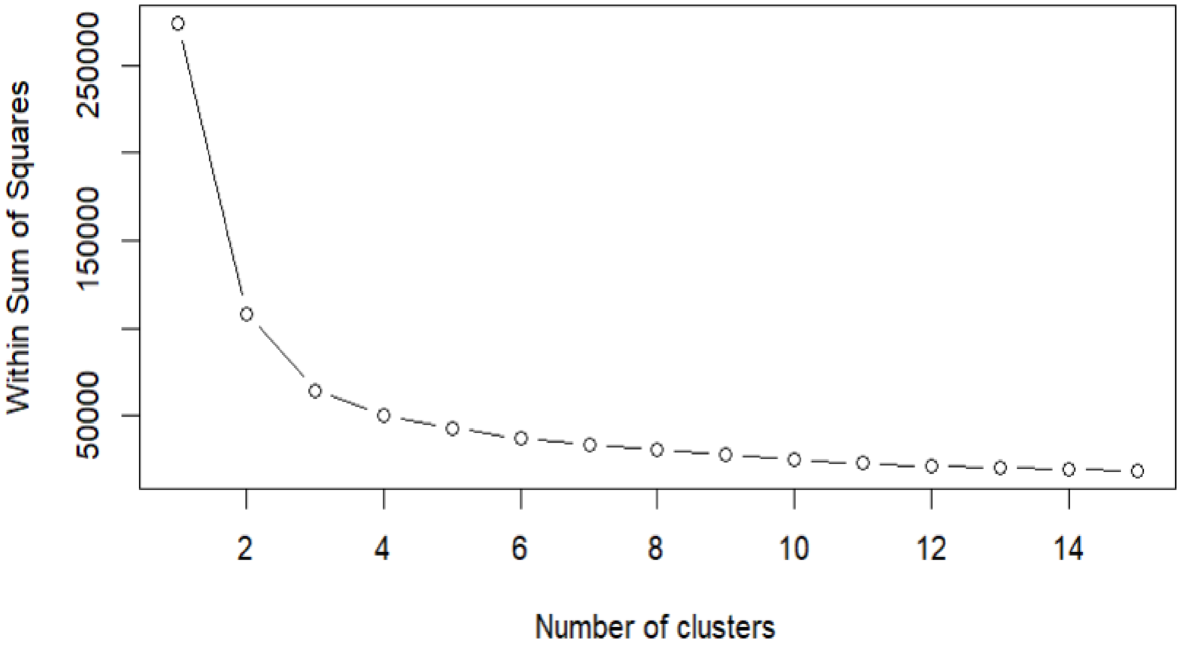
KMEANS

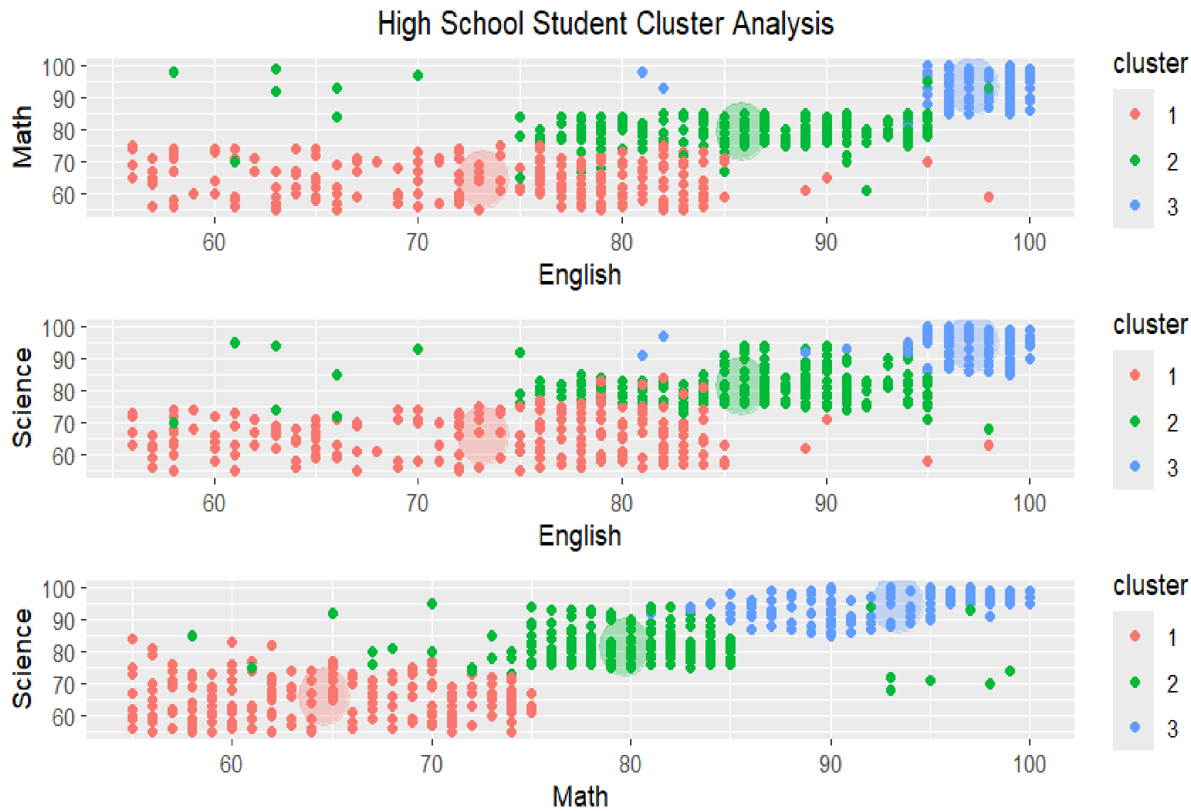
```
library(plyr)
library(ggplot2)
library(cluster)
library(lattice)
library(graphics)
library(grid)
library(gridExtra)
#import the student grades
grade_input=as.data.frame(read.csv("D:/DSA_DATASETS/datasets/grades_km_input.csv"))
kmdata_orig=as.matrix(grade_input[,c("Student","English","Math","Science")])



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$cluster)
km$cluster
centers=as.data.frame(km$centers)
ggplot()
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.legend = 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.legend=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.legend = 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 Sum of Squares
