Comparison the performance of the scratched code of KNN with the pre-existing library

Ryutaro Takanami

January 2020

1 Abstract

This paper aims to implement a scratched code of a machine learning classification algorithm and analyze differences of results between the code and the pre-existing library. The library is pre-installed class in Python to implement the algorithm easily. In order to analyze results, this research compares the performance of each program in different parameter and their calculation time to its prediction in different amount of data. Accuracy, precision and recall are used as performance indicators.

2 Introduction

this research adopt Wine Quality data-set as a data-set in order to train and predict labels, which is provided on UCI Machine Learning Repository (1). Wine Quality data-set evaluates variants of the Portuguese "Vinho Verde" wine with the levels from 0 to 10: level 0 indicates poor quality and 10 means the highest quality. It has 11 numerical columns which describe chemical characteristics of wine. For example, pH, density and alcohol content except for the label column which evaluates the quality of the wine. There are both red and white wine data-sets and the amount of data is 1599 and 4897 respectly. This research combines these data-sets and produces the data-set which has 6497 data. About the label, the quality of 6 is converted into 1 and the other into 0 since some qualities do not exist actually and around half data, 2836 data, are 6. When data is imbalanced, it should be treated generally because imbalanced label might make hard to predict because of the shortage of minority label. Therefore, values in quality column are converted (label 0: 3661 data, label 1: 2836 data)(2).

In order to explain the content of this research, this paper is split in five sections. At first, this section introduces the outline of this research. The next section, Methods, explains a classification algorithm which this paper chose and the way to analyze the result of both the scratched program and the library. Moreover, the Methods section shows the way of validation. In the Results section, Graphs of each result are illustrated and described briefly. The fourth section discusses the cause of difference results between the scratched program and the library. Finally, this paper concludes the results of the analysis.

3 Methodology

As a machine learning classification algorithm, this research implements K-Nearest Neighbor (KNN). Classification is a supervised learning which has labeled data to identify the answer of categories (in this case, labels are the quality of the wine) in order to classify data appropriately. After learning pattern of the classification with training data, the classifier predicts labels of test data which has no labels. KNN is one of the classification algorithms and classifies labels with close trained data. When KNN receives the new data, it predicts the label as the majority of labels which close data has. In order to calculate the distance between the new data and trained data, Euclidean distance is used and the number of trained data which is taken account as close data depends on the parameter K. The parameter K is decided manually, and this paper examines the change of accuracy, precision and recall with the different number of K. That is why this algorithm is called K-Nearest Neighbor(3).

After calculating accuracy, precision and recall, results are validated with the K-fold Cross Validation. This method prevents the overfitting and splits a data-set into K data-sets. Overfitting means the algorithm learns the data too well for the specific test data and the accuracy is extremely high in spite of the accuracy rate is low when it predicts the other test data. K is a parameter and could be selected the number freely but 5 is chosen generally because it is enough to validate. A separated data becomes a test data and others become a train data. After that, the other data in train data become a test data and others including the previous test data become a train data. This repeats *K-1* times and every data becomes a test data once. Therefore, through K times validation with different data, overfitting is prevented(2).

This research implements programs to obtain results of accuracy, precision, recall and times with Python and analyzes those data using R. KNN is implemented as a class of Python which has a function to predict labels as a list. Labels are calculated one by one in the predict function. Initially, Euclidean distances against

all train data are computed as in the equation below and sorted in ascending order. Subsequently, K data are used to count the number of labels and a predicted label is decided as the majority of labels. When there are several labels of the same amount, the smallest label becomes a predicted label(3).

$$D(p,q) = \sqrt{\sum_{i=1}^{n}(q_i - p_i)^2} \tag{1}$$

In order to ensure the appropriate parameter K, this paper changes the number of K from 1 to 500 and examines accuracy rates, precision and recall in both the scratched program and the library. Accuracy, precision and recall can be calculated in equation below.

$$Accuracy = (TruePositive + TrueNegative) \div All \tag{2}$$

$$Precision = TruePositive \div (TruePositive + FalsePositive) \tag{3}$$

$$Recall = TruePositive \div (TruePositive + FalseNegative) \tag{4}$$

The amount of time of computing is then examined 100 times when the parameter is optimized as it obtains the highest accuracy rate. Times are recorded for both the scratched program and the library to compare them in R.

