Practice2

Sebastian Montesinos

Due by midnight, Friday, Feb. 25

Practice2

Reminder: Practice assignments may be completed working with other individuals.

Reading

The associated reading for the week is Chapter 4, Chapter 5, Chapter 6 (skip 6.4), and Sections 8.3 and 8.4.

Practicing Academic Integrity

If you worked with others or used resources outside of provided course material (anything besides our textbook, course materials in the repo, labs, R help menu) to complete this assignment, please acknowledge them below using a bulleted list.

I acknowledge the following individuals with whom I worked on this assignment:

Name(s) and corresponding problem(s)

-

I used the following sources to help complete this assignment:

Source(s) and corresponding problem(s)

-

1 - Hardest Concept

We've covered many different data wrangling concepts and associated verbs during this unit. This problem will help you identify ways to get support on concepts you find challenging, beyond what we have in class and in the textbook.

> part a - What concept or data wrangling verb did you find most challenging to work with during this unit?

Solution:

> part b - Look in our Resources folder at the tidyr and data-transformation cheat sheets. Can you find information related to your selected concept or verb? If so, what sheet is it in? What if any insights do you get from the cheatsheet?

(If you picked a concept or verb not on these cheatsheets, try to find it on a different one, or ask me where it is likely to be. These are just the two most common cheatsheets to reference for these chapters.)

Solution:

> part c - Most of the packages we use have vignettes that have been created for them. Vignettes are designed to show how functions are used. Identify either a function related to your concept or your selected verb (which is a function), and find what package it is in. Then look for a package vignette. What package did you look for a vignette for? Is your concept or verb illustrated in the vignette?

(Searching with Google or within R are possible.)

Solution:

> part d - Many people blog examples of different R functions. Search for an R example of your concept or verb using Google. Look over the search results and identify one that demonstrates correct use of the concept or verb. List the URL.

Solution:

2 - MDSR 5.2

Use the `Batting`, `Pitching`, and `Master` tables in the *Lahman* package to answer the following questions. Remember that you are responsible for loading packages in the setup chunk.

> part a - List the name of every player in baseball history who has accumulated at least 300 home runs (HR) and at least 300 stolen bases (SB). You can find the first and last name of the player in the `Master` data frame. Join this to your result along with the total home runs and total bases stolen for each of these elite players.

Solution:

```
head(Batting)
```

```
##    playerID yearID stint teamID lgID  G  AB  R  H X2B X3B HR RBI SB CS BB SO
## 1 abercda01   1871     1    TRO   NA  1   4  0  0   0   0  0   0  0  0  0  0
## 2  addybo01   1871     1    RC1   NA 25 118 30 32   6   0  0  13  8  1  4  0
## 3 allisar01   1871     1    CL1   NA 29 137 28 40   4   5  0  19  3  1  2  5
## 4 allisdo01   1871     1    WS3   NA 27 133 28 44  10   2  2  27  1  1  0  2
## 5 ansonca01   1871     1    RC1   NA 25 120 29 39  11   3  0  16  6  2  2  1
## 6 armstbo01   1871     1    FW1   NA 12  49  9 11   2   1  0   5  0  1  0  1
##   IBB HBP SH SF GIDP
## 1  NA  NA NA NA    0
## 2  NA  NA NA NA    0
## 3  NA  NA NA NA    1
## 4  NA  NA NA NA    0
## 5  NA  NA NA NA    0
## 6  NA  NA NA NA    0
```

```
top300 <- Batting %>%
  left_join(Master, by = c("playerID")) %>%
  select(nameFirst, playerID, nameLast, HR, SB) %>%
  group_by(playerID, nameFirst, nameLast) %>%
  summarize(HR = sum(HR), SB = sum(SB)) %>%
  filter(HR > 300 & SB > 300) %>%
  select(-playerID) %>%
  rename("First Name" = nameFirst,
         "Last Name" = nameLast,
         "Home Runs" = HR,
         "Stolen Bases" = SB) %>%
  kable(booktabs = TRUE)
```

```
## `summarise()` has grouped output by 'playerID', 'nameFirst'. You can override using the `.groups` ar
## Adding missing grouping variables: `playerID`
```

```
top300
```

