Data 621 - Blog 1

Leticia Salazar

November 1, 2022

Exploration and Visualizations

Data exploration is one of the first things you'll do when you are analyzing data sets for any project. It allows you to have an initial understanding the variables in your given data set, detect outliers and explore relationships of your data before any preparation or deeper analysis.

Data visualization helps present and understand your data to later share it with others in an accessible way. Yet visualizations alone can also have a downside, if not presented well it can be misrepresented or misinterpreted. There's plenty of things to consider when creating visualization like color palette, text sizing, ledger, etc but before we get to all that there's a couple different types of visualizations to know:

- Univariate Graphs
  - Categorical
  - Quantitative
- Bivariate Graphs
  - Categorical vs Categorical
  - Quantitative vs Quantitative
  - Categorical vs Quantitative
- Multivariate Graphs
  - Grouping
  - Faceting

Using the `marriage`, `mpg` and `salaries` data set from the library `ggplot2` as well as the `iris` data set from the library `datasets` to explore and create different types of visualizations.

Load Libraries

```
library(ggplot2) # used for data sets and graphing
library(tidyverse) # used for exploration of data
library(mosaicData) # used to get data set marriage
library(datasets) # iris data set
library(hrbrthemes) # adds themes to ggplot2
library(carData) # Salaries data set
```

Data Exploration

For this portion we will use the `mpg` data set to explore. After importing your data set you'd want to display the data. You'll find that plenty of times you have a large data set and loading it all will just end up being chaotic. By using the `head()` you'll be able to display the first 6 rows of the data set as seen below:

```
# display the data
head(mpg)
```

```
## # A tibble: 6 x 11
##   manufacturer model displ  year   cyl trans      drv     cty   hwy fl    class
##   <chr>        <chr> <dbl> <int> <int> <chr>      <chr> <int> <int> <chr> <chr>
## 1 audi         a4      1.8  1999     4 auto(l5)   f        18    29 p     compa~
## 2 audi         a4      1.8  1999     4 manual(m5) f        21    29 p     compa~
## 3 audi         a4      2    2008     4 manual(m6) f        20    31 p     compa~
## 4 audi         a4      2    2008     4 auto(av)   f        21    30 p     compa~
## 5 audi         a4      2.8  1999     6 auto(l5)   f        16    26 p     compa~
## 6 audi         a4      2.8  1999     6 manual(m5) f        18    26 p     compa~
```

You may also want to just get the column names in case you want to rename them.

```
# display column names
colnames(mpg)
```

```
##  [1] "manufacturer" "model"        "displ"        "year"         "cyl"
##  [6] "trans"        "drv"          "cty"          "hwy"          "fl"
## [11] "class"
```

The function below allows you to view the number of columns and rows in a data set

```
# display columns and rows
dim(mpg)
```

```
## [1] 234  11
```

You may also want to explore the structure of the data by using the `str()`

```
# display structure of data
str(mpg)
```

```
## tibble [234 x 11] (S3: tbl_df/tbl/data.frame)
##  $ manufacturer: chr [1:234] "audi" "audi" "audi" "audi" ...
##  $ model       : chr [1:234] "a4" "a4" "a4" "a4" ...
##  $ displ       : num [1:234] 1.8 1.8 2 2 2.8 2.8 3.1 1.8 1.8 2 ...
##  $ year        : int [1:234] 1999 1999 2008 2008 1999 1999 2008 1999 1999 2008 ...
##  $ cyl         : int [1:234] 4 4 4 4 6 6 6 4 4 4 ...
##  $ trans       : chr [1:234] "auto(l5)" "manual(m5)" "manual(m6)" "auto(av)" ...
##  $ drv         : chr [1:234] "f" "f" "f" "f" ...
##  $ cty         : int [1:234] 18 21 20 21 16 18 18 18 16 20 ...
##  $ hwy         : int [1:234] 29 29 31 30 26 26 27 26 25 28 ...
##  $ fl          : chr [1:234] "p" "p" "p" "p" ...
##  $ class       : chr [1:234] "compact" "compact" "compact" "compact" ...
```

If you want to find the statistical structure of any data set, the `summary()`will allow you to do just that. This function give you the minimum, first quartile, median, mean, third quartile and maximum values of each variable. All other non numerical values you get the count of values in that variable.

```
# finding statistical structure of the data
summary(mpg)
```

```
##  manufacturer          model               displ            year
##  Length:234         Length:234         Min.   :1.600   Min.   :1999
##  Class :character   Class :character   1st Qu.:2.400   1st Qu.:1999
##  Mode  :character   Mode  :character   Median :3.300   Median :2004
##                                        Mean   :3.472   Mean   :2004
##                                        3rd Qu.:4.600   3rd Qu.:2008
##                                        Max.   :7.000   Max.   :2008
##       cyl           trans               drv                 cty
##  Min.   :4.000   Length:234         Length:234         Min.   : 9.00
##  1st Qu.:4.000   Class :character   Class :character   1st Qu.:14.00
##  Median :6.000   Mode  :character   Mode  :character   Median :17.00
##  Mean   :5.889                                         Mean   :16.86
##  3rd Qu.:8.000                                         3rd Qu.:19.00
##  Max.   :8.000                                         Max.   :35.00
##       hwy             fl                class
##  Min.   :12.00   Length:234         Length:234
##  1st Qu.:18.00   Class :character   Class :character
##  Median :24.00   Mode  :character   Mode  :character
##  Mean   :23.44
##  3rd Qu.:27.00
##  Max.   :44.00
```

