Mid-term project report *

Xiangdong He

October 26, 2021

1 Dataset preprocessing

Currently, the missing feature values are regarded as a special attribute value, i.e., regarding the ? as an attribute value. The median of the numeric feature values were used to serve as a threshold and convert them to two categories, i.e., 'yes' and 'no'. 'yes' means the number exceed the threshold, i.e., the mean. 'no' means the opposite. Also, other bin size were tried for 6, and 10, where each bin has the same number of attribute values. Also, for convenience the labels, i.e., 0 were all replaced with -1. In this report, 1 denots positive case while -1 means the negative case.

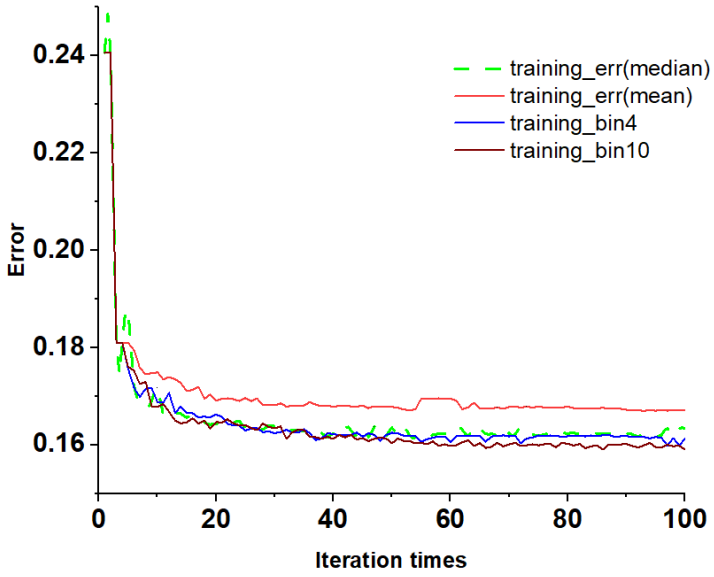

Figure 1: AdaBoost training error vs iteration times under different bin size

2 AdaBoost Algorithm

For Adaboost algorithm, increasing the bin numbers cannot improve the prediction accuracy. The training errors decrease significantly with iteration times, which can be observed in FIGURE 1. After the first 30 times of iteration, the error curve becomes flat. Also from FIGURE 1, in AdaBoost

*CS6530 Machine Learning; Fall 2021 Instructor: Shandian Zhe, University of Utah

Algorithm, the training errors versus iteration times show that the bin size has limited impact on the accuracy. Thus, here it is still enough accurate to use the median as the threshold, and the bin size was just taken as two.

3 Decision Tree Algorithm

The fully expanded decision tree algorithm was also considered to explore the training data and predict the errors of the training dataset. For the bin size being 2, the confusion matrix was also computed to construct the AUC-ROC curve for estimating the performance of the classifier model. The smallest training error of the decision tree is **16.6**% at the tree depth of 14. The training error along with the tree depth is shown in FIGURE 2. The AUC-ROC curve of the decision tree curve is shown as below. The AUC value, ie., the area enveloped by the ROC curve with horizontal axis, is **0.31**. It is found by increase the bin numbers for the numeric values of the feature, the training error can decrese significantly. At the tree depth being 14, the training error for the 6 and 10, bin size are **7.6**%, and **3.5**%, respectively. The errors are much smaller than those in AdaBoost. The bins for the numeric value columns at column 0, 2, 4, 10, 11, 12 were set at 10 in a way such that each bin contains the same amount of the values. From bin 1 to bin 10, the values in the bins increases.

