LABWORK 4

DECISION TREE
&
RANDOM FOREST

Mai Hải Đăng - BI12-076

26.12.2023

Machine Learning and Data Mining II

Table of Contents

1. Introduction 3
2. Concepts 3
   2.1. Ensemble Models 3
   2.2. Decision Tree 3
   2.3. Random Forest 4
3. Data Analysis 5
   3.1. Iris 5
   3.2. Breast Cancer Wisconsin (Diagnostic) 6
4. Experiments and Evaluation 10
   4.1. Data Pre-Processing 10
   4.2. Decision Tree 11
   4.3. Random Forest 13
5. Evaluation 14

1. Introduction

- This report is dedicated to the study and understanding of Decision Tree (DT) and Random Forest
  - Definitions, principles, and algorithms used.
  - Evaluation methods.
- The original dataset are Iris [1], Breast Cancer Wisconsin (Diagnostic) [2],

2. Concepts

2.1. Ensemble Models

- Supervised learning algorithms perform the task of searching through a hypothesis space to find a suitable hypothesis that will make good predictions with a particular problem. Ensembles combine multiple hypotheses to form a better hypothesis.
- In simpler terms, ensemble models consist of various learning models to obtain better predictive performance. And models can be similar types or different.
  - Advantages:
    - Avoid overfitting problems.
    - Robust to outliers and noise.
  - Disadvantages:
    - Not a compact presentation.
    - Not easy to compute the complexity.
    - Intuitively difficult to understand.
- Fast algorithms such as decision trees are commonly used in ensemble models i.e., random forests.

2.2. Decision Tree

2.2.1. Principles

- A predictive model uses a set of binary rules applied to calculate a target value.
- Can be used for classification (categorical variables) or regression (continuous variables) applications.
- A tree is built by splitting the source set, constituting the root node of the tree into subsets which constitute the successor children of:
  - Internal nodes represent a split rule: the feature is used to split and the value of the split.
  - Terminal (leaf) node consists of a data object of a class. Each leaf node is marked with a class label.

2.2.2. Algorithm

- INPUT $X = \{x_1, \ldots, x_n\} \in R^d$

1. Create K bootstrap sets $X_1, X_2, \ldots, X_k$ from the input sample X.
2. Learn an unpruned decision tree on each learning set. Learning: at each internal node
   a. Randomly select m < d features
   b. Determine the best split using only selected features.

i. Choose the feature providing the highest information to obtain to split the training set into two subsets.
    1. Gini impurity
    2. Distance measure
    3. Information gained.
ii. Construct child nodes after splitting, each one contains a respective subset.
iii. Repeat i and ii until each child node consist of only instances from a pure class (leaf node)

3. Predict instance class, each instance from the test set.
    a. In each tree built in forest, it moves top-down from the root and gets verified by the rules in internal nodes.
    b. When it reaches a leaf node, this instance will be assigned to the class label of this leaf.
    c. A majority vote is performed. The majority class will finally decide to label this instance.

2.2.3. Determine Best Split

- For a set of items with J classes and relative frequencies $p_i, i \in \{1,2,\dots,J\}$, the probability of choosing an item with label i
- Gini Impurity (GI)

$$GI = \sum_{i=1}^{J} p_i(1 - p_i)$$

- Information Gain (IG)

$$IG = E_0 - \sum_{i=1}^{k} \frac{n_i}{n} log_2(E_i)$$

$$E_0 = -\sum_{j=1}^{J} \frac{n_j}{n} log_2\left(\frac{n_j}{n}\right), E_i = -\sum_{j=1}^{J} \frac{n_{j_i}}{n_i} log_2\left(\frac{n_{j_i}}{n_i}\right)$$

2.2.4. Decision Tree Pruning

- Puring reduces the size of DT by removing parts of tree that do not provide information, reducing the complexity of the final classifier, and hence improves predictive accuracy by the reduction of overfitting.
- Pruning processes
    - Pre-pruning: Prevent a complete induction of the training set by replace a stop criterion in induction algorithm (i.e., IG (attr) > minGain)
    - Post-pruning: Simplifying trees, nodes and subtrees are replaced with leaves to reduce complexity.

