Holly Cohan,Yesh Onipede, Yu Xia

Week 7 Report: Model Training, Hyperparameter Tuning, and Model Evaluation

The objective of this week's assignment is to implement a more complex modeling approach compared to week 6. We applied a regression tree (Decision Tree Regressor) to our dataset and evaluated different hyperparameter settings to improve the model's performance. Decision trees are inherently more complex than linear models, as they can capture non-linear relationships and interactions between features. However, this flexibility comes at a cost: decision trees are prone to overfitting, especially when left unregularized, as they can create intricate structures that fit the training data perfectly but fail to generalize to unseen data.

To address this challenge, we focused on techniques for controlling the complexity of the decision tree. We continued to use standardized data and imputed missing values for the validation and testing datasets to ensure no data leakage. Additionally, we sought to explore how tuning the max_depth, min_samples_split, and min_samples_leaf parameters could help the model better generalize by preventing it from fitting too closely to the training data. The primary goal was to control overfitting and improve the model's generalization performance across the training and validation sets.

Decision Tree Regression: Initial Approach

The first model that we attempted on our data is a basic Decision Tree Regressor without hyperparameter tuning. This initial model showed signs of severe overfitting. Although it achieved near-perfect metrics on the training set as indicated by MSE and RMSE values close to zero and an R^2 of 1.000, the performance on the validation set was much worse with significantly higher MSE and RMSE and only a slight decrease in R^2. This discrepancy suggests that the model was overfitting the training data and failed to generalize well to unseen data. The complexity of the initial tree, with a maximum depth of 52 and over 31,000 leaves, contributed to this poor generalization.

Hyperparameter Tuning

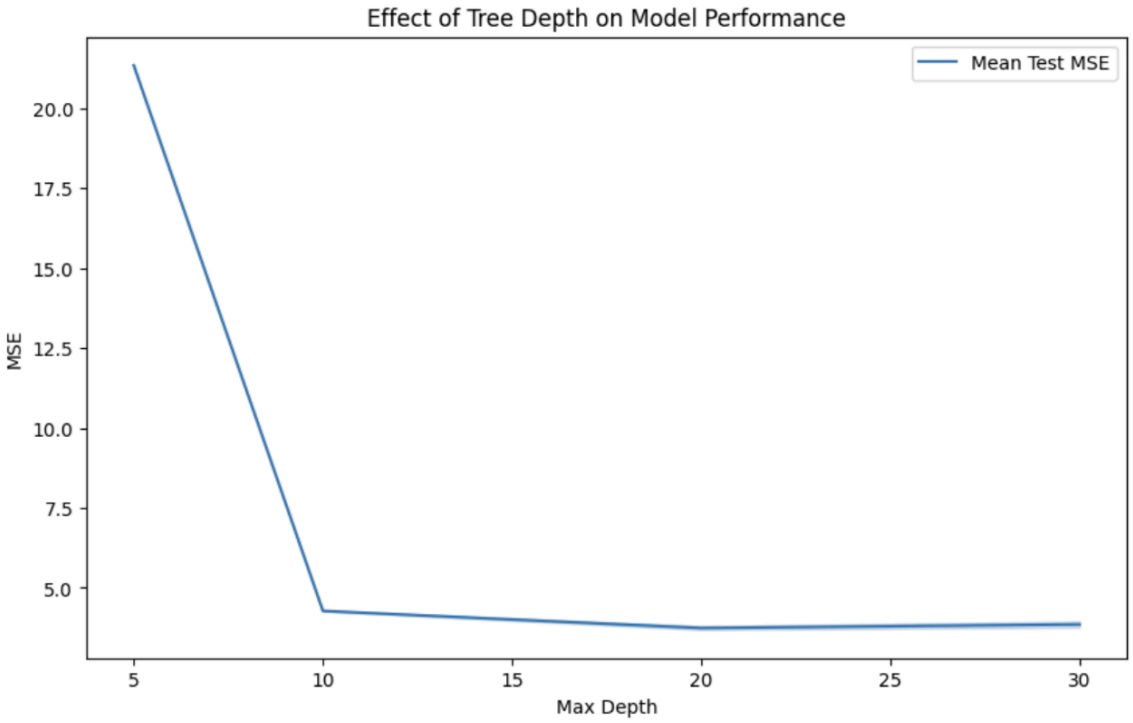
To improve the model's generalization,we performed hyperparameter tuning using a grid search with 2-fold cross-validation. To reduce overfitting, we chose a smaller set of numbers for `max_depth`. The following parameters were explored:

- **max_depth**: [5, 10, 15]
    - The max_depth parameter controls how deep the decision tree is allowed to grow. Setting a maximum depth helps prevent the tree from becoming too complex and overfitting the training data.
    - The chosen values [5, 10, 15] provide a range of tree depths to test the model's performance with both shallow and deeper trees. By evaluating different depths, we aim to find a balance between capturing enough patterns in the data without overfitting.

- **min_samples_split**: [2, 5, 10]
    - The minimum samples split parameter determines the minimum number of samples required to split a node. Setting a higher value can prevent the model from making splits based on small, potentially noisy subsets of the data.
    - By exploring the different values [2, 5, 10], we can see how allowing the tree to grow based on larger splits impacts the model's ability to generalize. Lower values may allow more splits and potentially overfit, while higher values can limit overfitting by requiring larger sample sizes for splits to occur.
- **min_samples_leaf**: [5, 10, 15]
    - The min_samples_leaf parameter specifies the minimum number of samples required to be present at a leaf node. Increasing this value forces the tree to create larger, more generalized leaf nodes rather than splitting until very few samples remain.
    - Evaluation values [5, 10, 15] helps us determine the appropriate level of regularization. A higher value makes the tree more generalized, which helps reduce overfitting. A lower value allows the model to fit more details, which could help capture important patterns but also could risk overfitting.

The best hyperparameters found were:

- max_depth: 15
- min_samples_split: 5
- min_samples_leaf: 10



The optimal depth of 15 found through hyperparameter tuning achieves a balance between model complexity and predictive performance. Going beyond this depth may result in a slightly higher MSE, which highlights the need to control tree growth to avoid overfitting.

Results After Hyperparameter Tuning

The tuned Decision Tree Regressor showed improvement with its prediction on unseen dataset. The results indicate better generalization with reduced overfitting. However, the performance difference between the training and validation metrics suggests that this model is still overfitting slightly.

