Comp551 Project2 Writeup

Sibo Yang(260906371), Rui Song(260890776), Jeff Zhang(260840033)

February 2021

Abstract

In this project we first implement the machine learning model Naive Bayes classifier and the K-fold cross validator. Next, we compare the performance of Naive Bayes classifier and Logistic Regression on two benchmark datasets: the 20 news group dataset and the IMDB reviews dataset. Based on our results, we conclude that Logistic Regression overall achieves better performance. Further, in order to obtain a set of parameters for optimizing the models, we examine how different hyper-parameters, such as the smoothing factor and the various hyperparameters for Logistic Regression, could affect test accuracy by plotting accuracy against the hyper-parameter values. Further, in order to better investigate which hyperparameter value is the best, we take advantage of K-fold cross validation. Finally, we examine how different sizes of training data could affect the accuracy of our models.

Introduction

Guided by the conceptual knowledge of Naive Bayes classifier and Logistic Regression given in class, we launch an investigation into the detailed implementations of these algorithms as well as how the performance of these algorithms could depend on varying hyper-parameter values and sizes of training data. Our investigation is based on two open-source datasets. The first one, the 20 news group dataset, contains 2000 news reports that were classified into 20 news categories. The second dataset, which is the IMDB reviews dataset, consists of 50,000 movie reviews that are classified as either positive or negative. The 20 news group dataset is drawn from the Scikit-Learn website. [3] The IMDB reviews dataset is drawn from the Stanford CS website [4] In the end, we also compare the performance of Gaussian Naive Bayes against that of the Logistic Regression to identify the more accurate approach, given the two real-world datasets. In our analysis of the Naive Bayes hyper-parameter test results that involved a discussion of the famous Laplace smoothing technique, we cite X. Ding and F. He's paper to bring about a non-conventional perspective to the idea of smoothing. [2] Similar to what Seyyed Dadgar established in his 2016 paper, we observe that TF-IDF processing is extremely effective in terms of accuracy improvement. [1]

Datasets

We import the 20 news group data from sklearn.datasets and used it directly. For the IMDB Reviews dataset, we downloaded the text documents from stanford IMDB Reviews dataset and for convenience, we wrote a python script to transform the text documents to two csv files (train.csv and test.csv).

20 News Group Dataset

This dataset is a collection of about 20,00 news group documents which have been approximately evenly divided into 20 groups (as shown in Figure 1) such as sports, religion, politics etc. We applied text cleaning techniques like removing stopwords, removing html tags, removing punctuations to the training set of both dataset.

IMDB Review Dataset

This dataset contains 50,000 movie reviews which are classified to either positive or negative evenly. A positive review has a rating >=7 out of 10, and a negative review has a rating <= 4

