Assignment 6_A

33140 (Sahil Naphade)

23/04/2020

Roll no. 33140

Batch: L9

P.S.: Application of Linear regression and Naive bayes on Heart disease dataset to predict the fate (prob. of heart disease)

Load the libraries and set the working directory

```
# Load the libraries

# install.packages(c("caret", "e1071"))
library('e1071')
library('caret')
```

```
## Loading required package: lattice
```

```
## Loading required package: ggplot2
```

```
# Set working directory
setwd("G:/College/SL6/Assignment6/")
```

Read the data and clean

```
# Read the CSV file and analyse
hdata <- read.csv("../../Sl-VI DataSets/HeartDisease/Cleavland.csv",header=TRUE,sep=",")
names(hdata)
```

```
##  [1] "X63.0"  "X1.0"   "X1.0.1" "X145.0" "X233.0" "X1.0.2" "X2.0"   "X150.0"
##  [9] "X0.0"   "X2.3"   "X3.0"   "X0.0.1" "X6.0"   "X0"
```

```
str(hdata)
```

```
## 'data.frame':    302 obs. of  14 variables:
##  $ X63.0 : num  67 67 37 41 56 62 57 63 53 57 ...
##  $ X1.0  : num  1 1 1 0 1 0 0 1 1 1 ...
##  $ X1.0.1: num  4 4 3 2 2 4 4 4 4 4 ...
##  $ X145.0: num  160 120 130 130 120 140 120 130 140 140 ...
##  $ X233.0: num  286 229 250 204 236 268 354 254 203 192 ...
##  $ X1.0.2: num  0 0 0 0 0 0 0 0 1 0 ...
##  $ X2.0  : num  2 2 0 2 0 2 0 2 2 0 ...
##  $ X150.0: num  108 129 187 172 178 160 163 147 155 148 ...
##  $ X0.0  : num  1 1 0 0 0 0 1 0 1 0 ...
##  $ X2.3  : num  1.5 2.6 3.5 1.4 0.8 3.6 0.6 1.4 3.1 0.4 ...
##  $ X3.0  : num  2 2 3 1 1 3 1 2 3 2 ...
```

```
##  $ X0.0.1: Factor w/ 5 levels "?","0.0","1.0",..: 5 4 2 2 2 4 2 3 2 2 ...
##  $ X6.0  : Factor w/ 4 levels "?","3.0","6.0",..: 2 4 2 2 2 2 2 4 4 3 ...
##  $ X0    : int  2 1 0 0 0 3 0 2 1 0 ...
```

```
dim(hdata)
```

```
## [1] 302  14
```

```
# Change the headers
names(hdata)[1] <- "age"
names(hdata)[2] <- "sex"
names(hdata)[3] <- "cp"
names(hdata)[4] <- "trestbps"
names(hdata)[5] <- "chol"
names(hdata)[6] <- "fbs"
