10_DonorsChoose_Clustering

February 28, 2020

1 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

How to scale current manual processes and resources to screen 500,000 projects so that they ca
<li>How to increase the consistency of project vetting across different volunteers to improve
<li>How to focus volunteer time on the applications that need the most assistance</li>
</ul>
```

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.

1.1 About the DonorsChoose Data Set

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

Feature	Description
`project_id`	A unique identifier for the proposed project. **Example:** `p036502`

`project_title` | Title of the project. **Examples:**
Art Will Make You Happy!
First Grade Fun
`project_grade_category` | Grade level of students for which the project is targeted. One of the following enumerated values:
Grades PreK-2
Grades 3-5
Grades 6-8
Grades 9-12
`project_subject_categories` | One or more (comma-separated) subject categories for the project from the following enumerated list of values:

Applied Learning
Care & Hunger
Health & Sports
History & Civics
Literacy & Language
Math & Science
Music & The Arts
Special Needs
Warmth
Examples:
Music & The Arts
Literacy & Language, Math & Science

`school_state` | State where school is located (Two-letter U.S. postal code). **Example:** `WY`

`project_subject_subcategories` | One or more (comma-separated) subject subcategories for the project. **Examples:**

Literacy
Literature & Writing, Social Sciences

`project_resource_summary` | An explanation of the resources needed for the project. **Example:**

My students need hands on literacy materials to manage sensory needs!

`project_essay_1` | First application essay

`project_essay_2` | *Second application essay* `project_essay_3` | Third application essay ***`project_essay_4`*** | *Fourth application essay* `project_submitted_datetime` | Datetime when project application was submitted. **Example:** `2016-04-28 12:43:56.245`

`teacher_id` | A unique identifier for the teacher of the proposed project. **Example:** `bdf8baa8fedef6bfeec7ae4ff1c15c56`

`teacher_prefix` | Teacher's title. One of the following enumerated values:

nan
Dr.
Mr.
Mrs.
Ms.
Teacher.

`teacher_number_of_previously_posted_projects` | Number of project applications previously submitted by the same teacher. **Example:** `2`

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

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

Feature	Description
`id`	A `project_id` value from the `train.csv` file. **Example:** `p036502`

Feature	Description
`description`	Desciption of the resource. **Example:** `Tenor Saxophone Reeds, Box of 25`
`quantity`	Quantity of the resource required. **Example:** `3`
`price`	Price of the resource required. **Example:** `9.95`

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

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

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

1.1.1 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_3: "Close by sharing why your project will make a difference"

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

project_essay_1: "Describe your students: What makes your students special? Specific details about their background, your neighborhood, and your school are all helpful."

project_essay_2: "About your project: How will these materials make a difference in your students' learning and improve their school lives?"

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

```
In [1]: %matplotlib inline
        import warnings
```

```
warnings.filterwarnings("ignore")

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

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

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

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

from tqdm import tqdm
import os

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

1.2 1.1 Reading Data

```
In [2]: project_data = pd.read_csv(r'C:\Users\ASUS\Downloads\Applied AI\Assignments - Applied A
        resource_data = pd.read_csv(r'C:\Users\ASUS\Downloads\Applied AI\Assignments - Applied
```

```
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

1.3 1.2 preprocessing of `project_subject_categories`

```
In [5]: catogories = list(project_data['project_subject_categories'].values)
        # remove special characters from list of strings python: https://stackoverflow.com/a/4

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

        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]))
```

1.4 1.3 preprocessing of project_subject_subcategories

In [6]:
```
sub_catogories = list(project_data['project_subject_subcategories'].values)
# remove special characters from list of strings python: https://stackoverflow.com/a/4

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

sub_cat_list = []
for i in sub_catogories:
    temp = ""
    # consider we have text like this "Math & Science, Warmth, Care & Hunger"
    for j in i.split(','): # it will split it in three parts ["Math & Science", "Warmt
        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 replace
        j = j.replace(' ','') # we are placeing all the ' '(space) with ''(empty) ex:"
        temp +=j.strip()+" "#" abc ".strip() will return "abc", remove the trailing sp
        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)

# count of all the words in corpus python: https://stackoverflow.com/a/22898595/408403
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]))
```

1.5 1.3 Text preprocessing

In [7]:
```
# 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 [8]:
```
project_data.head(2)
```

Out[8]:
```
   Unnamed: 0       id                         teacher_id teacher_prefix  \
0      160221  p253737  c90749f5d961ff158d4b4d1e7dc665fc           Mrs.
1      140945  p258326  897464ce9ddc600bced1151f324dd63a            Mr.

  school_state project_submitted_datetime project_grade_category  \
0           IN        2016-12-05 13:43:57          Grades PreK-2
1           FL        2016-10-25 09:22:10             Grades 6-8

                                        project_title  \
0  Educational Support for English Learners at Home
1             Wanted: Projector for Hungry Learners

                                       project_essay_1  \
0  My students are English learners that are work...
1  Our students arrive to our school eager to lea...

                                       project_essay_2 project_essay_3  \
0  \"The limits of your language are the limits o...             NaN
1  The projector we need for our school is very c...             NaN

  project_essay_4                         project_resource_summary  \
0             NaN  My students need opportunities to practice beg...
1             NaN  My students need a projector to help with view...

   teacher_number_of_previously_posted_projects  project_is_approved  \
0                                             0                   0
1                                             7                   1

               clean_categories           clean_subcategories  \
0             Literacy_Language                   ESL Literacy
1  History_Civics Health_Sports  Civics_Government TeamSports

                                              essay
0  My students are English learners that are work...
1  Our students arrive to our school eager to lea...
```

