

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
project_title	Title of the project. Examples: <ul style="list-style-type: none"> • Art Will Make You Happy! • First Grade Fun
project_grade_category	Grade level of students for which the project is targeted. Enumerated values: <ul style="list-style-type: none"> • Grades PreK-2 • Grades 3-5 • Grades 6-8 • Grades 9-12
project_subject_categories	One or more (comma-separated) subject categories from the following enumerated list of values: <ul style="list-style-type: none"> • Applied Learning • Care & Hunger • Health & Sports • History & Civics • Literacy & Language • Math & Science • Music & The Arts • Special Needs • Warmth Examples: <ul style="list-style-type: none"> • Music & The Arts • Literacy & Language, Math & Science
school_state	State where school is located (<u>Two-letter U.S. postal code</u> (https://en.wikipedia.org/wiki/List_of_U.S._state_abbreviations)). Example: WY
project_subject_subcategories	One or more (comma-separated) subject subcategories. Examples: <ul style="list-style-type: none"> • Literacy • Literature & Writing, Social Sciences
project_resource_summary	An explanation of the resources needed for the project. <ul style="list-style-type: none"> • My students need hands on literacy materials to address sensory needs!

Feature	Description
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: 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: <ul style="list-style-type: none"> • nan • Dr. • Mr. • Mrs. • Ms. • Teacher.
teacher_number_of_previously_posted_projects	Number of project applications previously submitted by 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 <code>train.csv</code> file. Example: p036502
description	Description of the resource. Example: Tenor Saxophone Reeds, Box of 25
quantity	Quantity of the resource required. Example: 3
price	Price of the resource required. Example: 9.95

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

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

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



Notes on the Essay Data

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

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

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

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

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

Importing Libraries

In [1]:

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

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

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

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

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

from tqdm import tqdm
import os

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

1.1 Reading Data

In [2]:

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

In [3]:

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

Number of data points in train data (109248, 17)

The attributes of data : ['Unnamed: 0' 'id' 'teacher_id' 'teacher_prefix'
'school_state'
'project_submitted_datetime' 'project_grade_category'
'project_subject_categories' 'project_subject_subcategories'
'project_title' 'project_essay_1' 'project_essay_2' 'project_essay_3'
'project_essay_4' 'project_resource_summary'
'teacher_number_of_previously_posted_projects' 'project_is_approved']

In [4]:

```
print("Number of data points in train data", resource_data.shape)
print(resource_data.columns.values)
resource_data.head(2)
```

Number of data points in train data (1541272, 4)

['id' 'description' 'quantity' 'price']

Out[4]:

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

1.2 Data Analysis

In [5]:

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

y_value_counts = project_data['project_is_approved'].value_counts()
print("Number of projects that 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 that 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", "Rejected"]

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

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

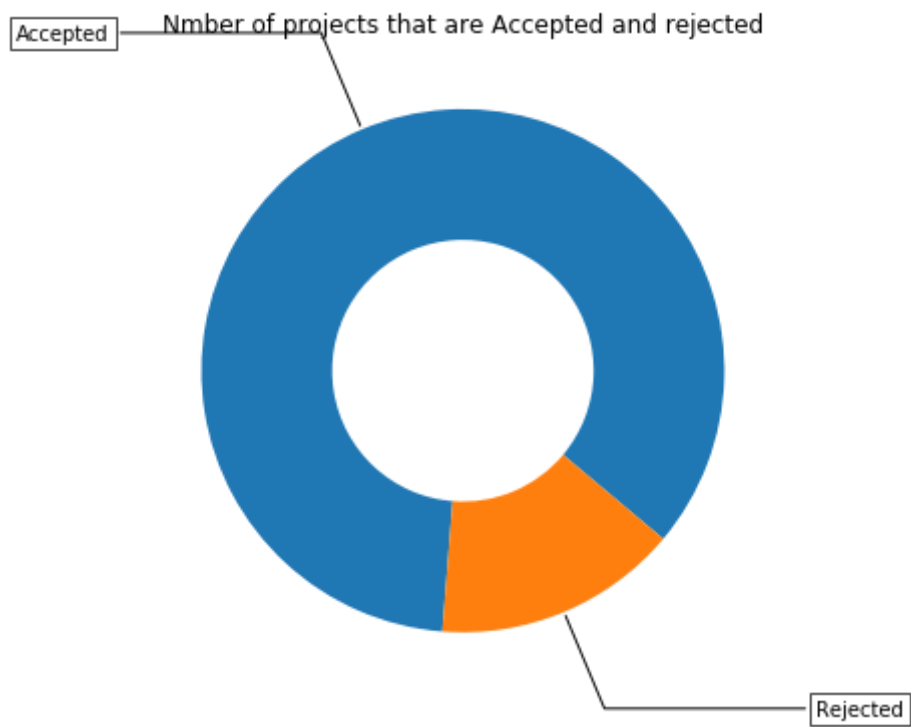
bbox_props = dict(boxstyle="square,pad=0.3", fc="w", ec="k", lw=0.72)
kw = dict(xycoords='data', textcoords='data', arrowprops=dict(arrowstyle="-"),
          bbox=bbox_props, zorder=0, va="center")

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("Number of projects that are Accepted and rejected")

plt.show()
```

Number of projects thar are approved for funding 92706 , (84.85830404217 927 %)
Number of projects thar are not approved for funding 16542 , (15.1416959 57820739 %)



In [6]:

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

temp = pd.DataFrame(project_data.groupby("school_state")["project_is_approved"].apply(n
p.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']

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

scl = [[0.0, 'rgb(242,240,247)'],[0.2, 'rgb(218,218,235)'],[0.4, 'rgb(188,189,220)'],\
      [0.6, 'rgb(158,154,200)'],[0.8, 'rgb(117,107,177)'],[1.0, 'rgb(84,39,14
3)']]

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

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

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

Out[6]:

```
'# How to plot US state heatmap: https://datascience.stackexchange.com/a/9
620\n\nscl = [[0.0, \'rgb(242,240,247)\'],[0.2, \'rgb(218,218,235)\'],[0.
4, \'rgb(188,189,220)\'],[0.6, \'rgb(158,154,200)\'],[0.8, \'r
gb(117,107,177)\'],[1.0, \'rgb(84,39,143)\']] \n\n data = [ dict(\n      t
ype=\'choropleth\',\n      colorscale = scl,\n      autocolorscale = F
alse,\n      locations = temp[\'state_code\'],\n      z = temp[\'num_p
roposals\'].astype(float),\n      locationmode = \'USA-states\',\n
text = temp[\'state_code\'],\n      marker = dict(line = dict (color =
\'rgb(255,255,255)\',width = 2)),\n      colorbar = dict(title = "% of p
ro")\n    ) ] \n\n layout = dict(\n      title = \'Project Proposals % of
Acceptance Rate by US States\',\n      geo = dict(\n          scope=
\'usa\',\n          projection=dict( type=\'albers usa\' ),\n
showlakes = True,\n          lakecolor = \'rgb(255, 255, 255)\',\n
),\n    ) \n\n fig = go.Figure(data=data, layout=layout)\noffline.iplot(fig,
filename=\'us-map-heat-map\') \n'
```

1.2.1 Univariate Analysis: School State

In [7]:

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

temp = pd.DataFrame(project_data.groupby("school_state")["project_is_approved"].apply(n
p.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']

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

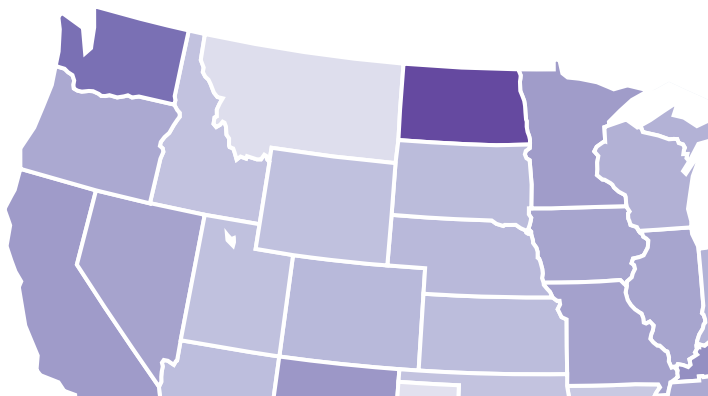
scl = [[0.0, 'rgb(242,240,247)'],[0.2, 'rgb(218,218,235)'],[0.4, 'rgb(188,189,220)'],\
       [0.6, 'rgb(158,154,200)'],[0.8, 'rgb(117,107,177)'],[1.0, 'rgb(84,39,143)']
      ]

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

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

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

Project Proposals % of Acceptance Rate by US States



In [8]:

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

States with lowest % approvals

	state_code	num_proposals
46	VT	0.800000
7	DC	0.802326
43	TX	0.813142
26	MT	0.816327
18	LA	0.831245

=====

States with highest % approvals

	state_code	num_proposals
30	NH	0.873563
35	OH	0.875152
47	WA	0.876178
28	ND	0.888112
8	DE	0.897959

In [9]:

```
#stacked bar plots matplotlib: https://matplotlib.org/gallery/lines\_bars\_and\_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 approved vs rejected')
    plt.xticks(ind, list(data[xtick].values))
    plt.legend((p1[0], p2[0]), ('total', 'accepted'))
    plt.show()
```

In [10]:

```
#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 approved vs rejected')
    plt.xticks(ind, list(data[xtick].values))
    plt.legend((p1[0], p2[0]), ('total', 'accepted'))
    plt.show()
```

In [11]:

```
def univariate_barplots(data, col1, col2='project_is_approved', top=False):
    # Count number of zeros in dataframe python: https://stackoverflow.com/a/51540521/4084039
    temp = pd.DataFrame(project_data.groupby(col1)[col2].agg(lambda x: x.eq(1).sum())).reset_index()

