Sentiment Analysis on Movie Reviews

Uan Sholanbayev
UCSD: A98057203
usholanb@ucsd.edu

Abstract

The problem was taken from Kaggle competition called "The Bag of Words Meets the Bags of Popcorn". Most of the time reviews on movies carry sentiment which indicates whether review is positive or negative. The goal of this paper is to predict the sentiments of reviews using basic algorithms and compare the results. In this assignment we are going to experiment several methods, namely, n-grams, Naïve Bayes Model and it's a little optimized version which we will call "Advanced Naïve Bayes" in this paper.

1. Introduction

The Sentiment Analysis is a field that evaluates the emotions and feelings in the review texts. In order to calculate the sentiment score of the review, each piece of text can be examined separately or in combination with others. In this manner, after calculating the sentiment scores of all the pieces of text in the review some aggregation technique is used to calculate the overall sentiment of the review. One of the simplest ways of combining the total score of the review is summing all the scores of all the pieces in that review. Certainly, it is our choice to select how big those pieces will be. We can choose every word, n subsequent words, sentence,and/or whole review to represent a feature. However, due to the fact that there is a little chance that there will be repeating reviews, it is more reasonable to have smaller pieces of text as representations of features. Therefore, we will calculate the scores of every bigram and trigram and compare the results. Then we will apply Naïve Bayes Model, which will have every word to represent a feature, and compare that models results with bigram and trigram models.

1.1 Exploratory Analysis:

We are given a training and a test set both including 25, 000 user ids and movie reviews. In training set we are also given a sentiment of the movie which is 1 if it is positive and 0 otherwise.We are also given 50, 000 ids and movie reviews as unlabeled training set but we will not use that data. Due to huge number of reviews in traning set and, therefore, enormous number of words we had to discard all non-necessary words by several steps:

- Lower case of all words
- Delete all the punctuations
- Delete all the stopword such as "the", "a", "to", etc.
- Stem the words(for example, fishing", "fished", and "fisher" to the root word, "fish")
- Discard all words that occur less than or equal to 15 times because there is little change they relate to the sentiment of the review
- Finally, collect all the left unique words and their frequency of occurrence

Overall, there are left only 49928 unique words. The following most and least popular wordss were found after stemming, converting to lower case and "cleaning" the reviews from stopwords, punctuations, etc. Each recorded word appears in the labeled train set more than 15 times(Table 1). The reason to keep only words of this

frequency and higher is because completely rare words are often just mistyped, non-existing or just meaningless words that would not contribute much as features to our models.

The same process was applied to unlabeled data(Table 2)

Unlabeled Train Set	
Most Popular Words	
Word	Frequency
movi	103613
film	97770
one	56253
like	45718
time	31882

Table 2 - Most and Least Popular words

Labeled Train Set			
Most Popular Words		Least Popular Words	
Word	Frequency	Word	Frequency
movi	51596	Abnorm	16
film	48191	Advani	16
one	27742	Afar	16
like	22799	Airhead	16
time	16191	Alaska	16

Table 1 - Most and Least Popular words

For the unlabeled train set the most popular words turned out to be very similar to the ones in labeled train set. This is not surprising because in movie reviews in average people tend to use the same words.

2. Predictive Task

After the data analysis that is done in the previous section, our goal is to make a prediction of the review sentiment based only on review text.

$$f(review\ text\ features) -> \ sentiment[0\ or\ 1]$$

One of our variable will be the review text features themselves. We will estimate the sentiment score based on trigrams. We will also switch the frequency threshold that will be responsible for considering only those trigrams whose positive/negative scores are higher than the threshold's value.

Then we will compute the score based on bigrams. We will compare different results obtained from bigrams approach based on changing the threshold of frequency of occurrences (the same principle as with trigrams)

Finally, we will use Naïve Bayes Model to predict the sentiment. Because by default Bayes uses all the features given and in our case those features are words, we will test how Bayes model will work if the features are selected more carefully. By choosing only those words that appear the most in positive or the most in negative, it is natural to assume that our prediction accuracy will improve.

3. Relative Work and Literature

Andy Bromberg used Naïve Bayes Model and made a great experiment varying the threshold that decided which words in texts the model can consider as features. Another very helpful source that gave a good kick-start for this paper are the tutorials for this competition on Kaggle. There are much more information and tricks that could also be helpful, however, only the simplest methods were chosen to make the prediction tasks.

4. Feature Selection

As we already discussed in **Section 2** our features will be single words, bigrams and trigrams. For Naïve Bayes Model and it's a little advanced version we will use words as features. When calculating scores according n-grams method we will compute scores of two or three subsequent words.

Also for each method we will change the threshold which will filter features that are most likely useless for our models and compare the results.

5. Experiments

One of the potentially good methods that can apply to this problem is analyzing the score of every review using n-grams. The idea here is to assign a score to every trigram and calculate the sum of scores of all the trigrams in the review. If the sum is greater than some threshold then assume the sentiment is positive and vice versa. Due to the fact that it possible to make a submission to the competition to Kaggle only five times a day the training set was split by 3 quarters used for training the models and one quarter for testing.

5.1 Trigrams

First we are going to test the accuracy of trigrams model. Every time a trigram appears in the positive review we increment its score by 1. We found out the trigrams that have the highestpositive scores. Not surprisingly those are:

Most Positive Trigrams	one best movi	new york citi	one best film	well worth watch	best movi ever
Sentiment Score	83	66	62	60	49

Table 3 - Most Positive Trigrams

According to Table 3 it is obvious that trigrams model's features are reasonable. The only unexpected trigram is New York City. It is natural to assume that the reason for this phenomena is people like NY and the movies that are shot there are mostly well financed films. The last assumption is based on fact that NY is a huge city, so the budget is expected to be big too.