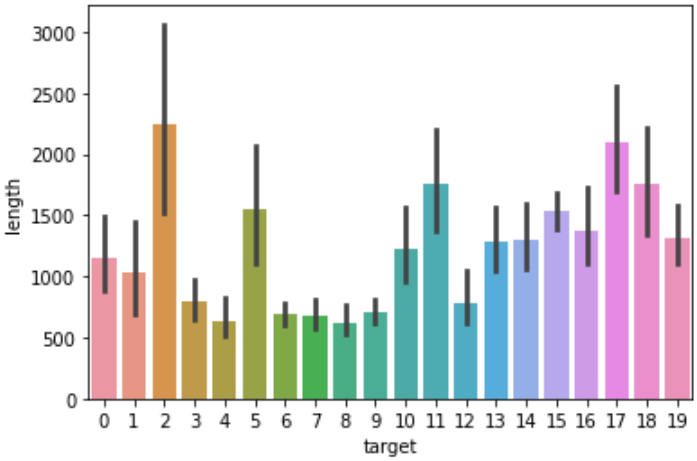

Figure 1: Class distribution of 20 news group dataset

Results

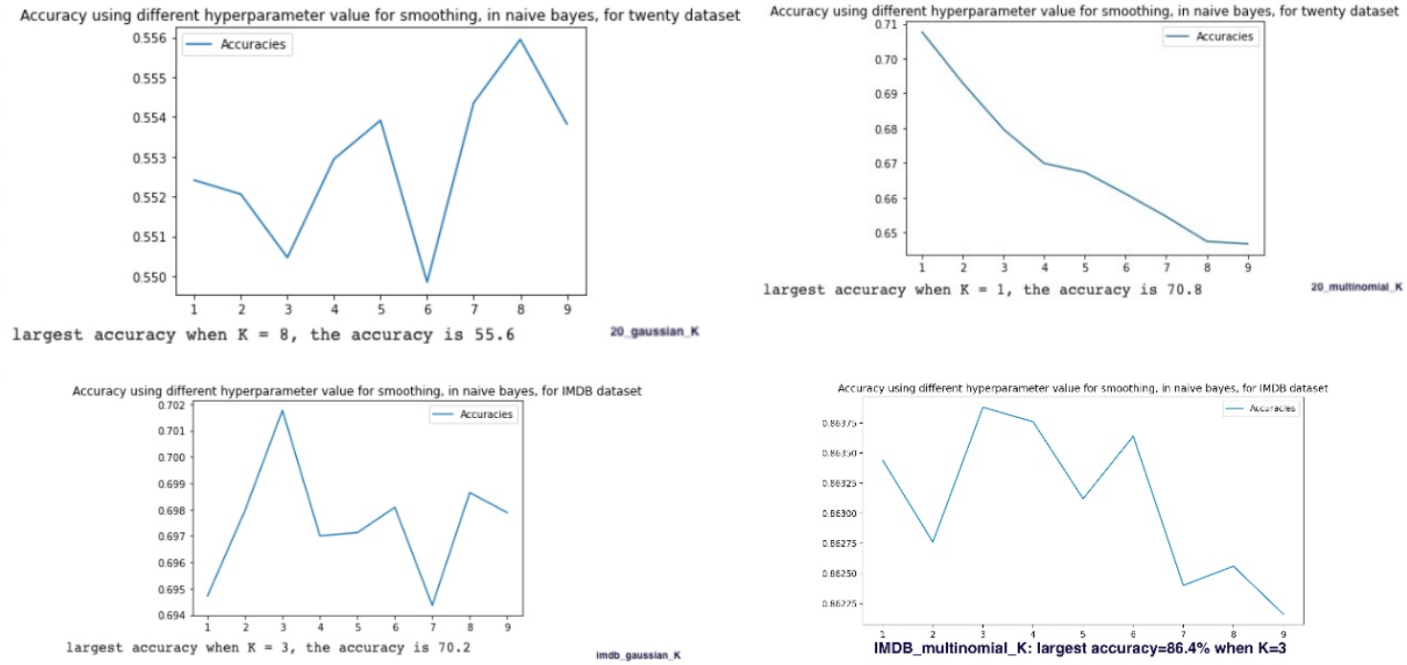

Figure 2: Top left: Different K values and the performance of Gaussian naive bayes, using the 20 news group dataset; Top right: Performance of Multinomial Naive Bayes against K, 20 news group dataset; Bottom left: Performance of Gaussian Naive Bayes against K, IMDB dataset; Bottom right: Performance of Multinomial Naive Bayes against K, IMDB dataset

Figure 2: As the first part of our experiments, we test which hyperparameter values could help our models achieve their best performance. Here, we implemented five-fold cross validation method to help us estimate the accuracies of each model. On the 20 news group dataset, we discovered that the smoothing parameter value of 8 will give Gaussian Naive Bayes model the highest accuracy of 55.6 percent. In comparison, the Multinomial Naive Bayes model provided by Scikit Learn could achieve a optimal accuracy of 70.8 percent when K=1. On the IMDB reviews dataset, we notice that the predictions are significantly more accurate. Our Gassian Naive Bayes model could attain an accuracy of 70.2 percent when the smoothing factor equals 2. In contrast, Scikit Learn's Multinomial Naive Bayes model has a peak accuracy of 86.4 percent when K=3. For the Logistic regression model, we decided to vary two hyper-parameters' values and test which pair of hyper-parameter values will yield the most

accurate model. One hyper-parameter is the solver used, the other one is the inverse of regularization strength (c). On the 20 news group dataset, we discovered that when using "newton-cg" solver and c=10 will give the highest validation accuracy. On the IMDB reviews dataset, using "liblinear" as solver and setting c to 10 will give the best model. We also tried using the set of features which CountVectorizer produces from the data before text cleaning to train the Multinomial Naive Bayes model, and we observed a validation accuracy of 64 percent when k = 1, which is lower than the accuracy of the model we trained with cleaned data.

