MÓDULO 1: HERRAMIENTAS BIG DATA

HERRAMIENTAS DE ANALISIS: PROGRAMACIÓN EN R - MICROACTIVIDADES

Nombre Alumno

Fecha

# 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))

## Warning: package 'tidyverse' was built under R version 3.6.2

## Warning: package 'tibble' was built under R version 3.6.2

## Warning: package 'tidyr' was built under R version 3.6.2

## Warning: package 'readr' was built under R version 3.6.2

## Warning: package 'purrr' was built under R version 3.6.2

## Warning: package 'dplyr' was built under R version 3.6.2

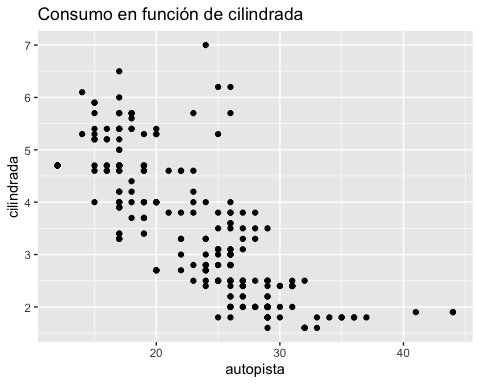
## Warning: package 'forcats' was built under R version 3.6.2

?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



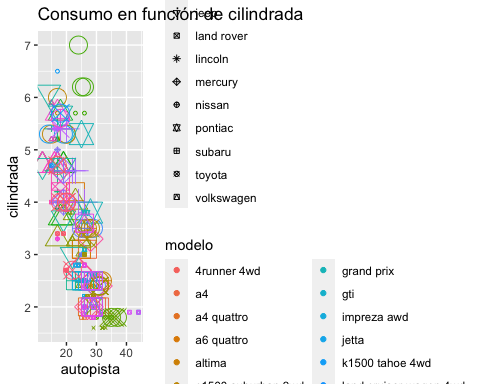
## EJERCICIO 1.2.

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 = fabricante, size = anio)) +   
 geom\_point() +  
 scale\_shape\_manual(values=seq(0,15)) +  
 labs(title = "Consumo en función de cilindrada"  
 , x = "autopista"  
 , y = "cilindrada")  
  
ejercicio\_2



## 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  
  
print(millas)

## # A tibble: 234 × 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  
## 7 audi a4 3.1 2008 6 auto(av) d 18  
## 8 audi a4 qu… 1.8 1999 4 manual(m5) 4 18  
## 9 audi a4 qu… 1.8 1999 4 auto(l5) 4 16  
## 10 audi a4 qu… 2 2008 4 manual(m6) 4 20  
## # … with 224 more rows, and 3 more variables: autopista <int>,  
## # combustible <chr>, clase <chr>

class(millas)

