UNIVERSITAT POLITÈCNICA DE CATALUNYA BARCELONATECH

Combining multiple classifiers - CART with Random Forest or Decision Forest

Universitat Politècnica de Catalunya
Facultad de Informática
Master Artificial Intelligence

Project Report
for "Supervised and Experiential Learning"

written by

Karl - Augustin Jahnel
karlaugustin.jahnel@estudiantat.upc.edu

May 2024

Professor: Salvador MIQUEL SANCHEZ MARRE assig-SEL-MAI@fib.upc.edu

Contents

1 Introduction

Random Forests offer an effective solution for complex classification and regression problems, making them indispensable in sectors such as finance and healthcare where robustness against overfitting and the ability to handle high-dimensional data are critical. Simultaneously, Decision Forests provide reliable, interpretable insights essential for dynamic environments, such as stock market analysis and fraud detection, where efficient management of noisy, large datasets is paramount. Implementing basic Machine Learning Algorithms provide a good opportunity to enhance the understanding of the Algorithms in detail. In this report I delve into "Ensemble Classifiers". Ensemble classifiers use the combination of multiple base classifier to tackle the dilemma of accuracy optimization and over-adaptation. I am going to compare the "Random Forest"Breiman (2001) and "Decision Forest" Ho (1998) Method. As a Base-Classifier I use CART (classification and regression trees) Breiman et al. (1984). As a splitting function I use "Gini Impurity" I implemented them from scratch in Python by only using basic libraries such as pandas and random and math. As these are "parallel machine learning algorithms" I implemented the building of different trees with parallelization. The report can be seen in original at CLick this Overleaf Link. The original code I provide on Github at Click here to get to my Github Repo.

2 Classification and Regression Tree

I used the typical Decision Tree implementation. This means I implemented a binary tree structure, which consists of Nodes and Edges and Each Node can have at most one parent node and can have two child nodes. A node can be the root node, which is the top-most node where the decision tree starts. It represents the entire dataset, which is then split into two or more homogeneous sets. A node can also be an internal node which are nodes where the splitting of the dataset continues based on a certain condition. Each internal node tests a specific attribute or feature, leading to branches that represent the outcomes of the test. A node can also be a leaf node represent the final output of the decision process. In classification trees, they represent the class label, while in regression trees, they represent a continuous value. I represented this data structure in classes. I implemented a "Condition class" which holds the feature and the threshold and methods for deciding and handling if its a numeric feature or a categorical one. A single node I represented also as a class. It contains a "Condition" and a left (true) child and a right (false) child and it can also contain a label which is the class. If it represents a LeafNode it has "None" Children and it contains a label. The following figures show the complete Decision Trees for the datasets I used. Further information about the datasets can be seen under the section **Results and discussion**. I wrote a visualization function which uses the graphviz library to parse a decision tree into a png.

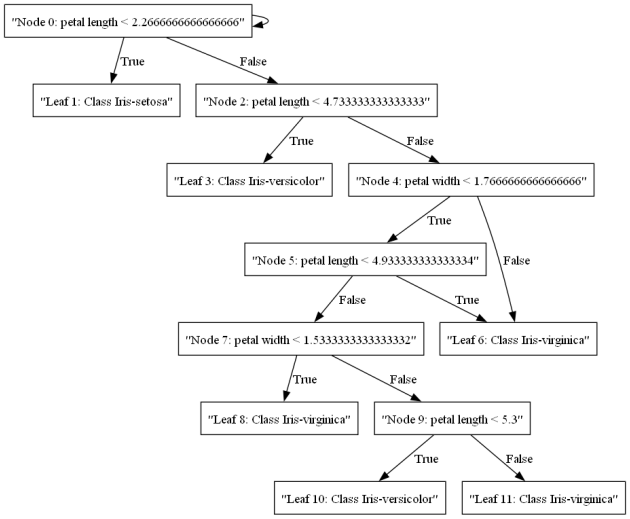

Figure 1: Decision tree of Iris dataset

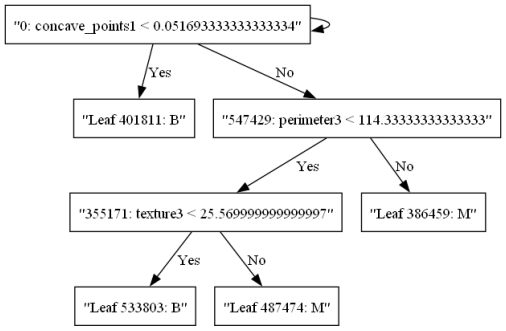

Figure 2: Decision tree of Wisconsin Breast Cancer dataset

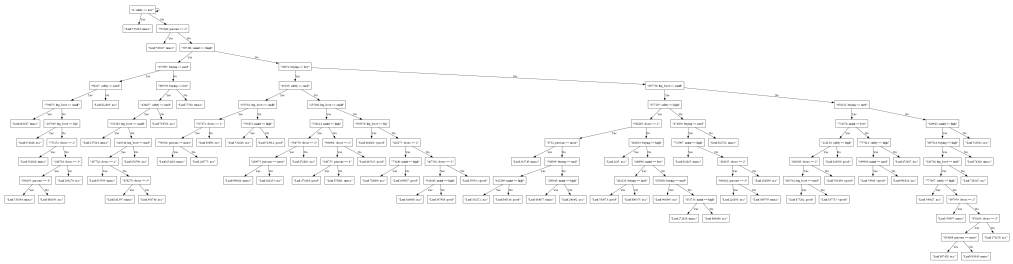

Figure 3: Decision tree of Car dataset

there you can see, that in a complete decision tree, complete meaning it takes all features into account, that the trees get very big and deep.

2.1 Gini Impurity

In the building process of a decision tree it is a neccessity to decide how to split the dataset in the next step. In the Paper of Ho (1998) it suggests the following splitting funcitons: "For instance, single-feature splits can be chosen by Sethi and Sarvarayudu's average mutual information [28], the Gini index proposed by Breiman et al. [3], Quinlan's information gain ratio [25], or Mingers's G statistic [20], [21], etc." I decided for the Gini index function as a splitting function: where the variables are defined as follows:

$$G = \frac{|L|}{|L|+|R|}G_L + \frac{|R|}{|L|+|R|}G_R$$

Figure 4: Gini index

- $|L|$ - Size of the left subset, used to weight its Gini impurity in the calculation of the total Gini index.
- $|R|$ - Size of the right subset, used to weight its Gini impurity in the calculation of the total Gini index.
- G_L - Gini impurity of the left subset, measuring the homogeneity of class labels in this subset.
- G_R - Gini impurity of the right subset, measuring the homogeneity of class labels in this subset.

The Gini Impurity is defined as follows:

$$I = 1 - \sum_{i=1}^{J} \left(\frac{\text{count}_i}{N}\right)^2$$

Figure 5: Gini impurity

where the variables are defined as follows:

- I - Represents the Gini impurity of the dataset. It measures the impurity or purity of a set of elements, where 0 indicates perfect purity (all elements are of the same class), and higher values indicate higher diversity of classes.
- count_i - The number of occurrences of class i within the dataset. It is used to calculate the probability of picking an element of class i.
- N - The total number of samples in the dataset. This is the denominator for calculating the probability of each class.
- J - The total number of different classes in the dataset. The summation runs over all these classes, ensuring each class's contribution to the impurity is considered.

2.2 Tree building Algorithm

The following pseudocode represents the implemented Algorithm for building a tree:

Algorithm 1: Build Decision Tree Algorithm

```
Function build_tree(data, numFeatures, impurity_threshold,
 decision_tree_features):
    if gini ==0 then
        mostCommonClass ← extractMostCommonClassfromDataset;
        return LeafwithMostCommonClass;
    end
    best_condition ← get_best_split(data, numFeatures, decision_tree_features);
    if best_condition is None then
        mostCommonClass ← extractMostCommonClassfromDataset;
        return LeafwithMostCommonClass;
    end
    node ← Node(best_condition);
    left_data, right_data ← data_split(data, best_condition);
    node.left ← build_tree(left_data, numFeatures, impurity_threshold,
     decision_tree_features);
    node.right ← build_tree(right_data, numFeatures, impurity_threshold,
     decision_tree_features);
    return node;
```

Base Cases for Recursion:

- Empty Data or Low Impurity: If the dataset is empty or its impurity is below the threshold, it returns a node marked as a leaf with the most common class label.
- No Suitable Split: If no valid condition to split the data is found, it again returns a node as a leaf with the most common class label.

Recursive Case:

- Node Creation: If a suitable split is found, it creates a new node based on the best condition.
- Splitting Data: It then splits the data into left and right subsets based on the best condition.
- Recursive Calls: Recursively builds the left and right subtrees by calling buildtree on the subsets.

The algorithm begins by initializing two variables: *bestGini* to infinity, which will hold the lowest Gini impurity found, and *bestCondition*, which will store the condition (feature and value) that achieves this impurity.

- If the number of features to consider (*numFeatures*) is specified, a random subset of these features is selected. Otherwise, the function checks if a predefined set of features (*decision_forest_features*) is provided. If neither is specified, all features (except the target variable) are used.

Algorithm 2: Find the Best Split for Decision Trees

```
Function get_best_split(data, numFeatures, decision_forest_features):
    bestGini ← ∞;
    bestCondition ← None;
    if isRandomForest then
        features ← getRandomsampleofFeatures(numberFeatures);
    else
        if isDecisionForest then
            features ← decision_forest_features;
        else
            features ← getAllFeaturesExceptTarget(data);
        end
    end
    foreach feature in features do
        if is_numeric(data[feature]) then
            sorted_data ← sort_data_by_feature(data, feature);
            values ← averageadjacentfromsorted(sorted_data);
        end
        foreach value in values do
            conditionsarray ← convertvaluestoconditions(value);
            foreach condition in conditionsarray do
                left, right ← data_split(data, condition);
                if not left.empty and not right.empty then
                    gini ← calc_gini_impurity(left, right);
                    if gini < bestGini then
                        bestGini ← gini;
                        bestCondition ← condition;
                    end
                end
            end
        end
    end
return
return bestCondition;
```

- For each feature, the function calculates unique values to consider as potential split points. For numeric features, an average of shifted values is used to create more candidate split points, enhancing the ability to find a more precise division.
- Each potential split point is evaluated by creating a split condition and dividing the dataset accordingly into left and right subsets.
- The Gini impurity for each split is calculated, and if this impurity is lower than any previously recorded, it updates *bestGini* and *bestCondition*.

The process is repeated for all features and their respective values, ensuring comprehensive coverage of potential splits. The optimal condition found—yielding the lowest Gini impurity—is returned, signifying the best way to split the dataset at that stage of the tree construction.

This methodical approach allows decision trees to make informed decisions at each node, facilitating the development of highly accurate and robust predictive models.

2.3 Predicting a sample with a decision Tree or a forest and calculate Performance

The "Forest Interpreter" or the Predict function of a forest classifier breaks down to the predict function of a tree. This is because a forest is just a list of trees. To calculate the predicted class the sample gets passed to the predict function of each tree. and the result gets saved. at the end the most common prediction is returned as result. To calculate the performance I use a simple average Function, that sums up the correct predictions and then divides it via the amount of samples. In the following you can see the pseudocode for a predict function for the forests. and you can see how a single tree is predicted. A single Tree is predicted via going through each node and comparing the sample with the condition until it reaches a leaf and then returns the leafnode label.

