Project Report

Heart Disease : Diagnosis with Support Vector Machines

Author:

Syed Faizan

October 14th, 2024

Contents

Introduction

In this project, we aim to apply **Support Vector Machines (SVM)** to the *Heart Disease* dataset in order to mine and classify the data effectively. *Support Vector Machines* are powerful classification algorithms that find the optimal hyperplane to separate classes within the dataset, which in this case is identifying the presence or absence of heart disease. Through this process, we will walk through the steps of data preparation, model training, tuning, and evaluation, while documenting each phase of the analysis.

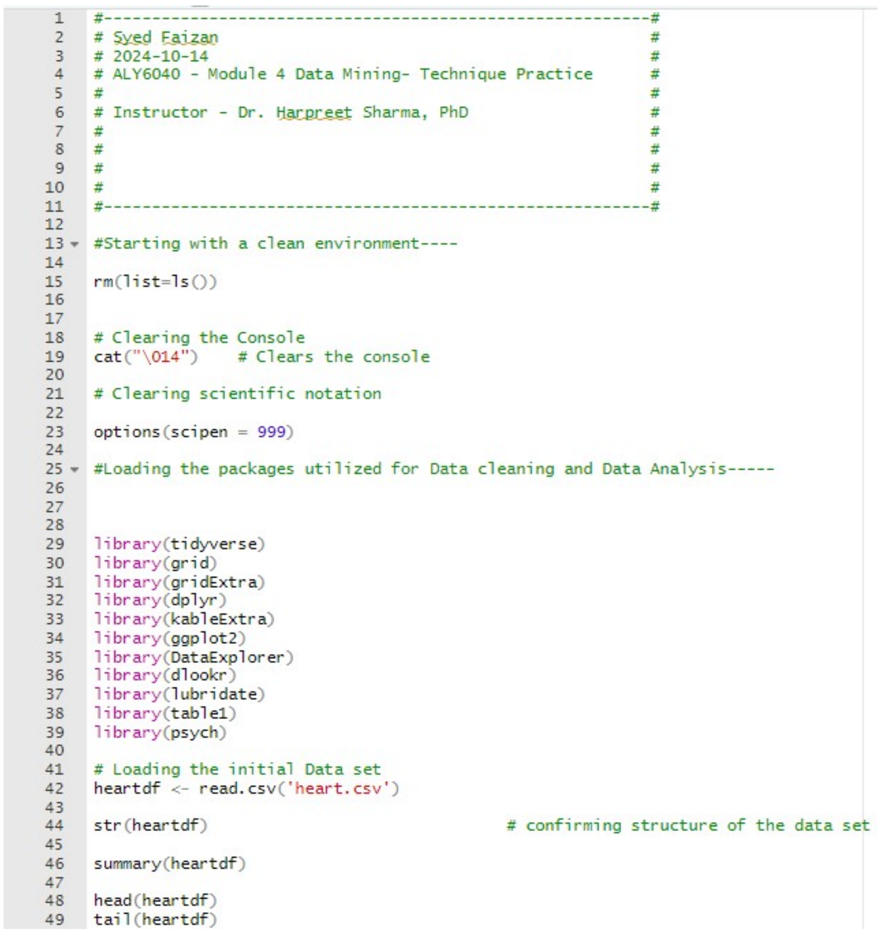
The report will follow a structured approach starting with a **code walk-through**, where the interactions between the code and the *Heart Disease* data will be detailed. Each transformation or modification of the data will be explained, with examples provided to support the explanations. This ensures a comprehensive understanding of how the data is processed through each stage of the analysis, including scaling of variables and splitting the dataset into training and testing subsets.

Next, the **analysis** section will focus on interpreting the output generated by the SVM model applied to the *Heart Disease* dataset. The relationships between the variables—such as age, cholesterol levels, blood pressure, and their impact on heart disease—will be analyzed. Visualizations such as *ROC curves*, *confusion matrices*, and other relevant plots will be utilized to support the conclusions drawn from the model's performance. Special attention will be given to tuning the model's hyperparameters (such as the *cost* and *gamma*) to optimize its classification accuracy.

Finally, the **interpretation and recommendations** section will provide actionable insights based on the findings from the SVM analysis. This section will interpret the results in the context of predicting heart disease and provide recommendations for further analysis or actions that can be taken by healthcare professionals or data scientists. We will suggest incorporating additional variables that could enhance future models and provide a more comprehensive view of heart disease risk factors. For instance, including external variables such as *lifestyle factors* or *dietary habits* may further improve the model's predictive power.

Overall, this project is designed to showcase the practical application of SVM in a healthcare data mining scenario, guiding the reader through the complete process from data preparation to actionable recommendations.

Code walk through



```
#----------------------------------------------------------#
# Syed Faizan                                              #
# 2024-10-14                                               #
# ALY6040 - Module 4 Data Mining- Technique Practice       #
#                                                          #
# Instructor - Dr. Harpreet Sharma, PhD                    #
#                                                          #
#                                                          #
#                                                          #
#                                                          #
#----------------------------------------------------------#

#Starting with a clean environment----

rm(list=ls())


# Clearing the Console
cat("\014")    # Clears the console

# Clearing scientific notation

options(scipen = 999)

#Loading the packages utilized for Data cleaning and Data Analysis-----



library(tidyverse)
library(grid)
library(gridExtra)
library(dplyr)
library(kableExtra)
library(ggplot2)
library(DataExplorer)
library(dlookr)
library(lubridate)
library(table1)
library(psych)

# Loading the initial Data set
heartdf <- read.csv('heart.csv')

str(heartdf)                              # confirming structure of the data set

summary(heartdf)

head(heartdf)
tail(heartdf)
```

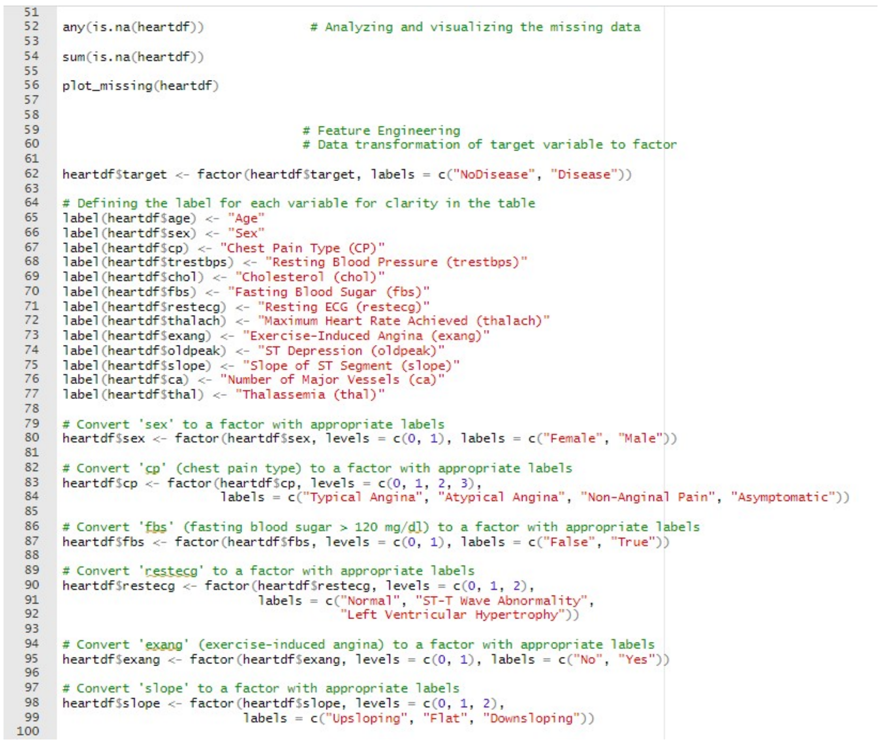
Figure 1: Loading packages and the dataset

This code begins by ensuring a clean environment with `rm(list=ls())`, which removes all objects from the workspace. The console is cleared using `cat("14")` and scientific notation is suppressed via `options(scipen = 999)`. These steps prepare the environment for data analysis.

The script then proceeds to load several libraries essential for data cleaning and analysis. The ***tidyverse*** package is a collection of R packages designed for data manipulation and visualization, while ***grid*** and ***gridExtra*** help with grid-based layouts in graphics. ***dplyr*** is part of *tidyverse* and is used for data manipulation, and ***kableExtra*** helps in creating HTML or LaTeX tables.

ggplot2 is also part of *tidyverse* and is a key package for data visualization. ***DataExplorer*** simplifies exploratory data analysis, and ***lookr*** is used for interactive data exploration. ***lubridate*** makes it easier to work with date-time objects, and ***table1*** allows formatted summaries for data reporting. ***psych*** provides tools for psychological and social sciences, focusing on factor analysis, data description, and psychometric tests.

The dataset is loaded with `read.csv('heart.csv')` and assigned to `heartdf`. Initial exploratory steps include checking the structure using `str(heartdf)`, providing a summary of the dataset via `summary(heartdf)`, and viewing the first and last few rows with `head()` and `tail()`.



```r

any(is.na(heartdf))                  # Analyzing and visualizing the missing data

sum(is.na(heartdf))

plot_missing(heartdf)


                                    # Feature Engineering
                                    # Data transformation of target variable to factor

heartdf$target <- factor(heartdf$target, labels = c("NoDisease", "Disease"))

# Defining the label for each variable for clarity in the table
label(heartdf$age) <- "Age"
label(heartdf$sex) <- "Sex"
label(heartdf$cp) <- "Chest Pain Type (CP)"
label(heartdf$trestbps) <- "Resting Blood Pressure (trestbps)"
label(heartdf$chol) <- "Cholesterol (chol)"
label(heartdf$fbs) <- "Fasting Blood Sugar (fbs)"
label(heartdf$restecg) <- "Resting ECG (restecg)"
label(heartdf$thalach) <- "Maximum Heart Rate Achieved (thalach)"
label(heartdf$exang) <- "Exercise-Induced Angina (exang)"
label(heartdf$oldpeak) <- "ST Depression (oldpeak)"
label(heartdf$slope) <- "Slope of ST Segment (slope)"
label(heartdf$ca) <- "Number of Major Vessels (ca)"
label(heartdf$thal) <- "Thalassemia (thal)"

# Convert 'sex' to a factor with appropriate labels
heartdf$sex <- factor(heartdf$sex, levels = c(0, 1), labels = c("Female", "Male"))

# Convert 'cp' (chest pain type) to a factor with appropriate labels
heartdf$cp <- factor(heartdf$cp, levels = c(0, 1, 2, 3),
                     labels = c("Typical Angina", "Atypical Angina", "Non-Anginal Pain", "Asymptomatic"))

# Convert 'fbs' (fasting blood sugar > 120 mg/dl) to a factor with appropriate labels
heartdf$fbs <- factor(heartdf$fbs, levels = c(0, 1), labels = c("False", "True"))

# Convert 'restecg' to a factor with appropriate labels
heartdf$restecg <- factor(heartdf$restecg, levels = c(0, 1, 2),
                          labels = c("Normal", "ST-T Wave Abnormality",
                                     "Left Ventricular Hypertrophy"))

# Convert 'exang' (exercise-induced angina) to a factor with appropriate labels
heartdf$exang <- factor(heartdf$exang, levels = c(0, 1), labels = c("No", "Yes"))

# Convert 'slope' to a factor with appropriate labels
heartdf$slope <- factor(heartdf$slope, levels = c(0, 1, 2),
                        labels = c("Upsloping", "Flat", "Downsloping"))

