Complementos

Sesión 4

Natalie Julian - www.nataliejulian.com

Estadística UC y Data Scientist en Zippedi Inc.

Pokemon

El archivo `pokemon.RData` contiene información de pokemones:

- Nombre del pokemon
- Tipo del pokemon
- Si el pokemon es legendario o no
- Velocidad de ataque del pokemon

Practica 1

a) Cargue el archivo pokemon.RData. ¿Qué información se ha cargado? ¿cuántos registros se poseen? ¿cuál es el formato de lactura de cada variable? ¿está correcto el formato de las variables? Comente.

b) Asegúrese de que cada pokemon aparezca una y solo una vez.

c) ¿Cuántos pokemones de tipo legendario hay? Proponga dos maneras diferentes de determinarlo.

d) Obtenga un resumen estadístico de la velocidad de ataque.

e) Determine cuáles son los primeros 6 pokemones que tienen mayor velocidad de ataque. ¿Cuántos y cuáles son los pokemones con velocidades de ataque superiores al tercer cuartil?. Elija 4 de estos pokemones e indique si estos pokemones son de tipo legendario o no.

f) Obtenga resúmenes estadísticos de la velocidad de ataque de aquellos pokemones del tipo Grass. Compare las estadísticas con aquellos pokemones de tipo Electric. Comente.

g) ¿Cómo se distribuyen los pokemones legendarios dentro del tipo de pokemon Ice? Compare con el tipo de pokemon Fighting. Comente.

Respuestas Práctica 1

Respuesta práctica 1a)

```
#a)

load(file.choose())

# Si se quiere direccionar a ruta, por ejemplo considere (en mi caso)

#load("C:/Users/HP/Desktop/Trabajo/2020-2/Laboratorio I. Estadística/Sesiones-Semanas/S4 Matrices/pokemon.RData")


#Se han cargado 4 vectores
ls()  #Indica los nombres de los objetos que hay en la sesión
[1] "legendario"  "pokemon"     "speedattack" "tipo"

class(legendario) #character
[1] "character"

class(pokemon) #character
[1] "character"

class(speedattack) #numeric
[1] "numeric"

class(tipo) #character
[1] "character"

#La unica variable con información de tipo numérica es
#la variable speedattack, se ha leido correctamente, la información.

length(legendario);length(pokemon);length(speedattack);length(tipo) #800 registros de pokemones
[1] 800
[1] 800
[1] 800
[1] 800
```

Respuesta práctica 1b)

```
#Con table podemos obtener una tabla de frecuencias enorme de los nombres de cada pokemon

table(pokemon)

#Dos maneras de probar rapidamente que cada nombre aparece una vez:


#FORMA 1

table(table(pokemon)) #Si solo aparece el valor 1 indica que cada nombre se repite solo una vez
  1
800

#FORMA 2
as.numeric(table(pokemon)) #Entrega la frecuencia de los nombres en un vector

as.numeric(table(pokemon))>1 #Podemos preguntarle si alguna frecuencia es mayor a 1

which(as.numeric(table(pokemon))>1) #Ninguna es mayor que 1, es decir, ningun pokemon se repite
integer(0)

#Esto es muy util para verificar que ningún ID se repita!
```

Respuesta práctica 1c)

```
#c)

#FORMA 1)

#Se puede realizar una tabla de frecuencias

table(legendario)
legendario
False  True
  735    65

#65 pokemones de tipo legendario

#Si quieren extraer un valor de una tabla:

as.numeric(table(legendario)) #Entrega las frecuencias en vector
[1] 735  65

as.numeric(table(legendario))[2] #Entrega frecuencia de casos True o legendarios
[1] 65

#FORMA 2)

length(which(legendario=="True"))
[1] 65

#FORMA 3)

sum(legendario=="True")
[1] 65{

#FORMA 4)

                                e")
```

Respuesta práctica 1d)

```
summary(speedattack)#Estadísticas descriptivas para variables numéricas
   Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
  10.00   49.75   65.00   72.82   95.00  194.00

range(speedattack) #Función sugerida por un compañero!
[1]  10 194
```

