**CS598: Practical Statistical Learning**

**Project 3 Report: Movie sentiment analysis**

Javier Huamani ([huamani2@illinois.edu](mailto:huamani2@illinois.edu))

Sudha Natarajan ([Sudha2@illinois.edu](mailto:Sudha2@illinois.edu))

# **Contributions**

Both teammates were equally involved in the exploration, data analysis, modeling, predicting and report creation of the movie sentiment analysis data; we had check-in meetings to share findings and updates on the progress of the model creation and the report creation steps.

# **Objective**

We were provided with a dataset consisting of 50,000 IMDB movie reviews, where each review is labelled as positive or negative. In this project we build a binary classification model to predict the sentiment of a movie review and label it positive or negative. The final goal is to predict the sentiment of a review with vocabulary size less than equal to 1000. Using AUC as the evaluation metric, our performance target is to produce AUC equal to or bigger than 0.96 over all the five splits of test data.

# **Data Assessment and analysis**

The initial dataset provided to us was alldata.tsv, which has 50,000 rows and 4 columns, and each row contains a movie review. The following are the 4 columns:

* Col 1: “id” the identification number.
* Col 2: “sentiment”, 0 = negative and 1 = positive.
* Col 3: “score”, the 10-point score assigned by the reviewer.
  + Scores 1-4 correspond to negative sentiment.
  + Scores 7-10 correspond to positive sentiment.
* Col 4: “review”

The following were the primary cleansing exercises that were undertaken for all Train/Test splits:

* Removal of html tags from “review”
* Lowercase “review”
* Remove “score”

An extensive process using t-tests and lasso regularization was utilized to reduce the overall vocabulary size, and therefore the DTM sizes. An html markdown file MyVocabGenerator.html was submitted to highlight the overall process.

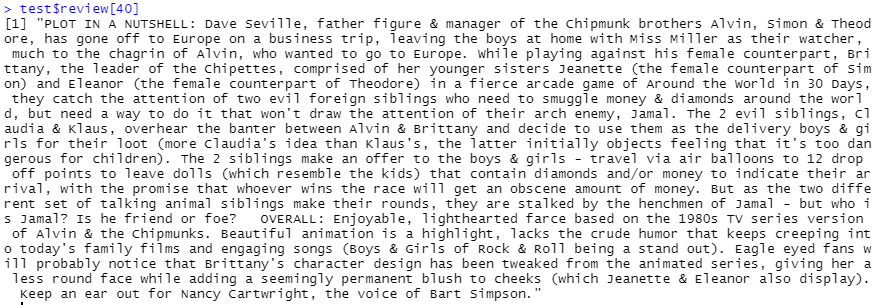
# **Approach**

We followed the various approaches suggested by Dr. Liang. Our inputs included the comprehensive vocabulary of 981 terms (1-grams to 4-grams) and the training and test data per split. We created Document Term Matrices (DTMs) for each train and test split using the comprehensive vocabulary. The training DTM was used to train a binary logistic regression model with ridge regression (alpha of 0). We decided to use logistic regression because its effective with binary classification problems, its simplicity leads to improved performance and the size of our dataset (more entries than features) work best with it. We used cross validation to estimate an optimal lambda value which maximized the AUC. However, initially we were unable to meet the threshold of 0.96 AUC for some splits. We decided to use elastic net with an alpha term of 0.1 which preferences an L2 over L1 penalty. Using cross validation again, we were able to output lambdas which surpassed the threshold 0.96 AUC for all 5 splits.

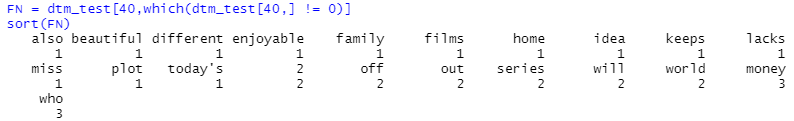
# **Validation**

Despite reaching the threshold AUC for all 5 splits, it is important to study a subset of incorrect predictions in order to work through possible improvements for the task at hand. Two incorrect predictions, a False Negative Sentiment and False Positive Sentiment, were gathered for discussion.

