A Comparison of Naive Bayes and Logistic Regression Performance in Text Classification

Parsa Yadollahi - 260869949
parsa.yadollahi@mail.mcgill.ca
Brian Hu - 260915192
brain.hu@mail.mcgill.ca
Liyun Huang - 260913905
liyun.huang@mail.mcgill.ca

Abstract

The importance surrounding the classification of textual data is driven by the ever-increasing popularity in our online inter-connectivity, leading to finding new ways in making predictions surrounding our online presence. Two fundamental algorithms are Naive Bayes (NB) and Logistic Regression (LR), both of which are supervised learning algorithms. In this project, we explore the performance of these two algorithms on two textual datasets: the 20Newsgroup dataset (20News) and the Sentiment140 dataset (SM140). We pre-processed the data, performed feature extraction and feature reduction, hyper-tuned and selected the best model for each algorithm with k-fold cross-validation and compared the accuracy of the two models on both datasets.

The results from our project indicate that, for the 20News dataset, the Logistic Regression model gave a better testing accuracy than Naive Bayes. Conversely, Naive Bayes performed better on the SM140 dataset. Both models performed better on the SM140 dataset over the 20News dataset. When feature reduction was performed, the testing accuracies of both models increased for the SM140 dataset, but decreased for the 20News dataset.

1. Introduction

1.1 Project Design

In this project, we were tasked with implementing Naive Bayes and k-fold cross-validation from scratch as well as importing an existing Logistic Regression algorithm and comparing how the two models perform on distinct textual datasets.

The Naive Bayes model was implemented as a Python class where the main methods were used to train the model and create a prediction based on a set of inputs. Given that we chose to preprocess our data in a discrete rather than continuous manner, we opted to implement Multinomial over Gaussian Naive Bayes. The Logistic Regression model was imported from the scikit-learn package, and set with the 'multinomial' flag to perform softmax regression.

K-fold cross-validation was used to determine the best hyperparameter value for the Multinomial NB model, while the imported GridSearchCV function was used for the Logistic Regression model, given that it was an existing imported model. The implemented function returned the best average k-fold result from performing cross-validation *k* times on the *k* different folds of training and validation sets.

1.2 Related Works

Text classification is often the focus of study for different classification algorithms. One such study is *Comparing automated text classification methods* (Hartmann et al., 2019), which concludes that although existing marketing research relies predominantly on the SVM classifier coupled with LIWC analysis, the Random Forest model performed consistently well for three-class sentiment and Naive Bayes for small sample sizes, with the SVM model never outperforming these two methods.

Additionally, considerable research is done on uncovering different ways to optimize the performance of a specific algorithm in classifying textual data. *Feature selection for text classification with Naive Bayes* (Chen et al, 2009) reasoned that with the high dimensionality of feature spaces, feature selection is crucial and among the dozen of evaluation metrics that exist, two dominantly performing metrics are

Multi-class Odds Ratio (MOR) and Class Discriminating Measure (CDM). Supporting this line of reason is *Large-Scale Bayesian Logistic Regression for Text categorization* (Genkin et al., 2012), which demonstrated that using a Laplace prior in their approach produced predictive models as effective as those using SVM classifiers or ridge logistic regression combined with feature selection.

2. Datasets

2.1 20 Newsgroup Dataset

The 20 Newsgroup (20News) dataset consists of documents partitioned across 20 newsgroups, with 11,314 instances in the training set and 7,532 instances in the testing set. Each instance's label indicates the newsgroup the document belongs to.

2.2 Sentiment 140 Dataset

The Sentiment 140 (SM140) training dataset consisted of 1,600,000 instances with five attributes (including the polarity attribute) classifying if a text was considered to have a positive or negative connotation, while the SM140 testing dataset consisted of 498 instances.

2.3 Preprocessing

We first sampled SM140's training dataset by randomly selecting 20,000 instances since 1,600,000 instances caused the session to crash. We then replaced all instances in the training and testing dataset that had label *4* with label *1* to make the class labels easier to comprehend, and removed instances in the testing dataset that had label *2* representing a neutral polarity since our training dataset did not contain any instances with that label.

