Machine Learning And Statistical Learning

Machine Learning Project: Tree Predictors for Binary Mushroom Classification

Prepared by Adiya Sapargali

DSE, second year

Matricola number: 33486A

Table of Contents

Introduction

This project implements *decision tree* and *random forest classifiers* from scratch for the binary classification task of identifying whether mushrooms are edible or poisonous using the Mushroom dataset. The main goal is to understand the internal mechanics of tree-based models, including their training, splitting criteria, and stopping conditions.

The implementation includes reading and preprocessing the Mushroom dataset, analyzing feature distributions, and building a custom decision tree classifier. Three splitting criteria *(Gini impurity, Entropy, and Classification Error)* were implemented to assess their influence on tree structure and model performance. Stopping criteria such as maximum tree depth and minimum samples per leaf were used to prevent overfitting. Additionally, a random forest classifier was developed by aggregating multiple decision trees trained on bootstrapped samples with random feature subsets to enhance model robustness.

Model performance was evaluated using accuracy, confusion matrices, and classification reports. The results highlighted the trade-offs between different splitting criteria, the impact of stopping rules, and the benefits of ensemble methods in improving generalization.

Data Loading and Preprocessing

The project utilized two datasets: the primary dataset and the secondary dataset, both in CSV format. They were loaded into the analysis environment using appropriate delimiters and encodings to ensure correct parsing. All column names were stripped of leading and trailing whitespace to maintain consistency throughout the analysis.

The primary dataset contained certain numerical columns represented as string ranges (e.g., “[4, 8]”). To ensure consistency and usability of these features, a cleaning process was performed that converted these string ranges into single numeric values by calculating their averages. This transformation standardized the numerical data and facilitated subsequent analysis and modeling.

For the secondary dataset, which was already clean, no further transformations were required beyond basic verification.

Missing values were evaluated during the exploratory data analysis stage. Based on the findings, rows containing missing values were removed from the datasets to ensure the models were trained on complete and consistent records. This approach aimed to maintain data quality without introducing potential bias from imputation.

Exploratory Data Analysis (EDA)

The Exploratory Data Analysis (EDA) stage aimed to understand the structure and characteristics of the primary and secondary datasets, identify potential issues, and inform subsequent modeling decisions.

Primary Dataset

The primary dataset was analyzed to determine the distribution of the target variable (class) and to explore the behavior of both numerical and categorical features.

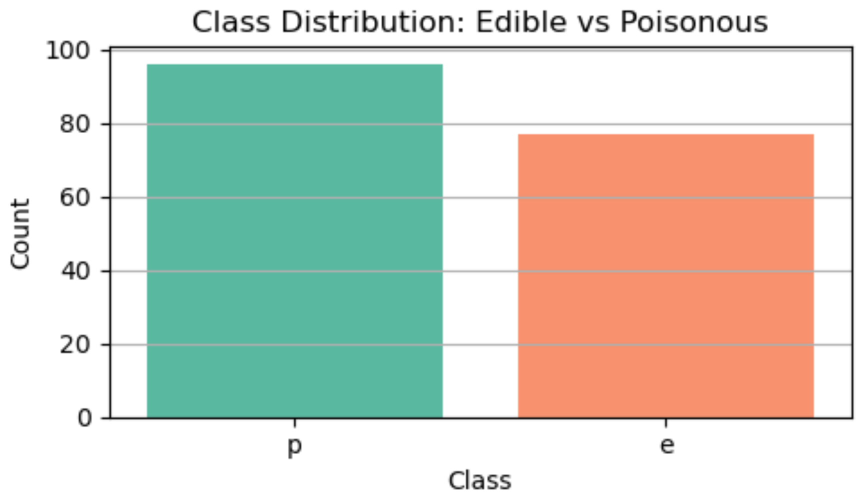
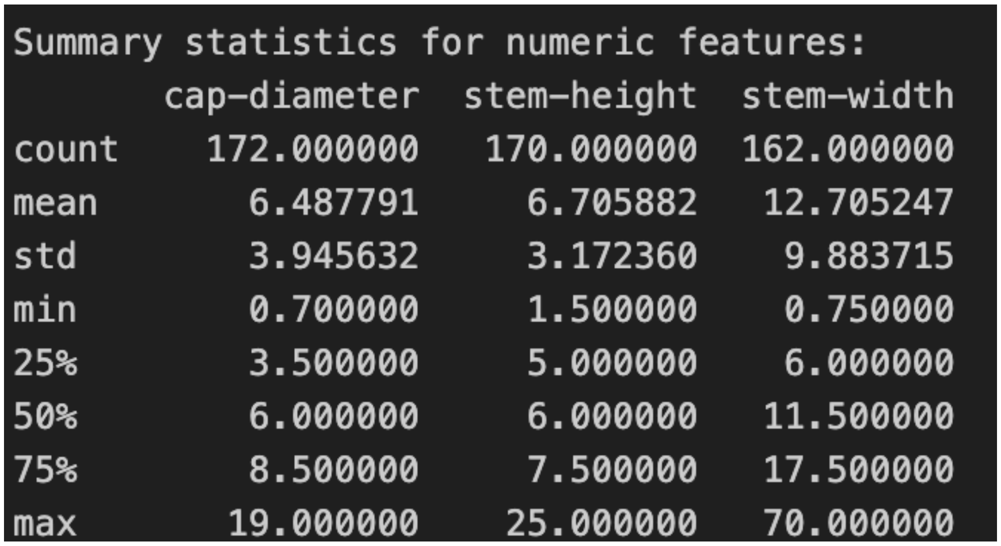


Figure 1. Class Distribution (Edible vs. Poisonous)

The dataset contains two target classes: edible (e) and poisonous (p). There are 96 poisonous samples and 77 edible ones, meaning approximately 55.5% of the mushrooms are poisonous and 44.5% are edible. This indicates a slight class imbalance, with poisonous mushrooms being the majority class. This distribution highlighted the importance of stratified sampling when splitting data for training and testing to ensure both classes were proportionally represented.



```
Summary statistics for numeric features:
       cap-diameter  stem-height  stem-width
count    172.000000   170.000000  162.000000
mean       6.487791     6.705882   12.705247
std        3.945632     3.172360    9.883715
min        0.700000     1.500000    0.750000
25%        3.500000     5.000000    6.000000
50%        6.000000     6.000000   11.500000
75%        8.500000     7.500000   17.500000
max       19.000000    25.000000   70.000000
```

Table 1. Summary Statistics for Numeric Features

The table shows summary statistics for three numeric features in the dataset: cap diameter, stem height, and stem width. On average, mushroom caps are about 6.49 cm in diameter, stems are about 6.71 cm tall, and 12.71 cm wide. The values vary, with cap diameter ranging from 0.7 to 19.0 cm, stem height from 1.5 to 25.0 cm, and stem width from 0.75 to 70.0 cm. The stem width has the highest variation, as shown by its large standard deviation and wide range, suggesting possible outliers. Most values fall within the interquartile range, which helps understand the typical sizes of these features in the dataset.

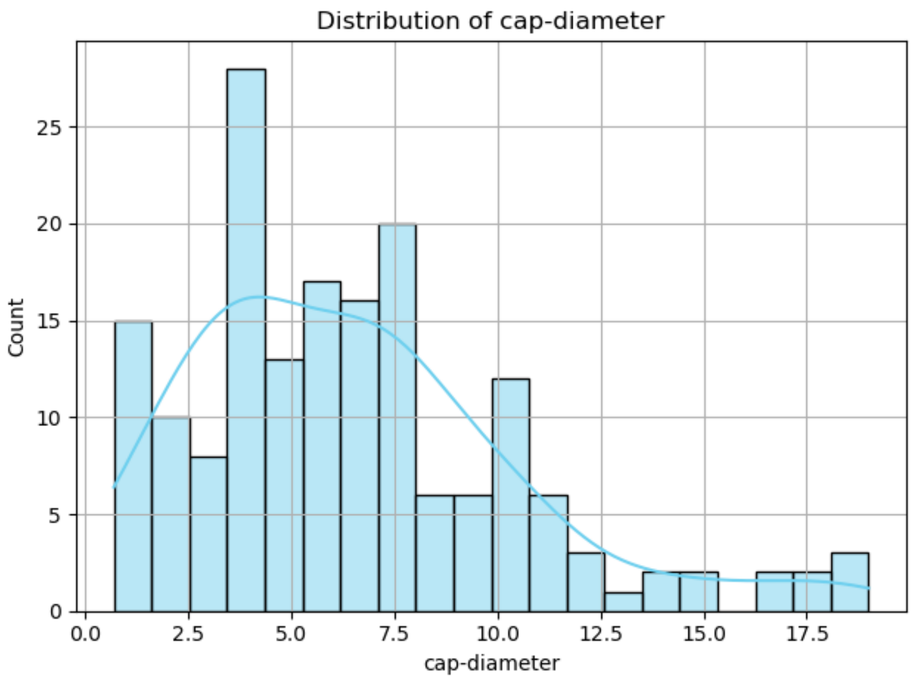


Figure 2. Distribution of cap-diameter

This plot shows the distribution of the cap-diameter feature in the dataset. Most mushrooms have a cap diameter between 2 and 8 centimeters, with a clear peak around 4 cm. The distribution is right-skewed, meaning there are a few mushrooms with much larger cap diameters, up to 18–19 cm, but they are relatively rare. The shape of the curve suggests that smaller caps are more common, and the number of samples decreases as the diameter increases. This information is useful to understand the variability of this feature and check whether it contains extreme values or outliers that could affect model performance.

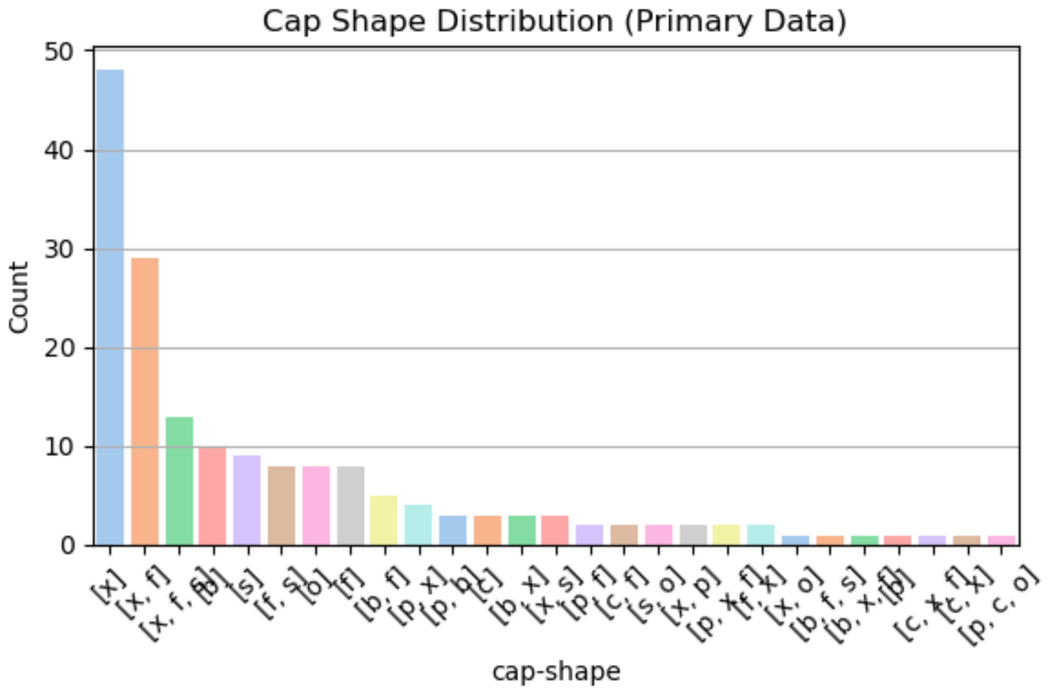


Figure 3. Cap shape distribution

This plot shows the distribution of different cap-shape combinations in the primary mushroom dataset. The most common cap shape is represented by the value [x], which appears nearly 50 times. The next most frequent shapes include [x, f] and [f], but their counts drop off quickly compared to the most common one. The distribution is highly skewed, with a small number of shape combinations making up the majority of samples, while many others occur only a few times. This suggests that certain cap shapes are much more typical in the dataset, and the rare combinations may have limited impact on the overall model but could be important for specific identification cases.

