COMP551 - Mini Project 2 - Classification of Textual Data

Group 57: Jean-Pierre Falet, Miguel Ibanez Salinas, Alex Le Blanc
{alex.leblanc,miguel.ibanezsalinas,jean-pierre.falet}@mail.mcgill.ca

February 28, 2021

Abstract

In this project, we investigated the performance of naive Bayes and softmax regression on two benchmark datasets for document classification using textual input: the 20 newsgroups and IMDb sentiment datasets. We tuned their hyperperparameters on both datasets using random search and cross-validation. We found that multinomial naive Bayes had the highest testing accuracy on the 20 newsgroups dataset (69.54%) while sofmtax regression performed best on the IMDb dataset (86.37%). We explored a variety of feature selection techniques including Chi squared statistics and the removal of movie-specific words in the IMDb dataset, both of which led to stable or improved peak accuracies while significantly reducing the dimensionality of the input. We also investigated the effect of training dataset size on each model's performance and found that both models performed better with larger training sets, although the naive Bayes model fared better than softmax regression on the smaller datasets.

1 Introduction

Text is a common feature used for a variety of classification tasks. For this project, we investigated document classification using two benchmark datasets: the 20 newsgroups dataset [1], and the IMDb sentiment dataset [2]. The 20 newsgroups dataset consists of news-related documents classified into news subtypes, while the IMDb dataset consists of movie reviews classified into the sentiment of the review.

Both datasets are commonly used in the natural language processing literature as benchmarks. While naive Bayes (NB) and softmax regression (SR) do not offer state-of-the-art performance on these datasets, their simplicity and greater interpretability are advantageous. Moreover, significant work has been done to improve their performance by selecting more predictive features using metrics such as mutual information, Chi statistic, and document frequency [3, 4].

In this project we compared performance of both models on each dataset following hyperparameter tuning, as well as the effect of dataset size and feature selection on prediction accuracy.

2 Dataset and Models

2.1 Preprocessing

Both datasets consist of text documents (collections of sentences) that were preprocessed in the same way. We defined a CorpusPreproc class which performs basic preprocessing. First, CorpusPreproc performs lemmatization using NTLK's WordNet, which transforms words into their dictionary form. Then, it removes stop words using NTLK's english list which contains frequently used words that generally add no semantic meaning to a sentence. Finally, it removes punctuation. These preprocessed documents are then transformed into lowercase and then into a vectorized bag of words representation where each word in the corpus represents a feature whose value is its count within each document. We further experimented with a scikit-learn's term frequency times inverse document frequency (TF-IDF) which converts vectorized counts into document frequencies weighted inversely to how often these terms are found across all documents. This serves to eliminate the effect of document length on word count, and to balance the relative representation of frequently used words and rarer words.

2.2 20 Newsgroups Dataset

The 20 newsgroups dataset consists of a collection of 18,846 news documents organized in 20 different topics. The topic of each document is its target label, which implies this is a multiclass classification problem with 20 different classes. Of note, this dataset contains highly predictive but non-generalizable features (e.g. name of frequent posters, university titles, etc.) within headers, footers, and quotes, so these were stripped from the documents during import of the dataset. The dataset comes split into a training (11,314 documents) and a testing (7,532 documents) set based on a specific date. Moreover, 88% of documents have less than 2,000 characters and 87% have less than 400 tokens. Classes are fairly balanced, with a mean of 565.7 documents per class (SD 56.8, range 377-600). Following the preprocessing described in Section 2.1 (without the use of ngrams and without exclusion of words based on frequency), the total number of features is 93,420.

2.3 IMDb Sentiment Dataset

The IMDb dataset consists of a collection of movie review documents posted by users on IMDb's website, organized into two classes: positive and negative reviews. The binarization is based on ratings $\leq$ 4/10 being negative and $\geq$ 6/10 being positive. Neutral reviews (5/10) are excluded from this dataset. This is therefore a binary classification problem. Similar to the 20 newsgroups dataset, 83% of documents have less than 2000 characters and 96% have less than 750 tokens. The dataset contains a total of 50,000 movie reviews, with 25,000 being in the training and 25,000 being in the testing set. In the training set, the 25,000 movie reviews are about 3,456 distinct movies, none of which are found in the test set. There are a maximum of 30 reviews for each movie to avoid having too many correlated reviews, and each class is balanced (there are an equal number of positive and negative reviews). Following the previously mentioned preprocessing steps, the total number of features is 67,147.