```

Figure 2: Data examination and transformation

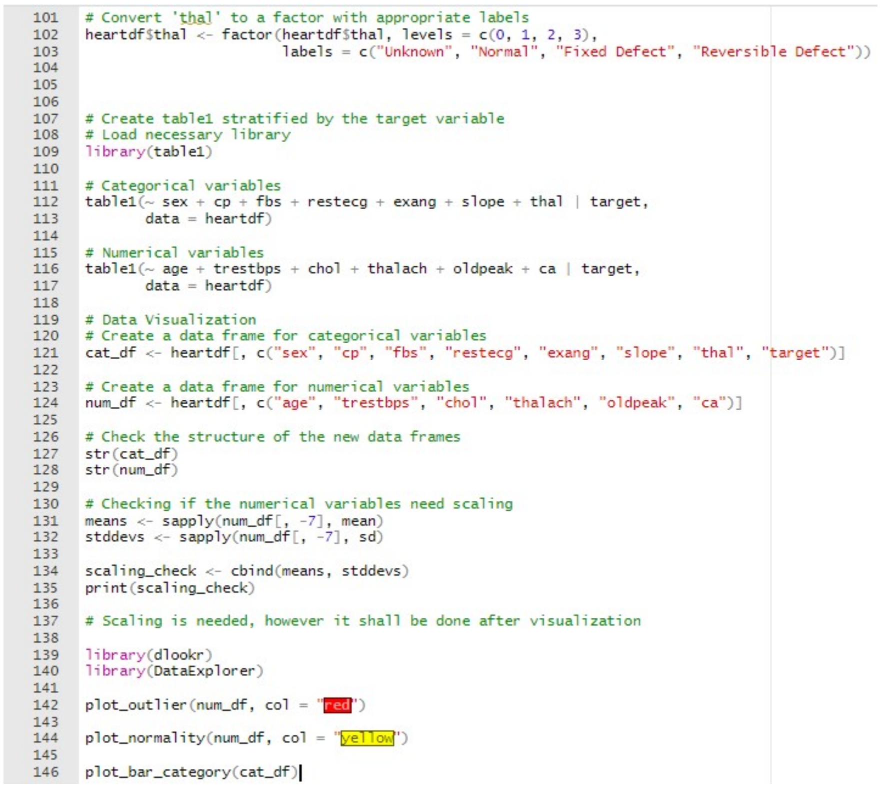
The above code starts by analyzing and visualizing missing data in `heartdf`. The function `any(is.na(heartdf))` checks if there are any missing values, while `sum(is.na(heartdf))` counts them. The `plot_missing()` function visually displays missing data.

The code then proceeds to **feature engineering**, where the `target` variable is transformed into a factor with two levels: *"NoDisease"* and *"Disease"*. This is crucial for classification tasks, ensuring the target variable is treated as categorical.

Next, variable labels are defined using `label()` for clear and interpretable output in tables. For instance, `label(heartdf$age)` is assigned *"Age"*, and similarly, other variables such as *"Resting Blood Pressure (trestbps)"* and *"Exercise-Induced Angina (exang)"* are labeled for clarity in reporting.

The script also converts several variables into factors with meaningful levels. For example, `heartdf$sex` is converted into a factor with levels *"Female"* and *"Male"* using the `factor()` function. Likewise, categorical variables such as *chest pain type (cp)*, *fasting blood sugar (fbs)*, *resting ECG (restecg)*, and others are transformed into factors with appropriate levels and labels for better interpretability.

These transformations ensure that the data is clean, properly labeled, and ready for further analysis and modeling, particularly for the application of classification algorithms like *Support Vector Machines (SVM)*.



```r
# Convert 'thal' to a factor with appropriate labels
heartdf$thal <- factor(heartdf$thal, levels = c(0, 1, 2, 3),
                         labels = c("Unknown", "Normal", "Fixed Defect", "Reversible Defect"))



# Create table1 stratified by the target variable
# Load necessary library
library(table1)

# Categorical variables
table1(~ sex + cp + fbs + restecg + exang + slope + thal | target,
       data = heartdf)

# Numerical variables
table1(~ age + trestbps + chol + thalach + oldpeak + ca | target,
       data = heartdf)

# Data Visualization
# Create a data frame for categorical variables
cat_df <- heartdf[, c("sex", "cp", "fbs", "restecg", "exang", "slope", "thal", "target")]

# Create a data frame for numerical variables
num_df <- heartdf[, c("age", "trestbps", "chol", "thalach", "oldpeak", "ca")]

# Check the structure of the new data frames
str(cat_df)
str(num_df)

# Checking if the numerical variables need scaling
means <- sapply(num_df[, -7], mean)
stddevs <- sapply(num_df[, -7], sd)

scaling_check <- cbind(means, stddevs)
print(scaling_check)

# Scaling is needed, however it shall be done after visualization

library(dlookr)
library(DataExplorer)

plot_outlier(num_df, col = "red")

plot_normality(num_df, col = "yellow")

plot_bar_category(cat_df)
```

Figure 3: Exploratory Data Analysis and Visualization

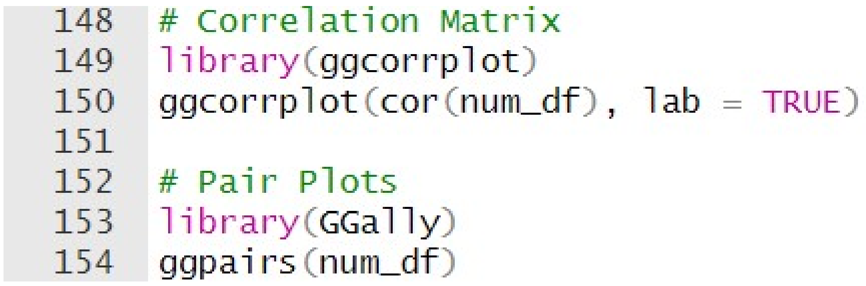
The code begins by converting the variable `thal` into a factor with levels *"Unknown"*, *"Normal"*, *"Fixed Defect"*, and *"Reversible Defect"* for clarity. This is done using the `factor()` function to improve interpretability in analyses.

Next, a table is generated using the **table1** package, stratified by the `target` variable. Categorical and numerical variables are separately summarized using `table1()`, allowing a clearer breakdown of data characteristics by target class.

The script then creates two separate data frames: `cat_df` for categorical variables and `num_df` for numerical variables. These are essential for handling the data differently depending on its type, with `str()` used to check the structure of these new data frames.

To assess if numerical variables require scaling, the code calculates their means and standard deviations using `sapply()` with `mean` and `sd` functions, combining them into a matrix for quick reference.

Scaling is deemed necessary but deferred until after visualization. For visualizing outliers, the `plot_outlier()` function from the **DataExplorer** package is used, with *red* as the color for the boxplot. Normality is assessed using `plot_normality()`, with *yellow* as the color of the plot. Additionally, `plot_bar_category()` visualizes the distribution of categorical variables, providing insight into their frequency distributions.

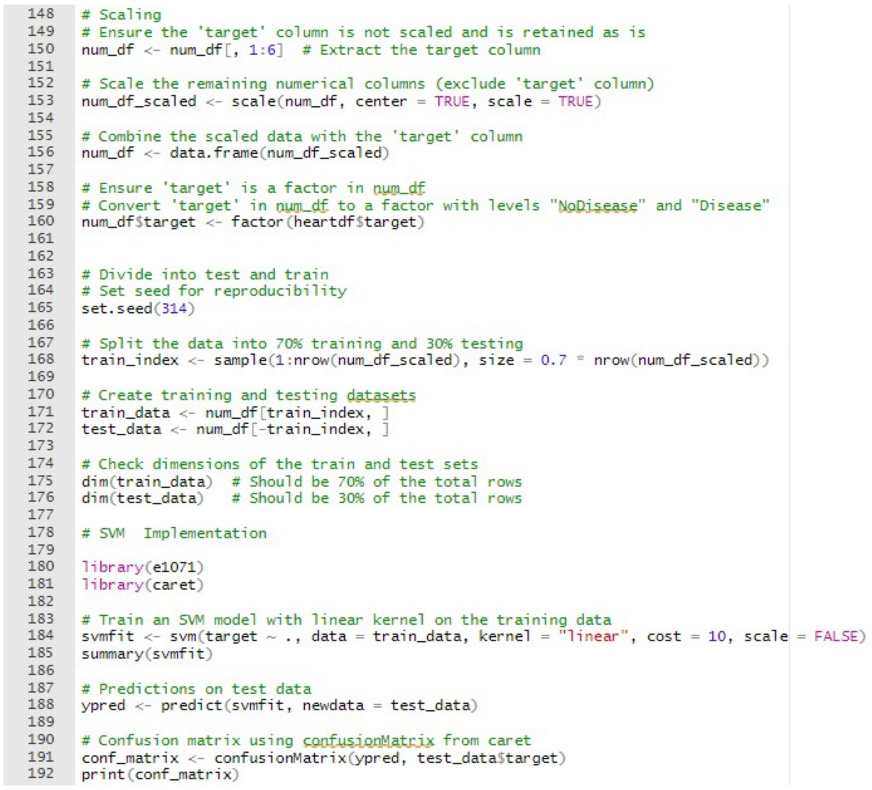


```r
# Correlation Matrix
library(ggcorrplot)
ggcorrplot(cor(num_df), lab = TRUE)

# Pair Plots
library(GGally)
ggpairs(num_df)
```

Figure 4: Correlation and Scatter pair plotting

This above code block begins by calculating and visualizing the correlation matrix for the numerical dataset `num_df`. The **ggcorrplot** package is used to display the correlation matrix, with the correlations computed via `cor()` and labeled on the plot using the `lab = TRUE` argument. This provides a clear view of the linear relationships between numerical variables.

Following this, the **GGally** package's `ggpairs()` function is utilized to create pair plots for `num_df`. Pair plots provide a matrix of scatter plots for each pair of variables, as well as histograms or density plots for individual variables along the diagonal. This is a powerful tool for exploratory data analysis, as it helps in visualizing potential relationships and patterns among variables in a multivariate dataset.



```r
# Scaling
# Ensure the 'target' column is not scaled and is retained as is
num_df <- num_df[, 1:6]  # Extract the target column

# Scale the remaining numerical columns (exclude 'target' column)
num_df_scaled <- scale(num_df, center = TRUE, scale = TRUE)

# Combine the scaled data with the 'target' column
num_df <- data.frame(num_df_scaled)

# Ensure 'target' is a factor in num_df
# Convert 'target' in num_df to a factor with levels "NoDisease" and "Disease"
num_df$target <- factor(heartdf$target)


# Divide into test and train
# Set seed for reproducibility
set.seed(314)

# Split the data into 70% training and 30% testing
train_index <- sample(1:nrow(num_df_scaled), size = 0.7 * nrow(num_df_scaled))

# Create training and testing datasets
train_data <- num_df[train_index, ]
test_data <- num_df[-train_index, ]

# Check dimensions of the train and test sets
dim(train_data)  # Should be 70% of the total rows
dim(test_data)   # Should be 30% of the total rows

# SVM  Implementation

library(e1071)
library(caret)

# Train an SVM model with linear kernel on the training data
svmfit <- svm(target ~ ., data = train_data, kernel = "linear", cost = 10, scale = FALSE)
summary(svmfit)

# Predictions on test data
ypred <- predict(svmfit, newdata = test_data)

# Confusion matrix using confusionMatrix from caret
conf_matrix <- confusionMatrix(ypred, test_data$target)
print(conf_matrix)
```

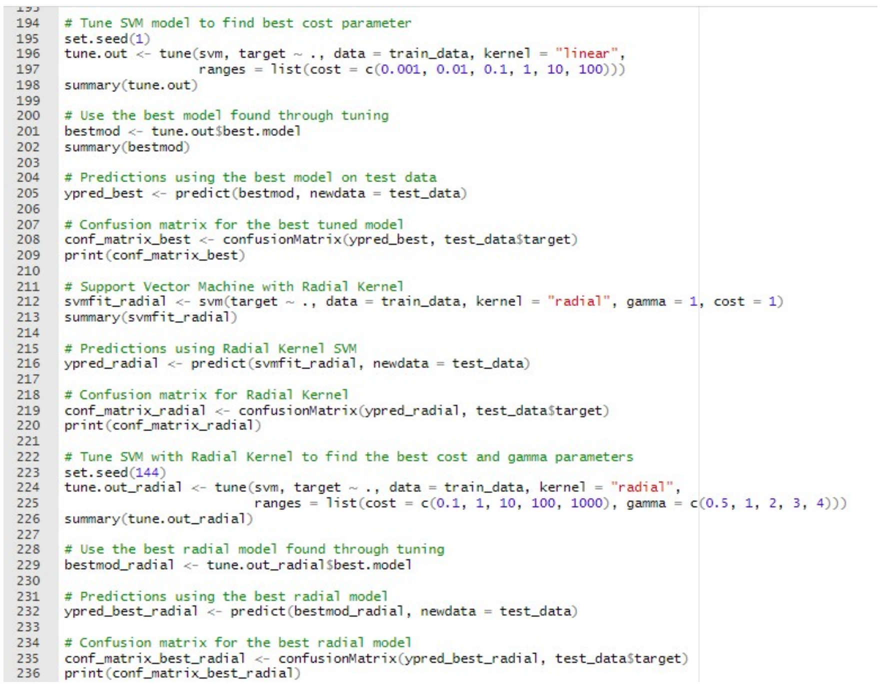
Figure 5: Scaling and Data Splicing

The above code chunk begins by scaling the numerical columns in `num_df`, excluding the `target` column, using the `scale()` function with centering and scaling enabled (`center = TRUE`, `scale = TRUE`). The scaled data is then recombined with the `target` variable to form `num_df_scaled`. The `target` variable is explicitly converted to a factor with levels *"NoDisease"* and *"Disease"* to ensure it is treated correctly in the classification task.

