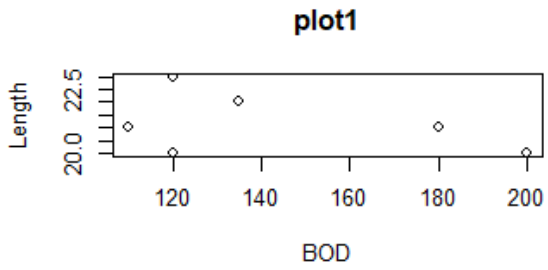
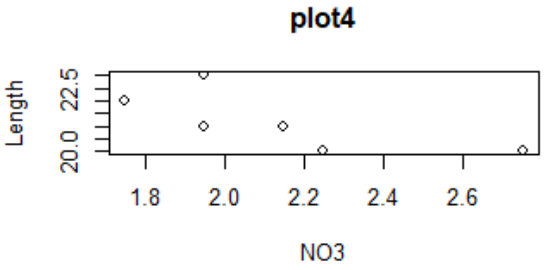
Multivariate Data Matrix(10th) OTUPUT

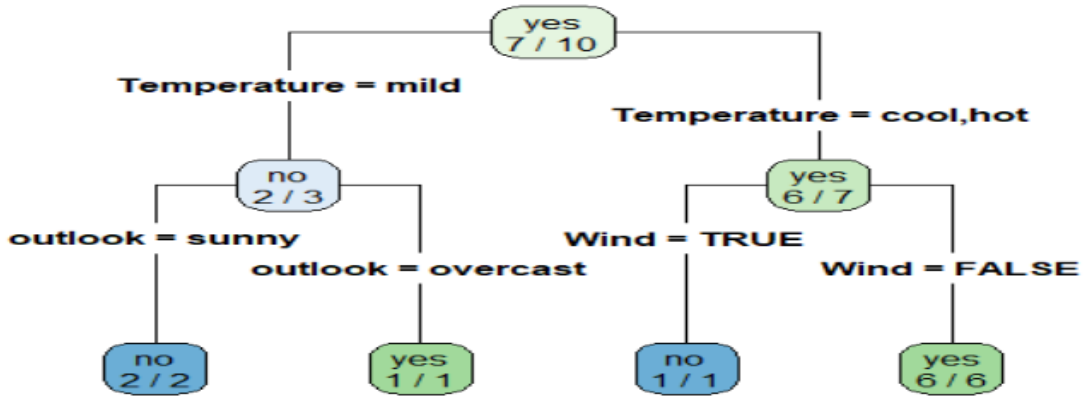
```
> Length<-c(20,21,22,23,21,20)
> Speed<-c(12,14,12,16,20,21)
> Algae<-c(40,45,45,80,75,65)
> NO3<-c(2.25,2.15,1.75,1.95,1.95,2.75)
> BOD<-c(200,180,135,120,110,120)
> mf<-data.frame(Length,Speed,Algae,NO3,BOD)
> mf
  Length Speed Algae  NO3 BOD
1     20    12    40 2.25 200
2     21    14    45 2.15 180
3     22    12    45 1.75 135
4     23    16    80 1.95 120
5     21    20    75 1.95 110
6     20    21    65 2.75 120
> Opt=par(mfrow=c(2,2))
> Opt
$mfrow
[1] 2 2

> plot(Length~BOD,data=mf, main='plot1')
> plot.new()
> plot.new()
> plot(Length ~ NO3,data=mf,main='plot4')
> -----MULTIVARIATIVE MATRIX PLOT OUTPUT----------------
```





Decision Tree(9th) OTUPUT

```
   play..outlook.Temperature.Humidity..Wind
1     yes    rainy        cool   normal FALSE
2      no    rainy        cool   normal  TRUE
3     yes overcast         hot     high FALSE
4      no    sunny        mild     high FALSE
5     yes    rainy        cool   normal FALSE
6     yes    sunny        cool   normal FALSE
7     yes    rainy        cool   normal FALSE
8     yes    sunny         hot   normal FALSE
9     yes overcast        mild     high  TRUE
10     no    sunny        mild     high  TRUE
>
> fit=rpart(play ~ outlook + Temperature  +Humidity +Wind,method="class",data
=p2,control=rpart.control(minsplit=1),parms=list(split='information'))
> fit
n= 10
```



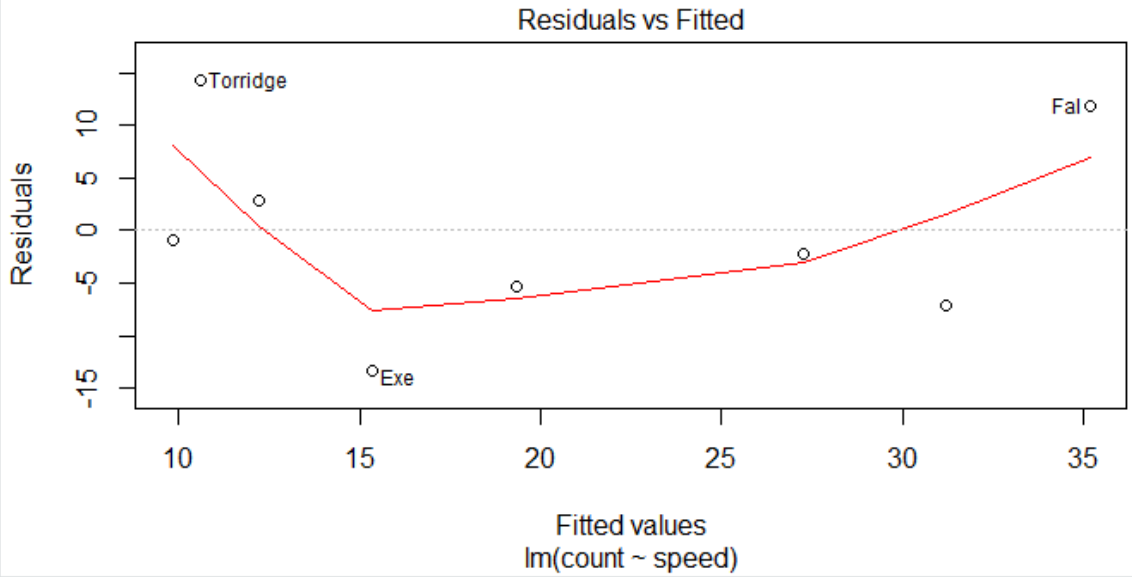
PROGRAM 2 OUTPUT

```
> BuffTail<-c(10,1,37,5,12)
> Gardenbee<-c(8,3,9,6,4)
> RedTail<-c(18,9,12,4,6)
> Carderbee<-c(8,277,6,32,23)
> HoneyBee<-c(12,13,16,9,10)
> beeframe<-data.frame(BuffTail,Gardenbee,RedTail,Carderbee,HoneyBee)
> beeframe
  BuffTail Gardenbee RedTail Carderbee HoneyBee
1       10         8      18         8       12
2        1         3       9       277       13
3       37         9      12         6       16
4        5         6       4        32        9
5       12         4       6        23       10
> names<-c("Thistle","Vipers","GoldenRain","Yellowalfala","blackberry")
> rownames(beeframe)<-names
> beeframe
             BuffTail Gardenbee RedTail Carderbee HoneyBee
Thistle            10         8      18         8       12
Vipers              1         3       9       277       13
GoldenRain         37         9      12         6       16
Yellowalfala        5         6       4        32        9
blackberry         12         4       6        23       10
>
```

