R Programming…….

>For Input

[1]For Output

#For Comment

>print("TDM ASSIGNMENT BY 122AD0027\_Pranjal")

[1] " TDM ASSIGNMENT BY 122AD0027\_Pranjal "

#Assignment method

> x=5

> print(x)

[1] 5

> x

[1] 5

> z<-7

> z

[1] 7

> 7-> y

> y

[1] 7

> assign("z",8)

> z

[1] 8

# C fn^ use combine multiple values

> x=c(1,2,5)

> x

[1] 1 2 5

> var1=c(TRUE,8)

> var1

[1] 1 8

#ls() for List Object

> ls()

[1] "a" "a1" "bplo" "cn" "df1" "f"

[7] "factor.f" "glfit" "irdata" "irtest" "irtrain" "l1"

[13] "l2" "m1" "m2" "merged" "newdata" "num1"

[19] "num2" "PL" "predict\_reg" "predicted\_val" "PW" "rn"

[25] "samp" "speciesID" "v1" "v2" "v3" "v4"

[31] "v5" "v6" "v7" "var1" "var2" "x"

[37] "y" "z"

#rm() for remove

> rm(x)

> ls()

[1] "a" "a1" "bplo" "cn" "df1" "f"

[7] "factor.f" "glfit" "irdata" "irtest" "irtrain" "l1"

[13] "l2" "m1" "m2" "merged" "newdata" "num1"

[19] "num2" "PL" "predict\_reg" "predicted\_val" "PW" "rn"

[25] "samp" "speciesID" "v1" "v2" "v3" "v4"

[31] "v5" "v6" "v7" "var1" "var2" "y"

[37] "z"

#Different Types of Data Types

#logical.

#numeric.

#integer.

#complex.

#character

#raw

> a="pranjal\_122AD0027"

> a

[1] " pranjal\_122AD0027"

> nchar(a)

[1] 17

> #readline for input

> var1=readline()

27

> var1

[1] "27"

> class(var1)

[1] "character"

> var2=as.integer(var1)

> var2

[1] 27

> {

+ num1=as.integer(readline("enter num1: "))

+ num2=as.integer(readline("enter num2: "))

+

+ print(num1+num2)

+ }

enter num1: 10

enter num2: 17

[1] 27

> b=scan()

1: 1

2: 2

3: 3

4: 4

5: 5

6: 6

7: 7

8: 8

9:

Read 8 items

> v4=charToRaw("hello jisna ")

> v4

[1] 68 65 6c 6c 6f 20 6a 69 73 6e 61

> class(v4)

[1] "raw"

> v5=as.Date("1233-12-23")

> v5

[1] "1233-12-23"

> class(v5)

[1] "Date"

> #assign integer

> x=as.integer(5)

> class(x)

[1] "integer"

> y=27L

> class(y)

[1] "integer"

# 26/05/23 OPERATORS in R

# v%%t - modular division

# v%/%t - integer division >> question part only

# Arithmetic Operator

# Relational Operator

# Logical Operators : "&" operator --> non zero values are true(1) "|" --> or operator

# Assignment Operators : LEFT (<- , <<- , =, ->> , ->) RIGHT

# mISCELLANEOUS oPERATOR :

# Matrix Multiplication (M1%\*%M2)

# (%in%)

# DESCCISION MAKING

# if | Else | Else if | Switch

# LOOPS

# repeat Loops | While Loops | for Loops

# for loops

#for (i in 1:5){ print i } ---> so values from 1 to 5 would be printed

# this would include last value also -- 1 2 3 4 5

# while loops

# while(\_\_){}

# repeat{

# do something

# if(some condition is true){

# break

# }

#}

#

> # EX 1 : Prog to print Square of Numbers from 1 to 20

>

> for (i in 1:20){

+ print (i^2)

+ }

[1] 1

[1] 4

[1] 9

[1] 16

[1] 25

[1] 36

[1] 49

[1] 64

[1] 81

[1] 100

[1] 121

[1] 144

[1] 169

[1] 196

[1] 225

[1] 256

[1] 289

[1] 324

[1] 361

[1] 400

> # EX 3: Program to display months of an year( Hint : use c() then traverse through month in each year)

> a<-c(1:12)

> month.name[a]

[1] "January" "February" "March" "April" "May" "June" "July" "August"

[9] "September" "October" "November" "December"

#Data Structures

# Strings

# Vectors

# Lists

# Array

# Matrices

# Factors

# Data Frames

#VECTORS -> Homogenies d.s

> v1=c(1,2,3,4)

> v1\*2

[1] 2 4 6 8

> v1^2

[1] 1 4 9 16

> v2=c(4,3,2,1)

> v1+v2

[1] 5 5 5 5

> length(v1)

[1] 4

> v3=c("red","seagreen","grey","yellow")

> v3[c(1,2,3)]

[1] "red" "seagreen" "grey"

> v3[c(-1,-2)]

[1] "grey" "yellow"

> v4=1:8

> v4

[1] 1 2 3 4 5 6 7 8

> v5=seq(1,9,by=2)

> v5

[1] 1 3 5 7 9

> v6=letters[1:10]

> v6

[1] "a" "b" "c" "d" "e" "f" "g" "h" "i" "j"

> v7=c(2.4,45,134524,46,-12,-46)

> sort(v7)

[1] -46.0 -12.0 2.4 45.0 46.0 134524.0

> #if any element is less than 5

> any(v1<5)

[1] TRUE

> #if all element is less than 5

> all(v1<5)

[1] TRUE

#Lists in R

> l1=list("here","we","go","again","27")

> l1

[[1]]

[1] "here"

[[2]]

[1] "we"

[[3]]

[1] "go"

[[4]]

[1] "again"

[[5]]

[1] "27"

> class(l1)

[1] "list"

> unlist(l1)

[1] "here" "we" "go" "again" "27"

> l1[3]

[[1]]

[1] "go"

> l1[1:3]

[[1]]

[1] "here"

[[2]]

[1] "we"

[[3]]

[1] "go"

