Assignment 1

By

Group Falcons

Benudhar Patra (c0834626)

Mohammad Al-Ajlouni (c0849924)

Viswanatha Reddy (c0836919)

Yashwanth Paladi (c0849467)

AIMT, Lambton College

CBD 3335 - Data Mining and Analysis

Prof: Ali Nouhi

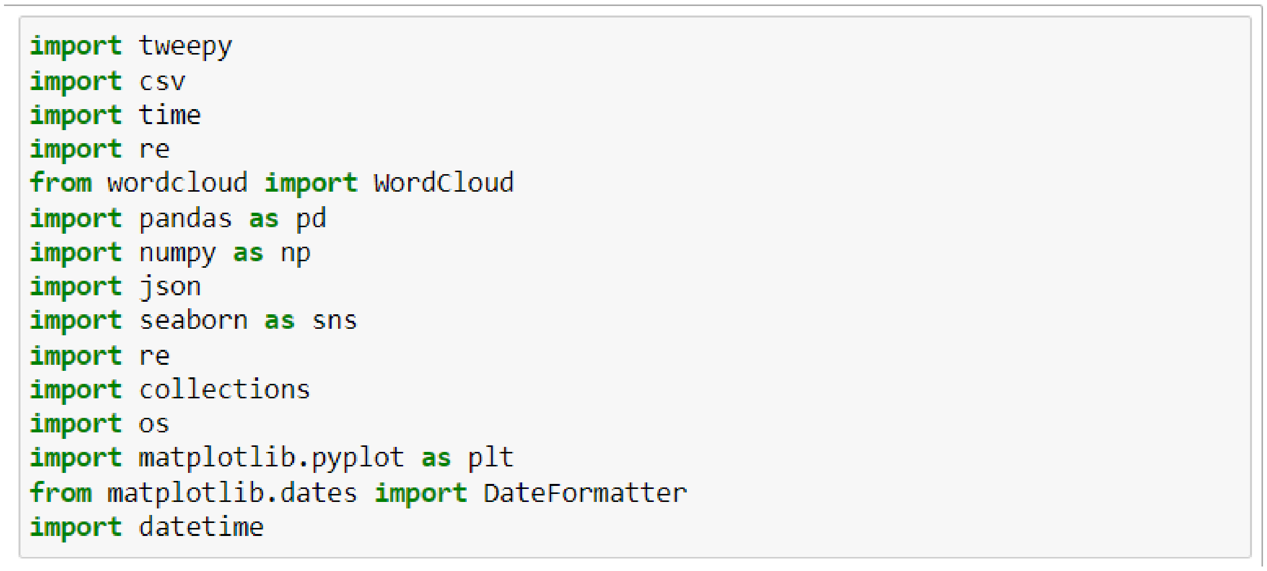
July 04, 2022

Abstract:

In this assignment, we gathered information on the stock market via Twitter from June 14, 2022 to June 20, 2022. We used eight hashtags to collect the data for each set of keywords and saved them in unique files. We have performed data cleaning and used the bar chart visualization techniques.

Steps performed:

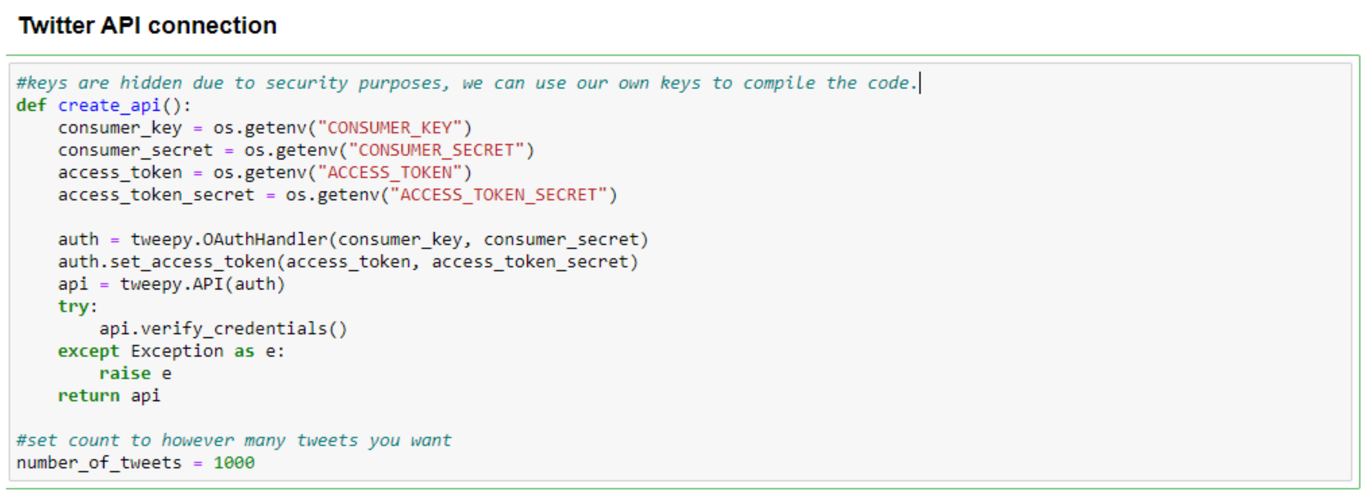
We have imported the below required libraries to perform the respective tasks.



```
import tweepy
import csv
import time
import re
from wordcloud import WordCloud
import pandas as pd
import numpy as np
import json
import seaborn as sns
import re
import collections
import os
import matplotlib.pyplot as plt
from matplotlib.dates import DateFormatter
import datetime
```

