# WEB SOCIAL MEDIA ANALYTICS AND VISUALIZATION

Twitter Sentiment Analysis, Community Detection and News Scrapping Analysis

**Maryann Inimfon Atakpa** - 20158627

Big Data Analytics
Birmingham City University
10th May 2021

# Contents

# 1 Twitter Statistical And Sentiment Analysis

## 1.1 Introduction

Social media platforms such as Twitter, Facebook, and Youtube are used to share our life experiences, spark ideas, and express our opinions in a free and open manner. As a result, businesses want to know what people think and feel about their goods and services. They're adding social media data extraction, understanding, and analysis to their business applications.

People have taken to social media to express themselves and communicate about the protest. It's important to look at issues like "#The police" and "#Kill the Bill" that are being discussed on social media sites. Analyzing this data will assist policymakers in determining what people need.

In this report, I will get the most popular Twitter trends in the UNITED KINGDOM and I will also use sentiment analysis to show users' opinions on #KillTheBill trend, and extract some insights from this trend, and I will present my findings using statistical descriptive methods.

## 1.2 Description Of Processes

### 1.2.1 Getting the data- Dataset

The data was collected from Twitter API (Application Programming Interface). APIs enable users to send a request for a specific resource, such as Facebook or Twitter, and receive some data in response. In this project, the API used to obtain twitter data is "Tweepy". I used this library to obtain 2500 tweets from the UK related to "#KillTheBill" and "police" search keywords between 1st march 2021 and 6th april 2021. I extracted text and metadata ( user profile name, location, mobile, device type, verified and time stamp)

It's easier to store tweets in a Pandas DataFrame if you want to analyze them at scale. This enables us to apply analysis methods to multiple rows and columns. To extract the text and meta data, I accessed the Twitter data using my Twitter Api credentials, then filtered and stored the search tweets "#KillTheBill" and "Police" in a dataframe. The figure below shows a snippet of the data extracted.

| | text | user | source | location | verified | created_at |
|---|---|---|---|---|---|---|
| 0 | During series # KillTheBill raid Bristol , pla... | netpol | Android | Britain | True | 2021-04-04 08:12:57 |
| 1 | Arrests made protester scuffle police `` Kill ... | Ruptly | Media Studio | Berlin, Germany | True | 2021-04-04 15:00:00 |
| 2 | What saw Bristol confirms Police , Crime , Sen... | labourlewis | Web App | Babylon 5, Brown Sector | True | 2021-03-29 19:49:57 |
| 3 | RT @ Muqadaam : Disgraceful scene , police ash... | GeoffSoltau | Android | The Shire | False | 2021-04-05 23:55:37 |
| 4 | RT @ Philsloers : First rule Undercover Police... | nickrpd | Android | England, Europe | False | 2021-04-05 23:52:57 |
| 5 | RT @ YourAnonNews : The police protect u , pro... | ChalecosAmarill | Android | | False | 2021-04-05 23:43:53 |
| 6 | RT @ netpol : During series # KillTheBill raid... | LadyHaloJones | Web App | London | False | 2021-04-05 23:40:33 |
| 7 | RT @ SistersUncut : Yesterday saw horrendous s... | gravedoggg | iPhone | UK | False | 2021-04-05 23:35:41 |
| 8 | RT @ Wpb_Liverpool : The working class Liverpo... | bkava | Web App | | False | 2021-04-05 23:30:33 |
| 9 | RT @ Dougmcg1 : Seeing right winger suggesting... | Hetty4ScotIndy | Web App | | False | 2021-04-05 23:28:19 |
| 10 | RT @ panny_antoniou : What protest looked like... | TheNort51261705 | Android | | False | 2021-04-05 23:27:09 |
| 11 | RT @ ima_press : Article day : 'Leaked briefin... | JoyceCatanzari2 | Web App | | False | 2021-04-05 23:27:05 |
| 12 | RT @ BristolAFed : # KillTheBill Liberals : It... | blonde_done | Android | | False | 2021-04-05 23:27:04 |
| 13 | RT @ Philsloers : First rule Undercover Police... | inight16 | iPhone | London, UK | False | 2021-04-05 23:23:55 |
| 14 | RT @ netpol : During series # KillTheBill raid... | marsbrewsbeer | iPhone | | False | 2021-04-05 23:22:23 |
| 15 | RT @ JamesEFoster : Police violence brutality ... | lclabour | iPhone | Wakefield, England | False | 2021-03-29 19:49:37 |
| 16 | RT @ SunderlandUnite : Whilst police battering... | sniff91 | Android | Sunderland SR4 | False | 2021-03-29 19:48:59 |
| 17 | Thousands protest UK policing bill http : //t.... | almost_sapiens | iPad | Leicester, UK (only 2020) | False | 2021-03-29 19:48:54 |

Figure 1: Snippet of dataframe

### 1.2.2 Preprocessing The Data

The data was subjected to a number of preprocessing steps. Punctuation, stopwords, and non printable characters such as emojis were removed from tweets and also the tweets were cleaned. Furthermore, the Natural Language Toolkit Python Libraries "WordNetLemmatizer" module was used to lemmatize various types of words by converting them to their root word.

## 1.3 Data Analysis

The processed tweets were analysed and tweet sources were plotted to show the Top trends in the UK, the locations that were mostly used, the devices that were mostly used to tweet and the twitter sources that would be trusted.

