MÓDULO 1: HERRAMIENTAS BIG DATA

TEMA 4.1. HERRAMIENTAS DE ANALISIS: PROGRAMACIÓN EN R - MICROACTIVIDADES

Ferran Carrascosa Mallafrè

Licenciado en Matemáticas por la Universidad de Barcelona. Data Scientist

# EJERCICIO 1

Para el ejercicio 1, utilizaremos los datos los datos millas que hay el el package datos. Estos datos consisten en 238 y 11 variables 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 1.2.

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

Truco: Observa que puedes seleccionar tanto variables continuas como categóricas. Si lo crees interesante, puedes repetir.

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

# Solución:

## 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().

¿Que sucede si sólo utilizas as.numeric()?

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:

## EJERCICIO 1.4.

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

¿Y el siguiente objeto obj?

# solución

## EJERCICIO 1.5.

Crea una función que tome un vector como input y retorne un lista con los siguientes resultados:

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 1.6.

¿Por qué x[-which(x > 0)] no es lo mismo que x[x <= 0]?

Pon un ejemplo de vector x donde el resultado de ambas expresiones sea distinto.

# Solución:

## 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.

# Solución:

## 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 en ciudad o en carretera no llega a 12 millas/galón.

Cuantos coches has encontrado?

# Solución:

## EJERCICIO 1.9.

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

# Solución:

## 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 la columna superiores a 0.7.

Presenta los datos con un summary del nuevo objeto obj.

# Solución:

# EJERCICIO 2

## EJERCICIO 2.1.

Escribe un bucle for que calcule le media todas las columnas numéricas de millas.

# Solución:

## EJERCICIO 2.2.

Haz lo mismo que en 2.1 pero utilizando sapply() en vez del bucle for.

# Solución:

## EJERCICIO 2.3.

Explica la diferencia entre la función if() e ifelse(). Pon un ejemplo de uso de ambas.

# Solución:

## EJERCICIO 2.4.

¿Qué parámetros es imprescindible especificar cuando se leen datos de ancho fijo mediante: read.fwf()?

Explica qué significan y pon un ejemplo.

# Solución:

## EJERCICIO 2.5.

Calcula la media de millas/galón en autopista para cada clase de coche de millas.

# Solución:

## EJERCICIO 2.6.

Incorpora la media de calculada en 2.5. en el data frame millas como una nueva variable llamada “autopista\_clase”.

Utiliza la funcion merge() para juntar la table de 2.5 con millas.

# Solución:

## EJERCICIO 2.7.

Utiliza las funciones del package dplyr: group\_by() y mutate() para realizar el mismo calculo que en 2.5. y 2.6. sin necesidad de utilizar merge(). Llama a la nueva columna “autopista\_clase\_dplyr”

Truco: Utiliza el siguiente ejemplo: datos %>% group\_by(var\_seg) %>% mutate(nueva\_variable=mean(variable))

Haz un summary() para verificar que el resultado es el mismo que en 2.6.

# Solución:

## EJERCICIO 2.8.

Analiza si millas tiene registros duplicado y en caso afirmativo crea un nuevo data frame que contenga una única copia de cada fila.

# Solución:

## EJERCICIO 2.9.

Crea una función que tenga como input la fecha de tu nacimiento (en formato date) y devuelva tu edad en años.

# Solución:

## EJERCICIO 2.10.

Explica porqué el resultado de fechahora\_1 y fechahora\_2 son distintos en la siguiente expresión:

library(lubridate)  
Sys.setlocale(locale="es\_ES.UTF-8")

## [1] "LC\_CTYPE=es\_ES.UTF-8;LC\_NUMERIC=C;LC\_TIME=es\_ES.UTF-8;LC\_COLLATE=es\_ES.UTF-8;LC\_MONETARY=es\_ES.UTF-8;LC\_MESSAGES=en\_US.UTF-8;LC\_PAPER=en\_US.UTF-8;LC\_NAME=C;LC\_ADDRESS=C;LC\_TELEPHONE=C;LC\_MEASUREMENT=en\_US.UTF-8;LC\_IDENTIFICATION=C"

fechahora <- ymd\_hms("2020-03-28 15:11:23", tz = "Europe/Madrid")  
fechahora\_1 <- fechahora + dhours(24)  
fechahora\_2 <- fechahora + hours(24)  
  
print(fechahora\_1)

## [1] "2020-03-29 16:11:23 CEST"

print(fechahora\_2)

## [1] "2020-03-29 15:11:23 CEST"