VISVESVARAYA TECHNOLOGICAL UNIVERSITY

JNANA SANGAMA, BELAGAVI – 590 018

An Internship Project Report on

WHATSAPP CHAT ANALYZER

Submitted in partial fulfillment of the requirements as a part of the internship for the award of degree of Bachelor of Engineering in Information Science and Engineering

Submitted by

PRAJNA S G — **1RN20IS107**

SUHITH P R — **1RN20IS165**

Under the Guidance of

Dr. R Rajkumar

Associate Professor

Department of ISE

ESTD:2001

An Institute with a Difference

Department of Information Science and Engineering

RNS Institute of Technology

Channasandra, Dr. Vishnuvardhan Road, RR Nagar Post,
Bengaluru – 560 098

2023 -2024

RNS Institute of Technology

Channasandra, Dr. Vishnuvardhan Road, RR Nagar Post,

Bengaluru – 560 098

DEPARTMENT OF INFORMATION SCIENCE & ENGINEERING

CERTIFICATE

This is to certify that the internship project report entitled ***WHATSAPP CHAT ANALYZER*** has been successfully completed by **PRAJNA S G** bearing USN **1RN20IS107** and **SUHITH P R** bearing USN **1RN20IS165**, presently VII semester students of **RNS Institute of Technology** in partial fulfillment of the requirements as a part of the ***AI/ML Internship*** for the award of the degree of ***Bachelor of Engineering in Information Science and Engineering*** under **Visvesvaraya Technological University, Belagavi** during academic year **2023 – 2024**. It is certified that all corrections/suggestions indicated for Internal Assessment have been incorporated in the report and deposited in the departmental library. The internship project report has been approved as it satisfies the academic requirements as a part of Internship work.

Dr. R Rajkumar	**Dr. R Rajkumar / Mr. Pramoda R**	**Dr. Suresh L**
Internship Guide	Internship Coordinators	Professor & HoD
Associate Professor	Associate/Assistant Professor	Department of ISE
Department of ISE	Department of ISE	RNSIT

External Viva

Name of the Examiners	**Signature with date**
1. _____	_____
2. _____	_____

COMPANY INTERNSHIP CERTIFICATE

To be inserted here (individual students)

DECLARATION

We, **Prajna S G[USN: 1RN20IS107]** and **Suhith P R [USN: 1RN20IS165]**, student of VII Semester BE, in Information Science and Engineering, RNS Institute of Technology hereby declare that the Internship project work entitled ***WHATSAPP CHAT ANALYZER*** has been carried out and submitted in partial fulfillment of the requirements for the *VII Semester degree of* ***Bachelor of Engineering in Information Science and Engineering*** *of Visvesvaraya Technological University, Belagavi* during academic year 2023-2024.

Place : Bengaluru

Date :

PRAJNA S G
1RN20IS107

SUHITH P R
1RN20IS165

ABSTRACT

The WhatsApp Chat Analyzer is a software tool designed to analyze and extract valuable insights from WhatsApp chat conversations. As WhatsApp has become one of the most popular messaging platforms worldwide, the need to understand and derive meaningful information from the vast amount of text data generated in chats has grown exponentially. This tool aims to address this need by providing users with a comprehensive set of features and functionalities for chat analysis.

The key features of the WhatsApp Chat Analyzer include:

1. Chat Data Extraction: The tool allows users to import WhatsApp chat logs, including individual and group chats, and supports various chat export formats. It ensures the secure handling of user data and maintains user privacy.
2. Word Frequency Analysis: Users can gain insights into the most frequently used words and phrases in a chat. This analysis helps in understanding the most discussed topics and themes within the conversation.
3. User Activity Metrics: The tool provides statistics on individual user activity, including the number of messages sent, average response times, and participation levels in group chats. This can be useful for assessing user engagement and communication patterns.
4. Chat Timeline Visualization: A visual representation of the chat timeline allows users to see trends and changes in the conversation flow over time. This feature is particularly valuable for tracking discussions and events in group chats.
5. Keyword Search: Users can search for specific keywords or phrases within the chat logs, making it easy to locate and reference specific information within a conversation.

The integration of KNN elevates the WhatsApp Chat Analyzer to a more sophisticated level of analysis. These machine learning techniques enable users to derive insights that go beyond traditional analytics, providing a deeper understanding of sentiment, user behavior, and context within their WhatsApp conversations. This tool is valuable for individuals, businesses, researchers, and analysts seeking to harness the power of machine learning for chat analysis, offering a user-friendly interface and advanced capabilities to unlock hidden patterns and valuable information within WhatsApp conversations.

ACKNOWLEDGMENT

The fulfillment and rapture that go with the fruitful finishing of any assignment would be inadequate without the specifying the people who made it conceivable, whose steady direction and support delegated the endeavors with success.

We would like to profoundly thank **Management** of **RNS Institute of Technology** for providing such a healthy environment to carry out this AI/ML Internship Project.

We would like to express our thanks to our Principal **Dr. M K Venkatesha** for his support and for inspiring us towards the attainment of knowledge.

We wish to place on record our words of gratitude to **Dr. Suresh L,** Professor and Head of the Department, Information Science and Engineering, for having accepted to patronize us in the right direction with all his wisdom.

We like to express our profound and cordial gratitude to our Guide**, Dr.R Rajkumar,** Associate Professor, Department of Information Science and Engineering for their valuable guidance, constructive comments, continuous encouragement throughout the Internship.