playerID	First Name	Last Name	Home Runs	Stolen Bases
beltrca01	Carlos	Beltran	435	312
bondsba01	Barry	Bonds	762	514
bondsbo01	Bobby	Bonds	332	461
dawsoan01	Andre	Dawson	438	314
finlest01	Steve	Finley	304	320
mayswi01	Willie	Mays	660	338
rodrial01	Alex	Rodriguez	696	329
sandere02	Reggie	Sanders	305	304

part b - Similarly, list the names every pitcher in baseball history who has accumulated at least 300 wins (W) and at least 3,000 strikeouts (SO).

Solution:

```
head(Batting)
```

```
##    playerID yearID stint teamID lgID  G  AB  R  H X2B X3B HR RBI SB CS BB SO
## 1 abercda01   1871     1    TRO   NA  1   4  0  0   0   0  0   0  0  0  0  0
## 2  addybo01   1871     1    RC1   NA 25 118 30 32   6   0  0  13  8  1  4  0
## 3 allisar01   1871     1    CL1   NA 29 137 28 40   4   5  0  19  3  1  2  5
## 4 allisdo01   1871     1    WS3   NA 27 133 28 44  10   2  2  27  1  1  0  2
## 5 ansonca01   1871     1    RC1   NA 25 120 29 39  11   3  0  16  6  2  2  1
## 6 armstbo01   1871     1    FW1   NA 12  49  9 11   2   1  0   5  0  1  0  1
##   IBB HBP SH SF GIDP
## 1  NA  NA NA NA    0
## 2  NA  NA NA NA    0
## 3  NA  NA NA NA    1
## 4  NA  NA NA NA    0
## 5  NA  NA NA NA    0
## 6  NA  NA NA NA    0
```

```
top300b <- Pitching %>%
  left_join(Master, by = c("playerID")) %>%
  select(nameFirst, playerID, nameLast, SO, W) %>%
  group_by(playerID, nameFirst, nameLast) %>%
  summarize(SO = sum(SO), W = sum(W)) %>%
  filter(SO > 3000 & W > 300)
```

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

```
head(top300b, 20)
```

```
## # A tibble: 10 x 5
## # Groups:   playerID, nameFirst [10]
##    playerID  nameFirst nameLast    SO     W
##    <chr>     <chr>     <chr>    <int> <int>
##  1 carltst01 Steve     Carlton   4136   329
##  2 clemero02 Roger     Clemens   4672   354
```

```
##  3 johnsra05 Randy     Johnson   4875   303
##  4 johnswa01 Walter    Johnson   3509   417
##  5 maddugr01 Greg      Maddux    3371   355
##  6 niekrph01 Phil      Niekro    3342   318
##  7 perryga01 Gaylord   Perry     3534   314
##  8 ryanno01  Nolan     Ryan      5714   324
##  9 seaveto01 Tom       Seaver    3640   311
## 10 suttodo01 Don       Sutton    3574   324
```

```
head(top300b, 30)
```

```
## # A tibble: 10 x 5
## # Groups:   playerID, nameFirst [10]
##    playerID  nameFirst nameLast    SO     W
##    <chr>     <chr>     <chr>    <int> <int>
##  1 carltst01 Steve     Carlton   4136   329
##  2 clemero02 Roger     Clemens   4672   354
##  3 johnsra05 Randy     Johnson   4875   303
##  4 johnswa01 Walter    Johnson   3509   417
##  5 maddugr01 Greg      Maddux    3371   355
##  6 niekrph01 Phil      Niekro    3342   318
##  7 perryga01 Gaylord   Perry     3534   314
##  8 ryanno01  Nolan     Ryan      5714   324
##  9 seaveto01 Tom       Seaver    3640   311
## 10 suttodo01 Don       Sutton    3574   324
```

> part c - Finally, list the name and year of every player who has hit at least 50 home runs in a single season. Which player had the lowest batting average in that season?

Note: Batting average is calculated as the number of hits (H) divided by the number of at bats (AB).