Algorithm 3: Predict Data with Decision Forest

Input : data, forest
Output: list of predictions

```
predictions ← empty list;
for each instance (index, instance) in data do
    votes ← empty list;
    for each tree in forest do
        vote ← predict(tree, instance[: −1]);
        append vote to votes;
    endfch
    prediction ← get maximum occurring item in votes;
    append prediction to predictions;
endfch
return predictions;
```

Algorithm 4: Predict Classification Using Decision Tree

```
Input  : node, instance
Output: class label
Function predict(node, instance):
    if node.is_leaf_node() then
        return node.label;
    endif
    if node.condition.is_numeric then
        if instance[node.condition.feature] < node.condition.value then
            return predict(node.left, instance);
        endif
        else
            return predict(node.right, instance);
        endif
    else
        if instance[node.condition.feature] == node.condition.value then
            return predict(node.left, instance);
        endif
        else
            return predict(node.right, instance);
        endif
    endif
return
```

3 Random Forest

Random Forests leverage a combination of tree predictors where each tree is influenced by independently sampled random vectors with a uniform distribution across the forest. As the forest grows in size, the generalization error asymptotically converges to a limit. The performance of a Random Forest is primarily determined by the strength of individual trees and their correlation, which is managed by introducing randomness through various methods such as bagging, random split selection, and random subspace method for feature selection. A Random Forest consists of a collection of tree classifiers $\{h(x,k), k = 1, \ldots\}$, where each tree $h(x,k)$ casts a vote for the most popular class for a given input x. The random vectors $\{k\}$ guiding the growth of each tree are independent and identically distributed. Key to the success of Random Forests is the selection of features at each node, which dictates the splits during tree growth. Typically, selecting just one or two features can yield near-optimal results. The internal mechanisms—out-of-bag estimates of generalization error, classifier strength, and dependency—offer insight into the model's performance and guide feature selection. Random Forests excel in accuracy, often outperforming other classifiers like AdaBoost. They are robust against outliers and noise, provide valuable internal estimates for error and variable importance, and are computationally efficient due to their parallelizable nature. The introduction of randomness, particularly through bagging and random feature selection, is crucial in minimizing tree correlation while maintaining classifier strength. In practice, trees within a Random Forest are grown to their maximum size using the CART methodology without pruning. For experimental validation, a subset of data can be set aside as a test set to evaluate the forest's performance with different parameters, such as the number of randomly selected features per node (F). Random Forests are a powerful tool for both classification and regression tasks, with their effectiveness underpinned by the Law of Large

Numbers, ensuring that they do not overfit. The strategic injection of randomness into the model not only enhances accuracy but also contributes to the ensemble's robustness, making Random Forests a preferred choice for complex predictive tasks.

3.1 Bootstrapping and Aggregation (Bagging)

The key features in Random forests are Bootstrapping which means it selects a random sample the size of the original data but with replacement allowed, which means there can be duplicates and it also selects a predefined number of random features from the original features at each splitting point for the node creation process, this provides more flexibility in creation of trees and it reduces the correlation between trees. Therefore gaining generalization ability and avoiding overfitting. and the Aggregation in Bagging means, to predict an example the forest uses plural voting, which means basically all trees predict the example and then the maximum class getrs chosen.

3.2 Random Forest Algorithm

The following pseudocode represents my Random forest algorithm. For simplicity i left out the parallelizing technique in the pseudocode. You can see the implementation in my github.

Algorithm 5: Build Random Forest

```
Function buildRandomForest(data, numFeatures, numTrees):
    forest ← [];
    foreach _ in range(numTrees) do
        bootstrappedData ← bootstrap(data);
        tree ← buildTree(bootstrappedData, numFeatures);
        forest.append(tree);
    end
    return forest;
return
```

4 Random Subspace Decision Forest

The Random Subspace Method for Constructing Decision Forests, as cited in Ho (1998), effectively maintains high accuracy as it increases in complexity, addressing the issue of overfitting. This technique constructs multiple decision trees by pseudorandomly selecting subsets of feature vector components within randomly chosen subspaces. Specifically, each tree is built by selecting a subset ranging from one to a predefined number of features (numFeatures), repeating this process until a specified number of trees (numTrees) is generated. Given a feature vector of size n, the method potentially considers 2^n different subsets, reducing correlation between trees and enhancing generalization accuracy.

This method employs binary trees that utilize various splitting functions, such as the Gini index or others that split data across hyperplanes. Importantly, these hyperplanes are not confined to being parallel to any axis of the feature space. They may also be situated in a transformed space where each feature dimension is a function of selected input features. Unlike many tree-building algorithms, it does not employ pruning. The stopping criterion for tree construction suggests continuing until the feature space is partitioned into regions each

containing samples from only one class unless inherent characteristics of the split prevent such partitioning. Alternatively, artificial stopping criteria, such as limiting the number of tree levels, can be imposed.

In terms of structural independence, the paper describes a simple measure of similarity between two decision trees based on the overlap of their decision regions, termed as tree agreement Ho (1998). Operating as a parallel learning algorithm, the Random Subspace Method allows each tree to be generated independently, which is ideal for parallel processing and facilitates rapid learning that is advantageous in practical settings. This approach ensures that as forests grow in complexity, they do not lose generalization accuracy while retaining maximum accuracy on training data. Furthermore, this forest construction technique is versatile, compatible with any splitting function, thereby ensuring robustness and efficiency without the risk of falling into local optima.

4.1 Algorithm

It is very similiar to the random forest method, except it does not generate a random set of features at each node. it gives the random subset of features in advance as an argument in the buildtree function and it does not bootstrap the dataset. for simplicity i left out the parallelization in the pseudocode.

Algorithm 6: Build Decision Forest

```
Function builddeciisonForest(data, numFeatures, numTrees):
    forest ← [];
    foreach _ in range(numTrees) do
        featureset ← randomSubsetof(data.columns[:-1]);
        tree ← buildTree(data, featureset);
        forest.append(tree);
    end
    return forest;
return
```

5 Methodology and Code guidance

The code is built like a library. you have the base classifier which is decisiontree.py. this contains everything to build one single decision tree like the classes of the node and the condition and fucntions for prediction, building, printing, calculating accuracy, splitting data, calculating gini impurity and gini index and the best split. The decisionforest.py is the same but for a decisionforest. it also contains the method for the hyperparameter tuning, which is a method, that runs tests on the same dataset and gets lists of number of features and number of trees and evaluates them, which memans calculating the accuracy and printing it aswell as the frequency of the features. the count features or counting the frequencies of the feature function is also in the forests file. The randomforests.py is built analog. The preprocessing.py contains only one function but for further development there can be more fucntions for preprocessing. this is the same for the visualization.py it parses a tree into a png with the graphviz library.

The two algorithms have been tested with 3 datasets one small, one medium and one large. I found everyone on UCI ML Repository. As a pre-processing task, entries with None got deleted. Then the dataset gets splitted into 80% training and 20% test dataset.

In the main file and fucntion the dataset is read and then the best tree is calulated of tuning the hyperparameter via testing all combinations of the given values from the given PW2 pdf from the assignment. The forest with the maximum accuracy is chosen and then an example tree is visualized and saved as a picture

6 Results and discussion

In this section I show the results of the two Algorithms on the training / test split and talk a bit about the datasets. I always splitted the dataset into 80% train data and hold back 20% of the data as test data.

Information about the number of features used

- -1 means in the evaluation table, that it uses int(math.sqrt(len(features))) for each node in random forest
- 0 means in the evaluation table that it uses random.randint(1, max(1,len(data.columns)-1)) for each tree constructed in decision fores
- the number of features tested was via the given functions in the assignment. you can see in the code.
- for the iris dataset, where M=4 for the features i took 1,2,3,4,special function

6.1 Dataset: Iris explaination and comments on the test

The Iris dataset, also known as the Fisher's Iris dataset, is a classic dataset from the UCI Machine Learning Repository that is commonly used in data science and machine learning for testing and learning algorithms. It was first introduced by the British statistician and biologist Ronald Fisher in 1936 as an example of discriminant analysis.

The dataset consists of 150 observations of iris flowers from three different species: Setosa, Versicolor, and Virginica. Each observation includes four features: Sepal Length (in cm), Sepal Width (in cm), Petal Length (in cm), Petal Width (in cm). These measurements are used to predict the species of the flower. The dataset is balanced, with 50 samples from each of the three species. In the following you see the tests.You can see the tests on the dataset in the appendix 12 from there on until the 6th and for decision forest 14 until the 6th of this type.

Comments on the tests The best accuracy for random forest scores the configuration with 75 trees, 2 features Highest accuracy: 0.9583333333333334 Feature counts for best configuration: 'petal length': 68, 'petal width': 67, 'sepal length': 26, 'sepal width': 11 Here you can see an example tree from the best random forest

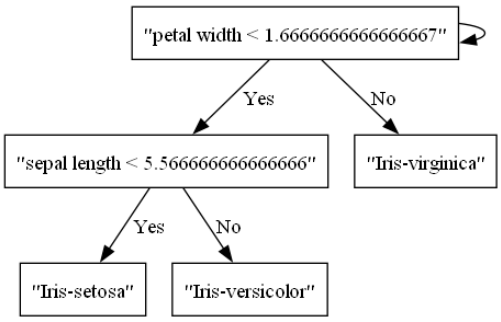

Figure 6: Random Forest Tree diagram Iris

In general we see, that if we only use 1 tree with 1 feature the result is not optimal. it gets better if we use more features and more trees but it converges very fast to $\tilde{0}.95$. The most important features are petal width and petal length, this can be seen by the frequency of the highest count in the feature count ordered list. So to interpret it, we can differentiate these flowers the fastest with a good accuracy by looking at petal width and petal length.

The best accuracy for decision forest scores the configuration with 1 trees, 1 features Highest accuracy: 0.95 Feature counts for best configuration: 'petal width': 2

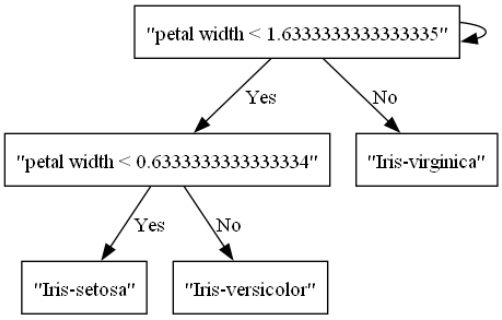

Figure 7: Decision Forest example Tree diagram Iris

In general if we look through the tests, we could always see a monotonic increase of accuracy, by adding more features. the best configuration here is actually an outlier example. We get more likely the sepal features to be most important through the tests. this is actually strange. It seems decision forest prefers them. The interpretation is the same, if we use the decision forest to classify an example we would know on which features to look at.

6.2 Dataset: Wisconsin Breast Cancer Dataset explaination and comments on the test

The Wisconsin Breast Cancer Dataset, often abbreviated as WBCD, is a classic dataset widely used in machine learning, particularly for testing algorithms in the field of medical diagnostic problems. It is available from the UCI Machine Learning Repository, a popular source for machine learning datasets.

The dataset consists of samples from breast masses collected via needle aspirate. Each sample in the dataset is described by several features that capture details of the cell nuclei present in the digitized images of the mass. Specifically, the dataset includes 569 instances, each with 30 numeric attributes (or features) that describe characteristics such as texture, radius, perimeter, area, smoothness, compactness, concavity, concave points, symmetry, and fractal dimension of the cell nuclei.

These features are used to classify each sample into one of two categories: benign (non-cancerous) or malignant (cancerous). .You can see the tests on the dataset in the appendix 16 from there on until the 6th and for decision forest 22 until the 6th of this type.

Comments on the tests The best accuracy for random forest scores the configuration with 25 trees, 2 features Highest accuracy: 0.9362637362637363 Feature counts for best configuration: 'area3': 10, 'area2': 9, 'radius1': 9, 'concavity3': 8, 'concavity2': 7, 'area1': 7, 'concave_points3': 7, 'radius3': 7, 'concave_points2': 7, 'compactness1': 7, 'concavity1': 7, 'texture2': 6, 'fractal_dimension3': 6, 'smoothness3': 6, 'perimeter3': 6, 'compactness3': 5, 'smoothness1': 5, 'radius2': 5, 'texture1': 5, 'symmetry3': 5, 'perimeter1': 5, 'texture3': 4, 'perimeter2': 4, 'concave_points1': 4, 'fractal_dimension2': 3, 'fractal_dimension1': 3, 'symmetry2': 3, 'smoothness2': 3, 'compactness2': 2 Here you can see an example tree from the best random forest In general we see, that if we only use 1 tree with 1 feature the result

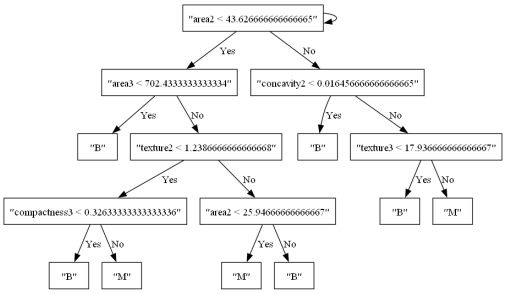

Figure 8: Random Forest Tree diagram wdbc

is not optimal. it gets better if we use more features and more trees but it converges very fast to $\tilde{0}.94$. The most important features are area3 concave points3 and perimeter3, this can be seen by the frequency of the highest count in the feature count ordered list. So to interpret it, we can differentiate these breast cancers the fastest with a good accuracy by looking at these features.