Linear Regression(7th) OTUPUT

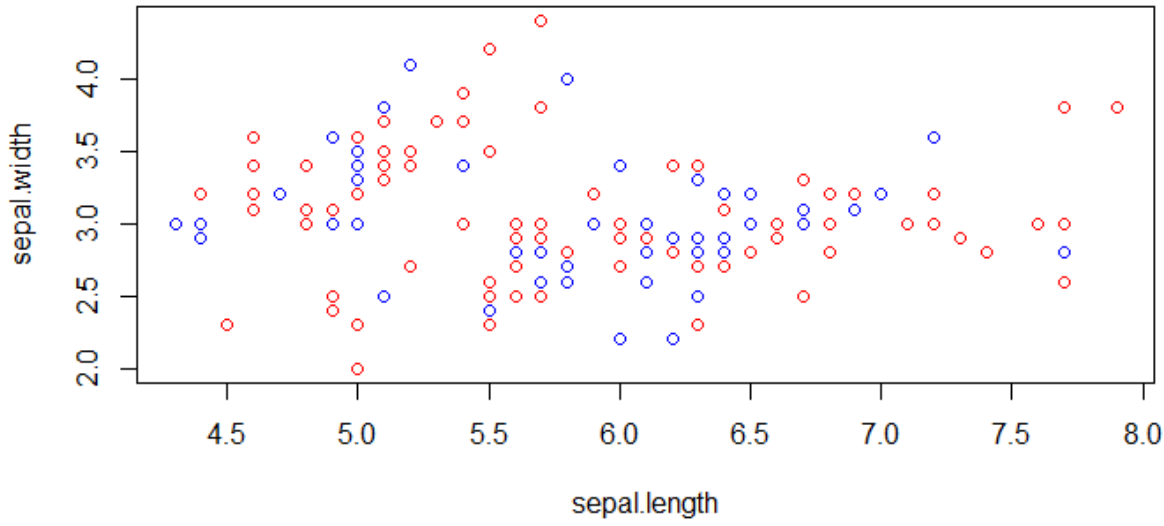
```
 [1] "coefficients"  "residuals"     "effects"       "rank"
 [5] "fitted.values" "assign"        "qr"            "df.residual"
 [9] "xlevels"       "call"          "terms"         "model"
> fw.lm$coefficients
(Intercept)       speed
  8.2545956   0.7913603
>
> #Gives slope and Intercept
> newypred<-fitted(fw.lm)                    #predict y values  for each x
ue
> newypred
      Taw  Torridge      Ouse       Exe       Lyn     Brook     Ditch
 9.837316 10.628676 12.211397 15.376838 19.333640 27.247243 31.204044
      Fal
35.160846
> #Obtaining confidence Intervals
> confint(fw.lm)  #obtain the confidence intervals
                  2.5 %    97.5 %
(Intercept) -6.06752547 22.576717
speed        0.03756445  1.545156
> confint(fw.lm,parm=c('(Intercept)','speed'),level =0.9)
                  5 %      95 %
(Intercept) -3.119113 19.628305
speed        0.192744  1.389977
> #Fitted Values
> fitted(fw.lm)
      Taw  Torridge      Ouse       Exe       Lyn     Brook     Ditch
 9.837316 10.628676 12.211397 15.376838 19.333640 27.247243 31.204044
      Fal
35.160846
> residuals(fw.lm)
        Taw    Torridge        Ouse         Exe         Lyn       Brook
 -0.8373162  14.3713235   2.7886029 -13.3768382  -5.3336397  -2.2472426
      Ditch         Fal
 -7.2040441  11.8391544
> #plotting the x , y values
> plot(fw$speed,fw$count,col="red")
> coef(fw.lm)
```

```
          count speed
Taw           9     2
Torridge     25     3
Ouse         15     5
Exe           2     9
Lyn          14    14
Brook        25    24
Ditch        24    29
Fal          47    34
> fw.lm=lm(count ~ speed,data=fw)
> summary(fw.lm)

Call:
lm(formula = count ~ speed, data = fw)

Residuals:
    Min      1Q  Median      3Q     Max
-13.377  -5.801  -1.542   5.051  14.371

Coefficients:
            Estimate Std. Error t value Pr(>|t|)
(Intercept)   8.2546     5.8531   1.410   0.2081
speed         0.7914     0.3081   2.569   0.0424 *
---
Signif. codes:  0 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 0.1 ‘ ’ 1

Residual standard error: 10.16 on 6 degrees of freedom
Multiple R-squared:  0.5238,    Adjusted R-squared:  0.4444
F-statistic: 6.599 on 1 and 6 DF,  p-value: 0.0424
```



