📘 SENTIMENT ANALYSIS PROJECT REPORT

Sentiment Analysis on IMDB Movie Reviews Using Logistic Regression

---

1. Introduction

Sentiment Analysis is a Natural Language Processing (NLP) task that determines whether a given text expresses a positive or negative opinion.
In this project, we built a complete machine-learning workflow to classify IMDB movie reviews into **positive** or **negative** sentiments.

The project includes:

- Dataset loading
- Text preprocessing
- Feature extraction
- Model training
- Model evaluation
- Result interpretation
- Report documentation

This report summarizes the entire implementation and findings.

---

2. Dataset Description

The dataset used is the **IMDB Movie Reviews Dataset**, containing **50,000 labeled movie reviews**.

Dataset Details

- **Total samples:** 50,000
- **Classes:**
    - Positive (25,000)
    - Negative (25,000)
- **Task type:** Binary text classification
- **Source:** Kaggle
  https://www.kaggle.com/datasets/lakshmi25npathi/imdb-dataset-of-50k-movie-reviews

The dataset is balanced, making it ideal for evaluating classification models.

---

3. Data Preprocessing

Movie reviews contain raw text that cannot be directly used by machine learning models.
Therefore, several preprocessing steps were applied:

✓ **Preprocessing Steps**

1. Removal of HTML tags
2. Conversion of text to lowercase
3. Removal of URLs
4. Removal of punctuation
5. Removal of numbers
6. Removal of extra whitespace

After preprocessing, a cleaned version of each review (cleaned_review) was created.

📌 **Purpose:** Improves signal-to-noise ratio and model accuracy.

---

4. Feature Extraction Using TF-IDF

Text data must be converted into numerical form.
For this, **TF-IDF (Term Frequency–Inverse Document Frequency)** was used.

✓ **Why TF-IDF?**

- Highlights important words in each review
- Reduces weight of common words (e.g., “the”, “is”)
- Works extremely well with Logistic Regression

✓ **TF-IDF Parameters**

- **max_features = 20,000**
- **ngram_range = (1, 2)** → includes unigrams and bigrams
- **stop_words = 'english'**

TF-IDF transformed textual reviews into a high-dimensional feature matrix for model training.

---

5. Model Selection – Logistic Regression

We selected **Logistic Regression**, a widely used model for binary classification, especially with sparse text data.

✓ **Why Logistic Regression?**

- Performs strongly with TF-IDF features
- Fast and efficient
- Works well in high-dimensional spaces
- Easy to interpret

- Baseline model for text classification

Pipeline used:

TF-IDF Vectorizer → Logistic Regression

---

6. Training and Test Split

To evaluate the model properly, the dataset was split:

- **Training Set:** 80%
- **Testing Set:** 20%

This ensures we test the model on unseen data.

---

7. Model Evaluation

After training the model, multiple evaluation metrics were calculated.

✓ Accuracy

Measures how many predictions were correct.

✓ Precision

Percentage of predicted positive reviews that were actually positive.

✓ Recall

Percentage of actual positive reviews the model correctly identified.

✓ F1-Score

Harmonic mean of precision and recall.

✓ Confusion Matrix

Shows correct and incorrect classifications:

- True Positives (TP)
- True Negatives (TN)
- False Positives (FP)
- False Negatives (FN)

✓ ROC Curve & AUC

Measures how well the classifier separates the two classes.

---

8. Results Summary

Overall Metrics

- **Accuracy:** 0.8996
- **Precision (Positive):** 0.8904
- **Recall (Positive):** 0.9114
- **F1 Score (Positive):** 0.9008

These values indicate that the model has strong predictive ability, correctly identifying positive reviews more than **90%** of the time.

Class-wise Metrics

Negative Sentiment

- Precision: 0.9093
- Recall: 0.8878
- F1-Score: 0.8984
- Support: 5000

Positive Sentiment

- Precision: 0.8904
- Recall: 0.9114
- F1-Score: 0.9008
- Support: 5000

Averaged Metrics

- **Macro Avg F1 Score:** 0.8996
- **Weighted Avg F1 Score:** 0.8996

These averages show that the model maintains consistent performance across both classes.

---

9. Saving the Model

The trained model was saved using pickle:

imdb_sentiment_pipeline.pkl

This allows future use without retraining.

Additionally, predictions on the test set were saved as:

imdb_test_predictions.csv

---

10. Conclusion

This project successfully demonstrates how machine learning can be applied to sentiment analysis. By using **TF-IDF** for vectorization and **Logistic Regression** for classification, we achieved high performance on the IMDB dataset.

✓ Key Achievements

- Built a complete ML/NLP pipeline
- Achieved strong classification metrics
- Implemented preprocessing, feature engineering, training, and evaluation
- Visualized confusion matrix and ROC curve
- Exported trained model for future inference

The approach used is robust and can be extended to other review-based sentiment analysis tasks such as:

- Twitter sentiment
- Product reviews
- Customer feedback analysis

---

11. Submitted Files

- Jupyter Notebook
- README.md
- Saved model (imdb_sentiment_pipeline.pkl)
- Predictions file (imdb_test_predictions.csv)
- This report (PDF)

---

12. References

- Scikit-learn Documentation
- Kaggle IMDB Dataset
- Machine Learning textbooks and NLP resources