I also like to use the `glimpse()` which semi combines the `dim()` and `str()` functions as seen below

```
glimpse(mpg)
```

```
## Rows: 234
## Columns: 11
## $ manufacturer <chr> "audi", "audi", "audi", "audi", "audi", "audi", "audi", "~
## $ model        <chr> "a4", "a4", "a4", "a4", "a4", "a4", "a4", "a4 quattro", "~
## $ displ        <dbl> 1.8, 1.8, 2.0, 2.0, 2.8, 2.8, 3.1, 1.8, 1.8, 2.0, 2.0, 2.~
## $ year         <int> 1999, 1999, 2008, 2008, 1999, 1999, 2008, 1999, 1999, 200~
## $ cyl          <int> 4, 4, 4, 4, 6, 6, 6, 4, 4, 4, 4, 6, 6, 6, 6, 6, 6, 8, 8, ~
## $ trans        <chr> "auto(l5)", "manual(m5)", "manual(m6)", "auto(av)", "auto~
## $ drv          <chr> "f", "f", "f", "f", "f", "f", "f", "4", "4", "4", "4", "4~
## $ cty          <int> 18, 21, 20, 21, 16, 18, 18, 18, 16, 20, 19, 15, 17, 17, 1~
## $ hwy          <int> 29, 29, 31, 30, 26, 26, 27, 26, 25, 28, 27, 25, 25, 25, 2~
## $ fl           <chr> "p", "p", "p", "p", "p", "p", "p", "p", "p", "p", "p", "p~
## $ class        <chr> "compact", "compact", "compact", "compact", "compact", "c~
```

We can't forget the missing values or NA's either and `colSums` is great at providing the count of the missing values for each column. `rowSums` provides the count of the missing values in each row.

```
# can't forget about the NA's
colSums(is.na(mpg))
```

```
## manufacturer        model        displ         year          cyl        trans
##            0            0            0            0            0            0
##          drv          cty          hwy           fl        class
##            0            0            0            0            0
```

```
#rowSums(is.na(mpg))
```

Visualizations

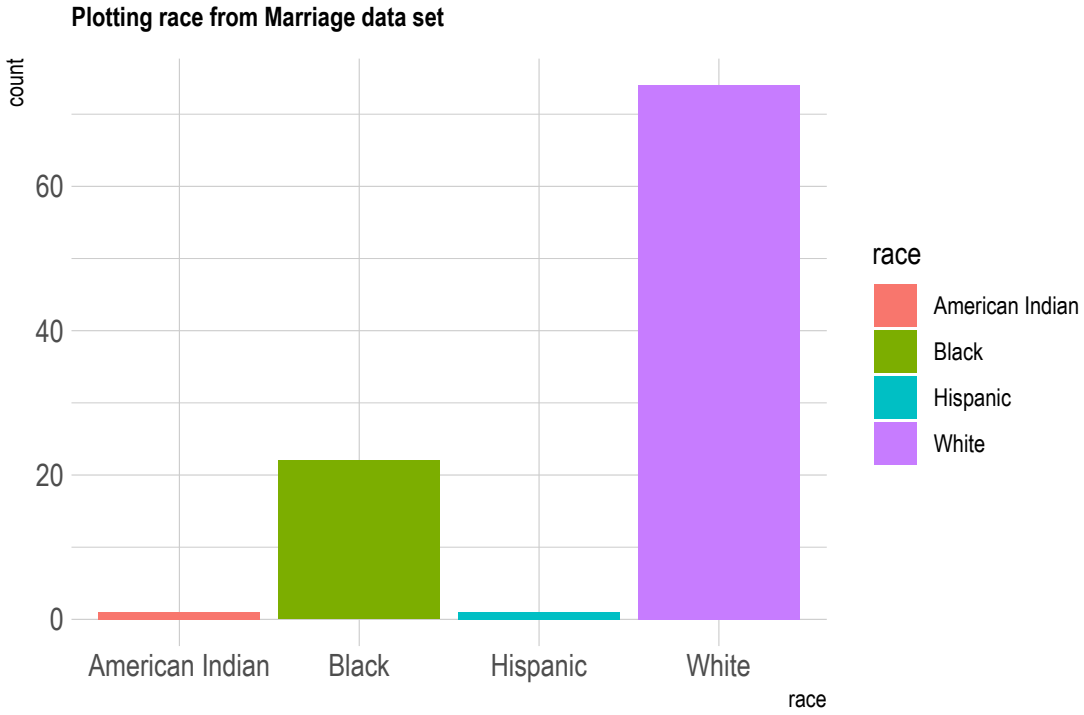
Part of the exploration process is to also visualize what the data looks like so far. Be mindful that every categorical and quantitative variable in a data set should be displayed accordingly to represent your data well and avoid misrepresentation / misinterpretation. Here's a couple of visualization you can start with:

Univariate Graphs

Univariate graphs help display each attribute **alone**

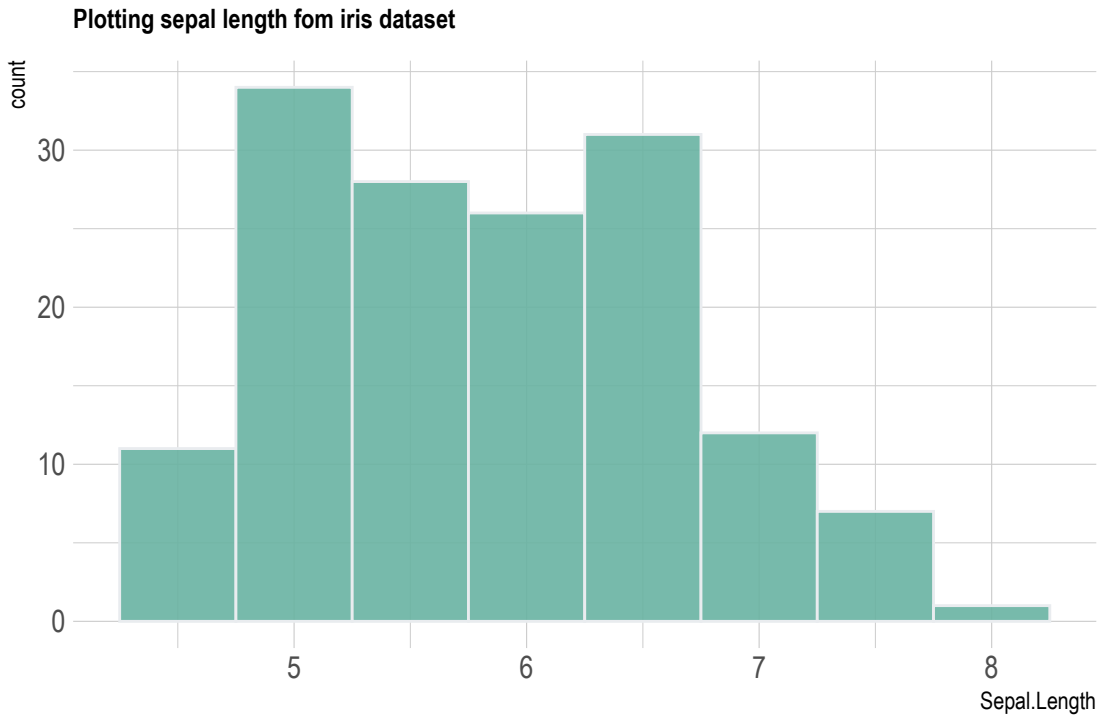
Categorical: Variables pertaining to race, sex, species, etc are labeled categorical

```
# bar chart
Marriage %>%
  ggplot(aes(x = race, fill = race)) +
    geom_bar() +
    ggtitle("Plotting race from Marriage data set") +
    theme_ipsum() +
    theme(
      plot.title = element_text(size=10)
    )
```

