Project 3: Assess Learners

Josh Adams
jadams334@gatech.edu

Abstract**—**This project will compare two types of decision tree algorithms—showcasing the positive and negative benefits for each—regarding regression. The decision trees discussed are based on an algorithm described by JR Quinlan and the random decision tree is a variation of that same algorithm. This paper will examine these algorithms and demonstrate the superiority of the latter. This project will also discuss ensemble algorithms and their associated benefits of higher accuracy and overfitting reduction. Lastly, the inverse relationship that exists between leaf size and overfitting will be discussed.

1 INTRODUCTION

Decision trees are built progressively by splitting up data based on a split criterion. It starts at the root of the tree and uses its splitting criterion to split the data into two parts. These parts are then recursively sent back through the tree building algorithm. Once the data is sufficiently small or some set threshold such as leaf size has been met, it will create a leaf node. Typically, information gain or entropy is used for finding this splitting criterion. The splitting method used is based on absolute correlation of the data for the decision tree. For the random decision tree, the split criteria is randomly selected at each branch. The first hypothesis is that the leaf size is directly correlated to the levels of overfitting in decision trees. As the leaf size is decreased the levels of overfitting increase.

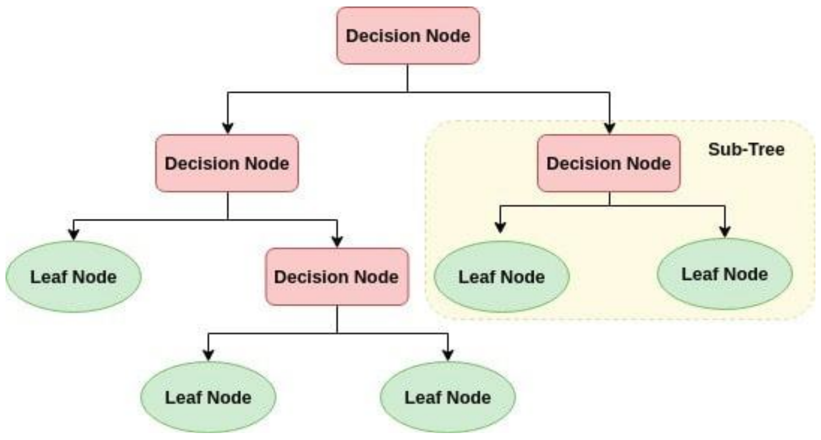

Figure 1 — An example of a decision tree. Source: DataCamp.

The next algorithm that is explored is an ensemble learning algorithm. This will take multiple instances of some arbitrary algorithm and train each of those algorithms on some subset of data. The ensemble method in this project is called Bootstrap Aggregating or Bagging for short. Since this project is based around regression the average of all predictions is what is returned. The next hypothesis is that the use of bagging will reduce the overall levels of overfitting while having higher accuracy than a single instance of the learner used in bagging. The final hypothesis is that the random decision tree will train faster and produce similar results as the decision tree.

2 METHODS

Each of the experiments are performed using the data which was provided in the 'Istanbul.csv' file. As well as each experiment has cross-validated results for every data point. The implementation of cross-validation used will take 10 random samples of the size of the training data set with replacement. This will essentially have each algorithm trained on different datasets. The testing set will be the same for all algorithms. The results for each of the various metrics for each of the cross-validation sets will be averaged to produce more accurate results. The bag learner will contain 20 different bags. Each bag will contain a decision tree learner and that learner will be trained on a randomly selected subset of the training set.

3 EXPERIMENTS

In the experiments, various algorithms are compared for accuracy as well as their ability to generalize. This ability to generalize can be quantified by the level of overfitting experienced on the data. Overfitting is the situation where an algorithm has been trained on a dataset and it learns to produce outputs specifically for that training set. The algorithm will perform poorly on new unseen examples due to overfitting. The ideal situation is for the algorithm to perform well on both the training set and testing set. Examples of overfitting will be shown in some of the experiments.

3.1 Experiment One

Experiment one will take the decision tree learner developed and train on the provided 'Istanbul.csv' file. It will compare the impact that leaf size has on the levels of overfitting, using root mean squared error as its metric.

