MÓDULO 1: HERRAMIENTAS BIG DATA

HERRAMIENTAS DE ANALISIS: PROGRAMACIÓN EN R - MICROACTIVIDADES

Ericsson Marc

20/12/2021

# EJERCICIO 1

#vamos a jugar

Para el ejercicio 1, utilizaremos los datos los datos millas que hay el package datos. Estos datos consisten en 238 filas y 11 columnas que describen el consumo de combustible de 38 modelos de coche populares.

Puedes consultar más sobre los datos en la ayuda: ?millas.

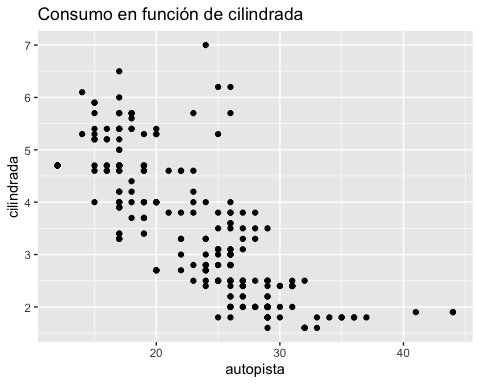
library(datos)  
suppressPackageStartupMessages(library(tidyverse))

?millas

## EJERCICIO 1.1.

A partir de los datos de **millas**, dibuja un gráfico de dispersión de puntos que muestre las millas recorridas en autopista por galón de combustible consumido (**autopista**) respecto a la **cilindrada** del motor de cada automóvil. No olvides añadir títulos al gráfico y a los ejes x e y.

# Solución:  
  
ejercicio\_1 <- ggplot(data = millas, mapping = aes(x =autopista, y = cilindrada)) +   
 geom\_point() +  
 labs(title = "Consumo en función de cilindrada"  
 , x = "autopista"  
 , y = "cilindrada")  
   
ejercicio\_1



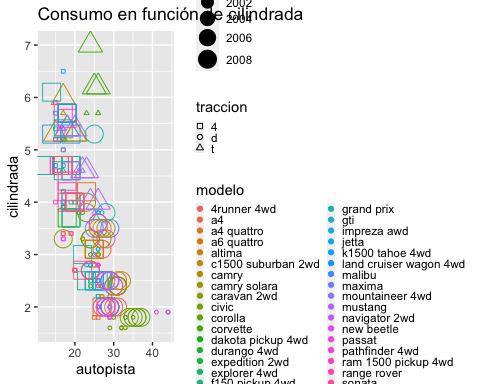
## EJERCICIO 1.2.

A partir del gráfico del ejercicio 1.1., escoge una columna para cada uno de los siguientes parámetros estéticos: color, size y shape.

Truco: Observa que puedes seleccionar tanto columnas numéricas como de tipo carácter o factor. Si lo crees interesante, puedes utilizar la misma columna para distintos parámetros del gráfico .

Comenta algún aspecto relevante que hayas descubierto sobre los coches a partir del gráfico.

# Solución:  
  
ejercicio\_2 <- ggplot(data = millas, mapping = aes(x =autopista, y = cilindrada, color = modelo, shape = traccion, size = anio)) +   
 geom\_point() +  
 scale\_shape\_manual(values=seq(0,15)) +  
 labs(title = "Consumo en función de cilindrada"  
 , x = "autopista"  
 , y = "cilindrada") +  
 theme(legend.key.size = unit(0.2,"cm"))  
  
ejercicio\_2



#Claramente podemos observar que cuanto más cilindrada, menos consumo.

## EJERCICIO 1.3.

Transforma el siguiente vector de tipo factor a tipo numeric de forma que el valor final mostrado sea exactamente el mismo en ambos vectores, pero con formato distinto. Para ello utiliza as.character() y as.numeric().