For both datasets, we removed all unnecessary columns leaving only the *text* and the *polarity/label* columns. In order to extract features from raw texts, we used functions from the nltk library to preprocess both datasets by tokenizing, lemmatizing, and removing all stop-words, numbers and punctuation.

2.4 Statistics

Before sampling the SM140 training dataset, the number of instances were distributed as 800,000 classes with label *0* indicating a negative polarity and the same number of classes with label *1* indicating a positive polarity. We were left with 10,056 classes with label *1* and 9,944 classes with label *0*. For the testing dataset, we had 177 classes with polarity *0*, and 182 classes with polarity *1*.

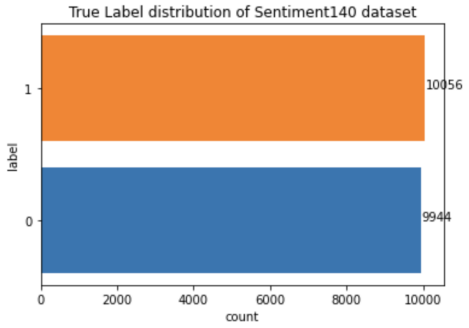

Figure 1: Distribution of Positive and Negative Labels in SM140 Training Dataset

In the 20News dataset, the instances are distributed fairly evenly across most newsgroups in both the training and testing sets. However, *talk.religion.misc*, *talk.politics.misc* and *alt.atheism* have fewer samples than other groups, so results produced by our models could skew towards the balanced labels.

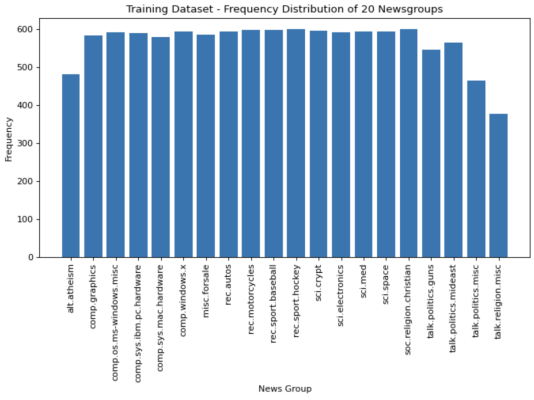

Figure 2: Distribution of News Groups in 20News Training Dataset

2.5 Feature Extraction & Feature Reduction

We used the TF-IDF Vectorizer from scikit-learn to extract features, which counts the frequency of each word and measures the importance of a word to a document in a collection.

To further balance the datasets, we used the undersampling method from the imblearn library. To make the models converge faster, we performed feature normalization by using the MinMaxScaler from scikit-learn. Since both datasets have a large number of features, we removed features that are insignificant to the dataset to avoid the curse of dimensionality, which suggests that the more features a dataset contains, the more difficult it is to model them. By using the SelectFromModel model from scikit-learn, we selected representative features from the datasets based on results produced from TF-IDF Vectorizer. For the SM140 dataset, we extracted 7,298 features out of 7,441 features; for the 20News dataset, we reduced 16,234 features, keeping 4,375 important ones.

3. Models

The implementation for the Naive Bayes model was heavily based on the examples given during the course's tutorial sessions (Rabbany, 2022) whereas the Logistic Regression Model was imported from the scikit-learn package.

3.1 Naive Bayes

Naive Bayes models are built based on Bayesian classification methods, relying on Bayes's theorem to find the probability of a label given some observed features, denoted by:

$$p(\theta \mid \mathcal{D}) = \frac{p(\theta)p(\mathcal{D} \mid \theta)}{p(\mathcal{D})}$$

Figure 3: Bayes's Rule for Posterior Probability

We chose to implement the Multinomial Naive Bayes model, with the alpha value as the hyperparameter to tune, in order to use the word frequency of each word in a certain type of document to calculate its conditional probability.

