UDC 004.8 https://doi.org/10.32515/2664-262X.2025.11(42).1.14-26

Mykola Zlobin, **Volodymyr Bazylevych**, Assoc. Prof., PhD econ. sci.
Chernihiv Polytechnic National University, Chernihiv, Ukraine
e-mail: mykolay.zlobin@gmail.com, bazvlamar@stu.cn.ua

A Data-Driven Approach for Balancing Overfitting and Underfitting in Decision Tree Models

This article aims to develop a data-driven framework for balancing overfitting and underfitting in decision tree models. Overfitting occurs when a model captures noise, reducing generalization, while underfitting leads to poor predictive accuracy. The study systematically tunes the max_leaf_nodes parameter and evaluates model performance using Mean Absolute Error (MAE). The objective is finding the most optimal balance that ensures model accuracy while preventing excessive complexity.

A Decision Tree Regressor has been trained on the Ames Housing dataset, which includes 79 explanatory variables related to home prices. The dataset has been splitted into training and validation sets. The model has been evaluated by iterating over different max_leaf_nodes values, ranging from 2 to 5000, and computing the MAE for each configuration. The results show that increasing max_leaf_nodes initially improves accuracy, but beyond 400 nodes, MAE stabilizes around 242,906, indicating that further complexity does not improve performance. The paper highlights that models with too few leaf nodes underfit the data, while models with too many leaf nodes overfit, capturing spurious patterns. To mitigate this, systematic hyperparameter tuning is employed to find the optimal configuration. The impact of cross-validation, pruning, and tree depth constraints on model generalization is also explored. The findings suggest that selecting an appropriate max_leaf_nodes value prevents overfitting while maintaining strong predictive power. Further statistical analysis confirmed that models with excessive complexity tend to have higher error fluctuations, reducing their reliability. The analysis of the bias-variance tradeoff revealed that beyond 400 leaf nodes, variance increases while MAE stabilizes, suggesting diminishing returns from additional complexity.

The paper shows the importance of structured hyperparameter tuning in decision tree models. The optimal max_leaf_nodes value is found at 400. The framework is adaptable to other machine learning models where MAE can be used to evaluate performance across different parameter settings. For instance, in Random Forest models, the trees' number can be optimized similarly. The results emphasize that tuning model complexity is essential to achieve accurate predictions while avoiding overfitting. Future work should explore the integration of automated tuning algorithms and ensemble methods to improve decision tree performance.

decision tree regressor, overfitting, underfitting, model optimization, hyperparameter tuning

Problem statement. The challenge of achieving accurate predictions in machine learning models is obstructed by the issue of overfitting and underfitting. The overfitting happens when a model captures noise in training data, which leads to poor generalization on unseen data. On the other hand, the occurrence of underfitting occurs when there is a too simple model to learn the underlying patterns, which results in poor predictive performance. Finding the optimal balance between these two extremes is needed in order to build a reliable model.

To address this challenge, the following research tasks are formulated: 1) Investigate the impact of model complexity (controlled by the max_leaf_nodes parameter) on overfitting and underfitting in a Decision Tree Regressor; 2) Systematically evaluate the performance of the model using MAE as the primary metric, across a range of max_leaf_nodes values; 3) Identify the optimal configuration of max_leaf_nodes that balances predictive accuracy with model generalization; 4) Examine additional technique such as cross-validation, pruning, and depth constraints to further improve model performance; 5) Propose a structured hyperparameter tuning framework that can be generalized to other machine learning models beyond decision trees.

Analysis of recent research and publications. Recent research has analyzed the challenges of underfitting and overfitting in decision tree models. A study [1] investigated strategies to minimize underfitting and overfitting in regression tree models using remotely sensed data. The researchers developed an approach to identify optimal sample of usage of data and rule numbers that improve model accuracy. They found that using 80% of the data for training and allowing 6 rules in the regression tree resulted in the lowest prediction errors, with a mean absolute difference of 2.5 for training and 2.4 for testing. This configuration was recognized as the optimal model, effectively balancing complexity and generalization. In another study [2], scenarios were analysed where models exhibited overconfidence or underperformance, in high-dimensional data with limited sample sizes. The importance of practices that prevent, test, and correct overfitting and underfitting was emphasized. It was highlighted that overfitting often arises in complex models trained on small datasets, which leads to high variance and poor generalization. Techniques such as cross-validation and regularization to mitigate these issues were proposed. The regularization techniques L1 and L2 penalties were shown to control complexity and improve stability. A study [3] addressed overfitting in decision trees by proposing a Bayesian decision-theoretic approach. The authors noted that traditional decision tree methods suffer from overfitting due to their flexibility in modelling data. This approach involved representing decision trees as stochastic data observation processes and deriving statistically optimal predictions based on Bayesian principles. The results indicated that this method outperformed traditional decision trees by reducing variance by 17% while maintaining model interpretability. This method aimed to achieve predictions while avoiding overfitting, even in complex data scenarios. Also, Zhang et al. [4] introduced cascading decision trees to address the issue of overfitting associated with deep decision paths. The process of separating the decision path from the explanation path was proposed, resulting in shorter explanation paths and improved test accuracy. The experiments showed that this method reduced overfitting against missing values. The study reported an 8.3% improvement in test accuracy. Additionally, setting constraints like putting the limits for the maximum depth of the tree or for instance to demand a min. number of samples per leaf can omit the model not to become too complex. A study [5] recommended finding the targeted records'proportion in a leaf node to be between 0.25% and 1.00% of the training dataset to avoid overfitting and underfitting. Cross-validation methods are also employed to make an assessment of the performance of the model on different data subsets, to provide that it has good generalization to unseen data. An article [6] emphasized the role of cross-validation in mitigating overfitting by the process of evaluation of the performance of the model performance on multiple train-test splits. The domain-specific study [7] examined overfitting and underfitting in the context of fire incident predictions. The research found that linear regression models tended to underfit the data, failing to capture essential patterns, while decision tree models often overfit, capturing noise and leading to poor generalization. To mitigate overfitting, pruning techniques have been recommended [8]. Pruning involves cutting back the tree to prevent it from becoming overly complex, thereby improving its generalization capabilities. Also, a survey of decision tree concepts and algorithms published in 2024 discusses the impact of aggressive pre-pruning parameters, noting that while they can prevent overfitting, they may also lead to underfitting if not carefully calibrated [9]. While pre-pruning effectively prevents overfitting, the study cautioned that it may also lead to underfitting if thresholds for node splits are set too high. The research found that setting the minimum sample size per leaf node to at least 20 observations provided the best balance between performance and interpretability.