2.3. Random Forest

- Consist of an ensemble of decision trees (see 2.2) such that each tree represents a classifier.
- The generalization error of a forest depends on the strength of the individual trees in the forest and the correlation between them.

3. Data Analysis

3.1. Iris

- The dataset contains 150 instances and 4 features. Each instance is a plant, and the Predicted attributes are classes of iris plant.

Table 1. Characteristics of Iris dataset.

Data	Role	Data Type	Mean	Variance
Sepal length	Feature	Continuous, Quantitative	5.843333	0.685694
Sepal width	Feature	Continuous, Quantitative	3.057333	0.189979
Petal length	Feature	Continuous, Quantitative	3.758000	3.116278
Petal width	Feature	Continuous, Quantitative	1.199333	0.581006
Class	Target	Categorical, Qualitative		

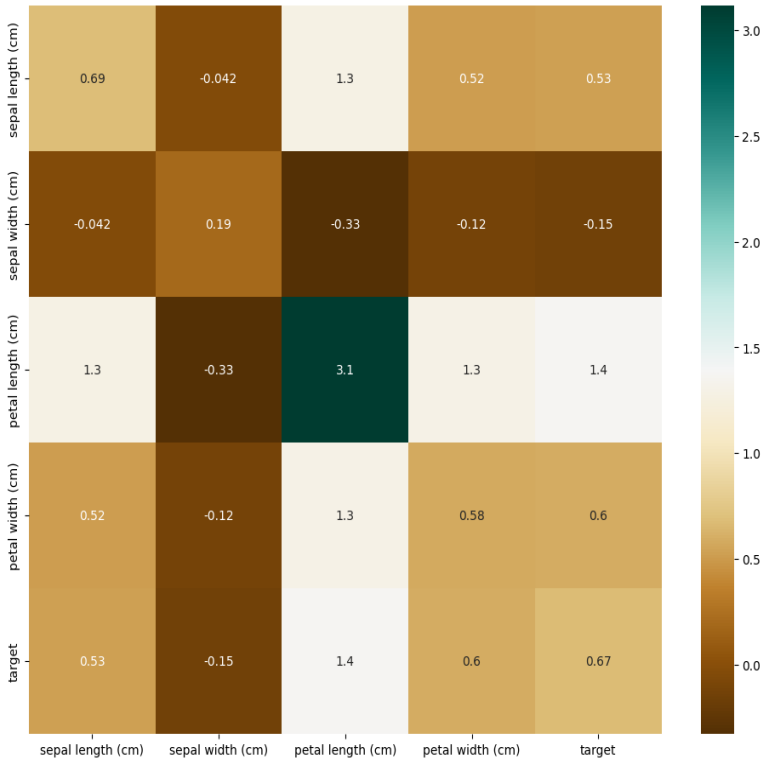

Figure 1. Iris dataset. Correlation matrix.

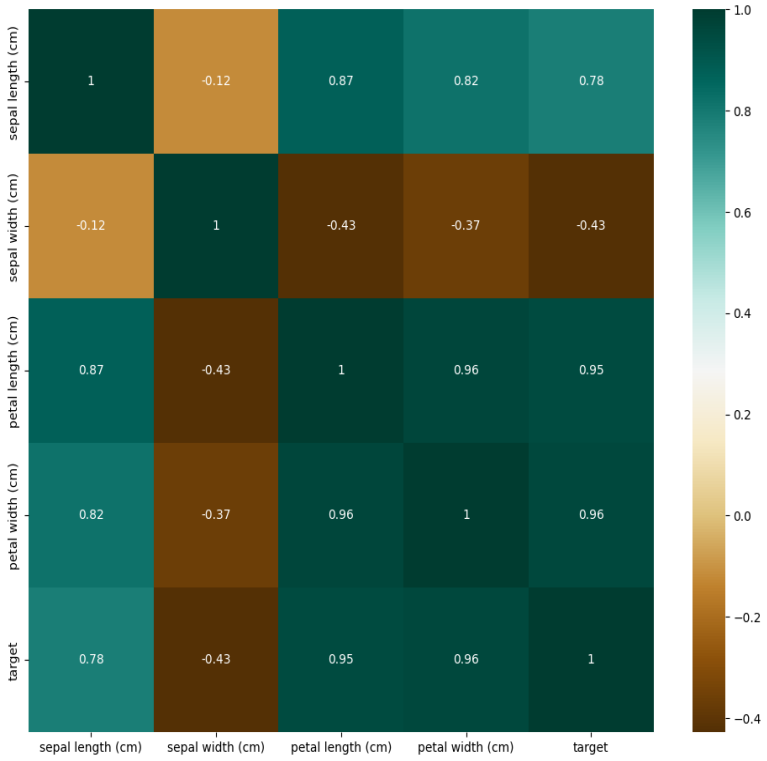

Figure 2. Iris dataset. Covariance matrix.

