An Industry-Oriented Mini Project on

WhatsApp Chat Analysis using Machine Language

A report submitted in partial fulfillment of the requirements for the award of

The B. Tech Degree

By

D Kranthi Kumar	(20EG105109)
M Vedansh Reddy	(20EG105118)
J Naveen Kumar	(20EG105138)
Thoundur Preethi	(20EG105151)

Under the Guidance of

Mr. P Rajasekhar Reddy

Assistant Professor

DEPARTMENT OF COMPUTER SCIENCE & ENGINEERING

ANURAG UNIVERSITY

VENKATAPUR – 500088

TELANGANA

YEAR 2023-24

DECLARATION

We hereby declare that the Report entitled "WhatsApp Chat Analysis using Machine learning" submitted for the award of Bachelor of technology Degree is our original work and the Report has not formed the basis for the award of any degree, diploma, associate ship or fellowship of similar other titles. It has not been submitted to any other University or Institution for the award of any degree or diploma.

Place: Anurag University, Hyderabad

D Kranthi Kumar
(20EG105109)

M Vedansh Reddy
(20EG105118)

J Naveen Kumar
(20EG105138)

Thoundur Preethi
(20EG105151)

CERTIFICATE

This is to certify that the Report / dissertation entitled “WhatsApp Chat Analysis using Machine Learning” that is being submitted by D Kranthi Kumar (20EG05109), M Vedansh Reddy (20EG105118), J Naveen Kumar (20EG105138), Thoundur Preethi (20EG105151) in partial fulfilment for the award of B. Tech in Computer Science and Engineering to the Anurag University is a record of bonafide work carried out by them under my guidance and supervision.

The results embodied in this Report have not been submitted to any other university or Institute for the award of any degree or diploma

.

HEAD OF THE DEPARTMENT

SIGNATURE OF SUPERVISOR

P Rajasekhar Reddy

(ASSISTANT PROFFESOR)

ACKNOWLEDGMENT

We would like to express our sincere thanks and deep sense of gratitude to project supervisor **Mr. P Rajasekhar Reddy** (Assistant Professor) for his constant encouragement and inspiring guidance without which this project could not have been completed. His critical reviews and constructive comments improved our grasp of the subject and steered to the fruitful completion of the work. His patience, guidance and encouragement made this project possible.

We would like express my special thanks to **Dr. V. Vijaya Kumar**, Dean School of Engineering, Anurag University, for their encouragement and timely support in our B. Tech program.

We would like to acknowledge our sincere gratitude for the support extended by **Dr. G. Vishnu Murthy**, Dean, Dept. of CSE, Anurag University. We also express my deep sense of gratitude to **Dr**. **V V S S S Balaram,** Academic Coordinator, **Dr. Pallam Ravi**, Project In-Charge, **Dr. G Prabhakar Raju** Project Coordinator and Project Review committee members, whose research expertise and commitment to the highest standards continuously motivated us during the crucial stage our project work.

ABSTRACT

WhatsApp has become a popular platform for communication, and its chat data contains a wealth of information that can be leveraged for various purposes. Users can use this platform to provide the chat analyzer with exported WhatsApp (.txt file) as input.

The objective of this research is to explore the application of NLP algorithms to extract meaningful insights from WhatsApp conversations. It provides a precise picture of total number of messages, activity of each person, total words, shared links, busiest day and more. These techniques enable the identification of key patterns, trends, and themes within the conversations. The analysis focuses on several aspects. Firstly, sentiment analysis is performed to evaluate the overall sentiment of the chats, allowing for an understanding of the emotional tone of the conversations. Secondly, named entity recognition is used to identify important entities mentioned in the chats, such as people, and dates. The results of the analysis shed light on various aspects of the WhatsApp chat data. They reveal the prevalent sentiment within the chats, the frequently mentioned entities. They can be leveraged for customer feedback analysis, social media monitoring, market research, and understanding user behavior. The insights derived from the WhatsApp chat analysis using NLP techniques can aid in decision-making processes, enhance communication strategies, and provide valuable feedback for organizations and individuals.

Overall, this study demonstrates the effectiveness of NLP techniques in analyzing WhatsApp chat data, enabling a deeper understanding of the conversations and extracting meaningful insights. We are developing an application using Machine Learning and NLP, which analyzes the WhatsApp conversation and makes appropriate recommendation.

LIST OF FIGURES

LIST OF TABLES

INDEX

1. INTRODUCTION

1.1 Introduction

One of the most widely used instant messaging services globally is WhatsApp. With over 2 billion active users, it has become a significant source of communication between people. WhatsApp chat analysis involves the examination of the content of these conversations, the participants involved, and the patterns of communication. One of the primary uses of WhatsApp chat analysis is for research purposes. Researchers can gather data from WhatsApp conversations to analyse various aspects of human behaviour, such as language use, social dynamics, and communication patterns. By studying WhatsApp chats, researchers can gain insights into the way people interact with each other, how they express themselves, and how they form relationships. Another use of WhatsApp chat analysis is in the business world. Companies can use WhatsApp to communicate with their customers and gather valuable feedback. By analysing these conversations, businesses can better understand their customers' needs and preferences, and tailor their products or services accordingly. Additionally, companies can monitor employee WhatsApp conversations to ensure that they comply with company policies and regulations. WhatsApp chat analysis can also be useful in personal relationships. By examining WhatsApp conversations, individuals can gain insight into their communication patterns with their friends or family members. They can identify areas where they may need to improve their communication skills or areas where they need to work on building stronger relationships. WhatsApp chat analysis is a valuable tool for researchers, businesses, and individuals. By examining WhatsApp conversations, one can gain valuable insights into human behaviour and communication patterns. Whether for research, business or personal use, analysing WhatsApp chats can provide valuable information for improving relationships and achieving better communication.

1.2 Project Scope

Dealing with huge data such as business chats or organizational group chats might be difficult. We can overcome this problem by using Python Programming alongside the Machine Learning algorithms for faster analysis and for accurate insights of the chats. Machine Learning is becoming increasingly important for WhatsApp chat analysis due to its ability to automate and streamline analysis, provide more accurate insights, and enable personalized and predictive analysis. As WhatsApp continues to be a popular platform for communication, the use of ML for chat analysis will likely become even more critical for businesses and individuals alike. We have used Python Programming for data analysis and Machine learning for detecting the emotion of the individual from their chats.

1.3 Project Overview

With the help of Python Programming, we have pre-processed the exported WhatsApp chat file. We have used pandas to convert the chat data into data frames. Using the data frames, we have analysed the timelines, count of messages and count of users. We used the external stop words dataset to remove the commonly used words for emotion detection. We later detected the emotion of the individual or group by using Sentiment analysis.

1.4 Objectives

The main objective of the project is to analyse thousands of WhatsApp chats in minutes. This project helps the business analysts to understand their customer behaviour and get the insights of the product or the services offered by them. In today's world WhatsApp has become the major media for communication be it for personal, professional or business purposes. It has become a great media for crimes as well which needs to be monitored. With our project we can identify the suspects based on their chat behaviour. It also helps to prevents suicides and find the people suffering with depression in life.

The Objectives of this project are:

•The main objective of this project is to develop a WhatsApp Chat Analyzer; the system can extract the words and emojis to predict the emotions.

•This system aims to exploit machine learning techniques to assist in the prediction of emotions and recommendations.

•Easy to analyse the WhatsApp chats and provide personalized recommendations.

•To reduce the time to analyse the WhatsApp chats.

•To detect any crime activities in prior.

2. LITERATURE REVIEW

● Wang, Yongming. "Using Machine Learning and Natural Language Processing to Analyse Library Chat Reference Transcripts." Information Technology and Libraries 41, no. 3 (2022). Improves library services by applying AI and machine learning techniques to library data. Chat reference in libraries generates a large amount of data in the form of transcripts. This study uses machine learning and natural language processing methods to analyse one academic library's chat transcripts over a period of eight years. The built machine learning model tries to classify chat questions into a category of reference or nonreference questions. The purpose is to predict the category of future questions by the model with the hope that incoming questions can be channelled to appropriate library departments or staff.

● Ahmad, Zishan, Raghav Jindal, Asif Ekbal, and Pushpak Bhattacharyya. "Borrow from rich cousin: transfer learning for emotion detection using cross lingual embedding." Expert Systems with Applications 139 (2020): 112851. This paper proposes an efficient technique to mitigate the problem of resource scarcity for emotion detection in Hindi by leveraging information from a resource-rich language like English. The model follows the methods of a deep transfer learning framework which efficiently captures relevant information through the shared space of two languages, showing significantly better performance compared to the monolingual scenario that learns in the vector space of only one language. And uses CNN and Bi-LSTM as base learning models.