names(hdata)[7] <- "restecg"
names(hdata)[8] <- "thalach"
names(hdata)[9] <- "exang"
names(hdata)[10] <- "oldpeak"
names(hdata)[11] <- "slope"
names(hdata)[12] <- "ca"
names(hdata)[13] <- "thal"
names(hdata)[14] <- "num"

hdata$ca
```

```
##   [1] 3.0 2.0 0.0 0.0 0.0 2.0 0.0 1.0 0.0 0.0 0.0 1.0 0.0 0.0 0.0 0.0 0.0 0.0
##  [19] 0.0 0.0 0.0 0.0 2.0 2.0 0.0 0.0 0.0 0.0 0.0 2.0 2.0 0.0 0.0 0.0 0.0 0.0
##  [37] 1.0 1.0 0.0 3.0 0.0 2.0 0.0 0.0 1.0 0.0 0.0 1.0 0.0 1.0 0.0 1.0 0.0 1.0
##  [55] 1.0 1.0 0.0 1.0 1.0 0.0 0.0 3.0 0.0 1.0 2.0 0.0 0.0 0.0 0.0 0.0 2.0 2.0
##  [73] 2.0 1.0 0.0 1.0 1.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0
##  [91] 3.0 3.0 0.0 0.0 1.0 1.0 2.0 1.0 0.0 0.0 0.0 1.0 1.0 3.0 0.0 1.0 1.0 1.0
## [109] 0.0 0.0 1.0 0.0 0.0 1.0 0.0 0.0 0.0 3.0 1.0 2.0 3.0 0.0 0.0 1.0 0.0 2.0
## [127] 1.0 0.0 0.0 0.0 1.0 0.0 0.0 0.0 0.0 0.0 1.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0
## [145] 0.0 3.0 0.0 0.0 1.0 0.0 0.0 0.0 1.0 1.0 3.0 0.0 2.0 2.0 1.0 0.0 3.0 0.0
## [163] 0.0 2.0 0.0 ?   1.0 0.0 0.0 1.0 0.0 0.0 0.0 2.0 1.0 3.0 1.0 1.0 3.0 0.0
## [181] 2.0 2.0 0.0 0.0 2.0 0.0 3.0 1.0 3.0 0.0 3.0 ?   3.0 0.0 2.0 1.0 0.0 0.0
## [199] 0.0 0.0 0.0 1.0 0.0 0.0 3.0 2.0 0.0 0.0 0.0 0.0 0.0 0.0 2.0 1.0 0.0 0.0
## [217] 0.0 2.0 0.0 0.0 0.0 0.0 2.0 2.0 0.0 0.0 1.0 1.0 1.0 0.0 0.0 3.0 1.0 1.0
## [235] 2.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 2.0 0.0 0.0 1.0 1.0 2.0 0.0 0.0 1.0 1.0
## [253] 0.0 0.0 0.0 2.0 0.0 0.0 0.0 1.0 2.0 0.0 0.0 1.0 0.0 0.0 1.0 0.0 0.0 1.0
## [271] 0.0 2.0 0.0 2.0 0.0 1.0 0.0 1.0 0.0 1.0 0.0 1.0 0.0 1.0 3.0 2.0 ?   0.0
## [289] 0.0 0.0 0.0 0.0 2.0 0.0 0.0 2.0 0.0 0.0 2.0 1.0 1.0 ?
## Levels: ? 0.0 1.0 2.0 3.0
```

```
levels(hdata$ca)[levels(hdata$ca) == "?"]<-"0.0"

hdata$ca[hdata$ca == 1.0]
```

```
## factor(0)
## Levels: 0.0 1.0 2.0 3.0
```

```
typeof(hdata$ca)
```

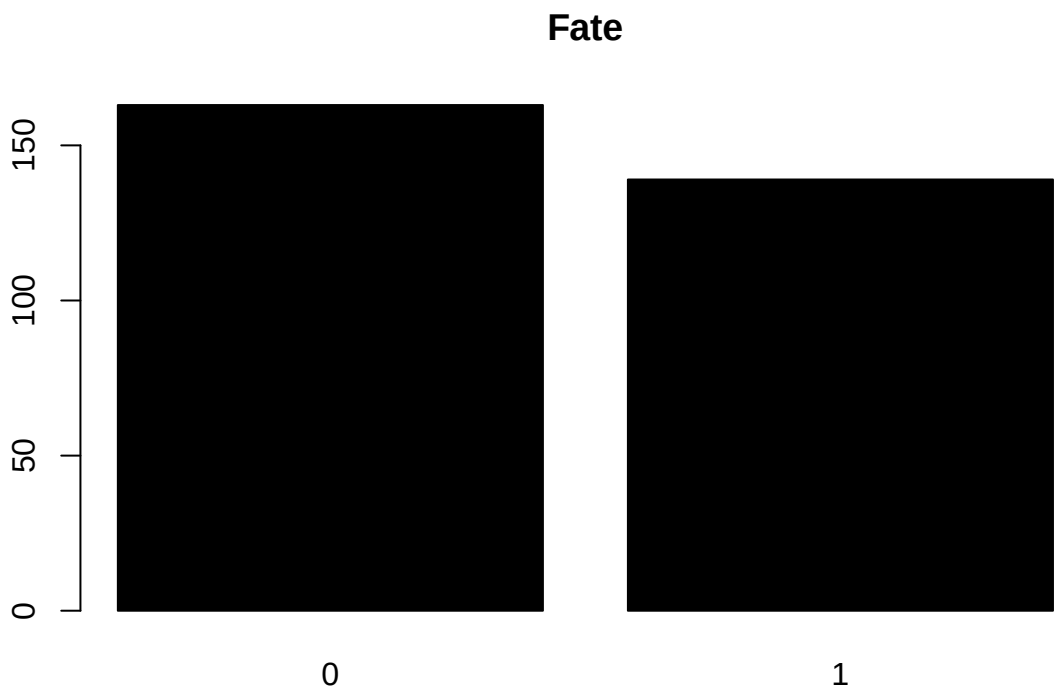
```
## [1] "integer"
```

```
str(hdata)
```

