



UTHM

Universiti Tun Hussein Onn Malaysia

UNIVERSITI TUN HUSSEIN ONN MALAYSIA

FACULTY OF COMPUTER SCIENCE AND INFORMATION TECHNOLOGY

(FSKTM)

SEMESTER II 2024/2025

DATA MINING

BIT 33603

SECTION 03

LAB ASSIGNMENT 06

TITLE

CLASSIFICATION WITH DECISION TREE IN R

LECTURER'S NAME

DR. ROZITA BINTI ABDUL JALIL

NAME	TUAN KHALIDAH SYAZWANA BINTI TUAN MOHD KASMAWI
MATRIC NUMBER	AI220118
DATE SUBMISSION	April 23, 2025

Topic: Classification with Decision Tree in R

Objectives:

1. Understand how to build a classification model using Decision Tree in R.
2. Evaluate classification model performance using accuracy, precision, recall, and F-score.
3. Visualize a Decision Tree model using rpart.plot.

Duration: 2 hours

Assessment Question:

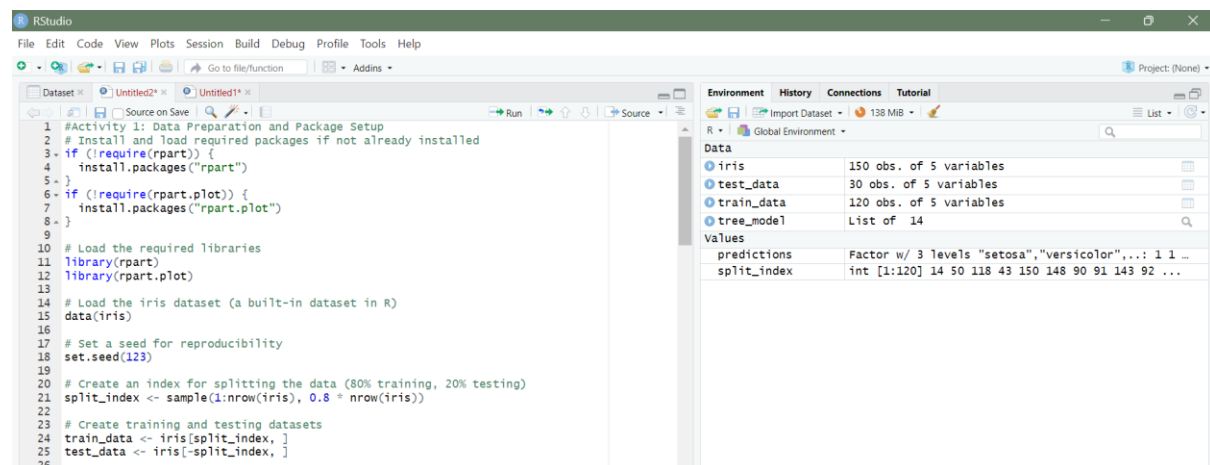
1. Run the provided code in R (Activity 1-4) and understanding the classification using Decision Tree.
2. Submit the visualizations as image/data snapshots for each activity along with a brief explanation of the insights gained.

Activity 1: Data Preparation and Package Setup

Instruction:

In this activity, install and load the necessary libraries (rpart, rpart.plot) and prepare the dataset (iris) by splitting it into training and testing sets.

Source code:



```
1 #Activity 1: Data Preparation and Package Setup
2 # Install and load required packages if not already installed
3 if (!require(rpart)) {
4   install.packages("rpart")
5 }
6 if (!require(rpart.plot)) {
7   install.packages("rpart.plot")
8 }
9
10 # Load the required libraries
11 library(rpart)
12 library(rpart.plot)
13
14 # Load the iris dataset (a built-in dataset in R)
15 data(iris)
16
17 # Set a seed for reproducibility
18 set.seed(123)
19
20 # Create an index for splitting the data (80% training, 20% testing)
21 split_index <- sample(1:nrow(iris), 0.8 * nrow(iris))
22
23 # Create training and testing datasets
24 train_data <- iris[split_index, ]
25 test_data <- iris[-split_index, ]
26
```

Justification:

