International Centre for Education and Research (ICER)
VIT-Bangalore

Analyzing Customers Preferences in Online Food Delivery Systems

CS6510 – PROJECT 1

REPORT

Submitted by

Jayanth Kumar – 24MSP3073

In partial fulfilment for the award of the degree of

POST GRADUATE PROGRAMME

INTERNATIONAL CENTRE FOR HIGHER EDUCATION AND RESEARCH

VIT BANGALORE

DECEMBER, 2024

International Centre for Education and Research (ICER)
VIT-Bangalore

BONAFIDE CERTIFICATE

Certified that this project report **"Analyzing Customers Preferences in Online Food Delivery Systems"** is the bonafide record of work done by **"Jayanth Kumar – 24MSP3073"** who carried out the project work under my supervision.

Signature of the Supervisor

Prof. Ramya Mohanakrishnan

Assistant Professor,

ICER

VIT Bangalore

Signature of Director

Prof. Prema M

Director,

ICER

VIT Bangalore.

Evaluation Date:

International Centre for Education and Research (ICER)
VIT-Bangalore

ACKNOWLEDGEMENT

I would like take this opportunity to express my sincere gratitude to our director of ICER **Prof. Prema M** for providing me an opportunity to widen my knowledge in field of computer science. It would take an extraordinary amount of time for me to personally recognise and appreciate each and every individual who has contributed to this project, whether directly or indirectly. A great deal of gratitude goes out to the numerous people whose cooperation and encouragement have made my job significantly simpler. **Prof. Ramya Mohanakrishnan**, ICER has provided me with an excellent opportunity to learn about my area of interest, without her continuous guidance and persistent help, this project would not have been a success for me. I am are grateful to show my genuine appreciation to her. Taking this opportunity to offer my appreciation to our parents and friends for their support and encouragement throughout the project has been a wonderful honour.

TABLE OF CONTENTS

Contents

ABSTRACT

Despite the popularity of online food delivery systems in the foodservice industry, there have been few studies into customers decision-making process to use online food delivery services. This project explores consumer preferences, satisfaction factors, and behavioral patterns in online food delivery services. Using a dataset of 388 records, the analysis leverages Python to identify key drivers influencing consumer choices, including demographics, service quality, and delivery efficiency. Techniques like exploratory data analysis, visualization, machine learning, text analysis and sentiment analysis will be employed to uncover insights. The findings aim to assist food delivery platforms in optimizing user experiences by addressing pain points like late delivery, poor hygiene, and missing items while enhancing factors like ease, quality, and packaging. This study offers actionable recommendations for businesses to improve customer retention and satisfaction in the competitive online food delivery market.

Keywords: *exploratory data analysis, machine learning, text analysis, sentiment analysis*

LIST OF FIGURES

LIST OF TABLES

CHAPTER 1
INTRODUCTION

1.1 Project Background

E-commerce is growing around the world while the digital economy is growing and as more people are getting access to high-speed Internet. Given the increasing importance of Online Food Delivery (OFD), there are many research opportunities, including the development of new data analytics methodologies for analyzing data from OFD. In this study, developed and demonstrated the applicability of such a data analytics methodology, which consists mainly of classification and regression analysis to predict the key performance indicators (KPIs) in OFD. This research has compared 5 algorithms and performed both textual and sentiment analysis.

1.2 Scope of Study

This study focuses on developing a robust sentiment analysis model utilizing a dataset of user reviews. The dataset comprises user-generated comments labelled with binary outputs ("Yes" and "No"), which presumably indicate positive and negative sentiments, respectively. The primary goal of this research is to construct a predictive model capable of accurately classifying sentiments based on the textual content of the reviews.

The methodology employs traditional machine learning techniques, particularly the Naive Bayes classifier, due to its efficiency and proven effectiveness in text classification tasks. To enhance the model's performance, various preprocessing techniques, such as text cleaning, stop word removal, and lemmatization, are applied to normalize the data. The research further addresses challenges such as data imbalance through oversampling techniques like Synthetic Minority Oversampling Technique (SMOTE). Performance evaluation is carried out using cross-validation to ensure model reliability and robustness.

The key findings of this study aim to contribute significant advancements to the online food delivery industry. By fostering improved customer satisfaction, retention, and facilitating sustained growth in this dynamic market landscape, the research endeavours to address critical challenges and pave the way for enhanced customer satisfaction to order food online.

1.3 Problem Statement

This study aims to address the gap in understanding the key factors that drive customer preferences in online food delivery systems.

By analyzing data on customer behaviours, experiences, and preferences, the research seeks to identify patterns and insights that can help food delivery platforms enhance their services.

1.4 Need of Study

- This project helps us to learn the consumer preferences and perceptions of online food delivery amenities.
- This project assesses the numerous elements that impact the consumer's decisions to place an order for food online.
- Organizations can use the sentiment analysis model to understand customer feedback, enabling them to make informed decisions about products and services.
- The project provides a stepping stone for developing interactive systems like chatbots, recommendation engines, or feedback analyzers.

1.5 Objectives

- The objective of this study is to analyze customer preferences and behaviours in online food delivery systems to identify the key factors influencing customer satisfaction and decision-making.
- By leveraging data-driven insights, the study aims to provide actionable recommendations for improving service quality, enhancing user experience, and optimizing operational efficiency for food delivery platforms.
- To successfully build a robust ML model and a user-interface to predict the sentiment of the customers.

CHAPTER-2

Literature Review

2.1 Previous Work

This section talks about the other previous works on online food delivery preferences. This paper [1], is an extension of previous research and the dataset was web scraped from an open-source website. The authors developed and demonstrated the applicability of such a data analytics methodology, which consists mainly of classification and regression analysis to predict the key performance indicators (KPIs) in online food delivery (OFD). This paper compared 11 classification algorithms and four regression algorithms as part of their research. Limitation of the research was, the data was scraped from a random website. The data had no proper features for prediction by which the author had to formulate his own features to perform regression.

In this paper [2], the authors present a reptile search algorithm with a hybrid DL-based UX detection (RSAHDL-UXD) technique on OFDSs. The primary objective of the RSAHDL-UXD system is to identify and classify UX. To accomplish this, the RSAHDL-UXD technique undergoes data pre-processing to convert input data into a beneficial layout, and the word2vec method is applied to the word embedding process. For UX detection, the sliced multi-head self-attention slice recurrent neural network (SMH-SASRNN) model. The hyperparameter tuning process was executed using RSA to enhance the detection results of the SMH-SASRNN system. To validate the amended performance of the RSAHDL-UXD methodology, many simulations have been executed on two online food services datasets.

This study [3], meticulously explores predictive models for accurate food delivery time estimation. The project involves comprehensive data analysis, including data cleaning, feature engineering, and rigorous model evaluation, resulting in a robust predictive model optimized through parameter tuning. The encoding of categorical variables enhances model effectiveness, with evaluation metrics emphasizing the model's accuracy. Notably, the project introduces innovative approaches, such as geospatial distance calculations for determining the physical distance between the restaurant and the delivery location, contributing valuable insights to optimize food delivery predictions in the dynamic online food delivery industry. Limitation of the models are highly dependent on the quality, completeness, and representativeness of historical data. Biases or missing data could reduce model accuracy unexpected weather conditions, traffic accidents, or road blockages are challenging to predict and might not be fully accounted for in the model.

In this research [4], the authors investigate the dispatching problem arising in the real food delivery scenario of Meituan platform and address an online food delivery problem (OFDP). To deal with high dynamism and urgency, the OFDP is transformed into a static optimization problem within a time window. A matching algorithm with an adaptive tie-breaking strategy (MAATS) is proposed to effectively solve the OFDP by determining the matching relationship between newly arrived orders and riders. The MAATS is mainly

composed of a best-matching heuristic, multiple tie-breaking operators, and a machine learning (ML) model.

2.2 Comparative Work

S.No	Title of the Paper	Authors	Journal Name/Conference Title & Publisher	Year of Publication	Methodology used	Performance of the proposed algorithm
1.	A Predictive Data Analytics Methodology for Online Food Delivery [1] [Base paper]	Mariam Al Akasheh, Nehal Eleyan, Gurdal Ertek	IEEE Explore	2022	Random Forest, Linear Regression, Neural Network, Decision Tree	97%, 2%, 12%, 97%
2.	Enhancing Online Food Service User Experience Through Advanced Analytics and Hybrid Deep Learning for Comprehensive Evaluation [2]	HUSSAIN ALSHAHRANI, HANAN ABDULLAH MENGASH, MASHAEL MAASHI, FADOUA KOUKI, AHMED MAHMUD, MESFER AL DUHAYYIM	IEEE Access	2024	RSAHDL-UXD model, CNN, Linear Regression, Neural Networks	98.57%, 94.92%, 86.15%, 87.51%
3.	Forecasting Food Delivery Time: An Exploration of Predictive Models and Factors Impacting Delivery Time Estimation [3]	Dr. Ayyappa Chakravarthi M, Shaik Eesa Ruhulla Haq, Madapakula Venkata Anil, Bathula Venkata Vamsi, Gogireddy Venkata Reddy	IEEE Explore		Linear Regression, Random Forest Regressor, AdaBoost Regressor, KNeighbors Regressor, Decision Tree Regressor, Gradient Boosting Regressor, XGB Regressor, LGBM Regressor.	41.8% 80.9% 57%, 49.5% 64.9%, 76.7%, 81.44%, 82.17%
4.	An effective matching algorithm with adaptive tie-breaking strategy for online food delivery problem	Jing-fang Chen, Ling Wang, Shengyao Wang, Xing Wang, Hao Ren	IEEE Explore	2021	MAATS framework,	91.8%

