Assess Learner Project Report

Q1: Does overfitting occur with respect to leaf_size? Consider the dataset istanbul.csv with DTLearner. For which values of leaf_size does overfitting occur? Use RMSE as your metric for assessing overfitting. Support your assertion with graphs/charts. (Don't use bagging).

In order to observe overfitting, I generate a graph (Figure 1) showing us the value of RMSE corresponding to leaf_size from 1 to 20 for the dataset istanbul.csv using DTLearner. The blue line stands for out of sample results while the red line for in sample result. In this graph we find that with the increasing of leaf_size, which means the decreasing in degree of freedom, the in sample RMSE (red line) increases. And if we look at the blue line from large leaf_size to small ones (the direction of increasing in degree of freedom), we find that the out of sample RMSE first decreases then increases. These correspond to the theoretical meaning for overfitting in machine learning. So overfitting does occur with respect to leaf_size according to my experimental result.

While looking at the blue line from large leaf_size to small ones (the direction of increasing in degree of freedom), we find that the out of sample RMSE first decreases then increases. And at leaf_size = 5 the out of sample RMSE becomes the least. After that point, with the increasing in degree of freedom, the out of sample RMSE becomes larger while decreasing leaf_size (leaf_size = 4,3⋯). So overfitting occurs on leaf_size = 5.

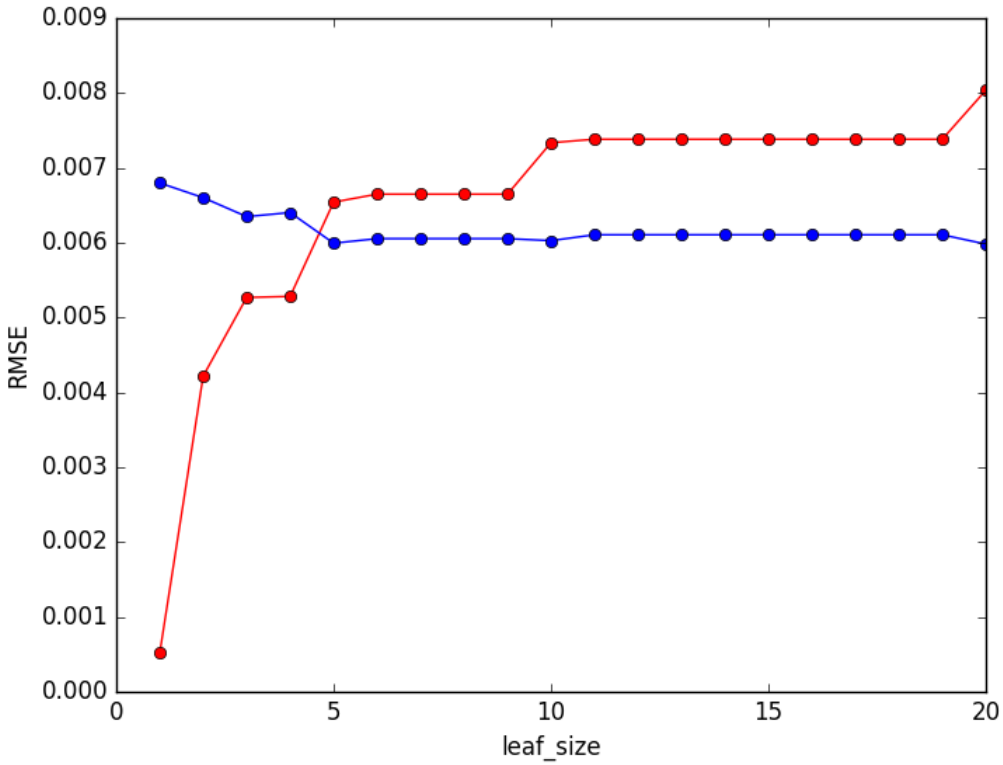

Figure 1: The in sample and out of sample RMSE of DTLearner

Q2: Can bagging reduce or eliminate overfitting with respect to leaf_size? Again consider the dataset istanbul.csv with DTLearner. To investigate this choose a fixed number of bags to use and vary leaf_size to evaluate. Provide charts and/or tables to validate your conclusions.

