INTERDEPARTMENTAL PROGRAMM OF POSTGRADUATE STUDIES IN INFORMATION SYSTEMS

MASTER THESIS

"TWITTER SENTIMENT ANALYSIS ON FAKE NEWS USING PYTHON AND NATURAL LANGUAGE PROCESSING"

SIDIROPOULOU SOUSANA

SUPERVISOR: ECONOMIDES ANASTASIOS

Submitted as required to obtain the Master's Degree
Diploma in Information Systems

SEPTEMBER 2021

ACKNOWLEDGEMENTS

Throughout the writing of this dissertation, I have received a great deal of support and assistance. I wish to thank all the people whose assistance was significant in the completion of this project.

First and foremost, I have to thank my research supervisor Mr Economides Anastasios whose expertise was invaluable in formulating the research questions and methodology. I have benefited greatly from your wealth of knowledge.

I would also like to pay my special regards to Professor Mrs Tzafilkou Aikaterini for her wonderful guidance and support throughout this project. She raised many precious points in our discussions regarding this project and I hope that I have managed to address several of them here.

Getting through my dissertation required more than academic support, and I have many people to thank for listening to me and kept me going.

I wish to express my deepest gratitude to my parents whose unconditional, unequivocal, and loving support kept me motivated and confident.

Last but not least, a big thank you to my friends for their wise counsel and support as well as for the happy distractions to rest my mind outside of this research.

Abstract

The aim of this thesis is to analyze the opinions, feelings, and emotions of Twitter users regarding fake news in account of the research process for the sentimental classification that is being conducted at the end. The theory has neglected to concentrate on fake news as a semantic analysis topic, something that this thesis addresses. It is asked in what level Twitter users express emphatical feelings and not only informative observations. This is done so the polarity of the public concerning fake news and misinformation can be determined, analyzed, and used in further researches regarding the detection of the aforementioned. Using Natural Language Processing (NLP) techniques, Python Programming Language, useful libraries as tweepy, pandas, NumPy, Textplob etc., and Jupyter Notebook as an integrated development environment (IDE), the final semantic score is determined. Every process and technique that participates to the calculation of the final score like data-extraction, data-preprocessing, data-analysis, and data-visualization are also part of the project and are displayed on it. It is shown that the majority of peoples' feelings toward fake news and misinformation in generally are more neutral than negative or positive. It is also shown that despite the numerical difference between the semantic categories, the more strongly expressed feeling was the negative one. The significance of this study is that it informs our understanding that most of the people doesn't give much of their attention in the spread of fake news and when they do, most of the times they are strongly opposed to them.

Keywords: sentiment analysis, polarity, fake news, misinformation, Twitter, NLP, Python

Table of Contents

Table of Figures

1 Introduction

In today's world, everyone is using internet to get informed and express themselves about everything that is happening around them. People are sharing feelings, opinions, experiences, and details about their everyday life. Due to the extensive growth of the Web and the digitization of almost every service, multitudinous feelings and opinions are accessible through internet. Virtual world has arisen not only as a communication bridge between users, but also as an influencing one. Human decision making is surpassingly influenced by the opinion of others. Customers for example, tend to analyze previous reviews regarding to the pursuit that are willing to do. Voters tend to make a research about the elections' candidates and campaigns. Users are searching for previous experiences about the service that they are going to receive. In these days, people are getting informed about the global health crisis of COVID 19 mostly via internet. Sentiment analysis research covers scientific and commercial areas such as analyzing opinions and feelings, detection (Manguri et. al 2020) and even predict future choices. Although it began as a public opinion analysis (Mäntylä, 2018), it is ending up being the most popular research topic in the field of natural language processing.

1.1 FakeNews

Influencing other's opinion doesn't stop there. A big amount of data that floats around the Web has a vast possibility of being fake. The term 'fake news' can be in reference to different forms of data related content. It can be a misleading video, a photoshopped image, a false spreading article or even posts in social media that contain false information. Fake news affects human judgment and leads to misinformation. This misinformation may have a tremendous impact in the society through people's actions and behaviors. Is a quite serious issue considering fake news spread easier and quicker than the regular news and they affect a big percentage of the society like finance (Rapoza et. al 2017, Kogan et. al 2020), politics (Scardingo and Mininni 2020), marketing (Domenico and Visentin 2021), health and even sensitive topics as COVID-19 (Loomba et. al 2021). Users' sentiment orientation about fake news and misinformation is the main objective of this research.

1.2 The Role of Twitter

Twitter is the most typically used web site through the web based social networks. It allows individuals to post their opinion and thoughts, as well as to publish the latest personal, political, social, or international news. Users do that through small messages, which mostly are links to external websites (Khattak et. al 2020), hashtags that are referring to the keywords of the message (and categorize it to make it searchable for other users), audio and visual content. These messages are called tweets and contain impactful data for a plethora of fields.

Twitter, being one of the most popular micro blogging site, has a vast amount of data (Kharde and Sonaware 2016, Koh and Liew 2020, Sharma and Ghose 2020). It is a known fact that a small percentage of the daily produced worldwide text data is structured. The rest is unstructured data, meaning that is comprised of data that is usually not as easily searchable as the structured ones, including formats like audio, video, links etc. Social media is one of the biggest sources of unstructured and sentiment rich data. Twitter is the pool of sentiments in which the hereunto sentiment analysis project is based on.

1.3 Sentimental Classification

To examine the emotional state of users and to perform the opinion and semantic mining, researchers are making use of the natural language processing procedure, known as 'Sentiment Analysis'. By performing sentimental classification, a lot of realizations and insights occur. Businesses are getting useful customer's reviews for their products, as well as sales review about their customer's satisfaction of the given product or service (Bakshi et. al 2016 and Rasool et. al 2018). Political parties identify the mood of the public towards an upcoming election (Bakshi et. al 2016). Streaming services can analyze user's opinion to predict the success of an upcoming movie (Alsaeedi and Khan 2019) or even produce a movie based on people's previous opinions about specific scenes. Digital services' recommender system can propose a new pursuit to a customer by analyzing the usage history. The stock market is getting influenced by the mood of the microblogging clusters of users too (Bakshi et. al 2016).

At the same time, users of social media tremendously affect the role of the health care service (Khattak et. al 2020). Through the semantic orientation and opinion of individuals, patients can get informed about treatments, hospitals and clinics

regarding to their needs. For instance, trace pervious patients with the same symptoms. Benefit from their journey by examining their experiences, reactions and emotions on treatments regarding to their disease. Another field that semantic research can be very useful is psychological health. Recent studies substantiate that social media can be used in favor of detecting and preventing suicidal actions by detecting content and phrases addressing to depression, cyber-bullying etc. A thorough research regarding this matter is done by Birjali et. al (2017).

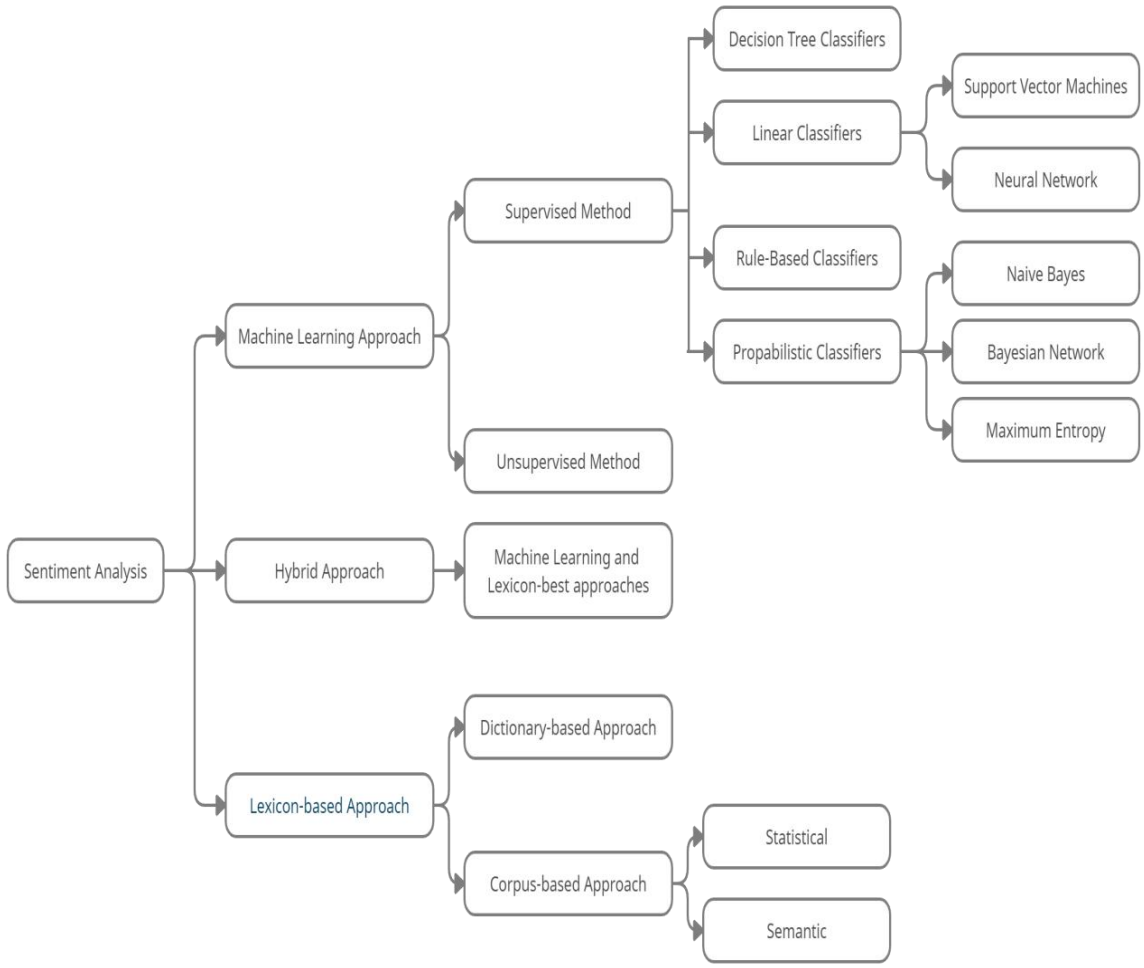
1.3.1 How It Works

Sentiment analysis, also known as opinion mining, can be defined as the process of extracting the sentiment, opinion, and emotion from text data. It will analyze users' sentiment orientation towards an entity like product, services, people, events, organizations etc. This expressed feeling can be classified into 3 categories: 'positive' (in favor of the entity), 'negative' (against the entity) or 'neutral' (neither in favor nor against the entity). The polarity score is expressed by a number within the range of [-1.0, 1.0]. Negative polarity is being represented by scores [-1.0, 0], neutral by [0] and positive by [0, 1.0]. The process of polarity detection on online data like tweets is called polarity classification and is a vital step of sentiment analysis.

Manual naming of sentiment is a time and money consuming job as according to Rutuja et. al (2020) requires man-power and deep knowledge. Therefore, three popular approaches are being used to automate this process. The first is making use of a dictionary of words and the second is based on machine learning approaches. The third one is the compilation of the previous two. In the lexicon-based approach, the sentiment values of the words in the dictionary are assigned to all positive and negative words of the document or sentence in order to match them and calculate the polarity. In difference to the machine learning approaches, lexicon based doesn't have the need to train the classifier first. Hybrid approach is making use of both lexicon based and machine learning approaches. Figure 1 displays every approach in a more detailed way.

Figure 1

Sentiment Analysis approaches and algorithms

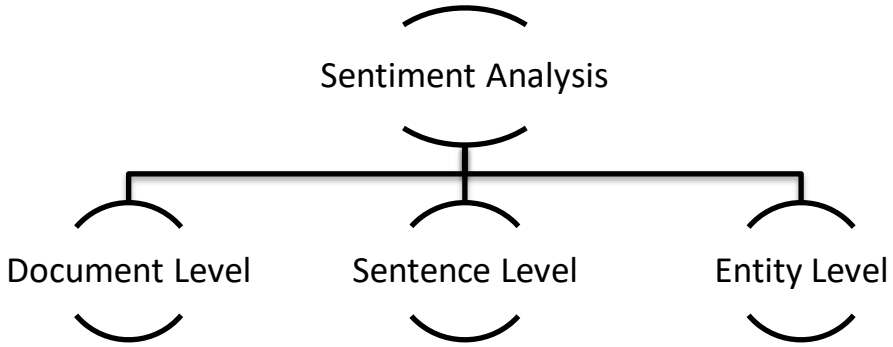


Based on different information providers such as articles, web and researchers, Sentiment analysis can be performed in 3 levels:

- Document level in which a whole document is being classified as ‘positive’, ‘negative’ or ‘neutral’
- Sentence level in which each sentence of a document is being classified as ‘positive’, ‘negative’ or ‘neutral’
- Aspect and feature level also known as entity level in which all the aspects of the sentence are considered and being classified as ‘positive’, ‘negative’ or ‘non-partisan’ (Alsaeedi and Khan 2019).

Figure 2

Sentiment Analysis levels



Nonetheless, Kharde and Sonaware (2016) provide a differentiation in which another level of sentiment analysis classification occurs. It is called word-level classification. According to Anandarajan et. al (2019) and Maria del Pilar et. al (2017), entity level approach may be the most detailed but is quite complicated. In this thesis, sentence level classification is being used.

To perform a sentimental classification, specific steps must be followed depending on the chosen method and the chosen tools. Most of the research on sentiment analysis on any level of classification is mainly based on lexicon-based approaches, machine learning approaches and hybrid approaches. In this project the lexicon-based method and Python programming language are being used to reach to the final semantic classification. The followed steps are being displayed on figure 3.

Figure 3

Sentiment Analysis steps

1.4 Scope

This research seeks to determine, examine and analyze the semantic orientation of Twitter users towards fake news. More specific, it focuses on gaining information

from their expressed opinions regarding fake news and misinformation through extracting and analyzing their tweets. This analysis aims to get access to information based on expressed human opinion and give statistical results about how does fake news make the people feel. Do people feel anger, disappointment and sadness? Or does the existence of misinformation doesn't affect them? Or maybe it doesn't affect them in such level to make them share their thoughts with the rest of the world. This dissertation aims to determine, describe and give answers to the above questions. Based on the related work that is being displayed in the next subsection, and a thorough research, there is no study or published research having as its' main research objective the analysis of peoples' sentiment orientation regarding fake news. There have been numerous articles reviewing fake news detection on a given moment and topic but none of them has sentiment analysis on fake news as their main focus, as it happens in this research.

The sentiment analysis performing in this dissertation is based on a lexicon-based approach, using Twitter's APIs to extract the data, and Python programming language and libraries to clean and pre-process the data. The semantic orientation is done by determining subjectivity and polarity of tweets and by assigning the appropriate sentiment score on them. The purpose is to analyze users' feelings and opinions regarding to fake news spread across Twitter. This research can give useful insights and can be used in the prior steps of fake news detection research as according to Alonso et. al (2021) it is a vital step of the detection process.

The rest of the research is structured as follows. In section 1.5 there is the related work in which comparisons between previous research's techniques and approaches on sentimental classification are discussed. In section 2 there is the followed methodology of the project analyzing every step taken from the beginning regarding the technical part. In section 3 is data analysis in which the techniques followed for analyzing the sentiment analysis result are introduced and discussed. Data visualization of the semantic score regarding subjectivity and polarity can be found on section 4. In section 5 is the conclusion of the project and the proposed future word. Lastly in the end of this dissertation there are the references used for this research.

1.5 Related work

The task of sentiment classification has received considerable interest due to its potential applications. In the recent years a lot of work has been done on this field by

numerous researchers. They have performed sentimental analysis on twitter data for different purposes and by different approaches. Most of the researchers are applying machine learning techniques, as according to Ahmad et. al (2017) are 'effective and reliable for opinion mining and sentiment classification'. Avast amount of research work is based on comparing different machine learning algorithms like: Naïve Bayes (NB), Maximum Entropy (MaxEnt), Random Forest, Support Vector Machine (SVM), Decision Tree, K-Nearest Neighbour (KNN), Logistic Regression etc. This is done in order to identify which one has the best accuracy and can be more useful and reliable in future work.

