Practice Exercise for Monday morning

Susan Chen

2022-06-06

Contents

Markdown files and reading in data

R Markdown files and getting started

Preliminaries:

I recommend you create a folder on your hard drive called `training`. For example, I created a folder and here is the path to my files: `C:/Users/ecsusan/Documents/training/`. Note the front slashes in the pathname!! If you already have a training folder then go ahead and use that one. Just make sure you can find the path really easily.

Starting with someone else's example:

It is always useful to look at someone else's code to get started. So I have saved on Teams in the training channel a file called `20220606_Training_Exercise.Rmd`. This is the file I used to generate the document you are viewing. You can download this file and open it in Rstudio. This will make it easy to *copy chunks of code from my file to yours* then you can make changes.

If you look at the code in my `R` markdown file `20220606_Training_Exercise.Rmd` you will see the first chunk of code called `setup`. In this chunk (see above code chunk if in Rmd file) you will see a package being called using a `library()` statement. Inside the parentheses you will see for example `library(gapminder)`. This statement is loading the `gapminder` package from an internal `R` library. FYI: Some nice person wrote the code for the `gapminder` package and they saved a dataset in the package so now it is publicly available for all to use when you use the `gapminder` package. This is what a number of you did for the exercise on Friday afternoon.

In real life you *will not* have a nice package to call that already has useful datasets built in. You will have to load your own data from an excel file on your hard drive.

Create a new Rmd file

You are now going to create a new *.Rmd file and call it `20200606Exercise.Rmd`. Save it in your `training` folder.

In your `20200606Exercise.Rmd` copy the first 14 lines of code from *my* `R` markdown file `20220606_Training_Exercise.Rmd`. These 14 lines are the YAML header. Make sure to change the name in the YAML header to your name. Preserve the formatting !!!! Now go ahead and knit your file. Knitting your file does 2 things. It saves it for you and it renders it to an html document like we did on Friday afternoon.

Loading excel data into R from your hard drive

In this exercise, you will load and analyze the data set `2021ACSapp.xlsx`. This is data we used for a 2021 DSPG project. Download this `xlsx` file from Teams and save it on your hard drive in your `training` folder. Now execute the following steps:

1. Read the *.xlsx file into R: Catherine and Riley can show you how to do this as I did this with them on Friday afternoon. Or, check Nathaniel's notes from the classes he taught last week. Or, use this hint: you will want to make sure you load the package `readxl` using the command `library(readxl)`. Then you can read in the excel spreadsheet using the command read_excel("filename") where you substitute the file you want to load including the path. So the command on my PC is `read_excel("C:/Users/ecsusan/Documents/training/2021ACSapp.xlsx")`. Note: you need to know where you stored the file on your computer and make sure you put the correct full pathname. Here is my chunk. I read in the excel spreadsheet and I save it to an object called `mydata2` so I can use it again further down in my R script.

```
library(readxl) #I call the library here instead of in the setup chunk
                #to make it clear you need to load this package to read
                #in an excel spreadsheet
mydata2 <- read_excel("C:/Users/ecsusan/Documents/training/2021ACSapp.xlsx")
```

2. Now that you have read in your excel spreadsheet and saved it as an object in `R`, you can now use it. For example, you can use the `head()` command to print out the first 6 lines of data. You should always do a check to make sure you are reading in your dataset correctly. Below in the code chunk I use the `head()` command:

```
head(mydata2)
```

```
## # A tibble: 6 x 44
##   geoid county_name   percapinc pctdis pcthi pctunemp housingownocc housingtotal
##   <dbl> <chr>             <dbl>  <dbl> <dbl>    <dbl>         <dbl>        <dbl>
## 1  1007 Bibb County,~     20778   17.2  89.3     7.30          5128         6891
## 2  1009 Blount Count~     24747   14.1  89.2     3.40         16423        20847
## 3  1015 Calhoun Coun~     25345   21.2  90.5     8.10         31254        44605
## 4  1017 Chambers Cou~     22729   18.1  89.7     3.90          9072        13448
## 5  1019 Cherokee Cou~     24301   17.8  91.4     4.40          8304        10737
## 6  1021 Chilton Coun~     24658   19.4  87       6.40         12610        16927
## # ... with 36 more variables: pctunder5 <dbl>, pctbet5_9 <dbl>,
## #   pctbet10_14 <dbl>, pctbet15_19 <dbl>, pctbet20_24 <dbl>, pctbet25_29 <dbl>,
## #   pctbet30_34 <dbl>, pctbet35_39 <dbl>, pctbet40_44 <dbl>, pctbet45_49 <dbl>,
## #   pctbet50_54 <dbl>, pctbet55_59 <dbl>, pctbet60_64 <dbl>, pctbet65_69 <dbl>,
## #   edphd <dbl>, edprof <dbl>, edmast <dbl>, edbach <dbl>, edassoc <dbl>,
## #   edscoll <dbl>, edscolllt1 <dbl>, edged <dbl>, edhsdip <dbl>,
## #   edhsnodip <dbl>, countyhh <dbl>, hhpctcompdev <dbl>, ...
```