Table 2.2: Comparative work of previous study

CHAPTER-3

Methodology

3.1 Design/Architecture Diagram

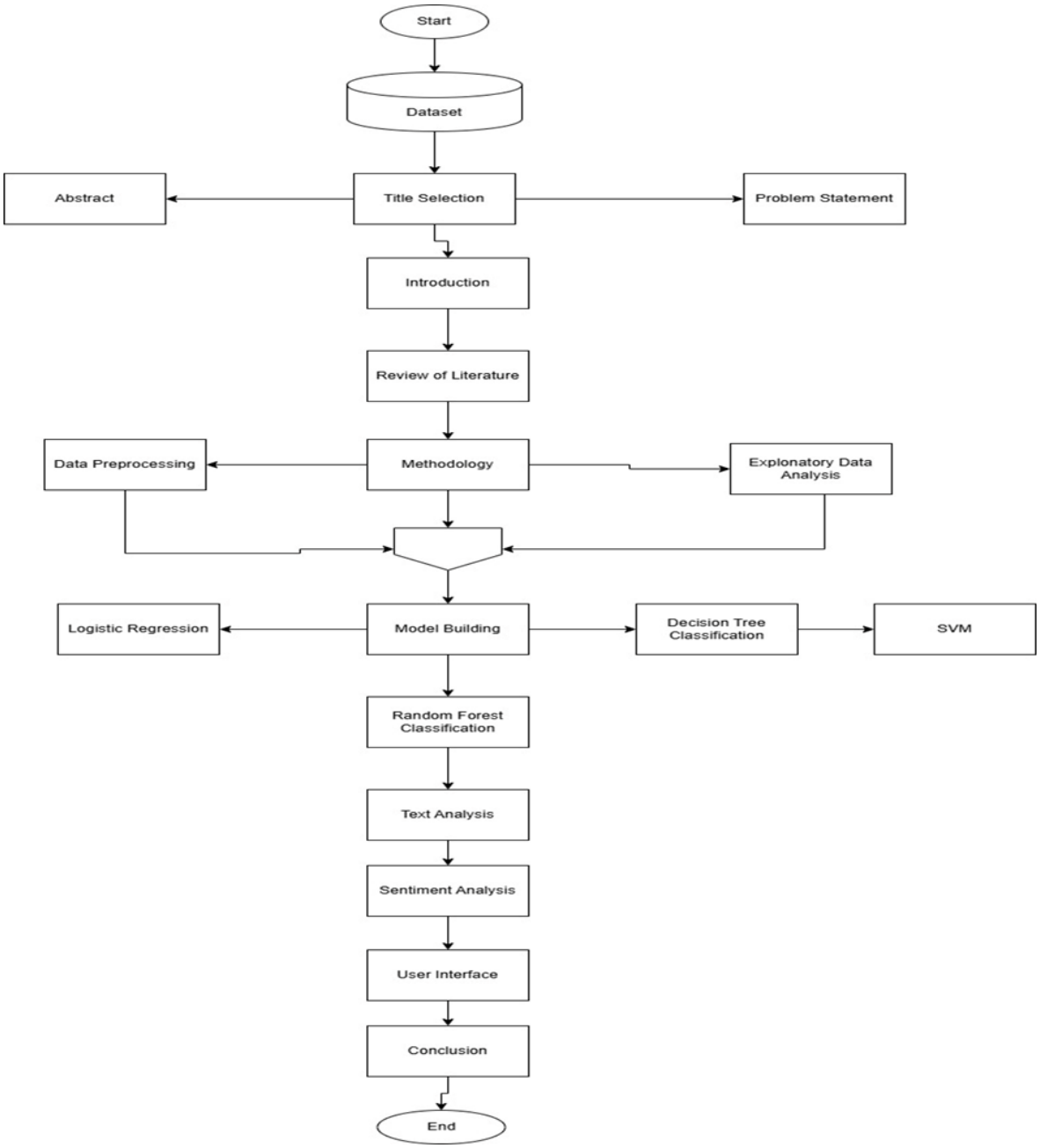


Figure 3.1: Flowchart

3.2 Dataset

The methodology was designed specifically for the dataset acquired from Kaggle which is an open-source website. The dataset contains 388 observations and 48 features. The dataset consists of numeric, text and categorical features. After the completion of preprocessing, in our study, the features listed in Table 3.2 were eventually used in modelling and analysis [1].

Features (Attributes)	**Data Type**	**Description**
Ease and convenient	Categorical	Which medium does the consumer feel convenient to order food.
Time Saving	Categorical	If the customer's order food online. Is it time saving or not?
More Restaurant Choices	Categorical	Which medium provides more restaurant choices.
More offers and discounts	Categorical	Offers and discounts
Easy payment options	Categorical	Ease of payment
Good Food Quality	Categorical	Quality of the ordered food.
Good Tracking System	Categorical	Which medium has good tracking system?
Influence of rating	Categorical	How does rating influence the decision of the customers
Output	Categorical	Yes/No, whether the customer want to order the

		food considering all the factors.
Reviews	Categorical	Reviews of the customers after eating the food.

Table 3.2: Features in the dataset, their datatypes and description

3.3 Data Preprocessing

Data preparation is the process of cleaning and transforming unprocessed data. It is the most important step before processing, and it often involves reformatting data, fixing the null values, to make the data more useful [2].

3.3.1 Label Encoding the categorical features

Label Encoding in Python is a technique used to convert categorical data (labels) into numerical format, so it can be processed by machine learning algorithms. It is especially useful when your target variable or features contain categorical values [1].

3.3.2 SMOTE Analysis

SMOTE (Synthetic Minority Oversampling Technique) is a method used in machine learning to address imbalanced datasets by generating synthetic samples for the minority class, rather than simply duplicating existing ones. This helps balance the class distribution and improves the performance of classification models.

3.4 Exploratory Data Analysis

EDA (Exploratory Data Analysis) is the process of analyzing and visualizing a dataset to understand its structure, identify patterns, detect anomalies, and summarize its main characteristics, often using statistical and graphical methods.

