An Analysis of Elon’s Twitter Takeover

By: Team 2 (Mostafa Omidi, Karunya Konathala, Jaeho Shin, and Meris Yates)

Prof. Maryam Heidari

AIT-526-DL1

George Mason University

12/10/2022

# Table of Contents

[Table of Contents](#_cz907wc8wafu)

[Abstract](#_vurwf0s99w72)

[Introduction](#_3oz5bkvgv4k6)

[Related Works](#_lhxpbm4vbndb)

[Objectives](#_9e9v346ylf4e)

[The Datasets](#_5r7gyqj0mar6)

[The System](#_aa1lytnu1f8f)

[Architecture](#_joa6t9n8hyht)

[Exploratory Data Analytics Approach](#_qlmu4199wm5n)

[Sentiment Analysis Approach](#_nz7te5vpxexl)

[Results](#_fkuln8siq7u0)

[Exploratory Analysis](#_ayjg0ri9rerm)

[Sentiment Analysis](#_newywlhj0q05)

[Analysis](#_6ucuwltccvgv)

[Conclusion](#_ybktksx0dn3t)

[Future Work](#_ijsepysqsnne)

[References](#_a82thob3qggn)

[Appendix A: Text Scraping and Data Preprocessing](#_t66upomuoo1x)

[Text Scraping](#_pfsncuaxvf4x)

[Data Preprocessing for Exploratory Analysis](#_y2s7d3khdd0u)

[Data Preprocessing for Sentiment Analysis](#_egc21kch71p2)

[Baseline Solution Preprocessing](#_4ebm91rrr7tf)

[Updated Solution Preprocessing](#_smsu6mt45pys)

[Baseline Sentiment Analysis Experiment and Results](#_62zpf0crgwig)

[Baseline Error Analysis](#_8gl3vc2e9bxz)

[Appendix B: Datasets](#_h8mrrtygnt6d)

[#ElonMuskTwitter Dataset](#_4ww4k4hgf5u)

[#TwitterTakeover Dataset](#_g17pg9lebgm5)

[#ThankYouElonMusk Dataset](#_1g7iugrsicxo)

[#RIPTwitter Dataset](#_ajh2yxczs0cn)

# 

# Abstract

Social media receives more and more attention nowadays. Public and private opinions about a wide variety of subjects are expressed and spread continually via numerous social media. Twitter is one of the social media that is gaining popularity. Twitter offers organizations a fast and effective way to analyze customers' perspectives toward the critical to success in the marketplace. Developing a program for sentiment analysis is an approach to be used to computationally measure customers' perceptions. The goal of tweet sentiment analysis is to find the positive, negative, or neutral sentiment part in the tweeter data. Sentiment analysis deals with identifying and classifying opinions or sentiments expressed in source text. Social media is generating a vast amount of sentiment rich data in the form of tweets, status updates, blog posts etc. Sentiment analysis can help any organization to find people's opinions of their company and products. Twitter sentiment analysis is difficult compared to general sentiment analysis due to the presence of slang words and misspellings. The maximum limit of characters that are allowed in Twitter is 140. We have applied sentiment analysis on Elon Musk’s twitter take over Tweets. Our model takes input tweet, sentiment, and output selected text starting and ending in input tweet.

**Keywords:** natural language processing, sentimental analysis, Natural Language Tool-Kit (NLTK), social media analysis

# Introduction

Information travels extremely quickly nowadays and one of the easiest places to see that happen is the social network service Twitter. Twitter allows people from all over the globe to communicate in a way that was never seen before, called tweets. For the past decade and a half, Twitter has been a hotbed of activity including but not limited to social justice, journalism, connecting with friends and family, and rapid reporting of disasters both natural and otherwise. Within the past two months, Twitter has very publicly been turned upside down by its new CEO with the removal of 2FA, high employee turnover, scaring advertisers off by releasing controversial statements, and reinstating previously banned figures. A tumultuous event such as this is ripe for analysis as anyone and everyone has opinions on the subject.

Despite Twitter natural language processing analyses being fairly easy to find online, it is no simple feat. Using a computer to analyze human language is quite complex because computers do not think the same way humans do and teaching one how to analyze the context behind certain words or phrases has been the subject of research for decades. In addition, analyzing text from social media is often much more complicated due to how different people express themselves. Our goal is to analyze the English language reactions on Twitter to Elon Musk’s recent ascension to CEO by focusing on tweets with specific hashtags relating to his takeover.

# Related Works

As social media applications are getting popular, there is more information exchange taking place where people share more data publicly via opinions, photos, media, Etc. This information can be used to make strategic decisions by the companies to provide better user insights. Among all the social media sites, Twitter plays a huge role in affecting society; our research is working on the sentiment analysis of Twitter users based on the hashtags, which are a pretty popular tool for segregating the data. The research work done by (Patel & Passi, 2020) is similar to our project; it aids in classifying and investigating the behavior and approach of the customers regarding brands, products, events, company and customer services. They performed sentiment analysis along with opinion mining and the validation & evaluation was done by sentiment analysis. This depended on the syntactical tree but did not use words or concepts that have positive, negative or neutral meanings. The approach the authors used was to process the data using a lexicon-based method and use machine learning (ML) algorithms (coupled with an understanding of which ML model is better suited for sentiment analysis on social media) for the rest. Their findings show that the naive Bayes followed by KNN perform better than the Random forests and SVM models (Patel & Passi, 2020).

Another similar research work on “Sentiment analysis of Twitter data” utilizes a similar design and implementation process to ours. First, they used the Twitter testing datasets by connecting to the Twitter API. Then, they performed the data preprocessing, comparing each word with the sentiment dictionary and classifying them as positive and negative counts. Finally, the result percentage decided the polarity (OSF, 2022). The “Sentiment analysis of twitter data: A survey of techniques” journal provides us a survey and a comparative analysis of existing techniques for opinion mining like lexicon based and ML approaches along with the evaluation metrics which is a similar work of (Patel & Passi, 2020) but along with that they also provide research on twitter data streams (Kharde & Sonawane, 2016).

# Objectives

In this project, we endeavored to do the following:

1. Determine overall sentiment (determined as a % out of 100) of Elon Musk’s takeover of Twitter.
2. Perform sentiment analysis on the subsets of the dataset comparing sentiment among different hashtags (two neutral, one positive, and one negative).
3. Perform any other analysis suitable to the data (word sense disambiguation, named entity recognition, etc).
4. Visualize any interesting results or trends that are unearthed during this analysis.

# 

# The Datasets

The datasets are generated by scraping the social networking system Twitter for Tweets based on specific hashtags (AKA, entity extraction). For this particular analysis, we elected to use two hashtags that are seemingly neutral (#ElonMuskTwitter and #TwitterTakeover), one hashtag that is seemingly positive (#ThankYouElonMusk) and one hashtag that is seemingly negative (#RIPTwitter). No calculations or analysis was performed when choosing which hashtag to use. We chose based on our own perceived sentiment of the hashtag itself.

Figure 1 below is an example of one of the datasets, and each hashtag has the same format. The datasets have five columns such as "User" (String), "Date Created" (Date), "Number of Likes" (Numeric), "Source of Tweet" (String), and "Tweet" (String). The column “User” is the username of the Twitter account that posted the tweet. It is a combination of real names and pseudonyms. The “Date Created” attribute is the date that the tweet was posted. The values in “Date Created” are mostly from 2022, but also have some tweets from 2021 and 2019. The “Source of Tweet” column is which platform the user used to post the tweet. For the majority of records, it is common Twitter applications like Twitter Web App, Twitter for Android, and Twitter for iOS. However, some of the sources explicitly say “bots” in the label. Those records will need to be removed before analysis.

### 

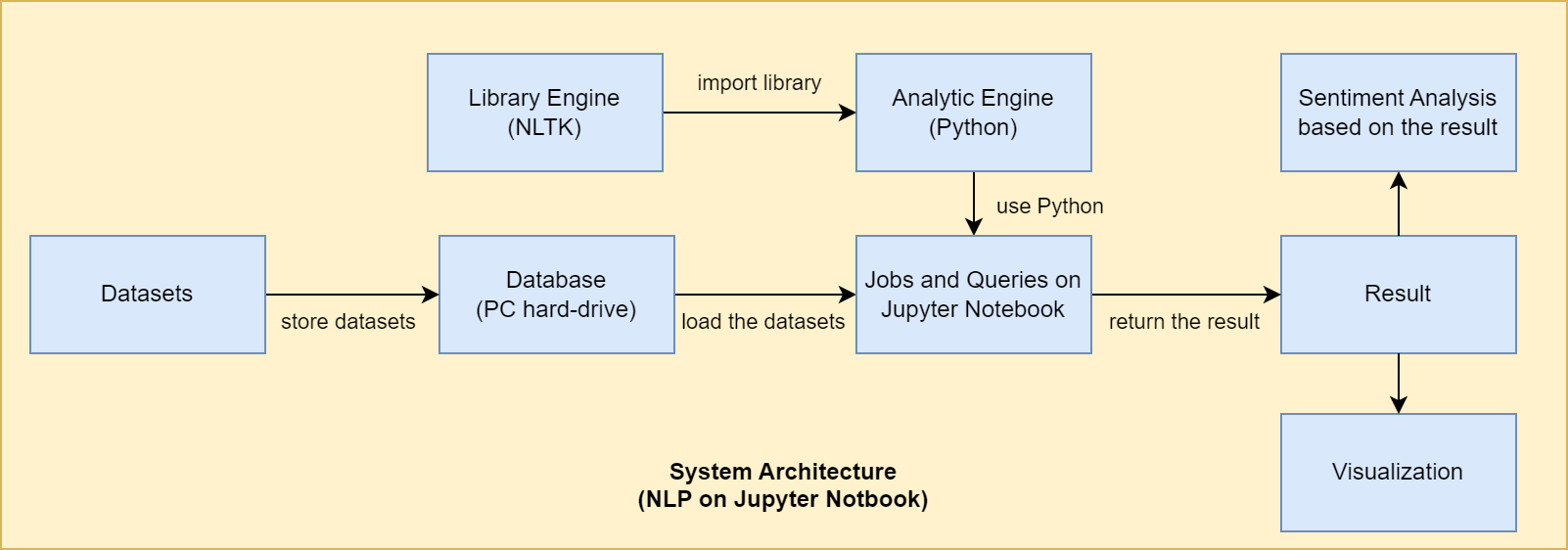
*Figure 1: a subset of the #ElonMuskTwitter dataset*

The number of records vary for each hashtag dataset, the details of which can be found in [Appendix B: Datasets](#_h8mrrtygnt6d). The datasets needed further data preprocessing, which can be found in [Data Preprocessing for Exploratory Analysis](#_y2s7d3khdd0u) in Appendix A.