In [9]:
```
#### 1.4.2.3 Using Pretrained Models: TFIDF weighted W2V
```

In [10]:
```
# printing some random reviews
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
==================================================
The 51 fifth grade students that will cycle through my classroom this year all love learning, a
==================================================
How do you remember your days of school? Was it in a sterile environment with plain walls, rows
==================================================
My kindergarten students have varied disabilities ranging from speech and language delays, cogn
==================================================
The mediocre teacher tells. The good teacher explains. The superior teacher demonstrates. The g
==================================================
```

```
In [11]: # 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 [12]: sent = decontracted(project_data['essay'].values[20000])
         print(sent)
         print("="*50)
```

```
My kindergarten students have varied disabilities ranging from speech and language delays, cogn
==================================================
```

```
In [13]: # \r \n \t remove from string python: http://texthandler.com/info/remove-line-breaks-p
         sent = sent.replace('\\r', ' ')
         sent = sent.replace('\\"', ' ')
         sent = sent.replace('\\n', ' ')
         print(sent)
```

```
My kindergarten students have varied disabilities ranging from speech and language delays, cogn
```

```
In [14]: #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 cogni
```

```
In [15]: # 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"]
```

```
In [16]: # Combining all the above stundents
         from tqdm import tqdm
         preprocessed_essays = []
         # tqdm is for printing the status bar
         for sentance in tqdm(project_data['essay'].values):
             sent = decontracted(sentance)
             sent = sent.replace('\\r', ' ')
             sent = sent.replace('\\"', ' ')
             sent = sent.replace('\\n', ' ')
             sent = re.sub('[^A-Za-z0-9]+', ' ', sent)
             # https://gist.github.com/sebleier/554280
             sent = ' '.join(e for e in sent.split() if e not in stopwords)
             preprocessed_essays.append(sent.lower().strip())
         project_data['preprocessed_essays'] = preprocessed_essays
```

```
100%|| 109248/109248 [01:56<00:00, 936.34it/s]
```

```
In [17]: # after preprocesing
         preprocessed_essays[20000]
```

```
Out[17]: 'my kindergarten students varied disabilities ranging speech language delays cognitive
```

```
In [18]: project_data.head(2)
```

```
Out[18]:    Unnamed: 0       id                          teacher_id teacher_prefix  \
         0      160221  p253737  c90749f5d961ff158d4b4d1e7dc665fc           Mrs.
         1      140945  p258326  897464ce9ddc600bced1151f324dd63a            Mr.
```

```
  school_state project_submitted_datetime project_grade_category  \
0           IN        2016-12-05 13:43:57          Grades PreK-2
1           FL        2016-10-25 09:22:10             Grades 6-8

                                    project_title  \
0  Educational Support for English Learners at Home
1             Wanted: Projector for Hungry Learners

                                    project_essay_1  \
0  My students are English learners that are work...
1  Our students arrive to our school eager to lea...

                                    project_essay_2 project_essay_3  \
0  \"The limits of your language are the limits o...             NaN
1  The projector we need for our school is very c...             NaN

  project_essay_4                         project_resource_summary  \
0             NaN  My students need opportunities to practice beg...
1             NaN  My students need a projector to help with view...

   teacher_number_of_previously_posted_projects  project_is_approved  \
0                                             0                    0
1                                             7                    1

               clean_categories           clean_subcategories  \
0             Literacy_Language                   ESL Literacy
1  History_Civics Health_Sports  Civics_Government TeamSports

                                              essay  \
0  My students are English learners that are work...
1  Our students arrive to our school eager to lea...

                                  preprocessed_essays
0  my students english learners working english s...
1  our students arrive school eager learn they po...
```

1.4 Preprocessing of project_title

In [19]:
```
# similarly you can preprocess the titles also
```

In [20]:
```
# 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 [21]: sent = decontracted(project_data['project_title'].values[2000])
         print(sent)
         print("="*50)
```

```
Steady Stools for Active Learning
==================================================
```

```
In [22]: # \r \n \t remove from string python: http://texthandler.com/info/remove-line-breaks-p
         sent = sent.replace('\\r', ' ')
         sent = sent.replace('\\"', ' ')
         sent = sent.replace('\\n', ' ')
         print(sent)
```

```
Steady Stools for Active Learning
```

```
In [23]: #remove spacial character: https://stackoverflow.com/a/5843547/4084039
         sent = re.sub('[^A-Za-z0-9]+', ' ', sent)
         print(sent)
```

```
Steady Stools for Active Learning
```

```
In [24]: # 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
                     "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', "t
                     'am', 'is', 'are', 'was', 'were', 'be', 'been', 'being', 'have', 'has', 'h
                     'did', 'doing', 'a', 'an', 'the', 'and', 'but', 'if', 'or', 'because', 'as
                     'at', 'by', 'for', 'with', 'about', 'against', 'between', 'into', 'throug
                     'above', 'below', 'to', 'from', 'up', 'down', 'in', 'out', 'on', 'off', 'o
                     'then', 'once', 'here', 'there', 'when', 'where', 'why', 'how', 'all', 'an
                     'most', 'other', 'some', 'such', 'only', 'own', 'same', 'so', 'than', 'too
                     's', 't', 'can', 'will', 'just', 'don', "don't", 'should', "should've", 'n
```

```
                    '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"]
```

