Sentiment Analysis of Movie Reviews: Developing an Accurate and Efficient Machine Learning Model

By

Koushik Sai Veerella

Contents

1. Introduction

1.1. Problem Statement

Movies have become an integral part of people's lives and a significant source of entertainment worldwide. With the increasing number of movies being produced every year, it is crucial for movie makers and critics to understand how their movies are received by the audience. One popular technique used to analyse the reception of movies is sentiment analysis of movie reviews. This technique involves the use of natural language processing (NLP) and machine learning algorithms to determine the emotional tone and sentiment behind text data.

Sentiment analysis of movie reviews can provide valuable insights into how the audience perceives a particular movie. By analysing the sentiment expressed in reviews, movie makers and critics can understand what aspects of the movie are being appreciated or criticized. This can help them make informed decisions about future movies, such as the type of storyline, actors, or direction, that may be better received by the audience.

1.2. Goal

The goal of this project is to create a machine learning model that accurately and efficiently performs sentiment analysis of movie reviews. To achieve this, the IMDB movie review dataset will be used, which contains 50,000 movie reviews, equally divided into positive and negative reviews.

Various algorithms such as Naive Bayes, Logistic Regression, Random Forest, XGBoost, Support Vector Machines (SVM), and LSTM will be experimented with to determine the most effective approach for this task. The performance of each algorithm will be evaluated with the help of measuring metrics. Once the model is trained and optimized, it will be used to predict the sentiment of new movie reviews. The input to the model will be the raw text of the movie review, and the output will be a label indicating whether the sentiment of the review is positive or negative. The success of this project will be measured by the accuracy and efficiency of the developed model in predicting the sentiment of movie reviews.

Accurate sentiment analysis can provide valuable insights into the reception of movies, enabling movie makers and critics to make informed decisions about future movies. Additionally, an efficient model can be scaled up to handle large volumes of movie reviews, making it a valuable tool for the movie industry.

1.3. Approach

We begin by pre-processing the raw data obtained from the IMDB movie review dataset. This involves cleaning and transforming the text data by removing irrelevant information such as stop words and punctuations and converting the text to lowercase to enable effective modelling. We then proceed to train various Machine Learning models such as Naive Bayes, Logistic Regression, Random Forest, XGBoost, Support Vector Machines (SVM), and LSTM to classify the reviews into two categories: positive or negative. Each model is evaluated based on its accuracy, precision, recall, and F1-score to determine the most effective approach for sentiment analysis of movie reviews.

Finally, the best performing model will be deployed as a web application for real-time use. The deployed model can be used to analyse and predict the sentiment of new movie reviews, providing valuable insights for movie makers and critics to understand the audience's reception of movies. The deployed web application is designed to be user-friendly, enabling users to enter a movie review and receive an immediate sentiment analysis prediction. The accuracy of the deployed model is constantly monitored to ensure that it remains efficient and reliable.

2. Methodology

2.1. Data

The Large Movie Review Dataset, also known as the IMDB dataset, is the primary dataset used in this study. The dataset is obtained from Kaggle. This dataset comprises 50,000 movie reviews that are labelled as either positive or negative. The reviews are written by a diverse range of reviewers, and they cover a wide variety of movie genres. The dataset provides a rich source of information for training and evaluating Machine Learning models for sentiment analysis of movie reviews.

2.2. Exploratory Data Analysis

Exploratory data analysis (EDA) is a crucial step in any data science project, including sentiment analysis of movie reviews. In this project, EDA was conducted to gain a better understanding of the IMDB movie review dataset and its characteristics.

To begin with, it was important to analyse the dataset in terms of the distribution of positive and negative reviews. This analysis will reveal the number of instances for each

sentiment, providing a better understanding of the data before moving forward with further analysis and model development.

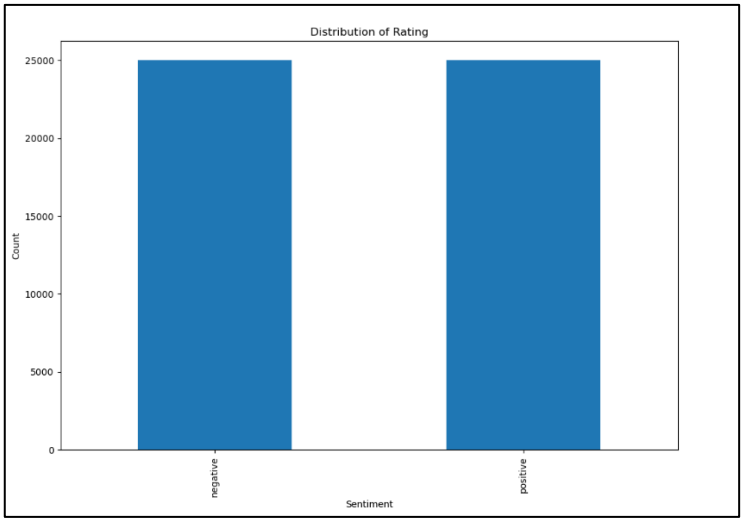


Figure 2.2.1 Distribution of each sentiment

Based on the plot, it is evident that the dataset is balanced, with an equal number of positive and negative reviews. This balance ensures that the model will not be biased towards any particular sentiment and can be expected to perform well in predicting both positive and negative reviews. With this understanding, we can proceed to the next steps in the analysis.

