Data Wrangling in `R`

STA 360: Assignment 1, Fall 2020

Due Friday August 21, 5 PM Standard Eastern Time

Today's agenda: Manipulating data objects; using the built-in functions, doing numerical calculations, and basic plots; reinforcing core probabilistic ideas.

General instructions for homeworks: Please follow the uploading file instructions according to the syllabus. You will give the commands to answer each question in its own code block, which will also produce plots that will be automatically embedded in the output file. Each answer must be supported by written statements as well as any code used. Your code must be completely reproducible and must compile.

Advice: Start early on the homeworks and it is advised that you not wait until the day of. While the professor and the TA's check emails, they will be answered in the order they are received and last minute help will not be given unless we happen to be free.

Commenting code Code should be commented. See the Google style guide for questions regarding commenting or how to write code https://google.github.io/styleguide/Rguide.xml. No late homework's will be accepted.

R Markdown Test

0. Open a new R Markdown file; set the output to HTML mode and "Knit". This should produce a web page with the knitting procedure executing your code blocks. You can edit this new file to produce your homework submission.

Working with data

Total points on assignment: 10 (reproducibility) + 22 (Q1) + 9 (Q2) + 3 (Q3) = 44 points

Reproducibility component: 10 points.

1. (22 points total, equally weighted) The data set **rnf6080.dat** records hourly rainfall at a certain location in Canada, every day from 1960 to 1980.

a. Load the data set into R and make it a data frame called `rain.df`. What command did you use?

I used the command, read.table, to load the data set into R and make it a data frame called rain.df.

```r
library(tidyverse)  #load in tidyverse package
data_file_url = "https://raw.githubusercontent.com/resteorts/modern-bayes/master/homeworks/homework-1/da
rain.df <- read.table(file = data_file_url, header = FALSE, sep = "")  #read in data
```

b. How many rows and columns does `rain.df` have? How do you know? (If there are not 5070 rows and 27 columns, you did something wrong in the first part of the problem.)

rain.df has 5070 rows and 27 columns and the reason I know is because of the output from the commands, nrow(x) and ncol(x) which output the number of rows/columns respectively of the given argument, x, which in this case was rain.df.

```r
nrow(rain.df) #nrow gets the number of rows of rain.df
```

```
## [1] 5070
```

```r
ncol(rain.df) #ncol gets the number of cols of rain.df
```

```
## [1] 27
```

c. What command would you use to get the names of the columns of `rain.df`? What are those names?

The command I would use to get the names of the columns is "colnames(x)". The output of that command is below where the names of those columns are V1…V27 respectively.

```r
colnames(rain.df, do.NULL = FALSE) #colnames return the column names of rain.df
```

```
##  [1] "V1"  "V2"  "V3"  "V4"  "V5"  "V6"  "V7"  "V8"  "V9"  "V10" "V11" "V12"
## [13] "V13" "V14" "V15" "V16" "V17" "V18" "V19" "V20" "V21" "V22" "V23" "V24"
## [25] "V25" "V26" "V27"
```

d. What command would you use to get the value at row 2, column 4? What is the value?

The command I used to get the value at row 2, column 4 is `rain.df[row, col]` where in this case x and y are 2 and 4 . The value is 0 based on the output.

```r
rain.df[2, 4] #obtain the value at row 2, and 4 using bracket notation
```

```
## [1] 0
```

e. What command would you use to display the whole second row? What is the content of that row?

The command I used to get the entire second row, is similar to the one above, except I didn't give a column argument. The content of the row is displayed below:

```r
rain.df[2,] #display the second row by only providing a row argument
```

```
##   V1 V2 V3 V4 V5 V6 V7 V8 V9 V10 V11 V12 V13 V14 V15 V16 V17 V18 V19 V20 V21
## 2 60  4  2  0  0  0  0  0  0   0   0   0   0   0   0   0   0   0   0   0   0
##   V22 V23 V24 V25 V26 V27
## 2   0   0   0   0   0   0
```

f. What does the following command do?

```r
names(rain.df) <- c("year","month","day",seq(0,23)) #change the column names to proper descriptors
```