### 1.3.1 Popular Trends On Twitter

Because hashtags begin with a '#,' we use a regex library for pattern matching to determine all of the hashtags and find the top Hashtags being used and determining their popularity. This is represented by a bar graph that compares the number of tweets posted for each top hashtag. The following code was used to retrieve the trends on twitter.

```
#F = api.trends_place(23424975)[0]
```

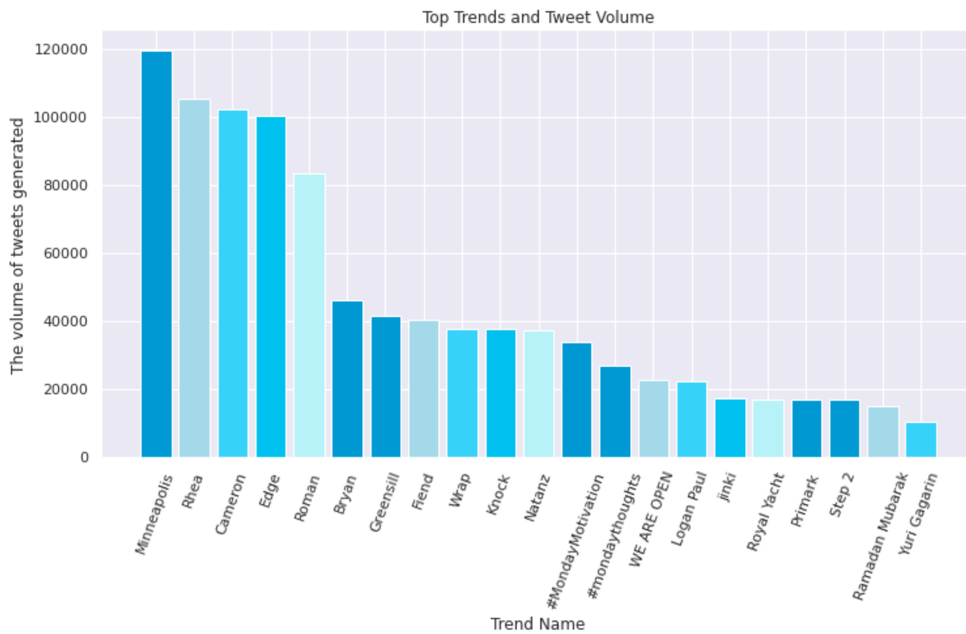


Figure 2: Top UK Twitter Trends

#### 1.3.2 What locations are used the most

I analysed the top 20 tweet locations using word frequencies and they were visualised through word cloud to identify the #KillThe-Bill trend hot zone

```
text = tweet_df.location.value_counts().head(20)
wordcloud = WordCloud(background_color='white',mode="RGB", width=1000 , height=500).generate(str(text))

plt.imshow(wordcloud)
plt.axis("off")
plt.show()
```

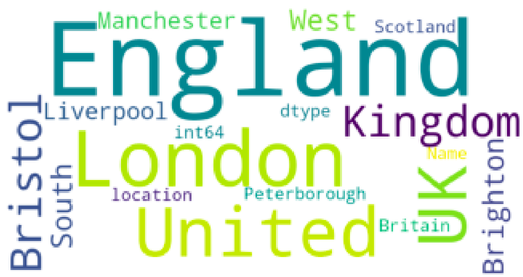


Figure 3: Tweet locations

#### 1.3.3 What Device Was Mostly Used

The processed tweets were analysed and the tweet sources were plotted to show what devices users are using to tweet the trending topic.

```
#Top 6 tweet sources
tweet_source = tweet_df.source.value_counts()

explode = (0.2, 0.1, 0.1, 0.1,
    0.1,0.1,0.1,0.1,0.1,0.1)
df2 = tweet_source[:6].plot(kind = 'pie',
    autopct='%1.0f%%', pctdistance=1.1,
    labeldistance=1.2, radius=2)
```

In the pie chart, it shows that users tweet more with an android device (39%), followed by the web app (29%), then iphone (24%), ipad (5%), tweetdeck (1%), echofon (0%). Basically this shows that most users on twitter tweet with their mobile devices.

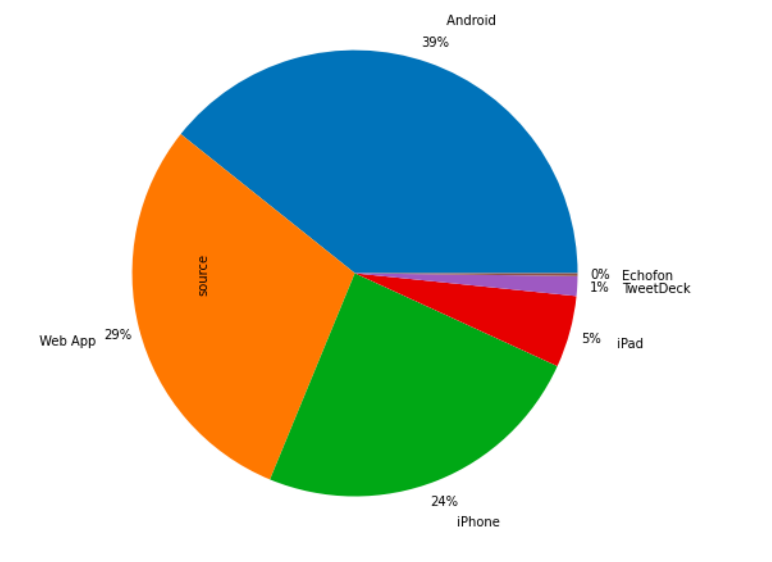


Figure 4: Pie Chart of Devices used for the Trend

#### 1.3.4 What Sources Can Be Trusted

The processed tweets were analysed to show sources you can trust. This was done by analysing and differentiating tweets that came from verified and unverified sources.

```
#verification = tweet_df.verified.value_counts
    ()

explode = (0.2, 0.1, 0.1, 0.1,
    0.1,0.1,0.1,0.1,0.1,0.1)
df3 = verification[:10].plot(kind = 'pie',
    autopct='%1.0f%%', pctdistance=1.1,
    labeldistance=1.2, radius=2)
```

My analysis on the pie chart shows that 99% of tweets were false (99% tweets were tweeted from unverified accounts) and 1% true (1 % of tweets came from verified accounts).