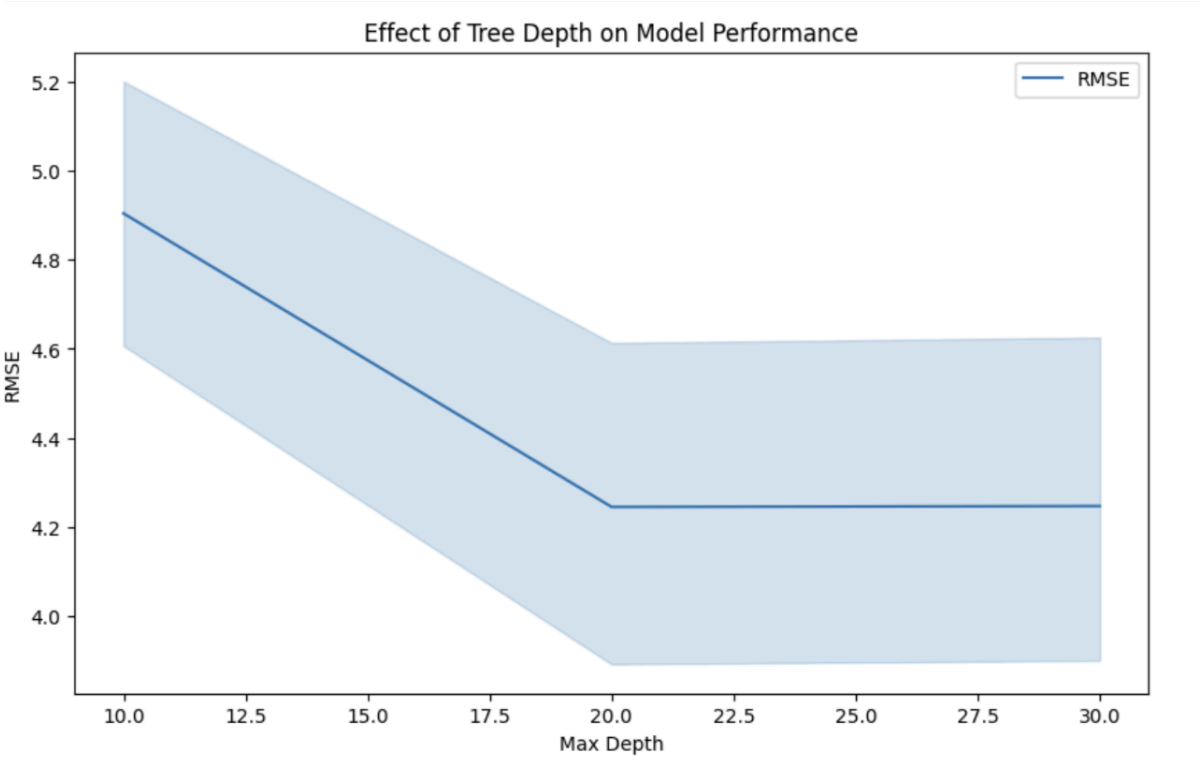
Pruning

To further reduce overfitting, we introduced pruning through the ccp_alpha parameter, which controls the tree's complexity. Now, we have introduced cost complexity parameters to balance the complexity and overfitting. We have chosen a wider range of values for max_depth. The hyperparameter grid was updated to include:

- **max_depth**: [10, 20, 30]
- **min_samples_split**: [2, 5, 7]
- **min_samples_leaf**: [5, 10, 15]
- **ccp_alpha**: [0.001, 0.01, 0.1]

The best hyperparameters for the pruned tree model were:

- ccp_alpha: 0.001
- max_depth: 20
- min_samples_split: 7
- min_samples_leaf: 10



The plot illustrates the impact of different tree depths on the Root Mean Squared Error (RMSE) for the model. The shaded region around the line represents the variability in RMSE across cross-validation folds, providing a measure of uncertainty. As the maximum depth increases, the RMSE generally decreases, indicating improved model performance. The trend flattens after a maximum depth of around 20, suggesting that further increasing the depth brings diminishing improvements.

The pruned model demonstrates improved validation performance while still achieving high training accuracy. Though the R^2 value for this model is the same for the model that did not undergo pruning, the reduced MSE and RMSE values on the validation sets indicate improvement and overall a well-generalized model.

Conclusion

The Decision Tree Regressor, combined with hyperparameter tuning and pruning, successfully addressed the overfitting issue observed in the initial model. The pruned model achieved a strong balance between complexity and generalization, with high R^2 and low MSE across all datasets. The feature importance analysis highlighted the most influential variables, indicating that location-related features played a significant role in predicting the target variable.

Therefore, the best-performing model in this analysis was the pruned Decision Tree Regressor with hyperparameter settings of max_depth: 20, min_samples_split: 7, min_samples_leaf: 10, and ccp_alpha: 0.001. This model achieved an optimal balance between complexity and generalization, with a lower mean squared error (MSE) and root mean squared error (RMSE) on the validation set compared to the unpruned model. While the R^2 values for both the training and validation sets remained high, indicating strong predictive power, the pruning process minimized overfitting by simplifying the tree structure and preventing it from fitting noise in the training data.

Feature Importance Analysis

The top contributing features were identified from the best model:

1. **source_Theatre_District** (Importance: 0.448)
2. **destination_Theatre_District** (Importance: 0.401)
3. **name_Taxi** (Importance: 0.142)
4. **name_Black_SUV** (Importance: 0.002)

Several other features had zero importance, suggesting their irrelevance in predicting the target variable.

Appendix 1.A Metrics Table

Model Type	Metrics	Training Set Values	Validation Set Values
Initial Decision Tree Regression Model	**MSE**	0.1008	5.9150
	RMSE	0.3175	2.4321
	R^2	1.0000	0.9993
Decision Tree Regression Model with Hyperparameter Tuning	**MSE**	2.3140	3.3827
	RMSE	1.5212	1.8392
	R^2	0.9997	0.9996
Pruned Model	**MSE**	2.6747	3.0939
	RMSE	1.6354	1.7589
	R^2	0.9997	0.9996