Predicting user ratings based on corresponding text reviews

Paul Disbeschl (i6075658), Timothy Smeets (i6071641)

May 28, 2019

Abstract

Much research has been done into binary sentiment analysis of IMDB reviews. IMDB reviews have a rating scale of 1-10, which indicates that there are more than simply 'negative' and 'positive' reviews. This paper shows two different approaches to analyzing user written IMDB reviews and shows three different data groups on which the classifiers are trained in order to predict the final user score as close as possible. We show that as more classes are trained and tested on, the Naive Bayes classifier outperforms the Linear Support Vector Machine classifier, even when Term frequency- inverse document frequency and N-gram approaches are applied.

1 Introduction

Sentiment analysis in the field of natural language processing is the extraction of feeling from a text. As feeling is a fundamental part of understanding natural language, a lot of research has been done in this area. A popular field of sentiment analysis is sentiment classification, mainly binary classification[1]. While a lot of research has been done to determine whether a text is positive or negative, less research has been done to determine the level of sentiment. One of the aims of this paper is to discover to what detail it is possible to predict a specific degree of sentiment.
The bag-of-words model is one of the simplest models for text analysis in natural language processing. In itself, it is not always the best technique, however, many additions may improve it such as n-grams, tf-idf, and appropriate pre-processing. The effectiveness of these techniques depends on the classifier employed as well; a Support Vector Machine classifier may perform completely differently from these techniques compared to a Naive Bayes classifier. It is the effect of these techniques on these classifiers which is also one of the aims of this paper. All in all, the question is how effective different implementations of the bag-of-words model in combination with the Naive Bayes and Linear Support Vector Machine classifiers are in predicting IMDB movie review sentiment scores from their associated text reviews.

2 Prior Literature

In a paper published in 2012 [5], Wang et al. described the use of Naive Bayes and Support Vector Machines techniques for sentiment analysis. These techniques, while relatively simple, were shown to be quite effective in performing sentiment analysis. Using IMDB reviews for sentiment analysis has proven to be successful in research.[3] [4] [2]

3 Data

The data-set used is the SAR14 data-set, which contains 234,000 written IMDB user reviews and accompanying ratings on a 1-10 scale. Specifically, it contains around 167,000 positive reviews, which are defined as having a score of greater than or equal to 7, and around 66.000 negative reviews, which are defined by having a score of 4 or lower. While the ratings are from a 1-10 scale, reviews with an associated score of of 5 or 6 are not included. The SAR14 dataset is free and for non-commercial use. It was created by Dai Quoc Nguyen, Dat Quoc Nguyen, Thanh Vu and Son Bao Pham [3] for their research on sentiment classification.

4 Model

The data-set consists of user reviews written in natural language to inform other users of their opinion. Some reviews are made up of many long and eloquent sentences while others do not contain more than a few words. A method through which these reviews can be compared is through the bag-of-words (BOW) technique, used in this model. This approach consists of creating a table that contains every unique word in the word sequences. After putting the data through a range of pre-processing and BOW vectorization techniques, the data is subjected to different classifiers to find out which techniques are best for the goal of finding the better technique for BOW sentiment analysis. Before the data is pre-processed, the data is split into random training and test sets, with an 80-20 split respectively. Since it is entirely possible for the split to happen unevenly, the data is stratified to ensure that the proportions of the scores (and hence, text) of the reviews in the test data are the same as in the training data.

4.1 Pre-processing

This is why a bag-of-words model was chosen, as it allows and pre-processing was applied in order to extract the useful data from the data-set. The data-set includes "LRB" and "RRB" tags, respectively indicating left and right round brackets, however these are removed in order not to be seen as words. Following this, all punctuation is removed and the text is set to lower case; this is done to reduce the vocabulary size.