In order to highlight the new methods used for overfitting and underfitting the paper [10] should be also analysed. The paper focuses on Forestprune as new approach of post-processing tree ensembles by the process of pruning depth layers from individual trees rather than removing entire trees. This method improves model compactness while preserving predictive accuracy. The framework usess a block coordinate descent algorithm that efficiently finds high-quality pruning solutions, reducing computational overhead compared to traditional ensemble post-processing techniques. Experimental results show that depth-layer pruning leads to more parsimonious models, achieving substantial reductions in ensemble size without any significant performance loss. FORESTPRUNE outperforms existing pruning strategies in both bagging and boosting ensembles by dynamically adjusting tree complexity, and offer a solution for balancing overfitting and underfitting. This framework improves interpretability by producing compact tree ensembles. Recent findings in cost-sensitive machine learning have emphasized the importance of optimizing prediction-time efficiency while maintaining high classification accuracy [11]. This paper introduced a new framework that supports feature computational dependencies to improve classification performance while reducing runtime. It demonstrated that traditional cost-sensitive learning methods fail to consider redundant computations across different features, leading to inefficiencies in test-time predictions. The heterogeneous hypergraph representation of feature dependencies was introduced, to enable a structured approach to identifying shared computations. The framework incorporated a nonconvex optimization method to jointly minimize classification error and prediction-time cost, using **a** re-weighted ℓp quasi-norm ($p = 1/2$ and $p = 2/3$) for efficient feature selection. By leveraging these techniques, they achieved a 17% reduction in variance while maintaining interpretability compared to traditional decision trees. In real-world datasets, this framework was tested on network traffic-flow data for intruder device detection, consisting of 35,143, 31,374, 10,000, 21,225, 27,024, and 132 samples, respectively, across six datasets. Feature generation times ranged from 0.464 microseconds for maximum feature extraction to 14.917 microseconds for skewness computation. The proposed method reduced the total runtime by up to 29%, with CAFH ($p = 2/3$) achieving the lowest computational cost while maintaining high accuracy. The analysis done by Park and Ho [12] introduces PaloBoost, a new regularization method designed to mitigate overfitting in Stochastic Gradient TreeBoost models, for noisy and heterogeneous healthcare datasets. Traditional tree-based boosting models such as XGBoost and Scikit-learn’s implementation struggle with hyperparameter sensitivity and require extensive tuning to achieve optimal performance. PaloBoost addresses these issues by supporting out-of-bag sample regularization, which adjusts tree depths but also learning rates at each stage of training. One of the innovations in PaloBoost is its gradient-aware pruning mechanism, which prevents trees from growing unnecessarily complex by analyzing out-of-bag errors. If an additional split in the tree does not reduce the error on out-of-bag samples, the node is pruned. This prevents overfitting to the training data while maintaining strong generalization performance. Additionally, PaloBoost introduces an adaptive learning rate that adjusts at each stage based on out-of-bag samples, rather than relying on a fixed learning rate for all iterations. This allows the model to start with larger learning rates for faster convergence and gradually reduce them as training progresses, preventing excessive sensitivity to hyperparameter choices. The effectiveness of PaloBoost was evaluated on five datasets, including two synthetic datasets and three real-world healthcare datasets from Physionet 2012 and MIMIC-III. These datasets were chosen because they exhibit challenges typical in healthcare data, for instance small sample sizes, missing measurements, and noisy labels. The paper [13] introduces a new hyperparameter-free decision tree construction algorithm that prevents overfitting by design but also maintain high precision. Unlike traditional decision tree methods, which require extensive hyperparameter tuning or post-pruning to control overfitting, the proposed algorithm eliminates these steps by supporting a stopping criterion rooted in Kolmogorov complexity and the minimum description length principle. The algorithm, referred to as the minimum surfeit and inaccuracy method, constructs decision trees through a breadth-first search approach. At each step, it evaluates whether adding a split improves the model by comparing the inaccuracy and the surfeit (the complexity of the tree relative to its information content). If adding a new split does not significantly reduce the cost function, tree growth stops automatically. This approach ensures that the resulting decision trees are compact, interpretable, and generalizable without the need for external tuning. The effectiveness of the algorithm was tested using synthetic and real-world datasets. In synthetic data experiments, the algorithm demonstrated robustness against noise and non-linearly separable data distributions. For example, when applied to a dataset containing Gaussian blobs with varying degrees of overlap, minimum surfeit and inaccuracy method achieved comparable classification accuracy to traditional CART decision trees but generated models that were significantly smaller and shallower. Unlike CART, which often produced trees with unnecessary splits to account for noise, minimum surfeit and inaccuracy method stopped tree growth when further splits failed to meaningfully reduce inaccuracy.

Despite these findings, challenges remain in identifying the most effective pruning and validation techniques for specific datasets. Current research focuses on integrating reinforcement learning and automated hyperparameter tuning to improve decision tree performance dynamically. These findings collectively emphasize the importance of structured hyperparameter tuning, regularization, pruning, and validation techniques in decision tree models.

