

Final Project

Navigate to 06_final project

Open variation.png

Open hirshey.Rmd

Knit the document, and review the output

Final Project Template

Navigate to 06_final project

Open 06_final_template.Rmd

Final Project Assignment

3 Documents

1. last_name.Rmd with full code
2. Knitted report
3. image.png

Final Presentation

2-3 minute oral presentation:

- Review the data
- Output file
- Salient observations (conclusion)
- Strategy to test the hypothesis

Final Presentation Feedback

You will receive feedback on your oral presentation based on:

- Content: Comments on the data and insight that were presented
- Visuals: Any ways the visuals could have been more clear?
- Mechanics of presentation: Comments on the student presentation
- Overall

Final Grades

As described in the syllabus,

“A final grade is an important metric summarizing your fluency and aptitude with the concepts presented in class. In this course, I will ask each student to be the judge of these criteria and submit their proposed grade and a justification for it after the conclusion of the class. The following metrics should be considered when proposing a grade:

40% Code, R markdown file, and summary interpretation of findings

30% In-class presentation of above findings

20% Help vignette

5% Attendance

5% Participation

While the instructors will ultimately determine the final grade for each student, thoughtful reflection and justification for a letter grade provided by each student will be heavily weighted. Self-critique is an important skill for scientists, and this class presents and opportunity for it.”

Concluding Thoughts

Data science enables **data-driven** information gathering and hypothesis generation

→ Scientific Research

→ Reviews

Data science enables the ability to ask new types of questions

Process centric, not necessarily gene centric

Making things computable makes them actionable at zero marginal cost.

Workflows save time, achieve reproducibility

Resources: Cheat Sheets

Cheat sheets make it easy to learn about and quickly refer to function in some of the common packages.

Data Import :: CHEAT SHEET

R's tidyverse is built around `tidy` stored in `tibbles`, which are enhanced data frames.

The front side of this sheet shows how to read text files into R with `readr`.

The reverse side shows how to create tibbles with `tibble` and to layout tidy data with `tidyR`.

OTHER TYPES OF DATA

Try one of the following packages to import other types of files

- `haven` - SPSS, Stata, and SAS files
- `readxl` - excel files (.xls and .xlsx)
- `DBI` - databases
- `jsonlite`
- `xml2` - XML
- `httr` - Web APIs
- `rvest` - HTML (Web Scraping)

Save Data

Save `x`, an object, to `path`, a file path, as:

Comma delimited file
`write_csv(x, path, na = "NA", append = FALSE, col_names = lapply)`

File with a single delimiter
`write_delim(x, file, delim = " ", na = "NA", append = FALSE, col.names = lapply)`

CSV for excel
`write_excel(x, path, na = "NA", append = FALSE, col.names = lapply)`

String for excel
`write_xlsx(x, path, append = FALSE)`

String vector to file, one element per line
`write_lines(path, na = "NA", append = FALSE)`

String to file
`write_file(path, text)`

String to file, path, append = FALSE)
`write_file(path, text, append = FALSE)`

String to file, path, compression = c("none", "gz", "bz2", "xz", ...)
`write_rds(x, path, compress = c("none", "gz", "bz2", "xz", ...))`

Tab delimited files
`write_tsv(x, path, na = "NA", append = FALSE, col.names = lapply)`

Read Tabular Data

- These functions share the common arguments:

`read_*` (file, col_names = TRUE, col_types = NULL, locale = `default_locale()`, na = c("", "NA"), quoted = TRUE, comment = "#", trim_ws = TRUE, skip = 0, n_max = Inf, guess_max = min(1000, n_max), progress = interactive())

Common Delimited Files

```
read_csv("file.csv")
# To make file.csv from
write_file("file.csv", file = "a,b,c\n1,2,3\n4,5,NA")
```

Semi-colon Delimited Files

```
read_csv2("file2.csv")
# write_file("file2.csv", file = "a;b;c\n1;2;3\n4;5;NA")
```

Files with Any Delimiter

```
read_delim("file.txt", delim = "|")
# write_file("file.txt", file = "a|b|c\n1|2|3\n4|5|NA")
```

Fixed Width Files

```
read_fwf("file.fwf", col_positions = c(1, 3, 5))
# write_file("file.fwf", file = "a   b   c\n1   2   3\n4   5   NA")
```

Tab Delimited Files

```
read_tsv("file.tsv") # Also read_table(), read_table2()
# write_file("file.tsv", file = "a\tb\tc\n1\t2\t3\n4\t5\tNA")
```

