CS7646: Project 3 – Assess Learners

Marcus Anderson
manderson332@gatech.edu

***Abstract*—**In this report, I present the findings received from implementing the four CART regression algorithms. These algorithms include: a "classic" Decision Tree, a Random Tree, a Bootstrap Aggregating (Bag) learner, and an Insane Learner. The goal of this project is to return a continuous numerical result as a regression learner, using techniques covered in course lectures.

1 INTRODUCTION

This report presents an inclusive evaluation of the four regression learners: DTLearner, RTLearner, BagLearner, and InsaneLearner. The goal of this project was to observe both the performance and efficiency of these algorithms, and display the results using a series of figures. This project consists of two main components: the implementation of the learners, using code, and the analysis of the experiments, using figures. The results of this project will deliver an in-depth view into the impact and effectiveness of different regression learners to readers. This will make it possible for readers to select the best algorithm for a specific issue. My hypothesis for this project is that the performance of the four learners will vary, especially because we are randomly selecting the training and testing data from the original dataset.

2 METHODS

The experiments in this project were created to give an insight to readers regarding the performances of the regression learners in certain conditions. In the first experiment, the focus was on exploring the relationship between leaf size and overfitting using the classic DTLearner. Using a for loop to run the DTLearner with various leaf sizes, ranging from 1 to 100, the in sample and out of sample root square mean errors (RSME) were calculated and recorded each run. These results were then plotted to determine if overfitting actually occurs in relation to leaf size.

The second experiment aimed to examine the impact that bagging has in regards to overfitting. Using the same for loop as the first experiment, the BagLearner and DTLearner were ran with leaf sizes ranging from 1 to 50, and a constant bag size of 20. The in sample and out of sample RSME results were then collected and plotted to see how much of an effect bagging has on overfitting, also, in relation to leaf size.

The third experiment compared the performance of the DTLearner and RTLearner algorithms. Three metrics were used to evaluate the performance: Mean Absolute Error (MAE), Mean Absolute Percentage Error (MAPE), and Maximum Error (ME). Two separate for loops were used, one for MAE and the other for ME, to calculate the in sample and out of sample results for both learners, respectively. For MAPE, the in sample and out of sample MAE results were

multiplied by 100 to obtain the percentages. The in sample and out of sample results for each learner were then plotted onto three separate graphs to show how the learners compare to one another.

3 EXPERIMENTS

3.1 Experiment 1

Overfitting is a common issue in machine learning where a model learns the noise in the training data to the extent that it negatively impacts the performance of the model on new data. According to the lectures, this occurs when the in sample error data decreases while the out of sample data error data increases.

In regards to this experiment, it appears that the results do not show a clear relationship between leaf size and overfitting as shown in Figure 1. In fact, from my result, it looks like the in sample data did not increase higher than the out of sample data for all 100 runs. However, I believe a reason for this is due to the training and testing data being selected form the original dataset at random, as opposed to in order.

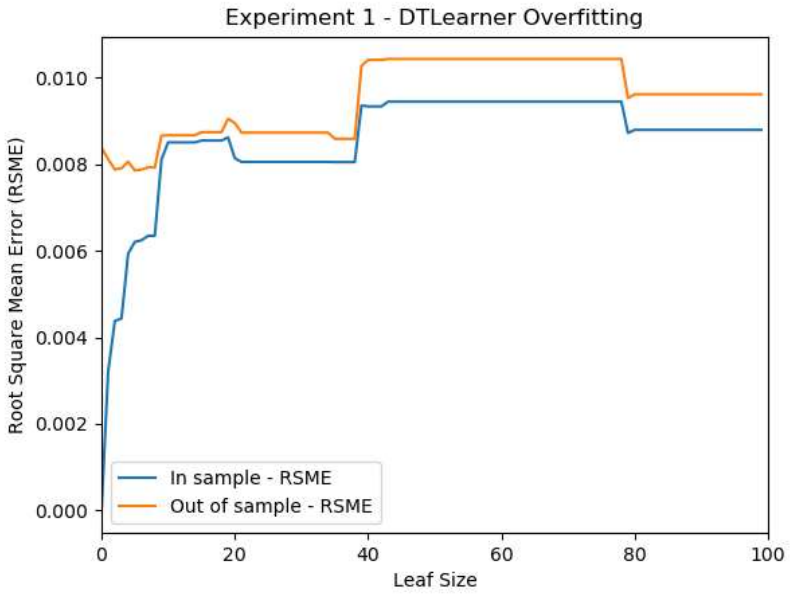

Figure 1—Generated graph from Experiment 1 results. This graph shows the Root Mean Squared Error (RSME) values in relation to leaf size using the DTLearner algorithm.