Next, the dataset is split into 70% training and 30% testing sets using `sample()` and indices generated from `nrow()`. The `train_data` and `test_data` subsets are created accordingly, with `dim()` used to check their dimensions to ensure they follow the 70-30 split.

The Support Vector Machine (SVM) model is implemented using the **e1071** package. The SVM is trained with a *linear kernel* on the `train_data` using `svm()` with a cost parameter of 10. After training, predictions are made on `test_data` using `predict()`.

Finally, the **caret** package's `confusionMatrix()` function is used to evaluate the predictions against the actual target values from `test_data`. This provides performance metrics such as accuracy, precision, and recall, which are printed to summarize the model's effectiveness.



```r
# Tune SVM model to find best cost parameter
set.seed(1)
tune.out <- tune(svm, target ~ ., data = train_data, kernel = "linear",
                 ranges = list(cost = c(0.001, 0.01, 0.1, 1, 10, 100)))
summary(tune.out)

# Use the best model found through tuning
bestmod <- tune.out$best.model
summary(bestmod)

# Predictions using the best model on test data
ypred_best <- predict(bestmod, newdata = test_data)

# Confusion matrix for the best tuned model
conf_matrix_best <- confusionMatrix(ypred_best, test_data$target)
print(conf_matrix_best)

# Support Vector Machine with Radial Kernel
svmfit_radial <- svm(target ~ ., data = train_data, kernel = "radial", gamma = 1, cost = 1)
summary(svmfit_radial)

# Predictions using Radial Kernel SVM
ypred_radial <- predict(svmfit_radial, newdata = test_data)

# Confusion matrix for Radial Kernel
conf_matrix_radial <- confusionMatrix(ypred_radial, test_data$target)
print(conf_matrix_radial)

# Tune SVM with Radial Kernel to find the best cost and gamma parameters
set.seed(144)
tune.out_radial <- tune(svm, target ~ ., data = train_data, kernel = "radial",
                        ranges = list(cost = c(0.1, 1, 10, 100, 1000), gamma = c(0.5, 1, 2, 3, 4)))
summary(tune.out_radial)

# Use the best radial model found through tuning
bestmod_radial <- tune.out_radial$best.model

# Predictions using the best radial model
ypred_best_radial <- predict(bestmod_radial, newdata = test_data)

# Confusion matrix for the best radial model
conf_matrix_best_radial <- confusionMatrix(ypred_best_radial, test_data$target)
print(conf_matrix_best_radial)
```

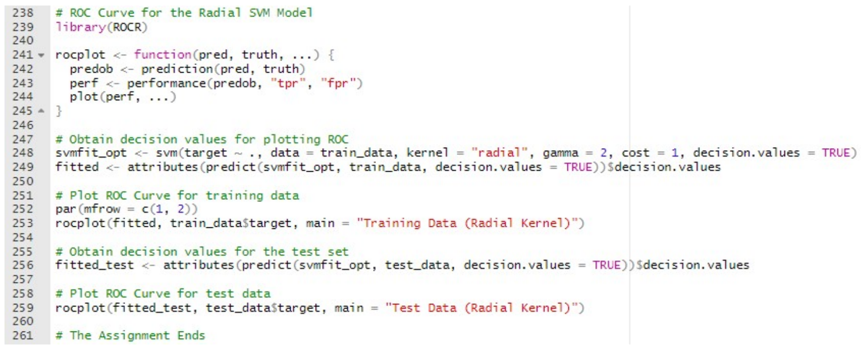
Figure 6: Support Vector Machines

The above code starts by tuning the SVM model with a *linear kernel* to find the best `cost` parameter using `tune()` from the **e1071** package. The `tune()` function searches over a range of cost values and selects the optimal one. The best model is then used to make predictions on the test data, followed by evaluating its performance with a confusion matrix using the **caret** package's `confusionMatrix()` function.

Next, the code applies the *Radial Kernel* SVM using `svm()` with default parameters (`gamma = 1`, `cost = 1`). Predictions are made on the test data using the trained model, and its performance is also evaluated via a confusion matrix.

The SVM model with a *radial kernel* is then fine-tuned using the `tune()` function. This tuning process optimizes both the `cost` and `gamma` parameters over a grid of values. The best radial model is extracted and used to predict the test set, and its performance is similarly evaluated through the confusion matrix.

Each step of this process involves identifying the best hyperparameters for both linear and radial kernel SVMs and assessing their predictive performance on the test data using confusion matrices, providing insight into the model's accuracy, precision, and recall.



```
# ROC Curve for the Radial SVM Model
library(ROCR)

rocplot <- function(pred, truth, ...) {
  predob <- prediction(pred, truth)
  perf <- performance(predob, "tpr", "fpr")
  plot(perf, ...)
}

# Obtain decision values for plotting ROC
svmfit_opt <- svm(target ~ ., data = train_data, kernel = "radial", gamma = 2, cost = 1, decision.values = TRUE)
fitted <- attributes(predict(svmfit_opt, train_data, decision.values = TRUE))$decision.values

# Plot ROC Curve for training data
par(mfrow = c(1, 2))
rocplot(fitted, train_data$target, main = "Training Data (Radial Kernel)")

# Obtain decision values for the test set
fitted_test <- attributes(predict(svmfit_opt, test_data, decision.values = TRUE))$decision.values

# Plot ROC Curve for test data
rocplot(fitted_test, test_data$target, main = "Test Data (Radial Kernel)")

# The Assignment Ends
```

Figure 7: ROC Plot of Radial Kernel SVM

The code begins by loading the **ROCR** package, which provides functions for generating ROC curves to evaluate model performance. The `rocplot()` function is defined to plot ROC curves by computing the *true positive rate (tpr)* and *false positive rate (fpr)* from the model's predictions using the `performance()` and `prediction()` functions from **ROCR**.

Next, a Support Vector Machine (SVM) model with a *radial kernel* is trained using the `svm()` function with parameters `gamma = 2` and `cost = 1`, and decision values are extracted using `predict()` with `decision.values = TRUE`. The decision values are needed to plot the ROC curve, which measures the model's performance in distinguishing between classes.

The ROC curve for the training data is plotted first, with the plotting area divided into two subplots using `par(mfrow = c(1, 2))`. The `rocplot()` function is used to visualize the ROC curve, labeled as *Training Data (Radial Kernel)*.

The same process is repeated for the test data, where decision values are extracted and used to plot the ROC curve for the test set, labeled as *Test Data (Radial Kernel)*. This allows for a comparative analysis of the model's performance on both the training and testing sets.

The project concludes with the completion of the ROC plots.

Data Analysis

The **Heart Disease dataset** is a widely used dataset in healthcare data analysis and machine learning, primarily for predicting the presence of heart disease in patients based on a variety of clinical and demographic factors. The dataset consists of n = *1025 observations* and contains p = *14 variables*, which include both categorical and numerical features.

Key variables include:

- **Age**: The age of the patient in years.
- **Sex**: The gender of the patient, coded as *1* for male and *0* for female.
- **Chest Pain Type (cp)**: A categorical variable representing the type of chest pain, with values ranging from *0* (Typical Angina) to *3* (Asymptomatic).
- **Resting Blood Pressure (trestbps)**: The patient's resting blood pressure in mm Hg.
- **Serum Cholesterol (chol)**: The serum cholesterol level in mg/dl.
- **Fasting Blood Sugar (fbs)**: Coded as *1* if fasting blood sugar is greater than 120 mg/dl, and *0* otherwise.
- **Resting Electrocardiographic Results (restecg)**: A categorical variable that reflects the results of the resting ECG, with values from *0* to *2*.
- **Maximum Heart Rate Achieved (thalach)**: The maximum heart rate achieved during exercise.
- **Exercise Induced Angina (exang)**: Coded as *1* if exercise-induced angina is present, and *0* if absent.
- **ST Depression (oldpeak)**: ST depression induced by exercise relative to rest.
- **Slope of ST Segment (slope)**: The slope of the peak exercise ST segment.
- **Number of Major Vessels (ca)**: The number of major vessels colored by fluoroscopy, ranging from *0* to *3*.

- **Thalassemia (thal)**: A categorical variable representing the type of thalassemia (Normal, Fixed Defect, Reversible Defect).

- **Target**: The target variable, which indicates the presence of heart disease, coded as *1* for disease and *0* for no disease.

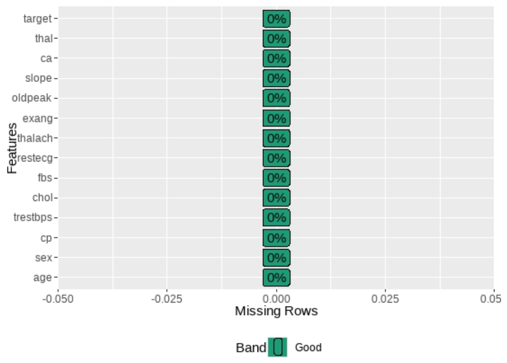
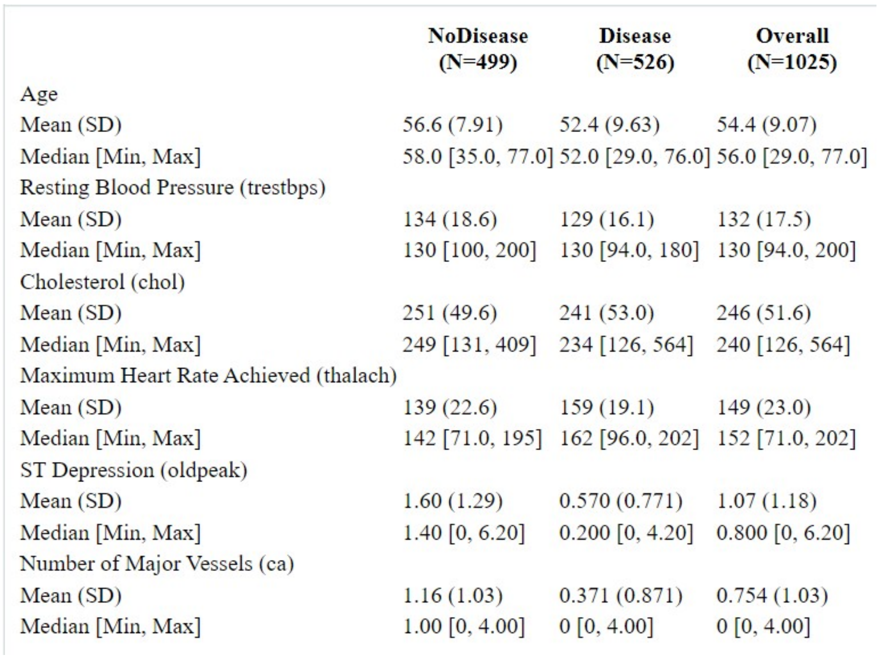


Figure 8: Missing values Plot

The missing value plot shows no missing values, indicating a clean and rich dataset.