3.4.1 Distribution of Age and Family Size:

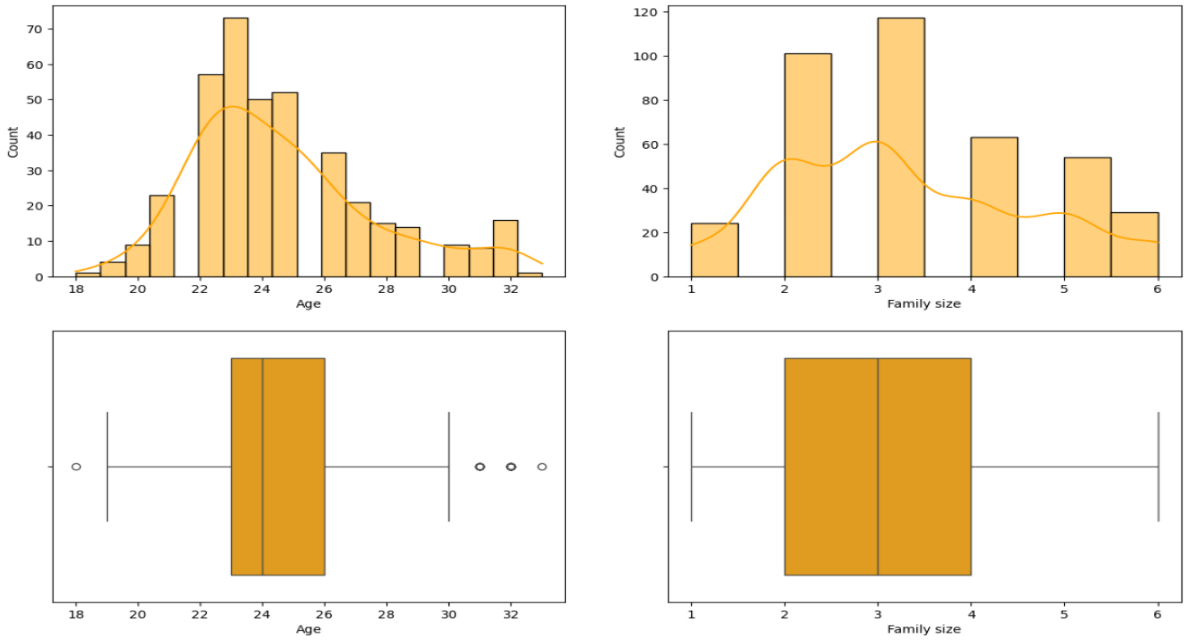


Figure 3.4.1: Distribution of numerical features

3.4.2 Correlation between Monthly Income and Educational Qualification:

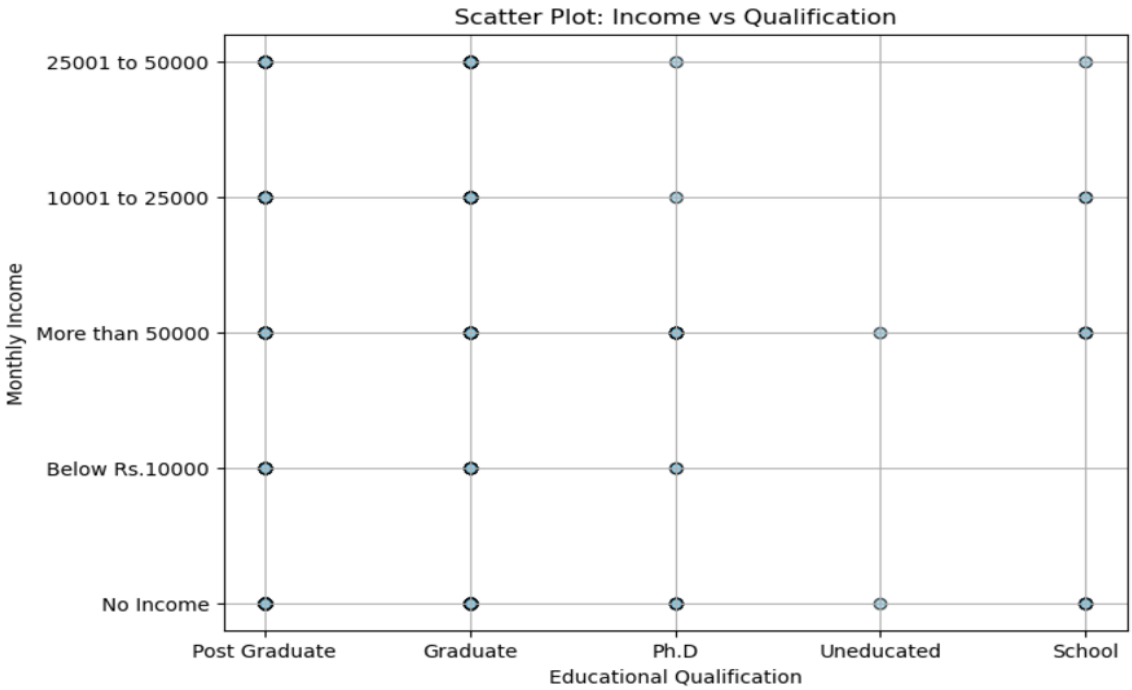


Figure 3.4.2: Correlation matrix

3.4.3 Count of Categorical Data:

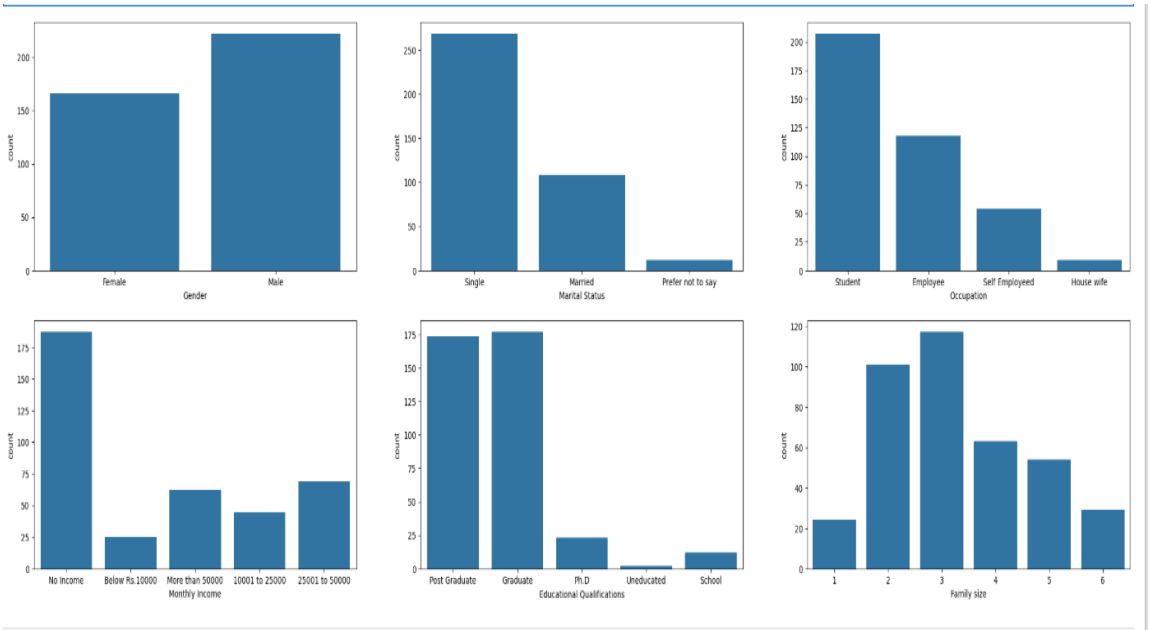


Figure 3.4.3: Count plot

3.4.4 Food preference of the customers:

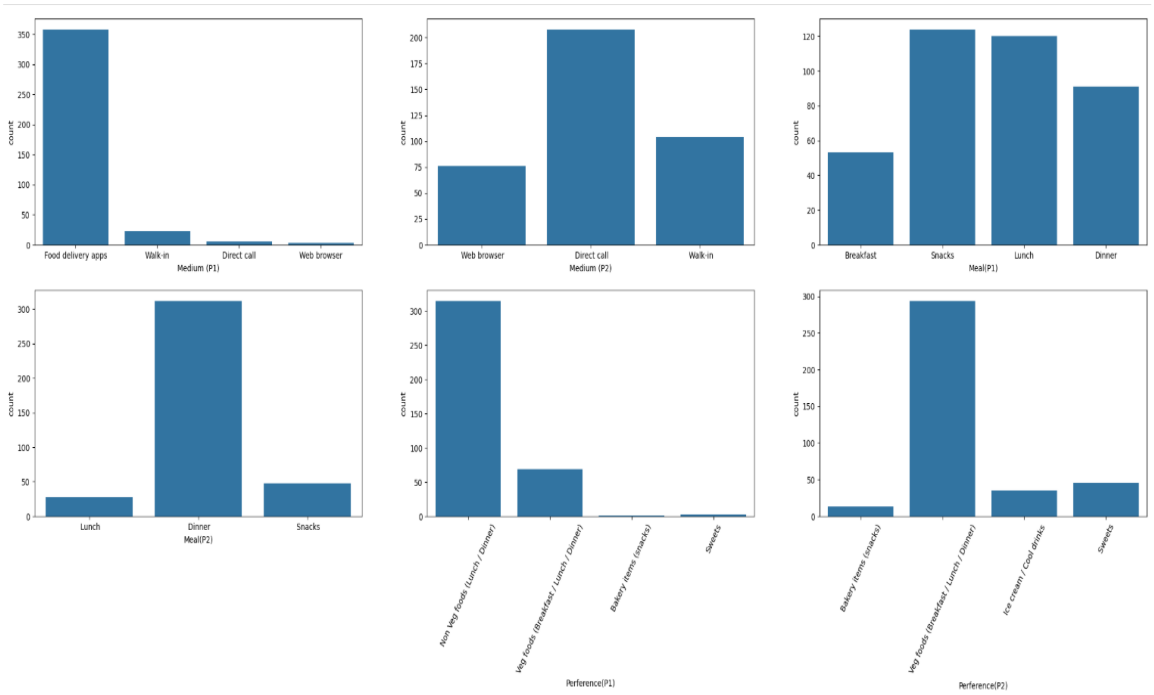


Figure 3.4.4: count of food preferences

3.4.5 Influence of positive factors on the customers:

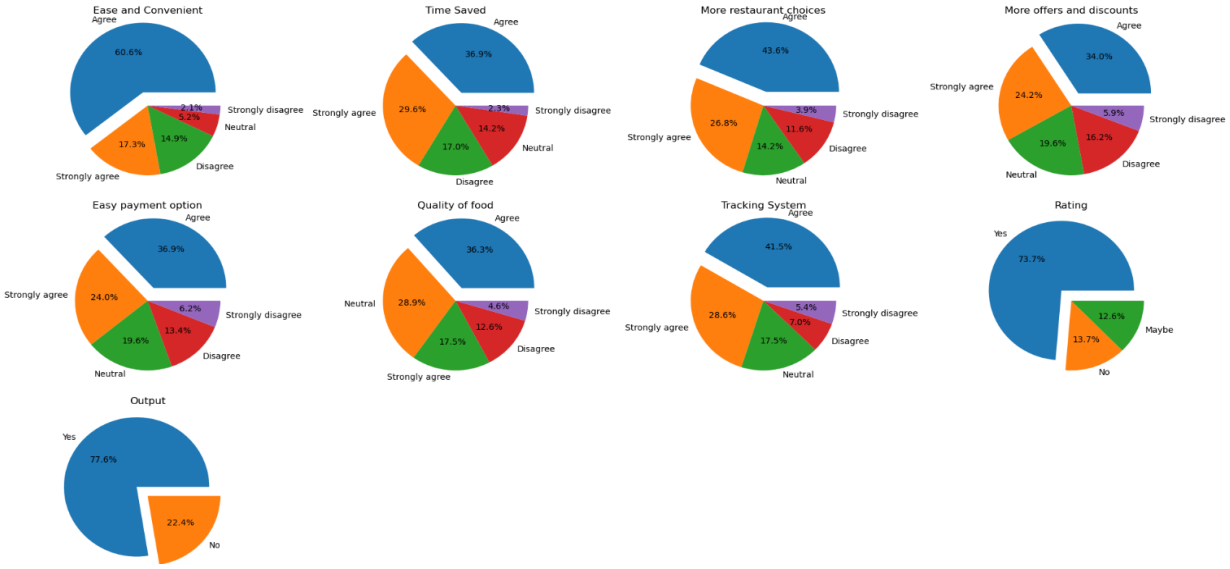


Figure 3.4.5: Influence of positive factors

3.4.6 Influence of negative factors on the customers:

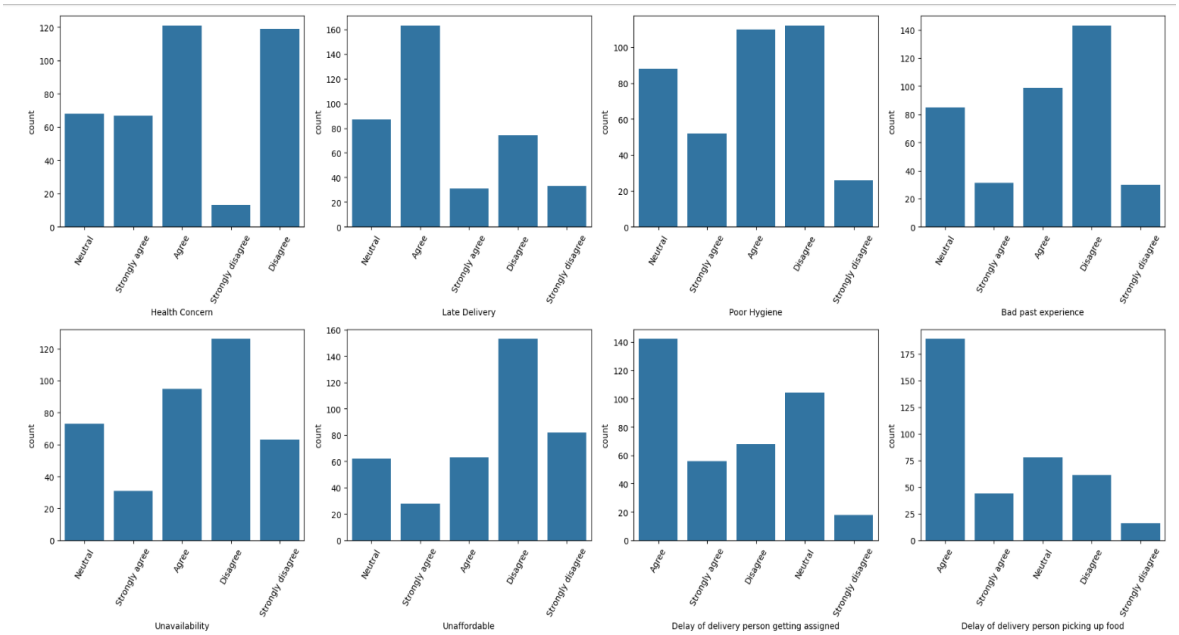


Figure 3.4.6: Count of negative factors

Chapter-4

Tools and Packages

4.1 Data Manipulation and Analysis:

- **Pandas**:
  Used for loading, cleaning, and manipulating the dataset.
  Example: pd.read_csv() to load the dataset.
- NumPy: For numerical computations and working with arrays.

4.2 Data Visualization

- **Matplotlib**: For creating static, animated, and interactive visualizations.
- **Seaborn**: Built on Matplotlib, provides high-level plotting functions for statistical data visualization.

4.3 Data Preprocessing

- **scikit-learn.preprocessing**: For label encoding, scaling, and feature transformations.
- **imblearn**: Specifically for oversampling techniques like SMOTE.

4.4 Natural Language Processing (NLP):

- **NLTK (Natural Language Toolkit)**:
  Used for preprocessing text, including lemmatization and stop word removal.
  Example:
  - WordNetLemmatizer() for lemmatization.
  - stopwords.words('english') for filtering out stop words.
- **Regular Expressions (re)**:
  Used for cleaning text by removing punctuation and unwanted characters.
  Example: re.sub(r'[^\w\s]', '', text).

4.5 Machine Learning:

- **Scikit-learn**:
  - TfidfVectorizer: Converts text into numerical features for the model.
  - MultinomialNB: The Naive Bayes model for sentiment analysis.
  - train_test_split: Splits data into training and testing sets.
  - accuracy_score: Evaluates model performance.
  - cross_val_score: Performs cross-validation for model evaluation.

4.6 Handling Class Imbalance:

- **Imbalanced-learn (imblearn)**:
    - SMOTE (Synthetic Minority Over-sampling Technique): Balances class distribution by oversampling the minority class.

4.7 UI and Interaction:

- **Gradio**:
  Used to create a simple and interactive web-based user interface for real-time sentiment analysis.
  Example: gr.Interface() to set up input and output components for user interaction.

Chapter-5

Results and Discussion

5.1 Regression

Regression in Python refers to a set of statistical and machine learning techniques used to model and analyze relationships between a dependent variable (target) and one or more independent variables (features). The goal of regression is to predict continuous outcomes, such as sales, prices, or temperatures.

5.2 Classification

Classification in Python refers to a supervised machine learning technique where the goal is to predict a categorical (discrete) output variable based on one or more input variables. It involves assigning data points to predefined categories or classes.

5.3 Models

After having performed data preprocessing, we apply both regression and classification models, namely

5.3.1 Logistic Regression

Logistic Regression is a statistical and machine learning algorithm used for binary or multi-class classification problems. It predicts the probability that a given input belongs to a particular class and is especially useful when the target variable is categorical.

5.3.2 Random Forest Classification

Random Forest Classification is an ensemble machine learning technique used for classification tasks. It builds multiple decision trees during training and combines their outputs (by majority voting) to make a final prediction. Random forests are robust, versatile, and effective for handling complex datasets with non-linear relationships.

5.3.3 Decision Tree Classification

Decision Tree Classification is a supervised machine learning algorithm used for classification tasks. It works by splitting the dataset into subsets based on feature values, creating a tree-like structure where each node represents a decision, and each leaf node represents a class label.

