University of Essex

School of Computer Science & Electronic Engineering

Final Project Report
AI In Marketing

Submitted as a part of:

CE903: Group Project

Group 11 Members

Jasmine Reena Winsy Edwin Rajarethinam – 2201019
Onyemelonu Nneoma Charity –2207938
Yalamanchili Abhishek – 2201851
Sidra Munaf – 2201168
Charles Bond – 2108115
Marri Priyanka – 2205231
Rashida Arakka – 2201071

Supervisor:
Dr. Vito De Feo

Table of Contents

1 Introduction

1.1 Project Objectives

Businesses may have strong internal insight into their sales, marketing, product metrics, brand positioning, and strategies, but they still don't have clear visibility into external elements, such as what their competitors are doing, to assist them allocate resources based on facts. In the modern world, keeping track of brand and company trends and sentiment is crucial if you want to use the data to make wise decisions.

As a brand strategist, you might be developing a brand strategy plan for a new company called Zero Strength business and need to ascertain which flavours sell the best, how people prefer to consume their energy drinks, which influencers best promote energy drinks, who the target audience is, which keywords are frequently used to engage the target audience, and how to be effectively positioned in the market. You must carefully arrange the new energy drink in order to collect the statistics. You may try to make predictions using your own observations, market research, or user surveys. These methods are sometimes erroneous since the data acquired may be incomplete, requiring a lot of time, resources, and financing, which are often scarce for start-ups.

The goal of this AI marketing project is to gather tweets about a specific good or service using data scraping techniques and draw valuable conclusions from them. The project's goal is to find patterns and trends that can assist marketers in making decisions about their marketing strategy by examining the sentiment, keywords, and themes covered in these tweets.

In order to effectively categorize tweets according to their relevance and sentiment while also extracting pertinent keywords and subjects, the project will include creating and training an AI model. Marketers will be able to refine their messaging and targeting to better engage with their audience on Twitter thanks to the data produced by this initiative.

1.2 Methodology

The AI marketing tool being developed, implements a Competitive Intelligence system that converts data into intelligence to provide a real-time view of the competitive landscape. The develop tool listens to traffic in an industry by collecting data and carries out thorough analysis on the data which guides decision making. In this project, we built a set of tools to Identifying the research question or issue that has to be solved is the first stage in every research endeavour. "How can AI marketing be used to derive insights from Twitter data for improved marketing strategies?" is the research question in this instance.

Data Collection: We will utilize a web scraping tool to extract tweets about the selected good or service in order to gather the data needed for this project. Real-time collection and database storage of the tweets will allow for additional analysis.

Data pre-processing: Data must be pre-processed to weed out any irrelevant information and make it ready for analysis before analysis. This entails eliminating stop words, addressing missing data, and standardizing the content.

Sentiment Analysis: Natural language processing techniques will be used to examine each tweet's sentiment to determine if it is favourable, negative, or neutral. We will learn more about how consumers feel about the good or service from this.

The next step is to extract pertinent keywords and topics from the tweets using topic modeling approaches. This will assist us in comprehending the key themes and conversations surrounding the good or service on Twitter.

Classification: Based on their relevance to the study issue, we will classify the tweets using machine learning methods. This will enable us to focus on the tweets that are most pertinent to our analysis and filter out any that aren't. Visualization: In order to better grasp the insights we have drawn from the data, we will finally use charts and graphs to illustrate the outcomes of our study. By doing this, we will be able to spot patterns and trends that will enable us to decide on our marketing tactics with knowledge.

In conclusion, this AI marketing project's technique entails gathering data from Twitter, preprocessing the data, assessing sentiment, extracting pertinent keywords and subjects, categorizing tweets, and visualizing the outcomes. This strategy will offer useful perceptions into consumer opinions and habits that can be used to boost consumer engagement and guide marketing tactics.

1.3 Tools and libraries

The following are some of the tools and libraries used while working on this project. This is by no means an exhaustive list.

● Tweepy - Unofficial Twitter API interaction using Tweepy, a library

● Pandas - it makes it possible for marketers to quickly preprocess huge datasets and derive insights from them, Pandas is commonly utilized in AI marketing.

● Seaborn, which is utilized to create stunning visualizations of the analysis's

findings.

● Numpy - All things considered, NumPy is an effective tool for numerical computation and data analysis in AI marketing, assisting marketers in quickly processing and analyzing enormous amounts of data.

● TextBlob: it enables marketers to evaluate and interpret the sentiment and tone of consumer comments, reviews, and social media postings, TextBlob is commonly utilized in AI marketing. Marketers can learn more about client preferences and attitudes, spot problem areas, and create more successful marketing strategies by performing sentiment analysis on text data.

● Matplot:Overall, Matplotlib is a useful tool for data visualization in AI marketing that aids in the successful communication of discoveries and insights as well as the production of data-driven decisions.

● Nltk.sentiment.vader- With the help of the effective sentiment analysis tool nltk.sentiment.vader, marketers may better understand client sentiment and enhance their marketing tactics.

● NRCLex - A strong and user-friendly tool for sentiment analysis in AI marketing, NRCLex is especially useful for social media data. It can assist marketers in quickly deriving insightful information from client feedback and refining their marketing plans in light of this information.

● NLTK – the most widely used NLP library which helped in cleaning the raw text data retrieved from various social media platforms.

● Gitlab - utilized for collaborative source code management.

● JIRA was utilized to manage this project's tasks in the sprints according to the agile methodology.

2 System Design

2.1 System design flow

Based on the system Architecture of this product, where we had the functionality sub-divided into five (5) Modules, the first being the client/ User side which involves an input of the competitors to be analyses, the remaining 4 modules included Data fetching/extraction, Data pre-processing, Visualization and Analysis expanded below

2.1.1 Data fetching/extraction

The fetching/extraction module is an essential part of the AI marketing system because it is responsible for collecting the necessary data from Twitter that is required for the analysis. Accessing several forms of Twitter data, such as tweets, retweets, followers, and trending topics with the help of the Tweepy library. Additionally, it offers ways to filter and arrange the data according to factors like language, location, and date. The collected information is saved in a csv file format or a pandas data frame and feed to the pre-processing engine.

2.1.2 Data pre-processing

The pre-processing module include converting the tweets to lowercase, removing mentions and hashtags, removing links, removing punctuations, filtering non-alphanumeric characters, removing stop words, and sorting the data date-wise. In this process, data is cleaned, transformed, and organized to make it more suitable for analysis.

2.1.3 Visualisation

For the purpose of analysis and interpretation, this module generates visual representations of the data using tools like heatmaps and pie charts. Various textual visualization techniques are applied to gain relevant information from the clean out data.

2.1.4 Analysis and Result

This module applies several techniques such as content analysis, similarity analysis, and sentiment analysis to extract insights from the data. In sentimental analysis involves categorizing tweets into five degrees of sentiment, including pure and mild neutral tweets and then using the NRC Emotion to extract emotional content from those tweets. Results are shown in the GUI from graphs and pie chart as well as breakdown of positive or negative connotations associated with the keyword entered.

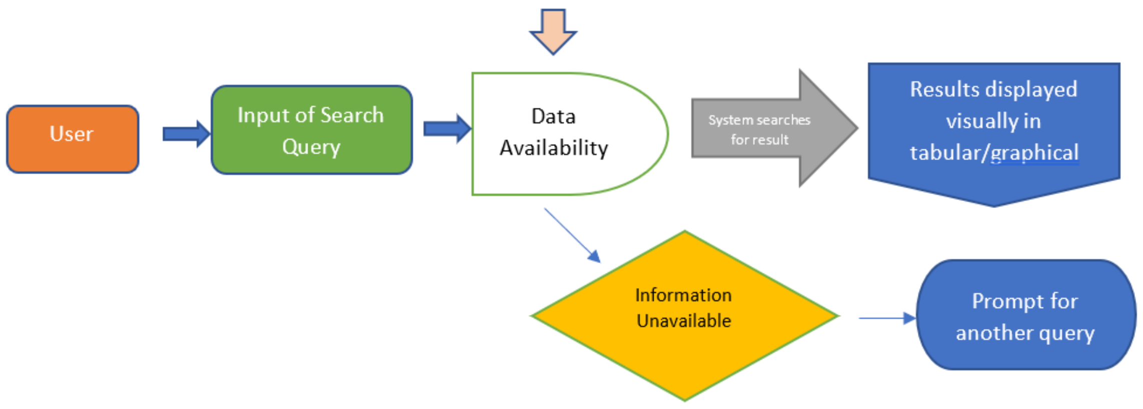
In Content Analysis the process begins with the extraction of data from Twitter using scrapping techniques. Next, a content analysis code is applied, which is a research approach utilized to study text data for identifying patterns, themes, and meanings. The code works on a Data Frame that consists of tweets and extracts the columns "created at" and "Tweets". It then calculates the frequency of each word in the tweets and displays the top 20 most frequent words along with their percentage frequency. Additionally, the code generates a Word Cloud and a bar chart to visualize the most common words. Furthermore, a new column is added to the Data Frame to identify words with a frequency of over 50%, and those words are printed. In summary, the code showcases how content analysis can be employed to analyze and visualize text data.

In Similarity Analysis following the extraction of tweets using a data extraction method, the tweets are saved in a csv file. Next, a similarity analysis code is applied to the csv file. The code operates on a CSV file that contains tweets and reads them in chunks, then appends them to an empty Data Frame. The user is prompted to input two keywords, for which similarity analysis is conducted. The code extracts tweets that include these keywords and transforms them into TF-IDF vectors. Subsequently, the cosine similarity is calculated between the two sets of vectors, and a heatmap is generated using seaborn to visualize the results. The code also identifies the index of the most similar tweets. The output presents an understanding of how comparable the tweets that contain the two keywords are to each other.

