Abstract

IMDB is an online platform for films, TV series, podcasts, videogames, and more, where users are free to express their opinions. In this study, we attempt to predict whether a given review has a positive or negative sentiment through analyzing its contents. We prepossessed 50,000 IMDB reviews and then performed binary classification on the numerically vectorized reviews using the Logistic Regression, KNN, LDA, QDA, and Random Forest supervised machine learning models. We found that [input results: which model performed best, and what was its accuracy].

Introduction

What are you going to watch tonight? IMDB is an online database of information about TV series, movies, video games, and more. As a leading review site in the film industry since 1990, IMDB has amassed millions of ratings and reviews from users all across the world. The website allows for users to input a rating (with 1 star being the lowest and 10 stars the highest) along with a written review of the content. In our project, we will conduct a sentiment analysis of an IMDB dataset consisting of 50,000 entries of reviews on the IMDB site. Our goal is to use statistical methodology to predict the overall binary sentiment of each review using only the words and phrases in each data entry.

Pre-Processing

~~In order to predict the sentiment of IMDB reviews, we first had to preprocess our dataset. We~~

~~Upon loading in our dataset of IMDB reviews and sentiments, we first conducted some elementary exploratory data analysis in order to understand the size and balance of our dataset. We found that the dataset contained 50,000 reviews and was exactly balanced, containing 25,000 positive reviews and 25,000 negative reviews. Thus, we did not have to employ any techniques to balance our data, since it was already balanced.~~

~~We then examined the lengths of reviews, [add more stuff about review length]~~

~~After performing our elementary exploratory analysis, we then cleaned the data in order to perform higher-level exploratory data analysis and to prepare it for vectorization. In order to do so, we~~

* ~~removed html line breaks~~
* ~~made all lowercase~~
* ~~removed punctuation, special characters, numbers~~
* ~~removed stopwords using tokenization~~

~~After cleaning the textual data, we were then able to perform more EDA:~~

* ~~word clouds~~
  + ~~We observed that positive reviews contained the words “good, great, real, well, love”~~
  + ~~and negative reviews contained the words “bad, little” as well as many words that are also contained in positive reviews~~
* ~~review length~~

~~Because our review data is textual, we had to transform our data from text to numbers. In the data cleaning step, we had already removed the line breaks, punctuation, special characters, numbers, and stop words.~~

~~[we did TF-IDF, explain what TF-IDF stands for and does]~~

~~After performing TF-IDF on our reviews, the vectorized data we got had extremely high dimensionality, with 162,401 features. [High-dimensional data is highly correlated, computationally expensive and hard to work with, etc etc] Thus, we then performed Principal Component Analysis, or PCA, to reduce the dimensionality of our data.~~

~~[explain what PCA does]~~

~~In order to determine how many components to use (i.e., how many features we wanted), we used a Scree plot [explain what a Scree plot does]~~

~~Based on the Scree plot, we saw that there was not a significant change in variance after 50 components, so we finally performed PCA on our vectorized dataset with 50 components.~~

**2 Preprocessing step**

In order to accurately predict the sentiment of the IMDB reviews, we first had to preprocess our dataset. Upon loading in our dataset of IMDB reviews and sentiments, we conducted some elementary, exploratory data analysis to understand the size and balance of our dataset. Out of the 50,000 reviews in the dataset, 25,000 were positive and 25,000 were negative—an exactly balanced split. Thus, we did not have to employ any techniques to balance our data, since it was already balanced. We then examined the lengths of reviews, and compared the review length for the positive and negative reviews to see if there were any significant differences

