Pranshav Thakkar

pthakkar7

9/24/2018

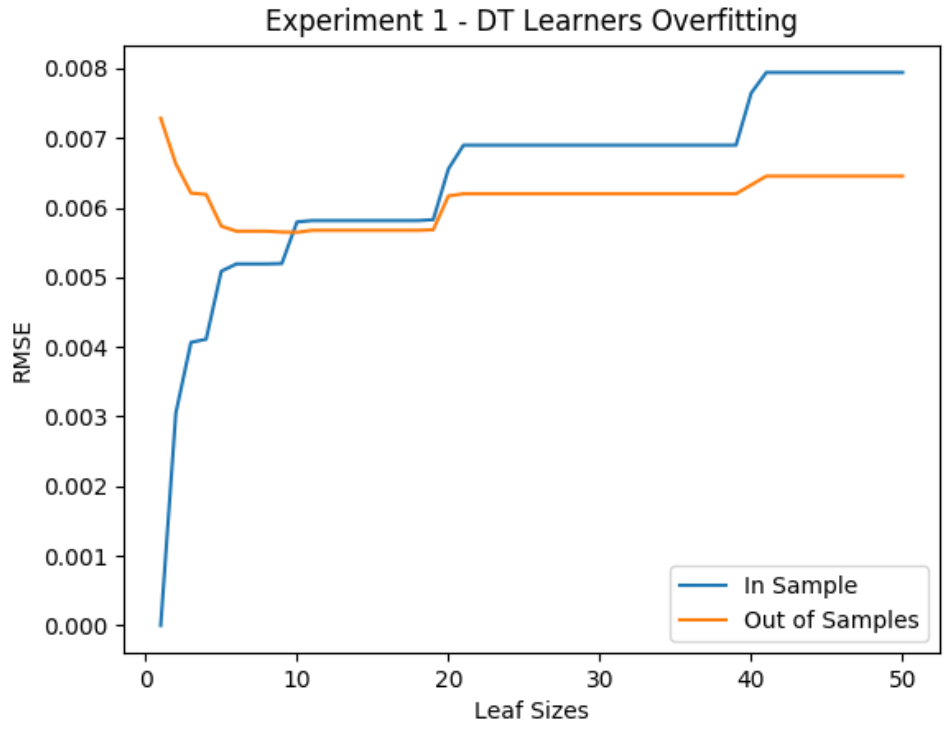
Assess Learners Report

Experiment 1:

In this experiment, we are testing whether overfitting occurs with respect to the leaf_size of the Decision Tree Learner. I chose to vary the leaf_size from 1 to 50 and record the RMSE for each leaf_size, for both in sample results and for out of sample results. I did this by running a loop over 50 indices and having the index be the number of leaf nodes. I collected the rmse data and stored them in lists for both in sample and out of sample. I used the Istanbul.csv file for my input data.

In this experiment, I noticed that there was some overfitting with regards to leaf size. In particular, I saw that the region from leaf size = 1 to leaf size = 7 had some overfitting. If we look in the direction of decreasing leaf size, we see that the in sample errors are decreasing while the out of sample errors are increasing. As mentioned by Professor Balch, this is a sign of overfitting. While this may be an anticipated result, as learners usually start off with a low in sample error and a high out of sample error before they generalize, I felt that this was worth mentioning.

I also noticed that the error rates for in sample results steadily increased at certain points as the leaf size increased. The out of sample error rate stabilized in comparison, and did not increase at as high a rate. The in sample error rate actual overtook the out of sample error rate at leaf size = 10. This shows that the learner was having more difficulty with the training data after a certain leaf size and stopped overfitting, and might have actually started underfitting.

