Datos cuantitativos agrupados

Ramon Ceballos

2/2/2021

EJERCICIO NOTAS DE BACHILLERATO

Ejemplo de agrupamiento de datos cuantitativo

Se han recogido las notas de un examen de historia a los 100 alumnos de primero de bachillerato de un instituto.

Vamos a hacer uso de todo lo aprendido para obtener la mayor información posible utilizando las funciones **cut** e **hist** y también, las proporcionadas por nosotros.

Los resultados obtenidos en la encuesta han sido:

```
set.seed(4)
notas = sample(0:10,100, replace = TRUE)
set.seed(NULL)
```

```
#Notas obtenidos de los 100 alumnos
notas
```

```
##   [1]  7 10  2  2  6  2  5  4  9  2  7  5  1  7  0  3 10  2 10  4  1  4  5  4  0
##  [26]  5 10  4  3  0  7  5 10  3  4  8  1  9  3  7  9  1  9 10  5 10 10  9  5  0
##  [51]  3  1  3  2  0  6  6  4  7  4  7  3  9  0  7  0  3  0  3  3  1  4 10  9  1
##  [76]  4  0  6 10  0 10  1  0  2  6  4  8  2  3  7  7  3  3  8  2  6  6  2  8  9
```

Vamos a agrupar las notas en los siguientes intervalos:

$$[0,5),\ [5,7),\ [7,9),\ [9,10]$$

Claramente, estos 4 intervalos no tienen la misma amplitud.

Fijémonos también en que el último intervalo está cerrado por la derecha.

```
#Definimos vector de extremos de los intervalos
L = c(0,5,7,9,10)

#Definimos notas1 como el resultado de la codificación en intervalos
#utilizando como etiquetas los propios intervalos
notas1 = cut(
  notas,
  breaks = L,
```

```
  right = FALSE,
  include.lowest = TRUE) #incluye el último extremo

notas1
```

```
##   [1] [7,9)  [9,10] [0,5)  [0,5)  [5,7)  [0,5)  [5,7)  [0,5)  [9,10] [0,5)
##  [11] [7,9)  [5,7)  [0,5)  [7,9)  [0,5)  [0,5)  [9,10] [0,5)  [9,10] [0,5)
##  [21] [0,5)  [0,5)  [5,7)  [0,5)  [0,5)  [5,7)  [9,10] [0,5)  [0,5)  [0,5)
##  [31] [7,9)  [5,7)  [9,10] [0,5)  [0,5)  [7,9)  [0,5)  [9,10] [0,5)  [7,9)
##  [41] [9,10] [0,5)  [9,10] [9,10] [5,7)  [9,10] [9,10] [9,10] [5,7)  [0,5)
##  [51] [0,5)  [0,5)  [0,5)  [0,5)  [0,5)  [5,7)  [5,7)  [0,5)  [7,9)  [0,5)
##  [61] [7,9)  [0,5)  [9,10] [0,5)  [7,9)  [0,5)  [0,5)  [0,5)  [0,5)  [0,5)
##  [71] [0,5)  [0,5)  [9,10] [9,10] [0,5)  [0,5)  [0,5)  [5,7)  [9,10] [0,5)
##  [81] [9,10] [0,5)  [0,5)  [0,5)  [5,7)  [0,5)  [7,9)  [0,5)  [0,5)  [7,9)
##  [91] [7,9)  [0,5)  [0,5)  [7,9)  [0,5)  [5,7)  [5,7)  [0,5)  [7,9)  [9,10]
## Levels: [0,5) [5,7) [7,9) [9,10]
```

```
#Definimos las marcas de clase
MC = (L[1:length(L)-1]+L[2:length(L)])/2

#Definimos notas2 como el resultado de la codificación en intervalos
#utilizando como etiquetas las marcas de clase
notas2 = cut(
  notas,
  breaks = L,
  labels = MC,
  right = FALSE,
  include.lowest = TRUE)

notas2
```

```
##   [1] 8   9.5 2.5 2.5 6   2.5 6   2.5 9.5 2.5 8   6   2.5 8   2.5 2.5 9.5 2.5
##  [19] 9.5 2.5 2.5 2.5 6   2.5 2.5 6   9.5 2.5 2.5 2.5 8   6   9.5 2.5 2.5 8
##  [37] 2.5 9.5 2.5 8   9.5 2.5 9.5 9.5 6   9.5 9.5 9.5 6   2.5 2.5 2.5 2.5 2.5
##  [55] 2.5 6   6   2.5 8   2.5 8   2.5 9.5 2.5 8   2.5 2.5 2.5 2.5 2.5 2.5 2.5
##  [73] 9.5 9.5 2.5 2.5 2.5 6   9.5 2.5 9.5 2.5 2.5 2.5 6   2.5 8   2.5 2.5 8
##  [91] 8   2.5 2.5 8   2.5 6   6   2.5 8   9.5
## Levels: 2.5 6 8 9.5
```

```
#Definimos notas3 como el resultado de la codificación en intervalos
#utilizando como etiquetas la posición ordenada del intervalo (1, 2, 3 o 4)
notas3 = cut(
  notas,
  breaks = L,
  labels = FALSE,
  right = FALSE,
  include.lowest = TRUE)

notas3
```