The lines of data produced by the `head(mydata2)` command show that you have data for a number of US counties. FYI: these are coal mining counties in Appalachia. Other interesting variables are:

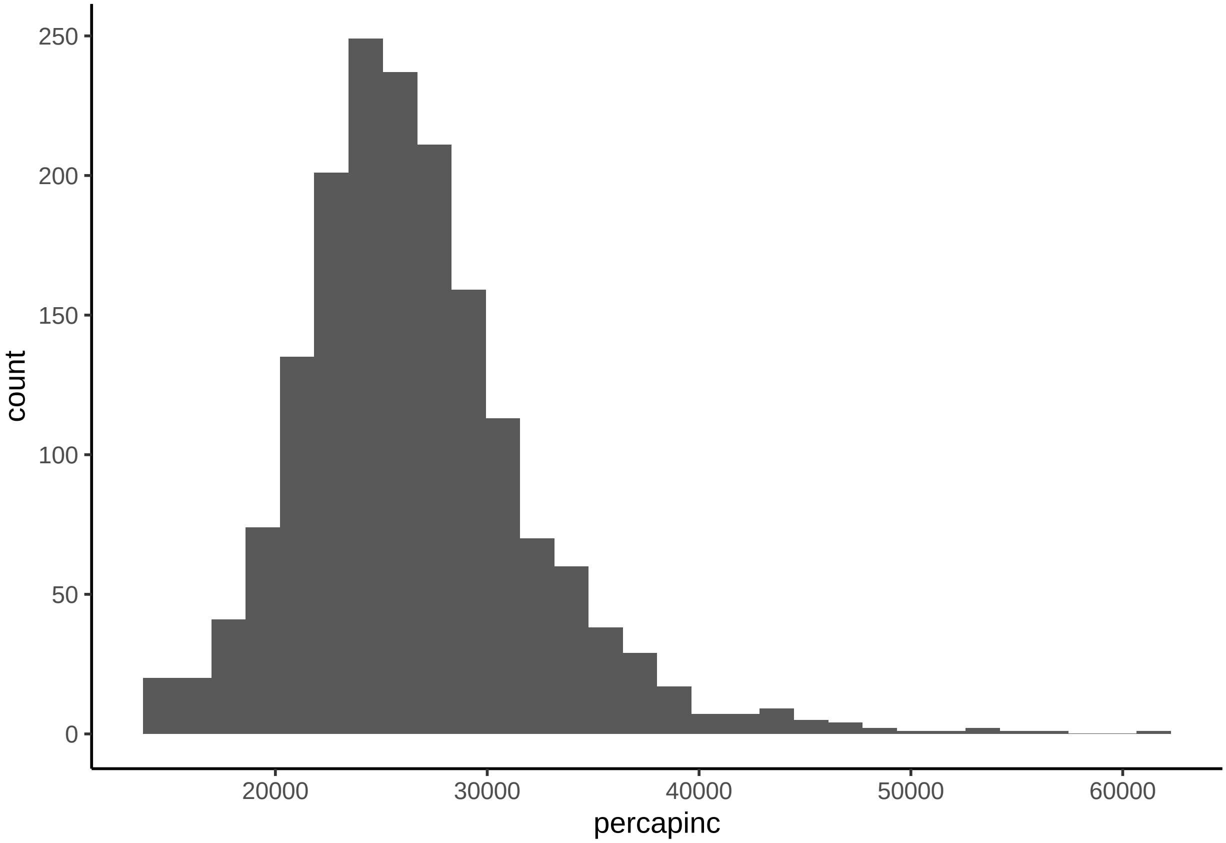
pctdis: percent with a disability <- almv_acs_var("S1810_C03_001") %>% rename(Pct.Dis=estimate) pcthi: percent covered by health insurance pctunemp: percent unemployed housingownocc: estimate of number of people in owner occumpied housing.

