MATH 540 Statistical Learning - Final Project

Nariman Kassymkhanov[1], **Otankhan Maikenov**[2,*]
Email: [1]nariman.kassymkhanov@nu.edu.kz, [2]otankhan.maikenov@nu.edu.kz

Abstract. In this project we considered a decision tree classifier on a binary classification problem. Firstly brief information about Decision trees. Implementation of the decision tree was imported from scikit-learn library. We train the algorithm on the data set generated by us and compare the results with theoretical values. Thereafter we evaluate our classifier on real data. Finally, we demonstrate our observations, which were obtained by experiments on the classifier.

Keywords: Decision tree, binary classification.

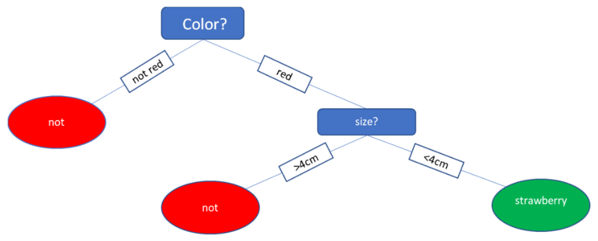

Figure 1: Simple example illustrating how Decision classifier works.

1 Introduction

In this project, we are going to consider a decision tree. Specifically, we will be working with training and true errors of the decision tree, and verify that true error is bounded. In general, the decision tree is a model which uses a tree-like structure to make a prediction. A node in the decision tree represents a test on the attribute of data. Simple illustration of how the decision tree works are shown in Figure 1. In the given example we are trying to find strawberries among other fruits. Particularly we will be dealing with classification tasks. Unlike a regression problem, in a classification one, the output variable is categorical and not numerical, i.e. classifier shows to which class the output variable belongs. In this project, we will be working with binary classification only, which means our output will be classified as one of the two classes.

2 Theoretical results

To show that true error is bounded, we need to introduce inequality. The inequality, which shows the relation between true error and training error and the fact that true error is bounded, is given as follows:

$$L_D(h) \leq L_S(h) + \sqrt{\frac{(n+1) * log_2(d+3) + log(2/\delta)}{2m}} \tag{1}$$

where $L_D(h)$ is a true error, $L_S(h)$ is a training error, n is number of nodes, d is a dimension of a feature vector, and m is a sample size. As you can see this inequality gives an upper bound for the true error of an algorithm. Intuitively, from looking at inequality, we can assume that by increasing the sample size, we decrease the training error and bound. And if we increase the dimension of the feature vector, the expression's value will increase. In addition, since there are a lot of features that need testing, the training error should also increase.

3 Empirical results

In order to verify that the theoretical upper bound holds, we run our algorithm several times, and each time we changed one of the two hyper-parameters: sample size (m) and the dimension of the feature vector (d). While changing the sample size we fixed the value of the feature vector's dimension, and vice versa while changing the dimension we had a fixed value for the sample size. Firstly we run this algorithm on the data-set which was generated by us. You can see the results in Figure 2 and Figure 3.

Thereafter, we evaluated this algorithm on real data set: A breast cancer data-set. As in the previous case, we evaluated the algorithm several times, and each time changed one

