The Tidyverse

BIOF 339

9/25/2018

What is the "Tidyverse"?

The tidyverse is an opinionated collection of R packages designed for data science. All packages share an underlying design philosophy, grammar, and data structures

What is the "Tidyverse"?

A set of R packages that:

- help make data more computer-friendly
- while making your code more human-friendly

- Most of these packages are (co-)written by Dr. Hadley Wickham, who has rockstar status in the R world
- They are supported by the company RStudio

Tidying data

Tidy data

Tidy datasets are all alike,
but every messy data is messy in its own way

Tidy data

Tidy data is a **computer-friendly** format based on the following characteristics:

- Each row is one observation
- Each column is one variable
- Each set of observational unit forms a table

All other forms of data can be considered **messy data**.

Let us count the ways

There are many ways data can be messy. An incomplete list….

- Column headers are values, not variables
- Multiple variables are stored in a single column
- Variables are stored in both rows and columns
- Multiple types of observational units are saved in the same table
- A single observational unit is stored in multiple tables

Ways to have messy (i.e. not tidy) data

1. Column headers contain values

Country	< $10K	$10-20K	$20-50K	$50-100K	> $100K
India	40	25	25	9	1
USA	20	20	20	30	10

Ways to have messy (i.e. not tidy) data

1. Column headers contain values

Country	Income	Percentage
India	< $10K	40
USA	< $10K	20

This is a case of reshaping or melting

Ways to have messy (i.e. not tidy) data

1. Multiple variables in one column

Country	Year	M_0-14	F_0-14	M_ 15-60	F_15-60	M_60+	F_60+
UK	2010						
UK	2011						

Country	Year	Gender	Age	Count

Separating columns into different variables

Tidying data

The typical steps are

- Transforming data from wide to tall (*gather*) and from tall to wide (*spread*)
- Separating columns into different columns
- Putting columns together into new variables

Cleaning data

Some actions on data

- Creating new variables (*mutate*)
- Choose some columns (*select*)
- Selecting rows based on some criteria (*filter*)
- Sort data based on some variables (*arrange*)

Example data

```
data(mtcars)
knitr::kable(head(mtcars, 3))
```

	mpg	cyl	disp	hp	drat	wt	qsec	vs	am	gear	carb
Mazda RX4	21.0	6	160	110	3.90	2.620	16.46	0	1	4	4
Mazda RX4 Wag	21.0	6	160	110	3.90	2.875	17.02	0	1	4	4
Datsun 710	22.8	4	108	93	3.85	2.320	18.61	1	1	4	1

- Car names are in an attribute of the data.frame called `rownames`. So it's not in a column
- We might want to convert fuel economy to metric
- We might just want to look at the relationship between displacement and fuel economy based on number of cylinders

Example data (link)

```
link <- 'https://dl.dropboxusercontent.com/s/pqavhcckshqxtjm/brca.csv'
download.file(link, 'brca.csv')
brca_data <- read.csv('brca.csv', stringsAsFactors=FALSE)
```

	id	diagnosis	radius_mean	texture_mean	perimeter_mean	area_mean	smoothness_mean	compactness_mean	concavity_mean	concave.points_mean	sy
1	842302	M	17.990	10.38	122.80	1001.0	0.11840	0.27760	0.300100	0.147100	
2	842517	M	20.570	17.77	132.90	1326.0	0.08474	0.07864	0.086900	0.070170	
3	84300903	M	19.690	21.25	130.00	1203.0	0.10960	0.15990	0.197400	0.127900	
4	84348301	M	11.420	20.38	77.58	386.1	0.14250	0.28390	0.241400	0.105200	
5	84358402	M	20.290	14.34	135.10	1297.0	0.10030	0.13280	0.198000	0.104300	
6	843786	M	12.450	15.70	82.57	477.1	0.12780	0.17000	0.157800	0.080890	
7	844359	M	18.250	19.98	119.60	1040.0	0.09463	0.10900	0.112700	0.074000	

The `tidyverse` package

The `tidyverse` package is a meta-package that installs a set of packages that are useful for data cleaning, data tidying and data munging (manipulating data to get a computationally "attractive" dataset)

The `tidyverse` package

```
# install.packages('tidyverse')
library(tidyverse)
```

```
## ── Attaching packages ─────────────────────────────────────── tidyverse 1.2.1 ──

## ✔ ggplot2 3.0.0     ✔ purrr   0.2.5
## ✔ tibble  1.4.2     ✔ dplyr   0.7.6
## ✔ tidyr   0.8.1     ✔ stringr 1.3.1
## ✔ readr   1.1.1     ✔ forcats 0.3.0

## ── Conflicts ────────────────────────────────────────── tidyverse_conflicts() ──
## ✖ dplyr::filter() masks stats::filter()
## ✖ dplyr::lag()    masks stats::lag()
```

You can specify a function from a particular package as `dplyr::filter`. Note there are two colons there

