Assignment_4

Tarun BT

2025-03-10

Imported necessary library files required for task

```
library(ggplot2)
library(lattice)
library(caret)
library(e1071)
library(dplyr)

##
## Attaching package: 'dplyr'

## The following objects are masked from 'package:stats':
##
##     filter, lag

## The following objects are masked from 'package:base':
##
##     intersect, setdiff, setequal, union

library(pROC)

## Type 'citation("pROC")' for a citation.

##
## Attaching package: 'pROC'

## The following objects are masked from 'package:stats':
##
##     cov, smooth, var

library(naivebayes)

## naivebayes 1.0.0 loaded

## For more information please visit:

## https://majkamichal.github.io/naivebayes/
```

Loading the data set

```
Heart_disease <- read.csv("C:\\Users\\Tarun\\OneDrive\\Desktop\\R
program\\4th Assignment\\Heart_disease.csv")
```

Checking for duplicate value and removing record from the data-set

```r
duplicates <- Heart_disease[duplicated(Heart_disease), ]
cat("Number of duplicate record ",nrow(duplicates),"\n")

## Number of duplicate record  1

print(duplicates)

##     Age Sex chest_pain_type Blood_Pressure Cholestrol Fasting_Blood_Sugar
## 165  38   1               1            138        175                   0
##     Rest_ECG MAX_HeartRate Exercise
## 165        1           173        0

df_unique <- Heart_disease %>% distinct() #data-set after removing the
duplicate value
nrow(df_unique)

## [1] 302
```

Creating dummy variables

```r
df_unique$Target <- ifelse(df_unique$MAX_HeartRate > 170, "Yes", "No")
df_unique$BP_New <- ifelse(df_unique$Blood_Pressure > 120, "Yes", "No")
```

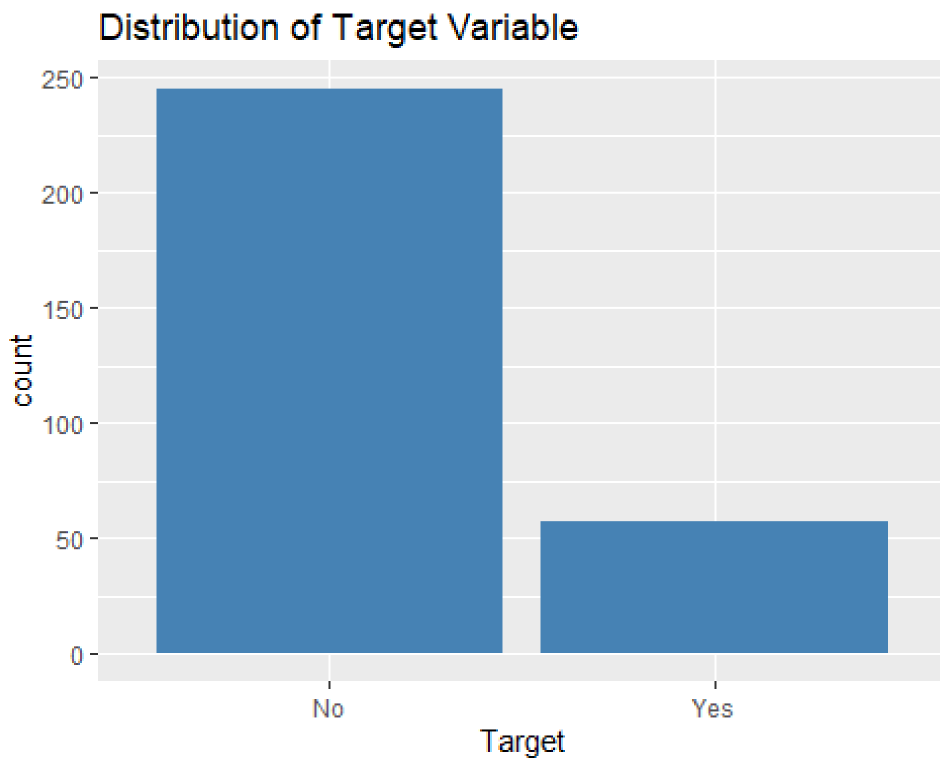
Q1: Initial Prediction Based on Target Distribution

```r
target_table <- table(df_unique$Target)
target_table

##
##   No Yes
## 245  57
```

Bar Plot for Target Distribution

```r
ggplot(df_unique, aes(x = Target)) + geom_bar(fill = "steelblue") +
ggtitle("Distribution of Target Variable")
```



Interpretation: The data set contains an imbalanced class distribution:

No Heart Disease (No): 245 occurrences.