3.2. Breast Cancer Wisconsin (Diagnostic)

- The dataset contains 569 instances and 30 features. Features are computed from a digitized image of a fine needle aspirate (FNA) of a breast mass. They describe characteristics of the cell nuclei present in the image.

Table 2. Characteristics of Breast Cancer Wisconsin (Diagnostic) dataset.

Data	Role	Data Type	Mean	Variance
mean radius	Feature	Continuous, Quantitative	14.12729	12.41892
mean texture	Feature	Continuous, Quantitative	19.28965	18.49891
mean perimeter	Feature	Continuous, Quantitative	91.96903	590.4405
mean area	Feature	Continuous, Quantitative	654.8891	123843.6

mean smoothness	Feature	Continuous, Quantitative	0.09636	0.000198
mean compactness	Feature	Continuous, Quantitative	0.104341	0.002789
mean concavity	Feature	Continuous, Quantitative	0.088799	0.006355
mean concave points	Feature	Continuous, Quantitative	0.048919	0.001506
mean symmetry	Feature	Continuous, Quantitative	0.181162	0.000752
mean fractal dimension	Feature	Continuous, Quantitative	0.062798	4.98E-05
radius error	Feature	Continuous, Quantitative	0.405172	0.076902
texture error	Feature	Continuous, Quantitative	1.216853	0.304316
perimeter error	Feature	Continuous, Quantitative	2.866059	4.087896
area error	Feature	Continuous, Quantitative	40.33708	2069.432
smoothness error	Feature	Continuous, Quantitative	0.007041	9.02E-06
compactness error	Feature	Continuous, Quantitative	0.025478	0.000321
concavity error	Feature	Continuous, Quantitative	0.031894	0.000911
concave points error	Feature	Continuous, Quantitative	0.011796	3.81E-05
symmetry error	Feature	Continuous, Quantitative	0.020542	6.83E-05
fractal dimension error	Feature	Continuous, Quantitative	0.003795	7E-06

worst radius	Feature	Continuous, Quantitative	16.26919	23.36022
worst texture	Feature	Continuous, Quantitative	25.67722	37.77648
worst perimeter	Feature	Continuous, Quantitative	107.2612	1129.131
worst area	Feature	Continuous, Quantitative	880.5831	324167.4
worst smoothness	Feature	Continuous, Quantitative	0.132369	0.000521
worst compactness	Feature	Continuous, Quantitative	0.254265	0.024755
worst concavity	Feature	Continuous, Quantitative	0.272188	0.043524
worst concave points	Feature	Continuous, Quantitative	0.114606	0.004321
worst symmetry	Feature	Continuous, Quantitative	0.290076	0.003828
worst fractal dimension	Feature	Continuous, Quantitative	0.083946	0.000326
Class	Target	Categorical, Qualitative		

