**Lab 2 Movie Reviews - DATA 3444 - Jessie Reynolds - Feb 7 2025**

## Background / Introduction

Our original IMDb dataset consisted of 50,000 examples with 3 features, including the movie number, the text review and a label, 0 representing a negative or 1 representing a positive review. This original dataset was quite large, so we will cut this down by taking a random sample of 2,000 examples.

## Data Preparation

To begin data preparation, we will convert labels 0 and 1 to negative and positive factors. Before standardizing our text, we will need to create a corpus. A corpus is a collection of texts and it will allow us to transform just the text review feature. From here, we will be standardizing our text and removing unnecessary words, numbers and punctuation that will not be beneficial when determining if a review is positive or negative. First, we will begin by converting text to all lowercase characters. This just removes all capital letters from the text. Next we will remove numbers from the text. Since we are trying to predict if a review is positive or negative, numbers will not be useful to us. Then we will remove filler words. Words like and, but, to are considered stop words because they occur so often and are not an indication of a review being positive or negative. At this point, the text reviews are no longer coherent sentences as filler words have been removed, but they are much simpler and consist of meaningful words. After this we will remove all punctuation. Periods, commas, exclamation points, etc., are removed from our text data because we are more focused on the words being used in the reviews. Finally, we will remove all blank spaces in the text because again, these will not be useful in determining a positive or negative review. At this point we can compare our cleaned dataset to the original and see that although these reviews can not be read coherently, only the important parts of the reviews are left. All of our text reviews are standardized and ready to be converted into a matrix.

The next step in our data preparation is to tokenize our text reviews into a matrix of individual words. Each cell in the matrix will tell us a count of the number of times the word represented by each column appears in the message represented by the row. Our document term matrix will be a sparse matrix because at this point our dataset has 20,495 words in total. Most of these words are not used very often, making the matrix sparse.

## Training and Test Datasets

For this lab we will be using 75% for our training dataset and 25% for our test dataset. We will begin by taking a random sample of 1,500 examples, or 75% of our 2,000 total. Thus, the test dataset will be made up of the remaining 500 examples. In our total dataset, 50% of the examples were classified as positive and the other 50% were classified as negative. To make sure our training and test datasets represent our whole dataset accurately, we need to check proportions of positive and negative reviews in these as well. My training dataset consists of 49.47% negative reviews and 50.53% positive reviews. My test dataset consists of 51.6% negative reviews and 48.4% positive reviews. Though these percentages seemed to be flipped for my training and test dataset, because of my seed number, they are close enough to the original to continue on with the algorithm.

## Naive Bayes Algorithm

Before running the algorithm, there is one final part of data preparation that we must tackle. To reduce our number of features in the matrix we will be filtering out words that aren't occurring frequently, in this case less than 5 times. At this point, our matrix is all numerical, so we must convert it to categorical variables so the algorithm can run smoothly. We can do so by creating a function that will convert counts higher than 0 to "Yes" and the rest to "No" and applying this function to both the training and test datasets. Finally, we can build the classifier using our training dataset and see how well our model predicted positive and negative reviews accurately. We will first run the algorithm without a Laplace estimator.

| | actual | | |
|---|---|---|---|
| predicted | negative | positive | Row Total |
| negative | 224 | 62 | 286 |
| | 0.448 | 0.124 | |
| positive | 34 | 180 | 214 |
| | 0.068 | 0.360 | |
| Column Total | 258 | 242 | 500 |

***Figure 1: Cross Table of predictions without Laplace estimator***

Based on ***Figure 1***, we can see that there were 62 reviews that were actually positive, but were predicted to be negative. There were also 34 reviews that were actually negative, but were predicted to be positive. Thus 96/500 or 19.2% of reviews were predicted incorrectly. Along with class type, we can also look at probabilities of movie reviews being predicted as positive or negative.

**Improving Model**

To improve our original model, we can use a Laplace estimator when training our model. In this lab we will be using a Laplace estimator of 1 to see if our results can improve.

| | actual | | |
|---|---|---|---|
| predicted | negative | positive | Row Total |
| negative | 227 | 62 | 289 |
| | 0.454 | 0.124 | |
| positive | 31 | 180 | 211 |
| | 0.062 | 0.360 | |
| Column Total | 258 | 242 | 500 |

***Figure 2: Cross Table of predictions with laplace estimator of 1***

Following a similar process as before, we ran a Cross Table. In ***Figure 2***, we can see that there were again 62 reviews that were actually positive, but were predicted to be negative. There were now 31 reviews that were actually negative, but were predicted to be positive. Thus 93/500 or 18.6% of reviews were predicted incorrectly when we changed the Laplace estimator to 1. Though there were only 3 less reviews predicted incorrectly, the Laplace estimator did improve our results.

Another way we wanted to try improving our model was by changing our threshold on frequent words. I wanted to see how our model would improve if the threshold was both lowered and raised. I changed the threshold to 3 and ran the model, then repeated this again with a threshold of 10.

| | actual | | |
|---|---|---|---|
| predicted | negative | positive | Row Total |
| negative | 221 | 63 | 284 |
| | 0.442 | 0.126 | |
| positive | 37 | 179 | 216 |
| | 0.074 | 0.358 | |
| Column Total | 258 | 242 | 500 |

***Figure 3: Cross Table of predictions with frequency threshold of 3***

In ***Figure 3***, we can see that there were 63 reviews that were actually positive, but were predicted to be negative. There were now 37 reviews that were actually negative, but were predicted to be positive. Thus 100/500 or 20% of reviews were predicted incorrectly when we changed the frequency threshold to 3. Based on our original results of 19.2% predicted incorrectly, changing the threshold 3 did not improve our model.

| | actual | | |
|---|---|---|---|
| predicted | negative | positive | Row Total |
| negative | 218 | 53 | 271 |
| | 0.436 | 0.106 | |
| positive | 40 | 189 | 229 |
| | 0.080 | 0.378 | |
| Column Total | 258 | 242 | 500 |

***Figure 4: Cross Table of predictions with frequency threshold of 10***

In ***Figure 4***, we can see that there were 53 reviews that were actually positive, but were predicted to be negative. There were now 40 reviews that were actually negative, but were predicted to be positive. Thus 93/500 or 18.6% of reviews were predicted incorrectly when we changed the frequency threshold to 10. Based on our original results of 19.2% predicted incorrectly, changing the threshold 10 did improve our model slightly.