Machine Learning
Bayes Classifier Implementation from Scratch

(i.e. No library function for core implementation)

Prepared By : Kisha Taylor

```
## Objective : Using Naive Bayes classifier to claasify data points
#              into classes ("Yes/No") for bank default based on
#              specific client attibutes.
#              N.B. Dependent variable is categorical.
#Datset found here -> https://archive.ics.uci.edu/ml/datasets/bank+marketing

###########################################################################
########                      BANK DATA                         #########
###########################################################################

#Bank Info dataset most interesting features are :
# (1) Age
# (2) nr.employed: number of employees - quarterly indicator (numeric)
# (3) previous: number of contacts performed before this campaign and for this client (numeric)
# (4) emp.var.rate: employment variation rate - quarterly indicator (numeric)

# (5) Subscription status "y": "Has subscribed to term deposit?"
setwd("C:/Users/Kisha/Documents")
Bkdata <- read.csv("bank-additional.csv",header=TRUE,sep=";")

head(Bkdata)
```

```
##   age         job marital         education default housing    loan
## 1  30 blue-collar married          basic.9y      no     yes      no
## 2  39    services  single       high.school      no      no      no
## 3  25    services married       high.school      no     yes      no
## 4  38    services married          basic.9y      no unknown unknown
## 5  47      admin. married university.degree      no     yes      no
## 6  32    services  single university.degree      no      no      no
##     contact month day_of_week duration campaign pdays previous    poutcome
## 1  cellular   may         fri      487        2   999        0 nonexistent
## 2 telephone   may         fri      346        4   999        0 nonexistent
## 3 telephone   jun         wed      227        1   999        0 nonexistent
## 4 telephone   jun         fri       17        3   999        0 nonexistent
## 5  cellular   nov         mon       58        1   999        0 nonexistent
## 6  cellular   sep         thu      128        3   999        2     failure
##   emp.var.rate cons.price.idx cons.conf.idx euribor3m nr.employed  y
## 1         -1.8         92.893         -46.2     1.313      5099.1 no
## 2          1.1         93.994         -36.4     4.855      5191.0 no
## 3          1.4         94.465         -41.8     4.962      5228.1 no
## 4          1.4         94.465         -41.8     4.959      5228.1 no
## 5         -0.1         93.200         -42.0     4.191      5195.8 no
## 6         -1.1         94.199         -37.5     0.884      4963.6 no
```

```
summary(Bkdata)
```

```
##       age                 job           marital    
##  Min.   :18.00   admin.     :1012   divorced: 446  
##  1st Qu.:32.00   blue-collar: 884   married :2509  
##  Median :38.00   technician : 691   single  :1153  
##  Mean   :40.11   services   : 393   unknown :  11  
##  3rd Qu.:47.00   management : 324                  
##  Max.   :88.00   retired    : 166                  
##                  (Other)    : 649                  
##                education       default        housing           loan     
##  university.degree  :1264   no     :3315   no     :1839   no     :3349  
##  high.school        : 921   unknown: 803   unknown: 105   unknown: 105  
##  basic.9y           : 574   yes    :   1   yes    :2175   yes    : 665  
##  professional.course: 535                                               
##  basic.4y           : 429                                               
##  basic.6y           : 228                                               
##  (Other)            : 168                                               
##       contact         month      day_of_week    duration     
##  cellular :2652   may    :1378   fri:768     Min.   :   0.0  
##  telephone:1467   jul    : 711   mon:855     1st Qu.: 103.0  
##                   aug    : 636   thu:860     Median : 181.0  
##                   jun    : 530   tue:841     Mean   : 256.8  
##                   nov    : 446   wed:795     3rd Qu.: 317.0  
##                   apr    : 215               Max.   :3643.0  
##                   (Other): 203                               
##     campaign          pdays          previous            poutcome   
##  Min.   : 1.000   Min.   :  0.0   Min.   :0.0000   failure    : 454  
##  1st Qu.: 1.000   1st Qu.:999.0   1st Qu.:0.0000   nonexistent:3523  
##  Median : 2.000   Median :999.0   Median :0.0000   success    : 142  
##  Mean   : 2.537   Mean   :960.4   Mean   :0.1903                     
##  3rd Qu.: 3.000   3rd Qu.:999.0   3rd Qu.:0.0000                     
##  Max.   :35.000   Max.   :999.0   Max.   :6.0000                     
##                                                                      
##   emp.var.rate      cons.price.idx  cons.conf.idx     euribor3m    
##  Min.   :-3.40000   Min.   :92.20   Min.   :-50.8   Min.   :0.635  
##  1st Qu.:-1.80000   1st Qu.:93.08   1st Qu.:-42.7   1st Qu.:1.334  
##  Median : 1.10000   Median :93.75   Median :-41.8   Median :4.857  
##  Mean   : 0.08497   Mean   :93.58   Mean   :-40.5   Mean   :3.621  
##  3rd Qu.: 1.40000   3rd Qu.:93.99   3rd Qu.:-36.4   3rd Qu.:4.961  
##  Max.   : 1.40000   Max.   :94.77   Max.   :-26.9   Max.   :5.045  
##                                                                    
##   nr.employed     y       
##  Min.   :4964   no :3668  
##  1st Qu.:5099   yes: 451  
##  Median :5191             
##  Mean   :5166             
##  3rd Qu.:5228             
##  Max.   :5228             
## 
```

```
dim(Bkdata)
```

```
## [1] 4119   21
```

```
str(Bkdata) # gives a nice decription of each column and its factors
```