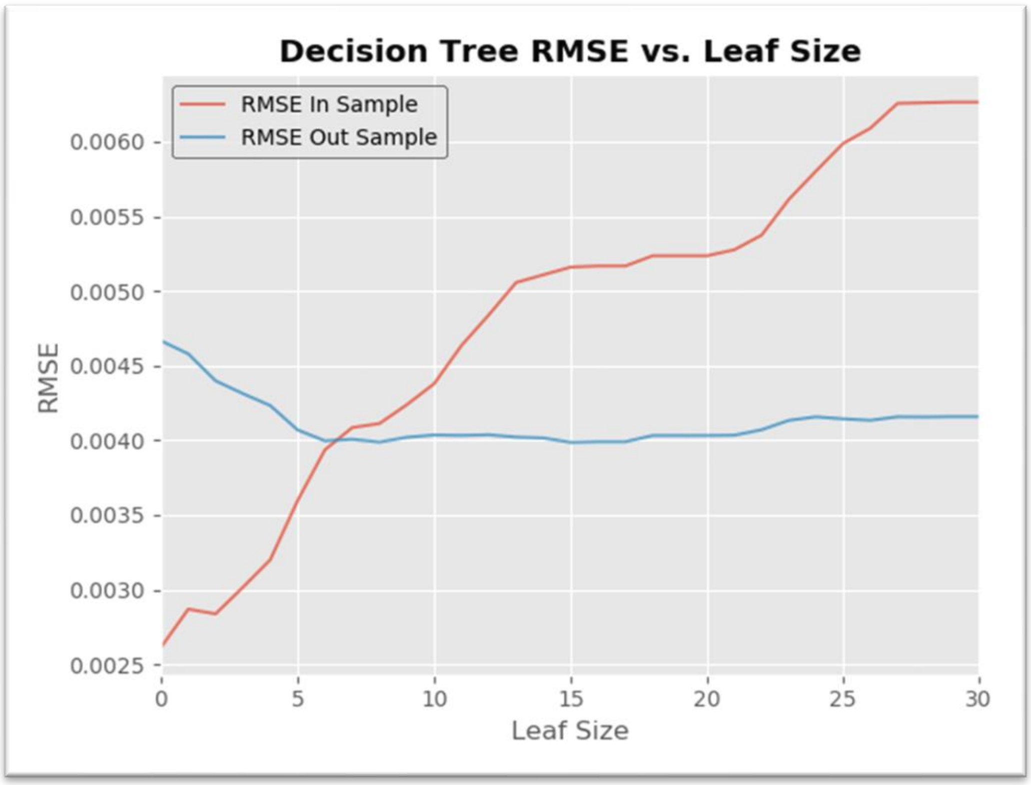

Figure 2—Decision tree root mean squared error impact as leaf size is increased.

Overfitting is characterized by the in-sample error decreasing and the out-sample error increasing. For the decision tree starting around a leaf size of 6, overfitting begins and quickly gains momentum. Looking at Figure 2 the direction of overfitting is left, which is associated with lowering the leaf size. This is expected because as the leaf size is lowered, more leaf nodes are created for the training observations. Thus, the tree can capture more of the variance within the data, so the in-sample error lowers. Conversely, it lowers its ability to generalize to new observations, which causes the out-sample error to increase.

3.2 Experiment Two

Experiment two will use the bagging algorithm and compare its performance as well as the levels of overfitting experienced by the algorithm. Root mean squared error will be the metric used for comparison and standardized to 20 bags per instance of bag learner.

