Managing and Manipulating Data Using R
Lecture 1.1

Karina Salazar

1 Student introductions

Student introductions

1. Preferred name
2. Preferred pronouns
3. Academic program (and how far along)
4. GA, RA, TA, and/or job?
5. Why are you interested in this course?

2 About your instructor

Karina Salazar, instructor

My start in data management/statistical analysis
- SPSS
  - ▷ evaluated retention programs within institutional research and assessment offices
  - ▷ student-level data on math remediation courses
  - ▷ College Academy for Parents, Think Tank, Assessment Institute
- Stata
  - ▷ used loops and user-defined functions to work with national datasets (IPEDS, Survey of Earned Doctorates)

Got sick of the limitations of survey data and/or available data
- No survey asked questions on what I was interested in
  - ▷ universities pledge commitment to access, but enrollments don't tell the whole story
  - ▷ who do they actually recruit?
- We realized "data science" could create data from publicly available data sources
  - ▷ Twitter
  - ▷ travel schedules on admissions websites

Recruiting research program and "data science"

- Python
  - ▷ web-scraping
  - ▷ connecting to Application Program Interfaces (API) (e.g., census data, Twitter, LinkedIn)
  - ▷ Natural Language Processing
- R
  - ▷ R can do all "data science" tasks Python can
  - ▷ R can do all statistical analyses that Stata can (and more!)
  - ▷ R has amazing mapping capabilities

Examples:
- The off-campus recruiting project
- Dissertation Defense

3 What is R

What is R

According to the Inter-university consortium for political and social research (ICPSR):

> R is "an alternative to traditional statistical packages such as SPSS, SAS, and Stata such that it is an extensible, open-source language and computing environment for Windows, Macintosh, UNIX, and Linux platforms. Such software allows for the user to freely distribute, study, change, and improve the software under the *Free Software Foundation's GNU General Public License.*"

- For more info visit R-project.org

Base R vs. R packages

There are "default" packages that come with R. Some of these include:

- `as.character`

- `print`

- `setwd`

And there are R packages developed and shared by others. Some R packages include:

- `tidyverse`

- `stargazer`

- `foreign`

more about these in later weeks...

Installing and Loading R packages

You only need to install a package once. To install an R package use `install.package()` function.

```
#install.packages("tidyverse")
```

However, you need to load a package everytime you plan to use it. To load a package use the `library()` function.

```
library(tidyverse)
#> -- Attaching packages ---------------------------------------------
#> v ggplot2 3.2.1      v purrr   0.3.2
#> v tibble  2.1.3      v dplyr   0.8.3
#> v tidyr   1.0.0      v stringr 1.4.0
#> v readr   1.3.1      v forcats 0.4.0
#> -- Conflicts ------------------------------------------------------
#> x dplyr::filter() masks stats::filter()
#> x dplyr::lag()    masks stats::lag()
```

RStudio

"RStudio is an integrated development environment (IDE) for R. It includes a console, syntax-highlighting editor that supports direct code execution, as well as tools for plotting, history, debugging and workspace management."
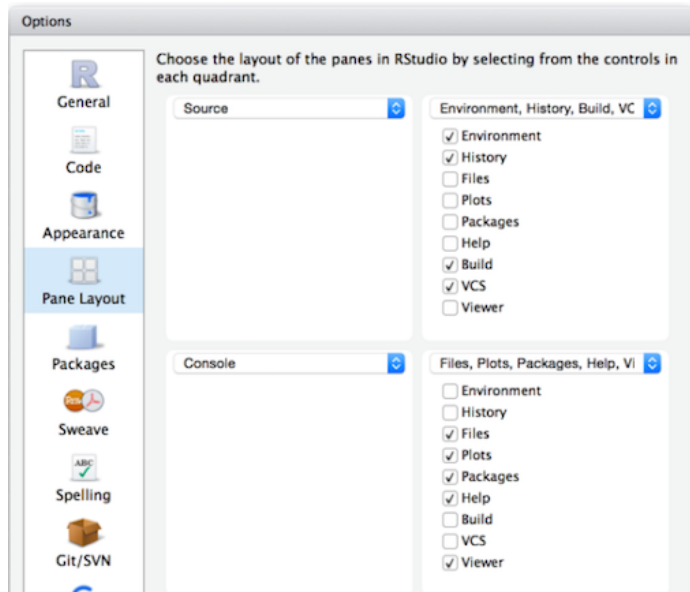
R Markdown

R Markdown produces dynamic output formats in html, pdf, MS Word, dashboards, Beamer presentations, etc.
  ○ We will be using R Markdown for lectures and homeworks.

Why learn R? R can do a lot of stuff!