In [25]:
```
# Combining all the above stundents
from tqdm import tqdm
preprocessed_titles = []
# tqdm is for printing the status bar
for sentance in tqdm(project_data['project_title'].values):
    sent = decontracted(sentance)
    sent = sent.replace('\\r', ' ')
    sent = sent.replace('\\"', ' ')
    sent = sent.replace('\\n', ' ')
    sent = re.sub('[^A-Za-z0-9]+', ' ', sent)
    # https://gist.github.com/sebleier/554280
    sent = ' '.join(e for e in sent.split() if e.lower() not in stopwords)
    preprocessed_titles.append(sent.lower().strip())
project_data['preprocessed_titles'] = preprocessed_titles
```

```
100%|| 109248/109248 [00:05<00:00, 20516.25it/s]
```

In [26]:
```
preprocessed_titles[2000]
```

Out[26]:
```
'steady stools active learning'
```

In [27]:
```
project_data.head(2)
```

Out[27]:
```
   Unnamed: 0       id                         teacher_id teacher_prefix  \
0      160221  p253737  c90749f5d961ff158d4b4d1e7dc665fc           Mrs.
1      140945  p258326  897464ce9ddc600bced1151f324dd63a            Mr.

  school_state project_submitted_datetime project_grade_category  \
0           IN        2016-12-05 13:43:57          Grades PreK-2
1           FL        2016-10-25 09:22:10             Grades 6-8

                                      project_title  \
0  Educational Support for English Learners at Home
1            Wanted: Projector for Hungry Learners

                                      project_essay_1  \
0  My students are English learners that are work...
1  Our students arrive to our school eager to lea...

                                      project_essay_2 project_essay_3  \
0  \"The limits of your language are the limits o...             NaN
1  The projector we need for our school is very c...             NaN
```

```
  project_essay_4                            project_resource_summary  \
0             NaN  My students need opportunities to practice beg...
1             NaN  My students need a projector to help with view...

   teacher_number_of_previously_posted_projects  project_is_approved  \
0                                             0                    0
1                                             7                    1

               clean_categories           clean_subcategories  \
0             Literacy_Language                   ESL Literacy
1  History_Civics Health_Sports  Civics_Government TeamSports

                                              essay  \
0  My students are English learners that are work...
1  Our students arrive to our school eager to lea...

                                  preprocessed_essays  \
0  my students english learners working english s...
1  our students arrive school eager learn they po...

                         preprocessed_titles
0  educational support english learners home
1           wanted projector hungry learners
```

1.6 1.5 Preparing data for models

```
In [28]: project_data.columns
```

```
Out[28]: Index(['Unnamed: 0', 'id', 'teacher_id', 'teacher_prefix', 'school_state',
                'project_submitted_datetime', 'project_grade_category', 'project_title',
                'project_essay_1', 'project_essay_2', 'project_essay_3',
                'project_essay_4', 'project_resource_summary',
                'teacher_number_of_previously_posted_projects', 'project_is_approved',
                'clean_categories', 'clean_subcategories', 'essay',
                'preprocessed_essays', 'preprocessed_titles'],
               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 (optinal)

- quantity : numerical (optinal)
- teacher_number_of_previously_posted_projects : numerical
- price : numerical

2 Train Test split

In [29]:
```
project_data = project_data.sample(n=50000)
```

In [30]:
```
y = project_data['project_is_approved'].values
X = project_data.drop(['project_is_approved'], axis=1)
X.head(2)
```

Out[30]:

	Unnamed: 0	id	teacher_id	teacher_prefix	school_state	project_submitted_datetime	project_grade_category	project_title	project_essay_1	project_essay_2	project_essay_3	project_essay_4	project_resource_summary	teacher_number_of_previously_posted_projects
8583	72676	p163667	f2d85b99ee3b171c9c111e8bec1e622a	Ms.	MO	2016-07-06 22:00:42	Grades PreK-2	Snug as a Bug in a Rug	As a teacher in a low-income/high poverty scho...	The carpet I have been using is old, falling a...	NaN	NaN	My students need a classroom rug for whole gro...	0
41191	121111	p022712	9152a4722aa7da39f3774d37f7ba7e85	Teacher	CA	2016-05-01 22:00:15	Grades 3-5	INNOVATIVE TECHNOLOGY FOR INNOVATIVE LEARNING	My students are very curious children and arri...	I have 15 students in my class. These boys and...	My students will use the requested materials t...	Since our environment is bombarded by many har...	My students need science books,water testing k...	15

```
                     clean_categories  \
8583   Literacy_Language Math_Science
41191                    Math_Science

                          clean_subcategories  \
8583                        Literacy Mathematics
41191  EnvironmentalScience Health_LifeScience

                                                   essay  \
8583   As a teacher in a low-income/high poverty scho...
41191  My students are very curious children and arri...