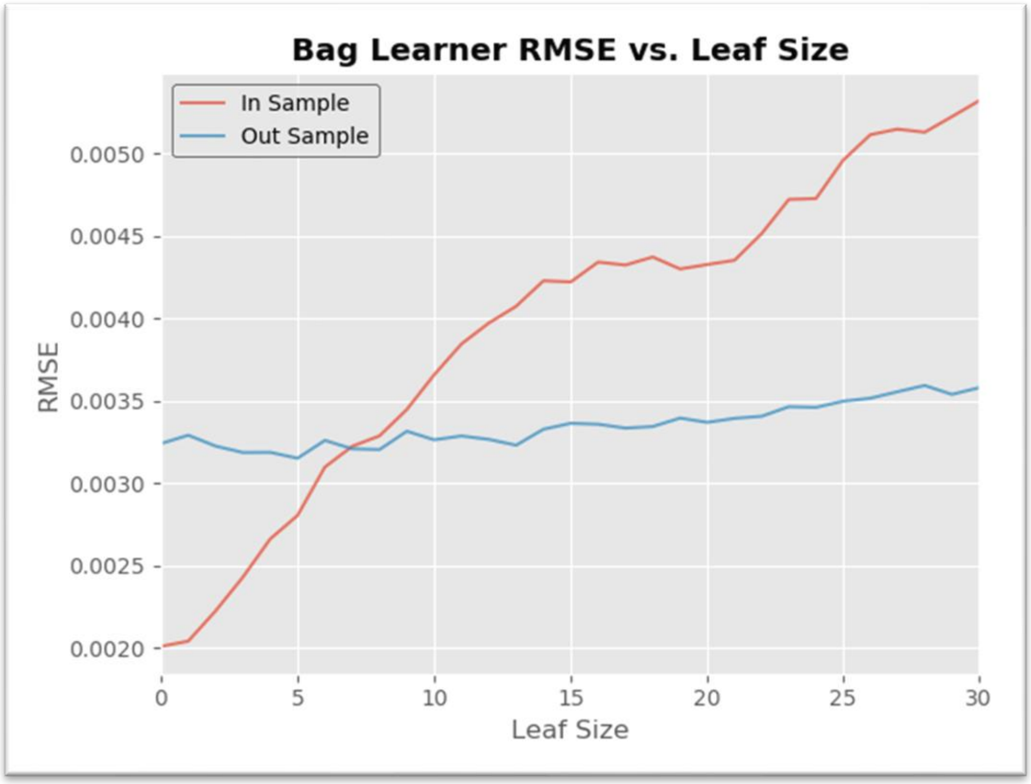

Figure 3—Bagging algorithms root mean squared error compared as leaf size is changed, default of 20 bags for each instance.

Figure 3 shows how the RMSE of the bag learner is lower at practically every leaf size in comparison to the decision tree learner in Figure 2. It appears that overfitting has been significantly reduced—almost eliminated. Starting around a leaf size of 6 or 7, the in-sample error is reducing, and the out-sample is only slightly increasing. The direction of the minor overfitting is leftward as the leaf size is reduced. The bag learner maintains its ability to generalize, shown with the only minor increase in out-sample error at lower leaf sizes.

3.3 Experiment Three

Experiment three will compare the decision tree learner with the random decision tree. Root mean squared error and correlation will not be used for comparison. Training time and variance will be used to compare the decision tree and the random decision tree.

