Movie Review Sentiment Prediction

Beatrice(Jiaqi) Xu, Kevin Li, Xuanang Li,
Yining Xu, Yuexuan Wu , Zehui Chen

December 16, 2023

Abstract

We want to examine the relationship between the 50000 movie reviews and their associated sentiments given in an IMDb dataset. After transforming texts to numerical data and reducing dimensions of the features, we utilize various statistical models such as logistic regression, KNN, LDA, QDA, random forests, SVM, and neutral network to predict the sentiments by keywords and key phrases used in the reviews.

1 Introduction

IMDb, or Internet Movie Database, is a comprehensive online database encompassing films, TV series, podcasts, home videos, video games, and streaming content. It offers details on cast, production crew, biographies, plot summaries, trivia, ratings, and reviews. In this project, we utilize the IMDb dataset, which comprises two primary columns. The initial column, labeled "review," encompasses comments and reflections written by various users pertaining to diverse films. The subsequent column, labeled "sentiment," contains the sentiment associated with each review, classifying it as either positive or negative. Our project focuses on predicting sentiment in IMDb reviews by analyzing the content and examining the correlation between sentiments and reviews using various statistical models.

2 Preparation Steps

In our data preprocessing, we utilized a lexicon to align words in reviews, aiming to predict sentiment. This involved meticulous word comparisons to establish their relevance for sentiment analysis. Subsequently, we employed data cleaning to optimize the dataset, eliminating noise and refining it for more meaningful analysis. We also applied feature reduction techniques—Term Frequency-Inverse Document Frequency (TF-IDF) and Principal Component Analysis (PCA). TF-IDF highlighted term significance, while PCA simplified data complexity by distilling essential patterns in the dataset. These steps collectively aimed to enhance the efficiency of our sentiment analysis.

2.1 Data Cleaning

To better understand the dataset, we first constructed some descriptive statistics.

Sentiment	Number of Reviews
Positive	25000
Negative	25000

Table 1: Distribution of Reviews by Sentiment

We observed an equal distribution of 25,000 reviews classified as positive and 25,000 reviews classified as negative. This equitable distribution underscores the balanced nature of the dataset, facilitating more evenly distributed training and testing sets for improving statistical model performance by preventing bias towards the majority class.

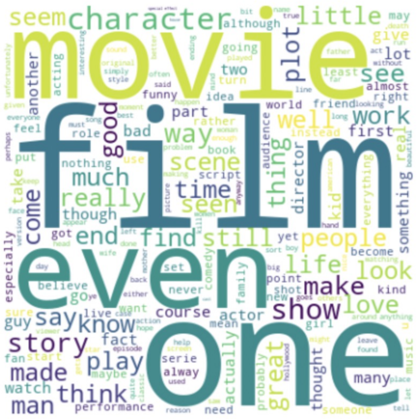

Figure 1: Frequency of Words in Reviews after Data Cleaning.

As we proceed to our basic data processing, we noticed a prevalent occurrence of stop words in the reviews. After the removal of HTML tags, non-letter characters, and common stop words, we reduced noise from the dataset and enhanced the quality of textual content. A large number of extraneous words are filtered out, and we could see a prevalence of terms such as "good", "bad", and "love" in the reviews which significantly contribute to sentiment expression.

However, despite these positive outcomes, we still observed a more substantial presence of neutral terms like “movies”, “film”, “one”, and “even” in the reviews. To further refine our dataset and enhance sentiment prediction accuracy, we opted to leverage lexicons downloaded from Kaggle by matching words in the reviews with entries in the lexicons, enabling us to extract a smaller set of words that are more indicative of strong sentiment tendencies.

We utilized both negative and positive lexicons to predict sentiment in reviews. To simplify our word-matching process, we merged the negative and positive lexicons into a dictionary with a total length of 6786. It is obvious that the dimension of this dictionary is very high. We need to reduce the number of calculations. Because not all words in this dictionary appear in comments, we calculated the number of occurrences of each word and deleted words that did not appear in comments. Now this dictionary still has 5814 words. We vectorized this dictionary into a 50000 $\times$ 5814 matrix based on 50000 comments and converted the array into a dataframe to prepare for the subsequent feature engineering. Moreover, for the IMDb dataset, we converted the sentiment column values, changing “positive” to 1 and “negative” to 0 for numerical representation.