maybe just summarize below steps

**Data Cleaning**

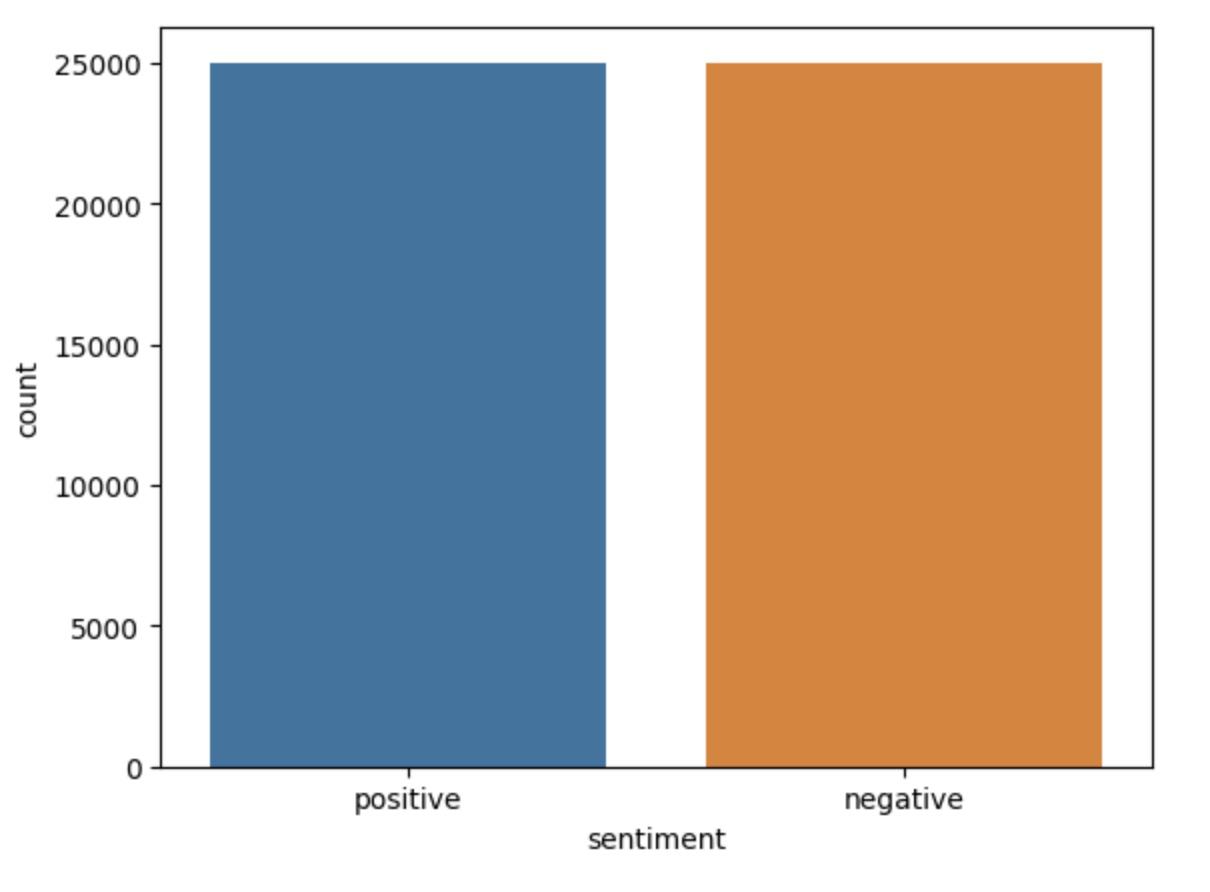
Before conducting any sophisticated analysis, we first cleaned the data to prepare for the later stages of experimentation and analysis. The general process we took to clean the data included:

1. Removing html line breaks.
2. Converting all characters to lowercase.
3. Removing punctuation, special characters, numbers.
4. Removing stop words using tokenization.