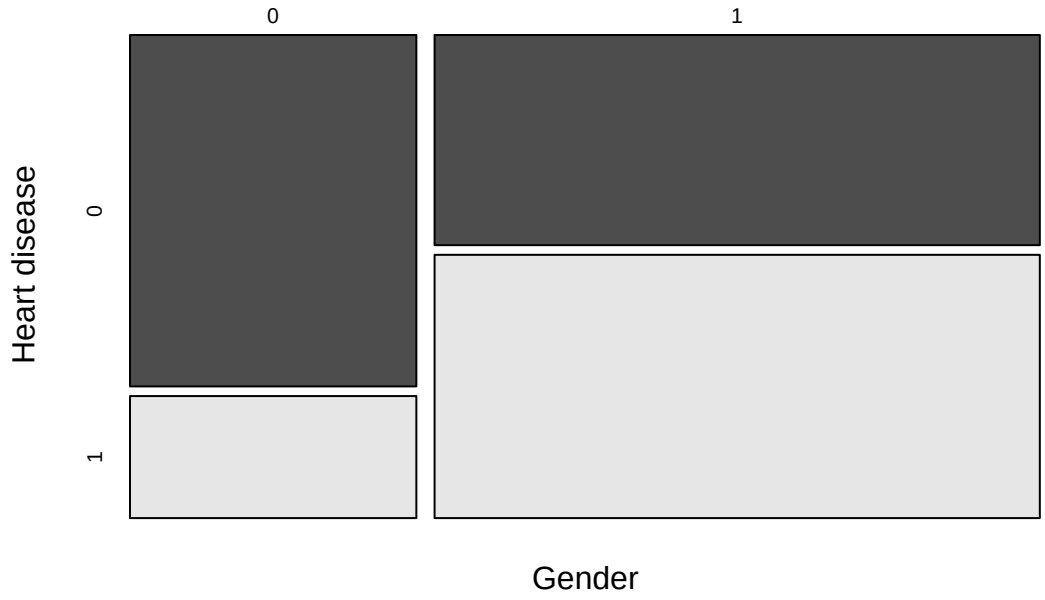
```
## 'data.frame':    302 obs. of  14 variables:
##  $ age     : num  67 67 37 41 56 62 57 63 53 57 ...
##  $ sex     : num  1 1 1 0 1 0 0 1 1 1 ...
##  $ cp      : num  4 4 3 2 2 4 4 4 4 4 ...
##  $ trestbps: num  160 120 130 130 120 140 120 130 140 140 ...
##  $ chol    : num  286 229 250 204 236 268 354 254 203 192 ...
##  $ fbs     : num  0 0 0 0 0 0 0 0 1 0 ...
##  $ restecg : num  2 2 0 2 0 2 0 2 2 0 ...
##  $ thalach : num  108 129 187 172 178 160 163 147 155 148 ...
##  $ exang   : num  1 1 0 0 0 0 1 0 1 0 ...
##  $ oldpeak : num  1.5 2.6 3.5 1.4 0.8 3.6 0.6 1.4 3.1 0.4 ...
##  $ slope   : num  2 2 3 1 1 3 1 2 3 2 ...
##  $ ca      : Factor w/ 4 levels "0.0","1.0","2.0",..: 4 3 1 1 1 3 1 2 1 1 ...
##  $ thal    : Factor w/ 4 levels "?","3.0","6.0",..: 2 4 2 2 2 2 2 4 4 3 ...
##  $ num     : int  2 1 0 0 0 3 0 2 1 0 ...
```

Visualize

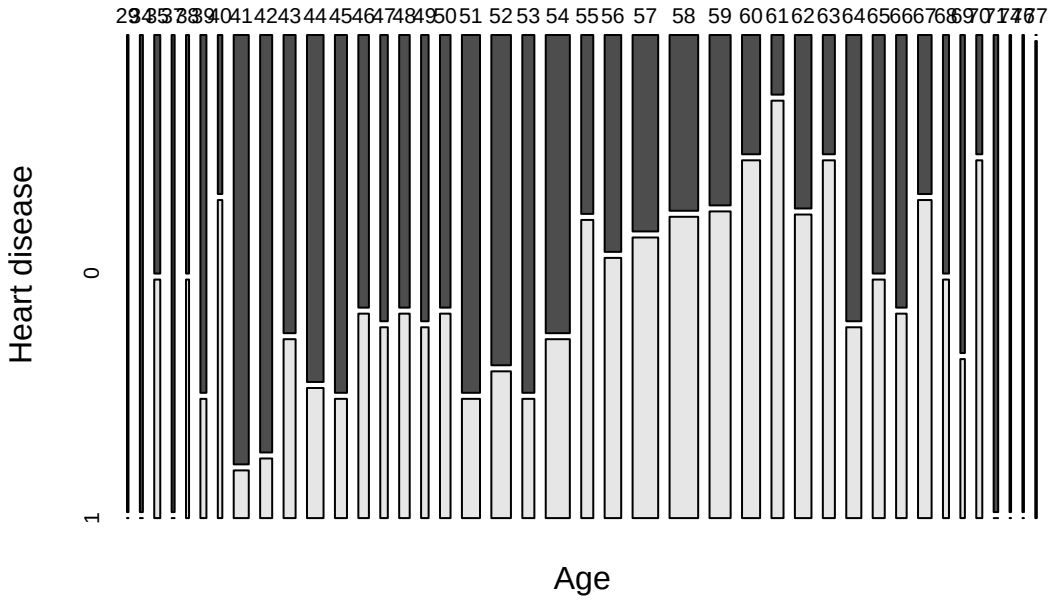
```
# Plotting Fate vs number of records
hdata$num[hdata$num >= 1] <- 1 # Edit the fate to 0 and 1
barplot(table(hdata$num), main="Fate", col="black")
```




```
# Plot Fate vs gender
mosaicplot(hdata$sex ~ hdata$num,main="Fate by Gender",
           shade=FALSE,color=TRUE,xlab="Gender", ylab="Heart disease")
```




```
# Plot Fate vs Age
mosaicplot(hdata$age ~ hdata$num,main="Fate by Age",
           shade=FALSE,color=TRUE,xlab="Age", ylab="Heart disease")
```



Application of Linear regression

```
# Most important step, change the values of NA
levels(hdata$thal)[levels(hdata$thal)=="?"]<-"3.0"

# removal of additional NA
hdata$thal
```

```
##   [1] 3.0 7.0 3.0 3.0 3.0 3.0 3.0 7.0 7.0 6.0 3.0 6.0 7.0 7.0 3.0 7.0 3.0 3.0
##  [19] 3.0 3.0 3.0 3.0 7.0 7.0 3.0 3.0 3.0 3.0 7.0 3.0 7.0 3.0 7.0 3.0 3.0 7.0
##  [37] 6.0 7.0 3.0 7.0 7.0 3.0 3.0 3.0 7.0 3.0 7.0 3.0 3.0 3.0 7.0 3.0 3.0 7.0
##  [55] 7.0 7.0 7.0 3.0 3.0 7.0 3.0 7.0 3.0 7.0 7.0 3.0 7.0 7.0 3.0 3.0 7.0 7.0
##  [73] 6.0 