50000
150000
250000
2
4
6
8
10
12
14
Number of clusters



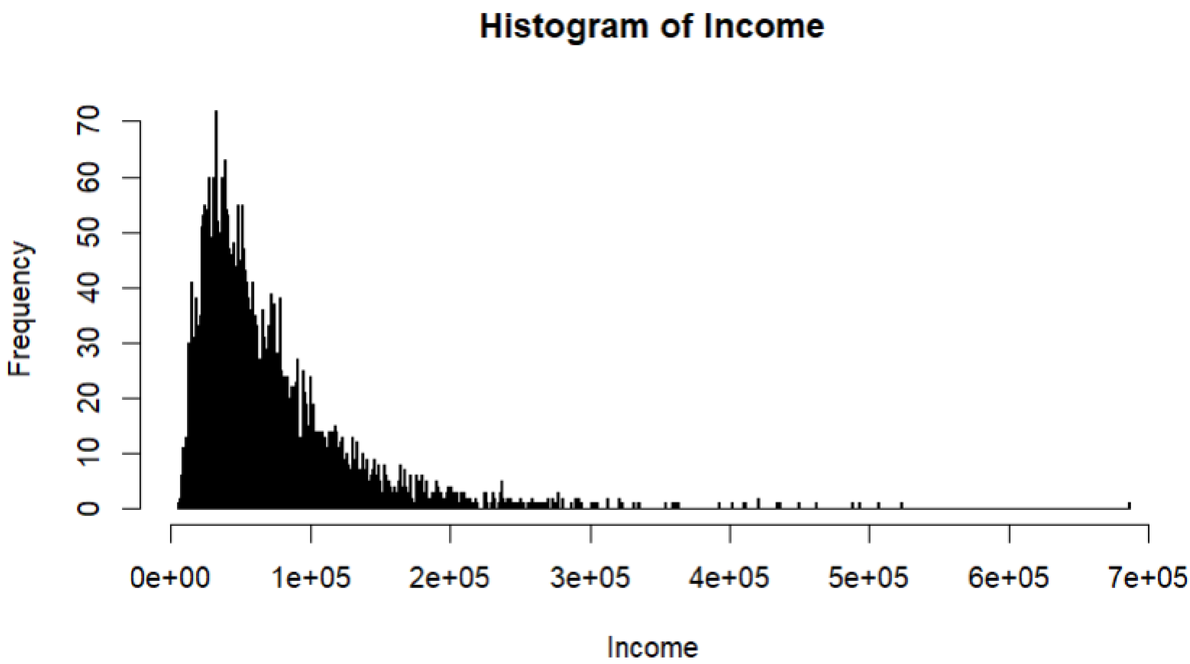
High School Student Cluster Analysis
Math
English
Science
English
Science
Math
cluster
1
2
3

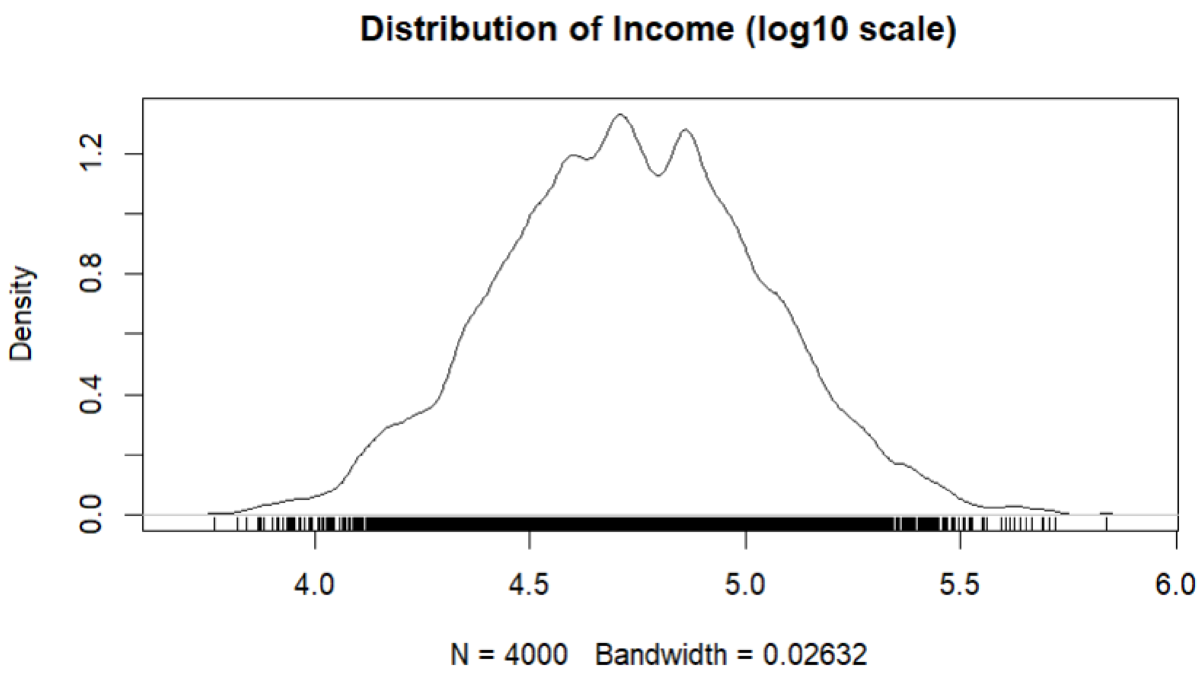
Plots

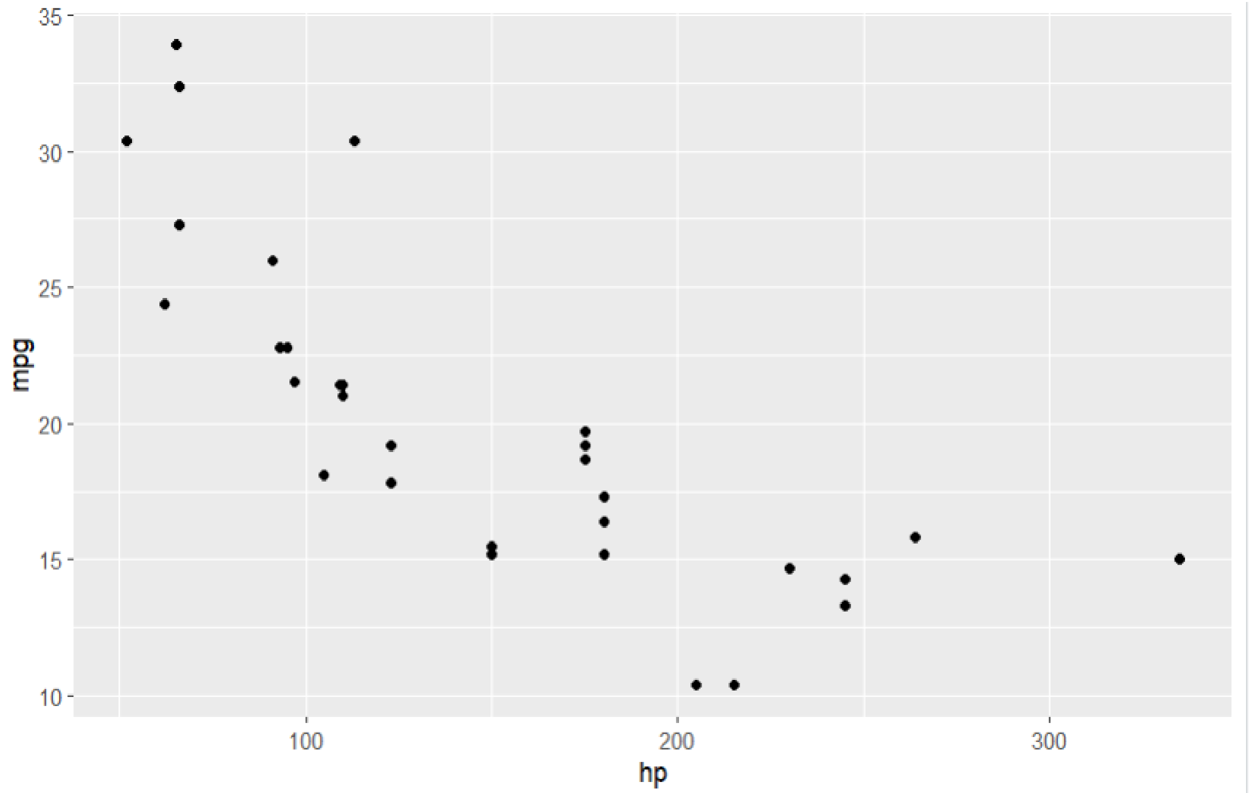
```
# Load required libraries
library(ggplot2)
library(GGally)

# Load dataset
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 (requires hexbin package)
if (!requireNamespace("hexbin", quietly = TRUE)) install.packages("hexbin")
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)
```

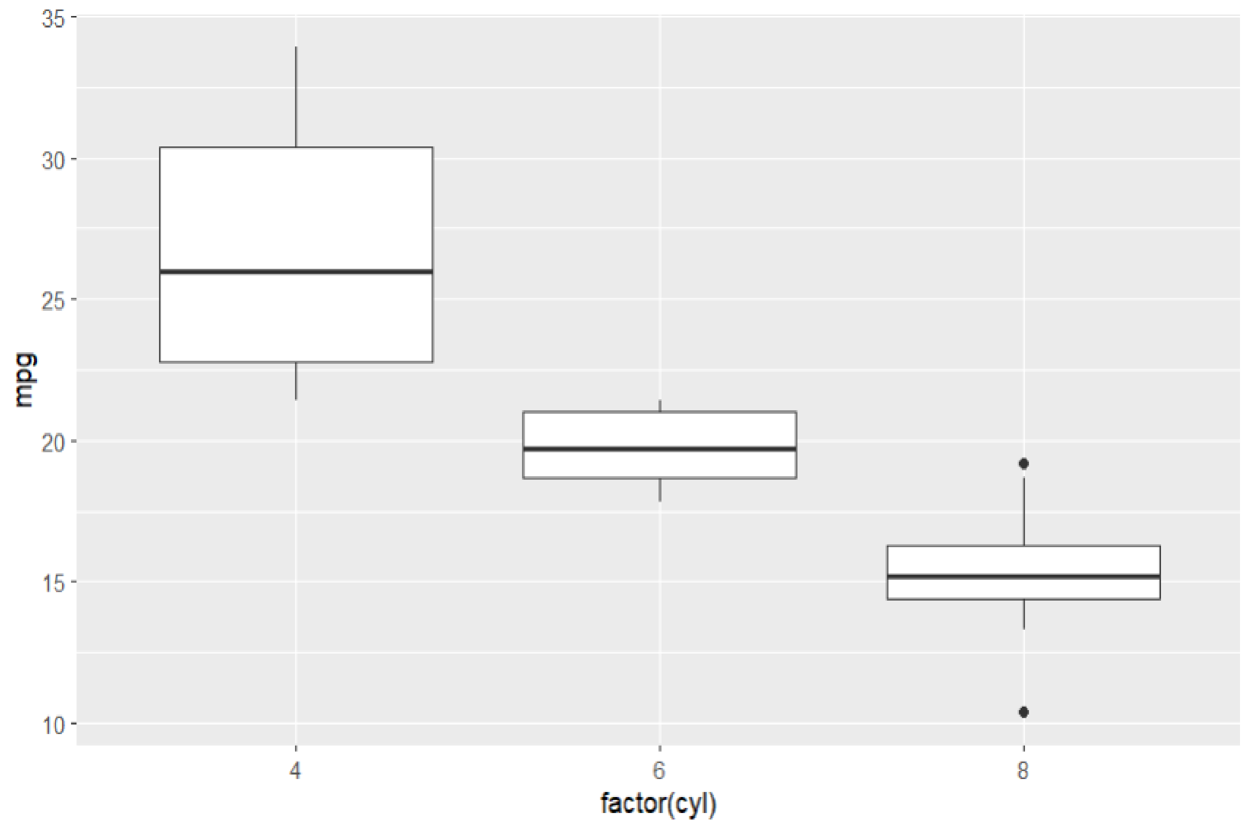
```
summary(income)

