Mini-project 2

Predicting the sentiment of an IMDb review

Hassan Haidar (260711061), Ahmed Hegazi (260725697), Saifullah Elsayed (260733168)

February 23, 2019

Abstract

We set out to predict the sentiment of IMdb reviews using four models which are Bernoulli Naïve Bayes, Logistic Regression, Decision Tree, and Linear Support Vector Classification. In addition to this we attempted to use the different feature extraction methods and searching for the most effective ones that improve the predictive performance of each model. Our best performing model on the test set was a linear Support Vector Classification (SVC). This model exhibited a precision = 0.90 when using the word count of the top 1000 most common recurring words in the dataset. We found that in general that the Bernoulli Naïve Bayes model did not have good precision relative to the other tested models. This is mainly because it accepts binary features. Finally, after using learning curves to check whether more data is required for better performance, it was found that using 1000 features was sufficient.

1 Introduction

IMDb is a popular website and online database of information related to television entertainment and games. The website allows the users to post reviews and give their ratings of the quality of the product (movies, video games, etc.). We aimed to classify the sentiment of posted reviews, whether they are positive or negative, based on their language, by training the four models on a set of reviews with known ratings which implied the sentiment of the review. As for the features, we used multiple feature extraction methods on the training set of reviews, after applying some natural language processing on its text. We will briefly talk about previous literature for sentiment analysis and how we incorporated some of their theories and methods. Moreover, we will explain how our dataset was treated before discussing the performance and comparison of the three models for classifying the sentiment of the review, as well as discussing the feature extraction methods and optimization done to each model. Finally, we will end with a discussion that analyzes these results in more detail.

2 Related Work

In an attempt to understand the project requirements, we explored further material that aided in understanding the problem we tackled. In "Predicting IMDB Movie Ratings Using Social Media" the author explores the prediction of the rating of a movie using social media reception.[1] Comments from various sources (Facebook, Twitter, etc.) that reflect the reception of a movie are taken and used to predict the rating of the movie. The best performing model was a linear regression model that takes in the ratio of likes and dislikes on Youtube videos as well as the textual features from Twitter. This has lead us to understanding that a linear model would be the best approach to tackling the project.

3 Dataset

The dataset was constructed from a list of IMDb reviews where each instance is composed of a text, and the sentiment category of the text i.e. positive or negative. Each instance's

text was pre-processed by striping its capitalization and splitting it into a list of words by white space. After, the top N words most common words in the training set were found. The number of occurrences for each of these words were counted for each element, creating N new features, one for each top occurring word, for every data point.[2] However, different feature extraction methods were used for each model since their input types were different.For example, Bernoulli Naive Bayes only accepts binary features; as such, word count feature extraction method was only used. However, for the other models we utilized more complex feature extraction methods, as discussed in the next section.For all the models discussed below, we split the provided training raw data by using 80 percent (20000 instances) for training, and the other 20 percent (5000) for validation. Moreover, we test every model on a test set containing 25000 thousand instances.

4 Approach

To classify the sentiment of IMDB reviews, we develop four different models. The first model we develop is the Bernoulli Naive Bayes classifier, which we implemented from scratch. For this classifier, we extract features by taking the top N-most frequently appearing words in the whole data-set. Then, each text document is modeled as a binary feature vector of size N, where the *i-th* entry in the vector represents the *i-th* entry in our vocabulary list.

The three other models we develop utilize the SciKit-learn package.[2] The first model we used is the Logistic Regression classifier, which models the log-odds ratio of the probabilities of the categories using a linear function. This linear function is a hyper-plane that separates the two classes. The second model we used is the Decision Tree, which provides a richer partitions of the input space i.e, more than one hyper-plane. The third and final model we used is the Support Vector Machine, which is non-probabilistic binary linear classifier. For feature extraction for these models, we vectorize each document by finding its "Bag of Words" (BoW). From the bag of words, we extract the document's *tf-idf*, which encapsulates the relative frequency of every word. Both BoW and tf-idf are found by methods implemented in the Sci-kit learn package.[2]

5 Results

	Number of Features	Avg. Runtime (s)	Avg. Percision
Bernoulli NB	1000	0.92	0.80
	2000	1.65	0.79
	3000	2.34	0.80
Logistic Regression	1000	3.61	0.86
	2000	3.70	0.88
	3000	3.71	0.89
Decision Tree	1000	1.90	0.70
	2000	2.13	0.71
	3000	2.37	0.70.
Linear SVC	1000	2.18	0.90
	2000	2.31	0.88
	3000	2.48	0.88

Table 1: ***shows the runtime of the models and the precision of the validation set.***

As shown in the above table and the graph below, the quickest model was the Bernoulli NB but at the cost of its precision. The model that required the longest time was the logistic regression model but it output precise predictions and so was a candidate for the best performing model. The linear SVC model was our best performing model as it had the highest precision for the shortest running time. This was hypothesized after researching similar published works as most of which used a linear model to predict popularity of comments/movies.[3] The predictions of the Linear SVC model on the test set were compared to the actual dataset and scored 0.87480.