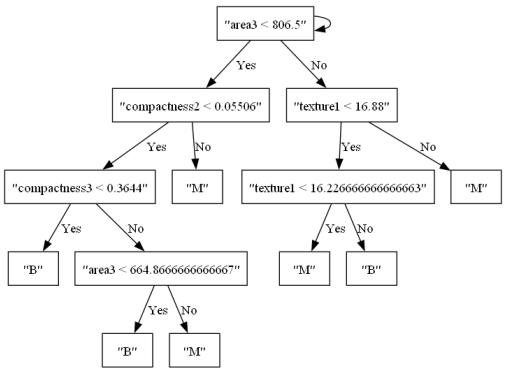

Figure 9: Decision Forest example Tree diagram WDBC

The best accuracy for decision forest scores the configuration with 25 trees, 7 features Highest accuracy: 0.9406593406593406 Feature counts for best configuration: 'smoothness2': 103, 'texture1': 33, 'compactness3': 22, 'concave_points1': 16, 'concavity3': 14, 'smoothness1': 13, 'smoothness3': 12, 'symmetry2': 11, 'area2': 9, 'perimeter1': 8, 'concavity1': 8, 'perimeter2': 7, 'perimeter3': 7, 'concavity2': 6, 'fractal_dimension3': 6, 'radius3': 6, 'concave_points2': 5, 'concave_points3': 5, 'area3': 4, 'symmetry1': 4, 'texture2': 3, 'compactness1': 3, 'compactness2': 2, 'fractal_dimension1': 2, 'symmetry3': 2, 'radius2': 2, 'area1': 2, 'fractal_dimension2': 1

I see again a monotonic increase from 0.72 accuracy to 0.91 when using only one tree but more features and we see also an increase when using more trees to starting with small amount of feature to 0.93. There is also a little bit of variance.

6.3 Dataset: Mushroom explaination and comments on the test

The Mushroom Dataset from the UCI Machine Learning Repository is a comprehensive collection of data used primarily for the task of classifying mushrooms as either edible or poisonous. This dataset is particularly popular in the machine learning community for testing classification algorithms.

It includes descriptions of hypothetical samples corresponding to 23 species of gilled mushrooms in the Agaricus and Lepiota families, drawn from the "Mushroom" field guide by Audubon Society. The dataset comprises 8124 instances, each represented by 22 attributes that cover various physical characteristics of the mushrooms, such as cap shape, cap surface, cap color, bruises, odor, gill attachment, gill spacing, gill size, gill color, stalk shape, stalk root, stalk surface above and below ring, stalk color above and below ring, veil type, veil color, ring number, ring type, spore print color, population, and habitat.

The primary challenge addressed by this dataset is to determine from these attributes whether a mushroom is edible or poisonous.You can see the tests on the dataset in the appendix 28 from there on until the 6th and for decision forest 34 until the 6th of this type.

Comments on the tests The best accuracy for random forest scores the configuration with Configuration: 50 trees, -1 features Highest accuracy: 1.0 Feature counts for best configuration: 'odor': 114, 'spore-print-color': 97, 'stalk-root': 70, 'cap-color': 65, 'habitat': 60, 'gill-color': 59, 'population': 55, 'stalk-surface-below-ring': 49, 'gill-size': 44, 'cap-surface': 42, 'bruises?': 42, 'gill-spacing': 41, 'stalk-color-below-ring': 39, 'ring-type': 37, 'stalk-surface-above-ring': 35, 'cap-shape': 31, 'stalk-shape': 28, 'stalk-color-above-ring': 25, 'ring-number': 23, 'veil-color': 4, 'gill-attachment': 1

Here you can see an example tree from the best random forest In general we see, that if

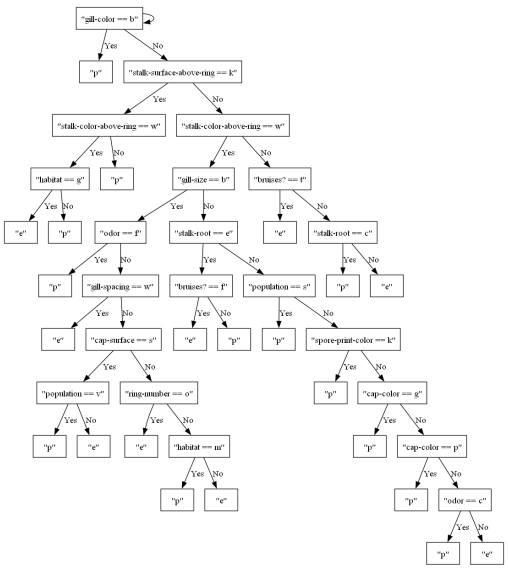

Figure 10: Random Forest Tree diagram Mushroom

we only use 1 tree with 1 feature the result is not optimal but surprisingly it is very good with accuracy of 0.92. it gets better if we use more features and more trees but it converges very fast to 1.0. Normally this is a sign of overfitting. The most important features are odor and

the other are random, this can be seen by the frequency of the highest count in the feature count ordered list. So to interpret it, we can differentiate these mushrooms if they are edible or not, by looking at the gill-color or cap-color. in random forest features are much more likely to be random

The best accuracy for decision forest scores the configuration with 1 trees, 1 features Highest accuracy: 0.95 Feature counts for best configuration: 'petal width': 2

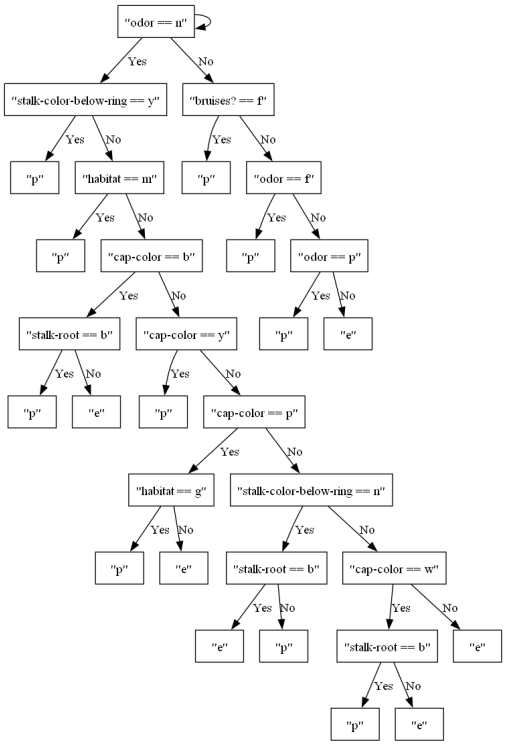

Figure 11: Decision Forest example Tree diagram Mushroom

In general if we look through the tests, we could always see a monotonic increase of accuracy, by adding more features. the best configuration here is actually an outlier example. The most important features are gill-color and cap-color, this can be seen by the frequency of the highest count in the feature count ordered list. So to interpret it, we can differentiate these mushrooms if they are edible or not, by looking at the gill-color or cap-color.

7 Code execution Instructions

Environment

- **OS:** Windows 10
- **Python Version:** 3.11
- **Laptop: Lenovo Legion 5**

Steps

1. Clone the Repository

Clone the repository and open a terminal in the root folder of the repo.

```
# Example command if needed
https://github.com/Barathaner/CART-method_with_randomf-forest_and_decision-forest.git
cd CART-method_with_randomf-forest_and_decision-forest
```

2. Set Up Virtual Environment and Install Dependencies

Create a Python virtual environment and install all required packages from `requirements.txt`.

```
python -m venv venv
venv\Scripts\activate  # Activate the virtual environment on Windows
pip install -r requirements.txt
```

3. Run the Application

Navigate to the source directory and start the main application.

```
cd src
python main.py
```

What to Expect

After starting the application, it will take approximately 20-40 minutes to process on a Lenovo Legion 5 If you run all three datasets and all possible permutations of the given number of features and trees. To just test if it is running I would suggest only run the Iris dataset. Since the implementation is a parralel one and it is using alot of cpu power, make sure to not run several other tasks on your pc, since this could cause freezing. If there are questions on how to run without parralelization please contact me. During this time, the application will:

- Execute random forests and decision forests on the given datasets, so it builds all classifier and runs all tests with the built classsifiers on all datasets, which you can choose in the main function and you also can choose which number of features and which number of trees you want to use.
- Save all the outputs in a generated output.txt file. You can see the inputs from previous runs inside their and you can also see an example in resultexample.txt

By following these steps, you will successfully execute and analyze the the random forests and decision forests. with detailed insights into its performance and accuracy. If you dont have graphviz installed comment out that lines.

8 Conclusion

In this work I showed a basic implementation of the CART classifier and the creation of random forests and the random subspace method from the paper. I tried to keep as close as possible to the implementation of the paper. The code is tested on various datasets and the built forests achieve overall good accuracies. Decent runtimes are archieved by paralellizing the process. Overal it was a good experience since it deepened my understanding in the domain of ensemble classsifiers. The downside of this project was to implement all the small

output steps. As recommended in the paper pruning is left out. I focused in this application mainly on classification and there were no regression datasets tested as it was not part of the assignment. Further work could be running benchmarks on the code to identify bottlenecks and optimize the runtime, change the implementation to also accept regression or making a parent class for the ensembel classsifier to avoid duplicate code for similiar techniques.

References

Breiman, L. (2001). Random forests. *Machine Learning*, 45(1):5–32.

Breiman, L., Friedman, J. H., Olshen, R. A., and Stone, C. J. (1984). *Classification and Regression Trees*. Wadsworth and Brooks, Monterey, CA.

Ho, T. K. (1998). The random subspace method for constructing decision forests. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 20(8):832–844.

9 Appendix

Figure 12: Random Forests Iris Tests 1/2

NumTrees	NumFeatures	Accuracy	Feature Counts
1	1	0.825000	'sepal width': 2, 'petal length': 2, 'sepal length': 2
1	2	0.816667	'petal length': 1, 'sepal width': 1
1	3	0.766667	'petal length': 1, 'sepal length': 1
1	4	0.925000	'petal length': 2
1	-1	0.825000	'petal length': 1, 'sepal width': 1
10	1	0.941667	'sepal length': 9, 'petal width': 9, 'petal length': 6, 'sepal width': 5
10	2	0.891667	'petal length': 11, 'sepal length': 7, 'petal width': 7, 'sepal width': 2
10	3	0.941667	'petal length': 9, 'petal width': 9, 'sepal length': 1, 'sepal width': 1
10	4	0.925000	'petal width': 11, 'petal length': 8, 'sepal length': 1
10	-1	0.950000	'petal width': 10, 'petal length': 10, 'sepal length': 2
25	1	0.916667	'sepal width': 29, 'petal width': 26, 'petal length': 19, 'sepal length': 18
25	2	0.933333	'petal width': 23, 'petal length': 23, 'sepal length': 4, 'sepal width': 3
25	3	0.950000	'petal width': 25, 'petal length': 22, 'sepal length': 2, 'sepal width': 2
25	4	0.933333	'petal width': 22, 'petal length': 20, 'sepal length': 6, 'sepal width': 2
25	-1	0.950000	'petal length': 28, 'petal width': 22, 'sepal length': 7
50	1	0.916667	'sepal length': 59, 'sepal width': 48, 'petal length': 43, 'petal width': 33
50	2	0.933333	'petal width': 47, 'petal length': 40, 'sepal length': 23, 'sepal width': 8
50	3	0.941667	'petal length': 46, 'petal width': 44, 'sepal length': 9, 'sepal width': 1
50	4	0.950000	'petal length': 49, 'petal width': 39, 'sepal length': 8, 'sepal width': 4
50	-1	0.950000	'petal width': 53, 'petal length': 44, 'sepal length': 16, 'sepal width': 5

Figure 13: Random Forest Iris Tests 2/2

NumTrees	NumFeatures	Accuracy	Feature Counts
75	1	0.933333	'sepal length': 79, 'sepal width': 76, 'petal width': 72, 'petal length': 68
75	2	0.958333	'petal length': 68, 'petal width': 67, 'sepal length': 26, 'sepal width': 11
75	3	0.950000	'petal width': 78, 'petal length': 62, 'sepal length': 9, 'sepal width': 5
75	4	0.958333	'petal width': 80, 'petal length': 59, 'sepal length': 6, 'sepal width': 5
75	-1	0.933333	'petal length': 79, 'petal width': 56, 'sepal length': 27, 'sepal width': 15
100	1	0.925000	'petal length': 100, 'sepal width': 92, 'sepal length': 90, 'petal width': 80
100	2	0.950000	'petal width': 99, 'petal length': 80, 'sepal length': 37, 'sepal width': 11
100	3	0.950000	'petal length': 94, 'petal width': 86, 'sepal length': 17, 'sepal width': 6
100	4	0.933333	'petal width': 92, 'petal length': 83, 'sepal length': 20, 'sepal width': 5
100	-1	0.933333	'petal length': 86, 'petal width': 85, 'sepal length': 50, 'sepal width': 16