# The System

## Architecture

Figure 2 is the diagram of the system architecture. First, datasets are created by saving the tweets of the hashtags selected for the research using the Twitter API. Then we save the datasets to the PC hard drive and then load the datasets in the Jupyter Notebook. After that, we load Python's NLTK library for natural language processing. After processing the steps necessary for NLP in the Jupyter Notebook, we obtain the results required for sentiment analysis. Finally, sentiment analysis is conducted with the results and visualizations are made to help understand the results.

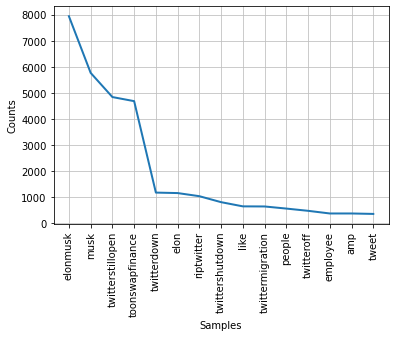


*Figure 2: the proposed system architecture*

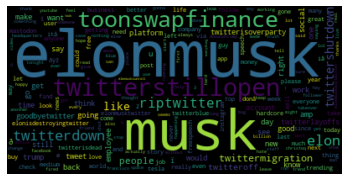
## Exploratory Data Analytics Approach

We will preprocess the tweets prior to analyzing. First, for smooth, natural language processing, we will convert all text to lowercase and remove all punctuation marks using the RegexpTokenizer function of the NLTK library. Also, since stopwords interfere with sentiment analysis, we will remove stopwords from the text using nltk's stopwords function and use WordNetLemmatizer to change all the words to their original form. Next, we will show the frequency of words through the FreqDist function of the NLTK library in the preprocessed dataset. Based on the result from the frequency of the words, linear graph charts were created. In addition, word clouds created using the Word Cloud library will show which words were the most frequently used for that hashtag. Finally, we will conduct sentiment analysis by judging whether the meanings of the words are negative or positive.

The below images are examples of data visualizations from the data analysis of the dataset with hashtag “#twittertakeover”. Those visualizations show the frequency of the words that are mentioned by users. More visualizations for other hashtags will be created as well during the further research.



*Figure 3: a frequency distribution of the 15 most common words in the dataset #twittertakeover*



*Figure 4: a word- cloud of the dataset #twittertakeover*

## Sentiment Analysis Approach

Before performing sentiment analysis, we preprocessed the datasets as demonstrated in the [Preprocessing](#_smsu6mt45pys) section for the updated solution in Appendix A. We chose to separate the preprocessing for sentiment analysis from the exploratory analysis because of the type of sentiment analysis we wanted to perform. Our data did not automatically have predetermined sentiment results like most datasets on Kaggle or wherever, so we could not train a custom made algorithm without manually determining the sentiment for thousands of tweets. This would have been an inefficient use of our time given the constraints we had to complete this analysis during this semester. With that in mind, we decided to use the VADER tool in the NLTK package.

VADER stands for **V**alence **A**ware **D**ictionary and s**E**ntiment **R**easoner and is a lexicon-based and rule-based sentiment analysis tool that is especially useful for determining sentiment in social media (Hutto & Gilbert, 2014). This tool focuses on the intensity (i.e., how passionate are the words) and polarity (i.e., how positive/negative is the text) of statements to determine sentiment. VADER calculates this by referencing a mega-dictionary of common sentiment word-banks like Linguistic Inquiry and Word Count (LIWC), Affective Norms for English Words (ANEW), and the General Inquirer, and and independent calculation the sentiment rating for each word. This dictionary (called ‘vader\_lexicon.txt’) houses 7,500 words that were rated by 10 independent humans from -4 (extremely negative) to +4 (extremely positive) and whose standard deviation was less than 2.5 (Hutto & Gilbert, 2014). The average rating is considered the score or valence of the word. The total score for the text is called a compound score and is calculated by adding together the valence scores for each word and then normalizing between -1 (extreme negative) and +1 (extreme positive) (Hutto & Gilbert, 2014). VADER not only takes into account words, but includes purposeful capitalization, emoticons/emojis, and punctuation in its calculations, which is why it’s incredibly helpful for social media analyses.

Using VADER, we will perform sentiment analysis on each individual tweet in the four datasets and consider the average compound score to be the overall sentiment for each hashtag.

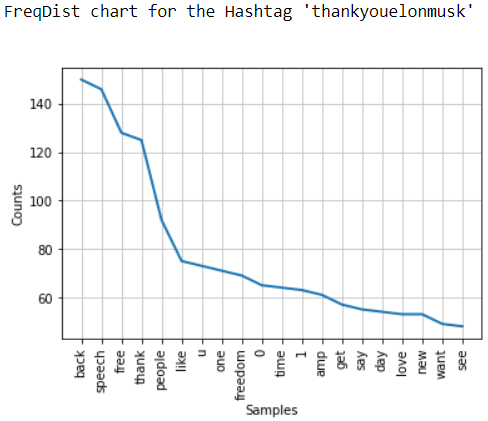
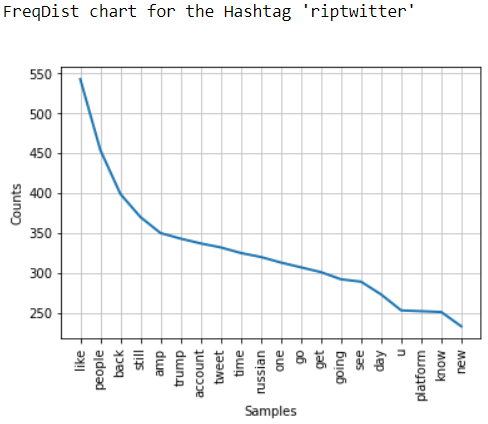
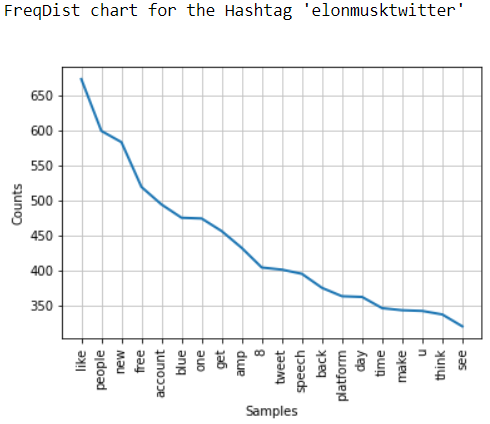
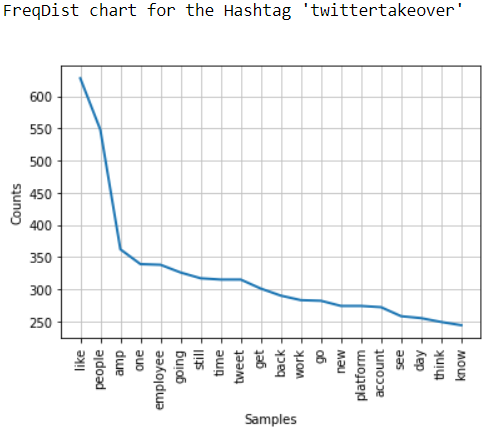
HW Development Platform  
Processor: Intel® Core™ i7 at 1.80 GHz or Apple M1  
Operating system: Windows® 11 or MacOS Ventura 13.0.1  
Installed RAM: 16 GB  
Dedicated Graphics Card: Nvidia Quadro P520 4GB

SW Development Platform  
Platform for Execution: Jupyter Notebook or JupyterLab  
Storage Platform: Personal Computer’s Hard-drive  
Querying Solution: Python  
Analytics and Visualisations: Preprocessing using Pandas library,  
Natural Language Processing using NLTK library,  
Visualization using WordCloud library  
Python Libraries: NLTK, SpaCy, snscrape, Re, String, Wordcloud, Pandas, CSV, en\_core\_web\_sm, collections

# Results

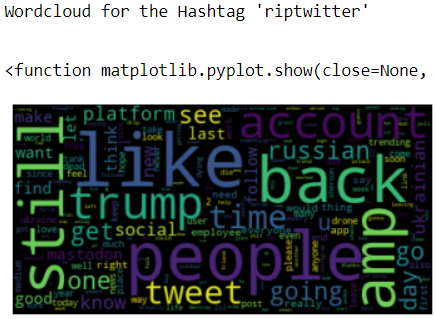
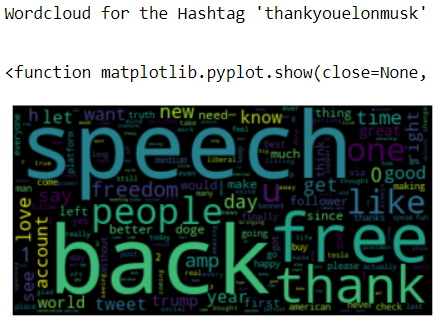
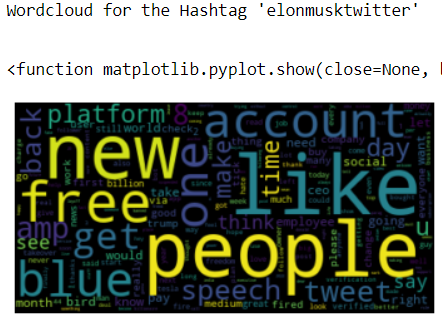
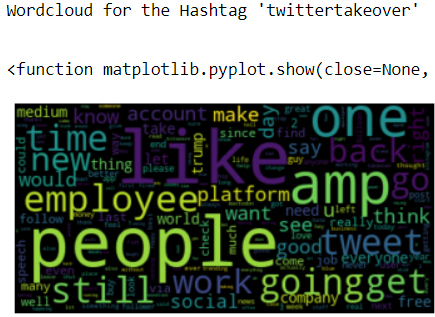
## Exploratory Analysis

Since exploratory analysis is required to get the best results, we have conducted two kinds of analyses for this research, exploratory analysis, and sentiment analysis. In the exploratory analysis, we wanted to get a feel for the different datasets and see how they differed from each other. The preprocessing for this analysis is explained in detail in the [Preprocessing for Exploratory Analysis](#_y2s7d3khdd0u) section in Appendix A. We applied all the functions we have learned from lab1, lab2, and lab3 to explore the datasets and have good analytical ideas for sentiment analysis. For the exploratory analysis, we first did data preprocessing for the datasets. Since there were many bot tweets in our datasets, we removed all the bot tweets from the datasets. Next, we removed all the punctuation, hashtags, and mentions that can distract our analysis using the re.sub function from RE library. After that, we completed the rest of the preprocessing and made the preprocessed datasets ready for analysis.

