COMP-551 Mini-Project 2

COMP-551 Winter 2021

Arneet Singh Kalra, Behnaz Ghefati Feizabadi, Hussein Lakkis

February 28, 2021

Abstract

The importance of machine learning algorithms in the field of Natural Language Processing and Text Classification is becoming more prominent now than ever. In this project, we investigated the performance of two classification models on two benchmark datasets: the 20 News Group dataset as well as the IMDB Reviews set. More precisely, we implemented Naive Bayes (a generative model) from scratch and used Logistic Regression (a discriminative model) from the scikit-learn package. Furthermore, we implemented our own custom cross-validation to conduct hyper-parameter tuning to choose best values for our models. The datasets were also additionally tested using Scikit Learn's SGDC, Decision Tree, and Ada Boost classifiers. From our analysis, we conclude that for the 20 News Group dataset, the logistic regression approach achieved worse accuracy than Naive Bayes and was also significantly slower to train when compared to Naive Bayes. However, on the IMDB dataset, logistic regression was more accurate and faster to train.

1 Introduction

1.1 Project Background and Goals

The goal of this project is to implement and compare two text classification algorithms (Naive Bayes and Logistic Regression), using two well-known datasets, the Twenty News Group and IMDB Reviews. In addition to these models, we also chose to test them using Scikit Learn's SGDC (SVM), Binary Decision Tree, and Ada Boost classifiers. The 20 News Group Dataset is provided by Sklearn and contains textual data from newsgroup documents across twenty different newsgroups. The dataset itself was originally collected by Ken Lang for his paper, "Newsweeder: Learning to filter netnews", however it was never referenced in it [1]. The IMDB dataset contains data for movie reviews with ratings. The dataset was originally collected by a group of researchers at Stanford University to be used in their paper "Learning Word Vectors for Sentiment Analysis" published in 2011 [2]. We chose these datasets because they are well known benchmarking datasets. After the modeling, and based on the results of our analyses for both datasets, we conclude that Multinomial Naive Bayes is more accurate for the 20 News Group dataset. It is also almost 10 times faster to train than logistic regression. On the other hand, the IMDB dataset received the most accurate result from Logistic Regression, and was also faster using that model than Naive Bayes.

1.2 Discussion of Related Work

Our research on these datasets and modeling techniques led us to see various other applications as well as other algorithms for this task. A paper by Verma and Yenter describes their use of Convolutional Neural Networks and Long Short-term Memory Layers to do textual sentiment classification on the IMDB dataset. Their novel model boasts accuracies above 89%, higher than our more accurate result, and can be applied on other datasets for sentimental analysis or text classification [5]. Other sources showcase the wide range of real-life applications of using such modelling techniques. For example, Naive Bayes is commonly used

as a filter to detect spam emails. A paper by Greek researches describes their work in using the dataset and developing various versions of the Naive Bayes models to increase the overall accuracy of detecting spam [3]. Another paper from India presents the use of such techniques to determine the overall sentiment of a tweet posted by a Twitter user. Their research uses these models to predict the general sentiment based off words often used in tweets [4].

2 Datasets

2.1 TFIDF Vectorizer

For both datasets, we used Scikit-Learn TFIDF Vectorizer to convert the textual data found in the sets to numerical features. This vectorizer specifically places less importance to words that appear often among all documents. We used the built-in "stop words" parameter of the vectorizer and set it to 'English' so that it ignores common words that are often not useful such as "and", "the", and "or". It also allows us to drop features that are noisy and lead to overfitting.

2.2 20 News Group

The 20 News Group dataset is a collection of approximately 20,000 newsgroup documents, partitioned fairly evenly across 20 different newsgroups, some of which are closely related while others have low correlation. We performed multi-class classification on the set to predict which of the 20 news groups is associated with a respective news document. While pre-processing the data, we first removed the header, footer, and quote features from the set to limit overfitting. Then we removed all punctuation, capital letters and numbers, and empty lines from the text documents.We also lower cased all words. After this, we used the TFIDF Vectorizer to convert the text features to numerical data. For this dataset, we set the vectorizer to exclude words that appear in less than 3 documents as it increases our accuracy. Our analysis showed that setting it to 3 resulted in a better result than the default of 1 document. Figures 1 and 2 represent the class distribution analysis of our features for the training and testing sets respectively. There is no significant imbalance between the different classes for both sets, which is beneficial to the accuracy of our model.

