Exercise 5: Functions

Marcy Shieh

9/24/2020

Prerequisites

1. Open you `ps811-exercises` folder.
2. Go to File > New File > R Script.
3. Save file as `exercise-6-functions.R`.
4. Load R packages installed in Lectures 5 and 6.

```
library(tidyverse)
library(here)
```

5. Download the national.csv file from Kaggle.
6. Load the CSV file into your R environment.

```
national <- read.csv(here("data/national.csv"))
```

7. View the CSV.

Loops

1. Write a loop to find how many variables there are per observations.

```
rows_loop <- numeric()

for (i in 1:ncol(national)) {
  rows_loop[i] <- length(national[i, ])
}

rows_loop
```

```
##  [1] 79 79 79 79 79 79 79 79 79 79 79 79 79 79 79 79 79 79 79 79 79 79 79 79 79
## [26] 79 79 79 79 79 79 79 79 79 79 79 79 79 79 79 79 79 79 79 79 79 79 79 79 79
## [51] 79 79 79 79 79 79 79 79 79 79 79 79 79 79 79 79 79 79 79 79 79 79 79 79 79
## [76] 79 79 79 79
```

2. Find the mean number of Protestant Christians in each country (i.e., the state column) using `tapply()` in both base R and Tidyverse.

```
# base R
tapply(
  X = national$christianity_protestant,
  INDEX = list(national$state),
```

```
  FUN = mean,
  na.rm = TRUE
)

# tidyverse
national %>%
  group_by(state) %>%
  summarize(
    mean_christ= mean(christianity_protestant, na.rm = TRUE)
  )
```

3. Check the column type for each variable that is just characters.

   Hint: You can use `sapply(dataset-name-here, class)`.

```
sapply(national, is.character)
```

4. Log the `buddhism` variables.

```
national %>%
  mutate_at(
    .vars = vars(starts_with("buddhism")),
    .funs = function(x) log(x)
  ) %>%
  select(starts_with("buddhism"))

# answers referring to only the "buddhism_all" variable are also acceptable
# I realize that this question is unclear (sorry!)
```

5. Write a function that lists all the unique years with more than 300,000 Christians in total.

```
unique_chr300k_yr <- function(x) {
    if(national$christianity_all > 300000) {
return(unique(national$year))
    }
}

unique_chr300k_yr()
```

```
## Warning in if (national$christianity_all > 3e+05) {: the condition has length >
## 1 and only the first element will be used
```

```
##  [1] 1945 1950 1955 1960 1965 1970 1975 1980 1985 1990 1995 2000 2005 2010
```

6. Group by the code variable.

   Hint: Use nesting.

```
nested_code <- national %>%
  group_by(code) %>%
  nest()
```

7. Create a model column. You need to include a linear regression model that examines whether the proportion of Jews (judaism_percent) affect whether a country has dual religions.

```
nested_models <- nested_code %>%
  mutate(
    model = map(
      .x = data,
```

```
      .f = ~ lm(dual_religion ~ judaism_percent, data = .x)
    )
  ) %>%
  print()
```

```
## # A tibble: 200 x 3
## # Groups:   code [200]
##    code  data               model
##    <chr> <list>             <list>
##  1 USA   <tibble [14 x 78]> <lm>
##  2 CAN   <tibble [14 x 78]> <lm>
##  3 BHM   <tibble [8 x 78]>  <lm>
##  4 CUB   <tibble [14 x 78]> <lm>
##  5 HAI   <tibble [14 x 78]> <lm>
##  6 DOM   <tibble [14 x 78]> <lm>
##  7 JAM   <tibble [11 x 78]> <lm>
##  8 TRI   <tibble [11 x 78]> <lm>
##  9 BAR   <tibble [9 x 78]>  <lm>
## 10 DMA   <tibble [7 x 78]>  <lm>
## # ... with 190 more rows
```

8. Extract the coefficients into a new column

```
nested_coefs <- nested_models %>%
  mutate(coefs = map(model, coefficients)) %>%
  print()
```

```
## # A tibble: 200 x 4
## # Groups:   code [200]
##    code  data               model  coefs
##    <chr> <list>             <list> <list>
##  1 USA   <tibble [14 x 78]> <lm>   <dbl [2]>
##  2 CAN   <tibble [14 x 78]> <lm>   <dbl [2]>
##  3 BHM   <tibble [8 x 78]>  <lm>   <dbl [2]>
##  4 CUB   <tibble [14 x 78]> <lm>   <dbl [2]>
##  5 HAI   <tibble [14 x 78]> <lm>   <dbl [2]>
##  6 DOM   <tibble [14 x 78]> <lm>   <dbl [2]>
##  7 JAM   <tibble [11 x 78]> <lm>   <dbl [2]>
##  8 TRI   <tibble [11 x 78]> <lm>   <dbl [2]>
##  9 BAR   <tibble [9 x 78]>  <lm>   <dbl [2]>
## 10 DMA   <tibble [7 x 78]>  <lm>   <dbl [2]>
## # ... with 190 more rows
```

9. Look at the coefficients.

```
nested_coefs$coefs
```

10. Pull out the model column.

```
nested_coefs %>% pull(model)
```

11. Unnest the coefficients column.

```
coefs <- nested_coefs %>%
  unnest(coefs) %>%
  print()
```

```
## # A tibble: 400 x 4
```

```
## # Groups:   code [200]
##    code  data               model  coefs
##    <chr> <list>             <list> <dbl>
##  1 USA   <tibble [14 x 78]> <lm>    0
##  2 USA   <tibble [14 x 78]> <lm>    0
##  3 CAN   <tibble [14 x 78]> <lm>    0
##  4 CAN   <tibble [14 x 78]> <lm>    0
##  5 BHM   <tibble [8 x 78]>  <lm>    0
##  6 BHM   <tibble [8 x 78]>  <lm>    0
##  7 CUB   <tibble [14 x 78]> <lm>    1.00
##  8 CUB   <tibble [14 x 78]> <lm>    0
##  9 HAI   <tibble [14 x 78]> <lm>    1.00
## 10 HAI   <tibble [14 x 78]> <lm>    0
## # ... with 390 more rows
```