Exercise 5: Base R vs. Tidyverse

Dillon Laaker

10/15/2020

Base R tasks

1. Download the food_coded.csv file
2. Load the CSV file into your R environment.

```
food <- read.csv("food_coded.csv")
```

```
Open the `codebook_food.docx` file for guidance.
```

3. Extract the first 95 rows.

```
food_2 <- food[1:95,]
```

4. Look at the following variables using both name and column index/number.
    - GPA
    - calories_chicken
    - drink
    - fav_cuisine
    - father_profession
    - mother_profession

```
food_3 <- food[, c("GPA", "calories_chicken", "drink", "fav_cuisine", "father_profession", "mother_profe
food_4 <- food[, c(1, 4, 16, 26, 25, 45)]
```

5. Create a new variable for how healthy each person feels but convert the scale from 1 to 10 to 1 to 100.

```
food_5 <- food
food_5$health <- food_5$healthy_feeling * 10
```

6. Filter to students who are female and have GPAs that are above 3.0.

```
food_6 <- food[food$Gender == 1 & food$GPA > 3, ]
```

7. Find the mean and standard deviation for the following variables, and summarize them in a data frame.

```
food_7 <- food[, c("calories_chicken", "tortilla_calories", "turkey_calories", "waffle_calories" )]

cal_means <- sapply(food_7, mean, na.rm = T)
cal_sd <- sapply(food_7, sd, na.rm = T)
cal_summary <- rbind(cal_means, cal_sd)
rownames(cal_summary) <- c("Mean calories", "SD of calories")
cal_summary
```

```
##                calories_chicken tortilla_calories turkey_calories
## Mean calories          577.3200          947.5806        555.0400
## SD of calories         131.2142          202.0902        152.3704
##                   waffle_calories
## Mean calories         1073.4000
## SD of calories         248.6671
```

```
* chicken_calories
* tortilla_calories
* turkey_calories
* waffle_calories
```

8. Summarize GPA and weight within the gender and cuisine variables.

```r
food$GPA <- as.numeric(food$GPA)
```

```
## Warning: NAs introduced by coercion
```

```r
food$weight <- as.numeric(food$weight)
```

```
## Warning: NAs introduced by coercion
```

```r
food$GPA[74] <- 3.79
food$weight[4] <- 240
food$weight[68] <- 144
food_men <- food[food$Gender == 1,]
food_women <- food[food$Gender ==2, ]
men_GPA_mean <- tapply(food_men$GPA, food_men$cuisine, mean, na.rm = T)
women_GPA_mean <- tapply(food_women$GPA, food_women$cuisine, mean, na.rm = T)
men_weight_mean <- tapply(food_men$weight, food_men$cuisine, mean, na.rm = T)
women_weight_mean <- tapply(food_women$weight, food_women$cuisine, mean, na.rm = T)

men_GPA_sd <- tapply(food_men$GPA, food_men$cuisine, sd, na.rm = T)
women_GPA_sd <- tapply(food_women$GPA, food_women$cuisine, sd, na.rm = T)
men_weight_sd <- tapply(food_men$weight, food_men$cuisine, sd, na.rm = T)
women_weight_sd <- tapply(food_women$weight, food_women$cuisine, sd, na.rm = T)

men_summary <- rbind(men_GPA_mean, men_GPA_sd, men_weight_mean, men_weight_sd)
men_summary <- cbind(men_summary, rep(0, length(men_summary[,1])))

rbind(women_GPA_mean, women_GPA_sd, women_weight_mean, women_weight_sd)
```

```
##                             1           2      3            4     5     6
## women_GPA_mean      3.3869375   3.6300000   3.87   2.75500000   3.2   3.9
## women_GPA_sd        0.3796562   0.2875761     NA   0.06363961    NA    NA
## women_weight_mean 177.8787879 178.0000000 195.00 138.00000000 185.0 185.0
## women_weight_sd    29.6929680  22.5277607     NA           NA    NA    NA
##                           NaN
## women_GPA_mean      3.2716667
## women_GPA_sd        0.5219355
## women_weight_mean 191.8000000
## women_weight_sd    38.8805864
```

Tidyverse tasks

1. Download the facebook-fact-check.csv
2. Load the CSV file into your R environment.

```r
fb_data <- read_csv("facebook-fact-check.csv")
```

```
## Rows: 2282 Columns: 12
## -- Column specification --------------------------------------------------------
## Delimiter: ","
## chr  (6): Category, Page, Post URL, Post Type, Rating, Debate
## dbl  (5): account_id, post_id, share_count, reaction_count, comment_count
## date (1): Date Published
##
## 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.
```

