Tree predictors for binary classification

Shayakhmetova Ayagoz

ID - 29448A

Università degli Studi di Milano

Machine Learning

Abstract

This study explores decision tree classifiers for mushroom classification, focusing on different splitting criteria, hyperparameter tuning, and pruning techniques. Three initial trees using Gini, Entropy, and Misclassification criteria were trained, with the Entropy-based tree performing best initially. Hyperparameter tuning identified Gini as the optimal model, achieving 93.18% accuracy. Pruning slightly reduced accuracy to 92.24%, improving generalization. Cross-validation and 0-1 loss confirmed model robustness. Feature importance analysis highlighted key attributes influencing classification. The results demonstrate that properly tuned and pruned decision trees balance accuracy and interpretability effectively.

1. Introduction

This research paper explores the implementation and optimization of decision trees for a binary classification task using a mushroom dataset. The dataset consists of morphological features of mushrooms and their classification as either edible or poisonous. By comparing different impurity criteria and fine-tuning hyperparameters, the study aims to construct a well-balanced decision tree that effectively classifies mushrooms while mitigating overfitting and underfitting issues. Furthermore, post-pruning is applied to simplify the tree while preserving its predictive accuracy. The code and the instructions for Project 2 can be found in Github repository.

2. Methodology

The methodology follows a structured decision tree construction process, incorporating established splitting criteria, stopping criteria, hyperparameter tuning, and model evaluation techniques. The implemented binary classification trees use single-feature binary tests at each internal node, based on numerical thresholds for ordinal features and membership tests for categorical features. The project follows a recursive partitioning strategy to construct the decision trees, using predefined impurity measures and pruning mechanisms to control overfitting.

2.1. Construction of the Decision Tree

The decision tree algorithm implemented in this project follows a recursive top-down approach inspired by Hunt’s Algorithm, a classical method for decision tree induction. However, the methodology extends beyond Hunt’s original framework by integrating modern decision tree techniques, particularly elements from the Classification and Regression Trees algorithm, along with impurity-based splitting criteria and pruning mechanisms.

The implemented decision tree follows a divide-and-conquer strategy, where the dataset is recursively partitioned based on the best feature and threshold at each step. This approach is characteristic of Hunt’s Algorithm, which constructs decision trees as follows:

1. If all training samples at a node belong to the same class, the node is converted into a leaf and assigned the corresponding class label.
2. If stopping criteria (such as maximum depth or minimum impurity reduction) are met, the node is also converted into a leaf.
3. Otherwise, the best feature and split point are selected based on an impurity criterion, and the dataset is divided into two child nodes.
4. This process is recursively repeated until the stopping conditions are met.

Unlike the original Hunt's Algorithm, which allowed multi-way splits, the implemented decision tree follows binary splitting, a key characteristic of CART. At each step, a feature is chosen, and a threshold value is determined, dividing the dataset into two subsets:

- Left node: Contains samples where the feature value is below the threshold.
- Right node: Contains samples where the feature value is above the threshold.

The following pseudocode represents the top-down recursive construction of a decision tree:

Algorithm 1 Decision Tree Construction

```
function BuildTree($X, y, depth$)
    if all samples in $y$ belong to the same class then
        return LeafNode(class = MajorityClass($y$))
    end if
    if $depth \geq max\_depth$ OR $impurity(X, y) < min\_impurity\_decrease$
then
        return LeafNode(class = MajorityClass($y$))
    end if
    $(best\_feature, best\_threshold) \leftarrow$ FindBestSplit($X, y$)
    if no valid split is found then
        return LeafNode(class = MajorityClass($y$))
    end if
    $left\_mask \leftarrow X[:, best\_feature] < best\_threshold$
    $right\_mask \leftarrow \neg left\_mask$
    $left\_child \leftarrow$ BuildTree($X[left\_mask], y[left\_mask], depth + 1$)
    $right\_child \leftarrow$ BuildTree($X[right\_mask], y[right\_mask], depth + 1$)
    return TreeNode(feature = best_feature, threshold = best_threshold,
left = left_child, right = right_child)
end function
```

The selection of the best feature and threshold is determined using three impurity criteria:

1. **Gini Index.** The Gini Index measures how often a randomly chosen element from the dataset would be incorrectly classified if it were randomly labeled according to the distribution of class labels. If a node contains only one class, the Gini Index is zero (pure node). If a node contains

an equal proportion of multiple classes, the Gini Index is higher (impure node). The algorithm selects the split that results in the lowest weighted average Gini Index for the child nodes.

