COMP551: Applied Machine Learning
Project 2

GROUP 54: Matthew Morvan, Alexander Tiou Fat, Anne-Sophie Fratzscher

February 2021

Abstract

The objective of this paper is to determine whether Naïve Bayes or Logistic Regression is more accurate for classifying text. Our objective was to classify the topic of posts in the 20 newsgroups dataset and to determine the sentiment of IMDB reviews. We pre-processed our datasets, then implemented and optimized both classification methods by selecting the most highly predictive features and perform hyper-parameter optimization. Ultimately, we determined that, for both datasets, logistic regression was better at predicting than Naïve Bayes, with 71% accuracy and 89% accuracy achieved for the 20 newsgroup dataset and IMDB reviews, respectively. Additionally, we discuss the effects of train-test split on accuracy and compare our NB and LR with other common text classification techniques.

1 Introduction

One of the most powerful and common applications of machine learning (ML) is text classification in natural language processing, with uses ranging from sorting text messages to organizing information feeds. With companies like Facebook processing nearly a billion messages a day, a large range of efficient ML methods, ranging from Support Vector Machines to Deep Learning, have been developed for text classification.

Two popular ML text classification techniques are Naïve Bayes (NB) and Logistic Regression (LR). NB is a generative statistical model based on implementing Bayes's model with strong (Naïve) independence between features. More precisely, in this paper, we will be using the Multinomial and Bernoulli Naïve Bayes models, in which instances are frequencies and predictions are made based on the likelihood of obtaining a particular feature. LR, on the other hand, is a discriminative statistical model that makes predictions through a logistic function and subsequently classifies according to a binary dependent variable.

Both algorithms have pros and cons. NB is simple to implement and scales to large datasets while remaining resilient to noise, but it is sensitive to features that are dependent to each other, implying that some features will be processed with a much higher bias. LR, while also simple to implement, can be sensitive to pre-processing and poorly-tuned features and performs better with larger datasets (compared to NB).

The objective of this project was to implement, optimize and determine the most accurate machine learning models to perform text classification on (1) newsgroup types and (2) IMDB Review. For the news group dataset, the goal was to determine which of the 20 news feeds a post belongs to, whereas for the IMDB dataset, the goal was to perform sentiment analysis (categorize the review as either positive or negative).

Previous research has shown success in categorizing the newsgroup dataset using support vector machines (SVM) and KNN, with 86% accuracy achieved using SVM [1]. Other works have also used multinomial NB to categorize the newsgroup dataset, with length normalization and Laplace smoothing yielding better results [2]. Work has also been done for sentiment analysis of IMDB reviews, notably using SVM, NB, and Markov blanket, with 82% accuracy achieved using NB. [3]. There has been even greater success using deep CNN-LSTM (a type of deep learning neural network), with accuracy of over 89% being achieved [4].

In this project, we compared the NB classifier and LR classifier for both datasets. To optimize accuracy and efficiency, datasets were cleaned and the text data was turned into numerical features using a bag of words representation.

Ultimately, from our cross validation we observe that the ideal alpha parameter for our optimized NB was 0.01 for dataset 1, giving an accuracy of 65%, and $\alpha = 0.1$ for dataset 2, yielding accuracy of 86.4%.

Tuned logistic regression yields an accuracy of 73% for an inverse regularization strength of 1 for dataset 1, and 89.6% accuracy using inverse regularization strength of 1 for dataset 2. Additionally, both classifiers perform better on the IMDB Review dataset than the 20 newsgroups dataset.

We also compare both of our methods with other common classification techniques given in the SciKit learn package and determine best train-test split ratios to maximize accuracy for each.

2 Datasets

We analyzed two datasets for this project: the 20 newsgroups dataset and the IMDB Reviews dataset. The 20 newsgroups dataset was retrieved from Scikit-learn, while the IMDB reviews was provided by Association for Computational Linguistics [5].