2.4 Models

Naive Bayes (NB) is a generative classifier that models the joint probability between features and labels and requires a strong assumption of conditional independence between features. We implemented two Bayesian NB versions: multinomial NB, with count of words as features, and binary NB, that considers only the presence or absence of words (binary features). The multinomial NB was implemented as per [5]. In this version, the conditional probability of a word w_i given a document class c is: $P(w_i \mid c) = \frac{count(w_i,c)+\alpha}{(\sum_{w \in V} count(w_i,c))+\alpha|V|}$, where $count(w_i, c)$ is the count of word w_i in the documents of class c, $|V|$ is the number of unique words in all documents and α is the smoothing factor. For the binary version, we implemented the version from the slides of the course [6]. In our binary NB implementation, the smoothing factors α (smoothing factor when feature is present) and β (smoothing factor when feature is absent) are set up to be equal.

Logistic regression is a discriminant classifier that models directly the posterior class probability given the features. Since there was multi-class data, we opted to implement softmax regression (SR) rather than logistic regression as it is a generalization of logistic regression that works both for binary and multi-class data. No batching was used, as the datasets were small enough that the regression and gradient descent functions could handle the entire datasets at once. Finally, L1 and L2 regularization were separately implemented as tunable hyperparameters.

3 Results

We evaluated the performance of SR, multinomial NB and binary NB with 5-fold cross-validation (CV). Our KfoldCV function also stratifies the splits so the proportion of each class is preserved. We report both the average CV accuracy and the test accuracy. The average CV accuracy refers to the average across the k validation splits, and was used to select optimal hyper-parameters during training. The test accuracy refers to the accuracy on the unseen test data, and is reported for the final model as an estimate of the generalization error.

3.1 Hyperparameter Tuning

We implemented a random search function (random_search()) for hyper-parameter optimization, as an exhaustive grid-search would not be practical. Between 40 and 50 searches in the hyperparameter space were done. The hyperparameters proper to SR are learning rate $\alpha \in \{0.1, 0.01, 0.001, 0.0001\}$, type of regularization (L1 vs L2) as well as the regularization scalar $\lambda \in \{0, 0.1, 0.5, 0.9\}$. The only hyperparameter that is just for the NB model is the smoothing parameter $\alpha \in \{0, 0.15, 0.5, 0.75, 0.9, 1\}$. For the word count vectorization, the optimization includes the minimum $\{2, 5, 7, 10\}$ and maximum $\{0.2, 0.35, 0.5, 0.75\}$ term frequencies and the use of matrix normalization (TF and/or IDF). For all experiments, ngram was fixed to one.

Table 1 shows the hyperparameters for the best average CV accuracy in the NB model with its corresponding test accuracy. In summary, multinomial NB outperforms binary NB in both datasets/classification tasks. Multinomial NB CV accuracy in 20newsgroup is 74.23% vs 58.79% from binary NB, which is the biggest difference in performance between both NB versions. The 20newsgroup dataset seems to be a particularly difficult problem for the NB classifier, but multinomial NB still takes advantage of the word count information. For the case of the IMDb dataset, multinomial NB shows a slightly better training accuracy (86.72%) than that of the binary NB (84.82%). The test accuracies on both datasets are slightly below the average CV accuracies (more so for 20 newsgroups), which is attributable the expected slight overfitting on the training set during CV and hyperparameter tuning. Regarding the best hyperparameters, TF preprocessing helps Multinomial NB, and a small smoothing parameter α ($\leq$0.5) seems to be preferable.

Table 2 presents the best hyperparameters for SR according to 5-fold CV, as well as the corresponding average CV and test accuracies. Note that for all training, the maximum number of epochs was fixed at 100. As was the case for the NB model, SR performs significantly better on the IMDb set than on the 20 newsgroups set, both in terms of CV and test accuracy. Moreover, small regularization constants ($\lambda \leq 0.1$) were preferred and L1 regularization performed better than L2 regularization, especially as λ grew, as exhibited in figure 1 for the IMDb data. This is expected, as even after pre-processing, there are so many features that lots of them are not useful for predictions and L1 regularization helps remove these features entirely by allowing the weights for these features to reach zero.