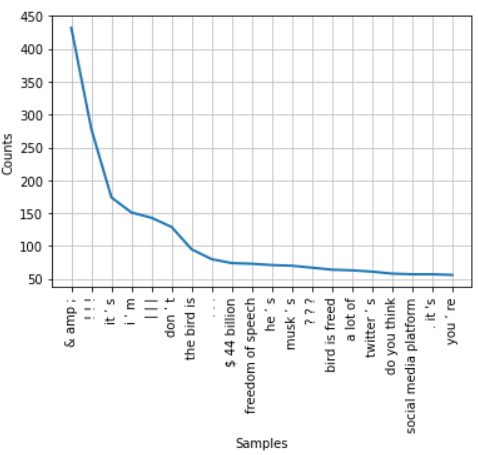
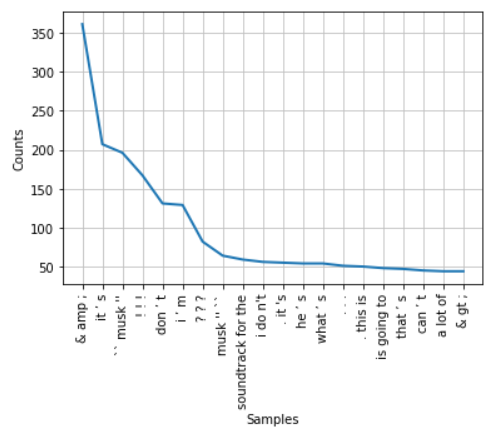
In the main code for the analysis, we first made the frequency distribution charts and word clouds for the datasets which we had learned from lab1. In order to make those charts, we divided the text by sentences and words using the nltk.tokenize function from NLTK library. Also, all the punctuation and stopwords were removed as well using functions from the NLTK library. Lastly, lemmatization was processed by NLTK’s lemmatization function to convert the words to original forms for better results on frequency distribution charts. The frequency distribution chart and word cloud for each dataset are shown below.  


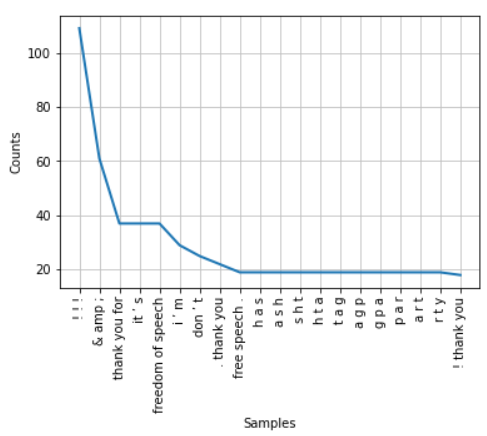
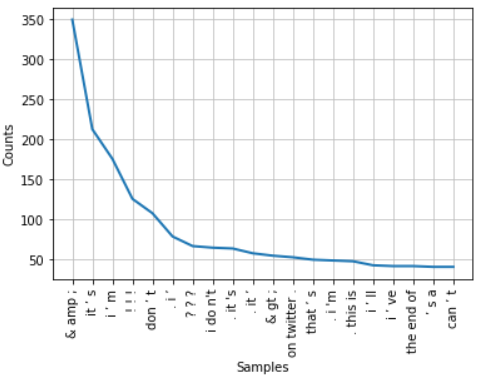
*Figure 5: Frequency distribution charts for each hashtag*

In the second part of the main code, we applied functions we had learned from lab2. We scored the sentences which were tokenized in the previous step and made a summary based on sentence, word, and percentage for each dataset. However, we have found that the scores and summaries do not really help our sentiment analysis because the datasets contain too many tweets and irrelevant texts in them. As we do not have to score and summarized the tweets for the sentiment analysis, we moved forward to N-gram charts. For the N-gram charts, we generated a 3-gram chart for each dataset using nltk.ngram function. We were able to check what phrases were mostly used in the tweets for each hashtag through the 3-gram charts. The 3-gram chart for each dataset(hashtag) is shown below.



*Figure 6: word clouds for each hashtag*

3-gram chart for the hashtag ‘twittertakeover’ 3-gram chart for the hashtag ‘elonmusktwitter’  


3-gram chart for the hashtag ‘riptwitter’ 3-gram chart for the hashtag ‘thankyouelonmusk’  


*Figure 7: 3-gram charts for each hashtag*

In the last part of the main code, we applied functions we had learned from lab3. We processed the same steps that we did previously for punctuation removal, stopwords removal, and lemmatization. After that, we applied named entity recognition(NER) to our datasets using en\_core\_web\_md function and displacy function from spaCy library. We could successfully identify all the entities and see the overview of the results in NER. However, we decided not to use NER function in the sentiment analysis because we do not have to identify the entities in the datasets, and it can confuse the results in the sentiment analysis.

In conclusion, through the exploratory analysis, we obtained insight into the datasets and got good analytical ideas of what processes are needed for the sentiment analysis. We found some issues that we need to fix for further analysis. First, simply running the process for the datasets takes much time since we have four datasets to run. In order to fix the problem, we have to write the code for the sentiment analysis efficiently to make the code runs fast. Also, as the datasets contain a lot of identical tweets, it generates a bad N-gram charts with meaningless phrases. Extra work on the removal of unnecessary words is needed to get better results. Finally, we will mostly use the same steps that we used in the exploratory analysis for the preprocessing part in the sentiment analysis. In addition, during the sentiment analysis, we will use SentimentIntensityAnalyzer function from NLTK library to accurately calculate the sentiment in the tweets such as negativeness, positiveness, or neutral.

## Sentiment Analysis

After preprocessing, we performed sentiment analysis on the four different datasets using the VADER tool and its function polarity\_scores as shown below in Figure 8.



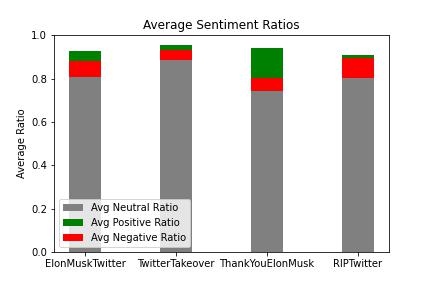
*Figure 8: average sentiment analysis results using VADER*

The polarity\_scores function outputs a dictionary of four elements: ratio of words with positive sentiment, ratio of words with negative sentiment, ratio of words with neutral sentiment, and the compound score. The compound score is explained in the approach section as being the “true” sentiment score and it is the sum of the valence scores of each nonneutral word in the tweet. According to (Hutto & Gilbert, 2014), if the compound score is > 0.05 the overall sentiment intensity is considered positive, if the compound score is between >= -0.05 and <= 0.05 the sentiment intensity is considered neutral, and if the compound score is < -0.05 the sentiment intensity is considered negative.

| **Hashtag** | **Avg Negative Ratio** | **Avg Neutral Ratio** | **Avg Positive Ratio** | **Avg Compound Score** | **Overall Sentiment and Score** |
| --- | --- | --- | --- | --- | --- |
| #TwitterTakeover | 0.0447 | 0.8864 | 0.0686 | 0.0460 | Neutral |
| #ElonMuskTwitter | 0.0721 | 0.8087 | 0.1190 | 0.0882 | Positive |
| #ThankYouElonMusk | 0.0605 | 0.7423 | 0.1970 | 0.2529 | Positive |
| #RIPTwitter | 0.0927 | 0.8022 | 0.0105 | 0.018 | Neutral |

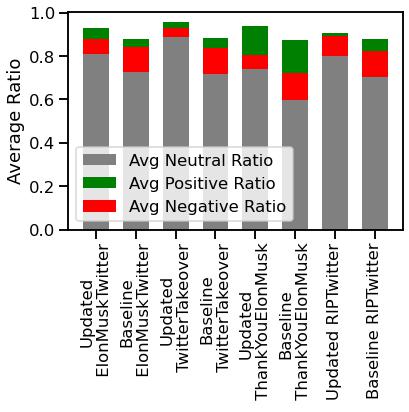
*Table 1: average sentiment analysis results using VADER*

Table 1 above and Figure 9 below gives the ratios and compound scores for each dataset analyzed in this report. Note that the total length of the stacked bars in the graphs below do not always add up nicely to 1.0 because they are the ratios of integers.



*Figure 9: average sentiment ratios using VADER*

In comparison, the next figure shows the sentiment ratios of the updated results compared to the baseline results. The baseline solution fully preprocessed the datasets by converting everything to lowercase, removing stopwords, punctuation, URLs, and tokenizing each word. Doing this type of preprocessing removed much of the things that VADER looks for when determining intensity (“Is a word purposefully capitalized compared to the others around it?”, “Is there excessive punctuation?”, etc. are some of the points VADER analyzes). (Orbiteers, 2019) has shown that removing these things by fuller preprocessing the text negatively affects VADER’s accuracy, so it’s interesting to note the differences between the updated so;ution and the baseline.



