# Práctica 4. Recodificación de variables

Jesús Esteban Hernández & Jesús Martín Fernández

## Contenidos



### 1. Introducción

El trabajo con datos en R es una parte esencial del análisis estadístico, y uno de los aspectos más importantes de este proceso es la recodificación de variables, especialmente en el caso de las variables de tipo factor. Los factores representan datos categóricos y, a menudo, contienen niveles que requieren ajustes para un análisis más preciso. Recodificar estos factores implica transformar sus niveles para que sean más representativos o relevantes para el estudio. Por ejemplo, puede ser necesario agrupar categorías, fusionar niveles o reordenar los factores de manera que reflejen un orden lógico. Además, los factores son esenciales para el correcto funcionamiento de muchos modelos estadísticos en R, ya que estos tratan las variables categóricas de manera diferente a las numéricas. No manejar adecuadamente los factores puede llevar a errores o interpretaciones incorrectas en los resultados. Por ello, es fundamental entender cómo trabajar con ellos y asegurarse de que las categorías reflejan de forma precisa la estructura de los datos.

### 2. Recodificación basada en intervalos de la distribución

Una de las formas más inmediatas de recodificar es recurrir a buscar puntos de corte en el rango de valores de un vector

Vamos a recuperar el dataframe df_prueba y a ver el recorrido de su variable (vector) edad

```
df_prueba <- read.csv("~/Práctica 4.csv")
range (df_prueba$edad)
```

```
[1] 42 83
```

Vemos que la edad se distribuye entre los 42 y 83 años, así que vamos a categorizarla en intervalos de décadas con la función `cut`

```
df_prueba$edad_r <- cut(df_prueba$edad, breaks = c(-Inf, 50, 60, 70, 80, Inf))

table (df_prueba$edad_r)
```

```
(-Inf,50]   (50,60]   (60,70]   (70,80] (80, Inf]
       17        21        29        24         9
```

Recuerda que si hubiésemos querido que las categorías estuviesen abiertas por la derecha, el código sería:

```
df_prueba$edad_r <- cut(df_prueba$edad,
                   breaks = c(-Inf, 50, 60, 70, 80, Inf), right=FALSE)

table (df_prueba$edad_r)
```

```
[-Inf,50)   [50,60)   [60,70)   [70,80) [80, Inf)
       16        20        28        25        11
```

## 3. Recodificar utilizando información de otros vectores

La creación de nuevas variables a partir de dos o más vectores existentes es una práctica común en el análisis de datos. Esta técnica permite generar información adicional o más compleja que puede ser útil para interpretar, analizar o modelar mejor los datos. Como ya hemos hecho anteriormente vamos a crear el vector `imc` a partir de los vectores `peso` y `alt`

```r
df_prueba$imc <- df_prueba$peso / (df_prueba$alt / 100)^2
df_prueba$imc <- round(df_prueba$imc, 2) #redondeamos a 2 decimales
```

Categoriza los valores en "Normal" si `imc`<30, y Obesidad" si `imc` >= 30, mira qué niveles tiene la nueva variable y en qué orden están

```r
df_prueba$imc <- cut(df_prueba$imc,
                                breaks = c(-Inf, 30, Inf),
                               labels = c("Normal", "Obesidad"))
df_prueba$imc
```

```
  [1] Normal   Normal   Normal   Normal   Normal   Obesidad Normal   Obesidad
  [9] Normal   Normal   Obesidad Normal   Normal   Obesidad Normal   Obesidad
 [17] Obesidad Obesidad Normal   Normal   Normal   Normal   Normal   Obesidad
 [25] Obesidad Normal   Normal   Normal   Obesidad Obesidad Obesidad Normal
 [33] Normal   Obesidad Normal   Obesidad Normal   Normal   Normal   Normal
 [41] Obesidad Normal   Normal   Obesidad Normal   Obesidad Normal   Normal
 [49] Normal   Normal   Normal   Normal   Normal   Normal   Normal   Normal
 [57] Normal   Normal   Normal   Normal   Normal   Normal   Obesidad Normal
 [65] Normal   Normal   Normal   Normal   Normal   Obesidad Normal   Normal
 [73] Normal   Normal   Obesidad Normal   Normal   Normal   Normal   Normal
 [81] Normal   Obesidad Normal   Normal   Normal   Normal   Normal   Normal
 [89] Obesidad Normal   Normal   Normal   Normal   Normal   Normal   Normal
 [97] Normal   Obesidad Normal   Normal
Levels: Normal Obesidad
```