Ahmad et. al (2017) is presenting a thorough machine learning approach for sentiment classification analyzing eight different algorithms. Maximum Entropy (MaxEnt), Stochastic Gradient Descet (SGD), Random Forest, SailAil Sentiment Analyzer (SASA), Multilayer Perceptron (MLP), Naïve Bayes (NB) Multinomial Naïve Bayes and Support Vector Machine (SVM) are being presented in the paper without having a specific data set thought. It shows that in the given time and with the given data, Random Forest algorithm have the biggest accuracy of 88,39 %.

In the case of Vishal and S.S. Sonawane (2016), the research is being conducted with the three basic machine learning algorithms which are: Naïve Bayes, Maximum Entropy and Support Vector Machine. They use the publicly available datasets for Twitter, and they pre-process the data. The survey continues with the feature extraction which helps with the determination of the opinion of the individuals. Lexicon-based approach is also mentioned and discussed as it is being compared with the machine learning approach. The conclusion of the research refers that SVM and NB have the highest accuracy in contrast with lexicon based, which is very effective but only in cases that requires few human effort.

In the research of Alsaeedi and Khan (2019) distinctive techniques for Twitter sentiment analysis methods are discussed such as lexicon-based, machine learning, hybrid and ensemble which is the combination of multiple classifiers. It occurs that the unsupervised approaches like ensemble and hybrid-based, perform better than the supervised machine learning ones. Something that the authors didn't expect in the begging.

Naïve Bayes and Maximum entropy are also being discussed in the research of Hemalatha et. al (2013) but with proposed classifiers of both algorithms being involved.

The reference involves media sites' data, and it is only about supervised learning. It is aiming to provide insight in future work as the proposed method achieves high accuracy.

The aforementioned classification algorithms are also mentioned by Ravinder et. al (2019). The research goes one step further by also considering the TF-IDF and N-Grams methods for feature extraction and their impact in the sentiment analysis. TF-IDF evaluates the performance of a single word in the document and N-Gram forms the features for the supervised algorithms. The outcome is that TF-IDF is the best choice between those two methods and Logistic Regression algorithm is the wining, giving the bigger predictions of sentiments.

Logistic Regression algorithm is a topic of discussion in the work of Mitra (2020) too alongside with other previously mentioned algorithms. An interesting work is being proposed as well for the sentiment analysis procedure. The proposed work is mainly based on the machine learning classification with the test and the training data set but also applies a further step that involves prediction set data. The scope is to achieve prediction using the model that gained experience in the training step. It concludes by measuring accuracy of the algorithms with Random Forest having the highest of 80%.

Some research works are focusing on a specific topic instead of specific algorithms. They provide the public with information and statistics about situations happening around the globe. Manguri et. al (2020)pertains to the global COVID-19 disease and aims to show how the situation is being issued by media, governments, and people. Twitter's APIs are used for data collection using only two keywords, #coronavirus and #COVID-19. Python's libraries such us Tweepy and TextBlob is also used for the determination of sentiment analysis and Naïve Bayes as the classification algorithm. An important parameter is that the collection of the data was conducted only for specific weeks that were the most spread weeks of coronavirus according to Johns Hopkins Coronavirus Resource Center. https://coronavirus.jhu.edu/. This has a vital influence in the outcome as is showed that more than 50% of twitter data characterized as 'neutral'. Thus mining, people's feelings vary day to day. In the same pattern, Rasool et. al (2019) are using the same algorithm but this time with the purpose to analyze apparel brands, Nike and Adidas and show which is peoples' preferable. Following the document-based approach, they compare the data of the 2 different datasets and give a

very useful insight to the investors of the brands. An also useful observation in this research is that even though the positive polarity of Adidas is bigger than the one of Nike's, the neutral polarity of Nike is bigger and according to the writers this means that the customer's satisfaction level is bigger too and in favor of Nike. It is an understanding that positive polarity by itself is not always enough to work in favor of the entity.

A different proposed sentiment classification system is being presented by Reddy et. al (2019). According to the writers the proposed system overcomes the existing by classifying emotions in 7 categories instead of 3 and by using real time data and not stored. Also,it presents the result of the analyzed tweets in a pie chart, something that according to the writers is an advantage between the existing and the proposed system. The proposed work could have been more inclusive as the existing system gives the opportunity to present the result in a pie chart as well or in any other data visualization panel. Also regarding to the drawback of using stored data before the analysis occur, a solution could be to execute the data extraction algorithm at the given moment before the data pre-processing procedure begin. With that said, the remaining emotion classification of 7 different emotions presented may need a large dataset to show useful information regarding to the case study, as in a small amount of tweets the result may be vague and without any useful insights.

Although machine learning approaches for sentiment classification may give big accuracy results therefore an ensured success, they require big set of datasets in order to train and test the classifier and this is time consuming. Hence researchers in that field are trying to find alternative ways to speed this process but preserve the accuracy. A very interesting propose regarding this matter is coming from Bakshi et. al (2016). They present a new algorithm that takes a tweet as an input, clean it, break it to words (tokenize it), compare every word in dictionary and if is found, label it with the appropriate sentiment score. Although it is an algorithm that includes both data pre-processing and sentiment classification, it needs a dictionary of positive, negative, and neutral words to use and assign the sentiment. This dictionary is made by the user, meaning that needs a tremendous amount of words for it to work properly. These words must also be assigned to sentiments manually and most importantly correctly. Moreover, the data extraction method it is not mentioned.

A different work comes from Samuels and Mcgonical, n.d. towards sentiment classification. It is a lexicon-based approach that operates with the BBC news dataset instead of tweets. Also uses tools like Rapid miner to pre-process the data and SentiWordNet3.0.0 dictionary to assign sentiment score. The final sentiment score provision is fulfilled by just calling a specific operator which is an extension of the SentiWordNet3.0.0. No feed on accuracy is being provided.

Even though fake news is a factor that influences big part of society, it's only the last few years that researches are being done regarding misleading information (Alonso et. al 2021).Most of them are on fake news detection and some of them are about sentiment analysis for fake news detection. Surprisingly none of them are just about semantic analysis on fake news. The present research is filling the void as its' main aim is to determine users' feelings about fake news and misinformation. Only a brief mention was done by Bagheri and Islam n.d. regarding this matter. By using a plethora of available Python libraries and different keywords, opinion mining referring to various queries was done including fake news.

Table 1

Comparison of previous related work

Author	Technique	Issues Addressed	Dataset	Results	Accuracy (%)	Suggested Future Work
Ahmad et. al (2017)	Machine learning classifiers	The performance of Machine Learning algorithms	Different datasets	The given algorithm comparison can be used as reference in future works	Based on different researches and different topics: Maximum Entropy 72.6% Random Forest 88.39% SASA 64.9% MLP 81.5% & 79.27% Naïve Bayes 75.5% & 62.5% SVM 81.15% & 79.4%	-

Author	Technique	Issues Addressed	Dataset	Results	Accuracy (%)	Suggested Future Work
Kharde and Sonawane (2016)	Machine learning and lexicon-based approaches	Comparison of existing techniques for opinion mining	Different datasets	A combination between machine learning and lexicon-based techniques improve the accuracy of sentiment classification	-	-
Alsaeedi and Khan (2019)	Supervised, unsupervised and hybrid methods	Theoretical comparison of the existing sentiment analysis methods	Different datasets	Combination of multiple classifiers and hybrid approaches perform better than supervised machine learning techniques	Machine learning algorithms (Naïve Bayes Maximum Entropy and SVM) ~ 80% Ensemble and hybrid approaches ~ 85%	Fluctuation in the performance of sentiment analysis algorithms where multiple features are considered
Hemalatha et. al (2013)	Naïve Bayes, Maximum Entropy and proposed classifier on both	Tweet collection and provisional of a view of business intelligence	Twitter	Maximum Entropy performs better than Naïve Bayes	-	The application tool can be used to provide a way to find out technological trends in the future
Ahuja et. al (2019)	6 different algorithms of classification	Comparison of two features (TF-IDF & N-Grams)	SS-Tweet Dataset	Logistic Regression algorithm is the best for sentiment analysis and both feature extraction techniques are good enough with TF-IDF having a small leverage.	TF-IDF KNN 46% Dec. Tree 46% SVM 46% Log. Regress. 57% Naïve Bayers. 53% Random Forest 51% N-Gramms KNN 46% Dec. Tree 48% SVM 46% Log. Regress. 49% Naïve Bayes. 50% Random Forest 51%	Comparison of other features classification like word polarity score, word embeddings, twitter specific features etc.

Author	Technique	Issues Addressed	Dataset	Results	Accuracy (%)	Suggested Future Work
Mitra (2020)	Proposed work of building classifier models using test, training and prediction set.	Comparison of algorithms accuracy	Movie Review Dataset	The strength of a sentiment classifier depends on the scale of the lexicon (dictionary)	Naïve Bayes 70% SKlearnBernoulliNB 70% SKlearn SVM 75% Decision Tree 52% Random Forest 80% KNN 71%	-
Manguri et. al (2020)	Naïve Byes Classifier	How people, governments and media broadcast the COVID-19 outbreak situation	COVID-19 outbreaks Dataset from Twitter	Neutral opinion based on polarity was over 50% and objective portion over 64%	-	Application of the research model in other social media too with the use of lexicon based algorithms regarding similar cases
Rasool et. al (2019)	Naïve Byes Classifier	Detection of polarity and emotion values of customers	Twitter data regarding 'Nike' and 'Adidas'	Online users compare other brands will making a decision about a brand	-	Use of hybrid sentiment classification for better accuracy
Reddy et. al (2019)	Proposed system with the use of Naïve Bayes classifier	Propose a system that overcomes the drawbacks of existing system	Video-games reviews from Twitter	The classification and visualization of seven major emotions instead of three	78.38%	-
Bakshi et. al (2016)	Proposed algorithm for sentiment classification	Design an algorithm that can efficiently compute the sentiments of Twitter data	Twitter	Performs good in both speed and time and can be used for larger datasets	80.6%	Reduce the sarcastic element by better processing

Author	Technique	Issues Addressed	Dataset	Results	Accuracy (%)	Suggested Future Work
Samuels and Mcgonical (n.d.)	Lexicon-based approach	Sentiment analysis on news and blogs	News articles from BBC	Categories of business and sports have more positive articles	-	Development of an online application for readers t read news, based on a sentiment analysis for better news customization
Bagheri and Islam (n.d.)	Proposed algorithm with combination of Python libraries	The importance of social network analysis and its application	Twitter dataset regarding different daily topics	Neutral sentiment is significantly high which indicates the need to improve Twitter sentiment analysis	-	Improval of Twitter sentiment analysis

2 Methodology

This project is aiming to explore an under-researched topic that is the sentimental classification of Twitter users about fake news. It is using the lexicon-based approach and with the use of python libraries the data extraction, pre-processing and assignation of the sentiment are being done. Twitter is used as the information provider as the data that this project is using are tweets of public users. Every action that is being performed is on that data. The type of data is qualitive, meaning that are expressed in words. As the Jupyter Notebook is being used for the project, the Anaconda science platform must be installed. All the python libraries that will be used, are being installed too through the anaconda prompt with the command "conda install *package name*".

2.1 Data extraction

The primary step is to collect the data. Tweets are not secondary data. They're being collected at the given moment, and they refer to the last 7 days (Beck, 2021) if not a specific date period is determined (this is a limitation given from Twitter).Then, they are being stored programmatically in a csv file. A lot of parameters occur in the process of extraction aiming to a more specific dataset. For instance, language: English (only tweets written in English are being extracted), keywords and hashtags: #fakenews, #misinformation, #hoax, #lies, #bots etc. Sometimes a combination of those keywords occurs. Also, there is no geographical limitation, meaning the tweets can be from everywhere. The extraction process took place at different time periods and not only just once, for the data to be more objective and to have a variety of emotions involved. The number of the extracted tweets in total is 58.454. More specific, for every used keyword: #bots: 4.487, #fake: 4.278, #fakenews: 16,788, #fakenewsvaccine: 6.945, #hoax: 9.486, #lies: 4.725, #misinformation: 7.528 and finally #misinformationcovid: 4.217.

2.1.1 Twitter API's

The method that is been used in this project to extract tweets from Twitter is with the help of Twitter APIs. APIs (Application Programming Interface) is a software that bridges the gap between two applications (Fontanella, 2021).Twitter APIs gives access to twitter data and permission to read and download big amounts of data through its' list of commands. Thus, is a vital tool in the extraction process. To get access on those APIs, the individual must have a twitter account. Secondly, an application must be done for a developer account, and this must be approved. It is a process that may last

one or two weeks until it is approved, as the user must give a thorough review about the use case and the usage of the APIs. When it's approved, the four passwords are accessible and ready to use. These passwords consist of:

- Consumer Key (API Key): xxxxxxxxxxxxxxxxxxxxxxx
- Consumer Secret (API Secret): xxxxxxxxxxxxxxxxxx
- Access Token: xxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxx
- Access Token Secret: xxxxxxxxxxxxxxxxxxxxxxxxxx

These keys and tokens are different for every user and must not be shared. These passwords must be a part of the code in order to give the user access to twitters' posts and thus to the data. If a big amount of data is downloaded in a short period of time, then Twitter bans the users for some days.

The connection with the APIs is done after the appropriate Python packages importation. To import the packages, first they must be installed through anaconda prompt. In this project tweepy, pandas, and csv are being used.

- Tweepy is an open-source Python package that bridges Twitter API with Python and through its methods it handles various implementations such as: data encoding and decoding, 0Auth authedication, HTTP requests etc. (Python, 2021)
- Pandas is an open-source data analysis and manipulation Python package. Pandas provide an easy way to create, manipulate and display the data in the data frame. ("Python Pandas Tutorial: DataFrame, Date Range, Use of Pandas", 2021)
- CSV is the type of file that the extracted tweets will be stored in order to process with the next step that is the pre-processing process.

Figure 4

Connection to Twitter APIs

```
#import the appropriate packages
import tweepy
import pandas as pd
import csv
```

```
consumer_key = '[redacted]'
consumer_secret = '[redacted]'
access_token = '[redacted]'
access_token_secret = '[redacted]'
```

```
#create the connection with twitter API
auth = tweepy.OAuthHandler(consumer_key, consumer_secret)
auth.set_access_token(access_token, access_token_secret)
api = tweepy.API(auth,wait_on_rate_limit=True)
```

The next step is to create a pandas data frame in which the collected tweets will be displayed depending in the parameters that had been set. For example, the code about extraction may demand extraction of text, URLs, usernames, retweets etc. The display of the extracted data may be with only the text and the username regarding to what is being asked in the command. A lot of other parameters in the code can be added too, like specific date, specific country of the origin of the tweet, username, specific numbers of tweets that the individual want to extract etc. The keywords that are being used in the code of this project to scrap the tweets including them are: fakenews, misinformation, covid19 and misinformation together, hoax, lies, bots, fakenews bots, misinformation bots and vaccine fake news and the only parameters are the keyword and the language. The code is running every time containing a different keyword. Sometimes a combination of keywords occurs, like: covid19 and misinformation, fakenews and covid19. If there is no limit in the number of tweets that are being extracted as a parameter in the code, then is limited to 18.000 tweets for a 15 minute run (Beck, 2021)

Figure 5

Tweets' extraction

```
#create a pandas data frame
df = pd.DataFrame(columns=['text'])
pd.set_option ('display.max_colwidth', 300)
msgs = []
msg =[]
```

```
#extract tweets and display them in the data frame
for tweet in tweepy.Cursor(api.search, q='#fakenews', lang="en").items(): #since="2017-04-03"
    msg = [tweet.text]
    msg = tuple(msg)
    msgs.append(msg)
df = pd.DataFrame(msgs)
df.columns = ['text']
df
```

When the extraction is done, tweets are visible in the data frame showing the sequence number of tweets and the text in the 'text' column.

