chandanachandu124@gmail.com_assignment-4 (1) (1)

June 20, 2019

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

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

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

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

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

        from tqdm import tqdm
        import os

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

        from sklearn.metrics import accuracy_score
```

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

1 Reading the 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  the train data", project_data.shape)
        print('-'*50)
        print("The attributes of data :", project_data.columns.values)
```

```
Number of data points in  the 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]: # how to replace elements in list python: https://stackoverflow.com/a/2582163/4084039
        cols = ['Date' if x=='project_submitted_datetime' else x for x in list(project_data.co


        #sort dataframe based on time pandas python: https://stackoverflow.com/a/49702492/4084
        project_data['Date'] = pd.to_datetime(project_data['project_submitted_datetime'])
        project_data.drop('project_submitted_datetime', axis=1, inplace=True)
        project_data.sort_values(by=['Date'], inplace=True)


        # how to reorder columns pandas python: https://stackoverflow.com/a/13148611/4084039
        project_data = project_data[cols]
        project_data.head(2)
```

```
Out[4]:        Unnamed: 0       id                        teacher_id teacher_prefix  \
        55660        8393  p205479  2bf07ba08945e5d8b2a3f269b2b3cfe5           Mrs.
        76127       37728  p043609  3f60494c61921b3b43ab61bdde2904df            Ms.

              school_state                Date project_grade_category  \
        55660           CA 2016-04-27 00:27:36          Grades PreK-2
```

```
76127          UT 2016-04-27 00:31:25           Grades 3-5

      project_subject_categories           project_subject_subcategories  \
55660             Math & Science  Applied Sciences, Health & Life Science
76127              Special Needs                            Special Needs

                                      project_title  \
55660  Engineering STEAM into the Primary Classroom
76127                      Sensory Tools for Focus

                                         project_essay_1  \
55660  I have been fortunate enough to use the Fairy ...
76127  Imagine being 8-9 years old. You're in your th...

                                         project_essay_2  \
55660  My students come from a variety of backgrounds...
76127  Most of my students have autism, anxiety, anot...

                                         project_essay_3  \
55660  Each month I try to do several science or STEM...
76127  It is tough to do more than one thing at a tim...

                                         project_essay_4  \
55660  It is challenging to develop high quality scie...
76127  When my students are able to calm themselves d...

                                project_resource_summary  \
55660  My students need STEM kits to learn critical s...
76127  My students need Boogie Boards for quiet senso...

       teacher_number_of_previously_posted_projects  project_is_approved
55660                                            53                    1
76127                                             4                    1
```

```
In [5]: print("Number of data points in train data", resource_data.shape)
        print(resource_data.columns.values)
        resource_data.head(2)
```

```
Number of data points in train data (1541272, 4)
['id' 'description' 'quantity' 'price']
```

```
Out[5]:        id                                        description  quantity  \
        0  p233245  LC652 - Lakeshore Double-Space Mobile Drying Rack         1
        1  p069063        Bouncy Bands for Desks (Blue support pipes)         3

            price
        0  149.00
        1   14.95
```

```
In [6]: project_data["teacher_prefix"].fillna(" ", inplace = True)

In [7]: teacher_prefix = []

        for i in range(len(project_data)):
            a = project_data["teacher_prefix"][i].replace('.',' ')
            teacher_prefix.append(a)

In [8]: project_data.drop(['teacher_prefix'], axis=1, inplace=True)

In [9]: project_data["teacher_prefix"] =teacher_prefix

In [10]: project_data.head(5)

Out[10]:        Unnamed: 0       id                        teacher_id school_state  \
         55660        8393  p205479  2bf07ba08945e5d8b2a3f269b2b3cfe5           CA
         76127       37728  p043609  3f60494c61921b3b43ab61bdde2904df           UT
         51140       74477  p189804  4a97f3a390bfe21b99cf5e2b81981c73           CA
         473        100660  p234804  cbc0e38f522143b86d372f8b43d4cff3           GA
         41558       33679  p137682  06f6e62e17de34fcf81020c77549e1d5           WA

                               Date project_grade_category project_subject_categories  \
         55660 2016-04-27 00:27:36          Grades PreK-2             Math & Science
         76127 2016-04-27 00:31:25             Grades 3-5              Special Needs
         51140 2016-04-27 00:46:53          Grades PreK-2        Literacy & Language
         473   2016-04-27 00:53:00          Grades PreK-2           Applied Learning
         41558 2016-04-27 01:05:25             Grades 3-5        Literacy & Language

                         project_subject_subcategories  \
         55660  Applied Sciences, Health & Life Science
         76127                           Special Needs
         51140                                Literacy
         473                         Early Development
         41558                                Literacy

                                           project_title  \
         55660   Engineering STEAM into the Primary Classroom
         76127                        Sensory Tools for Focus
         51140  Mobile Learning with a Mobile Listening Center
         473            Flexible Seating for Flexible Learning
         41558          Going Deep: The Art of Inner Thinking!

                                             project_essay_1  \
         55660  I have been fortunate enough to use the Fairy ...
         76127  Imagine being 8-9 years old. You're in your th...
         51140  Having a class of 24 students comes with diver...
         473    I recently read an article about giving studen...
         41558  My students crave challenge, they eat obstacle...
```

```
                                          project_essay_2  \
55660  My students come from a variety of backgrounds...
76127  Most of my students have autism, anxiety, anot...
51140  I have a class of twenty-four kindergarten stu...
473    I teach at a low-income (Title 1) school. Ever...
41558  We are an urban, public k-5 elementary school...

                                          project_essay_3  \
55660  Each month I try to do several science or STEM...
76127  It is tough to do more than one thing at a tim...
51140  By having a mobile listening and storage cente...
473    We need a classroom rug that we can use as a c...
41558  With the new common core standards that have b...

                                          project_essay_4  \
55660  It is challenging to develop high quality scie...
76127  When my students are able to calm themselves d...
51140  A mobile listening center will help keep equip...
473    Benjamin Franklin once said, \"Tell me and I f...
41558  These remarkable gifts will provide students w...

                                 project_resource_summary  \
55660  My students need STEM kits to learn critical s...
76127  My students need Boogie Boards for quiet senso...
51140  My students need a mobile listening center to ...
473    My students need flexible seating in the class...
41558  My students need copies of the New York Times ...

       teacher_number_of_previously_posted_projects  project_is_approved  \
55660                                            53                    1
76127                                             4                    1
51140                                            10                    1
473                                               2                    1
41558                                             2                    1

      teacher_prefix
55660           Mrs
76127            Mr
51140            Ms
473             Mrs
41558           Mrs
```

```
In [11]: project_grade_category = []

         for i in range(len(project_data)):
             a = project_data["project_grade_category"][i].replace('-','_').replace(' ','_')
             project_grade_category.append(a)
```

```
In [12]: project_data.drop(['project_grade_category'], axis=1, inplace=True)
In [13]: project_data["project_grade_category"] = project_grade_category

In [14]: project_data.head(5)

Out[14]:        Unnamed: 0       id                        teacher_id school_state  \
       55660        8393  p205479  2bf07ba08945e5d8b2a3f269b2b3cfe5           CA
       76127       37728  p043609  3f60494c61921b3b43ab61bdde2904df           UT
       51140       74477  p189804  4a97f3a390bfe21b99cf5e2b81981c73           CA
       473        100660  p234804  cbc0e38f522143b86d372f8b43d4cff3           GA
       41558       33679  p137682  06f6e62e17de34fcf81020c77549e1d5           WA

                            Date project_subject_categories  \
       55660 2016-04-27 00:27:36              Math & Science
       76127 2016-04-27 00:31:25               Special Needs
       51140 2016-04-27 00:46:53         Literacy & Language
       473   2016-04-27 00:53:00            Applied Learning
       41558 2016-04-27 01:05:25         Literacy & Language

                        project_subject_subcategories  \
       55660  Applied Sciences, Health & Life Science
       76127                            Special Needs
       51140                                 Literacy
       473                          Early Development
       41558                                 Literacy

                                              project_title  \
       55660    Engineering STEAM into the Primary Classroom
       76127                           Sensory Tools for Focus
       51140  Mobile Learning with a Mobile Listening Center
       473            Flexible Seating for Flexible Learning
       41558          Going Deep: The Art of Inner Thinking!

                                                 project_essay_1  \
       55660  I have been fortunate enough to use the Fairy ...
       76127  Imagine being 8-9 years old. You're in your th...
       51140  Having a class of 24 students comes with diver...
       473    I recently read an article about giving studen...
       41558  My students crave challenge, they eat obstacle...

                                                 project_essay_2  \
       55660  My students come from a variety of backgrounds...
       76127  Most of my students have autism, anxiety, anot...
       51140  I have a class of twenty-four kindergarten stu...
       473    I teach at a low-income (Title 1) school. Ever...
       41558  We are an urban, public k-5 elementary school...

                                                 project_essay_3  \
       55660  Each month I try to do several science or STEM...
```

```
76127  It is tough to do more than one thing at a tim...
51140  By having a mobile listening and storage cente...
473    We need a classroom rug that we can use as a c...
41558  With the new common core standards that have b...

                                          project_essay_4  \
55660  It is challenging to develop high quality scie...
76127  When my students are able to calm themselves d...
51140  A mobile listening center will help keep equip...
473    Benjamin Franklin once said, \"Tell me and I f...
41558  These remarkable gifts will provide students w...

                                 project_resource_summary  \
55660  My students need STEM kits to learn critical s...
76127  My students need Boogie Boards for quiet senso...
51140  My students need a mobile listening center to ...
473    My students need flexible seating in the class...
41558  My students need copies of the New York Times ...

       teacher_number_of_previously_posted_projects  project_is_approved  \
55660                                            53                    1
76127                                             4                    1
51140                                            10                    1
473                                               2                    1
41558                                             2                    1

      teacher_prefix project_grade_category
55660           Mrs          Grades_PreK_2
76127            Mr            Grades_6_8
51140            Ms            Grades_6_8
473             Mrs          Grades_PreK_2
41558           Mrs          Grades_PreK_2
```

2 Pre-processing the project subject categories

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

         # https://www.geeksforgeeks.org/removing-stop-words-nltk-python/
         # https://stackoverflow.com/questions/23669024/how-to-strip-a-specific-word-from-a-st
         # https://stackoverflow.com/questions/8270092/remove-all-whitespace-in-a-string-in-py
         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", "Warm
                 if 'The' in j.split(): # this will split each of the catogory based on space
```

```
                j=j.replace('The','') # if we have the words "The" we are going to replac
            j = j.replace(' ','_') # we are placeing all the ' '(space) with ''(empty) ex
            temp+=j.strip()+" " #" abc ".strip() will return "abc", remove the trailing s
            temp = temp.replace('&','_') # we are replacing the & value into
        cat_list.append(temp.strip())

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

```
In [16]: from collections import Counter
         my_counter = Counter()
         for word in project_data['clean_categories'].values:
             my_counter.update(word.split())

         cat_dict = dict(my_counter)
         sorted_cat_dict = dict(sorted(cat_dict.items(), key=lambda kv: kv[1]))
```

3 pre-processing of project subject subcategories

```
In [17]: sub_categories = list(project_data['project_subject_subcategories'].values)
         # remove special characters from list of strings python: https://stackoverflow.com/a/

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

         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", "Warm
                 if 'The' in j.split(): # this will split each of the catogory based on space
                     j=j.replace('The','') # if we have the words "The" we are going to replac
                 j = j.replace(' ','') # we are placeing all the ' '(space) with ''(empty) ex:
                 temp +=j.strip()+" "#" abc ".strip() will return "abc", remove the trailing s
                 temp = temp.replace('&','_')
             sub_cat_list.append(temp.strip())

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

```
In [18]: # count of all the words in corpus python: https://stackoverflow.com/a/22898595/40840
         my_counter = Counter()
         for word in project_data['clean_subcategories'].values:
             my_counter.update(word.split())

         sub_cat_dict = dict(my_counter)
         sorted_sub_cat_dict = dict(sorted(sub_cat_dict.items(), key=lambda kv: kv[1]))
```

number of words in title

```
In [19]: title_word_count = []

         for a in project_data["project_title"] :
             b = len(a.split())
             title_word_count.append(b)

         project_data["title_word_count"] = title_word_count
```

```
In [20]: project_data.head(5)
```

```
Out[20]:        Unnamed: 0       id                        teacher_id school_state  \
         55660        8393  p205479  2bf07ba08945e5d8b2a3f269b2b3cfe5           CA
         76127       37728  p043609  3f60494c61921b3b43ab61bdde2904df           UT
         51140       74477  p189804  4a97f3a390bfe21b99cf5e2b81981c73           CA
         473        100660  p234804  cbc0e38f522143b86d372f8b43d4cff3           GA
         41558       33679  p137682  06f6e62e17de34fcf81020c77549e1d5           WA

                               Date                                 project_title  \
         55660 2016-04-27 00:27:36    Engineering STEAM into the Primary Classroom
         76127 2016-04-27 00:31:25                         Sensory Tools for Focus
         51140 2016-04-27 00:46:53  Mobile Learning with a Mobile Listening Center
         473   2016-04-27 00:53:00          Flexible Seating for Flexible Learning
         41558 2016-04-27 01:05:25          Going Deep: The Art of Inner Thinking!

