Manipulating Data in R

Data Wrangling in R

Reshaping Data

In this module, we will show you how to:

1. Reshaping data from wide (fat) to long (tall)
2. Reshaping data from long (tall) to wide (fat)

Setup

We will show you how to do each operation in base R then show you how to use the `dplyr` or `tidyr` package to do the same operation (if applicable).

See the “Data Wrangling Cheat Sheet using `dplyr` and `tidyr`”:

- https://www.rstudio.com/wp-content/uploads/2015/02/data-wrangling-cheatsheet.pdf

What is wide/long data?

See http://www.cookbook-r.com/Manipulating_data/Converting_data_between_wide_and_long_format/

- Wide - multiple columns per observation
  - e.g. visit1, visit2, visit3

```
# A tibble: 2 x 4
     id visit1 visit2 visit3
  <int>  <dbl>  <dbl>  <dbl>
1     1     10      4      3
2     2      5      6     NA
```

- Long - multiple rows per observation

```
# A tibble: 5 x 3
     id visit value
  <dbl> <int> <dbl>
1     1     1    10
2     1     2     4
3     1     3     3
4     2     1     5
5     2     2     6
```

What is wide/long data?

More accurately, data is wide or long **with respect** to certain variables.

Data used: Charm City Circulator

```
circ = read_csv("../data/Charm_City_Circulator_Ridership.csv")
head(circ, 2)
```

```
# A tibble: 2 x 15
  day   date  orangeBoardings orangeAlightings orangeAverage purpleBoardings
  <chr> <chr>           <dbl>            <dbl>         <dbl>           <dbl>
1 Mond… 01/1…             877             1027           952              NA
2 Tues… 01/1…             777              815           796              NA
# … with 9 more variables: purpleAlightings <dbl>, purpleAverage <dbl>,
#   greenBoardings <dbl>, greenAlightings <dbl>, greenAverage <dbl>,
#   bannerBoardings <dbl>, bannerAlightings <dbl>, bannerAverage <dbl>, daily
```

```
class(circ$date)
```

```
[1] "character"
```

Creating a Date class from a character date

```
library(lubridate) # great for dates!
```

```
sum(is.na(circ$date))
```

```
[1] 0
```

```
sum( circ$date == "")
```

```
[1] 0
```

```
circ = mutate(circ, date = mdy(date))
sum( is.na(circ$date) ) # all converted correctly
```

```
[1] 0
```

```
head(circ$date, 3)
```

```
[1] "2010-01-11" "2010-01-12" "2010-01-13"
```

```
class(circ$date)
```

```
[1] "Date"
```

Reshaping data from wide (fat) to long (tall): base R

The `reshape` command exists. It is a **confusing** function. Don't use it.

tidyr package

`tidyr` allows you to “tidy” your data. We will be talking about:

- `gather` - make multiple columns into variables, (wide to long)
- `spread` - make a variable into multiple columns, (long to wide)
- `separate` - string into multiple columns
- `unite` - multiple columns into one string

Reshaping data from wide (fat) to long (tall): tidyr

`tidyr::gather` - puts column data into rows.

We want the column names into "`var`" variable in the output dataset and the value in "`number`" variable. We then describe which columns we want to "gather:"

```
long = gather(circ, key = "var", value = "number",
              -day, -date, -daily)
head(long, 4)
```

```
# A tibble: 4 x 5
  day       date       daily var              number
  <chr>     <date>     <dbl> <chr>             <dbl>
1 Monday    2010-01-11  952  orangeBoardings     877
2 Tuesday   2010-01-12  796  orangeBoardings     777
3 Wednesday 2010-01-13 1212. orangeBoardings    1203
4 Thursday  2010-01-14 1214. orangeBoardings    1194
```