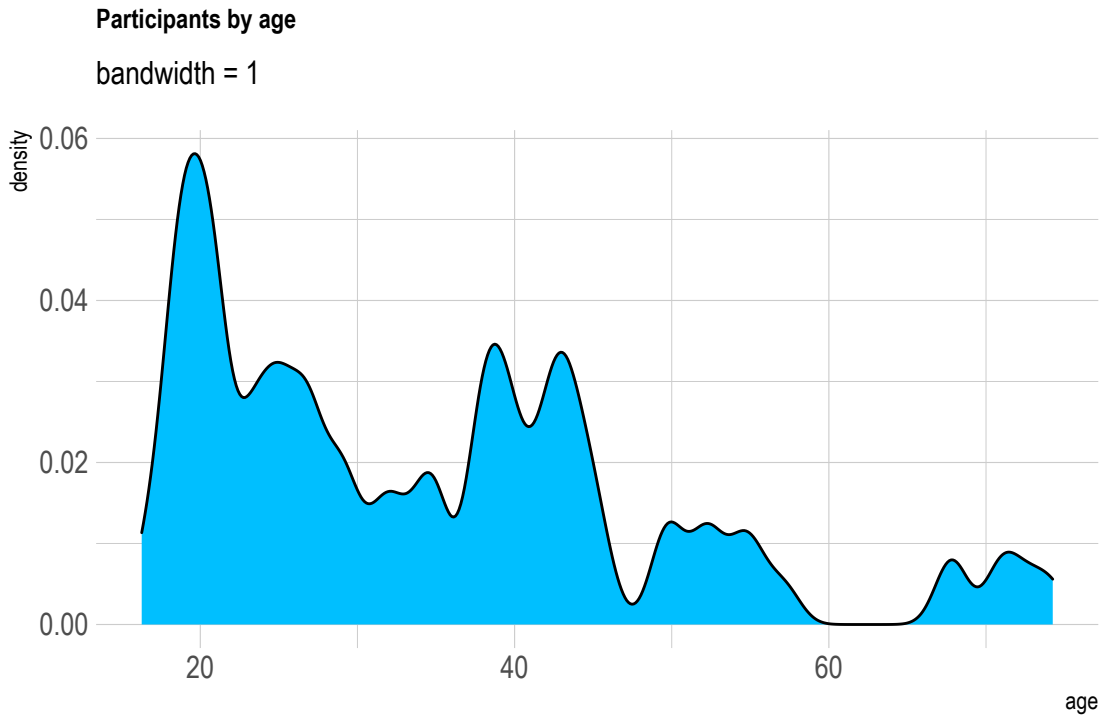


Quantitative Variables such as age, weight, length, etc are labeled quantitative

```
# histogram
iris %>% # pipe through the data set
ggplot( aes(x = Sepal.Length)) +
    geom_histogram( binwidth = 0.5, fill = "#69b3a2", color = "#e9ecef", alpha=0.9) +
  ggtitle("Plotting sepal length fom iris dataset") +
    theme_ipsum() +
    theme(
      plot.title = element_text(size=10)
    )
```

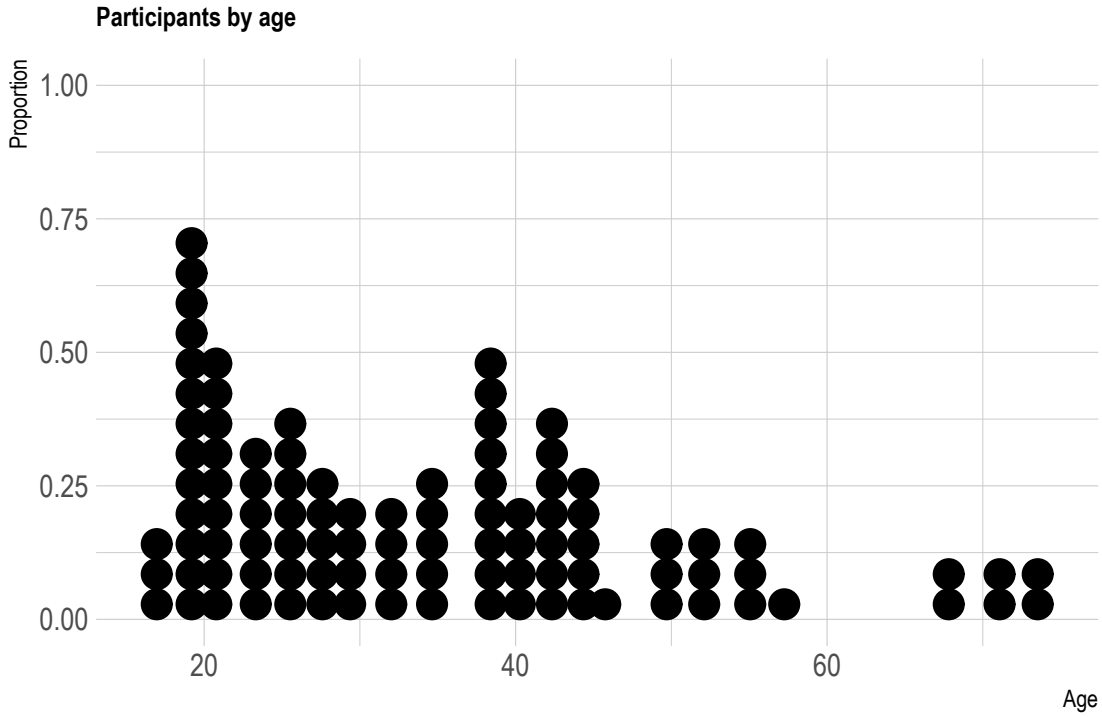



```
# density plot
ggplot(Marriage, aes(x = age)) +
  geom_density(fill = "deepskyblue",
               bw = 1) +
  labs(title = "Participants by age",
       subtitle = "bandwidth = 1") +
    theme_ipsum() +
    theme(
      plot.title = element_text(size=10)
    )
```




```
# dot chart
ggplot(Marriage, aes(x = age)) +
  geom_dotplot() +
  labs(title = "Participants by age",
       y = "Proportion",
       x = "Age") +
    theme_ipsum() +
    theme(
      plot.title = element_text(size=10)
    )
```

```
## Bin width defaults to 1/30 of the range of the data. Pick better value with `binwidth`.
```

