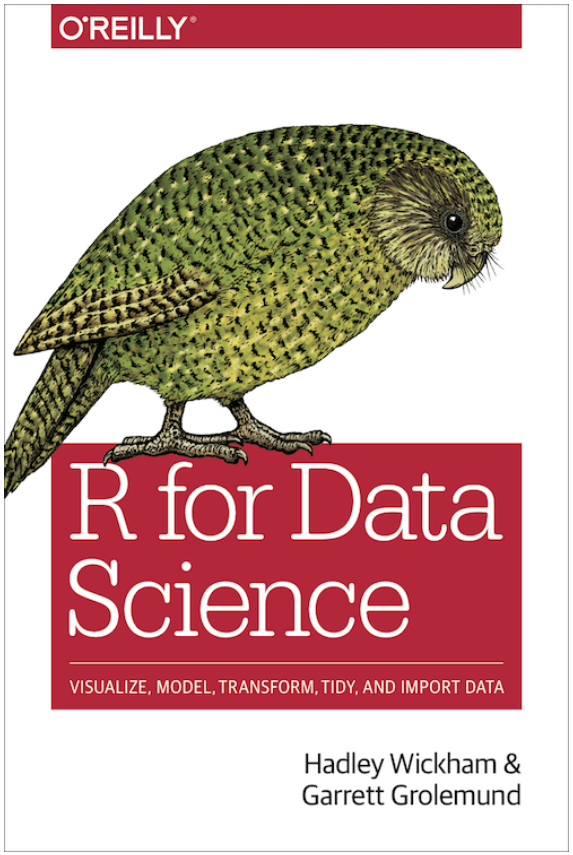
Socio-Informatics 348

Data Transformation
Factors

Dr Lisa Martin

Department of Information Science
Stellenbosch University

Today's Reading



R for Data Science, Chapter 16

What is a factor?

- Used for categorical variables, variables that have a fixed and known set of possible values (levels).
- Useful for displaying character vectors in plots and models.

Basics:

- Created with `factor()` function.
- Start by creating a vector of the levels in the desired order.
- Otherwise: Levels are stored in alphabetical order by default.

```r
x1 <- c("Dec", "Apr", "Jan", "Mar")
```

```r
month_levels <- c(
  "Jan", "Feb", "Mar", "Apr", "May", "Jun",
  "Jul", "Aug", "Sep", "Oct", "Nov", "Dec"
)
```

Basics:

- Created with `factor()` function.
- Start by creating a vector of the levels in the desired order.
- Otherwise: Levels are stored in alphabetical order by default.
- Use `levels()` to check the levels of a factor.

```
y1 <- factor(x1, levels = month_levels)
y1
#> [1] Dec Apr Jan Mar
#> Levels: Jan Feb Mar Apr May Jun Jul Aug Sep Oct Nov Dec

sort(y1)
#> [1] Jan Mar Apr Dec
#> Levels: Jan Feb Mar Apr May Jun Jul Aug Sep Oct Nov Dec
```

`forcats` package

- Part of the `tidyverse`.
- `forcats` provides many useful functions for working with factors.
- Use `forcats::fct()` as an alternative to `factor`.
- Handles NA values more strictly than base R.

```
x2 <- c("Dec", "Apr", "Jam", "Mar")
```

```
y2 <- fct(x2, levels = month_levels)
#> Error in `fct()`:
#> ! All values of `x` must appear in `levels` or `na`
#> ℹ Missing level: "Jam"
```

forcats package

- Part of the tidyverse.
- forcats provides many useful functions for working with factors.
- Use forcats::fct() as an alternative to factor.
- Handles NA values more strictly than base R.
- Handles missing levels differently than base R.

```
fct(x1)
#> [1] Dec Apr Jan Mar
#> Levels: Dec Apr Jan Mar
```

readr

- Levels can also be set whnen reading in data with `readr`.
- Use the `col_factor()` function in the `col_types` argument of `read_csv()`.
- Specify the levels in the order you want them to appear.