Figure 6

Tweets' display

	text
0	RT @cis_india: Send in your abstracts by the end of the week (13 August), to participate in the first edition of the CIS Seminar series on...
1	Wildfire smoke linked to thousands of coronavirus cases on West Coast\n\nThis is some serious #BullShit \n#LeftWingBS... https://t.co/drr3Sj0sZB
2	'Storm'? They walked quietly around, you #FakeNews muppets!!! https://t.co/MutW0iSEHv
3	What happens now in #Afghanistan is an argument for the 20 years American & NATO mission who brought security and s... https://t.co/Qirm5oiyf5
4	RT @larryelder: CNN Reports 4 Teachers from Broward County Have Died of Covid to Trash Gov. DeSantis, Fails to Mention that Schools Are Not...
...	...
8524	RT @ManushyaFdn: #ไฟเขียวร์หาย'ไปไหน ⚠ Thai Police Violence during #ม็อบ7สิงหา protest was REAL and is against Human Rights Standards! Check...
8525	RT @PakistanFauj: India are masters of #FakeNews & propaganda against #Pakistan\n\n265 coordinated #Fake local media outlets serving indian i...
8526	Of course! Ppl w 1/2 of a brain know this. However, the right-wing, conservative science-denying anti-vaxxers will... https://t.co/L5hTNbhV2s
8527	@ashishkjha Of course! Ppl w 1/2 of a brain know this. However, the right-wing, conservative science-denying anti-v... https://t.co/l1yZsdNj9T
8528	RT @ManushyaFdn: #ไฟเขียวร์หาย'ไปไหน ⚠ Thai Police Violence during #ม็อบ7สิงหา protest was REAL and is against Human Rights Standards! Check...

8529 rows × 1 columns

2.1.2 Data Storage

After the extraction, a command is being used to store these tweets into a csv file and name it automatically. This happens because it is more organized and easier to apply pre-processing methods in a csv file rather than in a pandas data frame.

Figure 7

Tweets' storage

```
#save tweets in a csv file
df.to_csv('fakenews.csv',encoding='utf-8', index =False)
csv = 'fakenews.csv'
```

Every csv is named after the corresponded keyword. Thus, meaning that for every keyword there is a csv full of tweets including this keyword. The result is having 15csv files with every csv referring to different keyword most of the times. There are files including tweets with the same keyword, but they have a different date that were extracted. Additionally, having one csv for every keyword can help in a further analysis. Analyzing the polarity of data referring to every keyword, thus every file individually, different comparison can be made like: data regarding to which keyword have the highest percentage of positivity and what does this mean? An analysis like this is being conducted in this project in the process.

2.2 Data pre-processing

The way that people express themselves is by natural language. Because of the casual approach on writing a tweet, except of text data, it can contain punctuation, links, URLs, numbers etc. Also, a lot of spelling mistakes and grammatical errors can be found too. Furthermore, a lot of times people are using irony and sarcasm to make a statement, that means that the same word may have a totally different usage and contain a totally different sentiment between different statements. Meaning it can be a positive word but with a negative meaning and vice versa. This can shift the sentimental polarity of the text (Li et. al n.d.). These types of noisy text data create an obstacle in the allocation of the sentiment and make it hard for the researchers to understand and classify them.

The solution to this is to pre-process these data in order for them to get cleaned and ready for the sentiment analysis. In this project, this is done by 1) removing the duplicates, 2) removing URLs, 3) removing punctuation, 4) tokenization, 5) POS-tagging, 6) removing stop words and 7) lemmatization. Before the application of these procedures, all 15csv files are combined into one by using an online tool. This will

make the identification of duplicates easier and more accurate. Furthermore, it is time saving as all the data cleaning techniques will be applied one time instead of 15.

The next step is to load the new combined csv file named ‘merged.csv’ and display it in a new data frame named ‘merged_df’. The summary number of tweets (before cleaning) is ‘58.454’.

Figure 8

Csv file display

```
#display the csv file
pd.set_option ('display.max_colwidth', 300)
merged_df = pd.read_csv(r'merged.csv', encoding='latin1', sep='delimiter', engine='python')
merged_df.columns = ['text']
merged_df
```

	text
0	Study: #Bots amplified pandemic #misinformation on #socialmediaÂ https://t.co/SUQQJWwCDE via @medcitynews #digitalhealth #medicalchatbot #ehealthInformation #hcsm;;;
1	"RT @MonarchinMN: New Report Confirms How #Iranâ□□s Regime #FakeNews And #Propaganda Work
2	https://t.co/BuWallsLPxÂ #infosec #cybersec #threatsâ□¦";;;
3	@condesm @jgnaredo @Katsuragui Seguro x ser un criadero de #bots y propagar #FakeNews;;;
4	"RT @carlos2015aceve: Lo dijo @JhonnyKandanga en @NoticiasBqto
...	...
58449	Giving misinformation that fiber will be fixed today.";;;
58450	"Facebook should have done something about Biden's ""dangerous misinformation"". https://t.co/zog7v6MapV";;;
58451	listen thereâ□□s a reason why tiktok is a hub of misinformation;;;
58452	President Biden clarifies his comment saying Facebook is â□□killing peopleâ□□: â□□My hope is that Facebook, instead of taking it personally â□¦ that they would do something about the misinformation about the vaccine.â□□ https://t.co/cXts1vN3uq;;;
58453	"They know they have been promoting lies for profit. Letâ□□s call it what if is & tell #Facebook whatever misinformation they took down hurt so many more & a bit late for many! FYI @mikememoli";;;

58454 rows × 1 columns

2.2.1 Duplicates Removal

The priority in this step is for the number of duplicated tweets to be determined. By using the command: *merged_df.text.duplicated().sum()*, the number of tweets that are double is visible, and those are a total of ’56.739’. Thus, meaning that from the ‘58.454’ tweets, only ‘1.715’ are originals. The originals tweets are being displayed and transferred to a new data frame named ‘cleaned_df’. This is done for practical reasons and to be easier saved in a new csv file that will contain only the originals and will be used for the rest of the data cleaning process. The name of this csv is ‘cleaned.csv’.

Figure 9

Original tweets

```
#display the original tweets
merged_df.drop_duplicates(keep='first')
```

	text
0	Study: #Bots amplified pandemic #misinformation on #socialmediaÂ https://t.co/SUQQJWwCDE via @medcitynews #digitalhealth #medicalchatbot #ehealthInformation #hcsm;;;
1	"RT @MonarchinMN: New Report Confirms How #Iranâ□□s Regime #FakeNews And #Propaganda Work
2	https://t.co/BuWallsLPxÂ #infosec #cybersec #threatsâ□¦";;;
3	@condesm @jgnaredo @Katsuragui Seguro x ser un criadero de #bots y propagar #FakeNews;;;
4	"RT @carlos2015aceve: Lo dijo @JhonnyKandanga en @NoticiasBqto
...	...
58318	"@mtgreenee Republican lies up 79%
58319	Republican conspiracy theories up 98%
58320	Republican insurrection up 100%
58321	Republicans are spreading more COVID misinformation than ever before. Like no one has ever seen.
58322	* Please RT * Vote Democrat for America!";;;

1715 rows × 1 columns

```
cleaned_df = merged_df.drop_duplicates(keep='first')
```

Figure 10

Saving data frame to a csv file

```
#save to csv
cleaned_df.to_csv('cleaned.csv',encoding='utf-8', index =False)
csv = 'cleaned.csv'
```

2.2.2 URLs Removal

Before any URL removal procedure occur, the new csv containing only the original tweets must be displayed in a new data frame with the name ‘cleaned_df’.

Figure 11

Tweets after duplicate removal

```
#display new data frame
cleaned_df = pd.read_csv(r'cleaned.csv', encoding='latin1', sep='delimiter', engine='python')
cleaned_df.columns = ['text']
cleaned_df.head(1715)
```

	text
0	Study: #Bots amplified pandemic #misinformation on #socialmediaÃ□Â https://t.co/SUQQJWwCDE via @medcitynews #digitalhealth #medicalchatbot #ehealthInformation #hcsm;;;
1	"""RT @MonarchinMN: New Report Confirms How #IranÃ¢Â□Âs Regime #FakeNews And #Propaganda Work"
2	"https://t.co/BuWaIlsLPxÃ□Â #infosec #cybersec #threatsÃ¢Â□Â¦"";;;"
3	@condesm @jgnaredo @Katsuragui Seguro x ser un criadero de #bots y propagar #FakeNews;;;
4	"""RT @carlos2015aceve: Lo dijo @JhonnyKandanga en @NoticiasBqto"
...	...
1710	"""@mtgreenee Republican lies up 79%"
1711	Republican conspiracy theories up 98%
1712	Republican insurrection up 100%
1713	Republicans are spreading more COVID misinformation than ever before. Like no one has ever seen.
1714	"* Please RT * Vote Democrat for America!"";;;"

1715 rows × 1 columns

Every result of a removal technique will be displayed in a new column next to the previous one. It's a practical way to compare the effectiveness of the applied procedure.

To start the process of removing URLs, first a new Python package called 're' must be imported. 're' stands for Regular Expressions and is a special sequence of characters that helps find and match other strings or set of stings("Python - Regular Expressions", 2021).After the determination of the special characters, the code is running and when the given character matches one identical in the text, it removes it. For example, having this text: *'My blog is https://www.tutorialexample.com and not https://tutorialexample.com'* after the URLs removal process it would end up being: *'My blog is and not'*("Best Practice to Extract and Remove URLs from Python String - Python Tutorial", 2021). In the project, the application of re package is applied on the 'text' column.

Figure 12

Importation of 're' library

```
#urls removal
import re
cleaned_df['clean_url'] = cleaned_df['text'].apply(lambda x: re.split('https:\/\/.*', str(x))[0])
cleaned_df
```

Figure 13

Tweets after URLs removal

	text	clean_url
0	Study: #Bots amplified pandemic #misinformation on #socialmediaÃ□Â https://t.co/SUQQJWwCDE via @medcitynews #digitalhealth #medicalchatbot #ehealthInformation #hcsm;;;	Study: #Bots amplified pandemic #misinformation on #socialmediaÃ□Â
1	"""RT @MonarchinMN: New Report Confirms How #IranÃ¢Â□Â□s Regime #FakeNews And #Propaganda Work"	"""RT @MonarchinMN: New Report Confirms How #IranÃ¢Â□Â□s Regime #FakeNews And #Propaganda Work"
2	"https://t.co/BuWallsLPxÃ□Â #infosec #cybersec #threatsÃ¢Â□Â¦""";;;	"
3	@condesm @jgnaredo @Katsuragui Seguro x ser un criadero de #bots y propagar #FakeNews;;;	@condesm @jgnaredo @Katsuragui Seguro x ser un criadero de #bots y propagar #FakeNews;;;
4	"""RT @carlos2015aceve: Lo dijo @JhonnyKandanga en @NoticiasBqto"	"""RT @carlos2015aceve: Lo dijo @JhonnyKandanga en @NoticiasBqto"
...	...	...
1710	"""@mtgreenee Republican lies up 79%"	"""@mtgreenee Republican lies up 79%"
1711	Republican conspiracy theories up 98%	Republican conspiracy theories up 98%
1712	Republican insurrection up 100%	Republican insurrection up 100%
1713	Republicans are spreading more COVID misinformation than ever before. Like no one has ever seen.	Republicans are spreading more COVID misinformation than ever before. Like no one has ever seen.
1714	"* Please RT * Vote Democrat for America!""";;;	"* Please RT * Vote Democrat for America!""";;;

1715 rows × 2 columns

As it was mentioned before, the outcome of every removal command is being displayed in a new column next to the original one. The new column is equal to the previous with the application of the wanted function.

2.2.3 Punctuation removal

The next step of the data preprocessing is the punctuation removal. Special characters like '!, @, #, $, %, ^, &, *, (,), :, ;, ', ",], {, } ,[, /, <, >, |, \' etc., and also numerical characters are being removed. After the application of this procedure, the text is left with only alphabetical characters and without any punctuation marks and numbers. To do this, a new function named 'clean' is being created and applied to the 'clean url' column.

Figure 14

Function for punctuation removal

```
#function to clean the text
def clean(text):

#removes all special characters and numericals leaving the alphabets
    text = re.sub('[^A-Za-z]+', ' ', text)
    return text
```

The results are being displayed in the new column named ‘clean_punct’.

Figure 15

Tweets after punctuation removal

```
#cleaning the text
cleaned_df['clean_punct'] = cleaned_df['clean_url'].apply(clean)
cleaned_df
```

	text	clean_url	clean_punct
0	Study: #Bots amplified pandemic #misinformation on #socialmediaÃ□Â https://t.co/SUQQJWwCDE via @medcitynews #digitalhealth #medicalchatbot #ehealthInformation #hcsm;;;	Study: #Bots amplified pandemic #misinformation on #socialmediaÃ□Â	Study Bots amplified pandemic misinformation on socialmedia
1	"""RT @MonarchinMN: New Report Confirms How #IranÃ¢Â□Â□s Regime #FakeNews And #Propaganda Work"	"""RT @MonarchinMN: New Report Confirms How #IranÃ¢Â□Â□s Regime #FakeNews And #Propaganda Work"	RT MonarchinMN New Report Confirms How Iran s Regime FakeNews And Propaganda Work
2	"https://t.co/BuWallsLPxÃ□Â #infosec #cybersec #threatsÃ¢Â□Â¦""";;;"	"	
3	@condesm @jgnaredo @Katsuragui Seguro x ser un criadero de #bots y propagar #FakeNews;;;	@condesm @jgnaredo @Katsuragui Seguro x ser un criadero de #bots y propagar #FakeNews;;;	condesm jgnaredo Katsuragui Seguro x ser un criadero de bots y propagar FakeNews
4	"""RT @carlos2015aceve: Lo dijo @JhonnyKandanga en @NoticiasBqto"	"""RT @carlos2015aceve: Lo dijo @JhonnyKandanga en @NoticiasBqto"	RT carlos aceve Lo dijo JhonnyKandanga en NoticiasBqto
...	...	...	...
1710	"""@mtgreenee Republican lies up 79%"	"""@mtgreenee Republican lies up 79%"	mtgreenee Republican lies up
1711	Republican conspiracy theories up 98%	Republican conspiracy theories up 98%	Republican conspiracy theories up
1712	Republican insurrection up 100%	Republican insurrection up 100%	Republican insurrection up
1713	Republicans are spreading more COVID misinformation than ever before. Like no one has ever seen.	Republicans are spreading more COVID misinformation than ever before. Like no one has ever seen.	Republicans are spreading more COVID misinformation than ever before Like no one has ever seen
1714	"* Please RT * Vote Democrat for America!""";;;"	"* Please RT * Vote Democrat for America!""";;;"	Please RT Vote Democrat for America

1715 rows × 3 columns

2.2.4 Tokenization

After the punctuation and numerical removal, comes the tokenization. Tokenization is a vital task in Natural Language Processing ("What is Tokenization | Tokenization In NLP", 2021). It is a process of converting “a sequence of characters into a sequence of tokens” ("Lexical analysis - Wikipedia", 2021). Thus meaning, separating raw text that can be a sentence, a paragraph, a phrase or even an entire text document into individual words. For example, the phrase: “Today is my birthday” after the application of tokenization would be: [‘Today’, ‘is’, ‘my’, ‘birthday’]. The use of

tokenization is to make further stages of NLP easier. For instance, in machine learning approaches, tokenization is the most important step because tokens are what machines read ("Tokenizers: How machines read", 2021). Since machines can't read any text data, the words are being converted into numbers.

The primary step for this to happen, is for the text to be broke into individual words. In lexicon-based approaches -as the approach that this project is based on- tokenization is being conducted as an early-stage process. The rest of pre-processing techniques like stop words removal, POS-tag, lemmatization, and determining subjectivity and polarity, require words (tokens) for their functions to be applied and not raw text. Libraries in the stop word removal stage are matching words with words, the POS-tag is referring in words, lemmatization, subjectivity, and polarity also require words as their input data. In this project, word level tokenization will be performed with the use of the nltk tokenize function: '*word_tokenize()*'. First this function must be imported from 'nltk.tokenize'.

