Movie Review Sentiment Analysis

The movie review data used for this report is downloaded from the CS 598 (Stat 542) Piazza forum under the resources tab. The dataset contains 50,000 IDMB movie reviews, where each review is labeled as positive or negative. Each review has an identification number, sentiment (0=negative, 1=positive), score (on a 10 point scale – higher is more positive sentiment), and the review text. The score of a movie is on a 1-10 point scale with 1 being the most negative sentiment and 10 being the most positive sentiment a reviewer could give to a particular movie. 1-4 are considered negative reviews and 7-10 are considered positive reviews. No movies in this dataset are rated 5-6 – which might be considered a neutral review. The goal of this project is to build a binary classifier that will predict whether a movie review is positive or negative.

The movie review data is broken up into 5 different splits of 25,000 rows of training/test data. The splits have been chosen by the course staff in the splits_F20.csv file found in the CS 598 (Stat 542) Piazza forum under the resources tab. The reviews not chosen for training in each of the splits will be used for testing. The score column is removed from the training file so that this feature cannot be used in training a model to determine movie sentiment.

The metric to measure the performance of this model will be the area under the curve (AUC). The AUC should be greater than 0.96 for all 5 splits of the data with a vocabulary under 1000 words – where words can be a term involving multiple words.

Model Selection

Since the model needed is a binary classifier, a natural choice would be to try a logistic regression model to classify a review as either negative (0) or positive (1). Table 1 below shows that the review sentiments are fairly evenly distributed between positive and negative, so that there isn't any processing that needs to be done to account for an unbalance between the two classes.

The only data available to the model would be the id of the review and the text of the review. The id of the model doesn't really add any value to the classification since each review id is unique. That leaves the text of the review as the input into the model. A natural language processing model is needed for this classification. This report uses an NLP method known as bag of words. In such a model, each review is reduced to a vector that represent how many times each word in the overall vocabulary is used in that review. It is also possible to look at terms involving two, three or more words (n-grams) that may help to better classify a review as positive or negative.

The R text2vec package was used to help use a bag of words model.

Table 1 - Histograms of sentiment in each training split

This vectorization method produces over 9.5 million words/phrases (up to 4-gram words) that can be used to train the model. This is a very high dimensional model and could take a significant amount of time to train. This report prunes this vocabulary to get below 1,000 words (970 in fact) to help reduce the time it takes to train the model. The exact methodology used to produce the 970 word up to 4-gram vocabulary is discussed in the *Project3_fre2.html* file that accompanies this report.

The training data was fit to a logistic regression model as described below:

(1) Remove html tags using regex expression <.*?> from each review
(2) Convert all letters to lower case
(3) Split text into lists of tokens
(4) Construct a Document-Term Matrix by mapping the list of tokens into the 970 word/phrase vocabulary
(5) Use a 10 fold cross validation glmnet with a ridge penalty term using the *binomial* family and AUC as the metric to find the best lambda that fits the data.
(6) Use glmnet with a ridge penalty term and this lowest lambda using the *binomial* family to fit a logistic regression model to the training data.

This fitted model was then used to predict the sentiment of the test data in much the same way:

(1) Remove html tags using regex expression <.*?> from each review
(2) Convert all letters to lower case
(3) Split text into lists of tokens
(4) Construct a Document-Term Matrix by mapping the list of tokens into the 970 word/phrase vocabulary
(5) Use glmnet.predict and *response* type using the fitted model above to predict the class probability for each review.

For this report AUC was used as the evaluation metric, so the output of the logistic regression needed to be the class probability, which is why the *response* type was used in the glmnet.predict function.

