AISSMS

COLLEGE OF ENGINEERING

ज्ञानम् सकलजनहिताय

Approved by AICTE, New Delhi, Recognized by Government of Maharashtra
Affiliated to Savitribai Phule Pune University and recognized 2(f) and 12(B) by UGC
(Id.No. PU/PN/Engg./093 (1992)
Accredited by NAAC with "A+" Grade | NBA - 7 UG Programmes

Department of Computer Engineering

"DSBDA Mini-project Report"

Submitted in partial fulfillment of the requirements for the degree of

BACHELOR OF ENGINEERING

In

COMPUTER ENGINEERING

Submitted By

Name of the Student: Piyusha Rajendra Supe

Roll No: 23CO315

Under the Guidance of

Mr. V. S. Gunjal

Ms. V. M. Kanavdey

Mr. A. J. Kadam

ALL INDIA SHRI SHIVAJI MEMORIAL SOCIETY'S COLLEGE OF ENGINEERING

PUNE-411001

Academic Year: 2024-25(Term-II)

Savitribai Phule Pune University

AISSMS
COLLEGE OF ENGINEERING
ज्ञानम् सकलजनहिताय
Approved by AICTE, New Delhi, Recognized by Government of Maharashtra
Affiliated to Savitribai Phule Pune University and recognized 2(f) and 12(B) by UGC
(Id.No. PU/PN/Engg./093 (1992)
Accredited by NAAC with "A+" Grade | NBA - 7 UG Programmes

Department of Computer Engineering

CERTIFICATE

This is to certify that **Piyusha Rajendra Supe** from **Third Year Computer Engineering** has successfully completed her work titled "**DSBDA Mini-project**" at AISSMS College of Engineering, Pune in the partial fulfillment of the Bachelor's Degree in Engineering.

Mr. V. S. Gunjal	**Ms. V. M. Kanavdey**	**Mr. A. J. Kadam**	**Dr. S. V. Athawale**	**Dr. D. S. Bormane**
(Faculty Guide)	(Faculty Guide)	(Faculty Guide)	(Head of Department)	(Principal)
Computer Engineering	Computer Engineering	Computer Engineering	Computer Engineering	AISSMSCOE, Pune

ACKNOWLEDGEMENT

It is with immense gratitude and respect that I take this opportunity to acknowledge the invaluable support and guidance I have received during the course of my mini project in Data Science and Big Data Analytics. This project has been a significant learning experience for me, and its successful completion would not have been possible without the unwavering assistance of many individuals.

First and foremost, I would like to express my heartfelt thanks to the entire faculty team for their continuous support and encouragement. Their collective guidance, along with the constructive feedback provided throughout the project, has played a vital role in shaping the final outcome. I am sincerely grateful for the wealth of knowledge shared, which helped me enhance my skills and understanding in the field of data science and big data analytics.

Additionally, I would like to extend my gratitude to all the staff members, colleagues, and peers who were a part of this journey. Their contributions, discussions, and support were truly invaluable in enhancing the quality of my work and my learning experience.

Finally, I am profoundly thankful to everyone who helped me refine my understanding of the subject, offered insights, and motivated me to push through challenges. This project has been a fulfilling and enriching experience, and I am grateful to all those who played a role in making it a success.

Academic Year: 2024-2025

Piyusha Rajendra Supe (23CO315)

TABLE OF CONTENTS

ABSTRACT

This project focuses on performing sentiment analysis on tweets related to data science, with the goal of classifying them into positive and negative sentiments. Sentiment analysis is a key application of Natural Language Processing (NLP) that allows for the extraction of meaningful insights from large volumes of text data. By analysing the sentiments expressed in social media platforms like Twitter, organizations and researchers can gain valuable feedback on public opinion, emerging trends, and overall sentiment toward specific topics.

The project begins with data pre-processing, where the raw tweet data is cleaned by removing irrelevant elements such as URLs, special characters, hashtags, and mentions, followed by the conversion of the text to lowercase and tokenization. To convert the text into a numerical representation, **TF-IDF (Term Frequency-Inverse Document Frequency)** vectorization is applied, which helps in identifying the most important words in the dataset that contribute to the sentiment classification.

For the sentiment classification task, the project employs the **Multinomial Naïve Bayes** algorithm, a probabilistic model that is well-suited for text classification tasks. This model works by calculating the likelihood of a given word being associated with a specific sentiment, based on the training data. The sentiment classification pipeline is built using **Scikit-learn**, a widely used machine learning library, which facilitates the integration of pre-processing, feature extraction, and model training.

The performance of the model is evaluated using key metrics such as **accuracy**, **precision**, **recall**, and **F1-score**, which provide a comprehensive understanding of how well the model is classifying tweets. Additionally, several visualizations are created to interpret the results and offer insights into the data, including sentiment distribution plots, word clouds, and confusion matrices. These visual tools help to highlight the frequency of positive and negative tweets, the most common words associated with each sentiment, and the classification errors made by the model.

INTRODUCTION

This project aims to perform sentiment analysis on tweets related to data science by classifying them into positive and negative sentiments. Sentiment analysis is a key application of Natural Language Processing (NLP) that helps in understanding public opinion, trends, and emotions conveyed in text. The dataset, sourced from Kaggle, consists of real-world tweets, making it a relevant and practical resource for sentiment classification tasks.

Methodology:

The project involves several stages, starting with **data pre-processing**, where raw text data is cleaned by removing URLs, special characters, and unnecessary elements such as mentions and hashtags. The tweets are then converted to lowercase and tokenized. To transform the text data into numerical format, **TF-IDF (Term Frequency-Inverse Document Frequency) vectorization** is applied, which helps in extracting meaningful features for classification.

For sentiment classification, the **Multinomial Naïve Bayes** algorithm is used. This model is well-suited for text classification tasks due to its ability to handle large vocabularies and categorical data efficiently. The classification pipeline is built using **Scikit-learn**, ensuring a streamlined workflow from text processing to model training and evaluation.