Figure 14: Decision Forests Tests 1/5

Number of Trees	Number of Features	Accuracy	Feature Counts
1	1	0.950000	'petal width': 2
1	2	0.650000	'sepal width': 4, 'sepal length': 3
1	3	0.941667	'petal length': 3
1	4	0.950000	'petal width': 2
1	0	0.625000	'sepal length': 8
10	1	0.716667	'sepal width': 50, 'petal width': 6, 'petal length': 6
10	2	0.883333	'sepal length': 24, 'sepal width': 14, 'petal width': 8, 'petal length': 3
10	3	0.925000	'sepal width': 20, 'petal length': 13, 'sepal length': 8, 'petal width': 5
10	4	0.950000	'petal width': 11, 'sepal length': 8, 'petal length': 7, 'sepal width': 3
10	0	0.925000	'sepal length': 16, 'sepal width': 10, 'petal length': 9, 'petal width': 8
25	1	0.933333	'sepal length': 56, 'sepal width': 40, 'petal width': 22, 'petal length': 9
25	2	0.925000	'sepal width': 45, 'sepal length': 35, 'petal length': 23, 'petal width': 16
25	3	0.933333	'sepal width': 44, 'sepal length': 29, 'petal width': 22, 'petal length': 14
25	4	0.950000	'petal length': 28, 'sepal length': 23, 'petal width': 21, 'sepal width': 6
25	0	0.950000	'sepal length': 43, 'sepal width': 35, 'petal width': 24, 'petal length': 10

Figure 15: Decision Forests Iris 2/2

Number of Trees	Number of Features	Accuracy	Feature Counts
50	1	0.925000	'sepal width': 160, 'sepal length': 56, 'petal length': 36, 'petal width': 30
50	2	0.925000	'sepal length': 86, 'sepal width': 60, 'petal length': 41, 'petal width': 33
50	3	0.950000	'sepal width': 60, 'sepal length': 51, 'petal width': 49, 'petal length': 34
50	4	0.950000	'petal width': 49, 'petal length': 49, 'sepal width': 34, 'sepal length': 10
50	0	0.933333	'sepal length': 85, 'sepal width': 48, 'petal width': 39, 'petal length': 37
75	1	0.925000	'sepal width': 240, 'sepal length': 96, 'petal length': 63, 'petal width': 36
75	2	0.925000	'sepal width': 152, 'petal length': 73, 'sepal length': 70, 'petal width': 49
75	3	0.933333	'sepal length': 127, 'sepal width': 107, 'petal width': 60, 'petal length': 51
75	4	0.950000	'petal width': 75, 'sepal length': 66, 'petal length': 62, 'sepal width': 48
75	0	0.925000	'sepal width': 118, 'sepal length': 105, 'petal length': 77, 'petal width': 41
100	1	0.875000	'sepal width': 240, 'sepal length': 208, 'petal length': 87, 'petal width': 42
100	2	0.933333	'sepal width': 149, 'sepal length': 135, 'petal width': 84, 'petal length': 68
100	3	0.933333	'sepal width': 156, 'petal length': 88, 'petal width': 82, 'sepal length': 82
100	4	0.950000	'petal width': 111, 'sepal width': 92, 'petal length': 65, 'sepal length': 65
100	0	0.925000	'sepal width': 167, 'petal width': 89, 'petal length': 77, 'sepal length': 74

Figure 16: Random Forests wisconsin breast cancer dataset Tests 1/6

NumTrees	NumFeatures	Accuracy	Feature Counts
1	1	0.846154	'compactness2': 2, 'fractal_dimension3': 1, 'radius1': 1, 'compactness1': 1, 'area1': 1, 'symmetry3': 1
1	2	0.885714	'area2': 1, 'perimeter1': 1, 'radius2': 1, 'smoothness3': 1
1	4	0.887912	'radius3': 1, 'smoothness3': 1, 'area3': 1, 'smoothness2': 1
1	-1	0.883516	'radius3': 1, 'concavity3': 1, 'smoothness3': 1
10	1	0.916484	'symmetry2': 6, 'area1': 6, 'smoothness3': 5, 'perimeter2': 5, 'compactness1': 5, 'compactness2': 4, 'smoothness1': 4, 'concavity1': 4, 'perimeter1': 4, 'perimeter3': 4, 'texture1': 4, 'fractal_dimension2': 4, 'radius2': 4, 'symmetry1': 3, 'texture3': 3, 'symmetry3': 3, 'concave_points1': 3, 'fractal_dimension3': 3, 'smoothness2': 3, 'concavity3': 3, 'concave_points2': 2, 'area3': 2, 'compactness3': 2, 'fractal_dimension1': 2, 'texture2': 1, 'radius1': 1, 'concave_points3': 1, 'area2': 1, 'radius3': 1, 'concavity2': 1
10	2	0.912088	'radius3': 7, 'symmetry3': 5, 'radius1': 5, 'concavity2': 5, 'perimeter2': 4, 'concave_points2': 4, 'concave_points1': 4, 'concavity1': 4, 'compactness1': 4, 'texture3': 4, 'symmetry2': 3, 'fractal_dimension3': 3, 'texture1': 3, 'fractal_dimension1': 3, 'compactness2': 2, 'smoothness3': 2, 'area3': 2, 'perimeter1': 2, 'concavity3': 2, 'smoothness2': 1, 'area2': 1, 'area1': 1, 'compactness3': 1, 'radius2': 1, 'symmetry1': 1, 'concave_points3': 1, 'fractal_dimension2': 1
10	4	0.912088	'area3': 5, 'perimeter3': 4, 'compactness3': 3, 'area2': 3, 'concave_points3': 3, 'area1': 2, 'radius1': 2, 'radius2': 2, 'radius3': 2, 'concavity1': 1, 'compactness1': 1, 'compactness2': 1, 'fractal_dimension3': 1, 'concavity2': 1, 'perimeter2': 1, 'symmetry3': 1, 'smoothness3': 1, 'symmetry1': 1, 'texture3': 1, 'concave_points2': 1, 'concave_points1': 1, 'smoothness2': 1

Figure 17: Random Forests wisconsin breast cancer dataset Tests 2/6

NumTrees	NumFeatures	Accuracy	Feature Counts
10	-1	0.920879	'radius3': 5, 'perimeter3': 4, 'concave_points3': 4, 'radius2': 3, 'perimeter1': 3, 'smoothness1': 2, 'fractal_dimension2': 2, 'radius1': 2, 'texture2': 2, 'fractal_dimension1': 1, 'area1': 1, 'compactness3': 1, 'area3': 1, 'symmetry2': 1, 'area2': 1, 'concavity3': 1, 'symmetry3': 1, 'concave_points1': 1, 'smoothness2': 1, 'texture1': 1
25	1	0.916484	'radius2': 12, 'compactness3': 11, 'smoothness1': 11, 'perimeter2': 11, 'perimeter1': 10, 'concave_points3': 9, 'area2': 9, 'symmetry1': 8, 'area1': 8, 'area3': 8, 'symmetry3': 8, 'concave_points1': 8, 'texture1': 7, 'fractal_dimension3': 7, 'smoothness2': 6, 'radius1': 6, 'concave_points2': 6, 'fractal_dimension1': 6, 'compactness2': 6, 'fractal_dimension2': 5, 'texture2': 5, 'compactness1': 5, 'concavity1': 5, 'smoothness3': 5, 'concavity3': 5, 'perimeter3': 5, 'radius3': 4, 'symmetry2': 4, 'concavity2': 4, 'texture3': 3
25	2	0.936264	'area3': 10, 'area2': 9, 'radius1': 9, 'concavity3': 8, 'concavity2': 7, 'area1': 7, 'concave_points3': 7, 'radius3': 7, 'concave_points2': 7, 'compactness1': 7, 'concavity1': 7, 'texture2': 6, 'fractal_dimension3': 6, 'smoothness3': 6, 'perimeter3': 6, 'compactness3': 5, 'smoothness1': 5, 'radius2': 5, 'texture1': 5, 'symmetry3': 5, 'perimeter1': 5, 'texture3': 4, 'perimeter2': 4, 'concave_points1': 4, 'fractal_dimension2': 3, 'fractal_dimension1': 3, 'symmetry2': 3, 'smoothness2': 3, 'compactness2': 2
25	4	0.916484	'concave_points3': 10, 'radius3': 9, 'area3': 9, 'perimeter3': 8, 'fractal_dimension3': 8, 'smoothness3': 7, 'compactness3': 6, 'concavity1': 5, 'compactness1': 5, 'perimeter1': 5, 'compactness2': 3, 'symmetry2': 3, 'radius1': 3, 'perimeter2': 3, 'fractal_dimension2': 3, 'radius2': 3, 'area2': 3, 'concavity3': 2, 'area1': 2, 'smoothness2': 2, 'fractal_dimension1': 2, 'smoothness1': 2, 'texture1': 2, 'texture3': 2, 'concave_points1': 1, 'symmetry3': 1, 'symmetry1': 1, 'concave_points2': 1

Figure 18: Random Forests wisconsin breast cancer dataset Tests 3/6

NumTrees	NumFeatures	Accuracy	Feature Counts
25	-1	0.912088	'area3': 10, 'concave_points3': 8, 'radius3': 8, 'radius1': 7, 'compactness3': 7, 'concavity1': 7, 'concavity3': 7, 'fractal_dimension3': 6, 'perimeter2': 5, 'area2': 4, 'perimeter1': 4, 'radius2': 4, 'fractal_dimension2': 3, 'compactness1': 3, 'area1': 3, 'concave_points1': 3, 'perimeter3': 3, 'texture2': 2, 'smoothness3': 2, 'texture3': 2, 'fractal_dimension1': 2, 'symmetry2': 1, 'concave_points2': 1, 'smoothness2': 1
50	1	0.931868	'fractal_dimension2': 23, 'perimeter3': 21, 'concavity3': 20, 'area1': 19, 'concavity2': 19, 'compactness2': 19, 'fractal_dimension1': 18, 'perimeter2': 18, 'radius1': 18, 'compactness1': 17, 'texture2': 17, 'smoothness3': 16, 'compactness3': 16, 'concave_points2': 15, 'symmetry2': 15, 'fractal_dimension3': 15, 'radius2': 15, 'texture3': 14, 'smoothness2': 14, 'texture1': 13, 'perimeter1': 13, 'concave_points3': 13, 'radius3': 12, 'symmetry3': 12, 'area3': 12, 'smoothness1': 12, 'concavity1': 11, 'symmetry1': 10, 'area2': 10, 'concave_points1': 6
50	2	0.936264	'concave_points3': 21, 'perimeter3': 19, 'radius1': 16, 'area1': 16, 'perimeter1': 16, 'radius2': 16, 'area3': 15, 'concave_points1': 15, 'area2': 13, 'concavity3': 11, 'fractal_dimension3': 11, 'smoothness1': 11, 'fractal_dimension1': 10, 'compactness1': 10, 'radius3': 10, 'compactness3': 9, 'texture3': 9, 'smoothness3': 9, 'texture1': 8, 'perimeter2': 8, 'compactness2': 8, 'symmetry2': 7, 'concavity1': 7, 'concave_points2': 7, 'symmetry1': 6, 'texture2': 6, 'concavity2': 6, 'symmetry3': 6, 'fractal_dimension2': 3, 'smoothness2': 2
50	4	0.918681	'perimeter3': 24, 'radius3': 16, 'area3': 16, 'fractal_dimension3': 12, 'concave_points3': 11, 'area1': 10, 'radius1': 10, 'perimeter2': 9, 'area2': 9, 'compactness1': 8, 'perimeter1': 8, 'concave_points1': 8, 'compactness3': 7, 'compactness2': 7, 'concavity3': 7, 'texture3': 7, 'radius2': 7, 'concave_points2': 7, 'smoothness3': 6, 'concavity2': 6, 'concavity1': 6, 'fractal_dimension1': 5, 'symmetry2': 4, 'symmetry1': 4, 'texture2': 3, 'symmetry3': 2, 'smoothness1': 1, 'texture1': 1,