There are more variables but these are good for now. Dr. Holmes will show you more about how to get ACS data later today.

Analyzing data

3. Now you can use the ggplot2 package to plot a histogram of per capita income across all these counties (percapinc). See my code chunk below. Go ahead an make a histogram of percapinc.

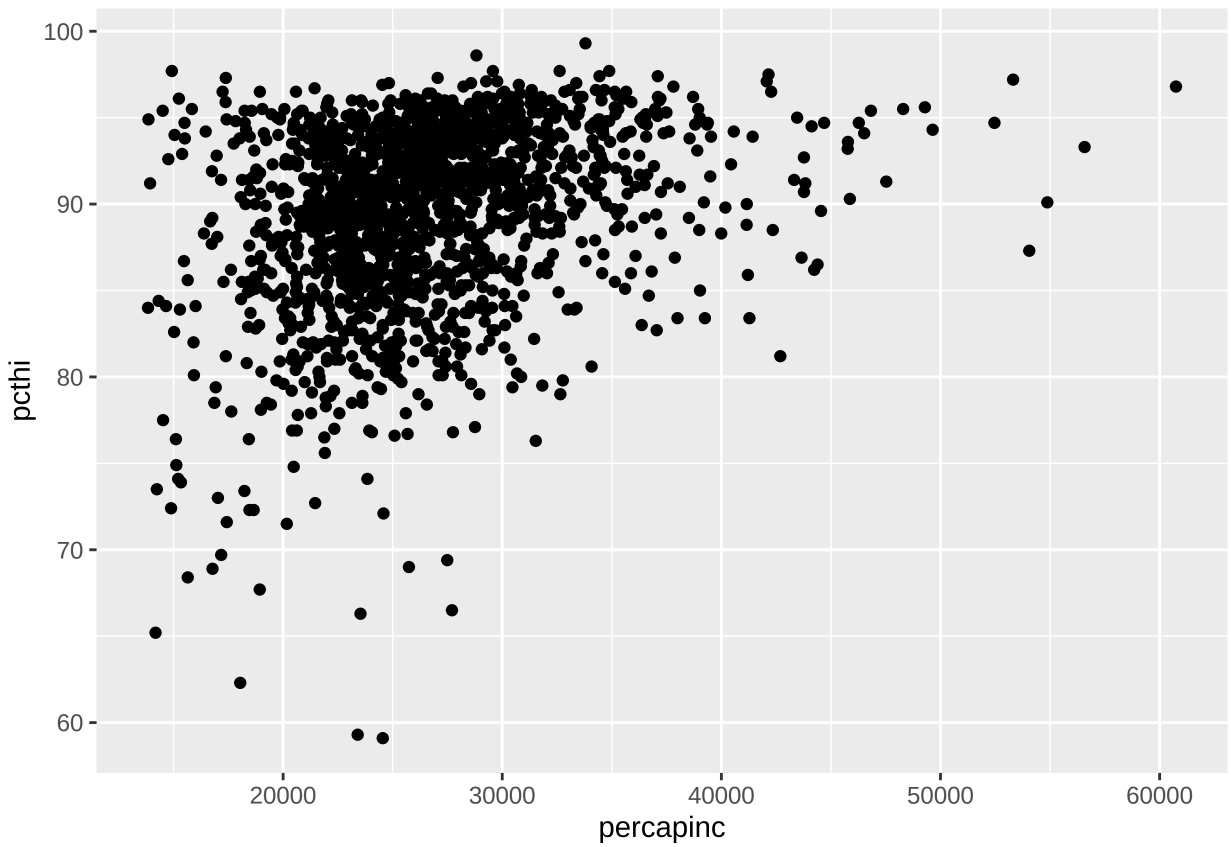
```r
library(ggplot2) #load the ggplot2 package that makes histograms, scatterplots, etc.
ggplot(data=mydata2, aes(x=percapinc)) +
  geom_histogram() + theme_classic()
```



4. Now make a histogram of percent covered by health insurance. You can try doing other variables too (eg. pctdis, pcthi, pctunemp).