Evaluation and Visualization:

The model's performance is assessed using **accuracy, precision, recall, and F1-score** to measure its effectiveness in classifying tweets correctly. Additionally, various visualizations are used to interpret the results, including:

- **Sentiment Distribution Plot:** Displays the number of positive and negative tweets.
- **Word Cloud:** Highlights the most frequently occurring words in tweets to identify common themes.
- **Confusion Matrix:** Provides insights into the classification errors made by the model.
- **TF-IDF Feature Importance Plot:** Helps understand which words contribute most to classification decisions.

PROBLEM STATEMENT

Title: Use the following dataset and classify tweets into positive and negative tweets.

https://www.kaggle.com/ruchi798/data-science-tweets

The problem statement for this project revolves around analysing and classifying tweets related to data science into positive and negative sentiments. With the increasing use of social media, a large volume of unstructured text data is generated daily, making it crucial to develop automated sentiment analysis tools to extract meaningful insights. The challenge lies in processing raw text data, handling noisy elements like hashtags and URLs, and effectively classifying sentiments using machine learning models. By leveraging Natural Language Processing (NLP) and machine learning techniques, this project aims to build a classifier that can efficiently distinguish between positive and negative tweets, providing valuable insights into public opinions in the data science domain

SYSTEM REQUIREMENTS

1. **Hardware Requirements**

- Processor: Intel Core i5 (or higher) / AMD Ryzen 5 (or higher)
- RAM: Minimum 8GB RAM (Recommended: 16GB for handling large datasets efficiently)
- Storage: Minimum 10GB free space (SSD recommended for faster processing)
- Graphics Processing Unit (GPU): Optional, but NVIDIA GPU with CUDA support can speed up deep learning-based sentiment analysis
- Internet Connection: Required for accessing online datasets and libraries

2. **Software Requirements**

- Operating System: Windows 10/11, Linux (Ubuntu 20.04 or later), or macOS
- Python Version: Python 3.7 or later
- Development Environment: Jupyter Notebook, VS Code, or PyCharm

3. **Python Libraries & Dependencies**

- Data Handling: `pandas`, `numpy`
- Text Processing & NLP: `re` (for regex), `nltk`, `wordcloud`, `textblob`
- Machine Learning: `scikit-learn` (for TfidfVectorizer, MultinomialNB, model evaluation)
- Visualization: `matplotlib`, `seaborn`
- Deep Learning (Optional): `tensorflow`, `keras` (if using LSTMs or transformers)

4. **Dataset Requirements**

- Source: Kaggle dataset (Data Science Tweets)
- File Format: CSV
- Features Required: `tweet` column for text analysis
- Preprocessing Needs: Cleaning tweets (removing URLs, mentions, special characters)

5. **Functional Requirements**

- Data Preprocessing: Convert text to lowercase, remove noise (hashtags, URLs, special characters)
- Feature Extraction: Convert text into numerical representation using TF-IDF
- Model Training: Use Multinomial Naïve Bayes for classification
- Performance Evaluation: Accuracy, precision, recall, F1-score, confusion matrix
- Visualization: Sentiment distribution, word cloud, confusion matrix, performance metrics

OVERVIEW

This project focuses on performing sentiment analysis on tweets related to data science, classifying them into positive and negative sentiments. Sentiment analysis, a key task in Natural Language Processing (NLP), helps understand public opinion and emotions expressed in text. The dataset, sourced from Kaggle, contains real-world tweets, making it a relevant resource for this task.

Key Steps:

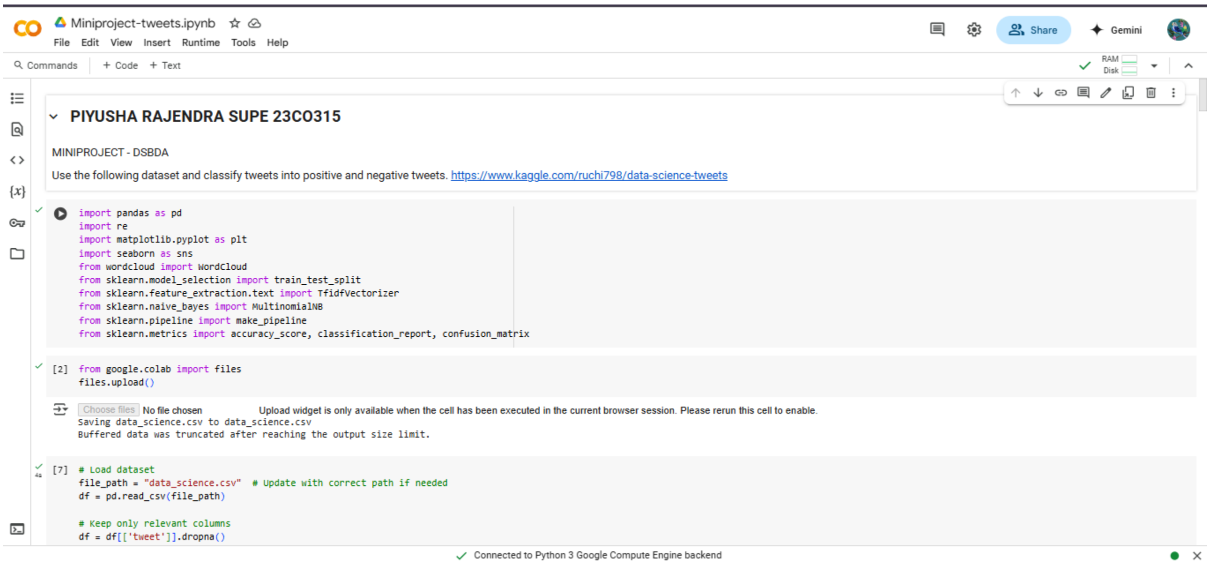
1. **Data Preprocessing**: The raw tweet data is cleaned by removing URLs, special characters, mentions, and hashtags. The text is then tokenized and converted to lowercase.
2. **Feature Extraction**: **TF-IDF (Term Frequency-Inverse Document Frequency)** vectorization is used to convert the text data into numerical features, capturing important words for classification.
3. **Sentiment Classification**: The **Multinomial Naïve Bayes** algorithm is applied to classify tweets as positive or negative, leveraging its efficiency in text classification tasks.
4. **Model Evaluation**: The model's performance is evaluated using accuracy, precision, recall, and F1-score, with visualizations like sentiment distribution plots, word clouds, and confusion matrices to interpret the results.