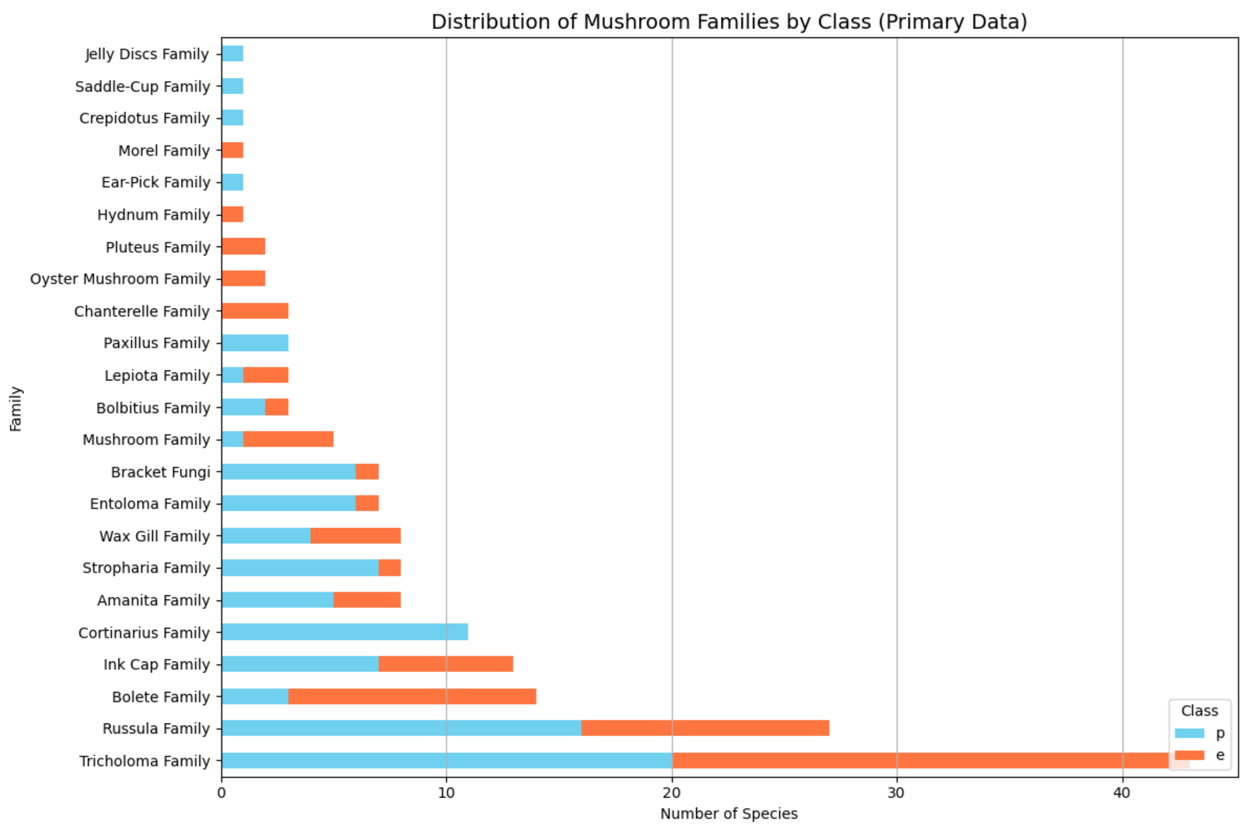


Figure 4. Distribution of Mushroom Families by Class

This plot shows the distribution of mushroom families by class (edible vs. poisonous) in the primary dataset. It helps reveal how different families are associated with each class. For example, the Tricholoma and Russula families contain many edible species, while the Cortinarius and Amanita families include more poisonous ones. Some families, like the Boletus and Ink Cap, contain both edible and poisonous species in nearly equal proportions. It helps identify which families might be strong indicators of class, uncover potential patterns in species distribution, and guide feature importance or selection decisions in model building. It also highlights class imbalance across categories, which can impact classification performance.

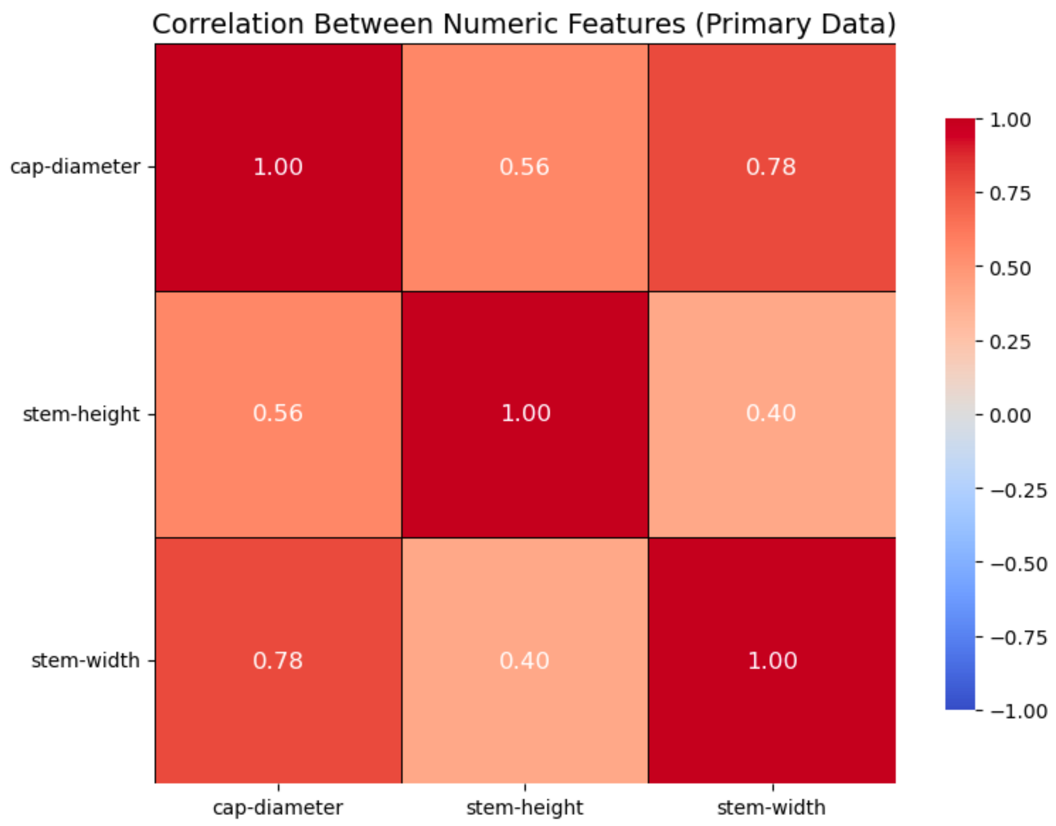


Figure 5. Correlation between Numeric Features

This heatmap shows the correlation between the three numeric features in the primary dataset: cap-diameter, stem-height, and stem-width. The values range from 0.40 to 0.78, indicating moderate to strong positive correlations. The strongest correlation is between cap-diameter and stem-width (0.78), meaning mushrooms with larger caps tend to also have thicker stems. The correlation between cap-diameter and stem-height is also moderate (0.56), while stem-height and stem-width are less correlated (0.40). It helps identify relationships between features, which can inform feature selection, model interpretation, or the need for dimensionality reduction. It also shows that none of the numeric features are redundant (correlation close to 1), so all of them may provide unique information for classification.

Secondary Dataset

The secondary dataset is used for model training. The analysis focuses on class balance, numeric feature behavior, categorical variable distributions, and potential feature relationships with the target variable.

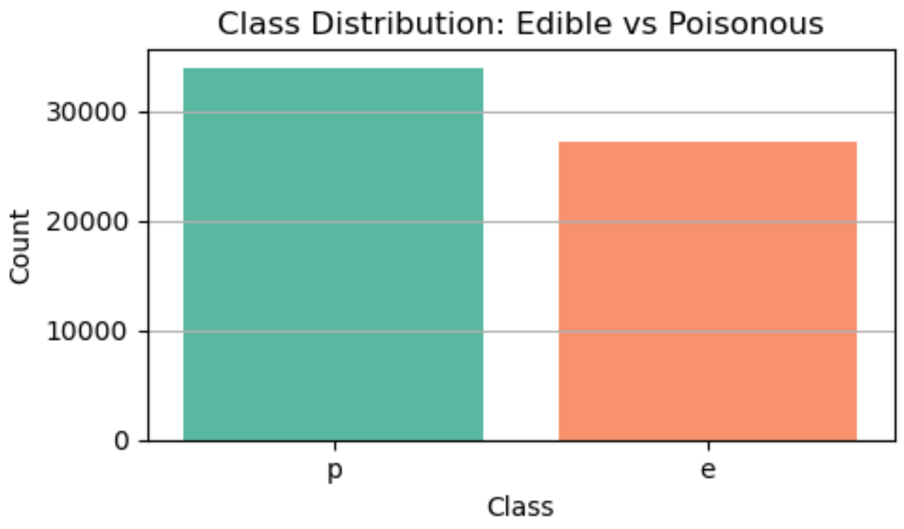


Figure 6. Class Distribution: Edible vs Poisonous

The class distribution plot shows the number of edible (e) and poisonous (p) mushrooms in the dataset. The dataset exhibits a slight class imbalance, with approximately 33,888 poisonous mushrooms and 27,181 edible mushrooms. Poisonous mushrooms make up about 55.5% of the dataset, while edible mushrooms comprise about 44.5%. This distribution underscores the importance of using stratified sampling during data splitting to ensure that both classes are proportionally represented in the training and testing sets.

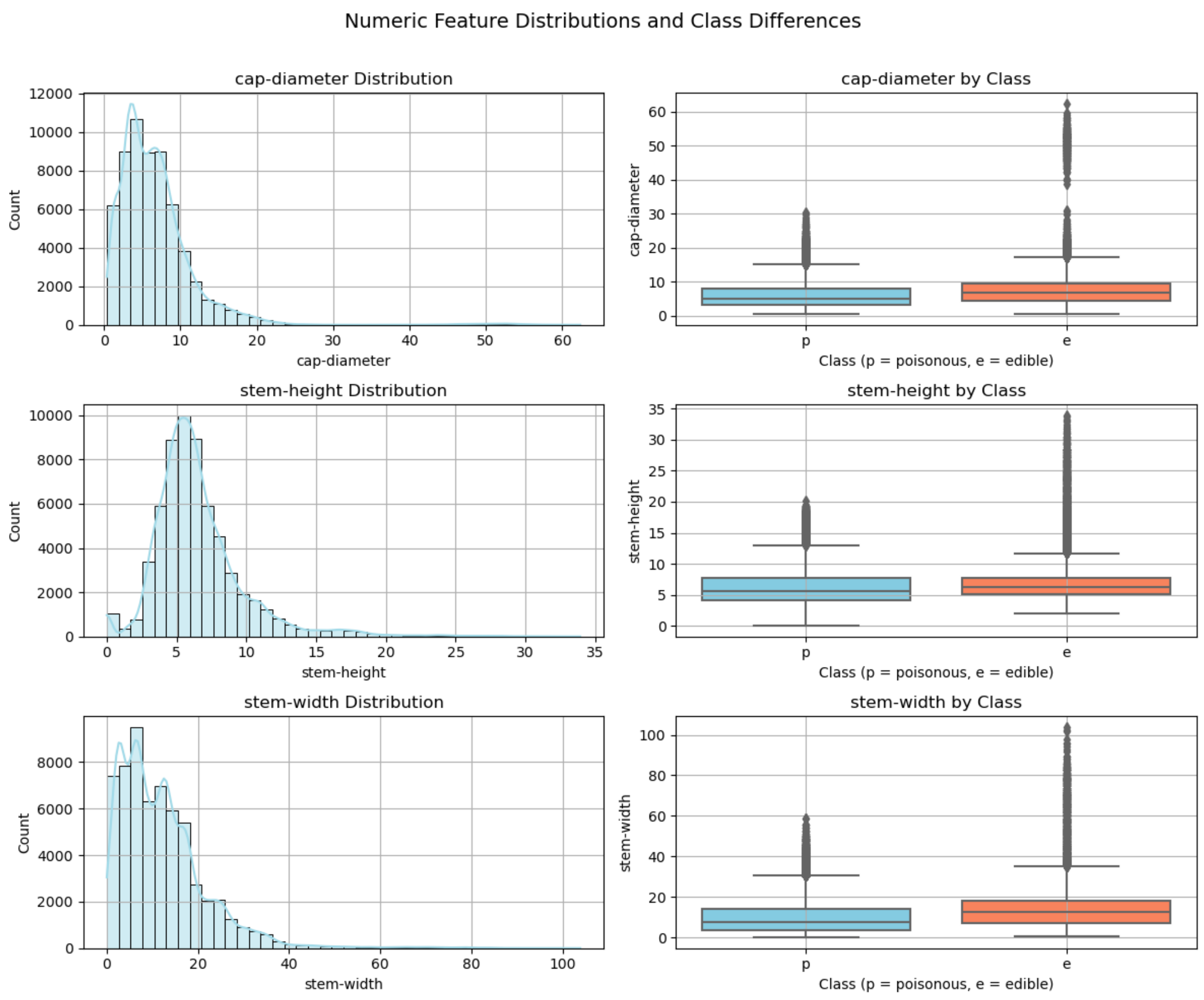


Figure 7. Numeric Feature Distributions and Class Differences

The distributions of the three numeric features (cap diameter, stem height, and stem width) were analyzed to understand their overall behavior and differences between edible and poisonous mushrooms.

