IMDb Sentiment Analysis

Kevin Hamakawa (UID: 505745969), Kevin Kim (UID: 604994907),
Matthew Woo (UID: 905589721), Min Jung Kim (UID: 305579800),
Alan Wong (UID: 705773980), Sung Joon Lee (UID: 905724818)

January 20, 2024

Abstract

IMDb is an online platform where users can provide ratings and reviews for movies, TV shows, and other forms of visual entertainment. Our analysis aims to explore the connection between the sentiment of the review and the text itself, ultimately focusing on predicting sentiment ratings from the textual content. To achieve this, we plan to employ various machine learning models, such as logistic regression, K-nearest neighbors (KNN), random forests, and more. Our ultimate goal is to use various methods of testing in order to figure out which model performs the best on unseen data.

1 Introduction

In machine learning, the capability to precisely evaluate, analyze, and predict the outcome from a large dataset is a key component of problem-solving. With a particular focus on the extensive and frequently nuanced dataset of IMDb movie reviews, this project explores the intriguing and complicated field of sentiment analysis. We are driven by the particular challenge posed by the textual data, which calls for advanced methods for converting subjective judgments into quantitative measures that can be used for analysis and future projection.

The main goal of our work is to create a model that can identify the underlying emotions in movie reviews. This task is essential for natural language processing and has practical applications in market analysis and consumer feedback. We intend to successfully handle the complexity of human language, prepare the textual data, and accurately apply machine learning to the vast datasets that are subjective and qualitative.

We want to develop a model that can generate sentiment predictions on a high accuracy level and offer valuable insights into the causes of influencing these feelings in the reviews by utilizing various methods such as principal component analysis (PCA), logistic regression, linear and quadratic discriminant analysis (LDA/QDA), K-nearest neighbors (KNN), and random forest. This venture can go beyond a simple academic study by taking a step forward to comprehending the intricate relationship between human emotions and their expression through literary language.

As we begin this project, we anticipate facing a variety of difficulties. The most important of these are the nuances of data preparation, where the objective is to carefully convert unprocessed textual data into a format that can be analyzed without ruining its contextual information. Model selection is an equally challenging operation, where we have to choose through many machine learning algorithms to find the one that best fits the properties of our data and our analytical goals. The objective of our project is to create a flexible structure that may be used for subsequent studies in this continuously developing subject matter, in addition to addressing the dataset as it stands.

1.1 Dataset

Our sentiment analysis research is built upon a comprehensive and intricate collection of IMDb movie reviews. This dataset offers a wide variety of text-based information that captures many facets of audience perception and movie criticism, making it an optimal testing ground for our machine learning algorithms. Fundamentally, the dataset is organized mostly on the review texts, which makes it a particularly difficult yet informative database for analyzing sentiment. The dataset's textual structure

requires a thorough pretreatment method to ensure that the data is both in an appropriate format for modeling and preserves the crucial subjective elements of the evaluations.

1.2 Exploratory Statistics

Before doing any kind of modeling, let us analyze our data with some exploratory data analysis. The first column we should look at is the "reviews" column, which we can visualize using a simple word cloud.

Figure 1: IMDb Reviews Word Cloud

This word cloud was made by removing all of the non-English words, as well as removing stop words such as "and", "the", and "is". From this word cloud, you can tell that the dataset has to deal with movie reviews - a large majority of the words contain either movie-related words such as "movie", "film", "see", and "watch" or sentiment-related words such as "great", "good", "bad", and "think".

Now, let us analyze the other column of data we have: the "sentiment" column. We can visualize this using a simple bar chart. As you can see, the distribution looks almost perfectly split. In fact, if

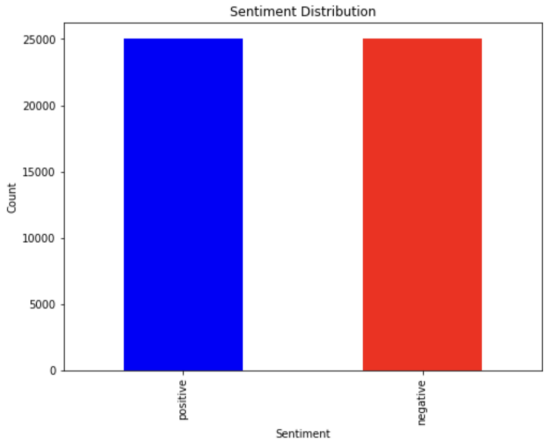

Figure 2: Side-by-Side Barplot of Sentiment

we look at the value counts themselves, we can learn that there are 25,000 positive reviews, and 25,000 negative reviews.

