Project 3: ASSESS LEARNERS

Jing Yu
jyu497@gatech.edu

Abstract— Practice supervised learning algorithms on regression problems. Implement four supervised learners: Decision tree learner, Random tree learner, Bootstrap aggregating learner and Insane learner. Assess and compare their performances and the effects of overfitting based on metrics including RMSE, correlation and MAE.

1 INTRODUCTION

The purpose of this project is to practice supervised learning algorithms on regression problems, including training, hyperparameter tuning, predicting and assessing performance. Four supervised learners are implemented: Decision tree learner, Random tree learner, Bootstrap aggregating learner and Insane learner. As regression learners, continuous numerical results are returned. Several experiments are conducted to evaluate and compare the behaviors of these learners using multiple metrics.

Decision tree [1] is a tree-like model, divides and conquers the problem based on a sequential decision process. Nodes are questions for each feature, edges are answers to that questions and leaves are final outputs or values. During learning time, a decision tree is built from the training set. Starting from the root, the data is recursively split into subsets until a stopping criterion is reached. During query time, follow the answer path until a leaf is reached.

In order to best classify the data, multiple approaches can be used to determine the good feature to split on at each node: maximizing information gain (minimizing entropy after split), maximizing correlation with targets or minimizing gini index.

Decision tree can describe non-linear dependencies. It does not need normalization or feature scaling and it is robust to noise and missing data. However, since the features are greedily/locally chosen, it may converge to local optima. And it is prone to overfitting if having unconstrained depth. When overfitting happens, prune the decision tree by restricting the max depth or increasing the min samples per leaf (leaf size).

Random trees are similar to Decision trees except they select random features and values to group the data at each node. This takes less computations therefore it has less learning time to build the tree. While it potentially has deeper tree which may increase the query time. Besides, it may be more sensitive to errors or protuberances in the data.

Ensemble learning is a better solution for the problems of decision tree. It trains multiple weak learners (usually decision tree), lets each learner make its predictions and aggregates those predictions to form a strong learner. The collaborated prediction is more robust and less prone to errors, therefore has better performance. Bootstrap aggregating/bagging is one example of ensemble learning. Each learner takes random subset of training data sampled with replacement and average the results of all learners. It has less overfitting since overfitting a subset will not overfit the overall dataset, the average will smooth out the biases of each individual learner.

Multiple metrics are using to tune hyperparameters and analyze training results:

$$RMSE = \sqrt{\frac{1}{n}\sum_{i=1}^{n}(\hat{y}_i - y_i)^2}$$

$$Correlation = \frac{Covariance\ of\ \hat{y}_i\ and y_i}{\sqrt{Variance\ of\ \hat{y}_i \times Variance\ of\ y_i}}$$

$$MAE = \frac{1}{n}\sum_{i=1}^{n}|\hat{y}_i - y_i|$$

2 METHODS

Decision tree learner: Simple decision tree picks the best feature having highest absolute value correlation with Y then splits on the median value of that feature.

Random tree learner: Simple decision tree picks a random feature then splits on the average of two random values of that feature.

For tree-based learners, they have a “leaf_size” hyperparameter that defines the maximum number of samples to be aggregated at a leaf.

Bag learner: Bootstrap aggregating learner supports any learner including Decision tree, Random tree, Linear regression, even another Bag learner. Each learner/bag is trained on a different subset of the data, same sized and sampled with replacement. It has another "bags" hyperparameter that defines the number of learners to train.

Insane learner: Contains 20 Bag learners where each learner is composed of 20 Linear regression learner.

The dataset used is Istanbul.csv, including the returns of multiple worldwide indexes for several days in history. The objective is to predict the return for the MSCI Emerging Markets (EM) index based on other index returns. It has 8 features and 536 instances. Although it is a time-series dataset, I ignore the time information and treat it as a non-time series dataset. For all experiments, I randomly select 60% of the data as training set and use the remaining 40% as test set.

Experiment 1: Evaluate overfitting of Decision tree learner with respect to leaf_size. Use RMSE as metric.

Experiment 2: Evaluate overfitting of bags of Decision tree learners with respect to leaf_size and bags. Use RMSE as metric.

Experiment 3: Quantitatively compare Decision tree learner with Random tree learner:

1. Train leaf_size=1 Decision tree learner and Random tree learner 100 times with training data to get the total training time and average maximum height.

2. Query leaf_size=1 Decision tree learner and Random tree learner 100 times with training data to get the total query time.

3. Evaluate and compare the performance and overfitting of Decision tree learner with Random tree learner. Use MAE as metric.