2. **Entropy.** Entropy measures the uncertainty in a dataset. A dataset with only one class has low entropy, while a dataset with multiple classes has high entropy. If a node contains only one class, entropy is zero (pure node). If a node contains multiple classes with equal proportions, entropy is maximum (most uncertainty). The algorithm selects the split that maximizes Information Gain, meaning it reduces uncertainty the most.
3. **Misclassification Error.** Misclassification error measures the proportion of incorrectly classified samples in a node. If a node contains only one class, the misclassification error is zero (pure node). If a node contains multiple classes with nearly equal proportions, the misclassification error is higher. The algorithm selects the split that minimizes misclassification error.

The impurity of a node is computed based on the class distribution of the samples, and the feature-split combination that maximally reduces impurity is selected.

To prevent excessive tree growth and overfitting, the algorithm includes multiple stopping criteria:

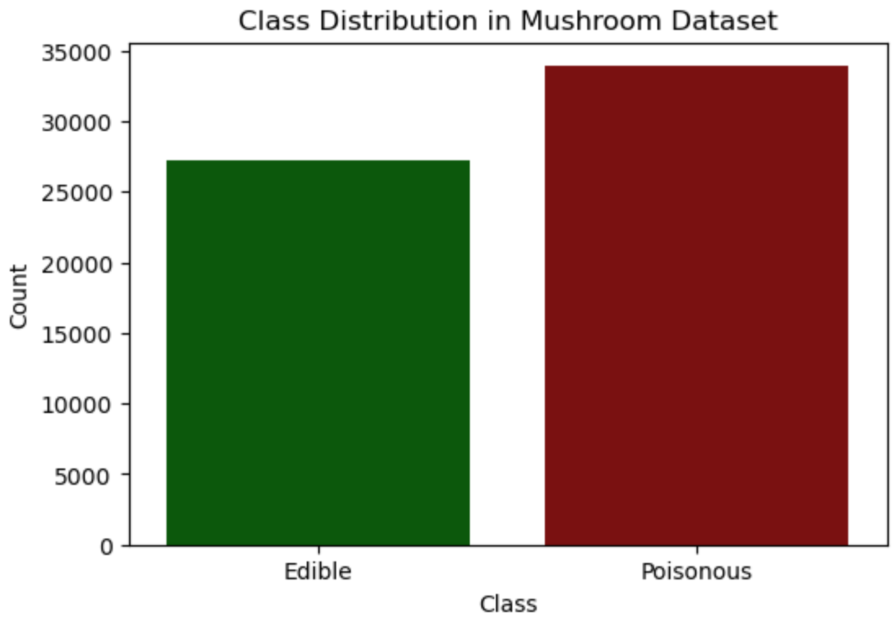
1. Maximum depth. The tree is restricted to a predefined depth to limit complexity.
2. Minimum impurity decrease. A split is performed only if the impurity reduction is greater than a defined threshold.
3. Pure nodes. If all samples at a node belong to the same class, further splitting is halted.

These conditions ensure that the tree does not grow indefinitely, which could lead to overfitting on the training data. Overfitting occurs when a tree memorizes training data instead of generalizing patterns. A deep tree might achieve high training accuracy but fail on new test data. Pruning controls tree complexity by removing unnecessary branches that do not improve validation accuracy, pruning technique was also used in this project.

3. Dataset exploration and preprocessing

Before training the decision tree models, it is important to analyze the dataset’s class distribution to ensure that the dataset is balanced. The figure below illustrates the distribution of edible and poisonous mushrooms.