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

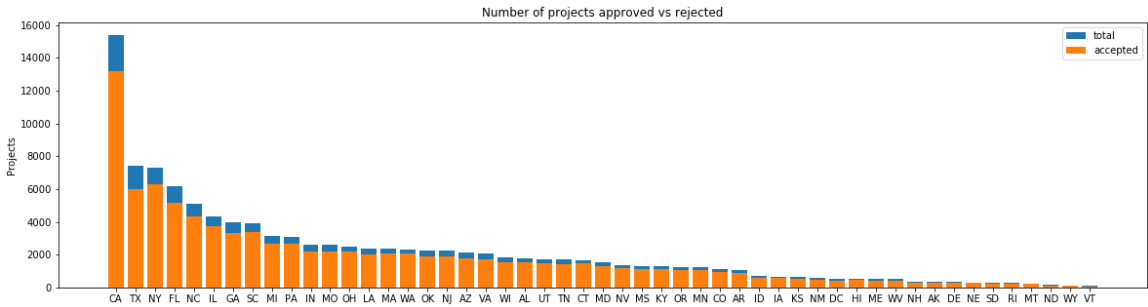
    temp.sort_values(by=['total'], inplace=True, ascending=False)

    if top:
        temp = temp[0:top]

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

In [12]:

```
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
=====				
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. CA state has higher approved rate. NC state has lower approved rate.

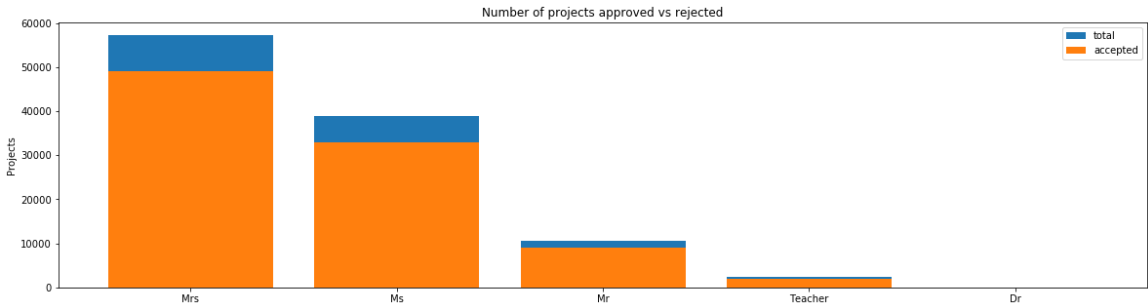
1.2.2 Univariate Analysis: teacher_prefix

In [13]:

```
project_data['teacher_prefix'] = project_data['teacher_prefix'].str.replace('.', '')
project_data['teacher_prefix'] = project_data['teacher_prefix'].fillna(project_data['teacher_prefix'].mode().iloc[0])
```

In [14]:

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



	teacher_prefix	project_is_approved	total	Avg
2	Mrs	49000	57272	0.855566
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	49000	57272	0.855566
3	Ms	32860	38955	0.843537
1	Mr	8960	10648	0.841473
4	Teacher	1877	2360	0.795339
0	Dr	9	13	0.692308

Summary : Number of projects approved are higher than number of projects rejected. for married woman approved rate is higher. Of Dr's out of 13 9 were approved.

1.2.3 Univariate Analysis: project_grade_category

In [15]:

```

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

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

grade_cat_list = []
for i in grade_categories:
    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 category based on space "Math & Science" => "Math", "&", "Science"
            j = j.replace('The', '') # if we have the words "The" we are going to replace it with '' (i.e removing 'The')
            j = j.replace(' ', '') # we are replacing all the ' ' (space) with '' (empty) ex: "Math & Science" => "Math&Science"
            j = j.replace('-', '_')
            temp += j.strip() + " #"
    temp = temp.replace('&', '_')
    grade_cat_list.append(temp.strip())

```

In [16]:

```

project_data['grade_category'] = grade_cat_list
project_data.drop(['project_grade_category'], axis=1, inplace=True)
project_data.head(2)

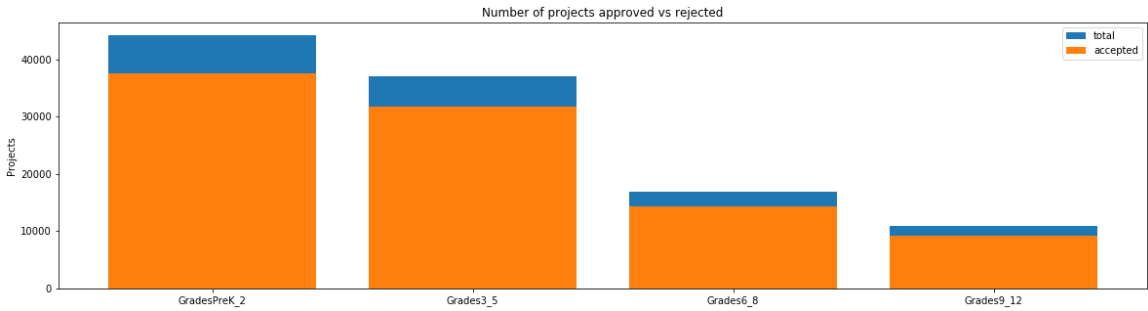
```

Out[16]:

	Unnamed: 0	id	teacher_id	teacher_prefix	school_s
0	160221	p253737	c90749f5d961ff158d4b4d1e7dc665fc	Mrs	IN
1	140945	p258326	897464ce9ddc600bced1151f324dd63a	Mr	FL

In [17]:

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



	grade_category	project_is_approved	total	Avg
3	GradesPreK_2	37536	44225	0.848751
0	Grades3_5	31729	37137	0.854377
1	Grades6_8	14258	16923	0.842522
2	Grades9_12	9183	10963	0.837636

=====

	grade_category	project_is_approved	total	Avg
3	GradesPreK_2	37536	44225	0.848751
0	Grades3_5	31729	37137	0.854377
1	Grades6_8	14258	16923	0.842522
2	Grades9_12	9183	10963	0.837636

Summary: On an average more than 83% of the projects are approved. Prekinder garden kids has higher approved rate. Students with grades 9-12 has lower approved rate.

1.2.4 Univariate Analysis: project_subject_categories

In [18]:

```

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

# https://www.geeksforgeeks.org/removing-stop-words-nltk-python/
# https://stackoverflow.com/questions/23669024/how-to-strip-a-specific-word-from-a-string
# https://stackoverflow.com/questions/8270092/remove-all-whitespace-in-a-string-in-python
cat_list = []
for i in categories:
    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 category based on space "Math & Science"=> "Math", "&", "Science"
            j=j.replace('The','') # if we have the words "The" we are going to replace it with ''(i.e removing 'The')
            j = j.replace(' ','') # we are replacing all the ' '(space) with ''(empty) ex:"Math & Science"=>"Math&Science"
            temp+=j.strip()+" " #" abc ".strip() will return "abc", remove the trailing spaces
    temp = temp.replace('&','_') # we are replacing the & value into
    cat_list.append(temp.strip())

```

In [19]:

```

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

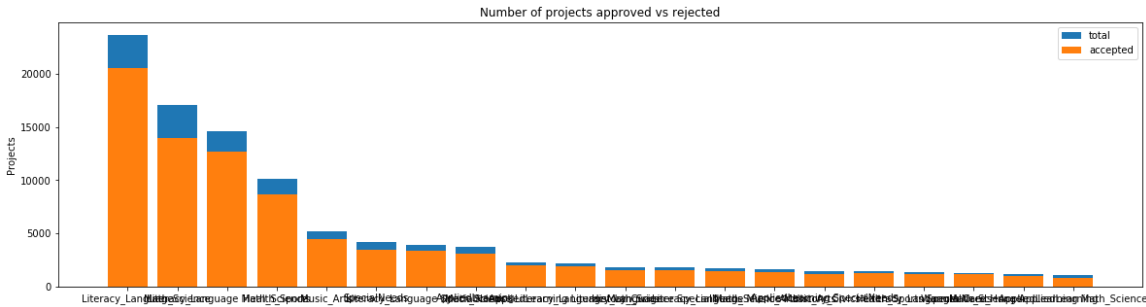
```

Out[19]:

	Unnamed: 0	id	teacher_id	teacher_prefix	school_s
0	160221	p253737	c90749f5d961ff158d4b4d1e7dc665fc	Mrs	IN
1	140945	p258326	897464ce9ddc600bced1151f324dd63a	Mr	FL

In [20]:

```
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 [21]:

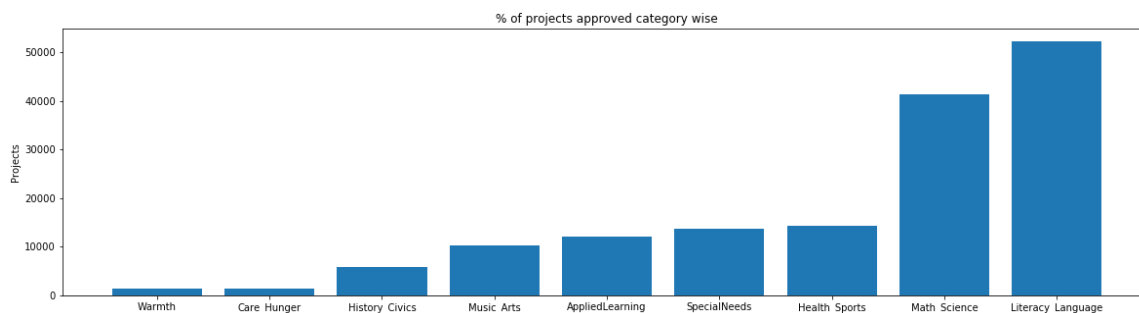
```
# count of all the words in corpus python: https://stackoverflow.com/a/22898595/4084039
from collections import Counter
my_counter = Counter()
for word in project_data['clean_categories'].values:
    my_counter.update(word.split())
```

In [22]:

```
# 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 approved category wise')
plt.xticks(ind, list(sorted_cat_dict.keys()))
plt.show()
```



In [23]:

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

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

Summary: Project of literacy language has higher approved rate. Project of Applied learning with math_science has lower approved rate.