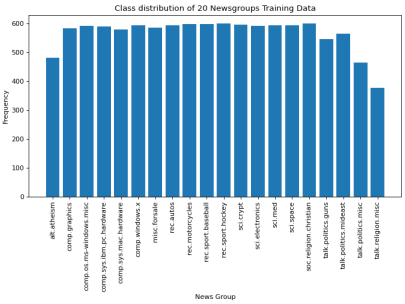

Figure 1: News Training Set Class Distribution

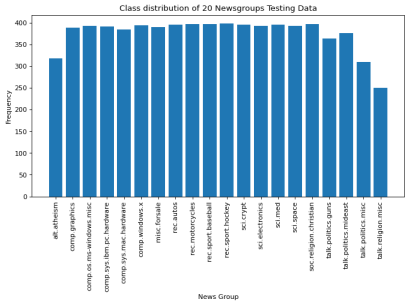

Figure 2: News Testing Set Class Distribution

2.3 IMDB Reviews

The IMDB Reviews dataset contains 50 000 reviews split evenly into training and testing sets. Alongside the reviews, a rating on 10 is given for the movie. This rating is translated to a positive class if it is between 7-10, or a negative class if it is below 5. A rating of 5 or 6 is ignored for the purposes of the analysis. We performed binary classification on the dataset to predict if a given review results in a positive or negative rating. During the preprocessing of the data, we removed text within '<>' brackets as well as any emoticons. We then used the TFIDF Vectorizer as outlined in 2.1. For this dataset, we limited the number of features to a max of 10 000 as it helped improve our accuracy and making modeling feasible due to the

limited resources. In terms of classes, the set contains 25 000 positive reviews and 25 000 negative reviews, therefore the class distribution is evenly balanced.

3 Results

3.1 Uncleaned Set versus Cleaned Set

We compared our models on the cleaned version of our sets with the uncleaned version to see how our preprocessing of the data affected our accuracy and training time. Table 1 highlights our findings. It is important to note that for the testing of Naive Bayes on the IMDB reviews uncleaned dataset, we used SciKit Learn's Naive Bayes model due to RAM constraints. Their Naive Bayes model performs the same as ours in terms of accuracy on this set, albeit their implementation is faster to run. Furthermore, we were unable to run logistic regression on the completely uncleaned news set due to RAM constraints, so we set a limit of 10 000 on the number of features to test its performance on unaltered data*.

For both datasets, we saw a significant increase in the testing accuracy on the cleaned version. The testset accuracy for the curated News set was approximately 5% and 10% higher with Naive Bayes and Logistic Regression than the uncleaned dataset respectively. The IMDB set received over 30% more accurate test set results. Furthermore, we also observe a training time that is significantly lower in the cleaned datasets. For example, while using the Logistic Regression classifier on the IMDB Reviews dataset, the training time for the cleaned version was 193% less than the uncleaned set. As a trend, we see varying levels of overfitting and lower test accuracy when using the uncleaned dataset. In the uncleaned Logistic Regression models, we observe that the testing accuracies are over 37% less than the training accuracies for the IMDB dataset.

Classifier	Dataset	Best Train Accuracy (%)	Best Test Accuracy (%)	Time (s)
Naive Bayes	Uncleaned News	86.04	64.92	173.70
Naive Bayes	Cleaned News	94.03	69.20	17.56
Logistic Regression	Uncleaned News*	96.86	58.14	600.01
Logistic Regression	Cleaned News	74.54	67.30	182.60
Naive Bayes	Uncleaned IMDB	89.81	51.29	12.83
Naive Bayes	Cleaned IMDB	88.19	83.65	5.66
Logistic Regression	Uncleaned IMDB	99.83	50.27	154.20
Logistic Regression	Cleaned IMDB	92.66	87.95	2.57

Table 1: Comparing Various Models with Best Accuracies on both datasets

3.2 Naive Bayes vs Logistic Regression

We then performed multiclass classification on the 20 News Group dataset using cleaned dataset with the selected features. Our results concluded that Naive Bayes worked slightly better than logistic regression. In the best case scenario, the logistic regression resulted in 67.30% testing accuracy. It was also almost 10 times slower than Naive Bayes, with a training time of 182.60 seconds. We chose to compare our implementation of Naive Bayes with the Scikit Learn's implementation. Ours performed slightly better as their model resulted in an accuracy of 67.75% vs 69.20%. It is also interesting to note that our Naive Bayes model is 3x slower than Scikit Learn's implementation because their model is parallelized and therefore can run much faster. Our implementation is vectorized and fast but not to the same extent.

Our binary classification on the IMDB Reviews dataset concluded that Logistic Regression was more accurate than Naive Bayes. In the best case scenario, the Naive Bayes received a testing accuracy of 83.65%, whereas logistic regression resulted in 87.95%. Naive Bayes was slower in this case, taking 5.66 seconds for training as opposed to Logistic Regression which took 2.57 seconds.

