Stat. 450 Section 1 or 2: Homework 2

**Prof. Eric A. Suess**

So how should you complete your homework for this class?

* First thing to do is type all of your information about the problems you do in the text part of your R Notebook.
* Second thing to do is type all of your R code into R chunks that can be run.
* If you load the tidyverse in an R Notebook chunk, be sure to include the “message = FALSE” in the {r}, so {r message = FALSE}.
* Last thing is to spell check your R Notebook. Edit > Check Spelling… or hit the F7 key.

Homework 2:

Read: Chapter 3, 4
Do 3.6.1 Exercises 1, 2, 3, 4, 5, 6
Do 3.7.1 Exercises 1, 2, 3, 4, 5
Do 3.8.1 Exercises 1, 4
Do 3.9.1 Exercises 4
Do 4.4 Practice 3

library(tidyverse)

# 3.6.1 Exercises

## 1.

geom\_line(), geom\_boxplot(), geom\_histogram(), geom\_area()

See Chapter 3 or the [RStudio Data Visualization with ggplot2 Cheat Sheet](https://www.rstudio.com/wp-content/uploads/2015/03/ggplot2-cheatsheet.pdf).

## 2.

This code will plot data from the mpg data frame, x is displ and y is hwy miles per gallon, with the color of the points from drv. And a smoother is included. Lets see what it looks like.

ggplot(data = mpg, mapping = aes(x = displ, y = hwy, color = drv)) +
 geom\_point() +
 geom\_smooth(se = FALSE)

## `geom\_smooth()` using method = 'loess' and formula 'y ~ x'



## 3.

With show.legend = FALSE, no legend is include in the plot. This was good earlier in the book where there are 3 plots made side-by-side. But legends should be include all the time so the visualization is clear.

ggplot(data = mpg) +
 geom\_smooth(
 mapping = aes(x = displ, y = hwy, color = drv),
 show.legend = FALSE
 )

## `geom\_smooth()` using method = 'loess' and formula 'y ~ x'



ggplot(data = mpg) +
 geom\_smooth(
 mapping = aes(x = displ, y = hwy, color = drv)
 )

## `geom\_smooth()` using method = 'loess' and formula 'y ~ x'



## 4.

The se option puts the error regions agound the smoothers. If se is removed there are not error regions included.

ggplot(data = mpg) +
 geom\_smooth(
 mapping = aes(x = displ, y = hwy, color = drv), se = FALSE
 )

## `geom\_smooth()` using method = 'loess' and formula 'y ~ x'



## 5.

They should look the same because if the aes is in the ggplot function it is used in the folling functions. If the aes is not in the ggplot function each of the following functions need aes.

ggplot(data = mpg, mapping = aes(x = displ, y = hwy)) +
 geom\_point() +
 geom\_smooth()

## `geom\_smooth()` using method = 'loess' and formula 'y ~ x'



ggplot() +
 geom\_point(data = mpg, mapping = aes(x = displ, y = hwy)) +
 geom\_smooth(data = mpg, mapping = aes(x = displ, y = hwy))

## `geom\_smooth()` using method = 'loess' and formula 'y ~ x'



ggplot() +
 geom\_point(data = mpg, mapping = aes(x = displ, y = hwy)) +
 geom\_smooth()



ggplot() +
 geom\_point() +
 geom\_smooth(data = mpg, mapping = aes(x = displ, y = hwy))

## `geom\_smooth()` using method = 'loess' and formula 'y ~ x'



## 6.

mpg %>% ggplot(aes(x=displ, y= hwy)) +
 geom\_point(size = 3) +
 geom\_smooth(se = FALSE)

## `geom\_smooth()` using method = 'loess' and formula 'y ~ x'



mpg %>% ggplot(aes(x=displ, y= hwy, group = drv)) +
 geom\_point(size = 3) +
 geom\_smooth(se = FALSE)

## `geom\_smooth()` using method = 'loess' and formula 'y ~ x'



mpg %>% ggplot(aes(x=displ, y= hwy, color = drv)) +
 geom\_point(size = 3) +
 geom\_smooth(se = FALSE)

## `geom\_smooth()` using method = 'loess' and formula 'y ~ x'



mpg %>% ggplot(aes(x=displ, y=hwy)) +
 geom\_point(aes(color = drv), size = 3) +
 geom\_smooth()

## `geom\_smooth()` using method = 'loess' and formula 'y ~ x'



mpg %>% ggplot(aes(x=displ, y= hwy, color=drv, linetype = drv)) +
 geom\_point(size = 3) +
 geom\_smooth(se = FALSE)

## `geom\_smooth()` using method = 'loess' and formula 'y ~ x'



mpg %>% ggplot(aes(x=displ, y= hwy, color=drv)) +
 geom\_point(size = 3)



# 3.7.1 Exercises

## 1.

The default geometry of stat\_summary() is identity(). The more general geom is geom\_pointrange(). It can be used to make the same plot and has more flexibility.

diamonds %>% ggplot(mapping = aes(x = cut, y = depth)) +
 stat\_summary(
 fun.ymin = min,
 fun.ymax = max,
 fun.y = median
 )



diamonds %>% ggplot(mapping = aes(x = cut, y = depth)) +
 geom\_pointrange(
 stat = "summary"
 )

## No summary function supplied, defaulting to `mean\_se()



diamonds %>% ggplot(mapping = aes(x = cut, y = depth)) +
 geom\_pointrange(
 stat = "summary",
 fun.ymin = min,
 fun.ymax = max,
 fun.y = median
 )



## 2.

See [Bar charts](https://ggplot2.tidyverse.org/reference/geom_bar.html).