5.3.4 Support Vector Machine (SVM)

Support Vector Machine (SVM) is a supervised machine learning algorithm used for classification and regression tasks. It is particularly effective for high-dimensional datasets and is robust to outliers.

5.3.5 Gradient Boost Model

Gradient Boosting Model is a powerful ensemble machine learning technique used for classification and regression tasks. It builds a series of weak learners, typically decision trees, and combines them into a single strong predictive model by focusing on minimizing errors.

5.3.6 Naïve Bayes

Naive Bayes is a supervised learning algorithm based on Bayes' Theorem, assuming independence between predictors (features). It calculates the probability of each class for a given input and assigns the class with the highest probability.

5.3.7 Cross Validation

Cross-validation is a technique used to evaluate a model's performance by splitting the dataset into multiple subsets (folds). The model is trained on some folds and tested on the remaining fold(s), ensuring the evaluation is robust and unbiased.

5.4 Text Analysis

Text Analysis, also known as Text Mining or Natural Language Processing (NLP), is the process of extracting meaningful information, insights, or patterns from textual data using computational techniques.

5.5 N-Gram Analysis

N-gram analysis is a text analysis technique that involves breaking down text into contiguous sequences of nnn items (e.g., words, characters, or tokens). These sequences are called N-grams, where nnn represents the number of items in each sequence.

5.5.1 Unigram

Consists of single items (words or characters).

5.5.2 Bigram

Consists of sequences of two consecutive items.

5.5.3 Trigram

Consists of sequences of three consecutive items.

5.6 Sentiment Analysis

Sentiment Analysis is a Natural Language Processing (NLP) task that determines the sentiment or emotional tone of a piece of text. It is commonly used to identify whether the sentiment is positive, negative, or neutral, although more granular sentiments can also be detected.

5.7 Evaluation metrics

5.7.1 Accuracy

Accuracy is measure of the proportion of correctly classified instances out of the total instances.

5.7.2 Precision

Precision indicates the proportion of correctly predicted positive results out of all predicted positives. Precision is used in binary classification tasks.

5.7.3 Recall

Recall measures the proportion of actual positive cases correctly identified by the model. Recall is often used in binary classification tasks.

5.7.4 F1-Score

The F1-score is a weighted average of precision and recall that combines the two metrics into a single score.

In this part, the experimentation validation of the predictive analytics methodology has been tested using a dataset that was acquired from Kaggle. The dataset includes 388 observations and 48 features. In the features there are both positive and negative factors that influence the customers from ordering food online. For the model deployment only, positive factors were included to know whether these factors had an impact on customers choices of placing order from online delivery platforms. The predictors used for the model building were “Ease and convenient, Time saving, More restaurant choices, More Offers and Discount, Easy Payment option, Good Food quality, Good Tracking system, Influence of rating” and “Output” was the target column. In this study, total of 7 models were compared and accuracy are stated in table 5.7. Among all the models Random Forest has the best accuracy followed by Gradient Boost, Naïve Bayes and cross-validation model.

Model	Accuracy
Logistic Regression	61.5%
Random Forest Classification	90%
Decision Tree Classification	92.3%
Support Vector Machine (SVM)	57.2%
Gradient Boost Model	87.2%
Naïve Bayes	89%

Cross Validation Model	89%

Table 5.7: Accuracy Summary

After the completion of model deployment, “Text Analysis” is performed using both “Reviews” and the “Output” column and rows with customer reviews are displayed. Further N-gram analysis is performed on the reviews column. It begins by downloading necessary NLTK resources for tokenization and stopword removal. The preprocessing step cleans the text by removing special characters, converting it to lowercase, tokenizing it into words, and filtering out common stopwords. Each review is then transformed into a tokenized list stored in a new column. Using a function, the code generates unigrams, bigrams, and trigrams (sequences of one, two, and three words, respectively) from these tokenized reviews and calculates their frequencies using Counter. Finally, it visualizes the top 10 most frequent unigrams, bigrams, and trigrams in bar plots, allowing for an intuitive understanding of the most common words and word combinations in the text data.

Sentiment analysis on the Reviews column of a Data Frame using the TextBlob library. First, it handles missing values by replacing them with empty strings to ensure smooth processing. The TextBlob library is used to calculate the sentiment polarity (how positive or negative the review is) and subjectivity (how opinion-based the review is) for each review. These values are stored in new columns, Polarity and Subjectivity, respectively. A custom function then classifies the sentiment as "Positive," "Negative," or "Neutral" based on the polarity value and stores this classification in a Sentiment column. The code counts the occurrences of each sentiment type and visualizes the sentiment distribution using a bar chart. Finally, it saves the updated Data Frame, including sentiment information, to a CSV file named reviews_with_sentiment.csv. This workflow provides insights into the sentiment trends within the review data.

Further, developing sentiment analysis application using a Naive Bayes classifier, TF-IDF vectorization, SMOTE for handling class imbalance, and Gradio for creating a user interface. The reviews are pre-processed through lemmatization, removal of punctuation, and stopword filtering. The data is then split into training and testing sets, and TF-IDF vectorization converts the textual data into numerical format for modelling. To address class imbalance, SMOTE is applied to oversample the minority class in the training data. A Multinomial Naive Bayes model is trained on the processed data with optimized smoothing (alpha=0.5). Finally, a Gradio interface is created to allow users to input text for real-time sentiment predictions. The app preprocesses user input, vectorizes it, and uses the trained model to predict and display the sentiment. The Gradio app is launched with an option to share publicly.

5.8 Performance Analysis

Model	Accuracy	Precision	Recall	F1-score
Logistic Regression	75%	75%	100%	86%
Random Forest Classification	91%	93%	96%	94%
Decision Tree Classification	86%	89%	93%	91%
Support Vector Machine (SVM)	77%	76%	100%	87%
Gradient Boost Model	89%	92%	94%	93%

Model(Base Paper)	Accuracy	Precision	Recall	F1-score
Logistic Regression	93%	93%	93%	93%
Random Forest Classification	94%	94%	94%	95%
Decision Tree Classification	95%	95%	95%	95%
Support Vector Machine(SVM)	74%	84%	74%	78%
Gradient Boost Model	94%	94%	94%	94%

Table 5.8: Performance Analysis

Chapter-6
Conclusion

The study of customers preferences in online food delivery systems reveals several key insights into what drives user satisfaction and engagement. Understanding these factors is essential for platforms to enhance user experience, retain customers, and remain competitive in the rapidly growing online food delivery market.

Key Findings:

- **Convenience and Time-Saving:** Features like ease of ordering, quick navigation, and efficient delivery times remain top priorities for customers. These aspects significantly influence user satisfaction and repeat usage.
- **Variety and Discounts:** Access to a diverse range of restaurants, coupled with promotional offers and discounts, attracts and retains customers.
- **Customer Concerns:** Negative experiences, including poor hygiene, unavailability of menu items, and affordability, were highlighted as pain points. Addressing these issues is critical to building trust and loyalty.
- This study successfully built a sentiment analysis model using Naive Bayes, focusing on user reviews and binary classification ("Yes" and "No").
- The model demonstrates practical applications by enabling real-time sentiment predictions, making it useful for businesses to analyze customer feedback.
- Overall, this project provides a strong foundation for sentiment analysis, contributing to better decision-making and improved understanding of user sentiments.

Chapter-7

Future Enhancement

Use More Advanced Models: Explore deep learning models like Long Short-Term Memory (LSTM) networks, Recurrent Neural Networks (RNNs), or Convolutional Neural Networks (CNNs) for text classification tasks.

Expand Sentiment Categories: Move from simple "Yes" and "No" classifications to multiple labels like "Positive," "Negative," and "Neutral.“

Handle Other Languages: Add support for analyzing reviews written in other languages by translating them to English. Use tools like Google Translate APIs or pretrained multilingual models.

Collect More Data: Add more reviews to the dataset to improve the model’s understanding of diverse comments.

Make the Model Dynamic: Allow the model to learn from new reviews by updating it regularly with fresh data.

Track Sentiment Trends: Analyze how sentiments change over time to identify patterns or trends in user feedback.

Explain Predictions: Show users which words or phrases influenced the sentiment prediction most, making the model more understandable.

Chapter-8
Appendices

8.1 Text Analysis Code

```
# copying the dataset
df1 = df.copy()
#removing irrelevant reviews- Nil value
df1=
df[~df['Reviews'].isin(['NIL','nil','Nil','No','Nil\n'])]
# considering the two columns
df1=df1[['Reviews','Output']]
df1
```

8.2 N-gram Analysis Code

```
# Importing required libraries
import pandas as pd
import nltk
from nltk.corpus import stopwords
from nltk.tokenize import word_tokenize
from collections import Counter
import re
import matplotlib.pyplot as plt
# Download required resources
import nltk
nltk.download('punkt')
nltk.download('stopwords')

# Preprocessing function
def preprocess(text):
```

```
    text = re.sub(r'[^a-zA-Z\s]', '', text)  # Remove special
characters
    text = text.lower()  # Convert to lowercase
    tokens = word_tokenize(text)  # Tokenize
    stop_words = set(stopwords.words('english'))  # Load
stopwords
    tokens = [word for word in tokens if word not in
stop_words]  # Remove stopwords
    return tokens


# Apply preprocessing to the Reviews column
df['Tokenized_Reviews'] =
df['Reviews'].fillna('').apply(preprocess)


# Function to generate N-grams and their frequencies
def generate_ngrams(tokens_list, n):
    all_ngrams = [ngrams(tokens, n) for tokens in
tokens_list]  # Generate N-grams for each tokenized review
    flat_ngrams = [item for sublist in all_ngrams for item in
sublist]  # Flatten the list of lists
    return Counter(flat_ngrams)  # Count frequencies


# Generate unigrams, bigrams, and trigrams
unigrams = generate_ngrams(df['Tokenized_Reviews'], 1)
bigrams = generate_ngrams(df['Tokenized_Reviews'], 2)
trigrams = generate_ngrams(df['Tokenized_Reviews'], 3)


# Function to plot N-gram frequencies
def plot_ngrams(ngram_counter, title, top_n=10):
    most_common = ngram_counter.most_common(top_n)  # Get the
top N most common N-grams
    ngrams_list, counts = zip(*most_common)
    ngrams_list = [' '.join(ngram) for ngram in ngrams_list]
# Join tuples into strings
```

```
    plt.figure(figsize=(10, 6))
    plt.bar(ngrams_list, counts, color='skyblue')
    plt.title(title)
    plt.ylabel("Frequency")
    plt.xticks(rotation=45, ha='right')
    plt.tight_layout()
    plt.show()

# Plot top 10 unigrams, bigrams, and trigrams
plot_ngrams(unigrams, "Top 10 Unigrams", top_n=10)
plot_ngrams(bigrams, "Top 10 Bigrams", top_n=10)
plot_ngrams(trigrams, "Top 10 Trigrams", top_n=10)
```