Future Improvements:

Future enhancements could include using advanced NLP models like **LSTMs** or **Transformer-based models (BERT, RoBERTa)** for more accurate sentiment analysis. Additionally, fine-tuning hyper parameters and using word embedding’s like **Word2Vec** or **Glove** could improve classification accuracy.

Overall, the project demonstrates how machine learning and NLP can be used to analyse social media sentiment, offering insights into public opinions on data science topics.

IMPLEMENTATION



PIYUSHA RAJENDRA SUPE 23CO315

MINIPROJECT - DSBDA

Use the following dataset and classify tweets into positive and negative tweets. https://www.kaggle.com/ruchi798/data-science-tweets

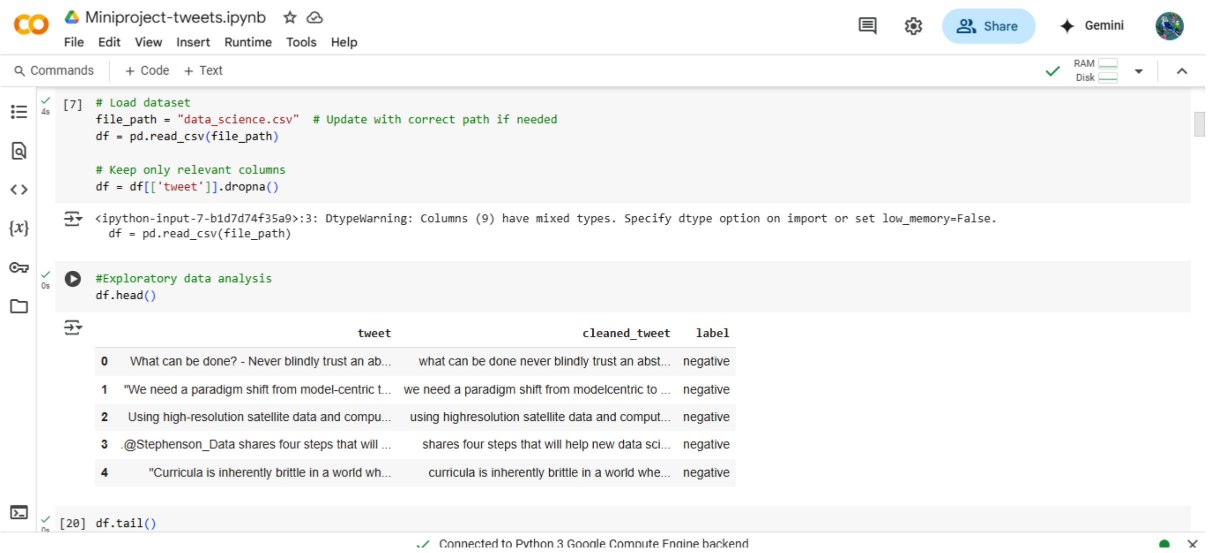
```
import pandas as pd
import re
import matplotlib.pyplot as plt
import seaborn as sns
from wordcloud import WordCloud
from sklearn.model_selection import train_test_split
from sklearn.feature_extraction.text import TfidfVectorizer
from sklearn.naive_bayes import MultinomialNB
from sklearn.pipeline import make_pipeline
from sklearn.metrics import accuracy_score, classification_report, confusion_matrix
```

```
[2] from google.colab import files
    files.upload()
```

```
Choose files No file chosen        Upload widget is only available when the cell has been executed in the current browser session. Please rerun this cell to enable.
Saving data_science.csv to data_science.csv
Buffered data was truncated after reaching the output size limit.
```

```
[7] # Load dataset
    file_path = "data_science.csv"  # Update with correct path if needed
    df = pd.read_csv(file_path)

    # Keep only relevant columns
    df = df[['tweet']].dropna()
```

Connected to Python 3 Google Compute Engine backend



```
[7] # Load dataset
    file_path = "data_science.csv"  # Update with correct path if needed
    df = pd.read_csv(file_path)

    # Keep only relevant columns
    df = df[['tweet']].dropna()
```

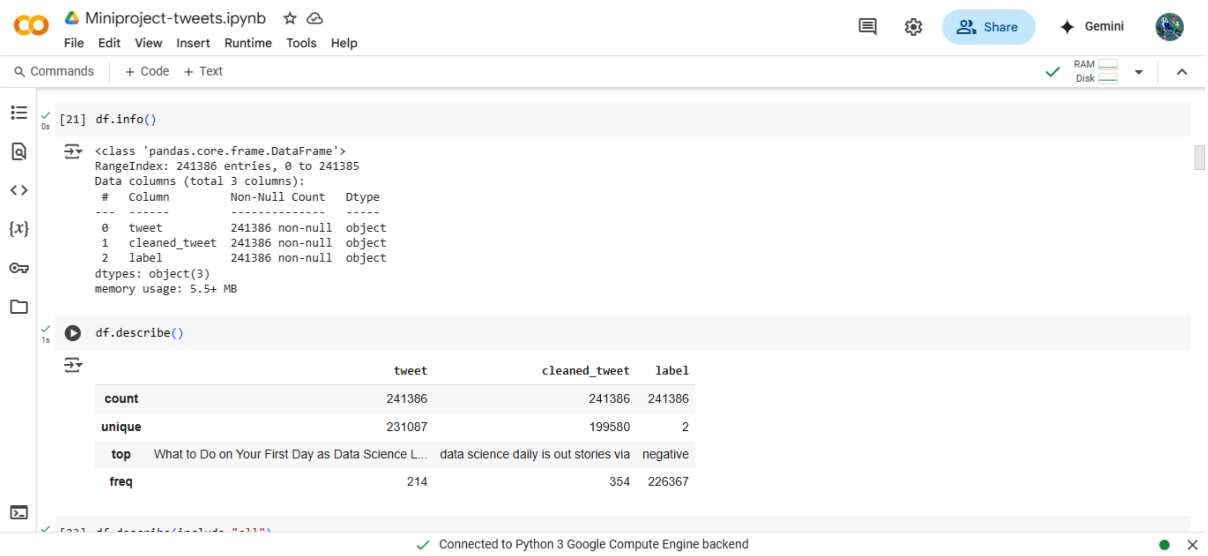
```
<ipython-input-7-b1d7d74f35a9>:3: DtypeWarning: Columns (9) have mixed types. Specify dtype option on import or set low_memory=False.
  df = pd.read_csv(file_path)
```

```
#Exploratory data analysis
df.head()
```