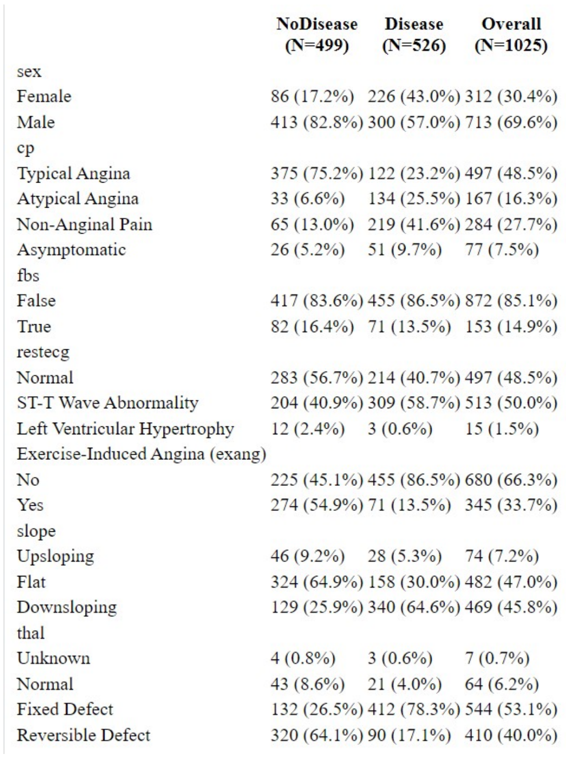
	NoDisease (N=499)	Disease (N=526)	Overall (N=1025)
Age			
Mean (SD)	56.6 (7.91)	52.4 (9.63)	54.4 (9.07)
Median [Min, Max]	58.0 [35.0, 77.0]	52.0 [29.0, 76.0]	56.0 [29.0, 77.0]
Resting Blood Pressure (trestbps)			
Mean (SD)	134 (18.6)	129 (16.1)	132 (17.5)
Median [Min, Max]	130 [100, 200]	130 [94.0, 180]	130 [94.0, 200]
Cholesterol (chol)			
Mean (SD)	251 (49.6)	241 (53.0)	246 (51.6)
Median [Min, Max]	249 [131, 409]	234 [126, 564]	240 [126, 564]
Maximum Heart Rate Achieved (thalach)			
Mean (SD)	139 (22.6)	159 (19.1)	149 (23.0)
Median [Min, Max]	142 [71.0, 195]	162 [96.0, 202]	152 [71.0, 202]
ST Depression (oldpeak)			
Mean (SD)	1.60 (1.29)	0.570 (0.771)	1.07 (1.18)
Median [Min, Max]	1.40 [0, 6.20]	0.200 [0, 4.20]	0.800 [0, 6.20]
Number of Major Vessels (ca)			
Mean (SD)	1.16 (1.03)	0.371 (0.871)	0.754 (1.03)
Median [Min, Max]	1.00 [0, 4.00]	0 [0, 4.00]	0 [0, 4.00]

Figure 9: Summary Statistics table for the numerical variables

This table presents the summary statistics for key numerical variables in the **Heart Disease dataset**, stratified by *NoDisease* and *Disease* groups. The sample sizes for each group are displayed: *N=499* for *NoDisease* and *N=526* for *Disease*, with a total of *N=1025*.

For each variable, the **mean** and **standard deviation (SD)** are shown, along with the **median**, and **minimum** and **maximum** values.

- **Age**: The mean age in the *NoDisease* group is *56.6 years* (SD = 7.91), slightly higher than the *Disease* group (*52.4 years*, SD = 9.63). - **Resting Blood Pressure (trestbps)**: The *NoDisease* group has a higher mean resting blood pressure (*134 mm Hg*, SD = 18.6) compared to the *Disease* group (*129 mm Hg*, SD = 16.1). - **Cholesterol (chol)**: The mean cholesterol level is slightly higher in the *NoDisease* group (*251 mg/dL*) than in the *Disease* group (*241 mg/dL*). - **Maximum Heart Rate (thalach)**: The *Disease* group exhibits a higher mean heart rate (*159 bpm*) compared to the *NoDisease* group (*139 bpm*). - **ST Depression (oldpeak)**: The *NoDisease* group has a significantly higher mean ST depression (*1.60*, SD = 1.29) than the *Disease* group (*0.57*, SD = 0.77). - **Number of Major Vessels (ca)**: The *NoDisease* group has a mean of *1.16*, compared to *0.37* for the *Disease* group.



	NoDisease (N=499)	Disease (N=526)	Overall (N=1025)
sex			
Female	86 (17.2%)	226 (43.0%)	312 (30.4%)
Male	413 (82.8%)	300 (57.0%)	713 (69.6%)
cp			
Typical Angina	375 (75.2%)	122 (23.2%)	497 (48.5%)
Atypical Angina	33 (6.6%)	134 (25.5%)	167 (16.3%)
Non-Anginal Pain	65 (13.0%)	219 (41.6%)	284 (27.7%)
Asymptomatic	26 (5.2%)	51 (9.7%)	77 (7.5%)
fbs			
False	417 (83.6%)	455 (86.5%)	872 (85.1%)
True	82 (16.4%)	71 (13.5%)	153 (14.9%)
restecg			
Normal	283 (56.7%)	214 (40.7%)	497 (48.5%)
ST-T Wave Abnormality	204 (40.9%)	309 (58.7%)	513 (50.0%)
Left Ventricular Hypertrophy	12 (2.4%)	3 (0.6%)	15 (1.5%)
Exercise-Induced Angina (exang)			
No	225 (45.1%)	455 (86.5%)	680 (66.3%)
Yes	274 (54.9%)	71 (13.5%)	345 (33.7%)
slope			
Upsloping	46 (9.2%)	28 (5.3%)	74 (7.2%)
Flat	324 (64.9%)	158 (30.0%)	482 (47.0%)
Downsloping	129 (25.9%)	340 (64.6%)	469 (45.8%)
thal			
Unknown	4 (0.8%)	3 (0.6%)	7 (0.7%)
Normal	43 (8.6%)	21 (4.0%)	64 (6.2%)
Fixed Defect	132 (26.5%)	412 (78.3%)	544 (53.1%)
Reversible Defect	320 (64.1%)	90 (17.1%)	410 (40.0%)

Figure 10: Summary Statistics table for the categorical variables

This table summarizes categorical variables in the **Heart Disease dataset**, stratified by *NoDisease* and *Disease* groups, with proportions provided for each category.

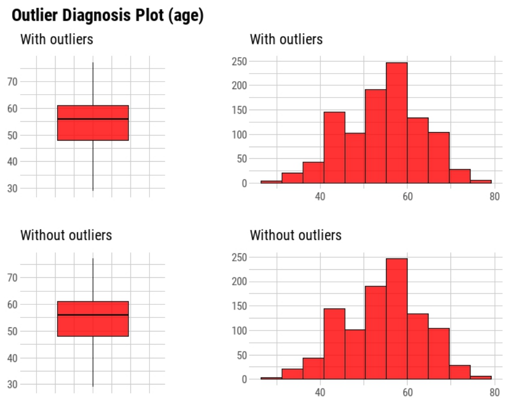
- **Sex**: The *NoDisease* group is predominantly male (*82.8%*), while the *Disease* group has a higher proportion of females (*43.0%*). - **Chest Pain Type (cp)**: In the *NoDisease* group, *75.2%* have *Typical Angina*, compared to *23.2%* in the *Disease* group, where *41.6%* report *Non-Anginal Pain*. - **Fasting Blood Sugar (fbs)**: The majority of both groups have normal fasting blood sugar, with *83.6%* in the *NoDisease* group and *86.5%* in the *Disease* group. - **Resting Electrocardiographic Results (restecg)**: *ST-T Wave Abnormality* is more common in the *Disease* group (*58.7%*) compared to *40.9%* in the *NoDisease* group. - **Exercise-Induced Angina (exang)**: *Exercise-Induced Angina* is much more frequent in the *NoDisease* group (*54.9%*), whereas the *Disease* group reports a significantly lower percentage (*13.5%*). - **Slope of ST Segment (slope)**: The *NoDisease* group predominantly has a *Flat* slope (*64.9%*), while the *Disease* group reports a higher frequency of *Downsloping* (*64.6%*). - **Thalassemia (thal)**: *Reversible Defect* is most prevalent in both groups, especially in the *NoDisease* group (*64.1%*).

This table highlights important categorical trends between the two groups, providing insight into the distribution of key clinical variables.

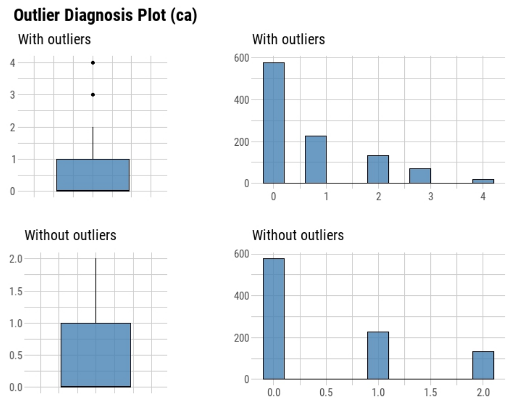
Outlier Analysis Figure 11 presents six **Outlier Diagnosis Plots**, showing histograms for different variables from the **Heart Disease dataset**, both with and without outliers.

- **(a) Age Outlier plot**: The histogram shows the distribution of *age*, where a few outliers above 65 years are visible. Removing outliers reveals a more normalized distribution. - **(b) Number of Major Vessels Outlier plot**: The *ca* variable exhibits outliers, particularly in higher vessel counts (*3* and *4*). Removing these highlights the prevalence of 0 vessels. - **(c) Resting Heart Rate Outlier plot (trestbps)**: There are outliers above 180 mm Hg, and once removed, the distribution centers around 120 to 140 mm Hg. - **(d) ST Depression Outlier plot (oldpeak)**: Significant outliers are observed in ST depression levels, particularly beyond *4.0*. Removing them reveals a skewed distribution centered around *0.5*. - **(e) Cholesterol Level Outlier plot (chol)**: Cholesterol levels show extreme outliers above *400 mg/dL*. After removing outliers, the distribution becomes more consistent, peaking around *200–250 mg/dL*. - **(f) Thallium Test (thalach) Outlier plot**: The *maximum heart rate achieved* has notable outliers above *200 bpm*, and removing them focuses the distribution around *140-170 bpm*.

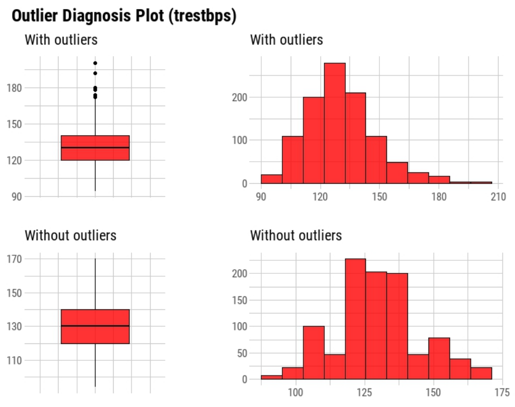
These plots visually emphasize the effect of outliers on the distribution of key clinical variables, providing a clearer picture of the data when outliers are excluded.