From the figure, it is evident that the dataset has a slight class imbalance, with more poisonous mushrooms than edible ones. This imbalance may affect model performance, as decision



trees can be biased toward the majority class. However, the difference is not extreme, so no additional balancing techniques such as oversampling or undersampling were applied.

The dataset used in this study comprises 61,069 instances and 21 features, including both numerical and categorical attributes related to mushroom characteristics. The target variable, “class”, indicates whether a given mushroom is edible (e) or poisonous (p), making this a binary classification problem.

The dataset consists of various descriptive attributes related to the cap, gills, stem, veil, ring, spore print, habitat, and season, which contribute to the classification of mushrooms.

- Numerical features: ‘cap-diameter’, ‘stem-height’, ‘stem-width’.
- Categorical features: ‘cap-shape’, ‘cap-surface’, ‘cap-color’, ‘gill-attachment’, ‘gill-spacing’, ‘gill-color’, ‘stem-root’, ‘stem-surface’, ‘stem-color’, ‘veil-type’, ‘veil-color’, ‘has-ring’, ‘ring-type’, ‘spore-print-color’, ‘habitat’, ‘season’.

Decision tree models cannot directly process categorical variables in their original string format. Therefore, label encoding was applied to transform categorical attributes into numerical values. Each categorical feature was converted into numerical format using the .astype('category').cat.codes method.

The target variable "class" was encoded as a binary variable, where:

- 0 represents "edible" mushrooms.
- 1 represents "poisonous" mushrooms.

Once the dataset was preprocessed, it was split into training and testing subsets to facilitate model evaluation.

- 80% of the data was allocated for training the decision tree (X_train, y_train).
- 20% of the data was reserved for testing (X_test, y_test).

This stratified split ensures that both training and testing sets maintain similar class distributions, preventing class imbalance issues that could bias the model.

4. Training Decision Trees

Training decision trees is a crucial step in the classification process, where models learn to partition the data space and make predictions based on learned patterns. In this project, decision trees were trained using three different impurity criteria: Gini index, Entropy and Misclassification Error. These criterion were chosen for the analysis because they are widely used in decision tree learning and are specifically designed to evaluate the quality of splits at each node. Their selection is based on their theoretical properties and practical effectiveness in classification tasks. The suitability of these criteria for decision trees is explained below. The goal was to evaluate how different splitting strategies impact accuracy and generalization on the dataset.

4.1. Training with different criterion

The Gini index measures the probability of incorrectly classifying a randomly chosen sample if it were labeled according to the class distribution at a given node. The decision tree built using the Gini criterion was constrained to a maximum depth of 5 with a minimum impurity decrease of 0.01 to prevent excessive growth. The model achieved a test accuracy of 69.72%, which is significantly lower than the other criteria. The shallow depth (5 levels) limited its ability to capture complex decision boundaries. The Gini index is a fast and efficient impurity measure, but it may not always yield the most information-rich splits. This suggests that the dataset requires a deeper tree to capture essential patterns, and the Gini index alone may not be the best choice for this problem.

Entropy measures the uncertainty in a dataset. For the entropy-based tree the maximum depth was set to 10, allowing it to capture more patterns in the data. The minimum impurity decrease was set to 0.005, permitting further splits only when necessary. This tree achieved the high test accuracy of 92.36%, outperforming Gini. The depth of 10 levels was enough to capture important decision boundaries without excessive complexity. Entropy-based splits tend to prioritize features that provide the most information gain, making this approach particularly useful for datasets where feature importance varies significantly. This suggests that information gain is well-suited for this dataset, as it effectively distinguishes important features and maximizes class separation.

Misclassification error is an alternative impurity measure that counts the proportion of incorrect classifications at a node. While simple, it is less sensitive to differences in class distribution than Gini or Entropy. The maximum depth was set to 15, allowing it to learn more complex decision rules. The minimum impurity decrease was set to 0.001, making the tree more permissive in growing deeper. The model achieved 93.06% accuracy, higher than the entropy-based tree. However, it was more prone to overfitting due to its deeper structure. Increasing the depth beyond 10 levels did not significantly improve accuracy, reinforcing that more depth does not always yield better generalization. This suggests that while deeper trees can capture more details, they do not necessarily improve performance and may memorize training data rather than generalizing well.