Core `tidyverse` packages

Package	Description
ggplot2	Data visualization (next week)
tibble	data.frame on steroids
tidyr	Data tidying (today)
readr	Reading text files (CSV)
purrr	Applying functions to data iteratively(later this sem)
dplyr	Data cleaning and munging (today)
stringr	String (character) manipulation
forcats	Manipulating categorical variables

Additional `tidyverse` packages

Package	Description
readxl	Read Excel files
haven	Read SAS, SPSS, Stata files
lubridate	Deal with dates and times
magrittr	Provides the pipe operator %>%
glue	Makes pasting text and data easier

Pipes

A pipe example (data munging)

```
library(tidyverse)
updated_cars <-
  mtcars %>%
  rownames_to_column(var = 'Model') %>%
  mutate(kmpg = mpg * 1.6) %>%
  select(Model, kmpg, cyl, disp) %>%
  filter(cyl == 6)
```

Model	kmpg	cyl	disp
Mazda RX4	33.60	6	160.0
Mazda RX4 Wag	33.60	6	160.0
Hornet 4 Drive	34.24	6	258.0
Valiant	28.96	6	225.0

A pipe example (data munging)

```
library(tidyverse)
updated_cars <-
  mtcars %>% # Take the data set
  rownames_to_column(var = 'Model') %>% # Make rownames a column
  select(Model, mpg, cyl, disp) %>% # Keep only certain columns
  mutate(kmpg = mpg * 1.6) %>% # Create a new variable
  filter(cyl == 6) # Keep only certain rows
```

The idea is to use **verbs** to express operations on a dataset, so it is easier to express what you want to do in code.

The pipe operator

The pipe operator `%>%` (technically from the package `magrittr`) takes a **data.frame or tibble object** on the left, then "pipes" it to a function that takes the **data.frame object as its first argument**.

```
mtcars %>% mutate(kmpg = mpg * 1.6)
```

would be the same as

```
mutate(mtcars, kmpg = mpg * 1.6)
```

The pipe operator

With pipes

```
mtcars %>%
  rownames_to_column(var = "Model") %>%
  select(Model:disp) %>%
  mutate(kmpg = mpg * 1.6) %>%
  filter(cyl == 6)
```

Without pipes

```
tmp <- rownames_to_column(mtcars, var="Model")
tmp3 <- select(tmp2, Model:disp)
tmp2 <- mutate(tmp, kmpg = mpg * 1.6)
tmp4 <- filter(tmp3, cyl == 6)
```

Both are fine, but I find pipes help translating my thoughts into code better

The pipe operator

With pipes + tidyverse

```
updated_cars <- mtcars %>%
  rownames_to_column(var = "Model") %>%
  select(Model:disp) %>%
  mutate(kmpg = mpg * 1.6) %>%
  filter(cyl == 6)
```

Without tidyverse

```
mtcars['Model'] <- rownames(mtcars)
tmp <- mtcars[,c('Model','mpg','cyl','disp')] # Can't use the : operator for
tmp['kmpg'] <- tmp['mpg'] * 1.6 # Note we need to quote the names
updated_cars <- tmp[tmp['cyl'] == 6,]
```

Idea was to make it easier to write expressive code without getting too hung up with syntax.

Going through step by step

```
mtcars
```

	mpg	cyl	disp	hp	drat	wt	qsec	vs	am	gear	carb
Mazda RX4	21.0	6	160	110	3.90	2.620	16.46	0	1	4	4
Mazda RX4 Wag	21.0	6	160	110	3.90	2.875	17.02	0	1	4	4
Datsun 710	22.8	4	108	93	3.85	2.320	18.61	1	1	4	1

```
mtcars %>%
  rownames_to_column(var = "Model")
```

Model	mpg	cyl	disp	hp	drat	wt	qsec	vs	am	gear	carb
Mazda RX4	21.0	6	160	110	3.90	2.620	16.46	0	1	4	4
Mazda RX4 Wag	21.0	6	160	110	3.90	2.875	17.02	0	1	4	4
Datsun 710	22.8	4	108	93	3.85	2.320	18.61	1	1	4	1

```
mtcars %>%
  rownames_to_column(var = "Model") %>%
  select(Model:disp)
```

Model	mpg	cyl	disp
Mazda RX4	21.0	6	160
Mazda RX4 Wag	21.0	6	160
Datsun 710	22.8	4	108

```
mtcars %>%
  rownames_to_column(var = "Model") %>%
  select(Model:disp) %>%
  mutate(kmpg = mpg * 1.6)
```

Model	mpg	cyl	disp	kmpg
Mazda RX4	21.0	6	160	33.60
Mazda RX4 Wag	21.0	6	160	33.60
Datsun 710	22.8	4	108	36.48

```
mtcars %>%
  rownames_to_column(var = "Model") %>%
  select(Model:disp) %>%
  mutate(kmpg = mpg * 1.6) %>%
  filter(cyl == 6)
```