Originally, I accidentally ran this experiment without any randomization and observed that overfitting occurred when the leaf size was less than 10. Which meant that the in sample data results started to increase once the leaf size was greater than 10. This suggests that increasing the leaf size can help overfitting, however, this may or may not hold true for the randomized training and testing data used in the current experiment. I think it's fair to say that if I were to

conduct this experiment with a different random seed, the results may have been different depending on what training and testing data got selected from the dataset.

To conclude my findings, I believe overfitting can occur in relation to the leaf size, but the relationship may not be consistent across all trials.

3.2 Experiment 2

"Bagging, also known as Bootstrap aggregating, is an ensemble learning technique that helps to improve the performance and accuracy of machine learning algorithms. It is used to deal with bias-variance trade-offs and reduces the variance of a prediction model. Bagging avoids overfitting of data and is used for both regression and classification models, specifically for decision tree algorithms."(Biswal, 2022)

In regards to this experiment, I noticed the effect of bagging the DTLearner algorithm appears to help reduce overfitting, but not completely eliminate it. By referencing Figure 2, you can see that the gap between the in sample and out of sample results did not come close until leaf size reached around 45. You can also see that the results for both graphs appeared to be more jagged than smooth. This may be because we introduced bagging which collected 20 instances of the DTLearner over the course of various leaf sizes. Similar to experiment one, I also ran this without randomizing the training and testing data selections and saw that overfitting still occurs at an early leaf size. However, after the lines crossed, the in sample data continued to increase, while the out of sample data decreased, without coming back together. This tells me that bagging can help reduce overfitting by increasing the model's stability, but the relationship between leaf size and overfitting is not eliminated.

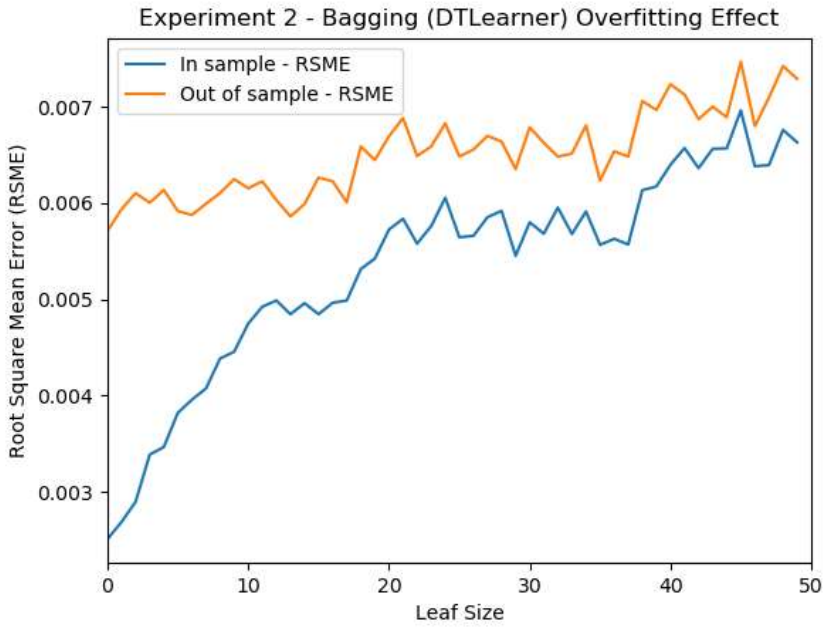

Figure 2—Generated graph from Experiment 2 results. This graph shows the Root Mean Squared Error (RSME) values in relation to leaf size with the use of the bagging and DTLearner algorithm.

In conclusion, bagging can be a great technique for reducing overfitting but it may not completely eliminate it. Again, this is based on the data received from my randomized seed, so the results may vary depending on how many times we run this experiment.

3.3 Experiment 3

The comparison between the "classic" decision tree (DTLearner) and random tree (RTLearner) algorithms in this experiment were performed using three different metrics: Mean Absolute Error (MAE), Mean Absolute Percentage Error (MAPE), and Maximum Error (ME).

Mean Absolute Error (|actual_i - predicted_i|/ actual_i) calculates the average difference between the calculated values and actual values, which is the accuracy of error in observations. Mean Absolute Percentage Error (MAE * 100) is just a variation of the MAE calculations, showing them in percentage format. Maximum Error (max(|actual_i - predicted_i|)) is the absolute value of the largest difference between a predicted variable and its actual value.

