PROJECT 3 – ASSESS LEARNERS

Seyhan Emre Gorucu
sgorucu3@gatech.edu

***Abstract*—This report contains the results of my findings after implementing decision tree learner, random tree learner, bagging and insane learner. In this project, we find that using fewer number of leaves can cause overfitting. However, this can be alleviating by using a Bag Learner. Decision Tree performs slightly better than the Random Tree on the testing set for the case analyzed.**

1 INTRODUCTION

Classification and regression trees (CART) consist of a series of techniques to group and estimate labelled data. These algorithms are mostly used for supervised learning. In this study, we looked at decision tree learner, random tree learner, bootstrap aggregating (bagging) and insane learner. In this section, we briefly describe these algorithms while we will give the details in the Methods sections. Decision tree learners start from the top of the tree with a node and the node creates two branches based on a criterion. As this study only includes regression, the criterion is whether a number is less than a number or more. The dataset that contains values less than or equal to the set criterion go to the left branch and the remainder data go to right. Then, the node created in the left branch again branches into two. The same procedure goes all the way to the bottom. By creating splitting criterion based on the numerical value of one of the x parameters, we can group the similar data into leaves. Decision tree learners can use a smart decision criterion such as entropy or correlation coefficient. Random tree learners are the same as decision tree learners except the decision criterion is random i.e. the x variable that determines the separation is determined randomly at each node. Bagging creates multiple datasets created from the original dataset by randomly picking with replacement. Then, each dataset is used to train one or a combination of the aforementioned learners, or other learners. The estimation of each learner is averaged as final estimation. This method has the advantage of eliminating the bias of a single learner. Insane learner creates multiple bagging learners again by randomly selecting from the original dataset with replacement.

2 METHODS

2.1 Decision Tree Learners

The following pseudocode is implemented. This is a recursive algorithm. The required input are x, y and leaf_size.

decision_tree

If data size is 1, or equal to or less than the leaf size return [-1 value -1 -1]. Here, value is the average of the y values in the dataset.

If all the y values are equal, return [-1 value -1 -1] where value is the y value

If all the x values are same in the dataset, return [-1 value -1 -1] where value is the mean of the y values.

Check the correlation coefficient matrix between the x matrix and y vector.

Select the largest of the absolute of the correlation coefficient index.

In the x matrix, if that index values are all equal, then return [-1 value -1 -1] where value is the mean of the y value.

Calculate the median of the selected x value. Select the data that is equal to or less than the median. If all of the data set is selected, then use the mean instead.

left_tree = decision_tree (Here, you call the same function again)

Select the data larger than the Split value (either median or mean).

right_tree = decision_tree

root = [selected index, split value, 1, 1 + number of rows in left_tree]

return append(root, left_tree, right_tree)

Once, the decision tree is created, then the user query another set of x values and y values are returned as estimations.

2.2 Random Decision Tree

This is exactly the same as decision tree learner except the splitting criterion is randomly selected at each node rather than using correlation coefficient matrix.

2.3 Bag Learner (Bagging)

Create any number of learners. Use the average of all of the learners y predictions. The learner and the number of bags are user specified for each bag learner.

2.4 Inside Learner

Inside learners creates 20 bag learners each of which contains 20 linear regression learners.

3 DISCUSSION

3.1 Experiment 1

Figure 1 shows the training error and testing error as a function of leaf size for the Decision tree learner (DTLearner) for Istanbul.csv. We can see that testing error is lowest around 7-18 leaf size. Below 7 leaf size, we see overfitting.

