Making the jump from learning to applying:

R training and documentation for different levels of expertise

Kate Hertweck

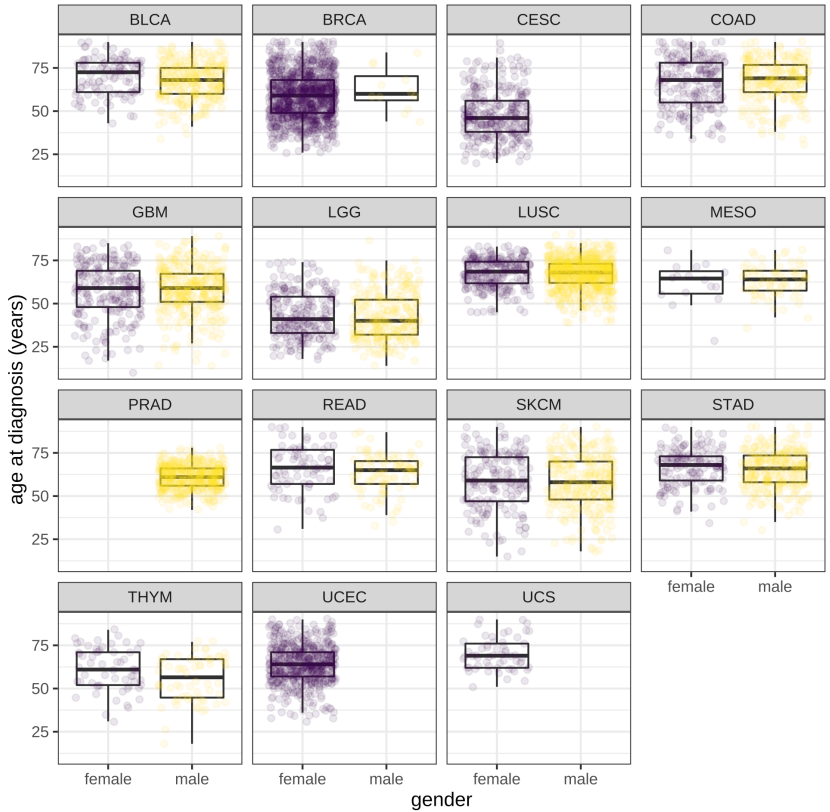
 @k8hert

```r
clinical <- clinical %>%
  select(primary_diagnosis, tumor_stage, age_at_diagnosis,
         vital_status, gender, disease) %>%
  filter(vital_status != "not reported", !is.na(vital_status)) %>%
  filter(!is.na(age_at_diagnosis)) %>%
  mutate(yrs_at_diagnosis = floor(age_at_diagnosis/365))

ggplot(data = clinical) +
  geom_boxplot(aes(x = gender, y = yrs_at_diagnosis),
               outlier.shape = NA) +
  geom_jitter(aes(x = gender, y = yrs_at_diagnosis,
                  color = gender), alpha = 0.1) +
  facet_wrap(vars(disease)) +
  ylab("age at diagnosis (years)") +
  scale_color_viridis_d() +
  theme_bw() +
  theme(legend.position = "none")
```

