Lab Exercises Week 02

Alison Lawyer

2024-09-23

```r
knitr::opts_chunk$set(echo = TRUE)
knitr::opts_chunk$set(tidy.opts = list(width.cutoff = 60), tidy = TRUE)
```

LAB EXERCISES

This week you will continue practicing basic R syntax, functions, and descriptive summary statistics. You will also make a pretty table that you could use to report your results in a document or presentation. For all exercises, we will use the palmer penguins data that you started using for homework 1.

Part 1

First, let's look at NA values a bit more closely

```r
library(tidyverse)
```

```
## -- Attaching core tidyverse packages ------------------------ tidyverse 2.0.0 --
## v dplyr     1.1.4     v readr     2.1.5
## v forcats   1.0.0     v stringr   1.5.1
## v ggplot2   3.5.1     v tibble    3.2.1
## v lubridate 1.9.3     v tidyr     1.3.1
## v purrr     1.0.2
## -- Conflicts ------------------------------------------ tidyverse_conflicts() --
## x dplyr::filter() masks stats::filter()
## x dplyr::lag()    masks stats::lag()
## i Use the conflicted package (<http://conflicted.r-lib.org/>) to force all conflicts to become error
```

```r
library(dplyr)

# Load the palmer penguin dataset
library(palmerpenguins)
penguins <- data.frame(penguins)

# You can use the function is.na() to test whether any
# value in a vector of values is NA Check for NAs in the
# column 'body_mass_g'
body_mass_na <- is.na(penguins$body_mass_g)
print(body_mass_na)
```

```
##   [1] FALSE FALSE FALSE  TRUE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE
##  [13] FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE
##  [25] FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE
##  [37] FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE
##  [49] FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE
##  [61] FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE
##  [73] FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE
##  [85] FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE
##  [97] FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE
## [109] FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE
## [121] FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE
## [133] FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE
## [145] FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE
## [157] FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE
## [169] FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE
## [181] FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE
## [193] FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE
## [205] FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE
## [217] FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE
## [229] FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE
## [241] FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE
## [253] FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE
## [265] FALSE FALSE FALSE FALSE FALSE FALSE FALSE  TRUE FALSE FALSE FALSE FALSE
## [277] FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE
## [289] FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE
## [301] FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE
## [313] FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE
## [325] FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE
## [337] FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE
```

```r
# It will be difficult to detect the TRUE values in a sea
# of FALSE values Since TRUE/FALSE are internally stored as
# 1/0 values, we can do numerical calculations on the
# result of is.na() Let's try that with sum() function by
# wrapping the is.na() function in a sum() function
# Wrapping a function means using one function inside of
# another function
body_mass_na_sum <- sum(is.na(penguins$body_mass_g))
print(body_mass_na_sum)
```

```
## [1] 2
```

```r
# This is the number of rows with missing values for body
# mass.  We can use this to incorporate the calculation in
# a summarize() function too.  Create a summary by species
# using the summarize() function, where a new variable
# holds the number of missing values for 'body_mass_g'.
grouped_missing_body_mass <- penguins %>%
    group_by(species) %>%
    summarize(sum_missing_body_mass = sum(is.na(body_mass_g)))
head(grouped_missing_body_mass)
```

```
## # A tibble: 3 x 2
##   species   sum_missing_body_mass
##   <fct>                     <int>
## 1 Adelie                        1
## 2 Chinstrap                     0
## 3 Gentoo                        1
```

```
# Now let's add to that also the proportion of missing
# values for each species, that is, the count of missing
# values divided by the total number of rows for each
# species. Add that to your code above and re-create the
# summary with two variables, one for number of missing
# rows and one for percentage of missing rows. Both
# variables will be created in the same summarize()
# function statement. Hint: you can use the function n() to
# count the number of rows.
grouped_proportion <- penguins %>%
    group_by(species) %>%
    summarize(total_missing_rows = sum(is.na(body_mass_g)), proportion_missing = (sum(is.na(body_mass_g
        100)
head(grouped_proportion)
```

```
## # A tibble: 3 x 3
##   species   total_missing_rows proportion_missing
##   <fct>                  <int>              <dbl>
## 1 Adelie                     1              0.658
## 2 Chinstrap                  0              0
## 3 Gentoo                     1              0.806
```

```
# When dealing with multiple variables that each may have
# missing values, one way to filter the dataset to only
# those rows with no missing values in any variable is to
# use the function complete.cases(). Look up the help for
# this function by typing ?complete.cases into your
# console. See if you can figure out how to use this
# function. If it is not obvious from the help page, use
# other resources online or play around until you get it to
# work (this is a very common way to solve coding
# problems!).
penguins_without_missing_rows <- penguins[complete.cases(penguins),
    ]
# penguins_without_missing_rows <- penguins %>%
# filter(complete.cases(penguins))
# penguins_without_missing_rows <- na.exclude(penguins)
# penguins_without_missing_rows <- na.omit(penguins)
head(penguins_without_missing_rows)
```

```
##   species    island bill_length_mm bill_depth_mm flipper_length_mm body_mass_g
## 1  Adelie Torgersen           39.1          18.7               181        3750
## 2  Adelie Torgersen           39.5          17.4               186        3800
## 3  Adelie Torgersen           40.3          18.0               195        3250
## 5  Adelie Torgersen           36.7          19.3               193        3450
## 6  Adelie Torgersen           39.3          20.6               190        3650
## 7  Adelie Torgersen           38.9          17.8               181        3625
##      sex year
## 1   male 2007
## 2 female 2007
## 3 female 2007
## 5 female 2007
## 6   male 2007
## 7 female 2007
```

