

Reporting with Data in R

Christian McDonald

2019-01-14

Contents

Chapter 1

About this class

This collection of lessons is intended to support the class Reporting With Data, taught by me, Christian McDonald, at the School of Journalism, Moody College of Communication, University of Texas at Austin.

I'm a strong proponent of Scripted Journalism, a method of committing data-centric journalism in a programmatic, repeatable and transparent way. There are a myriad of programming languages that further this, including Python (pandas and Jupyter) and JavaScript (Observable), but we'll be using R, RMarkdown and RStudio.

R is a super powerful, open-source programming language for data that is deep with features and an awesome community of users who build upon it. No matter the challenge before you in your data storytelling, there is probably a package available to help you solve that challenge. Probably more than one.

There is always more than one way to do things in R. This course is an opinionated collection of lessons intended to teach students new to R and programming for the expressed act of committing journalism. As a beginner course, I strive to make it as simple as possible, which means I may not go into detail about alternative (and possibly better) ways to accomplish tasks.

About the author

I'm a career journalist who most recently served as Data and Projects Editor at the Austin American-Statesman before coming to the University of Texas at Austin full-time in Fall 2018. I've taught data-related course at UT since 2013.

- UT Github: [utdata](#)
- Github: [critmcdonald](#)
- Twitter: [crit](#)
- Email: christian.mcdonald@utexas.edu

Chapter 2

Introduction to R

Let's get this party started.

NOTE: R and RStudio are already install on lab computers.

2.1 Installing R

Our first task is to install the R programming language onto your computer. There are a number of “mirrors” which have the software.

- Go to the download site.
- Go down to USA and choose one of the links there. They should all work the same.
- Click on the link for your operating system.
- The following steps will differ slightly based on your operating system.
- For Macs, you want the “latest package”
- For Windows, you want the “base” package. You'll need to decide whether you want the 32- or 64-bit version. (Unless you've got a pretty old system, chances are you'll want 64-bit.)

Here's hoping it will be self explanatory after that.

2.2 Installing RStudio

RStudio is an “integrated development environment” – or IDE – for programming in R. Basically, it's the program you will use when doing work for this class.

- Go to <https://www.rstudio.com> and find the “Download RStudio” button.
- Find the “Free” versions and find the installer for your operating system and download it.
- Install it. Should be like installing any other program.

2.3 Getting started with RStudio

2.3.1 Class project folder

To keep things consistent and help with troubleshooting, I'd like you to save your work in the same location all the time.

