Exploration of Naive Bayes and Softmax Regression on Two Text Datasets

Lambert Francis, Patrick Janulewicz, Kevin Sohn

Abstract

The objective of this project was to study the behaviour of two linear classifiers, Naive Bayes and logistic regression, on two text-based datasets. The first dataset, 20 Newsgroups, consists of newsgroup posts filtered based on topic into 20 different categories. The second dataset, Sentiment140, is a collection of tweets grouped into either a positive or negative sentiment category based on the emoticons present in the tweet. Both algorithms resulted in accuracies of around 70 percent for the Sentiment140 dataset, whereas the accuracy dropped to around 30 percent for the 20 Newsgroups dataset. We attribute this to the fact that the 20 Newsgroups dataset has 20 classes that vary wildly in terms of relatedness, whereas the Sentiment140 dataset has just two classes that combine to act as a single binary class. Hyperparameters were tuned using k-fold cross validation. We implemented different variants of both algorithms. The best performing variant for the Sentiment140 dataset was the logistic regression algorithm with a bias term and an L1 regularization penalty, which had an accuracy of 0.730. This variant also showed the best performance for the 20 Newsgroups dataset, with an accuracy of 0.331. Overall, the logistic regression variants performed much better than the Naive Bayes variants by a large margin.

Introduction

In this paper, we investigate the performance of two linear classifiers - Naive Bayes (NB) and logistic regression (LR) - on the problem of multiclass classification of two text-based datasets - 20 Newsgroups and Sentiment140. Although NB is generally not a linear classifier, the Gaussian and Discrete variants are linear, which are the variants we used.

Steps taken to process the data and tune the model were in part inspired by current literature. For instance, results from the Sentiment140 dataset were compared to a study by Behara et al [4], which obtained similar a ROC curve and accuracy. Furthermore, studies from Wang et al [3] and Gopal and Yang [5] studied the 20 Newsgroupd dataset and logistic regression. Our findings reinforce their studies in accuracy and parameter tuning.

The data was first preprocessed using Python's CountVectorizer and TfidfTransformer functions. The Sentiment140 dataset contained roughly 1.6 million training samples, while the 20 Newsgroups dataset was significantly smaller at a size of 18846. The data was loaded and used to train the two main algorithms: Naive Bayes and logistic regression. Several different modifications were made to the models. For instance, both Gaussian Naive Bayes and binary Naive Bayes were implemented from scratch and were used to train the data. Different values of the Laplace smoothing factor were also calculated using cross validation. However, even large differences in the value of the scaling factor showed almost zero difference in accuracy. As a result, the validation curves were deemed irrelevant and were therefore omitted from the report. Among the different models used, the binary Naive Bayes gave a greater accuracy for the Sentiment140 data, but extremely poor accuracy for the 20 Newsgroups data. The Gaussian Naive Bayes showed mediocre performance on the news classification at 0.180 accuracy and a somewhat high accuracy of 0.687 on the sentiment analysis. Two logistic regression algorithms were also studied: one from Python's Sklearn library and one implemented by us from scratch. Both models had comparable performances. Much like the Naive Bayes, different hyperparameters were also studied for the logistic regression model. The main parameters were bias and regularization. It was found that a bias term significantly improves the algorithm's performance. Regularization also added a small improvement, though the choice of L1, L2, or a hybrid penalty seemed to make little difference. The highest accuracy on the sentiment dataset was 0.730, and the highest accuracy on the news dataset was 0.331. Both occurred with a bias term and a L1 regularization. Cross validation on the learning rate found that the sentiment data could get to a learning rate as high as 4 before diverging. However, this is not true for the 20 Newsgroups data, which began losing accuracy for even small increases in the learning rate. Overall, it is expected that the sentiment will perform better than the news data. This is because sentiment is binary, while the news had 20 different categories to fit. All things considered, the accuracies of models were relatively high.

Datasets

The 20 Newsgroups dataset is a collection of 18846 newsgroup documents split evenly into 20 classes, with each class corresponding to a topic. Some examples include computer hardware, sports, and religion [2]. Some classes are inti-

mately related to each other (e.g. religion and atheism) while others are not (e.g. hockey and Mac hardware). The training data was taken to be a random sample comprising of 80% of the data, with the remaining comprising the test set.