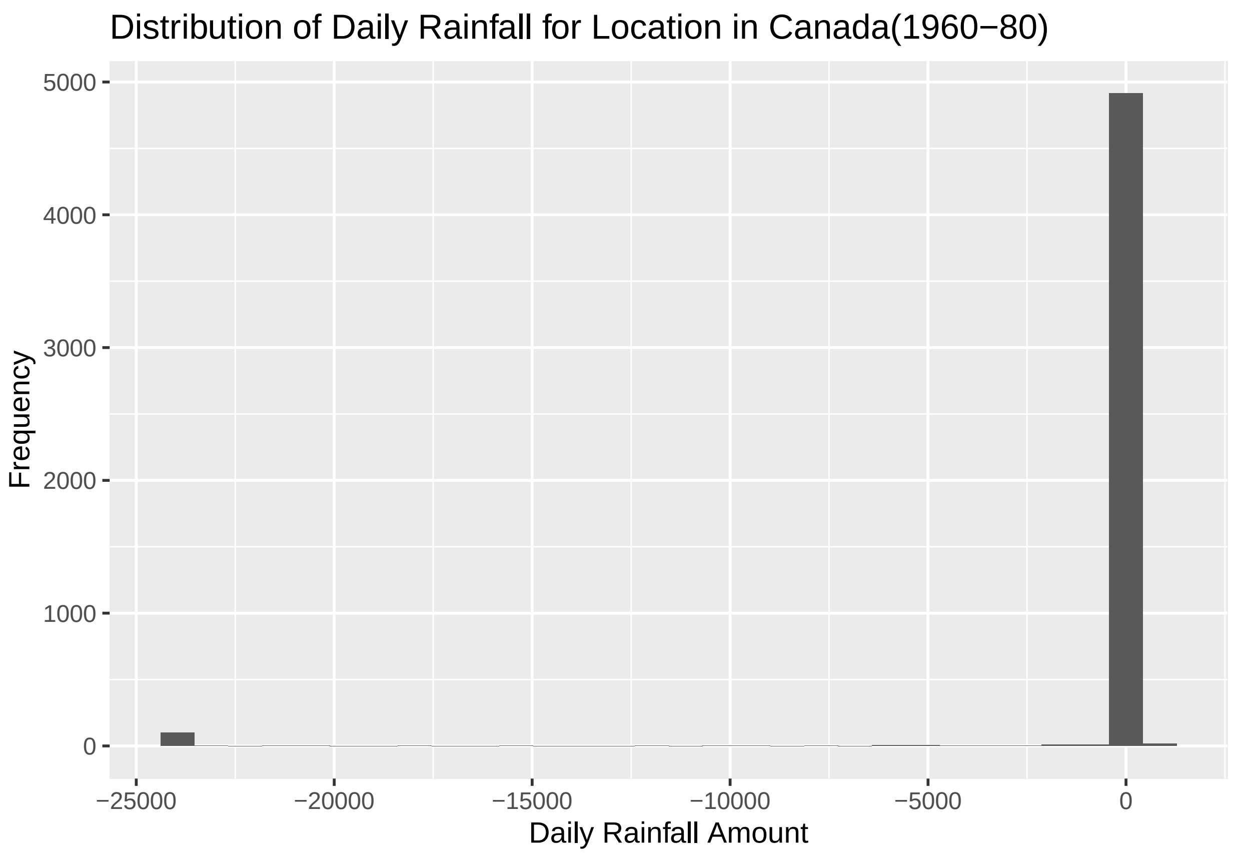
This command sets the column names of rain.df, to be (going from left to right), 'year', 'month', 'day', and then 0-23 with 0 representing the time 12AM - 1AM, 1 representing 1AM-2AM and etc, all the way to 23 where 23 represents 11PM to 12AM. So instead of the generic column name of V1- V27, now, the column names are more descriptive of what values are stored in each respective column. The c() command takes the arguments and forms them into a vector which is then set as the column names of rain.df.

g. Create a new column called `daily`, which is the sum of the 24 hourly columns.

```r
rain.df <- cbind(rain.df, daily = rowSums(rain.df[4:27])) # add new column daily
```

h. Give the command you would use to create a histogram of the daily rainfall amounts. Please make sure to attach your figures in your .pdf report.

```r
ggplot(data=rain.df, mapping = aes(x = daily)) + #create histogram with specified data being daily
  geom_histogram() +
  labs(title = "Distribution of Daily Rainfall for Location in Canada(1960-80)",
       x = "Daily Rainfall Amount", y = "Frequency") #set histogram labels
```



i. Explain why that histogram above cannot possibly be right.

