Project Title

Semester project

Session 2023-2027

BS in Artificial Intelligence



Department of Software Engineering

Faculty of Computer Science & Information Technology

The Superior University, Lahore

FALL 2024

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Type (Nature of project) | | | [ ✓ ] **D**evelopment [ ] **R**esearch [ ] **R**&**D** | | |
| Area of specialization | | | Sentiment Analysis | | |
| **Project Group Members** | | | | | |
| Sr.# | Roll. # | Student Name | | Email ID | \*Signature |
| (i) | 113 | Warisha Hassan | | warishahassan506@gmail.com | Warisha |

**Table of Contents**

[Chapter 1 3](#_Toc184232380)

[**Introduction** 3](#_Toc184232381)

[**Data Loading and Understanding** 4](#_Toc184232382)

[**Visualization of Ratings** 4](#_Toc184232383)

[**Text Preprocessing** 4](#_Toc184232384)

[**Word Cloud Visualization** 4](#_Toc184232385)

[**Feature Extraction** 5](#_Toc184232386)

[**Machine Learning Model Training** 5](#_Toc184232387)

[**Evaluation and Visualization** 5](#_Toc184232388)

[**Sentiment Classification** 5](#_Toc184232389)

[**Conclusion** 5](#_Toc184232390)

[Chapter 2 6](#_Toc184232391)

[**Tool & Technology** 6](#_Toc184232392)

[**1.Programming Language:** 6](#_Toc184232393)

[**2. Libraries and Frameworks** 6](#_Toc184232394)

[**3. Development Environment** 7](#_Toc184232395)

[**4. Dataset** 7](#_Toc184232396)

[**5. Algorithms** 7](#_Toc184232397)

[**6. Preprocessing Techniques** 7](#_Toc184232398)

[**7. Operating System** 7](#_Toc184232399)

[**Chapter 3** 8](#_Toc184232400)

[**Implementation Code** 8](#_Toc184232401)

[**Code** 8](#_Toc184232402)

[Chapter 4 12](#_Toc184232403)

[**Result** 12](#_Toc184232404)

**Table of Figures**

[**Figure 1:** Dataset Overview 12](#_Toc184232314)

[**Figure 2 :** Common Words and Their Frequencies 13](#_Toc184232315)

[**Figure 3:**Model Performance Metrics 13](#_Toc184232316)

[**Figure 4:**Sentiment Classification Results 14](#_Toc184232317)

# 

# Chapter 1

**Introduction**

In the digital era, sentiment analysis has become a vital tool for businesses, organizations, and individuals to understand the opinions and feelings expressed in text data. This sentiment analysis project is designed to process and classify textual data (restaurant reviews) into different categories of sentiment, such as "positive" and "negative." By leveraging natural language processing (NLP) techniques and machine learning algorithms, this project automates the process of analyzing customer feedback, enabling businesses to gain actionable insights efficiently.

The primary objective of this project is to classify reviews based on their sentiment using a machine learning model, specifically the **Multinomial Naive Bayes classifier**, which is well-suited for text classification tasks. This program involves several systematic steps: data cleaning, feature extraction, model training, evaluation, and visualization of results. Each of these stages employs robust methods and libraries like NLTK, Scikit-learn, Pandas, Seaborn, and Matplotlib to ensure an accurate and meaningful analysis.

**Data Loading and Understanding**

The analysis begins by importing the necessary libraries and loading a dataset containing restaurant reviews. The dataset consists of rows of text data (customer reviews) alongside numerical ratings. The first task is to assess the data for any missing or inconsistent values, as such issues can hinder the machine learning process. A comprehensive overview of the dataset, including its size, structure, and any null values, is provided to help identify potential preprocessing needs.

**Visualization of Ratings**

Understanding the data distribution is crucial for identifying potential biases and trends. The program visualizes the distribution of ratings using a bar chart created with the Seaborn library. This provides an immediate understanding of the data spread and the prevalence of positive or negative reviews. Visualization serves as an important first step in exploring the sentiment landscape of the dataset.