3.0 3.0 7.0 3.0 3.0 7.0 3.0 3.0 3.0 7.0 3.0 3.0 3.0 3.0 3.0 3.0 3.0
##  [91] 7.0 7.0 3.0 3.0 7.0 7.0 7.0 3.0 3.0 3.0 3.0 3.0 3.0 7.0 7.0 7.0 7.0 7.0
## [109] 7.0 7.0 3.0 6.0 7.0 7.0 6.0 3.0 3.0 7.0 7.0 7.0 7.0 3.0 7.0 3.0 3.0 7.0
## [127] 7.0 3.0 3.0 7.0 7.0 3.0 3.0 3.0 3.0 7.0 7.0 7.0 3.0 3.0 7.0 3.0 7.0 7.0
## [145] 3.0 7.0 3.0 3.0 3.0 7.0 3.0 7.0 7.0 3.0 3.0 7.0 7.0 7.0 7.0 7.0 3.0 3.0
## [163] 3.0 3.0 7.0 3.0 3.0 7.0 3.0 7.0 7.0 3.0 3.0 6.0 7.0 7.0 6.0 3.0 3.0 7.0
## [181] 7.0 3.0 7.0 3.0 3.0 7.0 6.0 7.0 7.0 3.0 7.0 7.0 3.0 3.0 3.0 3.0 3.0 3.0
## [199] 3.0 3.0 3.0 7.0 7.0 7.0 7.0 7.0 7.0 3.0 3.0 3.0 7.0 3.0 7.0 3.0 7.0 3.0
## [217] 3.0 3.0 3.0 3.0 3.0 3.0 7.0 3.0 3.0 3.0 3.0 3.0 3.0 3.0 3.0 3.0 3.0 3.0
## [235] 3.0 7.0 7.0 3.0 3.0 3.0 3.0 3.0 3.0 3.0 3.0 7.0 3.0 7.0 3.0 6.0 7.0 7.0
## [253] 3.0 3.0 3.0 3.0 3.0 3.0 7.0 3.0 3.0 3.0 3.0 3.0 6.0 3.0 6.0 7.0 3.0 7.0
## [271] 6.0 7.0 3.0 3.0 7.0 3.0 3.0 3.0 3.0 7.0 3.0 7.0 3.0 7.0 6.0 6.0 7.0 7.0
## [289] 3.0 7.0 3.0 6.0 7.0 3.0 3.0 6.0 7.0 7.0 7.0 7.0 3.0 3.0
## Levels: 3.0 6.0 7.0
```

```r
table(hdata$thal)
```

```
##
## 3.0 6.0 7.0
## 168  17 117
```

```r
# hdata$thal[is.na(hdata$thal)]<-'3.0'


table(hdata$ca)
```

```
##
## 0.0 1.0 2.0 3.0
## 179  65  38  20
```

```r
# import library caTools
library(caTools)
library(e1071)
library(caret)

n<- sapply(hdata[, c(1)], mean) # get the average values
set.seed(123) # generate a pseudo-random number

# Subset
v3 <- hdata[c(11:14),c(2,7:9)]
v3
```

```
##    sex restecg thalach exang
## 11   0       2     153     0
## 12   1       2     142     1
## 13   1       0     173     0
## 14   1       0     162     0
```

```r
m<- sapply(v3,max)
m
```

```
##     sex restecg thalach   exang
##       1       2     173       1
```

```r
#  Pseudo-random
set.seed(121)

# Divide the dataset into 2/3 for training, and 1/3 for testing
split = sample.split(hdata$num, SplitRatio = 2/3)
train_hdata = subset(hdata, split == TRUE)