2.2 Feature Engineering

Following the completion of our data cleaning procedures, our dataset was reduced to 5814 features. The challenges posed by managing such substantial features prompted us to perform feature reduction techniques. Feature engineering is pivotal, particularly because 1) our task is a prediction task, and 2) we are facing high-dimensional text data. By applying feature reduction, we strike a balance between eliminating redundant attributes that may lead to overfitting and retaining as much meaningful variation as possible in the predictors.

In our pursuit of an optimized dataset, we explored the unprocessed dataset alongside two prominent feature reduction methods: Term Frequency-Inverse Document Frequency (TF-IDF) and Principal Component Analysis (PCA). Opting for an initial analysis, we utilized the unprocessed dataset to directly evaluate the performance of the K-Nearest Neighbors (KNN) and Random Forest algorithms. Setting the parameter for KNN at 5 neighbors yielded an accuracy of 0.6939, while Random Forest exhibited a notable accuracy of 0.8350. However, other models were not able to produce results because the dimension is too high and they are computationally expensive, implying we need to use feature reduction to select more representative data for prediction.

We then implemented the Term Frequency-Inverse Document Frequency (TF-IDF) technique. Term Frequency (TF) measures the frequency of a term, typically a word, in a given textual document, while Inverse Document Frequency (IDF) is calculated as

$$IDF(t) = \log \frac{\text{Number of documents}}{\text{Number of documents with term t}}$$

A higher IDF indicates the term is more significant as a feature. By employing TF-IDF = TF $\times$ IDF, we curated a dictionary of words and reduced the feature set. Subsequently, to determine the TF-IDF thresholds for feature selection, we first selected words with TF-IDF values ranging from 0.2 to 0.7, and observed an accuracy of 0.5824 for a KNN model (K = 5) and 0.6418 for a random forest model. To refine our feature selection, we narrowed our scope to words with TF-IDF values between 0.5 and 0.9. Through accuracy assessments on the same models, we aimed to discern improvements in predictive performance. The results indeed demonstrated enhanced accuracy, yielding 0.61196 for the 5-NN model and 0.7294 for the random forest model. However, despite this refinement, our feature set still encompassed more than 1,000 words, indicating a persistently high-dimensional space. The corresponding accuracy, while improved, did not reach our desired levels, which prompted us to try other methods for dimension reduction.

In our quest for more effective dimensionality reduction, we then turned to the Principal Component Analysis (PCA) to simplify the dataset and assess its impact on accuracy. The process began with the standardization of the continuous initial variables to ensure equitable contributions to the analysis. Following this, we computed the covariance matrix of the standardized features. After calculating eigenvalues and eigenvectors and subsequently arranging eigenvalues in descending order, we selected the top eigenvectors to capture the desired variance.

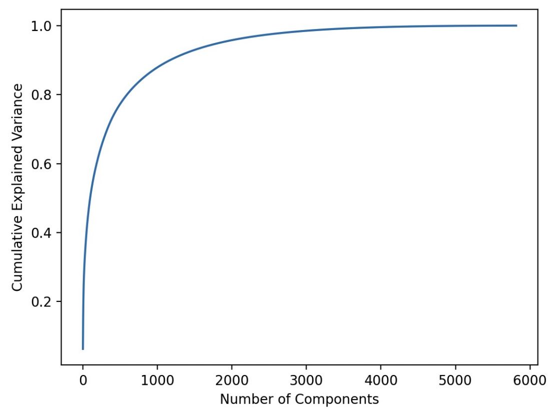

Figure 2: Principal Components by Proportion of Variance

PCA finds linear combinations of the given raw predictors that are orthogonal to each other and contain the most variance in order to reduce the overall number of predictors while still containing as much variation as possible. In order to determine the best number of predictors, we calculated all 5814 components in order by variance explained and then plotted their cumulative variance explained by each component.