We like to express our profound and cordial gratitude to our Internship Project Coordinators**, Dr. R Rajkumar,** Associate Professor and **Mr. Pramoda R,** Assistant Professor, Department of Information Science and Engineering for their valuable guidance, constructive comments, continuous encouragement throughout the Internship Work and guidance in preparing report.

We would like to thank all other teaching and non-teaching staff of Information Science & Engineering who have directly or indirectly helped us to carry out the Internship Project Work.

Also, we would like to acknowledge and thank our parents who are source of inspiration and instrumental in carrying out this Internship Project Work.

PRAJNA S G
USN:1RN20IS107

SUHITH P R
USN:1RN20IS165

TABLE OF CONTENTS

LIST OF FIGURES

Chapter 1

INTRODUCTION

1.1 ORGANIZATION/INDUSTRY

1.1.1 COMPANY PROFILE

NASTECH is formed with the purpose of bridging the gap between Academia and IndustryNastech is one of the leading Global Certification and Training service providers for technical and management programs for educational institutions. We collaborate with educational institutes to understand their requirements and form a strategy in consultation with all stakeholders to fulfill those by skilling, reskilling and upskilling the students and faculties on new age skills and technologies.

1.1.2 DOMAIN/TECHNOLOGY

The domain chosen for our project is AI/ML. Machine learning, the fundamental driver of AI, is possible through algorithms that can learn themselves from data and identify patterns to make predictions and achieve your predefined goals, rather than blindly following detailed programmed instructions, like in traditional computer programming. This technology allows the machine to perceive, learn, reason and communicate through observation of data, like a child that grows up and acquires knowledge from examples. Machines also have the advantage of not being limited by our inherent biological limitations. With machine learning, manufacturing companies have increased production capacity up to 20%, while lowering material consumption rates by 4%.

Nowadays, the revolutionary AI technology evolved from rule-based expert systems to machine learning and more advanced subcomponents such as deep learning (learning representations instead of tasks), artificial neural networks (inspired by animal brains) and reinforcement learning (virtual agents rewarded if they made good decisions).

The AI can master the complexity of the intertwining industrial processes to enhance the whole flow of production instead of isolated processes. This enormous cognitive capacity gives the AI the ability to consider the spatial organization of plants and thetiming constraints of live production. Another key advantage is the capability of AI algorithms to think probabilistically with all the subtlety this allows in edge cases, instead of traditional rule-based methods that require rigid theories and a full comprehension of problems.

1.1.3 Department

With the blessings of our beloved Chairman Dr. R N Shetty and the able guidance from Principal Dr. M K Venkatesha, the department of Information Science and Engineering has completed 20 years of academic excellence with a current intake of 180. The department is accredited by the NBA in the year 2011 and 2018. The department follows a quality procedure to prepare students to be industry ready and encourages them to pursue higher education. Department maintains higher academic standards with outcome based education, witnessing higher university results andranks. More than 95% of students are placed in top companies with paid internships. The department is blessed with expertized teachers with an average experience of 14+ years. Industry interaction and Alumni connect is a hallmark of the department with the Centre of Excellence in Data Science and Cyber Security.

1.2 PROBLEM STATEMENT

WhatsApp Chat Analyzer is a powerful tool designed to help users gain deeper insights into their WhatsApp conversations. It offers essential statistics, sentiment analysis, user activity tracking, and more, making it easier to understand and enhance your messaging experience. With an intuitive interface and customizable options, this tool is ideal for individuals, businesses, and researchers looking to unlock the secrets hidden within their chat data**.**

The WhatsApp Chat Analyzer provides a snapshot of your group chat's dynamics. It reveals message frequencies, active days, top users (including potential "Ghosts"), media sharers, popular emojis, and even the most-used words. This concise tool delivers valuable insights into your group's communication habits and content preferences, making it a valuable asset for understanding and optimizing your chat experience.

Chapter 2

REQUIREMENT ANALYSIS, TOOLS &TECHNOLOGIES

2.1 Hardware Requirements

Hardware Requirements: The Hardware requirements are very minimal, and the program can be run on most of the machines. Table 3.1 gives details of hardware requirements.

Table 2.1: Hardware Requirements

Processor	Intel Core i3 processor
Processor Speed	1.70 GHz
RAM	4 GB
Storage Space	40 GB
Monitor Resolution	1024*768 or 1336*768 or 1280*1024

2.2 Software Requirements

The software requirements are description of features and functionalities of the system. Table 2.2 gives details of software requirements.

Table 2.2: Software Requirements

Operating System	Windows 8.1
IDE	Anaconda
Tools	Power BI
Libraries	Pandas, Streamlit, Matplotlib,Seaborn, Regex, WordCloud, Emoji

2.2.1 Anaconda

Anaconda is the birthplace of Python data science. It is a distribution of the Python and R programming languages for scientific computing, that aims to simplify package managementand deployment. The distribution includes data-science packages suitable for Windows, Linux, and macOS. It is developed and maintained by Anaconda, Inc., which was founded byPeter Wang and Travis Oliphant in 2012. Anaconda distribution comes with over 250 packages automatically installed, and over 7,500 additional open-source packages can be.

installed from PyPI as well as the conda package and virtual environment manager. It also includes a GUI, Anaconda Navigator, as a graphical alternative to the command-line interface.