La etiqueta `Levels` te indica que priero ha colocado el nivel "Normal" y luego "Obesidad"

## 4. Recodificación en variables tipo factor

La recodificación de variables categóricas es una de las tareas más comunes en la manipulación de datos. En R, este proceso se realiza utilizando factores, una clase especial para manejar variables categóricas. Los factores asignan a cada valor único del vector (variable) original un nivel numérico subyacente (level), mientras que la etiqueta es el valor visible asociado (label). La confusión habitual entre niveles y etiquetas se resuelve al entender que los niveles son los valores internos que utiliza R para operar, mientras que las etiquetas son las representaciones visibles de esos niveles.

Vamos a recorrer ese camino con un ejemplo. Primero, generamos un vector numérico `x` que contiene los valores `0` y `1` repetidos varias veces. Luego, convertimos este vector en un factor utilizando la función `factor()`, lo que permite a R identificar los valores únicos y asignarles

niveles numéricos subyacentes. Al no especificar etiquetas, R utiliza los valores originales como etiquetas y organiza los niveles en orden alfanumérico. Esto demuestra cómo R asigna niveles automáticamente a los valores de una variable categórica al convertirlos en factores.

```
df_prueba$sex <- factor (df_prueba$sex)
df_prueba$sex
```

```
  [1] 0 1 0 1 1 0 1 1 1 0 1 0 1 1 0 1 0 0 0 1 1 1 1 1 1 1 1 0 0 1 1 1 1 0 0 1
 [38] 0 0 0 0 0 0 0 0 0 0 0 0 0 1 0 0 1 0 1 0 0 1 1 0 1 0 0 0 1 0 1 1 1 0 1 1 1 0
 [75] 0 0 0 1 0 0 0 1 0 1 0 0 1 1 1 0 0 1 0 1 0 0 1 0 0 1
Levels: 0 1
```

Si no asignas etiquetas explícitas al convertir un vector numérico en un factor en R, los valores visibles (las etiquetas) que asigna el factor serán los mismos que los valores originales del vector. Por ejemplo, si tienes un vector con valores `0` y `1` y lo conviertes en un factor sin especificar etiquetas, R usará esos mismos valores (`0` y `1`) como etiquetas visibles, pero internamente sigue asignando niveles numéricos (1 para "0" y 2 para "1", según el orden alfanumérico). Así, aunque visualmente parezcan iguales, los números internos (niveles) son distintos de las etiquetas que ves. Puedes comprobarlo conviertiendo el vector en numérico.

```
df_prueba$sex2<- as.numeric(df_prueba$sex)
df_prueba$sex2
```

```
  [1] 1 2 1 2 2 1 2 2 2 1 2 1 2 2 1 2 1 1 1 2 2 2 2 2 2 2 2 1 1 2 2 2 2 1 1 2
 [38] 1 1 1 1 1 1 1 1 1 1 1 1 2 1 1 2 1 2 1 1 2 2 1 2 1 1 1 2 1 2 2 2 1 2 2 2 1
 [75] 1 1 1 2 1 1 1 2 1 2 1 1 2 2 2 1 1 2 1 2 1 1 2 1 1 2
```