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

```r
#Definimos notas4 como el resultado de la codificación en intervalos
#utilizando como etiquetas Susp, Aprob, Not y Exc
notas4 = cut(
  notas,
  breaks = L,
  labels = c("Susp", "Aprob", "Not", "Exc"),
  right = FALSE,
  include.lowest = TRUE)

notas4
```

```
##   [1] Not   Exc   Susp  Susp  Aprob Susp  Aprob Susp  Exc   Susp  Not   Aprob
##  [13] Susp  Not   Susp  Susp  Exc   Susp  Exc   Susp  Susp  Susp  Aprob Susp
##  [25] Susp  Aprob Exc   Susp  Susp  Susp  Not   Aprob Exc   Susp  Susp  Not
##  [37] Susp  Exc   Susp  Not   Exc   Susp  Exc   Exc   Aprob Exc   Exc   Exc
##  [49] Aprob Susp  Susp  Susp  Susp  Susp  Susp  Aprob Aprob Susp  Not   Susp
##  [61] Not   Susp  Exc   Susp  Not   Susp  Susp  Susp  Susp  Susp  Susp  Susp
##  [73] Exc   Exc   Susp  Susp  Susp  Aprob Exc   Susp  Exc   Susp  Susp  Susp
##  [85] Aprob Susp  Not   Susp  Susp  Not   Not   Susp  Susp  Not   Susp  Aprob
##  [97] Aprob Susp  Not   Exc
## Levels: Susp Aprob Not Exc
```

El resultado de **cut** ha sido, en cada caso, una lista con los elementos del vector original codificados con las etiquetas de las clases a las que pertenecen indicandolo en el parámetro **labels**.

Las dos primeras aplicaciones de la función **cut** han producido factores (cuyos niveles son los intervalos y las marcas de clase, respectivamente, en ambos casos ordenados de manera natural), mientras que aplicándole **labels = FALSE** hemos obtenido un vector.

¿Qué habría ocurrido si le hubiéramos pedido a R que cortase los datos en 4 intervalos?

Pues en este caso no nos hubiera servido de mucho, sobre todo porque la amplitud de nuestros intervalos era, desde buen inicio, diferente.

```r
cut(notas,
    breaks = 4,
    right = FALSE,
    include.lowest = TRUE)
```

```
##   [1] [5,7.5]      [7.5,10]     [-0.01,2.5) [-0.01,2.5) [5,7.5)      [-0.01,2.5)
##   [7] [5,7.5)      [2.5,5)      [7.5,10]    [-0.01,2.5) [5,7.5)      [5,7.5)
##  [13] [-0.01,2.5)  [5,7.5)      [-0.01,2.5) [2.5,5)     [7.5,10]     [-0.01,2.5)
##  [19] [7.5,10]     [2.5,5)      [-0.01,2.5) [2.5,5)     [5,7.5)      [2.5,5)
##  [25] [-0.01,2.5)  [5,7.5)      [7.5,10]    [2.5,5)     [2.5,5)      [-0.01,2.5)
##  [31] [5,7.5)      [5,7.5)      [7.5,10]    [2.5,5)     [2.5,5)      [7.5,10]
##  [37] [-0.01,2.5)  [7.5,10]     [2.5,5)     [5,7.5)     [7.5,10]     [-0.01,2.5)
##  [43] [7.5,10]     [7.5,10]     [5,7.5)     [7.5,10]    [7.5,10]     [7.5,10]
##  [49] [5,7.5)      [-0.01,2.5)  [2.5,5)     [-0.01,2.5) [2.5,5)      [-0.01,2.5)
##  [55] [-0.01,2.5)  [5,7.5)      [5,7.5)     [2.5,5)     [5,7.5)      [2.5,5)
##  [61] [5,7.5)      [2.5,5)      [7.5,10]    [-0.01,2.5) [5,7.5)      [-0.01,2.5)
##  [67] [2.5,5)      [-0.01,2.5)  [2.5,5)     [2.5,5)     [-0.01,2.5)  [2.5,5)
##  [73] [7.5,10]     [7.5,10]     [-0.01,2.5) [2.5,5)     [-0.01,2.5)  [5,7.5)
##  [79] [7.5,10]     [-0.01,2.5)  [7.5,10]    [-0.01,2.5) [-0.01,2.5)  [-0.01,2.5)
##  [85] [5,7.5)      [2.5,5)      [7.5,10]    [-0.01,2.5) [2.5,5)      [5,7.5)
```

```
##  [91] [5,7.5)     [2.5,5)     [2.5,5)     [7.5,10]    [-0.01,2.5) [5,7.5)
##  [97] [5,7.5)     [-0.01,2.5) [7.5,10]    [7.5,10]
## Levels: [-0.01,2.5) [2.5,5) [5,7.5) [7.5,10]
```