1.2.5 Univariate Analysis: project_subject_subcategories

In [24]:

```

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

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

sub_cat_list = []
for i in sub_categories:
    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 category based on space "Math & Science"=> "Math", "&", "Science"
            j=j.replace('The','') # if we have the words "The" we are going to replace it with ''(i.e removing 'The')
            j = j.replace(' ','') # we are replacing all the ' '(space) with ''(empty) ex:"Math & Science"=>"Math&Science"
            temp +=j.strip()+" #" "abc ".strip() will return "abc", remove the trailing spaces
        temp = temp.replace('&','_')
    sub_cat_list.append(temp.strip())

```

In [25]:

```

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

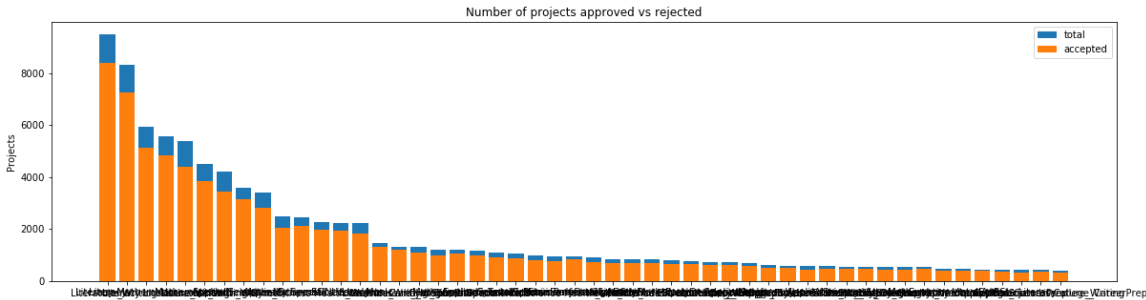
```

Out[25]:

	Unnamed: 0	id	teacher_id	teacher_prefix	school_s
0	160221	p253737	c90749f5d961ff158d4b4d1e7dc665fc	Mrs	IN
1	140945	p258326	897464ce9ddc600bced1151f324dd63a	Mr	FL

In [26]:

```
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	0.876
126				
127	ESL	349	421	0.828
979				
79	College_CareerPrep	343	421	0.814
727				
17	AppliedSciences Literature_Writing	361	420	0.859
524				
3	AppliedSciences College_CareerPrep	330	405	0.814
815				

In [27]:

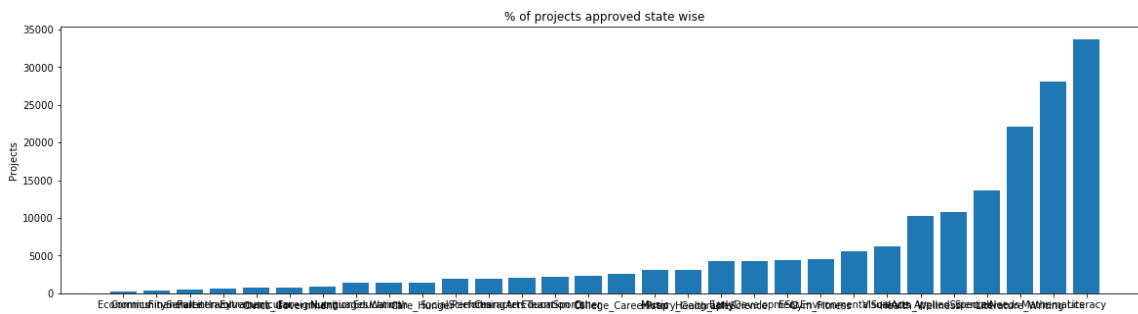
```
# count of all the words in corpus python: https://stackoverflow.com/a/22898595/4084039
from collections import Counter
my_counter = Counter()
for word in project_data['clean_subcategories'].values:
    my_counter.update(word.split())
```

In [28]:

```
# 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 approved state wise')
plt.xticks(ind, list(sorted_sub_cat_dict.keys()))
plt.show()
```



In [29]:

```
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

Summary : Projects with literacy has higher approved rate. Projects with appliedsciences college career prep has lower rate.

1.2.6 Univariate Analysis: Text features (Title)

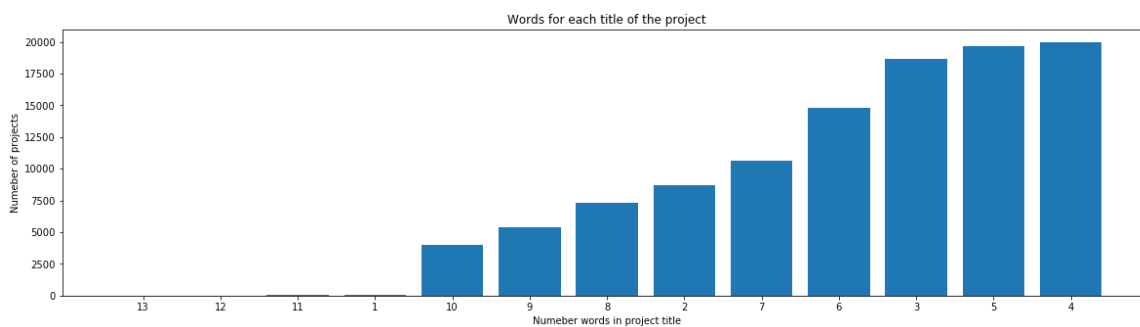
In [30]:

#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()
```



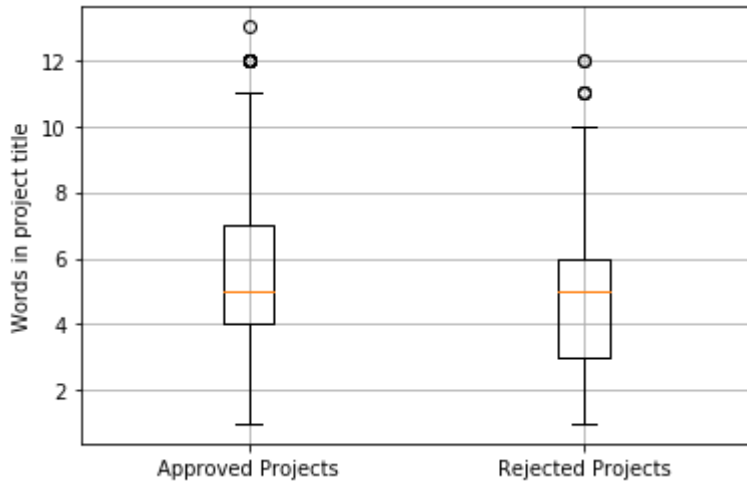
In [31]:

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

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

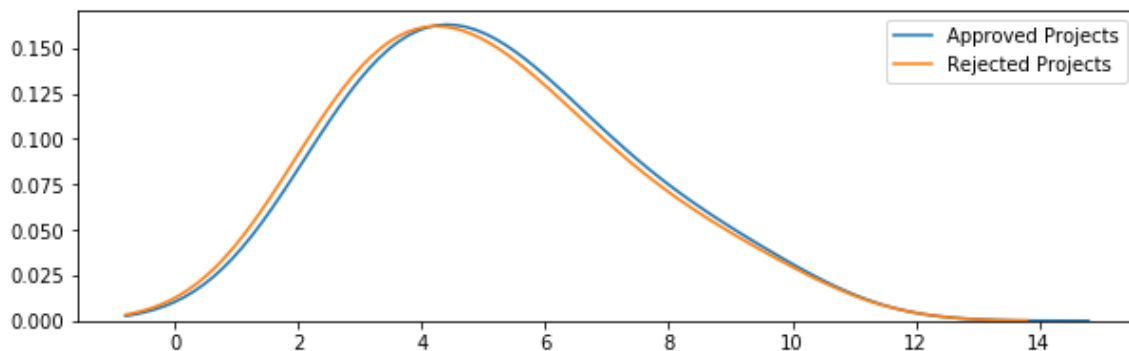
In [32]:

```
# 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 [33]:

```
plt.figure(figsize=(10,3))
sns.kdeplot(approved_title_word_count,label="Approved Projects", bw=0.6)
sns.kdeplot(rejected_title_word_count,label="Rejected Projects", bw=0.6)
plt.legend()
plt.show()
```



Summary: Number of words in titles approved are slightly higher than rejected.

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

In [34]:

```
# 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)
```

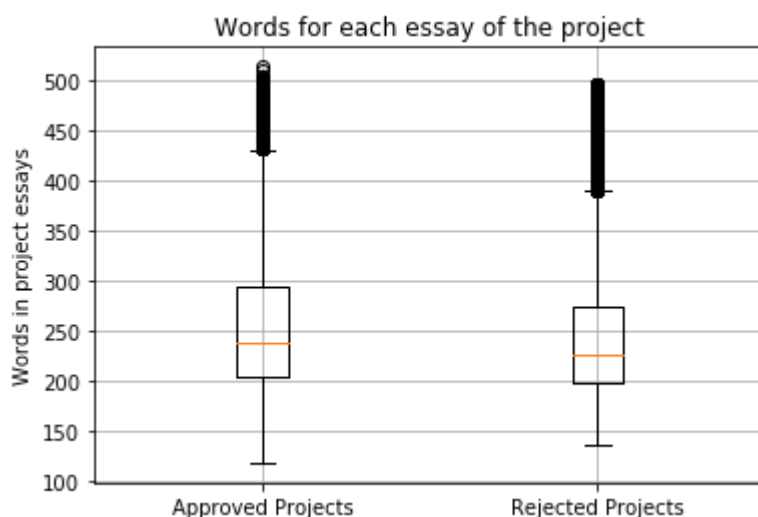
In [35]:

```
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
```