The same way we will decrement the score of the trigram every time it appears in a negative review.

After calculating scores for every trigram we will evaluate the scores of reviews in test set. If the review score is higher than 0 we assume it is positive, otherwise negative.

The resulting prediction was 74% of correctness. Because some trigrams cannot be classified with certainty, it is reasonable to get rid of trigrams those trigrams from the list of our features. Therefore, we will filter our model from trigrams that are undetermined. Those will be the trigrams whose absolute scores are below our filter. After playing a little with the value of the filter the following results were found(Table 4)

Trigram Sentiment Score Filter	Prediction Correctness
0(no filter)	74.72 %
3	65.376 %
5	61.728 %
10	56.816 %
15	54.768 %

Table 4 – Trigram Model

It is clear that even though we got rid of misleading trigramsour prediction correctness decreased. The explanation for this unintuitive concept is the fact that many reviews have scores of 0. There are a lot ofreviews that consist completely new trigrams and our model could not classify them.As it was conditioned in the beginning that if the review gets a score of 0 then it is considered to be negative. Therefore, in order to increase the correctness percentage we need either more data to have a better trained trigrams model or more features. However, whenever the absolute value of a review's score is high in this model, it is safe to assume that prediction is accurate.

5.2 Bigrams

Now we will try to implement the same principle but with bigrams. According Table 5 we see that bigrams also collected reasonable features. The final results of this model are in Table 6

The results obtained from bigrams are a little higher than those we got from trigrams. One obvious reason is that trigrams are more unique than bigrams and because our data set is not big enough bigrams are more informative whether the review is positive or negative.

Our next step will be adjusting the bar below which all the less informative bigrams are ignored. Although we made a reasonable adjustment to our bigram method the results become less accurate as we raise the value of our filter. The intuition here is that having less features will worsen our bigram model.

Bigram Sentiment Score Filter	Prediction Correctness
0(no filter)	77.392 %
3	76.736 %
5	76.192 %
10	56.816 %
15	54.768 %

Table 6 Bigrams Model

Labeled Train Set			
Most Positive Bigrams		Most Negative Bigrams	
Bigram	Sentiment Score	Bigram	Sentiment Score
one best	680	look like	-845
high recommend	400	wast time	-828
must see	317	worst movi	-560
first time	267	bad movi	-541
love movi	265	one worst	-469

Table 5 - Most Positive and Negative Bigrams

5.3 Combination of Trigrams and Bigrams Models:

As it was seen in previous steps the trigrams model can make solid prediction but at certain times when it meets already familiar trigrams in the test data reviews. Bigrams model appears to be more trained because bigrams are not as unique as trigrams. Therefore in this step we will combine previous two models. **Whenever the trigrams model gives a reasonably high absolute score to a review, we will trust its prediction. However, whenever the resulting score will be less than the initially considered threshold, we will trust the prediction of the bigrams model.** Based on previous steps we know that the modaels are most efficient when all bigrams and all trigrams are used as features, so we won't filter them for this model.

Table 7 – Trigram&Bigram Model

Trigram Sentiment ScoreFilter	Prediction Correctness
1	78.544 %
10	78.752 %
30	77.584 %
50	77408 %
100	77.424 %
150	77.408 %

The results from the combined model are little higher. They fluctuate a little around 78% and the reason is probably because trigrams model is usually consistent with the decision of bigrams model. Because trigrams rarely can make a reasonable decision due to lack of training, the predictions do not change substantially. Nevertheless, the accuracy is getting better, and the intuition behind this is that trigrams model gives a more definite prediction than bigrams modelwhenever it has a voice.

5.4 Naïve Bayes Model

In short, Naïve Bayes Model classifies based on the features of frequencies of words in the text. The initial assumption of Naïve Bayes Classifier is that all its feature are independent from each other. Surely, this algorithm is quite simple and cannot make very accurate results. However, in practice and in our case, Naïve Bayes model proved to be very useful. Wikipedia was very helpful refreshing the concepts of Naïve Bayes Model (http://en.wikipedia.org/wiki/Naive_Bayes_classifier)

Fortunately, there are many useful Python libraries that lets us easily implement Bayes model and evaluate the results. The final result of the Bayes Model increase the accuracy up to 82.0%.

5.5 Advanced Bayes Rule

The next step is to apply a filter that would choose only most informative words. This is still Naïve Bayes Model but it is a little optimized. All the words that carry little information regarding the sentiment of the review will be dropped as features of Naive Bayes Model. First, we need to sort all the words according their scores. Then we will cut off all the words that carry little information and pass everything else to the Naïve Bayes classifier.

Number of features(words)	Prediction Correctness
500	81.3%
2000	83.9%
15000	83.3%
30000	83.0%
49928(all words)	82.0%

Table 8 – Advanced Naïve Bayes

6. Conclusions

In this paper we use basic machine learning techniques and explore how useful they can be in predicting sentiment of movie reviews. Even with small amount data and using simple approaches to train our model we can make a quite accurate prediction of the text's sentiment. Unfortunately, the n-grams method was not as precise as it was expected to be(Table 4, Table 6). However, the Naïve Bayes Model performed relatively well on our data set. By choosing what features our model must consider we can increase the accuracy of our results. The basic methods such as Naïve Bayes Model consider every feature to have the same weight, however, as we have seen in this paper, to have more accurate results they all must be treated differently.

7. References

1. Andy Bromberg(2013) Second Try. Sentiment Analysis in Python. Retrieved from http://andybromberg.com/sentiment-analysis-python/
2. Vik Paruchuri(2015) Using Naïve Bayes to Predict Movie Review Sentiment. Retrieved from http://blogdataquestion/blog/naive-bayes-movies/