Figure 16

Function for tokenization

```
import nltk
nltk.download('punkt')

from nltk.tokenize import word_tokenize

from nltk import pos_tag

nltk.download('stopwords')
from nltk.corpus import stopwords

nltk.download('wordnet')
from nltk.corpus import wordnet
```

The application of tokenization technique, POS-tag -that is being analyzed in the next paragraph- and stop words removal is being performed altogether by one algorithm and by applying one function: 'token_stop_pos' to the 'clean_punct' column.

2.2.5 POS-tag

One of the most vital and important step in Natural Language Processing (NLP) and in data cleaning is the Part-Of-Speech (POS) tagging("NLP Guide: Identifying Part of Speech Tags using Conditional Random Fields", 2021). POS-tagging that is also called grammatical tagging, is the process of converting a sentence to list of words and assigning the appropriate tag to each word of the sentence. The tag signifies whether the word is a noun, adjective, adverb, verb etc. ("NLP | Part of Speech - Default Tagging -

GeeksforGeeks", 2021). It is a function of NLTK which uses features like previous word, next word, and the substance of the sentence to determine the grammatical tagging of the words (tokens) ("Learning POS Tagging & Chunking in NLP", 2021). It is performed after tokenization process as the software uses individual words (tokens) to do the tag assignation.POS tagging is also very crucial for the lemmatization process that is performed in next steps and is used to convert a word to its' root form ("NLP Guide: Identifying Part of Speech Tags using Conditional Random Fields", 2021). The assigned part-of-speech tag explains how the particular word is being used in the sentence. The main part of speeches and some examples are being displayed on the table above.

Table 2

Tag meaning and examples

Tag	Meaning	English Examples
ADJ	adjective	*new, good, high, special, big, local*
ADP	adposition	*on, off, at, with, by, into, under*
ADV	adverb	*really, already, still, early, now*
CONJ	conjunction	*and or, but, if, while, although*
DET	determiner, article	*the, a, some, most, every, no, which*
NOUN	noun	*year, home, costs, time, Africa*
NUM	numeral	*twenty-four, fourth, 1991, 14:24*
PRT	particle	*at, on, out, over per, that, up, with*
PRON	pronoun	*he, their, her, its, my, I, us*
VERB	verb	*is, say, told, given, playing, would*
.	punctuation marks	*. , ; !*
X	other	*ersatz, esprit, dunno, gr8, university*

Part-Of-Speech tagging is performed through Natural Language Toolkit (NLTK) in Python. In order to start the process, the foremost steps are:

- import nltk
- download required nltk packages with *'nltk.download()*'

- in the case of POS-tag the *'nltk.download('punkt')*' is used because is required for tokenization
- function *'pos_tag'* is imported in order to return the list of tuples in each word.

2.2.6 Stop Words Removal

The next step is for the stop words to be removed. Stop words are the most common words in the natural language ("How To Remove Stopwords In Python | Stemming and Lemmatization", 2021).Are words that don't have any usage, meaning that can't provide any sentiment to the analysis. Some name them as useless data. ("Removing stop words with NLTK in Python - GeeksforGeeks", 2021). Some of the most common stopwords are: 'the', 'I', 'and', 'in', 'a', 'an', 'me', 'you', 'we' etc. They must be removed so words with bigger subjectivity remain, and the cleaning process doesn't waste time to analyze useless words. Inorder to remove the stopwords, Python provides NLTK (Natural Language Toolkit) which contains several stopwords in 16 different languages. This is accessible by calling a simple command to show which stopwords are going to be removed.

Figure 17

List of stop words in English

```
import nltk
from nltk.corpus import stopwords
print(stopwords.words('english'))
```

```
['i', 'me', 'my', 'myself', 'we', 'our', 'ours', 'ourselves', 'you', "you're", "you've", "you'll", "you'd", 'your', 'yours', 'y
ourself', 'yourselves', 'he', 'him', 'his', 'himself', 'she', "she's", 'her', 'hers', 'herself', 'it', "it's", 'its', 'itself',
'they', 'them', 'their', 'theirs', 'themselves', 'what', 'which', 'who', 'whom', 'this', 'that', "that'll", 'these', 'those',
'am', 'is', 'are', 'was', 'were', 'be', 'been', 'being', 'have', 'has', 'had', 'having', 'do', 'does', 'did', 'doing', 'a', 'a
n', 'the', 'and', 'but', 'if', 'or', 'because', 'as', 'until', 'while', 'of', 'at', 'by', 'for', 'with', 'about', 'against', 'b
etween', 'into', 'through', 'during', 'before', 'after', 'above', 'below', 'to', 'from', 'up', 'down', 'in', 'out', 'on', 'of
f', 'over', 'under', 'again', 'further', 'then', 'once', 'here', 'there', 'when', 'where', 'why', 'how', 'all', 'any', 'both',
'each', 'few', 'more', 'most', 'other', 'some', 'such', 'no', 'nor', 'not', 'only', 'own', 'same', 'so', 'than', 'too', 'very',
's', 't', 'can', 'will', 'just', 'don', "don't", 'should', "should've", 'now', 'd', 'll', 'm', 'o', 're', 've', 'y', 'ain', 'ar
en', "aren't", 'couldn', "couldn't", 'didn', "didn't", 'doesn', "doesn't", 'hadn', "hadn't", 'hasn', "hasn't", 'haven', "have
n't", 'isn', "isn't", 'ma', 'mightn', "mightn't", 'mustn', "mustn't", 'needn', "needn't", 'shan', "shan't", 'shouldn', "should
n't", 'wasn', "wasn't", 'weren', "weren't", 'won', "won't", 'wouldn', "wouldn't"]
```

Those are all the stopwords in Natural Language Toolkit in the English language. Every word in the dataset that matches any of the above stopwords is being removed. To apply the function for the stopwords removal, first the stopwords must be downloaded through: *'nlt.download('stopwords')'*and then stopwords must be imported from nltk.corpus as seen in figure 16

As was said before, the application of the stopwords removal function is performed with the one of tokenization and POS-tag. The algorithm for those 3 techniques is being applied in the ‘clean_punct’ column.

Figure 18

Function for tokenization, POS-tagging and stop words removal

```
#create function for tokenization, pos-tag, stopwords removal
pos_dict = {'J':wordnet.ADJ, 'V':wordnet.VERB, 'N':wordnet.NOUN, 'R':wordnet.ADV}
def token_stop_pos(text):
    tags = pos_tag(word_tokenize(text))
    newlist = []
    for word, tag in tags:
        if word.lower() not in set(stopwords.words('english')):
            newlist.append(tuple([word, pos_dict.get(tag[0])]))
    return newlist
```

```
#apply function
cleaned_df['POS'] = cleaned_df['clean_punct'].apply(token_stop_pos)
cleaned_df
```

The results that are subjectivity high individual words with their POS assigned, are being stored in a new column called ‘POS’.

Figure 19

Tweets after POS-tagging

	text	clean_url	clean_punct	POS
0	Study: #Bots amplified pandemic #misinformation on #socialmediaÃ□Â https://t.co/SUQQJWwCDE via @medcitynews #digitalhealth #medicalchatbot #ehealthInformation #hcsm;;;	Study: #Bots amplified pandemic #misinformation on #socialmediaÃ□Â	Study Bots amplified pandemic misinformation on socialmedia	[(Study, n), (Bots, n), (amplified, v), (pandemic, a), (misinformation, n), (socialmedia, n)]
1	"""RT @MonarchinMN: New Report Confirms How #IranÃ¢Â□Â□s Regime #FakeNews And #Propaganda Work"	"""RT @MonarchinMN: New Report Confirms How #IranÃ¢Â□Â□s Regime #FakeNews And #Propaganda Work"	RT MonarchinMN New Report Confirms How Iran s Regime FakeNews And Propaganda Work	[(RT, n), (MonarchinMN, n), (New, n), (Report, n), (Confirms, n), (Iran, n), (Regime, n), (FakeNews, n), (Propaganda, n), (Work, n)]
2	"https://t.co/BuWallsLPxÃ□Â #infosec #cybersec #threatsÃ¢Â□Â¦"";;;"	"		[]
3	@condesm @jgnaredo @Katsuragui Seguro x ser un criadero de #bots y propagar #FakeNews;;;	@condesm @jgnaredo @Katsuragui Seguro x ser un criadero de #bots y propagar #FakeNews;;;	condesm jgnaredo Katsuragui Seguro x ser un criadero de bots y propagar FakeNews	[(condesm, n), (jgnaredo, n), (Katsuragui, n), (Seguro, n), (x, n), (ser, n), (un, a), (criadero, n), (de, None), (bots, None), (propagar, n), (FakeNews, n)]
4	"""RT @carlos2015aceve: Lo dijo @JhonnyKandanga en @NoticiasBqto"	"""RT @carlos2015aceve: Lo dijo @JhonnyKandanga en @NoticiasBqto"	RT carlos aceve Lo dijo JhonnyKandanga en NoticiasBqto	[(RT, n), (carlos, n), (aceve, v), (Lo, n), (dijo, n), (JhonnyKandanga, n), (en, v), (NoticiasBqto, n)]
...	...	...	...	...
1710	"""@mtgreenee Republican lies up 79%"	"""@mtgreenee Republican lies up 79%"	mtgreenee Republican lies up	[(mtgreenee, a), (Republican, n), (lies, v)]
1711	Republican conspiracy theories up 98%	Republican conspiracy theories up 98%	Republican conspiracy theories up	[(Republican, a), (conspiracy, n), (theories, n)]
1712	Republican insurrection up 100%	Republican insurrection up 100%	Republican insurrection up	[(Republican, a), (insurrection, n)]
1713	Republicans are spreading more COVID misinformation than ever before. Like no one has ever seen.	Republicans are spreading more COVID misinformation than ever before. Like no one has ever seen.	Republicans are spreading more COVID misinformation than ever before Like no one has ever seen	[(Republicans, n), (spreading, v), (COVID, n), (misinformation, n), (ever, r), (Like, None), (one, n), (ever, r), (seen, v)]
1714	"* Please RT * Vote Democrat for America!"";;;"	"* Please RT * Vote Democrat for America!"";;;"	Please RT Vote Democrat for America	[(Please, v), (RT, n), (Vote, n), (Democrat, n), (America, n)]

1715 rows × 4 columns

2.2.7 Lemmatization

The last step of data cleaning is the lemmatization process. Lemmatization is the process of grouping together words with inflectional endings only to return the base of a word that is known as lemma. The process is aiming to group together all inflected forms of a single word so they can be analyzed as a single item. For example, goes, going, gone are all different forms of the word 'go', therefore 'go' is the lemma of all these words and would be the result of the lemmatization. In addition, sometimes the same word can have different lemmas. In cases like this the part-of-speech (POS) tag of the word must be identified in order to extract the appropriate lemma. This is done beforehand in the previous step. In order to lemmatize, 'WordNetLemmatizer' from 'nltk.stem' must be imported. Additionally, a new function lemmatize() must be created to be able to lemmatize the POS data of the 'POS' column and store the results in a new column called 'lemma'.

Figure 20

Function for lemmatization

```
#import WordNetLemmatizer
from nltk.stem import WordNetLemmatizer
wordnet_lemmatizer = WordNetLemmatizer()
```

```
#create the lemmatize function
def lemmatize(pos_data):
    lemma_rew = " "
    for word, pos in pos_data:
        if not pos:
            lemma = word
            lemma_rew = lemma_rew + " " + lemma
        else:
            lemma = wordnet_lemmatizer.lemmatize(word, pos=pos)
            lemma_rew = lemma_rew + " " + lemma
    return lemma_rew
```

```
#apply the function
cleaned_df['lemma'] = cleaned_df['POS'].apply(lemmatize)
cleaned_df
```

The output of the lemmatize function gives the final dataset of words which will be the input data for the sentiment analysis.

Figure 21

Tweets after lemmatization

	text	clean_url	clean_punct	POS	lemma
0	Study: #Bots amplified pandemic #misinformation on #socialmediaÃ□Â https://t.co/SUQQJWwCDE via @medcitynews #digitalhealth #medicalchatbot #ehealthInformation #hcsm;;;	Study: #Bots amplified pandemic #misinformation on #socialmediaÃ□Â	Study Bots amplified pandemic misinformation on socialmedia	[(Study, n), (Bots, n), (amplified, v), (pandemic, a), (misinformation, n), (socialmedia, n)]	Study Bots amplify pandemic misinformation socialmedia
1	"""RT @MonarchinMN: New Report Confirms How #IranÃ¢Â□Â□s Regime #FakeNews And #Propaganda Work"	"""RT @MonarchinMN: New Report Confirms How #IranÃ¢Â□Â□s Regime #FakeNews And #Propaganda Work"	RT MonarchinMN New Report Confirms How Iran s Regime FakeNews And Propaganda Work	[(RT, n), (MonarchinMN, n), (New, n), (Report, n), (Confirms, n), (Iran, n), (Regime, n), (FakeNews, n), (Propaganda, n), (Work, n)]	RT MonarchinMN New Report Confirms Iran Regime FakeNews Propaganda Work
2	"https://t.co/BuWaIlsLPxÃ□Â #infosec #cybersec #threatsÃ¢Â□Â¦"";;;	"		[]	
3	@condesm @jgnaredo @Katsuragui Seguro x ser un criadero de #bots y propagar #FakeNews;;;	@condesm @jgnaredo @Katsuragui Seguro x ser un criadero de #bots y propagar #FakeNews;;;	condesm jgnaredo Katsuragui Seguro x ser un criadero de bots y propagar FakeNews	[(condesm, n), (jgnaredo, n), (Katsuragui, n), (Seguro, n), (x, n), (ser, n), (un, a), (criadero, n), (de, None), (bots, None), (propagar, n), (FakeNews, n)]	condesm jgnaredo Katsuragui Seguro x ser un criadero de bots propagar FakeNews
4	"""RT @carlos2015aceve: Lo dijo @JhonnyKandanga en @NoticiasBqto"	"""RT @carlos2015aceve: Lo dijo @JhonnyKandanga en @NoticiasBqto"	RT carlos aceve Lo dijo JhonnyKandanga en NoticiasBqto	[(RT, n), (carlos, n), (aceve, v), (Lo, n), (dijo, n), (JhonnyKandanga, n), (en, v), (NoticiasBqto, n)]	RT carlos aceve Lo dijo JhonnyKandanga en NoticiasBqto
...	...	...	...	...	...
1710	"""@mtgreenee Republican lies up 79%"	"""@mtgreenee Republican lies up 79%"	mtgreenee Republican lies up	[(mtgreenee, a), (Republican, n), (lies, v)]	mtgreenee Republican lie
1711	Republican conspiracy theories up 98%	Republican conspiracy theories up 98%	Republican conspiracy theories up	[(Republican, a), (conspiracy, n), (theories, n)]	Republican conspiracy theory
1712	Republican insurrection up 100%	Republican insurrection up 100%	Republican insurrection up	[(Republican, a), (insurrection, n)]	Republican insurrection
1713	Republicans are spreading more COVID misinformation than ever before. Like no one has ever seen.	Republicans are spreading more COVID misinformation than ever before. Like no one has ever seen.	Republicans are spreading more COVID misinformation than ever before Like no one has ever seen	[(Republicans, n), (spreading, v), (COVID, n), (misinformation, n), (ever, r), (Like, None), (one, n), (ever, r), (seen, v)]	Republicans spread COVID misinformation ever Like one ever see
1714	"* Please RT * Vote Democrat for America!"";;;"	"* Please RT * Vote Democrat for America!"";;;"	Please RT Vote Democrat for America	[(Please, v), (RT, n), (Vote, n), (Democrat, n), (America, n)]	Please RT Vote Democrat America

1715 rows × 5 columns

2.3 Sentiment Analysis

When the data cleaning process is finished, the dataset is ready to be used for the determination of the sentiment analysis score. Every line of tweets, now contains tokens that are being used as individual words. The polarity of every word in a sentence (tweet) is being calculated and added, for the total sentiment of the tweet to be determined. Thus, the sentiment of one tweet depends on the sentiment of each word that consist of.