3 DISCUSSION

3.1 Experiment 1

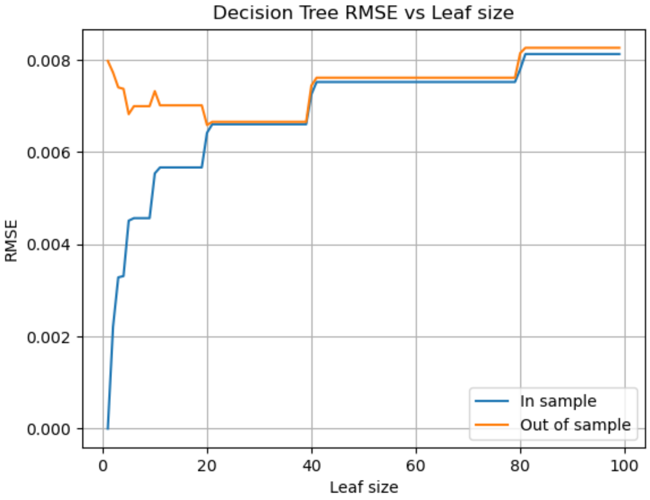


Figure 1—The RMSE results of Decision tree learner with respect to tree_leaf on training and test set.

Leaf_size of value 1 to 99 are tested. The in sample results are from training set and out of sample results are from test set.

When in sample error decreasing while out of sample error starting to increase, it indicates overfitting. From Figure 1, overfitting does occur at around leaf_size 20 and it getting worse with smaller leaf_size.

When overfitting, it means that the model is too complex and overcommits to the data. If smaller leaf_size is used, the decision tree tends to be deeper. As described in Section 1, decision tree is prone to overfitting if having unconstrained depth. The solution is increasing the leaf_size to early stop the splitting or pruning the tree.

3.2 Experiment 2

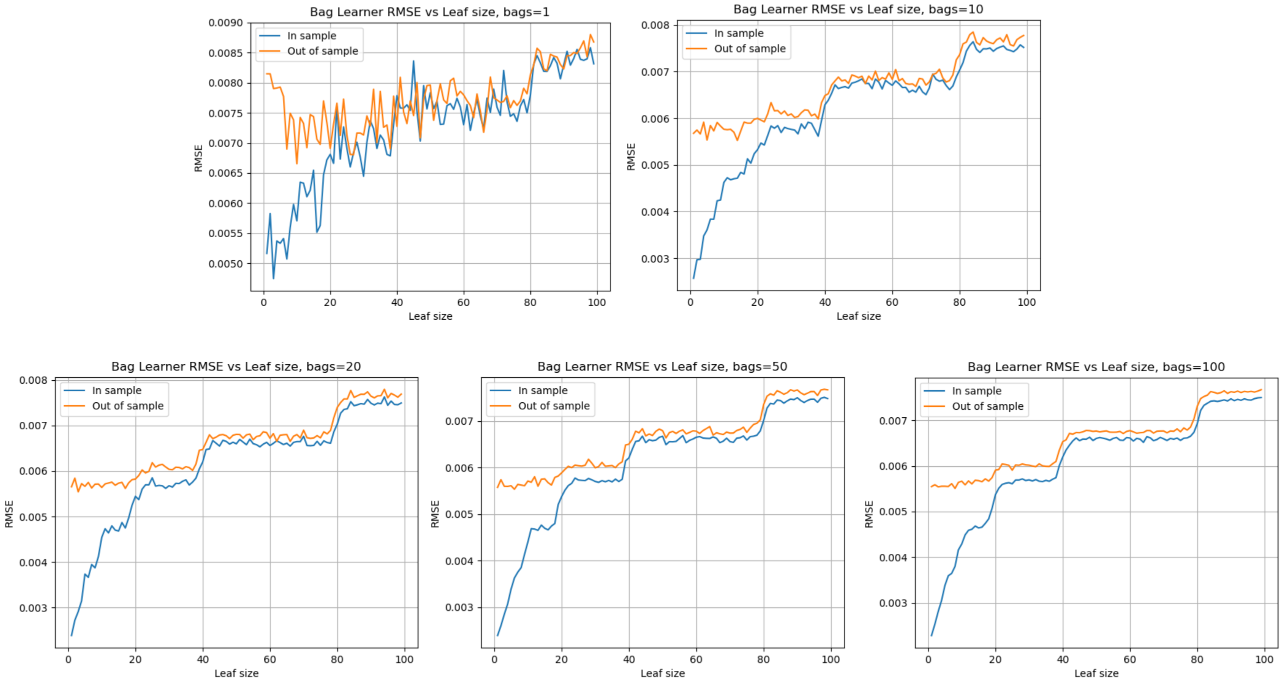


Figure 2 — The RMSE results of bag of Decision tree learners with respect to tree_leaf and bags on training and test set.