USEFUL ARGUMENTS

Example file	1 2 3	4 5 NA	skip lines
write_file("file1,n1,n2,n3,n4,n5,NA","file.csv") f <- "file.csv"	1 2 3 4 5 NA		read_csv(f, skip = 1)
No header	1 2 3 4 5 NA		read_csv(f, n_max = 1)
Provide header	A B C 1 2 3 4 5 NA		read_csv(f, col.names = c("x", "y", "z"))
	A B C 1 2 3 4 5 NA		read_csv(f, col.names = c("1", "2", "3"))
	A B C 1 2 3 4 5 NA		read_csv(f, skip = 1, n_max = 1)
	A B C 1 2 3 4 5 NA		read_csv(f, n_max = 1)

Read in a subset

Missing Values

Read Non-Tabular Data

Read a file into a single string
`read_file(file, locale = default_locale())`

Read each line into its own string
`read_lines(file, skip = 0, n_max = -1, na = character(), locale = default_locale(), progress = interactive())`

Read Apache style log file
`read_logfile(col_names = FALSE, col_types = NULL, skip = 0, n_max = -1, progress = interactive())`

Read a file into a raw vector
`read_file(file)`

Read each line into a raw vector
`read_lines_raw(file, skip = 0, n_max = -1, na = character(), locale = default_locale(), progress = interactive())`

RSStudio® is a trademark of RStudio, Inc. | CC BY SA RStudio® - info@rstudio.com | 844-448-1221 | rstudio.com | Learn more at [rstudio.com](#) | Read: 1.1.0 - tibble 1.1.2 - tidy 0.6.6 - Updated: 2017-01-01

Data Transformation with dplyr :: CHEAT SHEET

dplyr functions work with pipes and expect **tidy data**. In tidy data:

- Each variable is in its own column
- Each observation, or case, is in its own row
-  $x \rightarrow y$ becomes $f(x, y)$

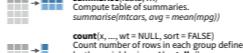
Summarise Cases

These apply **summary functions** to columns to create a new table of summary statistics. Summary functions take vectors as input and return one value (see back).



summary function

`summarise(data, ...)`
Compute table of summaries.
`summarise(niris, avg = mean(mpg))`



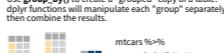
`count(x, ..., wt = NULL, sort = FALSE)`
Count number of entries per group defined by the variables in ... also tally.
`count(niris, Species)`

VARIATIONS

`summarise_all` - Apply funs to every column.
`summarise_at` - Apply funs to specific columns.
`summarise_if` - Apply funs to all cols of one type.

Group Cases

Use `group_by` to create a "grouped" copy of a table. dplyr functions will manipulate each "group" separately and then combine the results.



`mtcars %>%
group_by(cyl) %>%
summarise(avg = mean(mpg))`

`group_by(data, ..., add = TRUE)`
Returns copy of table grouped by ...
`g_iris <- group_by(iris, Species)`



`ungroup(...)`
Returns ungrouped copy of table.
`ungroup(g_iris)`

Manipulate Cases

EXTRACT CASES

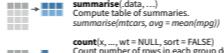
Row functions return a subset of rows as a new table.



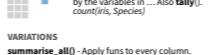
`filter(data, ...)`
Extract rows that meet logical criteria.
`filter(iris, Sepal.Length > 7)`



`distinct(data, ..., keep = FALSE)`
Remove rows with duplicate values



`sample_frac(data, size = 1, replace = FALSE,
weight = NULL, env = parent.frame())`
Randomly select fraction of rows



`sample_n(data, n, replace = TRUE)`
Randomly select n rows.
`sample_n(iris, 10, replace = TRUE)`



`slice(data, ...)`
Select rows by position.
`slice(iris, 10:15)`

`top_n(data, n, wt)` Select and order top n entries (by group if grouped data).
`top_n(iris, 5, Sepal.Width)`

Logical and boolean operators to use with filter()

< <= == != %in% %in% ! | &

See `?base::logic` and `?Comparison` for help.

ARRANGE CASES



`arrange(data, ...)`
Order rows by values of a column or columns (low to high), use with `desc()` to reverse order.
`arrange(mtcars, mpg)`



`arrange_(data, ..., .by_group = TRUE)`
Order rows by values of a column or columns (low to high).
`arrange_(mtcars, desc(mpg))`

ADD CASES



`add_row(data, ..., before = NULL, after = NULL)`
Add one or more rows to a table.
`add_row(toothful, eruptions ~ 1, waiting = 1)`

Manipulate Variables

EXTRACT VARIABLES

Column functions return a set of columns as a new vector or table.



