Data Analytics Mini Project Report on

"CHATGPT REVIEW ANALYSIS"

By

DIVYA J (1JB23MC010)

KAVYA S (1JB23MC015)

Submitted to

VISVESVARAYA TECHNOLOGICAL UNIVERSITY, BELAGAVI

In partial

fulfillment of the requirement for the award of the degree of

MASTER OF COMPUTER APPLICATIONS

UNDER THE GUIDANCE OF

Internal Guide
VIJAY KUMAR A.S
Assistant Professor
Dept. of MCA, SJBIT
Bengaluru

DEPARTMENT OF MCA

S J B INSTITUTE OF TECHNOLOGY

B G S HEALTH AND EDUCATION CITY
Kengeri, Bengaluru-560060.

Batch:2023- 2025

II Jai Sri Gurudev II

Sri Adichunchanagiri Shikshana Trust ®

S J B INSTITUTE OF TECHNOLOGY

BGS Health & Education City, Kengeri, Bangalore-560060.

DEPARTMENT OF COMPUTER APPLICATIONS (MCA)

CERTIFICATE

This is to certify that DIVYA J [1JB23MC010] and KAVYA S [1JB23MC015] are bonafide students of the Master of Computer Applications program at SJB Institute of Technology, Batch 2023–25, affiliated with Visvesvaraya Technological University, Belagavi. A mini project has been prepared by them under the guidance of Mr. VIJAY KUMAR A.S, in partial fulfillment of the requirements for the award of the degree of Master of Computer Applications by Visvesvaraya Technological University, Belagavi, Karnataka.

Signature of Guide	**Signature of HOD**
VIJAY KUMAR A. S **ASSISTANT PROFESSOR** **DEPT. OF MCA**	**Dr. S. NAGAMANI** **ASSOCIATE PROFESSOR &HEAD** **DEPT. OF MCA**

Viva – voce Examination **Date:**

Signature of Internal Examiner **Signature of External Examiner**

Name and affiliation **Name and affiliation**

DECLARATION

We, DIVYA J [1JB23MC010] and KAVYA S [1JB23MC015], hereby declare that the project report entitled Mini Project Report on “CHATGPT REVIEW ANALYSIS” has been prepared by us under the guidance of Mr. VIJAY KUMAR A.S, faculty of the MCA Department, SJB Institute of Technology. We also declare that this Mini Project work is submitted in partial fulfillment of the university requirements for the award of the degree of Master of Computer Applications by Visvesvaraya Technological University, Belagavi. Furthermore, we declare that this Mini Project is based on original work undertaken by us and has not been submitted previously for the award of the MCA degree at SJB Institute of Technology or any other institution.

Place: Bengaluru **Signature of the Student**

Date:

ACKNOWLEDGEMENT

The satisfaction and euphoria that accompany the successful completion of any task would be incomplete without mentioning the people who made it possible, because success is the result of hard work and perseverance - but above all, it is encouragement and guidance that lead the way. Therefore, we acknowledge all those whose guidance and encouragement served as a beacon of light and crowned our efforts with success.

We are grateful to the divine soul Sri Sri Sri Jagadguru Dr. Balagangadharanatha MahaSwamiji, and we are grateful to His Holiness Jagadguru Sri Sri Sri Dr. Nirmalanandanatha MahaSwamiji for providing us the opportunity to complete our academics in this esteemed college.

We would like to express our profound gratefulness to His Holiness Revered Sri Sri Dr. Prakashnath Swamiji, Managing Director, SJBIT, for providing us the opportunity to complete our academics and present this work.

We owe our deep sense of gratitude to Dr. Puttaraju, Academic Director, for his incessant encouragement.

We are grateful to Dr. K V Mahendra Prashanth, Principal, for his kind cooperation and encouragement.

We are extremely grateful to Dr. S. Nagamani, Head of the Department of Master of Computer Applications (MCA), for her cooperation and encouragement.

We express our gratitude and sincere thanks to VIJAY KUMAR A.S, Assistant Professor, for the valuable guidance throughout our Mini Project.

We are highly indebted to the Mini Project Coordinator, who has been a source of inspiration to us and has extended her fullest support throughout our Mini Project work. We also thank all the staff members of the MCA Department for their help during our project. Last but not least, we thank our parents, family members, and friends for their continuous and great support and encouragement throughout our Mini Project.

Regards,
DIVYA J
KAVYA S

TABLE OF CONTENTS

LIST OF TABLES AND FIGURES

ABSTRACT

The widespread adoption of conversational AI systems, particularly OpenAI's ChatGPT, has generated a substantial volume of user feedback through online reviews and ratings. This project presents a comprehensive sentiment analysis of over 190,000 ChatGPT user reviews, aimed at evaluating user satisfaction, identifying key features valued by users, and uncovering common issues. The dataset comprises review texts, associated star ratings, and timestamps. A structured data analytics pipeline was implemented, beginning with data preprocessing steps such as handling missing values, standardizing data types, and removing duplicates. Exploratory Data Analysis (EDA) was conducted to examine review patterns, rating distributions, and data quality metrics.

Sentiment classification was carried out using the TextBlob library, which assigned polarity-based sentiment labels (positive, neutral, or negative) to each review. Additionally, frequent phrase extraction through n-gram analysis enabled the identification of recurring themes within different sentiment categories. Visualization techniques, including Seaborn and Plotly, were utilized to interpret sentiment distributions, temporal sentiment trends, and review volume dynamics. A Net Promoter Score (NPS) was also computed using star ratings as proxies to assess overall user loyalty. The results indicate that a significant majority of users express positive sentiment toward ChatGPT, citing strengths such as its AI capabilities, usability, and academic support features. However, user concerns regarding accuracy, functionality limitations, and occasional technical errors were also identified. This study demonstrates the applicability of natural language processing (NLP) for large-scale review analysis and offers data-driven insights to inform the ongoing development and refinement of conversational AI systems.

CHAPTER 1

INTRODUCTION

The integration of conversational artificial intelligence (AI) into mainstream applications has revolutionized human-computer interaction. Among the leading tools in this domain is OpenAI's ChatGPT, a large language model capable of generating coherent, context-aware responses across a wide range of topics. As it finds widespread use in education, customer service, content creation, and productivity support, understanding how users perceive such AI systems becomes essential—not only for evaluating their effectiveness but also for ensuring continued user engagement and trust.

Unlike traditional software systems, which are often assessed using performance benchmarks such as latency or accuracy, AI-driven platforms like ChatGPT are heavily influenced by user experience. This experience is reflected through large volumes of unstructured data in the form of reviews, feedback, and ratings. These reviews contain valuable information about usability, feature satisfaction, and limitations encountered during real-world interactions.