                                                  project_essay_1  \
         55660  I have been fortunate enough to use the Fairy ...
         76127  Imagine being 8-9 years old. You're in your th...
         51140  Having a class of 24 students comes with diver...
         473    I recently read an article about giving studen...
         41558  My students crave challenge, they eat obstacle...

                                                  project_essay_2  \
         55660  My students come from a variety of backgrounds...
         76127  Most of my students have autism, anxiety, anot...
         51140  I have a class of twenty-four kindergarten stu...
         473    I teach at a low-income (Title 1) school. Ever...
         41558  We are an urban, public k-5 elementary school...

                                                  project_essay_3  \
         55660  Each month I try to do several science or STEM...
         76127  It is tough to do more than one thing at a tim...
         51140  By having a mobile listening and storage cente...
         473    We need a classroom rug that we can use as a c...
         41558  With the new common core standards that have b...

                                                  project_essay_4  \
         55660  It is challenging to develop high quality scie...
         76127  When my students are able to calm themselves d...
         51140  A mobile listening center will help keep equip...
         473    Benjamin Franklin once said, \"Tell me and I f...
         41558  These remarkable gifts will provide students w...

                                         project_resource_summary  \
         55660  My students need STEM kits to learn critical s...
         76127  My students need Boogie Boards for quiet senso...
         51140  My students need a mobile listening center to ...
         473    My students need flexible seating in the class...
         41558  My students need copies of the New York Times ...

                teacher_number_of_previously_posted_projects  project_is_approved  \
         55660                                            53                    1
         76127                                             4                    1
         51140                                            10                    1
         473                                               2                    1
         41558                                             2                    1

               teacher_prefix project_grade_category     clean_categories  \
         55660           Mrs          Grades_PreK_2        Math___Science
         76127            Mr             Grades_6_8         Special_Needs
         51140            Ms             Grades_6_8    Literacy___Language
         473             Mrs          Grades_PreK_2      Applied_Learning
         41558           Mrs          Grades_PreK_2    Literacy___Language

                               clean_subcategories  title_word_count
         55660  AppliedSciences Health_LifeScience                    6
         76127                           SpecialNeeds                 4
         51140                               Literacy                 7
         473                         EarlyDevelopment                 5
         41558                               Literacy                 7
```

4 combining all the 4 project essays into 1 essay

```
In [21]: # 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)
```

number of words in essay

```
In [22]: essay_word_count = []

         for ess in project_data["essay"] :
             c = len(ess.split())
```

```
    essay_word_count.append(c)

project_data["essay_word_count"] = essay_word_count
```

```
In [23]: project_data.head(5)
```

```
Out[23]:        Unnamed: 0       id                        teacher_id school_state  \
         55660        8393  p205479  2bf07ba08945e5d8b2a3f269b2b3cfe5           CA
         76127       37728  p043609  3f60494c61921b3b43ab61bdde2904df           UT
         51140       74477  p189804  4a97f3a390bfe21b99cf5e2b81981c73           CA
         473        100660  p234804  cbc0e38f522143b86d372f8b43d4cff3           GA
         41558       33679  p137682  06f6e62e17de34fcf81020c77549e1d5           WA

                               Date                                project_title  \
         55660 2016-04-27 00:27:36    Engineering STEAM into the Primary Classroom
         76127 2016-04-27 00:31:25                         Sensory Tools for Focus
         51140 2016-04-27 00:46:53  Mobile Learning with a Mobile Listening Center
         473   2016-04-27 00:53:00          Flexible Seating for Flexible Learning
         41558 2016-04-27 01:05:25          Going Deep: The Art of Inner Thinking!

                                                  project_essay_1  \
         55660  I have been fortunate enough to use the Fairy ...
         76127  Imagine being 8-9 years old. You're in your th...
         51140  Having a class of 24 students comes with diver...
         473    I recently read an article about giving studen...
         41558  My students crave challenge, they eat obstacle...

                                                  project_essay_2  \
         55660  My students come from a variety of backgrounds...
         76127  Most of my students have autism, anxiety, anot...
         51140  I have a class of twenty-four kindergarten stu...
         473    I teach at a low-income (Title 1) school. Ever...
         41558  We are an urban, public k-5 elementary school...

                                                  project_essay_3  \
         55660  Each month I try to do several science or STEM...
         76127  It is tough to do more than one thing at a tim...
         51140  By having a mobile listening and storage cente...
         473    We need a classroom rug that we can use as a c...
         41558  With the new common core standards that have b...

                                                  project_essay_4  \
         55660  It is challenging to develop high quality scie...
         76127  When my students are able to calm themselves d...
         51140  A mobile listening center will help keep equip...
         473    Benjamin Franklin once said, \"Tell me and I f...
         41558  These remarkable gifts will provide students w...
```

```
                                 project_resource_summary  \
55660  My students need STEM kits to learn critical s...
76127  My students need Boogie Boards for quiet senso...
51140  My students need a mobile listening center to ...
473    My students need flexible seating in the class...
41558  My students need copies of the New York Times ...

       teacher_number_of_previously_posted_projects  project_is_approved  \
55660                                            53                    1
76127                                             4                    1
51140                                            10                    1
473                                               2                    1
41558                                             2                    1

      teacher_prefix project_grade_category     clean_categories  \
55660            Mrs          Grades_PreK_2       Math___Science
76127             Mr             Grades_6_8        Special_Needs
51140             Ms             Grades_6_8  Literacy___Language
473              Mrs          Grades_PreK_2     Applied_Learning
41558            Mrs          Grades_PreK_2  Literacy___Language

                    clean_subcategories  title_word_count  \
55660  AppliedSciences Health_LifeScience                 6
76127                        SpecialNeeds                 4
51140                            Literacy                 7
473                      EarlyDevelopment                 5
41558                            Literacy                 7

                                                   essay  essay_word_count
55660  I have been fortunate enough to use the Fairy ...               285
76127  Imagine being 8-9 years old. You're in your th...               345
51140  Having a class of 24 students comes with diver...               177
473    I recently read an article about giving studen...               225
41558  My students crave challenge, they eat obstacle...               184
```

5 splitting of train and test data

```
In [24]: # train test split

         from sklearn.model_selection import train_test_split

         X_train, X_test, y_train, y_test = train_test_split(project_data, project_data['proje
         X_train, X_cv, y_train, y_cv = train_test_split(X_train, y_train, test_size=0.33, stra
```

```
In [25]: X_train.drop(['project_is_approved'], axis=1, inplace=True)
         X_test.drop(['project_is_approved'], axis=1, inplace=True)
         X_cv.drop(['project_is_approved'], axis=1, inplace=True)
```

6 pre-processing of text

```
In [26]: # printing some random reviews

         print(X_train['essay'].values[0])
         print("="*50)
         print(X_train['essay'].values[505])
         print("="*50)
         print(X_train['essay'].values[1010])
         print("="*50)
         print(X_train['essay'].values[10101])
         print("="*50)
         print(X_train['essay'].values[20000])
         print("="*50)
```

```
My are energetic, enthusiastic, caring , hard-working, giving and loving students are very dive
==================================================
Our soccer program is fast-growing and in its first year of offering soccer as a class. Up to 
==================================================
This group of students is the reason I love to teach! Students are students of course: youthfu
==================================================
My Students are enthusiastic, collaborative and caring! I currently have 15 students in my clas
==================================================
My students come from low-income families with high expectations for their child's education. 
==================================================
```

```
In [27]: # 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 [28]: sent = decontracted(project_data['essay'].values[20101])
         print(sent)
         print("="*50)
```

```
The Robotics program at Mannion MS will be a part of the Explorations Elective. As an explorat
==================================================
```

```
In [29]: # \r \n \t remove from string python: http://texthandler.com/info/remove-line-breaks-
         sent = sent.replace('\\r', ' ')
         sent = sent.replace('\\"', ' ')
         sent = sent.replace('\\n', ' ')
         print(sent)
```

```
The Robotics program at Mannion MS will be a part of the Explorations Elective. As an explorat
```

```
In [30]: #removing special charecters
         sent = re.sub('[^A-Za-z0-9]+', ' ', sent)
         print(sent)
```

```
The Robotics program at Mannion MS will be a part of the Explorations Elective As an exploratio
```

```
In [31]: # 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'
                     "you'll", "you'd", 'your', 'yours', 'yourself', 'yourselves', 'he', 'him'
                     'she', "she's", 'her', 'hers', 'herself', 'it', "it's", 'its', 'itself',
                     'theirs', 'themselves', 'what', 'which', 'who', 'whom', 'this', 'that', "
                     'am', 'is', 'are', 'was', 'were', 'be', 'been', 'being', 'have', 'has', '
                     'did', 'doing', 'a', 'an', 'the', 'and', 'but', 'if', 'or', 'because', 'a
                     'at', 'by', 'for', 'with', 'about', 'against', 'between', 'into', 'throug
                     'above', 'below', 'to', 'from', 'up', 'down', 'in', 'out', 'on', 'off', '
                     'then', 'once', 'here', 'there', 'when', 'where', 'why', 'how', 'all', 'a
                     'most', 'other', 'some', 'such', 'only', 'own', 'same', 'so', 'than', 'to
                     's', 't', 'can', 'will', 'just', 'don', "don't", 'should', "should've", '
                     've', 'y', 'ain', 'aren', "aren't", 'couldn', "couldn't", 'didn', "didn't
                     "hadn't", 'hasn', "hasn't", 'haven', "haven't", 'isn', "isn't", 'ma', 'mi
                     "mustn't", 'needn', "needn't", 'shan', "shan't", 'shouldn', "shouldn't",
                     'won', "won't", 'wouldn', "wouldn't"]
```

7 pre-processed train data

```
In [32]: # Combining all the above

         from tqdm import tqdm
         preprocessed_essays_train = []
         # tqdm is for printing the status bar
         for sentence in tqdm(X_train['essay'].values):
             sent = decontracted(sentence)
             sent = sent.replace('\\r', ' ')
```

```
        sent = sent.replace('\\"', ' ')
        sent = sent.replace('\\n', ' ')
        sent = re.sub('[^A-Za-z0-9]+', ' ', sent)
        # https://gist.github.com/sebleier/554280
        sent = ' '.join(e for e in sent.split() if e.lower() not in stopwords)
        preprocessed_essays_train.append(sent.lower().strip())
```

```
100%|| 49041/49041 [00:32<00:00, 1528.49it/s]
```

```
In [33]: preprocessed_essays_train[1010]
```

```
Out[33]: 'group students reason love teach students students course youthful bright eyes full 
```

8 pre-processed test data

