**Experiment:01**

**Basics of Language R**

* **Data Types in R:**

1. **Numeric (Double)**

> x <- 10.5 # Numeric (default is double)

> print(x)

[1] 10.5

> class(x)

# Output: "numeric"

[1] "numeric"

2. **Integer**

> y <- 10L # Integer (suffix "L" is used)

> print(y)

[1] 10

> class(y)

# Output: "integer"

[1] "integer"

3. **Character (String)**

> z <- "Hello, R!"

> print(z)

[1] "Hello, R!"

> class(z) # Output: "character"

[1] "character"

4. **Logical (Boolean)**

> a <- TRUE

> b <- FALSE

> print(a)

[1] TRUE

> print(b)

[1] FALSE

> class(a)

# Output: "logical"

[1] "logical"

5. **Complex**

> c <- 3 + 4i

> print(c)

[1] 3+4i

> class(c)

# Output: "complex"

1. "complex"

* **Vectors in R**

1. **Numeric vector**

> num\_vec <- c(1, 2, 3, 4, 5)

> print(num\_vec)

[1] 1 2 3 4 5

> class(num\_vec)

# Output: "numeric"

[1] "numeric"

1. **Character vector**

> char\_vec <- c("apple", "banana", "cherry")

> print(char\_vec)

[1] "apple" "banana" "cherry"

> class(char\_vec)

# Output: "character"

[1] "character"

* **List in R**

> ID = c(1,2,3,4)

> emp.name =c("Man","Rag","Sha","Din")

> num.emp = 4

> emp.list = list(ID, emp.name,num.emp)

> print(emp.list)

[[1]]

[1] 1 2 3 4

[[2]]

[1] "Man" "Rag" "Sha" "Din"

[[3]]

[1] 4

* **Data Frames in R**

> vec1 = c(1,2,3)

> vec2 = c("R","Scilab","Java")

> vec3 = c("For prototyping",

+ "For prototyping","For Scaleup")

> df = data.frame(vec1,vec2,vec3)

> print(df)

vec1 vec2 vec3

1 1 R For prototyping

2 2 Scilab For prototyping

3 3 Java For Scaleup

* **Control structures in R**

1. **If statement**

> x <- 10

> if (x > 5) {

+ print("x is greater than 5")

+ }

[1] "x is greater than 5"

1. **If-else if-else**

> x=6

> if(x>7)

{

x=x+1

}

else if(x>8)

{

x=x+2

}else

{

x=x+3}

> x

[1] 9

1. **If-else statement**

> x <- 3

> if (x > 5) {

+ print("x is greater than 5")

+ } else {

+ print("x is less than or equal to 5")

+ }

[1] "x is less than or equal to 5"

1. **For Loop with if break statement**

for (i in 1:10) {

+ if (i == 6) {

+ break # Exit loop when i is 6

+ }

+ print(i)

+ }

[1] 1

[1] 2

[1] 3

[1] 4

[1] 5

* **Matrix Operation in R**

> A= matrix(c(1,2,3,4,5,6,7,8,9), nrow =3 ,ncol=3 , byrow=TRUE)

> A

[,1] [,2] [,3]

[1,] 1 2 3

[2,] 4 5 6

[3,] 7 8 9

**Basic Graphs in R**

**Input:**

library(ggplot2)

data <- mtcars

**# Scatter Plot**

ggplot(mtcars, aes(x=hp, y=mpg)) + geom\_point()

**# Boxplot**

ggplot(mtcars, aes(x=factor(cyl), y=mpg)) + geom\_boxplot()

**# Bar Chart**

ggplot(mtcars, aes(x=factor(cyl))) + geom\_bar()

**# Line Chart**

ggplot(mtcars, aes(x=hp, y=mpg)) + geom\_line()

# Hexbin Plot

ggplot(mtcars, aes(x=hp, y=mpg)) + geom\_hex()

**#dot plot**

ggplot(mtcars, aes(x=mpg)) + geom\_dotplot(binwidth=0.5)

**#Histogram , Density and Rug Plot**

income <- rlnorm(4000, meanlog = 4, sdlog = 0.7)

summary(income)

income <- 1000\*income

summary(income)

# plot the histogram

hist(income, breaks=500, xlab="Income", main="Histogram of Income")

# density plot

plot(density(log10(income), adjust=0.5),

main="Distribution of Income (log10 scale)")

# add rug to the density plot

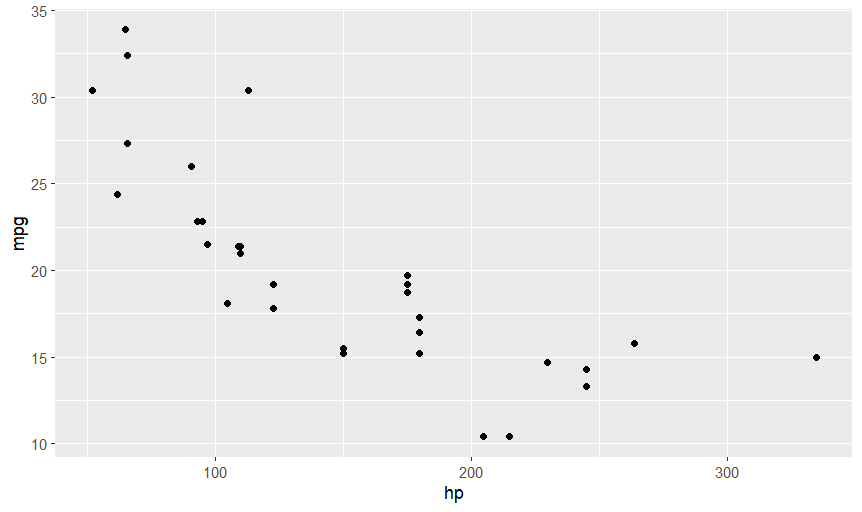
rug(log10(income))