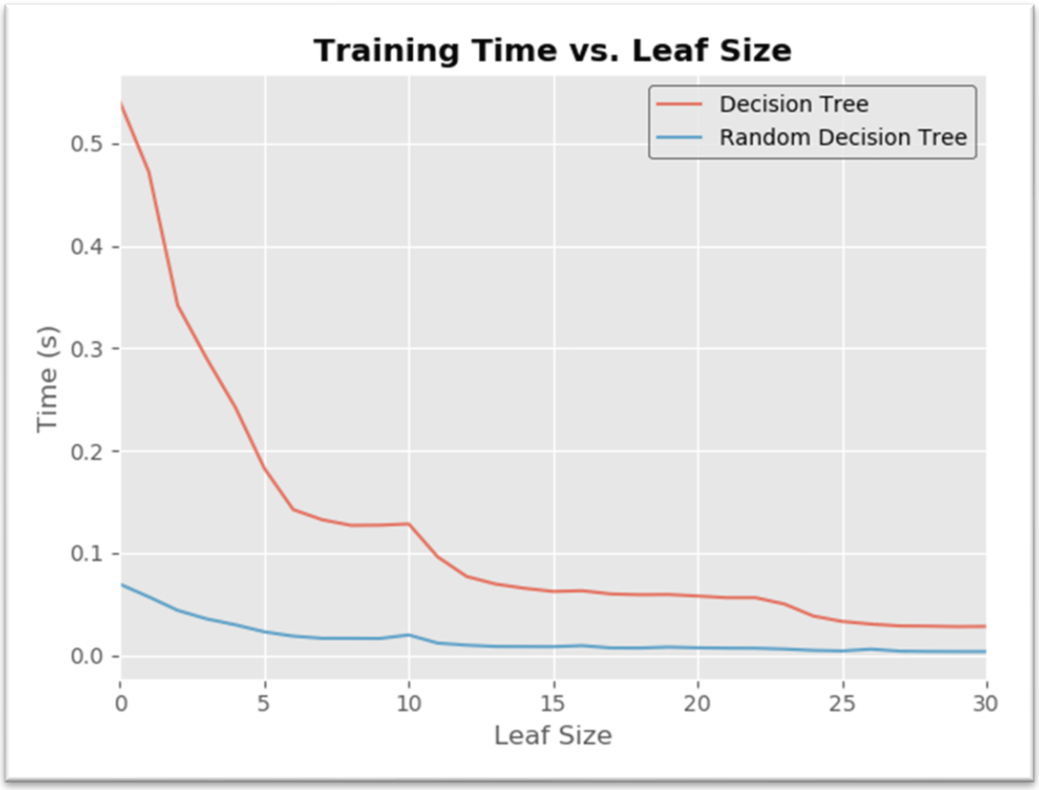

Figure 4—Decision tree vs Random decision tree training time compared to various leaf sizes.

When comparing the training times for both the decision tree and random decision tree, the random decision tree is the clear winner. The reason for the faster training time in the random decision tree is due to selecting a random feature to split on. Compared to the decision tree which compares some metric for each feature and picks the best one.

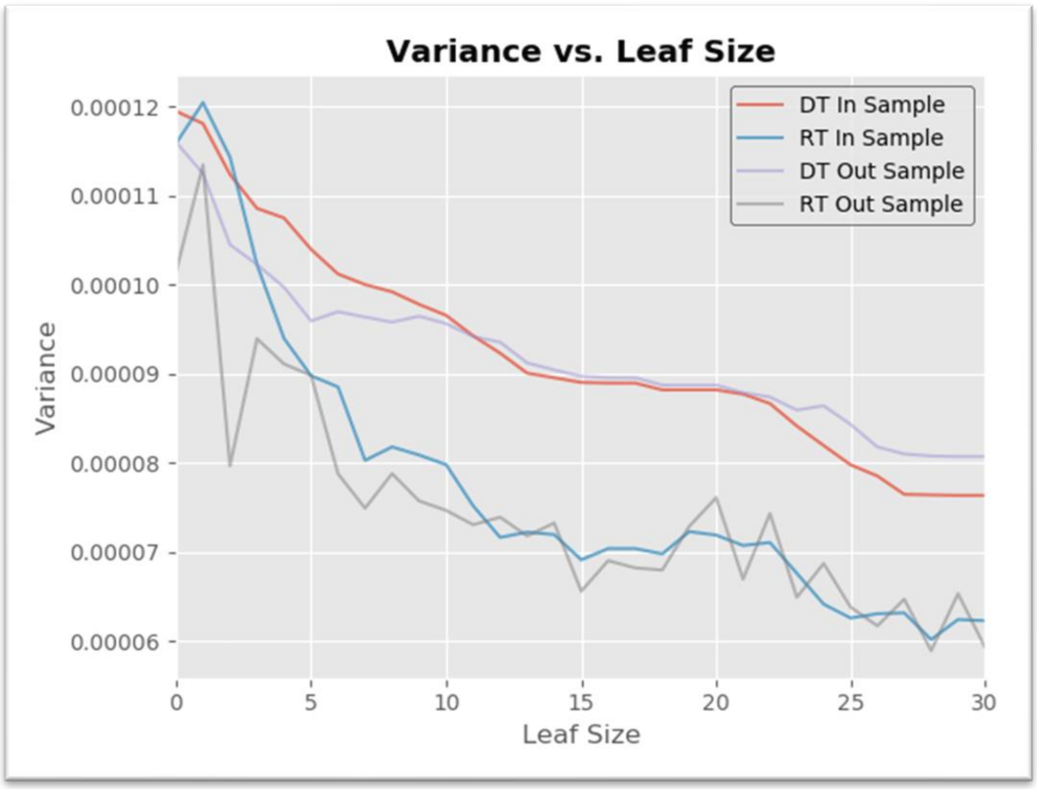

Figure 5 — Variance of in sample and out of sample data.