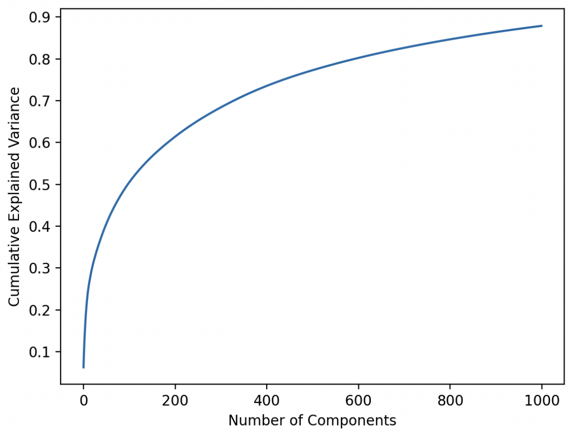

Figure 3: Top 1000 Principal Components by Proportion of Variance

We can see in Figure 2 that there seems to be a plateau in variance explained after 1500 components. However, that would not be a sizable reduction in dimensionality. In Figure 3 and from outputting the cumulative variance, around 80 % of the variance can be explained by the first 500 components. For this reason, we selected the top 500 components.

The chosen eigenvectors were then utilized to project the data onto a reduced set of components, effectively reducing the dimensionality of the dataset. Post-PCA implementation, our analysis revealed an accuracy of 0.6916 for KNN with 5 neighbors and an accuracy of 0.7485 for the Random Forest algorithm. These outcomes shed light on the effectiveness of PCA in enhancing the efficiency of our models, demonstrating its ability to streamline complex datasets while still maintaining a high degree of accuracy in predictive analyses.

2.3 Comparison between different techniques

Upon comparing the accuracy of the three cases mentioned earlier, it becomes evident that both PCA and the original dataset yield higher accuracy than TF-IDF, with PCA achieving a slightly higher prediction accuracy than using the unprocessed dataset. Further, when working with raw data, the abundance of features posed a limitation, preventing the application of certain statistical models like LDA and QDA. Consequently, to address the challenge of feature overload and optimize the model, we favor PCA for its effectiveness in feature reduction and overall model enhancement.

3 Experiment

3.1 Logistics Regression

Logistic Regression is a key statistical tool for predicting categorical outcomes by modeling the relationship between a dependent variable and independent variables. Extending from Linear Regression, it focuses on probabilities, employing the logit function to transform its characteristic S-curve into a linear form, making it ideal for categorical prediction tasks. Given our focus on predicting sentiments as either positive or negative – a binary classification task – alongside the logistic model's effectiveness in interpretation, we opt for Logistic Regression aiming to derive accurate predictions and insightful insights from the reviews.

Logistic Regression usually fits well with the training data but can struggle with testing data, often called overfitting. To deal with this issue, regularization comes in handy by keeping large coefficients in check to ensure better generalization and prevent overfitting. When it comes to regularization, we have two options: L1 regularization, also known as Lasso Regression, and L2 regularization, or Ridge Regression. L1 tends to zero out certain coefficients, making the model sparser, while L2 encourages small, non-zero coefficients, leading to less sparse models. In our logistic regression model, we've chosen to go with L2 regularization because L1 tends to introduce sparsity. To find the optimal penalty parameter, we chose to use Grid Search Cross Validation, yielding the optimal value of 0.001. Considering we've already applied PCA for predictor reduction, we didn't find it necessary to further reduce variables. This decision aligns with the aim of striking a balance between avoiding overfitting and maintaining a model that captures essential features.

After training the Logistic Regression model, we can get the result in figure 4.

As we can see, the ROC curve very much approaches 1, indicating a good performance of the logistic regression model in classifying the positive and negative instances correctly. An AUC of 0.92 suggests that there is a 92% chance that the model will be able to distinguish between positive class and negative class. Therefore, we can conclude that the logistic regression model has a high predictive performance and is much better than a random guess at classifying the positive and negative cases.