8.3 Sentiment Count Code (Positive, Negative, Neutral)

```
!pip install textblob
from textblob import TextBlob

# Handle missing values in the Reviews column
df['Reviews'] = df['Reviews'].fillna('')

# Function to calculate sentiment polarity
def get_sentiment_polarity(text):
    blob = TextBlob(text)
    return blob.sentiment.polarity

# Function to calculate sentiment subjectivity
def get_sentiment_subjectivity(text):
    blob = TextBlob(text)
    return blob.sentiment.subjectivity
```

```
# Apply sentiment analysis
df['Polarity'] = df['Reviews'].apply(get_sentiment_polarity)
df['Subjectivity'] =
df['Reviews'].apply(get_sentiment_subjectivity)

# Classify reviews based on polarity
def classify_sentiment(polarity):
    if polarity > 0:
        return 'Positive'
    elif polarity < 0:
        return 'Negative'
    else:
        return 'Neutral'

df['Sentiment'] = df['Polarity'].apply(classify_sentiment)

# Display sentiment counts
sentiment_counts = df['Sentiment'].value_counts()
print("Sentiment Counts:\n", sentiment_counts)

# Plot sentiment distribution
plt.figure(figsize=(8, 5))
sentiment_counts.plot(kind='bar', color=['orange', 'green',
'red'])
plt.title("Sentiment Distribution")
plt.xlabel("Sentiment")
plt.ylabel("Number of Reviews")
plt.xticks(rotation=0)
plt.tight_layout()
plt.show()
```

```
# Save results to a new CSV file
df.to_csv('reviews_with_sentiment.csv', index=False)
print("Sentiment analysis results saved to
'reviews_with_sentiment.csv'")
```

8.4 Sentiment Analysis for user inputs

```
import pandas as pd
from sklearn.model_selection import train_test_split,
cross_val_score
from sklearn.feature_extraction.text import TfidfVectorizer
from sklearn.naive_bayes import MultinomialNB
from imblearn.over_sampling import SMOTE
from nltk.corpus import stopwords
from nltk.stem import WordNetLemmatizer
import re
from sklearn.metrics import accuracy_score

# Load the data
file_path = 'D:\onlinedeliverydata.csv'  # Replace with your
file path



# Extract relevant columns
data_cleaned = df[['Reviews', 'Output']].dropna()

# Preprocessing function
lemmatizer = WordNetLemmatizer()
stop_words = set(stopwords.words('english'))

def preprocess_text(text):
    text = text.lower()  # Convert to lowercase
```

```
    text = re.sub(r'[^\w\s]', '', text)  # Remove punctuation
    text = ' '.join([lemmatizer.lemmatize(word) for word in
text.split() if word not in stop_words])  # Lemmatize
    return text


# Apply preprocessing to the reviews
data_cleaned['Reviews'] =
data_cleaned['Reviews'].apply(preprocess_text)


# Split the data into features and target variable
x = data_cleaned['Reviews']
y = data_cleaned['Output']


# Split the data into training and testing sets
x_train, x_test, y_train, y_test = train_test_split(x, y,
test_size=0.4, random_state=100)


# Vectorize the text using TF-IDF
vectorizer = TfidfVectorizer(stop_words='english',
lowercase=True)
x_train_vec = vectorizer.fit_transform(x_train)
x_test_vec = vectorizer.transform(x_test)


# Handle class imbalance using SMOTE (Synthetic Minority
Over-sampling Technique)
smote = SMOTE(random_state=42)
x_train_vec, y_train = smote.fit_resample(x_train_vec,
y_train)


# Train the Naive Bayes model with hyperparameter tuning
(alpha=0.5)
model = MultinomialNB(alpha=0.5)
model.fit(x_train_vec, y_train)
```

```
# Evaluate the model with accuracy score
y_pred = model.predict(x_test_vec)
print(f"Accuracy on Test Set: {accuracy_score(y_test,
y_pred):.2f}")


# Cross-validation to evaluate the model's performance
cv_scores = cross_val_score(model, x_train_vec, y_train,
cv=5)
print(f"Cross-Validation Accuracy: {cv_scores.mean():.2f}")


# Interactive prediction
while True:
    user_input = input("\nEnter a comment to analyze
sentiment (or type 'exit' to quit): ")
    if user_input.lower() == 'exit':
        break
    user_input_vec = vectorizer.transform([user_input])
    prediction = model.predict(user_input_vec)
    print(f"Predicted Sentiment: {prediction[0]}")
```

8.5 Sentiment Analysis using Gradio

```
!pip install --upgrade gradio
import pandas as pd
from sklearn.model_selection import train_test_split
from sklearn.feature_extraction.text import TfidfVectorizer
from sklearn.naive_bayes import MultinomialNB
from imblearn.over_sampling import SMOTE
from nltk.corpus import stopwords
from nltk.stem import WordNetLemmatizer
import re
```

```
from sklearn.metrics import accuracy_score

import gradio as gr


# Load the data

file_path = 'D:/onlinedeliverydata.csv'  # Replace with your
file path

df = pd.read_csv(file_path)


# Extract relevant columns

data_cleaned = df[['Reviews', 'Output']].dropna()


# Preprocessing function

lemmatizer = WordNetLemmatizer()

stop_words = set(stopwords.words('english'))


def preprocess_text(text):

    text = text.lower()  # Convert to lowercase

    text = re.sub(r'[^\w\s]', '', text)  # Remove punctuation

    text = ' '.join([lemmatizer.lemmatize(word) for word in
text.split() if word not in stop_words])  # Lemmatize

    return text


# Apply preprocessing to the reviews

data_cleaned['Reviews'] =
data_cleaned['Reviews'].apply(preprocess_text)


# Split the data into features and target variable

x = data_cleaned['Reviews']

y = data_cleaned['Output']


# Split the data into training and testing sets
```

```
x_train, x_test, y_train, y_test = train_test_split(x, y,
test_size=0.4, random_state=100)


# Vectorize the text using TF-IDF

vectorizer = TfidfVectorizer(stop_words='english',
lowercase=True)

x_train_vec = vectorizer.fit_transform(x_train)

x_test_vec = vectorizer.transform(x_test)


# Handle class imbalance using SMOTE (Synthetic Minority
Over-sampling Technique)

smote = SMOTE(random_state=42)

x_train_vec, y_train = smote.fit_resample(x_train_vec,
y_train)


# Train the Naive Bayes model with hyperparameter tuning
(alpha=0.5)

model = MultinomialNB(alpha=0.5)

model.fit(x_train_vec, y_train)


# Function for Gradio

def predict_sentiment(user_input):

    user_input = preprocess_text(user_input)

    user_input_vec = vectorizer.transform([user_input])

    prediction = model.predict(user_input_vec)

    return f"Predicted Sentiment: {prediction[0]}"


# Create Gradio interface

interface = gr.Interface(

    fn=predict_sentiment,

    inputs=gr.Textbox(lines=2, placeholder="Enter a comment
here..."),

    outputs=gr.Textbox(label="Predicted Sentiment"),
```

```
    title="Sentiment Analysis",
    description="Enter a comment to determine whether the sentiment is positive or negative.",
)

# Launch the app
interface.launch(share=True)
```

8.6 Screenshots

8.6.1 Loading the dataset

```
[5]: # Loading the dataset
     df = pd.read_csv('onlinedeliverydata.csv')
     df
```

[5]:

	Age	Gender	Marital Status	Occupation	Monthly Income	Educational Qualifications	Family size	Pin code	Medium (P1)	Medium (P2)	...	Good Road Condition	Influence of rating	Less Delivery time	High Quality of package	Freshness	Te
0	20	Female	Single	Student	No Income	Post Graduate	4	560001	Food delivery apps	Web browser	...	Neutral	Yes	Moderately Important	Moderately Important	Moderately Important	
1	24	Female	Single	Student	Below Rs.10000	Graduate	3	560009	Food delivery apps	Web browser	...	Disagree	Yes	Very Important	Very Important	Very Important	
2	22	Male	Single	Student	Below Rs.10000	Post Graduate	3	560017	Food delivery apps	Direct call	...	Neutral	Yes	Important	Very Important	Very Important	
3	22	Female	Single	Student	No Income	Graduate	6	560019	Food delivery apps	Walk-in	...	Agree	Yes	Very Important	Important	Very Important	
4	22	Male	Single	Student	Below Rs.10000	Post Graduate	4	560010	Walk-in	Direct call	...	Agree	Yes	Important	Important	Important	
...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	
383	23	Female	Single	Student	No Income	Post Graduate	2	560001	Food delivery apps	Direct call	...	Neutral	Maybe	Important	Important	Important	
384	23	Female	Single	Student	No Income	Post Graduate	4	560048	Food delivery apps	Walk-in	...	Agree	Yes	Moderately Important	Very Important	Moderately Important	
385	22	Female	Single	Student	No Income	Post Graduate	5	560010	Food delivery apps	Direct call	...	Strongly Agree	Yes	Important	Very Important	Very Important	
386	23	Male	Single	Student	Below Rs.10000	Post Graduate	2	560009	Food delivery apps	Walk-in	...	Agree	Yes	Important	Very Important	Very Important	
387	23	Male	Single	Student	No Income	Post Graduate	5	560078	Walk-in	Direct call	...	Neutral	Maybe	Slightly Important	Unimportant	Moderately Important	

388 rows × 48 columns

8.6.2 Preprocessing

```
# dataset info

df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 388 entries, 0 to 387
Data columns (total 48 columns):
 #   Column                                  Non-Null Count  Dtype
---  ------                                  --------------  -----
 0   Age                                     388 non-null    int64
 1   Gender                                  388 non-null    object
 2   Marital Status                          388 non-null    object
 3   Occupation                              388 non-null    object
 4   Monthly Income                          388 non-null    object
 5   Educational Qualifications              388 non-null    object
 6   Family size                             388 non-null    int64
 7   Pin code                                388 non-null    int64
 8   Medium (P1)                             388 non-null    object
 9   Medium (P2)                             388 non-null    object
 10  Meal(P1)                                388 non-null    object
 11  Meal(P2)                                388 non-null    object
 12  Perference(P1)                          388 non-null    object
 13  Perference(P2)                          388 non-null    object
 14  Ease and convenient                     388 non-null    object
 15  Time saving                             388 non-null    object
 16  More restaurant choices                 388 non-null    object
 17  Easy Payment option                     388 non-null    object
 18  More Offers and Discount                388 non-null    object
 19  Good Food quality                       388 non-null    object
 20  Good Tracking system                    388 non-null    object
 21  Health Concern                          388 non-null    object
 22  Late Delivery                           388 non-null    object
 23  Poor Hygiene                            388 non-null    object
 24  Bad past experience                     388 non-null    object
 25  Unavailability                          388 non-null    object
 26  Unaffordable                            388 non-null    object
 27  Long delivery time                      388 non-null    object
 28  Delay of delivery person getting assigned  388 non-null  object
 29  Delay of delivery person picking up food   388 non-null  object
 30  Wrong order delivered                   388 non-null    object
 31  Missing item                            388 non-null    object
 32  Order placed by mistake                 388 non-null    object
 33  Influence of time                       388 non-null    object
 34  Order Time                              388 non-null    object
 35  Maximum wait time                       388 non-null    object
 36  Residence in busy location              388 non-null    object
 37  Google Maps Accuracy                    388 non-null    object
 38  Good Road Condition                     388 non-null    object
 39  Influence of rating                     388 non-null    object
 40  Less Delivery time                      388 non-null    object
 41  High Quality of package                 388 non-null    object
 42  Freshness                               388 non-null    object
 43  Temperature                             388 non-null    object
 44  Good Taste                              388 non-null    object
 45  Good Quantity                           388 non-null    object
 46  Output                                  388 non-null    object
 47  Reviews                                 387 non-null    object
dtypes: int64(3), object(45)
memory usage: 145.6+ KB
```