> #name is fn

> names(l1)=c("a","b","c","d","e")

> l1[["a"]]

[1] "here"

> l2=list(a=1:27,b=letters[1:5],c=27)

> l2$a

[1] 1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 16 17 18 19 20 21 22 23 24 25 26 27

> l2[1]=NULL

> merged=c(l1,l2)

> merged

$a

[1] "here"

$b

[1] "we"

$c

[1] "go"

$d

[1] "again"

$e

[1] "27"

$b

[1] "a" "b" "c" "d" "e"

$c

[1] 27

#Matrixes

> m1=matrix(c(26:50),nrow=5,byrow = TRUE)

> m1

[,1] [,2] [,3] [,4] [,5]

[1,] 26 27 28 29 30

[2,] 31 32 33 34 35

[3,] 36 37 38 39 40

[4,] 41 42 43 44 45

[5,] 46 47 48 49 50

> m1[1,2]

[1] 27

> dim(m1)

[1] 5 5

> rn=c(1,2,3,4,5)

> cn=c("a","b","c","d","e")

> m2=matrix(c(1:25),nrow=5,byrow = TRUE,dimnames = list(rn,cn))

> m2

a b c d e

1 1 2 3 4 5

2 6 7 8 9 10

3 11 12 13 14 15

4 16 17 18 19 20

5 21 22 23 24 25

> length(m1)

[1] 25

> length(m2)

[1] 25

> m1\*m2

a b c d e

1 26 54 84 116 150

2 186 224 264 306 350

3 396 444 494 546 600

4 656 714 774 836 900

5 966 1034 1104 1176 1250

> m1%\*%m2 #This operator is used to multiply a matrix with its transpose.

a b c d e

[1,] 1590 1730 1870 2010 2150

[2,] 1865 2030 2195 2360 2525

[3,] 2140 2330 2520 2710 2900

[4,] 2415 2630 2845 3060 3275

[5,] 2690 2930 3170 3410 3650

#array

> a1=array(c("red","seagreen"),dim=c(3,3,2))

> a1

, , 1

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

[1,] "red" "seagreen" "red"

[2,] "seagreen" "red" "seagreen"

[3,] "red" "seagreen" "red"

, , 2

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

[1,] "seagreen" "red" "seagreen"

[2,] "red" "seagreen" "red"

[3,] "seagreen" "red" "seagreen"

> a1[1,2,2]

[1] "red"

> print(a1[,3,2])

[1] "seagreen" "red" "seagreen"

> a1[,,2]

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

[1,] "seagreen" "red" "seagreen"

[2,] "red" "seagreen" "red"

[3,] "seagreen" "red" "seagreen"

#Factors

> f=c("m","f","c","o","m","f","c","o","m","f","c","o")

> f

[1] "m" "f" "c" "o" "m" "f" "c" "o" "m" "f" "c" "o"

> factor.f=factor(f)

> factor.f

[1] m f c o m f c o m f c o

Levels: c f m o

> nlevels(factor.f)

[1] 4

#DataFrame

BMI <- data.frame(

+ gender = c("Male","Female","Male"),

+ height = c(145,136,157),

+ weight = c(56,67,89),

+ age = c(23,19,21)

+ )

> print(BMI)

gender height weight age

1 Male 145 56 23

2 Female 136 67 19

3 Male 157 89 21

> X = BMI

> nrow(X)

[1] 3

> ncol(X)

[1] 4

> dim(X)

[1] 3 4

> names(X)

[1] "gender" "height" "weight" "age"

> head(X)

gender height weight age

1 Male 145 56 23

2 Female 136 67 19

3 Male 157 89 21

> tail(X)

gender height weight age

1 Male 145 56 23

2 Female 136 67 19

3 Male 157 89 21

> # selecting specific elements

> X[2,3] # second row third col

[1] 67

> X[2] # second col gives priority if not in cordinated stuructured form

height

1 145

2 136

3 157

> X[,2] # will not show row only col

[1] 145 136 157

> X[2,] # will only show row

gender height weight age

2 Female 136 67 19

> print(X[,c(2,3)]) # show second and third col

height weight

1 145 56

2 136 67

3 157 89

> print(X[, -c(2,3)]) # show all the colummns except the ones removed in this case 2 and 3

gender age

1 Male 23

2 Female 19

3 Male 21

> # creating a data frame

> emp.data <- data.frame(

+ empid = c(1:4),

+ emp\_name = c("Sam", "Rob",

+ "Max", "John"

+ ),

+ emp\_dept = c("Sales", "RobMarketing",

+ "HR", "R&D")

+

+ )

> emp.newdata <- data.frame(

+ empid = c(1:4),

+ emp\_name = c("Sam", "Rob",

+ "Max", "John"

+ ),

+ emp\_dept = c("Sales", "RobMarketing",

+ "HR", "R&D")

+

+ )

> # rbind is for row binding only

> emp.finaldata <- rbind(emp.data,emp.newdata)

> print(emp.finaldata)

empid emp\_name emp\_dept

1 1 Sam Sales

2 2 Rob RobMarketing

3 3 Max HR

4 4 John R&D

5 1 Sam Sales

6 2 Rob RobMarketing

7 3 Max HR

8 4 John R&D

> emp.data$salary <- c(20000,30000,23000,65000)

> print(emp.data)

empid emp\_name emp\_dept salary

1 1 Sam Sales 20000

2 2 Rob RobMarketing 30000

3 3 Max HR 23000

4 4 John R&D 65000

emp.finaldata <- rbind(emp.data,emp.newdata)

> print(emp.finaldata)

empid emp\_name emp\_dept

1 1 Sam Sales

2 2 Rob RobMarketing

3 3 Max HR

4 4 John R&D

5 1 Sam Sales

6 2 Rob RobMarketing

7 3 Max HR

8 4 John R&D

> emp.data$salary <- c(27,12,22,4)

