Laboratorio 2, Tópicos en análisis datos 1

Joshua Isaac Cervantes Artavia

2023-09-02

tryCatch(  
#To set the default folder, it is going to be the folder where is the script  
{  
 directory <- dirname(rstudioapi::getSourceEditorContext()$path)  
setwd(directory)  
},  
error = function(e){  
  
}  
)

NULL

# 1

# Se el dataset  
df\_europa <- read.table("europa.csv", dec = ".", sep = ";", header = TRUE)

Se verifican los datos:

names(df\_europa)

[1] "Pais" "Duracion.horas."

head(df\_europa)

Pais Duracion.horas.  
1 Alemania 41.7  
2 Austria 44.1  
3 Belgica 41.0  
4 Chipre 41.8  
5 Dinamarca 40.5  
6 Eslovaquia 41.6

str(df\_europa)

'data.frame': 25 obs. of 2 variables:  
 $ Pais : chr "Alemania" "Austria" "Belgica" "Chipre" ...  
 $ Duracion.horas.: num 41.7 44.1 41 41.8 40.5 41.6 42.5 42.2 41.5 40.5 ...

summary(df\_europa)

Pais Duracion.horas.  
 Length:25 Min. :39.8   
 Class :character 1st Qu.:41.0   
 Mode :character Median :41.5   
 Mean :41.7   
 3rd Qu.:42.5   
 Max. :44.1

range(df\_europa$Duracion.horas.)

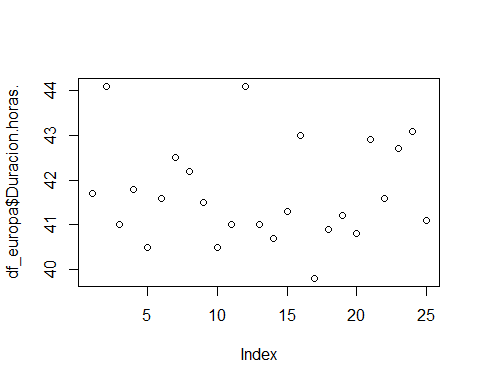
[1] 39.8 44.1

sd(df\_europa$Duracion.horas.)

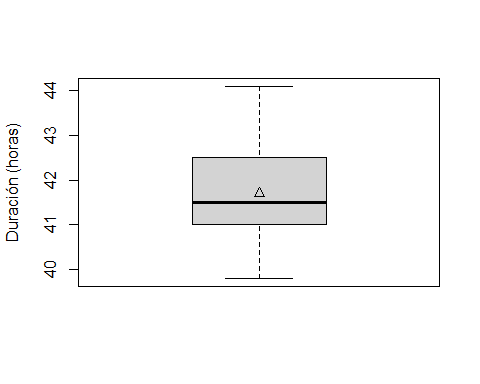
[1] 1.113358

Se realizan algunos gráficos

plot(df\_europa$Duracion.horas.)



boxplot(df\_europa$Duracion.horas., ylab = "Duración (horas)")  
points(1, mean(df\_europa$Duracion.horas.), pch = 2)



# 2

a-

library(MASS)  
data(iris)  
iris\_numeric <- iris[, sapply(iris, is.numeric)]  
dim(iris)

[1] 150 5

Se puede observar que tiene 150 filas y 5 columnas.

Los nombres de las variables son

colnames(iris)

[1] "Sepal.Length" "Sepal.Width" "Petal.Length" "Petal.Width" "Species"

La muestra son 50 observaciones.

b-

### Media

apply(iris\_numeric, 2, mean)

Sepal.Length Sepal.Width Petal.Length Petal.Width   
 5.843333 3.057333 3.758000 1.199333

### Mediana

apply(iris\_numeric, 2, median)

Sepal.Length Sepal.Width Petal.Length Petal.Width   
 5.80 3.00 4.35 1.30

### Desviación estándar

apply(iris\_numeric, 2, sd)

Sepal.Length Sepal.Width Petal.Length Petal.Width   
 0.8280661 0.4358663 1.7652982 0.7622377

### Cuartiles

apply(iris\_numeric, 2, quantile, probs = c(0.25, 0.75))

Sepal.Length Sepal.Width Petal.Length Petal.Width  
25% 5.1 2.8 1.6 0.3  
75% 6.4 3.3 5.1 1.8

### Máximo

apply(iris\_numeric, 2, max)

