American University - DATA 312

Instructions for Homework 1-C

Unit 1 - Data Visualization
Part C - Communication with Rmd

Pre-requisite work:
Please finish Homework 1-B prior to starting this assignment.

Work Process:
The main objective for this assignment is to explore how to use R Markdown to create a report based on your work in the previous assignment. However, we will continue to discuss exploratory tools, as you will undoubtedly have more questions about your data now that you are familiar with it. Your final report should demonstrate your ability to interpret your results. This is a purely human activity. No algorithm can do this part for you. Your results may be rather dull, or they may be exciting. The important thing is that they are correct. A correct boring writeup will receive full credit. Focus on only reporting results supported by your data.

The first stage of this process is to convert your previous assignment from an R Script to an R Markdown file. Then, you will "knit" (compile) the R Markdown file to a PDF file which will launch for you to read. As you thin out some of the graphics and results, you will naturally come up with more questions and wish to create more graphics. You will choose which results to present in your writeup. You are encouraged to rearrange, add, and delete them - subject to the project constraints.

1. Open your last assignment (an R Script file) so you can see it in your editor. You may wish to Source it once more to remind yourself of what is in it. Note the name of this file.

INSTALL/CALL knitr

2. We will use the following library, so please check to see if you have it. If not, install it.
```
library(knitr)
```

PREPARE YOUR R SCRIPT

3. Before we convert our R Script, we should break our work into parts which are called "chunks" in R Markdown. In your writeup, you will need places to type words between your graphics, but you won't keep all your graphics. You will be able to adjust your chunks as you go along. However, making chunks is easier at this stage, so let's be generous and create lots of chunks now, knowing we can always combine them later.

   Between each of your graphics or results, place a line containing simply these two characters:
   ```
   #+
   ```

4. To preserve your sanity, remove any lines where you asked for help on a topic, like ?dataname or ?geom_bar. If you leave those in, R will keep launching help pages on your browser when you knit your file. You probably don't want to keep your View(dataname) lines either, if you still have those. Comments in your .R file will not cause issues in your .Rmd file, so leave those in.

CONVERT .R to .Rmd

5. Using the knitr package, we could convert between several file types. We will only convert from R Script to R Markdown in this assignment. Use your filename in the command below. Your R Markdown file will be in the same directory as your R Script file and will have the same name but a different extension. Your new file will end in .Rmd instead of .R.

   knitr::spin(***filename*** , FALSE, format = "Rmd")

6. Find and open the new .Rmd file you have just made. Notice that each separator you added at the previous step has caused a chunk to end and a new chunk to start. Here is an example chunk:

   ~~~
   ```{r}
   library(tidyverse)
   ```
   ~~~

   You should think of your R Markdown file as being a bit like the opposite of your R Script. In your R Script, all the lines that were not specially marked were R code, and you had to specifically mark comments so R didn't try to execute them. Now, the default assumption is that you are writing a document and any R code must be specifically marked as such.

   You might be asking yourself what else could be used in the {} other than {r}. If you are asking this, you are really thinking ahead! You can actually put code for languages other than R in your .Rmd files. Pretty cool right!?

7. Next, we will knit to a pdf. Note: As before, you can run this document or any portion of it! You will notice you have the option to knit to html or doc. Explore those options on your own. They are very similar to what we're doing today with pdf.

   To knit to a pdf, just click on the "Knit" button and choose "PDF". This button is near a blue ball of yarn at the top of your .Rmd file.

   The results of your first knit will be poor. Just make sure the process itself works. A pdf viewer should launch and show you the completed pdf file, which is also stored in the same directory with your .R and .Rmd files. You should be sure to close this file prior to knitting again, or you will get an error message.

CONTROLLING .Rmd

8. You may notice quite a bit of output that you do not wish to see. Right now, we will shut off all the output for all the chunks.

   Use *Edit > Replace and Find* to replace {r } with
   {r echo=FALSE, message=FALSE, error=FALSE, warning=FALSE}

   To turn the output back on, turn FALSE values to TRUE or just delete all the options, leaving only "r".
   If your diagrams run off the page, you may need to insert new lines between chunks.
   Save often. Control-S will do this nicely for you!

9. This step is optional. You may find that the graphics are too wide for you. The knit process is presuming your paper is 8.5x11 inches, so if you make 6" wide graphics, this may look nicer to you. Add this chunk just after your heading, and experiment.

~~~
```{r include=FALSE}
knitr::opts_chunk$set(fig.width=6)
```
~~~

10. Create a heading for your paper, complete with abstract. (You will be told specifically what to put in your abstract.) Edit the heading at the top of your .Rmd file so it looks like this. The "thanks" option will add a footnote to your title. Remove the # if you want to use it. Be very careful about the use of “whitespace” (space, tab, returns) in this header zone! The words “output”, “title”, “author”, etc must all begin on the first character of their lines.

````
---
output:
  pdf_document: default
  html_document: default
title: "Homework 1C - STAT 312"
#thanks: "No thanks!"
author: "Your Name"
abstract: "Abstract is in Quotes! Do not hit return while typing it!"
date: '`r format(Sys.time(), "%B %d, %Y")`'
---
````

11. You should exert control over the paging. Ideally, you want your text to be right above or right below the graphic it is about. Use one of these commands to break the page:

```
\pagebreak
\newpage
```

For our purposes, these commands do the same thing. However, they sometimes behave differently. One will cause the end the page abruptly, while the other will permit the remaining content to be spread evenly over the remaining space on the page.