The Sentiment140 dataset is a collection of tweets labeled into two classes; positive and negative or negative sentiment [1]. The dataset has 1.6 million training data and 358 test data.

Primary differences between the two datasets include their sizes in terms of available data, with Sentiment140 being vastly larger. Also, the text content itself likely differs in style as tweets and news group documents are typically not written in the same tone. Another consideration could be the length of each document. For Sentiment140 the average document was 74 characters long, compared to 1218 characters for the average 20 Newsgroups document.

To prepare the data, all text was transformed to downcase, then all punctuation and numbers were removed and replaced with white spaces. Finally, all words from *nltk*'s repository of english stopwords were removed in order to reduce noise.

Extracting numerical features from the documents was performed using *scikit-learn*'s *CountVectorizer*. A $m \times n$ matrix was produced where m is the number of documents and n is the number of words kept as features. An entry c_{ij} corresponds to the number of occurrences of word j in document i. To obtain binary features for the Bernoulli Likelihood Naive Bayes we apply the transformation

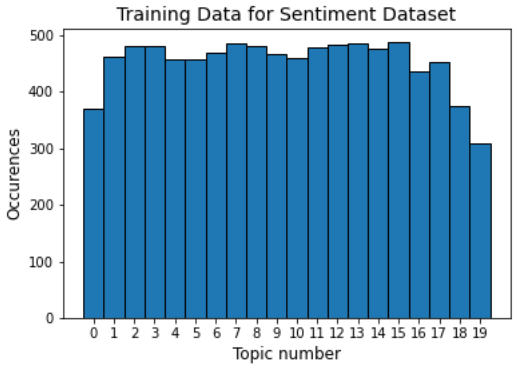
$$b_{ij} = \begin{cases} 0 & c_{ij} = 0 \\ 1 & c_{ij} \geq 1, \end{cases}$$

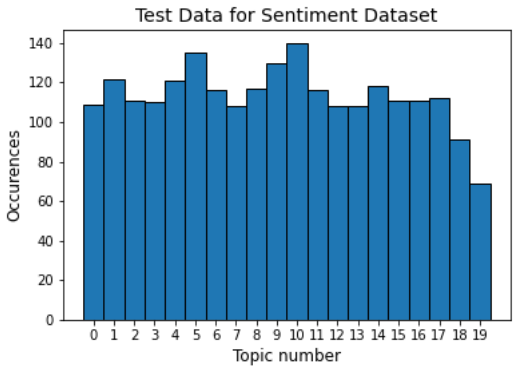
which gives us the feature matrix B saying whether a word was present in a document or not.

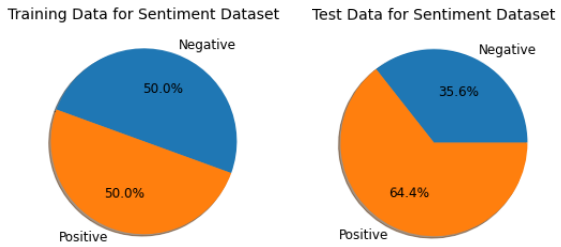
Finally, to get continuous data, *scikit-learn*'s *TfidfTransformer* was applied to the counts. Inverse document frequency gives the added benefit of weighting words that appear less commonly and hence reducing the importance of common words that give little information about the output class. This feature matrix was used for the Gaussian Naive Bayes and Logistic Regression models.

In each case, the number of words selected as features was limited to the top 300 most frequent. This choice was both to eliminate noise in the data and due to the limited computational power available.

In both datasets, the data was balanced in the number of classes as can be seen in the first two figures.







Results

Scikit-learn's Logistic Regression model was the best performer on both datasets, with an accuracies of 0.331 and 0.730 for 20 Newsgroups and Sentiment140, respectively. Notably the L1 penalty and inclusion of bias yielded the best results for both.

Examining the ROC plots shown in Figure 1 for the two best models shows the Logistic Regression behaving normally with a good shape and AUC of 0.79. However, the Gaussian Naive Bayes clusters around a specific false positive rate.