As we can conclude, our classifiers perform much better on the IMDB Reviews dataset than they do on the 20 News Group dataset. Naive Bayes performs 14.45% better on IMDB, and Logistic Regression performs 20.65% better. They are also significantly faster on this dataset. This is perhaps due to the binary nature of the IMDB dataset.

3.3 Hyper Parameter Tuning

To tune our model, we used 5-fold cross validation. In Figure 3, we plot the varying alpha values (smoothing constant) for Naive Bayes against their respective accuracies on the 20 News Group dataset. We used alphas ranging from 0.01 to 4, with the majority of out test alphas being between 0.01 and 1. We receive an optimal result for alpha = 0.05. Similarly, we performed hyperparameter tuning for the Logistic Regression model. For our hyperparameters, we looked at solvers (newtown-cg, lbfgs, liblinear), penalty (l1,l2), and C values (0.01,0.1,1.0,10,100). We achieved the optimal accuracy using a liblinear solver, a C value of 10, and a penalty of l2.

Figure 4 plots the alpha values with their respective accuracies for the IMDB Reviews dataset. We achieve an optimal accuracy for alpha = 1.3. For logistic regression, we used Grid Search CV from Scikit-learn with their cross validation method because we struggled with memory issues on this dataset with our cross-validation. The hyperparameters we tested are the same as for the 20 News Group data set using the custom cross validation function. We achieved optimal accuracy with C = 1, a liblinear solver, and penalty = l2.

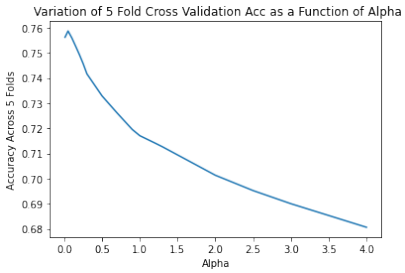

Figure 3: 20 News Group Naive Bayes Hyperparameter Tuning

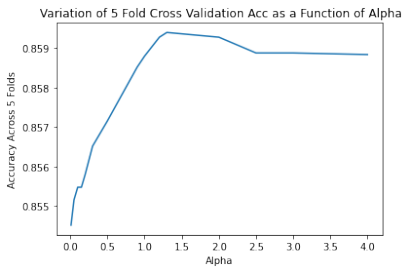

Figure 4: IMDB Naive Bayes Hyperparameter Tuning

3.4 Accuracy Depending on Dataset Size

We further compared the accuracy of the two models as a function of the size of the dataset by arbitrarily selecting 20% intervals of the available training data to train our model on the 20 News Group set. Figures 5 and 6 plot our accuracies on the training set and test set respectively. The Naive Bayes performs consistently better on the testing set. Interestingly, logistic regression performs just as well and better than Naive Bayes for the testing set.

Similarly, we ran this experiment for the IMDB Reviews set. Figures 7 and 8 plot our accuracies on the training set and test set respectively. As expected, we achieve higher accuracy as we train more of the dataset. Furthermore, Logistic Regression is consistently more accurate especially for a larger dataset as it is a discriminative algorithm.

3.5 Comparison with Other Models

Table 2 presents the models we used on both datasets. In addition to the Naive Bayes and Logistic Regression classifiers, we looked into SGDC (SVM since loss = hinge), Decision Trees, and Ada Boost to test how they would perform on our datasets. SVMs (Support Vector Machines) are some of the best natural language processing tools in the industry, especially the ones with linear kernels. Therefore, we suspected it would perform well on our sets. Furthermore, we previously used the decision tree classifier and found it effective,