2.2 System architecture

The layout of the system originally described in the SRS document is as shown in below. As earlier mentioned, the major subsections are User/ Client entry, Data Fetching, Data Pre-processing, Model building and Analysis/ Result.

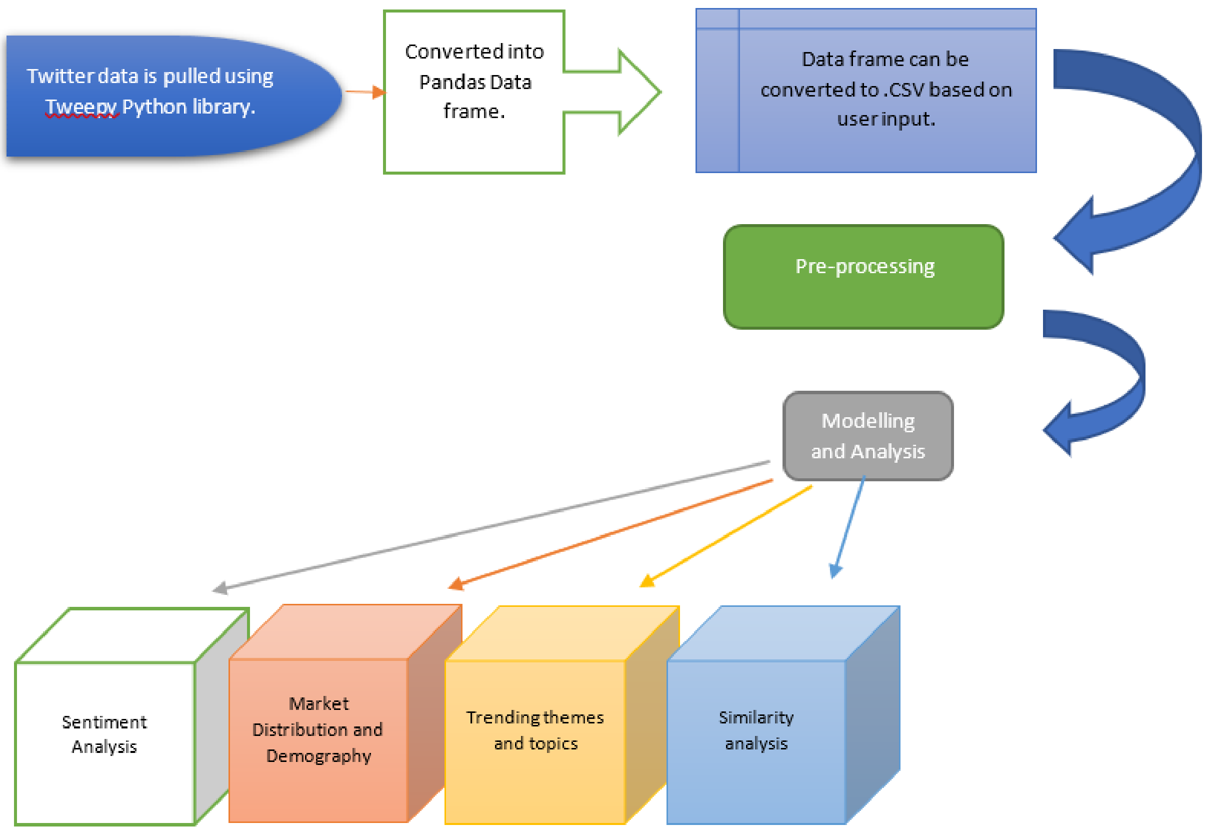
Figure 1. As above, this details the outline of the system architecture.



2.2.1 System User Interface Diagram

Figure 2.

Details the process flow of the expected functionality of the user interface. This is the front-end user facing side which will be implemented using a simple GUI, that will return desired results.


Twitter data is pulled using Tweepy Python library.
Converted into Pandas Data frame.
Data frame can be converted to .CSV based on user input.
Pre-processing
Modelling and Analysis
Sentiment Analysis
Market Distribution and Demography
Trending themes and topics
Similarity analysis

2.3 Use Cases

Table 1

Use Case	UC 1.1
Use Case Name	Input details for data extraction from Twitter.
Actors	User
Description	User enters query such as hashtags.
Trigger	GUI will ask user to enter input in the appropriate field as specified.
Precondition	Correct accesses to Twitter using access keys are required.
Postcondition	Network Internet Connection access required.
Normal Flow	Twitter_Smart_Scraping.ipynb extracts data based on user input.
Alternative Flow	N/A
Exceptions	N/A
Includes	N/A
Assumptions	N/A
Notes and Issues	If information entered is not valid

Table 2

Use Case	UC 1.2
Use Case Name	Input Guidelines
Actors	User
Description	Instructions/Suggestions are given to input appropriate query.
Trigger	In GUI, user enters details as requested, i.e. hashtags or keywords that information can be found out about.
Precondition	Information is entered correctly.
Postcondition	Network Internet Connection access required.
Normal Flow	Twitter_Smart_Scraping.ipynb runs to extract data based on user input.
Alternative Flow	N/A
Exceptions	N/A
Includes	N/A
Assumptions	Assumes Internet Connection is made available.
Notes and Issues	If the information is unavailable, no tweets may be returned.

2.2 Product Function.

Table 3. Product Functionality FR 1.1.

Identifier	FR 1.1
Title	Fetching required data
Requirements	Prompt is given to user for required search parameters
Source	Web
Rationale	Data Collection to perform further analysis to gain insights.
Restriction and Risk	If internet connection is unavailable, task will not be able to be run.
Dependencies	Network Internet Connection access required.
Priority	High

Table 4. Product Functionality FR 1.2.

Identifier	**FR 1.2**
Title	Pre-Processing Data
Requirements	Read from CSV to begin pre-processing stage.
Source	Collected data in CSV format.
Rationale	Certain modelling functions require data to be in a specific format, i.e. all lowercase, removal of ascii characters.
Restriction and Risk	N/A
Dependencies	FR 1.1
Priority	High

Table 5 Product Functionality FR 1.3.

Identifier	**FR 1.3**
Title	Sentiment Analysis
Requirements	Key words taken from tweets which are divided into 5 subcategories to explain an users particular feeling or sentiment towards a particular topic. This can also include neutral tweets to find out twitter users lack of sentiment.
Source	Fetched data.
Rationale	Return actionable data that has value.
Restriction and Risk	N/A
Dependencies	FR 1.1 and FR 1.2
Priority	High

Table 6. Product Functionality FR 1.4

Identifier	**FR 1.4**
Title	Speech Emotion Recognition – Emotion extraction
Requirements	FR 1.3, Access to CSV containing tweets.
Source	Retrieved data.
Rationale	Providing further analysis to give meaningful insights into tweets.
Restriction and Risk	N/A
Dependencies	FR 1.1, FR 1.2, FR 1.3
Priority	High

Table 7 Product Functionality FR 1.5

Identifier	**FR 1.5**
Title	Competitor Overview
Requirements	Results will display information of competitors, engagement, and other twitter data from a summary of activity.
Source	User

Rationale	Required to know both qualitative and quantitative engagement between brands and market positioning.
Restriction and Risk	N/A
Dependencies	FR 1.1, FR 1.2, FR 1.3, FR 1.4
Priority	High

Table 8 Product Functionality FR 1.6

Identifier	**FR 1.8**
Title	Visualization and Analysis
Requirements	Application should be able to display analysis performed on twitter data to provide visual representation using graphs, pie charts, Market Demographics, Brand description and distribution, and frequency of occurring terms.
Source	User input.
Rationale	Visualization gives greater leverage to make informed business decisions.
Restriction and Risk	N/A
Dependencies	FR 1.1, FR 1.2, FR 1.3, FR 1.4, FR 1.5
Priority	High

3. Implementation

A significant amount of man hours was needed to implement the plan. The phase of development was divided into several sections, and the team developed in concert. To store programme code and other files, including flowcharts and data, the university Gitlab was used. The setting for cooperation was Jira. On a tiny set of data, preliminary testing and debugging were conducted. As it was discovered that this was effective, the software was scaled up to handle more data, and any bottlenecks were dealt with as the development went forward. The team was divided into smaller groups, and each group received a specific project task. The team will meet twice a week to discuss difficulties encountered and look for solutions from other team members or from the project manager. Initial data was provided by the client Dr. Erik, who defined an industry and listed its competitors and key influencers. He was also useful in pointing out the software acceptance criteria to the team.

Module implementation details

The design was carried out in accordance with the flow depicted in figure 2 below. The backend has the phases of development in place. No website or app has been created.

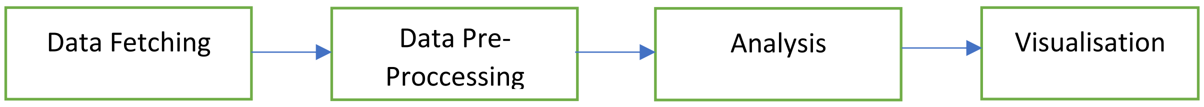


Figure 2: Developing Stage

3.1 Data Fetching Module

A software library or module known as a "data fetching module" offers the ability to retrieve data from a certain data source or API. These modules are frequently used in applications that require data to be gathered from many sources and processed for additional analysis, such as data scraping or web scraping. Data fetching modules can be more general-purpose or tailored to a single data source or API, such as a module for retrieving data from Twitter's API, or they can be both. For example, a module for web scraping can retrieve data from web pages. As shown in the figure above, during this initial stage of implementation, we have written functions into a file named get **final.py** to extract the necessary data from various platforms like websites and social media network like Twitter. These functions combine several existing libraries, which are explained in more detail below. A csv file from each of the corresponding platforms is the output. The module listed below is utilised.