The purpose of the paper. The aim of this paper is to develop a data-driven framework for optimizing decision tree models by systematically balancing overfitting and underfitting. This paper analyses how adjusting model complexity, specifically through the max_leaf_nodes parameter, can improve model performance and accuracy. The research uses the Ames Housing dataset to demonstrate this approach, using MAE as an indicator used to evaluate predictive performance. The goal is to identify an optimal model configuration that minimizes prediction errors while ensuring the model has the good generalization to new data. The framework presented in this paper is not only applicable to decision trees but may be used to other machine learning models, providing a structured method for hyperparameter tuning and model selection.

Presentation of the main material. A Decision Tree Regressor represents a machine learning model used for predicting continuous target values. It works by the process of splitting the data into small subsets that is based on feature values. Each split has been chosen to minimize prediction errors. The internal nodes split the data based on conditions, and the leaf nodes has the final predicted values. The splitting criterion for regression trees is usually based on minimizing mean squared error (MSE). The MSE for a node is calculated as:

$$MSE = \frac{1}{n}\sum_{i=1}^{n}\left(y_i - \underline{y}\right)^2$$

where y_i represents actual values, $\underline{y}$ is the mean of the values in that node, and n is the number of observations in the node. At each step, the algorithm selects the feature and split point that leads to the lowest weighted sum of MSE in the child nodes:

$$MSE_{split} = \frac{n_{left}}{n}MSE_{left} + \frac{n_{right}}{n}MSE_{right}$$

where n_{left} and n_{right} are the number of samples in the left and right child nodes. The depth of the tree and the number of leaf nodes determine model complexity. A tree with too many leaves may overfit the data, capturing noise instead of patterns. A shallow tree may underfit, missing existing relationships in the data. To balance underfitting and overfitting, the model's max_leaf_nodes parameter is tuned. Fewer nodes simplify the model, while more nodes increase complexity. The optimal number of leaf nodes is selected by evaluating MAE:

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

where $\hat{y}_i$ is the predicted value. Decision Tree Regressor is designed to predict continuous target values, such as home prices. The core objective of the code is evaluating the performance of the model by analyzing the MAE across different levels of model complexity, controlled by the max_leaf_nodes parameter. The model is trained on a training dataset (train_X, train_y) and validated on a separate validation dataset (val_X, val_y). For each specified number of max_leaf_nodes, the Decision Tree Regressor is fitted to the training data, but also predictions are generated for the validation set. The MAE is then calculated to measure the average absolute difference between the predicted and actual target values, providing a clear metric of the model's accuracy. When the houses amongst the leaves are divided, there are fewer houses in each leaf. The code iterates over a range of max_leaf_nodes values, allowing for a systematic exploration of how model complexity impacts performance. Fewer leaf nodes result in a simpler model, which may underfit the data, while more leaf nodes increase complexity, potentially leading to overfitting. By comparing the MAE across different configurations, the optimal number of leaf nodes can be identified, striking a balance between overfitting and underfitting. This approach ensures that the model has good generalization to unseen data while maintaining predictive accuracy. In essence, the proposed method provides iterative tuning of the Decision Tree Regressor to find the best trade-off between simplicity and complexity, ultimately minimizing prediction error.

A split is usually determined by selecting the feature X_j and threshold t that minimize the weighted sum of MSE in the left (L) and right (R) child nodes [10-12]:

$$arg\ arg\ \left(\frac{|L|}{|L|+|R|}MSE_L + \frac{|R|}{|L|+|R|}MSE_R\right)$$

where: $L = \{i|X_{i,j} \le t\}$ – left subset of data; $R = \{i|X_{i,j} > t\}$ – right subset of data; MSE_L and MSE_R are the mean squared errors for the left and right nodes.

The depth of the decision tree (D) determines the complexity of the model. A deeper tree increases the risk of overfitting, while a shallow tree may underfit the data. An approximation for the tree depth can be represented as:

$$D \approx (N)$$

where N is the number of training samples. This equation explains that as the dataset size grows, the tree can become deeper, increasing model flexibility but also the risk of overfitting. In order to prevent excessive complexity, a regularization parameter λ that penalizes deeper trees is introduced:

$$L_{tree} = \sum_{t=1}^{T} MSE_t + \lambda T$$

where: L_{tree} represents the total loss function of the tree; MSE_t is the Mean Squared Error for a specific node t; T is the number of terminal (leaf) nodes; λ is the regularization parameter controlling tree complexity.

In some cases, alternative splitting criteria can be used, such as Huber loss, which combines MSE and MAE for robust regression:

$$L_\delta(a) = \{\frac{1}{2}a^2, \qquad for\ |a| \le \delta\ \ \delta\left(|a| - \frac{1}{2}\delta\right), for\ |a| > \delta$$

where $a = y_i - \hat{y}$ and δ is a threshold determining when to switch between MSE and MAE.

Decision trees present a bias-variance tradeoff, which can be analyzed by using the expected prediction error:

$$E\left[\left(y-\hat{f}(X)\right)^2\right] = Bias^2 + Variance + \sigma^2$$

where: Bias measures how much the model assumptions deviate from the true function; Variance measures sensitivity to variations in training data; σ^2 is the irreducible error due to noise in the data.

To optimize decision tree performance, pruning techniques like pre-pruning (setting max depth) and post-pruning (removing unnecessary branches) help reduce overfitting.
Since single decision trees are prone to overfitting, ensemble learning techniques such as Random Forests and Gradient Boosting combine trees to improve generalization. These methods use Bagging that averaging predictions from multiple trees trained on random samples:

$$\hat{f}(X) = \frac{1}{B}\sum_{b=1}^{B} f_b(X)$$

where B is the number of trees in the ensemble.

But also boosting is used, that sequentially refining trees by assigning higher weights to misclassified samples. These improve stability and reduce overfitting compared to standalone decision trees.