```
In [34]: preprocessed_essays_test = []
    # tqdm is for printing the status bar
    for sentence in tqdm(X_test['essay'].values):
        sent = decontracted(sentence)
        sent = sent.replace('\\r', ' ')
        sent = sent.replace('\\"', ' ')
        sent = sent.replace('\\n', ' ')
        sent = re.sub('[^A-Za-z0-9]+', ' ', sent)
        # https://gist.github.com/sebleier/554280
        sent = ' '.join(e for e in sent.split() if e.lower() not in stopwords)
        preprocessed_essays_test.append(sent.lower().strip())
```

```
100%|| 36052/36052 [00:23<00:00, 1503.79it/s]
```

```
In [35]: preprocessed_essays_test[1010]
```

```
Out[35]: 'technology specialist works high need community school 650 students 68 percent studer
```

9 pre-processed cross validation data

```
In [36]: preprocessed_essays_cv = []
    # tqdm is for printing the status bar
    for sentence in tqdm(X_cv['essay'].values):
        sent = decontracted(sentence)
        sent = sent.replace('\\r', ' ')
        sent = sent.replace('\\"', ' ')
        sent = sent.replace('\\n', ' ')
        sent = re.sub('[^A-Za-z0-9]+', ' ', sent)
        # https://gist.github.com/sebleier/554280
        sent = ' '.join(e for e in sent.split() if e.lower() not in stopwords)
        preprocessed_essays_cv.append(sent.lower().strip())
```

```
100%|| 24155/24155 [00:16<00:00, 1435.34it/s]
```

```
In [37]: preprocessed_essays_cv[1010]
```

```
Out[37]: 'currently 25 six seven year old students classroom teaching first graders title scho
```

10 pre-processing project titles

```
In [38]: # printing some random essays.
         print(project_data['project_title'].values[0])
         print("="*50)
         print(project_data['project_title'].values[150])
         print("="*50)
         print(project_data['project_title'].values[1010])
         print("="*50)
         print(project_data['project_title'].values[20101])
         print("="*50)
```

```
Engineering STEAM into the Primary Classroom
==================================================
Building Blocks for Learning
==================================================
Extra, Extra, Read All About It!
==================================================
Lego Mindstorm Activate!
==================================================
```

11 pre-processing project title for train data

```
In [39]: preprocessed_titles_train = []

         for titles in tqdm(X_train["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_train.append(title.lower().strip())
```

```
100%|| 49041/49041 [00:01<00:00, 31648.10it/s]
```

```
In [40]: preprocessed_titles_train[1010]
```

```
Out[40]: 'connecting world around us'
```

12 pre-processing project title for test data

```
In [41]: preprocessed_titles_test = []

         for titles in tqdm(X_test["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_test.append(title.lower().strip())
```

```
100%|| 36052/36052 [00:01<00:00, 31792.77it/s]
```

```
In [42]: preprocessed_titles_test[1010]
```

```
Out[42]: 'the crestwood dash success'
```

13 pre-processing of project title for cross validation data

```
In [43]: preprocessed_titles_cv = []

         for titles in tqdm(X_cv["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_cv.append(title.lower().strip())
```

```
100%|| 24155/24155 [00:00<00:00, 30931.79it/s]
```

```
In [44]: preprocessed_titles_cv[1010]
```

```
Out[44]: 'books rescue'
```

14 preparing data for models

```
In [45]: project_data.columns
```

```
Out[45]: Index(['Unnamed: 0', 'id', 'teacher_id', 'school_state', 'Date',
                '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',
                'teacher_prefix', 'project_grade_category', 'clean_categories',
                'clean_subcategories', 'title_word_count', 'essay', 'essay_word_count'],
               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

15 vectorizing categorical data

one hot encode clean categories of projects

```
In [46]: # we use count vectorizer to convert the values into one

         from sklearn.feature_extraction.text import CountVectorizer

         vectorizer_proj = CountVectorizer(vocabulary=list(sorted_cat_dict.keys()), lowercase=
         vectorizer_proj.fit(X_train['clean_categories'].values)

         categories_one_hot_train = vectorizer_proj.transform(X_train['clean_categories'].valu
         categories_one_hot_test = vectorizer_proj.transform(X_test['clean_categories'].values
         categories_one_hot_cv = vectorizer_proj.transform(X_cv['clean_categories'].values)

         print(vectorizer_proj.get_feature_names())

         print("Shape of matrix of Train data after one hot encoding ",categories_one_hot_trai
         print("Shape of matrix of Test data after one hot encoding ",categories_one_hot_test.
         print("Shape of matrix of CV data after one hot encoding ",categories_one_hot_cv.shap
['_Warmth', '_Health___Sports', 'Warmth', '_Care___Hunger', '_History___Civics', '_Applied_Lea
Shape of matrix of Train data after one hot encoding  (49041, 17)
Shape of matrix of Test data after one hot encoding  (36052, 17)
Shape of matrix of CV data after one hot encoding  (24155, 17)
```

one hot encode clean sub categories of projects

```
In [47]: vectorizer_sub_proj = CountVectorizer(vocabulary=list(sorted_sub_cat_dict.keys()), low
         vectorizer_sub_proj.fit(X_train['clean_subcategories'].values)

         sub_categories_one_hot_train = vectorizer_sub_proj.transform(X_train['clean_subcatego
         sub_categories_one_hot_test = vectorizer_sub_proj.transform(X_test['clean_subcategorie
         sub_categories_one_hot_cv = vectorizer_sub_proj.transform(X_cv['clean_subcategories']


         print(vectorizer_sub_proj.get_feature_names())

         print("Shape of matrix of Train data after one hot encoding ",sub_categories_one_hot_
         print("Shape of matrix of Test data after one hot encoding ",sub_categories_one_hot_te
         print("Shape of matrix of Cross Validation data after one hot encoding ",sub_categorie
```

```
['Economics', 'CommunityService', 'FinancialLiteracy', 'ParentInvolvement', 'Extracurricular',
Shape of matrix of Train data after one hot encoding  (49041, 30)
Shape of matrix of Test data after one hot encoding  (36052, 30)
Shape of matrix of Cross Validation data after one hot encoding  (24155, 30)
```

one hot encode for school states

```
In [48]: my_counter = Counter()
         for state in project_data['school_state'].values:
             my_counter.update(state.split())

In [49]: school_state_cat_dict = dict(my_counter)
         sorted_school_state_cat_dict = dict(sorted(school_state_cat_dict.items(), key=lambda k

In [50]: ## we use count vectorizer to convert the values into one hot encoded features

         vectorizer_states = CountVectorizer(vocabulary=list(sorted_school_state_cat_dict.keys
         vectorizer_states.fit(X_train['school_state'].values)

         school_state_categories_one_hot_train = vectorizer_states.transform(X_train['school_st
         school_state_categories_one_hot_test = vectorizer_states.transform(X_test['school_sta
         school_state_categories_one_hot_cv = vectorizer_states.transform(X_cv['school_state']

         print(vectorizer_states.get_feature_names())

         print("Shape of matrix of Train data after one hot encoding ",school_state_categories_
         print("Shape of matrix of Test data after one hot encoding ",school_state_categories_o
         print("Shape of matrix of Cross Validation data after one hot encoding ",school_state_
['VT', 'WY', 'ND', 'MT', 'RI', 'SD', 'NE', 'DE', 'AK', 'NH', 'WV', 'ME', 'HI', 'DC', 'NM', 'KS
Shape of matrix of Train data after one hot encoding  (49041, 51)
Shape of matrix of Test data after one hot encoding  (36052, 51)
Shape of matrix of Cross Validation data after one hot encoding  (24155, 51)
```

one hot encode project grade category

```
In [51]: my_counter = Counter()
         for project_grade in project_data['project_grade_category'].values:
             my_counter.update(project_grade.split())

In [52]: project_grade_cat_dict = dict(my_counter)
         sorted_project_grade_cat_dict = dict(sorted(project_grade_cat_dict.items(), key=lambda

In [53]: vectorizer_grade = CountVectorizer(vocabulary=list(sorted_project_grade_cat_dict.keys
         vectorizer_grade.fit(X_train['project_grade_category'].values)

         project_grade_categories_one_hot_train = vectorizer_grade.transform(X_train['project_g
         project_grade_categories_one_hot_test = vectorizer_grade.transform(X_test['project_gra
         project_grade_categories_one_hot_cv = vectorizer_grade.transform(X_cv['project_grade_c

         print(vectorizer_grade.get_feature_names())

         print("Shape of matrix of Train data after one hot encoding ",project_grade_categories
         print("Shape of matrix of Test data after one hot encoding ",project_grade_categories_
         print("Shape of matrix of Cross Validation data after one hot encoding ",project_grade
```

```
['Grades_9_12', 'Grades_6_8', 'Grades_3_5', 'Grades_PreK_2']
Shape of matrix of Train data after one hot encoding  (49041, 4)
Shape of matrix of Test data after one hot encoding  (36052, 4)
Shape of matrix of Cross Validation data after one hot encoding  (24155, 4)
```

one hot encode for teacher prefix

```
In [54]: my_counter = Counter()
         for teacher_prefix in project_data['teacher_prefix'].values:
             teacher_prefix = str(teacher_prefix)
             my_counter.update(teacher_prefix.split())

In [55]: teacher_prefix_cat_dict = dict(my_counter)
         sorted_teacher_prefix_cat_dict = dict(sorted(teacher_prefix_cat_dict.items(), key=lamb

In [56]: ## we use count vectorizer to convert the values into one hot encoded features
         ## Unlike the previous Categories this category returns a
         ## ValueError: np.nan is an invalid document, expected byte or unicode string.
         ## The link below explains hOw to tackle such discrepancies.
         ## https://stackoverflow.com/questions/39303912/tfidfvectorizer-in-scikit-learn-value
```

```
        vectorizer_teacher = CountVectorizer(vocabulary=list(sorted_teacher_prefix_cat_dict.ke
        vectorizer_teacher.fit(X_train['teacher_prefix'].values.astype("U"))

        teacher_prefix_categories_one_hot_train = vectorizer_teacher.transform(X_train['teache
        teacher_prefix_categories_one_hot_test = vectorizer_teacher.transform(X_test['teacher_
        teacher_prefix_categories_one_hot_cv = vectorizer_teacher.transform(X_cv['teacher_pre

        print(vectorizer_teacher.get_feature_names())

        print("Shape of matrix after one hot encoding ",teacher_prefix_categories_one_hot_trai
        print("Shape of matrix after one hot encoding ",teacher_prefix_categories_one_hot_tes
        print("Shape of matrix after one hot encoding ",teacher_prefix_categories_one_hot_cv.s
```

```
['Dr', 'Teacher', 'Mr', 'Ms', 'Mrs']
Shape of matrix after one hot encoding  (49041, 5)
Shape of matrix after one hot encoding  (36052, 5)
Shape of matrix after one hot encoding  (24155, 5)
```

vectorizing text data

16 Bag of words for train data- essays

```
In [57]: # We are considering only the words which appeared in at least 10 documents(rows or p

         vectorizer_bow_essay = CountVectorizer(min_df=10)

         vectorizer_bow_essay.fit(preprocessed_essays_train)

         text_bow_train = vectorizer_bow_essay.transform(preprocessed_essays_train)

         print("Shape of matrix after one hot encoding ",text_bow_train.shape)
```

```
Shape of matrix after one hot encoding  (49041, 11995)
```

17 Bag of words for test data- essays

```
In [58]: text_bow_test = vectorizer_bow_essay.transform(preprocessed_essays_test)
         print("Shape of matrix after one hot encoding ",text_bow_test.shape)
```

```
Shape of matrix after one hot encoding  (36052, 11995)
```