> print(emp.data)

empid emp\_name emp\_dept salary

1 1 Sam Sales 27

2 2 Rob RobMarketing 12

3 3 Max HR 22

4 4 John R&D 4

#Data Analysis

> x=read.csv("C:/Users/PRANJAL SINGH/Downloads/digit-recognizer/train.csv")

> dim(x)

[1] 42000 785

> head(x)[1:5]

label pixel0 pixel1 pixel2 pixel3

1 1 0 0 0 0

2 0 0 0 0 0

3 1 0 0 0 0

4 4 0 0 0 0

5 0 0 0 0 0

6 0 0 0 0 0

> tail(x)[1:5]

label pixel0 pixel1 pixel2 pixel3

41995 4 0 0 0 0

41996 0 0 0 0 0

41997 1 0 0 0 0

41998 7 0 0 0 0

41999 6 0 0 0 0

42000 9 0 0 0 0

> x[5,]

> x[2,1:9]

> x[,c(3,4)]

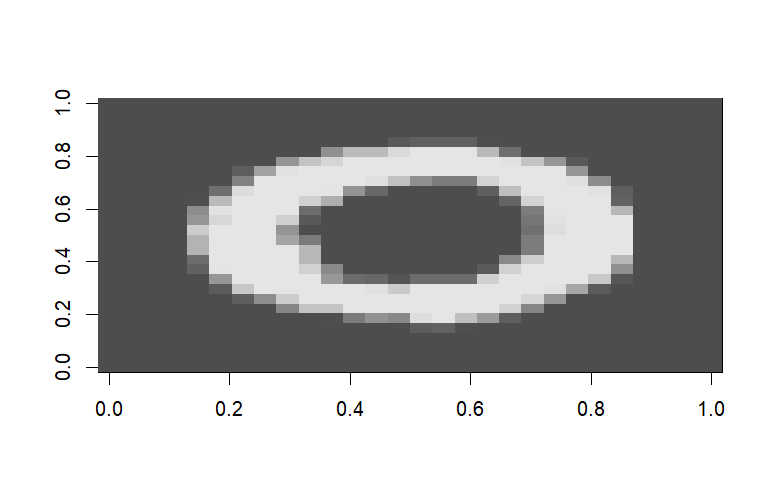
> s1=matrix(as.numeric(x[2,-1]),nrow = 28, byrow = TRUE)

> image(s1,col=grey.colors(255))

> install.packages("OpenImageR")

> library(OpenImageR)

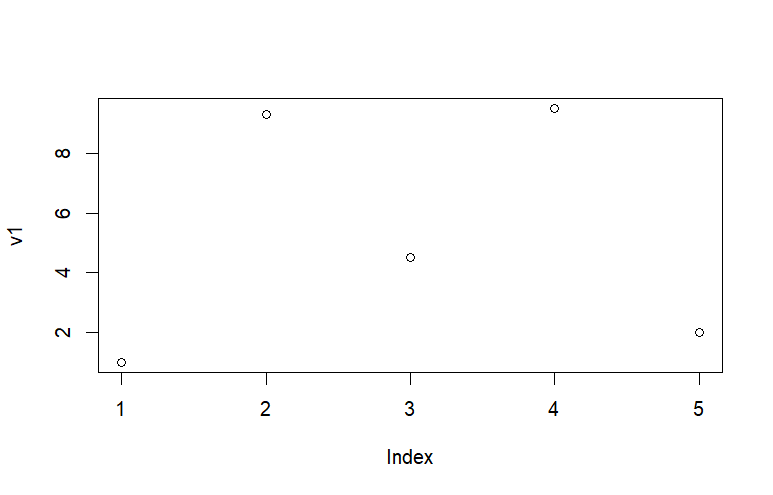
> s2=rotateFixed(s1,90)



#Plot

> v1<-c(1,9.3,4.5,9.5,2)

> plot(v1)



> v2=rep(1,10,by=3)

> length(v2)

[1] 10

> v2=rep(1,5,by=2)

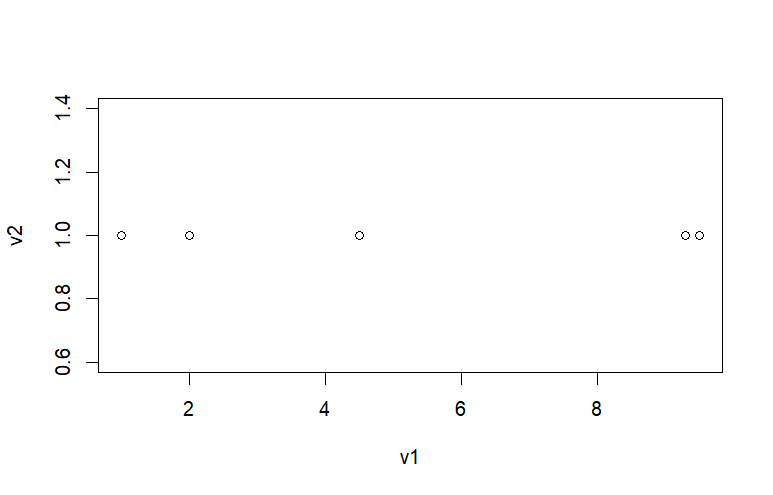
> length(v2)

[1] 5

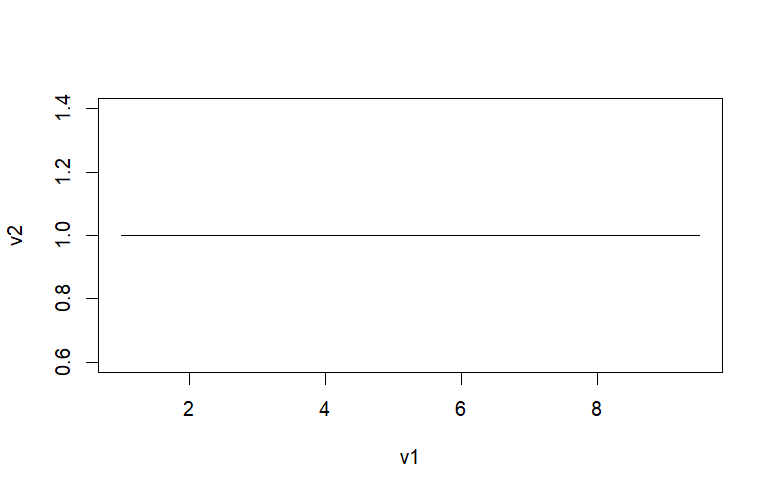
> length((v1))

