Laboratory Exercise Week 4

Brian Tipton - 001

9/13/23

Directions:

- Write your R code inside the code chunks after each question.
- Write your answer comments after the `#` sign.
- To generate the word document output, click the button `Knit` and wait for the word document to appear.
- RStudio will prompt you (only once) to install the `knitr` package.
- Submit your completed laboratory exercise using Blackboard's Turnitin feature. Your Turnitin upload link is found on your Blackboard Course shell under the Laboratory folder.

For this exercise, you will need to use the package `mosaic` to find numerical and graphical summaries.

```
# install packages if necessary
if (!require(mosaic)) install.packages(mosaic)
if (!require(dplyr)) install.packages(dplyr)
if (!require(gapminder)) install.packages(gapminder)
# load the package in R
library(mosaic) # load the package mosaic to use its functions
library(dplyr) # load the package dplyr to use its functions
library(gapminder)  # load the package gapminder for question 1
```

1. Using the gapminder data in the lesson, do the following:
   i) use `filter` to select all countries with the following arguments:
      a) life expectancy larger than 60 years.

      b) United Kingdom and Vietnam and years greater than 1990.

   ii) use `arrange` and `slice` to select the countries with the top 15 GDP per capital `gdpPercap`. Use the pipe `%>%` operator to string multiple functions.
   iii) use `mutate` to create a new variable called `gdpPercap_lifeExp` which is the quotient of `gdpPercap` and `lifeExp` and display the output.

   iv) use `summarise` to find the average or mean value of the variable `gdpPercap_lifeExp` created in part (iii).

   v) use `group_by` to group the countries by `continent`; and `summarise` to compute the average life expectancy `lifeExp` within each continent. Use the pipe `%>%` operator to string multiple functions.

Code chunk

```
library(mosaic)
library(dplyr)
```

```r
library(gapminder)
 ##  i) use `filter` to select all countries with the following arguments:
 ##  a) life expectancy larger than 60 years.
 ##  b) United Kingdom and Vietnam and years greater than 1990.
dplyr::filter(gapminder,
              lifeExp > 60,
              country %in% c("United Kingdom", "Vietnam") & year > 1990)
```

```
## # A tibble: 8 x 6
##   country        continent  year lifeExp      pop gdpPercap
##   <fct>          <fct>     <int>   <dbl>    <int>     <dbl>
## 1 United Kingdom Europe     1992    76.4 57866349    22705.
## 2 United Kingdom Europe     1997    77.2 58808266    26075.
## 3 United Kingdom Europe     2002    78.5 59912431    29479.
## 4 United Kingdom Europe     2007    79.4 60776238    33203.
## 5 Vietnam        Asia       1992    67.7 69940728      989.
## 6 Vietnam        Asia       1997    70.7 76048996     1386.
## 7 Vietnam        Asia       2002    73.0 80908147     1764.
## 8 Vietnam        Asia       2007    74.2 85262356     2442.
```

```r
## ii) use `arrange` and `slice` to select the countries with the top 15 GDP per capital `gdpPercap`.
## Use the pipe `%>%` operator to string multiple functions.
gapminder %>%
  dplyr::arrange(desc(gdpPercap)) %>%
  dplyr::slice(1:15)
```

```
## # A tibble: 15 x 6
##    country          continent  year lifeExp       pop gdpPercap
##    <fct>            <fct>     <int>   <dbl>     <int>     <dbl>
##  1 Kuwait           Asia       1957    58.0    212846   113523.
##  2 Kuwait           Asia       1972    67.7    841934   109348.
##  3 Kuwait           Asia       1952    55.6    160000   108382.
##  4 Kuwait           Asia       1962    60.5    358266    95458.
##  5 Kuwait           Asia       1967    64.6    575003    80895.
##  6 Kuwait           Asia       1977    69.3   1140357    59265.
##  7 Norway           Europe     2007    80.2   4627926    49357.
##  8 Kuwait           Asia       2007    77.6   2505559    47307.
##  9 Singapore        Asia       2007    80.0   4553009    47143.
## 10 Norway           Europe     2002    79.0   4535591    44684.
## 11 United States    Americas   2007    78.2 301139947    42952.
## 12 Norway           Europe     1997    78.3   4405672    41283.
## 13 Ireland          Europe     2007    78.9   4109086    40676.
## 14 Kuwait           Asia       1997    76.2   1765345    40301.
## 15 Hong Kong, China Asia       2007    82.2   6980412    39725.
```

```r
## iii) use `mutate` to create a new variable called `gdpPercap_lifeExp` which is the quotient of
##`gdpPercap` and `lifeExp`. and display the output.
gapminder <- gapminder %>%
  dplyr::mutate(gdpPercap_lifeExp = gdpPercap / lifeExp)
dplyr::select(gapminder, gdpPercap_lifeExp)
```

```
## # A tibble: 1,704 x 1
##    gdpPercap_lifeExp
##                <dbl>
##  1              27.1
```

```
##  2              27.1
##  3              26.7
##  4              24.6
##  5              20.5
##  6              20.5
##  7              24.5
##  8              20.9
##  9              15.6
## 10              15.2
## # i 1,694 more rows
```

```
## iv) use `summarise` to find the average or mean value of the variable `gdpPercap_lifeExp` created in
gapminder %>%
  dplyr::summarise(mean_gdpPercap_lifeExp = mean(gdpPercap_lifeExp))
```

```
## # A tibble: 1 x 1
##   mean_gdpPercap_lifeExp
##                    <dbl>
## 1                   106.
```

```
## use `group_by` to group the countries by `continent`;
## and `summarise` to compute the average life expectancy `lifeExp` within each continent.
## Use the pipe `%>%` operator to string multiple functions.
gapminder %>%
  dplyr::group_by(continent) %>%
  dplyr::summarise(mean_lifeExp = mean(lifeExp))
```

```
## # A tibble: 5 x 2
##   continent mean_lifeExp
##   <fct>            <dbl>
## 1 Africa            48.9
## 2 Americas          64.7
## 3 Asia              60.1
## 4 Europe            71.9
## 5 Oceania           74.3
```