2 Preprocessing Data

One of the most important steps for any machine learning pipeline is preprocessing the data, or making sure the data that we are modeling with is in the proper format to be modeled and analyzed. Because there was only one predictor column for this dataset, we only needed to process the actual reviews themselves. No programming language can understand text data on its own, and no model we are going to be testing out can as well. As a result, a crucial step for this IMDb dataset was "vectorizing" the reviews, or turning the text data into numbers that KNN, logistic regression, or any other model can understand, train, and optimize.

2.1 Cleaning Reviews

Before vectorizing, we needed to clean the actual text itself, to make sure that the words being vectorized were words that adequately summarized the review. In other words, it was important to remove stop words such as "and", "the", and "is" from our reviews, as including them would very likely skew our vectors and predictions. Beyond this, we also decided to remove special characters and convert the review columns to lowercase, so that capitalization would not create two or more separate terms in TF-IDF.

As a result, a sentence that may have originally looked like this:

"I really loved the movie Star Wars!!! I enjoyed all of the action and plot twists :)",

will now look like this:

"really loved movie star wars enjoyed action plot twists".

This will allow for our TF-IDF vectors to truly capture the proper and most important meanings of the IMDB reviews and sentences.

2.2 TF-IDF

TF-IDF [Gee23c] is a metric that assesses the significance or relevance of word occurrences within a document in comparison to a set of documents. In TF-IDF, there are two main parts:

- Term frequency, or TF, is the number of times a word appears in relation to the total number of words in the document.

$$TF(t,j) = \frac{\text{Term t frequency in document } j}{\text{Total words in document } j}$$

- Inverse document frequency, or IDF, is the measure of how rare or unique a word is in a collection of documents.

$$IDF(t) = \log_2 \frac{\text{Total number of documents}}{\text{Documents with term } t}$$

Furthermore, we also decided to use both unigrams and bigrams as a consideration for our TF-IDF conversion. In other words, if we have the phrase "I love stats", the phrases added as terms would be "I", "love", "stats", "I love", "love stats". The reason we decided to also utilize bigrams was to add more context to specific phrases and words. We did not decide to include trigrams in our TF-IDF vectors, as this would create far too many features and phrases that would potentially lead to overfitting issues. Bigrams allowed us to give words and phrases a small amount of context without tremendously increasing the number of features.

Finally, we believed limiting the number of maximum features for our vectors was necessary. Because of this, we set the maximum number of features to 2,000. This means that our TF-IDF vectors will contain the 2,000 most common unigrams and bigrams. Through our preprocessing steps, we were able to transition from a body of text to a numerical vector that we can now input into numerical classification models.

2.3 Splitting Data

We also decided to do an 80-20 split for our dataset, with 80% of the data being used to train our models, and the remaining 20% being used to test our models.

2.4 Principal Component Analysis

Unfortunately, having a vector space of 2,000 is not ideal for any kind of binary classification problem. The main reasons for this are because of:

1. The curse of dimensionality: Models such as KNN get significantly worse with such a large vector space
2. Overfitting: A high number of features in comparison to the number of observations in the dataset can lead to potential overfitting
3. Computational Efficiency: Fitting some of the larger and more complex models would not be very efficient with a dataset that is 40,000 by 2,000.

As a result, we will use an unsupervised learning technique called Principal Component Analysis (PCA) [Set20] to help with dimensionality reduction. PCA works by finding which directions in the vector space that the data varies the most. This will allow us to filter out the noise and help emphasize the most important parts of our data.

By using PCA, we were able to reduce the number of columns from 2,000 all the way down to 100, while still maintaining roughly 80% of the original variance. This results in a data frame with 100 predictor columns and 1 output column - a much more convenient dataset to model with than the original TF-IDF dataset.

3 Supervised Models

3.1 Logistic Regression

In logistic regression [EM17], we aim to estimate the conditional probability of a binary outcome based on a set of predictors. This methodology extends the principles of linear regression, which is typically employed for predicting continuous variables. A common challenge in predictive modeling is the tendency for models to overfit the training data. Overfitting is characterized by a model's diminished performance on unseen data, often attributed to large coefficient values in the model. Regularization techniques are employed to mitigate such issues by constraining the magnitude of the coefficients, thereby enhancing the model's generalizability. The two common regularization techniques are L1 (lasso) and L2 (ridge) regularization. In this study, L2 (ridge) regularization was chosen. The rationale behind this selection lies in the nature of L1 regularization to induce sparsity in the model coefficients. Given that Principle Component Analysis (PCA) was already utilized to reduce the predictor space, further variable elimination was deemed unnecessary.