[1] 5

> plot(v1,v2)



plot(v1,v2,type="l")



> v1

[1] 1.0 9.3 4.5 9.5 2.0

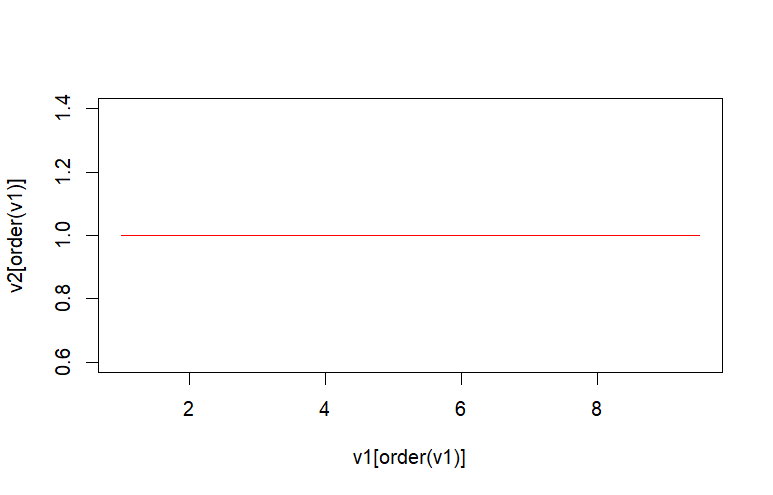
> print(order(v1))#return index of sorted

[1] 1 5 3 2 4

> print(order(v1,decreasing = FALSE))

[1] 1 5 3 2 4

> plot(v1[order(v1)],v2[order(v1)],type = "l",col="red")



#histogram plotting

> v3=c(1,1,2,3,3,4,5,6,7,8,9)

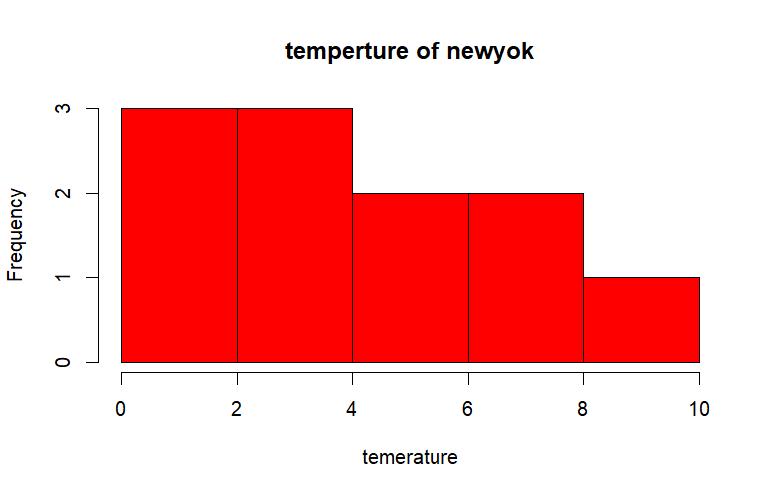
> hist(v3,

+ main="temperture of newyok",

+ xlab="temerature",

+ col="red",

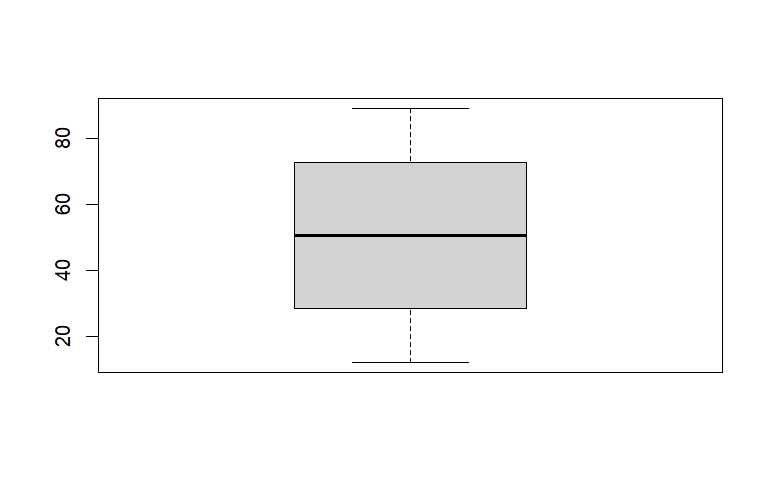
+ border="black")



> #box plot

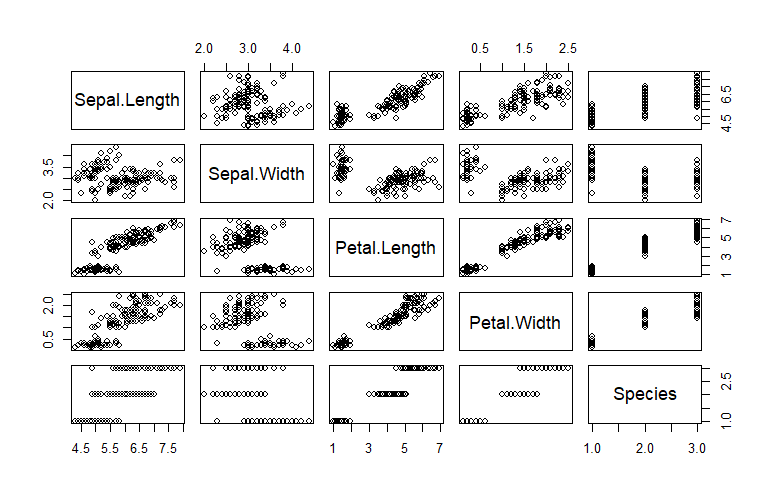
> bplo=c(12,23,34,45,56,67,78,89)