```
## 'data.frame':    4119 obs. of  21 variables:
##  $ age           : int  30 39 25 38 47 32 32 41 31 35 ...
##  $ job           : Factor w/ 12 levels "admin.","blue-collar",..: 2 8 8 8 1 8 1 3 8 2 ...
##  $ marital       : Factor w/ 4 levels "divorced","married",..: 2 3 2 2 2 3 3 2 1 2 ...
##  $ education     : Factor w/ 8 levels "basic.4y","basic.6y",..: 3 4 4 3 7 7 7 7 6 3 ...
##  $ default       : Factor w/ 3 levels "no","unknown",..: 1 1 1 1 1 1 1 2 1 2 ...
##  $ housing       : Factor w/ 3 levels "no","unknown",..: 3 1 3 2 3 1 3 3 1 1 ...
##  $ loan          : Factor w/ 3 levels "no","unknown",..: 1 1 1 2 1 1 1 1 1 1 ...
##  $ contact       : Factor w/ 2 levels "cellular","telephone": 1 2 2 2 1 1 1 1 1 2 ...
##  $ month         : Factor w/ 10 levels "apr","aug","dec",..: 7 7 5 5 8 10 10 8 8 7 ...
##  $ day_of_week   : Factor w/ 5 levels "fri","mon","thu",..: 1 1 5 1 2 3 2 2 4 3 ...
##  $ duration      : int  487 346 227 17 58 128 290 44 68 170 ...
##  $ campaign      : int  2 4 1 3 1 3 4 2 1 1 ...
##  $ pdays         : int  999 999 999 999 999 999 999 999 999 999 ...
##  $ previous      : int  0 0 0 0 0 2 0 0 1 0 ...
##  $ poutcome      : Factor w/ 3 levels "failure","nonexistent",..: 2 2 2 2 2 1 2 2 1 2 ...
##  $ emp.var.rate  : num  -1.8 1.1 1.4 1.4 -0.1 -1.1 -1.1 -0.1 -0.1 1.1 ...
##  $ cons.price.idx: num  92.9 94 94.5 94.5 93.2 ...
##  $ cons.conf.idx : num  -46.2 -36.4 -41.8 -41.8 -42 -37.5 -37.5 -42 -42 -36.4 ...
##  $ euribor3m     : num  1.31 4.86 4.96 4.96 4.19 ...
##  $ nr.employed   : num  5099 5191 5228 5228 5196 ...
##  $ y             : Factor w/ 2 levels "no","yes": 1 1 1 1 1 1 1 1 1 1 ...
```

```
class(Bkdata)
```

```
## [1] "data.frame"
```

```
#####################################################

# Finding correlation matrix for all attributes
# Then selecting the ones with moderate to high correlation to response variable i.e target of Subscription is "Yes"
#Repeating exercise to select attributes with fairly strong or moderate correlation coefficient

######################################################################



#remove.packages("caret")
#install.packages('Rcpp', dependencies = TRUE)
#install.packages('caret', dependencies = TRUE)
library(caret)
```

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

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

```
#remove.packages("ggplot2")
#install.packages("ggplot2")
library(ggplot2)




############ EXPLORATION ####################################
int_factorsExplore <- dummyVars(" ~ .",data = Bkdata)
newBkdataExplore <- data.frame(predict(int_factorsExplore,Bkdata))
dim(newBkdataExplore)
```

```
## [1] 4119   65
```

```
explore_cormatBk <- cor(newBkdataExplore)
dim(explore_cormatBk)
```

```
## [1] 65 65
```

```
rownames(explore_cormatBk)[65]  # limiting correlation to all variables versus my response variable
```

```
## [1] "y.yes"
```

```
refined_corBk <- explore_cormatBk[,65]

refined_corBk2 <- refined_corBk[abs(refined_corBk[])>=0.20] # refining further to identify correlations above a specific val
ue
# varied the min corrrelation amount several times to aid in selecting input variables for desired output variable
refined_corBk2 # reflects only values that have correlation above a min
```

```
##            duration               pdays            previous
##           0.4185654          -0.3320115           0.2556966
## poutcome.nonexistent    poutcome.success        emp.var.rate
##          -0.2071789           0.3258037          -0.2832157
##           euribor3m         nr.employed                y.no
##          -0.2985650          -0.3492412          -1.0000000
##               y.yes
##           1.0000000
```

