L90 Assignment 1:
Naive Bayes for Movie Review Sentiment Analysis
A Partial Replication of Pang et al. 2002

Jennifer White (jw2088)

October 29, 2019

Abstract

This work partially replicates previous work done in the domain of sentiment classification by using a Naive Bayes classification model to classify movie reviews as being either positive or negative. It uses Bag of Word features with both unigram and bigram modelling, individually and combined, both with and without Laplace smoothing and recording both presence and frequency of features. It finds that combining unigrams and bigrams offers the best results and that Laplace smoothing results in a statistically significant improvement in performance regardless of the other details of the model. Using cross-validation, the highest average accuracy reached on the test set was achieved using a model incorporating both unigrams and bigrams, with add-one smoothing and a feature vector indicating only presence of vocabulary items. This model achieved an average accuracy of 84.15%, and a variance of 0.00106 across the held-out test sets.

1 Introduction

1.1 Sentiment Classification

Sentiment Classification of text is the task of classifying the overall sentiment of a piece of text as postive or negative. In particular, this is often useful for reviews of a product or, in this case, of films.

1.2 Pang et al. (2002)

This work is a partial replication of the work done by Pang et al [1] on sentiment classification of movie reviews. Pang's paper examines the use of a Naive Bayes model, Support Vector Machines and Maximum Entropy classfiers for carrying out the sentiment classification of movie reviews. This work will partially replicate their use of the Naive Bayes model.

2 Data

The data used for this work consisted of 2000 IMDB reviews, equally divided between positive and negative reviews. The data was available both in its raw text form and as a list of tokens with Part of Speech tags. For this work, the raw text was used as Pang (2002) found that using tags as a feature did not result in a statistically significant improvement in their results.

10-fold round-robin cross-validation testing was performed on the dataset.

3 Method

The code used for this work can be found on my user area on the MPhil machines (user: jw2088) in the file `naivebayesvec.py`[1].

[1] Alternatively, it can be viewed at `https://bit.ly/31XUkui`

Table 1: Table of average accuracy percentages achieved for each model, with variance shown in brackets

Features	**No smoothing, freq**	**Smoothing, freq**	**No smoothing, pres**	**Smoothing, pres**
Unigrams	64.65 (0.00032)	81.95 (0.00102)	64.60 (0.00050)	82.45 (0.00104)
Bigrams	60.30 (0.00075)	81.75 (0.00161)	60.30 (0.00077)	82.55 (0.00156)
Unigrams and Bigrams	56.50 (0.00052)	83.95 (0.00091)	56.50 (0.00052)	84.15 (0.00106)

3.1 Naive Bayes

For this work a Naive Bayes model was implemented. Cross-validation was used, so each model was trained on 9 of 10 file sets and the final set was then used to test the classification accuracy of the model. The accuracies found through this method were averaged for each model.

In training each model, vocabulary items (either unigrams or bigrams) wrere counted in the training texts and statistics regarding their appearance were calculated. In testing, a review was encided as a bag of words vector and the statistics that had been calculated were used to calculate the probability of it being a positive or negative review. This calculation was performed in log space to avoid encountered problems caused by dealing with the multiplication of many small numbers.

Frequency cutoff for the vocabulary was used in each model. Only vocabulary items that appeared 5 times or more in the text were featured in the vocabulary.

3.2 Smoothing

Due to the way that Naive Bayes models train, any review that contains a word that was not seen in the training set for a document of a certain class will always receive a probability of 0 for being in that class. This is clearly not accurate and has the possibility to degrade the quality of our classification model. One way to address this is by using Laplace Smoothing. Laplace smoothing involves giving some initial value to the count of each unigram or bigram so that no vocabulary item from the training set will have a probability of zero. This is desirable as even very rare and unlikely words can occur in a new, unseen review.

For the tests conducted in this work, add-one smoothing was used, so every unigram or bigram in the vocabulary was given an initial count of one.

3.3 Unigrams and Bigrams

In this work, the features considered took the form of either unigrams, bigrams or a combination of both in a Bag of Words model. Unigrams are the simplest feature to consider, but they may miss additional context around the word that bigrams should ideally be able to capture. Although bigrams should do a better job at capturing this additional context, they may struggle with a sparsity of data for each bigram resulting in worse probability estimates.

3.4 Feature Vector

In this work, two methods of handling the feature vector were tested. In the presence method, the i^{th} element of the feature vector contains a 1 or a 0 to indicate whether the vocabulary item indexed as i appears in the review or not. In the frequency method, it instead contains an integer indicating the number of times that the vocabulary item occurs in the review.

4 Results

The full results for all of the model configurations are shown in Table 1. Here I will only discuss which confguration decisions gave sta-

tistically significant differences according to the sign test ($p < 0.05$).

4.1 Statistically Significant Differences

Across all configurations of the model, models with add-one smoothing were consistently found to perform better than their counterparts without smoothing. This difference was statistically significant in all cases and the difference in average accuracy was typically quite large.

In the models that did not use any form of smoothing, unigrams were found to perform better than bigrams, with statistical significance. However, in the models where smoothing was used no statistically significant difference was found between unigram and bigram models. Models combining both unigrams and bigrams were found to be better in a statistically significant way for all models with the exception of the model using smoothing and presence of features for which the difference was not statistically signiifcant.

Using frequency of vocabulary items in feature vectors was not found to give different results compared to recording presence, within limits of statistical significance.

5 Discussion

Overall, this

5.1 Possible Improvements

It should be possible to achieve better results for this task using other classification methods, such as Support Vector Machines as used by Pang et al.

It may be possible to achieve further improvements in the accuracy of the models presented here by experimenting further with the value chosen for the frequency cutoff and the value used as the smoothing constant. In this work both unigram and bigram statistics used a minimum frequency of 5 for vocabulary items. However, the value chosen did not necessarily affect the bigram and unigram models in the same way. With further investigation it may be possible to find an optimum value for each model. The smoothing constant was chosen to be 1 (add-one smoothing) for this work, but it is possible that changing this constant may improve or degrade the performance of the model.

Word count: 1073 words

References

[1] Bo Pang, Lillian Lee, and Shivakumar Vaithyanathan. Thumbs up?: sentiment classification using machine learning techniques. In *Proceedings of the ACL-02 conference on Empirical methods in natural language processing-Volume 10*, pages 79–86. Association for Computational Linguistics, 2002.