STAT 542 Project 2 Report

Bin Feng

1 INTRODUCTION

In this project, we are provided with a dataset consists of IMDB movie reviews, where each review is labelled as positive or negative. The goal is to build a binary classification model to predict the sentiment of a movie review. Three classification models were built to achieve the goal:

1) logistic regression with LASSO;
2) linear discriminant analysis with PCA;
3) logistic regression with boosting tree and LASSO.

One .R code file and one .txt file were produced as follow:

1) mymain.R (serve as the main file to read datasets, run the three prediction models, and write predictions to txt submission files);
2) myvocab.txt (helper file stores words for prediction based on two-sample t-test);

Classification evaluation metric is AUC (i.e. area under the receiver operating characteristic curve). Full credit, if one model can produce AUC equal to or bigger than 0.96 over all three test data. Another requirement is that the vocabulary size is restricted to be less than or equal to 3000.

2 DATA PREPARATION

2.1 Words Extraction

The IMDB movie review dataset has 50,000 rows (i.e., reviews) and 3 columns. Column 1 "new_id" is the ID for each review (same as the row number), Column 2 "sentiment" is the binary response, and Column 3 is the review.

Another “Project2_splits.csv” file was also used for separating dataset into training and test samples. The dataset has 3 columns, and each column consists of the "new_id" (or equivalently the row ID) of 25,000 test samples. That is, using "Project2_splits.csv", we can split the 50,000 reviews into Three sets of training and test data.

After reading the whole dataset into R, following preprocess steps were conducted to extract meaningful words for classification:

1) Remove html tags: movie reviews in the dataset contain some html tags (e.g.). I want to remove them because these tags are don’t meaningful in terms of classifying whether a review is positive or negative.

2) define stop words: stop words refer to the most common words in a language, there is no single universal list of stop words used by all natural language processing tools. Therefore, I need to define them by ourselves and remove them from our analysis.
3) create word phrases consist of one to four words: sometimes words are meaningful only when they are used together. Therefore, I create phrases that can contain word up to four.
4) prune word phrases: after creating phrases, I got 9293186 phrases which are much more than I can handle. I filter them by minimum number of occurrences over all documents and proportion of documents which should contain term.
5) word selection based on two-sample t-test: after word pruning, I still have 28628 phrases left which are more than the 3000 words requirement. I select 2000 significant phrases in terms of their t statistics in the two-sample t-test between positive and negative reviews. These 2000 phrases were saved into a .txt for future usage.
6) vectorizer pruned word phrases: I transform the selected list of phrases into vector space for further classification.

2.2 Logistic Regression with LASSO

I used *"glmnet"* package to performe logistic regression and LASSO regularization. Ridge regularization and mixed regularization were also tested by tuning the alpha value, the classification performance has some improvement in a magnitude of 0.001. Such improvement is expected since more features would be used under ridge regularization.

To find the lambda value in a cross validated form, I used *"cv.glmnet"* function and chose the minimum lambda for classification. For classification robustness, 1se lambda can be selected. Results were written into a "mysubmission1.txt" file.

2.3 Linear discriminant analysis with PCA

To perform PCA on the training dataset, I calculated the covariance matrix and find the corresponding eigenvector. Based on past experience, I chose the first 1000 PCA directions and map the training dataset into them as the transformed data.

I used *"MASS"* package to perform LDA on the transformed data. To make prediction on the test dataset, I also needed to map the original test data into the selected 1000 PCA dimensions. Results were written into a "mysubmission2.txt" file.

2.4 Logistic regression with boosting tree and LASSO.

I used *"xgboost"* package for generating new nonlinear features. In total, 500 trees with max depth of 2 were created. Based on the created trees, I used *"xgb.create.features"* to generate a binary design matrix for the new categorical features.

After obtaining the new feature matrix, similar logistic regression with LASSO regularization as discussed above is conducted. Results were written into a "mysubmission3.txt" file.

3 MODEL RUNNING TIME

All three classification models were run a MacBook Pro. Detailed computer configuration are as follow:

> macOS Mojave (version 10.14.3), 2.6 GHz Intel Core i7, 16 GB 2400 MHz DDR4, Radeon Pro 560X 4096 MB, Intel UHD Graphics 630 1536 MB.

Running time for all three prediction models are summarized in Table 1 as below:

TABLE 1 Running time for three classification models

Data split No.	Logistic/s	LDA/min	Boosting Tree/min
1	**15.98115**	3.792313	8.322562
2	**16.76817**	4.074984	7.016621
3	**18.09169**	4.055041	6.746919

Note that running time for Logistics model is much smaller than the ones for LDA and Boosting. This is because the other two model contains some time-consuming steps: LDA model requires performing covariance matrix and eigenvector calculation; Boosting Tree needs to build 500 trees and generating new features from them. Logistics model, however, only needs to process the original phrases matrix.

4 MODEL CLASSIFICATION ACCURACY

Model classification accuracy (AUC) is calculated based on the evaluation metric provided in the introduction section. Classification accuracy for all three models are summarized in Table 2 as below:

TABLE 2 AUC for three classification models

Data split No.	Logistic	LDA	Boosting Tree
1	**0.9623419**	0.8865600	0.9522102
2	**0.9617383**	0.8862826	0.9525340
3	**0.9605843**	0.8832643	0.9525392

For the 3 data splits, it is very interesting to see that the logistic regression model with LASSO regularization always has the best performance and beats the benchmark. Another interesting finding is that if we use Ridge regularization instead of LASSO for the logistic regression model, corresponding AUC will increase in a magnitude of 0.001. Possible improvement for the inferior performance of the other two models can be: LDA model only uses the first 1000 PCA directions which can miss some important dimensions; Boosting Tree only generates 500 trees which can also miss some important features. Increase number of PCA directions and boosting trees will increase the AUC performance. The reason for only using 1000 PCA directions and 500 trees is to make the running time for the whole program acceptable.