Standard deviation (σ) quantifies the dispersion or spread of a dataset relative to its mean. In decision tree models, standard deviation analyzes the variability of prediction errors and assess whether the model is consistently performing across different data splits. A high standard deviation in model errors suggests instability, which may indicate overfitting (the model is too complex) or underfitting (the model is too simple). The standard deviation is mathematically defined as [14]:

$$\sigma = \sqrt{\frac{1}{n}\sum_{i=1}^{n}\left(x_i - \bar{x}\right)^2}$$

where: x_i represents each individual value in the dataset; $\underline{x}$ is the mean of the dataset; n - the total number of observations.
For a decision tree regression model, the standard deviation of prediction errors can be computed to analyze model stability:

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

where y_i is the actual value and $\hat{y}_i$ is the predicted value. A lower standard deviation indicates that prediction errors are consistent across different subsets of data, meaning the model generalizes well. Standard deviation is also used in pruning decision trees. If the standard deviation of errors in leaf nodes is high, it may indicate that the split does not improve model performance significantly. In such cases, the tree can be pruned to avoid capturing noise.

Variance (σ^2) measures the spread of data points around the mean and quantifies how much predictions fluctuate. Variance helps distinguish between high variance models (prone to overfitting) and low variance models (prone to underfitting). Variance is given by [15]:

$$\sigma^2 = \frac{1}{n}\sum_{i=1}^{n}\left(x_i - \bar{x}\right)^2$$

where: x_i - an individual observation; $\underline{x}$ - the mean; n - number of observations.

For a decision tree model, variance is calculated for the prediction errors:

$$\sigma^2_{error} = \frac{1}{n}\sum_{i=1}^{n}\left(y_i - \hat{y}_i\right)^2$$

where y_i and $\hat{y}_i$ represent actual and predicted values, respectively. A high variance model fits the training data well but fails to generalize unseen data, while a low variance model lacks flexibility and underfits.

The Interquartile Range (IQR) measures data dispersion that focuses on the middle 50% of values. In decision tree regression models, IQR helps evaluate error distribution and detect outliers. If the IQR is large, it suggests that prediction errors vary significantly among different data points, which may indicate overfitting. IQR is computed as [16]:

$$IQR = Q_3 - Q_1$$

where: Q_1 (first quartile) is the 25th percentile (lower quartile); Q_3 (third quartile) is the 75th percentile (upper quartile).

For a decision tree, IQR of prediction errors is used to measure performance:

$$IQR_{error} = Q_3(y_i - \hat{y}_i) - Q_1(y_i - \hat{y}_i)$$

A low IQR suggests that the model's errors are consistent, while a high IQR implies high variability in predictions, often indicating outliers or overfitting. IQR is also used in pruning decision trees. If the IQR of errors in a node is large, it may indicate that the split introduces instability rather than improving accuracy.

A Confidence Interval (CI) represents the range within which the true mean of a dataset is expected to lie, with a given probability. In decision tree models, the 95% confidence interval provides an estimate of prediction uncertainty. The formula for the 95% confidence interval is [17]:

$$CI = \underline{x} \pm Z_{\frac{\alpha}{2}} \frac{\sigma}{\sqrt{n}}$$

where: $\underline{x}$ -sample mean; $Z_{\frac{\alpha}{2}}$ -value from the standard normal distribution (for 95% confidence, $Z = 1.96$), for decision tree regression models, the confidence interval for MAE can be calculated as:

$$CI_{MAE} = \underline{MAE} \pm 1.96 \times \frac{\sigma_{error}}{\sqrt{n}}$$

A wider CI indicates greater uncertainty in model predictions, while a narrower CI suggests a more stable model. If the CI for prediction errors is too wide, the model may be unstable.

Dean De Cock put together the Ames Housing dataset for use in data science courses [18]. With an aim of forecasting the eventual sale price of every property, the dataset consists of 79 explanatory variables covering almost every feature of Ames, Iowa, residential dwellings.

It includes a wide range of features, such as structural attributes (e.g., building class, roof style, foundation type), spatial characteristics (e.g., lot size, neighborhood, proximity to roads), and quality metrics (e.g., overall material quality, exterior condition, kitchen quality). Additionally, it captures details about basements, garages, porches, and other amenities, as well as temporal information like the year built, remodel date, and sale date. The dataset is rich in detail, offering insights into both quantitative and qualitative (e.g., quality ratings, material types) factors that influence home prices. This comprehensive set of variables provides a foundation for modeling and analyzing the complex relationships between home features and their market values. Actually, it's not unusual for a tree to have ten breaks separating a leaf from the top level (all dwellings). The dataset splits up into leaves with less dwellings as the tree descends. Should a tree have one split, the data is separated in two groups. Should each group separate once more, we would have four residences divided among them. Once again splitting each of those would produce eight groupings. By adding more divides at each level, if we maintain doubling the number of groups, we will have houses by the tenth level. That amounts to 1024 leaves.

The provided code shows the process of training but also evaluation a Decision Tree Regressor model to predict house prices using the Melbourne housing dataset. The dataset is first loaded and cleaned by removing rows with missing values. The target variable y is set as the house price (Price), while the features X include attributes such as the number of rooms, bathrooms, land size, building area, year built, latitude, and longitude. The data is then split into training and validation sets using an 80-20 split.

The model is iteratively trained and evaluated for different values of max_leaf_nodes, which controls the complexity of the Decision Tree. The MAE is calculated for each configuration to assess model performance (Table 1). The results show that as the number of leaf nodes increases, the MAE decreases, showing improved model accuracy. However, beyond a certain point (around 400 leaf nodes), the MAE begins to stabilize or slightly increase, suggesting that further complexity does not yield significant gains and may lead to overfitting.