From the results of this experiment, it showcases that the DTLearner and RTLearner perform relatively similarly in terms of MAE, with both results ending around the same value once reaching a leaf size of 50, as shown in Figure 3. However, the RTLearner had overfitting occur with leaf sizes under 35 and over 40, while the DTLearner showcased overfitting during all runs, similar to experiment one and two. I also calculated and plotted the bonus metric of MAPE, shown in Figure 4, as an extra visual to show the MAE results in percentage format.

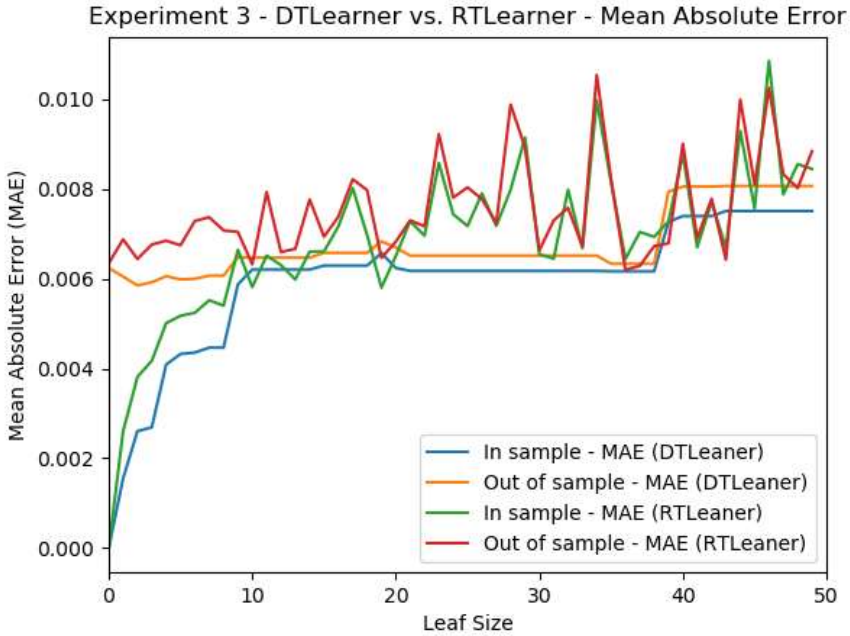

Figure 3—Generated graph from Experiment 3 results. This graph shows the Mean Absolute Error (MAE) values in relation to leaf size with both the DTLeaner and RTLearner algorithms.

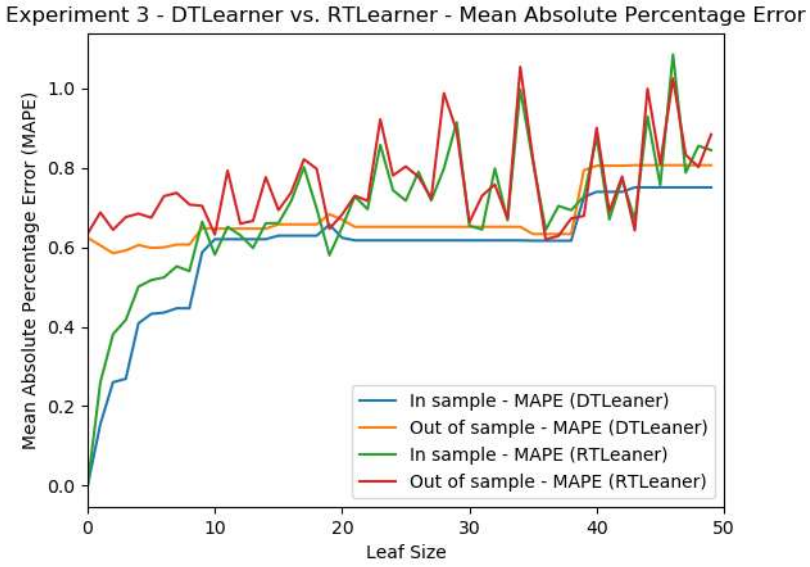

Figure 4—Generated graph from Experiment 3 results. This graph shows the Mean Absolute Percentage Error (MAPE) values in relation to leaf size with both the DTLeaner and RTLearner algorithms.

When it comes to the calculations of ME, there was a noticeable pattern with the out of sample data from the DTLearner staying at a constant maximum error value around .03 after leaf size reaches 10. On the other hand the out of sample data for RTLearner had a lone spike, reaching above .06 when getting to a leaf size of 20, as shown in Figure 5. We can also see that the in sample and out of sample lines for both leaners show various forms of overfitting through their respective runs, however, the RTLearner shows it more often.