2.1 20 Newsgroups Dataset

Scikit-learn's 20 newsgroups dataset consists of 18,846 newsgroups posts on 20 topics. The goal for this dataset was to determine which topic a given post belonged to. Each instance is labeled between 0 and 19, which corresponds to the 20 different categories; these categories were atheism, graphics, ms-windows miscellaneous, PC hardware, mac hardware, windows, for sale, autos, motorcycles, baseball, hockey, crypt, electronics, medicine, space, Christianity, guns, middle east, politics miscellaneous, and religion miscellaneous.

For pre-processing, the headers, footers and quotes were removed for each documents. Text was pre-processed and tokenized to create 134,410 features that represent each word in the dataset using CountVectorizer. Additionally, we used a technique called "Term Frequency - Inverse Document Frequency" (or TF-IDF), which penalizes words when they appear frequently in different documents. For example, words such as "the" and "is", which occur in most documents, have a high frequency but a low TF-IDF value.

Analysis of the data shows that labels 0 to 19 have 799, 973, 985, 982, 963, 988, 975, 990, 996, 994, 999, 991, 984, 990, 987, 997, 910, 940, 775 and 628 instances respectively.

2.2 IMDB review Dataset

The IMDB review dataset consists of 50,000 instances, with each instance being labeled as either positive or negative. The goal for this dataset was to determine the either a positive or a negative sentiment for a given review.

The dataset was split in two folders, one containing the training set and the other containing the testing set, where both subsets contain 25,000 instances each. We combine the training and testing folders together in order to obtain a harmonized amount of features after vectorization using CountVectorizer. We end up with 50,000 instances and 101,895 features (note that the first half was used for training, whereas the second half was used for testing purposes). Again, we used the TF-IDF technique with goal of increasing accuracy.

The labels present in this dataset are 0 and 1 which represent negative and positive reviews, respectively. Analysis of the data shows that the classes 0 and 1 and evenly distributed.

3 Results

We implemented, optimized and compared two classification techniques: (1) Naïve Bayes and (2) Logistic Regression. We implemented the NB from scratch and the LR using the Scikit-learn library. For both methods, datasets were split into training and test sets, where the test set was used to estimate the performance after training the model with the training set. The final performance was evaluated using accuracy, which we implemented in "evaluate_acc" function.

3.1 Naïve Bayes Classifier

To build our Naïve Bayes models, a categorical distribution for the prior class probability is chosen for the 20newsgroup dataset and a Bernoulli distribution for the IMDB Review dataset. For the likelihood of

both models, a multinomial distribution is chosen through optimization (see section 3.3).

In order to handle the zero probability in Naïve Bayes (i.e. a word appears in the test set but not in the training set, therefore a frequency of 0 is given to this word in a specific instance), we use the Laplace smoothing technique.

The optimal smoothing parameter alpha(α) is found using a 5-fold cross validation on an 80-20 split training/test set. The error rates for different α can be seen below:

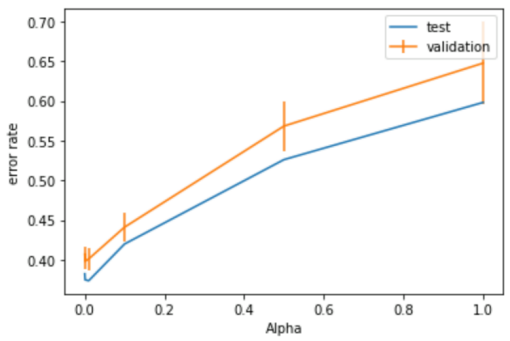

Figure 1: 20newsgroups dataset error rates

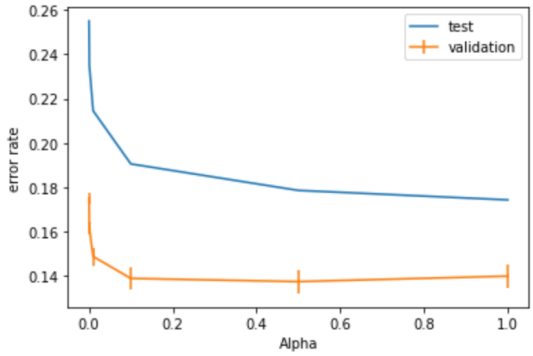

Figure 2: IMDB Review dataset error rates

We see that using $\alpha = 0.01$ for smoothing gives the lowest validation error rate of around 41.2% for the 20newsgroups dataset, and $\alpha = 0.1$ for the IMDB review dataset gives a validation error rate as low as 14%.