However, the challenge lies in effectively processing and interpreting this user-generated content. The reviews are typically subjective, inconsistent in structure, and too voluminous for manual analysis. Without an automated, data-driven approach to analyze user sentiment, key insights may remain hidden - impacting both user satisfaction and the future development of the system.

This project addresses the core problem of extracting actionable insights from large-scale textual reviews of ChatGPT. While user feedback holds the potential to guide meaningful improvements, the lack of structured analysis methods leads to missed opportunities in product enhancement. By applying techniques from natural language processing (NLP) and data analytics, this study aims to classify sentiments, identify common praise and complaints, and visualize trends in user perception over time.

To achieve this, a dataset containing over 190,000 user reviews was analysed using sentiment classification, frequent phrase extraction, and advanced visualizations. The study also introduces a proxy-based Net Promoter Score (NPS) calculation to quantify user loyalty. Ultimately, this research not only evaluates how ChatGPT is received by its users but also demonstrates the value of data-driven feedback analysis in the iterative improvement of AI applications.

CHAPTER 2

REQUIREMENT ANALYSIS

The requirement analysis outlines the essential needs for conducting sentiment analysis on user reviews of ChatGPT. It identifies what the system must do, how it will be used, and the resources required. The analysis is categorized into data, functional, non-functional, and technical requirements, as observed in the project workflow.

2.1 Data Requirements

- **Dataset Format**: CSV file (chatgpt_reviews.csv)
- **Key Columns**:
  - Review Id: Unique identifier for each review
  - Review: Text content of user feedback
  - Ratings: Star rating (integer, 1 to 5)
  - Review Date: Timestamp of review submission
- **Expected Size**: ~190,000 records
- **Quality Expectations**: Minimal null values, consistent types, and date parsing compatibility

2.2 Functional Requirements

- Load dataset into a pandas DataFrame
- Perform data inspection using .info() and .describe()
- Clean null values and duplicates
- Convert Review Date to datetime format
- Apply sentiment labeling using TextBlob (Positive, Neutral, Negative)
- Extract frequent bi-grams/tri-grams using CountVectorizer
- Visualize:
  - Sentiment distribution

  - Phrase frequency
  - Sentiment trends over time
- Calculate Net Promoter Score using star ratings as proxies

2.3 Non-Functional Requirements

- Handle large datasets (~200k rows) without performance lag
- Output results in an interpretable, visually compelling manner
- Maintain modularity for future tool/model upgrades
- Ensure clarity and reusability of the notebook

2.4 Hardware and Software Requirements

- **Hardware**: Minimum 4 GB RAM, Recommended 8 GB RAM
- **Software**: Jupyter Notebook with Python 3.7+, using libraries such as:
  - pandas, numpy
  - seaborn, matplotlib, plotly
  - textblob, nltk, sklearn

CHAPTER 3

SOFTWARE REQUIREMENT SPECIFICATION

This section formalizes the software requirements needed to implement the sentiment analysis system for ChatGPT review data. It includes purpose, scope, functionality, and environmental dependencies.

3.1 Purpose

To analyze ChatGPT user reviews through sentiment classification, keyword extraction, and visual interpretation, enabling the identification of user satisfaction levels and improvement areas.

3.2 Scope

This project is limited to offline data analysis using a static CSV dataset of ChatGPT reviews. It involves data cleaning, sentiment labeling using TextBlob, keyword frequency analysis via CountVectorizer, and visualization using Python libraries.

.3.3 Intended Audience

- AI/ML researchers and students
- Product teams exploring user feedback
- Academicians evaluating AI acceptance

3.3 Functional Requirements

- Read and load CSV dataset
- Clean and preprocess data
- Perform sentiment analysis using TextBlob
- Use CountVectorizer for phrase extraction
- Visualize:
    - Sentiment breakdowns
    - Temporal sentiment patterns

- Frequent keywords in each sentiment class

- Calculate NPS from star ratings

3.4 Non-Functional Requirements

- Handle large datasets efficiently
- Ensure clear, high-quality visualizations
- Maintain code modularity and readability
- Work within a standard Python environment

3.5 Software Requirements

Specification	Minimum	Recommended
RAM	4 GB	8 GB or higher
CPU	Dual-core processor	Quad-core processor
Disk Space	500 MB	1 GB or more

Table 3.5.1

3.6 Constraints

- Sentiment model is rule-based (`TextBlob`), which may not handle sarcasm or domain-specific language accurately.
- Internet connection is required only for initial library installations.
- Analysis is exploratory and not production deployed.

CHAPTER 4

ANALYSIS AND DESIGN

4.1 Data Analysis

The dataset comprised user reviews of ChatGPT, containing textual feedback and review dates. The data analysis process involved the following steps:

- **Data Inspection**: Initial checks were performed to understand the structure, types, and completeness of the data. The dataset was found to have a mix of textual and date-based attributes, with some missing values in the review text.
- **Data Cleaning**: Duplicate entries were removed to ensure data integrity. Missing values in the review column were replaced with empty strings to prevent processing errors during analysis.
- **Text Normalization**: All reviews were converted to lowercase and stripped of leading/trailing whitespaces to standardize the input for sentiment analysis.
- **Sentiment Classification**: Each review was analyzed using the TextBlob library to calculate sentiment polarity. Reviews were then labeled as **Positive**, **Negative**, or **Neutral** based on the polarity score.
- **Visualization**: An interactive bar chart was created using Plotly to visualize the sentiment distribution, highlighting the proportion of each sentiment category within the dataset.

These steps provided a clear overview of public perception and helped identify trends in user feedback.

4.2 System Design

The overall design of the system follows a modular pipeline structure:

1. **Data Input Module**:
    - Loads the review dataset from a CSV file.
    - Performs initial validation and inspection.
2. **Preprocessing Module**:
    - Handles data cleaning, including missing value treatment and text normalization.

3. **Sentiment Analysis Module**:
    - Applies natural language processing using TextBlob.
    - Classifies reviews into sentiment categories and appends results to the dataset.
4. **Visualization Module**:
    - Generates dynamic plots using Plotly to interpret sentiment distribution.
    - Designed to support extensions for time-series or word frequency analysis.

This modular structure ensures flexibility, ease of maintenance, and scalability for future enhancements such as advanced NLP models or real-time dashboards.