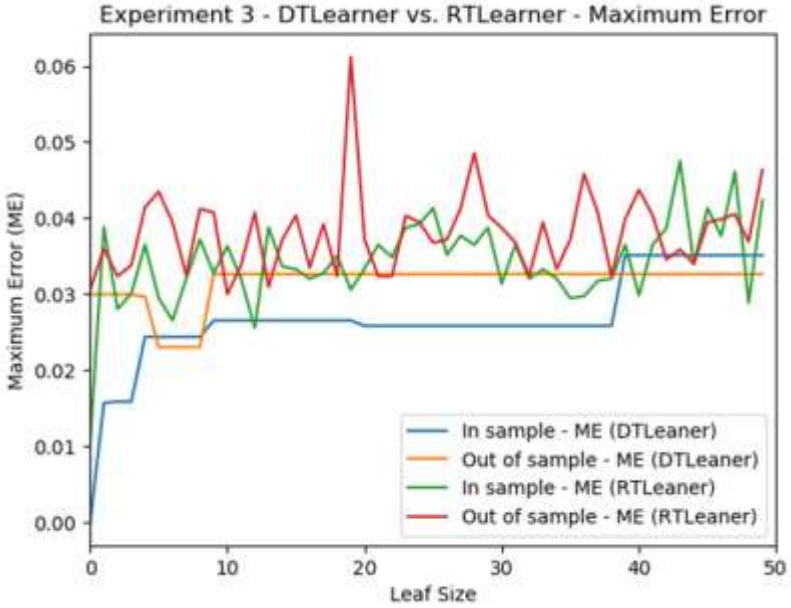

Figure 5—Generated graph from Experiment 3 results. This graph shows the Maximum Error (ME) values in relation to leaf size with both the DTLeaner and RTLearner algorithms.

Both DTLearner and RTLearner showcased their strengths and weaknesses during this experiment. To say that one leaner is always going to be more superior than the other is not really true since it's highly dependent on the specific requirements of a given problem. Based on this experiment, DTLearner tends to overfit more, but has a smoother performance, while RTLearner has a random element that can lead to a more jagged performance but can avoid overfitting in certain cases. If I had to give a performance score based on the chosen metrics, I would give the advantage to the DTLearner, as the results from this leaner appeared more smooth and constant for a lot of leaf sizes. This is especially shown in the maximum error calculations when the in sample data and out of sample data become more constant after reaching a leaf size of 10.

Similar to the comparisons of the DTLearner and RTLearner, the metrics I chose both have their respective strengths and weaknesses. The MAE and MAPE calculations gave us a more complete overview of which leaf sizes cause a higher critical error. This is helpful to see how this leaners perform overall rather than at specific leaf sizes, whereas ME calculations shows us notable outliers to keep an eye out for, like the out of sample data when leaf size reaches 20 in the RTLearner.

4 SUMMARY

In summary, the results of the three experiments shown in this project indicate that overfitting can be a common issue in machine learning depending on how the training and testing data get selected from the original dataset. Experiment 1 showed that increasing the leaf size had little to no impact on overfitting, even though it almost occurred at some sizes. The second experiment demonstrated that bagging can help reduce overfitting but not eliminate it entirely. In the final experiment, we compared the DTLearner and RTLearner algorithms and found that they performed similarly in terms of Mean Absolute Error, but RTLearner had more instances of overfitting. The Maximum Error showed us that the DTLearner produced more constant results in terms of error values, while the RTLearner had more jagged results throughout the entire experiment.

All in all, I believe my hypothesis was proven to be true. The results from each leaner varied for all experiments, which might have been a result from random training and testing data selection. This is why it's important to continuously run multiple trials to get a more accurate picture of how these learners behave overall.

5 REFERENCES

1. Biswal, A. (2022, November 30). Bagging in Machine Learning. Simplilearn. https://www.simplilearn.com/tutorials/machine-learning-tutorial/bagging-in-machine-learning
2. Brownlee, J. (2016, March 21). Overfitting and Underfitting With Machine Learning Algorithms. Machine Learning Mastery. https://machinelearningmastery.com/overfitting-and-underfitting-with-machine-learning-algorithms/
3. GeeksforGeeks. (2021, November, 28). How to calculate Mean Absolute Error in Python? GeeksforGeeks. https://www.geeksforgeeks.org/how-to-calculate-mean-absolute-error-in-python/
4. Regression Algorithms: Which Machine Learning Metrics? (n.d.). MyDataModels. https://www.mydatamodels.com/learn/guide/a-path-to-discover-ai/regression-algorithms-which-machine-learning-metrics/