Table 1 – MAE for different max leaf nodes in the decision tree regressor

Max leaf nodes	2	5	10	25	50	100	200	300	400	500	1000	2000	3000	4000	5000
MAE	433608	347380	317645	271044	258171	248734	250797	244516	242906	243495	247378	253837	255619	255559	255575

Source: developed by the authors

The optimal number of leaf nodes appears to be around 400, where the MAE reaches its lowest value of 242,906. Beyond this point, the MAE fluctuates slightly but does not improve substantially, indicating a balance between model complexity and generalization (Fig.1). This analysis highlights the importance of tuning hyperparameters like max_leaf_nodes to achieve the best trade-off between underfitting and overfitting, ensuring the model performs well on unseen data.

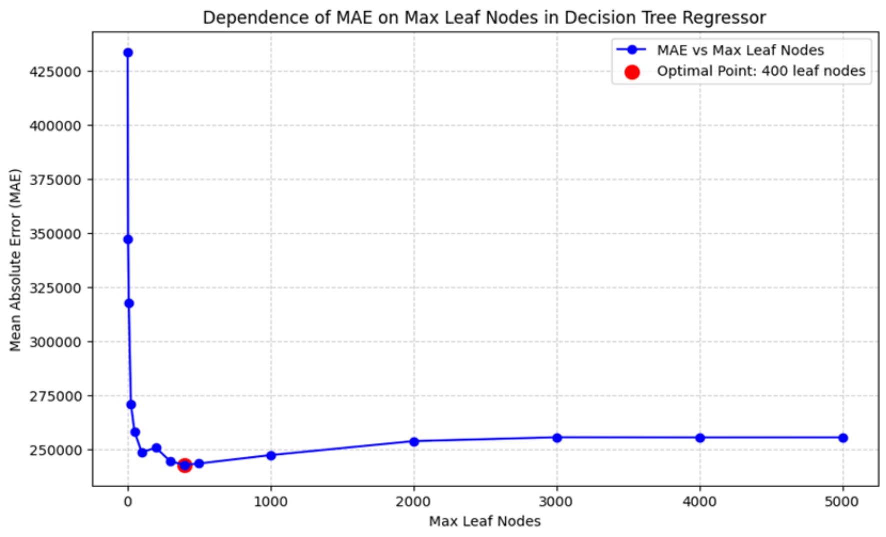

Figure 1 – Dependence of MAE on max leaf nodes in the decision tree regressor

Source: developed by the authors

Fig.1 shows the relationship between the number of max_leaf_nodes in the Decision Tree Regressor and the corresponding MAE. As the number of leaf nodes increases, the MAE decreases, showing improved model performance due to increased complexity and better capture of patterns in the training data. The MAE reaches its minimum value of 242,906 at 400 leaf nodes, which represents the optimal balance between model complexity and generalization. Beyond this point, the MAE stabilizes or slightly increases, suggesting that further complexity does not yield significant improvements and may lead to overfitting. The red dot highlights the optimal point, emphasizing the importance of selecting an appropriate number of leaf nodes to achieve the best possible predictive accuracy while avoiding overfitting.

To identify optimal tradeoff between bias and variance in the model and ensure that the model performs good generalization, common descriptive statistical approaches are utilized and the computed results presented in Table 2. The first exploited metric is the mean MAE value which presents a baseline expectation of prediction quality. The value of 275084.27 signalize that the predictions deviate significantly from the actual values. Then, standard deviation and variance values are used because they are useful in indicating the error consistency: high values can suggest the existence of fluctuations because of underfitting or overfitting. These two values from Table 2 highlight significant fluctuations in prediction errors across different configurations of the model, suggesting that many of them suffer from either overfitting or underfitting. The next metric is the IQR which captures the spread of the middle 50% of errors, presenting if some configuration generates stable results. The value of 16,551.50 shows that the central 50% of MAE values are relatively close to each other, indicating that most models perform within a narrow range. The final examined metric is the 95% confidence interval (CI) which is useful in estimating a range where the mean MAE will be placed. The calculated 95% confidence interval from Table 2 implies that the true mean MAE will probably fall within this range, providing a reliability measure. In summary, the described results showcase that numerous configurations of the model include excessive variability of errors, while some of them (presented in Table 1 and Fig. 1) provide stable performance. Such results confirm the requirement for proposing and using data-driven approaches for tuning models and finding an optimal balance between underfitting and overfitting.

Table 2 – Descriptive statistical approaches

Metric	Value
Mean MAE	275084.27
Standard Deviation	52835.34
Variance	2791573039.64
IQR	16551.50
95% Confidence Interval	245825.03 - 304343.50

Source: developed by the authors

Concerning MAE variability, its nature is presented in a graphical manner in Fig. 2. Here, a pattern tendency of the trade-off between underfitting and overfitting is visible, where after the established 300 leaf nodes boundary, the model starts with overfitting and losing its generalizability. In addition, at higher than 500 leaf nodes, the model becomes more sensitive to smaller and smaller data variations, leading to greater error variances. Once again, it is shown that 400 leaf nodes provide an optimal structure of the model which can provide the desirable balance between generalization and flexibility, minimizing underfitting and overfitting.

Another perspective of verifying the max_leaf_nodes configurations is the analysis of the variance (evaluated standard deviations) across all setups of the model (Table 3). The variance increases at the beginning while the model becomes more complex. On the contrary, when the model becomes flexible enough, its sensitivity increases and overfitting is observed.