¿Qué sucede si sólo utilizas as.numeric() directamente sobre la columna factor?

vec <- factor(c("8","5","9","8","1","7"))  
print(vec) # valor mostrado

## [1] 8 5 9 8 1 7  
## Levels: 1 5 7 8 9

# Solución:  
  
vec\_numeric <- as.numeric(as.character(vec))  
print(vec\_numeric)

## [1] 8 5 9 8 1 7

#si solo usamos as.numeric() entonces le asignamos numericamente a cada factor distinto un orden (categorizamos)  
vec\_numeric\_2 <- as.numeric(vec)  
print(vec\_numeric\_2)

## [1] 4 2 5 4 1 3

## EJERCICIO 1.4.

Es millas un objeto de la clase *data.frame* o *matrix*?

¿Y el siguiente objeto obj?

obj1 <- cbind(millas$cilindrada,millas$cilindros)  
  
# solución  
  
head(millas)

## # A tibble: 6 × 11  
## fabricante modelo cilindrada anio cilindros transmision traccion ciudad  
## <chr> <chr> <dbl> <int> <int> <chr> <chr> <int>  
## 1 audi a4 1.8 1999 4 auto(l5) d 18  
## 2 audi a4 1.8 1999 4 manual(m5) d 21  
## 3 audi a4 2 2008 4 manual(m6) d 20  
## 4 audi a4 2 2008 4 auto(av) d 21  
## 5 audi a4 2.8 1999 6 auto(l5) d 16  
## 6 audi a4 2.8 1999 6 manual(m5) d 18  
## # … with 3 more variables: autopista <int>, combustible <chr>, clase <chr>

class(millas)

## [1] "tbl\_df" "tbl" "data.frame"

head(obj1)

## [,1] [,2]  
## [1,] 1.8 4  
## [2,] 1.8 4  
## [3,] 2.0 4  
## [4,] 2.0 4  
## [5,] 2.8 6  
## [6,] 2.8 6

class(obj1)

## [1] "matrix" "array"

## EJERCICIO 1.5.

Crea una función que tome un vector de tipo integer como input y retorne un objetido de tipo lista que contega los siguientes elementos:

1. El último valor del vector
2. Los elementos de las posiciones impares.
3. Todos los elementos excepto el primero.
4. Solo números impares (y no valores faltantes).

# solución  
  
  
ejercicio\_5 <- function(x) {  
   
 resultados <- list()  
   
 resultados[["Last value"]] <- tail(x, n = 1)  
   
 resultados[["Evens"]] <- x[c(TRUE,FALSE)]  
   
 resultados[["All - first value"]] <- x[-1]  
   
 resultados[["Only evens"]] <- x[x%%2 != 0]  
   
 print(resultados)  
}  
  
vec1 <- c(3,2,4,5,10,9)  
  
ejercicio\_5(vec1)

## $`Last value`  
## [1] 9  
##   
## $Evens  
## [1] 3 4 10  
##   
## $`All - first value`  
## [1] 2 4 5 10 9  
##   
## $`Only evens`  
## [1] 3 5 9

## EJERCICIO 1.6.

Busca un ejemplo de objeto x en el que la expresión x[-which(x > 0)] no devuelve el mismo resultado que x[x <= 0]

# Solución:  
  
x <- c(NaN, 2)  
  
x[-which(x > 0)]

## [1] NaN

x[x <= 0]

## [1] NA

## EJERCICIO 1.7.

Añade a millas una nueva columna llamada “fabr\_mod” que contenga la concatenación del nombre del fabricante, un guion “-” y el modelo del coche. Presenta la nueva columna mediante la función head().

# Solución:  
  
millas$fabr\_mod <- 1:nrow(millas)  
  
  
millas$fabr\_mod <- paste(millas$fabricante,"-",millas$modelo)  
head(millas)

## # A tibble: 6 × 12  
## fabricante modelo cilindrada anio cilindros transmision traccion ciudad  
## <chr> <chr> <dbl> <int> <int> <chr> <chr> <int>  
## 1 audi a4 1.8 1999 4 auto(l5) d 18  
## 2 audi a4 1.8 1999 4 manual(m5) d 21  
## 3 audi a4 2 2008 4 manual(m6) d 20  
## 4 audi a4 2 2008 4 auto(av) d 21  
## 5 audi a4 2.8 1999 6 auto(l5) d 16  
## 6 audi a4 2.8 1999 6 manual(m5) d 18  
## # … with 4 more variables: autopista <int>, combustible <chr>, clase <chr>,  
## # fabr\_mod <chr>

## EJERCICIO 1.8.

Selecciona todos los coches de millas que cumplan con todas todas las condiciones siguientes:

* La marca es distinta a “dodge”
* Tiene tracción en las cuatro puertas
* Han estado fabricados antes del 2008
* Las millas/galón, o bién en ciudad, o bién en carretera, no llegan a 12 millas/galón.