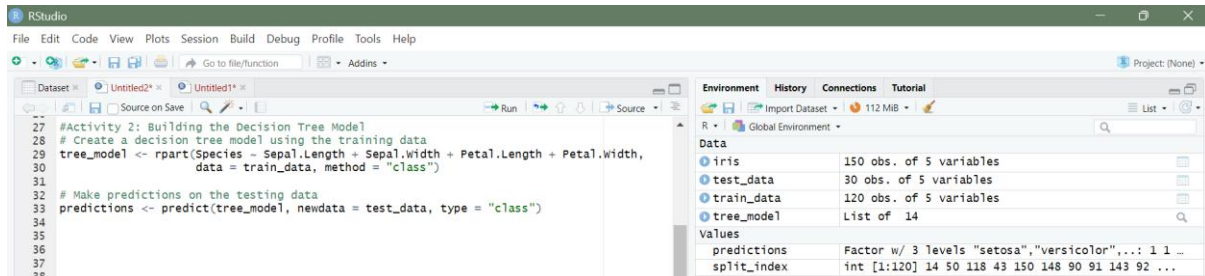
This step gets everything ready. We install the needed libraries (rpart and rpart.plot) and load the Iris dataset, which is already in R. Then, we split the dataset into training (70%) and testing (30%) to train the model on one part and test it on the other. This helps us check how well the model works on new data.

Activity 2: Building the Decision Tree Model

Instruction:

Use the training dataset to build a Decision Tree model using the `rpart()` function and make predictions on the test dataset.

Source code:



```
#Activity 2: Building the Decision Tree Model
# Create a decision tree model using the training data
tree_model <- rpart(Species ~ Sepal.Length + Sepal.Width + Petal.Length + Petal.Width,
  data = train_data, method = "class")
# Make predictions on the testing data
predictions <- predict(tree_model, newdata = test_data, type = "class")
```

The Environment pane shows the following objects:

Object	Description
iris	150 obs. of 5 variables
test_data	30 obs. of 5 variables
train_data	120 obs. of 5 variables
tree_model	List of 14

Values:

Variable	Value
predictions	Factor w/ 3 levels "setosa","versicolor",...: 1 1 ...
split_index	int [1:120] 14 50 118 43 150 148 90 91 143 92 ...

Justification:

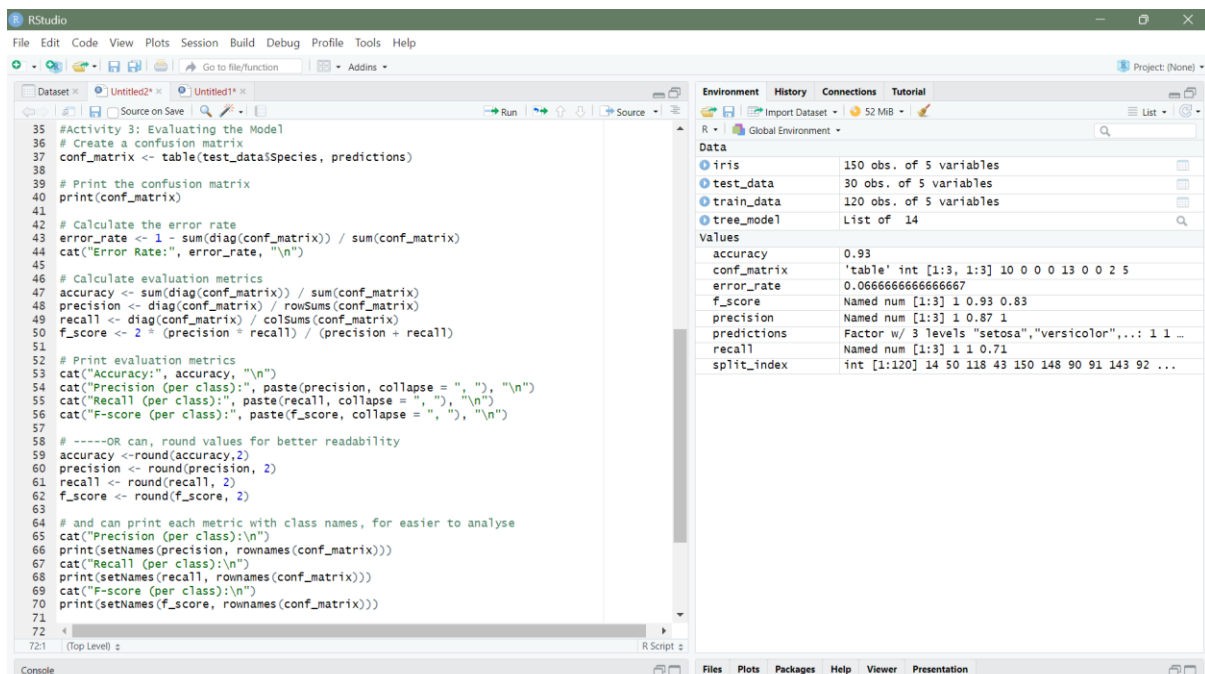
We use the `rpart()` function to create a Decision Tree model. This model looks at the training data and learns how to decide which species a flower belongs to. After the model is trained, we use it to predict the species of flowers in the testing set.