*Figure 10: average sentiment ratios using VADER*

## Analysis

The results in the results table (Table 1) are very different from our baseline solution. However, they are still more positive than expected, even the hashtag we thought would have negative sentiment. Even more interesting is the sentiment analysis figure comparing the updated results to the baseline results. It seems like VADER classifies text without punctuation, stopwords, and all lowercase as more negative than not. This could also be due to a known issue with VADER where it is less accurate with larger strings like paragraphs (Hutto & Gilbert, 2014). A few of these issues were delved into further in the [Baseline Error Analysis](#_8gl3vc2e9bxz) section.

There are many reasons why the data is classified as more positive than expected. It could be that the dataset we scraped is actually a more positive snapshot of the sentiment. It also could be that there are still other things VADER takes into account that we overlooked. There’s a possibility that the words with negative sentiment used in the tweets aren’t intense or passionate enough for VADER to classify it as negative. Another point VADER might be missing is sarcasm detection, which is prominent on Twitter. It’s also possible our preprocessing was not as good as it needed to be to properly apply VADER to the datasets.

Another point to take into account with our results is that our results are only good for the date range of the data we gathered. Twitter is a transient place where opinions move and change quickly, so while the data we collected shows a certain sentiment, it might not be the same sentiment of the same hashtag if we collected the data a couple of weeks before or after.

# Conclusion

In conclusion, through this research, we demonstrated that sentiment analysis of the text can be done through natural language processing. We learned that preprocessing for datasets is significant in producing good results. In addition, we learned that in the timeline of the data we collected the overall sentiment for Elon Musk’s Twitter takeover is more positive and neutral than negative. At least, our data snapshot of this event leans that way. We also learned that VADER works better for smaller chunks of test as opposed to a novella-length string. Our results could be true or the data might not tell the full story, considering VADER’s limitations and the fact that Twitter is a platform where users regularly communicate using phrases with double meanings. Further research is needed to delve more into the issues presented in this report.

## Future Work

There are a few things that can be done to improve upon these results. One idea is to analyze even more hashtags to get a fuller picture. We only looked at four hashtags that we specifically chose because of our own perceived sentiment of what they were instead of analyzing all data Twitter holds on this subject.

In addition, one could train different machine learning algorithms on similar Twitter data and analyze which algorithms perform the best when classifying this data. This option might have a higher up-front cost in terms of time because the researcher would need to manually classify each tweet if they didn’t want to take VADER’s output as fact.

# 

# References

George Mason Federated Login Service. (2022). Gmu.edu. <https://ieeexplore-ieee-org.mutex.gmu.edu/document/9121057>

Hutto, C. J., & Gilbert, E. E. (2014). VADER: A Parsimonious Rule-Based Model for Sentiment Analysis of Social Media Text. *Eighth International Conference on Weblogs and Social Media (ICWSM-14)*. <https://ojs.aaai.org/index.php/ICWSM/article/view/14550/14399>, <https://github.com/cjhutto/vaderSentiment>

JustAnotherArchivist. Snscrape. (2022). GitHub. Retrieved November 30, 2022, from <https://github.com/JustAnotherArchivist/snscrape>

Kharde, V., & Sonawane, S. (2016). Sentiment Analysis of Twitter Data: A Survey of Techniques. International Journal of Computer Applications, 139(11), 975–8887. <https://arxiv.org/pdf/1601.06971.pdf>

Mesevage. (2021, May 24). What Is Data Preprocessing & What Are The Steps Involved? MonkeyLearn Blog. Retrieved December 5, 2022, from <https://monkeylearn.com/blog/data-preprocessing/>

Orbiteers. (2019, March 21). *When not to Lemmatize Remove Stop Words Text Preprocessing*. ModelOp. <https://www.modelop.com/blog/when-not-to-lemmatize-or-remove-stop-words-in-text-preprocessing/>

OSF. (2022). Sentiment analysis of twitter data. [Https://Osf.io](https://osf.io); Open Science Framework. [https://www.google.com/url?sa=t&rct=j&q=&esrc=s&source=web&cd=&ved=2ahUKEwji8u6T3Nn7AhWOcGwGHaONA7UQFnoECAwQAQ&url=https%3A%2F%2Fosf.io%2F6xc4y%2Fdownload&usg=AOvVaw3X7RGlGzASNPuPEzxyR7y6](https://osf.io/6xc4y/download)

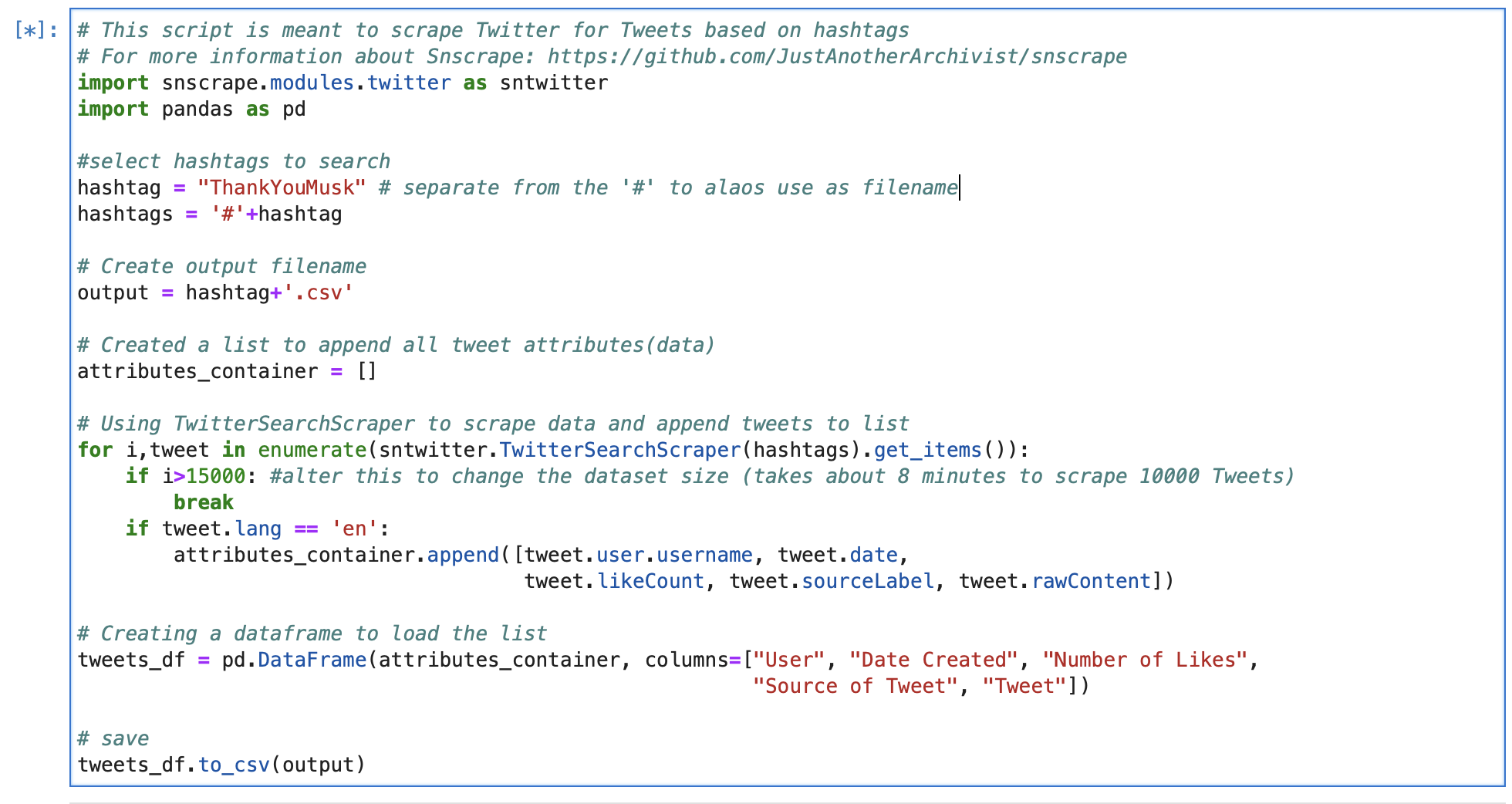
Patel, R., & Passi, K. (2020). Sentiment Analysis on Twitter Data of World Cup Soccer Tournament Using Machine Learning. IoT, 1(2), 218–239. <https://doi.org/10.3390/iot1020014>

Srinidhi, S. (2021, December 12). Understanding Word N-grams and N-gram Probability in Natural Language Processing. Medium. <https://towardsdatascience.com/understanding-word-n-grams-and-n-gram-probability-in-natural-language-processing-9d9eef0fa058>

# 

# Appendix A: Text Scraping and Data Preprocessing

## Text Scraping



*Figure 11: the tweet scraping script using snscrape package and JupyterLab/Jupyter Notebook*

The above script was used to scrape the four datasets used in this analysis one hashtag at a time. This script takes input of a hashtag (minus the octothorpe), searches Twitter for all English language results relating to the input, and extracts all information specified in the *attributes\_container* as the output dataset. This process will automatically iterate over the number of tweets specified in the if statement “if i> 15000”. 15,000 was chosen arbitrarily as a number large enough to scrape a large enough dataset for useful analysis. All datasets used in this analysis are approximately 10,000 records or less with the smallest being 1261 records.