BOW for cross validation

```
In [59]: text_bow_cv = vectorizer_bow_essay.transform(preprocessed_essays_cv)
         print("Shape of matrix after one hot encoding ",text_bow_cv.shape)
```

```
Shape of matrix after one hot encoding  (24155, 11995)
```

bow for train data- titles

```
In [60]: vectorizer_bow_title = CountVectorizer(min_df=10)

         vectorizer_bow_title.fit(preprocessed_titles_train)

         title_bow_train = vectorizer_bow_title.transform(preprocessed_titles_train)
         print("Shape of matrix after one hot encoding ",title_bow_train.shape)
```

```
Shape of matrix after one hot encoding  (49041, 2113)
```

bow for test data- titles

```
In [61]: title_bow_test = vectorizer_bow_title.transform(preprocessed_titles_test)
         print("Shape of matrix after one hot encoding ",title_bow_test.shape)
```

```
Shape of matrix after one hot encoding  (36052, 2113)
```

bow cross-validation data for titles

```
In [62]: title_bow_cv = vectorizer_bow_title.transform(preprocessed_titles_cv)
         print("Shape of matrix after one hot encoding ",title_bow_cv.shape)
```

```
Shape of matrix after one hot encoding  (24155, 2113)
```

18 tfidf vectorizer

Tfidf train data- essays

```
In [63]: from sklearn.feature_extraction.text import TfidfVectorizer

         vectorizer_tfidf_essay = TfidfVectorizer(min_df=10)
         vectorizer_tfidf_essay.fit(preprocessed_essays_train)

         text_tfidf_train = vectorizer_tfidf_essay.transform(preprocessed_essays_train)
         print("Shape of matrix after one hot encoding ",text_tfidf_train.shape)
```

```
Shape of matrix after one hot encoding  (49041, 11995)
```

Tfidf for test data- essays

```
In [64]: text_tfidf_test = vectorizer_tfidf_essay.transform(preprocessed_essays_test)
         print("Shape of matrix after one hot encoding ",text_tfidf_test.shape)
```

```
Shape of matrix after one hot encoding  (36052, 11995)
```

Tfidf for cross validation data- essays

```
In [65]: text_tfidf_cv = vectorizer_tfidf_essay.transform(preprocessed_essays_cv)
         print("Shape of matrix after one hot encoding ",text_tfidf_cv.shape)
```

```
Shape of matrix after one hot encoding  (24155, 11995)
```

Tfidf train data- titles

```
In [66]: vectorizer_tfidf_titles = TfidfVectorizer(min_df=10)

         vectorizer_tfidf_titles.fit(preprocessed_titles_train)
         title_tfidf_train = vectorizer_tfidf_titles.transform(preprocessed_titles_train)
         print("Shape of matrix after one hot encoding ",title_tfidf_train.shape)
```

```
Shape of matrix after one hot encoding  (49041, 2113)
```

tfidf for test data -titles

```
In [67]: title_tfidf_test = vectorizer_tfidf_titles.transform(preprocessed_titles_test)
         print("Shape of matrix after one hot encoding ",title_tfidf_test.shape)
```

```
Shape of matrix after one hot encoding  (36052, 2113)
```

tfidf for cross validation data - titles

```
In [68]: title_tfidf_cv = vectorizer_tfidf_titles.transform(preprocessed_titles_cv)
         print("Shape of matrix after one hot encoding ",title_tfidf_cv.shape)
```

```
Shape of matrix after one hot encoding  (24155, 2113)
```

19 Vectorizing Numerical features

Price

```
In [69]: # https://stackoverflow.com/questions/22407798/how-to-reset-a-dataframes-indexes-for-
         price_data = resource_data.groupby('id').agg({'price':'sum', 'quantity':'sum'}).reset_
         price_data.head(2)
```

```
Out[69]:         id   price  quantity
         0  p000001  459.56         7
         1  p000002  515.89        21
```

```
In [70]: # join two dataframes in python:
         X_train = pd.merge(X_train, price_data, on='id', how='left')
         X_test = pd.merge(X_test, price_data, on='id', how='left')
         X_cv = pd.merge(X_cv, price_data, on='id', how='left')
```

```
In [71]: from sklearn.preprocessing import Normalizer

         normalizer = Normalizer()

         # normalizer.fit(X_train['price'].values)
         # this will rise an error Expected 2D array, got 1D array instead:
         # array=[105.22 215.96  96.01 ... 368.98  80.53 709.67].
         # Reshape your data either using
         # array.reshape(1,-1) if your data has a single feature
         # array.reshape(1, -1)  if it contains a single sample.

         normalizer.fit(X_train['price'].values.reshape(-1,1))

         price_train = normalizer.transform(X_train['price'].values.reshape(-1,1))
         price_cv = normalizer.transform(X_cv['price'].values.reshape(-1,1))
         price_test = normalizer.transform(X_test['price'].values.reshape(-1,1))

         print("After vectorizations")
         print(price_train.shape, y_train.shape)
         print(price_cv.shape, y_cv.shape)
         print(price_test.shape, y_test.shape)
         print("="*100)
```

```
After vectorizations
(49041, 1) (49041,)
(24155, 1) (24155,)
(36052, 1) (36052,)
==============================================================================================
```

Quantity

```
In [72]: normalizer = Normalizer()

         # normalizer.fit(X_train['price'].values)
         # this will rise an error Expected 2D array, got 1D array instead:
         # array=[105.22 215.96  96.01 ... 368.98  80.53 709.67].
         # Reshape your data either using
         # array.reshape(-1, 1) if your data has a single feature
         # array.reshape(1, -1)  if it contains a single sample.

         normalizer.fit(X_train['quantity'].values.reshape(-1,1))

         quantity_train = normalizer.transform(X_train['quantity'].values.reshape(-1,1))
         quantity_cv = normalizer.transform(X_cv['quantity'].values.reshape(-1,1))
         quantity_test = normalizer.transform(X_test['quantity'].values.reshape(-1,1))

         print("After vectorizations")
         print(quantity_train.shape, y_train.shape)
         print(quantity_cv.shape, y_cv.shape)
         print(quantity_test.shape, y_test.shape)
         print("="*100)
```

```
After vectorizations
(49041, 1) (49041,)
(24155, 1) (24155,)
(36052, 1) (36052,)
===============================================================================================
```

Teacher previously proposed projects

```
In [73]: normalizer = Normalizer()

         # normalizer.fit(X_train['price'].values)
         # this will rise an error Expected 2D array, got 1D array instead:
         # array=[105.22 215.96  96.01 ... 368.98  80.53 709.67].
         # Reshape your data either using
         # array.reshape(-1, 1) if your data has a single feature
         # array.reshape(1, -1)  if it contains a single sample.

         normalizer.fit(X_train['teacher_number_of_previously_posted_projects'].values.reshape

         prev_projects_train = normalizer.transform(X_train['teacher_number_of_previously_poste
         prev_projects_cv = normalizer.transform(X_cv['teacher_number_of_previously_posted_pro
         prev_projects_test = normalizer.transform(X_test['teacher_number_of_previously_posted

         print("After vectorizations")
         print(prev_projects_train.shape, y_train.shape)
         print(prev_projects_cv.shape, y_cv.shape)
         print(prev_projects_test.shape, y_test.shape)
         print("="*100)
```

```
After vectorizations
(49041, 1) (49041,)
(24155, 1) (24155,)
(36052, 1) (36052,)
===============================================================================================
```

Title Word count

```
In [74]: normalizer = Normalizer()
```

```
        normalizer.fit(X_train['title_word_count'].values.reshape(-1,1))

        title_word_count_train = normalizer.transform(X_train['title_word_count'].values.resha
        title_word_count_cv = normalizer.transform(X_cv['title_word_count'].values.reshape(-1
        title_word_count_test = normalizer.transform(X_test['title_word_count'].values.reshape

        print("After vectorizations")
        print(title_word_count_train.shape, y_train.shape)
        print(title_word_count_cv.shape, y_cv.shape)
        print(title_word_count_test.shape, y_test.shape)
        print("="*100)
```

```
After vectorizations
(49041, 1) (49041,)
(24155, 1) (24155,)
(36052, 1) (36052,)
==============================================================================================
```

Essay word count

```
In [75]: normalizer = Normalizer()

         normalizer.fit(X_train['essay_word_count'].values.reshape(-1,1))

         essay_word_count_train = normalizer.transform(X_train['essay_word_count'].values.resha
         essay_word_count_cv = normalizer.transform(X_cv['essay_word_count'].values.reshape(-1
         essay_word_count_test = normalizer.transform(X_test['essay_word_count'].values.reshape

         print("After vectorizations")
         print(essay_word_count_train.shape, y_train.shape)
         print(essay_word_count_cv.shape, y_cv.shape)
         print(essay_word_count_test.shape, y_test.shape)
         print("="*100)
```

```
After vectorizations
(49041, 1) (49041,)
(24155, 1) (24155,)
(36052, 1) (36052,)
==============================================================================================
```

20 Navie Bayes

Set 1: categorical, numerical features + project_title(BOW) + preprocessed_essay (BOW)

```
In [76]: # merge two sparse matrices: https://stackoverflow.com/a/19710648/4084039
         from scipy.sparse import hstack
```

```
        X_tr = hstack((categories_one_hot_train, sub_categories_one_hot_train, school_state_ca
        X_te = hstack((categories_one_hot_test, sub_categories_one_hot_test, school_state_cate
        X_cr = hstack((categories_one_hot_cv, sub_categories_one_hot_cv, school_state_categor
```

```
In [77]: print("Final Data matrix")
         print(X_tr.shape, y_train.shape)
         print(X_cr.shape, y_cv.shape)
         print(X_te.shape, y_test.shape)
         print("="*100)
```

```
Final Data matrix
(49041, 14220) (49041,)
(24155, 14220) (24155,)
(36052, 14220) (36052,)
===============================================================================================
```

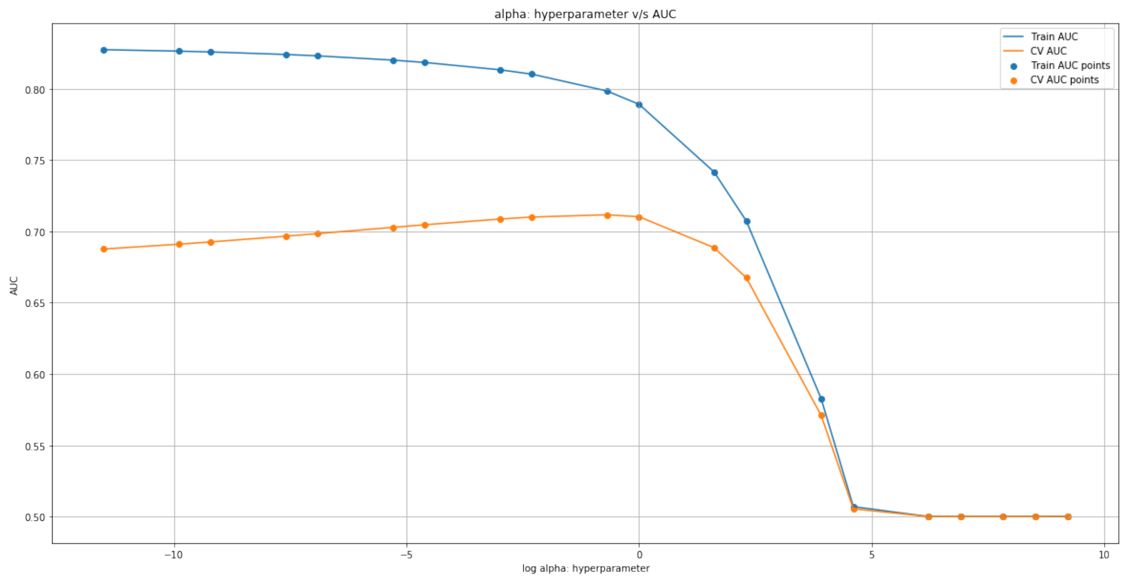
Finding the best hyper parameter which results in the maximum AUC value

```
In [78]: def batch_predict(clf, data):
             # roc_auc_score(y_true, y_score) the 2nd parameter should be probability estimate
             # not the predicted outputs

             y_data_pred = []
             tr_loop = data.shape[0] - data.shape[0]%1000
             # consider you X_tr shape is 49041, then your cr_loop will be 49041 - 49041%1000
             # in this for loop we will iterate unti the last 1000 multiplier
             for i in range(0, tr_loop, 1000):
                 y_data_pred.extend(clf.predict_proba(data[i:i+1000])[:,1])
             # we will be predicting for the last data points
             y_data_pred.extend(clf.predict_proba(data[tr_loop:])[:,1])

             return y_data_pred
```