                                     preprocessed_essays  \
8583   as teacher low income high poverty school dist...
41191  my students curious children arrive school man...

                             preprocessed_titles
8583                                 snug bug rug
41191  innovative technology innovative learning
```

2.0.1 1.5.1 Vectorizing Categorical data

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

In [31]:
```
# we use count vectorizer to convert the values into one
from sklearn.feature_extraction.text import CountVectorizer
vectorizer = CountVectorizer(vocabulary=list(sorted_cat_dict.keys()), lowercase=False
categories_one_hot = vectorizer.fit_transform(X['clean_categories'].values)
print(vectorizer.get_feature_names())
print("Shape of matrix after one hot encodig ",categories_one_hot.shape)
```

```
['Warmth', 'Care_Hunger', 'History_Civics', 'Music_Arts', 'AppliedLearning', 'SpecialNeeds', '
Shape of matrix after one hot encodig  (50000, 9)
```

In [32]:
```
# we use count vectorizer to convert the values into one
vectorizer = CountVectorizer(vocabulary=list(sorted_sub_cat_dict.keys()), lowercase=Fa
sub_categories_one_hot = vectorizer.fit_transform(X['clean_subcategories'].values)
print(vectorizer.get_feature_names())
print("Shape of matrix after one hot encodig ",sub_categories_one_hot.shape)
```

```
['Economics', 'CommunityService', 'FinancialLiteracy', 'ParentInvolvement', 'Extracurricular',
Shape of matrix after one hot encodig  (50000, 30)
```

In [33]:
```
# you can do the similar thing with state, teacher_prefix and project_grade_category
```

In [34]:
```
# count of all the words in corpus python: https://stackoverflow.com/a/22898595/40840
my_counter = Counter()
```

```
        for word in project_data['school_state'].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]))
```

```
In [35]: # we use count vectorizer to convert the values into one
        from sklearn.feature_extraction.text import CountVectorizer
        vectorizer = CountVectorizer(vocabulary=list(sorted_sub_cat_dict.keys()), lowercase=Fa
        state_one_hot = vectorizer.fit_transform(X['school_state'].values)
        print(vectorizer.get_feature_names())
        print("Shape of matrix after one hot encodig ",state_one_hot.shape)
```

```
['VT', 'WY', 'ND', 'MT', 'RI', 'DE', 'NE', 'AK', 'NH', 'SD', 'DC', 'HI', 'ME', 'WV', 'NM', 'KS
Shape of matrix after one hot encodig  (50000, 51)
```

```
In [36]: # count of all the words in corpus python: https://stackoverflow.com/a/22898595/40840
        my_countr = Counter()
        for word in project_data['teacher_prefix'].values.astype('str'):  #https://stackoverf
            my_countr.update(word.split())

        sub_fix_dict = dict(my_countr)
        sorted_sub_fix_dict = dict(sorted(sub_fix_dict.items(), key=lambda kv: kv[1]))
```

```
In [37]: # we use count vectorizer to convert the values into one
        from sklearn.feature_extraction.text import CountVectorizer
        vectorizer = CountVectorizer(vocabulary=list(sorted_sub_fix_dict.keys()), lowercase=Fa
        prefix_one_hot = vectorizer.fit_transform(X['teacher_prefix'].values.astype('str'))
        print(vectorizer.get_feature_names())
        print("Shape of matrix after one hot encodig ",prefix_one_hot.shape)
```

```
['nan', 'Dr.', 'Teacher', 'Mr.', 'Ms.', 'Mrs.']
Shape of matrix after one hot encodig  (50000, 6)
```

```
In [38]: # count of all the words in corpus python: https://stackoverflow.com/a/22898595/40840
        my_countr1 = Counter()
        for word in project_data['project_grade_category'].values:
            my_countr1.update(word.split())

        sub_grade_dict = dict(my_countr1)
        sorted_sub_grade_dict = dict(sorted(sub_grade_dict.items(), key=lambda kv: kv[1]))
```

```
In [39]: # we use count vectorizer to convert the values into one
        from sklearn.feature_extraction.text import CountVectorizer
        vectorizer = CountVectorizer(vocabulary=list(sorted_sub_grade_dict.keys()), lowercase=
        grade_one_hot = vectorizer.fit_transform(project_data['project_grade_category'].values
        print(vectorizer.get_feature_names())
        print("Shape of matrix after one hot encodig ",grade_one_hot.shape)
```

```
['9-12', '6-8', '3-5', 'PreK-2', 'Grades']
Shape of matrix after one hot encodig  (50000, 5)
```

2.0.2 1.5.3 Vectorizing Numerical features