# Convert income to thousands
income <- 1000 * income
summary(income)

# 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))

# Scatter Plot Matrix using GGally
ggpairs(data[, c("mpg", "hp", "wt")])
```

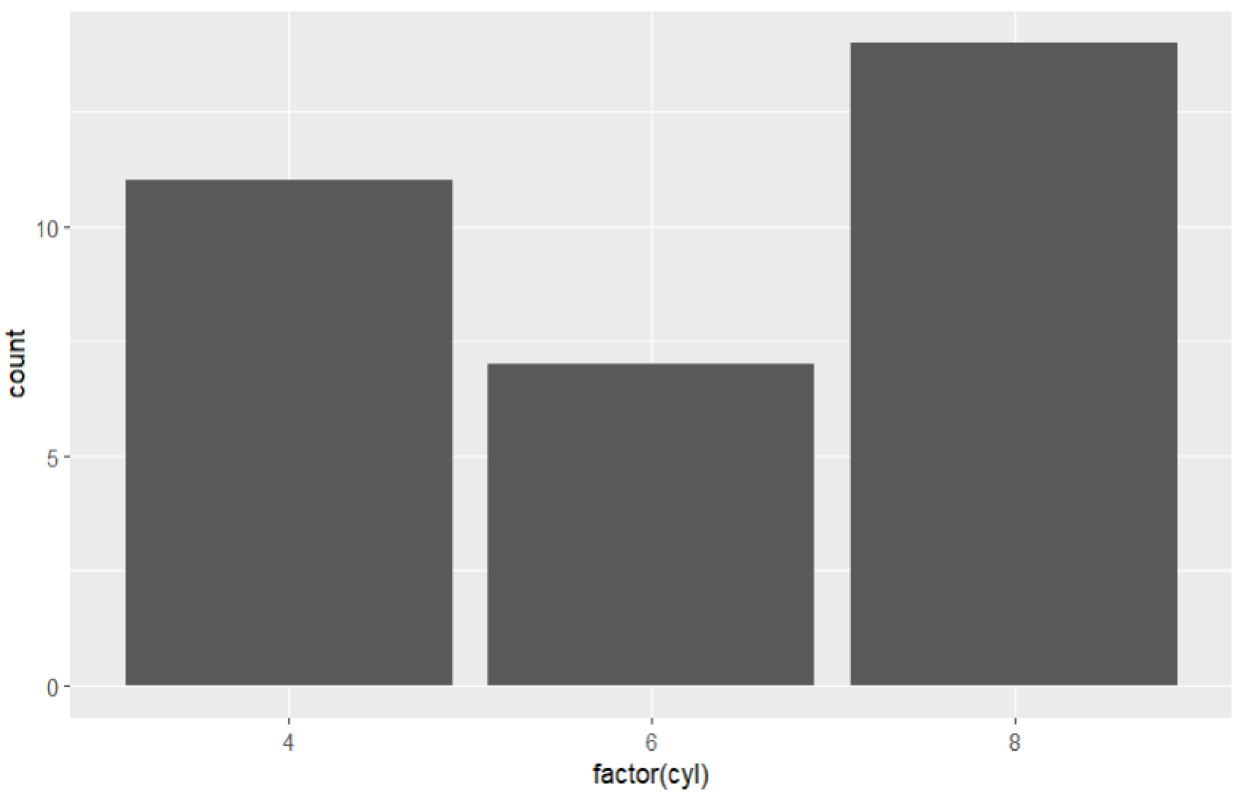
OUTPUTS:

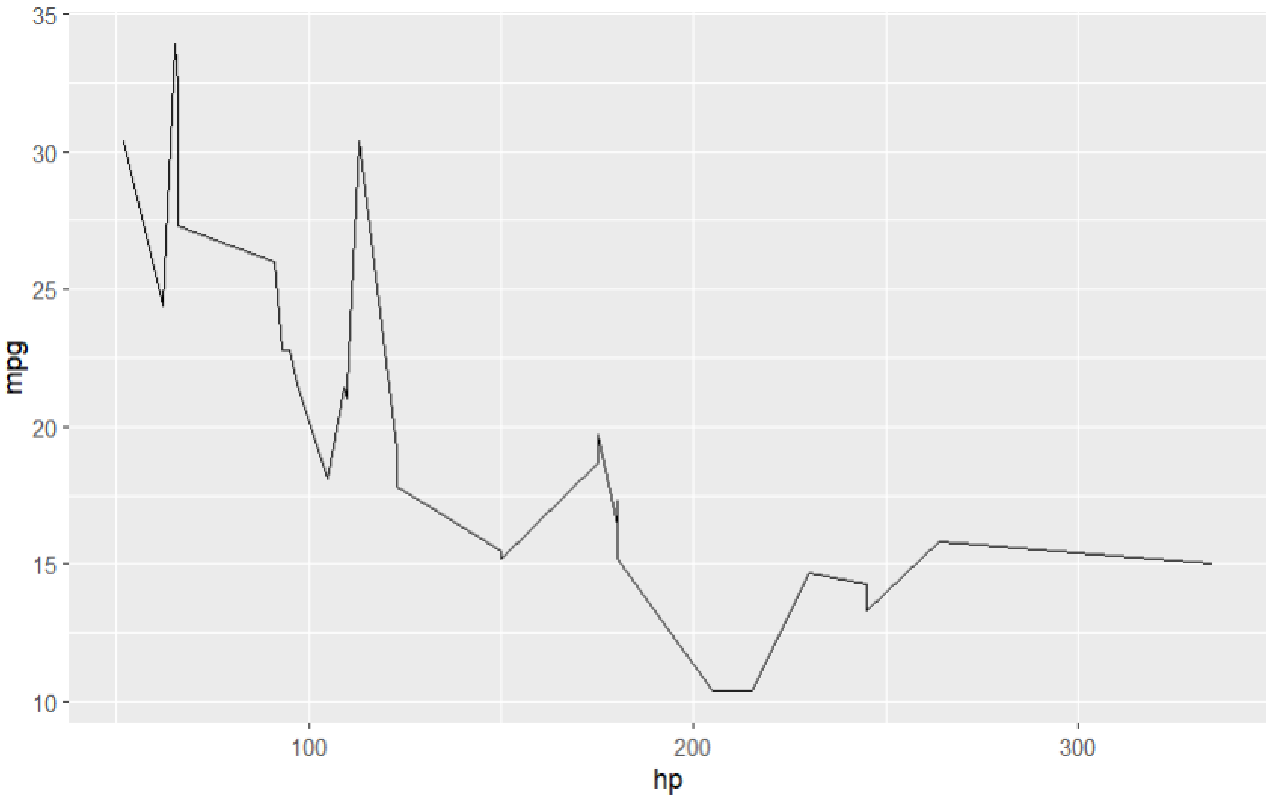

Histogram of Income
Frequency
0
10
20
30
40
50
60
70
0e+00
1e+05
2e+05
3e+05
4e+05
5e+05
6e+05
7e+05
Income

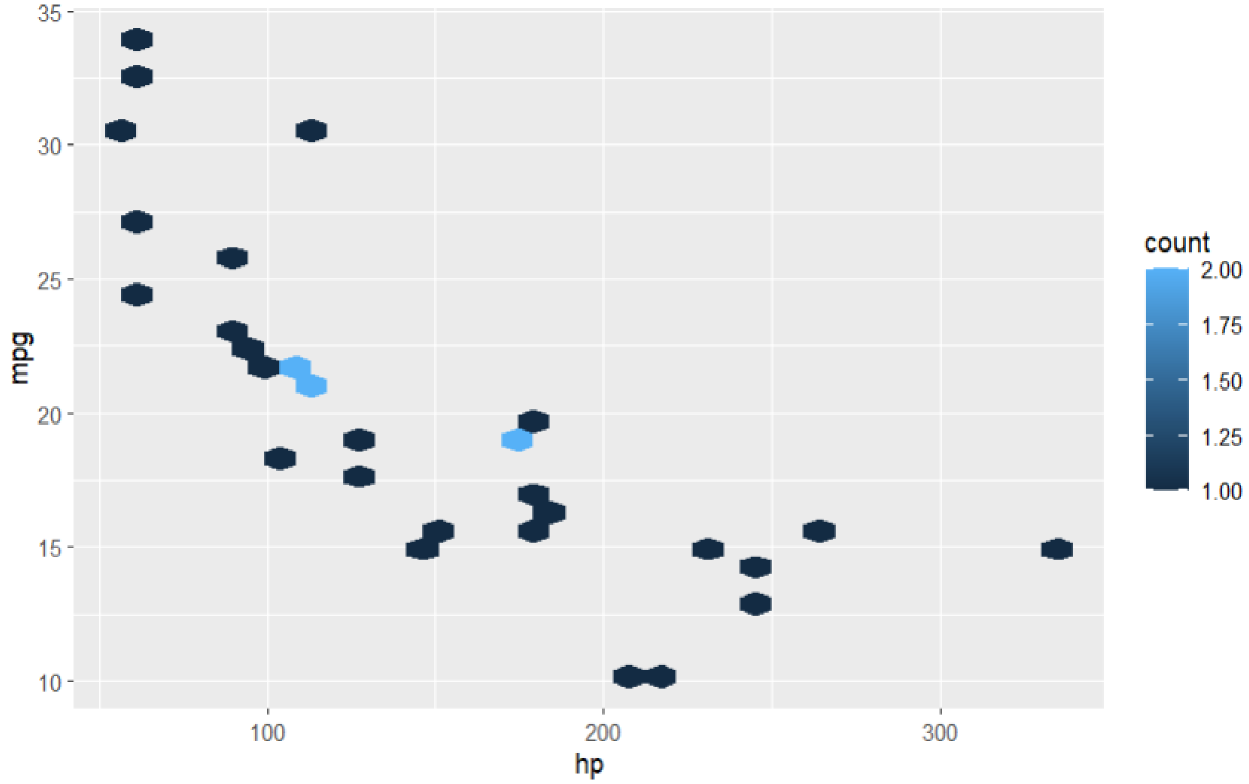