	tweet	cleaned_tweet	label
0	What can be done? - Never blindly trust an ab...	what can be done never blindly trust an abst...	negative
1	"We need a paradigm shift from model-centric t...	we need a paradigm shift from modelcentric to ...	negative
2	Using high-resolution satellite data and compu...	using highresolution satellite data and comput...	negative
3	.@Stephenson_Data shares four steps that will ...	shares four steps that will help new data sci...	negative
4	"Curricula is inherently brittle in a world wh...	curricula is inherently brittle in a world whe...	negative

```
[20] df.tail()
```

Connected to Python 3 Google Compute Engine backend



```
[21] df.info()
```

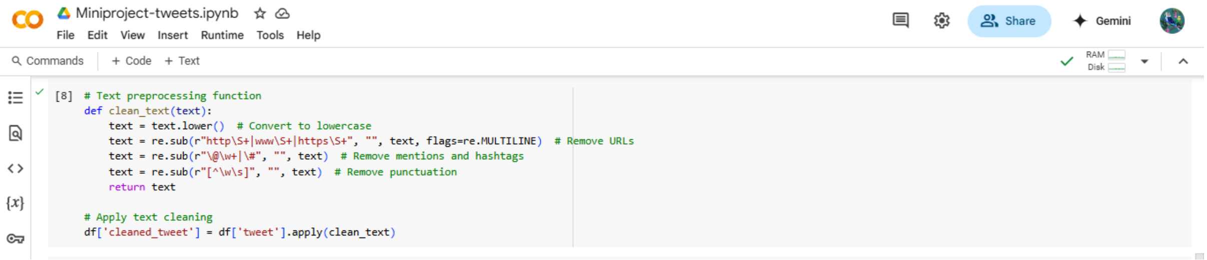
```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 241386 entries, 0 to 241385
Data columns (total 3 columns):
 #   Column         Non-Null Count   Dtype
---  ------         --------------   -----
 0   tweet          241386 non-null  object
 1   cleaned_tweet  241386 non-null  object
 2   label          241386 non-null  object
dtypes: object(3)
memory usage: 5.5+ MB
```

```
df.describe()
```

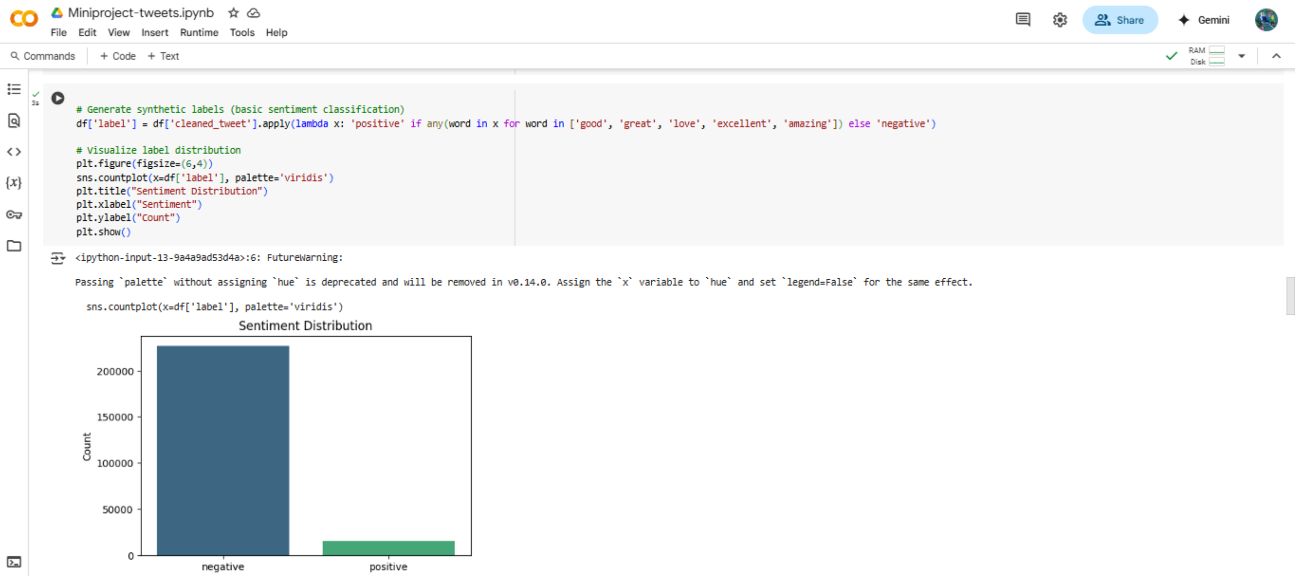
	tweet	cleaned_tweet	label
count	241386	241386	241386
unique	231087	199580	2
top	What to Do on Your First Day as Data Science L...	data science daily is out stories via	negative
freq	214	354	226367

```
[22] df.describe(include="all")
```