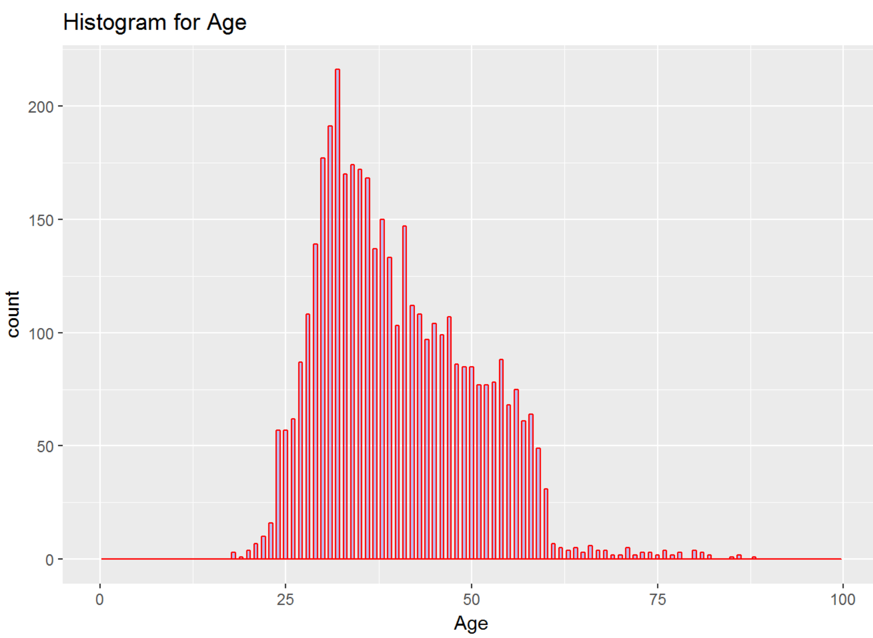
```
########################################################
# Attributes were finally selected based on the strength of the correlaton to response variable y.

#Attribute :  Age

mean_age <- mean(Bkdata$age) # mean  age of client
median_age <- median(Bkdata$age) # mean  age of client
var_age <- var(Bkdata$age) # variance in age of client

#graphics.off()
#par("mar")
#par(mar=c(1,1,1,1))

qplot(Bkdata$age,
      geom="histogram",
      binwidth = 0.5,
      main = "Histogram for Age",
      xlab = "Age",
      fill=I("blue"),
      col=I("red"),
      alpha=I(.2),
      xlim=c(0,100))
```




```
#Code REf. for hostogram:https://www.r-bloggers.com/how-to-make-a-histogram-with-ggplot2/
#############################################################
# nr.employed: number of employees - quarterly indicator (numeric)

mean_numempl  <- mean(Bkdata$nr.employed) # mean
median_numempl <- median(Bkdata$nr.employed) # median
var_numempl <- var(Bkdata$nr.employed) # variance




mean_numempl
```

```
## [1] 5166.482
```

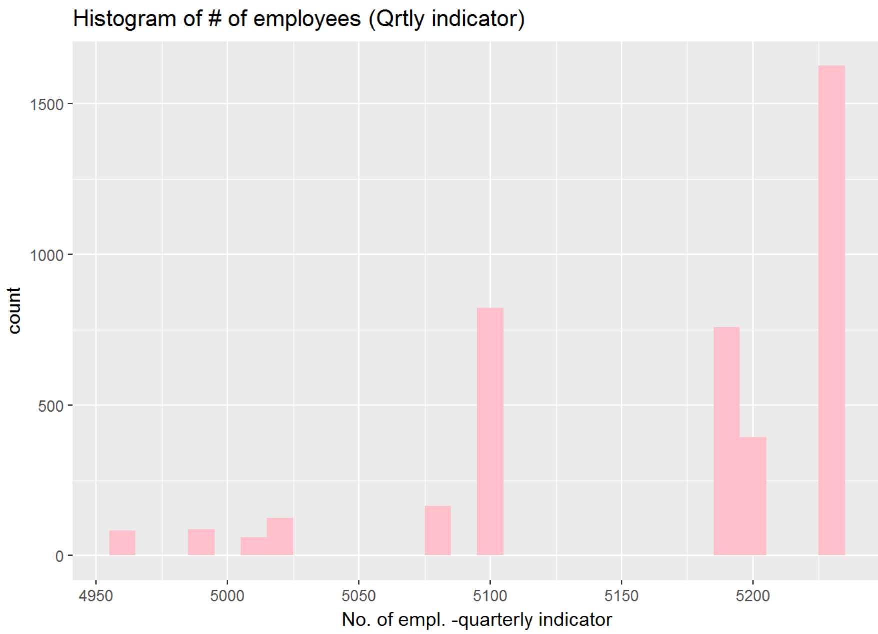
```
median_numempl
```

```
## [1] 5191
```

```
var_numempl
```

```
## [1] 5426.96
```

```
#Histogram of nr.employed: number of employees - quarterly indicator (numeric)
ggplot(Bkdata, aes(Bkdata$nr.employed)) +labs(title="Histogram of # of employees (Qrtly indicator)") + xlab("No. of empl. -q
uarterly indicator") +
  geom_histogram(binwidth=10,fill=I("pink"))
```

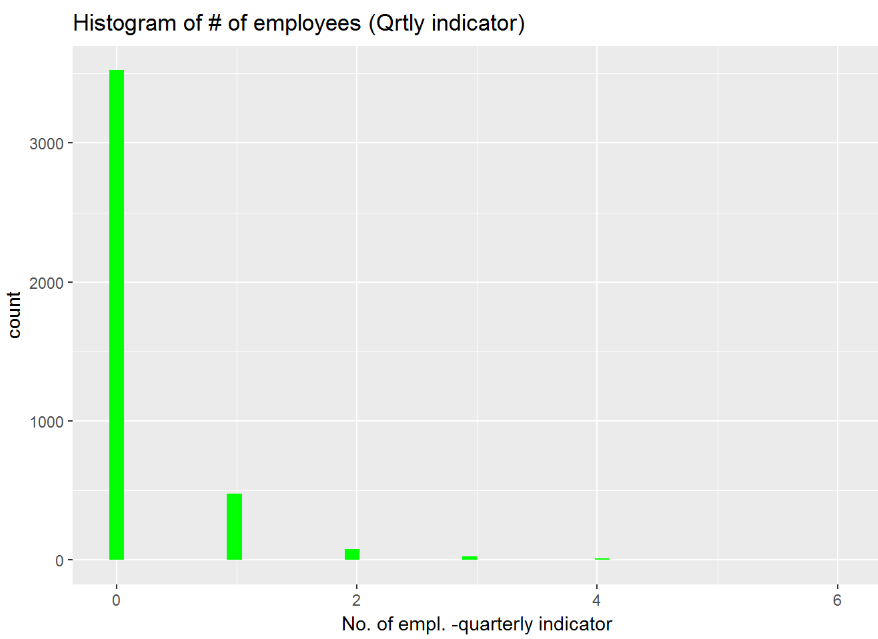



```
######################################################


#Attribute :  previous: number of contacts performed before this campaign and for this client (numeric)

mean_prev <- mean(Bkdata$previous) # mean No. of prev client contacts
median_prev <- median(Bkdata$previous)
var_prev <- var(Bkdata$previous) # variance in No. of prev client contacts

ggplot(Bkdata, aes(Bkdata$previous)) + labs(title="Histogram of # of employees (Qrtly indicator)") +
