American International University

Data Warehousing & Data Mining (final\_report)

Title: "Predictive Modeling of Heart Disease Classification Using Decision Trees”

**Submitted To**

Dr. Akinul Islam Jony

**Project Overview/Discussion:** In this project, we undertook the task of classifying individuals as either having heart disease or not using decision tree models. The project involved several stages, including data loading, preprocessing, modeling, and evaluation. Let's delve into the key aspects and outcomes of the project.

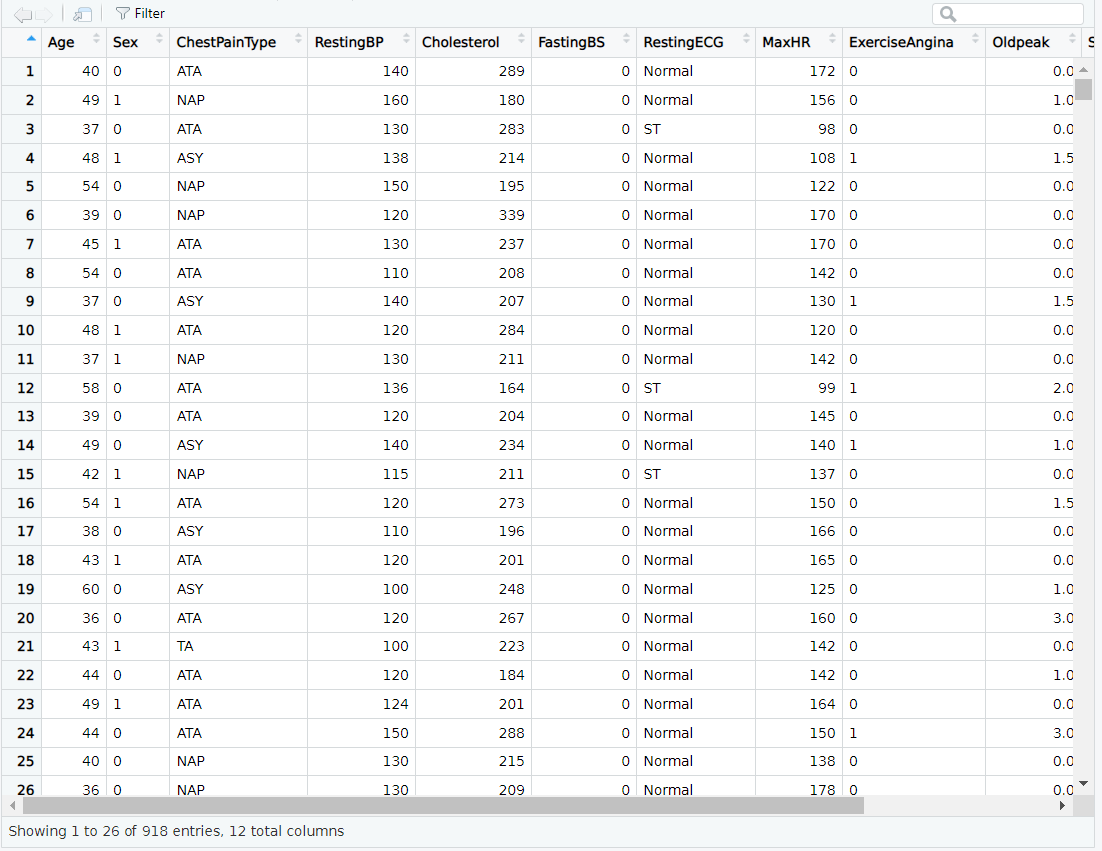
**Dataset Overview:** The dataset we are worked contains information related to various features and attributes of individuals, with the aim of predicting whether a person has heart disease or not. Each row in the dataset represents an individual, and the columns provide information about their health status, medical measurements, and other relevant characteristics.