After obtaining results in Python, these are analyzed in R. The data are visualized using a library of ggplot to compare the results both the scratched program and the library.

4 Results

Accuracy precision and recall in both the scratched and the library were the same when the parameter of *weights* is default in the library. The library of KNN can select in two types of parameters about the method to decide the predicted label, *uniform* and *distance*. The default parameter is *uniform* and that means all points of test data in each label are weighted equally like the scratched program in this report. According to the official description, there is no mention about the method to decide the label in the case some labels are same number, but the library seems select the smallest label because accuracy rates are the same as the scratched program (4). The other parameter is *distance* which weights each point of the test data by the inverse of their distance. In this method, closer labels have a greater influence than labels which are further away. In Figure 2, the library selected "distance" to ensure the difference from the scratched program.

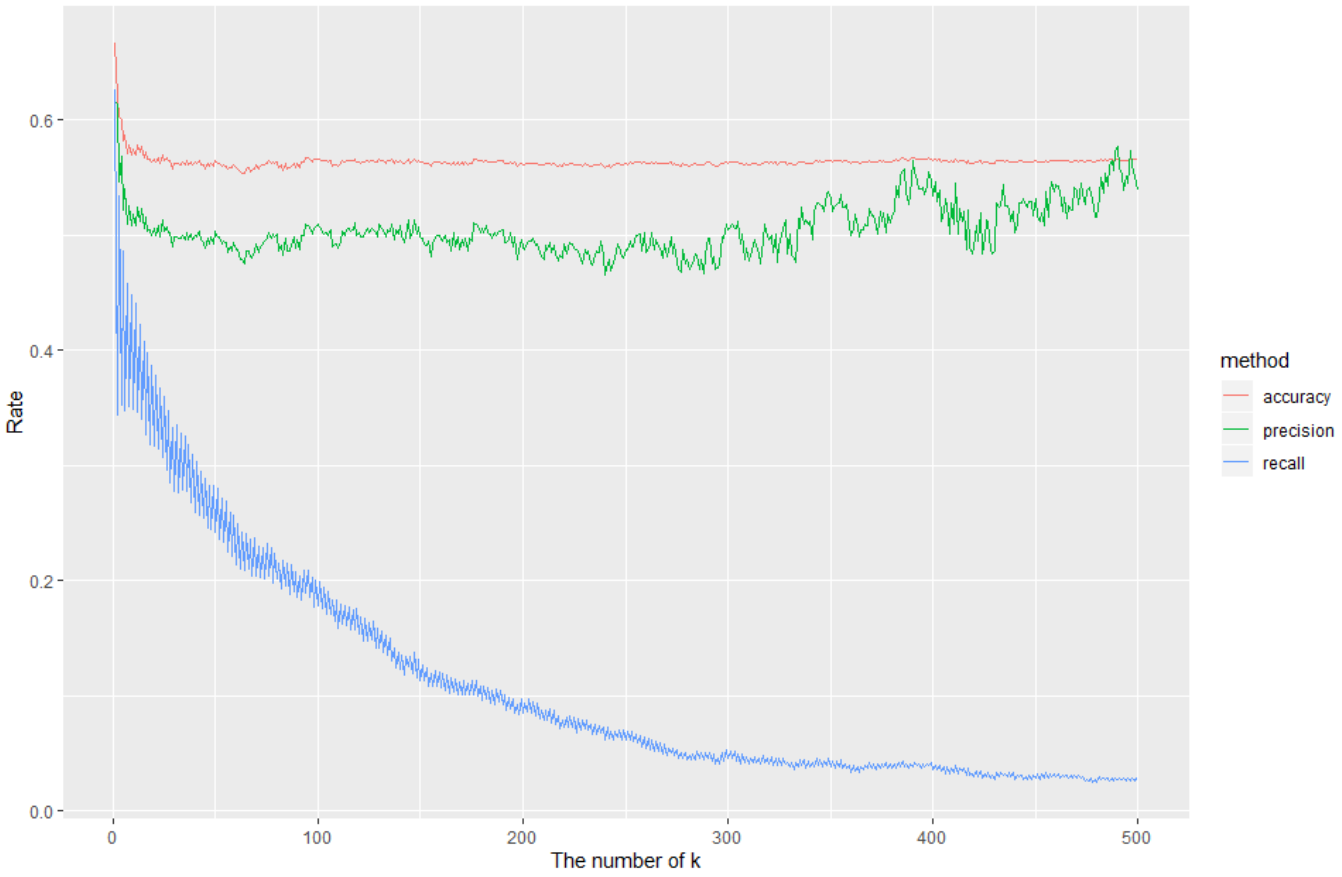

Figure 1: The results of the scratched program

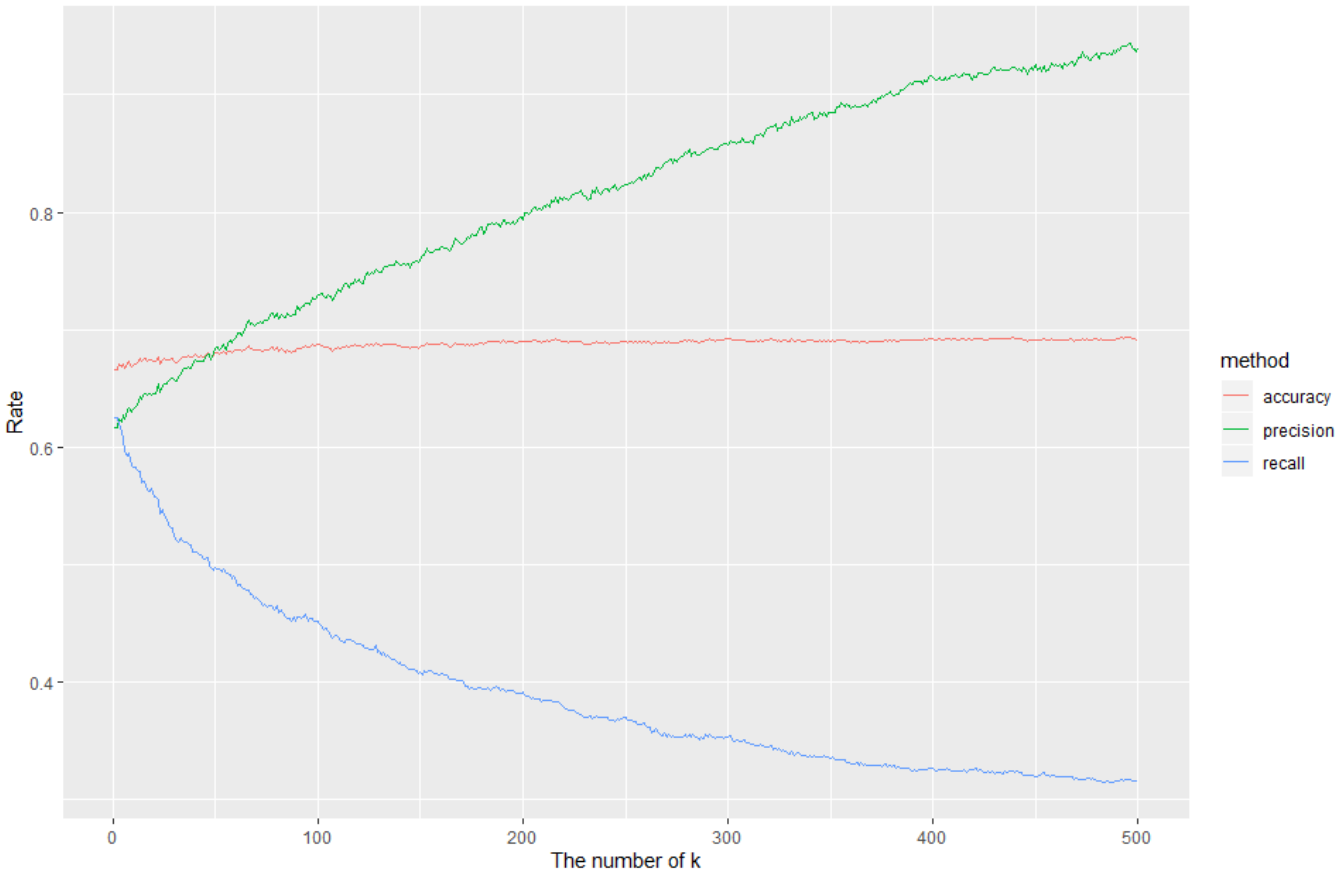

Figure 2: The results of the library

Accuracies of both the scratched program and the library were stable in around 60% and the accuracy of the library was higher than the scratched one. However, changes of the precision were different since the precision of scratched program rose slightly compared to the precision of the library which increased sharply. With regard to recalls, trends of both were same but the library obtained higher recall. Based on results of accuracy, it can be considered that the optimized parameter K in KNN is 1.

Figure 3 illustrates changes of the computing time in both the scratched program and the library when the parameter K was 1. The parameter of the library was set as *uniform* and vertical lines at each point are standard deviation since this experiment was repeated 100 times. Results of computing time were obviously different since the time of scratched program rose exponentially compared to the library, which increased gradually.

