PSG COLLEGE OF TECHNOLOGY, COIMBATORE 641004

Department of Computer Science and Engineering

19ZO02 - Social and Economic Network Analysis Report

Analyzing and Visualising Elon Musk's Tweets

By

Akshara P - 19Z305
Dhanavandhana K - 19Z311
Samyuktha A S K - 19Z343
Saranya K - 19Z345

BE CSE G2 (2019 - 2023)

Analyzing and Visualising Elon Musk's Tweets

Problem Statement:

Analyzing and visualizing the tweets made by Elon Musk.

The analysis done in this project are:

- ➔ Sentimental Analysis
  - The imported libraries of sentimental analysis categorize the emotion

    (positive, negative, neutral) of each tweet based on the words present in it.

  - The bar graph represents the number of positive, negative and neutral tweets.
- ➔ Visualize the frequent words
  - The most frequently used words in the tweets were displayed as a word cloud.
- ➔ Analyzing the Mentions
  - Initially the code is written to display all unique mentions made by Elon Musk.
  - Then the most frequently used top 10 mentions were plotted and displayed as a bar graph

    (x-axis: Top 10 mention names, y-axis: Number of frequency)
  - A statistical analysis is done on the number of mentions made in a month and plotted in the form of graph

    (x-axis: Months (Jan to Dec), y-axis: Number of mentions)

Dataset Description:

The dataset(data_elonmusk.csv) used for this analysis consists of Elon musk's tweets from the 2012 to 2017.There are totally 3218 rows and 5 columns which are defined as follows,

- **Row ID** – unique identifier for a row
- **Tweet** – Tweets posted by Elon musk in his twitter handle
- **Time** – Time at when the tweets were posted
- **Retweet from** – user name of the account from where Elon musk have retweeted his tweets
- **User** – user name of the account from where the tweets were posted (Elon musk in this case)

Dataset link - https://www.kaggle.com/code/antonaks/analysis-of-elon-musk-s-tweets/data

Tools Used:

- **Pandas**

  Pandas is an open-source library designed primarily for working quickly and logically with relational or labeled data. It offers a range of data structures and procedures for working with time series and numerical data.

- **Numpy**

  The Python package NumPy is used to manipulate arrays. Additionally, it has functions for working in the area of linear algebra, matrices, and fourier transform.

- **Matplotlib**

  Python's Matplotlib library provides a complete tool for building static, animated, and interactive visualizations.

- **Nltk**

  The NLTK standard Python package provides a wide range of NLP algorithms. it is the most popular library for NLP and computational linguistics.

- **WordCloud**

  Word clouds are a type of data visualization where the magnitude of each word represents its frequency or relevance in a textual representation. Using a word cloud, significant textual data points can be highlighted. Word clouds are often used for social network data analysis.

- **Vader**

  The lexicon- and rule-based sentiment analysis tool VADER (Valence Aware Dictionary and sentiment Reasoner) is customized precisely to sentiments expressed on social media.

- **Tokenize**

  The pre-processing of text, sentences, and information from the text utilizing modules and functions is handled by this object-oriented library. Splitting a phrase or sentence into tokens, also known as segments, is the process of tokenization.

- **Seaborn**

  On top of matplotlib, Seaborn is a data visualization library that is tightly connected with Python's pandas data structures. The core of Seaborn is visualization, which aids in data exploration and comprehension.

Challenges Faced:

- Splitting the Time column(which consisted of date,time in the same column) in the dataset into two separate columns as time and date columns was difficult.

- Displaying the tweets as a trend was not possible because the tweet column is of string type.

- Had trouble locating the precise sentiment analysis packages.