#ScatterPlot Mattrix

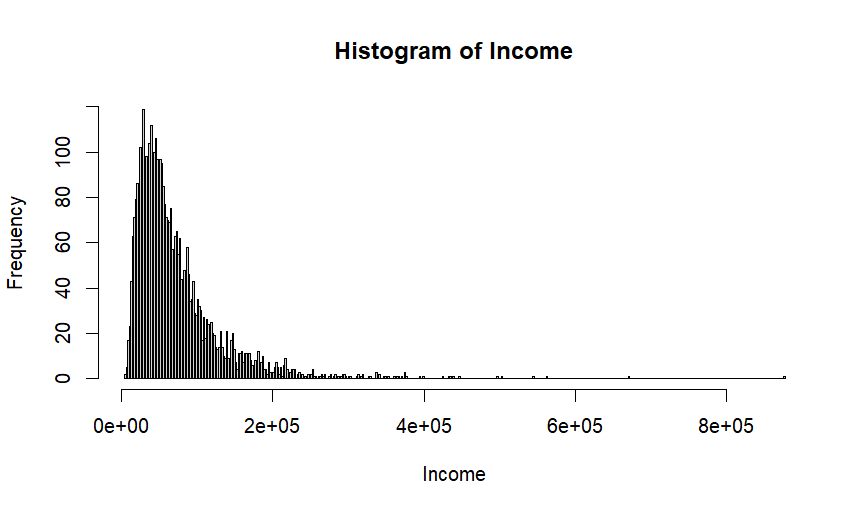
ggpairs(data[, c("mpg", "hp","wt")])

