Project 3 Report

Assess Learners

Juejing Han
jhan446@gatech.edu

***Abstract*—**In this project, the performance of the regression tree (DTLearner) and random tree (RTLearner) are compared, and the bagging strategy is conducted to reduce overfitting. Results show that DTLearner yields smaller errors than RTLearner. However, RTLearner bears less training time and bagged RTLearner yields a smaller out-of-sample error than that of DTLearner. Overfitting occurs in both DTLearner and RTLearner when leaf size is small, and bagging can reduce and might ultimately eliminate overfitting.

1 INTRODUCTION

"A decision tree is a decision support tool that uses a tree-like model of decisions and their possible consequences."[1] It contains nodes (representing the features and thresholds/split values) to split data and leaves (or end nodes) to return the final outcomes. Classification and Regression Trees (CARTs) is a classification method that uses historical data to construct a decision tree (Roman, 2004). In this project, four supervised machine learning algorithms from CARTs are implemented in Python by building three CARTs learners: DTLearner, RTLearner, and BagLearner.

Among these learners, the classic decision tree (DTLearner) selects the best feature (explicitly described in 2.1) to split on, while the random tree (RTLearner) uses a random feature.

A universe challenge for CARTs is overfitting, which occurs when a model fits the training data well but fails to generalize to new data, and it is encountered in

[1] https://en.wikipedia.org/wiki/Decision_tree

all supervised machine learning schemes (Paris, 2003). When overfitting occurs, the in-sample error decreases as the out-of-sample error increases.

Bagging is one of the solutions to overfitting. Its principle is to create an ensemble of learners to reduce variance and prevent overfitting by "smoothing out" the predictions.

Therefore, this project aims to test three hypotheses:

1) Overfitting occurs with respect to leaf size.

2) Bagging can prevent or even eliminate overfitting.

3) Since DTLearner carefully selects its split index, it is supposed to be better.

2 METHODS

2.1 DTLearner

Based on the algorithm by Quinlan (1986), a single decision tree (a regression tree) is built through binary recursive partitioning as the core of DTLearner. The "best feature to split on" in DTLearner is the feature (X_i) that has the highest absolute correlation with Y, and the median of the best feature is the split value, which determines the content of the left/right tree.

2.2 RTLearner

RTLearner is built on DTLearner, except that the best feature in RTLearner to split on is made randomly.

2.3 BagLearner

BagLearner is built on DTLearner to generate a learner ensemble. Training data is chosen randomly from the training data pool with replacement, i.e., the same individual training data can be chosen more than once.

2.4 Statistics

Root mean square error (RMSE) and mean absolute error (MAE) are used in evaluating the learners.

3 DISCUSSION

Three experiments are conducted on Istanbul data[2] with different learners introduced in section 2. For each training/testing process, 60% of Istanbul data is randomly selected as the training data, and the rest 40% is the testing data, so each training/testing shares different random-picked inputs from the same original data source (i.e., Istanbul). Each experiment repeats 10 times, and the analysis is based on its average result.

3.1 Experiment 1 (DTLearner)

Overfitting occurs with respect to leaf size when leaf size $\leqslant$ 10 (Fig. 1). It starts from leaf size = 10 and goes all the way towards 1. The in-sample error approaches 0 when the leaf size approaches 1. Meanwhile, since the model tends to present the exact characteristics of the training set, it fails to generalize to other data; therefore, the out-of-sample error increases.

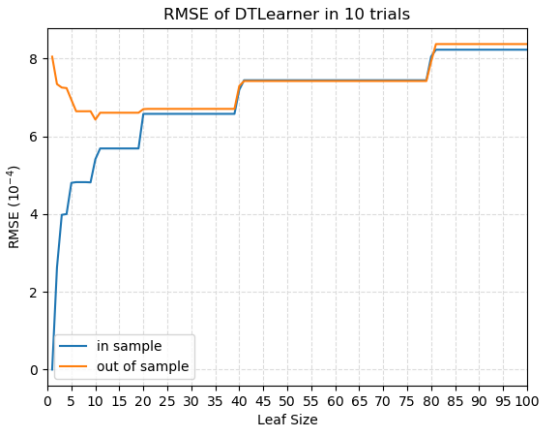

Figure 1—Root Mean Square Error of DTLearner in 10 trials

When leaf size is between 10 and 19, both in-sample and out-of-sample errors stay flat. It implies that the model captures significant signals from training data and generalizes itself to testing data.