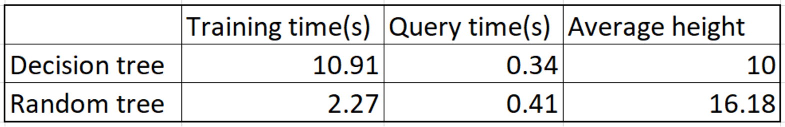
Leaf_size of value 1 to 99 and bags of value 1, 10, 20, 50, 100 are tested. The in sample results are from training set and out of sample results are from test set. The bags=1 means a single Decision tree learner is trained on a subset of data.

By define of overfitting, it occurs when bags=1 but does not occur with larger value of bags from Figure 2. Although the out of sample error doesn't increase, it starts to flatten out and diverge from in sample error at around leaf_size 20 and it getting worse with smaller leaf_size. It's safer to say that bagging does reduce overfitting but does not eliminate overfitting.

From the results of Experiment 1, if smaller leaf_size is used, a single decision tree tends to overfit. While bagging aggregates the predictions of multiple decision trees. Overfitting a subset will not overfit the overall dataset, the average will smooth out the biases of each individual learner. Therefore, bagging has less overfitting than a single decision tree. Besides, since the collaborated prediction is more robust and less prone to errors, bagging has better performance. This can be proved by with larger bags, the in sample and out of sample errors are much smaller compared to a single decision tree and the curves are getting smoother.

3.3 Experiment 3

Table 1 — 100 times total training time, total query time and average height of leaf_size=1 Decision tree learner and Random tree learner.



	Training time(s)	Query time(s)	Average height
Decision tree	10.91	0.34	10
Random tree	2.27	0.41	16.18

Table 1 shows the results of training and query leaf-size=1 Decision tree learner and Random tree learner 100 times with training data, which in accordance with the assumptions in Section 1. Random trees select random features and values to group the data at each node. This takes less computations therefore the training time to build the tree is much shorter. And since the split is done randomly, the random tree tends to be unbalanced and deeper, thus the query time is longer.

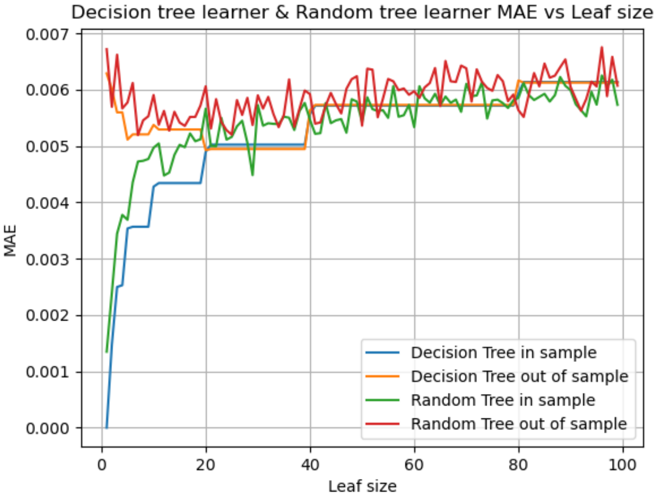


Figure 3 — The MAE results of Decision tree learner and Random tree learner with respect to tree_leaf on training and test set.

Leaf_size of value 1 to 99 are tested. The in sample results are from training set and out of sample results are from test set.

Figure 3 shows the results of Decision tree learner vs Random tree learner based on MAE. Because of the random nature of random tree, both the in sample and out of sample results are noisier than decision tree. It's more sensitive to errors since it may accidently choose the protuberances in the data to split on. This could be proved by the worse average performance. While there are instances that random tree outperforms decision tree. Since random tree may avoid the local optimal that stuck the decision tree.

Overfitting affects them in similar way. Both random tree and decision tree overfit at around leaf_size 20 and it getting worse with smaller leaf_size.

4 SUMMARY

In conclusion, a single decision tree tends to overfit with small leaf_size. This could be alleviated by increasing the leaf_size to prune the decision tree or using bagging instead. Bagging has less overfitting and better performance than a single decision tree. Random tree has less learning time, longer query time, unbalanced deeper tree structure and worse performance than decision tree. While it may avoid the local optimal that affects the decision tree. Overfitting affects decision tree and random tree in similar way.

5 REFERENCES

1. Quinlan, J.R. Induction of decision trees. Mach Learn 1, 81–106 (1986).
2. Bootstrap aggregating. https://en.wikipedia.org/wiki/Bootstrap_aggregating