R_intro_8

Karel Fišer, 2016

Preparing data for analysis

It is often said that 80% of data analysis is spent on the cleaning and preparing data.

Hadley Wickham

Hadley Wickham., Tidy data. The Journal of Statistical Software, vol. 59, 2014. DOI: 10.18637/jss.v059.i10

Dasu, T. and Johnson, T. (2003) Data Quality, in Exploratory Data Mining and Data Cleaning, John Wiley & Sons, Inc., Hoboken, NJ, USA. doi: 10.1002/0471448354.ch4

Data – what is what

Everything in the table is either variable name, observation identifier or value (… or annotation)

Description of the data in separate file

Consistent analysis habits. e.g. data and scripts live in one folder

(consider making analysis protocol)

Rectangular data

Pretty much all of the data we deal with is rectangular: has columns and rows.

Statistically, we want variables and observations.

Variables go in columns, observations go in rows (with occasional exceptions)

Hadley Wickham

reminder ...

from previous lectures

- **change rownames**
- **order**
- **change columns order**
- **add column**
- **substitute value**
- **NAs**
- **unique**

get data into R

If spreadsheet data, check them in spreadsheet software for:

- **multiline headers**
- **colour or formatting bears important info**
- **+/- decimal separator**
- **+/- encoding**
- **sheets**
- **outlier cells**

export spreadsheet to text

```
> library(gdata)
> dx <- read.xls("iris.xls")
```

When iris.xls → iris.txt → R

- **tab (or other delimited)**
- **encoding**
- **place on disk :-)**

```
> d <- read.table("iris.txt")
```

examine & fix data

- colnames
- missing values
- outliers
- character vectors and factors
    - letter cases
    - numbers as strings
    - strip white spaces

note: factor to numeric through character

examine & fix data

```
> dd <- read.table("iris_ex.txt", header = TRUE)
> str(dd)
> head(dd)
> dd
>
> colnames(dd)[4] <- "Petal.Width"
>
> dd[dd=="na"] <- NA
> # or use na.strings = "na" in read.table()
> # is.na()
```

character vectors and factors

```
> fs <- c("one", "3", "2", "One", "5     ")
> # [1] "one"    "3"      "2"      "One"    "5     "
>
> fs <- tolower(fs)
> fs[fs=="one"] <- 1
> fs <- trimws(fs) # or (better?) strip.white=TRUE in
  read.table()
```

factor levels

```
> ff <- factor(fs)
>
> as.numeric(ff)
> as.numeric(as.character(ff))
>
> factor(ff, levels = sort(levels(ff), decreasing =
  TRUE))
>
> levels(ff)[levels(ff)=="1"] <- "one"
```

Your turn

iris.xls → iris.txt → R

examine & fix iris_ex.txt

tidy and manipulate data

```
> library(tidyverse)
Loading tidyverse: ggplot2
Loading tidyverse: tibble
Loading tidyverse: tidyr
Loading tidyverse: readr
Loading tidyverse: purrr
Loading tidyverse: dplyr
Conflicts with tidy packages ----------------------------
filter(): dplyr, stats
lag():    dplyr, stats
```

http://r4ds.had.co.nz/

flights

```
> library(nycflights13)
> data(flights)
> flights
> # A tibble: 336,776 × 19
>     year month   day dep_time sched_dep_time dep_delay arr_time sched_arr_time
>    <int> <int> <int>    <int>          <int>     <dbl>    <int>          <int>
> 1   2013     1     1      517            515         2      830            819
> 2   2013     1     1      533            529         4      850            830
> 3   2013     1     1      542            540         2      923            850
> 4   2013     1     1      544            545        -1     1004           1022
> 5   2013     1     1      554            600        -6      812            837
> 6   2013     1     1      554            558        -4      740            728
> 7   2013     1     1      555            600        -5      913            854
> 8   2013     1     1      557            600        -3      709            723
> 9   2013     1     1      557            600        -3      838            846
> 10  2013     1     1      558            600        -2      753            745
> # ... with 336,766 more rows, and 11 more variables: arr_delay <dbl>,
> #   carrier <chr>, flight <int>, tailnum <chr>, origin <chr>, dest <chr>,
> #   air_time <dbl>, distance <dbl>, hour <dbl>, minute <dbl>, time_hour <dttm>
```

