Homework 6

David Edwards

2019-10-15

Problem 3

a. Create a function that computes the proportion of successes in a vector. Use good programming practices.

This function is just counting up the number of non-zero values in the vector and then dividing that number by the length of the vector.

b. Create a matrix to simulate 10 flips of a coin with varying degrees of "fairness" (columns = probability) as follows:

```
set.seed(12345)
P4b_data <- matrix(rbinom(10, 1, prob = (30:40)/100), nrow = 10, ncol = 10, byrow = FALSE)
```

Fairly straight forward, just run the code above.

c. Use your function in conjunction with apply to compute the proportion of success in P4b_data by column and then by row. What do you observe? What is going on?

All of the columns have the same proportion of success and each of the rows either have all 1's or all 0's. It would appear that our matrix is not the random values we were looking for!

d. You are to fix the above matrix by creating a function whose input is a probability and output is a vector whose elements are the outcomes of 10 flips of a coin. Now create a vector of the desired probabilities. Using the appropriate apply family function, create the matrix we really wanted above. Prove this has worked by using the function created in part a to compute and tabulate the appropriate marginal successes.

The BinomOutComes10 will take in a probability value and return simulated data of 10 Bernoulli trials given that probability. I then apply this to the probabilities given above (there are actually 11 probabilities given above but since problem asks for 10X10 matrix I dropped the value .3). After getting my matrix of 10 trials for a Bernoulli at 10 different probabilities I then applied my function to compute the proportion of success to each column. Of course this won't come out to be the exact values chosen but it will be close to 0.4 for each of them.

Problem 4

In Homework 4, we had a data set we were to compute some summary statistics from. The description of the data was given as "a data set which has multiple repeated measurements from two devices by thirteen Observers". Where the device measurements were in columns "dev1" and "dev2". Re-import that data set, change the names of "dev1" and "dev2" to x and y and do the following:

1. create a function that accepts a dataframe of values, title, and x/y labels and creates a scatter plot
2. use this function to create:
   (a) a single scatter plot of the entire data set
   (b) a separate scatter plot for each observer (using the apply function)

I would have liked to have found away to add in the Observer number to the title of each scatter plot but was unable to find a way to do this without a loop.

Problem 5

Our ultimate goal in this problem is to create an annotated map of the US. I am giving you the code to create said map, you will need to customize it to include the annotations.

Part a. Get and import a database of US cities and states. Here is some R code to help:

```
    #we are grabbing a SQL set from here
    # http://www.farinspace.com/wp-content/uploads/us_cities_and_states.zip

    #download the files, looks like it is a .zip
    library(downloader)
    ##download("http://www.farinspace.com/wp-content/uploads/us_cities_and_states.zip",dest="us_cities_
    unzip("us_cities_states.zip", exdir= "C:/Users/dle08/OneDrive/Documents/VT Classes/5044_Regression/

    #read in data, looks like sql dump, blah
    library(data.table)
    states <- fread(input = "./us_cities_and_states/states.sql",skip = 23,sep = "'", sep2 = ",", header
    ### YOU do the CITIES
    ### I suggest the cities_extended.sql may have everything you need
    ### can you figure out how to limit this to the 50?

    ## Code for cities
    cities <- fread(input = "./us_cities_and_states/cities_extended.sql",skip = 26,sep = "'",
                    sep2 = ",", header = F, select = c(2,4,6,8,10,12))
    names(cities) = c("City", "State_Code", "Zip", "Latitude", "Longitude", "County")
```

Part b. Create a summary table of the number of cities included by state.

For this problem I just used the tapply function to index by the state codes and compute the length of each vector of all the cities in each state. The output is given as a kable table.

Part c. Create a function that counts the number of occurrences of a letter in a string. The input to the function should be "letter" and "state_name". The output should be a scalar with the count for that letter.

Create a for loop to loop through the state names imported in part a. Inside the for loop, use an apply family function to iterate across a vector of letters and collect the occurrence count as a vector.

```
    ##pseudo code
    letter_count <- data.frame(matrix(NA,nrow=51, ncol=26))
## I switched this up to loop through all the letters instead of all the states.  For some reason
## I couldn't get my function, countLetters, to read in the letters correctly as a vector.  Also
## I included 51 rows in the matrix as the original data includes the District of Columbia
    for(i in 1:26){
        letter_count[,i] <- sapply(states$V2, countLetters, letters[i])
    }

    ## Change the names of the rows and columns in the matrix.
```

```
    colnames(letter_count) <- letters
    rownames(letter_count) <- states$V2
```

Part d.Create 2 maps to finalize this. Map 1 should be colored by count of cities on our list within the state. Map 2 should highlight only those states that have more than 3 occurrences of ANY letter in their name.

Below is my adapted code from what was provided. I have created a new data frame with the state names (in lower case) and the values computed above for the number of cities in each state.

The legend at the bottom seems to be a little compressed but I haven't been able to find a way to fix it. Also, it would be nice to be able to include other colors and not just different shades of blue. (Note: for some reason when I knit this to pdf it does not include the maps. I have included them as separate files.)

Below is my code for the map that shows states that have the same letter 3 or more times in their name. First I apply the max function to the rows of my letter count matrix. After that I determine which of these are greater than or equal to 3. Finally I create a new data frame that has all of the states names (in lower case) and the true/false values computed for each state.