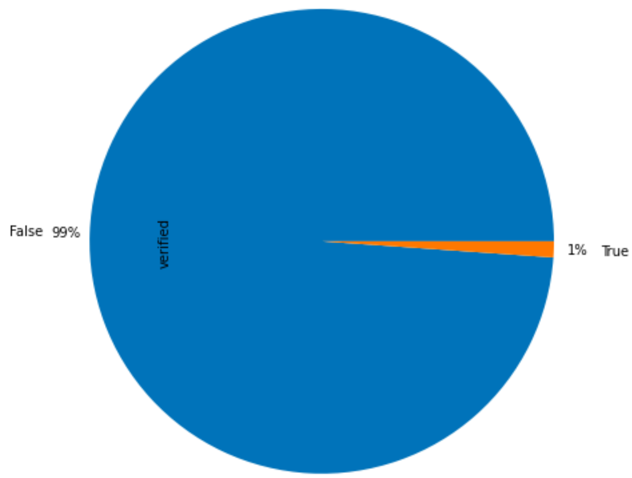


Figure 5: Pie chart visualizing sources you can trust

### 1.4 Sentiment Analysis of Tweets related to #Kill the Bill and Police impact on UK twitter.

Sentiment analysis involves classifying comments or opinions in text into categories such as "positive" or "negative" often with an implicit category of "neutral". Tracking what people think about various topics is a classic sentiment application. Sentiment analysis is also known as "opinion mining" or "voice of the customer" in data science and machine learning.

I will use the VADER SentimentIntensityAnalyzer included in the Natural Language Toolkit or nltk. It is useful for analyzing short documents, especially tweets. I looked for positive, neutral, and negative sentiments in tweets. Tweets with negative sentiments are more common than tweets with neutral or positive sentiments. The tweets express people's dissatisfaction with the police and how it affects them.

A bar chart is most often a very effective tool for results visualization and interpretation

```
#Plot bar chart showing the sentiment levels
uk_tweets.groupby('sentiment').size().plot(
    kind='bar')
```

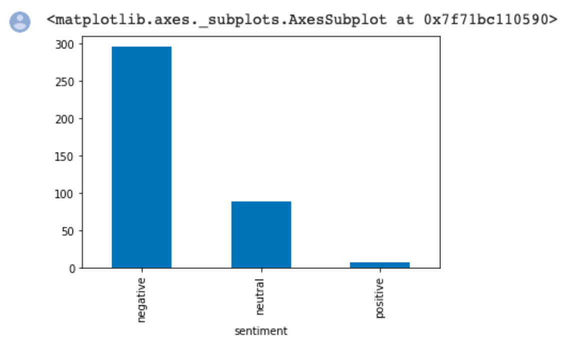


Figure 6: Bar chart representing sentiment analysis

In general, it shows overall negative affirmation in regard to our topic of the Kill The Bill and Police tweets. There are more than twice negative tweets than positive and there is a substantial amount of neutral tweets as well.

## 1.5 Limitations

There is a limit to the number of tweets that can be retrieved from Twitter's API per second, and the API does not allow users to retrieve tweets older than a week.

## 2 Social Network Analysis And Community Detection

### 2.1 Introduction

Social Network analysis is the process of investigating social structures through the use of networks and graph theory.
In this project I will use the python package, Tweepy to download twitter data from the Twitter API and NetworkX to build a network out of the data and run some analysis and use Gephi to visualize the network.

### 2.2 Data Processing and Analysis

I will use tweepy to scrape twitter for all of my followers and some of their followers. Create a pandas dataframe from all of these connections. Use NetworkX to extract a network from data and run some basic network analytics. Visualize the network in Gephi.

```
#Import libraries
import tweepy
import pandas as pd
```

```
#Twitter API credentials

consumer_key='xxxxxxxxxxxxx'
consumer_secret_key='xxxxxxxxxxxxxxxx'
access_token='xxxxxxxxxxxxxxxxx'
access_token_secret='xxxxxxxxxxxxxxxxx'
```

Since i would be downloading big datasets it's important to specify some parameters when i initialize the API. I'll set 'wait_on_rate_limit' and wait_on_rate_limit_notify to true. By setting these parameters to True, i won't break the connection to the API when i hit these limits.

```
auth = tweepy.OAuthHandler(consumer_key,
   consumer_secret_key)
auth.set_access_token(access_token,
   access_token_secret)
```

```
api = tweepy.API(auth, wait_on_rate_limit=True, wait_on_rate_limit_notify=True, compression=True)
```

```
#my twitter ID

me = api.get_user(screen_name = 'the_annea')
me.id
```

My twitter ID is; 1404718962

A network consists of nodes and links. For this network I will use individual user accounts as nodes and followers as links. The following code creates a list of my 776 followers.

```
user_list = ["1404718962"]
follower_list = [ ]
for user in user_list:
    followers = [ ]
    try:
        for page in tweepy.Cursor(api.followers_ids, user_id=user).pages():
            followers.extend(page)
            print(len(followers))
    except tweepy.TweepError:
        print("error")
        continue
    follower_list.append(followers)
```

Then i will put all my followers in a dataframe

```
df = pd.DataFrame(columns=['source','target']) #Empty DataFrame
df['target'] = follower_list[0] #Set the list of followers as the target column
df['source'] = 1404718962 #Set my user ID as the source
```