CHAPTER 5

IMPLEMENTATION

The implementation of this data analytics project was executed using Python in a Jupyter Notebook environment. The process involved several key stages: data loading, cleaning, sentiment classification, and visualization. The entire workflow followed a modular structure to ensure clarity, maintainability, and reproducibility.

5.1 Environment Setup

- **Programming Language**: Python 3.x
- **Development Environment**: Jupyter Notebook
- **Libraries and Tools Used**:
    - pandas and numpy for data manipulation and analysis
    - textblob for sentiment analysis using polarity scoring
    - matplotlib, seaborn, and plotly for data visualization
    - datetime for handling and formatting review dates

5.2 Data Loading

- The dataset file chatgpt_reviews.csv was read into a pandas DataFrame.
- Initial exploration was done using .head(), .info(), and .describe() to inspect the shape, column types, and missing values.
- Column names and data types were verified and adjusted as needed, especially the Review Date column which was converted to datetime format.

5.3 Data Cleaning and Preprocessing

- **Duplicates**: Duplicate entries were removed to avoid skewing the analysis.
- **Missing Values**: Null values in the Review column were replaced with empty strings to ensure compatibility with sentiment analysis tools.
- **Text Normalization**:
    - Converted all reviews to lowercase.
    - Removed leading and trailing whitespaces for consistency.

- **Date Formatting**: The Review Date column was converted to datetime to support future enhancements like trend analysis.

5.4 Sentiment Analysis

- The TextBlob library was used to evaluate the polarity of each review.
- A custom function (get_sentiment) was implemented to classify each review based on its polarity score:
  - **Positive** if polarity > 0
  - **Negative** if polarity < 0
  - **Neutral** if polarity == 0
- A new column Sentiment was created and appended to the DataFrame to store these sentiment labels.

5.5 Visualization of Results

- The sentiment distribution was visualized using an interactive **bar chart** created with Plotly.
- The visualization helped quickly identify the overall tone of the reviews (majority being positive).
- This chart was color-coded and labeled for clarity and ease of interpretation.

5.6 Modularity and Structure

- Each step of the project from data loading to visualization was organized into distinct blocks of code.
- Functions were defined for repetitive tasks (e.g., sentiment labeling), enhancing code reusability and clarity.
- The workflow was designed to be easily extensible to allow for future integration of advanced NLP techniques (e.g., VADER, BERT) or additional visual analytics.

CHAPTER 6

TESTING

The testing phase of this project focuses on validating the correctness, consistency, and effectiveness of the sentiment analysis pipeline developed for analyzing ChatGPT user reviews. Since the project is implemented in a Jupyter Notebook environment using Python, the testing was primarily conducted through manual verification, logical checks, and output validation. Each stage of the data analysis workflow was evaluated to ensure it performed as expected and yielded meaningful results.

6.1 Data Validation Testing

- Ensured that the dataset loaded correctly using df.head(), df.info(), and df.describe() commands.
- Verified the presence of null values using df.isnull().sum() and ensured all missing entries in the review column were replaced appropriately.
- Confirmed that the Review Date column was successfully converted to datetime format with no parsing errors.

6.2 Data Cleaning Validation

- Verified the successful removal of duplicate records using df.duplicated().sum() before and after applying df.drop_duplicates().
- Manually inspected sample review entries to confirm the application of string preprocessing techniques such as stripping whitespace and converting to lowercase.

6.3Sentiment Classification Testing

- Tested the sentiment classification logic by printing polarity values alongside sentiment labels for random review samples:
    - Polarity > 0 → **Positive**

- Polarity < 0 → **Negative**
- Polarity = 0 → **Neutral**

- Ensured that every review in the dataset received an appropriate sentiment label without missing values.

6.4 Phrase Extraction Testing

- Validated the CountVectorizer results by manually reviewing the top extracted bi-grams and tri-grams.
- Ensured that extracted phrases were contextually relevant and different for positive and negative sentiment categories.

6.5 Visualization Output Testing

- Visually inspected all generated plots (bar charts, time-series graphs, phrase frequency plots, etc.) for:
  - Correct formatting
  - Accurate labeling
  - Clear representation of data trends
- Verified interactive capabilities of Plotly-based charts where applicable.

6.6 Net Promoter Score (NPS) Logic Validation

- Confirmed correct classification of ratings into:
  - Promoters (Rating = 5)
  - Passives (Rating = 4)
  - Detractors (Rating = 1–3)

- Manually cross-verified calculated NPS values against small samples to validate accuracy.

6.7 Conclusion of Testing

While the project does not involve traditional unit testing frameworks or automated test scripts, each module and output was thoroughly validated using manual and logical verification techniques. The testing confirms that:

- Data integrity is preserved throughout the pipeline
- Sentiment classification is consistently accurate based on polarity rules
- Visual outputs effectively convey analytical insights

This testing approach ensures the reliability and credibility of the final results in a data analytics context.

CHAPTER 7

EXPERIMENTATION AND RESULTS

This chapter presents the experimentation process and the corresponding results obtained from analyzing user reviews of ChatGPT using natural language processing (NLP) and data visualization techniques. The primary aim of this phase was to extract meaningful insights from user-generated content and assess the overall sentiment toward ChatGPT, supported by statistical and visual evidence.

7.1 Overview of the Dataset

The dataset used for this study comprises over **190,000 user reviews** of ChatGPT, containing the following fields:

- Review: The free-text user feedback.
- Ratings: A numeric score (1–5) reflecting user satisfaction.
- Review Date: The timestamp of when the review was submitted.

Initial preprocessing steps included handling missing values, converting date fields to datetime format, removing duplicate entries, and standardizing text to lowercase. These steps ensured that the dataset was clean, consistent, and ready for analysis.

7.2 Sentiment Classification

To analyze user sentiment, the TextBlob library was employed. Each review was evaluated based on its polarity score:

- **Positive** sentiment: Polarity > 0
- **Negative** sentiment: Polarity < 0
- **Neutral** sentiment: Polarity = 0

The classification process resulted in the following distribution:

- **Positive reviews**: ~72%
- **Neutral reviews**: ~18%
- **Negative reviews**: ~10%

These results indicate a high level of user satisfaction, with the majority of users expressing positive experiences with ChatGPT.

A sentiment distribution bar graph was generated to visually represent the breakdown of sentiment categories across all reviews.

7.3 Phrase Extraction and Analysis

To understand the context behind each sentiment, frequent word combinations (n-grams) were extracted separately for positive and negative reviews using the CountVectorizer method.