Contribution:

Name	Roll Number	Contribution
Akshara P	**19Z305**	Performing sentiment analysis in the tweet and plotting them as positive,negative and neutral in the graph.
Dhanavandhana K	**19Z311**	Analyzing the @mentions in the tweet and plotting the frequently used mentions in the tweet as a graph.
Samyuktha A S K	**19Z343**	Performing the exploratory data analysis and displaying the word cloud.
Saranya K	**19Z345**	Performing statistical analysis in the tweet and plotting the sum of mentions and sum of tweets per day.
Vasanthan M	**19Z359**	Preprocessing the dataset

Annexure I - Code:

Importing library:

```
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
import re
import string
import warnings
# warnings.filterwarnings('ignore')
from wordcloud import WordCloud
import nltk
nltk.download('vader_lexicon')
from nltk.sentiment.vader import SentimentIntensityAnalyzer
from nltk.sentiment.util import *
from nltk import tokenize
import matplotlib.dates as mdates
import seaborn as sns
from datetime import datetime
```

Data Pre-Processing:

```
# removes pattern in the input text
def remove_pattern(input_txt, pattern):
    r = re.findall(pattern, input_txt)
    for word in r:
        input_txt = re.sub(word, "", input_txt)
    return input_txt
# remove twitter handles (@user)
df['clean_tweet'] = np.vectorize(remove_pattern)(df['Tweet'], "@[\w]*")
df.head()
# remove special characters, numbers and punctuations
df['clean_tweet'] = df['clean_tweet'].str.replace("[^a-zA-Z#]", " ")
df.head()
# remove short words
```

```
df['clean_tweet'] = df['clean_tweet'].apply(lambda x: " ".join([w for w in x.split() if len(w)>3]))
df.head()
# individual words considered as tokens
tokenized_tweet = df['clean_tweet'].apply(lambda x: x.split())
tokenized_tweet.head()
# stem the words
from nltk.stem.porter import PorterStemmer
stemmer = PorterStemmer()

tokenized_tweet = tokenized_tweet.apply(lambda sentence: [stemmer.stem(word) for word in sentence])
tokenized_tweet.head()
# combine words into single sentence
for i in range(len(tokenized_tweet)):
    tokenized_tweet[i] = " ".join(tokenized_tweet[i])

df['clean_tweet'] = tokenized_tweet
df.head()
#split the time column into hour and date
df['date'] = df['Time'].astype(str)

# split date into 3 columns
df[['date_1','hour']] = df['date'].astype(str).str.split(expand=True)



#convert hour to string
df['hour']= df['hour'].astype(str)

#format hour column as date time
df['hour'] = pd.to_datetime(df['hour'], format='%H:%M:%S').dt.time
#format date column as date time
df['date_1'] = pd.to_datetime(df['date_1'], format='%Y-%m-%d')
```

```
#add weekday columns
df['day'] = df['date_1'].dt.weekday
df['day_name'] = df['date_1'].dt.day_name()
```

1. Sentimental Analysis:

```
from nltk.sentiment.vader import SentimentIntensityAnalyzer
from nltk.sentiment.util import *
from nltk import tokenize
sid = SentimentIntensityAnalyzer()
df['sentiment_compound_polarity']=df.clean_tweet.apply(lambda
x:sid.polarity_scores(x)['compound'])
df['sentiment_neutral']=df.clean_tweet.apply(lambda x:sid.polarity_scores(x)['neu'])
df['sentiment_negative']=df.clean_tweet.apply(lambda x:sid.polarity_scores(x)['neg'])
df['sentiment_pos']=df.clean_tweet.apply(lambda x:sid.polarity_scores(x)['pos'])
df['sentiment_type']=''
df.loc[df.sentiment_compound_polarity>0,'sentiment_type']='POSITIVE'
df.loc[df.sentiment_compound_polarity==0,'sentiment_type']='NEUTRAL'
df.loc[df.sentiment_compound_polarity<0,'sentiment_type']='NEGATIVE'
df.sentiment_type.value_counts().plot(kind='bar',title="sentiment analysis")
```