- *Cap Diameter:* The distribution is right-skewed, with most mushrooms having caps between 2 cm and 10 cm. Both classes show similar patterns, although edible mushrooms tend to have slightly larger cap diameters on average.

- *Stem Height:* The stem height feature also shows a right-skewed distribution, with most values between 4 cm and 10 cm. There is a slight overlap between the classes, with edible mushrooms showing slightly higher median stem heights.

- *Stem Width:* Stem width displays the largest variability, with values ranging from less than 1 cm to over 70 cm. Edible mushrooms tend to have higher stem widths on average, but there is significant overlap between classes.

Boxplots for each feature by class highlight that none of the features alone perfectly separates the classes. While there are small differences in the medians and ranges, substantial overlap exists, suggesting that tree-based models will need to consider combinations of features to achieve accurate classification.

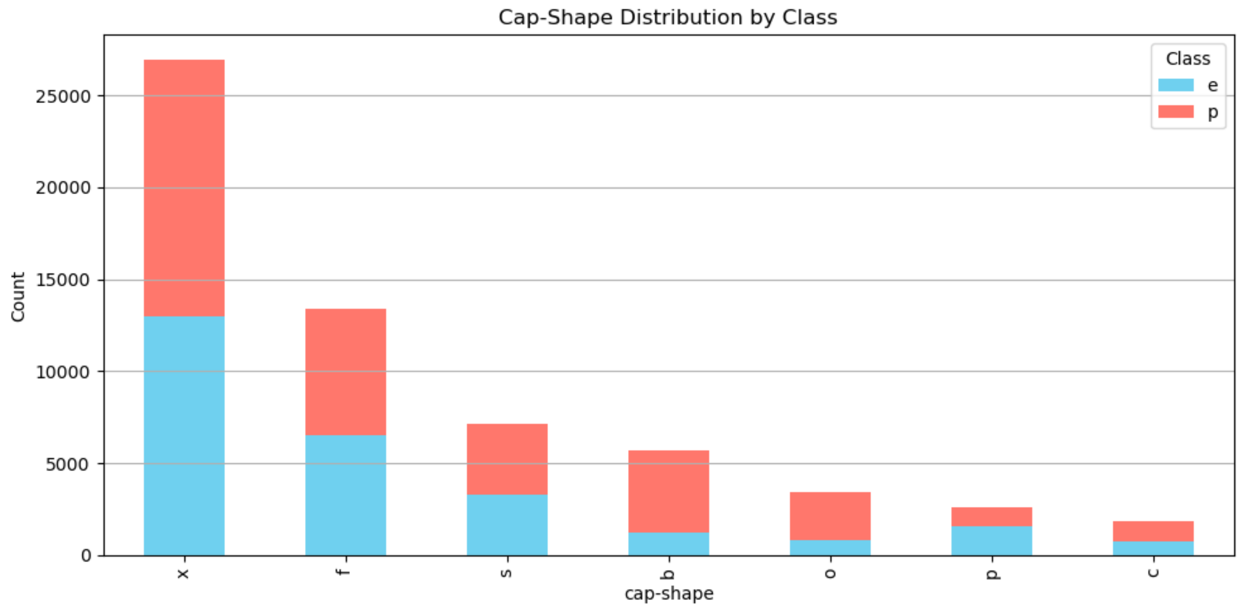


Figure 8. Cap-Shape Distribution by Class

The distribution of cap shapes among edible and poisonous mushrooms was analyzed. Convex-shaped caps (denoted as "x") were the most common in both classes, followed by flat ("f") and sunken ("s") shapes. Poisonous mushrooms showed higher counts overall across most cap shapes, especially in the convex and flat categories. Less common shapes such as bell ("b"), knobbed ("o"), and others were less frequent and showed similar distributions across classes. This analysis highlights that cap shape alone does not strongly differentiate between edible and poisonous mushrooms, reinforcing the need for multi-feature models in classification.

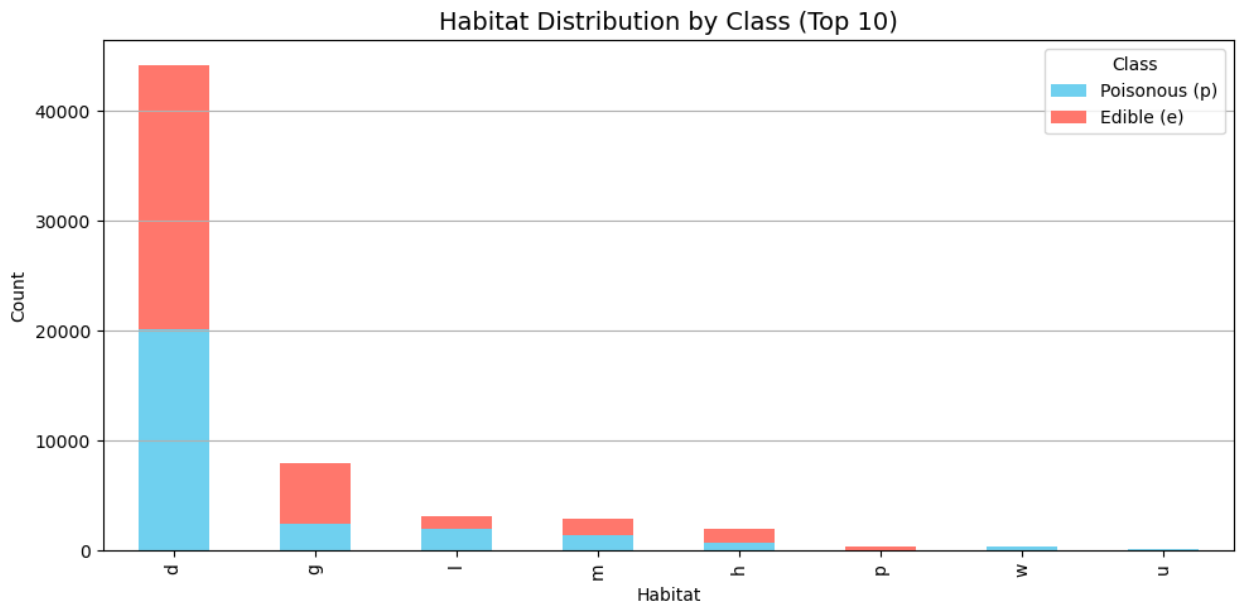


Figure 9. Habitat Distribution by Class

The habitat distribution plot illustrates the frequency of mushrooms in various habitats, split by class. The most common habitat overall is habitat "d," which contains a large proportion of both edible and poisonous mushrooms. Poisonous mushrooms are more frequent in every habitat, with particularly high counts in habitat "d." Other habitats such as "g," "l," "m," and "h" show lower counts but still include both classes. Some habitats like "w" and "u" are very rare and have minimal representation in the dataset. This distribution suggests that habitat alone does not fully separate edible from poisonous mushrooms, but may still contribute useful information when combined with other features in classification models.

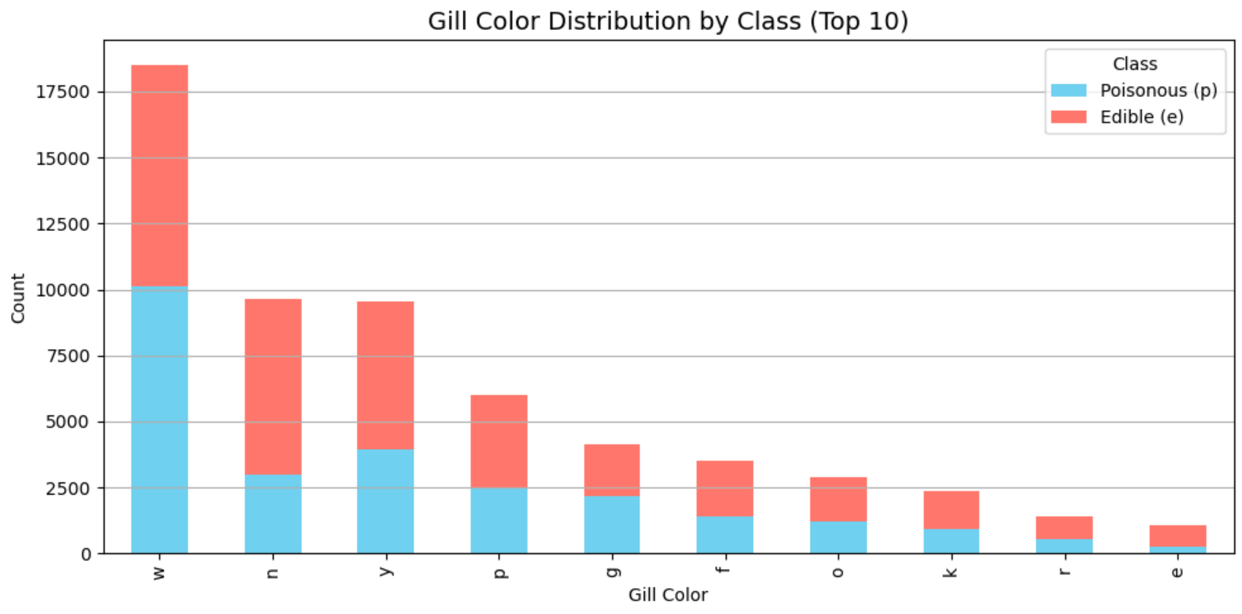


Figure 10. Gill Color Distribution by Class

The gill color distribution plot shows the frequency of different gill colors among edible and poisonous mushrooms. White ("w") is the most common gill color overall, appearing in both classes but slightly more frequently in the poisonous class. Other colors such as "n" (brown), "y" (yellow), and "p" (pink) also appear in both classes, with relatively even distribution but a slight tendency toward poisonous mushrooms. Less common gill colors like "g," "f," and "o" show lower counts. This analysis indicates that gill color alone is not a perfect discriminator but can provide valuable information when combined with other features in classification models.