Sepal.Length Sepal.Width Petal.Length Petal.Width   
 7.9 4.4 6.9 2.5

### Mínimo

apply(iris\_numeric, 2, min)

Sepal.Length Sepal.Width Petal.Length Petal.Width   
 4.3 2.0 1.0 0.1

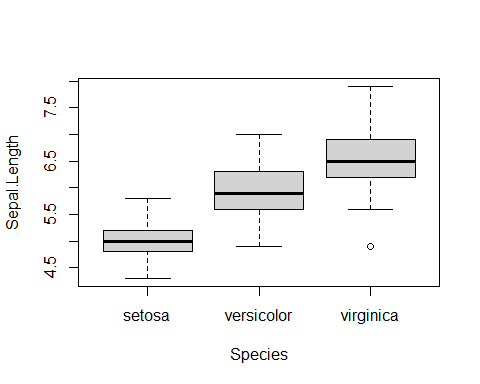
### Rango

apply(iris\_numeric, 2, function(x) max(x) - min(x))

Sepal.Length Sepal.Width Petal.Length Petal.Width   
 3.6 2.4 5.9 2.4

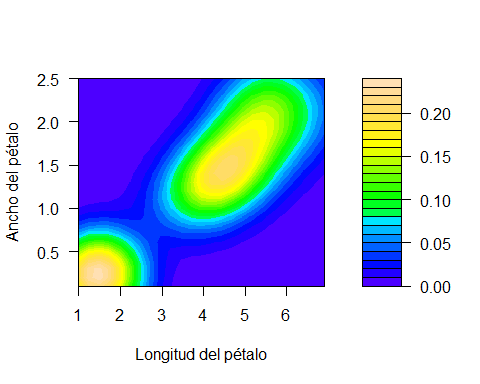
c-

boxplot(Sepal.Length ~ Species, data = iris)



d-

# Se estima un kernel dos dimensional para los datos  
densidad <- kde2d(iris$Petal.Length, iris$Petal.Width)  
filled.contour(densidad, color = topo.colors, xlab = "Longitud del pétalo", ylab = "Ancho del pétalo")



# 3

a-

Creación de matrices

m1 <- matrix(1, nr = 2, nc = 2)  
m2 <- matrix(2, nr = 2, nc = 2)  
  
# Se coloca una matriz debajo de la otra (concatenacion por fila)  
m <- rbind(m1, m2)  
m

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

# Concatenacion por columna  
cbind(m1, m2)

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

b-

Muestreo con repetición

a <- sample(1:10, 30, rep = TRUE)  
a <- matrix(a, nr = 6)  
  
# Hacen 0 los numeros menores de 3  
a[a < 3] <- 0  
  
# Se extrae la diagonal  
diag(a)

[1] 4 10 6 7 8

# Se crea la matriz diagonal 5x5  
diag(5)

[,1] [,2] [,3] [,4] [,5]  
[1,] 1 0 0 0 0  
[2,] 0 1 0 0 0  
[3,] 0 0 1 0 0  
[4,] 0 0 0 1 0  
[5,] 0 0 0 0 1

c-

# Se suman los elementos de la matriz m1  
sum(m1)

[1] 4

# Se suma por columna  
apply(m, 2, sum)

[1] 6 6

# Se suma por filas  
apply(m, 1, sum)

[1] 2 2 4 4

d-

Operación de matrices

# Suma  
  
m1 + m2

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

# Resta  
  
m1 - m2

[,1] [,2]  
[1,] -1 -1  
[2,] -1 -1

# Producto  
  
m1 %\*% m2

[,1] [,2]  
[1,] 4 4  
[2,] 4 4

# Producto ii  
  
m1 \* m2

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

Transposición de la matriz

a

[,1] [,2] [,3] [,4] [,5]  
[1,] 4 6 7 7 9  
[2,] 10 10 7 3 0  
[3,] 6 9 6 4 5  
[4,] 5 3 5 7 0  
[5,] 4 9 3 4 8  
[6,] 9 0 7 0 0

t(a)

[,1] [,2] [,3] [,4] [,5] [,6]  
[1,] 4 10 6 5 4 9  
[2,] 6 10 9 3 9 0  
[3,] 7 7 6 5 3 7  
[4,] 7 3 4 7 4 0  
[5,] 9 0 5 0 8 0

e-

Se construye la matriz solicitada

z <- matrix(1:12, nc = 4, byrow = TRUE)