We have established a Twitter API connection to retrieve the tweets data related to the stock market for one week. Here, we have hidden the keys for consumer_key, consumer_secret, access_token and access_token_secret for security reasons, in order to compile the code we can use our own Developer account elevated access keys for compilation. We have kept a limit to collect 1000 numbers of tweets for each keyword for one week till the date specified.

Tweepy is an open-source, user-friendly Python library for accessing the Twitter API. It provides you with an interface so that your Python program may use the API access.



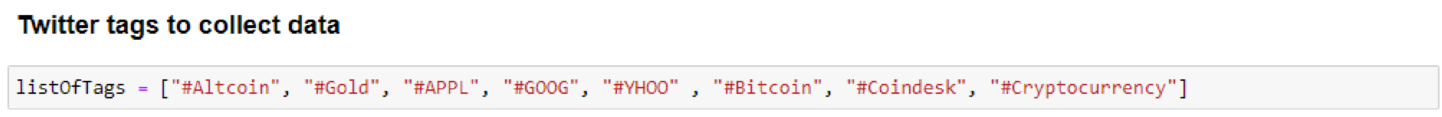
Twitter API connection

```
#keys are hidden due to security purposes, we can use our own keys to compile the code.
def create_api():
    consumer_key = os.getenv("CONSUMER_KEY")
    consumer_secret = os.getenv("CONSUMER_SECRET")
    access_token = os.getenv("ACCESS_TOKEN")
    access_token_secret = os.getenv("ACCESS_TOKEN_SECRET")

    auth = tweepy.OAuthHandler(consumer_key, consumer_secret)
    auth.set_access_token(access_token, access_token_secret)
    api = tweepy.API(auth)
    try:
        api.verify_credentials()
    except Exception as e:
        raise e
    return api

#set count to however many tweets you want
number_of_tweets = 1000
```

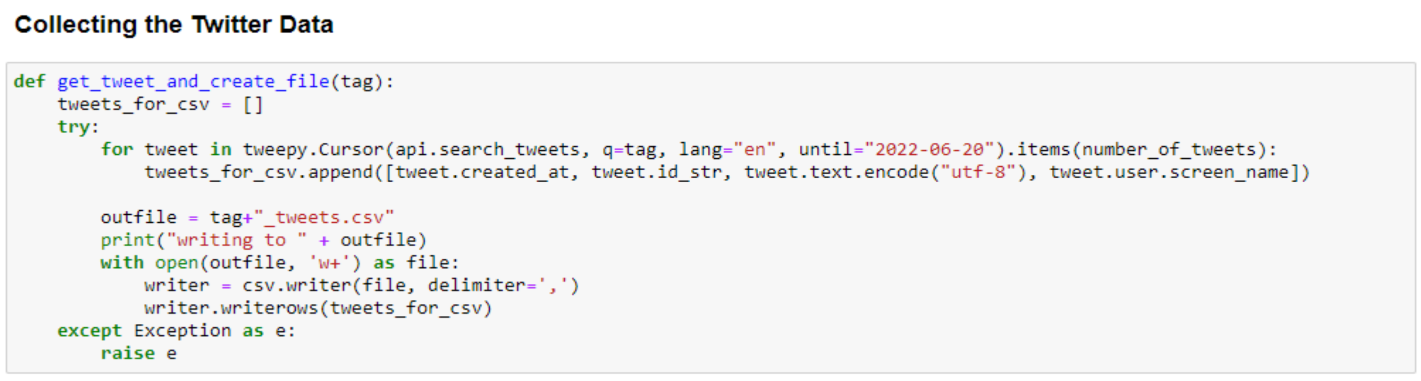
Below are the ticker symbols #key terms on which we are trying to collect tweets data and users information along with their creation date to perform visualization.