**Text Preprocessing**

Text preprocessing is a critical step in preparing the data for analysis. This project incorporates advanced NLP techniques to clean and normalize the text data. Specifically:

1. **Tokenization**: The reviews are split into individual words (tokens) for further processing.
2. **Lowercasing**: All text is converted to lowercase to avoid inconsistencies due to case sensitivity.
3. **Stopword Removal**: Commonly used words like "the" and "and," which do not carry significant meaning, are removed to reduce noise.
4. **Punctuation and Number Removal**: Punctuation marks and numbers are eliminated as they do not contribute to the sentiment.

The result of preprocessing is a "cleaned" version of the review text, which is stored in a new column.

**Word Cloud Visualization**

A word cloud is generated to visualize the most frequently used words in the cleaned reviews. This step highlights the prominent themes and topics in customer feedback, providing insights into common sentiments expressed by reviewers. The word cloud is a powerful visual tool that supplements the numerical analysis of sentiment data.

**Feature Extraction**

The text data is then transformed into a numerical format that machine learning models can process. This is achieved using the **CountVectorizer** from Scikit-learn, which converts the cleaned text into a matrix of token counts. The vectorized features represent the frequency of words in each review, forming the basis for training the sentiment analysis model.

**Machine Learning Model Training**

The program employs the **Multinomial Naive Bayes classifier**, a probabilistic machine learning algorithm well-suited for text classification. The dataset is split into training and testing sets, with 80% of the data used for model training and 20% for testing. The model learns patterns from the training data to predict the sentiment of unseen reviews.

**Evaluation and Visualization**

After training, the model's performance is evaluated using metrics such as accuracy, precision, recall, and F1-score, displayed in a classification report. A confusion matrix is also plotted to provide a visual representation of the model's predictions compared to actual sentiments. This step helps identify strengths and weaknesses in the model's performance.

**Sentiment Classification**

To provide a business-oriented insight, the program classifies ratings into broader sentiment categories like "Positive" and "Negative." This classification aids in understanding the overall sentiment trends, which are visualized using a bar chart.

**Conclusion**

This sentiment analysis project is a complete pipeline for analyzing text data, from preprocessing raw text to training and evaluating a machine learning model. By leveraging natural language processing and supervised learning, the program achieves efficient and scalable sentiment classification. This implementation is valuable for industries like hospitality, retail, and social media analysis, where understanding customer sentiment is essential for decision-making and improving customer experiences.

Chapter 2

**Tool & Technology**

**1.Programming Language:**

* **Python**

**2. Libraries and Frameworks**

**(a) Data Manipulation and Analysis**

* **Pandas**: Used for loading, cleaning, and manipulating the dataset. It simplifies operations like handling missing values, creating new features, and analyzing data distributions.
* **NumPy**: Used for numerical operations and efficient handling of large arrays, especially during matrix operations in feature extraction.

**(b) Data Visualization**

* **Matplotlib**: Utilized for creating static visualizations like plots of model performance metrics and confusion matrices.
* **Seaborn**: A high-level library built on Matplotlib, used for creating visually appealing and informative statistical graphs, such as bar charts for rating distribution and sentiment trends.
* **WordCloud**: Generates a word cloud to visualize the most frequently occurring words in the cleaned review dataset, helping identify key themes in the text.

**(c) Natural Language Processing (NLP)**

* **NLTK (Natural Language Toolkit)**: Provides tools for tokenizing text, removing stopwords, and performing other text preprocessing tasks critical for preparing data for analysis.
  + stopwords: Removes common words that do not contribute to sentiment.
  + word\_tokenize: Tokenizes text into individual words.
* **Scikit-learn's CountVectorizer**: Converts text data into a matrix of token counts, enabling the use of machine learning algorithms.