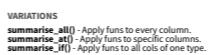
`pull(data, var = -1)`
Extract column values as a vector. By name or index.
`pull(mtcars, Sepal.Length)`



`select(data, ...)`
Extract columns as a table. Also `select_(If)`.
`select(iris, Sepal.Length, Species)`

MAKE NEW VARIABLES

These apply **vectorized** functions to columns. Vectorized funs take vectors as input and return vectors of the same length as output (see back).



`mutate(data, ...)`
Compute new column(s).
`mutate(mtcars, gpm = 1/mpg)`



`transmute(data, ...)`
Compute new column(s), drop others.
`transmute(mtcars, gpm = 1/mpg)`



`mutate_all(tbl, funs, ...)`
Apply funs to every column. Use with `funs(...)`. Also `mutate_(If)`,
`mutate_all(fifthtbl, funs(log(.)))`



`mutate_at(tbl, vars, funs, ...)`
Apply funs to the helper functions for `select(...)`.
`mutate_at(iris, vars ~ Species, funs(log(.)))`



`add_column(data, ..., before = NULL, add = TRUE)`
Add new column(s). Also `add_count_(If)`,
`add_cally_(If)`, `add_column(mtcars, newvar = 1:32)`



`rename(data, ...)`
Rename columns.
`rename(iris, Length ~ Sepal.Length)`

R Markdown Cheat Sheet

learn more at rmarkdown.rstudio.com

 R Studio

1. Workflow R Markdown is a format for writing reproducible, dynamic reports with R. Use it to embed R code and results into slideshows, pdfs, html documents, Word files and more. To make a report:

- i. Open - Open a file that uses the .Rmd extension.
- ii. Write - Write content with the easy to use R Markdown syntax
- iii. Embed - Embed R code that creates output to include in the report
- iv. Render - Replace R code with its output and transform the report into a slideshow, pdf, html or ms Word file.

2. Open File Start by saving a text file with the extension .Rmd, or open an RStudio Rmd template

In the menu bar, click **File > New File > R Markdown...**. A window will open. Select the class of output you would like to make with your .Rmd file.

Select the specific type of output to make with the radio buttons (you can change this later)

Click OK.

3. Markdown Next, write your report in plain text. Use markdown syntax to describe how to format text in the final report.

syntax

```
Plain text
End a line with two spaces to start a new paragraph.
**italic** and _italic_
**bold** and __bold__
superscript + strikethrough
[[link]](www.rstudio.com)

# Header 1
## Header 2
### Header 3
#### Header 4
##### Header 5
##### Header 6

endash: ---
endash: ---
ellipsize: ...
image: ...
image-relation: EA = lgp+r^235
image: [[path/to/smallor*.png]]

horizontal rule (or slide break):
***
```

becomes

```
Pain text
End a line with two spaces to start a new paragraph.
italics and italics
bolds and bolds
superscript and strikethrough
link: www.rstudio.com

Header 1
Header 2
Header 3
```

Header 4

Header 5

Header 6

endash: ---

endash: ---

ellipsize: ...

image: ...

image-relation: EA = $lgp + r^{235}$

image: [[path/to/smallor*.png]]

horizontal rule (or slide break):

block quote

unordered list

- * item 2
 - + sub-item 1
 - + sub-item 2

1. ordered list

1. item 2
 1. sub-item 1
 1. sub-item 2

Table Header | **Second Header**

Table Cell	Second Header
Cell 1	Cell 2
Cell 3	Cell 4

Table Header | **Second Header**

Table Cell	Second Header
Cell 1	Cell 2
Cell 3	Cell 4

4. Choose Output Write a YAML header that explains what type of document to build from your R Markdown file.

The RStudio template writes the YAML header for you

A YAML header is a set of key: value pairs at the start of your file. Begin and end the header with a line of three dashes (---)

The output value determines which type of file R will build from your .Rmd file (in Step 6)

output: html_document	html file (web page)
output: pdf_document	pdf document
output: word_document	Microsoft Word .docx
output: beamer_presentation	beamer slideshow (pdf)
output: ioslides_presentation	ioslides slideshow (html)

RStudio is a trademark of RStudio, Inc. | www.RStudio.com | 844-448-3212 | support@rstudio.com