Respuesta práctica 1e)

```
#e)

pokemon[order(speedattack, decreasing=TRUE)] #Ordena los pokemones de mayor a menor velocidad de ataque

#Quiero los primeros 6 (los que atacan mas rapido)

head(pokemon[order(speedattack, decreasing=TRUE)])
[1] "MewtwoMega Mewtwo Y"   "KyogrePrimal Kyogre"   "RayquazaMega Rayquaza"
[4] "DeoxysAttack Forme"    "AlakazamMega Alakazam" "GengarMega Gengar"

#Determinar aquellos cuya velocidad de ataque sea superior al 3er cuartil:
#El tercer cuartil es Q tal que P(X<=Q)=0.75

which(speedattack>quantile(speedattack, 0.75))

#¿Cuántos?
length(which(speedattack>quantile(speedattack, 0.75))) #181
[1] 181

#¿Cuáles?

pokemon[which(speedattack>quantile(speedattack, 0.75))] #Nombres
```

Respuesta práctica 1e)

```
#Elegire los pokemones:

#Sunflora
#BlazikenMega Blaziken
#Xerneas
#Aurorus

#¿Son legendarios?

#FORMA A) (larga)

legendario[which(pokemon=="Sunflora")]
[1] "False"

legendario[which(pokemon=="BlazikenMega Blaziken")]
[1] "False"

legendario[which(pokemon=="Xerneas")] #Es legendario
[1] "True"

legendario[which(pokemon=="Aurorus")]
[1] "False"

#FORMA B) (operador o inclusivo)

legendario[which(pokemon=="Sunflora"|pokemon=="BlazikenMega Blaziken"|pokemon=="Xerneas"|pokemon=="Aurorus")]
[1] "False" "False" "False" "True"

pokemon[which(pokemon=="Sunflora"|pokemon=="BlazikenMega Blaziken"|pokemon=="Xerneas"|pokemon=="Aurorus")]
[1] "Sunflora"              "BlazikenMega Blaziken" "Aurorus"
[4] "Xerneas"
```

Respuesta práctica 1f)

```
unique(tipo)
[1] "Grass"    "Fire"     "Water"    "Bug"      "Normal"   "Poison"   "Electric" "Ground"
 [9] "Fairy"    "Fighting" "Psychic"  "Rock"     "Ghost"    "Ice"      "Dragon"   "Dark"
[17] "Steel"    "Flying"

#Para grass

speedattack[which(tipo=="Grass")] #Filtra a pokemones de tipo grass

speedattack[which(tipo=="Electric")] #Filtra a pokemones de tipo Electric

summary(speedattack[which(tipo=="Grass")]) #Resumen estadístico de velocidad de ataque para grupo grass
   Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
   24.0    57.0    75.0    77.5    99.5   145.0

summary(speedattack[which(tipo=="Electric")]) #Resumen estadístico de velocidad de ataque para grupo Electric
   Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
  35.00   65.00   95.00   90.02  106.00  165.00

#La velocidad de ataque tiende a obtener mayores estadísticas para el tipo electric (en comparacion a grass)
```

Respuesta práctica 1g)

```
#g)

table(legendario[which(tipo=="Ice")])
False  True
   22     2

table(legendario[which(tipo=="Fighting")])
False
   27

#En el tipo Ice tiene 24 pokemones en total y 2 son legendarios

#En el tipo Fighting hay 27 pokemones en total y ninguno es legendario
```

Ejercicio adicional

¿Cómo hacer que el vector legendario se lea como vector de valores logicos?

Notar que R distingue entre True y TRUE, es por esto que True no se reconoce como el valor logico TRUE. Esto se puede modificar de la siguiente manera:

```
#Forma 1:

legendario[which(legendario=="True")]<-"TRUE" #Casos True a TRUE
legendario[which(legendario=="False")]<-"FALSE" #Casos False a FALSE

legendario<-as.logical(legendario) #Definimos el formato de caracter a logical

class(legendario)
[1] "logical"

sum(legendario) #TRUE==1, suma de un vector logico indica la suma de valores TRUE
[1] 65

#Forma 2:

legendario<-ifelse(legendario=="True", "TRUE", "FALSE") #si es True lollena con TRUE, sino con FALSE

class(legendario)
[1] "logical"

sum(legendario)
[1] 65
```