● Akhilesh Kumar, Bhavna Bajpai, Rachit Adhvaryu, Suthar Dhruvi Pankaj Kumar, Prajapati Parth Kumar Gordhanbhai, and Atul Kumar. "An Efficient Approach of Product Recommendation System using NLP Technique." Materials Today: Proceedings (2021). This is a system, which can recommend the products which are like the searched products. This will help the consumer to find out another product in case the item is unavailable, or the searched product is not good enough, or when they would like to look through different similar products. A good recommendation system has been found out to be financially beneficial for the companies also. It is found out that consumer is 35% more likely to buy a product if the recommendation is good enough for consumers. NLP technologies and CNN to help in predicting similar products. CNN used at last to create a feature vector from product images, and use this vector combined with all the other vectors, for prediction. VGG-16 architecture used to extract the features from the images.

● Shashank and Pushpak Bhattacharya. "Emotion Analysis of Internet Chat". A system for Emotion Analysis of Instant Messages (IM). Using Instance Based classifier we have shown that our system can outperform similar systems in the IM domain. Tagged instant messages and elaborate feature engineering can help a lot in increasing the performance of text classification of unstructured, ungrammatical text. The impact of class imbalance on classification has been studied and demonstration has been made of how under sampling can help mitigate this problem.

2.1 Existing System

In olden days there was no analysis for WhatsApp chat. One had to look after each and every message from the chat. We didn't have any system which could analyse thousands of messages and extract emotion from it. There isn't a CSV file available for analysis if someone wants to do it. WhatsApp Application provides an export txt file which is in raw format. It is very complicated for analysis. Therefore, we must disregard that system and instead use the WhatsApp Chat Analyzer.

Disadvantages of Existing system:
- It was time consuming
- Required man power
- Raw data
- Difficult to analyse
- Emotion could not be extracted

2.2 Proposed system

The "WhatsApp Chat Analysis Using Machine Learning" provides a platform to the user which enables users to analyse WhatsApp chats. We can get the insights of the chat. It becomes easy to understand the data with the help of the graphs and charts which are used to represent the data. It reduces a lot of time for analysis. Most importantly our system helps to detect the emotion of the individual by extracting the words used and running them through the Machine learning model that we have developed. It provides an upper hand to the businesses by making it easy for them to understand the customers, gaining insights into the customers interests helping them to increase their repeated customers count. It also provides the typical information such as the number of users, Number of chats, busiest days, most active hours, most active users.

Advantages of Proposed System:
- Compatible with all devices
- Time efficient
- Easy to use
- Emotion detection
- Free to use

3. IMPLEMENTATION AND RESULTS

3.1 Languages Used

1. **Python**: Python is a high-level, interpreted programming language that Guido van Rossum developed and was made available in 1991. Since then, it has grown in popularity and is now one of the most widely used programming languages, with uses ranging from web development to data analysis and scientific computing. Python's simplicity and usability are among its most important characteristics. The syntax is simple and straightforward, making the language easy to learn and use. It is also designed to be readable and intuitive. With a sizable standard library that offers a wide range of built-in functions and modules for a number of applications, Python is also incredibly adaptable. Because Python is an interpreted language, code can be run without first needing to be compiled.

2. **Machine Learning**: In the quickly expanding subject of machine learning, algorithms and models are created that can recognise patterns in data and predict outcomes without explicit programming. Python is a well-known programming language with a large community of tools and libraries that makes it ideal for machine learning. Python offers a variety of potent machine learning libraries and frameworks, such as NumPy, SciPy, Pandas, Scikit-learn, and TensorFlow. In terms of data preprocessing, feature extraction, model selection, and model evaluation, these technologies offer a wide range of functionalities. With support for massive, multi-dimensional arrays and matrices as well as a variety of mathematical operations, NumPy is a well-known Python library for numerical computing. A library called SciPy, which is based on NumPy, adds support for operations like optimisation, integration, and interpolation in scientific computing.

3.2 Steps involved in Implementation

3.2.1 Constructing Dataset

A dataset of words and their emotions must first be compiled. This can be done by observing the terms that are frequently used in the chats and assigning the appropriate emotion to them. We need a diverse set of terms and their spelling in order to obtain representative data. It is appropriate to refer to numerous chats with other people to gain a thorough understanding of the words, sentences, and emotions. Before analysing the chat, we must also make a list of stop words, such as articles, punctuation, and prepositions, to remove them from the conversation.

3.2.2 Importing Libraries

In this project, we're utilizing a variety of libraries, including TensorFlow, Matplotlib, NumPy, SK-Learn and Pandas is used to extract the chat and transform them into data frames.

TensorFlow: TensorFlow is an open-source machine learning and artificial intelligence software library. It was developed by Google and is used for a variety of machine learning operations, including image classification, natural language processing, and predictive modelling.

Matplotlib: Matplotlib is a Python module for data visualization. It is used to deliver dynamic, animated, and interactive visualizations in Python. With Matplotlib, you can create a wide range of visualizations, including line plots, scatter plots, bar plots, error bars, histograms, bar charts, pie charts, box plots, and more.

Pandas: An open-source Python library for data analysis and manipulation is called Pandas. It offers tools for data analysis and working with structured data that are simple to use. Series, a one-dimensional labelled array, and DataFrame, a two-dimensional labelled array, are the two primary data structures in Pandas. For activities including data wrangling, data cleansing, data exploration, and data analysis, Pandas is frequently employed. As well as handling missing data, reshaping and pivoting data, and merging and joining datasets, it offers functionality for these tasks. In a variety of disciplines, including finance, economics, social sciences, and more, Pandas is a popular choice for data analysis.

3.2.3 Data Pre-Processing

The user inputs the Exported WhatsApp chat text file to the system. This text file consists of raw data with date, usernames, message, media data and group notification messages. The main objective of this data processing is to split the data into different categories and represent the whole set of data separately. We use the data frames to represent the data where each column consists of a particular set of data. The stop words are removed from the messages before inputting the words into the machine learning model for the emotion detection. The words other than the stop words which help in detecting the emotion of the individual user are recorded into a text file.

3.2.4 Model Creation

The machine learning model is created using the dataset of the words and the emotions. We use the Multinomial Naive Bayes theorem to build the model which is trained on 70% data of the dataset and is tested on the remaining 30% of the data. This model takes the text file created and runs the naive bayes algorithm in order to predict the emotion from the trained data and later predicts the overall emotion of the user and outputs the results to the user in the form of pie chart.

3.2.5 Data Visualization

The data which is analysed such as the busiest days, most active users, count of messages, count of media messages, top active users, emotion of the individual etc are given as a result to the users in the form of graphs and charts.

3.3 Sample Code

1. Importing Required Libraries:

Import re: The functions in this module allow you to determine whether a given string matches a given regular expression.

Import os: Python offers a mechanism to communicate with the underlying operating system using its "os" module. It offers tools for managing processes, environment variables, and many other things in addition to working with files and directories.

Import pandas as pd: This library provides data structures to deal with different types of data.

Import matplotlib. pyplot as plt: This library helps in data visualization like plotting graphs, bar charts.

Import numpy as np: This library provides function to deal with arrays in python

Import url extract: This library provides functions to deal with URL and work on the URL's.

```
[1]: import re
     import os
     import pandas as pd
     import matplotlib.pyplot as plt
     import numpy as np
     !pip install urlextract
     import urlextract
     !pip install wordcloud
     from wordcloud import WordCloud
     !pip install emoji
     import emoji
```