test_hdata = subset(hdata, split == FALSE)

# Apply linear regression for Fate vs age
regressor=lm(formula = num~age, data=train_hdata)

# View(regressor)
regressor
```

```
##
## Call:
## lm(formula = num ~ age, data = train_hdata)
##
```

```
## Coefficients:
## (Intercept)          age
##    -0.33038      0.01453
```

```
# Apply regression on test data
hd_age_predict = predict(regressor, newdata=test_hdata)
hd_age_predict
```

```
##         2         4         7         8        10        18        19        20
## 0.6430055 0.2652722 0.4977234 0.5848927 0.4977234 0.3669696 0.3814978 0.5994209
##        25        29        30        31        33        34        42        43
## 0.3960260 0.2507440 0.6720619 0.5413081 0.5267799 0.3088568 0.7011183 0.5267799
##        45        47        52        59        61        65        68        70
## 0.5122517 0.3960260 0.3088568 0.4105542 0.3379132 0.5413081 0.5267799 0.6139491
##        73        75        78        79        83        84        85        86
## 0.6139491 0.6139491 0.3669696 0.5122517 0.6575337 0.4250824 0.3088568 0.3524414
##       108       109       110       111       115       118       122       129
## 0.5558363 0.2362158 0.5558363 0.4831952 0.2652722 0.5848927 0.4105542 0.5703645
##       134       136       137       139       141       144       147       149
## 0.2943286 0.6865901 0.5703645 0.4105542 0.5267799 0.5122517 0.2652722 0.5413081
##       155       156       158       168       169       173       175       176
## 0.6865901 0.4105542 0.5413081 0.1781030 0.3233850 0.5703645 0.4977234 0.4250824
##       179       184       185       186       187       192       194       197
## 0.4396106 0.5413081 0.5848927 0.2798004 0.6284773 0.2943286 0.6575337 0.3233850
##       201       204       206       207       214       217       219       220
## 0.5994209 0.2943286 0.5122517 0.3960260 0.4250824 0.3379132 0.5267799 