For our logistic regression model using ridge regression, only one hyperparameter required tuning, which was the λ value controlling the regularization strength. Lower values of lambda allow the model to fit more closely to the training data by placing less emphasis on the regularization penalty, whereas higher values of λ increase the weight of this penalty, leading to greater shrinkage of the coefficients. We employed Grid Search Cross Validation to find the optimal λ, which was determined to be $\lambda = 0.01629751$. This value provided the best balance (of Bias Variance Tradeoff) in our model's performance. After training the logistic regression model, we received the following results from the testing dataset.

Metric	Logistic Regression
Accuracy	0.8572
AUC	0.9306

Figure 3: Performance Metrics for the Logistic Regression Model

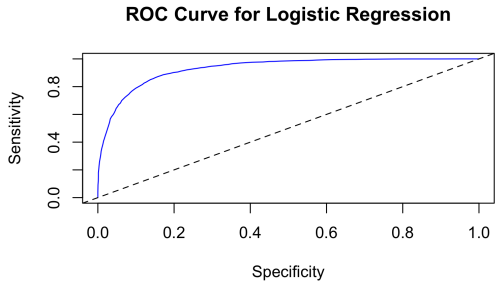

Figure 4: ROC curve for Logistic Regression Model

Our logistic regression model came out to have a sensitivity of 84.92% and specificity of 89.51%. Since the ROC curve is [Nar21] closer to the upper left corner of the graph, we concluded that there is high accuracy. The area under the curve (AUC) has a value of 0.9306, which presents that the Logistic Regression model performed well.

3.2 LDA/QDA

Discriminant models (Linear Discriminant Analysis (LDA)[Gee23b], Quadratic Discriminant Analysis (QDA)[W+22]), stand out as robust models when dealing with small datasets where predictors follow approximately normal distributions within classes. These methods are used, especially in scenarios with more than two non-ordinal response classes. Moreover, their advantages come with assumptions: both LDA and QDA assume predictors follow a multivariate Gaussian distribution, with LDA further assuming equal covariances among predictors for all response levels—a condition relaxed in QDA.

LDA is less flexible but has a lower variance than QDA, potentially leading to superior prediction performance with fewer training observations, where reducing variance is critical. However, LDA's reliance on the assumption of shared variance across response classes may lead to high bias if this assumption is significantly violated. On the other hand, QDA's increased flexibility becomes advantageous with large training sets or when the assumption of a common covariance matrix is implausible. We tried LDA and QDA models with a training set, and the experiment obtained the following results.

Metric	LDA	QDA
Accuracy	0.8549	0.7905
AUC	0.9291	0.8613

Table 1: Performance Metrics for LDA & QDA Model

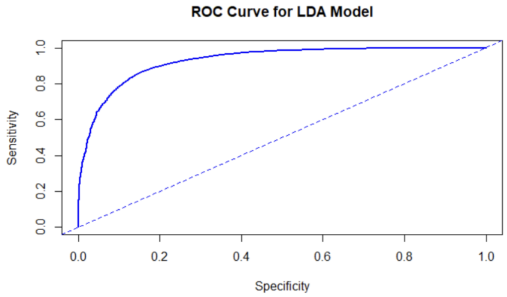

Figure 5: ROC curve for LDA

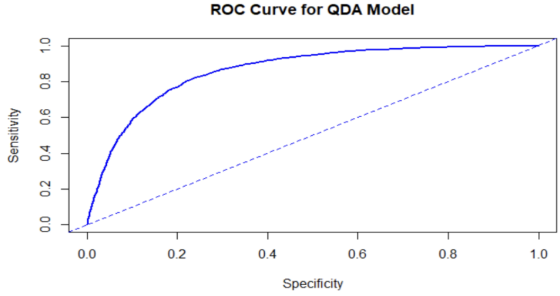

Figure 6: ROC curve for QDA

Based on the result, LDA seems to perform better in both accuracy and AUC compared to QDA. This implies that, based on these evaluation metrics and the dataset used, LDA might be a more suitable model for this particular classification problem.