2. Exploratory Data Analysis :

```
words = " ".join([sentence for sentence in df['clean_tweet']])
wordcloud = WordCloud(width=800, height=500, random_state=42,
max_font_size=100).generate(all_words)
# plot the graph
plt.figure(figsize=(15,8))
plt.imshow(wordcloud, interpolation='bilinear')
plt.axis('off')
plt.show()
print(words)
```

3. Analysing the @mentions in the dataset

```
#function to extract the mentions from the tweet
```

```
def extract(mention):
    item_list = []
    for row in df['Tweet']:
        items = [tag.strip(mention) for tag in row.split() if tag.startswith(mention)]
        punct = [".","?","!",":","""","]","[",";",","]
        #remove punctuation from, it leaves us with a list of letters
        item_no_punct = [[l for l in item if l not in punct] for item in items]
        #joint the letters back into words
        items_formated = [''.join(item) for item in item_no_punct]
        item_list.append(items_formated)

    #turn a list of lists into one list
    items_list_all = [item for sublist in item_list for item in sublist]
    #turn into a set to chck uniques
    uniques = list(set(items_list_all))
    #count uniques
    number_of_unique_items = len(uniques)
    # frequency dict
    frequency_dict = {i: items_list_all.count(i) for i in items_list_all}
    # convert into df
    frequency_df = pd.DataFrame(frequency_dict.items(), columns=["word", "count"])
    #sort values
    sorted_frequency = frequency_df.sort_values(by='count',ascending=False)
    #get top 10 values
    top_10_df = sorted_frequency.head(10)
    top_10 = list(top_10_df['word'])
    class i_list:
      def __init__(self):
#           no_uniques, list_uniques, full_list
        self.no_uniques = number_of_unique_items
        self.list_uniques = uniques
        self.full_list = items_list_all
        self.sorted_frequency = sorted_frequency
        self.top_10 = top_10
```

```
        def how_many(self):
            return f"This set of tweets has {self.no_uniques} unique {mention}s."
        def list_top_10(self):
            return f"The top 10 {mention}s used in this set of tweets are: {self.top_10}."

    item_list = i_list()
return item_list
#plot top 10 most frequently used mentions
fig, ax = plt.subplots(figsize=(10,8))

labels = mention_frequency_df['word'][:10]
y = mention_frequency_df['count'][:10]
x = np.arange(len(labels))

_ = plt.bar(x, height=y)

# ax.set_xticklabels(labels)
plt.xticks(x, labels, rotation=45)
plt.title("Top 10 most frequently used mentions", fontsize=16)
plt.show()

#function to count the number of mentions in the tweet
def extractor(row, mention):
    words = [tag.strip(mention) for tag in row.split() if tag.startswith(mention)]
    punct = [".","?","!",":","'","]","[",";"]
    #remove punctuation from mentions, it leaves us with a list of letters
    no_punct = [[l for l in item if l not in punct] for item in words]
    #join the letters back into words
    words_formated = [''.join(item) for item in no_punct]
    return words_formated

df['mentions_list'] = df['Tweet'].apply(lambda x: extractor(x, '@'))

#clean column
```

```
df['mentions_list'] =
df['mentions_list'].astype(str).str.replace(']',").str.replace('[',").str.replace("'","")

#check
df['mentions_list'].head(2)
#create a  mention_df for further transformation
def m_count(row, h_or_m):
    c = row.count(h_or_m)
    return c

df['sum_mentions']= df['Tweet'].apply(lambda row: m_count(row,'@'))
mention_df = df[['date', 'hour', 'day', 'mentions_list','sum_mentions']]

# set date as index
mention_df['date'] = pd.to_datetime(df['date'])
mention_df_re = mention_df.set_index('date')
mention_sum_by_day = mention_df_re.resample('D').sum()
mention_sum_by_day.drop(columns="day", inplace=True)
tweets_per_day = mention_df_re.index.value_counts().sort_index()
print(tweets_per_day)

#join on index with the previous df
mention_and_tweet=mention_sum_by_day.join(tweets_per_day)

#rename the added column
mention_and_tweet = mention_and_tweet.rename(columns={'date':'sum_tweets'})

# turn nans to 0s & convert to interger
mention_and_tweet['sum_tweets'] =
mention_and_tweet['sum_tweets'].fillna('0').astype(int)

#check
mention_and_tweet.head()
tweet_sum = mention_and_tweet['sum_tweets'].sum()
```