Ahora vamos a asignar etiquetas al factor. Vamos a pedirle que asigne la etiqueta "Mujer" al nivel `1` (que se corresponde con el valor "0") y "Hombre" al nivel `2` (que corresponde al valor "1").

```
df_prueba$sex2 <- factor(df_prueba$sex, labels = c("Mujer", "Hombre"))
df_prueba$sex2
```

```
  [1] Mujer  Hombre Mujer  Hombre Hombre Mujer  Hombre Hombre Hombre Mujer
 [11] Hombre Mujer  Hombre Hombre Mujer  Hombre Mujer  Mujer  Mujer  Hombre
 [21] Hombre Hombre Hombre Hombre Hombre Hombre Hombre Hombre Mujer  Mujer
 [31] Hombre Hombre Hombre Hombre Mujer  Mujer  Hombre Mujer  Mujer  Mujer
 [41] Mujer  Mujer  Mujer  Mujer  Mujer  Mujer  Mujer  Mujer  Mujer  Hombre
 [51] Mujer  Mujer  Hombre Mujer  Hombre Mujer  Mujer  Hombre Hombre Mujer
```

```
 [61] Hombre Mujer  Mujer  Mujer  Hombre Mujer  Hombre Hombre Hombre Mujer
 [71] Hombre Hombre Hombre Mujer  Mujer  Mujer  Mujer  Hombre Mujer  Mujer
 [81] Mujer  Hombre Mujer  Hombre Mujer  Mujer  Hombre Hombre Hombre Mujer
 [91] Mujer  Hombre Mujer  Hombre Mujer  Mujer  Hombre Mujer  Mujer  Hombre
Levels: Mujer Hombre
```

Pero los niveles subyacentes seguirán siendo `1` y `2`

```
df_prueba$sex2<- as.numeric (df_prueba$sex2)
df_prueba$sex2
```

```
  [1] 1 2 1 2 2 1 2 2 2 1 2 1 2 2 1 2 1 1 1 2 2 2 2 2 2 2 2 2 1 1 2 2 2 2 1 1 2
 [38] 1 1 1 1 1 1 1 1 1 1 1 1 2 1 1 2 1 2 1 1 2 2 1 2 1 1 1 2 1 2 2 2 1 2 2 2 1
 [75] 1 1 1 2 1 1 1 2 1 2 1 1 2 2 2 1 1 2 1 2 1 1 2 1 1 2
```

Si queremos saber los niveles de un vector factor, simplemente hay que usar la función `levels`

```
df_prueba$sex2 <- factor(df_prueba$sex, labels = c("Mujer", "Hombre"))
levels (df_prueba$sex2)
```

```
[1] "Mujer"  "Hombre"
```

Finalmente, señalamos que se pueden cambiar de orden los niveles a criterio del investigador , utilizando también la función `level`

```
df_prueba$sex3 <- factor(df_prueba$sex, levels = c("1", "0"))
df_prueba$sex3
```

```
  [1] 0 1 0 1 1 0 1 1 1 0 1 0 1 1 0 1 0 0 0 1 1 1 1 1 1 1 1 0 0 1 1 1 1 0 0 1
 [38] 0 0 0 0 0 0 0 0 0 0 0 0 1 0 0 1 0 1 0 0 1 1 0 1 0 0 0 1 0 1 1 1 0 1 1 1 0
 [75] 0 0 0 1 0 0 0 1 0 1 0 0 1 1 1 0 0 1 0 1 0 0 1 0 0 1
Levels: 1 0
```

```
df_prueba$sex3 <- factor(df_prueba$sex3, labels = c("Mujer", "Hombre"))
df_prueba$sex3
```

```
  [1] Hombre Mujer  Hombre Mujer  Mujer  Hombre Mujer  Mujer  Mujer  Hombre
 [11] Mujer  Hombre Mujer  Mujer  Hombre Mujer  Hombre Hombre Hombre Mujer
 [21] Mujer  Mujer  Mujer  Mujer  Mujer  Mujer  Mujer  Mujer  Hombre Hombre
 [31] Mujer  Mujer  Mujer  Mujer  Hombre Hombre Mujer  Hombre Hombre Hombre
 [41] Hombre Hombre Hombre Hombre Hombre Hombre Hombre Hombre Hombre Mujer
 [51] Hombre Hombre Mujer  Hombre Mujer  Hombre Hombre Mujer  Mujer  Hombre
 [61] Mujer  Hombre Hombre Hombre Mujer  Hombre Mujer  Mujer  Mujer  Hombre
 [71] Mujer  Mujer  Mujer  Hombre Hombre Hombre Hombre Mujer  Hombre Hombre
 [81] Hombre Mujer  Hombre Mujer  Hombre Hombre Mujer  Mujer  Mujer  Hombre
 [91] Hombre Mujer  Hombre Mujer  Hombre Hombre Mujer  Hombre Hombre Mujer
Levels: Mujer Hombre
```