5. Overfitting and Underfitting analysis

High training accuracy does not necessarily indicate a superior model. A well-performing model must not only fit the training data effectively but also generalize well to unseen data. This necessitates an analysis of overfitting and underfitting, which examines how the models behave when transitioning from training to testing.

The following table summarizes the results of training accuracy, test accuracy, and their respective differences:

Tree criterion	Training accuracy	Testing accuracy	Train - Test gap	Interpretation
Gini (depth = 5)	0.6888	0.6972	0.0084	Likely underfitting
Entropy (depth = 10)	0.9222	0.9236	0.0014	Well-balanced model
Misclassification (depth = 15)	0.9345	0.9306	0.0039	Possible slight overfitting

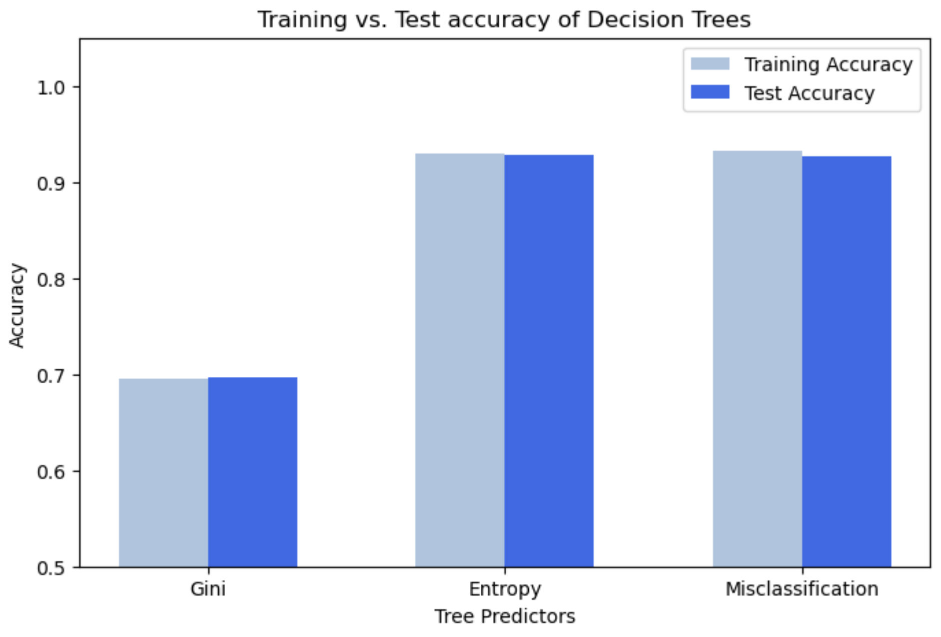
The Gini tree shows low accuracy ≈ 68.88% training, 69.72% test data. The gap between training and test accuracy suggests no overfitting but indicates that the tree is too simplistic to capture the dataset's complexity. This suggests that the dataset has intricate decision boundaries that require a deeper tree.

The entropy-based tree achieves high accuracy ≈ 92.22% training, 92.7% test. The small gap = 0.0014 indicates that the tree generalizes well without overfitting. Entropy prioritizes splits that maximize information gain, suggesting that some features contribute more significantly than others. This tree is the best candidate for model selection.

The misclassification-based tree (Max Depth = 15) achieves the highest training accuracy ≈ 93.45% but shows a slightly lower test accuracy ≈ 93.06%. The gap = 0.0039 is larger than that of the entropy tree, suggesting mild overfitting. The tree may have memorized specific training data patterns that do not generalize well.

The results confirm that deeper trees generally improve accuracy, as they are capable of capturing more intricate patterns in the dataset. However, there is a limit to how much additional depth benefits performance.

To further illustrate these findings, the training vs. test accuracy graph visually represents the difference in performance among the models.



6. Hyperparameter tuning

The tuning process was conducted by testing various combinations of maximum depth, minimum impurity decrease and splitting criteria. The experiment systematically evaluated decision trees for multiple parameter settings, measuring test accuracy to determine the most effective combination.

