**Muhammad Abdullah**

**Tech Horizon-Task 03**

**IMDB Movie Reviews Sentiment Analysis**

**1. Introduction**

In this project, I aimed to build a binary classification model that predicts whether an IMDB movie review expresses a positive or negative sentiment. Sentiment analysis is one of the most common applications of Natural Language Processing (NLP), and IMDB reviews provide a large real-world dataset for this task.

The dataset consists of 50,000 reviews, evenly split between positive and negative classes. The challenge lies in preprocessing the raw text data, transforming it into numerical form, and then applying a machine learning model to classify sentiments accurately.

**2. Objectives**

The main objectives of this project were:

* To clean and preprocess textual reviews.
* To convert text into numerical representations using TF-IDF Vectorization.
* To train a Multinomial Naive Bayes classifier for sentiment classification.
* To evaluate the model using metrics such as accuracy, precision, recall, F1-score, and confusion matrix.

**3. Methodology**

**Step 1: Import Libraries**

I imported essential Python libraries for:

Data handling: pandas, numpy

Visualization: matplotlib, seaborn

NLP preprocessing: nltk (stopwords removal)

Machine learning: sklearn modules for preprocessing, vectorization, model training, and evaluation

**Step 2: Load Dataset**

I used the IMDB\_Dataset.csv file (from Kaggle) which contains two columns:

* review → the text of the review
* sentiment → positive or negative label

I checked dataset shape, structure, and previewed the first few rows.

**Step 3: Exploratory Data Analysis (EDA)**

* Removed duplicate reviews to ensure clean data.
* Checked for missing values.
* Plotted the sentiment distribution using a count plot and a pie chart, which showed that the dataset was balanced (50% positive, 50% negative).

**Step 4: Data Preprocessing**

To prepare the reviews for modeling, I applied several text preprocessing steps:

1. Converted text to lowercase.
2. Removed HTML tags, punctuation, and special characters using regular expressions.
3. Removed stopwords (like "the", "is", "and") using NLTK.
4. Encoded the target column:
   * Positive → 1
   * Negative → 0

I applied these steps to every review in the dataset to create a clean text column.

**Step 5: Train-Test Split**

I divided the dataset into:

* 80% training data
* 20% testing data

This ensured that the model was trained on one part of the dataset and evaluated on unseen data.

**Step 6: Text Vectorization**

Since machine learning algorithms cannot directly handle text, I converted reviews into numerical features using TF-IDF Vectorizer with max\_features=5000.

* TF-IDF assigns weights to words based on how important they are in the dataset.
* Limiting to 5000 features helped balance performance and computational efficiency.

**Step 7: Model Training**

I trained a Multinomial Naive Bayes (MNB) classifier:

* MNB is widely used for text classification problems.
* It works well with high-dimensional, sparse datasets like TF-IDF vectors.

**Step 8: Model Evaluation**

I evaluated the model on the test set using:

* **Accuracy** → Overall correctness of the model
* **Precision, Recall, F1-score** → To measure performance on each class
* **Confusion Matrix** → To visualize misclassifications

Example of results:

* Accuracy: ~0.85 – 0.87
* F1-score for both classes was high, showing balanced performance.
* Confusion matrix showed most predictions were correct, with relatively fewer false positives/negatives.

**4. Results & Insights**

The Naive Bayes model performed well on IMDB reviews, achieving good accuracy and F1-score.

Preprocessing steps like removing stopwords and applying TF-IDF were crucial in improving model performance.

Sentiment distribution was balanced, which avoided bias in training.

**5. Conclusion**

This project successfully demonstrated how to apply NLP techniques and machine learning to perform sentiment analysis on IMDB movie reviews.

* The pipeline included data cleaning, preprocessing, vectorization, model training, and evaluation.
* The model achieved strong performance using a simple yet powerful Naive Bayes classifier.
* The trained model and vectorizer can be saved and deployed into real-world applications such as movie review analyzers or customer feedback sentiment tools.