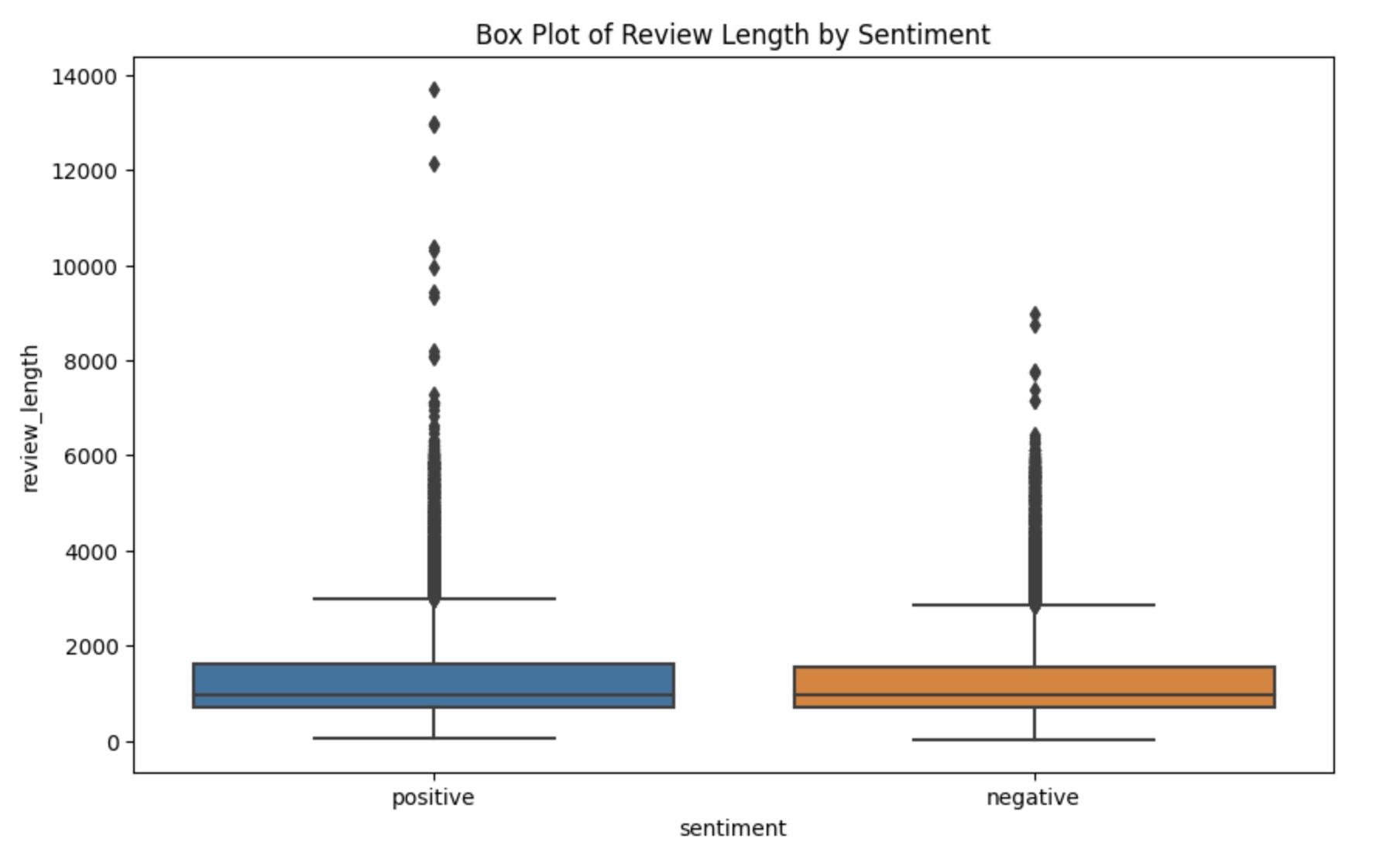
By taking these steps to clean the data, we avoid muddling the analysis with neutral, meaningless words such as "a" or "the" in order to focus on keywords that might better predict sentiment. Now that the data is clean, the exploratory data analysis and visualization can begin.