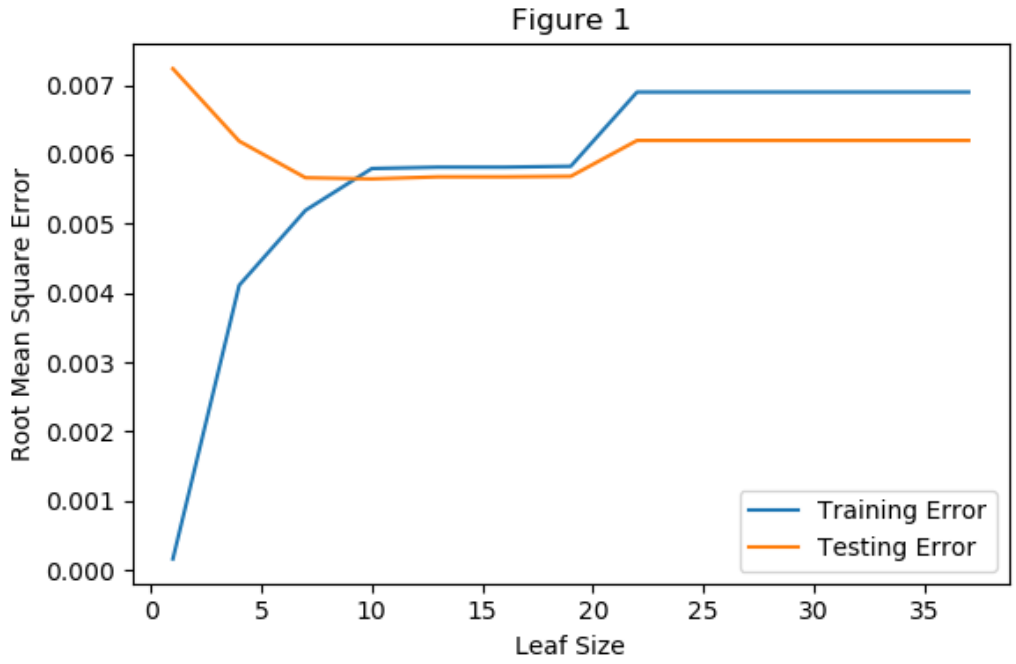

Figure 1 — Training error and testing error values as a function of leaf size for Decision Tree Learner.

3.2 Experiment 2

Increasing the number of bags reduces overfitting and eventually it can eliminate overfitting by cancelling the bias of each learner. Testing error in the figure below is pretty stable. This means, having more leaves is not an advantage but fewer leaves do not cause overfitting, either.

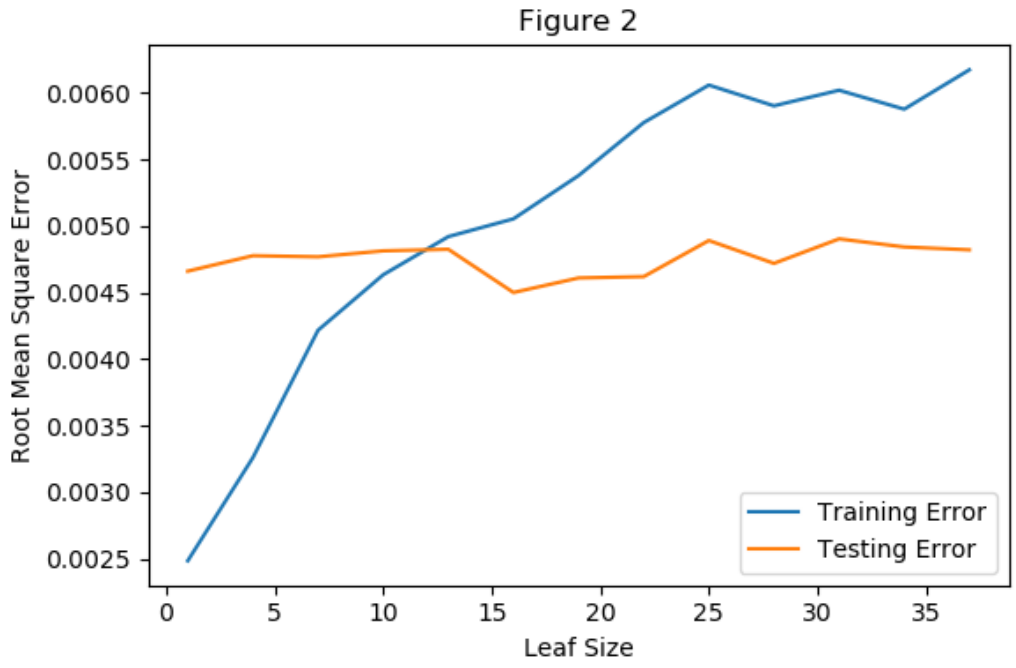

Figure 2 — Training error and testing error values as a function of leaf size for a Bag Learner with 20 bags using Decision Tree Learner.

3.3 Experiment 3

Figure 3 shows the R-squared value for Decision Tree and Random Tree by using Bag Learners with varying bags. Leaf size for all learners is 1. Similarly, Fig 4 compares the two learners by looking at the standard deviation. the compared data is the testing data of Istanbul.csv. For my test cases, decision tree seems to be performing better as std is lower and R-squared is closer to 1. However, as the bag numbers increase, the difference between the performance of Decision tree and Random tree narrows. As the decision criterion for Decision tree is more informed, I would expect the Decision Tree to perform better and make better

distinction on which x parameter is more decisive on the outcome. However, if the heuristics used in the Decision Tree is biased, it is possible to overfit. I think overfitting with Random Tree would be less of an issue and Random Tree performs well on large bag sizes. For my case, I haven't seen a significant difference while Decision Tree performs slightly better.

