

**STA401**

**Fall 2024**

**Dr. Ayman Alzaatreh**

**Homework 4**

**Halla, Rumisal, Eleonara**

**20/10/2024**

**Q2, part 11)**

setwd("/Users/halla.d/Library/Mobile Documents/com~apple~CloudDocs/Desktop/STA401")

auto = read.csv("Auto.csv")

#a

median = median(auto$mpg)

median

mpg01 = as.numeric(auto$mpg > median)

mpg01

df = auto[, names(auto) != "mpg"] # Remove the 'mpg' column

df$mpg01 = mpg01 # Add 'mpg01' as a new column

df$mpg01 = as.factor(df$mpg01)

df$origin = as.factor(df$origin)

df$cylinders = as.factor(df$cylinders)

df = na.omit(df)

sum(is.na(df))

df$horsepower = as.integer(df$horsepower)

head(df)

tail(df)

#b

str(df)

plot(df)

par(mfrow = c(2, 4))

boxplot(df$cylinders ~ df$mpg01,

ylab = "Number of Cylinders",

xlab = "mpg01",

main = "Boxplot of #Cylinders vs mpg01")

boxplot(df$displacement ~ df$mpg01,

ylab = "Displacement",

xlab = "mpg01",

main = "Boxplot of dislpacement vs mpg01")

boxplot(df$year ~ df$mpg01,

ylab = "Year",

xlab = "mpg01",

main = "Boxplot of Year vs mpg01")

boxplot(df$weight ~ df$mpg01,

ylab = "weight",

xlab = "mpg01",

main = "Boxplot of Weight vs mpg01")

boxplot(df$horsepower ~ df$mpg01,

ylab = "horsepower",

xlab = "mpg01",

main = "Boxplot of horsepower vs mpg01")

boxplot(df$acceleration ~ df$mpg01,

ylab = "acceleration",

xlab = "mpg01",

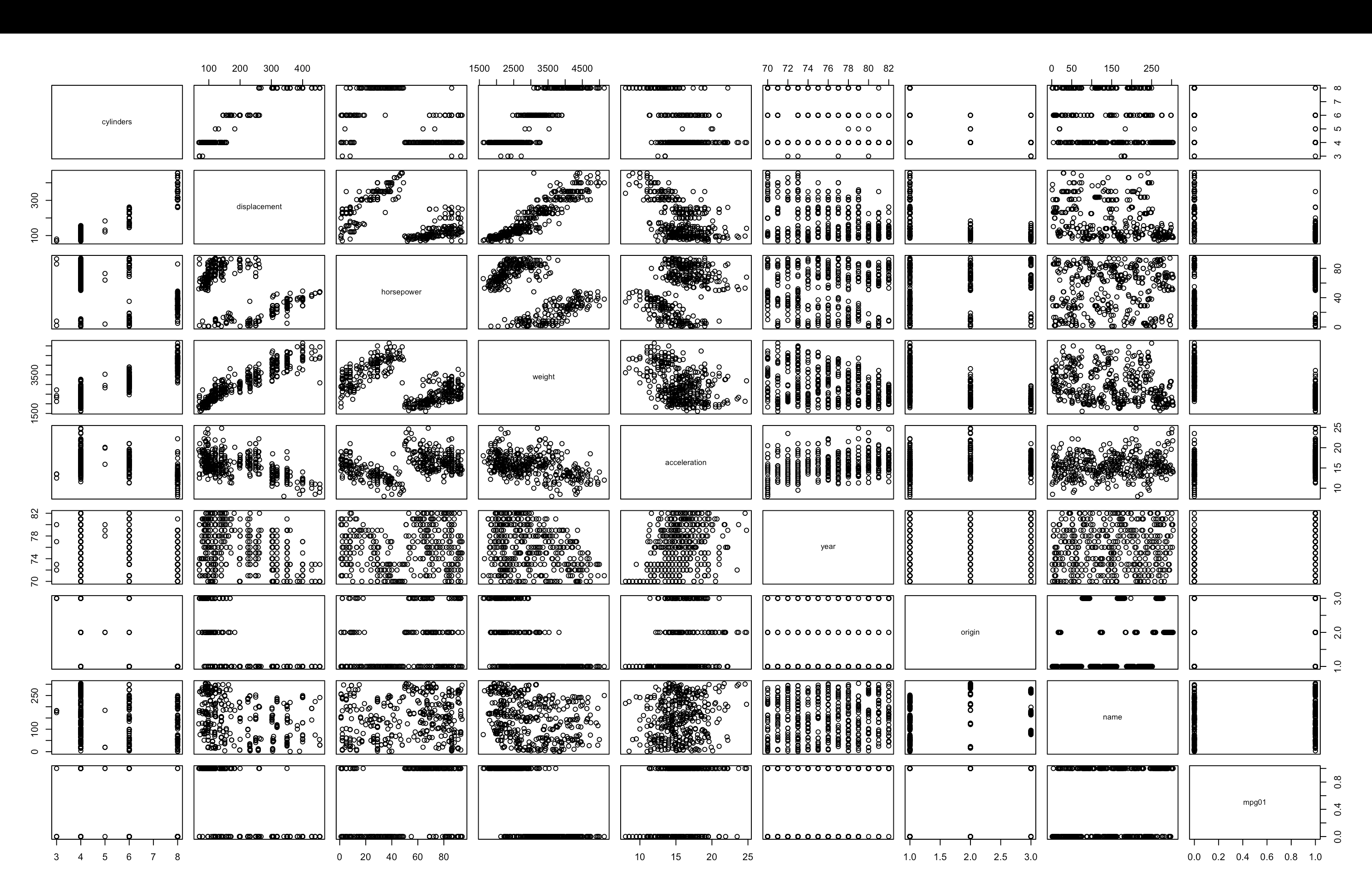
main = "Boxplot of acceleration vs mpg01")

