Stats 101C Final Project

Anthony Rio [redacted] Angelique Rubio [redacted] Eric Liu [redacted]
group members UID redacted for privacy
Jason Vo [redacted] William Sun [redacted]

Abstract

IMDb is a website primarily dedicated for films and television shows where users can leave reviews for such media. We will analyze the correlation between the written review left by users and whether the review was considered positive/negative in order to predict the review's sentiment utilizing opinion lexicon paired with PCA, TFIDF frequency, KNN, logistic regression, and other model fitting techniques.

1 Introduction

It's finally Friday night after an exhausting week and you're searching for a movie to watch, but have no idea which one to select? IMDB, a useful and reputable resource for the average movie-enjoyer to professional film critics, is a public website containing reviews and ratings for various forms of media, and in particular movies. An IMDB review is of the structure:

1. A numerical rating on a 1-10 scale, where 1 denotes an awful movie and 10 denotes a award winner.
2. A descriptive portion, where the user justifies the numerical rating with words.

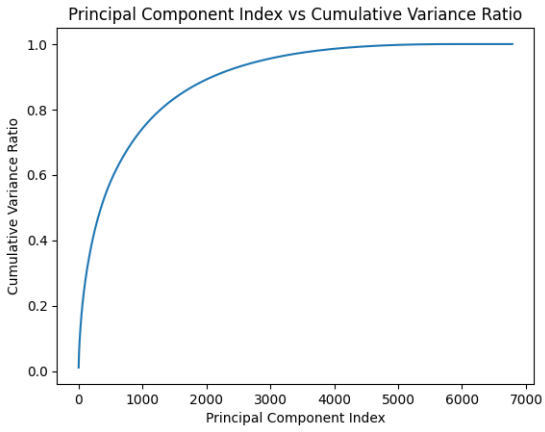
For our project, we are given 50,000 movie reviews, half of which are to be used for training and the other half for testing. The reviews range from positive to negative sentiments. This brings us to our goal: to use statistical methods to predict the numerical rating of any given IMDB review using the descriptive portion, via underlying connotations, emotions, word-selection, etc.

2 Data Processing

The raw dataset is a csv file of text, thus presenting the challenge of converting text data into numerical data. To do so, we chose to eliminate the "junk", or elements in the text that are independent of sentiment including, personal pronouns (he, she, I, ...), determiners (a, an, ...), coordinating conjunctions (for, an, nor, but, ...), and/or prepositions (in, under, towards, ...). We also transformed adverbs into adjectives (ie. happily to happy) in order to achieve uniformity in the text.

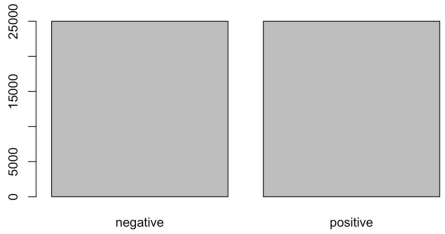
To associate sentiment with words, we opted for the following pre-defined opinion lexicon: https://www.kaggle.com/datasets/nltkdata/opinion-lexicon, which classifies around 6800 English words as either positive or negative.

Textual data is very high-dimensional and therefore requires some form of dimension reduction before we can manipulate it for further analysis. We chose to use principal component analysis (PCA), a dimension reduction technique that transforms high-dimensional data into a new coordinate system where the principal components capture the maximum variance in the data. We chose to utilize 1300 principal components, accounting for 80% of the variance in our data. In doing so, we can retain as much relevant information as possible while still simplifying our data and making it easier to manipulate. A graph of the principal components by the cumulative proportion of variance is shown below:



3 Descriptive Statistics

To gain a better understanding of the data, let's take a look at the frequency of our binary response variable "sentiment", which can either be "positive" or "negative". In the barplot seen below, it is shown that there is a complete even split between "positive" and "negative" sentiment reviews, as there are 25,000 of each.



Further, to learn more about our initial singular feature, "review", let's take a look at a word cloud of all words within each review. Some of the more commonly used words include good, like, and bad; many these can be found in the positive and negative word lexicons.

Because the data set is incredibly large with 50,000 reviews, a random sample of 10,000 reviews was taken and was cleaned to display this word cloud. While there are many words in green indicating that these words are relatively insignificant, the bigger words are more indicative of the trend in the data. For instance, "good" is more commonly found in the data than "bad" which is telling of the results that we will report in the following sections.

4 Balancing the Dataset

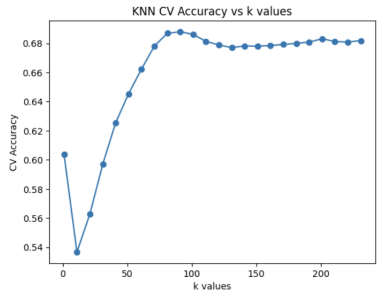
As previously mentioned, the dataset had an equal number of "positive" and "negative" sentiment reviews. Additionally, we mostly used k-fold cross-validation, which will be explained later on. Due to this fact, we didn't make a singular training and testing data split, because it is not necessary for cross-validation, and so the singular, unsplit dataset is balanced. Furthermore, because of this, we did not use Synthetic Minority Oversampling Technique.