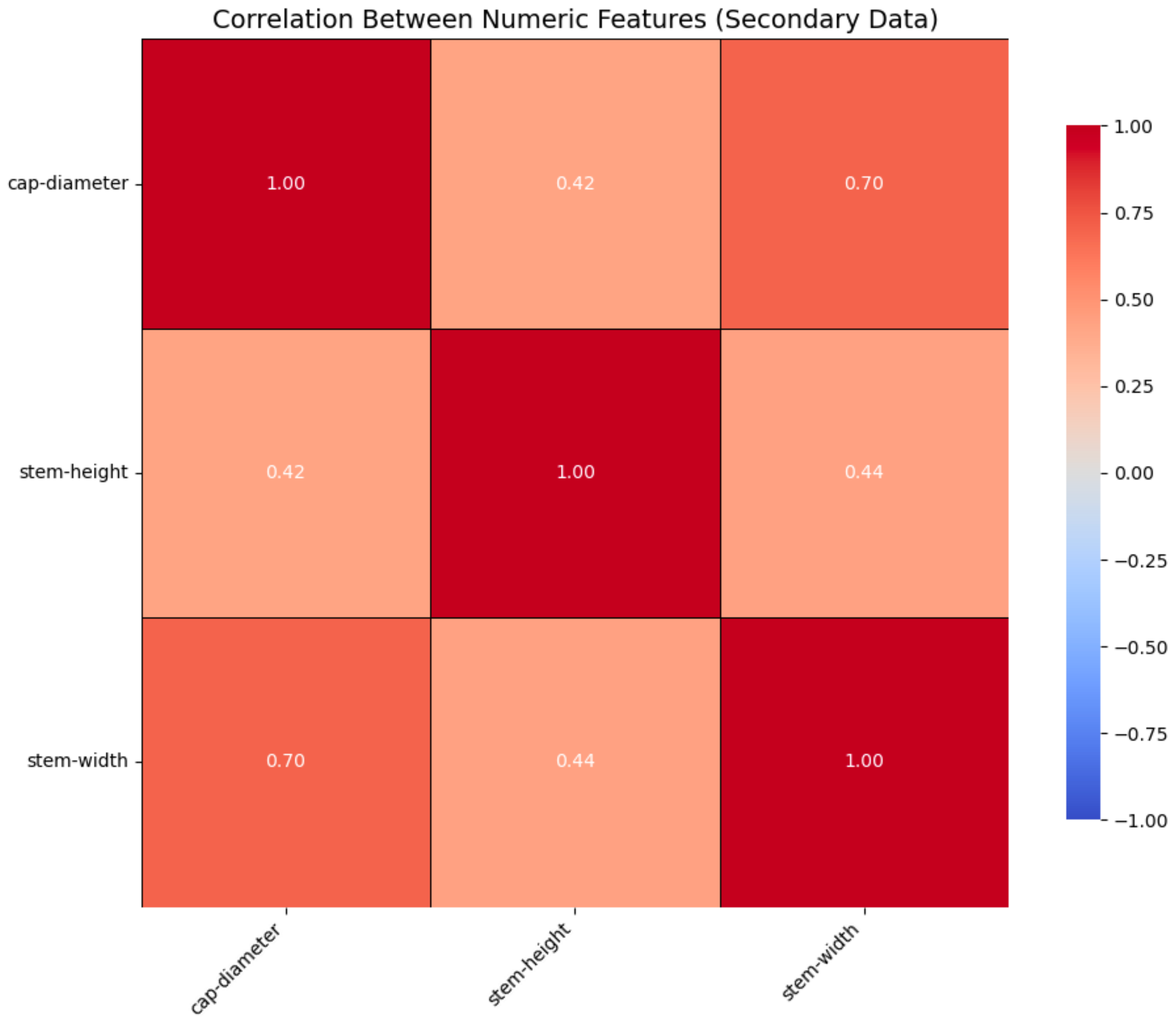


Figure 11. Correlation between Numeric Features

The correlation heatmap illustrates pairwise relationships among the three numeric features in the secondary dataset: cap diameter, stem height, and stem width. Cap diameter and stem width exhibit the strongest positive correlation (0.70), suggesting that mushrooms with larger caps often also have thicker stems. Moderate correlations are observed between cap diameter and stem height (0.42) and between stem height and stem width (0.44). None of the correlations are close to 1, indicating that each feature provides unique information that may help improve classification performance when used in a tree-based model.

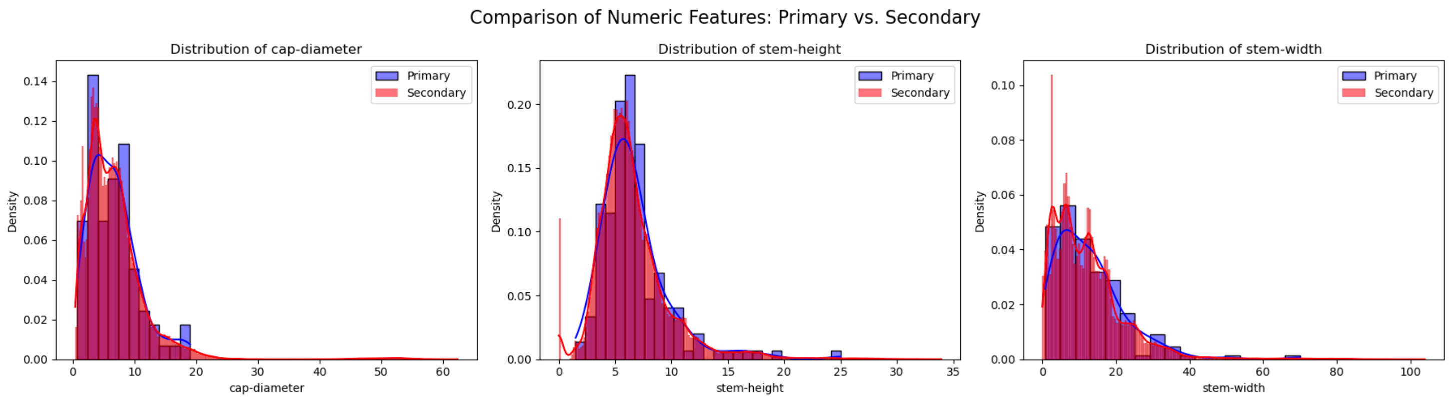


Figure 12. Comparison of Numeric Features: Primary vs Secondary

The density plots compare the distributions of cap diameter, stem height, and stem width between the primary and secondary datasets. Overall, the distributions are similar across datasets, with slight variations:

- *Cap Diameter:* The secondary dataset shows a slightly broader distribution with more samples at higher diameters, but both datasets are similarly right-skewed.
- *Stem Height:* Both datasets have consistent distributions, peaking around 5 to 7 cm, with no significant differences.
- *Stem Width:* The secondary dataset has a slightly longer tail, indicating the presence of a few outliers with larger stem widths compared to the primary dataset.

These similarities support using the secondary dataset for model training and evaluation, while acknowledging some differences that may influence model generalization.

Methodology

This section describes the step-by-step approach used to develop, train, and evaluate decision tree and random forest classifiers for predicting mushroom edibility. The methodology integrates data preparation, model construction, splitting and stopping criteria, hyperparameter tuning, and evaluation metrics. Each component is described in detail to provide a comprehensive overview of how the project was implemented from both a theoretical and practical standpoint.

The Mushroom dataset was preprocessed by converting the target variable into a binary format, where poisonous mushrooms were encoded as 1 and edible mushrooms as 0. All categorical features were label-encoded to facilitate numerical splitting. The dataset was divided into training, validation, and test sets using stratified sampling to maintain class balance across splits.

Decision Tree Implementation

The decision tree classifier was designed using two main classes: Node and TreePredictor. The Node class represents each point in the tree and includes attributes to indicate whether the node is a leaf, the predicted class label, the feature index and threshold for splitting, and references to the left and right child nodes. The TreePredictor class manages the construction and evaluation of the decision tree. It builds the tree recursively by evaluating all possible splits across all features at each node, calculating the impurity reduction using the chosen splitting criterion. The split with the highest impurity reduction is selected, and the data is divided into left and right subsets. This process repeats recursively until a stopping condition is reached. This approach mirrors the recursive process of building a decision tree, where each split progressively refines the data into increasingly homogeneous subsets, ultimately improving the model's classification accuracy.

To decide the best split at each internal node, three splitting criteria were implemented:

- *Gini Impurity.* Measures the probability of incorrectly classifying a randomly chosen element if it were randomly labeled according to the distribution of labels in the subset. Mathematically defined as $Gini(p) = 1 - \sum p_i^2$, where p_i is the proportion of class i in the node. This criterion is sensitive to node purity and is commonly used because it is computationally efficient and smooth.
- *Entropy.* Measures the uncertainty of the node in terms of class distribution. Defined as $Entropy(p) = -\sum p_i \, log_2 \, (p_i)$. A split is chosen to maximize the reduction in entropy (information gain). This criterion aligns with information theory and often favors balanced splits that reduce uncertainty in the data.
- *Classification Error.* Measures the fraction of misclassified samples at a node. Defined as $Error(p) = 1 - \max(pi)$. This criterion directly reflects the misclassification rate but may be less sensitive to subtle changes in class distribution compared to Gini and Entropy. It was included to compare its performance against the other two criteria.

During training, the algorithm evaluated all possible splits on each feature using these criteria and selected the split with the highest impurity reduction (information gain).

To control the complexity of the trees and reduce overfitting, the following stopping criteria were implemented:

- *Maximum Tree Depth.* Limits the number of levels the tree can grow, thereby constraining the model's capacity to memorize training data. This reduces variance and aligns with theoretical risk bounds that emphasize controlling model complexity.
- *Minimum Samples per Split.* Ensures that each internal node has a minimum number of samples before attempting to split. This prevents splits that could create leaves with very few data points, which might capture noise rather than true patterns.

These criteria directly reflect concepts from statistical learning theory, particularly the trade-off between model complexity and generalization error.

Decision trees were trained on the training set and evaluated on the validation set. Hyperparameter tuning was conducted using a grid search approach over the maximum depth and minimum samples per split, selecting the configuration that achieved the lowest validation error for each splitting criterion.

The best-performing models for each splitting criterion were evaluated on the test set using accuracy, confusion matrices, and classification reports to assess performance and highlight potential issues such as overfitting or underfitting.

Random Forest Implementation

A Random Forest classifier was implemented to enhance model performance by aggregating multiple decision trees into an ensemble. This method follows the principles of ensemble learning, where combining multiple base learners reduces variance and improves generalization compared to a single decision tree.

In the Random Forest implementation, each individual tree was trained on a bootstrap sample of the training data. Bootstrap sampling involves randomly selecting samples from the training set with replacement, creating diverse training subsets for each tree. This approach introduces variability among the trees and helps decorrelate their errors, thereby stabilizing the ensemble's predictions.

At each split within a tree, a random subset of features was considered instead of evaluating all features. This random feature selection at each node, also known as the Random Subspace Method, increases the diversity among trees by ensuring that different trees focus on different aspects of the data. This technique leverages randomization at both the data and feature levels, which enhances the diversity of the trees and contributes to the overall robustness of the ensemble.

Each tree was constructed using the same splitting criteria (Gini impurity, Entropy, or Classification Error) as the single decision trees, along with the same stopping criteria (maximum depth and minimum samples per split). This ensures consistency between the base learners and the stand-alone decision tree models, facilitating a fair comparison of performance.

The final prediction of the Random Forest was obtained through majority voting among the individual trees, where each tree casts a vote for a class label, and the class with the highest number of votes is selected as the final prediction. This voting mechanism reduces the risk of overfitting by balancing the biases of individual trees.

The Random Forest implementation in this project thus leverages both bagging (bootstrap aggregation) and random feature selection, aligning with ensemble learning theory and providing a robust, high-performing classifier for the mushroom classification task.

Experiments and Results

This section presents a series of experiments conducted to evaluate the performance of the custom decision tree classifiers implemented from scratch using the Mushroom dataset. The primary focus was to investigate how different splitting criteria (Gini impurity, Entropy, and Classification Error) and various hyperparameter settings affect model performance and generalization.

The experiments were carried out using the preprocessed secondary Mushroom dataset described in the Methodology section, where categorical features were label-encoded and the target variable was binarized. The data was partitioned into training, validation, and test sets using stratified sampling to maintain class balance.

For each splitting criterion, models were trained using a grid search approach over various hyperparameter configurations. As detailed in the Methodology section, the grid included maximum tree depths of 3, 5, 7, and 9, and minimum samples per split values of 2, 5, and 10. These values were selected to provide a systematic exploration of both shallow and deeper trees, allowing assessment of underfitting and overfitting tendencies.

Each model was trained on the training set, and performance was evaluated on the validation set using classification accuracy and zero-one loss. The configuration yielding the lowest validation error was chosen as the best model for each splitting criterion.

Following hyperparameter tuning, the best-performing models were evaluated on the held-out test set. Performance metrics such as test accuracy, classification reports, and confusion matrices were used to assess the models' generalization capabilities.

