Part A: IMDb Movie Review Sentiment Analysis

1. Introduction

Sentiment analysis is a key Natural Language Processing (NLP) technique that helps in understanding opinions, attitudes, and emotions from textual data. In this project, we analyze movie reviews from the IMDb dataset to classify them into *positive* or *negative* sentiments. This binary classification task applies several machine learning algorithms and compares their performance.

2. Dataset Overview

We use a balanced dataset with **50,000 movie reviews**, equally divided into **25,000 positive and 25,000 negative** samples. This balance helps avoid bias in classification.

3. Data Exploration

Using pandas and matplotlib, we explored:

- **Class Distribution:** Equal (positive: 25,000; negative: 25,000)
- **Length of Reviews:** Ranges from short snippets to long paragraphs.
- **Common Words:** Visualized using basic text plots (e.g., word clouds, bar charts).

4. Preprocessing

Raw reviews contain noise such as HTML tags, special characters, and stopwords. To clean the text, we used:

- **Lowercasing**
- **Removing HTML tags** (re.sub)
- **Tokenization** using nltk.word_tokenize
- **Stopword removal** using nltk.corpus.stopwords
- **Stemming** using PorterStemmer

This cleaned corpus is crucial for vectorization.

5. Feature Extraction with TF-IDF

The cleaned reviews were transformed using **TF-IDF Vectorizer**, which assigns importance to words based on their frequency in a document versus the whole corpus.

TfidfVectorizer(max_features=5000)

This produced a **5000-dimensional feature vector** for each review, ideal for input to ML models.

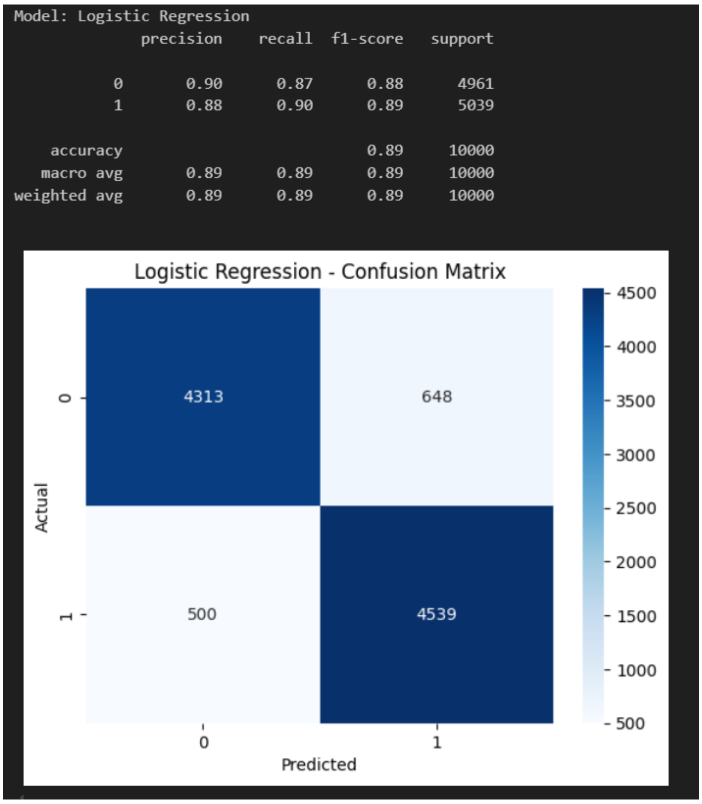
6. Model Building

We split the dataset: **80% training**, **20% testing**, using train_test_split.

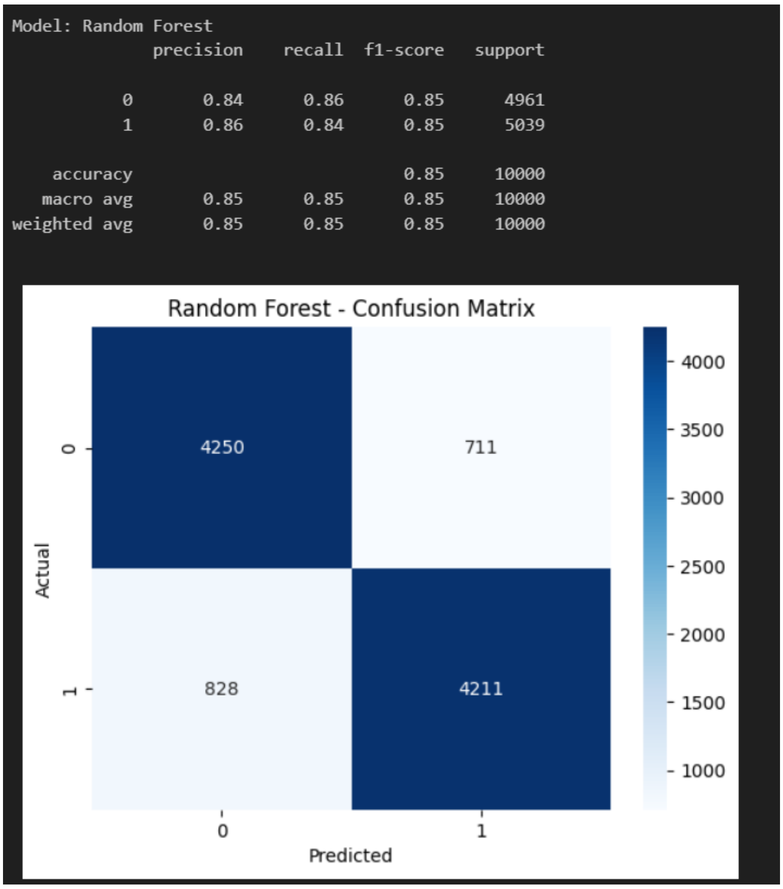
Three models were trained:

1. **Logistic Regression**
2. **Random Forest**
3. **Multinomial Naive Bayes**

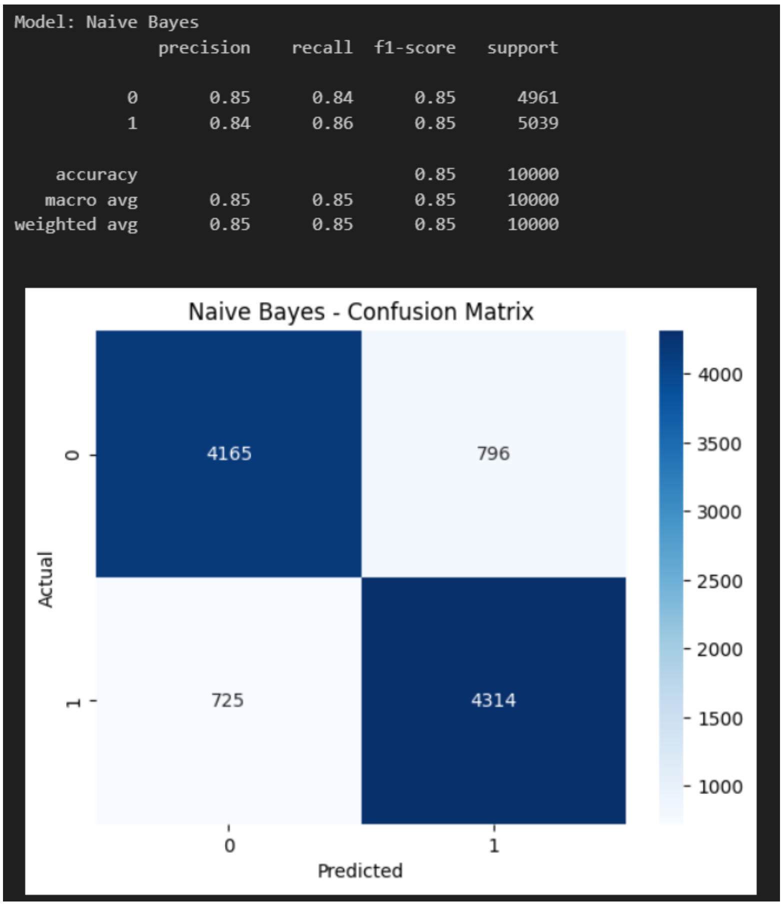
Each was trained using the TF-IDF vectors.



```
Model: Logistic Regression
              precision    recall  f1-score   support

           0       0.90      0.87      0.88      4961
           1       0.88      0.90      0.89      5039

    accuracy                           0.89     10000
   macro avg       0.89      0.89      0.89     10000
weighted avg       0.89      0.89      0.89     10000
```



```
Model: Random Forest
              precision    recall  f1-score   support

           0       0.84      0.86      0.85      4961
           1       0.86      0.84      0.85      5039

    accuracy                           0.85     10000
   macro avg       0.85      0.85      0.85     10000
weighted avg       0.85      0.85      0.85     10000
```



```
Model: Naive Bayes
              precision    recall  f1-score   support

           0       0.85      0.84      0.85      4961
           1       0.84      0.86      0.85      5039

    accuracy                           0.85     10000
   macro avg       0.85      0.85      0.85     10000
weighted avg       0.85      0.85      0.85     10000
```

7. Evaluation Metrics

Models were evaluated using:

- **Accuracy**
- **Precision, Recall, F1-score**
- **Confusion Matrix**

Logistic Regression:

- Accuracy: **~88.6%**
- Balanced Precision/Recall: Excellent at detecting both classes.

Random Forest:

Precision: 0.85 | Recall: 0.85 | Accuracy: 85%

Naive Bayes:

Accuracy: 85%

All models performed well, but **Logistic Regression** gave the best **cross-validation score** (mean CV accuracy ~88.6%).

---

8. Visualizations

- **Confusion Matrices**: Showed true positives/negatives.
- **Sentiment Distribution Bar Plot**
- **Sample prediction outputs** (for qualitative inspection).

---

9. Key Insights

- **Text preprocessing** has a significant impact.
- **TF-IDF** works well with linear models like Logistic Regression.
- Naive Bayes performs surprisingly well on sparse data.
- Ensemble models (e.g., Random Forest) offer robust performance but may overfit slightly.
- Logistic Regression is fast, interpretable, and highly accurate for text classification.

---

10. Conclusion

This sentiment analysis pipeline shows that traditional ML techniques, with solid preprocessing and vectorization, can yield strong results on NLP tasks. Logistic Regression, backed by TF-IDF, remains a top choice for binary text classification tasks.