**Dataset link:** https://www.kaggle.com/datasets/fedesoriano/heart-failure-prediction

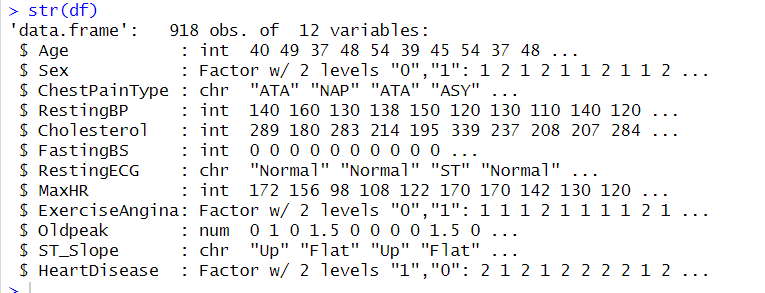
**Attribute information:** Input variables

1. **Age:** The age of the individual.
2. **Sex:** The gender of the individual (M: Male, F: Female).
3. **ChestPainType:** The type of chest pain experienced by the individual (categorical).
4. **RestingBloodPressure:** The resting blood pressure of the individual.
5. **Cholesterol:** The cholesterol level of the individual.
6. **FastingBloodSugar:** Fasting blood sugar levels greater than 120 mg/dl (1: True, 0: False).
7. **RestingECG:** Resting electrocardiographic results (categorical).
8. **MaxHeartRate:** The maximum heart rate achieved by the individual during exercise.
9. **ExerciseAngina:** Exercise-induced angina (chest pain) (N: No, Y: Yes).
10. **ST\_Depression:** ST depression induced by exercise relative to rest.
11. **ST\_Slope:** The slope of the peak exercise ST segment (categorical).
12. **NumMajorVessels:** Number of major vessels colored by fluoroscopy.13.
13. **Thalassemia:** A blood disorder (categorical).
14. **HeartDisease:** The target variable indicating the presence of heart disease (1: Positive, 0: Negative).
15. **Importing or read the Dataset**

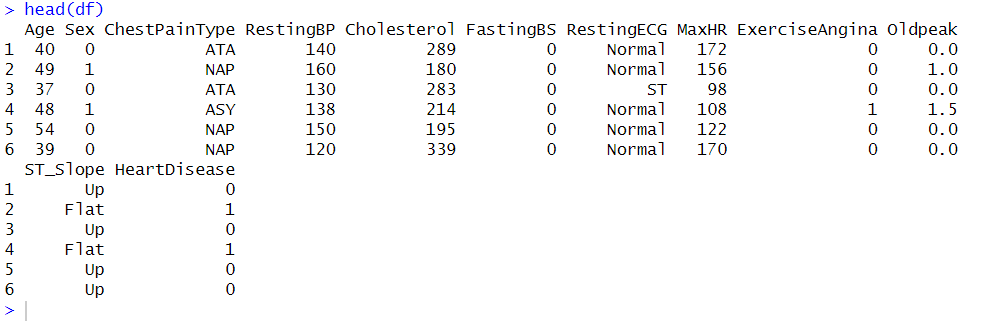
Code: df <-read.csv("C:/Users/Asus/Desktop/Data mining/heart.csv")



1. **Structure of the Dataset & first few rows of Data frame:**

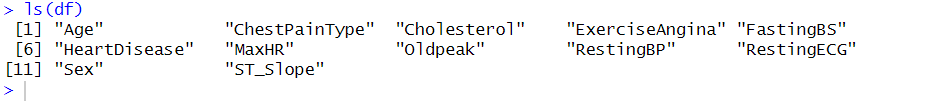
****

Code: head(df)

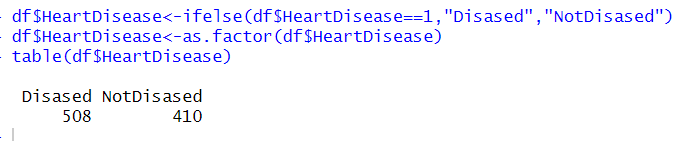


1. **Attributes**

Code:



1. **The code snippet converts the 'HeartDisease' column values from 1 to "Diseased" and other values to "NotDiseased", then converts the column to a categorical factor, and finally displays the frequency distribution of these categories.**

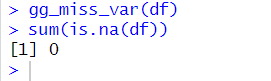


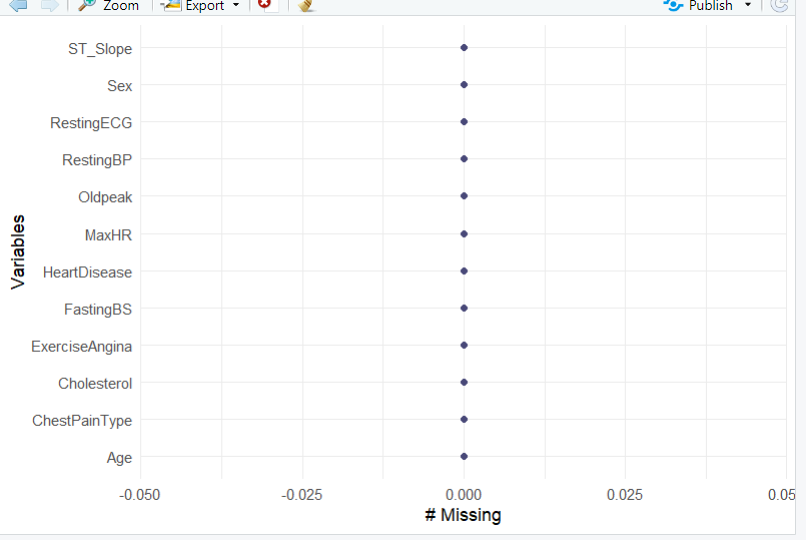




**4.Check for missing values:**

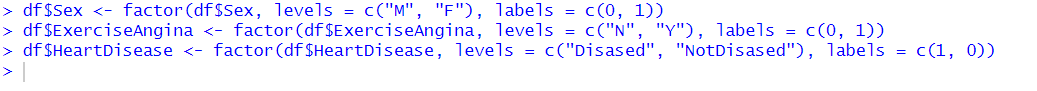
Code:



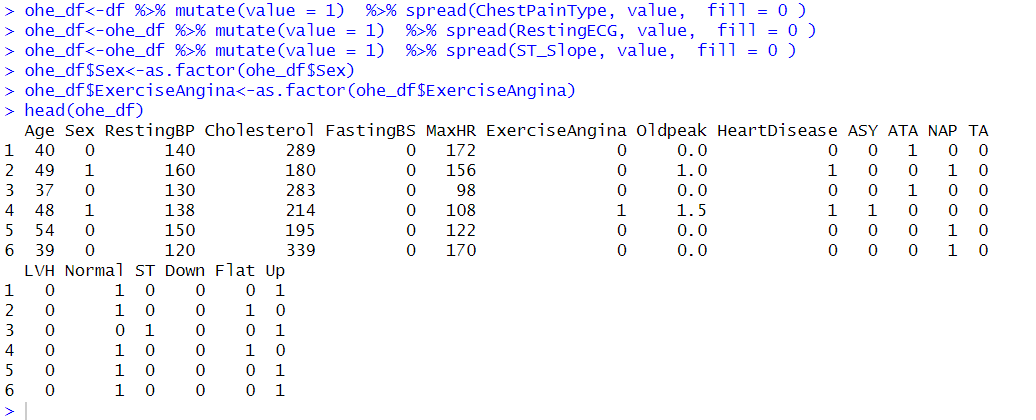


1. **Replacing binary values:**

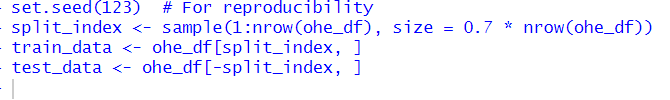
Code:



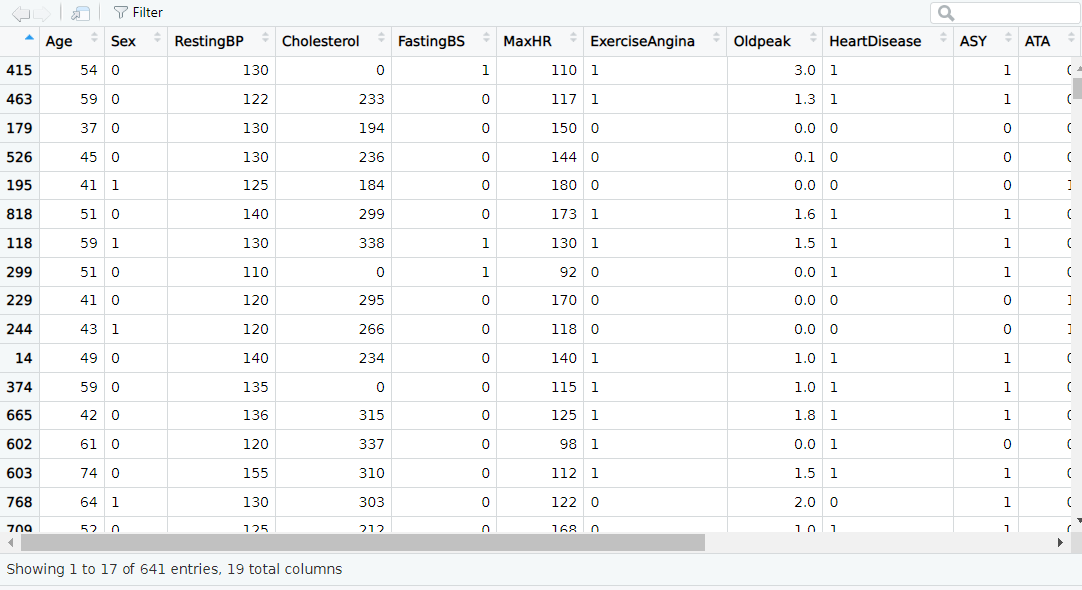
1. **This lines perform one-hot encoding for categorical variables 'ChestPainType', 'RestingECG', and 'ST\_Slope' using the 'mutate' and 'spread' functions.**