Solution: Too many observations, and what do I do with duplicate names?

```
homeruns <- Batting %>%
  left_join(Master, by = c("playerID")) %>%
  select(nameFirst, yearID, nameLast, playerID, HR, H, AB) %>%
  filter(HR > 50) %>%
  group_by(yearID, nameFirst, nameLast) %>%
  summarize(battingaverage = H/AB) %>%
  arrange(battingaverage)
```

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

```
head(homeruns, 20)
```

```
## # A tibble: 20 x 4
## # Groups:   yearID, nameFirst [20]
##    yearID nameFirst nameLast  battingaverage
##     <int> <chr>     <chr>              <dbl>
##  1   2019 Pete      Alonso             0.260
##  2   2010 Jose      Bautista           0.260
```

```
##  3   2005 Andruw     Jones             0.263
##  4   1961 Roger      Maris             0.269
##  5   1990 Cecil      Fielder           0.277
##  6   1999 Mark       McGwire           0.278
##  7   2017 Giancarlo  Stanton           0.281
##  8   2017 Aaron      Judge             0.284
##  9   1998 Ken        Griffey           0.284
## 10   2013 Chris      Davis             0.286
## 11   2006 David      Ortiz             0.287
## 12   1999 Sammy      Sosa              0.288
## 13   1998 Mark       McGwire           0.299
## 14   2002 Alex       Rodriguez         0.300
## 15   1947 Johnny     Mize              0.302
## 16   2002 Jim        Thome             0.304
## 17   1997 Ken        Griffey           0.304
## 18   1998 Sammy      Sosa              0.308
## 19   1949 Ralph      Kiner             0.310
## 20   1996 Mark       McGwire           0.312
```

3 - MDSR 4.11 (modified)

The `Violations` data set in the **mdsr** package contains information regarding the outcome of health inspections of restaurants in New York City. Note that higher inspection scores indicate worse violations: "restaurants with an inspection score between 0 and 13 points earn an A, those with 14 to 27 points receive a B and those with 28 or more a C" (nyc.gov).

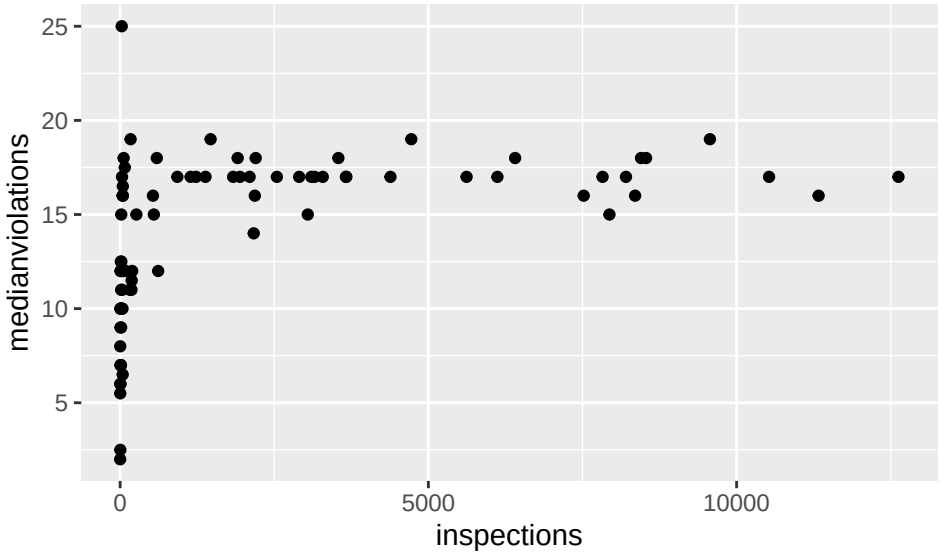
> part a - Use these data to calculate the median violation score by zip code for zip codes in Manhattan. What pattern, if any, do you see between the number of inspections and the median score? Generate a visualization to support your response.