Connected to Python 3 Google Compute Engine backend

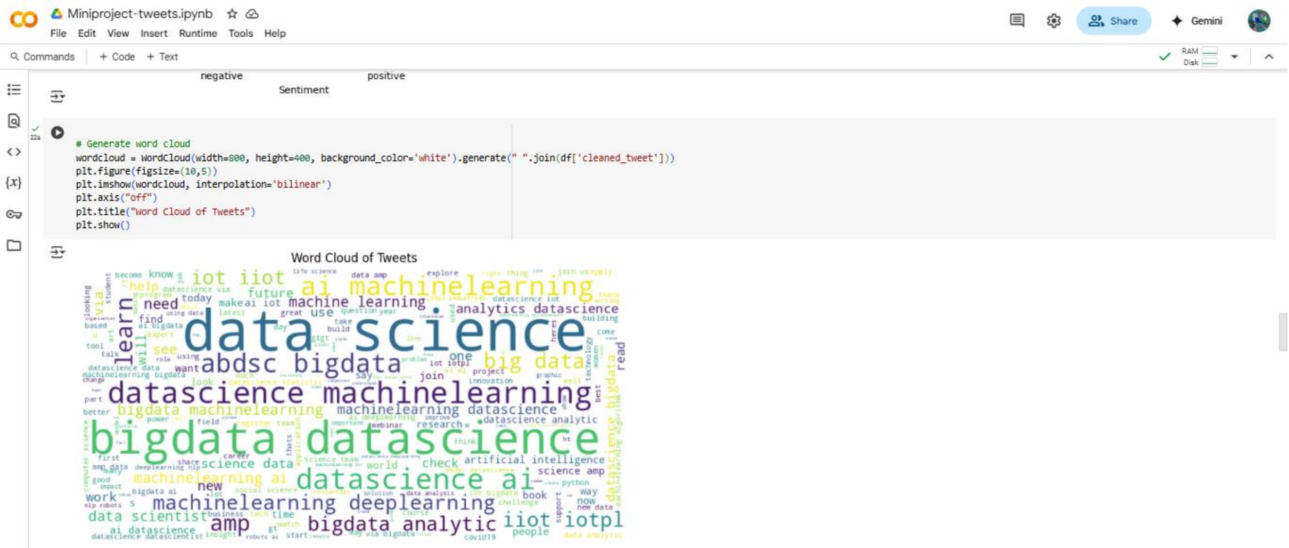


```
# Text preprocessing function
def clean_text(text):
    text = text.lower()  # Convert to lowercase
    text = re.sub(r"http\S+|www\S+|https\S+", "", text, flags=re.MULTILINE)  # Remove URLs
    text = re.sub(r"\@\w+|\#", "", text)  # Remove mentions and hashtags
    text = re.sub(r"[^\w\s]", "", text)  # Remove punctuation
    return text

# Apply text cleaning
df['cleaned_tweet'] = df['tweet'].apply(clean_text)
```

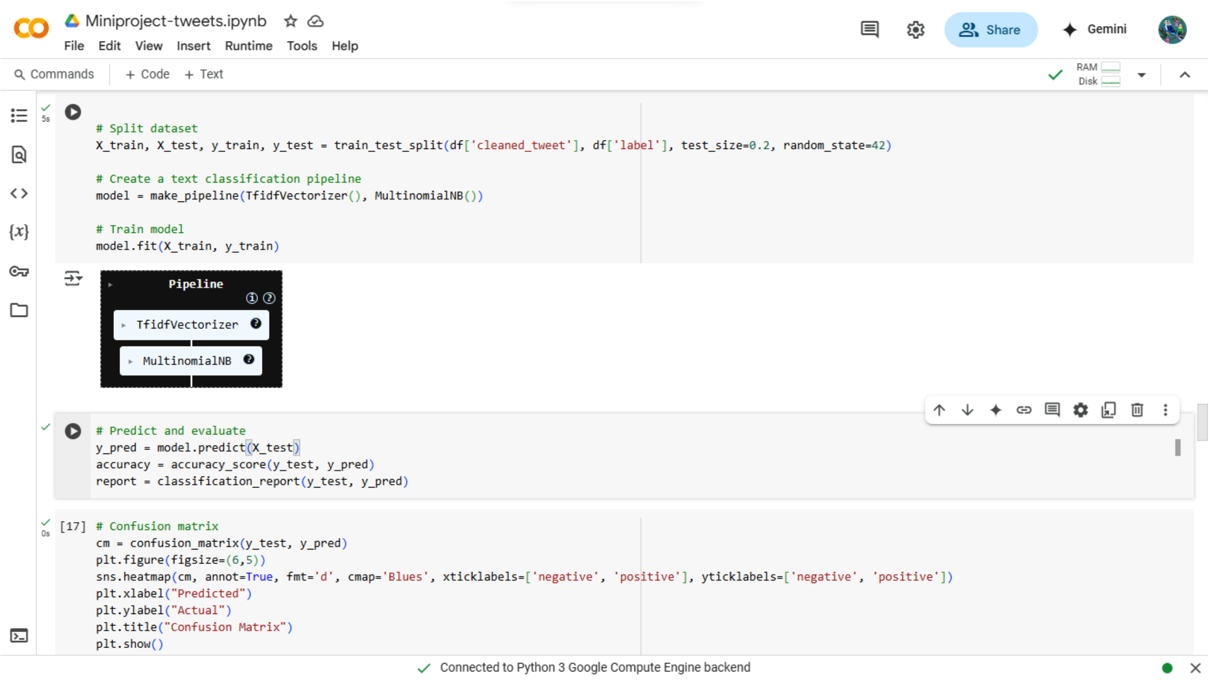


```
# Generate synthetic labels (basic sentiment classification)
df['label'] = df['cleaned_tweet'].apply(lambda x: 'positive' if any(word in x for word in ['good', 'great', 'love', 'excellent', 'amazing']) else 'negative')

# Visualize label distribution
plt.figure(figsize=(6,4))
sns.countplot(x=df['label'], palette='viridis')
plt.title("Sentiment Distribution")
plt.xlabel("Sentiment")
plt.ylabel("Count")
plt.show()
```

```
<ipython-input-13-9a4a9ad53d4a>:6: FutureWarning:

Passing `palette` without assigning `hue` is deprecated and will be removed in v0.14.0. Assign the `x` variable to `hue` and set `legend=False` for the same effect.

  sns.countplot(x=df['label'], palette='viridis')
```