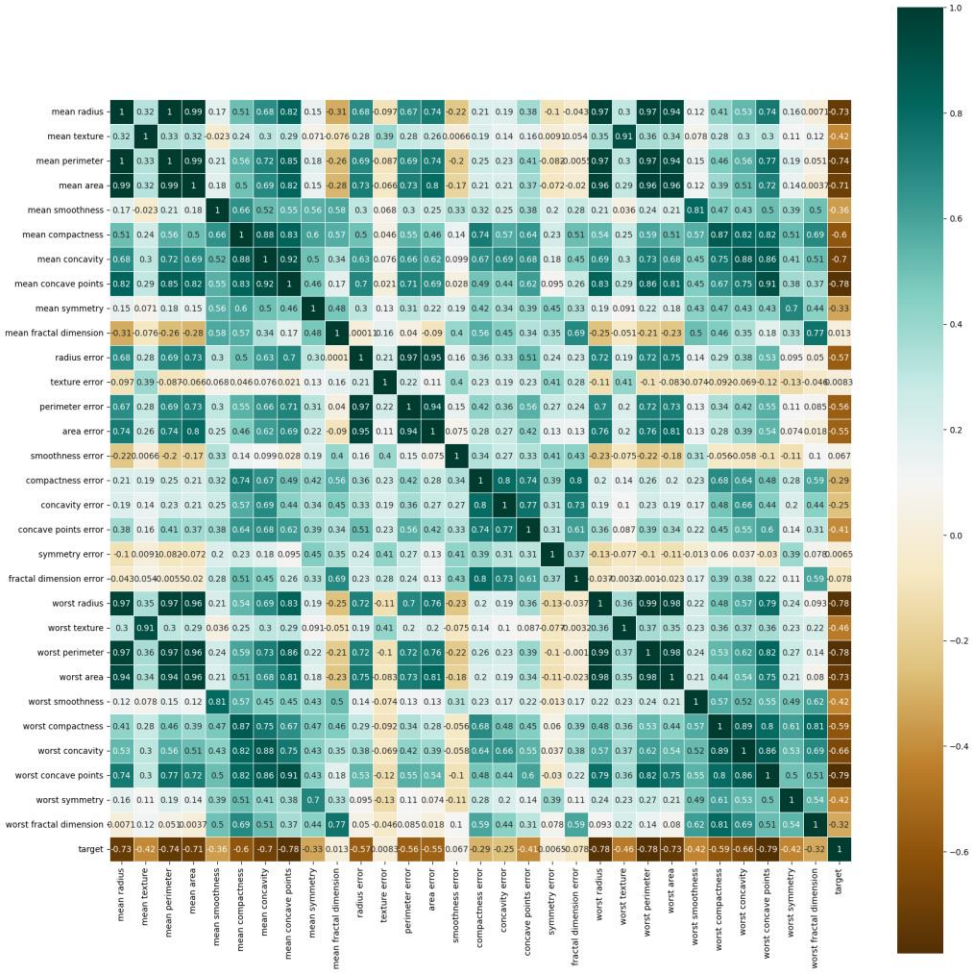

Figure 3. Breast Cancer Wisconsin (Diagnostic) dataset. Correlation matrix.