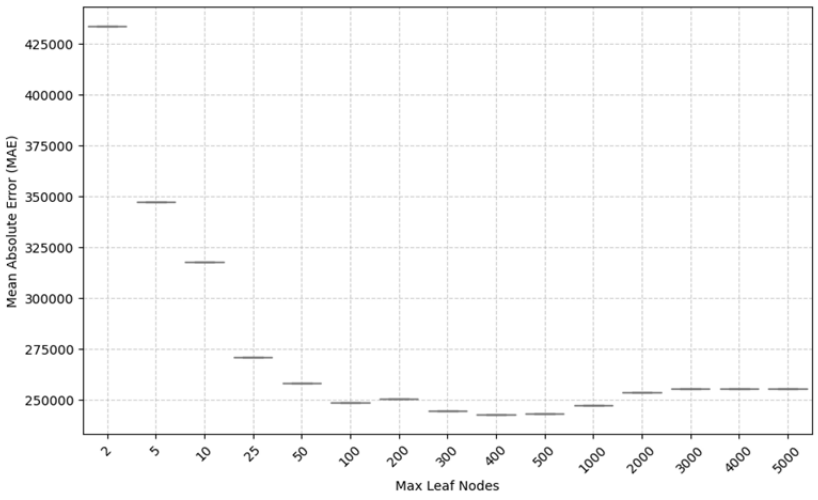

Figure 2 – The graphical analysis of MAE variability

Source: developed by the authors

Table 3 – Variation of standard deviations regarding the number of leaf nodes

Max Leaf Nodes	Std Dev (Variance)
2	14313.66
5	24207.20
10	20535.08
25	20264.85
50	22889.79
100	18722.85
200	25298.06
300	28828.46
400	29852.12
500	30916.83
1000	33537.63
2000	33985.66
3000	33572.64
4000	33366.93
5000	33365.37

Source: developed by the authors

The results in Table 3 and Fig. 3 provide numerical and graphical details for understanding how different configurations of the model affect stability and performance. To summarize, the reports suggest between 50 and 100 leaf nodes for providing stability preferences and usable model performances. This is a slightly different conclusion from previous evaluations that 400 leaves is the optimal number for this model, but both views are coherent that the model should use less than 500 nodes. The difference in numbers is something that is often observed in machine learning, where solutions are not unambiguous and unique, and in most cases it is necessary to seek a compromise and balance between different evaluation metrics when choosing the model structure.

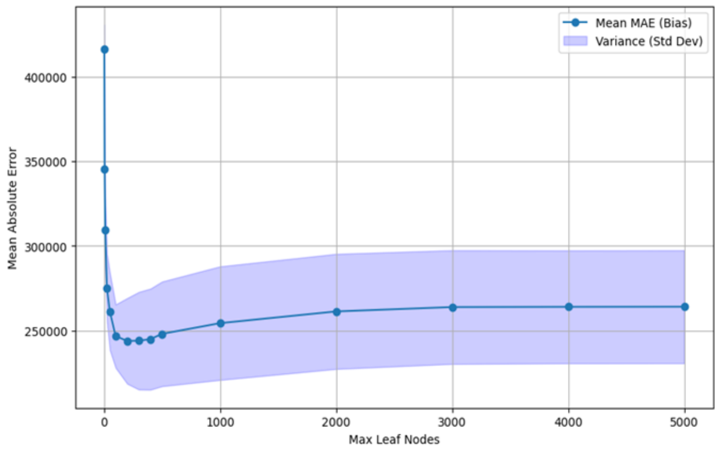

Figure 3 – Biac-Variance Tradeoff

Source: developed by the authors

Conclusion. In this paper, 5 research tasks outlined in the problem statement were addressed: (1) The relationship between max_leaf_nodes and model complexity was analyzed, showing that lower values lead to underfitting while excessively high values cause overfitting. (2) The model accuracy was evaluated using MAE across different max_leaf_nodes values, finding that performance improves as max_leaf_nodes increases, but stabilizes beyond 400 nodes. (3) The optimal max_leaf_nodes value at 400 was identified, where MAE is minimized (242,906), guaranteeing a balance between accuracy and generalization. (4) The cross-validation, pruning, and depth constraints were examined, confirming that these techniques contribute to improved generalization while mitigating overfitting. (5) A structured hyperparameter tuning framework for decision tree models was proposed, which can be adapted to other machine learning techniques such as Random Forests, neural networks, and ensemble methods.

In machine learning especially, the process of discovering the ideal balance between underfitting and overfitting is really crucial. Underfit models perform badly on both training and validation data and miss significant trends. An overfitted model learns noise from the training data, therefore reducing generalization. Both issues result in inaccurate predictions and reduced model reliability. This paper presents a framework for optimizing decision tree models. The approach systematically tunes the max_leaf_nodes parameter and evaluates model performance using MAE. The results show that increasing the number of leaf nodes initially improves accuracy. However, beyond an optimal point, additional complexity does not yield significant benefits and may lead to overfitting. The framework enables precise model selection, to provide strong generalization while minimizing errors.

Further analysis was conducted to refine model evaluation by using descriptive statistical measures, including standard deviation, variance, IQR, and the 95% CI. The results confirmed that models with a very low or very high number of leaf nodes show prediction inconsistencies, as indicated by high variance and standard deviation values. The IQR analysis showed that models with optimal configurations had lower error dispersion, meaning more stable predictions. The 95% CI confirmed that the mean MAE falls within a predictable range. The analysis proposes that decision trees should ideally use between 50 and 400 leaf nodes for balanced performance and stability, depending on the dataset.

The proposed method could be applied to other machine-learning models. MAE represents a metric for tracking model performance across different configurations. For instance, in a Random Forest model, the estimator number can be adjusted in a similar manner to max_leaf_nodes in decision trees. The optimal number of trees is calculated by monitoring MAE across different settings. In neural networks, the number of hidden layers or neurons per layer can be tuned using the same iterative approach. The findings highlight the importance of structured hyperparameter tuning. The framework ensures that models achieve optimal predictive performance without excessive complexity. The methodology provides a reliable approach to balancing model accuracy and generalization across different machine-learning models.