Figure 19: Random Forests wisconsin breast cancer dataset Tests 4/6

NumTrees	NumFeatures	Accuracy	Feature Counts
50	-1	0.925275	'perimeter3': 20, 'radius3': 20, 'area3': 14, 'concavity1': 12, 'compactness3': 12, 'concave_points3': 11, 'concavity3': 11, 'perimeter1': 11, 'fractal_dimension3': 10, 'area2': 9, 'concave_points1': 9, 'radius2': 8, 'smoothness3': 8, 'radius1': 7, 'area1': 6, 'perimeter2': 6, 'symmetry3': 5, 'concave_points2': 4, 'fractal_dimension2': 3, 'texture3': 3, 'concavity2': 3, 'texture1': 2, 'compactness1': 2, 'fractal_dimension1': 1, 'symmetry1': 1, 'smoothness2': 1, 'symmetry2': 1, 'smoothness1': 1
75	1	0.931868	'concave_points3': 33, 'smoothness2': 33, 'symmetry1': 29, 'radius3': 27, 'area3': 27, 'texture2': 27, 'radius2': 26, 'concavity2': 26, 'perimeter3': 26, 'fractal_dimension2': 24, 'area2': 24, 'concave_points1': 24, 'perimeter2': 24, 'smoothness1': 23, 'concavity1': 22, 'compactness2': 22, 'fractal_dimension1': 22, 'radius1': 21, 'texture3': 21, 'area1': 21, 'texture1': 21, 'smoothness3': 21, 'fractal_dimension3': 21, 'perimeter1': 20, 'symmetry2': 20, 'symmetry3': 19, 'concavity3': 18, 'compactness1': 16, 'compactness3': 16, 'concave_points2': 16
75	2	0.929670	'perimeter3': 35, 'area3': 25, 'concave_points3': 25, 'radius1': 24, 'concavity1': 23, 'smoothness3': 21, 'radius3': 20, 'perimeter1': 20, 'compactness3': 19, 'concavity2': 17, 'compactness1': 16, 'area2': 16, 'fractal_dimension3': 15, 'concave_points2': 15, 'concavity3': 14, 'concave_points1': 14, 'fractal_dimension1': 13, 'symmetry3': 12, 'area1': 12, 'perimeter2': 12, 'radius2': 12, 'symmetry1': 11, 'texture1': 11, 'symmetry2': 10, 'smoothness1': 10, 'texture3': 10, 'smoothness2': 8, 'compactness2': 7, 'fractal_dimension2': 6, 'texture2': 3

Figure 20: Random Forests wisconsin breast cancer dataset Tests 5/6

NumTrees	NumFeatures	Accuracy	Feature Counts
75	4	0.916484	'perimeter3': 28, 'area3': 24, 'radius3': 23, 'concave_points3': 21, 'perimeter1': 18, 'area2': 17, 'concave_points1': 16, 'compactness3': 16, 'area1': 15, 'concavity1': 15, 'concavity3': 13, 'texture3': 12, 'radius1': 11, 'smoothness3': 10, 'fractal_dimension3': 9, 'texture1': 9, 'radius2': 8, 'concavity2': 8, 'concave_points2': 7, 'symmetry3': 6, 'perimeter2': 6, 'smoothness1': 6, 'compactness1': 5, 'symmetry2': 5, 'texture2': 4, 'compactness2': 4, 'fractal_dimension2': 3, 'smoothness2': 3, 'symmetry1': 2, 'fractal_dimension1': 1
75	-1	0.931868	'perimeter3': 27, 'radius3': 24, 'concave_points3': 20, 'area2': 20, 'concavity3': 18, 'concavity1': 16, 'area3': 16, 'concave_points1': 14, 'compactness3': 13, 'radius2': 13, 'compactness1': 12, 'radius1': 12, 'perimeter2': 11, 'area1': 9, 'perimeter1': 7, 'concave_points2': 7, 'fractal_dimension1': 7, 'texture3': 6, 'symmetry3': 6, 'concavity2': 5, 'symmetry1': 5, 'symmetry2': 5, 'texture1': 5, 'smoothness3': 4, 'compactness2': 4, 'fractal_dimension3': 4, 'smoothness2': 3, 'texture2': 3, 'smoothness1': 2, 'fractal_dimension2': 2
100	1	0.936264	'smoothness2': 42, 'concave_points3': 39, 'texture2': 38, 'fractal_dimension2': 38, 'symmetry1': 37, 'symmetry3': 36, 'concavity1': 35, 'symmetry2': 35, 'concave_points1': 33, 'radius2': 33, 'smoothness1': 33, 'texture1': 32, 'area3': 32, 'area2': 32, 'compactness2': 31, 'compactness1': 31, 'concavity3': 31, 'fractal_dimension1': 31, 'perimeter1': 30, 'radius3': 30, 'perimeter2': 29, 'compactness3': 29, 'radius1': 29, 'area1': 29, 'fractal_dimension3': 28, 'concave_points2': 28, 'texture3': 27, 'concavity2': 27, 'smoothness3': 26, 'perimeter3': 26

Figure 21: Random Forests wisconsin breast cancer dataset Tests 6/6

NumTrees	NumFeatures	Accuracy	Feature Counts
100	2	0.931868	'area3': 36, 'radius1': 34, 'perimeter3': 32, 'perimeter1': 32, 'area1': 27, 'radius3': 26, 'concavity3': 26, 'perimeter2': 25, 'concavity1': 24, 'concave_points3': 24, 'area2': 23, 'compactness3': 22, 'concave_points1': 22, 'compactness1': 22, 'smoothness1': 22, 'texture3': 21, 'radius2': 19, 'smoothness3': 19, 'texture1': 18, 'concavity2': 14, 'compactness2': 14, 'fractal_dimension2': 14, 'symmetry3': 13, 'symmetry2': 13, 'fractal_dimension3': 12, 'symmetry1': 10, 'smoothness2': 10, 'texture2': 9, 'concave_points2': 8, 'fractal_dimension1': 8
100	4	0.920879	'perimeter3': 36, 'concave_points3': 33, 'radius3': 31, 'radius1': 25, 'area1': 21, 'concavity3': 21, 'area3': 21, 'concavity1': 20, 'perimeter2': 20, 'perimeter1': 19, 'concave_points1': 19, 'compactness3': 17, 'fractal_dimension3': 17, 'radius2': 15, 'area2': 13, 'compactness1': 13, 'compactness2': 9, 'texture1': 9, 'fractal_dimension2': 8, 'fractal_dimension1': 8, 'smoothness3': 8, 'texture2': 6, 'symmetry3': 6, 'concave_points2': 6, 'texture3': 5, 'smoothness2': 5, 'smoothness1': 5, 'concavity2': 4, 'symmetry2': 3, 'symmetry1': 3
100	-1	0.916484	'area3': 37, 'perimeter3': 32, 'concave_points3': 31, 'radius3': 28, 'concavity3': 23, 'perimeter1': 21, 'area2': 19, 'area1': 17, 'smoothness3': 14, 'concavity1': 13, 'perimeter2': 12, 'radius2': 12, 'radius1': 12, 'symmetry3': 11, 'compactness3': 11, 'concave_points1': 11, 'compactness1': 9, 'smoothness2': 8, 'fractal_dimension1': 8, 'fractal_dimension3': 7, 'texture3': 7, 'texture1': 6, 'symmetry2': 6, 'smoothness1': 5, 'concavity2': 5, 'symmetry1': 4, 'texture2': 4, 'concave_points2': 3, 'compactness2': 3, 'fractal_dimension2': 3

Figure 22: Decision Forests wisconsin breast cancer dataset Tests 1/6

NumTrees	NumFeatures	Accuracy	Feature Counts
1	7	0.723077	'compactness1': 21, 'concavity2': 6
1	15	0.865934	'fractal_dimension1': 2, 'perimeter3': 1, 'perimeter2': 1, 'fractal_dimension2': 1
1	22	0.914286	'area1': 2, 'perimeter3': 1, 'compactness1': 1, 'texture3': 1
1	0	0.892308	'radius3': 2, 'fractal_dimension3': 1, 'smoothness3': 1
10	7	0.925275	'fractal_dimension3': 50, 'compactness3': 19, 'texture3': 10, 'concave_points1': 6, 'compactness1': 5, 'perimeter1': 5, 'radius3': 5, 'area3': 4, 'perimeter3': 3, 'radius2': 3, 'symmetry2': 2, 'texture2': 2, 'texture1': 2, 'area2': 2, 'smoothness3': 2, 'fractal_dimension2': 2, 'symmetry3': 2, 'radius1': 1, 'concavity2': 1, 'concave_points3': 1
10	15	0.912088	'concavity1': 8, 'perimeter3': 7, 'concave_points3': 7, 'concave_points2': 5, 'symmetry2': 4, 'compactness1': 4, 'perimeter2': 2, 'smoothness1': 2, 'smoothness3': 2, 'compactness2': 2, 'texture1': 2, 'area3': 1, 'compactness3': 1, 'symmetry3': 1, 'fractal_dimension3': 1, 'texture3': 1, 'radius2': 1, 'radius3': 1, 'concavity3': 1, 'area2': 1, 'texture2': 1, 'area1': 1, 'radius1': 1, 'concavity2': 1
10	22	0.914286	'fractal_dimension2': 49, 'perimeter3': 4, 'area3': 4, 'concavity3': 4, 'symmetry2': 4, 'smoothness3': 3, 'concave_points3': 3, 'radius1': 3, 'perimeter2': 3, 'fractal_dimension1': 3, 'texture1': 2, 'compactness1': 2, 'compactness3': 1, 'smoothness2': 1, 'texture3': 1, 'perimeter1': 1, 'compactness2': 1, 'texture2': 1, 'radius3': 1
10	0	0.925275	'smoothness1': 49, 'perimeter2': 27, 'area3': 12, 'smoothness2': 11, 'concave_points3': 6, 'concavity1': 5, 'concave_points1': 4, 'area1': 4, 'perimeter3': 3, 'smoothness3': 3, 'fractal_dimension3': 2, 'radius2': 2, 'concavity3': 1, 'fractal_dimension1': 1, 'texture3': 1, 'symmetry1': 1, 'texture1': 1
25	7	0.940659	'smoothness2': 103, 'texture1': 33, 'compactness3': 22, 'concave_points1': 16, 'concavity3': 14, 'smoothness1': 13, 'smoothness3': 12, 'symmetry2': 11, 'area2': 9, 'perimeter1': 8, 'concavity1': 8, 'perimeter2': 7, 'perimeter3': 7, 'concavity2': 6, 'fractal_dimension3': 6, 'radius3': 6, 'concave_points2': 5, 'concave_points3': 5, 'area3': 4,

Figure 23: Decision Forests wisconsin breast cancer dataset Tests 2/6

NumTrees	NumFeatures	Accuracy	Feature Counts
25	15	0.923077	'texture2': 61, 'concavity1': 30, 'perimeter3': 19, 'compactness1': 18, 'smoothness1': 18, 'smoothness3': 15, 'concavity2': 13, 'concavity3': 13, 'perimeter2': 10, 'area3': 8, 'concave_points3': 7, 'concave_points1': 5, 'perimeter1': 5, 'radius3': 4, 'radius2': 3, 'compactness3': 3, 'symmetry1': 2, 'fractal_dimension2': 2, 'area2': 2, 'area1': 1, 'texture1': 1, 'fractal_dimension3': 1, 'radius1': 1
25	22	0.929670	'symmetry2': 27, 'area3': 23, 'symmetry3': 18, 'compactness1': 16, 'fractal_dimension3': 16, 'area2': 14, 'concave_points3': 10, 'smoothness3': 9, 'perimeter1': 6, 'concavity3': 6, 'perimeter3': 5, 'concave_points1': 5, 'radius3': 5, 'area1': 5, 'compactness3': 4, 'symmetry1': 4, 'perimeter2': 4, 'fractal_dimension1': 3, 'texture3': 3, 'radius1': 3, 'compactness2': 2, 'smoothness2': 2, 'concavity2': 2, 'texture1': 1, 'concavity1': 1, 'concave_points2': 1
25	0	0.923077	'texture2': 59, 'compactness2': 50, 'concave_points2': 48, 'area1': 17, 'area3': 15, 'radius3': 13, 'concave_points3': 9, 'symmetry3': 8, 'radius2': 7, 'area2': 6, 'compactness3': 6, 'smoothness3': 6, 'concavity3': 6, 'compactness1': 5, 'perimeter1': 5, 'symmetry2': 4, 'perimeter3': 4, 'fractal_dimension3': 4, 'fractal_dimension1': 2, 'smoothness1': 2, 'texture1': 2, 'radius1': 2, 'smoothness2': 2, 'concave_points1': 1, 'texture3': 1, 'concavity2': 1, 'fractal_dimension2': 1, 'perimeter2': 1
50	7	0.920879	'smoothness1': 51, 'symmetry2': 49, 'fractal_dimension1': 45, 'compactness2': 44, 'concave_points2': 44, 'radius1': 33, 'concave_points3': 25, 'area1': 23, 'fractal_dimension2': 22, 'symmetry3': 21, 'texture3': 19, 'concavity1': 17, 'perimeter3': 16, 'area3': 16, 'texture2': 16, 'compactness1': 15, 'area2': 14, 'fractal_dimension3': 13, 'concavity3': 13, 'radius3': 12, 'smoothness3': 12, 'concavity2': 12, 'radius2': 11, 'smoothness2': 11, 'symmetry1': 8, 'texture1': 7, 'perimeter2': 5, 'perimeter1': 5, 'compactness3': 4, 'concave_points1': 3

