Exploring data #3

Follow-ups

"Giving Open-Source Projects Life After a Developer's Death", Wired Magazine

"Where's _why?", slate.com

forcats

Hadley Wickham has developed a package called `forcats` that helps you work with categorical variables (factors). I'll show some examples of its functions using the `worldcup` data set:

```
library(forcats)
library(faraway)
data(worldcup)
```

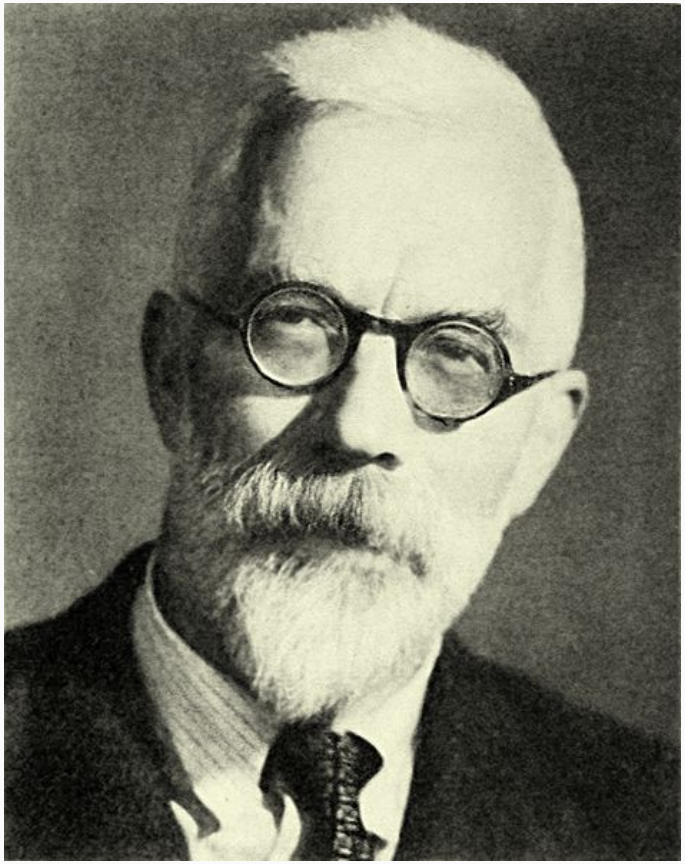
forcats

The `fct_recode` function can be used to change the labels of a function (along the lines of using `factor` with `levels` and `labels` to reset factor labels).

One big advantage is that `fct_recode` lets you change labels for some, but not all, levels. For example, here are the team names:

Simulations

The lady tasting tea



Source: Flikr commons, https://www.flickr.com/photos/internetarchivebookimages/20150531109/

Other computationally-intensive approaches

Bootstrap and friends

- **Bootstraping:** Sample the data set with replacement and re-estimate the statistical parameter(s) each time.
- **Jackknifing:** Rake out one observation at a time and re-estimate the statistical parameter(s) with the rest of the data.
- **Permutation tests:** See how unusual the result from the data is compared to if you shuffle your data (and so remove any relationship in observed data between variables).
- **Cross-validation:** See how well your model performs if you pick a subset of the data, build the model just on that subset, and then test how well it predicts for the rest of the data, and repeat that many times.

Bayesian analysis

Suggested books for learning more about Bayesian analysis in R:

- *Doing Bayesian Data Analysis, Second Edition: A Tutorial with R, JAGS, and Stan* - John Kruschke

Speeding up R

When to worry about this

Not until you need it!

$$\text{Total time} = \text{Time to write code} + \text{Time to run code}$$

Optimizing code means it takes longer to write the code. Often, even if it's slow to compute, it's still faster overall to not take the time to write faster code.

When to worry about this

- Data with $>$ 1,000,000 observations (memory + code speed)
- Running complex models many, many times (code speed)
- Large data created by the analytic methods (memory + code speed)

Compiled code

Compiled languages:

Working with large datasets

`data.table`

`data.table` is a package in R that can efficiently read in and manipulate large data sets. It offers a **substantial** speed improvement over the classic `data.frame` when working with large data sets.

Example: US precipitation

As an example, I have a file with daily precipitation measures for every US county from 1979 through 2011:

- 365 days * 33
- ~3,000 counties

This file has $>$ 37,000,000 lines. The total file size is 2.26 GB.

Reading in a large text file

`fread` is the `data.table` equivalent of the `read.table` family of functions:

```
library(data.table)
```