Distribution of Income (log10 scale)
Density
0.0
0.4
0.8
1.2
4.0
4.5
5.0
5.5
6.0
N = 4000 Bandwidth = 0.02632

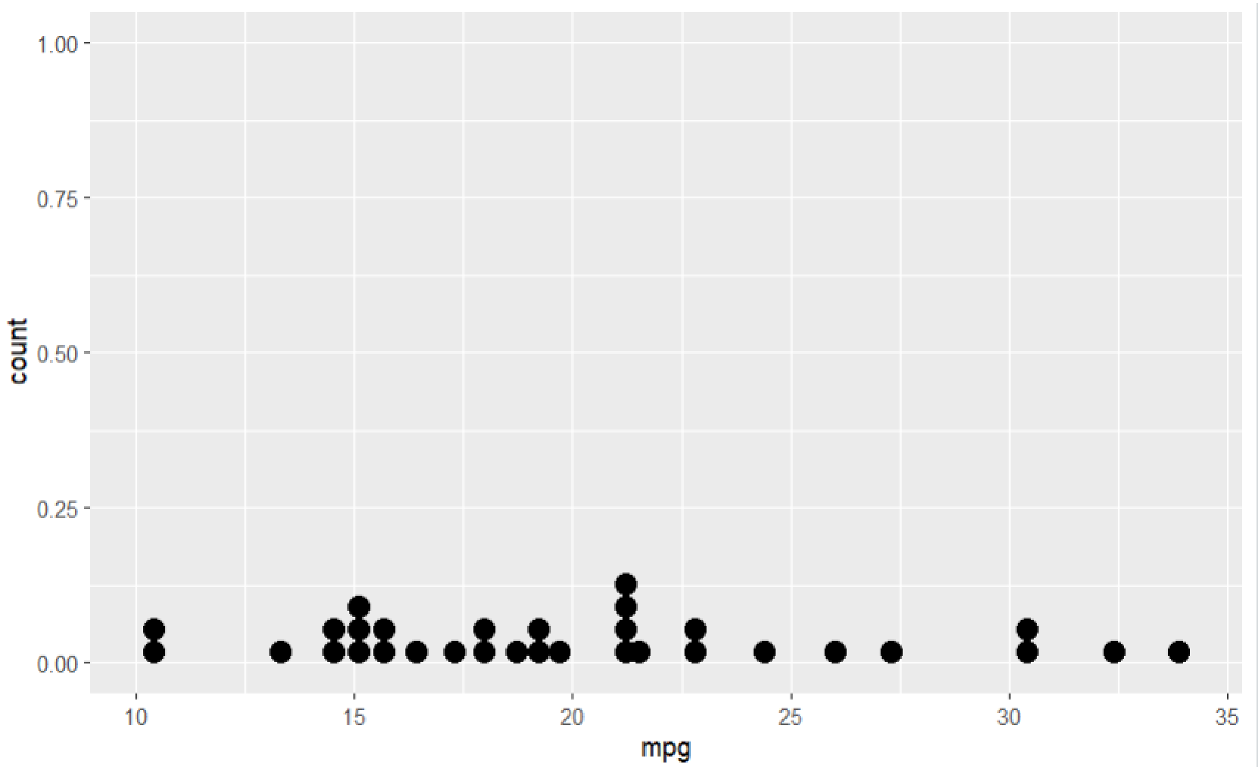

35
30
25
20
15
10
mpg
100
200
300
hp

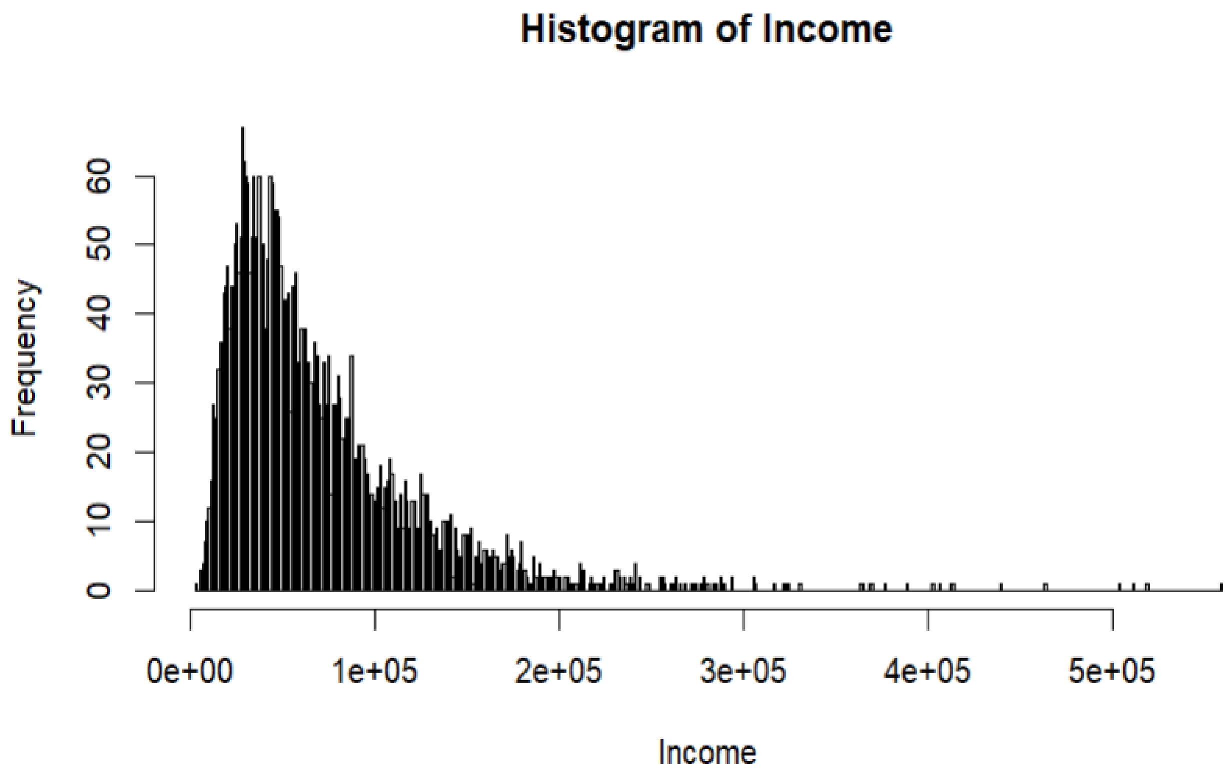

35
30
25
20
15
10
mpg
4
6
8
factor(cyl)

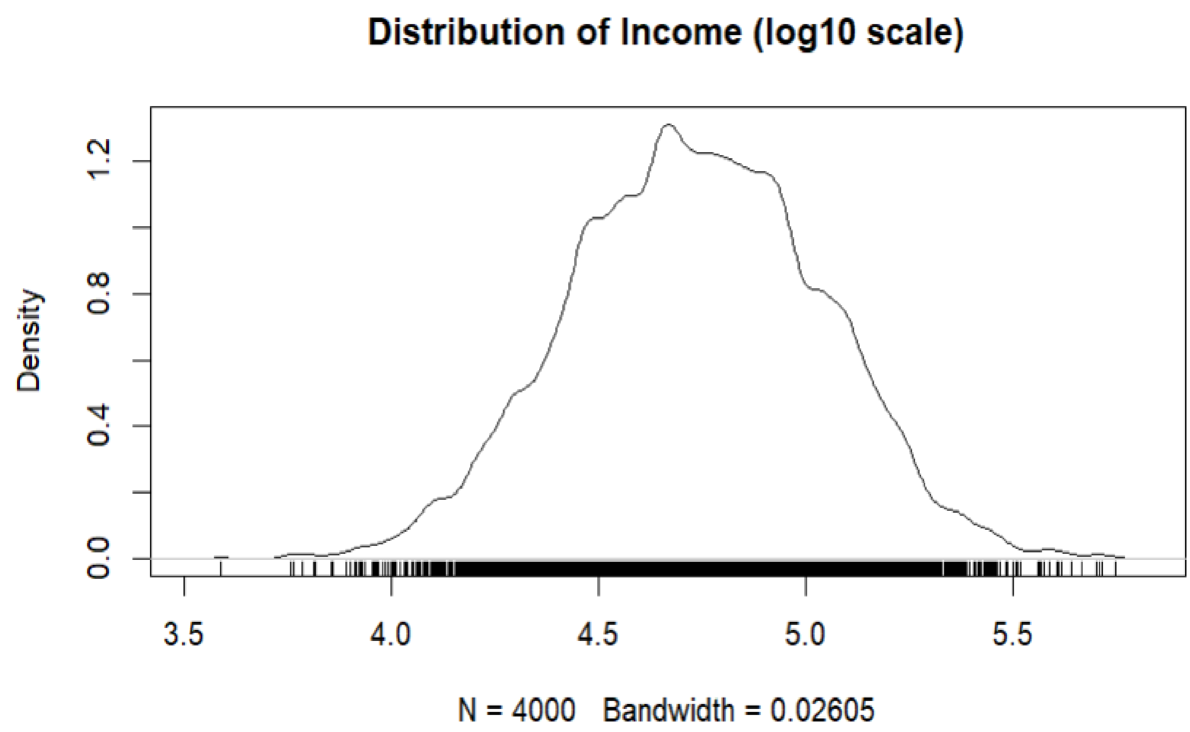

count
0
5
10
4
6
8
factor(cyl)

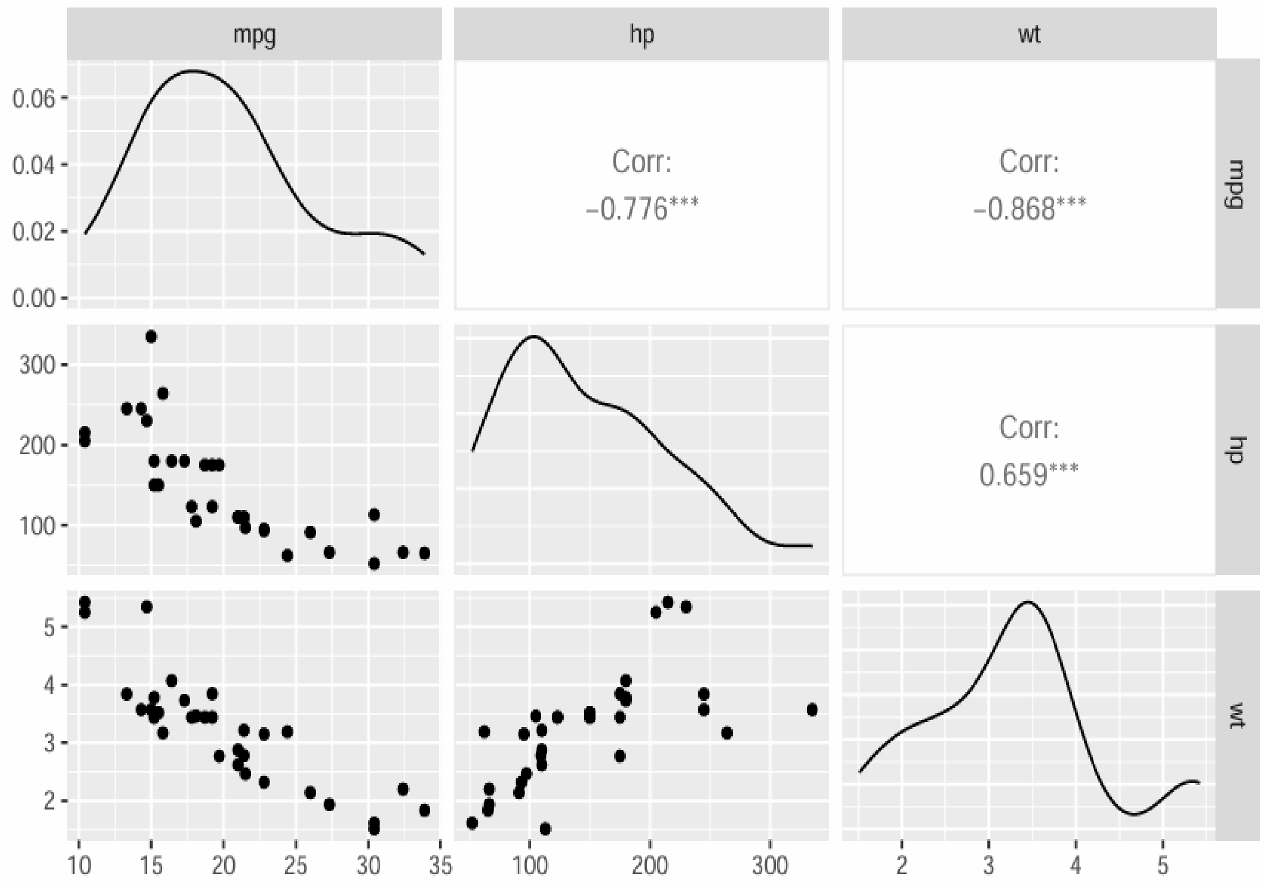

mpg
10
15
20
25
30
35
100
200
300
hp


35
30
25
20
15
10
mpg
100
200
300
hp
count
2.00
1.75
1.50