3.2 Logistic Regression

Softmax Regression, also referred to as Multi-class Logistic Regression, is a generalized version of the logistic regression that can be used for multi-class classification with label $y^{(i)} \in \{1,.., k\}$ where k represents the number of classes, assuming they are independent, whereas the standard logistic regression handles classifying binary labels: $y^{(i)} \in \{0, 1\}$.

Logistic Regression assigns these labels to data points and finds a decision boundary (threshold) that separates classes using the following cost function:

$$J(w) = \sum_{n=1}^{N} y^{(n)} \log\left(1 + e^{-w^\top x}\right) + (1 - y^{(n)}) \log\left(1 + e^{w^\top x}\right)$$

Figure 4: Logistic Regression Cost Function

where N is the number of instances, $y(n)$ is the labels for the instances, x is the matrix instances and w is the weight vector.

3.2.1 Effects of Regularization Strength

We used the GridSearchCV function provided by scikit-learn's model_selection package to select the best hyper-parameters for the Logistic Regression model given the input data and labels. We run the experiment with 5-fold cross-validation to search for the best hyper-parameter C, which is the inverse of regularization strength. GridSearchCV returns the accuracy for these combinations where we can select the best performing combination.

Regularization is useful for avoiding overfitting; the strength of the regularization can heavily affect the model's accuracy. We graph the values of C vs. the accuracy of the logistic regression model and other SVM functions, as shown in Figure 5. We can see that there is not a one-to-one mapping from increasing the regularization strength to increasing the model's accuracy; we see this in the graph below where initially increasing the value of C rapidly increases the accuracy of the models, followed by a slow decline past a certain point.

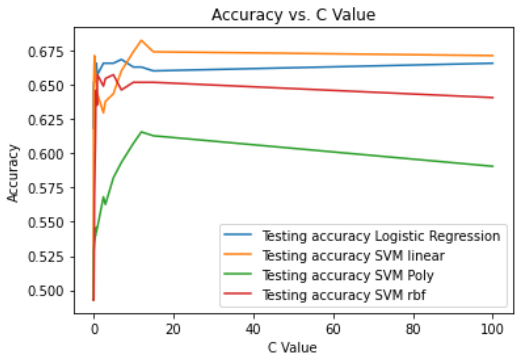

Figure 5: Accuracy vs. C value of LR and SVM Models

4. Results

A summary of the results recorded during this project can be found at the bottom of the paper in Table 1.

4.1 TF-IDF vs. Raw Count

Given that TF-IDF differs from Count Vectorizer by not only focusing on the frequency of words in the text but also providing insights into the relative importance of the words, we wanted to investigate its influence on each model's accuracy. We compared the two vectorizers' accuracies on each model. For both vectorizers, we set min_df to 3 to ignore terms that appear in less than 3 documents.

For the 20News dataset, the testing accuracy for Logistic Regression improved from 49.2% to 62.0%, and from 55.4% to 64.0% for Naive Bayes. For the SM140 dataset, the testing accuracy for Logistic Regression improved from 75.43% to 76.76%, and from 75.88% to 76.60% for Naive Bayes. Thus, all following experiments were conducted using the TF-IDF vector dataset.

4.2 Naive Bayes vs. Logistic Regression

For the SM140 dataset, the testing accuracy for the Naive Bayes is the best with 80% of the dataset. Softmax Regression performed best with 60% of the dataset. For the 20News dataset, both models' testing accuracies increased with the training dataset size, as shown in Figures 7 and 8. With 100% of the dataset, the Naive Bayes model had the highest testing accuracy of 61.79%, and Softmax Regression an accuracy of 62.07%.