After finding our optimal α for each models, we end up with a test set accuracy of 75.3% and 81% for 20 newsgroups dataset and IMDB review dataset, respectively.

3.2 Logistic Regression Results

To build the best logistic regression model for both datasets, we used SciKit-learn's built in GridSearchCV and Best_Params_ functions, which allow us to carry out an optimized cross-validated grid-search over a parameter grid and then select the parameter settings that gave the best results on the hold out data.

As a result, for dataset 1, we found that multinomial loss fit with no penalty using the SAGA solver yielded the best accuracy of 71 %. For dataset 2, the multinomial loss fit, with l2 penalty and SAG solver yielded the best accuracy, with an accuracy of 89%.

We also carried out manual tuning of these hyper-parameters by performing a grid search with a 5-fold cross validation for both datasets to confirm these results. For both datasets, we deduced from our 5-fold cross validation that an inverse regularization strength (C) of 1 gave us the best accuracy, rendering an error

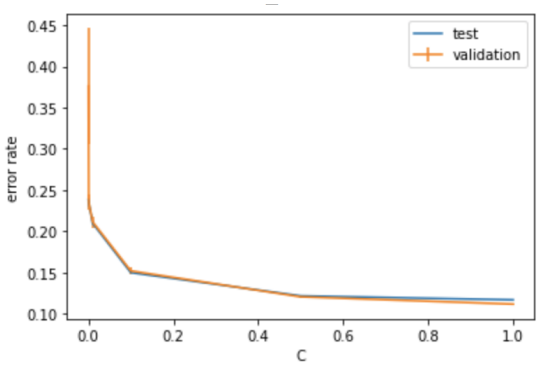

Figure 3: Logistic Regression for 20newsgroups

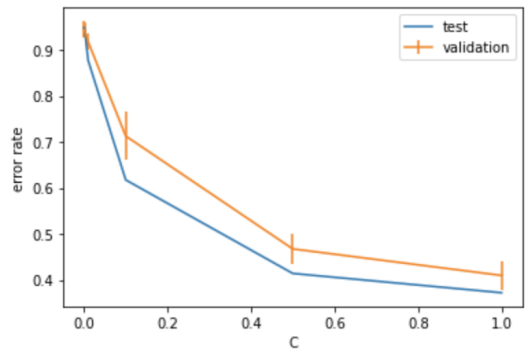

Figure 4: Logistic Regression for IMDB Reviews

rate of 38% for dataset 1 and an error rate of 12.6% for dataset 2. We graph our result in Figures 3 and 4. which plot our error rate (1 - accuracy) according to C.

3.3 Comparison with other statistical models and train set size

In order to complete our study and determine how both implementations of NB and LR performed relative to other text classification methods, we also tried training both NB and LR on different train-test splits and compared them to other statistical models, including Bernoulli NB (BNB), Optimized Logistic Regression (OLR), Default Logistic Regression (DLR), Gradient Descent Classifier (GDC) and Support Vector Classification (SVC).

For both datasets we observe that OLR, DLR, GDC and SVC have similar accuracies ranging between 88-90% for both dataset 2 and 65-75% for dataset 1. For dataset 1, a maximum seems to be reached for all methods at the 90-10% training-test split, while it appears between the 80-20% and 90-10% for dataset 2. In both datasets, SVC comes in first with the highest accuracies.

In comparing NB with LR (optimized), we notice that the optimized logistic regression performs substantially better than the Naïve Bayes regardless of the train-test split, with LR reaching a maximum of 74% and 98.5% accuracy for dataset 1 and dataset 2, respectively (both have their maximum at the 90-10% train-test split), whereas NB reaches a maximum of 65% (90-10% train-test split) and 86.4% (60-40% train-test split) for dataset 1 and 2, respectively. We also notice that Naïve Bayes does quite poorly until it reaches a train-test split of at least 50-50%.

Figures 5 and 6 show the various statistical methods and their accuracies for different train-test splits.