1.25
1.00


1.00
0.75
0.50
0.25
0.00
count
10
15
20
25
30
35
mpg


Histogram of Income
Frequency
0
10
20
30
40
50
60
0e+00
1e+05
2e+05
3e+05
4e+05
5e+05
Income


Distribution of Income (log10 scale)
Density
0.0
0.4
0.8
1.2
3.5
4.0
4.5
5.0
5.5
N = 4000 Bandwidth = 0.02605


mpg
hp
wt
Corr:
-0.776***
Corr:
-0.868***
Corr:
0.659***
0.00
0.02
0.04
0.06
100
200
300
2
3
4
5
10
15
20
25
30
35
100
200
300
2
3
4
5

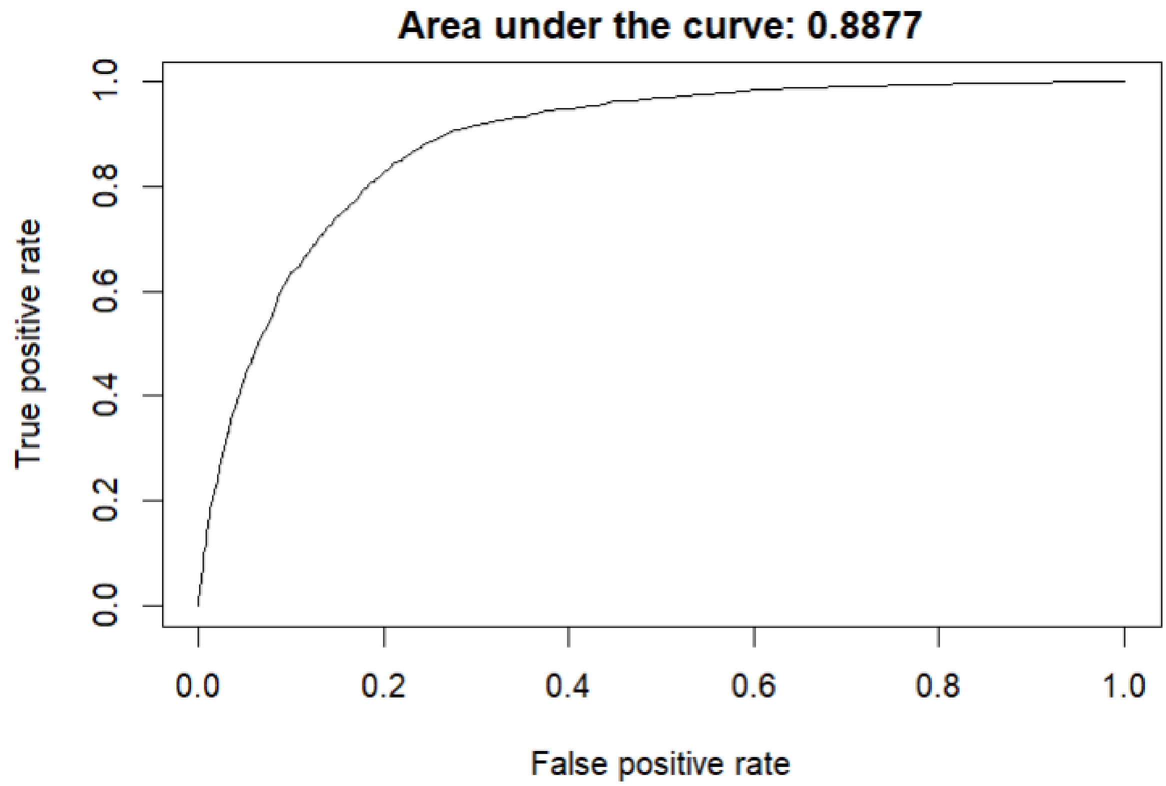
LOGISTIC REGRESSION