Gini Splitting Criterion

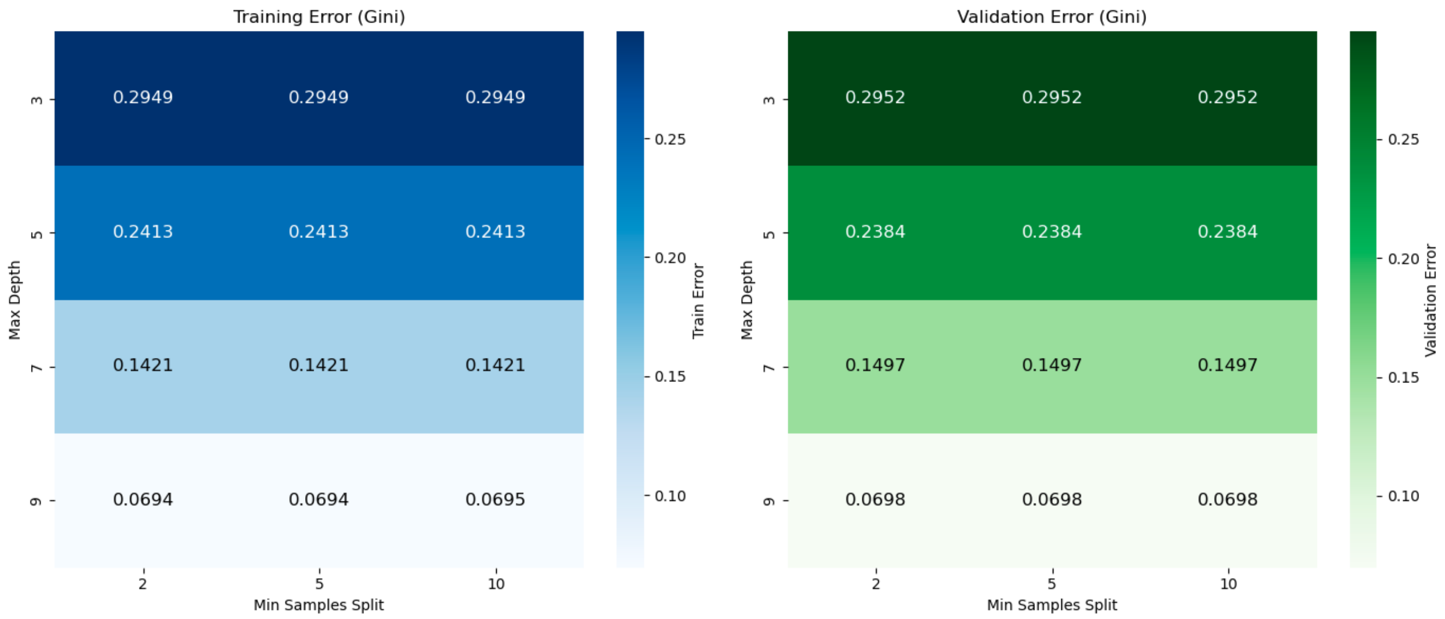
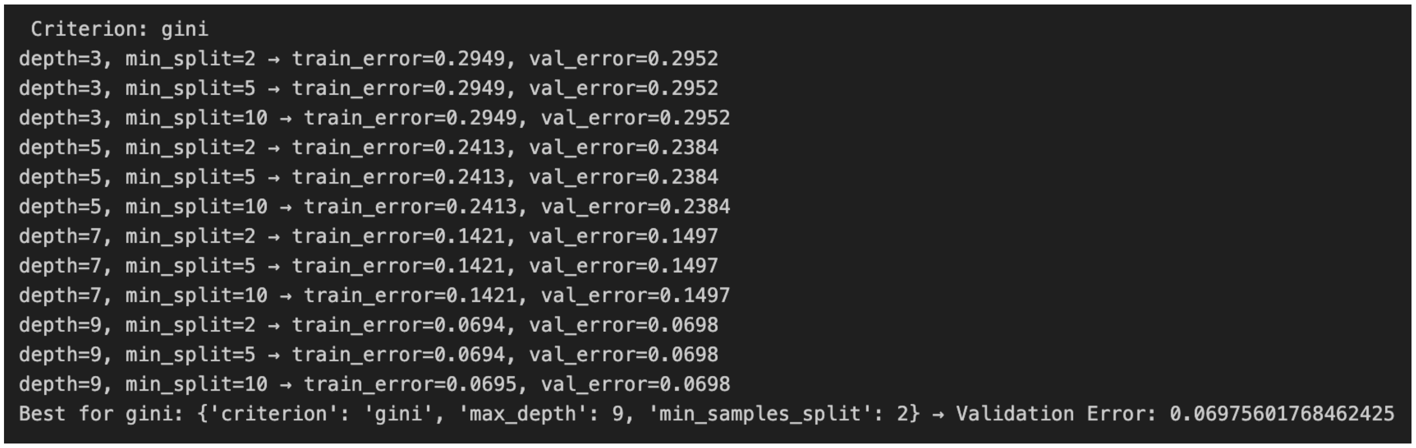


Figure 13. Training and Validation Error Heatmaps for Gini Splitting Criterion



```
 Criterion: gini
depth=3, min_split=2 → train_error=0.2949, val_error=0.2952
depth=3, min_split=5 → train_error=0.2949, val_error=0.2952
depth=3, min_split=10 → train_error=0.2949, val_error=0.2952
depth=5, min_split=2 → train_error=0.2413, val_error=0.2384
depth=5, min_split=5 → train_error=0.2413, val_error=0.2384
depth=5, min_split=10 → train_error=0.2413, val_error=0.2384
depth=7, min_split=2 → train_error=0.1421, val_error=0.1497
depth=7, min_split=5 → train_error=0.1421, val_error=0.1497
depth=7, min_split=10 → train_error=0.1421, val_error=0.1497
depth=9, min_split=2 → train_error=0.0694, val_error=0.0698
depth=9, min_split=5 → train_error=0.0694, val_error=0.0698
depth=9, min_split=10 → train_error=0.0695, val_error=0.0698
Best for gini: {'criterion': 'gini', 'max_depth': 9, 'min_samples_split': 2} → Validation Error: 0.06975601768462425
```

Table 2. Numerical Results of Training and Validation Errors for Gini Splitting Criterion

The plots summarize the training and validation errors for decision trees trained using the Gini impurity criterion. Both heatmaps show that increasing the maximum tree depth consistently reduced training error, from approximately 0.295 at depth 3 to around 0.069 at depth 9. Validation error followed a similar decreasing trend, reaching its lowest value of roughly 0.070 at depth 9.

Interestingly, the minimum samples per split parameter had little impact on either training or validation errors, as the values remained almost constant across different settings. This suggests that, for this dataset and criterion, tree depth was the dominant factor influencing performance.

Overall, these results indicate that deeper trees improved both training and validation accuracy without clear signs of overfitting at the evaluated parameter ranges, supporting the robustness of the Gini impurity criterion for this classification task. Notably, the best model was achieved with a maximum tree depth of 9 and a minimum samples per split of 2, yielding the lowest validation error of approximately 0.070.

Entropy Splitting Criterion

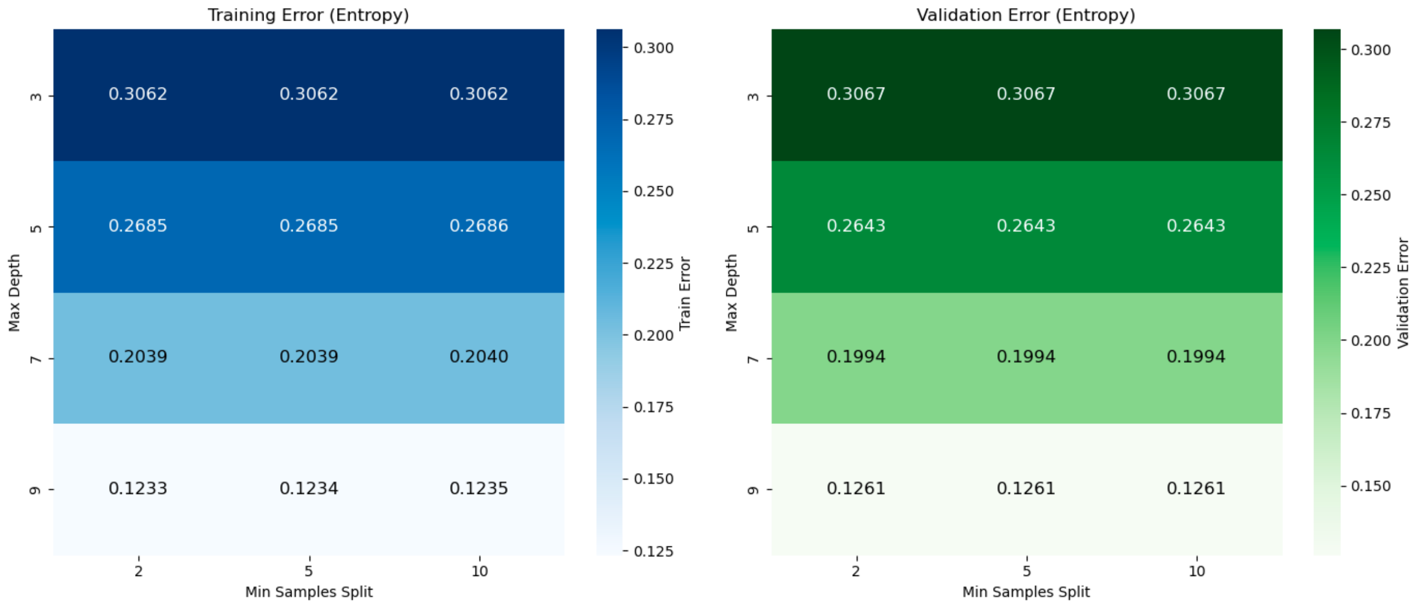
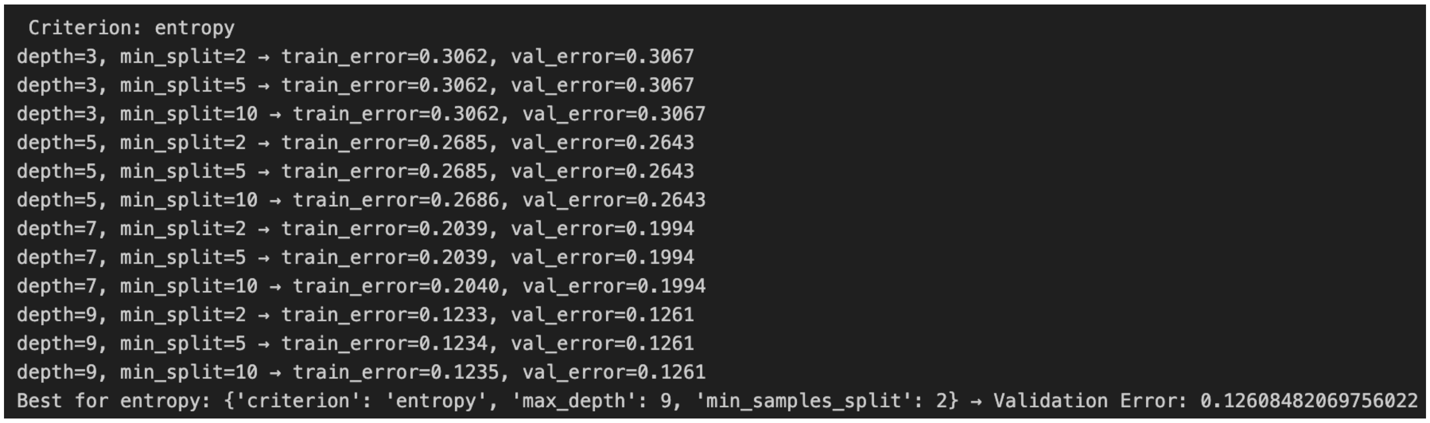


Figure 14. Training and Validation Error Heatmaps for Entropy Splitting Criterion



```
 Criterion: entropy
depth=3, min_split=2 → train_error=0.3062, val_error=0.3067
depth=3, min_split=5 → train_error=0.3062, val_error=0.3067
depth=3, min_split=10 → train_error=0.3062, val_error=0.3067
depth=5, min_split=2 → train_error=0.2685, val_error=0.2643
depth=5, min_split=5 → train_error=0.2685, val_error=0.2643
depth=5, min_split=10 → train_error=0.2686, val_error=0.2643
depth=7, min_split=2 → train_error=0.2039, val_error=0.1994
depth=7, min_split=5 → train_error=0.2039, val_error=0.1994
depth=7, min_split=10 → train_error=0.2040, val_error=0.1994
depth=9, min_split=2 → train_error=0.1233, val_error=0.1261
depth=9, min_split=5 → train_error=0.1234, val_error=0.1261
depth=9, min_split=10 → train_error=0.1235, val_error=0.1261
Best for entropy: {'criterion': 'entropy', 'max_depth': 9, 'min_samples_split': 2} → Validation Error: 0.12608482069756022
```

Table 3. Numerical Results of Training and Validation Errors for Entropy Splitting Criterion

The training and validation error heatmaps illustrate the performance of the decision tree classifier using the Entropy splitting criterion. As seen in the training error heatmap, increasing maximum tree depth reduced the training error from approximately 0.306 at depth 3 to around 0.123 at depth 9, indicating improved fit to the training data.