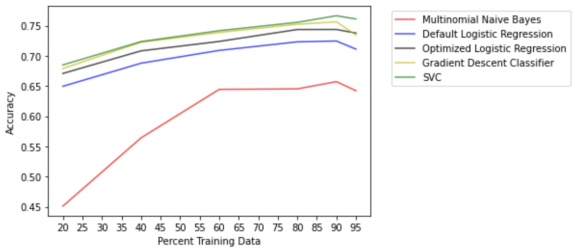

Figure 5: Train-Test comparison for 20newsgroups

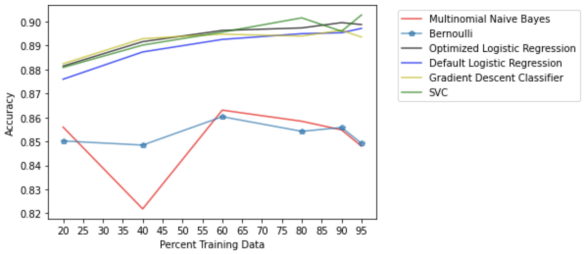

Figure 6: Train-Test comparison for IMDB Reviews

4 Discussion and Conclusion

Our objective was to find the ideal machine learning model and ideal hyper-parameters for text classification. For both datasets, we show that hyper-parameter tuning can greatly affect model accuracy. Additionally, training to test data ratio has an effect on accuracy, with a higher training-to-test ratio leading to better accuracy. This was particularly true for NB, which required train-test split to be greater than 50-50 for efficient predictions. This is likely due to the fact that NB processes some features with a much higher bias than others (especially when their exists dependence between them).

Ultimately, from our cross validation we observed that the ideal alpha parameter for our optimized NB was 0.01 for dataset 1 and 0.1 for dataset 2, deriving an accuracy of 65% for dataset 1 and 86.4% for dataset 2. Tuned logistic regression received an accuracy of 73% for an inverse regularization strength of 1 on dataset 1, and 89.6% accuracy using an inverse regularization strength of 1 on dataset 2. Additionally, both classifiers performed better on the IMDB Review dataset. The poor pre-processing of our words and the fact that NB tends to have biases towards interdependent features may explain why our accuracy was so low using this particular classification method.

We should also note that our NB method was less accurate than scitkit-learn's NB method, meaning that there was room for improvement such as specifying weights of certain features and adjusting prior probabilities of classes differently to our data. Similarly for LR, we could have adjusted other parameters such as the learning rate, our epsilon, or the maximum number of iterations for our gradient descent. We

also could have done more work on pre-processing our data such as removing stop words/common words, "lemmatizing" words (grouping together different word inflections), as well as using n-grams (counting groups of several words rather than just one).

Finally, as was explored in the last part of our analysis, there exists other forms of statistical methods for classification, such as support vector classification, which were shown to be more accurate and stable on our two datasets, with accuracy reaching up to 90%. Beyond statistical models of analysis for text-classification, natural language processing can apply other popular ML methods that are very effective on large datasets, such as Deep Learning.

5 Statement of Contributions

Matthew implemented cross-validation and wrote part of results as well as both the discussion and introduction. Anne-Sophie pre-processed, ran statistics on both datasets and performed all the experimentation in task 3. Alexander implemented NB from scratch and wrote part of the results as well as the section on datasets.

References

[1] G. Siolas and F. d'Alche Buc. Support vector machines based on a semantic kernel for text categorization. In *Proceedings of the IEEE-INNS-ENNS International Joint Conference on Neural Networks*, pages 205–209. IJCNN, 2000.

[2] Alfons Juan and Hermann Ney. Reversing and smoothing the multinomial naive bayes text classifier. pages 200–212, 2002.

[3] Bai Xue. Predicting consumer sentiments from online text. volume 4, pages 732–742, 2011.

[4] Alec Yenter and Verma Abhishek. 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)*, volume 4, pages 540–546. IEEE, 2017.

[5] Andrew L. Maas, Raymond E. Daly, Peter T. Pham, Dan Huang, Andrew Y. Ng, and Christopher Potts. Learning word vectors for sentiment analysis. In *Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies*, pages 142–150, Portland, Oregon, USA, June 2011. Association for Computational Linguistics.