



A Project Report on

Whatsapp Chat Sentiment Analysis using

Natural Language Processing (NLP) Algorithm

Submitted by,

Nikhil Kulthe	(Exam Seat No. T224022)
Jatin Shevkari	(Exam Seat No. T224143)
Omkar Harkulkar	(Exam Seat No. T224011)
Abhijeet Patil	(Exam Seat No. T228018)

Guided by,

Mrs.Chetana Nemade

A Report submitted to MIT Academy of Engineering, Alandi(D), Pune,
An Autonomous Institute Affiliated to Savitribai Phule Pune University
in partial fulfillment of the requirements of

THIRD YEAR BACHELOR OF TECHNOLOGY in
Computer Engg.

School of Computer Engineering
MIT Academy of Engineering
(An Autonomous Institute Affiliated to Savitribai Phule Pune University)
Alandi (D), Pune – 412105

(2022–2023)

CERTIFICATE

It is hereby certified that the work which is being presented in the Third Year Project Implementation Report entitled “ **Whatsapp Chat Sentiment Analysis using Natural Language Processing (NLP) Algorithm**”, in partial fulfillment of the requirements for the award of the Bachelor of Technology in Computer Engg.and submitted to the **School of Computer Engineering of MIT Academy of Engineering, Alandi(D), Pune, Affiliated to Savitribai Phule Pune University (SPPU), Pune**, is an authentic record of work carried out during Academic Year **2022–2023 Semester VI**, under the supervision of **Mrs.Chetana Nemade, School of Computer Engineering**

Nikhil Kulthe (Exam Seat No. **T224022**)

Jatin Shevkari (Exam Seat No. **T224143**)

Omkar Harkulkar (Exam Seat No. **T224011**)

Abhijeet Patil (Exam Seat No. **T228018**)

Mrs.Chetana Nemade
Project Advisor

Dr.Amol Bhosale
Project Coordinator

Dr.Rajeshwari Goudar
Dean

External Examiner

DECLARATION

We the undersigned solemnly declare that the project report is based on our own work carried out during the course of our study under the supervision of **Mrs.Chetana Nemade**.

We assert the statements made and conclusions drawn are an outcome of our project work. We further certify that

1. The work contained in the report is original and has been done by us under the general supervision of our supervisor.
2. The work has not been submitted to any other Institution for any other degree/diploma/certificate in this Institute/University or any other Institute/University of India or abroad.
3. We have followed the guidelines provided by the Institute in writing the report.
4. Whenever we have used materials (data, theoretical analysis, and text) from other sources, we have given due credit to them in the text of the report and giving their details in the references.

Nikhil Kulthe	(Exam Seat No. T224022)
----------------------	--------------------------------

Jatin Shevkari	(Exam Seat No. T224143)
-----------------------	--------------------------------

Omkar Harkulkar	(Exam Seat No. T224011)
------------------------	--------------------------------

Abhijeet Patil	(Exam Seat No. T228018)
-----------------------	--------------------------------

Abstract

Today social media is the most used way to express emotions and share feelings, and WhatsApp is one of the leading social media platforms. The model emerging from the paper aims to perform a sentimental and emotional analysis of WhatsApp chats including text messages and emojis of the user. The creation of a model that categorizes opinions as either positive, negative, or neutral is crucial. The mining of behavior is another name for sentiment analysis. Using Natural Language Processing (NLP) and information retrieval techniques, sentiment analysis comprises behavior, opinions, and sentiments of the text, chat, or online communication. The model proposed in the paper aims to examine the impacts of a variety of machine-learning techniques for sentiment analysis. The behavior patterns are eradicated during the project. It will make it easier to determine whether the person enjoys using emojis and whether they do so as an adjunct to verbal communication or to express their emotions more fully. The benefit of the model is that it was developed using simple Python modules like pandas, matplotlib, seaborn, and NLTK which are used to create information frames and plot various types of graphs. The front end of the application is built in Python itself using Tkinter.

Acknowledgment

We want to express our gratitude towards our respected project advisor/guide Mrs. Chetana Nemade for her constant encouragement and valuable guidance during the completion of this project work. We also want to express our gratitude towards the respected School Dean Dr.Rajeshwari Goudar for her continuous encouragement.

We would be failing in our duty if we do not thank all the other staff and faculty members for their experienced advice and evergreen cooperation.

Nikhil Kulthe

Jatin Shevkari

Omkar Harkulkar

Abhijeet Patil

Contents

Abstract	iv
Acknowledgement	v
1 Introduction	1
1.1 Background	1
1.2 Motivation	2
1.3 Project Idea	2
2 Literature Review	3
2.1 Existing System	6
2.2 Proposed System	6
3 Problem Definition and Scope	7
3.1 Problem statement	7
3.2 Goals and Objectives	7
3.3 Hardware and Software Requirements	8
4 System Requirement Specification	9
4.1 Overall Description	9

4.1.1	Block diagram	9
4.1.2	Use Case Diagram	10
4.1.3	Sequence Diagram	11
4.1.4	Activity Diagram	12
5	Proposed Methodology	15
5.1	Algorithm	15
5.2	Approach	20
6	Implementation	25
6.1	System Implementation	25
6.2	Experiment/Implementation Parameters	28
6.3	Functional Implementation	30
6.4	Data Description	32
6.5	Output	33
7	Results and Discussion	35
7.1	Cleaning of data	35
7.2	Representing text in charts	36
8	Conclusion	56
8.1	Conclusion	56
8.2	Future Scope	57
	References	58

List of Figures

4.1	Block Diagram of WhatsApp Chat Sentiment Analyzer	9
4.2	Use case diagram of WhatsApp Chat Sentiment Analyzer	11
4.3	Sequence Diagram of WhatsApp Chat Sentiment Analyzer	12
4.4	Activity Diagram of WhatsApp Chat Sentiment Analyzer	13
6.1	Architecture Diagram	25
6.2	Data Flow Diagram	26
6.3	Flow Chart	27
6.4	WhatsApp Chat data in .txt format	28
6.5	Kaggle Dataset (preorganized dataset)	29
7.1	filtering all the messages	35
7.2	parsed chats	36
7.3	most media sent by the person.	36
7.4	most deleted chats by the person.	37
7.5	highest chats by the person.	38
7.6	highest positive polarity.	38
7.7	lowest polarity and negative polarity.	39
7.8	lowest polarity and negative polarity.	39

7.9	lowest polarity and negative polarity.	40
7.10	The sentiment analysis of chat.	41
7.11	Number of messages in the day	41
7.12	Number of messages in the day	42
7.13	Message and word count	42

List of Tables

5.1	Generalized Confusion Matrix	19
5.2	Confusion Matrix With Advanced Classification Metrics	20

Chapter 1

Introduction

1.1 Background

With more than 2.24 billion registered users across the world and one billion messages delivered daily in 60 different languages, WhatsApp is one of the most used social media platforms today [1]. WhatsApp plays a significant role in our lives. WhatsApp is used to communicate and express the most private feelings, which are frequently described as deep, meaningful conversations. Youth generation between the ages of 18 and 25 was found to spend 8 to 16 hours per day using WhatsApp, according to a survey conducted in India by Mood of the Nation (MOTN). It can be inferred from this that WhatsApp messages can reflect a person's mental process and express their feelings [2].

The pre-processing stage of text data mining is crucial. Any process involving the usage of data should start with data pre-processing. This is so that it produces data sets that are far more manageable, clearer, and coherent [3]. Pre-processing data consists of various steps depending on the type of data. In this project, we will obtain the data consisting of textual and graphic data comprising emojis as well. Hence, it becomes imperative that segregation is carried out before using them for any further analysis. In the project "WhatsApp Chat Sentiment Analyzer," we are going to work on pre-processing of data and NLP used for sentimental analysis as well as the importance of emojis during texting and emotion categories for analysis.

1.2 Motivation

Sentiment analysis has two driving forces. Customers' opinions about products and services are tremendously valuable to both consumers and manufacturers. Sentiment analysis has therefore observed correct smart effort from the industry as a whole. Sentiment analysis is to use this data to gather the essential knowledge about beliefs in order to create better corporate decisions, political campaigns, and increased product consumption [4].

1.3 Project Idea

future technologies are heavily dependent on data in this decade. Only by conducting a study on the context of the tool's requirements will it be possible to gather this data. Since a lot of machine learning enthusiasts create models that assist in solving numerous issues, a lot of appropriate data is needed. The model work on data pre-processing and NLP used for sentimental analysis as well as the significance of emojis while messaging and emotion categories for sentiment analysis. The model proposed in the paper aims to give users a better understanding of various chat types. This investigation demonstrates that it can provide ML models which essentially investigate chat data with superior input. The proposed model needs the right learning cases, which increases its accuracy. In-depth exploratory data analysis of numerous WhatsApp chat types is guaranteed by the model proposed

Chapter 2

Literature Review

WhatsApp got here into the marketplace rather than SMS. it is far used to make voice or video calls in addition to percentage media. Authors of “Survey Analysis on the Usage and Impact of WhatsApp Messenger” of their study of a collection of WhatsApp customers of ages 18-50 found that approximately seventy-9 percent of their topics use WhatsApp every day for a minimum of 15-60 minutes [4]. In the look “Impact of WhatsApp on Youth: A Sociological Study,” a hundred WhatsApp customers had been decided on randomly, lying among 18 and 30. They have a look at observed that the frequency of utilization for sixty-three percent of users is fifty times a day, twenty-one customers are twenty instances an afternoon and 16 percent is more significant than one hundred times an afternoon. it can be concluded that for some kids, WhatsApp greatly influences how they communicate [5].

Authors of “Facebook Sentiment: Reactions and Emojis,” argue that linguistic textual messages and emojis adjust one another’s that means. The interplay of linguistic textual content with emojis can range. An emoji can replace a phrase, toughen a word, autonomously bring the emotion or attitude of the subject, provide emphasis on an already existing emotion in the message, or can be used out of politeness. Given the context, emojis can be used to hit upon difficult emotions [6].

Authors of “Text Classification based Behavioral Analysis of WhatsApp Chats” posted in 2019, observed a machine to take multiple chats for each situation, and with the aid of an impartial community, every sentence and emoji is scored. The composition of the feelings expressed by the subject is less [7]. In the paper “Pre-Processing

and Emoji Classification of WhatsApp Chats for Sentiment Analysis” published in 2020, the authors discovered that WhatsApp is widely used internationally and the growth in usage of emojis is growing which is the most beneficial fact for sentiment analysis. we ought to carry out various statistics mining techniques, due to which our version will become heavy and slows down velocity [8]. From the paper “A Combination of Machine Learning and Lexicon Based techniques for Sentiment Analysis” posted in April 2020 we get there may be a want for a method that will categorize the critiques into wonderful, negative, and impartial sentiments and we will do it via NLP set of rules and records retrieval strategies. The method that categorizes the opinion of sentiment is mechanically run, so it could have much less accuracy [9].

“Code Mixing: A Challenge for Language Identification in the Language of Social Media” describes that despite most social media content material is within the English Language; this nevertheless makes up for the simplest half of the content material worldwide. this is because most customers switch between languages mid-way [10]. In a look at the finish with the aid of M Tafaquh Fiddin Al Islami, Ali Ridho Barakbah, and Tri Harsono within the paper “Interactive Applied Graph Chatbot with Semantic Recognition” posted in October 2020, authors have discovered that there is a want to increase the retention charge of the customer and we can do it by using chat-bot and another cost-effective way is the way of studying sentiment via chats and emoji’s [11]

“Code Mixing: A Challenge for Language Identification in the Language of Social Media” describes that despite most social media content material is within the English Language; this nevertheless makes up for the simplest half of the content material worldwide. this is because most customers switch between languages mid-way [10]. In a look at the finish with the aid of M Tafaquh Fiddin Al Islami, Ali Ridho Barakbah, and Tri Harsono within the paper “Interactive Applied Graph Chatbot with Semantic Recognition” posted in October 2020, authors have discovered that there is a want to increase the retention charge of the customer and we can do it by using chat-bot and another cost-effective way is the way of studying sentiment via chats and emoji’s [11]

In a have a look achieved by the scholars of Abu Dhabi, “Trends and Impact of WhatsApp as a Mode of Communication among Abu Dhabi Students,” 80-five per-

centage of girls and seventy percent of guys expressed that they used emojis rather than facial expressions in the textual content. Their average usage becomes 1 to 7 hours/in line with the day [12]. In an observation “Learning from the ubiquitous language: an empirical analysis of emoji usage of smartphone users” on four million customers of different international locations, it became proven that about six million i.e., seven percent of all messages contained at least one emoji. this is a sign of the recognition of emojis among users [13]. In the paper “An Emotional Topic Recommendation Chat Assistant” published in might also 2021, the authors have found out that assistants looking to address this trouble with the aid of recommending fantastic and negative emotional topics to customers. there is an absence of tone and nonverbal cues whilst chatting on WhatsApp [14]

In the version prescribed within the paper “Social emotion mining techniques for Facebook posts reaction prediction,” preliminary pre-processing like converting to lowercase, casting off hashtags, and so on. has been carried out using the Stanford CoreNLP Parser that is observed the use of a tokenizer to split posts based on spaces after filtering the stop phrases [15]. The paper “Are Emoticons Good Enough to Train Emotion Classifiers of Arabic Tweets?” , showcases how emojis may be used to carry out the automatic labeling of tweets. automated labeling of tweets outperforms guide labeling of tweets [16].

The software of sentiment evaluation has been prolonged in latest years. for example, the sentiment evaluation techniques used for processing reviews given about clinical papers. ”Sentiment analysis and opinion mining applied to scientific paper reviews” [17]. The sentiment analysis implemented Twitter textual content in ”Twitter text mining for sentiment analysis on people’s feedback about Oman tourism” to investigate the comments of vacationers [18]. In another painting, the sentiment analysis software on product evaluations is presented in ”Sentiment analysis on product reviews using machine learning techniques” [19]. In this paper, the class strategies which include NB have been used. For a complete survey on sentiment evaluation, readers are referred to ”The evolution of sentiment analysis—A review of research topics, venues, and top cited papers” [20].

2.1 Existing System

There are heaps of development within the current system. within the older version, there was no feature to show standing, there was no feature to share documents and there was no feature to share the situation. within the current version, of these options are obtainable. within the older version, we tend to could not share pictures through the doc's format. during this system, users will access WhatsApp in windows through the WhatsApp net application, which might be connected through a QR code. there is another feature known as export chat wherever users will send or share or get the chat detail for knowledge analysis through email, Facebook, or some traveller application.

2.2 Proposed System

In the projected system, knowledge pre-processing, the initial part of the project is to know the implementation and usage of assorted python-built modules. The top method helps the U.S.A. perceive why totally different modules are useful instead of the developer implementing those functions from scratch. These numerous modules offer higher code illustration and user quality. the subsequent libraries are used: NumPy, SciPy pandas, CSV, sklearn, matplotlib, sys, re, emoji, nltk seaborn, etc. In beta knowledge analysis, the primary step this to use a sentiment analysis formula that gives positive negative, and neutral elements of the chat and is employed to plot a chart supported by these parameters. To plot a line graph that shows the author and message count of every date, to plot a line graph that shows the author and message count of every author, ordered graph of date vs message count, media sent by authors and their count, show the message that is di not have authors, plot graph of associate degree hour vs message count.