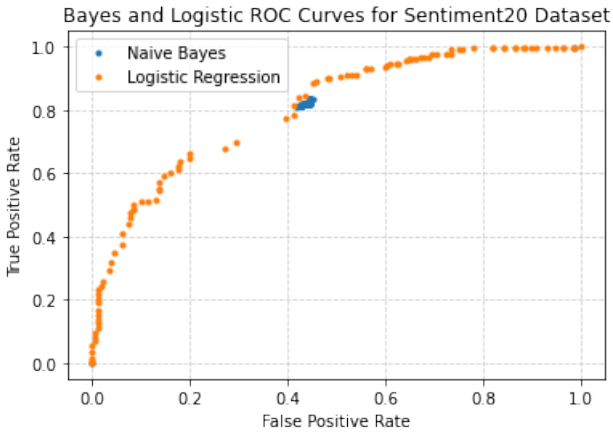


Figure 1: ROC plot for the Sentiment20 dataset for the Naive Bayes and Logistic Regression models. Logistic AUC = 0.79

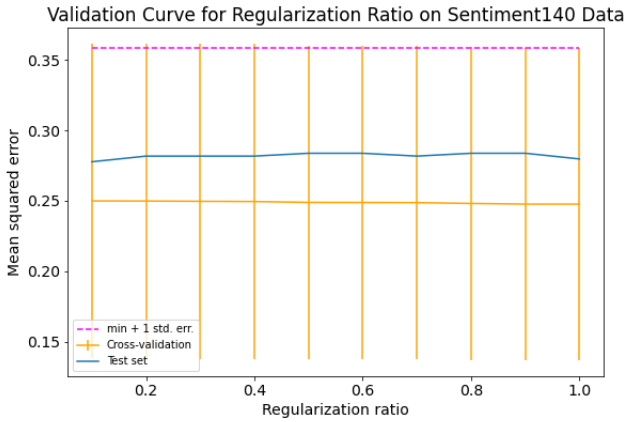


Figure 2: Validation curve for regularization ratios varying from 0 to 1 in increments of 0.1 for Sentiment dataset. Virtually no fluctuations occur.

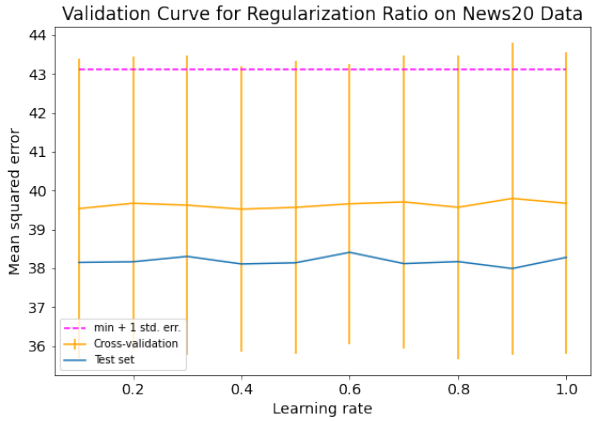


Figure 3: Validation curve for regularization ratios varying from 0 to 1 in increments of 0.1 for 20 Newsgroups dataset. Very slight fluctuations occur.

The likelihoods that performed best for Naive Bayes were Bernoulli for the Sentiment140 dataset and Gaussian for the 20 Newsgroups dataset. We notice that the Bernoulli Naive Bayes performed very poorly for 20 Newsgroups. This is possibly because of the larger size of the text, making it much more likely for many words to be present even if they don't contribute to the topic of the document; Whereas for tweets, given the limited length, it is much more liekly that an individual word being present is relevant to the topic of the tweet.

For logistic regression, both Sklearn's logistic regression model and our own model implemented from scratch were tested. The model from scratch used standard gradient descent methods and no regularization. Both models had similar performances when given similar parameters; our model performed slightly better on the 20 Newsgroups dataset, while Sklearn's model performed slighlty better on the Sentiment140 dataset. However, these errors are small, and there is insufficient evidence to declare that one model had superior performance.

