Collection, Pre-processing and Visualization of Tweets

Syed Farhan Hassan
Artificial Intelligence & Machine Learning
Lambton College
Toronto, Canada
c0827236@mylambton.ca

Nishanth Parthasarathy
Artificial Intelligence & Machine Learning
Lambton College
Toronto, Canada
c0828612@mylambton.ca

Ravneet Kaur Ajji
Artificial Intelligence & Machine Learning
Lambton College
Toronto, Canada
c0825099@mylambton.ca

Nishanth Nimesh Patel
Artificial Intelligence & Machine Learning
Lambton College
Toronto, Canada
c0827799@mylambton.ca

Mohammed Imran
Artificial Intelligence & Machine Learning
Lambton College
Toronto, Canada
c0825366@mylambton.ca

***Abstract*—Tweets have become important source of data today for different types of analysis and study. In this assignment, we have learnt how to fetch tweets in real time using different libraries, storing fetched tweets in the provided format, pre-processing of stored textual tweets, clustering of tweets, and visual comparison of tweets. Tweets related to provided keywords related to crypto and tech giants were only part of our study. It was found that tweets related to Bitcoin was most trending among all the given keywords. Tweets related Google was more trending among the tech giants.**

1. Introduction

The objective of the assignment was to collect tweets in real time for the provided keywords, store them in the given format, clean them with different natural language processing techniques, and visualize the count of tweets and users for each keyword.

NLP is a field of artificial intelligence that deals with enabling computers with the ability to understand text and spoken words in much the same way human beings can. It can be used process textual data collected from the tweets

2. Tweets Collection

There are Python libraries available which can be used to read and fetch tweets using the twitter developer account with official twitter API.

A. Tweepy

Tweepy is a widely used, open-sourced Python library to access the Twitter API with Python. We also tried to use Tweepy to fetch tweets for the provided keywords. However, it couldn't fulfil our requirement to fetch trending keywords due to its restrictions of fetching limited number of tweets for a week. So, we used another library to overcome this restriction.

B. Snscrape

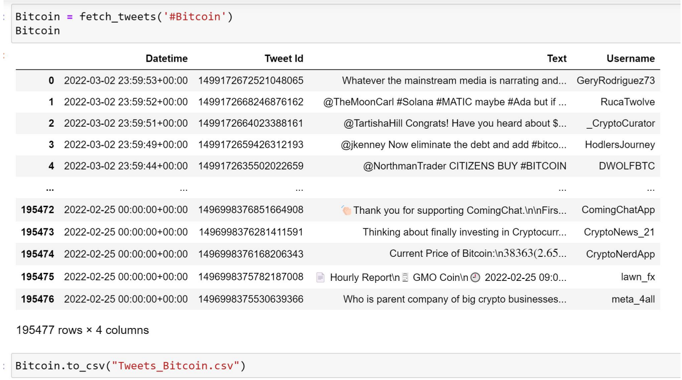
snscrape is a Python library that can be used to scrape tweets through Twitter's API without any restrictions or request limits. It really helped us to fetch large number of tweets related to trending keywords. Furthermore, we don't even need a Twitter developer account to scrape tweets using snscrape.

We fetched and stored tweets related to below keywords:

- Altcoin
- Bitcoin
- Coindesk
- Cryptocurrency
- Gold
- APPL
- GOOG
- YHOO

3. Tweets Storage

All the fetched tweets were saved into CSV files consisting of columns – Datetime, Tweet Id, Text and Username.



```
Bitcoin = fetch_tweets('#Bitcoin')
Bitcoin
```

	Datetime	Tweet Id	Text	Username
0	2022-03-02 23:59:53+00:00	1499172672521048065	Whatever the mainstream media is narrating and...	GeryRodriguez73
1	2022-03-02 23:59:52+00:00	1499172668246876162	@TheMoonCarl #Solana #MATIC maybe #Ada but if ...	RucaTwolve
2	2022-03-02 23:59:51+00:00	1499172664023388161	@TartishaHill Congrats! Have you heard about $...	_CryptoCurator
3	2022-03-02 23:59:49+00:00	1499172659426312193	@jkenney Now eliminate the debt and add #bitco...	HodlersJourney
4	2022-03-02 23:59:44+00:00	1499172635502022659	@NorthmanTrader CITIZENS BUY #BITCOIN	DWOLFBTC
...	...	...	...	...
195472	2022-02-25 00:00:00+00:00	1496998376851664908	👏 Thank you for supporting ComingChat.\n\nFirs...	ComingChatApp
195473	2022-02-25 00:00:00+00:00	1496998376281411591	Thinking about finally investing in Cryptocurr...	CryptoNews_21
195474	2022-02-25 00:00:00+00:00	1496998376168206343	Current Price of Bitcoin:\n38363(2.65...	CryptoNerdApp
195475	2022-02-25 00:00:00+00:00	1496998375782187008	📄 Hourly Report\n🗒 GMO Coin\n🕘 2022-02-25 09:0...	lawn_fx
195476	2022-02-25 00:00:00+00:00	1496998375530639366	Who is parent company of big crypto businesses...	meta_4all