**Output:**

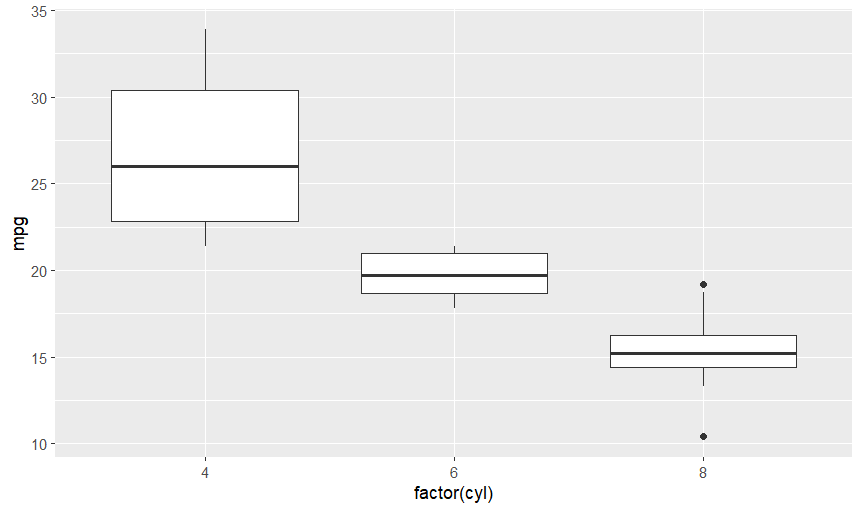
1. Scatter Plot



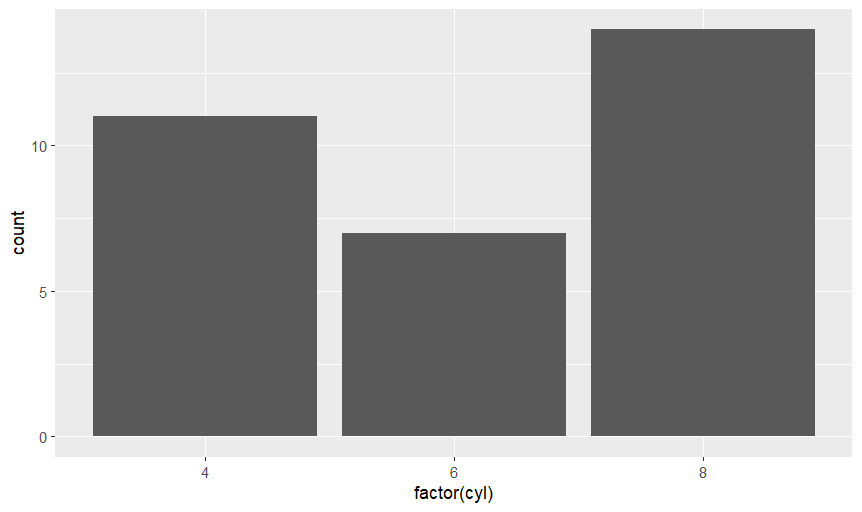
1. Histogram



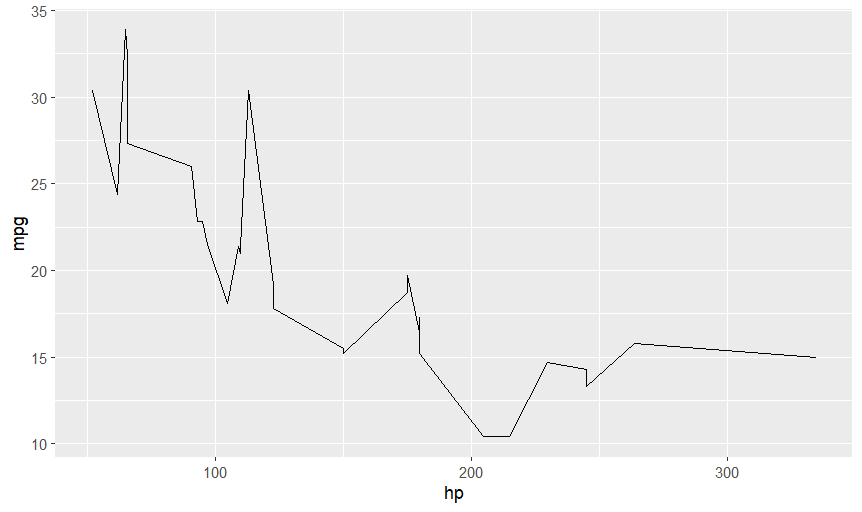
1. Box Plot



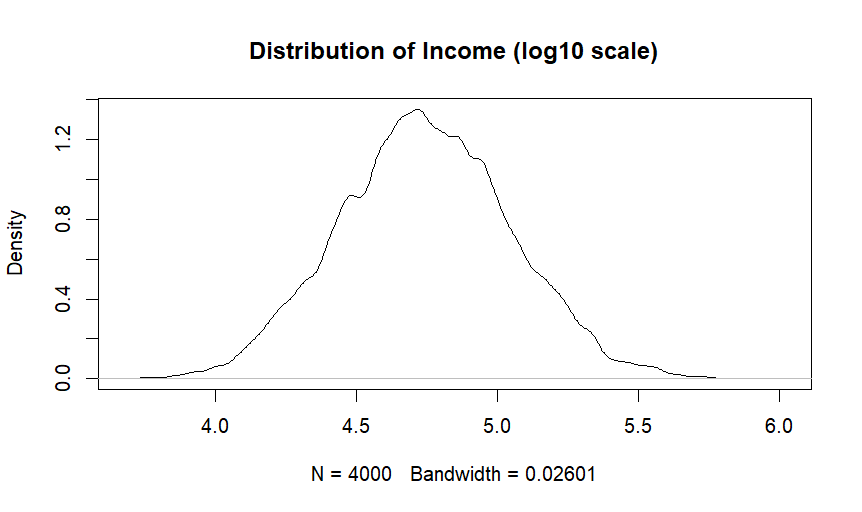
1. Bar Chart



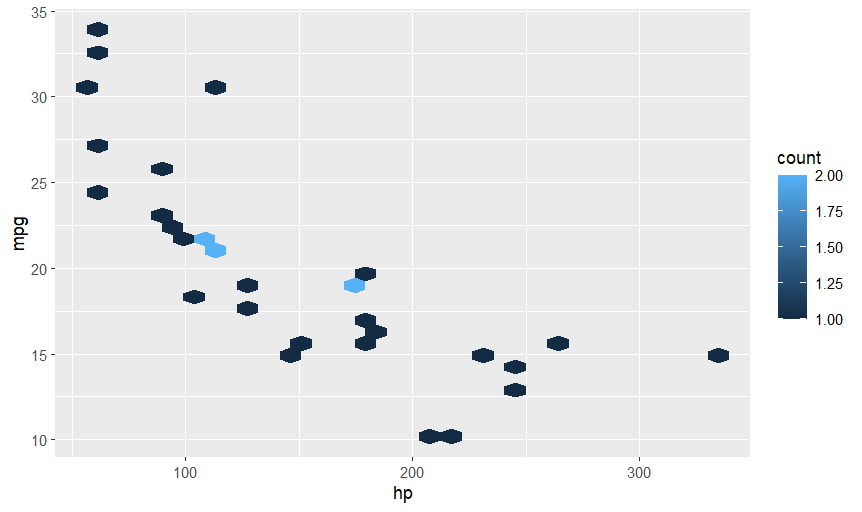
5 .Line Chart



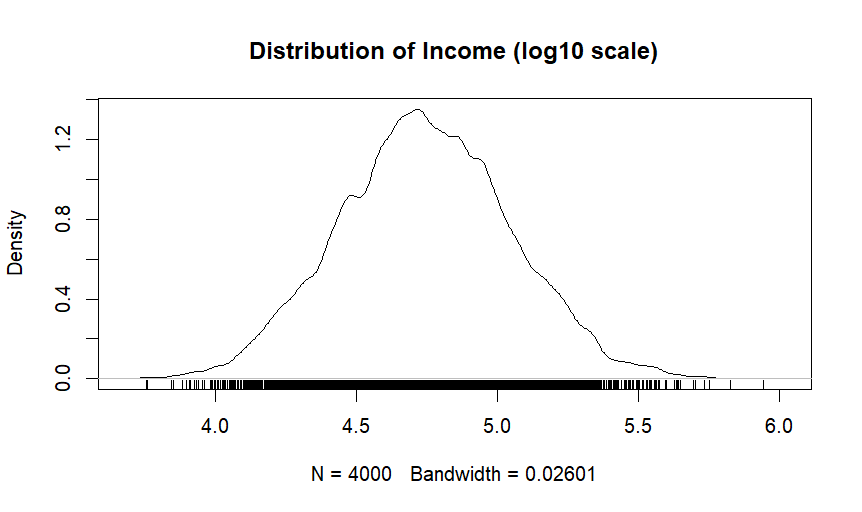
6.Density Plot



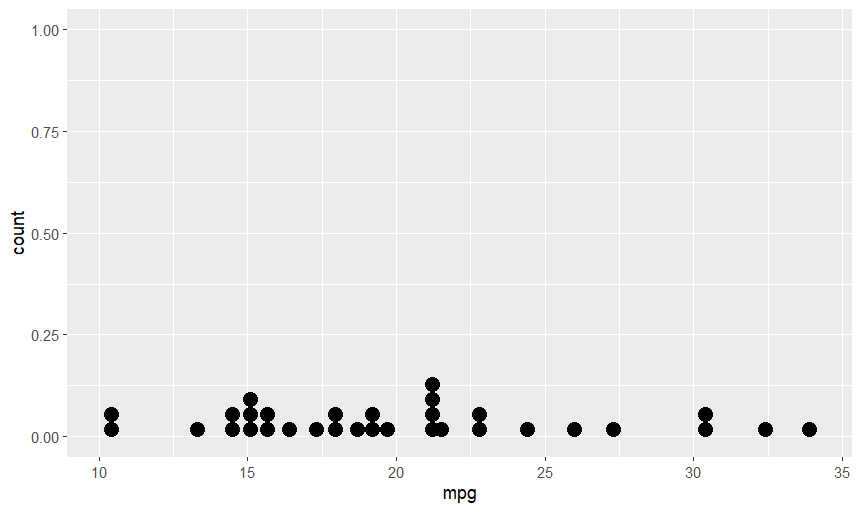
7.Hexabin Plot



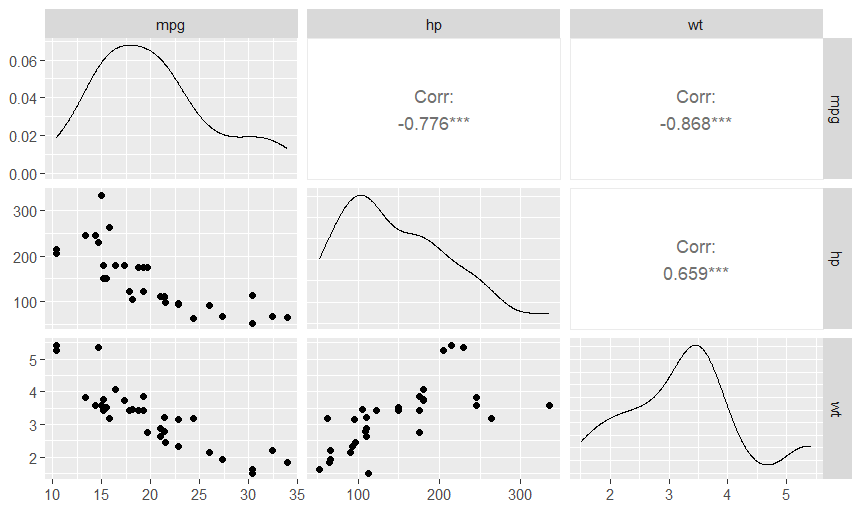
8.Rug Plot



9.Dot Plot



10.ScatterPlot Mattrix



**Experiment:02**

**Input:**

library(plyr)

library(ggplot2)

library(cluster)

library(lattice)

library(graphics)

library(grid)

library(gridExtra)

#import the student grades

grade\_input =as.data.frame(read.csv("C:/Users/Sonal\_Patil/Documents/DSV/datasets/grades\_km\_input.csv"))

kmdata\_orig = as.matrix(grade\_input[,c("Student","English","Math","Science")])

kmdata <- kmdata\_orig[,2:4]

kmdata[1:10,] wss <- numeric(15)

For (k in 1:15) wss[k] <- sum(kmeans(kmdata, centers=k,nstart=25)$withinss)

plot(1:15, wss, type="b", xlab="Number of Clusters", ylab="Within Sum of Squares")

Km = kmeans(kmdata,3, nstart=25) km

c(wss[3], sum(km$withinss))

