PP4RS | R Module

Slot 4

Dora Simon

07.09.2018

Outline of the R-Module

Slot 1: Intro & Data Types

Slot 2: Conditionals and Functions & Loops

Slot 3: Read in Data

Slot 4: Data Manipulation

Slot 5: Regressions

Slot 6: Graphs

Slot 7: knitR

Now: Data Manipulation

Reshaping

Tidy Data

According to the 'tidy data' philosophy, datasets should follow some basic principles:

1. Every row is an observation.
2. Every column is a variable.
3. Each cell contains the value of a variable for a specific observation.

The Wide Format

- a subject's repeated responses will be in a single row
- each response is in a separate column

	County	LandArea	NatAmenity	College1970	College1980	College1990	College2000	Jobs1970	Jobs1980	Jobs1990	Jobs2000
1	Autauga	599	4	.064	.121	.145	.180	6853	11278	11471	16289
2	Baldwin	1578	4	.065	.121	.168	.231	19749	27861	40809	70247
3	Barbour	891	4	.073	.092	.118	.109	9448	9755	12163	15197
4	Bibb	625	3	.042	.049	.047	.071	3965	4276	5564	6098
5	Blount	639	4	.027	.053	.070	.096	7587	9490	11811	16503

The Wide Format

- a subject's repeated responses will be in a single row
- each response is in a separate column

Variables that do not change over time:

- land area
- presence of a natural amenity

	County	LandArea	NatAmenity	College1970	College1980	College1990	College2000	Jobs1970	Jobs1980	Jobs1990	Jobs2000
1	Autauga	599	4	.064	.121	.145	.180	6853	11278	11471	16289
2	Baldwin	1578	4	.065	.121	.168	.231	19749	27861	40809	70247
3	Barbour	891	4	.073	.092	.118	.109	9448	9755	12163	15197
4	Bibb	625	3	.042	.049	.047	.071	3965	4276	5564	6098
5	Blount	639	4	.027	.053	.070	.096	7587	9490	11811	16503

The Wide Format

- a subject's repeated responses will be in a single row
- each response is in a separate column

Variables that do not change over time:

- land area
- presence of a natural amenity

Variables that do change over time:

- Jobs
- College

They have a different variable (column) for each year.

	County	LandArea	NatAmenity	College1970	College1980	College1990	College2000	Jobs1970	Jobs1980	Jobs1990	Jobs2000
1	Autauga	599	4	.064	.121	.145	.180	6853	11278	11471	16289
2	Baldwin	1578	4	.065	.121	.168	.231	19749	27861	40809	70247
3	Barbour	891	4	.073	.092	.118	.109	9448	9755	12163	15197
4	Bibb	625	3	.042	.049	.047	.071	3965	4276	5564	6098
5	Blount	639	4	.027	.053	.070	.096	7587	9490	11811	16503

The Long Format

Each row is one time point per subject.

- each subject (county) will have data in multiple rows
- any variables that do not change across time will have the same value in all the rows
- we no longer need four columns for either Jobs or College
- to keep track of which observation occurred in which year, we need to add a variable, `Year`

	County	LandArea	NatAmenity	Year	College	Jobs
1	Autauga	599	4	1970	.064	6853
2	Autauga	599	4	1980	.121	11278
3	Autauga	599	4	1990	.145	11471
4	Autauga	599	4	2000	.180	16289
5	Baldwin	1578	4	1970	.065	19749
6	Baldwin	1578	4	1980	.121	27861
7	Baldwin	1578	4	1990	.168	40809
8	Baldwin	1578	4	2000	.231	70247
9	Barbour	891	4	1970	.073	9448
10	Barbour	891	4	1980	.092	9755
11	Barbour	891	4	1990	.118	12163
12	Barbour	891	4	2000	.109	15197
13	Bibb	625	3	1970	.042	3965
14	Bibb	625	3	1980	.049	4276
15	Bibb	625	3	1990	.047	5564
16	Bibb	625	3	2000	.071	6098
17	Blount	639	4	1970	.027	7587
18	Blount	639	4	1980	.053	9490
19	Blount	639	4	1990	.070	11811
20	Blount	639	4	2000	.096	16503

When to use which?

Wide

When to use which?

Wide

- unit of analysis is the subject

When to use which?

Wide

- unit of analysis is the subject
- good for visualization and human understanding

When to use which?

Wide

- unit of analysis is the subject
- good for visualization and human understanding

Long

When to use which?

Wide

- unit of analysis is the subject
- good for visualization and human understanding

Long

- unit of analysis is each measurement occasion for each subject

When to use which?

Wide

- unit of analysis is the subject
- good for visualization and human understanding

Long

- unit of analysis is each measurement occasion for each subject
- good for data analysis

When to use which?

Wide

- unit of analysis is the subject
- good for visualization and human understanding

Long

- unit of analysis is each measurement occasion for each subject
- good for data analysis
- use each decade's college education rate as a covariate for the same decade's Jobs value

When to use which?