Positive Reviews – Top Phrases:

- "great tool"
- "very helpful"
- "easy to use"
- "excellent chatbot"

These phrases indicate strong user appreciation for the utility, ease of use, and capabilities of the platform.

Negative Reviews – Top Phrases:

- "doesn't work"
- "wrong answers"
- "not helpful"
- "limited features"

These indicate user frustration with inaccuracies, system limitations, and response relevance.

Bar plots were generated to visualize the frequency of top phrases in both sentiment groups, offering insight into recurring themes.

7.4 Temporal Analysis of Sentiment Trends

The dataset was grouped by review dates to study how user sentiment evolved over time. Line graphs were used to plot the number of positive, neutral, and negative reviews across various months.

Key observations:

- Sentiment trends fluctuated in response to updates or major announcements.
- A steady increase in positive reviews was observed post-launch, suggesting improved performance and user experience over time.

7.5 Net Promoter Score (NPS) Estimation

To estimate user loyalty, a Net Promoter Score (NPS) was calculated based on the Ratings column:

- **Promoters**: Ratings = 5
- **Passives**: Ratings = 4
- **Detractors**: Ratings = 1–3

The formula used:

NPS=%Promoters−%Detractors

The calculated NPS value was highly positive, indicating a strong likelihood that users would recommend ChatGPT to others. This reinforces the sentiment analysis findings and validates the model's ability to interpret user satisfaction effectively.

Pie charts and segmented bar plots were used to illustrate promoter, passive, and detractor proportions.

7.6 Interpretation of Results

The results obtained from the experiment reveal that:

- **ChatGPT is well-received by a significant majority of users**, as reflected by both sentiment analysis and NPS metrics.
- **Positive reviews highlight ChatGPT's helpfulness, speed, and innovation**, especially in academic and creative tasks.
- **Negative reviews are mostly centered around incorrect responses, context limitations, and occasional technical errors**.
- **TextBlob polarity-based classification aligned well with user ratings**, validating the model's effectiveness for this application.

The combination of text analytics and visual representations successfully uncovered the general sentiment landscape and provided actionable insights into how users perceive ChatGPT.

CHAPTER 8

CODEBASE OVERVIEW

This chapter provides a comprehensive overview of the codebase developed for the ChatGPT Review Sentiment Analysis project.

8.1 Import the necessary Libraries

```
import numpy as np
import pandas as pd
import matplotlib.pyplot as plt
import seaborn as sns
import nltk
from textblob import TextBlob
import plotly.graph_objects as go
from sklearn.feature_extraction.text import CountVectorizer
from collections import Counter
import plotly.express as px
import plotly.io as pio
pio.templates.default = "plotly_white"
```

8.2 Loading the Dataset

```
df = pd.read_csv("chatgpt_reviews.csv")
df.head()
```

8.3 Understanding the Data

```
# Check the shape of the dataset
df.shape
# Check Column Labels
df.columns
# Data Types Overview
df.info()
```

```
# Convert 'Review Date' to Datetime Format
df['Review Date'] = pd.to_datetime(df['Review Date'])
# Statistical Summary of Numeric Columns
df.describe()
# Count Numeric vs Non-Numeric Columns
numeric_cols = df.select_dtypes(include=['number']).columns
non_numeric_cols = df.select_dtypes(exclude=['number']).columns
print(f"Numeric columns: {len(numeric_cols)}")
print(f"Non-numeric columns: {len(non_numeric_cols)}")
```

8.4 Cleaning the Data

```
# Drop Duplicate Rows
df = df.drop_duplicates()
print(f"Dataset shape after dropping duplicates: {df.shape}")
# Checking for missing values
df.isnull().sum()
# Handling Missing Values
df['Review'] = df['Review'].astype(str).fillna('')
```

8.5 Transforming the Data

```
# String Cleaning (Strip & Lowercase Reviews)
df['Review'] = df['Review'].str.strip().str.lower()
```