****

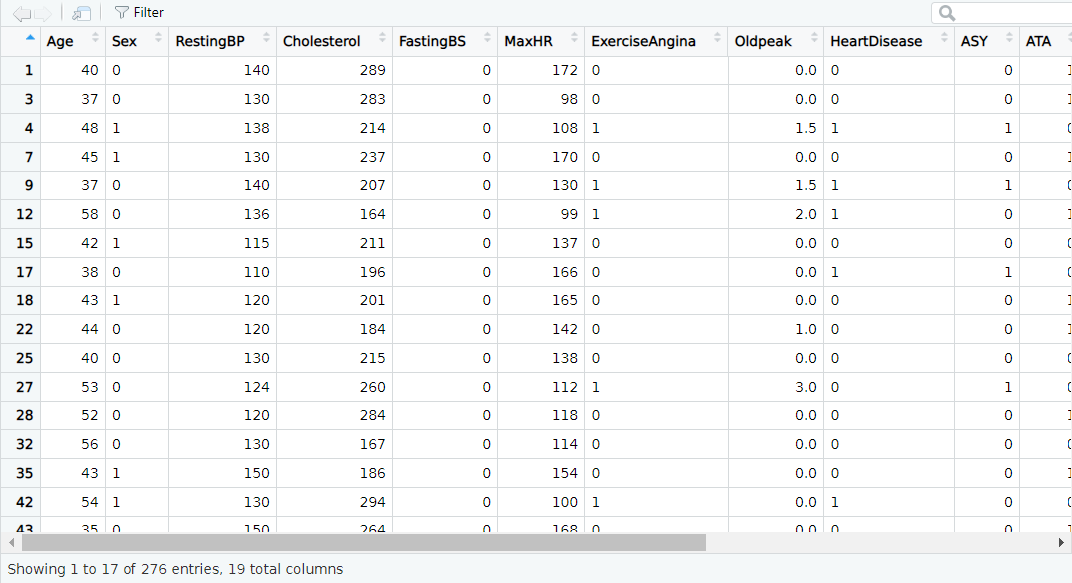
1. **Train-Test Split:**

****

**Train:**

****

**Test:**

****

1. **Fitting Decision Tree Models:**

**# Implementing TDIDT Decision Tree Algorithm**

# Recursive function to build decision tree

build\_tree <- function(data, max\_depth, criterion = "gini") {

if (max\_depth == 0 || length(unique(data$HeartDisease)) == 1) {

leaf\_class <- names(sort(table(data$HeartDisease), decreasing = TRUE))[1]

return(list(node\_type = "leaf", class = leaf\_class))

}

best\_split <- NULL

best\_score <- -Inf

for (feature in colnames(data)[colnames(data) != "HeartDisease"]) {

possible\_splits <- unique(data[[feature]])

for (split in possible\_splits) {

left\_subset <- data[data[[feature]] == split, ]

right\_subset <- data[data[[feature]] != split, ]

if (nrow(left\_subset) > 0 && nrow(right\_subset) > 0) {

if (criterion == "gini") {

score <- calculate\_gini\_gain(data$HeartDisease, left\_subset$HeartDisease, right\_subset$HeartDisease)

} else if (criterion == "gini\_ratio") {

score <- calculate\_gini\_ratio(data$HeartDisease, left\_subset$HeartDisease, right\_subset$HeartDisease)

} else {

stop("Invalid criterion specified")

}

if (score > best\_score) {

best\_score <- score

best\_split <- list(feature = feature, split = split,

left\_subset = left\_subset, right\_subset = right\_subset)

}

}

}

}

if (is.null(best\_split)) {

leaf\_class <- names(sort(table(data$HeartDisease), decreasing = TRUE))[1]

return(list(node\_type = "leaf", class = leaf\_class))

}

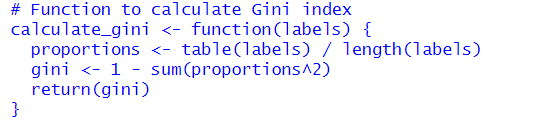
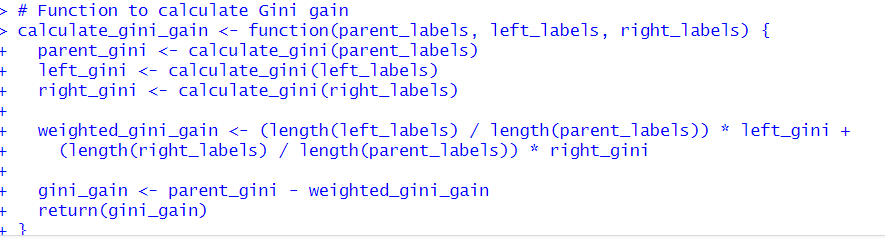
left\_node <- build\_tree(best\_split$left\_subset, max\_depth - 1, criterion)

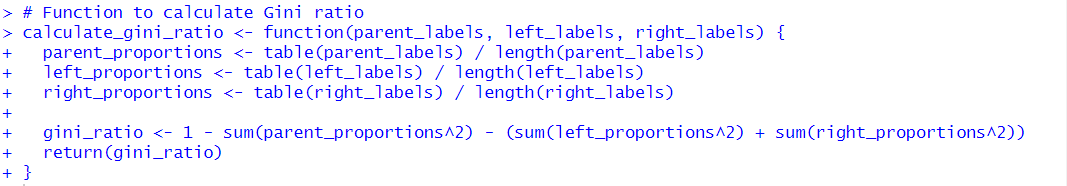
right\_node <- build\_tree(best\_split$right\_subset, max\_depth - 1, criterion)

return(list(node\_type = "internal", feature = best\_split$feature, split = best\_split$split,

left\_node = left\_node, right\_node = right\_node))

}

****

****

1. **Predictions and Accuracy Calculation:** Predictions are made on the test data using the fitted models, and accuracy is calculated for each model.