One of the objectives of the EDA was to identify the most common words used in the reviews. This helps to gain insights into the language used by the audience when reviewing movies. The most frequent words identified in the reviews can be used to develop a better understanding of the audience's preferences and their emotional tone towards specific movies.

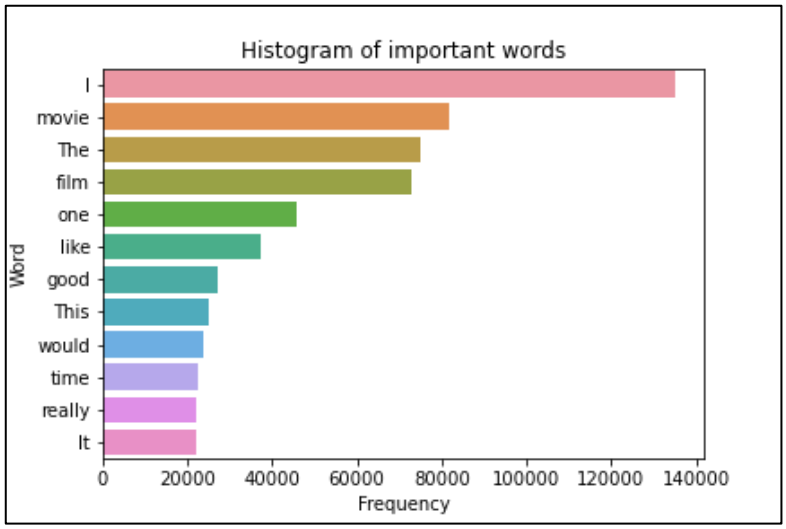


Figure 2.2.2. Most common words in the reviews

Another objective of the EDA was to determine the average length of words in the reviews. This helps to identify the complexity of the language used in the reviews and assists

in selecting the most effective text pre-processing techniques. The EDA results also showed that the dataset contained a diverse range of vocabulary. This is important as it indicates that the dataset is representative of the language used by the audience in their movie reviews. The diverse range of vocabulary also helps to develop more accurate and effective Machine Learning models as it reduces the bias towards specific words or phrases.

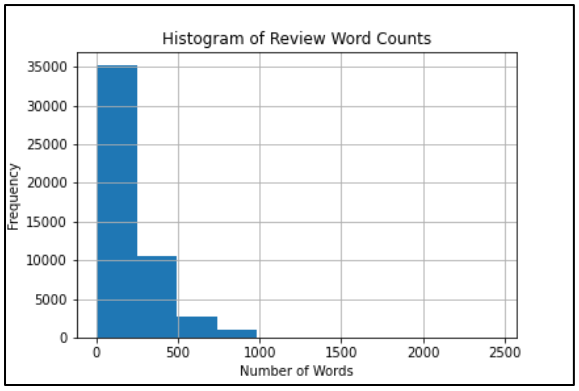


Figure 2.2.3. Average word lengths in reviews

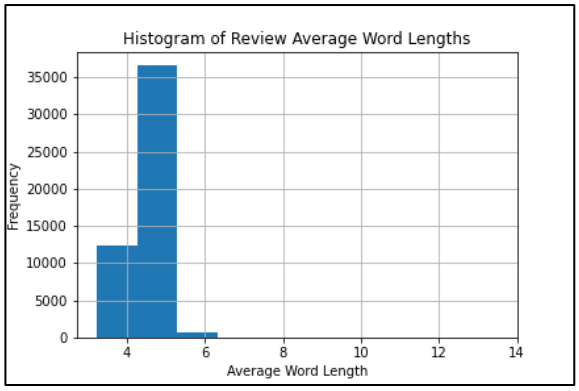


Figure 2.2.4. Word counts in the reviews

Next, an analysis of the frequency of stopwords in the reviews was conducted. This analysis aided in identifying the words that do not contribute much to the sentiment of the reviews, which enabled their removal from the dataset to improve the accuracy of the model.

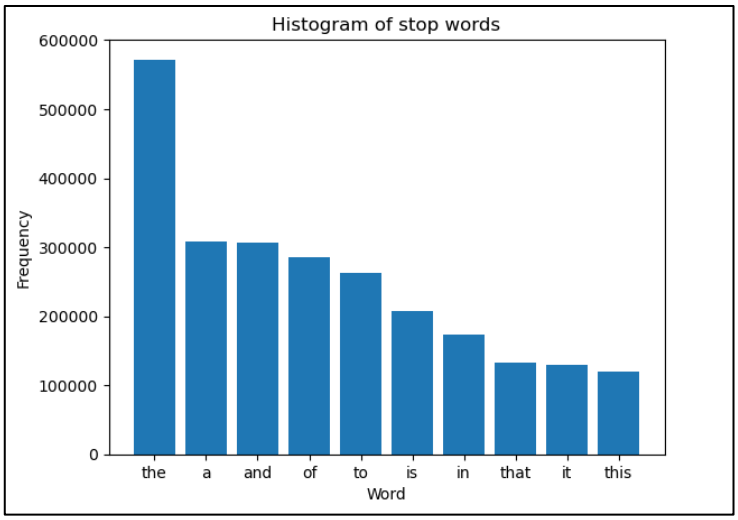


Figure 1.2.5. Frequency of Stop Words

Overall, the EDA conducted in this project provided valuable insights into the IMDB movie review dataset, helping to develop a better understanding of the language used by the audience when reviewing movies. The results of the EDA were used to inform the text pre-processing techniques and develop more accurate and effective Machine Learning models for sentiment analysis of movie reviews.