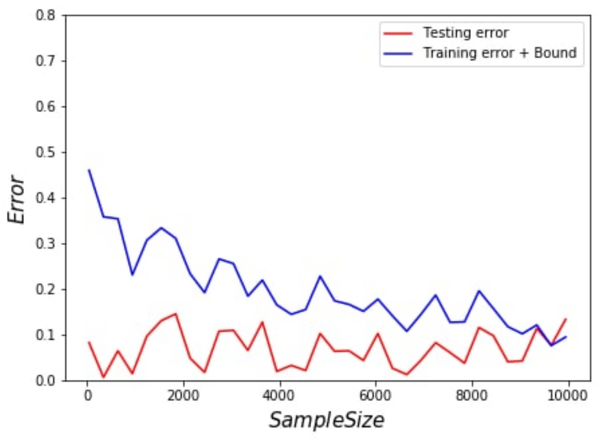

Figure 2: Generated data. Sample size vs. Error bound

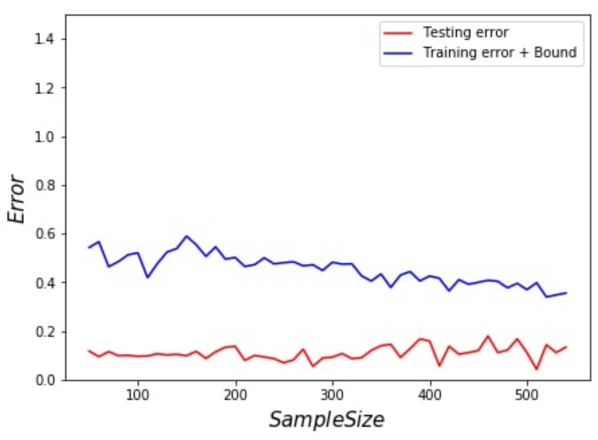

Figure 4: Breast cancer data. Sample size vs. Error bound

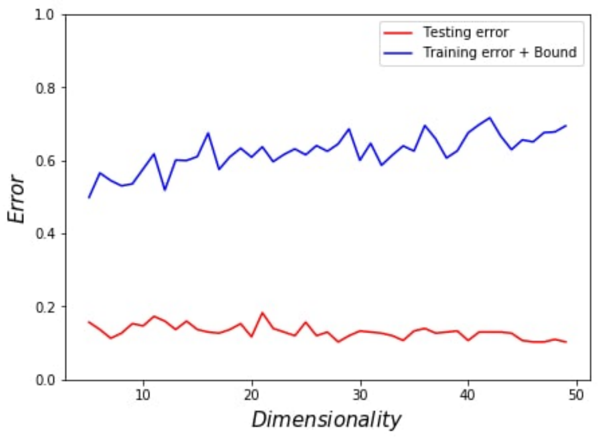

Figure 3: Generated data. Dimension of the Feature vector vs. Error bound

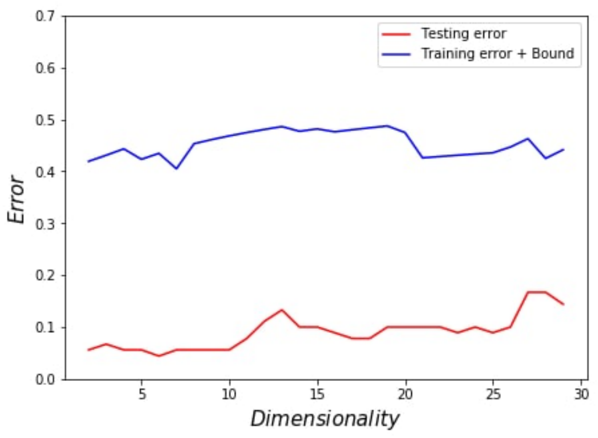

Figure 5: Breast cancer data. Dimension of the Feature vector vs. Error bound

of the above-mentioned two hyper-parameters while giving a fixed value for another one. The results are demonstrated in Figure 3 and Figure 4.

4 Discussion

Firstly, to mention, we generated data by using the "make-classification" function from the scikit-learn library. We used the default parameter of a number of classes, which is equal to 2 because our problem is binary classification problem. So in order to change the sample size or dimension of the feature vector, we assigned different values to this function. In addition, we fixed the number of max-depth with value 5 and fixed δ as 0.1, which means that our inequality holds with probability 90%.

As you can see, in all of the graphs the true error is bounded by a given expression. Let's discuss some of the nuances of each graph. From Figure 2, we can observe that the blue line is decreasing as the sample size increases. The intuitive reason for the decrease in training error is that sample size is inversely proportional to the bound. In addition, the term in the root decreases because of the value in the denominator. As it is seen from the graph, when the sample size exceeds 9000, the inequality above does not hold anymore, as it is the result of overfitting the data.

Since there are more features, i.e more information on each data, the prediction of the output vector is better, so true error will decrease. However, the term in the root increases because of d term is in the numerator. So the behavior of the blue line is increasing. This observation can be seen in Figure 3.

Similar results can be seen in Figures 4 and 5. However, Figure 4 and 5 describes real data on Breast cancer, sample size, and the feature vector given initially: sample size is 569 and d is 30. In addition, we can observe that the blue line is decreasing and the red line is increasing.

This is because our data has only a size of 569, so our data tends to be overfitted. However, the surprising behavior can be seen in Figure 5, particularly the testing error. This can be caused by the result of an increase in the dimension of the feature vector, while the sample size is fixed and relatively small.

5 Conclusion

The main purpose of this project was to prove that the true error of the binary decision tree classifier is bounded above. After empirical analysis, we can verify that true error is in fact bounded above. The report can be found in https://github.com/Moonwalker17/Final-Project

6 Contribution

Nariman Kassymkhanov has worked on generating data set and fitting the decision tree model for it, and Otankhan Maikenov has used the results of his work and has done the work on real data. On the report the work was done together, each of them describing their works on their own, and Otankhan wrote the Abstract and Introduction parts.