Plotting: ggplot2

TheRBootcamp, http://therbootcamp.github.io (http://therbootcamp.github.io)

Source: https://www.rstudio.com/ (https://www.rstudio.com/)

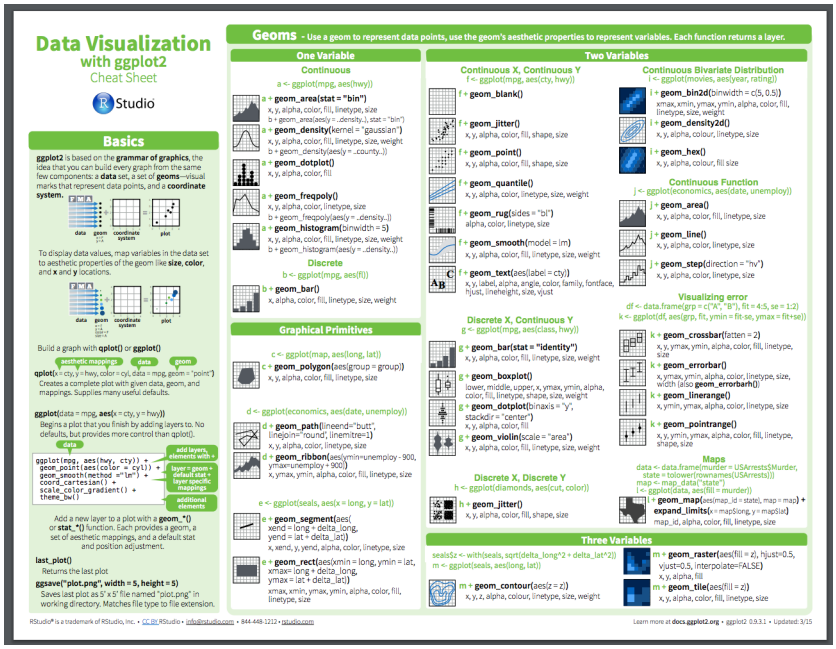
Slides

Here are the introduction slides for this practical on Plotting 1.0: ggplot! (https://therbootcamp.github.io/_sessions/D3S2_PlottingI/PlottingI.html)

Overview

In this practical you'll practice plotting data with the `ggplot2` package.

Cheatsheet



https://www.rstudio.com/wp-content/uploads/2015/03/ggplot2-cheatsheet.pdf (https://www.rstudio.com/wp-content/uploads/2015/03/ggplot2-cheatsheet.pdf)

If you don't have it already, you can access the `ggplot2` cheatsheet here https://www.rstudio.com/wp-content/uploads/2015/03/ggplot2-cheatsheet.pdf (https://www.rstudio.com/wp-content/uploads/2015/03/ggplot2-cheatsheet.pdf). This has a nice overview of all the major functions in ggplot2.

Examples

- The following examples will take you through the steps of creating both simple and complex plots with `ggplot2` . Try to go through each line of code and see how it works!

```r
# ------------------------------------------------
# Examples of using ggplot2 on the mpg data
# ------------------------------------------------

library(tidyverse)         # Load tidyverse (which contains ggplot2!)

mpg # Look at the mpg data

# Just a blank space without any aesthetic mappings
ggplot(data = mpg)

# Set the overall plotting theme
theme_set(theme_bw())   # theme_bw(), theme_minimal(), theme_classic()

# Now add a mapping where engine displacement (displ) and highway miles per gallon (h
wy) are mapped to the x and y aesthetics
ggplot(data = mpg,
       mapping = aes(x = displ, y = hwy))   # Map displ to x-axis and hwy to y-axis

#  Add points with geom_point()
ggplot(data = mpg,
       mapping = aes(x = displ, y = hwy)) +
       geom_point()

#  Add points with geom_count()
ggplot(data = mpg,
       mapping = aes(x = displ, y = hwy)) +
       geom_count()

# Again, but with some additional arguments

ggplot(data = mpg,
       mapping = aes(x = displ, y = hwy)) +
       geom_point(col = "red",                  # Red points
                  size = 3,                     # Larger size
                  alpha = .5,                   # Transparent points
                  position = "jitter") +        # Jitter the points
         scale_x_continuous(limits = c(1, 15)) +  # Axis limits
         scale_y_continuous(limits = c(0, 50))


# Assign class to the color aesthetic and add labels with labs()

ggplot(data = mpg,
  mapping = aes(x = displ, y = hwy, col = class)) +  # Change color based on class co
lumn
  geom_point(size = 3, position = 'jitter') +
  labs(x = "Engine Displacement in Liters",
       y = "Highway miles per gallon",
       title = "MPG data",
       subtitle = "Cars with higher engine displacement tend to have lower highway mp
g",
       caption = "Source: mpg data in ggplot2")


# Add a regression line for each class
```

```
ggplot(data = mpg,
       mapping = aes(x = displ, y = hwy, color = class)) +
  geom_point(size = 3, alpha = .9) +
  geom_smooth(method = "lm")

# Add a regression line for all classes

ggplot(data = mpg,
       mapping = aes(x = displ, y = hwy, color = class)) +
  geom_point(size = 3, alpha = .9) +
  geom_smooth(col = "blue", method = "lm")



# Facet by class
ggplot(data = mpg,
       mapping = aes(x = displ, y = hwy, color = factor(cyl))) +
  geom_point() +
  facet_wrap(~ class)


# Another fancier example

ggplot(data = mpg,
       mapping = aes(x = cty, y = hwy)) +
       geom_count(aes(color = manufacturer)) +     # Add count geom (see ?geom_count)
       geom_smooth() +                   # smoothed line without confidence interval
       geom_text(data = filter(mpg, cty > 25),
                 aes(x = cty,y = hwy,
                     label = rownames(filter(mpg, cty > 25))),
                     position = position_nudge(y = -1),
                               check_overlap = TRUE,
                     size = 5) +
       labs(x = "City miles per gallon",
            y = "Highway miles per gallon",
            title = "City and Highway miles per gallon",
            subtitle = "Numbers indicate cars with highway mpg > 25",
            caption = "Source: mpg data in ggplot2",
            color = "Manufacturer",
            size = "Counts")
```

Tasks

Getting the data and project setup

1. For this practical we'll play around with a few different datasets that are contained in different packages. The datasets, and the packages that contain them, are listed below. If you don't have any of these packages already, make sure to install them!

Dataset	Package
ACTG175	speff2trial
diamonds	ggplot2
Davis	car

Dataset	Package
heartdisease	FFTrees

2. Load the `tidyverse` package.

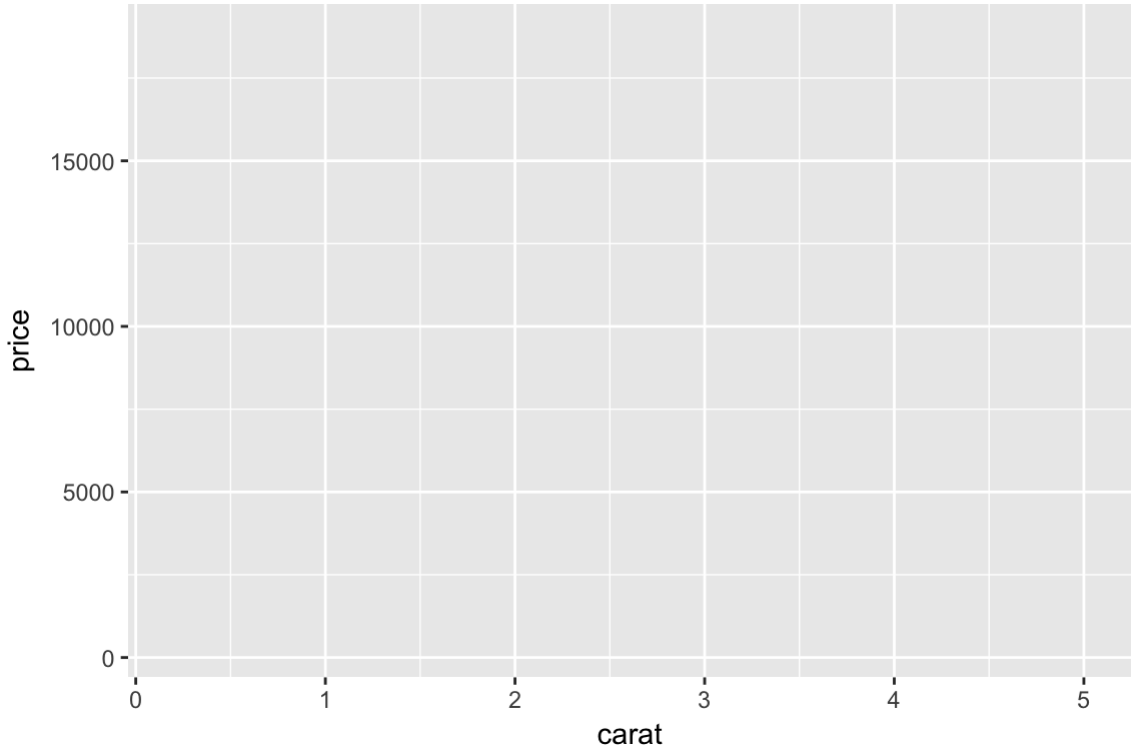
Building a plot step-by-step

3. The `diamonds` dataset in the `ggplot2` package shows information about 50,000 round cut diamonds. Print the `diamonds` dataset, it should look like this:

