HW2

Jadon Fowler jaf582 5778191

2022-10-03

Instructions

Download a copy of this markdown. Change the `author:` tag above to have your name and NAU's ID. Fill in the file with your solution to the proposed problems. Knit your final document to PDF and submit it through BBLearn in the assignment `[HW] Homework 2: data visualization` by the end of the day on **Thursday, October 6 (11:59 PM)**.

Note: to knit your homework to PDF, you need to have MikTex installed. You can download and install MikTex from here. If you don't want to install MikTex, you can knit to HTML, open the file in the browser and print the page to PDF.

For all the problems, please import the useful libraries in this chunk of code:

Problem 1: the dataset

Download the `titanic.csv` file attached to this homework on BBLearn. Import the data into a dataframe/tibble. Show the head and the tail of your new dataset.

```
#import the dataset here
titanic <- as.data.frame(read_csv("titanic.csv"))
```

```
## Rows: 891 Columns: 11
```

```
## -- Column specification --------------------------------------------------------
## Delimiter: ","
## chr (4): Name, Sex, Cabin, Embarked
## dbl (7): Passenger Id, Survived, Pclass, Age, Siblings Spouse, Parents child...
```

```
##
## i Use `spec()` to retrieve the full column specification for this data.
## i Specify the column types or set `show_col_types = FALSE` to quiet this message.
```

```
head(titanic)
```

```
##   Passenger Id Survived Pclass
## 1            1        0      3
## 2            2        1      1
## 3            3        1      3
## 4            4        1      1
## 5            5        0      3
```

```
## 6            6        0      3
##                                                  Name    Sex Age
## 1                             Braund, Mr. Owen Harris   <NA>  22
## 2 Cumings, Mrs. John Bradley (Florence Briggs Thayer)   <NA>  38
## 3                              Heikkinen, Miss. Laina   <NA>  26
## 4        Futrelle, Mrs. Jacques Heath (Lily May Peel) female  35
## 5                            Allen, Mr. William Henry   male  35
## 6                                    Moran, Mr. James   male  NA
##   Siblings Spouse Parents children    Fare Cabin Embarked
## 1               1                0  7.2500  <NA>        S
## 2               1                0 71.2833   C85        C
## 3               0                0  7.9250  <NA>        S
## 4               1                0 53.1000  C123        S
## 5               0                0  8.0500  <NA>        S
## 6               0                0  8.4583  <NA>        Q
```

Inspect the dataframe. How many variables and observations it has? Confirm your inference using R code that shows this information.

There are 11 columns of data for each passenger, and there are 891 passengers

```
#write the R code to confirm your answer here
ncol(titanic)
```

```
## [1] 11
```

```
nrow(titanic)
```

```
## [1] 891
```

Inspect the variables. Check the variable names and types to see if adjustments are needed. Change the variable's names if they are not standardized or are inconvenient to work with. Change the variable's types if they do not comply with the data in the column. You must decide whether a value should be treated as factors, ordinal factors or numerical/continuous values. Write an explanation that justify your choices and show the R code you used to make and check the changes.

All spaces and underscores were changed to dots, and all letters were made lowercase so the column names are standardized. Sex, survived status, pclass, and embarked status were changed to factors because there is a finite choice for each data type.

```
titanic <- rename_with(titanic, function(x) tolower(str_replace_all(str_replace_all(x, " ", "."), "_",
titanic$sex <- as.factor(titanic$sex)
titanic$survived <- as.factor(titanic$survived)
titanic$embarked <- as.factor(titanic$embarked)
titanic$pclass <- as.factor(titanic$pclass)
head(titanic)
```

```
##   passenger.id survived pclass
## 1            1        0      3
## 2            2        1      1
## 3            3        1      3
## 4            4        1      1
## 5            5        0      3
```

```
## 6            6        0      3
##                                                    name    sex age
## 1                             Braund, Mr. Owen Harris   <NA>  22
## 2 Cumings, Mrs. John Bradley (Florence Briggs Thayer)   <NA>  38
## 3                               Heikkinen, Miss. Laina   <NA>  26
## 4        Futrelle, Mrs. Jacques Heath (Lily May Peel) female  35
## 5                            Allen, Mr. William Henry   male  35
## 6                                    Moran, Mr. James   male  NA
##   siblings.spouse parents.children    fare cabin embarked
## 1               1                0  7.2500  <NA>        S
## 2               1                0 71.2833   C85        C
## 3               0                0  7.9250  <NA>        S
## 4               1                0 53.1000  C123        S
## 5               0                0  8.0500  <NA>        S
## 6               0                0  8.4583  <NA>        Q
```

