Loading Data into R and Data Wrangling

Introduction to Lab

The objectives of this lab is to learn the basics about using R. You will learn the following in this lab:

1. How to load datasets into R Notebooks as a dataframe using the read.csv() command
2. How to view loaded datasets and use the `subset()` command to select a subset of a data frame
3. How to use the `summary()` command and basic functions to summarize data
4. How to use the aggregate command to get basic statistics from data frames

This is possibly the most important lab all semester.

Seventy-five percent of the questions I get about R involve putting data into R and getting the data to load correctly. This lab covers this valuable skill. It will seem repetitive but this lab is very important, since every other lab will require you to load data from outside sources.

As mentioned in the readings and in the last lab, it's hard to put data into R manually, like what you did in the last lab assignment with the grades.

Normally, we will input data from other sources. I use a spreadsheet program to enter data and then import the completed data into R. As we covered in class, there is a very important way to enter data into R. A spreadsheet should be set up so that variables in columns and each observation as a row. The first row will contain the column names and each of the other rows will contain the data.

Getting Set Up

The first thing you should do is create a folder on your computer for this lab. After you do this, create an R notebook which contains this lab. Go ahead and delete the text that is there, change the title to something more appropriate, like "Lab 2", and save this as Lab2 in that folder.

Now, download the data for this lab. This data should be available on Canvas as lab2score.csv. Download the data onto your computer. When you download the file, make sure you select "Save File As" and save the file in the same folder that contains your R Notebook. Do not open this file.

The most important part is to make sure your R notebook and your data files are in the same folder. If this does not happen, you will get an error that says:

```
Error in file(file, "rt") : cannot open the connection
```

When you see this error, that means you do not have the data in the same file folder, or you mistyped the name of the data frame.

Now we will create a code chunk which loads the data into R. The code chunk will use the `read.csv` command and it will save the data as a data frame named `lab2`. Type the following and run the code chunk:

```r
lab2 = read.csv('lab2score.csv')
```

Note that you have to type the file name exactly correct for R to load the data. If you misspell it, you will get the error that happens above. One common misspelling is to type cvs instead of csv.

If you loaded the data correctly, you should see a lab2 dataframe in your environment. Click on the icon on the far right to view that dataframe.

Whenever I give you data to enter into R, I will also give you a key that tells you about each of the variables. When you make your own data (for a research methods project) it is a good idea to make your own key for when you share it with other people. Here is the key below.

Variables: * Person: person's name * Education: highest level of education completed: none, HS * for high school, college for college, and graduate for graduate work * Sex: F for female, M for male * Test: type of test: Math or English * Score: score on a 50 question test

Whenever we create a new object, we may want to check what kind of object it is. We can use the `class()` command to do so. Type the following:

```r
class(lab2)
```

```
## [1] "data.frame"
```

This tells us that lab2 is a data frame. So we know we input the data correctly.

If we look at lab 2 in the environment, we can also see the number of variables and observations.

1. How many observations are in x? How many variables? Please include this information in your annotation in your R Notebook.

This step is important because I will usually tell you how many rows and columns should be in the data frame. If you have typos in your spreadsheet, this can cause errors. That is why it is important to make sure all your data is present and view your data.

You can also find out how many rows and columns are in x by using the `nrow(x)` and `ncol(x)` command. Another command to find out the names of the columns is the `colnames(x)` command.

2. Type a code chunk to list the column names. Are they the same as you expect based on the guide above?

An interesting thing about colnames is that you can use it to change the column names as well. Let's say we want to change the name of the Person column to subject. Go ahead and type the following.

```r
colnames(lab2)[1] = "Subject"
```

This command changes the column name of the column specified with the brackets. Now look at your data by clicking on the dataframe `lab2` in the environment. Did the column name change?

4. Now let's change the name of column back from "Subject" to "Name". Write a code chunk to do this. Make sure you note as an annotation what question you are answering with your code chunk.

Selecting specific observations

Above, we described how to select variables. Sometimes in R, we would like to select specific observations or rows Sometimes, we want to do statistics only on a smaller subset of the data. In R, we have a command that can do that, called the subset command.