  xlab("No. of empl. -quarterly indicator") +
  geom_histogram(bins=50,fill=I("green"))
```



```
#############################################################
# emp.var.rate: employment variation rate - quarterly indicator (numeric)

mean_empvarrt <- mean(Bkdata$emp.var.rate) # mean age
median_empvarrt <- median(Bkdata$emp.var.rate) # median age
var_empvarrt <- var(Bkdata$emp.var.rate) # variance in age




mean_empvarrt
```

```
## [1] 0.08497208
```

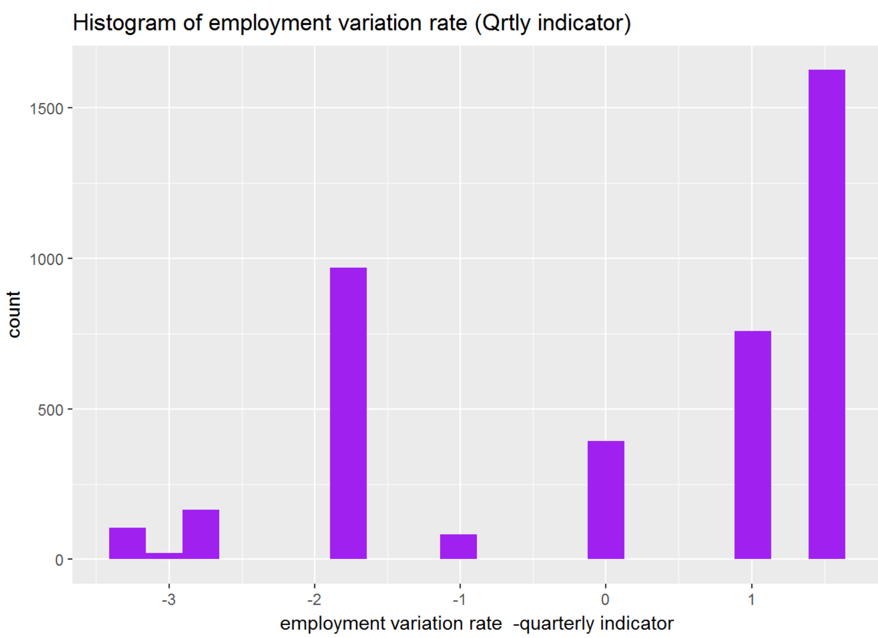
```
median_empvarrt
```

```
## [1] 1.1
```

```
var_empvarrt
```

```
## [1] 2.443327
```

```
#Histogram of employment variation rate- quarterly indicator (numeric)
ggplot(Bkdata, aes(Bkdata$emp.var.rate)) + labs(title="Histogram of employment variation rate (Qrtly indicator)") +
  xlab("employment variation rate  -quarterly indicator") +
  geom_histogram(bins=20,fill=I("purple"))
```




```
#####################################################

#Attribute : Subscription status

Bk_subscript <- Bkdata$y
head(Bk_subscript)
```

```
## [1] no no no no no no
## Levels: no yes
```

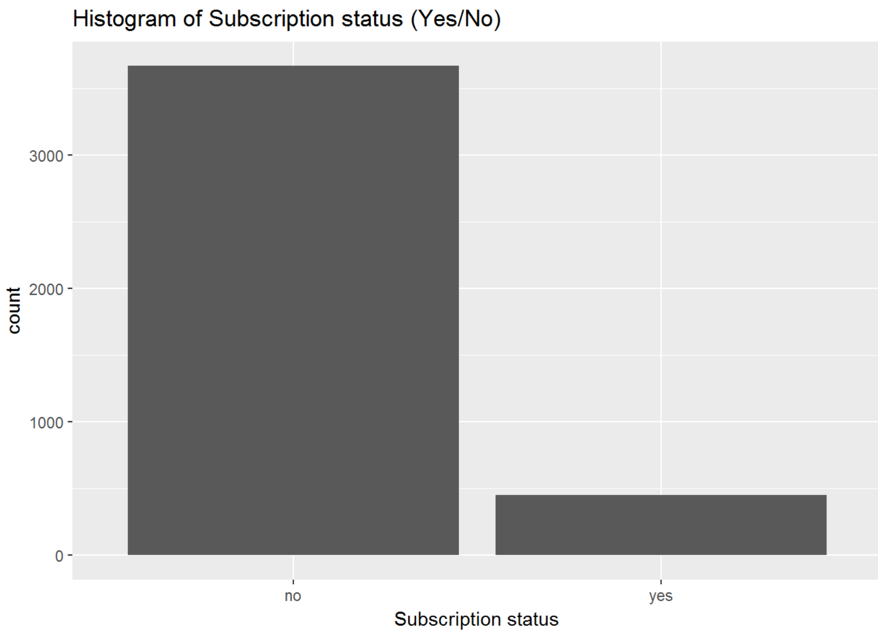
```
colnames(Bkdata)
```

```
##  [1] "age"            "job"            "marital"        "education"
##  [5] "default"        "housing"        "loan"           "contact"
##  [9] "month"          "day_of_week"    "duration"       "campaign"
## [13] "pdays"          "previous"       "poutcome"       "emp.var.rate"
## [17] "cons.price.idx" "cons.conf.idx"  "euribor3m"      "nr.employed"
## [21] "y"
```

```
target <- as.character(Bkdata$y) # Deposit subscription is referrred to as target


targetplot <- ggplot(data.frame(target), aes(x=Bk_subscript)) +
  geom_bar() + ggtitle(label="Histogram of Subscription status (Yes/No)") + xlab(label="Subscription status")

targetplot
```




```
str(Bkdata) # gives a nice decription of each column and its factors
```