Validation error showed a similar decreasing trend, dropping from around 0.307 at depth 3 to about 0.126 at depth 9. This suggests that deeper trees enhanced the model's ability to generalize, at least within the evaluated parameter range.

The minimum samples per split parameter had minimal impact on either training or validation errors, as results were consistent across all tested values. Overall, the Entropy splitting criterion performed reliably across depths, with the best configuration identified at a maximum depth of 9 and minimum samples split of 2, yielding a validation error of approximately 0.126.

Classification Error Splitting Criterion

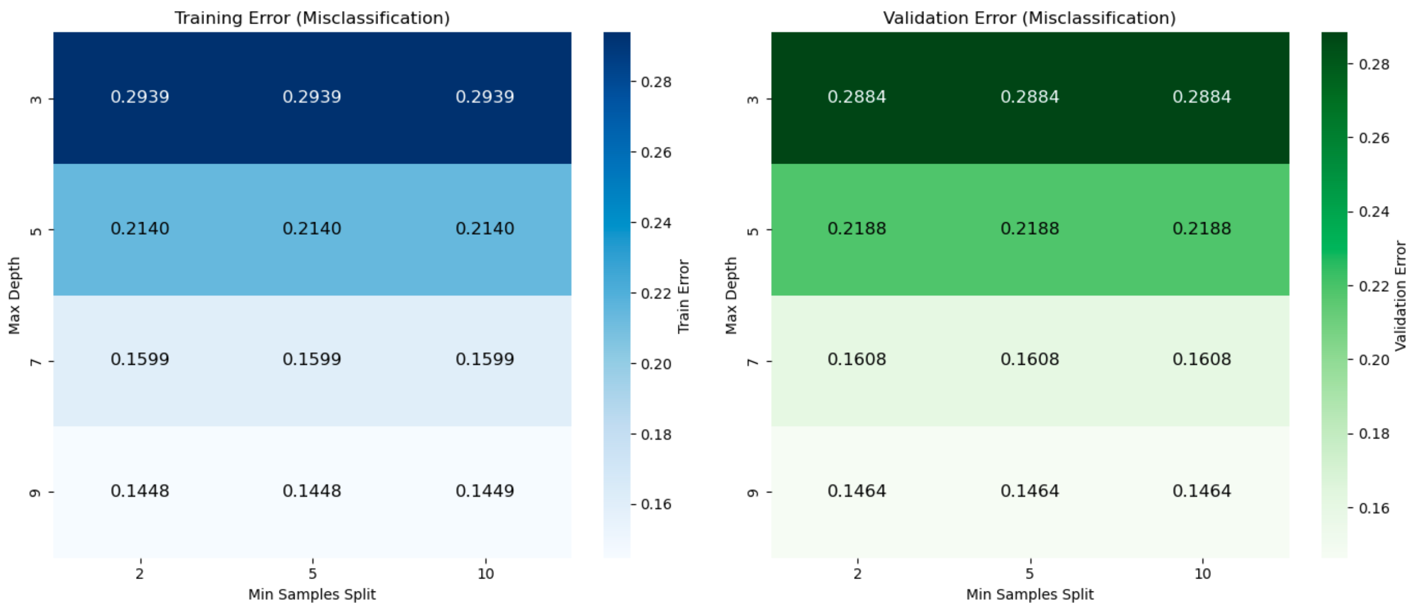
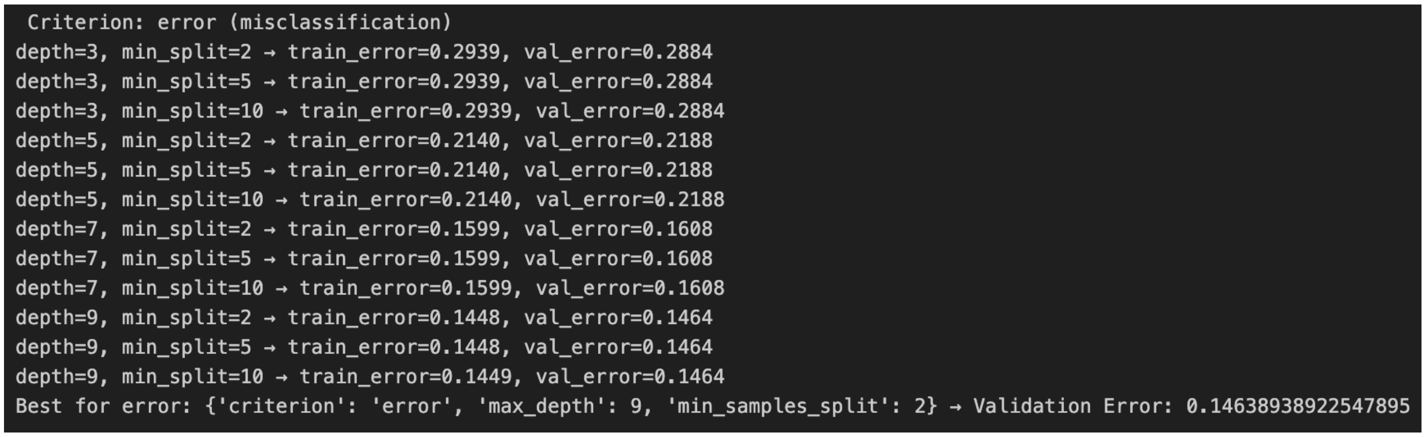


Figure 15. Training and Validation Error Heatmaps for Classification Error Splitting Criterion



```
 Criterion: error (misclassification)
depth=3, min_split=2 → train_error=0.2939, val_error=0.2884
depth=3, min_split=5 → train_error=0.2939, val_error=0.2884
depth=3, min_split=10 → train_error=0.2939, val_error=0.2884
depth=5, min_split=2 → train_error=0.2140, val_error=0.2188
depth=5, min_split=5 → train_error=0.2140, val_error=0.2188
depth=5, min_split=10 → train_error=0.2140, val_error=0.2188
depth=7, min_split=2 → train_error=0.1599, val_error=0.1608
depth=7, min_split=5 → train_error=0.1599, val_error=0.1608
depth=7, min_split=10 → train_error=0.1599, val_error=0.1608
depth=9, min_split=2 → train_error=0.1448, val_error=0.1464
depth=9, min_split=5 → train_error=0.1448, val_error=0.1464
depth=9, min_split=10 → train_error=0.1449, val_error=0.1464
Best for error: {'criterion': 'error', 'max_depth': 9, 'min_samples_split': 2} → Validation Error: 0.14638938922547895
```

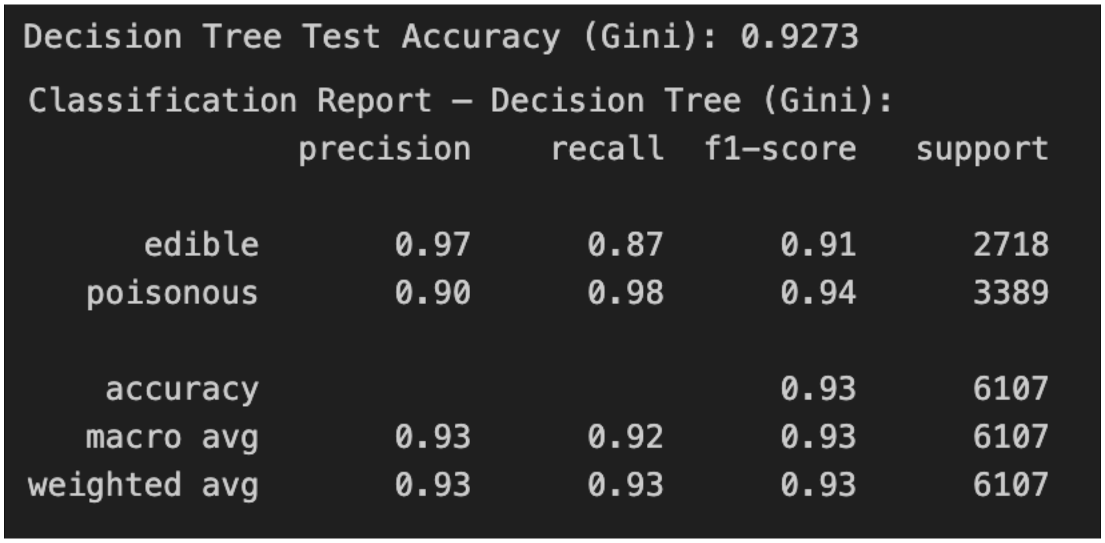
Table 4. Numerical Results of Training and Validation Errors for Classification Error Splitting Criterion

The training and validation error heatmaps illustrate the performance of the decision tree classifier using the Classification Error (misclassification) splitting criterion. As observed in the training error heatmap, increasing the maximum tree depth consistently decreased the training error—from approximately 0.294 at depth 3 to around 0.145 at depth 9—indicating that deeper trees improved the model's fit to the training data.

A similar trend appeared in the validation error heatmap, with the error decreasing from around 0.288 at depth 3 to approximately 0.146 at depth 9. This pattern suggests that deeper trees enhanced the model's ability to generalize without significant overfitting within the tested parameter range.

The minimum samples per split parameter again had minimal impact on both training and validation errors, as indicated by the uniformity of values across different settings. Overall, the Classification Error criterion demonstrated reliable performance, with the best configuration achieved at a maximum depth of 9 and a minimum samples split of 2, yielding a validation error of approximately 0.146.

Final Model Evaluation



```
Decision Tree Test Accuracy (Gini): 0.9273

Classification Report – Decision Tree (Gini):
              precision    recall  f1-score   support

      edible       0.97      0.87      0.91      2718
   poisonous       0.90      0.98      0.94      3389

    accuracy                           0.93      6107
   macro avg       0.93      0.92      0.93      6107
weighted avg       0.93      0.93      0.93      6107
```

Table 5. Decision Tree (Gini) — Test Accuracy and Classification Report

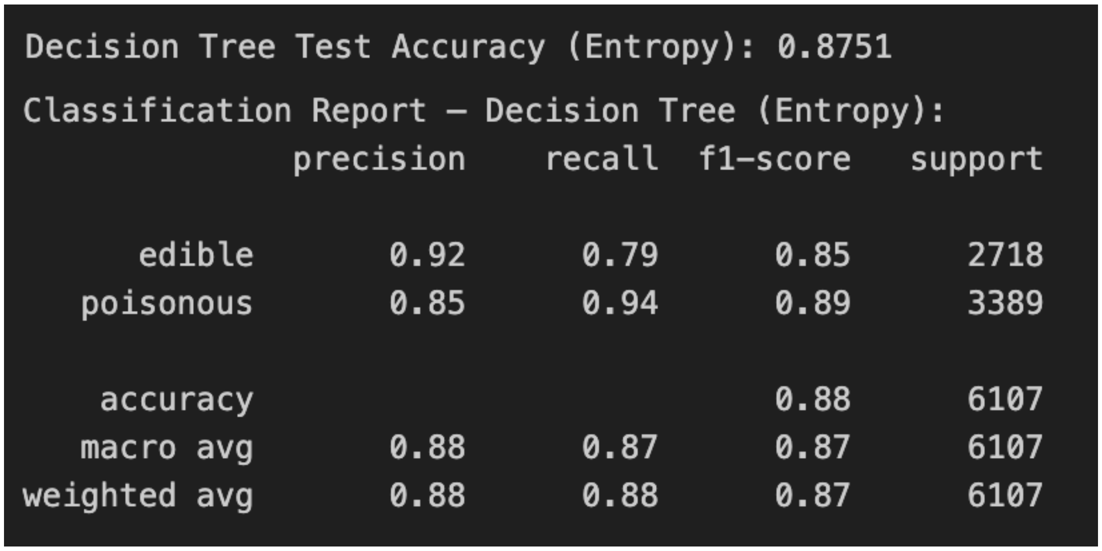
The decision tree model trained with the Gini impurity criterion achieved a test accuracy of approximately 0.927, indicating that the classifier correctly identified the edibility of mushrooms in about 93% of the test cases.