We then inspected the validation and training accuracy for one split using the best hyperparameters for SR ($\alpha = 0.1$) in figure 2a. Note that the plots are for the IMDb data, but very similar trends were observed in the 20newsgroup data. Accuracies generally trend upwards as the number of epochs increases, but due to the high learning rate, there are large oscillations due to overshooting the minimum during gradient descent. With a lower $\alpha = 0.0001$ (figure 2b), gradient descent progresses much more stably and predictably towards the minimum. The fact that $\alpha = 0.1$ was found to be the best hyperparameter by random search was due to the constraint imposed by the fixed epoch of 100; by chance, epoch 100 happens to be at the peak of the oscillation. Fixing the epoch at 90 would have led to a very low validation accuracy and the best hyperparameters would likely have been quite different. Random search results can therefore be misleading in this setting, so we propose some solutions in our Discussion.

Based on average CV accuracy, for the 20newsgroups dataset, Multinomial NB model performs better (74.23%) than SR (68.05%). For the IMDB dataset, we notice the opposite: SR performs slightly better (86.98%) than Multinomial NB (86.72%).

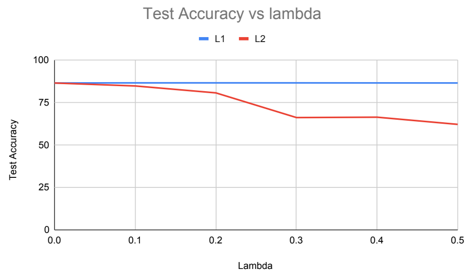
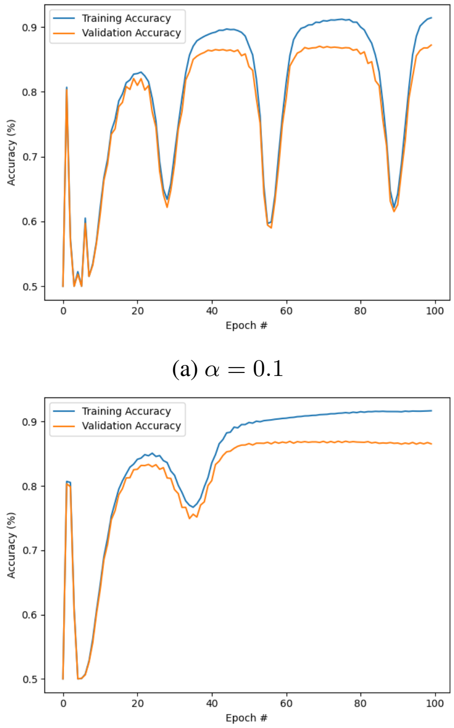


Figure 1: L1 vs L2 regularization (IMDb data) - SR.



(a) $\alpha = 0.1$

(b) $\alpha = 0.0001$

Figure 2: IMDb accuracy vs epochs - SR.

The test accuracy for the best model/dataset are reported in Table 3. We notice that the CV accuracy is, in general, a good estimation of the test accuracy (small overestimation previously discussed). Regarding the selected models, the test accuracy of Multinomial NB on 20 newsgroups is 69.54% and the test accuracy of SR on IMDB is 86.37%.

3.2 Effect of Training Dataset Size

Figure 3 shows the effect that different training set sizes has on test accuracy. Both models behave similarly within each dataset. For a smaller data set (20 newsgroups), increasing the proportion-of-total-training size from 20% to 80% increases the test accuracy by 8 or 9 pp.

Naive Bayes - Accuracy for best hyperparameters - ngram (1,1)								
Data	**Features**	**Max df**	**Min df**	**TF**	**IDF**	α	**CV Acc. (%)**	**Test Acc. (%)**
20NEWS	Multinomial	0.2	5	True	True	0.15	*__74.23__	69.54
20NEWS	Binary	0.75	5	N/A	N/A	0.15	58.79	54.34
IMDB	Multinomial	0.35	2	True	False	0.5	*__86.72__	84.24
IMDB	Binary	0.35	5	N/A	N/A	0.5	84.82	82.03

Table 1: Naive Bayes - Average cross-validation (CV) accuracy and test accuracy. *Best models chosen based on average CV accuracy.