Reshaping data from wide (fat) to long (tall): tidyr

- Could be explicit on what we want to gather

```
long = gather(circ, key = "var", value = "number",
              starts_with("orange"), starts_with("purple"),
              starts_with("green"), starts_with("banner"))
long
```

```
# A tibble: 13,752 x 5
   day       date        daily var                 number
   <chr>     <date>      <dbl> <chr>                <dbl>
 1 Monday    2010-01-11   952  orangeBoardings       877
 2 Tuesday   2010-01-12   796  orangeBoardings       777
 3 Wednesday 2010-01-13 1212. orangeBoardings      1203
 4 Thursday  2010-01-14 1214. orangeBoardings      1194
 5 Friday    2010-01-15 1644  orangeBoardings      1645
 6 Saturday  2010-01-16 1490. orangeBoardings      1457
 7 Sunday    2010-01-17  888. orangeBoardings       839
 8 Monday    2010-01-18  999. orangeBoardings       999
 9 Tuesday   2010-01-19 1035  orangeBoardings      1023
10 Wednesday 2010-01-20 1396. orangeBoardings      1375
# … with 13,742 more rows
```

Reshaping data from wide (fat) to long (tall): tidyr

```
long %>% count(var)
```

```
# A tibble: 12 x 2
   var                   n
   <chr>             <int>
 1 bannerAlightings   1146
 2 bannerAverage      1146
 3 bannerBoardings    1146
 4 greenAlightings    1146
 5 greenAverage       1146
 6 greenBoardings     1146
 7 orangeAlightings   1146
 8 orangeAverage      1146
 9 orangeBoardings    1146
10 purpleAlightings   1146
11 purpleAverage      1146
12 purpleBoardings    1146
```

Lab Part 1

Website

Making a separator

We will use `str_replace` from `stringr` to put periods in the names (periods are **not** special when in a replacement)

```
long = long %>% mutate(
  var = var %>%
    str_replace("Board", ".Board") %>%
    str_replace("Alight", ".Alight") %>%
    str_replace("Average", ".Average")
)
long %>% count(var)
```

```
# A tibble: 12 x 2
   var                   n
   <chr>             <int>
 1 banner.Alightings  1146
 2 banner.Average     1146
 3 banner.Boardings   1146
 4 green.Alightings   1146
 5 green.Average      1146
 6 green.Boardings    1146
 7 orange.Alightings  1146
 8 orange.Average     1146
 9 orange.Boardings   1146
10 purple.Alightings  1146
11 purple.Average     1146
12 purple.Boardings   1146
```

Reshaping data from wide (fat) to long (tall): tidyr

Now each `var` is boardings, averages, or alightings. We want to separate these so we can have these by line. Remember "." is special character:

```
long = separate(long, var, into = c("line", "type"),
                 sep = "[.]")
head(long, 2)
```

```
# A tibble: 2 x 6
  day     date        daily line   type       number
  <chr>   <date>      <dbl> <chr>  <chr>       <dbl>
1 Monday  2010-01-11    952 orange Boardings     877
2 Tuesday 2010-01-12    796 orange Boardings     777
```

```
unique(long$line)
```

```
[1] "orange" "purple" "green"  "banner"
```

```
unique(long$type)
```

```
[1] "Boardings"  "Alightings" "Average"
```

Re-uniting all the lines

If we had the opposite problem, we could use the `unite` function:

```
reunited = long %>%
  unite(col = var, line, type, sep = ".")
reunited %>% select(day, var) %>% head(3) %>% print
```

```
# A tibble: 3 x 2
  day       var
  <chr>     <chr>
1 Monday    orange.Boardings
2 Tuesday   orange.Boardings
3 Wednesday orange.Boardings
```

We could also use `paste`/`paste0`.

Lab Part 2

Website

Reshaping data from long (tall) to wide (fat): tidyr

In `tidyr`, the `spread` function spreads rows into columns. Now we have a long data set, but we want to separate the Average, Alightings and Boardings into different columns:

```
# have to remove missing days
wide = long %>% filter(!is.na(date))
wide = wide %>% spread(type, number)
head(wide)
```

```
# A tibble: 6 x 7
  day    date       daily line   Alightings Average Boardings
  <chr>  <date>     <dbl> <chr>       <dbl>   <dbl>     <dbl>
1 Friday 2010-01-15 1644  banner         NA      NA        NA
2 Friday 2010-01-15 1644  green          NA      NA        NA
3 Friday 2010-01-15 1644  orange       1643    1644      1645
4 Friday 2010-01-15 1644  purple         NA      NA        NA
5 Friday 2010-01-22 1394. banner         NA      NA        NA
6 Friday 2010-01-22 1394. green          NA      NA        NA
```

Lab Part 3

Website