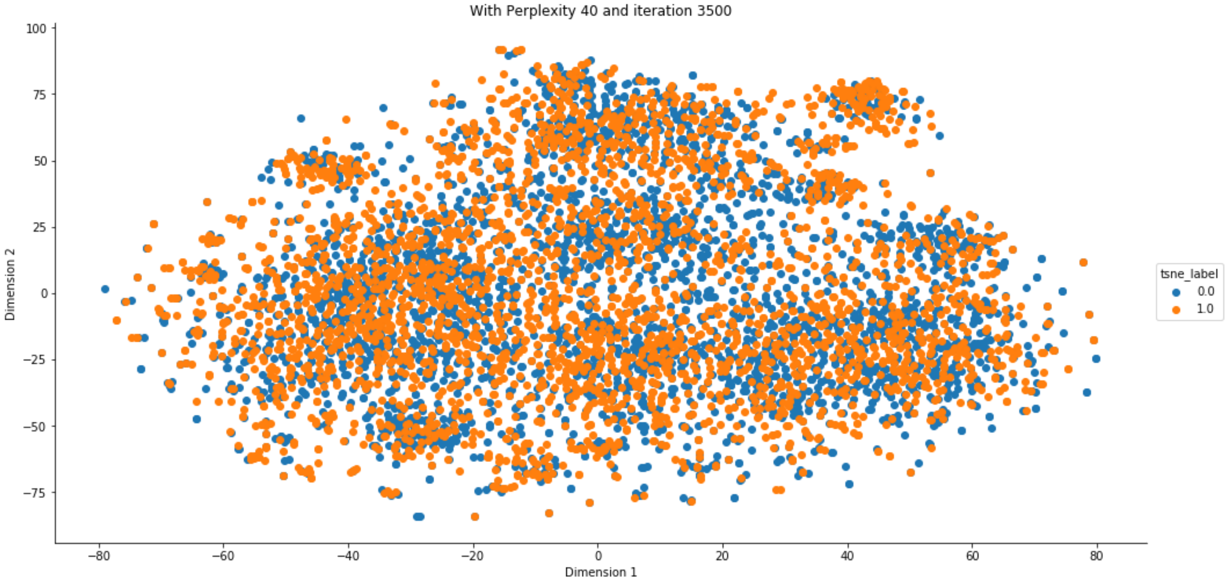
2.4 TSNE with `TFIDF Weighted W2V` encoding of `project_title` feature

In [100]:

```
# Implementation of TSNE with TFIDF Weighted W2V encoding of project_title featur

ip_data = X_tfidf_w2v_title[:,:X_tfidf_w2v_title.shape[1]-1]
label = X_tfidf_w2v_title[:,-1]
trial_tsne_values = [(30,2000),(50,2500),(40,3500)] #(perplexity,n_iter)
# executing TSNE and plotting with trial tsne values
plot_tsne(param=trial_tsne_values,tsne_data=ip_data,tsne_label=label)
```