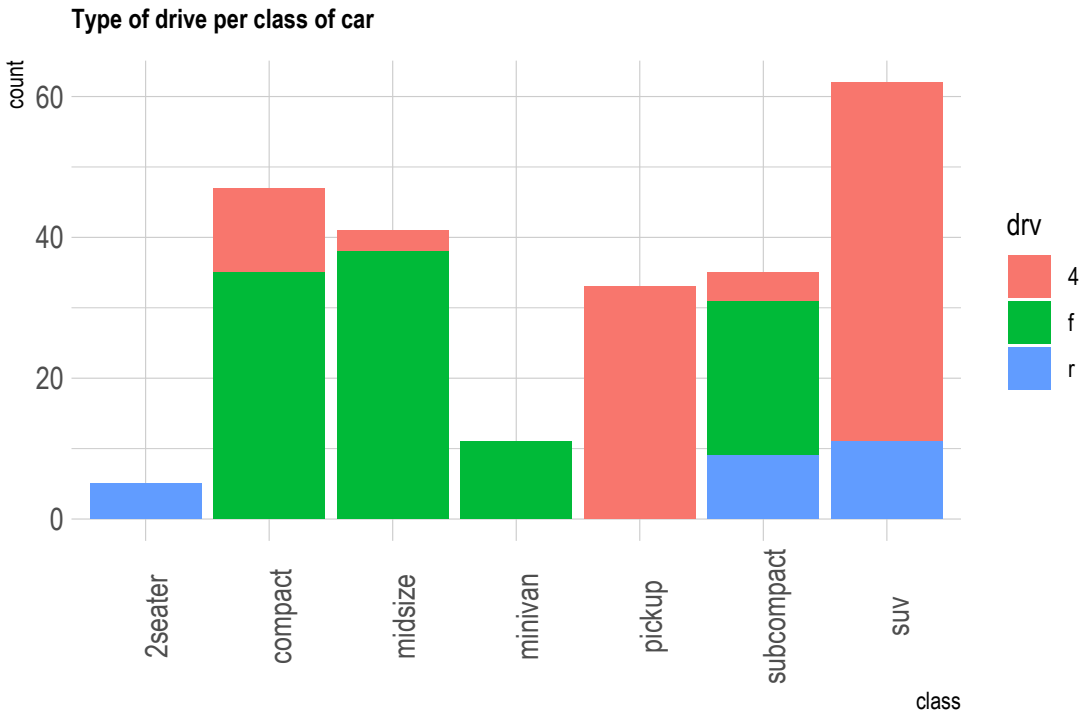


Bivariate Graphs

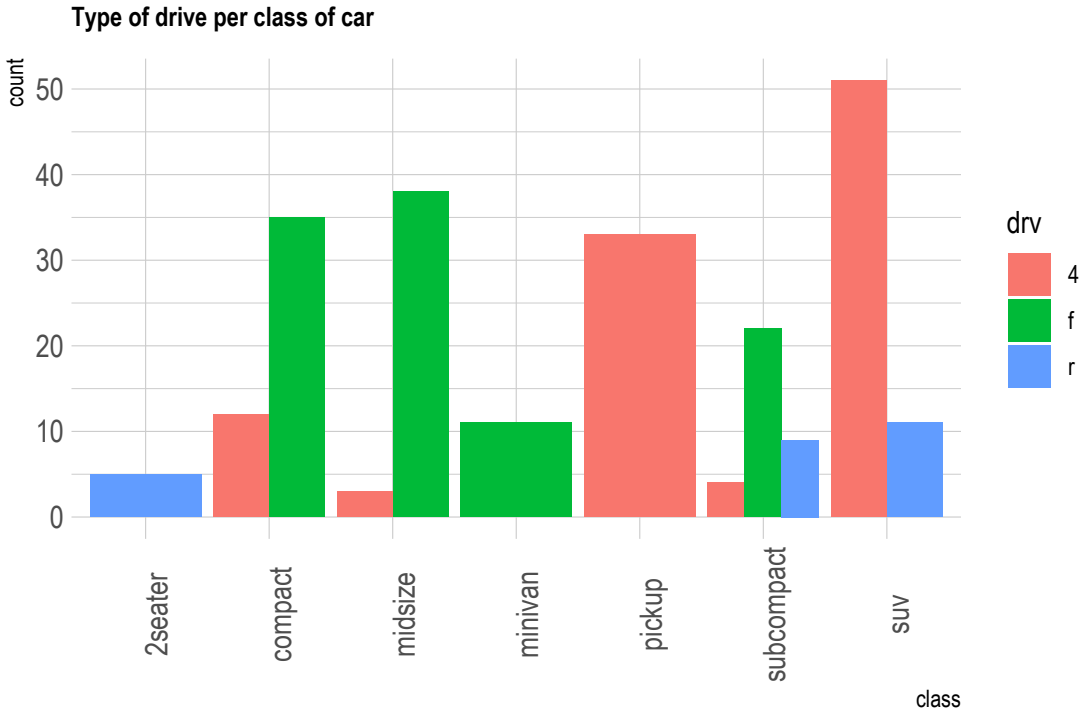
Bivariate graphs help you display **two** attributes

```
# stacked bar chart
mpg %>%
  ggplot(aes(x = class, fill = drv)) +
    geom_bar(position = "stack") +
    ggtitle("Type of drive per class of car") +
    theme_ipsum() +
    theme(
      plot.title = element_text(size=10), axis.text.x = element_text(angle = 90)
    )
```