Solution:

```
median_violation <- Violations %>%
  select(boro, zipcode, score) %>%
  filter(boro == "MANHATTAN") %>%
  na.omit() %>%
  group_by(zipcode) %>%
  summarize(medianviolations = median(score),
            inspections = n())

head(median_violation)
```

```
## # A tibble: 6 x 3
##   zipcode medianviolations inspections
##     <int>            <dbl>       <int>
## 1   10001               15        7937
## 2   10002               18        8449
## 3   10003               17       12625
## 4   10004               14        2167
## 5   10005               17        1144
## 6   10006               17         928
```

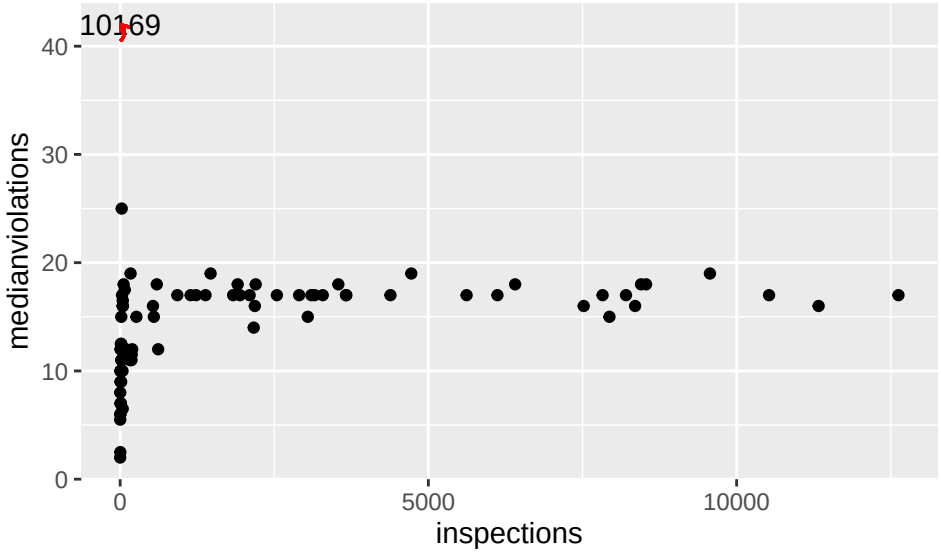
```
p <- ggplot(data = median_violation, aes(x = inspections, y =medianviolations)) +
  geom_point()
p
```



part b - In your visualization above, there are several potential outliers but there is one zipcode in particular that does not seem to fall along the general trend. Add text to the outlier identifying what zipcode it is, and add an arrow pointing from the text to the observation. Note: first, you may want to `filter()` to identify the zipcode (so you know what text to add to the plot).

Solution:

```
p <- ggplot(data = median_violation, aes(x = inspections, y =medianviolations)) +
  geom_point() +
  annotate("text", x = 5, y = 42, label = "10169") +
  geom_curve(aes(x = 5.3, y = 40.5, xend = 6.5, yend = 42), arrow = arrow(length = unit(.02, "npc")), c
p
```



4 - MDSR 6.5

Generate the code to convert the data frame from the starting point to the results.

Figures available in text online in Section 6.6.

The starting data frame is provided. Hint (from text): Use *pivot_longer()* in conjunction with *pivot_wider()*.

```
OrigData <- data.frame(grp = c("A","A","B", "B")
                       , sex = c("F", "M", "F", "M")
                       , meanL = c(0.22, 0.47, 0.33, 0.55)
                       , sdL = c(0.11, 0.33, 0.11, 0.31)
                       , meanR = c(0.34, 0.57, 0.40, 0.65)
                       , sdR = c(0.08, 0.33, 0.07, 0.27))
```