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

print(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  
## [7,] 3.1 6  
## [8,] 1.8 4  
## [9,] 1.8 4  
## [10,] 2.0 4  
## [11,] 2.0 4  
## [12,] 2.8 6  
## [13,] 2.8 6  
## [14,] 3.1 6  
## [15,] 3.1 6  
## [16,] 2.8 6  
## [17,] 3.1 6  
## [18,] 4.2 8  
## [19,] 5.3 8  
## [20,] 5.3 8  
## [21,] 5.3 8  
## [22,] 5.7 8  
## [23,] 6.0 8  
## [24,] 5.7 8  
## [25,] 5.7 8  
## [26,] 6.2 8  
## [27,] 6.2 8  
## [28,] 7.0 8  
## [29,] 5.3 8  
## [30,] 5.3 8  
## [31,] 5.7 8  
## [32,] 6.5 8  
## [33,] 2.4 4  
## [34,] 2.4 4  
## [35,] 3.1 6  
## [36,] 3.5 6  
## [37,] 3.6 6  
## [38,] 2.4 4  
## [39,] 3.0 6  
## [40,] 3.3 6  
## [41,] 3.3 6  
## [42,] 3.3 6  
## [43,] 3.3 6  
## [44,] 3.3 6  
## [45,] 3.8 6  
## [46,] 3.8 6  
## [47,] 3.8 6  
## [48,] 4.0 6  
## [49,] 3.7 6  
## [50,] 3.7 6  
## [51,] 3.9 6  
## [52,] 3.9 6  
## [53,] 4.7 8  
## [54,] 4.7 8  
## [55,] 4.7 8  
## [56,] 5.2 8  
## [57,] 5.2 8  
## [58,] 3.9 6  
## [59,] 4.7 8  
## [60,] 4.7 8  
## [61,] 4.7 8  
## [62,] 5.2 8  
## [63,] 5.7 8  
## [64,] 5.9 8  
## [65,] 4.7 8  
## [66,] 4.7 8  
## [67,] 4.7 8  
## [68,] 4.7 8  
## [69,] 4.7 8  
## [70,] 4.7 8  
## [71,] 5.2 8  
## [72,] 5.2 8  
## [73,] 5.7 8  
## [74,] 5.9 8  
## [75,] 4.6 8  
## [76,] 5.4 8  
## [77,] 5.4 8  
## [78,] 4.0 6  
## [79,] 4.0 6  
## [80,] 4.0 6  
## [81,] 4.0 6  
## [82,] 4.6 8  
## [83,] 5.0 8  
## [84,] 4.2 6  
## [85,] 4.2 6  
## [86,] 4.6 8  
## [87,] 4.6 8  
## [88,] 4.6 8  
## [89,] 5.4 8  
## [90,] 5.4 8  
## [91,] 3.8 6  
## [92,] 3.8 6  
## [93,] 4.0 6  
## [94,] 4.0 6  
## [95,] 4.6 8  
## [96,] 4.6 8  
## [97,] 4.6 8  
## [98,] 4.6 8  
## [99,] 5.4 8  
## [100,] 1.6 4  
## [101,] 1.6 4  
## [102,] 1.6 4  
## [103,] 1.6 4  
## [104,] 1.6 4  
## [105,] 1.8 4  
## [106,] 1.8 4  
## [107,] 1.8 4  
## [108,] 2.0 4  
## [109,] 2.4 4  
## [110,] 2.4 4  
## [111,] 2.4 4  
## [112,] 2.4 4  
## [113,] 2.5 6  
## [114,] 2.5 6  
## [115,] 3.3 6  
## [116,] 2.0 4  
## [117,] 2.0 4  
## [118,] 2.0 4  
## [119,] 2.0 4  
## [120,] 