Chapter 3

Problem Definition and Scope

Motivation for Sentiment Analysis is two-fold. Both consumers and producers highly value “customer’s opinion” about products and services. Thus, Sentiment Analysis has seen considerable effort from the industry as well as academia. The goal of Sentiment Analysis is to harness this data in order to obtain important information regarding public opinion, that would help make smarter business decisions, political campaigns, and better product consumption.

3.1 Problem statement

To Analyze WhatsApp Chat Sentiments using the Natural Language Processing (NLP) Algorithm.

3.2 Goals and Objectives

1. Provide an in-depth exploratory data analysis on various types of chat.
2. Recommend emojis for the chats.
3. Analyze and process the content of chats.

3.3 Hardware and Software Requirements

(I) Hardware Requirements:

1. Windows 7 or 10,
2. x86 64-bit CPU (Intel/AMD architecture),
3. 4GB RAM,
4. 5GB free disk space.

(II) Software Requirements:

1. Browsers: Google Chrome, Edge, Safari, etc.
2. Interface: Windows
3. Supported system to: Python, JSON, DART, Flutter.

(III) Frontend: GUI using Python Tkinter

(IV) Backend: Data Analysis using Python

(V) Library Packages:

1. NumPy
 2. Pandas
 3. Sk-learn
 4. Matplotlib
 5. Seaborn
 6. Tkinter
 7. NLTK
-

Chapter 4

System Requirement Specification

4.1 Overall Description

4.1.1 Block diagram

A block diagram is a picture of a system that shows relationships between objects by connecting basic, labelled blocks to form single and many objects, entities, or concepts.

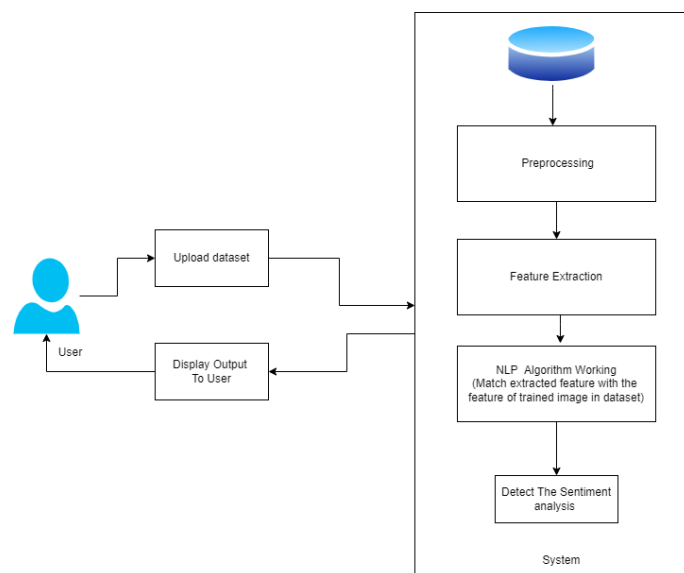


Figure 4.1: Block Diagram of WhatsApp Chat Sentiment Analyzer

1. Upload dataset: Upload the dataset in textual format.

2. Pre-processing: Unwanted or unimportant material may be present in the raw text data, which could cause our results to be inaccurate and difficult to understand and analyse.
- 3 . Feature extraction: A variety of techniques are available for removing features from text, including Bag of Words, TF-IDF, word embedding, and NLP (Natural Language Processing)-based traits like word count and noun count. In the study, we investigated the impacts of TF-IDF word level and N-Gram on the dataset of sentiment analysis.
4. NLP Algorithm working: Sentiment analysis is a method of analysis that identifies the emotional content of communications by combining machine learning, natural language processing, and statistics.
5. Detect the sentiment Analysis: It detects the output

4.1.2 Use Case Diagram

The simplest way to describe a user's interaction with the system and to show its specifications is via a use case diagram. A use case diagram can show the many system kinds and the various ways in which they communicate with the system. Figure 4.2 depicts the use case diagram of the WhatsApp Chat Sentiment Analyzer.

1. The First user will enter the data into the analyzer. The data is a WhatsApp chat in a textual format document.
2. Retrieving the metadata for each set and further it will get stored into the database of the analyzer for future processing.
3. The store data now get trained. Extract the significant phrases for each WhatsApp chat. Then perform the sentiment analysis on each WhatsApp chat. Again, collect all the trained data into the database of an analyzer.
4. Now testing of data takes place. Here the plotting graphs and word cloud mean sentiment analysis report working is going as well as proper emoji suggestion report working takes place.

5. Finally, the analyzer displays the result to the user as sentiment analysis of her entered chats on display and proper emoji suggestions for the chats or feelings.

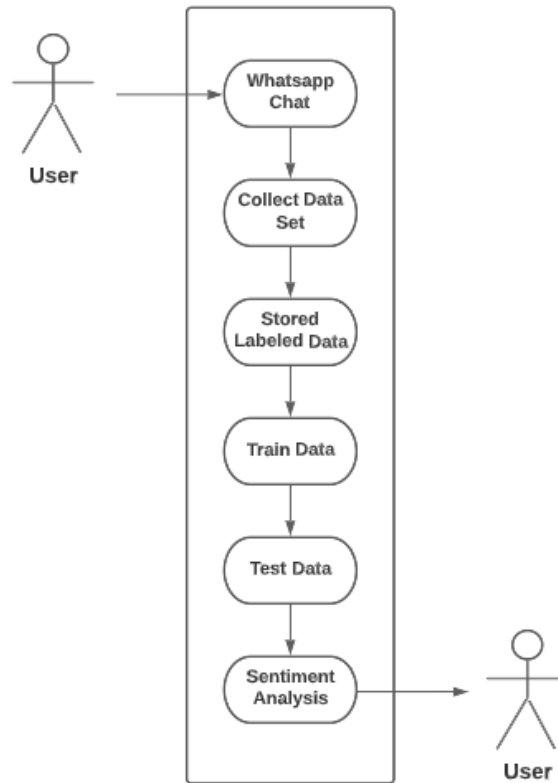


Figure 4.2: Use case diagram of WhatsApp Chat Sentiment Analyzer

4.1.3 Sequence Diagram

The sequence diagram describes how and in what order the group of objects works together. In this project, the given sequence diagram, figure 4.3, shows the order of execution starting from the user where we get WhatsApp data, verify that data is valid and make the analysis and determine the chat and at the end give analysis to the user block.

1. Firstly, the application is opened by the user. The application demands WhatsApp chat data as input to analyze it. The user must submit the demanded data in a textual format to the application for further processing.
2. The application takes the input given by the , user. It checks whether the given data is valid or not. If it is not a valid process will be terminated by the

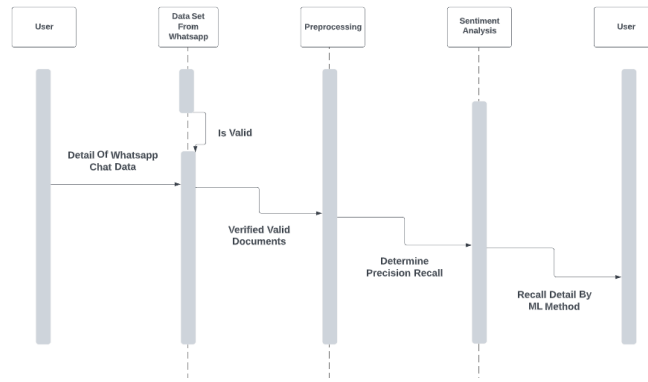


Figure 4.3: Sequence Diagram of WhatsApp Chat Sentiment Analyzer

application and notify to the user. If it is valid, documents get verified, and send these verified documents to the next step of sentiment analysis.

3. Pre-processing block accepts the data send for pre-processing. Here the pre-processing methodology is getting applied and determines the precision and recall.
4. Precision and recall are performance metrics used for pattern recognize on and classification. The proportion of accurately classified positive samples (True Positive) to all classified positive samples is what is referred to as precision (either correctly or incorrectly). Precision enables us to see the ML model's dependability in identifying the model as positive. Precision is defined as $\text{True Positive} / (\text{True Positive} + \text{False Positive})$. The recall is determined as the proportion of positively classified positive samples to all positively classed samples. The model's capacity to identify positive samples is measured by recall. Recall is defined as $\text{True Positive} / (\text{True Positive} + \text{False Negative})$.
5. Pre-processed data is sent to the sentiment analyzer block. Here NLP algorithm is applied to get a sentiment analysis of the user's data.
6. Finally, users get the result into sentiment analysis and proper emoji suggestions

4.1.4 Activity Diagram

The activity diagram illustrates the flow of management in an exceedingly system and refers to the steps concerned within the execution of the use case. we tend to

model consecutive and coincidental activities exploitation activity diagrams. So, we tend to primarily depict workflows visually exploitation associated with a nursing activity diagram. The above figure 4.4 depicts the activity of the WhatsApp Chat

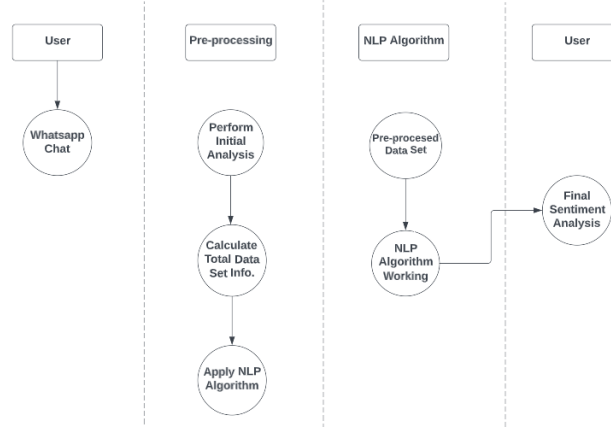


Figure 4.4: Activity Diagram of WhatsApp Chat Sentiment Analyzer

Sentiment Analyzer

1. The start node is the user and the workflow of the sentiment analyzer starts from the user. The user gives the WhatsApp chat as input to the analyzer.
2. Then the control flow goes to the pre-processing step. Pre-processing step performs the three activities. First, perform an initial analysis of the input given by the user. Then calculate the total dataset information. Lastly, apply the NLP algorithm. Here are some pre-processing tasks:

- Tokenization: reduces text to single clauses or semantic components
- Making up words like nouns, verbs, adjectives, adverbs, etc. is known as part-of-speech tagging.
- Standardizing words by reducing them to their base forms is known as stemming and lemmatization.
- Stop word removal: removing typical words like prepositions and articles that don't contribute much or any distinctive information.

Next, the control flow goes to the NLP Algorithm block. Here actual working of the NLP algorithm takes place. The pre-processed dataset is undergoing

feature extraction and then calculations of NLP algorithms take place.

General mathematical formulas for understanding NLP calculations:

$$IDF(t) = \log N / (1 + df) \quad (4.1)$$

$TF(t,d) = (\text{Number of times } t\text{-appears in } d) / (\text{total number of terms in } d)$

$$TF - IDF(t, d) = TF(t, d) * IDF(t) \quad (4.2)$$

here, d=concept of documents N=number of document t=frequency

Lastly control flow goes to the user again and the workflow of activity ends here. The user gets the result of data that he entered into sentiment analysis and emoji suggestions.

Chapter 5

Proposed Methodology

5.1 Algorithm

The four stages of the method for sentiment analysis that is suggested in the study are:

- A. Data Collection
- B. Data Pre-processing
- C. Feature Extraction
- D. NLP Algorithm working

Data Collection:

The gathering of data is the first step in the analysis. One of the most important components of the sentiment analysis application is data collection. On a task that is particular to a domain, greater performance is not automatically guaranteed by massive datasets. The quality of the dataset affects the model's performance.

There are two possible ways to gather the data. The first is to collect preorganized data. Preorganized data is available on different sites, on these sites data is uploaded by developers for research purposes. The second way is the collect data manually. The model requires textual data so, chat data should be extracted into text files. This can be done by following steps:

- Creation of a WhatsApp account.
- Open WhatsApp. Select the chat (person/group) who wants to check his sentiment.
- Click on the tab, click on more, and select ‘export chat’.
- Select export chat ‘without media’.
- Copy the chats into the ‘.txt’ file.

During the proposed experimentation, we used the Python ‘textblob’ library to annotate the polarity of chats in pre-data.

Data Pre-processing:

Data pre-processing includes the steps we must take to change or encrypt data so that a machine can quickly and easily understand it. The algorithm’s ability to swiftly analyze the characteristics of the data is essential for the model to produce precise and accurate predictions. The majority of real-world machine-learning datasets are very susceptible to missing, inconsistent, and noisy data because of their diverse origins. Hence, pre-processing data is an essential step in sentiment analysis. Here are some pre-processing tasks:

- Tokenization: reduces the text to single clauses or semantic components
- Making up words like nouns, verbs, adjectives, adverbs, etc. is known as part-of-speech tagging.
- Standardizing words by reducing them to their base forms is known as stemming and lemmatization.
- Stop word removal: removing typical words like prepositions and articles that do not contribute much or any distinctive information.

Several opinions are conveyed in chats in various methods by various people. The WhatsApp dataset utilized in the proposed experimentation model work has already been categorized into classes for the neutral, negative, and positive polarity, making

it simple to do sentiment analysis on the data and see how different factors affect sentiment. Raw data with polarity is very prone to redundancy and inconsistency. Pre-processing of chats includes the following points,

- Eliminate all URLs (such as www.abcd.com), targets (such as @username) and media files
- Check the spelling; repeating characters should be handled.
- Substitute their sentiments for each emoticon.
- Eliminate all punctuations, symbols, and numbers.
- Eliminate Stop Words.
- Expand Acronyms.
- Delete non-English chats.

Feature Extraction:

The technique of turning raw data into numerical features that can be handled while keeping the information in the original data set is known as Feature Extraction. The pre-processed dataset has a lot of unique characteristics. We take the features from the processed dataset using the feature extraction approach. Eventually, using models like unigram and bigram, this characteristic is utilized to compute the positive and negative polarity of a sentence, which is helpful for assessing people's opinions. Another open-source Python module that is available and that can be used to extract similar traits is called Natural Language Tool Kit (NLTK).

In order to process text or documents, machine learning approaches need to be able to express their main aspects. These important characteristics are referred to as feature vectors and are utilized in the classification process, these are mentioned as follows:

1. *Words and Their Frequencies:* Features include unigrams, bigrams, and n-gram models with frequency counts. The use of word presence as opposed to word frequencies to better explain this attribute has received greater research.