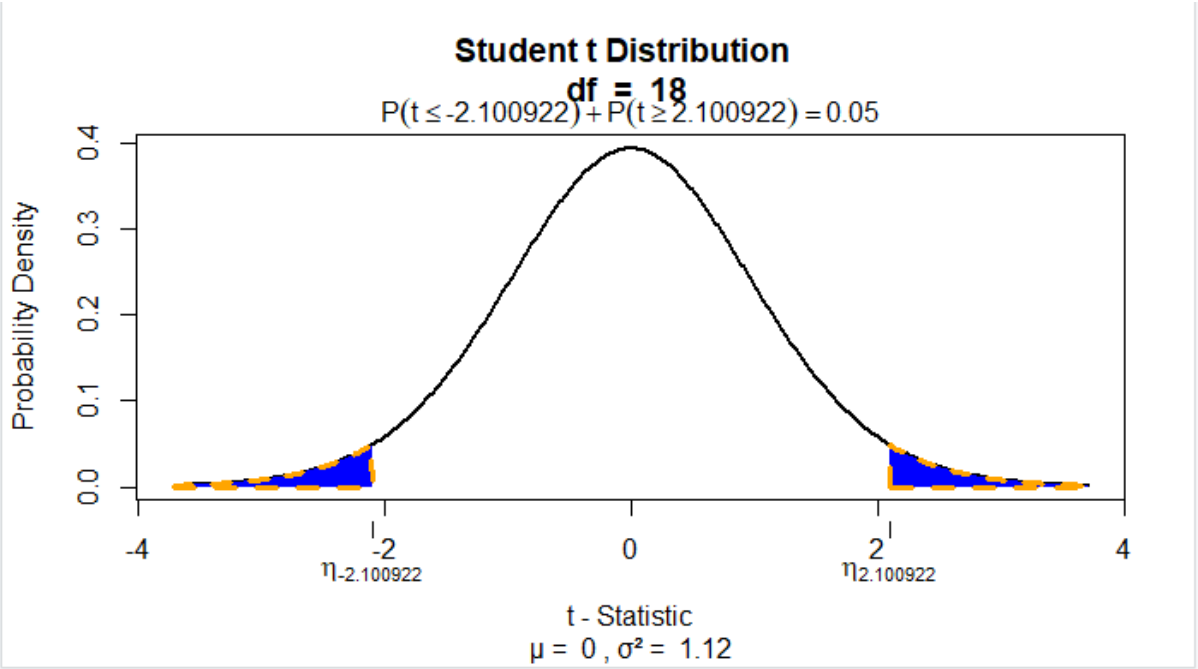
K-Means Cluster(6th) OTUPUT

```
K-means clustering with 2 clusters of sizes 4, 2

Cluster means:
      x     y
1 183.5 72.25
2 169.0 58.00

Clustering vector:
[1] 1 2 2 1 1 1

Within cluster sum of squares by cluster:
[1] 85.75 10.00
 (between_SS / total_SS =  85.2 %)

Available components:

[1] "cluster"      "centers"      "totss"        "withinss"
[5] "tot.withinss" "betweenss"    "size"         "iter"
[9] "ifault"
> km$cluster
[1] 1 2 2 1 1 1
> km$centers
      x     y
1 183.5 72.25
2 169.0 58.00
> km$withinss
[1] 85.75 10.00
> km$betweenss
[1] 551.0833
> km$totss
[1] 646.8333
>
> #Visualizing clusters
```



T-test Hypothesis (5th) OTUPUT

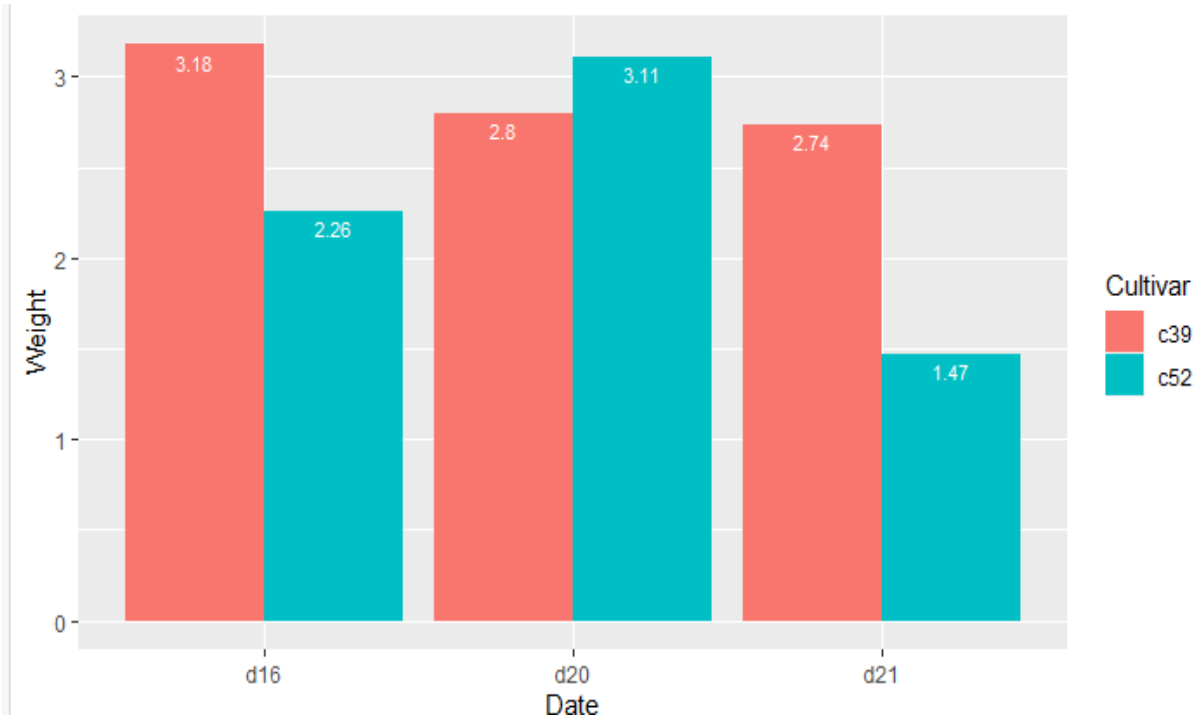
```
[1] 2.89
> xurbar=mean(urban)
> xurbar
[1] 3.17
> var(rural)
[1] 0.02766667
> sd(rural)
[1] 0.166333
> var(urban)
[1] 0.06011111
> sd(urban)
[1] 0.2451757
> #Obtaining t-calculated value
> t.test(x=rural,y=urban,var.equal = TRUE,conf.level = 0.95)

        Two Sample t-test

data:  rural and urban
t = -2.9886, df = 18, p-value = 0.007878
alternative hypothesis: true difference in means is not equal to 0
95 percent confidence interval:
 -0.47683496 -0.08316504
sample estimates:
mean of x mean of y
     2.89      3.17

> #t.test(x=xrbar,y=xurbar,var.equal = TRUE)
> #Obtain t value for two sided test at 0.05 significance levels
> #From t distribution table or t-significant,t-critical
> qt(p=0.05/2,df=18,lower.tail = FALSE)
[1] 2.100922
> visualize.t(stat=c(-2.9886,2.9886),df=18,section="tails")
> visualize.t(stat=c(-2.100922,2.100922),df=18,section="tails")
>
```

