Mini Project 2: Classification of Textual Data

Group 60: Alvin Chen, Ziqi Li, and Emily Tam

February 28, 2021

Abstract

In this project, we investigated the performance of two linear classification models, multinomial naive Bayes and logistic regression, on two benchmark datasets: the 20 news group dataset and the IMDB dataset. It is worth mentioning that we conducted multi-class classification on the 20 news group dataset and binary classification on the IMDB dataset. We used 5-fold cross validation for the hyperparameter tuning of our models and found that the logistic regression model achieved better accuracy than naive Bayes model but was significantly slower to train compared to the naive Bayes model. Our results also showed that the accuracy of the two models increased as the training dataset size increased. For the creativity section, we experimented with using the bag of words text embedding method for data pre-processing but the accuracy performance was found to be worse than using the tf-idf text embedding method. We also investigated the effect of various n-grams on the accuracy of the two datasets, and found that n-grams can increase and decrease accuracy, depending on the situation.

Introduction

In this mini-project, we implemented naive Bayes and K-fold cross-validation from scratch, and used logistic regression from the scikit-learn package. The goal of this project was to gain experience implementing these algorithms and comparing their performance on two distinct textual datasets: the 20 news group dataset and an IMBD dataset. The 20 news group dataset is a collection of 20,000 news group documents divided into 20 different news groups while the IMBD dataset is a set of 50,000 highly polar movie reviews, which is divided in half for testing and training [1,2]. Both datasets are commonly used in machine learning to experiment with text classification and clustering [1]. We processed both datasets in a similar manner: cleaned punctuation, used CountVectrizer() to calculate word occurrence, used tf-idf to calculate word frequency, reduced dimensionality with TruncatedSVD(), and scaled word frequencies to between 0 and 1. The results of our experiments found that after hyperparameters were tuned, logistic regression on the IMDB dataset was the most accurate, followed closely by multinomial naive Bayes on the IMDB dataset. Meanwhile, logistic regression on the 20 news group dataset performed significantly worse, as did multinomial naive Bayes on the 20 news group dataset, which had the lowest accuracy out of the four groups. When we compared the accuracy of the two models as a function of the size of the dataset, we found that accuracy increased as the training set size increased. In that experiment, logistic regression on the IMDB dataset also had the highest accuracy while multinomial naive Bayes on the 20 news group dataset had the lowest accuracy. In exploring other ways to tune our model, we investigated the effect of different ranges of n-gram preprocessing on accuracies, using a basic bag-of-words implementation accuracy as a baseline. We found that the newsgroup accuracy decreased overall with n-grams, and that the IMDB accuracy increased.

Other papers analyzing the 20 news group dataset guided our decisions for pre-processing and inspired us to pursue specific questions in the creativity section of our project. Especially salient was work by Albishre et al., who addressed common issues encountered when cleaning the 20 news group dataset [8]. They noted that punctuation is often overlooked during text cleaning, and suggested that it be removed [8]. They also discussed how pre-processing serves to decrease the dimensionality of a given dataset and make data more uniform, improving both experiment performance and efficiency [8]. We experimented with stop word removal and stemming, but ultimately decided against implementing it in our experiments as it did not perform as well when combined with tf-idf, which already helps to scale down the impact of words that occur very frequently in a given set. The IMDB

dataset is another large dataset that serves as a robust benchmark for work in text classification [10]. Related work using this dataset has suggested the elimination of stop words and the use of N-grams [9]. As such, we decided to explore the use of n-grams in text classification for the creativity part of this project.

Datasets

For the 20 news group dataset, we used the default train subset, removing headers, footers, and quotes, to train the model and then reported the final performance on the test subset. The IMDB dataset required the use of reviews in the train folder for training, while the performance was reported from the test folder data.