2. *Language Structure* Adjectives, adverbs, and specific verb and noun groupings are effective indications of subjectivity and sentiment. Through parsing or dependency trees, we can produce syntactic dependence patterns.
3. *Opinion Terms and Phrases*: In addition to specific words, idioms, and phrases that express feelings can be employed as features.
4. *Words' Placement*: A term's place inside a text can have an impact on how much the term changes the tone of the text as a whole.
5. *Negation*: Interpreting negation is a crucial but challenging aspect. The inclusion of a negative typically modifies the opinion's polarity. e.g., I am not feeling comfortable.
6. *Syntax*: Several researchers employ syntactic patterns, such as collocations, as features to learn subjective patterns.

NLP Algorithm Working: The field of study known as natural language processing, or NLP for short, studies how language and computers interact with one another. It is at the intersection of computer science, artificial intelligence, and computational linguistics.

There are two parts to the algorithm first is the model selection and the second is model evaluation.

1. Model Selection: The pre-processed data must then be given to a classification model for additional processing. These models are constructed using various classification methods. In the study, the categorization was carried out using the k-nearest neighbor approach.

An algorithm for classifying a set of data into its assigned goal values—in this case, positive, neutral, or negative—is known as the KNN or k-Nearest Neighbor algorithm. Although KNN is frequently used for classification issues, it might also be utilized for regression issues.

Any classification model must now be trained on a target set before being used. Most of the references in the section on the literature review have manually set these target values to positive, negative, or neutral. To set the target for chats automatically, the

Python Textblob package is used in the proposed system.

For this, a different Python module called sklearn, which includes a variety of classification models and encoders is utilized. This library is used for model selection, label encoding, and model evaluation, which are covered in the next section.

2. Model Evaluation: Logistic Regression or Confusion matrices are one of the most popular and useful techniques for classifier evaluation. In the table below, a generalized version of the confusion matrix is provided:

Table 5.1: Generalized Confusion Matrix

	Actually Positive (1)	Actually Negative (0)
Predicted Positive (1)	True Positives (TP)	False Positives (FP)
Predicted Negative (0)	False Negatives (FN)	True Negatives (TN)

The generalized evaluation parameters can be determined by using this method. These parameters consist of:

- 1. Accuracy:** Accuracy is one metric for evaluating classification models. Informally, accuracy is the fraction of predictions our model got right. Formally, accuracy has the following definition:

$$\text{Accuracy} = \text{Number of correct predictions} / \text{Total Number of predictions}$$

Accuracy can be calculated in terms of positive and negative as:

$$\text{Accuracy}(a) = (TP + TN) / (TP + TN + FP + FN) \quad (5.1)$$

- 2. Precision:** This is the portion of actually translated items relative to the total number of items to be translated, it can be calculated as:

$$\text{Precision}(p) = TP / (TP + FP) \quad (5.2)$$

3. **Recall:** This is the portion of successfully retrieved items to the total number of demanded items, and is calculated as:

$$Recall(r) = TP / (TP + FN) \quad (5.3)$$

4. **F-measure:** This is the harmonic mean of recall and precision, and calculated as:

$$F - measure = 2p.r / (p + r) \quad (5.4)$$

The confusion matrix with advanced classification metrics is given below:

Table 5.2: Confusion Matrix With Advanced Classification Metrics

	Positive	Negative	Predicted Class
Positive	True Positive (TP)	False Negative (FN)	Recall $\frac{TP}{TP+FN}$
Negative	False Positive (FP)	True Negative (TN)	Specificity $\frac{TN}{TN+FP}$
Actual Class	Precision $\frac{TP}{TP+FP}$	Negative Predictive Value $\frac{TN}{TN+FN}$	Accuracy $\frac{TP+TN}{TP+TN+FP+FN}$

5.2 Approach

A. Machine Learning Approach:

There are two approaches, namely Unsupervised learning, and Supervised Learning

to classify text. In the proposed experimentation model, a supervised learning approach used for classification as the classifier makes simpler future predictions for unknowable data.

It is distinguished by the way it trains computers to accurately classify data or predict outcomes using labeled datasets. The model modifies its weights as input data is fed into it until the model has been properly fitted, which takes place as part of the cross-validation process. A training set is used in supervised learning to instruct models to produce the desired results. This training dataset has both the right inputs and outputs, enabling the model to develop over time. The loss function serves as a gauge for the algorithm's correctness, and iterations are made until the error is sufficiently reduced.

Three techniques are used for supervised learning in the proposed experimentation model are:

1. TF-IDF Vectorizer
2. Support Vector Machines
3. Grid Search

1. TF-IDF Vectorizer:

TF-IDF means the Term Frequency Inverse Document Frequency of Records. It is among the most well-liked and successful methods for natural language processing. With this method, you may determine how significant a term is about all other terms in a document.

The Term frequency (tf) with which a keyword appears in a specific document. Hence, it is unique to a document. It can be calculated by,

$$tf(t) = (No. \text{ of times term 't' occurs in a document}) / (No. \text{ of terms in document})$$

The Python sklearn library uses the following formula for calculating tf,

$$tf(t) = No. \text{ of times term 't' occurs in a document}$$

Inverse Document Frequency (idf) is a metric used to assess a term's frequency or rarity across the full corpus of documents. The important thing to remember is that it is shared by all of the documents. The idf value (normalized) will be close to 0 if the term is widespread and many documents contain it, or it will be close to 1 if the word is uncommon. It can be calculated by,

$$idf(t) = 1 + \ln(n/df(t)) \quad (5.5)$$

Where,

n = Total number of documents available

t = term for which the idf value has to be calculated

df(t) = Number of documents in which the term t appears

The Python sklearn library uses the following formula for calculating idf,

$$idf(t) = \ln(1 + n/1 + df(t)) + 1 \quad (5.6)$$

(default i.e smooth idf = True)

and

$$idf(t) = \ln(n/df(t)) + 1 \quad (5.7)$$

(when smooth idf = False)

The product of a term's tf and idf determines its tf-idf value. The more relevant a term is in that document, the higher its value. TF-IDF Algorithm the following sequence:

1. Calculate each term's TF-values (word).
2. Take the IDF values associated with these phrases.
3. Calculate the TF-IDF values for each term by dividing the TF by the IDF.
4. We obtain a dictionary that includes TF-IDF calculations for each term.

2. Support Vector Machines:

A supervised learning machine learning algorithm called the Support Vector Machine (SVM) can be applied to classification or regression problems. Most of the time, it is utilized in classification issues like text classification. In the SVM algorithm, each data point is represented as a point in n -dimensional space (where n is the number of features you have) with each feature's value being the value of a certain coordinate. Then, we accomplish classification by identifying the ideal hyper-plane that effectively distinguishes the two classes.

Support Vectors are just individual observation coordinates, and a hyper-plane is a particular SVM visualization. A frontier that best separates the two classes (hyper-plane/line) is the SVM classifier.

3. Grid Search:

Grid search is a tuning method that seeks to determine the ideal hyperparameter values. It is a thorough search that is done on a model's particular parameter values. An estimator is another name for the model.

A machine learning model with hyperparameters has several parameters that were not learned from the training set. The model's accuracy is controlled by these parameters. Before the model is trained, the caller of the model configures and provides the hyperparameters. Consider the Support Vector Classifier (SVC) model as well, which is employed to categorize data sets. The model needs a lot of different hyperparameters. The sk-learn library version of SVC can then be configured with a wide range of hyperparameters, some of which are widely used are:

- This regularisation parameter is C .
- The kernel parameter has several options, including linear, poly, rbf, sigmoid, precomputed, and callable.
- We can enter a custom degree to support the poly kernel parameter.
- Gamma: The rbf, poly, and sigmoid kernel parameters are all affected by this coefficient.

- The solver's maximum iteration count is indicated by the variable max Iter.
-

Chapter 6

Implementation

6.1 System Implementation

The general model for the sentiment analyzer, the architectural diagram,

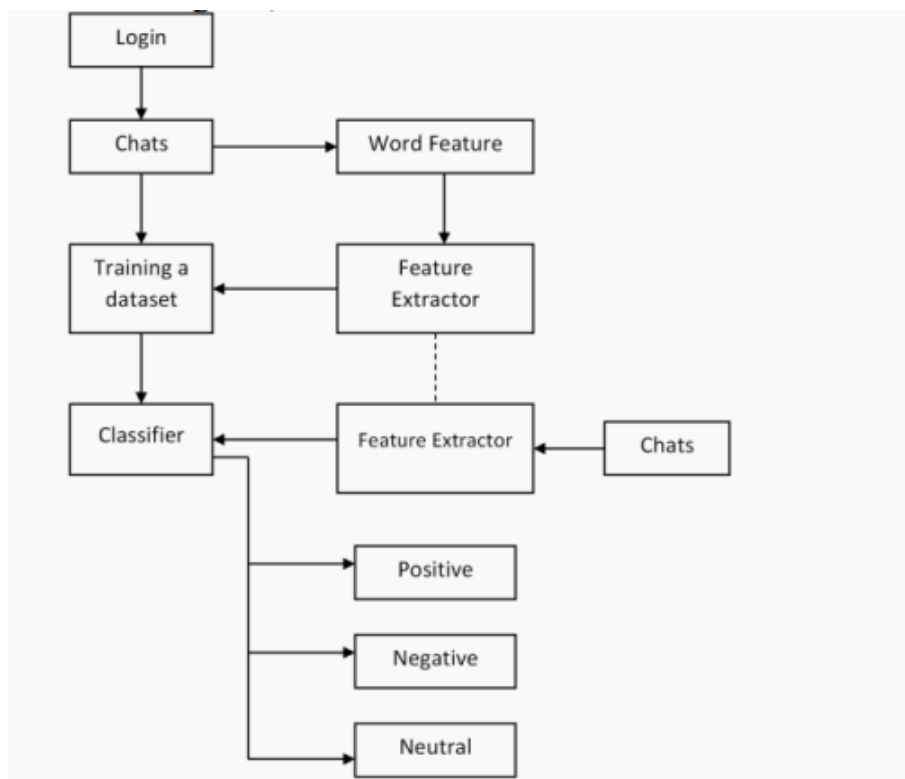


Figure 6.1: Architecture Diagram

The sentiment analysis architecture for phrases is shown in Figure 6.1 and is built

on the model developed by Kharde and Sonawane [21]. To classify something, a training set of sentences must be generated that are positive, negative, and neutral. It is applied a text steaming and stop words are eliminated in the training set and the input text. After that, the input text's polarity is established.

Data Flow:

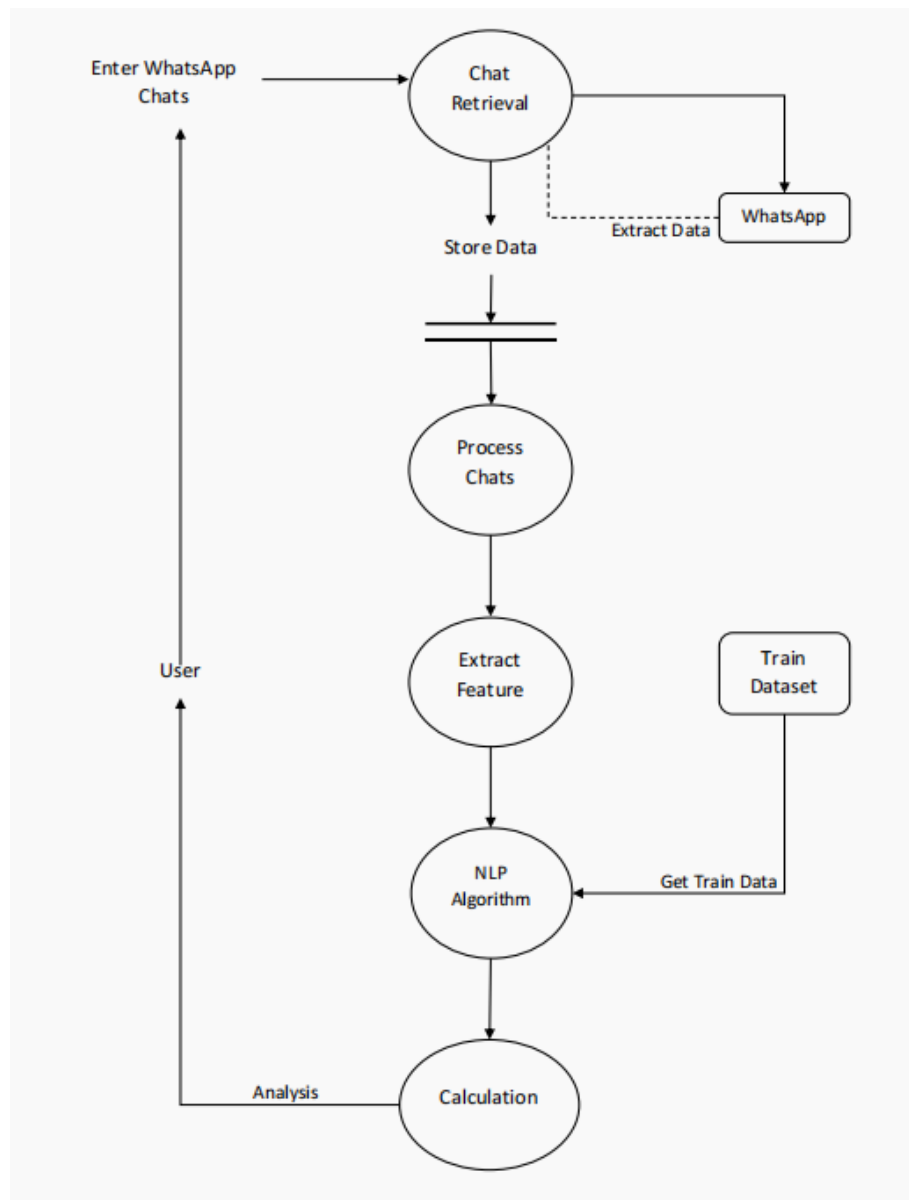


Figure 6.2: Data Flow Diagram

The data flow of the proposed experimentation model is shown in Figure 6.2. WhatsApp chats are the key source of data. In a previous section of the approach, it

is detailed how the data is gathered and processed. The dataset is uploaded by the user to the analyzer, where it is subsequently stored. The analyzer then begins processing the chats. The user receives an analysis report of the uploaded dataset after processing and applying the NLP algorithm, covered in an earlier portion of the technique.