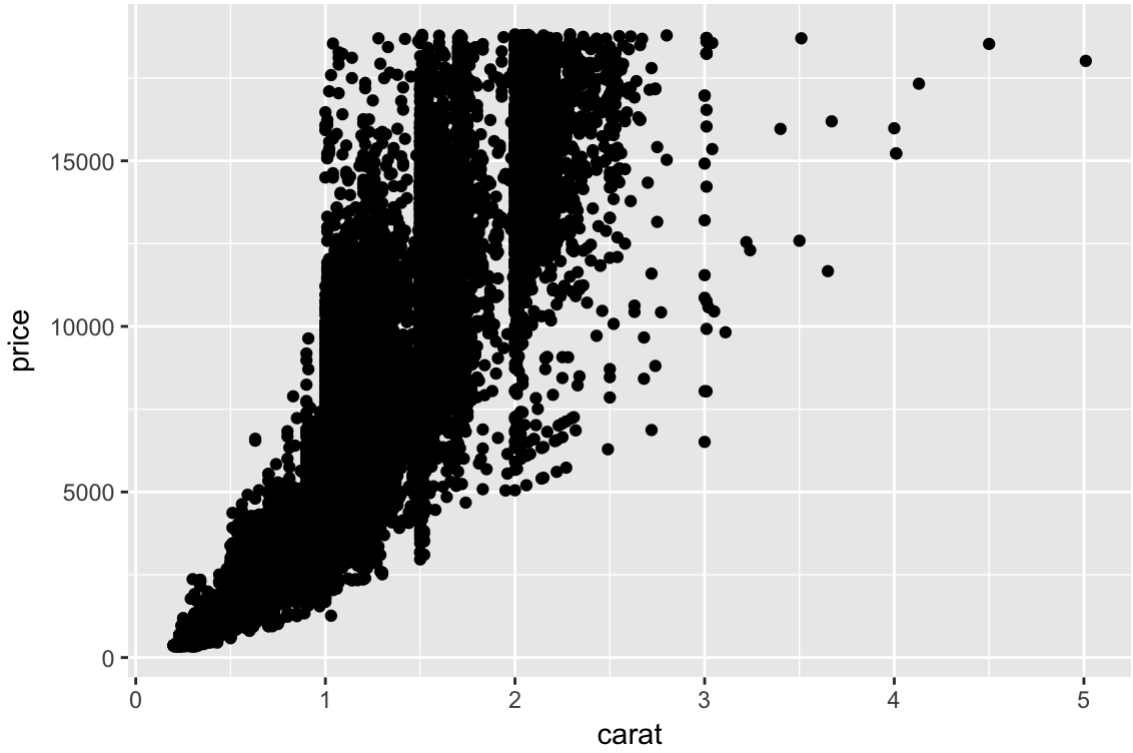
```
diamonds
```

```
# A tibble: 53,940 x 10
   carat       cut color clarity depth table price     x     y     z
   <dbl>     <ord> <ord>   <ord> <dbl> <dbl> <int> <dbl> <dbl> <dbl>
 1  0.23     Ideal     E     SI2  61.5    55   326  3.95  3.98  2.43
 2  0.21   Premium     E     SI1  59.8    61   326  3.89  3.84  2.31
 3  0.23      Good     E     VS1  56.9    65   327  4.05  4.07  2.31
 4  0.29   Premium     I     VS2  62.4    58   334  4.20  4.23  2.63
 5  0.31      Good     J     SI2  63.3    58   335  4.34  4.35  2.75
 6  0.24 Very Good     J    VVS2  62.8    57   336  3.94  3.96  2.48
 7  0.24 Very Good     I    VVS1  62.3    57   336  3.95  3.98  2.47
 8  0.26 Very Good     H     SI1  61.9    55   337  4.07  4.11  2.53
 9  0.22      Fair     E     VS2  65.1    61   337  3.87  3.78  2.49
10  0.23 Very Good     H     VS1  59.4    61   338  4.00  4.05  2.39
# ... with 53,930 more rows
```

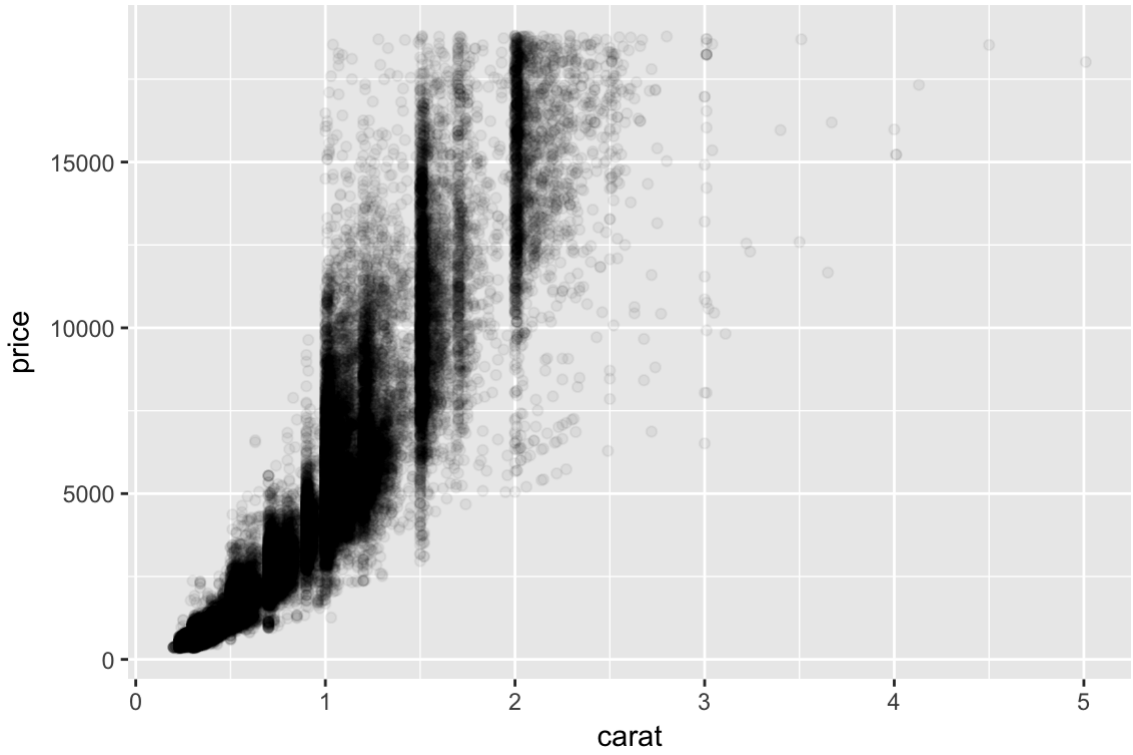
4. Create the following blank plot



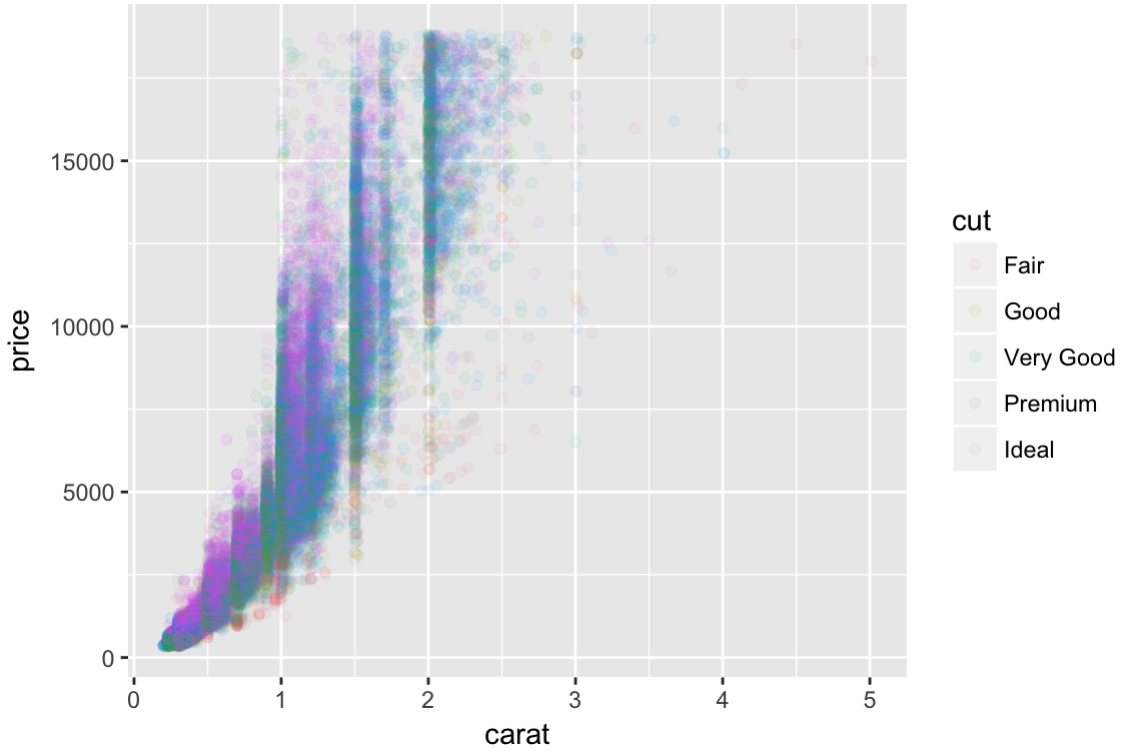
5. Now add points showing the relationship between the number of carats in the diamonds (`carat`) and its price (`price`)



