Factors, and the order a report needs

What a factor really is, the as.numeric() trap that silently returns level positions, and the forcats verbs that put a JMP ladder in ladder order instead of alphabetical order.

Pierre Robentz Cassion

Lesson 4 of 8

R and the Tidyverse for Programme Data — Data that keeps its meaning

What this lesson covers

- A factor is an integer with a lookup table
- The `as.numeric()` trap
- Setting the order you want
- Ordered factors
- Unused levels, and the two defaults that disagree
- Collapsing a long tail
- Factors and missing values
- Where factors bite in a join
- What comes next

A factor is an integer with a lookup table — In R

```r
service <- factor(c("basic", "limited", "unimproved", "basic"))

levels(service)
#> [1] "basic"      "limited"    "unimproved"

as.integer(service)
#> [1] 1 2 3 1
```

The `as.numeric()` trap — In R

```r
age <- factor(c("10", "9", "8"))

as.numeric(age)
#> [1] 1 3 2
```

The `as.numeric()` trap — In R

```r
as.numeric(as.character(age))
#> [1] 10 9 8
```

Setting the order you want — In R

```
library(forcats)

ladder <- fct_relevel(service, "unimproved", "limited", "basic")
levels(ladder)
#> [1] "unimproved" "limited"    "basic"
```

Setting the order you want — In R

```r
fct_infreq(source)      # most common level first
fct_reorder(name, value)  # order one factor by another column
```

Setting the order you want — In R

```
library(dplyr)

by_commune |>
  mutate(commune = fct_reorder(commune, gam_rate)) |>
  ggplot2::ggplot(ggplot2::aes(gam_rate, commune)) +
  ggplot2::geom_col()
```

Ordered factors — In R

```
service[1] >= service[2]
#> Warning: '>=' not meaningful for factors
#> [1] NA
```

Ordered factors — In R

```r
ladder <- factor(
  c("basic", "limited"),
  levels = c("unimproved", "limited", "basic"),
  ordered = TRUE
)

ladder[1] >= ladder[2]
#> [1] TRUE
```

Ordered factors

- Use ordered factors sparingly — They change how models treat the variable (polynomial contrasts rather than dummies),...

Unused levels, and the two defaults that disagree — In R

```r
d <- tibble::tibble(k = factor(c("a", "b", "a"), levels = c("a", "b", "c")))

nrow(dplyr::count(d, k))
#> [1] 2

length(table(d$k))
#> [1] 3
```

Unused levels, and the two defaults that disagree — In R

```r
dplyr::count(d, k, .drop = FALSE)   # keep the empty level
droplevels(d$k)                     # remove levels with no rows
forcats::fct_drop(d$k)              # the forcats spelling of the same thing
```

Collapsing a long tail — In R

```
source <- factor(wash$water_source)
length(levels(source))
#> [1] 10

levels(fct_lump_n(source, 4))
#> [1] "borehole" "piped-into-dwelling" "piped-into-yard" "public-tap" "Other"
```

Collapsing a long tail

- Do not lump before computing an indicator — `Other` here mixes protected springs, which are improved, with surface…

Collapsing a long tail — In R

```
fct_recode(source,
  improved = "borehole",
  improved = "protected-well",
  unimproved = "surface-water"
)
```

Factors and missing values — In R

```
table(muac$outcome, useNA = "ifany")
```

Factors and missing values — In R

```
muac <- muac |> mutate(outcome = fct_na_value_to_level(outcome, level = "not recorded"))
```

Where factors bite in a join — In R

```
left_join(muac, sites, by = "commune")
```

What comes next

- Columns now hold their meaning and their order.

Read the full lesson, with runnable code

https://data-analysis.cassion.dev/courses/r-for-programme-data/r-04-factors