The classification report shows strong performance across both classes:

- *Edible* mushrooms: Precision of 0.97, recall of 0.87, and F1-score of 0.91. This means the model was highly precise in identifying edible mushrooms but missed some true edible samples, as indicated by the lower recall.

- *Poisonous* mushrooms: Precision of 0.90, recall of 0.98, and F1-score of 0.94. The model was very effective at capturing poisonous mushrooms, with a high recall indicating it rarely missed actual poisonous samples.

Overall, the macro and weighted averages both yielded F1-scores of approximately 0.93, indicating balanced performance between classes. The model's high precision and recall values demonstrate its effectiveness at correctly distinguishing between edible and poisonous mushrooms, with particular strength in identifying poisonous mushrooms.



```
Decision Tree Test Accuracy (Entropy): 0.8751

Classification Report – Decision Tree (Entropy):
              precision    recall  f1-score   support

      edible       0.92      0.79      0.85      2718
   poisonous       0.85      0.94      0.89      3389

    accuracy                           0.88      6107
   macro avg       0.88      0.87      0.87      6107
weighted avg       0.88      0.88      0.87      6107
```

Table 6. Decision Tree (Entropy) — Test Accuracy and Classification Report

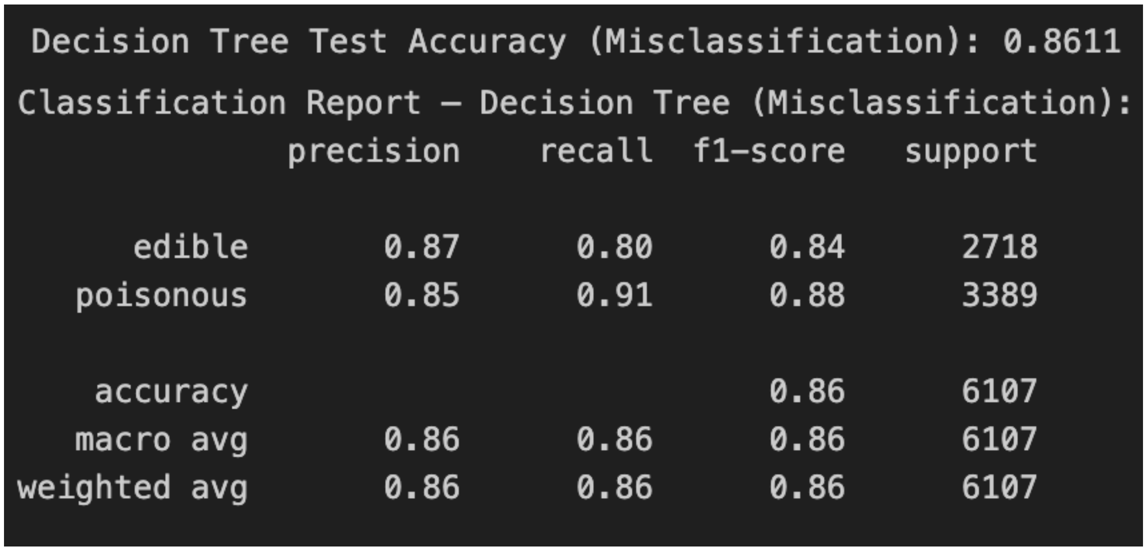
The final decision tree classifier trained with the Entropy splitting criterion achieved a test accuracy of approximately 0.875, indicating reliable performance in predicting mushroom edibility on unseen data.

The classification report shows solid metrics for both edible and poisonous classes:

- *Edible* mushrooms*:* Precision of 0.92, recall of 0.79, and F1-score of 0.85. This indicates the model was highly precise in identifying edible mushrooms, though it missed a small proportion of true edible samples, as reflected in the recall.

- *Poisonous* mushrooms*:* Precision of 0.85, recall of 0.94, and F1-score of 0.89, highlighting strong sensitivity in identifying poisonous mushrooms—a crucial factor for safety-related applications.

Overall, both macro and weighted averages achieved F1-scores of approximately 0.87–0.88, demonstrating balanced performance across classes. These results confirm that the Entropy-based decision tree effectively captures class distributions and performs well on the test set, although slightly lower than the Gini-based model.



```
 Decision Tree Test Accuracy (Misclassification): 0.8611
Classification Report – Decision Tree (Misclassification):
              precision    recall  f1-score   support

      edible       0.87      0.80      0.84      2718
   poisonous       0.85      0.91      0.88      3389

    accuracy                           0.86      6107
   macro avg       0.86      0.86      0.86      6107
weighted avg       0.86      0.86      0.86      6107
```

Table 7. Decision Tree (Classification Error) — Test Accuracy and Classification Report

The final decision tree classifier using the Classification Error splitting criterion achieved a test accuracy of approximately 0.861, indicating good performance on the test set for distinguishing between edible and poisonous mushrooms.

The classification report highlights the model's precision, recall, and F1-scores for both classes:

- *Edible* mushrooms: Precision of 0.87, recall of 0.80, and F1-score of 0.84. This indicates the model was fairly precise in identifying edible mushrooms, though it missed some edible samples, as reflected by the lower recall.

- *Poisonous* mushrooms: Precision of 0.85, recall of 0.91, and F1-score of 0.88. The model demonstrated a strong ability to capture most poisonous mushrooms, which is critical for safety-related applications.

The macro and weighted averages both achieved F1-scores of approximately 0.86, indicating balanced performance across classes. Overall, these results confirm that the Classification Error-based decision tree performs effectively on the test set, though with slightly lower accuracy compared to the Gini and Entropy models.

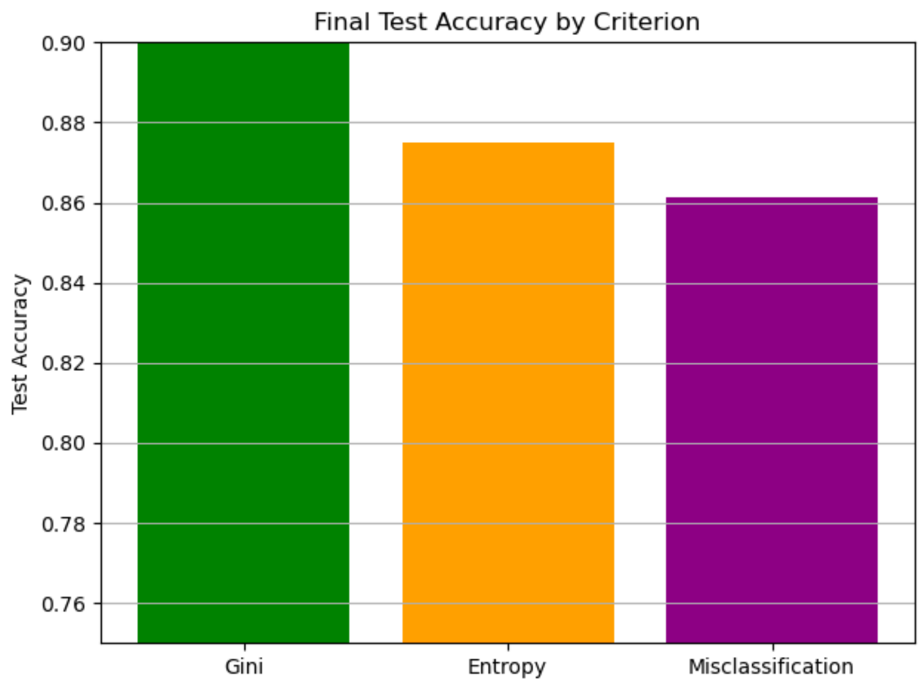


Figure 16. Comparison of Test Accuracy Across Splitting Criteria

The bar chart illustrates the final test accuracy achieved by the decision tree classifiers trained with different splitting criteria: Gini, Entropy, and Classification Error. The *Gini* criterion achieved the highest test accuracy at approximately 0.93, indicating the strongest overall performance in distinguishing between edible and poisonous mushrooms on the test set. These results suggest that the Gini impurity splitting criterion offers a small but consistent advantage in terms of test set accuracy for this particular mushroom classification task.

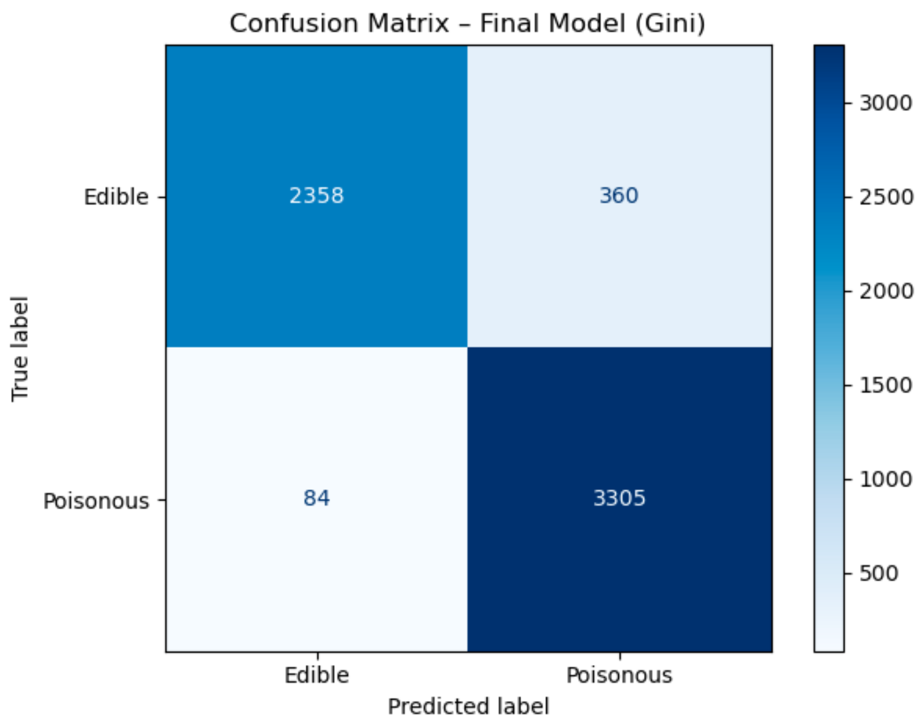


Figure 17. Confusion Matrix for Decision Tree Classifier (Gini)

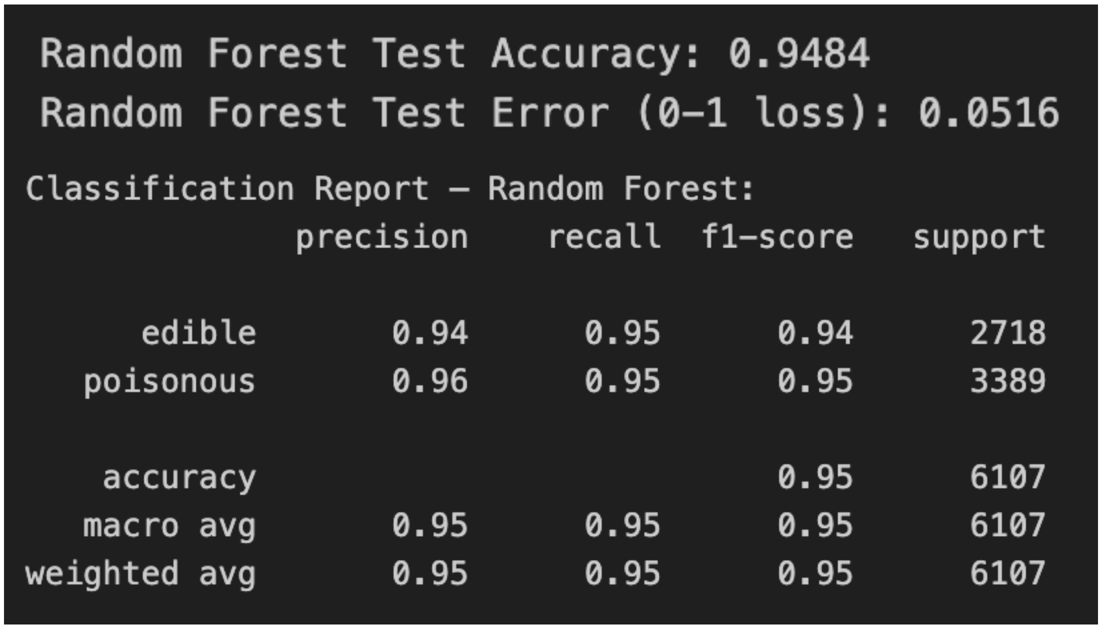
The confusion matrix for the final Decision Tree model trained with the Gini criterion illustrates the model's performance in classifying edible and poisonous mushrooms on the test set.