Figure 24: Decision Forests wisconsin breast cancer dataset Tests 3/6

NumTrees	NumFeatures	Accuracy	Feature Counts
50	15	0.923077	'texture2': 67, 'smoothness2': 56, 'texture3': 45, 'compactness3': 44, 'area1': 28, 'concave_points1': 21, 'concavity3': 19, 'perimeter3': 17, 'area3': 14, 'radius3': 13, 'concave_points3': 13, 'symmetry2': 12, 'radius2': 12, 'smoothness3': 11, 'fractal_dimension1': 11, 'symmetry3': 11, 'concavity1': 10, 'fractal_dimension3': 9, 'perimeter2': 9, 'perimeter1': 9, 'compactness2': 8, 'smoothness1': 7, 'concave_points2': 7, 'area2': 6, 'fractal_dimension2': 3, 'radius1': 2, 'symmetry1': 2, 'concavity2': 2, 'compactness1': 2
50	22	0.920879	'radius1': 25, 'area3': 24, 'concave_points1': 23, 'perimeter3': 21, 'perimeter1': 17, 'concave_points3': 16, 'smoothness3': 15, 'concavity1': 13, 'concavity3': 11, 'area1': 11, 'fractal_dimension3': 10, 'texture1': 9, 'compactness3': 9, 'radius3': 8, 'concave_points2': 8, 'perimeter2': 8, 'fractal_dimension2': 7, 'compactness1': 6, 'texture3': 6, 'smoothness2': 5, 'radius2': 5, 'area2': 5, 'symmetry3': 4, 'texture2': 4, 'concavity2': 4, 'compactness2': 4, 'fractal_dimension1': 2, 'symmetry2': 2
50	0	0.923077	'texture2': 62, 'perimeter2': 36, 'concave_points1': 33, 'texture1': 31, 'fractal_dimension2': 27, 'concave_points3': 26, 'area3': 21, 'symmetry3': 21, 'symmetry2': 20, 'concavity2': 20, 'fractal_dimension1': 17, 'perimeter3': 16, 'smoothness1': 16, 'perimeter1': 15, 'compactness1': 15, 'radius1': 14, 'texture3': 14, 'area1': 10, 'radius3': 9, 'compactness3': 9, 'smoothness3': 8, 'compactness2': 7, 'symmetry1': 6, 'concavity3': 5, 'concavity1': 5, 'smoothness2': 4, 'concave_points2': 4, 'fractal_dimension3': 3, 'radius2': 3, 'area2': 3
75	7	0.936264	'concavity2': 114, 'texture2': 94, 'texture3': 75, 'smoothness3': 71, 'concave_points1': 53, 'texture1': 41, 'perimeter2': 37, 'area1': 34, 'radius1': 34, 'perimeter1': 33, 'area2': 32, 'area3': 30, 'concave_points2': 30, 'fractal_dimension2': 26, 'compactness2': 26, 'compactness3': 23, 'symmetry1': 22, 'concavity1': 21, 'radius2': 20, 'concave_points3': 19, 'concavity3': 17, 'fractal_dimension1': 17, 'smoothness2': 17,

Figure 25: Decision Forests wisconsin breast cancer dataset Tests 4/6

NumTrees	NumFeatures	Accuracy	Feature Counts
75	15	0.920879	'fractal_dimension3': 100, 'symmetry3': 66, 'texture3': 53, 'perimeter2': 48, 'concave_points3': 39, 'radius2': 36, 'concave_points2': 35, 'area2': 32, 'perimeter3': 30, 'compactness2': 30, 'texture2': 29, 'smoothness3': 28, 'concavity2': 25, 'radius3': 24, 'concavity3': 22, 'texture1': 22, 'area1': 22, 'concavity1': 21, 'area3': 13, 'concave_points1': 13, 'radius1': 13, 'compactness3': 12, 'compactness1': 9, 'symmetry1': 7, 'fractal_dimension1': 5, 'symmetry2': 2, 'smoothness1': 2, 'smoothness2': 1, 'fractal_dimension2': 1, 'perimeter1': 1
75	22	0.923077	'perimeter2': 42, 'fractal_dimension3': 40, 'perimeter3': 37, 'smoothness1': 34, 'texture3': 33, 'smoothness3': 32, 'concave_points3': 30, 'concavity1': 27, 'area3': 25, 'smoothness2': 21, 'texture1': 19, 'radius3': 18, 'symmetry2': 18, 'texture2': 17, 'concave_points1': 14, 'fractal_dimension1': 14, 'symmetry3': 13, 'concave_points2': 11, 'concavity3': 9, 'perimeter1': 9, 'area1': 9, 'compactness3': 9, 'radius2': 8, 'area2': 7, 'compactness2': 5, 'symmetry1': 4, 'radius1': 4, 'compactness1': 3, 'fractal_dimension2': 2, 'concavity2': 1
75	0	0.925275	'symmetry2': 126, 'symmetry1': 110, 'compactness3': 86, 'fractal_dimension2': 66, 'perimeter2': 53, 'smoothness2': 44, 'radius3': 34, 'concavity1': 28, 'concave_points3': 27, 'texture1': 26, 'texture3': 24, 'fractal_dimension3': 23, 'smoothness3': 22, 'compactness2': 22, 'area3': 21, 'perimeter3': 19, 'concave_points1': 17, 'perimeter1': 16, 'concavity3': 15, 'radius2': 14, 'smoothness1': 13, 'fractal_dimension1': 12, 'concave_points2': 12, 'compactness1': 11, 'radius1': 8, 'symmetry3': 8, 'concavity2': 7, 'area1': 6, 'texture2': 5, 'area2': 4

Figure 26: Decision Forests wisconsin breast cancer dataset Tests 5/6

NumTrees	NumFeatures	Accuracy	Feature Counts
100	7	0.929670	'fractal_dimension1': 178, 'smoothness3': 104, 'symmetry2': 86, 'texture3': 84, 'texture2': 82, 'concave_points1': 76, 'symmetry1': 72, 'smoothness2': 66, 'area1': 55, 'symmetry3': 54, 'concave_points3': 50, 'perimeter2': 48, 'compactness2': 48, 'compactness3': 48, 'compactness1': 42, 'perimeter3': 36, 'radius3': 36, 'concavity1': 31, 'area2': 31, 'radius1': 31, 'radius2': 30, 'texture1': 24, 'concave_points2': 23, 'smoothness1': 21, 'concavity2': 20, 'concavity3': 17, 'fractal_dimension2': 16, 'area3': 13, 'fractal_dimension3': 12, 'perimeter1': 10
100	15	0.923077	'symmetry3': 73, 'texture2': 67, 'smoothness1': 54, 'concave_points3': 44, 'smoothness3': 42, 'perimeter3': 41, 'area1': 41, 'radius2': 40, 'area3': 39, 'fractal_dimension1': 32, 'concave_points1': 28, 'radius1': 27, 'radius3': 26, 'concavity1': 25, 'concavity2': 23, 'concavity3': 20, 'fractal_dimension3': 18, 'smoothness2': 18, 'texture3': 17, 'perimeter1': 16, 'texture1': 15, 'perimeter2': 15, 'area2': 15, 'compactness1': 13, 'compactness3': 11, 'symmetry1': 11, 'compactness2': 10, 'symmetry2': 5, 'fractal_dimension2': 4, 'concave_points2': 3
100	22	0.923077	'smoothness3': 73, 'fractal_dimension2': 56, 'perimeter3': 47, 'concave_points3': 37, 'area3': 29, 'texture2': 26, 'compactness1': 24, 'area1': 24, 'texture1': 23, 'texture3': 22, 'radius3': 20, 'perimeter2': 20, 'fractal_dimension1': 19, 'smoothness1': 19, 'concavity3': 18, 'area2': 18, 'fractal_dimension3': 15, 'concavity1': 15, 'compactness3': 14, 'concave_points1': 14, 'radius2': 13, 'radius1': 13, 'symmetry2': 13, 'compactness2': 12, 'perimeter1': 12, 'smoothness2': 9, 'concavity2': 8, 'concave_points2': 4, 'symmetry3': 3, 'symmetry1': 1

Figure 27: Decision Forests wisconsin breast cancer dataset Tests 6/6

NumTrees	NumFeatures	Accuracy	Feature Counts
100	0	0.925275	'smoothness1': 94, 'smoothness2': 81, 'smoothness3': 68, 'symmetry2': 67, 'symmetry1': 51, 'concave_points1': 50, 'perimeter3': 47, 'perimeter2': 45, 'concave_points3': 44, 'concavity2': 39, 'compactness2': 34, 'radius3': 32, 'area2': 28, 'compactness3': 26, 'area3': 25, 'area1': 24, 'concave_points2': 24, 'perimeter1': 23, 'radius1': 23, 'symmetry3': 22, 'fractal_dimension1': 22, 'concavity1': 21, 'fractal_dimension3': 15, 'texture2': 15, 'radius2': 12, 'texture1': 12, 'texture3': 12, 'concavity3': 11, 'compactness1': 9, 'fractal_dimension2': 7

Figure 28: Random Forests Mushroom dataset Tests 1/6

NumTrees	NumFeatures	Accuracy	Feature Counts
1	1	0.919680	'ring-number': 1, 'cap-surface': 1, 'ring-type': 1, 'stalk-color-below-ring': 1, 'odor': 1, 'stalk-color-above-ring': 1, 'stalk-root': 1
1	2	0.993537	'spore-print-color': 4, 'gill-color': 3, 'stalk-shape': 3, 'stalk-color-below-ring': 2, 'stalk-color-above-ring': 2, 'cap-shape': 2, 'population': 2, 'stalk-surface-below-ring': 2, 'veil-color': 1, 'odor': 1, 'habitat': 1, 'stalk-surface-above-ring': 1, 'cap-color': 1, 'gill-size': 1, 'gill-attachment': 1, 'cap-surface': 1
1	4	0.989691	'spore-print-color': 4, 'cap-color': 2, 'stalk-shape': 2, 'gill-size': 2, 'cap-shape': 2, 'odor': 1, 'bruises?': 1, 'ring-type': 1, 'habitat': 1, 'stalk-color-below-ring': 1, 'population': 1, 'gill-spacing': 1, 'cap-surface': 1, 'stalk-root': 1, 'gill-color': 1
1	-1	0.996307	'odor': 4, 'cap-color': 3, 'habitat': 3, 'population': 2, 'spore-print-color': 2, 'stalk-surface-above-ring': 1, 'ring-number': 1, 'bruises?': 1, 'cap-surface': 1, 'stalk-root': 1, 'stalk-surface-below-ring': 1, 'stalk-color-above-ring': 1
10	1	0.931682	'cap-shape': 10, 'habitat': 10, 'gill-color': 10, 'population': 9, 'cap-color': 9, 'stalk-color-below-ring': 9, 'gill-spacing': 9, 'stalk-surface-above-ring': 8, 'stalk-color-above-ring': 8, 'cap-surface': 8, 'gill-size': 8, 'stalk-shape': 7, 'ring-number': 6, 'odor': 6, 'spore-print-color': 5, 'veil-color': 4, 'stalk-root': 4, 'bruises?': 4, 'stalk-surface-below-ring': 4, 'ring-type': 4, 'gill-attachment': 1
10	2	0.997692	'odor': 20, 'population': 18, 'gill-color': 15, 'stalk-root': 15, 'spore-print-color': 15, 'gill-spacing': 13, 'habitat': 12, 'cap-color': 12, 'ring-type': 9, 'stalk-shape': 9, 'cap-shape': 9, 'bruises?': 8, 'ring-number': 8, 'gill-size': 8, 'stalk-surface-below-ring': 7, 'stalk-color-above-ring': 7, 'cap-surface': 7, 'stalk-color-below-ring': 6, 'stalk-surface-above-ring': 5, 'veil-color': 2, 'gill-attachment': 1