Both the 20 news group dataset and the IMDB dataset required similar steps for feature extraction and pre-processing. We started with the raw data and cleaned punctuation before using scikit-learn's CountVectorizer function to convert the text data into numerical feature vectors, which involved calculating word occurrences. Since longer documents would have higher word occurrences, we converted occurrences to frequencies using tf-idf (term frequency times inverse document frequency). Next, we conducted a linear dimensionality reduction by means of truncated singular value decomposition (TruncatedSVD), which works efficiently with sparse matrices [4]. As this was done on term count/tf-idf matrices, it was actually a latent semantic analysis (LSA) [4]. Finally, we scaled word frequencies to [0,1], since negative values cannot be used for the MultinomialNB function.

Results

Task 3.1

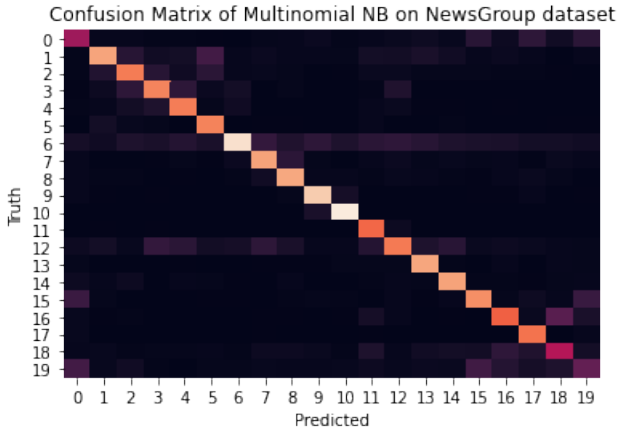
This task involved conducting multiclass classification on the 20 news group dataset and binary classification on the IMDB reviews. Using the multinomial naive Bayes model, we were able to achieve an average accuracy of 0.66 using 5-fold cross-validation on the 20 news group classification, and 0.84 on the IMDB classification. Below are two tables showing the values obtained from 5-fold cross-validation on the 20 news group and IMDB datasets:

5-Fold Cross Validation on 20 News Group with MultinomialNB						
Fold	1	2	3	4	5	Average
Accuracy	0.65030946	0.66003537	0.65030946	0.6489832	0.67860301	0.6576481

5-Fold Cross Validation on IMDB with Multinomial NB						
Fold	1	2	3	4	5	Average
Accuracy	0.8377542	0.84924845	0.86604775	0.83908046	0.82360743	0.843147658

We achieved more accurate results with the IMDB dataset: the testing accuracy from the IMDB dataset was 0.85 whereas the testing accuracy from the 20 news group dataset was 0.61.

To further visualize the predicted outcome of the multinomial naive Bayes model, we generated a confusion matrix to see which labels were frequently being misclassified. From the confusion matrix we see that there seemed to be a higher rate of misclassification near the political categories of class 16-19.



When we were implementing the multinomial naive Bayes model, the first working model had an accuracy of about 0.58 on the newsgroup classification test set, the exact same as the sklearn package implementation (exact same posteriors and likelihoods as well). However, in an attempt to make the predicting method more efficient, we moved the addition of the log-prior outside of the nested loops and just added the scalar to the entire column of the posterior matrix. This change increased our accuracy dramatically, by around 0.1 overall. The binomially distributed IMDB reviews, however, did not change in accuracy.

