**19OH01- SOCIAL AND ECONOMIC NETWORK ANALYSIS**

BABITHA REDDY C H - 18Z211

M BHAVYASREE - 18Z214

CHERUKURI SHALINI - 18Z215

KEERTHANA.B -18Z226

KOPPETI JOSHNA - 18Z332

Group Report submitted in partial fulfilment of the

requirements for the degree of

**BACHELOR OF ENGINEERING**

**Branch: COMPUTER SCIENCE AND ENGINEERING**

Of Anna University APRIL 2021



**PSG COLLEGE OF TECHNOLOGY**

(Autonomous Institution)

**COIMBATORE – 641 004**

Table of Contents

1. Problem Statement…………………………………………………………………...3
2. Dataset description…………………………………………………………………...4
3. Tools used……………………………………………………………...…………….5
4. Challenges Faced……………………………………………………………….……7
5. Contribution of Team Members…………………………………………………..….8
6. Annexure I: Code………………………………………………….…………………9
7. Annexure II: Snapshots of the Output……………………………………………….21
8. References…………………………………………………………………………………24
9. **PROBLEM STATEMENT**

A great activity concerning COVID-19 has emerged on Twitter when on March 11, the World Health Organization declared it a pandemic. As the social media platforms can help track natural disasters in real-time, the most preferred platform Twitter is used in this project to study discourse around the novel virus using some data science techniques.

Social media, in particular, plays an important role in successfully communicating risk to the larger public. Monitoring public discourse on social media during a pandemic situation is critical to evaluate the effectiveness of risk communication efforts. We research discourse on Twitter around coronavirus disease 2019 (COVID-19) which focuses on extracting themes from tweets mentioning “Coronavirus” and “COVID-19” and other hashtags. Tweets can be either a reply to another tweet and/or a quote. Otherwise, it is just a simple tweet. We use these properties to see whether debate or sharing explains an activity on Twitter around a specific topic. Hashtags and mentions in the tweets are taken into account as graphs and analyzed.

1. **DATASET DESCRIPTION**

According to the WHO (World Health Organization), risk communication is essential to help people understand the cause, how to prevent the cause, stop the spread of disease, and limit the social and economic impact of an outbreak. Social media, in particular, plays an important role in successfully communicating risk to the larger public. The dataset contains variables associated with Twitter: the text of various tweets and the accounts that tweeted them, the hashtags used and the locations of the accounts.

Due to the large volume of Tweets, there may be some gaps for some hashtags (not all Tweets with a given hashtag may be captured). Because some hashtags are used less frequently than other hashtags, less frequently used hashtags may span a longer period of time (going back earlier) than more frequently used hashtags. The hashtag "#coronavirus" seems to be the most frequently used - despite scraping 500,000 Tweets, there was no overlap between Tweets with this hashtag in version 1 and version 5, therefore gaps remain. The retweets argument has been set to FALSE, so this dataset does not include retweets (although a count of retweets is provided as a variable). A dataset containing these tweets were considered in the middle of April 2020.

**3. TOOLS USED**

**PYTHON:**

Python is a widely used scripting language. Python's design philosophy emphasizes code readability with its notable use of significant indentation. Python consists of various libraries which help view and analyze various networks by plotting graphs.

**Jupiter Notebook:** The Jupiter Notebook has been used to execute PYTHON 3 code for dataset cleaning and analysis.

These are the following libraries that have been used in this project:

1. **Pandas:**

pandas is a Python package providing fast, flexible, and expressive data structures designed to make working with “relational” or “labeled” data both easy and intuitive. In particular, it offers data structures and operations for manipulating numerical tables and time series. It aims to be the fundamental high-level building block for doing practical, real-world data analysis in Python.

1. **Numpy:**

NumPy is a library for the Python programming language, adding support for large, multi-dimensional arrays and matrices, along with a large collection of high-level mathematical functions to operate on these arrays.

1. **Plotly:**

plotly.py is an interactive, open-source, and browser-based graphing library for Python sparkles. Built on top of plotly.js, plotly.py is a high-level, declarative charting library. plotly.js ships with over 30 chart types, including scientific charts, 3D graphs, statistical charts, SVG maps, financial charts, and more.

