Data Science Course in a Box

Mine Çetinkaya-Rundel

2018-05-17

Contents

Welcome

Some welcome words...

Chapter 1

Introduction

This is not a book per se, at least not yet. It's a place for organizing materials for teaching (and learning) data science with R, RStudio, the tidyverse and tidyverse friendly packages. It's called **Data Science Course in a Box**, as it contains all the materials you (an educator) might need to teach data science or you (a learner) might find useful to learn about them.

1.1 Who is this course for?

The materials in this box are designed for learners who have no background in data science, statistics, or programming. However, they also assume that the learners are interested in making sense of (sometimes messy) data and willing to dive into the documentation of the packages we introduce.

1.2 What is in the box?

- Slides
- Labs
- Assignments
- Exams
- Infrastructure guide

1.3 How is the course content organized?

- Part 1: Exploring data - wrangle + visualize + collect
- Part 2: Making rigorous conclusions - modeling + inference
- Part 3: Looking forward - …

1.4 Why R?

Unlike most other software designed specifically for teaching statistics, R is free and open source, powerful, flexible, and relevant beyond the introductory statistics classroom. Arguments against using and teaching R at especially the introductory statistics level generally cluster around the following two points: teaching

programming in addition to statistical concepts is challenging and the command line is more intimidating to beginners than the graphical user interface (GUI) most point-and-click type software offer.

One solution for these concerns is to avoid hands-on data analysis completely. If we do not ask our students to start with raw data and instead always provide them with small, tidy rectangles of data then there is never really a need for statistical software beyond spreadsheet or graphing calculator. This is not what we want in a modern statistics course and is a disservice to students.

Another solution is to use traditional point-and-click software for data analysis. The typical argument is that the GUI is easier for students to learn and so they can spend more time on statistical concepts. However, this ignores the fact that these software tools also have nontrivial learning curves. In fact, teaching specific data analysis tasks using such software often requires lengthy step-by-step instructions, with annotated screenshots, for navigating menus and other interface elements. Also, it is not uncommon that instructions for one task do not easily extend to another. Replacing such instructions with just a few lines of R code actually makes the instructional materials more concise and less intimidating.

Many in the statistics education community are in favor of teaching R (or some other programming language, like Python) in upper level statistics courses, however the value of using R in introductory statistics courses is not as widely accepted. We acknowledge that this addition can be burdensome, however we would argue that learning a tool that is applicable beyond the introductory statistics course and that enhances students' problem solving skills is a burden worth bearing.

1.5 Why RStudio?

The RStudio IDE includes a viewable environment, a file browser, data viewer, and a plotting pane, which makes it less intimidating than the bare R shell. Additionally, since it is a full fledged IDE, it also features integrated help, syntax highlighting, and context-aware tab completion, which are all powerful tools that help flatten the learning curve.

Students access the RStudio IDE through a centralized RStudio server instance, which allows us to provide students with uniform computing environments. Additionally, RStudio's direct integration with other critically important tools for teaching computing best practices and reproducible research also influenced our decision for making it central in our toolkit.

It should be noted that we do not want to completely dissuade students from downloading and installing R and RStudio locally, we just do not want it to be a prerequisite for getting started. We have found that teaching personal setup is best done progressively throughout a semester, usually via one-on-one interactions during office hours or after class. Our goal is that all students will be able to continue using R even if they no longer have access to departmental resources.

Chapter 2

Setup

There are four things you need to run the code in this book: R, RStudio, a collection of R packages called the **tidyverse**, and a handful of other packages. Packages are the fundamental units of reproducible R code. They include reusable functions, the documentation that describes how to use them, and sample data.

2.1 On the Cloud

You can access all on the cloud, via RStudio Cloud, and avoid local installation.

See RStudio Cloud workspace that you can copy (soon?) for your course.

2.2 Locally

[SOME OF WHAT'S BELOW IS BORROWED FROM R4DS AND HERE AS A PLACEHOLDER AND WILL BE REVISED AND REMOVED.]