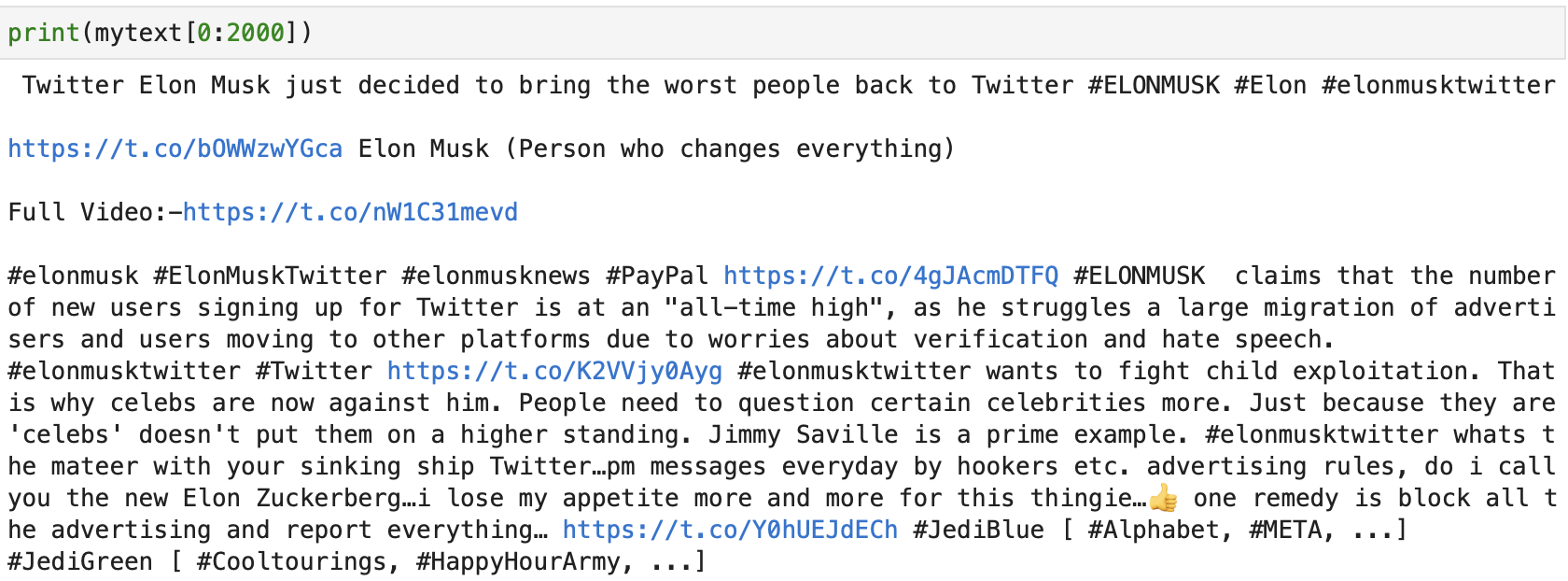
Snscrape was chosen over packages like Twint or Tweepy for its ease in collecting a massive amount of tweets from any point in time without having to jump through hoops for special access to Twitter’s API (*JustAnotherArchivist*, 2022). Both Tweepy/Twint and anything below Academic or Enterprise level access to the Twitter API only allow access to the most recent 7 days’ worth of tweets, which is not helpful unless you’re planning far enough in advance to scrape data as events occur in real time. One downside of this particular package is that there is no documentation for the python library. The examples in the README file on GitHub are for using snscrape in the terminal.

## Data Preprocessing for Exploratory Analysis



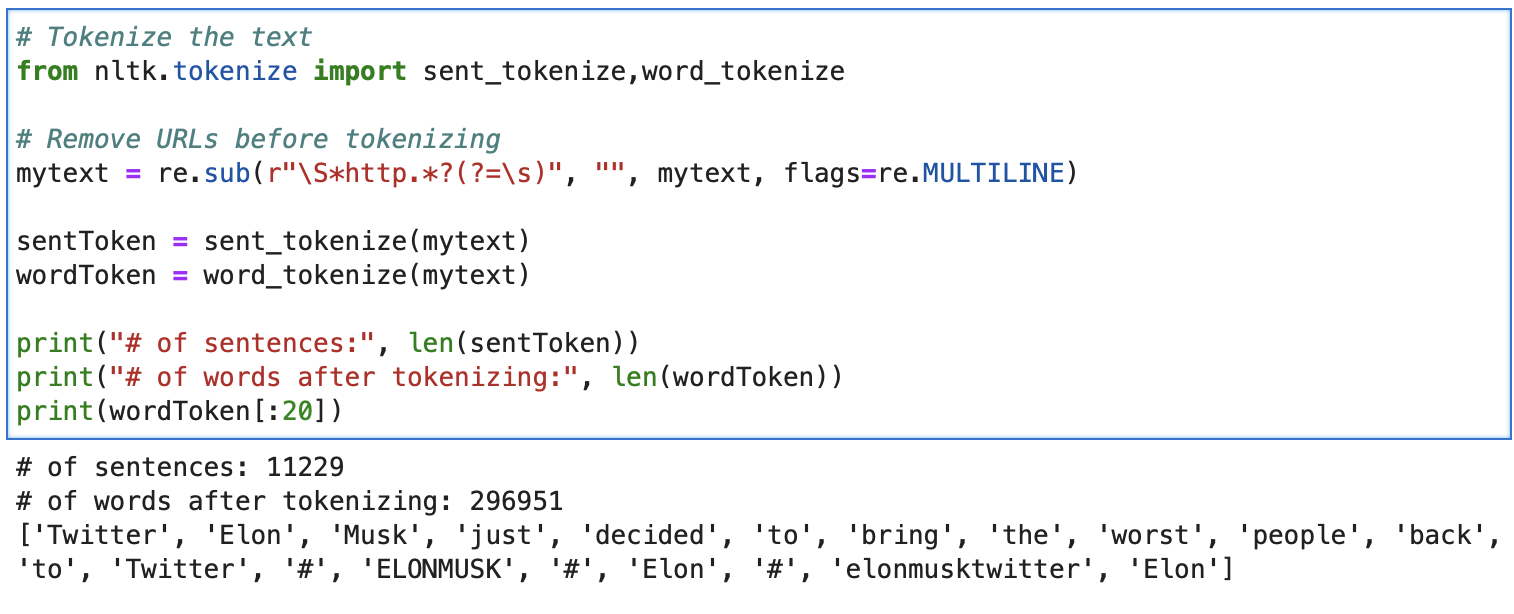
*Figure 12: the script to read in data using csv package in JupyterLab/Jupyter Notebook*

Once the data was scraped from Twitter, we read in each hashtag separately to perform exploratory analysis. The tweets were concatenated into a string as can be seen in Figure 13 below.

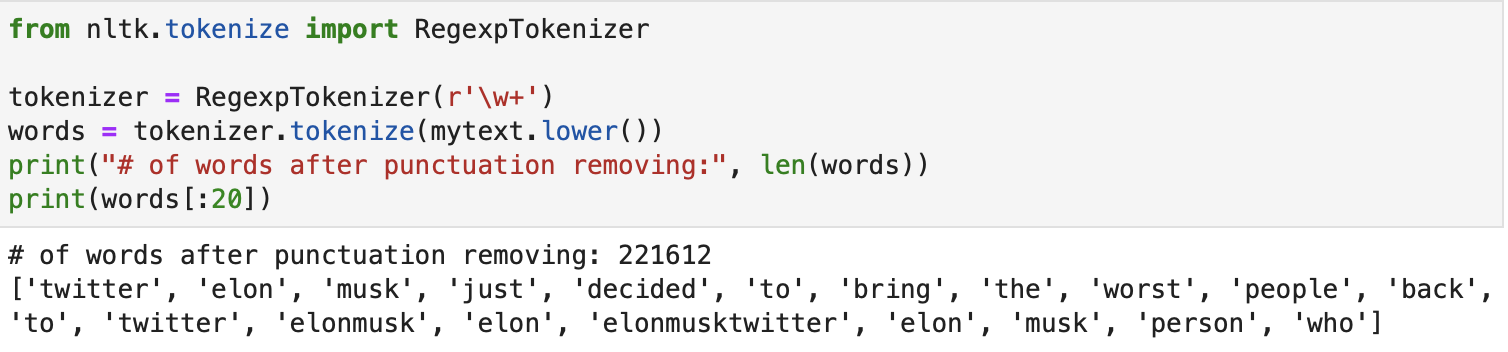


*Figure 13: a subset of the tweets in the #TwitterTakeover dataset*

The text was then tokenized by breaking into smaller chunks. Doing this helps make it easier for computers to understand human language because it’s much easier to understand a single word than a novel. We performed both word tokenization and sentence tokenization on the text (see Figure 14) before deciding to use a tokenizer called RegexprTokenizer that was more suited to the data (Figure 15). RegexprTokenizer took into account regular expressions and special characters like octothorpes (#) that are especially common in social media data.

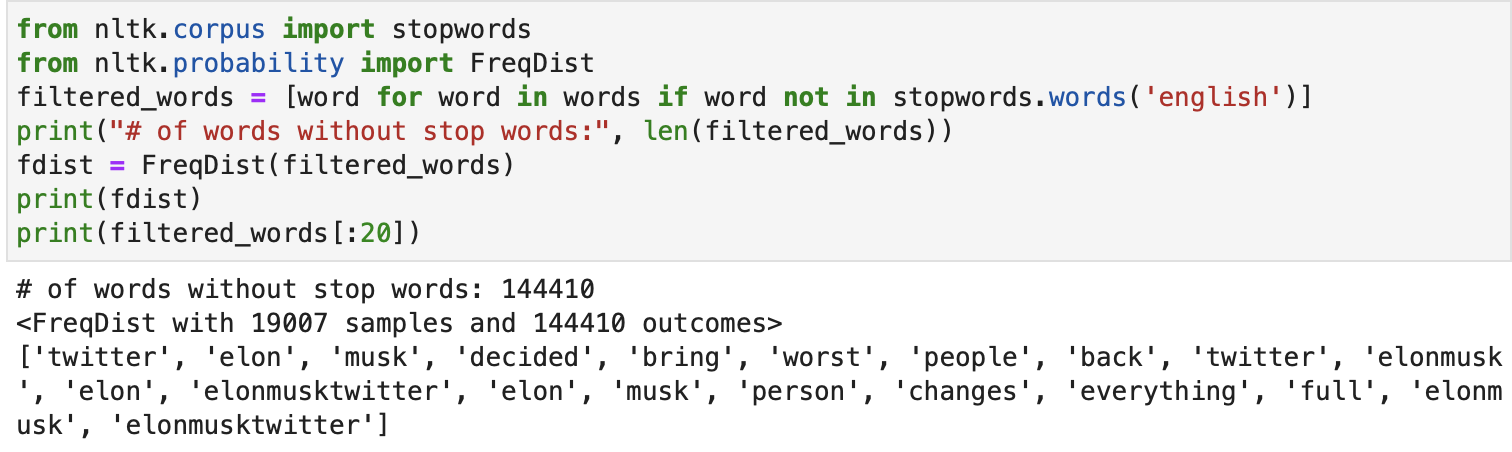
**

*Figure 14: tokenizing text with word\_tokenize and sent\_tokenize from nltk package in JupyterLab/Jupyter Notebook*



*Figure 15: tokenizing text using RegexprTokenizer from nltk in JupyterLab/Jupyter Notebook*

Tokenizing the text is just the first step in preprocessing the data. The output in the above figure still has a lot of filler words, called stopwords, with little meaning besides providing context to nouns, like “just” or “to” or “the”. Removing these does not really affect understanding the meaning of the text, as can be seen in the figure below. “Elon Musk decided bring worst people back twitter” makes just as much sense as “Elong Musk decided to bring the worst people back to twitter”, it’s just slightly less eloquent.