# Function to calculate the best split score

calculate\_split\_score <- function(left\_labels, right\_labels, parent\_labels, criterion = "gini") {

if (criterion == "gini") {

score <- calculate\_gini\_gain(parent\_labels, left\_labels, right\_labels)

} else if (criterion == "gini\_ratio") {

score <- calculate\_gini\_ratio(parent\_labels, left\_labels, right\_labels) # Pass parent\_labels here

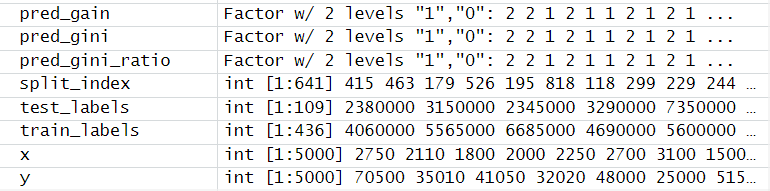
} else {

stop("Invalid criterion specified")

}

return(score)

}

****

# Calculate accuracy for each model

accuracy\_gain <- sum(pred\_gain == test\_data$HeartDisease) / length(test\_data$HeartDisease)

accuracy\_gini <- sum(pred\_gini == test\_data$HeartDisease) / length(test\_data$HeartDisease)

accuracy\_gini\_ratio <- sum(pred\_gini\_ratio == test\_data$HeartDisease) /

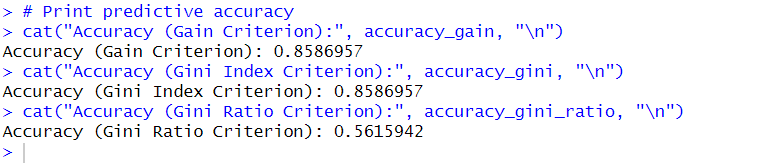
length(test\_data$HeartDisease)

# Print predictive accuracy

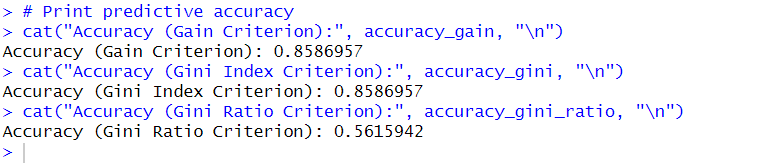
cat("Accuracy (Gain):", accuracy\_gain, "\n")

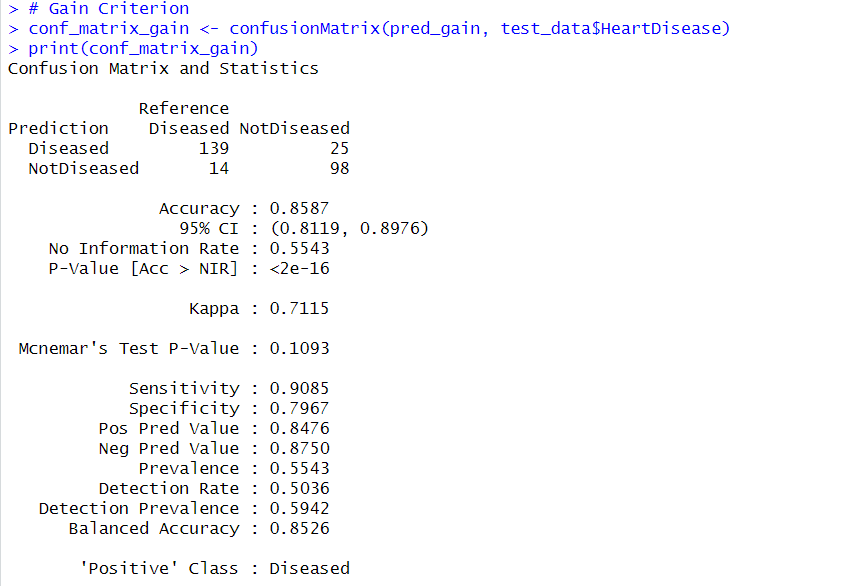
cat("Accuracy (Gini Index):", accuracy\_gini, "\n")

cat("Accuracy (Gini Ratio):", accuracy\_gini\_ratio, "\n")