La importancia del orden en los niveles de un factor en R radica en cómo R trata esas categorías en modelos estadísticos, especialmente en modelos lineales o regresiones. En estos modelos, uno de los niveles de la variable categórica se utiliza como nivel de referencia, es decir, como punto base para comparar las otras categorías.

Por defecto, R usa el nivel más bajo (en orden alfanumérico) como referencia. Por ejemplo, si tienes un factor con los niveles "Mujer" (1) y "Hombre" (2) (como ocurría con `sex` y `sex2`, el modelo tomará "Mujer" como referencia y calculará los efectos de "Hombre" en relación a "Mujer". Esto influye en cómo se interpretan los coeficientes del modelo: los valores obtenidos para "Hombre" serán comparaciones con respecto a "Mujer". Cambiar el orden de los niveles altera qué categoría se usa como referencia, lo que puede ser crucial para una correcta interpretación de los resultados.

Si prefieres que "Hombre" sea la referencia en lugar de "Mujer", debes ajustar los niveles del factor para que "Hombre" sea el primer nivel (como ocurría con la variable `sex3`).

## 5. Haciendo factor un vector `character`

Es común convertir un vector de texto en un factor. Supongamos que tenemos un vector con 100 valores con categorías socioeconómicas `nse`, como cadenas de texto que representan 4 niveles:

```
nse <- c(rep("nse1", 18), rep("nse2", 28), rep("nse3", 32), rep("nse4", 22))
```

Este vector tiene valores repetidos de cada nivel socioeconómico (nse1, nse2, etc). Al convertirlo en un factor, R asignará niveles según el orden alfanumérico de las etiquetas:

```
nse_f <- factor(nse)
nse_f
```

```
  [1] nse1 nse1 nse1 nse1 nse1 nse1 nse1 nse1 nse1 nse1 nse1 nse1 nse1 nse1 nse1
 [16] nse1 nse1 nse1 nse2 nse2 nse2 nse2 nse2 nse2 nse2 nse2 nse2 nse2 nse2 nse2
 [31] nse2 nse2 nse2 nse2 nse2 nse2 nse2 nse2 nse2 nse2 nse2 nse2 nse2 nse2 nse2
 [46] nse2 nse3 nse3 nse3 nse3 nse3 nse3 nse3 nse3 nse3 nse3 nse3 nse3 nse3 nse3
 [61] nse3 nse3 nse3 nse3 nse3 nse3 nse3 nse3 nse3 nse3 nse3 nse3 nse3 nse3 nse3
 [76] nse3 nse3 nse3 nse4 nse4 nse4 nse4 nse4 nse4 nse4 nse4 nse4 nse4 nse4 nse4
 [91] nse4 nse4 nse4 nse4 nse4 nse4 nse4 nse4 nse4 nse4
Levels: nse1 nse2 nse3 nse4
```

R ordena automáticamente los niveles en orden alfabético: `nse1` , `nse2` , `nse3` y `nse4`