2. The data set `MLB-TeamBatting-S16.csv` contains MLB Team Batting Data for selected variables. Load the data set from the given url using the code below. This data set was obtained from Baseball Reference.
   - Tm - Team
   - Lg - League: American League (AL), National League (NL)
   - BatAge - Batters' average age
   - RPG - Runs Scored Per Game
   - G - Games Played or Pitched
   - AB - At Bats
   - R - Runs Scored/Allowed
   - H - Hits/Hits Allowed
   - HR - Home Runs Hit/Allowed

- RBI - Runs Batted In
- SO - Strikeouts
- BA - Hits/At Bats
- SH - Sacrifice Hits (Sacrifice Bunts)
- SF - Sacrifice Flies

Using the `mlb16.data` data, do the following:

i) use `filter` to select teams with the following arguments:
   a) Cardinals team `STL`.
   b) teams with Hits `H` more than 1400 last 2016 season.
   c) team league `Lg` is National League `NL`.

ii) use `arrange` to select teams in decreasing number of home runs `HR`.

iii) use `arrange` to display the teams in decreasing number of `RBI`.

iv) use `group_by` to group the teams per league; and `summarise` to compute the average `RBI` within each league. Use the pipe `%>%` operator to string multiple functions.

Code chunk

```
library(mosaic)
library(dplyr)

# load the data set
mlb16.data <-
  read.csv(
    "https://raw.githubusercontent.com/jpailden/rstatlab/master/data/MLB-TeamBatting-S16.csv"
  )
str(mlb16.data) # check structure
```

```
## 'data.frame':    30 obs. of  14 variables:
##  $ Tm    : chr  "ARI" "ATL" "BAL" "BOS" ...
##  $ Lg    : chr  "NL" "NL" "AL" "AL" ...
##  $ BatAge: num  26.7 28.9 28.4 28.5 27.4 28.3 27.8 28.9 27.8 29.8 ...
##  $ RPG   : num  4.64 4.03 4.59 5.42 4.99 4.23 4.42 4.83 5.22 4.66 ...
##  $ G     : int  162 161 162 162 162 162 162 161 162 161 ...
##  $ AB    : int  5665 5514 5524 5670 5503 5550 5487 5484 5614 5526 ...
##  $ R     : int  752 649 744 878 808 686 716 777 845 750 ...
##  $ H     : int  1479 1404 1413 1598 1409 1428 1403 1435 1544 1476 ...
##  $ HR    : int  190 122 253 208 199 168 164 185 204 211 ...
##  $ RBI   : int  709 615 710 836 767 656 678 733 805 719 ...
##  $ SO    : int  1427 1240 1324 1160 1339 1285 1284 1246 1330 1303 ...
##  $ BA    : num  0.261 0.255 0.256 0.282 0.256 0.257 0.256 0.262 0.275 0.267 ...
##  $ SH    : int  43 64 17 8 42 29 58 31 54 17 ...
##  $ SF    : int  38 52 36 40 37 44 44 60 34 38 ...
```

```
head(mlb16.data)  # show first six rows
```

```
##    Tm Lg BatAge  RPG   G   AB   R    H  HR RBI   SO    BA SH SF
## 1 ARI NL   26.7 4.64 162 5665 752 1479 190 709 1427 0.261 43 38
```

```
