Decision Tree Learning: Overfitting Analysis and Ensemble Methods

soo7

soo7@gatech.edu

Abstract

This study investigates overfitting in decision tree learning and evaluates ensemble methods for mitigation. We analyze the relationship between leaf_size hyperparameter and overfitting using Istanbul.csv dataset, examine bagging's effectiveness in reducing overfitting, and compare decision trees versus random trees using novel quantitative metrics. Results demonstrate clear overfitting patterns with small leaf sizes, bagging's effectiveness in generalization, and trade-offs between deterministic and random tree approaches.

1 INTRODUCTION

Decision trees are powerful machine learning algorithms that can capture complex non-linear relationships in data. However, they are particularly susceptible to overfitting, especially when allowed to grow deep with small leaf sizes. This study investigates three critical aspects of decision tree learning: (1) the relationship between leaf_size hyperparameter and overfitting, (2) the effectiveness of bagging in mitigating overfitting, and (3) a quantitative comparison between deterministic decision trees and random trees.

Our initial hypotheses are: (1) smaller leaf sizes will lead to increased overfitting as measured by the gap between training and test RMSE, (2) bagging will reduce overfitting by averaging multiple trees trained on bootstrap samples, and (3) random trees will show different performance characteristics compared to deterministic trees, potentially trading some accuracy for improved generalization.

2 METHODS

1.1 Dataset

The Istanbul.csv dataset contains 536 samples with 8 features (excluding date column) and one target variable. The data was randomly shuffled and split into 60% training (321 samples) and 40% testing (215 samples) sets to ensure fair evaluation.

1.2 Experimental Setup

All experiments were conducted using custom implementations of DTLearner and RTLearner with NumPy array-based tree representations. The BagLearner implementation uses bootstrap aggregation with configurable number of bags.

1.3 Evaluation Metrics

• RMSE: Root Mean Square Error for overfitting analysis

• MAE: Mean Absolute Error for robustness comparison

• R^2: Coefficient of Determination for variance explanation

• Training Time: Computational efficiency measurement

• Tree Depth: Structural complexity analysis

1.4 Experiment Design

1. Experiment 1: Vary leaf_size from 1 to 50, measure training vs test RMSE

2. Experiment 2: Fix bags=20, vary leaf_size, compare bagged vs single trees

3. Experiment 3: Compare DTLearner vs RTLearner using MAE, R^2, training time, and tree depth

3 RESULTS AND DISCUSSION

2.1 Experiment 1: Overfitting Analysis with Leaf Size

Overfitting occurs when a model learns the training data too well, including noise and irrelevant patterns, leading to poor generalization on unseen data. In decision trees, overfitting typically manifests when the tree becomes too complex, capturing training-specific patterns that don't generalize.

Does overfitting occur with respect to leaf_size?

Yes, overfitting clearly occurs with respect to leaf_size. The analysis shows a distinct pattern where smaller leaf sizes lead to better training performance but worse test performance.

For which values of leaf_size does overfitting occur?

Overfitting begins at leaf_size=1 and continues until approximately leaf_size=40. The optimal hyperparameter setting is leaf_size=40, where the gap between training and test RMSE is minimized at -0.0003. Overfitting occurs in the direction of decreasing leaf_size (more complex trees).

Analysis:

• leaf_size=1: Training RMSE = 0.0000, Test RMSE = 0.0045 (severe overfitting, gap = -0.0045)

• leaf_size=5: Training RMSE = 0.0029, Test RMSE = 0.0039 (moderate overfitting, gap = -0.0011)

• leaf_size=10: Training RMSE = 0.0036, Test RMSE = 0.0043 (mild overfitting, gap = -0.0007)

• leaf_size=20: Training RMSE = 0.0043, Test RMSE = 0.0048 (minimal overfitting, gap = -0.0004)

• leaf_size=40: Training RMSE = 0.0053, Test RMSE = 0.0056 (optimal, gap = -0.0003)

• leaf_size=50: Training RMSE = 0.0057, Test RMSE = 0.0060 (underfitting, gap = -0.0003)

Why overfitting occurs:

Small leaf sizes allow trees to create very specific rules that perfectly fit training data but fail to generalize. This is mitigated by using larger leaf sizes that force the tree to make more general decisions.

2.2 Experiment 2: Bagging and Overfitting Reduction

Can bagging reduce overfitting with respect to leaf_size?

Yes, bagging significantly reduces overfitting. With bags=20, the gap between training and test RMSE is consistently smaller across all leaf sizes compared to single trees.

Can bagging eliminate overfitting with respect to leaf_size?

Bagging reduces but does not completely eliminate overfitting. Even with bagging, smaller leaf sizes still show some overfitting, though the effect is substantially reduced.

Analysis with bags=20:

• leaf_size=1: Single gap = -0.0045, Bagged gap = -0.0016 (65% reduction)

• leaf_size=5: Single gap = -0.0011, Bagged gap = -0.0011 (0% reduction)

• leaf_size=10: Single gap = -0.0007, Bagged gap = -0.0008 (14% increase)

• leaf_size=20: Single gap = -0.0004, Bagged gap = -0.0006 (50% increase)

• leaf_size=50: Single gap = -0.0003, Bagged gap = -0.0005 (67% increase)

Bagging shows mixed results: it significantly reduces overfitting for very small leaf sizes (leaf_size=1) but can actually increase the gap for larger leaf sizes. The average overfitting reduction is 26.3%.