Solution:

```
glimpse(OrigData)
```

```
## Rows: 4
## Columns: 6
## $ grp   <chr> "A", "A", "B", "B"
## $ sex   <chr> "F", "M", "F", "M"
## $ meanL <dbl> 0.22, 0.47, 0.33, 0.55
## $ sdL   <dbl> 0.11, 0.33, 0.11, 0.31
## $ meanR <dbl> 0.34, 0.57, 0.40, 0.65
## $ sdR   <dbl> 0.08, 0.33, 0.07, 0.27
```

```
newData <- OrigData %>%
pivot_longer(cols = meanL:sdR, names_to = "class", values_to = "values")  %>%
pivot_wider(
  names_from = c(class, sex),
  names_glue = ("{sex}.{class}"),
  values_from = c(values))
head(newData)
```

```
## # A tibble: 2 x 9
##   grp   F.meanL F.sdL F.meanR F.sdR M.meanL M.sdL M.meanR M.sdR
##   <chr>   <dbl> <dbl>   <dbl> <dbl>   <dbl> <dbl>   <dbl> <dbl>
## 1 A        0.22  0.11    0.34  0.08    0.47  0.33    0.57  0.33
## 2 B        0.33  0.11    0.4   0.07    0.55  0.31    0.65  0.27
```

5 - Combining your Wrangling and Visualization Skills

When we looked at our first UN votes visual, some wrangling was required to get the data into a format appropriate for the visual. Now that we've examined both visualization and wrangling, you can combine the skills too! (And you did a little of this above).

We will be looking at a data set on high school students in Portugal. We have information on their performance in a Math course and a Portugeuse course (think of this as your natural language course, i.e. English for English speakers, etc.), as well as a host of demographic variables. Detailed information about the data set is provided on the following pages - you should look it over as you tackle this problem. (Feel free to remove the info when knitting to the final version of your assignment.)

We want to visualize the relationship between final Math and final Portugeuse grade for students who were in both courses. In addition, we want to be sure all students in the visual were under 20 years old, and had fewer than 10 absences in either course (not total). We also want to factor in weekend alcohol use and travel time as reported in the Math data set in our examination of the relationship, treating these as appropriate group variables (categorical). (Students filled out the survey twice and not all responses match between them, even for the same student.)

1. Wrangle the data you need into an appropriate format, and save it as a new data set with the variables you need for your visual.