Fig. 1 Importing Libraries

2. Importing a Data file:

f= open('data/WhatsApp chat with WAGDemoGroup (4).txt','r',encoding='utf-8'): This line opens the file in the specified location in the read mode and assigns the it to the variable f

data=f.read(): This line reads the data from the file and assigns to 'data' variable

print(data): This line displays the data read from the file.

```
[15]: pattern='\d{1,2}/\d{1,2}/\d{2,4},\s\d{1,2}:\d{2}\s-\s'

[17]: messages = re.split(pattern,data)[1:]
      messages

[17]: ['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.\n',
       'You created group "WAGDemoGroup"\n',
       'Vedansh: Hey all\n',
       'Vedansh: How are you all?\n',
       'Vedansh: 😀😀\n',
       'Naveen AU: Hloo\n',
       'Vedansh: How are you Naveen?\n',
       'Vedansh: All good?\n',
       'Naveen AU: Yeah good man\n',
       'Naveen AU: What about you\n',
       "Vedansh: I'm fine 😊\n",
       'Vedansh: Just a little stressed about college and exams 😔\n',
       'Naveen AU: Say more\n',
       'Naveen AU: Yes I Too!\n',
       "Naveen AU: I'm tensed\n",
       'Vedansh: It is really very consuming... getting really tired after college 😔\n',
       'Naveen AU: Yes\n',
       'Naveen AU: Where are our friends\n',
       'Vedansh: Hey @919948603542\n',
       'Vedansh: Hey @918522922311\n',
       'Vedansh: You guys there?\n',
       'Preethi AU: https://youtu.be/h3vZF9w2cVk?si=tvIRWnpirUx4_TVq\n',
       'Vedansh: How are you guys preparing for the exam?\n',
       'Vedansh: Woww... i really like her channel\n',
       'Preethi AU: Watch out this to releif your stress😍\n',
       'Vedansh: Her videos are really funny\n',
       'Vedansh: Thanks Preethi 😁\n',
       'Vedansh: Where is @919948603542 ?\n',
       'Vedansh: I guess he is busy studying\n',
       "Preethi AU: Mee too and I'm very much addicted to stress Busters\n",
       'Preethi AU: Maybe🔥\u200d♀\n',
       'Vedansh: Fair enough 😂\n',
```

Fig.2 Splitting the Data

3. Data Pre-processing:

pattern='\d{1,2}/\d{1,2}/\d{2,4},\s\d{1,2}:\d{2}\s-\s': This line helps to create a pattern to split the data from the imported data

messages = re.split(pattern,data)[1:]: This line helps to split the data according to the pattern

dates=re.findall(pattern,data): This line helps to find all the data which satisfies the pattern.

df = pd.DataFrame({'user_message': messages, 'message_date': dates}): This line helps to convert data to data frames to access and modify the data easily

df['message_date'] = pd.to_datetime(df['message_date'], format='%d/%m/%y, %H:%M - '): This line helps to convert the datetime data into proper sequence.

df.rename(columns={'message_date':'date'} , inplace=True): This line helps to rename the columns in the data frame

df.head() : This line is used to display the first few rows in the data frame

df.shape : This line shows the size of the data frames as rows and columns

[19]:
```
df = pd.DataFrame({'user_message': messages, 'message_date': dates})
df['message_date'] = pd.to_datetime(df['message_date'], format='%d/%m/%y, %H:%M - ')

df.rename(columns={'message_date':'date'} , inplace=True)
df.head()
```

[19]:

	user_message	date
0	Messages and calls are end-to-end encrypted. N...	2023-09-27 19:11:00
1	You created group "WAGDemoGroup"\n	2023-09-27 19:11:00
2	Vedansh: Hey all\n	2023-09-27 19:12:00
3	Vedansh: How are you all?\n	2023-09-27 19:12:00
4	Vedansh: 😀😀\n	2023-09-27 19:14:00

[21]:
```
df.shape
```

[21]:
```
(43, 2)
```

Fig.3 Converting data into data frames

[24]:
```
users = []
messages = []
for message in df['user_message']:
    entry = re.split('([\w\W]+?):\s', message)
    if entry[1:]:  # user name
        users.append(entry[1])
        messages.append(" ".join(entry[2:]))
    else:
        users.append('group_notification')
        messages.append(entry[0])

df['user'] = users
df['message'] = messages
df.drop(columns=['user_message'], inplace=True)

df.head()
```

[24]:

	date	user	message
0	2023-06-07 19:10:00	group_notification	Messages and calls are end-to-end encrypted. N...
1	2023-06-07 19:11:00	group_notification	You created group "WAGDemoGroup"\n
2	2023-06-16 09:12:00	Vedansh	Hey all\n
3	2023-07-16 09:12:00	Vedansh	How are you all?\n
4	2023-07-17 11:14:00	Vedansh	😀😀\n

Fig.4 Splitting the messages

The above code (Fig.4) helps to split the data into user names and the message sent by the user. We create two columns in the data frame namely user and message and assign the respective values obtained from splitting the data.

```
[25]: df['only_date'] = df['date'].dt.date
      df['year'] = df['date'].dt.year
      df['month_num'] = df['date'].dt.month
      df['month'] = df['date'].dt.month_name()
      df['day'] = df['date'].dt.day
      df['day_name'] = df['date'].dt.day_name()
      df['hour'] = df['date'].dt.hour
      df['minute'] = df['date'].dt.minute
```

```
[116]: df.head()
```

[116]:

	date	user	message	only_date	year	month_num	month	day	day_name	hour	minute
0	2023-06-07 19:10:00	group_notification	Messages and calls are end-to-end encrypted. N...	2023-06-07	2023	6	June	7	Wednesday	19	10
1	2023-06-07 19:11:00	group_notification	You created group "WAGDemoGroup"\n	2023-06-07	2023	6	June	7	Wednesday	19	11
2	2023-06-16 09:12:00	Vedansh	Hey all\n	2023-06-16	2023	6	June	16	Friday	9	12
3	2023-07-16 09:12:00	Vedansh	How are you all?\n	2023-07-16	2023	7	July	16	Sunday	9	12
4	2023-07-17 11:14:00	Vedansh	😀😀\n	2023-07-17	2023	7	July	17	Monday	11	14

Fig.5 Converting datetime data

The above code is used to create separate columns for date, year, month, month number, day, day_name, hour and minute in the data frame for further analysis.

4. Data Analysis

[25]:
```
user_list=df['user'].unique().tolist()
if 'group_notification' in user_list:
    user_list.remove('group_notification')
user_list.sort()
users = pd.DataFrame(user_list,columns=['Users'])
users = users.rename(index = lambda x: x + 1)
```

[26]:
```
msgs=df['user'].tolist()
nom=[]
for i in user_list:
    nom.append(msgs.count(i))
users['Total Number of Messages']=nom
users.head()
```

[26]:

	Users	Total Number of Messages
1	Naveen AU	11
2	Preethi AU	5
3	Vedansh	25

[27]:
```
y=np.array(nom)
plt.pie(y,labels=user_list,autopct='%1.0f%%')
plt.show()
```

Fig. 6 Extracting unique users and data

The above code helps to identify the unique users and extract the total number of messages sent by each user and is displayed using the data frame's head function. We further plot the pie chart to gain the insights of the busiest user.

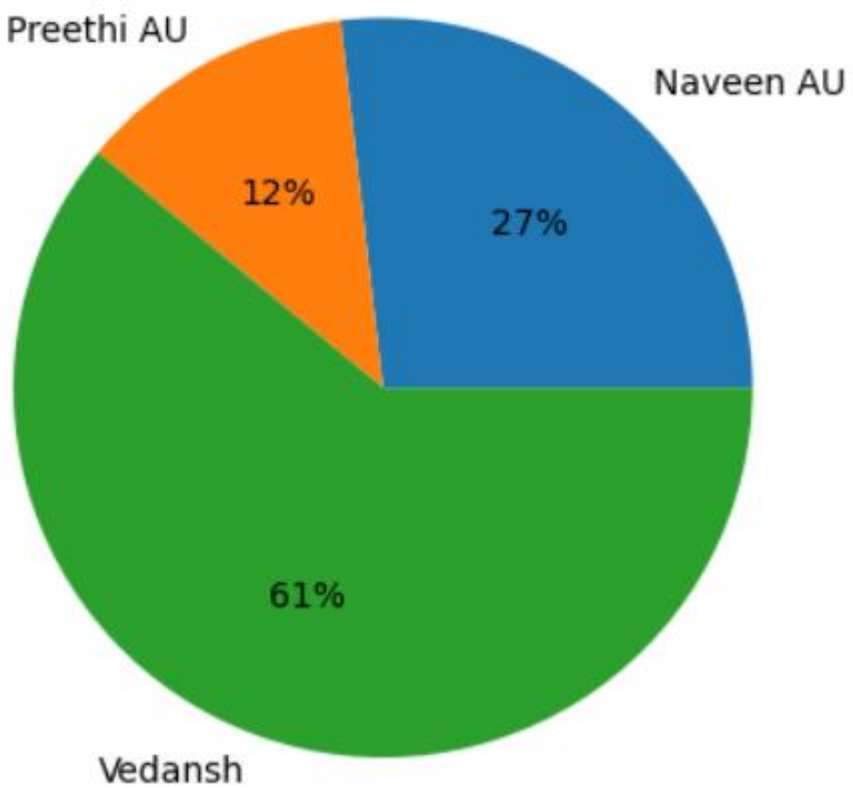

Fig. 7 Pie chart of Unique users and messages sent

The below code enables us to analyse the chat data based on the dates. Here we can extract the busiest days and number of messages sent on each day of a week. We then plot a pie chart based on the percentage of messages on each day in a week.