Resources: Stackoverflow

stack overflow Products Customers Use cases Log in Sign up

Home PUBLIC Stack Overflow Tags Users Jobs TEAMS What's this? Join Private Q&A

Search Results

Results for removing legend title ggplot

removing legend title ggplot

20 results

Q: remove legend title in ggplot
99 votes 3 answers
I'm trying to **remove the title of a legend** in ggplot2: `df <- data frame(g = rep(letters[1:2], 5), x = rnorm(10), y = rnorm(10)) library(ggplot2) ggplot(df, aes(x, y, colour=g ...`
asked Feb 8 '13 by [smiling](#)

Q: How to change the position of plotly figures using ggplotly
0 votes 1 answer
I am trying to **remove the legend title** using ggplotly without success. I'm sure there is an easy fix, but I cannot find the documentation for it - and **removing the legend title** (or changing the ...) + `geom_boxplot() + theme(legend.title=element_blank())` a # No Legend Title # plotly puts back the **legend title** ggplotly(a) Any ideas how to change / **remove the title** of the graph? Should it be done using ggplotly or ggplot? ...
asked Dec 26 '15 by [Nick](#)

Q: Remove legend in ggplot in Python
1 vote 0 answers
The **title** says it all: Is there any way to **remove the legend** in ggplot in Python? I tried to google but could only find the solution for R ...
asked Jul 25 '15 by [Giang Do](#)

Q: ggplot legend / label change with various guides
0 votes
already used to **remove** the titles and also **remove legend** for a 3rd aes, but I do not manage to do it. Do you have a solution? Here is my code : `p <- ggplot(data, aes(x = Nb)) p + geom_ribbon(aes(ymin ...` I am trying to

Siemens Siemens AG 9 Singapore Electronics Public 10k+ people
Senior Data Scientist machine-learning sql
HTML5 Code King html5 css3
Business Architect enterprise

[View all 14 job openings!](#)