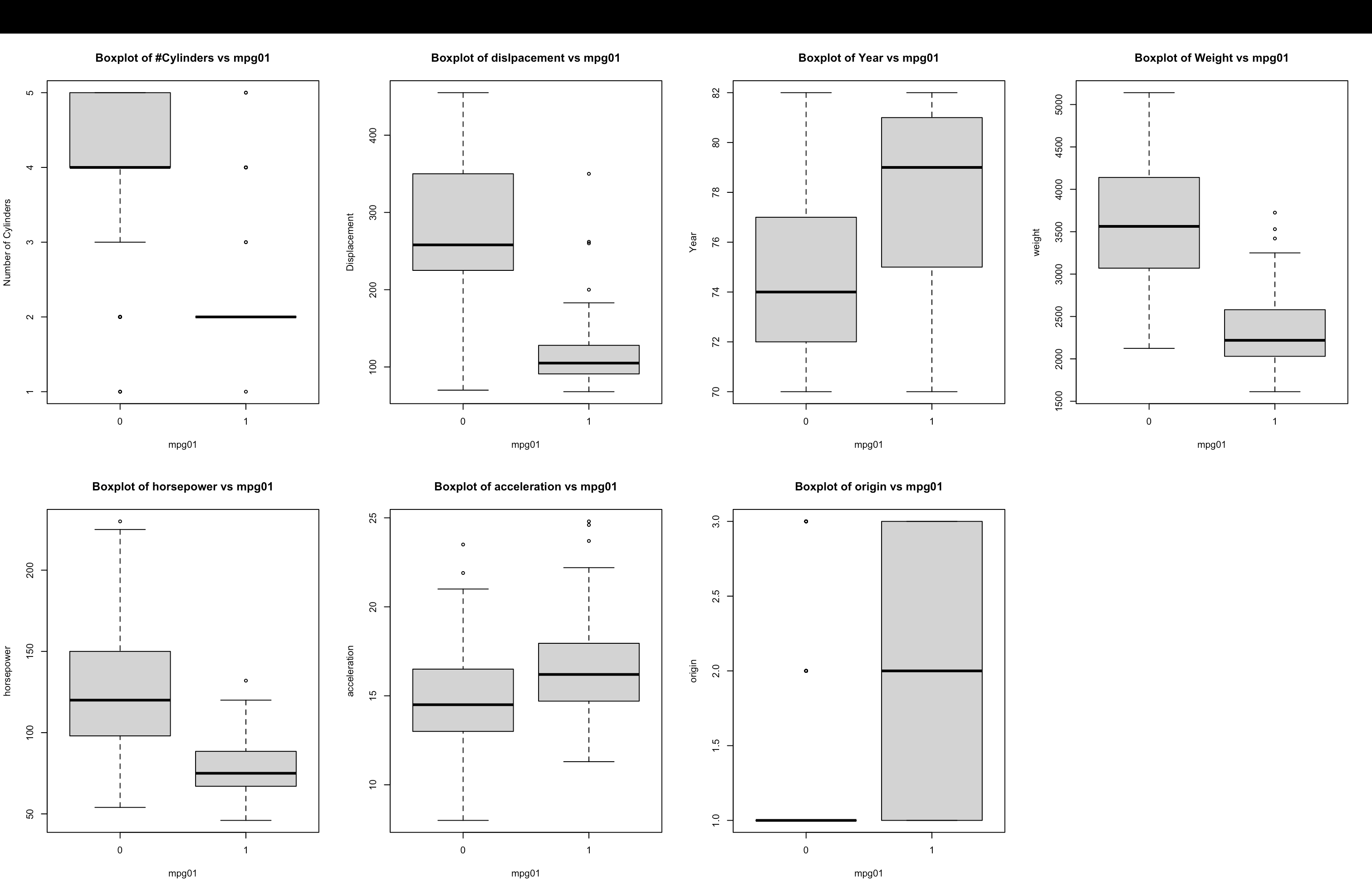
boxplot(df$origin ~ df$mpg01,

ylab = "origin",

xlab = "mpg01",

main = "Boxplot of origin vs mpg01")





As we can see from the boxplots, there could be a relationship between mpg01 and displacement, year, weight, horsepower, cylinders, and acceleration whereas the boxplots with mpg01 against # origins does not show much.

* displacement: the median displacement covered by cars with a higher mileage (around 120) is around half as much as that of cars with a lower mileage
* newer cars seem to have a higher median mileage
* heavier cars seem to have a lower median mileage
* cars with higher mileage tend to have a smaller median horsepower
* cars with higher mileage have a higher median acceleration
* cars of a lower mileage have atleast 3 cylinders with a median of 4 cylinders. whereas cars of higher mileage have 2 cylinders on average

#c