```
#####################################################################
# This code covers the code presented in
# Section 6.2 Logistic Regression
#####################################################################


#####################################################################
# Section 6.2.3 Diagnostics
#####################################################################
churn_input = as.data.frame(read.csv("C:/Users/vanif/Desktop/churn.csv"))
head(churn_input)  #displays first few rows n columns from the dataset

sum(churn_input$Churned)

Churn_logistic1 <- glm (Churned~Age + Married + Cust_years + Churned_contacts,
                        data=churn_input, family=binomial(link="logit"))#"glm()=Generalized Linear Models"

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
```

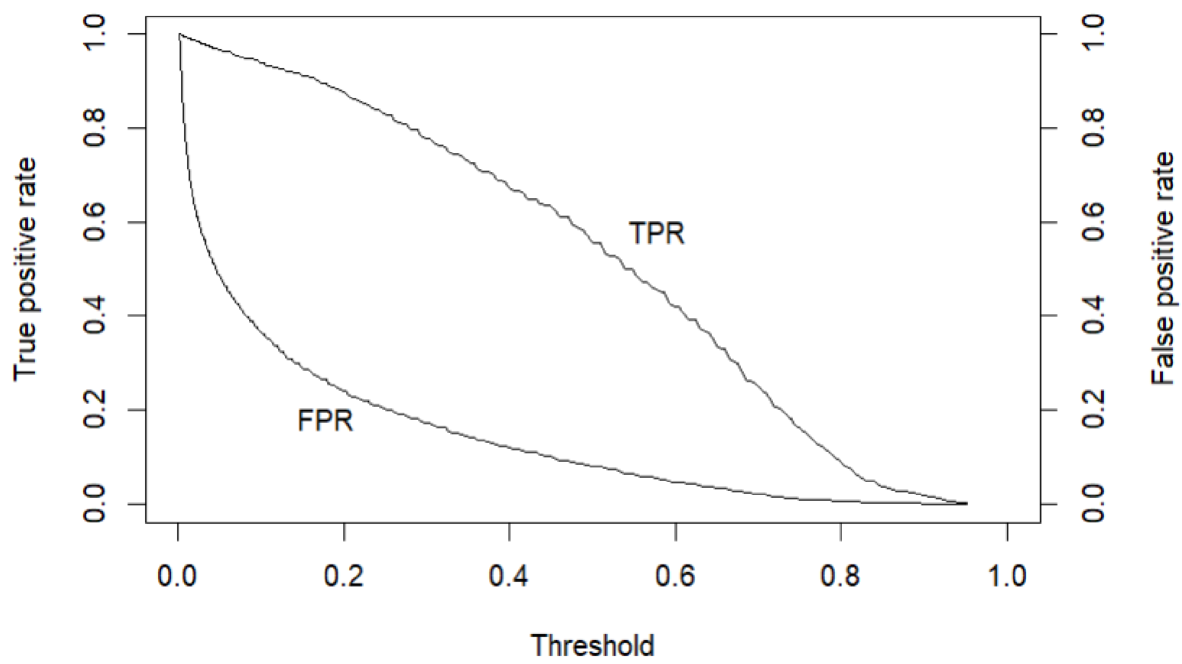
```
# Using the residual deviances from Churn_logistics2 and Churn_logistic3
# determine the signficance of the computed test statistic
summary(Churn_logistic2)
pchisq(.9 , 1, lower=FALSE)

# 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:



Area under the curve: 0.8877
True positive rate
False positive rate
0.0
0.2
0.4
0.6
0.8
1.0



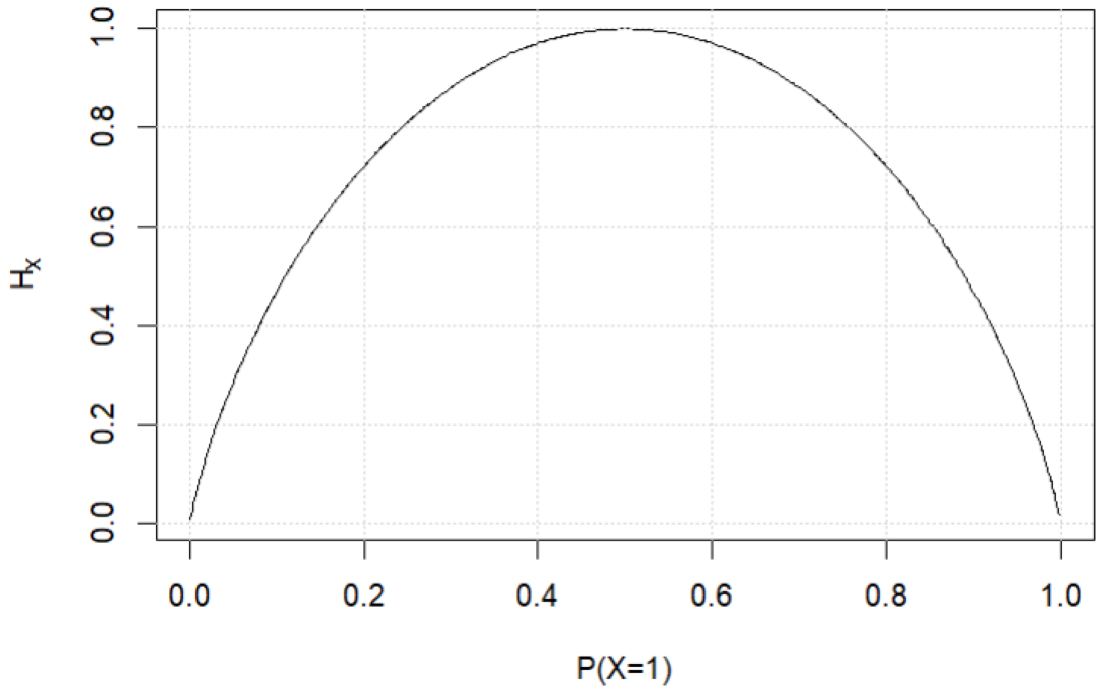
TPR
FPR
True positive rate
False positive rate
Threshold
0.0
0.2
0.4
0.6
0.8
1.0