```
In [48]: price_data = resource_data.groupby('id').agg({'price':'sum', 'quantity':'sum'}).reset_
         X = pd.merge(X, price_data, on='id', how='left')
         X.head(2)
```

Out[48]:

	Unnamed: 0	id	teacher_id	teacher_prefix	school_state	project_submitted_datetime	project_grade_category	project_title	project_essay_1	project_essay_2	...	clean_subcategories	essay	preprocessed_essays	preprocessed_titles	price_x	quantity_x	price_y	quantity_y	price	quantity
0	72676	p163667	f2d85b99ee3b171c9c111e8bec1e622a	Ms.	MO	2016-07-06 22:00:42	Grades PreK-2	Snug as a Bug in a Rug	As a teacher in a low-income/high poverty scho...	The carpet I have been using is old, falling a...	...	Literacy Mathematics	As a teacher in a low-income/high poverty scho...	as teacher low income high poverty school dist...	snug bug rug	479.00	1	479.00	1	479.00	1
1	121111	p022712	9152a4722aa7da39f3774d37f7ba7e85	Teacher	CA	2016-05-01 22:00:15	Grades 3-5	INNOVATIVE TECHNOLOGY FOR INNOVATIVE LEARNING	My students are very curious children and arri...	I have 15 students in my class. These boys and...	...	EnvironmentalScience Health_LifeScience	My students are very curious children and arri...	my students curious children arrive school man...	innovative technology innovative learning	1612.12	52	1612.12	52	1612.12	52

```
         1         52  1612.12        52

         [2 rows x 25 columns]
```

```
In [51]: #Normalizing the numerical features: Price

         from sklearn.preprocessing import Normalizer
         price_scalar = Normalizer()
         price_scalar.fit(X['price'].values.reshape(-1,1)) # finding the mean and standard dev

         x_price_nrm = price_scalar.transform(X['price'].values.reshape(-1,1))


         print("After vectorizations")
         print(x_price_nrm.shape)
```

```
After vectorizations
(50000, 1)
```

```
In [52]: #Normalizing the numerical features:quantity

         from sklearn.preprocessing import StandardScaler
         quantity_scalar = Normalizer()
         quantity_scalar.fit(X['quantity'].values.reshape(-1,1)) # finding the mean and standa

         x_quantity_nrm = quantity_scalar.transform(X['quantity'].values.reshape(-1,1))

         print("After vectorizations")
         print(x_quantity_nrm.shape)
```

```
After vectorizations
(50000, 1)
```

__ Computing Sentiment Scores__

```
In [0]: import nltk
        from nltk.sentiment.vader import SentimentIntensityAnalyzer

        # import nltk
        # nltk.download('vader_lexicon')

        sid = SentimentIntensityAnalyzer()

        for_sentiment = 'a person is a person no matter how small dr seuss i teach the smalles
        for learning my students learn in many different ways using all of our senses and multi
        of techniques to help all my students succeed students in my class come from a variety
        for wonderful sharing of experiences and cultures including native americans our schoo
```

```
learners which can be seen through collaborative student project based learning in and
in my class love to work with hands on materials and have many different opportunities
mastered having the social skills to work cooperatively with friends is a crucial aspe
montana is the perfect place to learn about agriculture and nutrition my students love
in the early childhood classroom i have had several kids ask me can we try cooking wit
and create common core cooking lessons where we learn important math and writing conce
food for snack time my students will have a grounded appreciation for the work that we
of where the ingredients came from as well as how it is healthy for their bodies this
nutrition and agricultural cooking recipes by having us peel our own apples to make ho
and mix up healthy plants from our classroom garden in the spring we will also create
shared with families students will gain math and literature skills as well as a life l
nannan'
ss = sid.polarity_scores(for_sentiment)

for k in ss:
    print('{0}: {1}, '.format(k, ss[k]), end='')

# we can use these 4 things as features/attributes (neg, neu, pos, compound)
# neg: 0.0, neu: 0.753, pos: 0.247, compound: 0.93
```

```
D:\installed\Anaconda3\lib\site-packages\nltk\twitter\__init__.py:20: UserWarning:

The twython library has not been installed. Some functionality from the twitter package will n
```

```
neg: 0.01, neu: 0.745, pos: 0.245, compound: 0.9975,
```

3 Set 1 : TFIDF

4 Vectorizing text data

In [54]:
```
# On preprocessed_Essay
from sklearn.feature_extraction.text import TfidfVectorizer
vectorizer8 = TfidfVectorizer(min_df=10,ngram_range = (1,4),max_features=5000)
preprocessed_essays_tfidf = vectorizer8.fit_transform(X['preprocessed_essays'])
print("Shape of matrix after one hot encodig ",preprocessed_essays_tfidf.shape)
```

```
Shape of matrix after one hot encodig  (50000, 5000)
```

In [55]:
```
# On Clean_title
vectorizer9 = TfidfVectorizer(min_df=10,ngram_range = (1,4),max_features=5000)
preprocessed_titles_tfidf = vectorizer9.fit_transform(X['preprocessed_titles'])
print("Shape of matrix after one hot encodig ",preprocessed_titles_tfidf.shape)
```

```
Shape of matrix after one hot encodig  (50000, 3486)
```

5 Concatinating all the features