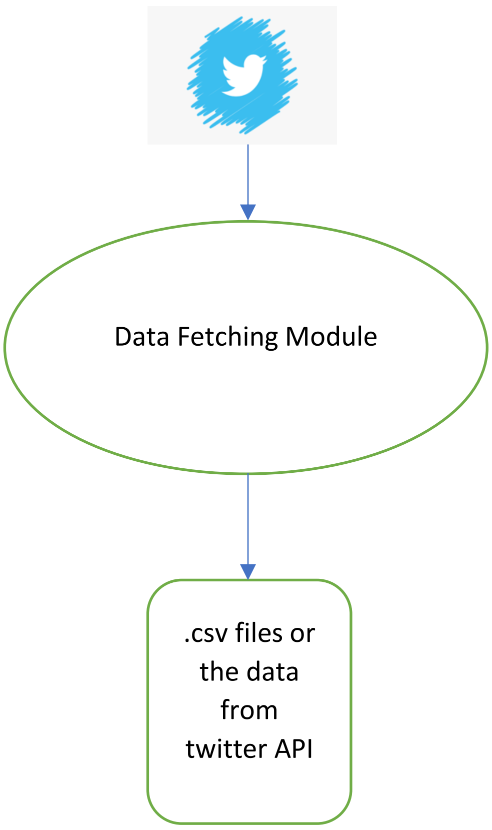


Figure 3: Data Fetching Module

Tweepy

Also, we have gathered information from tweets posted by brands to their tweeter sites and by social influencers. We utilised the open-source Python package Tweepy to develop an extractor with quick access to the Twitter API. With the help of this module, we have created a function to pull plain text information from brand and social influencer twitter profiles. To complete this process, the usernames of these brands and influencers are needed.

3.2 Pre-processing module

A software library or module known as a pre-processing module offers capabilities to alter or clean raw data before it can be used for additional analysis or processing. Pre-processing is a crucial stage in many data analysis workflows because it ensures that the data are in a format that is acceptable for analysis and that any noise or errors are eliminated or fixed. A data integrator and a data cleaner named data **final.py** were made for the pre-processing module ([2]) and published to the Gitlab repository.

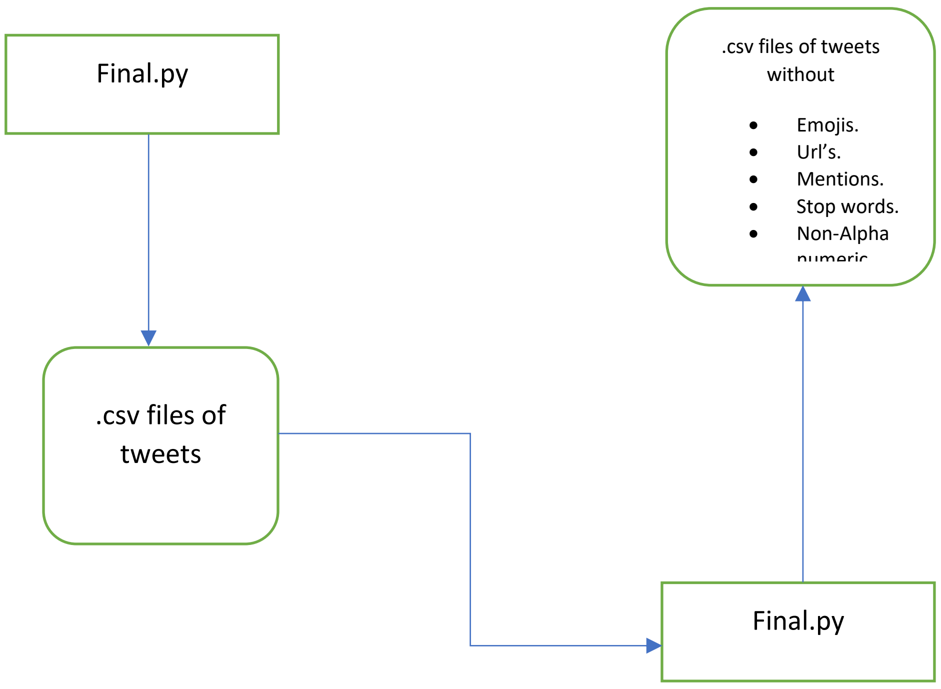


Figure 4: Data Cleaning Module

The following are some typical responsibilities of pre-processing modules:

Data Cleaning: Cleaning up the data involves removing or fixing errors, missing values, and other irregularities.

Data transformation: Changing the data's structure or format to make it better suited for analysis.

Data normalisation: It's the process of scaling the data to a standard range or distribution to make it easier to compare various data points.

Text processing: Preparing text data for analysis by tokenizing, stemming, deleting stop words, or using other methods.

Feature Extraction: Finding and extracting pertinent characteristics from the data for use in machine learning or other analysis models is known as feature extraction.

Pre-processing modules that we have used include:

NumPy: A Python package for scientific computing called NumPy has routines for manipulating arrays, using linear algebra, and performing other mathematical processes.

Pandas: Data cleaning, transformation, and normalisation tools are included in the Python data manipulation and analysis toolkit known as Pandas.

Scikit-learn: It's a Python toolkit for machine learning that has routines for pre-processing tasks including feature extraction and data normalisation.

NLTK: A Python natural language processing package including text processing features like tokenization, stemming, and stop word elimination.

3.3 Analysis Module

For this module we have created three Files Content_analysis.ipynb, Similarity_analysis.ipynb and Sentiment_analysis.ipynb. Let's have a look into them.

3.3.1 Content Analysis

After extracting the data from twitter using scrapping methods we apply content analysis code, which is a research method used to analyse text data to identify patterns, themes, and meanings. The code takes a Data Frame containing tweets and extracts the columns "created at" and "Tweets". It then counts the frequency of each word in the tweets and prints the top 20 most frequent words along with their percentage frequency. The code also creates a Word Cloud and a bar chart to visualize the most frequent words. Additionally, it adds a column to the Data Frame to identify words with a frequency of more than 50% and prints those words. Overall, the code demonstrates how content analysis can be used to analyse and visualize text data.

Libraries

The libraries used in the code are as follows:
Pandas: used for data manipulation and analysis.
Matplotlib: used for data visualization, including creating bar charts.
WordCloud: used for creating word clouds, which are a visual representation of the most frequent words in a dataset.

The code defines a function called **content_analysis** that takes a Pandas DataFrame as input, which is assumed to contain tweet data. The function first extracts the **created_at** and **Tweets** columns from the input DataFrame, then counts the frequency of each word in the tweets using the **str.split** method and the **value_counts** method of Pandas Series.

The top 20 most frequent words are printed, along with their percentage frequency, using a for loop. A word cloud is then created using the WordCloud library, and a bar chart of the top 20 most frequent words is also plotted using Matplotlib.

A new column called **High_Frequency** is added to the DataFrame, which indicates whether each tweet contains at least one word with a frequency of more than 0.5 (i.e., 50% of the tweets contain that word). The words with a frequency of more than 50% are printed using the **loc** method of Pandas DataFrame.

After this is performed, we run another piece of code that is used to calculate the engagement of tweets that are extracted from the csv file in the data extraction method this code calculates and visualizes the daily engagement level of a set of tweets stored in a Pandas DataFrame called **tweetsDataFrame**.

First, the code creates a new column called "engagement level" in **tweetsDataFrame** by adding the "retweet_count" and "favorite_count" columns together for each tweet. This represents the total level of engagement (i.e., interaction) for each tweet.
Next, the tweets in **tweetsDataFrame** are grouped by date using the "created_at" column, and the total engagement level is calculated for each day using the **sum()** function. The resulting data is stored in a new Data Frame called **dailyEngagement**.

Finally, the code creates a line graph of the daily engagement level over time using the **plot()** function from the matplotlib library. The x-axis shows the date, while the y-axis shows the total engagement level for each day. The graph is given a title and the labels for both axes are set. The resulting graph is displayed using the **show()** function.

3.3.2 Similarity Analysis

After extracting tweets from the data extraction method, the tweets are saved in a csv file and then we perform similarity analysis code on the csv files. The code performs similarity analysis on a CSV file of tweets. It reads in the CSV file in chunks and appends them to an empty Data Frame. The user is prompted to enter two keywords for which similarity analysis will be performed. The code then extracts tweets containing these keywords and transforms them into TF-IDF vectors. The cosine similarity between the two sets of vectors is then calculated and plotted as a heatmap using seaborn. Finally, the index of the most similar tweets is obtained. The output provides an insight into how similar the tweets containing the two keywords are to each other.

This code performs similarity analysis on a set of tweets based on two given keywords. Here is a step-by-step explanation of the code: The required libraries are imported: pandas, seaborn, matplotlib, TfidfVectorizer, and cosine similarity.