1. **Seaborn:**

Seaborn is a Python data visualization library based on matplotlib. It provides a high-level interface for drawing attractive and informative statistical graphics.

1. **NLTK:**

Natural Language Processing with Python NLTK is one of the leading platforms for working with human language data and Python, the module NLTK is used for natural language processing. NLTK is literally an acronym for Natural Language Toolkit. It helps us to tokenize data by words and sentences.

1. **Wordcloud:**

Word Cloud is a data visualization technique used for representing text data in which the size of each word indicates its frequency or importance. Significant textual data points can be highlighted using a word cloud. Word clouds are widely used for analyzing data from social network websites.

1. **Advertools:**

advertools is a Python package that manages, manipulates, visualizes, communicates, understands, and make decisions based on data.

1. **Matplot:**

Matplotlib is an amazing visualization library in Python for 2D plots of arrays. Matplotlib is a multi-platform data visualization library built on NumPy arrays and designed to work with the broader SciPy stack

**4. CHALLENGES FACED**

* The dataset used in this project initially had a number of empty columns that were unnecessary. Those columns had to be removed and the dataset had to be preprocessed.
* The dataset contains language column that had various languages other than English. Those rows had to be removed and only English tweets were taken into account for further analysis.
* Due to the large volume of Tweets, there may be some gaps for some hashtags (not all Tweets with a given hashtag may be captured).
* Due to the large volume of the dataset, the execution process takes more time and thus, system may freeze at times.

**5. CONTRIBUTION OF TEAM MEMBERS**

|  |  |  |
| --- | --- | --- |
| Roll no | Name | Contribution |
| 18Z211 | Babitha Reddy C H | Preprocessing and frequency of words |
| 18Z214 | M Bhavysree | Word cloud and sentiment analysis |
| 18Z215 | Cherukuri Shalini | Word cloud based on sentiment scores |
| 18Z226 | B Keerthana | Hashtag analysis |
| 18Z332 | Koppeti Joshna | Top hashtags and mention analysis |

**6. ANNEXURE I: CODE**

#importing the packages panda and numpy

import pandas as pd

import numpy as np

#Reading the dataset(csv file) into the dataframe

df = pd.read\_csv(r'/Users/bhavya/Downloads/Coronavirus Tweets.CSV')

#Preprocessing the dataset

#Dropping columns that aren't required

to\_drop = ['reply\_to\_status\_id',

'reply\_to\_user\_id',

'reply\_to\_screen\_name',

'country\_code',

'place\_full\_name',

'place\_type',

'account\_lang']

df.drop(to\_drop, inplace=True, axis=1)

df = df.dropna(how = 'all')

#Filtering the dataset such that it contains tweets only of english language

df.drop(df[df['lang'] != "en"].index, inplace = True)

#After applying above filters the dataframe is converted to csv and it will be our dataset.

df.to\_csv("File2.csv", index=False)

#importing the basic packages

import io

import random

import string

import warnings

import pandas as pd

import numpy as np

import advertools as adv

#importing package to generate wordcloud

from wordcloud import WordCloud

#In the context of the project this package is required for generating stopwords

from sklearn.feature\_extraction.text import TfidfVectorizer

#Required Package to measure similarities between texts

from sklearn.metrics.pairwise import cosine\_similarity

#package to display warning messages

import warnings

warnings.filterwarnings('ignore')

#importing natural language processing toolkits

import nltk

#Package to break the sentences from the paragaraph of the tweet text into tokens

from nltk.tokenize import sent\_tokenize

#importing words and stopwords from the corpus of nltk for wordcloud and sentiment analysis

from nltk.corpus import words

from nltk.corpus import stopwords

#package to break the sentences of the tweet text into seperate words

from nltk.tokenize import word\_tokenize

#This package is imported to accept the list of tokenized words and stems it into root word.

from nltk.stem import WordNetLemmatizer

#This package is imported to accept the list of tokenized words and stems it into root word.