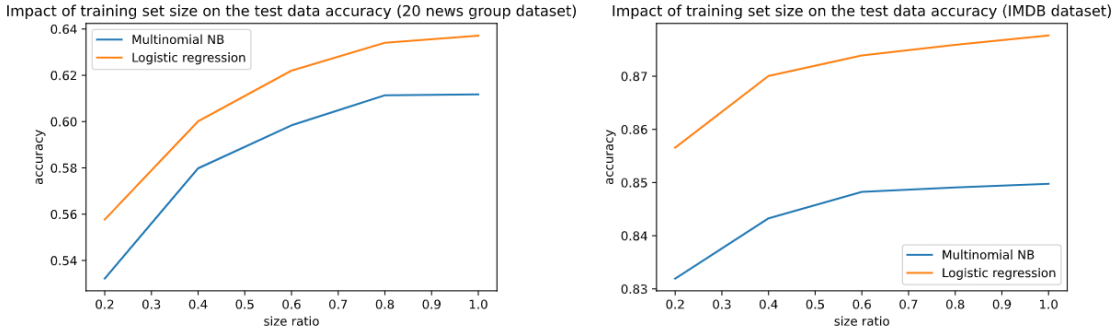
Task 3.2

Task 3.2 involved comparing the performance of multinomial naive Bayes model and logistic regression model on both datasets with their best hyper-parameters. For the multinomial naive Bayes model, we tuned its only hyperparameter "alpha" using 5-fold cross validation. For the logistic regression model, we used "sag" (Stochastic Average Gradient) for gradient descent since it performs faster for large datasets and "L2" for regularization penalty. Then, we tuned its hyperparameter "lambda," which is the inverse strength of regularization (i.e. lower lambda gives higher regularization). Using the best hyperparameters for both models, we observed the accuracy performance of each model on the testing dataset. As shown in the table, we found that the logistic regression model had a higher testing accuracy on both datasets after hyperparameter tuning, and the IMDB dataset had a higher testing accuracy than the 20 news group dataset. The accuracy values were approximately 0.64 for the 20 news group dataset and 0.88 for the IMDB dataset.

```
NB_newsGroup_best_alpha:  30
NB_IMDB_best_alpha:  1e-10
LR_newsGroup_best_lambda:  0.1
LR_IMDB_best_lambda:  1
+-----------------------------------------------------------------------------+
| Best performance of MultinomialNB and Logistic Regression on both testing datasets |
+-------------------------------+-----------------------------+--------------+
|                               |         20 news group       |     IMDB     |
+-------------------------------+-----------------------------+--------------+
|         MultinomialNB         |      0.6117896972915561     |   0.84976    |
|      Logistic Regression      |      0.6371481678173128     |    0.8776    |
+-------------------------------+-----------------------------+--------------+
```

Task 3.3

Task 3.3 involved comparing the accuracy of the two models as a function of the size of training dataset. In this experiment, we selected 20%, 40%, 60%, 80% and 100% of the available training data and trained the models on these subsets. It is worth mentioning that for obtaining the best performance, we also conducted 5-fold cross validation on each subset to find the best hyperparameters. From the graphs we can clearly see that increasing the size of our training dataset can improve the testing accuracy for both multinomial naive Bayes and logistic regression model. This result shows that the size of training dataset has a proportional relationship with the testing accuracy for text classification experiments. However, we can see from the graphs that the rate of change of accuracy decreases as the size of training dataset increases. Thus, the accuracy will probably become nearly stable at a certain value if the size of training dataset keeps increasing.



Creativity

Part 1: Predict rating for IMDB dataset using linear regression instead of binary classification

Instead of binary classification for the IMDB dataset, we conducted an experiment to predict the actual rating values using linear regression. Since the predicted ratings are continuous by the linear regression model, we have rounded the results to their nearest integer. The accuracy performance turns out to be not ideal (0.17368). This result shows us that linear regression may not be a practical model for textual data classification, which explains why it is not typically used for data classification [5].

Part 2: Use the bag of words text embedding method for data pre-processing

Instead of using tf-idf (term frequency times inverse document frequency) for our text embedding method, we used the bag of words text embedding method for data pre-processing. The average accuracy on the 20 news group classification using 5-fold cross-validation was approximately 0.57, and 0.84 on the IMDB classification. We can clearly see that the accuracy performance decreases from 0.66 (shown in Task 3.1) to 0.57 when replacing the tf-idf text embedding method with the bag of words text embedding method.

Part 3: N-grams

As part of pre-processing, we also decided to investigate the usage of n-grams in text classification. We used the ngram-generator built into the sklearn library, specifically the Count Vectorizer. To maintain consistency, we compared the results of various n-gram ranges to the baseline BOW accuracy from part 2 above.