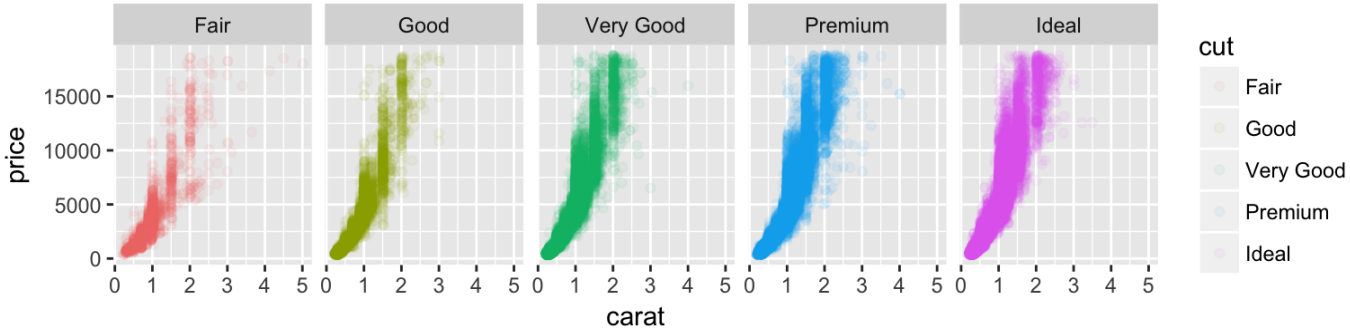
6. Make the points transparent using the `alpha` argument to `geom_point()`



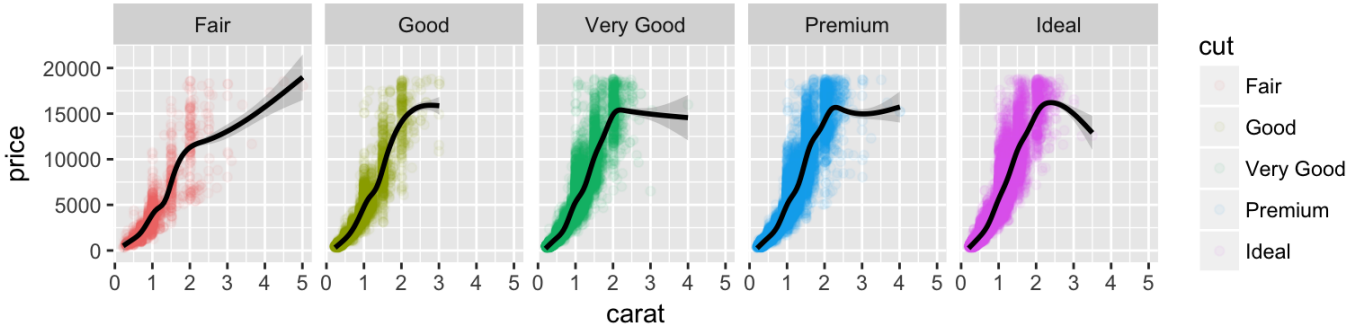
7. Color the points by their cut



8. Create different plots for each value of `cut` using the `facet_wrap` function:



9. Add a black, smoothed mean line to each plot using `geom_smooth()` (You can also try turning the line into a regression line using the `method` argument)

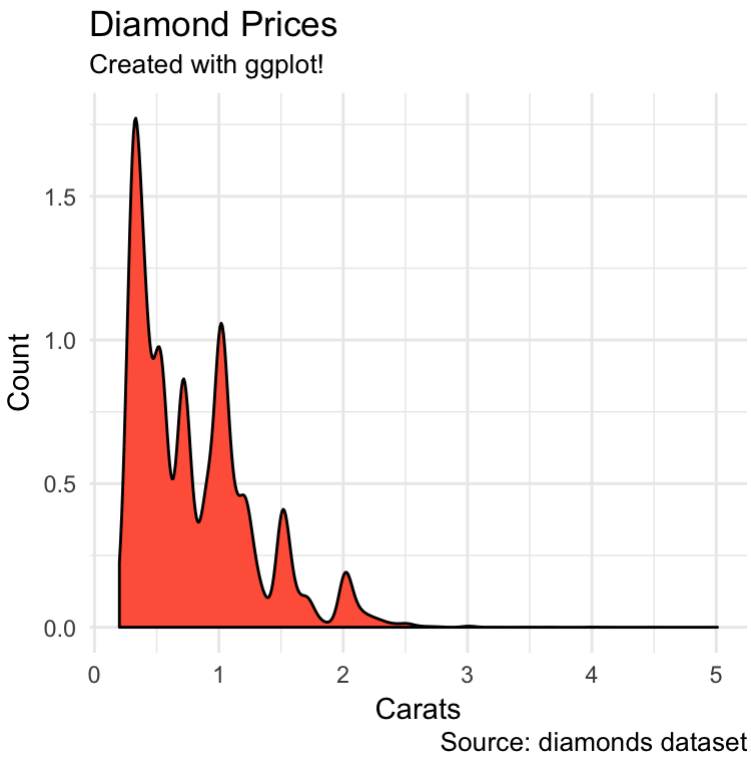


Density geom with `geom_density()`

10. Create the following density plot of prices from the `diamonds` data using the following template:

- Set the `data` argument to `diamonds`
- Map `carat` to the x aesthetic
- Add a density geom with `geom_density()` and set the fill to `"tomato1"`
- Add labels
- Use the minimal theme with `theme_minimal()`

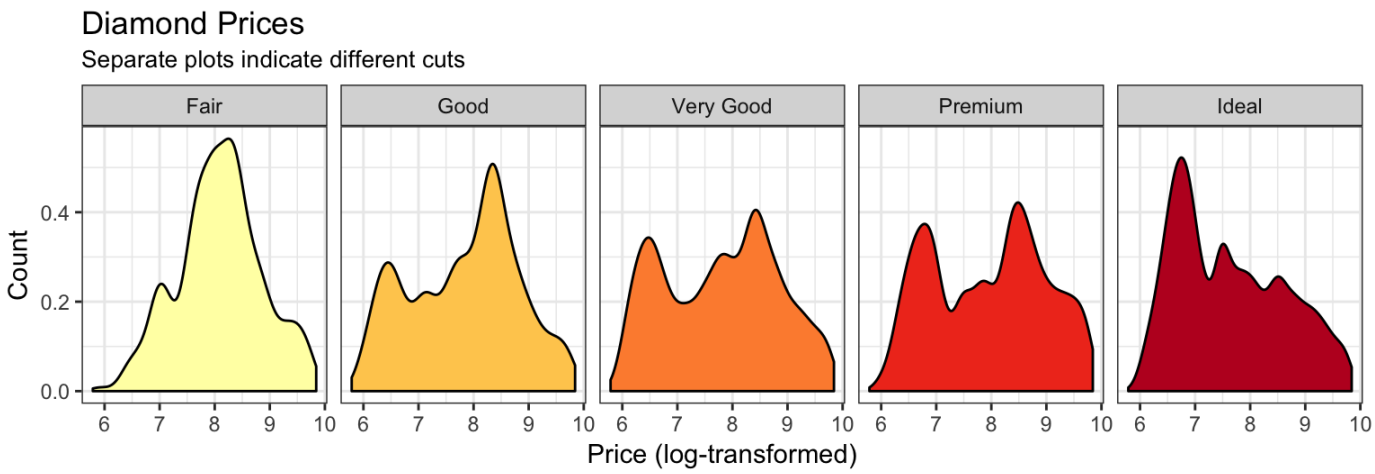
```
ggplot(data = XX,
       mapping = aes(x = XX)) +
       geom_density(fill = "XX") +
       labs(x = "XX",
            y = "XX",
            title = "XX",
            subtitle = "XX",
            caption = "XX") +
       theme_minimal()
```