4.2 Vectorization

Using the BOW, four approaches are used for the vectorization of the text after stop words have been removed. These approaches are pure BOW, BOW with n-grams, BOW with Tf-Idf, and BOW with both n-grams and Tf-Idf. N-grams are implemented to make distinctions between for example "not", "good" and "not good". The bigram "not good" is generally more apparent in negative reviews and so can be distinguished from the unigram "good" on itself which appears in both positive and negative reviews. In this model unigrams and bigrams are used; trigrams, four-grams and beyond were not found to provide a significant improvement (if any) to the model in initial testing and hence are not used.
Tf-Idf is a useful statistic which provides a weighting to terms in the document; frequent terms like "movie" get a low weight, while less frequent terms like "godawful" get a high weight. Since a pre-trained sentiment vocabulary is not used, Tf-Idf's ability to give certain terms large weights can be advantageous for words like "fantabulous", which is exclusively used in positive reviews. A possible drawback to this approach is that other terms such as certain cast and crew members' names, which may be assigned a high weight, appear in both very positive and very negative reviews.

4.3 Classifier/regression

A Multinomial Naive Bayes is implemented through MultinomialNB from sklearn. For the Support Vector Machine, LinearSVC, also from sklearn, is used.

5 Results

Both the Naive Bayes and Linear Support Vector Machine classifiers were tested on the dataset with different configurations; both were tested by themselves, with an tf-idf implementation, with an N-gram implementation and with both implemented at the same time. These tests were trained on three different criteria of the dataset:

1. A version where all reviews are classified as 'positive' (a score of larger or equal to 7) or 'negative' (a score of smaller or equal to 4)
2. A version where the scores are grouped in groups of 2, in order to reduce the amount of classes of the score column. The groups are based on the corresponding scores of each review. There are a total of 4 classes, which are grouped as follows:

   A score of 1-2: group 1

   A score of 3-4: group 2

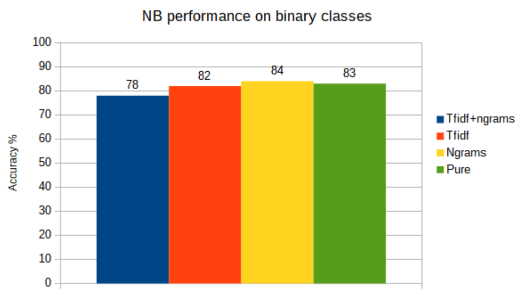
   A score of 7-8: group 3

   A score of 9-10: group 4 This effectively halves the total amount of classes, which should increase the accuracy of the classifiers compared to having the total amount of classes.
3. A version where the classifiers are trained on the scores themselves, so a total of 8 classes. It should be noted that the model is quite strict, as there is no difference between mislabeling a review by one point or by several points.

5.1 Naive Bayes

On the binary classes, NB overall has a performance that does not go below 78%. There is no relevant performance gain from adding either Tf-idf or N-grams implementations. If both are added, the performance worsens by an average of 4 to 5 percent, as shown in figure 1.

Figure 1: Naive Bayes performance on the binary classes



On the grouped score classes, performance worsens in every case except for when only the N-gram implementation is added, as is shown in figure 2. Adding the N-gram implementation greatly improves the result, not only relevant to the other variations of the NB classifier, but also compared to all other variations on both the NB and SVM classifier on all training classes.

For all values of the score classes, performance greatly shifts from the N-grams implementation to the Tf-idf implementation, as shown in figure 3. N-gram performance drops by about 60%, while the Tf-idf implementation increases by about 20%. Both the pure and the variation where both Tf-idf and N-grams are implemented drop in performance by a significant amount.