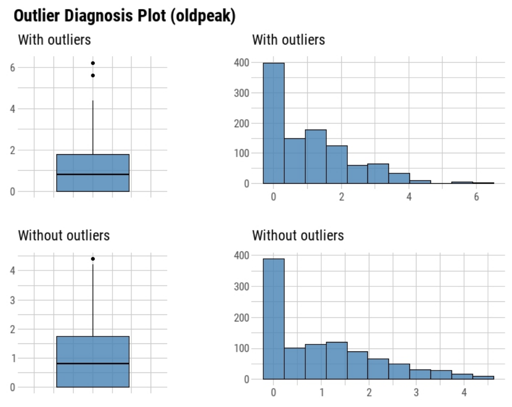
(a) *Age* Outlier plot



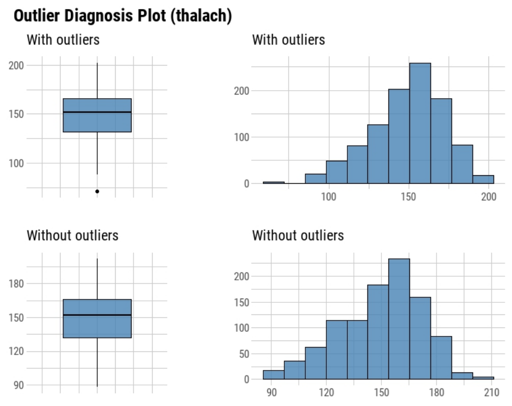
(b) *Number of Major Vessels* Outlier plot



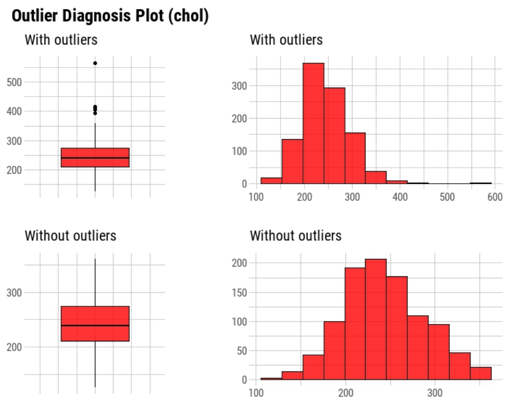
(c) *Resting Heart Rate* Outlier plot



(d) *ST Depression* Outlier plot

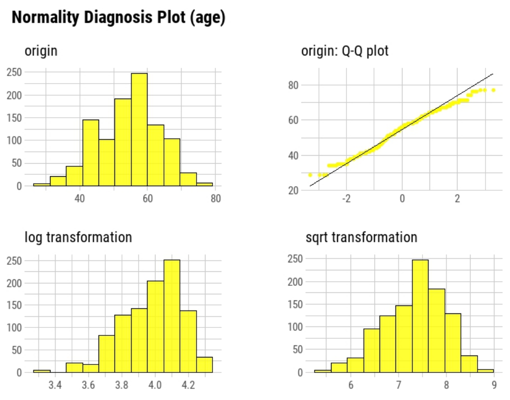


(f) *Thallium Test maximum Heart Rate achieved* Outlier plot

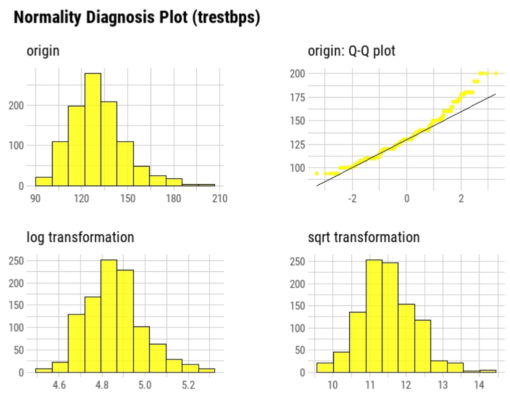


(e) *Cholesterol Level* Outlier plot

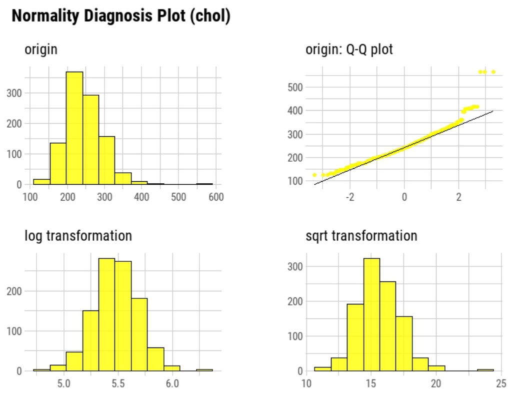
Figure 11: Outlier Diagnosis Plots



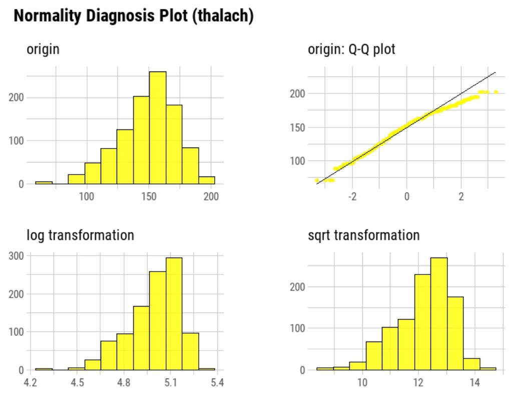
(a) *Age* Normality plot



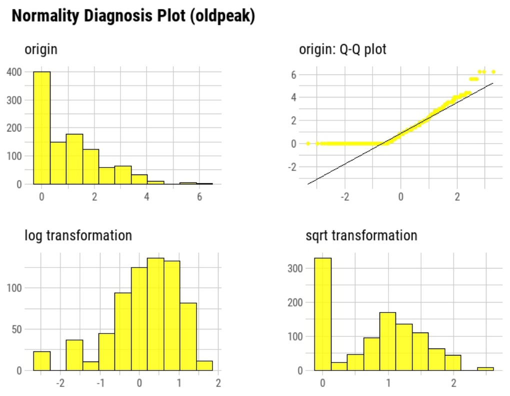
(b) *Resting Heart Rate* Normality plot



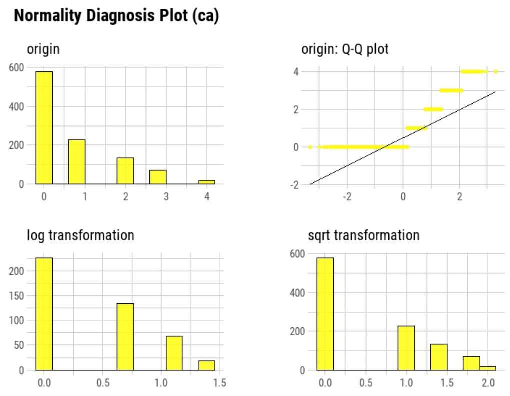
(c) *Cholesterol Levels* Normality plot



(d) *Thallium Test maximum Heart Rate achieved* Normality plot



(e) *ST Depression* Normality plot



(f) *Number of Major Vessels* Normality plot

Figure 12: Normality Diagnosis Plots

Figure 12 above displays six **Normality Diagnosis Plots** for key variables in the **Heart Disease dataset**, illustrating the distributions and transformations of each variable.

- **(a) Age Normality plot**: The original age distribution shows slight skewness, and the

Q-Q plot confirms minor deviations from normality. Log and square root transformations attempt to normalize the distribution. - **(b) Resting Heart Rate (trestbps) Normality plot**: The original resting heart rate is moderately skewed, as seen in the *Q-Q plot*. Both transformations reduce the skewness but do not fully normalize the data. - **(c) Cholesterol Levels (chol) Normality plot**: The cholesterol levels display significant skewness in the original distribution, and the *Q-Q plot* confirms this. Log and square root transformations successfully reduce skewness. - **(d) Thallium Test maximum Heart Rate achieved (thalach) Normality plot**: The distribution is less skewed than other variables. The *Q-Q plot* indicates a reasonable fit to normality, with transformations slightly improving symmetry. - **(e) ST Depression (oldpeak) Normality plot**: The original distribution is heavily skewed, as seen in the *Q-Q plot*. Log and square root transformations substantially reduce the skewness. - **(f) Number of Major Vessels (ca) Normality plot**: The original variable has a skewed, discrete distribution. Both transformations reduce skewness but do not fully normalize this categorical-like variable.

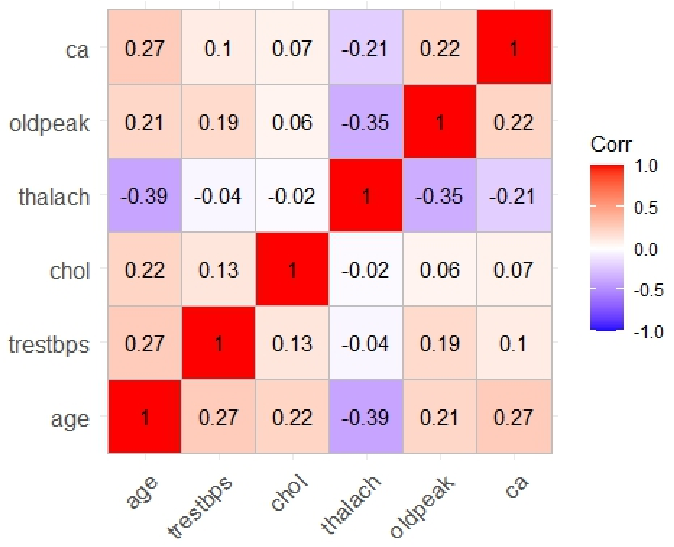
These plots offer insights into the normality of each variable and demonstrate the effectiveness of log and square root transformations in reducing skewness.

Correlation Analysis Figure 13 presents two important visualizations: the **Correlation plot** (a) and the **Pair plot** (b) for key numerical variables in the **Heart Disease dataset**.

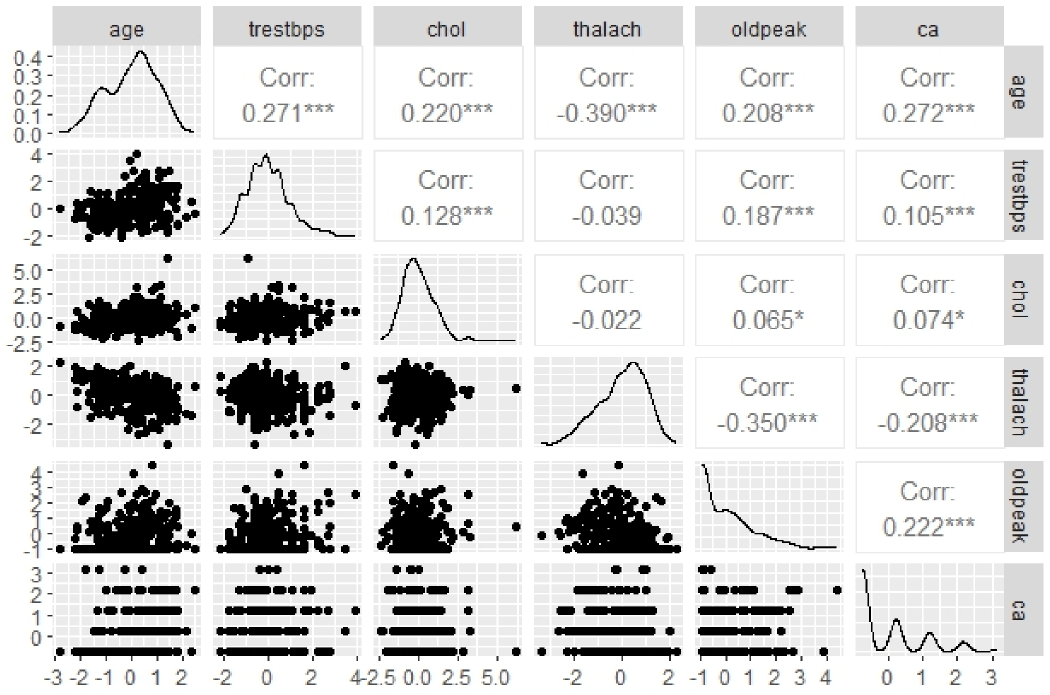
- **(a) Correlation plot**: This heatmap displays the pairwise correlations between variables, with color intensity reflecting the strength of the correlation. Positive correlations are shaded in red, while negative correlations are shaded in blue. For instance, *age* shows a positive correlation with *resting blood pressure (trestbps)* (*0.27*), while *maximum heart rate achieved (thalach)* has a strong negative correlation with *age* (*-0.39*) and *ST depression (oldpeak)* (*-0.35*).