library(caret)

s <- createDataPartition(y = df$mpg01,p = 0.7,list = FALSE)

train\_df <- df[s,] # 70% training

test\_df<- df[-s,] # 30% testing

summary(train\_df$mpg01)

131/(131+144)

summary(test\_df$mpg01)

56/(56+61)

#d

library(MASS)

lda.fit=lda(mpg01~ horsepower + cylinders + weight + displacement + year + acceleration,data=train\_df)

lda.fit

lda.pred=predict(lda.fit, test\_df) #don't put "data =" before test.2005

lda.class=lda.pred$class

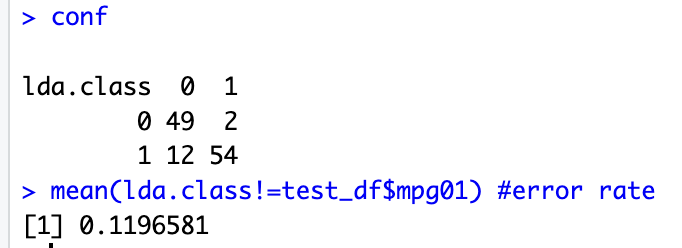
lda.class

lda.class[1:20]

conf = table(lda.class,test\_df$mpg01)

conf

mean(lda.class!=test\_df$mpg01) #error rate



The LDA test error rate is 11.97%

#e

qda.fit = qda(mpg01~ horsepower + weight + displacement + year + acceleration,data=train\_df)

qda.pred=predict(qda.fit, test\_df) #don't put "data =" before test.2005

qda.class=qda.pred$class

qda.class[1:20]

conf = table(qda.class,test\_df$mpg01)