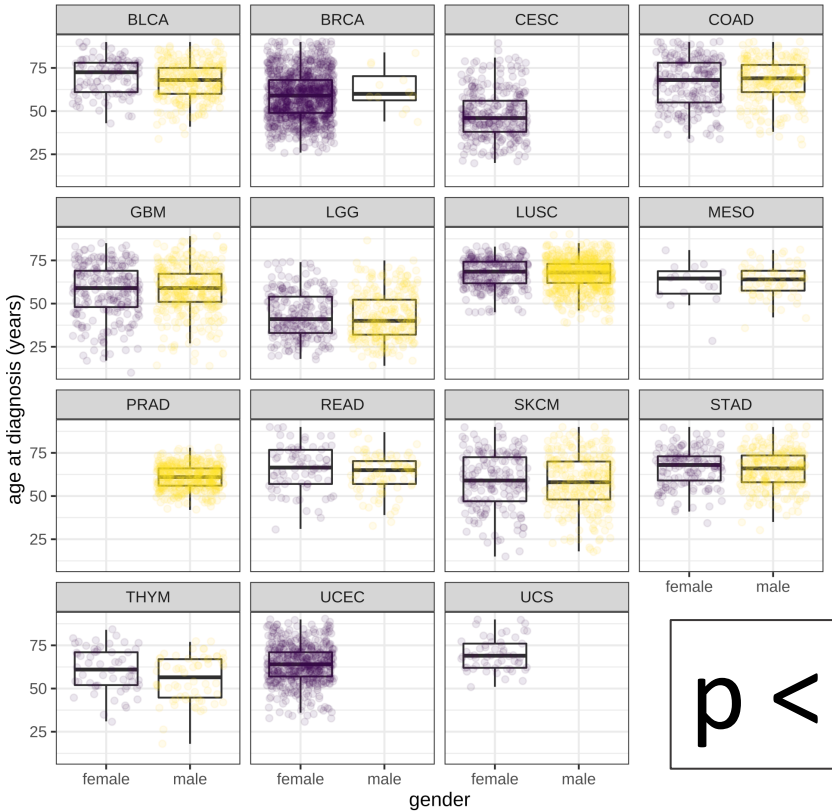
```
clinical <- clinical %>%
  select(primary_diagnosis, tumor_stage, age_at_diagnosis,
         vital_status, g
  filter(vital_status !=
  filter(!is.na(age_at_d
  mutate(yrs_at_diagnosi

ggplot(data = clinical)
  geom_boxplot(aes(x = g
                   outlier.s
  geom_jitter(aes(x = ge
                  color
  facet_wrap(vars(diseas
  ylab("age at diagnosis
  scale_color_viridis_d(
  theme_bw() +
  theme(legend.position
```



```
clinical <- clinical %>%
  select(primary_diagnosis, tumor_stage, age_at_diagnosis,
         vital_status, g
  filter(vital_status !=
  filter(!is.na(age_at_d
  mutate(yrs_at_diagnosi

ggplot(data = clinical)
  geom_boxplot(aes(x = g
                   outlier.s
  geom_jitter(aes(x = ge
                  color
  facet_wrap(vars(diseas
  ylab("age at diagnosis
  scale_color_viridis_d(
  theme_bw() +
  theme(legend.position
```



$p < 0.05$!!!!


Photo by Alain Audet on Pixabay

```
> data.csv
Error: object 'data.csv' not found
> load(data.csv)
Error in load(data.csv) : object 'data.csv' not found
> read_csv(data.csv)
Error in read_csv(data.csv) : could not find function "read_csv"
> read_csv("data.csv")
Error in read_csv("data.csv") : could not find function "read_csv"
> library(tidyverse)
── Attaching packages ─────────────────────────────────────── tidyverse 1.3.0 ──
✓ ggplot2 3.3.2     ✓ purrr   0.3.4
✓ tibble  3.0.3     ✓ dplyr   1.0.1
✓ tidyr   1.1.1     ✓ stringr 1.4.0
✓ readr   1.3.1     ✓ forcats 0.5.0
── Conflicts ────────────────────────────────────────── tidyverse_conflicts() ──
x dplyr::filter() masks stats::filter()
x dplyr::lag()    masks stats::lag()
> read_csv("data.csv")
Error: 'data.csv' does not exist in current working directory ('/Users/k8hertweck').
> BUT WHY!?!?
```