[28]:
```
mostdays=df['day_name'].tolist()
days=['Monday','Tuesday','Wednesday','Thursday','Friday','Saturday','Sunday']
md=df['day_name'].unique().tolist()
y=[]
for i in days:
    if i in md:
        y.append(mostdays.count(i))
    else:
        y.append(0)
xy=np.array(y)
plt.pie(xy,labels=days,autopct='%1.0f%%')
plt.legend(bbox_to_anchor=(0.7, 1.15))
plt.show()
```

Fig. 8 Daily Analysis

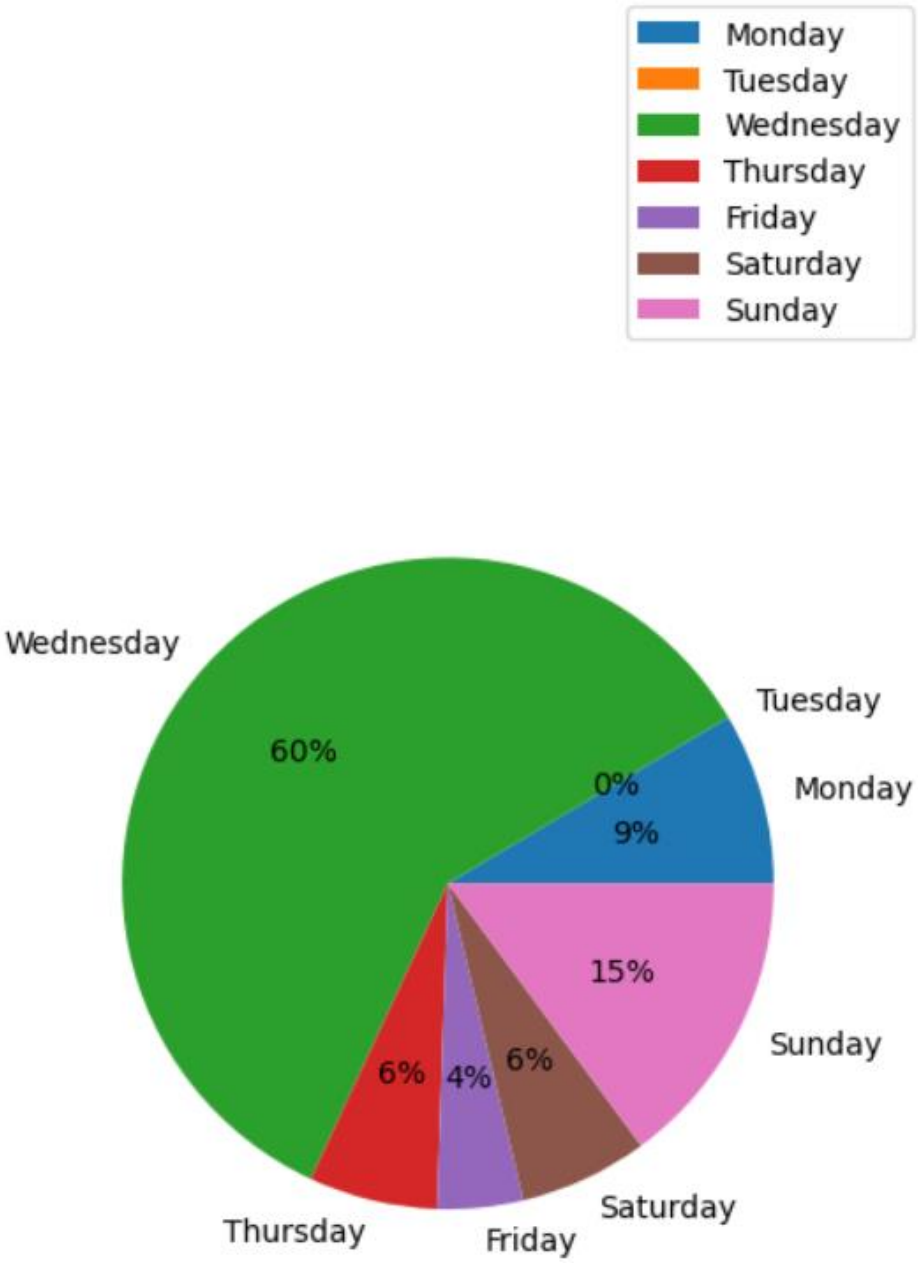

Fig. 9 Pie Chart of Daily Analysis

```
[33]: wc=WordCloud(width=500,height=500,min_font_size=10,background_color='white')
      wcount=wc.generate(df['message'].str.cat(sep=' '))
      plt.imshow(wcount)
```

```
[33]: <matplotlib.image.AxesImage at 0x13ffffd10>
```

```
[125]: <matplotlib.image.AxesImage at 0x25c5afe5d90>
```

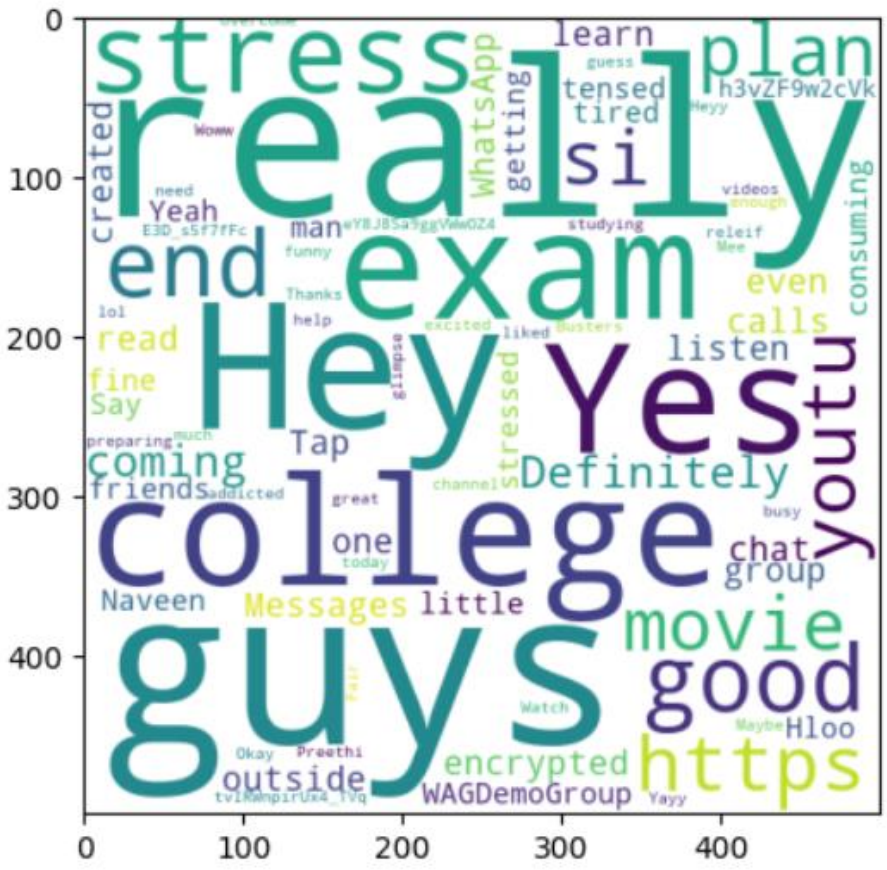

Fig. 10 WordCloud Analysis

The above code splits the messages into words . We use the ‘stop words.txt’ data set to remove the commonly used stop words from the messages . Then we use the WordCloud function from the module wordcloud and create a word cloud which represents the most commonly used words in the chats .The generate function in the word cloud is used to generate the word cloud chart . We then display the word cloud chart using the image show function .

[42]:
```
df['month_num']=df['date'].dt.month
```

[43]:
```
timeline=df.groupby(['year','month_num','month']).count()['message'].reset_index()
```

[44]:
```
timeline
```

[44]:

	year	month_num	month	message
0	2023	6	June	3
1	2023	7	July	5
2	2023	8	August	6
3	2023	9	September	28
4	2023	10	October	5

[45]:
```
time=[]
for i in range(timeline.shape[0]):
    time.append(timeline['month'][i] + ' - '+str(timeline['year'][i]))
```

[46]:
```
timeline['time']=time
timeline
```

[46]:

	year	month_num	month	message	time
0	2023	6	June	3	June - 2023
1	2023	7	July	5	July - 2023
2	2023	8	August	6	August - 2023
3	2023	9	September	28	September - 2023
4	2023	10	October	5	October - 2023

Fig. 11 Analysis based on Datetime

In the above code we extract the date year and month data to analyse the chats based on the months. We plot a timeline graph in order to get monthly usage of WhatsApp chats.