1. A maximum number of rows to read at a time is set to 1000.
2. An empty pandas Data Frame called Similarity is created to store the data.
3. The CSV file is read in chunks of 1000 rows using the pd.read_csv() function and the current chunk is appended to the Similarity Data Frame using the append() function.
4. The loop breaks when the number of rows in the Similarity Data Frame exceeds 1000.
5. The user is prompted to enter two keywords to search for in the tweets.
6. Tweets containing the first keyword are extracted and stored in a new Data Frame called tweets_kw1 using the loc() function of pandas.
7. If no tweets are found containing the first keyword, a message is printed, and the program exits.
8. Tweets containing the second keyword are extracted and stored in a new Data Frame called tweets_kw2 using the loc() function of pandas.
9. If no tweets are found containing the second keyword, a message is printed, and the program exits.
10. The tweets are combined into two lists called docs1 and docs2, one for each keyword.
11. A TF-IDF vectorizer is created using the TfidfVectorizer() function from scikit-learn, with stop words removed.
12. The vectorizer is fit on the documents by calling the fit() function on the vectorizer object.
13. The documents are transformed into TF-IDF vectors by calling the transform() function on the vectorizer object for each list of documents.
14. The cosine similarity between the two TF-IDF vectors is computed using the cosine similarity() function from scikit-learn.

15. A heatmap is plotted using the heatmap() function from seaborn, with the cosine similarity matrix as input.
16. A title for the heatmap is set to indicate the two keywords being compared.
17. The indices of the most similar tweets are computed using the argmax() function on the cosine similarity matrix.

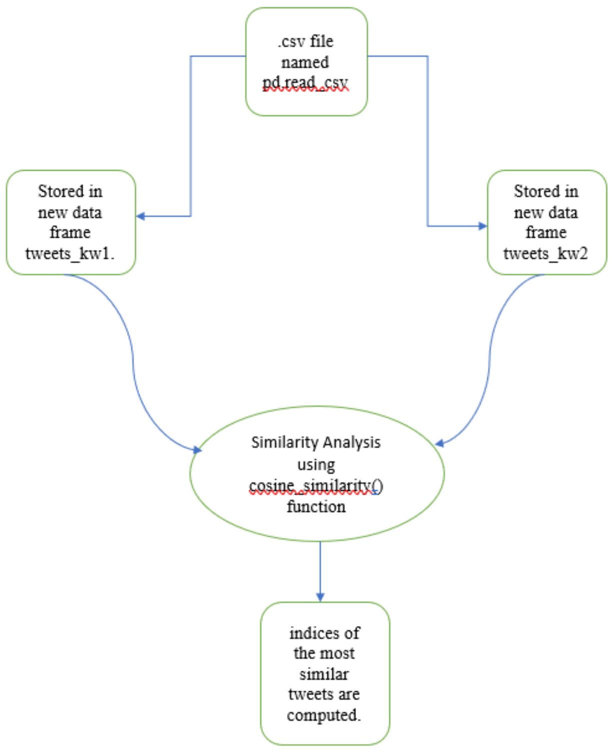

.csv file named pd.read_csv
Stored in new data frame tweets_kw1.
Stored in new data frame tweets_kw2
Similarity Analysis using cosine_similarity() function
indices of the most similar tweets are computed.

Figure 5: Similarity Analysis

3.3.3 Sentiment Analysis

We have taken the csv file that has been extracted from the data extraction code and performed sentiment analysis on a set of tweets using the Text Blob and Vader sentiment analysis libraries. It stores the tweets in a list variable, and then calculates the sentiment scores for each tweet using the Vader sentiment analyser. It then classifies each tweet as positive, negative, or neutral based on the polarity score, and stores each tweet in its respective category list.

The code also calculates the percentage of positive, negative, and neutral tweets, as well as the percentage of strongly positive and strongly negative tweets. It then formats these percentages and prints out the total number of tweets, as well as the number of tweets in each category.

Overall, this code is useful for analysing the sentiment of a set of tweets and categorizing them based on their polarity scores. Creating box plots and pie charts to visualize the distribution of different sentiments and separating neutral tweets into two categories based on their level of neutrality. The code performs sentiment analysis on a set of neutral tweets using the NRCLex library. It creates a data frame to store the results and initializes variables to count the frequency of different emotions in the tweets. The code then iterates over each tweet and calculates the emotion scores using the affect frequencies function of the NRCLex object. The maximum emotion score is stored, and the corresponding emotion is added to a list for each tweet along with the emotionless tweet. Finally, the emotion counts and lists are added to the results data frame.

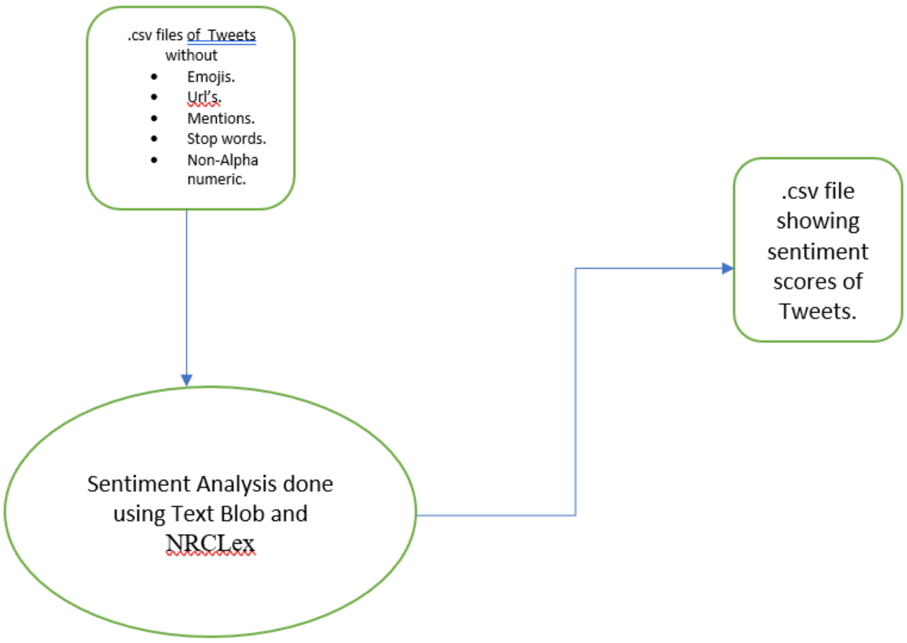

.csv files of Tweets without
Emojis.
Url's.
Mentions.
Stop words.
Non-Alpha numeric.
Sentiment Analysis done using Text Blob and NRCLex
.csv file showing sentiment scores of Tweets.

Figure 6: Sentiment Analysis

3.3.4 Visualization module

Transforming information into a visual context is part of data visualisation. A map or graph could be used to represent this information. Making data easy to understand and interpret for the human brain is advantageous in this situation. Data visualisation helps users more quickly spot patterns, trends, and outliers in very big data sets. In order to track results and make sure that models are operating as intended, visualisation is also crucial. Complex algorithm visualisations are typically simpler to understand than their numerical results.

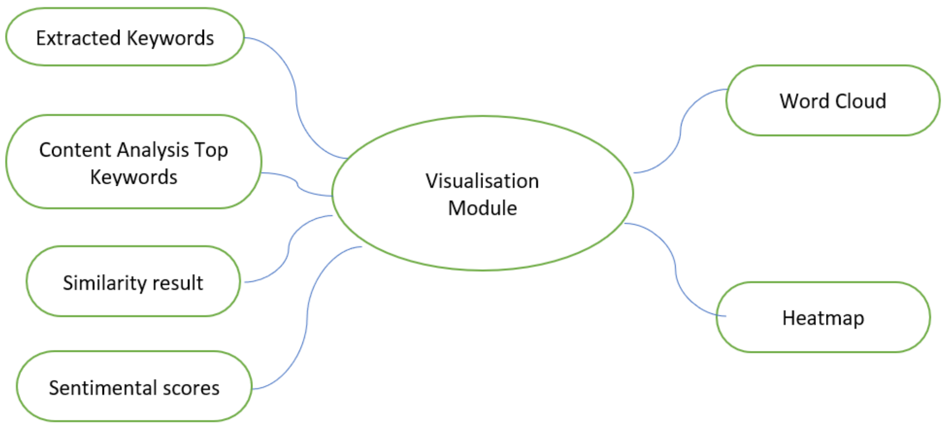


Figure 7: Visualisation Module

4. Testing

To ensure that the implemented features in a software system work properly and deliver the highest quality of performance, a unit testing process was conducted for all the functions used in each module as they were being developed. This means that each function was tested individually to check if it works as expected and meets the requirements. The unit testing process is a critical part of the software development process as it helps to identify any defects, errors, or issues in the functions before the entire system is integrated and tested as a whole.

During the unit testing process, each function was tested against a set of test cases designed to cover all possible scenarios and inputs. The purpose of this was to verify that the function performs as expected and returns the correct output for each input. If any issues were found during the testing process, they were addressed and fixed by the development team before moving on to the next stage of development.

The following information provides details about the tests that were conducted on each function.

4.1 Retrieving Data Module

Test Case ID:

TC-RD-01

Summary:

Input should be correct

Input:

1- Enter a city.

2- Enter a country

3- Enter a radius (in km)

4- Enter hashtags separated by a space

Steps:

Take “London” as a city

Take” UK” as country

Take Radius as “5000km”

Take one or two hashtags “iPhone” “Samsung”

Expected Result:

The code should retrieve tweets containing any of the hashtags "#iphone", "#samsung" within a radius of 500 km from London, UK, and store the tweet data in CSV files named "iphone_london_uk_5000km.csv", "samsung_london_uk_5000km.csv",

Actual Result:

Tweets will be retrieved based on the specified location, radius, and hashtags and saved in the CSV Successfully.

Test Case ID:

TC-RD-02

Summary:

City name should be incorrect

Input:

1- Enter a city.

2- Enter a country

3- Enter a radius (in km)

4- Enter hashtags separated by a space

Steps:

Take "Londoooon" as a city

Take" UK" as country

Take Radius as "5000km"

Take one or two hashtags "iPhone" "Samsung"

Expected Result:

The code should raise a geopy.exc.GeocoderTimedOut exception since the location "Londooon, United Kingdom" cannot be geocoded.