```
In [56]: # Concatenate TFIDF
         from scipy.sparse import hstack
         X_tfidf=hstack((preprocessed_essays_tfidf,preprocessed_titles_tfidf, x_price_nrm, x_qu
                         sub_categories_one_hot)).tocsr()

         print(X_tfidf.shape)
```

```
(50000, 8589)
```

6 Assignment 10: Clustering

- step 1: Choose any vectorizer (data matrix) that you have worked in any of the assignments, and got the best AUC value.
- step 2: Choose any of the feature selection/reduction algorithms ex: selectkbest features, pretrained word vectors, model based feature selection etc and reduce the number of features to 5k features.
- step 3: Apply all three kmeans, Agglomerative clustering, DBSCAN
  - K-Means Clustering: Find the best 'k' using the elbow-knee method (plot k vs inertia_)
  - Agglomerative Clustering: Apply agglomerative algorithm and try a different number of clusters like 2,5 etc. As this is very computationally expensive, take 5k datapoints only to perform hierarchical clustering because they do take a considerable amount of time to run.
  - DBSCAN Clustering: Find the best 'eps' using the elbow-knee method. Take 5k datapoints only.
- step 4: Summarize each cluster by manually observing few points from each cluster.
- step 5: You need to plot the word cloud with essay text for each cluster for each of algorithms mentioned in step 3.

2. Clustering

2.1 Choose the best data matrix on which you got the best AUC

```
In [0]: # please write all the code with proper documentation, and proper titles for each subs
        # go through documentations and blogs before you start coding
        # first figure out what to do, and then think about how to do.
        # reading and understanding error messages will be very much helpfull in debugging you
        # when you plot any graph make sure you use b
            # a. Title, that describes your plot, this will be very helpful to the reader
            # b. Legends if needed
            # c. X-axis label
            # d. Y-axis label
```

```
In [63]: import matplotlib.pyplot as plt
         from matplotlib.image import imread
```

```
import pandas as pd
import seaborn as sns
from sklearn.datasets.samples_generator import (make_blobs,
                                                 make_circles,
                                                 make_moons)
from sklearn.cluster import KMeans, SpectralClustering
from sklearn.preprocessing import StandardScaler
from sklearn.metrics import silhouette_samples, silhouette_score
```

7 Dimensionality Reduction on the selected features

In [58]:
```
from sklearn.feature_selection import SelectKBest, chi2

Selector = SelectKBest(chi2,k=1000)
Selector.fit(X_tfidf,y)

X_tfidf_sample = Selector.transform(X_tfidf)
print(X_tfidf_sample.shape)

X_tfidf_new = X_tfidf_sample[0:5000]
print(X_tfidf_new.shape)
```

```
(50000, 1000)
(5000, 1000)
```

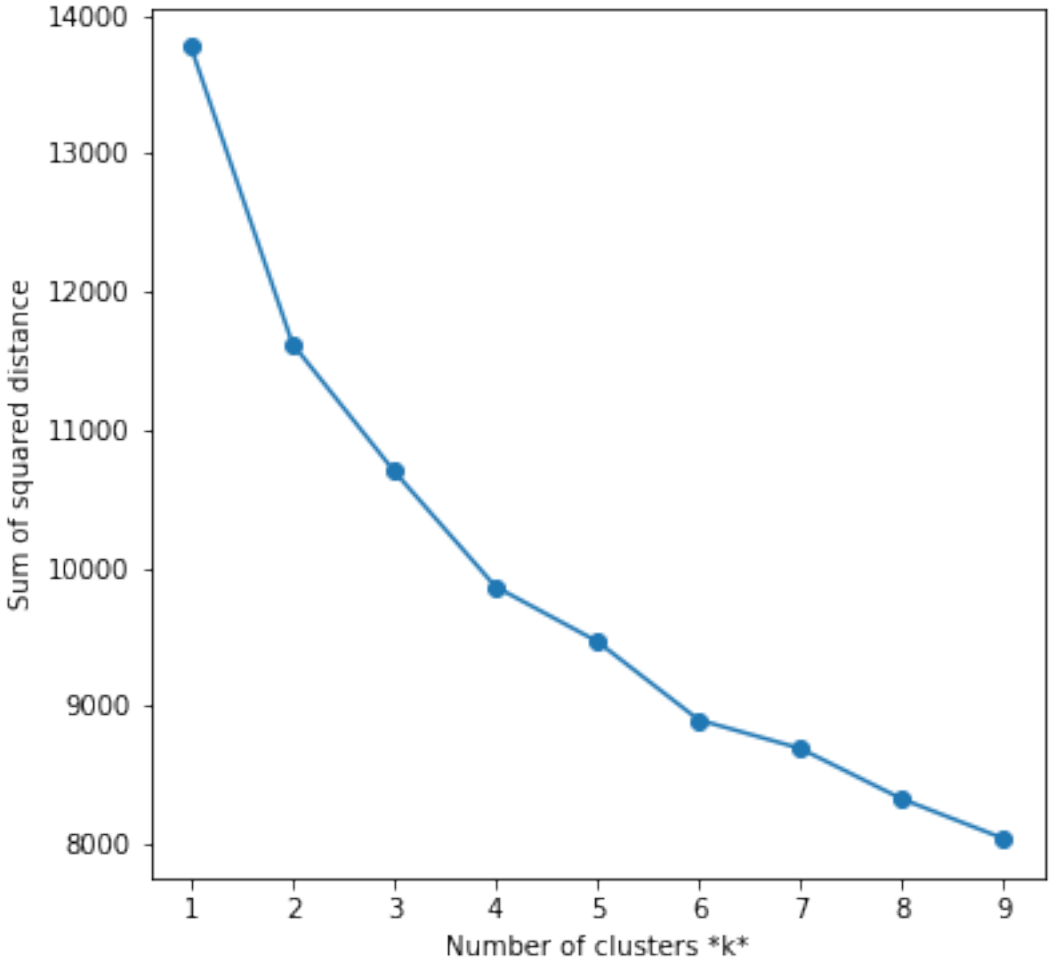
In [59]:
```
#to find the hyperparameter for K-means clutering, we have used elbow method to find
```

In [60]:
```
# Run the Kmeans algorithm and get the index of data points clusters