R ha repartido los datos en 4 intervalos de longitud 2.5, y ha desplazado ligeramente a la izquierda el extremo izquierdo del primer intervalo.

Trabajaremos ahora con **notas4** (suspenso, aprobado...) y calcularemos sus frecuencias:

```
table(notas4) #Fr. Abs
```

```
## notas4
##  Susp Aprob   Not   Exc
##    53    14    14    19
```

```
prop.table(table(notas4)) #Fr. Rel
```

```
## notas4
##  Susp Aprob   Not   Exc
##  0.53  0.14  0.14  0.19
```

```
cumsum(table(notas4)) #Fr. Abs. Cum
```

```
##  Susp Aprob   Not   Exc
##    53    67    81   100
```

```
cumsum(prop.table(table(notas4))) #Fr. Rel. Cum
```

```
##  Susp Aprob   Not   Exc
##  0.53  0.67  0.81  1.00
```

Podríamos haber obtenido todo lo anterior haciendo uso de la función **hist**.

```
notasHist = hist(
  notas,
  breaks = L,
  right = FALSE,
  include.lowest = TRUE,
  plot = FALSE)

notasHist
```

```
## $breaks
## [1]  0  5  7  9 10
##
## $counts
## [1] 53 14 14 19
##
## $density
## [1] 0.106 0.070 0.070 0.190
##
```

```
## $mids
## [1] 2.5 6.0 8.0 9.5
##
## $xname
## [1] "notas"
##
## $equidist
## [1] FALSE
##
## attr(,"class")
## [1] "histogram"
```

```
FAbs = notasHist$count #Frecuencias absolutas

FRel = prop.table(FAbs)

FAbsCum = cumsum(FAbs)

FRelCum = cumsum(FRel)
```

Ahora ya podemos crear un data frame con todas estas frecuencias:

```
intervalos = c("[0,5)","[5,7)","[7,9)","[9,10]")
calificacion = c("Suspenso", "Aprobado", "Notable", "Excelente")
marcas = notasHist$mids
tabla.Fr = data.frame(intervalos,calificacion,marcas,FAbs,FAbsCum,FRel,FRelCum)
tabla.Fr
```

```
##   intervalos calificacion marcas FAbs FAbsCum FRel FRelCum
## 1      [0,5)     Suspenso    2.5   53      53 0.53    0.53
## 2      [5,7)     Aprobado    6.0   14      67 0.14    0.67
## 3      [7,9)      Notable    8.0   14      81 0.14    0.81
## 4     [9,10]    Excelente    9.5   19     100 0.19    1.00
```

O bien, podríamos haber utilizado las funciones que os hemos proporcionado:

```
#Segunda función
TablaFrecs.L = function(x,L,V){
  x_cut = cut(x, breaks=L, right=FALSE, include.lowest=V)
  intervals = levels(x_cut)
  mc = (L[1:(length(L)-1)]+L[2:length(L)])/2
  Fr.abs = as.vector(table(x_cut))
  Fr.rel = round(Fr.abs/length(x),4)
  Fr.cum.abs = cumsum(Fr.abs)
  Fr.cum.rel = cumsum(Fr.rel)
  tabla = data.frame(intervals, mc, Fr.abs, Fr.cum.abs, Fr.rel, Fr.cum.rel)
  tabla
  }
```

```
TablaFrecs.L(notas, L, TRUE)
```

```
##   intervals  mc Fr.abs Fr.cum.abs Fr.rel Fr.cum.rel
```

```
## 1     [0,5) 2.5     53         53   0.53       0.53
## 2     [5,7) 6.0     14         67   0.14       0.67
## 3     [7,9) 8.0     14         81   0.14       0.81
## 4    [9,10] 9.5     19        100   0.19       1.00
```