[2] https://www.dropbox.com/s/dl/vcprhmffvh8m9dg/Istanbul.csv.zip

In general, when leaf size > 19, both errors increase as the leaf size increases, and underfitting tends to occur. This means that the model tends to be too simple (or the leaf size tends to be too large) to perform well with both data sets.

3.2 Experiment 2 (BagLearner)

Bagging can reduce overfitting with respect to leaf size. Overfitting reduces as bag size increases, and ultimately, bagging eliminates overfitting.

Fig. 2 and 3 show the result with 2 and 20 bags of DTLearner respectively. Overfitting exists with 2 bags when leaf size $\leqslant$ 10 (Fig. 2). It starts from leaf size = 10 and goes all the way towards 1. As the number of bags increases, overfitting tends to reduce. When bag size = 20, both error lines become smooth, and the out-of-sample error has an overall decreasing trend as the leaf size decreases (Fig. 3). Although there are slight fluctuations of out-of-sample error as the leaf size approaches 1, overfitting generally disappears.

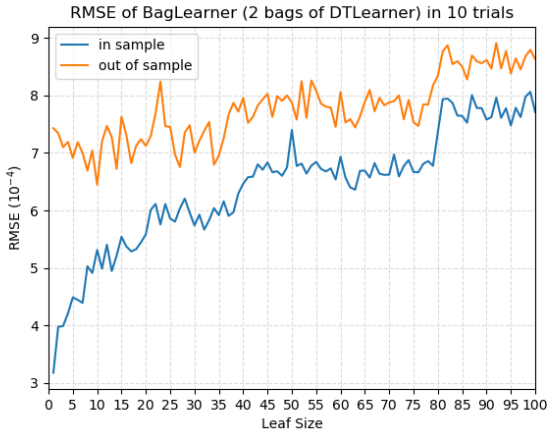

Figure 2—Root Mean Square Error of BagLearner (2 bags of DTLearner) in 10 trials

Fig. 2 and 3 show the result with 2 and 20 bags of DTLearner respectively. Overfitting exists with 2 bags when leaf size $\leqslant$ 10 (Fig. 2). As the number of bags increases, overfitting tends to reduce. When bag size = 20, both error lines become smooth, and the out-of-sample error has an overall decreasing trend as the leaf size decreases (Fig. 3). Although there are slight fluctuations of out-of-sample error as the leaf size approaches 1, overfitting generally disappears.

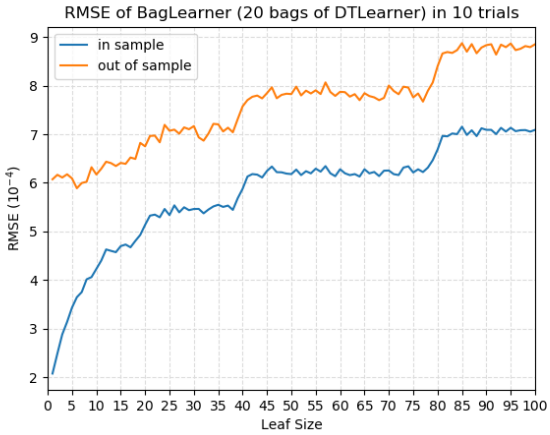

Figure 3—Root Mean Square Error of BagLearner (20 bags of DTLearner) in 10 trials

Meanwhile, bagging reduces out-of-sample error (Fig. 1-3) for it reduces variance within the training data and yields more accurate predictions. Although it seems that more bags are better, the 20-bag ensemble already has a low variance, so more bags do not significantly improve its performance but could be computationally expensive.

3.3 Experiment 3 (DTLearner vs. RTLearner)

RTLearner uses less training time, while DTLearner yields smaller errors, and statistical operations (for selecting the best features) play an important role here. For a single learner, neither DTLearner nor RTLearner is always superior to the other. However, applying strategies like bagging can improve RTLearner's performance.

3.3.1 *Training time*

Because of the straightforward strategy of locating the best feature (i.e., a random pick) to split data, RTLearner uses less training time than DTLearner dose (Fig. 4).

Additionally, when leaf size increases to a certain amount (40 in this experiment), the training time of both learners tends to converge though DTLearner still needs more training time. For a given training data, a bigger leaf size means less frequency to split, and hence less accumulative advantage of RTLearner's fast training time.