**(d)Machine Learning**

* **Scikit-learn**: A machine learning library that supports a variety of models and evaluation tools:
  + **Multinomial Naive Bayes**: A probabilistic machine learning model used for text classification tasks.
  + **train\_test\_split**: Splits the dataset into training and testing sets to evaluate the model's performance.
  + **accuracy\_score, classification\_report, confusion\_matrix**: Tools for measuring the model’s accuracy and analyzing its predictions.

**3. Development Environment**

* **Anaconda/ Jupyter Notebook**: A development environment used for writing, running, and debugging the code. It offers a user-friendly interface and support for visualizations inline.

**4. Dataset**

* **Restaurant Reviews Dataset**: Contains textual customer reviews and corresponding ratings. The dataset is used to train the machine learning model to classify reviews based on their sentiment.

**5. Algorithms**

* **Multinomial Naive Bayes**: A supervised learning algorithm commonly used in text classification due to its efficiency and simplicity.

**6. Preprocessing Techniques**

* **Tokenization**: Breaking text into individual words.
* **Stopword Removal**: Removing irrelevant words that do not contribute to the sentiment.
* **Lowercasing**: Standardizing text to lowercase for consistency.
* **Punctuation Removal**: Eliminating symbols and special characters.

**7. Operating System**

* The project is platform-independent and can run on any operating system (e.g., Windows, Linux, macOS) that supports Python and its libraries.

# **Chapter 3**

# **Implementation Code**

## **Code**

import pandas as pd

import numpy as np

import matplotlib.pyplot as plt

import seaborn as sns

from wordcloud import WordCloud

from sklearn.feature\_extraction.text import CountVectorizer

from sklearn.model\_selection import train\_test\_split

from sklearn.naive\_bayes import MultinomialNB

from sklearn.metrics import classification\_report, confusion\_matrix,accuracy\_score

import nltk

from nltk.corpus import stopwords

from nltk.tokenize import word\_tokenize

import string

nltk.download('punkt')

nltk.download('stopwords')

df=pd.read\_csv('Restaurant\_reviews.csv')

print(f"Dataset contains {df.shape[0]} rows and {df.shape[1]} columns")

df.head(10)

print("Missing values per column:")

print(df.isnull().sum())

sns.countplot(data=df,x='Rating',palette='viridis')

plt.title('Distribution of Ratings')

plt.xlabel('Rating')

plt.ylabel('Count')

plt.show()

def preprocess\_text(text):

if not isinstance(text,str):

return ""

stop\_words = set(stopwords.words('english'))

tokens=word\_tokenize(text.lower())

tokens=[word for word in tokens if word not in stop\_words and word.isalpha()]

return " ".join(tokens)

df['Review']=df['Review'].fillna("")

df['Cleaned\_Review']=df['Review'].apply(preprocess\_text)

df[['Review','Cleaned\_Review']].head()

text\_combined = " ".join(df['Cleaned\_Review'])

wordcloud = WordCloud(width=1200,height=500,background\_color='white').generate(text\_combined)

plt.figure(figsize=(10,6))

plt.imshow(wordcloud, interpolation='bilinear')

plt.axis('off')

plt.title('Most Common Words in Reviews')

plt.show()

vectorizer = CountVectorizer(max\_features=1000)

X = vectorizer.fit\_transform(df['Cleaned\_Review']).toarray()

y = df['Rating']

X\_train, X\_test, y\_train, y\_test = train\_test\_split(X, y, test\_size=0.2, random\_state=42)

print(f"Training set size: {X\_train.shape[0]} rows")

print(f"Test set size: {X\_test.shape[0]} rows")

print('Unique Ratings Before Cleaning:')

print(df['Rating'].value\_counts())

model = MultinomialNB()

model.fit(X\_train, y\_train)

df=df[df['Rating'].apply(lambda x: str(x).replace('.','',1).isdigit())]

df['Rating'] = df['Rating'].astype(float)

df['Rating'] = df['Rating'].round().astype(int)

df = df.dropna(subset=['Rating'])

y\_prediction = model.predict(X\_test)

print("Classification Report:")

print(classification\_report(y\_test, y\_prediction))