The geom\_bar() counts up the number of diamonds in each cut in the data frame.

diamonds %>% ggplot(mapping = aes(x = cut)) +
 geom\_bar()



The geom\_col(\*) can be used after counting directly then piping the result into ggplot with geom\_col().

diamonds %>% group\_by(cut) %>%
 count() %>%
 ggplot(aes(x=cut, y=n)) +
 geom\_col()



## 3.

All of the geom functions take raw data and work with the basics of the plot. The stat functions perform further transformations on the data and can do more with the plot.

[stackoverflow: What is the difference …](https://stackoverflow.com/questions/38775661/what-is-the-difference-between-geoms-and-stats-in-ggplot2)

mpg %>% ggplot(aes(y=hwy)) + geom\_boxplot()



mpg %>% ggplot(aes(y=hwy)) + stat\_boxplot()



geom\_bar() stat\_count()

geom\_col() stat\_count()

geom\_bin2d() stat\_bin\_2d()

geom\_boxplot() stat\_boxplot()

geom\_contour() stat\_contour()

geom\_count() stat\_sum()

geom\_density() stat\_density()

geom\_density\_2d() stat\_density\_2d()

geom\_hex() stat\_bin\_hex()

geom\_freqpoly() stat\_bin()

geom\_histogram() stat\_bin()

geom\_qq\_line() stat\_qq\_line()

geom\_qq() stat\_qq()

geom\_quantile() stat\_quantile()

geom\_smooth() stat\_smooth()

geom\_violin() stat\_ydensity()

stat\_sf() geom\_sf()

## 4.

Here is the link to the ggplot website for [geom\_smooth](https://ggplot2.tidyverse.org/reference/geom_smooth.html). See the bottom of the page for the computed variables.

**Computed variables**

1. **y** predicted value
2. **ymin** lower pointwise confidence interval around the mean
3. **ymax** upper pointwise confidence interval around the mean
4. **se** standard error

The main two arguments that control the behavior are

1. **method** Smoothing method (function) to use, eg. lm, glm, gam, loess, MASS::rlm.

For method = “auto” the smoothing method is chosen based on the size of the largest group (across all panels). loess() is used for less than 1,000 observations; otherwise mgcv::gam() is used with formula = y ~ s(x, bs = “cs”). Somewhat anecdotally, loess gives a better appearance, but is O(n^2) in memory, so does not work for larger datasets.

If you have fewer than 1,000 observations but want to use the same gam model that method = “auto” would use then set method = “gam”, formula = y ~ s(x, bs = “cs”).

1. **formula** Formula to use in smoothing function, eg. y ~ x, y ~ poly(x, 2), y ~ log(x)

## 5.

The group = 1 makes the bars add up to one, without it each bar adds to one.

ggplot(data = diamonds) +
 geom\_bar(mapping = aes(x = cut, y = ..prop..))



ggplot(data = diamonds) +
 geom\_bar(mapping = aes(x = cut, fill = color, y = ..prop..))



ggplot(data = diamonds) +
 geom\_bar(mapping = aes(x = cut, y = ..prop.., group=1))



ggplot(data = diamonds) +
 geom\_bar(mapping = aes(x = cut, fill = color, y = ..prop.., group = color))



# 3.8.1 Exercises

## 1.

It has no title.

ggplot(data = mpg, mapping = aes(x = cty, y = hwy)) +
 geom\_point() +
 ggtitle("Help me make a title.")



## 4.

The default position is what is called position\_dodge(). See the ggplot2 website for [box and whisker](https://ggplot2.tidyverse.org/reference/geom_boxplot.html).

glimpse(mpg)

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

mpg %>% ggplot(aes(x = trans, y = hwy, color = class)) +
 geom\_boxplot()



# 3.9.1 Exercises

It tells us that hwy miles per gallon always higher than the city miles per gallon for all cars.

The coord\_fixed makes the plot so the scales are the same. So the x = y line is 45 degrees.

The geom\_abline() draws a 45 degree line.

ggplot(data = mpg, mapping = aes(x = cty, y = hwy)) +
 geom\_point() +
 geom\_abline()



Making maps is cool. Just put this here to look at the map of the USA.

usa <- map\_data("usa")

##
## Attaching package: 'maps'

## The following object is masked from 'package:purrr':
##
## map

ggplot(usa, aes(long, lat, group = group)) +
 geom\_polygon(fill = "white", colour = "black")



ggplot(usa, aes(long, lat, group = group)) +
 geom\_polygon(fill = "white", colour = "black") +
 coord\_quickmap()



# 4.4

# Practice 3.

It gives the keyboard short cuts.

Or use Tools > Keyboard shortcuts.