DECISION TREE

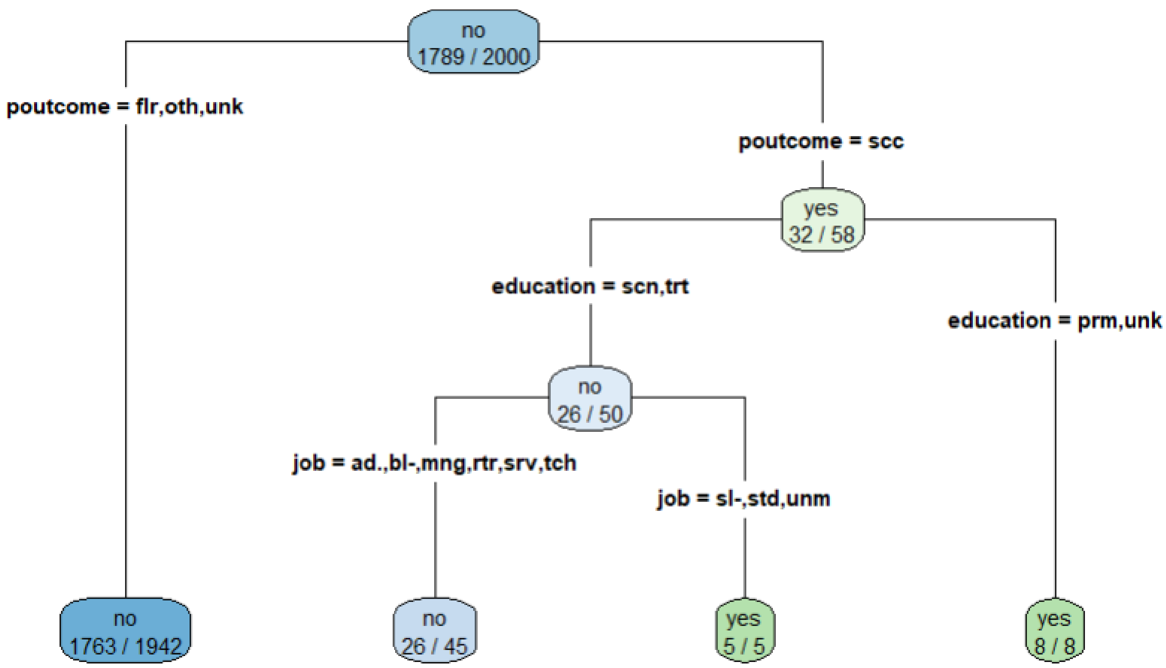
```
install.packages("rpart.plot")  #install package rpart.plot
library("rpart")            #load libraries
library("rpart.plot")

play_decision  <-
read.table("D:/DSA_DATASETS/datasets/banksample.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:



1.0
0.8
0.6
0.4
0.2
0.0
H_X
0.0
0.2
0.4
0.6
0.8
1.0
P(X=1)



no
1789 / 2000
poutcome = flr,oth,unk
poutcome = scc
yes
32 / 58
education = scn,trt
education = prm,unk
no
26 / 50
job = ad.,bl-,mng,rtr,srv,tch
job = sl-,std,unm
no
1763 / 1942
no
26 / 45
yes
5 / 5
yes
8 / 8

NAIVE BAYES CLASSIFIER

```
############################################################
# Naïve Bayes Classifier in R
############################################################

# Install required packages if not already installed
if (!require("e1071")) install.packages("e1071", dependencies=TRUE)
if (!require("rpart")) install.packages("rpart", dependencies=TRUE)
if (!require("rpart.plot")) install.packages("rpart.plot", dependencies=TRUE)

# Load necessary libraries
library(e1071)
library(rpart)
library(rpart.plot)

# Load dataset
banktrain <- read.csv("D:/DSA_DATASETS/datasets/bank-sample.csv", header=TRUE,
sep=",")

# Drop unnecessary columns
drops <- c("balance", "day", "campaign", "pdays", "previous", "month")
banktrain <- banktrain[, !(names(banktrain) %in% drops)]
summary(banktrain)

# Train Naïve Bayes model
model <- naiveBayes(subscribed ~ ., data=banktrain)
print(model)
```

```
# Predict on training data
predictions <- predict(model, banktrain)
table(predictions, banktrain$subscribed)

# Load test data (uncomment if needed)
# banktest <- read.csv("D:/DSA_DATASETS/datasets/bank-sample-test.csv", header=TRUE,
sep=",")
# banktest <- banktest[, !(names(banktest) %in% drops)]

# Predict on test data (if available)
# test_predictions <- predict(model, banktest)
# table(test_predictions, banktest$subscribed)

# Load another dataset for Naïve Bayes example
sample_data <- read.csv("D:/DSA_DATASETS/datasets/sample1.csv", header=TRUE,
sep=",")

# Split data into training and test sets
traindata <- sample_data[1:14, ]
testdata <- sample_data[15, ]

# Train Naïve Bayes model on sample dataset
model_sample <- naiveBayes(Enrolls ~ ., data=traindata)
print(model_sample)

# Predict on test sample
test_prediction <- predict(model_sample, testdata)
print(test_prediction)

# Train Naïve Bayes model with Laplace smoothing
```

```
model_sample_laplace <- naiveBayes(Enrolls ~ ., data=traindata, laplace=1)
print(model_sample_laplace)

# Predict using Laplace smoothing
laplace_prediction <- predict(model_sample_laplace, testdata)
print(laplace_prediction)
```

OUTPUT:

```
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  0.8000000 0.6000000 0.2000000
  Yes 0.3333333 0.4444444 0.5555556

     Income
Y          High       Low    Medium
  No  0.6000000 0.4000000 0.6000000
  Yes 0.3333333 0.4444444 0.5555556

     JobSatisfaction
Y            No       Yes
  No  1.0000000 0.4000000
  Yes 0.4444444 0.7777778

     Desire
Y     Excellent      Fair
  No  0.8000000 0.6000000
  Yes 0.4444444 0.7777778

[1] Yes
Levels: No Yes
```

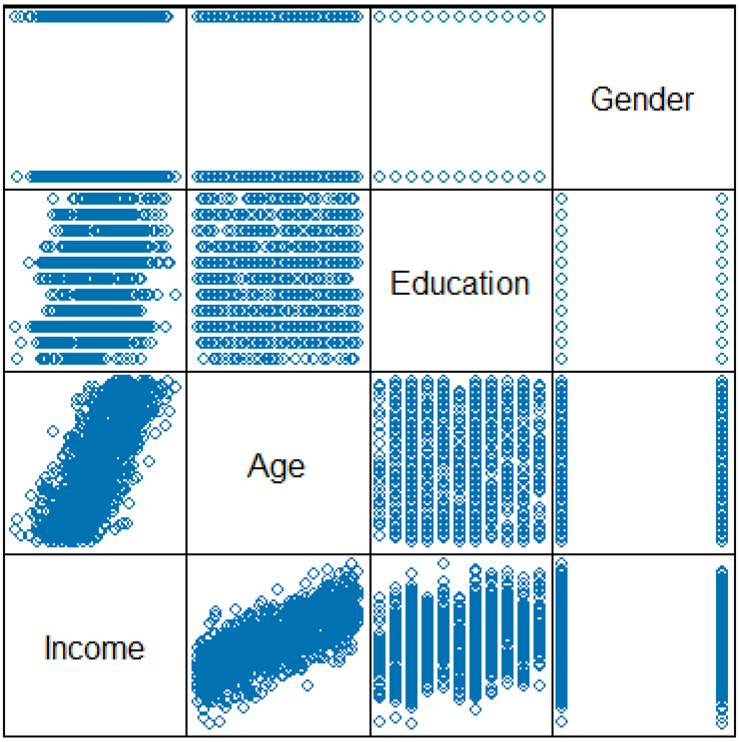
LINEAR REGRESSION

```
####################################################################
# This code covers the code presented in
# Section 6.1 Linear Regression
####################################################################

####################################################################
# Section 6.1.2
####################################################################