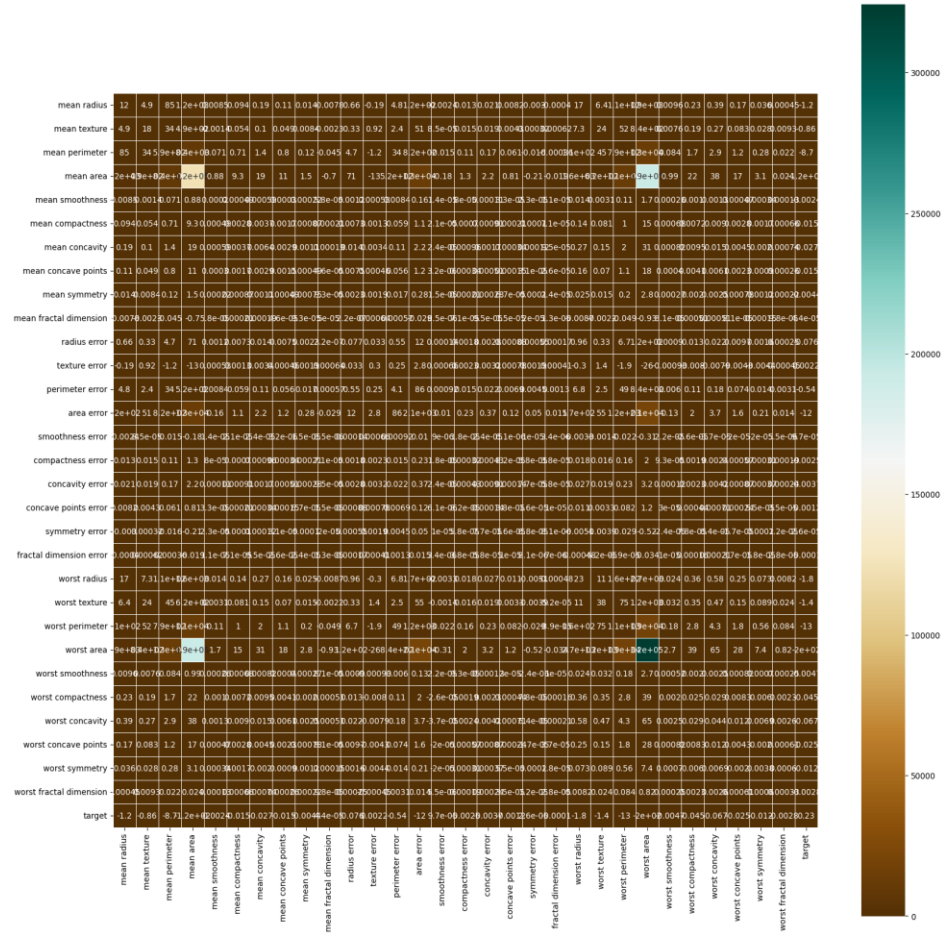

Figure 4. Breast Cancer Wisconsin (Diagnostic) dataset. Covariance matrix.

4. Experiments and Evaluation

4.1. Data Pre-Processing

- For experiment with **random forest**, we will use **Breast Cancer Wisconsin (Diagnostic) dataset**. We apply K-Fold cross validation that randomly partitioned into k equal sized subsamples, often referred to as "folds" in order to split the data into 100 training set and 1 testing set.
- For experiment with **decision trees**, we will use **Iris dataset**. We apply a simple data split into two subsets: one for training (80%) and one for testing (20%)

4.2. Decision Tree

- We will use Sklearn Decision Tree, and Sklearn Decision Boundary implementation for decision tree, plot the decision boundary of the tree and the logic split of the tree.
    - Selecting all features.
    - No maximum depth for tree.
    - Evaluation method for splitting: Information Gain