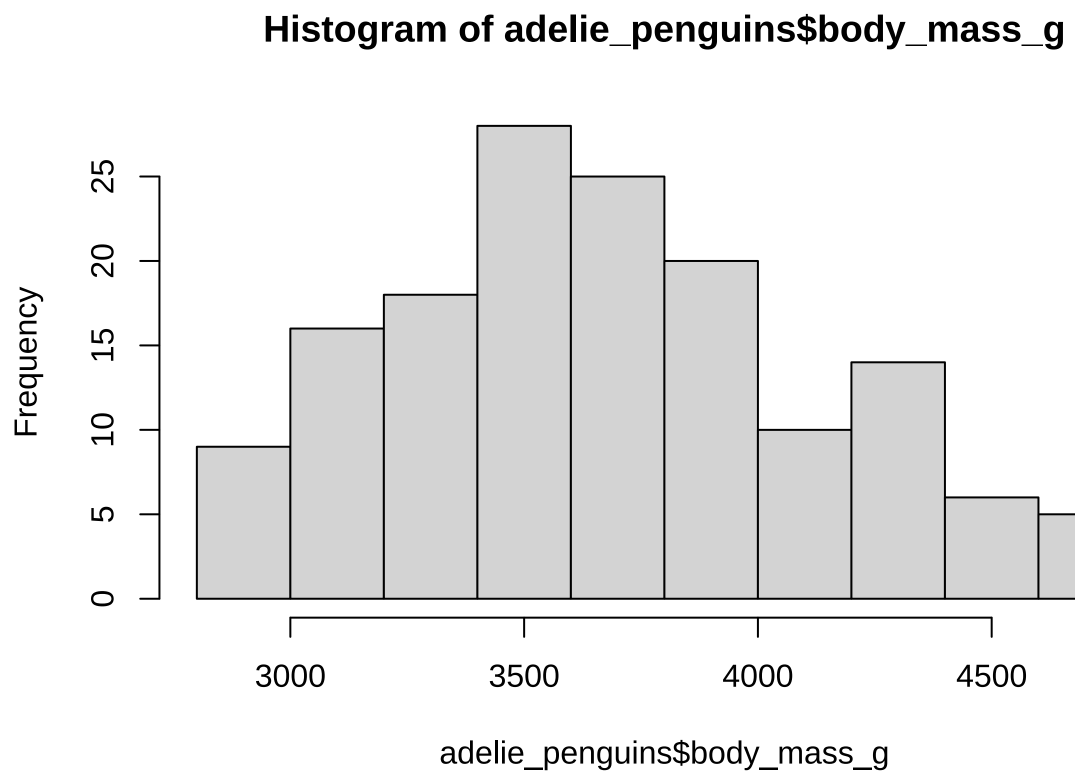
Part 2

Now let's look at the distribution of body mass for each species by making histograms