from nltk.stem import PorterStemmer

#importing these packages to perform Sentiment Analysis

from nltk.sentiment.vader import SentimentIntensityAnalyzer

from nltk.sentiment.util import \*

# sklearn imports

from sklearn.model\_selection import train\_test\_split

from sklearn.feature\_extraction.text import TfidfVectorizer

from sklearn.naive\_bayes import MultinomialNB

from sklearn import metrics

# python imports

import re

import os

from collections import Counter

import datetime as dt

# Visualization

from matplotlib import pyplot as plt

from matplotlib import ticker

import seaborn as sns

from sklearn import feature\_extraction, linear\_model, model\_selection, preprocessing

from wordcloud import WordCloud

from tqdm import tqdm\_notebook

# Saving models

import pickle

import plotly.express as px

import chart\_studio.plotly as py

#Location of our csv file

data\_dir = 'NewFolder'

#initialising a list

tweets = []

#Extracting the tweets from the dataset and appending it to the tweets list and storing it to dataframe

for file in sorted(os.listdir(data\_dir)):

tweets.append(pd.read\_csv(data\_dir + '/' + file, lineterminator = '\n'))

df = pd.concat(tweets)

df.tail()

'''Code to analyse frequency of words from the tweet text'''

#Extracting only tweet text from dataframe

text\_en = df['text']

#Dataset Preparation : Removing URLs,Stopwords,punctuations and hashtags and converting text into lower case

text\_en\_lr = text\_en.apply(lambda x: re.sub(r"https\S+", "", str(x)))

text\_en\_lr\_lc = text\_en\_lr.apply(lambda x: x.lower())

text\_en\_lr\_lc\_pr = text\_en\_lr\_lc.apply(lambda x: x.translate(str.maketrans('', '', string.punctuation)))

stop\_words = set(stopwords.words('english'))

stop\_words.update(['#coronavirus', '#coronavirusoutbreak', '#coronavirusPandemic', '#covid19', '#covid\_19', '#epitwitter', '#ihavecorona', 'amp','coronavirus', 'covid19'])

#extracting the words and storing

text\_en\_lr\_lc\_pr\_sr = text\_en\_lr\_lc\_pr.apply(lambda x: ' '.join([word for word in x.split() if word not in stop\_words]))

word\_list = [word for line in text\_en\_lr\_lc\_pr\_sr for word in line.split()]

#Using seaborn library for styling

sns.set(style="darkgrid")

#making use of counter package to count the frequency of words

counts = Counter(word\_list).most\_common(50)

counts\_df = pd.DataFrame(counts)

counts\_df

counts\_df.columns = ['word', 'frequency']

#The bargraph is plotted against frequency and the word

#plots displayed using matlplots and seaborn

fig, ax = plt.subplots(figsize = (12, 12))

ax = sns.barplot(y="word", x='frequency', ax = ax, data=counts\_df)

plt.savefig('wordcount\_bar.png')

'''Code for generating Wordcloud'''

#generating the wordcloud

wordcloud = WordCloud(

background\_color='black',

max\_words=50,

max\_font\_size=40,

scale=5,

random\_state=1,

collocations=False,

normalize\_plurals=False

).generate(' '.join(word\_list))

#displaying the wordcloud using matplot

plt.figure(figsize = (12, 10), facecolor ='none')

plt.imshow(wordcloud)

plt.axis("off")

plt.tight\_layout(pad = 0)

plt.savefig('wordcloud.png')

'''Code for finding Sentiments of tweets'''

# Creating a SentimentIntensityAnalyzer object

sid = SentimentIntensityAnalyzer()

'''The polarity\_scores method below of SentimentIntensityAnalyzer

object gives a sentiment dictionary.

which contains pos, neg, neu, and compound scores.'''

sentiment\_scores = text\_en\_lr\_lc\_pr\_sr.apply(lambda x: sid.polarity\_scores(x))

#storing the sentiment scores in a dataframe

sent\_scores\_df = pd.DataFrame(list(sentiment\_scores))

#calculating count of the positive,negative and neutral statements

sent\_scores\_df['val'] = sent\_scores\_df['compound'].apply(lambda x: 'neutral' if x == 0 else ('positive' if x > 0 else 'negative'))

sent\_scores\_df['val'] = sent\_scores\_df['compound'].apply(lambda x: 'neutral' if x == 0 else ('positive' if x > 0 else 'negative'))

sent\_counts = pd.DataFrame.from\_dict(Counter(sent\_scores\_df['val']), orient = 'index').reset\_index()

sent\_counts.columns = ['sentiment', 'count']