Problem 2: fixing the sex's missing values using titles

The column sex has a few missing values. Although we did not learn how to deal with missing values (or why doing it is important), I can tell you that one of the techniques used for that purpose is to infer the missing value from other columns. In our dataset, the `Name` variable has a title along with the passenger's name. Because titles are frequently gendered, our best guest on the value of the `sex` is an association with the person's title. Thus, do the following:

1. Using regular expressions (and perhaps some other cool functions!), find a way to extract the unique titles from the column `Name`. The outcome can be either a vector or a data.frame/tibble with the titles.

```
titles <- str_match(titanic$name, "M(s|r|rs|iss|aster|ajor)")[,1]
```

2. Decide, by manual inspection, to which gender each of the titles are associate with. For example, we can assume that the title `Mr.` is associated with a `male` value while `Mrs.` is associated with a `female` value for the `sex` variable. Then, create two regex patterns: one that matches any of the female titles, and one that matches any of the male titles. Ignore the titles that do not define gender. Demonstrate that the patterns are detecting the correct titles.

```
male <- "M(r|aster|ajor)"
female <- "M(s|rs|iss)"
is.male <- !is.na(str_match(titles, male)[,1])
is.female <- !is.na(str_match(titles, female)[,1])
```

3. Write an R code that replaces the missing values in the `sex` column according to the gender associated with the person's title:

If the `Sex` variable is NA and the `Name` variable matches a pattern with the female patterns, then the variable `Sex` receives `"female"` If the `Sex` variable is NA and the `Name` variable matches a pattern with the male patterns, then the variable `Sex` receives `"male"`

```
titanic$sex[is.na(titanic$sex) & is.male] <- "male"
titanic$sex[is.na(titanic$sex) & is.female] <- "female"
```

Problem 3: plotting some data

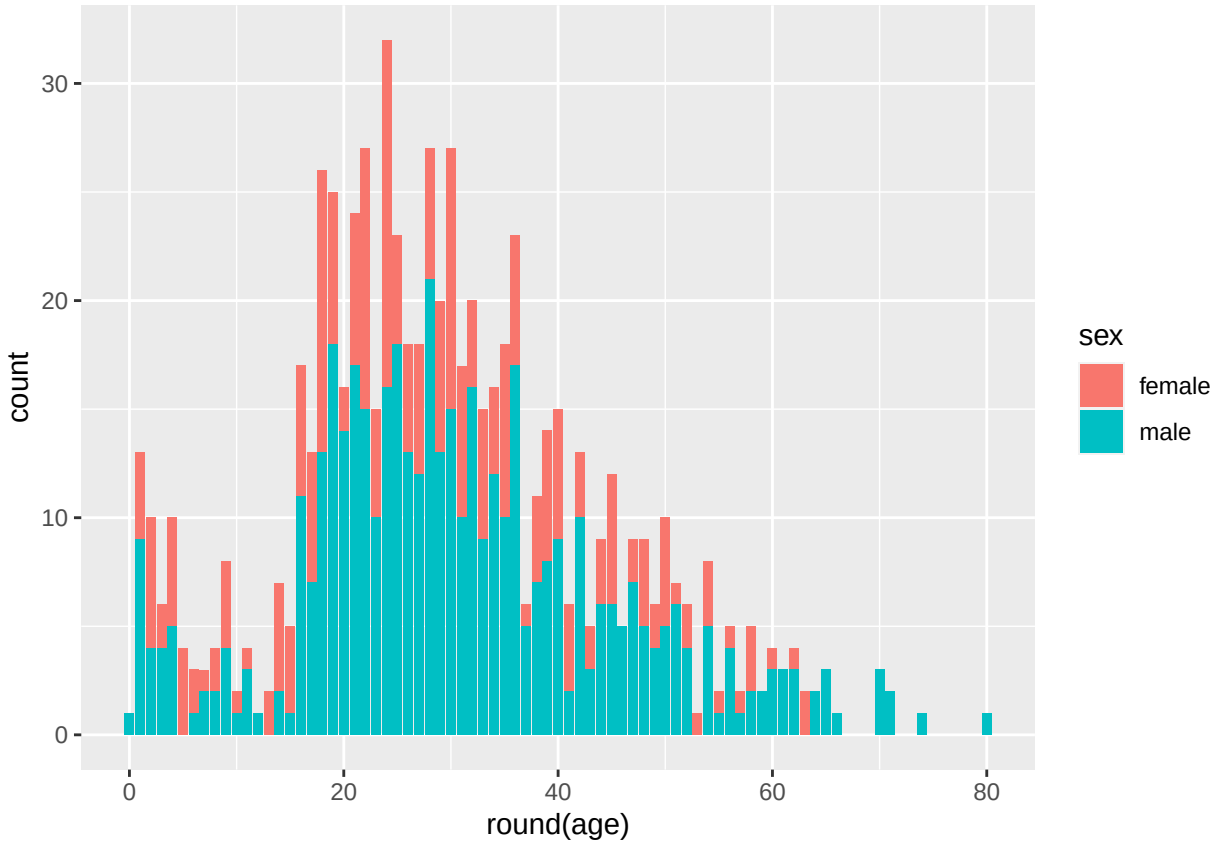
1. Create a plot to represent the passenger's age. You must decide what `geom` representation you'll use. Justify your choice and explain what you can observe about the data from the visualization.