Actual Result:

Giving the error of incorrect city name successfully

Test Case ID:

TC-RD-03

Summary:

Input should be empty.

Input:

1- Enter a city.

2- Enter a country

3- Enter a radius (in km)

4- Enter hashtags separated by a space

Steps:

city: " "

country: " "

radius: " "

hashtags: " "

Expected Result:

The code should raise a geopy.exc.GeocoderQueryError exception since the location ", " cannot be geocoded and The code should print "Enter a city: " to prompt the user to enter a city.

Actual Result:

Error prompt generated "Enter a city."

Test Case ID:

TC-RD-04

Summary:

API credentials should be incorrect.

Input:

1- Enter a city.

2- Enter a country

3- Enter a radius (in km)

4- Enter hashtags separated by a space

Steps:

Take "London" as a city

Take" UK" as country

Take Radius as "5000km"

Take one or two hashtags "iPhone" "Samsung"

Expected Result:

The code should raise a tweepy.error.TweepError exception since the API credentials are incorrect.

Actual Result:

Incorrect API credentials error is displaying successfully.

4.2 Data Preprocessing

Test Case ID:

TC-DPR-01

Summary

Convert data to lowercase.

Input:

Take the extracted tweets from the CSV

Steps:

Convert the tweets into lower case using str.lower() function

Expected Result:

All the tweets' words should be converted into the lower case.

Actual Result:

Uppercase data converted into the lowercase successfully.

Test Case ID:

TC-DPR-02

Summary

Remove mentions and hyperlinks.

Input:

Take the extracted tweets from the CSV

Steps:

Remove the mentions "@[A-Za-z0-9_]+" and the (r"http\S+",) from the tweets

Expected Result:

All the links and mentions should be remove from the tweets

Actual Result:

Links and mentions removed successfully.

<u>Test Case ID:</u>

TC-DPR-03

<u>Summary</u>

Removal of stop words.

<u>Input:</u>

Take the extracted tweets from the CSV

<u>Steps:</u>

Remove the words that are repeating multiple times and has no value by importing the NLTK library stop words.

<u>Expected Result:</u>

All stop words should be removed

<u>Actual Result:</u>

All stop words removed successfully.

<u>Test Case ID:</u>

TC-DPR-04

<u>Summary</u>

Sorting the data date wise.

<u>Input:</u>

Take the extracted tweets from the CSV

<u>Steps:</u>

sort the tweet data according to the date sort_values(by='Date Created')

<u>Expected Result:</u>

Data should be sort according to the earlier date.

<u>Actual Result:</u>

Data sorted successfully.

4.3 Content Analysis

<u>Test Case ID:</u>

TC-CA-01

<u>Summary</u>

Empty Input

Input:

No data in the CSV enter empty data as an input

Steps:

perform content analysis on empty dataset

Expected Result:

Error message should be appeared.

Actual Result:

An error message is printed successfully since the function is not able to extract any relevant columns from an empty data frame.

Test Case ID:

TC-CA-02

Summary

Non-Empty Input

Input:

Normalized tweet data in the CSV taken as input

Steps:

perform content analysis on normalized dataset from data frames and count the frequency of each word in the tweet.

Expected Result:

Print the top 20 most frequent words with their percentage frequency.

Actual Result:

The top 20 most frequent words with their percentage frequency of more than 50% are printed successfully.

Test Case ID:

TC-CA-03

Summary

Input with Repeated Words

Input:

A data frame with several tweets containing the same words and phrases multiple times

Steps:

perform content analysis on input data frames

Expected Result:

Top 20 most frequent used words frequency should be printed proportional to the number of time word occurs.

Actual Result:

The top 20 most frequent words with their percentage frequency are printed, with the frequency percentage being proportional to the number of times the word occurs and also print the high frequency words successfully.

Test Case ID:

TC-CA-04

Summary

Input with Special Characters

Input:

A data frame with several tweets containing special characters like emojis, hashtags, and punctuation marks

Steps:

perform content analysis on input data frames with special characters, hashtags, emojis and punctuation marks

Expected Result:

Top 20 most frequent used words frequency should be printed proportional without special characters

Actual Result:

The top 20 most frequent words with their percentage frequency are printed, without including any special characters successfully.

4.4 Similarity Analysis

Test Case ID:

TC-SIMA-01

Summary

Both keywords return tweets.

Input:

Take two keywords as an input kw1 = "women", kw2 = "equality"

Steps:

extract the tweets of both the keywords do similarity analysis

Expected Result:

Similar tweets result should be display.

Actual Result:

A heatmap showing the cosine similarity between tweets containing "women" and "equality”. A list of tweets related to "women" and "equality" display successfully.

Test Case ID:

TC-SIMA-02

Summary

Only one keyword as input.

Input:

Take two keywords as an input kw1 = "women", kw2 = “ “

Steps:

try to extract the tweets of keywords and do similarity analysis

Expected Result:

A message should be indicating that no tweets were found for kw2

Actual Result:

Error message appears successfully.

Test Case ID:

TC-SIMA-03

Summary

Both keywords return the same tweets.

Input:

Take two keywords as an input kw1 = "feminism", kw2 = “feminism “

Steps:

try to extract the tweets of both keywords perform similarity analysis and save them in CSV

Expected Result:

Tweets that only contain feminism keyword should be display.

Actual Result:

A heatmap showing the cosine similarity between all tweets containing "feminism".

Test Case ID:

TC-SIMA-04

Summary

No tweets found for both keywords.

Input:

Take two keywords as an input kw1 = "sadadsa", kw2 = “4324dee3“

Steps:

try to extract the tweets of keywords for saving them in CSVs

Expected Result:

A message should be indicating that no tweets were found for both the keywords

Actual Result:

no tweets found message appears successfully.

4.5 Sentiment Analysis

Test Case ID:

TC-SA-01

Summary

Three-five-degree tweets Analysis.

Input:

Take input of tweets from the CSV

Steps:

Perform sentiment analysis through NLTK and sentiment analyzer on extracted tweets

Expected Result:

Count of total, positive, strongly positive, negative, strongly negative, and neutral tweets should be display.

Actual Result:

no tweets found message appears successfully.

Test Case ID:

TC-SA-02

Summary

sentiment Analysis on neutral tweets

Input:

Take input of tweets from the CSV

Steps:

Perform sentiment analysis through NLTK and vader_lexicon on extracted tweets

Expected Result:

neutral, mild neutral and pure neutral tweets count should be display

Actual Result:

neutral, mild neutral and pure neutral tweets counts display successfully.

Test Case ID:

TC-SA-03

Summary

NRC Emotion Extraction

Input:

Take input of tweets from the CSV

Steps:

Iterate over each tweet and perform emotion analysis

Expected Result:

Result of NRC emotions and emotionless tweets count should be display

Actual Result:

counts of NRC basic emotions and emotionless tweets display successfully.

4.6 Visualization and User Interface

Test Case ID:

TC-FC-01

Summary

Function files selection

Input:

Take input of the functions that you want to run after scrapping the data

Steps:

called that selected function from the files

Expected Result:

selected functions should perform its functionality on the scrapped tweets

Actual Result:

selected function works successfully.

Test Case ID:

TC-FC-02

Summary

invalid Function files selection

Input:

select invalid option from the list of the functions that you want to run after scrapping the data

Steps:

called that selected function from the files

Expected Result:

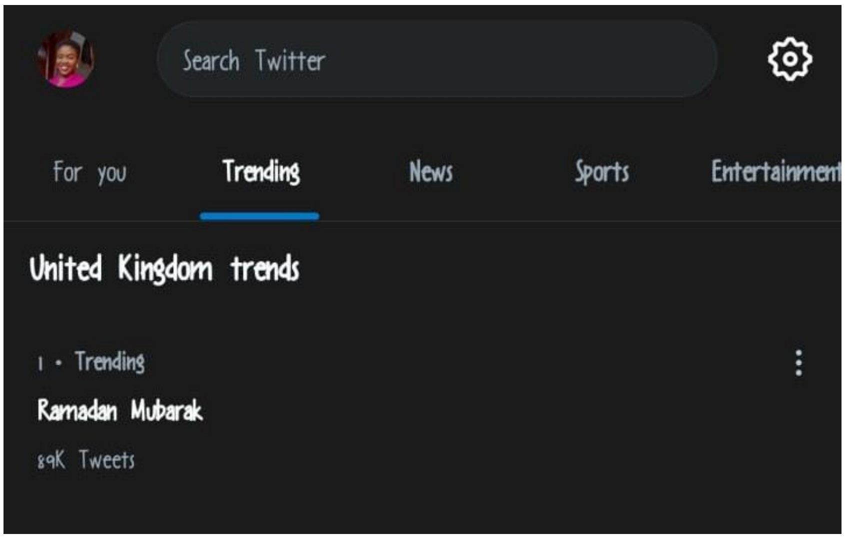
invalid number selection error message should be appear

Actual Result:

Error message for invalid function selection appears successfully.

5. Results

To display the effectiveness and implementation of the developed AI marketing tool and to suit the needs of our retrieval system, the codes were implemented on the 22 of March 2023, before submission. Now, according to Twitter the trending hashtag for the day within the United Kingdom was 'Ramadan Mubarak', which seems reasonable as the Muslim society had begun their fasting and we are getting into the Ramadan season.