#plotting frequency against sentiments

sns.barplot(y="count", x='sentiment', data=sent\_counts)

plt.savefig('sentiment.png')

'''Code to generate WORDCLOUDs based on sentiment scores'''

polar\_tweets\_df = pd.DataFrame()

polar\_tweets\_df['tweet'] = text\_en\_lr\_lc\_pr\_sr

polar\_tweets\_df['polarity'] = sent\_scores\_df['val']

#Separating tweet words based on polarity scores

positive = polar\_tweets\_df[polar\_tweets\_df['polarity'] == 'positive']['tweet']#positive tweets

negative = polar\_tweets\_df[polar\_tweets\_df['polarity'] == 'negative']['tweet']#negative tweets

neutral = polar\_tweets\_df[polar\_tweets\_df['polarity'] == 'neutral']['tweet']#neutral tweets

#Creating word lists of positive negative and neutral words

positive\_list = [word for line in positive for word in line.split()]

negative\_list = [word for line in negative for word in line.split()]

neutral\_list = [word for line in neutral for word in line.split()]

#Generating wordcloud for positive words

positive\_cloud = WordCloud(

background\_color='black',

max\_words=50,

max\_font\_size=40,

scale=5,

random\_state=1,

collocations=False,

normalize\_plurals=False

).generate(' '.join(positive\_list))

#Generating wordcloud for negative words

negative\_cloud = WordCloud(

background\_color='black',

max\_words=50,

max\_font\_size=40,

scale=5,

random\_state=1,

collocations=False,

normalize\_plurals=False

).generate(' '.join(negative\_list))

#Generating wordcloud for neutral words

neutral\_cloud = WordCloud(

background\_color='black',

max\_words=50,

max\_font\_size=40,

scale=5,

random\_state=1,

collocations=False,

normalize\_plurals=False

).generate(' '.join(neutral\_list))

#Setting up layout to plot four different wordclouds

fig, axs = plt.subplots(2, 2, figsize = (20, 12))

fig.tight\_layout(pad = 0)

#using the imshow function of the matplot package to display the generated wordclouds

#Positive word cloud

axs[0, 0].imshow(positive\_cloud)

axs[0, 0].set\_title('Words from positive tweets', fontsize = 20)

axs[0, 0].axis('off')

#negative word cloud

axs[0, 1].imshow(negative\_cloud)

axs[0, 1].set\_title('Words from negative tweets', fontsize = 20)

axs[0, 1].axis('off')

#Neutral word cloud

axs[1, 0].imshow(neutral\_cloud)

axs[1, 0].set\_title('Words from neutral tweets', fontsize = 20)

axs[1, 0].axis('off')

#Also displaying the wordcloud generated before classifying into positive,negative and neutral

axs[1, 1].imshow(wordcloud)

axs[1, 1].set\_title('Words from all tweets', fontsize = 20)

axs[1, 1].axis('off')

#saving the plot

plt.savefig('joint\_cloud.png')

tweets = pd.read\_csv("File1.csv")

#Function to get the hashtags

#if a particular hashtag is present in the text appending the hashtag to the list

def get\_hashtag(row):

tweet = []

text = row["text"].lower()

#Check for hashtag coronavirus

if "#coronavirus" in text :

tweet.append("#coronavirus")

#Check for hashtag covid19

if "#covid19" in text :

tweet.append("#covid19")

#check for hashtag lockdown

if "#lockdown" in text :

tweet.append("#lockdown")

#check for hashtag stayhomestaysafe

if "#stayhomestaysafe" in text :

tweet.append("#stayhomestaysafe")

return ",".join(tweet)

#Function call

#Using the apply method to work for every row and column

tweets["hashtags"] = tweets.apply(get\_hashtag,axis=1)

#finding the frequency of each hashtag

counts = tweets["hashtags"].value\_counts()

#plotting the count as a bargraph

plt.bar(range(len(counts)), counts)

plt.show()

print(counts)