Softmax Regression - Accuracy for best hyperparameters - ngram (1,1)									
Data	**Max df**	**Min df**	**TF**	**IDF**	α	**Regul. type**	λ	**CV Acc. (%)**	**Test Acc. (%)**
20NEWS	0.75	7	False	False	0.001	L2	0.1	68.05	63.48
IMDB	0.2	10	False	False	0.1	N/A	0	86.98	86.37

Table 2: Softmax Regression - Average cross-validation (CV) accuracy and test accuracy.

Naive Bayes vs Softmax Regression - ngram (1,1)		
Data	**Naive Bayes**	**Softmax Regression**
20NEWS	*__69.54__	63.48
IMDB	84.24	*__86.37__

Table 3: Test accuracy (%). *Best models based on test accuracy.

This effect is less noticeable in a larger dataset (IMDb), where the same difference in test accuracy is around 1 or 2 pp. When comparing between models, multinomial NB in general fares better on smaller datasets (20 newsgroups) and seems less sensitive to reductions in the IMDb dataset size than SR.

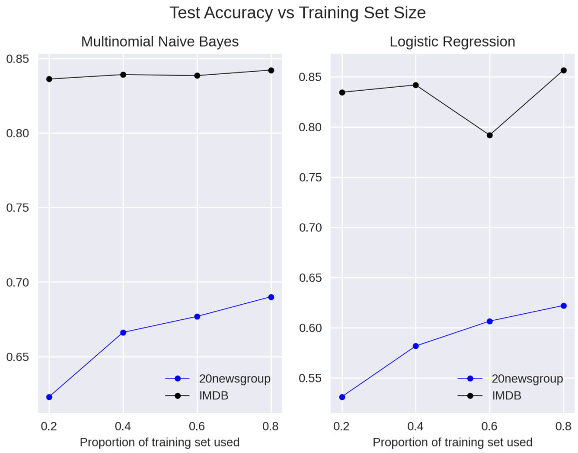


Figure 3: Effects of training size on test cccuracy.

3.3 Feature Selection

We hypothesized that a majority of the 67,147 features in the IMDb dataset are not necessary or even helpful to achieve high prediction accuracy, and tried two methods to reduce input dimensionality: removing movie-specific words and selecting important features with a Chi statistic.

First, we removed words from the original IMDb feature set that are highly specific for a specific movie, as these would not be expected to be predictive of sentiment on unseen test movies. We created an additional function (review_to_url()) which concatenates all reviews about the same movie (given the same movie url) into one document. We then performed count vectorization on this new movie-based corpus. The words with document frequency of pne (words that appear in reviews about only a single movie) were identified and fed into the review-based corpus vectorization as stopwords. This technique identified 29,824 movie-specific words out of the original 67,147, which leads to a considerable reduction of 44% in feature dimensionality.

Adding this preprocessing step and running Multinomial NB with the best hyperparameters identified in table 1 yielded a slight reduction in average CV accuracy to 86.70% (given that the train and validation splits share the same movies because of random shuffling, movie-specific features may still benefit prediction on the validation splits), but generalized slightly better to the unseen test set (84.33% accuracy). Using the same

Effect of feature selection size (K best) on accuracy (%)				
		K Best		
Data	Model	2000	5000	10000
20NEWS	*NB - Multinomial	71.21	73.51	74.16
IMDB	*NB - Multinomial	86.32	86.72	86.82

Table 4: Naive Bayes - Average cross-validation accuracy for K highest scored features (chi-square). *Best models from Table 1.

preprocessing on SR, an average CV accuracy of 87.05% was observed, which is a very slight increase compared to that obtained without this preprocessing. A testing accuracy of 86.38% was obtained, which is also slightly greater than the testing accuracy obtained without this pre-processing.