0.2652722
##       225       226       229       230       232       236       237       240
## 0.1635748 0.3524414 0.6284773 0.4250824 0.3814978 0.4831952 0.3379132 0.2652722
##       241       247       248       252       255       258       262       265
## 0.2652722 0.3524414 0.4250824 0.5994209 0.2798004 0.6865901 0.5413081 0.2798004
##       267       273       276       282       284       285       288       292
## 0.5267799 0.7011183 0.6284773 0.4686670 0.5558363 0.5122517 0.4831952 0.3088568
##       293       296       301       302
## 0.5848927 0.5267799 0.4977234 0.2216876
```

```
# Round the values of fate in prediction
round_age=hd_age_predict
r=round(round_age)
View(r)
r
```

```
##   2   4   7   8  10  18  19  20  25  29  30  31  33  34  42  43  45  47  52  59
##   1   0   0   1   0   0   0   1   0   0   1   1   1   0   1   1   1   0   0   0
##  61  65  68  70  73  75  78  79  83  84  85  86 108 109 110 111 115 118 122 129
##   0   1   1   1   1   1   0   1   1   0   0   0   1   0   1   0   0   1   0   1
## 134 136 137 139 141 144 147 149 155 156 158 168 169 173 175 176 179 184 185 186
##   0   1   1   0   1   1   0   1   1   0   1   0   0   1   0   0   0   1   1   0
## 187 192 194 197 201 204 206 207 214 217 219 220 225 226 229 230 232 236 237 240
##   1   0   1   0   1   0   1   0   0   0   1   0   0   0   1   0   0   0   0   0
## 241 247 248 252 255 258 262 265 267 273 276 282 284 285 288 292 293 296 301 302
##   0   0   0   1   0   1   1   0   1   1   1   0   1   1   0   0   1   1   0   0
```

```
table(r,test_hdata$num)
```

```
##
## r    0  1
```

```
##   0 34 20
##   1 20 26
```

```
typeof(r)
```

```
## [1] "double"
```

```
levels(r)
```

```
## NULL
```

```
levels(test_hdata$num)
```

```
## NULL
```

```
str(r)
```

```
##  Named num [1:100] 1 0 0 1 0 0 0 1 0 0 ...
##  - attr(*, "names")= chr [1:100] "2" "4" "7" "8" ...
```

```
r1 = as.data.frame(r)
str(r1)
```