11. Create the following densities of different facets by the diamond `cut`, and use different colors for each plot.

- Map `log(price)` (the logarithm of price) to the x aesthetic and `cut` to the fill aesthetic
- Create the density geom with `geom_density()` and add a black border color with the `color` argument
- Facet the plot by `cut`
- Add labels with `labs(x, y, title, subtitle, caption)`
- Using `scale_fill_brewer()`, change the color palette for the geom fillings to `"YlOrRd"`
- Use the black and white theme with `theme_bw()`
- Turn off the legend with `theme(legend.position = "none")`

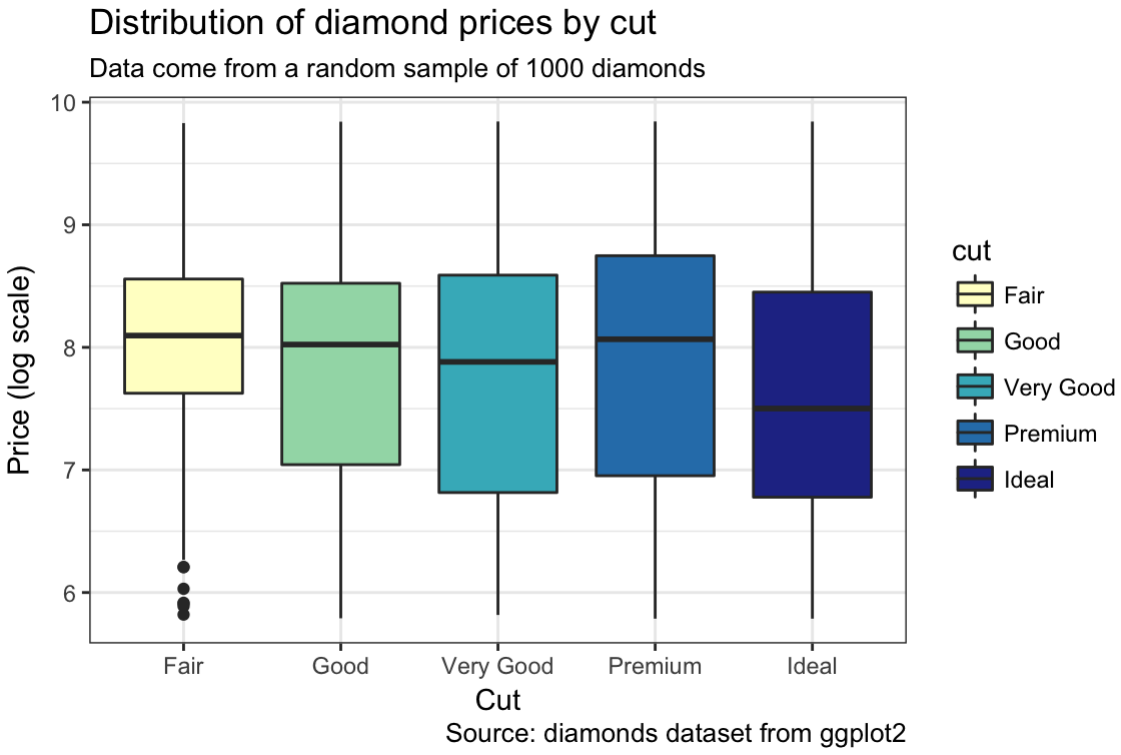
```
ggplot(data = XX,
       mapping = aes(x = log(XX), fill = XX)) +
       geom_density(color = "XX") +
       facet_wrap(~ XX, nrow = 1) +
       labs(x = "XX",
            y = "XX",
            title = "XX",
            subtitle = "XX",
            caption = "XX") +
       scale_fill_brewer(palette = "XX") +
       theme_bw() +
       theme(legend.position = "none")
```



Boxplot geom `geom_boxplot()`

12. Look at the help menu for `geom_boxplot()`. Then, create the following boxplot using the following template

```
ggplot(data = XX,
  mapping = aes(x = XX, y = log(XX), fill = XX)) +
  geom_boxplot()  +
  labs(y = "XX",
       x = "XX",
       color = "XX",
       title = "XX",
       subtitle = "XX") +
  scale_fill_brewer(palette = "XX") +
  theme_bw()
```



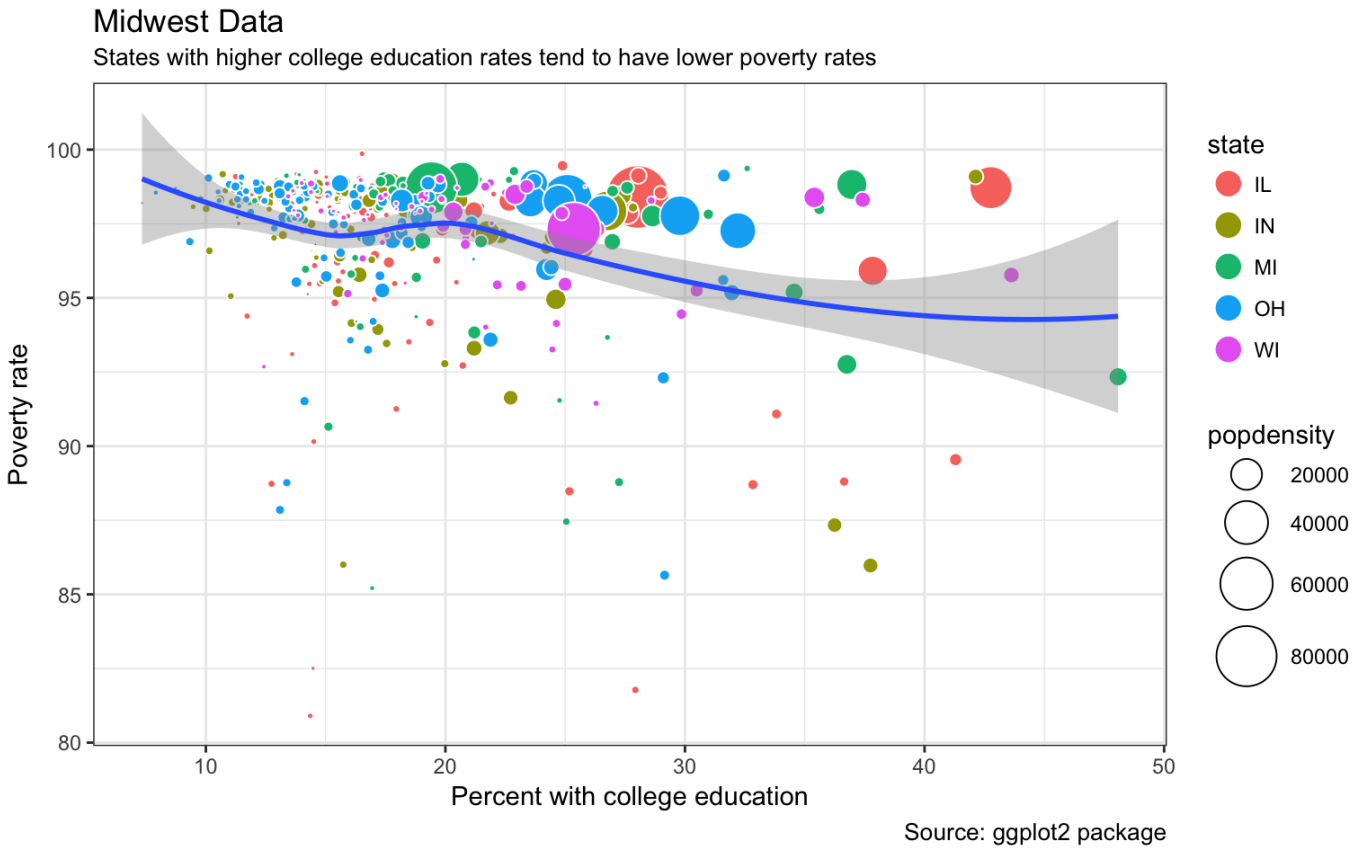
Demographic information of midwest counties in the US

13. Print the `midwest` dataset and look at the help menu to see what values it contains. It should look like this:

```
# A tibble: 437 x 28
     PID    county state  area poptotal popdensity popwhite popblack
   <int>     <chr> <chr> <dbl>    <int>      <dbl>    <int>    <int>
 1   561     ADAMS    IL 0.052    66090  1270.9615    63917     1702
 2   562 ALEXANDER    IL 0.014    10626   759.0000     7054     3496
 3   563      BOND    IL 0.022    14991   681.4091    14477      429
 4   564     BOONE    IL 0.017    30806  1812.1176    29344      127
 5   565     BROWN    IL 0.018     5836   324.2222     5264      547
 6   566    BUREAU    IL 0.050    35688   713.7600    35157       50
 7   567   CALHOUN    IL 0.017     5322   313.0588     5298        1
 8   568   CARROLL    IL 0.027    16805   622.4074    16519      111
 9   569      CASS    IL 0.024    13437   559.8750    13384       16
10   570 CHAMPAIGN    IL 0.058   173025  2983.1897   146506    16559
# ... with 427 more rows, and 20 more variables: popamerindian <int>,
#   popasian <int>, popother <int>, percwhite <dbl>, percblack <dbl>,
#   percamerindan <dbl>, percasian <dbl>, percother <dbl>,
#   popadults <int>, perchsd <dbl>, percollege <dbl>, percprof <dbl>,
#   poppovertyknown <int>, percpovertyknown <dbl>, percbelowpoverty <dbl>,
#   percchildbelowpovert <dbl>, percadultpoverty <dbl>,
#   percelderlypoverty <dbl>, inmetro <int>, category <chr>
```