Df = as.data.frame(kmdata\_orig[,2:4])

df$cluster = factor(km$cluster)

centers=as.data.frame(km$centers)

g1= ggplot(data=df, aes(x=English, y=Math, color=cluster )) +

geom\_point() + theme(legend.position="right") +

geom\_point(data=centers,aes(x=English,y=Math, color=as.factor(c(1,2,3))),size=10, alpha=.3,show\_guide=FALSE) g2 =ggplot(data=df, aes(x=English, y=Science, color=cluster))+

geom\_point() + geom\_point(data=centers, aes(x=English,y=Science, color=as.factor(c(1,2,3))),

size=10, alpha=.3, show\_guide=FALSE)

g3=ggplot(data=df, aes(x=Math, y=Science, color=cluster ))+ geom\_point()+ geom\_point(data=centers, aes(x=Math,y=Science, color=as.factor(c(1,2,3))), size=10, alpha=.3, show\_guide=FALSE)

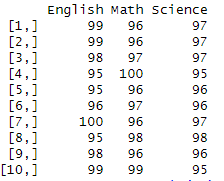
Tmp = ggplot\_gtable(ggplot\_build(g1))

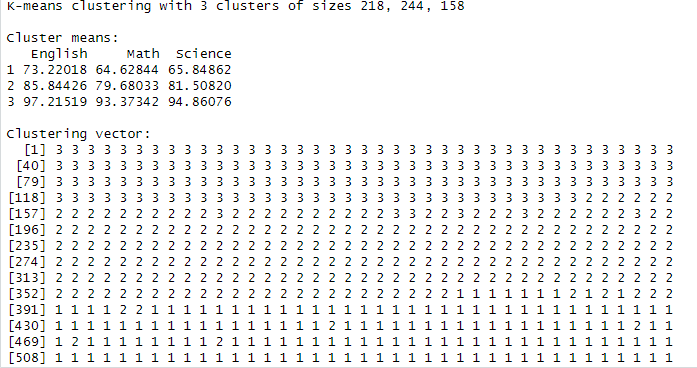
library(grid)

library(gridExtra)

grid.arrange(g1,g2,g3,ncol=1,top ="High School Student Cluster Analysis")

**Output:**





Within cluster sum of squares by cluster:

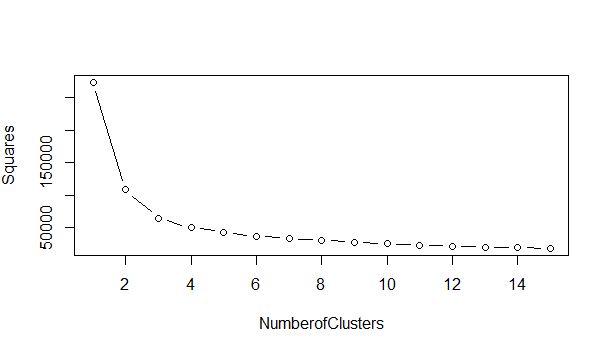
[1] 6692.589 22984.131 34806.339

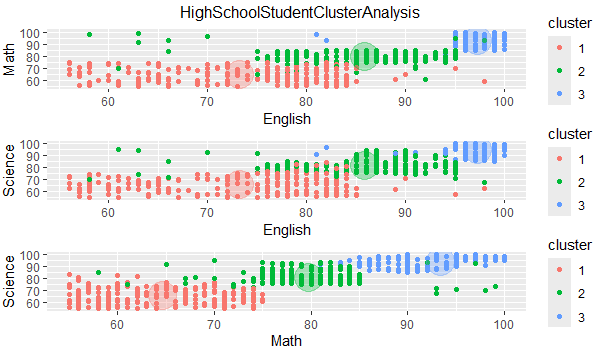
(between\_SS / total\_SS = 76.5 %)