from sklearn.cluster import KMeans
sse = []
list_k = list(range(1, 10))

for k in list_k:
    km = KMeans(n_clusters=k)
    km.fit(X_tfidf_new)
    sse.append(km.inertia_) ## Sum of squared distances of samples to their closest c

# Plot sse against k
plt.figure(figsize=(6, 6))
plt.plot(list_k, sse, '-o')
plt.xlabel(r'Number of clusters *k*')
plt.ylabel('Sum of squared distance');
```




```
In [76]: # Run local implementation of kmeans
         km = KMeans(n_clusters=6, max_iter=100)
         km.fit(X_tfidf_new)
         kms = km.fit_predict(X_tfidf_new)
         centroids = km.cluster_centers_
```

```
In [ ]: #The labels_ property contains the list of clusters and their respective points
        labels = km.labels_
        labels.shape[0]
```

8 Summarizing each cluster by manually observing few points from each cluster

```
In [ ]: clean_essays = X["essays"].values

        cluster1 = []
```

```
cluster2 = []
cluster3 = []
cluster4 = []
cluster5 = []
cluster6 = []

for k in range(labels.shape[0]):
    if labels[k] == 0:
        cluster1.append(clean_essays[k])
    elif labels[k] == 1:
        cluster2.append(clean_essays[k])
    elif labels[k] == 2:
        cluster3.append(clean_essays[k])
    elif labels[k] == 3:
        cluster4.append(clean_essays[k])
    elif labels[k] == 4:
        cluster5.append(clean_essays[k])
    elif lables[k] == 5:
        cluster6.append(clean_essays[k])

print("Number of data points of essays in cluster 1 :",len(cluster1))
print("Number of data points of essays in cluster 2 :",len(cluster2))
print("Number of data points of essays in cluster 3 :",len(cluster3))
print("Number of data points of essays in cluster 4 :",len(cluster4))
print("Number of data points of essays in cluster 5 :",len(cluster5))
print("Number of data points of essays in cluster 6 :",len(cluster6))
```

9 Plotting Word Cloud for each cluster of Kmeans

```
In [ ]: # for cluster 1,2,3,4 & 5 respectively

from wordcloud import WordCloud
essays_wc = [cluster1 ,cluster2,cluster3,cluster4,cluster5,cluster6]
print("Word Cloud for clusters:")
for j in range(0,6):
    word=str(essays_wc[j])
    wordcloud4 = WordCloud(width = 800, height = 800,
                background_color ='black').generate(word)

    # plot the WordCloud image
    plt.figure(figsize = (5, 5), facecolor = None)

    plt.imshow(wordcloud4)
    plt.axis("off")
    plt.tight_layout(pad = 0)

    plt.show()
```

2.6 Apply AgglomerativeClustering

In [ ]:
```
#https://towardsdatascience.com/machine-learning-algorithms-part-12-hierarchical-agglo
```

In [ ]:
```
import pandas as pd
import numpy as np
from matplotlib import pyplot as plt
from sklearn.cluster import AgglomerativeClustering
import scipy.cluster.hierarchy as sch
```

In [ ]:
```
#dendograms are helpful to find the number of optimal clusters to use.
dendrogram = sch.dendrogram(sch.linkage(X_tfidf_new, method='ward'))
```

In [ ]:
```
#AgglomerativeClustering using the euclidean distance as the measure of distance betwe
model = AgglomerativeClustering(n_clusters=5, affinity='euclidean', linkage='ward')
model.fit(X_tfidf_new)
```

In [ ]:
```
#The labels_ property contains the list of clusters and their respective points
labels1 = model.labels_
labels1.shape[0]
```

10 Summarizing each cluster by manually observing few points from each cluster

In [ ]:
```
clean_essays = X["essays"].values

cluster1 = []
cluster2 = []
cluster3 = []
cluster4 = []
cluster5 = []
cluster6 = []

