Assess Learner Report

Shrikanth Mahale

Abstract—Decision Tree Learner Algorithm Definition

Experiment 1: Overfitting in DT Learner

1. **Question: Research and discuss overfitting as observed in the experiment. (Use the dataset Istanbul.csv with DTLearner). Support your assertion with graphs/charts. (Do not use bagging in Experiment 1). At a minimum, the following question(s) that must be answered in the discussion:**
   - **Does overfitting occur with respect to leaf_size?**
   - **For which values of leaf_size does overfitting occur? Indicate the starting point and the direction of overfitting. Support your answer in the discussion or analysis. Use RMSE as your metric for assessing overfitting.**

Answer:

DT Learner a decision tree algorithm where features of the tree are selected based on its correlation with the response variable. Median of the feature with highest absolute correlation is calculated and split occurs based on the median. This process is made to occur recursively till a desired leaf size is achieved.

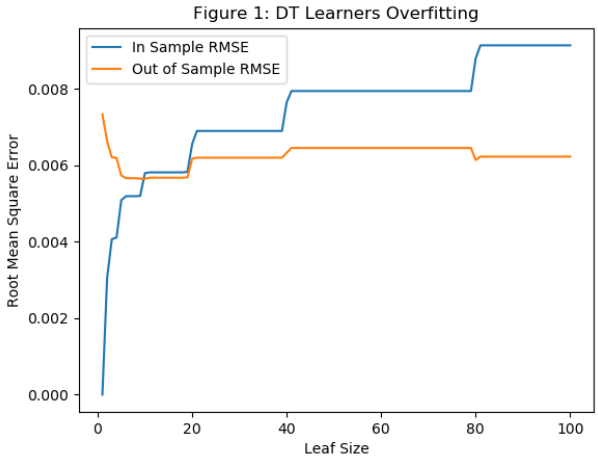
In order to answer this question, a DT learner was algorithm was applied to the **Istanbul.csv** dataset. The learner was implemented in loop for leaf size ranging from 1 to 100 and root mean squared error for In Sample data and Out of sample data was calculated for each leaf size in the loop. The result was plotted as shown in Figure 1 below.

If we notice figure 1, we can see that In Sample RMSE and Out of Sample RMSE are apart from each other for leaf-size approximately up to 7, and lower the value, more the distance. This shows that, lesser the leaf size, the model/learner tends to fit very closely specifically to the training dataset and does a bad job of predicting the test data set. But as leaf size starts increasing the learner gets more data points per leaf and it becomes more liberal in learning the data. So we can say that there is overfitting leaf size ranging from 1 to approximately 7.

If we further observe Figure 1, The In-Sample RMSE is steadily increasing as leaf size increases, but the Out of Sample error stabilizes with small increments at some points. Approximately, beyond leaf size = 10, the In-Sample RMSE

surpasses the Out of Sample RMSE. This indicates that the learner is struggling to study the training data after a certain leaf size and is starting to underfit with increasing leaf-size. The underfitting gets even bigger beyond leaf size = 80.

In order to get the best prediction using the learner, I would keep my leaf size between 10 and 20.



Experiment 2: Effects of Bagging on overfitting.

2. **Question: Research and discuss the use of bagging and its effect on overfitting. (Again, use the dataset Istanbul.csv with DTLearner.) Provide charts to validate your conclusions. Use RMSE as your metric. At a minimum, the following questions(s) must be answered in the discussion.**
   - **Can bagging reduce overfitting with respect to leaf_size? Can bagging eliminate overfitting with respect to leaf_size? To investigate this, choose a fixed number of bags to use and vary leaf_size to evaluate. If there is overfitting, indicate the starting point and the direction of overfitting. Support your answer in the discussion or analysis.**

Answer:

Bagging is a method where subset of data is collected into desired number of group aka Bags and an appropriate machine learning algorithm is applied to each group/bag. In this experiment DT learner algorithm is applied to each bag.

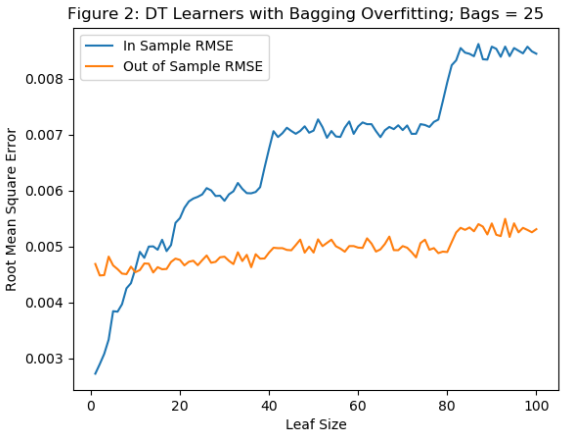
As suggested in the question a fixed bag size and a variable leaf size was chosen. The details given below:

Bag Size = 25
Leaf Size = 1 to 100
Like in in the previous question, bag learner was implemented in loop for leaf size ranging from 1 to 100 for each of the 25 Bags and root mean squared error for In Sample data and Out of sample data was calculated for each leaf size in the loop.

Looking at Figure 2 below, like in experiment 1, the In-Sample RMSE is steadily increasing as leaf size increases and surpasses the Out of Sample RMSE at approximately Leaf Size = 10. The Out of Sample happens to be stable throughout with few ups and downs but no major jumps like in experiment 1.