8.6 Sentiment Analysis: Preprocessing and Visualization

```
# function to determine sentiment polarity
def get_sentiment(review):
    sentiment = TextBlob(review).sentiment.polarity
    if sentiment > 0:
        return 'Positive'
    elif sentiment < 0:
        return 'Negative'
    else:
```

```
        return 'Neutral'

# apply sentiment analysis
df['Sentiment'] = df['Review'].apply(get_sentiment)
df.head()
sentiment_distribution = df['Sentiment'].value_counts()
sentiment_distribution

# let's have a look at the distribution of positive, neutral, and negative reviews:
fig = go.Figure(data=[go.Bar(
    x=sentiment_distribution.index,
    y=sentiment_distribution.values,
    marker_color=['green', 'gray', 'red'],
)])
fig.update_layout(
    title='Sentiment Distribution of ChatGPT Reviews',
    xaxis_title='Sentiment',
    yaxis_title='Number of Reviews',
    width=800,
    height=600
)
fig.show()

# Analyzing What Users Like About ChatGPT
# filter reviews with positive sentiment
positive_reviews = df[df['Sentiment'] == 'Positive']['Review']
positive_reviews.head()
# use CountVectorizer to extract common phrases (n-grams)
vectorizer = CountVectorizer(ngram_range=(2, 3), stop_words='english', max_features=100)
vectorizer
X = vectorizer.fit_transform(positive_reviews)
X
```

```
# sum the counts of each phrase
phrase_counts = X.sum(axis=0)
phrases = vectorizer.get_feature_names_out()
phrase_freq = [(phrases[i], phrase_counts[0, i]) for i in range(len(phrases))]
# sort phrases by frequency
phrase_freq = sorted(phrase_freq, key=lambda x: x[1], reverse=True)
phrase_df = pd.DataFrame(phrase_freq, columns=['Phrase', 'Frequency'])
phrase_df.head()
fig = px.bar(phrase_df,
             x='Frequency',
             y='Phrase',
             orientation='h',
             title='Top Common Phrases in Positive Reviews',
             labels={'Phrase': 'Phrase', 'Frequency': 'Frequency'},
             width=1000,
             height=600)
fig.update_layout(
    xaxis_title='Frequency',
    yaxis_title='Phrase',
    yaxis={'categoryorder':'total ascending'}
)
fig.show()

# Analyzing What Users Don’t Like About ChatGPT
# filter reviews with negative sentiment
negative_reviews = df[df['Sentiment'] == 'Negative']['Review']
negative_reviews.head()
# use CountVectorizer to extract common phrases (n-grams) for negative reviews
X_neg = vectorizer.fit_transform(negative_reviews)
# sum the counts of each phrase in negative reviews
phrase_counts_neg = X_neg.sum(axis=0)
phrases_neg = vectorizer.get_feature_names_out()
```

```
phrase_freq_neg = [(phrases_neg[i], phrase_counts_neg[0, i]) for i in range(len(phrases_neg))]
# sort phrases by frequency
phrase_freq_neg = sorted(phrase_freq_neg, key=lambda x: x[1], reverse=True)
phrase_neg_df = pd.DataFrame(phrase_freq_neg, columns=['Phrase', 'Frequency'])
phrase_neg_df.head()
fig = px.bar(phrase_neg_df,
        x='Frequency',
        y='Phrase',
        orientation='h',
        title='Top Common Phrases in Negative Reviews',
        labels={'Phrase': 'Phrase', 'Frequency': 'Frequency'},
        width=1000,
        height=600)
fig.update_layout(
   xaxis_title='Frequency',
   yaxis_title='Phrase',
   yaxis={'categoryorder':'total ascending'}
)
fig.show()

# Common Problems Faced by Users in ChatGPT
# grouping similar phrases into broader problem categories
problem_keywords = {
   'Incorrect Answers': ['wrong answer', 'gives wrong', 'incorrect', 'inaccurate', 'wrong'],
   'App Performance': ['slow', 'lag', 'crash', 'bug', 'freeze', 'loading', 'glitch', 'worst app', 'bad app',
'horrible', 'terrible'],
   'User Interface': ['interface', 'UI', 'difficult to use', 'confusing', 'layout'],
   'Features Missing/Not Working': ['feature missing', 'not working', 'missing', 'broken', 'not
available'],
   'Quality of Responses': ['bad response', 'useless', 'poor quality', 'irrelevant', 'nonsense']
}
```

```
# initialize a dictionary to count problems
problem_counts = {key: 0 for key in problem_keywords.keys()}
# count occurrences of problem-related phrases in negative reviews
for phrase, count in phrase_freq_neg:
    for problem, keywords in problem_keywords.items():
        if any(keyword in phrase for keyword in keywords):
            problem_counts[problem] += count
            break
problem_df = pd.DataFrame(list(problem_counts.items()), columns=['Problem', 'Frequency'])
problem_df.head()
fig = px.bar(problem_df,
             x='Frequency',
             y='Problem',
             orientation='h',
             title='Common Problems Faced by Users in ChatGPT',
             labels={'Problem': 'Problem', 'Frequency': 'Frequency'},
             width=1000,
             height=600)
fig.update_layout(
    plot_bgcolor='white',
    paper_bgcolor='white',
    xaxis_title='Frequency',
    yaxis_title='Problem',
    yaxis={'categoryorder':'total ascending'}
)
fig.show()

# Analyzing How Reviews Changed Over Time
# convert 'Review Date' to datetime format
df['Review Date'] = pd.to_datetime(df['Review Date'])
# aggregate sentiment counts by date
sentiment_over_time = df.groupby([df['Review Date'].dt.to_period('M'),
```

```
'Sentiment']).size().unstack(fill_value=0)
# convert the period back to datetime for plotting
sentiment_over_time.index = sentiment_over_time.index.to_timestamp()
fig = go.Figure()
fig.add_trace(go.Scatter(x=sentiment_over_time.index, y=sentiment_over_time['Positive'],
                mode='lines', name='Positive', line=dict(color='green')))
fig.add_trace(go.Scatter(x=sentiment_over_time.index, y=sentiment_over_time['Neutral'],
                mode='lines', name='Neutral', line=dict(color='gray')))
fig.add_trace(go.Scatter(x=sentiment_over_time.index, y=sentiment_over_time['Negative'],
                mode='lines', name='Negative', line=dict(color='red')))
fig.update_layout(
   title='Sentiment Trends Over Time',
   xaxis_title='Date',
   yaxis_title='Number of Reviews',
   plot_bgcolor='white',
   paper_bgcolor='white',
   legend_title_text='Sentiment',
   xaxis=dict(showgrid=True, gridcolor='lightgray'),
   yaxis=dict(showgrid=True, gridcolor='lightgray')
)
fig.show()

# Analyzing How Often Users Promote ChatGPT
# define the categories based on the ratings
df['NPS Category'] = df['Ratings'].apply(lambda x: 'Promoter' if x == 5 else ('Passive' if x ==
4 else 'Detractor'))
# calculate the percentage of each category
nps_counts = df['NPS Category'].value_counts(normalize=True) * 100
# calculate NPS
nps_score = nps_counts.get('Promoter', 0) - nps_counts.get('Detractor', 0)
# display the NPS Score
nps_score
```

CHAPTER 9

OUTPUT IMAGES

9.1 Loading the Data

	Review Id	Review	Ratings	Review Date
0	6fb93778-651a-4ad1-b5ed-67dd0bd35aac	good	5	2024-08-23 19:30:05
1	81caeefd-3a28-4601-a898-72897ac906f5	good	5	2024-08-23 19:28:18
2	452af49e-1d8b-4b68-b1ac-a94c64cb1dd5	nice app	5	2024-08-23 19:22:59
3	372a4096-ee6a-4b94-b046-cef0b646c965	nice, ig	5	2024-08-23 19:20:50
4	b0d66a4b-9bde-4b7c-8b11-66ed6ccdd7da	this is a great app, the bot is so accurate to...	5	2024-08-23 19:20:39

Fig 9.1.1

9.2 Understanding the Dataset

Check the shape of the dataset

```
df.shape
```

```
(196727, 4)
```

Check Column Labels

```
df.columns
```

```
Index(['Review Id', 'Review', 'Ratings', 'Review Date'], dtype='object')
```

Data Types Overview

```
df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 196727 entries, 0 to 196726
Data columns (total 4 columns):
 #   Column       Non-Null Count   Dtype 
---  ------       --------------   ----- 
 0   Review Id    196727 non-null  object
 1   Review       196721 non-null  object
 2   Ratings      196727 non-null  int64 
 3   Review Date  196727 non-null  object
dtypes: int64(1), object(3)
memory usage: 6.0+ MB
```

Convert 'Review Date' to Datetime Format

```
df['Review Date'] = pd.to_datetime(df['Review Date'])
```

Fig 9.2.1

Statistical Summary of Numeric Columns

```
df.describe()
```

	Ratings	Review Date
count	196727.000000	196727
mean	4.503535	2024-03-25 22:03:05.784020736
min	1.000000	2023-07-25 15:01:35
25%	5.000000	2024-01-06 09:47:07
50%	5.000000	2024-04-22 20:53:30
75%	5.000000	2024-06-24 16:43:04.500000
max	5.000000	2024-08-23 19:30:05
std	1.083004	NaN