conf

mean(qda.class!=test\_df$mpg01) #error rate

A white background with black text

Description automatically generated

The QDA test error rate is 13.68%

#f

glm.fit=glm(mpg01~ horsepower + weight + displacement + year + acceleration,data=train\_df,family=binomial)

summary(glm.fit)

glm.probs = predict(glm.fit, newdata = test\_df, type = "response")

glm.probs[1:10]

n=length(glm.probs)

glm.pred=rep(0,n)

glm.pred

glm.pred[glm.probs>.5]= 1

glm.pred

table(predict=glm.pred,actual=test\_df$mpg01) #classifcation matrix

mean(glm.pred!=test\_df$mpg01)

A white background with blue text

Description automatically generated

The logistic regression test error rate is 11.97%

#g

set.seed(12345)

library(caret)

library(FNN)

train.knn = train\_df[ , - (8:9)] #removing response and name

test.knn = test\_df[ , - (8:9)] #removing response and name

log(dim(train\_df)[1]) #5.6 so we'll consider 5 and 7

tct <- trainControl(method="repeatedcv")

knn.pred <- train(mpg01~ horsepower + weight + displacement + year + acceleration,data=train\_df,

method = "knn", trControl = tct, preProcess = c("center","scale"),tuneLength = 15)

knn.pred

plot(knn.pred)

knn.pred=knn(train.knn,test.knn,train\_df$mpg01,k=5)

conf=table(knn.pred,test\_df$mpg01)

mean(knn.pred != test\_df$mpg01)

knn.pred=knn(train.knn,test.knn,train\_df$mpg01,k=1)

conf=table(knn.pred,test\_df$mpg01)

mean(knn.pred != test\_df$mpg01)

knn.pred=knn(train.knn,test.knn,train\_df$mpg01,k=7)

conf=table(knn.pred,test\_df$mpg01)

mean(knn.pred != test\_df$mpg01)

knn.pred=knn(train.knn,test.knn,train\_df$mpg01,k=25)

conf=table(knn.pred,test\_df$mpg01)

mean(knn.pred != test\_df$mpg01)

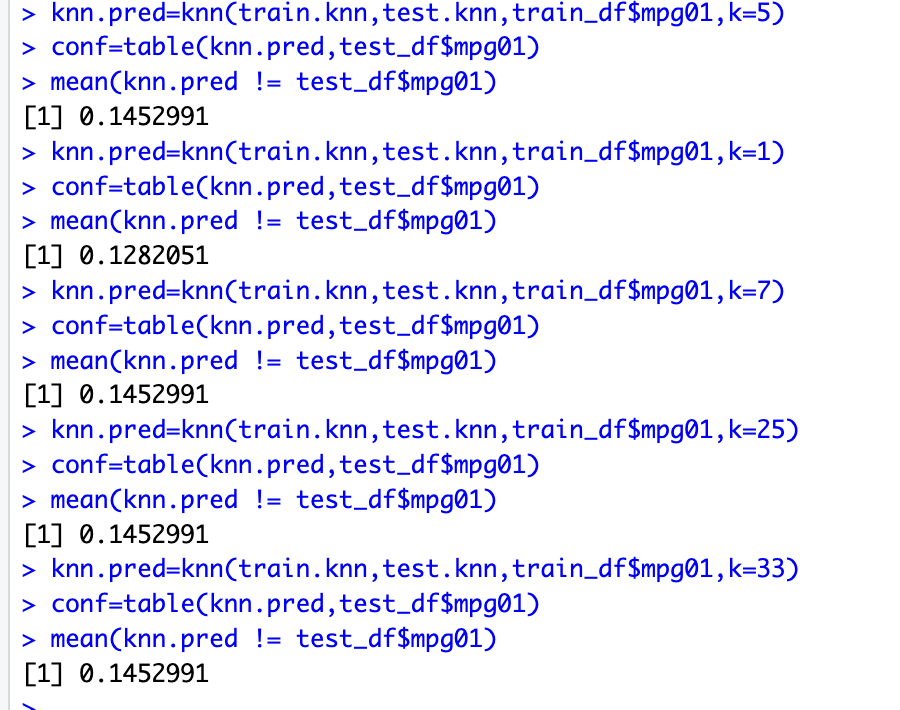
knn.pred=knn(train.knn,test.knn,train\_df$mpg01,k=33)

conf=table(knn.pred,test\_df$mpg01)

mean(knn.pred != test\_df$mpg01)