Twitter tags to collect data

```
listOfTags = ["#Altcoin", "#Gold", "#APPL", "#GOOG", "#YHOO" , "#Bitcoin", "#Coindesk", "#Cryptocurrency"]
```

The below function is used to collect and store the tweets information in the respective #key_tweets.csv files. Here, we are using Cursors which are implemented as a Tweepy class named Cursor. To use this Cursor, we select the API method to fetch items and the number of items needed to be stored. We are collecting the data as below format i.e., created_at, tweet_id, tweet_textinfo, screenname(user_details). The until parameter is used to fetch the tweets from 2022-06-14 till 2022-06-20.

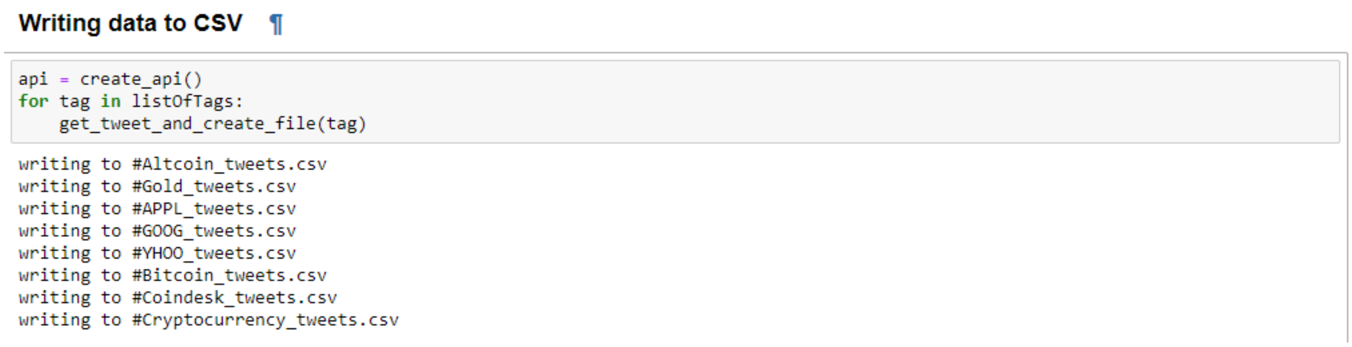


Collecting the Twitter Data

```
def get_tweet_and_create_file(tag):
    tweets_for_csv = []
    try:
        for tweet in tweepy.Cursor(api.search_tweets, q=tag, lang="en", until="2022-06-20").items(number_of_tweets):
            tweets_for_csv.append([tweet.created_at, tweet.id_str, tweet.text.encode("utf-8"), tweet.user.screen_name])

        outfile = tag+"_tweets.csv"
        print("writing to " + outfile)
        with open(outfile, 'w+') as file:
            writer = csv.writer(file, delimiter=',')
            writer.writerows(tweets_for_csv)
    except Exception as e:
        raise e
```

Here, we are trying to call the create_api() user defined function to gain access to Twitter API for reading and writing the data to the respective listofTags mentioned as .csv format by using get_tweet_and_create_file(tag) function. We can observe that 8 different csv files have been saved with tweets information.

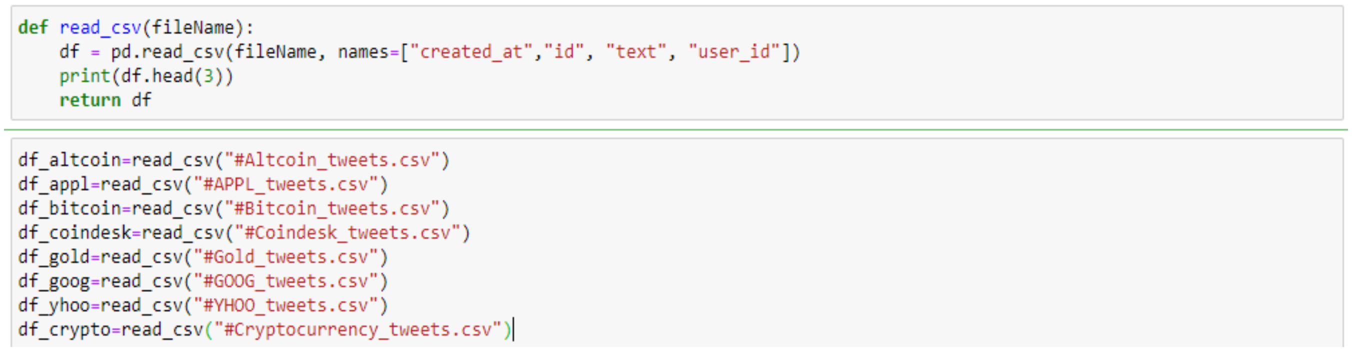


Writing data to CSV ¶

```
api = create_api()
for tag in listOfTags:
    get_tweet_and_create_file(tag)
```

```
writing to #Altcoin_tweets.csv
writing to #Gold_tweets.csv
writing to #APPL_tweets.csv
writing to #GOOG_tweets.csv
writing to #YHOO_tweets.csv
writing to #Bitcoin_tweets.csv
writing to #Coindesk_tweets.csv
writing to #Cryptocurrency_tweets.csv
```