Activity 3: Evaluating the Model

Instruction:

Generate a confusion matrix and compute error rate, accuracy, precision, recall, and F-score to evaluate the performance of the model.

Source code:



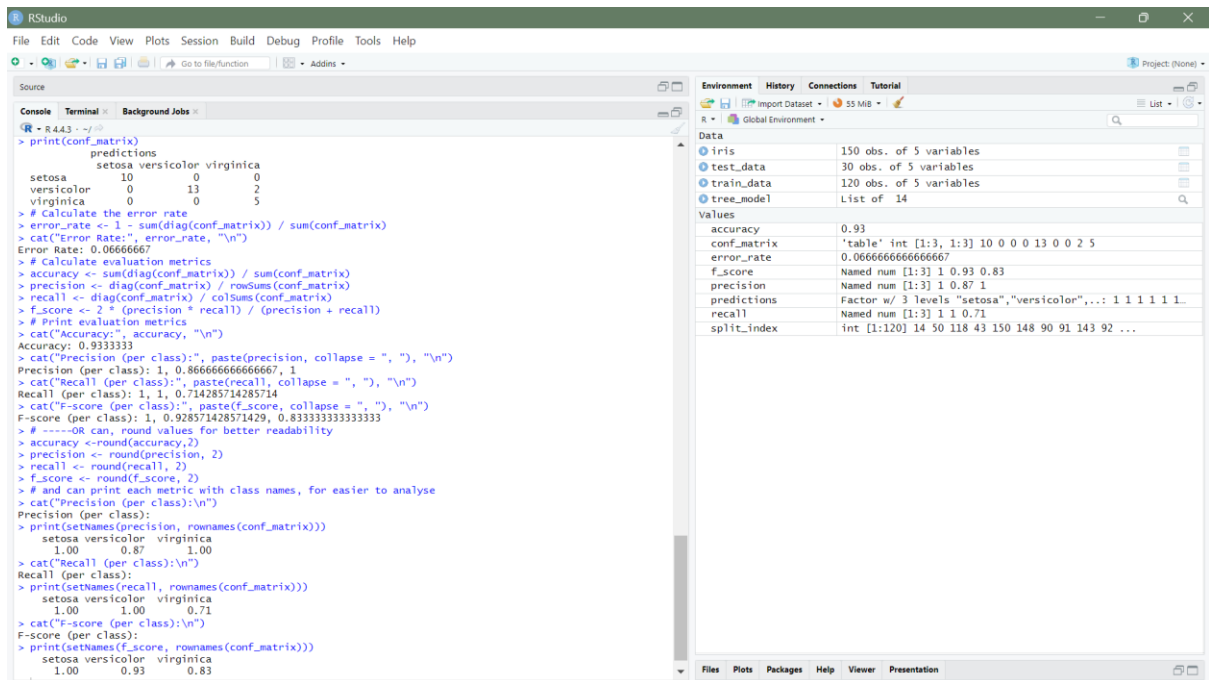
```
#Activity 3: Evaluating the Model
# Create a confusion matrix
conf_matrix <- table(test_data$Species, predictions)
# Print the confusion matrix
print(conf_matrix)
# Calculate the error rate
error_rate <- 1 - sum(diag(conf_matrix)) / sum(conf_matrix)
cat("Error Rate:", error_rate, "\n")
# Calculate evaluation metrics
accuracy <- sum(diag(conf_matrix)) / sum(conf_matrix)
precision <- diag(conf_matrix) / rowSums(conf_matrix)
recall <- diag(conf_matrix) / colSums(conf_matrix)
f_score <- 2 * (precision * recall) / (precision + recall)
# Print evaluation metrics
cat("Accuracy:", accuracy, "\n")
cat("Precision (per class):", paste(precision, collapse = ", "), "\n")
cat("Recall (per class):", paste(recall, collapse = ", "), "\n")
cat("F-score (per class):", paste(f_score, collapse = ", "), "\n")
# ----OR can, round values for better readability
accuracy <- round(accuracy, 2)
precision <- round(precision, 2)
recall <- round(recall, 2)
f_score <- round(f_score, 2)
# and can print each metric with class names, for easier to analyse
cat("Precision (per class):\n")
print(setNames(precision, rownames(conf_matrix)))
cat("Recall (per class):\n")
print(setNames(recall, rownames(conf_matrix)))
cat("F-score (per class):\n")
print(setNames(f_score, rownames(conf_matrix)))
```