1. **Confusion Matrix and Visualization:**

****

****

# Gain Criterion

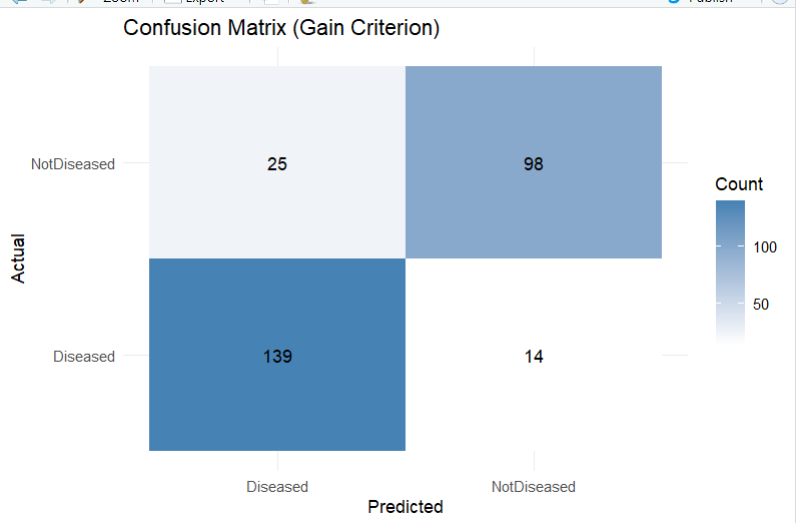
conf\_matrix\_gain <- confusionMatrix(pred\_gain, test\_data$HeartDisease)

print(conf\_matrix\_gain)

# Create a data frame for visualization

conf\_matrix\_data <- as.data.frame(as.table(conf\_matrix\_gain$table))

colnames(conf\_matrix\_data) <- c("Predicted", "Actual", "Count")

****

# Gini Index Criterion

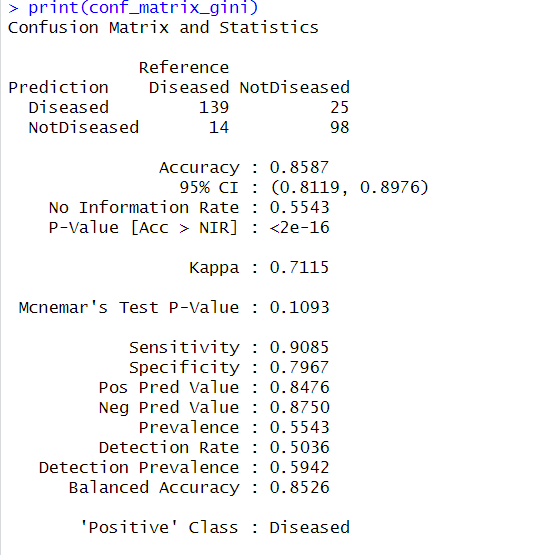
conf\_matrix\_gini <- confusionMatrix(pred\_gini, test\_data$HeartDisease)