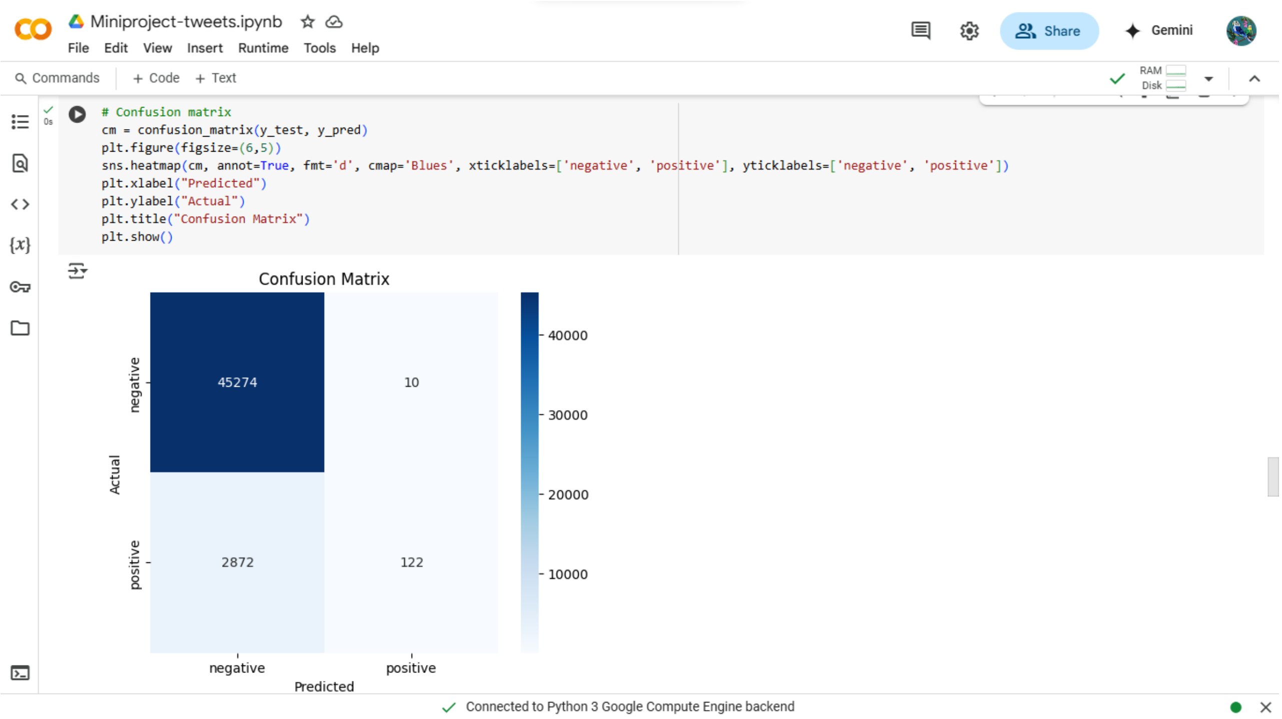
```
# Generate word cloud
wordcloud = WordCloud(width=800, height=400, background_color='white').generate(" ".join(df['cleaned_tweet']))
plt.figure(figsize=(10,5))
plt.imshow(wordcloud, interpolation='bilinear')
plt.axis("off")
plt.title("Word Cloud of Tweets")
plt.show()
```



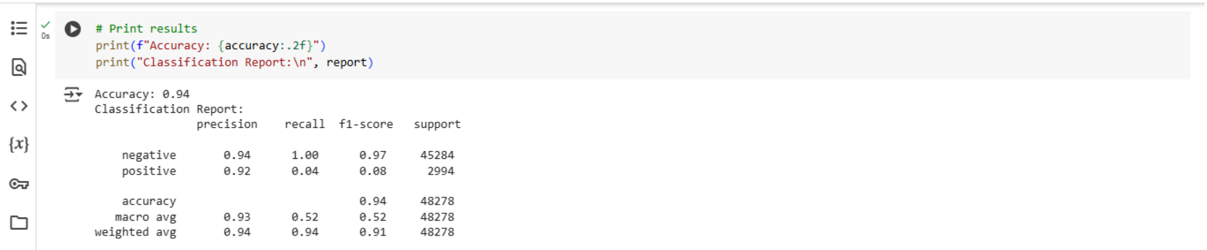
```
# Split dataset
X_train, X_test, y_train, y_test = train_test_split(df['cleaned_tweet'], df['label'], test_size=0.2, random_state=42)

# Create a text classification pipeline
model = make_pipeline(TfidfVectorizer(), MultinomialNB())

# Train model
model.fit(X_train, y_train)
```

```
# Predict and evaluate
y_pred = model.predict(X_test)
accuracy = accuracy_score(y_test, y_pred)
report = classification_report(y_test, y_pred)
```

```
# Confusion matrix
cm = confusion_matrix(y_test, y_pred)
plt.figure(figsize=(6,5))
sns.heatmap(cm, annot=True, fmt='d', cmap='Blues', xticklabels=['negative', 'positive'], yticklabels=['negative', 'positive'])
plt.xlabel("Predicted")
plt.ylabel("Actual")
plt.title("Confusion Matrix")
plt.show()
```