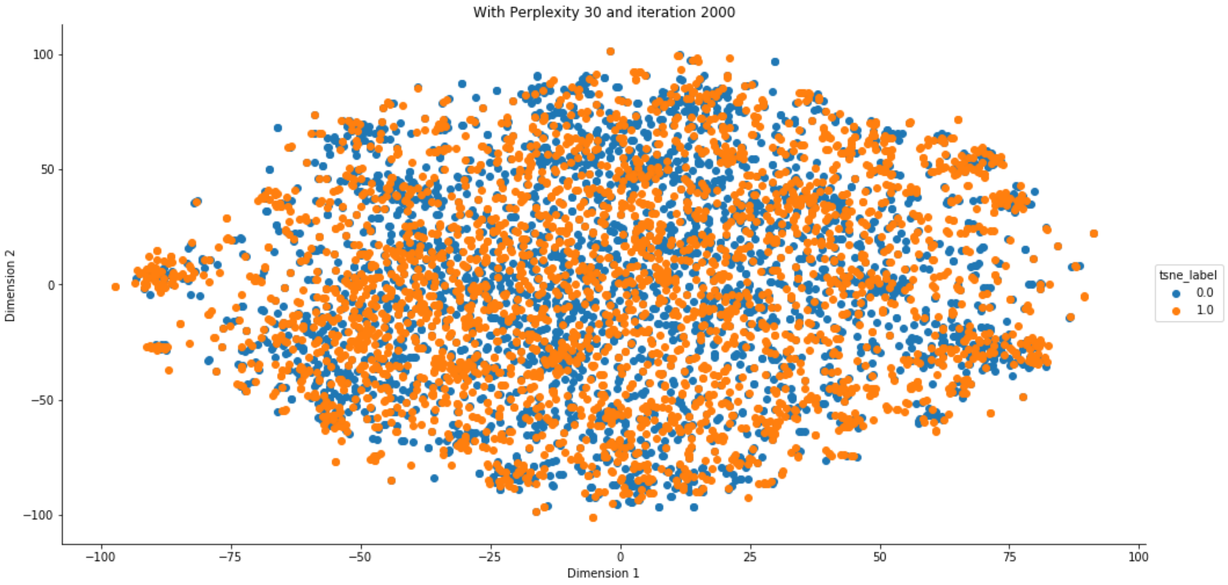
TSNE with all feature combined encoding of `project_title` feature

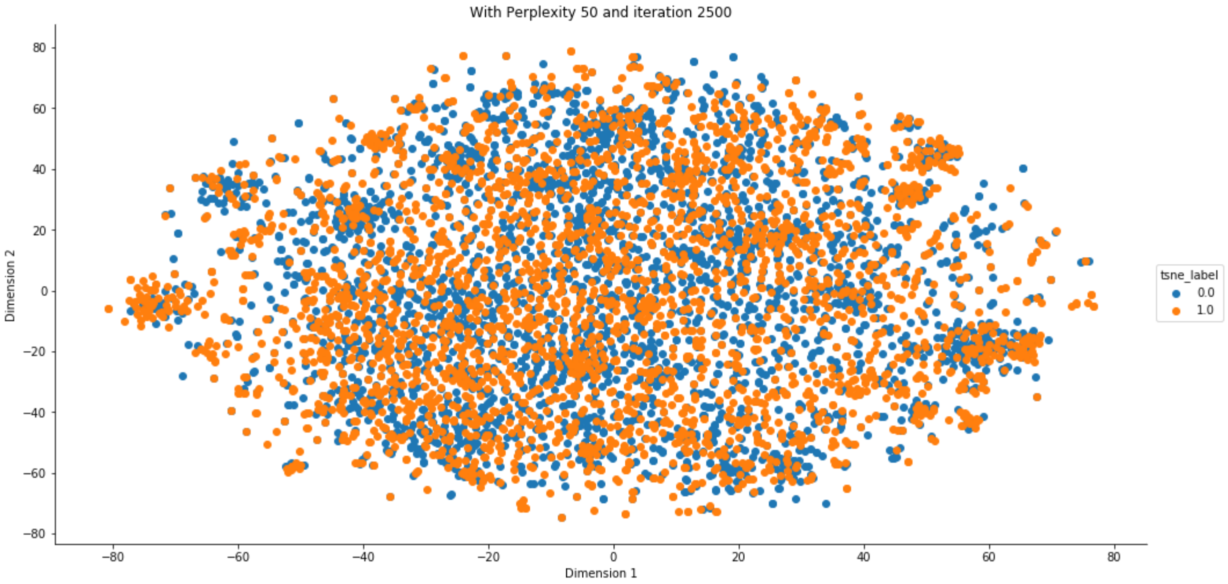
In [101]:
```
count=3000 #no of data points
x_all = hstack((X,title_bow,title_tfidf,np.array(avg_w2v_vectors_title),np.array(
x_all = x_all.todense()
x_all = np.c_[x_all,project_data.project_is_approved]
x_pos = x_all[project_data.project_is_approved==1,:]
x_neg = x_all[project_data.project_is_approved==0,:]
x_pos = x_pos[:count,:]
x_neg = x_neg[:count,:]
print("Shape of text vectors ",x_all.shape)
print("Shape of positive reviews matrix ",x_pos.shape)
print("Shape of negative reviews matrix ",x_neg.shape)
x_all_mat = np.vstack((x_pos,x_neg))
print("Shape of concatenated matrix ",x_all_mat.shape)
np.random.shuffle(x_all_mat) #inplace operation return none
```