VARIABLES AS LITERALS

12. One of the best reasons for using your programming environment as your document creation tool is that you can be sure the results are really from your data and are reproducible. A way to assure yourself that you didn't type something incorrectly is to simply dump the results of your calculations directly into your writeup! If the underlying data change, you just run everything again, and the new values automatically show up in your document! Here is a short example in .Rmd.

````
The area of a circle with a radius of 9 is `r 9*9*pi`.
````

And here is a more interesting example.

~~~
```{r}
theAverageIncome = mean(my_data_frame$income, na.rm=T)
```
The average income was found to be $`r format(round(theAverageIncome,2),nsmall=2)`.
~~~

This second example includes a dollar symbol (which is literally a dollar sign, not a special character), a rounding command, and the requirement that we will retain at least two digits after the decimal.

CORRELATION MATRICES

13. As you continue exploring your data, you may wish to complete a correlation matrix. These are quite useful, as they will help you see which (if any) of your numerical variables are closely connected. Unfortunately, you have to pre-process your data so you get rid on any non-numerical columns prior to doing this. So, make a copy of your data, then delete all the columns you don't want to keep.

```
sapply( dataset , class) #  This will tell you the type of each variable, based on how it's stored.
mycopy <- dataset
mycopy$bye = NULL # This removes a variable named "bye".
cor ( mycopy )          # This is the correlation matrix request.
```

You can retain any binary or factor variables as long as they're stored numerically. Then, R will give you back a correlation matrix.

as.factor(), as.numeric() etc.

14. There are situations in which you want a certain type of calculation or tool for your data, but according to R, your data type is incorrect. It may be fine, but R thinks otherwise. In this case, you can "wrap" it in a function that will "cast" it to another type. In this example, the owned variable is a binary variable which R has interpreted as a number. Try this plot with and without the as.factor option to see the difference. You will need to install and invoke the Ecdat library to run this.

```
ggplot(data=Workinghours)+
        geom_bar(mapping=aes(x=as.factor(owned),     fill=occupation),position="fill")
```

Project Requirements

15. The Heading: Keep the title provided in the above template for your submission. Edit the template to include your own name. The date should be automatic. For an abstract, state that you are exploring the dataset (give its name) included in the R library (give its name). Then, list all variables in the data set by type. If you have more than 10 variables in your data set, please give up to 5 categorical variables and up to 5 numerical variables. As explained, many variables can be used in more than one way. In the abstract, report the variable type you feel is most natural, not what R claims it is.

16. Call any libraries right after you finish your heading (or just after your command to control the size of your diagrams).

17. Variable introduction: Call the following three commands (in any order) to introduce your data. For these chunks, we want to see the echo of the commands, so just use a plain {r} to start them.

    head(*dataset*)

    summary(*dataset*)

    sapply(*dataset*, class)

18. Credit the original data source if this information is available. (Usually it is!) To create a block quote, use the > symbol at the start of a line and don't hit return until you're done with the block quote. If you can't find the original data source, ask Google. If that still doesn't help, at least make sure it's clear which R library you used and the name of the dataset. Explain what each of the variables means in the dataset. For example, if the variable name is "income", don't just say it's "income", but whose income is it? What are the units? Some variables will be easier to explain than others, and often you will need to read the references to figure this out.

19. The purpose of this stage is to visually explore each individual variable, but if you add a context, that is fine as well. For each numerical variable, make a histogram. For each categorical variable, make a bar graph. For this step, you may use either the base R commands or the tidyverse options. Be sure that either the title or one of the axes makes it clear what is going on. You may just make these plots boring, or if you wish, you may opt to color them by other variables or do similar things. If you prefer pie charts for categorical variables, that's fine.

20. At least once in your document, make use of a variable directly in your written work using `r variable` as shown. This can be done anywhere in the document, but it's probably easiest right after you've introduced your variables.

21. Include at least one (tidyverse) scatterplot which contains a coloring by another variable, or a Loess curve, or both. Include a title and check to see that the x and y labels make sense. If not, override the defaults and set them. Make sure your title and/or axes labels fully explain what the graphic is representing. Optional: report the p-value for a linear fit of this data, if that is meaningful in your context.

22. Create a geom_bar plot which allows you to easily compare two categorical variables. Then, switch the order of the two variables for a second geom_bar plot. Here is an example. Note the use of as.factor(). It is not always necessary to use as.factor(). Make sure your title and/or axes labels fully explain what the graphic is representing. Optional: report the p-value for a chi-square test on these variables, if you feel it is meaningful.

    ```
    ggplot(data=Workinghours)+
            geom_bar(mapping=aes(x=occupation,   fill=as.factor(owned)), position="fill")
    ```

23. Create a geom_boxplot or a stat_summary plot, where the graphic extends from the min to the max of the data range. For this requirement, you may also choose to use the base R boxplot if you prefer. In any case, be sure the variable names and/or plot title fully explain what the graphic is representing. Optional: report the p-value for a t-test or ANOVA if you feel it's meaningful in this context.

24. In a summarizing paragraph, give the main findings for your analysis. Use page breaks as needed to assure flow of the document. Avoid the case where there is some text on one page, with lots of space after it, and the associated diagram is on the next page.

25. For submission, include your .Rmd file as well as your .pdf file.