[12]:
```
# displaying rows and columns
df.shape
```

[12]:
```
(388, 48)
```

[84]:
```
# finding null values

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

[84]:
```
Age                                       0
Gender                                    0
Marital Status                            0
Occupation                                0
Monthly Income                            0
Educational Qualifications                0
Family size                               0
Pin code                                  0
Medium (P1)                               0
Medium (P2)                               0
Meal(P1)                                  0
Meal(P2)                                  0
Perference(P1)                            0
Perference(P2)                            0
Ease and convenient                       0
Time saving                               0
More restaurant choices                   0
Easy Payment option                       0
More Offers and Discount                  0
Good Food quality                         0
Good Tracking system                      0
Health Concern                            0
Late Delivery                             0
Poor Hygiene                              0
Bad past experience                       0
Unavailability                            0
Unaffordable                              0
Long delivery time                        0
Delay of delivery person getting assigned    0
Delay of delivery person picking up food     0
Wrong order delivered                     0
Missing item                              0
Order placed by mistake                   0
Influence of time                         0
Order Time                                0
Maximum wait time                         0
Residence in busy location                0
Google Maps Accuracy                      0
Good Road Condition                       0
Influence of rating                       0
Less Delivery time                        0
High Quality of package                   0
Freshness                                 0
Temperature                               0
Good Taste                                0
Good Quantity                             0
Output                                    0
Reviews                                   1
dtype: int64
```

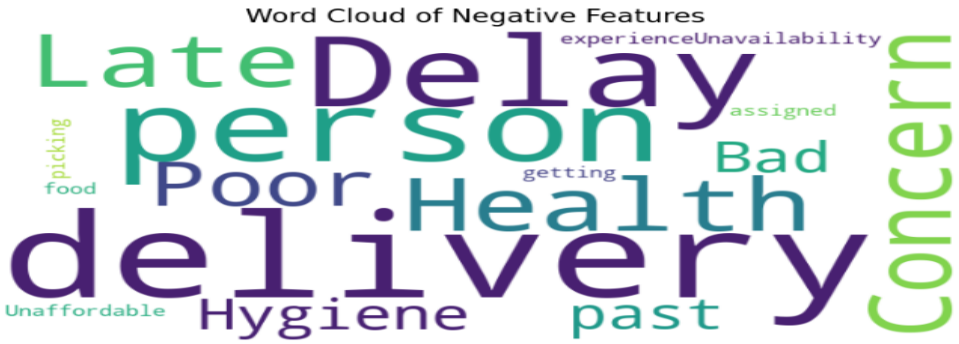
8.6.3 WordCloud of positive features

```
# Wordcloud of positive features
```

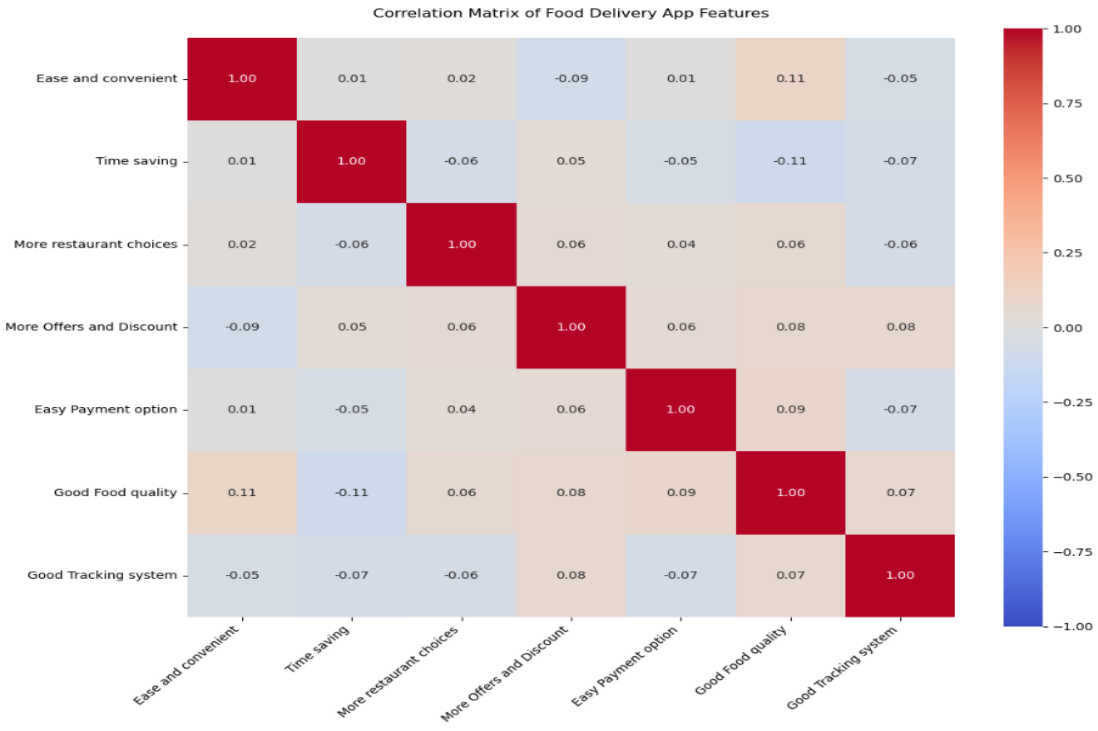
```
from wordcloud import WordCloud

# List of features
pos_features = [
    'Ease and convenient', 'Time saving', 'More restaurant choices',
    'More Offers and Discount', 'Easy Payment option', 'Good Food quality',
    'Good Tracking system', 'Influence of rating'
]

# Convert the list into a single string (word cloud needs text input)
features_text = " ".join(pos_features)

# Generate the word cloud
wordcloud = WordCloud(
    width=800,
    height=400,
    background_color='white',
    colormap='viridis',
    random_state=42
).generate(features_text)

# Plot the word cloud
plt.figure(figsize=(10, 6))
plt.imshow(wordcloud, interpolation='bilinear')
plt.axis('off')  # Turn off axis
plt.title("Word Cloud of Positve Features", fontsize=16)
plt.show()
```

8.6.4 WordCloud of negative features

```
# Import necessary libraries
from wordcloud import WordCloud
import matplotlib.pyplot as plt

# List of features
neg_features = [
    'Health Concern','Late Delivery','Poor Hygiene','Bad past experience'
    'Unavailability','Unaffordable','Delay of delivery person getting assigned',
    'Delay of delivery person picking up food'
]

# Convert the list into a single string (word cloud needs text input)
features_text = " ".join(neg_features)

# Generate the word cloud
wordcloud = WordCloud(
    width=800,
    height=400,
    background_color='white',
    colormap='viridis',
    random_state=42
).generate(features_text)

# Plot the word cloud
plt.figure(figsize=(10, 6))
plt.imshow(wordcloud, interpolation='bilinear')
plt.axis('off')  # Turn off axis
plt.title("Word Cloud of Negative Features", fontsize=16)
plt.show()
```



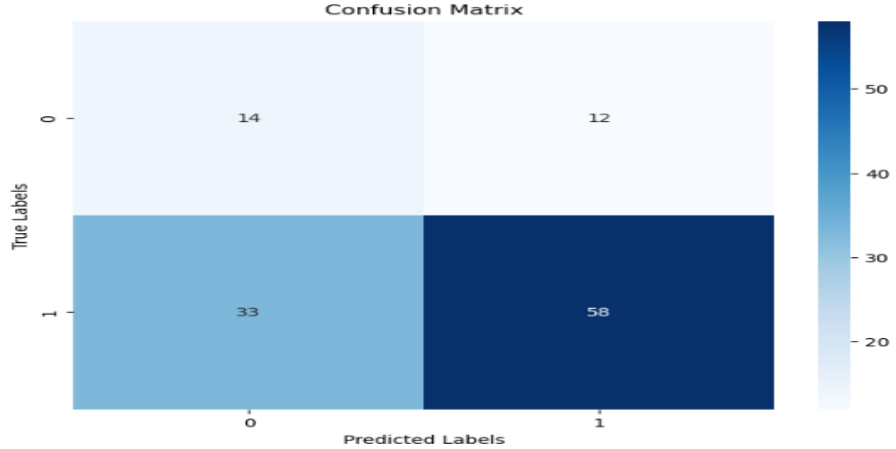
8.6.5 Heat map for correlation matrix



8.7 Model Deployment

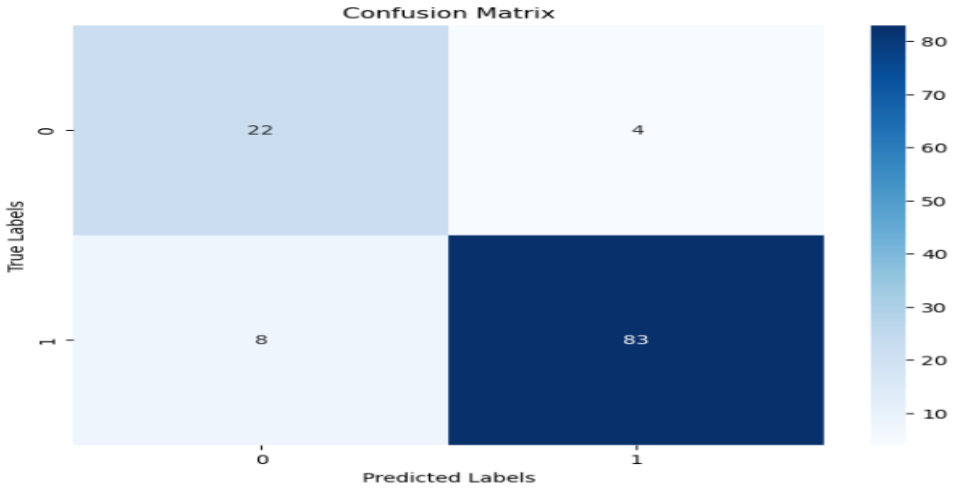
8.7.1 Logistic Regression

```
Confusion Matrix:
[[14 12]
 [33 58]]

Classification Report:
              precision    recall  f1-score   support

           0       0.30      0.54      0.38        26
           1       0.83      0.64      0.72        91

    accuracy                           0.62       117
   macro avg       0.56      0.59      0.55       117
weighted avg       0.71      0.62      0.65       117