Categorical vs. Categorical

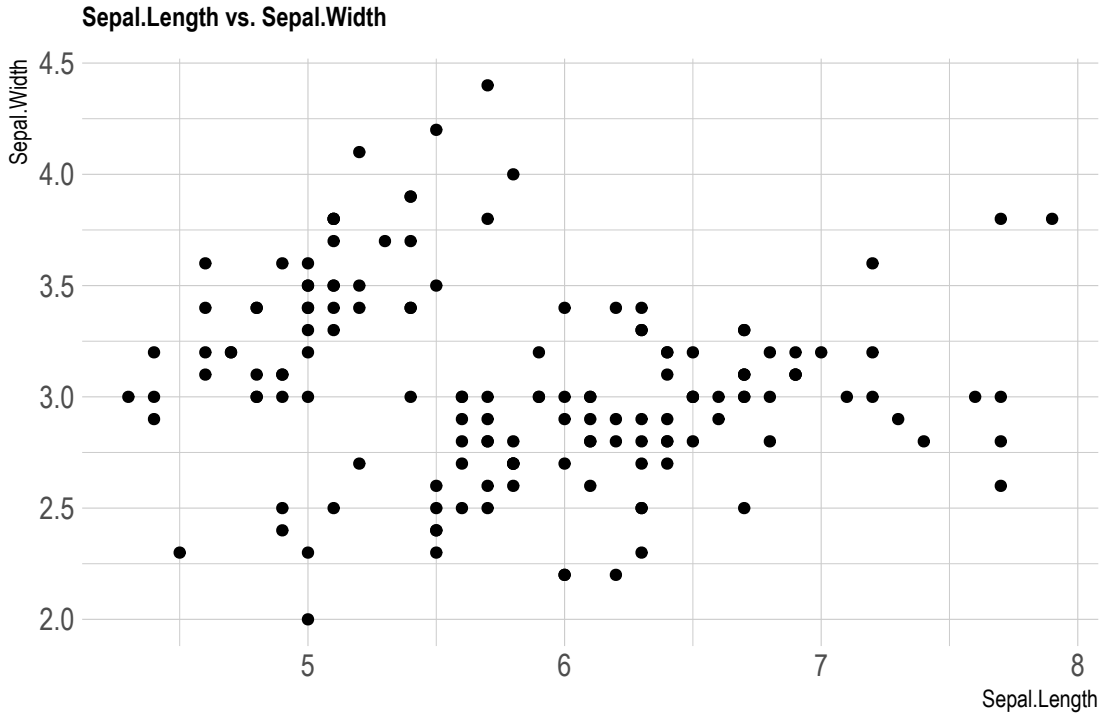



```
# grouped bar chart
mpg %>%
  ggplot(aes(x = class, fill = drv)) +
    geom_bar(position = "dodge") +
    ggtitle("Type of drive per class of car") +
    theme_ipsum() +
    theme(
      plot.title = element_text(size=10), axis.text.x = element_text(angle = 90)
    )
```

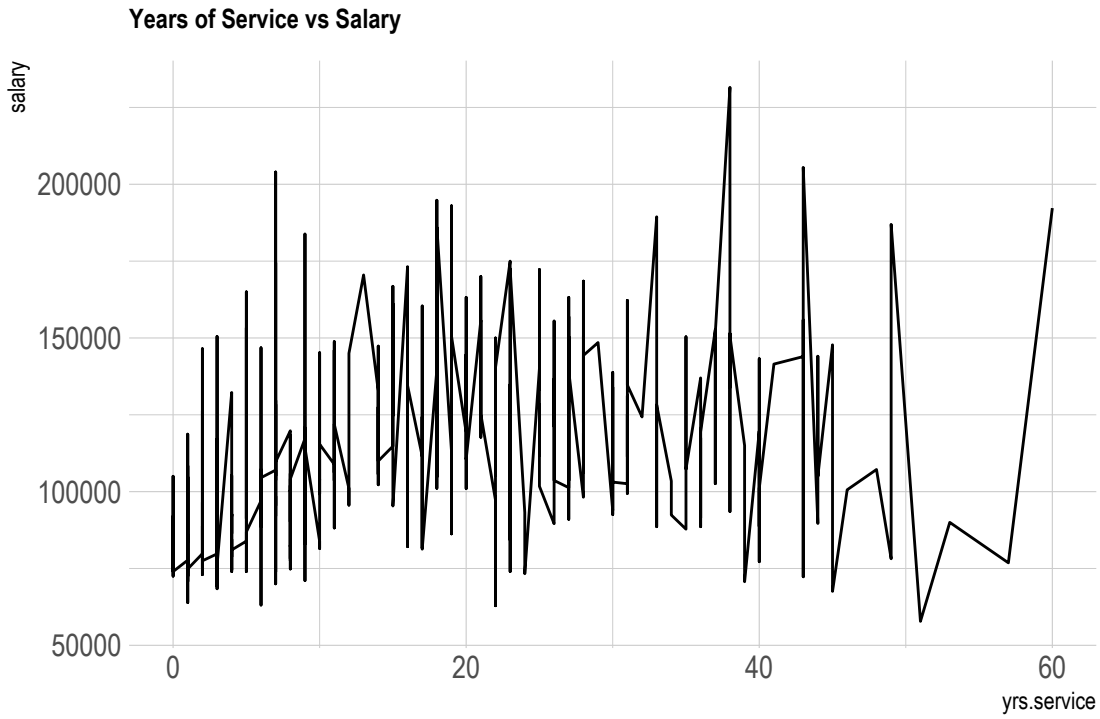



```
# scatter plot
iris %>%
ggplot(aes(x = Sepal.Length,
           y = Sepal.Width)) +
  geom_point() +
  labs(title = "Sepal.Length vs. Sepal.Width") +
    theme_ipsum() +
    theme(
      plot.title = element_text(size=10)
    )
```