Available components:

[1] "cluster" "centers" "totss" "withinss" "tot.withinss"

[6] "betweenss" "size" "iter" "ifault"

 [1] 64483.06 64483.06



**Experiment:03**

**Input:**

income\_input = as.data.frame(read.csv("C:/Users/Sonal\_Patil/Documents/DSV/datasets/income.csv"))

income\_input[1:10,]

summary(income\_input)

library(lattice)

splom(~income\_input[c(2:5)], groups=NULL, data=income\_input,

axis.line.tck =0,

axis.text.alpha=0)

results<- lm (Income ~ Age+Education+Gender,income\_input)

summary(results)

results2 <- lm (Income ~ Age + Education, income\_input)

summary(results2)

results3 <- lm(Income ~ Age + Education,

+ Alabama,

+ Alaska,

+ Arizona,

,

,

,

+ WestVirginia,

+ Wisconsin,

income\_input)

confint(results2, level = .95)

Age <- 41

Education <- 12

new\_pt <- data.frame(Age, Education)

conf\_int\_pt <- predict(results2,new\_pt,level=.95,interval="confidence")

conf\_int\_pt

pred\_int\_pt <- predict(results2,new\_pt,level=.95,interval="prediction")

pred\_int\_pt

with(results2, {

plot(fitted.values, residuals,ylim=c(-40,40) )

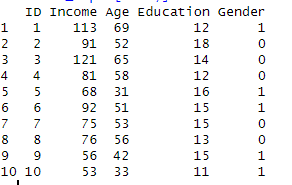
points(c(min(fitted.values),max(fitted.values)),

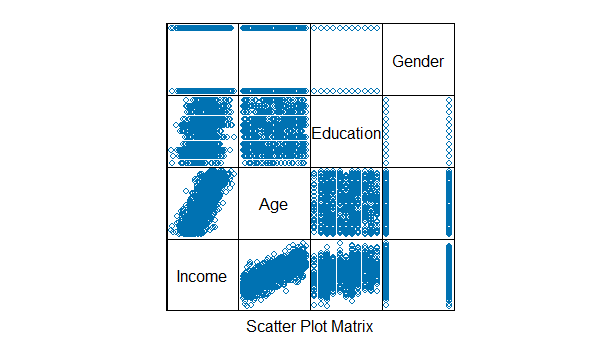
c(0,0), type = "1")})

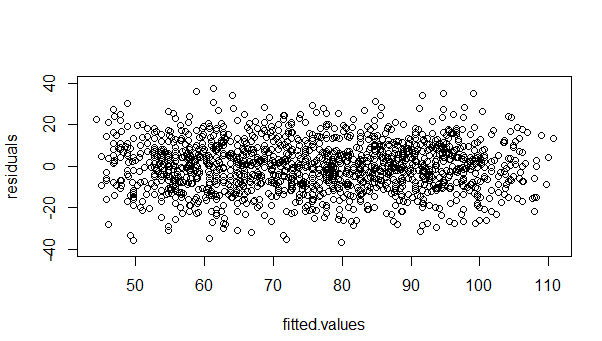
hist(results2$residuals, main="")

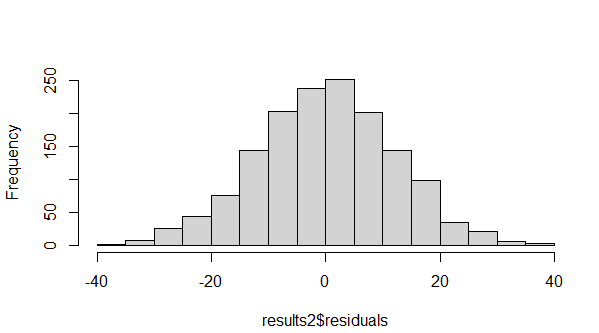
qqnorm(results2$residuals, ylab="Residuals", main="")

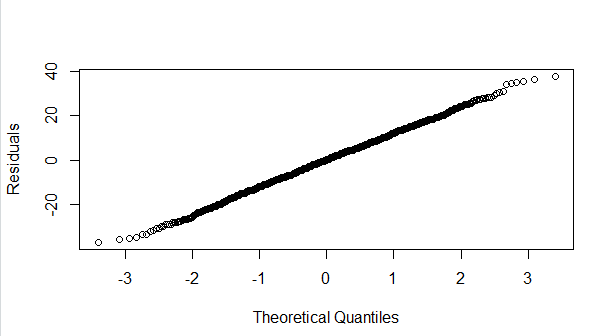
qqline(results2$residuals)

**Output:**









**Experiment:04**

**Input:**

# Section 6.2 Logistic Regression

churn\_input = as.data.frame( read.csv("C:/Users/Sonal\_Patil/Documents/DSV/datasets/churn.csv") )

head(churn\_input)

sum(churn\_input$Churned)

Churn\_logistic1 <- glm (Churned~Age + Married + Cust\_years + Churned\_contacts,

data=churn\_input, family=binomial(link="logit"))

summary(Churn\_logistic1)

Churn\_logistic2 <- glm (Churned~Age + Married + Churned\_contacts,

data=churn\_input, family=binomial(link="logit"))

summary(Churn\_logistic2)

Churn\_logistic3 <- glm (Churned~Age + Churned\_contacts,

data=churn\_input, family=binomial(link="logit"))

summary(Churn\_logistic3)