2.2.1 R

To download R, go to CRAN, the **c**omprehensive **R** **a**rchive **n**etwork. CRAN is composed of a set of mirror servers distributed around the world and is used to distribute R and R packages. Don't try and pick a mirror that's close to you: instead use the cloud mirror, https://cloud.r-project.org, which automatically figures it out for you.

A new major version of R comes out once a year, and there are 2-3 minor releases each year. It's a good idea to update regularly. Upgrading can be a bit of a hassle, especially for major versions, which require you to reinstall all your packages, but putting it off only makes it worse.

2.2.2 RStudio

[TO DO: THERE ARE SOME WORDS BORROWED FROM R4DS BELOW, CLEAN THEM UP.]

RStudio is an integrated development environment, or IDE, for R programming. Download and install it from http://www.rstudio.com/download. RStudio is updated a couple of times a year. When a new version is available, RStudio will let you know. It's a good idea to upgrade regularly so you can take advantage of the latest and greatest features.

2.2.3 The tidyverse

You'll also need to install some R packages. An R **package** is a collection of functions, data, and documentation that extends the capabilities of base R. Using packages is key to the successful use of R. The majority of the packages that you will learn in this book are part of the so-called tidyverse. The packages in the tidyverse share a common philosophy of data and R programming, and are designed to work together naturally.

You can install the complete tidyverse with a single line of code:

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

On your own computer, type that line of code in the console, and then press enter to run it. R will download the packages from CRAN and install them on to your computer. If you have problems installing, make sure that you are connected to the internet, and that https://cloud.r-project.org/ isn't blocked by your firewall or proxy.

You will not be able to use the functions, objects, and help files in a package until you load it with `library()`. Once you have installed a package, you can load it with the `library()` function:

```
library(tidyverse)
#> -- Attaching packages -------------------- tidyverse 1.2.1 --
#> ✓ ggplot2 2.2.1          ✓ purrr   0.2.4
#> ✓ tibble  1.4.2          ✓ dplyr   0.7.4.9002
#> ✓ tidyr   0.8.0          ✓ stringr 1.3.0
#> ✓ readr   1.1.1          ✓ forcats 0.3.0
#> -- Conflicts ----------------------- tidyverse_conflicts() --
#> x dplyr::filter() masks stats::filter()
#> x dplyr::lag()    masks stats::lag()
```

This tells you that tidyverse is loading the ggplot2, tibble, tidyr, readr, purrr, and dplyr packages. These are considered to be the **core** of the tidyverse because you'll use them in almost every analysis.

Packages in the tidyverse change fairly frequently. You can see if updates are available, and optionally install them, by running `tidyverse_update()`.

2.2.4 Other packages

There are many other excellent packages that are not part of the tidyverse, because they solve problems in a different domain, or are designed with a different set of underlying principles. This doesn't make them better or worse, just different. In other words, the complement to the tidyverse is not the messyverse, but many other universes of interrelated packages. As you tackle more data science projects with R, you'll learn new packages and new ways of thinking about data.

In this book we'll use three data packages from outside the tidyverse:

```
install.packages(c("nycflights13", "fivethirtyeight"))
```

[TO DO: ADD OTHERS]

These packages provide data on airline flights, world development, and baseball that we'll use to illustrate key data science ideas.

Chapter 3

Day one

You only get one first day of class, so start with something that excites students, teach the necessary evils later. This means getting to a meaningful, and hopefully interesting (for the students) data visualization as quickly as possible.

3.1 What is data science?

Data science is an exciting discipline that allows you to turn raw data into understanding, insight, and knowledge. We're going to learn to do this in a `tidy` way – more on that later!

3.2 What is this course?

This is an introductory data science course designed for learners with no background in data science, programming, or statistics, but a willingness to learn in class and explore independently.