The sentiment score comes after the determination of subjectivity and polarity of every tweet. Sentiment refers to the opinion and emotion of a word. Meaning that is subjective expression to a fact (Sims, 2021). Text subjectivity is a measure of how subjective or objective a statement is. Subjectivity refers to personal expressions, feelings, and emotions of the writer of the text. Simply, subjectivity shows how sentimental a text is. It lays in the range of [0, 1]. As closest to '1' the float is, the most subjective the sentence is. For example, the sentence 'I like coffee' is a subjective sentence that is expressing an opinion about an entity. The sentence ' I think that he went home' also is a subjective sentence but it doesn't contain any sentiment in it. In

this example the subjectivity rate would be '0'. Contrary to objectivity that doesn't refer to personal opinion and expressed feeling ("Objective vs Subjective", 2021). An objective statement for instance has true information but not expressed emotion for that information. Comparing the two definitions, an objective sentence would be: 'Today it is raining'. The subjective version of this sentence would be: 'Unfortunately, today it is raining'. Thus, the determination of objectivity has no use for sentimental classification and is not being used in this project.

Another step that is necessary for the sentiment analysis is the determination of the polarity. The term sentiment polarity of an element stands for the orientation of the expressed sentiment (Kumar & Gupta, 2021).The sentimental polarity can be expressed as positive, neutral, or negative. The polarity can take floats between the range of [-1, 1].

- [-1, 0] → Negative sentiment
- [0] → Neutral sentiment
- [0, 1] → Positive sentiment

2.3.1 TextBlob

TextBlob is an open-source Python library that performs a variety of NLP tasks ("TextBlob Sentiment: Calculating Polarity and Subjectivity", 2021). It processes textual data and performs tasks such as Part-Of-Speech tagging, sentiment analysis, classification etc. ("TextBlob: Simplified Text Processing — TextBlob 0.16.0 documentation", 2021).

To determine subjectivity and polarity on the dataset in this project, TextBlob is being used and its sentiment function that returns subjectivity and polarity. The first step is the installation by the command *'conda install textblob'*. After, the library must be imported together with the creation of the subjectivity and polarity functions.

Figure 22

Functions for subjectivity/polarity

```
#create functions for subjectivity and polarity
from textblob import TextBlob

def getSubjectivity(text):
    return TextBlob(text).sentiment.subjectivity

def getPolarity(text):
    return TextBlob(text).sentiment.polarity
```

After the import, these two functions must be applied to the 'lemma' column in order to determine the subjectivity and polarity scores of the tokens.

Figure 23

Creation of new columns for subjectivity/polarity

```
#aply functions
cleaned_df['Subjectivity'] = cleaned_df['lemma'].apply(getSubjectivity)
cleaned_df['Polarity'] = cleaned_df['lemma'].apply(getPolarity)

cleaned_df
```

The results are automatically saved in a new column named after each function.

Figure 24

Determination of Subjectivity and Polarity

	text	clean_url	clean_punct	POS	lemma	Subjectivity	Polarity
0	Study: #Bots amplified pandemic #misinformation on #socialmediaÃ□Â https://t.co/SUQQJWwCDE via @medcitynews #digitalhealth #medicalchatbot #ehealthInformation #hcsm;;;	Study: #Bots amplified pandemic #misinformation on #socialmediaÃ□Â	Study Bots amplified pandemic misinformation on socialmedia	[(Study, n), (Bots, n), (amplified, v), (pandemic, a), (misinformation, n), (socialmedia, n)]	Study Bots amplify pandemic misinformation socialmedia	0.000000	0.000000
1	"""RT @MonarchinMN: New Report Confirms How #IranÃ¢Â□Â□s Regime #FakeNews And #Propaganda Work"	"""RT @MonarchinMN: New Report Confirms How #IranÃ¢Â□Â□s Regime #FakeNews And #Propaganda Work"	RT MonarchinMN New Report Confirms How Iran s Regime FakeNews And Propaganda Work	[(RT, n), (MonarchinMN, n), (New, n), (Report, n), (Confirms, n), (Iran, n), (Regime, n), (FakeNews, n), (Propaganda, n), (Work, n)]	RT MonarchinMN New Report Confirms Iran Regime FakeNews Propaganda Work	0.277273	0.018182
2	"https://t.co/BuWaIlsLPxÃ□Â #infosec #cybersec #threatsÃ¢Â□Â□""";;;"	"		[]		0.000000	0.000000
3	@condesm @jgnaredo @Katsuragui Seguro x ser un criadero de #bots y propagar #FakeNews;;;	@condesm @jgnaredo @Katsuragui Seguro x ser un criadero de #bots y propagar #FakeNews;;;	condesm jgnaredo Katsuragui Seguro x ser un criadero de bots y propagar FakeNews	[(condesm, n), (jgnaredo, n), (Katsuragui, n), (Seguro, n), (x, n), (ser, n), (un, a), (criadero, n), (de, None), (bots, None), (propagar, n), (FakeNews, n)]	condesm jgnaredo Katsuragui Seguro x ser un criadero de bots propagar FakeNews	0.000000	0.000000
4	"""RT @carlos2015aceve: Lo dijo @JhonnyKandanga en @NoticiasBqto"	"""RT @carlos2015aceve: Lo dijo @JhonnyKandanga en @NoticiasBqto"	RT carlos aceve Lo dijo JhonnyKandanga en NoticiasBqto	[(RT, n), (carlos, n), (aceve, v), (Lo, n), (dijo, n), (JhonnyKandanga, n), (en, v), (NoticiasBqto, n)]	RT carlos aceve Lo dijo JhonnyKandanga en NoticiasBqto	0.000000	0.000000
...	...	...	...	...	...	...	...
1710	"""@mtgreenee Republican lies up 79%"	"""@mtgreenee Republican lies up 79%"	mtgreenee Republican lies up	[(mtgreenee, a), (Republican, n), (lies, v)]	mtgreenee Republican lie	0.000000	0.000000
1711	Republican conspiracy theories up 98%	Republican conspiracy theories up 98%	Republican conspiracy theories up	[(Republican, a), (conspiracy, n), (theories, n)]	Republican conspiracy theory	0.000000	0.000000
1712	Republican insurrection up 100%	Republican insurrection up 100%	Republican insurrection up	[(Republican, a), (insurrection, n)]	Republican insurrection	0.000000	0.000000
1713	Republicans are spreading more COVID misinformation than ever before. Like no one has ever seen.	Republicans are spreading more COVID misinformation than ever before. Like no one has ever seen.	Republicans are spreading more COVID misinformation than ever before Like no one has ever seen	[(Republicans, n), (spreading, v), (COVID, n), (misinformation, n), (ever, r), (Like, None), (one, n), (ever, r), (seen, v)]	Republicans spread COVID misinformation ever Like one ever see	0.000000	0.000000
1714	"* Please RT * Vote Democrat for America!"";;;"	"* Please RT * Vote Democrat for America!"";;;"	Please RT Vote Democrat for America	[(Please, v), (RT, n), (Vote, n), (Democrat, n), (America, n)]	Please RT Vote Democrat America	0.000000	0.000000

1715 rows × 7 columns

2.3.2 Sentimental classification

Continuing to the last step of the sentiment analysis, the sentiment score must be determined based on the polarity. For this to happen, a new function must be created. It will get the polarity of the tweet as an input, and it will give the sentimental score of every tweet. If polarity score is <0 then the sentiment is negative, if it is ==0 then it is neutral and if it is >0 then it is positive. In addition, in this project this function will be applied in the 'Polarity' column and will give the final sentiment score in a new column called 'Analysis'.

Figure 25

Function for semantic classification

```
#create function for defining sentiment
def getAnalysis(score):
    if score < 0:
        return 'Negative'
    elif score == 0 :
        return 'Neutral'
    else:
        return 'Positive'
```

```
#apply function
cleaned_df['Analysis'] = cleaned_df['Polarity'].apply(getAnalysis)

cleaned_df
```

Figure 26

Semantic Classification of tweets

	text	clean_url	clean_punct	POS	lemma	Subjectivity	Polarity	Analysis
0	Study: #Bots amplified pandemic #misinformation on #socialmediaÃ□Â https://t.co/SUQQJWwCDE via @medcitynews #digitalhealth #medicalchatbot #ehealthInformation #hcsm;;;	Study: #Bots amplified pandemic #misinformation on #socialmediaÃ□Â	Study Bots amplified pandemic misinformation on socialmedia	[(Study, n), (Bots, n), (amplified, v), (pandemic, a), (misinformation, n), (socialmedia, n)]	Study Bots amplify pandemic misinformation socialmedia	0.000000	0.000000	Neutral
1	"""RT @MonarchinMN: New Report Confirms How #IranÃ¢Â□Â□s Regime #FakeNews And #Propaganda Work"	"""RT @MonarchinMN: New Report Confirms How #IranÃ¢Â□Â□s Regime #FakeNews And #Propaganda Work"	RT MonarchinMN New Report Confirms How Iran s Regime FakeNews And Propaganda Work	[(RT, n), (MonarchinMN, n), (New, n), (Report, n), (Confirms, n), (Iran, n), (Regime, n), (FakeNews, n), (Propaganda, n), (Work, n)]	RT MonarchinMN New Report Confirms Iran Regime FakeNews Propaganda Work	0.277273	0.018182	Positive
2	"https://t.co/BuWaIlsLPxÃ□Â #infosec #cybersec #threatsÃ¢Â□Â¦"""...." ;;;	"		[]		0.000000	0.000000	Neutral
3	@condesm @jgnaredo @Katsuragui Seguro x ser un criadero de #bots y propagar #FakeNews;;;	@condesm @jgnaredo @Katsuragui Seguro x ser un criadero de #bots y propagar #FakeNews;;;	condesm jgnaredo Katsuragui Seguro x ser un criadero de bots y propagar FakeNews	[(condesm, n), (jgnaredo, n), (Katsuragui, n), (Seguro, n), (x, n), (ser, n), (un, a), (criadero, n), (de, None), (bots, None), (propagar, n), (FakeNews, n)]	condesm jgnaredo Katsuragui Seguro x ser un criadero de bots propagar FakeNews	0.000000	0.000000	Neutral
4	"""RT @carlos2015aceve: Lo dijo @JhonnyKandanga en @NoticiasBqto"	"""RT @carlos2015aceve: Lo dijo @JhonnyKandanga en @NoticiasBqto"	RT carlos aceve Lo dijo JhonnyKandanga en NoticiasBqto	[(RT, n), (carlos, n), (aceve, v), (Lo, n), (dijo, n), (JhonnyKandanga, n), (en, v), (NoticiasBqto, n)]	RT carlos aceve Lo dijo JhonnyKandanga en NoticiasBqto	0.000000	0.000000	Neutral
...	...	...	...	...	...	...	...	...
1710	"""@mtgreenee Republican lies up 79%"	"""@mtgreenee Republican lies up 79%"	mtgreenee Republican lies up	[(mtgreenee, a), (Republican, n), (lies, v)]	mtgreenee Republican lie	0.000000	0.000000	Neutral
1711	Republican conspiracy theories up 98%	Republican conspiracy theories up 98%	Republican conspiracy theories up	[(Republican, a), (conspiracy, n), (theories, n)]	Republican conspiracy theory	0.000000	0.000000	Neutral

3 Data Analysis

3.1 Data Visualization

To be able to better understand the outcome of every research, the best solution is to visualize the data. Data visualization is the graphical representation of information ("What Is Data Visualization? Definition, Examples, And Learning Resources", 2021). With the use of charts, pies, graphs and any other visual elements, there is a better understanding and an easier identification of patterns and outcomes in large datasets ("What is data visualization and why is it important?", 2021). A very useful and widespread way to visualize data after the sentimental classification is the word clouds. Word clouds are images including words (tokens) that are the result of data analysis methods. There are different ways to create word clouds such as Microsoft 8, Business Intelligence Tools, online tools etc. ("Word Cloud in Python | How to Build Word Cloud in Python?", 2021).The practice of translating text data into visual data in this project is done with the use of appropriate Python libraries such as WordCloud and Matplotlib.

3.1.1 WordCloud& Matplotlib

WordCloud and Matplotlib are both Python libraries that are used to visualize datasets and make the result livelier. WordCloud gives graphical representations of words of a data set depending on their frequency. The more the use of a word in a text, the bigger and the bolder the word will be presented in the word cloud. Thus meaning, the size of each word in the word cloud depends on the frequency and the importance of the word ("Generating Word Cloud in Python - GeeksforGeeks", 2021). The steps that had been followed in this project to create a word cloud in Python are:

1. Importation of necessary libraries.

Figure 27

Importation of libraries for data visualization

```
import nltk
from nltk.corpus import stopwords
from wordcloud import WordCloud, STOPWORDS, ImageColorGenerator
import matplotlib.pyplot as plt
import numpy as np
from PIL import Image
```

2. Selection of the text for the word cloud. Here the column ‘lemma’ from the dataset is selected as text. Words from only this column will be presented in the word cloud as only in that column are fully cleaned.
3. Addition of the text from step 2, into a new variable. The new variable is called text and is being used further in the code.
4. Plotting and customizing the word cloud. The.*imshow()* method of matploblib.pyplot is used in order to display the word cloud as an image. Another argument from.*imshow()* that is being used in the code is the *interpolation='bilinear'* for a better image quality. Other customizations can be added too like the background color or the size of the word cloud image.

Figure 28

WordCloud customizations

```
# Create stopword list:
#stopwords = set(STOPWORDS)
#stopwords.update(["br", "href"])
text = " ".join(tweets for tweets in cleaned_df.lemma)
wordcloud = WordCloud(collocations = False).generate(text) #stopwords=stopwords
fig = plt.figure(figsize=(10,10))
plt.imshow(wordcloud, interpolation='bilinear')
plt.axis("off")
plt.savefig('wordcloud11.png')
plt.show()
```

Another interesting variation of this visualization technique is to create word clouds based on sentiments. Meaning that every word cloud contains words that refer to a specific sentiment. For instance, only positive, negative, or neutral words. Something like this can help the user to easier understand the gap of differentiation between sentiments by the most used words. In addition, the user can understand a lot about the strengths and the weakness of the product/service if the data set is based on a product or service review. For instance, words as tasty, good, great, fresh, useful, detailed, friendly and terrible, ignorance, bad, dirty, spoiled can give detailed customers' feelings and opinions without having to go through all the reviews. Furthermore, words like amazing, wonderful, lovely have bigger subjectivity than words like good and nice. This indicates a bigger percentage of positive and strongly opinioned sentiment.

Four different word clouds are being created in this project. All of them refer to the column 'lemma' of tweets' dataset. The first one includes words from the whole dataset before the categorization of sentimental orientation. The second one is created

based on words with negative sentiment, the third one is about the neutral sentiment and lastly, the fourth is about the positive sentiment.

For the sentiment-based word clouds to be created, an extra process must be done:

1. Creation of a new data frame and drop null values

```
#create a new dataframe to perform exploratory data analysis
tweets = cleaned_df
#droping null values
tweets.dropna(inplace=True)
```

2. Storage of each sentiment in a new string

```
#store each sentiment in a new variable
neg_sent = tweets[tweets['Analysis'] == "Negative"]
neut_sent = tweets[tweets['Analysis'] == "Neutral"]
pos_sent = tweets[tweets['Analysis'] == "Positive"]
```

3. Addition of all sentiment strings in one because WordCloud accepts a single string

```
#add all sentiment strings in one string because wordcloud takes a single string
tweets_sample = pd.concat([neg_sent,neut_sent,pos_sent], axis=0)
tweets_sample.reset_index(drop=True,inplace=True)
```

4. Separation of the data from the new string into negative, positive, and neutral

```
#split the data into positive,neutral and negative sentiment
negative_tweets = tweets_sample[tweets_sample['Analysis'].isin(["Negative"]) ]
positive_tweets = tweets_sample[tweets_sample['Analysis'].isin(["Positive"]) ]
neutral_tweets = tweets_sample[tweets_sample['Analysis'].isin(["Neutral"]) ]
```

5. Transformation of the data from previous step to a single string by using str.cat()

```
#tranform to single string..for this we are using str.cat()
negative_tweets_str = negative_tweets.lemma.str.cat()
positive_tweets_str = positive_tweets.lemma.str.cat()
neutral_tweets_str = neutral_tweets.lemma.str.cat()
```

6. Customization of the string of every sentiment with different color for better recognition

```
wordcloud_negative = WordCloud(background_color='black').generate(negative_tweets_str)
wordcloud_positive = WordCloud(background_color='oldlace').generate(positive_tweets_str)
wordcloud_neutral = WordCloud(background_color='lightblue').generate(neutral_tweets_str)
```

7. Generation of the word cloud by using wordcloud_negative, wordcloud_neutral, wordcloud_positive correspondingly

```
#plot #negative
fig = plt.figure(figsize=(10,10))
ax1 = fig.add_subplot()
ax1.imshow(wordcloud_negative, interpolation='bilinear')
ax1.axis("off")
ax1.set_title('Tweets with Negative Sentiment',fontsize=20)
```

With a further use of Python and the appropriate libraries, the total count value of each sentiment is being determined as well as the mean value of subjectivity and polarity in each sentiment too. A chart that displays every sentiment in comparison with the rest by using the total value count is being used as well to help with the visualization of the sentiment classification result and to answer the addressed problem of the project. For the creation of this chart, some extra coding is required as seen in figure 29.