#Storing the status id that used the particular hashtag

cl\_tweets = tweets["status\_id"][tweets["hashtags"] == "#coronavirus"]

sa\_tweets = tweets["status\_id"][tweets["hashtags"] == "#covid19"]

tr\_tweets = tweets["status\_id"][tweets["hashtags"] == "#lockdown"]

a=tweets["status\_id"][tweets["hashtags"] == "#stayhomestaysafe"]

#plotting the histogram

plt.hist([

cl\_tweets,

sa\_tweets,

tr\_tweets,

a

],

stacked=True,

label=["coronavirus", "covid", "lockdown","stayhomestaysafe"])

#Giving labels to the plot

plt.legend()

plt.title("Tweets mentioning each hashtag")

plt.xlabel("Status id")

plt.ylabel("# of tweets")

plt.show()

#using the advertools method to extract mentions and hashtags

[x for x in dir(adv) if x.startswith('extract')]

#methods to analyse hashtags

hashtag\_summary = adv.extract\_hashtags(tweets['text'])

hashtag\_summary.keys()

#number of hashtags in the dataset

hashtag\_summary['overview']

#To get hashtags used by people

hashtag\_summary['hashtags']

#Hashtags limited to a count of 10 sets

#Multidimensional list of the hashtags

hashtag\_summary['hashtags'][:10]

#Single dimensional list of the first 10 hashtags extracted

hashtag\_summary['hashtags\_flat'][:10]

#Counting the hashtags used overall and hashtags used per tweet

hashtag\_summary['hashtag\_counts'][:20]

hashtag\_summary['hashtag\_freq'][:15]

#plotting the hashtags per tweet against the total number of tweets

plt.figure(facecolor='#ebebeb', figsize=(11, 8))

plt.bar([x[0] for x in hashtag\_summary['hashtag\_freq'][:15]],

[x[1] for x in hashtag\_summary['hashtag\_freq'][:15]])

#Labelling the plot

plt.title('Hashtag frequency', fontsize=18)

plt.xlabel('Hashtags per tweet', fontsize=12)

plt.ylabel('Number of tweets', fontsize=12)

#Setting up the dimensions

plt.xticks(range(16))

plt.yticks(range(0,28000, 1000))

plt.grid(alpha=0.5)

plt.gca().set\_frame\_on(False)

fig = px.bar(x=freq[2:], y=top\_hashtags[2:], orientation='h')

fig.update\_layout(

height=600, width=700,

title\_text='Most Popular Hashtags',

xaxis = {'title': 'Frequency'},

yaxis = {'autorange': "reversed", 'title':''}

)

fig.show()

mention\_summary = adv.extract\_mentions(tweets['text'])

mention\_summary.keys()

#extracting the number of mentions from tweet set

mention\_summary['overview']

#Multidimensional list of mentions

mention\_summary['mentions'][:15]

#single dimensional list of mentions

mention\_summary['mentions\_flat'][:10]

#extracting top 20 mentions

mention\_summary['top\_mentions'][:20]

#using the matplot package to plot the top mentions

plt.figure(facecolor='#ebebeb', figsize=(8, 8))

plt.barh([x[0] for x in mention\_summary['top\_mentions'][:15]][::-1],

[x[1] for x in mention\_summary['top\_mentions'][:15]][::-1])

#labeling the plot

plt.title('Top Mentions')