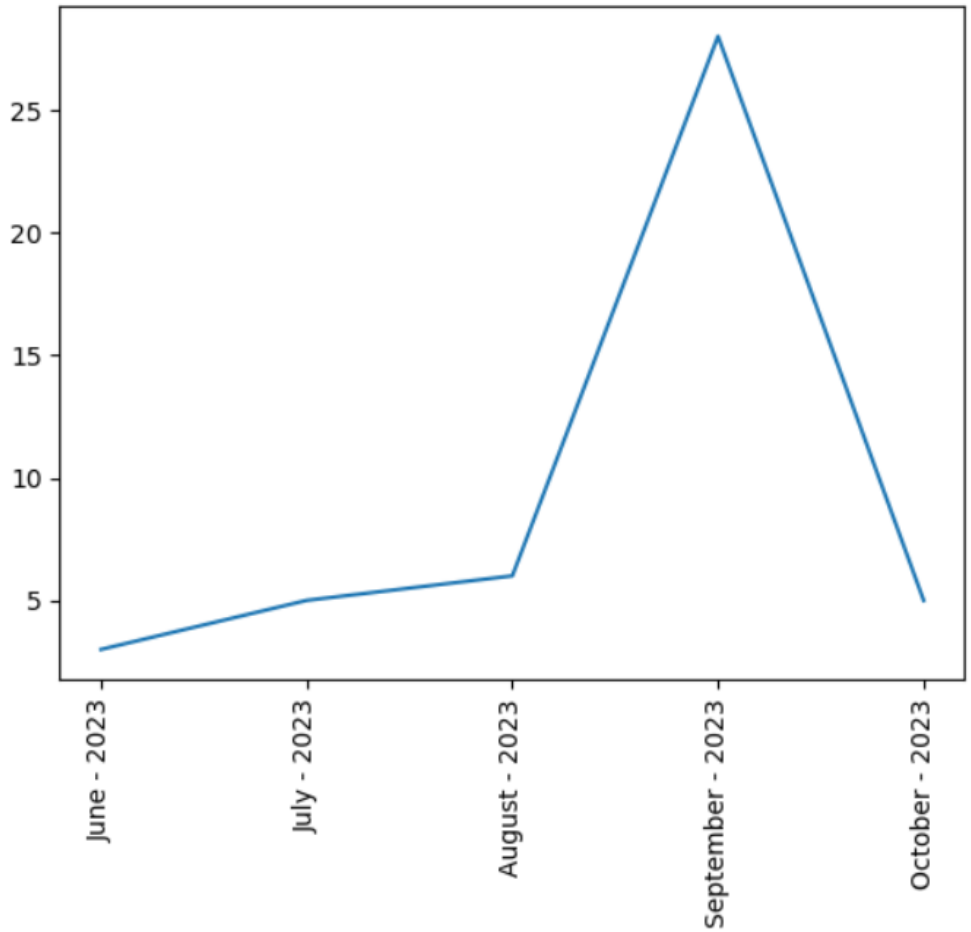

Fig. 11 Graph on Monthly Peak

The below code helps to divide the messages based on time. We divide the timelines of the messages into a fixed frequency and then plot a heatmap in order to get insights of the data based on timelines of a particular day in a week.

```
[48]: period=[]
      for hour in df[['day_name','hour']]['hour']:
          if hour==23:
              period.append(str(hour)+"-"+str('00'))
          elif hour==0:
              period.append(str('00')+'-'+str(hour+1))
          else:
              period.append(str(hour)+'-'+str(hour+1))

[49]: df['period']=period

[51]: import seaborn as sns
      plt.figure(figsize=(20,6))
      sns.heatmap(df.pivot_table(index='day_name',columns='period',values='message',aggfunc='count').fillna(0))
      plt.yticks(rotation='horizontal')
      plt.show()
```

Fig. 12 Timeline based Analysis

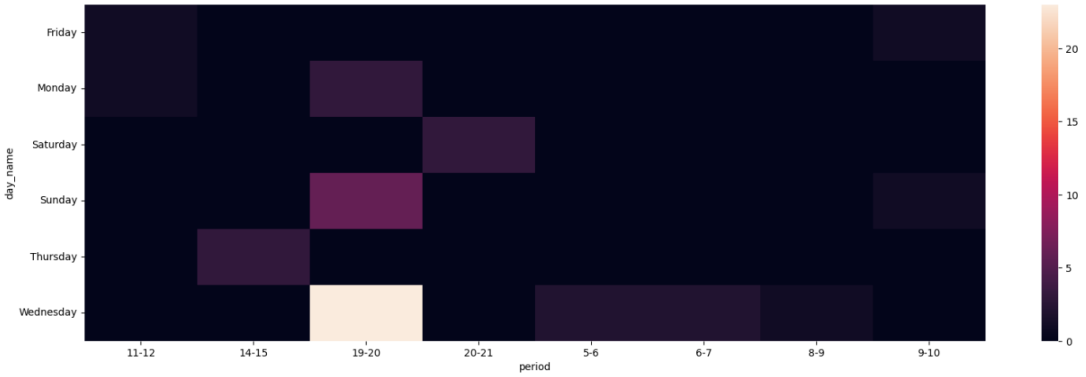

Fig. 13 Heatmap of a period of a day

3.4 Emotion Detection

```
In [2]: import pandas as pd
        import numpy as np

In [3]: import matplotlib.pyplot as plt
        import seaborn as sns
        from textblob import TextBlob

In [4]: #pip install textblob

In [5]: #! pip install neattext --upgrade
        from collections import Counter
        import neattext.functions as nfx
```

```
In [38]: #Load ml packages
         from sklearn.linear_model import LogisticRegression
         from sklearn.naive_bayes import  MultinomialNB

         #vectorizer
         from sklearn.feature_extraction.text import CountVectorizer,TfidfVectorizer

         #metrics
         from sklearn.metrics import accuracy_score,confusion_matrix,classification_report,ConfusionMatrixDisplay
         #split our dataset
         from sklearn.model_selection import train_test_split
```

Fig. 14 Importing Modules

1. Importing the modules:

Import pandas: This library helps to deal with huge amounts of data by providing special data structures

Import numpy as np: This library helps to deal with arrays and for converting the data into arrays

Import seaborn as sns: This library provides functions for data visualization using the data frames

Import Sklearn: This library provides the machine learning functions to create a model.

2. Data Extraction:

variable 'data data_set = "data/emotion_dataset_2.csv" : This line assigns the path of the data file to the _set'

df = pd.read_csv(data_set) : This line is used to read the csv file from the path assigned to 'data_set' into a data frame

df.shape : It provides the size of the data frame

df.dtypes : This line returns the list of datatypes of all the columns of the dataframe.

```
In [6]: #reading data from remote link
        data_set = "data/emotion_dataset_2.csv"
        df = pd.read_csv(data_set)

In [7]: df.head()

Out[7]:
```

	Unnamed: 0	Emotion	Text	Clean_Text
0	0	neutral	Why ?	NaN
1	1	joy	Sage Act upgrade on my to do list for tommorow.	Sage Act upgrade list tommorow
2	2	sadness	ON THE WAY TO MY HOMEGIRL BABY FUNERALIII MAN ...	WAY HOMEGIRL BABY FUNERAL MAN HATE FUNERALS SH...
3	3	joy	Such an eye ! The true hazel eye-and so brill...	eye true hazel eyeand brilliant Regular feat...
4	4	joy	@iluvmiasantos ugh babe.. hugggzzz for u .! b...	ugh babe hugggzzz u babe naamazed nga ako e...

```
In [8]: df.shape
Out[8]: (34792, 4)

In [9]: df.dtypes
Out[9]: Unnamed: 0     int64
        Emotion       object
        Text          object
        Clean_Text    object
        dtype: object
```

Fig. 15 Extracting the Data

The above code gives us the data which is grouped based on different emotions. We can also plot the Bar graph using the data from the data frames. We use the seaborn library's function countplot in order to plot a graph for the count of number of records of each emotion .