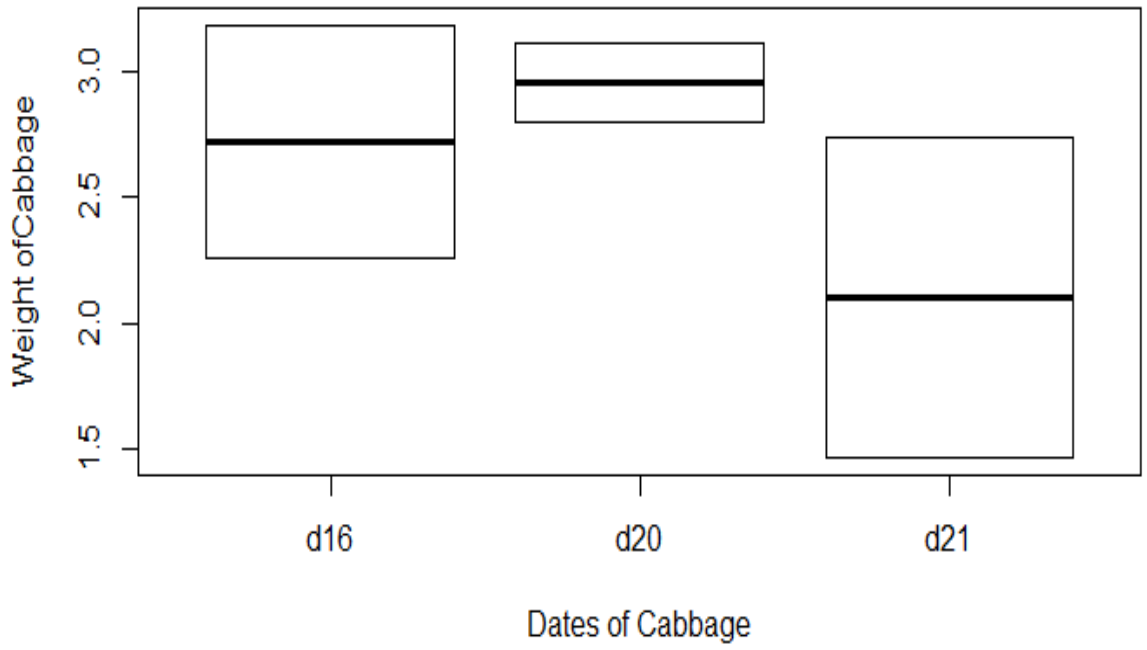


Bar Graph and Box Plot (5th A & B) OTUPUT

```
> library(ggplot2)
> library(gcookbook)
> cabbage_exp
  Cultivar Date Weight        sd  n         se
1      c39  d16   3.18 0.9566144 10 0.30250803
2      c39  d20   2.80 0.2788867 10 0.08819171
3      c39  d21   2.74 0.9834181 10 0.31098410
4      c52  d16   2.26 0.4452215 10 0.14079141
5      c52  d20   3.11 0.7908505 10 0.25008887
6      c52  d21   1.47 0.2110819 10 0.06674995
> ggplot(cabbage_exp,aes(x=Date,y=Weight,fill=Cultivar))+geom_bar(stat="ident
y",position = "dodge")+ geom_text(aes(label=Weight),vjust=1.5,colour="white",
sition = position_dodge(.9),size=3)
>
```

```
> library(ggplot2)
> library(gcookbook)
> cabbage_exp
  Cultivar Date Weight        sd  n         se
1      c39  d16   3.18 0.9566144 10 0.30250803
2      c39  d20   2.80 0.2788867 10 0.08819171
3      c39  d21   2.74 0.9834181 10 0.31098410
4      c52  d16   2.26 0.4452215 10 0.14079141
5      c52  d20   3.11 0.7908505 10 0.25008887
6      c52  d21   1.47 0.2110819 10 0.06674995
> boxplot(Weight~Date,data=cabbage_exp,range=0,ylab="Weight ofCabbage",xlab="Da
tes of Cabbage")
>
```


3.18
2.26
2.8
3.11
2.74
1.47
Weight
0
1
2
3
d16
d20
d21
Date
Cultivar
c39
c52


Weight ofCabbage
1.5
2.0
2.5
3.0
d16
d20
d21
Dates of Cabbage

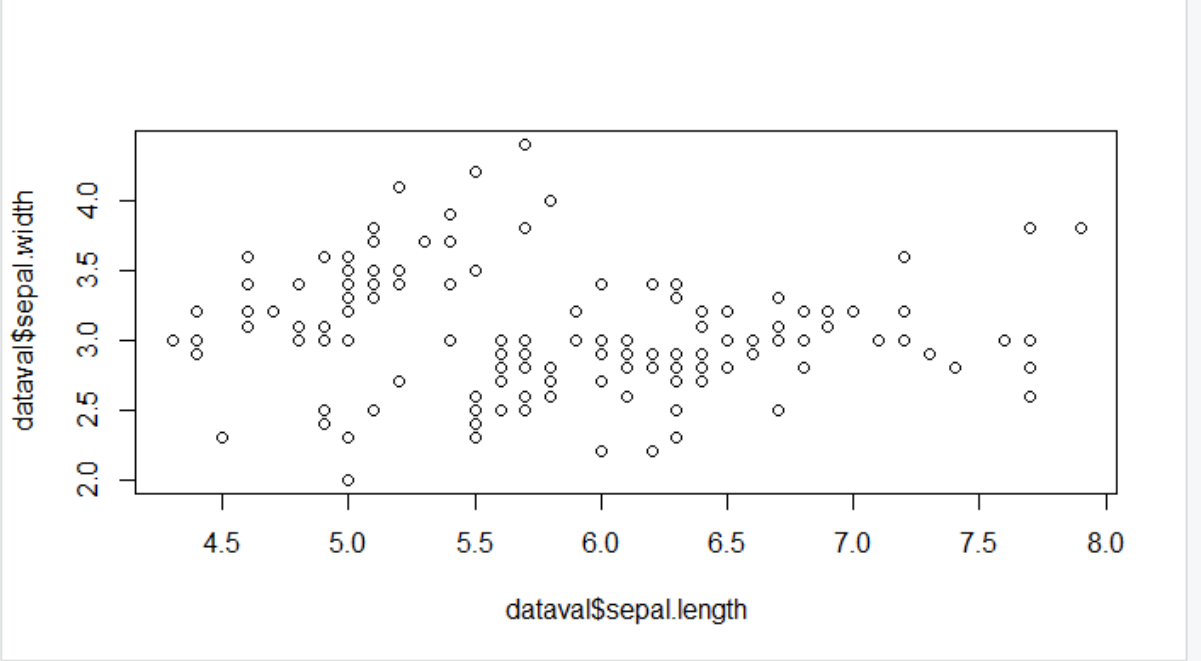
Program(3a & 3b) OTUPUT

```
> intnum=c(10,20,30)
> strv=c("Water","lemon","juice")
> numnum=c(23.3,44.5,89.0)
> list1=list(intnum,strv,numnum)
> names(list1)=c("integer number","string","numeric number")
> list1
$`integer number`
[1] 10 20 30

$string
[1] "Water" "lemon" "juice"

$`numeric number`
[1] 23.3 44.5 89.0

> 
```

```
> matrixbee=matrix(data=c(10,1,37,5,12,8,3,9,6,4,18,9,12,4,6,8,27,6,32,23,12,1
3,16,9,10),nrow=5,ncol=5)
> matrixbee
     [,1] [,2] [,3] [,4] [,5]
[1,]   10    8   18    8   12
[2,]    1    3    9   27   13
[3,]   37    9   12    6   16
[4,]    5    6    4   32    9
[5,]   12    4    6   23   10
> plantnames=list("Thistle","Vipers","GoldenRain","Yellowalfala","blackberry")
> plantframe=as.data.frame(plantnames)
> str(plantframe)
'data.frame':   1 obs. of  5 variables:
 $ X.Thistle.      : Factor w/ 1 level "Thistle": 1
 $ X.Vipers.       : Factor w/ 1 level "Vipers": 1
 $ X.GoldenRain.   : Factor w/ 1 level "GoldenRain": 1
 $ X.Yellowalfala.: Factor w/ 1 level "Yellowalfala": 1
 $ X.blackberry.   : Factor w/ 1 level "blackberry": 1
> plantmatrix=as.matrix(plantframe)
> str(plantmatrix)
 chr [1, 1:5] "Thistle" "Vipers" "GoldenRain" "Yellowalfala" ...
 - attr(*, "dimnames")=List of 2
  ..$ : NULL
  ..$ : chr [1:5] "X.Thistle." "X.Vipers." "X.GoldenRain." "X.Yellowalfala."
 ...
> rownames(matrixbee)=plantmatrix
> matrixbee
            [,1] [,2] [,3] [,4] [,5]
Thistle       10    8   18    8   12
Vipers         1    3    9   27   13
GoldenRain    37    9   12    6   16
Yellowalfala   5    6    4   32    9
blackberry    12    4    6   23   10
> class(matrixbee)
[1] "matrix"
> 
```