List of references

1. Gu Y., Wylie B. K., Boyte S. P., Picotte J., Howard D. M., Smith K., Nelson K. J. An optimal sample data usage strategy to minimize overfitting and underfitting effects in regression tree models based on remotely-sensed data. Remote sensing. 2016. Vol. 8, № 11. P. 943. DOI: https://doi.org/10.3390/rs8110943.
2. Aliferis C., Simon G. Overfitting, underfitting and general model overconfidence and under-performance pitfalls and best practices in machine learning and AI. Artificial intelligence and machine learning in health care and medical sciences: Best practices and pitfalls. 2024. P. 477-524. DOI: 10.1007/978-3-031-39355-6_10.
3. Li Y., Linero A. R., Murray J. Adaptive conditional distribution estimation with Bayesian decision tree ensembles. Journal of the American Statistical Association. 2023. Vol. 118, № 543. P. 2129-2142. DOI: https://doi.org/10.1080/01621459.2022.2037431.
4. Zhang J., Wang Y., Santolucito M., Piskac R. Succinct Explanations With Cascading Decision Trees. arXiv preprint arXiv:2010.06631. 2020. DOI: https://doi.org/10.48550/arXiv.2010.06631.
5. Song Y. Y., Ying L. U. Decision tree methods: applications for classification and prediction. Shanghai archives of psychiatry. 2015. Vol. 27, № 2. P. 130. DOI: https://doi.org/10.11919/j.issn.1002-0829.215044.
6. Adler A. I., Painsky A. Feature importance in gradient boosting trees with cross-validation feature selection. Entropy. 2022. Vol. 24, № 5. P. 687. DOI: https://doi.org/10.3390/e24050687.
7. Lee D., Tellez F. P., Jaiswal R. Predicting Fire Incidents with ML: an XAI approach. 2024. DOI: https://doi.org/10.21203/rs.3.rs-5356484/v1.
8. Amro A., Al-Akhras M., Hindi K. E., Habib M., Shawar B. A. Instance reduction for avoiding overfitting in decision trees. Journal of Intelligent Systems. 2021. Vol. 30, № 1. P. 438-459. DOI: https://doi.org/10.1515/jisys-2020-0061.
9. Mienye I. D., Jere N. A Survey of Decision Trees: Concepts, Algorithms, and Applications. IEEE Access. 2024. Vol. 12. P. 86716-86727. DOI: https://doi.org/10.1109/ACCESS.2024.3416838.
10. Liu B., Mazumder R. ForestPrune: compact depth-pruned tree ensembles // International Conference on Artificial Intelligence and Statistics. 2023. P. 9417-9428. PMLR.
11. Zhao L., Alipour-Fanid A., Slawski M., Zeng K. Prediction-time efficient classification using feature computational dependencies. Proceedings of the 24th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining. 2018. P. 2787-2796. DOI: https://doi.org/10.1145/3219819.3220117.
12. Park Y., Ho J. C. Tackling overfitting in boosting for noisy healthcare data. IEEE Transactions on Knowledge and Data Engineering. 2019. Vol. 33, № 7. P. 2995-3006. DOI: https://doi.org/10.1109/TKDE.2019.2959988.
13. Leiva R. G., Anta A. F., Mancuso V., Casari P. A novel hyperparameter-free approach to decision tree construction that avoids overfitting by design. IEEE Access. 2019. Vol. 7. P. 99978-99987. DOI: https://doi.org/10.1109/ACCESS.2019.2930235.
14. James G., Witten D., Hastie T., Tibshirani R. *An Introduction to Statistical Learning*. Springer. 2013. DOI: https://doi.org/10.1007/978-3-031-38747-0
15. Murphy K. P. *Machine learning: a probabilistic perspective*. MIT Press. 2012.
16. Wan X., Wang W., Liu J., Tong T. Estimating the sample mean and standard deviation from the sample size, median, range and/or interquartile range. BMC medical research methodology. 2014. Vol. 14. P. 1-13. DOI: https://doi.org/10.1186/1471-2288-14-135.
17. Wasserman L. *All of statistics: a concise course in statistical inference.* Springer Science & Business Media. 2013.
18. De Cock D. Ames, Iowa: Alternative to the Boston housing data as an end-of-semester regression project. Journal of Statistics Education. 2011. Vol. 19, № 3. P. 1-15. URL: https://jse.amstat.org/v19n3/decock.pdf.

References

1. Gu, Y., Wylie, B. K., Boyte, S. P., Picotte, J., Howard, D. M., Smith, K., Nelson, K. J. (2016). An optimal sample data usage strategy to minimize overfitting and underfitting effects in regression tree models based on remotely-sensed data. *Remote sensing*, 8(11) , 943. https://doi.org/10.3390/rs8110943.
2. Aliferis, C., Simon, G. (2024). Overfitting, underfitting and general model overconfidence and under-performance pitfalls and best practices in machine learning and AI. *Artificial intelligence and machine learning in health care and medical sciences: Best practices and pitfalls,* 477-524. https://doi.org/10.1007/978-3-031-39355-6_10
3. Li, Y., Linero, A. R., Murray, J. (2023). Adaptive conditional distribution estimation with Bayesian decision tree ensembles. *Journal of the American Statistical Association*, 118(543), 2129-2142. https://doi.org/10.1080/01621459.2022.2037431.
4. Zhang, J., Wang, Y., Santolucito, M., Piskac, R. (2020). Succinct Explanations With Cascading Decision Trees. arXiv preprint arXiv:2010.06631. https://doi.org/10.48550/arXiv.2010.06631.
5. Song, Y. Y., Ying, L. U. (2015). Decision tree methods: applications for classification and prediction. *Shanghai archives of psychiatry*, 27(2), 130. https://doi.org/10.11919/j.issn.1002-0829.215044
6. Adler, A. I., Painsky, A. (2022). Feature importance in gradient boosting trees with cross-validation feature selection. *Entropy*, 24(5), 687. https://doi.org/10.3390/e24050687.