A graph with a line

Description automatically generated



we can see that k = 1 has the lowest error rate of 12.82% as expected since it’s prone to overfitting.

k=5 and above all give the same error rate of 14.53% so we pick k = 5 which is also log(n)

**Q2, part 12)**

#a

Power <- function() {

result <- 2^3

print(result)

}

Power()



#b

Power2 <- function(x,y) {

result <- x^y

print(result)

}

Power2(3,8)

A close up of numbers

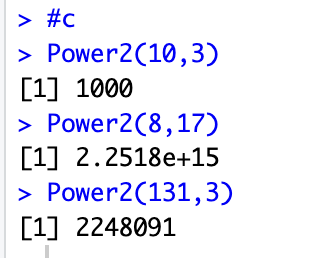
Description automatically generated

#c

Power2(10,3)

Power2(8,17)

Power2(131,3)



#d

Power3 <- function(x,y) {

result <- x^y

return(result)

}

#e

par(mfrow = c(2,2))

x = seq(1:10)

x

x\_sq = Power3(x,2)

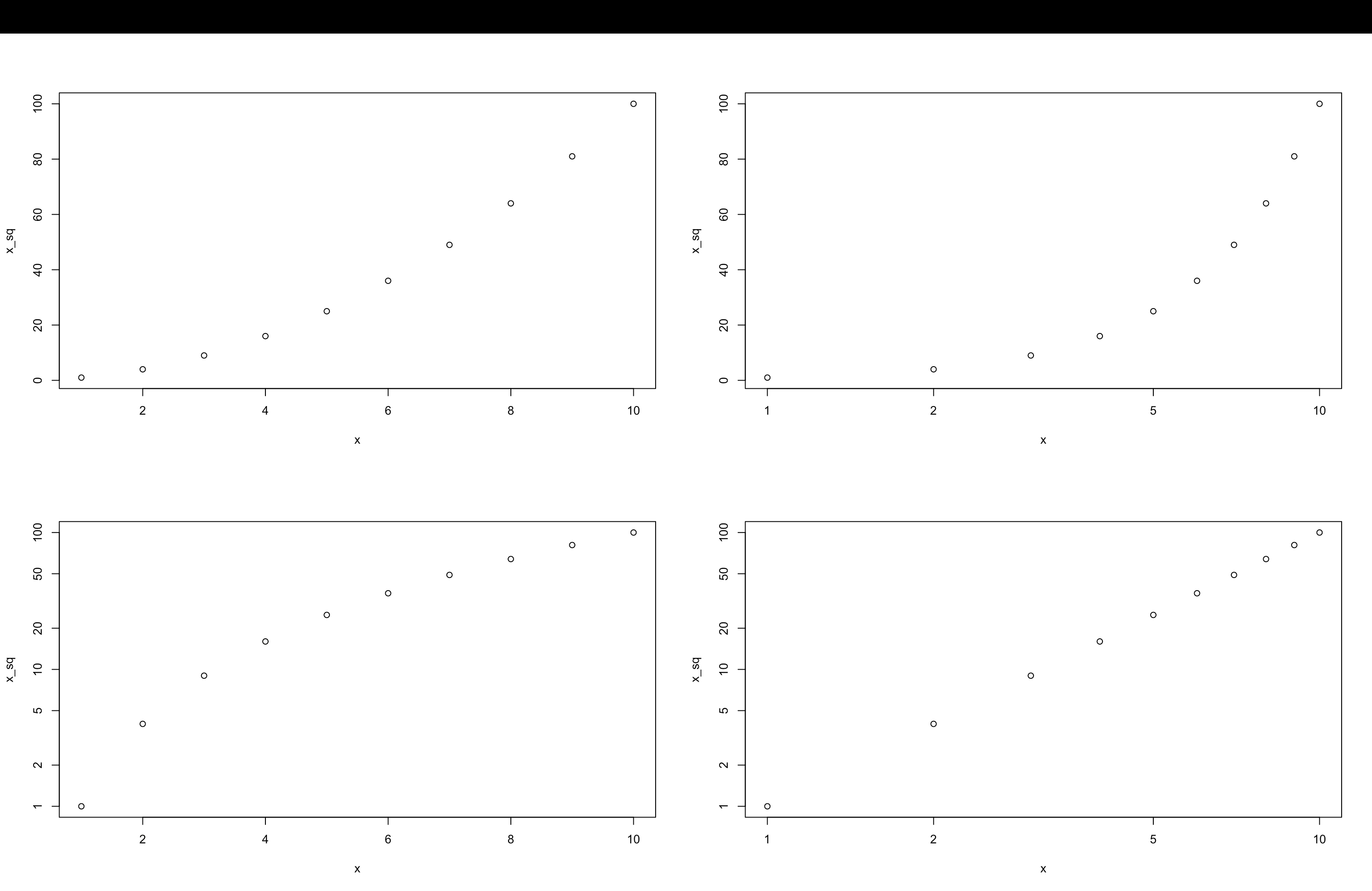
x\_sq

plot(x,x\_sq)

plot(x,x\_sq, log = 'x')

plot(x,x\_sq, log = 'y')

plot(x,x\_sq, log = 'xy')



#f

PlotPower <- function(xi, xf , a){

par(mfrow = c(1,1))

x = seq(xi:xf)

y = x^a

plot(x,y)

}

PlotPower(1,10,3)

A screen shot of a graph

Description automatically generated

**Q2, part 13)**

library(MASS)

data(Boston)

df = Boston

head(df)

median = median(df$crim)

median

crime\_med = as.numeric(df$crim > median) #1 is crime rate above median

crime\_med

df = df[, names(df) != "crim"] # Remove the crime column

df$crime\_med = crime\_med # Add 'crime\_med ' as a new column

df$crime\_med = as.factor(df$crime\_med)

str(df)

df = na.omit(df)

sum(is.na(df))

head(df)