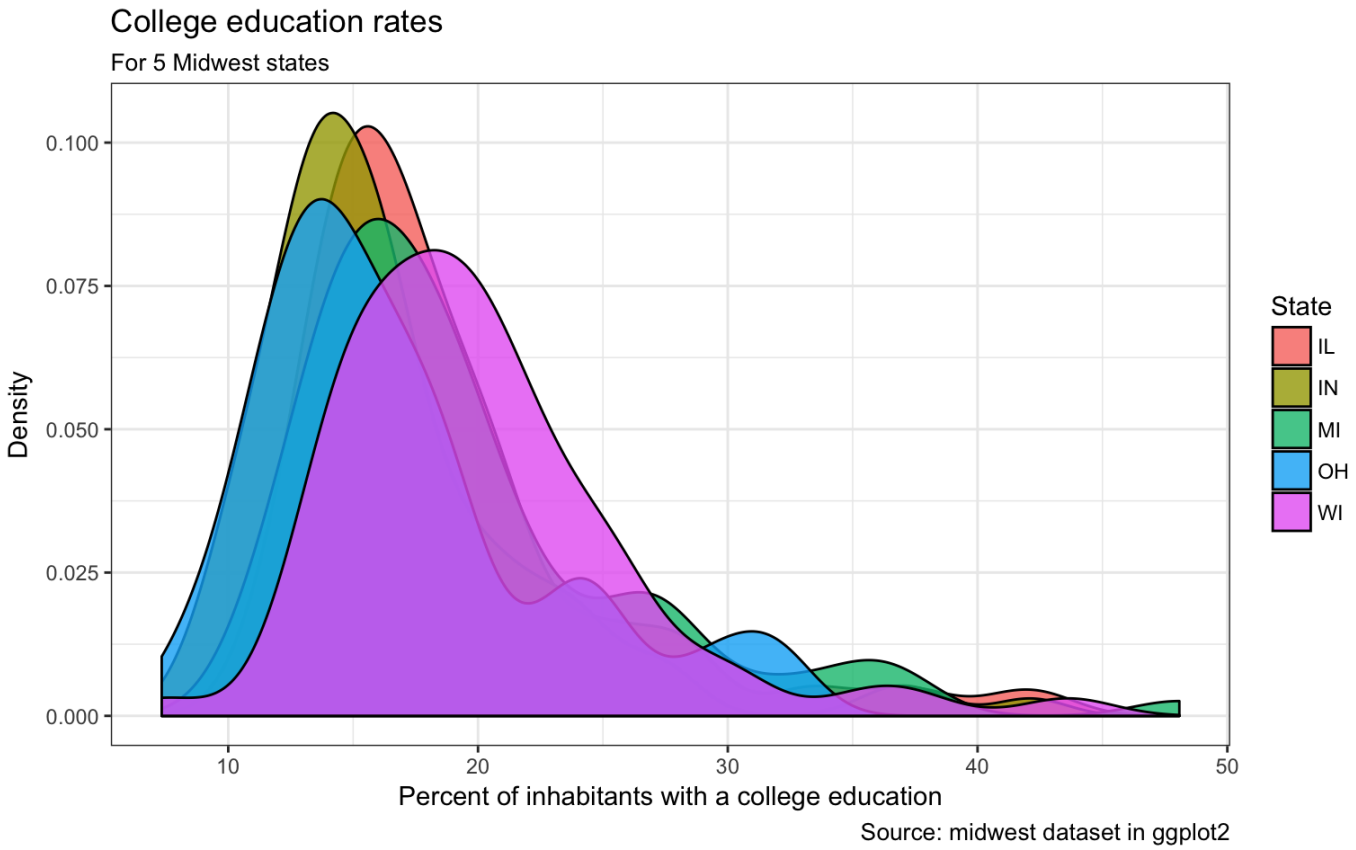
14. Using the following code as a template, create the Following plot showing the relationship between college education and poverty

```
ggplot(data = XX,
    mapping = aes(x = XX, y = XX)) +
    geom_point(aes(fill = XX, size = XX), shape = 21, color = "white") +
    geom_smooth(aes(x = XX, y = XX)) +
    labs(
        x = "XX",
        y = "XX",
        title = "XX",
        subtitle = "XX",
        caption = "XX") +
    scale_color_brewer(palette = "XX") +
    scale_size(range = c(XX, XX)) +
    guides(size = guide_legend(override.aes = list(col = "black")),
           fill = guide_legend(override.aes = list(size = 5))) +
    theme_bw()
```



15. Create the following density plot showing the density of inhabitants with a college education in different states using the following template

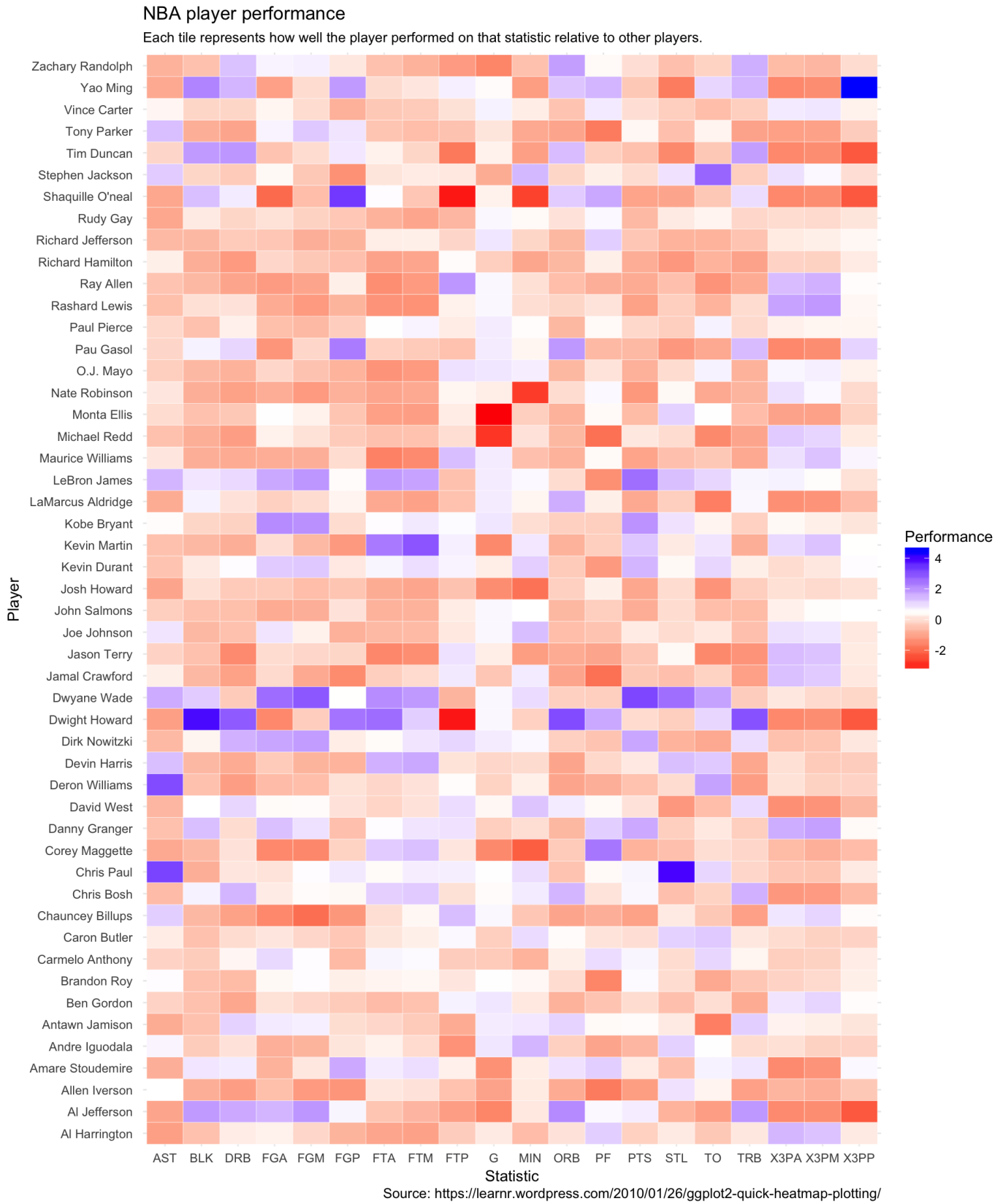
```
ggplot(data = XX,
       mapping = aes(XX, fill = XX)) +
  geom_density(alpha = XX) +
  labs(title = "XX",
       subtitle = "XX",
       caption = "XX",
       x = "XX",
       y = "XX",
       fill = "XX") +
  theme_bw()
```



