Introduction to Data Science Midterm Exam Fall 2018

No electronics. Closed notes. You may use the distributed crib sheet. To receive full credit on multi-point problems, you must thoroughly explain how you arrived at your solutions.

Use the back of the page for extra space if you need it. If you do, note this on the exam so we know where to look for your answers.

The exam is worth 62 points and has 7 pages.
Good luck!

Class ID: _____

Question 1 (20 points)

(a) [2 pts] Generate the vector of values without using the catenate function (i.e., don't use the c() function): `2 2 4 4 6 6 8 8 10 10 12 12`

(b) [2 pts] Generate the vector of values without using the catenate function (i.e., don't use the c() function): `2 3 4 6 7 8 10 11 12 14 15 16 18 19 20`

(c) [2 pts] Give a line of R code that creates a numeric vector called `pattern` that contains the following numbers in the order `[1 3 5 1 3 5 1 3 5]`

(d) [2 pts] Give a line of R code that creates a numeric vector called `pattern.subset` that contains the remainders when `pattern` is divided by 3.

(e) [2 pts] Give a line of R code that removes the values in `pattern.subset` that are not divisible by 2 (including 0). This should be a general calculation that would carry out this operation on any vector.

(f) [2 pts] Give a line of R code that changes every second element of `pattern.subset` to `0`. Again, make sure this is a general calculation that would do this for any vector.

(g) [2 pts] Write out `pattern.subset`, after the above calculations (parts (c) through (f)).

(h) [2 pts] Now give a line of R code that generates a vector called `normsamps` containing 1000 random samples from a normal distribution with mean 1 and SD 2. Then give one line of R code to calculate the standard deviation of `normsamps`.

(i) [2 pts] Using implicit coercion of logical to numeric, give a line of R code that calculates what fraction of the values in `normsamps` are less than 3.

(j) [2 pts] Why would we want to carry out the commands in part 1(i)? In other words, what would the analysis in (i) tell us?

Question 2: Linear Regression (12 points)

We will work with the "cars" dataset in R. Here are the variable descriptions and summary statistics:

```
[,1]  speed numeric    Speed (mph)
[,2]  dist  numeric    Stopping distance (ft)

> summary(cars)
     speed           dist
 Min.   : 4.0   Min.   :  2.00
 1st Qu.:12.0   1st Qu.: 26.00
 Median :15.0   Median : 36.00
 Mean   :15.4   Mean   : 42.98
 3rd Qu.:19.0   3rd Qu.: 56.00
 Max.   :25.0   Max.   :120.00
```

For this question, we are interested in predicting stopping distance (dist) from speed.

(a) [2 pts] What would a linear model look like (expressed in math only) that predicts stopping distance from speed?

(b) [2 pts] Give the R code that implements this model.