¿Cuantos coches has encontrado?

# Solución:  
  
ejercicio\_8 <- subset(millas,fabricante != "dodge" & traccion == "4" & anio < 2008 & (ciudad <= 12 | autopista <= 12))  
ejercicio\_8

## # A tibble: 5 × 12  
## fabricante modelo cilindrada anio cilindros transmision traccion ciudad  
## <chr> <chr> <dbl> <int> <int> <chr> <chr> <int>  
## 1 chevrolet k1500 tahoe… 5.7 1999 8 auto(l4) 4 11  
## 2 ford f150 pickup… 5.4 1999 8 auto(l4) 4 11  
## 3 land rover range rover 4 1999 8 auto(l4) 4 11  
## 4 land rover range rover 4.6 1999 8 auto(l4) 4 11  
## 5 toyota land cruise… 4.7 1999 8 auto(l4) 4 11  
## # … with 4 more variables: autopista <int>, combustible <chr>, clase <chr>,  
## # fabr\_mod <chr>

## EJERCICIO 1.9.

Añade una nueva columna “vol\_por\_cil” a obj del ejercicio 1.4. que contenga el ratio de la cilindrada sobre el número de cilindros. Presenta el summary de la nueva columna.

# Solución:  
  
obj <- as.data.frame(cbind(millas$cilindrada,millas$cilindros, NA))  
colnames(obj) <- c("cilindrada","cilindros","vol\_por\_cil")  
obj$vol\_por\_cil <- as.numeric(obj$cilindrada) / as.numeric(obj$cilindros)  
summary(obj$vol\_por\_cil)

## Min. 1st Qu. Median Mean 3rd Qu. Max.   
## 0.4000 0.5000 0.5875 0.5780 0.6500 0.8750

## EJERCICIO 1.10.

Modifica los valores de la columna “vol\_por\_cil” del objeto obj del ejercicio 1.9. asignando NA a los valores de esta columna que sean superiores a 0.7.

Presenta los datos con un summary del nuevo objeto obj. ¿Cuántos valores NA se han creado en esta columna?

# Solución:  
  
ejercicio\_10 <- obj  
  
ejercicio\_10$vol\_por\_cil <- ifelse ( ejercicio\_10$vol\_por\_cil >= 0.7, NA, ejercicio\_10$vol\_por\_cil)  
  
ejercicio\_10\_1 <- subset(ejercicio\_10, is.na(ejercicio\_10$vol\_por\_cil))  
print(nrow(ejercicio\_10\_1))

## [1] 19

summary(ejercicio\_10)

## cilindrada cilindros vol\_por\_cil   
## Min. :1.600 Min. :4.000 Min. :0.4000   
## 1st Qu.:2.400 1st Qu.:4.000 1st Qu.:0.5000   
## Median :3.300 Median :6.000 Median :0.5750   
## Mean :3.472 Mean :5.889 Mean :0.5638   
## 3rd Qu.:4.600 3rd Qu.:8.000 3rd Qu.:0.6250   
## Max. :7.000 Max. :8.000 Max. :0.6750   
## NA's :19

summary(ejercicio\_10\_1)

## cilindrada cilindros vol\_por\_cil   
## Min. :4.200 Min. :6.000 Min. : NA   
## 1st Qu.:5.700 1st Qu.:8.000 1st Qu.: NA   
## Median :5.700 Median :8.000 Median : NA   
## Mean :5.758 Mean :7.789 Mean :NaN   
## 3rd Qu.:6.050 3rd Qu.:8.000 3rd Qu.: NA   
## Max. :7.000 Max. :8.000 Max. : NA   
## NA's :19