Figure 29

Creation of sentiment chart

```
plt.figure(figsize=(10,5))
result=cleaned_df['Analysis'].value_counts()
result.plot(kind='bar', rot=0, color=['plum', 'cyan', 'black']);
```

Lastly, in addition to understanding the sentiment analysis report, subjectivity has a big part in analyzing sentimental score. A sentiment category may have the most tweets and thus the biggest sentiment count percentage within the rest, but it can be very low in subjectivity meaning that almost no sentiment was expressed. For this reason, a new chart is being created to accompany the result with the mean values of subjectivity and polarity. Furthermore, it will enlighten the relationship between every sentiment and its' expressed intensity. By using the code shown in figure 30, the above-mentioned diagram is being created.

Figure 30

Creation of subjectivity/polarity diagram

```
plt.figure(figsize=(10,6))
for i in range(0, cleaned_df.shape[0]):
    plt.scatter(cleaned_df['Polarity'][i], cleaned_df['Subjectivity'][i], color='Blue')

plt.title('Sentiment Analysis')
plt.xlabel('Polarity')
plt.ylabel('Subjectivity')
plt.show()
```

4 Results

The first step is to retrieve the count of each sentiment. This calculation is done for a more specific view of each sentiment, to count all negative, neutral, and positive scores. It can give a more detailed picture of the sentimental classification.

Figure 31

Total count of sentiments

```
#count sentiments
tb_counts = cleaned_df.Analysis.value_counts()

tb_counts
```

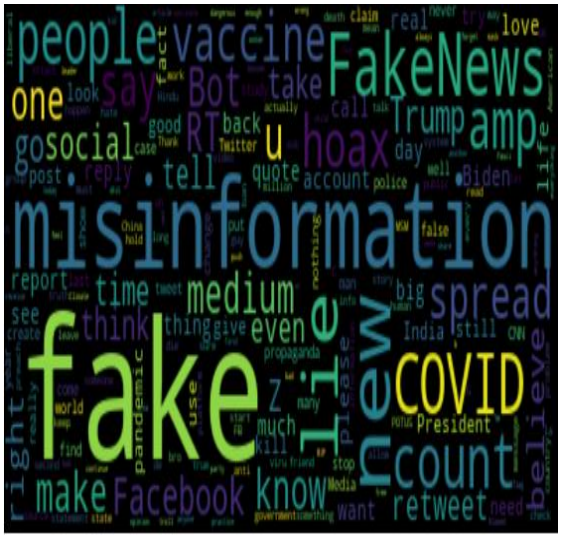
```
Neutral     780
Negative    561
Positive    374
Name: Analysis, dtype: int64
```

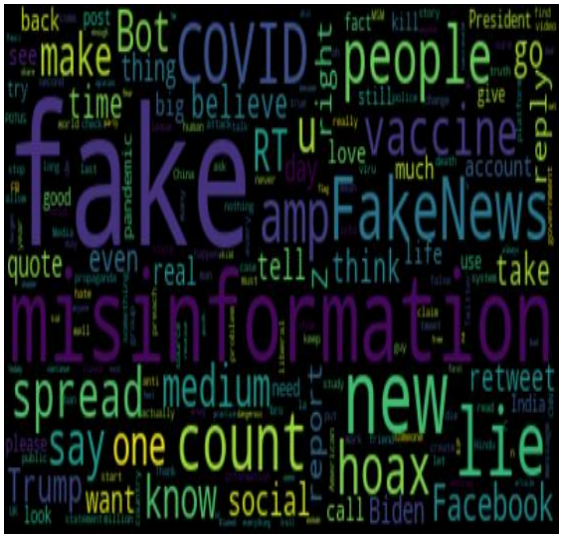
Neutral sentiment comes first with a significant difference between both negative and positive sentiment. That means that the majority of users expressed themselves in a more objective way without using personal opinion and emotion.

Word clouds were created in order to find out which the most used words were, and which words were sentimentally strong. The result of the code regarding to the first word cloud including all sentiments was an image with all words on the ‘lemma’ column. Every time that the code run, it produced a different image but the words with the biggest frequencies, remains the same in every result.

Figure 32

Word clouds based in all sentiments





Comparing word clouds of different sentiments referring to the same dataset can also sometimes show a small difference between every sentiment instead of a big one as it is waited. In this project for example. Three different word clouds were created regarding to every sentiment (negative, neutral, positive).

Figure 33

Word cloud based in negative sentiment

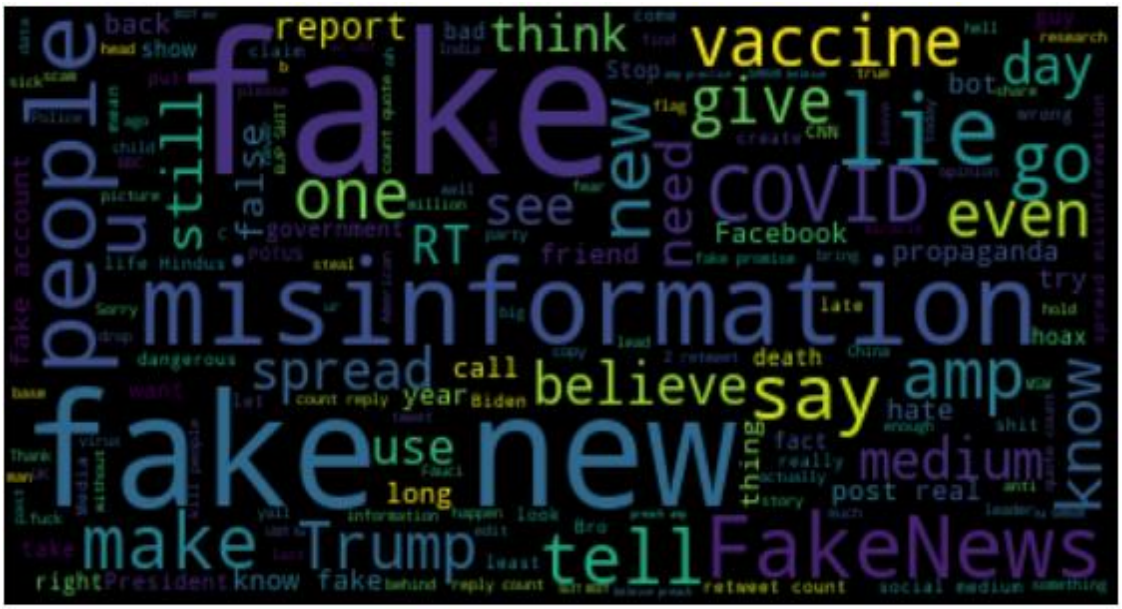


Figure 34

Word cloud based in neutral sentiment

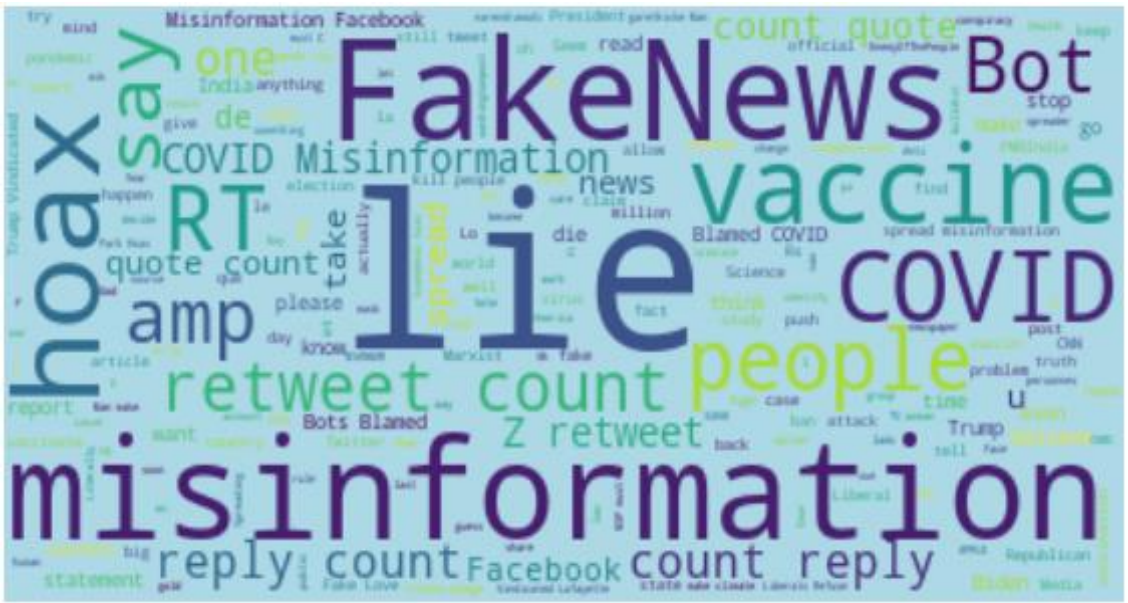
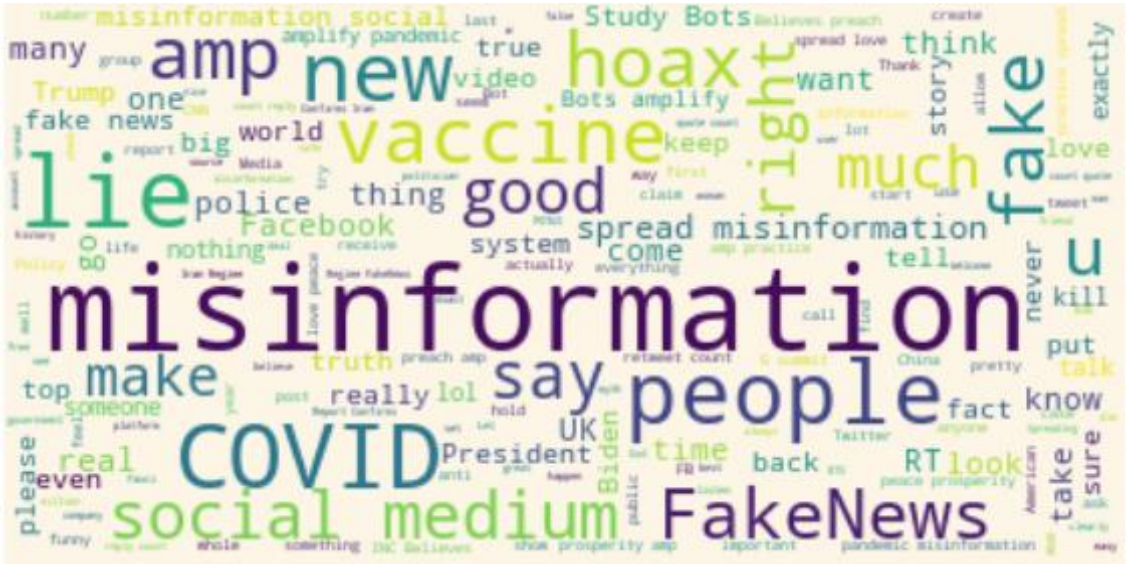


Figure 35

Word cloud based in positive sentiment



After comparing the above 3 images, it is evident that all of them consist of the same words in the majority. 'Fakenews', 'fake', 'misinformation', 'lie', 'people', 'covid', are words that are part of every word cloud but with a difference in their appearance. In the word cloud that includes words with negative sentiment, the words that are similar with the other 2 images, come out bigger and bolder meaning that they appear more frequently in the dataset and with a stronger emphasis. The most used word here is 'fake' with a big difference beside the rest of the words. This indicates a high subjectivity of negative sentiment.

Continuing with the neutral sentiment word cloud, same words are being seen but smaller and not so bold. This means that users didn't emphasis only in specific words, but they expressed themselves with the use of other words too. Only few words stand out. The rest of them are small and not bold, and the difference between their size and the size of the bold ones is big. That means that the users didn't express a strong opinion or they didn't express an opinion at all. It is a possibility to only comment on an event without giving a personal opinion. This is an indication for low subjectivity with no feeling involved in their tweets.

Lastly, regarding the word cloud with the positive sentiment, a more even allocation of words occurs. The majority of words have approximately the same frequency and words like 'good' and 'love' are being seen. Besides that, 'misinformation',

'lie', 'hoax' and 'FakeNews' own a big percentage of this dataset as well. Here most of words have the same size. Meaning that the subjectivity is dived evenly, and people didn't emphasize in specific keywords. The expressed opinion was more subjective than in neutral category but les subjective than the negative one.

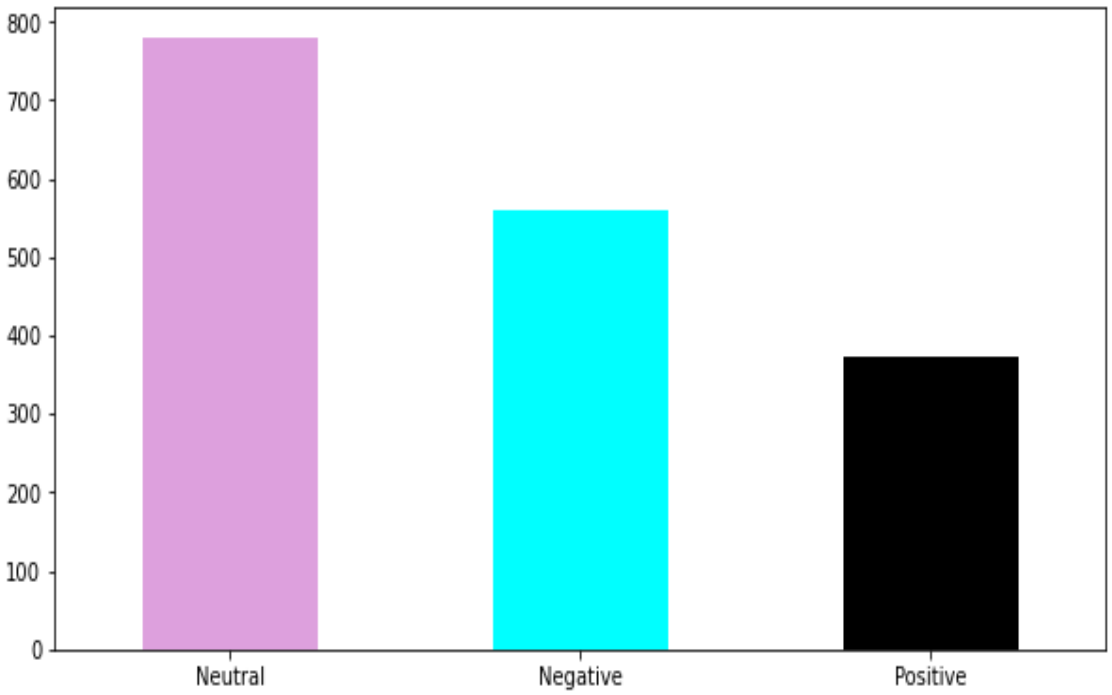
It is clear that users centered upon expressing their disappointment and negativity in their tweets. They strongly communicated their negative emotions about fake news but didn't emphasize to the topic of them.