Furthermore, different hyperparameters were modified to study their effects on accuracy. For the softmax regression, the bias and regularization penalty were the main hyperparameters that were altered. Both the Sklearn's model and our own model were tested with and without a bias term. It was found that including a bias term consistently improved the accuracy. This was particularly true for the Sentiment140 dataset, which saw increases in performance of roughly 10%. A regularization penalty was also introduced and studied for Sklearn's model. The *elasticnet* method was used, which takes a parameter between 0 and 1 and introduces a mixed regularization. In this method, a parameter of 0 is purely L1 and a parameter of 1 is purely L2. In most cases, the regularization term provided a small improvement on the model's performance, though the choice of the *elasticnet* coefficient did not make a noticeable difference. This can be seen in the validation curves ahead.

In an additional experiment the maximum number of features was lowered to 50 to see the effect on the accuracy of the models for the 20 Newsgroups dataset. The Gaussian Naive Bayes and L1 Bias Logistic Regression (Logistic 5) dropped to 0.087 and 0.155, respectively, cutting the accuracies in half. This suggests that playing with the expressiveness of the model by further increasing our maximum number of features could be worthwhile. This could be done either by further optimizing the code or increasing our available computational power.

The final hyperparameter studied for softmax regression was the learning rate. Because Sklearn's model does not support changes in the learning rate, this was performed on our own model. Specifically, we were interested in the maximum size of the learning rate such that the gradient descent algorithm did not overshoot. While a smaller learning rate leads to greater accuracy, it can also greatly increase the al-

gorithm's runtime. Due to the large scale of the data, it was deemed useful to find the largest possible rate that did not hurt accuracy. As a result, validation curves were plotted for learning rates between 0.1 (the default setting) and 10. These curves can be found below.

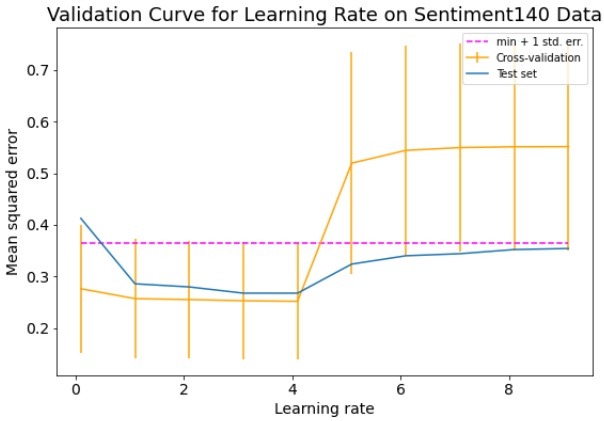


Figure 4: Validation curve for learning rate on Sentiment140 dataset. Note that the mean squared error only increases significantly around a learning rate of 4.

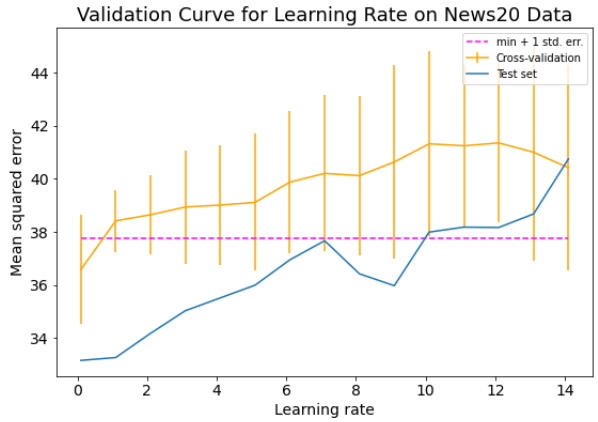


Figure 5: Validation curve for learning rate on 20 Newsgroups dataset. Note that the mean squared error immediately increases as the learning rate does.

For the 20 Newsgroups data, increasing the learning rate immediately increases the mean squared error. We can therefore infer that increasing the learning rate is not a viable option. However, for the Sentiment140 data, the learning rate can reach as high as 4 before it begins to hurt the model's accuracy.

Varying the proportion of training data used was consistent with our expectations; The test set accuracy tended to increase with more training data. Notably this was more pronounced in the 20 Newsgroups dataset. This makes sense as the Sentiment140 dataset contains roughly 100 times the training data and we expect diminishing returns from including more training data after a certain point. Another relationship we might expect is Naive Bayes performing relatively well even with less data. This is supported by the fact that in both datasets the Logistic Regression made greater gains in test accuracy for a given training size jump. Given the limited data points, though, this relationship would have to be explored further to make any conclusions. A plot of the accuracies can be found below.