Here, we have created a user defined function called read_csv(filename) to read 8 different files.



```
def read_csv(fileName):
    df = pd.read_csv(fileName, names=["created_at","id", "text", "user_id"])
    print(df.head(3))
    return df
```

```
df_altcoin=read_csv("#Altcoin_tweets.csv")
df_appl=read_csv("#APPL_tweets.csv")
df_bitcoin=read_csv("#Bitcoin_tweets.csv")
df_coindesk=read_csv("#Coindesk_tweets.csv")
df_gold=read_csv("#Gold_tweets.csv")
df_goog=read_csv("#GOOG_tweets.csv")
df_yhoo=read_csv("#YHOO_tweets.csv")
df_crypto=read_csv("#Cryptocurrency_tweets.csv")
```

For the data cleaning step - the punctuations, numbers in tweets text, words less than 2 letters and duplicate records needs to be handled. To perform these tasks a function named clean_data(df) was defined which takes the dataframe as the input and performs all the cleaning operations accordingly.

Data Cleaning

```
def clean_data(df):
    df['text'] = df['text'].str.replace(r'[^\w\s]+', '')
    df['text'] = df['text'].str.replace('\d+', '')
    df['text']=df.text.str.replace(r'\b(\w{1,2})\b', '')
    df['created_at'] = pd.to_datetime(df['created_at']).dt.date
    df.drop_duplicates(subset ="text",keep = False, inplace = True)
```

```
for dataframe in df_list:
    clean_data(dataframe)
```

For the visualization, we have used both the bar plot and word cloud methods. For the bar plot, we have utilized both the number of users and number of tweets per day for each keyterm. This was achieved by defining two functions visualize_tweets(df, tag) and visualize_users(df, tag) which takes the df and tag as input and gives out the barplots. We have tried to find the most trendy words related to tweets collected with respect to the #keyterms and visualized using WordCloud technique.

Visualization

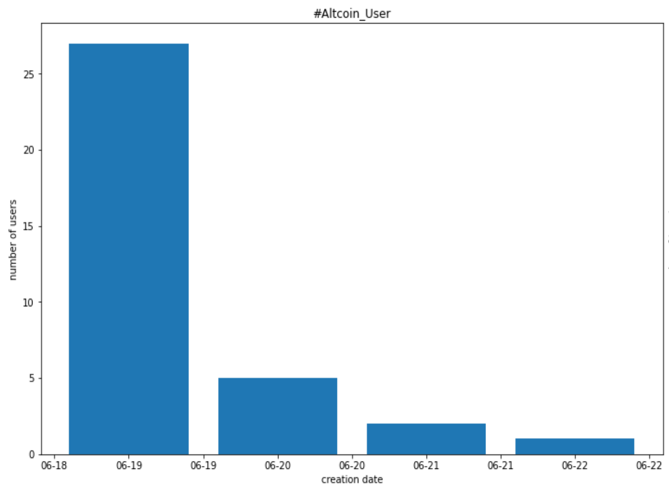
```
def visualize_tweets(df, tag):
    x = np.unique(df["created_at"])
    y = df['created_at'].value_counts()
    fig, ax = plt.subplots(figsize=(12, 8))
    ax.xaxis.set_major_formatter(DateFormatter("%m-%d"))
    plt.ylabel("number of tweets")
    plt.xlabel("creation date")
    plt.title(tag+"_Tweets")
    plt.bar(x,y)
    plt.show()
```