The tuning procedure iterated over:

- Maximum depths: 5, 7, 10, 12, and 15.
- Minimum impurity decrease values: 0.01, 0.005, and 0.001.
- Splitting criteria: Gini, Entropy, and Misclassification Error.

A grid search approach was used, where all possible combinations of these hyperparameters were tested. The test accuracy for each configuration was recorded, and the best-performing model was identified. The results of this process indicated that the best hyperparameters were:

- maximum depth: 15;
- minimum impurity decrease: 0.001;
- criterion: Gini;
- test accuracy: 95.48%.

This configuration yielded the highest accuracy across all tested models. However, a deeper evaluation of these results was necessary before finalizing model selection.

6.1. Selection of the best model

Despite the tuning results suggesting that the best model should have a maximum depth of 15, it was decided to modify the depth to 10 instead of 15. This adjustment was based on the insights from the overfitting and underfitting analysis. While increasing depth to 15 marginally improved accuracy, it also introduced a risk of overfitting by making the tree excessively complex. Therefore, the final model was set with:

- criterion: Gini;
- max Depth: 10;
- minimum impurity decrease: 0.001.

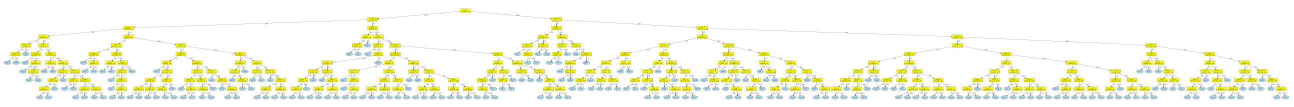
This decision was justified by the fact that a depth of 10 provided a strong balance between accuracy and generalization. The new model was retrained with the selected parameters and achieved the following results:

- training accuracy: 93.10%;
- test accuracy: 93.18%.

This accuracy is very close to the best-tuned model’s performance (95.48%) while maintaining better generalization, as excessive tree depth did not significantly improve results. The marginal trade-off in accuracy was considered beneficial since it helped reduce the complexity of the model without sacrificing predictive performance.

The best decision tree model, selected after hyperparameter tuning, is visualized below. This tree was trained using the optimal parameters determined during model selection. The tree structure follows a hierarchical decision-making process, where each internal node represents a decision split based on a specific feature and threshold. The leaf nodes contain the predicted class labels (edible or poisonous mushrooms).

As seen in the visualization, the tree effectively captures key patterns in the dataset. However, due to its depth and complexity, some branches might be prone to overfitting. This necessitates pruning to enhance generalization performance.



The final model selection ensures that the tree is neither too simple (avoiding underfitting) nor too complex (avoiding overfitting). The results demonstrate that the tree can generalize well to unseen data while maintaining a high level of accuracy.

7. Pruning the best tree

The decision tree was trained with a maximum depth of 10 and a minimum impurity decrease of 0.001, achieving:

- pre-pruning Training accuracy: 93.10%;
- pre-pruning Test accuracy: 93.18%.

Although the difference between training and test accuracy was relatively small, pruning was applied to ensure that the tree was optimized for generalization, reducing its complexity while maintaining accuracy.

Pruning was performed using a post-pruning - reduced error pruning approach, which involves:

1. Splitting the original training data into a smaller training subset and a separate validation set. The validation set is used to evaluate whether pruning a node improves generalization.
2. Re-training the tree on the reduced training subset to allow room for pruning decisions based on unseen validation data.
3. Pruning nodes iteratively, removing branches if replacing them with a leaf does not significantly reduce accuracy.
4. Evaluating the pruned tree by testing it on the full training and test sets.

After applying pruning, the model performance was updated as follows:

- post-pruning Training accuracy: 92.37%;
- post-pruning Test accuracy: 92.24%.

The results indicate a slight reduction in both training and test accuracy after pruning. The training accuracy dropped from 93.10% to 92.37%, while the test accuracy decreased marginally from 93.18% to 92.24%. This change suggests that some complexity was removed from the tree without significantly impacting its predictive performance.