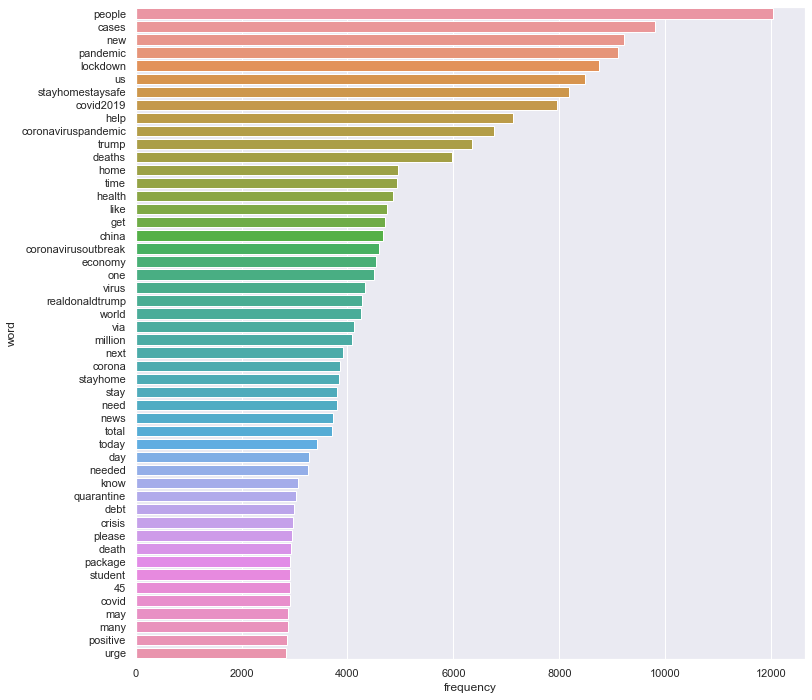
plt.grid(alpha=0.5)

plt.xticks(range(0, 5000, 1000))

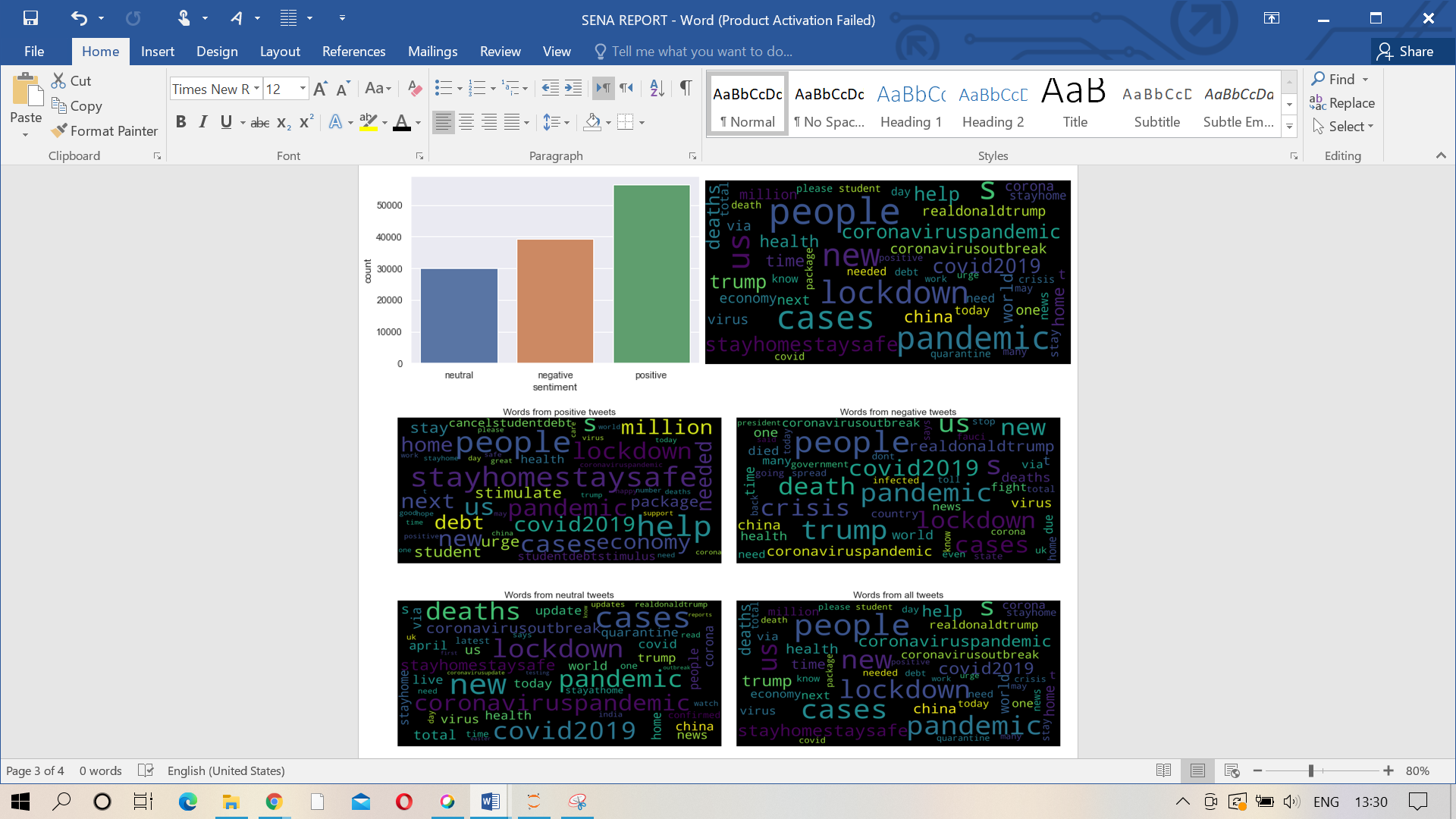
plt.gca().set\_frame\_on(False)