C. Flow of Model:

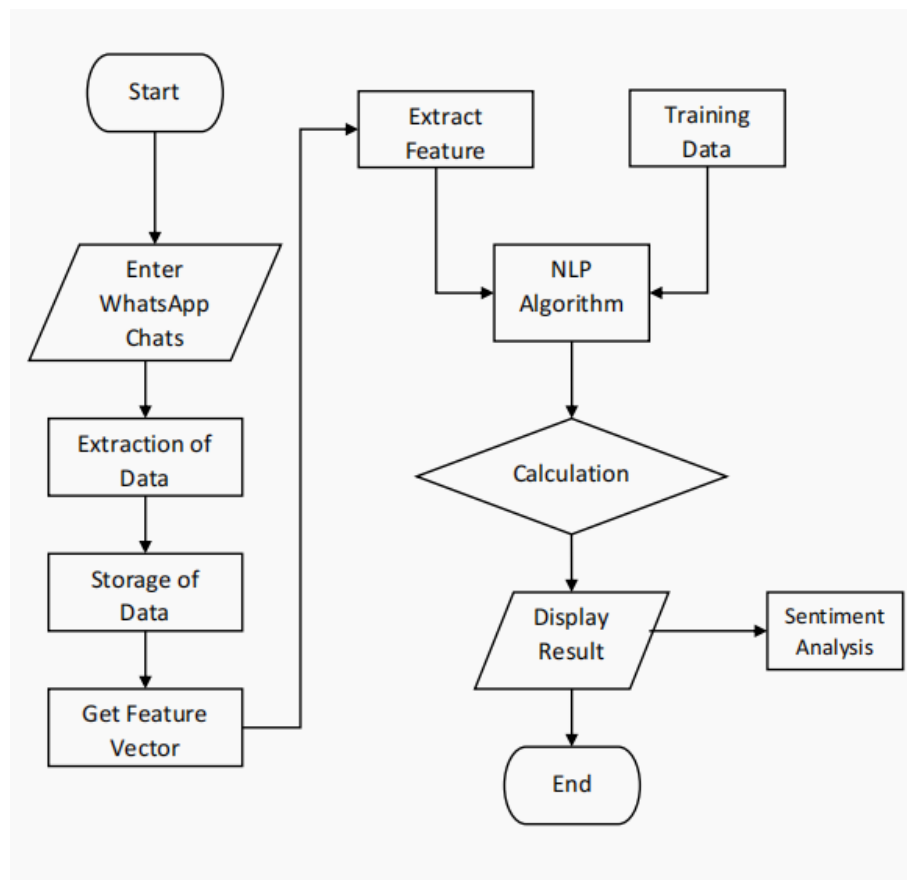


Figure 6.3: Flow Chart

First, WhatsApp chats from various users are gathered. These conversations contain both sender and receiver texts. The chats of the intended subject are separated and stored separately in the second step because only the sender's chats need to be analyzed.

Next, the dataset goes through pre-processing where from the chat non required things are omitted such that URLs, media files, and usernames. The spell gets corrected, non-English words are omitted and acronyms are expanded in this stage.

Next, we used a supervised learning approach for the classification of data, we used tf-idf vectorizer, SVM, and grid search, discussed in the previous section.

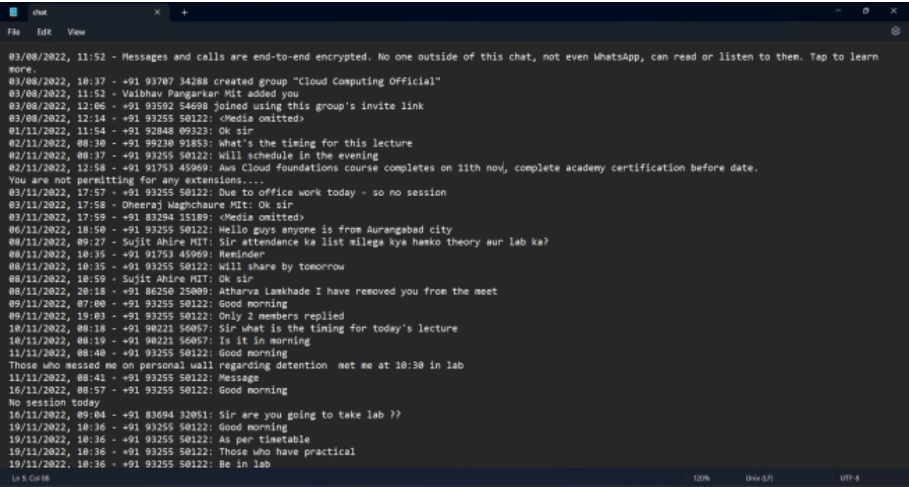
Following that, the chat file is run through a neutrality classifier. Here, the confusion matrix is calculated, we discussed the confusion matrix in an earlier section of the methodology. In this case, the sentence might be categorized as Positive, Negative, or Neutral. Calculate the Weightage Factor as 0.000-1.000 to lessen the impact of Neutral on our scores. The Neutrality Score in this case is 1.000. Hence, less neutral statements will have a greater impact on the score than less neutral sentences.

Now we get trained data with formatted order. The data with numerical calculations go through the testing phase where the sentiment of a sentence or whole chat is detected. Lastly, the detected sentiment is displayed to the user. The flow of the model is shown in the figure 6.3.

6.2 Experiment/Implementation Parameters

A. Data Gathering:

For research purposes, two types of data are used, raw data collected from users and preorganized data from the internet.



```
chat
File Edit View
03/08/2022, 11:52 - Messages and calls are end-to-end encrypted. No one outside of this chat, not even WhatsApp, can read or listen to them. Tap to learn more.
03/08/2022, 10:37 - +91 93787 34288 created group "Cloud Computing Official"
03/08/2022, 11:52 - Vaibhav Pangarkar MIT added you
03/08/2022, 12:00 - +91 93592 54698 joined using this group's invite link
03/08/2022, 12:14 - +91 93255 50122: <Media omitted>
01/11/2022, 11:54 - +91 92848 89323: Ok sir
02/11/2022, 08:38 - +91 99238 91853: What's the timing for this lecture
02/11/2022, 08:37 - +91 93255 50122: Will schedule in the evening
02/11/2022, 12:58 - +91 93753 45969: Aac Cloud foundations course completes on 11th nov, complete academy certification before date.
You are not permitting for any extensions....
03/11/2022, 17:57 - +91 93255 50122: Due to office work today - so no session
03/11/2022, 17:58 - Dheeraj Waghchaure MIT: Ok sir
03/11/2022, 17:59 - +91 83284 15189: <Media omitted>
06/11/2022, 18:50 - +91 93255 50122: Hello guys anyone is from Aurangabad city
08/11/2022, 09:27 - Sujit Ahire MIT: Sir attendance ka list milega kya haseko theory aur lab ka?
08/11/2022, 18:35 - +91 91753 45969: Reminder
08/11/2022, 18:35 - +91 93255 50122: Will share by tomorrow
08/11/2022, 18:59 - Sujit Ahire MIT: Ok sir
08/11/2022, 20:18 - +91 86250 25089: Atharva Laskhade I have removed you from the meet
09/11/2022, 07:00 - +91 93255 50122: Good morning
09/11/2022, 19:03 - +91 93255 50122: Only 2 members replied
10/11/2022, 08:18 - +91 98221 56957: Sir what is the timing for today's lecture
10/11/2022, 08:19 - +91 98221 56957: Is it in morning
11/11/2022, 08:40 - +91 93255 50122: Good morning
Those who messaged me on personal wall regarding detention met me at 10:30 in lab
11/11/2022, 08:41 - +91 93255 50122: Message
16/11/2022, 08:57 - +91 93255 50122: Good morning
No session today
16/11/2022, 09:04 - +91 83694 32051: Sir are you going to take lab ??
19/11/2022, 18:36 - +91 93255 50122: Good morning
19/11/2022, 18:36 - +91 93255 50122: As per timetable
19/11/2022, 18:36 - +91 93255 50122: Those who have practical
19/11/2022, 18:36 - +91 93255 50122: Be in lab
```

Figure 6.4: WhatsApp Chat data in .txt format

Particularly, group chat data rather than private conversation data is gathered for experimentation. Different types of ten groups are selected which are based on different subjects of engineering backgrounds between users of age groups of 18-25 years. The chats are extracted into the textual format without including the media files in the ‘.txt’ format using the ‘Export Chats’ feature of WhatsApp with the consent of concerned users. The group chat’s raw data is shown in Figure 6.4 after being extracted and saved in ‘.txt’ format.

The preorganized or open-source dataset from Kaggle [22] is utilized for experimenting. There are 12413 distinct chat statements in the sample. The Kaggle dataset can be shown in Figure 6.5.

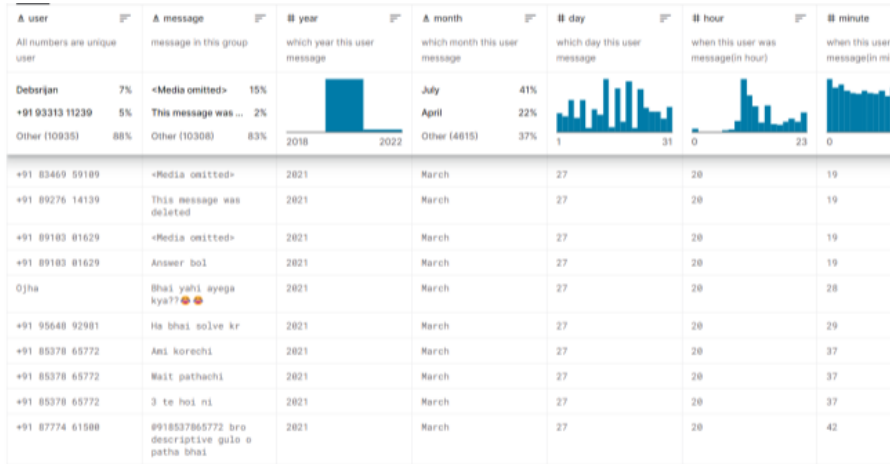


Figure 6.5: Kaggle Dataset (preorganized dataset)

B. K-Nearest Neighbor (KNN):

A classification model must then be provided with the pre-processed data for further processing. These models are built using a variety of classification techniques. The k-nearest neighbor (KNN) method was used in the experimentation to carry out the categorization.

A non-parametric supervised classification approach called k-nearest neighbor (KNN) is straightforward but frequently efficient. Due to its effectiveness in producing efficient results and its simplicity, the KNN classifier is regarded as the most used classifier for pattern recognition. The fields of pattern recognition, machine learning, text classification, data mining, object recognition, and many more all make exten-

sive use of them [23].

KNN was chosen since sentiment analysis is a binary classification and there are large datasets that can be processed. In the proposed experimentation, the classifier is trained using a manually created training set. Within the training set, there is an X:Y connection that shows the score of an opinion word as represented by x and the score of the word's positivity or negativity as represented by y [24]. KNN classifier receives as input a score of the opinion term associated with a feature in the chat.

C. Validity:

A dataset that had been pre-classified into the three emotions—positive, negative, and neutral—was needed to validate the experimentation model that had been suggested. After the collection of data, data is filtered into the required three emotions i.e., positive, negative, and neutral.

If the emotions corresponding to the top score matched the identified emotion in the data set, then that statement is labeled as successfully classified; if not, it is not. This is done to validate the model.

Several performance analysis criteria, such as precision, recall, and accuracy, which are covered in depth in the methodology section above, are taken into account to examine the performance of the proposed system.

$$\text{Accuracy} = (\text{No. of correctly classified chats}) / (\text{total number of chats})$$

The proposed system's performance is compared to the current system, which classifies chats as positive, negative, or neutral using an SVM classifier

6.3 Functional Implementation

Step 1: Importing required libraries such as numpy, pandas, matplotlib, seaborn, tensorflow, keras.

1. NumPy:

NumPy is a Python library used for scientific computing. It provides a multidimensional array object, various derived objects (such as masked arrays and matrices), and an assortment of routines for fast operations on arrays, including mathematical, logical, shape manipulation, sorting, selecting, I/O, discrete Fourier transforms, basic linear algebra, basic statistical operations, and much more.

2. Pandas:

Pandas is a Python library used for data manipulation and analysis. It provides data structures for efficiently storing and manipulating large and complex data sets. The two main data structures in Pandas are Series and DataFrame. A Series is a one-dimensional array-like object that can hold any data type. A DataFrame is a two-dimensional table of data with rows and columns, similar to a spreadsheet or a SQL table.

3. emoji:

The emoji library provides a set of functions and methods that allow us to work with emojis in our Python programs. It offers features such as emoji character conversion, emoji extraction, and emoji rendering. By installing and importing the emoji library, we can easily incorporate emojis into our text, print them to the console, or perform various operations on them.

4.Dateparser:

Dateparser is a powerful library that simplifies the parsing and manipulation of dates and times in Python. It provides a convenient way to parse date strings in various formats and convert them into Python datetime objects.

5. Jovian:

Jovian is a Python library and platform that simplifies the process of sharing and collaborating on Jupyter notebooks. It provides a set of functions and utilities to

capture, save, and upload Jupyter notebooks to the Jovian platform, where they can be shared with others.

6. TextBlob:

The TextBlob library in Python is used for natural language processing (NLP) tasks, such as text processing, sentiment analysis, part-of-speech tagging, noun phrase extraction, language translation, and more. By importing TextBlob, we gain access to its powerful NLP capabilities, allowing us to analyze and manipulate text data effectively.

6.4 Data Description

The key aspects of data description in the analysis of data are :

1.Structure and Format:

Determine the structure of the WhatsApp chat data. It may include information such as message timestamp, sender, and the actual message content. Identify the format of the data file, such as CSV, text file, or any other format, and ensure it is compatible with the analysis tools and libraries you plan to use.

2.Data Size and Volume:

Assess the size of the WhatsApp chat data. How many chat messages are included in the dataset? This information helps understand the scale of the analysis and potential computational requirements.

3.Message Content:

Explore the content of the chat messages. Analyze the language used, including any special characters, emojis, URLs, or multimedia elements. Consider the language specifics, including the use of slang, abbreviations, or local expressions, which might require additional preprocessing steps.

4.Metadata and Contextual Information:

Consider the metadata associated with the chat data, such as sender information, message timestamps, and any other relevant contextual information. Examine the distribution of messages across different senders, as it may influence the sentiment analysis and the interpretation of results.

6.5 Output

The output of a WhatsApp chat sentiment analysis using natural language processing (NLP) techniques can vary based on the specific goals and requirements of the analysis. Here are some common outputs you can expect from a sentiment analysis on WhatsApp chat data:

1.Sentiment Scores:

For each chat message, sentiment analysis assigns a sentiment score indicating the sentiment polarity. The sentiment score could be a numeric value indicating positive, negative, or neutral sentiment. For example, sentiment scores could range from -1 (negative sentiment) to 1 (positive sentiment) or 0 (neutral sentiment).

2.Sentiment Category:

Based on the sentiment scores, each chat message can be categorized into sentiment categories such as positive, negative, or neutral. The sentiment category indicates the overall sentiment expressed in the message.

3.Sentiment Distribution: The sentiment analysis output may include the distribution of sentiment categories across the WhatsApp chat data. It could provide insights into the proportion of positive, negative, and neutral messages, helping to understand the overall sentiment of the chat.

4.Visualizations:

Data visualizations are often generated to provide a graphical representation of the sentiment analysis results. These visualizations can include bar charts, pie charts,

or histograms showing the distribution of sentiment categories. Time-series graphs may also be used to track sentiment changes over time.