# Example in R

income_input=as.data.frame(read.csv("C:/Users/admin/Desktop/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)
```

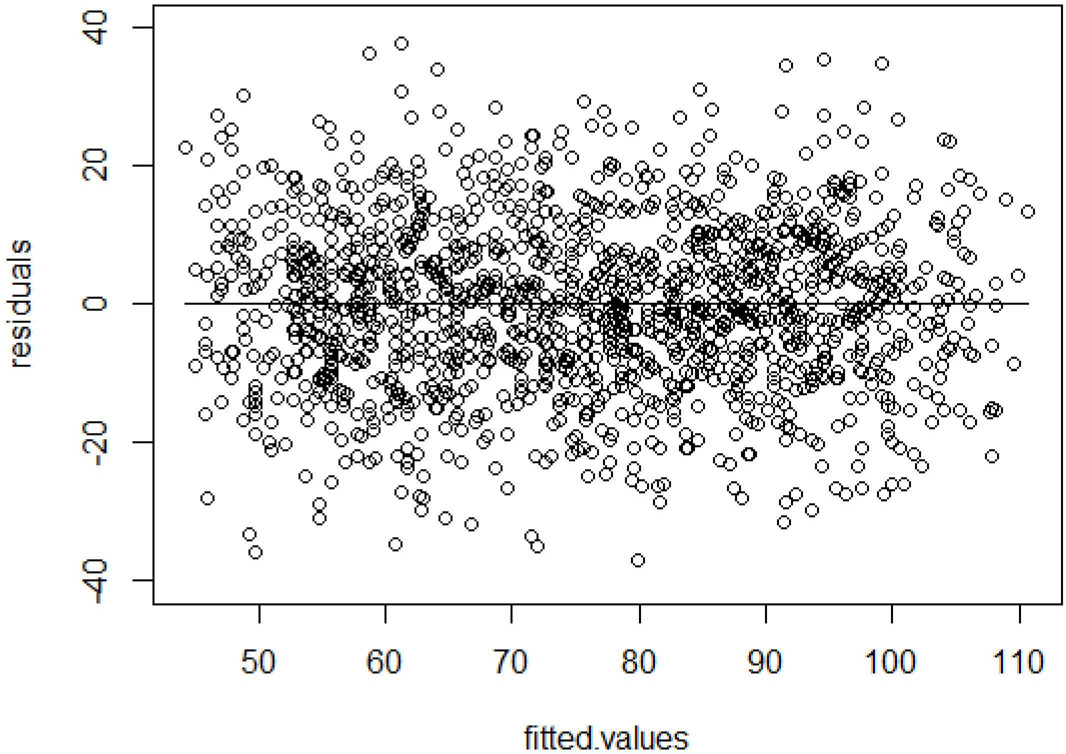
```
####################################################################
# this code from the text is for illustrative purposes only
# the income_input variable does not contain the U.S. states
results3 <- lm(Income ~ Age + Education,
            + Alabama,
            + Alaska,
            + Arizona,
            + WestVirginia,
            + Wisconsin,
            income_input)
####################################################################

# compute confidence intevals for the model parameters
confint(results2, level = .95)

# compute a confidence interval on the expected income of a person
Age <- 41
Education <- 12
new_pt <- data.frame(Age, Education)

conf_int_pt <- predict(results2, new_pt, level=.95, interval="confidence")
conf_int_pt

# compute a prediction interval on the income of the same person
pred_int_pt <- predict(results2, new_pt, level=.95, interval="prediction")
pred_int_pt
```

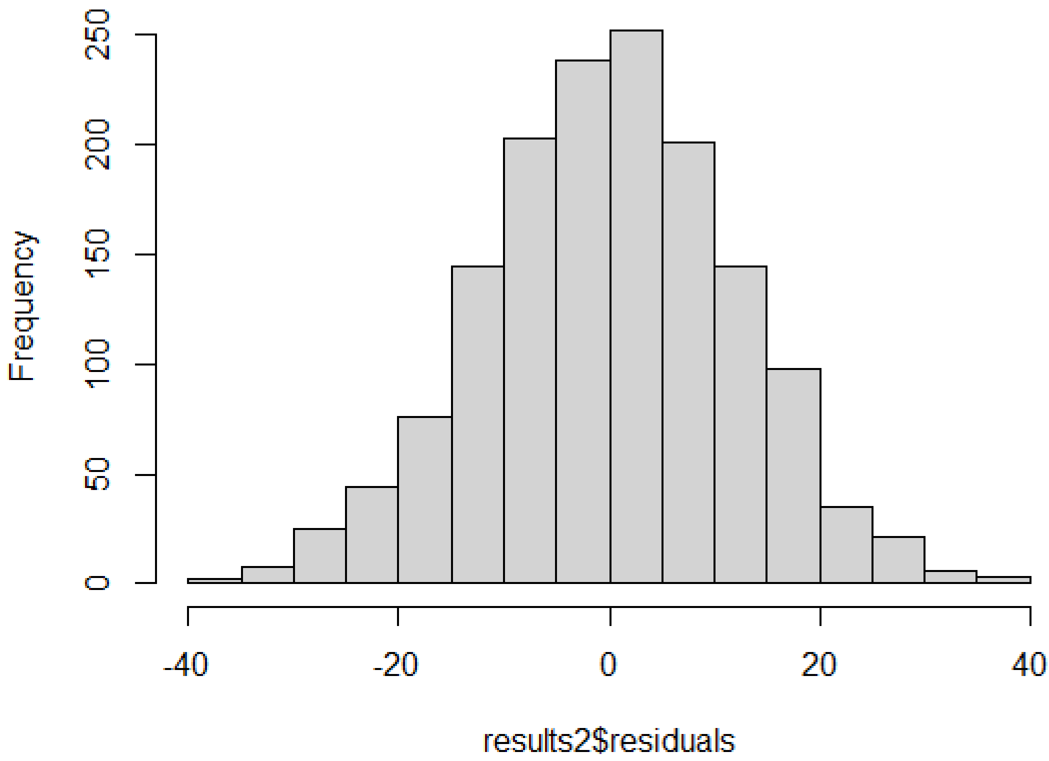
```
##################################################################
# section 6.1.3 Diagnostics
##################################################################

with(results2, {
  plot(fitted.values, residuals,ylim=c(-40,40) )
  points(c(min(fitted.values),max(fitted.values)), c(0,0), type = "l")})

hist(results2$residuals, main="")

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

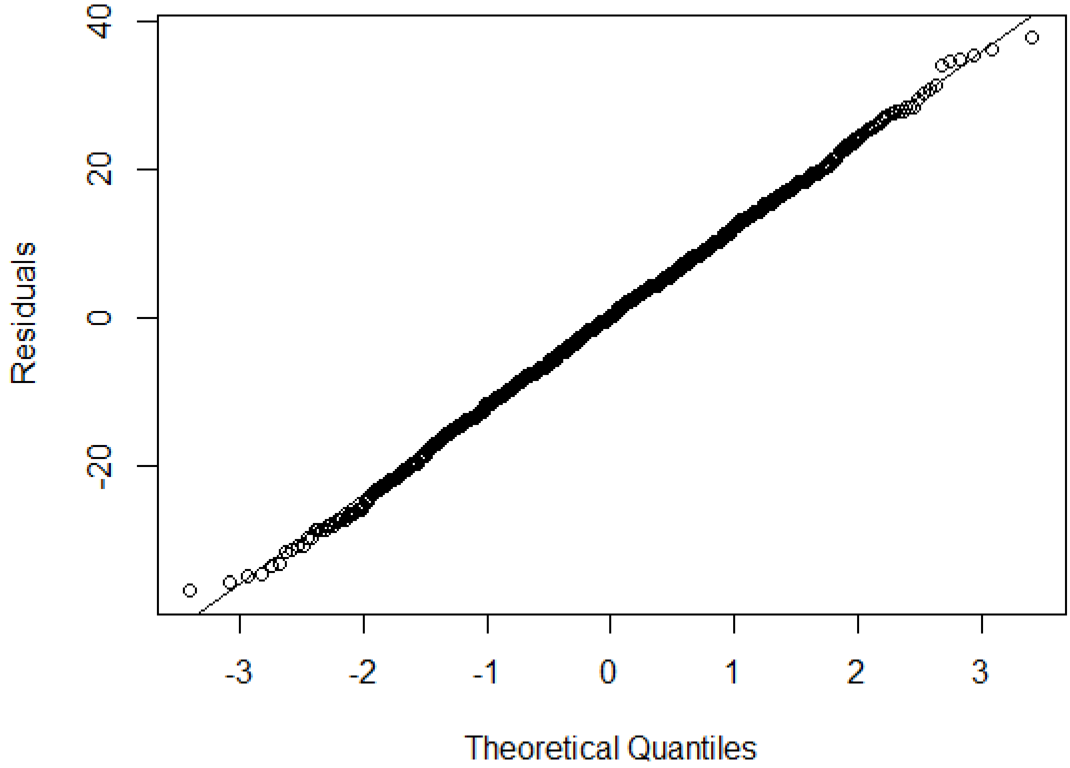
Outputs:



Scatter Plot Matrix




Frequency
0
50
100
150
200
250
-40
-20
0
20
40
results2$residuals


Residuals
-20
0
20
40
-3
-2
-1
0
1
2
3
Theoretical Quantiles