3.3 KNN

The K-nearest number algorithm [Har18] is a non-parametric, supervised learning classifier that operates based on the principle that in a dataset, similar data points are near each other. Within our dataset, we are classifying our data points based on its neighbors' majority classification within the "K" closest-in-distance neighbors: either "positive" meaning that the review gave positive feedback for a movie or "negative" meaning that the review gave negative feedback for a movie.

We start by training our model using five-fold cross-validation on our training dataset. In doing so, we can select our optimal "K" value of 49 when iterating over odd values of "K" from 1 to 49. We also have a best mean accuracy value of 0.7811, which is the highest average accuracy score obtained from the cross-validation process for different values of "K." Lastly, we have a value for test accuracy of our best model of 0.7842, which indicates the model's predictive performance within real-world scenarios or on data that has not been exposed during the training phase of the model.

Best k:	49
Best Mean Accuracy:	0.7811
Test Accuracy of Best Model:	0.7842

Table 2: Results of KNN Model Performance

We will then create an ROC curve in order to illustrate the performance of a binary classifier model at varying threshold values.

Metric	KNN
Accuracy	0.7842
AUC	0.8659

Figure 7: Performance Metrics for KNN Model

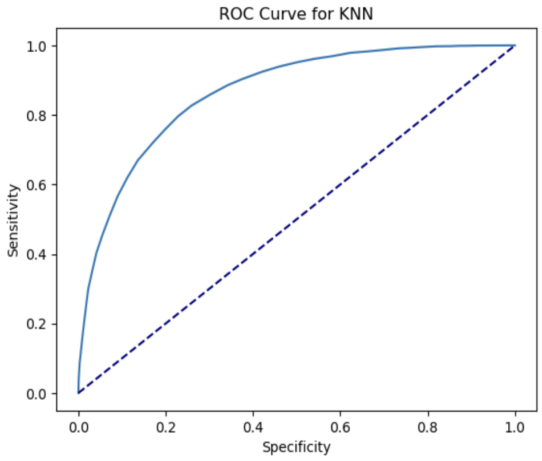

Figure 8: ROC curve for KNN Model

When performing our ROC curve analysis, we see that our area under the curve (AUC) value is 0.8659, which means that the model has a good separability measure and can distinguish between positive and negative classes. We will then generate a confusion matrix (using percentages) to describe the performance of the model. The confusion matrix provided insights into the model's sensitivity given the percentage of True Positives (82.71%) and specificity given the percentage of True Negatives (74.06%), indicating a balanced performance in identifying both positive and negative sentiments.

Given our value for test accuracy of our best model of 0.7842 or 78.42%, this indicates a poorer performance for our KNN model compared to the previous models described above. Furthermore, our AUC value of 0.8659 falls short of the higher AUC values that the other models had. While our model performed well in correctly identifying positive and negative cases, there was a sizable number of times the model made incorrect positive and negative classifications.

3.4 Random Forests

Random forest [Gee23a] is a combination of decision trees. We trained the Random Forest model using the randomForest package in R. We set the number of trees (ntree) to 1000 initially and monitored the out-of-bag (OOB) error every 50 trees to observe the model's performance and adjusted the parameters accordingly. A seed value was set for reproducibility.

Metric	Random Forest
Accuracy	0.8169
AUC	0.8168

Figure 9: Performance Metrics for Random Forest Model

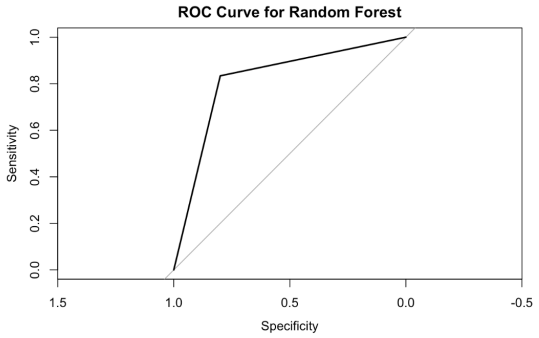

Figure 10: ROC curve for Random Forest Model

The model performed commendably, achieving an accuracy of 81.69%. The confusion matrix provided insights into the model's sensitivity (79.94%) and specificity (83.41%), indicating a balanced performance in identifying both positive and negative sentiments. The balance between sensitivity and specificity underscores the model's efficacy in both identifying true positives and avoiding false positives. The ROC curve analysis yielded an AUC of 0.8168, demonstrating the model's strong capability to distinguish between the two sentiment classes. The relatively high Kappa statistic (0.6337) also indicates a good agreement between the predicted and actual values, above what would be expected by chance.

