STAT 33A Workbook 8

CJ HINES (3034590053)

Oct 22, 2020

This workbook is due **Oct 22, 2020** by 11:59pm PT.

The workbook is organized into sections that correspond to the lecture videos for the week. Watch a video, then do the corresponding exercises *before* moving on to the next video.

Workbooks are graded for completeness, so as long as you make a clear effort to solve each problem, you'll get full credit. That said, make sure you understand the concepts here, because they're likely to reappear in homeworks, quizzes, and later lectures.

As you work, write your answers in this notebook. Answer questions with complete sentences, and put code in code chunks. You can make as many new code chunks as you like.

In the notebook, you can run the line of code where the cursor is by pressing `Ctrl` + `Enter` on Windows or `Cmd` + `Enter` on Mac OS X. You can run an entire code chunk by clicking on the green arrow in the upper right corner of the code chunk.

Please do not delete the exercises already in this notebook, because it may interfere with our grading tools.

You need to submit your work in two places:

- Submit this Rmd file with your edits on bCourses.
- Knit and submit the generated PDF file on Gradescope.

If you have any last-minute trouble knitting, **DON'T PANIC**. Submit your Rmd file on time and follow up in office hours or on Piazza to sort out the PDF.

Apply Function Basics

Watch the "Apply Function Basics" lecture video.

No exercises for this section.

```r
dogs = readRDS("dogs.rds")
```

Apply Function Examples

Watch the "Apply Function Examples" lecture video.

Exercise 1

1. Use `sapply` and `is.numeric` to identify all of the numeric columns in the dogs data frame.
2. Use `sapply`, your result from part 1, and `range` to compute the range of every numeric column in the dogs data frame. What's the class of the result? What do the rows and columns correspond to? Explain.

YOUR ANSWER GOES HERE:

Part 1

```
nummydog = sapply(dogs, is.numeric)
nummydog = nummydog[nummydog == TRUE]
```

Part 2

```
doggienum = dogs[names(nummydog)]
numranges = sapply(doggienum, range, na.rm = TRUE)
class(numranges)
```

```
## [1] "matrix" "array"
```

> What do the rows and columns correspond to? The first row is the minimum of all the given arguments and the second row is the maximum. The columns correspond to the dogs column names.

The Split-Apply Strategy

Watch the "The Split-Apply Strategy" lecture video.

No exercises for this section.

Split-Apply and dplyr

Watch the "Split-Apply and dplyr" lecture video.

Exercise 2

1. Use the split-apply strategy to compute the minimum weight (ignoring missing values) for each `size` of dog.
2. Use `tapply` to compute the same result as part 1.
3. Use dplyr to compute the same result as part 1 (but as a data frame).

YOUR ANSWER GOES HERE:

Part 1

```r
splitted = split(dogs$weight, dogs$size)
sapply(splitted, min, na.rm = TRUE)
```

```
##  large medium  small 
##     55     16      5
```

Part 2

```r
tapply(dogs$weight, dogs$size, min, na.rm = TRUE)
```

```
##  large medium  small 
##     55     16      5
```

Part 3

```r
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
```

```r
aggregate(dogs$weight, dogs["size"], min, na.rm = TRUE)
```

```
##     size  x
## 1  large 55
## 2 medium 16
## 3  small  5
```

Choosing an Apply Function

Watch the “Choosing an Apply Function” lecture video.

No exercises for this section. Almost finished!

Conditional Expressions

Watch the “Conditional Expressions” lecture video.

Exercise 3

A leap year is a year that contains an additional day, to correct for the fact that Earth's orbit lasts 365.25 days.

Every year divisible by 4 is a leap year, except years that are divsible by 100 and not divisible by 400.

Write an if-statement that checks whether the value of the variable `year` is a leap year. Assign the result to a variable called `result`. If `year` is a leap year, the result should be `"Leap"`. Otherwise, the result should be `"No leap"`.

Test that your code works for several different values of `year`. Make sure to include these tests in your answer.

Hint: Recall that the modulus operator `%%` does remainder division, so it is useful for checking whether one number is divisible by another.

YOUR ANSWER GOES HERE:

```
year = 2020


if (year %% 4 == 0 && year %% 100 != 0 && year %% 400 != 0) {
  result = "Leap"
} else {
  result = "No leap"
}
```

#Tests

```
year = 1943
if (year %% 4 == 0 && year %% 100 != 0 && year %% 400 != 0) {
  result = "Leap"
} else {
  result = "No leap"
}

year = 1560
if (year %% 4 == 0 && year %% 100 != 0 && year %% 400 != 0) {
  result = "Leap"
} else {
  result = "No leap"
}

year = 500
if (year %% 4 == 0 && year %% 100 != 0 && year %% 400 != 0) {
  result = "Leap"
} else {
  result = "No leap"
}
```

The Congruent Vectors Strategy

Watch the "The Congruent Vectors Strategy" lecture video.

Exercise 4

Use the congruent vectors strategy (not `ifelse`) to write a vectorized version of the leap year detector from Exercise 3.

Given a vector `year` of years, your code should return a congruent vector `result` that contains `"Leap"` or `"No leap"` for each element.

Test that your code works for several different values and lengths of `year`. Make sure to include these tests in your answer.

YOUR ANSWER GOES HERE:

```
year = c(2020, 1943, 1560, 500)
output = year
output[year %% 4 == 0 & year %% 100 != 0 & year %% 400 != 0] = "Leap"
output[year %% 4 != 0 | year %% 100 == 0 | year %% 400 == 0] = "No leap"
```

#Tests

```
year = c(1850, 1348)
output = year
output[year %% 4 == 0 & year %% 100 != 0 & year %% 400 != 0] = "Leap"
output[year %% 4 != 0 | year %% 100 == 0 | year %% 400 == 0] = "No leap"

year = c(1999, 1424, 1600, 1700, 1111, 1550, 1690)
output = year
output[year %% 4 == 0 & year %% 100 != 0 & year %% 400 != 0] = "Leap"
output[year %% 4 != 0 | year %% 100 == 0 | year %% 400 == 0] = "No leap"

year = c(2012)
output = year
output[year %% 4 == 0 && year %% 100 != 0 && year %% 400 != 0] = "Leap"
output[year %% 4 != 0 | year %% 100 == 0 | year %% 400 == 0] = "No leap"
```