```
## 'data.frame':    100 obs. of  1 variable:
##  $ r: num  1 0 0 1 0 0 0 1 0 0 ...
```

Display the accuracy of Linear regression

```
lm_accuracy = confusionMatrix(as.factor(r1$r),as.factor(test_hdata$num))
lm_accuracy
```

```
## Confusion Matrix and Statistics
##
##           Reference
## Prediction  0  1
##          0 34 20
##          1 20 26
##
##                Accuracy : 0.6
##                  95% CI : (0.4972, 0.6967)
##     No Information Rate : 0.54
##     P-Value [Acc > NIR] : 0.1347
##
##                   Kappa : 0.1948
##
##  Mcnemar's Test P-Value : 1.0000
##
##             Sensitivity : 0.6296
##             Specificity : 0.5652
##          Pos Pred Value : 0.6296
##          Neg Pred Value : 0.5652
##              Prevalence : 0.5400
##          Detection Rate : 0.3400
##    Detection Prevalence : 0.5400
##       Balanced Accuracy : 0.5974
##
##        'Positive' Class : 0
##
```

```r
# APPLICATION OF NB

# 1. converting all values to factor
hdata$age <- factor(hdata$age)
hdata$sex <- factor(hdata$sex)
hdata$cp <- factor(hdata$cp)
hdata$trestbps <- factor(hdata$trestbps)
hdata$chol <- factor(hdata$chol)
hdata$fbs <- factor(hdata$fbs)
hdata$restecg <- factor(hdata$restecg)
hdata$thalach <- factor(hdata$thalach)
hdata$exang <- factor(hdata$exang)
hdata$oldpeak <- factor(hdata$oldpeak)
hdata$slope <- factor(hdata$slope)
hdata$ca <- factor(hdata$ca)
hdata$thal <- factor(hdata$thal)
hdata$num <- factor(hdata$num)
```

```r
# 2. Divide the factored dataset into 2/3 for training, and 1/3 for testing
split = sample.split(hdata$num, SplitRatio = 2/3)
train_hdata = subset(hdata, split == TRUE)
test_hdata = subset(hdata, split == FALSE)

# 3. Apply Naive Bayes on Dataset
nb_model <- naiveBayes(num ~ age+sex+cp+trestbps+chol+fbs+restecg+thalach+exang+oldpeak+slope+ca+thal,da
str(nb_model)
```

```
## List of 5
##  $ apriori  : 'table' int [1:2(1d)] 109 93
##   ..- attr(*, "dimnames")=List of 1
##   .. ..$ Y: chr [1:2] "0" "1"
##  $ tables   :List of 13
##   ..$ age     : 'table' num [1:2, 1:41] 0.00917 0 0 0 0.00917 ...
##   .. ..- attr(*, "dimnames")=List of 2
##   .. .. ..$ Y  : chr [1:2] "0" "1"
##   .. .. ..$ age: chr [1:41] "29" "34" "35" "37" ...
##   ..$ sex     : 'table' num [1:2, 1:2] 0.413 0.161 0.587 0.839
##   .. ..- attr(*, "dimnames")=List of 2
##   .. .. ..$ Y  : chr [1:2] "0" "1"
##   .. .. ..$ sex: chr [1:2] "0" "1"
##   ..$ cp      : 'table' num [1:2, 1:4] 0.0917 0.0538 0.2477 0.086 0.4495 ...
##   .. ..- attr(*, "dimnames")=List of 2
##   .. .. ..$ Y : chr [1:2] "0" "1"
##   .. .. ..$ cp: chr [1:4] "1" "2" "3" "4"
##   ..$ trestbps: 'table' num [1:2, 1:50] 0.00917 0 0.01835 0.02151 0.00917 ...
##   .. ..- attr(*, "dimnames")=List of 2
##   .. .. ..$ Y       : chr [1:2] "0" "1"
##   .. .. ..$ trestbps: chr [1:50] "94" "100" "101" "102" ...
##   ..$ chol    : 'table' num [1:2, 1:152] 0 0 0 0 0 ...
##   .. ..- attr(*, "dimnames")=List of 2
##   .. .. ..$ Y   : chr [1:2] "0" "1"
##   .. .. ..$ chol: chr [1:152] "126" "131" "141" "149" ...
##   ..$ fbs     : 'table' num [1:2, 1:2] 0.844 0.839 0.156 0.161
##   .. ..- attr(*, "dimnames")=List of 2
```