To visualise this simple network, i will use the NetworkX package to convert the data frame to a graph or network

```
import networkx as nx
G = nx.from_pandas_edgelist(df, 'source', '
    target') #Turn df into graph
pos = nx.spring_layout(G) #specify layout for
    visual
```

Then i will plot the graph using matplotlib

```
import matplotlib.pyplot as plt
f, ax = plt.subplots(figsize=(10, 10))
plt.style.use('ggplot')
nodes = nx.draw_networkx_nodes(G, pos,
                               alpha=0.8)
nodes.set_edgecolor('k')
nx.draw_networkx_labels(G, pos, font_size=8)
nx.draw_networkx_edges(G, pos, width=1.0,
    alpha=0.2
```

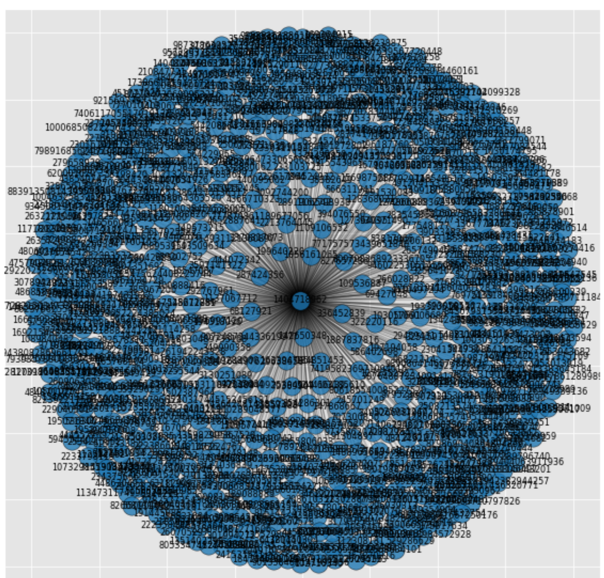

Figure 7: Network of my followers

I'm going to get all the followers of my 776 users and to achieve this i will loop through the list of all the 776 users, get their followers and add those links to the original dataframe. This code usually takes a very long time to run because of the rate limits.

There are 151396 nodes in my network. To find the most influential node in my network i used centrality measures and the most simple measure of centrality is Degree centrality, which is just a function of the number of connections each nodes has. The following code finds the number of connections each node has.

```
g_sorted = pd.DataFrame(sorted(gtf.degree,key
    = lambda x:x[1],reverse = True))

g_sorted.columns = ['names', 'degree']

g_sorted.head
```

The node in my network with the highest degree is node 1901298962 with screen name 'Naija_Pr'. The Naija_Pr has a degree of 5019 which means 5000 of this connections are the 5000 followers of this nodes that were scraped and there are 19 additional connections meaning that Naija_Pr follows 19 accounts that follow me.

I filtered the network down to a more manageable number of nodes , i will be using the k_core function of NetworkX. The k_core function filters out nodes with degree less than a given number.

```
g_r = nx.k_core(gtf, 4)
```

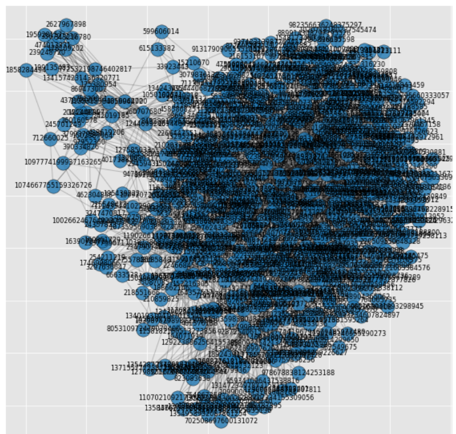

Figure 8: Undirected graph

The graph above represents an undirected graph which means the adjacency matrix is symmetrical

This graph was to reduce the nodes by 4 degree, because there were lots of nodes in the previous graph

“Community Detection is one of the key tasks in social networking analysis. It seeks to identify the number of communities in a given network (Kewalramani, 2011; Lu Halappanavar 2014)”. “The objective of Community Detection is to classify each node or vertex in the graph as belonging to the same community (Cornellisen others, 2019)” It then attempts to identify where connection exists between each community and between each node in the community.

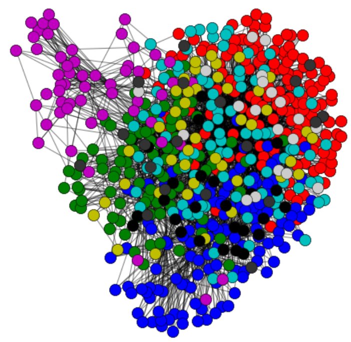

Figure 9: Network clusters

The following code is used to plot Network clusters and nodes of community graph showing different communities detected.

```
pos = nx.spring_layout(G)

f, ax = plt.subplots(figsize=(10, 10))
plt.style.use('ggplot')
nodes = nx.draw_networkx_nodes(G, pos,
                               alpha=0.8)
nodes.set_edgecolor('k')
nx.draw_networkx_labels(G, pos, font_size=8)
nx.draw_networkx_edges(G, pos, width=1.0,
   alpha=0.2)

from cdlib import algorithms, viz
import networkx as nx

coms = algorithms.louvain(G, weight = "weight"
   , resolution=1.)

viz.plot_network_clusters(G,coms,pos)
viz.plot_community_graph(G,coms)
```

“Network Visualization for communities obtained with Louvain method on the Zachary’s Karate Club graph. CDLIB allows visualizing communities on the original graph by identifying them using the same color palette for the nodes and collapsing each community in a single node to visualize the community connection graph.”