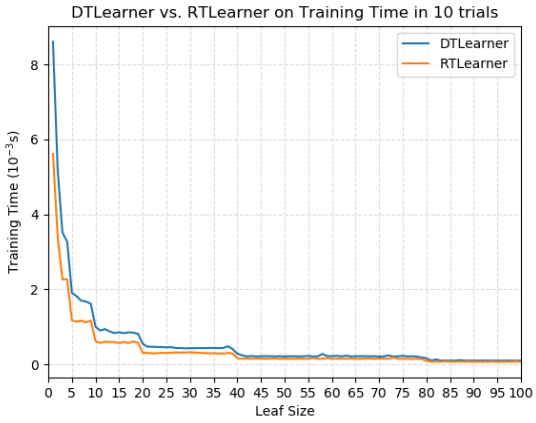

Figure 4—DTLearner vs. RTLearner on training time in 10 trials

3.3.2 *Mean Absolute Error (MAE)*

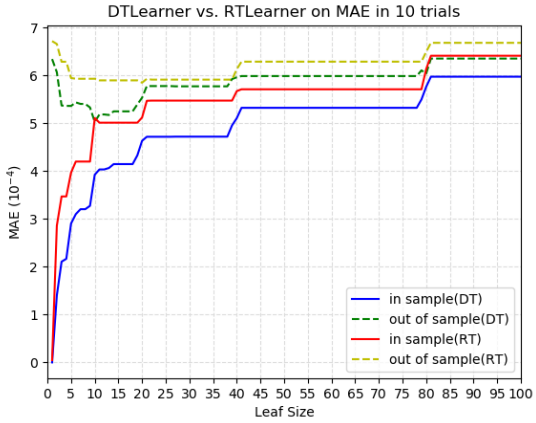

Figure 5—DTLearner vs. RTLearner on mean absolute error in 10 trials

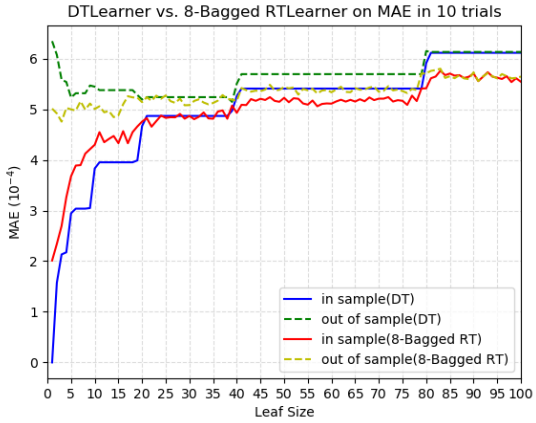

Figure 6—DTLearner vs. 8-bagged RTLearner on mean absolute error in 10 trials

For mean absolute error (MAE), DTLearner and RTLearner share the same trend, while the MAE (either in-sample or out-of-sample) of RTLearner is larger than that of DTLearner because the best features in DTLearner are selected by statistical reasoning (Fig. 5). Besides, overfitting occurs in both learners. However, bagged RTLearner can defeat DTLearner by yielding smaller out-of-sample MAE (Fig. 6).

3 SUMMARY

In this project, the hypotheses raised in section 1 are examined by three experiments with the following conclusions:

1) Overfitting occurs in DTLearner/RTLearner with respect to leaf size. It tends to happen as leaf size decreases (starts from a typical leaf size N to 1, and in this project, N = 10) because the model tends to be more specific about the training data set as leaf size decreases, and hence it fails to fit other data.

2) Bagging is an ensemble scheme that can reduce overfitting. As the number of bags increases, the fluctuation of error becomes "smoother" and the overfitting becomes slighter. In this project, results show that bagging can eliminate overfitting. Nevertheless, more experiments with different data sets, which might bear different noises and characteristics, should be conducted for a sound and confident conclusion. To save computational time, average outcomes of 10 trials are conducted in this project, but more trials (such as Monte Carlo method) should be introduced to yield a more reliable result.

3) For training data, DTLearner is more accurate because of the statistical approach it adopts. On the other hand, RTLearner uses less training time than DTLearner because it is not subject to a series of statistical calculations while locating the best features to split data. The combination of RTLearner and bagging yields a smaller out-of-sample error than that of DTLearner. More experiments need to be done to thoroughly compare DTLearner and RTLearner, such as introducing more metrics (mean absolute percentage error, coefficient of determination) or applying them in a real-case prediction, and etc.

4 REFERENCES

1. Roman Timofeev (2004). Classification ad Regression Trees (CART) Theory and Applications. Master Thesis, Humboldt University. Berlin, Germany.
2. Paris, G., Robilliard, D., Fonlupt, C. (2004). Exploring Overfitting in Genetic Programming. Lecture Notes in Computer Science, vol 2936. Springer.
3. Quinlan, J.R. (1986). Induction of Decision Tree. Mach Learn 1, 81-106. https://doi.org/10.1007/BF00116251.