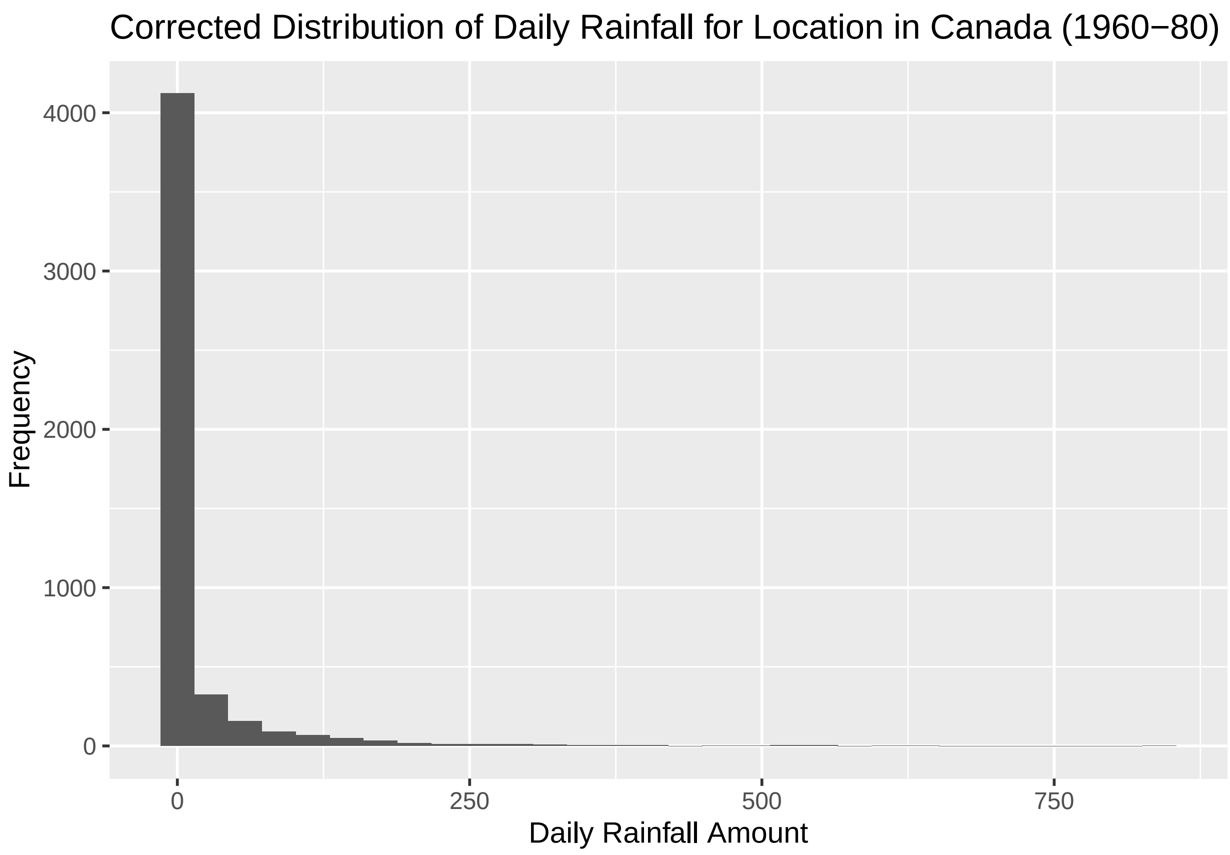
The histogram above cannot possibly be right because you cannot have a negative daily rainfall amount. You should only have amount that are greater than or equal to 0.