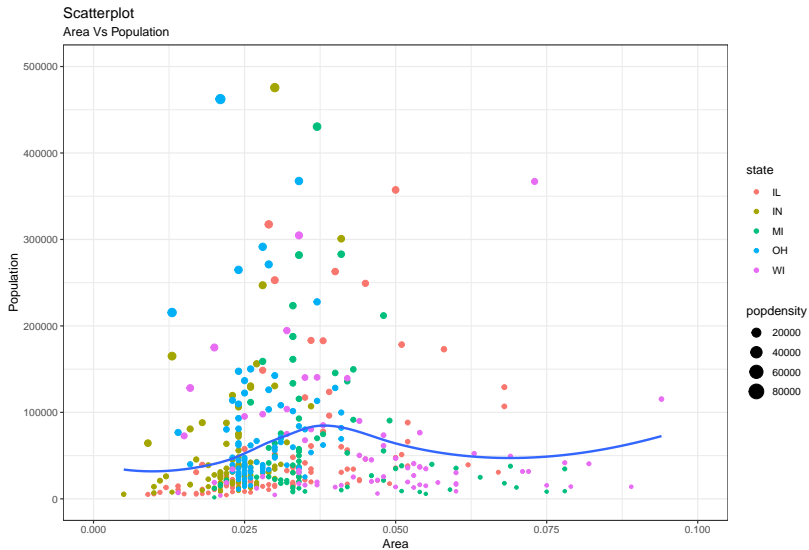How we have used R+RStudio+RMarkdown in our research team
- Stuff traditional statistical software (e.g., SPSS, Stata) can do
  - ▷ Data manipulation, creating analysis datasets
  - ▷ Descriptive statistics and statistical models
  - ▷ Graphs
- Stuff traditional statistical software cannot do
  - ▷ Static policy reports
  - ▷ Static presentations
    - — All lectures for this class written in RMarkdown
  - ▷ Interactive presentations
  - ▷ Interactive maps
  - ▷ Interactive dashboards
  - ▷ Interactive graphs

Some of the other stuff R can create/do:
- Websites; journals; books; web- scraping; network analysis; machine learning/artificial intelligence

Graphs

- Create graphs with ggplot2 package

Scatterplot
Area Vs Population

Source: midwest

Journal articles

- Journal articles with rticles package

Interactive web apps

- Interactive web apps with shiny package
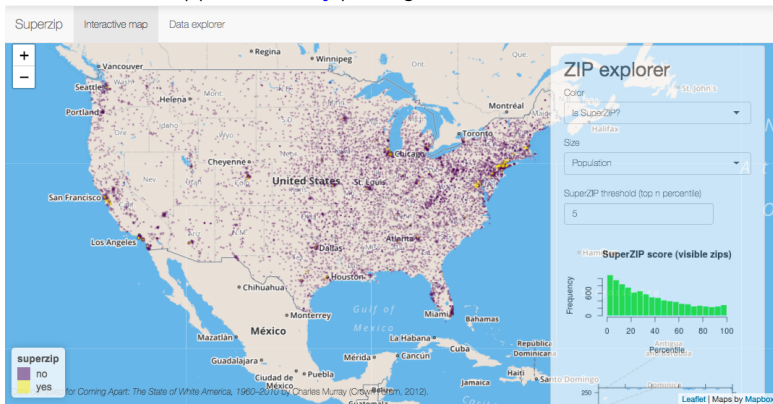
Mapping

○ Mapping with sf package & ggplot

Area of counties in North Carolina
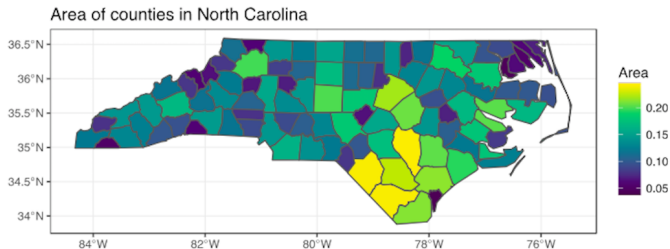
4 What is this course about?

What is data management?

- All the stuff you have to do to create analysis datasets that are ready to analyze:
  - ▷ collect data
  - ▷ read/import data into statistical programming language
  - ▷ clean data
  - ▷ integrate data from multiple sources (e.g, join/merge, append)
  - ▷ change organizational structure of data so it is suitable for analysis
  - ▷ create "analysis variables" from "input variables"
  - ▷ Make sure that you have created analysis variables correctly

Why I don't call this class "R for data science"

Learn to walk before you can run!

- "data science" implies doing "fancy" things like mapping, network analysis, web-scraping, etc.

- But if you don't know how to clean data, these "fancy" analyses and visualizations will be useless

- "80% of data science is data cleaning"

- The skills you learn in this data management class are foundational to data science tasks! (and a prerequisite to taking data science seminar)

Who is this class for?

This class is for anyone who wants to work with data, that is people who want to be:

- researchers working with survey data and doing traditional statistical analyses
- researchers who want to do "data science" oriented research involving
- analysts working at think tanks or non-profits
- "Data scientists"

5 Course logistics

Course logistics

- follow the syllabus

6 Create "R project" and directory structure

What is an R project? Why are you doing this?