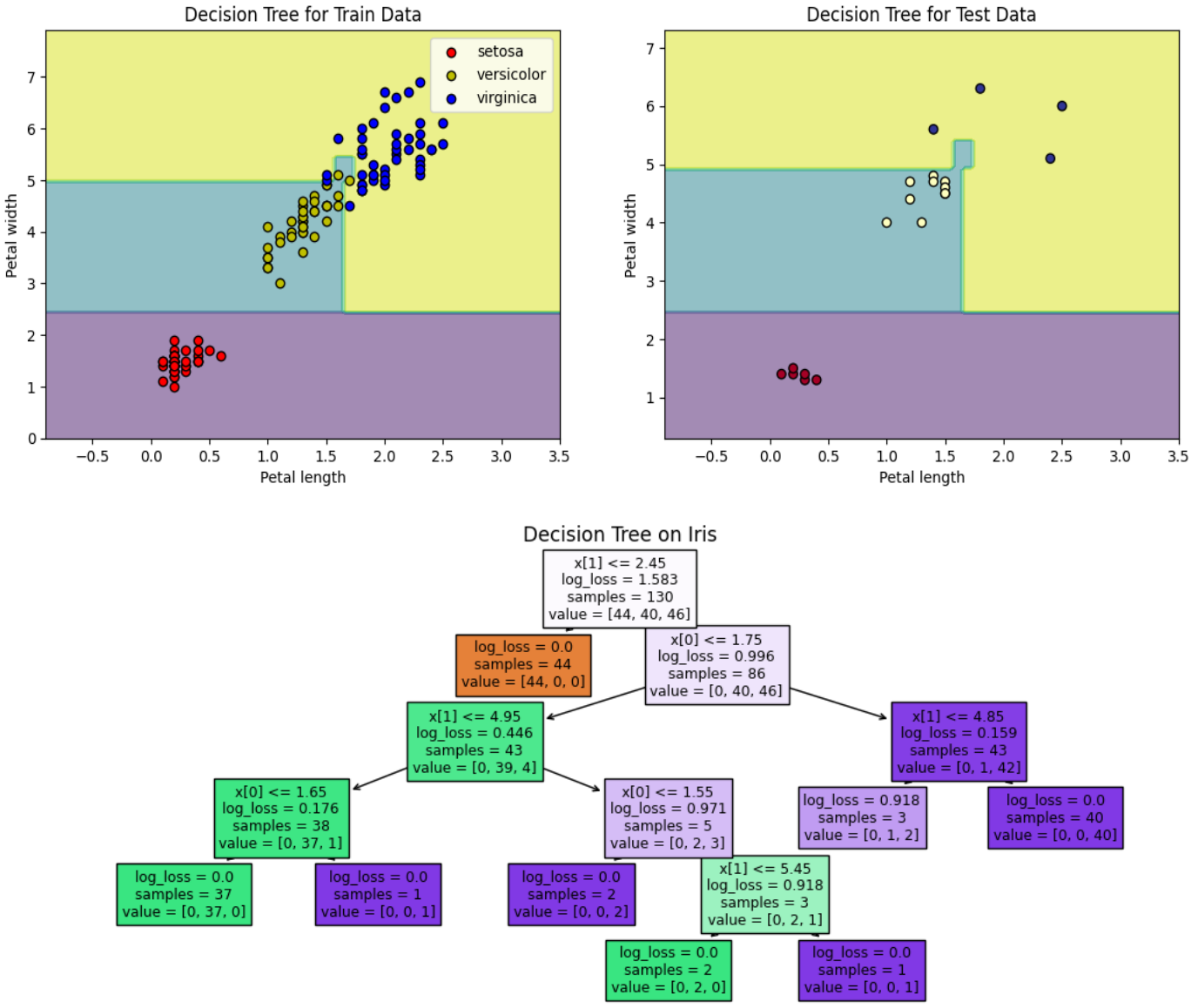

Figure 5. Common output (1/2) of Decision Tree on Iris Dataset

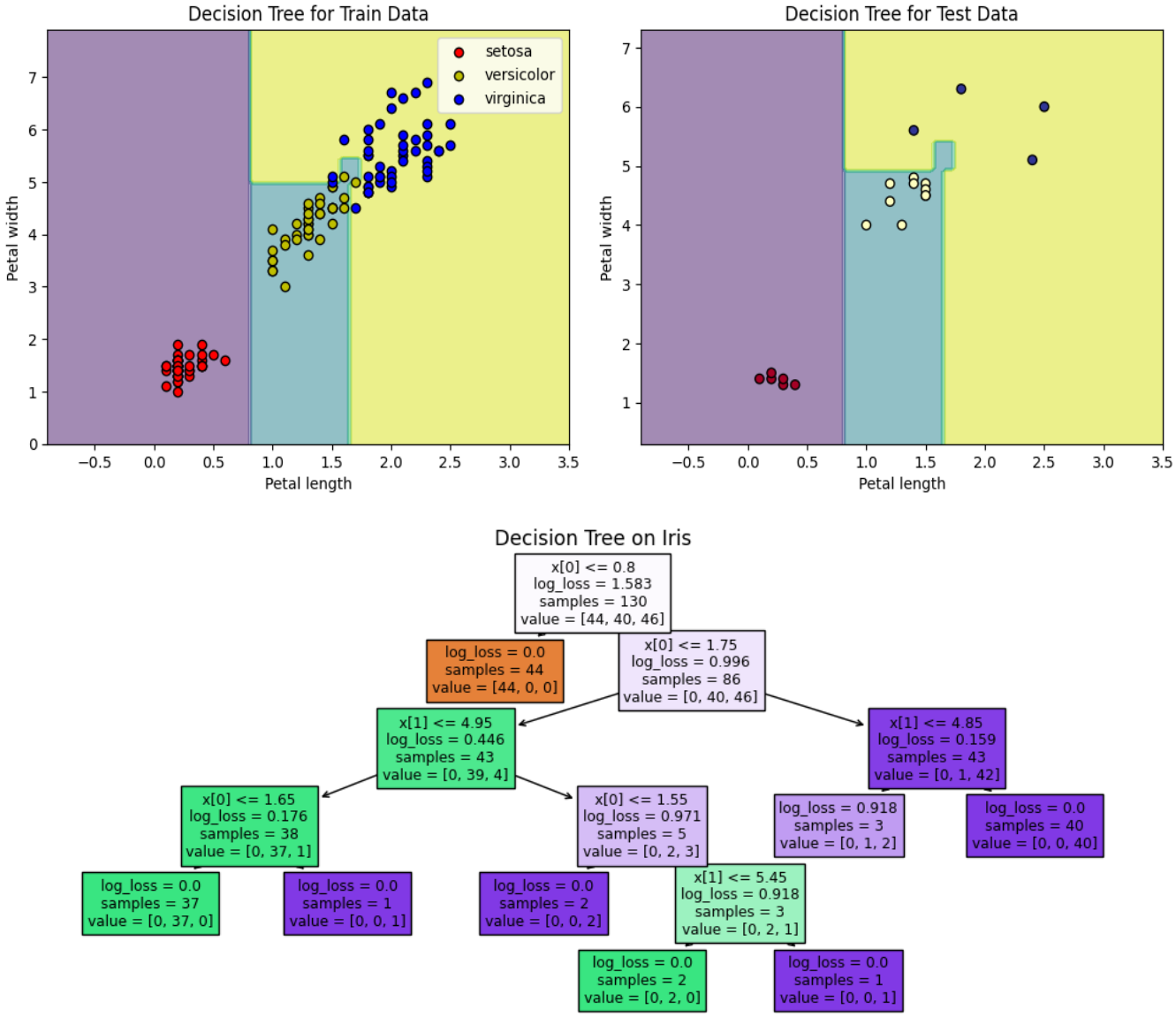

Figure 6. Common output (2/2) of Decision Tree on Iris Dataset.

4.3. Random Forest

- Apply similar logic code as section 4.2 with many decision trees.