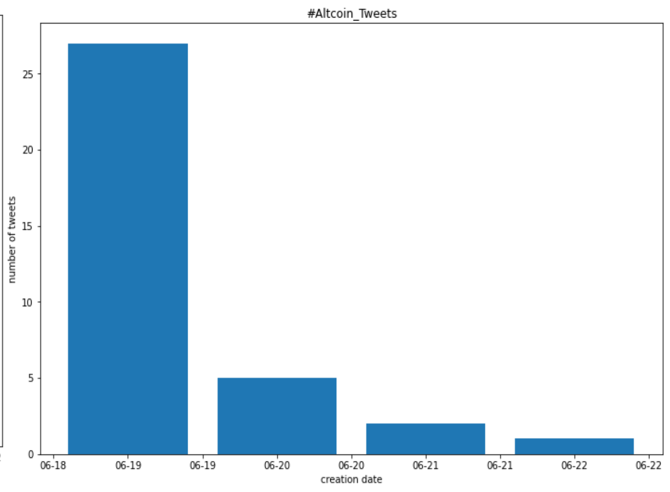
```
def visualize_users(df, tag):
    df_users=df[["created_at", 'user_id']].value_counts()
    x = np.unique(df["created_at"])
    y = df_users.groupby(['created_at']).sum()
    fig, ax = plt.subplots(figsize=(12, 8))
    ax.xaxis.set_major_formatter(DateFormatter("%m-%d"))
    plt.ylabel("number of users")
    plt.xlabel("creation date")
    plt.title(tag+"_User")
    plt.bar(x,y)
    plt.show()
```

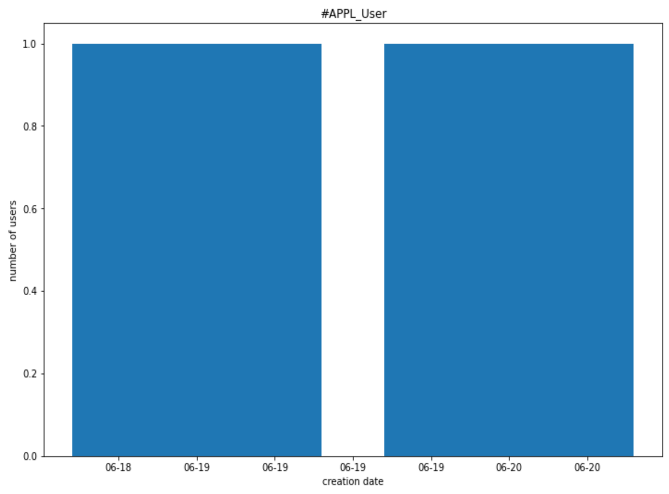
```
listOfTags = ["#Altcoin",  "#APPL",  "#Bitcoin", "#Coindesk","#Gold", "#GOOG", "#Cryptocurrency", "#YHOO"]
```

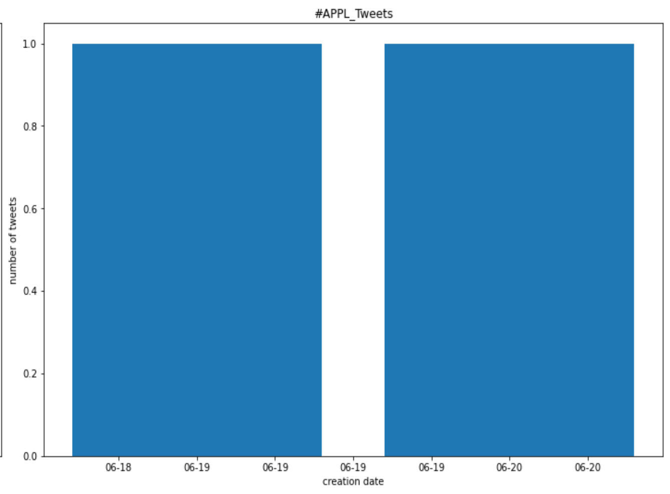
```
df_list=[df_altcoin,df_appl,df_bitcoin,df_coindesk,df_gold,df_goog,df_crypto,df_yhoo]
count=0
for i in df_list:
    visualize_users(i, listOfTags[count])
    visualize_tweets(i, listOfTags[count])
    count=count+1
```

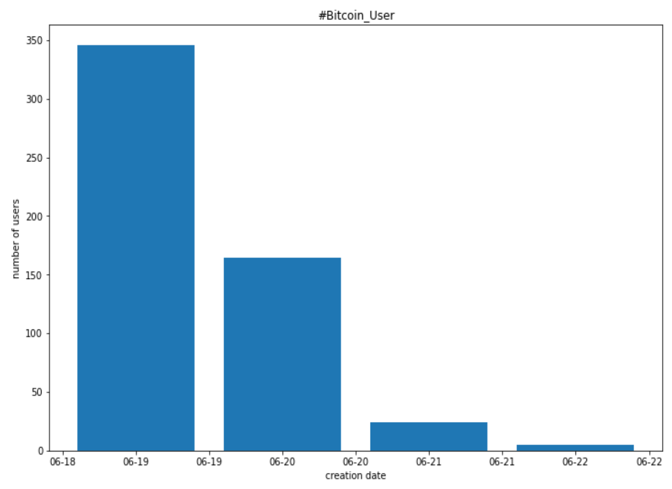
The following graphs were plotted by using the tweets data which we have saved in the csv format. The below figures shows the number of users and number of tweets with respect to creation date for all the eight hashtags.