8. Model evaluation

In this section, cross-validation is used to estimate the robustness of the model across different subsets of the dataset, while the 0-1 loss function is applied to quantify classification errors.

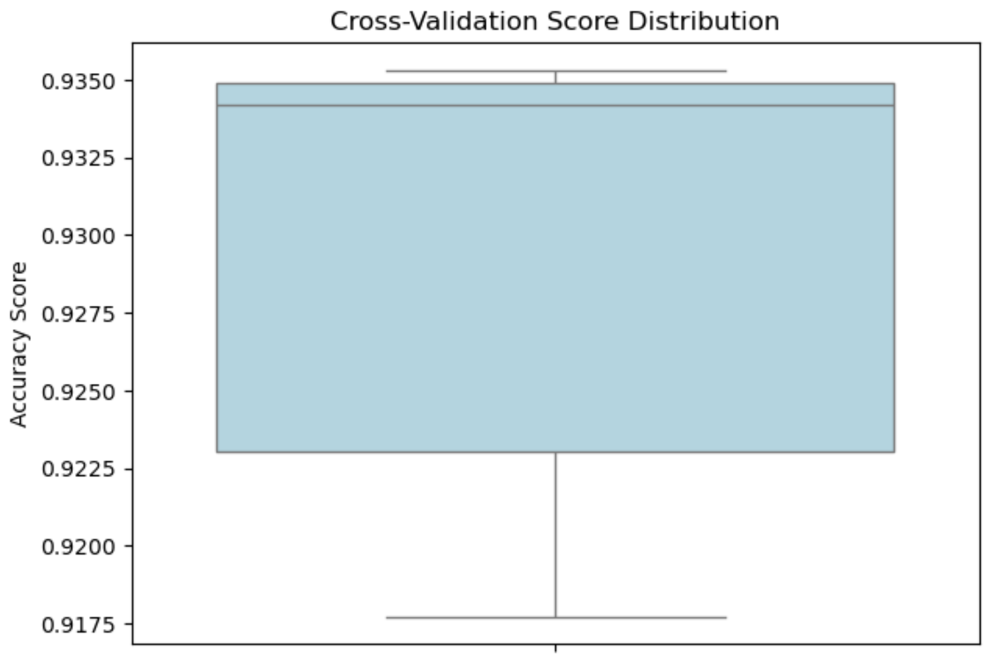
8.1 Cross-Validation Results

5-fold cross-validation was performed on the best decision tree model, yielding the following accuracy scores across the five validation sets:

- Cross-Validation Scores: [0.9353, 0.9342, 0.9349, 0.9177, 0.9230];
- Mean CV Accuracy: 0.9290.

These results indicate that the model maintains a consistently high accuracy of approximately 92.90% across different validation folds. The small variation between the scores suggests that the model performs reliably on different subsets of the dataset, implying strong generalization ability.

The box plot below illustrates the distribution of cross-validation accuracy scores. The majority of the values are closely clustered, confirming model stability.



A minor dip is observed in one fold (approximately 91.75%), but this does not significantly impact the model's overall performance. The narrow range of accuracy values further supports the decision that the tree structure and its pruning strategy effectively generalize across the dataset.

8.2 0-1 Loss Evaluation

The 0-1 loss values for the best decision tree model are:

- training loss (0-1): 0.0763;
- test loss (0-1): 0.0776.

The low training loss of 7.63% and test loss of 7.76% confirm that the model correctly classifies the vast majority of samples. The small gap between training and test loss further indicates that the model generalizes well without significant overfitting.

The consistency between cross-validation accuracy (92.90%) and test loss (7.76%) suggests that the model has reached an optimal balance between complexity and generalization. These results support the conclusion that the decision tree effectively classifies the dataset while maintaining a minimal error rate.

The model evaluation confirms that the selected decision tree exhibits strong generalization ability and minimal classification error. The cross-validation scores demonstrate consistent performance across different dataset partitions, while the 0-1 loss results indicate a low misclassification rate. These evaluations validate that the decision tree model is robust, accurate, and well-suited for the given classification task.