```
## 'data.frame':    4119 obs. of  21 variables:
##  $ age           : int  30 39 25 38 47 32 32 41 31 35 ...
##  $ job           : Factor w/ 12 levels "admin.","blue-collar",..: 2 8 8 8 1 8 1 3 8 2 ...
##  $ marital       : Factor w/ 4 levels "divorced","married",..: 2 3 2 2 2 3 3 2 1 2 ...
##  $ education     : Factor w/ 8 levels "basic.4y","basic.6y",..: 3 4 4 3 7 7 7 7 6 3 ...
##  $ default       : Factor w/ 3 levels "no","unknown",..: 1 1 1 1 1 1 1 2 1 2 ...
##  $ housing       : Factor w/ 3 levels "no","unknown",..: 3 1 3 2 3 1 3 3 1 1 ...
##  $ loan          : Factor w/ 3 levels "no","unknown",..: 1 1 1 2 1 1 1 1 1 1 ...
##  $ contact       : Factor w/ 2 levels "cellular","telephone": 1 2 2 2 1 1 1 1 1 2 ...
##  $ month         : Factor w/ 10 levels "apr","aug","dec",..: 7 7 5 5 8 10 10 8 8 7 ...
##  $ day_of_week   : Factor w/ 5 levels "fri","mon","thu",..: 1 1 5 1 2 3 2 2 4 3 ...
##  $ duration      : int  487 346 227 17 58 128 290 44 68 170 ...
##  $ campaign      : int  2 4 1 3 1 3 4 2 1 1 ...
##  $ pdays         : int  999 999 999 999 999 999 999 999 999 999 ...
##  $ previous      : int  0 0 0 0 0 2 0 0 1 0 ...
##  $ poutcome      : Factor w/ 3 levels "failure","nonexistent",..: 2 2 2 2 2 1 2 2 1 2 ...
##  $ emp.var.rate  : num  -1.8 1.1 1.4 1.4 -0.1 -1.1 -1.1 -0.1 -0.1 1.1 ...
##  $ cons.price.idx: num  92.9 94 94.5 94.5 93.2 ...
##  $ cons.conf.idx : num  -46.2 -36.4 -41.8 -41.8 -42 -37.5 -37.5 -42 -42 -36.4 ...
##  $ euribor3m     : num  1.31 4.86 4.96 4.96 4.19 ...
##  $ nr.employed   : num  5099 5191 5228 5228 5196 ...
##  $ y             : Factor w/ 2 levels "no","yes": 1 1 1 1 1 1 1 1 1 1 ...
```

```
class(Bkdata)
```

```
## [1] "data.frame"
```

```
#install.packages("ggplot2")
#library(ggplot2)

#remove.packages("caret")
#install.packages('Rcpp', dependencies = TRUE)
#install.packages('caret', dependencies = TRUE)
library(caret)




############################################################

myBkdata <- data.frame(Bkdata$nr.employed,Bkdata$previous,Bkdata$emp.var.rate,Bkdata$age,Bkdata$y)
colnames(myBkdata)
```

```
## [1] "Bkdata.nr.employed"  "Bkdata.previous"     "Bkdata.emp.var.rate"
## [4] "Bkdata.age"          "Bkdata.y"
```

```
library(caret)

myint_factors <- dummyVars(" ~ .",data = myBkdata)
newBkdata <- data.frame(predict(myint_factors,myBkdata))
class(newBkdata)
```

```
## [1] "data.frame"
```

```
colnames(newBkdata)
```

```
## [1] "Bkdata.nr.employed"  "Bkdata.previous"     "Bkdata.emp.var.rate"
## [4] "Bkdata.age"          "Bkdata.y.no"         "Bkdata.y.yes"
```

```
dfBk <- newBkdata
class(dfBk)
```

```
## [1] "data.frame"
```

```
head(dfBk)
```

```
##   Bkdata.nr.employed Bkdata.previous Bkdata.emp.var.rate Bkdata.age
## 1             5099.1               0                -1.8         30
## 2             5191.0               0                 1.1         39
## 3             5228.1               0                 1.4         25
## 4             5228.1               0                 1.4         38
## 5             5195.8               0                -0.1         47
## 6             4963.6               2                -1.1         32
##   Bkdata.y.no Bkdata.y.yes
## 1           1            0
## 2           1            0
## 3           1            0
## 4           1            0
## 5           1            0
## 6           1            0
```

```
colnames(dfBk)
```

```
## [1] "Bkdata.nr.employed"  "Bkdata.previous"     "Bkdata.emp.var.rate"
## [4] "Bkdata.age"          "Bkdata.y.no"         "Bkdata.y.yes"
```

```
dim(dfBk)
```

```
## [1] 4119    6
```

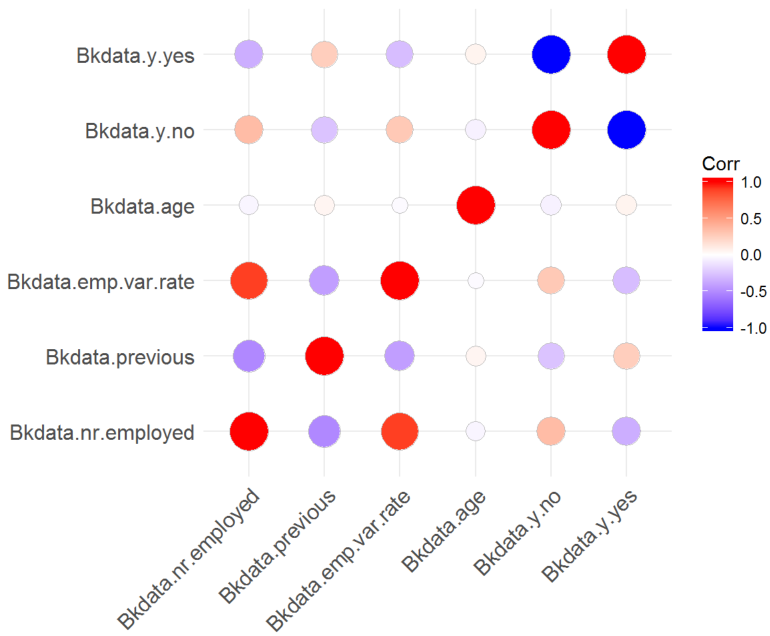
```
############################################################

###Visualizaton of Correlation matrix

#install.packages("ggcorrplot")
library(ggcorrplot)

mycorBank <- cor(dfBk)

# method = "circle"
ggcorrplot(mycorBank, method = "circle")
```




```
####################################################
###Dependencies observed
# The graph clearly depicts the strength of the correlation
#by the size and colour coding of diagram.

# The strongest positive dependencies were between:
#   (i) emp.variation rate & Number employed (Qtrly indicators)
#              - reflecting pos. cor of 0.90

# The strongest negative correlation were :
#   (i) between the previous: number of contacts performed before this campaign
#       and for this client and the number emplyed Qrtly indicator.
#               - reflecting neg. cor of -0.51