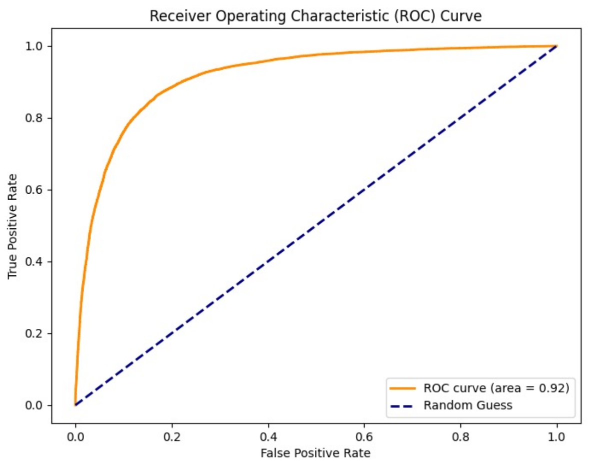

Figure 4: ROC Curve of the Logistic Regression Model

3.2 K-Nearest Neighbors

In the K-Nearest Neighbors (K-NN) algorithm, the parameter “k” signifies the number of neighbors considered when classifying a given query point. Smaller “k” values result in higher variance and lower bias, while larger “k” values tend to introduce higher bias and lower variance. The optimal choice of “k” depends on the nature of the input data. We want to use an odd “k” to prevent tie-break issues in classification and employ cross-validation techniques to identify the most suitable “k” for our dataset.

In our dataset, PCA effectively reduced the feature dimensionality to 500, leading to the classification of points within a high-dimensional 500-space. The classification of a data point is determined by the majority classification of its neighbors, either labeled as “true” for a positive classification or “false” for a negative one.

After training the KNN model on dataset using 5-fold cross validation, we selected multiple “k” values and calculated the accuracy:

k	Accuracy
5	0.6955
15	0.7608
25	0.7804

Table 2: KNN CV Results

As we can see, among K = 5, 15, and 25, setting K = 25 yields the highest accuracy of 0.7804. This accuracy serves as a baseline when we perform more advanced technique to obtain a better K.

One technique that we used is the Grid Search Cross Validation. This technique is a method to find the best hyperparameter in a model. When we employed the technique on KNN, we provided a predefined set of possible hyperparameter K we wish to be considered, and an exhaustive search over the predefined set is performed. The best hyperparameter is then chosen via cross-validation on each of the possible hyperparameters. The result of Grid Search CV is that setting K = 151 results in the highest accuracy of 0.8101.

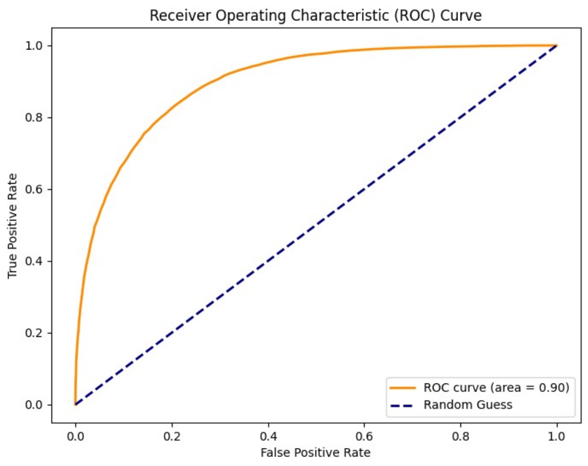

Figure 5: ROC Curve of the KNN Model

Based on our results, we can see that the ROC curve of K equaling to 151 approaches 1, so it is much better than random guess. However, even though KNN is recognized for its flexibility and suitability for large datasets, certain limitations persist. Particularly, in the context of a high-dimensional dataset like ours, the computational cost associated with calculating distances for every data point can become prohibitively expensive. Given these considerations, it is crucial to explore alternative models to address these challenges effectively.

3.3 Support Vector Machines

Support Vector Machines (SVMs) are designed for both classification and regression tasks. The primary objective of SVMs is to find an optimal hyperplane that maximally separates different classes in the input space. This hyperplane is determined by identifying support vectors, which are data points that define the decision boundary. Also, the kernel trick achieved through the use of kernel functions allows SVMs to implicitly map data into higher-dimensional spaces, indicating SVMs’ effectiveness in capturing complex relationships.