21 Random Alpha Values

```
In [79]: import matplotlib.pyplot as plt
         from sklearn.naive_bayes import MultinomialNB
         from sklearn.metrics import roc_auc_score
         import math

         train_auc = []
         cv_auc = []
         log_alphas = []

         alphas = [0.00001, 0.00005, 0.0001, 0.0005, 0.001, 0.005, 0.01, 0.05, 0.1, 0.5, 1, 5,
         for i in tqdm(alphas):
             nb = MultinomialNB(alpha = i)
             nb.fit(X_tr, y_train)

             y_train_pred = batch_predict(nb, X_tr)
             y_cv_pred = batch_predict(nb, X_cr)


             # roc_auc_score(y_true, y_score) the 2nd parameter should be probability estimate
             # not the predicted outputs
             train_auc.append(roc_auc_score(y_train,y_train_pred))
             cv_auc.append(roc_auc_score(y_cv, y_cv_pred))

         for a in tqdm(alphas):
             b = math.log(a)
             log_alphas.append(b)

100%|| 20/20 [00:03<00:00,  5.41it/s]
100%|| 20/20 [00:00<00:00, 20097.29it/s]
```

```
In [80]: plt.figure(figsize=(20,10))
         plt.plot(log_alphas, train_auc, label='Train AUC')
         plt.plot(log_alphas, cv_auc, label='CV AUC')

         plt.scatter(log_alphas, train_auc, label='Train AUC points')
         plt.scatter(log_alphas, cv_auc, label='CV AUC points')

         plt.legend()
         plt.xlabel("log alpha: hyperparameter")
         plt.ylabel("AUC")
         plt.title("alpha: hyperparameter v/s AUC")
         plt.grid()
         plt.show()
```



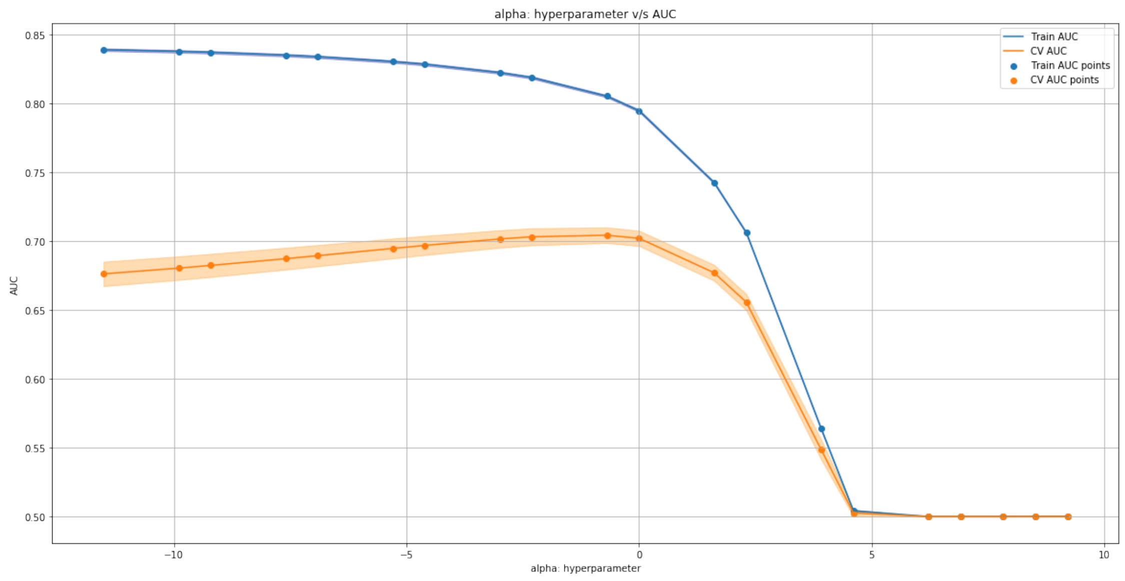
Summary: 1. Values ranging between 10ˆ-4 to 10ˆ4 for alpha parameter are considered. 2. Log of Alphas was plotted on the X axis with the AUC values on the Y axis. 3. We have observed that very low or very high values of Alpha seem to be not effective while developing the required model.

Gridsearch-cv using cv = 10 (K fold cross validation)

```
In [81]: from sklearn.model_selection import GridSearchCV

         nb = MultinomialNB()

         parameters = {'alpha':[0.00001, 0.00005, 0.0001, 0.0005, 0.001, 0.005, 0.01, 0.05, 0.

         clf = GridSearchCV(nb, parameters, cv= 10, scoring='roc_auc')

         clf.fit(X_tr, y_train)

         train_auc= clf.cv_results_['mean_train_score']
         train_auc_std= clf.cv_results_['std_train_score']
         cv_auc = clf.cv_results_['mean_test_score']
         cv_auc_std= clf.cv_results_['std_test_score']

In [82]: lphas = [0.00001, 0.00005, 0.0001, 0.0005, 0.001, 0.005, 0.01, 0.05, 0.1, 0.5, 1, 5, 
         log_alphas =[]

         for a in tqdm(alphas):
             b = math.log(a)
             log_alphas.append(b)

         plt.figure(figsize=(20,10))
```

```
    plt.plot(log_alphas, train_auc, label='Train AUC')
    # this code is copied from here: https://stackoverflow.com/a/48803361/4084039
    plt.gca().fill_between(log_alphas,train_auc - train_auc_std,train_auc + train_auc_std

    plt.plot(log_alphas, cv_auc, label='CV AUC')
    # this code is copied from here: https://stackoverflow.com/a/48803361/4084039
    plt.gca().fill_between(log_alphas,cv_auc - cv_auc_std,cv_auc + cv_auc_std,alpha=0.3,co

    plt.scatter(log_alphas, train_auc, label='Train AUC points')
    plt.scatter(log_alphas, cv_auc, label='CV AUC points')


    plt.legend()
    plt.xlabel("alpha: hyperparameter")
    plt.ylabel("AUC")
    plt.title("alpha: hyperparameter v/s AUC")
    plt.grid()
    plt.show()
```

```
100%|| 20/20 [00:00<?, ?it/s]
```



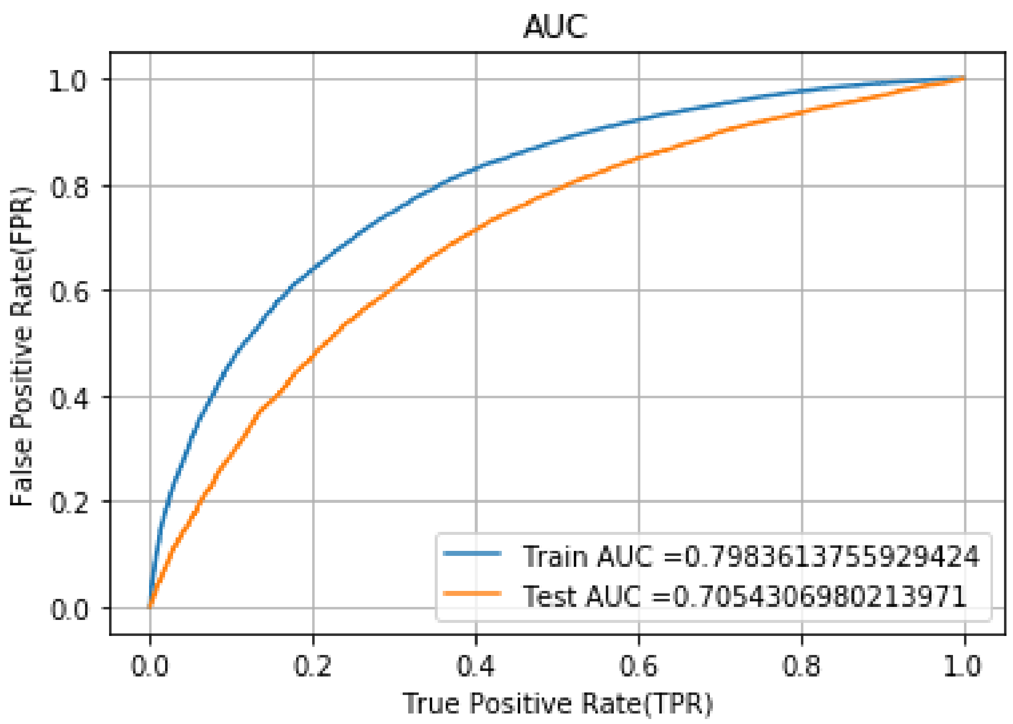
Summary of Alpha values for BOW model : Alpha values ranging from 0.00001 to 10000.0 are taken and the results are as follows : 0.00001 as alpha values seemed to work very well on train data and the model seems to not work that efficiently on cross validation data. Values closer to 1.0 works well both on Train data and Cross Validation data. Values more than 1.0 also doesnt seem to be effective on both Train data and Cross Validation data.

0.5 as alpha value was chosen. Even 1.0 resulted in an almost similar result.

Train model using the best hyper-parameter value

```
In [83]: best_k_1 = 0.5
```

```
In [84]: # https://scikit-learn.org/stable/modules/generated/sklearn.metrics.roc_curve.html#sk
         from sklearn.metrics import roc_curve, auc

         nb_bow = MultinomialNB(alpha = best_k_1)

         nb_bow.fit(X_tr, y_train)
         # roc_auc_score(y_true, y_score) the 2nd parameter should be probability estimates of
         # not the predicted outputs

         y_train_pred = batch_predict(nb_bow, X_tr)
         y_test_pred = batch_predict(nb_bow, X_te)

         train_fpr, train_tpr, tr_thresholds = roc_curve(y_train, y_train_pred)
         test_fpr, test_tpr, te_thresholds = roc_curve(y_test, y_test_pred)

         plt.plot(train_fpr, train_tpr, label="Train AUC ="+str(auc(train_fpr, train_tpr)))
         plt.plot(test_fpr, test_tpr, label="Test AUC ="+str(auc(test_fpr, test_tpr)))
         plt.legend()
         plt.xlabel("True Positive Rate(TPR)")
         plt.ylabel("False Positive Rate(FPR)")
         plt.title("AUC")
         plt.grid()
         plt.show()
```



Confusion Matrix

```
In [85]: def predict(proba, threshould, fpr, tpr):

             t = threshould[np.argmax(fpr*(1-tpr))]

             # (tpr*(1-fpr)) will be maximum if your fpr is very low and tpr is very high

             print("the maximum value of tpr*(1-fpr)", max(tpr*(1-fpr)), "for threshold", np.ro
             predictions = []
             for i in proba:
                 if i>=t:
                     predictions.append(1)
                 else:
                     predictions.append(0)
             return predictions
```

Train data

```
In [86]: print("="*100)
         from sklearn.metrics import confusion_matrix
         print("Train confusion matrix")
         print(confusion_matrix(y_train, predict(y_train_pred, tr_thresholds, train_fpr, train_
```

