Práctica 0. FMAD 2021-2022

ICAI. Master en Big Data. Fundamentos Matemáticos del Análisis de Datos (FMAD).

Gutiérrez, Laura

Curso 2021-22. Última actualización: 2021-09-10

Ejercicio 0 (ejemplo).

Enunciado: Usa la función `seq` de R para fabricar un vector `v` con los múltiplos de 3 del 0 al 300. Muestra los primeros 20 elementos de `v` usando `head` y calcula:

- la suma del vector `v`,
- su media,
- y su longitud.

Respuesta:

```
v = seq(from = 0, to = 300, by = 3)
head(v, 20)
```

```
##  [1]  0  3  6  9 12 15 18 21 24 27 30 33 36 39 42 45 48 51 54 57
```

Suma de `v`

```
sum(v)
```

```
## [1] 15150
```

Media:

```
mean(v)
```

```
## [1] 150
```

Longitud:

```
length(v)
```

```
## [1] 101
```

Ejercicio 1.

Enunciado: Usando la función sample crea un vector dado_honesto con 100 números del 1 al 6. Haz una tabla de frecuencias absolutas (de dos maneras, con table y dplyr) y una tabla de frecuencias relativas.

Respuesta: Creo el vector:

```
set.seed(2021)
(dado_honesto <- sample(1:6,100, replace = T))
```

```
##   [1] 6 6 2 4 4 6 6 3 6 6 5 1 4 3 4 2 3 4 5 3 6 2 4 5 6 2 3 4 5 6 5 1 6 2 3 3 2
##  [38] 6 6 6 2 5 6 3 2 1 1 6 5 4 4 6 3 3 2 1 2 1 1 1 1 5 3 3 1 4 6 6 6 2 1 3 4 2
##  [75] 5 2 6 6 4 6 6 3 4 5 1 6 5 3 1 5 3 1 3 6 4 6 6 5 1 3
```

Tabla de frecuencias absolutas

```
table(dado_honesto)
```

```
## dado_honesto
##  1  2  3  4  5  6
## 15 13 18 14 13 27
```

```
library(dplyr)
```

```
##
## Attaching package: 'dplyr'

## The following objects are masked from 'package:stats':
##
##     filter, lag

## The following objects are masked from 'package:base':
##
##     intersect, setdiff, setequal, union
```

```
# Creo el dataframe para poder trabajar con dplyr
df <- data.frame(c(1:length(dado_honesto)),dado_honesto)
names(df) <- c("Tirada","dado_honesto")

df %>%
  count(dado_honesto)
```

```
##   dado_honesto  n
## 1            1 15
## 2            2 13
## 3            3 18
## 4            4 14
## 5            5 13
## 6            6 27
```

Tabla de frecuencias relativas

```
prop.table(table(dado_honesto))
```

```
## dado_honesto
##    1    2    3    4    5    6
## 0.15 0.13 0.18 0.14 0.13 0.27
```

```
      df %>%
        count(dado_honesto) %>%
          mutate(dado_honesto, relFreq = prop.table(n), n=NULL)
```

```
##   dado_honesto relFreq
## 1            1    0.15
## 2            2    0.13
## 3            3    0.18
## 4            4    0.14
## 5            5    0.13
## 6            6    0.27
```

Ejercicio 2.

Enunciado: A continuación crea un nuevo vector dado_cargado de manera que la probabilidad de que el número elegido valga 6 sea el doble que la probabilidad de elegir cualquiera de los cinco números restantes. Lee la ayuda de sample si lo necesitas. De nuevo, haz tablas de frecuencias absolutas y relativas de este segundo vector.

Respuesta: Creo el vector:

```
set.seed(2021)
(dado_cargado <- sample(1:6,100, replace = T, prob = c(rep(1/7,5), 2/7)))
```

```
##   [1] 4 2 5 3 5 5 5 6 2 1 6 2 5 4 2 6 4 1 1 4 6 1 3 6 5 1 1 1 1 2 6 1 1 1 4 6 1
##  [38] 6 1 4 6 2 3 1 6 2 1 6 6 2 4 5 6 4 3 6 4 3 1 2 6 4 6 3 2 6 6 5 6 6 6 5 6 1
##  [75] 6 4 4 3 5 1 4 3 1 5 3 6 1 2 4 3 6 5 1 4 6 1 5 2 2 5
```