Since 1) our data is high-dimensional and sparse text data, 2) our task is a prediction task and the decision boundary will be a hyperplane, we then trained an SVM model to determine how well it could be used to predict sentiments from the written reviews. Similar to our previous models, we trained the model on the data with 500 principal components from PCA as predictors and whether the sentiment is positive or negative as the response variable.

The training process involved meticulous adjustments to hyperparameters to optimize the model's performance. Fine-tuning iterations included the exploration of various kernel functions, regularization parameters, and the selection of appropriate values for the cost parameter (C). This iterative refinement aimed to strike a balance between model complexity and generalizability. Additionally, we conducted cross-validation to assess the model's robustness and to gauge its performance on the testing data.

Upon applying the tuned SVM model to the testing set, we arrived at the following result:

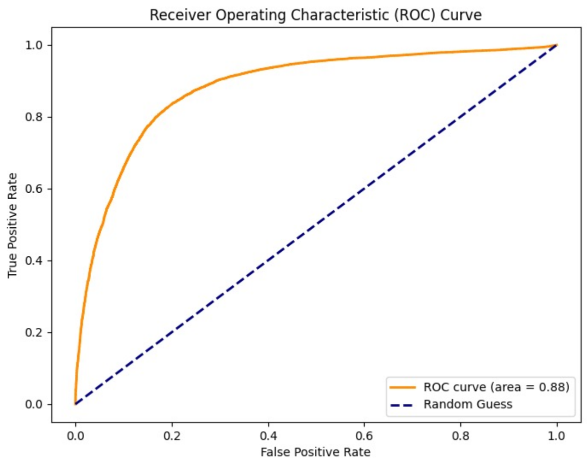

Figure 6: ROC Curve of SVM Model

AUC equaling 0.88 indicates the SVM model is an effective model predicting sentiments from the reviews. However, since SVMs are very sensitive to the choice of hyperparameters, and they are computationally expensive due to our dataset and number of features being large, it leaves room for exploration of further optimization with other models.

3.4 Linear and Quadratic Determinant Analysis

The three models we have examined so far are discriminative models that focus on capturing the conditional probability of the response variable given the data. On the other hand, generative models offer a different perspective by capturing the joint probability distribution of the response variable and the predictors. Therefore, we now shifted our interest to exploring generative models' performance on prediction. The first two models we looked at are Linear Discriminant Analysis (LDA) and Quadratic Discriminant Analysis (QDA).

LDA and QDA are classic classifiers, each employing distinct decision surfaces – linear and quadratic, respectively. As implied by their names, LDA and QDA serve as effective tools for supervised classification. In particular, LDA can be utilized for supervised dimensionality reduction. This involves projecting input data onto a linear subspace characterized by directions that maximize the separation between classes. Notably, the output dimension is inherently less than the number of classes, rendering it a potent form of dimensionality reduction, especially in multiclass scenarios.

In both LDA and QDA, we make a key assumption: the conditional distribution of predictors given the response variable adheres to a multivariate normal distribution. The distinction lies in their treatment of the covariance matrix. LDA assumes that the covariance matrix of predictors is uniform across both classes, while QDA does not impose this assumption, allowing for different covariance matrices for each class.

LDA and QDA hold appeal due to their lack of hyperparameters to fine-tune. Upon fitting both LDA and QDA models to the training set, the subsequent results were obtained when applied to the test set:

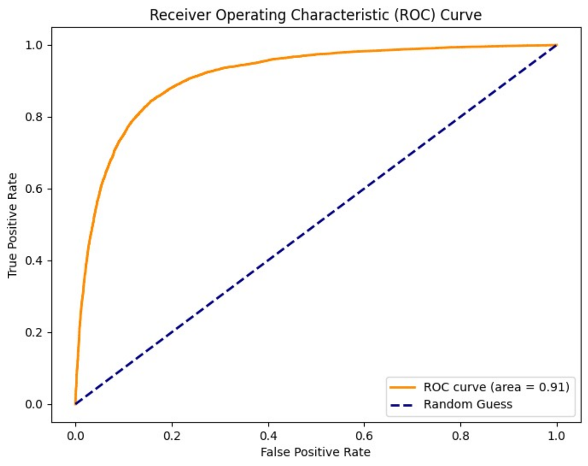

Figure 7: ROC Curve of the LDA Model

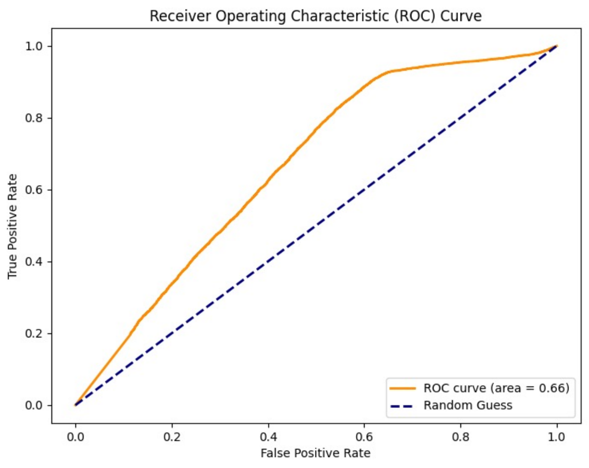

Figure 8: ROC Curve of the QDA Model

In our comparison, LDA has demonstrated superior performance across all evaluation metrics compared to QDA. This outcome can be attributed to the inherent flexibility of QDA, considering that LDA is a specific instance within the broader QDA framework. Notably, the performance alignment between LDA and Logistic Regression is consistent with general expectations and is reflected in our specific models.

3.5 Random Forests

Random Forest algorithms involve three key hyperparameters: node size, the number of trees, and the number of features sampled. This versatile classifier is effective for both regression and classification problems. The algorithm creates an ensemble of decision trees, each constructed from a data sample drawn with replacement from the training set (bootstrap sample). The sample is divided into training and out-of-bag (oob) portions, enhancing randomness. Feature bagging further diversifies the dataset, reducing correlation among trees. In regression tasks, decision trees are averaged, while classification relies on a majority vote for the predicted class. The oob sample aids in cross-validation for final predictions.

We employed a Random Forest model to assess its fit to our movie review dataset. Additionally, we trained the model using 500 features derived from PCA feature reduction. Given the numerous hyperparameters influencing the training process in Random Forest models, we opted for Random Search Cross Validation to explore various combinations as it is more computationally efficient than the exhaustive Grid Search Cross Validation. Given a set of combinations, the Random Search CV selected optimal hyperparameters to include a maximum tree depth of 150, using the square root to determine the maximum number of features per tree, a minimum of 2 samples per leaf, a minimum of 10 samples for a split to occur, and the training of a total of 200 trees, achieving an accuracy of 0.82684.

After plotting the ROC curve, we observed:

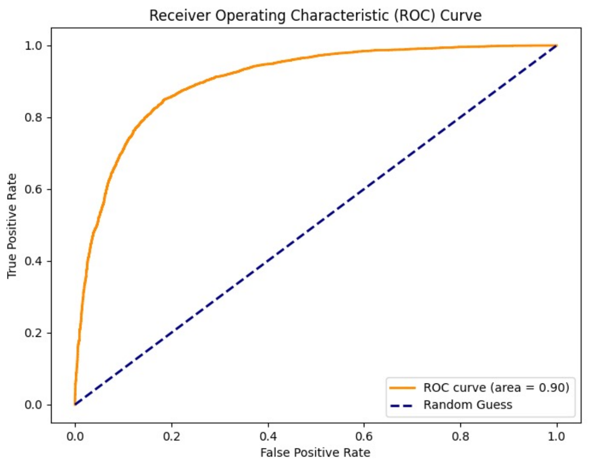

Figure 9: ROC Curve of the Random Forest

From the ROC curve, we can see that the random forest model is effective in correctly identifying positive instances (high sensitivity) while minimizing the misclassification of negative instances (high specificity). AUC equaling 0.90 also reflects that random forest is a relatively strong model for predicting sentiments from reviews correctly.