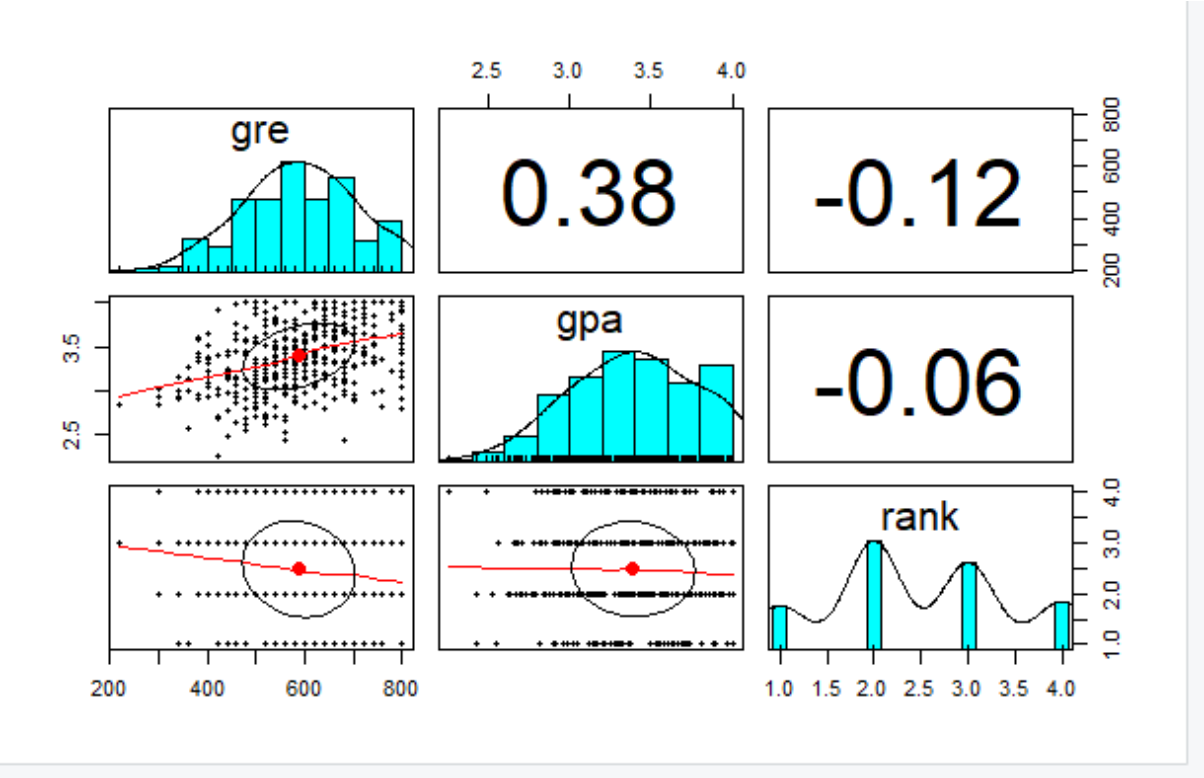
Program 1a & 1b OUTPUT

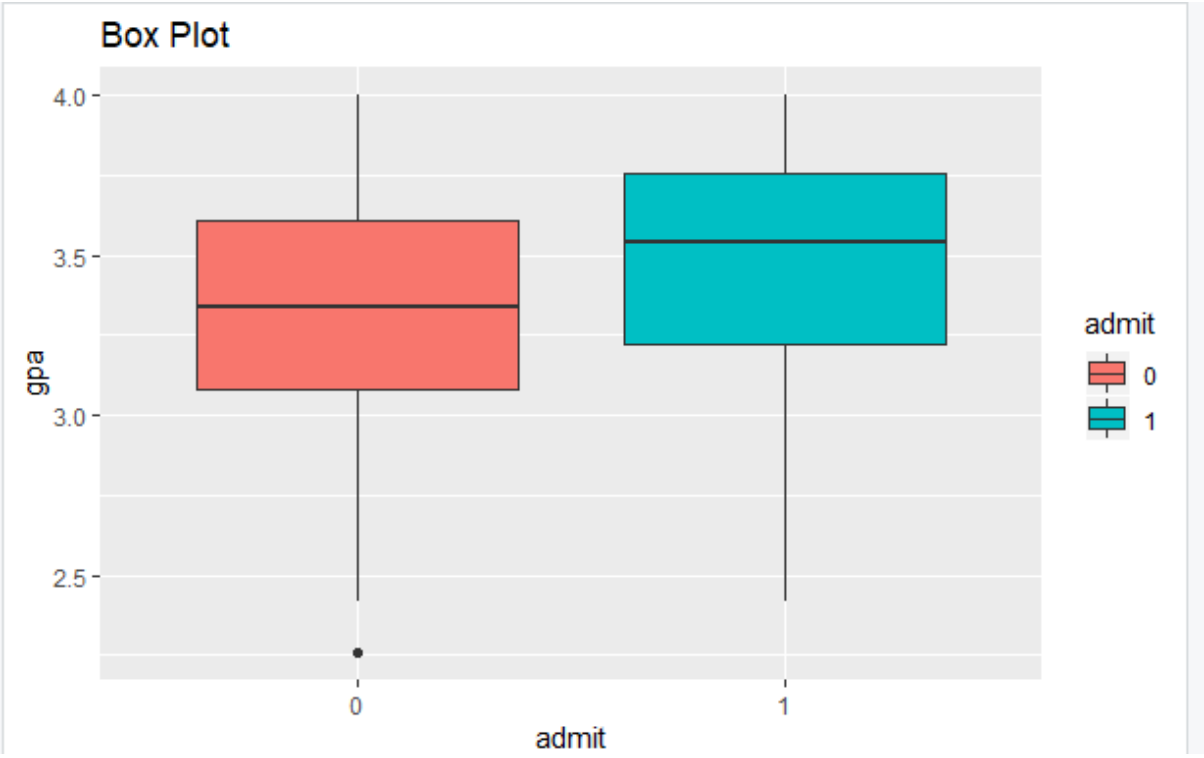
```
> path="C:/Users/jyothiramesh/Desktop/DSRLAB/DATA_SET"
> setwd(path)
> dataval=read.csv("iris.csv")
> dataval
    sepal.length sepal.width petal.length petal.width     variety
1            5.1         3.5          1.4         0.2      Setosa
2            4.9         3.0          1.4         0.2      Setosa
3            4.7         3.2          1.3         0.2      Setosa
4            4.6         3.1          1.5         0.2      Setosa
5            5.0         3.6          1.4         0.2      Setosa
6            5.4         3.9          1.7         0.4      Setosa
7            4.6         3.4          1.4         0.3      Setosa
8            5.0         3.4          1.5         0.2      Setosa
9            4.4         2.9          1.4         0.2      Setosa
10           4.9         3.1          1.5         0.1      Setosa
11           5.4         3.7          1.5         0.2      Setosa
12           4.8         3.4          1.6         0.2      Setosa
13           4.8         3.0          1.4         0.1      Setosa
14           4.3         3.0          1.1         0.1      Setosa
15           5.8         4.0          1.2         0.2      Setosa
16           5.7         4.4          1.5         0.4      Setosa
17           5.4         3.9          1.3         0.4      Setosa
18           5.1         3.5          1.4         0.3      Setosa
19           5.7         3.8          1.7         0.3      Setosa
20           5.1         3.8          1.5         0.3      Setosa
```