5.Insights and Summary:

The output of sentiment analysis may include insights and summaries derived from the analysis. These insights could highlight significant sentiment trends, specific events impacting sentiment, or patterns observed in the chat data. The analysis output aims to provide actionable insights and a comprehensive understanding of the sentiment expressed in the WhatsApp chat.

Chapter 7

Results and Discussion

7.1 Cleaning of data

First we take the WhatsApp chat text file and extract the date/time, author, and message content for each message.

```
In [25]: parsedChat
        'Joanna',
        'Happy new year. The first when Mtn is civilized'],
['01/01/2018, 00:09', 'TJ Musiitwa', 'Civilized how?'],
['01/01/2018, 00:10', 'Joanna', 'We remain in touch'],
['01/01/2018, 00:11', 'Ben', 'Happy New Year people ... Jah Bless'],
['01/01/2018, 00:12', 'Anthony Busulwa', 'Happy New year everyone.'],
['01/01/2018, 00:15', '+1 (647) 704-2525', 'Happy new year!!!'],
['01/01/2018, 00:39', 'Patsy', 'happy new year loves'],
['01/01/2018, 01:00', 'Irene Nakalembe', 'Happy new year fam'],
['01/01/2018, 01:21', 'Micheal', 'A HAPpy new year to.you all'],
['01/01/2018, 01:32', 'Lena', 'Happy New year!'],
['01/01/2018, 01:47', 'Olivia Namulindwa', '🎉Happy New Year!!!!🎉'],
['01/01/2018, 09:46', 'Lawrence Kibuuka', 'Wishing you a prosperous and a'],
['01/01/2018, 10:36',
'Joanna',
"Happy New Year! Wish each and everyone of you God's abundant blessings. May good things overwhelm you."],
['01/01/2018, 13:16', 'Anna Christine', 'Thank you Jo; and to you too'],
['01/01/2018, 13:17',
'Anna Christine',
'Happy new year fam! May all your hearts desires be realised!'],

In [26]: df=pd.DataFrame(parsedChat,columns=['DateTime','Author','Message'])

In [27]: df.head()

Out[27]:
```

	DateTime	Author	Message
0	None	None	26/12/2017, 12:28 - Pauline: Lena and Pauline ...
1	26/12/2017, 15:58	Joanna	Testing 3-5 outdoor venues by night today and ...
2	26/12/2017, 22:45	Joanna	<Media omitted>
3	28/12/2017, 21:14	Micheal	It will happen to you
4	28/12/2017, 21:39	+256 792 754083	See yourself

Figure 7.1: filtering all the messages

then we parse the chat in the DateTime,Author,Message format.In the figure 7.2 the output of the parsed chat is represented where the authour date and messages of the group are epreated by each other in tabular format.

```
In [26]: df=pd.DataFrame(parsedChat,columns=['DateTime','Author','Message'])
```

```
In [27]: df.head()
```

```
Out[27]:
```

	DateTime	Author	Message
0	None	None	26/12/2017, 12:28 - Pauline: Lena and Pauline ...
1	26/12/2017, 15:58	Joanna	Testing 3-5 outdoor venues by night today and ...
2	26/12/2017, 22:45	Joanna	<Media omitted>
3	28/12/2017, 21:14	Micheal	It will happen to you
4	28/12/2017, 21:39	+256 792 754083	See yourself

Figure 7.2: parsed chats

7.2 Representing text in charts

we analyze the chats and then transform them in a graphical representation format.

```
Out[32]: Text(0.5, 1.0, 'Most Media Items sent per Author')
```

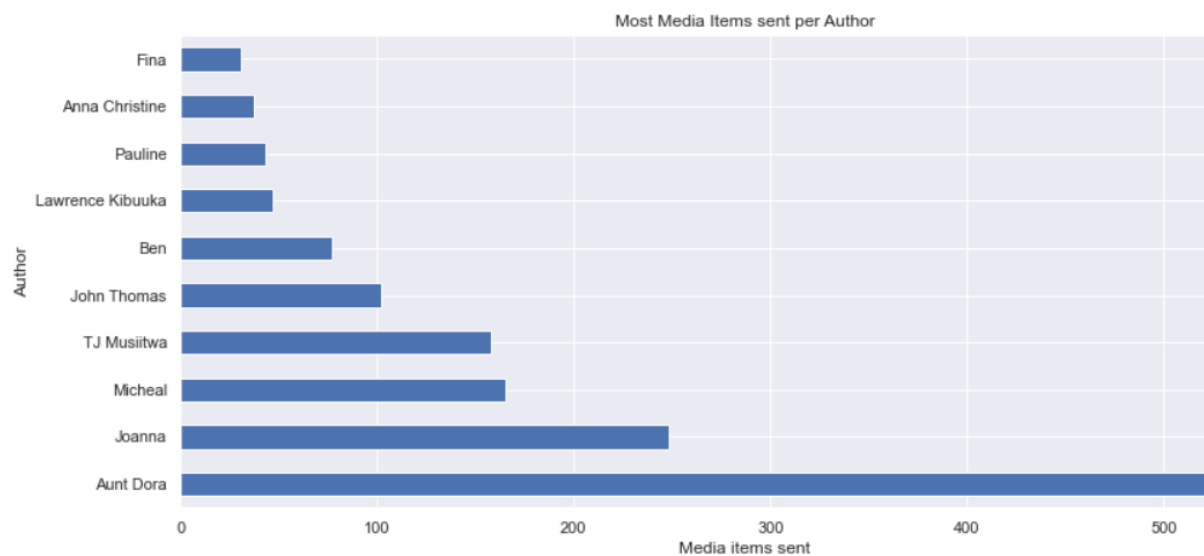


Figure 7.3: most media sent by the person.

Here figure 7.3 shows the top 10 persons in the group who have sent the data such as files, videos and documents in the chat.

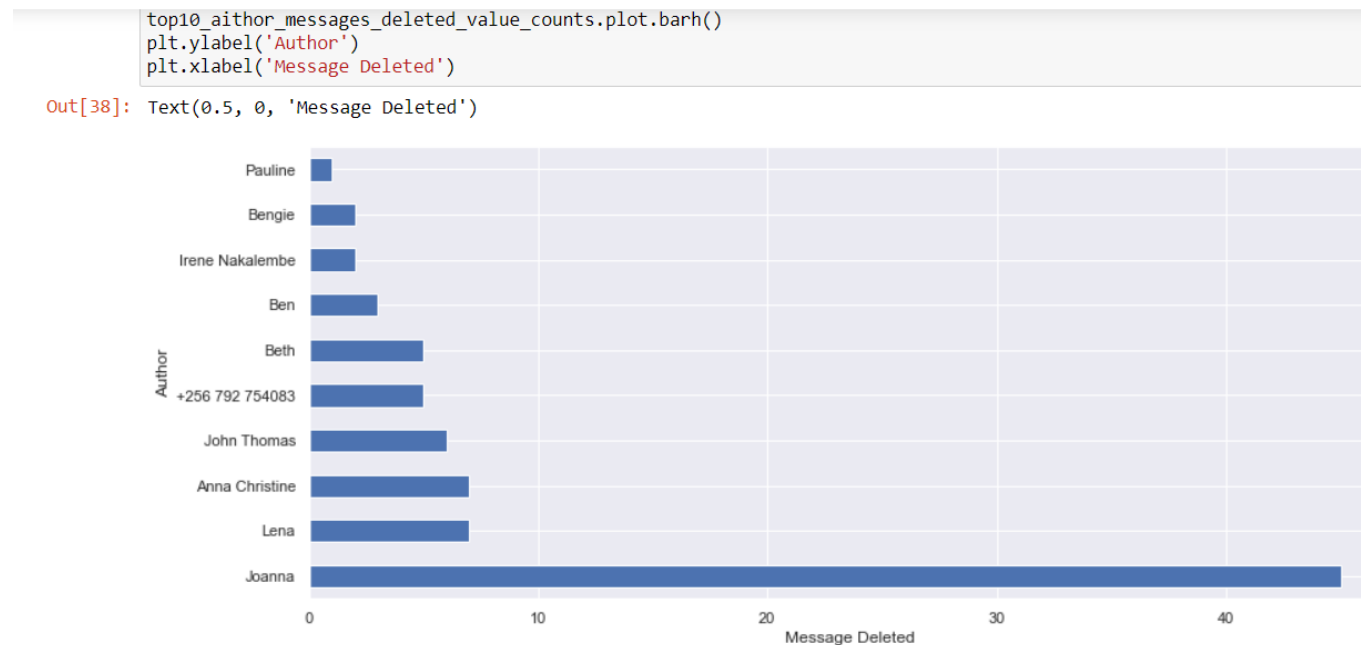


Figure 7.4: most deleted chats by the person.

Figure 7.4 list outs the group participants who have deleted the messages for everyone in the group chat.

Figure 7.5 list outs the group participants whoares most active and have sent the most messages in the chat.

Figure 7.6 shows the output of the highest positive sentiment polarity score which is selected randomly and then message which have the high polarity it is printed in the output.

Figure 7.7 shows the output of the most neutral sentiment polarity score and most negative neutral sentiment polarity score which is selected randomly.

```
Out[57]: Text(0.5, 1.0, 'The top 10 Most Talkative Persons')
```

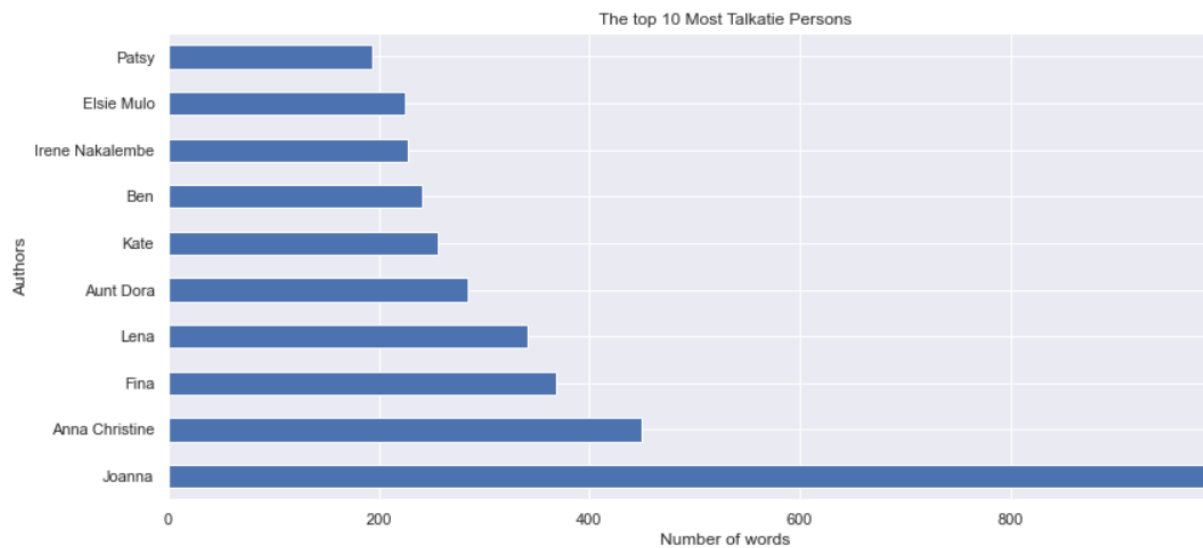


Figure 7.5: highest chats by the person.

Sentiment Analysis

```
In [94]: df['Polarity']=df['Message'].map(lambda text: TextBlob(text).sentiment.polarity)
```

```
In [ ]:
```

Randomly select 5 reviews with the highest positive polarity score

```
In [95]: print('5 random reviews with the highest positive sentiment polarity: \n ')
c1=df.loc[df['Polarity']==1,['Message']].sample(5).values
for c in c1:
    print(c[0])
```

5 random reviews with the highest positive sentiment polarity:

```
Happy Birthday Beth!!! 🎉
Happy birthday Beth!
Happy birthday Swelii!
HAPPY BIRTHDAY IRENE!!! ❤️🥳
happy birthday JNDK! May this be the best year yet, and to plenty more 🥰
```

Figure 7.6: highest positive polarity.


```
In [96]: print('5 reviews with the most neutral sentiment(zero) polarity: \n')
cl=df.loc[df['Polarity']==0,['Message']].sample(5).values
for c in cl:
    print(c[0])
```

5 reviews with the most neutral sentiment(zero) polarity:

We dont dance lingala.. we r not Congolese 🤔🤔🤔
 We shall be lost some of us...
 Details
 Some jokes should not be shared... these are fickle times..
 Almost there

```
In [97]: print('5 reviews with the most negative polarity: \n')
cl=df.loc[df['Polarity']==-0.50,['Message']].sample(5).values
for c in cl:
    print(c[0])
```

5 reviews with the most negative polarity:

Tuesday is so fake
 About 70 survivors and 12 bodies retrieved. Sad
 Oh sorry. May she rest in peace
 Sorry may
 Sorry I'm just opening my eyes

The distribution of review sentiment polarity score

Figure 7.7: lowest polarity and negative polarity.

```
In [106]: # feature extraction
from sklearn.feature_extraction.text import CountVectorizer
bow_vectorizer = CountVectorizer(max_df=0.90, min_df=2, max_features=1000, stop_words='english')
bow = bow_vectorizer.fit_transform(df['Message'])
```

In []:

```
In [107]: bow.toarray()
```

```
Out[107]: array([[0, 0, 0, ..., 0, 0, 0],
 [0, 0, 0, ..., 0, 0, 0],
 [0, 0, 0, ..., 0, 0, 0],
 ...,
 [0, 0, 0, ..., 0, 0, 0],
 [0, 0, 0, ..., 0, 0, 0],
 [0, 0, 0, ..., 0, 0, 0]], dtype=int64)
```

```
In [108]: from sklearn.model_selection import train_test_split
```

```
In [109]: x_train, x_test, y_train, y_test = train_test_split(bow, df['Message'], random_state=42, test_size=0.25)
```

Figure 7.8: lowest polarity and negative polarity.

Figure 7.8 extracts the data and converted into vector form so that the date positivity or negativity is calculated by matrix

```
Model Training

In [106]: from sklearn.linear_model import LogisticRegression
          from sklearn.metrics import f1_score, accuracy_score

In [107]: # training
          model = LogisticRegression()
          model.fit(x_train, y_train)

Out[107]: LogisticRegression()

In [110]: from sklearn.metrics import f1_score

          # testing
          pred = model.predict(x_test)
          average_f1_score = f1_score(y_test, pred, average='weighted')
          print(f"Average F1 score: {average_f1_score:.4f}")

          # Store the average F1 score in a variable for later use
          f1_score=average_f1_score

          Average F1 score: 0.0176

In [111]: accuracy_score(y_test,pred)

Out[111]: 0.04317656129529684

In [112]: from sklearn.metrics import f1_score

          # Assuming y_test and pred are your target and predicted values, respectively
          binary_f1_score = f1_score(y_test, pred, average='weighted')
          print(binary_f1_score)
```

Figure 7.9: lowest polarity and negative polarity.