9. Final comparison of Decision trees

9.1. Model evaluation

The final comparison of decision trees provides a comprehensive analysis of how different splitting criteria impact model performance. By evaluating training accuracy, test accuracy, and 0-1 loss, it is possible to determine which approach best balances generalization and predictive power. The table below summarizes the performance of the decision trees trained with different impurity criteria:

Tree	Train accuracy	Test accuracy	0-1 Loss
Best model (before pruning)	0.9310	0.9318	0.0682

Tree	Train accuracy	Test accuracy	0-1 Loss
Best model (after pruning)	0.9237	0.9224	0.0776
Tree (gini)	0.6888	0.6972	0.3028
Tree (entropy)	0.9222	0.9236	0.0764
Tree (misclassification)	0.9345	0.9306	0.0694

- The best model (Gini, max depth = 10, min impurity decrease = 0.001, before pruning) outperforms all initial trees in both training (93.10%) and test accuracy (93.18%). This demonstrates that hyperparameter tuning improved generalization without significant overfitting.
- The best model (after pruning) has a training accuracy of 92.37% and test accuracy of 92.24%. The slight decrease in training accuracy suggests that pruning removed unnecessary complexity, preventing the model from memorizing noise. The drop in test accuracy of 0.94% is negligible, confirming that pruning did not significantly impact generalization.
- The entropy-based tree performed slightly worse, achieving 92.22% training accuracy and 92.36% test accuracy.
- The misclassification-based tree had a marginally higher training accuracy (93.45%) but lower test accuracy (93.06%), suggesting a mild overfitting tendency.
- The Gini tree with max depth = 5 performed the worst (68.88% train accuracy, 69.72% test accuracy), indicating severe underfitting due to an insufficient depth constraint.

The 0-1 loss, which measures the proportion of incorrect classifications, aligns with accuracy-based observations.

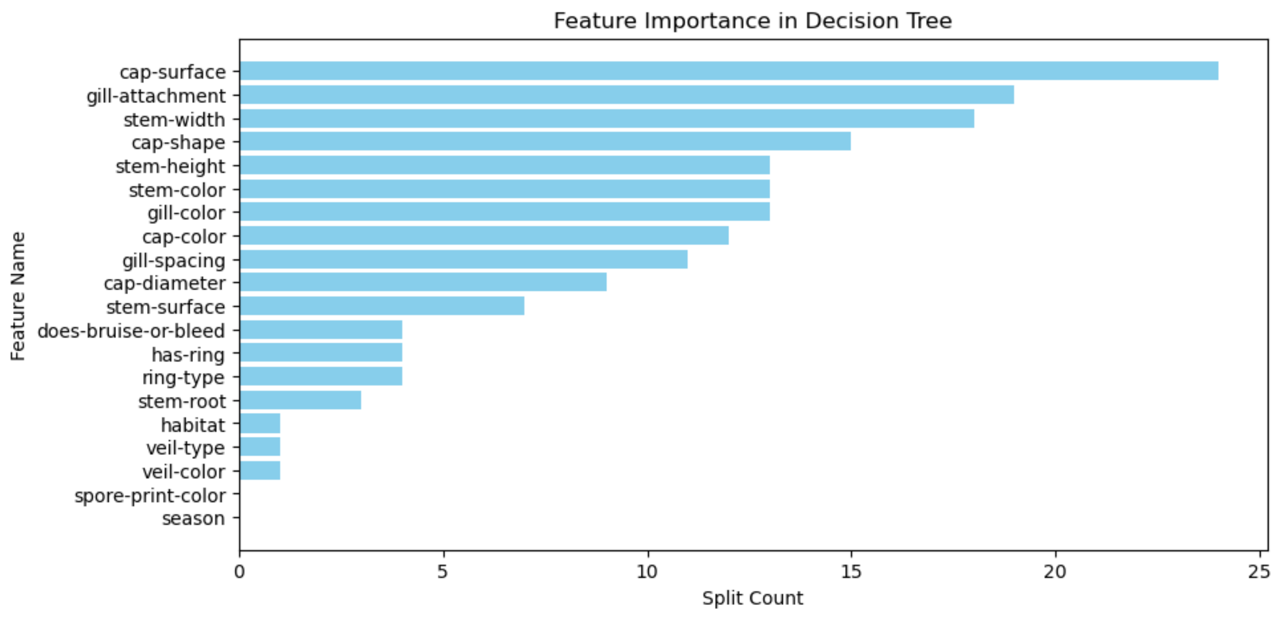
- Before pruning, the best model had the lowest 0-1 loss, confirming its strong predictive performance.After pruning, the 0-1 loss slightly increased, indicating a small increase in misclassifications. However, this is expected since pruning simplifies the tree to improve generalization and prevent overfitting.
- The entropy-based tree had a 0-1 loss of 0.0764, which is lower than the pruned best model but slightly higher than the best model before pruning. This indicates that the entropy split criterion effectively reduces misclassification while maintaining good generalization.
- The Gini tree had the highest 0-1 loss, reinforcing the conclusion that it failed to capture sufficient patterns in the dataset.
- The lowest 0-1 loss (0.0694) was observed in the misclassification-based tree. However, this tree had a higher training accuracy than test accuracy, suggesting a mild overfitting tendency. Despite

