COMP 551 Mini Porject 2 Write-up

Group 33: Kaicheng Yan, Yuxuan Liu, Junxiang Mao

March 9, 2022

Abstraction

In this project, we investigate the performance of three machine learning models covered in class (K-nearest Neighbor model, Naive Bayes model and Logistic regression) on two benchmark datasets (20news group and 140 sentiment data). Since these two datasets are both text data, we use the bags of words (BOW) representation to extract numerical feature from the text for further training. During experiment, we used cross validation for selecting hyperparameter of each model to acquire best average performace. By feeding these three models with processed training data splitted from the original datasets, we acquire the predict test labels and compare it with true labels to get each model's accuracy. After experimenting models with their best hyperparameters, we found generally logistic regression model achieves better performance than rest of models we test. We also find that increasing training sample is usually with the increase of accuracy .

Introduction

Basic Introduction

Document classification is the task of using trained model to automatically assign documents to a fixed number of categories and is widely used in many text applications. In our project, our main effort is to compare the performance of three ML models (K-nearest Neighbor model, Naive Bayes[1] model and Logistic regression) on this multiclass classification task after trained by two different datasets. The two datasets used in this process are two large text packages in which each text has a assigned label(sentiment/no sentiment or news category). Considering such large data load, the slice of original data and the application of data preprocessing is necessary, so for the dataset 2(sentiment 140) we slice only the 20 percent of the original data as our training part for implementation efficiency and also compare the impact of trying several basic text preprocessing methods. As for tuning hyperparameters, we used 5-fold cross validation to estimate performance of different hyperparameters on these learning algorithms and configure the model with the parameter which achieved highest average accuracy. After feeding unpreprocessed raw training data to our models, we observed on two datasets, logistic regression models usually has higher accuracy result.

[1] It is also important to indicate that during data processing stage, since we only use CountVectorizer() without tfidf(), it would return discrete data (the counts of each word). Thus, it is more appropriate for us to use multinomial Naive bayes as our bayes model without using Gaussian NB.

Related Work

In the article ”The influence of preprocessing on text classification using a bag-of-words representation” (1), the authors explores the impact of systematically using combinations of five/six basic preprocessing methods on their text classification using ML methods, and they find for all the datasets processed, there was always at least one combination of basic preprocessing methods that could be recommended to improve the accuracy results when using a bag-of-words representation. Inspired by this article and considering the huge amount of features(words) the two assigned datasets contain, we decide to use the result got from the unpreprocessed data as baseline, and examine if new features (words) filtered by different text preprocessing method could improve the final accuracy result of each ML algorithm. Finally, we find compared with baseline raw data, removing stop words and doing lemmatization to the text data shows limited improvement on both models in terms of accuracy.

Datasets

In this project we train and test our models with two different datasets: the first one is the 20 news groups datasets from scikit-learn, which contains 20 categories of news; the second one is the sentiment140 dataset, which records a huge amount of tweets together with their corresponding labels (the label 0 represents negative tweet, and the label 4 represents positive tweet) and for this dataset, we did an extra step of converting the label 4 into 1, which results in easier use by our models. For both datasets, we use the ”bages of words representation” to extract features, i.e. each single word appears in original data is considered a feature.

For the most basic data preprocessing, we convert all uppercase letters in both datasets into lowercase, and then remove punctuations to simplify our process work later on. We then go one step further by removing all stopwords appear in both datasets since these words are obviously uninformative and will not influence our prediction accuracy, and lemmatizing all words in these two datasets. We consider these extracted words (features) as our ”new features”. This step eliminates uninformative words and converts all possible words to their primitive forms, resulting in less ”noisy” features and will boost the performance of models.

After analyzing the datasets, we found that in dataset 1, the news samples for each category are not evenly distributed, where label 7 has the largest amount of samples (Figure 1), and label 19 has the least amount of samples. When sampling this as our training data, it may cause errors in the prediction of our models. For dataset 2, we found that the numbers of tweets in both labels are similar (Figure 2), so compared with dataset 1, this distribution is more ideal for our future training.