I chose a bar plot that plots the rounded age of everyone and colors based on sex. I rounded it so the graph is more readable. Knowing that certain passengers are 0.5 the way through a year isn't useful.

The graph shows that most people are between 20 and 40 years old.

```
ggplot(titanic) +
  geom_bar(aes(x = round(age), fill = sex))
```

```
## Warning: Removed 177 rows containing non-finite values (stat_count).
```



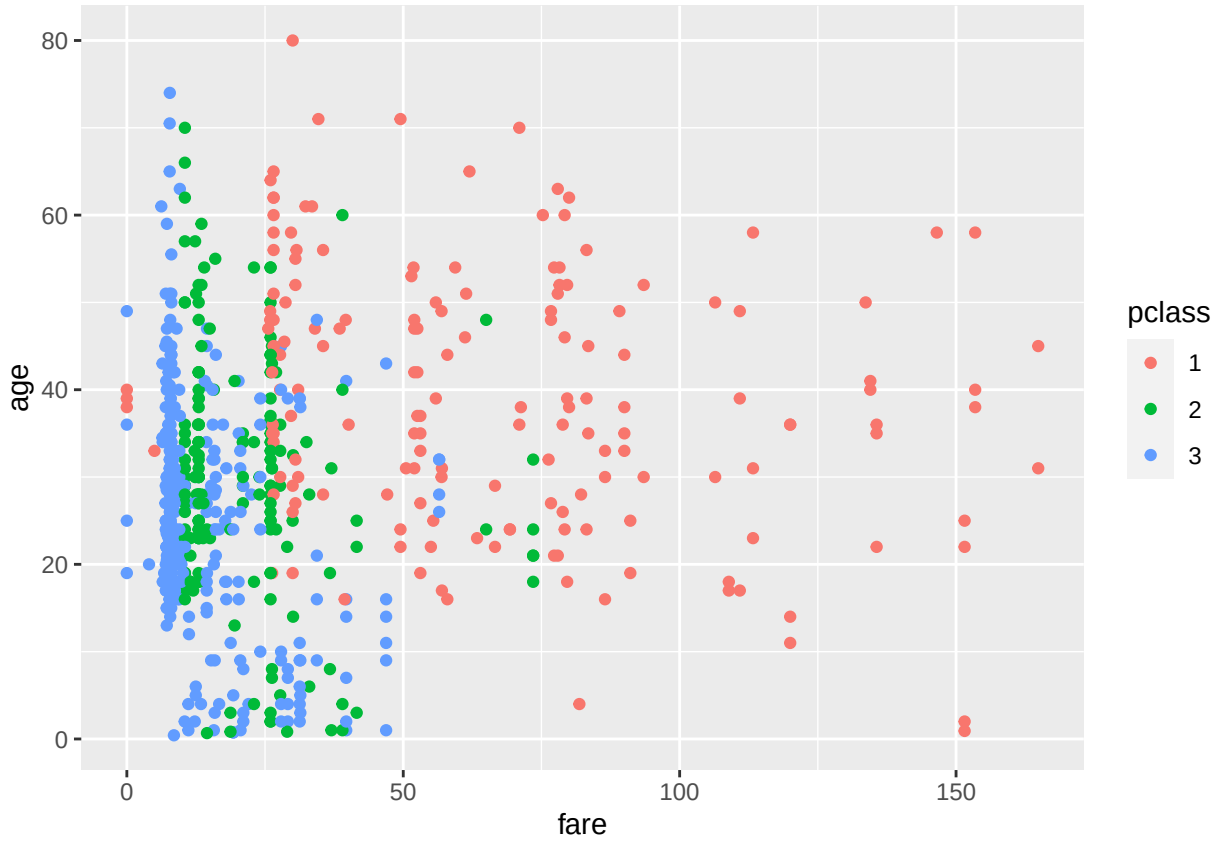
write your answer here

2. Create a plot to represent the relationship between the fare and the passenger's age. You must decide what `geom` representation you'll use. Justify your choice and explain what you can observe about the data from the visualization. Considering that one goal of the data analysis is to understand the wealthy of passengers, is age a possible relevant variable? Justify your answer.

I chose a point plot because the ranges are large for both variables. I also colored the points by pclass to identify what ticket prices were in which pclass. It looks like only people above 23 years old were buying the ticket for $25.

```
ggplot(filter(titanic, titanic$fare < 200)) +
#ggplot(titanic) +
  geom_point(aes(y = age, x = fare, color = pclass))
```

```
## Warning: Removed 175 rows containing missing values (geom_point).
```