Figure 2: Naive Bayes performance on the grouped score classes

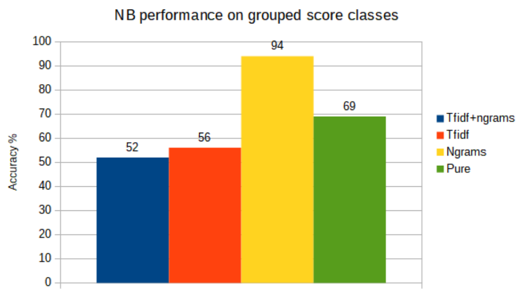
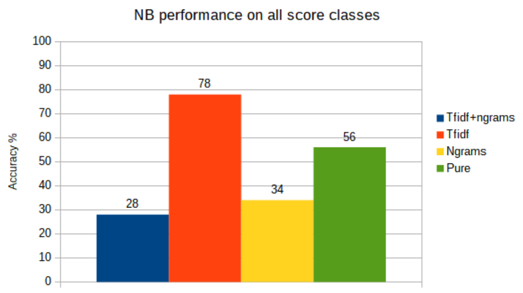


Figure 3: Naive Bayes performance on all score classes

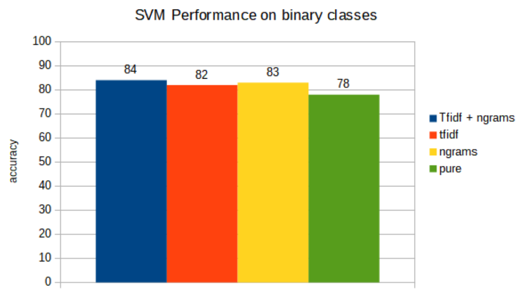


5.2 Linear Support Vector Machine

The SVM classifier performed worse than the Naive Bayes when both did not incorporate Tf-idf and N-grams for all the different criteria of the training sets. However, both the Tf-idf and the N-grams implementation showed performance gains when implemented by themselves as well as together.

On the binary classes, the SVM classifier gives a relatively even performance across all implementations as shown in figure 4. Implementing either Tf-idf or N-grams or both shows a small performance increase and overall these results are similar to the NB classifier.

Figure 4: SVM Bayes performance on binary classes



This pattern is similar for both the grouped score classes and all score classes, as for both of the class criteria performance shows that implementing either Tf-idf or N-grams or both increases performance. However, the increase of the amount of classes shows a clear performance drop across the board, dropping lowest when the classifier is tested on the maximum number of classes as shown in figures 5 and 6.

Figure 5: SVM performance on grouped score classes

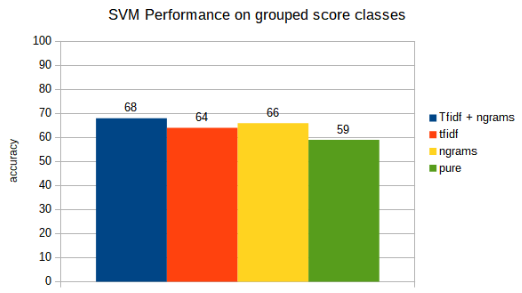
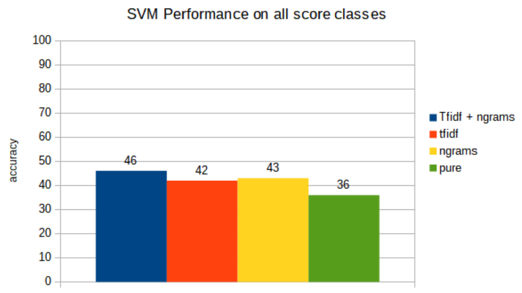


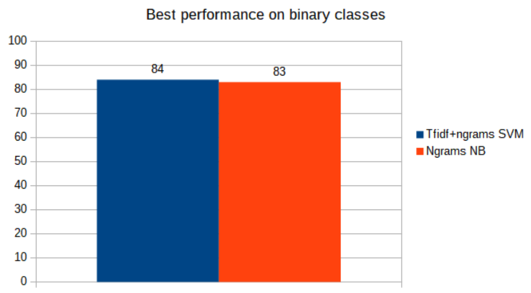
Figure 6: SVM performance on all score classes



6 Analysis

For the binary classes, the SVM classifier manages to beat the NB classifier when both Tf-idf and N-grams are implemented. Overall, the SVM classifier shows the most consistent increase in performance when either Tf-idf or N-grams or both are implemented.