Heart Disease (Yes): 57 occurrences.

Since the majority class is No Heart Disease, the model is likely to predict NO more frequently, which could lead to misleading accuracy in an imbalanced data set (without any further information,).

Q2: Analysis of the First 30 Records

```
Heart_disease30 <- df_unique[1:30, c("Target", "BP_New", "chest_pain_type")]
Object1 <- ftable(Heart_disease30)
Object1

##                chest_pain_type 0 1
## Target BP_New
## No     No                      2 2
##        Yes                     7 8
## Yes    No                      0 3
##        Yes                     3 5
```

pivot table without target column

```r
Object2 <- ftable(Heart_disease30[ ,-1])
Object2
```

```
##        chest_pain_type  0  1
## BP_New
## No                      2  5
## Yes                    10 13
```

2a. Computing Bayes Conditional Probabilities

```r
p1 <- Object1[3,1]/Object2[1,1] #Target=yes , BP_New=No & chest_pain_type=0
p2 <- Object1[3,2]/Object2[1,2] #Target=yes , BP_New=No & Chest_pain_type=1
p3 <- Object1[4,1]/Object2[2,1] #Target=yes , BP_New=Yes & Chest_pain_type=0
p4 <- Object1[4,2]/Object2[2,2] #Target=yes , BP_New=Yes & Chest_pain_type=1
```

Conditional probabilities values

```r
cat("The conditional probability of having heart disease with no BP and No
chest Pain:",p1,"\n")
```

```
## The conditional probability of having heart disease with no BP and No
chest Pain: 0
```

```r
cat("The conditional probability of having heart disease with high BP and
chest Pain:", p2,"\n")
```

```
## The conditional probability of having heart disease with high BP and chest
Pain: 0.6
```

```r
cat("The conditional probability of having heart disease with BP and No chest
Pain:", p3 ,"\n")
```

```
## The conditional probability of having heart disease with BP and No chest
Pain: 0.3
```

```r
cat("The conditional probability of having heart disease with BP and chest
Pain:", p4 ,"\n")
```

```
## The conditional probability of having heart disease with BP and chest
Pain: 0.3846154
```

2b. Classification using a cutoff of 0.5

```r
Probability_Target <- rep(0, 30)

for (i in 1:30) {
  BP_New <- Heart_disease30$BP_New[i]  # Getting BP_New value for row i
  chest_pain <- Heart_disease30$chest_pain_type[i]  # Getting chest_pain for
row i

  # Matching the probability from pre-computed values (p1, p2, p3, p4)
  if (BP_New == "No" & chest_pain == 0) {
```

```
    Probability_Target[i] <- p1
  } else if (BP_New == "Yes" & chest_pain == 1) {
    Probability_Target[i] <- p2
  } else if (BP_New == "Yes" & chest_pain == 0) {
    Probability_Target[i] <- p3
  } else if (BP_New == "No" & chest_pain == 1) {
    Probability_Target[i] <- p4
  }
}

Heart_disease30$Probability_Target <- Probability_Target
Heart_disease30$Pred_Probability <- ifelse(Heart_disease30$Probability_Target
> 0.5, "Yes", "No")
Heart_disease30
```

```
##    Target BP_New chest_pain_type Probability_Target Pred_Probability
## 1      No    Yes               0          0.3000000               No
## 2     Yes    Yes               1          0.6000000              Yes
## 3     Yes    Yes               1          0.6000000              Yes
## 4     Yes     No               1          0.3846154               No
## 5      No     No               0          0.0000000               No
## 6      No    Yes               0          0.3000000               No
## 7      No    Yes               1          0.6000000              Yes
## 8     Yes     No               1          0.3846154               No
## 9      No    Yes               1          0.6000000              Yes
## 10    Yes    Yes               1          0.6000000              Yes
## 11     No    Yes               0          0.3000000               No
## 12     No    Yes               1          0.6000000              Yes
## 13    Yes    Yes               1          0.6000000              Yes
## 14     No     No               0          0.0000000               No
## 15     No    Yes               0          0.3000000               No
## 16     No     No               1          0.3846154               No
## 17    Yes     No               1          0.3846154               No
## 18     No    Yes               0          0.3000000               No
## 19    Yes    Yes               0          0.3000000               No
## 20     No    Yes               0          0.3000000               No
## 21     No    Yes               0          0.3000000               No
## 22    Yes    Yes               1          0.6000000              Yes
## 23    Yes    Yes               0          0.3000000               No
## 24     No    Yes               1          0.6000000              Yes
## 25    Yes    Yes               0          0.3000000               No
## 26     No    Yes               1          0.6000000              Yes
## 27     No    Yes               1          0.6000000              Yes
## 28     No     No               1          0.3846154               No
## 29     No    Yes               1          0.6000000              Yes
## 30     No    Yes               1          0.6000000              Yes
```