- Will we be doing computing? Yes.
- Is this an intro CS course? Not really, but many themes are shared.
- Is this an intro stat course? In a way, but it's not your high school statistics course.
- What computing language will we learn? R.
- Why not language X? We can discuss that over :coffee:.

3.3 Data in the wild

I like starting off by showing a few examples of easy to follow but not so simple data analyses done in R, preferably presented along with the R code. This should be the type of analysis they could do for their final project. Blog posts are often good examples for these. In Spring 2018 I used the following as examples:

- A year as told by fitbit by Nick Strayer
- R-Ladies global tour by Maelle Salmon
- Text analysis of Trump's tweets confirms he writes only the (angrier) Android half by David Robinson

3.4 Your turn: UN Votes

It's now time to let the students work on their first data visualization in R.

Using the `unvotes` package, and a pre-populated R Markdown document on RStudio Cloud, they can create and modify the two multi-variate plots, visualizing the voting history of countries in the United Nations General Assembly.

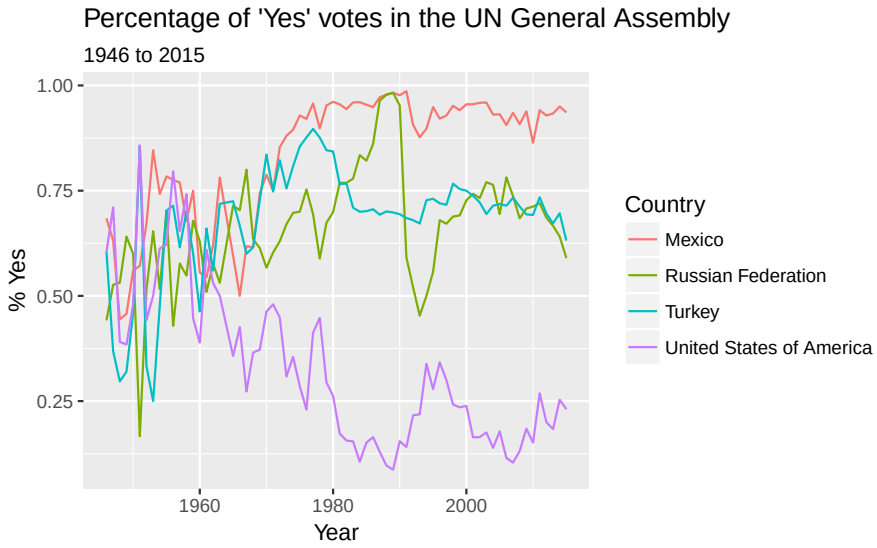
```
library(unvotes)
#> If you use data from the unvotes package, please cite the following:
#>
#> Erik Voeten "Data and Analyses of Voting in the UN General Assembly" Routledge Handbook of Internati
library(tidyverse)
#> -- Attaching packages ------------------- tidyverse 1.2.1 --
#> ✓ ggplot2 2.2.1          ✓ purrr   0.2.4
#> ✓ tibble  1.4.2          ✓ dplyr   0.7.4.9002
#> ✓ tidyr   0.8.0          ✓ stringr 1.3.0
#> ✓ readr   1.1.1          ✓ forcats 0.3.0
#> -- Conflicts ---------------------- tidyverse_conflicts() --
#> x dplyr::filter() masks stats::filter()
#> x dplyr::lag()    masks stats::lag()
library(lubridate)
#> Loading required package: methods
#>
#> Attaching package: 'lubridate'
#> The following object is masked from 'package:base':
#>
#>     date
```

We will narrow down the analysis to four countries: United States, Russian Federation, Mexico, and Turkey.

```
country_list <- c("United States of America", "Russian Federation",
                  "Mexico", "Turkey")
```