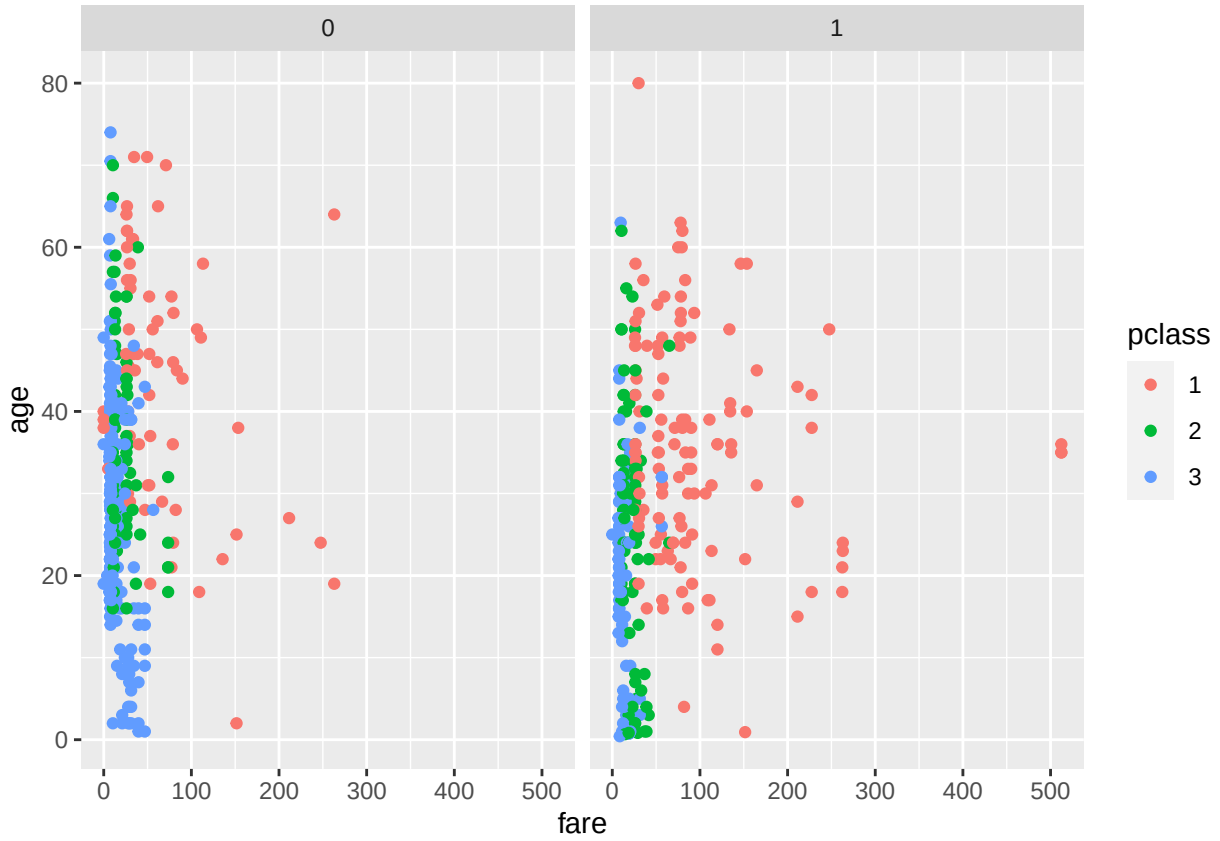
write your answer here

3. Create a plot to represent the distribution of the fare, separated by whether the passenger survived. You must decide what `geom` representation you'll use. Justify your choice and explain what you can observe about the data from the visualization. Can we infer whether the fare influenced passenger's chance to survive?

I split this into two point plots: one for the survivors and one for those that didn't survive. This makes it simple to see both groups. It looks like the survivors had higher fares.

```
ggplot(titanic) +
  geom_point(aes(y = age, x = fare, color = pclass)) +
  facet_wrap(. ~ survived)
```

```
## Warning: Removed 177 rows containing missing values (geom_point).
```