2.3. Pre-Processing

The IMDB dataset used in this study contains two essential columns: a review column that holds the text of the movie review and a sentiment column that indicates whether the review is positive or negative. However, to use this data for training a Machine Learning model, the categorical sentiment values must be converted into numerical values.

To achieve this, a new column is added to the dataset, which is assigned a value of 1 for positive sentiment and 0 for negative sentiment. This approach allows us to transform the categorical sentiment column into a binary classification target, which can be used to train a model to predict the sentiment of a given movie review. The process of converting categorical data into numerical data is a crucial step in preparing data for Machine Learning algorithms. The numerical data is easier to process and analyse, and it can be used as input for various Machine Learning algorithms.

Before training the models, the raw text data from the IMDB movie review dataset is pre-processed to ensure that the Machine Learning models can effectively learn from the data. The pre-processing of the text data involves several steps.

Firstly, all HTML tags are removed using the BeautifulSoup library. This step is necessary as the raw text data from the dataset may contain HTML tags, which are not relevant to the sentiment analysis task. Next, non-alphabetic characters and stop words are removed from the text data. This helps to reduce the complexity of the text data and remove irrelevant information. Stop words are commonly used words that do not carry significant meaning, such as "the" and "and". After removing non-alphabetic characters and stop words, the remaining words are stemmed using the Porter stemmer. This process involves reducing words to their root form, which helps to simplify the text data and reduce the dimensionality of the dataset.

Finally, the pre-processed data is vectorized using the Count Vectorizer and TF-IDF method. This process involves converting the pre-processed text data into a numerical representation that can be used by the Machine Learning models. The Count Vectorizer creates a matrix of word counts for each document, while the TF-IDF method assigns weights to each word based on its frequency and importance in the document.

2.4. Feature Extraction

In this project, the Bag-of-Words (BoW) approach is used to extract features from the movie reviews in the training set. The BoW approach involves creating a vocabulary of all the unique words in the training set, and then representing each review as a vector where each element is the count of the corresponding word in the vocabulary.

The first step in the BoW approach is to create a vocabulary of unique words in the training set. This is done by tokenizing the text data, which involves splitting the reviews into individual words. The resulting set of words is then used as the vocabulary. Next, each review in the training set is represented as a vector of the same length as the vocabulary. Each element in the vector represents the count of the corresponding word in the review. For example, if the

word "good" appears three times in a review, the element corresponding to "good" in the review vector would have a value of 3.

The BoW approach is a simple and effective way to represent text data as numerical features that can be used by Machine Learning models. It has been widely used in Natural Language Processing tasks such as sentiment analysis, text classification, and topic modelling. By using the BoW approach in this project, features can be effectively extracted from the movie reviews and utilized to train and evaluate the Machine Learning models.