3.6 Neural Network

To further improve the prediction accuracy and since our data is sequential text data, we decided to explore alternative methods beyond these methods. In this section, we used Neural Networks, specifically employing a Long Short-Term Memory (LSTM) architecture, to tackle the challenges posed by the dataset and enhance sentiment analysis.

The core of our LSTM-based Neural Network (Model 1) architecture consists of the following key layers (in order):

1. Embedding Layer: Converts tokenizer-generated indices to dense vectors, facilitating the learning of meaningful representations of words that are crucial for sentiment analysis.

2. LSTM Layer: With 60 neurons and return_sequences = True, this layer captures and remembers long-term dependencies within sequential data, making it capable of recognizing sentiment nuances in movie reviews.

3. GlobalMaxPool1D Layer: Extracts the most relevant features from the sequences, reducing dimensionality and providing a global perspective on the data.

4. Dropout Layer (×2): Prevent overfitting by randomly deactivating a fraction (we chose 0.2) of neurons during training.

5. Dense Layer (×2): Used after each dropout layer to introduce non-linearity, allowing the model to learn and approximate complex, non-linear relationships in the data. The first dense layer contains 50 neurons and utilizes the rectified linear unit (ReLU) activation function. The second dense layer, the output layer, contains 2 neurons and utilizes the softmax activation function to convert the output into probabilities for the two sentiment classes, which enables the binary classification of input data.

This Neural Network approach applied in our sentiment analysis task provides a sophisticated solution for handling the challenges posed by the dataset. Unlike the other approaches, a Neural Network, specifically an architecture employing Long Short-Term Memory (LSTM), is adept at capturing intricate patterns and relationships within sequential data, making it well-suited for natural language processing tasks.

In order to discover more about the possibilities of our neural network approach, we modified the configurations of layers of our initial model (Model 1) architecture and created two additional new models with modified layer structures (Model 2 and Model 3). We then applied the new models to our dataset to see whether they could result in better test accuracy:

- Model 2: This model modifies the structure of Model 1 by changing the LSTM layer into a Bidirectional LSTM layer while keeping the same parameters. Another Bidirectional LSTM layer with 30 neurons is also added right after the first Bidirectional LSTM layer.

- Model 3: This model modifies the structure of Model 2 by adding a Conv1D layer with 128 filters and a kernel size of 5 right after the Embedding layer. A MaxPooling1D layer with a kernel size of 5 is then added right after the Conv1D layer.

Model	Accuracy
1	0.8698
2	0.8735
3	0.8881

Table 3: Neural Network Results

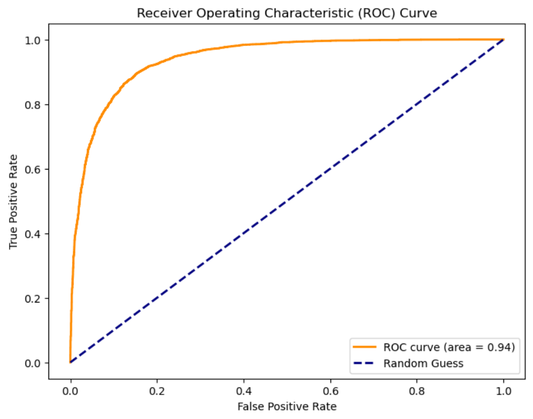

Figure 10: ROC Curve of Neural Network Model 3

It appears that the third model achieves the highest test accuracy among the three models, suggesting that the added complexity and features in the third model contribute to better performance in sentiment prediction compared to the previous two.

The test accuracy of 0.8881 underscores the model's proficiency in classifying sentiments within movie reviews. This performance surpasses the results achieved with methods used in previous sections, indicating the better capabilities of the LSTM-based Neural Network approach.

In conclusion, the Neural Network approach adeptly addresses the challenges of transforming textual data into numerical form and handling high-dimensional features, and manifests as a powerful tool for sentiment classification on this extensive movie review dataset.