Wide

- unit of analysis is the subject
- good for visualization and human understanding

Long

- unit of analysis is each measurement occasion for each subject
- good for data analysis
- use each decade's college education rate as a covariate for the same decade's Jobs value

You can switch from one format to the other using the `tidyr` or the older `reshape2` package.

We will only use the `tidyr` package, more on the difference to `reshape2` can be found here.

The tidyr package

- `gather`: use it when you have columns that are not variables (wide to long)
- `spread`: opposite of gather (long to wide)
- `separate`: split a column into multiple
- `extract`: like separate, using regexp groups
- `unite`: opposite of extract/separate

Tibbles

- Tibbles are data frames
- two main differences: printing and subsetting

Tibbles

- Tibbles are data frames
- two main differences: printing and subsetting

Printing

- shows only the first 10 rows
- shows all the columns that fit on screen
- makes it much easier to work with large data

Tibbles

- Tibbles are data frames
- two main differences: printing and subsetting

Printing

- shows only the first 10 rows
- shows all the columns that fit on screen
- makes it much easier to work with large data

- `df$x`
- `df[["x"]]`
- `df[[1]]`

Tibbles do not allow you to do partial matching!

If your variable is called `x1`, a data frame will take the value `x1`. A tibble will give you a warning.

Piping

> The magrittr (to be pronounced with a sophisticated french accent) is a package with two aims: to decrease development time and to improve readability and maintainability of code. Or even shortr: to make your code smokin' (puff puff)![1]

Piping

> The magrittr (to be pronounced with a sophisticated french accent) is a package with two aims: to decrease development time and to improve readability and maintainability of code. Or even shortr: to make your code smokin' (puff puff)![1]

- helps for clearly expressing a sequence of multiple operations
- "pipe"-like operator %>%
- pipe a value forward into an expression or function call[2]

Piping

> The magrittr (to be pronounced with a sophisticated french accent) is a package with two aims: to decrease development time and to improve readability and maintainability of code. Or even shortr: to make your code smokin' (puff puff)![1]

- helps for clearly expressing a sequence of multiple operations
- "pipe"-like operator `%>%`
- pipe a value forward into an expression or function call[2]

Instead of `f(x)`, write `x %>% f`

[1] Source

[2] If you want to know more about when to pipe and when not to pipe, look here

Exercises

Define the following dataset and have a first look.

```
messy <- data.frame(id = 1:4,
                    trt = sample(rep(c('control', 'treatment'),
                                     each = 2)),
                    work_T1 = runif(4),
                    home_T1 = runif(4),
                    work_T2 = runif(4),
                    home_T2 = runif(4))
```

We have measurements of how much time people spend on their phones, measured at two locations (work and home), at two times.

1. Which format is the data in? Create a new dataset that has the other format.
2. The column `key` still contains information that we want to have in separate variables. Create two different columns out of it that contain the location and time of the experiment.
3. Gather and separate the messy data into a tidy dataset using the pipe operator `%>%`.

Solutions

Define the following dataset and have a first look.

```
messy <- data.frame(id = 1:4,
                    trt = sample(rep(c('control', 'treatment'),
                                     each = 2)),
                    work_T1 = runif(4),
                    home_T1 = runif(4),
                    work_T2 = runif(4),
                    home_T2 = runif(4))
```

Solutions

Have a look at the data. Which format is it in? Create a new dataset that has the other format.

Solutions

The column `key` still contains information that we want to have in separate variables. Create two different columns out of it that contain the location and time of the experiment.

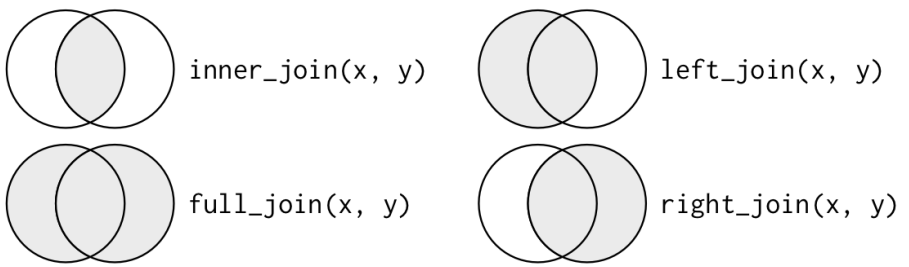
Solutions

Gather and separate the messy data into a tidy dataset using the pipe operator %>%.