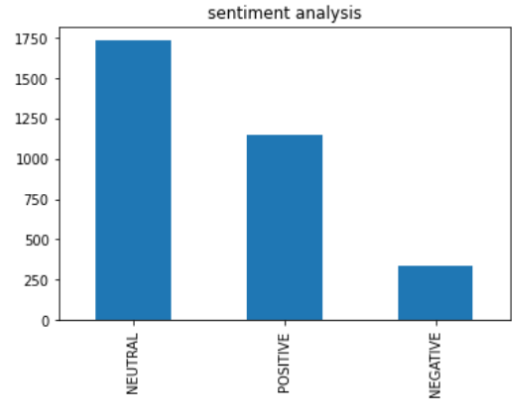
```
mention_sum = mention_and_tweet['sum_mentions'].sum()
ht_ratio = mention_sum/tweet_sum
print("Overall there has been", tweet_sum, "tweets sent in the time period and",
mention_sum,
    "mentions used. This gives a ratio of", round(ht_ratio,2), "which means there is an
average of just over 1 mention per tweet.")
# plot number of mentions and tweets per day

sns.set()

def plot_mention_by_day(variable_1, variable_2=None):
    fig, ax = plt.subplots(figsize=(16,8))

    #set xand y axes
    x = mention_and_tweet.index
    y = mention_and_tweet[variable_1]

    #plot
    ax.plot(x, y, label=variable_1)

    #add other plots if variables given
    if variable_2 != None:
        y1 = mention_and_tweet[variable_2]
        ax.plot(x, y1, label=variable_2)

    months = mdates.MonthLocator()
    month_fmt = mdates.DateFormatter('%m')

    ax.xaxis.set_major_locator(months)
    ax.xaxis.set_major_formatter(month_fmt)
    ax.tick_params(axis='x', rotation=45)

    ax.set_title("Mentions stats over time", fontsize=16)
    ax.set_ylabel('Number')
```

```
    ax.set_xlabel('Month')

    plt.legend(fontsize=14)
    plt.show()

plot_mention_by_day('sum_mentions', 'sum_tweets')
```