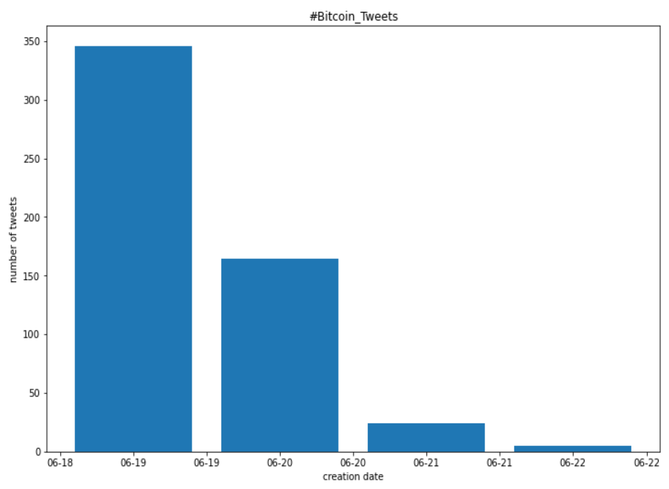


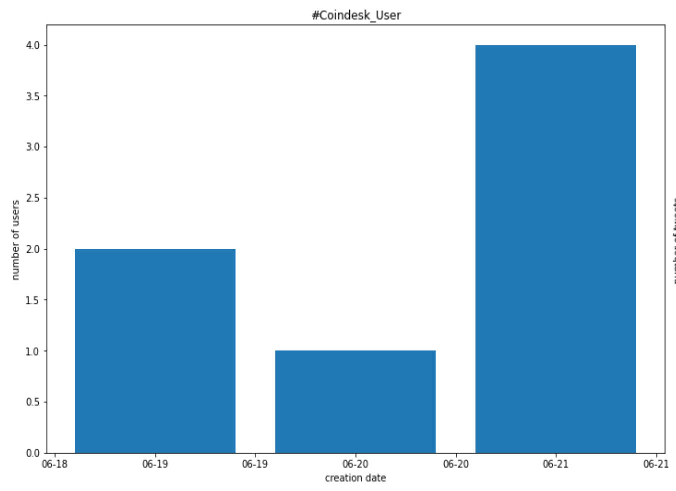


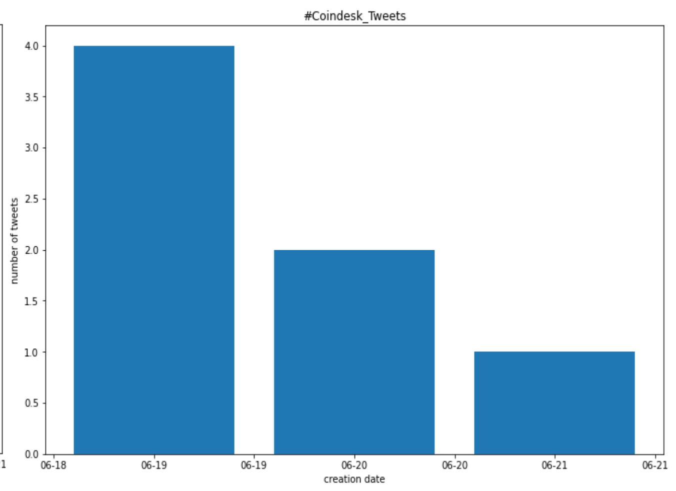


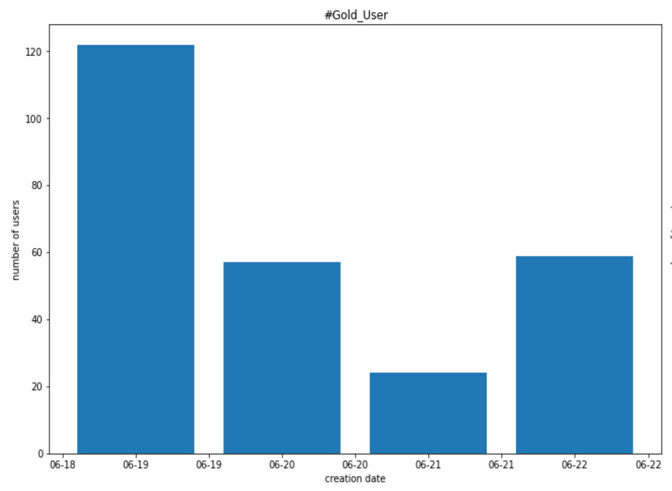


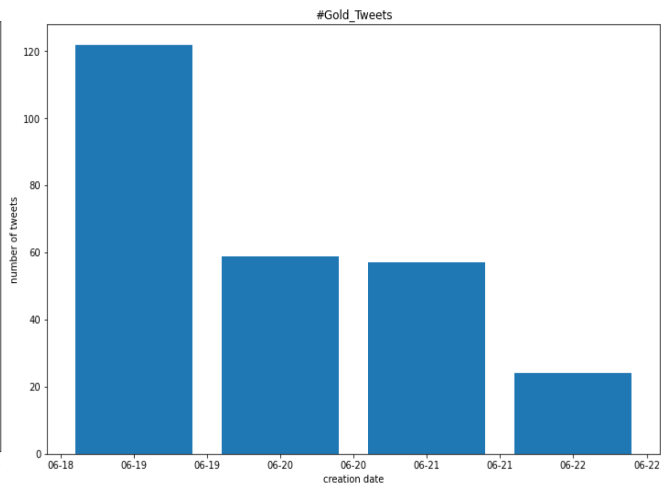

#Bitcoin_User
number of users
creation date
0
50
100
150
200
250
300
350
06-18
06-19
06-19
06-20
06-20
06-21
06-21
06-22
06-22