Heatplots with `geom_tile()`

16. You can create heatplots using the `geom_tile()` function. Try creating the following heatplot of statistics of NBA players using the following template:

```
# Read in nba data
nba_long <- read.csv("https://raw.githubusercontent.com/therbootcamp/therbootcamp.github.io/master/_sessions/_data/nba_long.csv")

ggplot(XX,
       mapping = aes(x = XX, y = XX, fill = XX)) +
  geom_tile(colour = "XX") +
  scale_fill_gradientn(colors = c("XX", "XX", "XX"))+
  labs(x = "XX",
       y = "XX",
       fill = "XX",
       title = "NBA XX performance",
       subtitle = "XX",
       caption = "XX") +
  coord_flip() +
  theme_minimal()
```



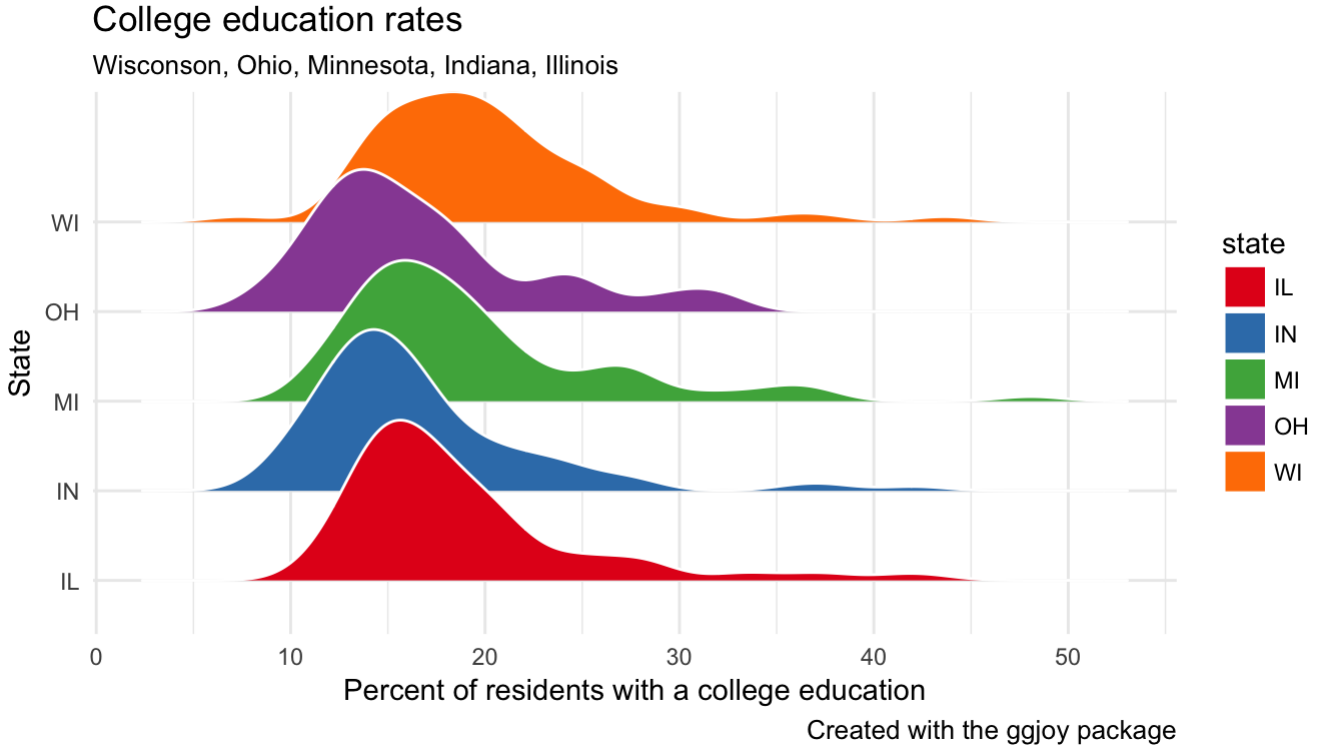
Joyplots with `ggjoy()`

17. The `ggjoy` package contains a new geom called `geom_joy()` that creates a joyplot (https://eagereyes.org/blog/2017/joy-plots). Install the `ggjoy` package, load it, and then look at the introductory vignette by running `vignette(topic = "introduction", package = "ggjoy")`,

```
# Install the ggjoy package from CRAN
install.packages("ggjoy")

# Load the package
library("ggjoy")

# Open the introduction vignette
vignette(topic = "introduction", package = "ggjoy")

# Open the gallery vignette
vignette(topic = "gallery", package = "ggjoy")
```

18. Now, create the following joyplot of college education rates from the `midwest` data using the following template:

```
library(ggjoy)

ggplot(data = XX,
       mapping = aes(XX, y = XX, fill = XX)) +
  geom_joy(col = "XX") +
  labs(title = "XX",
       subtitle = "XX",
       caption = "XX",
       x = "XX",
       y = "XX") +
  scale_fill_brewer(palette = "XX") +
  theme_minimal()
```



Correlation plot with ggcorplot

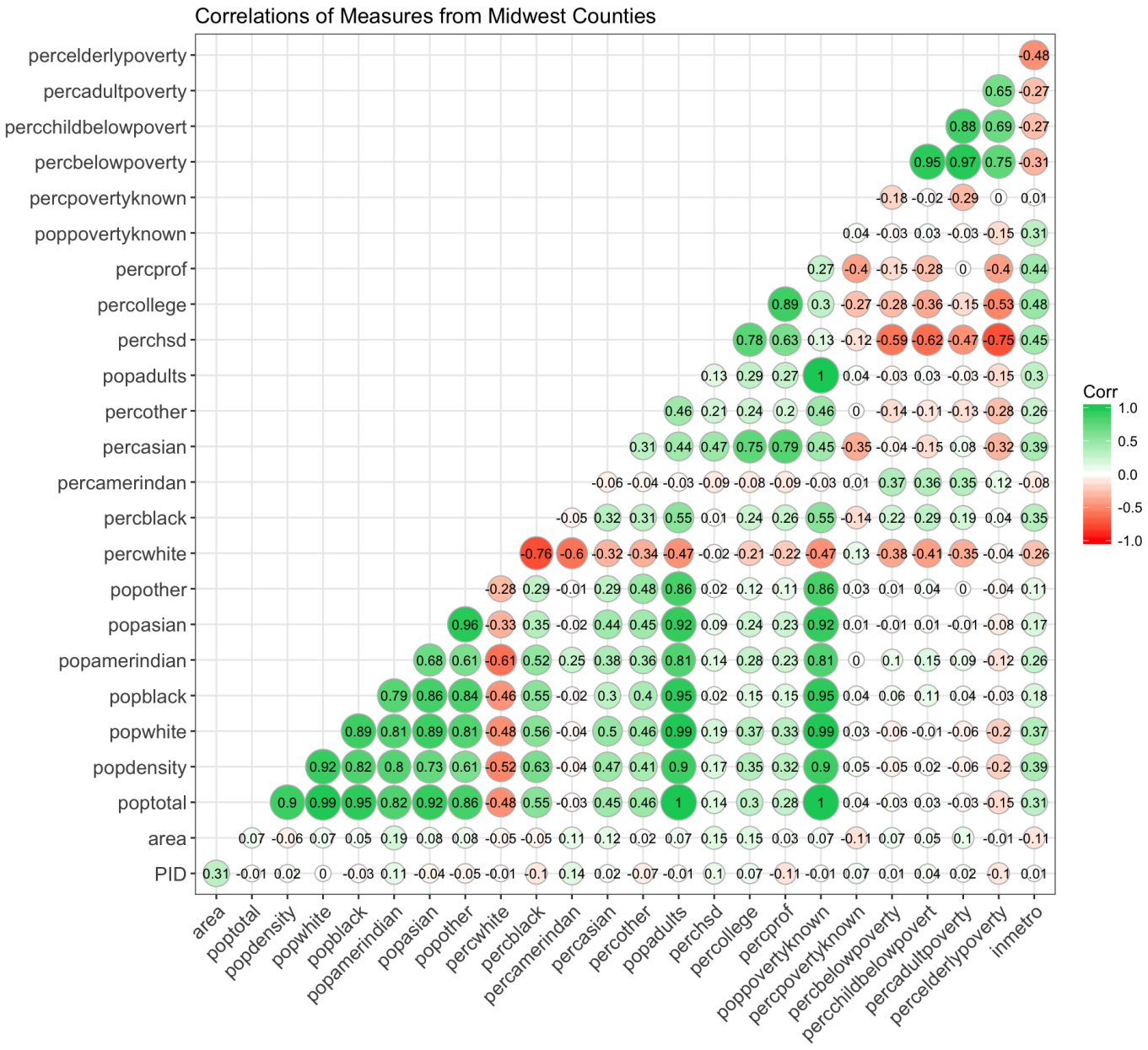
19. The `ggcorplot` package lets you make really nice plots of correlation matrices. Install the package from github with the following code:

```
# Install the ggcorrplot package

#install.packages("devtools")   # If you don't have the devtools pacakge, you'll have
 to install it first
devtools::install_github("kassambara/ggcorrplot")
```