- **(b) Pair plot**: This matrix of scatter plots provides a detailed view of the relationships between variables, with diagonal elements showing the distribution of each variable. For example, the scatter plot between *age* and *thalach* shows a negative relationship, consistent with the correlation coefficient of *-0.39*. The *resting blood pressure (trestbps)* and *age* scatter plot exhibits a positive trend (*0.27* correlation).

The pair plot helps visualize the distribution and relationships between variables, while the correlation plot numerically summarizes their strength and direction. These visualizations provide key insights into how variables are interrelated within the dataset.



(a) *Correlation* plot



(b) *Pair* plot

Figure 13: Scatter plots and Correlation Plot

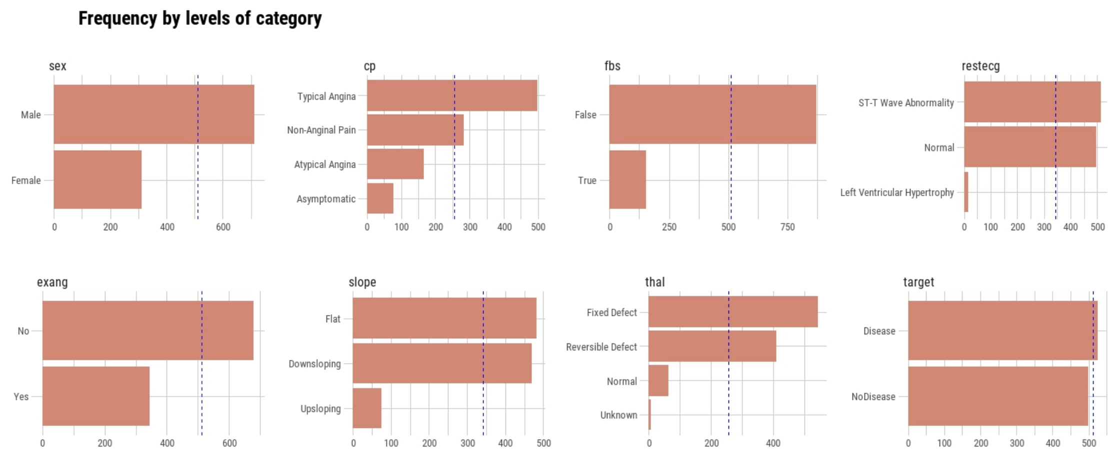


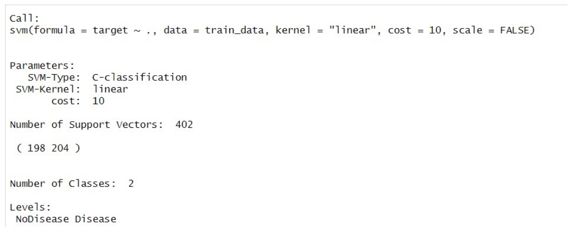
Figure 14: Categorical Variables and their levels

The figure presents **bar plots** showing the frequency distribution of categorical variables in the **Heart Disease dataset**. Each plot depicts the count of observations for different levels within each categorical variable.

- **Sex**: The dataset is predominantly male (*713 males* compared to *312 females*). - **Chest Pain Type (cp)**: *Typical Angina* is the most common category, while *Asymptomatic* is the least frequent. - **Fasting Blood Sugar (fbs)**: The majority of patients have *false* for high fasting blood sugar. - **Resting ECG (restecg)**: Most patients have either *Normal* or *ST-T Wave Abnormality* results. - **Exercise-Induced Angina (exang)**: *No* is more frequent than *Yes*. - **Slope of ST Segment (slope)**: *Flat* is the most common, followed by *Downsloping*. - **Thalassemia (thal)**: *Reversible Defect* is the dominant category. - **Target**: There are slightly more patients with heart disease (*526*) compared to those without (*499*).

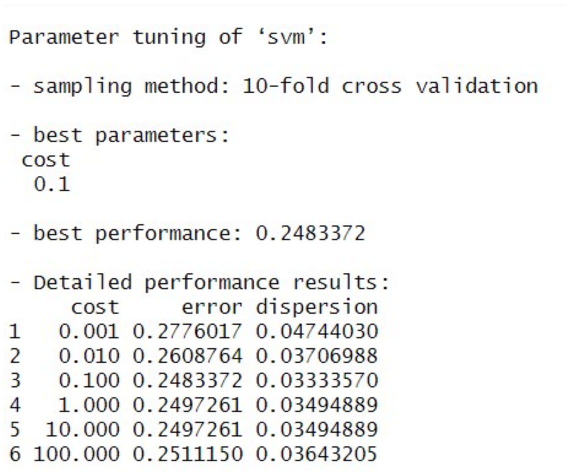
These bar plots provide a clear visualization of the distribution of key categorical features in the dataset.

We shall next use Support Vector Machines to classify and predict based on the predictors whether a person has heart disease or not. The utilization of Support Vector Machines in the medical data analysis is well documented and attested in the literature. (Luo & Chen, 2020) (Mousavi & Hashemi, 2021)



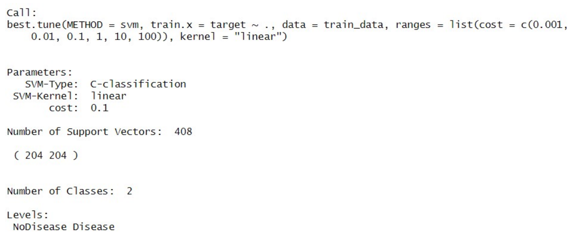
```
Call:
svm(formula = target ~ ., data = train_data, kernel = "linear", cost = 10, scale = FALSE)


Parameters:
   SVM-Type:  C-classification
 SVM-Kernel:  linear
       cost:  10

Number of Support Vectors:  402

 ( 198 204 )


Number of Classes:  2

Levels:
 NoDisease Disease
```

(a) *Support Vector Classifier Training*



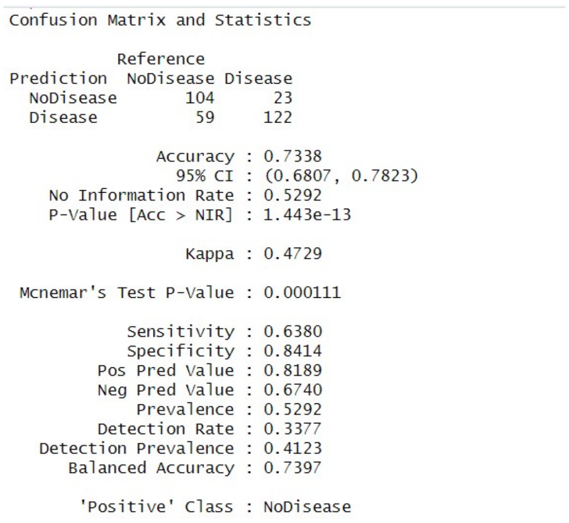
```
Parameter tuning of 'svm':

- sampling method: 10-fold cross validation

- best parameters:
 cost
  0.1

- best performance: 0.2483372

- Detailed performance results:
     cost     error dispersion
1   0.001 0.2776017 0.04744030
2   0.010 0.2608764 0.03706988
3   0.100 0.2483372 0.03333570
4   1.000 0.2497261 0.03494889
5  10.000 0.2497261 0.03494889
6 100.000 0.2511150 0.03643205
```

(b) *Tuning of the Support Vector Classifier*



```
Call:
best.tune(METHOD = svm, train.x = target ~ ., data = train_data, ranges = list(cost = c(0.001, 
    0.01, 0.1, 1, 10, 100)), kernel = "linear")


Parameters:
   SVM-Type:  C-classification
 SVM-Kernel:  linear
       cost:  0.1

Number of Support Vectors:  408

 ( 204 204 )


Number of Classes:  2

Levels:
 NoDisease Disease
```

(c) *Best Support Vector Classifier Model*



```
Confusion Matrix and Statistics

           Reference
Prediction  NoDisease Disease
  NoDisease       104      23
  Disease          59     122
                                          
               Accuracy : 0.7338          
                 95% CI : (0.6807, 0.7823)
    No Information Rate : 0.5292          
    P-Value [Acc > NIR] : 1.443e-13       
                                          
                  Kappa : 0.4729          
                                          
 Mcnemar's Test P-Value : 0.000111        
                                          
            Sensitivity : 0.6380          
            Specificity : 0.8414          
         Pos Pred Value : 0.8189          
         Neg Pred Value : 0.6740          
             Prevalence : 0.5292          
         Detection Rate : 0.3377          
   Detection Prevalence : 0.4123          
      Balanced Accuracy : 0.7397          
                                          
       'Positive' Class : NoDisease       
```

(d) Support Vector Classifier Confusion Matrix on the train set

Figure 15: Support Vector Machine with a Linear Kernel also called a *Support Vector Classifier*

The above described Support Vector Classifier model is a **Support Vector Machine (SVM)** with a *Linear Kernel*, applied to classify heart disease presence using a dataset split into training and testing sets. The SVM is used to find the optimal hyperplane that separates the classes, which in this case are *NoDisease* and *Disease*. This process is tuned using cross-validation and a grid search of cost parameters to enhance the model's accuracy.

Initially, the SVM model was trained using a range of *cost* parameters `(0.001, 0.01, 0.1, 1, 10, 100)` to find the optimal value of `cost`. The tuning process utilized **10-fold cross-validation**, which splits the dataset into 10 subsets, using nine for training and one for validation. The error rates across these folds are averaged to give a reliable estimate of model performance.

The **tuning results** indicate that the best performing `cost` value was *0.1*, which achieved the lowest error rate (*0.2483*). A detailed summary of error rates for each cost value shows a slight difference between lower and higher values, with the best performance corresponding to smaller cost values, which penalize misclassification less aggressively, leading to better generalization.

The **SVM Model Summary** for the best-tuned model reveals that:

- The model uses `C-classification`, meaning it is a binary classifier optimized to separate the *NoDisease* and *Disease* classes.
- The model uses a `linear kernel`, which attempts to separate the data linearly by maximizing the margin between the two classes.
- The optimal `cost` parameter is *0.1*, which controls the trade-off between allowing misclassification and ensuring margin maximization.
- There are **408 support vectors**, with `204` for each class. Support vectors are critical data points that lie closest to the hyperplane and help define the decision boundary.

The **Confusion Matrix** for the best-tuned model on the test data shows an overall **accuracy of 73.38%**, with a *95% confidence interval* ranging from *68.07% to 78.23%.* The model is significantly better than random guessing (*P-value is less than 0.001*).

Key performance metrics include:

- **Sensitivity** (Recall): *63.80%*, indicating the proportion of actual *Disease* cases correctly predicted.
- **Specificity**: *84.14%*, indicating the proportion of actual *NoDisease* cases correctly predicted.
- **Precision**: *81.89%*, showing the proportion of positive predictions (i.e., *NoDisease*) that are correct.