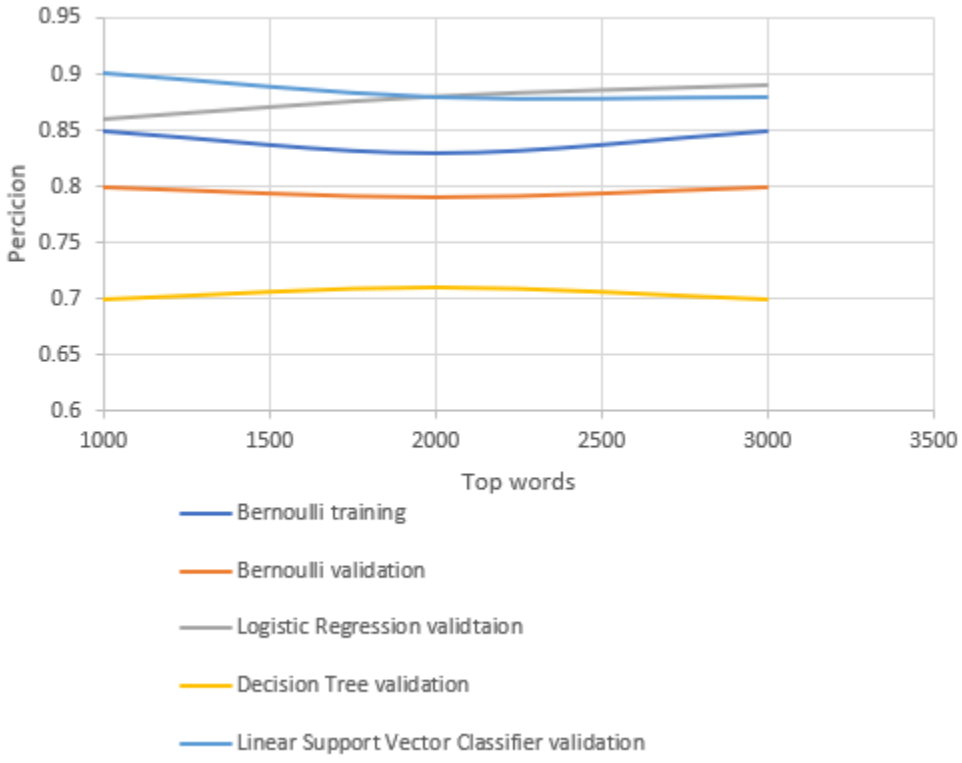

Figure 1: ***shows the precision of all 4 models as the number of features extracted increases***

6 Discussion and Conclusion

In conclusion, our best performing model was the Linear SVC model. This model is similar to SVC with parameter kernel='linear', but implemented in terms of liblinear rather than libsvm, and so it has more pliability in the choice of penalties and loss functions and should be able to handle larger dataset with ease. Linear SVC is used for text classification tasks such as category assignment, detecting spam and sentiment analysis.It is likewise regularly utilized for Image recognition challenges, performing especially well in aspect-based recognition and color-based classification. Linear SVC likewise assumes a fundamental job in numerous territories of handwritten digit recognition, for example, postal automation services. We plan on improving the pre-processing of the text by considering n-grams of the text (quick-brown-fox, brown-quick-fox, etc.) as well as prefixes and suffixes of words (perplex, perplex-es, perplex-ing, etc.). This should allow for a better prediction as the total number

of features would decrease but the features would have a higher importance and so leading to more precise results.

7 Statement of Contributions

Hassan Haidar wrote the code for feature extraction and to pre-process the data, Saifullah Elsayed wrote the code for the Bernoulli Naïve Byes class. Ahmed Hegazi implemented the other three models and ran the tests with the help of Saifullah Elsayed for the completed code and models. All three worked on and edited the report together.

8 Citations

[1] Oghina, A. (2012). Predicting IMDB Movie Ratings Using Social Media. Amsterdam: Springer-Verlag Berlin Heidelberg.

[2] Bird, Steven, Edward Loper and Ewan Klein (2009).Natural Language Processing with Python. O'Reilly Media Inc.

[3] Scikit-learn: Machine Learning in Python, Pedregosa et al., JMLR 12, pp. 2825-2830, 2011.