Quantitative vs. Quantitative

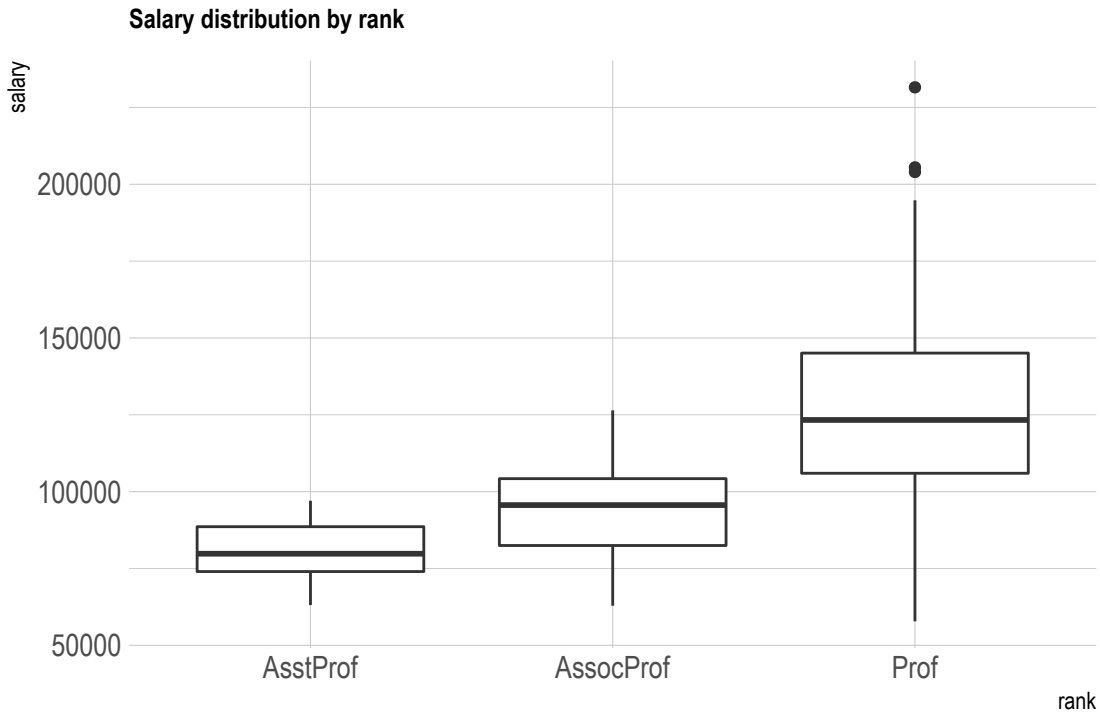



```
# line plot
Salaries %>%
ggplot( aes(x = yrs.service,
            y = salary)) +
  geom_line() +
  labs(title = "Years of Service vs Salary") +
    theme_ipsum() +
    theme(
      plot.title = element_text(size=10)
    )
```




```
# box plot
Salaries %>%
ggplot(aes(x = rank,
           y = salary)) +
  geom_boxplot() +
  labs(title = "Salary distribution by rank") +
    theme_ipsum() +
    theme(
      plot.title = element_text(size=10)
    )
```