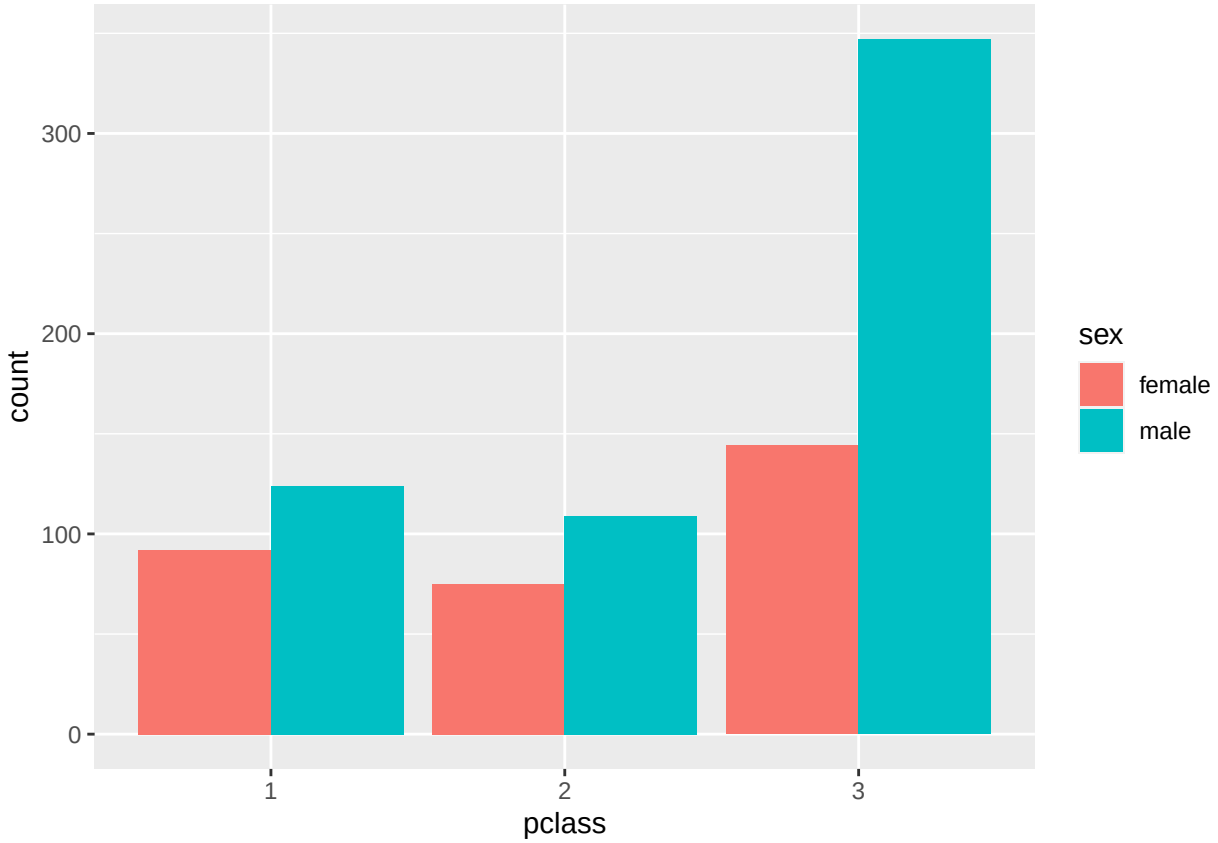


write your answer here

4. Create a plot to represent the relationship between the sex and the class in which the passenger's traveled. You must decide what `geom` representation you'll use. Justify your choice and explain what you can observe about the data from the visualization.

I used a dodging bar plot that shows male and female counts for each pclass. There were more males than females in every class.

```
ggplot(titanic) +
  geom_bar(position="dodge", aes(x = pclass, fill = sex))
```



write your answer here

5. Create a subset of the titanic dataset that: removes the column `Cabin` (because it has way too many `NA` values!), filter out passengers who were traveling alone (no siblings/spouse nor parents/children). Then, export this subset in a new CSV file.

```
titanic.alone <- filter(titanic[(! titanic %in% c('Cabin'))], titanic$siblings.spouse == 0 & titanic$pa
write_csv(titanic.alone, "titanic_alone.csv")
head(titanic.alone)
```

```
##   passenger.id survived pclass                          name    sex age
## 1            3        1      3         Heikkinen, Miss. Laina female  26
## 2            5        0      3       Allen, Mr. William Henry   male  35
## 3            6        0      3               Moran, Mr. James   male  NA
## 4            7        0      1        McCarthy, Mr. Timothy J   male  54
## 5           12        1      1       Bonnell, Miss. Elizabeth female  58
## 6           13        0      3 Saundercock, Mr. William Henry   male  20
##   siblings.spouse parents.children    fare cabin embarked
## 1               0                0  7.9250  <NA>        S
