CS 7646 – ML4T – Project 3 Assess Learners

Christopher Fleisher (GTID# 903421975)

Does overfitting occur with respect to leaf_size for DTLearner?

Yes, DTLearner overfits the Istanbul dataset based on the leaf_size parameter. When the leaf_size is sufficiently small, the DTLearner is effectively memorizing the training set which leads to poor generalization on the testing set. To test for overfitting, I first split the data into 10 randomly selected training and testing subsets. For each in-sample (IS) and out-of-sample (OOS) dataset I then measured the DTLearner RMSE across increasing leaf_sizes. When the leaf_size is very small, the IS RMSE is materially lower than the OOS RMSE indicating that the DTLearner is learning the training data noise instead of the underlying function. In Figure 1, overfitting is occurring prior to the highlighted green area (up thru ~leaf size 6). As the leaf size increases initially, the IS and OOS RMSE converges quickly as the learner is prevented from memorizing data examples and instead required to generalize feature components based on the expected target value of the leaf node constituents. After this initial convergence, the OOS RMSE steadily increases in-line with the IS RMSE, indicating a lack of depth is preventing the model from learning the concept. The figure's highlighted region where IS and OOS RMSE converge provides appropriate leaf size options for balancing the overfitting and underfitting.

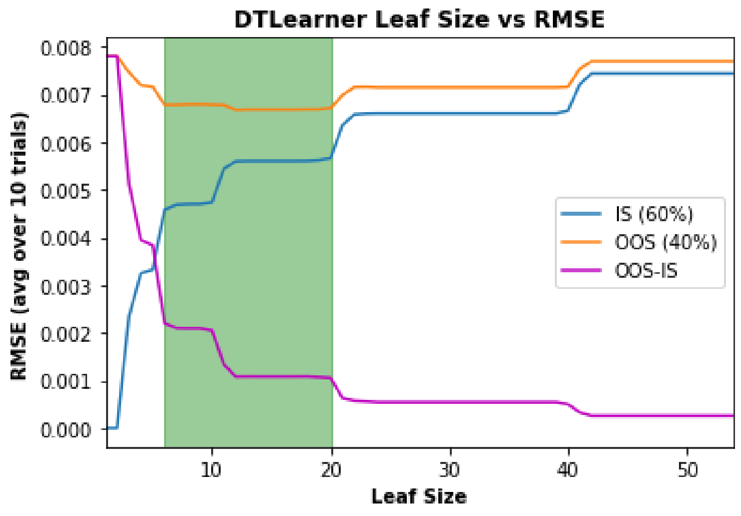

Figure 1. DTLearner Leaf Size vs RMSE Experiment

Can bagging reduce or eliminate overfitting with respect to leaf size?

Bagging can help reduce overfitting with respect to leaf size, but it cannot eliminate the concern. Overfitting results from the model learning the noise in the training examples instead of the inherent concept. Although bagging is able to mitigate the impact noisy examples have on a given DTLearner model, it does not fundamentally alter the feature selection process resulting in each DTLearning making similar decisions at each node. If the data samples are independent and identically distributed then the DTLearner's will construct models based on a systematically similar features selection process at each non-leaf depth node. Unfortunately, this means that while bagging may reduce variance caused by the leaf nodes without impacting the expected value, highly correlated decisions limit the search space making the model vulnerable to data variance. For the Istanbul dataset, bagging meaningfully mitigated the impact of overfitting as can be seen in the Figure 2 plot. To test this, I again separated the data into 10 in-bag (IB) and out-of-bag (OOB) subsets and then measured the RMSE using the BagLearner. I fixed the number of bags at 25 and again maintained a max depth of 20 for the underlying DTLearner. The plot shows the average IB and OOB RMSE for the 10 trials across leaf sizes. The IS RMSE is inversely related to the leaf size, but the slope of the curve has been significantly reduced highlighting the positive impact bagging has on reducing variance at the leaves. The OS RMSE also benefits from bagging, largely eliminating the initial decline in RMSE seen in the prior DTLearner experiment. Although the OOB plot is flatter, there is still a slight decline in RMSE until ~ leaf size 5 at which point the trend steadily increases. More importantly, the OOB curve is materially flatter than that of the IB curve for small leaf sizes, but the differential is reduced as the leaf size increases. This indicates that there is still overfitting occurring up until the curve differential steadies. It should be noted that the max depth constraint likely bounds the leaf size parameter for larger datasets, thereby flattening the RMSE plot's curvature.