Model	mpg	cyl	disp	kmpg
Mazda RX4	21	6	160	33.6
Mazda RX4 Wag	21	6	160	33.6

Data tidying

```
link <- 'https://dl.dropboxusercontent.com/s/pqavhcckshqxtjm/brca.csv'
download.file(link, 'brca.csv')
brca_data <- read.csv('brca.csv', stringsAsFactors=FALSE)
```

	id	diagnosis	radius_mean	texture_mean	perimeter_mean	area_mean	smoothness_mean	compactness_mean	concavity_mean	concave.points_mean	sy
1	842302	M	17.990	10.38	122.80	1001.0	0.11840	0.27760	0.300100	0.147100	
2	842517	M	20.570	17.77	132.90	1326.0	0.08474	0.07864	0.086900	0.070170	
3	84300903	M	19.690	21.25	130.00	1203.0	0.10960	0.15990	0.197400	0.127900	
4	84348301	M	11.420	20.38	77.58	386.1	0.14250	0.28390	0.241400	0.105200	
5	84358402	M	20.290	14.34	135.10	1297.0	0.10030	0.13280	0.198000	0.104300	
6	843786	M	12.450	15.70	82.57	477.1	0.12780	0.17000	0.157800	0.080890	
7	844359	M	18.250	19.98	119.60	1040.0	0.09463	0.10900	0.112700	0.074000	

```
library(tidyverse)
names(brca_data)
```

```
##  [1] "id"                "diagnosis"
##  [3] "radius_mean"       "texture_mean"
##  [5] "perimeter_mean"    "area_mean"
##  [7] "smoothness_mean"   "compactness_mean"
##  [9] "concavity_mean"    "concave.points_mean"
## [11] "symmetry_mean"     "fractal_dimension_mean"
## [13] "radius_se"         "texture_se"
## [15] "perimeter_se"      "area_se"
## [17] "smoothness_se"     "compactness_se"
## [19] "concavity_se"      "concave.points_se"
## [21] "symmetry_se"       "fractal_dimension_se"
## [23] "radius_worst"      "texture_worst"
## [25] "perimeter_worst"   "area_worst"
## [27] "smoothness_worst"  "compactness_worst"
## [29] "concavity_worst"   "concave.points_worst"
## [31] "symmetry_worst"    "fractal_dimension_worst"
## [33] "X"
```

Need for tidying

1. For the same variable (e.g., radius) there are 3 columns giving the mean, se and worst value. So the names of the metric are stored in the column names
2. There are really 3 kinds of summaries for each metric – mean, se and worst.

Tidying

```
library(tidyverse)
brca_data
```

id	diagnosis	radius_mean	texture_mean	perimeter_mean	area_mean	smoothness_mean	compact
842302	M	17.99	10.38	122.8	1001	0.11840	
842517	M	20.57	17.77	132.9	1326	0.08474	
84300903	M	19.69	21.25	130.0	1203	0.10960	

Tidying (selecting)

```
library(tidyverse)
brca_data %>%
  select(id, diagnosis, ends_with('mean'), ends_with('se'), ends_with('worst
         -starts_with('fractal'))
                              # removes columns starting with "fractal"
```

id	diagnosis	radius_mean	texture_mean	perimeter_mean	area_mean	smoothness_mean	compact
842302	M	17.99	10.38	122.8	1001	0.11840	
842517	M	20.57	17.77	132.9	1326	0.08474	
84300903	M	19.69	21.25	130.0	1203	0.10960	

Tidying (gathering)

```
library(tidyverse)
brca_data %>%
  select(id, diagnosis, ends_with('mean'),
         ends_with('se'), ends_with('worst'),
         -starts_with('fractal')) %>%
  gather(variable, value, -id, -diagnosis) # operate on everything but
```

Column names become variable, everything stays aligned with id and diagnosis

id	diagnosis	variable	value
842302	M	radius_mean	17.99
842517	M	radius_mean	20.57
84300903	M	radius_mean	19.69

Tidying (separating)

```
library(tidyverse)
brca_data %>%
  select(id, diagnosis, ends_with('mean'),
         ends_with('se'), ends_with('worst'),
         -starts_with('fractal')) %>%
  gather(variable, value, -id, -diagnosis) %>%
  separate(variable, c("Variable","stat"), sep="_", remove = T)
```

Split `variable` into 2 cols, `Variable` and `stat`

id	diagnosis	Variable	stat	value
842302	M	radius	mean	17.99
842517	M	radius	mean	20.57
84300903	M	radius	mean	19.69

Tidying (spreading)

```
library(tidyverse)
brca_data %>%
  select(id, diagnosis, ends_with('mean'),
         ends_with('se'), ends_with('worst'),
         -starts_with('fractal')) %>%
  gather(variable, value, -id, -diagnosis) %>%
  separate(variable, c("Variable","stat"), sep="_", remove = T) %>%
  spread(stat, value)
```

id	diagnosis	Variable	mean	se	worst
8670	M	area	748.90000	48.31000	1156.0000
8670	M	compactness	0.12230	0.01484	0.2394
8670	M	concave.points	0.08087	0.01093	0.1514

References

1. Introduction to dplyr
2. Tidy data
3. The paper on tidy data