2.2.1 Jupyter Notebook

Jupyter Notebook (formerly IPython Notebook) is a web-based interactive computational environment for creating notebook documents. Jupyter Notebook is built using several open- source libraries, including IPython, ZeroMQ, Tornado, jQuery, Bootstrap, and MathJax. A Jupyter Notebook document is a browser-based REPL containing an ordered list of input/output cells which can contain code, text (using Markdown), mathematics, plots and rich media. Underneath the interface, a notebook is a JSON document, following a versioned schema, usuallyending with the ".ipynb" extension. Jupyter Notebook is similar to the notebook interface of otherprograms such as Maple, Mathematica, and SageMath, a computational interface style that originated with Mathematica in the 1980s. Jupyter interest overtook the popularity of the Mathematica notebook interface in early 2018.

2.2.2 Power BI

Power BI is an interactive data visualization software product developed by Microsoft with a primary focus on business intelligence. It is part of the Microsoft Power Platform. In March 2016, Microsoft released an additional service called Power BI Embedded on its Azure cloud platform. Power BI is a collection of software services, apps, and connectors that work together to turn unrelated sources of data into coherent, visually immersive, and interactive insights. Data may be input by reading directly from a database, webpage, or structured files such as spreadsheets, CSV, XML, and JSON. It was released on 11th July 2011.Power BI Features:

- Power BI supports powerful data discovery and exploration that enables users to answer important questions in seconds.
- No prior programming knowledge is needed; users without relevant experience can start immediately with creating visualizations using Power BI.

- It can connect to several data sources that other BI tools do not support. Power BI enables users to create reports by joining and blending different datasets.

- Power BI Server supports a centralized location to manage all published data sources within an organization.

- Power BI software's drag-and-drop features, intuitive drill-down capabilities, and natural language querying (that we'll share more about later), new users and data analysts alike can quickly create visualizations and dashboards to gain insight almost instantly.

- Power BI software's role-based permissions user can manage who has access to what data down to the row, and user can even define who can make changes to every data source or workbook.

- All views and dashboards in Power BI software are mobile and tablet compatible.
- Power BI software's Ask Data feature can be a time-saver when it comes to asking simple questions and needing to create quick visualizations.

- Additionally, sharing in Power BI software is especially powerful because, instead of sending a static report, user can share a dashboard that is interactive and provides morethan just a single view.

- Power BI software's expansive and supportive community ignites learning across the organization and equips employees with extensive training and tutorial options, collaborative forums, and support. It is part of the Microsoft Power Platform.

CHAPTER 3

Design And Implementation

3.1 Problem Statement

The following procedure shows the steps in data manipulation:

1. Preparation and reading data:

Data Gathering: To commence the analysis, the first step is to obtain the WhatsApp chat data. Users typically export their chat conversations from the app, and these exports can come in various formats such as text files, CSV, or JSON. The WhatsApp Chat Analyzer should be designed to accept these file formats and handle variations in data structure that may arise from different device platforms, such as Android or iOS.

Data Parsing: Once the chat data is imported, it needs to be parsed effectively to extract essential information. This includes identifying and separating individual messages, associating them with respective senders, and timestamping each message. Parsing also entails recognizing media content like images or videos and distinguishing them from text messages.

Cleaning and Preprocessing: Real-world chat data often contains noise, such as system-generated messages, emojis, URLs, and other non-essential elements. Cleaning and preprocessing involve removing or filtering out this noise to ensure that the subsequent analysis focuses on meaningful content. Additionally, it may involve handling missing data or resolving discrepancies in date and time formats.

Data Storage and Structuring: To facilitate efficient analysis, the parsed and cleaned data should be organized into a structured format, such as a data frame or database. This structured representation simplifies data retrieval and manipulation during the analysis phase.

Data Security and Privacy: Throughout the data preparation process, ensuring the privacy and security of user data is paramount. Implement encryption and anonymization techniques to protect sensitive information and give users the confidence that their data is handled securely.

2. Preprocessing:

Handling Missing Data: If there are missing messages or data points, you need to decide how to handle them. Options include omitting them from analysis, imputing missing values, or using interpolation techniques.

Date and Time Normalization: Ensure that date and time information is consistent and formatted correctly. This step is crucial for time-based analysis, such as identifying active hours or days.

3. Creation of DataFrame:

Import Data: After preprocessing, you have clean and structured text data ready for analysis. You can create a DataFrame in a programming language like Python using libraries like Pandas. Import your preprocessed data into the DataFrame.

Column Organization: In the DataFrame, each column can represent specific attributes of the chat data, such as 'Sender,' 'Message,' 'Timestamp,' and 'Media Type.' This organized structure allows for efficient data manipulation and analysis.

Data Types: Ensure that the data types of each column are appropriate. Dates and timestamps should be in datetime format, while text messages can be stored as strings.

Indexing: Assign a unique identifier or index to each chat message. This allows for easy referencing and retrieval of specific messages during analysis.

Data Exploration: Before diving into in-depth analysis, explore the DataFrame to understand its structure and verify that preprocessing has been successful. Check for data integrity and consistency.