	20 News Group Dataset	IMDB reviews dataset
Gaussian Naive Bayes	55.6%	70.2%
Multinomial Naive Bayes	70.8%	84%
Logistic Regression	66.6%	85.1%
Winner	Multinomial Naive Bayes	Logistic Regression

Figure 3

Figure 3: In the second part of our experiment, we build on top of our results from part1 of our experiments and make a comparison across various models to find out the most accurate one. Here, we estimated the accuracies using test set, and used all the training data provided to train models. Our conclusion is that Multinomial Naive Bayes is the fittest for the 20 news group dataset while Logistic Regression is the most appropriate for the IMDB Reviews dataset.

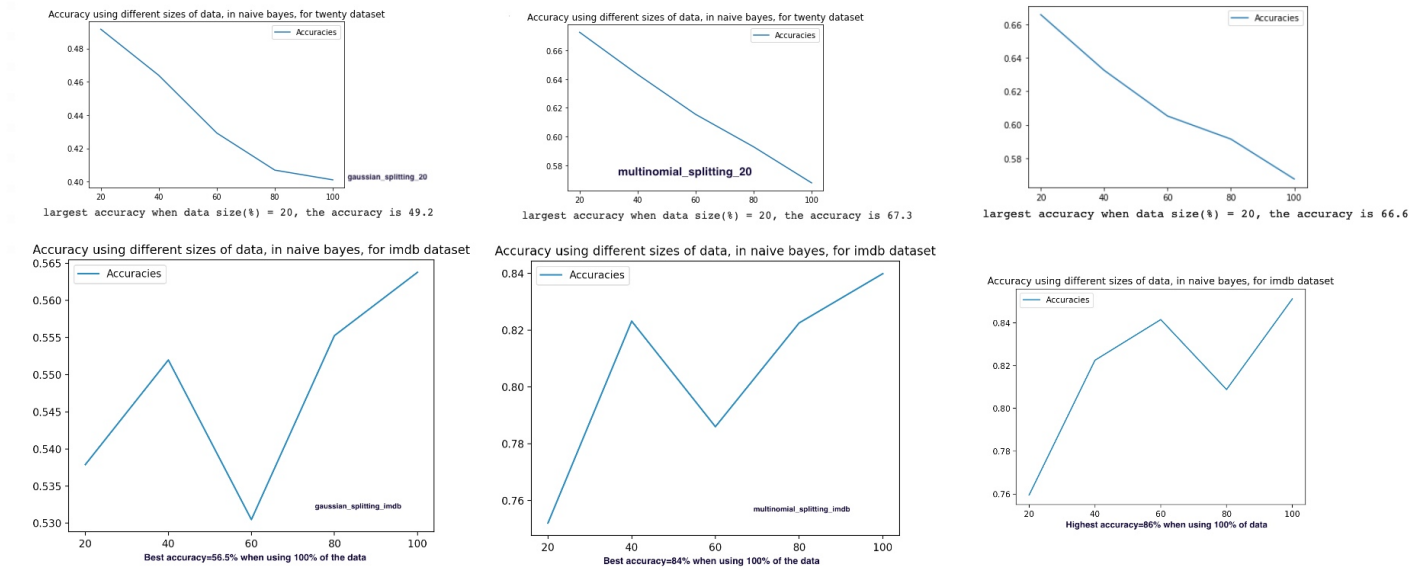

Figure 4: Varying sizes of train data and the models' corresponding accuracies; Top to bottom: 20 news group dataset $->$ IMDB reviews dataset; Left to right: Gaussian Naive Bayes $->$ Multinomial Naive Bayes $->$ Logistic Regression