#   (i) between the previous: number of contacts performed before this campaign
#       and emp.var.rate.
#               - reflecting neg. cor of -0.42

# Low to moderate dependencies were between deposit yes:
#    (i)  nr.employed (# employed) : neg. cor of -0.35
#    (ii) previous : pos cor of 0.26
#    (iii) emp.var.rate : neg. cor of -0.28


mycorBank[,"Bkdata.nr.employed"]
```

```
##  Bkdata.nr.employed     Bkdata.previous Bkdata.emp.var.rate
##          1.00000000         -0.51485341          0.89717322
##          Bkdata.age          Bkdata.y.no         Bkdata.y.yes
##         -0.04193615          0.34924123         -0.34924123
```

```
mycorBank["Bkdata.y.yes",]
```

```
##  Bkdata.nr.employed     Bkdata.previous Bkdata.emp.var.rate
##         -0.34924123          0.25569663         -0.28321568
##          Bkdata.age          Bkdata.y.no         Bkdata.y.yes
##          0.06037408         -1.00000000          1.00000000
```

```
mycorBank[,"Bkdata.previous"]
```

```
##  Bkdata.nr.employed     Bkdata.previous Bkdata.emp.var.rate
##         -0.51485341          1.00000000         -0.41523813
##          Bkdata.age         Bkdata.y.no        Bkdata.y.yes
##          0.05093104         -0.25569663          0.25569663
```

```
mycorBank[,"Bkdata.emp.var.rate"]
```

```
##  Bkdata.nr.employed     Bkdata.previous Bkdata.emp.var.rate
##          0.89717322         -0.41523813          1.00000000
##          Bkdata.age         Bkdata.y.no        Bkdata.y.yes
##         -0.01919176          0.28321568         -0.28321568
```

```
############################################################
##########             ANALYSIS                  ###########
###########################################################################
###        TRAINING the model                               ##############

### We are selecting the response variable as y - has the client subscribed a term deposit? (binary: 'yes','no')

#### Step 1

###  Splitting the data set into two parts : 75% - training set & 25% - test set

bkTr_rnum <- 0.75*nrow(dfBk)

mydfBk_Tr <- dfBk[(1:bkTr_rnum),]
dim(mydfBk_Tr)
```

```
## [1] 3089    6
```

```
mydfBk_Test <- dfBk[((bkTr_rnum+1):nrow(dfBk)),]



##Step #2 : Training the model - we use training data set only
#### Find Posterior probability of each class P(Ci|x) i = 1, 2 only ( i.e. 2 classes)

#To find the posterior we need the Prior Probability P(x|Ci) i.e class likelihood
# and Probability of each class P(Ci)
# Note we do need the evidence since it will be common in both classes we can
#eliminate/ignore it as we will be comparing the posterior in both classes to
#choose a max for class assignment. Hence, a common denominator ( evidence : P(X))
#can be ignored

# Let us proceed with deriving an estimate for Sigma (covariance matrix)
# since it will be needed to compute the Posterior Prob.
# We must separate training set into the different classes and find
#Posterior for each class

colnumBk_Tr <- dim(mydfBk_Tr)[2]

# using only input variables and ensuring data points belong to class 1
mydfBk_Tr_class1 <- mydfBk_Tr[(mydfBk_Tr[,colnumBk_Tr]==1),-c(colnumBk_Tr-1,colnumBk_Tr)]

mydfBk_Tr_class0 <- mydfBk_Tr[(mydfBk_Tr[,colnumBk_Tr]==0),-c(colnumBk_Tr-1,colnumBk_Tr)]

# Separating the input values from the response values for the test set as well
mydfBk_TestInput <- dfBk[((bkTr_rnum+1):nrow(dfBk)),-c(colnumBk_Tr-1,colnumBk_Tr)]

# response variable only for test set
r_test <-  dfBk[((bkTr_rnum+1):nrow(dfBk)),colnumBk_Tr]

dim(mydfBk_Tr_class1) # confirming dim of class for training set
```

```
## [1] 342   4
```

```
dim(mydfBk_Tr_class0)
```

```
## [1] 2747    4
```

```
#Finding the estimate for the covarince matrix denoted by Si ( for class i).
# Starting with class i=1 ( representing success re deposit subscription)

m1 <- apply(mydfBk_Tr_class1,2,mean)
m0 <- apply(mydfBk_Tr_class0,2,mean)

x_Cl1 <-mydfBk_Tr_class1
x_Cl0 <-mydfBk_Tr_class0

nrows1 <- dim(x_Cl1)[1]  # # instances (rows) for class1
nrows0 <- dim(x_Cl0)[1]

#install.packages("MASS")
#library(MASS)



covariance_matrix1 <- function(d) {
  # calculating covriance matrix from the given sample d

  cn <- ncol(d)
  tn <- nrow(d)
  result <- matrix(c(rep(0,cn*cn)),cn,cn)
  ms <- apply(d,2,mean) #calculates actual mean of Sample per column
  i <- 1
  while (i <= cn)
  {
    j <- 1
    while (j <=cn) {
      if (j !=i) {
        diffi <- matrix(c(rep(0,tn)),tn,cn)
        diffj <- matrix(c(rep(0,tn)),tn,cn)
        prodt <- matrix(c(rep(0,tn)),tn,cn)
        for (t in 1:tn) {
          diffi[t,i] <- d[t,i] - ms[i]

          diffj[t,j] <- d[t,j] -ms[j]

          prodt[t] <- diffi[t,i] * diffj[t,j]

        }#end for
        result[i,j] <- (sum(prodt))/tn

      }# endif
      j <- j +1
    }#end nested while
    result[i,i] <- (sum((d[,i]-ms[i])^2))/tn # cal. of variance
    i <- i +1

  } #end 1st while
  return(result);
} #end function


S1 <- covariance_matrix1(x_Cl1)
S0 <- covariance_matrix1(x_Cl0)



#calculating  class likelihood p(x|Ci)
Class_likeihoodi <- function(di,mySi,mi){

  drn <- dim(di)[1]
  cn <- dim(di)[2]
  result_like <- rep(0,drn)
  for (t in 1:drn){
       a <- (t(di[t,]-mi))
       b <- solve(mySi)
       c <- as.matrix(di[t,]-mi)
       dd <- sqrt(det(mySi))
       e <- (2*pi)^(cn*0.5)

       result_like[t] <- (1/(e*dd))*exp(-0.5*c%*%b%*%a)