Solution:

```
oldgrades <- math_data %>%
  inner_join(port_data, c("school", "sex", "age", "address", "famsize","Pstatus", "Medu","Fedu","Mjob",
glimpse(oldgrades)
```

```
## Rows: 382
## Columns: 53
## $ school       <chr> "GP", "GP", "GP", "GP", "GP", "GP", "GP", "GP", "GP", "GP~
## $ sex          <chr> "F", "F", "F", "F", "F", "M", "M", "F", "M", "M", "F", "F~
## $ age          <int> 18, 17, 15, 15, 16, 16, 16, 17, 15, 15, 15, 15, 15, 15, 1~
## $ address      <chr> "U", "U", "U", "U", "U", "U", "U", "U", "U", "U", "U", "U~
## $ famsize      <chr> "GT3", "GT3", "LE3", "GT3", "GT3", "LE3", "LE3", "GT3", "~
## $ Pstatus      <chr> "A", "T", "T", "T", "T", "T", "T", "A", "A", "T", "T", "T~
## $ Medu         <int> 4, 1, 1, 4, 3, 4, 2, 4, 3, 3, 4, 2, 4, 4, 2, 4, 4, 3, 3, ~
## $ Fedu         <int> 4, 1, 1, 2, 3, 3, 2, 4, 2, 4, 4, 1, 4, 3, 2, 4, 4, 3, 2, ~
## $ Mjob         <chr> "at_home", "at_home", "at_home", "health", "other", "serv~
## $ Fjob         <chr> "teacher", "other", "other", "services", "other", "other"~
## $ reason       <chr> "course", "course", "other", "home", "home", "reputation"~
## $ guardian.x   <chr> "mother", "father", "mother", "mother", "father", "mother~
## $ traveltime.x <int> 2, 1, 1, 1, 1, 1, 1, 2, 1, 1, 1, 3, 1, 2, 1, 1, 1, 3, 1, ~
## $ studytime.x  <int> 2, 2, 2, 3, 2, 2, 2, 2, 2, 2, 2, 3, 1, 2, 3, 1, 3, 2, 1, ~
## $ failures.x   <int> 0, 0, 3, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 3, ~
## $ schoolsup.x  <chr> "yes", "no", "yes", "no", "no", "no", "no", "yes", "no", ~
## $ famsup.x     <chr> "no", "yes", "no", "yes", "yes", "yes", "no", "yes", "yes~
## $ paid.x       <chr> "no", "no", "yes", "yes", "yes", "yes", "no", "no", "yes"~
## $ activities.x <chr> "no", "no", "no", "yes", "no", "yes", "no", "no", "no", "~
## $ nursery      <chr> "yes", "no", "yes", "yes", "yes", "yes", "yes", "yes", "y~
## $ higher.x     <chr> "yes", "yes", "yes", "yes", "yes", "yes", "yes", "yes", "~
## $ internet     <chr> "no", "yes", "yes", "yes", "no", "yes", "yes", "no", "yes~
## $ romantic.x   <chr> "no", "no", "no", "yes", "no", "no", "no", "no", "no", "n~
## $ famrel.x     <int> 4, 5, 4, 3, 4, 5, 4, 4, 4, 5, 3, 5, 4, 5, 4, 4, 3, 5, 5, ~
```

```
## $ freetime.x   <int> 3, 3, 3, 2, 3, 4, 4, 1, 2, 5, 3, 2, 3, 4, 5, 4, 2, 3, 5, ~
## $ goout.x      <int> 4, 3, 2, 2, 2, 2, 4, 4, 2, 1, 3, 2, 3, 3, 2, 4, 3, 2, 5, ~
## $ Dalc.x       <int> 1, 1, 2, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 2, ~
## $ Walc.x       <int> 1, 1, 3, 1, 2, 2, 1, 1, 1, 1, 2, 1, 3, 2, 1, 2, 2, 1, 4, ~
## $ health.x     <int> 3, 3, 3, 5, 5, 5, 3, 1, 1, 5, 2, 4, 5, 3, 3, 2, 2, 4, 5, ~
## $ absences.x   <int> 6, 4, 10, 2, 4, 10, 0, 6, 0, 0, 0, 4, 2, 2, 0, 4, 6, 4, 1~
## $ G1.x         <int> 5, 5, 7, 15, 6, 15, 12, 6, 16, 14, 10, 10, 14, 10, 14, 14~
## $ G2.x         <int> 6, 5, 8, 14, 10, 15, 12, 5, 18, 15, 8, 12, 14, 10, 16, 14~
## $ G3.x         <int> 6, 6, 10, 15, 10, 15, 11, 6, 19, 15, 9, 12, 14, 11, 16, 1~
## $ guardian.y   <chr> "mother", "father", "mother", "mother", "father", "mother~
## $ traveltime.y <int> 2, 1, 1, 1, 1, 1, 1, 2, 1, 1, 1, 3, 1, 2, 1, 1, 1, 3, 1, ~
## $ studytime.y  <int> 2, 2, 2, 3, 2, 2, 2, 2, 2, 2, 2, 3, 1, 2, 3, 1, 3, 2, 1, ~
## $ failures.y   <int> 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 3, ~
## $ schoolsup.y  <chr> "yes", "no", "yes", "no", "no", "no", "no", "yes", "no", ~
## $ famsup.y     <chr> "no", "yes", "no", "yes", "yes", "yes", "no", "yes", "yes~
## $ paid.y       <chr> "no", "no", "no", "no", "no", "no", "no", "no", "no", "no~
## $ activities.y <chr> "no", "no", "no", "yes", "no", "yes", "no", "no", "no", "~
## $ higher.y     <chr> "yes", "yes", "yes", "yes", "yes", "yes", "yes", "yes", "~
## $ romantic.y   <chr> "no", "no", "no", "yes", "no", "no", "no", "no", "no", "n~
## $ famrel.y     <int> 4, 5, 4, 3, 4, 5, 4, 4, 4, 5, 3, 5, 4, 5, 4, 4, 3, 5, 5, ~
## $ freetime.y   <int> 3, 3, 3, 2, 3, 4, 4, 1, 2, 5, 3, 2, 3, 4, 5, 4, 2, 3, 5, ~
## $ goout.y      <int> 4, 3, 2, 2, 2, 2, 4, 4, 2, 1, 3, 2, 3, 3, 2, 4, 3, 2, 5, ~
## $ Dalc.y       <int> 1, 1, 2, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 2, ~
## $ Walc.y       <int> 1, 1, 3, 1, 2, 2, 1, 1, 1, 1, 2, 1, 3, 2, 1, 2, 2, 1, 4, ~
## $ health.y     <int> 3, 3, 3, 5, 5, 5, 3, 1, 1, 5, 2, 4, 5, 3, 3, 2, 2, 4, 5, ~
## $ absences.y   <int> 4, 2, 6, 0, 0, 6, 0, 2, 0, 0, 2, 0, 0, 0, 0, 6, 10, 2, 2,~
## $ G1.y         <int> 0, 9, 12, 14, 11, 12, 13, 10, 15, 12, 14, 10, 12, 12, 14,~
## $ G2.y         <int> 11, 11, 13, 14, 13, 12, 12, 13, 16, 12, 14, 12, 13, 12, 1~
## $ G3.y         <int> 11, 11, 12, 14, 13, 13, 13, 13, 17, 13, 14, 13, 12, 13, 1~
```