Tabla de frecuencias absolutas

```
table(dado_cargado)
```

```
## dado_cargado
##  1  2  3  4  5  6
## 22 13 10 15 14 26
```

```
library(dplyr)
# Creo el dataframe para poder trabajar con dplyr
df2 <- data.frame(c(1:length(dado_cargado)),dado_cargado)
names(df) <- c("Tirada","dado_cargado")

df2 %>%
  count(dado_cargado)
```

```
##   dado_cargado  n
## 1            1 22
## 2            2 13
## 3            3 10
## 4            4 15
## 5            5 14
## 6            6 26
```

Tabla de frecuencias relativas

```
prop.table(table(dado_cargado))
```

```
## dado_cargado
##    1    2    3    4    5    6
## 0.22 0.13 0.10 0.15 0.14 0.26
```

```
      df2 %>%
        count(dado_cargado) %>%
          mutate(dado_cargado, relFreq = prop.table(n), n=NULL)
```

```
##   dado_cargado relFreq
## 1            1    0.22
## 2            2    0.13
## 3            3    0.10
## 4            4    0.15
## 5            5    0.14
## 6            6    0.26
```

Ejercicio 3.

Enunciado: Utiliza las funciones rep y seq para crear tres vectores v1, v2 y v3 con estos elementos respectivamente:

4, 4, 4, 4, 3, 3, 3, 3, 2, 2, 2, 2, 1, 1, 1, 1

1, 2, 2, 3, 3, 3, 4, 4, 4, 4, 5, 5, 5, 5, 5

1, 2, 3, 4, 1, 2, 3, 4, 1, 2, 3, 4, 1, 2, 3, 4

Respuesta: Creo los vectores:

```
(v1 <- sort(rep(seq(4:1), each = 4),decreasing = T))
```

```
##  [1] 4 4 4 4 3 3 3 3 2 2 2 2 1 1 1 1
```

```
(v2 <- rep(seq(1:5), times = 1:5))
```

```
##  [1] 1 2 2 3 3 3 4 4 4 4 5 5 5 5 5
```

```
(v3 <- rep(seq(4:1), times = 4))
```

```
##  [1] 1 2 3 4 1 2 3 4 1 2 3 4 1 2 3 4
```

Ejercicio 4.

Enunciado: Utilizando la tabla mpg de la librería tidyverse crea una tabla mpg2 que: I) contenga las filas en las que la variable class toma el valor pickup. II) y las columnas de la tabla original cuyos nombres empiezan por c. No se trata de que las selecciones a mano, por sus nombres

Respuesta: Hago la selección:

```
library(tidyverse)
```

```
## -- Attaching packages --------------------------------------- tidyverse 1.3.1 --

## v ggplot2 3.3.5     v purrr   0.3.4
## v tibble  3.1.4     v stringr 1.4.0
## v tidyr   1.1.3     v forcats 0.5.1
## v readr   2.0.1

## -- Conflicts ------------------------------------------ tidyverse_conflicts() --
## x dplyr::filter() masks stats::filter()
## x dplyr::lag()    masks stats::lag()
```

```
mpg %>%
  filter(class == "pickup") %>%
  select(starts_with("c"))
```

```
## # A tibble: 33 x 3
##      cyl   cty class
##    <int> <int> <chr>
##  1     6    15 pickup
##  2     6    14 pickup
##  3     6    13 pickup
##  4     6    14 pickup
##  5     8    14 pickup
##  6     8    14 pickup
##  7     8     9 pickup
##  8     8    11 pickup
##  9     8    11 pickup
## 10     8    12 pickup
## # ... with 23 more rows
```

Ejercicio 5.

Enunciado: Descarga el fichero census.dta. Averigua de qué tipo de fichero se trata y usa la herramienta Import DataSet del panel Environment de RStudio para leer con R los datos de ese fichero. Asegúrate de

copiar en esta práctica los dos primeros comandos que llevan a cabo la importación (excluye el comando View) y que descubrirás al usar esa herramienta. Después completa los siguientes apartados con esos datos y usando dplyr y ggplot:

Respuesta: Importo datos stata:

```
library(haven)
census <- read_dta("census.dta")
```

¿Cuáles son las poblaciones totales de las regiones censales?

