loops-after.Rmd

Jai Jot Kaur

11/24/2022

Exercise 1: The data set

The UHURU experiment in Kenya has conducted a survey of Acacia and other tree species in ungulate exclosure treatments. Each of the individuals surveyed were measured for tree height (HEIGHT), circumference (CIRC) and canopy size in two directions (AXIS_1 and AXIS_2).

Read the UHURU tree data available for download in a tab delimited ("") format using the following code:

```
tree_data <- read.csv("https://ndownloader.figshare.com/files/5629536",
                      sep = '\t',
                      na.strings = c("dead", "missing", "MISSING", "NA", "?", "3.3."))
```

What is the code doing? Explain the meaning of each argument and how the values used for each argument afect the outcome.

The code is reading the file, which is indicated by a link, and the sep = '/t' allows for the UHURU tree data to be read in a tab delimited way. Additionally, the na.strings part of the functions ensures that everything within the parenthesis will be replaced by the value of NA.

Exercise 2: Tree volumes

You want to estimate the crown volumes for the different tree species and have developed equations for species in the Acacia genus:

`{r.eval = FALSE} volume = 0.16 * HEIGHT^0.8 * pi * AXIS_1 * AXIS_2`

and the Balanites genus:

`{r.eval = FALSE} volume = 1.2 * HEIGHT^0.26 * pi * AXIS_1 * AXIS_2`

For all other genera you'll use a general equation developed for any tree:

`{r.eval = FALSE} volume = 0.5 * HEIGHT^0.6 * pi * AXIS_1 * AXIS_2`

Write a function called tree_volume_calc() that calculates the canopy volume for the Acacia species in the dataset. To do so, use an if statement in combination with the str_detect() function from the stringr R package. The code str_detect(SPECIES, "Acacia") will return TRUE if the string stored in this variable contains the word "Acacia" and FALSE if it does not. This function will have to take the following arguments as input: SPECIES, HEIGHT, AXIS_1, AXIS_2. Then run the following line:

```
tree_volume_calc <- function(SPECIES, HEIGHT, AXIS_1, AXIS_2) {
  if (SPECIES == "Acacia") {
    str_detect(SPECIES, "Acacia")
  }
```

```
  volume = 0.16 * HEIGHT^0.8 * pi * AXIS_1 * AXIS_2
  return(TRUE)
}

tree_volume_calc("Acacia_brevispica", 2.2, 3.5, 1.12)
```

```
## [1] TRUE
```

```
# TRUE
```

Write a function called tree_volume_calc() that calculates the canopy volume for the Acacia species in the dataset. To do so, use an if statement in combination with the str_detect() function from the stringr R package. The code str_detect(SPECIES, "Acacia") will return TRUE if the string stored in this variable contains the word "Acacia" and FALSE if it does not. This function will have to take the following arguments as input: SPECIES, HEIGHT, AXIS_1, AXIS_2. Then run the following line:

Expand this function to additionally calculate canopy volumes for other types of trees in this dataset by adding if/else statements and including the volume equations for the Balanites genus and other genera. Then run the following lines:

```
tree_volume_calc <- function(SPECIES, HEIGHT, AXIS_1, AXIS_2) {
  if (SPECIES == "Acacia") {
    str_detect(SPECIES, "Acacia")
  volume = 0.16 * HEIGHT^0.8 * pi * AXIS_1 * AXIS_2
  } else if (SPECIES == "Balanites") {
  volume = 1.2 * HEIGHT^0.26 * pi * AXIS_1 * AXIS_2
} else {
  volume = 0.5 * HEIGHT^0.6 * pi * AXIS_1 * AXIS_2}
return(TRUE)
}

tree_volume_calc("Acacia_brevispica", 2.2, 3.5, 1.12)
```

```
## [1] TRUE
```

```
tree_volume_calc("Balanites", 2.2, 3.5, 1.12)
```

```
## [1] TRUE
```

```
tree_volume_calc("Croton", 2.2, 3.5, 1.12)
```

```
## [1] TRUE
```

Now get the canopy volumes for all the trees in the tree_data dataframe and add them as a new column to the data frame. You can do this using tree_volume_calc() and either mapply() or using dplyr with rowwise and mutate.

```
library(dplyr)
```

```
##
## Attaching package: 'dplyr'
```

```
## The following objects are masked from 'package:stats':
##
##     filter, lag

## The following objects are masked from 'package:base':
##
##     intersect, setdiff, setequal, union
```

```
tree_data %>%
  rowwise() %>%
  mutate(VOLUME = tree_volume_calc(SPECIES, HEIGHT, AXIS_1, AXIS_2))
```

```
## # A tibble: 7,508 x 17
## # Rowwise:
##    SURVEY  YEAR SITE  TREATMENT BLOCK PLOT  SPECIES ORIGI~1 NEW_TAG DEAD  HEIGHT
##     <int> <int> <chr> <chr>     <int> <chr> <chr>     <int>   <int> <chr>  <dbl>
##  1      1  2009 SOUTH TOTAL         2 S2TO~ Acacia~       1      NA N       3.4
##  2      2  2010 SOUTH TOTAL         2 S2TO~ Acacia~       1      NA N       3.32
##  3      3  2011 SOUTH TOTAL         2 S2TO~ Acacia~       1      NA N       3.65
##  4      4  2012 SOUTH TOTAL         2 S2TO~ Acacia~       1      NA N       3.74
##  5      5  2013 SOUTH TOTAL         2 S2TO~ Acacia~       1      NA N       3.59
##  6      1  2009 SOUTH TOTAL         2 S2TO~ Acacia~       2      NA N       2.3
##  7      2  2010 SOUTH TOTAL         2 S2TO~ Acacia~       2      NA N       2.32
##  8      3  2011 SOUTH TOTAL         2 S2TO~ Acacia~       2      NA N       2.75
##  9      4  2012 SOUTH TOTAL         2 S2TO~ Acacia~       2      NA Y      NA
## 10      5  2013 SOUTH TOTAL         2 S2TO~ Acacia~       2      NA N       2.86
## # ... with 7,498 more rows, 6 more variables: AXIS_1 <dbl>, AXIS_2 <dbl>,
## #   CIRC <dbl>, MEASUREMENT <chr>, STEMS <chr>, VOLUME <lgl>, and abbreviated
## #   variable name 1: ORIGINAL_TAG
```

Exercise 3: Tree Growth

Write a function named get_growth() that takes two inputs, a vector of sizes and a vector of years, and calculates the average annual growth rate. Pseudo-code for calculating this rate is (size_in_last_year - size_in_first_year) / (last_year - first_year). Test this function by running get_growth(c(40.2, 42.6, 46.0), c(2020, 2021, 2022)).

“‘{r.eval = FALSE} sizes <- vector() years <- vector() get_growth <- function(sizes, years) { sizes <- c(size_in_last_year, size_in_first_year) years <- c(first_year, last_year) (size_in_last_year - size_in_first_year) / (last_year - first_year) }

get_growth(c(40.2, 42.6, 46.0), c(2020, 2021, 2022)) # Error in get_growth(c(40.2, 42.6, 46), c(2020, 2021, 2022)) : # object ‘size_in_last_year’ not found “‘