The Overfitting trend we observed in Experiment 1 is not seen in this case due to the stable nature of Out of Sample RMSE line. <u>So, it is safe to say that Bagging has reduced small amount of Overfitting that occurred in Experiment 1.</u>

Overall, it is observed that Bagging did not bring a significant change to the way how learner performs but the Figure 2 as compared to Figure 1 also suggests that RMSE rates are lower for both In-Sample and Out of Sample records.



<u>Experiment 3: DT Learner vs. RT Learner</u>

3. **Question: Quantitatively compare "classic" decision trees (DTLearner) versus random trees (RTLearner). Provide at least two new quantitative measures in the comparison. Using two similar measures that illustrate the same broader metric does not count as two separate measures. (For example, do not use two measures for accuracy.) Note for this part of the report you**

must conduct new experiments, don't use the results of the experiments above for this. Importantly, RMSE and correlation are not allowed as a new experiment. Provide charts to support your conclusions. At a minimum, the following question(s) must be answered in the discussion.

- **In which ways is one method better than the other?**

Answer:

DT Learner algorithm is a decision tree algorithm where features are chosen based on its correlation with its response variable and DT Learner algorithm is a decision tree algorithm where features are randomly chosen. They can be compared based on certain measures. My chosen measures are Run Time of algorithm and Mean Absolute error for accuracy.

Both DTLearner as well RTLearner algorithms were executed using the Istanbul dataset as the input dataset. The algorithms were run using leaf size equal to 1 up to 100, and the both the chosen measures were calculated to run was calculated in each iteration.

Measure 1: Run time of Algorithm

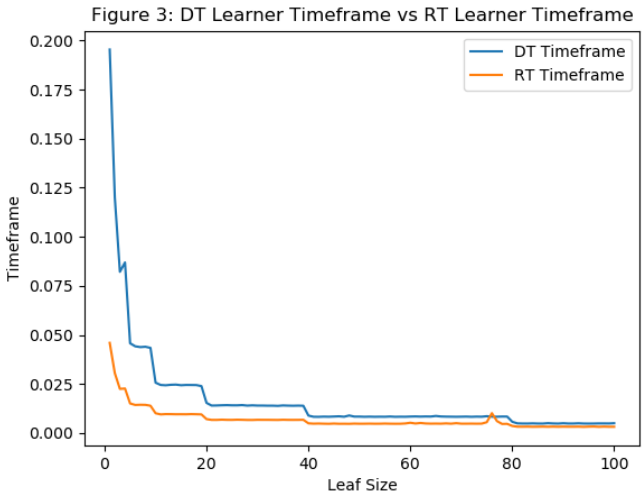
Our first measure of comparison we are choosing is run time of both the algorithms. Runtime in my opinion is an important measure specially if we want to build an AI application that uses data in millions of records has to be quick in results. In Istanbul dataset, even though the data is small, the runtime tells a lot about both the algorithm.

Figure 3 shows the time calculated to run DTLearner and RTLearner on Istanbul dataset, plotted as a line chart.

Initially, RTLearner performs far better than DTLearner. This occurs when the leaf size is small. This is due to the fact that DTLearner requires more time to calculate correlation of each feature with the response variable. The feature with the highest absolute correlation is chosen and the median is calculated for that feature. RTLearner on the other hand, will randomly pick a feature to split the dataset without any calculation, thus reducing the execution time.

However, as the leaf size is increased, the difference in execution times for DTLearner and RTLearner is diminished. This is because with a higher leaf size, the number of splits reduce, which in turn reduces the number of time correlation is calculated on the split datasets, which results fast execution of output. Looking at the figure 3 below, approximately after leaf size of 80, DTLearner approaches near RT Learner in execution time.

In general, though the feature selection in RTLearner may not be as good as in DTLearner, RTLearner compensates with improved efficiency.



Measure 2: Mean Absolute Error

Our second measure of comparison we are choosing is Mean Absolute Error (MAE) of both the algorithms which helps is determining accuracy. Figure 4 shows the Out of Samples MAE of DTLearner and RTLearner, plotted as a line chart.

To begin with my personal opinion, I always thought DTLearner would be more accurate, as the feature selection is based on correlation which makes it better suited to the data than the tree created using RTLearner where features are chosen randomly. Figure 4 below proves my assumption.

Given below is the formula of accuracy

Accuracy = (Number of Correct Predictions)/(Total Predictions made)

Looking at the above formula, it shows that lower the MAE, higher the accuracy. When we see the figure 4 picture below, DT learner overall has lower MAE as compared to RT Learner and DT learner shows the best accuracy (lowest MAE) for leaf size approximately between 10 to 40 where MAE is very near to zero. As leaf-size increases, algorithm takes the mean of all the values of the response feature which can result in increase in MAE as the mean value may be far from the actual value. Ever after that we still see DT learner having better accuracy than RT learner up to leaf size of 80.

RT Learner on other hand, as seen in Fig 4, is all over the place throughout due to its random nature of selecting features. RTLearner performs better than

DTLearner for few leaf-sizes above 80, which may either mean that the learner has randomly happened to choose the correct feature for this dataset or it can also mean that after a certain leaf size the feature to be selected does not matter. In conclusion I would say, RTLearner is a better choice when we have very large dataset with wide varieties of features that have low absolute correlation to the response feature. But if the data features are well correlated then DT Learner is a better choice.

Figure 4: DT Learner MAE vs RT Learner MAE - Out of Samples Comparison