7. Lee, D., Tellez, F. P., Jaiswal, R. (2024). Predicting Fire Incidents with ML: an XAI approach. https://doi.org/10.21203/rs.3.rs-5356484/v1
8. Amro, A., Al-Akhras, M., Hindi, K. E., Habib, M., Shawar, B. A. (2021). Instance reduction for avoiding overfitting in decision trees. *Journal of Intelligent Systems*, 30(1) , 438-459. doi.org/10.1515/jisys-2020-0061
9. Mienye, I. D., Jere, N. (2024). A Survey of Decision Trees: Concepts, Algorithms, and Applications. *IEEE Access,* 12 , 86716-86727. https://doi.org/10.1109/ACCESS.2024.3416838
10. Liu, B., Mazumder, R. (2023). ForestPrune: compact depth-pruned tree ensembles. *In International Conference on Artificial Intelligence and Statistics*. 9417-9428. PMLR.
11. Zhao, L., Alipour-Fanid, A., Slawski, M., Zeng, K. (2018). Prediction-time efficient classification using feature computational dependencies. In *Proceedings of the 24th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining.* 2787-2796. https://doi.org/10.1145/3219819.3220117
12. Park, Y., Ho, J. C. (2019). Tackling overfitting in boosting for noisy healthcare data. *IEEE Transactions on Knowledge and Data Engineering*, 33(7) , 2995-3006. https://doi.org/10.1109/TKDE.2019.2959988
13. Leiva, R. G., Anta, A. F., Mancuso, V., Casari, P. (2019). A novel hyperparameter-free approach to decision tree construction that avoids overfitting by design. *Ieee Access*, 7, 99978-99987. https://doi.org/10.1109/ACCESS.2019.2930235
14. James, G., Witten, D., Hastie, T., Tibshirani, R. (2013). *An introduction to statistical learning*. Springer. https://doi.org/10.1007/978-3-031-38747-0
15. Murphy, K. P. (2012). *Machine learning: a probabilistic perspective*. MIT Press.
16. Wan, X., Wang, W., Liu, J., & Tong, T. (2014). *Estimating the sample mean and standard deviation from the sample size, median, range and/or interquartile range.* BMC Medical Research Methodology, 14, 1-13. DOI: https://doi.org/10.1186/1471-2288-14-135.
17. Wasserman, L. (2013). *All of statistics: a concise course in statistical inference.* Springer Science & Business Media.
18. De Cock, D. (2011). *Ames, Iowa: Alternative to the Boston housing data as an end-of-semester regression project*. *Journal of Statistics Education, 19*(3), 1-15. https://jse.amstat.org/v19n3/decock.pdf.

М. М. Злобін, **В. М. Базилевич**, доц., канд. екон. наук
НУ «Чернігівська політехніка», Чернігів, Україна

Підхід на основі даних для збалансування перенавчання та недонавчання в моделях дерева рішень

Стаття присвячена розробці підходу на основі даних для балансування надмірної (overfitting) та недостатньої пристосованості (underfitting) в моделях дерев рішень. Надмірна пристосованість зазвичай виникає, коли модель вловлює шум, зменшуючи узагальнення, тоді як недостатня пристосованість призводить до низької точності прогнозування. У дослідженні систематично налаштовувався параметр max_leaf_nodes та оцінювалась ефективність моделі за допомогою середньої абсолютної помилки (MAE). Мета полягала в тому, щоб знайти оптимальний баланс, який забезпечує точність моделі, запобігаючи при цьому її надмірній складності.

Регресор дерева рішень (A Decision Tree Regressor) навчався на наборі даних Ames Housing, який включає 79 пояснювальних змінних, пов'язаних з цінами на житло. Набір даних було розділено на навчальний та валідаційний набори (тобто на набори для навчання та перевірки). Модель оцінювалася шляхом ітерації над різними значеннями max_leaf_nodes, від 2 до 5000, і обчислення MAE для кожної конфігурації. Результати показали, що збільшення max_leaf_nodes спочатку покращувало точність, але після 400 вузлів MAE стабілізувалося на рівні 242,906, що свідчило про те, що подальше ускладнення не покращувало продуктивність. У статті підкреслено, що моделі з надто малою кількістю листкових вузлів не відповідають даним, тоді як моделі з надто великою кількістю листкових вузлів - надмірно пристосовуються, захоплюючи помилкові патерни. Для пом'якшення цієї проблеми використано систематичне налаштування гіперпараметрів для пошуку оптимальної конфігурації. Також досліджено вплив перехресної перевірки, скорочення та обмежень на глибину дерева на узагальнення моделі. Висновки свідчать, що вибір відповідного значення max_leaf_nodes запобігає надмірному пристосуванню, зберігаючи при цьому сильну прогностичну силу.

У статті показано важливість структурованого налаштування гіперпараметрів у моделях дерева рішень. Оптимальне значення max_leaf_nodes знаходиться на рівні 400. Фреймворк можна адаптувати до інших моделей машинного навчання, де MAE можна використовувати для оцінки продуктивності при різних налаштуваннях параметрів. Наприклад, у моделях випадкового лісу (Random Forest) кількість дерев можна оптимізувати аналогічно.

Результати підкреслюють, що налаштування складності моделі має важливе значення для досягнення точних прогнозів, уникаючи при цьому надмірного пристосування. У подальших роботах слід дослідити інтеграцію алгоритмів автоматизованого налаштування та ансамблевих методів для покращення продуктивності дерев рішень.

регресор дерева рішень, надмірне пристосування, перенавчання, недостатнє пристосування, недонавчання, оптимізація моделі, гіперпараметричне налаштування

Одержано (Received) 10.03.2025 *Прорецензовано (Reviewed) 12.03.2025*
Прийнято до друку (Approved) 14.03.2025