R4DS

Cohort 4

Wed 6:00 – 7:00 US Central

Twitter: @Rspjut

5-MINUTE ICE BREAKER

Have you seen these R memes?

AGENDA

- 5-Minute Ice breaker
- Quick Housekeeping Reminders
- R-exercises.com
- Chapter 15 – Factors
- Next Week
- Getting Help

QUICK HOUSEKEEPING REMINDERS

- Video camera is optional, but encouraged.
- I purposely err on the side of going fast. Slowing me down does not hurt my feelings.
- Take time to learn the theory (Grammar of Graphics, Tidy Data whitepaper, Relational Database theory, Appropriate Visualization Types, Wrangling Categorical Data in R, etc.).
- Please do the chapter exercises. Second-best learning opportunity!
- Please plan on teaching one of the lessons. Best learning opportunity!

R-EXERCISES.COM

Rexercises

S

Start here to learn R! | Consulting | Learn R like a pro | R Courses | Why exercise? | About

You are here: Home / Start here to learn R!

Start here to learn R!

0 0

Ready, set, go!

On R-exercises, you will find *more than 4,000* R exercises. We've bundled them into exercise *sets*, where each set covers a specific concept or function. An exercise set typically contains about 10 exercises, progressing from easy to somewhat more difficult. In order to give you a full picture of all the amazing content on this website, we've categorized all sets into broader topics below.

EXAMPLES

In R, factors are used to work with categorical variables, variables that have a ***fixed and known set of possible values****.*

-Wickham and Grolemund, Chapter 15

Months
Dec
Apr
Jan
Mar

As a Group of Strings (Not a Factor)

1) R takes no action to control for typos or invalid entries

Months
Dec
Apr
Jam
Mar

2) Sorting is alphabetical instead of in a common order

Months
Apr
Dec
Jan
Mar

FACTORS ASSIGN LEVELS

Before looking at what the data presents, create the factor's LEVELS.

The factor's LEVELs are the valid and ordered set of values the variable can take.

"I know there are twelve months, and I know what valid abbreviations I need, and what order they go in."

Create an object that holds the levels.

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

FACTORS ASSIGN LEVELS

Now you can apply the factor to your data.

```
sample_months <- c("Dec", "Apr", "Jan", "Mar", "Jan")
factored_sample_months <- factor(sample_months, levels = month_levels)
```

Output includes the LEVELs of the factor

```
typo_months <- c(“Dec”, “Apr”, “Jam”, “Mar”)
factored_typo_month <- factor(typo_month, levels = month_levels)
```

Errors converted to NA

DEMO IN R

GENERAL SOCIAL SURVEY

?gss_cat

Year	Age	Marital	Race	Rincome	Partyid	Relig	Tvhours
Year of survey 2000 - 2014	Survey participant age	Survey participant marital status	Survey participant race	Survey participant reported income	Survey participant political party affiliation	Survey participant religious denomination	Survey participant hours per day watching TV

	year	marital	age	race	rincome	partyid	relig	denom	tvhours
1	2000	Never married	26	White	$8000 to 9999	Ind,near rep	Protestant	Southern baptist	12
2	2000	Divorced	48	White	$8000 to 9999	Not str republican	Protestant	Baptist-dk which	NA
3	2000	Widowed	67	White	Not applicable	Independent	Protestant	No denomination	2
4	2000	Never married	39	White	Not applicable	Ind,near rep	Orthodox-christian	Not applicable	4
5	2000	Divorced	25	White	Not applicable	Not str democrat	None	Not applicable	1
6	2000	Married	25	White	$20000 - 24999	Strong democrat	Protestant	Southern baptist	NA

GENERAL SOCIAL SURVEY

?gss_cat %>% str()

```
> gss_cat %>% str()
tibble [21,483 x 9] (S3: tbl_df/tbl/data.frame)
 $ year   : int [1:21483] 2000 2000 2000 2000 2000 2000 2000 2000 2000 2000 ...
 $ marital: Factor w/ 6 levels "No answer","Never married",..: 2 4 5 2 4 6 2 4 6 6 ...
 $ age    : int [1:21483] 26 48 67 39 25 25 36 44 44 47 ...
 $ race   : Factor w/ 4 levels "Other","Black",..: 3 3 3 3 3 3 3 3 3 3 ...
 $ rincome: Factor w/ 16 levels "No answer","Don't know",..: 8 8 16 16 16 5 4 9 4 4 ...
 $ partyid: Factor w/ 10 levels "No answer","Don't know",..: 6 5 7 6 9 10 5 8 9 4 ...
 $ relig  : Factor w/ 16 levels "No answer","Don't know",..: 15 15 15 6 12 15 5 15 15 15 ...
 $ denom  : Factor w/ 30 levels "No answer","Don't know",..: 25 23 3 30 30 25 30 15 4 25 ...
 $ tvhours: int [1:21483] 12 NA 2 4 1 NA 3 NA 0 3 ...
```

A number of variables are already stored as factors, so we can inspect the levels

gss_cat$race %>% levels()

```
> gss_cat$race %>% levels()
[1] "Other"          "Black"          "White"          "Not applicable"
```