## 2 ATL NL   28.9 4.03 161 5514 649 1404 122 615 1240 0.255 64 52
## 3 BAL AL   28.4 4.59 162 5524 744 1413 253 710 1324 0.256 17 36
## 4 BOS AL   28.5 5.42 162 5670 878 1598 208 836 1160 0.282  8 40
## 5 CHC NL   27.4 4.99 162 5503 808 1409 199 767 1339 0.256 42 37
## 6 CHW AL   28.3 4.23 162 5550 686 1428 168 656 1285 0.257 29 44
```

```
# i) use `filter` to select teams with the following arguments:
# a) Cardinals team `STL`.
# b) teams with Hits `H` more than 1400 last 2016 season.
# c) team league `Lg` is National League `NL`.
dplyr::filter(mlb16.data, Tm == "STL", H > 1400, Lg == "NL")
```

```
##    Tm Lg BatAge  RPG   G   AB   R    H  HR RBI   SO    BA SH SF
## 1 STL NL   28.5 4.81 162 5548 779 1415 225 745 1318 0.255 37 41
```

```
# use `arrange` to select teams in decreasing number of home runs `HR`.
mlb16.data %>% dplyr::arrange(desc(HR))
```

```
##     Tm Lg BatAge  RPG   G   AB   R    H  HR RBI   SO    BA SH SF
## 1  BAL AL   28.4 4.59 162 5524 744 1413 253 710 1324 0.256 17 36
## 2  STL NL   28.5 4.81 162 5548 779 1415 225 745 1318 0.255 37 41
## 3  SEA AL   30.4 4.74 162 5583 768 1446 223 735 1288 0.259 24 41
## 4  TOR AL   30.0 4.69 162 5479 759 1358 221 728 1362 0.248 26 40
## 5  NYM NL   29.5 4.14 162 5459 671 1342 218 649 1302 0.246 35 41
## 6  TBR AL   27.7 4.15 162 5481 672 1333 216 647 1482 0.243 18 28
## 7  TEX AL   28.4 4.72 162 5525 765 1446 215 746 1220 0.262 18 40
## 8  DET AL   29.8 4.66 161 5526 750 1476 211 719 1303 0.267 17 38
## 9  BOS AL   28.5 5.42 162 5670 878 1598 208 836 1160 0.282  8 40
## 10 COL NL   27.8 5.22 162 5614 845 1544 204 805 1330 0.275 54 34
## 11 WSN NL   28.8 4.71 162 5490 763 1403 203 735 1252 0.256 48 63
## 12 MIN AL   27.0 4.46 162 5618 722 1409 200 690 1426 0.251 27 43
## 13 CHC NL   27.4 4.99 162 5503 808 1409 199 767 1339 0.256 42 37
## 14 HOU AL   26.6 4.47 162 5545 724 1367 198 689 1452 0.247 27 31
## 15 MIL NL   27.5 4.14 162 5330 671 1299 194 641 1543 0.244 53 39
## 16 ARI NL   26.7 4.64 162 5665 752 1479 190 709 1427 0.261 43 38
## 17 LAD NL   28.9 4.48 162 5518 725 1376 189 680 1321 0.249 30 32
## 18 CLE AL   28.9 4.83 161 5484 777 1435 185 733 1246 0.262 31 60
## 19 NYY AL   29.9 4.20 162 5458 680 1378 183 647 1188 0.252 21 49
## 20 SDP NL   28.1 4.23 162 5419 686 1275 177 654 1500 0.235 36 36
## 21 OAK AL   28.7 4.03 162 5500 653 1352 169 634 1145 0.246 13 34
## 22 CHW AL   28.3 4.23 162 5550 686 1428 168 656 1285 0.257 29 44
## 23 CIN NL   27.8 4.42 162 5487 716 1403 164 678 1284 0.256 58 44
## 24 PHI NL   26.9 3.77 162 5434 610 1305 161 574 1376 0.240 46 30
## 25 LAA AL   28.5 4.43 162 5431 717 1410 156 686  991 0.260 36 49
## 26 PIT NL   28.9 4.50 162 5542 729 1426 153 696 1334 0.257 41 36
## 27 KCR AL   28.6 4.17 162 5552 675 1450 147 640 1224 0.261 38 34
## 28 SFG NL   29.2 4.41 162 5565 715 1437 130 675 1107 0.258 42 46
## 29 MIA NL   28.3 4.07 161 5547 655 1460 128 626 1213 0.263 46 38
## 30 ATL NL   28.9 4.03 161 5514 649 1404 122 615 1240 0.255 64 52
```