3.2 Pseudo Code of Whatsapp Chat Analyzer

```
# Step 1: Read and preprocess chat data
chat_data = read_chat_data("chat.txt")
cleaned_data = preprocess_text(chat_data)
normalized_data = normalize_timestamps(cleaned_data)

# Step 2: Create a DataFrame
chat_df = create_dataframe(normalized_data)

# Step 3: Calculate overall frequency of total messages
total_messages = chat_df.shape[0]

# Step 4: Calculate the top 10 most active days
active_days = calculate_active_days(chat_df)
top_10_active_days = active_days.head(10)
```

```
# Step 5: Calculate the top 10 active users with a twist (detecting Ghosts)
active_users = calculate_active_users(chat_df)
top_10_active_users = active_users.head(10)
ghosts = detect_ghosts(chat_df, top_10_active_users)

# Step 6: Analyze media sharing and emojis
media_counts = calculate_media_counts(chat_df)
top_10_media_users = media_counts.head(10)

emoji_counts = calculate_emoji_counts(chat_df)
top_10_emojis = emoji_counts.head(10)

# Step 7: Analyze active hours, weekdays, and months
active_hours = calculate_active_hours(chat_df)
active_weekdays = calculate_active_weekdays(chat_df)
active_months = calculate_active_months(chat_df)

# Step 8: Create heatmaps for weekdays and months
weekday_heatmap = create_weekday_heatmap(active_weekdays)
month_heatmap = create_month_heatmap(active_months)

# Step 9: Generate a WordCloud of most used words
word_cloud = generate_wordcloud(chat_df['Message'])

# Step 10: Display results or save them to a report
display_results(total_messages, top_10_active_days, top_10_active_users, ghosts,
        top_10_media_users, top_10_emojis, active_hours, weekday_heatmap,
        month_heatmap, word_cloud)
#Step 11: Implement KNN Algorithm
```

3.3 Libraries

Pandas

Pandas is a Python package that provides fast, flexible, and expressive data structures designed to make working with" relational" or "labeled" data both easy and intuitive. It aims to be the fundamental high-level building block for doing practical, real world data analysis in Python. Additionally, it has the broader goal of becoming the most powerful and flexible open-source data analysis / manipulation tool available in any language. It is already well on its way towards this goal.

Main Features

- Easy handling of missing data (represented as NaN, NA, or NaT) in floating point as well as non-floating point data
- Size mutability: columns can be inserted and deleted from DataFrame and higher dimensionalobjects.

- Automatic and explicit data alignment: objects can be explicitly aligned to a set of labels, or the user can simply ignore the labels and let Series, DataFrame, etc. automatically align the data for user in computations.

- Powerful, flexible group by functionality to perform split-apply-combine operations on data sets, for both aggregating and transforming data.
- Make it easy to convert ragged, differently-indexed data in other Python and NumPy data structures into DataFrame objects
- Intelligent label-based slicing, fancy indexing, and subsetting of large data sets.
  - Intuitive merging and joining data sets.
  - Flexible reshaping and pivoting of data sets.
  - Hierarchical labeling of axes (possible to have multiple labels per tick).

Seaborn

Seaborn is an amazing visualization library for statistical graphics plotting in Python. It provides beautiful default styles and color palettes to make statistical plots more attractive. It is built on the top of matplotlib library and also closely integrated to the data structures from pandas. It provides dataset-oriented APIs, so that we can switch between different visual representations for same variables.

Main Features

- Built in themes for styling matplotlib graphics.
- Visualizing univariate and bivariate data.
- Fitting in and visualizing linear regression models.
- Seaborn works well with NumPy and Pandas data structures.
- It comes with built in themes for styling Matplotlib graphics.

Matplotlib

Matplotlib is a low-level graph plotting library in python that serves as a visualization utility. It was created by John D. Hunter. It is open source, and we can use it freely. Matplotlib is mostly written in python, a few segments are written in C, Objective-C and JavaScript for Platform compatibility.

Main Features

- Creates publication quality plots.

- Makes interactive figures that can zoom, pan, update.
- Customizes visual style and lausert.
- Export to many file formats.
- Can be embedded in JupyterLab and Graphical User Interfaces.
- Uses a rich array of third-partypackages built on Matplotlib.

Regular expression

This module provides regular expression matching operations similar to those found in Perl. Both patterns and strings to be searched can be Unicode strings (str) as well as 8-bit strings (bytes). However, Unicode strings and 8-bit strings cannot be mixed: that is, you cannot match a Unicode string with a byte pattern or vice-versa; similarly, when asking for a substitution, the replacement string must be of the same type as both the pattern and the search string.

Regular expressions use'\\b\\a'ckslash character ('\') to indicate special forms or to allow special characters to be used without invoking their special meaning. This collides with Python's usage of the same character for the same purpose in string literals; for example, to match a literal backslash, one might have to write as the pattern string, because the regular expression must be \\, and each backslash must be expressed as \\ inside a regular Python string literal. Also, please note that any invalid escape sequences in Python's usage of the backslash in string literals now generatea DeprecationWarning and in the future this will become a SyntaxError. This behavior will happeneven if it is a valid escape sequence for a regular expression.

Emoji

The entire set of Emoji codes as defined by the Unicode consortium is supported in addition to a bunch of aliases. By default, only the official list is enabled but doing emoji.emojize(language='alias') enables both the full list and aliases.

By default, the language is English (language='en') but also supported languages are:

- Spanish ('es')
- Portuguese ('pt')
- Italian ('it')
- French ('fr')

- German ('de')
- Farsi/Persian ('fa')
- Indonesian ('id')
- Simplified Chinese ('zh')

The utils/get_codes_from_unicode_emoji_data_files.py is used to generate unicode_codes/data_dict.py. Generally speaking it scrapes a table on the Unicode Consortium's website with BeautifulSoup and prints the contents to stdout as a Python dictionary.