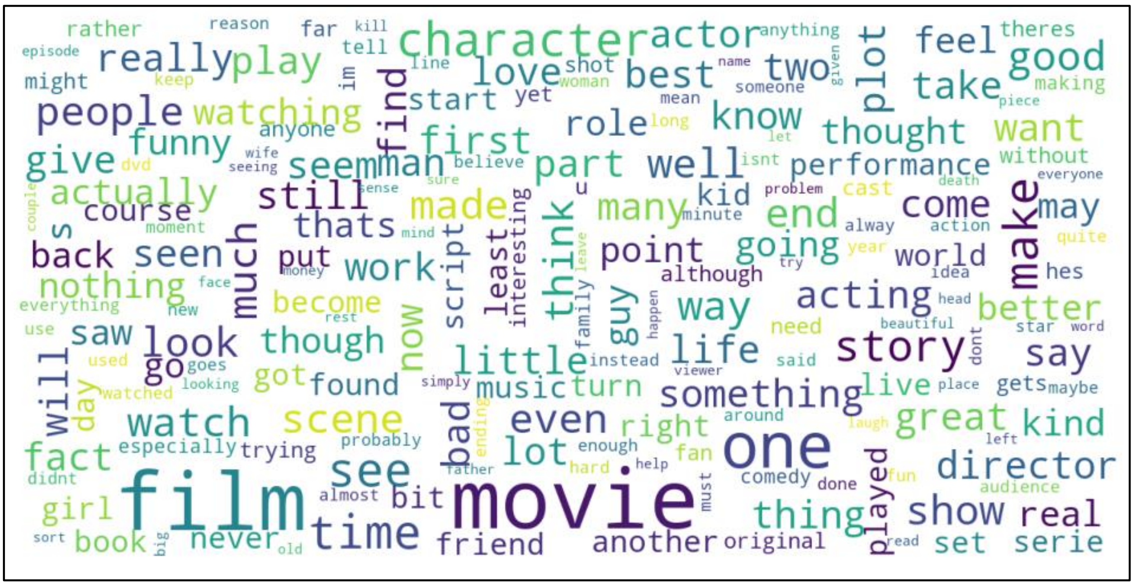


Figure 2.4.1. Bag of Words

2.5. Algorithms

In this project, a variety of Machine Learning models have been experimented and implemented to classify movie reviews as either positive or negative. The models that have been explored include Multinomial Naïve Bayes, Random Forest, Logistic Regression, LSTM, XGBoost, Support Vector Machine (SVM). Each of these models utilizes different algorithms and techniques to classify the reviews. Through exploration and implementation of multiple Machine Learning models, their performance can be evaluated, and the most accurate and efficient model for sentiment analysis of movie reviews can be identified.

2.5.1. Multinomial Naïve Bayes:

Multinomial Naive Bayes (MNB) is a popular algorithm for text classification, assuming that the features (words) are independent, given the class label. MNB is well-suited to text classification, where the input data is represented as a bag of words. The algorithm models the probability distribution of each class label, given the frequency of each word in the document. MNB is called "Multinomial" because it assumes that the word frequencies in the

documents are generated by a multinomial distribution. During the prediction phase, the algorithm computes the posterior probability of each class label, given the input document, using Bayes' rule. MNB is simple, scalable, and works well with small training datasets, making it an effective machine learning algorithm for text classification tasks.

2.5.2. Random Forest:

Random Forest is machine learning algorithm that operates by constructing a large number of decision trees and averaging their predictions. Each decision tree is constructed by selecting a random subset of features and training it on a subset of the data. The algorithm then combines the predictions of all the trees to produce the final result. Random Forest is robust to overfitting and works well with high-dimensional data, making it an effective algorithm for both classification and regression tasks. In scikit-learn, the RandomForestClassifier and RandomForestRegressor classes provide implementations of Random Forest for classification and regression tasks, respectively.

2.5.3. Logistic Regression:

Logistic Regression is the most common machine learning algorithm for classification tasks that models the probability of each class label given the input features using a logistic function. It is trained by optimizing a loss function, typically the cross-entropy loss, using gradient descent. Logistic Regression is well-suited to binary classification problems but can also be used for multi-class problems. In scikit-learn, the LogisticRegression class provides an implementation of the algorithm for both binary and multi-class classification tasks. Logistic Regression is interpretable, scalable, and handles large datasets, making it a powerful tool for classification problems.

2.5.4. LSTM (Long Short-Term Memory):

Long Short-Term Memory (LSTM) is a type of Recurrent Neural Network (RNN) architecture that is widely used for processing sequential data such as natural language text, speech, and time-series data. LSTMs are capable of capturing long-term dependencies in the input data and can maintain memory over extended periods. This makes them well-suited to tasks such as language modeling, machine translation, and speech recognition. In Keras, a popular deep learning library, the LSTM class provides an implementation of the LSTM architecture that can be easily integrated into larger neural network models.

2.5.5. XGBoost (Extreme Gradient Boosting):

XGBoost (Extreme Gradient Boosting) is a widespread open-source gradient boosting framework that is widely used for supervised learning tasks, including classification, regression, and ranking. It is a fast, scalable, and high-performance machine learning library that is designed to optimize accuracy while also being computationally efficient. In the context of sentiment analysis of movie reviews, XGBoost can be used as a powerful Machine Learning algorithm to build a predictive model. It works by iteratively improving the accuracy of the model by building new models that focus on the instances that were not well predicted by the previous models.

2.5.6. Support Vector Machines (SVM):

Support Vector Machines (SVM) is another popular algorithm used for classification tasks, including sentiment analysis. SVM aims to find the optimal hyperplane that separates the positive and negative classes in the feature space. The SVM algorithm can be utilized in this project to classify the movie reviews as positive or negative. The reviews can be represented as a vector of features using the Bag-of-Words approach, and then the SVM model can be trained on these feature vectors to classify the reviews.

Linear SVM is a variant of the SVM algorithm that uses a linear kernel to map the input feature vectors to a higher-dimensional space. Linear SVM is well-suited for text classification tasks, such as sentiment analysis, where the number of features is very high compared to the number of training samples.

Linear SVMs are computationally efficient and have shown to perform well on text classification tasks. Therefore, Linear SVM will be used for this project of instead SVM.

3. Modelling Process

3.1 Model Building

To ensure the effectiveness of the models, the test-train split method from the scikit-learn library was employed. The data was divided into 75% for training and 25% for testing, which allowed for the training of models using a substantial portion of the data while still being able to evaluate their performance accurately. This approach enabled us to validate the model's ability to generalize to new data by testing it on the remaining 25% of data that was not utilized for training.

Multinomial Naïve Bayes, Logistic Regression, Random Forest Classifier, and linear SVM models were incorporated into our analysis, which were imported from the scikit-learn library. Furthermore, XGBoost from the XGBoost library and LSTM using the Keras library were utilized. By employing a diverse range of models, the best-performing algorithm for sentiment analysis was identified.