Limitations of Model

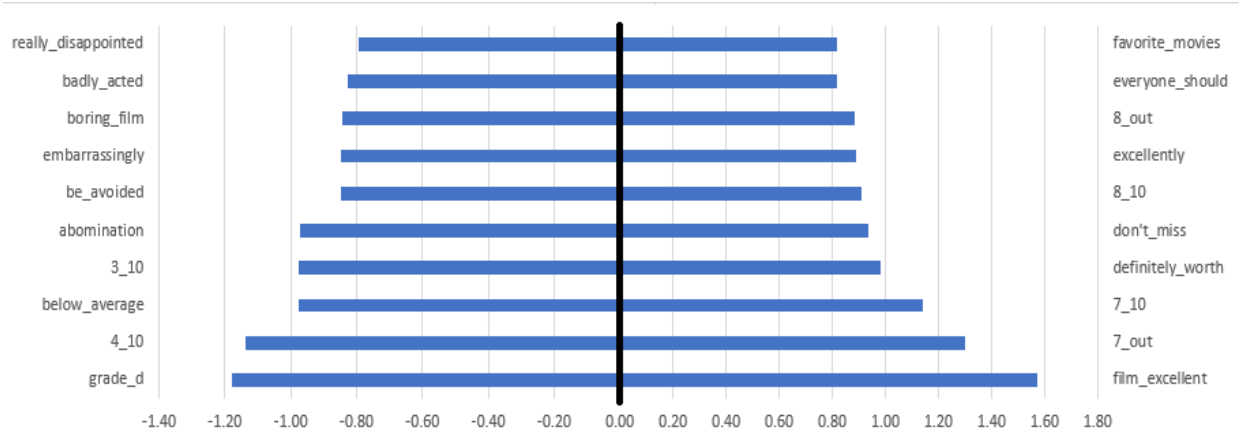
As mentioned above, the size of the vocabulary used to train the model was reduced from over 9.5 million to below 1,000 up to 4-gram words. The reduction methods could have affected the accuracy of the model. While the words pruned from the original vocabulary were shown to have a low two sample t-statistic, the removed words in combination with other words could have been significant to the accuracy. This was a calculated risk in order to improve training times.

In looking at some of the misclassified reviews, another limitation was seen - the ability of the model to correctly predict the sentiment of reviews that use language that is positive as well as language that is negative. Sometimes this was due to sarcasm, other times not. The model certainly is not good at detecting sarcasm and can be seen to wrongly classify reviews because of this. Other times a reviewer simply used words that had negative sentiment and words that had positive sentiment and the model did the best it could and sometimes it was correct and other times it was not.

This report uses a logistic regression model using a ridge penalty term. The model simply classifies a review as positive or negative. There is a range of values (1-10) that a reviewer can use to gauge the sentiment of the movie. This dataset did not have any values of 5 or 6, but 1-4 are considered negative sentiment and 7-10 are considered positive sentiment. Someone using this model to decide on a movie to watch might have a higher inclination to watch a move rated as a 10 as opposed to a movie rated at 7. So, while the model in this report does a good job of classifying a movie review as positive or negative, a user may want this sentiment on a scale to make a more informed decision of what movie to watch.

Interpretability

This report employed a logistical regression model, so the weights/coefficients of the different vocabulary words/phrases can be viewed to understand how a particular review was predicted as either positive or negative. The magnitude of the coefficient shows the importance of the corresponding term and the sign shows whether the term contributes to positive or negative sentiment. The top 10 words showing negative and positive sentiment and their corresponding weight/coefficient are shown below.



Future Steps

This report used raw word counts to train a logistic regression model, but there is a transformation known as term frequency–inverse document frequency (tf-idf) which adjusts values according to the number of reviews that use the word. Words that occur across many reviews may be less useful in classification than words that occur less frequently. The tf-idf transformation reduces the value of those words based on the number of reviews that word is in. Words with a higher tf-idf are considered more important words because they occur less frequently. This method may be able to improve accuracy of the model.

Another step may be to use a multinomial model in order to give better recommendations. As mentioned above a user may be inclined to watch a movie rated as 10 rather one rated at 7.

Another possible algorithm could use neural networks to generate a model to classify sentiment. Neural networks have been shown to be very powerful with the right configuration.

Results

The below table shows the AUC values for the 5 different splits for the final model.

	Split 1	Split 2	Split 3	Split 4	Split 5
AUC	0.9601	0.963	0.9631	0.9636	0.9628

The model was run under the following environment:

Computer Component	Component Characteristic
Processor	**Intel® Core™ I3-3227U CPU @ 1.90 GHz**
Installed RAM	**8.00 GB (7.89 GB usable)**
System type	**64-bit operating system, x-64-based processor**
Operating System	**Windows 10 Home, Version 2004, Build 19041.508**
Running time for 1 split	**Approx. 127 seconds**