Confusion\_matrix = confusion\_matrix(y\_test, y\_prediction)

sns.heatmap(Confusion\_matrix, annot=True, fmt='d',cmap='Blues', xticklabels=sorted(y.unique()), yticklabels=sorted(y.unique()))

plt.title('Confusion Matrix')

plt.xlabel('Predicted')

plt.ylabel('Actual')

plt.show()

accuracy = accuracy\_score(y\_test, y\_prediction)

print(f"Model Accuracy: {accuracy: .2f}")

df['Sentiment'] = df['Rating'].apply(lambda x: 'Positive' if x > 3 else 'Negative')

sns.countplot(data=df, x='Sentiment', palette='coolwarm')

plt.title('Sentiment Distribution')

plt.xlabel('Sentiment')

plt.ylabel('Count')

plt.show()

# Balance the dataset

positive\_reviews = df[df['Sentiment'] == 'Positive']

negative\_reviews = df[df['Sentiment'] == 'Negative']

balanced\_df = pd.concat([positive\_reviews.sample(len(negative\_reviews)), negative\_reviews])

# Update X and y

X = vectorizer.fit\_transform(balanced\_df['Cleaned\_Review']).toarray()

y = balanced\_df['Rating']

def test\_custom\_review(review, model, vectorizer):

stop\_words = set(stopwords.words('english'))

tokens = word\_tokenize(review.lower())

tokens = [word for word in tokens if word not in stop\_words and word.isalpha()]

cleaned\_review = " ".join(tokens)

review\_vector = vectorizer.transform ([cleaned\_review]).toarray()

# Predict sentiment

prediction = model.predict (review\_vector)

sentiment = 'Negative' if prediction [0] >= 4 else 'Positive'

print (f" Review: {review}")

print (f"Predicted Sentiment: {sentiment}")

custom\_review = "Very Disgusting"

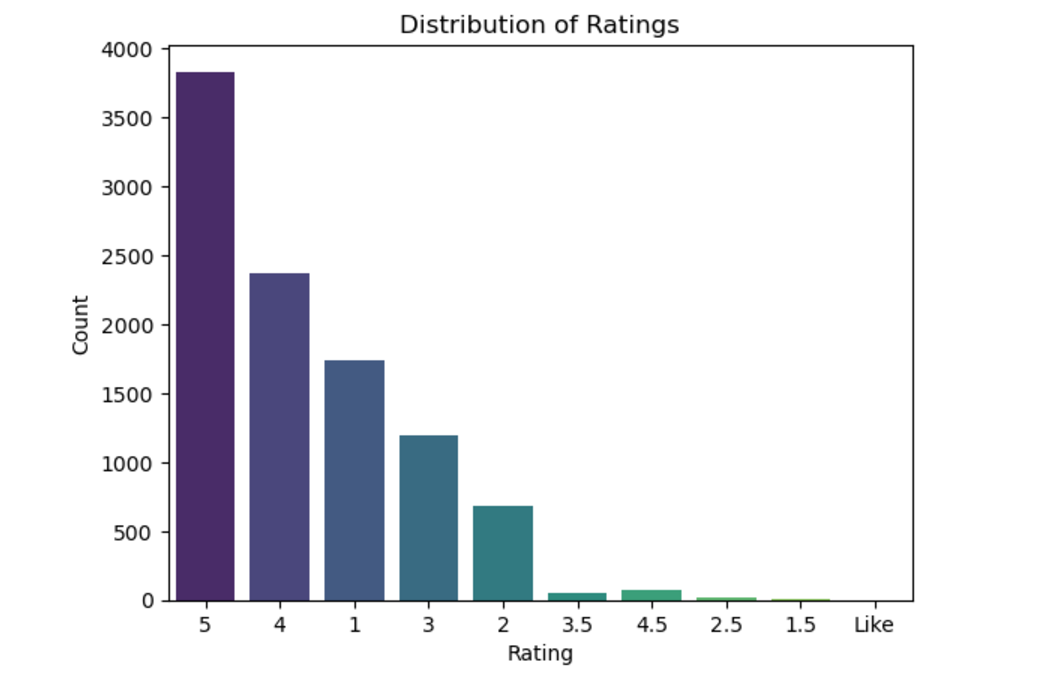
test\_custom\_review (custom\_review, model, vectorizer)

