DonorsChoose

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

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

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

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

About the DonorsChoose Data Set

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

Feature	Description
`project_id`	**A unique identifier for the proposed project. Example:** `p036502`
`project_title`	**Title of the project. Examples:** • `Art Will Make You Happy!` • `First Grade Fun`
`project_grade_category`	**Grade level of students for which the project is targeted. One of the following enumerated values:** • `Grades PreK-2` • `Grades 3-5` • `Grades 6-8` • `Grades 9-12`
`project_subject_categories`	**One or more (comma-separated) subject categories for the project from the following enumerated list of values:** • `Applied Learning` • `Care & Hunger` • `Health & Sports` • `History & Civics` • `Literacy & Language` • `Math & Science` • `Music & The Arts` • `Special Needs` • `Warmth` **Examples:** • `Music & The Arts` • `Literacy & Language, Math & Science`
`school_state`	**State where school is located (Two-letter U.S. postal code). Example:** `WY`
`project_subject_subcategories`	**One or more (comma-separated) subject subcategories for the project. Examples:** • `Literacy`

Feature	Description
	• `Literature & Writing, Social Sciences`
`project_resource_summary`	**An explanation of the resources needed for the project. Example:** • `My students need hands on literacy materials to manage sensory needs!`
`project_essay_1`	**First application essay***
`project_essay_2`	**Second application essay***
`project_essay_3`	**Third application essay***
`project_essay_4`	**Fourth application essay***
`project_submitted_datetime`	**Datetime when project application was submitted. Example:** `2016-04-28 12:43:56.245`
`teacher_id`	**A unique identifier for the teacher of the proposed project. Example:** `bdf8baa8fedef6bfeec7ae4ff1c15c56`
`teacher_prefix`	**Teacher's title. One of the following enumerated values:** • `nan` • `Dr.` • `Mr.` • `Mrs.` • `Ms.` • `Teacher.`
`teacher_number_of_previously_posted_projects`	**Number of project applications previously submitted by the same teacher. Example:** `2`

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

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

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

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

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

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

Notes on the Essay Data

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

- **__project_essay_1:__ "Introduce us to your classroom"**
- **__project_essay_2:__ "Tell us more about your students"**
- **__project_essay_3:__ "Describe how your students will use the materials you're requesting"**
- **__project_essay_4:__ "Close by sharing why your project will make a difference"**

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

- **__project_essay_1:__ "Describe your students: What makes your students special? Specific details about their background, your neighborhood, and your school are all helpful."**
- **__project_essay_2:__ "About your project: How will these materials make a difference in your students'**

- __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 [0]:

```
%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
```

Output hidden; open in https://colab.research.google.com to view.

1.1 Reading Data

In [0]:

```
import tensorflow as tf
device_name = tf.test.gpu_device_name()
if device_name != '/device:GPU:0':
  raise SystemError('GPU device not found')
print('Found GPU at: {}'.format(device_name))
```

```
Found GPU at: /device:GPU:0
```

In [0]:

```
#importing dataset from googledrive
from google.colab import drive
drive.mount('/content/gdrive')
```

```
Go to this URL in a browser: https://accounts.google.com/o/oauth2/auth?client_id=94731898
9803-6bn6qk8qdgf4n4g3pfee6491hc0brc4i.apps.googleusercontent.com&redirect_uri=urn%3Aietf%
```

```
3Awg%3Aoauth%3A2.0%3Aoob&scope=email%20https%3A%2F%2Fwww.googleapis.com%2Fauth%2Fdocs.test%20https%3A%2F%2Fwww.googleapis.com%2Fauth%2Fdrive%20https%3A%2F%2Fwww.googleapis.com%2Fauth%2Fdrive.photos.readonly%20https%3A%2F%2Fwww.googleapis.com%2Fauth%2Fpeopleapi.readonly&response_type=code

Enter your authorization code:
··········
Mounted at /content/gdrive
```

In [0]:

```
!ls 'gdrive/My Drive/Machine Learning/Assignmnets/Assignment2/Assignments_DonorsChoose_2018'
```

```
'06 Implement SGD.ipynb'       confusion_matrix.png
 10_DonorsChoose_Clustering.ipynb     cooc.JPG
 11_DonorsChoose_TruncatedSVD.ipynb   glove.42B.300d.zip
 2_DonorsChoose_EDA_TSNE.ipynb       glove_vectors
 2letterstabbrev.pdf         haberman.csv
 3_DonorsChoose_KNN.ipynb        haberman.gsheet
 3d_plot.JPG          heat_map.JPG
 3d_scatter_plot.ipynb         imdb.txt
 4_DonorsChoose_NB.ipynb        resources.csv
 5_DonorsChoose_LR.ipynb        response.JPG
 7_DonorsChoose_SVM.ipynb        summary.JPG
 8_DonorsChoose_DT.ipynb        test_data.csv
 9_DonorsChoose_RF_GBDT.ipynb       train_cv_auc.JPG
 Assignment_SAMPLE_SOLUTION.ipynb     train_data.csv
 Assignment_tips.docx         train_test_auc.JPG
 Assignment_tips.gdoc
```

In [0]:

```
project_data = pd.read_csv('gdrive/My Drive/Machine Learning/Assignmnets/Assignment2/Assignments_DonorsChoose_2018/train_data.csv')
resource_data = pd.read_csv('gdrive/My Drive/Machine Learning/Assignmnets/Assignment2/Assignments_DonorsChoose_2018/resources.csv')
```

In [0]:

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

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

In [0]:

```
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[0]:

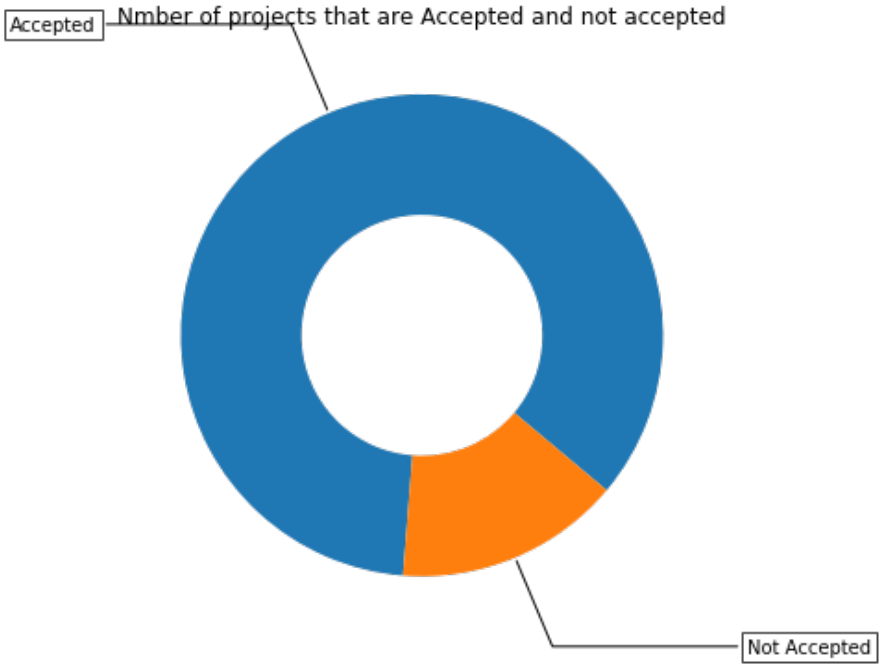
	id	description	quantity	price
0	p233245	LC652 - Lakeshore Double-Space Mobile Drying Rack	1	149.00
1	p069063	Bouncy Bands for Desks (Blue support pipes)	3	14.95

1.2 Data Analysis

In [0]:

```
# PROVIDE CITATIONS TO YOUR CODE IF YOU TAKE IT FROM ANOTHER WEBSITE.
# https://matplotlib.org/gallery/pie_and_polar_charts/pie_and_donut_labels.html#sphx-glr-gallery-pie-and-polar-charts-pie-and-donut-labels-py


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

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

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

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

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

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

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

plt.show()
```

```
Number of projects thar are approved for funding  92706 , ( 84.85830404217927 %)
Number of projects thar are not approved for funding  16542 , ( 15.141695957820739 %)
```



1.2.1 Univariate Analysis: School State

In [0]:

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

In [0]:

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

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

In [0]:

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

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

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

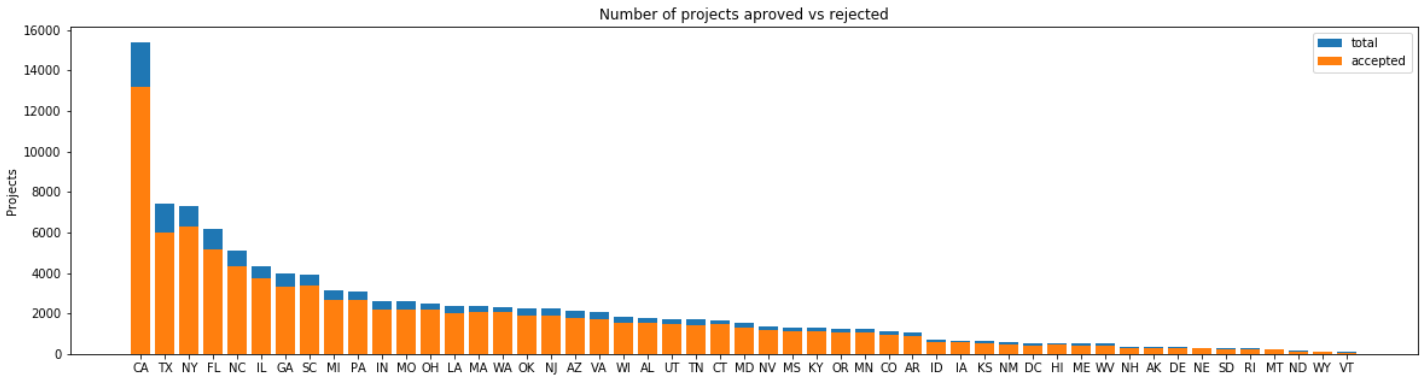
In [0]:

```
def univariate_barplots(data, col1, col2='project_is_approved', top=False):
    # Count number of zeros in dataframe python: https://stackoverflow.com/a/51540521/408
4039
    temp = pd.DataFrame(project_data.groupby(col1)[col2].agg(lambda x: x.eq(1).sum())).r
eset_index()

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

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

    if top:
        temp = temp[0:top]

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

```
    print(temp.tail(5))
```

In [0]:

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



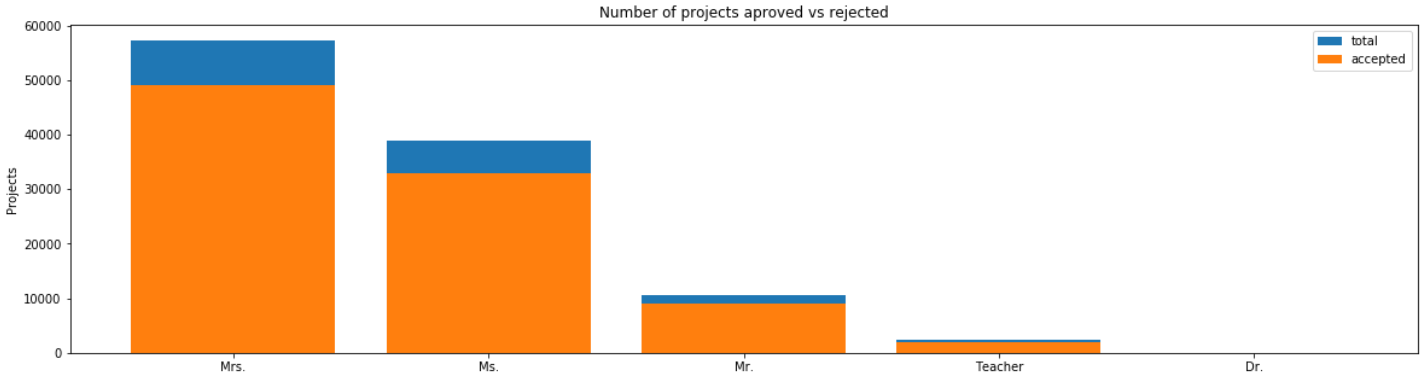

```
   school_state  project_is_approved  total       Avg
4            CA                13205  15388  0.858136
43           TX                 6014   7396  0.813142
34           NY                 6291   7318  0.859661
9            FL                 5144   6185  0.831690
27           NC           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

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

1.2.2 Univariate Analysis: teacher_prefix

In [0]:

```
univariate_barplots(project_data, 'teacher_prefix', 'project_is_approved' , top=False)
```



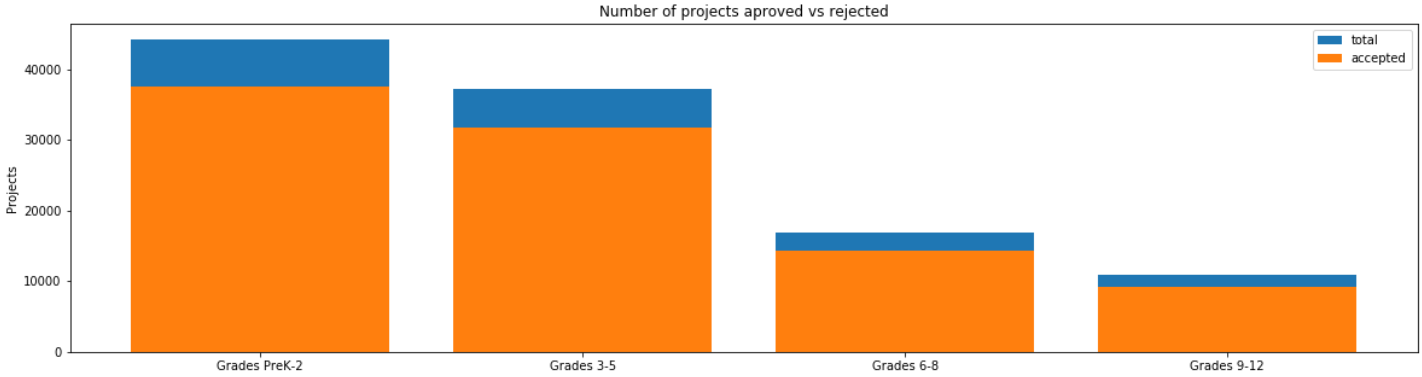

```
  teacher_prefix  project_is_approved  total       Avg
2           Mrs.                48997  57269  0.855559
3            Ms.                32860  38955  0.843537
1            Mr.                 8960  10648  0.841473
4        Teacher                 1877   2360  0.795339
0            Dr.     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

1.2.3 Univariate Analysis: project_grade_category

1.2.3 Univariate Analysis: project_grade_category

In [0]:

```
univariate_barplots(project_data, 'project_grade_category', 'project_is_approved', top=Fa
lse)
```




