SOCI 385, Week One, Class Two

Matt Lawrence

9/11/2019

Getting Started

To begin, copy all the text on this screen. Then open RStudio, and in the "File" menu select "New File" then "R Markdown". Add a title like "SOCI 385, Week One, Class Two", include your name in the "Author" box, select "PDF" as the Default Output Format, and click Ok. A default markdown file will open. Delete all the text in the default file below the header (everything after line 7) and paste in the text you copied to your clipboard. Make sure that "R Markdown" is selected in the file type drop down menu in the bottom right corner of this pane. Save this file to your desktop.

Welcome to R Studio!

Some navigation before we begin:

- The upper left pane is the *text editor* where you keep all of your commands.
- The lower left pane is the *console* where the commands will be executed.
- The upper right pane's `Environment` window will show the data that you load.
- The lower right pane will show previews of your R Markdown documents, plots, help documents, packages, and other features.

This file is an R Notebook which is an R Markdown file that allows you to integrate text, R code, and R output into a single document. This is the recommended file type for our course, but you may want to use other file types in the future. All the possible file types are found when you click File>New File. For our purposes - and for producing transparent and reproducible research - the advantage of an R Notebook is that you can easily include text, notes, and comments that are not code (like this paragraph).

How does R know what is code and what is regular text? In an R Notebook, code goes in a fenced code block. To open the fence, type three backticks and an r in curly brackets on a single line; to close the fence, type three backticks on a single line like this:

Everything between those fences will be run as code (unless it is preceded by a hashtag).

Let's start by using R as a basic calculator. Since even the following calculation is considered code we need to place it within a code block. After the r, in the line where you open your fence, you can provide a short description of what the code in that chunk does. Those short descriptions are collected as bookmarks in the drop down menu on the bottom of this pane and can help you find pieces of your file.

```
300 + 85
```

```
## [1] 385
```

To run this line of code, click the green arrow on the far right side of the first line of the code block.

For subtraction, use a -; for multiplication, use a *; for division, use a /.

What is 770 divided by 2?

REPLACE THIS LINE WITH YOUR CODE

```
770 / 2
```

```
## [1] 385
```

Our course number could be helpful to save so we do not have to calculate it each time we want to reference it. We save a value by creating an object. The syntax for doing so is to name our object first, then use the symbol "<-" to say what value our object takes. Let's call our new object `coursenumber`:

```
coursenumber <- 770 / 2
```

Run this line of code. In the top right quadrant, we should see a new value saved. That's promising, but there's no output below the code chunk! That's because we have to separately recall the object for it to be displayed. The only line of code we need to do so is our object name:

```
coursenumber
```

```
## [1] 385
```

Note that case matters in R code. See how by replacing the `c` in the previous chunk with a capital `C`.

We can now use the object we created in other calculations by referring to it by name:

```
coursenumber - 145
```

```
## [1] 240
```

Objects can take numerical or character values. The difference is that if we are using text we have to enclose the value in quotation marks:

```
coursename <- "Social Statistics"
coursename
```

```
## [1] "Social Statistics"
```

We can now reference these saved objects in our code as well as our text. To use *inline code*, wrap the code in single backticks with a preceding r:

> This class is called Social Statistics and it is SOCI 385.

The printed document will replace `coursename` and `coursenumber` with their saved values.

Introducing R Markdown

We will be using R Notebooks with R Markdown to write and run our code. R Markdown is a plain text-based form of writing with lots of neat tricks. For example, look at the heading for this paragraph. The two hashtags are how we start a new section in R Markdown. If we used one hashtag, the section heading would be bigger in the output. If we used three (or four) hashtags, the section headings would be smaller (or even smaller) in the output, which would be helpful if you have lots of sub- (or sub-sub-) headings

There's an R Markdown cheat sheet in the pages folder on Canvas that has much more info on how to use R Markdown. Here are a few first day features to get you started:

Wrapping a word in `single backticks` will highlight it when you print.

Wrapping text in one asterisk will italicize it when you print.

Wrapping text in two asterisks will bold it when you print.

Wrapping text in three asterisks will bold and italicize it when you print.

And three (or more) dashes on a single line will create a horizontal rule:

Getting Started With R

Today we will learn one way to import data into R and how to get simple summaries of different types of variables.

We will use the `hurricanes` dataset to explore types of variables. This dataset is provided online by Jung et al as a supplement to their 2014 paper. To find this archival dataset and the datasets for their experiments, click here.

First, we need to install a package. Packages are additional functions that supplement base R. We will use the `tidyverse` package extensively this semester.

Here's the code to install the package. You only have to install the package once on any computer you use. Once your computer has the package installed, you can load the package with the `library()` function.