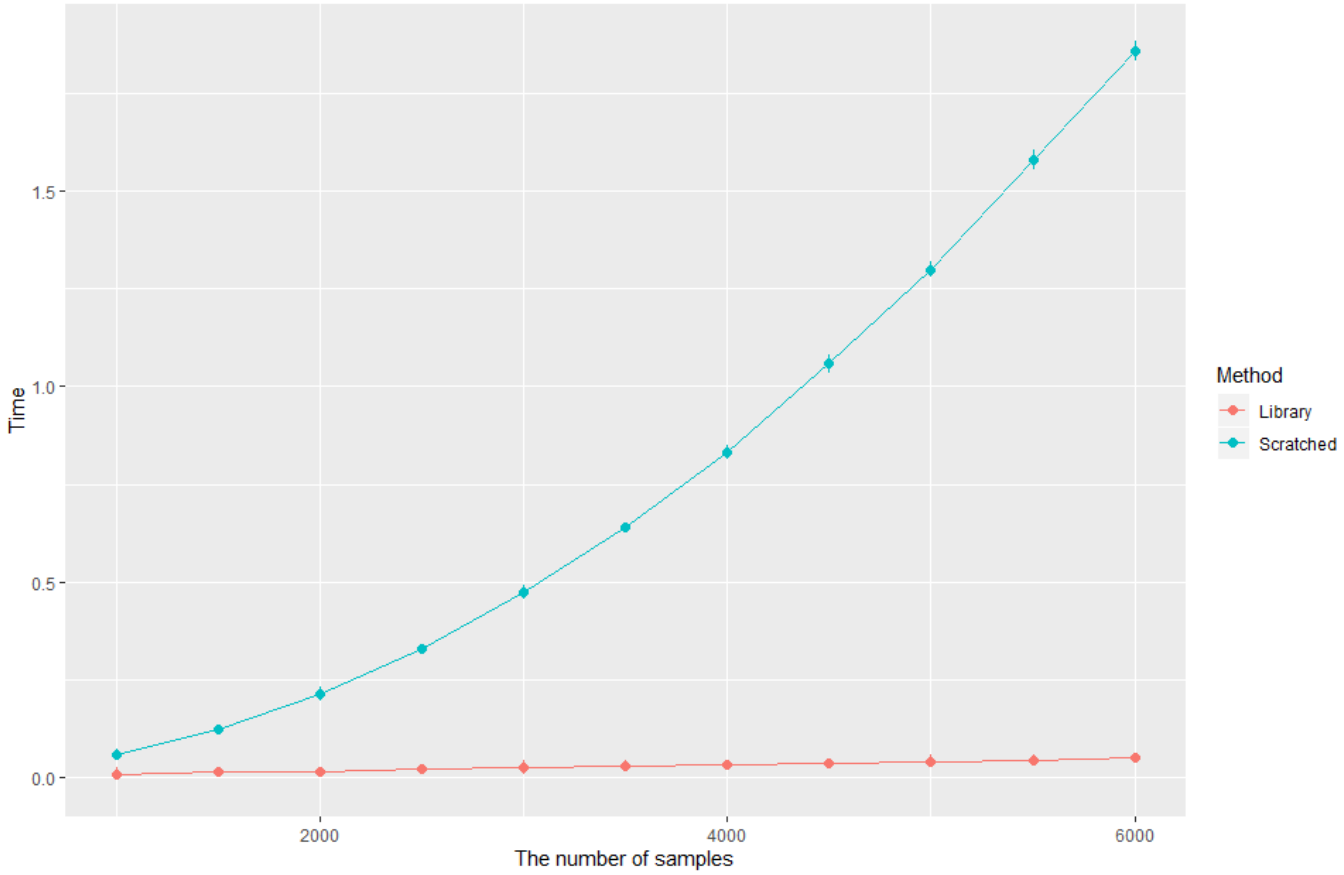

Figure 3: The amount of computing time by the number of samples

5 Discussion

The accuracy of the library was much higher than the result of the scratched program. In addition, the maximum accuracy of the library was 0.6933939 compare with 0.6663042 with the scratched program. Different results could be because the library computes distances as inversed distances and this method seems to be suitable to this data-set compared with simple Euclidean distance. Precision indicates the percentage of correct label in the algorithm predicts as the positive value (in this case, predict as 1). For the scratched program, the precision was lower than accuracy: the program tended to predict as 0 rather than 1. The precision of the library was higher than accuracy overall and the percentage of correct answers was higher as the parameter K is bigger. However, as we can see in the recall, the predicted value tended to 0 and there were many cases where the algorithm mistook 0 for 1 because recall indicates the degree of answering correctly about positive value.

Computing times increased by both programs, but the difference were expanded as the number of samples were bigger. The scratched program was created only for predicting as KNN but the library seems to calculate efficiently by using a mathematical technique. Both standard deviations were short ranges. However, as the number of samples were larger, the standard deviation of the scratched program became bigger. This difference could be due to an additional mathematical technique of the library.