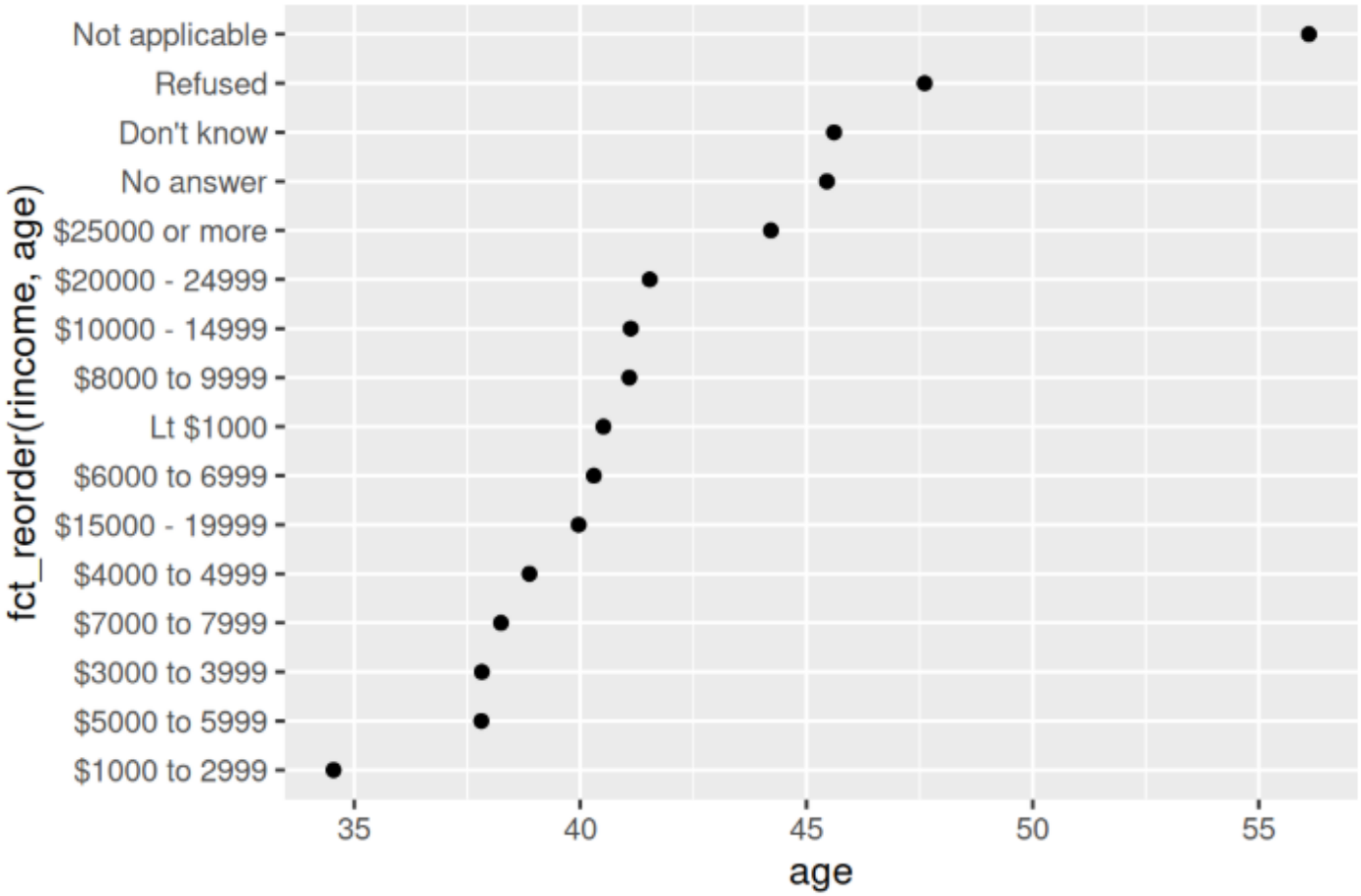
```
csv <- "
month,value
Jan,12
Feb,56
Mar,12"

df <- read_csv(csv, col_types = cols(month = col_factor(month_levels)))
df$month
#> [1] Jan Feb Mar
#> Levels: Jan Feb Mar Apr May Jun Jul Aug Sep Oct Nov Dec
```