> boxplot(bplo)

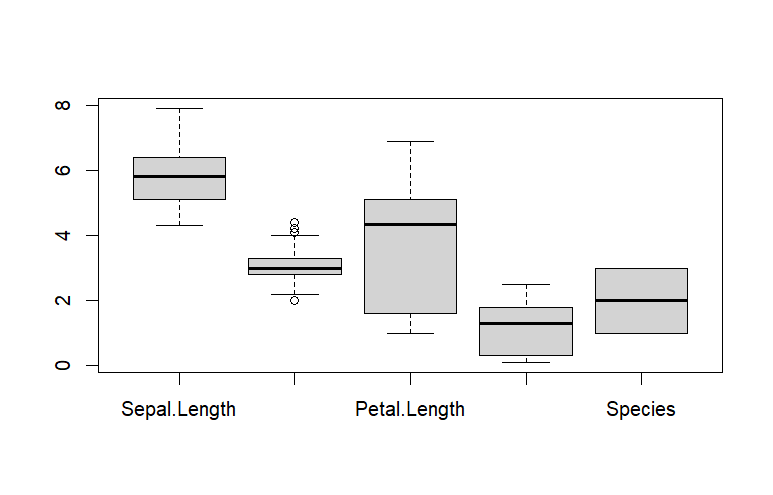


df1=iris[,1:5]

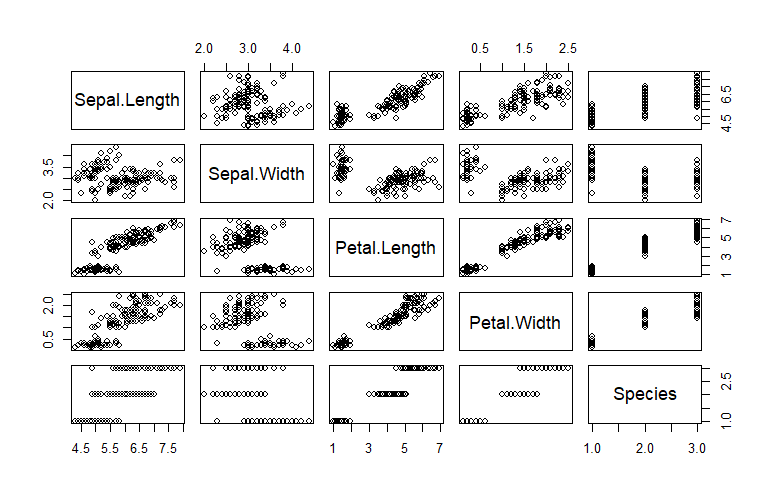
plot(df1)



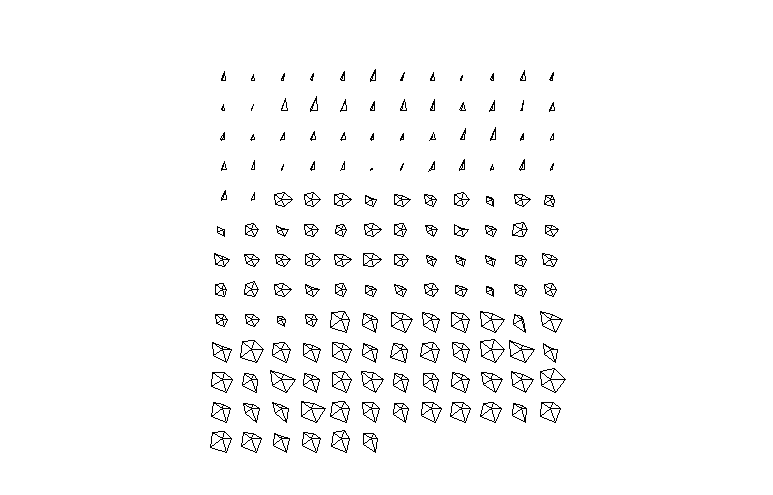
boxplot(df1)



> pairs(df1)

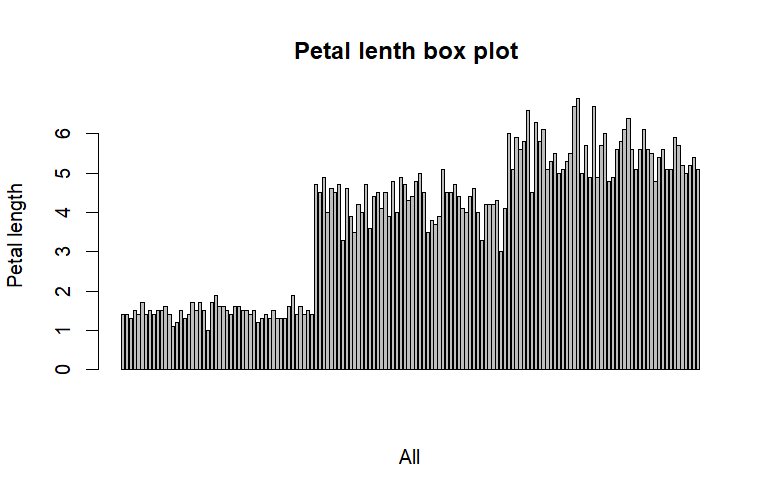


>stars(df1)