Before performing hyperparameter tuning, baseline models for all the classifiers, including Multinomial Naïve Bayes, Logistic Regression, Random Forest Classifier, linear SVM, XGBoost, and LSTM were implemented. Then, the best-performing baseline model was identified, and hyperparameter tuning was conducted using the GridSearchCV function from scikit-learn. This enabled fine-tuning of the hyperparameters and optimization of the model's performance.

3.2 Model Evaluation

To evaluate the performance of the machine learning models for sentiment analysis, several widely used performance metrics were used. These metrics provided us with a quantitative way of measuring how well the models are performing on the task at hand.

The metrics which were used include accuracy, precision, recall, F1-score, and confusion matrix. Accuracy is a measure of how many correct predictions the model made out of all the predictions it made. Precision measures how many of the predicted positive reviews were actually positive, while recall measures how many of the actual positive reviews were correctly identified by the model. F1-score is a combined measure of precision and recall that provides a balanced view of the model's performance. The confusion matrix is a tool used to evaluate the performance of a classification model, which helps to determine the number of true positive (TP), false positive (FP), true negative (TN), and false negative (FN) predictions made by the model.

Let’s look at each models’ performance using these performance metrics.

3.2.1 Multinomial Naïve Bayes Performance

The Multinomial Naive Bayes model achieved an accuracy of 84.76% on the test set, which indicates that it performed reasonably well in classifying the sentiment of movie reviews. In terms of precision and recall, the model achieved scores of 85% for both, which

suggests that it was able to correctly identify positive and negative reviews with similar levels of accuracy.

```
Accuracy on validation set: 0.8476

AUC score : 0.8481

Classification report :
              precision    recall  f1-score   support

           0       0.82      0.88      0.85      6157
           1       0.87      0.82      0.84      6343

    accuracy                           0.85     12500
   macro avg       0.85      0.85      0.85     12500
weighted avg       0.85      0.85      0.85     12500
```

Figure 2.1.1. Performance Metrics of Multinomial Naïve Bayes

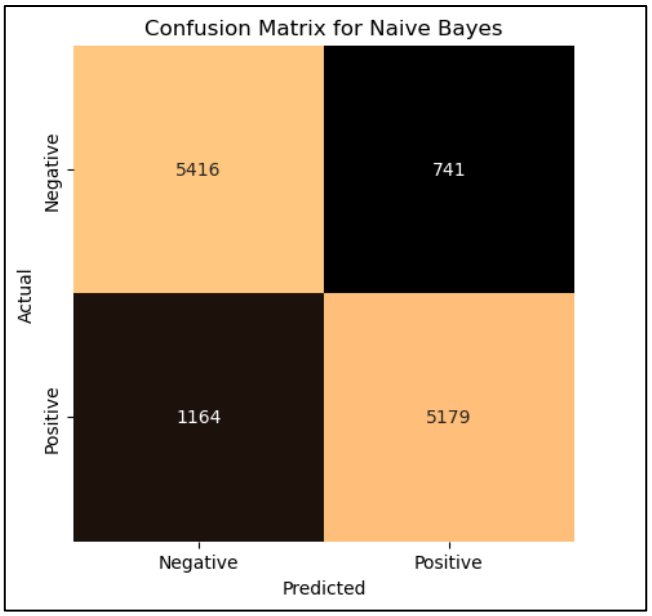


Figure 3.1.2. Confusion Matrix of Naïve Bayes

However, it is important to note that the model incorrectly classified 1,164 positive reviews as negative and 741 negative reviews as positive, as shown by the confusion matrix. Overall, while the Multinomial Naive Bayes model did not perform as well as expected but, it still provides a useful baseline for sentiment analysis of movie reviews.

3.2.2 Random Forest Performance

The Random Forest model showed an accuracy of 84.95% on the test dataset, which suggests that it is a relatively accurate model for predicting the sentiment of movie reviews. In terms of precision and recall, the model had a value of 85% for both, which means that the model is able to correctly identify positive and negative reviews with a good degree of accuracy.

```
Accuracy on validation set: 0.8495

AUC score : 0.8497

Classification report :
              precision    recall  f1-score   support

           0       0.84      0.86      0.85      6157
           1       0.86      0.84      0.85      6343

    accuracy                           0.85     12500
   macro avg       0.85      0.85      0.85     12500
weighted avg       0.85      0.85      0.85     12500
```

Figure 3.2.1. Performance Metrics of Random Forest Model

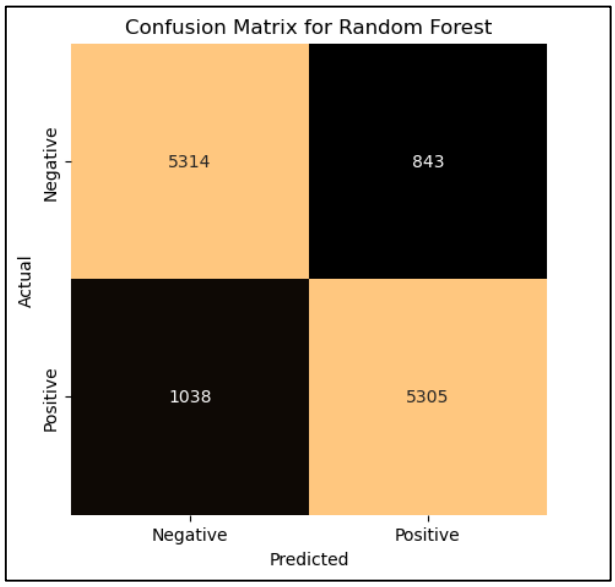


Figure 3.2.2 Confusion Matrix of Random Forest

The confusion matrix indicates that it misclassified 1,038 positive reviews as negative and 843 negative reviews as positive.