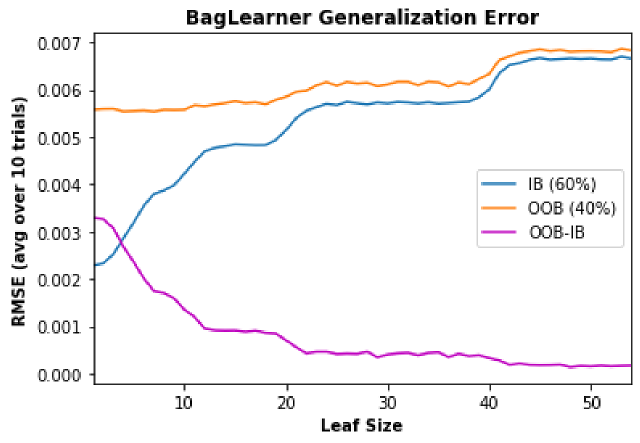

Figure 2. BagLearner Generalization Error Experiment

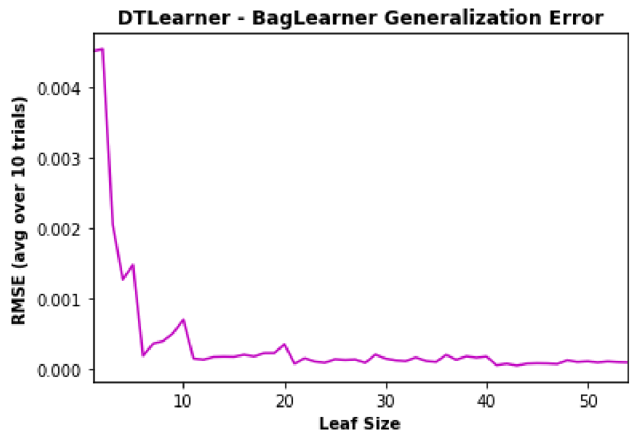

Figure 3. DTLearner - BagLearner RMSE Difference

Quantitatively compare "classic" decision trees versus random trees.

Random trees help to reduce model variance at the cost of increased bias and build time. I compared the bias-variance tradeoff between the two techniques by measuring the prediction variance across the same randomly sampled 10 IB and OOB datasets. I built 10 BaggedLearners with DTLearner as the underlier and 10 BaggedLearners with RTLearner as the underlier (20 total) and tracked the variance across bag sizes while keeping the leaf size constant at 6 based on the previously discussed experiments. The result was that the BaggedLearner based on the DTLearner had consistently higher average prediction variance over the 10 different datasets across all bag sizes than did the RTLearner (Figure 4). The DTLearner's higher variance likely resulted from the model's inherently correlated approach to feature selection as compared to the RTLearner's random approach. For each new trial the DTLearner constructed a tree by selecting the features with the highest absolute correlation to the target. Since each trial's dataset was drawn from the same distribution the learner selected similar features at each depth. The DTLearner has a strong opinion and it's not afraid to share it when it encounters highly correlated features in the OOB samples. Even though the samples were all drawn independently from the same distribution, the random differences in composition, which we colloquially referred to as noise, manifested higher prediction variance as the DTLearner had not fully explored that portion of the feature space. Put differently, DTLearner will make very different predictions for the same example across models constructed from different sample distributions. Although this isn't necessarily bad, it does make the DTLearner susceptible to overfitting since it is inclined to treat variance in the dataset as a signal rather than noise. The random approach enabled different RTLearner trees to explore alternative paths through the feature space which helped reduce the model's reliance on the given data set for the OOB predictions, but at the cost of introducing bias. When the RTLearner encounters the OOB examples it's relative indifference to correlation means that only inherently strong structural relationships are expressed in the model. While this helps RTLearner generalize across OOB samples, it also makes it susceptible to poor accuracy. Although I didn't measure the relative RMSE of the two approaches given the rubric, the reduction in structural variance should translate into improved generalization up until underfitting results caused by the increased bias. I would expect that the RMSE plot would have a flatter curvature to the prior two experiments unless the models were underfitting the data.