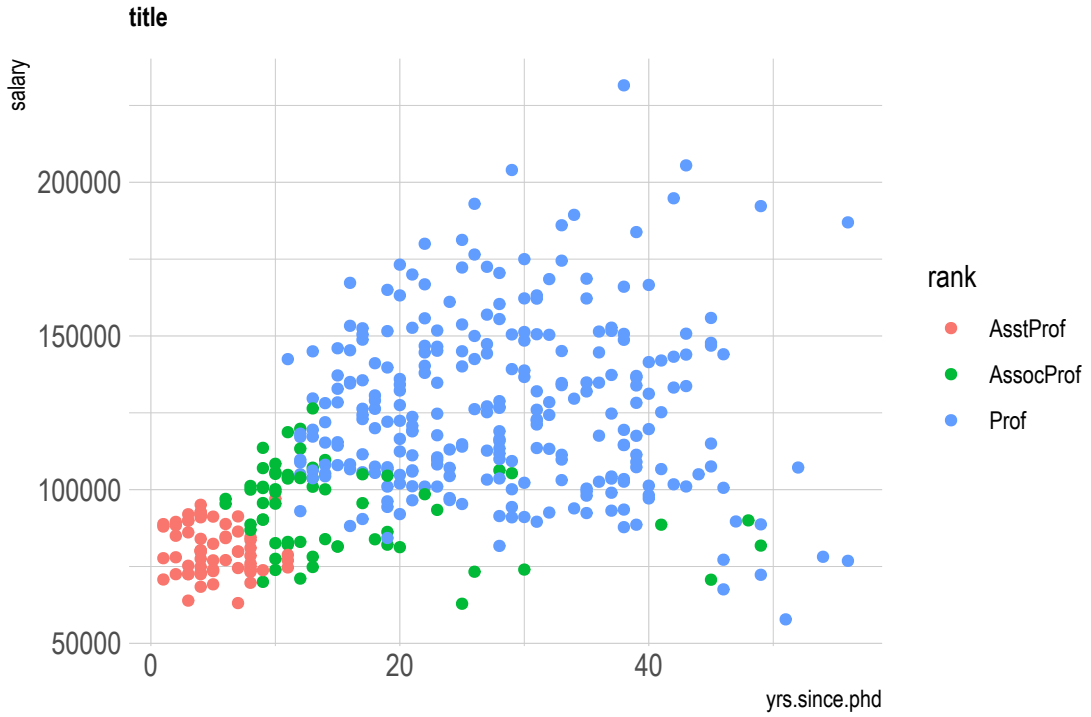
Categorical vs. Quantitative



Multivariate Graphs

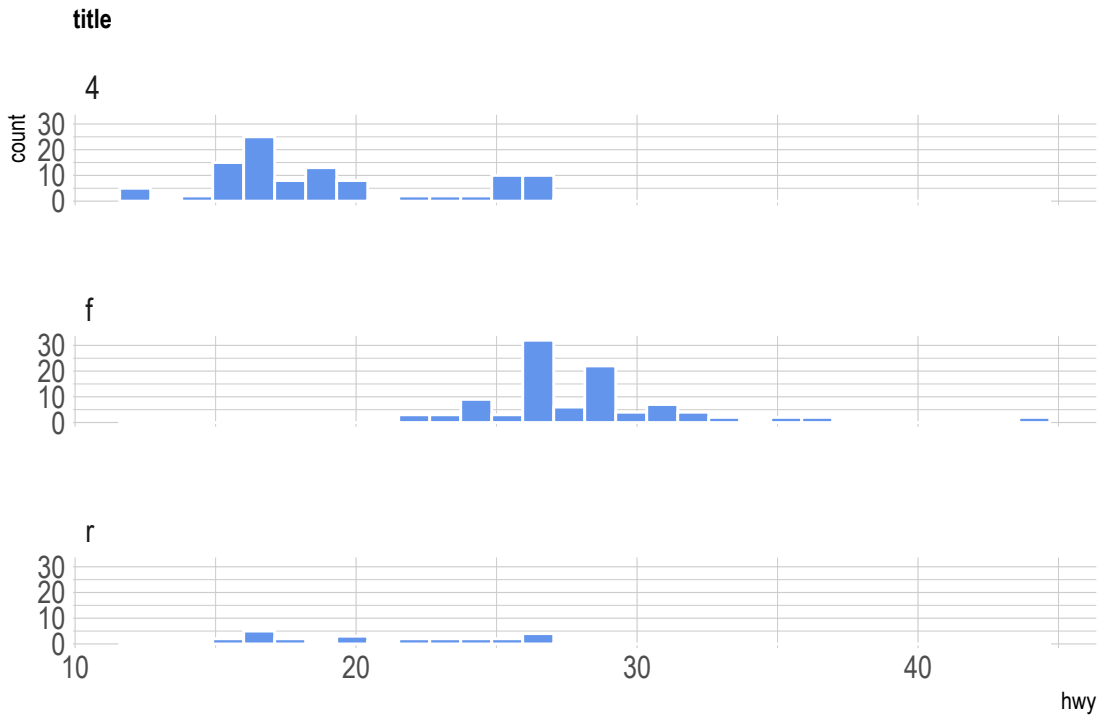
Multivariate graphs help you better understand the **interactions between attributes**

```
# grouping scatter plots
Salaries %>%
ggplot(aes(x = yrs.since.phd,
                    y = salary,
                    color=rank)) +
  geom_point() +
  labs(title = "title") +
    theme_ipsum() +
    theme(
      plot.title = element_text(size=10)
    )
```




```
# faceting
mpg %>%
ggplot(aes(x = hwy)) +
  geom_histogram(fill = "cornflowerblue",
                 color = "white") +
  facet_wrap(~drv, ncol = 1) +
  labs(title = "title") +
    theme_ipsum() +
    theme(
      plot.title = element_text(size=10)
    )
```

```
## `stat_bin()` using `bins = 30`. Pick better value with `binwidth`.
```



While there's plenty of other ways to perform these tasks being able to know how to start will make it easier to use other functions to explore your data set or libraries other than `ggplot2` to upgrade your visualizations.

References:

- 9 useful R data visualization packages for Data Visualization. Mode. (n.d.). Retrieved October 28, 2022, from https://mode.com/blog/r-data-visualization-packages/
- Data visualization with R - github pages. (n.d.). Retrieved November 8, 2022, from https://rkabacoff.github.io/datavis/
- What is data visualization? definition, examples, and learning resources. Tableau. (n.d.). Retrieved November 8, 2022, from https://www.tableau.com/learn/articles/data-visualization
- Brownlee, J. (2019, August 22). Better understand your data in R using visualization (10 recipes you can use today). Machine Learning Mastery. Retrieved November 8, 2022, from https://machinelearningmastery.com/data-visualization-in-r/