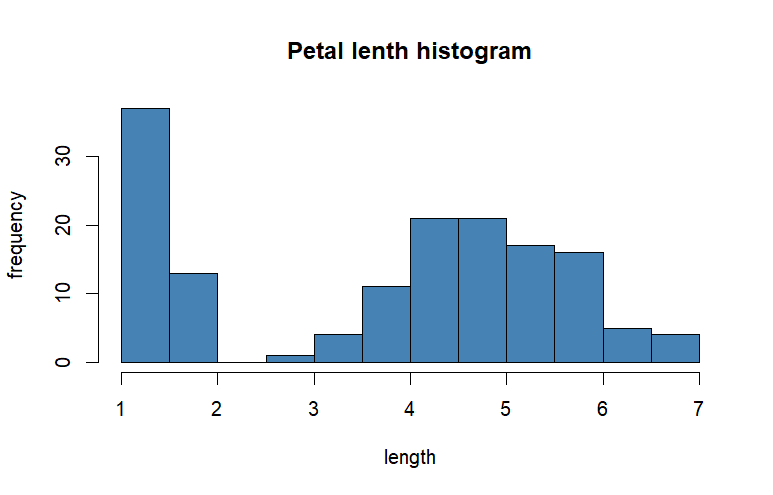


> PL=df1$Petal.Length

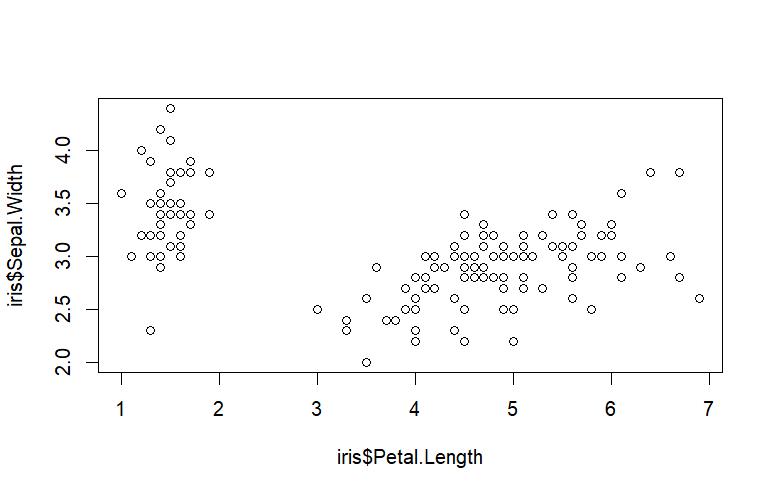
> barplot(PL,xlab="All",ylab="Petal length",main="Petal lenth box plot ")



hist(iris$Petal.Length,col = "steelblue", main="Petal lenth histogram",xlab="length",ylab = "frequency")



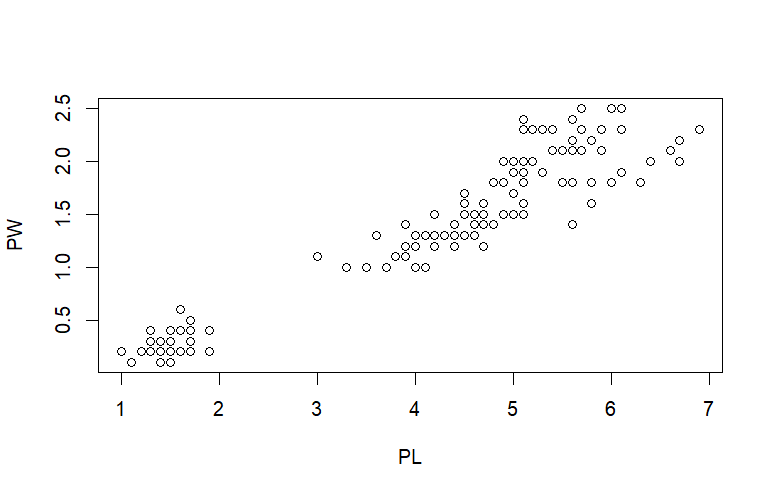
plot(iris$Petal.Length,iris$Sepal.Width)



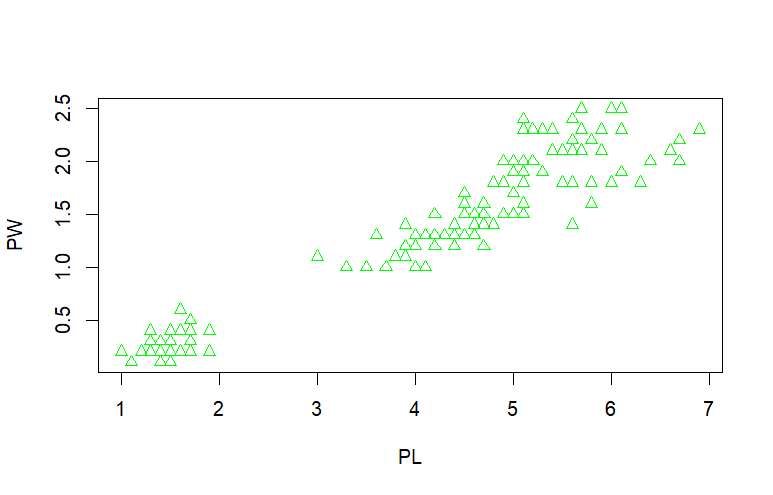
PL=iris$Petal.Length

PW=iris$Petal.Width

plot(PL,PW)



plot(PL,PW,pch=2,type = "p",col="green")



> #convert species to factors

> class(iris$Species)

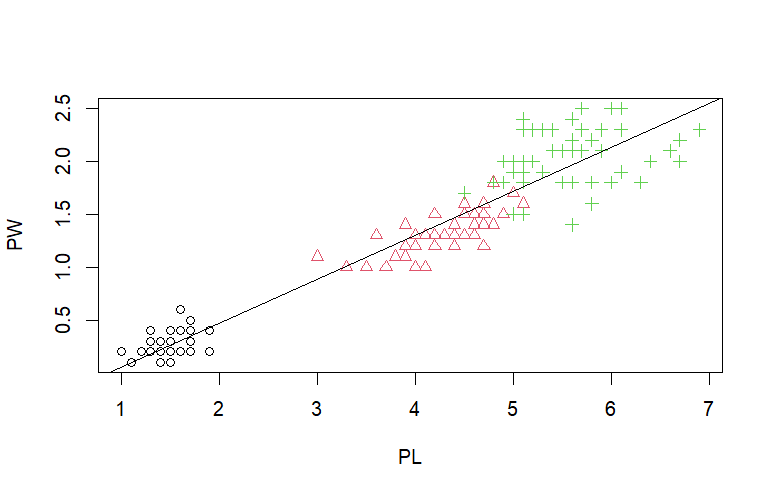
[1] "factor"