```
##   .. .. ..$ Y  : chr [1:2] "0" "1"
##   .. .. ..$ fbs: chr [1:2] "0" "1"
##   ..$ restecg : 'table' num [1:2, 1:3] 0.55046 0.37634 0.00917 0.02151 0.44037 ...
##   .. ..- attr(*, "dimnames")=List of 2
##   .. .. ..$ Y      : chr [1:2] "0" "1"
##   .. .. ..$ restecg: chr [1:3] "0" "1" "2"
##   ..$ thalach : 'table' num [1:2, 1:91] 0 0 0 0.0108 0 ...
##   .. ..- attr(*, "dimnames")=List of 2
##   .. .. ..$ Y      : chr [1:2] "0" "1"
##   .. .. ..$ thalach: chr [1:91] "71" "88" "90" "95" ...
##   ..$ exang   : 'table' num [1:2, 1:2] 0.862 0.473 0.138 0.527
##   .. ..- attr(*, "dimnames")=List of 2
##   .. .. ..$ Y    : chr [1:2] "0" "1"
##   .. .. ..$ exang: chr [1:2] "0" "1"
##   ..$ oldpeak : 'table' num [1:2, 1:40] 0.422 0.1828 0.0367 0.0323 0.055 ...
##   .. ..- attr(*, "dimnames")=List of 2
##   .. .. ..$ Y      : chr [1:2] "0" "1"
##   .. .. ..$ oldpeak: chr [1:40] "0" "0.1" "0.2" "0.3" ...
##   ..$ slope   : 'table' num [1:2, 1:3] 0.6055 0.2903 0.3303 0.6452 0.0642 ...
##   .. ..- attr(*, "dimnames")=List of 2
##   .. .. ..$ Y    : chr [1:2] "0" "1"
##   .. .. ..$ slope: chr [1:3] "1" "2" "3"
##   ..$ ca      : 'table' num [1:2, 1:4] 0.8165 0.3441 0.1284 0.3011 0.0275 ...
##   .. ..- attr(*, "dimnames")=List of 2
##   .. .. ..$ Y : chr [1:2] "0" "1"
##   .. .. ..$ ca: chr [1:4] "0.0" "1.0" "2.0" "3.0"
##   ..$ thal    : 'table' num [1:2, 1:3] 0.8165 0.2688 0.0275 0.086 0.156 ...
##   .. ..- attr(*, "dimnames")=List of 2
##   .. .. ..$ Y   : chr [1:2] "0" "1"
##   .. .. ..$ thal: chr [1:3] "3.0" "6.0" "7.0"
##  $ levels   : chr [1:2] "0" "1"
##  $ isnumeric: Named logi [1:13] FALSE FALSE FALSE FALSE FALSE FALSE ...
##   ..- attr(*, "names")= chr [1:13] "age" "sex" "cp" "trestbps" ...
##  $ call     : language naiveBayes.default(x = X, y = Y, laplace = laplace)
##  - attr(*, "class")= chr "naiveBayes"
```

```r
# 4. Predict
pred_nb <- predict(nb_model,newdata = test_hdata, type = "class")

# 5. convert to table
table(pred_nb,test_hdata[,14])
```

```
##
## pred_nb  0  1
##       0 42 11
##       1 12 35
```

```r
# 6. Prepare confusion matrix
nb_accur <- confusionMatrix(as.factor(test_hdata$num),as.factor(pred_nb))

# 7. Display
nb_accur
```

```
## Confusion Matrix and Statistics
##
##           Reference
```

```
## Prediction  0  1
##          0 42 12
##          1 11 35
##
##                Accuracy : 0.77
##                  95% CI : (0.6751, 0.8483)
##     No Information Rate : 0.53
##     P-Value [Acc > NIR] : 6.211e-07
##
##                   Kappa : 0.5378
##
##  Mcnemar's Test P-Value : 1
##
##             Sensitivity : 0.7925
##             Specificity : 0.7447
##          Pos Pred Value : 0.7778
##          Neg Pred Value : 0.7609
##              Prevalence : 0.5300
##          Detection Rate : 0.4200
##    Detection Prevalence : 0.5400
##       Balanced Accuracy : 0.7686
##
##        'Positive' Class : 0
##
```