      }
      return(result_like)
}
likelihood_cl1<- Class_likeihoodi(mydfBk_TestInput,S1,m1)

likelihood_cl0<- Class_likeihoodi(mydfBk_TestInput,S0,m0)
```

```r
Pc1 <- (dim(x_Cl1)[1]) / ((dim(x_Cl1)[1])+(dim(x_Cl0)[1])) # Prob. of Class 1
Pc0 <- (dim(x_Cl0)[1]) / ((dim(x_Cl1)[1])+(dim(x_Cl0)[1])) # Prob. of class 0


PostProb1 <- likelihood_cl1*Pc1 # Posterior Prob for class 1 for test dataset
PostProb0 <- likelihood_cl0*Pc0 # Posterior Prob for class 0 for test dataset

# Classifying based on the maximum post. prob. for each test data point
classification <- function(p1,p0){
  lenp <- length(p1)
  rclass <- rep(99,lenp) # 99 depicts the rejected datapoint
                        # value will be preserved if p0 = p1

  for (cnt in 1:lenp){

    if (p1[cnt] > p0[cnt]){
      # if posterior prob. for class1 is greater than that of class0
      # then assign the data point to class 1
      rclass[cnt]<- 1}

    else if (p0[cnt] > p1[cnt]){
         rclass[cnt] <- 0}
 }# end for-loop
  return(rclass);
}#end function


modelresultTest <- classification(PostProb1,PostProb0)
ErrorTest <- function(r_model,r_actual){
  # Calculating how accurate model was vs. actual "correct" values

  lenr <- length(r_actual)
  resultTest <- rep(0,lenr)

  for (a in 1:lenr){
  if (r_model[a] == r_actual[a]){
     resultTest[a] = 1

  } else if (r_model[a] != 99){
    resultTest[a] = 0

    }#end else if
  }#end for loop

  return(resultTest);
}# end function

ErrorTestresults <- ErrorTest(modelresultTest,r_test)
PercError <- 100*sum(ErrorTestresults==0)/length(ErrorTestresults)
PercCorrect <- 100*sum(ErrorTestresults==1)/length(ErrorTestresults)
PercReject <- 100*sum(ErrorTestresults==99)/length(ErrorTestresults)

PercError
```

```
## [1] 10.30126
```

```r
PercCorrect
```

```
## [1] 89.69874
```

```r
PercReject
```

```
## [1] 0
```

```
# Model has an error rate of 10.30% or an Acurracy rate of 89.70%
#and zero rejection rate

# We will now test the accuracy of model with different parameters,
#specifically, we will vary the covariance matrix
### Using the equation below we wil vary alpha ("A") and Beta ("B")
#Sinew rep new Si (covariance matrix after varying A and B)
# s2 rep. variance which is going to be an average of the vars in S
# where S is derived by using P(C1)*S1 +P(C0)*S0
# and S1 rep covriance matrix for class 1 when i = 1 in Si
#and S0 rep covariance matrix for class 0 when i = 0 in Si

#Sinew <- A*s2*I + B*S + (1-A-B)*Si

S <- Pc1*S1 + Pc0*S0  # Matrix for entire dataset; notice weighted based on resp. prob.


# Let us derive s2, the variance, which is going to be an average of the vars in S
avgs2 <- function(covS){
  dimS <- dim(covS)[2] # symmentrical so either row or column will denote data dimension

   getavg <- (1/dimS)*(sum(diag(covS))) # changes diagonals (variance values) to the avg.

   return(getavg)
}# end function

s2 <- avgs2(S) # returns avg. variance value of given covariance matrix
               # in this case, it is the avg. using covriance matrix irrespective of class


# Case 1 : let A ("alpha") = 1 & B ("Beta") = 0
# Equation reduces to: Sinew = s2*I (I rep Identity matrix)
# In this case, Sinew for each class will have the attributes that are independent
# since the covariance values will be zero and all the atributes will have
# the same variance value.
### Note the main equation in  the function below Sinew which will be reduced
### as alpha and beta values vary or the various scenarios
### In case #2 and additional constraint is imposed furter to a change of value
#for alpha and beta

Sinew <- function(s2,Si,S,A,B){
  I <- diag(dim(S)[1])

  resultSinew <- A*s2*I + B*S + (1-A-B)*Si
  return(resultSinew)

}


offdiagzero <- function(Scov){
  ## takes matrix and sets off-diagonal elements to zero

  I <- diag(dim(S)[1]) # Identity matrix created

  for (cnti in 1:dim(S)[1]){
    #setting diagonal elements in matrix
    #to the identity matrix
       I[cnti,cnti] <- Scov[cnti,cnti]

  }#end for loop
  return(I)
}# end function

#s2 rep variance
# S rep. cov. matrix derived using both classes
# S1 & S0 rep. cov. matrix from class1 and class0 respectively

S1newCase1 <- Sinew(s2,S1,S,1,0) #alpha = 1 & Beta = 0
S1newCase2 <- offdiagzero(Sinew(s2,S1,S,0,1)) #alpha = 0 & Beta = 1 and offdiag.set to zero
S1newCase3 <- Sinew(s2,S1,S,0,1) #alpha = 0 & Beta = 1
S1newCase4 <- Sinew(s2,S1,S,0,0) #alpha = 0 & Beta = 0

S0newCase1 <- Sinew(s2,S0,S,1,0)
S0newCase2 <- offdiagzero(Sinew(s2,S0,S,0,1))
S0newCase3 <- Sinew(s2,S0,S,0,1)
S0newCase4 <- Sinew(s2,S0,S,0,0)


### Case1 cont'd. Deriving posterior and effecting classification based on
#### new cov matrix (where newSi = s2*I); recall, we let A ("alpha") = 1 & B ("Beta") = 0
#### as shown above S1newCase1, look at arguments passed to function on RHS
```

```
like_Newcase1_cl1<- Class_likeihoodi(mydfBk_TestInput,S1newCase1,m1)
like_Newcase1_cl0<- Class_likeihoodi(mydfBk_TestInput,S0newCase1,m0)

###These will not change as we vary S (covariance matrix from case to case)
Pc1 <- (dim(x_Cl1)[1]) / ((dim(x_Cl1)[1])+(dim(x_Cl0)[1])) # Prob. of Class 1
Pc0 <- (dim(x_Cl0)[1]) / ((dim(x_Cl1)[1])+(dim(x_Cl0)[1])) # Prob. of class 0


PostProb1_newCase1 <- like_Newcase1_cl1*Pc1 # Posterior Prob for class 1 for test dataset