2.7 6  
## [121,] 2.7 6  
## [122,] 2.7 6  
## [123,] 3.0 6  
## [124,] 3.7 6  
## [125,] 4.0 6  
## [126,] 4.7 8  
## [127,] 4.7 8  
## [128,] 4.7 8  
## [129,] 5.7 8  
## [130,] 6.1 8  
## [131,] 4.0 8  
## [132,] 4.2 8  
## [133,] 4.4 8  
## [134,] 4.6 8  
## [135,] 5.4 8  
## [136,] 5.4 8  
## [137,] 5.4 8  
## [138,] 4.0 6  
## [139,] 4.0 6  
## [140,] 4.6 8  
## [141,] 5.0 8  
## [142,] 2.4 4  
## [143,] 2.4 4  
## [144,] 2.5 4  
## [145,] 2.5 4  
## [146,] 3.5 6  
## [147,] 3.5 6  
## [148,] 3.0 6  
## [149,] 3.0 6  
## [150,] 3.5 6  
## [151,] 3.3 6  
## [152,] 3.3 6  
## [153,] 4.0 6  
## [154,] 5.6 8  
## [155,] 3.1 6  
## [156,] 3.8 6  
## [157,] 3.8 6  
## [158,] 3.8 6  
## [159,] 5.3 8  
## [160,] 2.5 4  
## [161,] 2.5 4  
## [162,] 2.5 4  
## [163,] 2.5 4  
## [164,] 2.5 4  
## [165,] 2.5 4  
## [166,] 2.2 4  
## [167,] 2.2 4  
## [168,] 2.5 4  
## [169,] 2.5 4  
## [170,] 2.5 4  
## [171,] 2.5 4  
## [172,] 2.5 4  
## [173,] 2.5 4  
## [174,] 2.7 4  
## [175,] 2.7 4  
## [176,] 3.4 6  
## [177,] 3.4 6  
## [178,] 4.0 6  
## [179,] 4.7 8  
## [180,] 2.2 4  
## [181,] 2.2 4  
## [182,] 2.4 4  
## [183,] 2.4 4  
## [184,] 3.0 6  
## [185,] 3.0 6  
## [186,] 3.5 6  
## [187,] 2.2 4  
## [188,] 2.2 4  
## [189,] 2.4 4  
## [190,] 2.4 4  
## [191,] 3.0 6  
## [192,] 3.0 6  
## [193,] 3.3 6  
## [194,] 1.8 4  
## [195,] 1.8 4  
## [196,] 1.8 4  
## [197,] 1.8 4  
## [198,] 1.8 4  
## [199,] 4.7 8  
## [200,] 5.7 8  
## [201,] 2.7 4  
## [202,] 2.7 4  
## [203,] 2.7 4  
## [204,] 3.4 6  
## [205,] 3.4 6  
## [206,] 4.0 6  
## [207,] 4.0 6  
## [208,] 2.0 4  
## [209,] 2.0 4  
## [210,] 2.0 4  
## [211,] 2.0 4  
## [212,] 2.8 6  
## [213,] 1.9 4  
## [214,] 2.0 4  
## [215,] 2.0 4  
## [216,] 2.0 4  
## [217,] 2.0 4  
## [218,] 2.5 5  
## [219,] 2.5 5  
## [220,] 2.8 6  
## [221,] 2.8 6  
## [222,] 1.9 4  
## [223,] 1.9 4  
## [224,] 2.0 4  
## [225,] 2.0 4  
## [226,] 2.5 5  
## [227,] 2.5 5  
## [228,] 1.8 4  
## [229,] 1.8 4  
## [230,] 2.0 4  
## [231,] 2.0 4  
## [232,] 2.8 6  
## [233,] 2.8 6  
## [234,] 3.6 6

