***Web Science Coursework Report***

*Alana Grant 22390384, heather cook 2375346*

[*https://github.com/hdunlop310/Web-Sci-2021*](https://github.com/hdunlop310/Web-Sci-2021)

2021

**Section 1: Introduction**

**a. Describe the software developed with appropriate details; if you have used code from elsewhere please specify it**

Software used – PyCharm, MongoDB    
Code used – Josemon's starting guide code to partly process tweets, Natural Learning Toolkit for removing stopwords, Sklearn for help clustering tweets.

**b. Specify the time and duration of data collected.**

We ran code.py for 1 hour.

**Section 2: Data crawl**

**a. Use Twitter Streaming API for collecting 1% data**   
**i. Specify the APIs used**

**1. Please do not include entire code here; just main description of the function.**

We used the Tweepy streaming API that was given to us from the starting code guide provided, the code accessed twitter through the authentication keys and began to collect tweets. The tweets were gathered based on a list of keywords that we had given it, the list was made up of different words based on UK politics and the COVID impact on the UK. As the tweets were gathered by the API, each one was processed through a function called process\_tweets which helped to clean the tweet (e.g. get rid of emojis, etc) and pick out the relevant information fields. Afterwards the tweet was inserted into a collection called ‘colTest’ in MongoDB in order to help us extract data from the collection of tweets.    
We created a new python file called partOne which helped count the number of tweets, retweets and quote tweets collected. By pulling the collection and iterating through the text fields to identify when a tweet started with either “RT” or extra quotation marks.

**2. Along with a short description/justification**

We can justify using this method seeing as the sample code was already provided so it would make most sense to make use of it and just tweak it to our requirements. The tweets we have gathered collect the appropriate data seeing as it looks for UK tweets as requested, the keywords searched for are topical and current for this climate therefore can generate a lot of data. The hashtags searched for are also topical for example we have #COVID, #vaccine and #BorisHasFailedTheNation which are hashtags that have been very popular recently so will also aid with the Twitter scraping.

**Describe the seed crawl data used – (some given in the sample code) Users, hashtags, words, location etc.**

Data collected:

|  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **Total** | **Streaming API** | **No of retweets** | **No of quotes** | **No of images** | **No of videos** | **No of verified** | **No of geo-tagged** | **No of place/location objects** |
| 179,991 | ?? | 42773 | 207 | 41208 | 115 | 2039 | 57694 | 111990 |

**b. Specify data grouping methods and associated statistics Similarity measure used - briefly explain the method.**   
**Grouping strategy: explain the method. You may use pseudo code if it is convenient.**

For the second part we decided to create combine some of the sample code with our own code in order to have only geo-enabled tweets, ensuring it was just UK tweets that our stream was gathering.

Our method for grouping similar data was through K-means, which is an algorithm that attempts to group similar data together in the form of clusters, the K number represents the number of groups.

The first step was to represent each tweet as a numerical vector, we used the Sklearn’s TfidfVectorizer function to easily compute the calculations. Once this was done, we were able to plot each vector representation on a graph in order to determine the appropriate number of clusters this was done using the Elbow method. (snip in a photo of the graph)

From the graph there was a bit of a decrease/dent around n, so we attempted to group the data into n clusters. (snip in a photo of clustering results)

Once the clusters had been created, we were able to put the tweets into a dictionary and insert them into individual collections within the database.

Data collected:

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Total** | **Formed groups** | **Min size** | **Max size** | **Average size** |
|  |  |  |  |  |

**Discuss the effectiveness of similarity, grouping strategy, nature of Twitter data etc.**

Due to the nature of tweets, they contain a lot of redundant words that are not needed when trying to analysis data. To reduce the effects of these words when clustering. We implemented the NLTK which has a list of common English stop words, by doing this the clusters became more accurate and more concise.

Due to only gathering Geo enabled tweets, this dramatically reduced the number of live tweets we gathered because of this we increased the number of keywords and hashtags being searched for. The coursework requested that it be only UK tweets gathered so this was the most accurate way of determining each tweet was from somewhere in the UK.

K-means as a measure of similarity has its advantages, its very simple to implement, its scalable and fast for large datasets which came in handy for our data and it’s visual representation of the K-mean through the Elbow method made it easier to visualise the dataset at hand and how best to move forward. However, there are some down sides to using K-means, it can be sensitive to outliers and manually choosing the K-means can be difficult and lead to human error.

Grouping the tweets into different collections, was the most logical approach from the given K-means clusters. It was easy to pull the required data fields and place them into individual collections. This allowed us to see each of the cluster’s data easily and identify the different grouped topics in each one. In each collection the data was stored as a dictionary, this was so the data would be well structured, and the information would be clear/easy to get to. Due to the dictionary key being the name of the different fields of a tweet, it made analysing the collections even quicker and this also helps for easy look up of data.

Enhanced streaming ideas:

Reduce tweet noise by adding more removing more stops words like punctuation (e.g., !, ?, etc)

Reduce the number of clusters.

Reduce number of keywords asking for, so focus either on COVID or politics, this would reduce number of tweets coming in, but we can just mention that in the report I figured.

Look at specific people on twitter (e.g., Boris, Trump, Biden)

Graphing geographical information

Master doc of all cities in the UK: <https://simplemaps.com/data/world-cities>. (have to give credit)

Other list <https://www.townslist.co.uk/>

March21New (from code.py)

Cluster0

quote tweets = 2508

retweets = 0

geotagged = 40231

location objects = 78956

verified = 1820

videos = 88

images = 33203

Cluster1

quote tweets = 93

retweets = 0

geotagged = 2555

location objects = 5587

verified = 46

videos = 0

images = 863

Cluster2

quote tweets = 34

retweets = 0

geotagged = 13025

location objects = 24521

verified = 159

videos = 27

images = 7141

Cluster3

quote tweets = 0

retweets = 0

geotagged = 796

location objects = 1103

verified = 2

videos = 0

images = 0

Cluster4

quote tweets = 0

retweets = 0

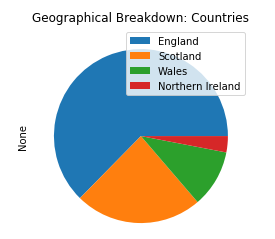
geotagged = 1087

location objects = 1823

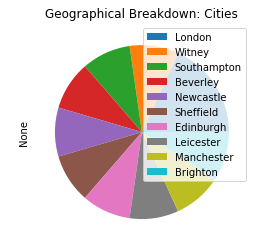
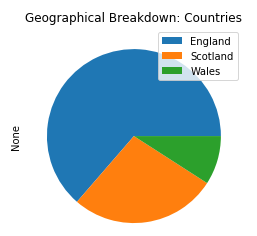
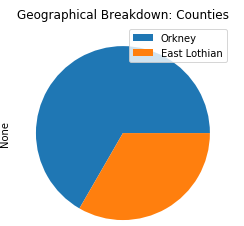
verified = 12

videos = 0

images = 1



Cluster 0 Enhanced



No data for towns

Cluster 1 Enhanced