The variance of the random decision tree is also lower than the decision tree on both the in-sample and out-sample data sets. This suggests that the predictions for the random decision tree are closer to the actual y values than the decision trees predictions.

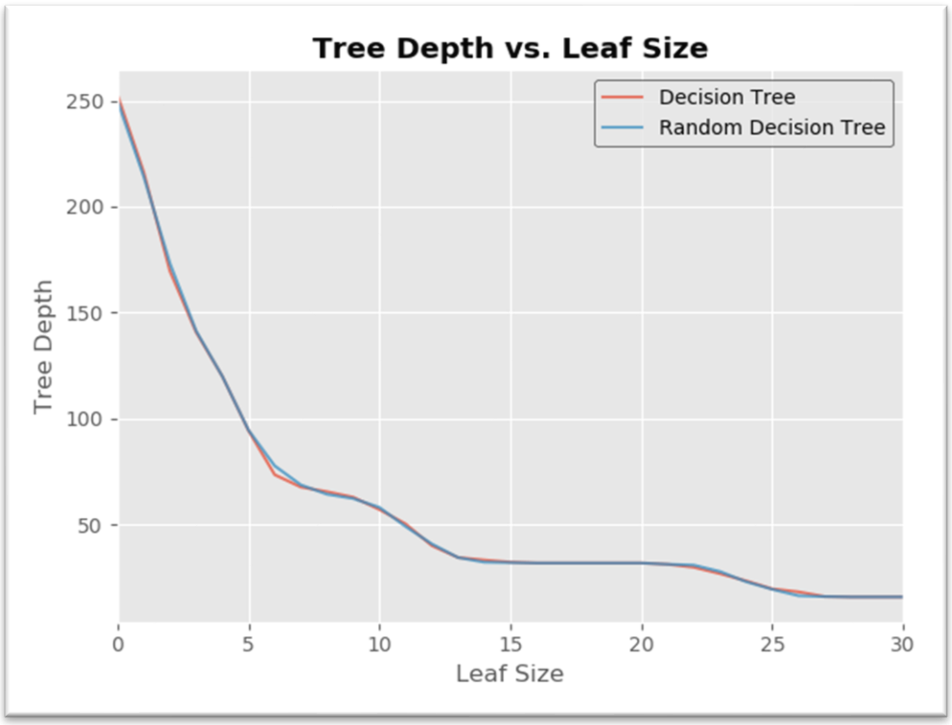

Figure 6—Decision tree vs Random decision average tree depth compared to various leaf sizes.

An interesting find was the average tree depth for the decision tree and the random decision tree. The random decision tree splits on a random feature; it is reasonable to expect the random tree to be much deeper due to the way it selects features. Both the decision tree and random tree ended up having about the same depth of tree regardless of the leaf size.

4 SUMMARY

As with all machine learning algorithms there are trade-offs which must be juggled accordingly. If accuracy is a priority then an ensemble learner is more than likely the route to go; conversely, if the priority is speed of the prediction then the random decision tree would be the solution. It is clear from the experiments that the initial hypotheses were correct. The bag learner produced the highest accuracy with the least amount of overfitting. The random decision tree was the fastest for predictions. As well as leaf size does correspond to the level of overfitting in the results.

5 REFERENCES

1. Joyner, David A. "Project 3: Assess Learners." *CS7646 Machine Learning for Trading*, Aug. 2020, lucylabs.gatech.edu/ml4t/fall2020/project-3/.
2. Balch, Tucker. [Ariel Bissett]. (2020, Sept). *CS 7646: Decision Trees Part 1 (new location)* [Video]. YouTube. https://www.youtube.com/watch?v=OBWL4oLT7Uc
3. Balch, Tucker. [Ariel Bissett]. (2020, Sept). *CS 7646: Decision Trees Part 1 (new location)* [Video]. YouTube. https://www.youtube.com/watch?v=WVc3cjvDHhw
4. CS 7646 – Machine Learning for Trading Piazza
5. Camp, Data. "Decision Tree Algorithm." DataCamp, https://www.datacamp.com/community/tutorials/decision-tree-classification-python. 9/8/2020.