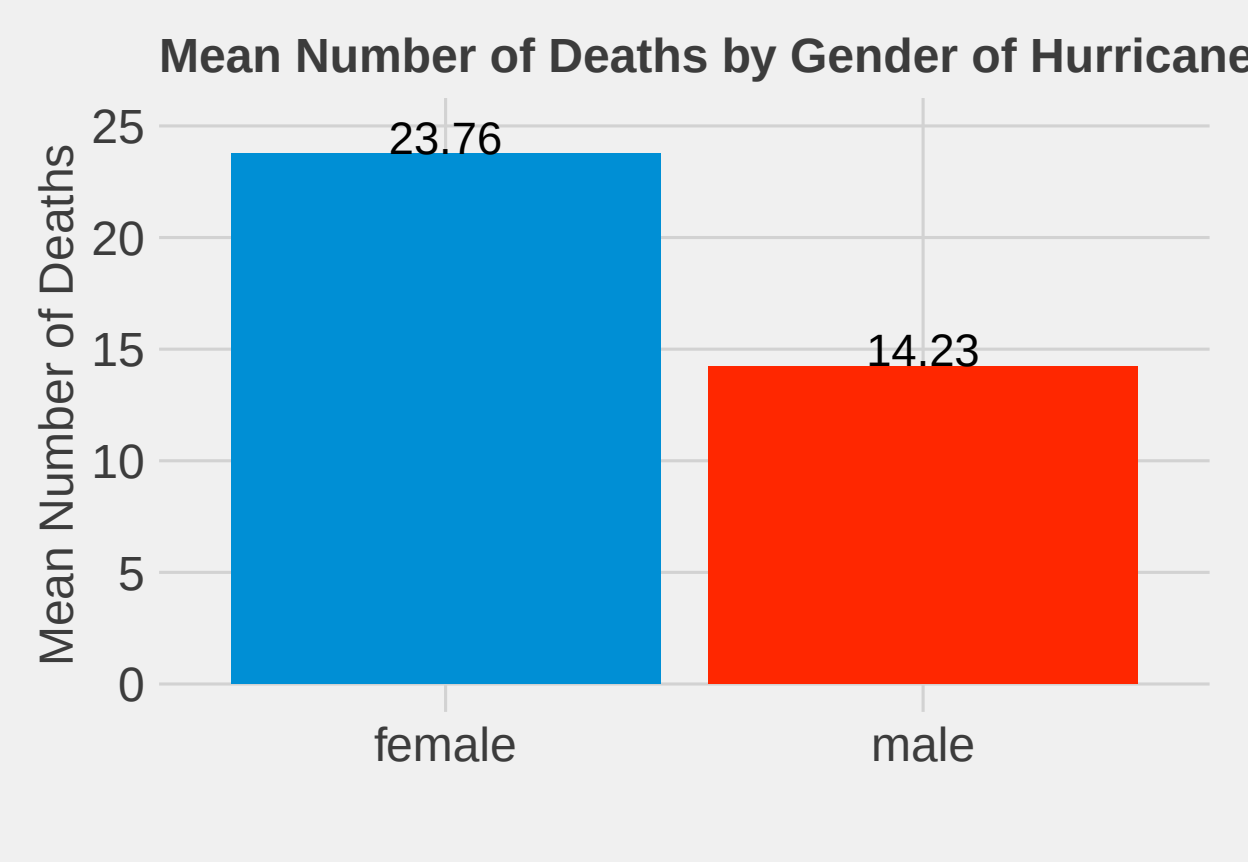
Now that the package is installed, you can put a hashtag in front of the `install.packages()` line so it does not run again.

Next we'll get the data.

```
hurricanes <- read.csv("https://raw.githubusercontent.com/mjclawrence/soci385/master/data/hurricanes.csv
```

Plot

```
library(ggthemes)
plot_data <- hurricanes %>% group_by(male_female) %>%
     summarise(deaths_mean = round(mean(deaths),2))
plot <- ggplot(plot_data, aes(x = male_female, y = deaths_mean, fill = male_female))
plot + geom_col() + geom_text(aes(label = deaths_mean), size = 6, vjust = 0) +
     labs(x = "", y = "Mean Number of Deaths",
          title = "Mean Number of Deaths by Gender of Hurricane Name") +
  theme_fivethirtyeight() + scale_fill_fivethirtyeight(name = "") + ylim(c(0,25)) +
       theme(axis.title = element_text(size = 18),
           axis.text = element_text(size = 18),
           title = element_text(size = 12)) + guides(fill = FALSE)
```



If the code above executed correctly, you should now see the data frame loaded and saved with the object name you created in the `environment` panel in the top right quadrant of R Studio. There should be 92 observations of 7 variables in the data frame named hurricanes.