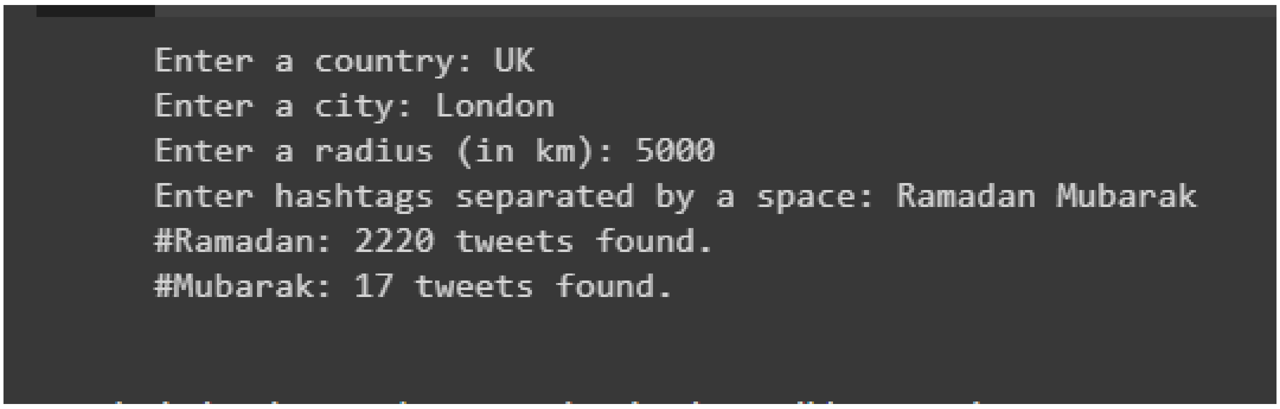
United Kingdom trends

1 · Trending
Ramadan Mubarak
89K Tweets

Fig 5.1 Twitter Mobile App Dashboard

Source: (1) Nneoma Charity (@nneoma_charity) / Twitter

From the Command line interface, the following were the results using the UK as country of choice, within a 5000km radius point from London, with the search hashtags being Ramadan and Mubarak. A total of 2220 tweets were retrieved out of the 89k tweets made all around the globe at the time. This portrayed that the search was really narrowed down to just the British community especially the environs of London.



```
Enter a country: UK
Enter a city: London
Enter a radius (in km): 5000
Enter hashtags separated by a space: Ramadan Mubarak
#Ramadan: 2220 tweets found.
#Mubarak: 17 tweets found.
```

Fig 5.2 Output Representation of the Command Line Interface

The first most indicative factor was the top 20 reoccurring words that were retrieved via the content analysis section of the AI tool. From this graph it is clearly shown that ‘Ramadan and month’ were the two top highest keywords popping up 1500 and 1000+ times respectively out of the 2220 tweets found.

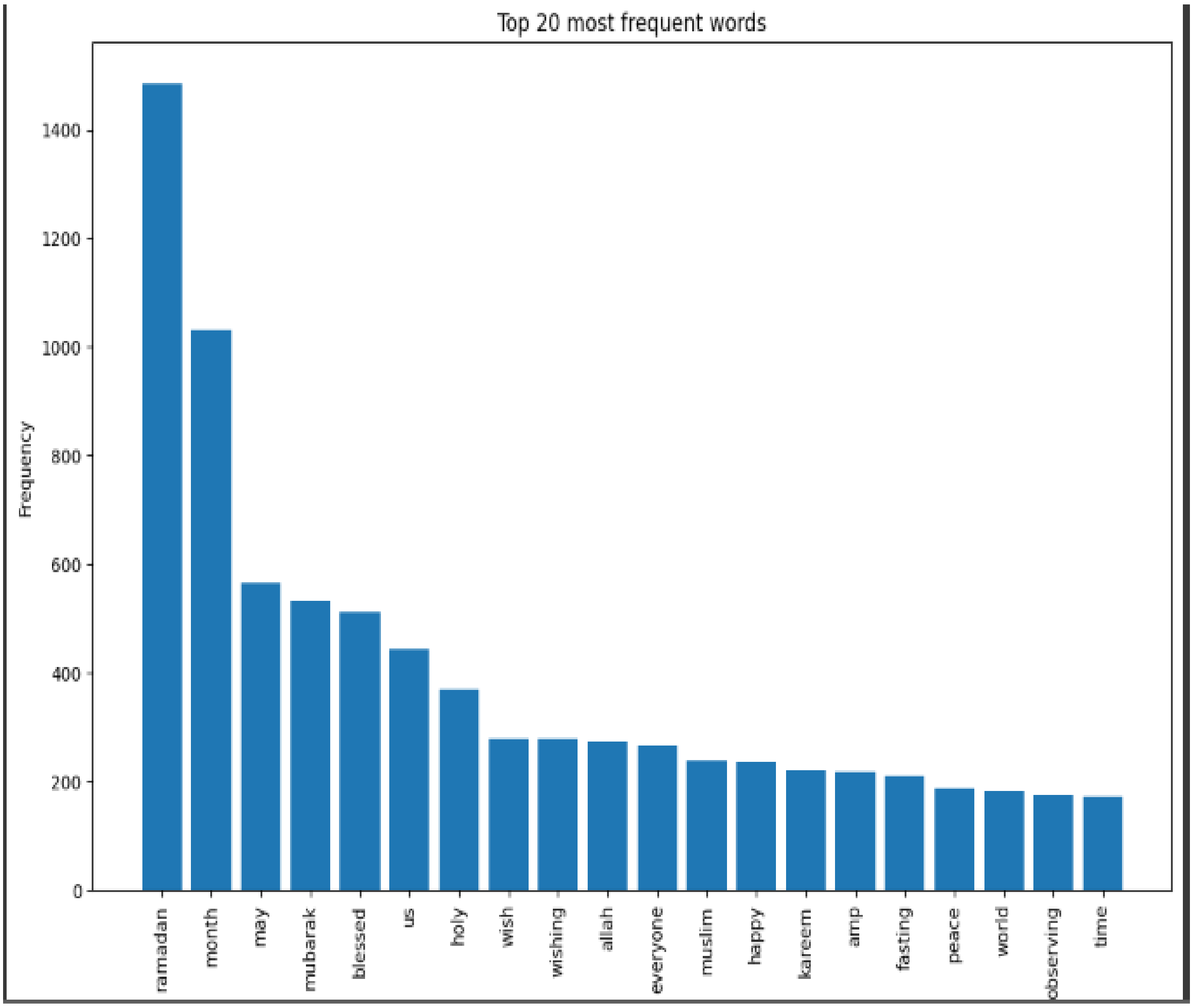


Fig 5. 3 Histogram of Top 20 words from Tweets

The word cloud library went further to give a clearer view and identification of these top keywords according to their capacities.

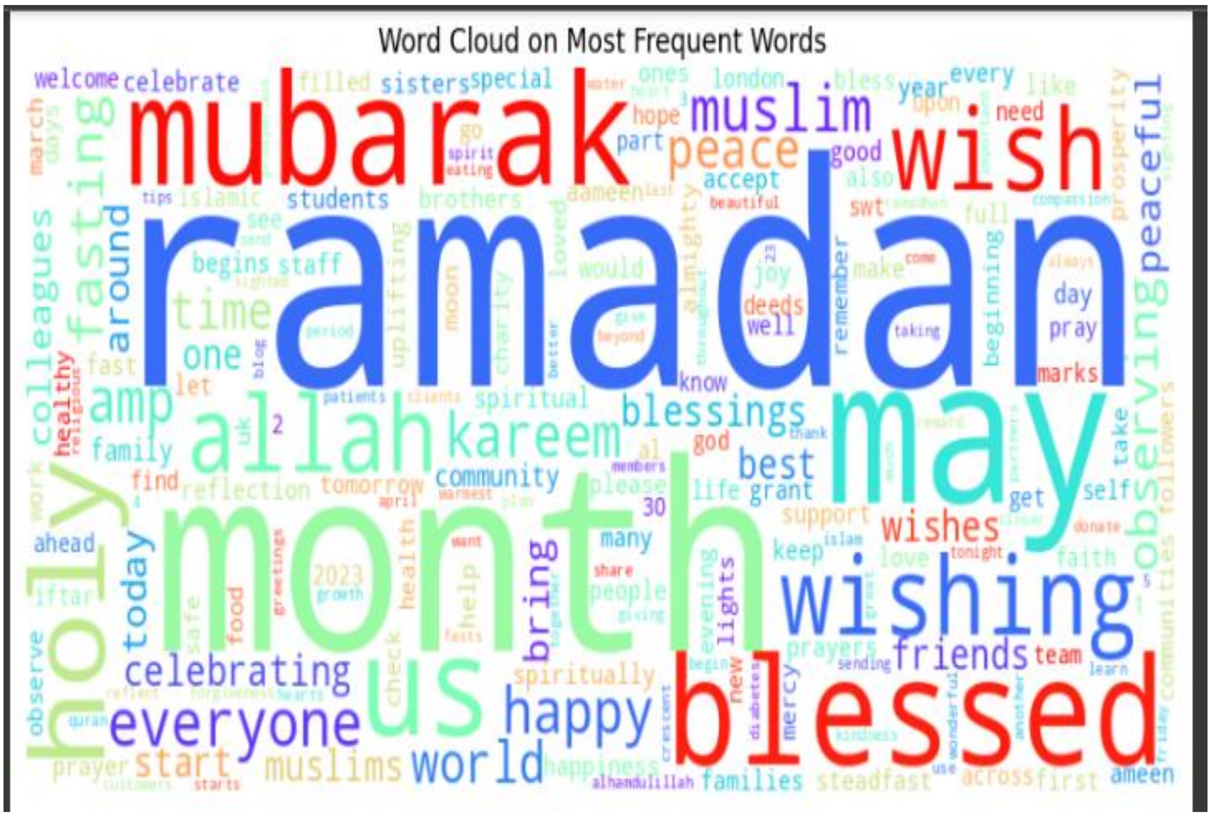


Fig 5.4 Word Cloud on Most Frequent Words

Using these top 2 highest ranking words, a similarity search was done yielding the below heat map:

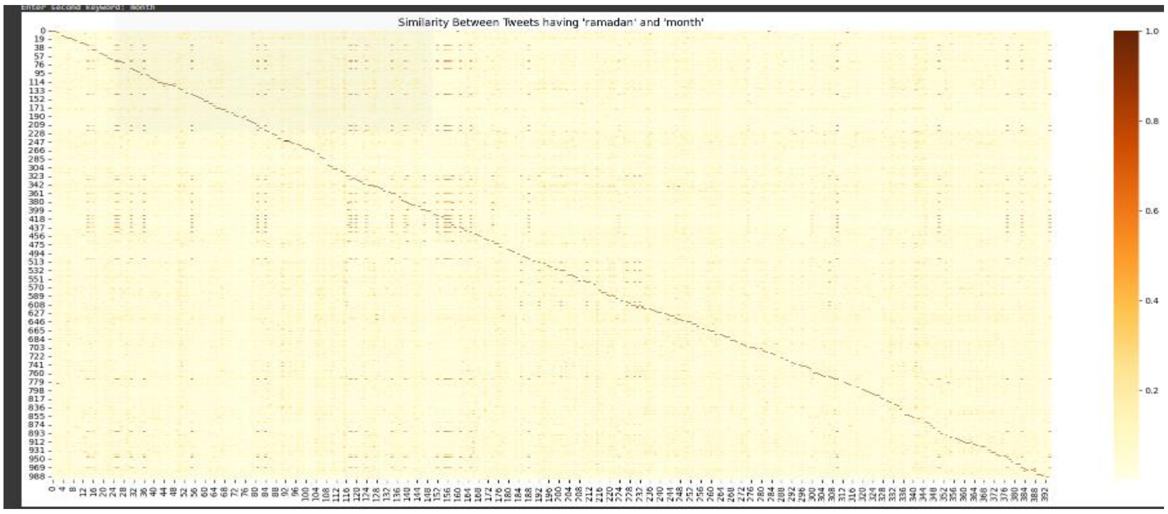


Fig 5.5 Heat Map of Similarity Analysis

To measure the level of engagement on the Ramadan and Mubarak hashtags according to date, the engagement section printed the following, indicating that these hashtags, within the past 7 days had zero data, hence were not used until on the 21st of March, spiking an engagement increase to an approximate level of 89000 till as at the time the data was retrieved.

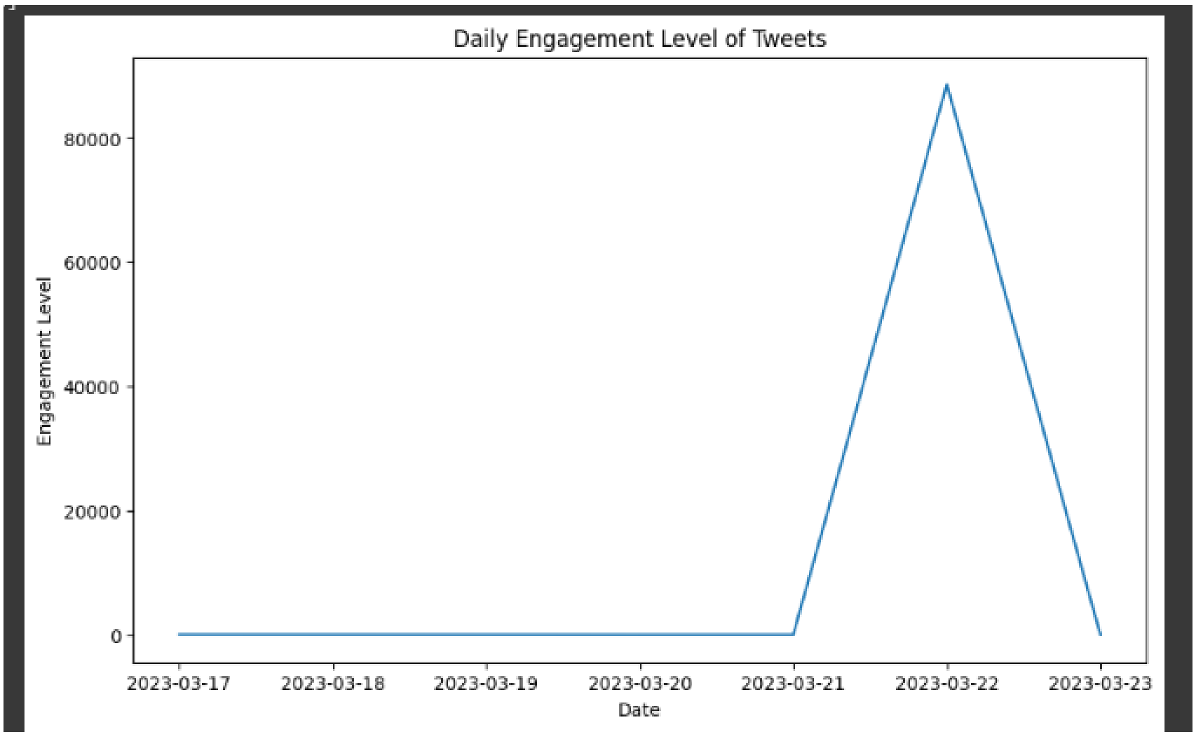


Fig 5. 6 Line Graph on the Engagement of Tweets having the hashtags.

Sentiments were the analysed in a boxplot to show the scores of the basic degrees of sentiments found in the data.

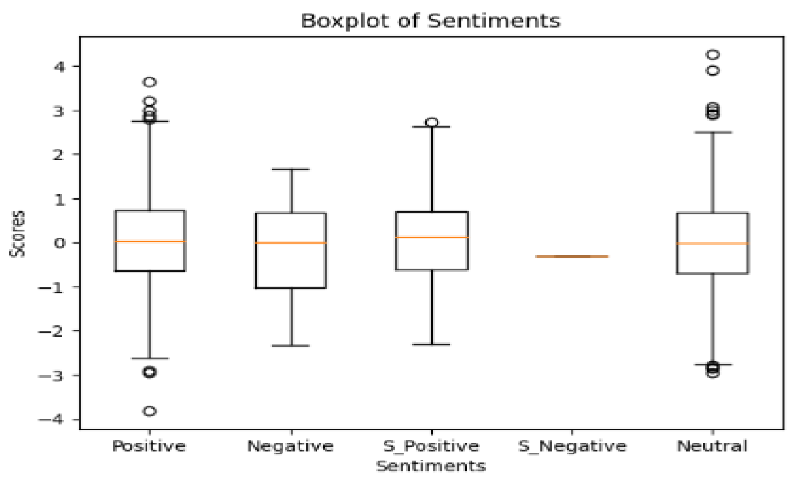


Fig 5. 6 Boxplots of the Degrees of Sentiments found in the Data.

A Pie chart was also used to demonstrate their different percentages for deeper insights.

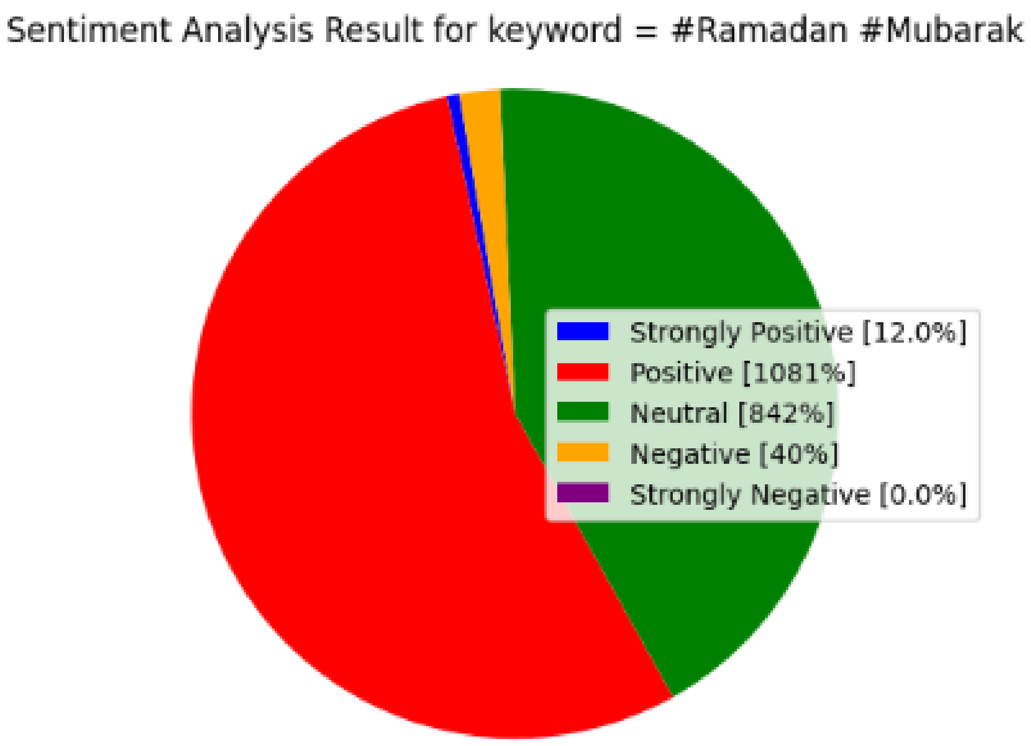


Fig 5.6 PieChart showing the Different sentiments categorized by colours in Percentages.