5-Fold C.V. Accuracy values for 20 News Group with Naive Bayes using B.O.W. N-Gram						
N-Gram Range	Fold 1	Fold 2	Fold 3	Fold 4	Fold 5	Average
Base	0.57559682	0.57736516	0.5495137	0.57869142	0.55393457	0.567020332
1, 2	0.49867374	0.49646331	0.45269673	0.47833775	0.45755968	0.476146242
2, 2	0.51149425	0.52122016	0.47701149	0.49381079	0.50044209	0.500795756
1, 3	0.50176835	0.49823165	0.46286472	0.48143236	0.46905393	0.482670202

5-Fold C.V. Accuracy values for IMDB with Naive Bayes using B.O.W. N-Gram						
N-Gram Range	Fold 1	Fold 2	Fold 3	Fold 4	Fold 5	Average
Base	0.83333333	0.84261715	0.84615385	0.83554377	0.84350133	0.840229885
1, 2	0.86206897	0.87798408	0.88107869	0.86781609	0.87135279	0.8720601
2, 2	0.8642794	0.87488948	0.88019452	0.8709107	0.877542	0.87356321
1, 3	0.87179487	0.88284704	0.88682582	0.87533156	0.87886826	0.879133510

Other creativity parts:

- Converting 50000 .txt files of IMDB dataset to two .csv file using python scripts and use .csv files to load the dataset since it will decreases enormously the number of I/O operations which is very expensive in terms of running time.
- Preprocessing the dataset to tf-idf (term frequency times inverse document frequency) instead of the bag of words. This scaled down the impact of words that occur very frequently in a given corpus that are less informative (e.g. “is”, “it”, “a”, “the”, etc.). We tried removing stop words in addition to using tf-idf but obtained a lower accuracy. This can be explained by the fact that tf-idf already accounts for such stop words.
- For decreasing the computation time and memory requirement, we applied dimensionality reduction on both datasets using TruncatedSVD and kept 500 features for each dataset. For latent semantic analysis (LSA), the recommended number of features after dimensionality reduction is 100 [4]. However, the accuracy performance was not ideal when we reduced to 100 features. Thus, after considering the trade-off between the computation time and the accuracy performance, we picked 500 as the desired dimensionality. After applying the dimensionality reduction, our accuracy performance was still very close to the accuracy given by the MultinomialNB model from the sklearn package.

- We used sklearn's SGDClassifier function to apply stochastic gradient descent to the 20 news group dataset. We also ran an exhaustive search of the best parameter combinations using the GridSearchCV function, ultimately finding that the highest test accuracy, 0.73, was obtained when $\alpha = 0.001$ was used and tf-idf was applied.

Discussion and Conclusion

Our data and results demonstrated that for both multinomial and binomial classification of textual data, logistic regression outperformed naive Bayes by a considerable margin. We also found that logistic regression took longer to train than naive Bayes. Although both had a relatively fast predict time, the naive Bayes method could require more memory to predict depending on its implementation (i.e. if all posteriors are returned). We demonstrated in Part 3.2 that the number of classes impacts how large α has to be to make a difference. In the multinomially distributed newsgroup dataset with 20 classes, Laplace smoothing ($\alpha = 1$) had little to no impact on the accuracy, and through tuning, we found the optimal value was $\alpha = 30$. On the other hand, the binomially distributed IMDB dataset had an optimal smoothing value of $\alpha = 1e - 10$. From these optimal hyperparameters, we compared the accuracies of both models on the two datasets. Logistic regression outperforms naive Bayes in both datasets, and the IMDB binomially distributed data were more accurately labeled for both models.

In the experiment of Task 3.3, we observed that the size of the training dataset had a proportional relationship with the testing accuracy for text classification experiments. However, this relationship does not appear linear based on our experiments: the rate of change of accuracy seems to decrease as the size of the training dataset increases. Therefore, it's not possible to obtain a 100% accuracy even if we have a large enough training dataset.