```
#  use `arrange` to display the teams in decreasing number of `RBI`.
mlb16.data %>% dplyr::arrange(desc(RBI))
```

```
##     Tm Lg BatAge  RPG   G   AB   R    H  HR RBI   SO    BA SH SF
## 1  BOS AL   28.5 5.42 162 5670 878 1598 208 836 1160 0.282  8 40
## 2  COL NL   27.8 5.22 162 5614 845 1544 204 805 1330 0.275 54 34
```

```
## 3  CHC NL   27.4 4.99 162 5503 808 1409 199 767 1339 0.256 42 37
## 4  TEX AL   28.4 4.72 162 5525 765 1446 215 746 1220 0.262 18 40
## 5  STL NL   28.5 4.81 162 5548 779 1415 225 745 1318 0.255 37 41
## 6  SEA AL   30.4 4.74 162 5583 768 1446 223 735 1288 0.259 24 41
## 7  WSN NL   28.8 4.71 162 5490 763 1403 203 735 1252 0.256 48 63
## 8  CLE AL   28.9 4.83 161 5484 777 1435 185 733 1246 0.262 31 60
## 9  TOR AL   30.0 4.69 162 5479 759 1358 221 728 1362 0.248 26 40
## 10 DET AL   29.8 4.66 161 5526 750 1476 211 719 1303 0.267 17 38
## 11 BAL AL   28.4 4.59 162 5524 744 1413 253 710 1324 0.256 17 36
## 12 ARI NL   26.7 4.64 162 5665 752 1479 190 709 1427 0.261 43 38
## 13 PIT NL   28.9 4.50 162 5542 729 1426 153 696 1334 0.257 41 36
## 14 MIN AL   27.0 4.46 162 5618 722 1409 200 690 1426 0.251 27 43
## 15 HOU AL   26.6 4.47 162 5545 724 1367 198 689 1452 0.247 27 31
## 16 LAA AL   28.5 4.43 162 5431 717 1410 156 686  991 0.260 36 49
## 17 LAD NL   28.9 4.48 162 5518 725 1376 189 680 1321 0.249 30 32
## 18 CIN NL   27.8 4.42 162 5487 716 1403 164 678 1284 0.256 58 44
## 19 SFG NL   29.2 4.41 162 5565 715 1437 130 675 1107 0.258 42 46
## 20 CHW AL   28.3 4.23 162 5550 686 1428 168 656 1285 0.257 29 44
## 21 SDP NL   28.1 4.23 162 5419 686 1275 177 654 1500 0.235 36 36
## 22 NYM NL   29.5 4.14 162 5459 671 1342 218 649 1302 0.246 35 41
## 23 NYY AL   29.9 4.20 162 5458 680 1378 183 647 1188 0.252 21 49
## 24 TBR AL   27.7 4.15 162 5481 672 1333 216 647 1482 0.243 18 28
## 25 MIL NL   27.5 4.14 162 5330 671 1299 194 641 1543 0.244 53 39
## 26 KCR AL   28.6 4.17 162 5552 675 1450 147 640 1224 0.261 38 34
## 27 OAK AL   28.7 4.03 162 5500 653 1352 169 634 1145 0.246 13 34
## 28 MIA NL   28.3 4.07 161 5547 655 1460 128 626 1213 0.263 46 38
## 29 ATL NL   28.9 4.03 161 5514 649 1404 122 615 1240 0.255 64 52
## 30 PHI NL   26.9 3.77 162 5434 610 1305 161 574 1376 0.240 46 30
```

```
# use `group_by` to group the teams per league;
# and `summarise` to compute the average `RBI` within each league.
# Use the pipe `%>%` operator to string multiple functions.
mlb16.data %>%
  dplyr::group_by(Lg) %>%
  dplyr::summarise(avg_RBI = mean(RBI))
```

```
## # A tibble: 2 x 2
##   Lg    avg_RBI
##   <chr>   <dbl>
## 1 AL       700.
## 2 NL       683.
```