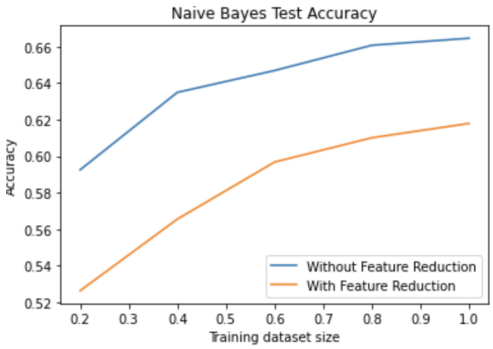

Figure 6: Naive Bayes Test Accuracy on 20News Dataset

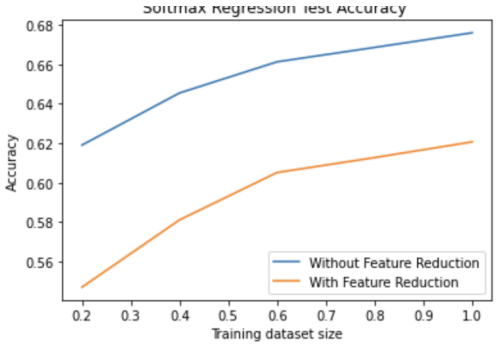

Figure 7: Softmax Regression Test Accuracy on 20News Dataset

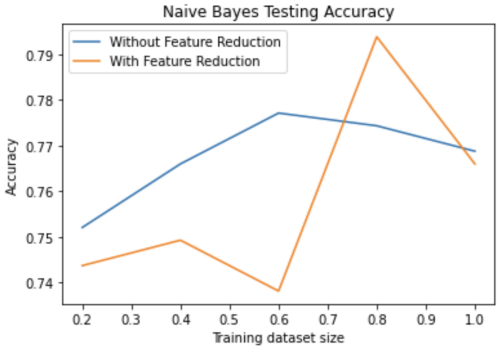

Figure 8: Naive Bayes Test Accuracy on SM140 Dataset

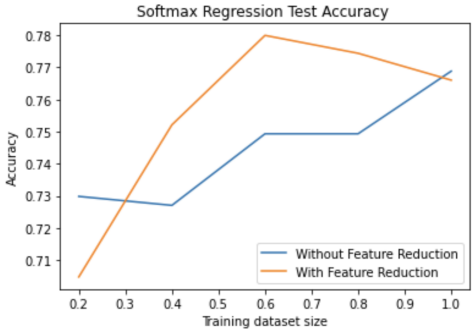

Figure 9: Softmax Regression Test Accuracy on SM140 Dataset

Naive Bayes performed better on the SM140 dataset, while Softmax Regression has higher accuracy on the 20News dataset.

4.3 Reduced vs. Full Feature Set

In order to investigate the effect of feature reductions on both models' performances, we

performed an analysis to compare the models' accuracies with and without feature reduction.

Both the Naive Bayes and Softmax Regression models had better performances on the SM140 dataset with feature reduction.

However, overall, the performances of these two models worsened with feature reduction for the 20News dataset. By using feature reduction on the 20News dataset, we had decreased testing accuracy for all models. Feature extraction has stripped away more than 78% of the original features in the 20News Dataset, reducing the dimensionality of the predictive models leading to reduced complexity of the model, all while retaining an accuracy only slightly lower than the original with all features present. Though we lost some accuracy, by removing many of these features, we were able to increase the model training time and convergence.

4.4 Creative Evaluations

We performed an analysis of the min_df parameter's effect on the model's training and testing accuracies. We set the min_df range from 1 to 7. We found that, for the SM140 dataset, min_df=3 produced the best testing accuracies, with 64.34% on Logistic Regression and 67.13% on Naive Bayes. For the 20News dataset, min_df=7 produced the highest testing accuracy, with 61.03% on Logistic Regression and 63.46% on Naive Bayes.

Given that our project is in the domain of NLP, we wanted to use another popular classifier for text classification - SVM (Support Vector Machine), and compare its performance against Logistic Regression and Naive Bayes. SVM is a supervised machine learning algorithm used for regression and classification. It classifies 2 classes of data by finding a (N-1)-Dimension hyper-plane in an N-Dimensional space, *N* denoting the number of features that separates them. We use the model provided by scikit-learn's library to run this experiment. We tested out three kernels for the SVM model: linear, polynomial, and Gaussian Radial Basis. Using the GridSearchCV to determine the best hyper-parameters to use for each model, we found that the linear function produced the best result, with 70.0% training accuracy and 65.46% testing accuracy.