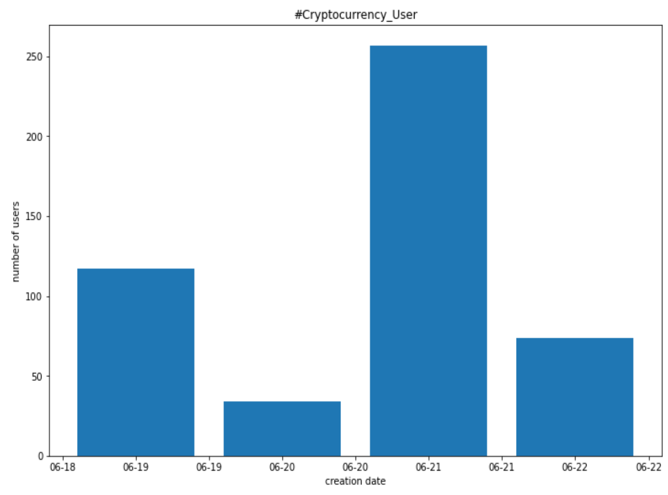

#Bitcoin_Tweets
number of tweets
creation date
0
50
100
150
200
250
300
350
06-18
06-19
06-19
06-20
06-20
06-21
06-21
06-22
06-22

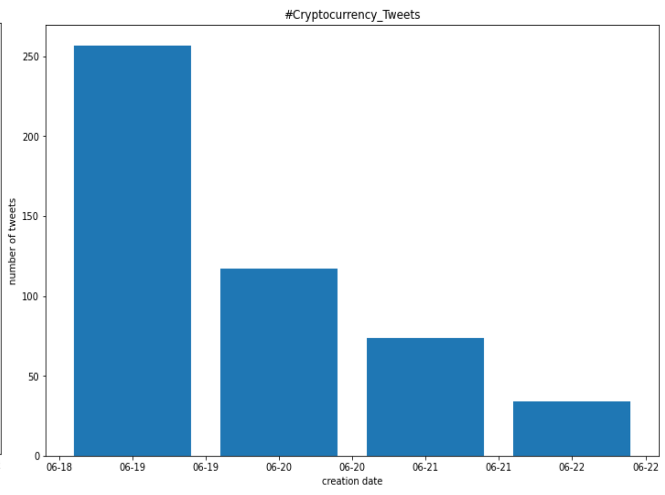

#Coindesk_User
number of users
creation date
0.0
0.5
1.0
1.5
2.0
2.5
3.0
3.5
4.0
06-18
06-19
06-19
06-20
06-20
06-21
06-21