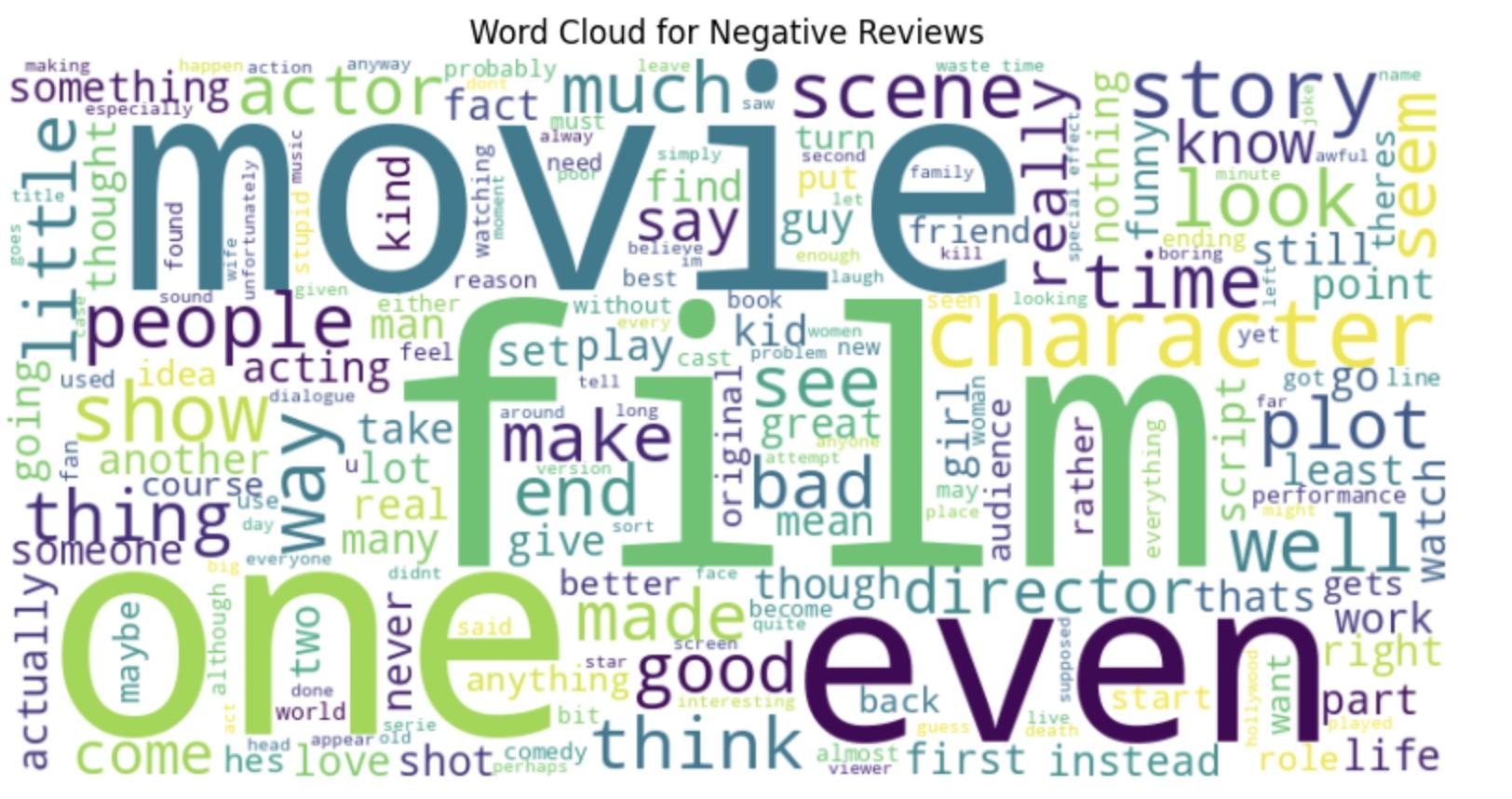
**Descriptive Statistics**

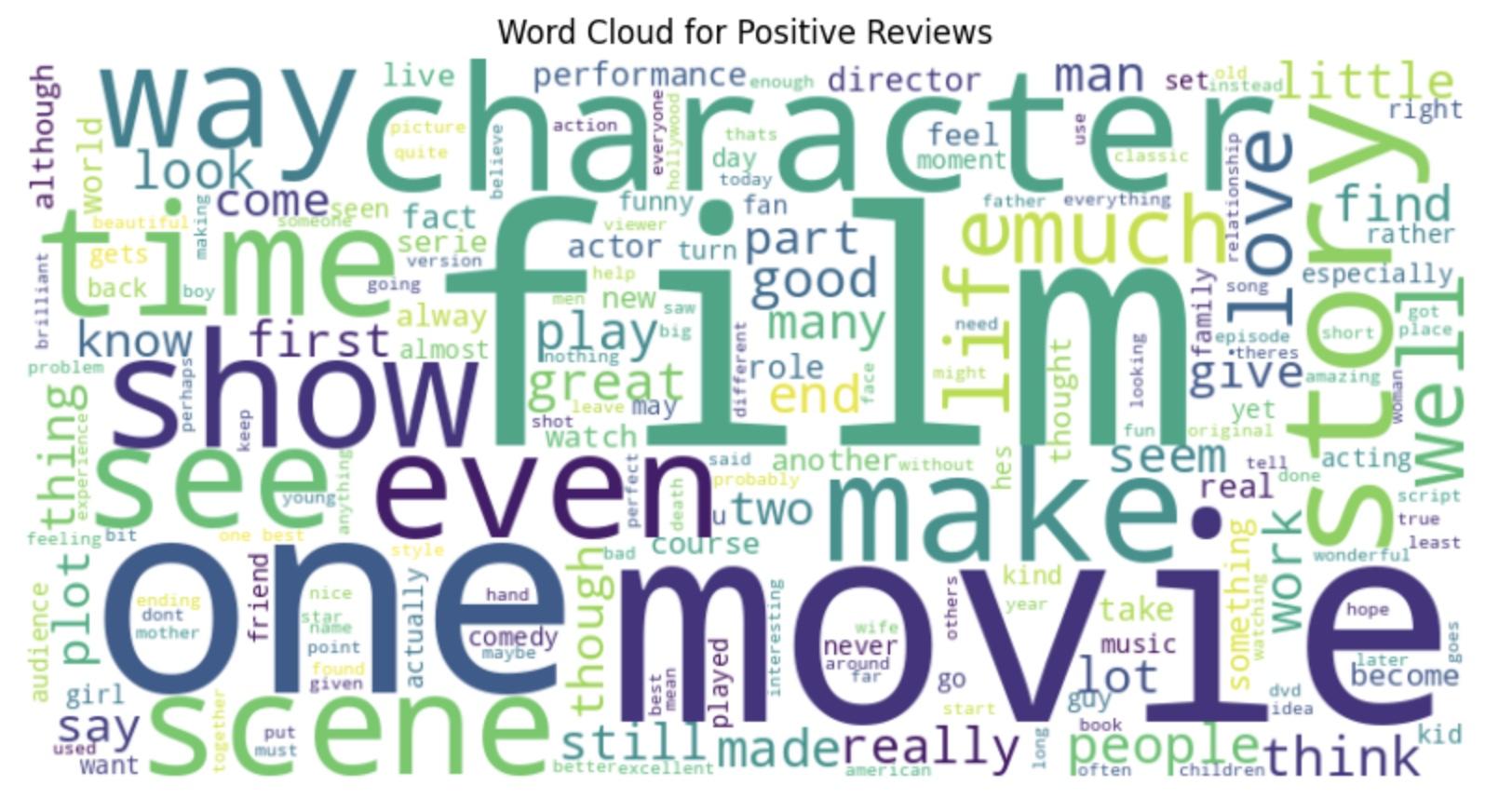
In order to get a better grasp of the dataset, we constructed plots to serve as visual representations of the data.



The figure above displays the sentiment makeup of the dataset. In this particular IMDB dataset, we have a perfect balance of positive and negative reviews.



Evidently, the negative and positive reviews have similar distributions of lengths. However, the positive review distribution contains more extreme outliers on the higher end of the length range.



Comparing the two figures above, we can see that there is not much difference between the two word clouds (aside from a few words such as "bad" versus "good"), illustrating the need for a more in depth look at and analysis of the reviews.

**TF-IDF**

Because our review data is textual, we had to transform our data from text to numbers. In the data cleaning step, we had already removed the line breaks, punctuation, special characters, numbers, and stop words.

TF-IDF (Term Frequency-Inverse Document Frequency):

During data cleaning, we had already removed line breaks, punctuation, special characters, numbers, and stop words. However, to convert our textual review data into a format suitable for analysis, we used TF-IDF, or Term Frequency-Inverse Document Frequency. Here is a breakdown of the method:

Term Frequency (TF):

TF measures the frequency of each term within a review, calculated as the ratio of term occurrences to the total number of terms in the review.

Inverse Document Frequency (IDF):

IDF quantifies the importance of a term across the entire set of reviews, computed as the logarithm of the ratio of the total reviews to those that contain the term.