3.2.3 Logistic Regression Performance

The Logistic Regression model achieved a higher accuracy of 89.82% on the test data compared to the Random Forest model. It also had a higher precision and recall value of 90%. The precision value indicates the proportion of true positives among all positive predictions, while the recall value represents the proportion of true positives among all actual positives. Thus, a higher precision and recall value suggests that the model is better at accurately classifying the reviews.

```
Accuracy on validation set: 0.8982

AUC score : 0.8980

Classification report :
              precision    recall  f1-score   support

           0       0.90      0.89      0.90      6157
           1       0.89      0.91      0.90      6343

    accuracy                           0.90     12500
   macro avg       0.90      0.90      0.90     12500
weighted avg       0.90      0.90      0.90     12500
```

Figure 3.3.1. Performance Metrics of Logistic Regression Model

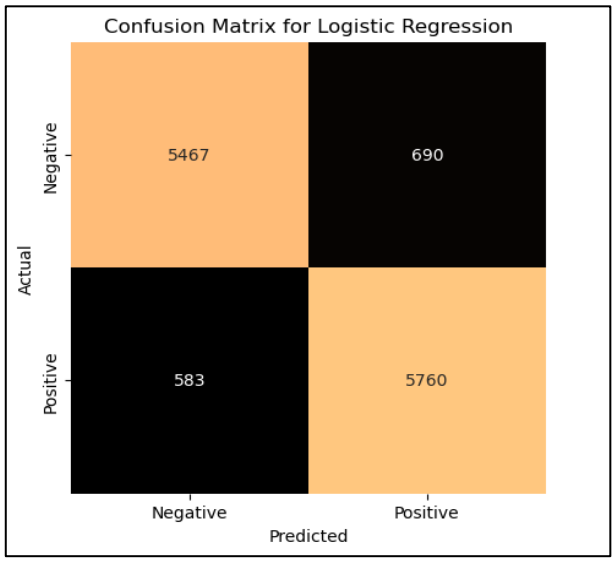


Figure 3.3.2. Confusion Matrix of Logistic Regression

The confusion matrix shows that out of the total 25,000 reviews in the test set, the model correctly classified 22,727 reviews (90.91%). However, there were 1,263 reviews (5.05%) that were falsely predicted as positive when they were actually negative, and 1,010 reviews (4.04%) that were falsely predicted as negative when they were actually positive. Specifically, the model incorrectly classified 583 positive reviews as negative and 690 negative reviews as positive.

This means that although the model is performing well with an accuracy of 89.82%, there is still room for improvement to reduce the number of false positives and false negatives.

3.2.4 XGBoost Performance

XGBoost performed even better with an accuracy of 90%, but despite hyperparameter tuning, its performance couldn't be improved any further.

```
Accuracy on validation set: 0.9000

AUC score : 0.8999

Classification report :
              precision    recall  f1-score   support

           0       0.90      0.89      0.90      6157
           1       0.90      0.91      0.90      6343

    accuracy                           0.90     12500
   macro avg       0.90      0.90      0.90     12500
weighted avg       0.90      0.90      0.90     12500
```

Figure 3.4.1. Performance Metrics of XGBoost

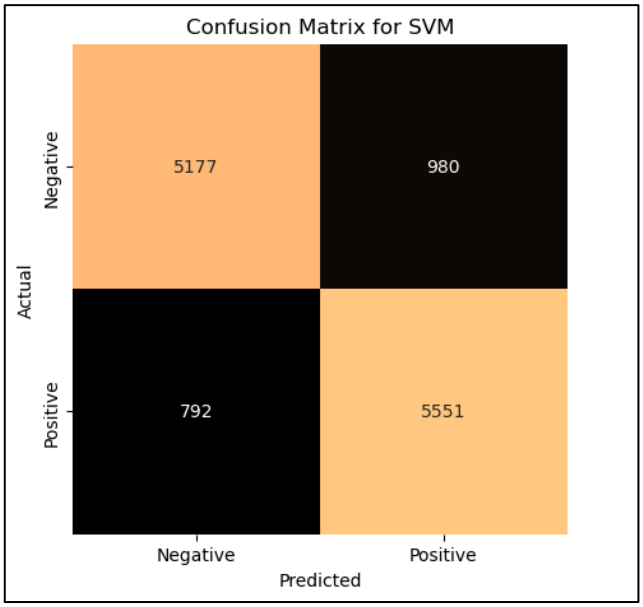


Figure 3.4.2. Confusion Matrix of baseline SVM

The confusion matrix of the XGBoost Model shows that, the model incorrectly classified 792 positive reviews as negative and 980 negative reviews as positive.