¿Cómo hacer que el vector legendario se lea como vector de valores lógicos?

```
#Forma 3:

legendario<-toupper(legendario) #Pasa todo a mayúsculas

legendario<-as.logical(legendario)

class(legendario)
[1] "logical"

sum(legendario)
[1] 65
```

Definiendo una matriz de datos con vectores

Supongamos que tenemos las mediciones de peso (en kilogramos) y altura (en centímetros) de 5 pacientes:

Peso (kg)	Altura (cms)
55	163
90	174
80	163
45	150
59	147

Como usted sabe, esto se puede definir en R utilizando ciertos comandos.

Usando cbind()

cbind() ordena la información por columnas (variables) de manera vertical:

```
peso<-c(55,90,80,45,59)   #Se define la información por columna
altura<-c(163,174,163,150,147)

datosc<-cbind(peso,altura)
datosc
```

```
    peso altura
[1,] 55 163
[2,] 90 174
[3,] 80 163
[4,] 45 150
[5,] 59 147
```

Usando rbind()

rbind() ordena la información por filas (registros) de manera horizontal:

```
paciente1<-c(55,163)    #Se define la información por fila
paciente2<-c(90,174)
paciente3<-c(80,163)
paciente4<-c(45,150)
paciente5<-c(59,147)

(datosf<-rbind(paciente1,paciente2,paciente3,paciente4,paciente5))

        [,1] [,2]
paciente1 55 163
paciente2 90 174
paciente3 80 163
paciente4 45 150
paciente5 59 147
```

Otros comandos para definir una matriz de datos

El comando `matrix()` al igual que los comandos `cbind()` o `rbind()` se utiliza cuando tenemos todas las variables del mismo tipo (todas `character` o todas `numeric`). En este caso, es necesario entregarle el argumento *nrow=5* y *ncol=2* para indicar que la matriz es tal que tiene 5 filas y 2 columnas:

```
datosm<-matrix(c(peso,altura),nrow=5,ncol=2)
datosm
```

```
 [,1] [,2]
[1,] 55 163
[2,] 90 174
[3,] 80 163
[4,] 45 150
[5,] 59 147
```

Definiendo los datos manualmente en una matriz

Es posible ir definiendo los datos directamente en la función a utilizar:

```
matrix(c(55,90,80,45,59,163,174,163,150,147),ncol=2,nrow=5)
     [,1] [,2]
[1,]   55  163
[2,]   90  174
[3,]   80  163
[4,]   45  150
[5,]   59  147

matrix(c(55,163,90,174,80,163,45,150,59,147),byrow=TRUE,ncol=2,nrow=5)
     [,1] [,2]
[1,]   55  163
[2,]   90  174
[3,]   80  163
[4,]   45  150
[5,]   59  147
```

Definiendo un conjunto de datos usando `data.frame()`

```
#Aquí se definen las variables peso y altura dentro del comando data.frame()

(DF<-data.frame("peso"=c(55,90,80,45,59), "altura"=c(163,173,163,150,147)))

 peso altura
1 55   163
2 90   173
3 80   163
4 45   150
5 59   147
```

Cambiando los nombres de las columnas de una tabla de datos

Muchas veces, los nombres de las columnas de una base de datos están en inglés o no son muy específicos. En este caso, considere que es relevante indicar que el peso se encuentra en kilogramos y que la altura se encuentra en centímetros. Para modificar el nombre de las columnas de un objeto de del tipo data.frame como el objeto `DF` definido anteriormente se utiliza el comando `colnames()`:

```
colnames(DF)<-c("peso (kg)","altura (cms)")

DF
    peso (kg)  altura (cms)
1      55          163
2      90          174
3      80          163
4      45          150
5      59          147
```

Cambiando los nombres de las filas de una planilla de datos

En este caso, tenemos que cada fila corresponde a la información de un paciente diferente, en este caso, al ser pocos pacientes, la idea de añadirle un nombre a cada fila de forma manual es factible, y se realiza de la siguiente forma:

```
rownames(DF)<-c("Paciente 1","Paciente 2","Paciente 3","Paciente 4","Paciente 5")

DF

           peso (kg)  altura (cms)
Paciente 1     55        163
Paciente 2     90        174
Paciente 3     80        163
Paciente 4     45        150
Paciente 5     59        147
```