WordCloud

A "WordCloud" library is a software tool or package used to create visual representations of text data, where the size of each word in the visualization corresponds to its frequency or importance within the text. WordCloud libraries typically take a collection of text documents or a dataset and generate a graphic in which words are arranged in a visually pleasing and informative manner. These word clouds help users quickly identify the most common or significant words in a text dataset, making it easier to analyze and interpret textual information. Popular programming languages like Python offer libraries such as WordCloud that make it easy to create these visualizations for text analysis and data visualization purposes.

3.4 Code Snippets

```
import re
import datetime
import numpy as np
import pandas as pd
import matplotlib.pyplot as plt
import seaborn as sns
from wordcloud import WordCloud, STOPWORDS
import emoji
import itertools
from collections import Counter
import warnings
```

```
%matplotlib inline

warnings.filterwarnings('ignore')


!pip install wordcloud

!pip install emoji
```

1. Overall frequency of total messages on the group.

```
df1 = df.copy()
df1['message_count'] = [1] * df1.shape[0]      # adding extra helper column --> message_count.
df1.drop(columns='year', inplace=True)         # dropping unnecessary columns, using `inplace=True`, since this is copy of the DF and won't affect the original DataFrame.
df1 = df1.groupby('date').sum().reset_index()  # grouping by date; since plot is of frequency of messages --> no. of messages / day.
df1

# Improving Default Styles using Seaborn
sns.set_style("darkgrid")

# For better readablity;
import matplotlib
matplotlib.rcParams['font.size'] = 20
matplotlib.rcParams['figure.figsize'] = (27, 6)      # Same as `plt.figure(figsize = (27, 6))`


# A basic plot
plt.plot(df1.date, df1.message_count)
plt.title('Messages sent per day over a time period');



# Could have used Seaborn's lineplot as well.
# sns.lineplot(df1.date, df1.message_count);


# Saving the plots
plt.savefig('msg_plots.svg', format = 'svg')
```

2. Top 10 most active days.

```
top10days = df1.sort_values(by="message_count", ascending=False).head(10)    # Sort values according to the number of messages per day.
top10days.reset_index(inplace=True)          # reset index in order.
top10days.drop(columns="index", inplace=True) # dropping original indices.
top10days

# Improving Default Styles using Seaborn
sns.set_style("darkgrid")

# For better readablity;
import matplotlib
matplotlib.rcParams['font.size'] = 10
matplotlib.rcParams['figure.figsize'] = (12, 8)
```

```
# A bar plot for top 10 days
sns.barplot(top10days.date, top10days.message_count, palette="hls");

# Saving the plots
plt.savefig('top10_days.svg', format = 'svg')
```

3. Top 10 active users on the group.

```
top10df['initials'] = ''
for i in range(10):
    top10df.initials[i] = top10df.user[i].split()[0][0] + top10df.user[i].split()[1][0]

top10df.initials[7] = "ME"
top10df.initials[8] = "DT"
```

4. Top 10 users most sent media

```
# Increasing the figure size
plt.figure(figsize=(15, 6))

# Beautifying Default Styles using Seaborn
sns.set_style("darkgrid")

# Plotting a bar graph;
sns.barplot(top10media.initials, top10media.media_sent, palette="CMRmap");

plt.title('Most Sent Media')
plt.xlabel('User')
plt.ylabel('Total Media Sent');

# Saving the plots
plt.savefig('top10media.svg', format = 'svg')
```

5. Top 10 most used Emojis

```
emoji_ctr = Counter()
emojis_list = map(lambda x: ''.join(x.split()), emoji.UNICODE_EMOJI.keys())
r = re.compile('|'.join(re.escape(p) for p in emojis_list))
for idx, row in df.iterrows():
    emojis_found = r.findall(row["message"])
    for emoji_found in emojis_found:
        emoji_ctr[emoji_found] += 1
top10emojis = pd.DataFrame()
# top10emojis = pd.DataFrame(data, columns={"emoji", "emoji_description",
"emoji_count"})
top10emojis['emoji'] = [''] * 10
top10emojis['emoji_count'] = [0] * 10
top10emojis['emoji_description'] = [''] * 10
```

```
i = 0
for item in emoji_ctr.most_common(10):
    # will be using another helper column, since during visualization, the emojis won't be
rendered.
    description = emoji.demojize(item[0])[1:-1]    # using `[1:-1]` to remove the colons ':' at
the end of the demojized strin

    # appending top 10 data of emojis.  # Loading into a DataFrame.
    top10emojis.emoji[i] = item[0]
    top10emojis.emoji_count[i] = int(item[1])
    top10emojis.emoji_description[i] = description
    i += 1

top10emojis
```

6. Most active days, most active hours, most active months.

```
# Better Readablity
import matplotlib
matplotlib.rcParams['font.size'] = 14
matplotlib.rcParams['figure.figsize'] = (18, 6)

# Beautifying Default Styles using Seaborn,
sns.set_style("darkgrid")

# Pre-Processing by month and day,
grouped_by_month_and_day = df3.groupby(['month', 'day']).sum().reset_index()[['month',
'day', 'message_count']]

# creating a pivot table,
pt = grouped_by_month_and_day.pivot_table(index = 'month', columns = 'day', values =
'message_count').reindex(index = months, columns = days)

# PLOT: heatmap.
sns.heatmap(pt, cmap = 'cividis');
plt.title('Heatmap of Month sent and Day sent');

# Saving the plots;
plt.savefig('month_day_heatmap.svg', format = 'svg')
```