3.2.5 LSTM Performance

Despite increasing the number of epochs to improve the accuracy of the LSTM deep learning model, it still achieved a slightly lower accuracy of 85.34% compared to the other models. Although the training accuracy improved with more epochs, the testing accuracy remained the same. This suggests that the LSTM model may not be the best fit for sentiment analysis of movie reviews in this dataset.

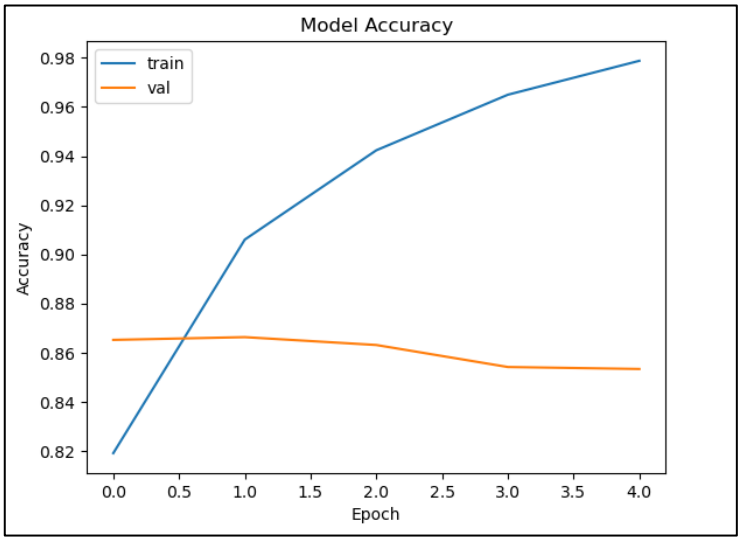


Figure 3.5.1. Training and testing accuracy graph of LSTM

3.2.6 SVM Performance (baseline model)

The SVM baseline model performed reasonably well with an accuracy of 90%, which is similar to the XGBoost model. The confusion matrix showed that it correctly predicted 5,505 positive reviews and 5,745 negative reviews, while misclassifying 598 positive reviews and 652 negative reviews.

```
Accuracy on validation set: 0.9000

AUC score : 0.8999

Classification report :
              precision    recall  f1-score   support

           0       0.90      0.89      0.90      6157
           1       0.90      0.91      0.90      6343

    accuracy                           0.90     12500
   macro avg       0.90      0.90      0.90     12500
weighted avg       0.90      0.90      0.90     12500


Confusion Matrix :
 [[5505  652]
 [ 598 5745]]
```

Figure 3.6.1. Performance Metrics of baseline SVM

3.2.7 SVM Performance (after Hyperparameter tuning)

Once it was discovered that the SVM baseline model had the highest accuracy, hyperparameter tuning using GridSearchCV was conducted to further enhance the model's performance. The hyperparameter tuning resulted in an increase in accuracy from 90% to 90.25%. The hyperparameters which resulted in the increased accuracy are –

{'C': 0.5, 'loss': 'squared_hinge', 'max_iter': 500, 'penalty': 'l2'}

Here are its performance metrics.

```
Accuracy on validation set: 0.9025

AUC score : 0.9024

Classification report :
              precision    recall  f1-score   support

           0       0.91      0.89      0.90      6157
           1       0.90      0.91      0.90      6343

    accuracy                           0.90     12500
   macro avg       0.90      0.90      0.90     12500
weighted avg       0.90      0.90      0.90     12500
```

Figure 3.7.1. Performance Metrics of tuned SVM

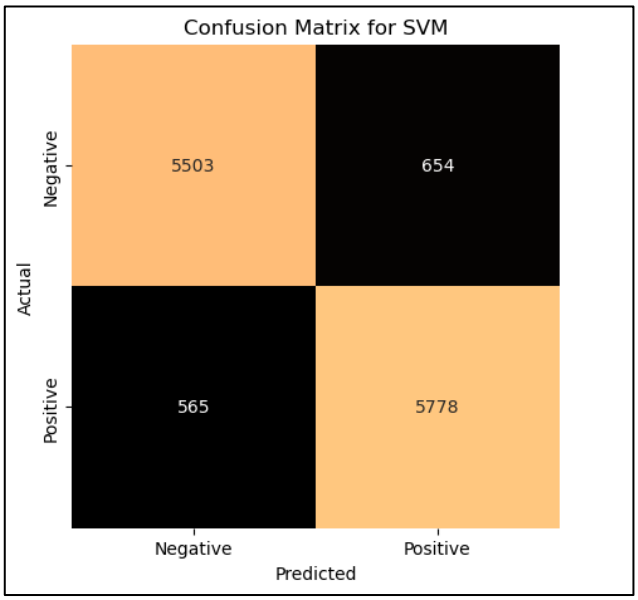


Figure 3.7.2. Confusion Matrix of tuned Linear SVM

The model accurately classified 5778 positive reviews and 5503 negative reviews, while misclassifying 565 positive reviews, and 654 negative reviews.