2c. Manual Calculation of Naive Bayes Probability

```
# Compute total instances
total_count <- nrow(Heart_disease30)

# Compute P(Target = Yes)
p_target_yes <- sum(Heart_disease30$Target == "Yes") / total_count

# Compute P(BP_New = Yes, chest_pain_type = 1 | Target = Yes)
p_given_yes <- sum(Heart_disease30$BP_New == "Yes" &
Heart_disease30$chest_pain_type == 1 & Heart_disease30$Target == "Yes") /
sum(Heart_disease30$Target == "Yes")

# Compute P(BP_New = Yes, chest_pain_type = 1)
p_denominator <- sum(Heart_disease30$BP_New == "Yes" &
Heart_disease30$chest_pain_type == 1) / total_count

# Compute the final probability using Bayes' Theorem
p_yes_given_bp_chest <- (p_given_yes * p_target_yes) / p_denominator

# Print result
cat("Manually compute the naive  Bayes conditional probability of an injury
given that BP_New is Yes and chest_pain_type is 1 = ",p_yes_given_bp_chest,
"\n")

## Manually compute the naive  Bayes conditional probability of an injury
given that BP_New is Yes and chest_pain_type is 1 =  0.3846154
```

Q3: Full Data set Analysis - Splitting into Training and Validation Sets

```
set.seed(123)

train.index <- sample(row.names(df_unique),0.6 * nrow(df_unique))
valid.index <- setdiff(row.names(df_unique),train.index)

train.df <- df_unique[train.index, ]
valid.df <- df_unique[valid.index, ]

nrow(train.df)

## [1] 181

nrow(valid.df)

## [1] 121

train.df<-train.df[,-9]
valid.df<-valid.df[,-9]
```

Running Naive Bayes Classifier

```
nb_model <- naiveBayes(Target ~ chest_pain_type + BP_New, data = train.df,1)
valid_pred <- predict(nb_model, valid.df)
```

```r
# Convert predictions and actual target variable to factors with the same
levels
valid.df$Target <- factor(valid.df$Target)
valid_pred <- factor(valid_pred, levels = levels(valid.df$Target))

# Compute confusion matrix
conf_matrix <- confusionMatrix(valid_pred, valid.df$Target,positive = "Yes")

conf_matrix

## Confusion Matrix and Statistics
##
##           Reference
## Prediction No Yes
##        No  96  25
##        Yes  0   0
##
##                Accuracy : 0.7934
##                  95% CI : (0.7103, 0.8616)
##     No Information Rate : 0.7934
##     P-Value [Acc > NIR] : 0.5533
##
##                   Kappa : 0
##
##  Mcnemar's Test P-Value : 1.587e-06
##
##             Sensitivity : 0.0000
##             Specificity : 1.0000
##          Pos Pred Value :    NaN
##          Neg Pred Value : 0.7934
##              Prevalence : 0.2066
##          Detection Rate : 0.0000
##    Detection Prevalence : 0.0000
##       Balanced Accuracy : 0.5000
##
##        'Positive' Class : Yes
##
```

As per the confusion matrix outcomes, the model has achieved an accuracy of 79.34%, but this is misleading due to class imbalance. The Naive Bayes model correctly identifies all non-heart disease cases, a Kappa value of 0. This shows that the model is highly biased towards the majority class and is not suitable for predictions.