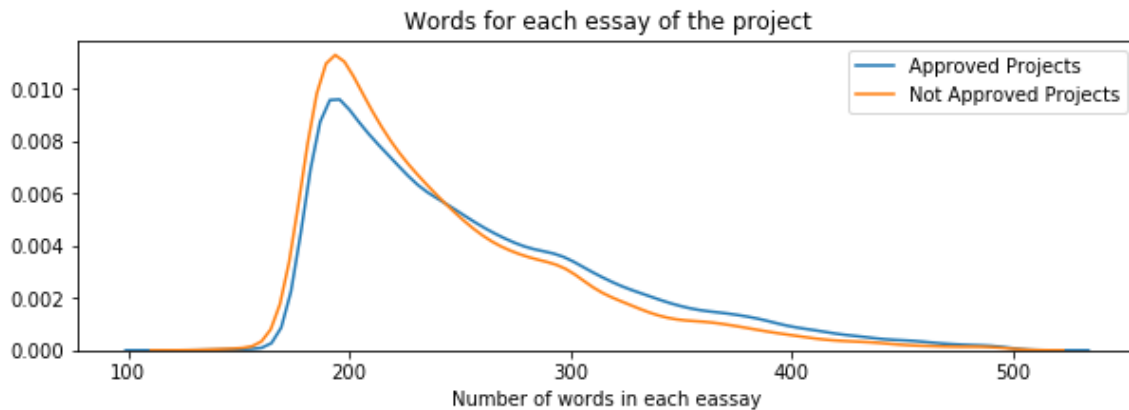
In [36]:

```
# 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 [37]:

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



Summary: Number of words in essay approved are slightly more than rejected.

1.2.8 Univariate Analysis: Cost per project

In [38]:

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

Out[38]:

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

In [39]:

```
# 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[39]:

	id	price	quantity
0	p0000001	459.56	7
1	p0000002	515.89	21

In [40]:

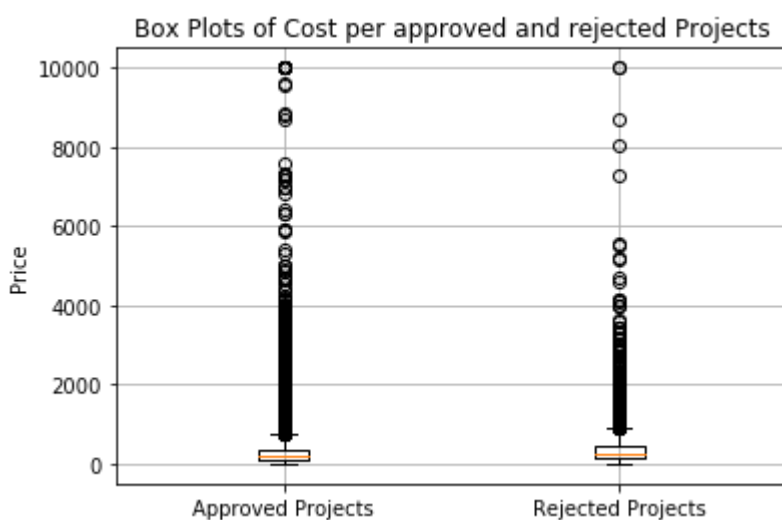
```
# join two dataframes in python:
project_data = pd.merge(project_data, price_data, on='id', how='left')
```

In [41]:

```
approved_price = project_data[project_data['project_is_approved']==1]['price'].values
rejected_price = project_data[project_data['project_is_approved']==0]['price'].values
```

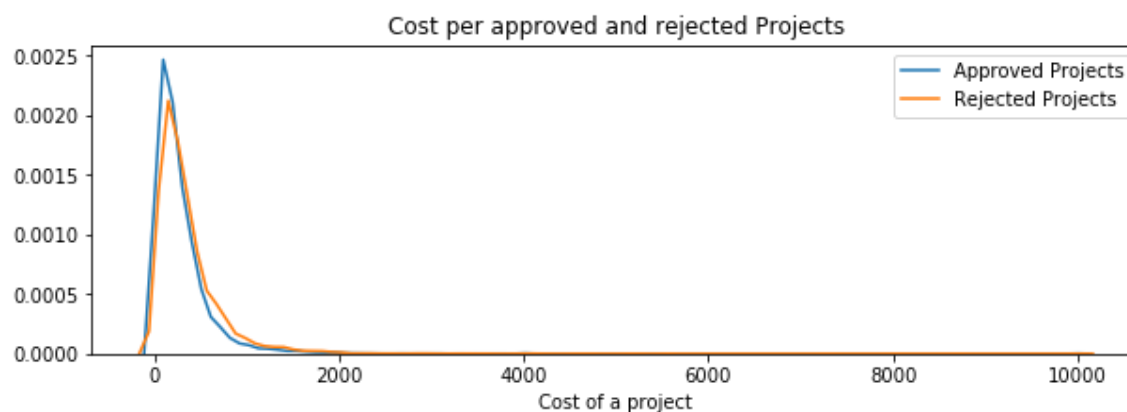
In [42]:

```
# 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 rejected Projects')
plt.xticks([1,2],('Approved Projects','Rejected Projects'))
plt.ylabel('Price')
plt.grid()
plt.show()
```



In [43]:

```
plt.figure(figsize=(10,3))
sns.distplot(approved_price, hist=False, label="Approved Projects")
sns.distplot(rejected_price, hist=False, label="Rejected Projects")
plt.title('Cost per approved and rejected Projects')
plt.xlabel('Cost of a project')
plt.legend()
plt.show()
```



Summary: The rate of approved projects based on price are slightly higher than rejected.

In [44]:

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

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

x = PrettyTable()
x.field_names = ["Percentile", "Approved Projects", "Rejected Projects"]

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

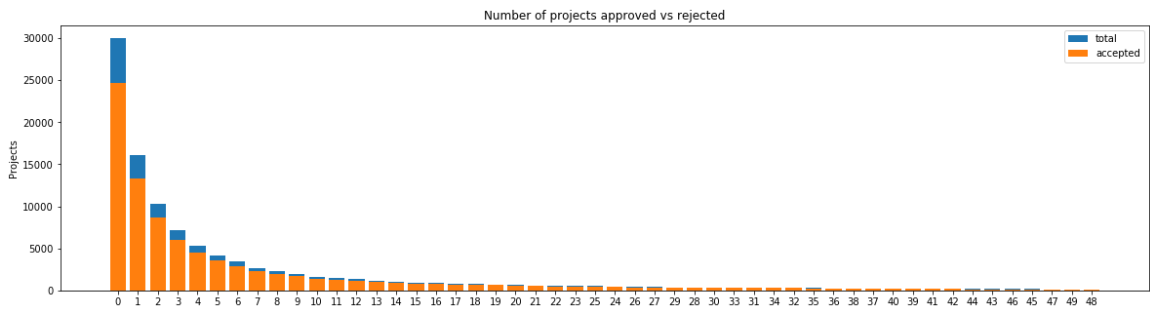
Percentile	Approved Projects	Rejected Projects
0	0.66	1.97
5	13.59	41.9
10	33.88	73.67
15	58.0	99.109
20	77.38	118.56
25	99.95	140.892
30	116.68	162.23
35	137.232	184.014
40	157.0	208.632
45	178.265	235.106
50	198.99	263.145
55	223.99	292.61
60	255.63	325.144
65	285.412	362.39
70	321.225	399.99
75	366.075	449.945
80	411.67	519.282
85	479.0	618.276
90	593.11	739.356
95	801.598	992.486
100	9999.0	9999.0

Summary: From the table we can understand that based on price rejection is higher

1.2.9 Univariate Analysis: teacher_number_of_previously_posted_projects

In [45]:

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



	teacher_number_of_previously_posted_projects	project_is_approved	tota
1 \			
0	0	24652	3001
4			
1	1	13329	1605
8			
2	2	8705	1035
0			
3	3	5997	711
0			
4	4	4452	526
6			

	Avg
0	0.821350
1	0.830054
2	0.841063
3	0.843460
4	0.845423

	teacher_number_of_previously_posted_projects	project_is_approved	tot
a1 \			
46	46	149	1
64			
45	45	141	1
53			
47	47	129	1
44			
49	49	128	1
43			
48	48	135	1
40			

	Avg
46	0.908537
45	0.921569
47	0.895833
49	0.895105
48	0.964286

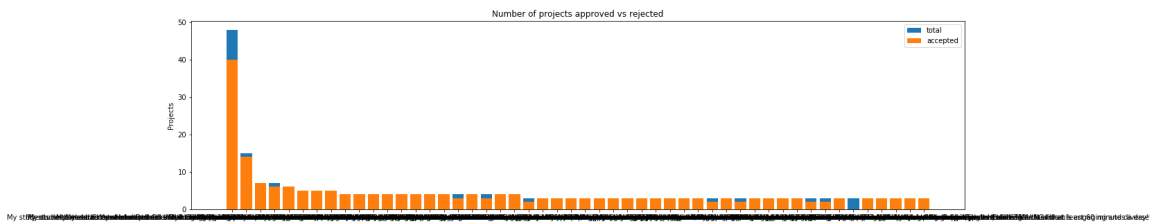
1.2.10 Univariate Analysis: project_resource_summary</h3>

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

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

In [46]:

```
univariate_barplots(project_data, 'project_resource_summary', 'project_is_approved', to
p=50)
```



	project_resource_summary	project_is_appro
ved \		
56539	My students need electronic tablets to do all ...	40
10193	My students need Chromebooks to do all the thi...	14
18828	My students need a Dell Chromebook 3120 and a ...	7
51417	My students need chromebooks to do all the thi...	6
18819	My students need a Dell Chromebook 3120 11 6 C...	6

	total	Avg
56539	48	0.833333
10193	15	0.933333
18828	7	1.000000
51417	7	0.857143
18819	6	1.000000

=====

	project_resource_summary	project_is_appro
ved \		
34033	My students need a variety of books for our cl...	3
42108	My students need an iPad to be prepared for th...	3
1705	My students need 2 Chromebooks, and 2 console ...	3
7837	My students need 7 Hokki stools to get ACTIVE ...	3
91743	My students need technology in the classroom. ...	3

	total	Avg
34033	3	1.0
42108	3	1.0
1705	3	1.0
7837	3	1.0
91743	3	1.0

In [47]:

```
summary = []  
  
for a in project_data["project_resource_summary"] :  
    summary.append(a)  
  
summary[0:10]
```