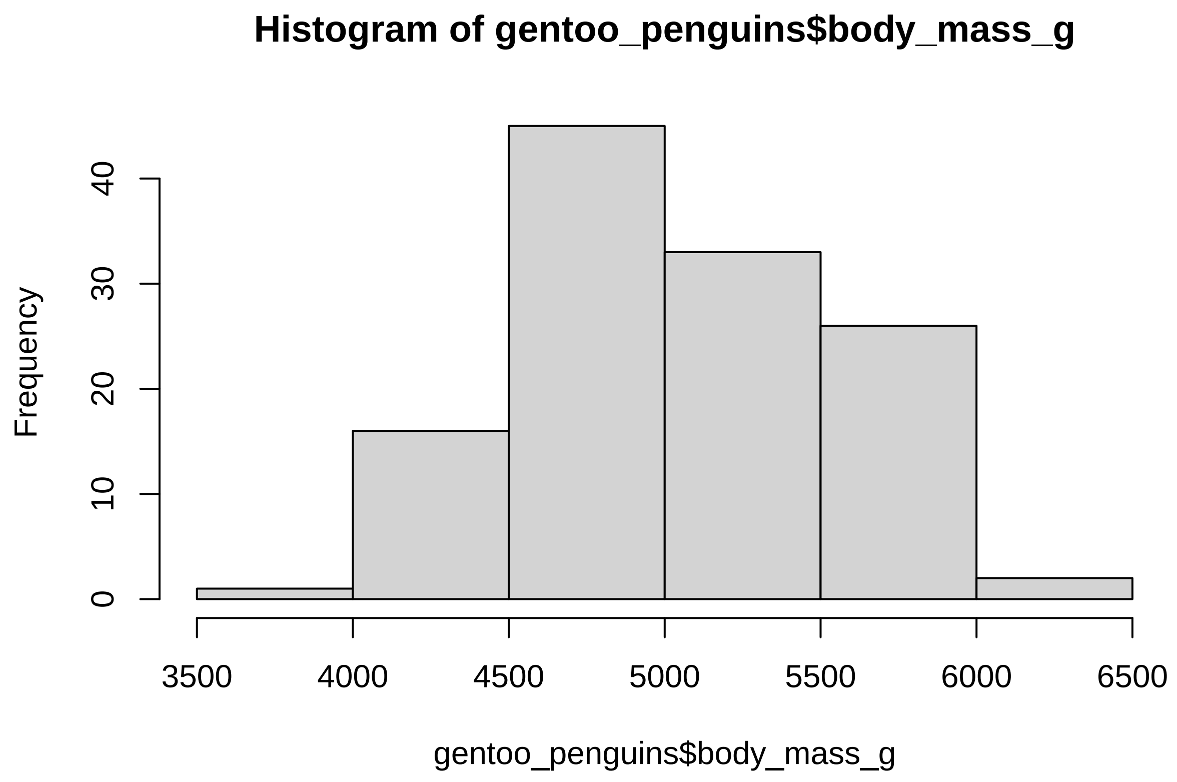
```
# Create a new dataframe holding body mass data for Adelie
# penguins only. Keep the species, sex, and body_mass_g
# columns only
adelie_penguins <- penguins %>%
    filter(species == "Adelie") %>%
    select(c(species, sex, body_mass_g))
gentoo_penguins <- penguins %>%
    filter(species == "Gentoo") %>%
    select(c(species, sex, body_mass_g))
chinstrap_penguins <- penguins %>%
    filter(species == "Chinstrap") %>%
    select(c(species, sex, body_mass_g))
head(gentoo_penguins)
```

```
##   species    sex body_mass_g
## 1  Gentoo female        4500
## 2  Gentoo   male        5700
## 3  Gentoo female        4450
## 4  Gentoo   male        5700
## 5  Gentoo   male        5400
## 6  Gentoo female        4550
```