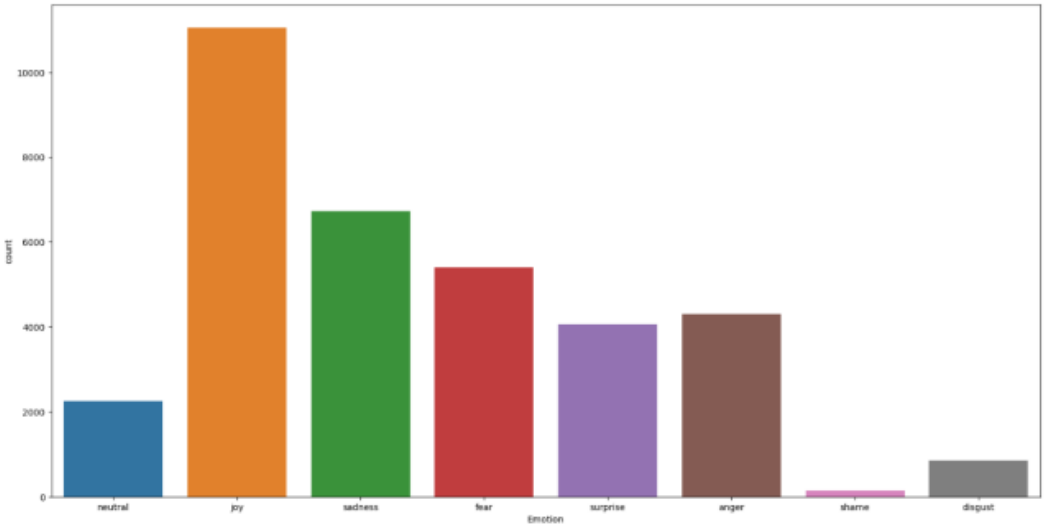

Fig. 16 Bar graph of data based on emotions

The below function 'get_sentiment' is used to extract the sentiment of the input. The sentiments are of three categories namely Positive, Neutral and Negative. We use the TextBlob function to get the polarity of the sentence. If the polarity is greater than 0 then it is considered Positive, if the polarity is less than 0 then it is considered Negative and if the polarity is 0 it is considered as Neutral. We add another column in the data frame namely sentiment to record the sentiment of each row.

```
In [15]: def get_sentiment(text):
           blob = TextBlob(text)
           sentiment = blob.sentiment.polarity
           if sentiment > 0:
             result = 'Positive'
           elif sentiment < 0:
             result = 'Negative'
           else:
             result = 'Neutral'
           return result

In [16]: get_sentiment("I love coding")
Out[16]: 'Positive'

In [17]: df['Sentiment'] = df['Text'].apply(get_sentiment)

In [18]: df.head()
Out[18]:
```

	Unnamed: 0	Emotion	Text	Clean_Text	Sentiment
0	0	neutral	Why ?	NaN	Neutral
1	1	joy	Sage Act upgrade on my to do list for tommorow.	Sage Act upgrade list tommorow	Neutral
2	2	sadness	ON THE WAY TO MY HOMEGIRL BABY FUNERALIII MAN ...	WAY HOMEGIRL BABY FUNERAL MAN HATE FUNERALS SH...	Negative
3	3	joy	Such an eye ! The true hazel eye-and so brill...	eye true hazel eyeand brilliant Regular feat...	Positive
4	4	joy	@Iluvmiasantos ugh babe.. hugggzzz for u .! b...	ugh babe hugggzzz u babe naamazed nga ako e...	Neutral

Fig. 17 Extract sentiment of the data

```
In [19]: #compare emotion vs sentiment
         df.groupby(['Emotion','Sentiment']).size()

Out[19]: Emotion   Sentiment
         anger     Negative     1787
                   Neutral      1386
                   Positive     1124
         disgust   Negative      325
                   Neutral       249
                   Positive      282
         fear      Negative     1534
                   Neutral      1843
                   Positive     2033
         joy       Negative     1682
                   Neutral      3648
                   Positive     5715
         neutral   Negative      178
                   Neutral      1523
                   Positive      553
         sadness   Negative     2630
                   Neutral      2127
                   Positive     1965
         shame     Negative       46
                   Neutral        50
                   Positive       50
         surprise  Negative      623
                   Neutral      1545
                   Positive     1894
         dtype: int64
```

Fig. 18 Emotion and Sentiment Relation

The above code helps to group the sentiments of each emotion. Then we plot a bar graph based on the grouped data.

```
In [20]: #first method: using matplotlib
         df.groupby(['Emotion','Sentiment']).size().plot(kind = 'bar')

Out[20]: <Axes: xlabel='Emotion,Sentiment'>
```

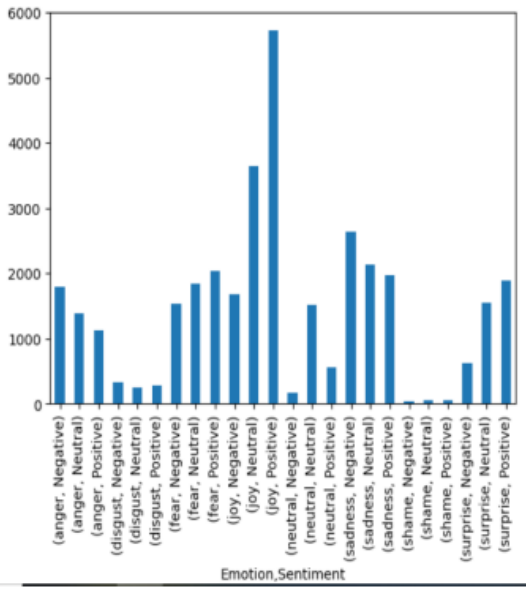

Fig. 19 Bar Graph of Emotion and Sentiment Relation

3. Data Pre-processing:

```
In [24]: df['Clean_Text'] = df['Text'].apply(nfx.remove_stopwords)
         df['Clean_Text'] = df['Text'].apply(nfx.remove_userhandles)
         df['Clean_Text'] = df['Text'].apply(nfx.remove_punctuations)
         df[['Text','Clean_Text'] ]
```

Out[24]:

	Text	Clean_Text
0	Why ?	Why
1	Sage Act upgrade on my to do list for tommorow.	Sage Act upgrade on my to do list for tommorow
2	ON THE WAY TO MY HOMEGIRL BABY FUNERALIII MAN ...	ON THE WAY TO MY HOMEGIRL BABY FUNERAL MAN I H...
3	Such an eye I The true hazel eye-and so brill...	Such an eye The true hazel eyeand so brillia...
4	@lluvmiasantos ugh babe.. hugggzzz for u .I b...	@lluvmiasantos ugh babe hugggzzz for u babe ...
...	...	...
34787	@MichelGW have you gift! Hope you like it! It'...	@MichelGW have you gift Hope you like it Its h...
34788	The world didnt give it to me..so the world MO...	The world didnt give it to meso the world MOST...
34789	A man robbed me today .	A man robbed me today
34790	Youu call it JEALOUSY, I call it of #Losing YO...	Youu call it JEALOUSY I call it of #Losing YOU
34791	I think about you baby, and I dream about you ...	I think about you baby and I dream about you a...

34792 rows × 2 columns

Key word extraction: extraxting the most commonest keywords

Fig. 20 Data Cleaning

df['Clean_Text'] = df['Text']. apply(nfx.remove_stopwords) : It is used to remove the stop words from the messages

df['Clean_Text'] = df['Text'].apply(nfx.remove_userhandles): It is used to remove usernames, URL's from the data

df['Clean_Text'] = df['Text'].apply(nfx.remove_punctuations) : It is used to eliminate punctuations from the data

df[['Text','Clean_Text']] : The cleaned text is replaced with the previous text

4. Model Creation:

We assign the text to x_featues and the respective emotions to y_labels as input to the model. CountVectorizer is used to convert the message into a numerical array We perform the training and testing of the model with 70% and 30% of the data respectively. We create a Multinomial Naive Bayes model and then feed the training data and testing data into the model.

```
In [39]: Xfeatures = df['Clean_Text']
         ylabels = df['Emotion']

In [40]: Xfeatures

Out[40]: 0                                                    Why
         1          Sage Act upgrade on my to do list for tommorow
         2       ON THE WAY TO MY HOMEGIRL BABY FUNERAL MAN I H...
         3        Such an eye  The true hazel eyeand so brillia...
         4       @Iluvmiasantos ugh babe hugggzzz for u   babe ...
                                      ...
         34787   @MichelGW have you gift Hope you like it Its h...
         34788   The world didnt give it to meso the world MOST...
         34789                              A man robbed me today
         34790      Youu call it JEALOUSY I call it of #Losing YOU
         34791   I think about you baby and I dream about you a...
         Name: Clean_Text, Length: 34792, dtype: object

In [41]: cv = CountVectorizer()
         X = cv.fit_transform(Xfeatures)

In [42]: #split dataset
         X_train,X_test,y_train,y_test = train_test_split(X,ylabels,test_size=0.3,random_state=42)

In [43]: #building model
         nv_model = MultinomialNB()
         nv_model.fit(X_train,y_train)

Out[43]: ▾ MultinomialNB
         MultinomialNB()
```

