EMT Learns R - Session 2

Lacey Hartigan

11/8/2021

Welcome to the Wonderful World of R Markdown

Welcome to (what may be) your first .Rmd file! You'll notice that one of the advantages of .Rmd files is the ability to intersperse text with code. The final product is a really cool mix of both!

First off, I want to introduce you to some great resources for understanding the different things that can go into an .Rmd file. Understanding these pieces will allow you to 1. recognize when things are out of order (and will therefore cause you problems) and 2. customize and make beautiful "knit" files.

.Rmd and other Resources

Why did I begin lines 10 and 14 with two number signs, you ask? That's because I want that text to be a header in my final output file. There are LOTS of formats you can apply to your file. You'll often see me using headers, italics, and bold, for example. There are numerous reference guides available on the web. Here's one that I like:

Another important piece of the .Rmd file is the YAML header. R is VERY picky about the formatting for the header (extra spaces, missing punctuation - they're all dealbreakers). Here are some great resources for how to format and what to include/not include in your YAML header. For producing an html output:. For producing a PDF output:.

.Rmd files contain three basic elements:

1. Script (or code) that can be interpreted by R.
2. Output generated by R, including tables and figures.
3. Text that can be read by humans.

From an .Rmd file you can generate html documents, pdf documents, word documents, powerpoint slides, etc.

R Packages

When we say that R is extensible, we mean that people in the community can write programs that everyone else can use. These are called "packages." In these first few lines of code, I load a set of packages using the library command in R. The set of packages, called `tidyverse` were written by Hadley Wickham and play a key role in his book. You only need to install a package once per computer, which is why we don't put installation code in our .Rmd file, but instead, type it directly into the console. You do, however, need to OPEN the package (using the library command), each time you want to use it and you want to do that in your .Rmd code file. Go ahead and install tidyverse on your system. Type `install.packages("tidyverse")` at the R command prompt (next to the > in the Console).

Please note that because R packages are written by different people in the community, it is common for different packages to use the same name for a function (e.g. filter, which is used by the packages dplyr and stats). If this problem exists in the packages that you are using, R will warn you that certain functions have been "masked." There are different ways to deal with this if it presents a problem: 1. Load the packages in order with the one you want to use loaded last (e.g., if I want to use the filter command from dplyr, then I would load that AFTER loading stats). 2. Force R to use the package you want by adding two colons before the function (e.g., dplyr::filter). 3. Detach one of the conflicting packages at the end of the code chunk in which you needed it by using the detach function (e.g., detach("package:maps", unload=TRUE).

While you don't need to worry TOO much about this now, you should be sure to look at what functions/commands are masked as you load packages. That way you'll be prepared to deal with that conflict when needed.

Now, back to our regularly scheduled programming... TIDYVERSE! Once you've installed tidyverse, you want to run the code chunk below. To do this, you can:

- Press the green "play" button at the upper right corner of the code chunk
- In OS X, place the cursor in the code chunk and hit `cmd+shift+enter`
- In Windows, place the cursor in the code chunk and hit `ctrl+shift+enter`

```
knitr::opts_chunk$set(echo = TRUE) #this will include all your code in your R output
#files; for your class assignments, you want this specified. Do this in the first
#code chunk of all your .Rmd files. Note that when you see something like this:
#knitr::opts_chunk$set this is setting a GLOBAL knit option that will apply to your whole
#.Rmd file.

library(tidyverse) #Get necessary libraries-- if this doesn't work it's likely because
```

```
## -- Attaching packages --------------------------------------- tidyverse 1.3.1 --

## v ggplot2 3.3.5     v purrr   0.3.4
## v tibble  3.1.5     v dplyr   1.0.7
## v tidyr   1.1.4     v stringr 1.4.0
## v readr   2.0.2     v forcats 0.5.1

## -- Conflicts ------------------------------------------ tidyverse_conflicts() --
## x dplyr::filter() masks stats::filter()
## x dplyr::lag()    masks stats::lag()
```

```
#you need to install the packages we're using!
```

A note about notes: Notice in the code chunk above, I have text that follows a # that seems to be a note to myself (and you). While anything OUTSIDE of code chunks is automatically interpreted as text/non-code in the .Rmd file; everything INSIDE code chunks is interpreted as code, unless you tell R otherwise. The way we add annotations/notes INSIDE code chunks is by using a single #. Go ahead and try adding some notes in the code chunk above.

Loading Data

Now we're ready to load in data. The `sc` data frame contains information from the college scorecard on 127 different colleges and universities, saved as an R dataset.

However, we first need to make sure that R is looking in the right place. R will automatically look for the data in the same folder where your .Rmd code file is saved. Therefore, if your data file is saved in the same folder as your .Rmd file, you can simply use the "load" command (for R datasets). If your data file is saved elsewhere, you need to modify your working directory to tell R where it can find your data. Both of these things are demonstrated below.

```
#Load in the data - IF the datafile is saved in the same place as my .Rmd file,
#this works:
load("college.Rdata")

#you can also use the dropdown under "More" on the "Files" tab in the lower right to navigate/reset the
#you can use the "Session" dropdown menu at the top of Rstudio to reset the working directory.

#Every so often, the working directory seems to get "lost" during the knitting
#process (particularly if you're on a Mac). One way to make SURE that your current
#.Rmd file is looking in the right place for data is to set a global option
#(typically in the first code chunk in your file) that forces R to look where you
#tell it to. Here's how you can do that:
knitr::opts_knit$set(root.dir="C:/Users/lacey/Desktop/EMT Learns R")

#If I'm ever uncertain about WHERE R is looking for files, I can run the getwd() function
#and it will tell me what my current working directory is
getwd()
```

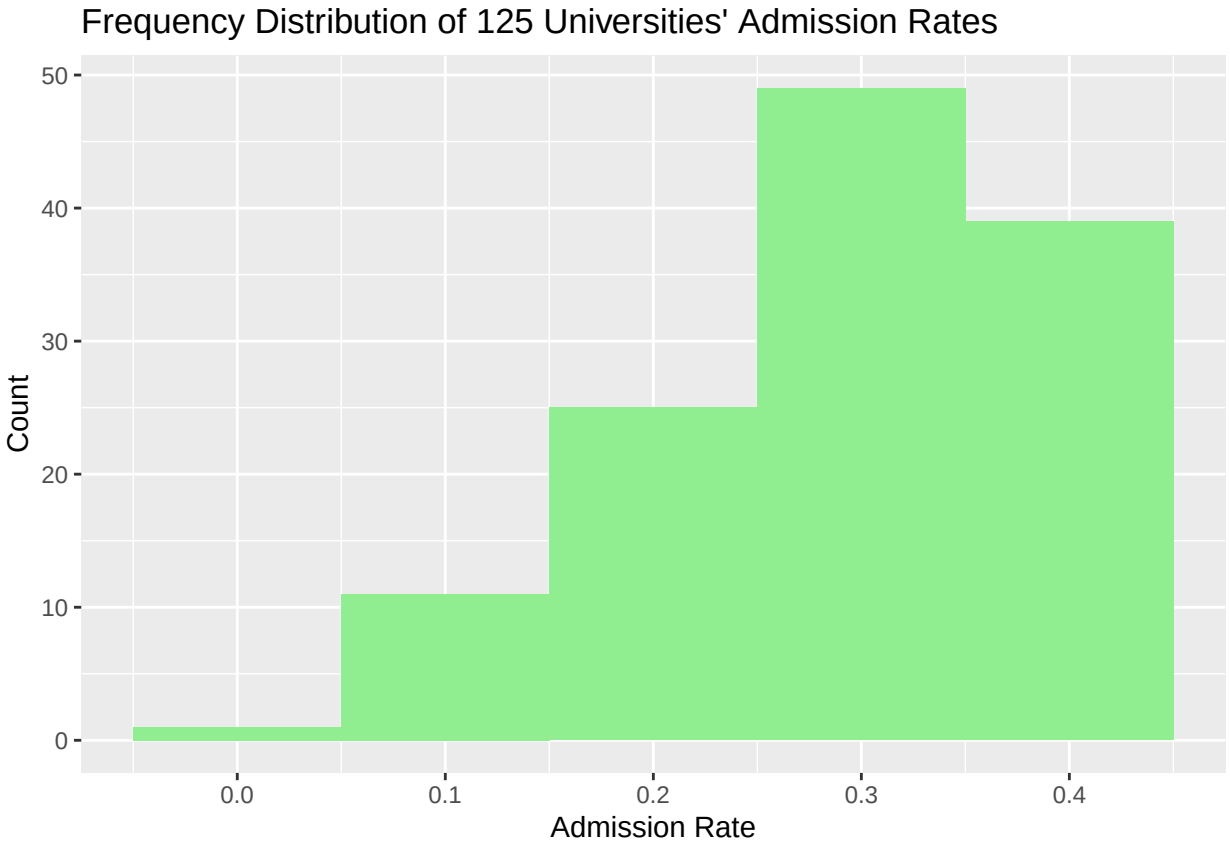
```
## [1] "C:/Users/lacey/Desktop/EMT Learns R"
```

Intro to Plots

To really communicate things about our data/analyses, we're often going to want to plot it. While we're not going to get TOO far into this right now, I want to do a simple histogram using ggplot to show you how we're going to build plots this term.

Let's plot a histogram of admission rates for the 125 universities in our dataset so we can understand its distribution.

```
adm_hist<-ggplot(sc, aes(x=adm_rate)) +
  geom_histogram(binwidth=.1, fill="lightgreen") +
  labs(x="Admission Rate", y="Count",
       title="Frequency Distribution of 125 Universities' Admission Rates")
adm_hist
```



Now, let's talk about knitting. Don't worry - no real needles are involved here!