195477 rows × 4 columns

```
Bitcoin.to_csv("Tweets_Bitcoin.csv")
```

Fig 1: Tweets storage into given format

4. Text Cleaning

Text cleaning is the process of preparing raw texts using different techniques of NLP, to make it suitable or compatible for downstream analysis processes. We also performed standard text cleaning on the stored tweets of each keyword.

A. *Reformatting texts to lower case*

All the texts in the tweets were transformed into lower case to bring them down to a common format, otherwise, it might create error or diminish precision in downstream analysis.

B. *Text Splitting*

Tweets contained texts in form of sentences or phrases. They were broken into individual words or textual entity by splitting them based on space between them.

C. *Reformatting Contractions*

Most of the tweets contained contractions which should be treated otherwise actual meaning of the words would be lost in downstream analysis. So, we used dictionary of mapped contractions to transform the contracted words in the tweets of each keyword. For example, *"ain't" was transformed into "am not"*

D. *Removing Stop words*

Stop words which are commonly used redundant words adding least meaning to the sentence, such as articles and prepositions – a, an, the, of, in etc. All the stop words in the tweets were removed using NLTK corpus (English) of stop words.

i. *Treating HTML elements*

Raw tweets contained many HTML elements which were non textual data, irrelevant to the downstream textual analysis. So, all such entities were removed from the tweets, using the regular expression filtering technique.

ii. *Removing https links*

All the URLs or https links present in the tweets were removed using regex filters.

iii. *Removing <href> Tags*

Sometimes, web URLs have seen to be extracting in form of <href> tags instead of https links. So, such cases were separately treated and “<a href” was removed from the texts.

re.sub(r'\<a href', ' ', text)

iv. *Removing other irrelevant HTML elements*

There were other irrelevant HTML entities, such as line break tag <br> and &amp encoding for ampersand. “<br” and “>” were being treated separate words so they had to be treated separately. Similarly, many tweets contained “&amp” encoded texts in place of & symbol, so they had to be treated as well.

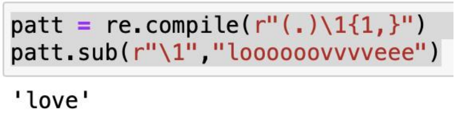
re.sub(r'&amp;', ' ', text)
re.sub(r'<br ', ' ', text)
re.sub(r'>', ' ', text)

E. *Removing Special Characters and Numbers*

Tweets also contained many special characters as well as numeric elements which are not relevant to textual comparison or textual analysis. So, all such data were removed using regex filtering and only alphabets were kept in the final textual data.

F. *Removing Word Lengthening*

Word lengthening happens, when characters are unnecessarily repeated. The maximum number of allowed repeated characters in any English words is two. If we don't remove the extra characters, we can end up with inaccurate information. So, we used regular expressions that rips off repeated characters with a length of more than 2 characters. For example, “love” word had so many repeated o’s as in below image, which we remove using regular expression.



```
patt = re.compile(r"(.)\1{1,}")
patt.sub(r"\1","loooooovvvveee")

'love'
```

Fig 2: Removing Word Lengthening

G. *Stemming and Lemmatization*

Every word has one base form but many variations; for example, "play" is a base word while "playing," "played," and "plays" are all various forms of the same word. As a result, these terms are stripped of their meanings and may contain additional inaccuracies. Stemming is the process of reducing inflections to their base forms in such a way that it depicts a collection of related words under the same stem, even if the base has no proper meaning. We used Porter Stemmer from NLTK library to perform stemming on our tweets’ texts.

Similarly, Lemmatization is a more advanced text normalization technique, which considers context and part of speech to determine the lemma i.e., root form of the words. We used WordNet Lemmatizer from NLTK library to perform lemmatization on the extracted tweets.

5. Visualization and Comparison of Tweets

After cleaning all the tweets for each keyword, tweets from each category were visualized to study the comparison between them.

A. *Altcoin Bar Plot*

For the past week, tweets related to Altcoin gradually picked up the trend. It started with around 1400 tweets and increased till 2500 tweets per day.