In order to observe if bagging reduces or eliminates overfitting, I generate a bag learner with a fixed number of bags = 20 for the dataset istanbul.csv using BagLearner. Similar as what I have done in Q1, I get a graph (Figure 2) showing us the value of RMSE corresponding to leaf_size from 1 to 20. The purple line stands for out of sample results while the green line for in sample result. In this graph we find that with the increasing of leaf_size, which means the decreasing in degree of freedom, the in sample RMSE (green line) generally increases on the trend. And if we look at the purple line from large leaf_size to small ones (the direction of increasing in degree of freedom), we find that the out of sample RMSE first decreases then increases. These give us the result that overfitting still exists.

But looking at the purple line, at leaf_size = 2 the out of sample RMSE becomes the least. After that point, with the increasing in degree of freedom, the out of sample RMSE becomes larger while decreasing leaf_size (leaf_size = 1). So overfitting occurs on leaf_size = 2. Comparing this to the result we get on Q1, we find that on BagLearner overfitting occurs with a larger degree of freedom (smaller leaf_size) than DTLearner. So bagging does reduce overfitting to some degree.

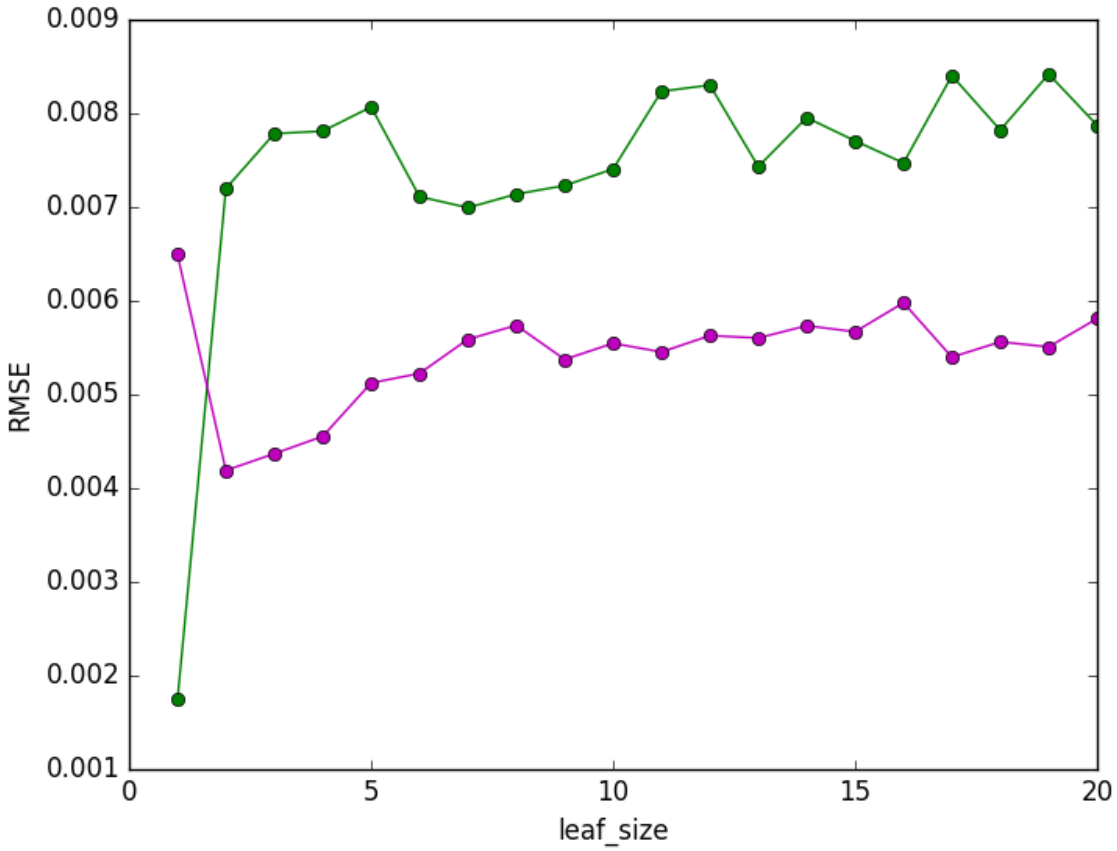

Figure 2: The in sample and out of sample RMSE of BagLearner

There is another graph (Figure 3) which can help us see results more clear. This graph shows us the out of sample RMSE corresponding to leaf_size from 1 to 20 for the dataset istanbul.csv using both DTLearner and BagLearner (same experiment as Figure 2). The purple line represents the out of sample RMSE for BagLearner while the blue line for DTLearner. In this graph we can explicitly observe that overfitting occurs in leaf_size = 5 for DTLearner and leaf_size = 2 for BagLearner. Moreover, we find that the out of sample RMSE of BagLearner is always less than this of DTLearner. So bagging does improve the total quality of training model as well as reducing overfitting.

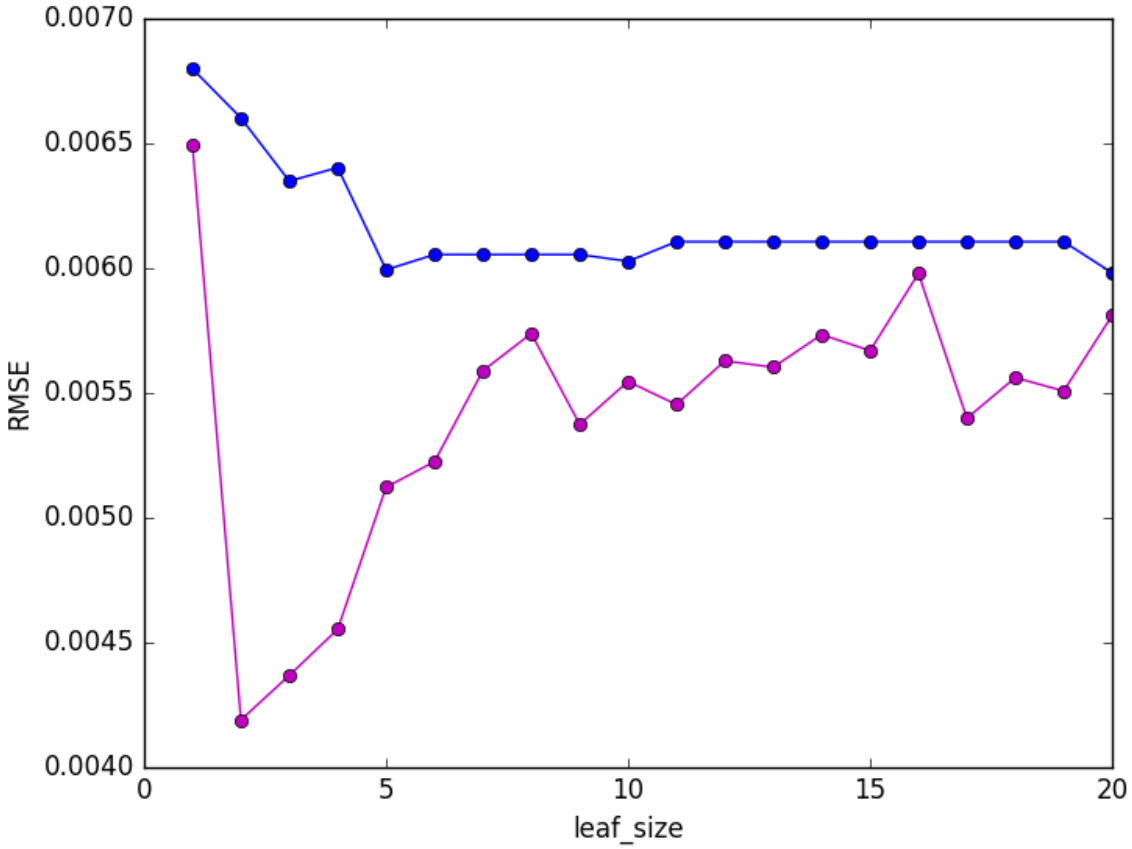

Figure 3: The out of sample RMSE of DTLearner and BagLearner

Q3: Quantitatively compare "classic" decision trees (DTLearner) versus random trees (RTLearner). In which ways is one method better than the other?

