Sentiment Analysis on IMDb Reviews – Final Report

Introduction

Sentiment analysis is a vital natural language processing (NLP) task that determines whether a given text expresses a positive or negative sentiment. In this project, we analyze IMDb movie reviews to predict their sentiment using machine learning and deep learning models. The results of this analysis can help movie producers, critics, and platforms like IMDb understand public opinion and optimize content strategies.

---

1. Problem Statement

The objective of this project is to develop a classification model that accurately predicts whether a movie review expresses a **positive** or **negative** sentiment. To achieve this, we employ **text preprocessing techniques, feature extraction methods, and multiple classification algorithms**. Model performance is evaluated using metrics such as **accuracy, precision, recall, and F1-score**.

---

2. Dataset Overview

We use the **IMDb dataset**, which contains a collection of movie reviews, each labeled as either **positive** or **negative**. The dataset consists of:

- **Review Text**: The actual review provided by the user.
- **Sentiment Label**: The sentiment of the review (either *positive* or *negative*).

Before model development, we perform **data exploration, cleaning, feature engineering, and vectorization** to prepare the data for training.

---

3. Data Exploration and Analysis

To understand the dataset, we conduct an **initial analysis** to identify trends, missing values, and class distribution.

3.1 Checking for Missing Values and Data Imbalance

- We examine whether there are any missing reviews or labels.
- We check if the dataset is balanced between positive and negative reviews. If it's imbalanced, we may apply **oversampling or undersampling techniques**.

3.2 Analyzing Review Lengths

- We compute the **word count** and **character count** for each review.
- We visualize the distribution using **histograms and box plots**.
- Outliers (very short or excessively long reviews) are identified.

---

4. Data Cleaning and Preprocessing

Text preprocessing is essential to improve model performance. The following steps were performed:

4.1 Removing Stop Words, Punctuation, and Special Characters

- Common stop words (e.g., *the, is, and, in*) are removed.
- Punctuation and special characters are eliminated to reduce noise.

4.2 Tokenization

- Each review is split into individual words (**tokens**) for further processing.

4.3 Lemmatization and Stemming

- **Lemmatization** converts words to their base form (e.g., *running* → *run*).
- **Stemming** reduces words to their root form (e.g., *playing* → *play*).

---

5. Feature Engineering

We convert textual data into numerical representations to be used by machine learning models.

5.1 Textual Features

- **Word Count**: Total number of words in a review.
- **Character Count**: Total number of characters.

- **Average Word Length**: Average length of words in the review.

5.2 Vectorization Methods

- **Bag-of-Words (BoW)**: Represents text as word frequency vectors.
- **TF-IDF (Term Frequency-Inverse Document Frequency)**: Assigns importance scores to words based on their frequency in a document versus the entire dataset.
- **Word2Vec & Embeddings**: Captures word relationships by mapping words into dense vectors of fixed size.

---

6. Model Development

We experiment with various machine learning and deep learning models.

6.1 Machine Learning Models

We implement and evaluate:

- **Logistic Regression**: A simple yet effective baseline model.
- **Naïve Bayes**: Suitable for text classification due to its probabilistic nature.
- **Support Vector Machine (SVM)**: Captures complex relationships in high-dimensional space.
- **Random Forest**: An ensemble learning method that combines multiple decision trees.

Each model is tuned using **hyperparameter optimization** and evaluated using classification metrics.

6.2 Deep Learning Models

We extend our analysis to deep learning models for improved performance:

- **LSTM (Long Short-Term Memory Networks)**: A type of recurrent neural network (RNN) that captures sequential dependencies in text.
- **BERT (Bidirectional Encoder Representations from Transformers)**: A transformer-based model pre-trained on vast text corpora, providing state-of-the-art NLP capabilities.

7. Model Development and Evaluation

We experimented with multiple models to identify the best-performing classifier for sentiment prediction.

Logistic Regression Results

- Accuracy: 88.73%
- Precision: 0.90 (class 0), 0.88 (class 1)
- Recall: 0.88 (class 0), 0.90 (class 1)
- F1-score: 0.89 for both classes

Naive Bayes Results

- Accuracy: 85.31%
- Precision: 0.86 (class 0), 0.85 (class 1)
- Recall: 0.84 (class 0), 0.86 (class 1)
- F1-score: 0.85 for both classes

8. Key Insights

- Logistic Regression outperformed Naive Bayes in terms of accuracy, precision, recall, and F1-score.
- The accuracy difference between the two models is 3.42%, favoring Logistic Regression.
- Both models showed balanced performance across classes.

9. Recommendations and Next Steps

- Given its superior performance, Logistic Regression is the preferred model for this dataset.
- Further improvements can be explored through hyperparameter tuning and feature engineering.
- Experimenting with advanced classifiers like SVM, Random Forest, LSTM, or BERT can enhance performance.

- Additional sentiment-related features like POS tagging, n-grams, and sentiment lexicons can be integrated.
- Model interpretability tools like SHAP can be used to understand feature importance.

10. Model Evaluation and Results

Each model is evaluated using:

- **Accuracy**: Percentage of correct predictions.
- **Precision**: Proportion of positive predictions that are actually positive.
- **Recall**: Ability to identify all positive cases.
- **F1-Score**: Harmonic mean of precision and recall.

10.1 Performance Comparison

- Traditional machine learning models like **Logistic Regression and SVM** perform well with TF-IDF.
- **LSTM and BERT outperform traditional models** due to their ability to capture deeper semantic meaning.
- **Hyperparameter tuning** improves model accuracy.

---

11. Conclusion and Insights

This project successfully develops a sentiment analysis model for IMDb reviews. Key takeaways:

- **TF-IDF and Word2Vec** provide robust feature representations.
- **Logistic Regression and SVM** serve as strong baselines.
- **Deep learning models (LSTM & BERT) outperform traditional approaches**, with BERT achieving the highest accuracy.
- The model can be deployed for real-world applications like **automated sentiment monitoring for movie reviews**.

Future Improvements

- Experimenting with **more transformer-based models** (e.g., GPT, RoBERTa).
- Fine-tuning **pretrained language models** for domain-specific sentiment analysis.
- Implementing **real-time review analysis** for streaming platforms.

Appendix

- Code snippets, hyperparameter tuning details, and dataset samples can be found in the project repository.
