Introduction to Data Science: Common operations for data tidying

Héctor Corrada Bravo

University of Maryland, College Park, USA

2020-02-17

Tidying data

- Common problems in data preparation:
- Use cases commonly found in raw datasets that need to be addressed to turn messy data into tidy data.
- We derive many of our ideas from the paper Tidy Data by Hadley Wickham.

Tidying data

Here we assume we are working with a data model based on rectangular data structures where

1. Each attribute (or variable) forms a column
2. Each entity (or observation) forms a row
3. Each type of entity (observational unit) forms a table

Tidying data

Here is an example of a tidy dataset:

```
library(nycflights13)

head(flights)
```

```
## # A tibble: 6 x 19

##    year month   day dep_time sched_dep_time dep_delay arr_time

##   <int> <int> <int>    <int>          <int>     <dbl>    <int>

## 1  2013     1     1      517            515         2      830

## 2  2013     1     1      533            529         4      850

## 3  2013     1     1      542            540         2      923

## 4  2013     1     1      544            545        -1     1004
```

Common problems in messy data

The set of common operations we will study are based on these common problems found in datasets.

- Column headers are values, not variable names (gather)
- Multiple variables stored in one column (split)
- Variables stored in both rows and column (rotate)
- Multiple types of observational units are stored in the same table (normalize)

Common problems in messy data

Headers as values

The first problem we'll see is the case where a table header contains values.

```
## # A tibble: 18 x 11
##    religion `<$10k` `$10-20k` `$20-30k` `$30-40k` `$40-50k` `$50-75k`
##    <chr>      <dbl>     <dbl>     <dbl>     <dbl>     <dbl>     <dbl>
##  1 Agnostic      27        34        60        81        76       137
##  2 Atheist       12        27        37        52        35        70
##  3 Buddhist      27        21        30        34        33        58
##  4 Catholic     418       617       732       670       638      1116
```

Common problems in messy data

A tidy version of this table would consider the *variables* of each observation to be `religion, income, frequency` where `frequency` has the number of respondents for each religion and income range.

Common problems in messy data

The function to use in the `tidyr` package is `gather`:

```
tidy_pew <- gather(pew, income, frequency, -religion)

tidy_pew
```

```
## # A tibble: 180 x 3
##    religion                 income frequency
##    <chr>                    <chr>      <dbl>
##  1 Agnostic                 <$10k         27
##  2 Atheist                  <$10k         12
##  3 Buddhist                 <$10k         27
##  4 Catholic                 <$10k        418
```

Common problems in messy data

Multiple variables in one column

```
tb <- read_csv(file.path(data_dir, "tb.csv"))

tb
```

```
## # A tibble: 5,769 x 22

##    iso2   year   m04  m514  m014 m1524 m2534 m3544 m4554 m5564   m65    mu

##    <chr> <dbl> <dbl> <dbl> <dbl> <dbl> <dbl> <dbl> <dbl> <dbl> <dbl> <dbl>

##  1 AD     1989    NA    NA    NA    NA    NA    NA    NA    NA    NA    NA

##  2 AD     1990    NA    NA    NA    NA    NA    NA    NA    NA    NA    NA

##  3 AD     1991    NA    NA    NA    NA    NA    NA    NA    NA    NA    NA

##  4 AD     1992    NA    NA    NA    NA    NA    NA    NA    NA    NA    NA
```

Common problems in messy data

- We need to `gather` the tabulation columns into a `demo` and `n` columns (for demographic and number of cases):

```
tidy_tb <- gather(tb, demo, n, -iso2, -year)

tidy_tb
```

```
## # A tibble: 115,380 x 4
##    iso2   year demo      n
##    <chr> <dbl> <chr> <dbl>
##  1 AD     1989 m04      NA
##  2 AD     1990 m04      NA
##  3 AD     1991 m04      NA
```

Common problems in messy data

Need to `separate` the values in the `demo` column into two variables `sex` and `age`

```
tidy_tb <- separate(tidy_tb, demo, c("sex", "age"), sep=1)

tidy_tb
```

```
## # A tibble: 115,380 x 5
##    iso2   year sex   age       n
##    <chr> <dbl> <chr> <chr> <dbl>
##  1 AD     1989 m     04       NA
##  2 AD     1990 m     04       NA
##  3 AD     1991 m     04       NA
```

Common problems in messy data

We can put these two commands together in a pipeline:

```
tidy_tb <- tb %>%
  gather(demo, n, -iso2, -year)  %>%
  separate(demo, c("sex", "age"), sep=1)
tidy_tb
```

```
## # A tibble: 115,380 x 5
##    iso2   year sex   age       n
##    <chr> <dbl> <chr> <chr> <dbl>
##  1 AD     1989 m     04       NA
##  2 AD     1990 m     04       NA
```