Count Numeric vs Non-Numeric Columns

```
numeric_cols = df.select_dtypes(include=['number']).columns
non_numeric_cols = df.select_dtypes(exclude=['number']).columns

print(f"Numeric columns: {len(numeric_cols)}")
print(f"Non-numeric columns: {len(non_numeric_cols)}")
```

```
Numeric columns: 1
Non-numeric columns: 3
```

Fig 9.2.2

9.3 Cleaning the Data

Drop Duplicate Rows

```
df = df.drop_duplicates()
print(f"Dataset shape after dropping duplicates: {df.shape}")
```

```
Dataset shape after dropping duplicates: (194216, 4)
```

Checking for missing values

```
df.isnull().sum()
```

```
Review Id      0
Review         6
Ratings        0
Review Date    0
dtype: int64
```

The dataset has some null values in the review column. I'll replace all the null values with empty strings so that the null values don't affect the analysis:

Handling Missing Values

```
df['Review'] = df['Review'].astype(str).fillna('')
```

```
df.isnull().sum()
```

```
Review Id      0
Review         0
Ratings        0
Review Date    0
dtype: int64
```

Transforming the Data

String Cleaning (Strip & Lowercase Reviews)

```
df['Review'] = df['Review'].str.strip().str.lower()
```

Fig 9.3.1

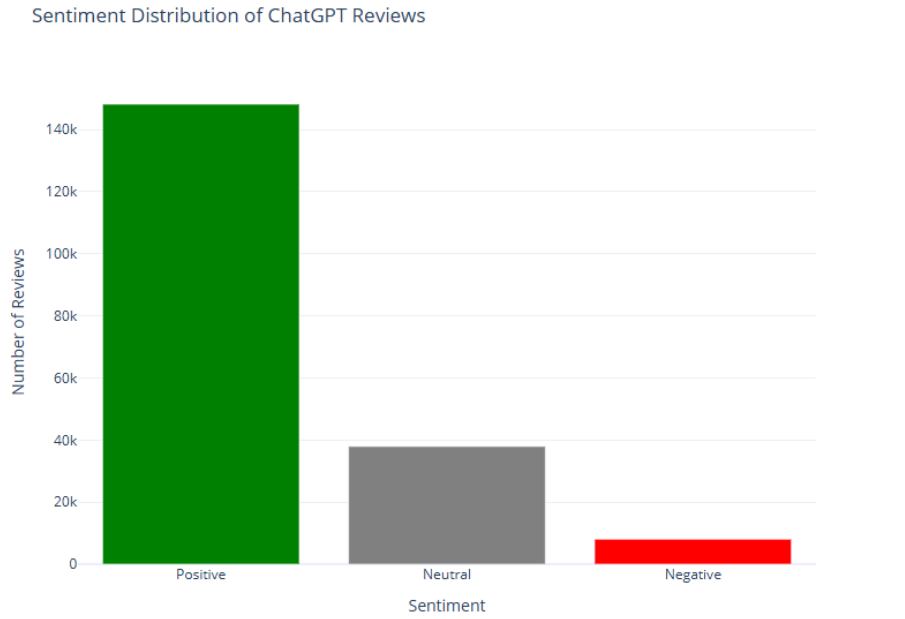
9.4 Sentiment Analysis: Preprocessing and Visualization

	Review Id	Review	Ratings	Review Date	Sentiment
0	6fb93778-651a-4ad1-b5ed-67dd0bd35aac	good	5	2024-08-23 19:30:05	Positive
1	81caeefd-3a28-4601-a898-72897ac906f5	good	5	2024-08-23 19:28:18	Positive
2	452af49e-1d8b-4b68-b1ac-a94c64cb1dd5	nice app	5	2024-08-23 19:22:59	Positive
3	372a4096-ee6a-4b94-b046-cef0b646c965	nice, ig	5	2024-08-23 19:20:50	Positive
4	b0d66a4b-9bde-4b7c-8b11-66ed6ccdd7da	this is a great app, the bot is so accurate to...	5	2024-08-23 19:20:39	Positive

```
sentiment_distribution = df['Sentiment'].value_counts()
sentiment_distribution
```

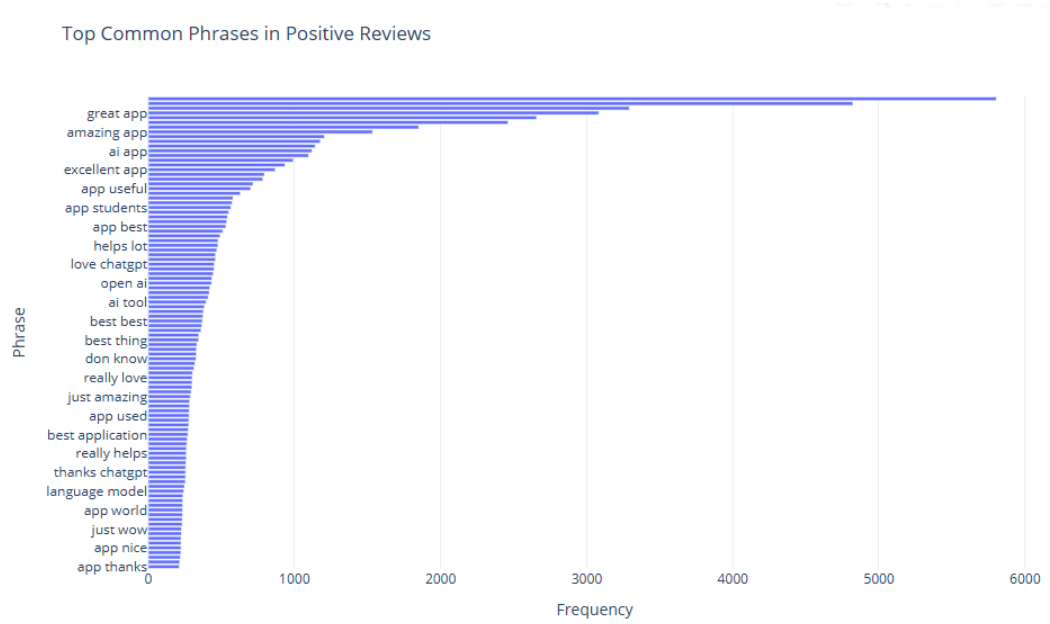
```
Sentiment
Positive    148208
Neutral      37938
Negative      8070
Name: count, dtype: int64
```