4. Results

Performance metrics were computed for each of the machine learning models that were implemented, such as Multinomial Naive Bayes, Logistic Regression, Random Forest, XGBoost, Linear SVM, and LSTM. Overall, it was found that all of the models performed reasonably well, with accuracies ranging from 84% to 90%. However, the Linear SVM model stood out as the top performer, with an accuracy of 90.25% and a high F1-score. This suggests that the Linear SVM model is the most accurate and reliable model for sentiment analysis of movie reviews in this dataset.

Model	Precision	Recall	F-1 Score	AUC Score	Accuracy
Multinomial Naïve Bayes	0.85	0.85	0.85	0.8481	0.8476
Random Forest	0.85	0.85	0.85	0.8497	0.8497
Logistic Regression	0.90	0.90	0.90	0.8980	0.8982
LSTM					0.8534
XGBoost	0.90	0.90	0.90	0.8899	0.8900
Linear SVM (Without tuning)	0.90	0.90	0.90	0.8999	0.9000
Linear SVM (With tuning)	0.90	0.90	0.90	0.9024	0.9025

Table 4.1. Comparision of all the models

5. Conclusion

In conclusion, an analysis of the Large Movie Review Dataset was conducted using various machine learning models for sentiment analysis. Following the assessment of their performance, it was determined that Linear SVM was the most accurate model for predicting the sentiment of movie reviews, with an accuracy of 90.25%. It was also found that the XGBoost model had a good performance but could not be further improved with hyperparameter tuning. On the other hand, the deep learning model LSTM had a lower accuracy of 85.34% but showed potential for improvement with more training epochs. Based on these results, the Linear SVM model is used in the Streamlit web app for predicting the sentiment of movie reviews. Users can enter their movie review and receive a predicted sentiment label, allowing them to quickly assess the overall sentiment of the review. This web app provides an easy-to-use tool for movie enthusiasts, critics, and industry professionals to evaluate the sentiment of movie reviews and make informed decisions.

6. References

6.1. Dataset Kaggle link

https://www.kaggle.com/datasets/lakshmi25npathi/imdb-dataset-of-50k-movie-reviews

7. Appendix

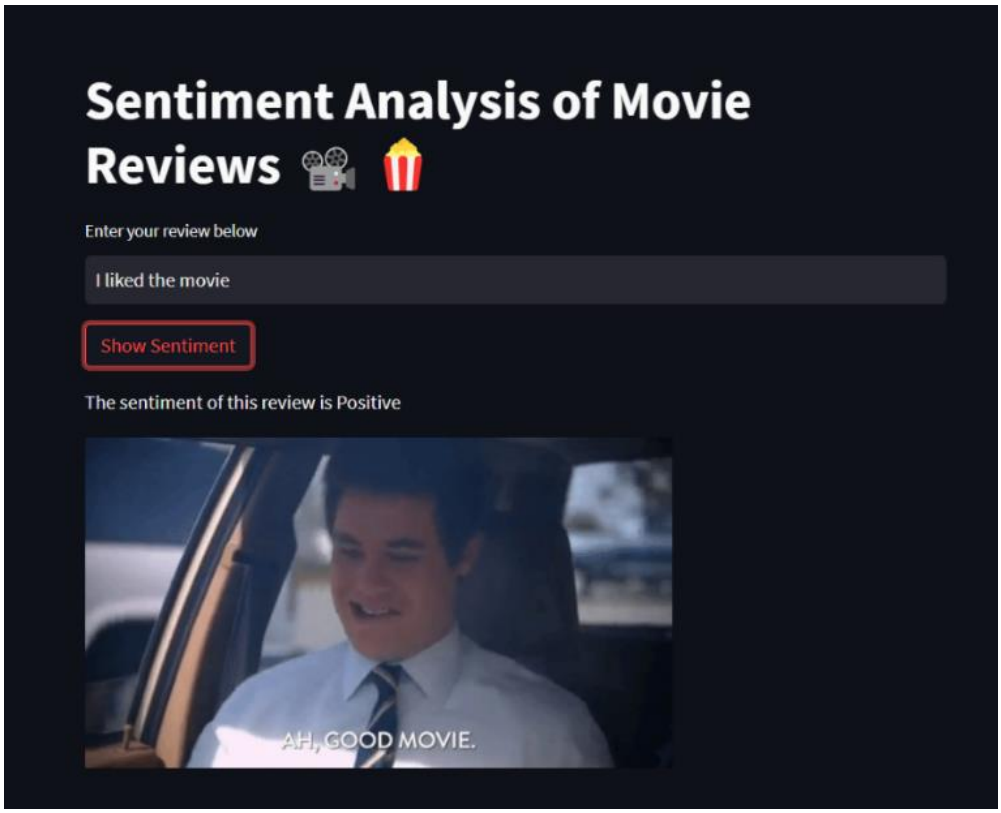
7.1. GitHub Link:

https://github.com/KoushikSai01/SentimentAnalysis

7.2. Streamlit Link:

https://koushiksai01-sentimentanalysi-movierev-sentimentanalysis-8qhwk5.streamlit.app/

7.3. Snapshot of Streamlit



Sentiment Analysis of Movie Reviews 📽️ 🍿

Enter your review below

I hated the movie

Show Sentiment

The sentiment of this review is Negative