Click the white arrow to the left of the object name to see all the variables (year, name, etc.). Click the spreadsheet icon on the far right side of the data name row to open the full data frame. Note that the *variables* are in the columns and the *values* are in the rows, with each row representating a separate *observation* or *case*.

Types of Variables

We have different types of variables in our data frame. Some have the `int` tag identifying them as interval variables, some have the `Factor` tag identifying them as categorical variables, and some have the `numeric` tag.

Let's look at the `name` variable to start. We can see in the `Environment` panel that this variable is a factor variable with 83 levels - or 83 different possible values. In this case, each value is a different name, making this an example of a *nominal variable*.

To see how many observations have each value - or how many hurricanes have each name - we will use the `table()` command. In R, we have to reference all variables by their data frame. So if we want a table of the `name` variable, we first have to tell R that the variable is in the `hurricanes` data frame. We separate the data frame and the variable name by a dollar sign:

```
table(hurricanes$name)
```

```
##
##      able     agnes      alex    alicia     allen      alma    andrew
```

```
##         1         1         1         1         1         1         1
##      babe   barbara     belle    bertha     betsy    beulah       bob
##         1         1         1         1         1         1         3
##    bonnie      bret   camille     carla    carmen     carol     celia
##         2         1         1         1         1         1         1
##   chantal   charley     cindy claudette      cleo    connie     danny
##         1         2         2         1         1         1         2
##     david     debra    dennis     diana     diane     dolly     donna
##         1         1         1         1         1         1         1
##      dora      earl      easy     edith      edna     elena    eloise
##         1         1         1         1         1         1         1
##     emily      erin     ethel      fern  florence    flossy     floyd
##         1         1         1         1         2         1         2
##      fran   frances  frederic    gaston   georges    ginger    gladys
##         1         1         1         1         1         1         1
##    gloria    gracie    gustav     hazel    helene     hilda      hugo
##         1         1         1         1         1         1         1
##  humberto       ike      inez      ione     irene     isaac    isabel
##         1         1         1         1         2         1         1
##    isbell      ivan    jeanne     jerry      juan      kate      king
##         1         1         1         1         1         1         1
##      lili      opal   ophelia      rita     sandy     wilma
##         1         1         1         1         1         1
```

When you run that line of code you should see a table with every possible value of the `name` variable and the number of observations with that value. There is one hurricane named Able, for example. How many hurricanes are named Florence? Which hurricane name is most common?

By default, R will sort factor variables alphabetically. An easier way to see which factor has the most observations - in this case, which name has the most hurricanes - is to use the `sort()` function to order the names by the number of observations in each. Note that functions "wrap" code in parentheses:

```
sort(table(hurricanes$name))
```

```
##
##      able     agnes      alex    alicia     allen      alma    andrew
##         1         1         1         1         1         1         1
##      babe   barbara     belle    bertha     betsy    beulah      bret
##         1         1         1         1         1         1         1
##   camille     carla    carmen     carol     celia   chantal claudette
##         1         1         1         1         1         1         1
##      cleo    connie     david     debra    dennis     diana     diane
##         1         1         1         1         1         1         1
##     dolly     donna      dora      earl      easy     edith      edna
##         1         1         1         1         1         1         1
##     elena    eloise     emily      erin     ethel      fern    flossy
##         1         1         1         1         1         1         1
##      fran   frances  frederic    gaston   georges    ginger    gladys
##         1         1         1         1         1         1         1
##    gloria    gracie    gustav     hazel    helene     hilda      hugo
##         1         1         1         1         1         1         1
##  humberto       ike      inez      ione     isaac    isabel    isbell
##         1         1         1         1         1         1         1
##      ivan    jeanne     jerry      juan      kate      king      lili
##         1         1         1         1         1         1         1
##      opal   ophelia      rita     sandy     wilma    bonnie   charley
```

```
##         1         1         1         1         1         2         2
##     cindy     danny  florence     floyd     irene       bob
##         2         2         2         2         2         3
```

```
sort(table(hurricanes$name), decreasing=TRUE)
```

```
##
##       bob    bonnie   charley     cindy     danny  florence     floyd
##         3         2         2         2         2         2         2
##     irene      able     agnes      alex    alicia     allen      alma
##         2         1         1         1         1         1         1
##    andrew      babe   barbara     belle    bertha     betsy    beulah
##         1         1         1         1         1         1         1
##      bret   camille     carla    carmen     carol     celia   chantal
##         1         1         1         1         1         1         1
## claudette      cleo    connie     david     debra    dennis     diana
##         1         1         1         1         1         1         1
##     diane     dolly     donna      dora      earl      easy     edith
##         1         1         1         1         1         1         1
##      edna     elena    eloise     emily      erin     ethel      fern
##         1         1         1         1         1         1         1
##    flossy      fran   frances  frederic    gaston   georges    ginger
##         1         1         1         1         1         1         1
##    gladys    gloria    gracie    gustav     hazel    helene     hilda
##         1         1         1         1         1         1         1
##      hugo  humberto       ike      inez      ione     isaac    isabel
##         1         1         1         1         1         1         1
##    isbell      ivan    jeanne     jerry      juan      kate      king
##         1         1         1         1         1         1         1
##      lili      opal   ophelia      rita     sandy     wilma
##         1         1         1         1         1         1
```