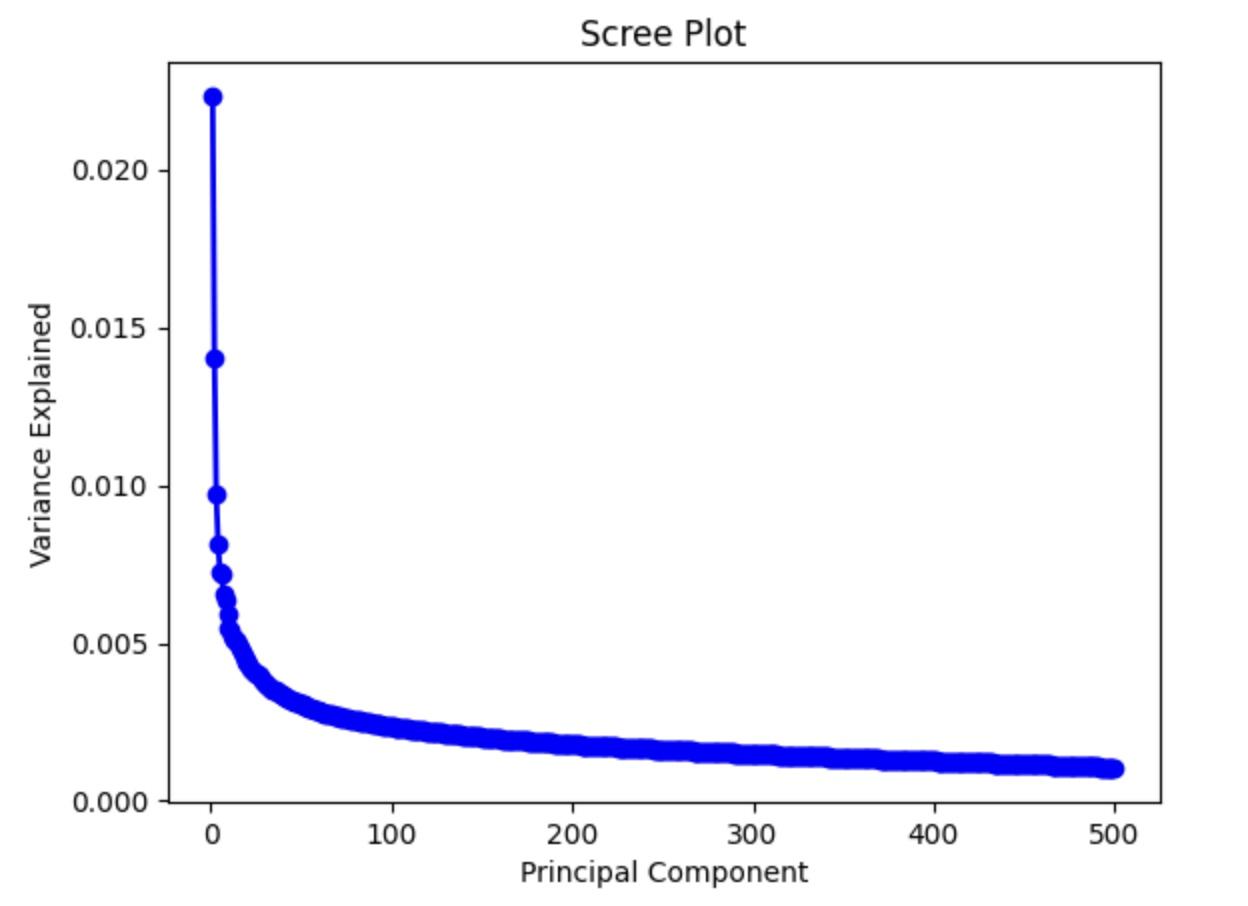
TF-IDF Score:

The TF-IDF score for a term in a document is the product of its TF and IDF scores, creating a numerical representation of the textual data. This results in a high-dimensional vector space, which will be discussed further in the following section.

**Principal Component Analysis**

After performing TF-IDF on our reviews, the resulting vectorized data had extremely high dimensionality at 162,401 total features. Since high-dimensional data is highly correlated, computationally expensive, and difficult to work with, we performed Principal Component Analysis, or PCA, to reduce the dimensionality of our data. By doing so, we can not only reduce the dimensionality of our data while retaining the most important information and minimizing the loss of information. PCA accomplishes this by transforming the original features into a new set of uncorrelated variables called principal components. These components are ordered by the amount of variance they explain in the data, allowing us to keep the most significant information while discarding less critical aspects.

In our case, PCA helps us overcome the above challenges associated with the high dimensionality of the TF-IDF vectorized data. However, we first had to determine how many components to use. We did so using a Scree plot to find the optimal number of components.



Scree plots display the amount of principal components on the x-axis and the corresponding explained variance on the y-axis. Based on the curve above, we saw that there was not a significant change in variance after 50 components, so we finally performed PCA on our vectorized dataset with 50 components.