# Deviance and the Log Likelihood Ratio Test

# Using the residual deviances from Churn\_logistics2 and Churn\_logistic3

# determine the signficance of the computed test statistic

summary(Churn\_logistic2)

pchisq(.9 , 1, lower=FALSE)

# Receiver Operating Characteristic (ROC) Curve

install.packages("ROCR") #install, if necessary

library(ROCR)

pred = predict(Churn\_logistic3, type="response")

predObj = prediction(pred, churn\_input$Churned )

rocObj = performance(predObj, measure="tpr", x.measure="fpr")

aucObj = performance(predObj, measure="auc")

plot(rocObj, main = paste("Area under the curve:", round(aucObj@y.values[[1]] ,4)))

# extract the alpha(threshold), FPR, and TPR values from rocObj

alpha <- round(as.numeric(unlist(rocObj@alpha.values)),4)

fpr <- round(as.numeric(unlist(rocObj@x.values)),4)

tpr <- round(as.numeric(unlist(rocObj@y.values)),4)

# adjust margins and plot TPR and FPR

par(mar = c(5,5,2,5))

plot(alpha,tpr, xlab="Threshold", xlim=c(0,1), ylab="True positive rate", type="l")

par(new="True")

plot(alpha,fpr, xlab="", ylab="", axes=F, xlim=c(0,1), type="l" )

axis(side=4)

mtext(side=4, line=3, "False positive rate")

text(0.18,0.18,"FPR")

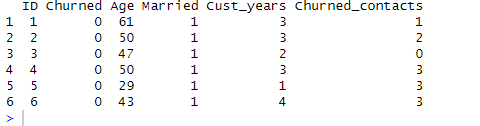
text(0.58,0.58,"TPR")

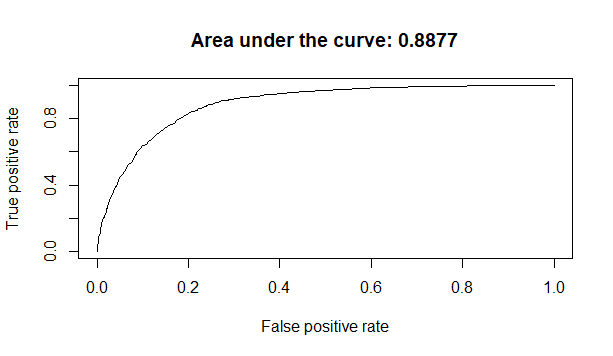
i <- which(round(alpha,2) == .5)

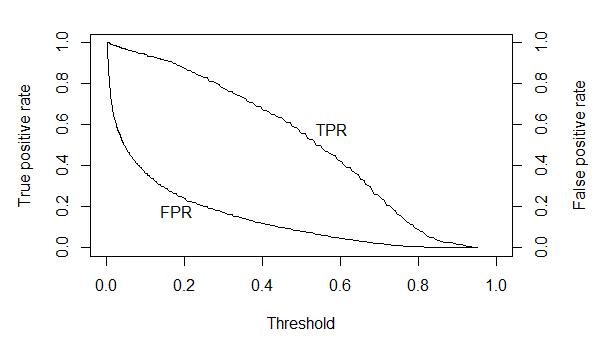
paste("Threshold=" , (alpha[i]) , " TPR=" , tpr[i] , " FPR=" , fpr[i])

i <- which(round(alpha,2) == .15)

paste("Threshold=" , (alpha[i]) , " TPR=" , tpr[i] , " FPR=" , fpr[i])

**Output:**





**Experiment:05**

**Input:**

install.packages("rpart.plot") # install package rpart.plot

library("rpart") # load libraries

library("rpart.plot")

play\_decision <- read.table("C:/Users/Sonal\_Patil/Document/DSV/datasets/bank-sample.csv",header=TRUE,sep=",")

play\_decision

summary(play\_decision)

x<- sort(runif(1000))

y<-data.frame(x=x,y=-x\*log2(x)-(1-x)\*log2(1-x))

plot(y,type="l",xlab="P(X=1)",ylab=expression("H"["X"]))

grid()

Fit <- rpart(subscribed~job+marital+education+default+housing+loan+contact+poutcome,

method="class",

data=play\_decision,

control=rpart.control(minsplit=1),

parms=list(split='information'))

summary(fit)

rpart.plot(fit, type=4, extra=2, clip.right.labs=FALSE,

varlen=0, faclen=3)

Newdata <-data.frame(job="retired",

marital="married",

education="secondary",

default="no",

housing="yes",

loan="no",

contact="cellular",

duration=598,

poutcome="unknown"

)

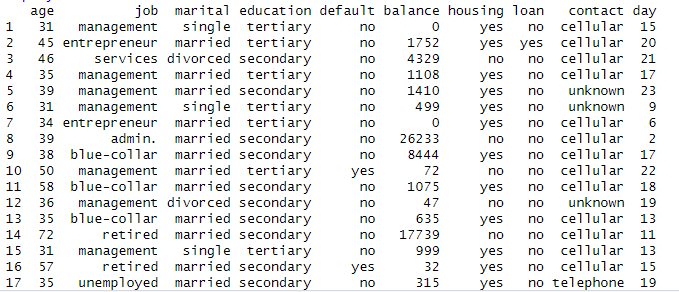
newdata

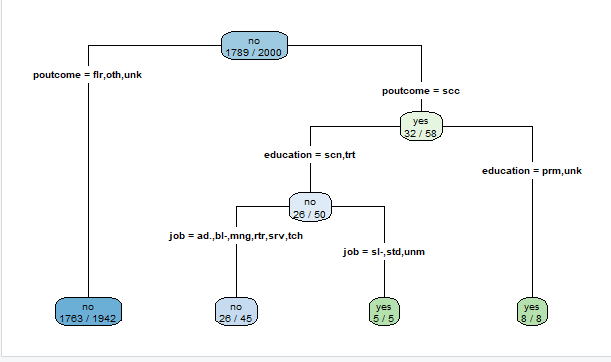
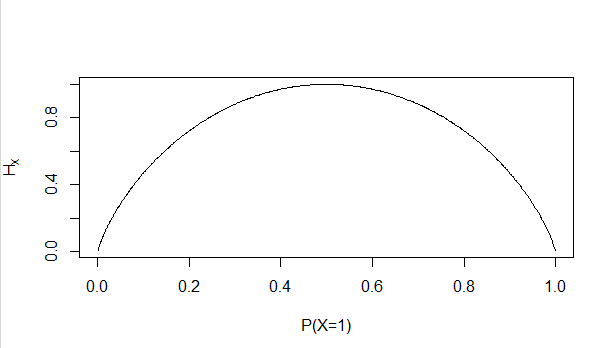
predict(fit,newdata=newdata,type=c("class"))