7. Most used words in the chat.

```
comment_words = ' '
stopwords = STOPWORDS.update(['group', 'link', 'invite', 'joined', 'message', 'deleted',
'yeah', 'hai', 'yes', 'okay', 'ok', 'will', 'use', 'using', 'one', 'know', 'guy', 'group', 'media',
'omitted'])

for val in df3.message.values:

    # typecaste each val to string.
    val = str(val)
```

```
    # split the value.
    tokens = val.split()

    # Converts each token into lowercase.
    for i in range(len(tokens)):
      tokens[i] = tokens[i].lower()

    for words in tokens:
      comment_words = comment_words + words + ' '

wordcloud = WordCloud(width = 600, height = 600,
            background_color ='white',
            stopwords = stopwords,
            min_font_size = 8).generate(comment_words)
wordcloud.to_image()
```

8. Implementation of KNearestNeighbours

```
!pip install pandas scikit-learn
import pandas as pd

# Load your preprocessed chat data
data = df

# Shuffle the data
data = data.sample(frac=1).reset_index(drop=True)

# Split the data into training and testing sets (e.g., 80% train, 20% test)
train_data = data.iloc[:int(0.8 * len(data))]
test_data = data.iloc[int(0.8 * len(data)):]

# Extract the message text
X_train = train_data["message"]
X_test = test_data["message"]
y_train = train_data["date_time"]
y_test = test_data["date_time"]
from sklearn.feature_extraction.text import TfidfVectorizer

tfidf_vectorizer = TfidfVectorizer()
X_train_tfidf = tfidf_vectorizer.fit_transform(X_train)
X_test_tfidf = tfidf_vectorizer.transform(X_test)
from sklearn.neighbors import KNeighborsClassifier


k = 5  # You can choose an appropriate value for k
knn_classifier = KNeighborsClassifier(n_neighbors=k)
knn_classifier.fit(X_train_tfidf, y_train)

y_pred = knn_classifier.predict(X_test_tfidf)

# Depending on your analysis goal, you can use appropriate metrics for evaluation
from sklearn.metrics import accuracy_score, classification_report
```

```
accuracy = accuracy_score(y_test, y_pred)
report = classification_report(y_test, y_pred)

print(f"Accuracy: {accuracy}")
print(report)
```

Chapter 4
OBSERVATION AND RESULTS

4.1 Final DataFrame

	date_time	user	message	day	month	year	date
0	2020-01-26 16:19:00	group_notification	Messages and calls are end-to-end encrypted. N...	Sun	Jan	2020	2020-01-26
1	2020-01-24 20:25:00	group_notification	Tanay Kamath (TSEC, CS) created group "CODERS 👷 ...	Fri	Jan	2020	2020-01-24
2	2020-01-26 16:19:00	group_notification	You joined using this group's invite link	Sun	Jan	2020	2020-01-26
3	2020-01-26 16:20:00	group_notification	+91 99871 38558 joined using this group's invi...	Sun	Jan	2020	2020-01-26
4	2020-01-26 16:20:00	group_notification	+91 91680 38866 joined using this group's invi...	Sun	Jan	2020	2020-01-26
...	...	...	...	...	...	...	...
13650	2020-10-02 02:05:00	Darshan Rander (TSEC, IT)	MCQs mark kiya	Fri	Oct	2020	2020-10-02
13651	2020-10-02 02:05:00	Darshan Rander (TSEC, IT)	Sign-in kiya🤩 😅	Fri	Oct	2020	2020-10-02
13652	2020-10-02 02:11:00	Tanay Kamath (TSEC, CS)	Incognito se na?	Fri	Oct	2020	2020-10-02
13653	2020-10-02 02:28:00	Darshan Rander (TSEC, IT)	Yup	Fri	Oct	2020	2020-10-02
13654	2020-10-02 10:13:00	Dheeraj Lalwani (TSEC, CS)	guys, please do me a favor and vote in this po...	Fri	Oct	2020	2020-10-02

13655 rows × 7 columns

Fig 4.1 Final DataFrame

4.2 Overall frequency of total messages on the group.

Fig 4.2 Overall frequency of total messages on the group.