tail(df)

library(caret)

s <- createDataPartition(y = df$crime\_med,p = 0.7,list = FALSE)

train\_df <- df[s,] # 70% training

test\_df<- df[-s,] # 30% testing

summary(train\_df$crime\_med)

summary(test\_df$crime\_med)

lda.fit=lda(crime\_med ~ .,data=train\_df)

lda.fit

lda.pred=predict(lda.fit, test\_df) #don't put "data =" before test.2005

lda.class=lda.pred$class

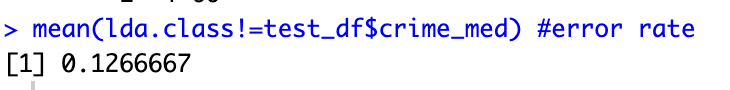
lda.class

lda.class[1:20]

conf = table(lda.class,test\_df$crime\_med)

conf

mean(lda.class!=test\_df$crime\_med) #error rate



**The LDA error rate is 12.67%**

glm.fit=glm(crime\_med ~., data = train\_df,family=binomial)

summary(glm.fit)

glm.probs = predict(glm.fit, newdata = test\_df, type = "response")

glm.probs[1:10]

n=length(glm.probs)

glm.pred=rep(0,n)

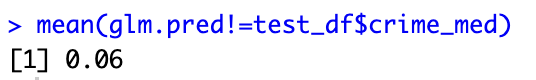
glm.pred

glm.pred[glm.probs>.5]= 1

glm.pred

table(predict=glm.pred,actual=test\_df$crime\_med) #classifcation matrix

mean(glm.pred!=test\_df$crime\_med)



**the logistic error rate is 6%**

set.seed(12345)

library(caret)

library(FNN)

head(train\_df)

train.knn = train\_df[ , - 14] #removing response

test.knn = test\_df[ , -14] #removing response

log(dim(train\_df)[1]) #5.9 so we'll consider 5 and 7

tct <- trainControl(method="repeatedcv")

knn.pred <- train(crime\_med~ .,data=train\_df,

method = "knn", trControl = tct, preProcess = c("center","scale"),tuneLength = 15)

knn.pred

plot(knn.pred)