In the creativity experiment using n-grams we observed that the use of n-grams slightly decreased accuracy for the multinomially distributed news group dataset, but increased the accuracy of the binomially distributed IMDB reviews. This could be due to overfitting 2- and 3-grams to one class in the multinomial distribution, which could be misclassified because many categories are similar to each other. For example, there are 5 computers and 3 politics classes, and similar classes would likely have similar groupings of words. We observed already some misclassification within similar classes from the confusion matrix earlier. A binomial distribution would not suffer from the same overfitting and only increase accuracy from the n-grouping, explaining the results from creative part 3.

Possible future experiments could be conducted with regards to the n-gram modeling. Although n-grams decrease accuracy when n gets too large, we did notice that the same n-gram range, if processed using the tf-idf Vectorizer, has a higher accuracy than the n-grams processed by the Count Vectorizer. Out of limitations of RAM, we were unable to process the n-grams with SVD reduction and scaling in the same way we processed data for our main models, so we decided to compare to the bag of words baseline outlined in part 2 of the creativity section for consistency. Another direction experiments could take would be to investigate the effect of different tokenization methods. In our research, we found tokenizers and lemmatizers that could break text down by their grammar semantic [7]. Such methods are extensively used in NLP, however text and sentiment classification would benefit from such pre-processing.

Statement of Contributions

Task 1: Ziqi
Task 2: Alvin, Ziqi, Emily
Task 3 and Creativity: Alvin, Ziqi, Emily
Report: Alvin, Ziqi, Emily

References

[1] SciKit Learn. (2007). Working With Text Data.
https://scikit-learn.org/stable/tutorial/text_analytics/working_with_text_data.html

[2] Maas, A. L., Daly, R. E., Pham, P. T., Huang, D., Ng, A. Y., & Potts, C. (2011). Learning Word Vectors for Sentiment Analysis. *The 49th Annual Meeting of the Association for Computational Linguistics*.
http://ai.stanford.edu/ amaas/data/sentiment/

[3] Shaikh, J. (2017, July 23). Machine Learning, NLP: Text Classification using scikit-learn, python, and NLTK.
https://towardsdatascience.com/machine-learning-nlp-text-classification-using-scikit-learn-python-and-nltk-c52b92a7c73a

[4] SciKit Learn. (2007). sklearn.decomposition.TruncatedSVD.
https://scikit-learn.org/stable/modules/generated/sklearn.decomposition.TruncatedSVD.html

[5] Jing, H. (2019, May 17). Why linear regression is not suitable for classification.
https://towardsdatascience.com/why-linear-regression-is-not-suitable-for-binary-classification-c64457be8e28

[6] Ng, A. Y. (2001). On Discriminative vs. Generative classifiers: A comparison of logistic regression and naive Bayes. https://papers.nips.cc/paper/2001/file/7b7a53e239400a13bd6be6c91c4f6c4e-Paper.pdf

[7] The Stanford Natural Language Processing Group. (2009, April 7). Stemming and lemmatization.
https://nlp.stanford.edu/IR-book/html/htmledition/stemming-and-lemmatization-1.html

[8] Albishre, K., Albathan, M., & Li, Y. (2015). Effective 20 Newsgroups Dataset Cleaning. *International Conference on Web Intelligence and Intelligent Agent Technology*, pp. 98-101, doi: 10.1109/WI-IAT.2015.90

[9] Li, A. (2019, December). Sentiment Analysis for IMDb Movie Review. https://www.andrew.cmu.edu/user /angli2/li2019sentiment.pdf

[10] Dey, L., Chakraborty, S., Biswas, A., Bose, B., & Tiwari, S. (2016). Sentiment analysis of review datasets using naive bayes and k-nn classifier. *arXiv preprint*, arXiv:1610.09982.