Photo by Jay Wennington on Unsplash

Photo by Matthew Henry on Unsplash


Novice
Practitioner
Expert

Expert

Photo by Charles Deluvio on Unsplash

Expert

Image from dkahle/ggmap

Practitioner

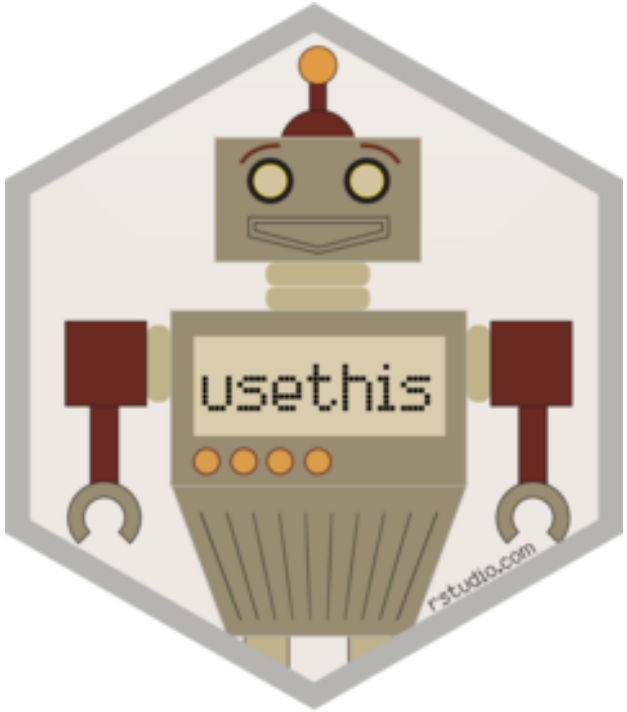


Photo by Victor Grabarczyk on Unsplash

Practitioner

FredHutch / VISCtemplates

Step Two: GitHub

Step Four: Start Analysis Work

Why use a R package structure?

The R package structure is used for analysis projects because it provides an easily recognizable format for file organization and allows for the use of packages like devtools and roxygen2. For more information, see:

1. Ben Marwick, Carl Boettiger, and Lincoln Mullen. Paper: Packaging data analytical work reproducibily using R (and friends)
2. Karthik Ram. Rstudio::conf 2019 talk: How To Make Your Data Analysis Notebooks More Reproducible
3. rOpenSci Community Call: Reproducible Research with R

Example from FredHutch/VISCtemplates

Novice

```
> read_csv("data.csv")
Error: 'data.csv' does not exist in current working directory
> BUT WHY!?!?
```

```
> help(read_csv)
```

Photo by Karsten Winegeart on Unsplash

Novice

Read a delimited file (including csv & tsv) into a tibble

Description

`read_csv()` and `read_tsv()` are special cases of the general `read_delim()`. They're useful for reading the most common types of flat file data, comma separated values and tab separated values, respectively. `read_csv2()` uses `;` for the field separator and `,` for the decimal point. This is common in some European countries.

Usage

```
read_delim(file, delim, quote = "\"", escape_backslash = FALSE,
  escape_double = TRUE, col_names = TRUE, col_types = NULL,
  locale = default_locale(), na = c("", "NA"), quoted_na = TRUE,
  comment = "", trim_ws = FALSE, skip = 0, n_max = Inf,
  guess_max = min(1000, n_max), progress = show_progress(),
  skip_empty_rows = TRUE)
```

Novice

read_delim {readr} R Documentation

Read a deli sv) into a tibble

Description

read_csv() and re d_delim(). They're useful for rea mma separated values and tab separ for the field separator and , for t ean countries.

Usage

```
read_delim(file ash = FALSE,
  escape_double = NULL,
  locale = default_locale(), na = c("", "NA"), quoted_na = TRUE,
  comment = "", trim_ws = FALSE, skip = 0, n_max = Inf,
  guess_max = min(1000, n_max), progress = show_progress(),
  skip_empty_rows = TRUE)
```

```
> clinical <- read_csv("data/clinical.csv")
```