Stored as 1 | Stored as 2 | Stored as 3 | Stored as 4

GENERAL SOCIAL SURVEY: REORDERING FACTORS

DEMO #2 IN R

- **relig_summary data, note factor order**
- **In ggplot data not in order, follows factor order**
- **Use fct_reorder to fix**

GENERAL SOCIAL SURVEY: MODIFYING FACTORS

DEMO #3 IN R

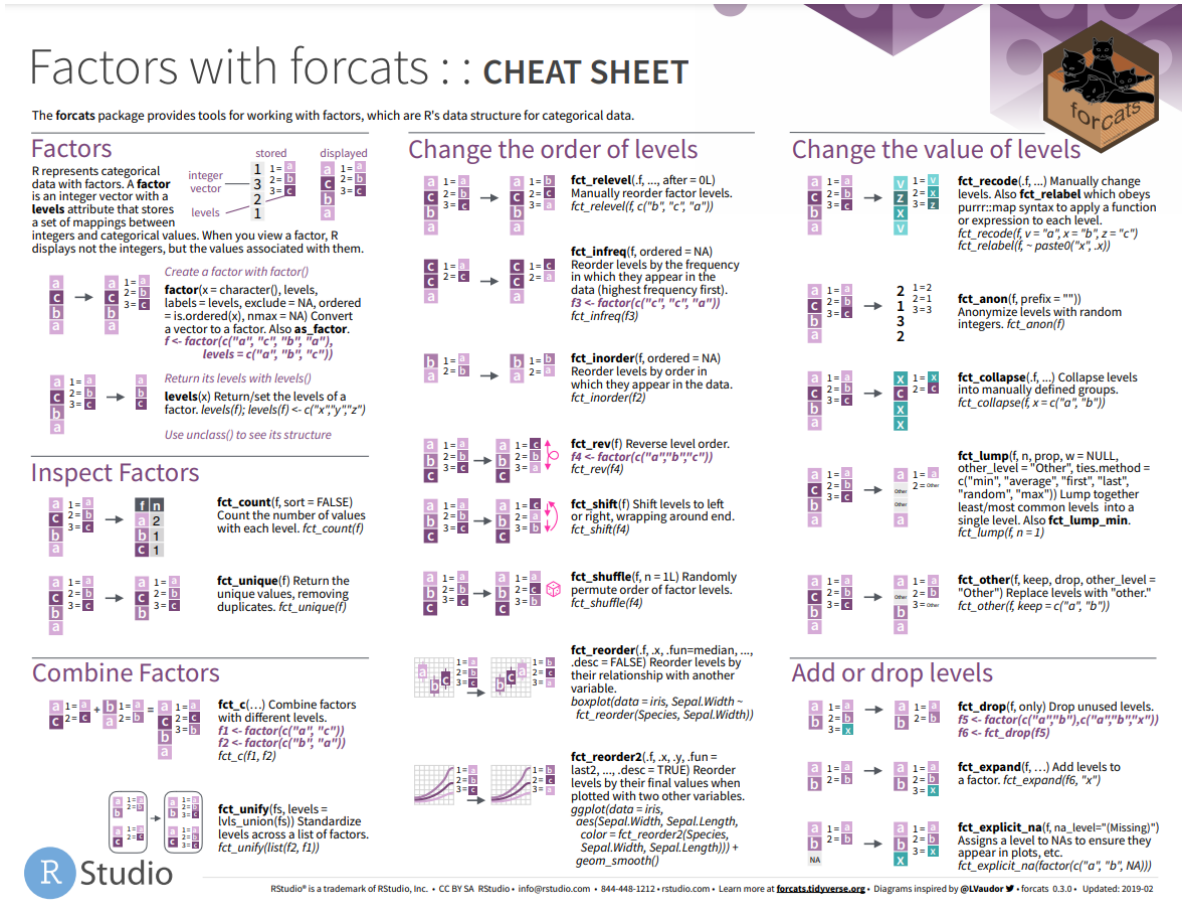
- **Gss_cat count**
- **Names are terse**
- **Use recode to designate new categories**

REORDERING AND MODIFYING PERMANENTLY

DEMO #4 IN R

- **Permanently change levels**

FORCAST CHEAT SHEET



Factors with forcats : : **CHEAT SHEET**

The **forcats** package provides tools for working with factors, which are R's data structure for categorical data.