Out[47]:

```
['My students need opportunities to practice beginning reading skills in English at home.',  
 'My students need a projector to help with viewing educational programs',  
 'My students need shine guards, athletic socks, Soccer Balls, goalie gloves, and training materials for the upcoming Soccer season.',  
 'My students need to engage in Reading and Math in a way that will inspire them with these Mini iPads!',  
 'My students need hands on practice in mathematics. Having fun and personalized journals and charts will help them be more involved in our daily Math routines.',  
 'My students need movement to be successful. Being that I have a variety of students that have all different types of needs, flexible seating would assist not only these students with special needs, but all students.',  
 'My students need some dependable laptops for daily classroom use for reading and math.',  
 'My students need ipads to help them access a world of online resources that will spark their interest in learning.',  
 "My students need three devices and three management licenses for small group's easy access to newly-implemented online programs--Go Noodle Plus, for increased in-class physical activity and Light Sail, an interactive reading program.",  
 'My students need great books to use during Independent Reading, Read Alouds, Partner Reading and Author Studies.']
```

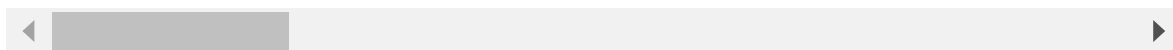
In [48]:

```
# https://stackoverflow.com/a/19859308/8089731
def Numbers(summary):
    return any(i.isdigit() for i in summary)
dt = project_data[['id','project_resource_summary']]
dt = pd.DataFrame(data=dt)
dt.columns = ['id','digits_in_summary']
dt['digits_in_summary'] = dt['digits_in_summary'].map(Numbers)
dt['digits_in_summary'] = dt['digits_in_summary'].astype(int)
project_data = pd.merge(project_data, dt, on='id', how='left')
project_data.head()
```

Out[48]:

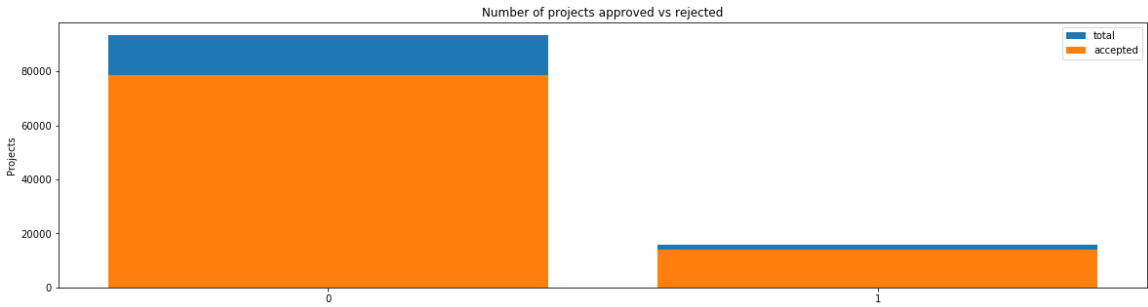
	Unnamed: 0	id	teacher_id	teacher_prefix	school_s
0	160221	p253737	c90749f5d961ff158d4b4d1e7dc665fc	Mrs	IN
1	140945	p258326	897464ce9ddc600bced1151f324dd63a	Mr	FL
2	21895	p182444	3465aaf82da834c0582ebd0ef8040ca0	Ms	AZ
3	45	p246581	f3cb9bffbba169bef1a77b243e620b60	Mrs	KY
4	172407	p104768	be1f7507a41f8479dc06f047086a39ec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

5 rows × 21 columns



In [49]:

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



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

Summary: The summary has digits which influence the data. Digits in the summary has higher approved rate.

1.3 Text preprocessing

1.3.1 Essay Text

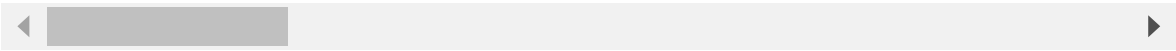
In [50]:

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

Out[50]:

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

2 rows × 21 columns



In [51]:

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


My students are English learners that are working on English as their second or third languages. We are a melting pot of refugees, immigrants, and native-born Americans bringing the gift of language to our school. \r\n\r\n We have over 24 languages represented in our English Learner program with students at every level of mastery. We also have over 40 countries 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.\r\n\r\nThe limits of your language are the limits of your world.\r\n\r\n-Ludwig Wittgenstein Our English 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. Sometimes this creates barriers for parents to be able to help their child learn phonetics, letter recognition, and other reading skills.\r\n\r\n\r\nBy providing these dvd's and players, 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 offered to be a part of this program. These educational videos will be specially chosen by the English Learner Teacher and will be sent home regularly to watch. The videos are to help the child develop early reading skills.\r\n\r\n\r\nParents that do not have access to a dvd player will have the opportunity to check out a dvd player to use for the year. The plan is to use these videos and educational dvd's for the years to come for other EL students.\r\n\r\nnannan

=====

The 51 fifth grade students that will cycle through my classroom this year all love learning, at least most of the time. At our school, 97.3% of the students receive free or reduced price lunch. Of the 560 students, 97.3% are minority students. \r\n\r\nThe school has a vibrant community that loves to get together and celebrate. Around Halloween there is a whole 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 school 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 individual one, they will be used in a variety of ways. During independent reading time they will be used as special chairs students will each use on occasion. I will utilize them in place of chairs at my small group tables during math and reading times. The rest of the day they will be used by the students who need the highest amount of movement in their life in order to stay focused on school.\r\n\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 are always moving, but at the same time doing their work. Anytime the students get to pick where they can sit, the Hokki Stools are the first to be taken. There are always students who head over to the kidney table to get one of the stools who are disappointed as there are not enough of them. \r\n\r\n\r\nWe ask a lot of students to sit for 7 hours a day. The Hokki stools will be a compromise that allow my students to do desk work and move at the same time. These stools will help students to meet their 60 minutes a day of movement by allowing them to activate their core muscles for balance while they sit. For many of my students, 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\n\r\nMy class is made up of 28 wonderfully unique boys and girls of mixed r

aces in Arkansas.\r\nThey attend a Title I school, which means there is a high enough percentage of free and reduced-price lunch to qualify. Our school is an \"open classroom\" concept, which is very unique as there are no walls separating the classrooms. These 9 and 10 year-old students are very eager learners; they are like sponges, absorbing all the information and 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 nets, 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 in 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 classroom ready for their first day of 4th grade. This kind gesture will set the tone before even the first day of school! The nautical thank you 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 from day one.\r\n\r\nIt costs lost of money out of my own pocket on resources to get our classroom ready. Please consider helping with this project to make our new school year a very successful one. Thank you!nannan

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

=====
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 makeup is 97.6% African-American, making up the largest segment of the student body. A typical 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 children and we focus not only on academics but one smart, effective, efficient, and disciplined students with good character. In our classroom we can utilize the Bluetooth for swift transitions during class. I use a speaker which doesn't amplify the sound enough to receive the message. Due to the volume of my speaker my students can't hear videos or books clearly and it isn't making the lessons as meaningful. But with the bluetooth speaker my students will be able to hear and I can stop, pause and replay it at any time.\r\n\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 pictures for students to learn about different letters and it is more accessible.nannan

=====

In [52]:

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

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

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

In [53]:

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

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

=====

In [54]:

```
# \r \n \t remove from string python: http://texthandler.com/info/remove-line-breaks-python/
sent = sent.replace('\r', ' ')
sent = sent.replace('\n', ' ')
sent = sent.replace('\t', ' ')
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 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 felt like you had ants in your pants and you needed to groove and move as you were in a meeting? This is how my kids feel all the time. They want to be able to move as they learn or so they say. Wobble chairs are the answer and I love them because they develop their core, which enhances gross motor and in turn fine motor skills. They also want to learn through games, my kids do not want to sit and do worksheets. They want to learn to count by jumping and playing. Physical engagement is the key to our success. The number toss and color and shape mats can make that happen. My students will forget they are doing work and just have the fun a 6 year old deserves. nannan

In [55]:

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

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

In [56]:

```
# https://gist.github.com/sebleier/554280
# we are removing the words from the stop words list: 'no', 'nor', 'not'
stopwords= ['i', 'me', 'my', 'myself', 'we', 'our', 'ours', 'ourselves', 'you', "you'r
e", "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', 't
hey', 'them', 'their',\
            'theirs', 'themselves', 'what', 'which', 'who', 'whom', 'this', 'that', "th
at'll", 'these', 'those', \
            'am', 'is', 'are', 'was', 'were', 'be', 'been', 'being', 'have', 'has', 'ha
d', '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', 'ov
er', 'under', 'again', 'further',\
            'then', 'once', 'here', 'there', 'when', 'where', 'why', 'how', 'all', 'an
y', '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", 'no
w', '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', 'migh
tn', "mighntn't", 'mustn',\
            "mustn't", 'needn', "needn't", 'shan', "shan't", 'shouldn', "shouldn't", 'w
asn', "wasn't", 'weren', "weren't", \
            'won', "won't", 'wouldn', "wouldn't"]
```

In [57]:

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

```
100%|████████████████████████████████████████████████████████████████████████████████|
109248/109248 [01:51<00:00, 976.88it/s]
```

In [58]:

```
# after preprocessing
preprocessed_essays[20000]
```

Out[58]:

```
'my kindergarten students varied disabilities ranging speech language delays cognitive delays 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 receive free reduced price lunch despite disabilities limitations students love coming school come eager learn explore have ever felt like ants pants needed groove move meeting this kids feel time the want able move learn say wobble chairs answer i love develop core enhances gross motor turn fine motor skills they also want learn games kids not want sit worksheets they want learn count jumping playing physical engagement key success the number toss color shape mats make happen my students forget work fun 6 year old deserves nannan'
```

1.3.2 Project title Text

In [59]:

```
# 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 [60]:

```
sent = decontracted(project_data['project_title'].values[37000])
print(sent)
print('='*50)
```

Focus our CORE!

```
=====
```

In [61]:

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

Focus our CORE!