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

1.2.4 Univariate Analysis: project_subject_categories

In [0]:

```
catogories = list(project_data['project_subject_categories'].values)
# remove special characters from list of strings python: https://stackoverflow.com/a/4730
1924/4084039

# https://www.geeksforgeeks.org/removing-stop-words-nltk-python/
# https://stackoverflow.com/questions/23669024/how-to-strip-a-specific-word-from-a-string
# https://stackoverflow.com/questions/8270092/remove-all-whitespace-in-a-string-in-python
cat_list = []
for i in catogories:
    temp = ""
    # consider we have text like this "Math & Science, Warmth, Care & Hunger"
    for j in i.split(','): # it will split it in three parts ["Math & Science", "Warmth"
, "Care & Hunger"]
        if 'The' in j.split(): # this will split each of the catogory based on space "Ma
th & Science"=> "Math","&", "Science"
            j=j.replace('The','') # if we have the words "The" we are going to replace i
t with ''(i.e removing 'The')
        j = j.replace(' ','') # we are placeing all the ' '(space) with ''(empty) ex:"Ma
th & Science"=>"Math&Science"
        temp+=j.strip()+" " #" abc ".strip() will return "abc", remove the trailing spac
es
        temp = temp.replace('&','_') # we are replacing the & value into
    cat_list.append(temp.strip())
```

In [0]:

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

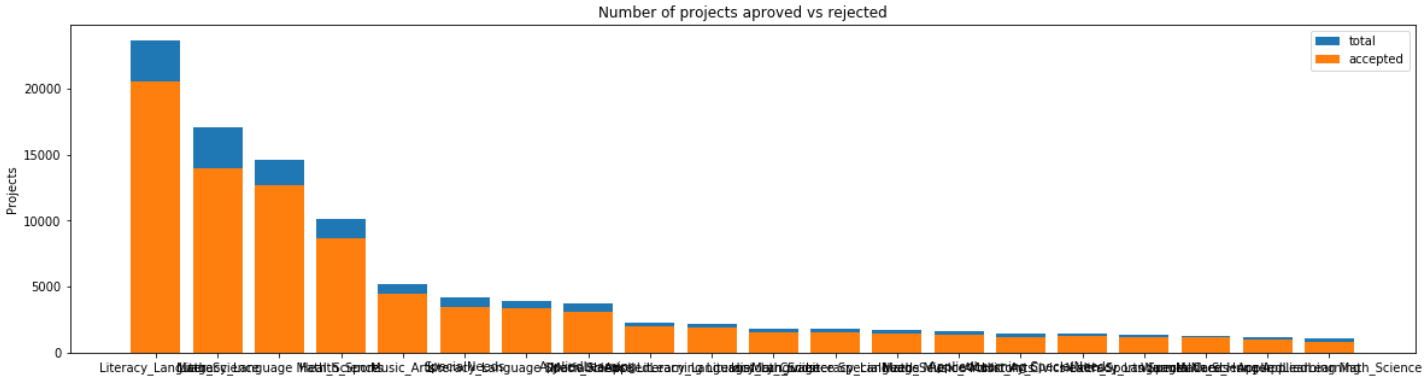
Out[0]:

Unnamed: 0	id	teacher_id	teacher_prefix	school_state	project_submitted_datetime	project

	Unnamed: 0	id	teacher_id	teacher_prefix	school_state	project_submitted_datetime	project_
0	160221	p253737	c90749f5d961ff158d4b4d1e7dc665fc	Mrs.	IN	2016-12-05 13:43:57	
1	140945	p258326	897464ce9ddc600bced1151f324dd63a	Mr.	FL	2016-10-25 09:22:10	

In [0]:

```
univariate_barplots(project_data, 'clean_categories', 'project_is_approved', top=20)
```



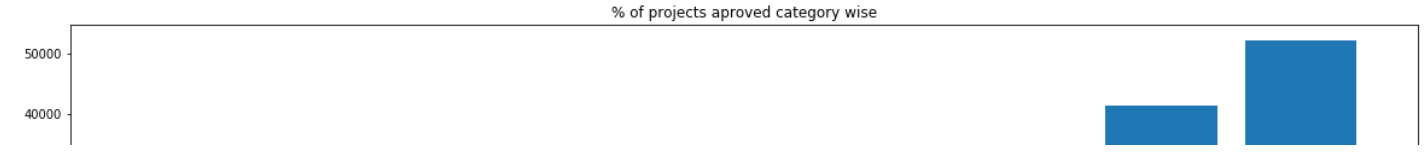

```
                  clean_categories  project_is_approved  total       Avg
24                Literacy_Language                20520  23655  0.867470
32                     Math_Science                13991  17072  0.819529
28  Literacy_Language Math_Science                12725  14636  0.869432
8 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

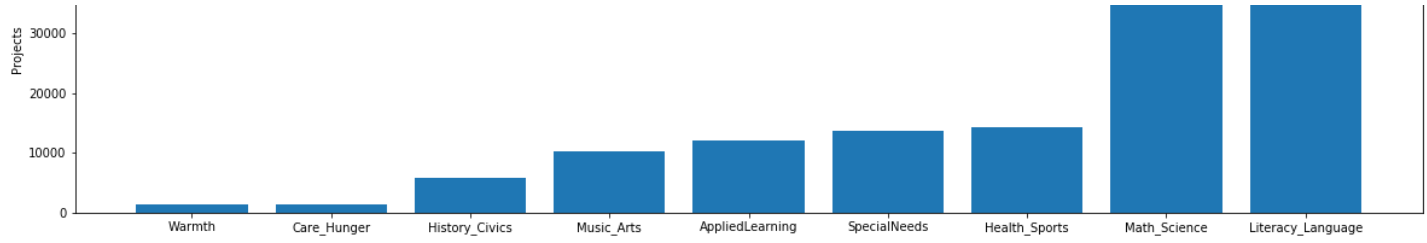
In [0]:

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

In [0]:

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


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

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





In [0]:

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

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

1.2.5 Univariate Analysis: project_subject_subcategories

In [0]:

```
sub_catogories = list(project_data['project_subject_subcategories'].values)
# remove special characters from list of strings python: https://stackoverflow.com/a/4730
1924/4084039

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

sub_cat_list = []
for i in sub_catogories:
    temp = ""
    # consider we have text like this "Math & Science, Warmth, Care & Hunger"
    for j in i.split(','): # it will split it in three parts ["Math & Science", "Warmth"
, "Care & Hunger"]
        if 'The' in j.split(): # this will split each of the catogory based on space "Ma
th & Science"=> "Math","&", "Science"
            j=j.replace('The','') # if we have the words "The" we are going to replace i
t with ''(i.e removing 'The')
        j = j.replace(' ','') # we are placeing all the ' '(space) with ''(empty) ex:"Ma
th & Science"=>"Math&Science"
        temp +=j.strip()+" "#" abc ".strip() will return "abc", remove the trailing spac
es
        temp = temp.replace('&','_')
    sub_cat_list.append(temp.strip())
```

In [0]:

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

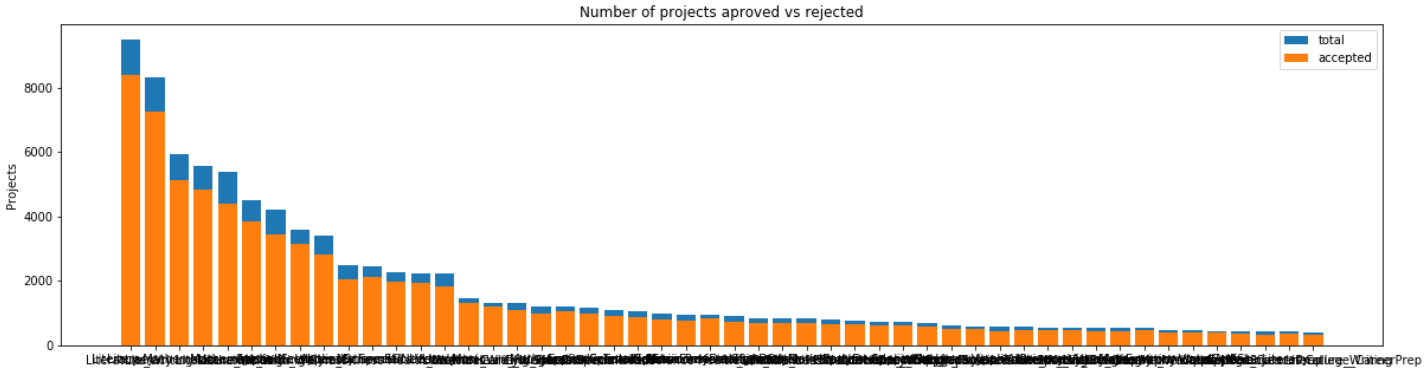
Out[0]:

Unnamed: 0	id	teacher_id	teacher_prefix	school_state	project_submitted_datetime	project
0	160221	p253737	c90749f5d961ff158d4b4d1e7dc665fc	Mrs.	IN	2016-12-05 13:43:57

	Unnamed: 0	id	teacher_id	teacher_prefix	school_state	project_submitted_datetime	project
1	140945	p258326	897464ce9ddc600bced1151f324dd63a	Mr.	FL	2016-10-25 09:22:10	

In [0]:

```
univariate_barplots(project_data, 'clean_subcategories', 'project_is_approved', top=50)
```



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

==========

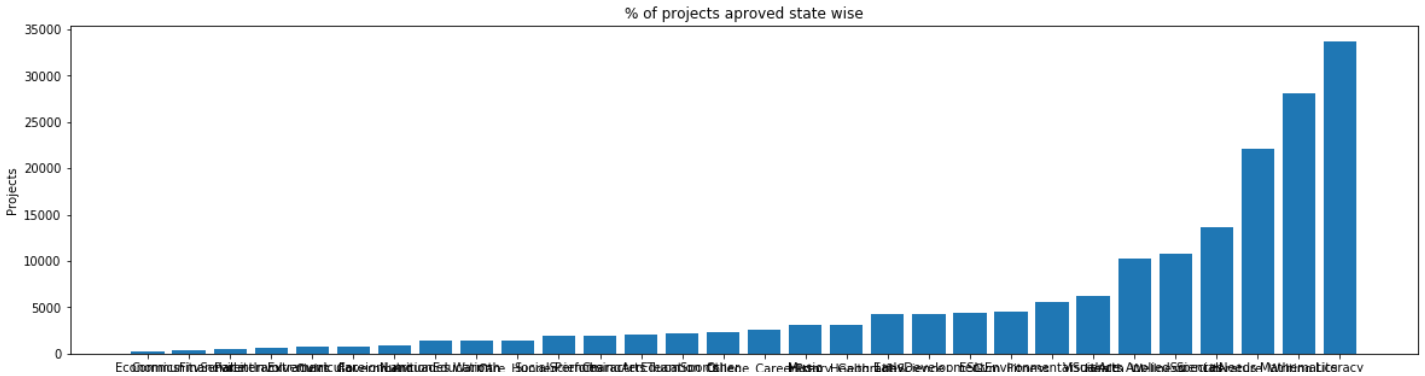
	clean_subcategories	project_is_approved	total	Avg
196	EnvironmentalScience Literacy	389	444	0.876126
127	ESL	349	421	0.828979
79	College_CareerPrep	343	421	0.814727
17	AppliedSciences Literature_Writing	361	420	0.859524
3	AppliedSciences College_CareerPrep	330	405	0.814815

In [0]:

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

In [0]:

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


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

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



In [0]:

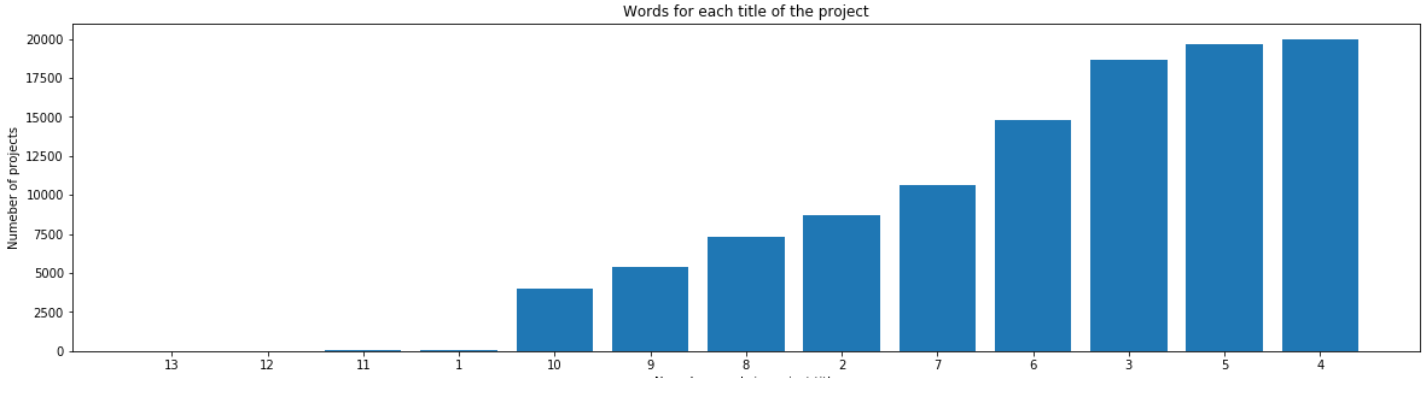
```
for i, j in sorted_sub_cat_dict.items():
    print("{:20} :{:10}".format(i,j))
```

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

1.2.6 Univariate Analysis: Text features (Title)

In [0]:

```
#How to calculate number of words in a string in DataFrame: https://stackoverflow.com/a/37483537/4084039
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()
```

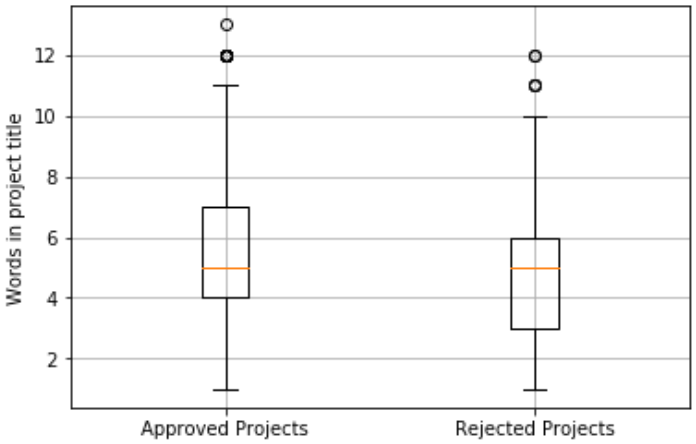


In [0]:

```
approved_title_word_count = project_data[project_data['project_is_approved']==1]['projec
t_title'].str.split().apply(len)
approved_title_word_count = approved_title_word_count.values