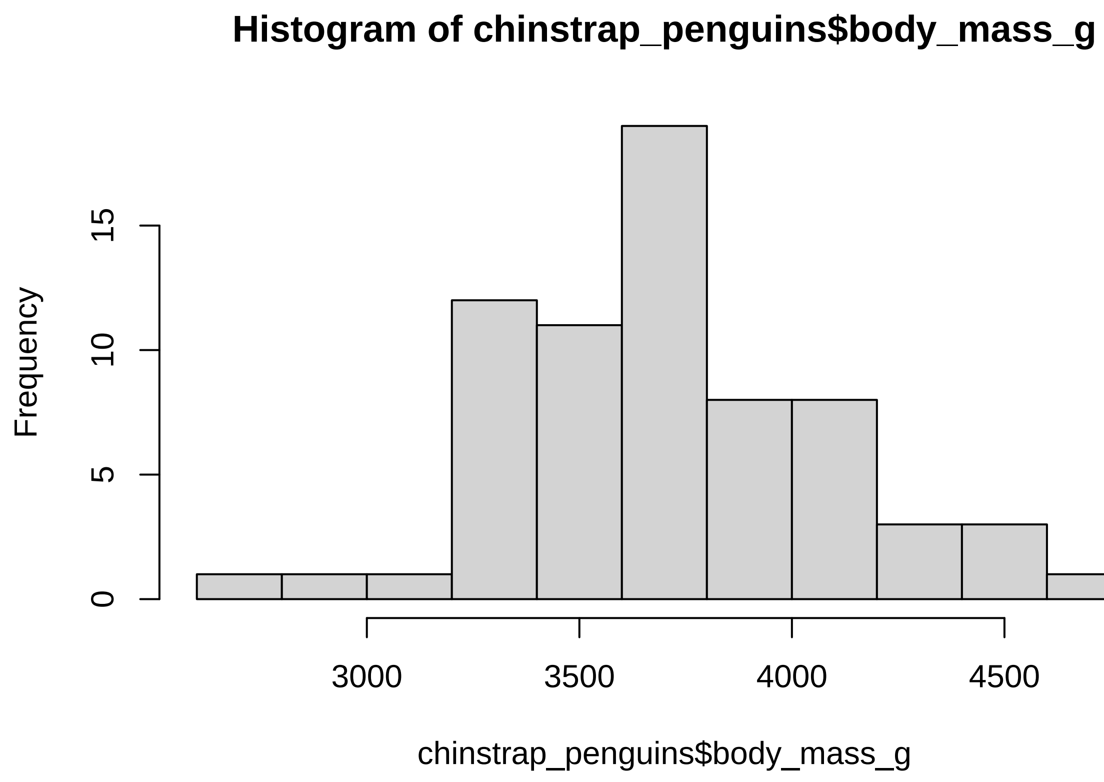
```
# Use the function hist() to make a histogram of weight for
# Adelie penguins
hist(adelie_penguins$body_mass_g)
```



```r
hist(gentoo_penguins$body_mass_g)
```



```
hist(chinstrap_penguins$body_mass_g)
```



```
# Is there skew in the data, and if so are the data left or
# right skewed?  plot mean and median lines to help
# visualize


# Repeat this for the other two species and answer the same
# question
```

Part 3

Now, let's practice adding some descriptive statistics of body mass and save everything in a summary dataframe.

```
# Create a summary dataframe of the variable body_mass_g by
# species that contains the following descriptive stats:
# Mean, Sample Variance, Sample Standard Deviation, Median,
# Interquartile Range (Tip: look up function IQR() to do
# this), Min, Max, Number of observations

penguins %>%
    group_by(species) %>%
    summarize(mean = mean(body_mass_g, na.rm = TRUE), variance = var(body_mass_g,
        na.rm = TRUE), sd = sd(body_mass_g, na.rm = TRUE), median = median(body_mass_g,
        na.rm = TRUE), IQR = IQR(body_mass_g, na.rm = TRUE),
        min = min(body_mass_g, na.rm = TRUE), max = max(body_mass_g,
            na.rm = TRUE), count = n())
```

```
## # A tibble: 3 x 9
##   species    mean variance    sd median   IQR   min   max count
##   <fct>     <dbl>    <dbl> <dbl>  <dbl> <dbl> <int> <int> <int>
## 1 Adelie    3701.  210283.  459.   3700  650   2850  4775   152
## 2 Chinstrap 3733.  147713.  384.   3700  462.  2700  4800    68
## 3 Gentoo    5076.  254133.  504.   5000  800   3950  6300   124
```

Part 4

Finally, let’s make a more automated data summary in form of a tidy table using the table1 package

```
# Make a pretty table of your data summary with the
# table1() function The table should list species as
# columns and summary statistics for body mass as rows
library(table1)
```

```
##
## Attaching package: 'table1'

## The following objects are masked from 'package:base':
##
##     units, units<-
```

```
table1(~body_mass_g | species, data = na.omit(penguins))
```

```
## Get nicer `table1` LaTeX output by simply installing the `kableExtra` package
```

	Adelie	Chinstrap	Gentoo	Overall
	(N=146)	(N=68)	(N=119)	(N=333)
body_mass_g				
Mean (SD)	3710 (459)	3730 (384)	5090 (501)	4210 (805)
Median [Min, Max]	3700 [2850, 4780]	3700 [2700, 4800]	5050 [3950, 6300]	4050 [2700, 6300]