Photo by Michelle Tresemer on Unsplash

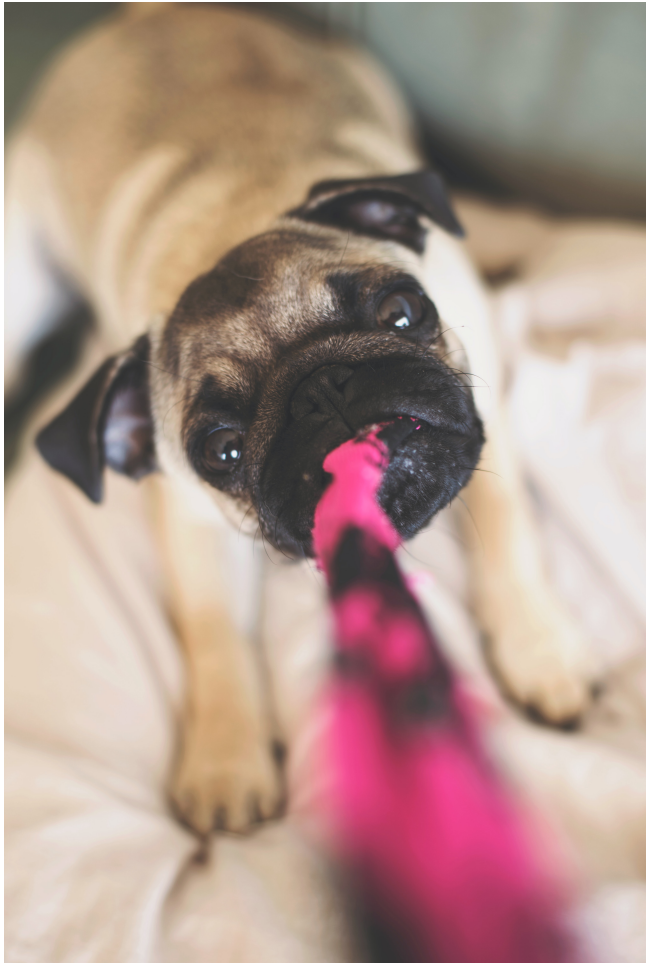


Photo by Yoad Shejtman on Unsplash

Photo by Hiro Takashima on Unsplash

Photo by MarlyneArt on Pixabay

Package ‘dplyr’

`filter` *Subset rows using column values*

Description

The `filter()` function is used to subset a data frame, retaining all rows that satisfy your conditions. To be retained, the row must produce a value of `TRUE` for all conditions. Note that when a condition evaluates to `NA` the row will be dropped, unlike base subsetting with `[`.

Usage

```
filter(.data, ..., .preserve = FALSE)
```

Arguments

`.data`	A data frame, data frame extension (e.g. a tibble), or a lazy data frame (e.g. from dbplyr or dtplyr). See *Methods*, below, for more details.

Data: starwars

To explore the basic data manipulation verbs of dplyr, we'll use the dataset `starwars`. This dataset contains 87 characters and comes from the Star Wars API, and is documented in `?starwars`

```
dim(starwars)
#> [1] 87 14
starwars
#> # A tibble: 87 x 14
#>   name  height  mass hair_color skin_color eye_color birth_year sex   g
#>   <chr>  <int> <dbl> <chr>      <chr>      <chr>          <dbl> <chr> <
#> 1 Luke…    172    77 blond      fair       blue            19   male  m
#> 2 C-3PO    167    75 <NA>       gold       yellow         112   none  m
#> 3 R2-D2     96    32 <NA>       white, bl… red             33   none  m
#> 4 Dart…    202   136 none       white      yellow          41.9 male  m
```

```r
#### Remove missing data in preparation for plotting ####

# filter out missing data
birth_complete <- clinical %>%
  filter(!is.na(year_of_birth)) %>%
  filter(!is.na(vital_status)) %>%
  filter(vital_status != "not reported")

# counting number of records in each cancer
cancer_counts <- clinical %>%
  count(disease) %>%
  arrange(n)

# get names of frequently occurring cancers
frequent_cancers <- cancer_counts %>%
  filter(n >= 500)

# extract data from cancers to keep
birth_reduced <- birth_complete %>%
  filter(disease %in% frequent_cancers$disease)

# save results to file in data/ named birth_reduced
write_csv(birth_reduced, "data/birth_reduced.csv")
```