**3 Experiment**

In our pursuit of identifying the optimal model for prediction, we employed four supervised learning algorithms.

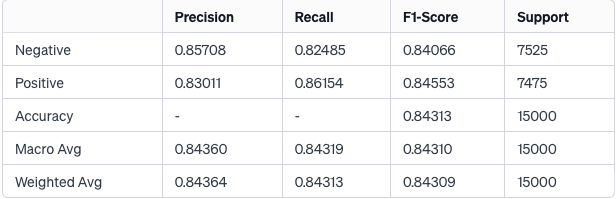
**3.1 Logistic Regression**

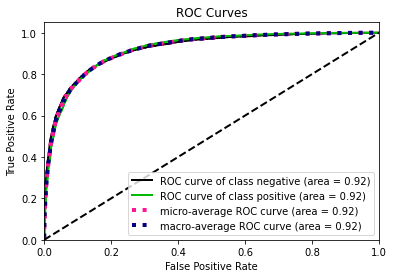
Logistic regression [1] analysis is a statistical technique to evaluate the relationship between various predictor variables (either categorical or continuous) and an outcome which is binary (dichotomous).

Often, predictive models exhibit a close fit to the training data, a phenomenon known as overfitting, wherein the model's performance falters when faced with unseen data. One primary contributor to overfitting is the presence of extreme coefficients. Regularization serves as a technique to mitigate this issue, restraining large coefficients to enhance model generalization and prevent overfitting. The two most prevalent regularization techniques are L1 and L2. For our logistic regression model, we opted for L2 regularization, as L1 introduces sparsity. Given that we had already performed Principal Component Analysis (PCA) to reduce the number of predictors, further variable reduction was deemed unnecessary.

In tuning our model, we focused on a single hyperparameter, namely the value of C, controlling the weight assigned to the regularization penalty. Higher C values lead to a closer fit to the training data, while lower values prioritize the penalty term during coefficient computation. Employing Grid Search Cross Validation, we determined that C=5 produced the optimal choice.

The test set produced the following results:





The model performs equally strong on negative reviews and positive reviews.

**3.2 K-Nearest Neighbors**

**3.3 Linear and Quadratic Discriminant Analysis**

**3.4 Random Forests**

**4 Conclusion and Summary**

**4.1 Results and Analysis**

The first, perhaps most important, step of comparing model efficacy is looking at overall accuracy percentages. The table below displays accuracy for each model we used.

[accuracy table]

Evidently, logistic regression had the highest accuracy, followed closely by LDA. QDA performed the worst. Intuitively, this makes sense since LDA and logistic regression are (computationally) the most similar models out of the methods we used, and their similar accuracy reinforces their high strength in predicting sentiment for our dataset. Additionally, our data is not thought to be normally distributed, so it makes sense that logistic regression outperformed LDA.

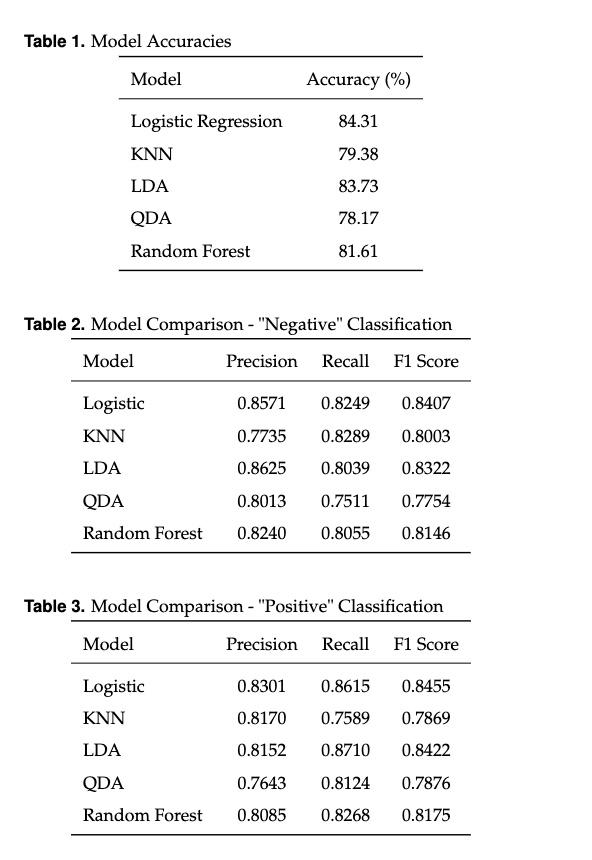
At the bottom of the accuracy list are KNN and QDA at a respective 79% and 78%. Once again, it makes sense that KNN and QDA may have similarly low accuracy scores due to their tendency to overfit. In other words, KNN and QDA may have been too flexible with the training data, capturing noise and specific patterns that don't generalize well to new, unseen data. The lower accuracy scores for KNN and QDA suggest that their decision boundaries might be overly tailored to the training dataset's oddities.