Same words take place in all of 3 different sentiments' final dataset. The difference is in the frequency of the words, the users' expressed intensity and chosen vocabulary. These words (fake, lie, hoax, misinformation, fakenews etc.,) have the same sentiment score in the lexicon corpus that the sentiment analysis is based on. That means that the way users choose to use these words and the rest of the words that choose to surround them with, gives the final expressed subjectivity and polarity of the tweet. The word 'fake' for example, is categorized to negative sentiment in the dictionary, but is from the most used words in negative, neutral, and positive sentiment word cloud as well. This result occurs due to the combination of words in every category of the sentiment. In the combination of the neutral sentiment, the word 'fake' is surrounded with words that are not sentimentally high on anger or disappointment and contain more neutral opinions and sentiments. Therefore, the emphasis of the word drops. In the positive sentiment, the same word is between words that express more positive and happy feelings that change polarity of the outcome.

Another visualization technique with the use of Matplotlib was performed in the project. A chart was created showing the 3 sentiments in comparison to each other. The results were illustrated based on each sentiment's total value count.

Figure 36

Sentiments' chart



Neutral sentiment has the most values followed by negative and lastly positive. The difference between neutral and the rest two sentiments is big -especially with the positive one- indicating the lack of sentiment expressed by users in the specific category. As neutral sentiment often consists of objective sentences, having a result like this one, means that users didn't give much attention on being more subjectively expressed. Their opinions didn't include much personal view and was more informative than sentimental.

An interesting observation is that, even though the count of neutral sentiment tweets is the biggest one among all, based on the word cloud images the most intense expressed sentiment is the negative followed by the positive and the least strongly opinioned is the neutral. The majority of users were more passionate to express their negative opinion instead of any other. To ensure that statement, the command: *analysis.mean().sort_values(by="Polarity",ascending=False).head()* was used to find the mean values of all 3 sentiments regarding subjectivity and polarity.

Figure 37

Mean values of Subjectivity and Polarity

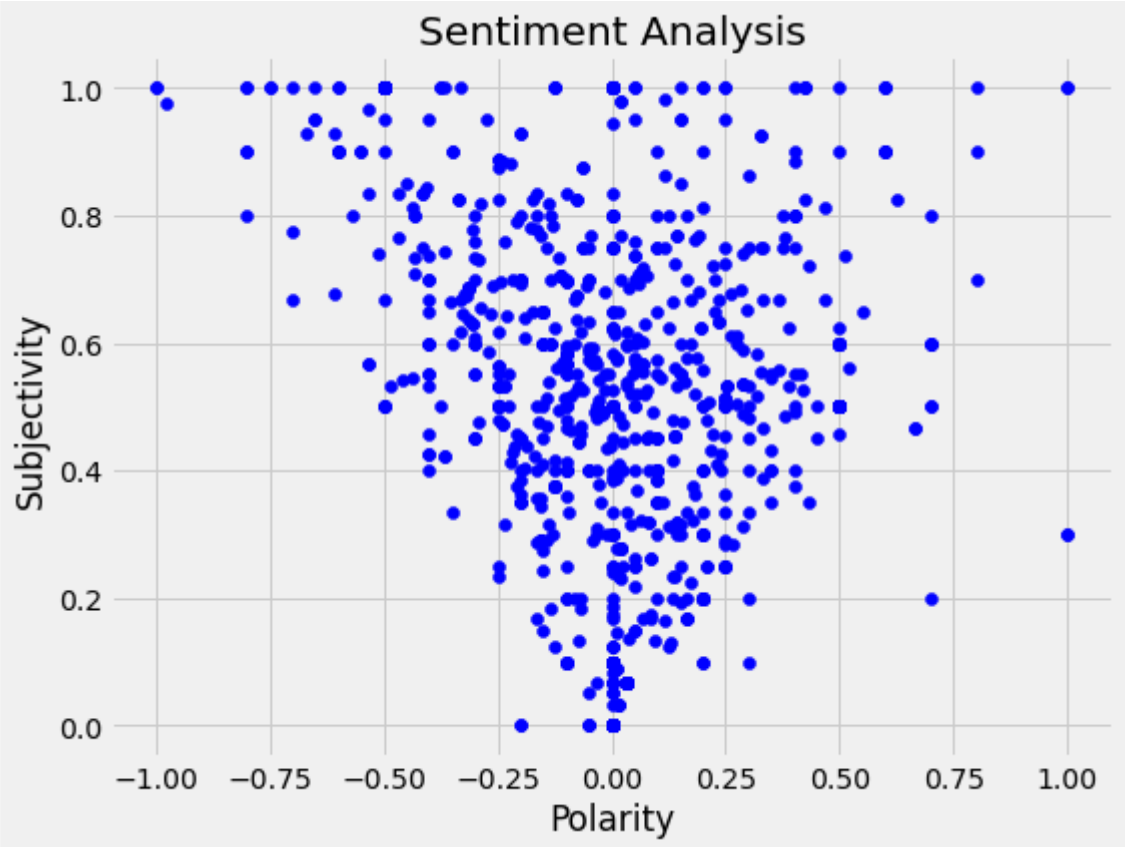
Analysis	Subjectivity	Polarity
Positive	0.500564	0.240635
Neutral	0.035771	0.000000
Negative	0.737790	-0.338230

This image comes to validate the results from the word clouds. Negative sentiment has the highest percentage of subjectivity with a huge difference with the neutral sentiment. Positive sentiment has also high subjectivity that is above average.

Subjectivity is a key factor for analyzing the intensity of the final sentiment scores. A chart was made consisting of polarity and subjectivity. On the x-axis are the polarity's possible floats and on the y-axis are the possible floats of subjectivity. Comparing polarity, subjectivity, and the relationship between them in one chart, is vital for understanding the results from the other data visualization techniques better and also confirm them.

Figure 38

Subjectivity and Polarity diagram



As this schema represents, in x-axis, in the range of [0] which refers to neutral polarity the subjectivity levels are really low, and they are centered upon range [0, 0.6]. The majority is between approximately [0, 0.3]. The fewer are between [0.6, 1] which is the range that indicates maximum subjectivity. This signifies that the biggest percentage on neutral sentiment contains a really low level of subjectivity as the majority is below average. In polarity range [0, 1] which refers to positive sentiment, a bigger scattering of subjectivity is being displayed. y-axis contains almost the same levels of subjectivity between [0, 0.5] and [0.5, 1] which means that users expressed in a more personal way than the users in neutral sentiment category. The schema also shows a dot in polarity range [1] and subjectivity range [1] which is the total positive sentiment with the ultimate emphasis. Most of positive polarity tweets are among [0, 0.5] in x-axis with a small amount being above average. This is in contrast with the negative polarity. In the range within [-1, 0] a more even distribution of polarity is displayed, compared to both other sentiments. In this category, subjectivity has the biggest percentage above average which means that negative sentiment was expressed in the most strongly way. Another

observation is that within [-1, 0.5] in x-axis and [0.6, 1] in y-axis, are included more dots in comparison with the correspondingly positive range and these dots represent expressed sentiment. This indicates that opinions and feelings regarding negativity were more strongly expressed and in bigger amounts than feelings regarding positivity.

5 Conclusion

In this thesis we addressed the problem of the semantic classification of tweets regarding fake news. Thus meaning the semantic classification of Twitter users. Twitter being the preference between a lot of social media for briefing, commenting and keeping up to date with the daily life, it gains a huge amount of opinions and sentiments in daily bases with reference to different topics. So, it's a treasure source for sentiment analysis referring to miscellaneous topics.

To perform the sentimental classification of Twitter users, the use of Twitter APIs, Python and its' libraries were required. Following the basic steps of a sentiment analysis, firstly tweets were extracted and saved. After the data cleaning process was performed. Following is the determination of subjectivity and polarity which is the most vital step to the semantic classification. Lastly, the sentiment of each tweet is being classified. To make the research livelier, the findings are being visualized.

Based on the results, neutral is the preponderant sentiment. The majority of users didn't express a negative sentiment as it was expected. In fact, the neutral sentiment score was more than 200 above the negative one. This is almost the half of negative score. An interesting observation is that neutral sentiment had the smallest score of subjectivity. Meaning that almost all neutral tweets were objective and didn't contain a serious amount of sentiment. Negative tweets though, were the most subjective (16 times more than neutral), something that clarifies that negative opinions regarding to fake news were the ones with the stronger expressed opinion. Users may not spend their time commenting about fake news and misinformation, but when they do, they have something bad to say with an a lot emphatic way.

As a result of my presented research, I became interested in exploring more in quality of sentiments and not in quantity. Anger and negativity are the ruling sentiments in a variety of areas today, and the ones that people are most passioned about. In particular, do people care about expressing a positive opinion or an opinion at all, or they seek the negative one?

5.1 Future Work

Due to limitations like lack of time and dealing with real time data (that are usually time consuming), some research areas have been left for future work. Future work cover dipper and sometimes better analysis in fields regarding the research, using

different techniques to achieve better results or using the result of the research as a strong foundation for several topic related future works. For instance, the hereunto research can be used as a step in a future work regarding to fake news detection in social media and especially Twitter.

Some other ideas that I would have liked to try and are suggested for future work are:

- Topic Modeling: A further analysis after sentiment classification can be done to clarify the specific topic in which users are expressing such feelings to. In this way a better understanding of human opinion and expression can be achieved.
- Emotion classification: Classifying the expressed emotions, a more detailed result is obtained. Negative sentiment for example can include sadness, anger, disappointment, denial, negativity etc. A deeper analysis regarding to each sentiment can give more detailed and with higher accuracy results that can be used in more distinguished researches and answer more to the point questions.
- Specific keyword sentiment analysis: Performing semantic classification on tweets including specific keywords (that were already used in the process) gives more integrated results and concludes the research in a more totalitarian way. For instance #misinformationvaccine and #misinformation can give results that are useful for comparison of feelings regarding related topics. More specific, how does the feeling changes when it comes to misinformation about vaccines and misinformation itself? Does it change at all? Or even comparing non same topic keywords like #hoax and #fakenewsvaccine. In which one do people have the most negative or neutral opinion? And which one had the most emphatic expressions leading to most subjective feeling?

References

5. Categorizing and Tagging Words. (2019, September 4). Book. https://www.nltk.org/book/ch05.html

A Python script to download all the tweets of a hashtag into a csv. (2017). Gist. https://gist.github.com/vickyqian/f70e9ab3910c7c290d9d715491cde44c

A. (2018a, November 20). *A gentle introduction to Sentiment Analysis*. Kaggle. https://www.kaggle.com/adarshchavakula/a-gentle-introduction-to-sentiment-analysis

A. (2019a, August 13). *Best Practice to Extract and Remove URLs from Python String – Python Tutorial*. Tutorial Example. https://www.tutorialexample.com/best-practice-to-extract-and-remove-urls-from-python-string-python-tutorial/

A. (2021a, June 24). *Extracting tweets from Twitter using API with Python*. AskPython. https://www.askpython.com/python/examples/extracting-tweets-using-twitter-api

A., V., & Sonawane, S. S. (2016). Sentiment Analysis of Twitter Data: A Survey of Techniques. *International Journal of Computer Applications*, *139*(11), 5–15. https://doi.org/10.5120/ijca2016908625

Ahmad, M., Aftab, S., Muhammad, S., & Ahmad, S. (2017). Machine Learning Techniques for Sentiment Analysis: A Review. *Int. J. Multidiscip. Sci. Eng*, *8*(3), 27–32.

Ahuja, R., Chug, A., Kohli, S., Gupta, S., & Ahuja, P. (2019). The impact of features extraction on the sentiment analysis. *Procedia Computer Science*, *152*, 341–348. https://doi.org/10.1016/j.procs.2019.05.008

Akhmad, A. (2020a, November 21). *Collecting Twitter Data using Python - Aron Akhmad*. Medium. https://aronakhmad.medium.com/collecting-twitter-data-using-python-f5f7867a7305

Akhmad, A. (2020b, November 21). *Twitter Data Cleaning using Python - Aron Akhmad*. Medium. https://aronakhmad.medium.com/twitter-data-cleaning-using-python-db1ec2f28f08

Alloghani, M., Baker, T., Hussain, A., Al-Khafajiy, M., Khalaf, M., & Mustafina, J. (2019). Sentiment Analysis for Decision-Making Using Machine Learning Algorithms. *Data Science, July*, 285–304. https://doi.org/10.1201/9780429263798-13

Alonso, M. A., Vilares, D., Gómez-Rodríguez, C., & Vilares, J. (2021). Sentiment analysis for fake news detection. *Electronics (Switzerland)*, *10*(11). https://doi.org/10.3390/electronics10111348

Alsaeedi, A. (2019, December 28). *A Study on Sentiment Analysis Techniques of Twitter Data*. SAI. https://thesai.org/Publications/ViewPaper?Volume=10&Issue=2&Code=IJACSA&SerialNo=48

Bagheri, H., & Islam, M. J. (2017). *Twitter Sentiment Analysis*. *8*(03), 1–2. https://doi.org/10.31219/osf.io/6xc4y

Bakshi, R. K. (2016). *[PDF] Opinion mining and sentiment analysis | Semantic Scholar*. SEMANTICSCHOLAR. https://www.semanticscholar.org/paper/Opinion-mining-and-sentiment-analysis-Bakshi-Kaur/87ee57c4915c5f7c5c7b9486df0c8af7e31bd747

Beck, M. (2021, May 28). *How to Scrape Tweets From Twitter - Towards Data Science*. Medium. https://towardsdatascience.com/how-to-scrape-tweets-from-twitter-59287e20f0f1

Bhandari, A. (2020, August 18). *Analyse Streaming Tweets Using Python & PostgreSQL |Analyse Live Data*. Analytics Vidhya. https://www.analyticsvidhya.com/blog/2020/08/analysing-streaming-tweets-with-python-and-postgresql/

Birjali, M., Beni-Hssane, A., & Erritali, M. (2017). Machine Learning and Semantic Sentiment Analysis based Algorithms for Suicide Sentiment Prediction in Social Networks. *Procedia Computer Science*, *113*(December), 65–72. https://doi.org/10.1016/j.procs.2017.08.290

Bonner, A. (2019, November 17). *The Ultimate Beginner's Guide to Data Scraping, Cleaning, and Visualization*. Medium. https://towardsdatascience.com/ultimate-beginners-guide-to-scraping-and-cleaning-twitter-data-a64e4aaa9343

Brahmananda Reddy, A., Vasundhara, D. N., & Subhash, P. (2019). Sentiment research on twitter data. *International Journal of Recent Technology and Engineering*, *8*(2 Special Issue 11), 1068–1070. https://doi.org/10.35940/ijrte.B1181.0982S1119

Brush, K., & Burns, E. (2020, February 20). *data visualization*. SearchBusinessAnalytics. https://searchbusinessanalytics.techtarget.com/definition/data-visualization

Burrows, M. (2020, August 14). *Objective vs Subjective*. SkillsTx | Professional Development Plan | SFIA | Skills Test. https://skillstx.com/objective-vs-subjective/

D'Souza, J. (2018, June 11). *Learning POS Tagging & Chunking in NLP - GreyAtom*. Medium. https://medium.com/greyatom/learning-pos-tagging-chunking-in-nlp-85f7f811a8cb#:%7E:text=How%20does%20POS%20Tagging%20works%3F&text=POS%20tagging%20is%20a%20supervised,set%20is%20Penn%20Treebank%20tagset

Daityari, S. (2021, July 29). *How To Perform Sentiment Analysis in Python 3 Using the Natural Language Toolkit (NLTK)*. DigitalOcean. https://www.digitalocean.com/community/tutorials/how-to-perform-sentiment-analysis-in-python-3-using-the-natural-language-toolkit-nltk

Duca, A. L. (2021, March 23). *How to build a dataset from Twitter using Python tweepy*. Medium. https://towardsdatascience.com/how-to-build-a-dataset-from-twitter-using-python-tweepy-861bdbc16fa5

Dudani, V. A. P. B. K. (2018, February 7). *Twitter Data Analysis using Python*. Begin Analytics. https://beginanalyticsblog.wordpress.com/2018/02/07/twitter-data-analysis-using-python/

Earth Data Analytics Online Certificate. (2020, September 11). Earth Data Science - Earth Lab. https://www.earthdatascience.org/courses/use-data-open-source-python/intro-to-apis/analyze-tweet-sentiment-in-python/