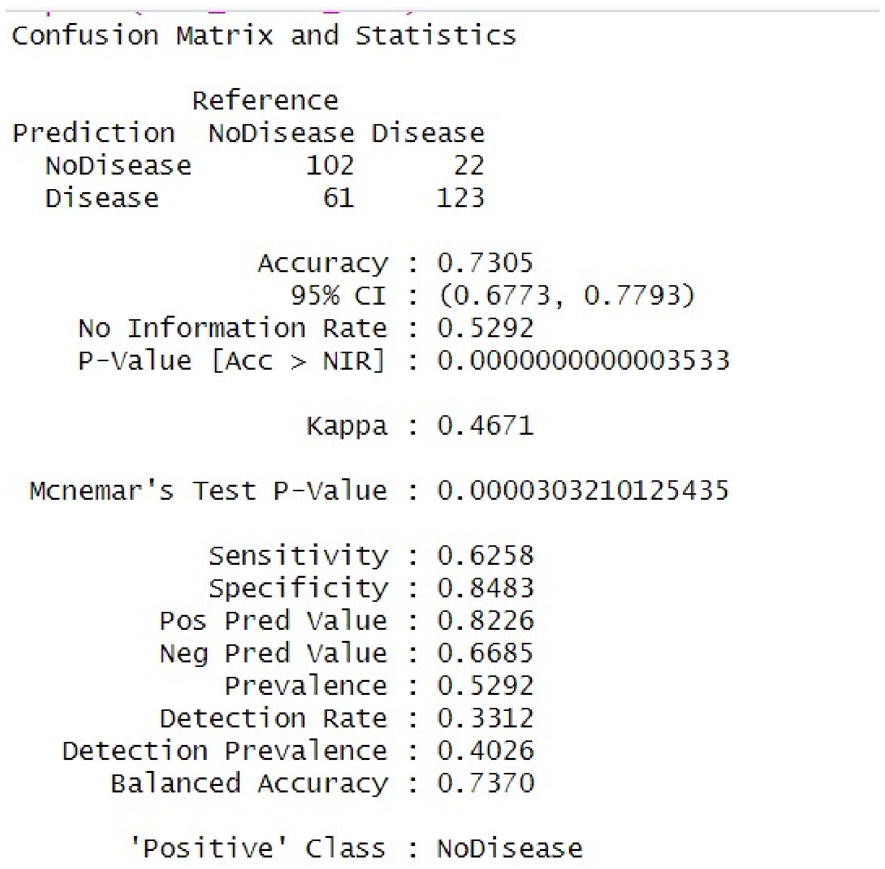
- **Kappa Score**: *0.4729*, which measures agreement between predicted and actual values, where *1.0* indicates perfect agreement.

The **McNemar's Test** P-value of *0.0001* indicates a significant difference between the misclassification of the two classes, implying that the model's ability to discriminate between *NoDisease* and *Disease* is meaningful.

The **support vectors** play a crucial role in determining the hyperplane, especially for linear SVMs, where data points closest to the boundary define the classification decision. A higher number of support vectors indicates that the boundary may be more sensitive to data perturbations.

For comparison, a model with `cost = 10` was also evaluated, which resulted in *402 support vectors* and similar performance. However, the cross-validation showed that lower cost values (around *0.1*) achieved the best balance between margin maximization and misclassification minimization, leading to better overall performance on unseen data.

In conclusion, the **SVM with a linear kernel** demonstrates a robust capacity to classify heart disease with a reasonably high accuracy of *73.38%*, favoring a low cost parameter to minimize error. The model's strength lies in its ability to generalize well on the test set, as demonstrated by the significant difference in classification performance over random guessing. Further improvements could involve exploring non-linear kernels or incorporating additional features to enhance the model's predictive power.



```
Confusion Matrix and Statistics

           Reference
Prediction  NoDisease Disease
  NoDisease       102      22
  Disease          61     123

               Accuracy : 0.7305
                 95% CI : (0.6773, 0.7793)
    No Information Rate : 0.5292
    P-Value [Acc > NIR] : 0.0000000000003533

                  Kappa : 0.4671

 Mcnemar's Test P-Value : 0.0000303210125435

            Sensitivity : 0.6258
            Specificity : 0.8483
         Pos Pred Value : 0.8226
         Neg Pred Value : 0.6685
             Prevalence : 0.5292
         Detection Rate : 0.3312
   Detection Prevalence : 0.4026
      Balanced Accuracy : 0.7370

       'Positive' Class : NoDisease
```

Figure 16: Support Vector Classifier Confusion Matrix on the test set

The above **Confusion Matrix** summarizes the performance of the *SVM with a Linear Kernel* on the test data. The model achieved an **accuracy** of *73.05%* with a *95% confidence interval* ranging from *67.73% to 77.93%.* The **P-value** is significant, indicating the model performs better than random guessing (*P is less than 0.001*).

Key performance metrics include:

- **Sensitivity (Recall)**: *62.58%*, representing the proportion of actual *Disease* cases correctly predicted.

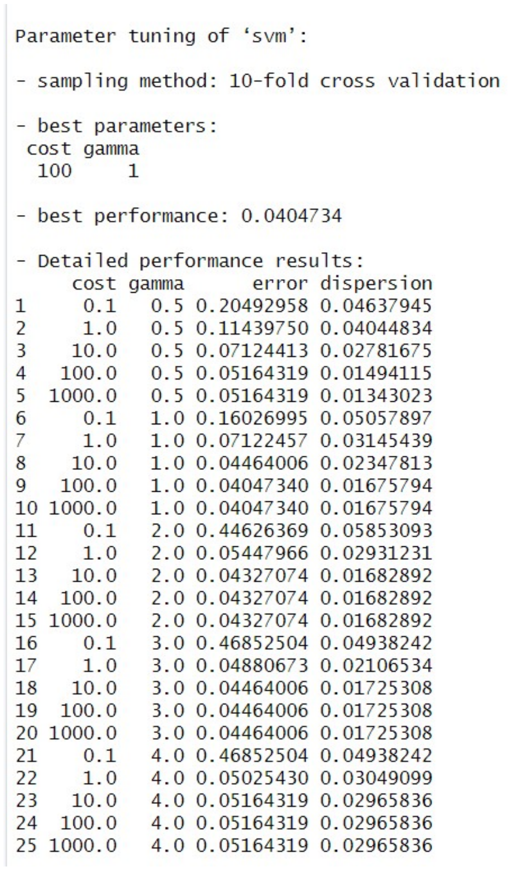
- **Specificity**: *84.83%*, showing the proportion of *NoDisease* cases correctly identified.

- **Precision**: *82.26%*, indicating the accuracy of positive *NoDisease* predictions.

- **Kappa Score**: *0.4671*, which reflects moderate agreement between predicted and actual values.

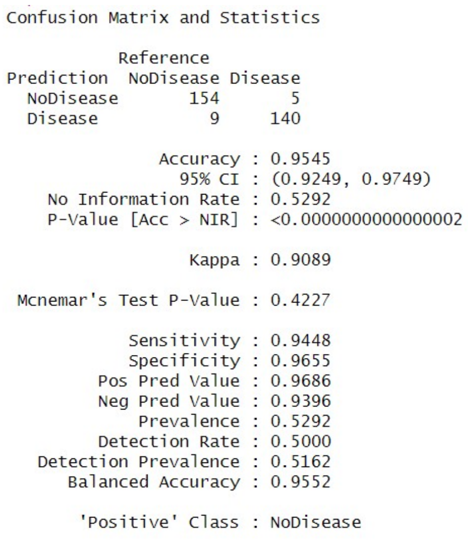
The **Balanced Accuracy** is *73.70%*, and the **McNemar's Test P-value** is *0.00003*, indicating a significant difference in misclassification between classes. Overall, the model performs well, particularly in distinguishing between the two classes.

Support Vector Machine with a *Radial Kernel*



```
Parameter tuning of 'svm':

- sampling method: 10-fold cross validation

- best parameters:
 cost gamma
  100     1

- best performance: 0.0404734

- Detailed performance results:
     cost gamma      error dispersion
1     0.1   0.5 0.20492958 0.04637945
2     1.0   0.5 0.11439750 0.04044834
3    10.0   0.5 0.07124413 0.02781675
4   100.0   0.5 0.05164319 0.01494115
5  1000.0   0.5 0.05164319 0.01343023
6     0.1   1.0 0.16026995 0.05057897
7     1.0   1.0 0.07122457 0.03145439
8    10.0   1.0 0.04464006 0.02347813
9   100.0   1.0 0.04047340 0.01675794
10 1000.0   1.0 0.04047340 0.01675794
11    0.1   2.0 0.44626369 0.05853093
12    1.0   2.0 0.05447966 0.02931231
13   10.0   2.0 0.04327074 0.01682892
14  100.0   2.0 0.04327074 0.01682892
15 1000.0   2.0 0.04327074 0.01682892
16    0.1   3.0 0.46852504 0.04938242
17    1.0   3.0 0.04880673 0.02106534
18   10.0   3.0 0.04464006 0.01725308
19  100.0   3.0 0.04464006 0.01725308
20 1000.0   3.0 0.04464006 0.01725308
21    0.1   4.0 0.46852504 0.04938242
22    1.0   4.0 0.05025430 0.03049099
23   10.0   4.0 0.05164319 0.02965836
24  100.0   4.0 0.05164319 0.02965836
25 1000.0   4.0 0.05164319 0.02965836
```

(a) *Support Vector Machine with Radial Kernel*



```
Confusion Matrix and Statistics

          Reference
Prediction  NoDisease Disease
  NoDisease       154       5
  Disease           9     140

               Accuracy : 0.9545
                 95% CI : (0.9249, 0.9749)
    No Information Rate : 0.5292
    P-Value [Acc > NIR] : <0.0000000000000002

                  Kappa : 0.9089

 Mcnemar's Test P-Value : 0.4227

            Sensitivity : 0.9448
            Specificity : 0.9655
         Pos Pred Value : 0.9686
         Neg Pred Value : 0.9396
             Prevalence : 0.5292
         Detection Rate : 0.5000
   Detection Prevalence : 0.5162
      Balanced Accuracy : 0.9552

       'Positive' Class : NoDisease
```

(b) ***Untuned SVM*** *on Test Data Confusion Matrix*

Figure 17: Support Vector Machine with a *Radial Kernel*

The model described above is a **Support Vector Machine (SVM)** with a *Radial Kernel*, applied to predict the presence of heart disease using a dataset. Unlike the linear kernel, the radial kernel is used to map the data into a higher-dimensional space, enabling it to handle non-linearly separable data more effectively.

The model was trained and tuned using **10-fold cross-validation** to find the optimal `cost` and `gamma` parameters. The radial kernel introduces the `gamma` parameter, which defines the influence of a single training example. A higher `gamma` results in a more complex decision boundary that fits the training data more closely, while a lower `gamma` results in a smoother decision boundary.

The **tuning results** reveal that the best performing model had `cost = 100` and `gamma = 1`. This combination achieved the best cross-validation performance with an error rate of *0.0405*, meaning the model has strong generalization capability. A detailed summary of the parameter grid search shows that higher `cost` and `gamma` values provided slightly better error rates, indicating the importance of capturing complex data relationships.

The **SVM Model Summary** for the best-tuned radial kernel model reveals:

- The model uses `C-classification`, designed for binary classification.
- The kernel is set to `radial`, which allows the decision boundary to be non-linear.
- The optimal `cost` parameter is *100*, while `gamma` is *1*, providing a balance between fitting the training data and ensuring generalization.

The **Confusion Matrix** on the test data reveals that the untuned model achieved an **accuracy** of *95.45%* with a *95% confidence interval* of *92.49% to 97.49%*, indicating high performance. The model's **P-value** is highly significant (*P is less than 0.001*), meaning the model is far better than random guessing.

Key metrics for this model include:

- **Sensitivity (Recall)**: *94.48%*, indicating that the model correctly identified *Disease* cases most of the time.
- **Specificity**: *96.55%*, showing the model's strong ability to identify *NoDisease* cases.
- **Precision**: *96.86%*, indicating the model's ability to avoid false positives when predicting *NoDisease.*
- **Negative Predictive Value (NPV)**: *93.96%*, reflecting how well the model avoids false negatives when predicting *Disease.*
- **Kappa**: *0.9089*, reflecting a very high level of agreement between the predicted and actual outcomes, demonstrating that the model is well-calibrated.

- **Balanced Accuracy**: *95.52%*, combining sensitivity and specificity to show how well the model performs across both classes.

In terms of **detection rates**, the model correctly identifies *50.00%* of all true positive *Disease* cases, indicating balanced detection between the classes.

In addition to the confusion matrix, the **McNemar's Test** P-value of *0.4227* suggests that there is no significant difference in how the model misclassifies the two classes, implying a balanced performance between *NoDisease* and *Disease* predictions.