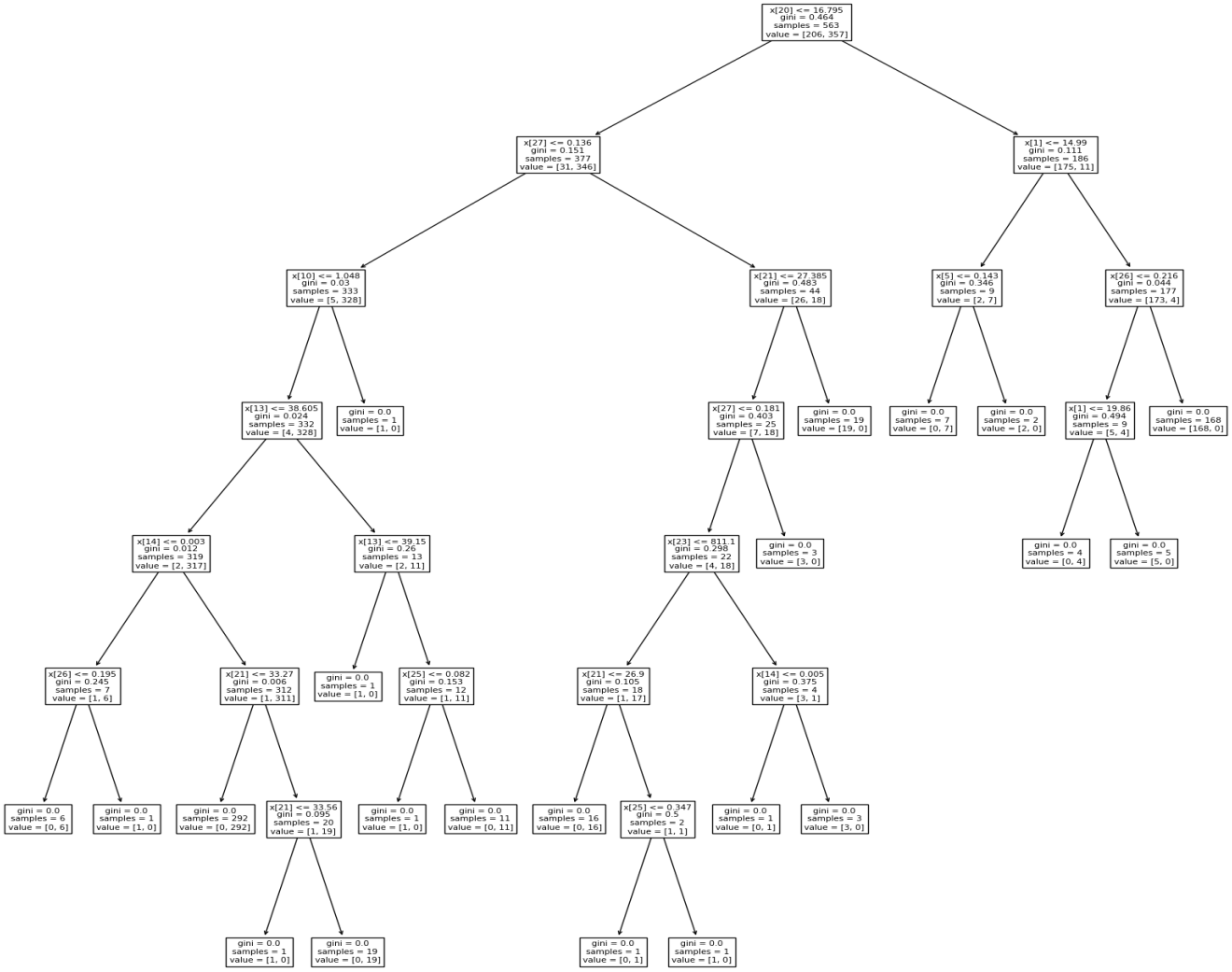

Figure 7. Splitting logic of a random tree inside the forest.

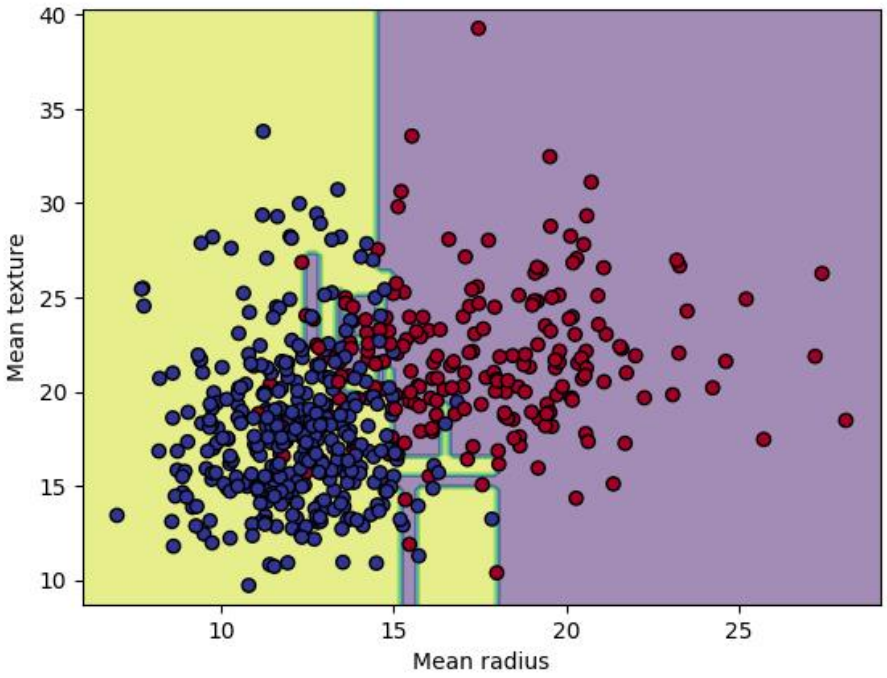

Figure 8. Decision boundary of a random tree inside a forest for the two features: Mean texture and mean radius of the dataset.

5. Evaluation

Table 3. Classification error for training and testing set of Decision Tree experiment on Iris Dataset

Set	Explained Variance
Training	0.992
Testing	1.000

Table 4. Classification error for sets of decision tree inside the Random Forest experiment on Breast Cancer Wisconsin (Diagnostic) dataset

Set	Explained Variance
One tree	0.000
Mean 100 trees	0.000

References

[1] Fisher,R. A.. (1988). Iris. UCI Machine Learning Repository. https://doi.org/10.24432/C56C76.

[2] Wolberg,William, Mangasarian,Olvi, Street,Nick, and Street,W.. (1995). Breast Cancer Wisconsin (Diagnostic). UCI Machine Learning Repository. https://doi.org/10.24432/C5DW2B.