Fig. 21 Model Creation

The predict_emotion function takes a sentence and a model as arguments. It converts the sentence into an array using the cv.transform . The transformed sentence is fed into the model.predict function. The emotion with most probability is given as output

```
In [53]: def predict_emotion(sample_text,model):
             myvect = cv.transform(sample_text).toarray()
             prediction = model.predict(myvect)
             pred_proba = model.predict_proba(myvect)
             pred_percentage_for_all = dict(zip(model.classes_,pred_proba[0]))
             print(prediction[0])
             return pred_percentage_for_all

In [54]: predict_emotion(sample_text,nv_model)

         anger

Out[54]: {'anger': 0.32858411428380846,
          'disgust': 0.013719550953727618,
          'fear': 0.14763130808694586,
          'joy': 0.1907303996025054,
          'neutral': 0.0034697708474193192,
          'sadness': 0.2643186281678986,
          'shame': 3.138562266513996e-05,
          'surprise': 0.05151484243502955}

In [55]: predict_emotion(["i hate you"],nv_model)

         anger

Out[55]: {'anger': 0.32858411428380846,
          'disgust': 0.013719550953727618,
          'fear': 0.14763130808694586,
          'joy': 0.1907303996025054,
          'neutral': 0.0034697708474193192,
          'sadness': 0.2643186281678986,
          'shame': 3.138562266513996e-05,
          'surprise': 0.05151484243502955}
```

Fig. 22 Model Prediction

5. Saving Model:

```
In [59]: import joblib
         model_file = open("data/emotion_classifier.pkl","wb")
         joblib.dump(nv_model,model_file)
         model_file.close()

         f=open('data/vector.pkl','wb')
         joblib.dump(cv,f)
         f.close()
```

Fig. 23 Saving the Model

The joblib module is used to save the Machine Learning models to extract and use them for predictions. We create a pickle file and write the bytes to the file of the model. We also save the CountVectorizer created for converting the data to arrays.

6. Data Visualisation:

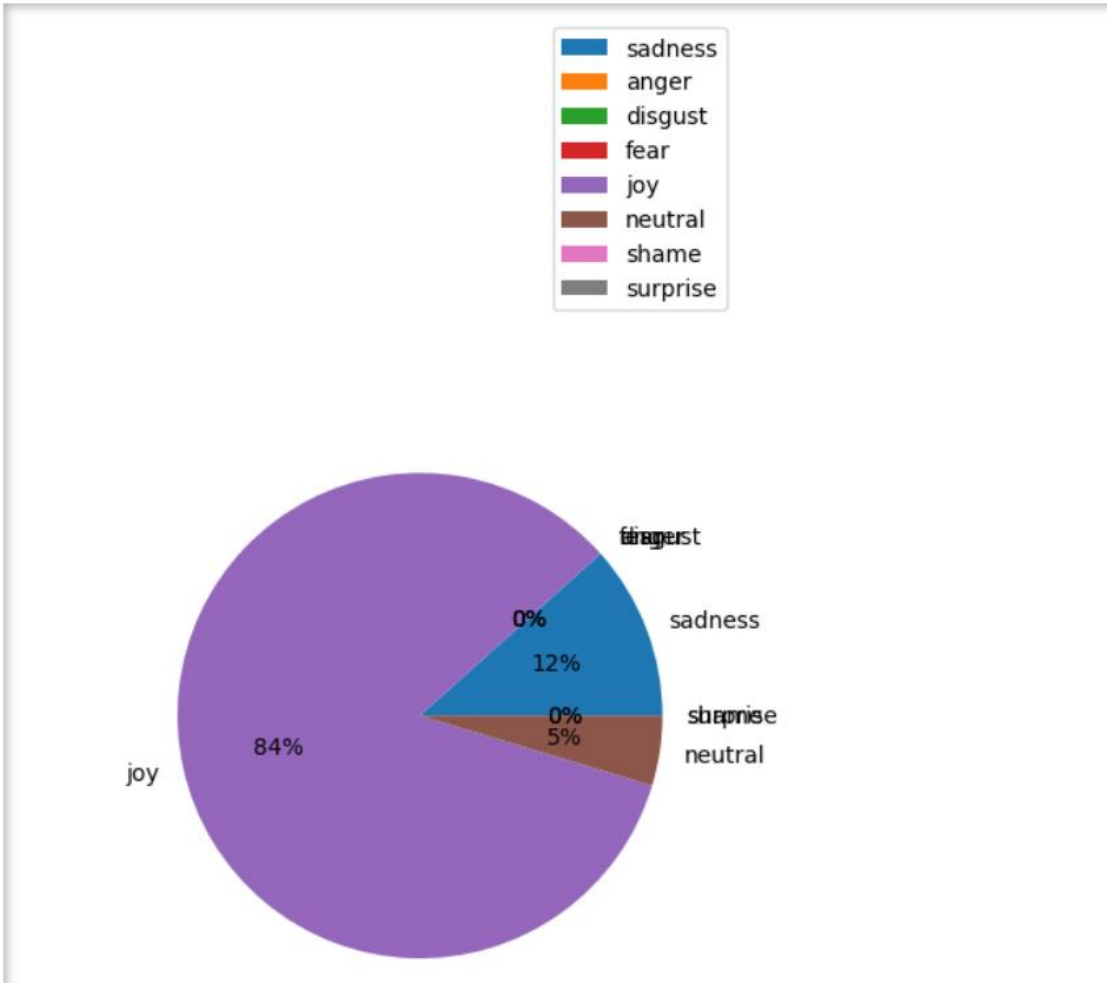

Fig. 24 Data Visualisation

In the above code, we use the model we created on the exported WhatsApp chat and we predict the emotion of each message and plot a pie chart from the whole analysis.