**7. ANNEXURE II: SCREENSHOTS OF THE OUTPUT**

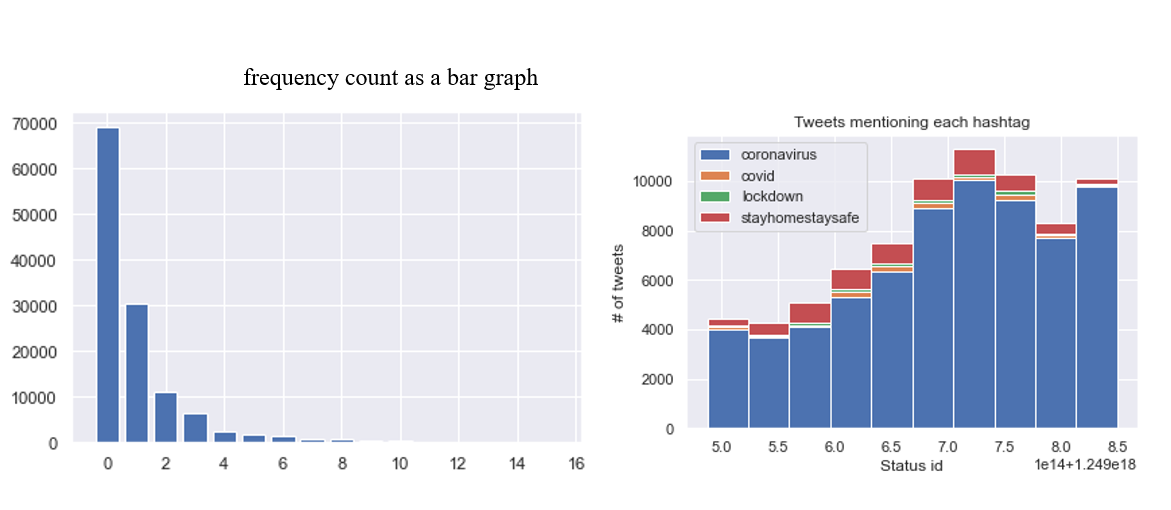
Frequency of the words:



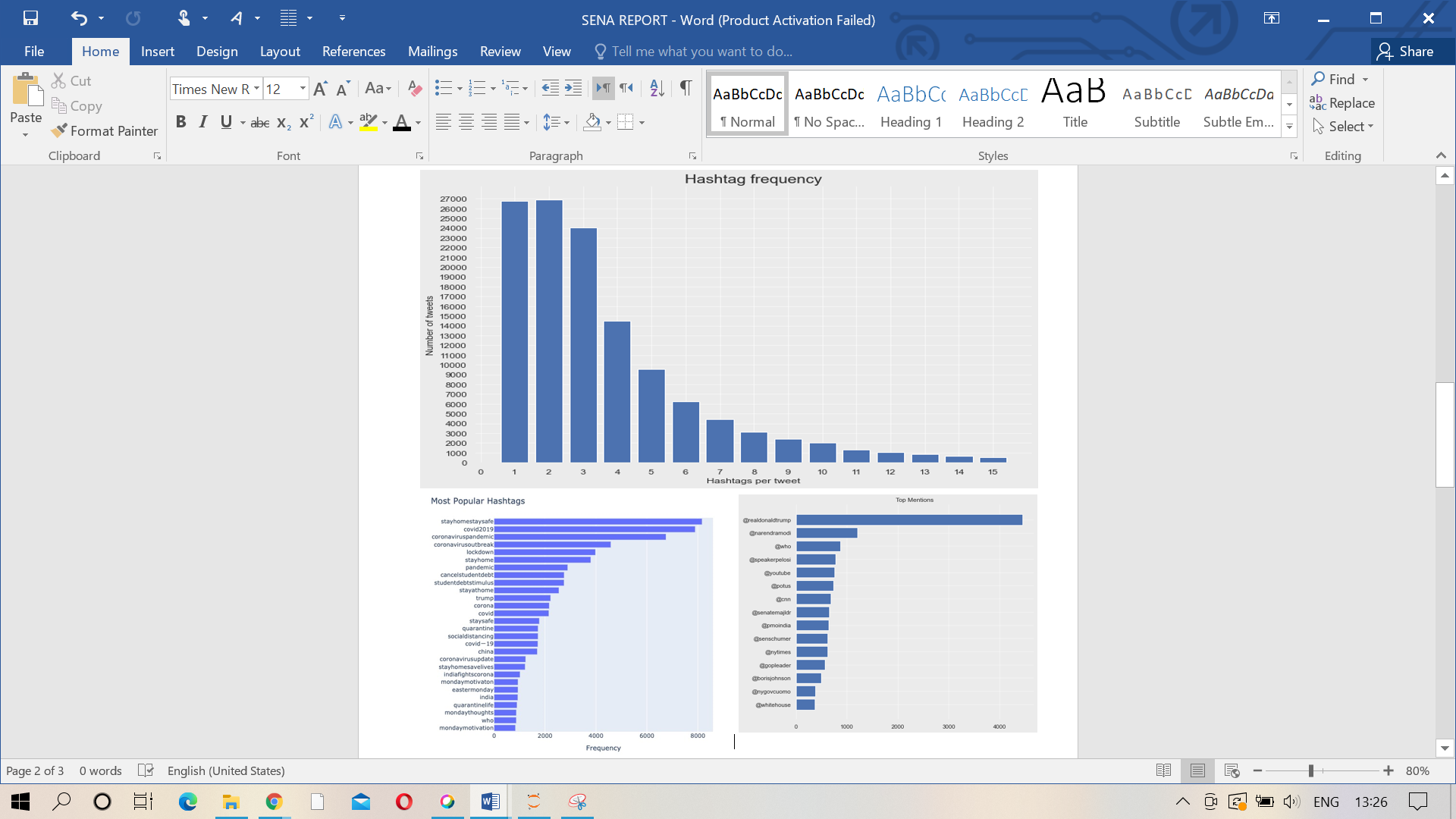
Word cloud and sentiment analysis:



Hashtag analysis – bar graph and histogram:



Hashtag frequency, Top hashtags and mentions:



1. **REFERENCES**
   1. <https://www.geeksforgeeks.org/twitter-sentiment-analysis-using-python/>
   2. <https://towardsdatascience.com/step-by-step-twitter-sentiment-analysis-in-python-d6f650ade58d>
   3. <https://www.kaggle.com/jagangupta/wordcloud-of-tweets>
   4. <https://realpython.com/python-nltk-sentiment-analysis/>
   5. <https://www.datacamp.com/community/tutorials/wordcloud-python>
   6. <https://www.cambridge.org/core/journals/disaster-medicine-and-public-health-preparedness/article/social-network-analysis-of-covid19-public-discourse-on-twitter-implications-for-risk-communication/5CF2824263F23693F9AEFB4A5E56880A>
   7. <https://www.analyticsvidhya.com/blog/2021/02/sentiment-analysis-predicting-sentiment-of-covid-19-tweets/>
   8. <https://www.makeschool.com/mediabook/oa/tutorials/tweet-generator--data-structures---probability-with-python/analyze-word-frequency-in-text/>
   9. <https://towardsdatascience.com/covid-19-data-visualization-using-python-3c8bcfaeff5f>
   10. <https://www.kaggle.com/smid80/coronavirus-covid19-tweets>