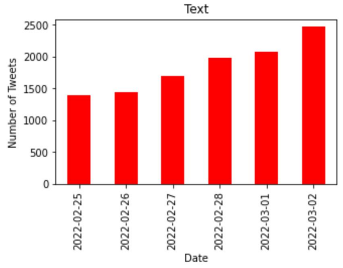


Fig 3: Altcoin Bar Plot

B. *Bitcoin Bar Plot*

Bitcoin tweets are the most trending tweets among the given keywords. In last week, the minimum no of tweets tweeted related to bitcoin was around 25,000 per day and it had maximum of around 40,000 tweets per day.

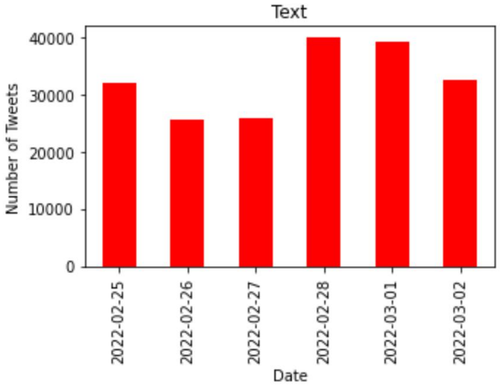


Fig 4: Bitcoin Bar Plot

C. *Cryptocurrency Bar Plot*

Like bitcoin, tweets related to cryptocurrency was a trending topic last week. It had a fluctuating trend with minimum of around 10,000 tweets per day and maximum of 14,000 tweets per day.

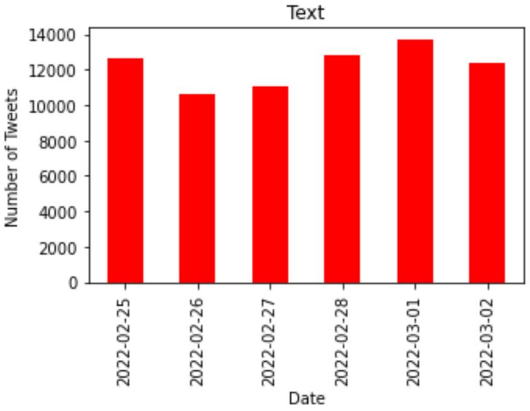


Fig 5: Cryptocurrency Bar Plot

D. *Tech Companies Tweets*

Tech companies, Apple, Google and Yahoo captured related tweets not more than 10, last week. Google and Apple had related tweets throughout the week, with minimum of 6 & 2 tweets per day and maximum of 9 & 8 tweets/day respectively. However, Yahoo had related tweets just one day with count of only one tweet on 28th Feb' 22.

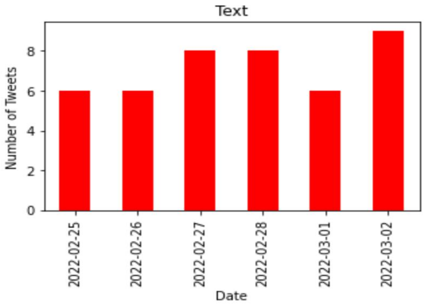


Fig 6: Tech Companies Tweets

6. VISUAL COMPARISON OF TWEETS RELATED TO DIFFERENT KEYWORDS

For the purpose of visual comparison, keywords were grouped into 2 categories:

Bitcoin, Cryptocurrency & Gold → Having tweets in thousands

CoinDesk, APPL, GOOG, YHOO → Having tweets less than 50

Otherwise, visualisations were not properly differentiable due to huge mismatch in scale, thousands versus less than 50.