> #if you read.csv() to read csv file all character coloums() into factors

> speciesID=as.numeric(iris$Species)

plot(PL,PW,pch=speciesID,col=speciesID)

abline(lm(PW~PL))#for get line



> print(mean(iris$Sepal.Length))

[1] 5.843333

> print(median(iris$Sepal.Length))

[1] 5.8

> mode=mfv(iris$Sepal.Length)

> print(mode)

function (x)

{

if (is.expression(x))

return("expression")

if (is.call(x))

return(switch(deparse(x[[1L]])[1L], `(` = "(", "call"))

if (is.name(x))

"name"

else switch(tx <- typeof(x), double = , integer = "numeric",

closure = , builtin = , special = "function", tx)

}

<bytecode: 0x000001c6fd824b00>

<environment: namespace:base>

> print(var(iris$Sepal.Length))

[1] 0.6856935

> print(sd(iris$Sepal.Length))

[1] 0.8280661

> summary(iris)

Sepal.Length Sepal.Width Petal.Length Petal.Width Species

Min. :4.300 Min. :2.000 Min. :1.000 Min. :0.100 setosa :50

1st Qu.:5.100 1st Qu.:2.800 1st Qu.:1.600 1st Qu.:0.300 versicolor:50

Median :5.800 Median :3.000 Median :4.350 Median :1.300 virginica :50

Mean :5.843 Mean :3.057 Mean :3.758 Mean :1.199

3rd Qu.:6.400 3rd Qu.:3.300 3rd Qu.:5.100 3rd Qu.:1.800

Max. :7.900 Max. :4.400 Max. :6.900 Max. :2.500

> aggregate(spe)

> print(mean(iris$Sepal.Length))

[1] 5.843333

> print(median(iris$Sepal.Length))

[1] 5.8

> library(modeest)

> mode=mfv(iris$Sepal.Length)

> print(mode)

[1] 5

> print(var(iris$Sepal.Length))

[1] 0.6856935

> print(sd(iris$Sepal.Length))

[1] 0.8280661

> summary(iris)

Sepal.Length Sepal.Width Petal.Length Petal.Width Species

Min. :4.300 Min. :2.000 Min. :1.000 Min. :0.100 setosa :50

1st Qu.:5.100 1st Qu.:2.800 1st Qu.:1.600 1st Qu.:0.300 versicolor:50

Median :5.800 Median :3.000 Median :4.350 Median :1.300 virginica :50

Mean :5.843 Mean :3.057 Mean :3.758 Mean :1.199

3rd Qu.:6.400 3rd Qu.:3.300 3rd Qu.:5.100 3rd Qu.:1.800

Max. :7.900 Max. :4.400 Max. :6.900 Max. :2.500

> aggregate(Sepal.Length~Species,iris,mean)

Species Sepal.Length

1 setosa 5.006

2 versicolor 5.936

3 virginica 6.588

> cor(iris[,1:4])#correlation matrix

Sepal.Length Sepal.Width Petal.Length Petal.Width

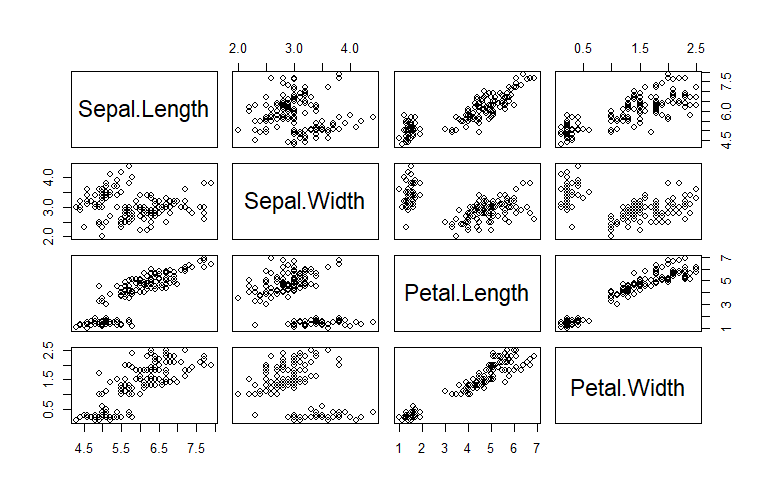
Sepal.Length 1.0000000 -0.1175698 0.8717538 0.8179411

Sepal.Width -0.1175698 1.0000000 -0.4284401 -0.3661259

Petal.Length 0.8717538 -0.4284401 1.0000000 0.9628654

Petal.Width 0.8179411 -0.3661259 0.9628654 1.0000000

> plot(iris[,1:4])



#plot correlated features

y <- iris[,"Petal.Width"]

x <- iris[,"Petal.Length"]

head(x)

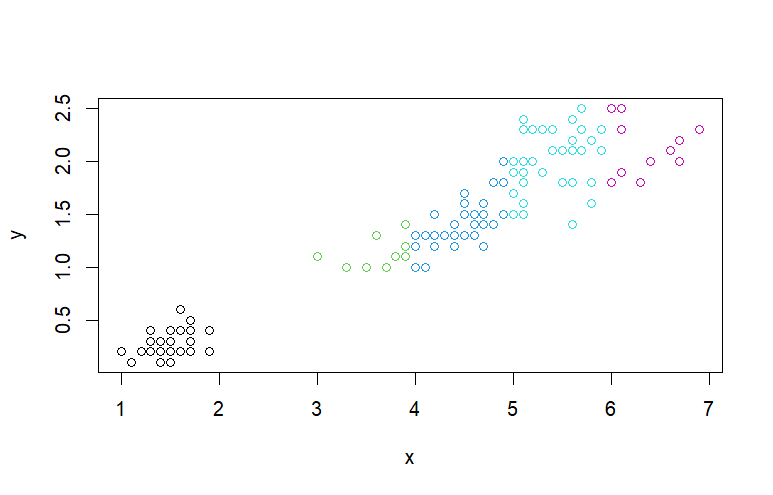
head(y)

xycorr <-cor(y,x,method = "pearson") # to find the corelation

xycorr

species\_col = as.numeric(iris$Species)

plot(y~x,col = x)