Thus, including all attributes improves both Kappa, leading to a better-performing and balanced model.

Running Naive Bayes Classifier including all the attributes. (Validation set)

```
nb_model_v <- naiveBayes(Target ~., data = train.df)
train_pred_v <- predict(nb_model_v, valid.df)

# Convert predictions and actual target variable to factors with the same
levels
valid.df$Target <- factor(valid.df$Target)
train_pred_v <- factor(train_pred_v, levels = levels(valid.df$Target))

# Compute confusion matrix
conf_matrix1_v <- confusionMatrix(train_pred_v, valid.df$Target, positive =
"Yes")
conf_matrix1_v

## Confusion Matrix and Statistics
##
##           Reference
## Prediction No Yes
##        No  90   8
##        Yes  6  17
##
##                Accuracy : 0.8843
##                  95% CI : (0.8135, 0.9353)
##     No Information Rate : 0.7934
##     P-Value [Acc > NIR] : 0.006446
##
##                   Kappa : 0.6363
##
##  Mcnemar's Test P-Value : 0.789268
##
##             Sensitivity : 0.6800
##             Specificity : 0.9375
##          Pos Pred Value : 0.7391
##          Neg Pred Value : 0.9184
##              Prevalence : 0.2066
##          Detection Rate : 0.1405
##    Detection Prevalence : 0.1901
##       Balanced Accuracy : 0.8088
##
##        'Positive' Class : Yes
##
```

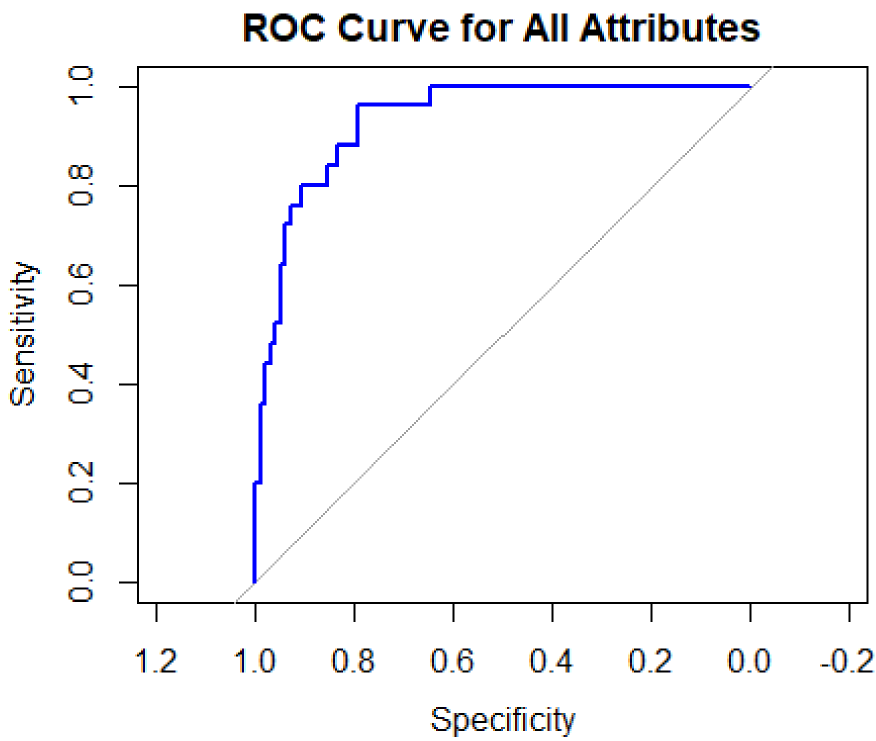
ROC Curve

```
pred_probs_full <- predict(nb_model_v, valid.df, type = "raw")
prob_yes_full <- pred_probs_full[, "Yes"]
roc_full <- roc(valid.df$Target, prob_yes_full)

## Setting levels: control = No, case = Yes

## Setting direction: controls < cases
```

```r
plot(roc_full, col = "blue", main = "ROC Curve for All Attributes")
```




```r
auc_value <- auc(roc_full)
auc_value

## Area under the curve: 0.9325
```

Plotted the ROC, to analyse the performance when all the attributes are included. AUC value is 0.9329. The model is performing good when all the attributes are considered.