Figure 4: Here we vary the size of the train sets of our models and test how each model performs under these conditions. We observe similar trends across different models in this experiment on the 20 news group dataset. That is: our Naive Bayes models tend to predict with high accuracy when using relatively small sizes of train data, despite that the common knowledge that models tend to perform better with more training data and the fact that the smaller sets of train data were selected in a completely randomized manner. On the other hand, our experiments on the IMDB Reviews dataset have results that are more aligned with the expected scenario. It could be observed from the figures that more data, in general, improves the prediction accuracies of our models. Additionally, in both datasets we notice that Logistic Regression finish with better prediction accuracy.

```
Coefficients:
 [ -5.65126048  -9.2368087   28.76856363 ...   8.36775388 -21.54192665
  30.75895458]
Mean squared error: 112.96
Coefficient of determination: -8.27
```

Figure 5

Figure 5: For predicting ratings of IMDB views, we try to use Linear Regression. And we use two method, mean squared error and R2 score, to evaluate the result. We can see that the mean squared error is so big, which implies that the error of our prediction is big. And unexpectedly the coefficient of determination is negative, which demonstrates that the prediction from the linear model is even worse than just choosing the mean value as prediction, which is another result to show the model is not considerable.

```
# Filters to be executed in pipeline
CLEAN_FILTERS = [strip_tags,
                 strip_numeric,
                 strip_punctuation,
                 strip_multiple_whitespaces,
                 transform_to_lower,
                 remove_stopwords,
                 remove_single_char]
```

Figure 6: The feature filter that we used

Figure 6 on the feature subset we used to improve performance: After turning the input text data into a "count vector", we further applied a set of filters to strip off the less informative keywords in the text data. As seen from the figure, we only preserve the frequencies of those words that are not "stop words" or "punctuation" or "numeric". As these features/words are not representative of what a text is truly about, they hardly indicate anything about the true type of a text data.

Discussion and Conclusion

One conclusion we could draw from our experiment results is that the widely known Laplace "add one" smoothing may not be the most optimal as a choice of hyper-parameter value. In fact, as seen from our testings, higher values of smoothing factor often achieve remarkably higher prediction accuracies. Indeed as F He and X Ding pointed out in their 2007 paper that other smoothing methods "can achieve better and more stable performance than Laplace smoothing". [2]

Another key take-away from our experiments is that data pre-processing could play a central role when it comes to accuracy improvement. Without pre-processing the text data, our models attain accuracy that is less than 50 percent of what is possible. We based our text pre-processing on two main components: The first is to divide the number of occurrences of each word in a document by the total number of words in the document, so that documents with more words (but on the same topic) don't skew our predictions. The second refinement is to downscale weights for words that occur in many documents and are therefore less informative. Furthermore, we specially remove "stop-words" in the English language to feed our models with more informative data. Also, we use two argument in CountVectorizer, maxdf and mindf, the former one is to guarantee that some words occur in almost every email would not be considered, the latter one is to prevent the over-fitting.

We have also observed that more train data sometimes don't result in the expected enhancement in test accuracy. This contradiction remains unexplained even after careful investigation. Nevertheless, it points us in the direction of a possible future improvement.

For predicting the rating of IMDB, it's not a good way to use Linear Regression Model, as we shown before. It shows that the data is not distributed linearly so we should try to use nonlinear basis to predict such as Gaussian, Polynomial,etc. However, these all take some time to finish, which points out to us that we may find some more efficient ways to make the prediction.

Statement of contributions

Everyone contributed to model implementation, experimenting, data processing and writing the writeup.

References

[1] Seyyed Mohammad Hossein Dadgar. *A novel text mining approach based on TF-IDF and Support Vector Machine for news classification.* IEEE, 2016.

[2] X Ding F He. *Improving Naive Bayes Text Classifier Using Smoothing Methods.* Springer, 2007.

[3] Scikit Learn. *20 news group dataset.* Scikit Learn, 2017.

[4] Andrew Maas. *IMDB reviews dataset.* Stanford University, 2011.