20. Using the following template, try to make your own correlation plot from the `midwest` data that looks like this:

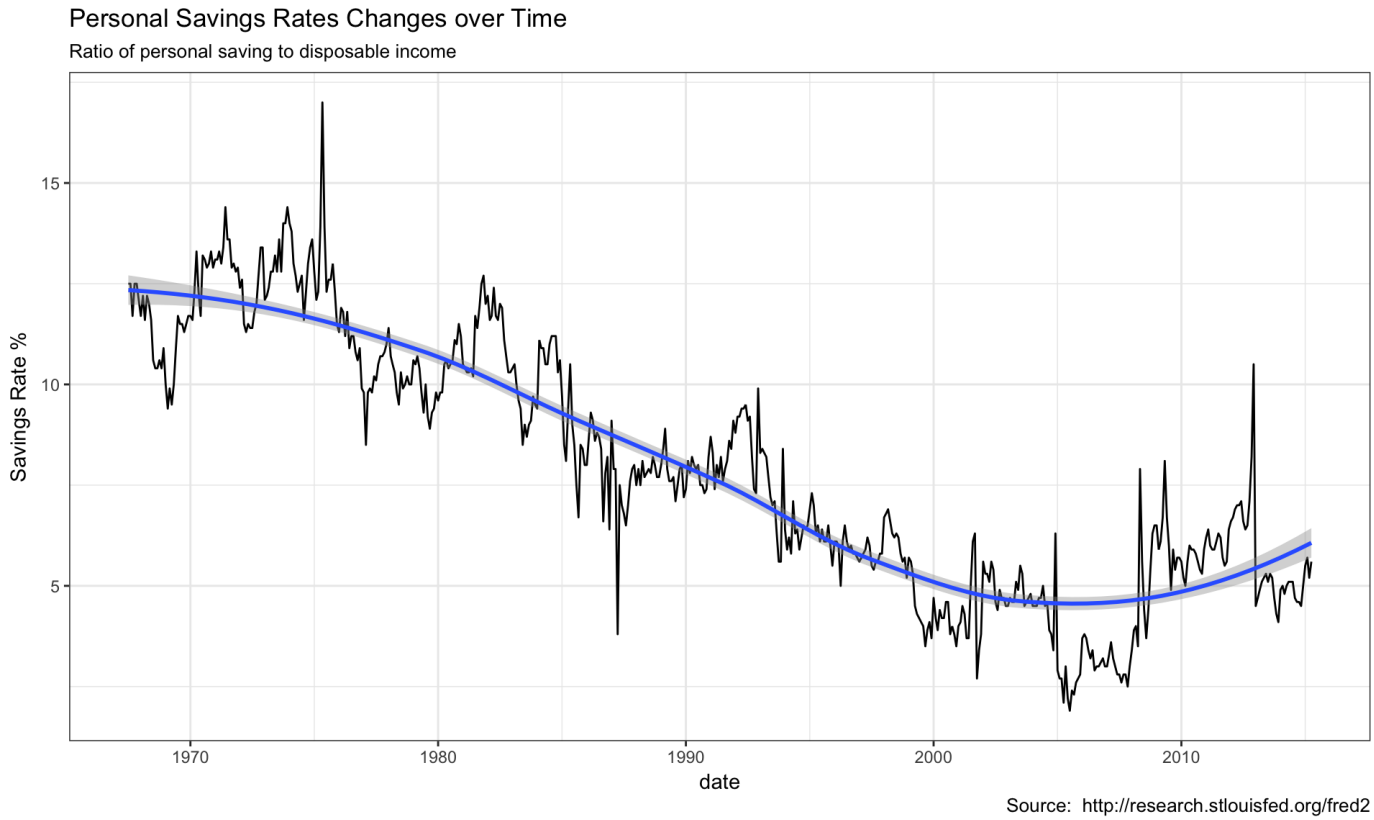
- First create a correlation matrix using all of the numeric columns in the `midwest` data seen in the plot.
- Then, create the plot with `ggcorrplot`

```
library(ggcorrplot)

# Select only numeric rows
midwest_num <- midwest %>% select_if(is.numeric)

# Create a correlation matrix from several numeric columns
corr_mtx <- cor(midwest_num)

# Create the correlation plot!
ggcorrplot(cor = XX,
           type = "XX",
           lab = TRUE,
           lab_size = 3,
           method = "X",
           colors = c("X", "X", "X"),
           title = "XX",
           subtitle = "XX",
           caption = "XX",
           ggtheme = XX)
```

