TidyR Tutorial

Amy Lee (@miniscience girl)

2017-03-15

Tidy Data and Data Analysis Pipeline

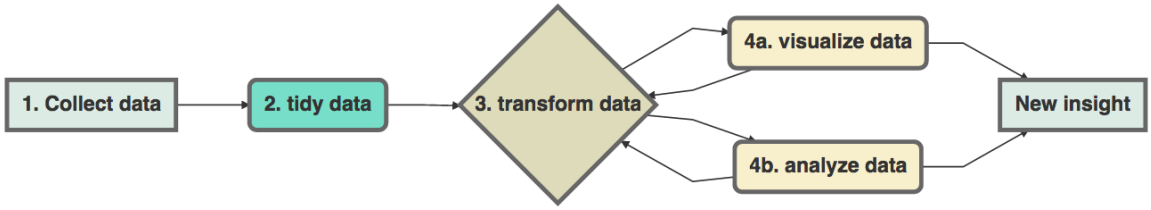

Figure 1: Data Analysis Pipeline, Joseph V. Casillas.

> It is often said that 80% of data analysis is spent on the process of cleaning and preparing the data. (Dasu and Johnson, 2003)

What is tidy data?

- every column in your dataframe represents a variable
- every row represents an observation
- also known as long format

Why do we need tidy data?

- easy to manipulate (variable are easy to access as vectors)
- easy to visualize
- easy to model
- hard for human eyeballs, but easy for computation

> If you are struggling to make a figure, for example, stop and think hard about whether your data is tidy. Untidiness is a common, often overlooked cause of agony in data analysis and visualization. (Jenny Bryan, STAT545)

Example of untidy data (Jenny Bryan, STAT545)

Why are these examples of untidy data?

- What's the total number of words spoken by male hobbits in all three movies?
- Is there a more talkative `Race`?

The Fellowship Of The Ring

Race	Female	Male
Elf	1229	971
Hobbit	14	3644
Man	0	1995

The Two Towers

Race	Female	Male
Elf	331	513
Hobbit	0	2463
Man	401	3589

The Return Of The King

Race	Female	Male
Elf	183	510
Hobbit	2	2673
Man	268	2459

Figure 2: Untidy data, Jenny Bryan.

How do we turn these tables into a tidy dataframe?

Let's load our required packages:

```
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
```

Let's read these dataframes in:

```
fship <- read_csv("The_Fellowship_Of_The_Ring.csv")
```

```
## Parsed with column specification:
## cols(
##   Film = col_character(),
##   Race = col_character(),
##   Female = col_integer(),
##   Male = col_integer()
## )
```

```
ttow <- read_csv("The_Two_Towers.csv")
```

```
## Parsed with column specification:
## cols(
##   Film = col_character(),
##   Race = col_character(),
##   Female = col_integer(),
##   Male = col_integer()
## )
```

```
rking <- read_csv("The_Return_Of_The_King.csv")
```

```
## Parsed with column specification:
## cols(
##   Film = col_character(),
```

```
##   Race = col_character(),
##   Female = col_integer(),
##   Male = col_integer()
## )
```

Collect untidy dataframes into one dataframe:

```
lotr_untidy <- dplyr::bind_rows(fship, ttow, rking)
```

This dataframe is still untidy because "word count" is spread out between two columns, `Male` and `Female`. So to make this dataframe tidy, we need to:

- `gather()` up the word counts into one column
- create a new column for `Gender`

Time to make this dataframe tidy!

```
lotr_tidy <-
  gather(lotr_untidy, key = 'Gender', value = 'Words', Female, Male)

lotr_tidy
```

```
## # A tibble: 18 × 4
##                          Film   Race Gender Words
##                         <chr>  <chr>  <chr> <int>
## 1  The Fellowship Of The Ring    Elf Female  1229
## 2  The Fellowship Of The Ring Hobbit Female    14
## 3  The Fellowship Of The Ring    Man Female     0
## 4              The Two Towers    Elf Female   331
## 5              The Two Towers Hobbit Female     0
## 6              The Two Towers    Man Female   401
## 7      The Return Of The King    Elf Female   183
## 8      The Return Of The King Hobbit Female     2
## 9      The Return Of The King    Man Female   268
## 10 The Fellowship Of The Ring    Elf   Male   971
## 11 The Fellowship Of The Ring Hobbit   Male  3644
## 12 The Fellowship Of The Ring    Man   Male  1995
## 13             The Two Towers    Elf   Male   513
## 14             The Two Towers Hobbit   Male  2463
## 15             The Two Towers    Man   Male  3589
## 16     The Return Of The King    Elf   Male   510
## 17     The Return Of The King Hobbit   Male  2673
## 18     The Return Of The King    Man   Male  2459
```

Want to see what else you can do with this dataset? Check out Jenny Bryan's LOTR GitHub Repo!

Challenge Question 1

`spread()` is another tidyr function that converts a dataframe from the long format to the wide format. How would you convert the `lotr_tidy` dataframe back into the `lotr_untidy` dataframe?

Challenge Question 2

In the EDAWR dataset, `cases`, we have the number of tuberculosis cases reported in France, Germany and United States from 2011 to 2013. What are the total number of tuberculosis cases reported over three years per country?

```
devtools::install_github("rstudio/EDAWR")
```

```
## Skipping install of 'EDAWR' from a github remote, the SHA1 (2652ea64) has not changed since last ins
##   Use `force = TRUE` to force installation
```

```
library(EDAWR)
```

```
##
## Attaching package: 'EDAWR'
```

```
## The following objects are masked from 'package:tidyr':
##
##     population, who
```

```
cases
```

```
##   country  2011  2012  2013
## 1      FR  7000  6900  7000
## 2      DE  5800  6000  6200
## 3      US 15000 14000 13000
```

Other useful functions from tidyr - Separate and Unite

Let's use the EDAWR dataset again. This time, we are going to use the `storms` data, which has the maximum wind speeds for six Atlantic hurricanes.

```
storms
```

```
## # A tibble: 6 × 4
##     storm  wind pressure       date
##     <chr> <int>    <int>     <date>
## 1 Alberto   110     1007 2000-08-03
## 2    Alex    45     1009 1998-07-27
## 3 Allison    65     1005 1995-06-03
## 4     Ana    40     1013 1997-06-30
## 5  Arlene    50     1010 1999-06-11
## 6  Arthur    45     1010 1996-06-17
```

`separate()` allows you to separate a column into multiple other columns by using a separator. For example, if we want to separate the `date` column into `year`, `month`, `day`, we can do that by:

```
storms.sep <- separate(storms, date, c("year", "month", "day"), sep = "-")
```

Challenge Question 3

How do you combine the three separate columns, `year`, `month`, `day`, that you just created in `storms.sep` back into one column, `date`? Hint: `unite()` works the opposite way as `separate()`.