COMP472 – Project 4 Report

Rhina Kim – 40130779

December 12, 2022

# Overview

The purpose of this project is to experiment with crawling and scraping Concordia website https://www.concordia.ca. After the clustering of websites using k-means as clustering algorithm the project then analyzes each cluster’s sentiment score using Afinn sentiment lexicon. This report will discuss the followings:

1. Choice of crawling tools and any other dependencies used & how it is implemented
2. Assess two results of the clusters brought by k-3 and k-6
3. Analyze behaviors of two result cluster collections with k-3 and k-6
4. Assess sentiment values for each cluster and derive own formula
5. Analyze the usefulness of sentiment values

# Technologies and Implementations

**Language**

* Python 3.8

**Dependencies**

* *beautifulsoup4* 4.11.1
* *scipy* 1.9.3
* *afinn* 0.1
* *scikit-learn* 1.2.0
  + *TfidfVectorizer*
  + *KMeans*
* *reppy* 0.4.14
* *urllib3* 1.26.13

Python 3.8 is used as programming language for this project since it supports any machine learning features from *scikit-learn* package and all the newest version of packages necessary for this project. This report only listed necessary packages for this project, yet all the other packages used are found in ***requirements.txt*** inside */P4* folder.

First of all, *reppy* package was used to look up and fetch robots.txt file for given URL. *Urllib* package is used to parse the URL as an input in order to find robots.txt file. *Spidy* was used as a crawling tool for this project since it comes incorporated with various set of user input and detailed loggings and is extensively configurable. Moreover, it has support for numerous useful features such as multi-threading or robot exclusion. Robot exclusion is also done by part of *Spidy* when RESPECT\_ROBOTS is set to True (see more details in *DEMO*.pdf file). However, part of Robot exclusion step was badly implemented (failure to look up robots.txt for most of the URL given during the execution), which I made another class RobotsFetcher() inside P4/*crawl.py* that replaces the given URL path with robots.txt to create path for Robots file in order to look up appropriate robots.txt file for given URL, prior to analyzing the content of Robots text. *BeautifulSoup4* was used to extract the text of the crawled and downloaded html pages, which is implemented in get\_text\_from\_pages() from *main.py* (see more details in *DEMO*.pdf file). After collecting the text from the web pages, this project then vectorizes every crawled article in order to represent them as a numeric vector, which was done via *TfidfVectorizer* method inside *scikit-learn* packages to have words ratio. The *TfidfVectorizer* method comes with preprocessing of the text before forming an inverted index, such as lowercasing, removing stop words, and other preprocessing. *KMeans* was used to clusterize the tfidf numeric vectors, which is implemented in cluster\_collection() from *main.py* (see more details in *DEMO*.pdf file). Finally, afinn was used to determine sentiment values for the each two set of clusters experiment (k=3 and k=6), which is implemented in compute\_afinn\_score() in *main.py* (see more details in *DEMO*.pdf file).

Although Spidy was used to crawl Concordia website, I have modified some codes inside the existing Spidy crawling script to adapt to this project’s purpose. Refer to *DEMO*.pdf for further details.

Additionally, to crawl Concordia website, some configurations have to be applied (e.g., maximum html files to download, number of threads to crawl with, etc.). The crawling config file came with *Spidy* package which resides under *P4/config*/ folder, and I have made additional config file that is suitable for this project’s purpose, which is in *P4/config*/***concordia.cfg*** file. Refer to *P4/config*/***blank.cfg*** for the description of every config input variable.

# Clusters Naming

One of the examples of interesting output with maximum document to download = 1000 was:

```

**CLUSTERING THE EXTRACTED TEXT WITH K=3**

Number of elements assigned to each cluster: **[124 19 3]**

Top 20 terms for cluster 0 with k = 3: school concordia science services academic calendar gina cody computer engineering research news graduate student university arts events colleges schools class

AFINN score for cluster 0 with k = 3 and top 20 terms: 0.0

Top 20 terms for cluster 1 with k = 3: concordia school calendar event science services campus graduate academic student arts schools class university colleges students news research cody gina

AFINN score for cluster 1 with k = 3 and top 20 terms: 0.0

Top 20 terms for cluster 2 with k = 3: concordiaâ concordia sustainable research says national glatard school director institute cyber university science calendar consortium smart graduate development future campus

AFINN score for cluster 2 with k = 3 and top 20 terms: 3.0

**CLUSTERING THE EXTRACTED TEXT WITH K=6**

Number of elements asigned to each cluster: **[124 16 1 3 1 1]**

Top 20 terms for cluster 0 with k = 6: school concordia science services academic calendar gina cody computer engineering research news graduate student university arts events colleges schools class

AFINN score for cluster 0 with k = 6 and top 20 terms: 0.0

Top 20 terms for cluster 1 with k = 6: concordia school calendar event science services campus graduate student academic arts university schools students class colleges news research gina cody

AFINN score for cluster 1 with k = 6 and top 20 terms: 0.0

Top 20 terms for cluster 2 with k = 6: test testing isolation execution production method risk cloud propose case methods activities software interferences environment fault event solution school providers

AFINN score for cluster 2 with k = 6 and top 20 terms: -1.0

Top 20 terms for cluster 3 with k = 6: concordiaâ concordia sustainable research says national glatard school director institute cyber university science calendar consortium smart graduate development future campus

AFINN score for cluster 3 with k = 6 and top 20 terms: 3.0

Top 20 terms for cluster 4 with k = 6: noma comp access multiple ris networks user fd techniques ceus bss cell integration power communication performance compared investigate literature spectral

AFINN score for cluster 4 with k = 6 and top 20 terms: 0.0

Top 20 terms for cluster 5 with k = 6: founded ba boutique based manette perfect retail 99 mary coffee concordia school suitablee fit pet holiday jordan comics seafood world

AFINN score for cluster 5 with k = 6 and top 20 terms: 4.0

```

Before assessing the result clusters, there is one important point to be noted: Here, the project used top 20 terms for each cluster as a display and **APPLIED** that top 20 terms to compute Afinn score for the purpose of easily highlighting terms that affected the Afinn score, instead of applying Afinn score to every term in each clusters). Refer to the analysis of Afinn score in section (4).

We can see with the above example that most of the articles resides in cluster 0 (826 articles assigned as cluster 0 for k=3 and 881 articles assigned as cluster 1 for k=6), and from the 20 terms displayed, we can guess that the topic/theme of this Concordia website is related to “school” in general.

The naming I would give to each cluster are:

k=3:

Cluster 0: Concordia Engineering department

Cluster 1: Concordia events and services

Cluster 2: researches

k=6:

Cluster 0: Concordia Engineering department

Cluster 1: Concordia events and services

Cluster 2: economy and investment

Cluster 3: urban city/community solutions

Cluster 4: ?

Cluster 5: artificial intelligence research

# Assess Clusters’ Behavior

In the k-means clustering algorithm, the value of k represents the number of clusters that the algorithm will attempt to identify within the data. As the value of k increases, the algorithm will attempt to divide the data into more clusters, which means that the clusters will be more specific and potentially more accurate in terms of capturing the inherent structure of the data.

From the above example, cluster 0 for both k=3 and k=6 seemed to have identical theme as many articles were distributed into this category.

However, increasing the value of k can also increase the risk of overfitting, where the clusters become too specific and do not generalize well to new data.

It is generally recommended to choose a value for k that is appropriate for the size and complexity of the dataset being analyzed. This can often be determined through trial and error, by trying different values of k and evaluating the resulting clusters using metrics such as the silhouette coefficient or the Calinski-Harabasz index.