4 Conclusion and Summary

According to the previous models, we can come to our final analysis and results.

4.1 Results and Analysis

To assess our models, we begin by comparing their accuracy scores on the test dataset. The following table showcases the accuracy scores of each model, arranged in ascending order from the lowest to the highest.

The LSTM-based Neural Network attains the highest test accuracy in sentiment classification, leveraging its inherent capacity to capture sequential dependencies in text data. LSTM networks excel in processing sequences, enabling them to effectively capture sentiment nuances and interdependencies within textual information. The recurrent nature of LSTMs allows for consideration of the entire context in a text, recognizing subtle sentiment

Model	Accuracy
QDA	0.5535
KNN	0.8101
SVM	0.8170
RF	0.8268
LDA	0.8414
LR	0.8463
Neural Network	0.8881

Table 4: Distribution of Reviews by Sentiment

cues and dependencies essential for accurate sentiment classification.

The fact that the Logistic Regression model achieves the second-highest test accuracy and the LDA model achieves the third-highest test accuracy suggests that both models are effective in recognizing the sentiment patterns within the reviews in the IMDb dataset. Logistic Regression is suitable for binary classification tasks such as sentiment analysis, leveraging linear decision boundaries to distinguish between positive and negative sentiments. Since movie reviews usually contain key terms that correlate linearly with specific sentiments, the textual characteristics of the reviews in our dataset are likely to display a discernible linearity that aligns well with the linear decision boundaries of Logistic Regression. On the other hand, Linear Discriminant Analysis (LDA) also performs well in capturing the underlying structure of sentiment by modeling the distribution of features in each class. LDA inherently performs dimensionality reduction by projecting the data onto a lower-dimensional space. This can be advantageous in sentiment analysis when dealing with a large number of features (words in reviews). The reduced-dimensional representation may highlight essential sentiment-related features. Furthermore, LDA considers the correlation between features, making it suitable for capturing complex relationships and dependencies within the textual content of reviews.

The Quadratic Discriminant Analysis (QDA) model has the lowest test accuracy, which could stem from its assumption of distinct covariance matrices for each class. While such flexibility enables QDA to capture complex relationships between features, it also increases the risk of overfitting. In the context of our IMDb dataset, where sentiment patterns might not clearly demonstrate quadratic separation, the QDA model's flexibility in estimating class covariance matrices could result in a fit that is not optimal.

4.2 Final Remarks

While our project has achieved accuracies above the 0.5 threshold, indicating performance better than random guessing, there remain areas for improvement and considerations for future iterations.

Firstly, the assumption of independence in our dataset might not hold, as newer reviews could be influenced by older ones. User perceptions may be shaped by the reviews they encounter before visiting a platform, potentially impacting their sentiments and subsequently the reviews they write. This interdependence introduces complexity and may warrant exploration in future analyses. Exploring causal inference techniques could be a promising avenue to disentangle these temporal relationships.

Secondly, employing a range of statistical models, including logistic regression, KNN, LDA, QDA, random forests, SVM, and neural networks, offers a comprehensive analysis of sentiment prediction. However, we should still be cautious about the potential overfitting. Ensuring robust model generalization to unseen data is still important.

Furthermore, the chosen text analysis method (tfidf) is only centered on keyword frequencies, it may benefit from a more nuanced understanding of contextual intricacies. Adopting a sophisticated natural language processing approach that considers the context surrounding words and phrases can potentially refine sentiment analysis. The Principal Component Analysis (PCA) step in dimension reduction is recognized for its subjective threshold of 0.80. Exploring alternative thresholds or adopting data-driven methods to determine the optimal dimension reduction level can contribute to a more effective data representation.

Lastly, despite the Neural Network demonstrating superior accuracy at 0.8706, using techniques such as K-fold cross-validation may further increase model robustness. This approach validates the model's performance across diverse data subsets, providing a more reliable estimate of its predictive capabilities.

In summary, despite the dataset's balance, addressing concerns related to causation and dependency, potential overfitting, nuances in text analysis, dimension reduction, and model validation can further elevate the accuracy and generalization of sentiment prediction models.