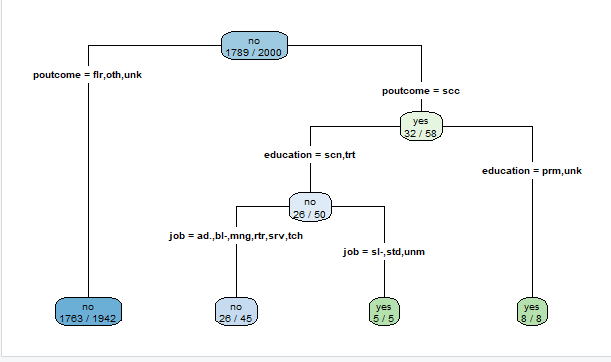
library("rpart") # load libraries

library("rpart.plot")

**Output:**







**Experiment:06**

**Input:**

# install some packages

install.packages("rpart.plot")

library("rpart")

library("rpart.plot")

## Read the data

setwd("C:/Users/Sonal\_Patil/Documents/DSV/datasets")

banktrain <- read.table("bank-sample.csv",header=TRUE,sep=",")

## drop a few columns to simplify the model

drops<-c("balance", "day", "campaign", "pdays", "previous", "month")

banktrain <- banktrain [,!(names(banktrain) %in% drops)]

summary(banktrain)

## testing set

# banktest <- read.table("bank-sample-test.csv",header=TRUE,sep=",")

# banktest <- banktest [,!(names(banktest) %in% drops)]

# summary(banktest)

## manually compute the conditional probabilities

maritalCounts <- table(banktrain[,c("subscribed", "marital")])

maritalCounts <- maritalCounts/rowSums(maritalCounts)

maritalCounts

maritalCounts["yes","divorced"]

jobCounts <- table(banktrain[,c("subscribed", "job")])

jobCounts <- jobCounts/rowSums(jobCounts)

jobCounts

educationCounts <- table(banktrain[,c("subscribed", "education")])

educationCounts <- educationCounts/rowSums(educationCounts)

educationCounts

defaultCounts <- table(banktrain[,c("subscribed", "default")])

defaultCounts <- defaultCounts/rowSums(defaultCounts)

defaultCounts

housingCounts <- table(banktrain[,c("subscribed", "housing")])

housingCounts <- housingCounts/rowSums(housingCounts)

housingCounts

loanCounts <- table(banktrain[,c("subscribed", "loan")])

loanCounts <- loanCounts/rowSums(loanCounts)

loanCounts

contactCounts <- table(banktrain[,c("subscribed", "contact")])

contactCounts <- contactCounts/rowSums(contactCounts)

contactCounts

poutcomeCounts <- table(banktrain[,c("subscribed", "poutcome")])

poutcomeCounts <- poutcomeCounts/rowSums(poutcomeCounts)

poutcomeCounts

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

# section 7.2.5 Na?ve Bayes in R

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

install.packages("e1071") # install package e1071

library(e1071) # load the library

sample <- read.table("C:/Users/Sonal\_Patil/Documents/DSV/datasets/sample1.csv",header=TRUE,sep=",")

print(sample)

# read the data into a table from the file

sample <- read.table("C:/Users/Sonal\_Patil/Documents/DSV/datasets/sample1.csv",header=TRUE,sep=",")

# define the data frames for the NB classifier

traindata <- as.data.frame(sample[1:14,])

testdata <- as.data.frame(sample[15,])

traindata

testdata

tprior <- table(traindata$Enrolls)

tprior

tprior <- tprior/sum(tprior)

tprior

ageCounts <- table(traindata[,c("Enrolls", "Age")])

ageCounts

ageCounts <- ageCounts/rowSums(ageCounts)

ageCounts

incomeCounts <- table(traindata[,c("Enrolls", "Income")])

incomeCounts <- incomeCounts/rowSums(incomeCounts)

incomeCounts

jsCounts <- table(traindata[,c("Enrolls", "JobSatisfaction")])

jsCounts <- jsCounts/rowSums(jsCounts)

jsCounts

desireCounts <- table(traindata[,c("Enrolls", "Desire")])

desireCounts <- desireCounts/rowSums(desireCounts)

desireCounts

prob\_yes <-

ageCounts["Yes",testdata[,c("Age")]]\*

incomeCounts["Yes",testdata[,c("Income")]]\*

jsCounts["Yes",testdata[,c("JobSatisfaction")]]\*

desireCounts["Yes",testdata[,c("Desire")]]\*

tprior["Yes"]

prob\_no <-

ageCounts["No",testdata[,c("Age")]]\*

incomeCounts["No",testdata[,c("Income")]]\*

jsCounts["No",testdata[,c("JobSatisfaction")]]\*

desireCounts["No",testdata[,c("Desire")]]\*

tprior["No"]

prob\_yes

prob\_no

max(prob\_yes,prob\_no)

model <- naiveBayes(Enrolls ~ Age+Income+JobSatisfaction+Desire, traindata)

# display model

model

# predict with testdata

results <- predict (model,testdata)

# display results

results

# use the NB classifier with Laplace smoothing

model1 = naiveBayes(Enrolls ~., traindata, laplace=.01)