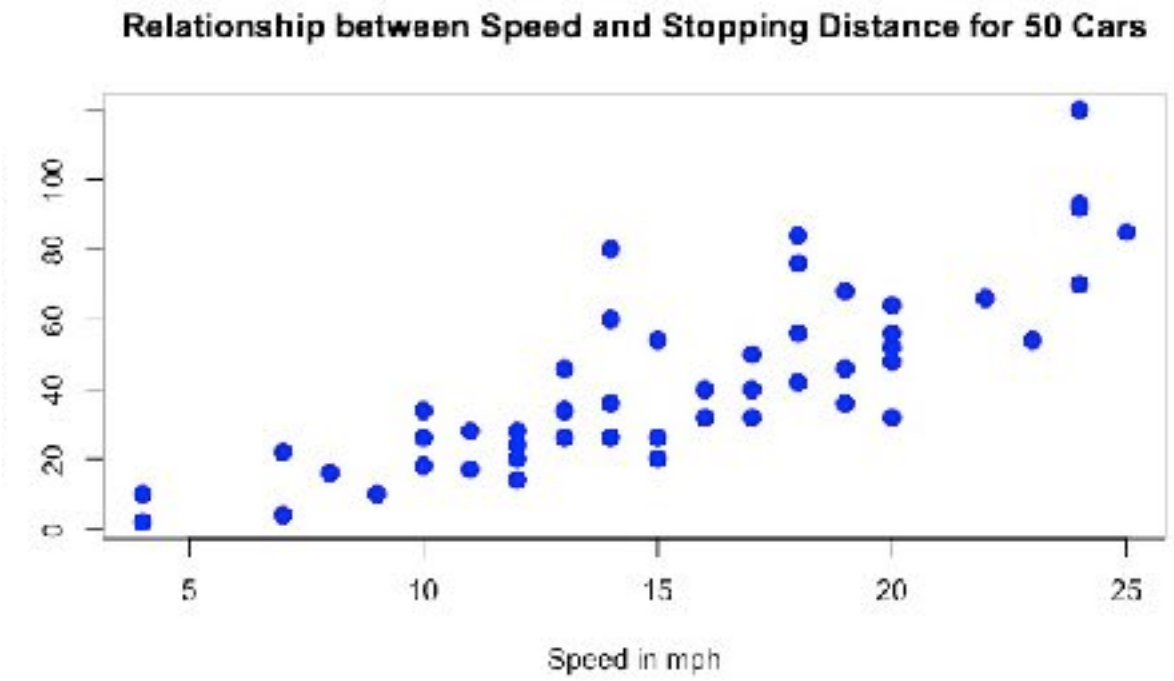
(c) [4 pts] After we run the R code in (b) suppose we get the following information (in part):

```
Coefficients:
            Estimate Std. Error t value Pr(>|t|)
(Intercept) -17.5791     6.7584  -2.601   0.0123 *
speed         3.9324     0.4155   9.464 1.49e-12 ***
```

What is the interpretation of -17.5791? What is the interpretation of 3.9324?

(d) [4 pts] Here is a plot of the data, generated with the following R code:

```
> plot(cars, col='blue', pch=20, cex=2, main="Relationship between
Speed and Stopping Distance for 50 Cars", xlab="Speed in mph”)
```



Give two ways to improve this plot, based on our discussions in class.

Question 3 (10 points)

Suppose we are still interested in the dataset "cars" in R, and we've just executed the following:

```
> class(cars)
[1] "data.frame"

> head(cars)
  speed dist
1     4    2
2     4   10
3     7    4
4     7   22
5     8   16
6     9   10
```

We want to create a vector containing the sum of the square root of the entries in each column of `cars`. Write R code to do this in two different ways:

(a) [5 pts] using a for loop

(b) [5 pts] using the sapply function

Question 4 (20 points)

Write a function called `my.reshape` that takes a general 2 dimensional array as an input and gives a dataframe as an output with each row of the data frame containing all the elements of the first dimension of the array as follows:

```
> our.array
[[1]]
, , 1
          [,1]      [,2]      [,3]      [,4]
[1,]  -88.4463  -84.2969  -84.2969  -88.4463

, , 2
         [,1]     [,2]    [,3]    [,4]
[1,]     NULL     NULL    NULL    NULL

[[2]]
, , 1
         [,1]     [,2]     [,3]     [,4]
[1,] 108.3619 108.4818 108.4818 108.3619

, , 2
          [,1]      [,2]      [,3]      [,4]
[1,]   -6.5370   -6.5370   -6.4391   -6.4391
```

We would like a dataframe as follows:

```
> our.dataframe
        X1       X2       X3       X4      X5      X6       X7      X8
1 -88.4463 -84.2969 -84.2969 -88.4463      NA      NA       NA      NA
2 108.3618 108.4817 108.4817 108.3618 -6.5370 -6.5370 -6.4391 -6.4391
```

This function has two parameters, `x`, which is the required array, and `na.rm`, which has a default value of `FALSE` and is used to specify whether any `NA`s in the array are to be ignored in the calculation or not. If the `NA`s are not ignored and there are `NA`s present, then we return `NA`s in the dataframe for them. The function should convert any NULL values to NA.

Bonus question: After writing this function, suggest some tests, written in `R`, you would give a user who would like to learn to use your `my.reshape` function.

(this page is for your `my.reshape` function)