## 2               0                0  8.0500  <NA>        S
## 3               0                0  8.4583  <NA>        Q
## 4               0                0 51.8625   E46        S
## 5               0                0 26.5500  C103        S
## 6               0                0  8.0500  <NA>        S
```

6. Write an R code that counts the number of passengers per boarding point. CHALLENGE: can you show the number of passengers per sex per boarding point?

```
count(titanic, titanic$embarked)
```

```
##   titanic$embarked   n
## 1                C 168
## 2                Q  77
## 3                S 644
## 4             <NA>   2
```

```
count(titanic, titanic$embarked, titanic$sex)
```

```
##   titanic$embarked titanic$sex   n
## 1                C      female  72
## 2                C        male  96
## 3                Q      female  36
## 4                Q        male  41
## 5                S      female 201
## 6                S        male 443
## 7             <NA>      female   2
```

Tips: some useful functions that you may (or may not!) use to solve this assignment: - `as.data.frame(x)`: converts a vector `x` to a data.frame - `unique(x)`: returns a vector/data.frame containing only the unique elements in `x` - `c(x, y, z, ...)`: creates a vector with the elements provided as parameters - `is.na(x)`: returns `TRUE` if the value of a particular observation in `x` is `NA` - `x[! x %in% c('a', 'b')]`: subsets an array, removing the positions with the elements `'a'` and `'b'` - `paste(x, y, z, ...)`: concatenate the strings provided as parameters - `ifelse(x, y, z)`: if `x` is `TRUE`, returns `y`, otherwise returns `z`