class(obj1)

## [1] "matrix"

## 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(input) {  
 if (input == 1) {  
 function\_return <- vec[length(vec)]  
 }  
 if (input == 2){  
 function\_return <- vec[c(1,3,5)]  
 }  
 if (input == 3){  
 function\_return <- vec[2:length(vec)]  
 }  
   
 if (input == 4){  
 numeric\_vector <- as.numeric(as.character(vec))  
 function\_return <- subset(numeric\_vector, numeric\_vector %% 2 != 0)  
 }  
 return(function\_return)  
   
  
}  
  
ejercicio\_5(1)

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

ejercicio\_5(2)

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

ejercicio\_5(3)

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

ejercicio\_5(4)

## [1] 5 9 1 7

## 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 … 5.7 1999 8 auto(l4) 4 11  
## 2 ford f150 p… 5.4 1999 8 auto(l4) 4 11  
## 3 land rover range … 4 1999 8 auto(l4) 4 11  
## 4 land rover range … 4.6 1999 8 auto(l4) 4 11  
## 5 toyota land c… 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)  
print(ejercicio\_10)

## cilindrada cilindros vol\_por\_cil  
## 1 1.8 4 0.4500000  
## 2 1.8 4 0.4500000  
## 3 2.0 4 0.5000000  
## 4 2.0 4 0.5000000  
## 5 2.8 6 0.4666667  
## 6 2.8 6 0.4666667  
## 7 3.1 6 0.5166667  
## 8 1.8 4 0.4500000  
## 9 1.8 4 0.4500000  
## 10 2.0 4 0.5000000  
## 11 2.0 4 0.5000000  
## 12 2.8 6 0.4666667  
## 13 2.8 6 0.4666667  
## 14 3.1 6 0.5166667  
## 15 3.1 6 0.5166667  
## 16 2.8 6 0.4666667  
## 17 3.1 6 0.5166667  
## 18 4.2 8 0.5250000  
## 19 5.3 8 0.6625000  
## 20 5.3 8 0.6625000  
## 21 5.3 8 0.6625000  
## 22 5.7 8 NA  
## 23 6.0 8 NA  
## 24 5.7 8 NA  
## 25 5.7 8 NA  
## 26 6.2 8 NA  
## 27 6.2 8 NA  
## 28 7.0 8 NA  
## 29 5.3 8 0.6625000  
## 30 5.3 8 0.6625000  
## 31 5.7 8 NA  
## 32 6.5 8 NA  
## 33 2.4 4 0.6000000  
## 34 2.4 4 0.6000000  
## 35 3.1 6 0.5166667  
## 36 3.5 6 0.5833333  
## 37 3.6 6 0.6000000  
## 38 2.4 4 0.6000000  
## 39 3.0 6 0.5000000  
## 40 3.3 6 0.5500000  
## 41 3.3 6 0.5500000  
## 42 3.3 6 0.5500000  
## 43 3.3 6 0.5500000  
## 44 3.3 6 0.5500000  
## 45 3.8 6 0.6333333  
## 46 3.8 6 0.6333333  
## 47 3.8 6 0.6333333  
## 48 4.0 6 0.6666667  
## 49 3.7 6 0.6166667  
## 50 3.7 6 0.6166667  
## 51 3.9 6 0.6500000  
## 52 3.9 6 0.6500000  
## 53 4.7 8 0.5875000  
## 54 4.7 8 0.5875000  
## 55 4.7 8 0.5875000  
## 56 5.2 8 0.6500000  
## 57 5.2 8 0.6500000  
## 58 3.9 6 0.6500000  
## 59 4.7 8 0.5875000  
## 60 4.7 8 0.5875000  
## 61 4.7 8 0.5875000  
## 62 5.2 8 0.6500000  
## 63 5.7 8 NA  
## 64 5.9 8 NA  
## 65 4.7 8 0.5875000  
## 66 4.7 8 0.5875000  
## 67 4.7 8 0.5875000  
## 68 4.7 8 0.5875000  
## 69 4.7 8 0.5875000  
## 70 4.7 8 0.5875000  
## 71 5.2 8 0.6500000  
## 72 5.2 8 0.6500000  
## 73 5.7 8 NA  
## 74 5.9 8 NA  
## 75 4.6 8 0.5750000  
## 76 5.4 8 0.6750000  
## 77 5.4 8 0.6750000  
## 78 4.0 6 0.6666667  
## 79 4.0 6 0.6666667  
## 80 4.0 6 0.6666667  
## 81 4.0 6 0.6666667  
## 82 4.6 8 0.5750000  
## 83 5.0 8 0.6250000  
## 84 4.2 6 NA  
## 85 4.2 6 NA  
## 86 4.6 8 0.5750000  
## 87 4.6 8 0.5750000  
## 88 4.6 8 0.5750000  
## 89 5.4 8 0.6750000  
## 90 5.4 8 0.6750000  
## 91 3.8 6 0.6333333  
## 92 3.8 6 0.6333333  
## 93 4.0 6 0.6666667  
## 94 4.0 6 0.6666667  
## 95 4.6 8 0.5750000  
## 96 4.6 8 0.5750000  
## 97 4.6 8 0.5750000  
## 98 4.6 8 0.5750000  
## 99 5.4 8 0.6750000  
## 100 1.6 4 0.4000000  
## 101 1.6 4 0.4000000  
## 102 1.6 4 0.4000000  
## 103 1.6 4 0.4000000  
## 104 1.6 4 