5 KNN

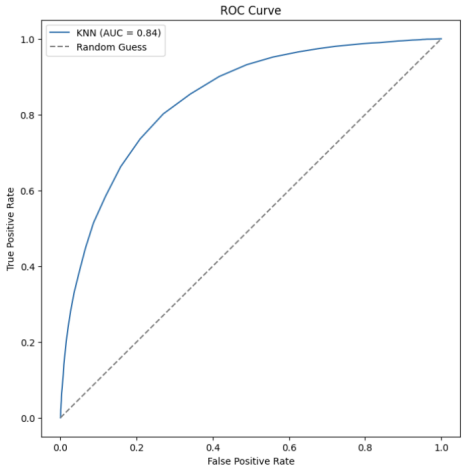
K nearest neighbors (KNN) is a non-parametric method in supervised learning that can handle classification or regression problems by considering the

closest data points. Thus, KNN is very flexible and there are no assumptions that need to hold like there are when using LDA for example. This benefits us because in such a large dimension-space with 1300 features, we do not need to check time-consuming assumptions like features being multivariate normal. To further explain KNN, the "K" nearest neighbors are averaged in a majority rules style, to determine a prediction. Because of this, K must be odd. So for example if K = 9, and for a given point, for the closest 9 points, 5 have "positive" sentiment and 4 have "negative" sentiment, we would follow the majority rule and classify the sentiment for this new point as "positive". Within the process for using KNN, we first decided to use 5-fold cross validation on our training set to evaluate our model and its performance. After doing this, we then tuned the K value, using values in increments of 10, ranging from 1 to 231. These results indicated that a K close to 91 would be ideal, and after more tuning, we found that the KNN model performed best when K = 89.

KNN



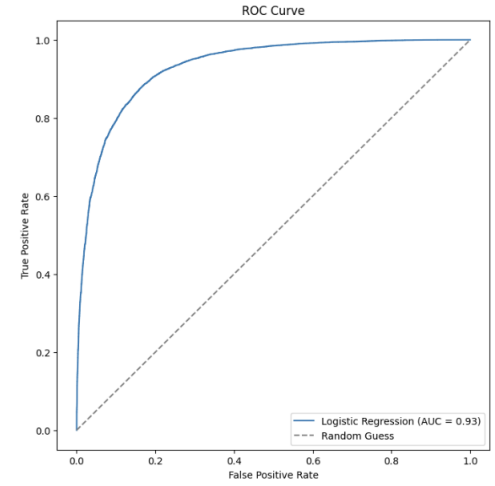
However, the accuracy for this KNN model still was not very high at 0.688, so overall the accuracy of KNN topped out below 0.7. We can likely attribute this to the curse of dimensionality. This arises when working with high-dimensional data, and can cause low accuracy for models, as there is less sparsity in the data and an increased volume of the feature space as the number of dimensions increases. Lastly, to create the visualizations, using our 50/50 train-test split, we ran the model on our test data to create a ROC curve and confusion matrix to provide more color on the overall performance of the method.

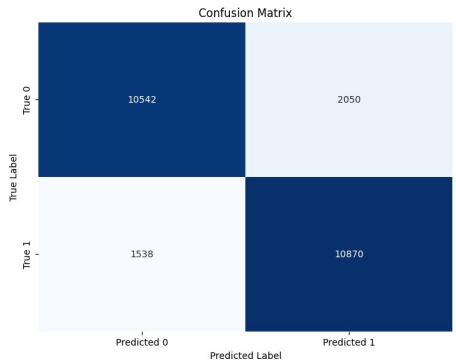


6 Logistic Regression

Logistic regression is a method that estimates the conditional probability of a binary event occurring given some data. Thus, for predicting the binary "positive" and "negative" sentiment of a review, logistic regression is a natural fit. Predictive models like logistic regression will fit better on the data it is trained on rather than data it is tested on, a phenomenon referred to as over-fitting (when the model performs worse on unseen testing data). Regularization is a way to reduce over-fitting, and L1 and L2 regularization are very popular techniques. However, for this model, we did not use regularization thus we did not need to tune any hyperparameters, but it is something that could be tried in the future. After performing PCA and reducing the number of predictors from over 6700 to 1300, there were already a lot of features. Due to this fact, we did not do any feature engineering to increase the number of predictors, because although it could increase the overall model performance, it would only make the model more complex and computationally intensive. Given the longer run time we experience from some of the code to execute feature manipulation, PCA, and modeling, we chose not to do any feature engineering. When calculating the accuracy and tuning the value of K, we used 5-fold cross validation with our train-

ing set. Lastly, for the visualizations displayed, based on the random train-test split previously mentioned, with 50 percent in each subset, we used our test set to produce a ROC curve and confusion matrix, to in turn provide more information on the general method and model effectiveness.