3.5 Output Screens:

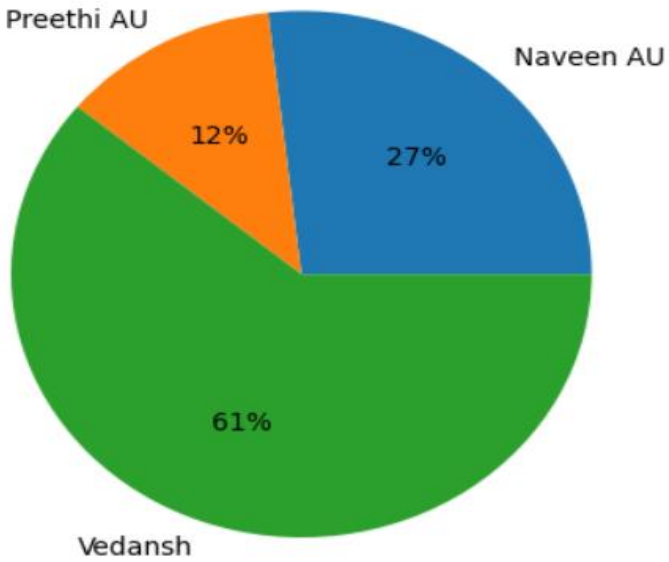

Fig. 25 Users Usage Chart

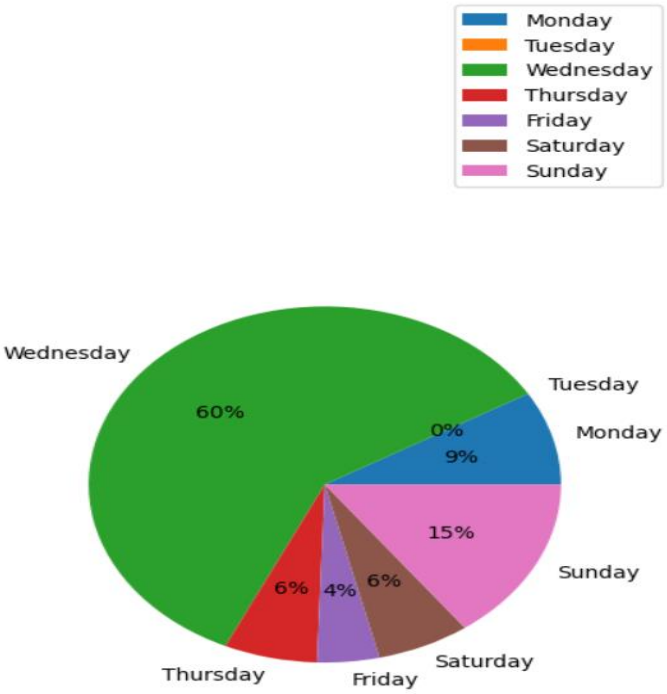

Fig. 26 Day-wise Usage Chart

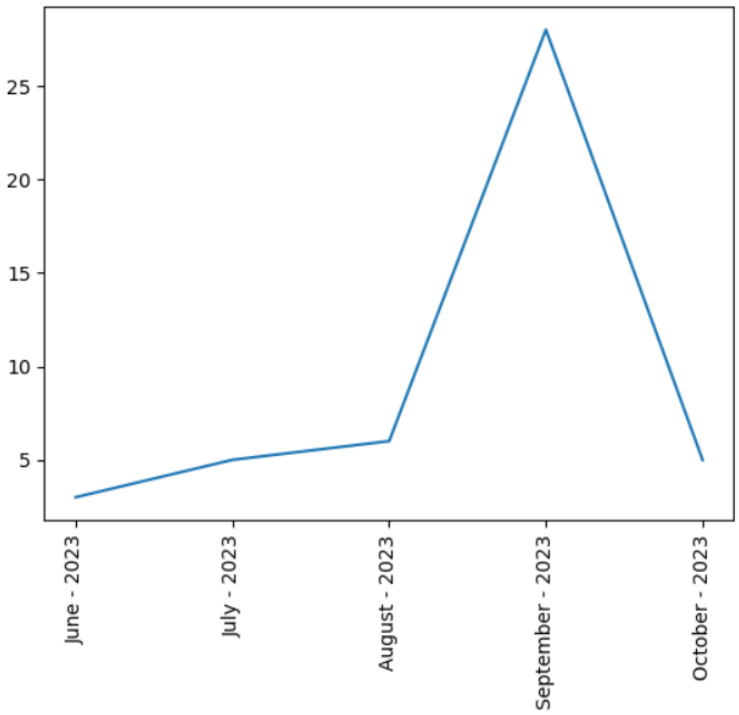

Fig. 27 Month Wise Usage Chart

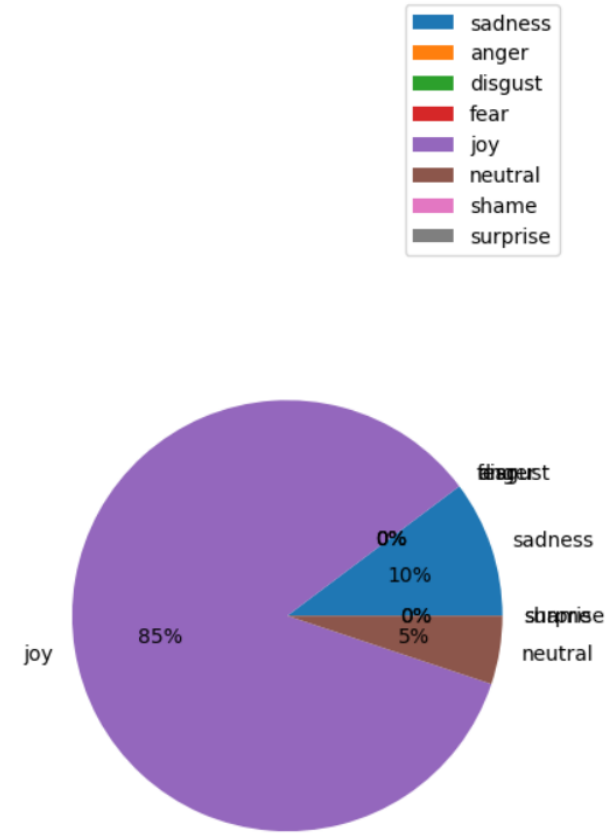

Fig. 28 Emotion Detection Chart

4. TESTING: TEST CASES AND RESULTS

Test Scenario	TC-ID	Feature Description	Pre-Requisite	Test Description	Input	Expected Result	Actual Result	P/F
Analysing Chat for frequent words	TC-01	To obtain most frequent words	Messages should be splitted into words	1. Input the chat data 2.Extract the messages 3. Split messages to words 4.Run the analysis code	Exported WhatsApp chat	top 10 frequently words in the chat	The analyser displays top 10 frequent words in the chat	PASS
Analysing the chat for most active users	TC-03	obtaining most active user	The users and count of messages should be calculated	1.Input the chat data 2. Extract unique users 3. Count the messages for each user	Exported WhatsApp chat	The analyser displays a pie chart for the most active user	The analyser displays a pie chart for the most active user	PASS
Analysing the emotion of the message	TC -03	To test the prediction on of emotion by the emotion detection n model	The model should be created and saved for importing	1.Input the emotion dataset 2. Build a model using Multinomial NB 3.Save the model using joblib	[“I hate you”]	Anger	Anger	PASS
Analysing the emotion of the message	TC -04	To test the prediction on of emotion by the emotion detection n model	The model should be created and saved for importing	1.Input the emotion dataset 2. Build a model using Multinomial NB 3.Save the model using joblib	[“Wow What a catch!”]	Joy	Joy	PASS
Analyse chat for active hours	TC -05	To analyse the activity and time spent by each user.	Date and time data Should be converted into the appropriate format.	1.Give the WhatsApp chat analyser the chat data. 2.Pick the option for active hours analysis.	Exported WhatsApp chat	messages sent during different hours of the day.	The active Hours are displayed Using charts for each user.	PASS

5. CONCLUSION AND FUTURE SCOPE

Conclusion:

The answer to a machine learning-based analysis of a WhatsApp message relies on the particular research topic and the data set employed. Here are some broad conclusions that may be drawn from such a study, though: To determine the primary subjects mentioned in the WhatsApp discussion, utilize machine learning. This can be helpful for figuring out the chat's overarching topic and locating areas of interest. Machine learning may be used to assess the tone of the chat messages. This might shed light on the participants' general state of mind and feelings. The social network of chat participants, including the frequency and kind of interactions between them, may be analysed using machine learning. This can provide light on the dynamics of the group and the functions played by the various members. The vocabulary, grammar, and syntax used by the participants may all be examined using machine learning. This can provide information about the individuals' educational background, sophistication level, and cultural and language background. Overall, a WhatsApp chat analysis employing machine learning can offer insightful information about the participants' social dynamics and communication styles. A wide range of research issues, from marketing and consumer behaviour to social and political analyses, can be informed by these findings.

Future Scope:

WhatsApp Chat Analysis is a major concern as it helps to predict the business and detect any unwanted activities. In the future we would like to add the analysis of WhatsApp media such as images, videos, voice notes, stickers, documents. We can also develop the system to run independently on the cloud so that it can be available for use from any corner of the world. We can also develop the system to provide a PDF of the whole Analysis report to the user which can be shared or stored on the cloud. Detection of spam accounts based on the chat analysis can be notified to the users. We can develop a third-party application so as to work on the smartphones directly by extracting the chats to which permission is granted by the user and send daily / weekly / monthly reports to the user to his WhatsApp or Email automatically.

6. REFERENCES

[1] Wang, Yongming. "Using Machine Learning and Natural Language Processing to Analyse Library Chat Reference Transcripts." Information Technology and Libraries 41, no. 3 (2022).

[2] Dahiya, Sonika, Astha Mohta, and Atishay Jain. "Text classification based behavioural analysis of WhatsApp chats." In 2020 5th international conference on communication and electronics systems (ICCES), pp. 717-724. IEEE, 2020.

[3] Ahmad, Zishan, Raghav Jindal, Asif Ekbal, and Pushpak Bhattacharyya. "Borrow from rich cousin: transfer learning for emotion detection using cross lingual embedding." Expert Systems with Applications 139 (2020): 112851.

[4] Akhilesh Kumar, Bhavna Bajpai, Rachit Adhvaryu, Suthar Dhruvi Pankaj Kumar, Prajapati Parth Kumar Gordhanbhai, and Atul Kumar. "An Efficient Approach of Product Recommendation System using NLP Technique." Materials Today: Proceedings (2021).

[5] “Analysis of WhatsApp Chat for Crime Investigation” by M A Rahman and A H M Kamal (2021)