S&DS 220: Homework 7

Due Friday March 29

Braeden Cullen

Instructions

1. Complete the questions below. Upload your knitted PDF solutions to Gradescope by the due date.
2. Your solutions should be a combination of writing and R code. When writing, use complete sentences.
3. Previous homework assignments already had code chunks created for you. Now it is up to you to insert R code chunks within each problem as needed.
4. You should aim for clear and concise communication (in both words and R code).

Problem set questions

Question 1: (Exercise 6.2) Jane Austen novels

Consider the `austen` data set in the `fosdata` package. This data frame contains the complete texts of *Emma* and *Pride and Prejudice*, with additional information which you can read about in the help page for the data set. Each of the following tasks corresponds to using a single `dplyr` verb.

(a) Create a new data frame that consists only of the observations in *Emma*.

```
library(fosdata)
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
```

```
emma <- austen %>% filter(novel == "Emma")
head(emma)
```

```
##     word sentence chapter word_length stop_word sentiment_score novel
## 1   emma        1       1           4     FALSE               0  Emma
## 2     by        1       1           2      TRUE               0  Emma
## 3   jane        1       1           4     FALSE               0  Emma
## 4 austen        1       1           6     FALSE               0  Emma
## 5 volume        1       1           6     FALSE               0  Emma
## 6      i        1       1           1      TRUE               0  Emma
```

(b) Create a new data frame that contains only the variables `word`, `word_length` and `novel`.

```
word_wordlen_novel <- austen %>% select(word, word_length, novel)
head(word_wordlen_novel)
```

```
##     word word_length novel
## 1   emma           4  Emma
## 2     by           2  Emma
## 3   jane           4  Emma
## 4 austen           6  Emma
## 5 volume           6  Emma
## 6      i           1  Emma
```

(c) Create a new data frame that has the words in both books arranged in descending word length.

```
sort_austen <- austen %>% arrange(desc(word_length))
head(sort_austen)
```

```
##                  word sentence chapter word_length stop_word sentiment_score
## 1 conscience-stricken     5281      34          19     FALSE               0
## 2 respectable-looking     4334      43          19     FALSE               0
## 3   companionableness      220       2          17     FALSE               0
## 4   cheerful-tempered     1732      11          17     FALSE               0
## 5   unceremoniousness     1743      12          17     FALSE               0
## 6   manchester-street     5832      37          17     FALSE               0
##                 novel
## 1                Emma
## 2 Pride and Prejudice
## 3                Emma
## 4                Emma
## 5                Emma
## 6                Emma
```

(d) Create a new data frame that contains only the longest words that appeared in either of the books.

```
longest_austen <- austen %>% filter(word_length == max(word_length))
head(longest_austen)
```

```
##                  word sentence chapter word_length stop_word sentiment_score
## 1 conscience-stricken     5281      34          19     FALSE               0
## 2 respectable-looking     4334      43          19     FALSE               0
##                 novel
## 1                Emma
## 2 Pride and Prejudice
```

(e) What was the mean word length in the two books together?

```
austen %>% summarize(mean(word_length))
```

```
##   mean(word_length)
## 1          4.325518
```

(f) Create a new data frame that consists only of the distinct words found in the two books, together with the word length and sentiment score variables. (Hint: use `distinct`).

```
new_data_frame <- austen %>%
  select(word, word_length, sentiment_score) %>%
  distinct()

head(new_data_frame)
```

```
##     word word_length sentiment_score
## 1   emma           4               0
## 2     by           2               0
## 3   jane           4               0
## 4 austen           6               0
## 5 volume           6               0
## 6      i           1               0
```

Question 2: (Exercise 6.11, 6.15) Baseball batting statistics from the `Lahman` package

This exercise uses the `Batting` data set from the `Lahman` package. This gives the batting statistics of every player who has played baseball from 1871 through the present day. For these problems, once you identify the `playerID` for the answer, match it with the player's first and last name in the `People` data set, either by filtering for the `playerID` or using a `*join` function.

(a) Which player has the most lifetime at bats without ever having hit a home run?

```
library(Lahman)
library(dplyr)

max_at_bats_no_hr <- Batting %>%
  group_by(playerID) %>% # group by playerID
  summarize(total_at_bat = sum(AB), # summarize total at bat
            total_home_run = sum(HR)) %>% # summarize total home run
  filter(total_home_run == 0) %>% # filter for total home run equal to 0
  # find most at bats without a home run
  slice_max(total_at_bat, n = 1)
# matching playerID with first and last name
People %>% filter(playerID == max_at_bats_no_hr$playerID) %>% select(nameFirst, nameLast)
```

```
##   nameFirst nameLast
## 1      Dave   Eggler
```

(b) Which active player has the most lifetime at bats without ever having hit a home run? (An active player is someone with an entry in the most recent year of the data set).

```
active_most_at_bat_no_home_run <- Batting %>%
  group_by(playerID) %>%
  summarize(lastest_year = max(yearID),
            total_ab = sum(AB), # summarize total at bat
            total_hr = sum(HR)) %>% # summarize total home run
  filter(total_hr == 0, # filter for total home run equal to 0
         lastest_year == max(Batting$yearID)) %>% # filter for active player
  slice_max(total_ab, n = 1) # find most at bats without a home run
# matching playerID with first and last name
People %>% filter(playerID == active_most_at_bat_no_home_run$playerID) %>% select(nameFirst, nameLast)
```

```
##   nameFirst nameLast
## 1 Magneuris   Sierra
```