Useful functions

- `fct_reorder()`: Reorder levels based on another variable.
- `fct_relevel()`: Moves specified levels to the front.

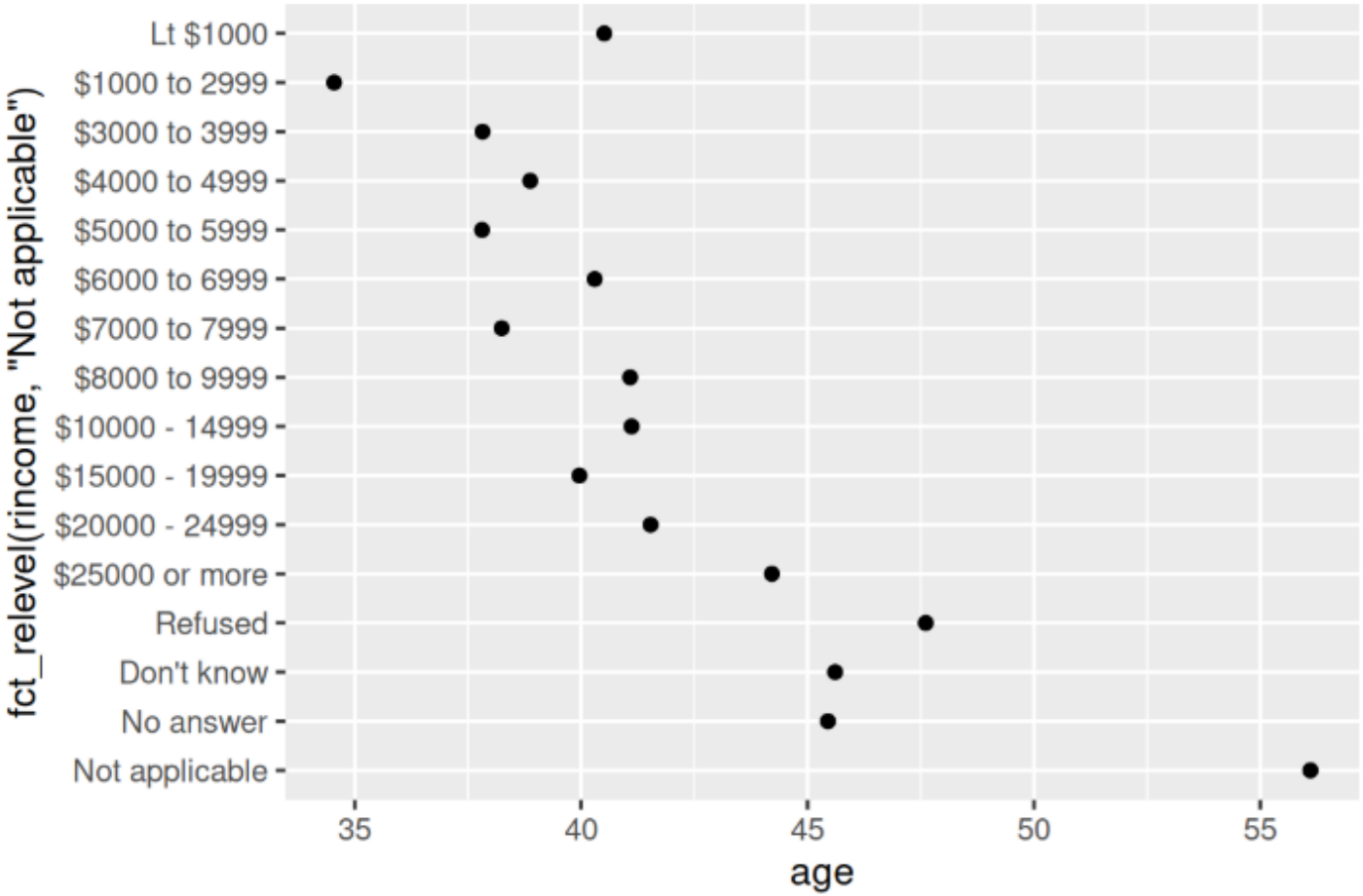
Useful functions

```
ggplot(rincome_summary, aes(x = age, y = fct_reorder(rincome, age))) +
  geom_point()
```