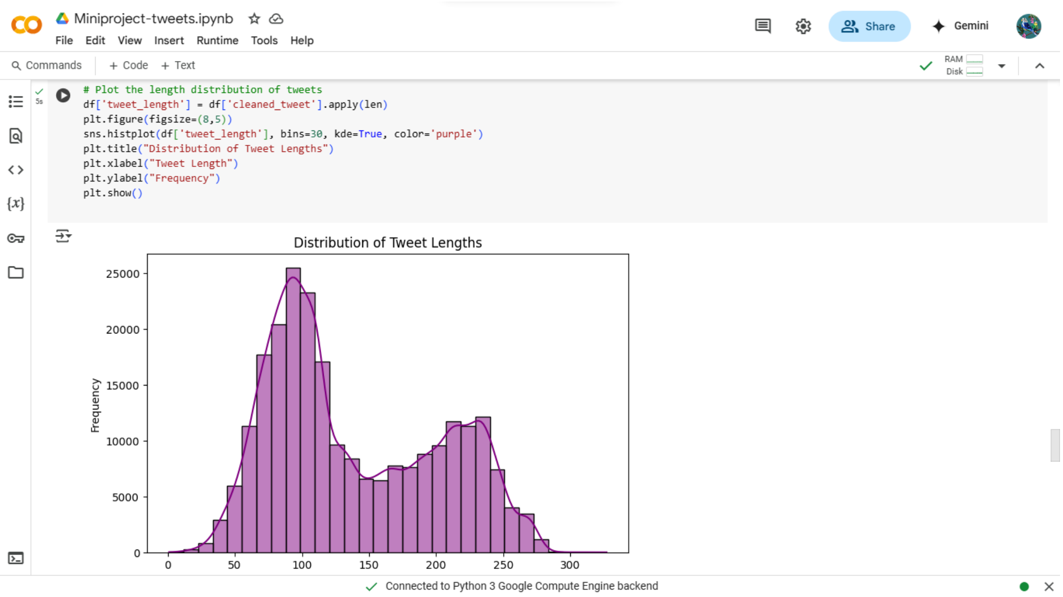


```
# Confusion matrix
cm = confusion_matrix(y_test, y_pred)
plt.figure(figsize=(6,5))
sns.heatmap(cm, annot=True, fmt='d', cmap='Blues', xticklabels=['negative', 'positive'], yticklabels=['negative', 'positive'])
plt.xlabel("Predicted")
plt.ylabel("Actual")
plt.title("Confusion Matrix")
plt.show()
```

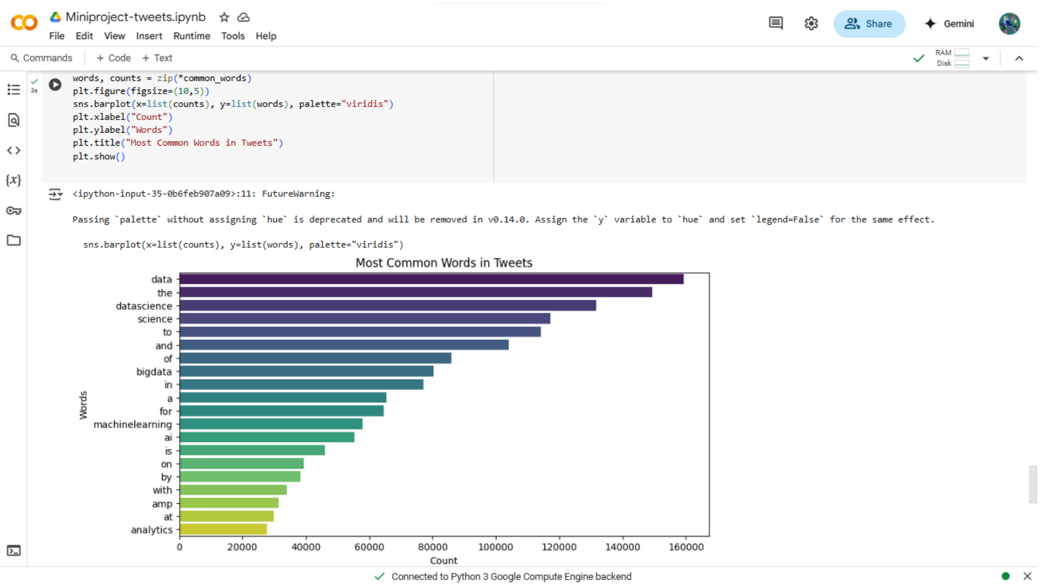


```
# Print results
print(f"Accuracy: {accuracy:.2f}")
print("Classification Report:\n", report)
```

```
Accuracy: 0.94
Classification Report:
               precision    recall  f1-score   support

    negative       0.94      1.00      0.97     45284
    positive       0.92      0.04      0.08      2994

    accuracy                           0.94     48278
   macro avg       0.93      0.52      0.52     48278
weighted avg       0.94      0.94      0.91     48278
```