While the model performed well, there is always room for improvement[You19]. For instance, exploring a wider range of hyperparameters, including the number of trees (ntree), could potentially yield further improvements. Additionally, the dimensionality reduction process (PCA) might have led to the loss of some nuanced information in the text data.

4 Conclusion

For sentiment analysis on IMDB movie reviews, many machine learning models have been examined in this study. The following are the models and the testing accuracies for the following: With an accuracy of 85.72% and an AUC of 0.931, logistic regression demonstrated its strong performance. QDA trailed behind with an accuracy of 79.05% and an AUC of 0.861, but LDA demonstrated its efficacy with an accuracy of 85.49% and an AUC of 0.929. KNN's accuracy of 78.42% indicated a poorer performance. The Random Forest model performed in a balanced manner, exhibiting 81.69% accuracy, 79.94% sensitivity, and 83.41% specificity.

Metric	Logistic Regression	LDA	QDA	KNN	Random Forest
Accuracy	0.8572	0.8549	0.7905	0.7842	0.8169
AUC	0.9306	0.9291	0.8613	0.8659	0.8168

Table 3: Performance Metrics for All Models

The most effective method with the highest level of precision turned out to be logistic regression. Its superiority may be attributed to the way it manages the binary results with resilience and to the skillful application of regularization methods such as Ridge, which reduces the possibility of overfitting. Its performance in our study demonstrates its dependability in tasks involving binary classification, especially in conditions like our datasets.

LDA/QDA came in a close second, both performing effectively. One possible explanation for the success of LDA/QDA is that it performs well in situations when it is reasonable to assume that all classes have a common variance. The reason for this model's somewhat worse performance in comparison to logistic regression might indicate that it is more sensitive to changes in its underlying assumptions, such as equivalent covariance. Though its accuracy was lower, the KNN model provided insightful information. Reduced performance might be associated with the high dimensionality of the data, especially after PCA, which can worsen the 'curse of dimensionality.' The choice of 'k' and the distance measure also have a sensitive impact on KNN's performance, indicating that its effectiveness may change depending on the parametric changes made.

The random forest model showed remarkable flexibility in balancing specificity and sensitivity. This balance suggests that it holds promise in scenarios where eliminating false positives and recognizing actual positives requires an appropriate trade-off. They may be more prone to overfitting than logistic regression, which might account for their marginally worse performance in situations with many features.

Overall, our sentiment analysis of IMDB movie reviews using various models, including logistic regression, LDA/QDA, KNN, and random forest, has shown remarkable trends in each model. The most successful model was logistic regression, which showed its strength in binary classification and resilience when using regularization approaches. This is also followed by the greatest testing accuracy and AUC scores. Nonetheless, it is important to denote that each model has advantages and disadvantages, emphasizing the value of considering different strategies in data-driven initiatives. This underlines that the best model selection depends significantly on the unique features of the dataset and the intended analytical objectives, arguing in favor of complex strategies in the realm of machine learning.

References

[EM17] Thomas W. Edgar and David O. Manz. Logistic regression. *ScienceDirect*, 2017.

[Gee23a] GeeksforGeeks. Decision tree. https://www.geeksforgeeks.org/decision-tree/, 2023.

[Gee23b] GeeksforGeeks. Ml — linear discriminant analysis. https://www.geeksforgeeks.org/ml-linear-discriminant-analysis/, 2023.

[Gee23c] GeeksforGeeks. Understanding tf-idf (term frequency-inverse document frequency), 2023.

[Har18] Onel Harrison. Machine learning basics with the k-nearest neighbors algorithm. https://towardsdatascience.com/machine-learning-basics-with-the-k-nearest-neighbors-algorithm-6a6e71d01761, 2018. Medium, Towards Data Science.

[Nar21] Sarang Narkhede. Understanding auc - roc curve, 2021.

[Set20] Neha Seth. An end-to-end comprehensive guide for pca. https://www.analyticsvidhya.com/blog/2020/12/an-end-to-end-comprehensive-guide-for-pca/, 2020. Analytics Vidhya.

[W+22] Ruiyang Wu et al. Quadratic discriminant analysis by projection. *Journal of Multivariate Analysis*, 2022.

[You19] Tony You. Understanding random forest: How the algorithm works and why it is so effective, 2019.