Module 1, Assignment 2: Getting to Know the Team

Ellen Bledsoe

2023-02-08

Assignment Description

Purpose

The goal of this assignment is to get comfortable using R to look at 1- and 2-dimensional data sets.

Task

Write R code to successfully answer each question below.

Criteria for Success

- Code is within the provided code chunks
- Code chunks run without errors
- Code produces the correct result
  - Code that produces the correct answer will receive full credit
  - Code attempts with logical direction will receive partial credit
- Written answers address the questions in sufficient detail

Due Date

February 7 at midnight MST

Assignment Questions

Remember to run each chunk of code as you go!

Each question is worth 2 points.

Vectors

Let's start working with vectors, or 1-dimensional data, first.

IMPORTANT!
Run this chunk of code to create a vector of data to use.

```
# vector with counts of penguins
counts <- c(2, 9, 4, 3, 6, 7, 1, 0, 3)
```

1. What data class is `counts`? Write a line of code that tells you.
2. Write a line of code that pulls out the 2nd value in the `counts` vector.
3. Calculate the average number of penguins that were counted.

Data Frames

Now that we've practiced with vectors, let's move on to 2-dimensional data.

Remember that quiz we took in class with fun/silly questions about our trip to Antarctica? It's time to play around with that data!

IMPORTANT!
The following code chunk will read in the data. Be sure to run it before try to answer the questions! A bunch of stuff will pop up, but it should work just fine.

```
library(tidyverse)
```

```
## -- Attaching packages --------------------------------------- tidyverse 1.3.2 --
## v ggplot2 3.4.0     v purrr   0.3.4
## v tibble  3.1.8     v dplyr   1.0.9
## v tidyr   1.2.0     v stringr 1.5.0
## v readr   2.1.2     v forcats 0.5.1
## -- Conflicts ------------------------------------------ tidyverse_conflicts() --
## x dplyr::filter() masks stats::filter()
## x dplyr::lag()    masks stats::lag()
```

```
team_data <- read_csv("../data/team_antarctica_responses.csv") %>% drop_na()
```

```
## Rows: 32 Columns: 11
## -- Column specification --------------------------------------------------------
## Delimiter: ","
## chr (4): swim, good_with_animals, parka_color, flag_mascot
## dbl (7): uniqueID, fishing_skill, cold_tolerance, remote_location, distance_...
##
## i Use ‘spec()‘ to retrieve the full column specification for this data.
## i Specify the column types or set ‘show_col_types = FALSE‘ to quiet this message.
```

Running the code chunk above produces a message that gives us some useful information, even before we look at the data set (alternatively, you can check it out in the environment tab to the right).

- What are the dimensions of the data?
- What are the names of the columns in the data?
- What data *class* do you expect to find in each column (i.e., numbers, character strings, T/F, etc.)

4. Take a look at the data frame using the `head()` function. Typically, `head()` provides the first 6 rows of data. Modify one of the arguments in `head()` so that the line of code prints the first 10 rows. (If you aren't sure how to do that, remember how you can look for help about functions!)

5. Using what you know about sub-setting data frames, write a line of code the pulls out the parka color for uniqueID 9 (row 9). (Hint: count the columns!)

When we have a large data set like this, it is often helpful to summarize the data in some way. The next few questions will help use get a better understanding of the content of the data set.

6. On average, how did people rate their cold tolerance?
7. What is the minimum and maximum distance that would be traveled by a team member to get to Antarctica? Use `min()` and `max()`.
8. Create a data frame that only includes rows of data for people who rated their cooking skills as a 5. (Hint: numbers do not need quotation marks around them).

```
team_data[team_data$cooking_skill == 5,]
```

```
## # A tibble: 2 x 11
##   uniqueID fishi~1 swim  cold_~2 good_~3 remot~4 parka~5 flag_~6 dista~7 bedsi~8
##      <dbl>   <dbl> <chr>   <dbl> <chr>     <dbl> <chr>   <chr>     <dbl>   <dbl>
## 1       13       2 Yes         4 Yes           3 black   Sea Sp~    8276       3
## 2       23       1 Yes         3 Yes           2 blue    Leopar~    8340       3
## # ... with 1 more variable: cooking_skill <dbl>, and abbreviated variable names
## #   1: fishing_skill, 2: cold_tolerance, 3: good_with_animals,
## #   4: remote_location, 5: parka_color, 6: flag_mascot, 7: distance_mi,
## #   8: bedside_manner
```

9. How many different parka colors will we have? This will require two steps (2 points per step).

   a. First, use the `unique()` function to pull out each distinct value from the `parka_color` column. Save this as an object called `colors`, which is now a vector.
   b. Use the `length()` function to count how many distinct specialties are in the `colors` vector.

```
# 9a

# 9b
```

Bonus (up to 2 points)!

What animal should be on our team flag?

First, create a vector called `mascot` that has only the values from the `flag_mascot` column. Next, use the `table()` function on the `mascot`. This will give you the number of times each option was chosen. According to the results, which animal should be on our team flag?

Answer:

Turning in Your Assignment

Follow these steps to successfully turn in your assignment on D2L.

1. Click the `Knit` button up near the top of this document. This should produce a PDF file that shows up in the `Files` panel on the bottom-right of your screen.

2. Click the empty box to the left of the PDF file.
3. Click on the blue gear near the top of the `Files` panel and choose Export.
4. Put your last name at the front of the file name when prompted, then click the Download button. The PDF file of your assignment is now in your “Downloads” folder on your device.
5. Head over to D2L and navigate to Module 1 Assignment 2. Submit the PDF file that you just downloaded.