In [62]:

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

Focus our CORE

In [63]:

```
# 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'r
e", "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', 't
hey', 'them', 'their',\
            'theirs', 'themselves', 'what', 'which', 'who', 'whom', 'this', 'that', "th
at'll", 'these', 'those', \
            'am', 'is', 'are', 'was', 'were', 'be', 'been', 'being', 'have', 'has', 'ha
d', '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', 'ov
er', 'under', 'again', 'further',\
            'then', 'once', 'here', 'there', 'when', 'where', 'why', 'how', 'all', 'an
y', '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", 'no
w', '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', 'migh
tn', "mighntn't", 'mustn',\
            "mustn't", 'needn', "needn't", 'shan', "shan't", 'shouldn', "shouldn't", 'w
asn', "wasn't", 'weren', "weren't", \
            'won', "won't", 'wouldn', "wouldn't"]
```


1.4.1 Vectorizing Categorical data

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

1.4.1.1 vectorizing clean categories

In [66]:

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

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

['Warmth', 'Care_Hunger', 'History_Civics', 'Music_Arts', 'AppliedLearnin
g', 'SpecialNeeds', 'Health_Sports', 'Math_Science', 'Literacy_Language']
Shape of matrix after one hot encodig (109248, 9)
```

1.4.1.2 Vectorizing clean subcategories

In [67]:

```
# 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'].value
s)
print("Shape of matrix after one hot encodig ",sub_categories_one_hot.shape)

['Economics', 'CommunityService', 'FinancialLiteracy', 'ParentInvolvemen
t', 'Extracurricular', 'Civics_Government', 'ForeignLanguages', 'Nutrition
Education', 'Warmth', 'Care_Hunger', 'SocialSciences', 'PerformingArts',
'CharacterEducation', 'TeamSports', 'Other', 'College_CareerPrep', 'Musi
c', 'History_Geography', 'Health_LifeScience', 'EarlyDevelopment', 'ESL',
'Gym_Fitness', 'EnvironmentalScience', 'VisualArts', 'Health_Wellness', 'A
ppliedSciences', 'SpecialNeeds', 'Literature_Writing', 'Mathematics', 'Lit
eracy']
Shape of matrix after one hot encodig (109248, 30)
```

1.4.1.3 Vectorizing school state

In [68]:

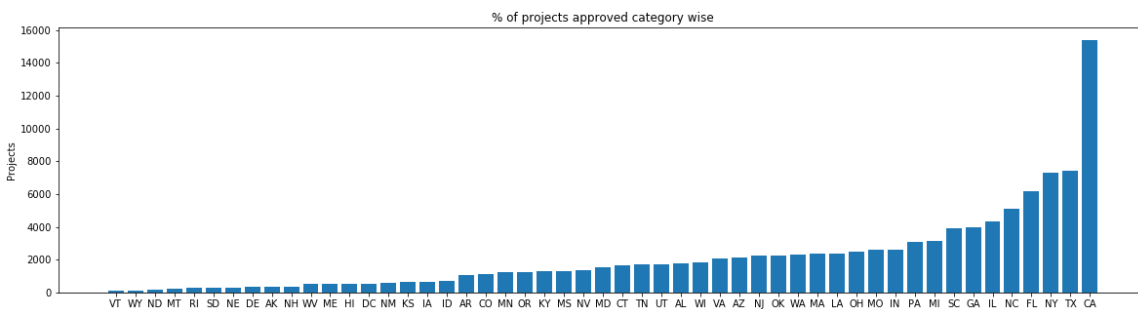
```
# count of all the words in corpus python: https://stackoverflow.com/a/22898595/4084039
from collections import Counter
my_counter = Counter()
for word in project_data['school_state'].values:
    my_counter.update(word.split())
```

In [69]:

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

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

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



In [70]:

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

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

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

1.4.1.4 vectorizing teacher prefix

In [71]:

```
project_data['teacher_prefix']= project_data['teacher_prefix'].str.replace('.', '')

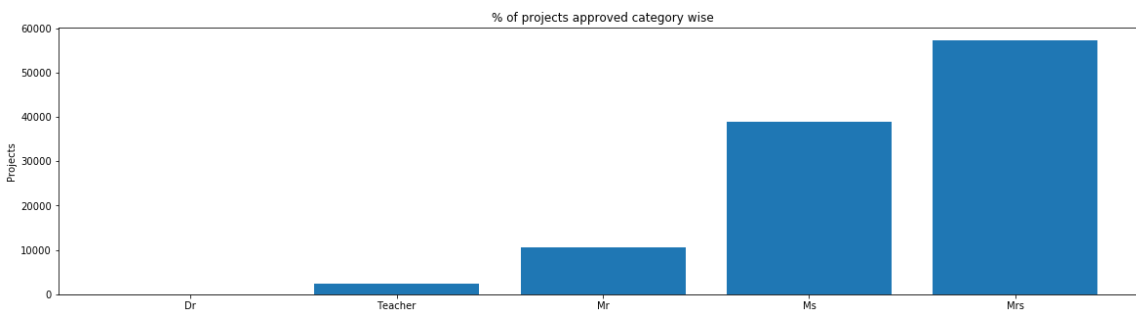
# count of all the words in corpus python: https://stackoverflow.com/a/22898595/4084039
from collections import Counter
my_counter = Counter()
for word in project_data['teacher_prefix'].values:
    my_counter.update(str(word).split())
```

In [72]:

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

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

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



In [73]:

```
vectorizer = CountVectorizer(vocabulary = list(sorted_teacher_dict.keys()), lowercase=False, binary=True)
vectorizer.fit(project_data['teacher_prefix'].values.astype('U'))
print(vectorizer.get_feature_names())
```

```
teacher_ohe = vectorizer.transform(project_data['teacher_prefix'].values.astype('U'))
print("Shape of matrix after one hot encoding ", teacher_ohe.shape)
print(np.array(project_data['teacher_prefix']))
```

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

In [74]:

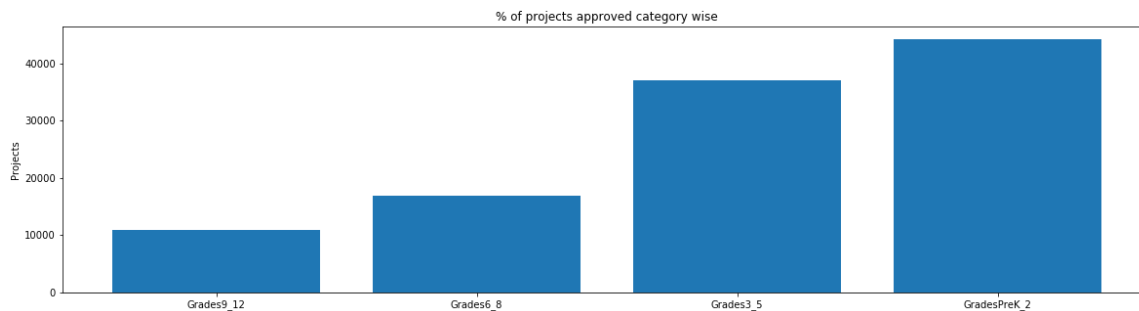
```
project_data['grade_category'] = project_data['grade_category'].str.replace('-', '_')
# count of all the words in corpus python: https://stackoverflow.com/a/22898595/4084039
from collections import Counter
my_counter = Counter()
for word in project_data['grade_category'].values:
    my_counter.update(str(word).split())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

In [75]:

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

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

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



In [76]:

```
vectorizer = CountVectorizer(vocabulary=list(sorted_grade_dict.keys()), lowercase=False,
                             binary=True)
vectorizer.fit(project_data['grade_category'].values)
print(vectorizer.get_feature_names())
```

```
grade_ohe = vectorizer.transform(project_data['grade_category'].values)
print("Shape of matrix after one hot encoding ", grade_ohe.shape)
print(np.array(project_data['grade_category']))
```

```
['Grades9_12', 'Grades6_8', 'Grades3_5', 'GradesPreK_2']
Shape of matrix after one hot encoding (109248, 4)
['GradesPreK_2' 'Grades6_8' 'Grades6_8' ... 'GradesPreK_2' 'Grades3_5'
 'Grades6_8']
```

1.4.2 Vectorizing Text data

1.4.2.1 Bag of words

In [77]:

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

Shape of matrix after one hot encoding (109248, 16623)

1.4.2.2 Bag of Words on project_title</h4>

In [78]:

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

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

1.4.2.3 TFIDF vectorizer

In [79]:

```
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 encoding ",text_tfidf.shape)
```

Shape of matrix after one hot encoding (109248, 16623)

1.4.2.4 TFIDF Vectorizer on project_title</h4>

In [80]:

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

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

1.4.2.5 Using Pretrained Models: Avg W2V

In [81]:

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

# =====
#Output:

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

# =====

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

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

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

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

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

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



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

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

1.4.3 Vectorizing Numerical features, previous projects

```
# check this one: https://www.youtube.com/watch?v=0H0qOcln3Z4&t=530s
# standardization sklearn: https://scikit-learn.org/stable/modules/generated/sklearn.preprocessing.StandardScaler.html
from sklearn.preprocessing import StandardScaler

# price_standardized = standardScaler.fit(project_data['price'].values)
# this will rise the error
# ValueError: Expected 2D array, got 1D array instead: array=[725.05 213.03 329. ...
# 399. 287.73 5.5 ].
# Reshape your data either using array.reshape(-1, 1)

price_scalar = StandardScaler()
price_scalar.fit(project_data['price'].values.reshape(-1,1)) # finding the mean and standard 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 mean and variance.
price_standardized = price_scalar.transform(project_data['price'].values.reshape(-1, 1))
```

file:///C:/Users/SUBHODAYA KUMAR/Downloads/DonorsChoose EDA TSNE (3).html

In [90]:

```
price_standardized
```

Out[90]:

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

In [91]:

```
price_scalar.fit(project_data['teacher_number_of_previously_posted_projects'].values.reshape(-1,1)) # finding the mean and standard 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 mean and variance.
proj_standardized = price_scalar.transform(project_data['teacher_number_of_previously_posted_projects'].values.reshape(-1, 1))
```

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

In [92]:

```
proj_standardized
```

Out[92]:

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

1.4.4 Merging all the above features

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

In [93]:

```
print(categories_one_hot.shape)
print(sub_categories_one_hot.shape)
print(school_ohe.shape)
print(teacher_ohe.shape)
print(grade_ohe.shape)
print(text_bow.shape)
print(title_bow.shape)
print(price_standardized.shape)
print(proj_standardized.shape)
```