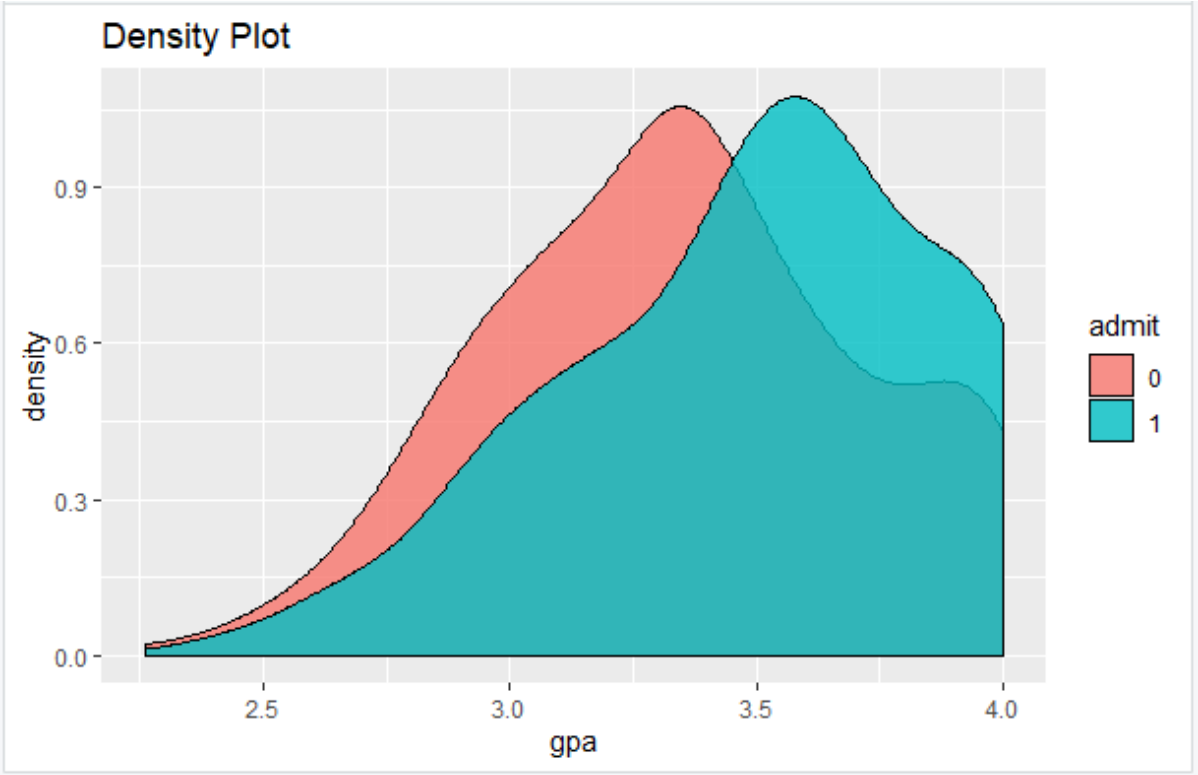


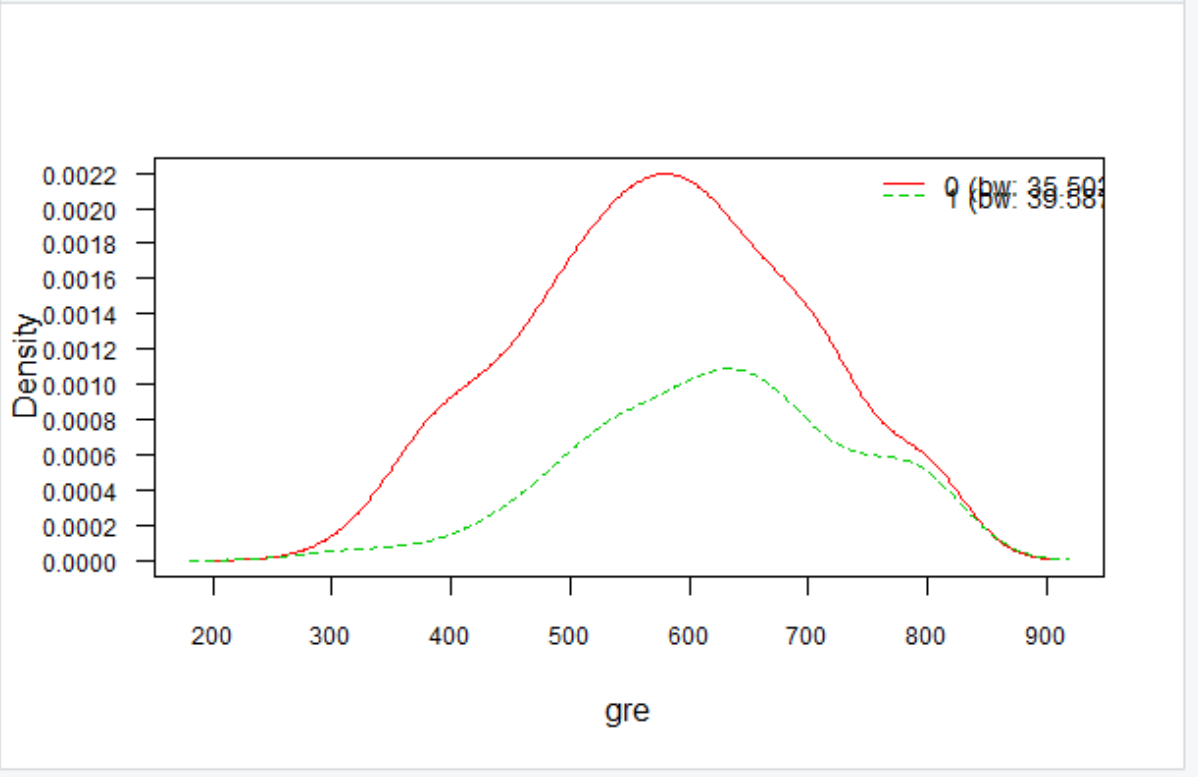

```
> path="C:/Users/jyothiramesh/Desktop/DSRLAB/DATA_SET"
> setwd(path)
> bankdata=read.delim("DT.csv")
> bankdata
    play..outlook.Temperature.Humidity..Wind
1      yes    rainy        cool   normal FALSE
2       no    rainy        cool   normal  TRUE
3      yes overcast         hot     high FALSE
4       no    sunny        mild     high FALSE
5      yes    rainy        cool   normal FALSE
6      yes    sunny        cool   normal FALSE
7      yes    rainy        cool   normal FALSE
8      yes    sunny         hot   normal FALSE
9      yes overcast        mild     high  TRUE
10      no    sunny        mild     high  TRUE
> val_new=vector(mode="numeric",length =length(bankdata$Wind))
> bankdata$num<- seq.int(nrow(bankdata))
> write.table(bankdata,file="file.csv", sep= "\t", row.names=FALSE)
>
```