*Figure 16: removing stopwords from text using stopwords from nltk in JupyterLab/Jupyter Notebook*

For the exploratory analysis, we chose to lemmatize the text and remove any extra unnecessary words that were not stopwords and occurred frequently. Lemmatizing stems the word back to its most basic form, so “boxes” would become “box” and “changed” would become “change”. This is useful when the exact tense or meaning is not necessary for analysis.

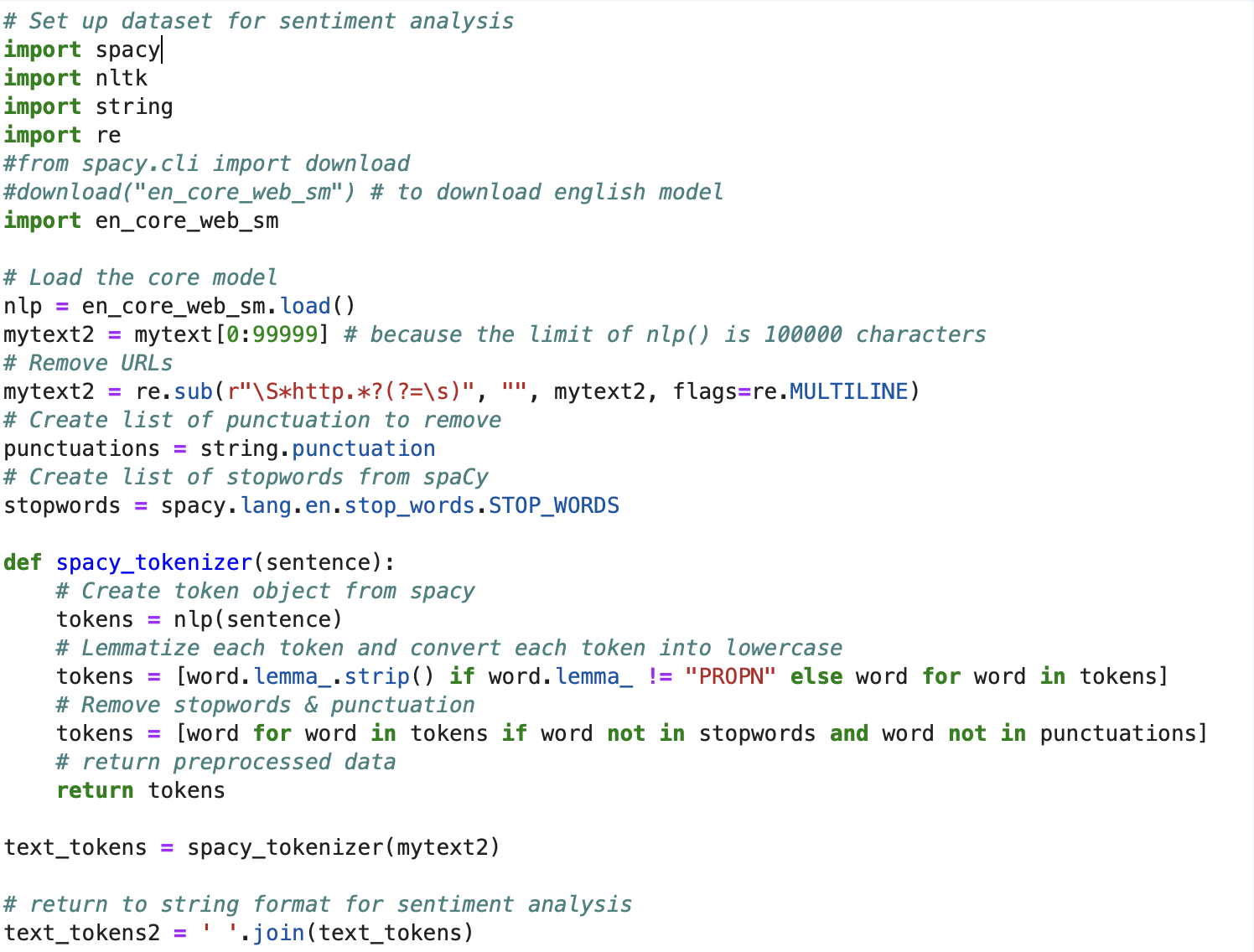


*Figure 17: lemmatizing and removing unnecessary words from text using WordNetLemmatizer from nltk in JupyterLab/Jupyter Notebook*

## Data Preprocessing for Sentiment Analysis

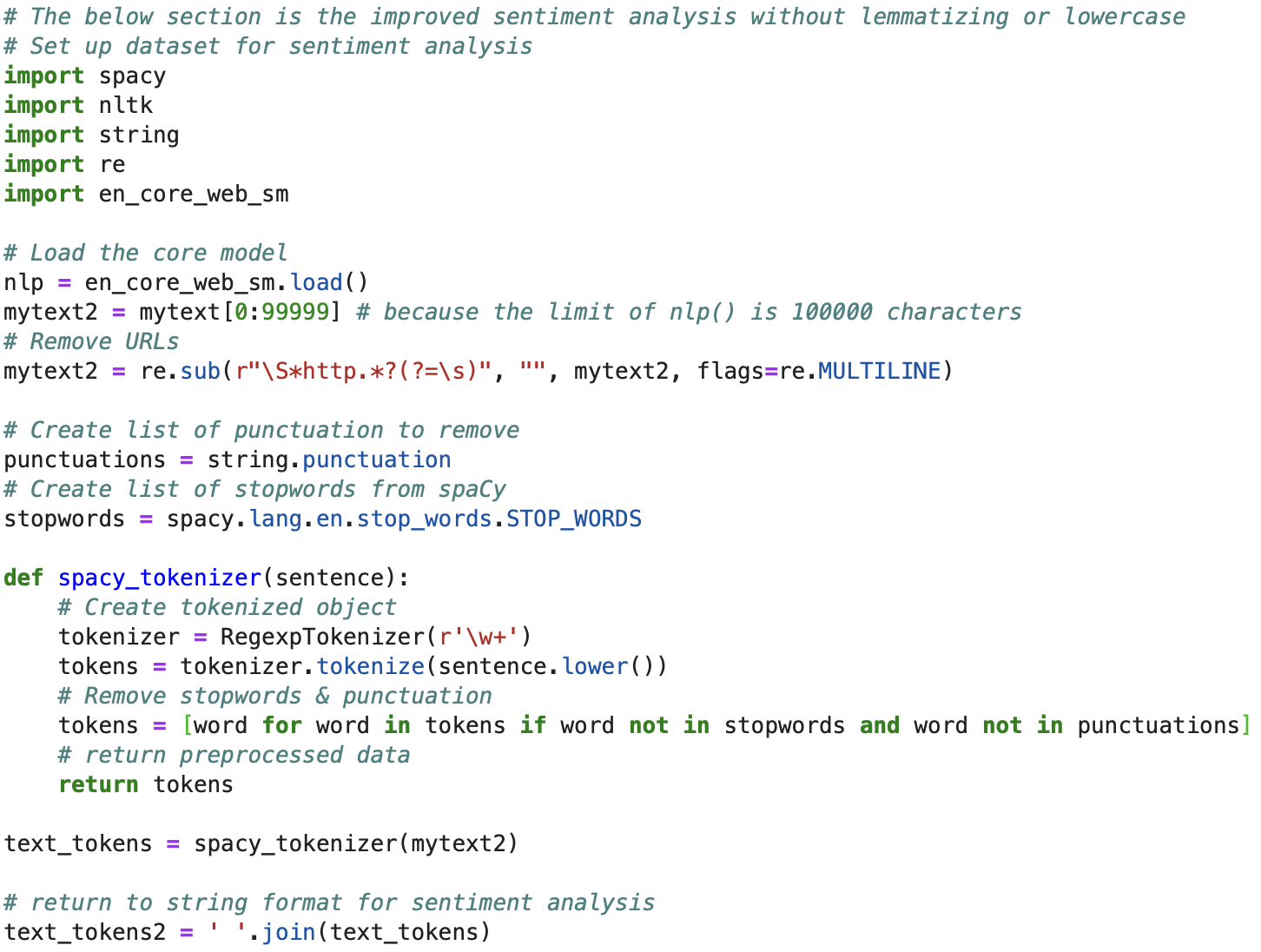
### Baseline Solution Preprocessing

We chose to separate the preprocessing for the sentiment analysis from the exploratory analysis because initially we were not sure what state the data should be in for sentiment analysis. Instead of using NLTK’s word\_tokenize, we used SpaCy’s nlp function as it included more attributes and protects the underlying data from being accidentally changed by performing everything in a document generator.



