PROJECT 3- Movie Review Sentiment Analysis

CS598- Practical Statistical Learning

Submitted by – Preethal Joseph – pjoseph3

Rajesh Maheswaran – rajeshm2

Tess Hagen – tessah2

1.INTRODUCTION

The objective of this project was to predict the sentiment of IMDB movie reviews which was provided as a dataset with 50,000 rows and 4 columns. The 4 columns consisted of the 'id' of the review, the 'sentiment' of the review, the 'score' of the review and the actual review itself. This dataset was split into 5 sets of training/testing splits and a binary classification model was developed to classify that review as either 0 which is negative or 1 which was positive. The constraints on the model were that the vocabulary size used to train the classifier must be less than 2000 words and the classification AUC must be equal to or bigger than 0.96.

The splits of the movie review data set were read in as input and the text was converted into word count vectors with some transformations. A vocabulary was selected from these results and saved to a text file. A binary classification model was then applied to these 5 splits with the selected vocabulary and the result file was generated for each spit as an output on which the AUC was calculated.

Let us look at the implementation steps that were done for this project.

2.DATA PROCESSING

The IMDB movie reviews dataset - alldata.tsv file which was provided to us was divided into 5 different splits. Each split had the following files: -

1. train.tsv – This file contained the training data for the model. It only consisted of the 'id', 'sentiment', and the 'review' of the movie. 'score' was removed as we didn't want it as an input feature.
2. test.tsv – This file contains 2 variables 'id' and 'review'. The 'sentiment' column was removed from this dataset.
3. test_y.tsv – This file contains the sentiment for each id in the test.tsv file so we can validate the performance of the model.

Using all of the 5 splits the custom vocabulary used to train the model was generated. A more detailed explanation of how the vocabulary is generated can be found in the myvocab_creation.html file included with the submission.

3.CLASSIFICATION MODEL

We decided to use a logistic regression with L2 (ridge) penalty as the sentiment prediction model. The rationale here is that all terms from the custom vocabulary were useful and important for the predictive model. We performed cross validation using cv.glmnet and were able to find the corresponding lambda min for a given split. With regards to the cv.glmnet we used **L1** penalty and the default **10 fold cross** validation. Using the **lambda min** obtained from cross validation, we predicted the sentiment using the logistic regression model.

4.RESULTS

The tabulated results for the AUC with a vocabulary less than 1000 terms for each respective split are as given below:

Split	AUC
Split 1	0.9638
Split 2	0.9623
Split 3	0.9628
Split 4	0.9628
Split 5	0.9623
Vocabulary Size	960

We can see from above table that as required, each of the splits have an AUC > 0.96 and a vocabulary size less than 1000 terms (960 terms).

The tabulated run times for the 5 splits on the three machines we tested are as follows: -

Machine 1	Machine 2	Machine 3
47.869 secs	57.9627 secs	1.1635 mins
50.5367 secs	1.311 mins	1.1291 mins
47.535 secs	59.713 secs	1.1891 mins
47.8119 secs	56.775 secs	1.1123 mins
47.870 secs	56.914 secs	1.2042 mins

The configurations of each machine can be found in the Section 9 – Technical Specifications.

5.INTERPRETABILITY

A logistic regression model is a relatively easier model to interpret compared to some more complex models for e.g. deep learning models (RNN/CNN). This is due to its monotonicity constraint that ensures that a relationship between a feature and its target always goes in the same direction over the entire range of the feature. In our sentiment analysis the relationship between a so called “positive word” and the positive sentiment remains the same for all values in the corpus. A positive word is always considered positive in a short context. Hence, we have obtained a clear list of positive words and negative words by segmenting the corpus in positive and negative n-grams and the model can then easily be interpreted to understand why a specific sentiment was given to a specific review. For example, if we take the given review:

“This is one of the best romantic movies I have ever seen. Especially girls who can identify with Nicole will love it(not only because of the handsome Dalton James) I also liked the music very much. A highlight was land of the sea and sun from baha-man. So watch the movie and enjoy it”

Lots of positive n-grams like “the best”, “love it”, and much more are frequent in the review thus the model gave it a positive sentiment. To contrast, here’s an example of a negative review:

“This movie has one of the worst lead characters ever. I say this because he is made out to be the hero when, in my opinion, everything he does in the whole movie screws up people's lives and causes problems. He can do nothing right, yet the movie makes him seem like the cool dude everyone should be looking up to….”

This review was assigned a negative sentiment, and clearly contains multiple negative n grams like “the worst”, “screws up”, etc. hence a negative review.

6.LIMITATIONS AND FUTURE IMPROVEMENTS

One limitation of this model lies in the selection of terms to include in the vocabulary. The t test statistic provides a simple method of weighing the relative use of terms in the corpus, but it does not take into account relationships between terms or the usefulness of terms for classification. Only n-grams up to size 4 were used which provided useful results but again larger n-grams didn’t prove to provide a much better performance. This is because ultimately Bag-of-ngrams only considers word order in short context; it loses ordering in higher dimension and thus causes context sparsity. Even though a simple linear model proved to be relatively effective, it failed to capture any possible relationship between terms or non-linearities in the decision boundary. Therefore, a deep learning model, possibly using LSTM and/or RNN layers, may result in better classification, as it would be able to capture subtle non-linearities and relations between terms. Use of word embeddings to represent words such that similarly meaning words have similar encodings could alleviate some of the observed misclassifications as well.

We also observed some other interesting findings when working on developing the model. The accuracy dropped when extra data preprocessing was done like removing numbers, special characters, extra spaces from the review. It also dropped when we used the stopwords library which contains a larger list of stopwords.

7.CONTRIBUTIONS

We began by approaching the project each independently, to ensure we learned by doing. We then had check ins where we discussed what strategies we tried, and shared findings. Tess got the AUC above 0.96 on all the splits for a vocabulary size of 2000. Preethal was able to troubleshoot the code and customize the vocabulary so we were able to get an AUC above 0.96 with a vocabulary size of 960. Rajesh worked on the report, researched other potential techniques and helped trouble shoot the code. We all learned quite a lot by working on this project.

8.CONCLUSION

We were able to meet the required AUC threshold and vocabulary constraints with a logistic regression model.

9.TECHNICAL SPECIFICATIONS

The two models in this project were tested on a Macbook Pro, 2.3 GHz Quad-Core Intel Core i7 and 16GB RAM

1. Macbook Pro, i5 1.4 GHz Quad-Core Intel Core, 8GB RAM
2. Macbook Air, 1.6 GHz Quad-Core Intel Core, 8GB RAM
3. Macbook Pro, 2.3 GHz Quad-Core Intel Core, 16GB RAM

10.ACKNOWLEDGEMENT

1. Dr. Liang's solution approach provided on Piazza
2. Piazza posts and Office Hours