Useful functions

```
ggplot(rincome_summary, aes(x = age, y = fct_relevel(rincome, "Not applicable"))) +
  geom_point()
```



More Useful functions

- `fct_infreq()`: Orders levels in decreasing order of frequency.
- `fct_rev()`: Reverses the order of levels.
- `fct_recode()`: Changes the names of levels.
- `fct_collapse()`: Combines multiple levels into a single level.
- `fact_lump_*()`: Combines levels into "other" based on frequency or count.

More Useful functions

```
gss_cat |>
  mutate(
    partyid = fct_recode(partyid,
      "Republican, strong"    = "Strong republican",
      "Republican, weak"      = "Not str republican",
      "Independent, near rep" = "Ind,near rep",
      "Independent, near dem" = "Ind,near dem",
      "Democrat, weak"        = "Not str democrat",
      "Democrat, strong"      = "Strong democrat"
    )
  ) |>
  count(partyid)
#> # A tibble: 10 × 2
#>   partyid                  n
#>   <fct>                <int>
#> 1 No answer              154
#> 2 Don't know               1
#> 3 Other party            393
#> 4 Republican, strong    2314
#> 5 Republican, weak      3032
#> 6 Independent, near rep 1791
```

More Useful functions

```
gss_cat |>
  mutate(
    partyid = fct_recode(partyid,
      "Republican, strong"    = "Strong republican",
      "Republican, weak"      = "Not str republican",
      "Independent, near rep" = "Ind,near rep",
      "Independent, near dem" = "Ind,near dem",
      "Democrat, weak"        = "Not str democrat",
      "Democrat, strong"      = "Strong democrat",
      "Other"                 = "No answer",
      "Other"                 = "Don't know",
      "Other"                 = "Other party"
    )
  )
```

More Useful functions

```
gss_cat |>
  mutate(
    partyid = fct_collapse(partyid,
      "other" = c("No answer", "Don't know", "Other party"),
      "rep" = c("Strong republican", "Not str republican"),
      "ind" = c("Ind,near rep", "Independent", "Ind,near dem"),
      "dem" = c("Not str democrat", "Strong democrat")
    )
  ) |>
  count(partyid)
#> # A tibble: 4 × 2
#>   partyid     n
#>   <fct>   <int>
#> 1 other     548
#> 2 rep      5346
#> 3 ind      8409
#> 4 dem      7180
```

More Useful functions

```
gss_cat |>
  mutate(relig = fct_lump_lowfreq(relig)) |>
  count(relig)
#> # A tibble: 2 × 2
#>   relig          n
#>   <fct>      <int>
#> 1 Protestant 10846
#> 2 Other      10637
```

```
gss_cat |>
  mutate(relig = fct_lump_n(relig, n = 10)) |>
  count(relig, sort = TRUE)
#> # A tibble: 10 × 2
#>   relig          n
#>   <fct>      <int>
#> 1 Protestant 10846
#> 2 Catholic    5124
#> 3 None        3523
#> 4 Christian    689
#> 5 Other        458
#> 6 Jewish       388
```