Common problems in messy data

Variables stored in both rows and columns

This is the messiest, commonly found type of data.

```
weather <- read_csv(file.path(data_dir, "weather.csv"))

weather
```

```
## # A tibble: 22 x 35
##    id     year month element    d1    d2    d3    d4    d5    d6    d7
##    <chr> <dbl> <dbl> <chr>   <dbl> <dbl> <dbl> <dbl> <dbl> <dbl> <dbl>
##  1 MX17…  2010     1 tmax       NA   NA    NA      NA   NA      NA    NA
##  2 MX17…  2010     1 tmin       NA   NA    NA      NA   NA      NA    NA
```

Common problems in messy data

We have two rows for each month:

- one with maximum daily temperature
- one with minimum daily temperature
- the columns starting with d correspond to the day in the where the measurements were made.

Common problems in messy data

```
weather %>%
  gather(day, value, d1:d31, na.rm=TRUE) %>%
  spread(element, value)
```

```
## # A tibble: 33 x 6
##    id       year month day    tmax  tmin
##    <chr>   <dbl> <dbl> <chr> <dbl> <dbl>
##  1 MX17004  2010     1 d30    27.8  14.5
##  2 MX17004  2010     2 d11    29.7  13.4
##  3 MX17004  2010     2 d2     27.3  14.4
##  4 MX17004  2010     2 d23    29.9  10.7
##  5 MX17004  2010     2 d3     24.1  14.4
```

Common problems in messy data

Multiple types in one table

Remember that an important aspect of tidy data is that it contains exactly one kind of observation in a single table.

```
## # A tibble: 5,307 x 7
##     year artist       track                   time  date.entered week   rank
##    <dbl> <chr>        <chr>                   <tim> <date>       <chr> <dbl>
##  1  2000 2 Pac        Baby Don't Cry (Keep… 04:22 2000-02-26   wk1      87
##  2  2000 2Ge+her      The Hardest Part Of … 03:15 2000-09-02   wk1      91
##  3  2000 3 Doors Down Kryptonite            03:53 2000-04-08   wk1      81
##  4  2000 3 Doors Down Loser                 04:24 2000-10-21   wk1      76
```

Common problems in messy data

Let's make a song table that only includes information about songs:

```
song <- tidy_billboard %>%
  dplyr::select(artist, track, year, time, date.entered) %>%
  unique()
song
```

```
## # A tibble: 317 x 5
##    artist        track                    year time   date.entered
##    <chr>         <chr>                   <dbl> <time> <date>
##  1 Nelly         (Hot S**t) Country G...  2000 04:17  2000-04-29
##  2 Nu Flavor     3 Little Words           2000 03:54  2000-06-03
```

Common problems in messy data

Next, we would like to remove all the song information from the rank table.

```
song <- tidy_billboard %>%
  dplyr::select(artist, track, year, time, date.entered) %>%
  unique() %>%
  mutate(song_id = row_number())
song
```

```
## # A tibble: 317 x 6
##    artist        track                    year time   date.entered song_id
##    <chr>         <chr>                   <dbl> <time> <date>         <int>
```

Common problems in messy data

Now we can make a rank table, we combine the tidy billboard table with our new song table using a `join`.

```
tidy_billboard %>%

  left_join(song, c("artist", "year", "track", "time", "date.entered"))
```

```
## # A tibble: 5,307 x 8

##     year artist track                time  date.entered week   rank song_id

##    <dbl> <chr>  <chr>                <tim> <date>       <chr> <dbl>   <int>

##  1  2000 Nelly  (Hot S**t) Country … 04:17 2000-04-29   wk1     100       1

##  2  2000 Nelly  (Hot S**t) Country … 04:17 2000-04-29   wk2      99       1

##  3  2000 Nelly  (Hot S**t) Country … 04:17 2000-04-29   wk3      96       1
```

Common problems in messy data

```
rank <- tidy_billboard %>%
  left_join(song, c("artist", "year", "track", "time", "date.entered")) %>%
  dplyr::select(song_id, week, rank)
rank
```

```
## # A tibble: 5,307 x 3
##    song_id week   rank
##      <int> <chr> <dbl>
##  1       1 wk1     100
##  2       1 wk2      99
##  3       1 wk3      96
##  4       1 wk4      76
```

Tidy data and the ER model

tidy data as presented here is purposefully parallel to the ER model formalism.

However, this formalism extends beyond what we've seen here targeted towards data analysis. Many features of the ER model formalism are more applicable to data management issues, especially consistency and redundancy.