Figure 7.9 shows the model training of the data where Logistic regression is used which is a statistical model used for binary classification, where the goal is to predict the probability of an instance belonging to a particular class. Here we get the average F1 score and accuracy score of the chats.

Figure 7.10 we get the accuracy score in array and feature extraction score in true or false

The graph in the figure 7.11 depicts the the number of messages comes on the group according to the phases of days. It has been observed that most of the members delivered messages around 10 am at day time and around 9 - 10 pm in the night time.

```

In [112]: from sklearn.metrics import f1_score

          # Assuming y_test and pred are your target and predicted values, respectively
          binary_f1_score = f1_score(y_test, pred, average='weighted')
          print(binary_f1_score)

0.017576525090408444

In [125]: accuracy_score(y_test, pred)

Out[125]: 0.6

In [128]: pred_prob[0]

Out[128]: array([0.00060966, 0.000293 , 0.00018528, ..., 0.00025154, 0.00029218,
                  0.000293  ])

In [126]: pred_prob[0][1] >= 0.3

Out[126]: False

In [ ]:

```

Figure 7.10: The sentiment analysis of chat.

```

In [65]: df.groupby('Time')['Message'].count().plot()
          plt.ylabel('Number of messages')

Out[65]: Text(0, 0.5, 'Number of messages')

```

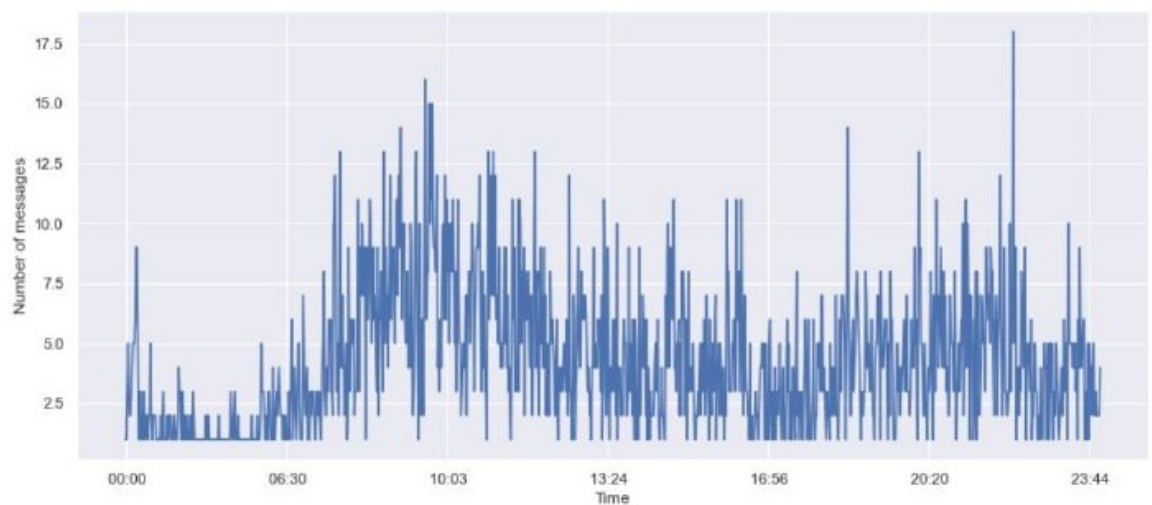


Figure 7.11: Number of messages in the day

```
In [66]: df.groupby('Hour')['Message'].count().plot()
plt.ylabel('Number of messages')
Out[66]: Text(0, 0.5, 'Number of messages')
```

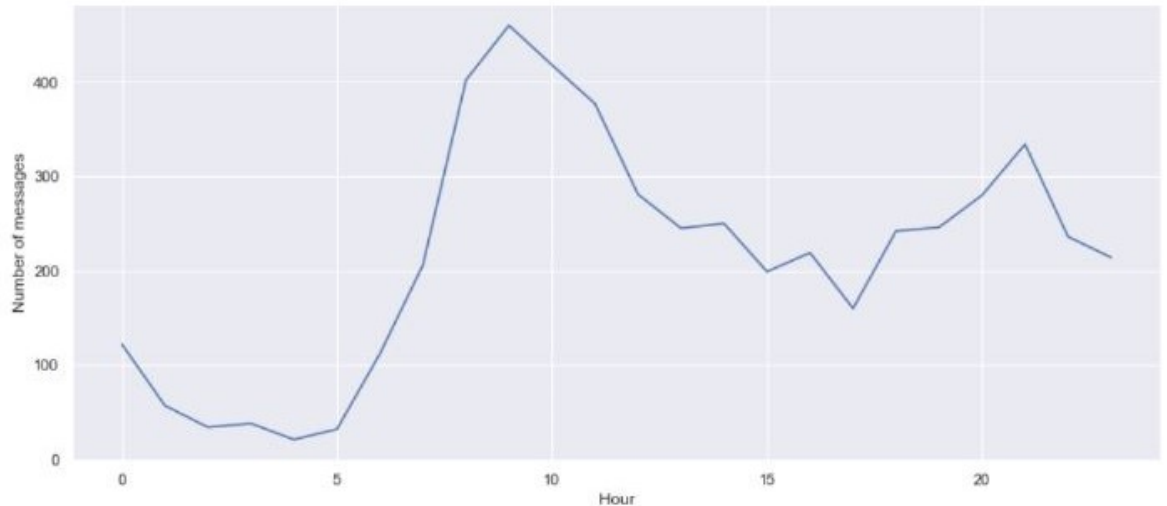


Figure 7.12: Number of messages in the day

The graph shown in 7.12 depicts the hourly ratio of messages incoming in the group. It has been observed that that highest messages incoming during the morning period that night. And the most messages comes during around 10 am.

```
In [74]: print('Total Letter Count in the group:'+str(df['Letter Count'].sum()))
Total Letter Count in the group:210965

In [75]: print('Total Word Count in the group:'+str(df['Word Count'].sum()))
Total Word Count in the group:38057

What is the most common number of words in a message?

In [76]: plt.figure(figsize=(20,4))
word_count_value_counts=df['Word Count'].value_counts()
top30_word_count=word_count_value_counts.head(30)
top30_word_count.plot.bar()
plt.xlabel('Word Count')
plt.ylabel('Frequency')
Out[76]: Text(0, 0.5, 'Frequency')
```

Figure 7.13: Message and word count

Figure 7.13 gives us an idea about the message and word count.

Install required packages

In [3]: `pip install emoji`

Requirement already satisfied: emoji in c:\users\admin\anaconda3\new folder\lib\site-packages (2.2.0)
Note: you may need to restart the kernel to use updated packages.

In [4]: `pip install dateparser`

Requirement already satisfied: dateparser in c:\users\admin\anaconda3\new folder\lib\site-packages (1.1.8)
Requirement already satisfied: python-dateutil in c:\users\admin\anaconda3\new folder\lib\site-packages (from dateparser) (2.8.2)
Requirement already satisfied: tzlocal in c:\users\admin\anaconda3\new folder\lib\site-packages (from dateparser) (5.0.1)
Requirement already satisfied: regex!=2019.02.19,!=2021.8.27 in c:\users\admin\anaconda3\new folder\lib\site-packages (from dateparser) (2021.8.3)
Requirement already satisfied: pytz in c:\users\admin\anaconda3\new folder\lib\site-packages (from dateparser) (2021.8.3)
Requirement already satisfied: six>=1.5 in c:\users\admin\anaconda3\new folder\lib\site-packages (from python-dateutil) (1.16.0)
Requirement already satisfied: tzdata in c:\users\admin\anaconda3\new folder\lib\site-packages (from tzlocal->dateparser) (2021.8.3)
Note: you may need to restart the kernel to use updated packages.

In [5]: `pip install jovian`

Requirement already satisfied: jovian in c:\users\admin\anaconda3\new folder\lib\site-packages (0.2.47)
Requirement already satisfied: uuid in c:\users\admin\anaconda3\new folder\lib\site-packages (from jovian) (1.30)
Requirement already satisfied: click in c:\users\admin\anaconda3\new folder\lib\site-packages (from jovian) (8.0.3)
Requirement already satisfied: requests in c:\users\admin\anaconda3\new folder\lib\site-packages (from jovian) (2.28.1)
Requirement already satisfied: pyyaml in c:\users\admin\anaconda3\new folder\lib\site-packages (from jovian) (6.0.1)
Requirement already satisfied: colorama in c:\users\admin\anaconda3\new folder\lib\site-packages (from click->jovian) (0.4.4)
Requirement already satisfied: urllib3<1.27,>=1.21.1 in c:\users\admin\anaconda3\new folder\lib\site-packages (from requests->jovian) (1.26.7)
Requirement already satisfied: certifi>=2017.4.17 in c:\users\admin\anaconda3\new folder\lib\site-packages (from requests->jovian) (2021.10.8)

Import all required packages for the project

In [6]: `import re`

In [7]: `import matplotlib.pyplot as plot`

In [8]: `import pandas as pd`

In [9]: `import numpy as np`

In [10]: `import seaborn as sns`

In [11]: `import calendar`

In [12]: `import datetime as dt`

In [13]: `from wordcloud import WordCloud, STOPWORDS`

In [14]: `import emoji`

In [15]: `from sklearn.feature_extraction.text import CountVectorizer`

In [16]: `import random`

In [17]: `from textblob import TextBlob`

In [18]: `from sklearn.decomposition import LatentDirichletAllocation`

```
In [20]: from sklearn.decomposition import NMF
```

```
In [21]: sns.set(rc={'figure.figsize':(14,6)})
```

Data Cleaning

```
In [22]: def startswithDate(s):
          pattern='^([0-2][0-9]|(3[0-1])(\|)((0[0-9]|(1[0-2]))(\|)\d{2}\d{4}), ([0-9][0-9]):([0-9][0-9]) - '
          result=re.match(pattern,s)
          if result:
              return True
          return False
```

```
In [23]: def startswithAuthor(s):
          patterns=['([\w]+):', # First Name
                   '([\w]+[\s]+[\w]+):', # First Name + Last Name
                   '([\w]+[\s]+[\w]+[\s]+[\w]+):', # First Name+ Middle Name + Last Name
                   '^(\+\d{1,2}\s)?\((?\d{3}\s) ((\(\d{3}\s) ?)(\d{3}-))?\d{3}-\d{4}:',
                   '([\+]\d{2} \d{4} \d{6}):',
                   '([\+]\d{3} \d{3} \d{6})'
                  ]
          pattern='^'+ '|'.join(patterns)
          result=re.match(pattern,s)
          if result:
              return True
          return False
```

```
In [24]: def getDataPoint(line):
          splitLine=line.split(' - ')
          dateTime=splitLine[0] # '18/06/2021, 22:47'
          message=' '.join(splitLine[1:])
          if startswithAuthor(message): #True
              splitmessage=message.split(': ')
              splitmessage=splitmessage[1:]
```

```
In [24]: def getDataPoint(line):
          splitLine=line.split(' - ')
          dateTime=splitLine[0] # '18/06/2021, 22:47'
          message=' '.join(splitLine[1:])
          if startswithAuthor(message): #True
              splitmessage=message.split(': ')
              author=splitmessage[0]
              message=" ".join(splitmessage[1:])
          else:
              author=None
          return dateTime,author,message
```

```
In [25]: parsedChat=[]
          convoPath='./whatsappChat.txt'
          with open(convoPath,encoding='utf-8') as fp:
              fp.readline
              messagebuffer=[]
              dateTime,author=None,None

              while True:
                  line=fp.readline()
                  if not line:
                      break
                  line=line.strip()
                  if startswithDate(line):
                      if len(messagebuffer)>0:
                          parsedChat.append([dateTime,author," ".join(messagebuffer)])
                          messagebuffer.clear()
                      dateTime,author,message=getDataPoint(line)
                      messagebuffer.append(message)
                  else:
                      messagebuffer.append(line)
```



```
Out[30]:
```

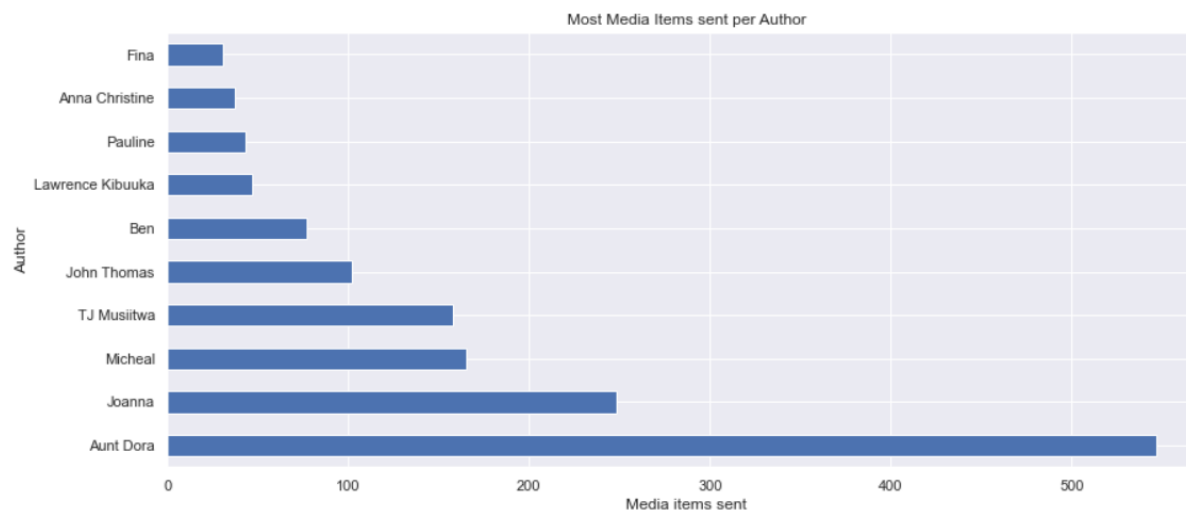
	DateTime	Author	Message
count	7333	6882	7334
unique	5662	35	4929
top	15/12/2019, 08:10	Joanna	<Media omitted>
freq	26	1300	1604

```
In [31]: # No. of images, images are represented by <Media omitted>
media=df[df['Message']=='<Media omitted>']
```

```
In [32]: author_media_messages_value_counts=media['Author'].value_counts()
top10_author_media_messages_value_counts=author_media_messages_value_counts.head(10)
```

```
In [33]: top10_author_media_messages_value_counts.plot.barh()
plot.ylabel('Author')
plot.xlabel('Media items sent')
plot.title('Most Media Items sent per Author')
```

```
Out[33]: Text(0.5, 1.0, 'Most Media Items sent per Author')
```



```
In [34]: message_deleted=df[df['Message']=='This message was deleted']
```

```
In [35]: message_deleted
```

```
Out[30]:
```

	DateTime	Author	Message
count	7333	6882	7334
unique	5662	35	4929
top	15/12/2019, 08:10	Joanna	<Media omitted>
freq	26	1300	1604