4.3 Top 10 most active days

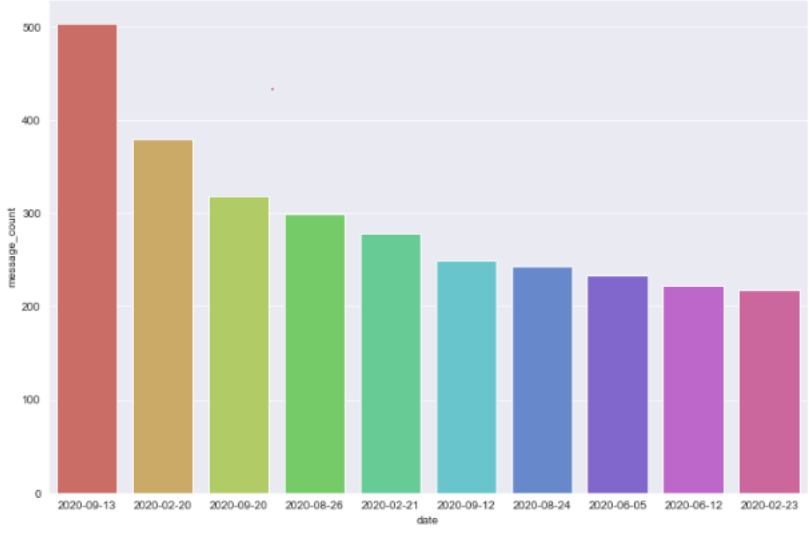

Fig 4.3 Top 10 most active days

4.4 Top 10 active users on the group.

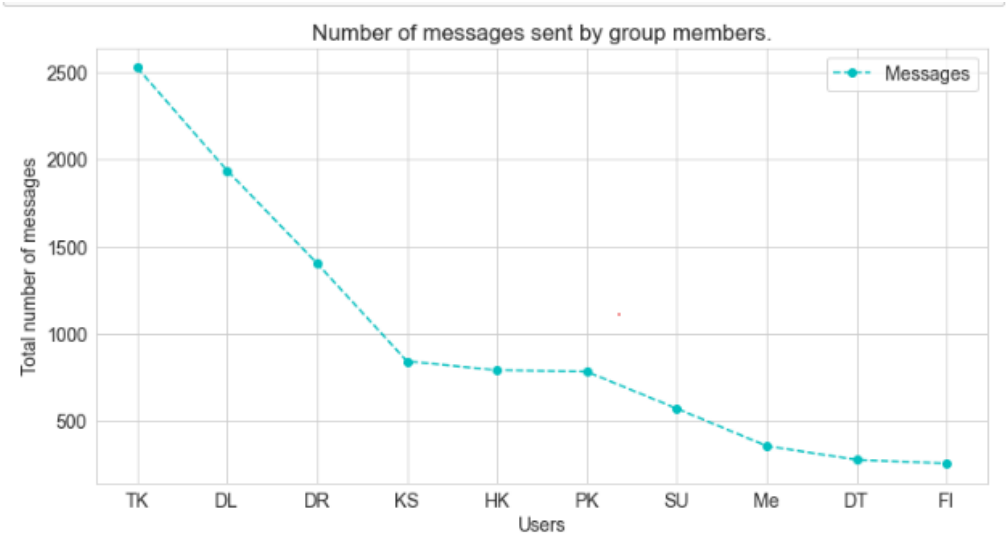

Fig 4.4 Top 10 most active users on the group

4.5 Top 10 users most sent media

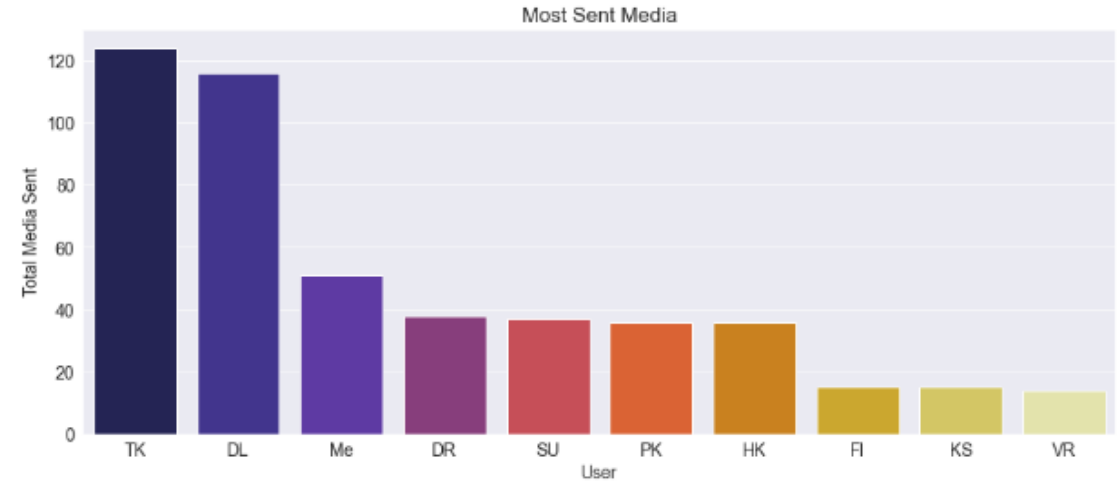

Fig 4.5 Top 10 users most sent media

4.6 Top 10 most used Emojis

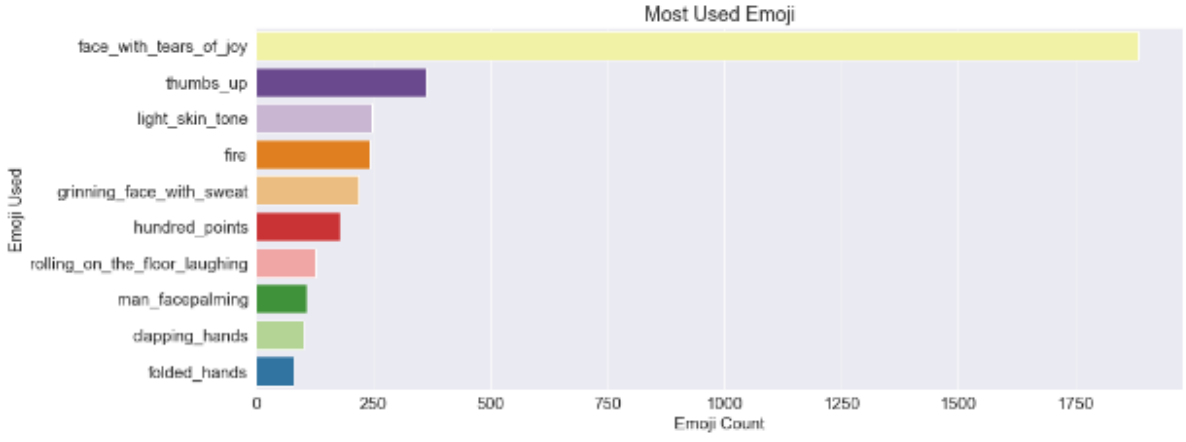

Fig 4.6 Top 10 most used emojis

4.7 Most active days, most active hours, most active months.

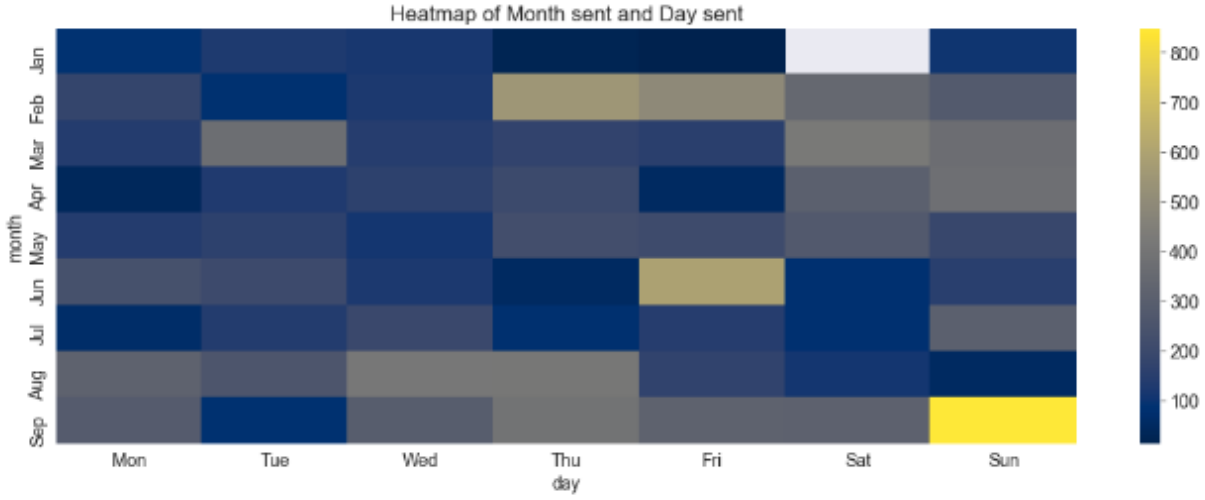

Fig 4.7 Heat Map of Most active months and days

4.8 Most used words in the chat.

Fig 4.8 Most used words in the chat.

Chapter 5

CONCLUSION AND FUTURE ENHANCEMENT

5.1 Conclusion

The insights were really interesting to look at!

We first loaded the data as a .txt file coverted it using RawtoDF function.

Then we added helper columns, manipulated datetime entries.

Then, we started analysing our whatsapp data!

Here is what we looked at!

1. Overall frequency of total messages on the group.

2. Top 10 most active days.

3. Top 10 active users on the group (Most active user had the least average message length).

Ghosts present in the group. (shocking results - 80+ participants who haven't even sent a single message!)

4. Top 10 users most sent media.- TK beats everyone by a mile!

5. Top 10 most used emojis.

6. Most active hours and weekdays.

7. Most used words - WordCloud

5.2 Future Enhancements

1. Multi-Platform Support: Extend compatibility to analyze chat data from various messaging platforms like Facebook Messenger, Telegram, and Signal, providing users with a unified messaging analysis tool.
2. Real-Time Analysis: Implement real-time chat analysis capabilities to provide users with live insights and trends as conversations unfold.
3. Chat Summarization: Develop a feature that automatically generates concise summaries of lengthy chat conversations,making it easier for users to extract key information.

4. Language Support: Expand language support to analyze chats in multiple languages, offering a more inclusive experience for users worldwide.