custom\_review = "Delicious and very good"

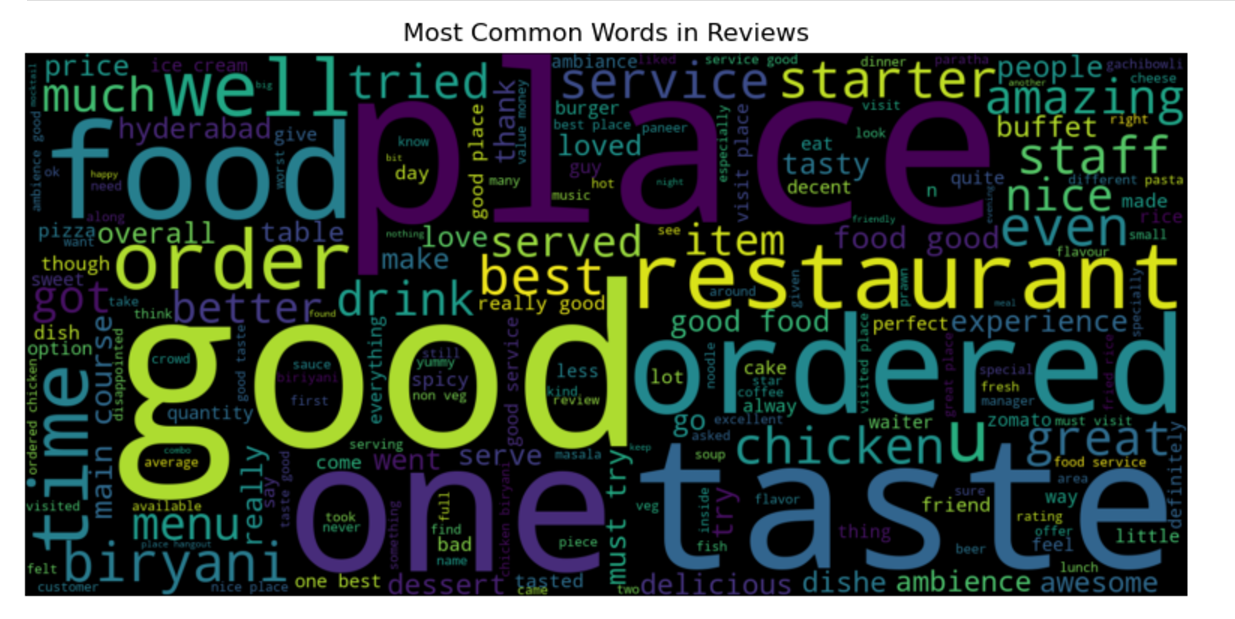
test\_custom\_review (custom\_review, model, vectorizer)