Second, we used the SelectKBest library (sklearn) to score the relation between features and targets using the chi-square function and to select the k highest scoring features for analysis. Table 4 shows the effect of feature selection on the average CV accuracy of the best NB model for each dataset. As feature size k increases from 2,000 to 10,000, we observe a small increase in average CV accuracy, particularly for the 20 newgroups set (from 71.21% to 74.16%). This effect is less pronounced in the IMDB dataset. The effect of selecting k best features resembles the effect of training size on test accuracy displayed in Figure 3, but overall shows that most of the predictive power comes from a small subset of features.

4 Discussion and Conclusion

State-of-the-art accuracy for classification on the 20 newsgroups dataset is 88.6% [7] and 97.4% for the IMDb dataset [8] using advanced neural networks architectures and modern word embeddings. This project demonstrates that simpler models such as NB and SR combined with traditional bag-of-words vectorization methods can produce useful baseline results in the both tasks, particularly in the IMDb dataset (SR test accuracy: 86.37%; 11 pp less than SOTA). Regarding the 20 newsgroups, unfortunately, our best result is 20 pp less than that of SOTA's (NB test accuracy: 69.54%).

NB predictions can benefit from the use of prior probabilities in small data regimes. Our results support this as Multinomial NB with smoothing ($\alpha \leq 0.5$) performs better than SR on the smaller dataset (20 newsgroups) while SR performs better on the larger dataset (IMDb) (Table 3). It can be argued that multiclass classification (20 newsgroups) is a more difficult problem than binary classification (IMDb) because the model needs to define more decision regions, which requires comparatively more data. Additionally, NB assumes features' conditional independence and SR does not consider possible interactions between features. These assumptions may not be as appropriate in the case of the 20 newsgroups classification task and could potentially explain part of the reduction in accuracy compared to the IMDb classification task.

We found small improvements in prediction accuracy with feature selection techniques such as selecting best features based on a chi-square function, and eliminating movie-specific words. This highlights the notion that more features is not always beneficial, and that dramatic reductions in feature dimensionality ($\geq$ 44% in our experiments) can even result in reduced generalization error and improved computational efficiency.

To address the issue identified in figure 2, we propose two future directions. First, instead of fixing an epoch for training, training SR with early stopping would enable finding the best accuracy associated with a given set of hyperparameters independent of training duration. Second, to improve the interpretation of the robustness of the accuracy vs. epoch curve when oscillations are present, one can smooth the validation accuracy with an exponential moving average.

Finally, for future research, we could consider experimenting with different document embeddings, either averaging word embeddings per document (doc2vec) or direct document embedding (Infersent). Given more computational resources, we would also experiment with bigrams or trigrams.

5 Statement of Contributions

All authors contributed equally to writing this report. Jean-Pierre Falet worked on dataloading, random search, and K-fold CV. Miguel Ibanez Salinas worked on dataloading, the NB model, experiments on both datasets, and studied the effect of dataset size on performance. Alex Le Blanc worked on the SR model, and experiments on both datasets.

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] Sang-Bum Kim et al. “Effective Methods for Improving Naive Bayes Text Classifiers”. In: *PRICAI 2002: Trends in Artificial Intelligence*. Ed. by Mitsuru Ishizuka and Abdul Sattar. Berlin, Heidelberg: Springer Berlin Heidelberg, 2002, pp. 414–423. ISBN: 978-3-540-45683-4.

[4] Yiming Yang and Jan O. Pedersen. “A Comparative Study on Feature Selection in Text Categorization”. In: Morgan Kaufmann Publishers, 1997, pp. 412–420.

[5] Dan Jurafsky and James H. Martin. *"Speech and Language Processing: An Introduction to Natural Language Processing, Computational Linguistics, and Speech Recognition "*. Pearson, 3st edition, 2020. ISBN: 9780130950697.

[6] Reihaneh Rabbany. *"Applied Machine Learning - Naive Bayes"*. 2021. URL: http://www.reirab.com/Teaching/AML21/naiveBayes.pdf.

[7] papers with code. *Sota accuracy 20newsgroup*. URL: https://paperswithcode.com/sota/text-classification-on-20news. (accessed: 27.02.2021).

[8] papers with code. *Sota accuracy imdb sentiment*. URL: https://paperswithcode.com/sota/sentiment-analysis-on-imdb. (accessed: 27.02.2021).