5. Chat Comparison: Allow users to compare and contrast different chat groups or conversations to identify commonalities and differences in communication patterns.
6. Advanced Sentiment Analysis: Enhance sentiment analysis by considering sarcasm, humor, and nuanced emotions, providing more accurate sentiment assessments.
7. Predictive Analytics: Implement predictive analytics to forecast future chat trends, helping users anticipate conversation shifts and plan accordingly.
8. Integration with Calendar Apps: Enable integration with calendar applications to schedule reminders and notifications based on chat activity and events mentioned within chats.
9. Machine Learning for Content Moderation: Utilize machine learning models to automatically detect and filter out inappropriate or offensive content within chats, ensuring a safer and more pleasant user experience.
10. Customizable Dashboards: Allow users to create customized dashboards with widgets displaying specific metrics and insights of interest, providing a personalized analytics experience.

These future enhancements for the WhatsApp Chat Analyzer aim to make the tool more versatile, insightful, and user-friendly, catering to evolving user needs and preferences in the realm of chat analytics.

REFERENCES

[1] Recommendation systems with python by Rounak Banik ,Published by packt publishing Ltd. First publication 2018.

[2] A note on pearson correlation coefficient as a metric of similarity in recommender systemPublisher: IEEE Authors:Leily Sheugh; Sasan H. Alizade

[3] S.K. Lee, Y.H. Cho and S.H. Kim, "Collaborative filtering with ordinal scale-based implicit ratings for mobile music recommendations", Information Sciences, vol. 180, no. 11, pp. 2142-2155, 2010.

[4] K. Choi, D. Yoo, G. Kim and Y. Suh, "A hybrid online-product recommendation system: combining implicit rating-based Collaborative filtering and sequential pattern analysis", Electronic Commerce Research and Applications, vol. 11, no. 4, pp. 309-317, 2012.