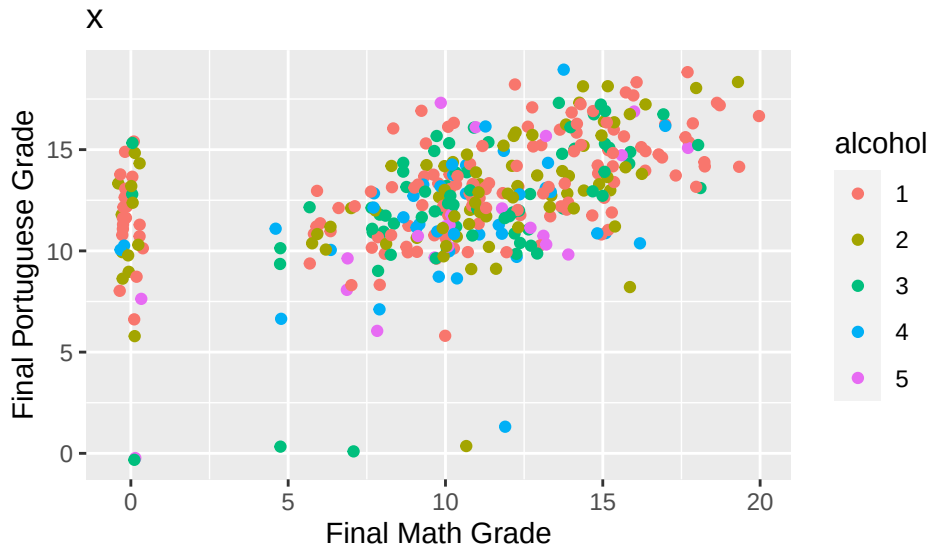
```
newgrades <- oldgrades %>%
  filter(age < 20) %>%
  filter(absences.x < 10 | absences.y < 10) %>%
  select(G3.x, G3.y, traveltime.x, Walc.x) %>%
  mutate(alcohol = as.factor(Walc.x))

glimpse(newgrades)
```

```
## Rows: 356
## Columns: 5
## $ G3.x         <int> 6, 6, 10, 15, 10, 15, 11, 6, 19, 15, 9, 12, 14, 11, 16, 1~
## $ G3.y         <int> 11, 11, 12, 14, 13, 13, 13, 13, 17, 13, 14, 13, 12, 13, 1~
## $ traveltime.x <int> 2, 1, 1, 1, 1, 1, 1, 2, 1, 1, 1, 3, 1, 2, 1, 1, 1, 3, 1, ~
## $ Walc.x       <int> 1, 1, 3, 1, 2, 2, 1, 1, 1, 1, 2, 1, 3, 2, 1, 2, 2, 1, 4, ~
## $ alcohol      <fct> 1, 1, 3, 1, 2, 2, 1, 1, 1, 1, 2, 1, 3, 2, 1, 2, 2, 1, 4, ~
```