Figure 29: Random Forests Mushroom dataset Tests 2/6

NumTrees	NumFeatures	Accuracy	Feature Counts
10	4	0.998923	'population': 25, 'spore-print-color': 23, 'odor': 22, 'gill-color': 17, 'cap-color': 13, 'stalk-color-above-ring': 12, 'gill-size': 12, 'stalk-shape': 12, 'cap-surface': 11, 'habitat': 11, 'ring-type': 11, 'stalk-surface-above-ring': 10, 'gill-spacing': 10, 'stalk-root': 10, 'cap-shape': 8, 'stalk-color-below-ring': 7, 'bruises?': 6, 'stalk-surface-below-ring': 6, 'ring-number': 3
10	-1	0.999077	'odor': 19, 'spore-print-color': 18, 'habitat': 13, 'cap-color': 12, 'stalk-surface-below-ring': 11, 'gill-color': 11, 'population': 10, 'stalk-root': 10, 'gill-spacing': 8, 'stalk-shape': 8, 'bruises?': 7, 'stalk-color-above-ring': 7, 'gill-size': 7, 'ring-number': 7, 'stalk-color-below-ring': 6, 'stalk-surface-above-ring': 6, 'ring-type': 6, 'cap-shape': 5, 'cap-surface': 5, 'gill-attachment': 1
25	1	0.945376	'habitat': 19, 'gill-color': 18, 'stalk-surface-above-ring': 17, 'spore-print-color': 17, 'ring-type': 17, 'odor': 17, 'gill-size': 16, 'cap-shape': 16, 'stalk-color-below-ring': 16, 'stalk-surface-below-ring': 16, 'stalk-shape': 15, 'population': 15, 'cap-color': 14, 'stalk-color-above-ring': 14, 'bruises?': 14, 'ring-number': 13, 'veil-color': 11, 'cap-surface': 11, 'stalk-root': 10, 'gill-spacing': 10, 'gill-attachment': 6
25	2	0.998769	'odor': 47, 'gill-color': 44, 'habitat': 40, 'stalk-root': 36, 'spore-print-color': 34, 'cap-color': 32, 'gill-size': 32, 'cap-shape': 30, 'population': 29, 'stalk-surface-below-ring': 28, 'stalk-color-below-ring': 28, 'stalk-surface-above-ring': 25, 'ring-type': 23, 'cap-surface': 23, 'bruises?': 22, 'stalk-color-above-ring': 20, 'stalk-shape': 19, 'gill-spacing': 17, 'ring-number': 17, 'gill-attachment': 7, 'veil-color': 4

Figure 30: Random Forests Mushroom dataset Tests 3/6

NumTrees	NumFeatures	Accuracy	Feature Counts
25	4	0.999538	'odor': 52, 'spore-print-color': 48, 'cap-color': 33, 'habitat': 32, 'stalk-root': 30, 'population': 28, 'gill-color': 25, 'stalk-shape': 24, 'gill-size': 24, 'cap-surface': 23, 'stalk-color-below-ring': 20, 'stalk-surface-above-ring': 18, 'cap-shape': 17, 'stalk-surface-below-ring': 15, 'ring-type': 15, 'ring-number': 14, 'gill-spacing': 13, 'bruises?': 12, 'stalk-color-above-ring': 7, 'veil-color': 6, 'gill-attachment': 2
25	-1	0.999692	'odor': 66, 'spore-print-color': 47, 'habitat': 40, 'gill-color': 38, 'stalk-root': 36, 'cap-color': 34, 'population': 31, 'gill-size': 22, 'stalk-surface-below-ring': 22, 'cap-surface': 22, 'cap-shape': 20, 'stalk-color-below-ring': 20, 'bruises?': 19, 'ring-type': 19, 'stalk-shape': 18, 'gill-spacing': 16, 'stalk-surface-above-ring': 15, 'stalk-color-above-ring': 12, 'ring-number': 9, 'veil-color': 2
50	1	0.944915	'cap-surface': 46, 'spore-print-color': 44, 'gill-color': 42, 'stalk-surface-below-ring': 39, 'population': 38, 'odor': 36, 'cap-color': 36, 'cap-shape': 34, 'stalk-shape': 34, 'habitat': 33, 'stalk-color-below-ring': 33, 'stalk-surface-above-ring': 32, 'bruises?': 31, 'ring-type': 31, 'stalk-root': 29, 'gill-spacing': 29, 'stalk-color-above-ring': 27, 'ring-number': 26, 'gill-size': 21, 'veil-color': 17, 'gill-attachment': 11
50	2	0.999385	'odor': 91, 'gill-color': 89, 'habitat': 86, 'stalk-root': 84, 'population': 70, 'spore-print-color': 68, 'cap-surface': 63, 'cap-shape': 63, 'cap-color': 63, 'gill-size': 62, 'stalk-surface-below-ring': 59, 'ring-type': 46, 'stalk-color-below-ring': 45, 'stalk-surface-above-ring': 43, 'bruises?': 41, 'gill-spacing': 41, 'stalk-color-above-ring': 40, 'stalk-shape': 39, 'ring-number': 34, 'veil-color': 13, 'gill-attachment': 8

Figure 31: Random Forests Mushroom dataset Tests 4/6

NumTrees	NumFeatures	Accuracy	Feature Counts
50	4	0.999538	'odor': 116, 'spore-print-color': 82, 'habitat': 75, 'cap-color': 67, 'stalk-root': 67, 'gill-color': 66, 'population': 53, 'gill-size': 52, 'stalk-surface-below-ring': 48, 'cap-surface': 43, 'ring-type': 43, 'stalk-color-below-ring': 40, 'stalk-shape': 36, 'ring-number': 35, 'bruises?': 32, 'gill-spacing': 31, 'stalk-surface-above-ring': 30, 'cap-shape': 29, 'stalk-color-above-ring': 29, 'veil-color': 2, 'gill-attachment': 1
50	-1	1.000000	'odor': 114, 'spore-print-color': 97, 'stalk-root': 70, 'cap-color': 65, 'habitat': 60, 'gill-color': 59, 'population': 55, 'stalk-surface-below-ring': 49, 'gill-size': 44, 'cap-surface': 42, 'bruises?': 42, 'gill-spacing': 41, 'stalk-color-below-ring': 39, 'ring-type': 37, 'stalk-surface-above-ring': 35, 'cap-shape': 31, 'stalk-shape': 28, 'stalk-color-above-ring': 25, 'ring-number': 23, 'veil-color': 4, 'gill-attachment': 1
75	1	0.981228	'stalk-surface-below-ring': 69, 'cap-shape': 68, 'habitat': 60, 'cap-surface': 59, 'odor': 57, 'cap-color': 55, 'spore-print-color': 50, 'gill-color': 49, 'ring-type': 49, 'stalk-root': 49, 'population': 47, 'gill-size': 44, 'gill-spacing': 44, 'stalk-color-above-ring': 44, 'bruises?': 43, 'stalk-surface-above-ring': 42, 'stalk-shape': 39, 'stalk-color-below-ring': 38, 'ring-number': 35, 'gill-attachment': 28, 'veil-color': 24
75	2	0.999077	'odor': 140, 'habitat': 109, 'stalk-root': 108, 'spore-print-color': 106, 'population': 98, 'cap-color': 96, 'gill-color': 93, 'cap-shape': 92, 'stalk-color-below-ring': 88, 'stalk-surface-below-ring': 87, 'ring-type': 76, 'gill-size': 72, 'cap-surface': 71, 'stalk-shape': 69, 'stalk-surface-above-ring': 66, 'bruises?': 58, 'stalk-color-above-ring': 57, 'gill-spacing': 49, 'ring-number': 43, 'gill-attachment': 16, 'veil-color': 11

Figure 32: Random Forests Mushroom dataset Tests 5/6

NumTrees	NumFeatures	Accuracy	Feature Counts
75	4	0.999538	'odor': 156, 'spore-print-color': 140, 'stalk-root': 114, 'habitat': 103, 'cap-color': 103, 'gill-color': 85, 'population': 73, 'gill-size': 71, 'stalk-surface-below-ring': 71, 'cap-surface': 65, 'stalk-shape': 61, 'ring-type': 58, 'bruises?': 58, 'cap-shape': 56, 'stalk-color-below-ring': 55, 'stalk-surface-above-ring': 45, 'gill-spacing': 43, 'ring-number': 35, 'stalk-color-above-ring': 35, 'veil-color': 6, 'gill-attachment': 3
75	-1	0.999538	'odor': 153, 'spore-print-color': 141, 'habitat': 112, 'gill-color': 101, 'stalk-root': 99, 'cap-color': 96, 'stalk-surface-below-ring': 82, 'population': 81, 'gill-size': 72, 'cap-surface': 71, 'ring-type': 59, 'stalk-color-below-ring': 58, 'gill-spacing': 55, 'stalk-shape': 54, 'bruises?': 52, 'stalk-surface-above-ring': 51, 'cap-shape': 43, 'stalk-color-above-ring': 36, 'ring-number': 34, 'veil-color': 7, 'gill-attachment': 5
100	1	0.964148	'cap-color': 68, 'odor': 65, 'stalk-color-below-ring': 64, 'spore-print-color': 63, 'stalk-surface-above-ring': 62, 'gill-color': 57, 'ring-type': 56, 'cap-surface': 55, 'stalk-root': 55, 'cap-shape': 52, 'habitat': 49, 'population': 48, 'bruises?': 47, 'stalk-color-above-ring': 47, 'stalk-surface-below-ring': 47, 'gill-size': 47, 'ring-number': 46, 'stalk-shape': 45, 'gill-spacing': 41, 'gill-attachment': 24, 'veil-color': 21
100	2	0.998923	'odor': 180, 'cap-color': 164, 'habitat': 161, 'spore-print-color': 152, 'population': 145, 'gill-color': 141, 'stalk-root': 135, 'stalk-surface-below-ring': 127, 'cap-surface': 107, 'stalk-color-below-ring': 104, 'stalk-color-above-ring': 103, 'cap-shape': 103, 'gill-size': 102, 'stalk-surface-above-ring': 102, 'ring-type': 95, 'bruises?': 87, 'gill-spacing': 75, 'ring-number': 69, 'stalk-shape': 65, 'veil-color': 34, 'gill-attachment': 15

Figure 33: Random Forests Mushroom dataset Tests 6/6

NumTrees	NumFeatures	Accuracy	Feature Counts
100	4	0.999538	'odor': 245, 'spore-print-color': 172, 'habitat': 143, 'stalk-root': 128, 'cap-color': 128, 'gill-color': 119, 'population': 116, 'stalk-surface-below-ring': 103, 'cap-surface': 99, 'gill-size': 87, 'ring-type': 86, 'cap-shape': 83, 'bruises?': 73, 'stalk-color-below-ring': 68, 'stalk-surface-above-ring': 67, 'stalk-shape': 64, 'gill-spacing': 59, 'stalk-color-above-ring': 49, 'ring-number': 49, 'veil-color': 7, 'gill-attachment': 3
100	-1	0.999846	'odor': 246, 'spore-print-color': 175, 'habitat': 151, 'population': 122, 'stalk-root': 120, 'cap-color': 118, 'gill-color': 107, 'ring-type': 92, 'gill-size': 89, 'stalk-surface-below-ring': 89, 'cap-surface': 85, 'stalk-surface-above-ring': 84, 'stalk-color-below-ring': 79, 'cap-shape': 79, 'stalk-shape': 68, 'gill-spacing': 62, 'bruises?': 58, 'ring-number': 58, 'stalk-color-above-ring': 45, 'veil-color': 4, 'gill-attachment': 3