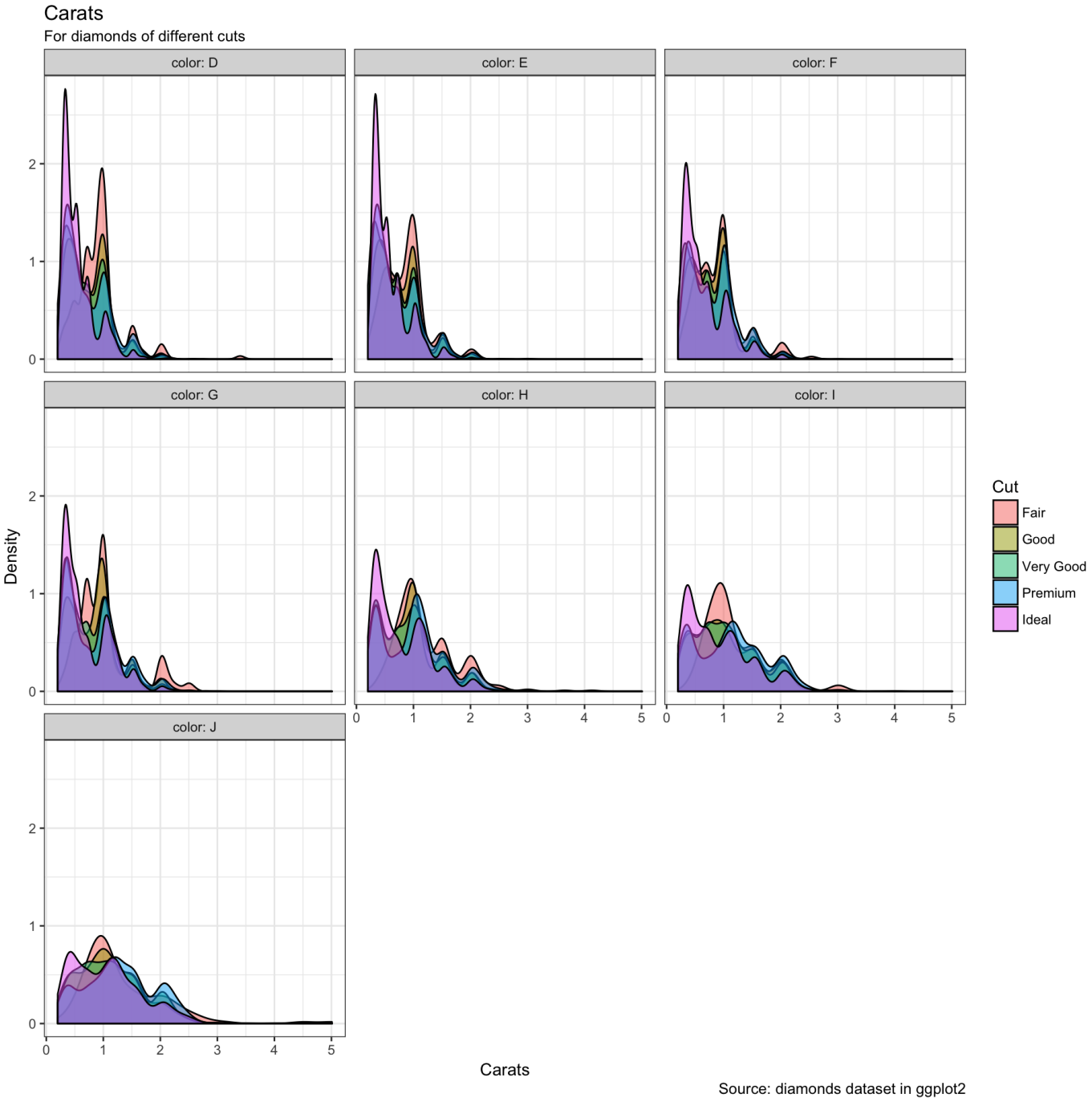


Challenges

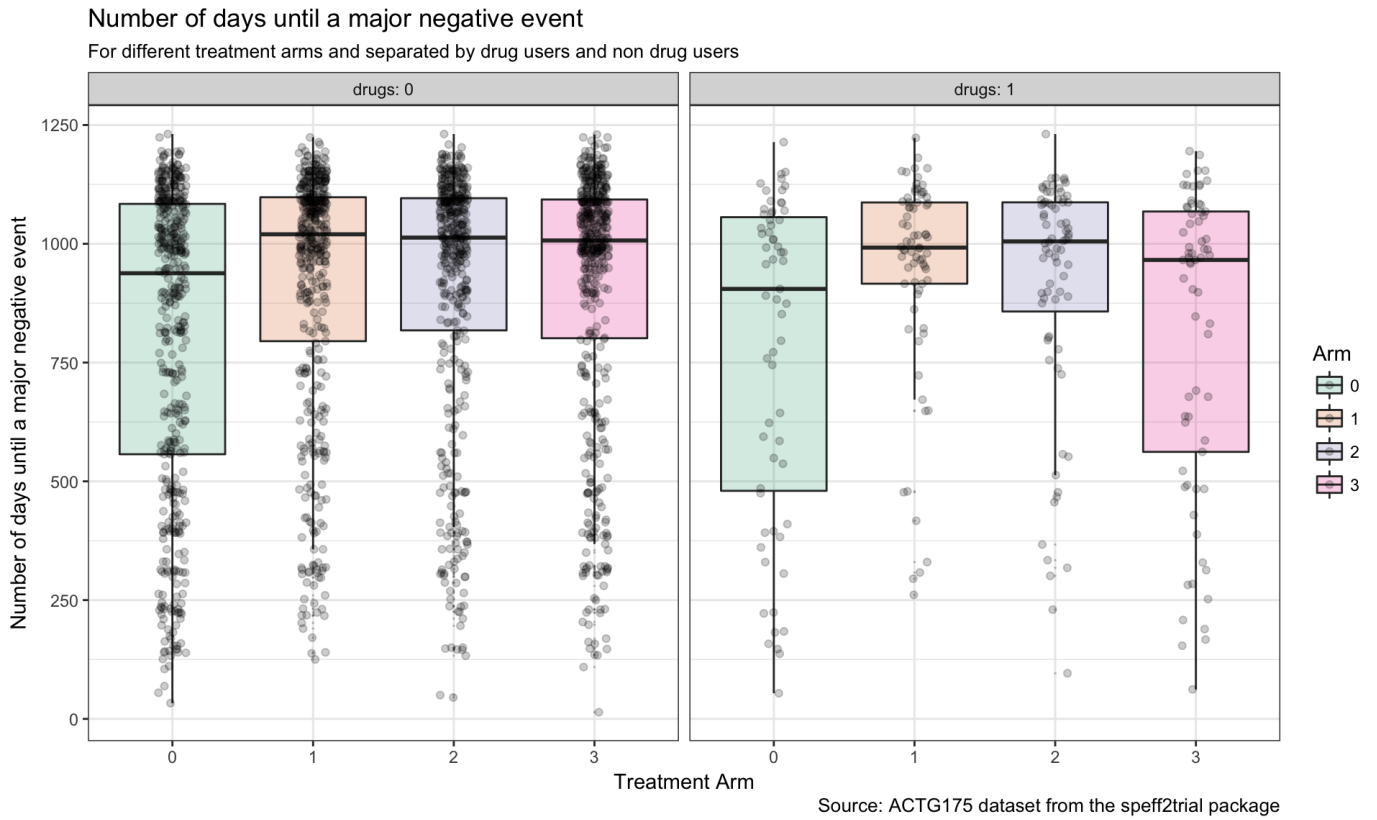
21. Make the following plot of savings data (`psavert`) from the `economics` dataset (hint: use the `geom_line()` function and map `date` to the x aesthetic and `psavert` to the y aesthetic.)



22. Make this plot from the `diamonds` data showing the relationship between diamond cut (`cut`), color (`color`), and carats (`carat`)

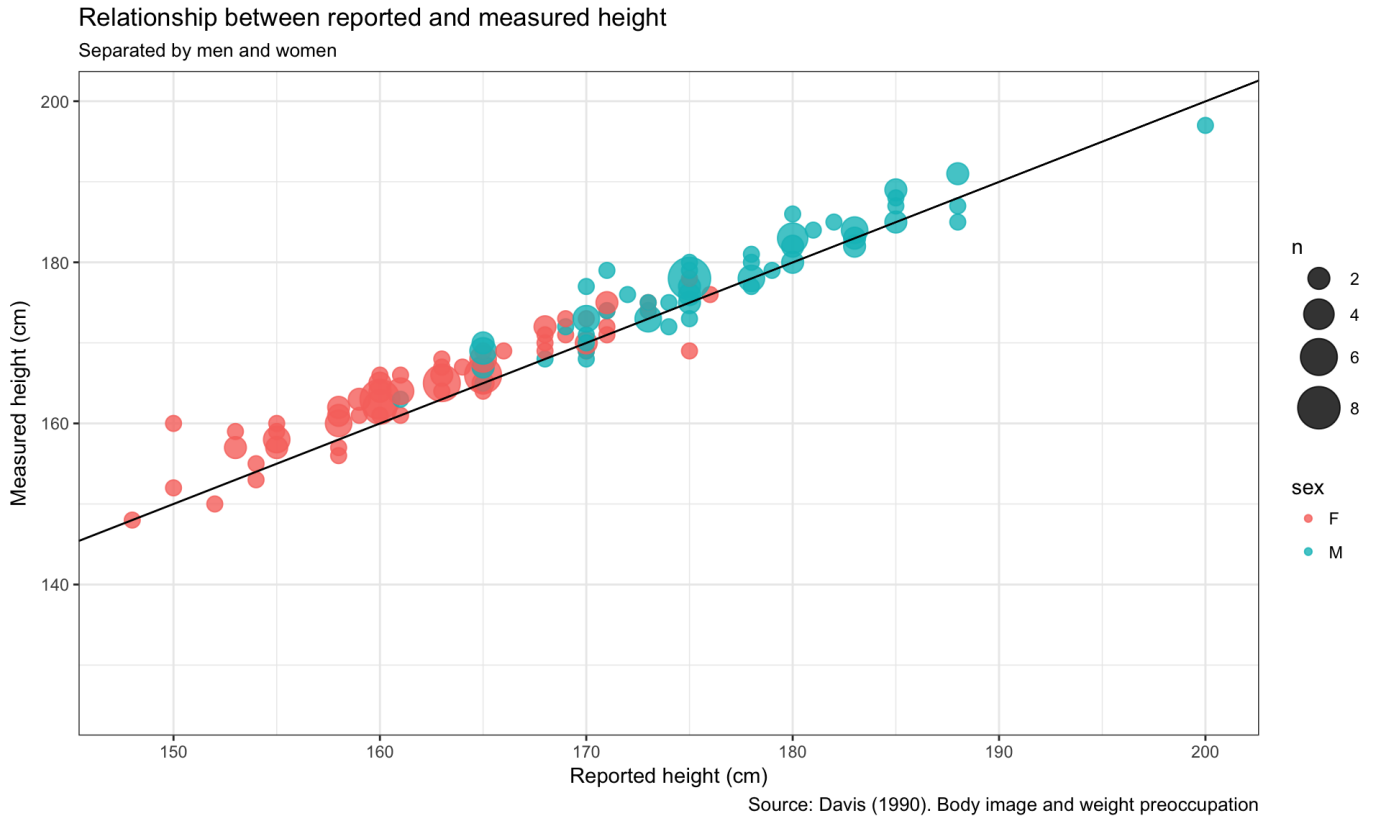


23. Make the following plot from the `ACTG175` dataset. To do this, you'll need to use both `geom_boxplot()` and `geom_point()`. To jitter the points, use the `position` argument to `geom_point()`, as well as the `position_jitter()` function to control how much to jitter the points.



24. The `Davis` dataframe in the `car` package contains measurements of the reported and measured (i.e.; real!) heights and weights of several men and women. Create the following plot showing the relationship between people's reported, and actual heights.

- Instead of using `geom_point()`, use `geom_count()` to create points whose size reflects the number of observations at that location.
- Include an "identity line" with the `geom_abline()` function.



You choose the plot!

`ACTG175` (From the `speff2trial` package)

25. Create a plot showing the relationship between CD4 T cell count at 96 weeks (`cd496`), treatment arm (`arms`) and gender (`gender`).
26. Create a plot showing the relationship between treatment arm (`arms`), number of days until a major negative event (`days`) and history of intravenous drug use (`drugs`).

`heartdisease` (From the `FFTrees` package)

27. Create a plot showing the relationship between `age` , `chol` and `diagnosis`
28. Create a plot showing the relationship between `cp` , `slope` , and `diagnosis`

References

- Many of the plots in this practical were taken from Selva Prabhakaran's website http://r-statistics.co/Top50-Ggplot2-Visualizations-MasterList-R-Code.html (http://r-statistics.co/Top50-Ggplot2-Visualizations-MasterList-R-Code.html)
- For making maps with ggplot, check out Eric Anderson's tutorial at http://eriqande.github.io/rep-res-web/lectures/making-maps-with-R.html (http://eriqande.github.io/rep-res-web/lectures/making-maps-with-R.html)