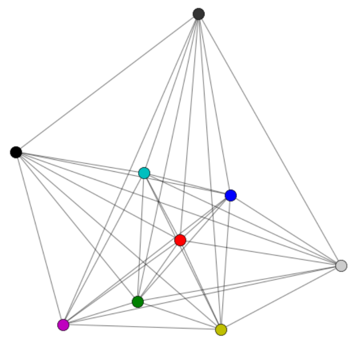

Figure 10: color coding graph for communities

The graph above is an algorithms-graph with node color coding for communities.

## 2.3 Limitations

Because of the rate limits, it was difficult to loop through all 776 accounts that follow me. So I had to run this code for days to get enough data for my study because it makes 15 API requests, then waits 15 minutes, then makes another 15 requests, and so on, which took a long time.

# 3 News Scrapping Statistical and Sentiment Analysis

## 3.1 Introduction

Web scraping includes access to and collection of data from a number of websites. This report intends to scrape news articles obtained from Newsapi.org relating to the topics of Dogecoin and cryptocurrency, after which some analysis such as sentiment analysis, topic modeling, frequency count, and Top words will be performed on the news articles.

Dogecoin is a cryptocurrency that was created to mock the rise of altcoin by Jackson Palmer and Billy Markus, turning the Internet meme doge into a currency.

Analyzing news articles on the subject would provide insight into how the rest of the world perceives dogecoin and cryptocurrency.

## 3.2 Source Of Data

The news data for this report was obtained from the developer api version of Newsapi `https://newsapi.org`. The information is obtained from news organizations such as Reuters, CNN, Reuters, and Business Insider. The data was cleaned and preprocessed to ensure that the results of our analysis were accurate. The code below was used to collect news about dogecoin and cryptocurrency from news outlets all over the world.

```
#replace with your developer key from newapi.org
secret = 'xxxxxxxxxxxxxxx'

everything_news_url = 'https://newsapi.org/v2/everything'
```

| | Title | Description | Source |
|---|---|---|---|
| 0 | Crypto market takes a dive with Bitcoin leadin... | Cryptocurrency prices continued to tumble Frid... | TechCrunch |
| 1 | Crypto trading on Robinhood spiked to 9.5M cus... | It's been a big year for crypto, and Robinhood... | TechCrunch |
| 2 | Dogecoin is mooning, and we're listening for t... | With dogecoin back in the news, the bubble pop... | Mashable |
| 3 | Dogecoin: Everything you need to know about th... | We're going to the moon. \nThe proponents of D... | Mashable |
| 4 | Move over, Bitcoin. Ethereum is at an all-time... | Bitcoin prices continued their rebound Saturda... | CNN |
| 5 | Meme Crypto Dogecoin Price Up 400% In 1 Week | Last week, the Dogecoin price spiked 400%, sho... | ValueWalk |
| 6 | DogeDay hashtags help meme-based cryptocurrenc... | Dogecoin prices hit an all-time on Tuesday, wi... | Reuters |
| 7 | Meme-based cryptocurrency Dogecoin soars 40% t... | Meme-based virtual currency Dogecoin soared on... | Reuters |
| 8 | UPDATE 1-Dogecoin cryptocurrency slumps after ... | Meme-based cryptocurrency Dogecoin fell on Tue... | Reuters |
| 9 | Dogecoin in spotlight as cryptocurrency backer... | With the price of dogecoin surging, investors ... | Reuters |
| 10 | Dogecoin Surged by More Than 38%, Reaching a R... | Dogecoin value rose over 38% on Wednesday, rea... | Entrepreneur |
| 11 | Dogecoin in spotlight as cryptocurrency backer... | With the price of dogecoin surging, investors ... | Reuters |
| 12 | Elon Musk Says Dogecoin Could Be Cryptocurrenc... | The billionaire tycoon may have sparked intens... | Entrepreneur |
| 13 | Billionaire Mike Novogratz says cryptocurrenci... | Summary List PlacementThe cryptocurrency marke... | Business Insider |

Figure 11: data frame of news articles

### 3.3 Data Processing

I used the newsapi to look for news containing the keywords "dogecoin," "cryptocurrency," "crypto" "market." The title, description, URL, and source were all extracted.

By removing symbols obtained from the news API, the data was cleaned. The data was converted from json to a pandas library dataframe object so that the analysis could be performed easily.

### 3.4 Data Analysis On News Articles

Stopwords and punctuation were removed from five Reuters articles, and a Python library called Beautiful Soup was used to extract structured data from a website. It is capable of parsing data from HTML and XML files. It acts as a helper module, interacting with HTML in a similar and improved manner to how other developer tools would interact with a web page. On each of the five articles, the following code was used.

```
import requests
import urllib.request
import time
from bs4 import BeautifulSoup
```

```
url1 = "https://www.reuters.com/article/us-crypto-currency-musk-idUSKBN2CO246"
page1 = requests.get(url1).text
# Turn page into BeautifulSoup object to access HTML tags
soup1 = BeautifulSoup(page1)

# Pares HTML for article body

# Get text from all <p> tags.
p_tags1 = sp.find_all('p')
# Get the text from each of the    p    tags and strip surrounding whitespace.
p_tags_text1 = [tag.get_text().strip() for tag in p_tags1]

#convert to string for easier manipulation
text1 = " ".join([word for word in p_tags_text1
                              if '\xa0' not in word
                 ])
```