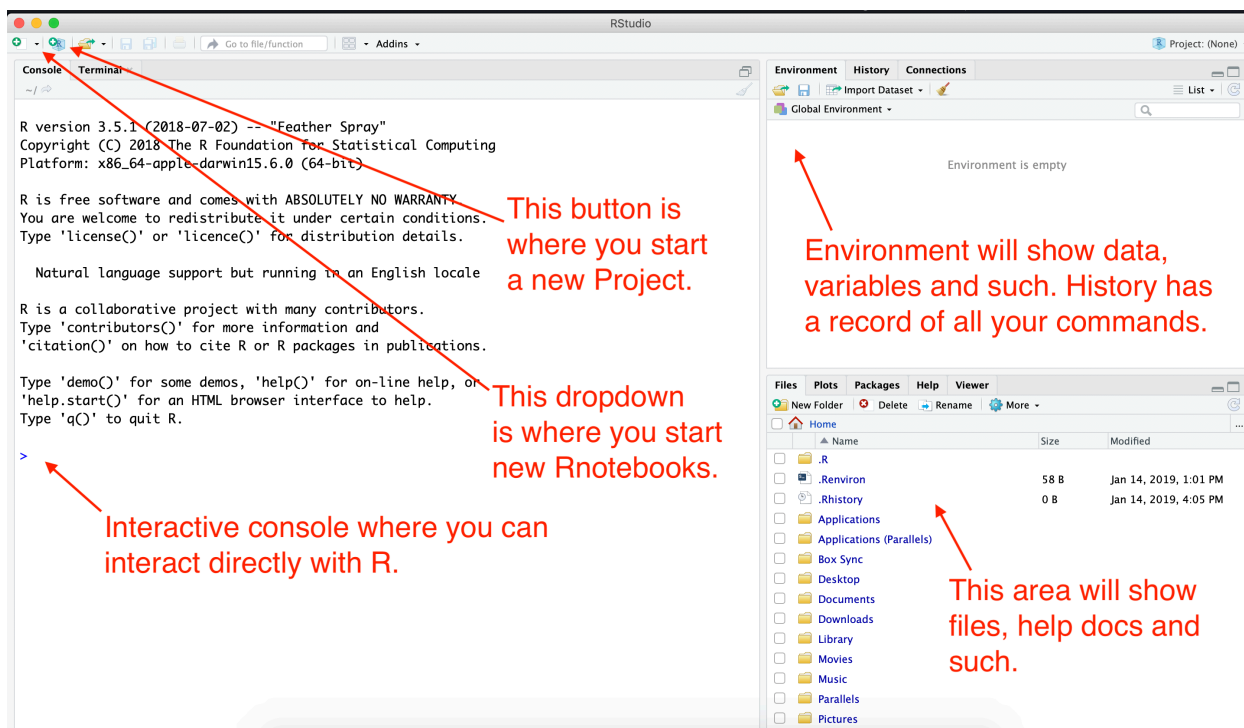


Figure 2.1: Rstudio launch screen

- On both Mac and Windows, every user has a “Documents” folder. Open that folder. (If you don’t know where it is, ask me to help you find it.)
- Create a new folder called “rwd”. Use all lowercase letters.

When we create new “Projects”, I want you to always save them in the Documents/rwd folder.

2.4 RStudio tour

When you launch RStudio, you’ll get a screen that looks like this:

2.5 Starting a new Project

When we work in RStudio, we will create “Projects” to hold all the files related to one another. This sets the “working directory”, which is a sort of home base for the project.

- Click on the second button that has a green +R sign.
- That brings up a box to create the project with several options. You want **New Directory** (unless you already have a Project directory, which you don’t for this.)
- For **Project Type**, choose **New Project**.
- Next, for the **Directory name**, choose a new name for your project folder. For this project, use “firstname-first-project” but use YOUR firstname.

I want you to be anal about naming your folders. It’s a good programming habit.

- Use lowercase characters.
- Don’t use spaces. Use dashes.

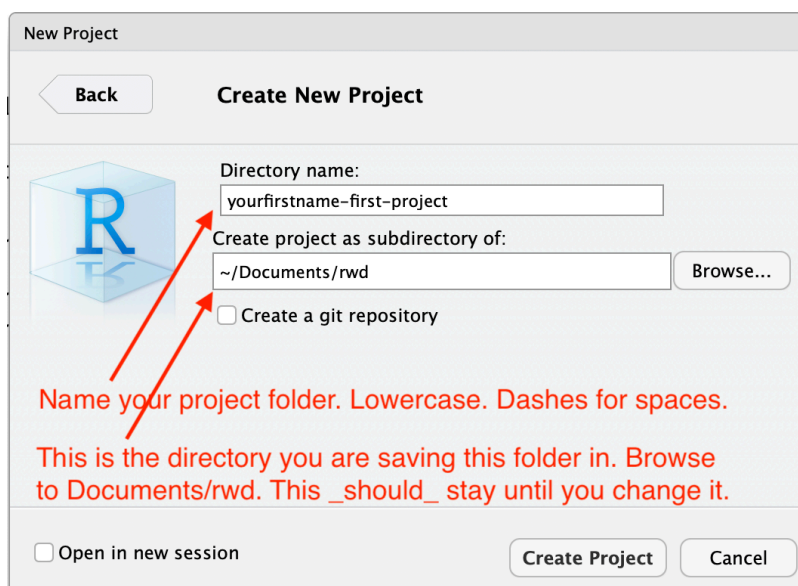


Figure 2.2: Rstudio project name, directory

- For this class, start with your first name.