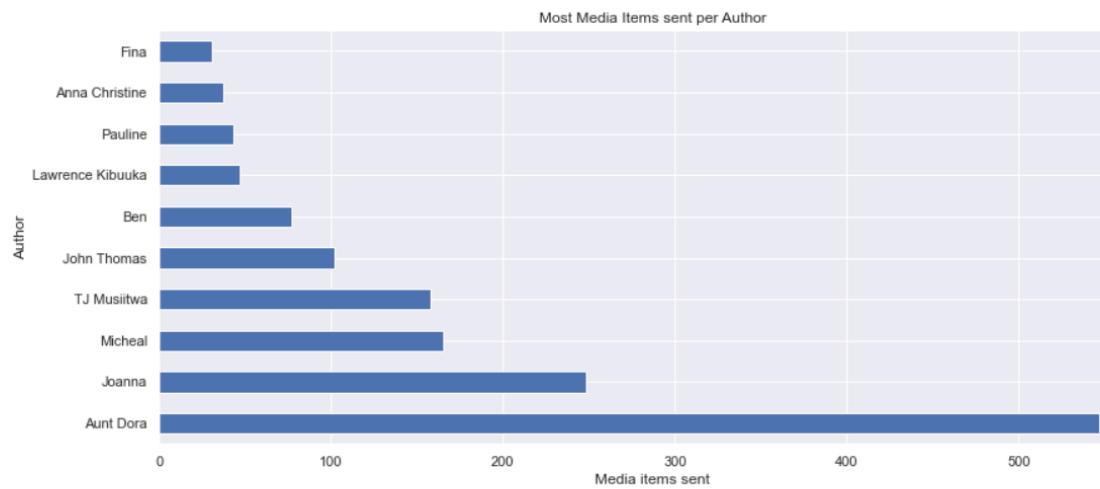
```
In [31]: # No. of images, images are represented by <Media omitted>
media=df[df['Message']=='<Media omitted>']
```

```
In [32]: author_media_messages_value_counts=media['Author'].value_counts()
top10_author_media_messages_value_counts=author_media_messages_value_counts.head(10)
```

```
In [33]: top10_author_media_messages_value_counts.plot.barh()
plot.ylabel('Author')
plot.xlabel('Media items sent')
plot.title('Most Media Items sent per Author')
```



```
Out[33]: Text(0.5, 1.0, 'Most Media Items sent per Author')
```



```
In [34]: message_deleted=df[df['Message']=='This message was deleted']
```

```
In [35]: message_deleted
```

```
Out[35]:
```

	DateTime	Author	Message
75	04/01/2018, 20:55	Lena	This message was deleted
600	24/01/2018, 08:46	Joanna	This message was deleted
601	24/01/2018, 08:46	Joanna	This message was deleted
679	01/02/2018, 12:03	Ben	This message was deleted
897	09/02/2018, 15:30	John Thomas	This message was deleted
...
7104	18/12/2019, 15:54	John Thomas	This message was deleted
7149	20/12/2019, 12:12	Joanna	This message was deleted
7150	20/12/2019, 12:13	Joanna	This message was deleted
7153	21/12/2019, 08:23	Joanna	This message was deleted
7303	23/12/2019, 21:11	Joanna	This message was deleted

93 rows × 3 columns

```
In [36]: author_messages_deleted_value_counts=message_deleted['Author'].value_counts()
```

```
In [37]: author_messages_deleted_value_counts
```

```
Out[37]: Joanna      45
Lena                7
Anna Christine      7
John Thomas         6
+256 792 754083      5
Beth                5
```

```

Ben          3
Irene Nakalembe  2
Bengie       2
Pauline      1
Adela Mulo   1
Junior       1
Patsy        1
Elsie Mulo   1
Kate         1
Kirijja Godfrey  1
Micheal      1
+44 7525 475883  1
Livia Livie   1
+256 776 584356  1
Name: Author, dtype: int64

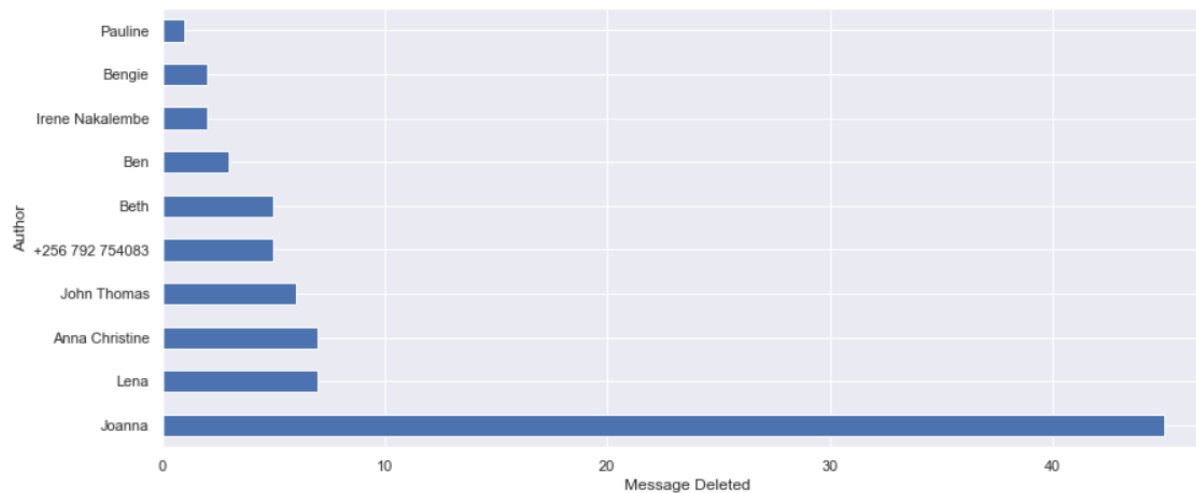
```

```
In [38]: top10_athor_messages_deleted_value_counts=author_messages_deleted_value_counts.head(10)
```

```
In [39]: import matplotlib.pyplot as plt
top10_athor_messages_deleted_value_counts.plot.barh()
plt.ylabel('Author')
plt.xlabel('Message Deleted')
```

```
Out[39]: Text(0.5, 0, 'Message Deleted')
```

```
Out[39]: Text(0.5, 0, 'Message Deleted')
```



```
In [40]: #Number of Group Notifications
grp_notif=df[df['Author']=='grp_notif']
```

```
In [41]: grp_notif.shape
```

```
Out[41]: (0, 3)
```

```
In [42]: # Drop the 'media omitted' messages, group notifications and deleted messages
df.drop(media.index, inplace=True)
df.drop(grp_notif.index, inplace=True)
df.drop(message_deleted.index, inplace=True)
```

```
In [43]: # Find the null values
df.isnull().sum()
```

```
Out[43]: DateTime      1
Author      452
Message      0
dtype: int64
```

```
In [44]: # Drop empty rows
df=df.dropna()
```

```
In [45]: df.shape
```

```
Out[45]: (5185, 3)
```

```
In [46]: df.reset_index(inplace=True, drop=True)
```

Add the dateTime object from dateTime column

```
In [47]: df['dateTime']=pd.to_datetime(df['DateTime'], infer_datetime_format=True)
```

```
In [48]: df['Day of Week']=pd.Series(pd.Categorical(df['dateTime'].dt.day_name(), categories=list(calendar.day_name)))
```

```
In [49]: df['Day of Week']
```

```
In [49]: df['Day of Week']
```

```
Out[49]: 0      Tuesday
1      Thursday
2      Thursday
3      Thursday
4      Friday
...
5180   Wednesday
5181   Wednesday
5182   Wednesday
5183   Wednesday
5184   Wednesday
Name: Day of Week, Length: 5185, dtype: category
Categories (7, object): ['Monday', 'Tuesday', 'Wednesday', 'Thursday', 'Friday', 'Saturday', 'Sunday']
```

```
In [50]: df['Hour']=pd.Series(pd.Categorical(df['dateTime'].dt.hour))
```

```
In [51]: df['Hour']
```

```
Out[51]: 0      15
1      21
2      21
3      23
4      15
...
5180   10
5181   12
5182   13
5183   13
5184   18
Name: Hour, Length: 5185, dtype: category
Categories (24, int64): [0, 1, 2, 3, ..., 20, 21, 22, 23]
```

```
In [52]: df=df.set_index('dateTime')
```

```
In [53]: df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
DatetimeIndex: 5185 entries, 2017-12-26 15:58:00 to 2019-12-25 18:47:00
Data columns (total 5 columns):
#   Column          Non-Null Count  Dtype
---  ---
0   DateTime         5185 non-null   object
1   Author           5185 non-null   object
2   Message          5185 non-null   object
3   Day of Week      5185 non-null   category
4   Hour             5185 non-null   category
dtypes: category(2), object(3)
memory usage: 173.2+ KB
```

```
In [54]: df.head()
```

```
Out[54]:
```

	DateTime	Author	Message	Day of Week	Hour
	dateTime				
2017-12-26 15:58:00	26/12/2017, 15:58	Joanna	Testing 3-5 outdoor venues by night today and ...	Tuesday	15
2017-12-28 21:14:00	28/12/2017, 21:14	Micheal	It will happen to you	Thursday	21
2017-12-28 21:39:00	28/12/2017, 21:39	+256 792 754083	See yourself	Thursday	21
2017-12-28 23:12:00	28/12/2017, 23:12	Beth	23/12/2017, 23:42:10 +256 701 839948 BEWARE.....	Thursday	23
2017-12-29 15:36:00	29/12/2017, 15:36	Micheal	Tina ng'okaye	Friday	15

Data Exploration

```
In [55]: df.describe()
```

```
Out[55]:
```

	DateTime	Author	Message	Day of Week	Hour
count	5185	5185	5185	5185	5185
unique	4309	35	4595	7	24
top	02/11/2018, 07:49	Joanna	Amen	Sunday	9
freq	8	1007	35	983	460

```
In [ ]:
```

```
In [56]: author_value_counts=df['Author'].value_counts()
top10_talkers=author_value_counts.head(10)
```

```
In [57]: top10_talkers
```

```
Out[57]: Joanna      1007
Anna Christine    449
Fina              368
Lena              341
Aunt Dora         285
Kate              256
Ben               241
Irene Nakalembe  227
Elsie Mulo        225
Patsy             194
Name: Author, dtype: int64
```

```
In [89]: df['Hour']
```

```
Out[89]: DateTime
2017-12-26 15:58:00    15
2017-12-28 21:14:00    21
2017-12-28 21:39:00    21
2017-12-28 23:12:00    23
2017-12-29 15:36:00    15
..
2019-12-25 10:48:00    10
2019-12-25 12:40:00    12
2019-12-25 13:20:00    13
2019-12-25 13:20:00    13
2019-12-25 18:47:00    18
Name: Hour, Length: 5185, dtype: category
Categories (24, int64): [0, 1, 2, 3, ..., 20, 21, 22, 23]
```

2019-12-25 10:48:00	25/12/2019, 10:48	Anna Christine	Wow....you should have printed this card earli...	Wednesday	10	25/12/2019	10:48
2019-12-25 12:40:00	25/12/2019, 12:40	Fina	Beautiful.	Wednesday	12	25/12/2019	12:40
2019-12-25 13:20:00	25/12/2019, 13:20	Joanna	Thank you 🥰💝❤️	Wednesday	13	25/12/2019	13:20
2019-12-25 13:20:00	25/12/2019, 13:20	Joanna	🙄🙄	Wednesday	13	25/12/2019	13:20
2019-12-25 18:47:00	25/12/2019, 18:47	Olivia Namulindwa	Merry Christmas Family 🎄🎁❤️	Wednesday	18	25/12/2019	18:47

5185 rows x 9 columns

```
In [74]: print('Total Letter Count in the group:'+str(df['Letter Count'].sum()))
```

Total Letter Count in the group:210965

```
In [75]: print('Total Word Count in the group:'+str(df['Word Count'].sum()))
```

Total Word Count in the group:38057

What is the most common number of words in a message?

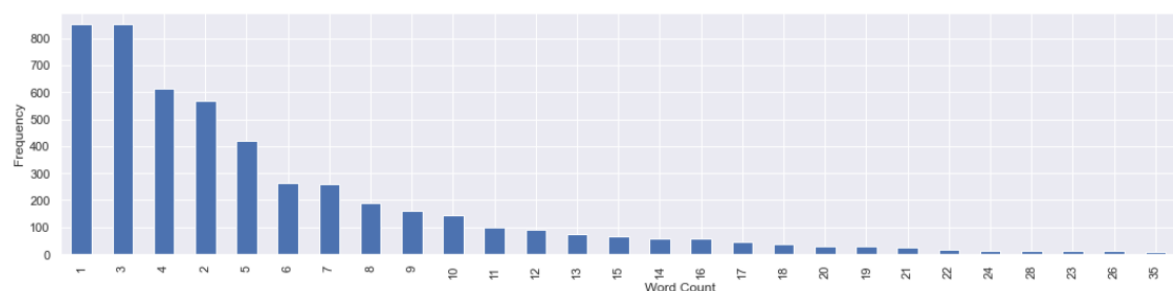
```
In [76]: plt.figure(figsize=(20,4))
word_count_value_counts=df['Word Count'].value_counts()
top30_word_count=word_count_value_counts.head(30)
top30_word_count.plot.bar()
plt.xlabel('Word Count')
plt.ylabel('Frequency')
```

```
Out[76]: Text(0, 0.5, 'Frequency')
```

What is the most common number of words in a message?

```
In [76]: plt.figure(figsize=(20,4))
word_count_value_counts=df['Word Count'].value_counts()
top30_word_count=word_count_value_counts.head(30)
top30_word_count.plot.bar()
plt.xlabel('Word Count')
plt.ylabel('Frequency')
```

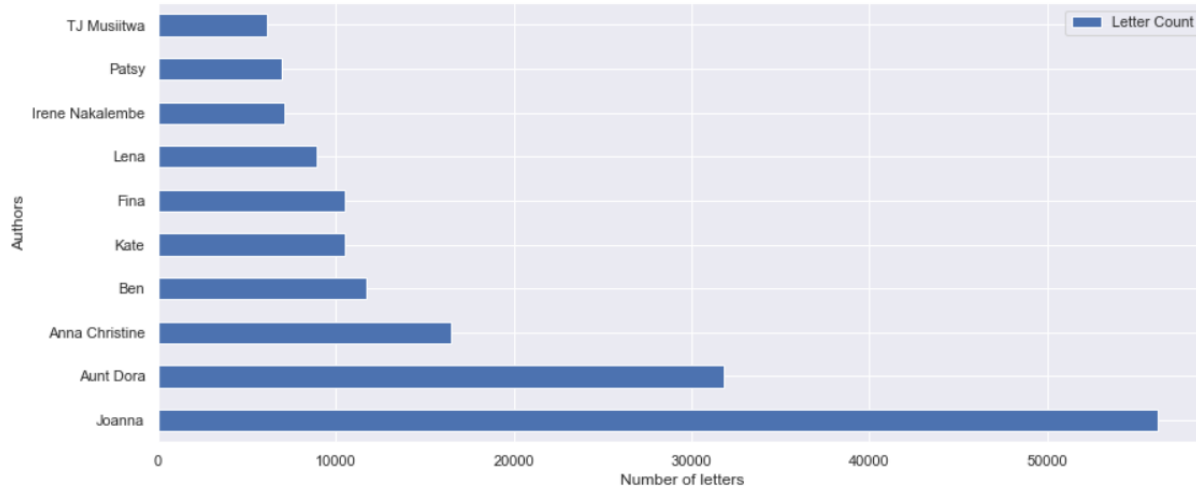
```
Out[76]: Text(0, 0.5, 'Frequency')
```



```
In [77]: ## Summary by user
users=df.groupby('Author')['Author'].count()
```

```
In [83]: top10_sorted_total_letter_grouped_by_author.plot.barh()
plt.xlabel('Number of letters')
plt.ylabel('Authors')
```