### 3.4.1 Sentiment Analysis

The article obtained from the data source was subjected to sentiment analysis. This analysis was carried out in order to gain a better understanding of the impact of dogecoin and cryptocurrency on the world and the economy. The following code was used to get the sentiment analysis and to plot the graph.

```
 #Lets have a look at some good news in the midst of so much negative news
positive = dfx_sentiment.loc[dfx_sentiment['sentiment'] == 'neutral']
positive


#Plot bar chart showing the sentiment levels
```

```
dfx_sentiment.groupby('sentiment').size().plot
    (kind='bar')
```

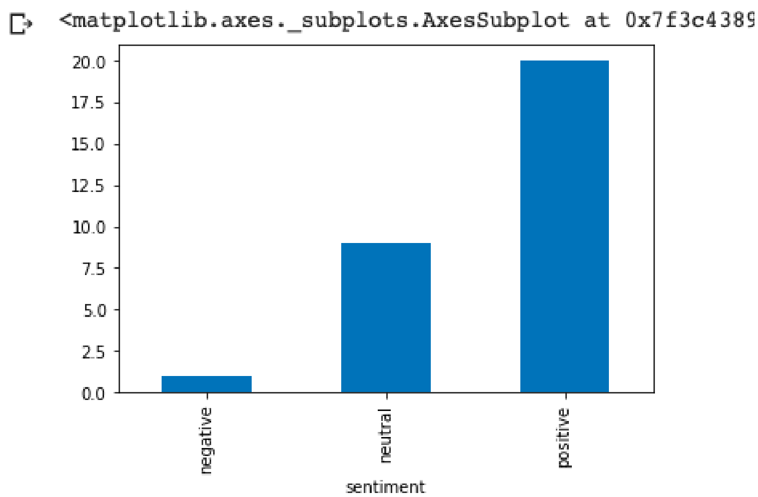


Figure 12: Sentiment analysis

The graph below depicts an overview of the sentiment in the news articles I obtained, which, as expected, were mostly positive.

### 3.4.2 Topic Modelling

The topic modeling technique was also used to find hidden topics in large amounts of text. For topic modeling, I used the LDA algorithm from the Python SKlearn package. Latent Dirichlet Allocation (LDA) is a popular algorithm for topic modeling with excellent implementations in Python's Gensim package.The following code was used on five news articles.

```python
import sklearn;
from sklearn.feature_extraction.text import CountVectorizer, TfidfVectorizer;
from sklearn.decomposition import LatentDirichletAllocation

#display topics
def display_topics(model, feature_names, no_top_words):
    for topic_idx, topic in enumerate(model.components_):
      print ("Topic", topic_idx)
      print (" ".join([feature_names[i]
         for i in topic.argsort()[:-no_top_words - 1:-1]]))

# LDA is able to use tf-idf
no_features = 5000
tfidf_vectorizer = TfidfVectorizer(max_df=0.50, min_df=1, max_features=no_features, stop_words='english')
tfidf = tfidf_vectorizer.fit_transform(filtered_n1, y=None)
tfidf_feature_names = tfidf_vectorizer.get_feature_names()


#Initialize the number of Topics we need to cluster:
num_topics = 10;

```

```

lda = LatentDirichletAllocation(n_components=num_topics, max_iter=5, learning_method='online', learning_offset=50.,random_state=0).fit(tfidf)





no_top_words = 6
display_topics(lda, tfidf_feature_names, no_top_words)
```

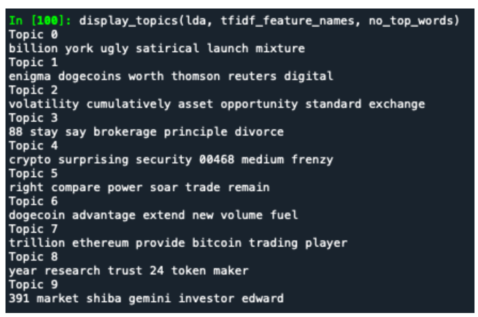


Figure 13: Topic modelling snippet from one of the articles

### 3.4.3 Word Count and Top Words

Each article's text was lemmatized, tokenized, and stop words were removed in order to perform a basic descriptive analysis that displayed the count of each word and top words using a frequency bar chart and word cloud graph. To accomplish this, the following code was used.

```
from pywsd.utils import lemmatize, lemmatize_sentence
nltk.download('punkt')
nltk.download('averaged_perceptron_tagger')
nltk.download('wordnet')
nltk.download('stopwords')
stop_words = set(stopwords.words("english"))
n1 =n1.lower()




#lemmatize and tokenize the words
ln1 = lemmatize_sentence(n1)



#clean the data by eliminating stopwords
filtered_n1=[w for w in ln1 if not w in stop_words]
freq_ln1 =nltk.FreqDist(filtered_n1)



#Eliminating words with three characters and below
large_ln1=dict([(k,v) for k,v in freq_ln1.items() if len(k) >3])


freq_ln1.plot(30, cumulative= False)

```

```
n1_cloud= WordCloud(max_font_size =50, max_words=100, background_color="white").generate_from_frequencies(freq_ln1)

plt.figure()
plt.imshow(n1_cloud, interpolation ="bilinear")
plt.axis("off")
plt.show()
```

Article 1

The following Reuters Url was extracted from the data frame `https://www.reuters.com/article/us-crypto-currency-musk-idUSKBN2CO246` and the descriptive analysis was carried out. The figures below represents the visualization of the analysis