j. Give the command you would use to fix the data frame. In order to fix the data frame, we should replace all the negative values (-999) and such with NA. If we were to replace the negative values with 0, we could skew the distribution. The commands below find the values that are less than 0 and replace them with NA. The second command then recalculates the daily column as now the results should be different with no negative daily rain amounts.

```r
rain.df[rain.df < 0] <- NA #replace negative values to be NA
rain.df$daily<-rowSums(rain.df[4:27]) #re-calculate daily column
```

k. Create a corrected histogram and again include it as part of your submitted report. Explain why it is more reasonable than the previous histogram.

```r
ggplot(data=rain.df, mapping = aes(x = daily)) + #same command as above, but daily is different
  geom_histogram() +
  labs(title = "Corrected Distribution of Daily Rainfall for Location in Canada (1960-80)",
       x = "Daily Rainfall Amount", y = "Frequency")
```



This corrected histogram is more reasonable because it doesn't have any negative values, instead, all the daily rainfall amounts are greater than or equal to zero. This is more reasonable, and makes more sense to the viewer, because people equate 0 rainfall as no rain. Having negative rainfall values would confuse most as to what that means. Since this histogram doesn't include those negative values, it's more reasonable and easier to interpret.

Data types

2. (9 points, equally weighted) Make sure your answers to different parts of this problem are compatible with each other.

a. For each of the following commands, either explain why they should be errors, or explain the non-erroneous result.

```
x <- c("5","12","7")
max(x)
sort(x)
sum(x)
```

The first line of code sets x as a vector with the values "5", "12", and "7". The second line of code returns "7" as the max instead of 12 as you might expect because the values are not as integers but as strings, so now R compares them lexicographically, in which "7" is the greatest. For sort(x), again because the values are strings, R will order them in lexicographic order in which "12" is the smallest, and "7" is the greatest. For sum(x), sum only excepts numeric or complex or logical vector as an argument, and since our vector x, is not, it throws an invalid 'type' error.

b. For the next two commands, either explain their results, or why they should produce errors.

```
y <- c("5",7,12)
y[2] + y[3]
```

These lines of code will produce an error because R vectors can only contain one type of element. In this case, it looks at "5" as a character and then converts the rest of the values in the vector into character types as well. The next line attempts to add 7 and 12 , however, because they are now character types, R throws an error since you can't add character types together.

c. For the next two commands, either explain their results, or why they should produce errors.

```
z <- data.frame(z1="5",z2=7,z3=12)
z[1,2] + z[1,3]
```

Here, the lines of code don't produce any errors because data frames support multiple types unlike vectors. z[1,2] equates to value at row 1 column 2, which in this case our data frame has 1 row and 3 columns, so the value is 7. z[1,3] equates to the value at row 1 column 3 which is 12. So 7 + 12 is equal to 19 which is the expected output of these commands.

3. (3 pts, equally weighted).

a.) What is the point of reproducible code?

The point of reproducible code is that it allows for others to "double-check" your work per se. Other people can run your code and ensure they get similar results as you. This also promotes transparency as others can see exactly how you arrived at your results.

b.) Given an example of why making your code reproducible is important for you to know in this class and moving forward.

Making code in this class reproducible, especially in this virtual environment, is to ensure that the code written is actually producing the results you are submitting. For example, someone could easily submit another person's PDF as their own if no code is required to be submitted. But because it is required, a TA/Professor could check the code, run the code, and see if the generated PDF matches what the person submitted. If the generated results from the code match what the person submitted, it's most likely they produced the work on their own. If the results don't match, the person most likely didn't do the work to produce the results. Going forward, having reproducible results is important for integrity and for transparency not only in academia but also in industry as well.

c.) On a scale of 1 (easy) – 10 (hard), how hard was this assignment. If this assignment was hard (> 5), please state in one sentence what you struggled with.

This assignment is a 6 as I found it difficult to remember some of the functions/methods in R which slowed me down a bit in completing this assignment.