A graph with a line

Description automatically generated

**We can see from the plot that k=5 has the highest accuracy. let’s test that**

knn.pred=knn(train.knn,test.knn,train\_df$crime\_med,k=5)

conf=table(knn.pred,test\_df$crime\_med)

mean(knn.pred != test\_df$crime\_med)

knn.pred=knn(train.knn,test.knn,train\_df$crime\_med,k=1)

conf=table(knn.pred,test\_df$crime\_med)

mean(knn.pred != test\_df$crime\_med)

knn.pred=knn(train.knn,test.knn,train\_df$crime\_med,k=7)

conf=table(knn.pred,test\_df$crime\_med)

mean(knn.pred != test\_df$crime\_med)

knn.pred=knn(train.knn,test.knn,train\_df$crime\_med,k=25)

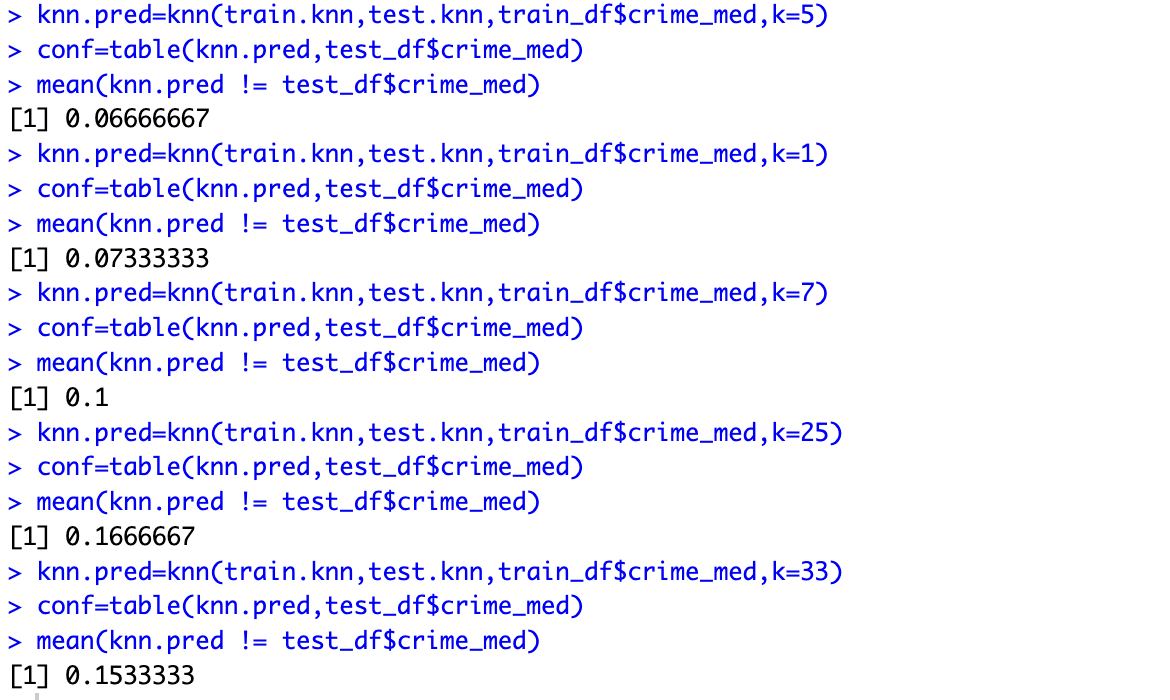
conf=table(knn.pred,test\_df$crime\_med)

mean(knn.pred != test\_df$crime\_med)

knn.pred=knn(train.knn,test.knn,train\_df$crime\_med,k=33)

conf=table(knn.pred,test\_df$crime\_med)

mean(knn.pred != test\_df$crime\_med)



**we can see that k = 5 has the lowest error rate of 6.67% hence k = 5 is the best amongst these models.**

**Overall, logistic regression produced the lowest error rate of 6%**