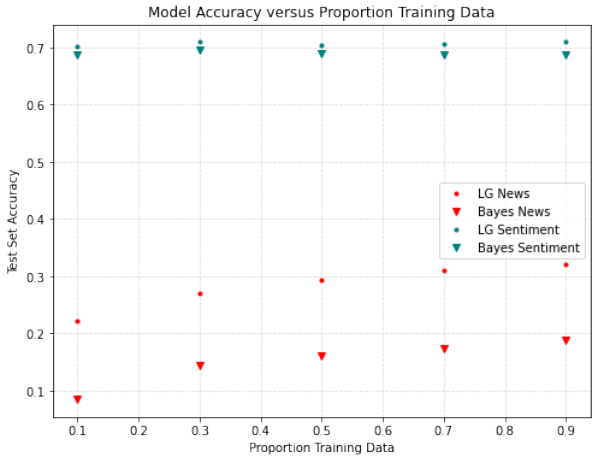


Figure 6: Test set accuracy for different proportions of the training data. Sentiment20 training data contains 1.6 million entries. 20 Newsgroups training data contains 15077 entries. Selected models are the best performing models for each dataset and algorithm.

To sum everything up, the accuracies across all models, datasets, and hyperparameters can be found in the tables on the following page.

Model	Dataset	Accuracy
G Naive Bayes	News	0.180
B Naive Bayes	News	0.039
G Naive Bayes	Sentiment	0.687
B Naive Bayes	Sentiment	0.704
Logistic 1	News	0.326
Logistic 2	News	0.321
Logistic 3	News	0.314
Logistic 4	News	0.316
Logistic 5	News	0.331
Logistic 6	News	0.325
Logistic 7	Sentiment	0.720
Logistic 8	Sentiment	0.616
Logistic 9	Sentiment	0.730
Logistic 10	Sentiment	0.638
Logistic 11	Sentiment	0.730
Logistic 12	Sentiment	0.730

Table 1: Accuracies for all tested models. G stands for Gaussian and B stands for Bernoulli. The best performing models are highlighted in cyan. Hyperparameter information for the Logistic models can be found in Table 1

Model	Dataset	Bias	Penalty	ID
Scratch	News	Yes	None	1
Scratch	News	No	None	2
Sklearn	News	Yes	None	3
Sklearn	News	No	None	4
Sklearn	News	Yes	L1	5
Sklearn	News	Yes	L2	6
Scratch	Sentiment	Yes	None	7
Scratch	Sentiment	No	None	8
Sklearn	Sentiment	Yes	None	9
Sklearn	Sentiment	No	None	10
Sklearn	Sentiment	Yes	L1	11
Sklearn	Sentiment	Yes	L2	12

Table 2: Logistic Regression hyperparameters for different showcased models.

Discussion and Conclusion

We found that in general Logistic Regression was the best performing algorithm for the two datasets. The performance of Naive Bayes and Logistic Regression was comparable for the Sentiment140 dataset. For the 20 Newsgroups dataset, however, Logistic Regression was significantly better.

Hyper parameter tuning showed that using bias and an L1 penalty for the Logistic Regression was the best performing option, with the highest test accuracies for both datasets. Naive Bayes did not have a clear winner, and performed better with binary features and a Bernoulli likelihood on the Sentiment140 dataset and a tfidf features and a Gaussian likelihood on the 20 Newsgroups dataset.

Increasing the proportion of training data used improved the test set accuracy as expected. Two interesting takeaways were the better increases on the 20 Newsgroups dataset likely due to the smaller total size (0.1×1.6 million is still a lot). A potential avenue for further investigation is seeing whether Naive Bayes can overtake Logistic Regression at low training sizes.

Another experiment needing more computational power to be performed is evaluating model performance as a function of the number of features. Searching for the maximum expressiveness before noise begins to erode the model's performance would be an interesting experiment.

Statement of Contributions

Lambert Francis, Patrick Janulewicz, and Kevin Sohn each contributed to this project fairly, evenly, and to the best of their abilities. Lambert focused on data analysis and algorithm implementation, Patrick focused on algorithm implementation and logistic regression experiments, and Kevin focused on Naive Bayes experiments and accuracy. All members contributed to the writing of the report.