0.4000000  
## 105 1.8 4 0.4500000  
## 106 1.8 4 0.4500000  
## 107 1.8 4 0.4500000  
## 108 2.0 4 0.5000000  
## 109 2.4 4 0.6000000  
## 110 2.4 4 0.6000000  
## 111 2.4 4 0.6000000  
## 112 2.4 4 0.6000000  
## 113 2.5 6 0.4166667  
## 114 2.5 6 0.4166667  
## 115 3.3 6 0.5500000  
## 116 2.0 4 0.5000000  
## 117 2.0 4 0.5000000  
## 118 2.0 4 0.5000000  
## 119 2.0 4 0.5000000  
## 120 2.7 6 0.4500000  
## 121 2.7 6 0.4500000  
## 122 2.7 6 0.4500000  
## 123 3.0 6 0.5000000  
## 124 3.7 6 0.6166667  
## 125 4.0 6 0.6666667  
## 126 4.7 8 0.5875000  
## 127 4.7 8 0.5875000  
## 128 4.7 8 0.5875000  
## 129 5.7 8 NA  
## 130 6.1 8 NA  
## 131 4.0 8 0.5000000  
## 132 4.2 8 0.5250000  
## 133 4.4 8 0.5500000  
## 134 4.6 8 0.5750000  
## 135 5.4 8 0.6750000  
## 136 5.4 8 0.6750000  
## 137 5.4 8 0.6750000  
## 138 4.0 6 0.6666667  
## 139 4.0 6 0.6666667  
## 140 4.6 8 0.5750000  
## 141 5.0 8 0.6250000  
## 142 2.4 4 0.6000000  
## 143 2.4 4 0.6000000  
## 144 2.5 4 0.6250000  
## 145 2.5 4 0.6250000  
## 146 3.5 6 0.5833333  
## 147 3.5 6 0.5833333  
## 148 3.0 6 0.5000000  
## 149 3.0 6 0.5000000  
## 150 3.5 6 0.5833333  
## 151 3.3 6 0.5500000  
## 152 3.3 6 0.5500000  
## 153 4.0 6 0.6666667  
## 154 5.6 8 NA  
## 155 3.1 6 0.5166667  
## 156 3.8 6 0.6333333  
## 157 3.8 6 0.6333333  
## 158 3.8 6 0.6333333  
## 159 5.3 8 0.6625000  
## 160 2.5 4 0.6250000  
## 161 2.5 4 0.6250000  
## 162 2.5 4 0.6250000  
## 163 2.5 4 0.6250000  
## 164 2.5 4 0.6250000  
## 165 2.5 4 0.6250000  
## 166 2.2 4 0.5500000  
## 167 2.2 4 0.5500000  
## 168 2.5 4 0.6250000  
## 169 2.5 4 0.6250000  
## 170 2.5 4 0.6250000  
## 171 2.5 4 0.6250000  
## 172 2.5 4 0.6250000  
## 173 2.5 4 0.6250000  
## 174 2.7 4 0.6750000  
## 175 2.7 4 0.6750000  
## 176 3.4 6 0.5666667  
## 177 3.4 6 0.5666667  
## 178 4.0 6 0.6666667  
## 179 4.7 8 0.5875000  
## 180 2.2 4 0.5500000  
## 181 2.2 4 0.5500000  
## 182 2.4 4 0.6000000  
## 183 2.4 4 0.6000000  
## 184 3.0 6 0.5000000  
## 185 3.0 6 0.5000000  
## 186 3.5 6 0.5833333  
## 187 2.2 4 0.5500000  
## 188 2.2 4 0.5500000  
## 189 2.4 4 0.6000000  
## 190 2.4 4 0.6000000  
## 191 3.0 6 0.5000000  
## 192 3.0 6 0.5000000  
## 193 3.3 6 0.5500000  
## 194 1.8 4 0.4500000  
## 195 1.8 4 0.4500000  
## 196 1.8 4 0.4500000  
## 197 1.8 4 0.4500000  
## 198 1.8 4 0.4500000  
## 199 4.7 8 0.5875000  
## 200 5.7 8 NA  
## 201 2.7 4 0.6750000  
## 202 2.7 4 0.6750000  
## 203 2.7 4 0.6750000  
## 204 3.4 6 0.5666667  
## 205 3.4 6 0.5666667  
## 206 4.0 6 0.6666667  
## 207 4.0 6 0.6666667  
## 208 2.0 4 0.5000000  
## 209 2.0 4 0.5000000  
## 210 2.0 4 0.5000000  
## 211 2.0 4 0.5000000  
## 212 2.8 6 0.4666667  
## 213 1.9 4 0.4750000  
## 214 2.0 4 0.5000000  
## 215 2.0 4 0.5000000  
## 216 2.0 4 0.5000000  
## 217 2.0 4 0.5000000  
## 218 2.5 5 0.5000000  
## 219 2.5 5 0.5000000  
## 220 2.8 6 0.4666667  
## 221 2.8 6 0.4666667  
## 222 1.9 4 0.4750000  
## 223 1.9 4 0.4750000  
## 224 2.0 4 0.5000000  
## 225 2.0 4 0.5000000  
## 226 2.5 5 0.5000000  
## 227 2.5 5 0.5000000  
## 228 1.8 4 0.4500000  
## 229 1.8 4 0.4500000  
## 230 2.0 4 0.5000000  
## 231 2.0 4 0.5000000  
## 232 2.8 6 0.4666667  
## 233 2.8 6 0.4666667  
## 234 3.6 6 0.6000000

ejercio\_10\_1 <- subset(ejercicio\_10, is.na(ejercicio\_10$vol\_por\_cil))  
print(nrow(ejercio\_10\_1))

## [1] 19

summary(ejercicio\_10$vol\_por\_cil)

## Min. 1st Qu. Median Mean 3rd Qu. Max. NA's   
## 0.4000 0.5000 0.5750 0.5638 0.6250 0.6750 19