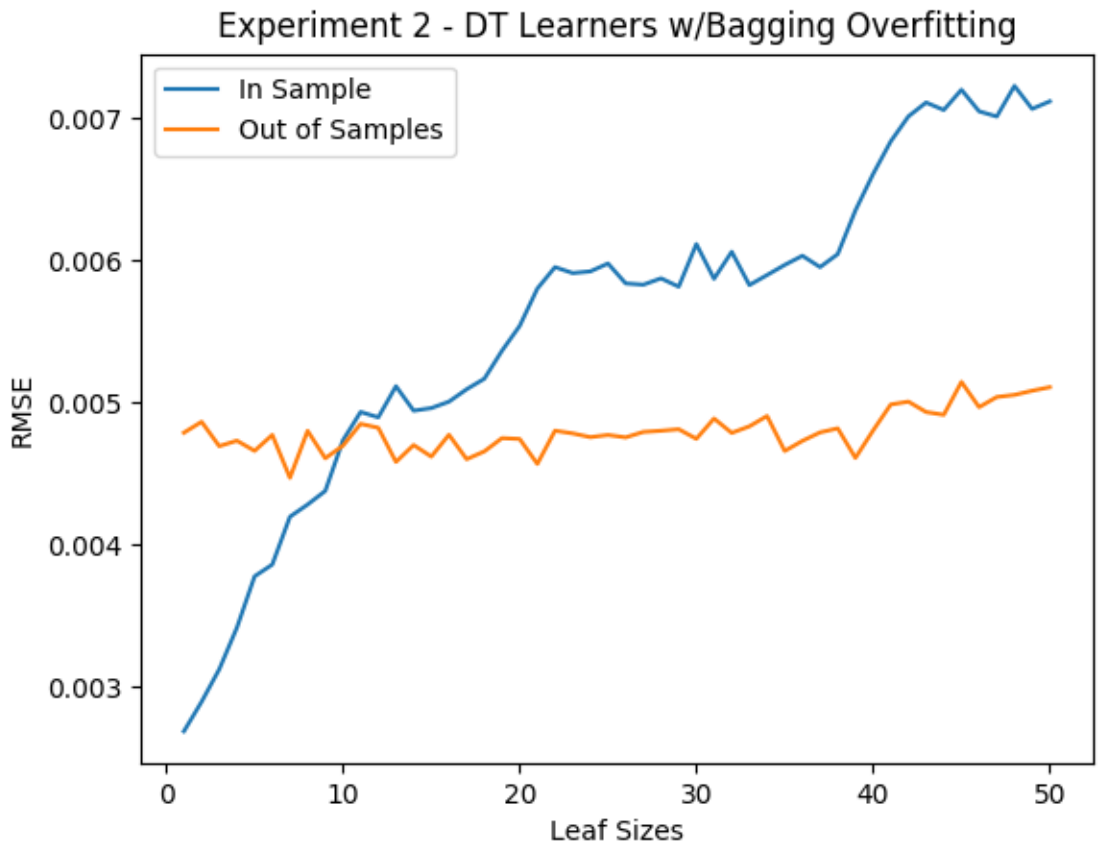


Experiment 2:

In this experiment, we are testing whether Bootstrap Aggregating, or bagging, has an effect on overfitting with respect to leaf_sizes of a DTLearner as compared to without bagging. In particular, we want to see if bagging can reduce or eliminate overfitting. So for this experiment, similar to the previous one, I changed the leaf size for the DTLearner from 1 to 50. The BagLearner that encompassed it had a fixed number of 20 bags. Again, I recorded the RMSE for both the in sample and the out of sample results and I used the Istanbul.csv file for my inputs.

In this experiment, I noticed the same trend where the in sample error increases with the increase of leaf sizes, and it crossed the out of sample error rate at leaf size = 10 again. This shows that bagging did not significantly change the overall trend of how the learner performs, which is to be expected. However, the error rates for the in sample results are marginally lower, and the rates for out of sample results are lower by a fair amount.

With respect to overfitting, we see that the out of sample error rates are pretty much stable along the plane with a few ups and downs, but no major area where it is increasing significantly while the in sample errors are decreasing. So, I believe that there is no overfitting that occurs in this experiment, and that bagging has taken care of the small amount that we saw in Experiment 1.



Experiment 3:

In this experiment we are trying to compare classic decision trees (DTLearner) against random trees (RTLearner) and see which ways the methods are better or worse than the other. One measure that I decided to test was the running time of the learners, from initializing, throughout adding evidence and predicting results. I tested the running time of each algorithm while increasing the size of the leaf nodes from 1 to 50. I used the Istanbul.csv file for my input data.

I saw that the random tree was pretty much faster than the classic tree all across the board. This makes sense, as it takes time for the classic tree to evaluate the function by which it splits the tree on, whereas the random tree just picks a random factor to split on. The random tree was much faster when the leaf sizes were smaller, especially at leaf size = 1, but the difference became smaller as the size of the leaves grew bigger. This was also fairly easy to anticipate, as when the leaf sizes are smaller the tree has to make more splits and so the classic tree loses time there.

This result makes it interesting to consider how the two perform accuracy-wise. I would believe that the classic decision tree would perform better in terms of accuracy as it actually finds the best factor to split the tree on using some metric, and thus can create a tree that is more adapted towards the data, versus a random tree. This is where the time cost vs performance cost discussion comes into play and we have to make a decision on whether we value time or performance more. It also helps if we know something about the data that makes it easier for us to decide whether we need a classic tree or we can opt for the random tree to try and save on time.