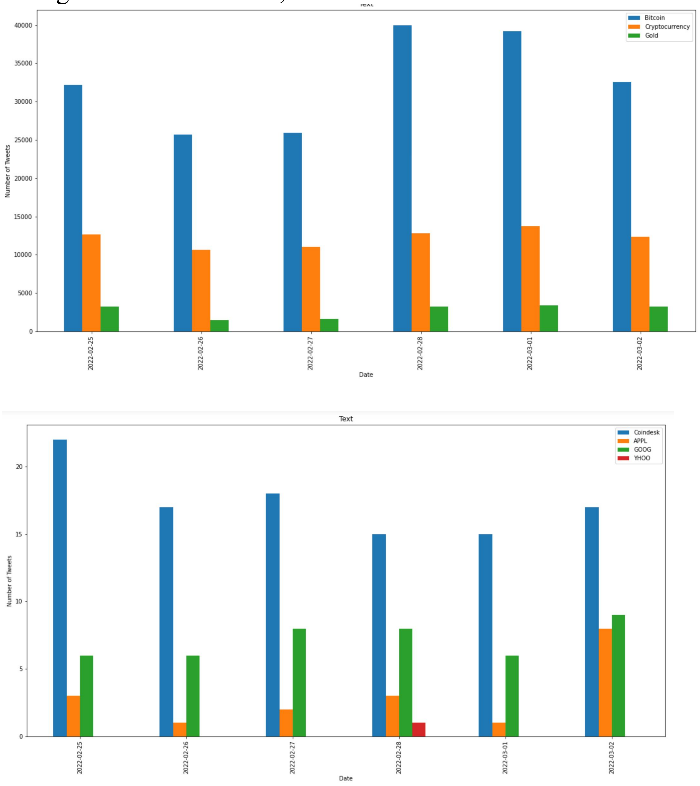


Fig 7: Bitcoin, Cryptocurrency & Gold

7. CLUSTERING OF SIMILAR TWEETS UNDER EACH CATEGORY

We tried clustering of tweets under each category of keywords. However, for some keywords, it was not feasible for keywords having humongous amount of tweets. KMeans clustering technique was used in grouping tweets.

A. *Non-Feasible Clustering*

As shown in visualization, keywords "Bitcoin" and "Cryptocurrency" had large no of related tweets, around 40,000 and 15,000 tweets/day respectively. On trying to clustering such huge no of texts under KMeans, system was unable to produce results due to shortage of memory. For Bitcoin related tweets, it required 146 GB and for Cryptocurrency related tweets, 20.3 GB of memory was required.

```
MemoryError: Unable to allocate 146. Gi
B for an array with shape (195477, 1000
39) and data type float64
```

B. *Feasible Clusters*

Apart from Bitcoin and Cryptocurrency, we got results for Kmeans clustering. Each keywords happened to be grouped into 3 separate and well segregated clusters.

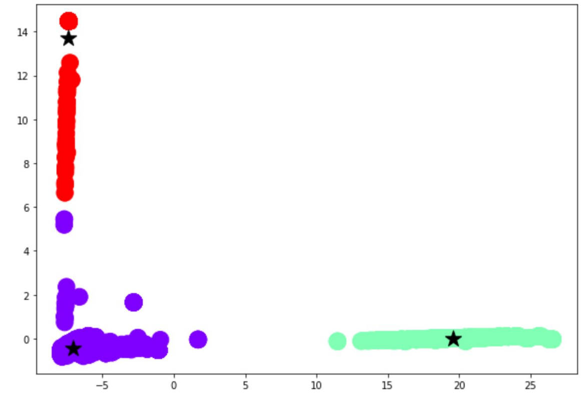
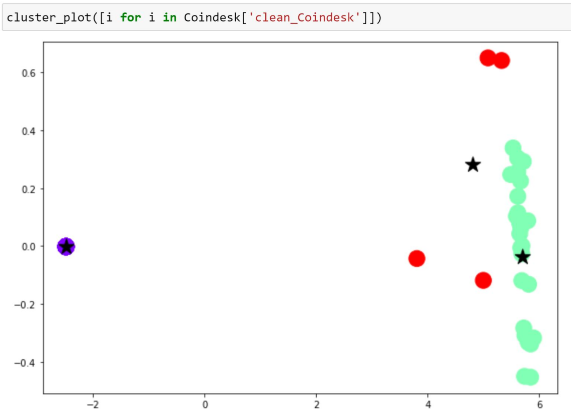


Fig 8: Clustering of Altcoin Tweets



```
cluster_plot([i for i in Coindesk['clean_Coindesk']])
```

Fig 9: Clustering of Coindesk Tweets

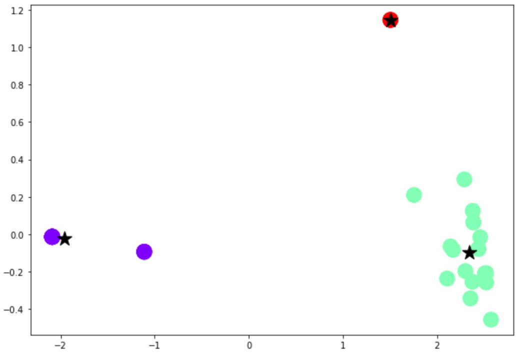


Fig 10: Clustering of Google Tweets

REFERENCES:

[1] IBM Cloud Education. (n.d.). Natural language processing. IBM. Retrieved March 15, 2022, from https://www.ibm.com/cloud/learn/natural-language-processing