In order to quantitatively compare DTLearner and RTLearner, I choose out of sample RMSE, out of sample correlation of coefficients of Y and predicted Y and the running time (unit: s) for building trees as metrics. I generate three graphs (Figure 4,5,6) showing us the value of those metrics corresponding to leaf_size from 1 to 20 for the dataset istanbul.csv using both DTLearner and RTLearner. In all three graphs, the blue line stands for RTLearner while the red line for DTLearner.

From Figure 4, we easily find that the out of sample RMSE of DTLearner is generally less than the out of sample RMSE of RTLeaner, which means DTLearner is more accurate than RTLearner. Also, the red line is more stable than blue line. So DTLearner has more stable predicting accuracy on different leaf_size comparing to RTLearner.

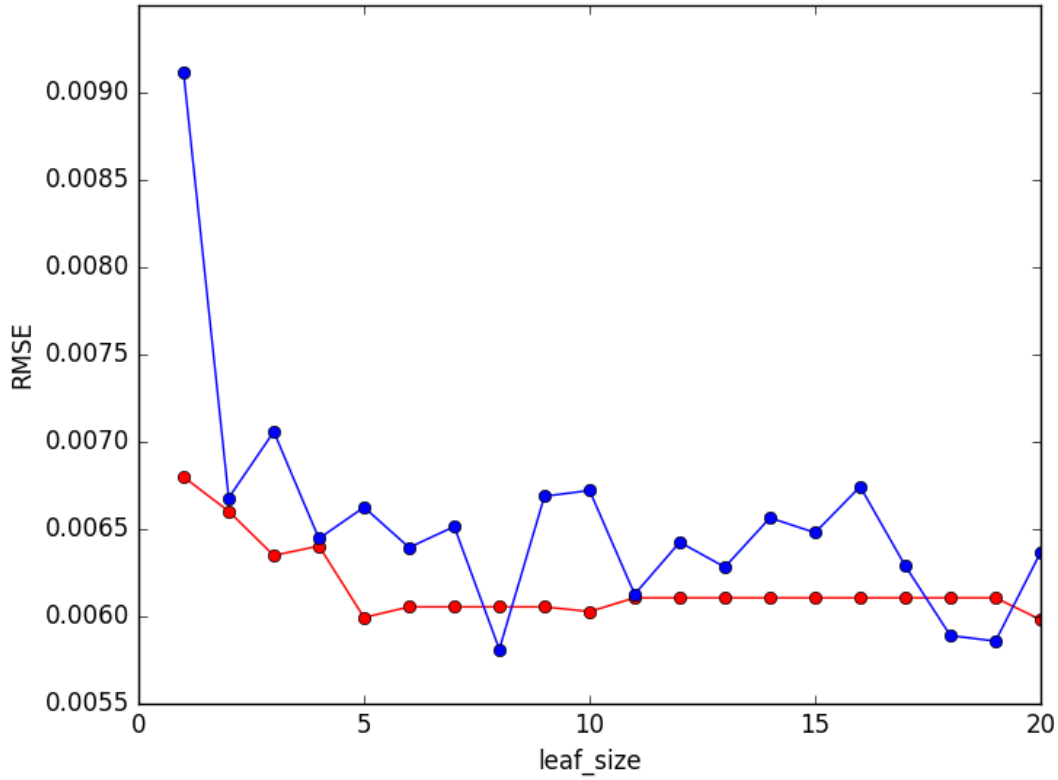

Figure 4: The out of sample RMSE of DTLearner and RTLearner

From Figure 5, we find that the out of sample correlation of coefficients of Y and predicted Y of DTLearner is relatively larger than this of RTLearner (this result may not be as clear as RMSE since there are too much unstable points in RTLearner), which means DTLearner is more accurate than RTLearner. Also, the red line is more stable than blue line. So DTLearner has more stable predicting accuracy on different leaf_size comparing to RTLearner. These results are similar as what we get from figure 4.