Figure 7: Best performance on binary classes



A much larger disparity starts to show itself when the amount of classes increases. The NB classifier is able to outperform the SVM classifier by more than 20% when comparing their best scores with their respective variations as shown in figure 8. The more classes there are, the more the SVM classifier struggles. In figure 9, the best score the SVM classifier was able to manage was an accuracy of 46%, whilst the NB classifier still had a very respectable accuracy, even though the performance did drop compared to the grouped classes data. Interestingly, while the N-grams implementation of NB performed best with the grouped classes, the Tf-idf version of NB performed better when tested on all score classes. A reason for this could be because the data and features cause convolution when applied to a certain amount of classes when combined with either Tf-idf or N-grams. This convolution is also the reason why the NB performance drops when both Tf-idf and N-grams are implemented at the same time.

Figure 8: Best performance on grouped score classes

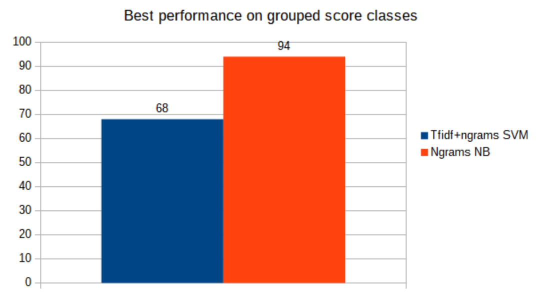
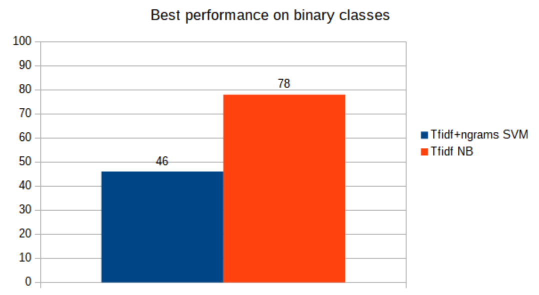


Figure 9: Best performance on all score classes



7 Conclusion

As the amount of classes increases, the Naive Bayes classifier starts to outperform the Linear Support Vector Machine by a relevant difference in accuracy score. Implementing either Tf-idf, N-grams or both at the same time showed consistent performance gains for the SVM classifier. This is not the case for the NB classifier, as implementing both shows a consistent performance drop compared to all other approaches. On binary classes, the SVM classifier is definitely able to compete and even outperform the NB classifier. All in all, the BOW model in combination with the Naive Bayes classifier is able to get a score which is comparable to the binary classes. Many more classifiers exist and different approaches could still be tested on this subject. Performance could perhaps be gained not only by testing other classifiers, but also by using a different vectorization technique such as word2vec instead of the BOW model.

References

[1] Dmitriy Bespalov et al. "Sentiment classification based on supervised latent n-gram analysis". In: *Proceedings of the 20th ACM international conference on Information and knowledge management.* ACM. 2011, pp. 375–382.

[2] Andrew L Maas et al. "Learning word vectors for sentiment analysis". In: *Proceedings of the 49th annual meeting of the association for computational linguistics: Human language technologies-volume 1.* Association for Computational Linguistics. 2011, pp. 142–150.

[3] Dai Quoc Nguyen et al. "Sentiment Classification on Polarity Reviews: An Empirical Study Using Rating-based Features". In: *Proceedings of the 5th Workshop on Computational Approaches to Subjectivity, Sentiment and Social Media Analysis.* 2014, pp. 128–135.

[4] Andrei Oghina et al. "Predicting imdb movie ratings using social media". In: *European Conference on Information Retrieval.* Springer. 2012, pp. 503–507.

[5] Sida Wang and Christopher D Manning. "Baselines and bigrams: Simple, good sentiment and topic classification". In: *Proceedings of the 50th annual meeting of the association for computational linguistics: Short papers-volume 2.* Association for Computational Linguistics. 2012, pp. 90–94.