Accuracy Score:
0.6153846153846154
```



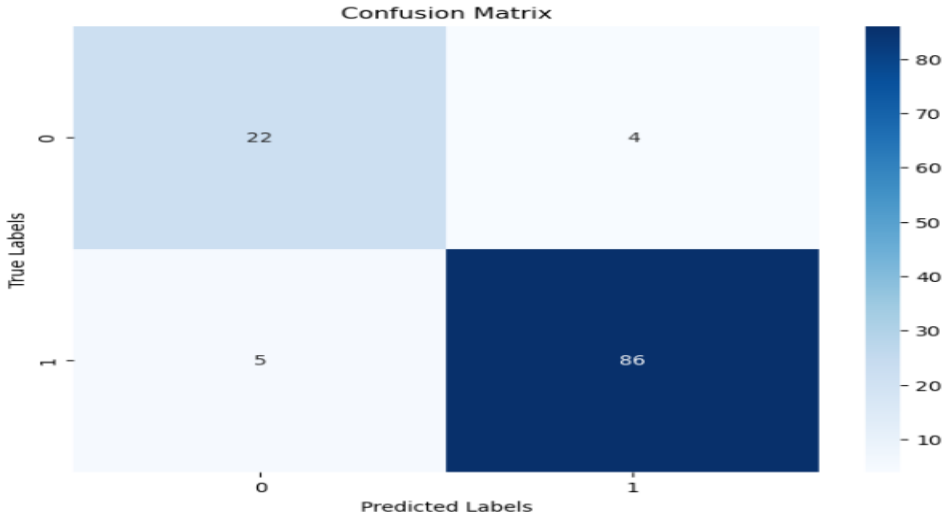
8.7.2 Random Forest Classification

```
Confusion Matrix:
[[22  4]
 [ 8 83]]

Classification Report:
              precision    recall  f1-score   support

           0       0.73      0.85      0.79        26
           1       0.95      0.91      0.93        91

    accuracy                           0.90       117
   macro avg       0.84      0.88      0.86       117
weighted avg       0.90      0.90      0.90       117

Accuracy Score:
0.8974358974358975
```



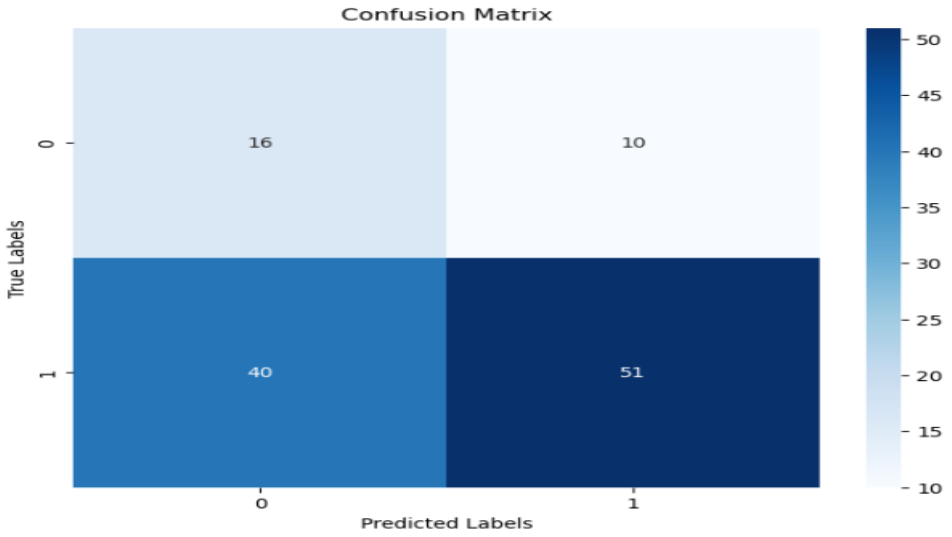
8.7.3 Decision Tree Classifier

```
Confusion Matrix:
[[22  4]
 [ 5 86]]

Classification Report:
              precision    recall  f1-score   support

           0       0.81      0.85      0.83        26
           1       0.96      0.95      0.95        91

    accuracy                           0.92       117
   macro avg       0.89      0.90      0.89       117
weighted avg       0.92      0.92      0.92       117

Accuracy Score:
0.9230769230769231
```



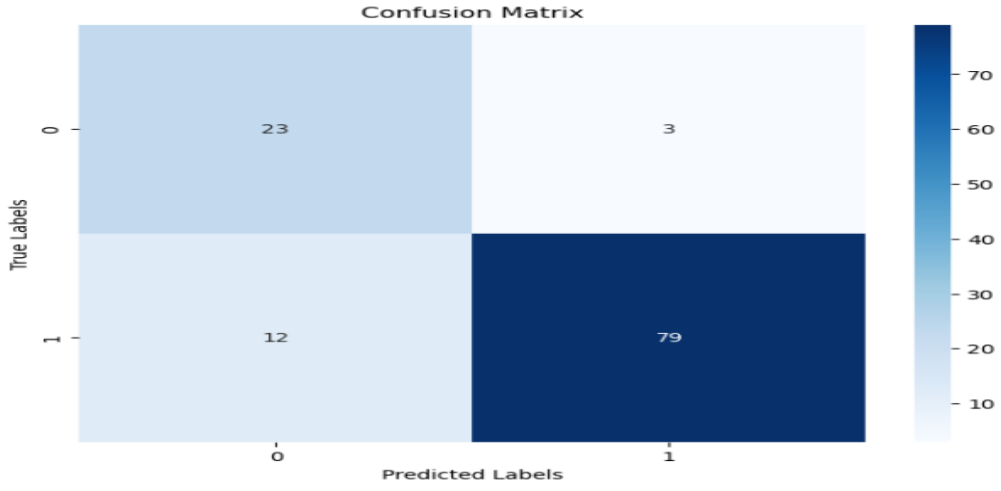
8.7.4 Support Vector Machine (SVM)

```
Confusion Matrix:
[[16 10]
 [40 51]]

Classification Report:
              precision    recall  f1-score   support

           0       0.29      0.62      0.39        26
           1       0.84      0.56      0.67        91

    accuracy                           0.57       117
   macro avg       0.56      0.59      0.53       117
weighted avg       0.71      0.57      0.61       117


Accuracy Score:
0.5726495726495726
```

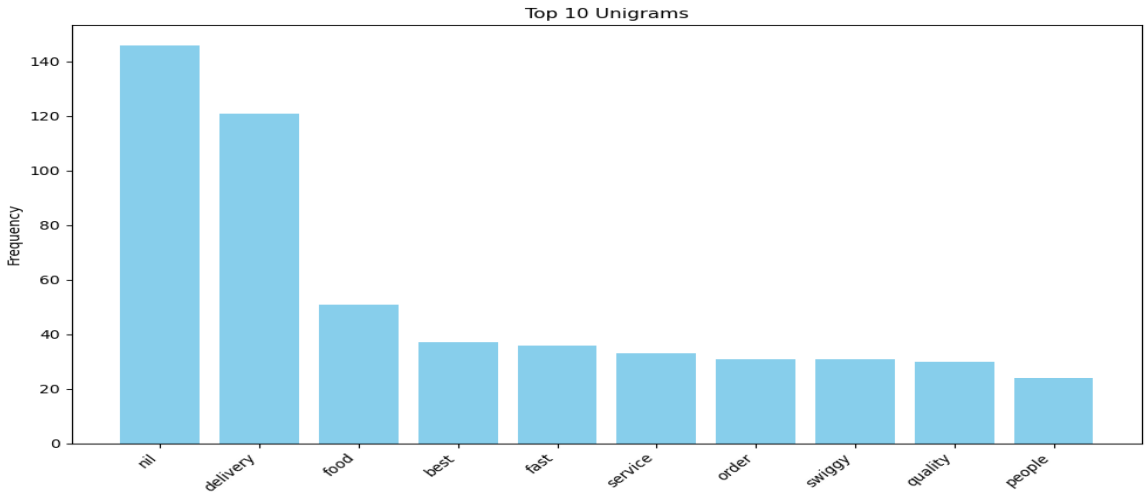


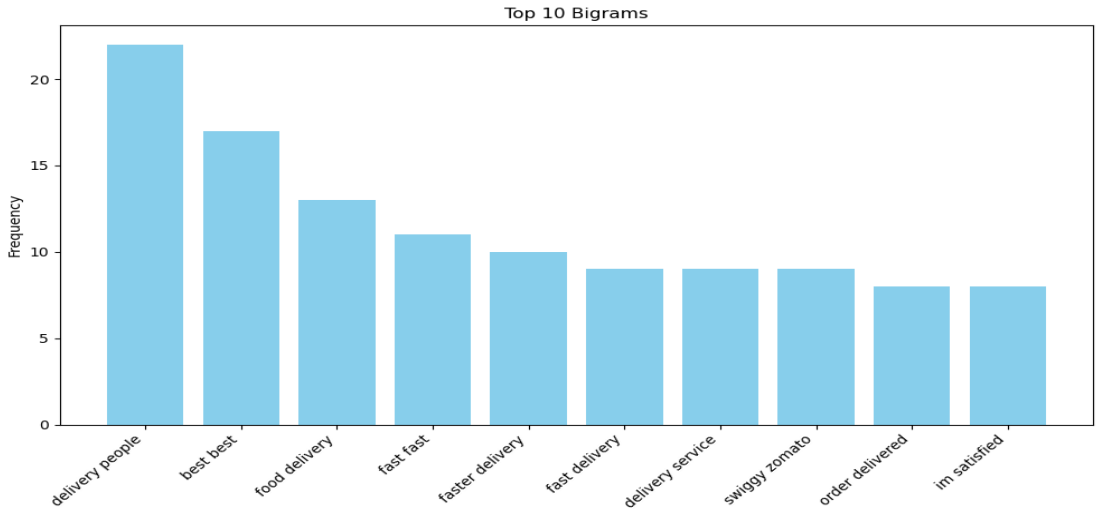
8.7.5 Gradient Boost Classifier

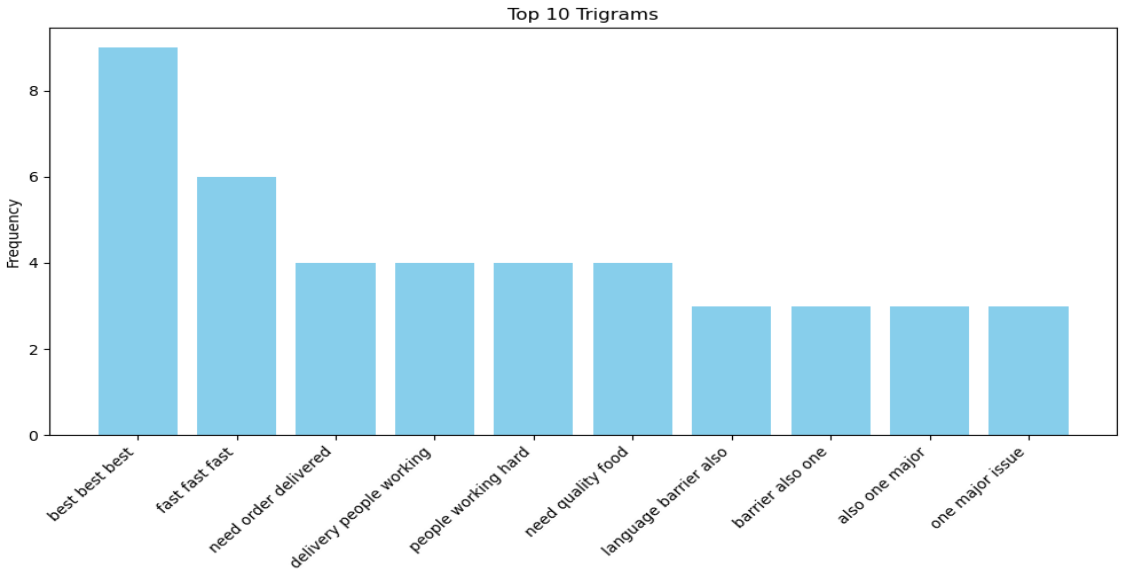
```
Confusion Matrix:
[[23  3]
 [12 79]]