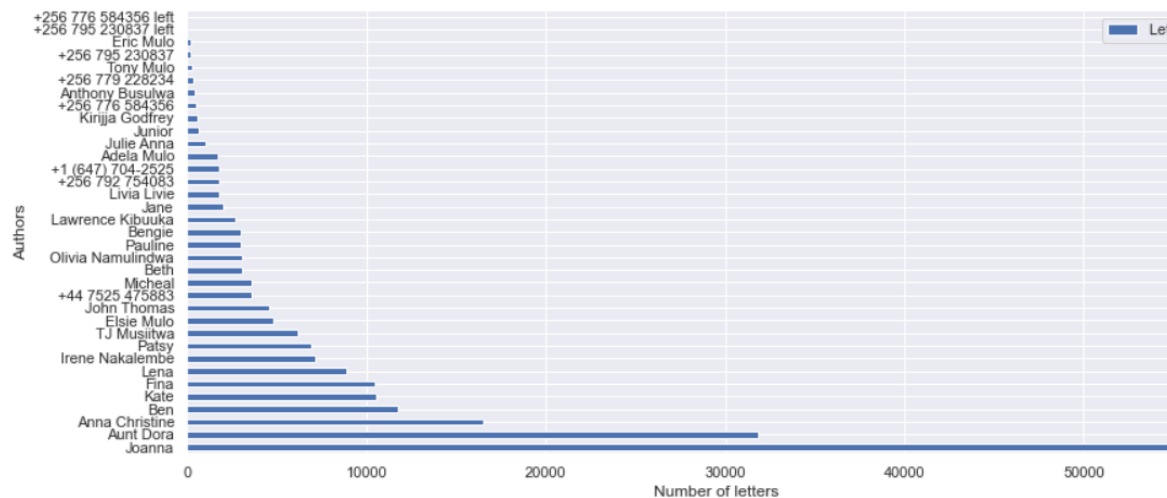
Out[83]: Text(0, 0.5, 'Authors')



```
In [84]: total_letter_count_by_author=df[['Author', 'Letter Count']].groupby('Author').sum()
sorted_total_letter_count_by_author=total_letter_count_by_author.sort_values('Letter Count',ascending=False)
sorted_total_letter_grouped_by_author=sorted_total_letter_count_by_author
sorted_total_letter_grouped_by_author.plot.barh()
plt.xlabel('Number of letters')
plt.ylabel('Authors')
```

Out[84]: Text(0, 0.5, 'Authors')

Out[84]: Text(0, 0.5, 'Authors')



```
In [78]: print(users)
```

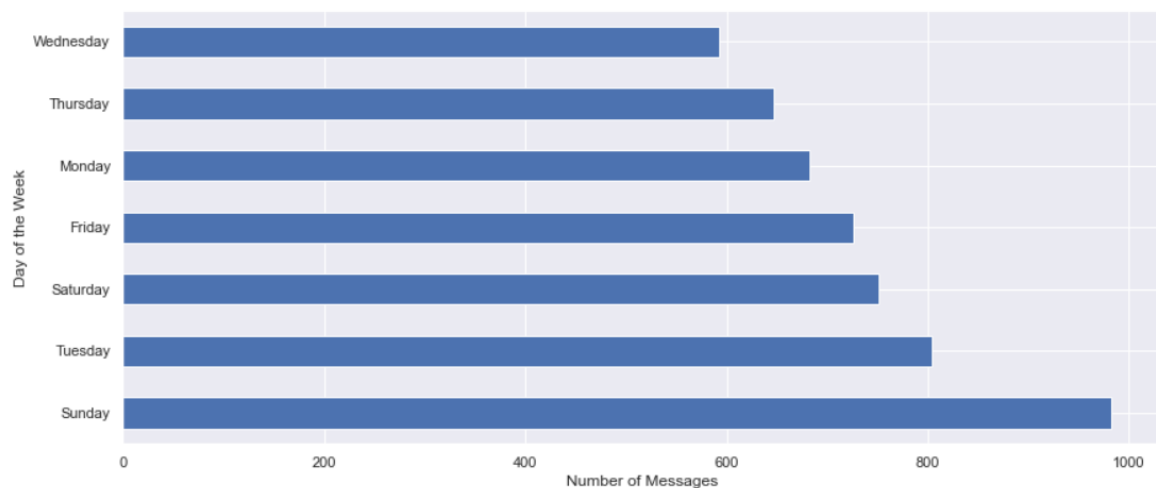
Author	
+1 (647) 704-2525	74
+256 776 584356	13
+256 776 584356 left	1
+256 779 228234	15
+256 792 754083	50
+256 795 230837	12
+256 795 230837 left	1
+44 7525 475883	89
Adela Mulo	62
Anna Christine	449
Anthony Busulwa	20
Aunt Dora	285
Ben	241
Bengie	110
Beth	64
Elsie Mulo	225
Eric Mulo	10
Fina	368
Irene Nakalembe	227
Jane	61
Joanna	1007
John Thomas	154
Julie Anna	38
Junior	41
Kate	256
Kirijja Godfrey	36
Lawrence Kibuuka	98
Lena	341
Livia Livie	50
Micheal	172
Olivia Namulindwa	80
Patsy	194
Pauline	185

When was the group most active?

```
In [86]: df['Day of Week'].value_counts().plot.barh()  
plt.xlabel('Number of Messages')  
plt.ylabel('Day of the Week')
```

```
Out[86]: Text(0, 0.5, 'Day of the Week')
```

```
Out[86]: Text(0, 0.5, 'Day of the Week')
```

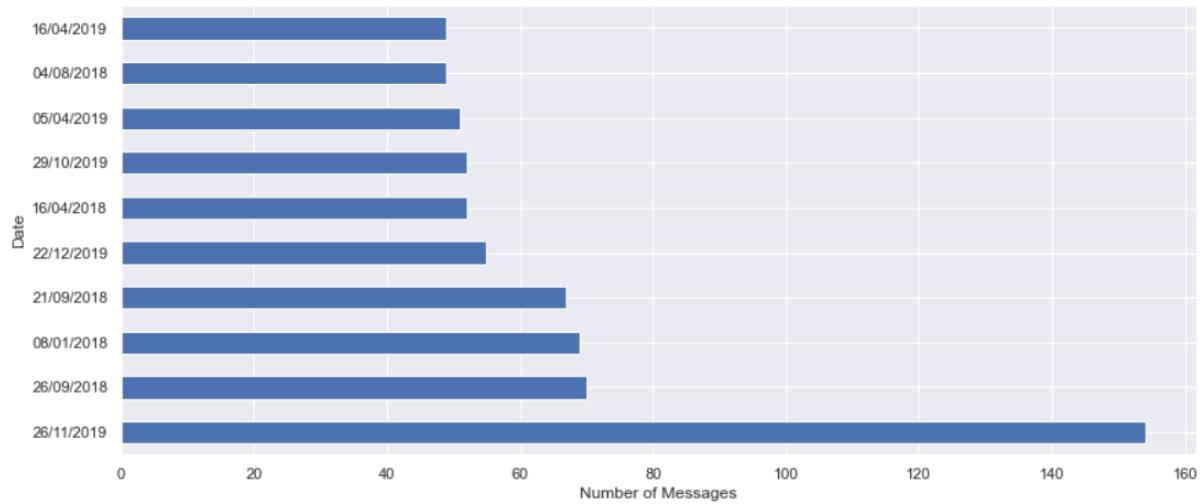


Any Particular dates?

```
In [87]: df['Date'].value_counts().head(10).plot.barh()  
plt.xlabel('Number of Messages')  
plt.ylabel('Date')
```

```
Out[87]: Text(0, 0.5, 'Date')
```

Out[87]: Text(0, 0.5, 'Date')



```
In [91]: common_words=''
for val in df['Message'].values:
    val=str(val)
    tokens=val.split()

    for i in range(len(tokens)):
        tokens[i]=tokens[i].lower()

    for words in tokens:
        common_words=common_words + words + ' '
famcloud=WordCloud(width=800,height=800).generate(common_words)

plt.figure(figsize=(8,8))
plt.imshow(famcloud)
plt.axis('off')
plt.tight_layout()
plt.show()
```


Chapter 8

Conclusion

8.1 Conclusion

A crucial stage in any machine learning method is data pre-processing. Before any further processing, raw data or unstructured data must be converted to a structured representation. Code-switching, which is quite common in text data gathered from social media, complicates processing, and must be eliminated. Emojis' interactions with text messages and this code-switching are not taken into consideration by the present approaches. The algorithm described in this study successfully distinguishes the messages of the two participants from a variety of WhatsApp chats. Emojis and sentences are then separated, and the sentences are then translated into a specific language to make them general and united. Using the suggested formula, the emojis are further categorized for sentiment analysis. The pre-processing data were then utilized to assess the respondents' social conduct in context and determine whether they were introverts or extroverts. The sentiment and behavioral analysis methods mentioned in this work can be applied to data pre-processing. A smarter and more intelligent method for building systems that interact with humans, such as chatbots, can be inspired by fusing a scientific approach for behavioural analysis with practical engineering goals of evaluating emotions in NLP texts. Additionally, psychologists can combine these strategies to improve their models for both sociological research and mental health treatment. Our next step is to create a classifier that, based on

emojis and text given by the user, reports a person's behavioural report of their feelings.

8.2 Future Scope

1. Adding a service for authorising it to make it an application.
 2. Establishing a database and keeping track of users.
 3. Adding voice chat will improve the application's effectiveness.
 4. Including Web Support.
-

References

- Barman, U., Das, A., Wagner, J., & Foster, J. (2014). Code Mixing: A Challenge for Language Identification in the Language of social media. In *Proceedings of the 9th international conference on language resources and evaluation (lrec)*.
- Bhatt, A., & Arshad, M. (2016). Impact of WhatsApp on Youth: A Sociological Study. *IRA-International Journal of Management & Social Sciences*.
- Dahiya, S., Mohta, A., & Jain, A. (2019). Text Classification based Behavioural Analysis of WhatsApp Chats. In *Preceding of fifth international conference on communication electronics system (icces)*.
- Hota, S., & Pathak, S. (2018). KNN classifier-based approach for multi-class sentiment analysis of Twitter data. *International Journal of Engineering and Technology*, 7(3), 1372-1375.
- Hussien, W., Tashtoush, Y., Al-Ayyoub, M., & Kabi, M. A. (2016). Are Emoticons Good Enough to Train Emotion Classifiers of Arabic Tweets? In *Proceedings of the international conference on arab network for natural language processing*.
- Islami, M. T. F. A., Barakbah, A. R., & Harsono, T. (October 2020). Interactive Applied Graph Chatbot with Semantic Recognition. In *Published by ieee*.
- Kharde, V. A., & Sonawane, S. S. (2016). Sentiment Analysis of Twitter Data: A Survey of Techniques. *International Journal of Computer Applications*, 139, 5-15.
- K. Norambuena, . V. C. M., B. E. F. L. (2019). Sentiment analysis and opinion mining applied to scientific paper reviews.
- Kootbodien, A., Prasad, N., & Ali, M. (2018). Trends and Impact of WhatsApp as a Mode of Communication among Abu Dhabi Students.
- Krebs, F., Lubascher, B., Moers, T., Schaap, P., & Spanakis, G. (2017). Social Emotion Mining Techniques for Facebook Posts Reaction Prediction. In *Proceedings of the 19th international conference on human-computer interaction with mobile devices*

and services.

- Kumar, N. S., & Sharma, S. (2017). Survey Analysis of “The Usage and Impact of WhatsApp Messenger”. *Global Journal of Enterprise Information System*.
- Live Mint. (July 2018). *How widespread is WhatsApp’s usage in India?* Available: <https://www.livemint.com/Technology/>.
- Lu, X., Ai, W., Liu, X., Li, Q., Wang, N., Huang, G., & Mei, Q. (2016). Learning from the ubiquitous language: an empirical analysis of emoji usage of smartphone users.
- Mika, V., Graziotin, D., & Kuuttila, M. (2018). The evolution of sentiment analysis – A review of research topics, venues, and top cited papers. *Computer Science Review*, 27, 16-32.
- Mohta, A., Jain, A., Saluja, A., & Dahiya, S. (2020). Pre-Processing and Emoji Classification of WhatsApp Chats for Sentiment Analysis. In *Proceedings of fourth international conference on i-smac*.
- Neshan, S. A. S., & Akbari, R. (April 2020). A Combination of Machine Learning and Lexicon Based Techniques for Sentiment Analysis. In *6th international conference on web research (icwr)*.
- Norambuena, K., Lettura, B. E. F., & Villegas, C. M. (2019). Sentiment analysis and opinion mining applied to scientific paper reviews. *Intelligent data analysis*, 23(1), 191-214.
- Pang, B., & Lee, L. (2008). Opinion Mining and Sentiment Analysis. *Foundations and Trends in Information Retrieval*, 2(1-2).
- Rajkumar, S. J., Shirsat, V. S., & Deshmukh, S. N. (2019). Sentiment analysis on product reviews using machine learning techniques. In *Cognitive informatics and soft computing* (p. 639-647).
- Tian, Y., Galery, T., Dulcinati, G., Molimpakis, E., & Sun, C. (2017). Facebook sentiment: Reactions and Emojis.
- Vallikannu, R., & Meyyappan, T. (2019). Twitter text mining for sentiment analysis on people’s feedback about Oman tourism. In *2019 4th mec international conference on big data and smart city (icbdsc)* (p. 1-5).
- WhatsApp - Chat. (2022). Available: <https://www.kaggle.com/datasets/rijudhara/whatsappchat>.
- WhatsApp Statics 2023 – Usage, Users, Revenue and more. (n.d.). Available: <https://meetanshi.com/blog/whatsapp-statistics/>.

Zhuhanling, X. (May 2021). An Emotional Topic Recommendation Chat Assistant. In *Ieee 15th international symposium on applied computational intelligence and informatics*.