Introduction to Computational Tools and Techniques in Social Science

Jae Yeon Kim

06 December, 2018

About me

- Substantially, interested in historical comparative approaches to the study of race, ethnicity, and politics
- Methodologically, interested in using natural language processing and text analysis as a way of creating new data in race scholarship
- Also, a former tech industry person married to an engineer

Motivation

Global search popularity for 'data science'
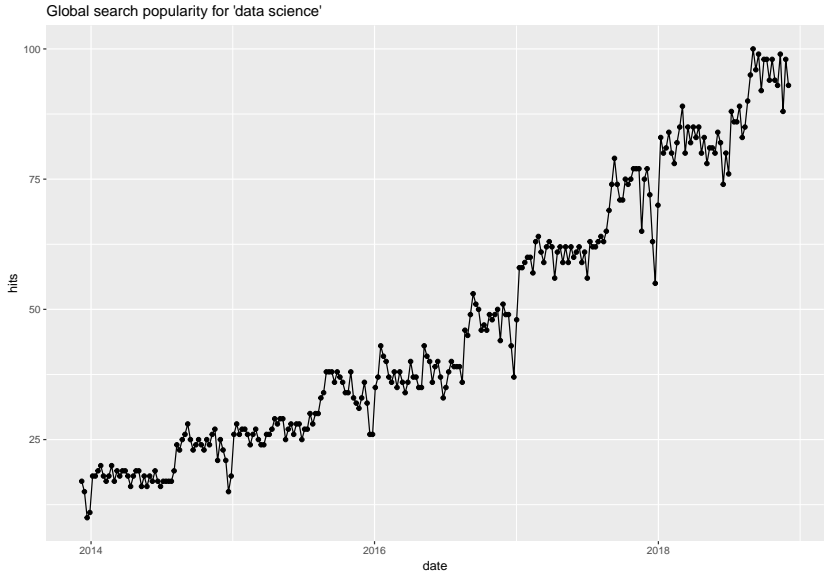
- ▶ Why should we care?
- ▶ Yes, big data is a trend.
- ▶ But being good at computational tools and techniques has more immediate benefits.
- ▶ It can make your life **easy** and **organized**.
  - ▶ Don't repeat yourself: AUTOMATE!

- In addition, there are new tools for
  - Data collection (e.g., APIs, webscraping)
  - Analysis (e.g., machine learning)
  - Visualization (e.g., maps, social networks)
- In sum, you can do cool stuff.

- ▶ But it takes some **efforts** to take advantages of these new tools.
    - ▶ You need to learn how to code **a little bit**.
    - ▶ However, learning on your own is inefficient.
    - ▶ More important, you can get **bad** habits.

▶ The following examples are adapted from
  https://style.tidyverse.org

```r
# Good
fit_models.R

if (y < 0 && debug) {
  message("y is negative")
}

# Bad
fit models.R

if (y < 0 && debug)
message("Y is negative")
```

```r
# Good
do_something_very_complicated(
  something = "that",
  requires = many,
  arguments = "some of which may be long"
)

# Bad
do_something_very_complicated("that", requires, many, argum
                                  "some of which may be long"
                                  )
```

Objectives

▶ Tasting a wide range of computational tools
▶ Getting programming fundamentals right
  ▶ Concepts
  ▶ Techniques
▶ Learning by doing
  ▶ Learning from your own MANY trials and errs
  ▶ Learning from others (please, do Google search before asking me)

- Coding is similar to **cooking**.
  - So many different cuisines (programming languages).
  - But there are fundamentals.
    - Ingredients (data)
    - Techniques (logic)
    - Recipes (workflow)

- ▶ Bad habits are **bad**.
    - ▶ Rule 1. Thou shall comment.
    - ▶ Rule 2. Thou shall reuse functions (no copy and paste).
    - ▶ Rule 3. Thou shall practice version control (no final_final_final.Rmd).

- ▶ **Learn to learn**
- ▶ Specifically, we are going to learn:
  - ▶ How to use the command line in a UNIX environment
  - ▶ How to use Git to do version control
  - ▶ How ch use R to clean, wrangle, analyze, and visualize data (with a focus on Wickham's tidy data principles)
  - ▶ How to use R/Python to parse HTML, CSS, and Javascript for webscraping and (a bit of) Qualtrics (by Julia)
  - ▶ How to use R/Python to do text analysis and machine learning (guest lectures by two Berkeley Institute for Data Science fellows)

- ▶ **Don't expect**
  - ▶ Becoming a data scientist within one semester
  - ▶ I can answer all of your questions
- ▶ We focus on learning **how to learn**.
  - ▶ Programming is one endless Google Search (aka "Rochelle's Law")

▶ Using Excel:
  ▶ 3 mins for copying, pasting, and reorganizing one article
  ▶ 80,000 newspaper articles
  ▶ Taking **4,000** hours or **166 days**

▶ Using python:
  ▶ A few hours for coding
  ▶ Less than 5 mins for creating the dataset
  ▶ Also, the code is reusable.

```python
In [1]: def parsing_proquest(x):

            # load libs

            from bs4 import BeautifulSoup

            import re

            # load file
            soup = BeautifulSoup(open(x,"r"), 'html.parser')

            # filter by strong tag
            doc = soup('strong')

            # save filtered results to new objects

            doc.text = soup.findAll(text=re.compile('Full text:'))
            doc.date = soup.findAll(text=re.compile('Publication year:'))
            doc.source = soup.findAll(text=re.compile('Publication date:'))
            doc.author = soup.findAll(text=re.compile('Publication info:'))
```

Previous final projects by students
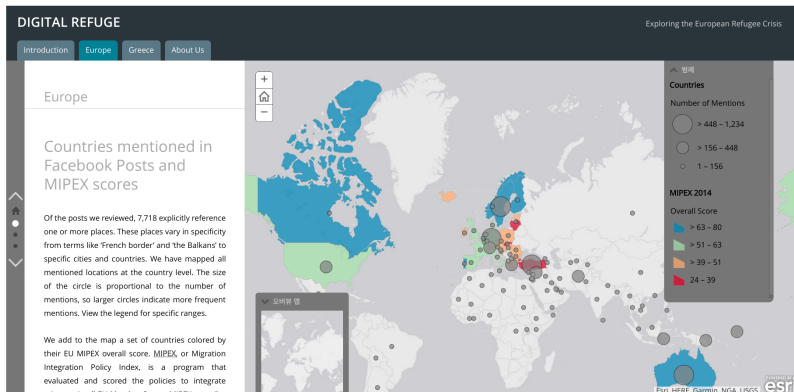
Focus on best practice

▶ Good habits are **good**.
  ▶ Commenting serves you and many other people.
  ▶ Reusing functions provides opportunities to learn and clean up your mess.
  ▶ Practicing version control is how we become a mature researcher and a coder.

Class

- ▶ Participation (25%)
  - ▶ Be nice to each other. We're all learning (especially me).
- ▶ Homework (50%)
  - ▶ Every week.
  - ▶ Learning how to code is like learning how to drive.
- ▶ Final project (25%)
  - ▶ Feasibility is your friend. Late Feb proposal, April presentations.

Logistics

- Learning by doing
- Pair-programming on in-class challenges
- Section is required
- Julia Christensen is a technical assistant to the course.

Special thanks

- Laura Stoker (University of Berkeley, California) for supporting this course at its initial developmental stage
- Rochelle Terman (University of Chicago) for creating this course
- Rachel Bernhard (University of Oxford, UC Davis) for continuing this course