Es, S. (2021, July 20). *Sentiment Analysis in Python: TextBlob vs Vader Sentiment vs Flair vs Building It From Scratch*. Neptune.Ai. https://neptune.ai/blog/sentiment-analysis-python-textblob-vs-vader-vs-flair

extract tweets with some special keywords from twitter using tweepy in python. (2018, March 4). Stack Overflow. https://stackoverflow.com/questions/49098726/extract-tweets-with-some-special-keywords-from-twitter-using-tweepy-in-python

GeeksforGeeks. (2019, October 28). *NLP | Part of Speech - Default Tagging*. https://www.geeksforgeeks.org/nlp-part-of-speech-default-tagging/

GeeksforGeeks. (2020, November 1). *Extracting Tweets containing a particular Hashtag using Python*. https://www.geeksforgeeks.org/extracting-tweets-containing-a-particular-hashtag-using-python/

GeeksforGeeks. (2021a, May 31). *Removing stop words with NLTK in Python*. https://www.geeksforgeeks.org/removing-stop-words-nltk-python/

GeeksforGeeks. (2021b, July 6). *Generating Word Cloud in Python*. https://www.geeksforgeeks.org/generating-word-cloud-python/

GeeksforGeeks. (2021c, July 22). *Twitter Sentiment Analysis using Python*. https://www.geeksforgeeks.org/twitter-sentiment-analysis-using-python/

H. (2021b, June 23). *Rule-Based Sentiment Analysis in Python for Data Scientists*. Analytics Vidhya. https://www.analyticsvidhya.com/blog/2021/06/rule-based-sentiment-analysis-in-python/

Hebbar, N. (2021, January 8). *Twitter Sentiment Analysis Using Python for Complete Beginners*. Medium. https://medium.com/swlh/tweet-sentiment-analysis-using-python-for-complete-beginners-4aeb4456040

Horan, C. (2020, February 10). *Tokenizers: How machines read*. FloydHub Blog. https://blog.floydhub.com/tokenization-nlp/

How do I find and remove duplicate rows in pandas? (2016, July 26). [Video]. YouTube. https://www.youtube.com/watch?v=ht5buXUMqkQ&ab_channel=DataSchool

How mood affects the stock market: Empirical evidence from microblogs. (2020, July 1). ScienceDirect. https://www.sciencedirect.com/science/article/abs/pii/S0378720618307183

Jarzynski, P. (2020, October 13). *Twitter Sentiment Analysis in Python - Towards Data Science*. Medium. https://towardsdatascience.com/twitter-sentiment-analysis-in-python-1bafebe0b566

Johnson, D. (2021, August 19). *Python Pandas Tutorial: DataFrame, Date Range, Use of Pandas*. Guru99. https://www.guru99.com/python-pandas-tutorial.html

Justin, L. (2020, November 18). *How to apply useful Twitter Sentiment Analysis with Python Step-by-Step Example*. Just into Data. https://www.justintodata.com/twitter-sentiment-analysis-python/#step-1-set-up-twitter-authentication-and-python-environments

Kalebu, J. (2020, November 16). *A quick guide to Twitter sentiment analysis in Python*. DEV Community. https://dev.to/kalebu/a-quick-guide-to-twitter-sentiment-analysis-in-python-55dh

Khattak, A. M., Batool, R., Satti, F. A., Hussain, J., Khan, W. A., Khan, A. M., & Hayat, B. (2020). Tweets classification and sentiment analysis for personalized tweets recommendation. *Complexity*, *2020*. https://doi.org/10.1155/2020/8892552

Kim, R. (2021, June 23). *Another Twitter sentiment analysis with Python — Part 1*. Medium. https://towardsdatascience.com/another-twitter-sentiment-analysis-bb5b01ebad90

Koh, J. X., & Liew, T. M. (2020). How loneliness is talked about in social media during COVID-19 pandemic: Text mining of 4,492 Twitter feeds. *Journal of Psychiatric Research*, *September*. https://doi.org/10.1016/j.jpsychires.2020.11.015

Kosaka, M. (2020, December 22). *Cleaning & Preprocessing Text Data for Sentiment Analysis*. Medium. https://towardsdatascience.com/cleaning-preprocessing-text-data-for-sentiment-analysis-382a41f150d6

Kumar, A. (2021). *Sentiment Analysis as a Restricted NLP Problem*. IGIGlobal. https://www.igi-global.com/chapter/sentiment-analysis-as-a-restricted-nlp-problem/259784

Leetaru, K. (2020). Sentiment Analysis. *Data Mining Methods for the Content Analyst*, 79–84. https://doi.org/10.4324/9780203149386-11

Leow, G. (2020, October 22). *Scraping Tweets with Tweepy Python - Python in Plain English*. Medium. https://python.plainenglish.io/scraping-tweets-with-tweepy-python-59413046e788

Li, S. (2019, January 10). *Introducing TextBlob - Towards Data Science*. Medium. https://towardsdatascience.com/having-fun-with-textblob-7e9eed783d3f

Li, S., Lee, S. Y. M., Chen, Y., Huang, C. R., & Zhou, G. (2010). Sentiment classification and polarity shifting. *Coling 2010 - 23rd International Conference on Computational Linguistics, Proceedings of the Conference*, *2*(January), 635–643.

M. (2018b, September 21). *twitter sentiment analysis basic*. Kaggle. https://www.kaggle.com/mistryjimit26/twitter-sentiment-analysis-basic

Malik, U. (2019a, August 28). *Python for NLP: Sentiment Analysis with Scikit-Learn*. Stack Abuse. https://stackabuse.com/python-for-nlp-sentiment-analysis-with-scikit-learn/

Manguri, K. H. (2020). *[PDF] Twitter Sentiment Analysis on Worldwide COVID-19 Outbreaks | Semantic Scholar*. SEMANTICSCHOLAR.

https://www.semanticscholar.org/paper/Twitter-Sentiment-Analysis-on-Worldwide-COVID-19-Manguri-Ramadhan/9421c96a79438d4498ca05cac799f7142b54c18f

Manoj, P. (2020, December 6). *Sentiment Analysis Using Python and NLTK - The Startup*. Medium. https://medium.com/swlh/sentiment-analysis-using-python-and-nltk-library-d68caba27e1d

Marco, V. A. P. B. (2017, July 19). *Mining Twitter Data with Python (Part 1: Collecting data)*. Marco Bonzanini. https://marcobonzanini.com/2015/03/02/mining-twitter-data-with-python-part-1/

Mitra, A. (2020). Sentiment Analysis Using Machine Learning Approaches (Lexicon based on movie review dataset). *Journal of Ubiquitous Computing and Communication Technologies*, *2*(3), 145–152. https://doi.org/10.36548/jucct.2020.3.004

Mondal, S. (2020, November 22). *Twitter Data Cleaning and Preprocessing for Data Science*. Medium. https://medium.com/swlh/twitter-data-cleaning-and-preprocessing-for-data-science-3ca0ea80e5cd

Ng, A. (2019, April 18). *Text mining and Sentiment Analysis using Python - Ashley Ng*. Medium. https://medium.com/@xyng17/scraping-twitter-and-sentiment-analysis-using-python-c5a44b9288ab

NLP for Beginners - Sentiment Analysis of Twitter Data Using Scikit-Learn in Python. (2020, August 18). [Video]. YouTube. https://www.youtube.com/watch?v=qzBtplRo91o&ab_channel=KGPTalkie

P. (2019b, January 2). *Twitter Sentiment Analysis*. Kaggle. https://www.kaggle.com/paoloripamonti/twitter-sentiment-analysis/comments

Pai, A. (2021, July 23). *What is Tokenization | Tokenization In NLP*. Analytics Vidhya. https://www.analyticsvidhya.com/blog/2020/05/what-is-tokenization-nlp/

Pavan Kumar, M. R., & Prabhu, J. (2018). Role of sentiment classification in sentiment analysis: A survey. *Annals of Library and Information Studies*, *65*(3), 196–209.

Potey, G., Jadhav, R., Shroff, K., Gore, A., Phalke, D., & Shimpi, J. (2020). Fake News Detection and Sentiment Analysis in Twitter. *International Journal for Research in Applied Science & Engineering Technology (IJRASET)*, *8*(IV), 72–75.

Prabhakaran, S. (2020, May 18). *Lemmatization Approaches with Examples in Python*. Machine Learning Plus. https://www.machinelearningplus.com/nlp/lemmatization-examples-python/

Python Tutorial. (2020, October 25). *How to Write to CSV Files in Python*. Python Tutorial - Master Python Programming For Beginners from Scratch. https://www.pythontutorial.net/python-basics/python-write-csv-file/

Qian, V. (2020, September 15). *Step-By-Step Twitter Sentiment Analysis: Visualizing Multiple Airlines' PR Crises [Updated for 2020]*. IPullRank. https://ipullrank.com/step-step-twitter-sentiment-analysis-visualizing-united-airlines-pr-crisis

R. (2018c, October 11). *Text Data Cleaning - tweets analysis*. Kaggle. https://www.kaggle.com/ragnisah/text-data-cleaning-tweets-analysis

R. (2021c, July 6). *Word Cloud in Python | How to Build Word Cloud in Python?* Analytics Vidhya. https://www.analyticsvidhya.com/blog/2021/05/how-to-build-word-cloud-in-python/

R. (2021d, July 6). *Word Cloud in Python | How to Build Word Cloud in Python?* Analytics Vidhya. https://www.analyticsvidhya.com/blog/2021/05/how-to-build-word-cloud-in-python/

Rachuta, K. (2018, April 9). *Importing and exporting CSV files in Python - Kasia Rachuta*. Medium. https://medium.com/@kasiarachuta/importing-and-exporting-csv-files-in-python-7fa6e4d9f408

Ramachandran, A. (2018, October 5). *NLP Guide: Identifying Part of Speech Tags using Conditional Random Fields*. Medium. https://medium.com/analytics-vidhya/pos-tagging-using-conditional-random-fields-92077e5eaa31

Rao, R., Polepaka, S., & Rafeeq, M. (2015). Design of Sentiment Analysis System using Polarity Classification Technique. *International Journal of Computer Applications*, *125*(15), 22–24. https://doi.org/10.5120/ijca2015906159

Rasool, A., Tao, R., Marjan, K., & Naveed, T. (2019). Twitter Sentiment Analysis: A Case Study for Apparel Brands. *Journal of Physics: Conference Series*, *1176*(2). https://doi.org/10.1088/1742-6596/1176/2/022015

Removing Punctuation | Pre-processing | Natural Language Processing with Python and NLTK. (2019, July 20). [Video]. YouTube. https://www.youtube.com/watch?v=9CpID8ZL1IQ&ab_channel=KnowledgeCenter

Removing stop words | Natural Language Processing with Python and NLTK. (2019, July 20). [Video]. YouTube. https://www.youtube.com/watch?v=ob6IlbV13IM&ab_channel=KnowledgeCenter

S. (2021e, June 15). *12 Twitter Sentiment Analysis Algorithms Compared*. AI Perspectives. https://www.aiperspectives.com/twitter-sentiment-analysis/

Selvaraj, N. (2020, September 12). *A Beginner's Guide to Sentiment Analysis with Python*. Medium. https://towardsdatascience.com/a-beginners-guide-to-sentiment-analysis-in-python-95e354ea84f6

Sentiment Analysis 101. (n.d.). KDnuggets. Retrieved August 24, 2021, from https://www.kdnuggets.com/2015/12/sentiment-analysis-101.html#:%7E:text=Sentiments%20refer%20to%20attitudes%2C%20opinions,as%20opposed%20to%20objective%20facts.&text=There%20are%20two%20main%20types,%2Faspect%2Dbased%20sentiment%20analysis

Sentiment Analysis Python | WordCloud, Vader Sentiment, TextBlob. (2020, November 17). YouTube. https://www.youtube.com/watch?v=AnvrJNLKp0k&ab_channel=PriyankaSharma

Sentiment Analysis: Measuring Opinions. (2015, January 1). ScienceDirect. https://www.sciencedirect.com/science/article/pii/S1877050915003956

Shah, P. (2020, November 6). *Sentiment Analysis using TextBlob - Towards Data Science*. Medium. https://towardsdatascience.com/my-absolute-go-to-for-sentiment-analysis-textblob-3ac3a11d524

Sharma, A., & Ghose, U. (2020). Sentimental Analysis of Twitter Data with respect to General Elections in India. *Procedia Computer Science*, *173*(2019), 325–334. https://doi.org/10.1016/j.procs.2020.06.038

Shrestha, M. (2018). *Detecting Fake News with Sentiment Analysis and Network Metadata*. https://portfolios.cs.earlham.edu/wp-content/uploads/2018/12/Fake_News_Capstone.pdf

Sims, S. S. (2021). *Sentiment Analysis 101*. KDnuggets. https://www.kdnuggets.com/2015/12/sentiment-analysis-101.html

Singh, S. (2020, December 23). *How To Remove StopwordsIn Python | Stemming and Lemmatization*. Analytics Vidhya.

https://www.analyticsvidhya.com/blog/2019/08/how-to-remove-stopwords-text-normalization-nltk-spacy-gensim-python/

Singhal, G. (2020, July 1). *Building a Twitter Sentiment Analysis in Python*. Pluralsight. https://www.pluralsight.com/guides/building-a-twitter-sentiment-analysis-in-python

Sistilli, A. (2017, June 7). *Twitter Data Mining: A Guide to Big Data Analytics Using Python*. Toptal Engineering Blog. https://www.toptal.com/python/twitter-data-mining-using-python

Team, T. (2021, June 14). *Sentiment Analysis using Python [with source code]*. TechVidvan. https://techvidvan.com/tutorials/python-sentiment-analysis/

Team, T. A. I. (2021, April 2). *Sentiment Analysis (Opinion Mining) with Python - NLP Tutorial | Towards AI*. Medium. https://pub.towardsai.net/sentiment-analysis-opinion-mining-with-python-nlp-tutorial-d1f173ca4e3c

TextBlob Sentiment: Calculating Polarity and Subjectivity. (n.d.). Planspace. Retrieved August 24, 2021, from https://planspace.org/20150607-textblob_sentiment/

TextBlob: Simplified Text Processing — TextBlob 0.16.0 documentation. (n.d.). TextBlob. Retrieved August 24, 2021, from https://textblob.readthedocs.io/en/dev/

Tokenization | Natural Language Processing with Python and NLTK. (2019, July 20). [Video]. YouTube. https://www.youtube.com/watch?v=Dh4El5MtxpE&ab_channel=KnowledgeCenter

Twitter Sentiment Analysis in Real-Time. (2019, June 7). MonkeyLearn Blog. https://monkeylearn.com/blog/sentiment-analysis-of-twitter/

Twitter Sentiment Analysis Using Python. (2020, February 4). [Video]. YouTube. https://www.youtube.com/watch?v=ujId4ipkBio&ab_channel=ComputerScience

Vadapalli, P. (2020, August 10). *How to Build a Twitter Sentiment Analysis Python Program? [Step-by-Step Tutorial]*. UpGrad Blog. https://www.upgrad.com/blog/build-twitter-sentiment-analysis-python/

Vidhya, A. (2020, July 12). *Infographic : Cleaning Text Data Python*. Analytics Vidhya. https://www.analyticsvidhya.com/blog/2015/06/quick-guide-text-data-cleaning-python/

What Is Data Visualization? Definition, Examples, And Learning Resources. (n.d.). Tableau. Retrieved August 24, 2021, from https://www.tableau.com/learn/articles/data-visualization

Wikipedia contributors. (2021, June 18). *Lexical analysis*. Wikipedia. https://en.wikipedia.org/wiki/Lexical_analysis

Wu, S. (2021, June 5). *Design your own Sentiment Score - Towards Data Science*. Medium. https://towardsdatascience.com/design-your-own-sentiment-score-e524308cf787

Yener, Y. (2020, November 7). *Step by Step: Twitter Sentiment Analysis in Python - Towards Data Science*. Medium. https://towardsdatascience.com/step-by-step-twitter-sentiment-analysis-in-python-d6f650ade58d