for k in range(labels.shape[0]):
    if labels[k] == 0:
        cluster1.append(clean_essays[k])
    elif labels[k] == 1:
        cluster2.append(clean_essays[k])
    elif labels[k] == 2:
        cluster3.append(clean_essays[k])
    elif labels[k] == 3:
        cluster4.append(clean_essays[k])
    elif labels[k] == 4:
        cluster5.append(clean_essays[k])
    elif lables[k] == 5:
        cluster6.append(clean_essays[k])

print("Number of data points of essays in cluster 1 :",len(cluster1))
```

```
print("Number of data points of essays in cluster 2 :",len(cluster2))
print("Number of data points of essays in cluster 3 :",len(cluster3))
print("Number of data points of essays in cluster 4 :",len(cluster4))
print("Number of data points of essays in cluster 5 :",len(cluster5))
print("Number of data points of essays in cluster 6 :",len(cluster6))
```

11 Plotting Word Cloud for each cluster of Kmeans

In [ ]:
```
# for cluster 1,2,3,4 & 5 respectively

from wordcloud import WordCloud
essays_wc = [cluster1 ,cluster2,cluster3,cluster4,cluster5,cluster6]
print("Word Cloud for clusters:")
for j in range(0,6):
    word=str(essays_wc[j])
    wordcloud4 = WordCloud(width = 800, height = 800,
                background_color ='black').generate(word)

    # plot the WordCloud image
    plt.figure(figsize = (5, 5), facecolor = None)

    plt.imshow(wordcloud4)
    plt.axis("off")
    plt.tight_layout(pad = 0)

    plt.show()
```

2.7 Apply DBSCAN

In [0]:
```
# please write all the code with proper documentation, and proper titles for each subs
# go through documentations and blogs before you start coding
# first figure out what to do, and then think about how to do.
# reading and understanding error messages will be very much helpfull in debugging you
# when you plot any graph make sure you use
    # a. Title, that describes your plot, this will be very helpful to the reader
    # b. Legends if needed
    # c. X-axis label
    # d. Y-axis label
```

In [ ]:
```
#the code for DBSCAN is taken from below link, which is slightly modified.
#https://medium.com/sfu-big-data/exploration-of-fundamental-clustering-algorithms-k-me
```

In [ ]:
```
X_tfidf_new = X_tfidf_sample[0:5000]
print(X_tfidf_new.shape)
```

In [ ]:
```
#Using elbow method to find the Eps.
```

In [ ]:
```
where NN = min_sample
min_samples = 2*X_tfidf_new.shape[1]
```

```
        from sklearn.neighbors import NearestNeighbors
        neigh = NearestNeighbors(n_neighbors=min_samples)
        nbrs = neigh.fit(X_tfidf_new)

        distances, indices = nbrs.kneighbors(X_tfidf_new)
        # distances[:,minPts-1] gives the distances to the kth nearerst neighbour
        distanceDec = sorted(distances[:,minPts-1], reverse=True)
        plt.plot(distanceDec)
        plt.ylabel('Distance of point')
        plt.xlabel('Points(sample) sorted by distance')
        plt.title('Elbow Method For Optimal eps')
        plt.show()
```

```
In [ ]: m = DBSCAN(eps=0.3, min_samples=min_samples)
        m.fit(X)
```

```
In [ ]: #The labels_ property contains the list of clusters and their respective points.
        labels = m.labels_
        labels.shape[0]
```

12 Summarizing each cluster by manually observing few points from each cluster

```
In [ ]: clean_essays = X["essays"].values

        cluster1 = []
        cluster2 = []
        cluster3 = []
        cluster4 = []
        cluster5 = []
        cluster6 = []

        for k in range(labels.shape[0]):
            if labels[k] == 0:
                cluster1.append(clean_essays[k])
            elif labels[k] == 1:
                cluster2.append(clean_essays[k])
            elif labels[k] == 2:
                cluster3.append(clean_essays[k])
            elif labels[k] == 3:
                cluster4.append(clean_essays[k])
            elif labels[k] == 4:
                cluster5.append(clean_essays[k])
            elif lables[k] == 5:
                cluster6.append(clean_essays[k])
```

```
print("Number of data points of essays in cluster 1 :",len(cluster1))
print("Number of data points of essays in cluster 2 :",len(cluster2))
print("Number of data points of essays in cluster 3 :",len(cluster3))
print("Number of data points of essays in cluster 4 :",len(cluster4))
print("Number of data points of essays in cluster 5 :",len(cluster5))
print("Number of data points of essays in cluster 6 :",len(cluster6))
```

13 Plotting Word Cloud for each cluster of Kmeans

In [ ]:
```
# for cluster 1,2,3,4 & 5 respectively

from wordcloud import WordCloud
essays_wc = [cluster1 ,cluster2,cluster3,cluster4,cluster5,cluster6]
print("Word Cloud for clusters:")
for j in range(0,6):
    word=str(essays_wc[j])
    wordcloud4 = WordCloud(width = 800, height = 800,
                background_color ='black').generate(word)

    # plot the WordCloud image
    plt.figure(figsize = (5, 5), facecolor = None)

    plt.imshow(wordcloud4)
    plt.axis("off")
    plt.tight_layout(pad = 0)

    plt.show()
```