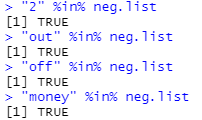
# **False Negative Sentiment**



The review above shows one example of a False Negative Sentiment review for the 1st split. The reviewer in this case used ~75% of their review stating a summary of the movie plot prior to stating their opinion of the actual movie. The following is a subset of the test DTM which shows the words with non-zero counts for the False Negative Sentiment review:

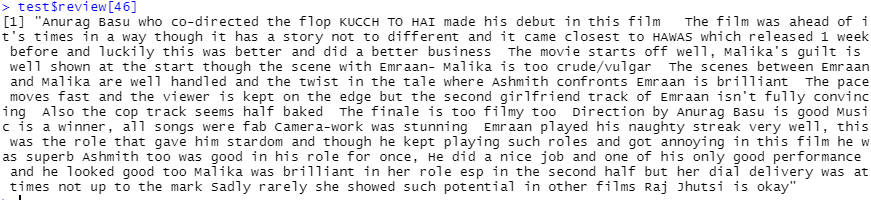


Words such as beautiful and enjoyable are positive words which have the lowest counts, possibly due to the fact that they were in the shorter opinion section of the review. However, the following words in the DTM were all found within the negative terms list:

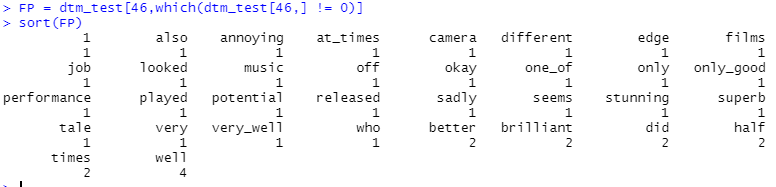


The words “2”, “off” and “money” were all found within the plot summary. Numbers have been found within both the negative and positive words list as a means for reviewers to indicate their own numeric rating of a movie. However, in this case the “2” was an arbitrary number used within the summary. It is possible that without the initial irrelevant summary section, the binary classifier would’ve classified the review as positive instead of negative

# **False Positive Sentiment**



This particular review was confusing because it gave praise to the actors, musical score and other minor aspects of the movie. However, it also criticized the actual score and the ground truth states that overall this is a negative sentiment review. The following is a subset of the test DTM which shows the words with non-zero counts for the False Positive Sentiment review:



Positive words like “well”, “better”, “brilliant”, “stunning”, etc. are shown to have the highest counts in the DTM. The reviewer used more impactful words more often when discussing the praise in the review. It is understandable that the classifier would’ve incorrectly assigned the review with a Positive sentiment due to the ambiguity of the review.

# **Results**

The following table shows the AUC we obtained for each of the 5 splits:

|  |  |
| --- | --- |
| **Split** | **AUC** |
| Split 1 | 0.9629556 |
| Split 2 | 0.9623818 |
| Split 3 | 0.9617547 |
| Split 4 | 0.9624445 |
| Split 5 | 0.9617127 |

The overall average AUC for all 5 splits is: **0.96224986**

The total running time for all 5 splits on Windows 10 PC, 16 GB Ram, Intel Core i5-8350U CPU system is: **7.15 minutes.** The average running time for each split is: **1.43 minutes.**

# **Conclusion**

We got to work in a real-life use case with a practical movie reviews dataset to predict the sentiment of movie reviews. Using Professor’s code with the various approaches helped us walk through and understand the different steps, trial and error process and tuning to the desired outcome. We can envision various real life use cases that this process might apply to and are confident about the approach and the process to follow.

Validating our results by taking a deep dive at the data helped us understand some possible reasons why our classifier made some mistakes despite surpassing the threshold AUC. The deep diver made it clear that ambiguity and irrelevant information can have a detrimental effect on the performance of the classifier.

# **Acknowledgement**

* Professor Dr. Liang’s code and approaches from various posts on Campuswire
* Sample project reports posted on Campuswire by Dr. Liang.