*Figure 18: preprocessing for baseline sentiment analysis in JupyterLab/Jupyter Notebook*

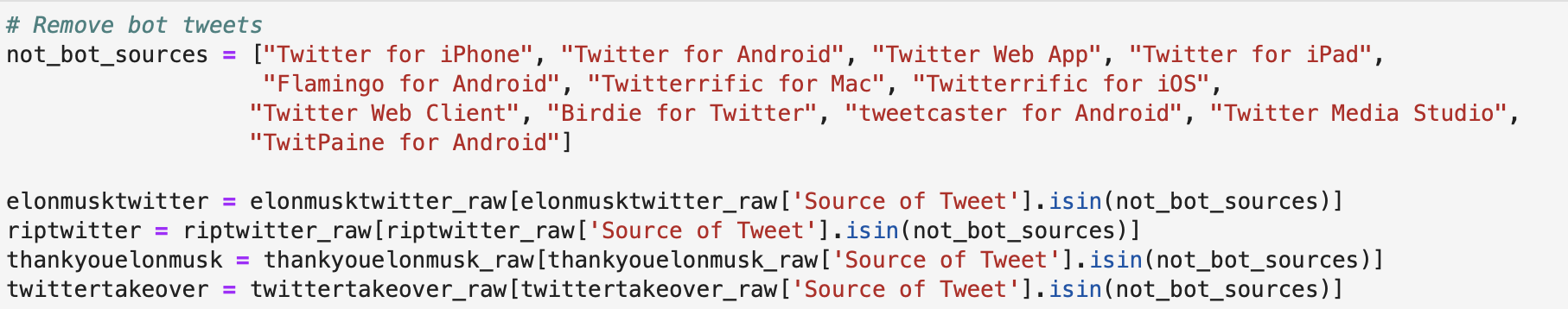
After determining what improvement we wanted to make in the preliminary solution, we changed the preprocessing for this section as shown in Figure XX below.



*Figure 19: improved preprocessing script for sentiment analysis*

### Updated Solution Preprocessing

When updating the sentiment analysis from the baseline solution, we chose not to lemmatize or convert all text to lowercase as different verb tenses and intentional capitalization can affect the perceived polarity of the text (Orbiteers, 2019). This was corroborated by (Hutto & Gilbert, 2014) in the tool’s README file where it’s demonstrated that VADER can better determine valence or intensity when using smaller pieces of text, when the text is as raw as possible (i.e., can take into account emojis/emoticons and any purposeful capitalization). To satisfy these conditions and the need for *some* preprocessing, we did the following: we kept each tweet as a separate row in a pandas data frame, we filtered out known bots by only keeping tweets from the sources shown in Figure 20, and remove hashtags, URLs, and random extra spaces as shown in Figure 21.



*Figure 20: removing tweets from bots in preprocessing*

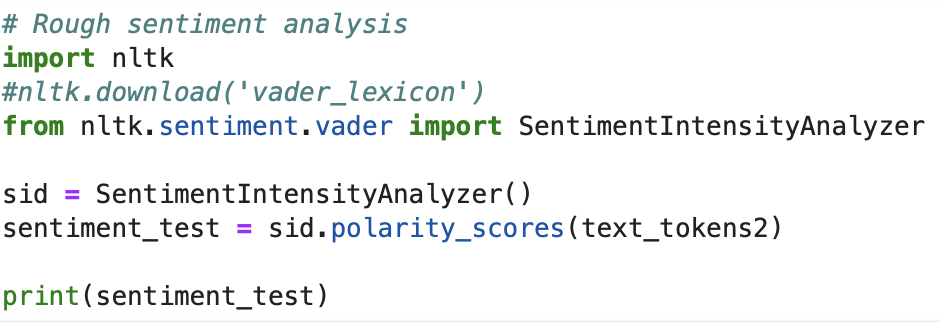
In Figure 21 below, we show the regular expressions substitutions used to remove URLs, new line breaks, and hashtags from the datasets. This was the entirety of the preprocessing done in order to keep as much raw sentiment in the code as possible.



*Figure 21: removing tweets from bots in preprocessing*

## Baseline Sentiment Analysis Experiment and Results

After the data was read into the system, we were able to run the script in Figure 4. SentimentIntensityAnalyzer is a class in VADER that computes the valence scores for a given string. Inside that class, the function polarity\_scores does the heavy lifting by sifting through words, emojis/emoticons, and punctuation to determine the sentiment intensity scores.



*Figure 22: sentiment analysis script using nltk.sentiment.vader.SentimentIntensityAnalyzer.polarity\_scores()*

Table 2 is our results from the code for each of the four hashtags we chose to analyze. The negative, neutral, and positive ratios are the proportions of text of each polarity (or lack thereof for neutral results) for all of the words in the dataset. For example, #TwitterTakeover’s negative ratio of 0.117 means that the number of words in the dataset with negative valence is around 11.7% of the total dataset.

| **Hashtag** | **Negative Ratio** | **Neutral Ratio** | **Positive Ratio** | **Compound Score** | **Overall Sentiment** |
| --- | --- | --- | --- | --- | --- |
| #TwitterTakeover | 0.117 | 0.719 | 0.164 | 0.9999 | Positive |
| #ElonMuskTwitter | 0.118 | 0.73 | 0.152 | 0.9998 | Positive |
| #ThankYouElonMusk | 0.127 | 0.597 | 0.276 | 1.0 | Positive |
| #RIPTwitter | 0.121 | 0.703 | 0.177 | 0.9998 | Positive |

*Table 2: sentiment analysis results using nltk.sentiment.vader.SentimentIntensityAnalyzer.polarity\_scores()*

## Baseline Error Analysis

There are several things that would be going on to give us positive polarity scores across the board. It could be that because every word or phrase isn’t as polarizing as ‘HATE HATE HATE’ that we’re bound to get more calm results. One thing to consider is that if a word had a mean valence or polarity score of zero, it was excluded from the dictionary VADER uses. This could mean that enough seemingly neutral words were used in the tweets that in reality weren’t neutral when interpreted in context that VADER did not consider in its calculations because they didn’t exist in the dictionary. These neutral words are only considered in the ratios in the total word count, but do not change the score.

It is possible our preprocessing was too strong. Due to VADER being a lexicon-based sentiment analysis library and the way it scores words based on the valence of the specified word against the other words around it, the results might not be what one would expect (Hutto & Gilbert, 2014). When preprocessing we lemmatized all words, removed punctuation and stopwords, and converted everything to lowercase in order to have a cleaner dataset for analysis. This might not have been the best option because we sanitized most of what VADER looks at to determine intensity. In our next iteration of the system, we will perform less preprocessing to see if that makes a difference.

It’s also possible that we aren’t utilizing enough SentimentIntensityAnalyzer functions to accurately analyze our dataset. It is possible we are utilizing these functions incorrectly. In the \_\_main\_\_ section of the source code, there is a demo that states VADER is best used for sentences, not paragraphs or novels (Hutto & Gilbert, 2014). In the next iteration, we will separate each tweet, perform sentiment analysis on each individual tweet, and then average the scores to get an overall polarity for each hashtag.

In addition to everything written above, we did not filter out tweets from bots in our baseline solution. In the #TwitterTakeover dataset, nearly 1000 records are bots posting URLs with trending hashtags and a few random words. This could affect our results by adding extra neutral words at best and adding random polarized words at worst. Our next iteration will remove the recognizable bots from the dataset before performing analysis.

# Appendix B: Datasets

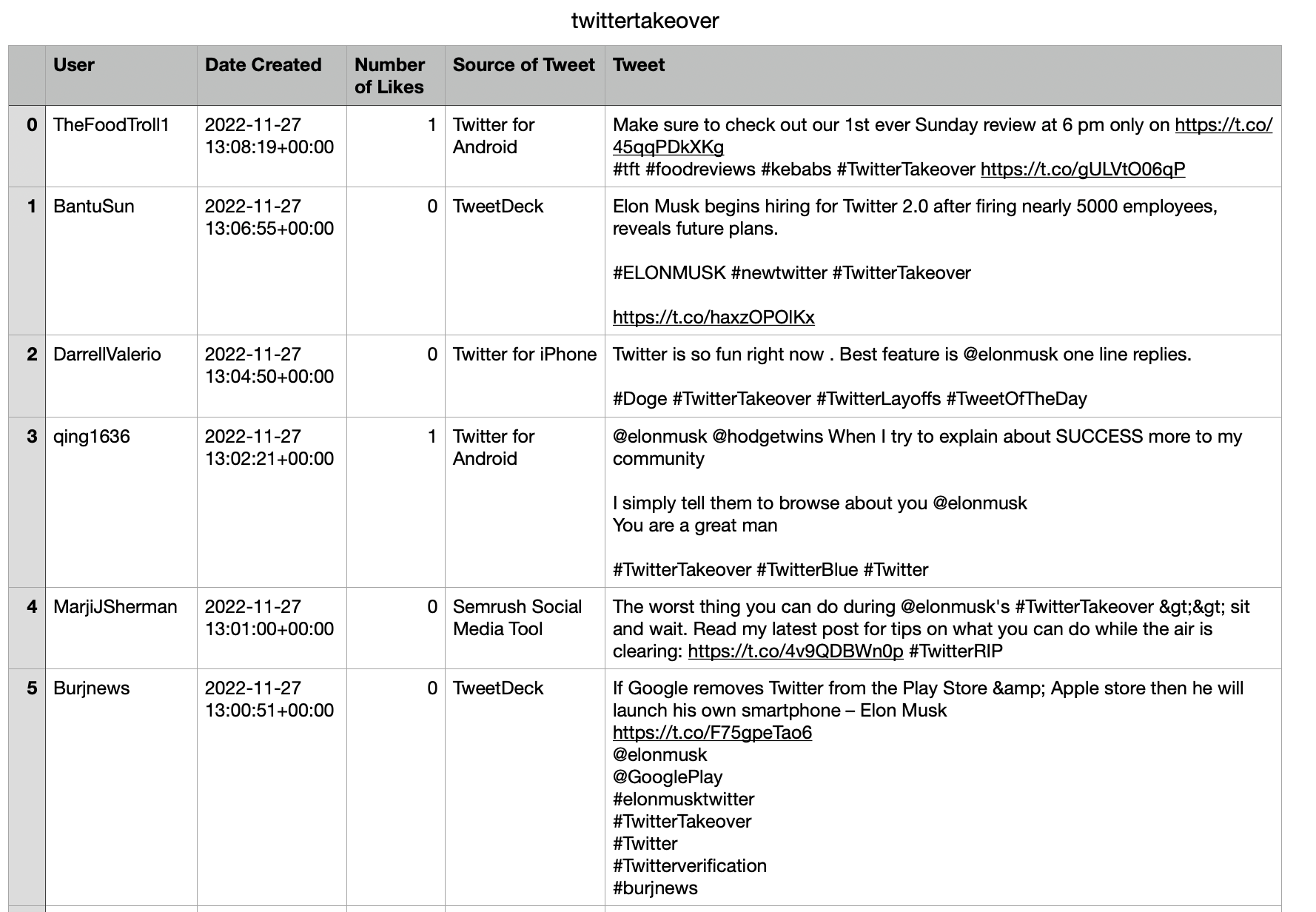
## #ElonMuskTwitter Dataset

### 

*Figure 23: a subset of the #ElonMuskTwitter dataset*

This dataset consists of 10039 records over the following five columns: Username, Date Created, Number of Likes, Source of Tweet, and the Tweet text itself. This hashtag was chosen for its seeming neutrality to the issue of this analysis.