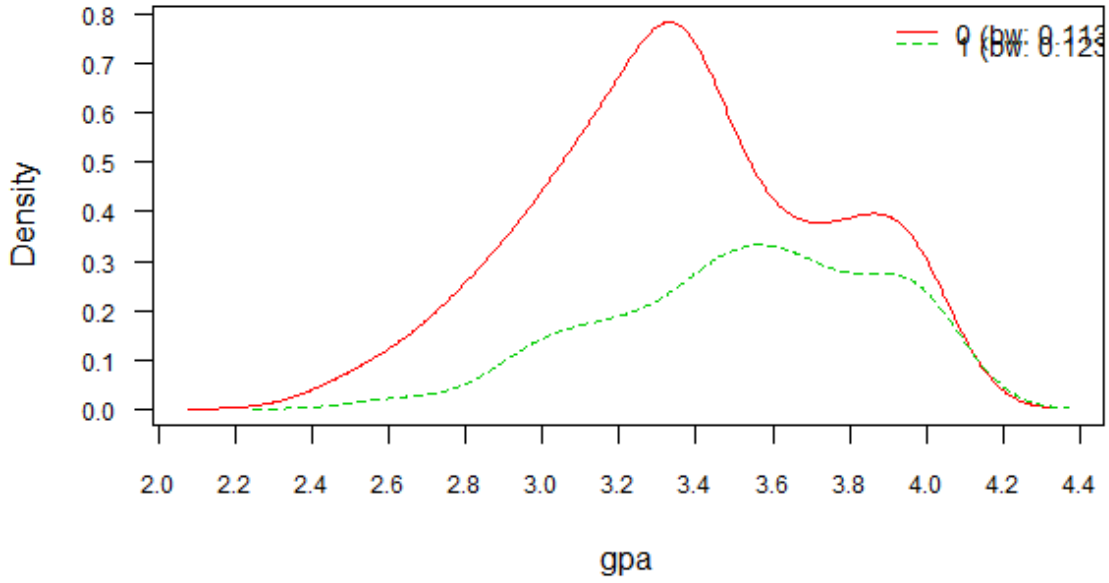
PROGRAM 7th(bayes algorithm)

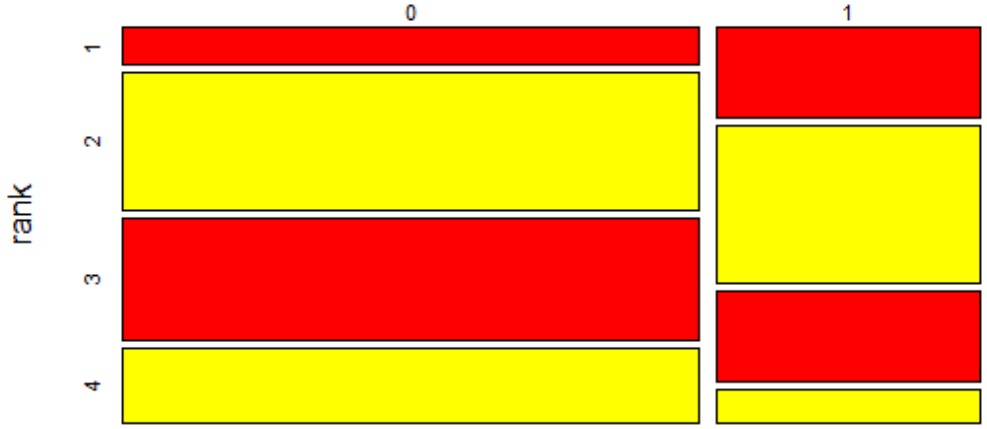





Density Plot
density
0.0
0.3
0.6
0.9
2.5
3.0
3.5
4.0
gpa
admit
0
1


Density
0.0000
0.0002
0.0004
0.0006
0.0008
0.0010
0.0012
0.0014
0.0016
0.0018
0.0020
0.0022
200
300
400
500
600
700
800
900
gre


Density
0.8
0.7
0.6
0.5
0.4
0.3
0.2
0.1
0.0
2.0
2.2
2.4
2.6
2.8
3.0
3.2
3.4
3.6
3.8
4.0
4.2
4.4
gpa


