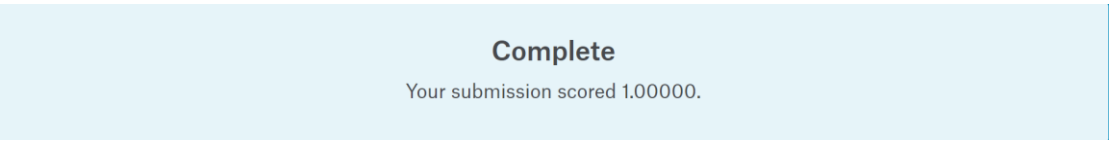
Stat 542 Third Project-Movie Review Sentiment Analysis

Team: RSS

0. Notice

In this project, the datasets have severe leaking.

The id column of both “testData.tsv” and “labeledTrainData.tsv” have format XXXX_N, where N is the true rating of the review, ranked from 1-4 and 7-10. Based on this leaking, we can easily reach prefect prediction (Area under Curve = 1) on Kaggle.



This is a skill less and meaningless method. In the following code and discussion, we won’t use this result in training and visualizing process.

The code is included in mymain.R and will generate a file called “cheating.txt”.

1. General Idea

(1) Data cleaning steps

First, we use package “text2vec” to generate a dgcmatrix with words as factors. By applying tuning parameters, we can select the word vector we need. NGram is set as (1L,4L) to make sure that some of the phrase like “I love this file”, “this is a great”, “one of the best” is included in the word vector. Also, prune vocabulary parameters are adjusted to make sure the length of vector is around 50,000.

Notice that in this case numbers like “7/10”, “4/10”, could give important information in modeling, so the number phrases must be included in the word vector.

(2) Model selection steps

Xgboost is selected to train the model due to its astonishing speed and powerful ability to deal with dgcmatrix.

Binary logistic classification model is chosen as algorithm. We abandoned tree method cause in this case, with 25,000 observations and 50,000+ features, applying a tree may lead to severe overfitting. So, we settled for the traditional binary logistic model.

We selected the tuning parameters simply based on Kaggle performance, some of those parameters may seem unreasonable but will lead to a higher AUC. If we apply a parameter selection process or perform some transformation on the dataset and model, we may lift the AUC by 0.01-0.02. But we are satisfied with AUC=0.96755 and running time of 3 minute.

2. Visualization

(1) Sample Visualization:

Based on the files provide by TA, we generate a sample visualization html file which contains the first 100 reviews from training data. Here are some improvements we made based on the original R code.

① We further divided the wordlist into 4 levels called “strong positive”, “positive”, “negative”, “strong negative”. The classification is based on the weight of each feature in binary logistic model. “strong positive/strong negative” words are defined as absolute value>0.2 and “positive/negative” words with absolute value between 0.1-0.2.

In the html output, the colors of strong positive/positive/negative /strong negative are orange/yellow/pink/red.

②Blanks are added in front of punctuations to make sure punctuations won’t appear in the word list.

③ A blank is added after both words in “grep” function to make sure the match is what we need. For example, “mess” is the strong negative word but “message” is a positive word. The blank is added to avoid marking “message” as red.

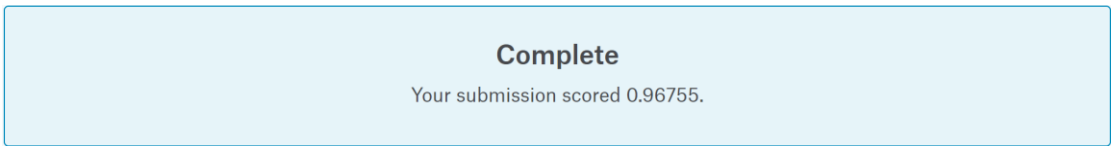
(2) Additional Visualization

We tried to make some word clouds to visualize the importance of each word in modeling. The importance is determined by the coefficient in training model and frequency in reviews.

3. Model Performance

	Performance
Running time (Based on i7-6820HQ with 32GB RAM)	Data cleaning time:102 seconds
	Modeling time:2.48 seconds
	Visualizing time:23.38 seconds

AUC (Kaggle Rating)	0.96755
Accuracy	0.91228



4. Conclusion

(1) This project requires text mining skills. Compared to Python, there are not many packages available in R in text mining field. But text2vec and feature hashing is enough for this project.

(2) Overfitting might be severe in training model of word features because they are too many features. So, choosing relatively simple model may come with better result.

(3) Xgboost method is efficient. Its speed and ability to deal with sparse matrix makes it a extremely useful package.

5. Acknowledgement

This modeling idea is from lewis ml wrote from Kaggle. His or iginal code is on https://github.com/wush978/FeatureHashing/blob/master/vignettes/SentimentAnalysis.Rmd.

In our project, we applied text2vec instead of directly apply feature hashing since we can prune and select some vocabulary in generating word vector. Also, we supplemented and modified some key tuning parameters to improve AUC value.

Sample visualization code is from resource on Piazza from TA.