This is an example of what you would type, if your dataframe x had a variable named variable and you wanted to select all the values of x$variablethatequaledvalue.‘y = subset(x, x$variable == value)‘

Try the following. In your annotation, write what part of the data frame is being selected here.

5a.

```r
subset(lab2, lab2$Education == "College")
```

5b.

```r
subset(lab2, lab2$Score > 20)
```

5c.

```r
subset(lab2, lab2$Subject != "Alex")
```

The last example is very important to remember because occasionally we might want to remove an outlier. This is how we would do so.

For R, you can compare two values with a statement called a conditional. The conditionals in R are `==` (two equals signs, which tests whether two variables are equal or if each element of a variable is equal to a certain value), `>` and `<` (doing greater than or lesser than), and `!=` which tests not equal. When doing a conditional, R returns `TRUE` or `FALSE`.

Try typing the following. What happens?

```r
lab2$Sex == "F"
```

Now use code chunks to answer the next two questions. Please include in your annotation which question you are answering.

6. Using the subset command, what command would you type to select only the high school education group?

7.What command would you select to select the English test?

8. Based on what you learned above, how would you remove Bob's scores?

Summarizing data

One easy way to look at the summary of a dataframe is to use the `summary()` command. This command gives the summary of an object, no matter what kind of object it is.

Go ahead and type the following:

```r
summary(lab2)
```

This gives us a summary of each of the variables including the minimum, maximum, mean, and median. This can be helpful to view what is happening in a data frame and reporting basic statistics. Another way of doing this is by using the dataframe$variable notation and using a basic command like `mean()` or `sd()`

For example, to get the mean of Score, I would type:

```r
mean(lab2$Score)
```

```
## [1] 26.94444
```

9. How would I get the standard deviation of score?

Using aggregate to get statistics separated by a grouping variable

Another really important way to analyze data is to compare groups. In the `lab2` dataframe, note that only one variable is actually containing numbers. All the other variables tell us about groups. For instance, the Person variable tells us what person's score it is. The Sex variable tells us the sex of the person.

We may want to look at summary statistics broken up by group. For instance, we may want to compare Math versus English scores. The aggregate command allows us to do so. The aggregate command is complicated and has several different parts. Here is the command that would give us the mean for Score, broken up by Test.

```r
aggregate(Score~Test, lab2, FUN = mean)
```

```
##      Test    Score
## 1 English 28.00000
## 2    Math 25.88889
```

The aggregate command has three parts. The first part is a **formula** which is a way of breaking up the data. The second part is the dataframe we are looking at. The third part is the function we are applying.

The formula has two parts, divided by a tilde (~) which is the key to the left of the 1 key on your keyboard. The first part of the formula is the dependent variable. This is the numeric variable that we want to analyze. The second part is the grouping variable. This is the variable which we use to determine the groups. In this case, we use Score as the dependent variable and Test as the grouping variable.

If we wanted to compare females to males, we would change the grouping variable from Test to Sex. Type the following:

```r
aggregate(Score~Sex, lab2, FUN = mean)
```

What results do you get?

If we want to change the type of function, we can change the `FUN = mean` part to some other command. If we really want to raise the fun, we might look at a median instead of a mean.

```r
aggregate(Score~Sex, lab2, FUN = median)
```

Using this logic, create code chunks to do the following:

10. Get the mean score for each person. This will involve using Person as your grouping variable.
11. Get the standard deviation for each test (English versus Math).
12. Get the mean and median for Score broken up by Education level. Is this what you would predict?.

Summary

Now go ahead and knit your document. You may have errors when you try to knit it. Find the line number and try to fix your errors. If you have an error with the read.csv() command, it may be because your data frame file name has a typo or the data and R notebook are saved in different file folders. Once you knit this assignment, go ahead and upload it and you're done! Great job. You're well on your way to being a world-class data analyst. But before you get too excited, you may want to make sure you know how to do the following, which we discussed in this lab:

- Load a dataframe into R using the `read.csv()` command
- Select a subset of the data frame using the `subset()` command
- Use the `summary()` command to get an overview of a data frame
- Use the `aggregate()` command to get the mean broken down by a variable