Módulo 2 — Tidyverse I

ifelse() y case_when()

CEPAL - Unidad de Estadísticas Sociales

2025-10-31

Introducción

En análisis de datos, con frecuencia necesitamos crear **nuevas variables** que dependan de condiciones lógicas.

Dentro del *tidyverse*, las funciones más comunes para ello son:

- `if_else()`: Evalúa una única condición lógica y retorna dos valores posibles (verdadero o falso).
- `case_when()`: Permite evaluar **múltiples condiciones** secuencialmente, devolviendo distintos valores según el caso.

Ambas funciones son compatibles con `mutate()` y permiten generar variables derivadas sin salir del flujo del *pipe* (`%>%`).

Lectura de datos

```
datos <- readRDS("data/base_personas_gasto.rds")
library(dplyr)
datos %>% mutate(log_ingreso = log(ingreso)  ) %>%
  select(id_hogar, id_pers, ingreso, log_ingreso) %>% head(4)
```

id_hogar	id_pers	ingreso	log_ingreso
262	1	5391.527	8.592584
262	2	5391.527	8.592584
265	1	7077.083	8.864617
265	2	7105.557	8.868632

Ejemplo básico con `if_else()`

```r
library(dplyr)

datos_if <- datos %>%
  mutate(
    adulto = if_else(edad >= 18, "Adulto", "Menor"),
    ingreso_cat = if_else(ingreso >= median(ingreso, na.rm = TRUE),
                          "Alto", "Bajo")
  ) %>%
  select(id_hogar, edad, ingreso, adulto, ingreso_cat)
```

Resultado

```
head(datos_if, 5)
```

id_hogar	edad	ingreso	adulto	ingreso_cat
262	51	5391.527	Adulto	Alto
262	46	5391.527	Adulto	Alto
265	26	7077.083	Adulto	Alto
265	24	7105.557	Adulto	Alto
265	7	7077.083	Menor	Alto

if_else() dentro de across()

Podemos aplicar la misma lógica a varias columnas simultáneamente.

```
datos_if2 <- datos %>%
  mutate(
    across(
      c(ingreso, gasto),
      ~ if_else(.x >= median(.x, na.rm = TRUE), "Alto", "Bajo"),
      .names = "cat_{.col}"
    )
  ) %>%
  select(id_hogar, ingreso, gasto, cat_ingreso, cat_gasto)
```

Resultado

```
head(datos_if2, 5)
```

id_hogar	ingreso	gasto	cat_ingreso	cat_gasto
262	5391.527	5391.527	Alto	Alto
262	5391.527	5391.527	Alto	Alto
265	7077.083	7077.083	Alto	Alto
265	7105.557	7105.557	Alto	Alto
265	7077.083	7077.083	Alto	Alto

¿Qué hace `case_when()`?

- Evalúa **múltiples condiciones** y devuelve diferentes valores según el caso.
- Es más flexible que `if_else()` y facilita clasificaciones multinivel.
- Se lee como una tabla de decisiones.

Ejemplo: clasificación por edad

```
datos_case <- datos %>%
  mutate(
    etapa_vida = case_when(
      edad < 13 ~ "Niñez",
      edad >= 13 & edad < 18 ~ "Adolescencia",
      edad >= 18 & edad < 60 ~ "Adultez",
      edad >= 60 ~ "Vejez",
      TRUE ~ NA_character_
    )
  ) %>%
  select(id_hogar, edad, etapa_vida)
```

Resultado

```
head(datos_case, 5)
```

id_hogar	edad	etapa_vida
262	51	Adultez
262	46	Adultez
265	26	Adultez
265	24	Adultez
265	7	Niñez

Ejemplo: niveles educativos

```r
datos_case2 <- datos %>%
  mutate(
    niveduc_ee = haven::as_factor(niveduc_ee, levels = "values" ),
    niveduc_ee = as.numeric(niveduc_ee),
    nivel_edu_cat = case_when(
      niveduc_ee <= 3 ~ "Baja",
      niveduc_ee <= 6 ~ "Media",
      niveduc_ee > 6  ~ "Alta",
      TRUE ~ "No reporta"
    )
  ) %>%
  select(id_hogar, niveduc_ee, nivel_edu_cat)
```

Resultado

```
head(datos_case2, 5)
```

id_hogar	niveduc_ee	nivel_edu_cat
262	7	Alta
262	5	Media
265	7	Alta
265	6	Media
265	1	Baja

Integración con transformaciones previas

Aquí combinamos `filter()`, `mutate()`, `round()` y condicionales.

```
datos_final <- datos %>%
  filter(edad >= 15, !is.na(ingreso)) %>%
  mutate(
    ingreso_mil = round(ingreso, -3),
    gasto_mil   = round(gasto, -3),
    situacion = case_when(
      gasto_mil > ingreso_mil ~ "Gasta más de lo que gana",
      gasto_mil == ingreso_mil ~ "Equilibrado",
      gasto_mil < ingreso_mil ~ "Ahorra",
      TRUE ~ NA_character_
    ),
    grupo = if_else(sexo == "Mujer" & edad >= 18,
                    "Mujer adulta", "Otro grupo")
  ) %>%
  select(id_hogar, sexo, edad, ingreso_mil, gasto_mil, situacion, grupo)
```

Resultado

```
head(datos_final, 6)
```

id_hogar	sexo	edad	ingreso_mil	gasto_mil	situacion	grupo
262	Hombre	51	5000	5000	Equilibrado	Otro grupo
262	Mujer	46	5000	5000	Equilibrado	Mujer adulta
265	Mujer	26	7000	7000	Equilibrado	Mujer adulta
265	Hombre	24	7000	7000	Equilibrado	Otro grupo
277	Mujer	42	2000	2000	Equilibrado	Mujer adulta
277	Mujer	20	2000	2000	Equilibrado	Mujer adulta

Combinar across() con case_when()

```
datos_multi <- datos %>%
  mutate(
    across(
      c(ingreso, gasto),
      ~ case_when(
        .x < quantile(.x, 0.25, na.rm = TRUE) ~ "Bajo",
        .x < quantile(.x, 0.75, na.rm = TRUE) ~ "Medio",
        .x >= quantile(.x, 0.75, na.rm = TRUE) ~ "Alto"
      ),
      .names = "cuartil_{.col}"
    )
  ) %>%
  select(id_hogar, ingreso, gasto, cuartil_ingreso, cuartil_gasto)
```

Resultado

```
head(datos_multi, 5)
```

id_hogar	ingreso	gasto	cuartil_ingreso	cuartil_gasto
262	5391.527	5391.527	Alto	Alto
262	5391.527	5391.527	Alto	Alto
265	7077.083	7077.083	Alto	Alto
265	7105.557	7105.557	Alto	Alto
265	7077.083	7077.083	Alto	Alto

Cierre

- `if_else()` se usa para **dos condiciones exclusivas**.
- `case_when()` permite **clasificar múltiples casos**.
- Ambas se integran con `mutate()` y `across()` para construir variables analíticas limpias y reproducibles.