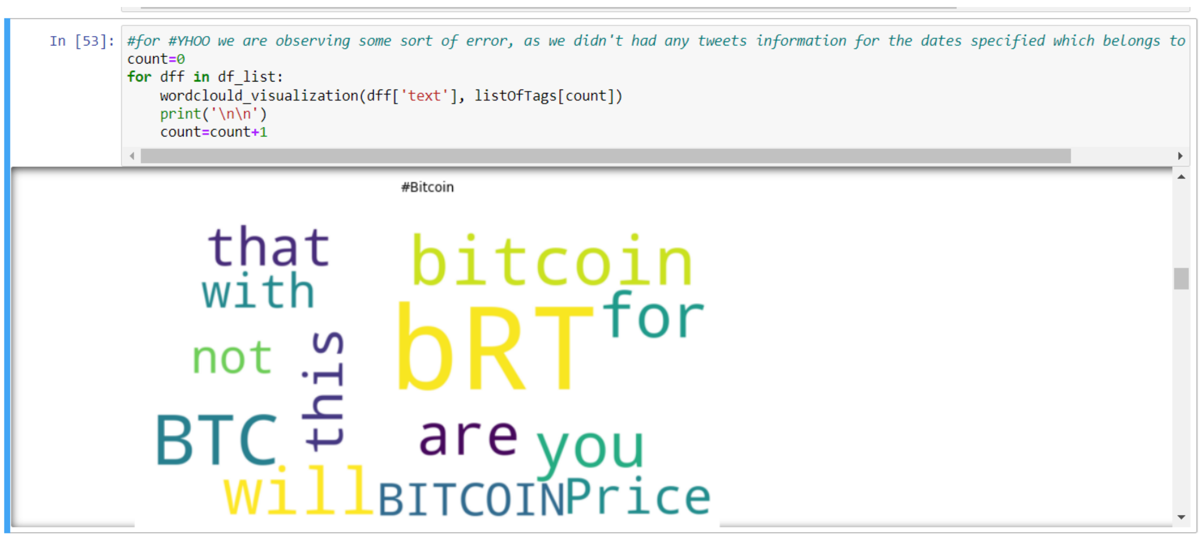

#Coindesk_Tweets
number of tweets
creation date
0.0
0.5
1.0
1.5
2.0
2.5
3.0
3.5
4.0
06-18
06-19
06-19
06-20
06-20
06-21
06-21


#Gold_User
number of users
creation date
0
20
40
60
80
100
120
06-18
06-19
06-19
06-20
06-20
06-21
06-21
06-22
06-22


#Gold_Tweets
number of tweets
creation date
0
20
40
60
80
100
120
06-18
06-19
06-19
06-20
06-20
06-21
06-21
06-22
06-22







```
In [53]: #for #YHOO we are observing some sort of error, as we didn't had any tweets information for the dates specified which belongs to
         count=0
         for dff in df_list:
             wordclould_visualization(dff['text'], listOfTags[count])
             print('\n\n')
             count=count+1
```

Conclusions:

We are able to connect and retrieve the tweets using Tweepy API for all the crypto and commodities specified in the task. The records retrieved are very less for some of the tages like #AAPL, #GOOG and #YHOO which is just having 2,3,0 records for these respective tags for the specified duration of one week.

Sometimes, we faced issues in retrieving the huge data (suppose 10,000 records for each term) in terms of the number of requests that have been sent to twitter.

After performing visualizations on the daily number of tweets for each keyterm as well as the daily numbers of users, we have found out that there were more tweets and users produced on 19-06-2022 for most of the keyterms.

References:

Atkinson, B,.(2019, Jun 25), *Visualisation of Information from Raw Twitter Data — Part 1,* https://towardsdatascience.com/visualization-of-information-from-raw-twitter-data-part-1-99181ad19c

Au-Yeung, J., (2021, October 23), *How to remove punctuation with Python Pandas?,* https://thewebdev.info/2021/10/23/how-to-remove-punctuation-with-python-pandas/#:~:text=To%20remove%20punctuation%20with%20Python%20Pandas%2C%20we%20can%20use%20the,replace%20method.&text=We%20call%20replace%20with%20a,replace%20them%20with%20empty%20strings.&text=replace%20returns%20a%20new%20DataFrame,df%5B%27text%27%5D%20.

GeeksforGeeks, (2021, Jan 21), *Convert CSV to JSON using Python* https://www.geeksforgeeks.org/convert-csv-to-json-using-python/

Garcia, M., (2022), *How to Make a Twitter Bot in Python With Tweepy,* https://realpython.com/twitter-bot-python-tweepy/

Luara, G., (2019, Jan 6), *Python Script to Download Tweets,* https://github.com/gitlaura/get_tweets

Pieters, M., (2014, Jan 20), *Remove words of length less than 4 from string [duplicate],* https://stackoverflow.com/questions/24332025/remove-words-of-length-less-than-4-from-string

pynative, (2022, May 6), *Python DateTime Format Using Strftime()* https://pynative.com/python-datetime-format-strftime/#:~:text=Use%20datetime.,hh%3Amm%3Ass%20format.

Python – *Get Filename from Path with Examples,* https://datascienceparichay.com/article/python-get-filename-from-path-with-examples/

Pandas.pydata.org, *pandas.DataFrame.groupby*

https://pandas.pydata.org/docs/reference/api/pandas.DataFrame.groupby.html

Roesslein, J., (2022) *Tweepy Documentation*

https://docs.tweepy.org/en/stable/client.html#tweepy.Client.search_recent_tweets

Roesslein, J., (2022) *Cursor Tutorial*

https://docs.tweepy.org/en/v3.4.0/cursor_tutorial.html