Figure 34: Decision Forests Mushroom dataset Tests 1/5

NumTrees	NumFeatures	Accuracy	Feature Counts
1	5	0.948915	'cap-color': 9, 'spore-print-color': 6, 'ring-number': 4, 'veil-color': 1
1	11	1.000000	'cap-color': 4, 'odor': 3, 'stalk-root': 3, 'stalk-color-below-ring': 2, 'habitat': 2, 'bruises?': 1
1	16	0.999077	'odor': 3, 'habitat': 3, 'gill-color': 3, 'cap-shape': 3, 'stalk-color-above-ring': 2, 'stalk-surface-below-ring': 1, 'gill-size': 1, 'population': 1
1	0	1.000000	'odor': 3, 'spore-print-color': 1, 'stalk-surface-below-ring': 1, 'ring-number': 1, 'habitat': 1, 'cap-color': 1
10	5	0.917987	'cap-shape': 46, 'gill-color': 40, 'stalk-surface-below-ring': 19, 'cap-color': 18, 'stalk-root': 14, 'stalk-color-below-ring': 14, 'habitat': 10, 'spore-print-color': 7, 'ring-type': 7, 'ring-number': 6, 'cap-surface': 5, 'odor': 3, 'gill-size': 2, 'gill-spacing': 1
10	11	0.996153	'cap-color': 42, 'cap-shape': 17, 'stalk-root': 16, 'stalk-shape': 16, 'bruises?': 14, 'spore-print-color': 12, 'gill-color': 11, 'stalk-color-above-ring': 10, 'habitat': 9, 'population': 8, 'odor': 6, 'stalk-color-below-ring': 6, 'cap-surface': 6, 'ring-type': 5, 'gill-spacing': 4, 'veil-color': 3, 'ring-number': 3, 'stalk-surface-below-ring': 3, 'gill-size': 2, 'gill-attachment': 1
10	16	0.999538	'gill-color': 53, 'cap-color': 25, 'habitat': 19, 'cap-shape': 16, 'gill-spacing': 12, 'stalk-color-below-ring': 12, 'population': 11, 'ring-type': 10, 'stalk-root': 10, 'spore-print-color': 9, 'cap-surface': 9, 'odor': 9, 'ring-number': 7, 'stalk-shape': 6, 'stalk-surface-below-ring': 6, 'gill-size': 5, 'stalk-surface-above-ring': 4, 'bruises?': 4, 'gill-attachment': 1
10	0	0.998461	'cap-color': 28, 'stalk-color-below-ring': 19, 'stalk-root': 15, 'spore-print-color': 13, 'odor': 12, 'gill-color': 11, 'ring-type': 11, 'stalk-surface-below-ring': 10, 'bruises?': 9, 'gill-spacing': 8, 'population': 6, 'cap-shape': 6, 'ring-number': 5, 'stalk-shape': 5, 'cap-surface': 4, 'stalk-surface-above-ring': 4, 'habitat': 4, 'gill-size': 3

Figure 35: Decision Forests Mushroom dataset Tests 2/5

NumTrees	NumFeatures	Accuracy	Feature Counts
25	5	0.958147	'cap-shape': 60, 'cap-color': 40, 'gill-color': 35, 'population': 33, 'stalk-color-above-ring': 33, 'stalk-surface-below-ring': 27, 'spore-print-color': 23, 'bruises?': 21, 'habitat': 16, 'stalk-root': 14, 'odor': 12, 'cap-surface': 12, 'ring-type': 9, 'stalk-color-below-ring': 9, 'gill-size': 7, 'stalk-surface-above-ring': 6, 'ring-number': 5, 'gill-attachment': 4, 'stalk-shape': 1, 'gill-spacing': 1
25	11	1.000000	'gill-color': 103, 'cap-color': 72, 'cap-shape': 52, 'stalk-color-below-ring': 41, 'population': 38, 'habitat': 32, 'stalk-root': 31, 'spore-print-color': 26, 'stalk-color-above-ring': 21, 'odor': 18, 'bruises?': 17, 'ring-number': 16, 'stalk-surface-below-ring': 14, 'gill-spacing': 14, 'cap-surface': 13, 'stalk-shape': 12, 'stalk-surface-above-ring': 8, 'gill-size': 8, 'veil-color': 7, 'ring-type': 4, 'gill-attachment': 2
25	16	0.999538	'gill-color': 63, 'habitat': 55, 'cap-color': 47, 'stalk-root': 28, 'spore-print-color': 26, 'odor': 24, 'cap-shape': 23, 'stalk-shape': 17, 'stalk-color-below-ring': 16, 'cap-surface': 16, 'stalk-surface-above-ring': 14, 'ring-type': 14, 'bruises?': 14, 'population': 13, 'stalk-color-above-ring': 12, 'stalk-surface-below-ring': 11, 'gill-spacing': 10, 'gill-size': 7, 'gill-attachment': 4, 'ring-number': 4, 'veil-color': 1
25	0	0.999077	'cap-color': 70, 'cap-surface': 49, 'habitat': 42, 'gill-color': 34, 'stalk-color-above-ring': 26, 'stalk-color-below-ring': 25, 'spore-print-color': 23, 'stalk-root': 23, 'odor': 21, 'population': 20, 'stalk-surface-above-ring': 19, 'stalk-shape': 17, 'cap-shape': 14, 'stalk-surface-below-ring': 13, 'bruises?': 10, 'gill-spacing': 9, 'ring-type': 8, 'ring-number': 8, 'gill-size': 3, 'gill-attachment': 1, 'veil-color': 1
50	5	0.933836	'cap-shape': 91, 'cap-color': 65, 'gill-color': 65, 'stalk-color-below-ring': 55, 'stalk-surface-below-ring': 42, 'stalk-surface-above-ring': 36, 'spore-print-color': 35, 'cap-surface': 31, 'odor': 30, 'habitat': 28, 'ring-type': 28, 'gill-spacing': 27, 'stalk-color-above-ring': 22, 'population': 20, 'stalk-shape': 15,

Figure 36: Decision Forests Mushroom dataset Tests 3/5

NumTrees	NumFeatures	Accuracy	Feature Counts
50	11	0.999385	'gill-color': 114, 'cap-color': 112, 'cap-shape': 90, 'spore-print-color': 88, 'cap-surface': 73, 'odor': 65, 'habitat': 53, 'population': 51, 'stalk-color-below-ring': 48, 'stalk-surface-below-ring': 47, 'stalk-surface-above-ring': 42, 'gill-spacing': 40, 'stalk-root': 38, 'ring-number': 29, 'stalk-color-above-ring': 24, 'ring-type': 23, 'bruises?': 22, 'gill-size': 16, 'stalk-shape': 12, 'veil-color': 11, 'gill-attachment': 5
50	16	0.999538	'gill-color': 110, 'cap-color': 93, 'cap-shape': 71, 'habitat': 65, 'stalk-surface-below-ring': 56, 'spore-print-color': 55, 'odor': 52, 'cap-surface': 52, 'stalk-root': 51, 'gill-spacing': 45, 'population': 38, 'stalk-color-below-ring': 33, 'bruises?': 30, 'ring-type': 29, 'stalk-surface-above-ring': 25, 'stalk-shape': 23, 'ring-number': 22, 'stalk-color-above-ring': 22, 'gill-size': 13, 'gill-attachment': 8, 'veil-color': 4
50	0	0.998923	'gill-color': 103, 'cap-color': 99, 'cap-shape': 79, 'spore-print-color': 58, 'habitat': 57, 'stalk-root': 49, 'population': 42, 'odor': 38, 'cap-surface': 33, 'gill-spacing': 26, 'ring-number': 24, 'ring-type': 24, 'bruises?': 22, 'stalk-color-above-ring': 20, 'stalk-surface-below-ring': 20, 'stalk-surface-above-ring': 17, 'stalk-color-below-ring': 16, 'stalk-shape': 11, 'gill-size': 11, 'veil-color': 6, 'gill-attachment': 5
75	5	0.959532	'gill-color': 218, 'cap-color': 174, 'spore-print-color': 118, 'cap-shape': 102, 'population': 91, 'habitat': 78, 'stalk-surface-below-ring': 56, 'stalk-root': 51, 'stalk-color-above-ring': 41, 'stalk-surface-above-ring': 35, 'stalk-color-below-ring': 34, 'cap-surface': 32, 'bruises?': 32, 'ring-type': 27, 'stalk-shape': 26, 'odor': 23, 'gill-size': 18, 'ring-number': 18, 'veil-color': 14, 'gill-spacing': 12, 'gill-attachment': 10

Figure 37: Decision Forests Mushroom dataset Tests 4/5

NumTrees	NumFeatures	Accuracy	Feature Counts
75	11	0.996307	'gill-color': 188, 'cap-color': 161, 'cap-shape': 129, 'cap-surface': 93, 'stalk-surface-below-ring': 85, 'habitat': 80, 'spore-print-color': 71, 'ring-number': 66, 'population': 56, 'odor': 56, 'stalk-color-below-ring': 54, 'stalk-root': 54, 'ring-type': 53, 'stalk-surface-above-ring': 53, 'bruises?': 40, 'stalk-color-above-ring': 33, 'gill-size': 30, 'gill-spacing': 30, 'stalk-shape': 29, 'veil-color': 16, 'gill-attachment': 10
75	16	0.999538	'gill-color': 168, 'cap-color': 156, 'population': 105, 'spore-print-color': 104, 'cap-surface': 91, 'odor': 90, 'cap-shape': 77, 'stalk-color-below-ring': 74, 'stalk-surface-below-ring': 71, 'habitat': 68, 'stalk-root': 64, 'ring-type': 63, 'stalk-surface-above-ring': 55, 'stalk-color-above-ring': 40, 'bruises?': 40, 'ring-number': 36, 'gill-spacing': 32, 'stalk-shape': 31, 'gill-size': 20, 'gill-attachment': 5, 'veil-color': 4
75	0	0.997384	'gill-color': 194, 'cap-color': 137, 'cap-shape': 131, 'habitat': 92, 'population': 79, 'cap-surface': 76, 'stalk-root': 73, 'stalk-color-above-ring': 59, 'spore-print-color': 54, 'odor': 53, 'bruises?': 53, 'gill-spacing': 46, 'ring-type': 42, 'stalk-shape': 40, 'stalk-surface-above-ring': 39, 'stalk-color-below-ring': 39, 'ring-number': 36, 'stalk-surface-below-ring': 35, 'gill-size': 15, 'gill-attachment': 9, 'veil-color': 3
100	5	0.974150	'cap-color': 220, 'gill-color': 167, 'cap-shape': 144, 'cap-surface': 125, 'stalk-color-below-ring': 98, 'habitat': 93, 'spore-print-color': 82, 'ring-type': 74, 'stalk-surface-below-ring': 74, 'stalk-color-above-ring': 71, 'stalk-surface-above-ring': 71, 'population': 70, 'stalk-root': 62, 'ring-number': 55, 'odor': 51, 'gill-spacing': 48, 'bruises?': 37, 'stalk-shape': 32, 'gill-size': 26, 'veil-color': 22, 'gill-attachment': 16

Figure 38: Decision Forests Mushroom dataset Tests 5/5

NumTrees	NumFeatures	Accuracy	Feature Counts
100	11	0.998615	'cap-color': 316, 'gill-color': 278, 'cap-shape': 184, 'spore-print-color': 143, 'habitat': 137, 'cap-surface': 124, 'population': 106, 'stalk-surface-below-ring': 104, 'stalk-color-below-ring': 90, 'stalk-root': 87, 'stalk-color-above-ring': 69, 'ring-type': 68, 'gill-spacing': 61, 'odor': 61, 'bruises?': 55, 'stalk-shape': 52, 'ring-number': 51, 'stalk-surface-above-ring': 51, 'gill-size': 31, 'gill-attachment': 10, 'veil-color': 4
100	16	1.000000	'gill-color': 251, 'cap-color': 201, 'habitat': 195, 'cap-shape': 135, 'cap-surface': 130, 'odor': 108, 'population': 104, 'stalk-root': 104, 'stalk-surface-below-ring': 103, 'spore-print-color': 99, 'stalk-surface-above-ring': 93, 'bruises?': 85, 'ring-type': 77, 'gill-spacing': 64, 'ring-number': 57, 'stalk-color-below-ring': 53, 'stalk-shape': 49, 'gill-size': 36, 'stalk-color-above-ring': 29, 'veil-color': 12, 'gill-attachment': 5
100	0	0.999385	'gill-color': 313, 'cap-shape': 194, 'cap-color': 152, 'spore-print-color': 130, 'habitat': 120, 'cap-surface': 116, 'population': 100, 'odor': 93, 'stalk-root': 80, 'bruises?': 75, 'stalk-color-below-ring': 69, 'stalk-color-above-ring': 61, 'stalk-surface-above-ring': 60, 'ring-number': 50, 'stalk-surface-below-ring': 43, 'gill-spacing': 36, 'stalk-shape': 32, 'gill-size': 31, 'ring-type': 29, 'gill-attachment': 12, 'veil-color': 10