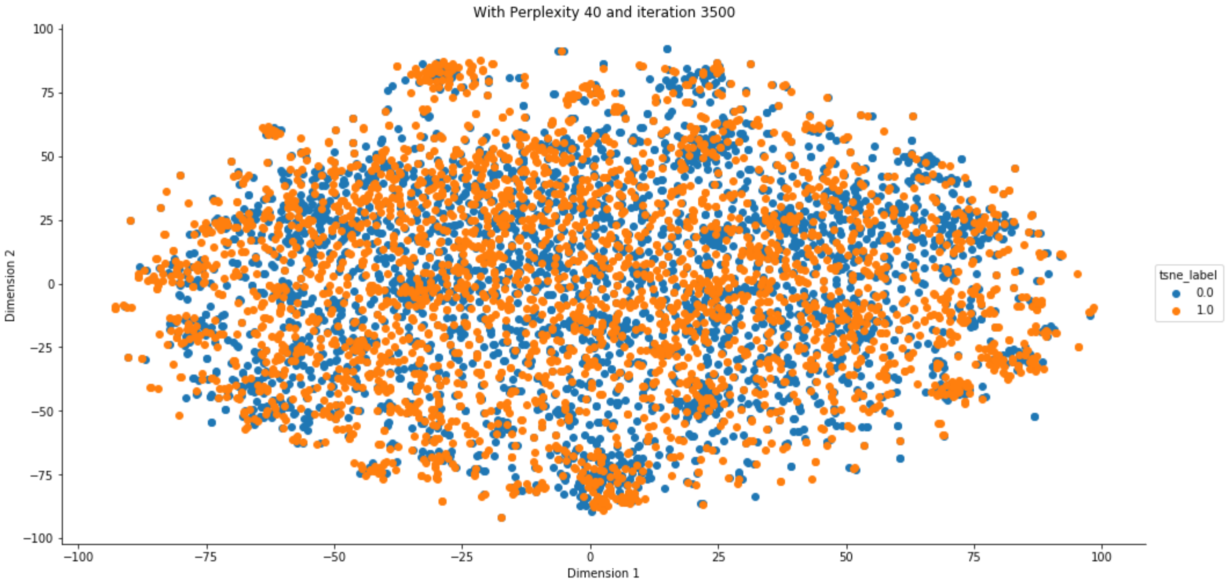
```
Shape of text vectors  (109245, 7356)
Shape of positive reviews matrix  (3000, 7356)
Shape of negative reviews matrix  (3000, 7356)
Shape of concatenated matrix  (6000, 7356)
```

In [103]:

```
# Implementation of TSNE with all features(BOW,TFIDF,Avg W2V, TFIDF W2V) encoding

ip_data = x_all_mat[:,:x_all.shape[1]-1]
label = x_all_mat[:,-1]
trial_tsne_values = [(30,2000),(50,2500),(40,3500)] #(perplexity,n_iter)
# executing TSNE and plotting with trial tsne values
plot_tsne(param=trial_tsne_values,tsne_data=ip_data,tsne_label=label)
```







2.5 Summary

- The features are not seperable in hyper dimension also. As they are inseparable in 2D embedded space.
- There are almost no variation of shape and seperation at these hyper parameters setting.