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

In [94]:

```
# merge two sparse matrices: https://stackoverflow.com/a/19710648/4084039
from scipy.sparse import hstack
# with the same hstack function we are concatenating a sparse matrix and a dense matrix
:
X_bow = hstack((categories_one_hot, sub_categories_one_hot, price_standardized, proj_standardized, title_bow, grade_ohe, school_ohe, grade_ohe,))
X_bow.shape
```

Out[94]:

```
(109248, 3429)
```

In [95]:

```
print(categories_one_hot.shape)
print(sub_categories_one_hot.shape)
print(school_ohe.shape)
print(teacher_ohe.shape)
print(grade_ohe.shape)
print(text_tfidf.shape)
print(title_tfidf.shape)
print(price_standardized.shape)
print(proj_standardized.shape)
```

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

In [96]:

```
# merge two sparse matrices: https://stackoverflow.com/a/19710648/4084039
from scipy.sparse import hstack
# with the same hstack function we are concatenating a sparse matrix and a dense matrix
:
X_tfidf = hstack((categories_one_hot, sub_categories_one_hot, price_standardized, proj_standardized, title_tfidf, grade_ohe, school_ohe, grade_ohe))
X_tfidf.shape
```

Out[96]:

(109248, 3429)

In [98]:

```
# merge two sparse matrices: https://stackoverflow.com/a/19710648/4084039
from scipy.sparse import hstack
# with the same hstack function we are concatenating a sparse matrix and a dense matrix
:
X_avg_w2v = hstack((categories_one_hot, sub_categories_one_hot, price_standardized, proj_standardized, avg_w2v_vectors_title, grade_ohe, school_ohe, grade_ohe))
X_avg_w2v.shape
```

Out[98]:

(109248, 400)

In [100]:

```
# merge two sparse matrices: https://stackoverflow.com/a/19710648/4084039
from scipy.sparse import hstack
# with the same hstack function we are concatenating a sparse matrix and a dense matrix
:
X_tfidf_w2v = hstack((categories_one_hot, sub_categories_one_hot, price_standardized, proj_standardized, tfidf_w2v_vectors_title, grade_ohe, school_ohe, grade_ohe))
X_tfidf_w2v.shape
```

Out[100]:

(109248, 400)

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

In [101]:

```
from sklearn.manifold import TSNE

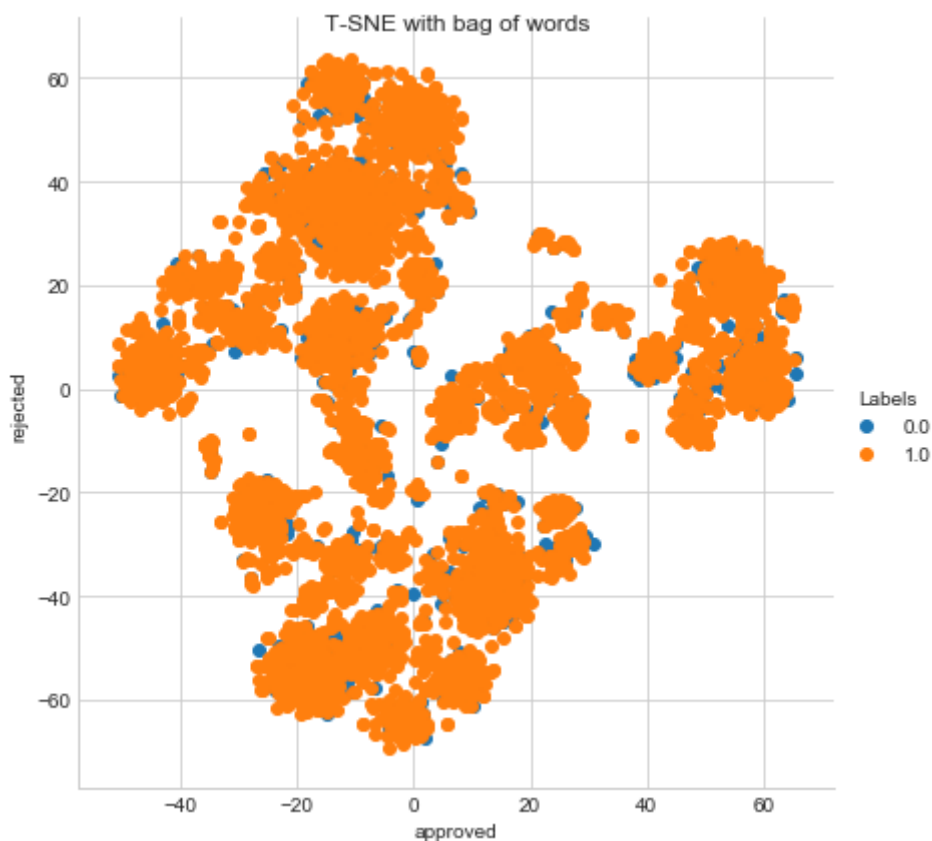
#https://stackoverflow.com/a/30175105/8089731
X_bow = X_bow.tocsr()
X_bow = X_bow[0:5000]
print(X_bow.shape)

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

tsne = TSNE(n_components=2, perplexity=30, learning_rate=200)
X1_embedding = tsne.fit_transform(X_bow.toarray())
for_tsne1 = np.hstack((X1_embedding, y.values.reshape(-1,1)))
for_tsne_df1 = pd.DataFrame(data=for_tsne1, columns=['approved', 'rejected', 'Labels'])
sns.set_style("whitegrid");
sns.FacetGrid(for_tsne_df1, hue="Labels", size=6)\
    .map(plt.scatter, "approved", "rejected") \
    .add_legend()\
    .fig.suptitle("T-SNE with bag of words ");
plt.show();
```

(5000, 3429)

(5000,)



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

In [102]:

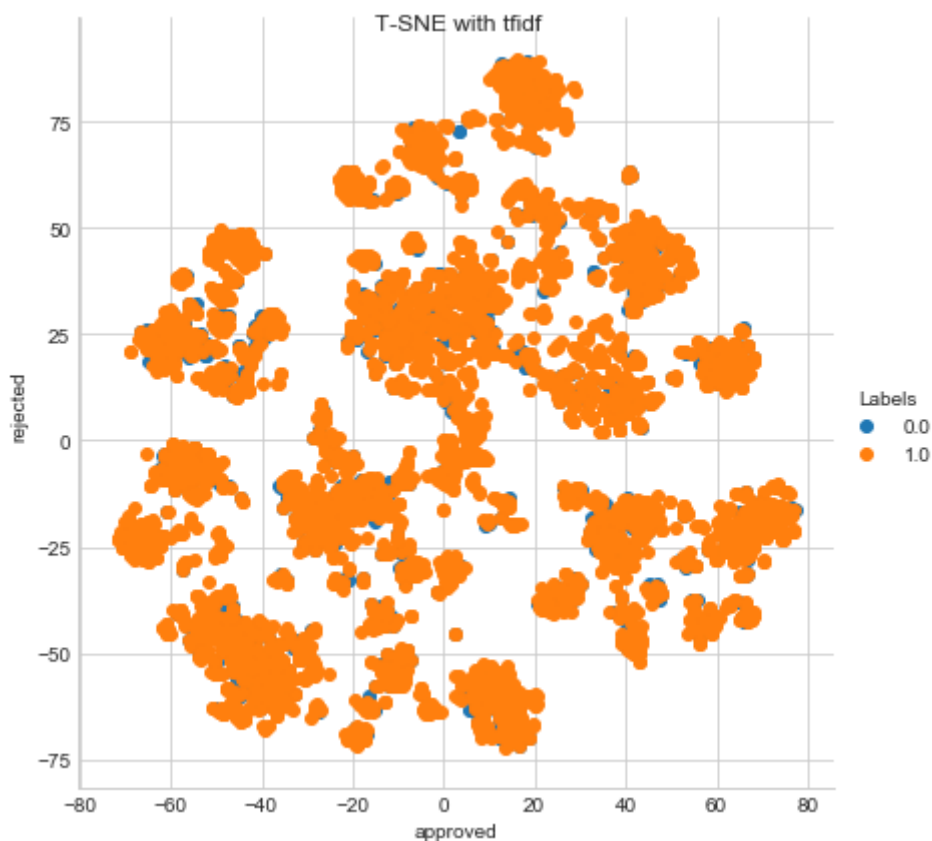
```
#https://stackoverflow.com/a/30175105/8089731
X_tfidf = X_tfidf.tocsr()
X_tfidf = X_tfidf[0:5000]
print(X_tfidf.shape)

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

tsne = TSNE(n_components=2, perplexity=30, learning_rate=200)
X1_embedding = tsne.fit_transform(X_tfidf.toarray())
for_tsne1 = np.hstack((X1_embedding, y.values.reshape(-1,1)))
for_tsne_df1 = pd.DataFrame(data=for_tsne1, columns=['approved', 'rejected', 'Labels'])
sns.set_style("whitegrid");
sns.FacetGrid(for_tsne_df1, hue="Labels", size=6)\
    .map(plt.scatter, "approved", "rejected") \
    .add_legend()\
    .fig.suptitle("T-SNE with tfidf ");
plt.show();
```

(5000, 3429)

(5000,)



2.3 TSNE with `avg_w2v` encoding of `project_title` feature

In [103]:

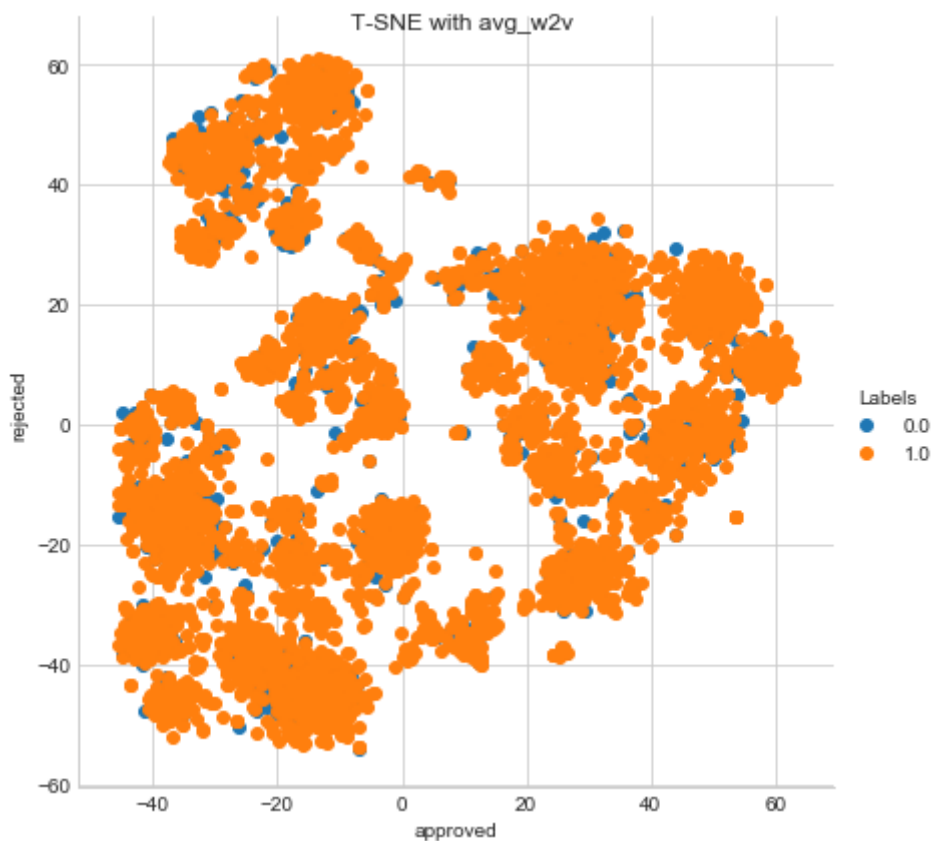
```
#https://stackoverflow.com/a/30175105/8089731
X_avg_w2v = X_avg_w2v.tocsr()
X_avg_w2v = X_bow[0:5000]
print(X_avg_w2v.shape)

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

tsne = TSNE(n_components=2, perplexity=30, learning_rate=200)
X1_embedding = tsne.fit_transform(X_avg_w2v.toarray())
for_tsne1 = np.hstack((X1_embedding, y.values.reshape(-1,1)))
for_tsne_df1 = pd.DataFrame(data=for_tsne1, columns=['approved', 'rejected', 'Labels'])
sns.set_style("whitegrid");
sns.FacetGrid(for_tsne_df1, hue="Labels", size=6)\
    .map(plt.scatter, "approved", "rejected") \
    .add_legend()\
    .fig.suptitle("T-SNE with avg_w2v");
plt.show();
```

(5000, 3429)

(5000,)



2.4 TSNE with `tfidf_w2v` encoding of `project_title` feature

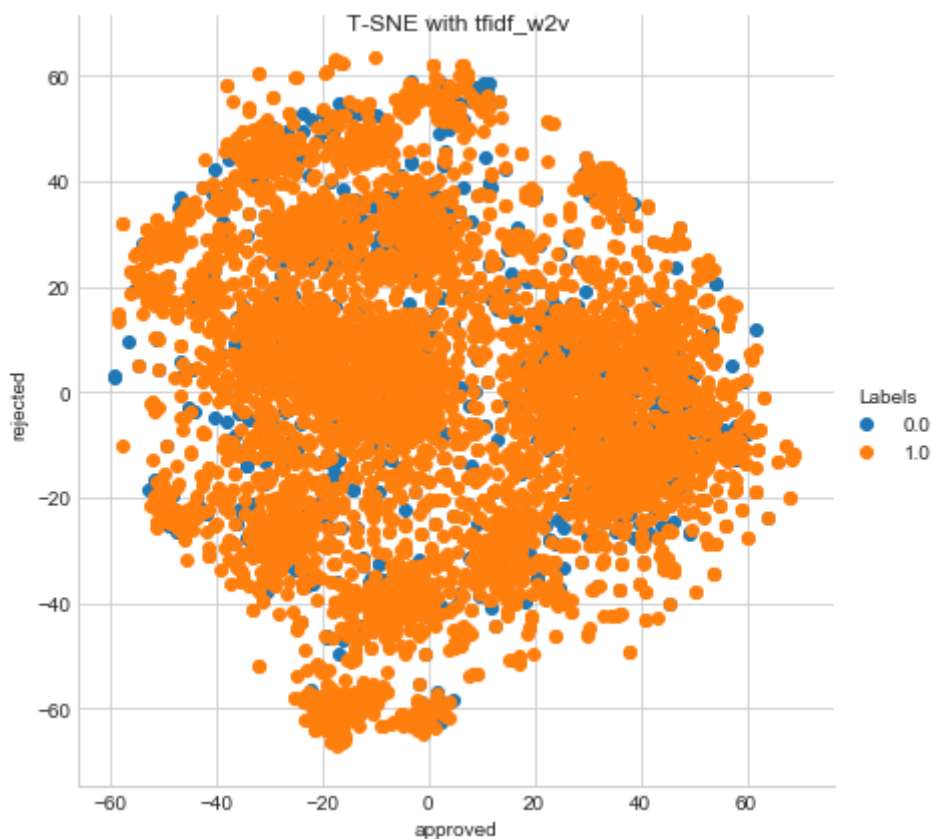
In [104]:

```
#https://stackoverflow.com/a/30175105/8089731
X_tfidf_w2v = X_tfidf_w2v.tocsr()
X_tfidf_w2v = X_tfidf_w2v[0:5000]
print(X_tfidf_w2v.shape)

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

tsne = TSNE(n_components=2, perplexity=30, learning_rate=200)
X1_embedding = tsne.fit_transform(X_tfidf_w2v.toarray())
for_tsne1 = np.hstack((X1_embedding, y.values.reshape(-1,1)))
for_tsne_df1 = pd.DataFrame(data=for_tsne1, columns=['approved', 'rejected', 'Labels'])
sns.set_style("whitegrid");
sns.FacetGrid(for_tsne_df1, hue="Labels", size=6)\
    .map(plt.scatter, "approved", "rejected") \
    .add_legend()\
    .fig.suptitle("T-SNE with tfidf_w2v ");
plt.show();
```

```
(5000, 400)
(5000,)
```



2.5 TSNE for encoding all `project_title` feature

In [105]:

```
X_all = hstack((categories_one_hot, sub_categories_one_hot, price_standardized, proj_standardized, title_bow, title_tfidf, avg_w2v_vectors_title, tfidf_w2v_vectors_title, grade_ohe, school_ohe, grade_ohe))
X_all.shape
```

Out[105]:

(109248, 7358)

In [106]:

```
#https://stackoverflow.com/a/30175105/8089731
```

```
X_all = X_all.tocsr()
```

```
X_all = X_all[0:4000]
```

```
print(X_all.shape)
```

```
y = project_data['project_is_approved'][0:4000]
```

```
print(y.shape)
```

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

```
X1_embedding = tsne.fit_transform(X_all.toarray())
```

```
for_tsne1 = np.hstack((X1_embedding, y.values.reshape(-1,1)))
```

```
for_tsne_df1 = pd.DataFrame(data=for_tsne1, columns=['approved', 'rejected', 'Labels'])
```

```
sns.set_style("whitegrid");
```

```
sns.FacetGrid(for_tsne_df1, hue="Labels", size=6)\
```

```
    .map(plt.scatter, "approved", "rejected") \
```

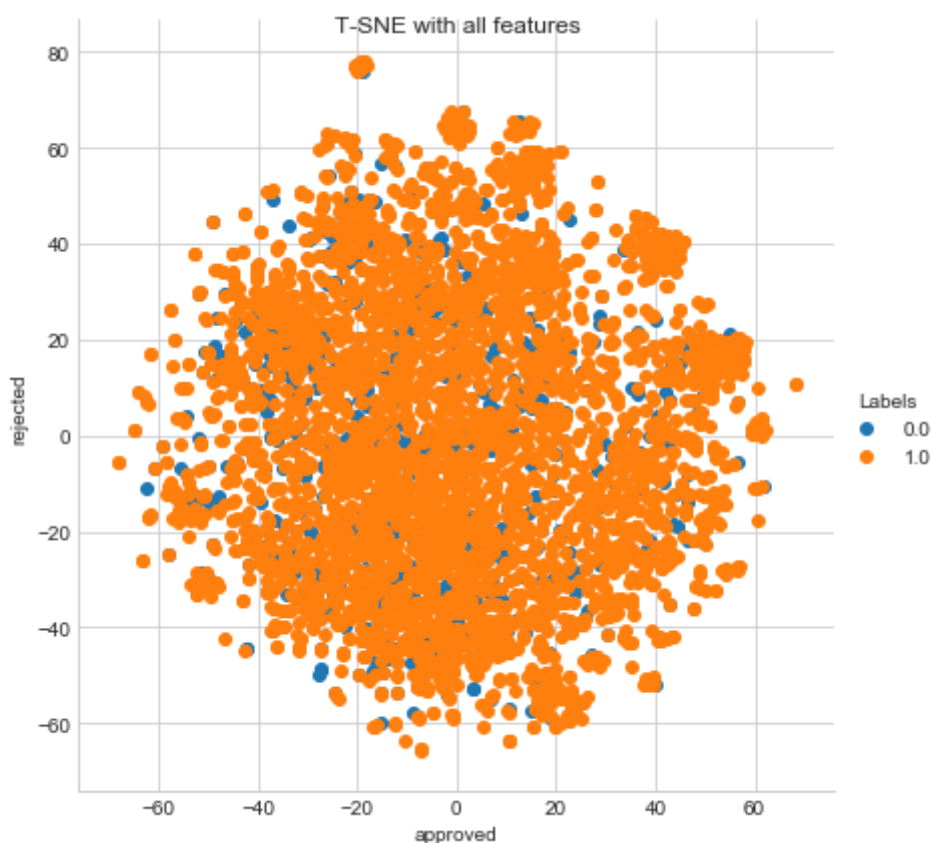
```
    .add_legend()\
```

```
    .fig.suptitle("T-SNE with all features ");
```

```
plt.show();
```

(4000, 7358)

(4000,)



Summary

Performed exploratory data analysis on donors choose. Using TSNE with bag of words, tfidf, avg_w2v small clusters are formed but can't conclude from it as approved projects and rejected projects overlap.