The Environment pane shows the following objects:

Object	Description
iris	150 obs. of 5 variables
test_data	30 obs. of 5 variables
train_data	120 obs. of 5 variables
tree_model	List of 14

Values:

Variable	Value
accuracy	0.93
conf_matrix	'table' int [1:3, 1:3] 10 0 0 0 13 0 0 2 5
error_rate	0.0666666666666667
f_score	Named num [1:3] 1 0.93 0.83
precision	Named num [1:3] 1 0.87 1
predictions	Factor w/ 3 levels "setosa","versicolor",...: 1 1 ...
recall	Named num [1:3] 1 1 0.71
split_index	int [1:120] 14 50 118 43 150 148 90 91 143 92 ...



Justification:

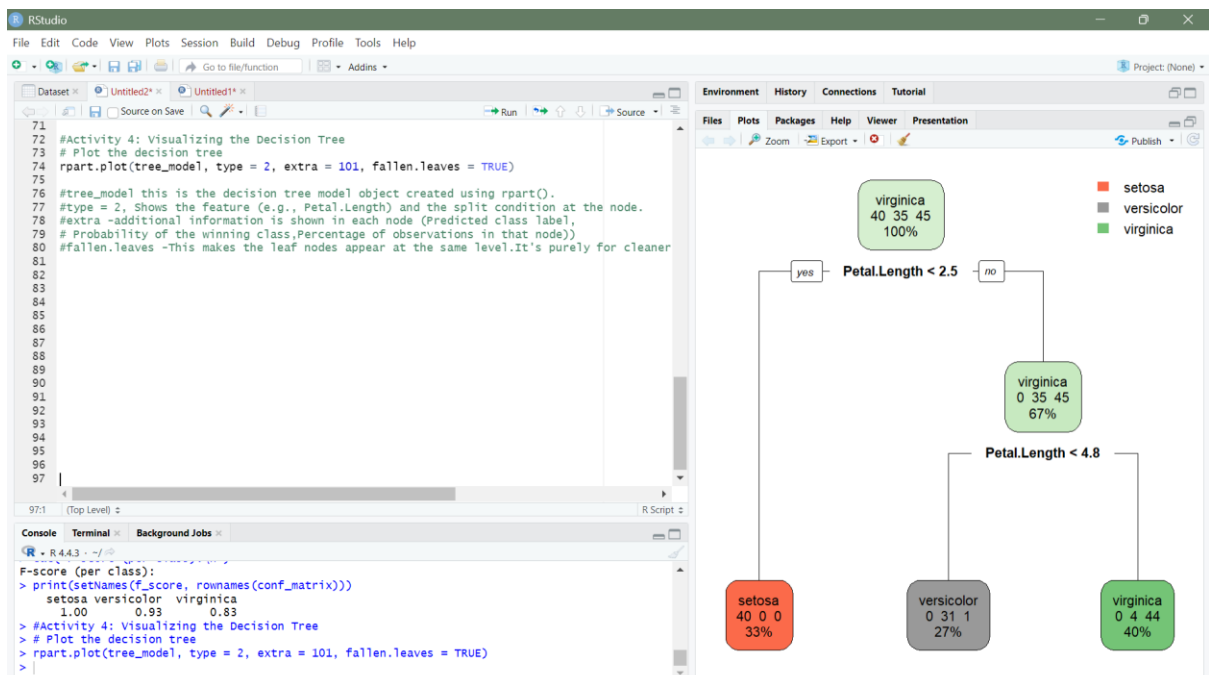
In this step, we check how accurate the model is. We use a confusion matrix to see which predictions were correct or wrong. Then, we calculate accuracy, precision, recall, and F-score to understand the model's strengths and weaknesses for each species.