3. Extract the last 500 rows.

   Hint: Check out the top_n() page to figure out how to extract the last 500 rows instead of the first 500 rows.

```r
fb_data %>% slice_tail(n = 500)
```

```
## # A tibble: 500 x 12
##     account_id post_id Category Page  `Post URL`    `Date Published` `Post Type`
##          <dbl>   <dbl> <chr>    <chr> <chr>         <date>           <chr>
##  1 62317591679 1.02e16 mainstr~ Poli~ https://www.~ 2016-09-26       video
##  2 62317591679 1.02e16 mainstr~ Poli~ https://www.~ 2016-09-26       video
##  3 62317591679 1.02e16 mainstr~ Poli~ https://www.~ 2016-09-26       link
##  4 62317591679 1.02e16 mainstr~ Poli~ https://www.~ 2016-09-26       video
##  5 62317591679 1.02e16 mainstr~ Poli~ https://www.~ 2016-09-26       video
##  6 62317591679 1.02e16 mainstr~ Poli~ https://www.~ 2016-09-26       video
##  7 62317591679 1.02e16 mainstr~ Poli~ https://www.~ 2016-09-26       link
##  8 62317591679 1.02e16 mainstr~ Poli~ https://www.~ 2016-09-26       video
##  9 62317591679 1.02e16 mainstr~ Poli~ https://www.~ 2016-09-26       video
## 10 62317591679 1.02e16 mainstr~ Poli~ https://www.~ 2016-09-26       video
## # ... with 490 more rows, and 5 more variables: Rating <chr>, Debate <chr>,
## #   share_count <dbl>, reaction_count <dbl>, comment_count <dbl>
```

4. Look at the even-numbered column indices only. Identify them by name.

```r
fb_data %>% select(seq(from = 0, to = ncol(fb_data), by = 2)) %>% colnames()
```

```
## [1] "post_id"        "Page"           "Date Published" "Rating"
## [5] "share_count"    "comment_count"
```

5. Using `mutate`, create a new variable called `post_type_coded` that renames each post type to the following:

   - link = 1
   - photo = 2
   - text = 3
   - video = 4

   Hint: look up case_when within tidyverse. You can also use if_else

```r
fb_data <- fb_data %>%
    mutate(post_type_coded = case_when(
        `Post Type` == "link" ~ 1,
        `Post Type` == "photo" ~ 2,
        `Post Type` == "text" ~ 3,
        `Post Type` == "video" ~ 4
    ))
```

6. Arrange page names in reverse order.

```
fb_data %>% arrange(desc(Page))
```

```
## # A tibble: 2,282 x 13
##    account_id post_id Category Page   `Post URL`    `Date Published` `Post Type`
##         <dbl>   <dbl> <chr>    <chr>  <chr>         <date>           <chr>
##  1    1.15e14 1.46e15 left     The O~ https://www.~ 2016-09-19       photo
##  2    1.15e14 1.46e15 left     The O~ https://www.~ 2016-09-19       video
##  3    1.15e14 1.46e15 left     The O~ https://www.~ 2016-09-19       link
##  4    1.15e14 1.46e15 left     The O~ https://www.~ 2016-09-19       link
##  5    1.15e14 1.46e15 left     The O~ https://www.~ 2016-09-19       link
##  6    1.15e14 1.46e15 left     The O~ https://www.~ 2016-09-19       video
##  7    1.15e14 1.46e15 left     The O~ https://www.~ 2016-09-19       video
##  8    1.15e14 1.46e15 left     The O~ https://www.~ 2016-09-19       link
##  9    1.15e14 1.46e15 left     The O~ https://www.~ 2016-09-19       link
## 10    1.15e14 1.46e15 left     The O~ https://www.~ 2016-09-19       video
## # ... with 2,272 more rows, and 6 more variables: Rating <chr>, Debate <chr>,
## #   share_count <dbl>, reaction_count <dbl>, comment_count <dbl>,
## #   post_type_coded <dbl>
```

7. Find the mean and standard deviation for the following variables, and summarize them.

   - share_count
   - reaction_count
   - comment_count

```
fb_data %>%
    summarise(across(c(share_count, reaction_count, comment_count), list(mean = mean, sd = sd), na.rm =
```

```
## # A tibble: 1 x 6
##   share_count_mean share_count_sd reaction_count_mean reaction_count_sd
##              <dbl>          <dbl>               <dbl>             <dbl>
## 1            4045.         29832.               5364.            19127.
## # ... with 2 more variables: comment_count_mean <dbl>, comment_count_sd <dbl>
```

```
fb_data %>%
    group_by(Page) %>%
    summarise(Mean_Share_Count = mean(share_count, na.rm=T),
              SD_Share_count = sd(share_count, na.rm = T))
```

```
## # A tibble: 9 x 3
##   Page               Mean_Share_Count SD_Share_count
##   <chr>                         <dbl>          <dbl>
## 1 ABC News Politics              44.5           108.
## 2 Addicting Info               1270.           2037.
## 3 CNN Politics                  183.           1159.
## 4 Eagle Rising                  616.           2004.
## 5 Freedom Daily                2474.           4844.
## 6 Occupy Democrats            29205.          89934.
## 7 Politico                      182.            904.
## 8 Right Wing News              1398.           3639.
## 9 The Other 98%               18007.          40251.
```

8. Summarize the mean and standard deviations in Question 7 with the "mainstream" values in the `category` variable.

```
fb_data %>%
    filter(Category == "mainstream") %>%
    summarise(across(c(share_count, reaction_count, comment_count), list(mean = mean, sd = sd), na.rm =
```

```
## # A tibble: 1 x 6
##   share_count_mean share_count_sd reaction_count_mean reaction_count_sd
##              <dbl>          <dbl>               <dbl>             <dbl>
## 1             161.           940.                694.             1864.
## # ... with 2 more variables: comment_count_mean <dbl>, comment_count_sd <dbl>
```

Submit

Email me (laaker@wisc.edu) the link to your `ps811-exercises` repository when you are done.