When you do this, your RStudio window will refresh and you'll see the `yourfirstname-first-project.Rproj` file in your Files list.

2.6 Using R Notebooks

For this class, we will almost always use R Notebooks. This format allows us to write text inbetween our blocks of code. The text is written in a language called R Markdown. It allows us to write text that gets turned into pretty HTML for our reports. The R Markdown syntax is not hard. It is based on vanilla Markdown, which is a common documentation syntax for programmers.

2.6.1 Create your first notebook

- Click on the button at the top-left of RStudio that has just the green + sign.
- Choose the item **R Notebook**.

This will open a new file with some boilerplate R Markdown code.

- At the top between the `---` marks, is the **metadata**. This is written using YAML, and what is inside are commands for the R Notebook. Don't sweat the YAML syntax too much right now, as we won't be editing it often.
- Next, you'll see a couple of paragraphs of text that describes how to use an R Notebooks. It is written in R Markdown, and has some inline links and bolding commands, which you will learn,
- Then you will see an R code chunk that looks like the figure below.

Let's take a closer look at this:

- The three backticks characters (found at the top left on your keyboard) followed by the `{r}` indicate that this is a chunk of R code. The last three backticks say the code chunk is over.
- The `{r}` bit can have some parameters added to it. We'll get into that later.
- The line `plot(cars)` is R programming code. We'll see what those commands do in a bit.

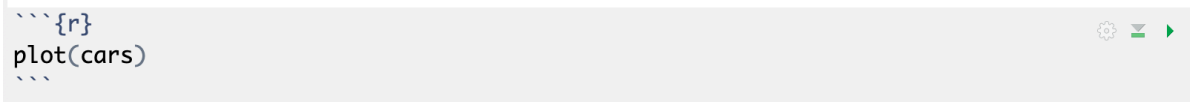


Figure 2.3: R code chunk

- The green right-arrow to the far right is a play button to run the code that is inside the chunk.
- The green down-arrow and bar to the left of that runs all the code in the Notebook up to that point.

2.6.2 Save the .Rmd file

- Do command-s or hit the floppy disk icon to save the file.
- It will ask you what you want to name this file. Call it `01-first-file.Rmd`.

When you do this, you may see another new file created in your Files directory. It's the pretty version of the notebook which we'll see in a minute.

In the metadata portion of the file, give your notebook a better title.

- Replace "R Notebook" in the `title: "R Notebook"` code to be "Christian's first notebook", but use your name.

2.6.3 Run the notebook

There is only one chunk to run in this notebook, so:

- Click on the green right-arrow to run the code.

You should get something like this:

What you've done here is create a plot chart of a piece of sample data that is already inside R. (FWIW, It is the speed of cars and the distances taken to stop. Note that the data were recorded in the 1920s.)

But that wasn't a whole lot of code to see there is a relationship with speed vs stopping distance, eh?

2.6.4 Adding new code chunks

The text after the chart describes how you can insert a new code chunk. After that text, I'd like you to do that.

- After the text that describes how to add code, but before the next bit of text, add a new code chunk with `Cmd+Option+I`.
- Your cursor will be inserted into the middle of the chunk. Type in this code:

```
# min age to date
age = 52
(age / 2) + 7
```

- Change for "52" to your real age.
- With your cursor somewhere in the code block, use the key command `Cmd+Shift+Return`, which is the key command to run an ENTIRE chunk.
- NOTE: To run an individual line, use `Cmd+Return` while on that line.

Congratulations! The answer given at the bottom of that code chunk is the socially-acceptable minimum age of anyone you should date.