Hot Network Questions

Resources

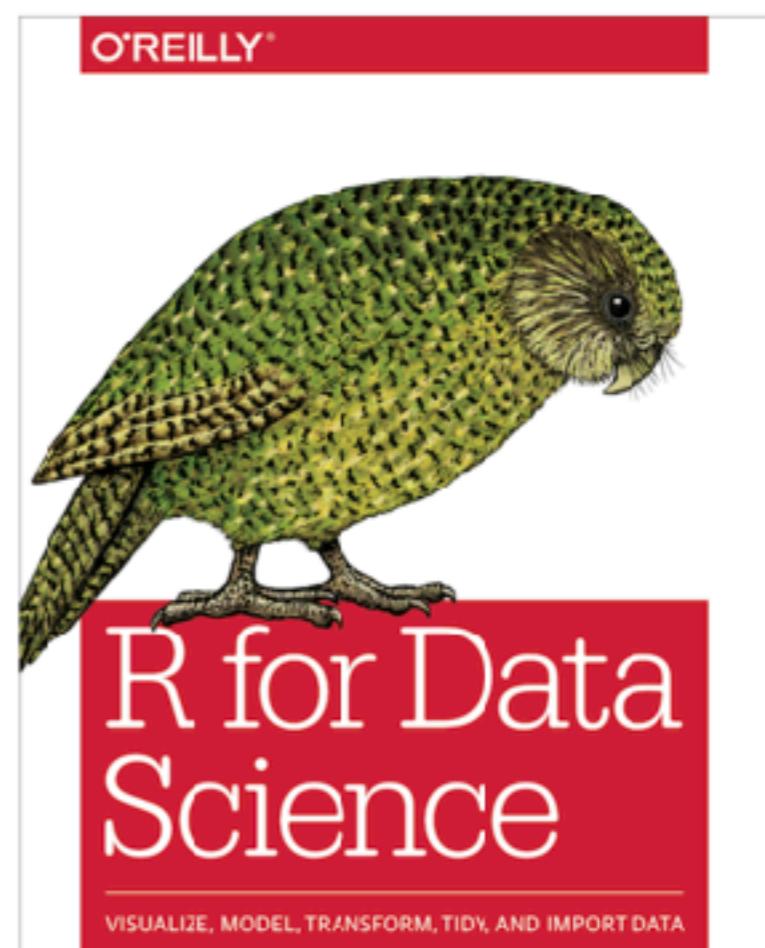
R for Data Science

Garrett Grolemund

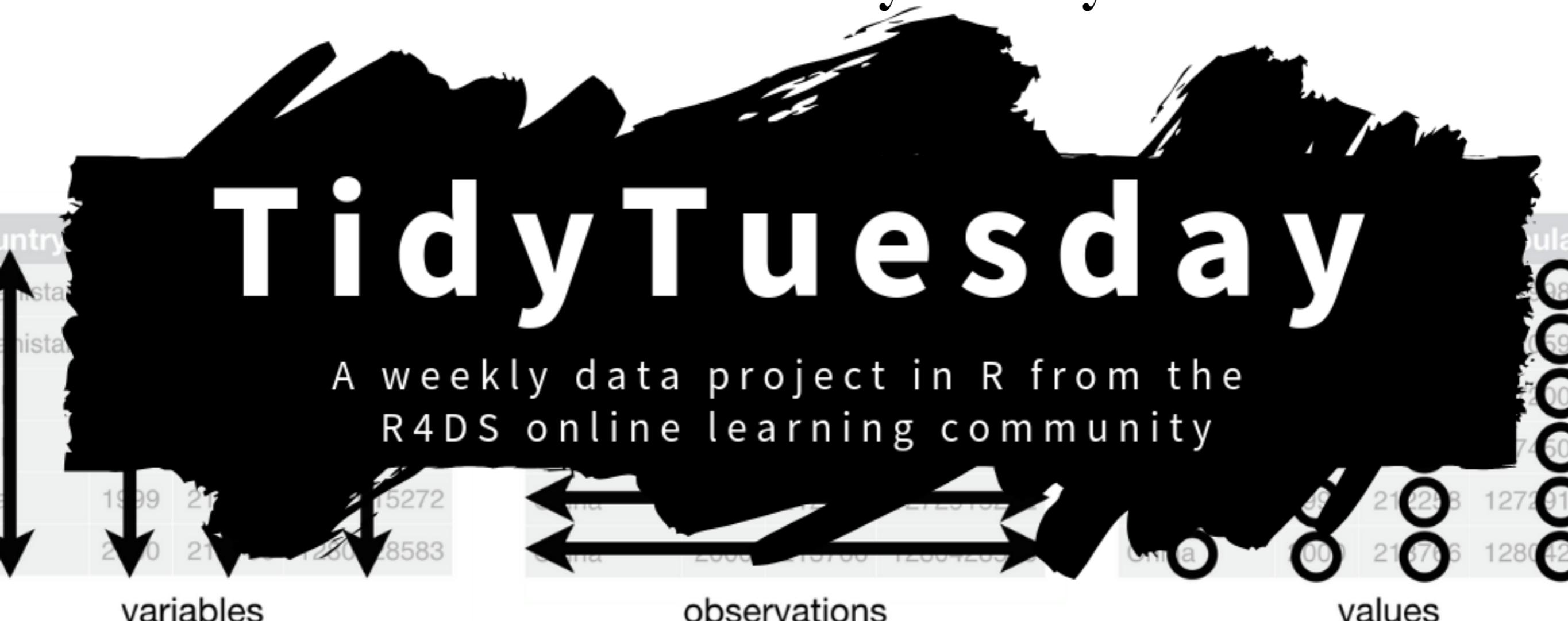
Hadley Wickham

Welcome

This is the website for “**R for Data Science**”. This book will teach you how to do data science with R: You’ll learn how to get your data into R, get it into the most useful structure, transform it, visualise it and model it. In this book, you will find a practicum of skills for data science. Just as a chemist learns how to clean test tubes and stock a lab, you’ll learn how to clean data and draw plots – and many other things besides. These are the skills that allow data science to happen, and here you will find the best practices for doing each of these things with R. You’ll learn how to use the grammar of graphics, literate programming, and reproducible research to save time. You’ll also learn how to manage cognitive resources to facilitate discoveries when wrangling, visualising, and exploring data.



Resources: #tidytuesday



rfordatascience / [tidytuesday](#)

Unwatch 258

Unstar 960

Fork 457

Code

Issues 48

Pull requests 0

Projects 0

Wiki

Security

Insights

Official repo for the #tidytuesday project

380 commits

2 branches

11 releases

8 contributors

MIT

Branch: master ▾

New pull request

Create new file

Upload files

Find File

Clone or download ▾

Acknowledgements

Teaching Assistants

Allie Mills, Ph.D.

Akshay Bareja, D.Phil.

Inspiration, ideas, packages, code

#R4DS (Garrett Grolemund and Hadley Wickham)

Mine Çetinkaya-Rundel (datasciencebox.org)

Chester Ismay and Albert Y. Kim (Modern Dive)

Garrett Grolemund (Remastering the Tidyverse)

Tidyverse devs and community

Rstudio

 @matthewhirsche

 www.hirscheylab.org

 github.com/hirscheylab