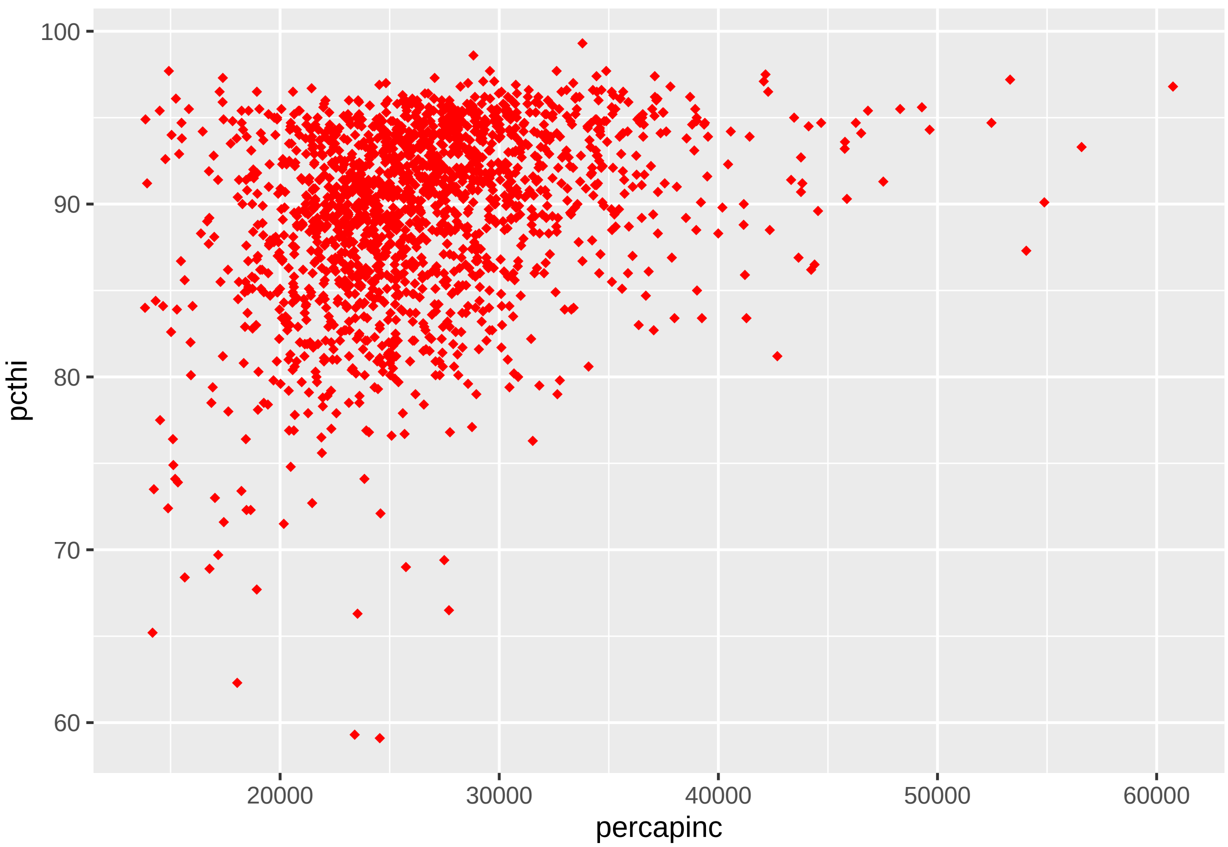
5. Now make a scatterplot of percapinc vs pct health insurance coverage

```r
ggplot(mydata2, aes(x=percapinc, y=pcthi)) + geom_point()
```



6. Now change the color of the points. I did red below but you should do blue. Also try changing the shape to shape=2. Try shape = 0, 3, 25 for fun.

```
ggplot(mydata2, aes(x=percapinc, y=pcthi)) + geom_point(shape=18, color="red")
```



7. Now knit your file to an html document. Click on the Knit button (knitting needle at the top of this page).
8. Now check in your training folder. You should see (1) your excel file, (2) your `20200606Exercise.Rmd` file, (3) the `20200606Exercise.html` you just created.

Congrats you have now created your first html from an Rmd file. You will be doing lots of this over the next few weeks. An Rmd file is awesome because it is a log of your analysis and your code. You can share it with your team mates and use it to help other teams to understand how to code. For example, I took my *.Rmd file and knit it to the pdf you are now reading. I also knitted it an html file also save on Teams. So Rmd files are pretty nice!

Recap

To recap: in this tutorial you learned how to:

1. Read in an excel spreadsheet from your hard drive;
2. How to save it as an object;
3. How to use the object to perform analysis.
4. You did this in R markdown so your steps are completely documented and reproducible when you need it tomorrow.