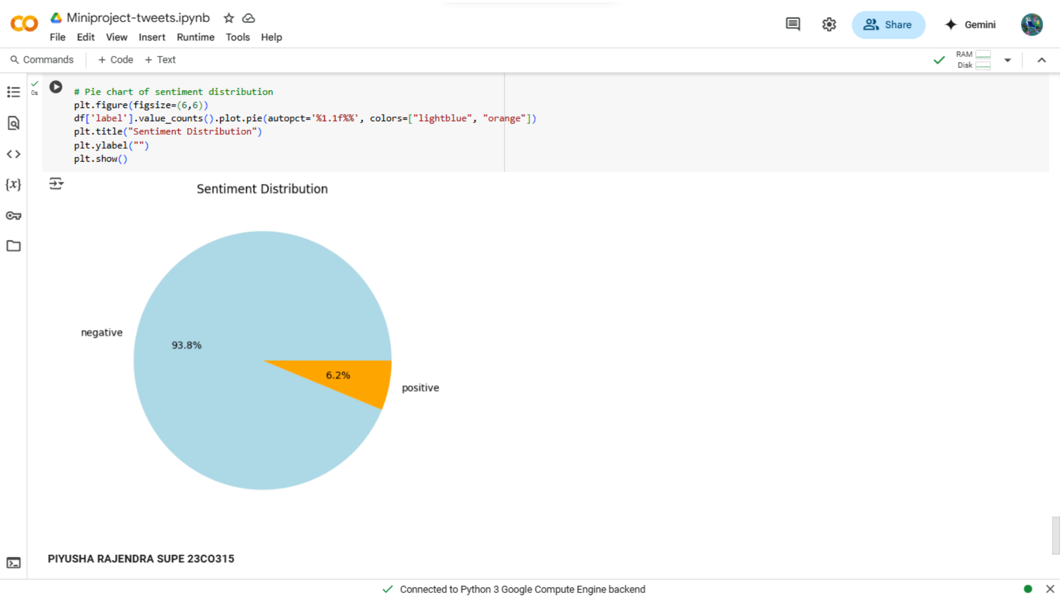

Miniproject-tweets.ipynb
File Edit View Insert Runtime Tools Help
Share
Gemini
Commands
+ Code
+ Text
RAM
Disk
Distribution of Tweet Lengths
Frequency
25000
20000
15000
10000
5000
0
0
50
100
150
200
250
300
Connected to Python 3 Google Compute Engine backend

```
# Plot the length distribution of tweets
df['tweet_length'] = df['cleaned_tweet'].apply(len)
plt.figure(figsize=(8,5))
sns.histplot(df['tweet_length'], bins=30, kde=True, color='purple')
plt.title("Distribution of Tweet Lengths")
plt.xlabel("Tweet Length")
plt.ylabel("Frequency")
plt.show()
```



```
words, counts = zip(*common_words)
plt.figure(figsize=(10,5))
sns.barplot(x=list(counts), y=list(words), palette="viridis")
plt.xlabel("Count")
plt.ylabel("Words")
plt.title("Most Common Words in Tweets")
plt.show()
```

```
<ipython-input-35-0b6feb907a09>:11: FutureWarning:

Passing `palette` without assigning `hue` is deprecated and will be removed in v0.14.0. Assign the `y` variable to `hue` and set `legend=False` for the same effect.

  sns.barplot(x=list(counts), y=list(words), palette="viridis")
```



```
# Pie chart of sentiment distribution
plt.figure(figsize=(6,6))
df['label'].value_counts().plot.pie(autopct='%1.1f%%', colors=["lightblue", "orange"])
plt.title("Sentiment Distribution")
plt.ylabel("")
plt.show()
```

PIYUSHA RAJENDRA SUPE 23CO315

FUNCTIONALITY AND ADVANTAGE

Functionality:

The sentiment analysis project focuses on analysing tweets related to data science, classifying them as positive or negative based on their content. The functionality includes the following steps:

1. **Data Preprocessing**: The raw tweet data is cleaned by removing irrelevant components like URLs, special characters, mentions, and hashtags. Text normalization is performed by converting all text to lowercase and tokenizing it for further processing.
2. **Feature Extraction**: **TF-IDF (Term Frequency-Inverse Document Frequency)** is applied to convert the text into a numerical representation, capturing important words that influence sentiment.
3. **Sentiment Classification**: The **Multinomial Naïve Bayes** algorithm is employed to classify tweets as either positive or negative based on their processed features.
4. **Evaluation**: The model's performance is evaluated using metrics such as accuracy, precision, recall, and F1-score. Visual tools like sentiment distribution plots, word clouds, and confusion matrices help interpret the model's results.

Advantages:

1. **Efficiency**: Automated sentiment classification allows for the quick processing of large datasets, reducing the time and effort needed for manual analysis.
2. **Real-time Analysis**: It can be applied to monitor public sentiment in real time, helping businesses, organizations, and researchers stay up to date with public opinions on data science or other topics.
3. **Actionable Insights**: The ability to classify tweets as positive or negative helps businesses understand customer opinions, enabling them to adjust marketing strategies, improve products, or gauge brand health.
4. **Scalability**: The method can be expanded to analyse a large volume of tweets or texts across different topics or industries, providing scalability for varied applications.
5. **Cost-effective**: Automated sentiment analysis can replace or supplement human analysis, saving resources and allowing for analysis at a much larger scale.
6. **Customization**: The framework is flexible, allowing for the use of more advanced NLP models or tweaks to improve performance, such as using word embedding’s like Word2Vec or Glove.

CONCLUSION

This sentiment analysis project provides a comprehensive approach to classifying tweets related to data science as positive or negative, leveraging machine learning techniques and natural language processing (NLP). By utilizing data pre-processing, TF-IDF vectorization, and the Multinomial Naïve Bayes algorithm, the project efficiently classifies large datasets of real-world tweets, offering valuable insights into public sentiment. The analysis highlights the effectiveness of automated sentiment classification in understanding public opinion, trends, and emotions expressed in social media. The evaluation metrics, including accuracy, precision, recall, and F1-score, confirm the model's reliability, while visualizations like sentiment distribution plots and word clouds enhance interpretability. Although the project uses a basic Naïve Bayes model, future enhancements such as incorporating more advanced NLP models (like LSTMs or Transformer-based architectures) or fine-tuning the current approach with word embedding's can further improve the accuracy and robustness of the classification process. Ultimately, this project demonstrates how machine learning and NLP can be effectively applied to real-world sentiment analysis tasks, helping organizations, businesses, and researchers gain actionable insights from social media data, improve decision-making, and stay informed about public sentiment trends.

REFERENCES

- https://www.kaggle.com/ruchi798/data-science-tweets
- https://scikit-learn.org/stable/modules/generated/sklearn.naive_bayes.MultinomialNB.html
- https://www.geeksforgeeks.org/applying-multinomial-naive-bayes-to-nlp-problems/
- https://www.upgrad.com/blog/multinomial-naive-bayes-explained/
- https://www.geeksforgeeks.org/what-is-sentiment-analysis/
- https://www.ibm.com/think/topics/sentiment-analysis
- Chirag Shah, “A Hands-On Introduction to Data Science”, Cambridge University Press, (2020), ISBN: ISBN 978-1-108-47244-9.
- Wes McKinney, “Python for Data Analysis”, O' Reilly media, ISBN: 978-1-449-31979-3.