filter

```
> library(tidyverse) # library(dplyr)

>
> ## filter (subsetting on values)
> filter(flights, month == 1, day == 1, hour == 5) # filter excludes NAs
> flights[flights$month == 1 & flights$day == 1 & flights$hour == 5, ]
> # A tibble: 6 × 19
>    year month   day dep_time sched_dep_time dep_delay arr_time sched_arr_time
>   <int> <int> <int>    <int>          <int>     <dbl>    <int>          <int>
> 1  2013     1     1      517            515         2      830            819
> 2  2013     1     1      533            529         4      850            830
> 3  2013     1     1      542            540         2      923            850
> 4  2013     1     1      544            545        -1     1004           1022
> # ... with 11 more variables: arr_delay <dbl>, carrier <chr>, flight <int>,
> #   tailnum <chr>, origin <chr>, dest <chr>, air_time <dbl>, distance <dbl>,
> #   hour <dbl>, minute <dbl>, time_hour <dttm>
```

arrange

```
> ## arrange (rows ordering)
> arrange(flights, year, month, day)
> # Missing values are always sorted at the end
> arrange(flights, desc(arr_delay))
> flights[order(flights$arr_delay, decreasing =
  TRUE), ]
> # btw compare sort and order:
> # sort(rnorm(10)); order(rnorm(10))
```

summaries

```
> summarise(flights, delay = mean(dep_delay, na.rm = TRUE))
>
> by_carr <- group_by(flights, carrier)
> summarise(by_carr, delay = mean(dep_delay, na.rm = TRUE))
> # A tibble: 16 × 2
>    carrier     delay
>      <chr>     <dbl>
> 1       9E 16.725769
> 2       AA  8.586016
> 3       AS  5.804775
> mean(flights$dep_delay[flights$carrier == "9E"], na.rm =
  TRUE) #  or better aggregate …
> [1] 16.72577
```

Your turn

filter

arrange

summarise

Multiple variables in one column

```
> data("table3")
> head(table3)
> # A tibble: 6 × 3
>       country  year              rate
>         <chr> <int>             <chr>
> 1 Afghanistan  1999      745/19987071
> 2 Afghanistan  2000     2666/20595360
> 3      Brazil  1999   37737/172006362
> 4      Brazil  2000   80488/174504898
> 5       China  1999 212258/1272915272
> 6       China  2000 213766/1280428583
```

separate

```
> separate(table3, rate, into = c("cases",
  "population"))
> # A tibble: 6 × 4
>       country  year  cases population
> *       <chr> <int>  <chr>      <chr>
> 1 Afghanistan  1999    745   19987071
> 2 Afghanistan  2000   2666   20595360
> 3      Brazil  1999  37737  172006362
>
> # By default, separate() will split values wherever
  it sees a non-alphanumeric character.
> # unite()
```

Column headers are values

=> wide to long data conversion

```
> data("table4a")
> table4a
> # A tibble: 3 × 3
>       country `1999` `2000`
> *       <chr>  <int>  <int>
> 1 Afghanistan    745   2666
> 2      Brazil  37737  80488
> 3       China 212258 213766
```

gather

```
> gather(table4a, `1999`, `2000`, key = "year", value = "cases")
> # A tibble: 6 × 3
>       country  year  cases
>         <chr> <chr>  <int>
> 1 Afghanistan  1999    745
> 2      Brazil  1999  37737
> 3       China  1999 212258
> 4 Afghanistan  2000   2666
> 5      Brazil  2000  80488
> 6       China  2000 213766
> # gather(table4a, key = "year", value = "cases", -country)
```

Your turn

separate table3 with explicit separator

iris_ex.txt -> wide to long

counts and tables

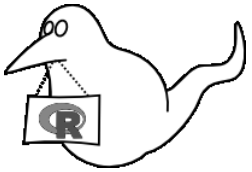
```
> data(mtcars)
> table(mtcars[, c("cyl", "gear")])
> as.data.frame(table(mtcars[, c("cyl", "gear")]))
```

Your turn

table fight

ggplot2 filter (subset) data

http://www.bioinformin.net



https://cran.r-project.org/web/packages/tidyr/vignettes/tidy-data.html
http://www.cookbook-r.com/Manipulating_data/
http://r4ds.had.co.nz/
https://github.com/tidyverse/tidyverse