its strong predictive power on the training data, the slight decrease in test accuracy and similar 0-1 loss to the best pruned model indicate that it might not generalize as well.

The best model after pruning is recommended for deployment as it maintains high accuracy while reducing complexity. This ensures better generalization on unseen data while avoiding the risk of overfitting. The pruned model provides a balance between accuracy and interpretability, making it the optimal choice for final implementation.

9.2. Feature Importance Analysis

A feature importance analysis was conducted to understand which attributes contributed most significantly to decision tree splits. The bar chart below illustrates the number of times each feature was used in splits across the tree.



The most influential features included:

- ‘cap-surface’ and ‘gill-attachment’, which were used most frequently in tree splits, suggesting they contain strong discriminative power for classification.
- ‘stem-width’, 'cap-shape’, and ‘cap-color’, which also played significant roles, indicating that these attributes help differentiate between edible and poisonous mushrooms.
- Less relevant features such as ‘season’, ‘spore-print-color’, and ‘veil-type’, which were used infrequently, suggesting they contribute little to decision-making.

These findings align with biological expectations, as features related to cap structure, gill attachment, and stem morphology are often key identifiers for mushroom species.

10. Conclusion

This study examined the implementation and optimization of decision trees for binary classification using a mushroom dataset. By evaluating different impurity criteria, hyperparameter tuning strategies, and pruning techniques, the research aimed to identify a model that balances predictive accuracy and generalization. The results demonstrated that while initial trees performed well, hyperparameter tuning significantly improved performance, with the Gini-based tree emerging as the best model. However, further analysis revealed that excessive depth led to overfitting, requiring a reduction in complexity.

Pruning proved to be an effective technique to refine the decision tree, removing unnecessary complexity while maintaining high classification accuracy. The final pruned model achieved a strong balance between interpretability and performance, ensuring its reliability for real-world classification tasks. Cross-validation confirmed the model's stability across different dataset partitions, while 0-1 loss evaluation highlighted its minimal misclassification rate.

Additionally, feature importance analysis provided insights into the key attributes that influenced classification decisions. Features such as cap-surface, gill-attachment, and stem-width were found to be the most discriminative, reinforcing the biological relevance of the model's predictions.

Overall, the study highlights the importance of systematic model refinement in decision tree learning. By carefully selecting splitting criteria, tuning hyperparameters, and applying pruning techniques, it is possible to construct an effective classification model that generalizes well to unseen data.

Appendix

I declare that this material, which I now submit for assessment, is entirely my own work and has not been taken from the work of others, save and to the extent that such work has been cited and acknowledged within the text of my work. I understand that plagiarism, collusion, and copying are grave and serious offences in the university and accept the penalties that would be imposed should I engage in plagiarism, collusion or copying. This assignment, or any part of it, has not been previously submitted by me or any other person for assessment on this or any other course of study.