Chapter 4

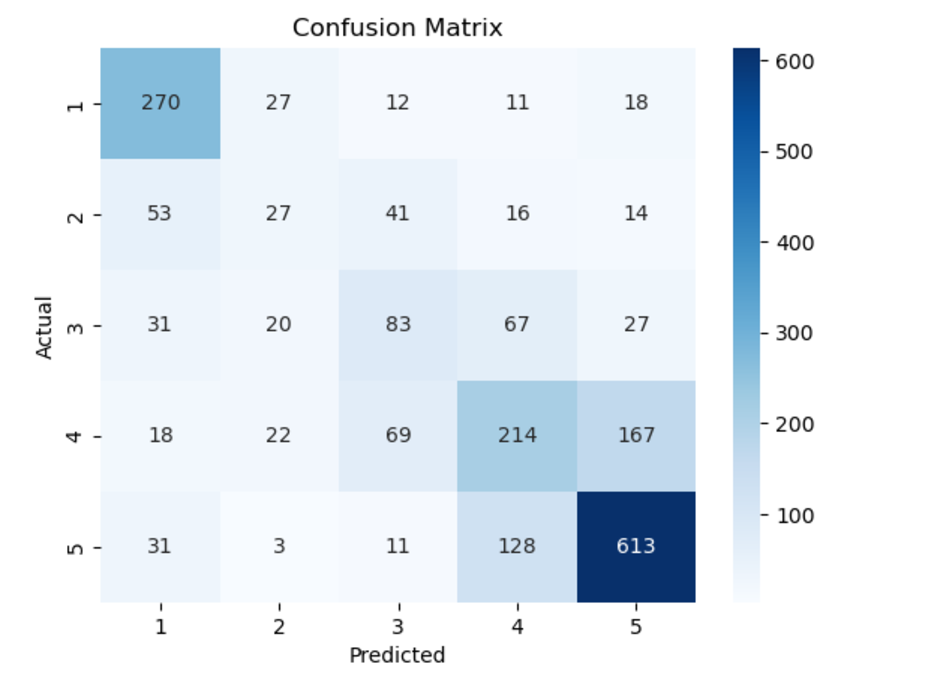
## **Result**



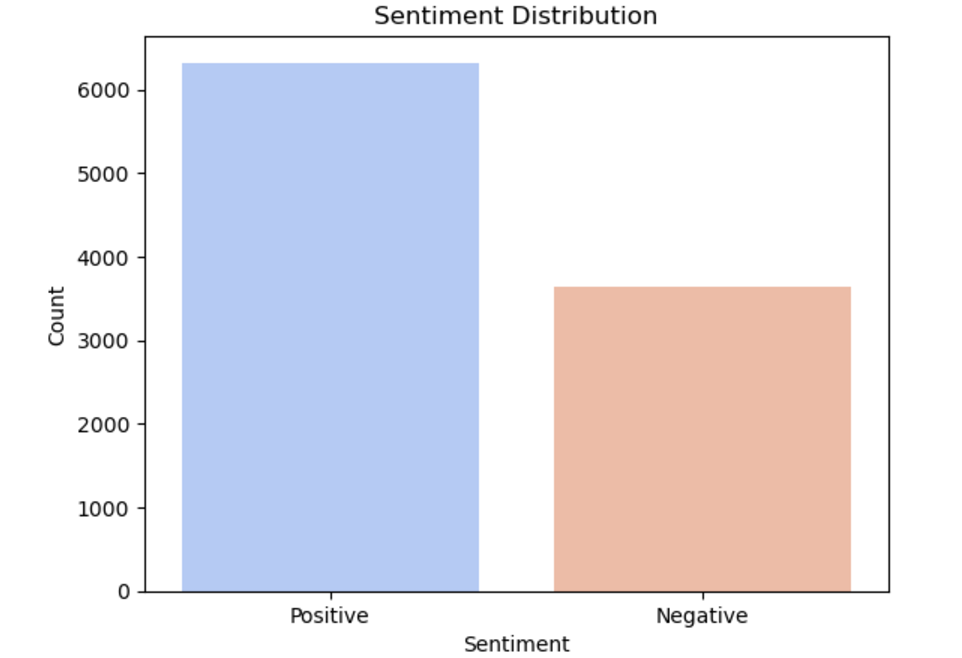
**Figure 1:** Dataset Overview



**Figure 2 :** Common Words and Their Frequencies



**Figure 3:**Model Performance Metrics



**Figure 4:**Sentiment Classification Results