STAT 33B Homework 7

Gunnar Mayer (3034515354)

This assignment is due **May 6, 2020** by 11:59pm.

Edit this file, knit to PDF, and:

- Submit the Rmd file on bCourses.
- Submit the PDF file on Gradescope.

If you think you'll need help with submission, please ask in office hours *before* the assignment is due.

Answer all questions with complete sentences, and put code in code chunks. You can make as many new code chunks as you like. Please do not delete the exercises already in this notebook, because it may interfere with our grading tools.

Return of the Bay Area Apartments Dataset

The exercises in this assignment reuse the the Bay Area Apartments Data Set.

Please make sure to download the **new version** of the the data set from the bCourse for this assignment.

You can find a full description of the data set in Lab 3.

Exercise 0

Put all of your calls to `library()` here:

```
# Your code goes here.
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
```

Exercise 1

Use dplyr to compute subsets and answer the following:

1. How many apartments are listed in Alameda?
2. How many of those apartments in Alameda have in-unit laundry?
3. What proportion of apartments between 800 and 1000 square feet (inclusive) have a rent price below 2200 dollars?

```
# Your code goes here.
ba_data = readRDS("apartments.rds")
alameda_data = filter(ba_data, county == "Alameda")

num_apart = summarise(alameda_data,n = n())
num_apart
```

```
## # A tibble: 1 x 1
##       n
##   <int>
## 1  2321
```

```
num_laundry = summarise(filter(alameda_data, laundry == "in-unit"), n = n())
num_laundry
```

```
## # A tibble: 1 x 1
##       n
##   <int>
## 1   863
```

```
total_size = between(ba_data$sqft, 800, 1000)
price_filter = summarize(filter(ba_data, total_size, price <2200), n=n())
price_filter
```

```
## # A tibble: 1 x 1
##       n
##   <int>
## 1   431
```

```
total_size_sum = sum(total_size, na.rm = TRUE)
total_size_sum
```

```
## [1] 1312
```

```
ans = price_filter / total_size
ans
```

```
##     n
## 1 Inf
```

YOUR WRITTEN ANSWERS GO HERE:

1. There are 2321 apartments in Alameda county
2. There are 863 apartments with in-unit laundry in Alameda county 3.Less than 33% of the apartments in the Bay Area that are between 800 and 100 sqft are less than $2200. The proportion specifcally is 431 / 1312

Exercise 2

Use dplyr's `count()` function to compute a data frame that shows the number of bedrooms versus the number of bathrooms for all apartments in Oakland and San Francisco.

Hint: You solved this exercise with base R functions in Lab 3.

```
# Your code goes here.

filter(ba_data, county == "San Francisco" | county == "Oakland") %>% group_by(bedrooms, bathrooms) %>% c
```

```
## # A tibble: 21 x 3
## # Groups:   bedrooms, bathrooms [21]
##    bedrooms bathrooms     n
##       <dbl>     <dbl> <int>
##  1        0       1     132
##  2        0       1.5     1
##  3        1       0       4
##  4        1       1     280
##  5        1       1.5     6
##  6        2       0       3
##  7        2       1      73
##  8        2       1.5     9
##  9        2       2      81
## 10        2       2.5     2
## # ... with 11 more rows
```

Exercise 3

Use dplyr to compute the maximum apartment price for each city, and sort your results. What are the 3 cities with the highest maximum apartment price?

Hint: See dplyr's online documentation for examples of how to use the `group_by()`, `summarize()`, and `arrange()` functions.

```
# Your code goes here.
ba_data %>% group_by(city) %>% summarize(max_price = max(price)) %>% arrange(desc(max_price))
```

```
## Warning: Factor `city` contains implicit NA, consider using
## `forcats::fct_explicit_na`
```

```
## # A tibble: 96 x 2
##    city          max_price
##    <fct>             <dbl>
##  1 Berkeley          16200
##  2 San Francisco      9500
##  3 Oakland            8900
##  4 Mountain View      6325
##  5 Redwood City       6195
##  6 Los Altos          5945
##  7 Millbrae           5900
##  8 Walnut Creek       5643
##  9 Mill Valley        5500
```

```
## 10 Foster City        5366
## # ... with 86 more rows
```

YOUR WRITTEN ANSWER GOES HERE: The cities with the highest max rent are:

Berkeley || $16200
San Francisco || $9500
Oakland || $8900

Exercise 4

Compute the median apartment price (ignoring missing values) for each of the different parking options. Use each of the three different approaches to the split-apply pattern:

1. With `split()` and `sapply()`.
2. With `tapply()`.
3. With dplyr's `group_by()` and `summarize()`.

Confirm that the numerical results from each are the same. Describe other differences between the results, such as the type and class of the returned object.

```
# Your code goes here.

# 1
d1 = split(ba_data$price, f= ba_data$parking)
ans1 = sapply(d1, median, na.rm = TRUE)
ans1
```

```
##    covered     garage       none off-street       paid     street
##       2189       2815       2150       2350       3890       2495
```

```
typeof(ans1)
```

```
## [1] "double"
```

```
class(ans1)
```

```
## [1] "numeric"
```

```
# 2
ans2 = tapply(ba_data$price, ba_data$parking, median)
ans2
```

```
##    covered     garage       none off-street       paid     street
##       2189         NA         NA       2350       3890       2495
```

```
typeof(ans2)
```

```
## [1] "double"
```

```r
class(ans2)
```

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

```r
# 3
ans3 = ba_data %>% filter(!is.na(parking), !is.na(price)) %>% group_by(parking) %>% summarise(median(pr
ans3
```

```
## # A tibble: 6 x 2
##   parking    `median(price)`
##   <fct>                <dbl>
## 1 covered               2189
## 2 garage                2815
## 3 none                  2150
## 4 off-street            2350
## 5 paid                  3890
## 6 street                2495
```

```r
typeof(ans3)
```

```
## [1] "list"
```

```r
class(ans3)
```

```
## [1] "tbl_df"     "tbl"        "data.frame"
```

YOUR WRITTEN ANSWER GOES HERE: The first 2 ways return doubles as answers, however using dplyr returns a list. The first class using sapply returns a numeric class. The second method returns an array class. Dplyr returns a dataframe. Using tapply was definetly the simplist way to do it.