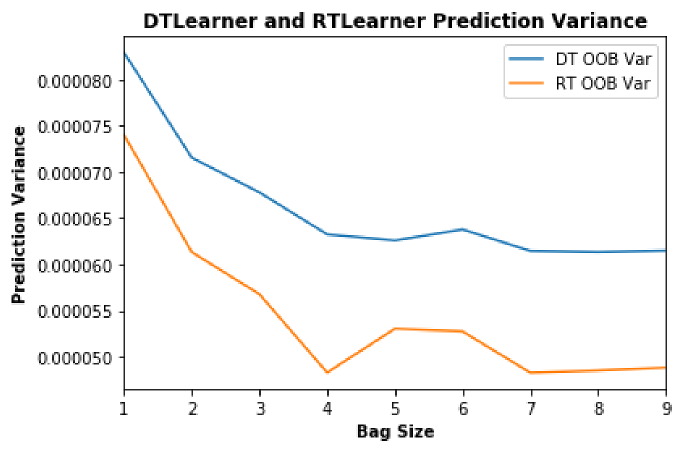

Figure 4. DTLearner and RTLearner Prediction Variance Experiment

To quantify the relative build cost of each technique I measured the total number of nodes the DTLearner and RTLearner used to represent the examples over 10 independent trials. All else being equal, the optimal model should be able to represent the underlying concept as efficiently as possible, which in this case would mean having fewer nodes in the tree. Unsurprisingly, the DTLearner required fewer nodes than the RTLearner to represent the same data examples for each trial. This intuitively makes sense because the DTLearner is evenly splitting examples at each depth by using the median as the split value. The RTLearner is randomly interpolating between the randomly selected feature values of two randomly selected examples. Whereas the median ensures an even distribution, the random split often results in uneven example allocations at subsequent depth levels. This indeterminism also happens to be at the root of what enables the random process to reduce prediction variance. As seen in Figure 5, the RTLearner's node counts were not only higher, but also exhibited greater variability. The reduction in structural determinism costs more from a build time perspective, but may potentially help the model distinguish between noise and the underlying trend.

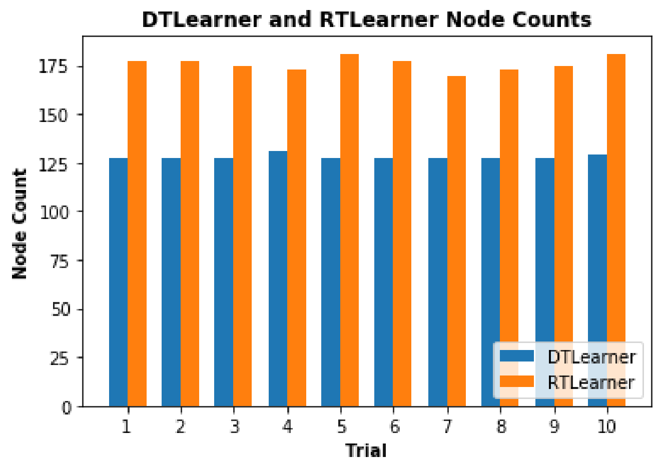

Figure 5. DTLearner and RTLearner Node Count Experiment