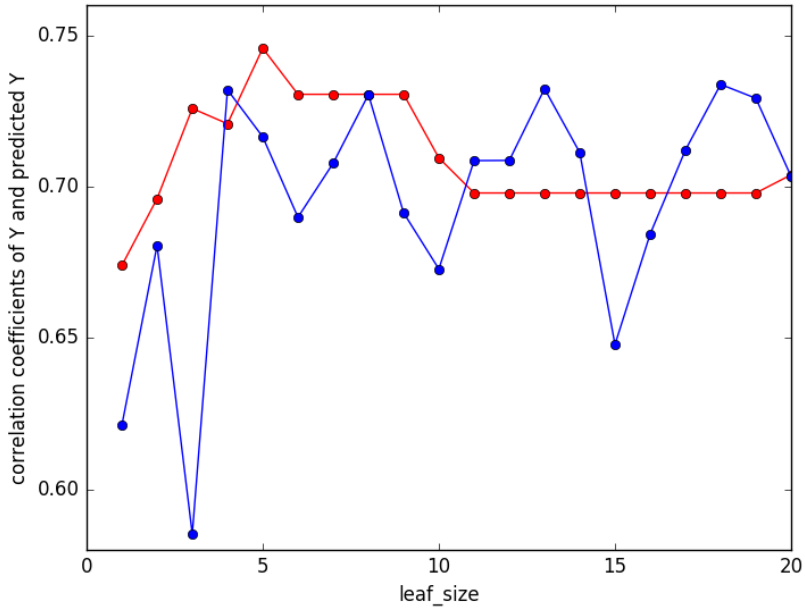

Figure 5: The out of sample correlation coefficients of Y and predicted Y of DTLearner and RTLearner

From Figure 6, we find that the tree building time (unit: s) of DTLearner is always larger than this of RTLearner, which means DTLearner is more time consuming on model building than RTLearner. This corresponds to the mathematical theory on tree building, since on each level we have to calculate the correlation of all Xi and Y in order to determine the best split feature on DTLearner but only randomly choose the feature on RTLearner. So for RTLearner, we sacrifice the predicting accuracy to increase efficiency on model building.

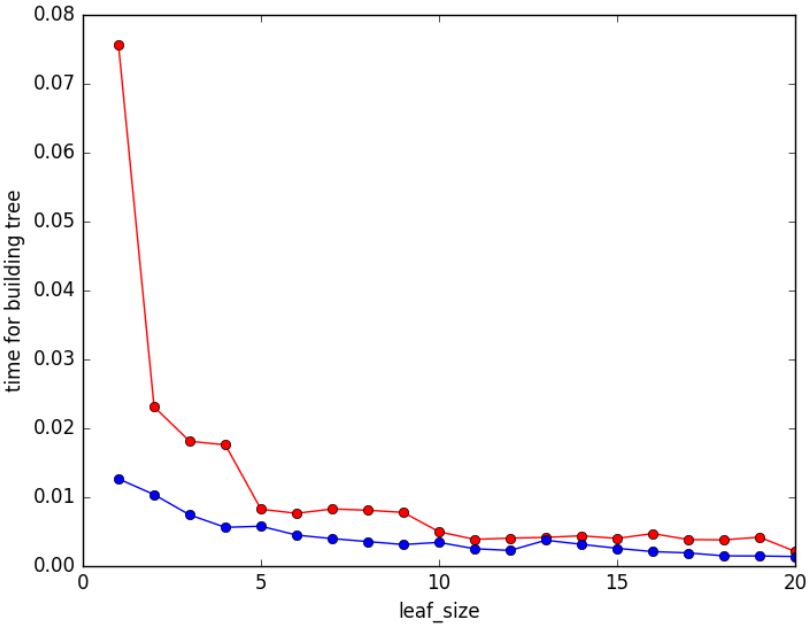

Figure 6: The tree building time of DTLearner and RTLearner