Exercise 5: Base R vs. Tidyverse

Katrina Q. Wang

10/15/2020

Base R tasks

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

```
foodcode <- read.csv("/Users/katrinaqiyaowang/Desktop/R1/Materials/food_coded.csv")
```

3. Extract the first 95 rows.

```
foodextract <- foodcode[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

```
foodindex <- foodextract[ , c("GPA", "calories_chicken", "drink", "fav_cuisine", "father_profession", "
```

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

```
library(scales)
foodcode$newhealthy <- rescale(foodcode$healthy_feeling, to = c(1, 100))
```

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

```
femalefilter <- foodcode[foodcode$Gender == "1" & foodcode$GPA > 3.0, ]
```

7. Find the mean and standard deviation for the following variables, and summarize them in a data frame.
    - chicken_calories
    - tortilla_calories
    - turkey_calories
    - waffle_calories

```
fourcalories <- foodcode[ , c("calories_chicken", "tortilla_calories", "turkey_calories", "waffle_calor
sapply(fourcalories,mean,na.rm=TRUE)
```

```
##  calories_chicken tortilla_calories   turkey_calories   waffle_calories
##          577.3200          947.5806          555.0400         1073.4000
```

```
sapply(fourcalories,sd,na.rm=TRUE)
```

```
##  calories_chicken tortilla_calories   turkey_calories   waffle_calories
##          131.2142          202.0902          152.3704          248.6671
```

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

```
foodcode$GPA[grep("Unknown", foodcode$GPA)] <- ""
foodcode$GPA[grep("Personal", foodcode$GPA)] <- ""
foodcode$GPA[grep("nan", foodcode$GPA)] <- ""
foodcode$GPA[grep("3.79 bitch", foodcode$GPA)] <- "3.79"
foodcode$GPA[nchar(foodcode$GPA) == 0] <- NA
foodcode$GPA_num <- as.numeric(foodcode$GPA)
foodcode$Gender_cat <- as.character(foodcode$Gender)

aggregate(GPA_num ~ Gender_cat, foodcode, FUN=mean, na.rm=TRUE)
```

```
##   Gender_cat  GPA_num
## 1          1 3.438562
## 2          2 3.388375
```

```
aggregate(GPA_num ~ Gender_cat, foodcode, FUN=sum, na.rm=TRUE)
```

```
##   Gender_cat GPA_num
## 1          1 251.015
## 2          2 162.642
```

```
aggregate(GPA_num ~ Gender_cat, foodcode, FUN=sd, na.rm=TRUE)
```

```
##   Gender_cat   GPA_num
## 1          1 0.3783816
## 2          2 0.4092151
```

```
foodcode$weight[grep("Not sure, 240", foodcode$weight)] <- "240"
foodcode$weight[grep("nan", foodcode$weight)] <- ""
foodcode$weight[grep("I'm not answering this.", foodcode$weight)] <- ""
foodcode$weight[grep("144 lbs", foodcode$weight)] <- ""
foodcode$weight[nchar(foodcode$weight) == 0] <- NA
foodcode$weight_num <- as.numeric(foodcode$weight)

aggregate(weight_num ~ Gender_cat, foodcode, FUN=mean, na.rm=TRUE)
```

```
##   Gender_cat weight_num
## 1          1   146.4595
## 2          2   179.1915
```

```
aggregate(weight_num ~ Gender_cat, foodcode, FUN=sum, na.rm=TRUE)
```

```
##   Gender_cat weight_num
## 1          1      10838
## 2          2       8422
```

```
aggregate(weight_num ~ Gender_cat, foodcode, FUN=sd, na.rm=TRUE)
```

```
##   Gender_cat weight_num
## 1          1   27.85736
## 2          2   29.17834
```

Tidyverse tasks

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

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.

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

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

6. Arrange page names in reverse order.

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

   - share_count
   - reaction_count
   - comment_count

```
fb %>%
  summarize(avgshare=mean(share_count, na.rm=T),
            sdshare=sd(share_count, na.rm=T),
            avgreact=mean(reaction_count, na.rm=T),
            sdreact=sd(reaction_count, na.rm=T),
            avgcomm=mean(comment_count, na.rm=T),
            sdcomm=sd(comment_count, na.rm=T)
            )
```

```
##   avgshare  sdshare avgreact  sdreact  avgcomm   sdcomm
## 1 4044.816 29831.92 5364.285 19126.54 516.1022 3569.355
```

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

```
fb %>%
  filter(Category == "mainstream") %>%
  summarize(
    avgshare_m=mean(share_count, na.rm=T),
    sdshare_m=sd(share_count, na.rm=T),
    avgreact_m=mean(reaction_count, na.rm=T),
    sdreact_m=sd(reaction_count, na.rm=T),
    avgcomm_m=mean(comment_count, na.rm=T),
    sdcomm_m=sd(comment_count, na.rm=T)
)
```

```
##   avgshare_m sdshare_m avgreact_m sdreact_m avgcomm_m sdcomm_m
## 1   160.6568  939.8083   694.3712   1863.64  207.2218 346.0285
```

Submit

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