It's worth noting that KNN and QDA could possibly benefit from methods such as regularization, which can reduce variance and help overfitting at the cost of introducing some bias into the model fitting process. However, due to the strong performance of logistic regression and LDA, we decided it was not necessary to attempt to address potential overfitting problems with QDA and KNN.

The random forest method ended up being middle of the road as far as accuracy goes at 81.61%. While it did demonstrate some effectiveness due to its superior accuracy rate over QDA and KNN, its computational cost was considerably higher than logistic regression, making it a worse, less efficient choice for a final model.

To go further in quantifying the performance of our various models, we also generated recall, precision, and F1 scores for each model for the positive and negative classifications.

[insert negative table here]

****

The table above displays the results for the "negative" side of precision, recall and F1 scores. Unsurprisingly, logistic regression and LDA come in as the best performers overall, and especially for the precision column. Interestingly enough, LDA has a slightly higher precision score than logistic regression, meaning that when it predicts negative class, it is slightly more accurate than logistic regression. However, the difference in this case is so small that it is negligible.

Once again, QDA is the worst performing method, being the only method with recall and F1 scores dipping well below the 0.8 level. This means QDA frequently has false positives, and the model struggles to balance precision and recall for the negative class.

[insert positive table here]

The positive table above tells the same tale; logistic regression and LDA are high performers, while QDA and KNN sit at the bottom and random forest takes the middle ground. Once again, this suggests that LDA and logistic regression correctly predict positive sentiments the strong majority of the time, while QDA and KNN are a level below.

Overall, the positive and negative tables simply reinforce the accuracy statistics for each model. The optimal prediction method seems to be logistic regression, closely followed by LDA, while KNN and QDA are generally methods to avoid if aiming for a high accuracy. While random forest had a decent performance, its computational cost makes it a less desirable technique to use for predicting sentiment.

**4.2 Final Comments and Reflection**

Across the board, our models were very effective in predicting the sentiment of IMDB reviews. Their accuracies were significantly above 50% (which would be the approximate percentage corresponding to randomly guessing "positive" or "negative"). Although our models were robust, there are a couple concerns that might have affected our results negatively.

*Correlated reviews:* The sentiments for a singular movie (or film series) likely has some correlation with each other. Because the reviews are publicly available, any potential, future reviewer may be biased by the reviews currently on the site. For instance, a movie with extremely negative reviews may cause future reviewers to have a more negative sentiment than they otherwise would have. Although this introduces some correlation between data points and we did not take it into account, the effect is likely minimal and our results are not threatened by it.

*Additional models:* We tested a total of five methods(KNN, QDA, LDA, logistic regression, and random forest) but there are certainly more available techniques out there for prediction. With these models alone, we managed to achieve a high accuracy rating and did not feel that it is necessary to use even more models, but it is possible that there is a better technique out there.

*Model complexity:* The models we used are relatively standard and void of any highly sophisticated tools. There are of course many more options that may aid prediction accuracy or improve the performance of a specific model. As mentioned before, QDA's low performance could have been partially due to overfitting and might have benefitted from a technique like regularization. However, for our needs, logistic regression and LDA were more than sufficient and we did not feel the need to introduce further methodology.

**References**

**[1] Ranganathan P, Pramesh CS, Aggarwal R. Common pitfalls in statistical analysis: Logistic regression. Perspect Clin Res. 2017 Jul-Sep;8(3):148-151. doi: 10.4103/picr.PICR\_87\_17. PMID: 28828311; PMCID: PMC5543767.**