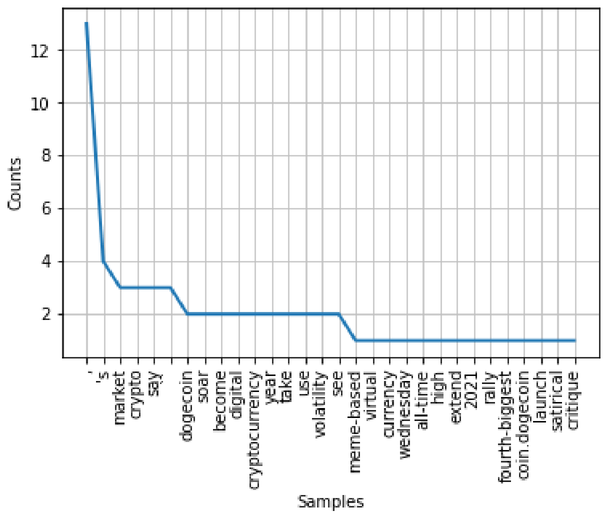


Figure 14: Frequency plot for Article 1


Figure 15: Top words for article 1

Article 2

The following Url was extracted from the data frame `https://www.reuters.com/article/us-crypto-currency-musk-idUSKBN2CO246` and the descriptive analysis was carried out. The figures below represents the visualization of the analysis

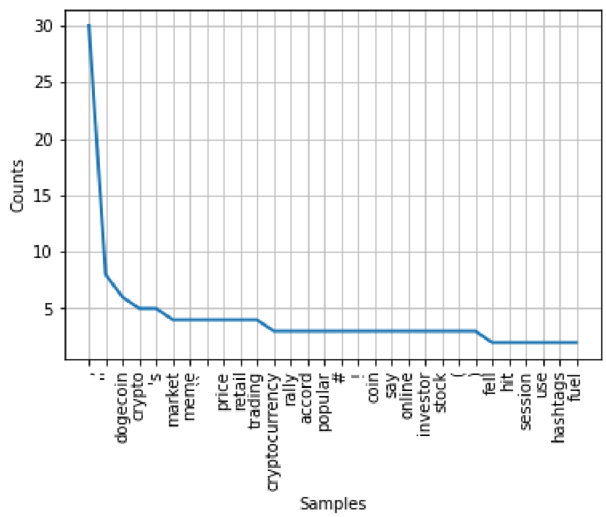


Figure 16: Frequency plot for Article 2


Figure 17: Top words for article 2

Article 3

The following Url was extracted from the data frame `https://www.reuters.com/article/us-crypto-currency-musk-copy-idUKKBN2C0271`and the descriptive analysis was carried out. The figures below represents the visualization of the analysis

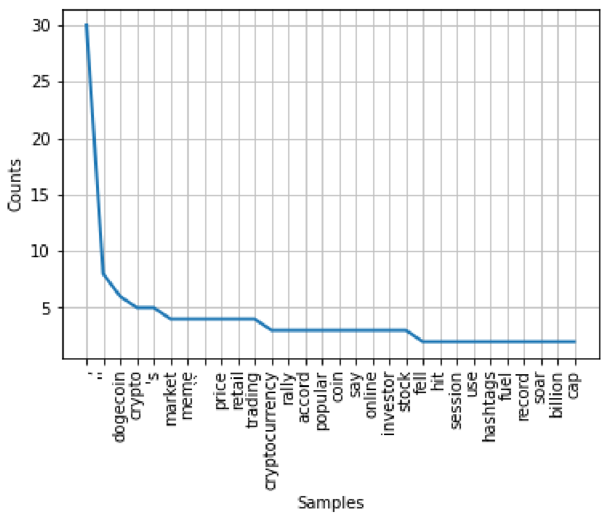


Figure 18: Frequency plot for article 3


Figure 19: Top words for article 3

Article 4 The following Url was extracted from the data frame `https://tinyurl.com/57pasfkp` and the descriptive analysis was carried out. The figures below represents the visualization of the analysis

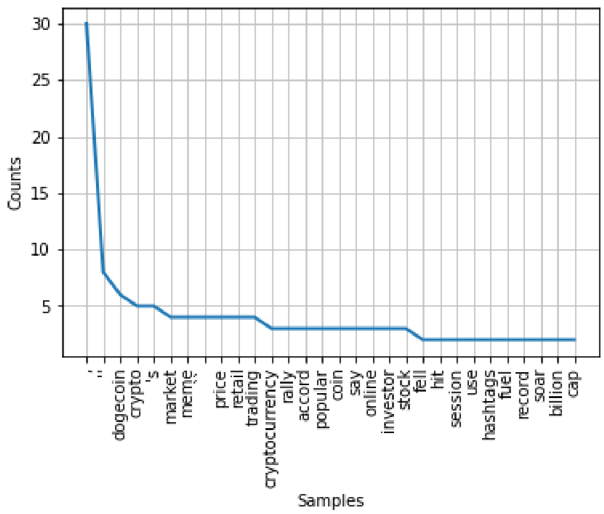


Figure 20: Frequency plot for article 4


Figure 21: Top words for article 4

Article 5 The following Url was extracted from the data frame `https://tinyurl.com/48xu2ex3` and the descriptive analysis was carried out. The figures below represents the visualization of the analysis.