Imaginemos que, por error o por otro motivo, se introducen códigos diferentes como `nse2`

```
nse2 <- c(rep("nse1", 18), rep ("sen2", 10), rep("nse2", 18), rep("nse3", 32), rep("nse4", 2
nse2
```

```
  [1] "nse1" "nse1" "nse1" "nse1" "nse1" "nse1" "nse1" "nse1" "nse1" "nse1"
 [11] "nse1" "nse1" "nse1" "nse1" "nse1" "nse1" "nse1" "nse1" "sen2" "sen2"
 [21] "sen2" "sen2" "sen2" "sen2" "sen2" "sen2" "sen2" "sen2" "nse2" "nse2"
 [31] "nse2" "nse2" "nse2" "nse2" "nse2" "nse2" "nse2" "nse2" "nse2" "nse2"
 [41] "nse2" "nse2" "nse2" "nse2" "nse2" "nse2" "nse3" "nse3" "nse3" "nse3"
 [51] "nse3" "nse3" "nse3" "nse3" "nse3" "nse3" "nse3" "nse3" "nse3" "nse3"
 [61] "nse3" "nse3" "nse3" "nse3" "nse3" "nse3" "nse3" "nse3" "nse3" "nse3"
 [71] "nse3" "nse3" "nse3" "nse3" "nse3" "nse3" "nse3" "nse3" "nse4" "nse4"
 [81] "nse4" "nse4" "nse4" "nse4" "nse4" "nse4" "nse4" "nse4" "nse4" "nse4"
 [91] "nse4" "nse4" "nse4" "nse4" "nse4" "nse4" "nse4" "nse4" "nse4" "nse4"
```

Al hacer esta nueva variable `nse2`factor, R reordenará las etiquetas alfabéticamente

```
nse2_f<- factor (nse2)
nse2_f
```

```
  [1] nse1 nse1 nse1 nse1 nse1 nse1 nse1 nse1 nse1 nse1 nse1 nse1 nse1 nse1 nse1
 [16] nse1 nse1 nse1 sen2 sen2 sen2 sen2 sen2 sen2 sen2 sen2 sen2 sen2 nse2 nse2
 [31] nse2 nse2 nse2 nse2 nse2 nse2 nse2 nse2 nse2 nse2 nse2 nse2 nse2 nse2 nse2
 [46] nse2 nse3 nse3 nse3 nse3 nse3 nse3 nse3 nse3 nse3 nse3 nse3 nse3 nse3 nse3
 [61] nse3 nse3 nse3 nse3 nse3 nse3 nse3 nse3 nse3 nse3 nse3 nse3 nse3 nse3 nse3
 [76] nse3 nse3 nse3 nse4 nse4 nse4 nse4 nse4 nse4 nse4 nse4 nse4 nse4 nse4 nse4
 [91] nse4 nse4 nse4 nse4 nse4 nse4 nse4 nse4 nse4 nse4
Levels: nse1 nse2 nse3 nse4 sen2
```

### 5.1 Recolocando los niveles

Hemos visto, al ejecutar el código anterior que, aunque se presentan los valores (las filas) en el orden en que se han creado, cuando R ajusta los nivles, coloca a `sne2` después de `nse4` debido al orden alfabético. Si queremos forzar un orden específico para los niveles, lo podemos hacer al crear el factor:

```
nse2_f2 <- factor(nse2, levels = c("nse1", "sen2", "nse2", "nse3", "nse4"))
nse2_f2
```

```
  [1] nse1 nse1 nse1 nse1 nse1 nse1 nse1 nse1 nse1 nse1 nse1 nse1 nse1 nse1 nse1
 [16] nse1 nse1 nse1 sen2 sen2 sen2 sen2 sen2 sen2 sen2 sen2 sen2 sen2 nse2 nse2
 [31] nse2 nse2 nse2 nse2 nse2 nse2 nse2 nse2 nse2 nse2 nse2 nse2 nse2 nse2 nse2
 [46] nse2 nse3 nse3 nse3 nse3 nse3 nse3 nse3 nse3 nse3 nse3 nse3 nse3 nse3 nse3
 [61] nse3 nse3 nse3 nse3 nse3 nse3 nse3 nse3 nse3 nse3 nse3 nse3 nse3 nse3 nse3
 [76] nse3 nse3 nse3 nse4 nse4 nse4 nse4 nse4 nse4 nse4 nse4 nse4 nse4 nse4 nse4
 [91] nse4 nse4 nse4 nse4 nse4 nse4 nse4 nse4 nse4 nse4
Levels: nse1 sen2 nse2 nse3 nse4
```