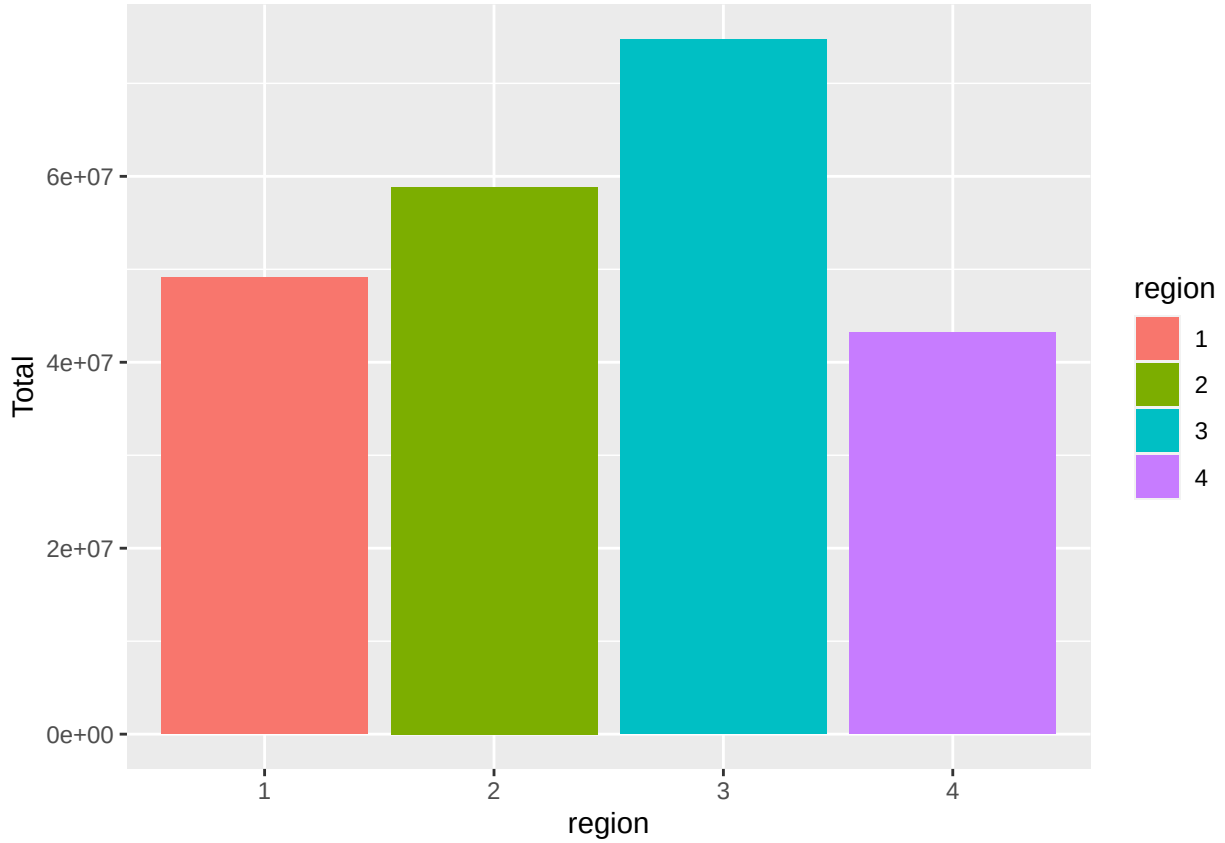
```
census %>%
    group_by(region)%>%
  summarise(Total = sum(pop))
```

```
## # A tibble: 4 x 2
##        region    Total
##     <dbl+lbl>    <dbl>
## 1 1 [NE]      49135283
## 2 2 [N Cntrl] 58865670
## 3 3 [South]   74734029
## 4 4 [West]    43172490
```

Representa esas poblaciones totales en un diagrama de barras (una barra por región censal).

```
pobl <- census %>%
         group_by(region)%>%
          summarise(Total = sum(pop))
pobl$region <- as.factor(pobl$region)

# Barplot
ggplot(pobl, aes(x=region, y=Total, fill=region)) +
  geom_bar(stat = "identity")
```