2. Then generate an appropriate visual. Make sure your graphic has appropriate labels, legends (as needed), and a title.

Solution:

```
p <- ggplot(data = newgrades, aes(x = G3.x, y = G3.y, color = alcohol)) +
  geom_jitter() +
  labs(title = "x", x = "Final Math Grade", y = "Final Portuguese Grade")
p
```



3. Finally, in a few sentences, describe what you find.

Solution:

Data Set Information for Problem 5

The data set is from a paper called "Using Data Mining To Predict Secondary School Student Alcohol Consumption" by Fabio Pagnotta and Hossain Mohammad Amran of the Department of Computer Science, University of Camerino, and the data set is hosted online in UCI's machine learning repository.

The information below was copied from the provided codebook online.

Attributes for both student-mat.csv (Math course) and student-por.csv (Portuguese language course) datasets:
1. school - student's school (binary: 'GP' - Gabriel Pereira or 'MS' - Mousinho da Silveira)
2. sex - student's sex (binary: 'F' - female or 'M' - male)
3. age - student's age (numeric: from 15 to 22)
4. address - student's home address type (binary: 'U' - urban or 'R' - rural)
5. famsize - family size (binary: 'LE3' - less or equal to 3 or 'GT3' - greater than 3)
6. Pstatus - parent's cohabitation status (binary: 'T' - living together or 'A' - apart)
7. Medu - mother's education (numeric: 0 - none, 1 - primary education (4th grade), 2 - 5th to 9th grade, 3 - secondary education or 4 - higher education)
8. Fedu - father's education (numeric: 0 - none, 1 - primary education (4th grade), 2 - 5th to 9th grade, 3 - secondary education or 4 - higher education)
9. Mjob - mother's job (nominal: 'teacher', 'health' care related, civil 'services' (e.g. administrative or police), 'at_home' or 'other')
10. Fjob - father's job (nominal: 'teacher', 'health' care related, civil 'services' (e.g. administrative or police), 'at_home' or 'other')
11. reason - reason to choose this school (nominal: close to 'home', school 'reputation', 'course' preference or 'other')
12. guardian - student's guardian (nominal: 'mother', 'father' or 'other')
13. traveltime - home to school travel time (numeric: 1 - <15 min., 2 - 15 to 30 min., 3 - 30 min. to 1 hour, or 4 - >1 hour)
14. studytime - weekly study time (numeric: 1 - <2 hours, 2 - 2 to 5 hours, 3 - 5 to 10 hours, or 4 - >10 hours)
15. failures - number of past class failures (numeric: n if 1<=n<3, else 4)
16. schoolsup - extra educational support (binary: yes or no)
17. famsup - family educational support (binary: yes or no)
18. paid - extra paid classes within the course subject (Math or Portuguese) (binary: yes or no)
19. activities - extra-curricular activities (binary: yes or no)
20. nursery - attended nursery school (binary: yes or no)
21. higher - wants to take higher education (binary: yes or no)
22. internet - Internet access at home (binary: yes or no)
23. romantic - with a romantic relationship (binary: yes or no)
24. famrel - quality of family relationships (numeric: from 1 - very bad to 5 - excellent)
25. freetime - free time after school (numeric: from 1 - very low to 5 - very high)
26. goout - going out with friends (numeric: from 1 - very low to 5 - very high)
27. Dalc - workday alcohol consumption (numeric: from 1 - very low to 5 - very high)
28. Walc - weekend alcohol consumption (numeric: from 1 - very low to 5 - very high)
29. health - current health status (numeric: from 1 - very bad to 5 - very good)
30. absences - number of school absences (numeric: from 0 to 93)

Finally, the grades are related with the course subject, Math or Portuguese:
31. G1 - first period grade (numeric: from 0 to 20)
32. G2 - second period grade (numeric: from 0 to 20)
33. G3 - final grade (numeric: from 0 to 20, output target)
Thus, these variables appear in each data set, but have different meaning in each.

The data was provided as two different .csv files online. I obtained some errors trying to work with them, so ended up saving them as .txt files on my website. Many of the students were in both courses, but not all.