Untitled

Cody

October 5, 2017

Choose your own adventure Pick one of the data reshaping prompts and do it.

Pick a join prompt and do it.

It is fine to work with a new dataset and/or create variations on these problem themes.

General data reshaping and relationship to aggregation Problem: You have data in one "shape" but you wish it were in another. Usually this is because the alternative shape is superior for presenting a table, making a figure, or doing aggregation and statistical analysis.

Solution: Reshape your data. For simple reshaping, gather() and spread() from tidyr will suffice. Do the thing that it possible / easier now that your data has a new shape.

Prompts:

Activity #1

Make you own cheatsheet similar to Tyler Rinker's minimal guide to tidyr.

Activity #2

Make a tibble with one row per year and columns for life expectancy for two or more countries. Use knitr::kable() to make this table look pretty in your rendered homework. Take advantage of this new data shape to scatterplot life expectancy for one country against that of another.

Activity #3

Compute some measure of life expectancy (mean? median? min? max?) for all possible combinations of continent and year. Reshape that to have one row per year and one variable for each continent. Or the other way around: one row per continent and one variable per year. Use knitr::kable() to make these tables look pretty in your rendered homework. Is there a plot that is easier to make with the data in this shape versis the usual form? If so (or you think so), try it! Reflect.

Activity #4

In Window functions, we formed a tibble with 24 rows: 2 per year, giving the country with both the lowest and highest life expectancy (in Asia). Take that table (or a similar one for all continents) and reshape it so you have one row per year or per year * continent combination.

Activity #5

Previous TA Andrew MacDonald has a nice data manipulation sampler. Make up a similar set of exercises for yourself, in the abstract or (even better) using Gapminder or other data, and solve them.

Join, merge, look up Problem: You have two data sources and you need info from both in one new data object.

Solution: Perform a join, which borrows terminology from the database world, specifically SQL.

Prompts:

Activity #1

Create a second data frame, complementary to Gapminder. Join this with (part of) Gapminder using a dplyr join function and make some observations about the process and result. Explore the different types of joins. Examples of a second data frame you could build: One row per country, a country variable and one or more variables with extra info, such as language spoken, NATO membership, national animal, or capitol city. If you really want to be helpful, you could attempt to make a pull request to resolve this issue, where I would like to bring ISO country codes into the gapminder package. One row per continent, a continent variable and one or more variables with extra info, such as northern versus southern hemisphere.

Activity #2

Create your own cheatsheet patterned after mine but focused on something you care about more than comics! Inspirational examples: Pets I have owned + breed + friendly vs. unfriendly + ??. Join to a table of pet breed, including variables for furry vs not furry, mammal true or false, etc. Movies and studios.... Athletes and teams.... You will likely need to iterate between your data prep and your joining to make your explorations comprehensive and interesting. For example, you will want a specific amount (or lack) of overlap between the two data.frames, in order to demonstrate all the different joins. You will want both the data frames to be as small as possible, while still retaining the expository value.

Activity #3

This is really an optional add-on to either of the previous activities. Explore the base function merge(), which also does joins. Compare and contrast with dplyr joins. Explore the base function match(), which is related to joins and merges, but is really more of a "table lookup". Compare and contrast with a true join/merge.