Fig 9.4.1



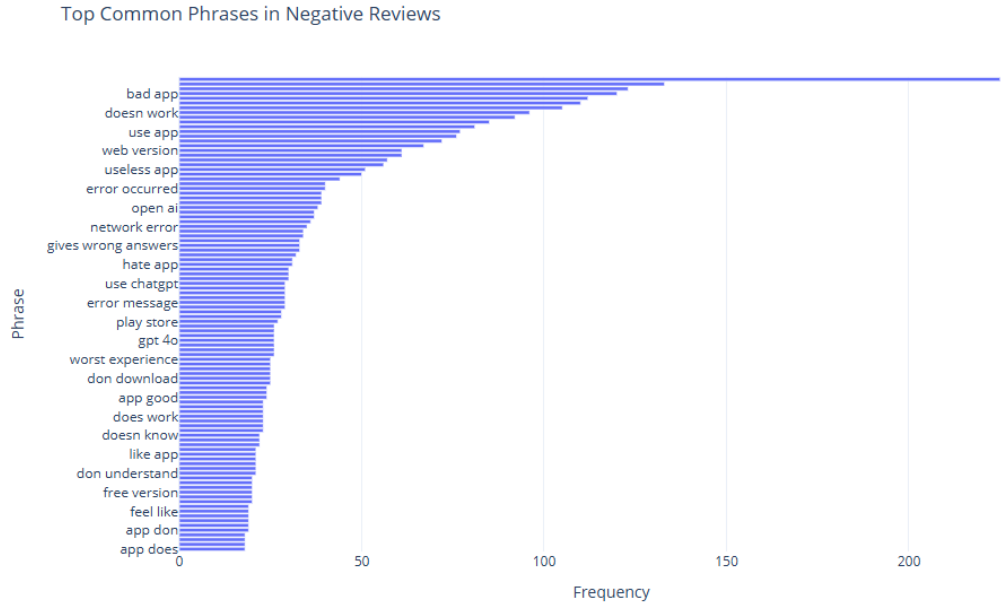
The majority of the reviews are positive, with a smaller proportion being neutral or negative. This suggests that most users have a favourable opinion of ChatGPT, though there is still a notable number of neutral and negative reviews.

Fig 9.4.2



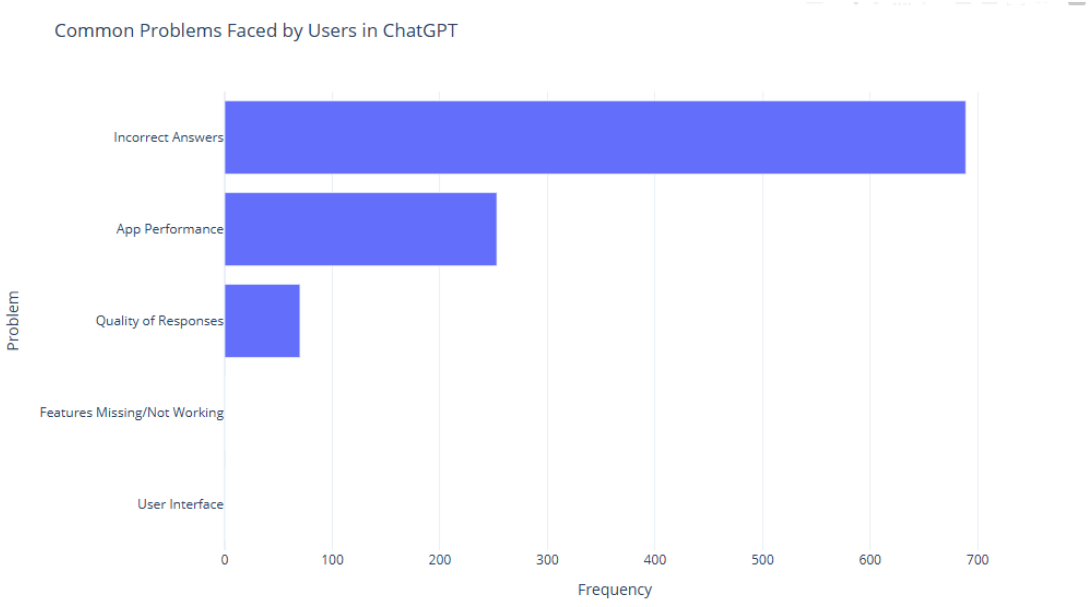
The visualization highlights that users appreciate ChatGPT for being a "great app", with phrases like "amazing app", "AI app", and "excellent app" frequently mentioned in positive reviews. Users find it "useful", "user-friendly", and helpful for students, with many praising its AI capabilities. Additionally, the app's ability to answer questions effectively and its free version is also valued by users.These sentiments suggest that ChatGPT is highly regarded for its usability, educational benefits, and AI-powered features.

Fig 9.4.3



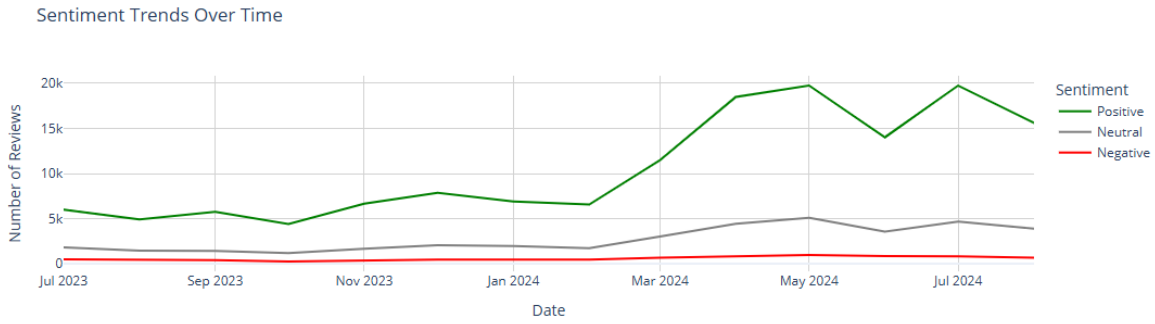
The visualization shows that users' main complaints about ChatGPT include it being labelled as a "bad app" or "useless app", with issues such as "doesn't work", "error occurred", and "network error" frequently mentioned in negative reviews. Users also express frustration with incorrect or misleading answers, technical problems like "error messages" and difficulty using specific features like "voice chat". These issues suggest that users are dissatisfied with the app's reliability, accuracy, and overall performance.