### 5.2 Recodificación de factores

La recodificación también permite reorganizar los niveles dentro de un vector (o variable). Veamos un ejemplo. Vamos a agrupar los dos valores centrales en valores en la variable `nse_f`

```
nse_f_rec <- nse_f
levels(nse_f_rec) <- c("nse1", "nse_2_3", "nse_2_3", "nse_4")
nse_f_rec
```

```
  [1] nse1    nse1    nse1    nse1    nse1    nse1    nse1    nse1    nse1
 [10] nse1    nse1    nse1    nse1    nse1    nse1    nse1    nse1    nse1
 [19] nse_2_3 nse_2_3 nse_2_3 nse_2_3 nse_2_3 nse_2_3 nse_2_3 nse_2_3 nse_2_3
 [28] nse_2_3 nse_2_3 nse_2_3 nse_2_3 nse_2_3 nse_2_3 nse_2_3 nse_2_3 nse_2_3
 [37] nse_2_3 nse_2_3 nse_2_3 nse_2_3 nse_2_3 nse_2_3 nse_2_3 nse_2_3 nse_2_3
 [46] nse_2_3 nse_2_3 nse_2_3 nse_2_3 nse_2_3 nse_2_3 nse_2_3 nse_2_3 nse_2_3
 [55] nse_2_3 nse_2_3 nse_2_3 nse_2_3 nse_2_3 nse_2_3 nse_2_3 nse_2_3 nse_2_3
 [64] nse_2_3 nse_2_3 nse_2_3 nse_2_3 nse_2_3 nse_2_3 nse_2_3 nse_2_3 nse_2_3
 [73] nse_2_3 nse_2_3 nse_2_3 nse_2_3 nse_2_3 nse_2_3 nse_4   nse_4   nse_4
 [82] nse_4   nse_4   nse_4   nse_4   nse_4   nse_4   nse_4   nse_4   nse_4
 [91] nse_4   nse_4   nse_4   nse_4   nse_4   nse_4   nse_4   nse_4   nse_4
[100] nse_4
Levels: nse1 nse_2_3 nse_4
```

Se puede ver que solo hay tres niveles y que se han reagrupado los dos valores centrales.

### 5.3 Reordenar los niveles

Recordamos la importancia del orden de lo sniveles, independientemente de las etiquetas que presenten. Tanto en las regresiones, como en lo smodelos lineales generalizados, R, por defecto, utilizará como catgoría de comparación el nivel mñas bajo. En algunos momentos puede interesarnos que el nivel decomparación sea otro. Vamos a hacer que R ponga como primer nivel `nse4`

```
nse_f_reord<-relevel(nse_f, ref = "nse4")
nse_f_reord
```

```
  [1] nse1 nse1 nse1 nse1 nse1 nse1 nse1 nse1 nse1 nse1 nse1 nse1 nse1 nse1 nse1
 [16] nse1 nse1 nse1 nse2 nse2 nse2 nse2 nse2 nse2 nse2 nse2 nse2 nse2 nse2 nse2
 [31] nse2 nse2 nse2 nse2 nse2 nse2 nse2 nse2 nse2 nse2 nse2 nse2 nse2 nse2 nse2
 [46] nse2 nse3 nse3 nse3 nse3 nse3 nse3 nse3 nse3 nse3 nse3 nse3 nse3 nse3 nse3
 [61] nse3 nse3 nse3 nse3 nse3 nse3 nse3 nse3 nse3 nse3 nse3 nse3 nse3 nse3 nse3
 [76] nse3 nse3 nse3 nse4 nse4 nse4 nse4 nse4 nse4 nse4 nse4 nse4 nse4 nse4 nse4
 [91] nse4 nse4 nse4 nse4 nse4 nse4 nse4 nse4 nse4 nse4
Levels: nse4 nse1 nse2 nse3
```

Fíjate que, independientemente del orden en que visualices los valores del vector, cuando te muestra los `Levels` el primero es `nse4` . Ahora, en cualquier análisis que pidamos, la categoría de referencia será `nse4`