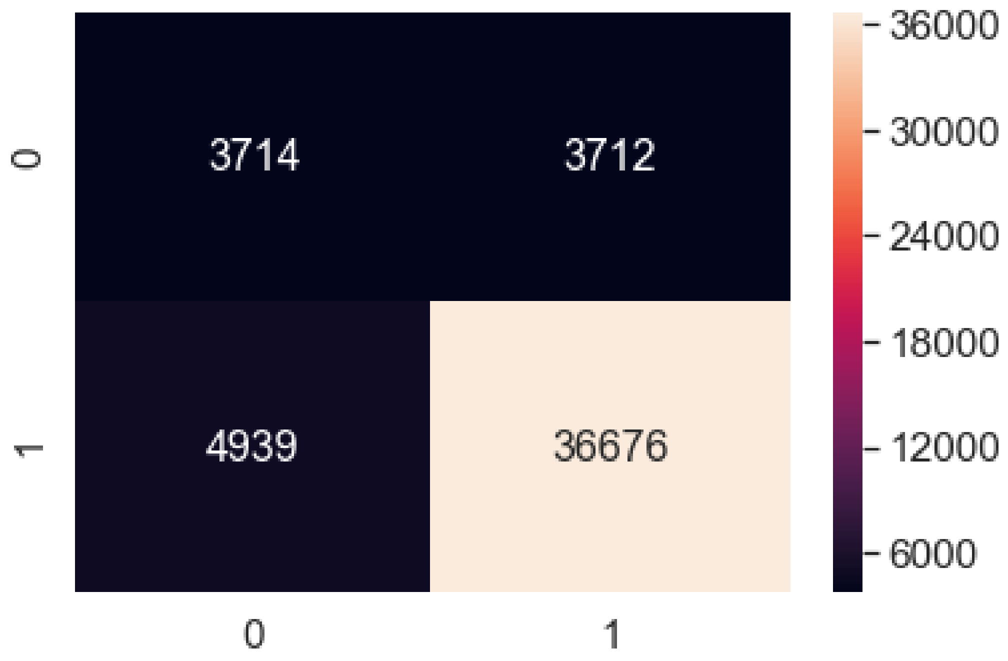
```
=================================================================================================
Train confusion matrix
the maximum value of tpr*(1-fpr) 0.2499999818661462 for threshold 0.129
[[ 3714  3712]
 [ 4939 36676]]
```

```
In [87]: conf_matr_df_train_1 = pd.DataFrame(confusion_matrix(y_train, predict(y_train_pred, tr
         range(2),range(2))
```

```
the maximum value of tpr*(1-fpr) 0.2499999818661462 for threshold 0.129
```

```
In [88]: sns.set(font_scale=1.4)#for label size
         sns.heatmap(conf_matr_df_train_1, annot=True,annot_kws={"size": 16}, fmt='g')
```

```
Out[88]: <matplotlib.axes._subplots.AxesSubplot at 0x123c2984588>
```



Test data

```
In [89]: print("="*100)
         print("Test confusion matrix")
         print(confusion_matrix(y_test, predict(y_test_pred, tr_thresholds, test_fpr, test_fpr
```

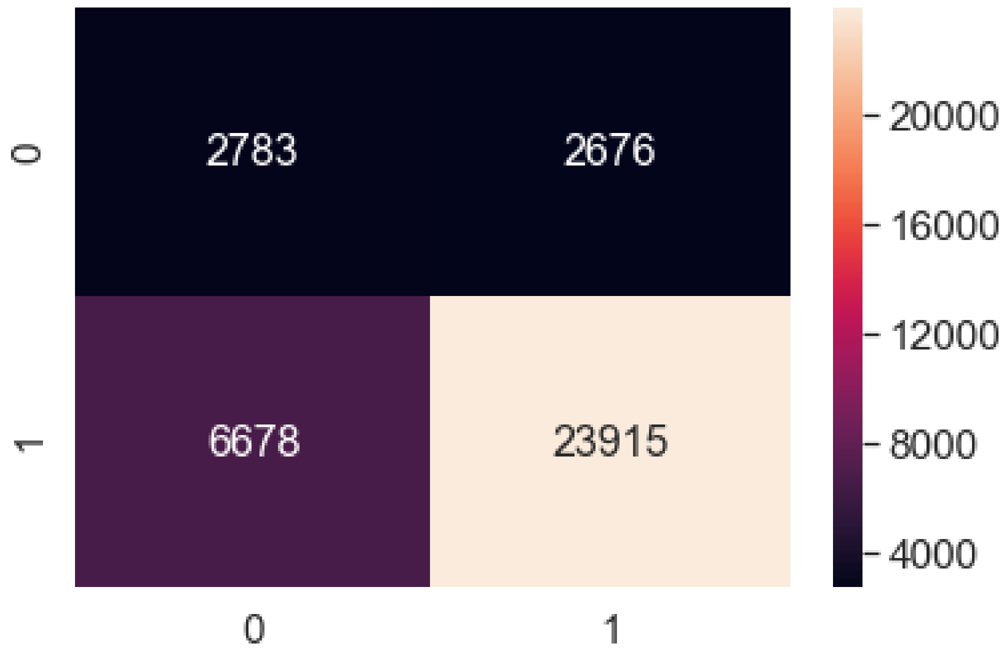
```
===================================================================================================
Test confusion matrix
the maximum value of tpr*(1-fpr) 0.24999999161092998 for threshold 0.558
[[ 2783  2676]
 [ 6678 23915]]
```

```
In [90]: conf_matr_df_test_1 = pd.DataFrame(confusion_matrix(y_test, predict(y_test_pred, tr_t
         range(2),range(2))
```

```
the maximum value of tpr*(1-fpr) 0.24999999161092998 for threshold 0.558
```

```
In [91]: sns.set(font_scale=1.4)#for label size
         sns.heatmap(conf_matr_df_test_1, annot=True,annot_kws={"size": 16}, fmt='g')
```

```
Out[91]: <matplotlib.axes._subplots.AxesSubplot at 0x12380c1c198>
```



22 Set 2 : categorical, numerical features + project_title(TFIDF) + preprocessed_essay (TFIDF)

```
In [92]: # merge two sparse matrices: https://stackoverflow.com/a/19710648/4084039
         from scipy.sparse import hstack

         X_tr = hstack((categories_one_hot_train, sub_categories_one_hot_train, school_state_ca
         project_grade_categories_one_hot_train, teacher_prefix_categories_one_hot_train, pric
         prev_projects_train, title_word_count_train, essay_word_count_train, text_tfidf_train

         X_te = hstack((categories_one_hot_test, sub_categories_one_hot_test, school_state_cat
         project_grade_categories_one_hot_test, teacher_prefix_categories_one_hot_test, price_
         title_word_count_test, essay_word_count_test, text_tfidf_test, title_tfidf_test)).toc

         X_cr = hstack((categories_one_hot_cv, sub_categories_one_hot_cv, school_state_categor
         essay_word_count_cv, text_tfidf_cv, title_tfidf_cv)).tocsr()
```

```
In [93]: print("Final Data matrix")
         print(X_tr.shape, y_train.shape)
         print(X_cr.shape, y_cv.shape)
         print(X_te.shape, y_test.shape)
         print("="*100)
```

```
Final Data matrix
(49041, 14220) (49041,)
```

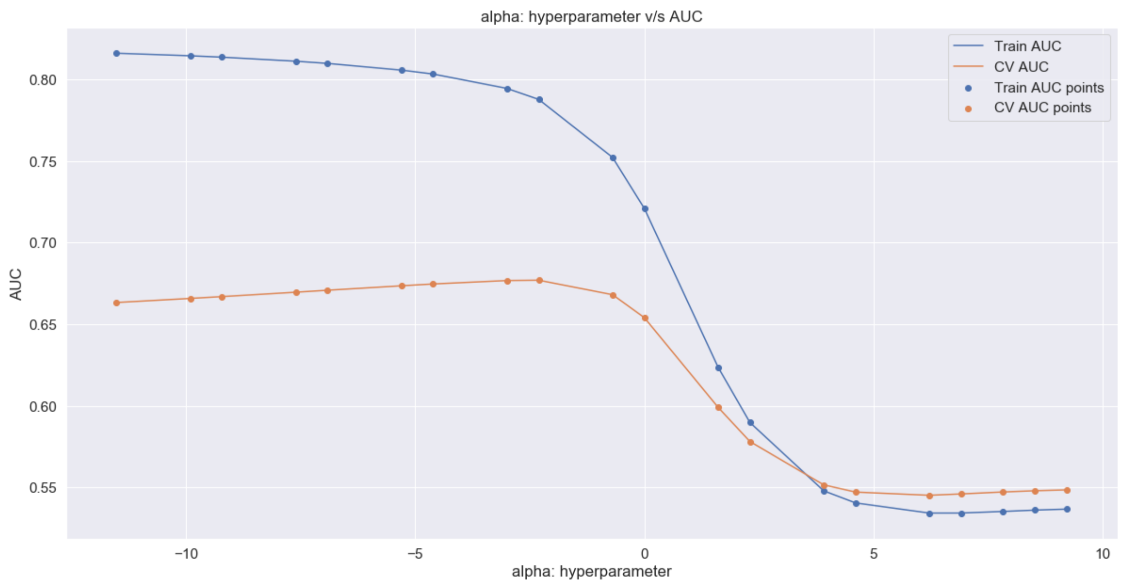
```
(24155, 14220) (24155,)
(36052, 14220) (36052,)
==============================================================================================
```

Random Alpha values

```
In [94]: train_auc = []
         cv_auc = []
         log_alphas =[]

         alphas = [0.00001, 0.00005, 0.0001, 0.0005, 0.001, 0.005, 0.01, 0.05, 0.1, 0.5, 1, 5,

         for i in tqdm(alphas):
             nb = MultinomialNB(alpha = i)
             nb.fit(X_tr, y_train)

             y_train_pred = batch_predict(nb, X_tr)
             y_cv_pred = batch_predict(nb, X_cr)


             # roc_auc_score(y_true, y_score) the 2nd parameter should be probability estimate
             # not the predicted outputs
             train_auc.append(roc_auc_score(y_train,y_train_pred))
             cv_auc.append(roc_auc_score(y_cv, y_cv_pred))

         for a in tqdm(alphas):
             b = math.log(a)
             log_alphas.append(b)
```

```
100%|| 20/20 [00:04<00:00,  4.32it/s]
100%|| 20/20 [00:00<00:00, 20054.05it/s]
```

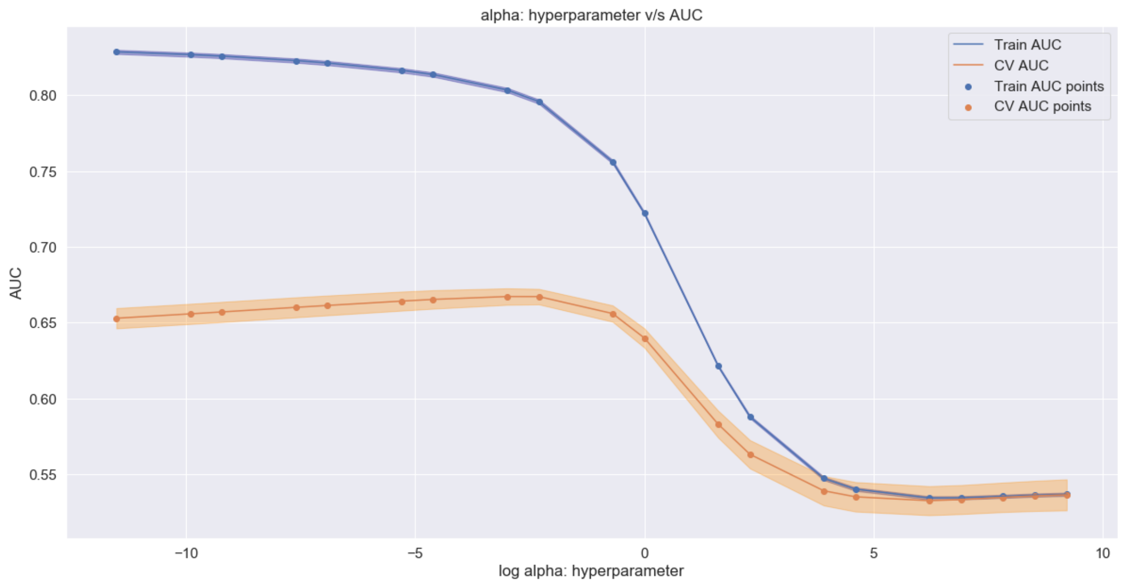
```
In [95]: plt.figure(figsize=(20,10))

         plt.plot(log_alphas, train_auc, label='Train AUC')
         plt.plot(log_alphas, cv_auc, label='CV AUC')

         plt.scatter(log_alphas, train_auc, label='Train AUC points')
         plt.scatter(log_alphas, cv_auc, label='CV AUC points')

         plt.legend()
         plt.xlabel("alpha: hyperparameter")
         plt.ylabel("AUC")
         plt.title("alpha: hyperparameter v/s AUC")
         plt.show()
```