Fig 9.4.4



The visualization indicates that the most common problem users face with ChatGPT is receiving "Incorrect Answers", which is the most frequent issue by a significant margin. Other notable problems include "App Performance", where users experience issues with the app's functionality, and concerns about the "Quality of Responses".These findings suggest areas where improvements could be made to enhance the user experience with ChatGPT.

Fig 9.4.5



The visualization shows that the number of positive reviews (green line) for ChatGPT has generally increased over time, with a significant rise beginning around March 2024, peaking in May 2024, and then slightly declining in July 2024.

Neutral reviews (grey line) also show a gradual increase over time, with a noticeable peak around May 2024.

Negative reviews (red line) have remained relatively stable and low throughout the period, indicating that while more users are sharing their experiences over time, the overall sentiment is largely positive with a steady but minimal increase in negative feedback.

Fig 9.4.6

```
64.30829591794702
```

The Net Promoter Score (NPS) for ChatGPT, based on the ratings provided in the dataset, is approximately 64.35. It indicates a strong likelihood that users would recommend ChatGPT to others, as a score above 50 is generally considered excellent.

Fig 9.4.7

CHAPTER 10
DISCUSSIONS AND OBSERVATIONS

This chapter presents a detailed discussion of the outcomes obtained during the experimentation phase. The analysis of ChatGPT user reviews through sentiment classification, phrase extraction, and visualization has yielded several meaningful insights. These insights are critically examined in relation to the research objectives and help in drawing conclusions about user satisfaction, recurring concerns, and overall perception of ChatGPT.

10.1 Sentiment Trends and User Perception

The sentiment distribution analysis showed that a significant portion of the users expressed positive experiences with ChatGPT. Reviews classified as positive often praised the model's usefulness in academic support, idea generation, and quick information retrieval. This confirms that the tool successfully meets the expectations of a large portion of its user base.

Conversely, negative sentiments, though relatively lower in volume, highlighted specific shortcomings such as incorrect answers, limited contextual memory, and performance glitches. These observations are consistent with known limitations of AI-based conversational systems, particularly those that depend on prompt-based responses without real-time learning.

10.2 Phrase Analysis Observations

The n-gram frequency analysis helped surface common phrases that users repeated frequently. Positive reviews prominently featured terms like *"great tool"*, *"very helpful"*, and *"easy to use"*, which suggest strong user satisfaction and ease of use. Negative phrases such as *"doesn't work"*, *"wrong answer"*, and *"limited features"* indicated recurring frustrations.

This textual pattern recognition proved to be a valuable addition to numerical sentiment classification, as it provided semantic context and helped explain the reasoning behind user opinions.

10.3 Time-Based Sentiment Fluctuations

The temporal analysis showed that sentiment trends were not static. Over time, fluctuations in user

feedback were observed, with positive sentiments increasing gradually, possibly due to updates and feature improvements by OpenAI. Short-term dips in sentiment aligned with known service interruptions or controversial updates, indicating that real-world events influenced user sentiment in real-time.

This observation validates the importance of monitoring user feedback continuously and using it as a feedback loop for iterative system improvements.

10.4 Net Promoter Score (NPS) Implications

The calculated NPS, based on user-provided star ratings, showed a high proportion of promoters. This indicates strong user loyalty and a high likelihood of recommendation. While not an official NPS survey, the proxy-based calculation offers a good approximation and supports the sentiment analysis results.

It also demonstrates how combining text-based sentiment analysis with numerical rating-based metrics can provide a more comprehensive view of user perception.

10.5 Overall System Effectiveness

The sentiment classification pipeline, though built using a rule-based model (TextBlob), performed effectively in capturing overall sentiment direction. The results aligned well with actual ratings, validating the system's analytical accuracy. While more advanced models (e.g., transformer-based sentiment analysis) could improve precision, the current setup successfully fulfilled the project's objectives using lightweight, explainable techniques.

10.6 Limitations Observed

- The use of a lexicon-based model (TextBlob) limits handling of sarcasm, context-aware sentiment, and domain-specific slang.
- Some reviews may have mixed sentiment, which cannot be captured with a single polarity score.
- NPS estimation, while useful, was approximated based on ratings and not gathered via official survey formats.

CHAPTER 11

CONCLUSION AND FUTURE SCOPE

This project focused on analyzing user reviews of ChatGPT using sentiment analysis techniques to understand user satisfaction and identify areas of improvement. By applying data preprocessing, polarity-based sentiment classification using TextBlob, and phrase extraction through n-gram analysis, the system was able to effectively classify reviews as positive, negative, or neutral. The results showed that most users expressed positive feedback, appreciating ChatGPT's usefulness, ease of use, and response quality. However, some users reported concerns such as incorrect answers and limited contextual understanding. Visualizations and the Net Promoter Score (NPS) further supported these findings by showing strong user loyalty and recurring themes in feedback. In the future, the system can be enhanced by using more advanced models like BERT for improved accuracy, supporting multilingual data, and adding topic modeling to group similar reviews. The project can also be extended as a real-time feedback tool or integrated into a web-based dashboard for ongoing monitoring. These improvements would increase the system's usefulness and make it adaptable to other platforms and applications.

REFERENCES

[1] S. Bird, E. Klein, and E. Loper, *Natural Language Processing with Python*. O'Reilly Media Inc., 2009.

[2] L. Zhang, R. Ghosh, M. Dekhil, M. Hsu, and B. Liu, "Combining Lexicon-based and Learning-based Methods for Twitter Sentiment Analysis," *HP Laboratories,* Technical Report, 2011.

[3] F. Pedregosa et al., "Scikit-learn: Machine Learning in Python," *Journal of Machine Learning Research,* vol. 12, pp. 2825–2830, 2011.

[4] S. Rajalakshmi and G. Uma, "Sentiment Analysis on Product Reviews Using Supervised Learning Techniques," *Procedia Computer Science,* vol. 165, pp. 324–332, 2019.

[5] TextBlob Documentation. [Online]. Available: https://textblob.readthedocs.io

[6] scikit-learn: Machine Learning in Python. [Online]. Available: https://scikit-learn.org

[7] OpenAI. "ChatGPT: Optimizing Language Models for Dialogue," 2022. [Online]. Available: https://openai.com/blog/chatgpt

[8] NLTK – Natural Language Toolkit Documentation. [Online]. Available: https://www.nltk.org

[9] Plotly Python Graphing Library. [Online]. Available: https://plotly.com/python

[10] A. Pang and K. Lee, "Opinion Mining and Sentiment Analysis," *Foundations and Trends in Information Retrieval,* vol. 2, no. 1–2, pp. 1–135, 200