PostProb0_newCase1 <- like_Newcase1_cl0*Pc0 # Posterior Prob for class 0 for test dataset

##Classifying


#initializing (only necessary for case1)
PercError_newCase <- rep(55,4)
PercCorrect_newCase<- rep(55,4)
PercReject<- rep(55,4)

Tot <- length(r_test) # Will not change as case varies

modelresultTest_newCase1 <- classification(PostProb1_newCase1,PostProb0_newCase1)
ErrorTestresults_newCase1 <- ErrorTest(modelresultTest_newCase1,r_test)
PercError_newCase[1] <- 100*sum(ErrorTestresults_newCase1==0)/Tot
PercCorrect_newCase[1] <- 100*sum(ErrorTestresults_newCase1==1)/Tot
PercReject[1] <- 100*sum(ErrorTestresults_newCase1==99)/Tot



### Case2 Deriving posterior and effecting classification based on
#### new cov matrix (where newSi = S where sij =0 i.e off-diag. are zero
#### in other words when the attributes are independent since covariances equal zero)

like_Newcase2_cl1<- Class_likeihoodi(mydfBk_TestInput,S1newCase2,m1)
like_Newcase2_cl0<- Class_likeihoodi(mydfBk_TestInput,S0newCase2,m0)


PostProb1_newCase2 <- like_Newcase2_cl1*Pc1 # Posterior Prob for class 1 for test dataset
PostProb0_newCase2 <- like_Newcase2_cl0*Pc0 # Posterior Prob for class 0 for test dataset

##Classifying



modelresultTest_newCase2 <- classification(PostProb1_newCase2,PostProb0_newCase2)
ErrorTestresults_newCase2 <- ErrorTest(modelresultTest_newCase2,r_test)

PercError_newCase[2] <- 100*sum(ErrorTestresults_newCase2==0)/Tot
PercCorrect_newCase[2] <- 100*sum(ErrorTestresults_newCase2==1)/Tot
PercReject[2] <- 100*sum(ErrorTestresults_newCase2==99)/Tot



### Case3 Deriving posterior and effecting classification based on
#### new cov matrix (where newSi = S so both classes share the same cov. matrix)
#### This occurs when "A" (alpha = 0) and "B" (Beta = 1)

like_Newcase3_cl1<- Class_likeihoodi(mydfBk_TestInput,S1newCase3,m1)
like_Newcase3_cl0<- Class_likeihoodi(mydfBk_TestInput,S0newCase3,m0)


PostProb1_newCase3 <- like_Newcase3_cl1*Pc1 # Posterior Prob for class 1 for test dataset
PostProb0_newCase3 <- like_Newcase3_cl0*Pc0 # Posterior Prob for class 0 for test dataset

##Classifying



modelresultTest_newCase3 <- classification(PostProb1_newCase3,PostProb0_newCase3)
ErrorTestresults_newCase3 <- ErrorTest(modelresultTest_newCase3,r_test)

PercError_newCase[3] <- 100*sum(ErrorTestresults_newCase3==0)/Tot
PercCorrect_newCase[3] <- 100*sum(ErrorTestresults_newCase3==1)/Tot
PercReject[3] <- 100*sum(ErrorTestresults_newCase3==99)/Tot



### Case4 - Keeping original values for Si
#### This occurs when "A" (alpha = 0) and "B" (Beta = 0)
```

```
like_Newcase4_cl1<- Class_likeihoodi(mydfBk_TestInput,S1newCase4,m1)
like_Newcase4_cl0<- Class_likeihoodi(mydfBk_TestInput,S0newCase4,m0)


PostProb1_newCase4 <- like_Newcase4_cl1*Pc1 # Posterior Prob for class 1 for test dataset
PostProb0_newCase4 <- like_Newcase4_cl0*Pc0 # Posterior Prob for class 0 for test dataset

##Classifying



modelresultTest_newCase4 <- classification(PostProb1_newCase4,PostProb0_newCase4)
ErrorTestresults_newCase4 <- ErrorTest(modelresultTest_newCase4,r_test)

PercError_newCase[4] <- 100*sum(ErrorTestresults_newCase4==0)/Tot
PercCorrect_newCase[4] <- 100*sum(ErrorTestresults_newCase4==1)/Tot
PercReject[4] <- 100*sum(ErrorTestresults_newCase4==99)/Tot



########################
### Case#5 - We will try one more case where alpha = 0.5 and beta = 0.5

S1newCase5 <- Sinew(s2,S1,S,0.5,0.5)
S0newCase5 <- Sinew(s2,S0,S,0.5,0.5)


like_Newcase5_cl1<- Class_likeihoodi(mydfBk_TestInput,S1newCase5,m1)
like_Newcase5_cl0<- Class_likeihoodi(mydfBk_TestInput,S0newCase5,m0)


PostProb1_newCase5 <- like_Newcase5_cl1*Pc1 # Posterior Prob for class 1 for test dataset
PostProb0_newCase5 <- like_Newcase5_cl0*Pc0 # Posterior Prob for class 0 for test dataset

##Classifying



modelresultTest_newCase5 <- classification(PostProb1_newCase5,PostProb0_newCase5)
ErrorTestresults_newCase5 <- ErrorTest(modelresultTest_newCase5,r_test)

PercError_newCase[5] <- 100*sum(ErrorTestresults_newCase5==0)/Tot
PercCorrect_newCase[5] <- 100*sum(ErrorTestresults_newCase5==1)/Tot
PercReject[5] <- 100*sum(ErrorTestresults_newCase5==99)/Tot







#########
## Comparing we get
ErrorComparisonCase1_2_3_4_5 <- data.frame(CaseNo=c(1,2,3,4,5),alpha=c(1,0,0,0,0.5),Beta=c(0,1,1,0,0.5),PercError=PercError_
newCase,PercCorrect=PercCorrect_newCase,PercReject)

ErrorComparisonCase1_2_3_4_5
```

```
##   CaseNo alpha Beta PercError PercCorrect PercReject
## 1      1   1.0  0.0  25.07289    74.92711          0
## 2      2   0.0  1.0  15.16035    84.83965          0
## 3      3   0.0  1.0  10.59281    89.40719          0
## 4      4   0.0  0.0  10.30126    89.69874          0
## 5      5   0.5  0.5  10.68999    89.31001          0
```

```
# Case#4 which is our original case where we have two different covariance matrices
# is when the accuracy of our model is the highest.

#Recall that Case#2 has an additional constraint imposed to that of case #2 where
# off-diagonal terms are set to zero
library(caret)

ErrorComparisonCase1_2_3_4_5
```

```
##   CaseNo alpha Beta PercError PercCorrect PercReject
## 1      1   1.0  0.0  25.07289    74.92711          0
## 2      2   0.0  1.0  15.16035    84.83965          0
## 3      3   0.0  1.0  10.59281    89.40719          0
## 4      4   0.0  0.0  10.30126    89.69874          0
## 5      5   0.5  0.5  10.68999    89.31001          0
```