- The model correctly identified 2358 edible mushrooms (true negatives) and 3305 poisonous mushrooms (true positives).
- 360 edible mushrooms were incorrectly classified as poisonous (false positives), while 84 poisonous mushroomswere misclassified as edible (false negatives).

These results reflect the model's strong overall performance, with especially high accuracy in identifying poisonous mushrooms. However, the higher count of false positives compared to false negatives suggests a conservative bias that prioritizes safety over false reassurance.

Random Forest Performance

The Random Forest model, trained on the secondary Mushroom dataset using the Gini splitting criterion, was evaluated on the test set to assess its classification performance. The Gini criterion was chosen because it is computationally efficient and sensitive to class impurity, making it particularly effective for evaluating splits in categorical data such as the mushroom dataset. The results presented below highlight the model's test accuracy, error rate (0–1 loss), and a detailed classification report summarizing precision, recall, and F1-scores across both classes.



```
Random Forest Test Accuracy: 0.9484
Random Forest Test Error (0–1 loss): 0.0516

Classification Report – Random Forest:
              precision    recall  f1-score   support

      edible       0.94      0.95      0.94      2718
   poisonous       0.96      0.95      0.95      3389

    accuracy                           0.95      6107
   macro avg       0.95      0.95      0.95      6107
weighted avg       0.95      0.95      0.95      6107
```

Table 8. Random Forest (Gini) — Test Accuracy and Classification Report

The Random Forest achieved an impressive test accuracy of 0.9484 with a test error (0–1 loss) of 0.0516, indicating a highly accurate classifier with minimal misclassification. The classification report shows strong precision and recall values for both classes:

- Edible mushrooms were classified with a precision of 0.94 and a recall of 0.95, yielding an F1-score of 0.94.

- Poisonous mushrooms achieved slightly higher precision (0.96) and an equally strong recall (0.95), reflecting excellent model sensitivity and specificity for the crucial poisonous class.

Overall, the model consistently performed well across both classes, with macro and weighted averages of 0.95 for precision, recall, and F1-score. These results underscore the effectiveness of the Random Forest in capturing patterns and generalizing to unseen data, confirming its robustness and reliability in the mushroom classification task.

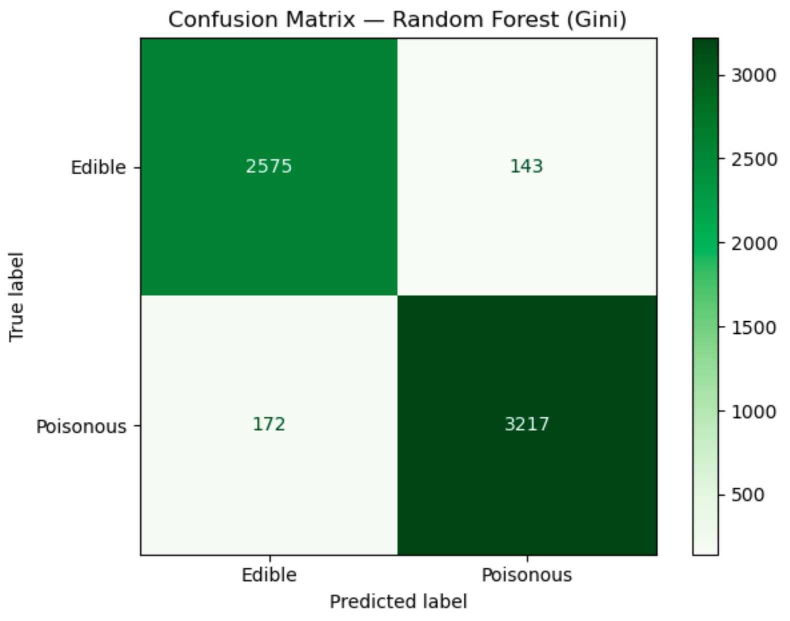


Figure 18. Confusion Matrix for Random Forest (Gini)

The confusion matrix demonstrates that the Random Forest model using the Gini splitting criterion performs well at distinguishing between edible and poisonous mushrooms. The model correctly classified 2575 edible mushrooms and 3217 poisonous mushrooms. However, there were 143 edible mushrooms incorrectly classified as poisonous (false positives) and 172 poisonous mushrooms incorrectly classified as edible (false negatives). These misclassifications highlight areas where the model could potentially be improved, such as fine-tuning hyperparameters or using additional features. Overall, the high number of correct predictions indicates the model's strong ability to classify mushrooms accurately, consistent with the high accuracy reported earlier.

Model Comparison

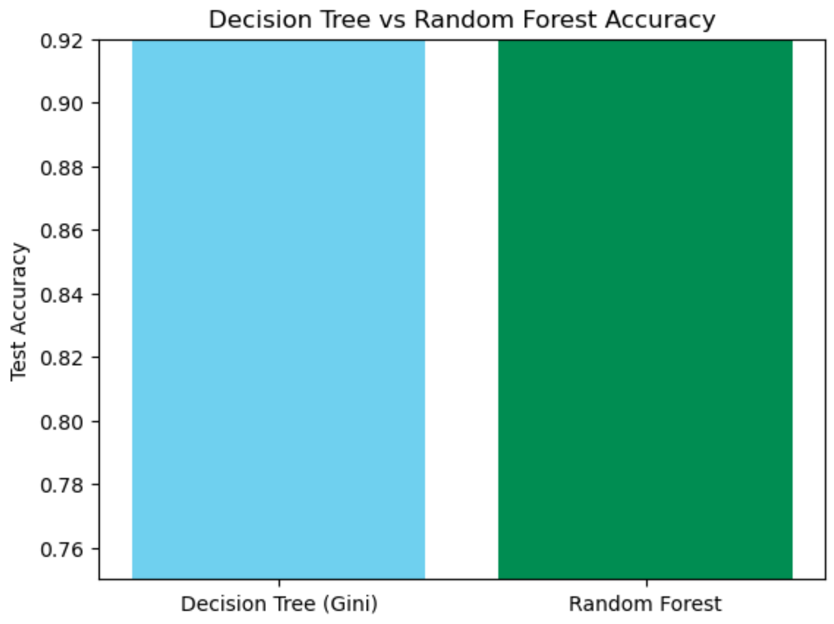


Figure 19. Model Comparison

The bar chart compares the final test accuracy achieved by the best-performing decision tree (using the Gini criterion) and the random forest ensemble classifier. The random forest achieved higher accuracy (approximately 0.95) compared to the single decision tree (approximately 0.93). This improvement highlights the effectiveness of the ensemble approach in enhancing classification performance by reducing variance and capturing a broader range of patterns in the data.

This project implemented and compared decision tree and random forest classifiers to predict mushroom edibility using a secondary mushroom dataset. Through a systematic evaluation of splitting criteria (Gini impurity, entropy, and classification error) and different hyperparameter settings (maximum depth and minimum samples per split), it was possible to understand the impact of these choices on model performance.

Discussion

This project compared decision tree and random forest classifiers for predicting mushroom edibility. The decision tree model using the Gini impurity criterion achieved a test accuracy of approximately 92.7%, demonstrating good performance in classifying both edible and poisonous mushrooms. Random forest further improved upon this, achieving a higher test accuracy of 94.8%, thanks to its ensemble learning approach that reduced variance and improved overall predictive power.

Observations on Overfitting and Underfitting

During hyperparameter tuning, varying maximum depth and minimum samples per split highlighted the risk of both overfitting and underfitting. Shallow trees (e.g., maximum depth = 3) exhibited relatively high validation error, indicating underfitting. They failed to capture the complexity of the data. Conversely, deeper trees (e.g., maximum depth = 9) improved training accuracy substantially, but without a significant increase in validation error, suggesting no substantial overfitting within the evaluated parameter ranges. Implementing stopping criteria, particularly limiting maximum depth and enforcing a minimum number of samples per split, effectively balanced model complexity and generalization performance. This approach helped prevent overfitting while maintaining good predictive performance.

Comparison of Different Splitting Criteria

Among the splitting criteria tested, Gini impurity consistently outperformed entropy and classification error in terms of validation accuracy and overall model stability. Entropy produced comparable but slightly lower performance, showing more sensitivity to deep tree structures. Classification error yielded the weakest performance, as it was less sensitive to subtle class distribution differences. Overall, Gini impurity demonstrated strong robustness and computational efficiency, justifying its use as the primary splitting criterion in both the decision tree and random forest models.

Effectiveness of Random Forest Compared to Decision Tree

The random forest model achieved a test accuracy of 94.8%, clearly outperforming the single decision tree. This improvement is attributed to random forest's use of bootstrap sampling and random feature selection, which mitigated the risk of overfitting and provided better generalization. Random forests combine the predictions of multiple, slightly different trees, reducing variance and stabilizing predictions compared to individual decision trees. This ensemble approach makes random forest a more reliable classifier in real-world applications.

Limitations of the Current Models and Implementation

While both the decision tree and random forest models performed well, there are several limitations to the current approach. First, the models relied on simple train-validation-test splitting rather than using more robust k-fold cross-validation, which could provide a better estimate of model performance. Additionally, no feature importance analysis was performed, which could offer valuable insights into which attributes most influence classification. Moreover, no pruning strategies were implemented in the decision tree, which could potentially enhance generalization and reduce overfitting in deeper trees.

Potential Future Work

Future work could address these limitations by incorporating pruning methods to control tree complexity and improve interpretability. Implementing cross-validation would provide a more reliable assessment of model performance and help refine hyperparameter tuning. Additionally, exploring other ensemble methods such as boosting or bagging variants could further enhance model accuracy and robustness. Incorporating feature importance analysis and interpretability tools would also help to better understand the key drivers of mushroom classification, adding transparency to the models' decisions.

Conclusion

This project implemented and evaluated decision tree and random forest classifiers to predict mushroom edibility using a custom-built tree implementation. Through careful data preprocessing, hyperparameter tuning, and testing different splitting criteria, the study demonstrated that decision trees using the Gini impurity criterion achieved strong classification performance with a test accuracy of approximately 92.7%. Furthermore, the random forest model surpassed the single decision tree, achieving a higher accuracy of 94.8%, highlighting the power of ensemble learning in reducing variance and improving generalization.

The experiments confirmed that deeper trees with controlled stopping criteria improved performance without significant overfitting, emphasizing the importance of managing model complexity. Among the splitting criteria, Gini impurity proved the most effective, offering robust performance and computational efficiency.

The project also highlighted the limitations of the current approach, including the absence of cross-validation, feature importance analysis, and pruning strategies. Future work should address these aspects to further refine model performance and enhance interpretability. Incorporating advanced ensemble methods and cross-validation techniques could also lead to additional improvements.

Overall, the project successfully demonstrated the applicability of decision trees and random forests to a real-world classification task, providing valuable insights into the influence of different modeling decisions on predictive accuracy and model robustness.