Extraer elementos de una planilla de datos

Extraer una variable completa
Por ejemplo, si quisiéramos extraer del objeto `DF` la variable `peso (kg)` utilizamos:

```
DF$'peso (kg)' #Entrega el vector de pesos (kg)
```

También, note que la variable `peso (kg)` corresponde a la primera columna de la planilla `DF`, por lo que, es posible también extraer dicha columna de la siguiente forma:

```
DF[,1] #Entrega el vector de pesos (kg)
```

Análogamente se puede extraer la información de la variable `altura (cms)` de las siguientes dos formas:

```
DF$`altura (cms)` # Entrega el vector de alturas (cms)
DF[,2] #Entrega el vector de alturas (cms)
```

Extraer elementos de una planilla de datos

Extraer una fila completa
Por ejemplo, suponga que al nutricionista a cargo, le interesa conocer la información del Paciente 4, que corresponde a la cuarta fila del objeto `DF`. Esto es posible utilizando la siguiente línea de código:

```
DF[4,]
```

Extraer una posición en particular
Si qusiéramos extraer solo la altura del tercer paciente del objeto `DF` se utiliza lo siguiente:

```
DF[3,2]
```

Extraer elementos de una planilla

Extraer varias filas o columnas

Para extraer varias filas o columnas es análogo al procedimiento anterior, pero se debe especificar un vector. Por ejemplo, suponga que interesa conocer el peso y altura de los pacientes 1, 3 y 5, esto se puede realizar de la siguiente forma:

```
DF[seq(1,5,by=2),1:2]
```

```
          peso (kg) altura (cms)
Paciente 1        55          163
Paciente 3        80          163
Paciente 5        59          147
```

Ejercicio

La fórmula del IMC es peso (kg) dividido en la altura (mt) al cuadrado. Calcule el IMC para todos los pacientes.

Solución

```
DF$‘peso (kg)‘/(DF$‘altura (cms)‘/100)**2
[1] 20.70082 30.07117 30.11028 20.00000 27.30344
```

Otra manera equivalente:

```
DF[,1]/(DF[,2]/100)**2
[1] 20.70082 30.07117 30.11028 20.00000 27.30344
```

Añadiendo el IMC al objeto DF

Es posible añadir la información recién calculada a la tabla DF en una nueva columna:

```
DF[,3]<-DF$‘peso (kg)‘/(DF$‘altura (cms)‘/100)**2
```

```
DF
           peso (kg)  altura (cms)   V3
 Paciente 1    55          163       20.70082
 Paciente 2    90          173       30.07117
 Paciente 3    80          163       30.11028
 Paciente 4    45          150       20.00000
 Paciente 5    59          147       27.30344
```

Y si queremos añadir el nombre de IMC a la tercera columna, utilizamos:
`colnames(DF)[3]<-"IMC"`

Resultado

```
DF
           peso (kg)  altura (cms)   IMC
 Paciente 1    55          163       20.70082
 Paciente 2    90          173       30.07117
 Paciente 3    80          163       30.11028
 Paciente 4    45          150       20.00000
 Paciente 5    59          147       27.30344
```

Una tabla de datos bien lograda siempre es autoexplicativa por sí misma.

Detalles importantes:

- Especificar el nombre de cada una de las variables
- Especificar el formato de las variables con cierta ambiguedad

Práctica 1

Practica 2

- ¿Qué sintaxis hubiera utilizado si solo hubiera querido calcular el IMC para aquellos pacientes que presentaban un peso mayor a 60 kilogramos?

Respuesta práctica 2

```
#Opción 1
DF[which(DF$‘peso (kg)‘>60),1]/(DF[which(DF$‘peso (kg)‘>60),2]/100)**2
[1] 30.07117 30.11028

#Opción 2
DF$‘peso (kg)‘[which(DF$‘peso (kg)‘>60)]/(DF$‘altura (cms)‘[which(DF$‘peso (kg)‘>60)]/100)**2
[1] 30.07117 30.11028
```

Ambas son equivalentes.