Ordena los estados por población, de mayor a menor.

```
# Con R
census$state[order(census$pop)]
```

```
##  [1] "Alaska"        "Wyoming"       "Vermont"       "Delaware"
##  [5] "N. Dakota"     "S. Dakota"     "Montana"       "Nevada"
##  [9] "New Hampshire" "Idaho"         "Rhode Island"  "Hawaii"
## [13] "Maine"         "New Mexico"    "Utah"          "Nebraska"
## [17] "W. Virginia"   "Arkansas"      "Kansas"        "Mississippi"
## [21] "Oregon"        "Arizona"       "Colorado"      "Iowa"
## [25] "Oklahoma"      "Connecticut"   "S. Carolina"   "Kentucky"
## [29] "Alabama"       "Minnesota"     "Washington"    "Louisiana"
## [33] "Maryland"      "Tennessee"     "Wisconsin"     "Missouri"
## [37] "Virginia"      "Georgia"       "Indiana"       "Massachusetts"
## [41] "N. Carolina"   "New Jersey"    "Michigan"      "Florida"
## [45] "Ohio"          "Illinois"      "Pennsylvania"  "Texas"
## [49] "New York"      "California"
```

```
# Con dplyr
census %>%
  select(state,pop) %>%
  arrange(across(pop,desc))
```

```
## # A tibble: 50 x 2
```

```
##    state             pop
##    <chr>           <dbl>
##  1 California   23667902
##  2 New York     17558072
##  3 Texas        14229191
##  4 Pennsylvania 11863895
##  5 Illinois     11426518
##  6 Ohio         10797630
##  7 Florida       9746324
##  8 Michigan      9262078
##  9 New Jersey    7364823
## 10 N. Carolina   5881766
## # ... with 40 more rows
```

Crea una nueva variable que contenga la tasa de divorcios /matrimonios para cada estado.

```
census <- census %>%
  mutate(tasa= divorce/marriage)

# Muestro el dataframe
head(census)
```

```
## # A tibble: 6 x 13
##   state        region    pop poplt5 pop5_17 pop18p pop65p popurban medage  death
##   <chr>      <dbl+lb>  <dbl>  <dbl>   <dbl>  <dbl>  <dbl>    <dbl>  <dbl>  <dbl>
## 1 Alabama    3 [Sout~ 3.89e6 2.96e5  865836 2.73e6 4.40e5  2337713   29.3  35305
## 2 Alaska     4 [West] 4.02e5 3.89e4   91796 2.71e5 1.15e4   258567   26.1   1604
## 3 Arizona    4 [West] 2.72e6 2.14e5  577604 1.93e6 3.07e5  2278728   29.2  21226
## 4 Arkansas   3 [Sout~ 2.29e6 1.76e5  495782 1.62e6 3.12e5  1179556   30.6  22676
## 5 California 4 [West] 2.37e7 1.71e6 4680558 1.73e7 2.41e6 21607606   29.9 186428
## 6 Colorado   4 [West] 2.89e6 2.16e5  592318 2.08e6 2.47e5  2329869   28.6  18925
## # ... with 3 more variables: marriage <dbl>, divorce <dbl>, tasa <dbl>
```

Si nos preguntamos cuáles son los estados más envejecidos podemos responder de dos maneras. Mirando la edad mediana o mirando en qué estados la franja de mayor edad representa una proporción más alta de la población total. Haz una tabla en la que aparezcan los valores de estos dos criterios, ordenada según la edad mediana decreciente y muestra los 10 primeros estados de esa tabla.

```
census %>%
  mutate(prop65= pop65p/pop) %>%
  select(state,medage,prop65) %>%
  arrange(desc(medage)) %>%
  head(10)
```

```
## # A tibble: 10 x 3
##    state         medage prop65
##    <chr>          <dbl>  <dbl>
##  1 Florida         34.7  0.173
##  2 New Jersey      32.2  0.117
##  3 Pennsylvania    32.1  0.129
##  4 Connecticut     32    0.117
##  5 New York        31.9  0.123
##  6 Rhode Island    31.8  0.134
```

```
##  7 Massachusetts   31.2  0.127
##  8 Missouri        30.9  0.132
##  9 Arkansas        30.6  0.137
## 10 Maine           30.4  0.125
```

Haz un histograma (con 10 intervalos) de los valores de la variable medage (edad mediana) y con la curva de densidad de la variable superpuesta.

```
ggplot(census, aes(x = medage)) +
  geom_histogram(aes(y=stat(density)),
                 bins=10, fill = "orange", color="black")  +
  geom_density(color="red", size=1.5)
```