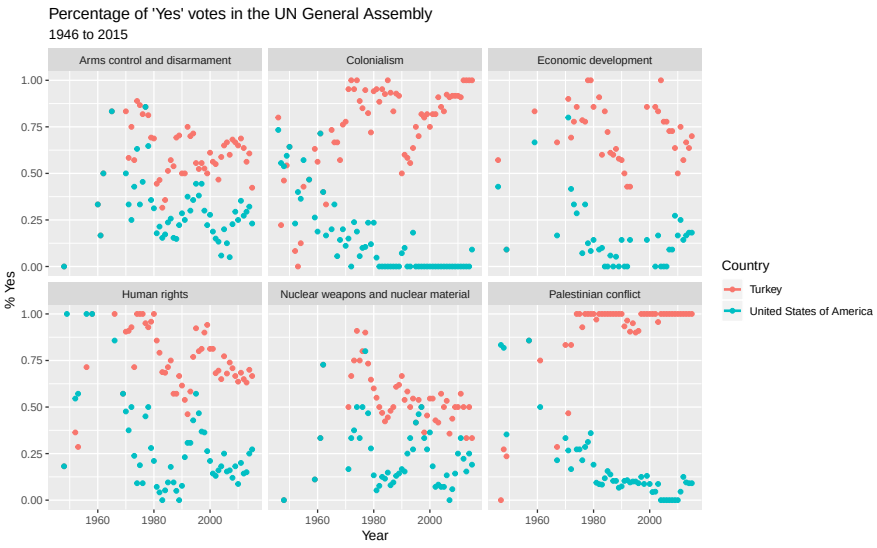
First we take a look at how often each country voted "yes" on a resolution in each year.

```
un_votes %>%
  filter(country %in% country_list) %>%
  inner_join(un_roll_calls, by = "rcid") %>%
  group_by(year = year(date), country) %>%
  summarize(
    votes = n(),
    percent_yes = mean(vote == "yes")
    ) %>%
  ggplot(mapping = aes(x = year, y = percent_yes, color = country)) +
    geom_line() +
    labs(
      title = "Percentage of 'Yes' votes in the UN General Assembly",
      subtitle = "1946 to 2015",
      y = "% Yes",
      x = "Year",
      color = "Country"
    )
```



And then, we create a visualization that compares how the voting record of the United States changed over time on a variety of issues, and compare it to another country.

```
un_votes %>%
  filter(country %in% c("United States of America", "Turkey")) %>%
  inner_join(un_roll_calls, by = "rcid") %>%
  inner_join(un_roll_call_issues, by = "rcid") %>%
  group_by(country, year = year(date), issue) %>%
  summarize(
    votes = n(),
    percent_yes = mean(vote == "yes")
    ) %>%
  filter(votes > 5) %>%  # only use records where there are more than 5 votes
  ggplot(mapping = aes(x = year, y = percent_yes, color = country)) +
    geom_point() +
    geom_smooth(method = "loess", se = FALSE) +
    facet_wrap(~ issue) +
    labs(
      title = "Percentage of 'Yes' votes in the UN General Assembly",
      subtitle = "1946 to 2015",
      y = "% Yes",
      x = "Year",
      color = "Country"
    )
```



At this point, instead of a formal introduction on R syntax, I recommend letting students change parameters passed to these functions, such as which countries are being plotted, and reknit the document to view the changes.

[TO DO: Find a good way to insert slides.]

Link to relevant slides: https://github.com/rstudio-education/datascience-box/tree/master/slides/p0_d01-welcome

Chapter 4

Meet the toolkit

4.1 ...

Part I

Exploring data

Chapter 5

Introduction

This is where into to part 1 goes.

Chapter 6

Data visualization

Some words.

Then embed slides: Visualizing data

Link to lab: Data wrangling and visualization

And to HW: Gotta catch 'em all

Part II

Making rigorous conclusions

Chapter 7

Introduction

This is where into to part 2 goes.

Part III

Looking forward

Chapter 8

Introduction

This part has a bunch of independent modules, each on a different topic. Pick and choose to your liking, depending on how much time you have to cover them.

Part IV

Infrastructure

Chapter 9

Introduction

Intro to part 4 goes here.