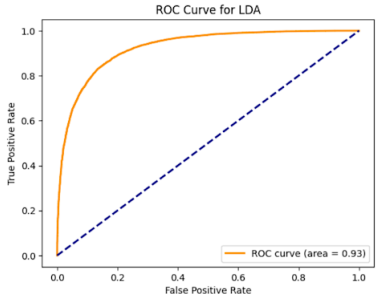


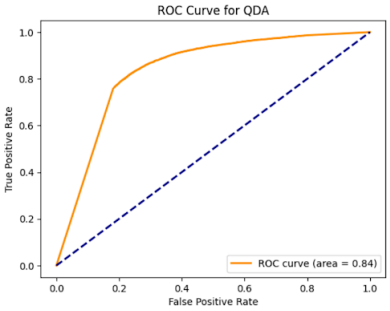

```
Logistic Regression Test Accuracy: 0.85648
```

Overall, transitional logistic regression performed better than KNN at .856.

7 LDA, QDA

Linear Discriminant Analysis (LDA) and Quadratic Discriminant Analysis (QDA) are both generative models. Since we are using these models in the context of classification, generative models are intended to replicate the probability distribution of the features within each class in order to classify new observations. While LDA is a special case of QDA, the two differ in various ways. Firstly, LDA assumes all classes share a common covariance matrix, while QDA does not. In that way, QDA allows for more flexible decision boundaries and is better suited for data that violates this assumption of sharing covariance matrices. Both models assume each class contains features that follow a multivariate normal distribution.

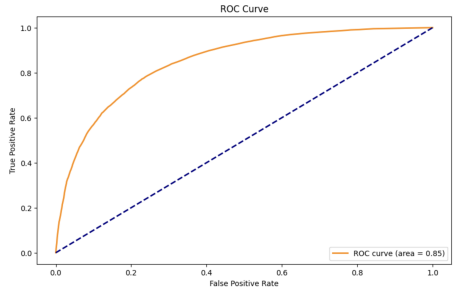




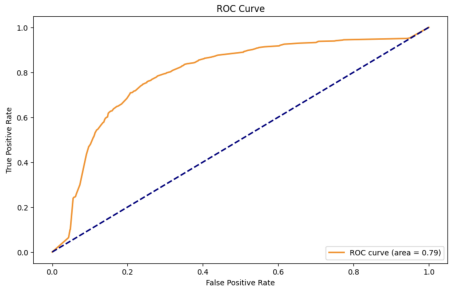
We observe that in the case of our data, LDA has an accuracy of .846 and QDA has an accuracy of .796, illustrating that LDA outperforms QDA.

8 Decision Tree

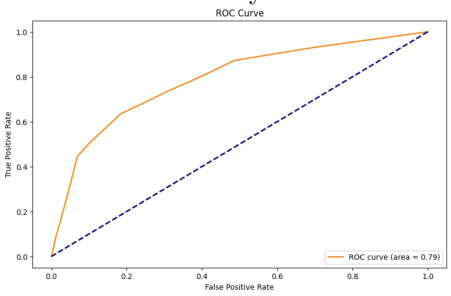
A Decision Tree is a supervised machine learning algorithm, consisting of a tree-like structure with nodes representing decisions or test conditions, edges representing the outcomes of these decisions, and leaves representing the final predictions or target values. In the context of this data, a decision tree of height 5 results in the greatest accuracy of .772 and the best ROC curve.



In higher-height trees we tested (such as 10), we observed a drop-off in accuracy and a lesser ROC curve, possibly due to the downward nature of decision trees, resulting in over-fitting.

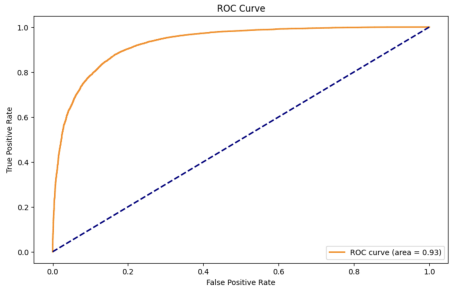


On the other hand, lower-height trees (such as 2) also led to lower accuracy and a lesser ROC curve.



9 SVM

Support Vector Machine (SVM) is also supervised machine learning algorithm, with the objective to derive the optimal hyperplane to desperate classes in the feature space. SVM is effective in high-dimensional spaces, robust in handling outliers, and versatile with the ability to handle linear and non-linear relationships through the kernel trick–this involves transforming the input features into a higher-dimensional space, making it possible to find a linear separation in that space.



The accuracy for this SVM model with regularlization was .856.

10 Conclusions

All of our models performed above .500–most notably Logistic Regression, SVM, and LDA being the top performers, at .857, .855, and .846 respectively. Overall, we could be content with the current results, however we point to possible areas that could be improved upon in deeper analyses:

1. Explore other methods to convert text data into numerical data: while words in isolation could carry their true sentiment, bigram (and larger) analysis could provide more context to certain words and display their true, underlying sentiment. For instance, with the current lexicon dictionary, the phrase "shockingly innovative" would register both 1 positive and negative word, even though the phrase as a whole would be deemed positive by most people. By doing so, there would likely be a further increase in accuracy for all models.

2. Modifications to current models: in order to address and minimize overfitting in Logistic Regression, in the future, we could look into L1 (Lasso) and L2 (Ridge) Regularization. And in the context of SVM, we could choose other methods such as polynomial kernel and radial.