# Elemento 1,3  
z[1, 3]

[1] 3

# Primera fila  
z[1, ]

[1] 1 2 3 4

# Tercera columna  
z[, 3]

[1] 3 7 11

# Se selecciona la submatriz de eliminar la primera fila y la primera columna  
(z\_submatriz <- z[-1, -1])

[,1] [,2] [,3]  
[1,] 6 7 8  
[2,] 10 11 12

# Transpuesta de z  
t(z)

[,1] [,2] [,3]  
[1,] 1 5 9  
[2,] 2 6 10  
[3,] 3 7 11  
[4,] 4 8 12

# Subamtriz al quitar la segunda columna de z, calcular la inversa  
  
(z\_sub\_matriz\_sin\_segunda\_col <- z[, -2])

[,1] [,2] [,3]  
[1,] 1 3 4  
[2,] 5 7 8  
[3,] 9 11 12

tryCatch(  
 {  
 (z\_sub\_matriz\_sin\_segunda\_col\_inv <- solve(z\_sub\_matriz\_sin\_segunda\_col))  
 },  
 error = function(e) {  
 message("La matriz no es invertible el determinate es")  
 print(det(z\_sub\_matriz\_sin\_segunda\_col))  
 }  
)

La matriz no es invertible el determinate es

[1] 3.552714e-15

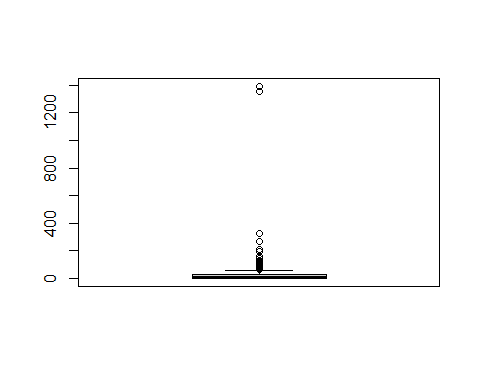
# 4

if (!("Lock5Data" %in% installed.packages())) {  
 install.packages("Lock5Data")  
}  
library(Lock5Data)  
  
# Se cargan los datos del dataset AllCountries  
  
data(AllCountries)  
  
fn\_sample\_mean <- function(sample\_size, df, variable, ...) {  
 # Se escoge una muestra de acuerdo al tamano indicado  
 sm <- sample(1:nrow(df), sample\_size, ...)  
  
 # Se toman las filas y la variable deseada  
 sm <- df[sm, variable]  
  
 # Se imprime un mensaje de que se tenga un NA  
 if (sum(is.na(sm)) > 0) print("El vector ejemplo contenia NA, se han eliminado para estimar el promedio")  
  
 # Se muestra el promedio de la muestra  
 mean(sm, na.rm = TRUE)  
}  
  
fn\_sample\_mean(10, AllCountries, "LandArea", rep = FALSE)

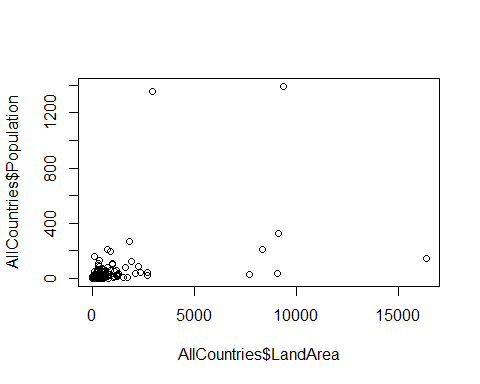
[1] "El vector ejemplo contenia NA, se han eliminado para estimar el promedio"

[1] 411.338

# Se hace un bloxplot de la variable de poblacion  
boxplot(AllCountries$Population)



# Se hace un grafico de dispersion de superficie vs poblacion  
plot(AllCountries$LandArea, AllCountries$Population)



# 5

df\_ejercicio\_1\_6 <- data.frame(sexo = c("masculino", "masculino", rep("femenino", 4), "masculino", "masculino", "femenino"), profesion = c("oficinista", "obrero", "obrero", "artista", "artista", "oficinista", "artista", "oficinista", "artista"))  
  
fn\_add\_column\_0\_1 <- function(variable) {  
 # We check the name of the variable  
 name\_variable <- colnames(variable)  
  