Now let's look at another categorical factor variable - `male_female` - which identifies whether a hurricane has a male or female name. Try making a table of this variable to see how many hurricanes have male and female names.

REPLACE THIS LINE WITH YOUR CODE

```
table(hurricanes$male_female)
```

```
##
## female   male
##     62     30
```

R calls the `male_female` variable a factor variable because its values are text categories rather than numbers. The `gender_mf` variable is a numerical variable that has the same exact information as the `male_female` variable but uses a 0 to indicate a hurricane with a male name and a 1 to indicate a hurricane with a female name. A numerical variable that takes the values 0 and 1 is called a *binary variable*.

We can use the `table()` function with binary and interval variables as well. But it can also be helpful to use the `summary()` function when we have numerical data. For example, execute the following code chunk:

```
summary(hurricanes$gender_mf)
```

```
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
##  0.0000  0.0000  1.0000  0.6739  1.0000  1.0000
```

The minimum and maximum values of a binary variable will always be 0 and 1. The mean of a binary variable is the proportion with a 1. *[Remember this... it will be a trick that will save you lots of time in future weeks!].* So the mean of `gender_mf` tells us that 67.39% of the hurricanes in this sample have female names (or have a value of 1 for the `gender_mf` variable).

How could you use R as a calculator to see that 67.39% of the hurricanes in this sample have female names?

REPLACE THIS LINE WITH YOUR CODE

```
62/92
```

```
## [1] 0.673913
```

The summary() command is more helpful when numerical variables have many possible values because we get a snapshot of the distribution of values. Let's look at the `deaths` variable, which is the number of deaths per hurricane. This is an example of a discrete variable. It is not a continuous variable because the values are whole numbers; for example, there are either 2 or 3 deaths, not 2.4 or 2.8.

Try summarizing the `deaths` variable to find the mean (or average) number of hurricane-related deaths in this sample:

REPLACE THIS LINE WITH YOUR CODE

```
summary(hurricanes$deaths)
```

```
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
##    0.00    2.00    5.00   20.65   20.25  256.00
```

How could we find the hurricane with 256 deaths? Sometimes we only want to use a function for some observations rather than all observations. `R` calls this *indexing*. We use brackets to indicate the variable names and values we want to index.

In this next chunk, we want to pull the value for the `name` variable for the observation where the value for the `deaths` variable equals 256. Note that in R we will use two equal signs when referecing a specific value that exists in a data frame.

```
hurricanes$name[hurricanes$deaths==256]
```

```
## [1] camille
## 83 Levels: able agnes alex alicia allen alma andrew babe barbara ... wilma
```

Other indexing tools that are helpful:

- For greater than, use >
- For less than, use <
- For greater than or equal to, use >=
- For less than or equal to, use <=
- For not equal to, use !=

Finally, let's combine indexing with `summary()` to find how mean number of deaths varies by the gender of a hurricane's name.

Here's the code for hurricanes with male names:

```
summary(hurricanes$deaths[hurricanes$male_female=="male"])
```

```
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
##    0.00    1.25    5.00   14.23   15.00   84.00
```

Repeat the summary of the `deaths` variable for hurricanes with female names:

REPLACE THIS LINE WITH YOUR CODE

```
summary(hurricanes$deaths[hurricanes$male_female=="female"])
```

```
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
##    0.00    2.00    5.00   23.76   21.00  256.00
```

```
#library(ggthemes)
plot_data <- hurricanes %>% group_by(male_female) %>%
     summarise(deaths_mean = round(mean(deaths),2))
plot <- ggplot(plot_data, aes(x = male_female, y = deaths_mean, fill = male_female))
plot + geom_col() + geom_text(aes(label = deaths_mean), size = 6, vjust = 0) +
     labs(x = "", y = "Mean Number of Deaths",
          title = "Mean Number of Deaths by Gender of Hurricane Name") +
  theme_fivethirtyeight() + scale_fill_fivethirtyeight(name = "") + ylim(c(0,25)) +
       theme(axis.title = element_text(size = 18),
           axis.text = element_text(size = 18),
           title = element_text(size = 12)) + guides(fill = FALSE)
```