23 Gridsearch-cv using cv = 10 (K fold cross validation)

```
In [96]: nb = MultinomialNB()

         parameters = {'alpha':[0.00001, 0.00005, 0.0001, 0.0005, 0.001, 0.005, 0.01, 0.05, 0.

         clf = GridSearchCV(nb, parameters, cv= 10, scoring='roc_auc')

         clf.fit(X_tr, y_train)

         train_auc= clf.cv_results_['mean_train_score']
         train_auc_std= clf.cv_results_['std_train_score']
         cv_auc = clf.cv_results_['mean_test_score']
         cv_auc_std= clf.cv_results_['std_test_score']

In [97]: alphas = [0.00001, 0.00005, 0.0001, 0.0005, 0.001, 0.005, 0.01, 0.05, 0.1, 0.5, 1, 5,
         log_alphas =[]

         for a in tqdm(alphas):
             b = math.log(a)
             log_alphas.append(b)

         plt.figure(figsize=(20,10))

         plt.plot(log_alphas, train_auc, label='Train AUC')
         # this code is copied from here: https://stackoverflow.com/a/48803361/4084039
         plt.gca().fill_between(log_alphas,train_auc - train_auc_std,train_auc + train_auc_std
         plt.plot(log_alphas, cv_auc, label='CV AUC')
         # this code is copied from here: https://stackoverflow.com/a/48803361/4084039
         plt.gca().fill_between(log_alphas,cv_auc - cv_auc_std,cv_auc + cv_auc_std,alpha=0.3,co

         plt.scatter(log_alphas, train_auc, label='Train AUC points')
         plt.scatter(log_alphas, cv_auc, label='CV AUC points')


         plt.legend()
         plt.xlabel("log alpha: hyperparameter")
         plt.ylabel("AUC")
         plt.title("alpha: hyperparameter v/s AUC")
         plt.show()
```

```
100%|| 20/20 [00:00<00:00, 20010.99it/s]
```



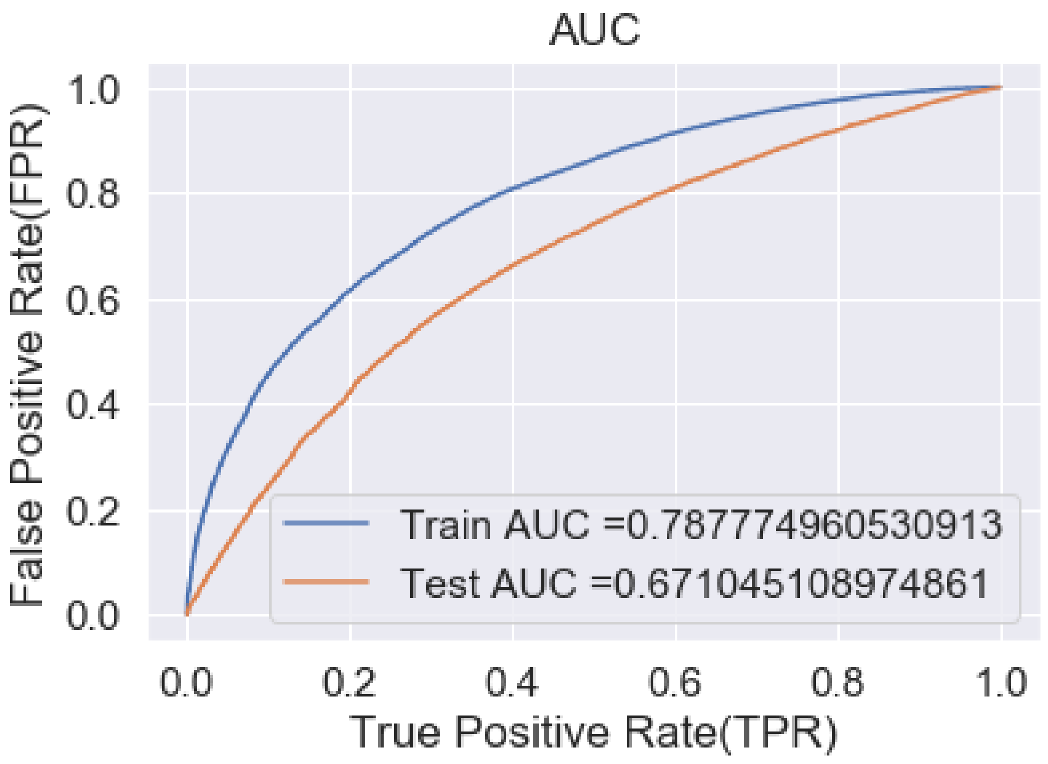
Summary of Alpha values for TFIDF model :

Alpha values ranging from 0.00001 to 10000.0 was taken and the results are as follows : 0.00001 as alpha values seemed to work very well on train data and the model seems to not work that efficiently on cross validation data. Values closer to 0.1 works pretty well both on Train data and Cross Validation data. Values more than 0.1 also doesnt seem to be effective on both Train and Cross Validation data.

0.1 as alpha value was chosen. Alpha values between 0.05 to 0.18 seemed to work better than the rest of the values

24 Train model using the best hyper-parameter value

```
In [98]: best_k_2 = 0.1
In [99]: nb_tfidf = MultinomialNB(alpha = best_k_2)

         nb_tfidf.fit(X_tr, y_train)
         # roc_auc_score(y_true, y_score) the 2nd parameter should be probability estimates of
         # not the predicted outputs

         y_train_pred = batch_predict(nb_tfidf, X_tr)
         y_test_pred = batch_predict(nb_tfidf, X_te)

         train_fpr, train_tpr, tr_thresholds = roc_curve(y_train, y_train_pred)
         test_fpr, test_tpr, te_thresholds = roc_curve(y_test, y_test_pred)

         plt.plot(train_fpr, train_tpr, label="Train AUC ="+str(auc(train_fpr, train_tpr)))
         plt.plot(test_fpr, test_tpr, label="Test AUC ="+str(auc(test_fpr, test_tpr)))
         plt.legend()
         plt.xlabel("True Positive Rate(TPR)")
         plt.ylabel("False Positive Rate(FPR)")
         plt.title("AUC")

         plt.show()
```



25 Confusion Matrix

Train data

```
In [100]: print("="*100)
          print("Train confusion matrix")
          print(confusion_matrix(y_train, predict(y_train_pred, tr_thresholds, train_fpr, trai
```

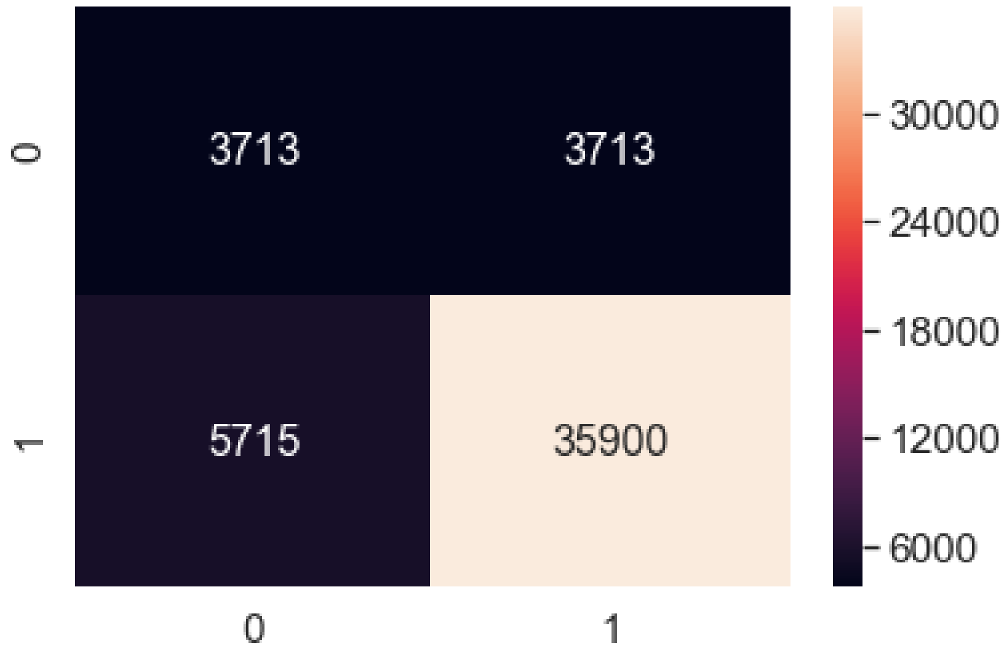
```
==========================================================================================
Train confusion matrix
the maximum value of tpr*(1-fpr) 0.25 for threshold 0.761
[[ 3713  3713]
 [ 5715 35900]]
```

```
In [101]: conf_matr_df_train_2 = pd.DataFrame(confusion_matrix(y_train, predict(y_train_pred, 
          range(2),range(2))
```

```
the maximum value of tpr*(1-fpr) 0.25 for threshold 0.761
```

```
In [102]: sns.set(font_scale=1.4)#for label size
          sns.heatmap(conf_matr_df_train_2, annot=True,annot_kws={"size": 16}, fmt='g')
```

```
Out[102]: <matplotlib.axes._subplots.AxesSubplot at 0x123800bef60>
```



Test data

```
In [103]: print("="*100)
          print("Test confusion matrix")
          print(confusion_matrix(y_test, predict(y_test_pred, tr_thresholds, test_fpr, test_fp
```

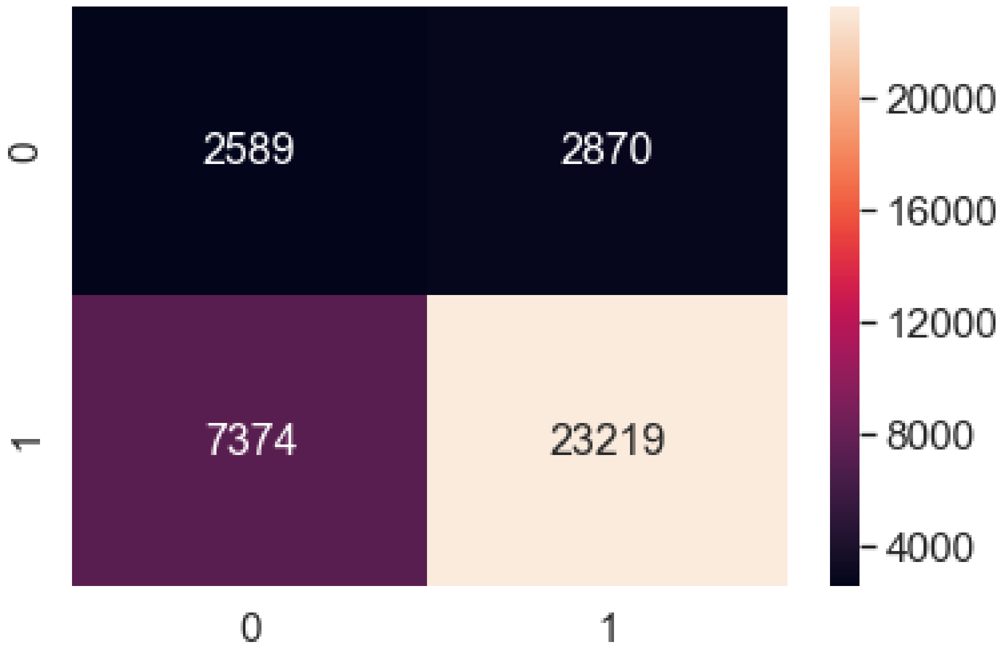
```
==========================================================================================
Test confusion matrix
the maximum value of tpr*(1-fpr) 0.24999999161092995 for threshold 0.817
[[ 2589  2870]
 [ 7374 23219]]
```

```
In [104]: conf_matr_df_test_2 = pd.DataFrame(confusion_matrix(y_test, predict(y_test_pred, tr_
          range(2),range(2))
```

```
the maximum value of tpr*(1-fpr) 0.24999999161092995 for threshold 0.817
```

```
In [105]: sns.set(font_scale=1.4)#for label size
          sns.heatmap(conf_matr_df_test_2, annot=True,annot_kws={"size": 16}, fmt='g')
```

```
Out[105]: <matplotlib.axes._subplots.AxesSubplot at 0x123c291d630>
```