What is an "R project"?
- helps you keep all files for a project in one place
- When you open an R project, the file-path of your current working directory is automatically set to the file-path of your R-project

Why am I asking you to create R project and download a specific directory structure?
- I want you to be able to run the .Rmd files for each lecture on your own computer
- Sometimes these .Rmd files point to certain sub-folders
- If you create R project and create directory structure I recommend, you will be able to run .Rmd files from your own computer without making any changes to file-paths!

Follow these steps to create "R project" and directory structure

1. Download this zip folder: LINK HERE
   ▷ Unzip the folder: this is a shell of the file directory you should use for this class
   ▷ Change the name to "rclass"
   ▷ Move it to your preferred location (e.g, documents, desktop, dropbox, etc)
2. In RStudio, click on "File" » "New Project" » "New Directory" » "New Project"
   ▷ In "Directory name", type "rclass_project" as the title of the Rproject for the course
   ▷ In "Create project as subdirectory of", click browse and:
      — save the R Project within the rclass folder (same general folder as data and lectures)
3. Save the following files in "rclass/lectures/lecture1"
   ▷ lecture1.1_ua.Rmd
   ▷ lecture1.1_ua.pdf
   ▷ lecture1.2_ua.Rmd
   ▷ lecture1.2_ua.Pdf
   ▷ lecture1.2_ua.R

After you follow these steps

- you can add any additional sub-folders you want to the "rclass" folder
  - ▷ e.g., "syllabus", "resources"
- You can add any additional files you want to the sub-directory folders you unzipped
  - ▷ e.g., in "rclass/lectures/lecture1" you might add an additional document of notes you took

7 Directories and filepaths

Working directory

(Current) Working directory

- the folder/directory in which you are currently working
- this is where R looks for files
- Files located in your current working directory can be accessed without specifying a filepath because R automatically looks in this folder

Function `getwd()` shows current working directory

```
getwd()
#> [1] "C:/Users/ozanj/Documents/rclass/lectures/lecture1"
```

Command `list.files()` lists all files located in working directory

```
getwd()
#> [1] "C:/Users/ozanj/Documents/rclass/lectures/lecture1"
list.files()
#>  [1] "data-structures-overview.png" "images.zip"
#>  [3] "lecture1.1.pdf"               "lecture1.1.Rmd"
#>  [5] "lecture1.1.tex"               "lecture1.1_files"
#>  [7] "lecture1.1_ua.pdf"            "lecture1.1_ua.Rmd"
#>  [9] "lecture1.1_ua.tex"            "lecture1.1_ua_files"
#> [11] "lecture1.2.pdf"               "lecture1.2.R"
#> [13] "lecture1.2.Rmd"               "lecture1.2.tex"
#> [15] "lecture1.2_ua.pdf"            "lecture1.2_ua.Rmd"
#> [17] "lecture1.pdf"                 "lecture1.Rmd"
#> [19] "lecture1.tex"                 "pane_layout.png"
#> [21] "problemset1.pdf"              "problemset1.Rmd"
#> [23] "problemset1_solutions.pdf"    "problemset1_solutions.Rmd"
```

Working directory, "Code chunks" vs. "console" and "R scripts"

When you run **code chunks** in RMarkdown files (.Rmd), the working directory is set to the filepath where the .Rmd file is stored

```
getwd()
#> [1] "C:/Users/ozanj/Documents/rclass/lectures/lecture1"
list.files()
#>  [1] "data-structures-overview.png" "images.zip"
#>  [3] "lecture1.1.pdf"               "lecture1.1.Rmd"
#>  [5] "lecture1.1.tex"               "lecture1.1_files"
#>  [7] "lecture1.1_ua.pdf"            "lecture1.1_ua.Rmd"
#>  [9] "lecture1.1_ua.tex"            "lecture1.1_ua_files"
#> [11] "lecture1.2.pdf"               "lecture1.2.R"
#> [13] "lecture1.2.Rmd"               "lecture1.2.tex"
#> [15] "lecture1.2_ua.pdf"            "lecture1.2_ua.Rmd"
#> [17] "lecture1.pdf"                 "lecture1.Rmd"
#> [19] "lecture1.tex"                 "pane_layout.png"
#> [21] "problemset1.pdf"              "problemset1.Rmd"
#> [23] "problemset1_solutions.pdf"    "problemset1_solutions.Rmd"
#> [25] "rticles.png"                  "sample_hw_oj.html"
#> [27] "sample_hw_oj.Rmd"             "sf.png"
#> [29] "shiny.png"
```

When you run code from the **R Console** or an **R Script**, the working directory is….

Command `getwd()` shows current working directory

```
getwd()
#> [1] "C:/Users/ozanj/Documents/rclass/lectures/lecture1"
```

Absolute vs. relative filepath

Absolute file path: The absolute file path is the complete list of directories needed to locate a file or folder.

```
setwd("Users/Karina/rclass/lectures/lecture2")
```

Relative file path: The relative file path is the path relative to your current location/directory. Assuming your current working directory is in the "lecture2" folder and you want to change your directory to the data folder, your relative file path would look something like this:

```
setwd("../../data")
```

File path shortcuts

Key	Description
~	tilde is a shortcut for user's home directory (mine is my name)
../	moves up a level
../../	moves up two level