 # Unique values  
 values <- unique(variable)  
  
 df\_aux <- data.frame(rep(0, length(variable)))  
  
 colnames(df\_aux) <- values[1]  
  
 for (i in values) {  
 df\_aux[[i]] <- ifelse(variable == i, 1, 0)  
 }  
 return(df\_aux)  
}  
  
fn\_add\_column\_0\_1(c("masculino", "masculino", rep("femenino", 4), "masculino", "masculino", "femenino"))

masculino femenino  
1 1 0  
2 1 0  
3 0 1  
4 0 1  
5 0 1  
6 0 1  
7 1 0  
8 1 0  
9 0 1

(list\_disjunctive\_tables <- apply(df\_ejercicio\_1\_6, 2, fn\_add\_column\_0\_1))

$sexo  
 masculino femenino  
1 1 0  
2 1 0  
3 0 1  
4 0 1  
5 0 1  
6 0 1  
7 1 0  
8 1 0  
9 0 1  
  
$profesion  
 oficinista obrero artista  
1 1 0 0  
2 0 1 0  
3 0 1 0  
4 0 0 1  
5 0 0 1  
6 1 0 0  
7 0 0 1  
8 1 0 0  
9 0 0 1

# 6

data(HairEyeColor)  
  
ji\_squared <- function(contingency\_table) {  
 # We sum all to get xdd  
 x\_dot\_dot <- sum(contingency\_table)  
  
 x\_j\_k\_squared <- contingency\_table \* contingency\_table  
  
 q <- apply(contingency\_table, 2, sum)  
 p <- apply(contingency\_table, 1, sum)  
  
 x\_j\_dot\_X\_x\_dot\_k <- sapply(q, FUN = function(x) x \* p)  
  
 elements\_sum <- x\_j\_k\_squared <- x\_j\_k\_squared / x\_j\_dot\_X\_x\_dot\_k  
  
 return(x\_dot\_dot \* sum(elements\_sum) - x\_dot\_dot)  
}  
  
ji\_squared(HairEyeColor[, , 1] + HairEyeColor[, , 2])

[1] 138.2898

T\_squared <- function(contingency\_table) {  
 # Se estima la suma de toda la tabla  
 x\_dot\_dot <- sum(contingency\_table)  
  
 # Se estima el numero de filas que coincide con el numero de modalidades de una de la variables  
 rows <- nrow(contingency\_table)  
  
 # Se estima el numero de columnas que coincide con el numero de modalidades de una de la variables  
 cols <- ncol(contingency\_table)  
  
 # Se retorna el T cuadrado  
 return(ji\_squared(contingency\_table) / (x\_dot\_dot \* (rows - 1) \* (cols - 1)))  
}  
  
# Se estima el T cuadrado de Chuprov sin diferenciar sexo  
T\_squared(HairEyeColor[, , 1] + HairEyeColor[, , 2])

[1] 0.0259553

# 7

# Funcion para estimar el coeficiente de correlacion  
# Recibe la tabla, la variable es el nombre de la variable de la que se quiere estimar el coeficiente  
# modalities son las clases  
eta <- function(df, variable, modalities) {  
 # Se ve cuales modalidades hay  
 different\_modalities <- unique(df[, modalities])  
  
 # Se selecciona el vector de modalidades  
 modalities <- df[, modalities]  
  
 # Se seleccionan los valores de la variable  
 values <- df[, variable]  
  
 # Suma de la media  
 sum\_means <- 0  
  
 # Se estima la sumatoria de la media de las modalidades por su peso  
 for (i in different\_modalities) {  
 # Promedio de la modalidad j  
 mean\_x\_j <- mean(values[modalities == i])  
  
 # Peso del promedio  
 eta\_j <- sum(modalities == i) / nrow(df)  
  
 # Se agrega a la sumatoria  
 sum\_means <- sum\_means + eta\_j \* mean\_x\_j^2  
 }  
  
 # Se estima coeficiente  
 return((sum\_means - mean(values)^2) / ((length(values) - 1) \* var(values) / length(values)))  
}  
  
# Se estima el coeficiente para cada variable cuantitativa  
sapply(colnames(iris)[-ncol(iris)], FUN = eta, df = iris, modalities = "Species")

Sepal.Length Sepal.Width Petal.Length Petal.Width   
 0.6187057 0.4007828 0.9413717 0.9288829