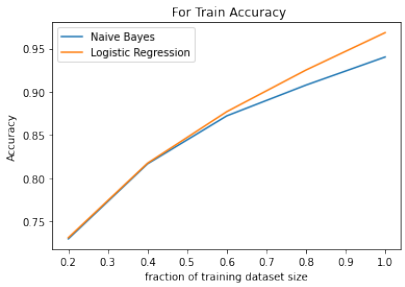

Figure 5: News Training Accuracy Based on Training Set Split

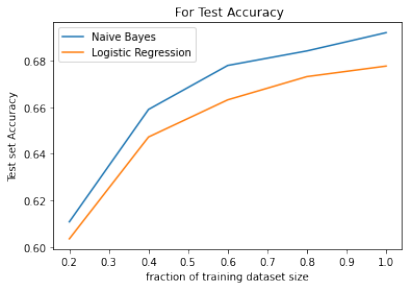

Figure 6: News Testing Accuracy Based on Training Set Split

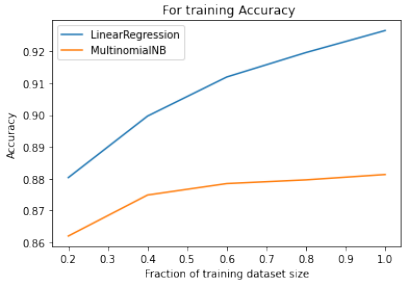

Figure 7: IMDB Training Accuracy Based on Training Set Split

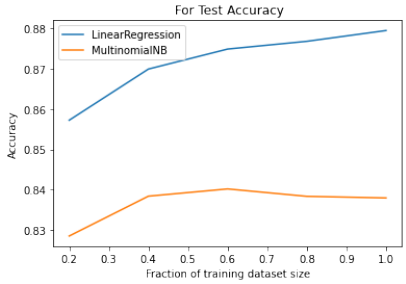

Figure 8: IMDB Testing Accuracy Based on Training Set Split

so we tested it as well. We conclude that for the 20 News Group dataset, our Naive Bayes model still performs the best, whereas SGDC is slightly better than logistic regression. Even more, it is the fastest classifier for training. The decision tree classifier has the highest training accuracy (most overfit). On the other hand, for the IMDB dataset, the Logistic Regression model is still the most accurate. It is also the fastest model for training on the dataset. As with the news dataset, the decision tree classifier once again has the highest training accuracy.

Classifier	Dataset	Best Train Accuracy (%)	Best Test Accuracy (%)	Time (s)
Naive Bayes	20 News Group	94.03	69.20	17.56
Logistic Regression		74.54	67.30	182.60
SGDC		89.33	67.90	6.89
Decision Tree		97.29	43.77	220.29
Ada Boost		41.42	36.36	241.15
Naive Bayes	IMDB Reviews	88.19	83.65	5.66
Logistic Regression		92.66	87.95	2.57
SGDC		86.09	84.26	16.69
Decision Tree		1.00	71.63	154.44
Ada Boost		80.67	80.16	366.64

Table 2: Comparing Various Models with Best Accuracies on both datasets

4 Discussion and Conclusion

Based on our models and analyses, we conclude that the 20 News Group and IMDB Reviews datasets require different classifiers for the most accurate predictions since one is binary and one is multiclass. First, both datasets must be cleaned and vectorized using a TFIDF vectorizer in order to get the most accurate results and to reduce the training time significantly. The uncleaned set contains a lot of noise and unnecessary features. Using the cleaned set, for the 20 News Group data set, we achieved the best

testing accuracy with our implementation of Multinomial Naive Bayes classifier. The logistic regression performed on this set being less accurate and 10 times slower than Naive Bayes. Interestingly, our extra research showed that the SGDC classifier is almost 3 times faster than Naive Bayes while being slightly less accurate. The IMDB Reviews dataset had the best testing accuracy based off of the Logistic Regression Classifier. This model also trained faster, almost twice as fast as Naive Bayes while being about 4% more accurate on the testing set.

5 Statement of Contributions

Arneet Singh Kalra wrote the maniuscript and assisted in developing K-fold cross validation. Behnaz Ghefati Feizabadi completed the analysis of the IMDB Reviews dataset. Hussein Lakkis developed the Naive Bayes model and did analysis of the 20 News Group dataset.

References

[1] Ken Lang. "Newsweeder: Learning to filter netnews." In: *Proceedings of the Twelfth International Conference on Machine Learning*. 1995, pp. 331–339.

[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*. Portland, Oregon, USA: Association for Computational Linguistics, June 2011, pp. 142–150. URL: http://www.aclweb.org/anthology/P11-1015.

[3] Vangelis Metsis, Ion Androutsopoulos, and Georgios Paliouras. "Spam filtering with naive bayes-which naive bayes?" In: *CEAS*. Vol. 17. Mountain View, CA. 2006, pp. 28–69.

[4] M Vadivukarassi, N Puviarasan, and P Aruna. "Sentimental analysis of tweets using Naive Bayes algorithm." In: *World Applied Sciences Journal* 35.1 (2017), pp. 54–59.

[5] Alec Yenter and Abhishek Verma. "Deep CNN-LSTM with combined kernels from multiple branches for IMDb review sentiment analysis." In: *2017 IEEE 8th Annual Ubiquitous Computing, Electronics and Mobile Communication Conference (UEMCON)*. IEEE. 2017, pp. 540–546.