(c) Which player has been hit-by-pitch the most number of times?

```
hit_by_pitch <- Batting %>%
  group_by(playerID) %>%
  summarize(total_hit_by_pitch = sum(HBP, na.rm = TRUE)) %>% # summarize total hit by pitch
  slice_max(total_hit_by_pitch, n = 1)
# hit_by_pitch playerID with first and last name
People %>% filter(playerID == hit_by_pitch$playerID) %>% select(nameFirst, nameLast)
```

```
##   nameFirst nameLast
## 1    Hughie Jennings
```

Question 3: (Exercise 6.25) Storms

Consider the `storms` data set in the `dplyr` package, from Example 6.5. Recall that `name` and `year` together identify all storms except Zeta (2005-2006).

(a) Which name(s) was/were given to the most storms?

```
storms %>%
  select(name, year) %>%  # select name and year
  distinct() %>%  # distinct name and year
  group_by(name) %>%  # group by name
  summarize(count = n()) %>% # summarize count
  slice_max(count, n = 1) # find most storms given to a name
```

```
## # A tibble: 5 x 2
##   name      count
##   <chr>     <int>
## 1 Ana           8
## 2 Bonnie        8
## 3 Claudette     8
## 4 Danielle      8
## 5 Earl          8
```

(b) Which year(s) had the most named storms?

```
storms %>%
  select(name, year) %>%  # select name and year
  distinct() %>%  # distinct name and year
  group_by(year) %>% # group by year
  summarize(count = n()) %>% # summarize count
  slice_max(count, n = 1) # find most named storms in a year
```

```
## # A tibble: 1 x 2
##    year count
##   <dbl> <int>
## 1  2020    30
```

(c) The second strongest storm named Lili had maximum wind speed of 100. Which name's second strongest storm in terms of maximum wind speed was the strongest among all names' second strongest storms? The `dplyr` function `nth` may be useful for doing this problem.

```
storms %>%
  group_by(name, year) %>%
  summarize(max_wind = max(wind)) %>% # summarize max wind
  arrange(name, desc(max_wind)) %>%  # arrange by name and max wind
  filter(max_wind == nth(max_wind, 2)) %>%  # filter for second strongest storm
  ungroup() %>% # ungroup
  slice_max(max_wind) # find strongest second strongest storm
```

```
## `summarise()` has grouped output by 'name'. You can override using the
## `.groups` argument.
```

```
## # A tibble: 1 x 3
##   name   year max_wind
##   <chr> <dbl>    <int>
## 1 Felix  1995      120
```

Question 4: (Exercise 6.29) Fruits

The data set `fruit` is built into the `stringr` package.

(a) How many fruits have the word "berry" in their name?

```
library(stringr)
sum(str_detect(fruit, pattern = "berry")) # count fruits with "berry" in their name
```

```
## [1] 14
```

(b) Some of these fruits have the word "fruit" in their name. Find these fruit and remove the word "fruit" to create a list of words that can be made into fruit. (Hint: use `str_remove`)

```
fruit_in_name <- str_detect(fruit, "fruit") # detect fruit with "fruit" in their name
remove_fruit <- str_remove(fruit[fruit_in_name], pattern = "fruit") %>% # remove "fruit" from fruit nam
  str_trim() # trim white space
remove_fruit
```

```
## [1] "bread"   "dragon"  "grape"   "jack"    "kiwi"    "passion" "star"
## [8] "ugli"
```

Question 5: (Exercise 6.36) Scotland births

Consider the `scotland_births` data set in the `fosdata` package. This data gives the number of births by the age of the mother in Scotland for each year from 1945-2019. This data is in wide format. (Completion of this exercise will be helpful for Exercise 7.28.)

(a) Convert the data into long format with three variable names: `age`, `year` and `births`, where each observation is the number of births in year to mothers that are age years old.

```
library(fosdata)
library(dplyr)
library(tidyr)

births_converted <- scotland_births %>%
  # convert data into long format
  pivot_longer(cols = starts_with("x"), names_to = "year", values_to = "births")
head(births_converted)
```

```
## # A tibble: 6 x 3
##     age year  births
##   <int> <chr>  <int>
## 1    12 x1945      0
## 2    12 x1946      0
## 3    12 x1947      0
## 4    12 x1948      0
## 5    12 x1949      0
## 6    12 x1950      0
```

(b) Convert the `year` to integer by removing the `x` and using `as.integer`.

```
births_converted <- births_converted %>%
  mutate(year = as.integer(str_remove(year, "x"))) # convert year to integer
head(births_converted)
```

```
## # A tibble: 6 x 3
##     age  year births
##   <int> <int>  <int>
## 1    12  1945      0
## 2    12  1946      0
## 3    12  1947      0
## 4    12  1948      0
## 5    12  1949      0
## 6    12  1950      0
```

(c) Which year had the most babies born to mothers 20-years-old or younger?

```
births_converted %>%
  filter(age <= 20) %>%
  group_by(year) %>%
  summarize(total_births = sum(births)) %>% # summarize total births
  slice_max(total_births, n = 1) # find year with most babies born to mothers 20-years-old or younger
```

```
## # A tibble: 1 x 2
##    year total_births
##   <int>        <int>
## 1  1967        15457
```