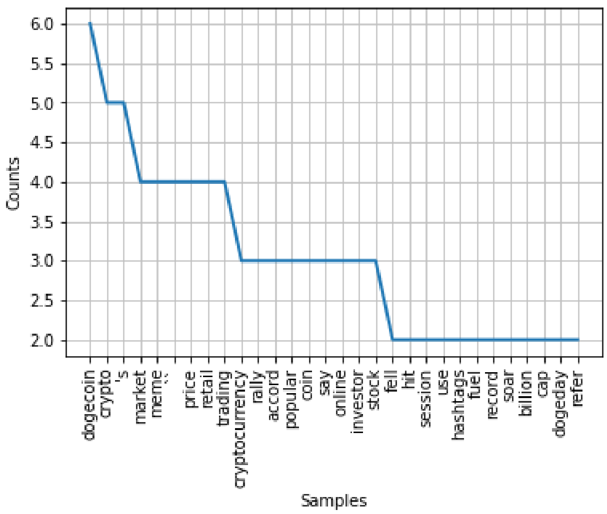


Figure 22: Frequency plot for Article 5


Figure 23: Top words for article 5

#### 3.4.4 Article Summarization

The task of creating a short, accurate, and fluent summary of an article is known as text summarization. An article was chosen from among those previously obtained from newsapi. The articles' content was scraped using a Python library called 'beautiful soup.' Each article was shortened to 300 words using another summarizer library from "genism."

The output of the summary was just as clear as the original article, and I was able to gain a better understanding of what the articles were discussing. I tried reducing the number of words to 200 and discovered that it was still quite understandable. The code below was used to extract the summary of a text from an article.

```
n1

#summarize text

from gensim.summarization.summarizer import
    summarize
print(summarize(n1, word_count= 300))
```

Below is the link of the full article https://www.reuters.com/article/us-crypto-currency-musk-idUSKBN2CO246 and a snippet of the summarized article

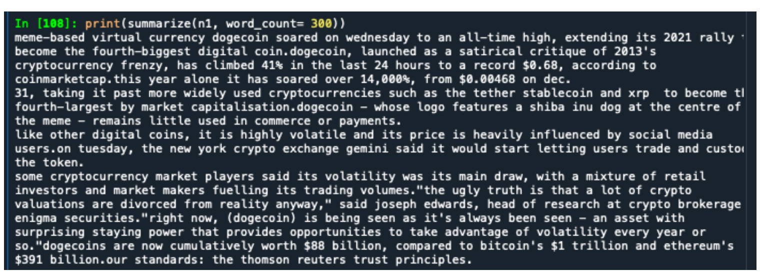


Figure 24: Article summarization

The summary's content was excellent; it captured all of the relevant details within the word count allotted, and it could be read and understood by everyone.

## 3.5 Limitations

It was difficult to remove some punctuation as stop words from each of the articles because they were not visible.

# 4 Summary and Conclusion

This report is divided into three sections that show how to extract, process, and analyze social media data, as well as how to visualize the results using bar charts, graphs, and so on. Some limitations to some of the methods are also mentioned.

The first section of this report discusses how Twitter API was used to collect tweets on the trending topic #KillTheBill for this paper. The Twitter API was used to collect and clean the data, as well as to retrieve information and data such as the devices used, the locations from which the majority of the tweets originated, and reliable sources for the topics #KillTheBill and "Police," with the results visualized using pie charts and bar graphs. In this phase of the report, sentiment analysis was used to examine the public's reaction to this subject on Twitter, and the results revealed that the majority of people were dissatisfied with the police, with the majority of tweets being negative.

The second section of this study covered social network analysis and community detection. My personal Twitter account was used to analyze the important nodes in my network using a centrality measure, and the results of my study were visualized using the degree centrality measure. My graph was subjected to a community detection algorithm in order to visualize the relationship that exists between my Twitter communities. On the Zachary's Karate Club graph, the network was visualized using the Louvain method.

The last section of this report discussed news scraping, topic modeling, summarization, and sentiment analysis on news articles. The information was obtained and cleaned from the NewsApi website. The sentiment analysis results show an overall positive response on the selected topics, frequency counts, and top words were visualized using a bar chart and a frequency graph. Topic modeling was also performed on five articles, and one of the articles was summarized using the genism library.

# 5 References

Rossetti, G., Milli, L. and Cazabet, R., 2019. CDLIB: a python library to extract, compare and evaluate communities from complex networks. Applied Network Science, 4(hal-02197272), p.52.

Lu, H., Halappanavar, M. and Kalyanaraman, A., 2015. Parallel heuristics for scalable community detection. Parallel Computing, 47(0167-8191), pp.19 - 37.

Kewalramani, M., 2011. Community Detection in Twitter.

Cornelissen, L., de Bruyn, C., Ledingwane, M., Theron, P., Schoonwinkel, P. and Barnett, R., 2019. Cross-Sample Community Detection and Sentiment Analysis. Proceedings of the South African Institute of Computer Scientists and Information Technologists 2019 on ZZZ - SAICSIT '19,.

En.wikipedia.org. 2021. Dogecoin - Wikipedia. [online] Available at: https://en.wikipedia.org/wiki/Dogecoin¿ [Accessed 10 May 2021].

Earth Data Science - Earth Lab. 2021. Use Twitter Social Media Data in Python - An Introduction. [online] Available at: https://www.earthdatascience.org/courses/use-data-open-source-python/intro-to-apis/social-media-text-mining-python/¿ [Accessed 10 May 2021].

Devika, M., Sunitha, S. and Ganesh, A., 2016. Sentiment Analysis: A Comparative Study on Different Approaches. Procedia Computer Science, [online] 87(1877-0509), pp.44-49. Available at: https://www.sciencedirect.com/science/article/pii/S187705091630463X¿ [Accessed May 2016].

Asmussen, C. and Møller, C., 2019. Smart literature review: a practical topic modelling approach to exploratory literature review. Journal of Big Data, 6(1).

Rananavare, L. and Reddy, P., 2018. Automatic News Article Summarization. International Journal of Computer Sciences and

Engineering, 6(2), pp.230-237.

Python, R., 2021. Beautiful Soup: Build a Web Scraper With Python – Real Python. [online] Realpython.com. Available at: https://realpython.com/beautiful-soup-web-scraper-python/¿ [Accessed 10 May 2021].