Factors

R represents categorical data with factors. A **factor** is an integer vector with a **levels** attribute that stores a set of mappings between integers and categorical values. When you view a factor, R displays not the integers, but the values associated with them.

integer vector — levels — stored — displayed

Create a factor with factor()

factor(x = character(), levels, labels = levels, exclude = NA, ordered = is.ordered(x), nmax = NA) Convert a vector to a factor. Also **as_factor**.
f <- factor(c("a", "c", "b", "a"), levels = c("a", "b", "c"))

Return its levels with levels()

levels(x) Return/set the levels of a factor. *levels(f); levels(f) <- c("x","y","z")*

Use unclass() to see its structure

Inspect Factors

fct_count(f, sort = FALSE) Count the number of values with each level. *fct_count(f)*

fct_unique(f) Return the unique values, removing duplicates. *fct_unique(f)*

Combine Factors

fct_c(…) Combine factors with different levels.
f1 <- factor(c("a", "c"))
f2 <- factor(c("b", "a"))
fct_c(f1, f2)

fct_unify(fs, levels = lvls_union(fs)) Standardize levels across a list of factors.
fct_unify(list(f2, f1))

Change the order of levels

fct_relevel(.f, ..., after = 0L) Manually reorder factor levels.
fct_relevel(f, c("b", "c", "a"))

fct_infreq(f, ordered = NA) Reorder levels by the frequency in which they appear in the data (highest frequency first).
f3 <- factor(c("c", "c", "a"))
fct_infreq(f3)

fct_inorder(f, ordered = NA) Reorder levels by order in which they appear in the data.
fct_inorder(f2)

fct_rev(f) Reverse level order.
f4 <- factor(c("a","b","c"))
fct_rev(f4)

fct_shift(f) Shift levels to left or right, wrapping around end.
fct_shift(f4)

fct_shuffle(f, n = 1L) Randomly permute order of factor levels.
fct_shuffle(f4)

fct_reorder(.f, .x, .fun=median, ..., .desc = FALSE) Reorder levels by their relationship with another variable.
boxplot(data = iris, Sepal.Width ~ fct_reorder(Species, Sepal.Width))

fct_reorder2(.f, .x, .y, .fun = last2, ..., .desc = TRUE) Reorder levels by their final values when plotted with two other variables.
ggplot(data = iris, aes(Sepal.Width, Sepal.Length, color = fct_reorder2(Species, Sepal.Width, Sepal.Length))) + geom_smooth()

Change the value of levels

fct_recode(.f, …) Manually change levels. Also **fct_relabel** which obeys purrr::map syntax to apply a function or expression to each level.
fct_recode(f, v = "a", x = "b", z = "c")
fct_relabel(f, ~ paste0("x", .x))

fct_anon(f, prefix = "")) Anonymize levels with random integers. *fct_anon(f)*

fct_collapse(.f, …) Collapse levels into manually defined groups.
fct_collapse(f, x = c("a", "b"))

fct_lump(f, n, prop, w = NULL, other_level = "Other", ties.method = c("min", "average", "first", "last", "random", "max")) Lump together least/most common levels into a single level. Also **fct_lump_min**.
fct_lump(f, n = 1)

fct_other(f, keep, drop, other_level = "Other") Replace levels with "other."
fct_other(f, keep = c("a", "b"))

Add or drop levels

fct_drop(f, only) Drop unused levels.
f5 <- factor(c("a","b"),c("a","b","x"))
f6 <- fct_drop(f5)

fct_expand(f, …) Add levels to a factor. *fct_expand(f6, "x")*

fct_explicit_na(f, na_level="(Missing)") Assigns a level to NAs to ensure they appear in plots, etc.
fct_explicit_na(factor(c("a", "b", NA)))

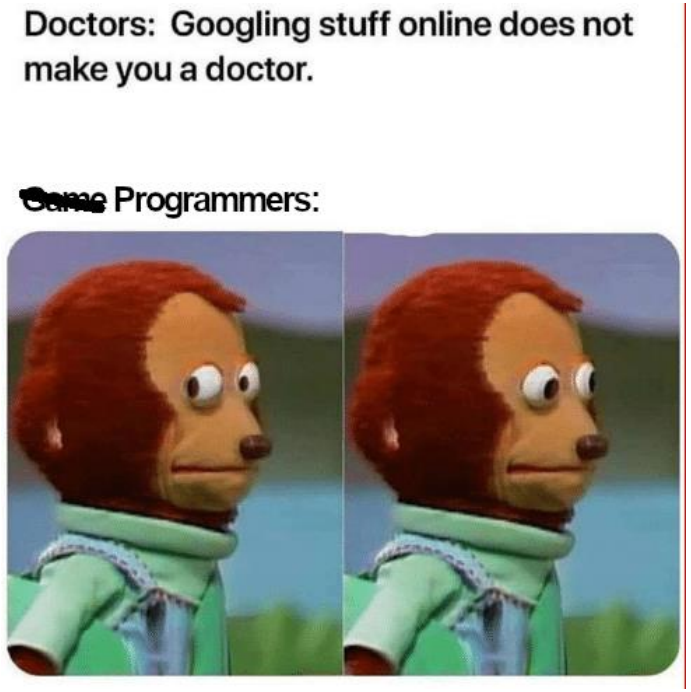
RStudio

NEXT WEEK…

- Chapter 16 – Dates and Times by Maria!

GETTING HELP

- Ask questions during our call
- Google
- Stack Overflow
- Slack
- Office Hours r4ds.io/calendar
- Twitter #rstats
- r4ds answer keys: Jeff Arnold (preferred) or Bryan Shalloway (also good)
- Cheatsheets




RStudio