**Project 3 Report**
**11/29/2022**
**Sarah Yang (sarahxy2), Ryan Berry (rpberry2)**

## Introduction

We were provided with a data set of 50,000 IMDB reviews where we were assigned with creating a binary classification to classify the review as either positive or negative. This was an exercise on text analysis and processing as we had to use various techniques to our reviews to create our models with. We created 5 training/test splits to process our model with. The metric used to measure the performance of the model is AUC and the goal was to produce an AUC greater or equal to .96 over the five test data sets, which we were successful in doing.

## Data Processing

The main data set, pulled from a Kaggle competition contains 50,000 rows with the columns id, sentiment (0=negative, 1=positive), score (out of 10 assigned by the reviewer), and review (text in string format of the review). Because we needed train/test splits, we ran a script to generate these from the alldata.tsv, and indexes in given tsv files in train.tsv and test.tsv for splitting between train and test respectively.

We used various text analysis steps specifically on the reviews column to help us develop our model. Some of the steps we did were removing stop words and other irrelevant characters from the reviews, as these words aren't relevant to deciding sentiment, tokenizing to reduce sentences in smaller chunks, converting words to lowercase, and pruning to further trim our vocabulary. After vectorizing, we construct a document-term matrix which acts as the explanatory variable for our model. To further reduce our vocabulary, we used a ridge regression to pick out a vocabulary less than 1000, to avoid overfitting and increase the interpretability of our model. We use this vocabulary for our vectorization of reviews before going into building our model. More information on the vocabulary can be found in our vocabulary html file.

## Model Training

We used glmnet to run cross-validation (CV) using lasso with logistic regression and also trained against a lasso logistic regression model. Both the CV and training steps use a document-term matrix as the explanatory variable, sentiment score as the response variable, an alpha value of 1 (lasso), family set to 'binomial', and area under curve as the measure.

In addition to those parameters, our CV step utilizes 10-fold cross-validation. The output of the CV step provides the best lambda value, which is also used as a parameter for the training step. This lambda value was key in establishing good accuracy.

We explored different combinations of alpha values, e.g. CV using ridge regression and train using lasso (and vice versa). The best performance came with both CV and train using lasso.

**Model Interpretation**

Our features include our reviews in terms of our vocabulary in vectorized format, and the frequency of those words. Our classification model learns what frequency amount of a certain word leads to positive and negative sentiments. If there are significantly more positive words than negative words that are in a review, then the sentiment will be positive, and vice versa.

As an example, the review with id=598 (included below) gives a score of 7 which indicates a positive sentiment of 1, and our model incorrectly assesses this review as negative, giving a sentiment score of 0.37. We can see how our model struggled with this review due to its verbiage. Phrases like “bad movie” and “not a great comedy” in isolation can skew the review negatively without proper context.

*Review 598: “Now, Throw Momma from the Train was **not a great comedy**, but it is a load of fun and makes you laugh… Some of the scenes seemed to **not fit** in for me, but this didn't make it a **bad movie**.<br /><br />For what it is, a wacky comedy, it pulls it off well and should be seen once just to say you saw it.”*

**Model Limitations**

Our model does not take into consideration phrases longer than 4 word, which can be an issue if a lot of the 4 word phrases are neutral rather than positive or negative sentiments. This can make our model predict well for the type of sentences we have in our data set, but not other types of sentences that can come from other data sources. Our model is also heavily influenced by the top frequent words, for those that lead reviews to be classified as positive sentiment, as well as those that do so for negative sentiment.

**Model Performance**

The model performance for the data splits are shown here. Computer System used for processing scores: MacBook Pro 2.6 GHz 4-core Intel Core i7 with 16 GB of RAM.

| Split | AUC | Processing Time (s) |
|---|---|---|
| 1 | 0.9664 | 49.9 |
| 2 | 0.9667 | 51.9 |
| 3 | 0.9664 | 51.1 |
| 4 | 0.9672 | 51.4 |
| 5 | 0.9658 | 51 |

AUC, the metric that we used to measure the performance of our model, is the area under the ROC curve, which shows the ability for the model to distinguish between positive and negative. An AUC value that is close to 1 means it is good at distinguishing between classes. In our case, we are obtaining AUC values of .96 or better across all our folds, which shows that our model is able to distinguish between positive and negative sentiment of reviews in our test set very well.

**Future Steps**

Further reducing our number of words, while keeping words that would have importance to us, would have been the first thing we would have continued doing to help improve our model. With a rich data set of reviews, we have a lot of room to pick out the words that would be most likely to influence the classification between positive and negative sentiment. We could also use TF-IDF to weight words in our documents, as it takes into account the frequency of words across the set of documents and penalizes contributions from words that appear in more documents. This may have enabled us to improve our accuracy of prediction of sentiments of our reviews. Adding more helpful features may also help our accuracy, as more information about the review will help us classify between positive and negative sentiment beyond just the words and frequencies of those words. One such feature that could be beneficial to add would be average sentiment of previous reviews that the author has created.

**Conclusion**

Our vocabulary selection, preprocessing and model creation steps proved successful with an average AUC of 0.9665. And while there are other models that we could have trained against, using a basic logistic regression approach assisted with the model's overall interpretability. We can see how words more often associated with negative reviews negatively impact the sentiment score. In the future, we are interested in how we could introduce new features or generalize the model so that it is not limited to movie reviews or a 0-1 classification metric.

**Contributions**

Sarah Yang - Did preprocessing, developed the model from the vocab and wrote the report.
Ryan Berry - Created the vocabulary and wrote the report.