The radial kernel's flexibility in mapping data to higher dimensions allows it to handle more complex relationships compared to the linear kernel. This, combined with an optimal choice of `cost` and `gamma`, results in a highly accurate model with excellent classification metrics across both classes. The model demonstrates strong generalization to the test set, with a low error rate and high precision and recall, making it a robust classifier for the heart disease dataset.

In conclusion, the **SVM with a Radial Kernel** provides superior performance over linear models, especially when dealing with complex, non-linear data distributions. However the tuning process will ensure that the decision boundary effectively balances overfitting and generalization, achieving high accuracy and strong class discrimination on the test data as we shall see next.

Fitting the tuned SVM with Radial Kernel on the test data

The **Confusion Matrix** in Figure 18 summarizes the performance of a *tuned Support Vector Machine (SVM) with a Radial Kernel* on the test data. The model achieved an exceptionally high **accuracy** of *99.03%*, with a *95% confidence interval* ranging from *97.18% to 99.80%*. The highly significant **P-value** of *is less than 0.0000000000000002* confirms the model's superior performance compared to random guessing.

Key performance metrics include:

- **Sensitivity (Recall)**: *100.00%*, indicating that the model correctly identified all true positive cases of *Disease*, making it highly reliable in detecting disease.

- **Specificity**: *97.93%*, demonstrating the model's ability to accurately predict *NoDis-*

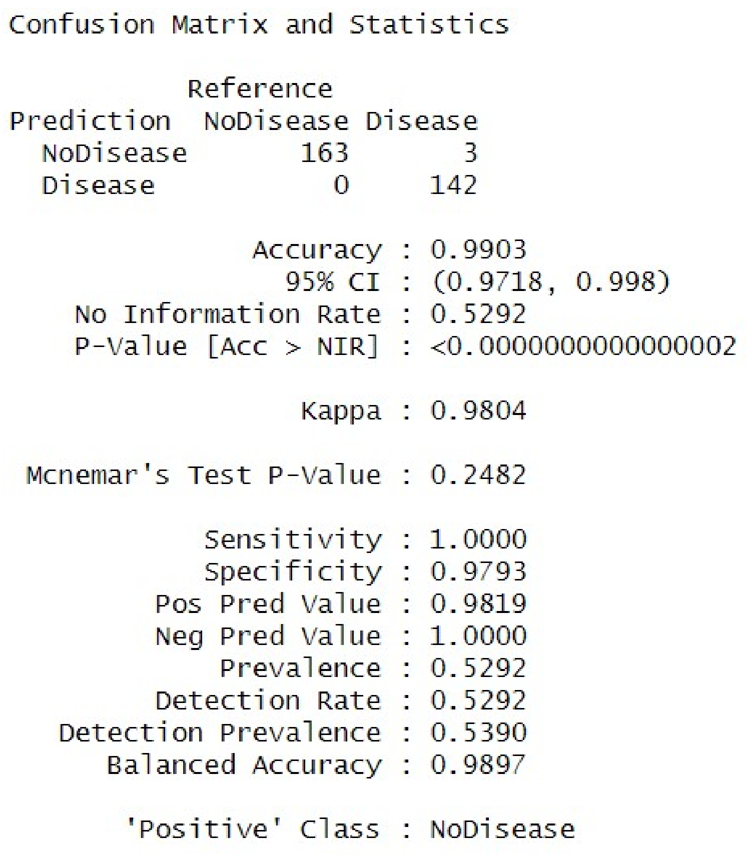
ease cases, with only three false positives.

- **Precision (Positive Predictive Value)**: *98.19%*, highlighting the model's accuracy when it predicts the *NoDisease* class.
- **Negative Predictive Value (NPV)**: *100.00%*, meaning that every *NoDisease* prediction is completely accurate.
- **Kappa Score**: *0.9804*, indicating excellent agreement between the predicted and actual classes, reflecting a near-perfect model fit.
- **Balanced Accuracy**: *98.97%*, combining both sensitivity and specificity to show the model's overall performance in handling both classes effectively.

The **McNemar's Test P-value** of *0.2482* suggests no significant difference in the misclassification of the two classes, confirming balanced performance between predicting *NoDisease* and *Disease*.

In terms of **detection rates**, the model correctly identified all *Disease* cases, resulting in a detection rate of *100.00%* for the *Disease* class, and a detection rate of *52.92%* for *NoDisease*, reflecting the model's strong performance in both identifying and classifying the presence and absence of disease.

This **tuned Radial Kernel SVM** exhibits excellent overall performance, effectively balancing sensitivity and specificity, with minimal misclassification. This makes it a highly accurate and reliable model for predicting heart disease in this dataset.



```
Confusion Matrix and Statistics

            Reference
Prediction  NoDisease Disease
  NoDisease       163       3
  Disease           0     142

               Accuracy : 0.9903
                 95% CI : (0.9718, 0.998)
    No Information Rate : 0.5292
    P-Value [Acc > NIR] : <0.0000000000000002

                  Kappa : 0.9804

 Mcnemar's Test P-Value : 0.2482

            Sensitivity : 1.0000
            Specificity : 0.9793
         Pos Pred Value : 0.9819
         Neg Pred Value : 1.0000
             Prevalence : 0.5292
         Detection Rate : 0.5292
   Detection Prevalence : 0.5390
      Balanced Accuracy : 0.9897

       'Positive' Class : NoDisease
```

Figure 18: Classification Matrix of the tuned SVM Model on the test set

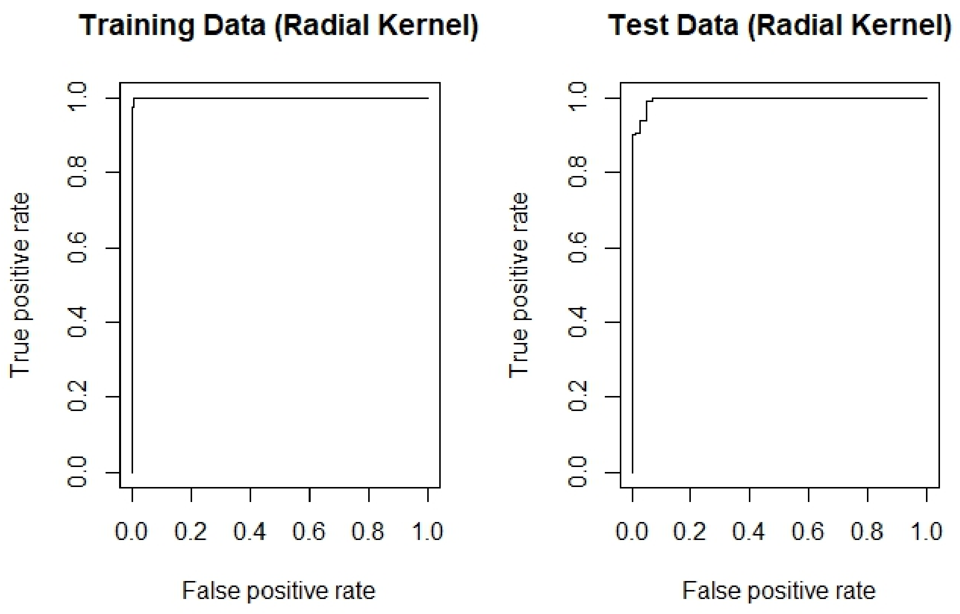


Figure 19: ROC Curve of the SVM with Radial Kernel

Figure 19 presents **ROC curves** for the *Tuned Support Vector Machine (SVM) with a Radial Kernel* on both the training and test datasets.

- The **Training Data (Radial Kernel)** ROC curve on the left shows a perfect model, as evidenced by the curve reaching the top-left corner, indicating a **true positive rate (sensitivity)** of *1.00* and a **false positive rate** of *0.00*. This indicates the model classifies all training instances without any error.

- The **Test Data (Radial Kernel)** ROC curve on the right exhibits a near-perfect performance, with the curve closely hugging the top-left corner. This reflects excellent model generalization on unseen data, maintaining a high **true positive rate** while keeping the **false positive rate** very low.

Both curves demonstrate the *Radial SVM's* exceptional capacity for classification, with almost flawless discrimination between *NoDisease* and *Disease* in both the training and test datasets.

Interpretation and Recommendations

The **Support Vector Machine (SVM) analysis** of the *Heart Disease dataset* yielded a high-performing model for predicting the presence of heart disease, especially with the *tuned radial kernel model*. The **tuned radial SVM** achieved an impressive **accuracy** of *99.03%*, **perfect sensitivity** (*100%*), and a **balanced accuracy** of *98.97%*, indicating it can accurately predict both the presence and absence of heart disease. This level of performance suggests that the model is highly reliable for use in medical diagnostics. Support Vector Machines are being integrated into deep learning (Verma & Sharma, 2023) and neural networks (Tian, Shi, & Liu, 2012) and are well attested in the literature for their effectiveness in mediacl prognostication.

Interpretation of Results:

- **High Sensitivity (100%)**: The model correctly identifies all patients with heart disease, minimizing the risk of false negatives. This is critical in medical settings where missing a diagnosis could lead to severe consequences.

- **Strong Specificity (97.93%)**: The model performs well in predicting patients without heart disease, reducing the number of false positives. This ensures individuals are not mistakenly diagnosed, avoiding unnecessary treatments or interventions.

- **Balanced Accuracy (98.97%)**: The model demonstrates strong overall performance across both classes, combining sensitivity and specificity.
- **Kappa (0.9804)**: This high kappa score indicates near-perfect agreement between predicted and actual outcomes, further validating the model's reliability.

Recommendations for the Data Owner:

- **Deployment for Clinical Use**: The model could be deployed in healthcare settings as a decision-support tool to assist clinicians in diagnosing heart disease. Given its high sensitivity, it is especially suitable for initial screenings where identifying all possible cases of heart disease is critical.
- **Model Improvement through Additional Variables**: Incorporating external variables such as lifestyle factors (e.g., smoking status, physical activity levels), dietary habits (e.g., cholesterol intake), and socioeconomic factors (e.g., income levels, education) could further enhance the model's predictive power. These variables could be collected from public databases like census.gov or other health-related surveys.
- **Periodic Model Updates**: As more patient data becomes available, retraining the model periodically will ensure it remains relevant and continues to perform at a high level, accounting for changes in population demographics or advancements in medical knowledge.
- **Clinical Validation**: Testing the model on new, unseen patient data is crucial to validate its performance in real-world clinical settings. This will ensure the model generalizes well beyond the training environment.

References

Luo, J., & Chen, K. (2020). Enhanced cancer detection using multi-instance support vector machines and deep learning techniques. *Expert Systems with Applications*, *159*, 113661. doi: 10.1016/j.eswa.2020.113661

Mehdi, A., & Ghazali, A. (2022). Support vector machine optimized by whale optimization algorithm for efficient text classification. *Computers, Materials & Continua*. doi: 10.32604/cmc.2022.023408

Mousavi, M., & Hashemi, R. (2021). A deep learning and support vector machine model for the classification of diabetic retinopathy stages. *Applied Soft Computing*, *113*, 107990. doi: 10.1016/j.asoc.2021.107990

Tian, Y., Shi, Y., & Liu, X. (2012). Recent advances on support vector machines research. *Technological and Economic Development of Economy*, *18*(1), 5-33. doi: 10.3846/20294913.2012.661205

Verma, R., & Sharma, A. (2023). Modified twin support vector machine for high-dimensional data classification. *IEEE Transactions on Neural Networks and Learning Systems*. doi: 10.1109/TNNLS.2023.3180464

Wu, K., Han, Z., & Du, Y. (2020). A novel support vector machine model for forecasting solar radiation using cloud coverage data. *Renewable Energy*, *155*, 328-339. doi: 10.1016/j.renene.2020.03.089

(Luo & Chen, 2020) (Mehdi & Ghazali, 2022) (Mousavi & Hashemi, 2021) (Tian et al., 2012) (Verma & Sharma, 2023) (Wu, Han, & Du, 2020)