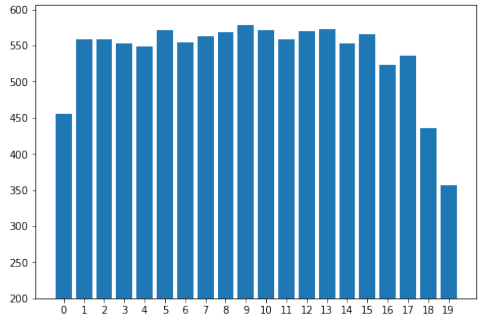


Figure 1: 20 News Group Data Distribution

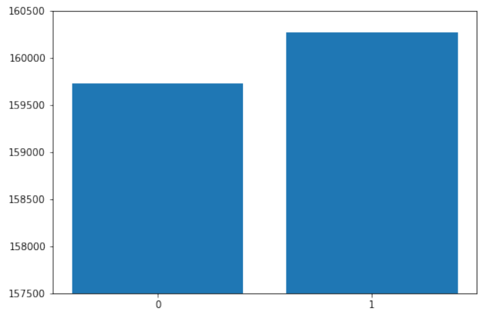
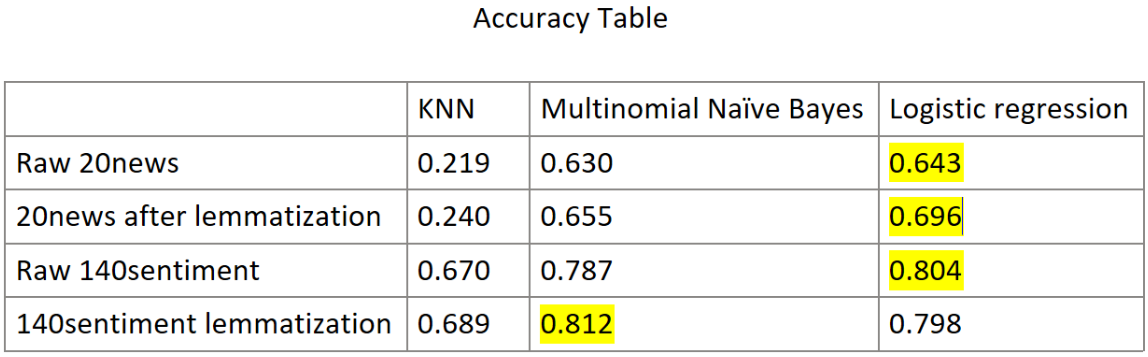


Figure 2: Sentiment 140 Distribution

Results

Firstly, we set distinct hyperparameter lists for each ML model:
As for K-Neartest Neighbor model, we change its K's value, for Multinomial naive bayes, we try different smoothing factor: alpha value and for logistic regression algorithm, we change its solver type and penalty strength (C values) with fixed L2 Regularization (penalty) and 1000 maximum iteration limit. For each model, we use 5-fold cross-validation to the given model with different hyper-parameter settings, select the best hyperparameter by the best average accuracy result.
From the result the fuction gives, for 20news group data, we set Multinomial NB 's smoothing factor (alpha) as 1, set KNN model's number of neighbors as 4. However, due to the time limit, we failed to get best hyperparameters of logistic regression, so we set temporal solver and penalty strength (C value) of this model for future test. (The related code is done, so we can do it in the future); and for sentiment140 data, we configure each model with same best hyperparameters as first dataset .

After tunning hyperparameters for each model, we then feed them with unpreprosessed two dataset and preprocessed dataset by different method and record their accuracy result respectively, and here is the result table (We highlight the winner model of each data):



Accuracy Table

	KNN	Multinomial Naïve Bayes	Logistic regression
Raw 20news	0.219	0.630	0.643
20news after lemmatization	0.240	0.655	0.696
Raw 140sentiment	0.670	0.787	0.804
140sentiment lemmatization	0.689	0.812	0.798

We can see that generally logistic regression achieves better performance than other modles we test.

Then we discorded the impact of different size of training data sample for four models. For doing that, we randomly select 20%; 40%; 60% and 80% of the available training data and train your modelon this subset, and here is the result plot:

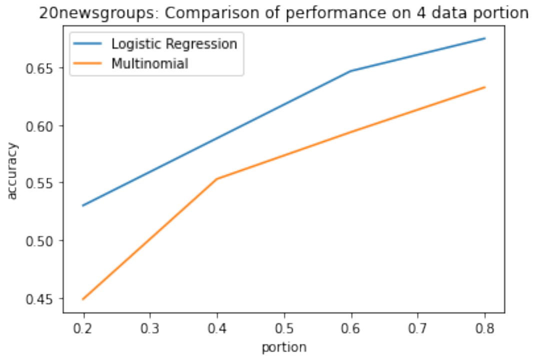


Figure 3: 20news group data

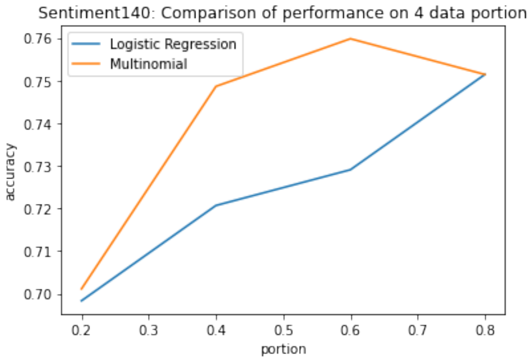


Figure 4: sentiment140 data

It is easy to see that as we increase the portion of training data sample, the performance of both models raise also, except for the Multinomial naive bayes's performance on the sentiment140 data, it turn to decrease when the training sample increase to 60% portrion of original data.

Discussion and Conclusion

Therefore, we can see from this project that different machine learning models have distinct predictions using different datasets, and that tuning hyperparameters and preprocessing of text data can greatly improve the final results. Since in both two test dataset, the logistic regression algorithm achieves better performance than the Multinomial Naive bayes and KNN model. We hypothesize this because we think one feature(word)'s appearance is not independent with another word's appearance since a bunch of words are usually bundled together under one category. In future investigation, we could try more machine learning algorithm without independent features assumption and compare the results with Multinomial Bayes model to justify our hypothesis.

In addition, we see if training our model under bigger data sample, it would acquire better performance. However, it is interesting to see this general accuracy raising does not occur when we test Multinomial NB model on the sentiment140 data.

Statement of Contributions

In this project, we break the tasks down to the following parts: Yuxuan is in charge of handling task 1, i.e. she acquires, pre-processes, analyzes, and cleans the original datasets; She also implements K-fold cross validation and implement experiment code. Kaicheng implements the Multinomial Naive Bayes model and Junxiang is in charge of running the experiments; and the whole group writes this report together.

References

[1] Y. HaCohen-Kerner, D. Miller, and Y. Yigal, "The influence of preprocessing on text classification using a bag-of-words representation," *PloS one*, vol. 15, no. 5, p. e0232525, 2020.