Classification Report:
              precision    recall  f1-score   support

           0       0.66      0.88      0.75        26
           1       0.96      0.87      0.91        91

    accuracy                           0.87       117
   macro avg       0.81      0.88      0.83       117
weighted avg       0.90      0.87      0.88       117


Accuracy Score:
0.8717948717948718
```



8.7.6 N-gram Analysis (Uni-gram, Bi-gram and Tri-gram)

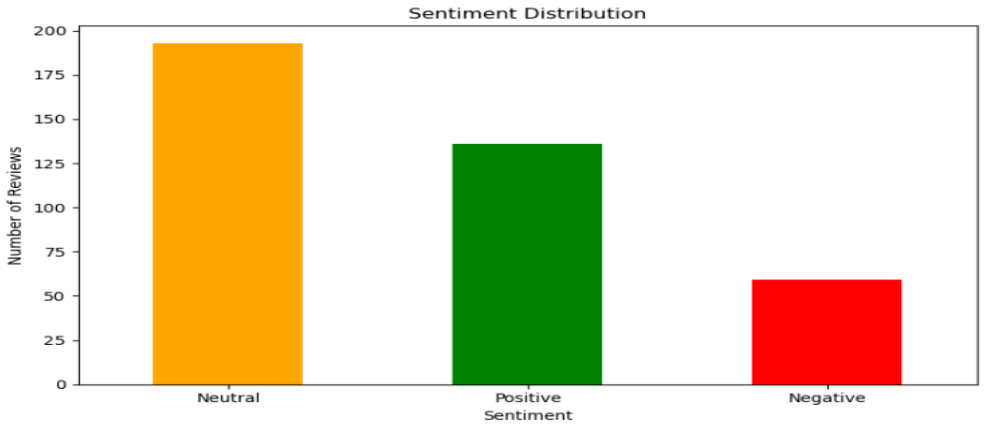






8.7.7 Sentiment Count

```
Defaulting to user installation because normal site-packages is not writeable
Requirement already satisfied: textblob in c:\users\jayanth\appdata\roaming\python\python312\site-packages (0.18.0.post0)
Requirement already satisfied: nltk>=3.8 in e:\anacondafiles\lib\site-packages (from textblob) (3.8.1)
Requirement already satisfied: click in e:\anacondafiles\lib\site-packages (from nltk>=3.8->textblob) (8.1.7)
Requirement already satisfied: joblib in e:\anacondafiles\lib\site-packages (from nltk>=3.8->textblob) (1.4.2)
Requirement already satisfied: regex>=2021.8.3 in e:\anacondafiles\lib\site-packages (from nltk>=3.8->textblob) (2023.10.3)
Requirement already satisfied: tqdm in e:\anacondafiles\lib\site-packages (from nltk>=3.8->textblob) (4.66.4)
Requirement already satisfied: colorama in e:\anacondafiles\lib\site-packages (from click->nltk>=3.8->textblob) (0.4.6)
Sentiment Counts:
 Sentiment
Neutral     193
Positive    136
Negative     59
Name: count, dtype: int64
```

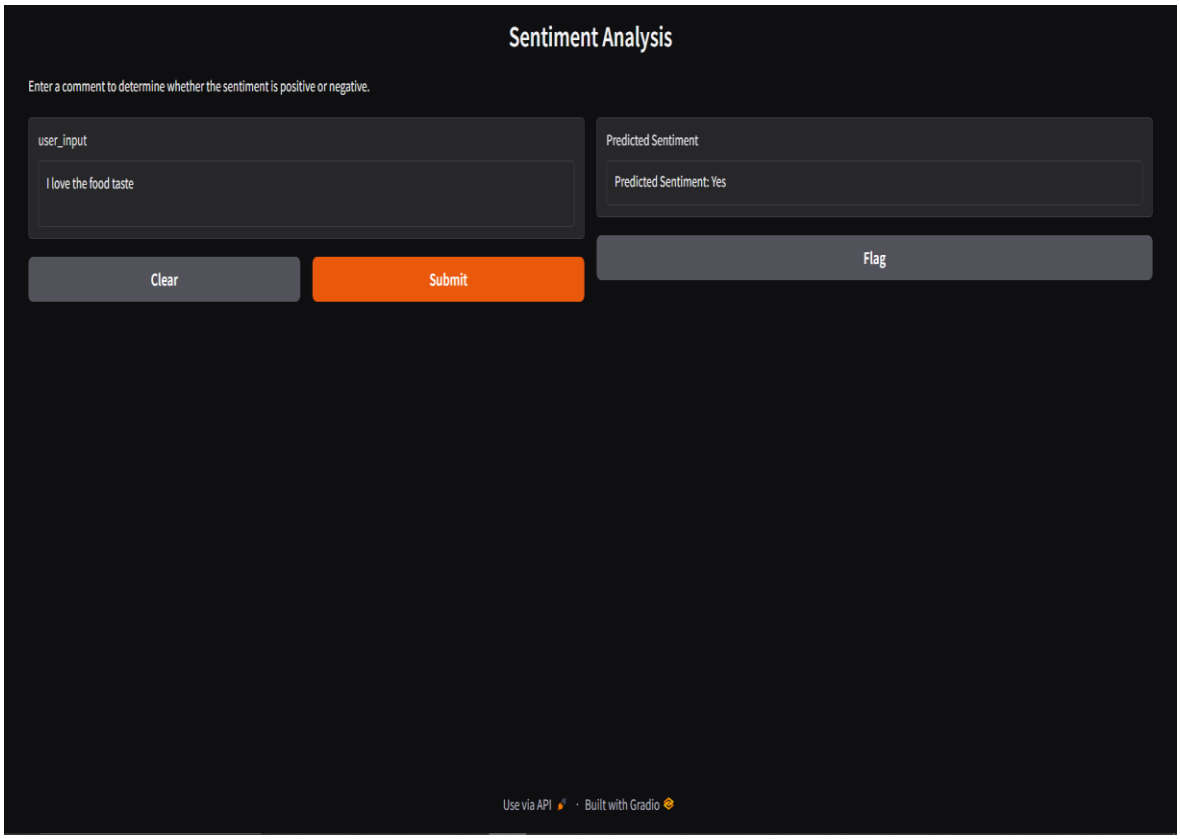



```
Sentiment analysis results saved to 'reviews_with_sentiment.csv'
```

8.7.6 Naïve Bayes and Cross Validation Test

```
<>:12: SyntaxWarning: invalid escape sequence '\o'
<>:12: SyntaxWarning: invalid escape sequence '\o'
C:\Users\Jayanth\AppData\Local\Temp\ipykernel_23224\1381401220.py:12: SyntaxWarning: invalid escape sequence '\o'
  file_path = 'D:\onlinedeliverydata.csv'  # Replace with your file path
Accuracy on Test Set: 0.89
Cross-Validation Accuracy: 0.89

Enter a comment to analyze sentiment (or type 'exit' to quit):  good taste
Predicted Sentiment: Yes

Enter a comment to analyze sentiment (or type 'exit' to quit):  bad taste
Predicted Sentiment: No

Enter a comment to analyze sentiment (or type 'exit' to quit):  late delivery
Predicted Sentiment: No

Enter a comment to analyze sentiment (or type 'exit' to quit):  fast delivery
Predicted Sentiment: Yes

Enter a comment to analyze sentiment (or type 'exit' to quit):  faster delivery
Predicted Sentiment: No

Enter a comment to analyze sentiment (or type 'exit' to quit):  exit
```

8.7.7 User Interface for sentiment analysis



Chapter-9
References

[1] M. A. Akasheh, N. Eleyan and G. Ertek, (2022) "A Predictive Data Analytics Methodology for Online Food Delivery," 2022 Ninth International Conference on Social Networks Analysis, Management and Security (SNAMS), Milan, Italy.

DOI: https://ieeexplore.ieee.org/abstract/document/10062613

[2] H. Alshahrani, H. Abdullah Mengash, M. Maashi, F. Kouki, A. Mahmud and M. A. Duhayyim, (2024), "Enhancing Online Food Service User Experience Through Advanced Analytics and Hybrid Deep Learning for Comprehensive Evaluation," in IEEE Access, vol. 12.

DOI: https://ieeexplore.ieee.org/document/10531271

[3] Dr. Ayyappa Chakravarthi M, Shaik Eesa Ruhulla Haq, Madapakula Venkata Anil, Bathula Venkata Vamsi, Gogireddy Venkata Reddy, (2024),"Forecasting Food Delivery Time: An Exploration of Predictive Models and Factors Impacting Delivery Time Estimation", IJARCCE International Journal of Advanced Research in Computer and Communication Engineering, vol. 13

DOI: https://doi.org/10.17148/IJARCCE.2024.13318.

[4] Jing-fang Chen, Ling Wang, Shengyao Wang, Xing Wang, Hao Ren, (2021), “An effective matching algorithm with adaptive tie-breaking strategy for online food delivery problem” IEEE Explore.

DOI: https://doi.org/10.1007/s40747-021-00340-x

Chapter-10
Worklog Sheet

International Centre for Education and Research (ICER), VIT-Bangalore

PROJECT
STUDENT'S WORKLOG SHEET

Student Name: Jayanth Kumar
Register Number: 24MSP3073
Project Title: Analysing Customers Preferences in Online Food Delivery Systems
Domain: Ecommerce and Analytics
Tools Learned / used: Machine Learning and Text Analysis

Day	Date	Task	Student's Remarks	Signature of Student with Date	Signature of the Project Mentor
Day 1	25/11/2024	Domain Selection Base Paper Selection, Finalising the Title and Problem statement	finalising dataset and title	25/11/24	26/11/24.
Day 2	26/11/2024	Final Abstract (Objective, Dataset, methodology, Tools, Expected Outcome) and Review of Related Literature	zeroth review abstract review of literature	26/11/24	26/11/24
Day 3	27/11/2024	Approach to the Problem write up 2 page (Aim, Research Objective, Research questions, Algorithm, Tools, Dataset, proposed architecture, Expected outcome, References). Perform EDA	explained abstract review of literature along with related papers	27/11/24	27/11/24
Day 4	28/11/2024	pre processing	python preprocessing	28/11/24	28/11/24
Day 5	29/11/2024	Implementation	Exploratory data analysis	29/11/24	29/11/24.

Day 6	30/11/24	Implementation	EDA	30/11/24	
Day 7	01/12/24	Implementation	EDA	01/12/24	
Day 8	02/12/24	Implementation	First review presentation	02/12/24	2/12/24.
Day 9	03/12/24	Second Review	Modelling Building	03/12/24	3/12/24.
Day 10	04/12/24	Results and Discussion	Data Developing UI for Sentiment analysis	04/12/24	4/12/24
Day 11	04/12/24	Conclusion and future scope	drafting Conclusion and future scope	04/12/24	04/12/24

Day 12	09/12/24	Rough Draft of Report submission	preparing report draft	09/12/24	9/12/24
Day 13	10/12/24	PPT for Final Review Submission	final review PPT preparation	10/12/24	10/12/24
Day 14	11/12/24 - 12/12/24	PPT for Final Review Submission and Project Report Submission on approval by Project Mentor	preparation of report	12/12/24	12/12/24.
Day 15	13/12/24	Final Review	final review	13/12/24.	13/12/24