# display model

model1

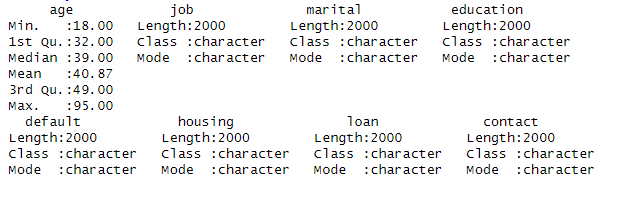
# predict with testdata

results1 <- predict (model1,testdata)

# display results

results1

**Output:**



> maritalCounts["yes","divorced"]

[1] 0.1232227

> jobCounts <- table(banktrain[,c("subscribed", "job")])

> jobCounts <- jobCounts/rowSums(jobCounts)

> jobCounts

job

subscribed admin. blue-collar entrepreneur housemaid management retired

no 0.118501956 0.223588597 0.034097261 0.032420347 0.210173281 0.042481833

yes 0.109004739 0.165876777 0.042654028 0.023696682 0.222748815 0.075829384

job

subscribed self-employed services student technician unemployed

no 0.036892119 0.086640581 0.013974287 0.169368362 0.027389603

yes 0.014218009 0.061611374 0.052132701 0.170616114 0.052132701

job

subscribed unknown

no 0.004471772

yes 0.009478673

education

subscribed primary secondary tertiary unknown

no 0.17216322 0.50978200 0.27613192 0.04192286

yes 0.12796209 0.46445498 0.33175355 0.07582938

default

subscribed no yes

no 0.979318055 0.020681945

yes 0.990521327 0.009478673

housing

subscribed no yes

no 0.4348798 0.5651202

yes 0.6540284 0.3459716

loan

subscribed no yes

no 0.8541084 0.1458916

yes 0.8957346 0.1042654

contact

subscribed cellular telephone unknown

no 0.61878144 0.06987144 0.31134712

yes 0.85308057 0.05213270 0.09478673

poutcome

subscribed failure other success unknown

no 0.10564561 0.03633315 0.01453326 0.84348798

yes 0.09952607 0.06635071 0.15165877 0.68246445

Age Income JobSatisfaction Desire Enrolls

1 <=30 High No Fair No

2 <=30 High No Excellent No

3 31 to 40 High No Fair Yes

4 >40 Medium No Fair Yes

5 >40 Low Yes Fair Yes

6 >40 Low Yes Excellent No

7 31 to 40 Low Yes Excellent Yes

8 <=30 Medium No Fair No

9 <=30 Low Yes Fair Yes

10 >40 Medium Yes Fair Yes

11 <=30 Medium Yes Excellent Yes

12 31 to 40 Medium No Excellent Yes

13 31 to 40 High Yes Fair Yes

14 >40 Medium No Excellent No

15 <=30 Medium Yes Fair

Age Income JobSatisfaction Desire Enrolls

1 <=30 High No Fair No

2 <=30 High No Excellent No

3 31 to 40 High No Fair Yes

4 >40 Medium No Fair Yes

5 >40 Low Yes Fair Yes

6 >40 Low Yes Excellent No

7 31 to 40 Low Yes Excellent Yes

8 <=30 Medium No Fair No

9 <=30 Low Yes Fair Yes

10 >40 Medium Yes Fair Yes

11 <=30 Medium Yes Excellent Yes

12 31 to 40 Medium No Excellent Yes

13 31 to 40 High Yes Fair Yes

14 >40 Medium No Excellent No

Age Income JobSatisfaction Desire Enrolls

15 <=30 Medium Yes Fair

No Yes

5 9

No Yes

0.3571429 0.6428571

Age

Enrolls <=30 >40 31 to 40

No 3 2 0

Yes 2 3 4

Age

Enrolls <=30 >40 31 to 40

No 0.6000000 0.4000000 0.0000000

Yes 0.2222222 0.3333333 0.4444444

Income

Enrolls High Low Medium

No 0.4000000 0.2000000 0.4000000

Yes 0.2222222 0.3333333 0.4444444

JobSatisfaction

Enrolls No Yes

No 0.8000000 0.2000000

Yes 0.3333333 0.6666667

Desire

Enrolls Excellent Fair

No 0.6000000 0.4000000

Yes 0.3333333 0.6666667

Yes

0.02821869

No

0.006857143

[1] 0.02821869

Naive Bayes Classifier for Discrete Predictors

Call:

naiveBayes.default(x = X, y = Y, laplace = laplace)

A-priori probabilities:

Y

No Yes

0.3571429 0.6428571

Conditional probabilities:

Age

Y <=30 >40 31 to 40

No 0.6000000 0.4000000 0.0000000

Yes 0.2222222 0.3333333 0.4444444

Income

Y High Low Medium

No 0.4000000 0.2000000 0.4000000

Yes 0.2222222 0.3333333 0.4444444

JobSatisfaction

Y No Yes

No 0.8000000 0.2000000

Yes 0.3333333 0.6666667

Desire

Y Excellent Fair

No 0.6000000 0.4000000

Yes 0.3333333 0.6666667

[1] Yes

Levels: No Yes

Naive Bayes Classifier for Discrete Predictors

Call:

naiveBayes.default(x = X, y = Y, laplace = laplace)

A-priori probabilities:

Y

No Yes

0.3571429 0.6428571

Conditional probabilities:

Age

Y <=30 >40 31 to 40

No 0.6020000 0.4020000 0.0020000

Yes 0.2233333 0.3344444 0.4455556

Income

Y High Low Medium

No 0.4020000 0.2020000 0.4020000

Yes 0.2233333 0.3344444 0.4455556

JobSatisfaction

Y No Yes

No 0.8020000 0.2020000

Yes 0.3344444 0.6677778

Desire

Y Excellent Fair

No 0.6020000 0.4020000

Yes 0.3344444 0.6677778

[1] Yes

Levels: No Yes