5. Discussion and Conclusion

Overall, both models produced higher accuracies on the SM140 dataset. More specifically, Naive Bayes produced a better result than the Logistic Regression model. For the 20News dataset, Logistic Regression had a better performance than Naive Bayes.

For the future extension, one can experiment with feature reduction to find the optimal number of features for each dataset that produces the highest testing accuracy, since removing too many features from the dataset can harm the model's performance. One flaw in our vectorizers is that they used the number of occurrences in the documents and not the type of word. For example in sentimental analysis, an adjective could contribute much more to classifying text; this can be resolved by adding a weight function.

With various text classification algorithms out there, another possible future investigation could be conducting experiments on these algorithms and finding the most suitable classifier for each dataset.

In this experiment, the models learned feature representations for words. As an extension of this project, we hope to perform learning on feature representations for sentences or documents by using the doc2vec model from Gensim, and comparing the performances on different ways of feature extraction.

6. Statement of Contribution

We all contributed equally to the completion of the project and report.

7. References

Chen, J., Huang, H., Tian, S., & Qu, Y. (2008, June 24). *Feature selection for text classification with Naïve Bayes*. Expert Systems with Applications. Retrieved March 4, 2022, from https://www.sciencedirect.com/science/article/pii/S0957417408003564?casa_token=uj14iHTy0eoAAAAA%3AjxVGWUJlNchGRIE0wHq0VNZ

OF5Z1fSF7s6wPzBTa2Y9RHjn7rpLMHC7wuduI8gkGqFqhAcxkUm9u

Hartmann, J., Huppertz, J., Schamp, C., & Heitmann, M. (2018, October 24). *Comparing automated text classification methods*. International Journal of Research in Marketing. Retrieved March 4, 2022, from https://www.sciencedirect.com/science/article/pii/S0167811618300545

Mingyong Liu and Jiangang Yang. *An improvement of TF-IDF weighting in text categorization*. Retrieved March 6, 2022, from http://www.ipcsit.com/vol47/009-ICCTS2012-T049.pdf

Rabbany, Reihaneh. (2022). *COMP 551: Applied Machine Learning - Winter 2022*. CS551. (n.d.). Retrieved March 4, 2022, from http://www.reirab.com/Teaching/AML22/index.html

Taylor & Francis. (n.d.). *Large-scale bayesian logistic regression for text categorization*. Retrieved March 4, 2022 from https://www.tandfonline.com/doi/abs/10.1198/004017007000000245?casa_token=k9jRyaCLi8UAAAAA%3AUJGC9uMsZglP4ou4QrORsgY-u9qkBblF-uqUW5eftzTmVDhj_khbW1FopxjnpE0Ka6OV7wiYq2u_Gw

Appendix

Model	Best α/C Value	Dataset	Training Dataset Size (%)	Testing Accuracy (%)	
				Without Feature Reduction	With Feature Reduction
Naive Bayes	0.01	20Newsgroup	0.2	59.25	52.64
			0.4	63.49	56.56
			0.6	64.68	59.68
			0.8	66.06	61.01
			1.0	66.45	61.79
	0.25	Sentiment140	0.2	75.21	74.37
			0.4	76.60	74.93
			0.6	77.72	73.82
			0.8	77.44	79.39
			1.0	76.88	76.60
Logistic Regression	12	20Newsgroup	0.2	61.91	54.70
			0.4	64.55	58.11
			0.6	66.13	60.52
			0.8	66.86	61.27
			1.0	67.60	62.07
	0.5	Sentiment140	0.2	72.98	70.47
			0.4	72.70	75.21
			0.6	74.93	77.99
			0.8	74.93	77.44
			1.0	76.88	76.60

Table 1: Accuracies of Naive Bayes and Logistic Regression as a function of varying dataset sizes with and without feature reduction. The winners of each model for both datasets are highlighted.