0
1
rank
1
2
3
4

```
'data.frame':   400 obs. of  4 variables:
 $ admit: int  0 1 1 1 0 1 1 0 1 0 ...
 $ gre  : int  380 660 800 640 520 760 560 400 540 700 ...
 $ gpa  : num  3.61 3.67 4 3.19 2.93 3 2.98 3.08 3.39 3.92 ...
 $ rank : int  3 3 1 4 4 2 1 2 3 2 ...
> xtabs(~admit+rank, data = data)
     rank
admit  1  2  3  4
    0 28 97 93 55
    1 33 54 28 12
> data$rank <- as.factor(data$rank)
> data$admit <- as.factor(data$admit)
>
> # Visualization
> pairs.panels(data[-1])
> data %>%
+          ggplot(aes(x=admit, y=gpa, fill = admit)) +
+          geom_boxplot() +
+          ggtitle("Box Plot")
>
> data %>% ggplot(aes(x=gpa, fill = admit)) +
+          geom_density(alpha=0.8, color= 'black') +
+          ggtitle("Density Plot")
>
> # Data Partition
> set.seed(1234)
> ind <- sample(2, nrow(data), replace = T, prob = c(0.8, 0.2))
> train <- data[ind == 1,]
> test <- data[ind == 2,]
>
> # Naive Bayes Model
> model <- naive_bayes(admit ~ ., data = train, usekernel = T)
> model


Call:
        density.default(x = x, na.rm = TRUE)

Data: x (102 obs.);     Bandwidth 'bw' = 0.1234

       x               y
 Min.   :2.25   Min.   :0.0005231
 1st Qu.:2.78   1st Qu.:0.0800747
 Median :3.31   Median :0.4801891
 Mean   :3.31   Mean   :0.4710851
 3rd Qu.:3.84   3rd Qu.:0.8626207
 Max.   :4.37   Max.   :1.0595464

--------------------------------------------------------------------------------
 ::: rank (Categorical)
--------------------------------------------------------------------------------

rank          0          1
   1 0.10313901 0.24509804
   2 0.36771300 0.42156863
   3 0.33183857 0.24509804
   4 0.19730942 0.08823529

--------------------------------------------------------------------------------

>
> train %>%
+          filter(admit == "1") %>%
+          summarise(mean(gre), sd(gre))
  mean(gre) sd(gre)
1  622.9412 110.924
>
```

```
Call:
        density.default(x = x, na.rm = TRUE)

Data: x (102 obs.);     Bandwidth 'bw' = 0.1234

       x               y
 Min.   :2.25   Min.   :0.0005231
 1st Qu.:2.78   1st Qu.:0.0800747
 Median :3.31   Median :0.4801891
 Mean   :3.31   Mean   :0.4710851
 3rd Qu.:3.84   3rd Qu.:0.8626207
 Max.   :4.37   Max.   :1.0595464

------------------------------------------------------------------------------
 ::: rank (Categorical)
------------------------------------------------------------------------------

rank          0          1
   1 0.10313901 0.24509804
   2 0.36771300 0.42156863
   3 0.33183857 0.24509804
   4 0.19730942 0.08823529

------------------------------------------------------------------------------

>
> train %>%
+          filter(admit == "1") %>%
+          summarise(mean(gre), sd(gre))
  mean(gre) sd(gre)
1  622.9412 110.924
>
```

```
> p <- predict(model, train, type = 'prob')
Warning message:
predict.naive_bayes(): More features in the newdata are provided as there are probabil
nd in the tables.
> head(cbind(p, train))
          0         1 admit gre  gpa rank
1 0.8528794 0.1471206     0 380 3.61    3
2 0.5621460 0.4378540     1 660 3.67    3
3 0.2233490 0.7766510     1 800 4.00    1
4 0.8643901 0.1356099     1 640 3.19    4
6 0.6263274 0.3736726     1 760 3.00    2
7 0.5933791 0.4066209     1 560 2.98    1
>
> # Confusion Matrix - train data
> p1 <- predict(model, train)
Warning message:
predict.naive_bayes(): More features in the newdata are provided as there are probabil
nd in the tables.
> (tab1 <- table(p1, train$admit))

p1    0   1
  0 203  69
  1  20  33
> 1 - sum(diag(tab1)) / sum(tab1)
[1] 0.2738462
>
> # Confusion Matrix - test data
> p2 <- predict(model, test)
Warning message:
predict.naive_bayes(): More features in the newdata are provided as there are probabil
nd in the tables.
> (tab2 <- table(p2, test$admit))

p2   0  1
  0 47 20
  1  3  5
> 1 - sum(diag(tab2)) / sum(tab2)
[1] 0.3066667
>
```

```
Call:
        density.default(x = x, na.rm = TRUE)

Data: x (102 obs.);     Bandwidth 'bw' = 39.59

       x                y
 Min.   :181.2   Min.   :1.145e-06
 1st Qu.:365.6   1st Qu.:2.007e-04
 Median :550.0   Median :1.129e-03
 Mean   :550.0   Mean   :1.354e-03
 3rd Qu.:734.4   3rd Qu.:2.375e-03
 Max.   :918.8   Max.   :3.465e-03

------------------------------------------------------------------------------
 ::: gpa::0 (KDE)
------------------------------------------------------------------------------

Call:
        density.default(x = x, na.rm = TRUE)

Data: x (223 obs.);     Bandwidth 'bw' = 0.1134

       x               y
 Min.   :2.080   Min.   :0.0002229
 1st Qu.:2.645   1st Qu.:0.0924939
 Median :3.210   Median :0.4521795
 Mean   :3.210   Mean   :0.4419689
 3rd Qu.:3.775   3rd Qu.:0.6603271
 Max.   :4.340   Max.   :1.1433285

------------------------------------------------------------------------------
 ::: gpa::1 (KDE)
------------------------------------------------------------------------------
```

```
 Call:
naive_bayes.formula(formula = admit ~ ., data = train, usekernel = T)

------------------------------------------------------------------------------

Laplace smoothing: 0

------------------------------------------------------------------------------

 A priori probabilities:

        0         1
0.6861538 0.3138462

------------------------------------------------------------------------------

 Tables:

------------------------------------------------------------------------------
 ::: gre::0 (KDE)
------------------------------------------------------------------------------

Call:
        density.default(x = x, na.rm = TRUE)

Data: x (223 obs.);     Bandwidth 'bw' = 35.5

       x                y
 Min.   :193.5   Min.   :6.010e-07
 1st Qu.:371.7   1st Qu.:2.924e-04
 Median :550.0   Median :1.291e-03
 Mean   :550.0   Mean   :1.401e-03
 3rd Qu.:728.3   3rd Qu.:2.405e-03
 Max.   :906.5   Max.   :3.199e-03

------------------------------------------------------------------------------
 ::: gre::1 (KDE)
------------------------------------------------------------------------------
```

```
> # Confusion Matrix - train data
> p1 <- predict(model, train)
Warning message:
predict.naive_bayes(): More features in the newdata are provided as there are probability tables in the ob
nd in the tables.
> (tab1 <- table(p1, train$admit))

p1    0   1
  0 203  69
  1  20  33
> 1 - sum(diag(tab1)) / sum(tab1)
[1] 0.2738462
>
> # Confusion Matrix - test data
> p2 <- predict(model, test)
Warning message:
predict.naive_bayes(): More features in the newdata are provided as there are probability tables in the ob
nd in the tables.
> (tab2 <- table(p2, test$admit))

p2   0  1
  0 47 20
  1  3  5
> 1 - sum(diag(tab2)) / sum(tab2)
[1] 0.3066667
>
```