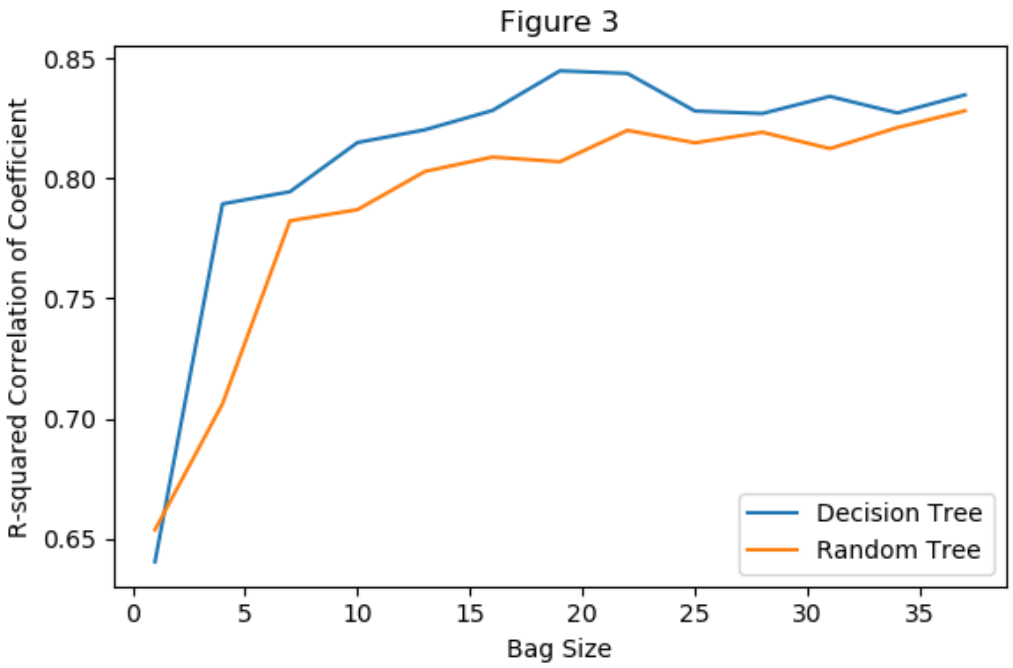

Figure 3 — R-Squared Correlation of Coefficient for Decision Tree vs Random Tree as a function of Bag size in a Bag Learner. Leaf size is 1.

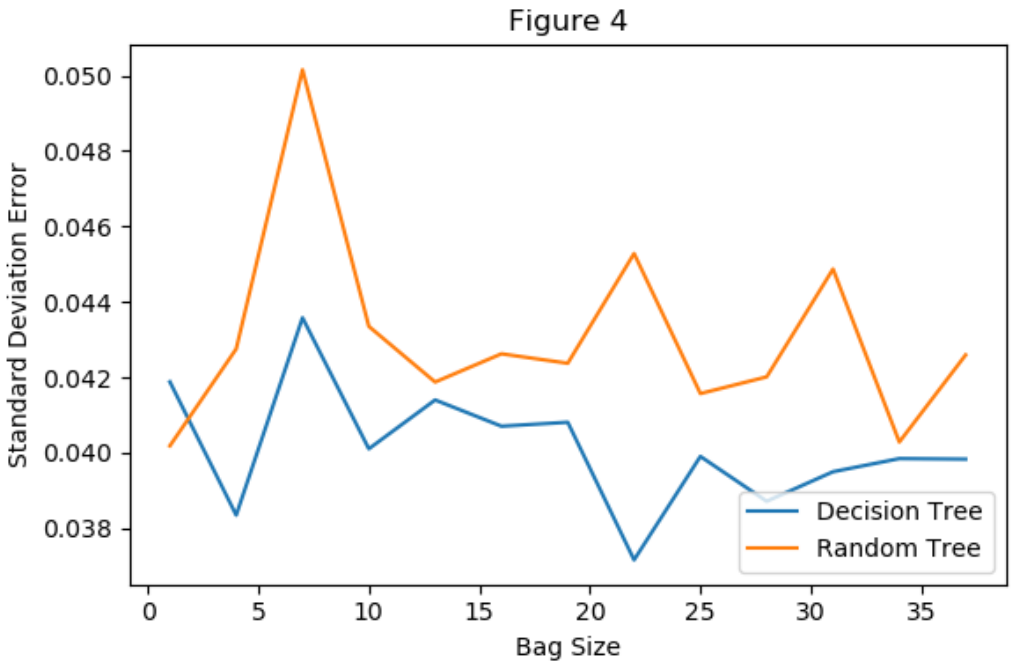

Figure 4 — Standard Deviation Error for Decision Tree vs Random Tree as a function of Bag size in a Bag Learner. Leaf size is 1.

4 SUMMARY

In this project, we analyze Decision Tree Learner, Random Tree Learner, Bag Learner and Insane Learner. We find that fewer number of leaves can cause overfitting for Decision Tree Learner. However, Bag learners alleviate this issue. Decision Tree Learner performs slightly better than the Random Tree Learner.