Combine and Summarize Datasets

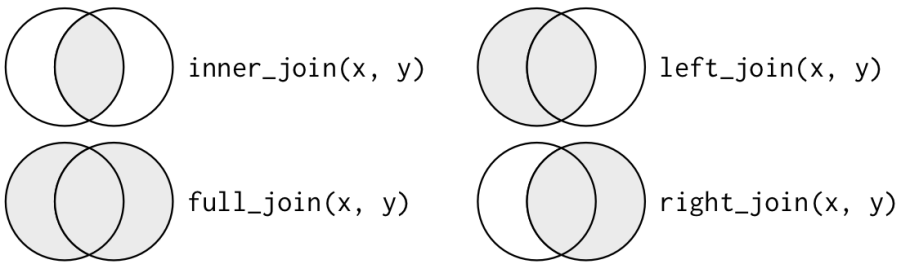
Join two datasets

- `full_join`: joins the data from both dataframes and keeps all observations
- `inner_join`: joins the data from both dataframes but only keeps those observations which exist in both dataframes



Join two datasets

- `full_join`: joins the data from both dataframes and keeps all observations
- `inner_join`: joins the data from both dataframes but only keeps those observations which exist in both dataframes
- `left_join`: matches rows from the second dataframe to the first dataframe. All rows from the first dataframe are kept
- `right_join`: opposite of `left_join`



Join examples

- dataset `x`: Data on country, gdp, exports for CH and HU
- dataset `y`: Data on country, population for CH and DE

full join

country	gdp	exports	population
CH	10000	300	2
HU	300	20	NA
DE	NA	NA	2

Join examples

- dataset `x`: Data on country, gdp, exports for CH and HU
- dataset `y`: Data on country, population for CH and DE

full join

country	gdp	exports	population
CH	10000	300	2
HU	300	20	NA
DE	NA	NA	2

inner join

country	gdp	exports	population
CH	10000	300	2

Join examples

- dataset `x`: Data on country, gdp, exports for CH and HU
- dataset `y`: Data on country, population for CH and DE

left join: keeps all rows from first dataset

country	gdp	exports	population
CH	10000	300	2
HU	300	20	NA

Join examples

- dataset `x`: Data on country, gdp, exports for CH and HU
- dataset `y`: Data on country, population for CH and DE

left join: keeps all rows from first dataset

country	gdp	exports	population
CH	10000	300	2
HU	300	20	NA

right join: keeps all rows from second dataset

country	gdp	exports	population
CH	10000	300	2
DE	NA	NA	2

Binding

Binding is helpful if you want to append the observations or variables of one dataset to another dataset.

- `bind_rows`: appends the the second dataframe to the first dataframe as new rows
- `bind_cols`: appends the the second dataframe to the first dataframe as new columns

Summarize Data

Summary tables

- `summarise` from `dplyr`: applies an aggregate function to the whole dataset
- `group_by`: groups data for further operations
- `ungroup`: ungroups data

Summary tables

- `summarise` from `dplyr`: applies an aggregate function to the whole dataset
- `group_by`: groups data for further operations
- `ungroup`: ungroups data

Aggregate functions:

- `mean(x)`: create group-wise mean for variable `x`
- `sum(x)`: create group-wise sum for variable `x`
- `median(x)`: create group-wise median for variable `x`
- `sd(x)`: group-wise standard deviation for variable `x`
- `min(x)`: minimum by group for variable `x`
- `max(x)`: see above
- `quantile(x, v)`: vth quantile for variable `x` by group
- `first(x)`: first x-value in group by group
- `last(x)`: last x-value in group by group
- `n()`: number of values in group by group
- `n_distinct(x)`: number of distinct values in group by group

Exercises

Define the following two datasets:

```
library(tibble)
first_df <- tibble(
  country = c('Afghanistan', 'Belgium', 'China', 'Denmark'),
  population = c(33369945, 11371928, 1382323332, 5690750),
  continent = c("Asia", "Europe", "Asia", "Europe")
)

# gdp in millions
second_df <- tibble(
  country = c('Afghanistan', 'Belgium', 'Denmark', 'Germany'),
  gdp = c(35146, 422809, 211916, 3232545)
)
```

Exercises

Define the following two datasets:

```
third_df <- tibble(
  country = c('Afghanistan', 'Belgium', 'China', 'Denmark'),
  population = c(33369945, 11371928, 1382323332, 5690750),
  continent = c("Asia", "Europe", "Asia", "Europe")
)

# gdp in millions
fourth_df <- tibble(
  country_name = c('Afghanistan', 'Belgium', 'Denmark', 'Germany'),
  gdp = c(35146, 422809, 211916, 3232545)
)
```

Exercises

Define the following two datasets:

```
fifth_df <- tibble(
  country = c('Switzerland', 'India', 'Japan', 'Spain'),
  population = c(8431702, 1320880000, 126750000, 46528966),
  continent = c("Europe", "Asia", "Asia", "Europe")
)
```

Exercises

1. Add the observation `Germany` to the first data frame.
2. Join the third and the fourth data sets and keep all observations.
3. Append the fifth dataset to the first dataset.
4. Get the average age of the subjects in the affairs data set.
5. Get the average age of women with, women without, men with and men without children in the affairs data set.

Solution

Add the observation `Germany` to the first data frame.

Solution

Join the third and the fourth data sets and keep all observations.

Solution

Append the fifth dataset to the first dataset.

Solution

Get the average age of all the subjects in the affairs data set.

Solution

Get the average age of women with, women without, men with and men without children in the affairs data set.