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

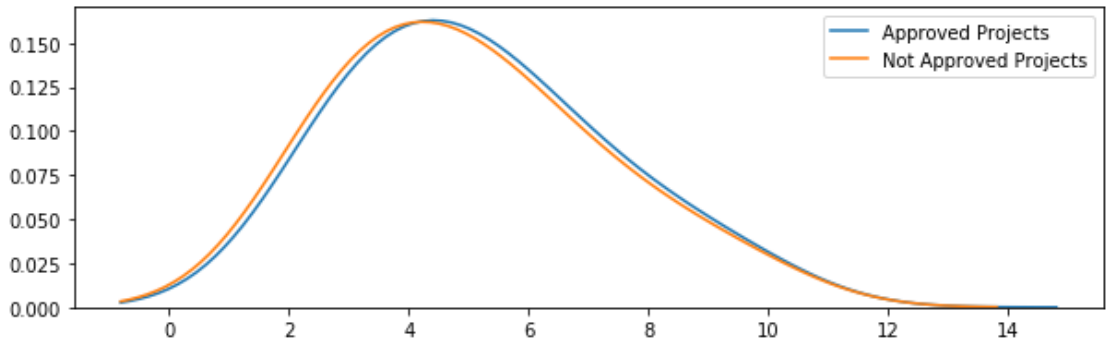
In [0]:

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



In [0]:

```
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()
```



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

In [0]:

```
# 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)
```

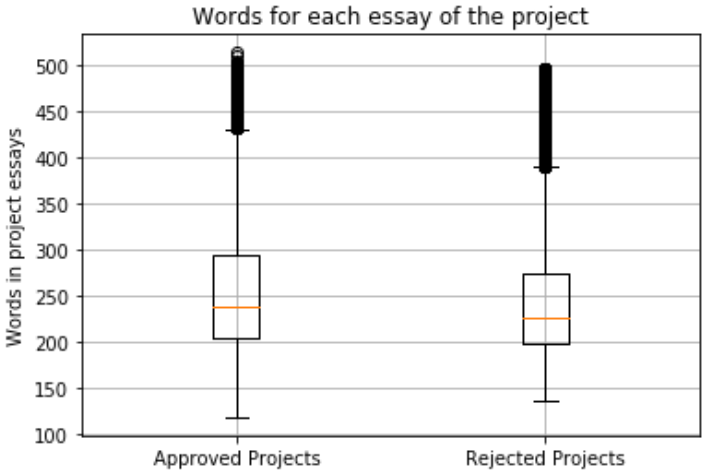
In [0]:

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

```
split().apply(len)
approved_word_count = approved_word_count.values

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

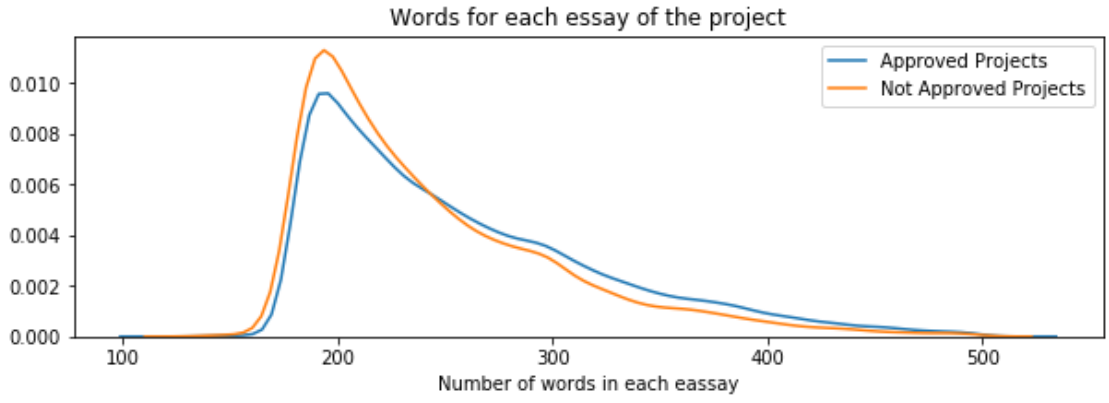
In [0]:

```
# 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 [0]:

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



1.2.8 Univariate Analysis: Cost per project

In [0]:

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

Out[0]:

	id	description	quantity	price
0	p233245	LC652 - Lakeshore Double-Space Mobile Drying Rack	1	149.00

	id	description	quantity	price
1	p069063	Bouncy Bands for Desks (Blue support pipes)	3	14.95

In [0]:

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

Out[0]:

	id	price	quantity
0	p000001	459.56	7
1	p000002	515.89	21

In [0]:

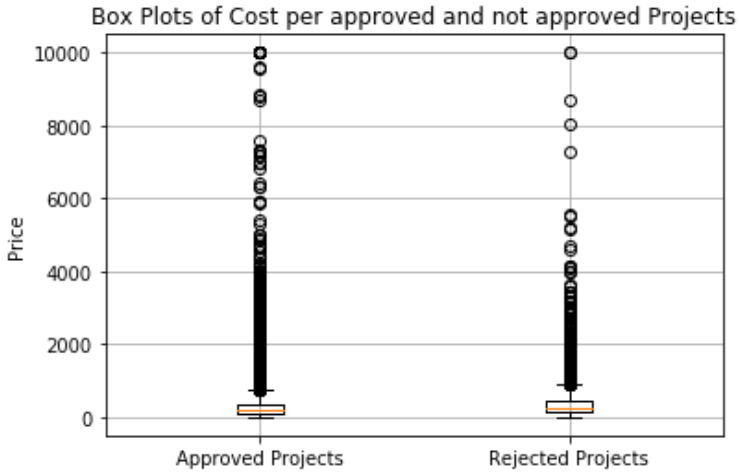
```
# join two dataframes in python:
project_data = pd.merge(project_data, price_data, on='id', how='left')
```

In [0]:

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

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

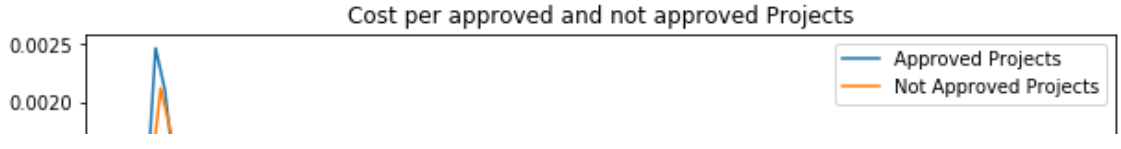
In [0]:

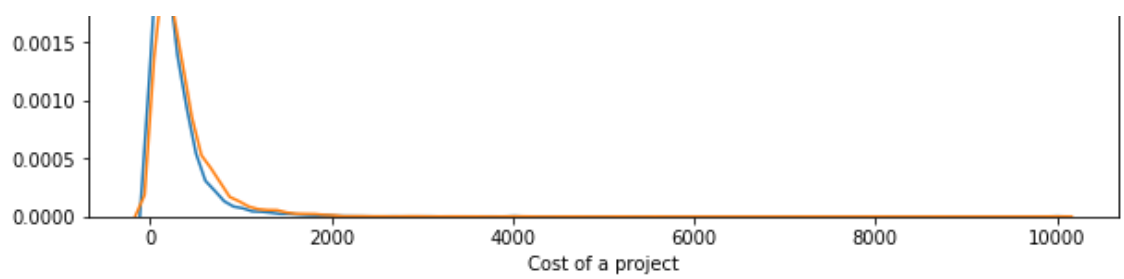
```
# https://glowingpython.blogspot.com/2012/09/boxplot-with-matplotlib.html
plt.boxplot([approved_price, rejected_price])
plt.title('Box Plots of Cost per approved and not approved Projects')
plt.xticks([1,2],('Approved Projects','Rejected Projects'))
plt.ylabel('Price')
plt.grid()
plt.show()
```



In [0]:

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





In [0]:

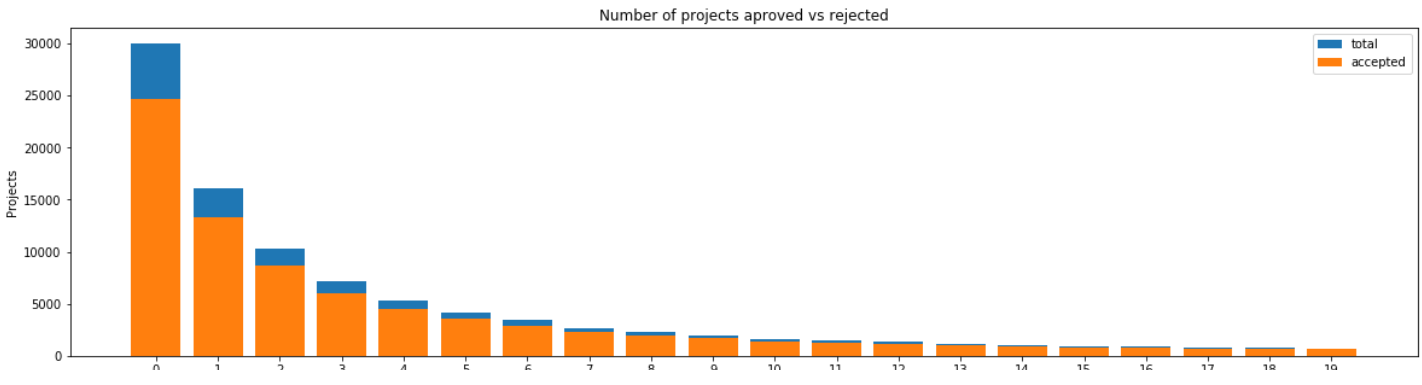
```
# http://zetcode.com/python/prettytable/
from prettytable import PrettyTable

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

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.percentile(re
jected_price,i), 3)])
print(x)
```

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

1.2.9 Univariate Analysis: teacher_number_of_previously_posted_projects

In [0]:

```
univariate_barplots(project_data, 'teacher_number_of_previously_posted_projects', 'projec
t_is_approved', top=20)
```



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

[5 rows x 4 columns]
==================================================
    teacher_number_of_previously_posted_projects  ...       Avg
15                                            15  ...  0.868365
16                                            16  ...  0.860179
17                                            17  ...  0.886675
18                                            18  ...  0.862694
19                                            19  ...  0.890141

[5 rows x 4 columns]
```

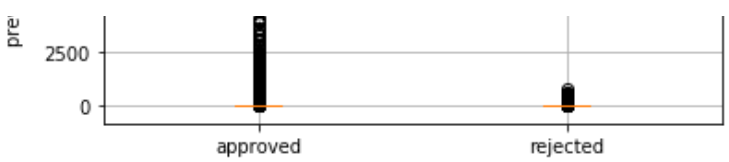
In [0]:

```
accepted_project=project_data[project_data['project_is_approved']==1][['teacher_id','tea
cher_number_of_previously_posted_projects']]

rejected_projects=project_data[project_data['project_is_approved']==0][['teacher_id','te
acher_number_of_previously_posted_projects']]
accepted_project=accepted_project.groupby(by='teacher_id').agg({'teacher_number_of_previ
ously_posted_projects':'sum'}).reset_index()
accepted_project.head()
rejected_projects=rejected_projects.groupby(by='teacher_id').agg({'teacher_number_of_pre
viously_posted_projects':'sum'}).reset_index()
```

In [0]:

```
# https://glowingpython.blogspot.com/2012/09/boxplot-with-matplotlib.html
plt.boxplot([accepted_project.teacher_number_of_previously_posted_projects,rejected_proje
cts.teacher_number_of_previously_posted_projects])
plt.xticks([1,2],['approved','rejected'])
plt.ylabel('previously posted projects')
plt.grid()

accepted_values=accepted_project['teacher_number_of_previously_posted_projects']
plt.title('No of projects teacher previously posted that are approved or rejected')
print("average no of projects teacher previously posted approved are : ",accepted_values.
sum()/len(accepted_values))
rejected_values=rejected_projects['teacher_number_of_previously_posted_projects']
print("average no of projects teacher previously posted  rejected are : ",rejected_values
.sum()/len(rejected_values))
print('maximum no of projects teacher previously posted  approved are: ',accepted_values.
max())
print('maximum no of projects teacher previously posted  rejected are: ',rejected_project
s['teacher_number_of_previously_posted_projects'].max())
print("\n")
```

```
average no of projects teacher previously posted approved are :  17.689157591287636
average no of projects teacher previously posted  rejected are :  7.43798955613577
maximum no of projects teacher previously posted  approved are:  17188
maximum no of projects teacher previously posted  rejected are:  750
```



In [0]:

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

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

x = PrettyTable()
x.field_names = ["Percentile", "Approved Projects", "Not Approved Projects"]
for i in range(0,101,10):
    x.add_row([i,np.round(np.percentile(accepted_project.teacher_number_of_previously_posted_projects,i), 3), np.round(np.percentile(rejected_projects.teacher_number_of_previously_posted_projects,i), 3)])
print(x)
```

```
+------------+--------------------+------------------------+
| Percentile | Approved Projects  | Not Approved Projects  |
+------------+--------------------+------------------------+
|     0      |        0.0         |          0.0           |
|     10     |        0.0         |          0.0           |
|     20     |        0.0         |          0.0           |
|     30     |        0.0         |          0.0           |
|     40     |        1.0         |          1.0           |
|     50     |        1.0         |          1.0           |
|     60     |        2.0         |          2.0           |
|     70     |        4.0         |          4.0           |
|     80     |        7.0         |          7.0           |
|     90     |        19.0        |          16.0          |
|    100     |      17188.0       |         750.0          |
+------------+--------------------+------------------------+
```

In [0]:

```
y = PrettyTable()
y.field_names = ["Percentile", "Approved Projects", "Not Approved Projects"]
for i in range(95,101,1):
    y.add_row([i,np.round(np.percentile(accepted_project.teacher_number_of_previously_posted_projects,i), 3), np.round(np.percentile(rejected_projects.teacher_number_of_previously_posted_projects,i), 3)])
print("\n",y)
```

```
 +------------+--------------------+------------------------+
| Percentile | Approved Projects  | Not Approved Projects  |
+------------+--------------------+------------------------+
|     95     |        49.0        |          31.0          |
|     96     |        65.0        |          37.0          |
|     97     |        92.0        |          49.0          |
|     98     |       148.22       |          67.0          |
|     99     |       310.61       |         106.0          |
|    100     |      17188.0       |         750.0          |
+------------+--------------------+------------------------+
```