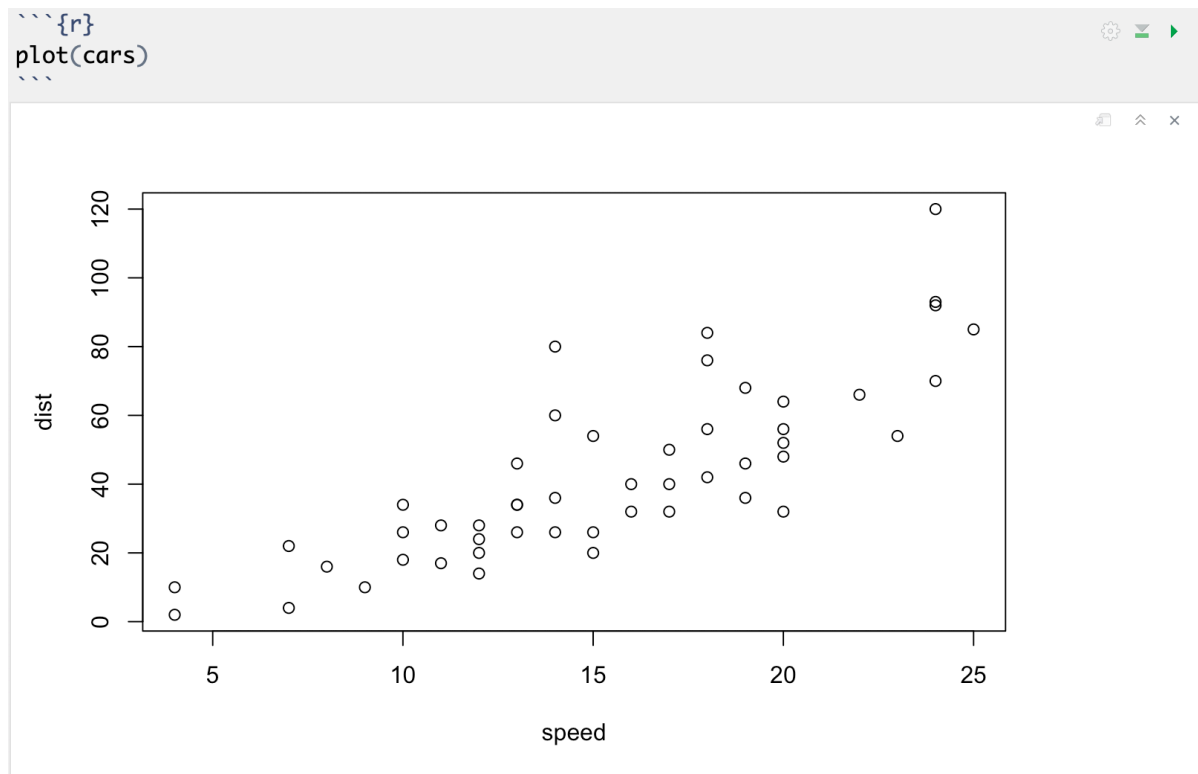


Figure 2.4: Cars plot

Throwing aside whether the formula is sound, let's break down the code.

- `# min age to date` is a comment. It's a way to explain what is happening in the code without being considered part of the code.
- `age = 51` is assigning a number (52) to a variable name (`age`).
- `(age / 2) + 7` takes the value of `age` and divides that by 2, then adds 7.

Now you can play with the `age` variable assignment to test out different ages.

2.6.5 Practice adding code chunks

Now, on your own, add a similar code chunk that calculates the maximum age of someone you should date, but using the formula $(\text{age} - 7) * 2$.

2.6.6 Preview the report

The rest of the boilerplate text here describes how you can *Knit* or *Preview* a notebook. Let's do that now.

- Press `Cmd+Shift+K` to open a Preview.

This will open a new window and show you the “pretty” notebook that we are building. It's really an HTML file that was created by RStudio. (You can also open this in a web browser).

Preview is a little different than *Knit*, which runs all the code, then creates the new knitted document.

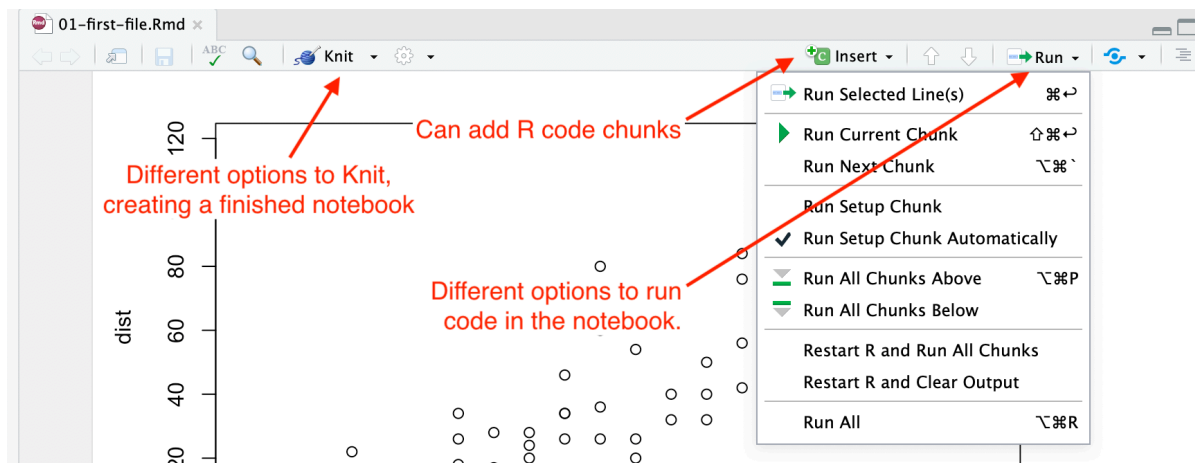


Figure 2.5: R Notebook toolbar

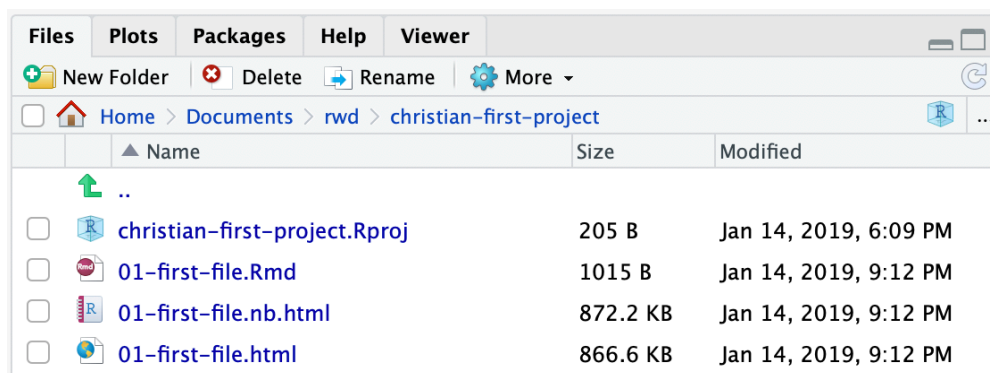


Figure 2.6: Files list

2.6.7 The toolbar

Last thing to describe before we turn this in is the toolbar that runs across the top of the R Notebook file window.

2.6.8 Knit the final workbook

- Save your File with *Cmd+S*.
- Use the **Knit** button to choose **Knit to HTML**.

2.7 Turning in our projects

So now if you look in your Files pane, you'll see you have four files in our project.

Now we have to zip these all up into a single file that you can turn into Canvas. (Note the only one you actually edit is the .Rmd file.)

- In your computer's finder, open the `Documents/rwd` folder.
- Follow the directions for your operating system linked below to create a compressed version of your `yourname-final-project` folder.