26 Select best 10 features of both Positive and negative class for both the sets of data

SET 1 : BOW

```
In [106]: X_tr = hstack((categories_one_hot_train, sub_categories_one_hot_train, school_state_c
          X_te = hstack((categories_one_hot_test, sub_categories_one_hot_test, school_state_cat
          X_cr = hstack((categories_one_hot_cv, sub_categories_one_hot_cv, school_state_categor
```

```
In [107]: nb_bow = MultinomialNB(alpha = 0.5)

          nb_bow.fit(X_tr, y_train)
```

```
Out[107]: MultinomialNB(alpha=0.5, class_prior=None, fit_prior=True)
```

```
In [108]: bow_features_probs = {}

          for a in range(14210) :

              bow_features_probs[a] = nb_bow.feature_log_prob_[0,a]
```

```
In [109]: len(bow_features_probs.values())
```

```
Out[109]: 14210
```

```
In [110]: bow_features_names = []
          bow_features_names.extend(vectorizer_proj.get_feature_names())
          bow_features_names.extend(vectorizer_sub_proj.get_feature_names())
          bow_features_names.extend(vectorizer_states.get_feature_names())
          bow_features_names.extend(vectorizer_grade.get_feature_names())
          bow_features_names.extend(vectorizer_teacher.get_feature_names())
          bow_features_names.extend(["price"])
          bow_features_names.extend(["quantity"])
          bow_features_names.extend(["prev"])
          bow_features_names.extend(["title_word_count"])
          bow_features_names.extend(["essay_word_count"])
          bow_features_names.extend(vectorizer_bow_title.get_feature_names())
          bow_features_names.extend(vectorizer_bow_essay.get_feature_names())
          len(bow_features_names)
```

```
Out[110]: 14220
```

```
In [111]: final_bow_features = pd.DataFrame({'feature_prob_estimates' : nb_bow.feature_log_prol
          'feature_names' : bow_features_names})
```

```
In [112]: np.argsort('feature_prob_estimates')
```

```
Out[112]: array([0], dtype=int64)
```

```
In [113]: a = final_bow_features.sort_values(by = ['feature_prob_estimates'], ascending =False
```

27 25 Negative features from BOW model

```
In [114]: a.head(25)

Out[114]:        feature_prob_estimates     feature_names
          12578               -3.049329          students
          11605               -4.139411            school
          8417                -4.464421          learning
          4226                -4.610088         classroom
          9444                -4.809541               not
          8413                -4.823607             learn
          7328                -4.830481              help
          108                 -4.971767          quantity
          110                 -4.971767  title_word_count
          111                 -4.971767  essay_word_count
          107                 -4.971767             price
          9277                -5.016938            nannan
          8790                -5.031554              many
          9324                -5.143041              need
          14091               -5.218877              work
          4368                -5.350821              come
          109                 -5.353974              prev
          10858               -5.392599           reading
          8656                -5.404359              love
          2434                -5.406020              able
          8846                -5.415419         materials
          12000               -5.416889            skills
          5024                -5.449787               day
          13635               -5.458951               use
          4212                -5.503089             class

In [115]: final_bow_features_pos = pd.DataFrame({'feature_prob_estimates' : nb_bow.feature_log
          'feature_names' : bow_features_names})

In [116]: np.argsort('feature_prob_estimates')

Out[116]: array([0], dtype=int64)

In [117]: b = final_bow_features_pos.sort_values(by = ['feature_prob_estimates'], ascending = 
```

28 25 Positive features from BOW model

```
In [118]: b.head(25)

Out[118]:        feature_prob_estimates     feature_names
          12578               -3.049329          students
          11605               -4.139411            school
          8417                -4.464421          learning
```

```
4226                -4.610088       classroom
9444                -4.809541             not
8413                -4.823607           learn
7328                -4.830481            help
108                 -4.971767        quantity
110                 -4.971767  title_word_count
111                 -4.971767  essay_word_count
107                 -4.971767           price
9277                -5.016938          nannan
8790                -5.031554            many
9324                -5.143041            need
14091               -5.218877            work
4368                -5.350821            come
109                 -5.353974            prev
10858               -5.392599         reading
8656                -5.404359            love
2434                -5.406020            able
8846                -5.415419       materials
12000               -5.416889          skills
5024                -5.449787             day
13635               -5.458951             use
4212                -5.503089           class
```

29 SET 2 : TFIDF

```
In [119]: X_tr = hstack((categories_one_hot_train, sub_categories_one_hot_train, school_state_c
          X_te = hstack((categories_one_hot_test, sub_categories_one_hot_test, school_state_cat
          X_cr = hstack((categories_one_hot_cv, sub_categories_one_hot_cv, school_state_categor

In [120]: nb_tfidf = MultinomialNB(alpha = 0.1)

          nb_tfidf.fit(X_tr, y_train)

Out[120]: MultinomialNB(alpha=0.1, class_prior=None, fit_prior=True)

In [121]: tfidf_features_probs_neg = {}

          for a in range(14216) :

              tfidf_features_probs_neg[a] = nb_tfidf.feature_log_prob_[0,a]

In [122]: tfidf_features_names = []
          tfidf_features_names.extend(vectorizer_proj.get_feature_names())
          tfidf_features_names.extend(vectorizer_sub_proj.get_feature_names())
          tfidf_features_names.extend(vectorizer_states.get_feature_names())
          tfidf_features_names.extend(vectorizer_grade.get_feature_names())
          tfidf_features_names.extend(vectorizer_teacher.get_feature_names())
          tfidf_features_names.append(["price"])
          tfidf_features_names.append(["quantity"])
          tfidf_features_names.append(["prev_proposed_projects"])
          tfidf_features_names.append(["title_word_count"])
          tfidf_features_names.append(["essay_word_count"])
          tfidf_features_names.extend(vectorizer_tfidf_titles.get_feature_names())
          tfidf_features_names.extend(vectorizer_tfidf_essay.get_feature_names())
          len(tfidf_features_names)
```

```
Out[122]: 14220
```

```
In [142]: final_tfidf_features_neg = pd.DataFrame({'feature_prob_estimates' : nb_tfidf.feature_
          'feature_names' : tfidf_features_names})
```

```
In [143]: np.argsort('feature_prob_estimates')
```

```
Out[143]: array([0], dtype=int64)
```

```
In [144]: c = final_tfidf_features_neg.sort_values(by = ['feature_prob_estimates'], ascending =
```

30 25 Negative features from TFIDF model

```
In [126]: c.head(25)
```

```
Out[126]:       feature_prob_estimates feature_names
          0                   -9.180968       _Warmth
          9484                -9.180968      numeracy
          9473                -9.180968      november
          9474                -9.180968        novice
          9475                -9.180968      nowadays
          9476                -9.180968       nowhere
          9477                -9.180968      nuances
          9478                -9.180968       nuclear
          9479                -9.180968         nudge
          9480                -9.180968          numb
          9481                -9.180968        number
          9482                -9.180968      numbered
          9483                -9.180968       numbers
          9485                -9.180968      numerals
          9471                -9.180968        novels
          9486                -9.180968       numeric
          9487                -9.180968     numerical
          9488                -9.180968      numerous
          9489                -9.180968         nurse
          9490                -9.180968       nursery
          9491                -9.180968        nurses
          9492                -9.180968       nursing
          9493                -9.180968       nurture
          9494                -9.180968      nurtured
          9495                -9.180968      nurtures
```

```
In [130]: tfidf_features_probs_pos = {}

          for a in range(14220) :

              tfidf_features_probs_pos[a] = nb_tfidf.feature_log_prob_[1,a]

In [145]: final_tfidf_features_pos = pd.DataFrame({'feature_prob_estimates' : nb_tfidf.feature_
          'feature_names' : tfidf_features_names})

In [146]: np.argsort('feature_prob_estimates')

Out[146]: array([0], dtype=int64)

In [148]: d = final_tfidf_features_pos.sort_values(by = ['feature_prob_estimates'], ascending =
```

31 25 Positive features from TFIDF model

```
In [149]: d.head(25)

Out[149]:       feature_prob_estimates feature_names
          0                 -10.882261       _Warmth
          9484              -10.882261      numeracy
          9473              -10.882261      november
          9474              -10.882261        novice
          9475              -10.882261      nowadays
          9476              -10.882261       nowhere
          9477              -10.882261       nuances
          9478              -10.882261       nuclear
          9479              -10.882261         nudge
          9480              -10.882261          numb
          9481              -10.882261        number
          9482              -10.882261      numbered
          9483              -10.882261       numbers
          9485              -10.882261      numerals
          9471              -10.882261        novels
          9486              -10.882261       numeric
          9487              -10.882261     numerical
          9488              -10.882261      numerous
          9489              -10.882261         nurse
          9490              -10.882261       nursery
          9491              -10.882261        nurses
          9492              -10.882261       nursing
          9493              -10.882261       nurture
          9494              -10.882261      nurtured
          9495              -10.882261      nurtures
```

32 Conclusions

In [150]:
```
# Please compare all your models using Prettytable library
# http://zetcode.com/python/prettytable/

from prettytable import PrettyTable

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

x = PrettyTable()
x.field_names = ["Vectorizer", "Model", "Alpha:Hyper Parameter", "AUC"]

x.add_row(["BOW", "Naive Bayes", 0.5, 0.7])
x.add_row(["TFIDF", "Naive Bayes", 0.1, 0.67])

print(x)
```

```
+------------+-------------+-----------------------+------+
| Vectorizer |    Model    | Alpha:Hyper Parameter | AUC  |
+------------+-------------+-----------------------+------+
|    BOW     | Naive Bayes |          0.5          | 0.7  |
|   TFIDF    | Naive Bayes |          0.1          | 0.67 |
+------------+-------------+-----------------------+------+
```

In [151]:
```
y = PrettyTable()
y.field_names = ["Vectorizer", "Model", "K:Hyper Parameter", "AUC"]

y.add_row(["BOW", "KNN", 91, 0.63])
y.add_row(["TFIDF", "KNN", 85, 0.57])

print(y)
```

```
+------------+-------+-------------------+------+
| Vectorizer | Model | K:Hyper Parameter | AUC  |
+------------+-------+-------------------+------+
|    BOW     |  KNN  |         91        | 0.63 |
|   TFIDF    |  KNN  |         85        | 0.57 |
+------------+-------+-------------------+------+
```

Summary 1. Naive bayes seems to function better than KNN for both Bag of Words model (BOW) as well as Term Frequency Inverse Document Frequency model (TFIDF). 2. This can be observed by taking look at the difference in AUC measures for both the models. Clearly Naive Bayes is a better model. 3. Also, Naive Bayes takes very very less time to compute compared to the KNN model.