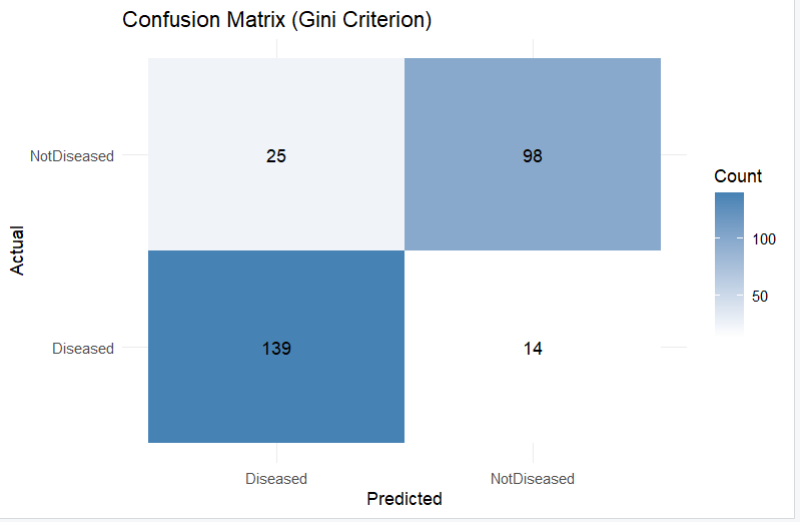
print(conf\_matrix\_gini)

# Create a data frame for visualization

conf\_matrix\_data <- as.data.frame(as.table(conf\_matrix\_gini$table))

colnames(conf\_matrix\_data) <- c("Predicted", "Actual", "Count")



****

# Gini Ratio Criterion

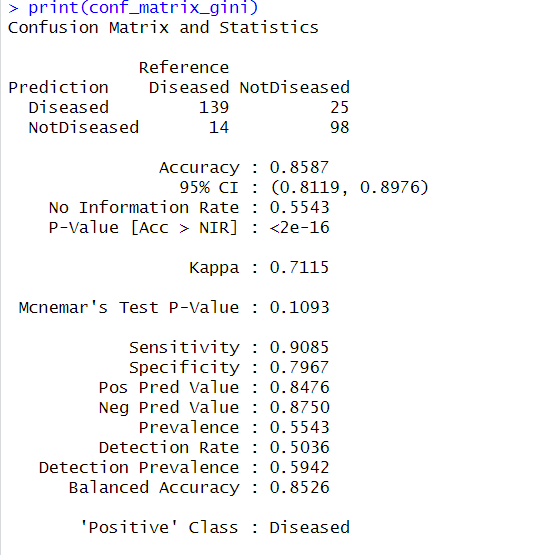
conf\_matrix\_gini\_ratio <- confusionMatrix(pred\_gini\_ratio, test\_data$HeartDisease)

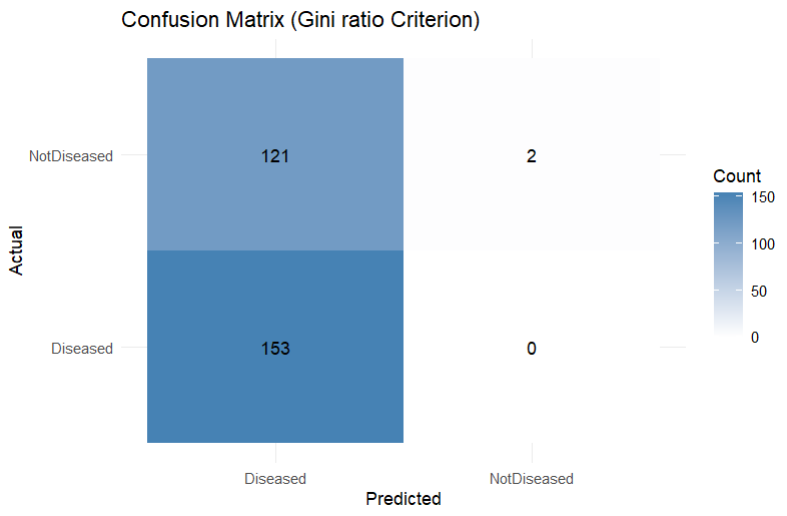
print(conf\_matrix\_gini\_ratio)

# Create a data frame for visualization

conf\_matrix\_data <- as.data.frame(as.table(conf\_matrix\_gini\_ratio$table))

colnames(conf\_matrix\_data) <- c("Predicted", "Actual", "Count")



****