Activity 4: Visualizing the Decision Tree

Instruction:

Visualize the decision tree model using the `rpart.plot()` function.

Source code:



Justification:

We use `rpart.plot()` to draw the decision tree. This shows us how the model makes decisions. It's a visual way to understand the rules the tree uses to predict flower species based on features like petal length and width.

3. What does the confusion matrix tell you about model performance? Identify any misclassifications.

A confusion matrix shows how many predictions were correct and how many were wrong. It compares the actual class with the predicted class. From the matrix, we can see which species the model predicted correctly and where it made mistakes (misclassifications). If numbers on the diagonal are high and off-diagonal are low, the model is doing well.

4. Using the iris dataset in R, perform a classification task using the Decision Tree algorithm with the following steps:
 1. Split the data into 70% training and 30% testing. Make sure to use `set.seed()` so your result is reproducible.
 2. Build a Decision Tree classifier using the training set to predict the flower species.
 3. Evaluate your model using the testing set and report:
 1. Confusion matrix
 2. Accuracy
 3. Precision, Recall, and F-score for each class
 4. Plot the decision tree and observe its structure.

```
1 # Step 1: Load dataset and packages
2 install.packages("rpart")
3 install.packages("rpart.plot")
4 library(rpart)
5 library(rpart.plot)
6
7 # Step 2: Prepare the data
8 data(iris)
9 set.seed(123) # for reproducibility
10 index <- sample(1:nrow(iris), 0.7 * nrow(iris))
11 train_data <- iris[index, ]
12 test_data <- iris[-index, ]
13
14 # Step 3: Build the decision tree model
15 model <- rpart(Species ~ ., data = train_data, method = "class")
16
17 # Step 4: Make predictions
18 predictions <- predict(model, test_data, type = "class")
19
20 # Step 5: Evaluate the model
21 conf_matrix <- table(Predicted = predictions, Actual = test_data$Species)
22 print(conf_matrix)
23
24 # Accuracy
25 accuracy <- sum(diag(conf_matrix)) / sum(conf_matrix)
26 print(paste("Accuracy:", round(accuracy, 2)))
27
28 # Step 6: Precision, Recall, F-score
29 library(caret)
30 confusionMatrix(predictions, test_data$Species)
31
32 # Step 7: Plot the decision tree
33 rpart.plot(model)
34
```

Environment

Object	Class	Size
iris	data.frame	150 obs. of 5 variables
model	rpart.model	List of 14
test_data	data.frame	45 obs. of 5 variables
train_data	data.frame	105 obs. of 5 variables
tree_model	rpart.model	List of 14

Values

Variable	Value
accuracy	0.977777777777778
conf_matrix	'table' int [1:3, 1:3] 14 0 0 0 18 0 0 1 12
error_rate	0.0666666666666667
f_score	Named num [1:3] 1 0.93 0.83
index	int [1:105] 14 50 118 43 150 148 90 91 143 92 ...
precision	Named num [1:3] 1 0.87 1
predictions	Factor w/ 3 levels "setosa","versicolor",...: 1 1 ...
recall	Named num [1:3] 1 1 0.71
split_index	int [1:120] 14 50 118 43 150 148 90 91 143 92 ...