Let's take a look at another example of piped commands:

```
# extract race, ethinicity, and disease from cases born prior to 1930
piped2 <- clinical %>%
  filter(year_of_birth < 1930) %>%
  select(race, ethnicity, disease)
```

In the code above, we're applying a mathematical condition to find specific rows, and the selecting certain columns. Does the order of commands differ? We can switch the order of the `filter` and `select` lines to see:

```
piped3 <- clinical %>%
  select(race, ethnicity, disease) %>%
  filter(year_of_birth < 1930)
```

The code above should give you an error, because in this case, the order does matter! The output from the second line does not include the `year_of_birth` column, so R is unable to apply the filter in the third line.

Challenge: Use pipes to extract the columns `gender`, `years_smoked`, and `year_of_birth` from the object `clinical` for only living patients (vital_status) who have smoked fewer than 1 `cigarettes_per_day` (solutions here)

Photo by MarlyneArt on Pixabay

borkr

Photo by Matt Nelson on Unsplash


SCICO
tsibble
rOpenSci
tidygraph
applicable
blogdown
bookdown
broom
covr
feasts
forecast
gganimate
nat
DATA SCIENCE IN A BOX
feather
flexdashboard
forcats
haven
hms
knitr
gtable
pins
%>%
Ceci n'est pas un pipe.
pkgdown
plumber
recipes
reprex
reticulate
rlang
rmarkdown
scales
Shiny
rvest

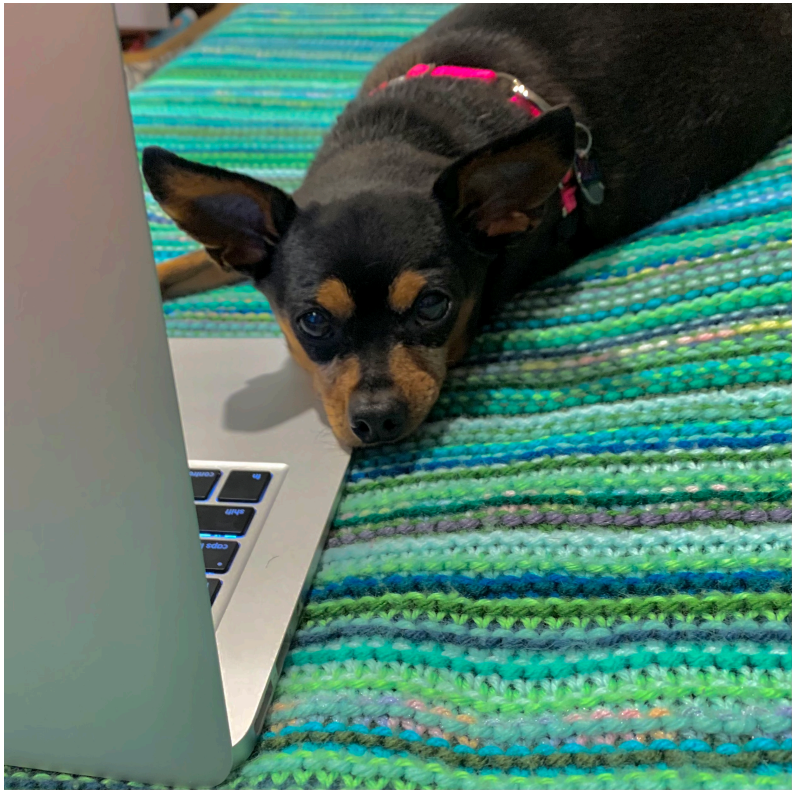
THE CARPENTRIES

FRED HUTCH

CURES START HERE®

MetaDocencia

rstd.io/global2021/katehertweck



@k8hert

@k8hertweck