The Sentiment analysis went further to build on the neutrality of tweets separating pure neutrals from mild neutrals. However, from this data, none of these levels of neutrality was found.

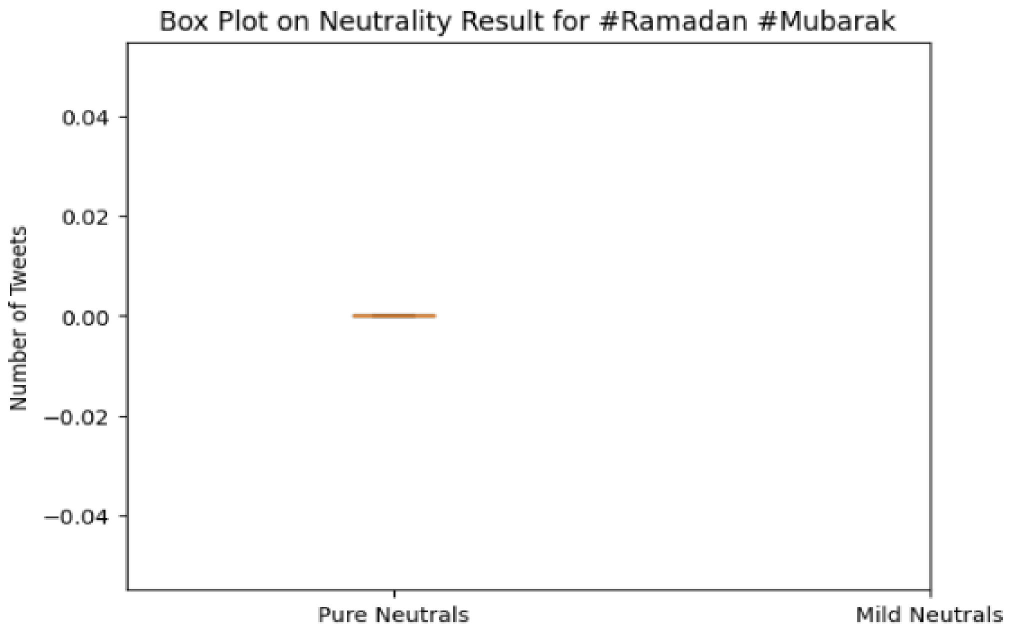


Fig 5.9 Boxplots of the Neutral Level of Tweets

6. Conclusion

This report depicts the efforts every member of the Team had enacted to make the identified goals/objectives come to play. Although we experienced some difficulties while trying to work together as some of us were not too grounded in our programming skills. But the combined motivation and push from the Supervisors and each team member led us to this stage. During the course of this project, we were able to delegate tasks, carrying out divisions of labour that made each one of us specialize in one thing. It's been a proper experience of everyone 'lifting where they stand'. The instruments utilized for project management, such as Gitlab and JIRA, have proven to be exceptionally efficient in fostering our capabilities in working as a team.

The objective of the project was to construct a prototype system that assists in understanding different digital marketing strategies through the employment of Artificial Intelligence. This tool serves as an aid for aspiring marketers to formulate marketing campaigns, with the basis of analytics on previous data from past marketing campaigns or social media trends.

Successfully, we have carried out a series of data manipulation and visualization using the AI tool we had developed that was built on different python Libraries. The entire exploratory data analysis was to gain insights into the distribution of sentiments in the dataset and from the example of the 'Ramadan Mubarak' keyword search, we found that the majority of the tweets were neutral in sentiment, followed by positive and negative. We were also able to identify the top trending words alongside the hashtags and show the similarity between one of the hashtags and another trending word. For the past weeks, we've developed and furnished this tool using different keywords and tags like IWD, Superbowl, Dream Crazier, etc. However, due to the limit bound of our retrieval API, we had to test run again using a trending keyword at this time in order to gain a good quantity of data and insightful results.

The limitations we experienced during this research were in line with the Data retrieval API as there was a 7days max access restriction from Twitter. This has limited our data towards just focusing on a one-week window period. However, this was tackled by an academic access request from Twitter, but as of the time of submission, this has still not yet been approved. Other issues faced were the generation of functions and explanations to facilitate users' accessibility around the code. We made strong efforts that every segment of the code had an ideal title, so non-technical users could easily find their way around it. During the advancement of this project, one crucial requirement that will be suggested will be to design a user-friendly interface in the form of a dashboard, that'll permit the user to have an array of options and flexibility and not be limited to just 5 analyses. Although these analyses build up the base, it’ll be more efficient and user-friendly if advanced in that way.

References

[1] G. S.-P. K. T.-M. V. M.-H. V. S. H. P.-M. A. Hernandez-Suarez, "A Web Scraping Methodology for Bypassing Twitter API Restrictions," 27 March 2018. [Online]. Available: https://arxiv.org/abs/1803.09875.

[2] Y. C. ,. A. A. ,. F. M. ,. Y. Q. ,. S. B. A. I. &. C. Authors: Irvin Dongo, "Web Scraping versus Twitter API: A Comparison for a Credibility Analysis," 27 January 2021. [Online]. Available: https://dl.acm.org/doi/abs/10.1145/3428757.3429104.

[3] Y. C. A. A. F. M. Y. Q. G. R. D. C. Irvin Dongo, "A qualitative and quantitative comparison between Web scraping and API methods for Twitter credibility analysis," 3 August 2021. [Online]. Available: https://www.emerald.com/insight/content/doi/10.1108/IJWIS-03-2021-0037/full/html.

[4] Z. Jianqiang and G. Xiaolin, "Comparison Research on Text Pre-processing Methods on Twitter Sentiment Analysis," 22 February 2017. [Online]. Available: https://ieeexplore.ieee.org/abstract/document/7862202.

[5] M. R. M. A F Hidayatullah, "Pre-processing Tasks in Indonesian Twitter Messages," 2017. [Online]. Available: https://iopscience.iop.org/article/10.1088/1742-6596/801/1/012072/pdf.

[6] S. S. &. A. A. Dimitrios Effrosynidis, "A Comparison of Pre-processing Techniques for Twitter Sentiment Analysis," 2 September 2017. [Online]. Available: https://link.springer.com/chapter/10.1007/978-3-319-67008-9_31.

[7] 2. A. O. I. Eradah O Hamad 1, M. Y. S. 3. A. O. Image, J. D. H. 4. A. O. Image, E. A. K. 4. A. O. Image and A. M. J. 3. A. O. Image, "Toward a Mixed-Methods Research Approach to Content Analysis in The Digital Age: The Combined Content-Analysis Model and its Applications to Health Care Twitter Feeds," 8 March 2016. [Online]. Available: https://www.jmir.org/2016/3/e60.

[8] S. M. J. S.-H. K. ,. A. W. Hwalbin Kim, "Evaluating Sampling Methods for Content Analysis of Twitter Data," 2018. [Online]. Available: https://journals.sagepub.com/doi/pdf/10.1177/2056305118772836.

[9] R. Z. ,. A. H. Seth C. Lewis, "Content Analysis in an Era of Big Data: A Hybrid Approach to Computational and Manual Methods," 12 March 2013. [Online]. Available: https://www.tandfonline.com/doi/abs/10.1080/08838151.2012.761702.

[10] N. S. ,. C. ,. Z. A. ,. J. M. C. Niloufar Shoeibi, "Similarity approximation of Twitter Profiles," 23 November 2021. [Online]. Available: https://www.researchgate.net/profile/Niloufar-Shoeibi/publication/356577895_Similarity_Approximation_of_Twitter_Profiles/links/61b76251a6251b553ab650d6/Similarity-Approximation-of-Twitter-Profiles.pdf.

[11] H. K. S. B. B. Kuldeep Singh, "Clustering of people in social network based on textual similarity," September 2016. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S2213020916301628.

[12] Z. J. M. A.-A. Y. J. Mohammad AL-Smadi, "Paraphrase identification and semantic text similarity analysis in Arabic news tweets using lexical, syntactic, and semantic features," May 2017. [Online]. Available: https://www.sciencedirect.com/science/article/abs/pii/S0306457316302382.

[13] S. A. E. Rahman, F. A. AlOtaibi and W. A. AlShehri, "Sentiment Analysis of Twitter Data," April 2019. [Online]. Available: https://ieeexplore.ieee.org/abstract/document/8716464.

[14] M. Trupthi, S. Pabboju and G. Narasimha, "Sentiment Analysis on Twitter Using Streaming API," 13 July 2017. [Online]. Available: https://ieeexplore.ieee.org/abstract/document/7976920.

[15] R. V. C. o. E. B. I. V. Prakruthi Department of MCA, D. Sindhu and D. S. A. Kumar, "Real Time Sentiment Analysis Of Twitter Posts," 25 July 2019. [Online]. Available: https://ieeexplore.ieee.org/abstract/document/8768774.

[16] F. N.-B. M. H. A. Zarei, "VISUALIZATION OF LOCAL MUNICIPAL SATISFACTION BY TWITTER DATA ANALYSIS," June 2019. [Online]. Available: https://www.csce.ca/elf/apps/CONFERENCEVIEWER/conferences/2019/pdfs/PaperPDFversion_130_0226091101.pdf.

[17] A. Sechelea, T. D. Huu, E. Zimos and N. Deligiannis, "Twitter data clustering and visualization," 30 June 2016. [Online]. Available: https://ieeexplore.ieee.org/abstract/document/7500379.

[18] W. Wolny, " Knowledge gained from twitter data," 7 November 2016. [Online]. Available: https://ieeexplore.ieee.org/abstract/document/7733390.