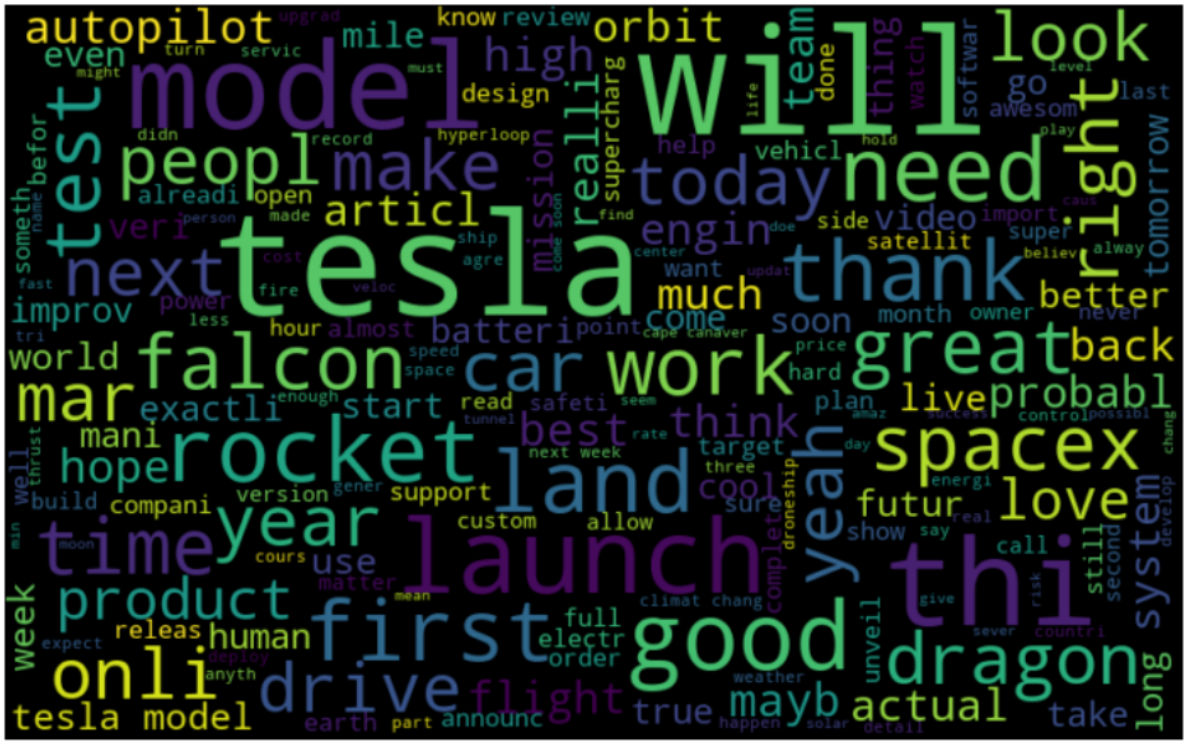
Annexure II - Output:

1. Sentimental Analysis

```
<matplotlib.axes._subplots.AxesSubplot at 0x7f3fcc302b10>
```

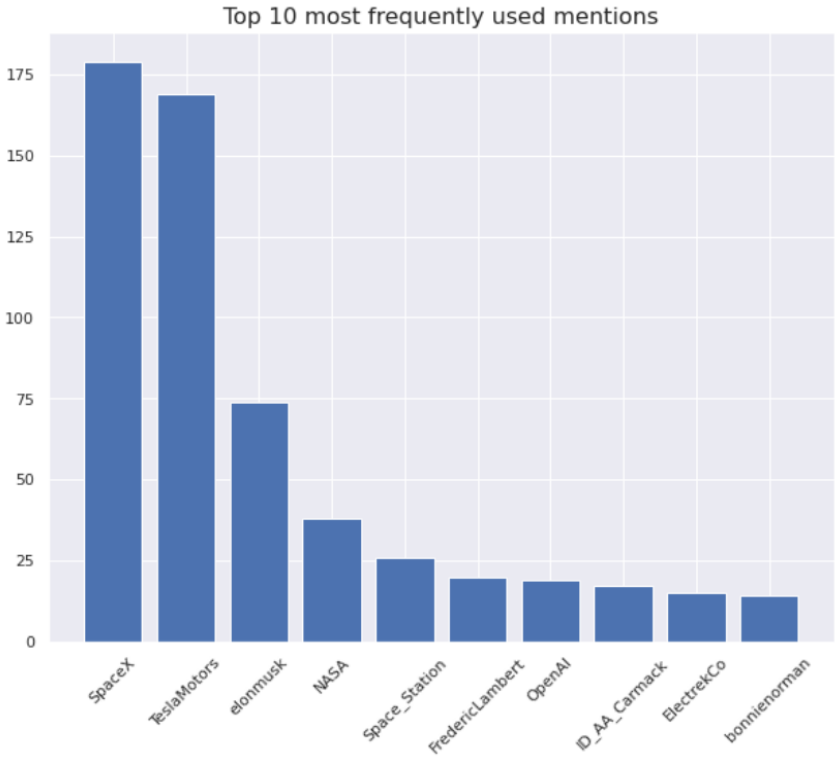


2. Exploratory Data Analysis (Word cloud)

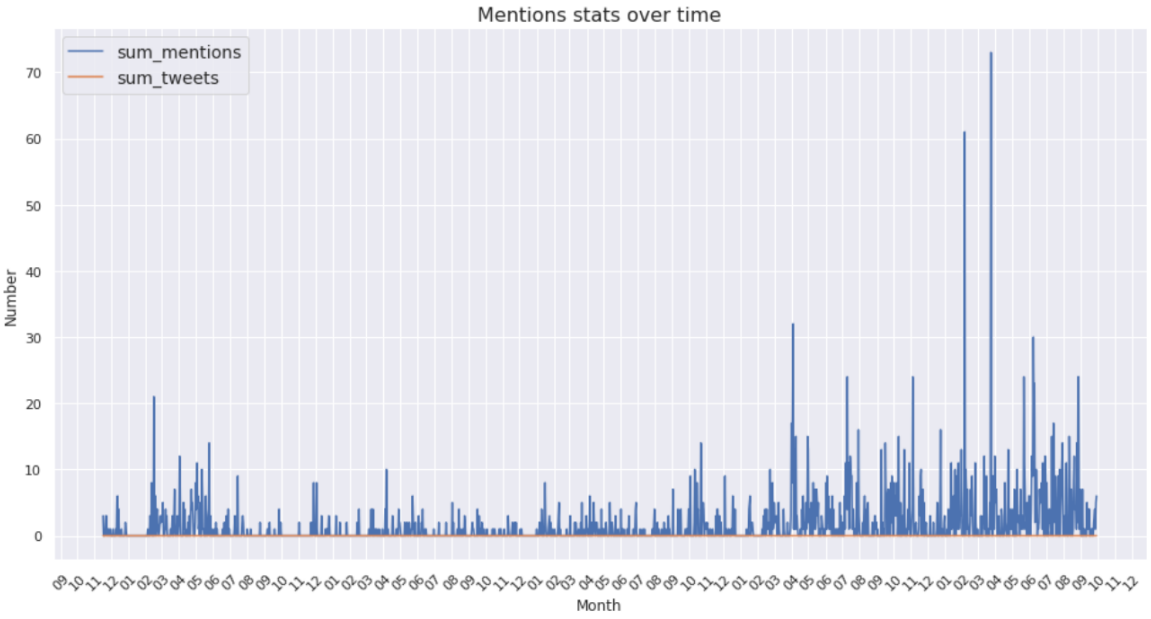


3. Analysing the @mentions in the dataset

[32]


Top 10 most frequently used mentions
175
150
125
100
75
50
25
0
SpaceX
TeslaMotors
elonmusk
NASA
Space_Station
FredericLambert
OpenAI
ID_AA_Carmack
ElectrekCo
bonnienorman

[38]


Mentions stats over time
sum_mentions
sum_tweets
Number
Month

References:

1.https://www.kaggle.com/code/antonaks/analysis-of-elon-musk-s-tweets/report
2. https://towardsdatascience.com/elon-musk-twitter-adf324120b3f
3.https://livecodestream.dev/post/detecting-the-sentiment-on-elon-musks-tweets-with-python/
4.https://www.visualcapitalist.com/a-decade-of-elon-musks-tweets-visualized/
5.https://inside-machinelearning.com/en/elon-musks-tweets-analysis-in-python-easy-ai-for-text/
6.https://betterprogramming.pub/detecting-sentiment-from-elon-musks-tweets-using-python-ec7820469ac0
7.https://medium.com/mlearning-ai/elon-musks-twitter-sentiment-analysis-with-transformers-hugging-face-roberta-49b9e61b1433
8.https://monkeylearn.com/blog/sentiment-analysis-of-twitter/
9.https://www.geeksforgeeks.org/twitter-sentiment-analysis-using-python/
10.https://www.analyticsvidhya.com/blog/2021/06/twitter-sentiment-analysis-a-nlp-use-case-for-beginners/

Plagiarism Screenshot: