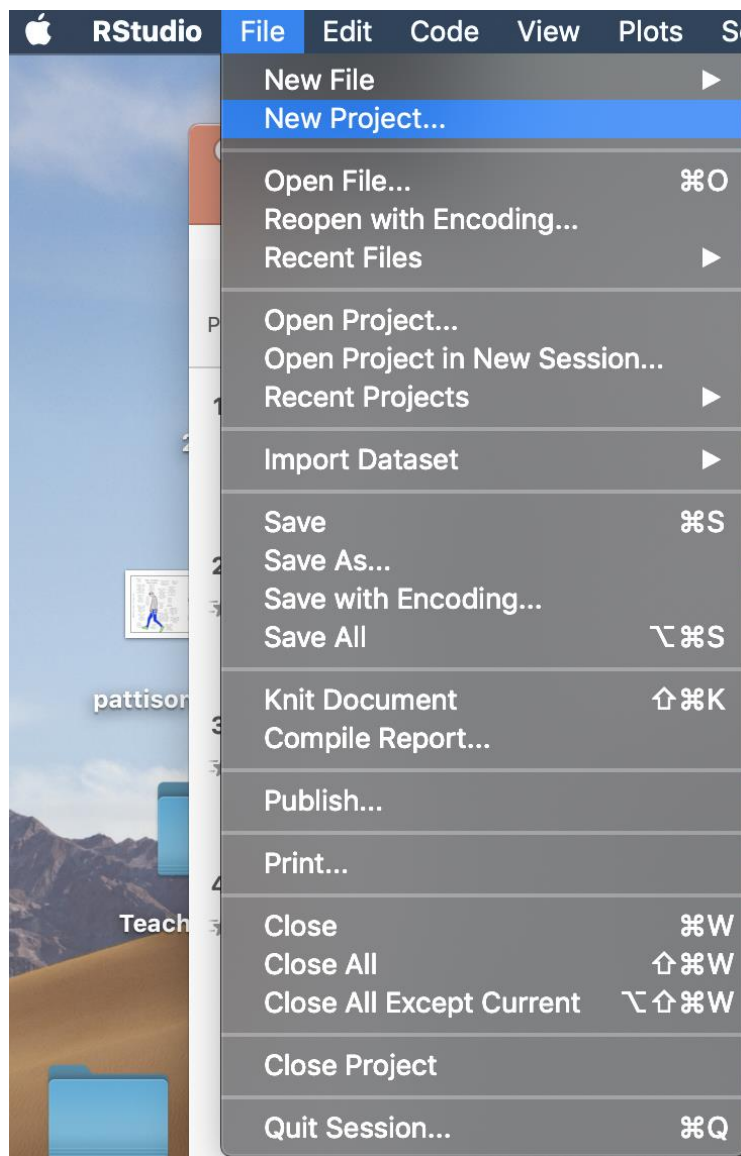


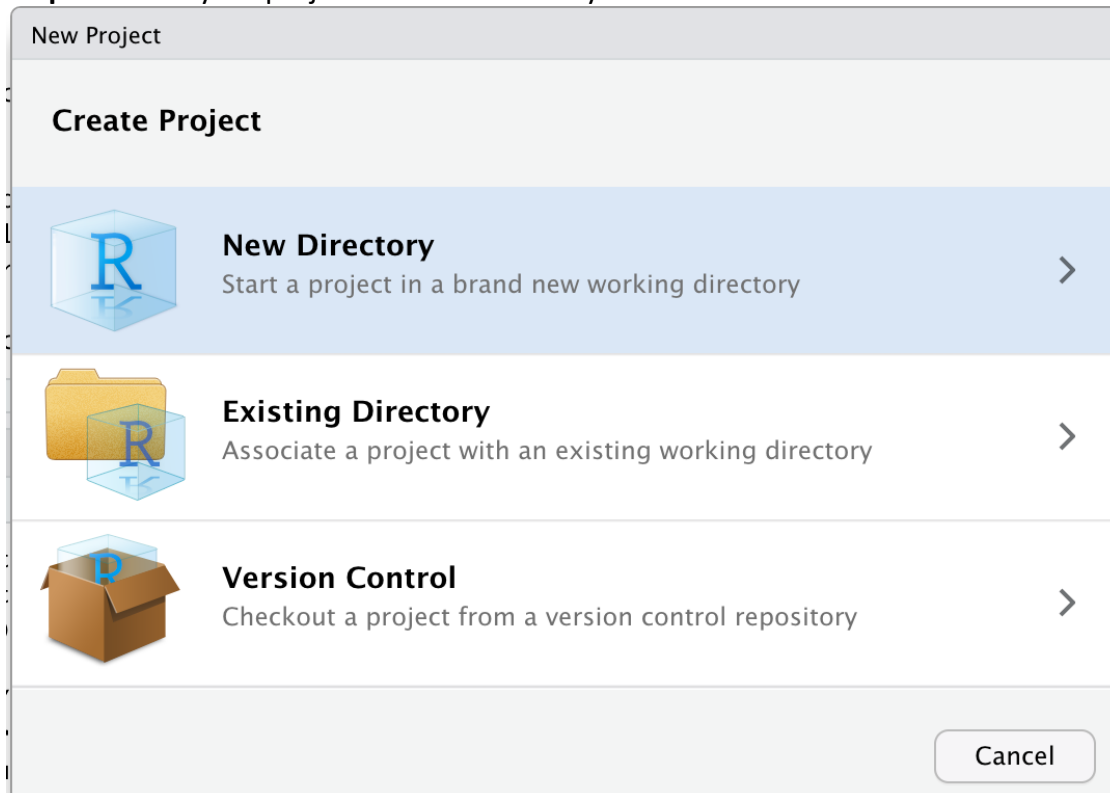
R Markdown Startup Guide

Based on some student feedback about my email message about how to get started in R Markdown, I've put this user guide together to hopefully make this process a bit clearer. Unfortunately I only have access to a Mac, so if you're using a PC, you'll have to figure out how to adapt these instructions to the Windows operating system. But I don't think that will be too hard. And if anyone who has a PC would like to volunteer to write a PC version of this guide, I will be extremely grateful.

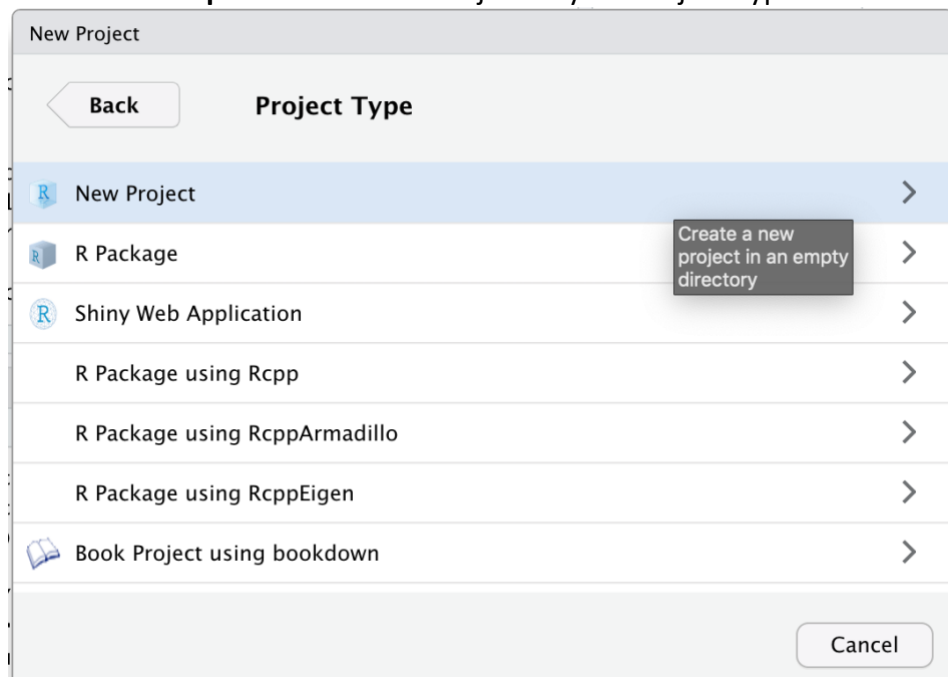
Step 1: Open RStudio and create a new project.



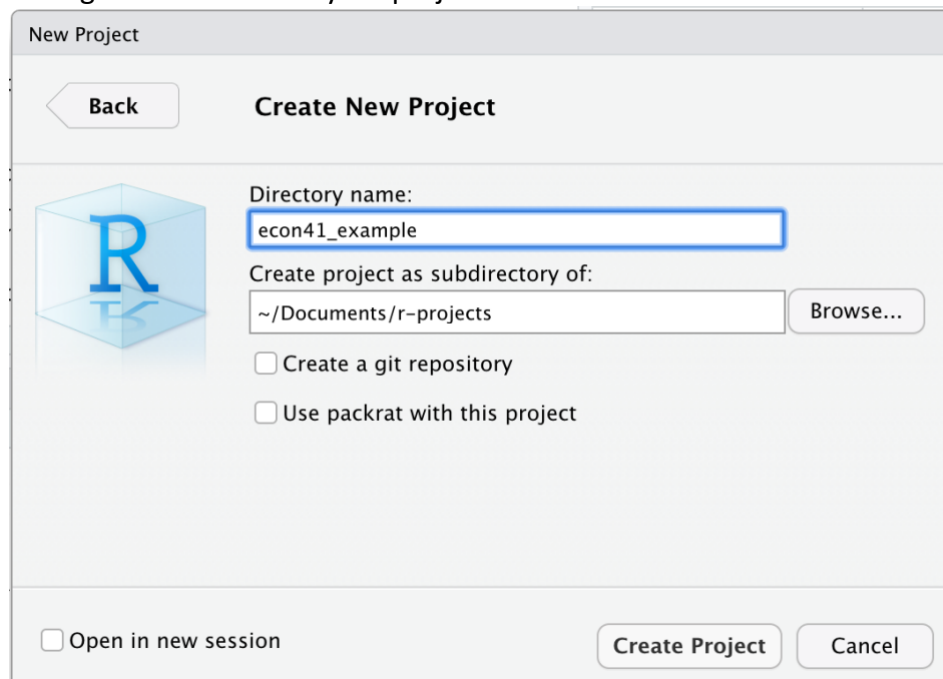
Step 2: Create your project in a new directory.



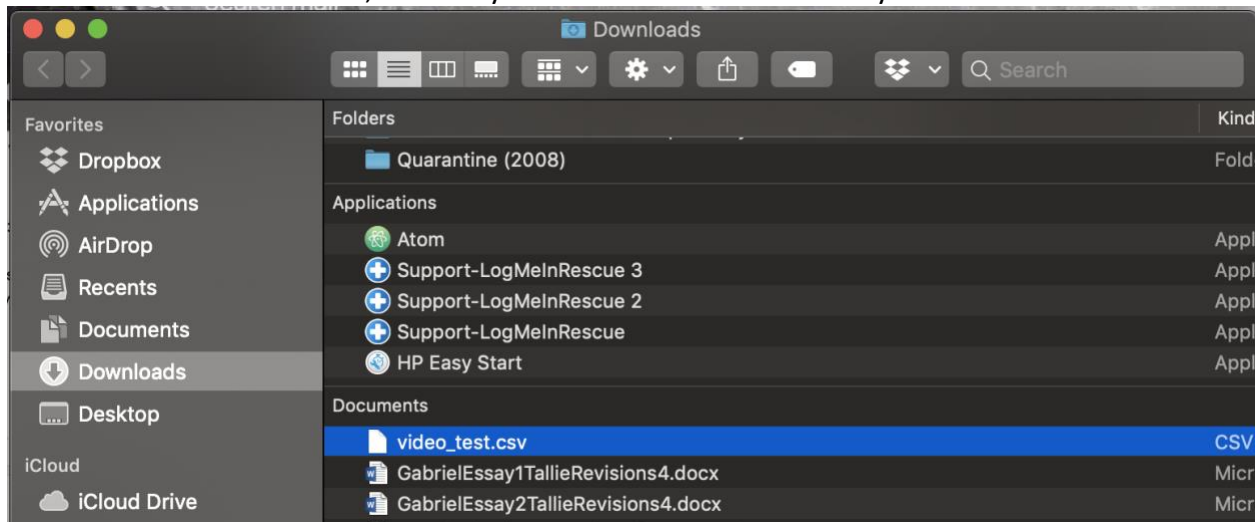
Step 3: Choose New Project as your Project Type.



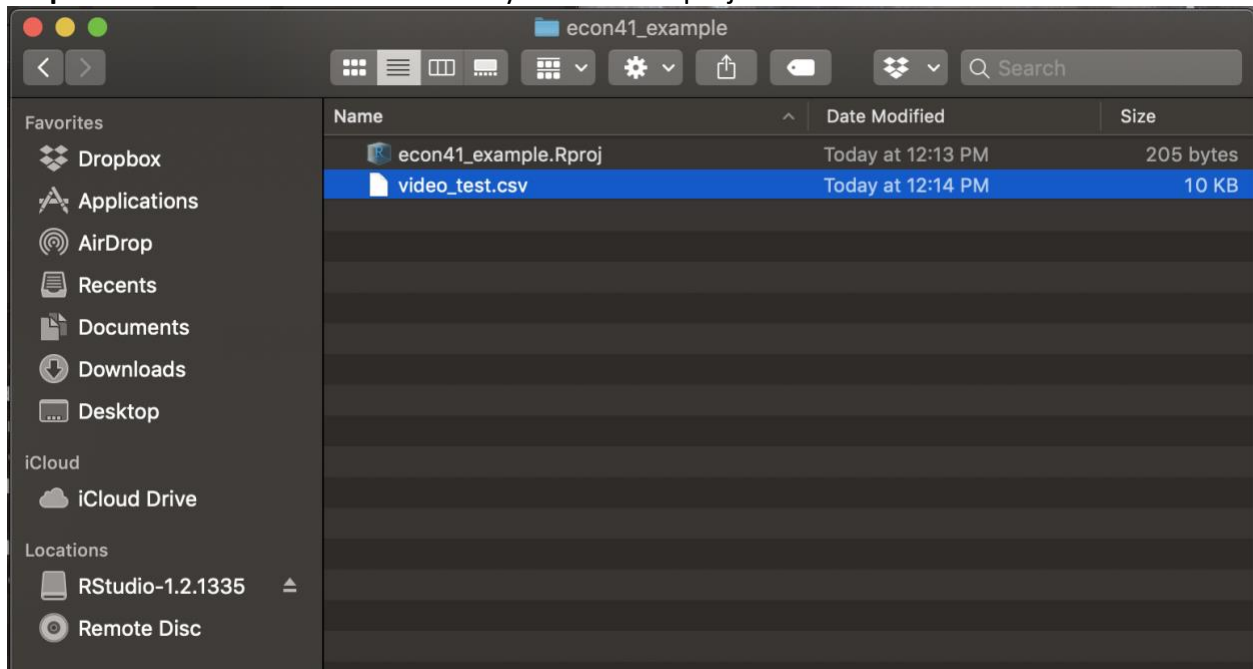
Step 4: Name your project after our class. (I chose “econ41_example” because I already have a project called “ECON41”. You probably don’t, so this is the project name I recommend for you.) Please do not forget the location of your project folder.



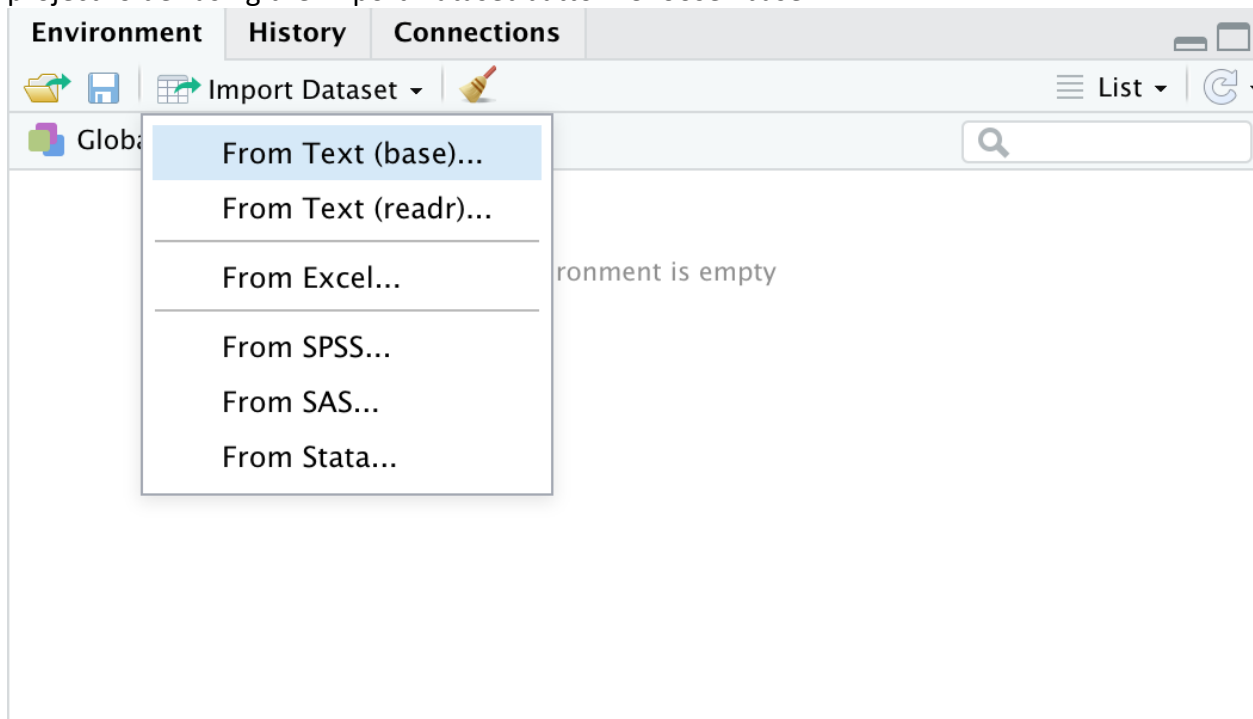
Step 5: Download this week’s dataset from the message I sent out last week, then open the dataset in its folder. For me, this is my Downloads folder. Yours may be different.



Step 6: Move this week's dataset into your ECON41 project folder.



Step 7: Return to your ECON41 project in RStudio and load this week's dataset from your project folder using the Import Dataset button. Choose "base".



Step 8: This window gives you some options for modifying how the dataset is imported into RStudio. You won't need to change anything here for now, though you can give the dataset a shorter name like "video" instead of "video_test" by editing the name in the "Name" field if you wish. To finish importing the dataset into your working environment, click the "Import" button.

Import Dataset

Name: video_test

Input File: [File path]

Encoding: Automatic

Heading: Yes

Row names: Automatic

Separator: Comma

Decimal: Period

Quote: Double quote (")

Comment: None

na.strings: NA

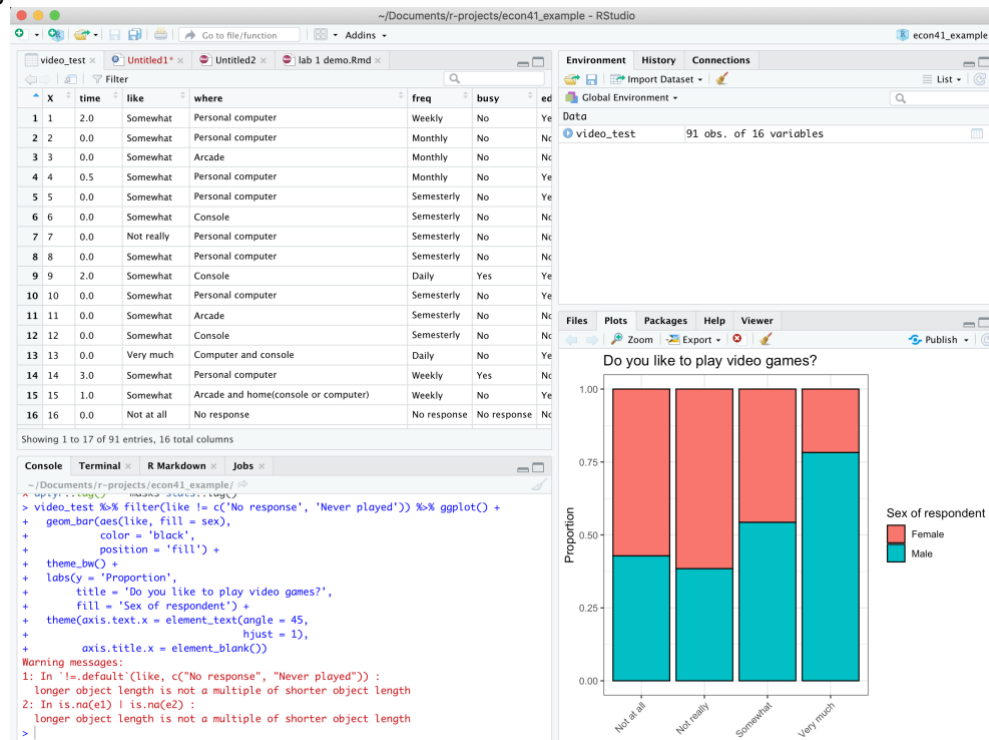
☒ Strings as factors

Data Frame Preview:

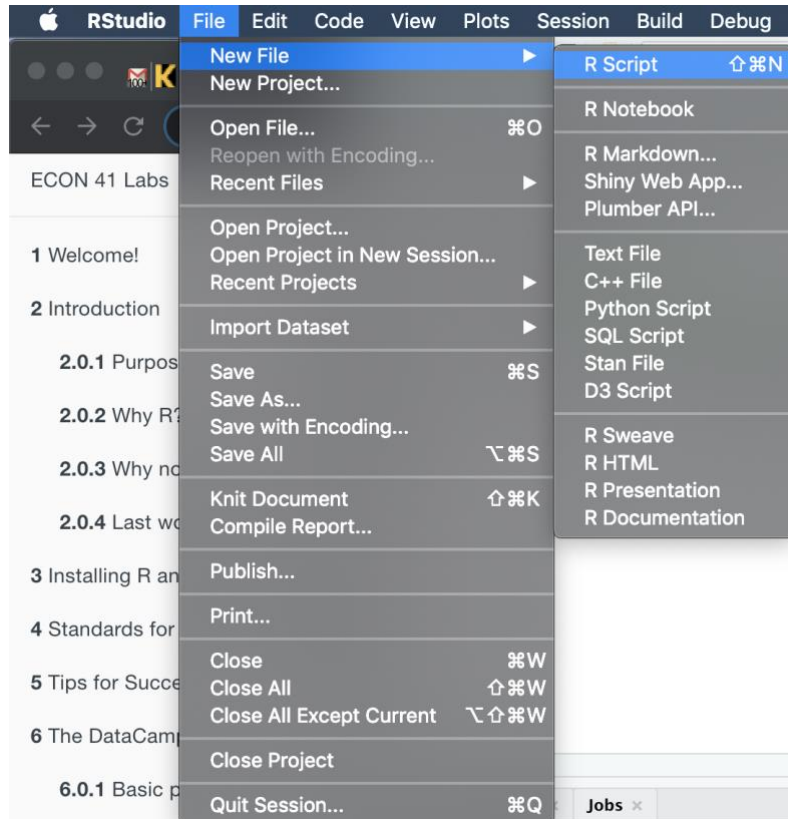
X	time	like	where
1	2.0	Somewhat	Personal computer
2	0.0	Somewhat	Personal computer
3	0.0	Somewhat	Arcade
4	0.5	Somewhat	Personal computer
5	0.0	Somewhat	Personal computer
6	0.0	Somewhat	Console
7	0.0	Not really	Personal computer
8	0.0	Somewhat	Personal computer
9	2.0	Somewhat	Console
10	0.0	Somewhat	Personal computer
11	0.0	Somewhat	Arcade
12	0.0	Somewhat	Console
13	0.0	Very much	Computer and console
14	3.0	Somewhat	Personal computer
15	1.0	Somewhat	Arcade and home(console or computer)
16	0.0	Not at all	No response

Import Cancel

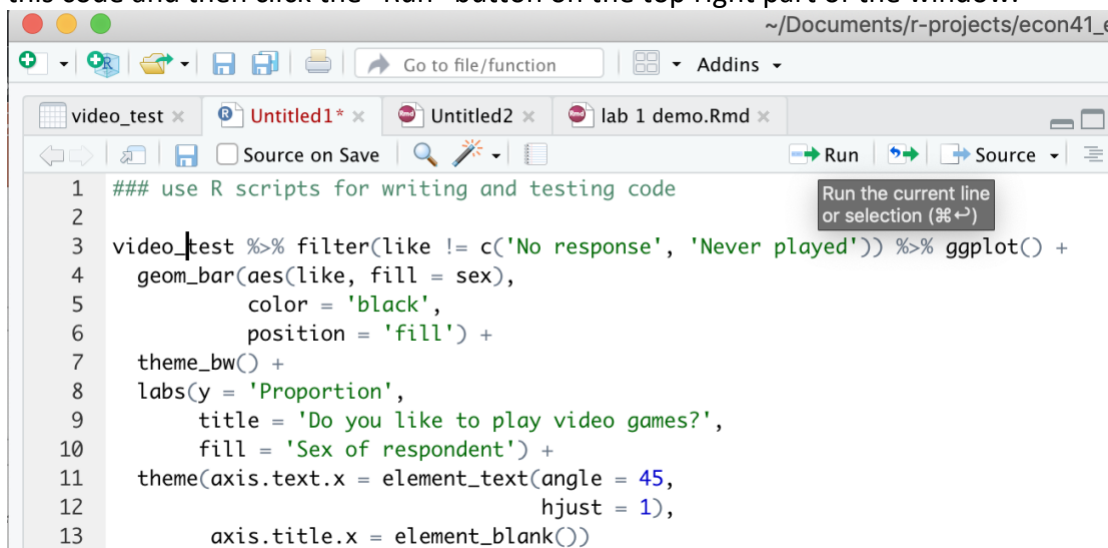
Step 9: If you've followed all of the above steps correctly, your RStudio window should look something like this. In particular, in the upper right window, you should see the name of the dataset you imported on the list of objects in your working environment. Now you can start analyzing this data.



Step 10: It's better to use R scripts to write and test code than relying on the console. (The console is the window on the bottom left of your RStudio window.) Using R scripts to write and test code will make it easier for you to keep track of your code during your analytical process. Saving R scripts will enable you to refer to code you've written in the past in case you need it later.



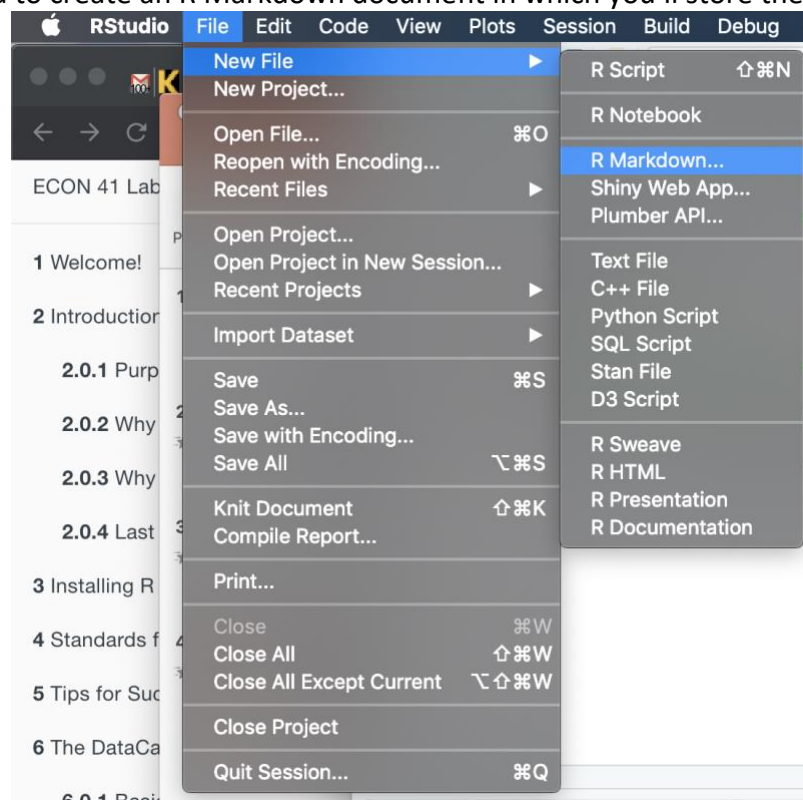
Step 11: Here's an example of how to use an R script. In this script I've written down the code for a visualization I want to generate. In order to test it, I can click somewhere in the block of this code and then click the "Run" button on the top right part of the window.



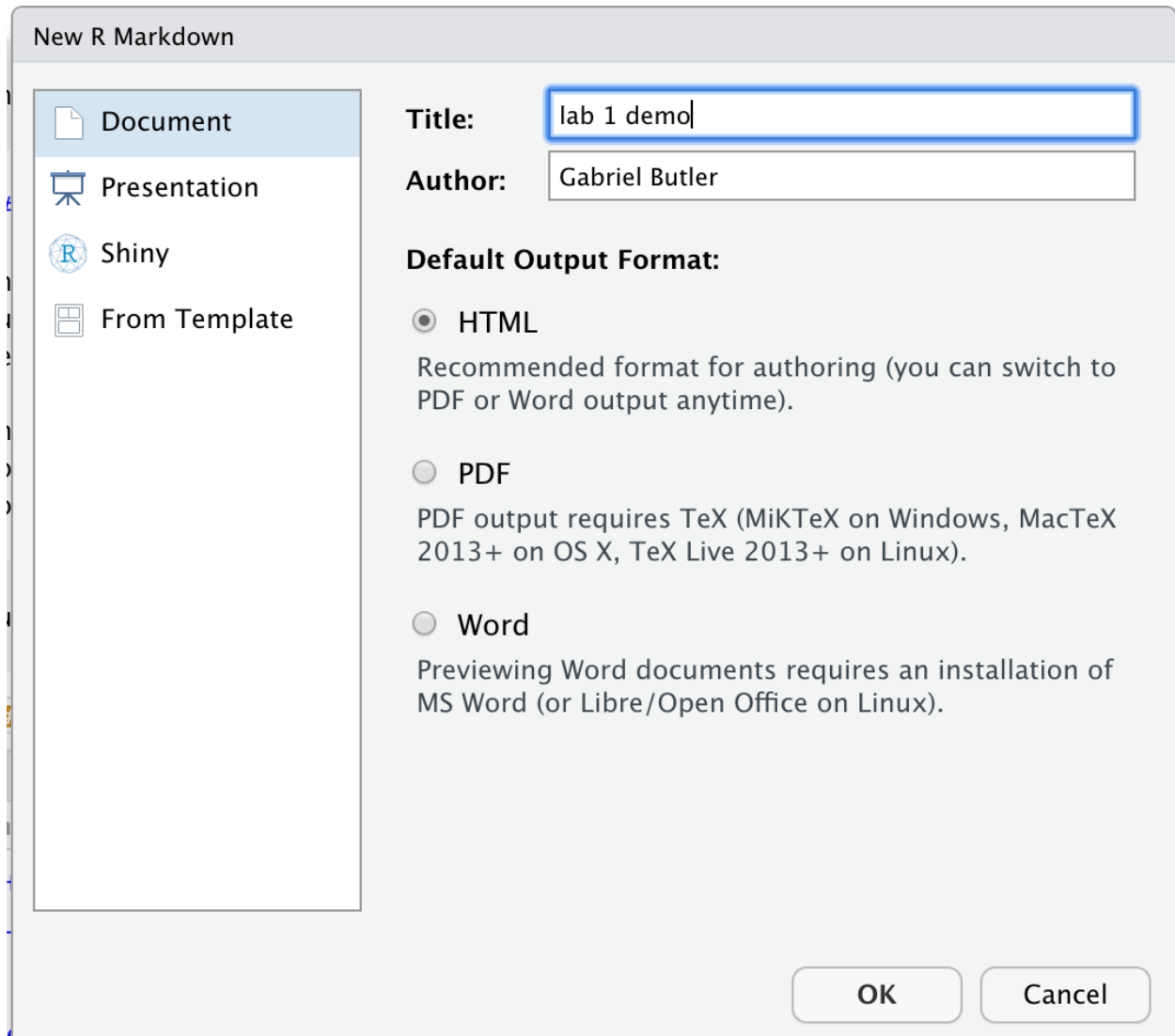
Step 11 (cote.): The code in the screenshot above happens to be working code. I.e., there are no bugs in it and it does what I want it to do, which is generate a certain type of visualization that looks a certain way. But it actually took me a lot of trial and error in order to get it to work exactly the way I wanted, and writing my code in a script and running it from a script made this editing process a lot more efficient than using the console would have.

One other thing about scripts: there is no limit to the number of code blocks you can insert into one. I.e., you do not need to make a new script for every type of object you want to create for a project...unless you really want to. You can use scripts in any way you want. Just remember that the goal is to keep your work as well organized as possible.

Step 12: Once you're finished creating all of the code objects you want to insert into your lab report, you need to create an R Markdown document in which you'll store them.



Step 13: You'll need to name the document too. As usual, you should give it a name that will be easy for you to keep track of later, so I recommend calling it "Lab 1". Make sure you choose "HTML" as the Default Output Format. The other output types take much longer for your computer to generate and I strongly discourage choosing an output format other than HTML.



The image shows a 'New R Markdown' dialog box. On the left, there is a sidebar with four options: 'Document' (selected), 'Presentation', 'Shiny', and 'From Template'. Each option has a corresponding icon. To the right of the sidebar, there are two text input fields: 'Title:' with the text 'lab 1 demo' and 'Author:' with the text 'Gabriel Butler'. Below these fields, the 'Default Output Format:' section contains three radio button options: 'HTML' (selected), 'PDF', and 'Word'. Each option has a descriptive text block below it. The 'HTML' option is described as the 'Recommended format for authoring (you can switch to PDF or Word output anytime)'. The 'PDF' option is described as requiring 'TeX (MiKTeX on Windows, MacTeX 2013+ on OS X, TeX Live 2013+ on Linux)'. The 'Word' option is described as requiring 'an installation of MS Word (or Libre/Open Office on Linux)'. At the bottom right of the dialog box, there are two buttons: 'OK' and 'Cancel'.

New R Markdown

Document

Presentation

Shiny

From Template

Title: lab 1 demo

Author: Gabriel Butler

Default Output Format:

☒ **HTML**

Recommended format for authoring (you can switch to PDF or Word output anytime).

☐ **PDF**

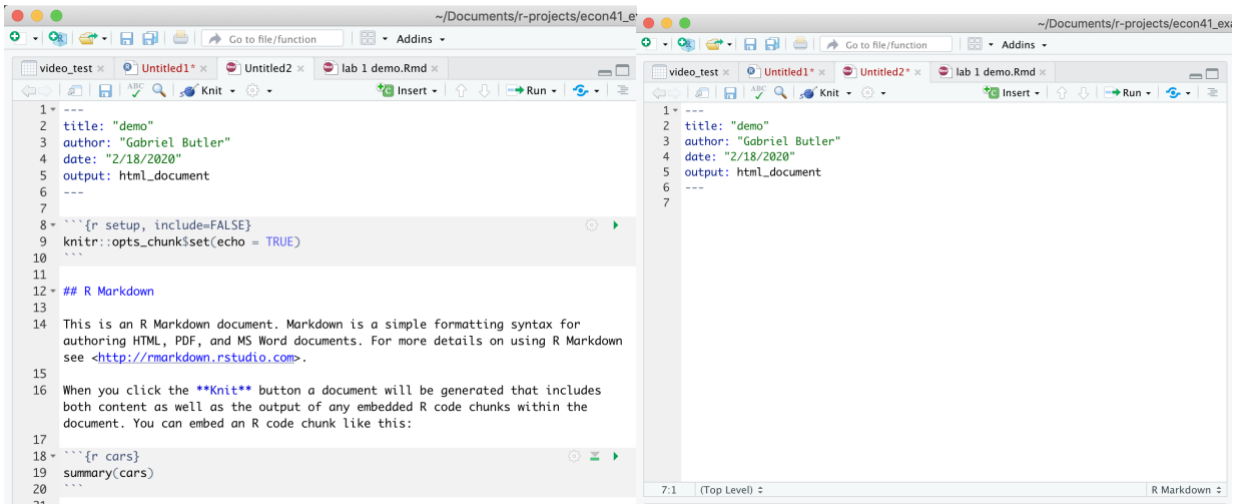
PDF output requires TeX (MiKTeX on Windows, MacTeX 2013+ on OS X, TeX Live 2013+ on Linux).

☐ **Word**

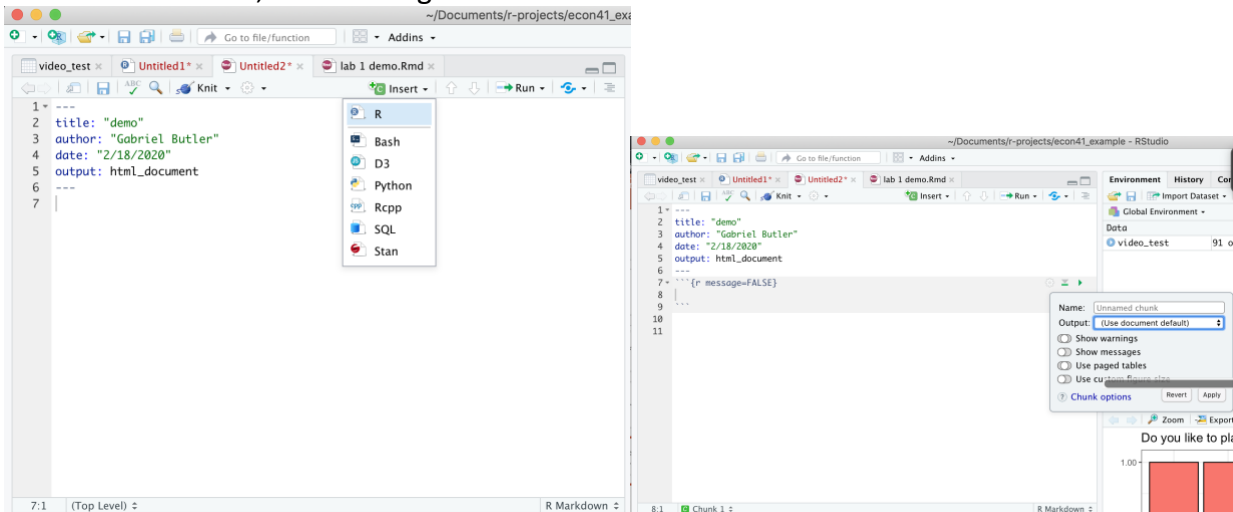
Previewing Word documents requires an installation of MS Word (or Libre/Open Office on Linux).

OK Cancel

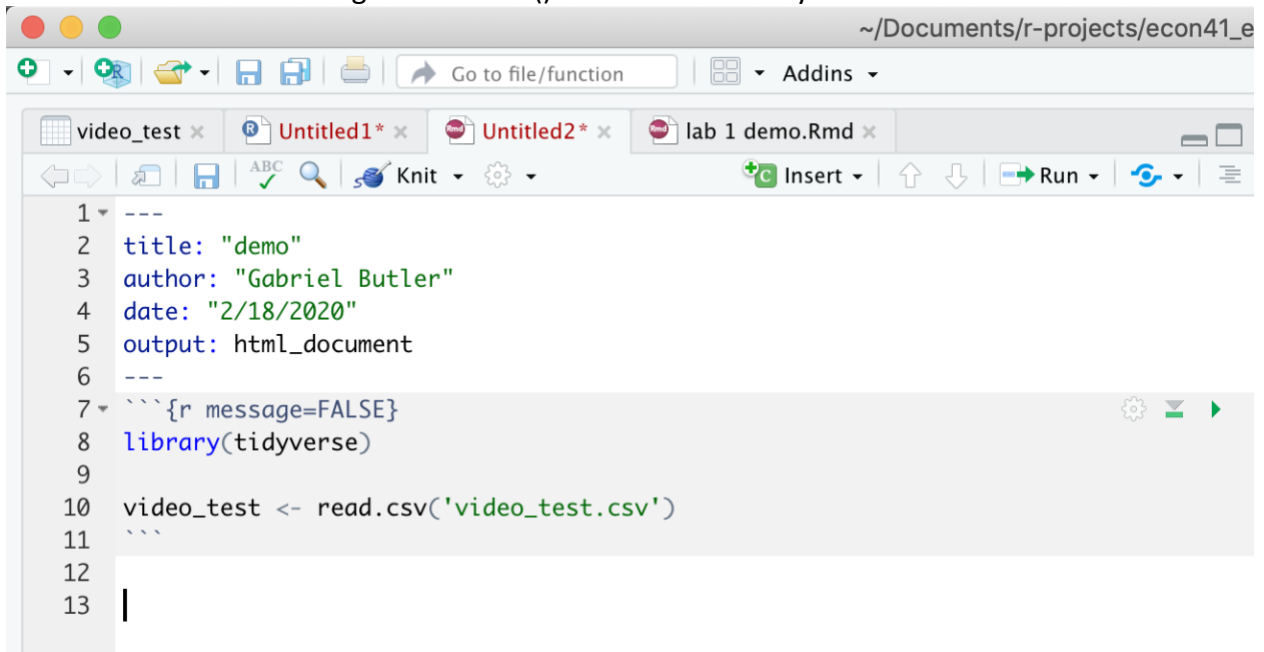
Step 14: The new R Markdown document will contain lots of example code. Delete everything below line 6.



Step 15: Although you loaded the dataset into your working environment earlier, you'll need to do it again inside of R Markdown. There is no button for this. Instead, you will have to write a bit of code. To do this, click the "Insert" button and then click "R" to create an R code chunk. For this code chunk, turn messages off.

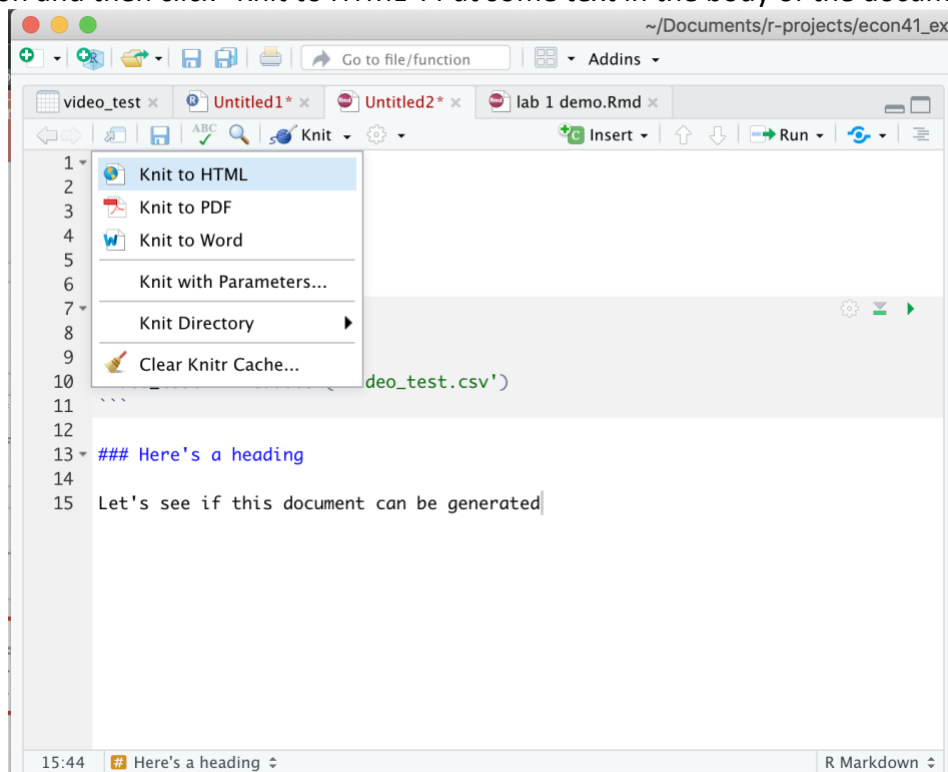


Step 16: Now you'll need to do two things: load the dataset and load the packages you're going to use in your analysis. At the very least, you'll probably have to load the tidyverse package. The dataset will be loaded using the `read.csv()` function in the way shown below.



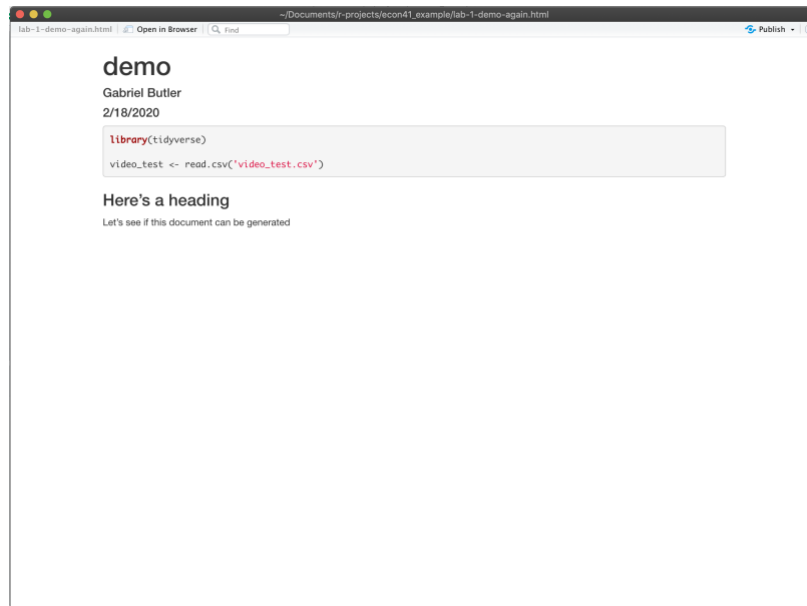
```
1 ---
2 title: "demo"
3 author: "Gabriel Butler"
4 date: "2/18/2020"
5 output: html_document
6 ---
7 {r message=FALSE}
8 library(tidyverse)
9
10 video_test <- read.csv('video_test.csv')
11
12
13 |
```

Step 17: Now you should test your Markdown file to make sure it works. To do this, click the “Knit” button and then click “Knit to HTML”. Put some text in the body of the document too.



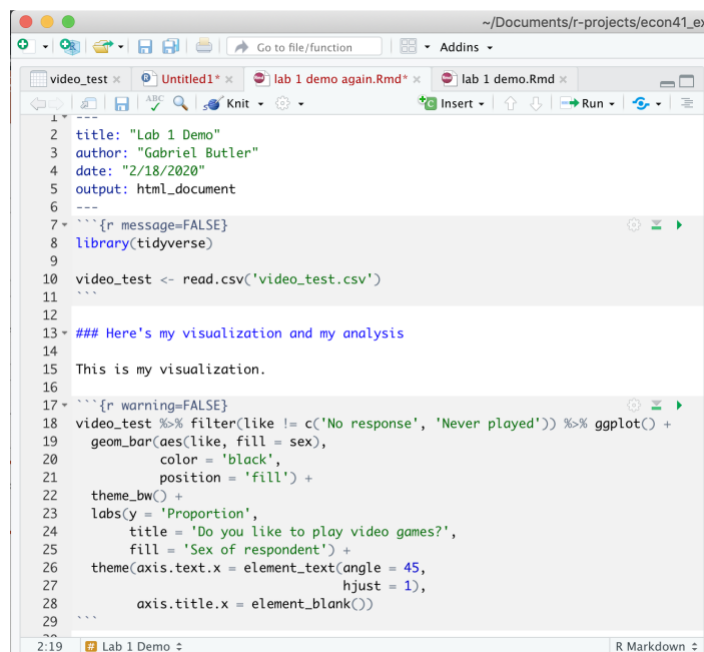
```
1
2
3
4
5
6
7
8
9
10 deo_test.csv')
11
12
13 ### Here's a heading
14
15 Let's see if this document can be generated
```

Step 18: If you've done everything correctly up to this point, when you knit the HTML document, a new window which contains the document should open when RStudio is finished generating it. Here's what mine looks like.

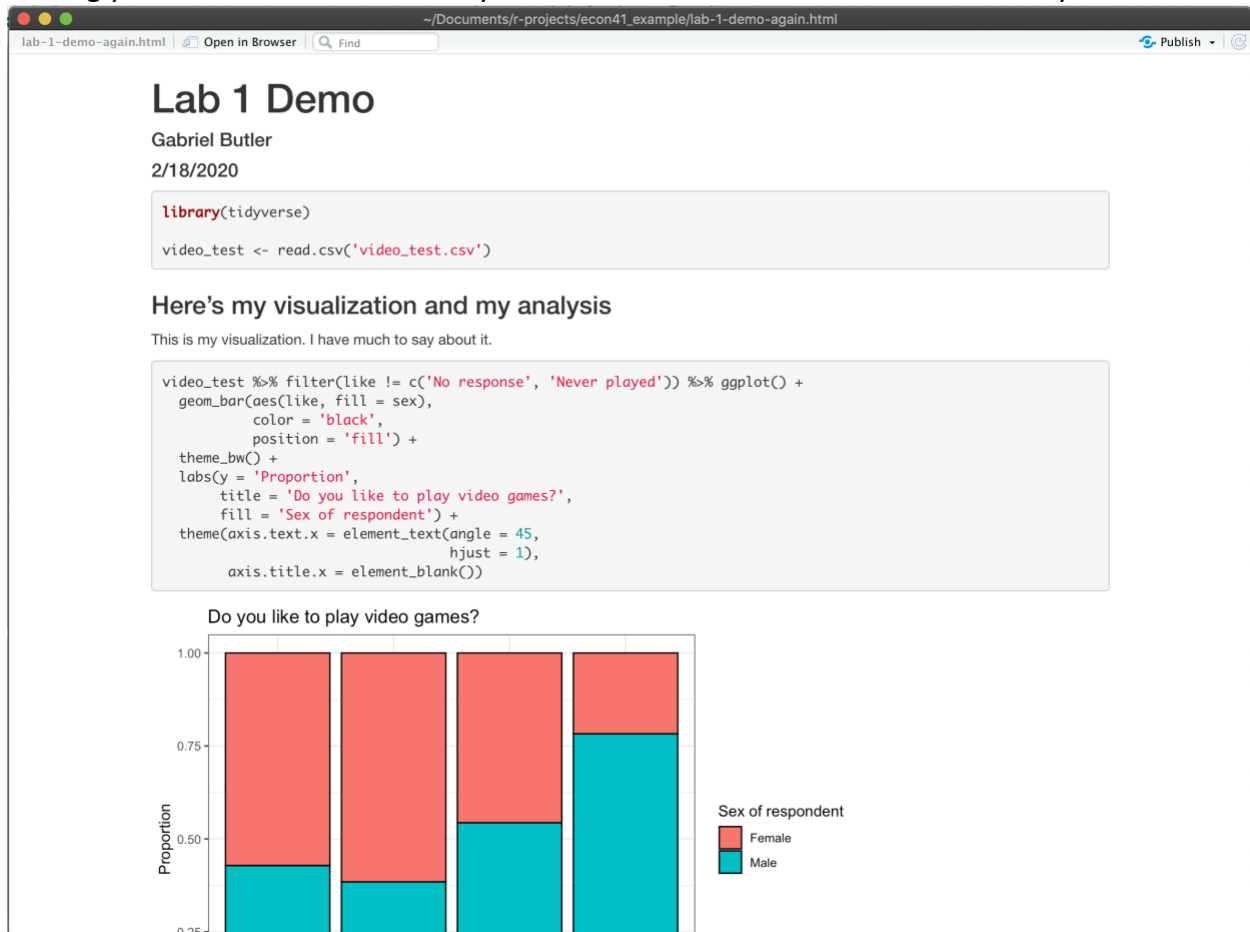


Note: If RStudio threw an error, it's probably because you didn't put your dataset into your project folder yet. If you were unable to generate this document, refer back to steps 5 and 6.

Step 19: Now suppose that you're finished with your analysis and you're ready to start writing your report. All you have to do now is copy and paste working code from your script into your R Markdown document. Recall that in Step 11 I showed some working R code that I wrote inside of an R script. I have copied and pasted that code into an R code chunk in my R Markdown document here.



Step 20: Here is the final result. Notice that all of my code is visible. Please remember that making your code visible is one of my basic standards for labs. Please never hide your code.



I hope this guide will be enough to get you guys started and carry you through the course. In some of your labs you'll have to load data in different ways, but the tutorials in our lab book cover how to do that.