Summary

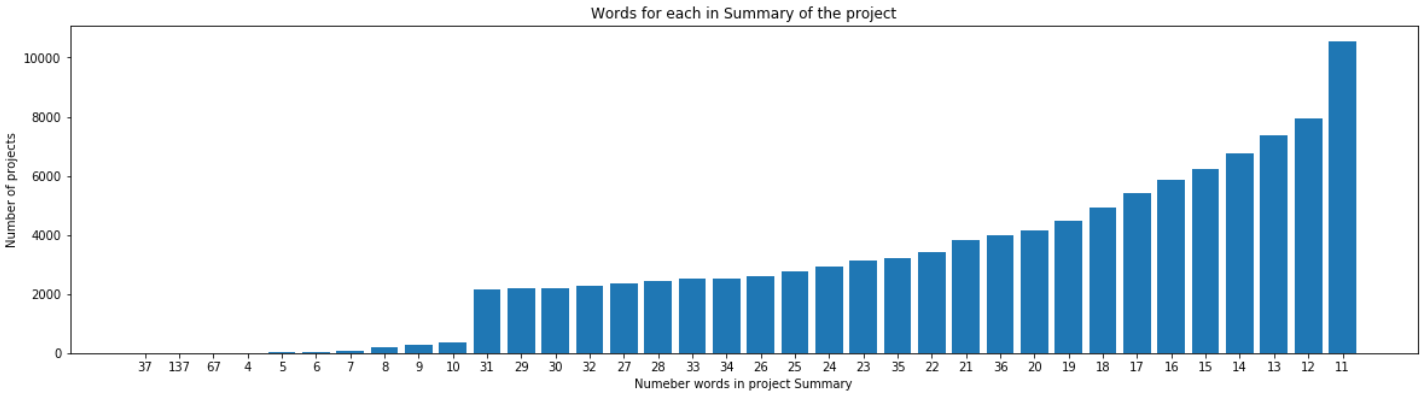
1.Average no of projects teacher previously posted approved are : 17.68
2. Average no of projects teacher previously posted rejected are : 7.43
3.Most of approved and not approved projects are in 90 to 100 percentile
4.Huge variation is from 99 to 100th percentile in no of projects teacher previously posted that are Approved

1.2.10 Univariate Analysis: project_resource_summary:

1.2.10 Univariate Analysis: project_resource_summary

In [0]:

```
#How to calculate number of words in a string in DataFrame: https://stackoverflow.com/a/37483537/4084039
word_count = project_data['project_resource_summary'].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('Number of projects')
plt.xlabel('Numeber words in project Summary')
plt.title('Words for each in Summary of the project')
plt.xticks(ind, list(word_dict.keys()))
plt.show()
```



In [0]:

```
accepted_resource_summary=project_data[project_data['project_is_approved']==1]['project_resource_summary'].reset_index()
rejected_resource_summary=project_data[project_data['project_is_approved']==0]['project_resource_summary'].reset_index()
print(accepted_resource_summary.head())
print('='*50)
print(rejected_resource_summary.head())
```

```
   index                            project_resource_summary
0      1  My students need a projector to help with view...
1      3  My students need to engage in Reading and Math...
2      4  My students need hands on practice in mathemat...
3      5  My students need movement to be successful. Be...
4      6  My students need some dependable laptops for d...
==================================================
   index                            project_resource_summary
0      0  My students need opportunities to practice beg...
1      2  My students need shine guards, athletic socks,...
2     12  My students need 3D and 4D life science activi...
3     14  My students need 5 tablets for our classroom t...
4     22  My students need books so that they can become...
```

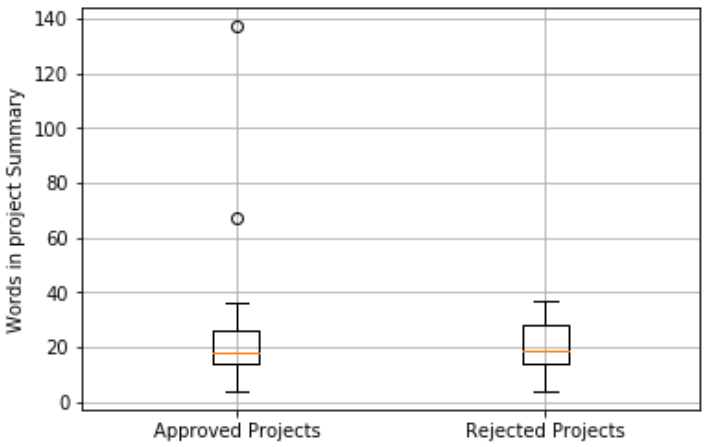
In [0]:

```
approved_summary_word_count = project_data[project_data['project_is_approved']==1]['project_resource_summary'].str.split().apply(len)

rejected_summary_word_count = project_data[project_data['project_is_approved']==0]['project_resource_summary'].str.split().apply(len)
```

In [0]:

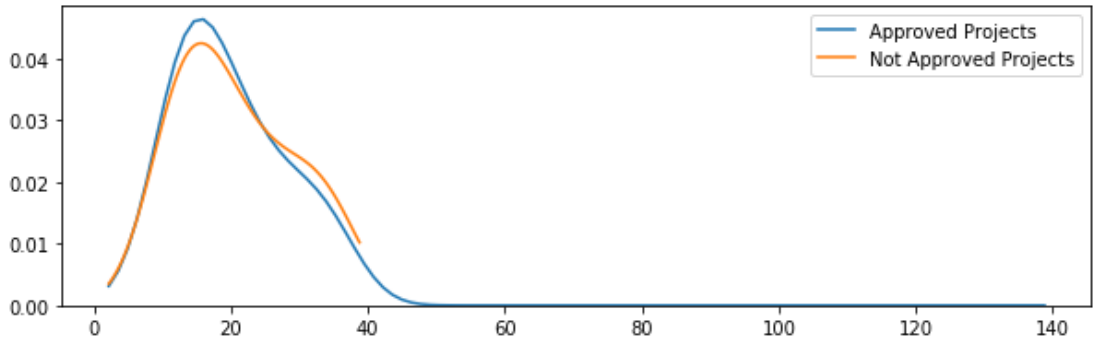
```
plt.boxplot([approved_summary_word_count, rejected_summary_word_count])
plt.xticks([1,2],('Approved Projects','Rejected Projects'))
```

```
plt.ylabel('Words in project Summary')
plt.grid()
plt.show()
```



In [0]:

```
plt.figure(figsize=(10,3))
sns.kdeplot(approved_summary_word_count,label="Approved Projects", bw=0.6)
sns.kdeplot(rejected_summary_word_count,label="Not Approved Projects", bw=0.6)
plt.legend()
plt.show()
```



In [0]:

```
from prettytable import PrettyTable

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

z = PrettyTable()
z.field_names = ["Percentile", "Approved Projects", "Not Approved Projects"]
for i in range(0,101,25):
    z.add_row([i,np.round(np.percentile(approved_summary_word_count,i), 3), np.round(np.
percentile(rejected_summary_word_count,i), 3)])
print(z)
```

```
+------------+------------------+------------------------+
| Percentile | Approved Projects | Not Approved Projects |
+------------+------------------+------------------------+
|     0      |        4.0       |          4.0           |
|     25     |        14.0      |          14.0          |
|     50     |        18.0      |          19.0          |
|     75     |        26.0      |          28.0          |
|    100     |       137.0      |          37.0          |
+------------+------------------+------------------------+
```

In [0]:

```
from prettytable import PrettyTable

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

```
a = PrettyTable()
a.field_names = ["Percentile", "Approved Projects", "Not Approved Projects"]
for i in range(95,101,1):
    a.add_row([i,np.round(np.percentile(approved_summary_word_count,i), 3), np.round(np.
percentile(rejected_summary_word_count,i), 3)])
print(a)
```

```
+------------+-------------------+-----------------------+
| Percentile | Approved Projects | Not Approved Projects |
+------------+-------------------+-----------------------+
|     95     |        35.0       |          35.0         |
|     96     |        35.0       |          36.0         |
|     97     |        36.0       |          36.0         |
|     98     |        36.0       |          36.0         |
|     99     |        36.0       |          36.0         |
|    100     |       137.0       |          37.0         |
+------------+-------------------+-----------------------+
```

SUMMARY :

1. **The number of words in the Project summary of Approved Projects are slightly more than the number of words in the Project summary of the Rejected Projects. This can be noticed by looking at the Blue Line (PDF Curve of Approved Projects) which is denser for words.**
2. **Maximum projects approved if the word count in Project summary is 11 3.The most count of words are fairly in range of 4 to 37 in project summry 3.Huge variation of percentile in approved projects from 99 to 100**

1.3 Text preprocessing

1.3.1 Essay Text

In [0]:

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

Out[0]:

	Unnamed: 0	id	teacher_id	teacher_prefix	school_state	project_submitted_datetime	project
0	160221	p253737	c90749f5d961ff158d4b4d1e7dc665fc	Mrs.	IN	2016-12-05 13:43:57	
1	140945	p258326	897464ce9ddc600bced1151f324dd63a	Mr.	FL	2016-10-25 09:22:10	

In [0]:

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

```
print("="*50)
print(project_data['essay'].values[99999])
print("="*50)
```

```
My students are English learners that are working on English as their second or third lan
guages. We are a melting pot of refugees, immigrants, and native-born Americans bringing
the gift of language to our school. \r\n\r\n We have over 24 languages represented in our
English Learner program with students at every level of mastery.  We also have over 40 co
untries represented with the families within our school.  Each student brings a wealth of
knowledge and experiences to us that open our eyes to new cultures, beliefs, and respect.
\"The limits of your language are the limits of your world.\"-Ludwig Wittgenstein  Our En
glish learner's have a strong support system at home that begs for more resources.  Many
times our parents are learning to read and speak English along side of their children.  S
ometimes this creates barriers for parents to be able to help their child learn phonetics
, letter recognition, and other reading skills.\r\n\r\nBy providing these dvd's and playe
rs, students are able to continue their mastery of the English language even if no one at
home is able to assist.  All families with students within the Level 1 proficiency status
, will be a offered to be a part of this program.  These educational videos will be speci
ally chosen by the English Learner Teacher and will be sent home regularly to watch.  The
videos are to help the child develop early reading skills.\r\n\r\nParents that do not hav
e access to a dvd player will have the opportunity to check out a dvd player to use for t
he year.  The plan is to use these videos and educational dvd's for the years to come for
other EL students.\r\nnannan
==================================================
The 51 fifth grade students that will cycle through my classroom this year all love learn
ing, at least most of the time. At our school, 97.3% of the students receive free or redu
ced price lunch. Of the 560 students, 97.3% are minority students. \r\nThe school has a v
ibrant community that loves to get together and celebrate. Around Halloween there is a wh
ole school parade to show off the beautiful costumes that students wear. On Cinco de Mayo
we put on a big festival with crafts made by the students, dances, and games. At the end
of the year the school hosts a carnival to celebrate the hard work put in during the scho
ol year, with a dunk tank being the most popular activity.My students will use these five
brightly colored Hokki stools in place of regular, stationary, 4-legged chairs. As I will
only have a total of ten in the classroom and not enough for each student to have an indi
vidual one, they will be used in a variety of ways. During independent reading time they
will be used as special chairs students will each use on occasion. I will utilize them in
place of chairs at my small group tables during math and reading times. The rest of the d
ay they will be used by the students who need the highest amount of movement in their lif
e in order to stay focused on school.\r\n\r\nWhenever asked what the classroom is missing
, my students always say more Hokki Stools. They can't get their fill of the 5 stools we
already have. When the students are sitting in group with me on the Hokki Stools, they ar
e always moving, but at the same time doing their work. Anytime the students get to pick
where they can sit, the Hokki Stools are the first to be taken. There are always students
who head over to the kidney table to get one of the stools who are disappointed as there
are not enough of them. \r\n\r\nWe ask a lot of students to sit for 7 hours a day. The Ho
kki stools will be a compromise that allow my students to do desk work and move at the sa
me time. These stools will help students to meet their 60 minutes a day of movement by al
lowing them to activate their core muscles for balance while they sit. For many of my stu
dents, these chairs will take away the barrier that exists in schools for a child who can
't sit still.nannan
==================================================
How do you remember your days of school? Was it in a sterile environment with plain walls
, rows of desks, and a teacher in front of the room? A typical day in our room is nothing
like that. I work hard to create a warm inviting themed room for my students look forward
to coming to each day.\r\n\r\nMy class is made up of 28 wonderfully unique boys and girls
of mixed races in Arkansas.\r\nThey attend a Title I school, which means there is a high
enough percentage of free and reduced-price lunch to qualify. Our school is an \"open cla
ssroom\" concept, which is very unique as there are no walls separating the classrooms. T
hese 9 and 10 year-old students are very eager learners; they are like sponges, absorbing
all the information and experiences and keep on wanting more.With these resources such as
the comfy red throw pillows and the whimsical nautical hanging decor and the blue fish ne
ts, I will be able to help create the mood in our classroom setting to be one of a themed
nautical environment. Creating a classroom environment is very important in the success i
n each and every child's education. The nautical photo props will be used with each child
as they step foot into our classroom for the first time on Meet the Teacher evening. I'll
take pictures of each child with them, have them developed, and then hung in our classroo
m ready for their first day of 4th grade.  This kind gesture will set the tone before eve
n the first day of school! The nautical thank you cards will be used throughout the year
by the students as they create thank you cards to their team groups.\r\n\r\nYour generous
donations will help me to help make our classroom a fun, inviting, learning environment f
rom day one.\r\n\r\nIt costs lost of money out of my own pocket on resources to get our c
lassroom ready. Please consider helping with this project to make our new school year a v
```

```
ery successful one. Thank you!nannan
==================================================
My kindergarten students have varied disabilities ranging from speech and language delays
, cognitive delays, gross/fine motor delays, to autism. They are eager beavers and always
strive to work their hardest working past their limitations. \r\n\r\nThe materials we hav
e are the ones I seek out for my students. I teach in a Title I school where most of the
students receive free or reduced price lunch.  Despite their disabilities and limitations
, my students love coming to school and come eager to learn and explore.Have you ever fel
t like you had ants in your pants and you needed to groove and move as you were in a meet
ing? This is how my kids feel all the time. The want to be able to move as they learn or
so they say.Wobble chairs are the answer and I love then because they develop their core,
which enhances gross motor and in Turn fine motor skills. \r\nThey also want to learn thr
ough games, my kids don't want to sit and do worksheets. They want to learn to count by j
umping and playing. Physical engagement is the key to our success. The number toss and co
lor and shape mats can make that happen. My students will forget they are doing work and
just have the fun a 6 year old deserves.nannan
==================================================
The mediocre teacher tells. The good teacher explains. The superior teacher demonstrates.
The great teacher inspires. -William A. Ward\r\n\r\nMy school has 803 students which is m
akeup is 97.6% African-American, making up the largest segment of the student body. A typ
ical school in Dallas is made up of 23.2% African-American students. Most of the students
are on free or reduced lunch. We aren't receiving doctors, lawyers, or engineers children
from rich backgrounds or neighborhoods. As an educator I am inspiring minds of young chil
dren and we focus not only on academics but one smart, effective, efficient, and discipli
ned students with good character.In our classroom we can utilize the Bluetooth for swift
transitions during class. I use a speaker which doesn't amplify the sound enough to recei
ve the message. Due to the volume of my speaker my students can't hear videos or books cl
early and it isn't making the lessons as meaningful. But with the bluetooth speaker my st
udents will be able to hear and I can stop, pause and replay it at any time.\r\nThe cart
will allow me to have more room for storage of things that are needed for the day and has
an extra part to it I can use.  The table top chart has all of the letter, words and pict
ures for students to learn about different letters and it is more accessible.nannan
==================================================
```

In [0]:

```
# 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 [0]:

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

```
My kindergarten students have varied disabilities ranging from speech and language delays
, cognitive delays, gross/fine motor delays, to autism. They are eager beavers and always
strive to work their hardest working past their limitations. \r\n\r\nThe materials we hav
e are the ones I seek out for my students. I teach in a Title I school where most of the
students receive free or reduced price lunch.  Despite their disabilities and limitations
, my students love coming to school and come eager to learn and explore.Have you ever fel
t like you had ants in your pants and you needed to groove and move as you were in a meet
ing? This is how my kids feel all the time. The want to be able to move as they learn or
so they say.Wobble chairs are the answer and I love then because they develop their core,
which enhances gross motor and in Turn fine motor skills. \r\nThey also want to learn thr
ough games, my kids do not want to sit and do worksheets. They want to learn to count by
```

jumping and playing. Physical engagement is the key to our success. The number toss and c
olor and shape mats can make that happen. My students will forget they are doing work and
just have the fun a 6 year old deserves.nannan
==========

In [0]:

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

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

In [0]:

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

My kindergarten students have varied disabilities ranging from speech and language delays
cognitive delays gross fine motor delays to autism They are eager beavers and always stri
ve to work their hardest working past their limitations The materials we have are the one
s I seek out for my students I teach in a Title I school where most of the students recei
ve free or reduced price lunch Despite their disabilities and limitations my students lov
e coming to school and come eager to learn and explore Have you ever felt like you had an
ts in your pants and you needed to groove and move as you were in a meeting This is how m
y kids feel all the time The want to be able to move as they learn or so they say Wobble
chairs are the answer and I love then because they develop their core which enhances gros
s motor and in Turn fine motor skills They also want to learn through games my kids do no
t want to sit and do worksheets They want to learn to count by jumping and playing Physic
al engagement is the key to our success The number toss and color and shape mats can make
that happen My students will forget they are doing work and just have the fun a 6 year ol
d deserves nannan

In [0]:

```
# 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're"
, "you've",\
            "you'll", "you'd", 'your', 'yours', 'yourself', 'yourselves', 'he', 'him', '
his', 'himself', \
            'she', "she's", 'her', 'hers', 'herself', 'it', "it's", 'its', 'itself', 'th
ey', 'them', 'their',\
            'theirs', 'themselves', 'what', 'which', 'who', 'whom', 'this', 'that', "tha
t'll", 'these', 'those', \
            'am', 'is', 'are', 'was', 'were', 'be', 'been', 'being', 'have', 'has', 'had
', 'having', 'do', 'does', \
            'did', 'doing', 'a', 'an', 'the', 'and', 'but', 'if', 'or', 'because', 'as',
'until', 'while', 'of', \
            'at', 'by', 'for', 'with', 'about', 'against', 'between', 'into', 'through',
'during', 'before', 'after',\
            'above', 'below', 'to', 'from', 'up', 'down', 'in', 'out', 'on', 'off', 'ove
r', 'under', 'again', 'further',\
            'then', 'once', 'here', 'there', 'when', 'where', 'why', 'how', 'all', 'any'
```

```
, 'both', 'each', 'few', 'more',\
            'most', 'other', 'some', 'such', 'only', 'own', 'same', 'so', 'than', 'too',
'very', \
            's', 't', 'can', 'will', 'just', 'don', "don't", 'should', "should've", 'now
', 'd', 'll', 'm', 'o', 're', \
            've', 'y', 'ain', 'aren', "aren't", 'couldn', "couldn't", 'didn', "didn't",
'doesn', "doesn't", 'hadn',\
            "hadn't", 'hasn', "hasn't", 'haven', "haven't", 'isn', "isn't", 'ma', 'might
n', "mightn't", 'mustn',\
            "mustn't", 'needn', "needn't", 'shan', "shan't", 'shouldn', "shouldn't", 'wa
sn', "wasn't", 'weren', "weren't", \
            'won', "won't", 'wouldn', "wouldn't"]
```

In [0]:

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

```
100%|██████████| 109248/109248 [00:56<00:00, 1936.39it/s]
```

In [0]:

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

Out[0]:

```
'my kindergarten students varied disabilities ranging speech language delays cognitive de
lays gross fine motor delays autism they eager beavers always strive work hardest working
past limitations the materials ones i seek students i teach title i school students recei
ve free reduced price lunch despite disabilities limitations students love coming school
come eager learn explore have ever felt like ants pants needed groove move meeting this k
ids feel time the want able move learn say wobble chairs answer i love develop core enhan
ces gross motor turn fine motor skills they also want learn games kids not want sit works
heets they want learn count jumping playing physical engagement key success the number to
ss color shape mats make happen my students forget work fun 6 year old deserves nannan'
```

1.3.2 Project title Text

In [0]:

```
# similarly you can preprocess the titles also
```

In [0]:

```
# printing some random Project titles.
print(project_data['project_title'].values[0])
print("="*50)
print(project_data['project_title'].values[2000])
print("="*50)
print(project_data['project_title'].values[10000])
```

```
Educational Support for English Learners at Home
==================================================
Steady Stools for Active Learning
==================================================
Family Book Clubs
```

In [0]:

```
project_data["project_title"]
```

Out[0]:

```
0          Educational Support for English Learners at Home
1                      Wanted: Projector for Hungry Learners
2         Soccer Equipment for AWESOME Middle School Stu...
3                                     Techie Kindergarteners
4                                     Interactive Math Tools
5         Flexible Seating for Mrs. Jarvis' Terrific Thi...
6         Chromebooks for Special Education Reading Program
7                                     It's the 21st Century
8                             Targeting More Success in Class
9           Just For the Love of Reading--\r\nPure Pleasure
10                                    Reading Changes Lives
11        Elevating Academics and Parent Rapports Throug...
12                      Building Life Science Experiences
13                         Everyone deserves to be heard!
14                          TABLETS CAN SHOW US THE WORLD
15                                    Making Recess Active
16                    Making Great LEAP's With Leapfrog!
17              Technology Teaches Tomorrow's Talents Today
18                                                Test Time
19                              Wiggling Our Way to Success
20                         Magic Carpet Ride in Our Library
21                 From Sitting to Standing in the Classroom
22                           Books for Budding Intellectuals
23                    Instrumental Power: Conquering STEAM!
24        S.T.E.A.M. Challenges(Science Technology Engin...
25                                              Math Masters!
26                                             Techy Teaching
27                   4th Grade French Immersion Class Ipads
28                           Hands-On Language and Literacy
29                          Basic Classroom Supplies Needed
                                    ...
109218                  ***Multi-Sensory Classroom Wish!***
109219                      Make Learning Fun in Grade One!
109220            Hooking Young Readers with Engaging Books
109221                                  Dual language Class
109222    Replenishing Our Supplies to Extend Our Learni...
109223                          Hunger Busters for Students
109224                                   STEM for 2nd Grade
109225                                    Together We Learn
109226                                Stand Up for Learning!
109227            Grab a Stool...the Fun is About to Start!
109228            Technology For Flooded Kindergarten Class
109229        Criss Cross Applesauce, we are ready to roll!
109230    Ipad Minis for Special Needs High School Students
109231               Keeping Students Informed and Inspired
109232                   Everyone Needs to have an Opinion!
109233                            Engagement through Tablets
109234       Developing A Growth Mindset for School Success
109235                          Let's focus through movement!
109236                                   Portable Projector
109237                     Choose Kindness Book Club: Wonder
109238    We Like to Move It, Move It! Flexible Seating ...
109239                                  Integrating the Arts
109240                          Spread the Love of Literature
109241                                   Read Your Heart Out!
109242                       STEM LEARNERS NEED AN IPAD MINI
109243    Privacy Shields Help Promote Independent Thinking
109244                          Technology in Our Classroom
109245               2016/2017 Beginning of the Year Basics
109246              Flexible Seating in Inclusive Classroom
109247       Classroom Tech to Develop 21st Century Leaders
Name: project_title, Length: 109248, dtype: object
```

In [0]:

```
preprocessed_titles = []
```

```
for titles in tqdm(project_data["project_title"]):
    title = decontracted(titles)
    title = title.replace('\\r', ' ')
    title = title.replace('\\"', ' ')
    title = title.replace('\\n', ' ')
    title = re.sub('[^A-Za-z0-9]+', ' ', title)
    title = ' '.join(f for f in title.split() if f not in stopwords)
    preprocessed_titles.append(title.lower().strip())
```

```
100%|██████████| 109248/109248 [00:02<00:00, 40304.71it/s]
```

In [0]:

```
print(preprocessed_titles[0])
print("="*50)
print(preprocessed_titles[2000])
print("="*50)
print(preprocessed_titles[10000])
print("="*50)
```

```
educational support english learners home
==================================================
steady stools active learning
==================================================
family book clubs
==================================================
```

1. 4 Preparing data for models

In [0]:

```
project_data.columns
```

Out[0]:

```
Index(['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'],
      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/

In [0]:

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


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

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

In [0]:

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


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

```
['Economics', 'CommunityService', 'FinancialLiteracy', 'ParentInvolvement', 'Extracurricu
lar', 'Civics_Government', 'ForeignLanguages', 'NutritionEducation', 'Warmth', 'Care_Hung
er', 'SocialSciences', 'PerformingArts', 'CharacterEducation', 'TeamSports', 'Other', 'Co
llege_CareerPrep', 'Music', 'History_Geography', 'Health_LifeScience', 'EarlyDevelopment'
, 'ESL', 'Gym_Fitness', 'EnvironmentalScience', 'VisualArts', 'Health_Wellness', 'Applied
Sciences', 'SpecialNeeds', 'Literature_Writing', 'Mathematics', 'Literacy']
Shape of matrix after one hot encodig  (109248, 30)
```

In [0]:

```
# Please do the similar feature encoding with state, teacher_prefix and project_grade_cat
egory also

#feature encoding on state code
vectorizer=CountVectorizer(vocabulary=list(project_data['school_state'].unique()),lowerca
se=False,binary=True)
state_one_hot=vectorizer.fit_transform(project_data['school_state'].values)
print('shape of matrix after one  hot encoding of state column ')
print(state_one_hot.shape)
```

```
shape of matrix after one  hot encoding of state column
(109248, 51)
```

In [0]:

```
#feature encoding on teacher prefix
project_data["teacher_prefix"].fillna(" ",inplace =True)
vectorizer=CountVectorizer(vocabulary=list(project_data['teacher_prefix'].unique()),lower
case=False,binary=True)
teacher_prefix_one_hot=vectorizer.fit_transform(project_data.teacher_prefix.values)
print('\nshape of matrix after one  hot encoding:\n',teacher_prefix_one_hot.shape)
```

```
shape of matrix after one  hot encoding:
 (109248, 6)
```

In [0]:

```
#feature encoding on project_grade_category
project_grade=project_data['project_grade_category']
project_grade.fillna(" ",inplace=True)
vectorizer=CountVectorizer(vocabulary=list(project_grade.unique()),binary=True,lowercase=
```

```
False)
project_grade_one_hot=vectorizer.fit_transform(project_grade.values)
print('shape of matrix after one  hot encoding::\n',project_grade_one_hot.shape)
```

```
shape of matrix after one  hot encoding::
 (109248, 4)
```

1.4.2 Vectorizing Text data

1.4.2.1 Bag of words

In [0]:

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

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

1.4.2.2 Bag of Words on `project_title`

In [0]:

```
# Similarly you can vectorize for title also
```

In [0]:

```
# We are considering only the words which appeared in at least 10 documents(rows or proje
cts).
vectorizer = CountVectorizer(min_df=10)
title_bow = vectorizer.fit_transform(preprocessed_titles)
print("Shape of matrix after one hot encoding ",title_bow.shape)
```

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

In [0]:

```
print ("There are {} unique words among the {} number of project titles, considering atle
ast 10 different projects have the same words".format(title_bow.shape[1], title_bow.shape
[0]))
```

```
There are 3329 unique words among the 109248 number of project titles, considering atleas
t 10 different projects have the same words
```

1.4.2.3 TFIDF vectorizer

In [0]:

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

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

1.4.2.4 TFIDF Vectorizer on `project_title`

In [0]:

```
# Similarly you can vectorize for title also
```

In [0]:

```
# We are considering only the words which appeared in at least 10 documents(rows or proje
```

```
cts).
vectorizer = TfidfVectorizer(min_df=10)
title_tfidf = vectorizer.fit_transform(preprocessed_titles)
print("Shape of matrix after one hot encoding ",title_tfidf.shape)
```

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

In [0]:

```
print ("There are {} unique words among the {} number of project titles, considering atle
ast 10 different projects have the same words".format(title_tfidf.shape[1], title_tfidf.s
hape[0]))
```

```
There are 3329 unique words among the 109248 number of project titles, considering atleas
t 10 different projects have the same words
```

1.4.2.5 Using Pretrained Models: Avg W2V

In [0]:

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

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

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

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

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

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

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

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


# stronging variables into pickle files python: http://www.jessicayung.com/how-to-use-pic
kle-to-save-and-load-variables-in-python/

import pickle
```

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


'''
```

Out[0]:

```
'\n# Reading glove vectors in python: https://stackoverflow.com/a/38230349/4084039\ndef l
oadGloveModel(gloveFile):\n    print ("Loading Glove Model")\n    f = open(gloveFile,\'r\
', encoding="utf8")\n    model = {}\n    for line in tqdm(f):\n        splitLine = line.s
plit()\n        word = splitLine[0]\n        embedding = np.array([float(val) for val in
splitLine[1:]])\n        model[word] = embedding\n    print ("Done.",len(model)," words
loaded!")\n    return model\nmodel = loadGloveModel(\'glove.42B.300d.txt\')\n\n# ========
====================\nOutput:\n    \nLoading Glove Model\n1917495it [06:32, 4879.69it/s]\
nDone. 1917495  words loaded!\n\n# ============================\n\nwords = []\nfor i in p
reproced_texts:\n    words.extend(i.split(\' \'))\n\nfor i in preproced_titles:\n    word
s.extend(i.split(\' \'))\nprint("all the words in the coupus", len(words))\nwords = set(w
ords)\nprint("the unique words in the coupus", len(words))\n\ninter_words = set(model.key
s()).intersection(words)\nprint("The number of words that are present in both glove vecto
rs and our coupus",       len(inter_words),"(",np.round(len(inter_words)/len(words)*100,3
),"%)")\n\nwords_courpus = {}\nwords_glove = set(model.keys())\nfor i in words:\n    if i
in words_glove:\n        words_courpus[i] = model[i]\nprint("word 2 vec length", len(word
s_courpus))\n\n\n# stronging variables into pickle files python: http://www.jessicayung.c
om/how-to-use-pickle-to-save-and-load-variables-in-python/\n\nimport pickle\nwith open(\'
glove_vectors\', \'wb\') as f:\n    pickle.dump(words_courpus, f)\n\n\n'
```

In [0]:

```
# stronging variables into pickle files python: http://www.jessicayung.com/how-to-use-pic
kle-to-save-and-load-variables-in-python/
# make sure you have the glove_vectors file
with open('gdrive/My Drive/Machine Learning/Assignmnets/Assignment2/Assignments_DonorsCho
ose_2018/glove_vectors', 'rb') as f:
    model = pickle.load(f)
    glove_words =  set(model.keys())
```

In [0]:

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

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

```
100%|██████████| 109248/109248 [00:29<00:00, 3725.46it/s]

109248
300
```

1.4.2.6 Using Pretrained Models: AVG W2V on `project_title`

In [0]:

```
# Similarly you can vectorize for title also
avg_w2v_project_titles = []; # the avg-w2v for each sentence/review is stored in this lis
t
for sentence in tqdm(preprocessed_titles): # for each title
    vector = np.zeros(300) # as word vectors are of zero length
```

```
    cnt_words =0; # num of words with a valid vector in the 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_project_titles.append(vector)

print(len(avg_w2v_project_titles))
print(len(avg_w2v_project_titles[0]))
```

```
100%|██████████| 109248/109248 [00:01<00:00, 71009.12it/s]
```

```
109248
300
```

1.4.2.7 Using Pretrained Models: TFIDF weighted W2V

In [0]:

```
# 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 [0]:

```
# average Word2Vec
# compute average word2vec for each review.
tfidf_w2v_vectors = []; # the avg-w2v for each sentence/review is stored in this list
for sentence in tqdm(preprocessed_essays): # for each review/sentence
    vector = np.zeros(300) # as word vectors are of zero length
    tf_idf_weight =0; # num of words with a valid vector in the sentence/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 value((sente
nce.count(word)/len(sentence.split())))
            tf_idf = dictionary[word]*(sentence.count(word)/len(sentence.split())) # get
ting the tfidf value for each word
            vector += (vec * tf_idf) # calculating tfidf weighted w2v
            tf_idf_weight += tf_idf
    if tf_idf_weight != 0:
        vector /= tf_idf_weight
    tfidf_w2v_vectors.append(vector)

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

```
100%|██████████| 109248/109248 [03:00<00:00, 604.63it/s]
```

```
109248
300
```

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

In [0]:

```
tfidf_model = TfidfVectorizer()
tfidf_model.fit(preprocessed_titles)
# we are converting a dictionary with word as a key, and the idf as a value
dictionary = dict(zip(tfidf_model.get_feature_names(), list(tfidf_model.idf_)))
tfidf_words = set(tfidf_model.get_feature_names())
```

In [0]:

```
# average Word2Vec
# compute average word2vec for each Project Title
tfidf_w2v_vectors_title = []; # the avg-w2v for each sentence/review is stored in this li
st
for sentence in tqdm(preprocessed_titles): # for each review/sentence
    vector = np.zeros(300) # as word vectors are of zero length
    tf_idf_weight =0; # num of words with a valid vector in the sentence/review
    for word in sentence.split(): # for each word in a review/sentence
        if (word in glove_words) and (word in tfidf_words):
            vec = model[word] # getting the vector for each word
            # here we are multiplying idf value(dictionary[word]) and the tf value((sente
nce.count(word)/len(sentence.split())))
            tf_idf = dictionary[word]*(sentence.count(word)/len(sentence.split())) # get
ting the tfidf value for each word
            vector += (vec * tf_idf) # calculating tfidf weighted w2v
            tf_idf_weight += tf_idf
    if tf_idf_weight != 0:
        vector /= tf_idf_weight
    tfidf_w2v_vectors_title.append(vector)

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

```
100%|██████████| 109248/109248 [00:03<00:00, 28487.35it/s]
```

```
109248
300
```

1.4.3 Vectorizing Numerical features

In [0]:

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

# price_standardized = standardScalar.fit(project_data['price'].values)
# this will rise the error
# ValueError: Expected 2D array, got 1D array instead: array=[725.05 213.03 329.   ... 39
9.   287.73   5.5 ].
# Reshape your data either using array.reshape(-1, 1)

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

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

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

In [0]:

```
price_standardized
```

Out[0]:

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

In [0]:

```
prev_projects_scalar = StandardScaler()

## Finding the mean and standard deviation of this data
prev_projects_scalar.fit(project_data['teacher_number_of_previously_posted_projects'].val
ues.reshape(-1,1))

print("Mean : {}".format(prev_projects_scalar.mean_[0]))

print("Standard deviation : {}".format(np.sqrt(prev_projects_scalar.var_[0])))

# Now standardize the data with above maen and variance.
prev_projects_standardized = prev_projects_scalar.transform(project_data['teacher_number_
of_previously_posted_projects'].values.reshape(-1, 1))
```

```
Mean : 11.153165275336848
Standard deviation : 27.77702641477403
```

1.4.4 Merging all the above features

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

In [0]:

```
print(categories_one_hot.shape)
print(sub_categories_one_hot.shape)
print(text_bow.shape)
print(price_standardized.shape)
```

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

In [0]:

```
# 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 matirx :
)
X = hstack((categories_one_hot, sub_categories_one_hot, text_bow, price_standardized))
X.shape
```

Out[0]:

```
(109248, 16663)
```

In [0]:

```
type(X)
```

Out[0]:

```
scipy.sparse.coo.coo_matrix
```

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.

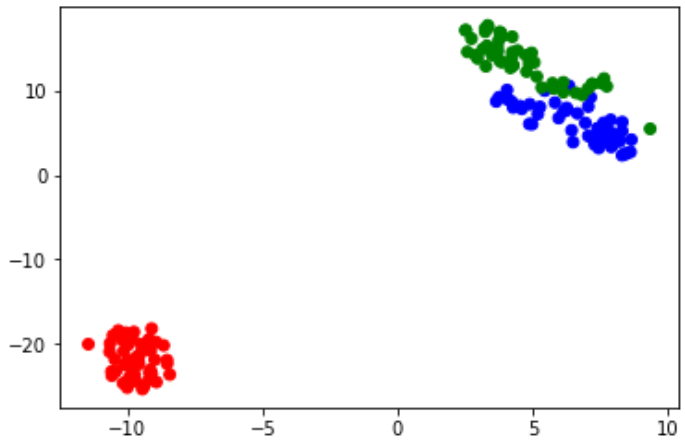
1. **In the above cells we have plotted and analyzed many features. Please observe the plots and write the observations in markdown cells below every plot.**
2. **EDA: Please complete the analysis of the feature: teacher_number_of_previously_posted_projects**
3. **Build the data matrix using these features**
   - **school_state : categorical data (one hot encoding)**
   - **clean_categories : categorical data (one hot encoding)**
   - **clean_subcategories : categorical data (one hot encoding)**

- clean_subcategories : categorical data (one hot encoding)
- teacher_prefix : categorical data (one hot encoding)
- project_grade_category : categorical data (one hot encoding)
- project_title : text data (BOW, TFIDF, AVG W2V, TFIDF W2V)
- price : numerical
- teacher_number_of_previously_posted_projects : numerical

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

In [0]:

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

iris = datasets.load_iris()
x = iris['data']
y = iris['target']

tsne = TSNE(n_components=2, perplexity=30, learning_rate=200)

X_embedding = tsne.fit_transform(x)
# if x is a sparse matrix you need to pass it as X_embedding = tsne.fit_transform(x.toarray()) , .toarray() will convert the sparse matrix into dense matrix

for_tsne = np.hstack((X_embedding, y.reshape(-1,1)))
for_tsne_df = pd.DataFrame(data=for_tsne, columns=['Dimension_x','Dimension_y','Score'])
colors = {0:'red', 1:'blue', 2:'green'}
plt.scatter(for_tsne_df['Dimension_x'], for_tsne_df['Dimension_y'], c=for_tsne_df['Score'].apply(lambda x: colors[x]))
plt.show()
```



2.1 TSNE with `BOW` encoding of `project_title` feature (3000 Data points)

In [0]:

```
X = hstack((categories_one_hot, sub_categories_one_hot, state_one_hot, project_grade_one_hot, teacher_prefix_one_hot, price_standardized, prev_projects_standardized, title_bow))
X.shape
```

Out[0]:

```
(109248, 3431)
```

In [0]:

```
type(X)
```

Out[0]:

```
scipy.sparse.coo.coo_matrix
```

In [0]:

```
from sklearn.manifold import TSNE
X = X.tocsr()
X_new = X[0:3000,:]
```

In [0]:

```
title_bow_data=X[0:3000]
title_bow_data=title_bow_data.toarray()
#print(title_bow_data)
print('shape of training data for tsne with bow',title_bow_data.shape)
```

```
shape of training data for tsne with bow (3000, 3431)
```

In [0]:

```
X_new = X_new.toarray()
model = TSNE(n_components = 2, perplexity = 100.0, random_state = 0)
tsne_data_b = model.fit_transform(X_new)
```

In [0]:

```
labels = project_data["project_is_approved"]
labels_new = labels[0: 3000]
```

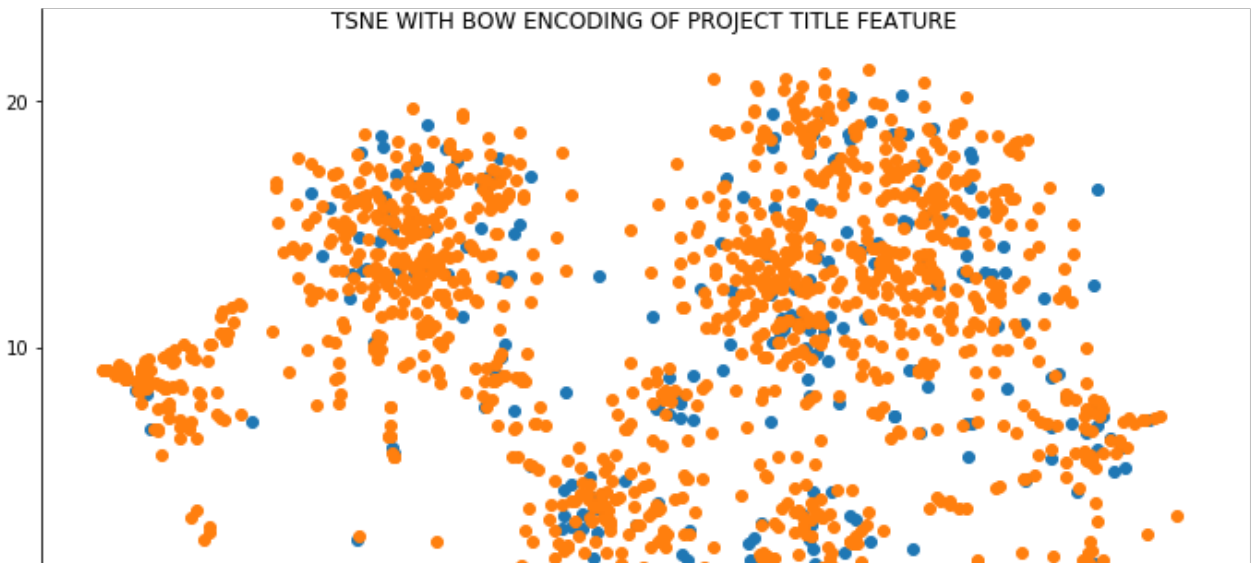
In [0]:

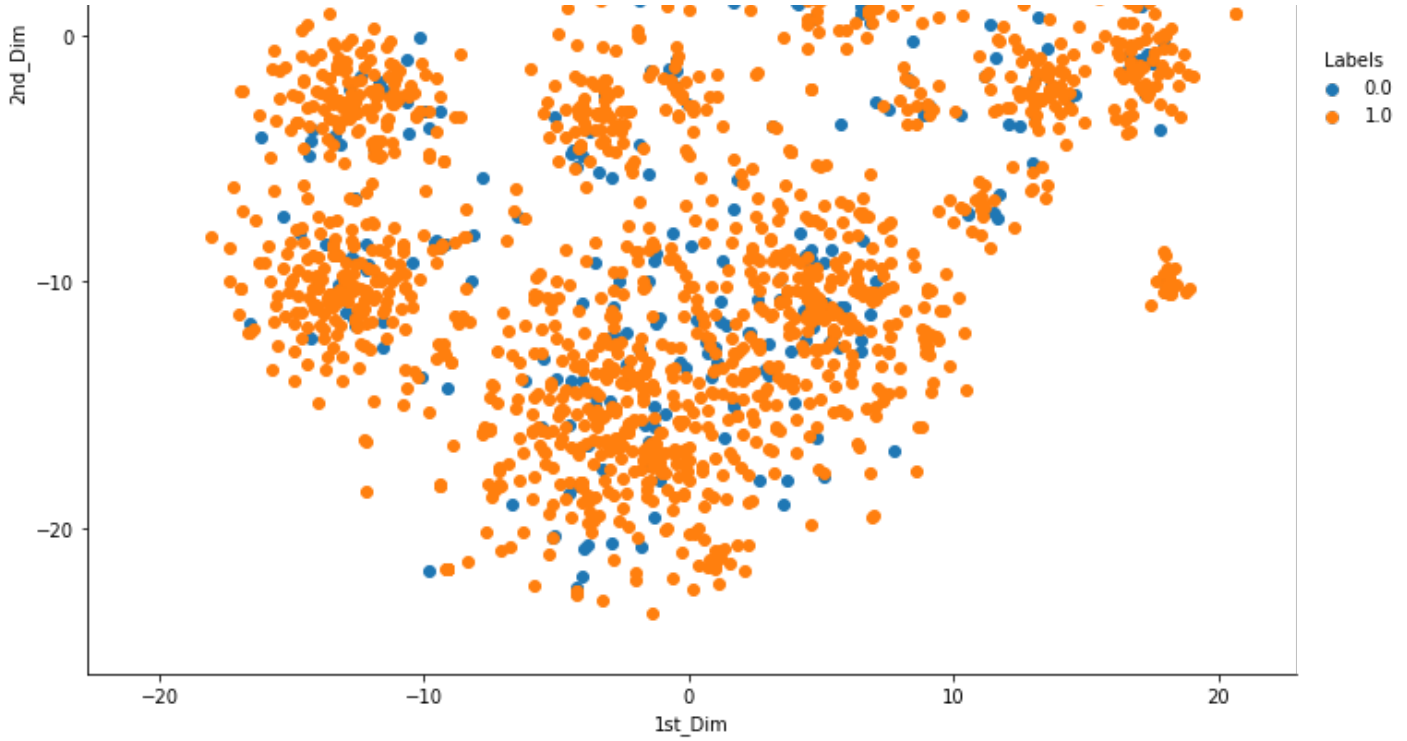
```
tsne_data_b = np.vstack((tsne_data_b.T, labels_new)).T
tsne_df_b = pd.DataFrame(tsne_data_b, columns = ("1st_Dim","2nd_Dim","Labels"))
```

In [0]:

```
sns.FacetGrid(tsne_df_b, hue = "Labels", size = 10).map(plt.scatter, "1st_Dim", "2nd_Dim"
).add_legend().fig.suptitle("TSNE WITH BOW ENCODING OF PROJECT TITLE FEATURE with  preple
xity 30 ")
plt.show()
```

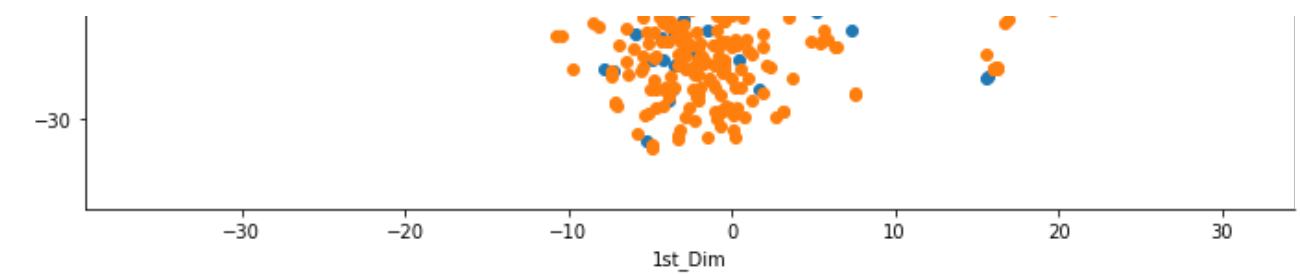
```
/usr/local/lib/python3.6/dist-packages/seaborn/axisgrid.py:230: UserWarning:

The `size` paramter has been renamed to `height`; please update your code.
```





In [0]:

```
import warnings
warnings.filterwarnings('ignore')
model = TSNE(n_components = 2, perplexity = 30, random_state = 0,n_iter=500)
tsne_data_b = model.fit_transform(X_new)
tsne_data_b = np.vstack((tsne_data_b.T, labels_new)).T
tsne_df_b = pd.DataFrame(tsne_data_b, columns = ("1st_Dim","2nd_Dim","Labels"))
sns.FacetGrid(tsne_df_b, hue = "Labels", size = 10).map(plt.scatter, "1st_Dim", "2nd_Dim"
).add_legend().fig.suptitle("TSNE WITH BOW ENCODING OF PROJECT TITLE FEATURE with  preple
xity 30 ")
plt.show()
```



In [0]:

```
import warnings
warnings.filterwarnings('ignore')
model = TSNE(n_components = 2, perplexity = 300, random_state = 0,n_iter=800)
tsne_data_b = model.fit_transform(X_new)
tsne_data_b = np.vstack((tsne_data_b.T, labels_new)).T
tsne_df_b = pd.DataFrame(tsne_data_b, columns = ("1st_Dim","2nd_Dim","Labels"))
sns.FacetGrid(tsne_df_b, hue = "Labels", size = 10).map(plt.scatter, "1st_Dim", "2nd_Dim"
).add_legend().fig.suptitle("TSNE WITH BOW ENCODING OF PROJECT TITLE FEATURE with  preple
xity 300 anf n_iter=800 ")
plt.show()
```

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

In [0]:

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

In [0]:

```
X = hstack((categories_one_hot, sub_categories_one_hot, state_one_hot, project_grade_one_
hot, teacher_prefix_one_hot, price_standardized, prev_projects_standardized, title_tfidf)
)
X.shape
```

Out[0]:

```
(109248, 3431)
```

In [0]:

```
X = X.tocsr()
X_new = X[0:3000,:]
```

In [0]:

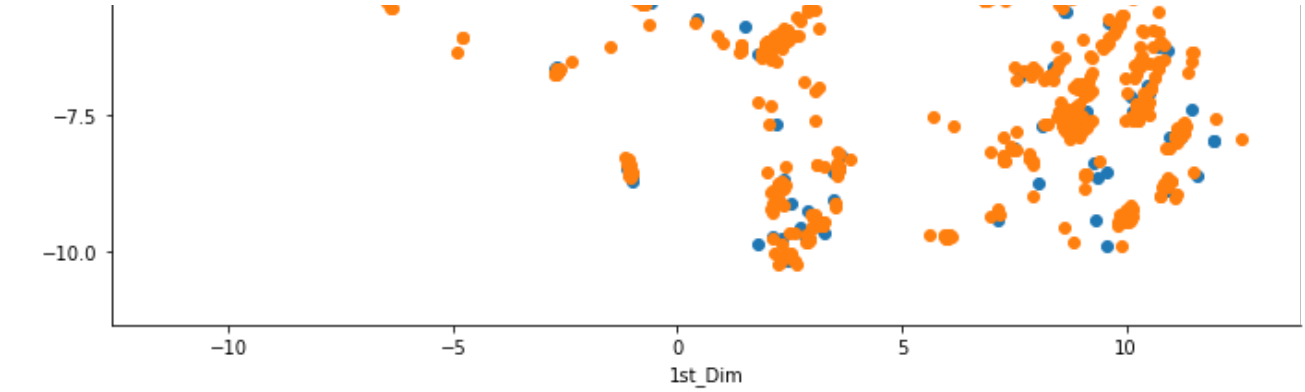
```
X_new = X_new.toarray()
model = TSNE(n_components = 2, perplexity = 150.0, random_state = 0,n_iter=300)
tsne_data_tfidf = model.fit_transform(X_new)
```

In [0]:

```
tsne_data_tfidf = np.vstack((tsne_data_tfidf.T, labels_new)).T
tsne_df_tfidf = pd.DataFrame(tsne_data_tfidf, columns = ("1st_Dim","2nd_Dim","Labels"))
```

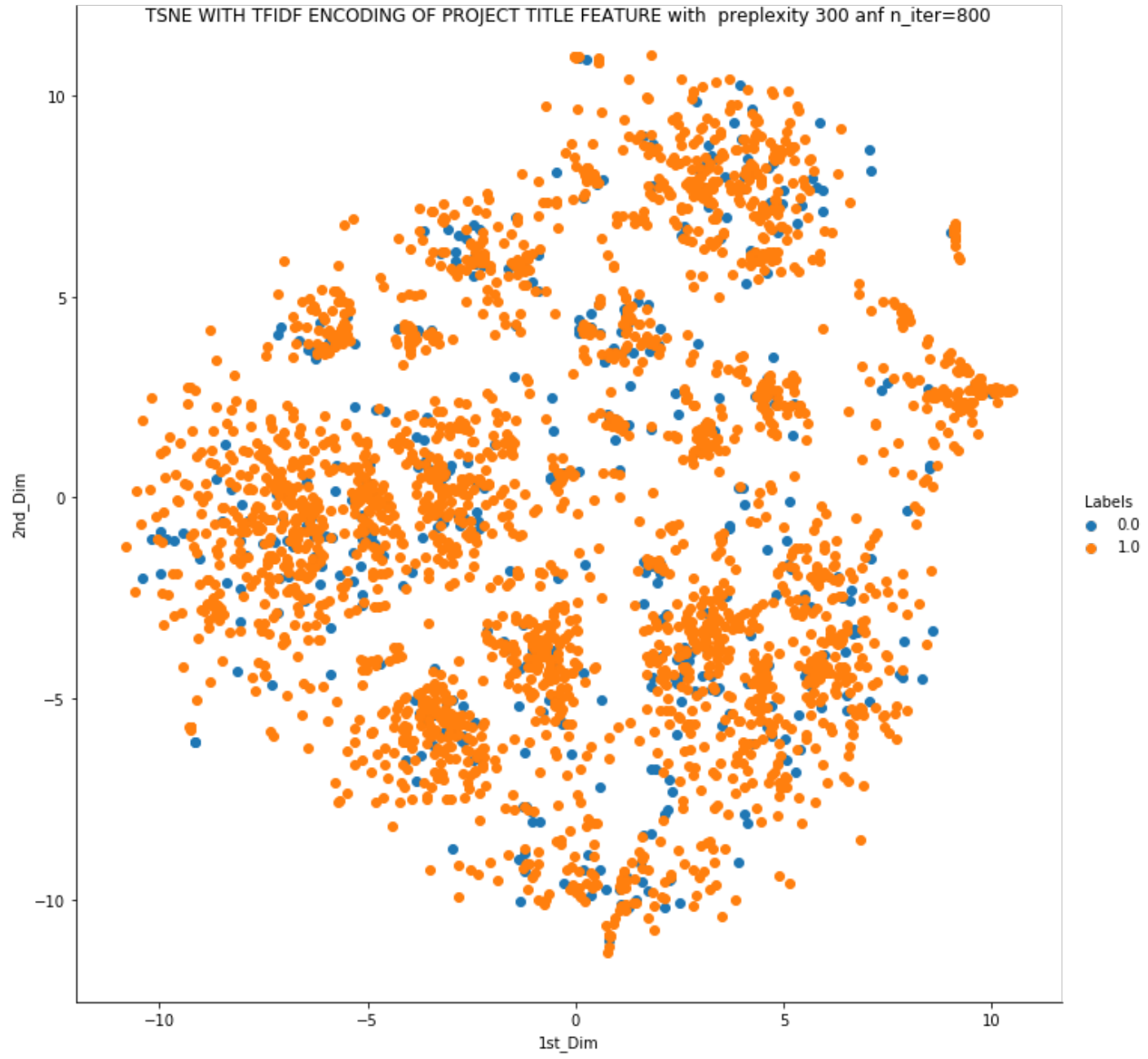
In [0]:

```
sns.FacetGrid(tsne_df_tfidf, hue = "Labels", size = 10).map(plt.scatter, "1st_Dim", "2nd_
Dim").add_legend().fig.suptitle("TSNE WITH TF-IDF ENCODING OF PROJECT TITLE FEATURE with
preplexity 150 and n_iter=300 ")
plt.show()
```



In [0]:

```
import warnings
warnings.filterwarnings('ignore')
model = TSNE(n_components = 2, perplexity = 300, random_state = 0,n_iter=800)
tsne_data_tfidf = model.fit_transform(X_new)
tsne_data_tfidf  = np.vstack((tsne_data_tfidf.T, labels_new)).T
tsne_df_tfidf = pd.DataFrame(tsne_data_tfidf, columns = ("1st_Dim","2nd_Dim","Labels"))
sns.FacetGrid(tsne_df_b, hue = "Labels", size = 10).map(plt.scatter, "1st_Dim", "2nd_Dim"
).add_legend().fig.suptitle("TSNE WITH TFIDF ENCODING OF PROJECT TITLE FEATURE with  prep
lexity 300 anf n_iter=800 ")
plt.show()
```



In [0]:

```
import warnings
warnings.filterwarnings('ignore')
model = TSNE(n_components = 2, perplexity = 30, random_state = 0,n_iter=500)
tsne_data_tfidf = model.fit_transform(X_new)
tsne_data_tfidf  = np.vstack((tsne_data_tfidf.T, labels_new)).T
tsne_df_tfidf = pd.DataFrame(tsne_data_tfidf, columns = ("1st_Dim","2nd_Dim","Labels"))
sns.FacetGrid(tsne_df_b, hue = "Labels", size = 10).map(plt.scatter, "1st_Dim", "2nd_Dim"
).add_legend().fig.suptitle("TSNE WITH TFIDF ENCODING OF PROJECT TITLE FEATURE with  prep
lexity 30 anf n_iter=500 ")
plt.show()
```

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

In [0]:

```
X = hstack((categories_one_hot, sub_categories_one_hot, state_one_hot, project_grade_one_
hot, teacher_prefix_one_hot, price_standardized, prev_projects_standardized, avg_w2v_proj
ect_titles))
```

In [0]:

```
X = X.tocsr()
X_new = X[0:3000,:]
```

In [0]:

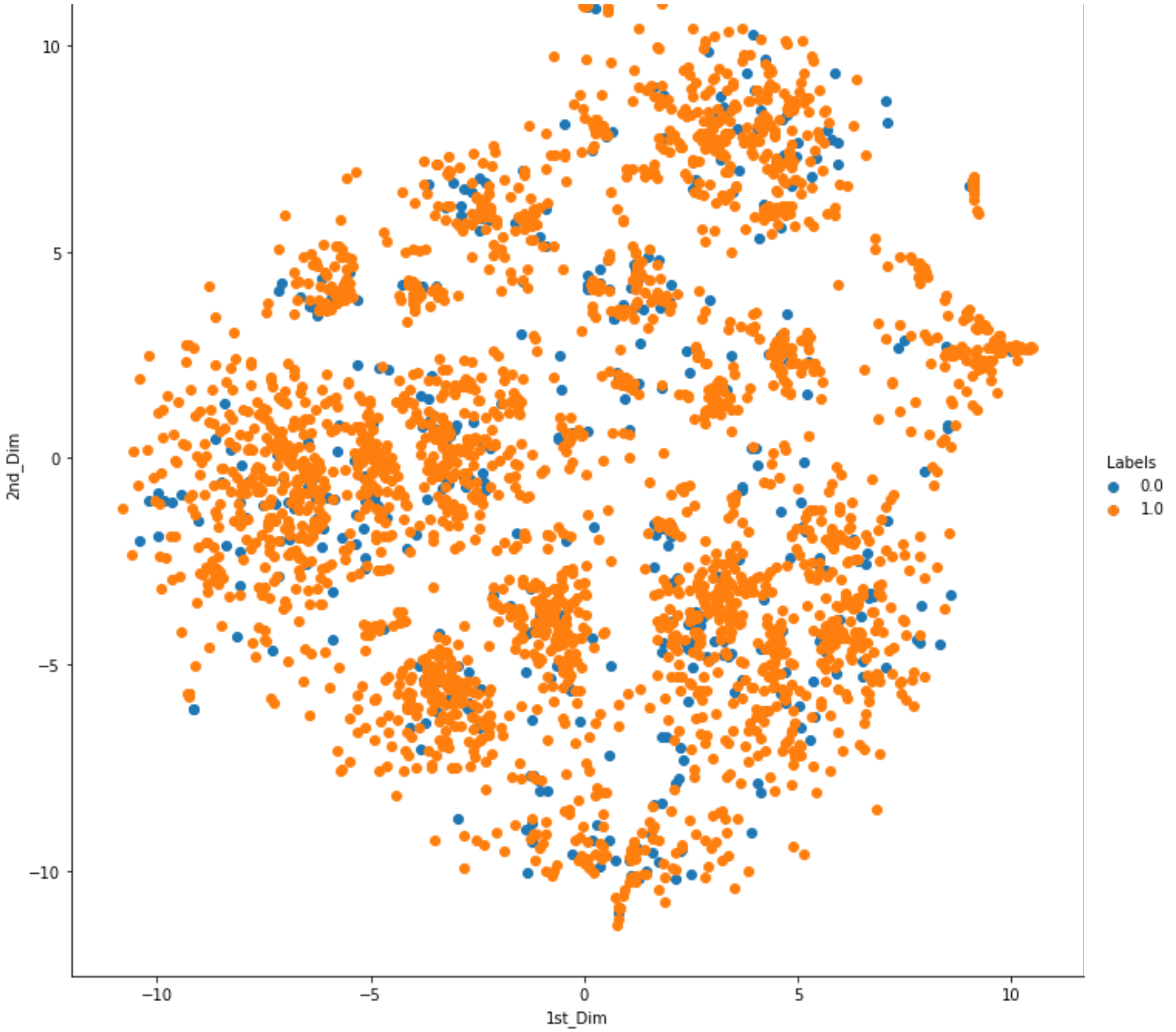
```
X_new = X_new.toarray()
model = TSNE(n_components = 2, perplexity = 300.0, random_state = 0,n_iter=800)
tsne_data_avg_w2v = model.fit_transform(X_new)
```

In [0]:

```
tsne_data_avg_w2v = np.vstack((tsne_data_avg_w2v.T, labels_new)).T
tsne_df_avg_w2v = pd.DataFrame(tsne_data_avg_w2v, columns = ("1st_Dim","2nd_Dim","Labels
"))
sns.FacetGrid(tsne_df_b, hue = "Labels", size = 10).map(plt.scatter, "1st_Dim", "2nd_Dim"
).add_legend().fig.suptitle("TSNE WITH TFIDF avg_w2v ENCODING OF PROJECT TITLE FEATURE wi
th  preplexity 300 anf n_iter=800 ")
plt.show()
```

In [0]:

```
model = TSNE(n_components = 2, perplexity = 500.0, random_state = 0,n_iter=500)
tsne_data_avg_w2v = model.fit_transform(X_new)
tsne_data_avg_w2v = np.vstack((tsne_data_avg_w2v.T, labels_new)).T
tsne_df_avg_w2v = pd.DataFrame(tsne_data_avg_w2v, columns = ("1st_Dim","2nd_Dim","Labels
"))
sns.FacetGrid(tsne_df_b, hue = "Labels", size = 10).map(plt.scatter, "1st_Dim", "2nd_Dim"
).add_legend().fig.suptitle("TSNE WITH TFIDF avg_w2v ENCODING OF PROJECT TITLE FEATURE wi
th  preplexity 500 anf n_iter=500 ")
plt.show()
```



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

In [0]:

```
X = hstack((categories_one_hot, sub_categories_one_hot, state_one_hot, project_grade_one_hot, teacher_prefix_one_hot, price_standardized, prev_projects_standardized, tfidf_w2v_vectors_title))
```

In [0]:

```
X = X.tocsr()
X_new = X[0:3000,:]
```

In [0]:

```
X_new = X_new.toarray()
```

In [0]:

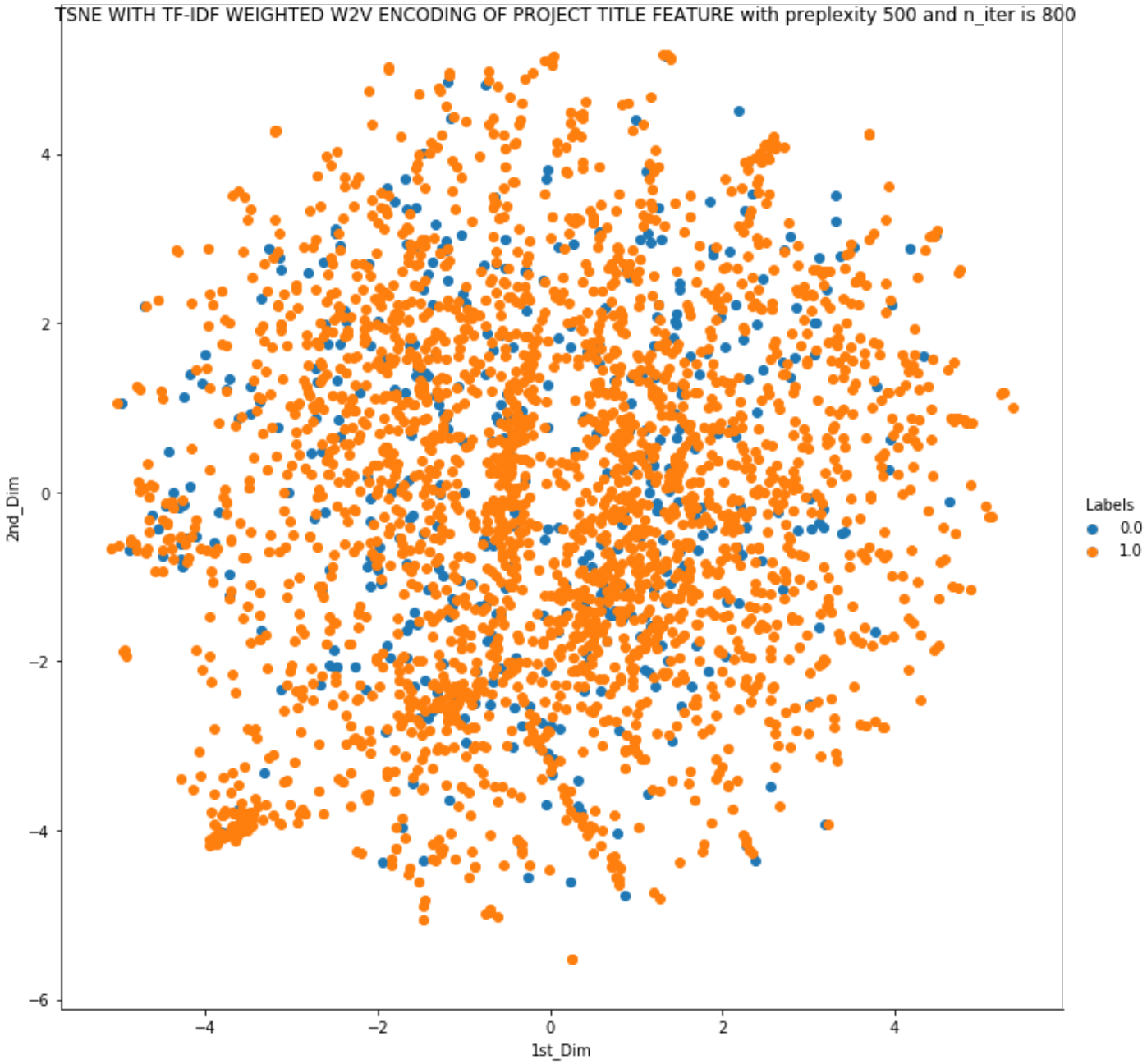
```
model = TSNE(n_components = 2, perplexity = 500.0, random_state = 0,n_iter=500)
tsne_data_tfidf_w2v = model.fit_transform(X_new)
```

In [0]:

```
tsne_data_tfidf_w2v = np.vstack((tsne_data_tfidf_w2v.T, labels_new)).T
tsne_df_tfidf_w2v = pd.DataFrame(tsne_data_tfidf_w2v, columns = ("1st_Dim","2nd_Dim","Labels"))
```

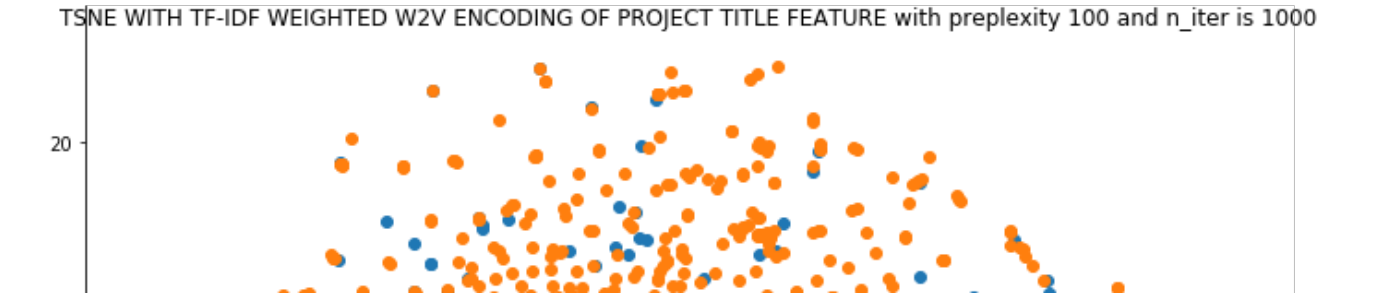
In [0]:

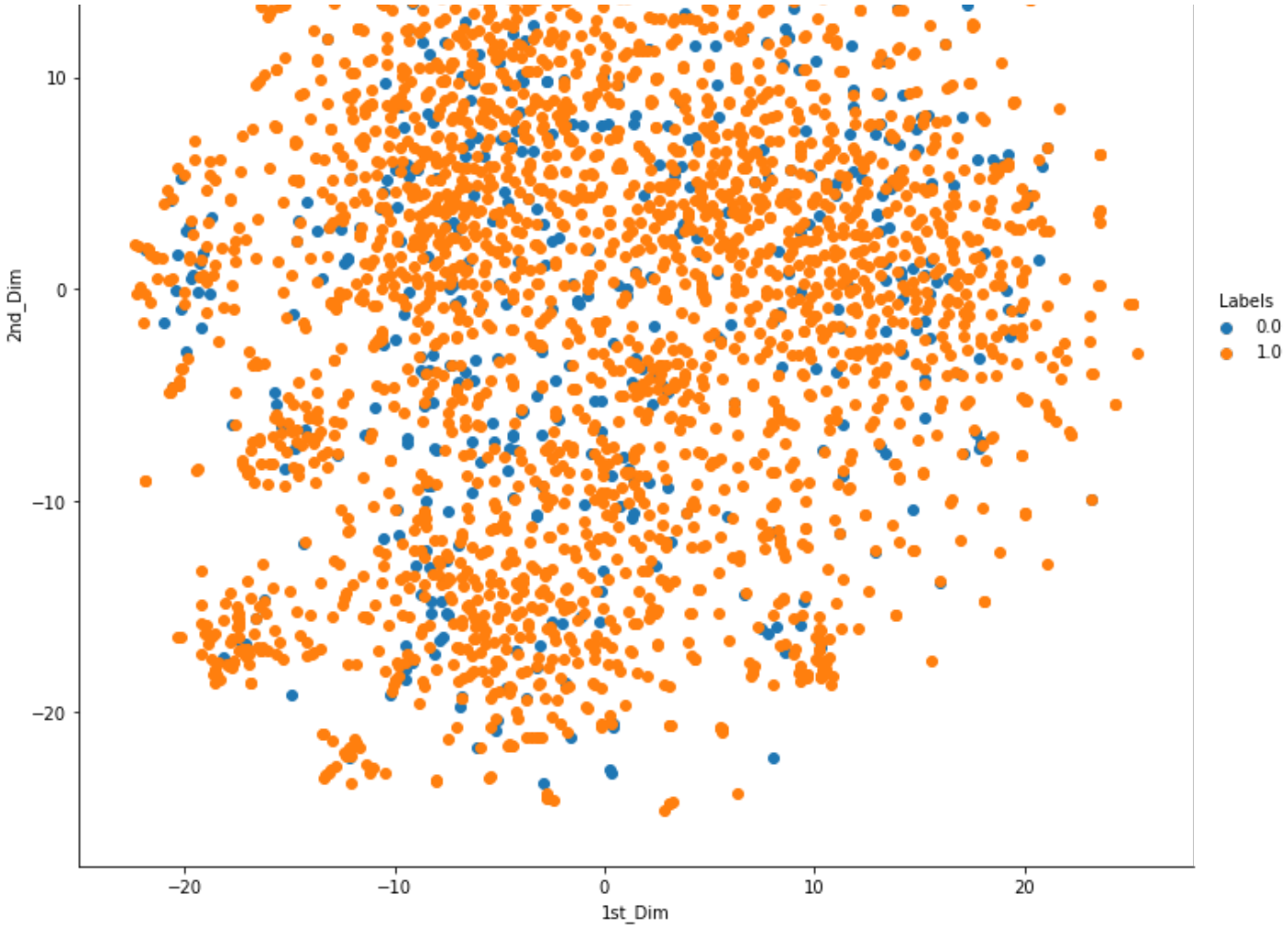
```
sns.FacetGrid(tsne_df_tfidf_w2v, hue = "Labels", size = 10).map(plt.scatter, "1st_Dim",
"2nd_Dim").add_legend().fig.suptitle("TSNE WITH TF-IDF WEIGHTED W2V ENCODING OF PROJECT T
ITLE FEATURE with preplexity 500 and n_iter is 800 ")
plt.show()
```



In [0]:

```
model = TSNE(n_components = 2, perplexity = 100.0, random_state = 0,n_iter=1000)
tsne_data_tfidf_w2v = model.fit_transform(X_new)
tsne_data_tfidf_w2v = np.vstack((tsne_data_tfidf_w2v.T, labels_new)).T
tsne_df_tfidf_w2v = pd.DataFrame(tsne_data_tfidf_w2v, columns = ("1st_Dim","2nd_Dim","La
bels"))
sns.FacetGrid(tsne_df_tfidf_w2v, hue = "Labels", size = 10).map(plt.scatter, "1st_Dim",
"2nd_Dim").add_legend().fig.suptitle("TSNE WITH TF-IDF WEIGHTED W2V ENCODING OF PROJECT T
ITLE FEATURE with preplexity 100 and n_iter is 1000 ")
plt.show()
```





2.5 Summary

1.Number of projects that had been approved 92,703(84.5%)

2.no of projects that has been rejeced 16,542(15.14 %)

3. Delaware (DE) state almost 90% acceptance rate.

4. Vermont (VT) has the lowest Approval rate with exactly 80%.

5. Acceptance of Pre Kindergarden and 2nd Grade projects are more compare to remaining grades.

6.The maximum number of accepted projects are from Literacy and Language.

7. The sub-Category Literacy has the highest number of projects approved with the accpetance rate is 88%.

8. In project title maximum number of words are 4.

9. The number of words in the Project Essays of Approved Projects are slightly more than the number of words in the Project Essays of the Rejected Projects.

10. Accepted projects are tend to have lower cost compared to rejected project cost.

11. 82% of the approved projects have been submitted by teachers with no previous submitted projects.

12. TSNE with Bag of Words :There is no linear division between approved and not approved projects.Its like overalapping and scattered one.Tried different preplexity and no of iterations but No change in shape too.

13. TSNE with TF-IDF too doesn't have linear division between approved and not approved projects.But tsne modeling is faster than BOW and the plot is like scattered one.Tried different preplexity and no of iterations but No change in shape too.

14. TSNE with Avg Word2Vec, TF-IDF Weighted Word2Vec too doesn't have linear division between approved and not approved projects.But tsne modeling is faster than BOW and the plot is like scattered one.Tried different preplexity and no of iterations but No change in shape too.

15. Imbalanced Datset.Number of Approved projects are almost 6 times more than Rejected projects