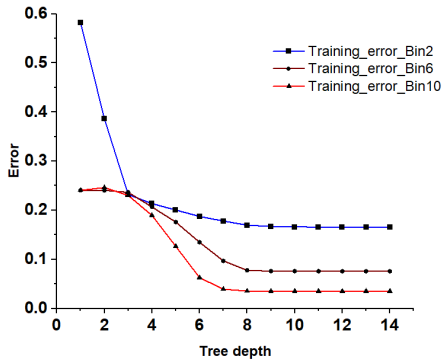

Figure 2: training error vs tree depth

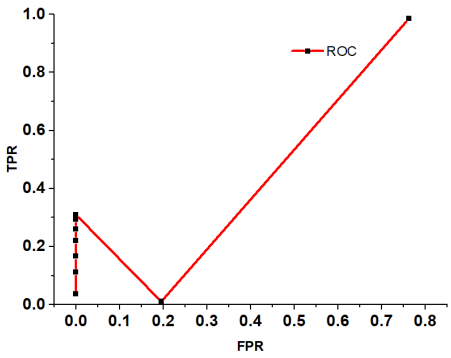

Figure 3: ROC curve on training data (bin size = 2)

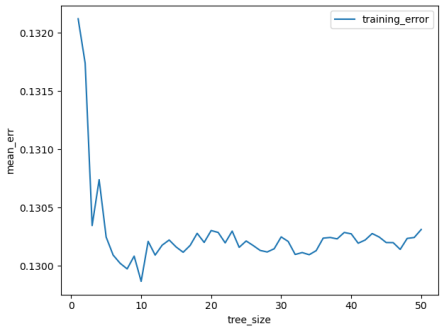

Figure 4: bagging training error vs iteration times

4 Bagging Tree Algorithm

The bagging algorithm also explored in this project as well, its training error along with iteration times is shown in FIGURE 4. The tree depth is set at 14 and the iteration times is set at 50 to lessen the computing burden. The tree depth can be set to 8 without the necessity to be 14, according to the training error plot of the decision tree in FIGURE 2. The training error of the bagging tree fluctuates around **13.0**%, which is still smaller than that of the AdaBoost algorithm. The input data is the 10 bins dataset, named *train_bin*10.*csv*. Sampling number is 25000, iteration times were 50 and the tree depth is 8.

5 Current conclusions

Consider the smaller training error in decision tree algorithm and in the bagging algorithm, the Kaggle submissions are only based on the two methods. The two predictions are shown in the csv files, named *bagging_tree_prediction.csv* and *decision_tree_prediction.csv*. The adaboost cannot achieve smaller training error, therefore, it was not used in this report.
Till now several conclusions as the followings can be drawn:
a) The bin size of the numeric feature values (i.e., bin the columns with numeric values (continuous feature values) into different bins of the same size) has limited impact on AdaBoost algorithm, but it can greatly reduce the training errors in the decision tree algorithms, where fully expaneded trees were used.
b) In the case of two bins, the threshold as median instead of as the average of that column will achieve better accuracy.
c) For the decision tree, its performance reached to its maximum when the tree depth is 8. This can be reflected in FIGURE 2, where the error curve went flat after the tree depth reached 8.

6 Plan for the rest of this semester

In the rest of this semester, more algorithms will be explored on this project. The general target is to find classifiers that not only have high running speed but also the accuracy and robustness can be ensured. The code need to be as bug-free as possible.
a) The accuracy of the current classifiers will be improved. The numeric values of the features will considered fully, instead of just regarding them as dicrete classes.
b) Another target is to improve the efficiency of the classifier, i.e., to accelerate the running speed of the models. Currently, running the fully-expanded decision tree takes time. I will try to find better way to optimize the current code so as to boost the speed.
c) Also, it may be better if the discrete attribute values can be converted to continuous ones, where the regression methods apply. By so, I can convert the problem to be an algebra one. Maybe, the classifier obtained in that way works better.
d) To check the current code, the machine learning library, like Sklearn, Tensorflow, etc. will also tried to be used to predict for the sake of evaluating the classifiers generated by my own code.
e) All the code in this project will be optimized and vectorizing computing is preferred in the code. Code debugging will be conducted to make the code more robust and efficient.
f) The missing feature values will be estimated instead of regarding them as a special attribute value.