> modal1 <- lm(y~x)#to find slope and intercept

> modal1 #provides regression line coefficient only

Call:

lm(formula = y ~ x)

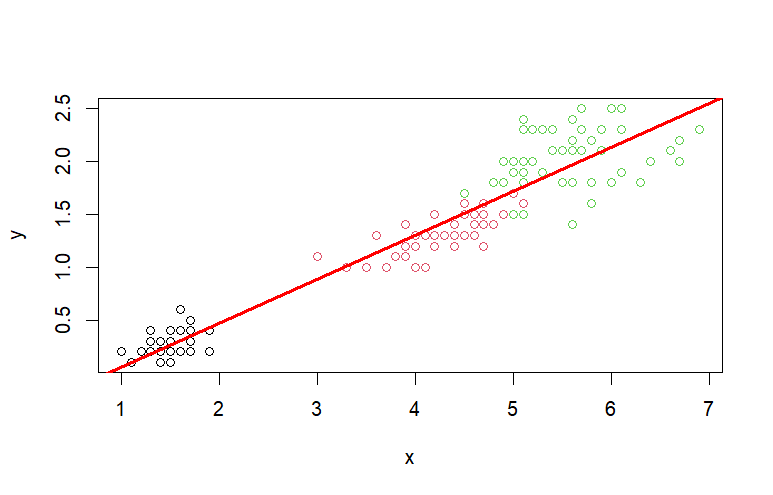
Coefficients:

(Intercept) x

-0.3631 0.4158

> plot(y~x,col = species\_col)

abline(modal1,col = "red",lwd = 4)# to draw the line



> p1 <- predict(modal1, data.frame("x" = 5))

> p1

1

1.715702

> irdata = iris

> str(irdata)

'data.frame': 150 obs. of 5 variables:

$ Sepal.Length: num 5.1 4.9 4.7 4.6 5 5.4 4.6 5 4.4 4.9 ...

$ Sepal.Width : num 3.5 3 3.2 3.1 3.6 3.9 3.4 3.4 2.9 3.1 ...

$ Petal.Length: num 1.4 1.4 1.3 1.5 1.4 1.7 1.4 1.5 1.4 1.5 ...

$ Petal.Width : num 0.2 0.2 0.2 0.2 0.2 0.4 0.3 0.2 0.2 0.1 ...

$ Species : Factor w/ 3 levels "setosa","versicolor",..: 1 1 1 1 1 1 1 1 1 1 ...

> levels(irdata$Species)

[1] "setosa" "versicolor" "virginica"

> sum(is.na(irdata))

[1] 0

> irdata <- irdata[1:100,]#taking inly two species setosha

> samp <- sample(1:100,80)

> irtrain <- irdata[samp,]

> str(irtrain)

'data.frame': 80 obs. of 5 variables:

$ Sepal.Length: num 5 4.9 6 6.3 5.1 5.1 6.9 5.5 5.8 6.7 ...

$ Sepal.Width : num 3 2.4 3.4 3.3 2.5 3.5 3.1 2.5 2.6 3 ...

$ Petal.Length: num 1.6 3.3 4.5 4.7 3 1.4 4.9 4 4 5 ...

$ Petal.Width : num 0.2 1 1.6 1.6 1.1 0.3 1.5 1.3 1.2 1.7 ...

$ Species : Factor w/ 3 levels "setosa","versicolor",..: 1 2 2 2 2 1 2 2 2 2 ...

> irtest <- irdata[-samp,]

> str(irtest)

'data.frame': 20 obs. of 5 variables:

$ Sepal.Length: num 5.4 5.4 5.7 5.4 4.6 4.8 5 5.2 4.8 4.4 ...

$ Sepal.Width : num 3.9 3.7 4.4 3.4 3.6 3.4 3.4 3.4 3.1 3.2 ...

$ Petal.Length: num 1.7 1.5 1.5 1.7 1 1.9 1.6 1.4 1.6 1.3 ...

$ Petal.Width : num 0.4 0.2 0.4 0.2 0.2 0.2 0.4 0.2 0.2 0.2 ...

$ Species : Factor w/ 3 levels "setosa","versicolor",..: 1 1 1 1 1 1 1 1 1 1 ...

> y <- irtrain$Species

> x <- irtrain$Sepal.Width

> glfit <- glm(y~x,family ="binomial")

> summary(glfit)

Call:

glm(formula = y ~ x, family = "binomial")

Coefficients:

Estimate Std. Error z value Pr(>|z|)

(Intercept) 17.630 4.127 4.272 1.94e-05 \*\*\*

x -5.697 1.337 -4.261 2.04e-05 \*\*\*

---

Signif. codes: 0 ‘\*\*\*’ 0.001 ‘\*\*’ 0.01 ‘\*’ 0.05 ‘.’ 0.1 ‘ ’ 1

(Dispersion parameter for binomial family taken to be 1)

Null deviance: 110.703 on 79 degrees of freedom

Residual deviance: 65.247 on 78 degrees of freedom

AIC: 69.247

Number of Fisher Scoring iterations: 6

> newdata=data.frame(x=2.0)

> predicted\_val=predict(glfit,newdata,type="response")

> predicted\_val

1

0.9980456

> newdata<-data.frame(x=ir\_test$Sepal.Width)

> predicted\_val<-predict(glfit,newdata,type="response")

> predicted\_val

1

0.9980456

> predict\_reg<- ifelse(predicted\_val>05, print("Versicolor"),print("Setosa"))

[1] "Setosa"

> predict\_reg <- ifelse(predicted\_val > 0.5 ,2,1)

> table(as.numeric(ir\_test$Species), predict\_reg)

> predict\_reg

1

2

> prediction<-data.frame()