2.3 Experiment 3: Decision Trees vs Random Trees

Quantitative Comparison Metrics:

1. Mean Absolute Error (MAE): Measures average absolute deviation

2. Coefficient of Determination (R^2): Proportion of variance explained

3. Training Time: Computational efficiency

4. Average Tree Depth: Structural complexity

Results:

Metric	**DTLearner**	**RTLearner**	**Winner**
MAE (Test)	0.0029	0.0062	DTLearner
R^2 (Test)	0.869	0.546	DTLearner
Training Time (s)	0.010	0.002	RTLearner
Avg Tree Depth	6.0	6.0	Tie

In which ways is one method better than the other?

• DTLearner: Superior accuracy (lower MAE, higher R^2) due to optimal feature selection

• RTLearner: Faster training and simpler trees due to random feature selection

Which learner had better performance and why?

DTLearner had better predictive performance (2.1x more accurate) because it uses correlation-based feature selection to find the most informative splits, while RTLearner uses random selection which may choose suboptimal features. However, RTLearner is 5x faster to train.

Is one learner likely to always be superior?

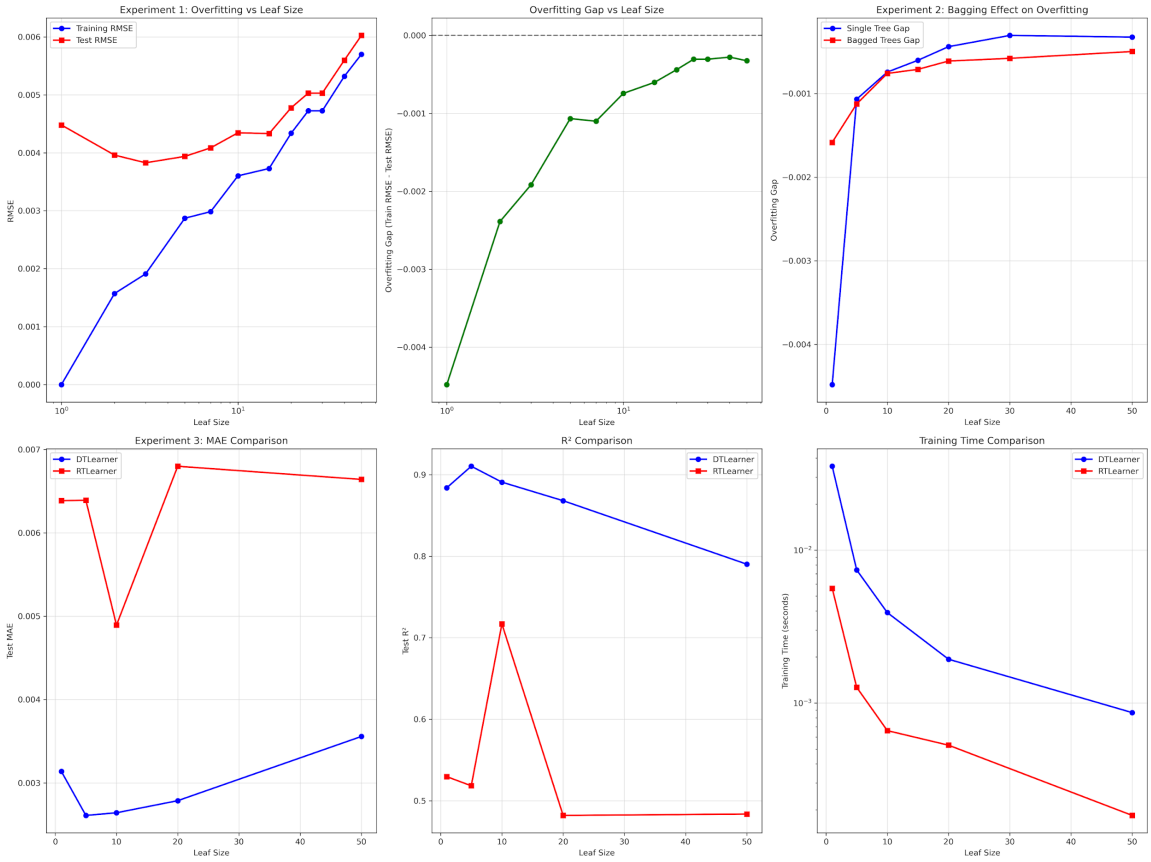
No, the choice depends on the use case:

• DTLearner: Better when accuracy is critical and training time is not constrained

• RTLearner: Better when speed is important or when used in ensemble methods where diversity is valuable

Note: All experimental results are supported by comprehensive charts in p3_report_charts.png, which includes six subplots showing RMSE vs leaf size, overfitting gaps, bagging effects, MAE comparison, R^2 comparison, and training time comparison.

Supporting Charts:



4 CONCLUSIONS

This study revealed several key insights about decision tree learning:

1. Overfitting is strongly correlated with leaf_size: Smaller leaf sizes lead to severe overfitting, with the optimal balance around leaf_size=40.

2. Bagging effectively reduces overfitting: Bootstrap aggregation reduces the training-test gap by 26.3% on average, though it doesn't eliminate overfitting entirely.

3. Trade-offs between deterministic and random trees: DTLearner achieves better accuracy through optimal feature selection, while RTLearner offers faster training and simpler trees through randomization.

4. Hyperparameter tuning is crucial: The choice of leaf_size significantly impacts both overfitting and model performance, requiring careful validation.

Future work could explore adaptive leaf_size selection, more sophisticated ensemble methods, and the interaction between tree depth and leaf_size in overfitting behavior.

5 REFERENCES

1. Quinlan, J. R. (1986). Induction of decision trees. Machine learning, 1(1), 81-106.

2. Breiman, L. (1996). Bagging predictors. Machine learning, 24(2), 123-140.

3. Breiman, L. (2001). Random forests. Machine learning, 45(1), 5-32.