6 Conclusions

In conclusion, accuracy, precision and recall of the scratched program and the library which selects *uniform* as parameter *weights* were the same. However, there were two differences about the computing time, the average and the standard deviation. Both differences expanded as the number of samples increased. The reason of it can be considered that the library has functions which reduce the amount of computing time with mathematical techniques. The specific mathematical techniques in the library is outside the scope of this research. Further research is required to reveal these specific techniques.

An additional method of calculating labels by selecting *distance* in parameter *weights*. The optimized accuracy of the *distance* method was higher than the *uniform* one. Moreover, the precision of the *distance* method was higher as the number of parameter K in KNN increased. Therefore, results seems to be better when *distance* is selected.

References

[1] University of California, Irvine, "Uci machine learning repository - wine quality data set," 2019. Available: `https://archive.ics.uci.edu/ml/datasets/Wine+Quality`.

[2] H. K. H. Y. K. Daisuke, H. Ryuji, *Techniques of the data analysis to win in the Kaggle competition.* Gijyutu Hyouronsya, 2019.

[3] H. Yuzo, *Pattern recognition for beginners.* Morikita Syuppan corporation, 2012.

[4] scikit-learn, "sklearn.neighbors.kneighborsclassifier," 2019. Available: `https://scikit-learn.org/stable/modules/generated/sklearn.neighbors.KNeighborsClassifier.html#sklearn.neighbors.KNeighborsClassifier`.