## #TwitterTakeover Dataset



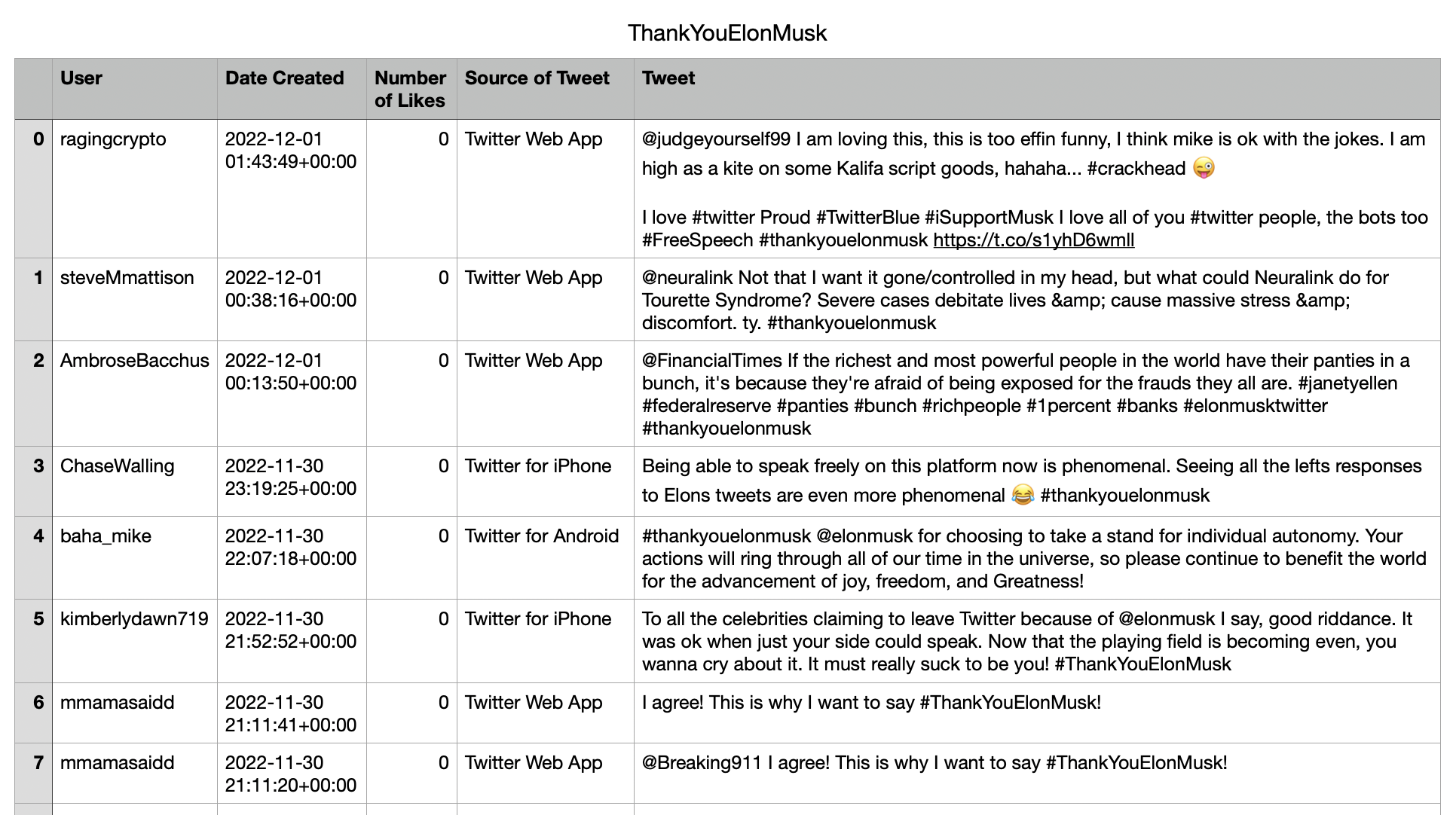
*Figure 24: a subset of the #TwitterTakeover dataset*

This dataset consists of 12406 records over the same five columns as the other datasets. Please refer to the #ElonMuskTwitter Dataset in the appendix or to the Selected Dataset section for more information on the columns themselves. This hashtag was also chosen for its seeming neutrality to the issue of this analysis.

## #ThankYouElonMusk Dataset

This dataset in the table below consists of 1261 records over the same five columns as the other datasets. Please refer to the #ElonMuskTwitter Dataset in the appendix or to the Selected Dataset section for more information on the columns themselves. This hashtag was chosen for its seeming positive outlook on the issue of this analysis.

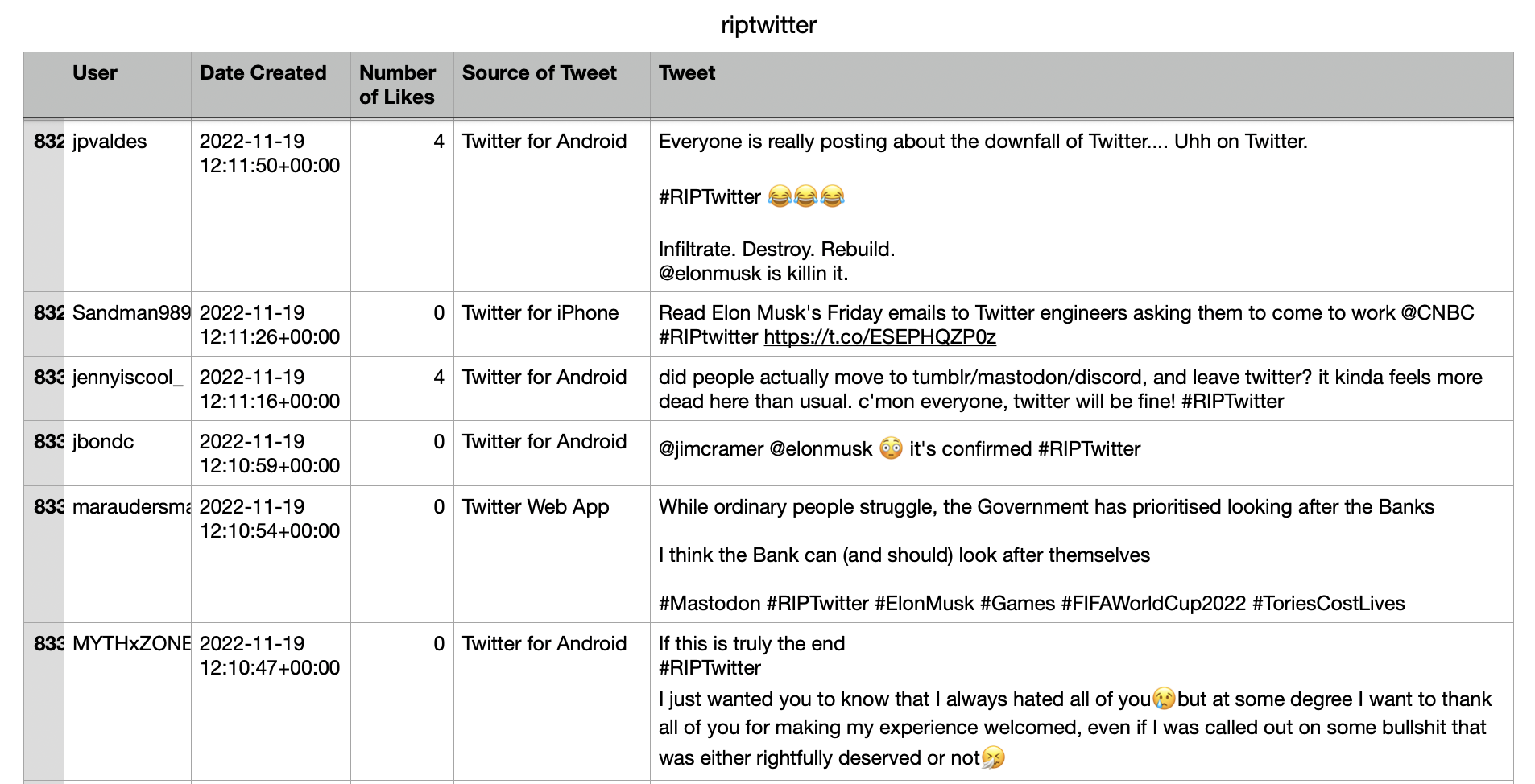
It was harder than expected to find a hashtag with a positive outlook on this event. We also tried #WelcomeMusk, #WelcomeElonMusk, and #ThankYouMusk, but those datasets were too sparse to use in an analysis.



*Figure 25: a subset of the #ThankYouElonMusk dataset*

## #RIPTwitter Dataset

This dataset consists of 8348 records over the same five columns as the other datasets. Please refer to the #ElonMuskTwitter Dataset in the appendix or to the Selected Dataset section for more information on the columns themselves. This hashtag was chosen for its seeming negative outlook on the issue of this analysis.



*Figure 26: a subset of the #RIPTwitter dataset*