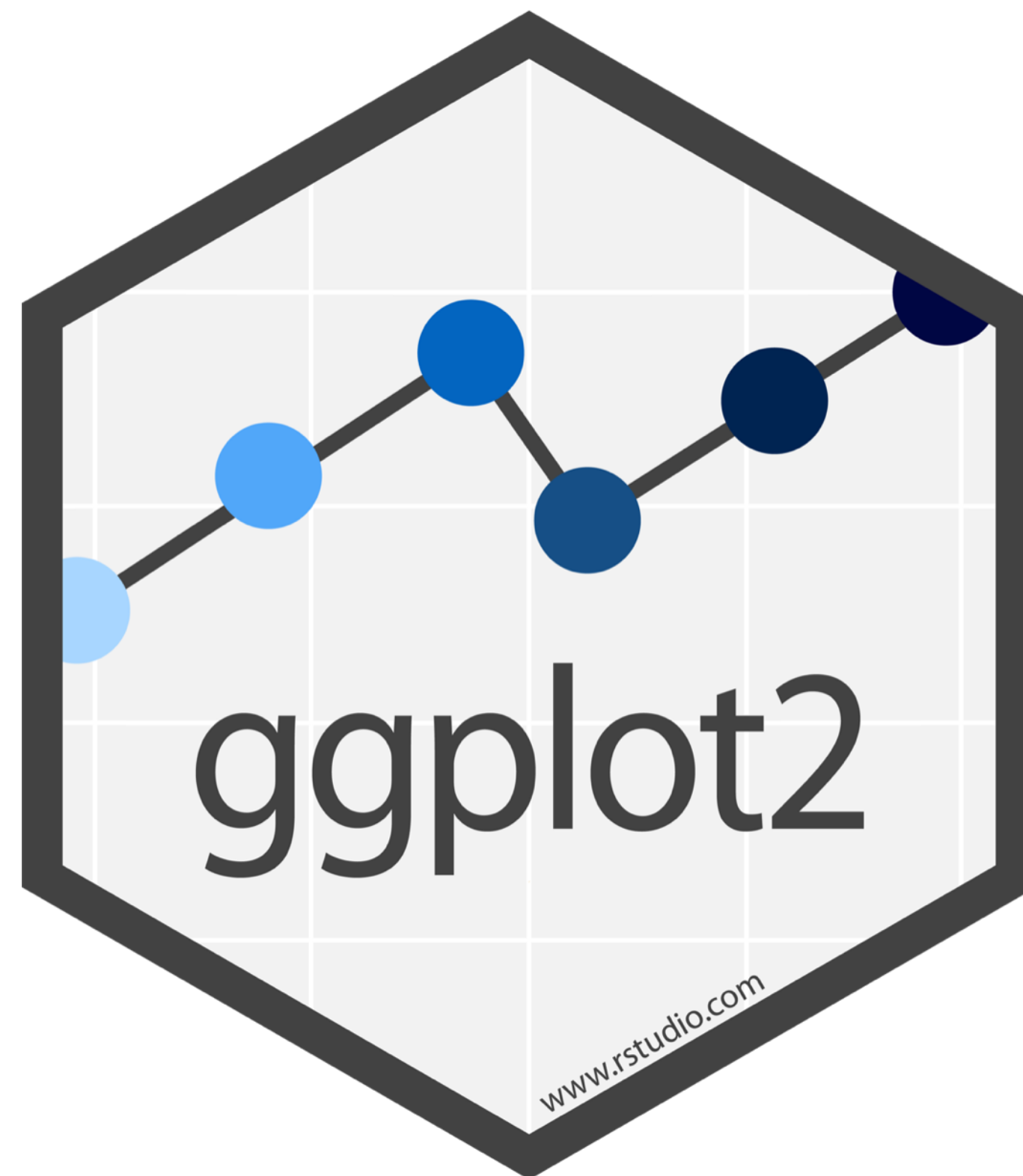


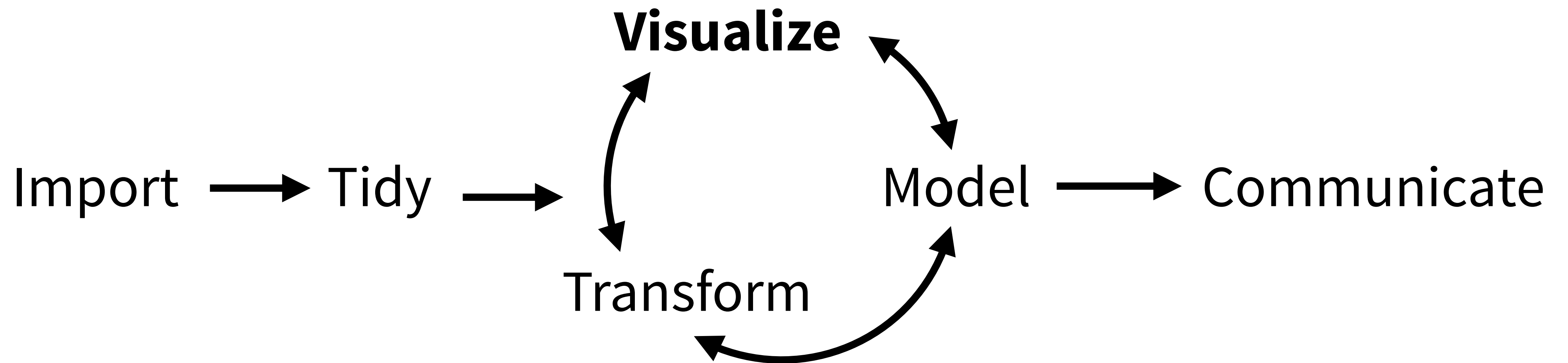
Visualize Data with



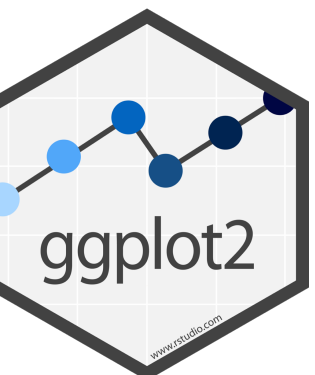
"The simple graph has brought more information to the data analyst's mind than any other device."

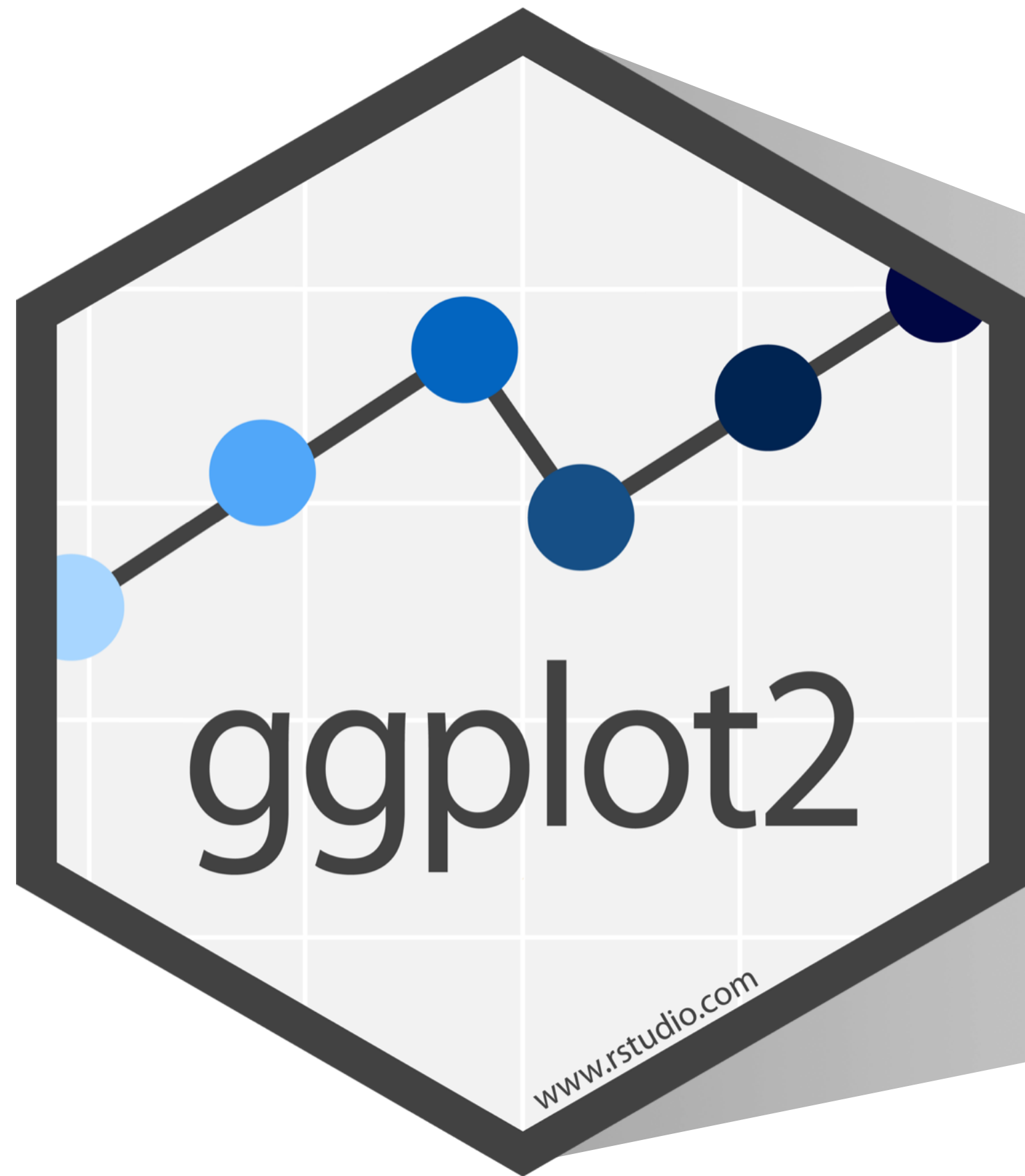
- John Tukey

(Applied) Data Science



Program





hflights

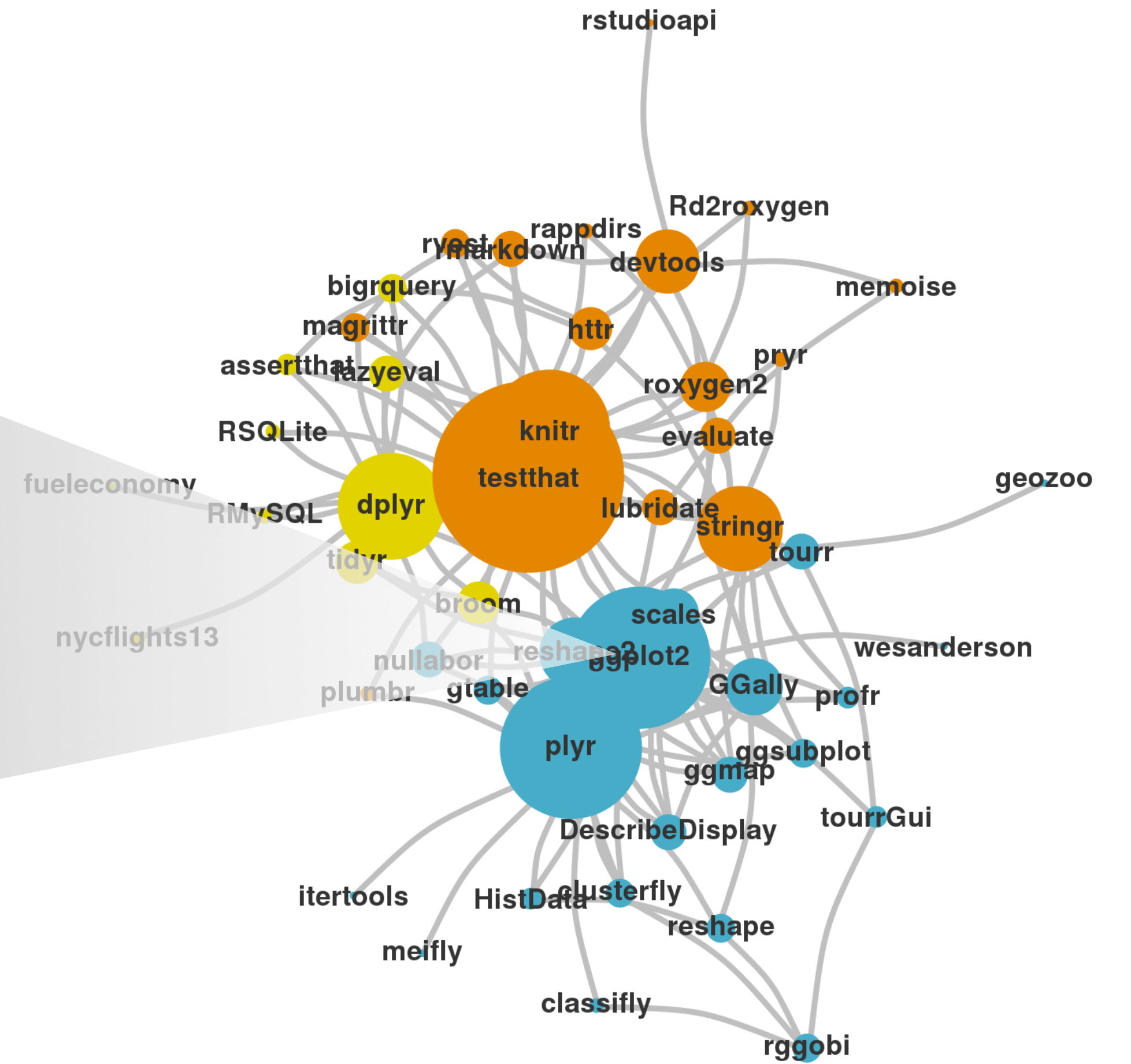
namespace

nasaweather

babynames

fda

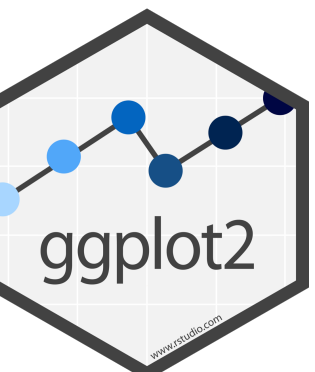
plotrix



The image shows a screenshot of the RStudio interface. The main editor window displays an R Notebook file named "02-Visualize-Data.Rmd". The code is written in R Markdown format and includes a title, output options, a setup section, and a chunk of R code to load the ggplot2 and fivethirtyeight packages. The right-hand pane shows the Environment, History, and Connections tabs, with the Global Environment selected. Below this, a file browser shows the directory structure: Home > Dropbox > Intro_to_R_and_RSTudio > Day1, with subfolders code, keynotes, and slides. The bottom status bar shows the time as 10:33 and the current chunk as "# Setup".

```
1 ---
2 title: "Visualize Data"
3 output:
4   html_document:
5     df_print: paged
6 ---
7
8 ## Setup
9
10 The first chunk in an R Notebook is usually titled
11 "setup," and by convention includes the R packages
12 you want to load. Remember, in order to use an R
13 package you have to run some `library()` code every
14 session. Execute these lines of code to load the
15 packages.
16
17 ## Bechdel test data
```

Open the R Notebook 01-Visualize.Rmd



Setup

The setup chunk is always run once before anything else

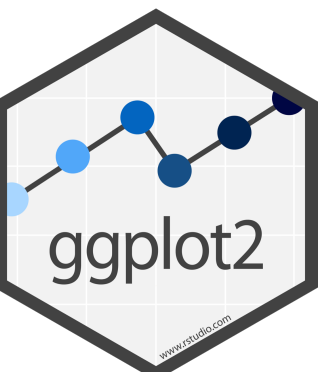


The screenshot shows an R Notebook editor window titled "01-Visualize-Data.Rmd". The code in the notebook is as follows:

```
1 ---
2 title: "Visualize Data"
3 output:
4   html_document:
5     df_print: paged
6 ---
7
8 ## Setup
9
10 The first chunk in an R Notebook is usually titled
11 "setup," and by convention includes the R packages
12 you want to load. Remember, in order to use an R
13 package you have to run some `library()` code every
14 session. Execute these lines of code to load the
15 packages.
16
17 ## Bechdel test data
```

A callout box points to the setup chunk code, containing the text: "(optional) label for chunk".

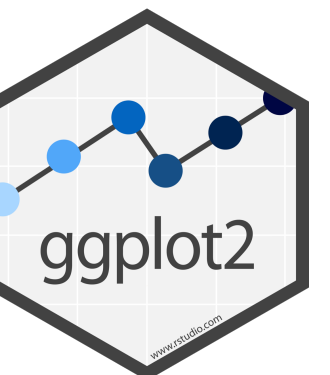
```
```{r setup}
library(ggplot2)
library(fivethirtyeight)
```
```



bechdel

Data on movies and the Bechtel test

```
bechdel  
?bechdel
```



Consider

Confer with the people around you.

What relationship do you expect to see between movie budget (`budget`) and domestic gross(`domgross`)?

01:00

Your Turn 1

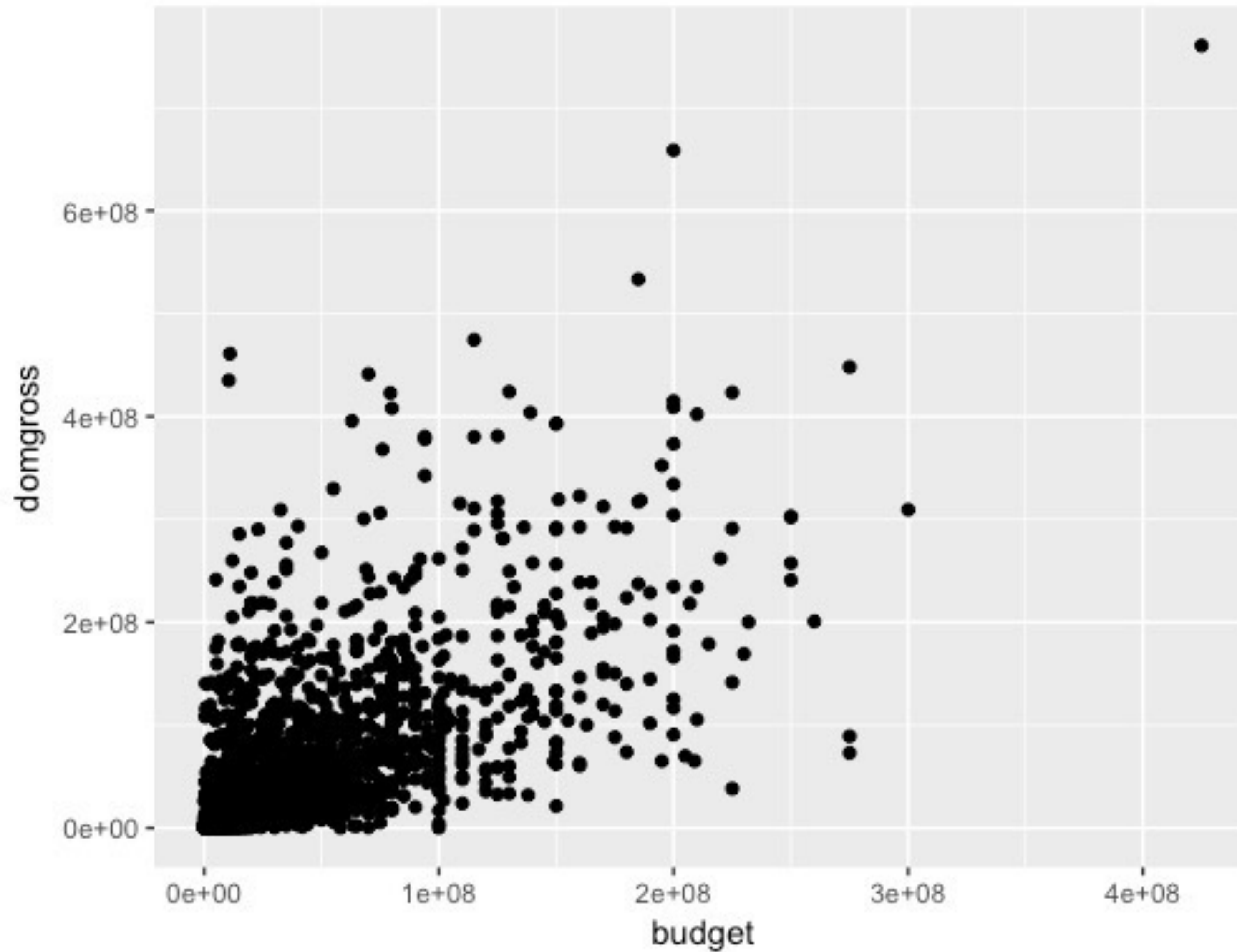
Run this code in your notebook to make a graph.

Pay strict attention to spelling, capitalization, and parentheses!

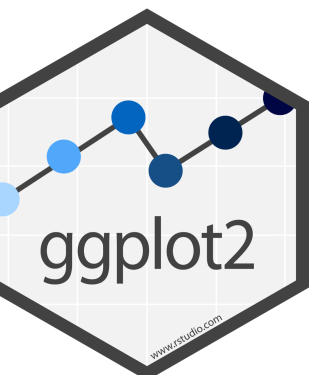
```
ggplot(data = bechdel) +  
  geom_point(mapping = aes(x = budget, y = domgross))
```



02:00



```
ggplot(data = bechdel) +  
  geom_point(mapping = aes(x = budget, y = domgross))
```

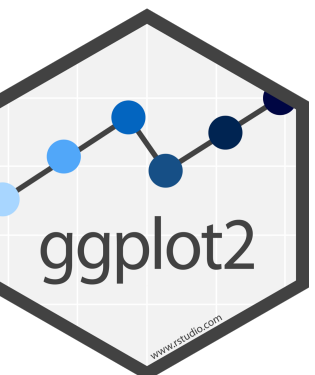


When you run this code, you will get what looks like an error, but is actually just a message from R. Some of the rows in the dataset didn't contain information for budget and/or domgross, so they're not plotted.



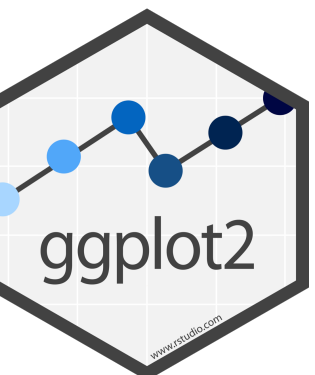
Removed 17 rows containing missing values (geom_point).

```
ggplot(data = bechdel) +  
  geom_point(mapping = aes(x = budget, y = domgross))
```



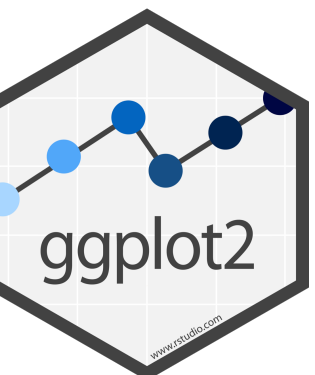
1. "Initialize" a plot with `ggplot()`
2. Add layers with `geom_` functions

```
ggplot(data = bechdel) +  
  geom_point(mapping = aes(x = budget, y = domgross))
```



Pro tip: Always put the +
at the end of a line,
Never at the start

```
ggplot(data = bechdel) +  
  geom_point(mapping = aes(x = budget, y = domgross))
```



data

+ before new line

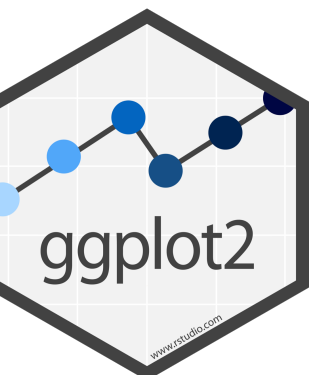
```
ggplot(data = bechdel) +  
  geom_point(mapping = aes(x = budget, y = domgross))
```

type of layer

aes()

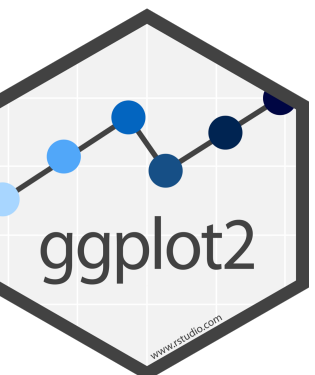
x variable

y variable



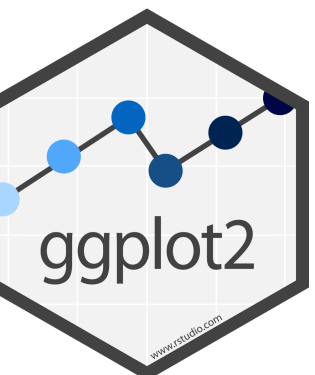
A template

```
ggplot(data = <DATA>) +  
  <GEOM_FUNCTION>(mapping = aes(<MAPPINGS>))  
  geom_point(mapping = aes(x = budget, y = domgross))
```



A template

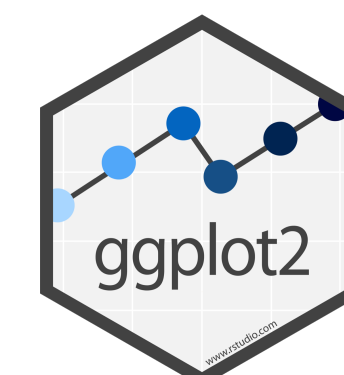
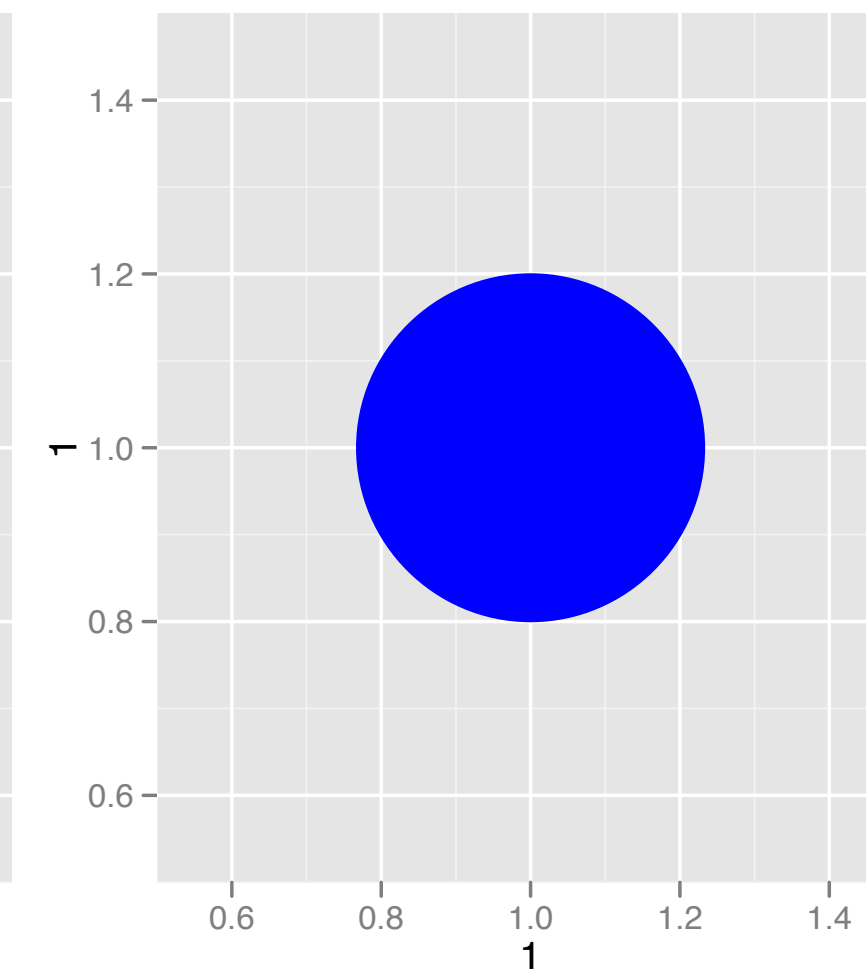
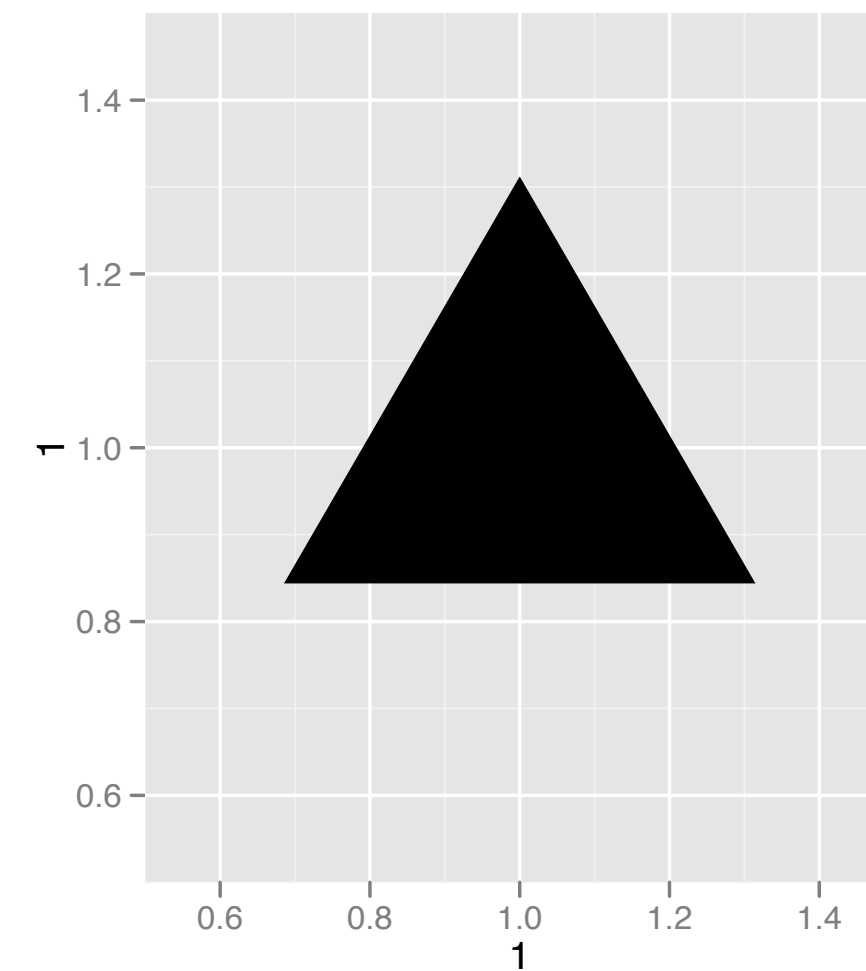
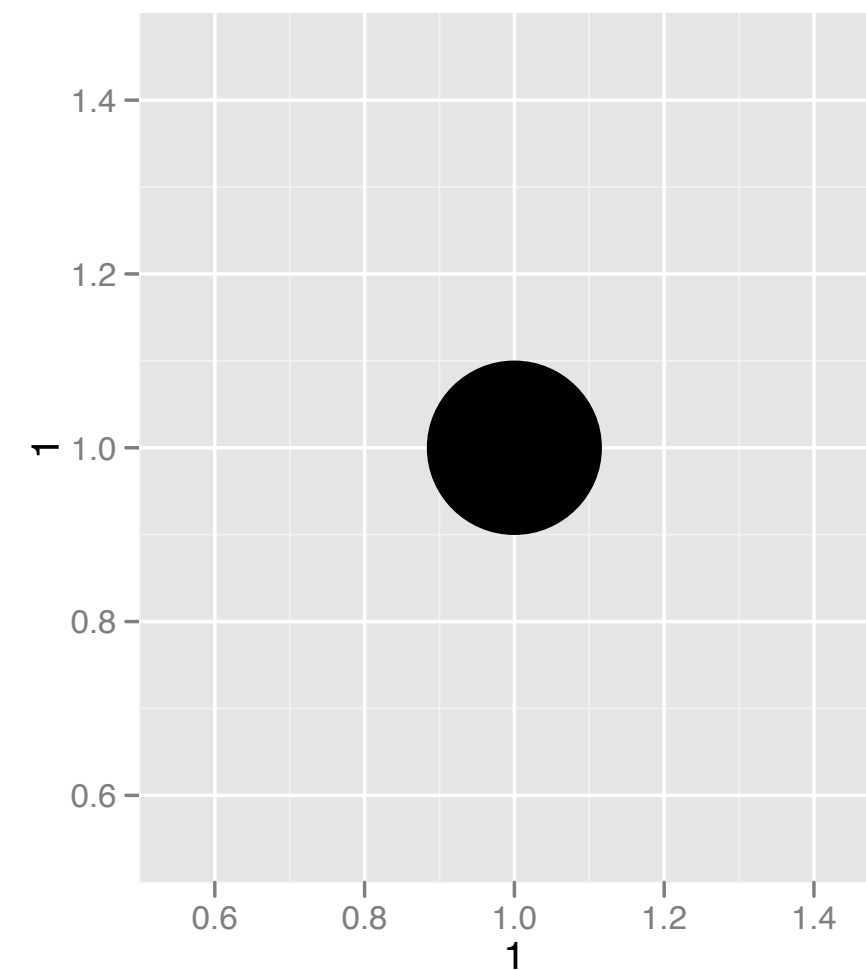
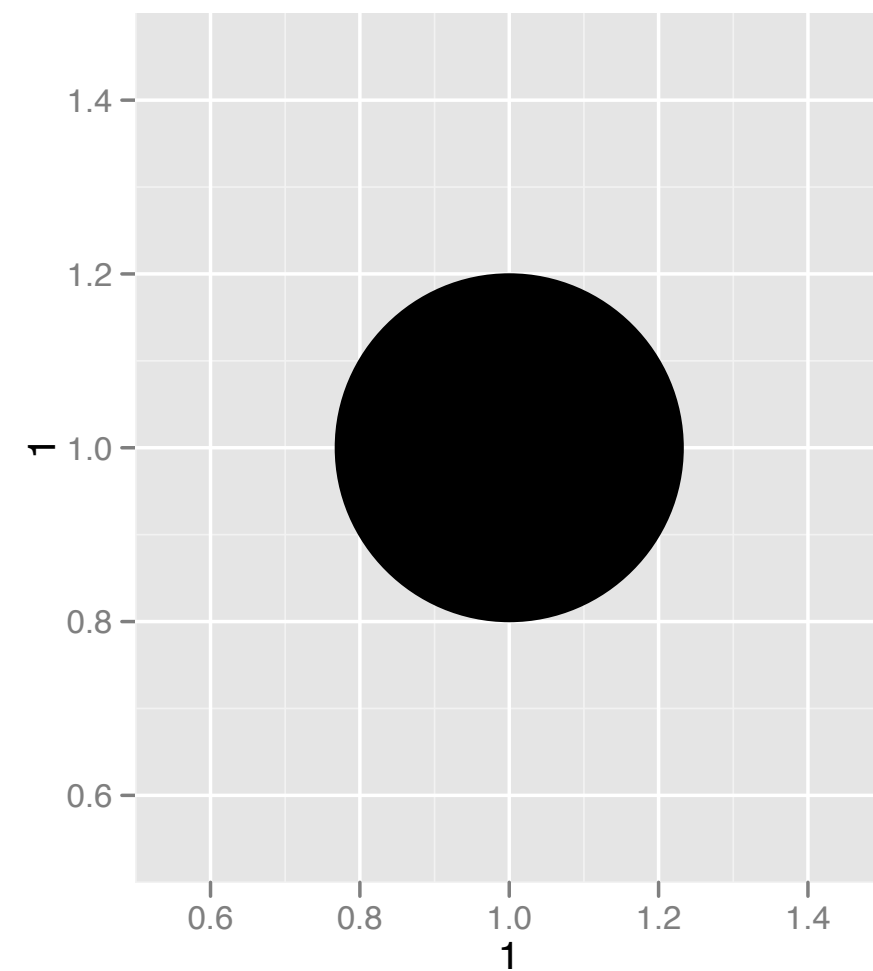
```
ggplot(data = <DATA>) +  
  <GEOM_FUNCTION>(mapping = aes(<MAPPINGS>))
```



Aesthetics



Aesthetics



Visual Space

Data Space

color ↔ clean_test

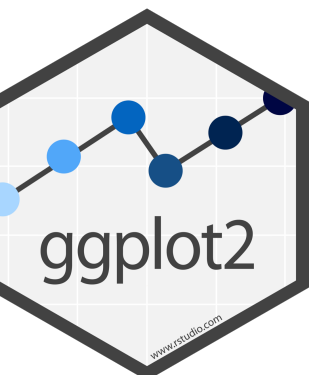
Purple ↔ nowomen

Blue ↔ notalk

Teal ↔ men

Lime ↔ dubious

Yellow ↔ ok



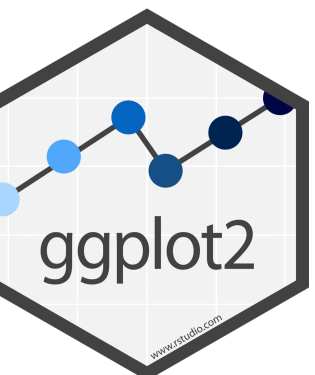
Aesthetics

aesthetic
property

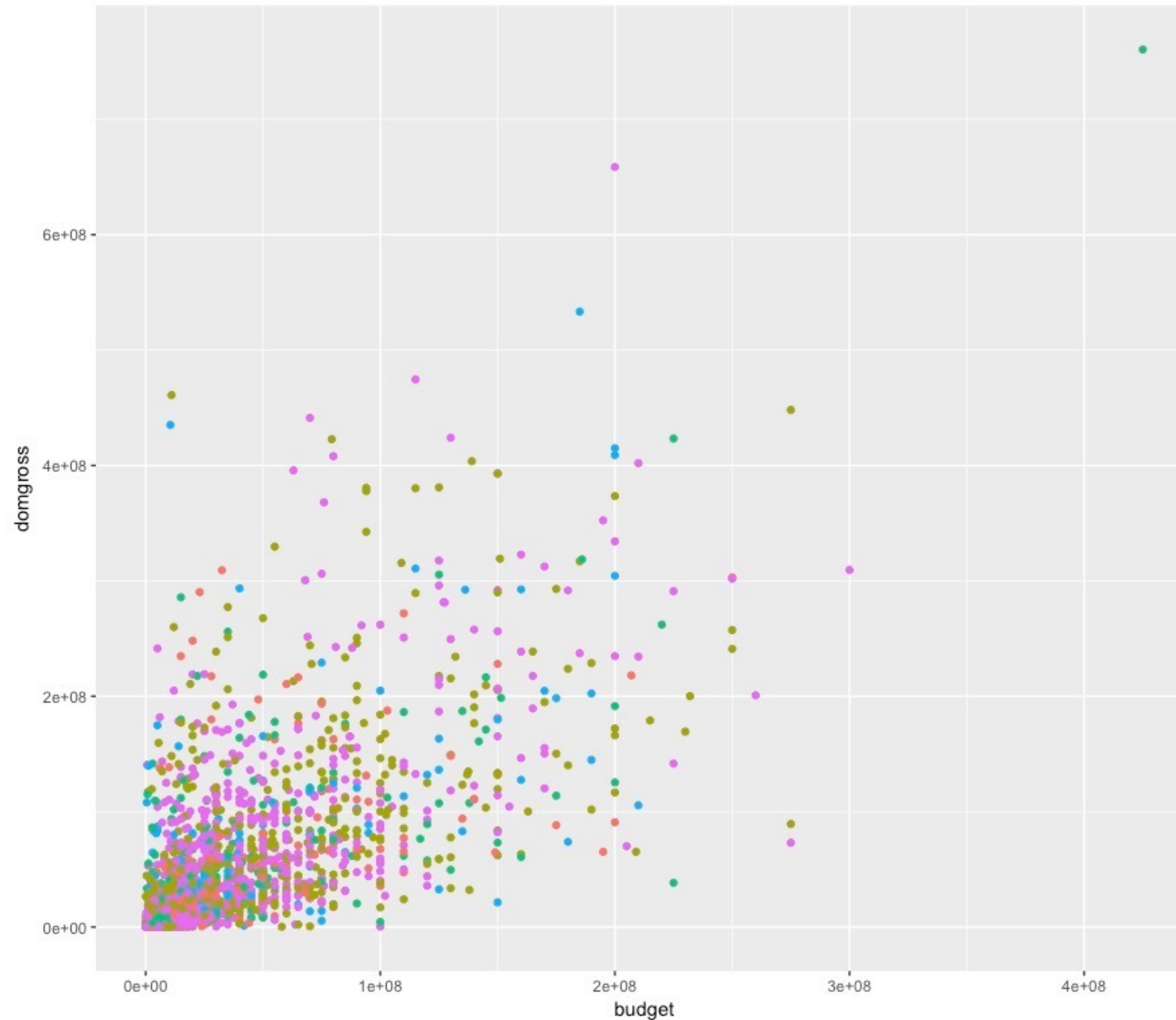
Variable to
map it to

```
ggplot(bechdel) +  
  geom_point(mapping = aes(x = budget, y = domgross, color = clean_test))
```

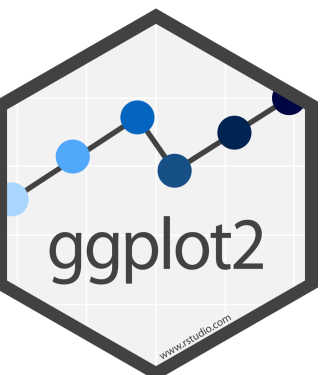
```
ggplot(bechdel) +  
  geom_point(mapping = aes(x = budget, y = domgross, size = clean_test))
```



```
ggplot(beachdel) +  
  geom_point(mapping = aes(x = budget, y = domgross, color=clean_test))
```



Legend added
automatically



Your Turn 2

In the next chunk, add color, size, alpha, and shape aesthetics to your graph. Experiment.

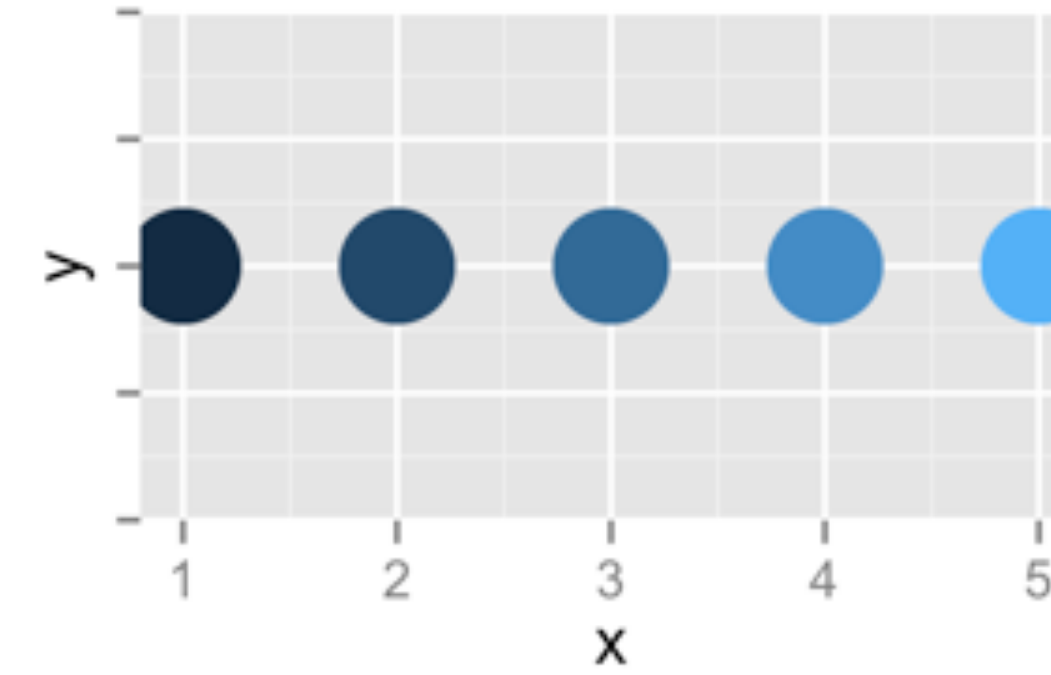
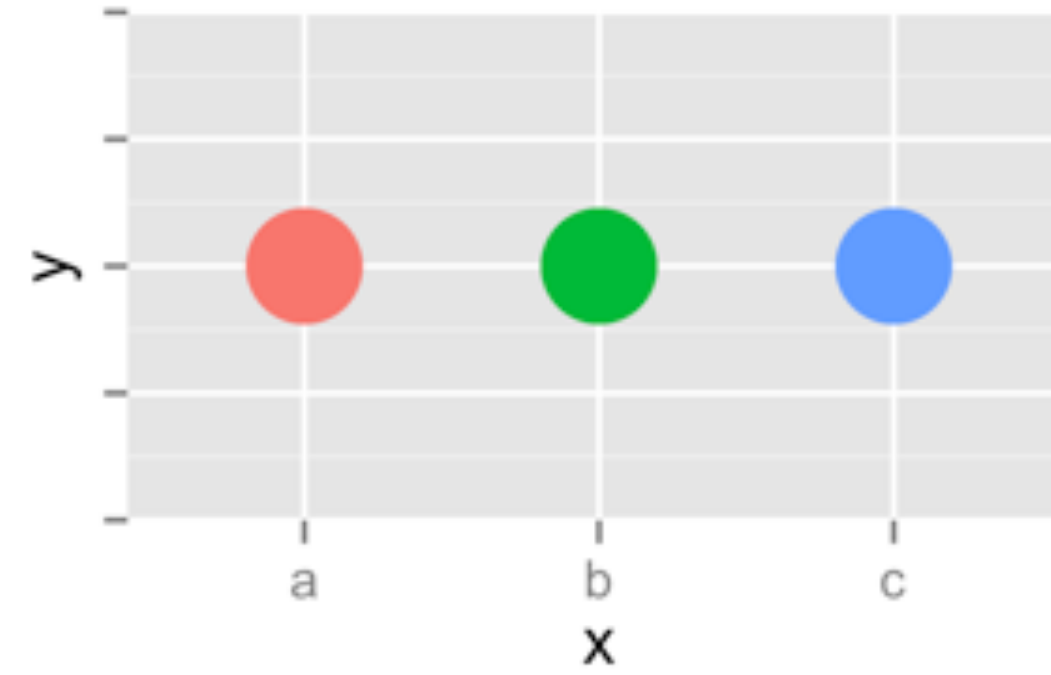
- Do different things happen when you map aesthetics to discrete and continuous variables?
- What happens when you use more than one aesthetic?

05:00

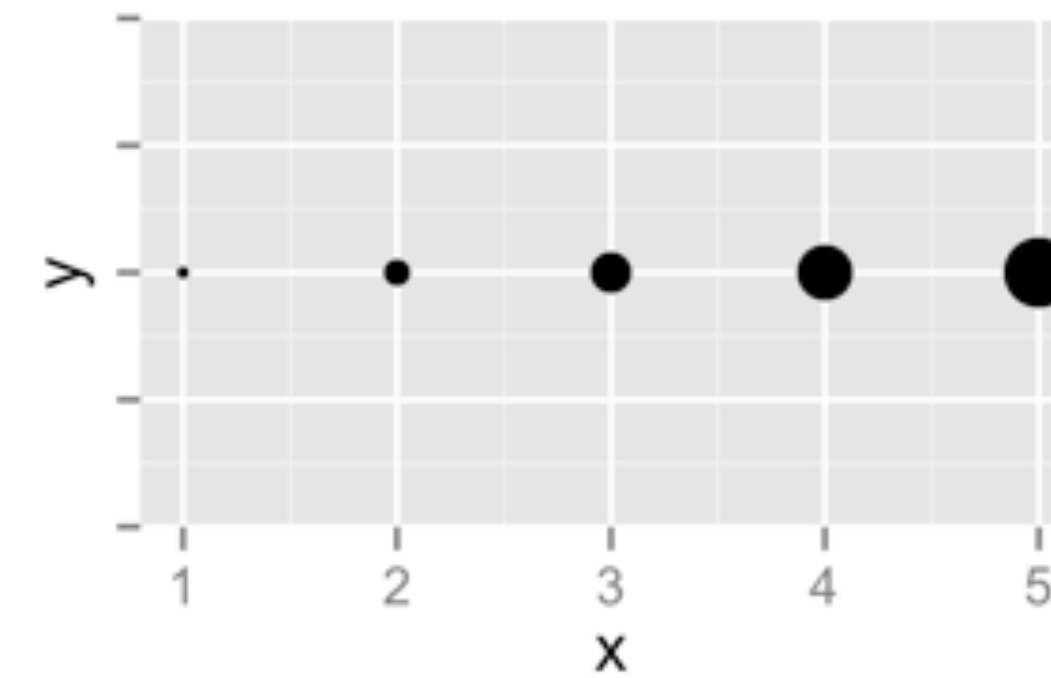
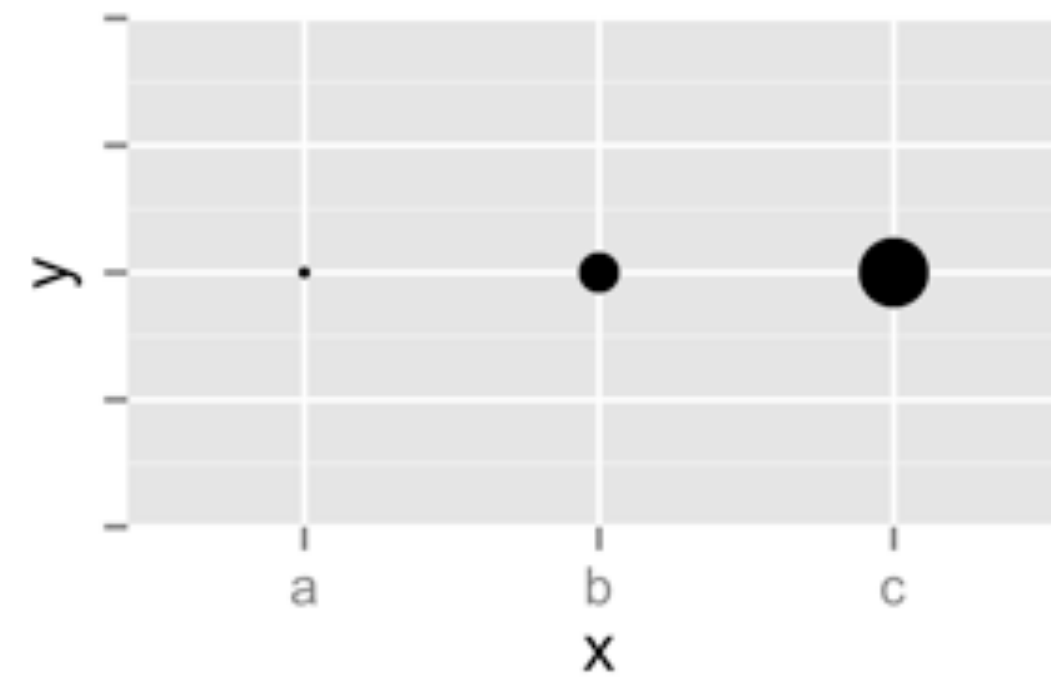
Discrete

Continuous

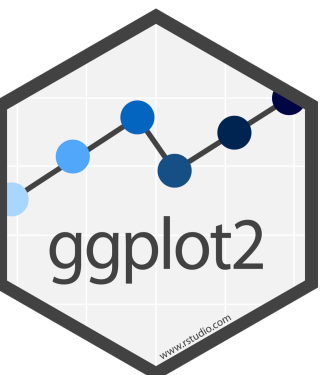
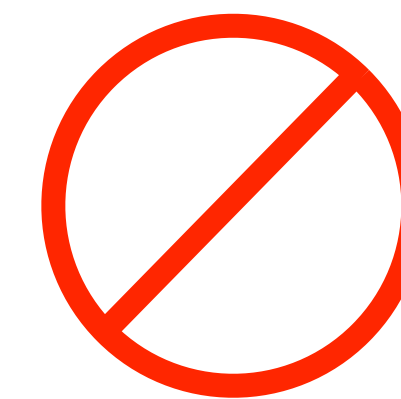
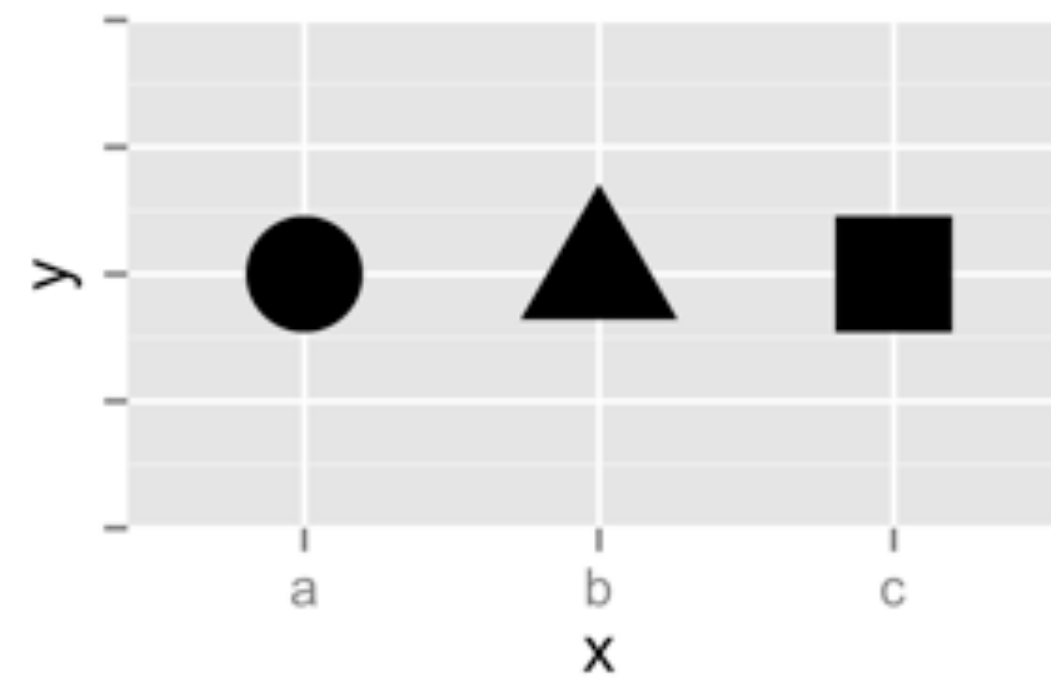
Color



Size



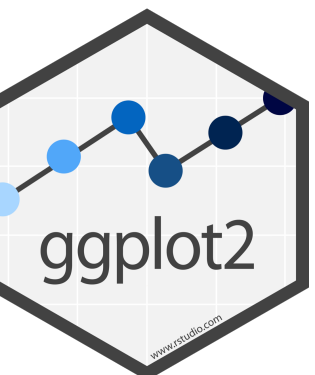
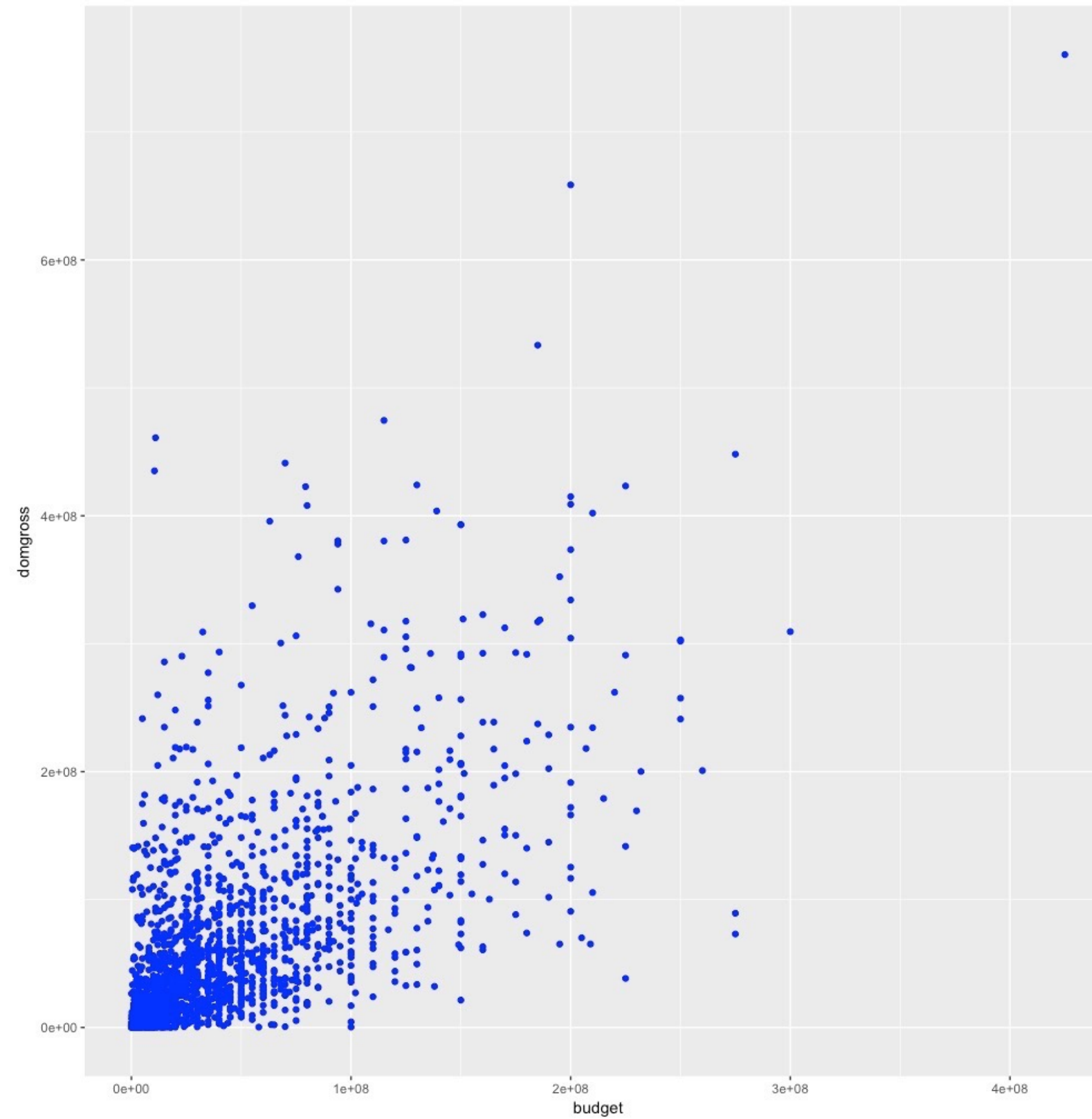
Shape

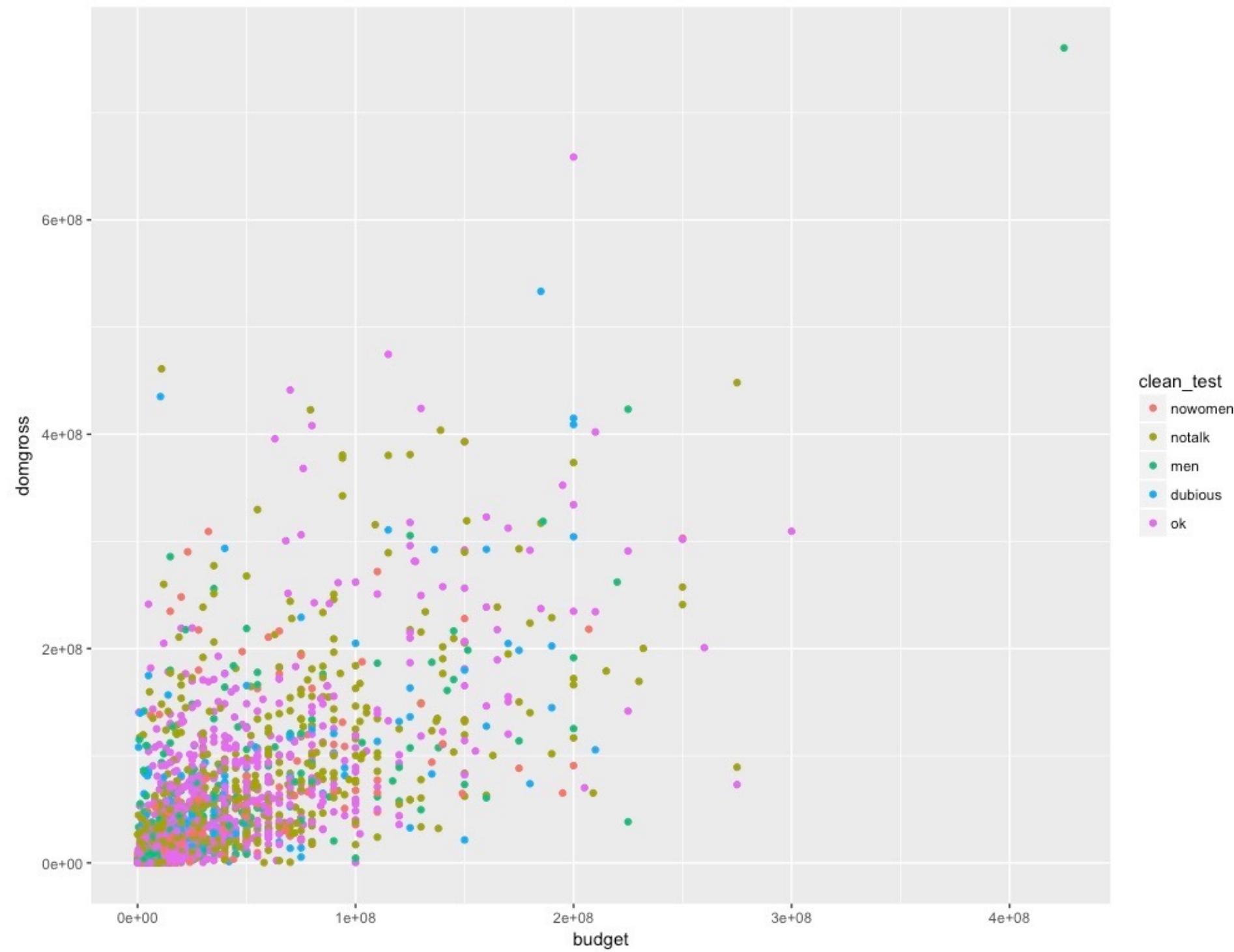


set vs. map



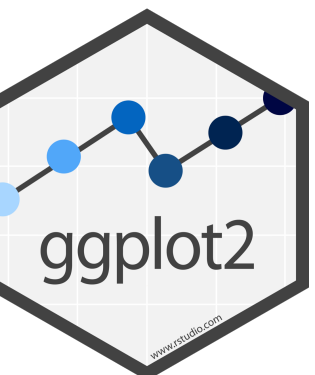
How would you make this plot?



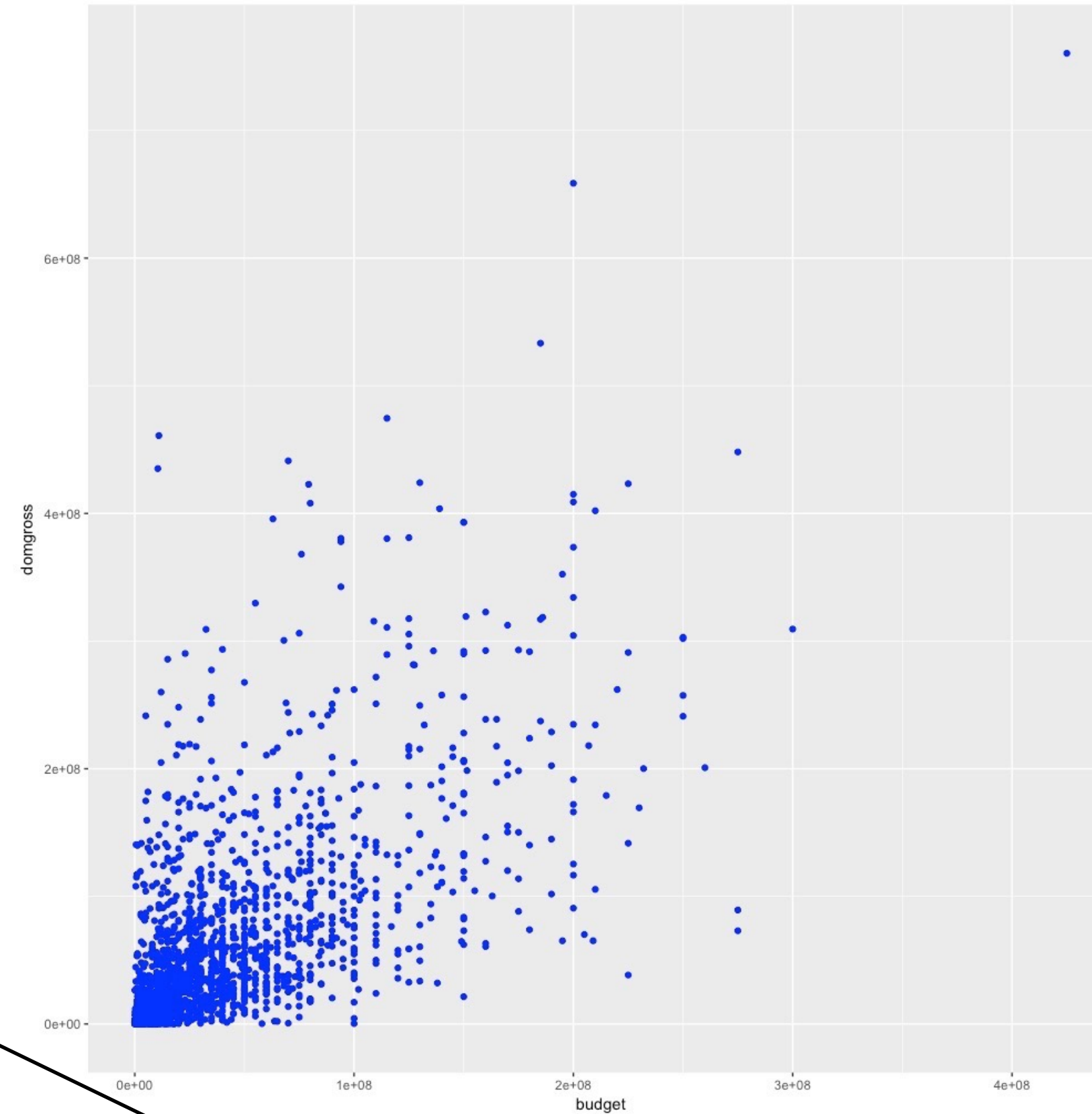


Inside of aes(): maps an aesthetic to a variable

```
ggplot(bechdel) +  
  geom_point(mapping = aes(x = budget, y = domgross, color=clean_test))
```

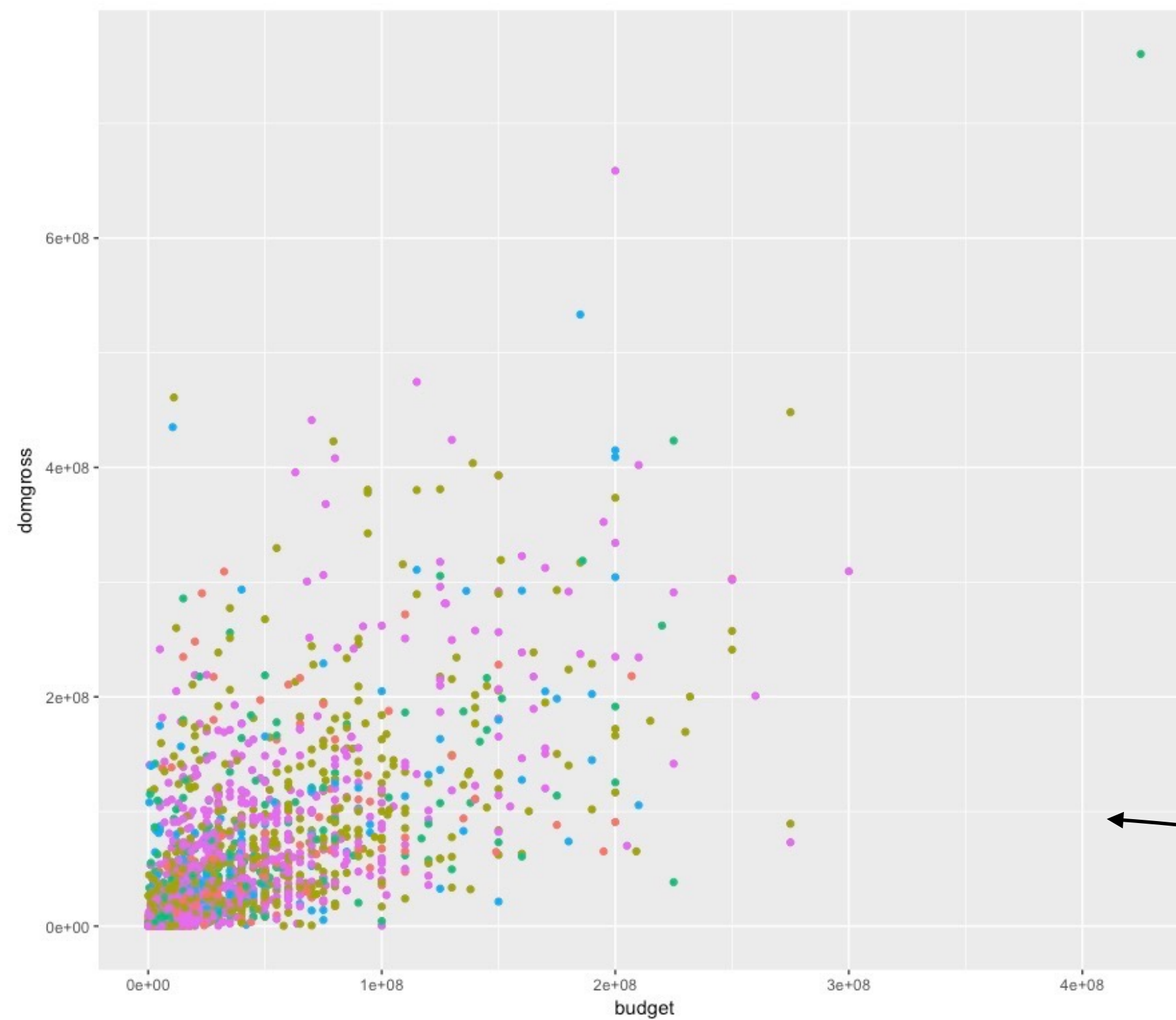


Outside of aes(): sets an aesthetic to a value



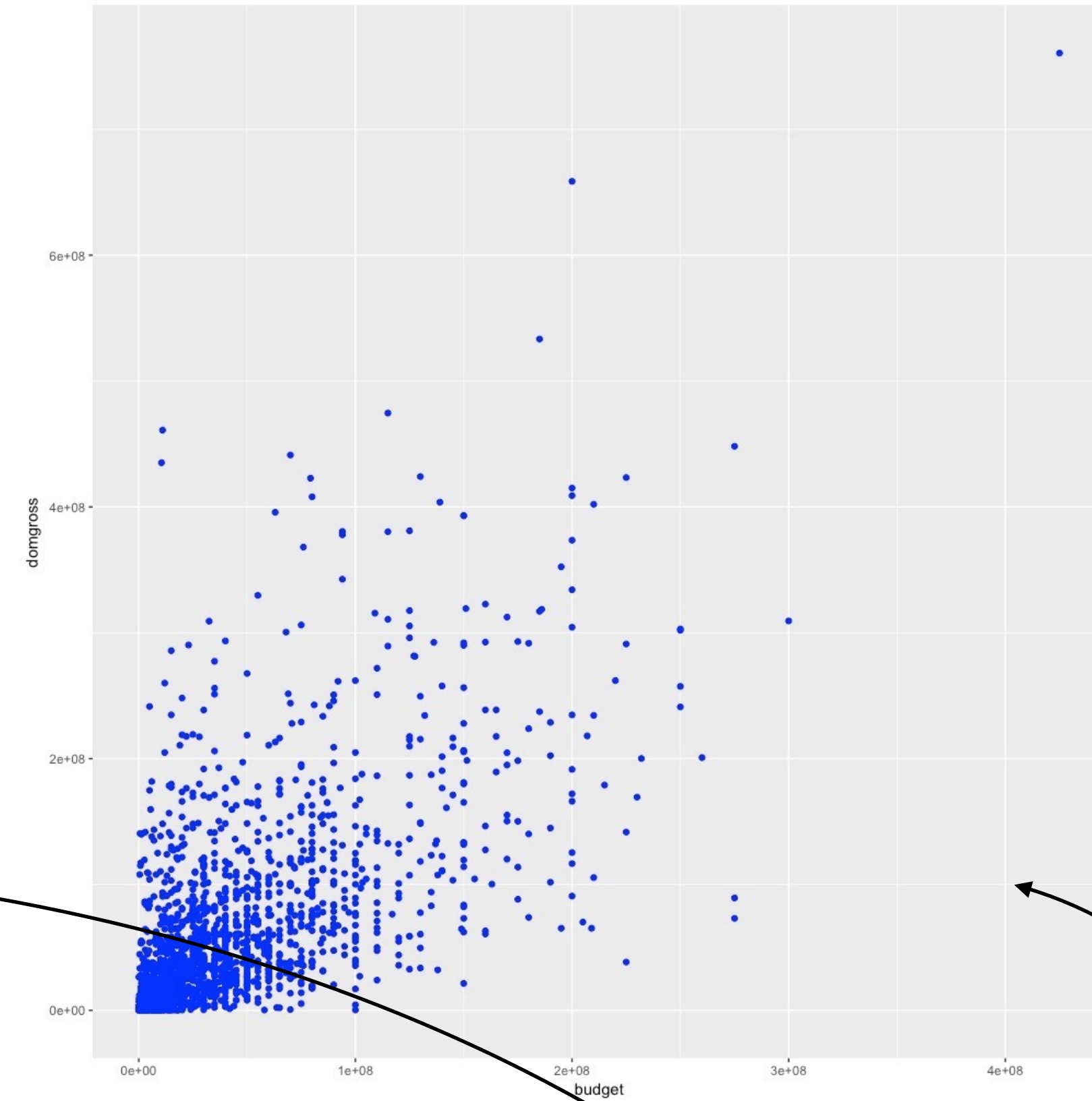
```
ggplot(bechdel) +  
  geom_point(mapping = aes(x = budget, y = domgross, color=clean_test))
```

```
ggplot(bechdel) +  
  geom_point(mapping = aes(x = budget, y = domgross), color="blue")
```



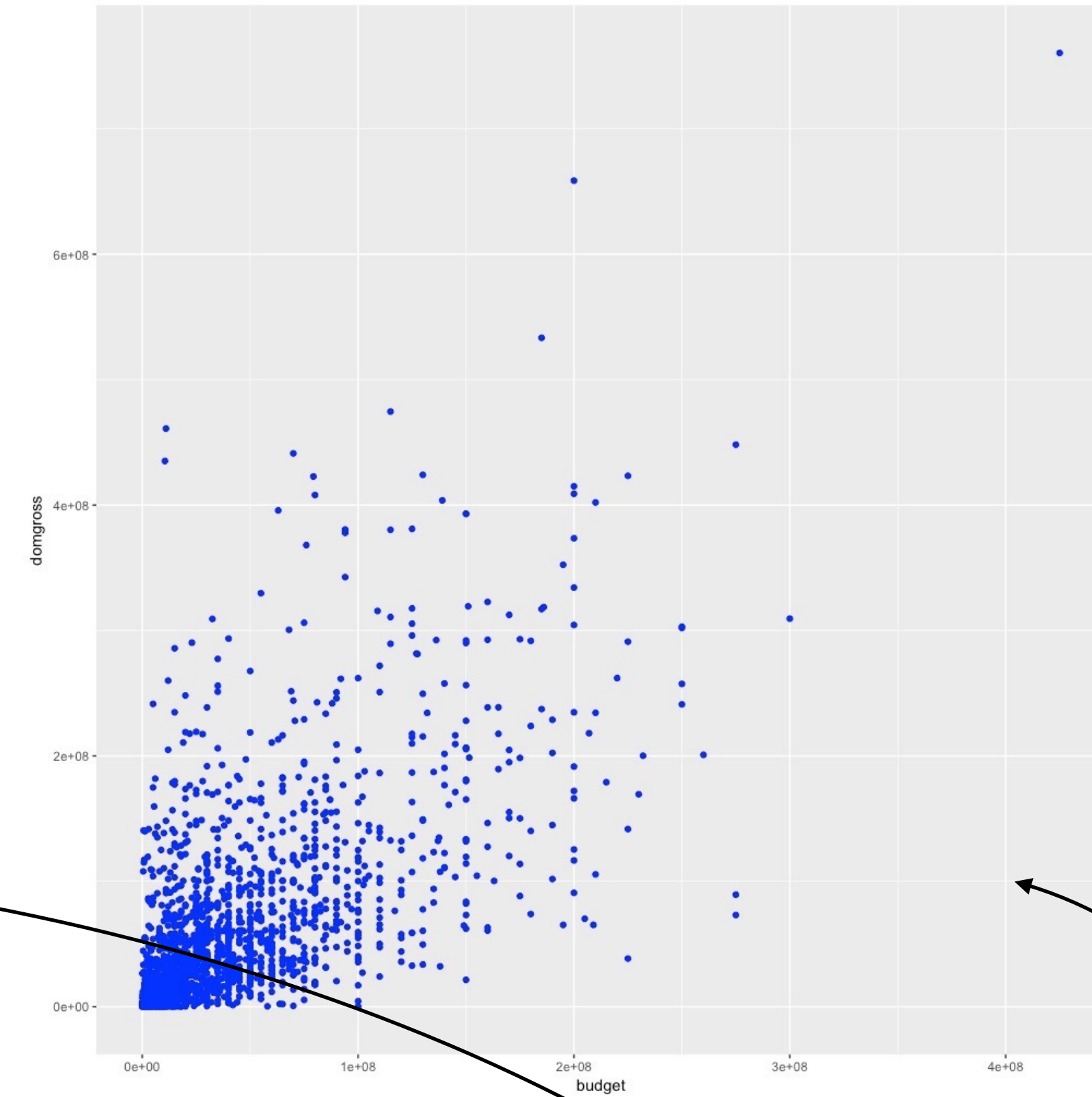
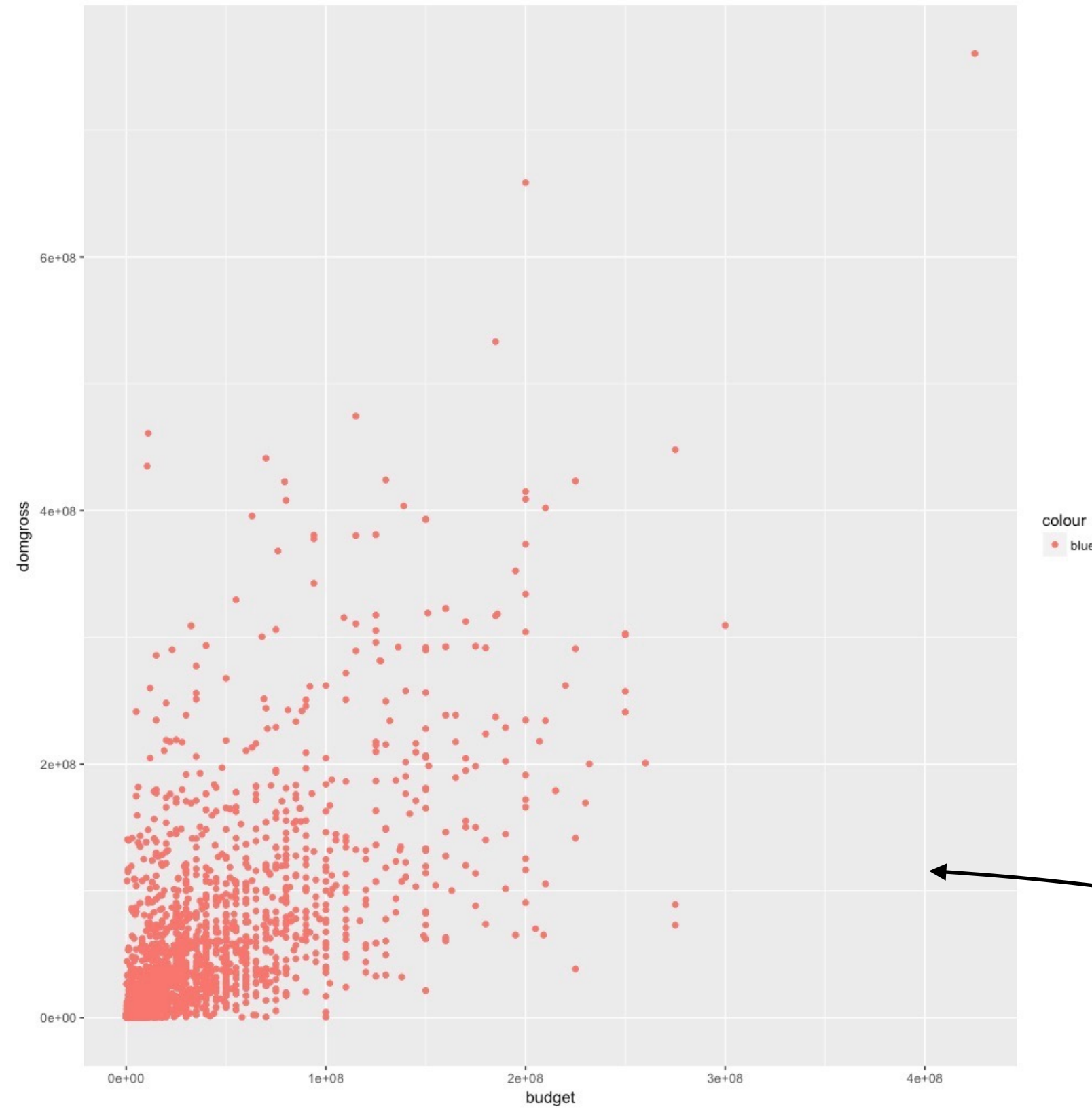
clean_test

- nowomen
- notalk
- men
- dubious
- ok



```
ggplot(bechdel) +  
  geom_point(mapping = aes(x = budget, y = domgross, color=clean_test))
```

```
ggplot(bechdel) +  
  geom_point(mapping = aes(x = budget, y = domgross), color="blue")
```



```
ggplot(bechdel) +  
  geom_point(mapping = aes(x = budget, y = domgross, color="blue"))
```

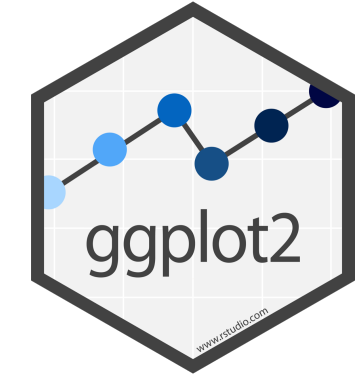
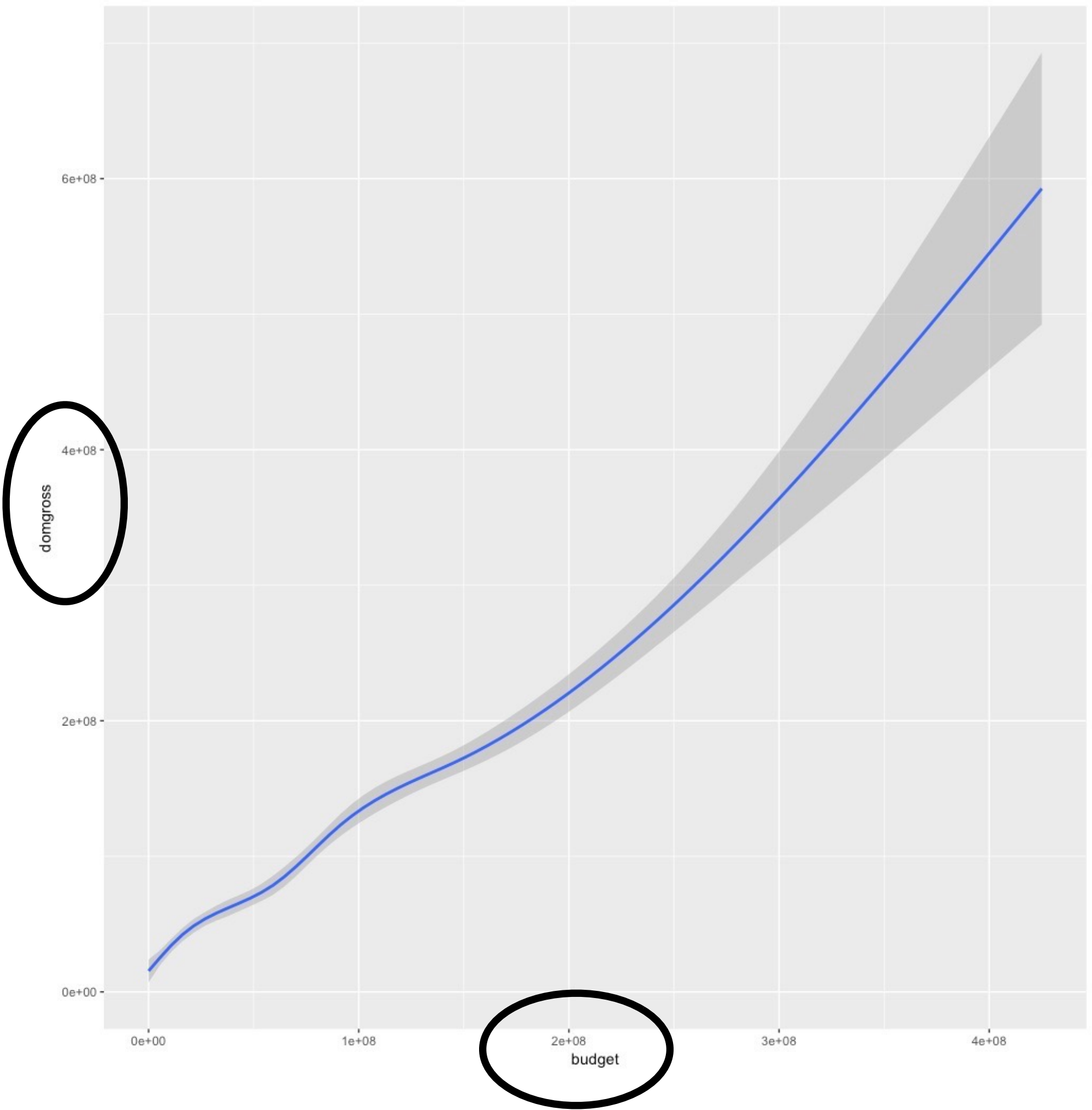
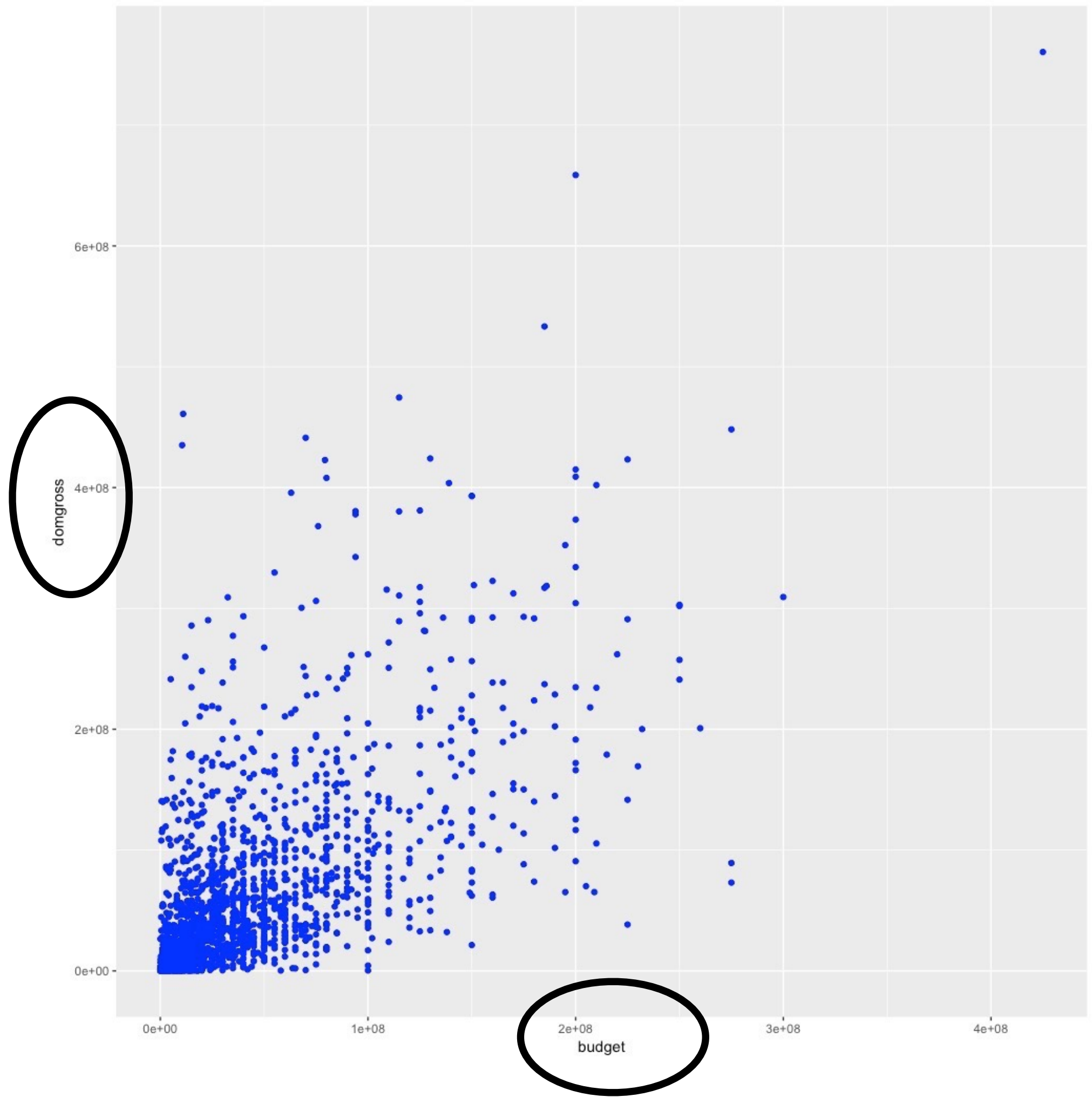
```
ggplot(bechdel) +  
  geom_point(mapping = aes(x = budget, y = domgross), color="blue")
```

Geoms



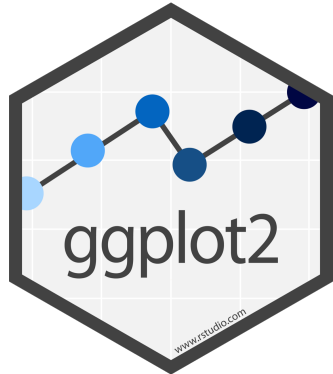
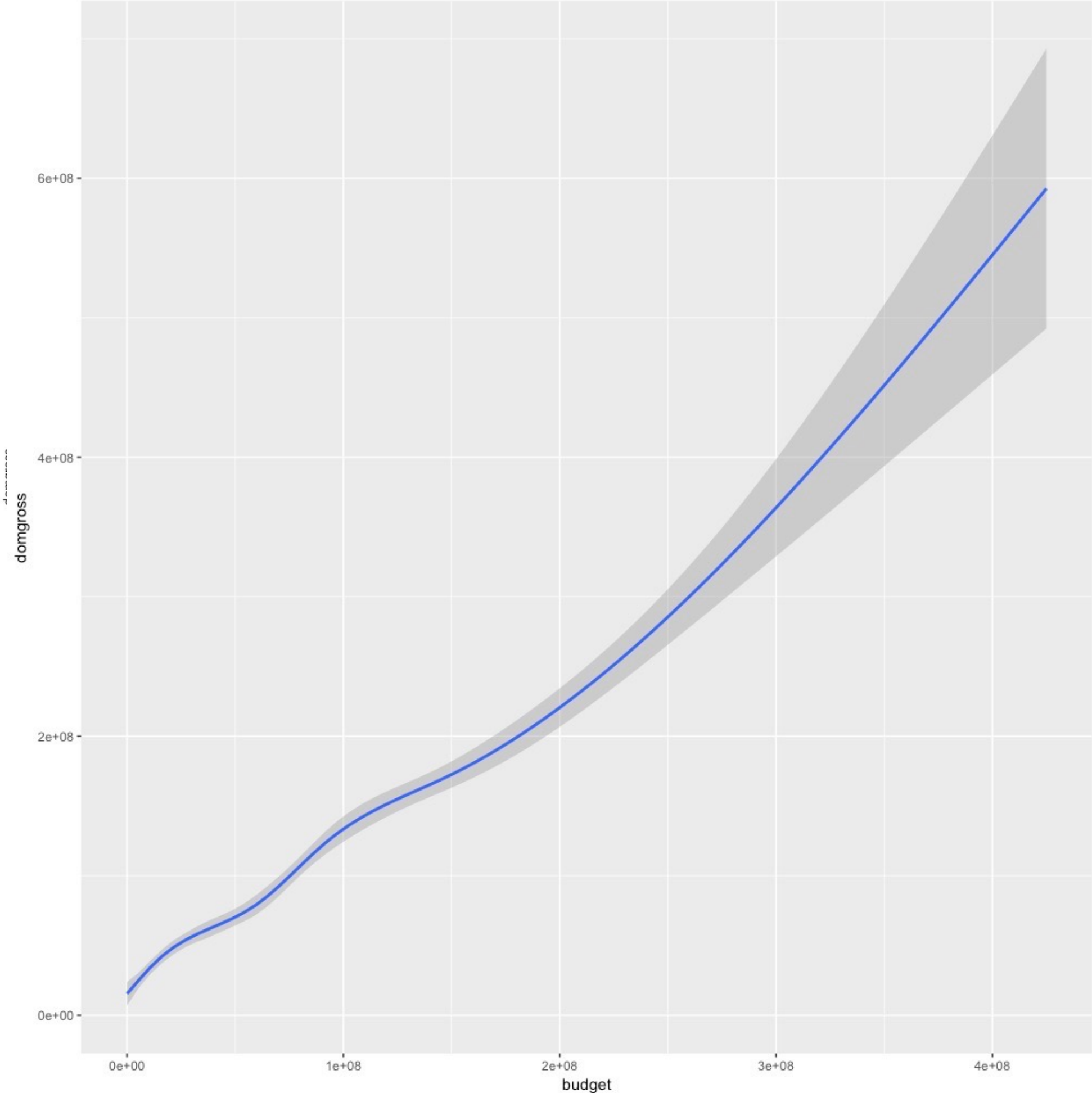
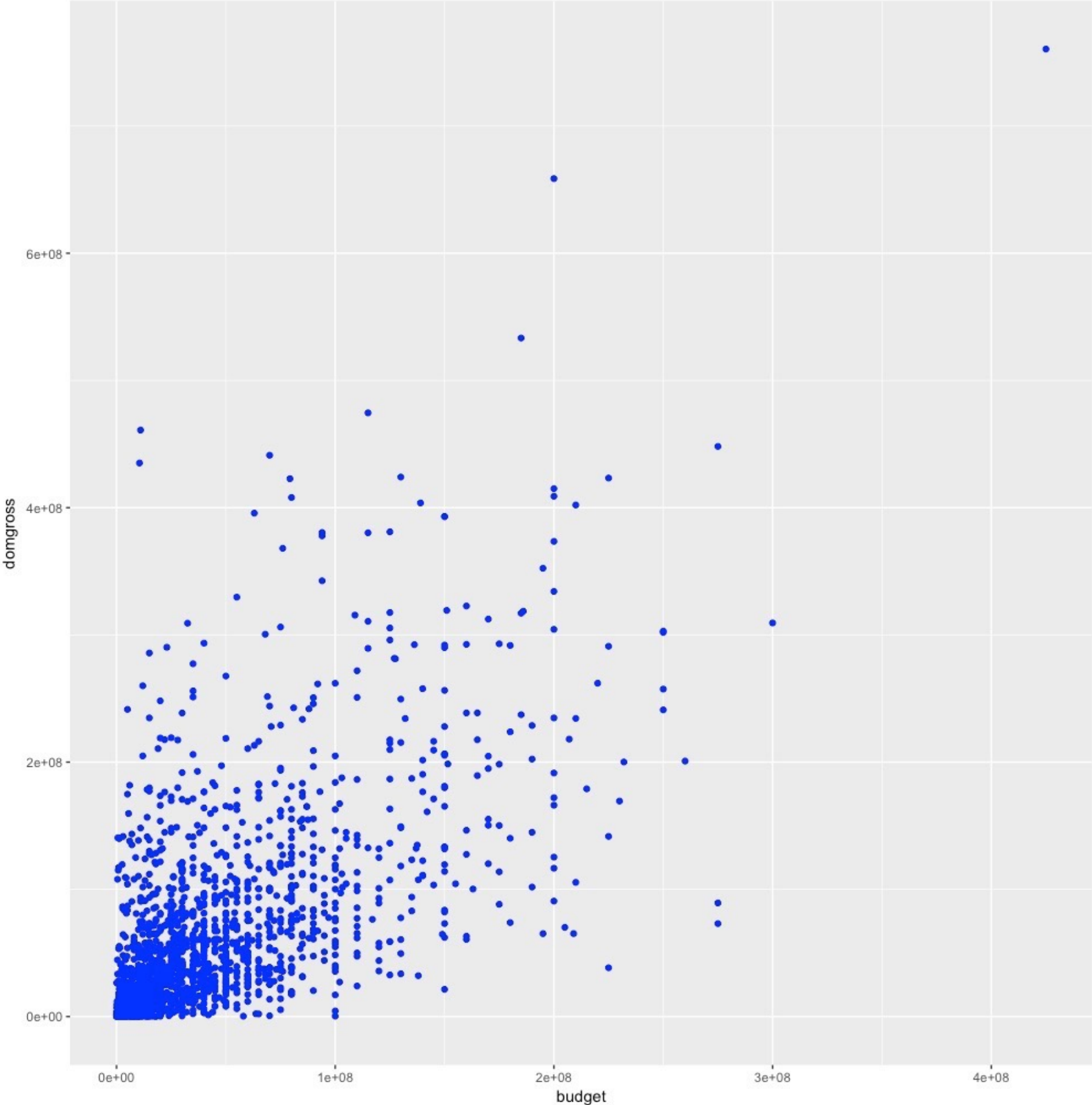
How are these plots similar?

Same: x var , y var , data



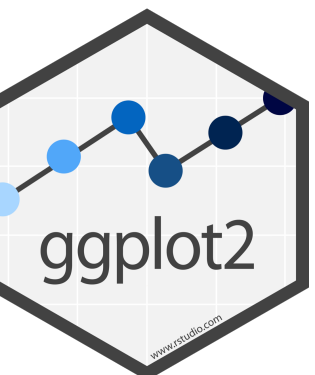
How are these plots different?

Different: geometric object (geom), e.g. the visual object used to represent the data



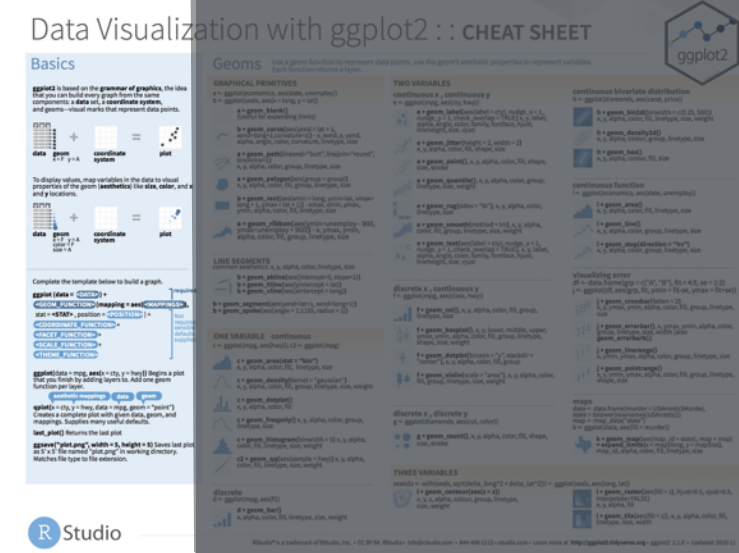
geoms

```
ggplot(data = <DATA>) +  
  <GEOM_FUNCTION>(mapping = aes(<MAPPINGS>))
```



geom_ functions

Each requires a mapping argument.



Geoms Use a geom function to represent data points, use the geom's aesthetic properties to represent variables. Each function returns a layer. ggplot2

GRAPHICAL PRIMITIVES
a <- ggplot(economics, aes(date, unemploy))
b <- ggplot(seals, aes(x = long, y = lat))

- a + geom_blank()** (Useful for expanding limits)
- b + geom_curve**(aes(yend = lat + 1, xend = long + 1, curvature = z)) - x, xend, y, yend, alpha, angle, color, curvature, linetype, size
- a + geom_path**(lineend = "butt", linejoin = "round", linewidth = 1) x, y, alpha, color, group, linetype, size
- a + geom_polygon**(aes(group = group)) x, y, alpha, color, fill, group, linetype, size
- b + geom_rect**(aes(xmin = long, ymin = lat, xmax = long + 1, ymax = lat + 1)) - xmax, xmin, ymax, ymin, alpha, color, fill, linetype, size
- a + geom_ribbon**(aes(ymin = unemploy - 900, ymax = unemploy + 900)) - x, ymax, ymih, alpha, color, fill, group, linetype, size

LINE SEGMENTS
common aesthetics: x, y, alpha, color, linetype, size

- b + geom_abline**(aes(intercept = 0, slope = 1))
- b + geom_hline**(aes(yintercept = lat))
- b + geom_vline**(aes(xintercept = long))
- b + geom_segment**(aes(yend = lat + 1, xend = long + 1))
- b + geom_spoke**(aes(angle = 1:1155, radius = 1))

ONE VARIABLE continuous
c <- ggplot(mpg, aes(hwy)); c2 <- ggplot(mpg)

- c + geom_area**(stat = "bin") x, y, alpha, color, fill, linetype, size
- c + geom_density**(kernel = "gaussian") x, y, alpha, color, fill, group, linetype, size, weight
- c + geom_dotplot**() x, y, alpha, color, fill
- c + geom_freqpoly**() x, y, alpha, color, group, linetype, size
- c + geom_histogram**(binwidth = 5) x, y, alpha, color, fill, linetype, size, weight
- c2 + geom_qq**(aes(sample = hwy)) x, y, alpha, color, fill, linetype, size, weight

discrete
d <- ggplot(mpg, aes(fl))

- d + geom_bar**() x, alpha, color, fill, linetype, size, weight

TWO VARIABLES
continuous x, continuous y
e <- ggplot(mpg, aes(cty, hwy))

- e + geom_label**(aes(label = cty), nudge_x = 1, nudge_y = 1, check_overlap = TRUE) x, y, label, alpha, angle, color, family, fontface, hjust, lineheight, size, vjust
- e + geom_jitter**(height = 2, width = 2) x, y, alpha, color, fill, shape, size
- e + geom_point**() x, y, alpha, color, fill, shape, size, stroke
- e + geom_quantile**() x, y, alpha, color, group, linetype, size, weight
- e + geom_rug**(sides = "bl") x, y, alpha, color, linetype, size
- e + geom_smooth**(method = lm) x, y, alpha, color, fill, group, linetype, size, weight
- e + geom_text**(aes(label = cty), nudge_x = 1, nudge_y = 1, check_overlap = TRUE) x, y, label, alpha, angle, color, family, fontface, hjust, lineheight, size, vjust

discrete x, continuous y
f <- ggplot(mpg, aes(class, hwy))

- f + geom_col**() x, y, alpha, color, fill, group, linetype, size
- f + geom_boxplot**() x, y, lower, middle, upper, ymax, ymin, alpha, color, fill, group, linetype, shape, size, weight
- f + geom_dotplot**(binaxis = "y", stackdir = "center") x, y, alpha, color, fill, group
- f + geom_violin**(scale = "area") x, y, alpha, color, fill, group, linetype, size, weight

discrete x, discrete y
g <- ggplot(diamonds, aes(cut, color))

- g + geom_count**() x, y, alpha, color, fill, shape, size, stroke

THREE VARIABLES
sealsSz <- with(seals, sqrt(delta_long^2 + delta_lat^2)) | <- ggplot(seals, aes(long, lat))

- l + geom_contour**(aes(z = z)) x, y, z, alpha, colour, group, linetype, size, weight
- l + geom_raster**(aes(fill = z), hjust = 0.5, vjust = 0.5, interpolate = FALSE) x, y, alpha, fill
- l + geom_tile**(aes(fill = z)) x, y, alpha, color, fill, linetype, size, width

continuous bivariate distribution
h <- ggplot(diamonds, aes(carat, price))

- h + geom_bin2d**(binwidth = c(0.25, 500)) x, y, alpha, color, fill, linetype, size, weight
- h + geom_density2d**() x, y, alpha, colour, group, linetype, size
- h + geom_hex**() x, y, alpha, colour, fill, size

continuous function
i <- ggplot(economics, aes(date, unemploy))

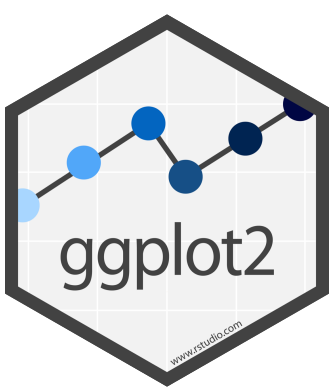
- i + geom_area**() x, y, alpha, color, fill, linetype, size
- i + geom_line**() x, y, alpha, color, group, linetype, size
- i + geom_step**(direction = "hv") x, y, alpha, color, group, linetype, size

visualizing error
df <- data.frame(grp = c("A", "B"), fit = 4.5, se = 1.2)
j <- ggplot(df, aes(grp, fit, ymin = fit - se, ymax = fit + se))

- j + geom_crossbar**(fatten = 2) x, y, ymax, ymin, alpha, color, fill, group, linetype, size
- j + geom_errorbar**() x, ymax, ymin, alpha, color, group, linetype, size, width (also geom_errorbarh())
- j + geom_linerange**() x, ymin, ymax, alpha, color, group, linetype, size
- j + geom_pointrange**() x, y, ymin, ymax, alpha, color, fill, group, linetype, shape, size

maps
data <- data.frame(murder = USArrests\$Murder, state = tolower(row.names(USArrests)))
map <- map_data("state")
k <- ggplot(data, aes(fill = murder))

- k + geom_map**(aes(map_id = state), map = map) + expand_limits(x = map\$long, y = map\$lat, map_id, alpha, color, fill, linetype, size)



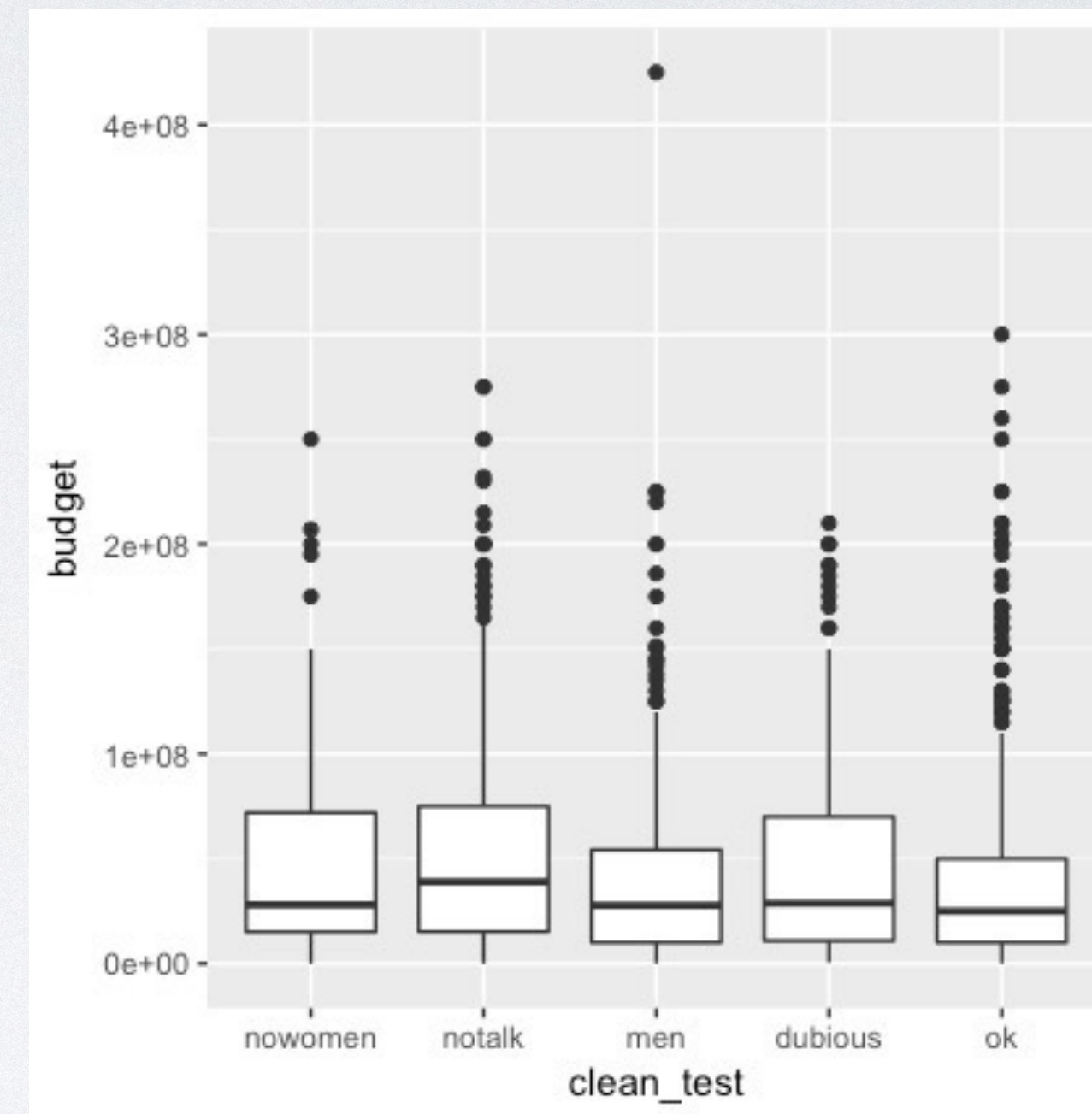
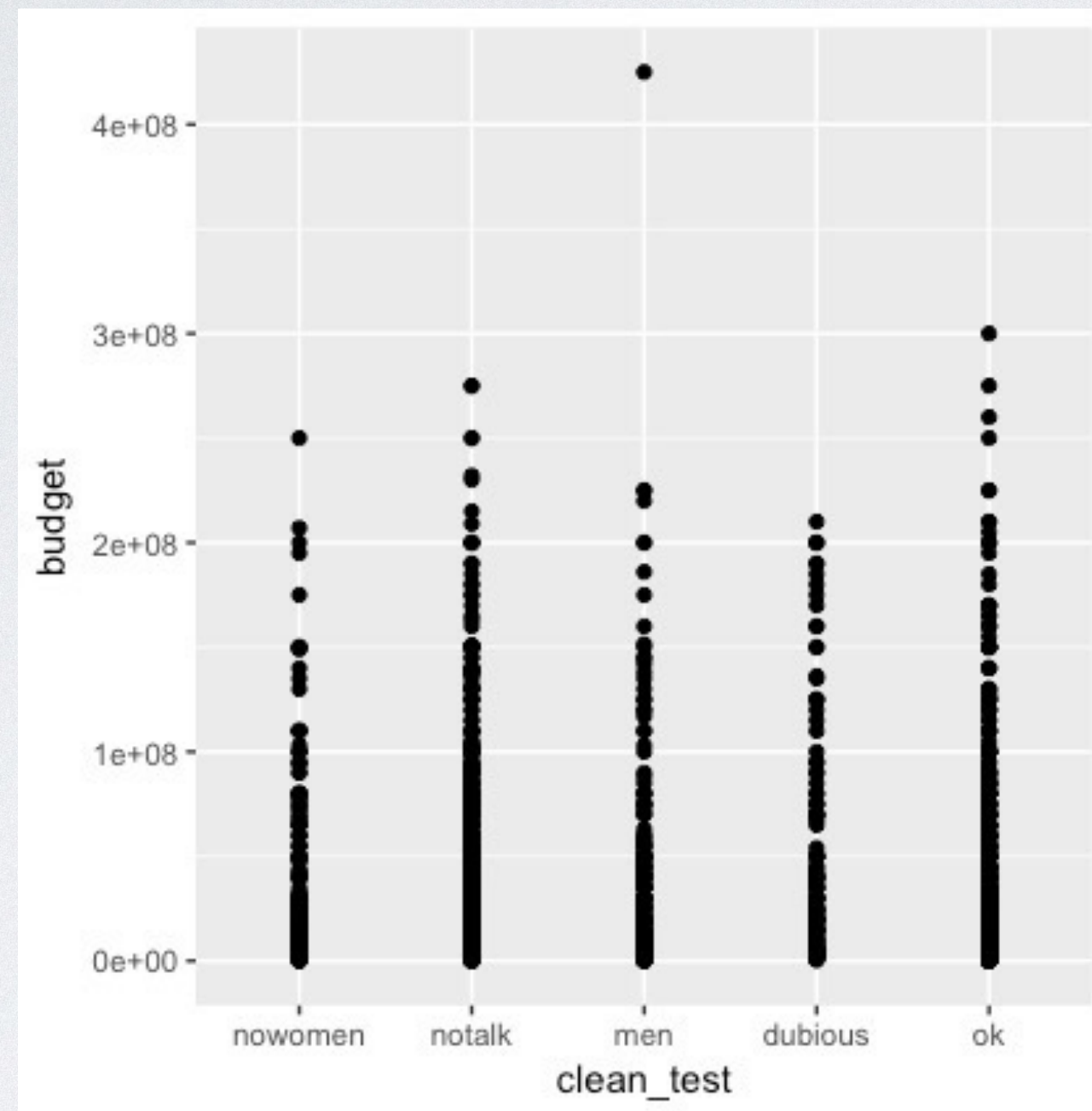
Your Turn

Pair up.

00:30

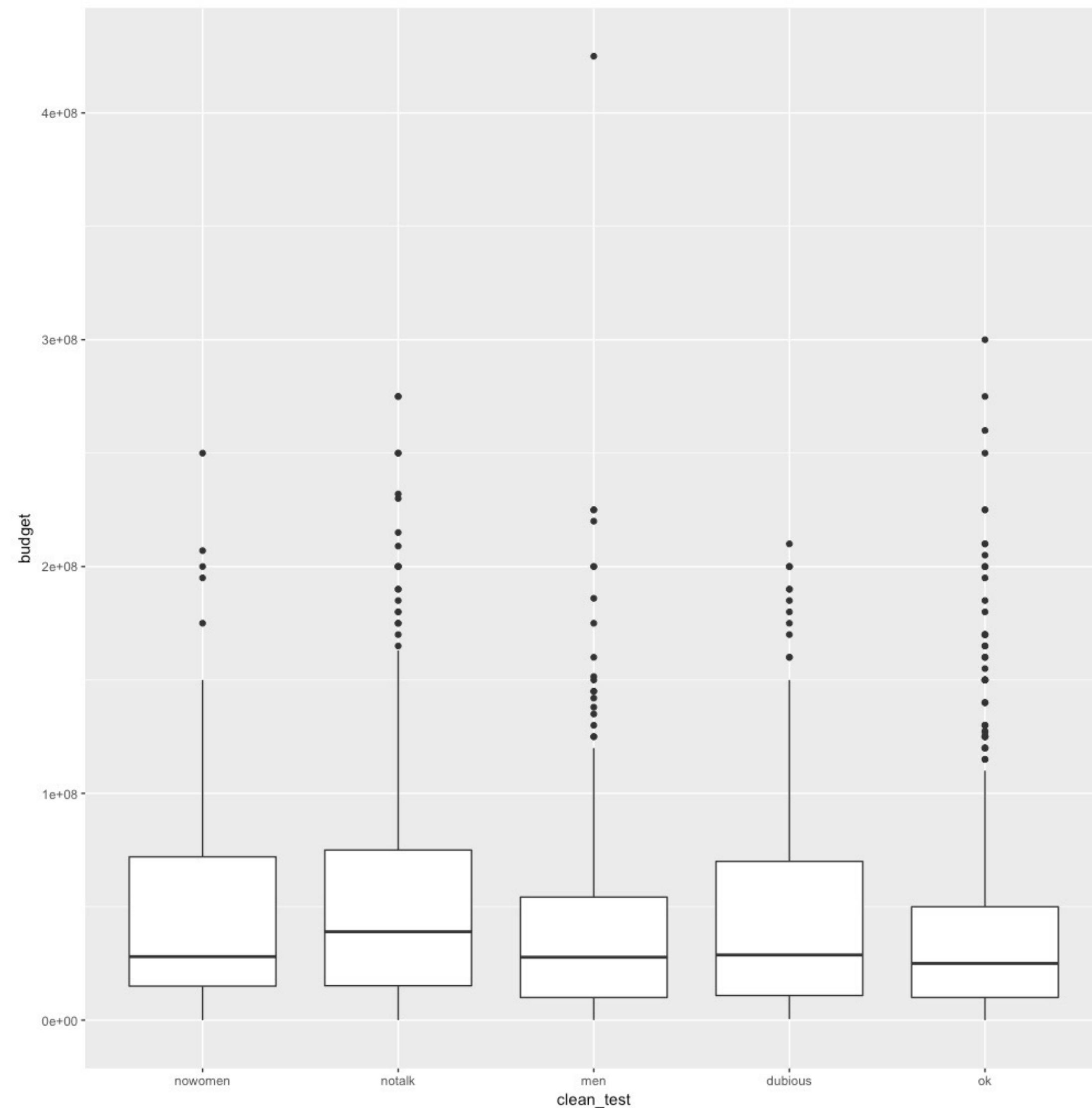
Your Turn 3

With your partner, decide how to replace this scatterplot with one that draws boxplots? Use the cheatsheet. Try your best guess.



```
ggplot(data = bechdel) +  
  geom_point(mapping = aes(x = clean_test, y = budget))
```

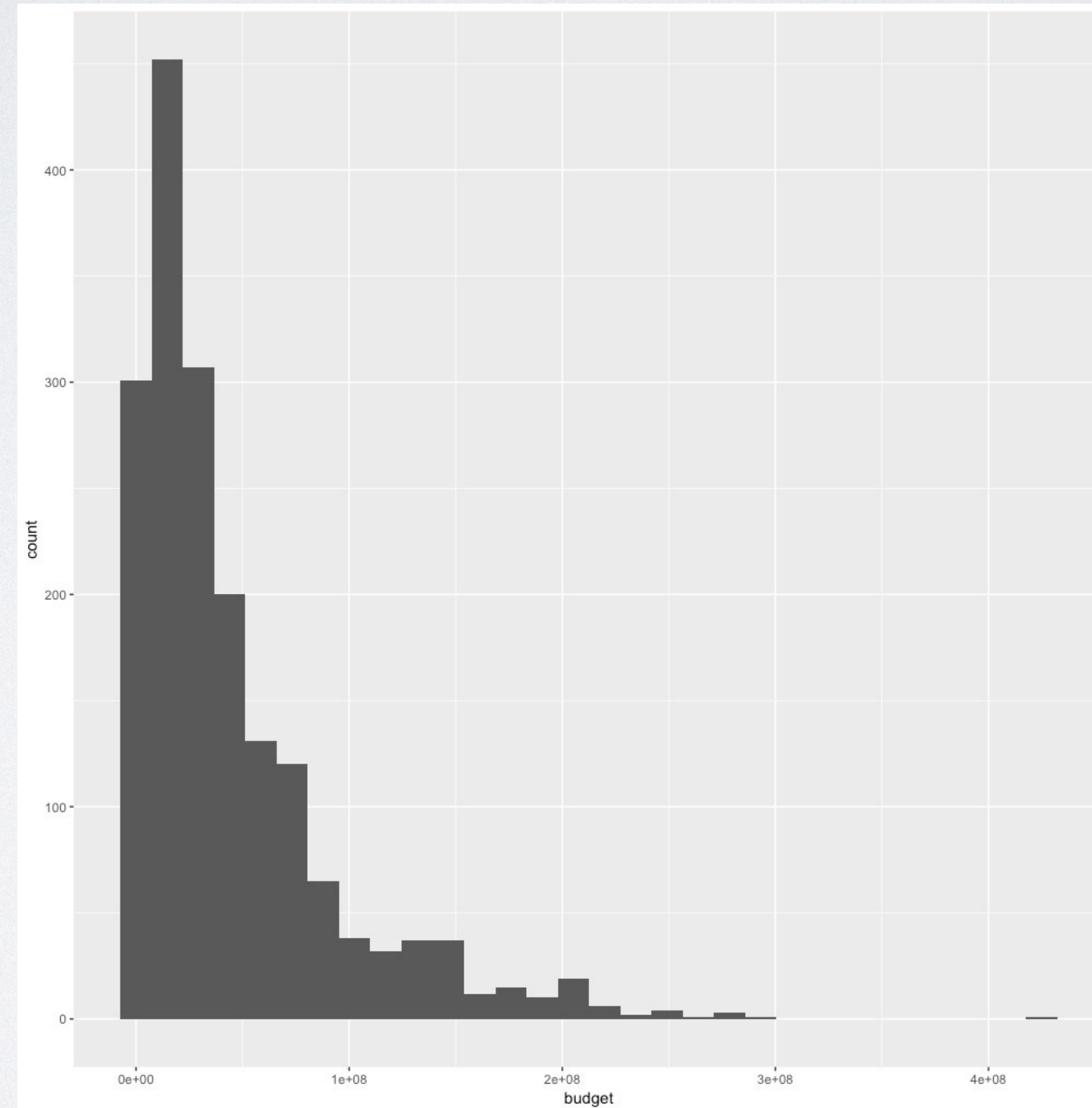
02:00



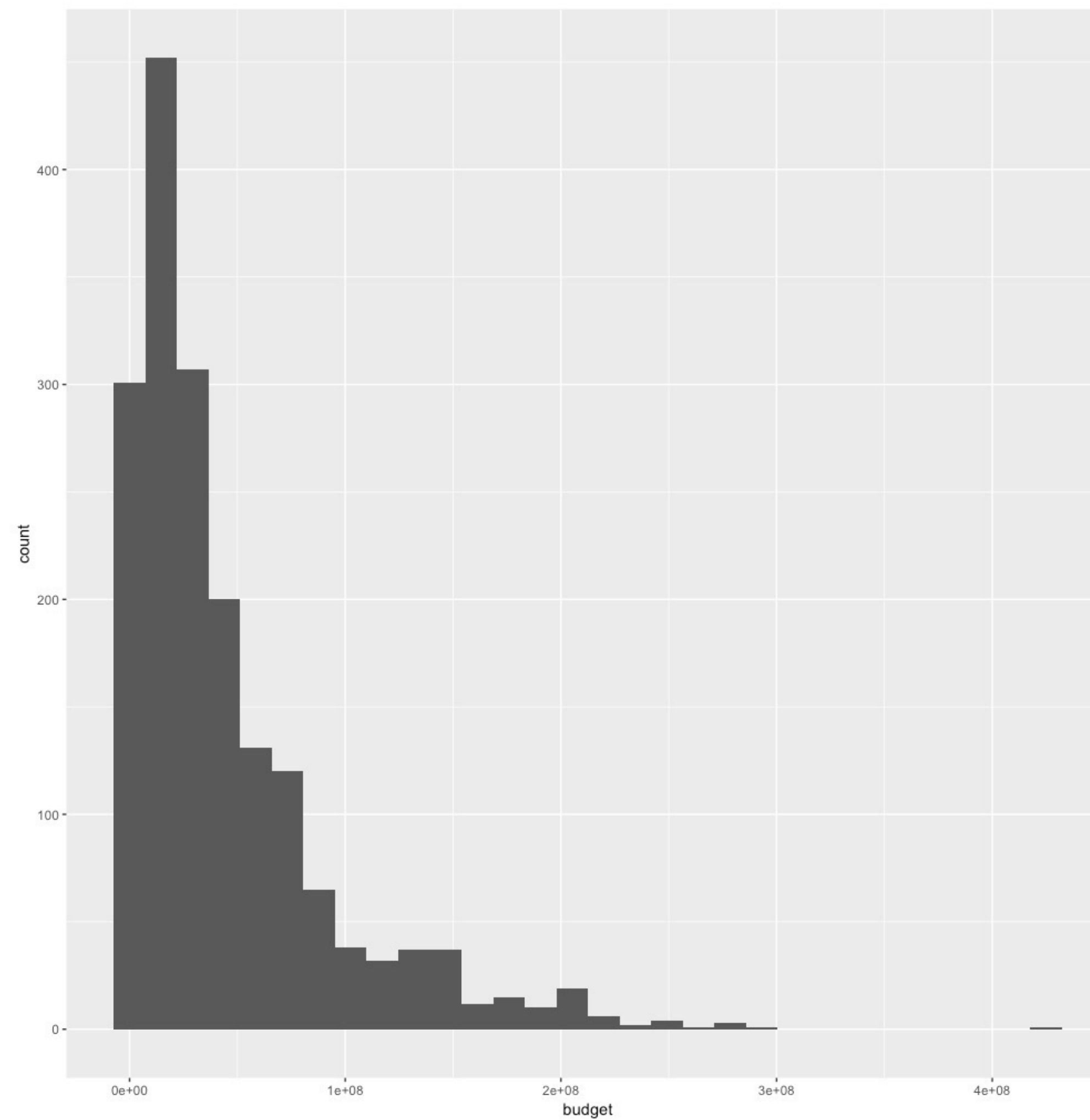
```
ggplot(data = bechdel) +  
  geom_boxplot(mapping = aes(x = clean_test, y = budget))
```

Your Turn 4

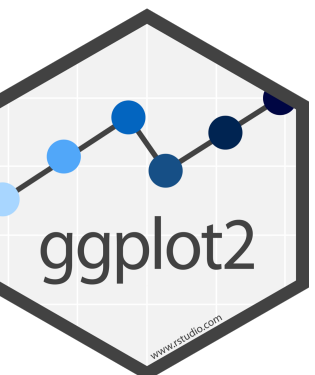
With your partner, make the histogram of **budget** below.
Use the cheatsheet. Hint: do not supply a **y** variable.



02:00



```
ggplot(data = bechdel) +  
  geom_histogram(mapping = aes(x = budget))
```

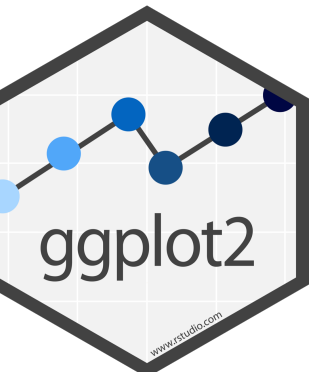


When you run this code, you will get what looks like an error, but is actually just a message from R. Because it's a bad idea to use default binwidths, the package is reminding you to pick your own



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

```
ggplot(data = bechdel) +  
  geom_histogram(mapping = aes(x = budget))
```

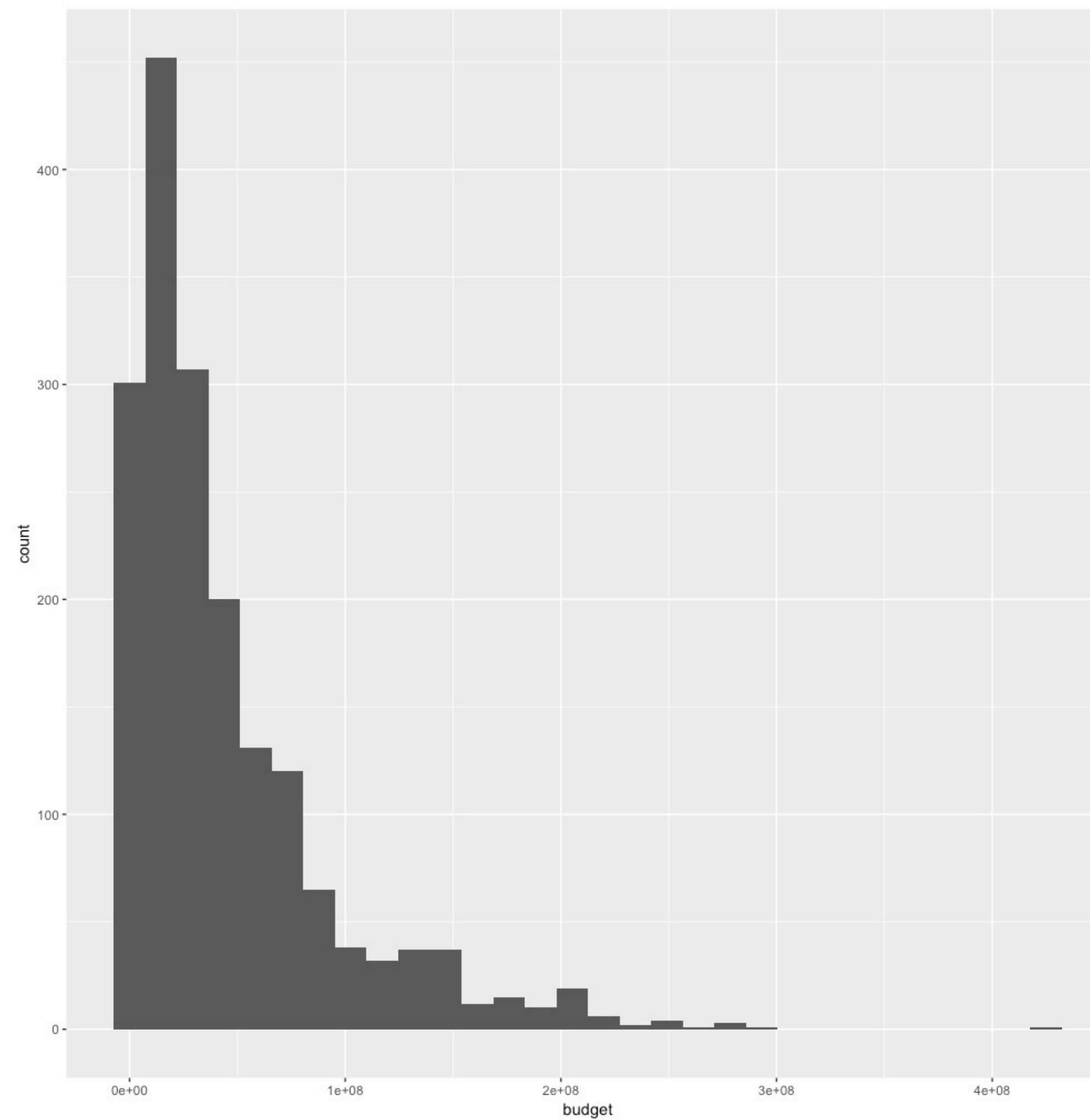


Your Turn 5

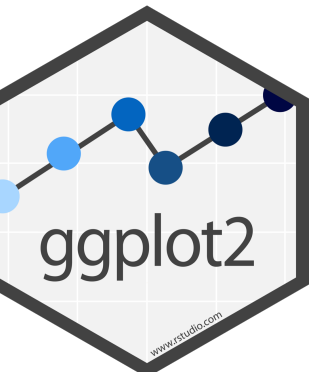
What would be a reasonable
binwidth for budget?

Try it out

00:30

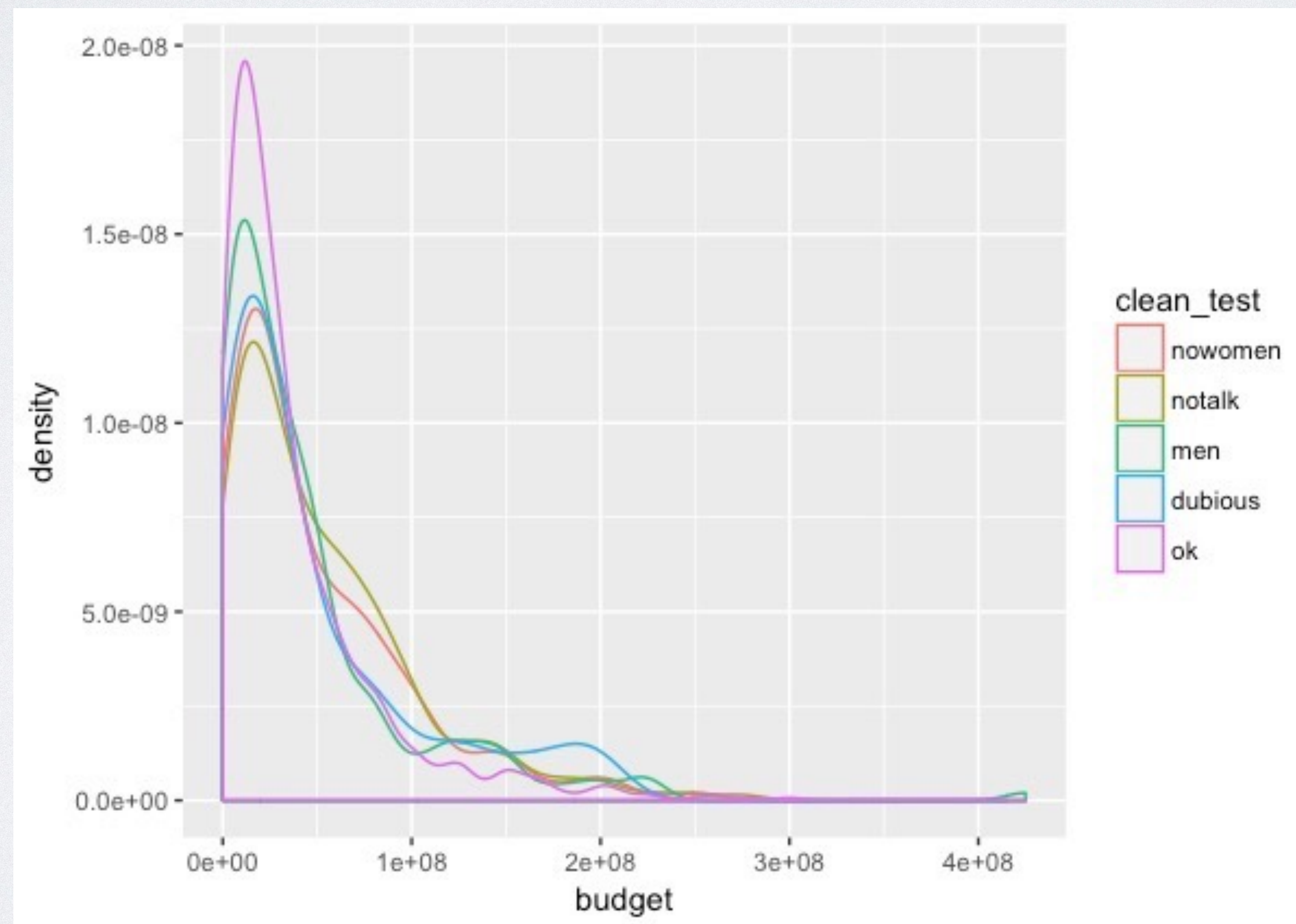


```
ggplot(data = bechdel) +  
  geom_histogram(mapping = aes(x = budget), binwidth=10000000)
```

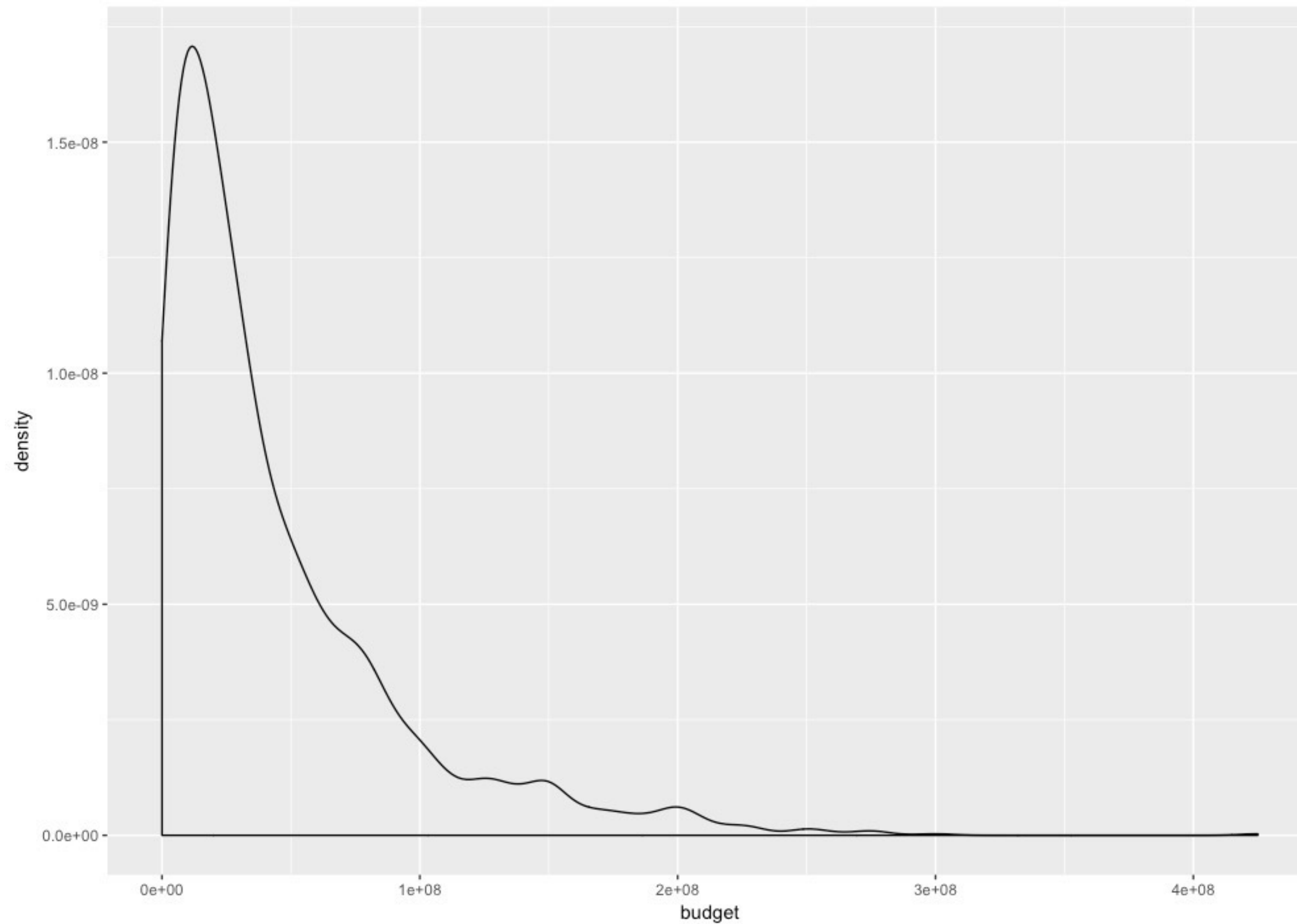


Your Turn 6

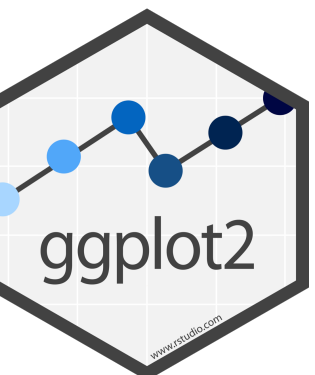
With your partner, make the density plot of **budget** colored by **clean_test** below. Use the cheatsheet. Try your best guess.

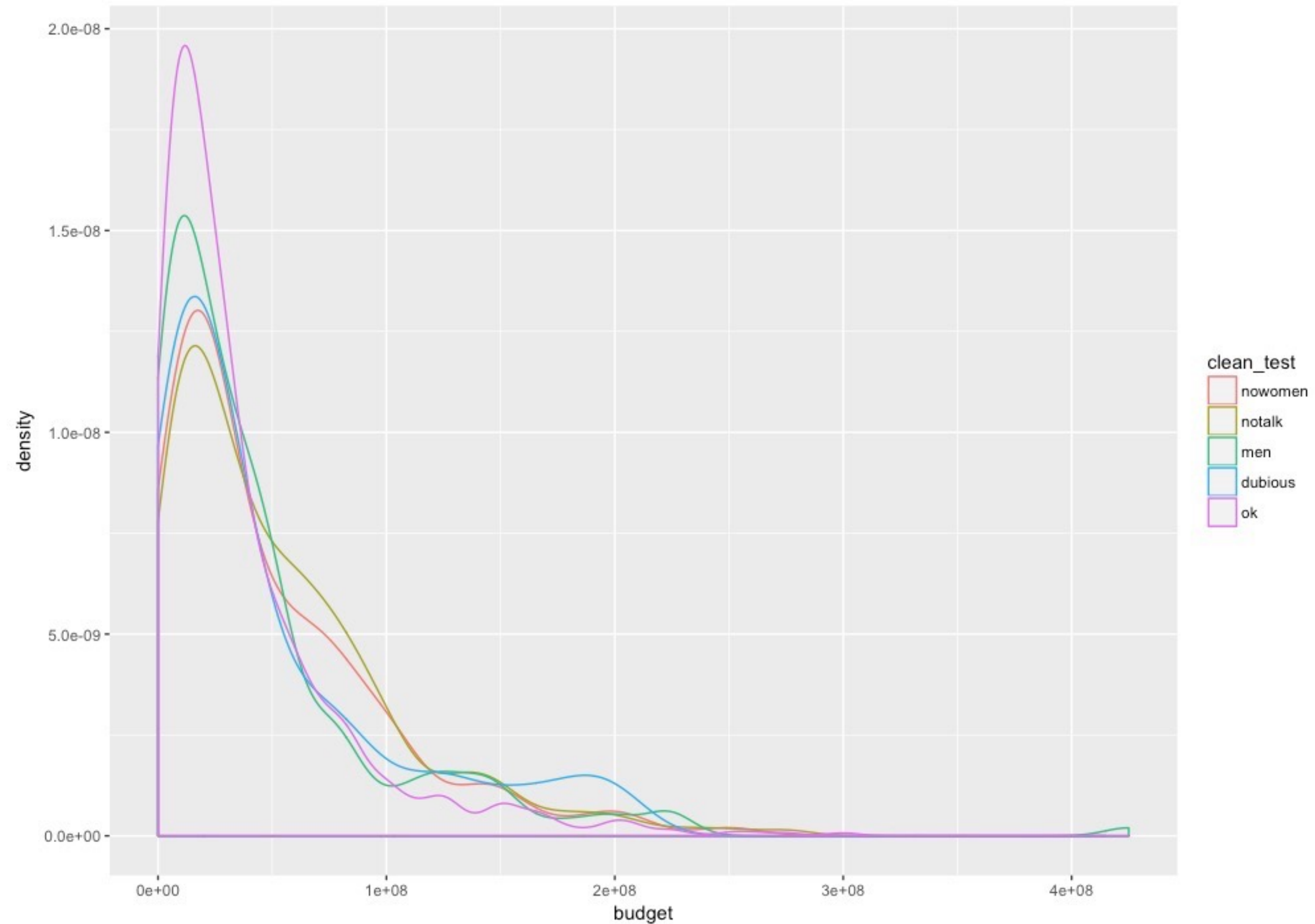


02:00

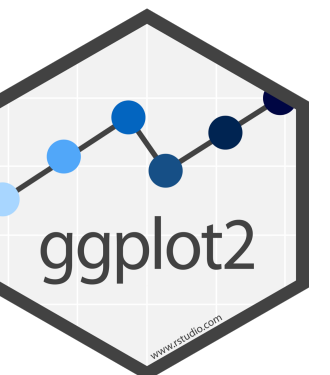


```
ggplot(data = bechdel) +  
  geom_density(mapping = aes(x = budget))
```



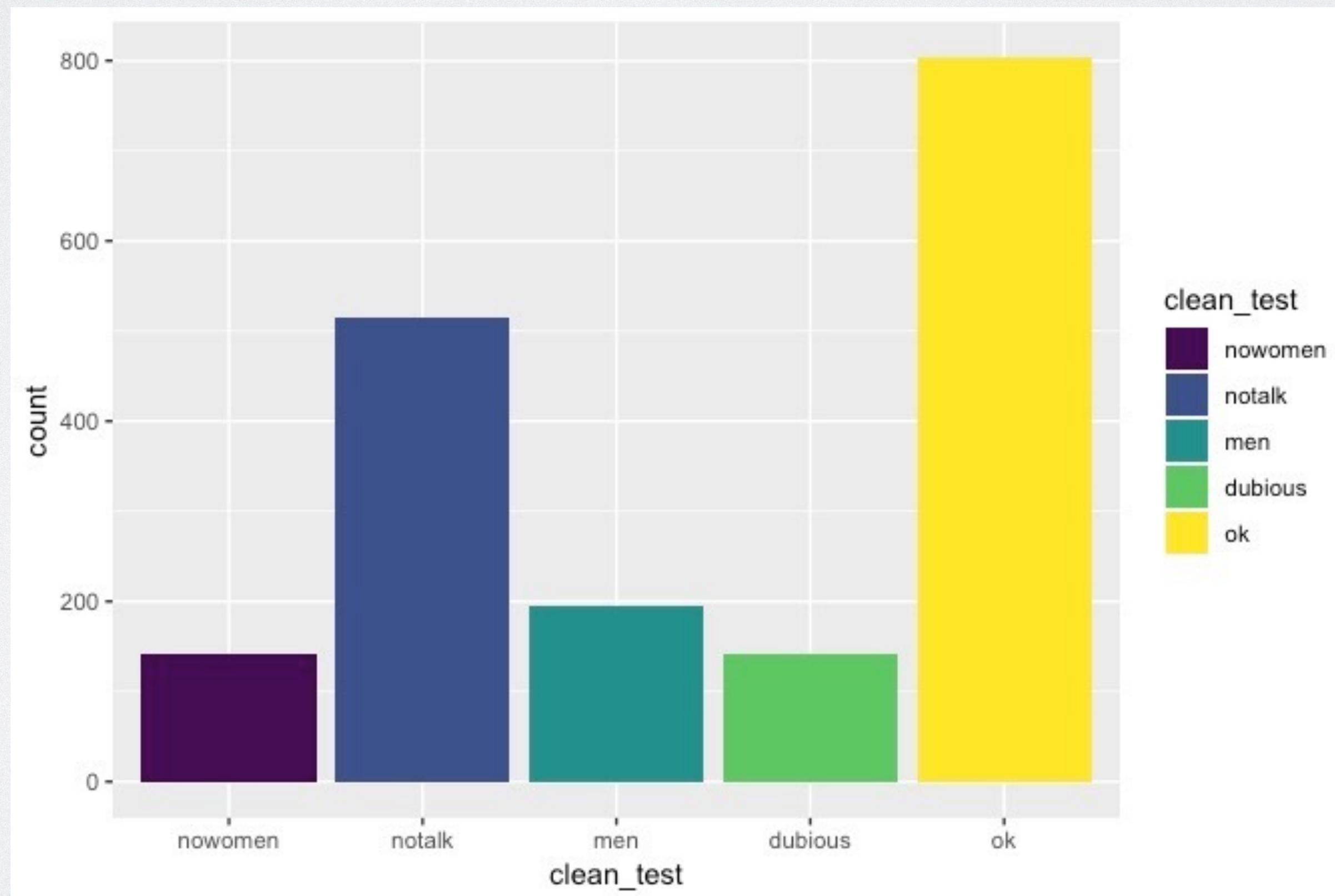


```
ggplot(data = bechdel) +  
  geom_density(mapping = aes(x = budget, color=clean_test))
```

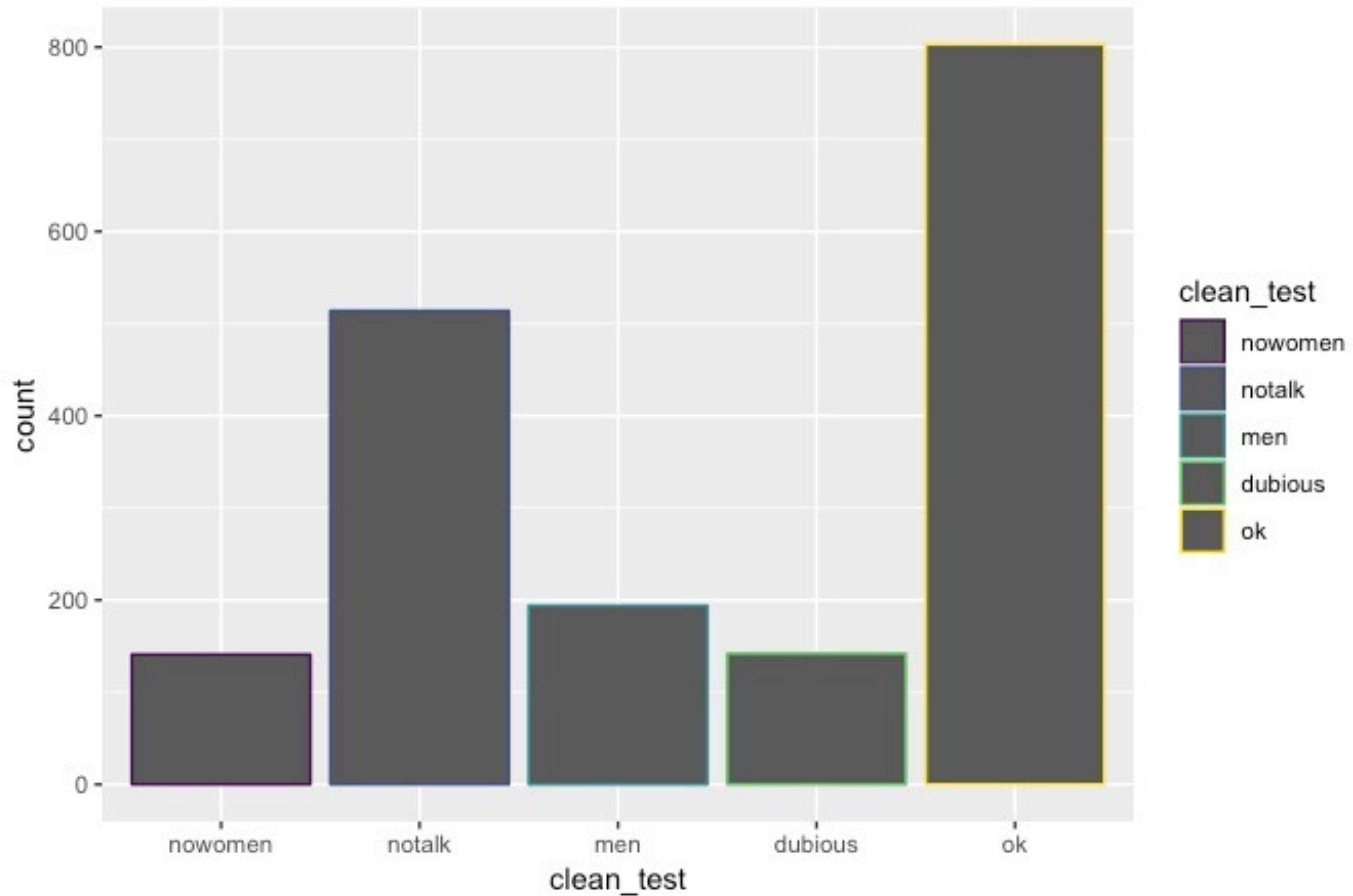


Your Turn 7

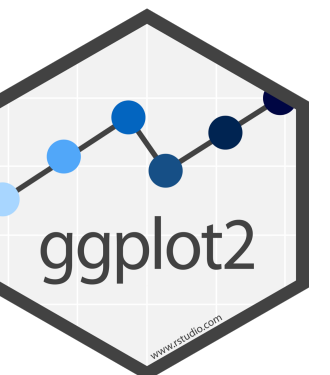
With your partner, make the bar chart of `clean_test` colored by `clean_test` below. Use the cheatsheet. Try your best guess.

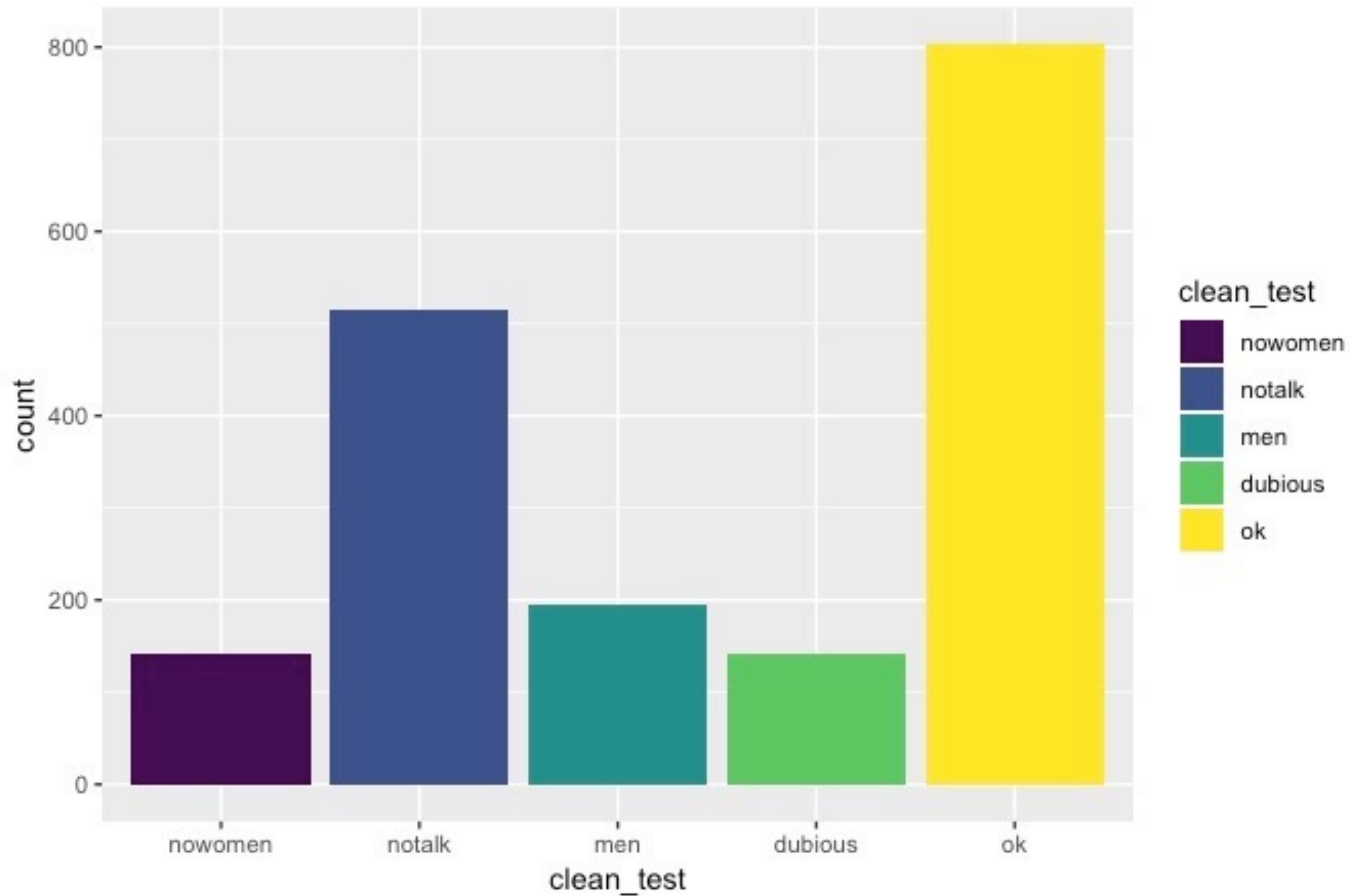


02:00

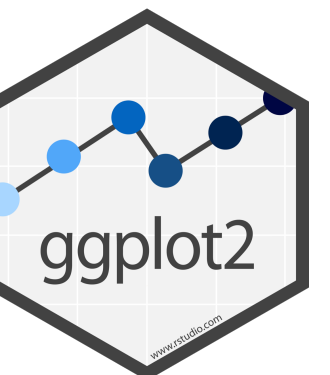


```
ggplot(data=bechdel) +  
  geom_bar(aes(x=clean_test, color=clean_test))
```





```
ggplot(data=bechdel) +  
  geom_bar(aes(x=clean_test, fill=clean_test))
```



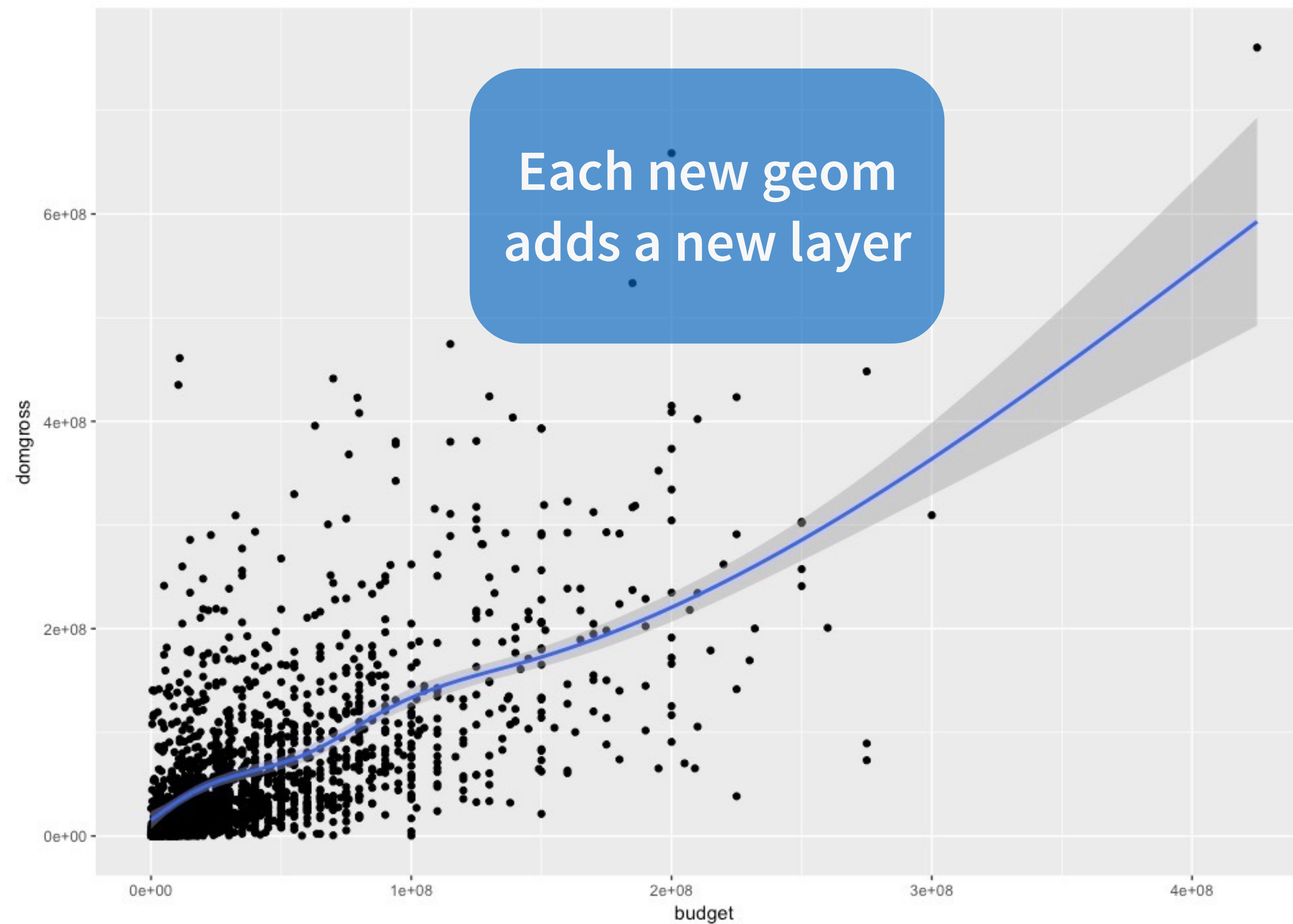
Your Turn 8

With a partner, predict what this code will do.

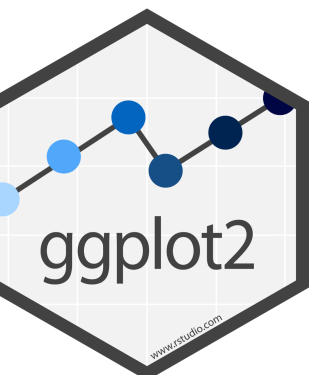
Then run it.

```
ggplot(data = bechdel) +  
  geom_point(mapping = aes(x = budget, y = domgross)) +  
  geom_smooth(mapping = aes(x = budget, y = domgross))
```

03:00

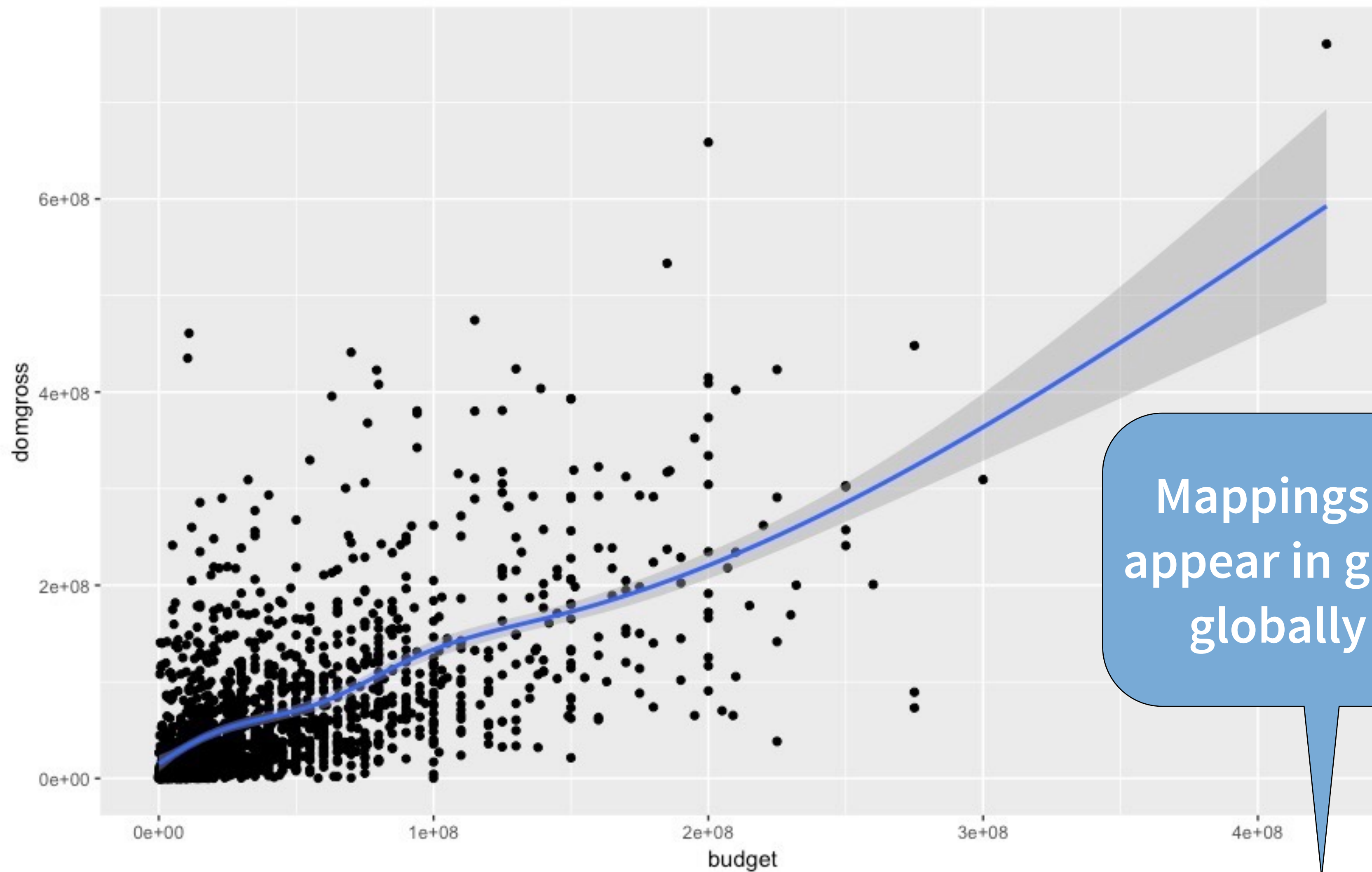


```
ggplot(data = bechdel) +  
  geom_point(mapping = aes(x = budget, y = domgross)) +  
  geom_smooth(mapping = aes(x = budget, y = domgross))
```

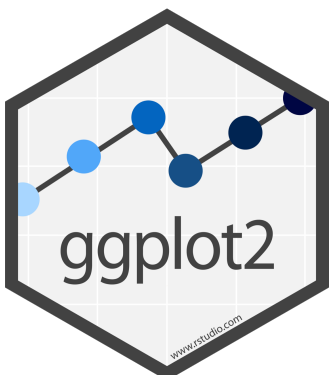


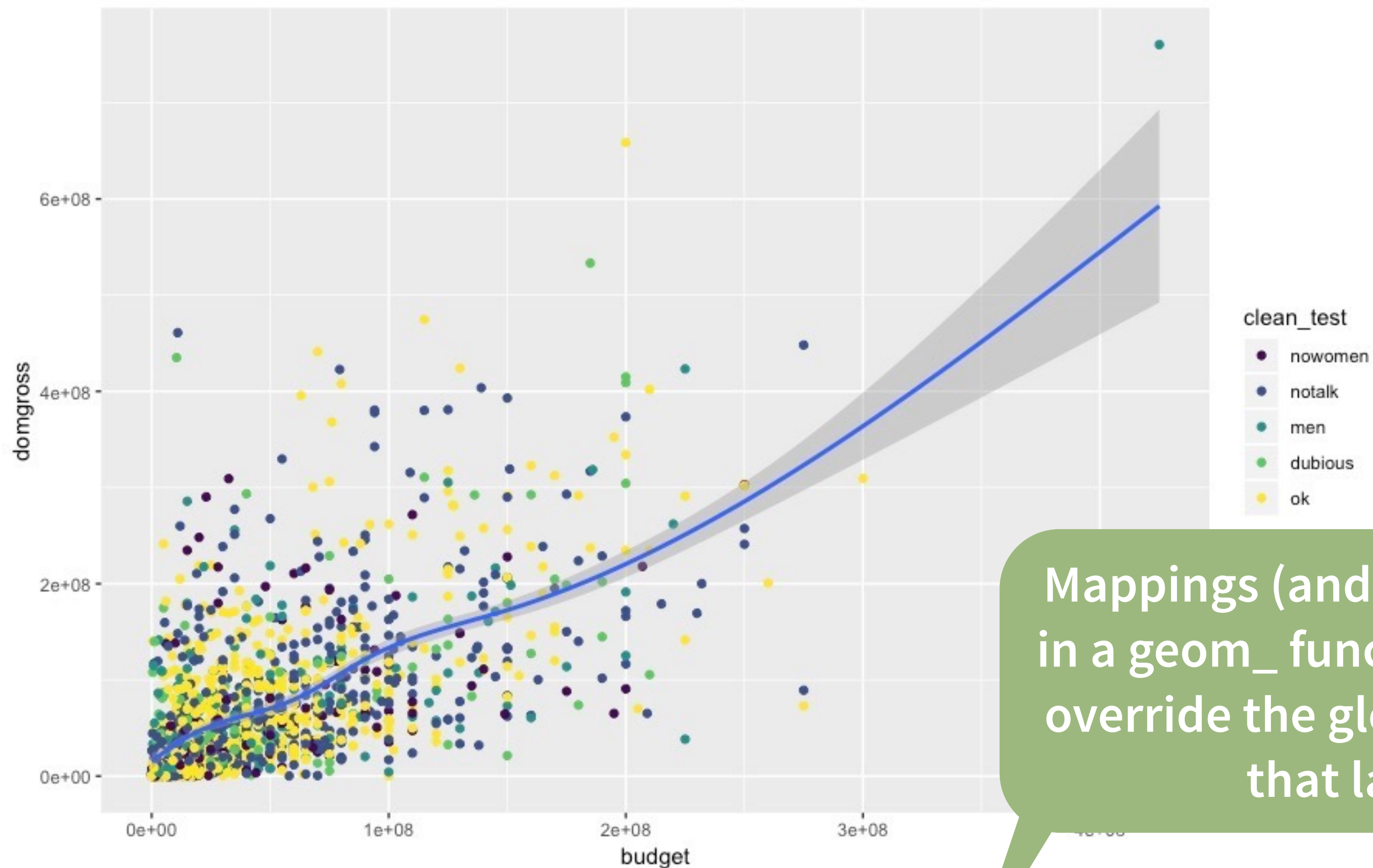
global vs. local





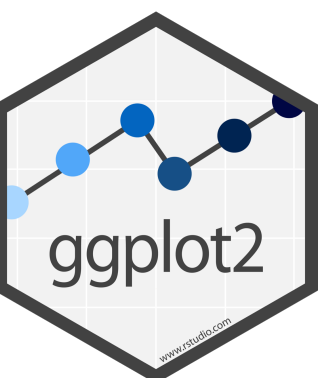
```
ggplot(data = bechdel, mapping = aes(x = budget, y = domgross)) +  
  geom_point() +  
  geom_smooth()
```

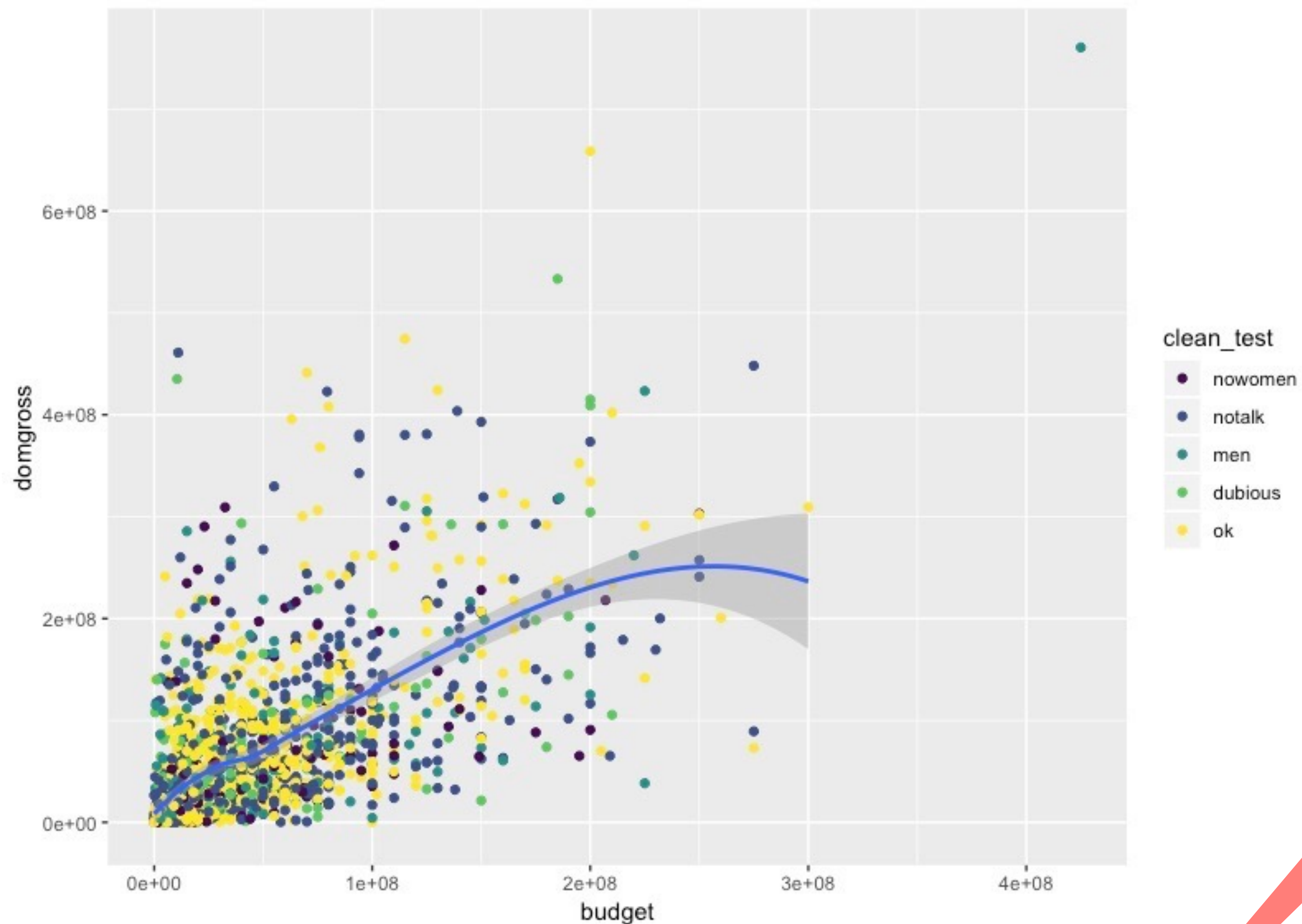




Mappings (and data) that appear in a geom_function will add to or override the global mappings for that layer only

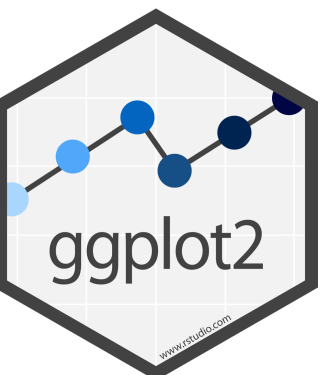
```
ggplot(data = bechdel, mapping = aes(x = budget, y = domgross)) +
  geom_point(mapping = aes(color = clean_test)) +
  geom_smooth()
```





data can also be set locally or globally

```
ggplot(data = bechdel, mapping = aes(x = budget, y = domgross)) +
  geom_point(mapping = aes(color = clean_test)) +
  geom_smooth(data = filter(bechdel, clean_test == "ok"))
```



Saving graphs



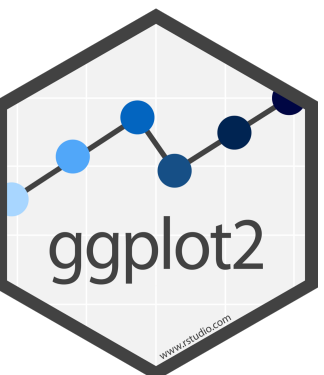
R Notebook Preview

The easiest way to save all your work (including graphs) is to include it in an R Notebook. While you're working, you can view your notebook by clicking "Preview"

The screenshot shows the RStudio interface with an R Notebook in preview mode. The notebook content is as follows:

```
1 ---
2 title: "Visualization"
3 output: html_notebook
4 editor_options:
5   chunk_output_type: inline
6 ---
7 |
8 ## Setup
9
10 The first chunk in an R Notebook is usually
    titled "setup," and by convention includes the
    R packages you want to load. Remember, in order
    to use an R package you have to run some
    `library()` code every session. Execute these
    lines of code to load the packages.
11
12 ```{r setup}
13 library(ggplot2)
14 library(fivethirtyeight)
15 ```
16
17 ## Bechdel test data
```

The viewer pane displays a scatter plot of 'domgross' (y-axis, 0e+00 to 6e+08) versus 'budget' (x-axis, 0e+00 to 4e+08). The plot features a blue trend line and a grey shaded confidence interval.



R Notebook Preview

Now, you'll see a beautifully typeset version of what you've done!

The screenshot displays the RStudio interface with an R Notebook in preview mode. The editor on the left shows the following R Markdown code:

```
1 ---
2 title: "Visualization"
3 output: html_notebook
4 editor_options:
5   chunk_output_type: inline
6 ---
7 |
8 ## Setup
9
10 The first chunk in an R Notebook is usually
    titled "setup," and by convention includes the
    R packages you want to load. Remember, in order
    to use an R package you have to run some
```

The preview pane on the right shows the rendered HTML output, including a "Setup" section with the following code:

```
library(ggplot2)
library(fivethirtyeight)
```

The "Bechdel test data" section contains the following text:

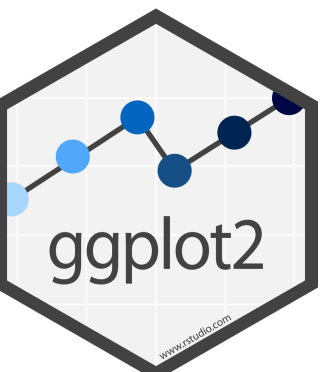
We're going to start by playing with data collected by the website FiveThirtyEight on movies and [the Bechdel test](#).

To begin, let's just preview our data. There are a couple ways to do that. One is just to type the name of the data and execute it like a piece of code.

```
bechdel
```

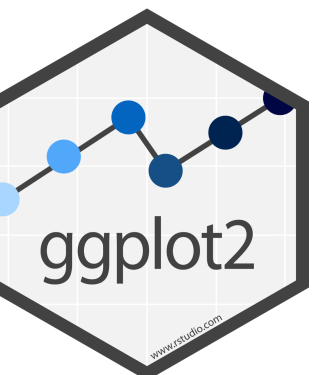
The console at the bottom shows the execution of the following code:

```
geom_point(aes(budget, domgross)) +
+ geom_smooth(aes(budget, domgross))
`geom_smooth()` using method = 'gam'
Warning messages:
1: Removed 17 rows containing non-finite values
  (stat_smooth).
2: Removed 17 rows containing missing values
  (geom_point).
> |
```



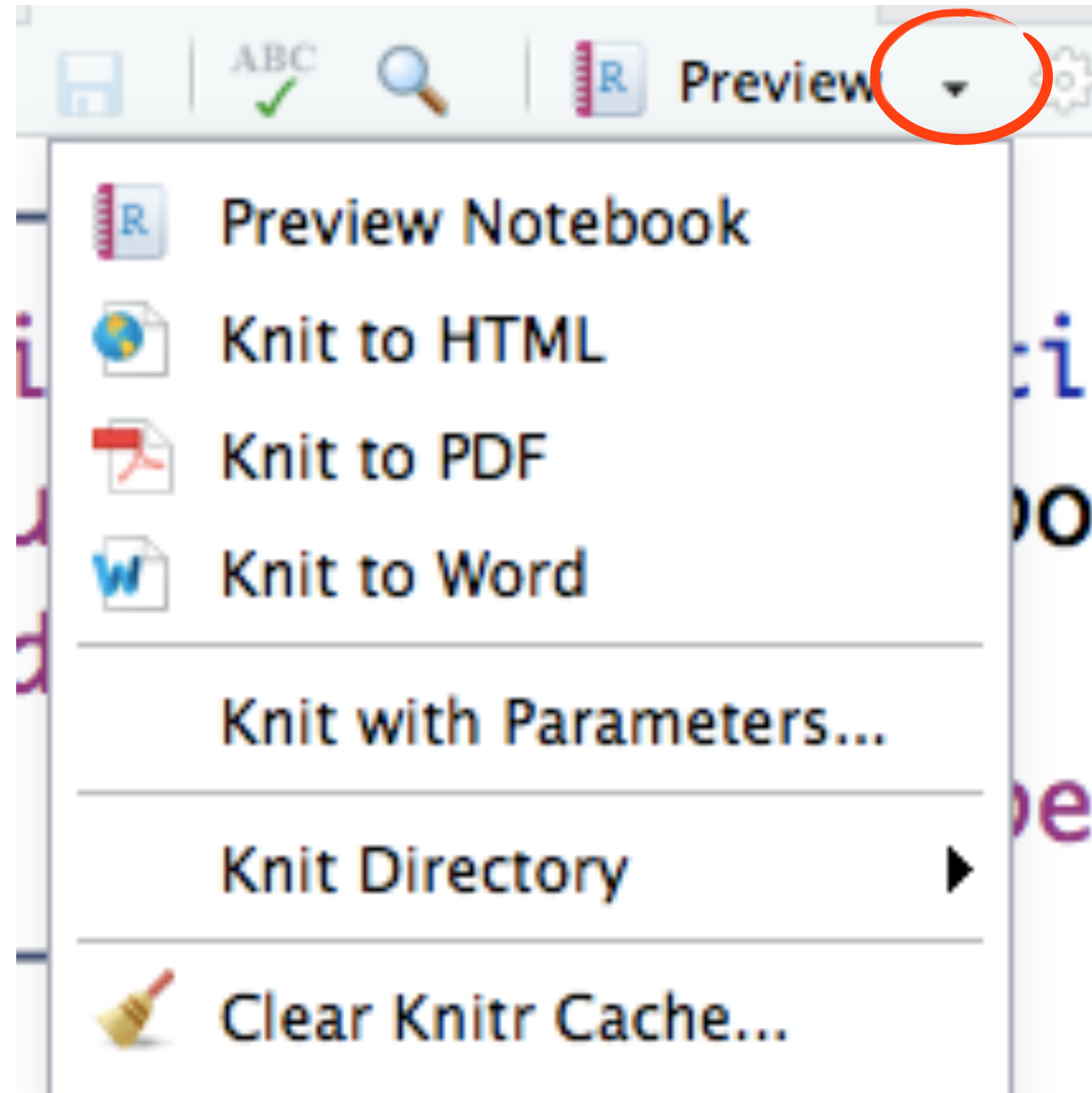
Sharing your work

The Preview is something only available to you, but RStudio automatically creates a file you can share with others when you save an R Notebook. The file it creates is an HTML file, and it has a name that corresponds to your Rmd



"knitting" a document

While RStudio automatically generates an HTML file of your work, you might want a different format. Clicking the down arrow next to Preview lets you see other options.



Your Turn

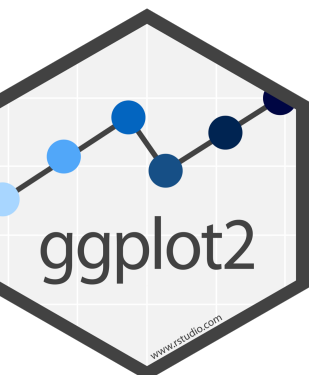
Locate the 01-Visualize.nb.html file in your Files pane. It will be located in your **working directory**

A digital timer with a black border and white background, displaying the time 00:30 in a black, segmented font.

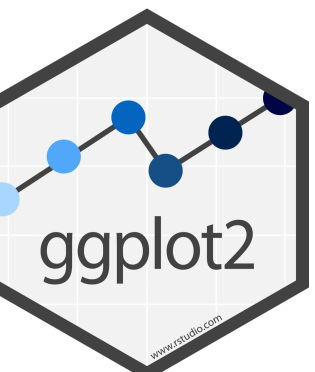
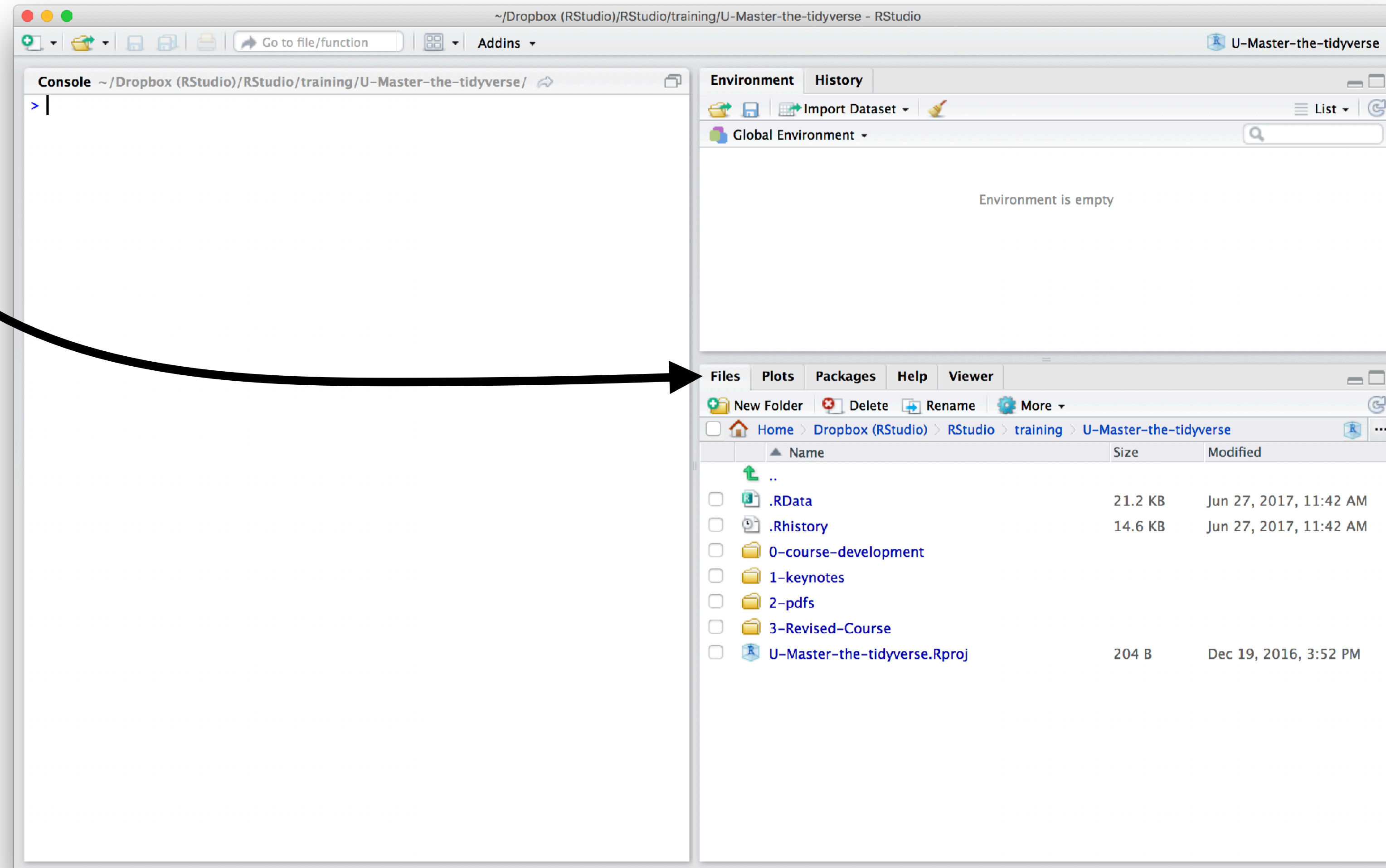
Working directory

R associates itself with a folder (i.e. directory) on your computer.

- This folder is known as your "**working directory**"
- When you save files, R will save them here
- When you load files, R will look for them here

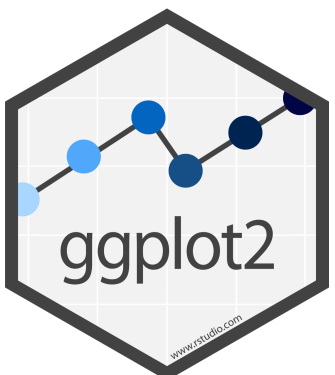
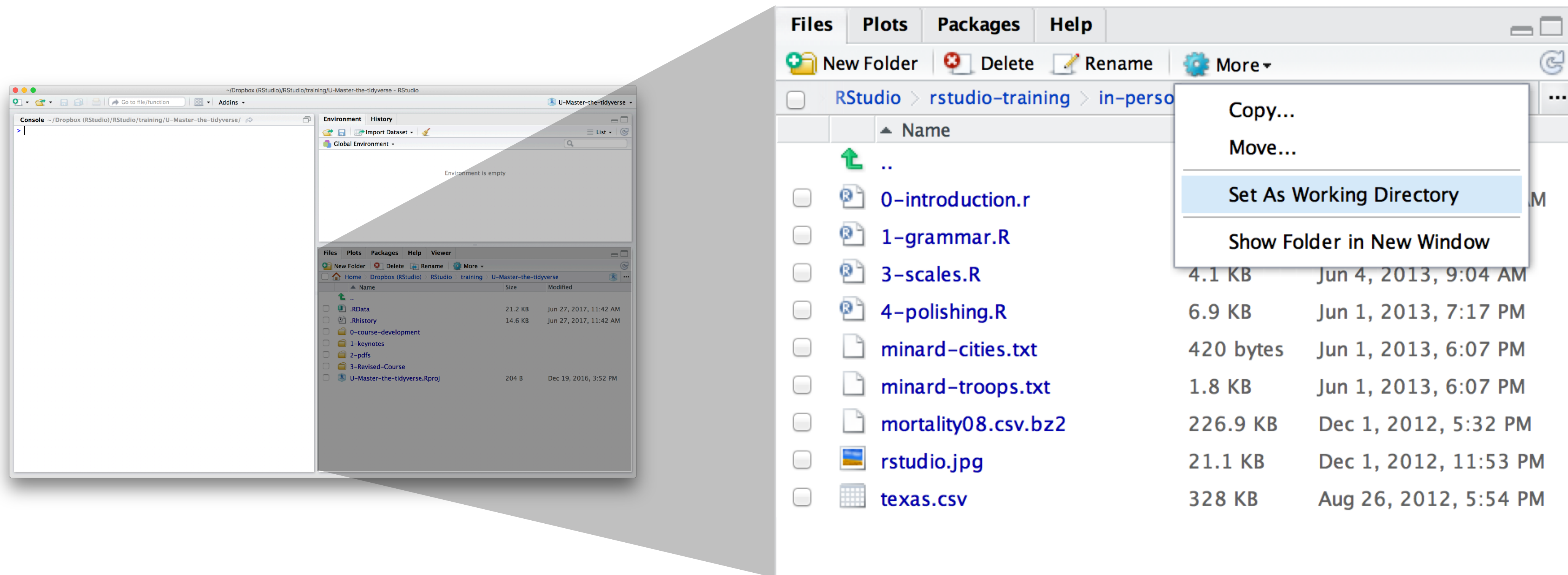


The files pane of the IDE displays the contents of your working directory



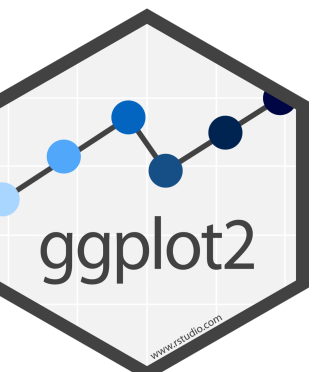
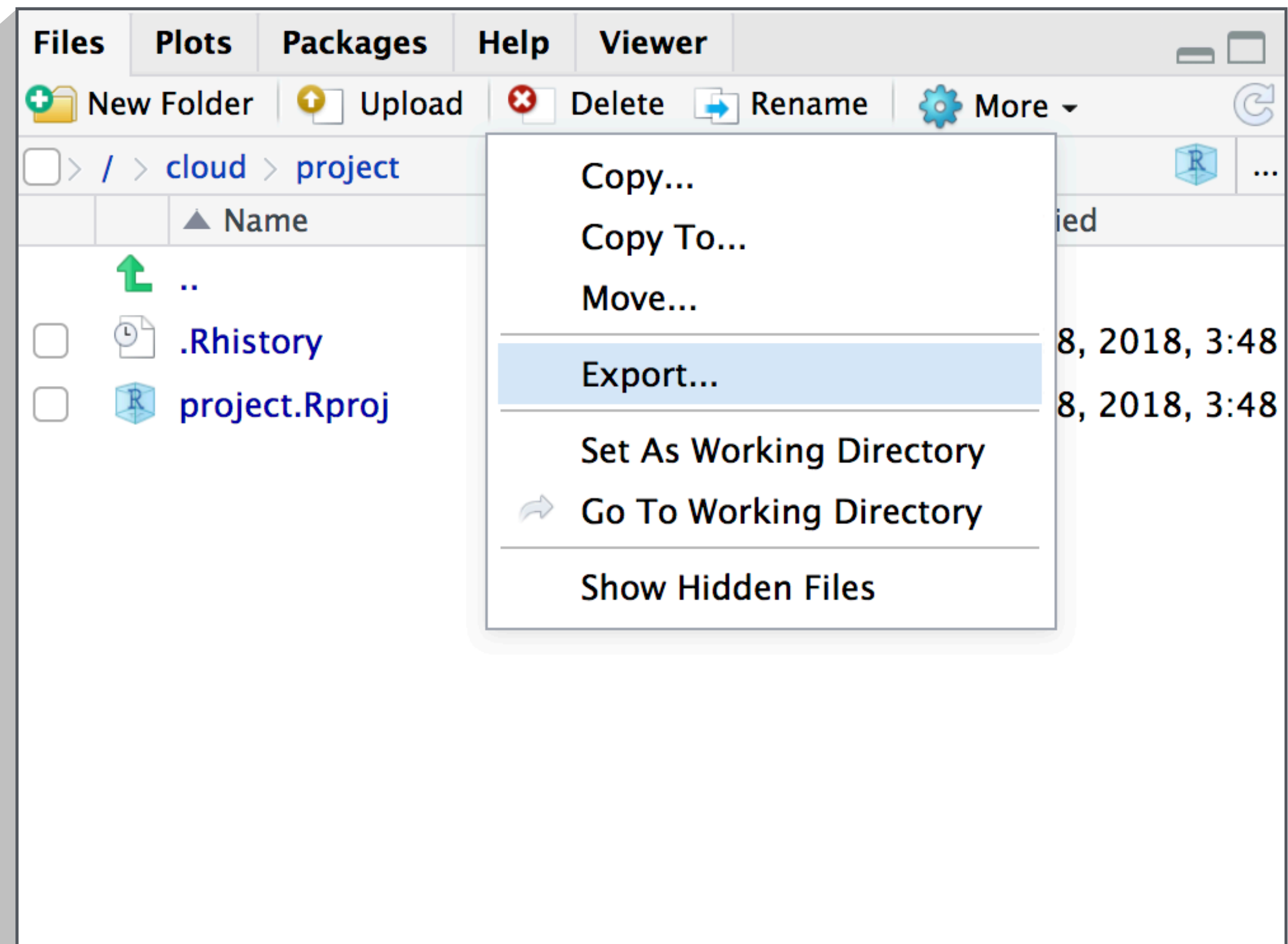
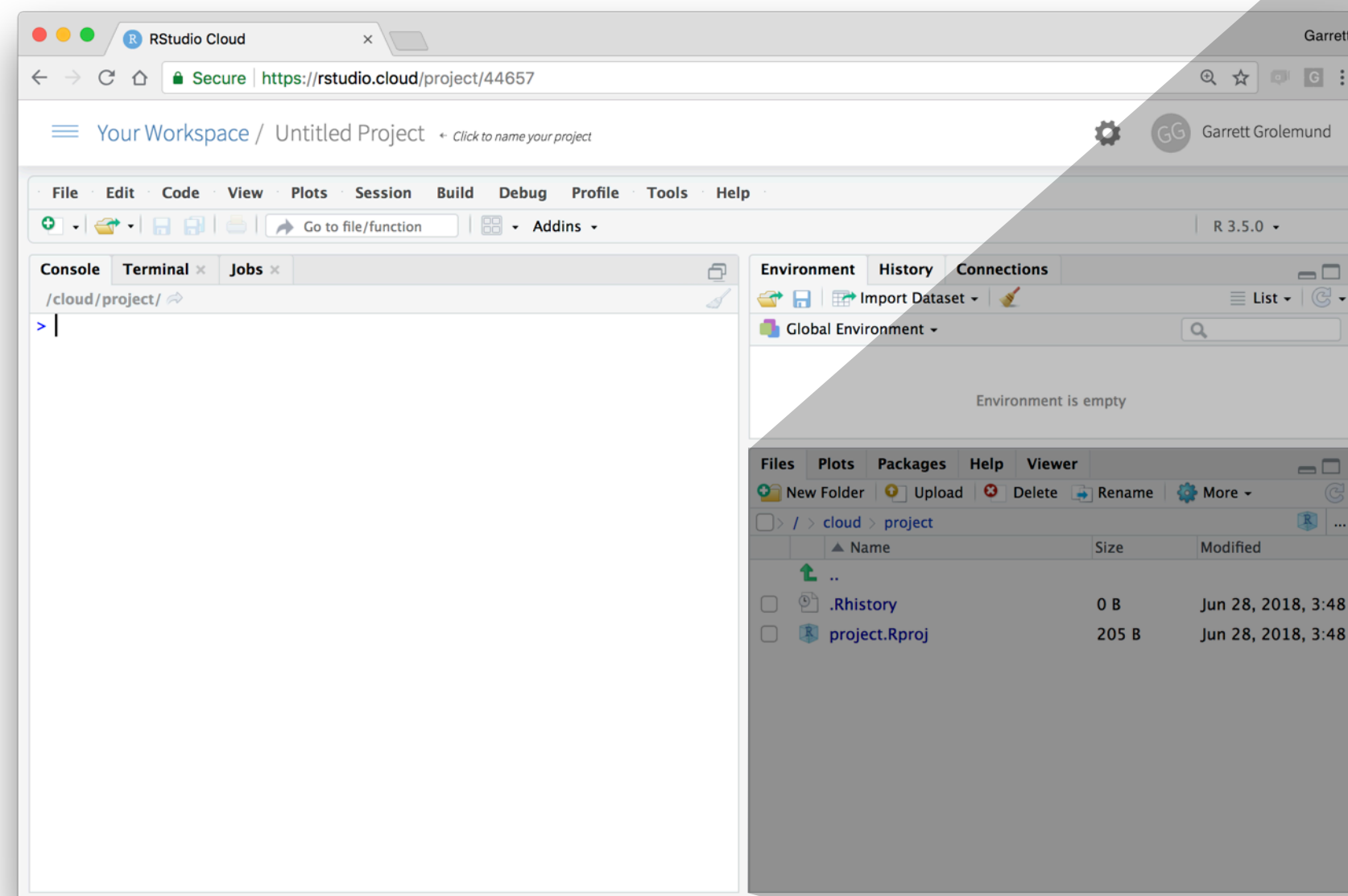
Changing the Working directory

Navigate in the files pane to a new directory. Click **More > Set As Working Directory**



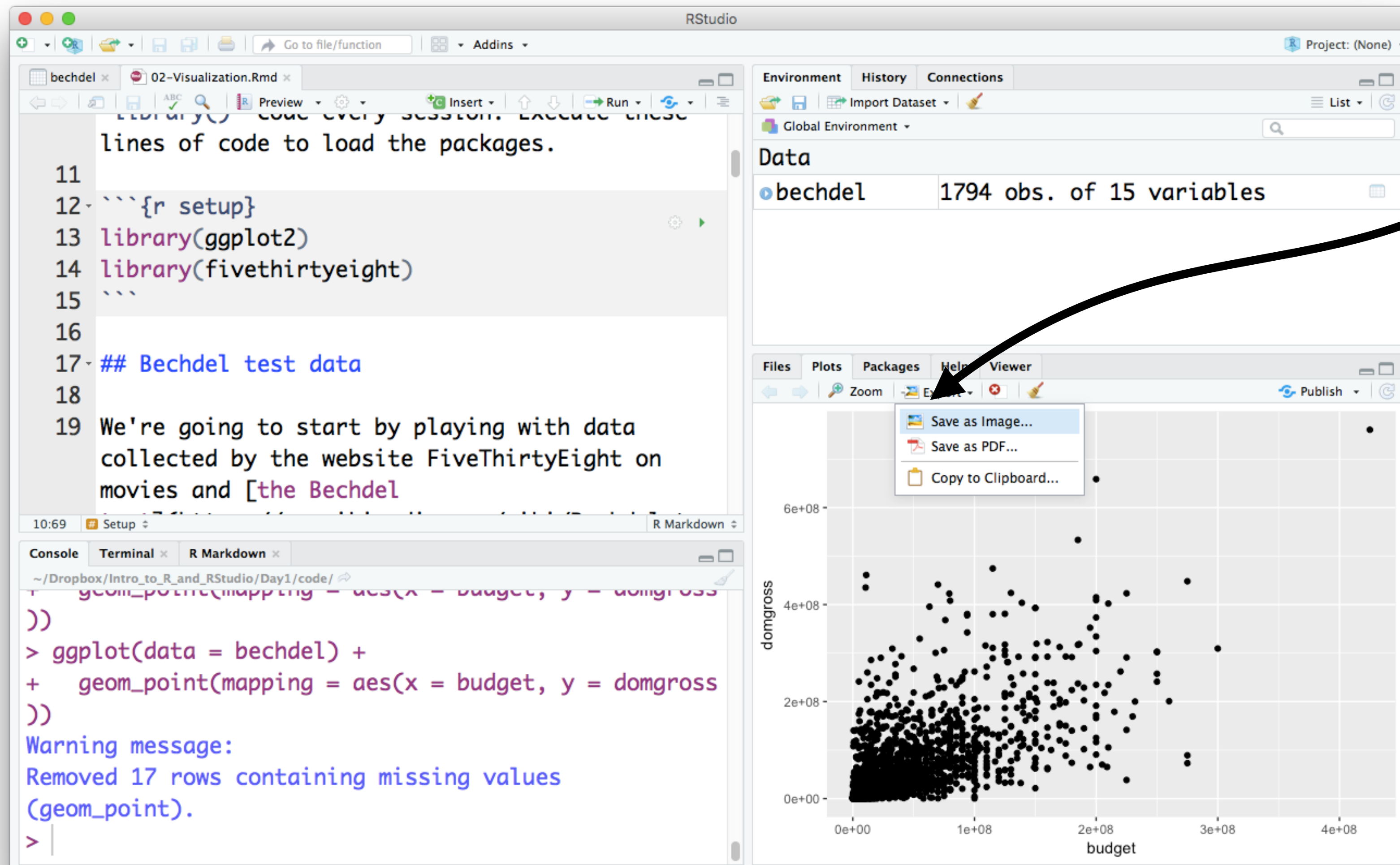
Download files

In the files pane, check next to the file(s) to download
More > Export



Manually saving plots

Save plots manually with the export menu



The screenshot shows the RStudio interface. The top-left pane displays R code for loading packages and plotting data. The bottom-left pane shows the console output, including a warning message about missing values. The top-right pane shows the Environment tab with the 'bechdel' data frame. The bottom-right pane shows a scatter plot of 'domgross' vs 'budget' with an export menu open, highlighting 'Save as Image...'. A large black arrow points from the text 'Save plots manually with the export menu' to the 'Export' menu in the plot viewer.

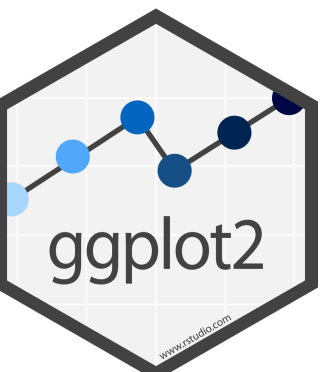
```
11 lines of code to load the packages.  
12- ```{r setup}  
13 library(ggplot2)  
14 library(fivethirtyeight)  
15 ```  
16  
17- ## Bechdel test data  
18  
19 We're going to start by playing with data  
collected by the website FiveThirtyEight on  
movies and [the Bechdel
```

```
> ggplot(data = bechdel) +  
+ geom_point(mapping = aes(x = budget, y = domgross  
)  
)  
Warning message:  
Removed 17 rows containing missing values  
(geom_point).  
>
```

Environment History Connections
Global Environment
Data
bechdel 1794 obs. of 15 variables

Files Plots Packages Help Viewer
Zoom Export
Save as Image...
Save as PDF...
Copy to Clipboard...

domgross
6e+08
4e+08
2e+08
0e+00
0e+00 1e+08 2e+08 3e+08 4e+08
budget



Saving plots

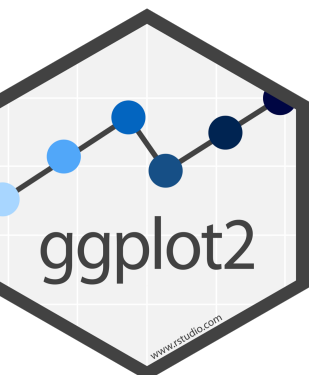
`ggsave()` saves the last plot.

Uses size on screen:

```
ggsave("my-plot.pdf")  
ggsave("my-plot.png")
```

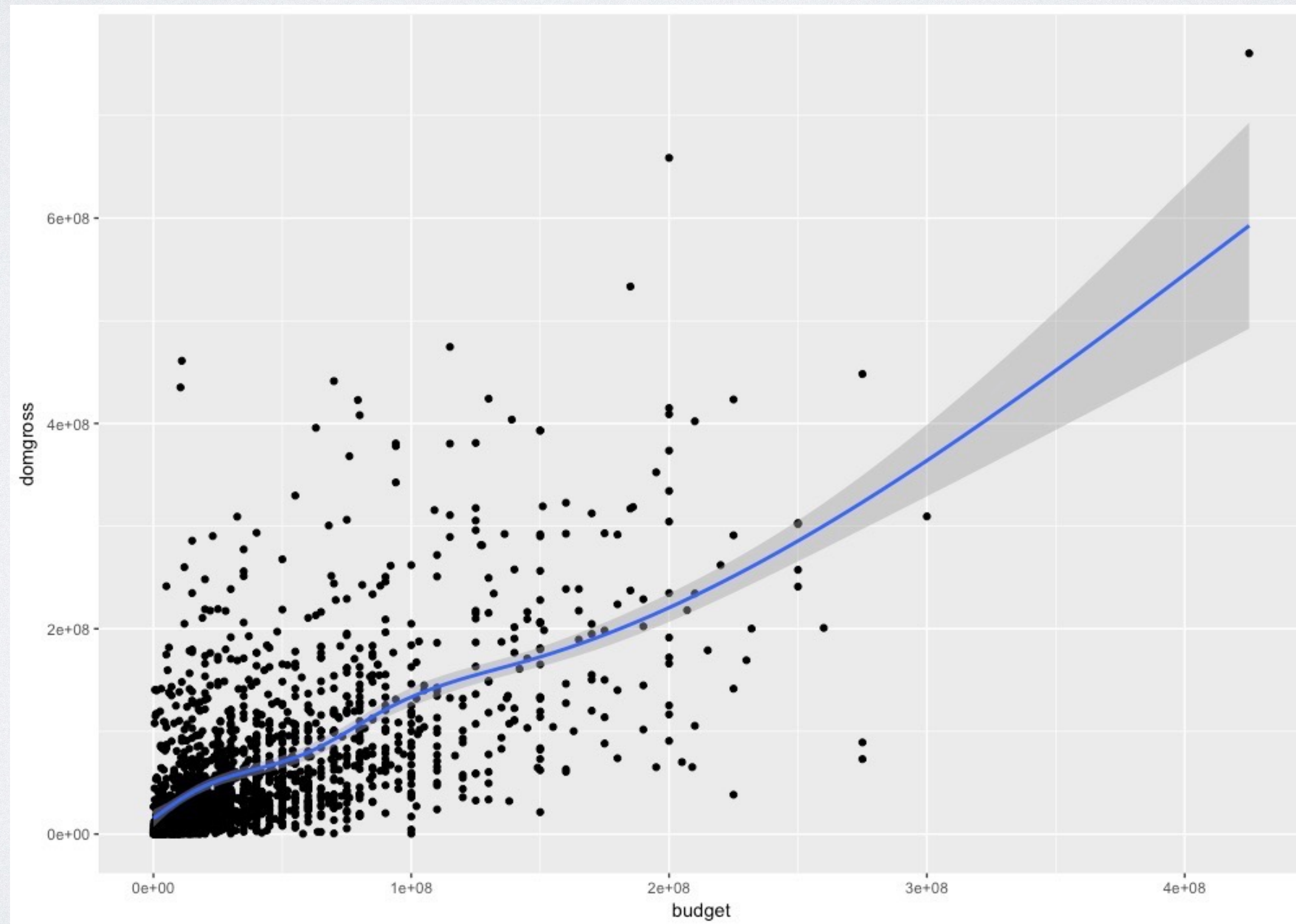
Specify size in inches

```
ggsave("my-plot.pdf", width = 6, height = 6)
```



Your Turn 9

Save your last plot and then locate it in your files pane and download it. (You may have to refresh the files list).

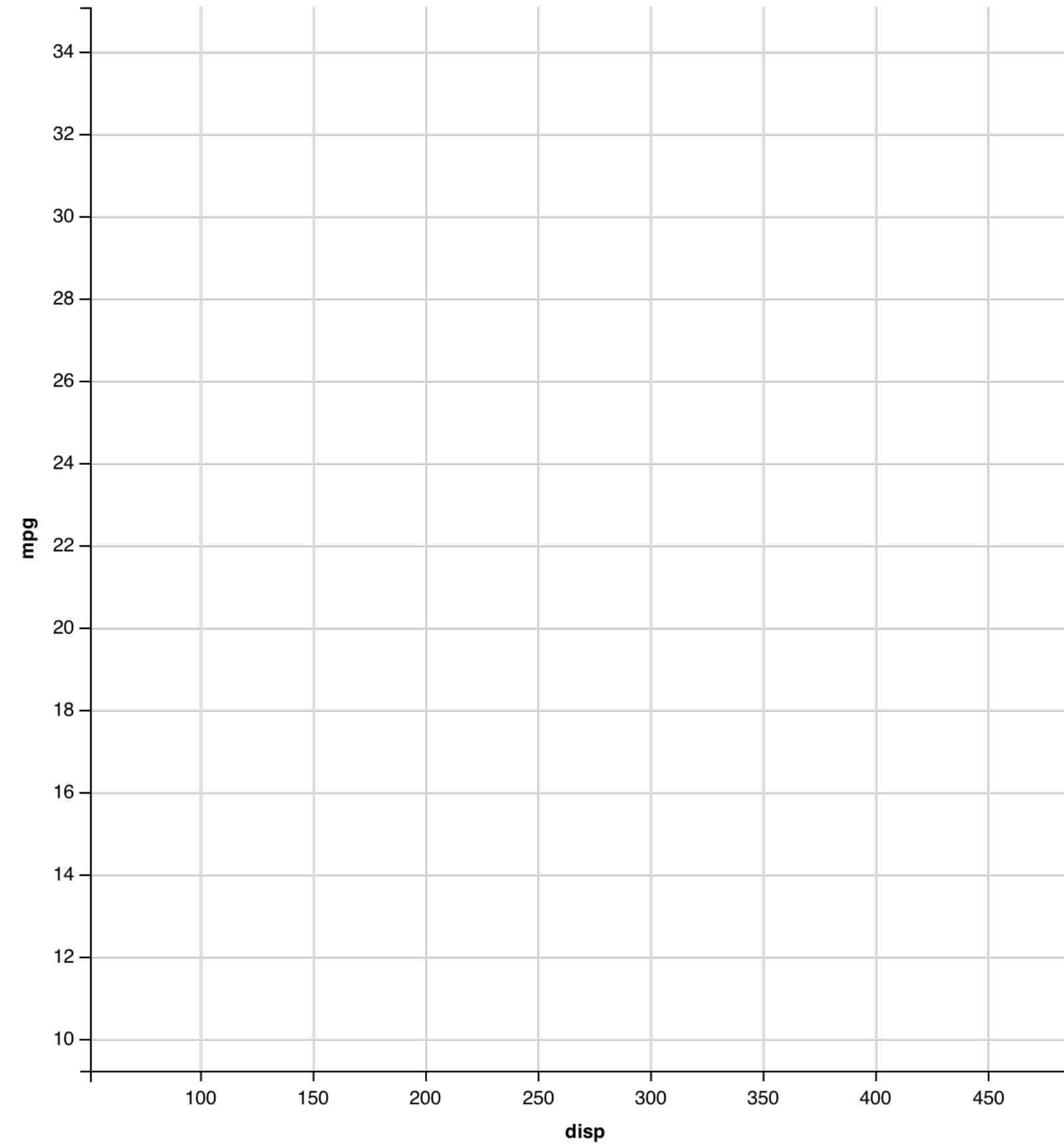


01:00

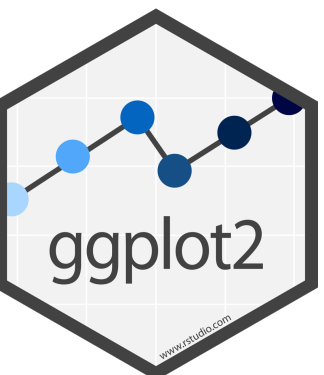
Grammar of Graphics



| mpg | cyl | disp | hp |
|------|-----|-------|----|
| 21.0 | 6 | 160.0 | 2 |
| 21.0 | 6 | 160.0 | 2 |
| 22.8 | 4 | 108.0 | 1 |
| 21.4 | 6 | 258.0 | 2 |
| 18.7 | 8 | 360.0 | 3 |
| 18.1 | 6 | 225.0 | 2 |
| 14.3 | 8 | 360.0 | 5 |
| 24.4 | 4 | 146.7 | 1 |
| 22.8 | 4 | 140.8 | 1 |
| 19.2 | 6 | 167.6 | 2 |
| 17.8 | 6 | 167.6 | 2 |
| 16.4 | 8 | 275.8 | 3 |
| 17.3 | 8 | 275.8 | 3 |
| 15.2 | 8 | 275.8 | 3 |
| 10.4 | 8 | 472.0 | 4 |
| 10.4 | 8 | 460.0 | 4 |
| 14.7 | 8 | 440.0 | 4 |
| 32.4 | 4 | 78.7 | 1 |
| 30.4 | 4 | 75.7 | 1 |
| 33.9 | 4 | 71.1 | 1 |



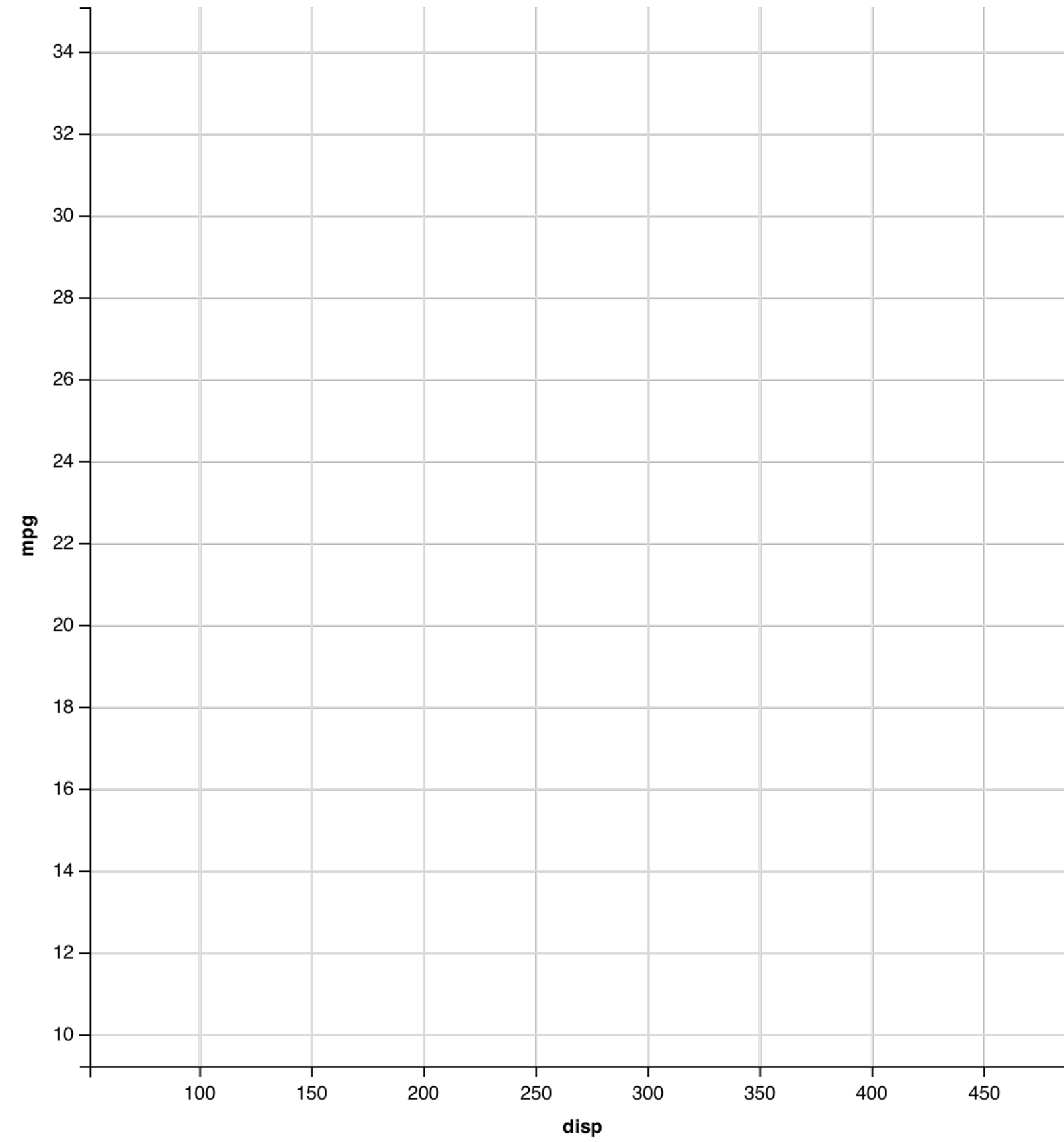
data geom



mappings

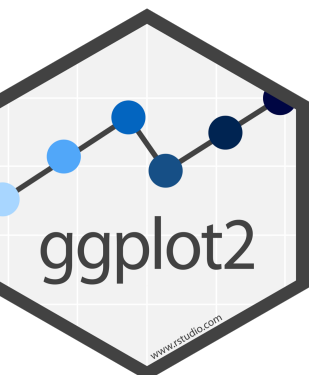
fill
↑↓

| mpg | cyl | disp | hp | fill |
|------|-----|-------|----|------|
| 21.0 | 6 | 160.0 | 2 | ● |
| 21.0 | 6 | 160.0 | 2 | ● |
| 22.8 | 4 | 108.0 | 1 | ● |
| 21.4 | 6 | 258.0 | 2 | ● |
| 18.7 | 8 | 360.0 | 3 | ● |
| 18.1 | 6 | 225.0 | 2 | ● |
| 14.3 | 8 | 360.0 | 5 | ● |
| 24.4 | 4 | 146.7 | 1 | ● |
| 22.8 | 4 | 140.8 | 1 | ● |
| 19.2 | 6 | 167.6 | 2 | ● |
| 17.8 | 6 | 167.6 | 2 | ● |
| 16.4 | 8 | 275.8 | 3 | ● |
| 17.3 | 8 | 275.8 | 3 | ● |
| 15.2 | 8 | 275.8 | 3 | ● |
| 10.4 | 8 | 472.0 | 4 | ● |
| 10.4 | 8 | 460.0 | 4 | ● |
| 14.7 | 8 | 440.0 | 4 | ● |
| 32.4 | 4 | 78.7 | 1 | ● |
| 30.4 | 4 | 75.7 | 1 | ● |
| 33.9 | 4 | 71.1 | 1 | ● |



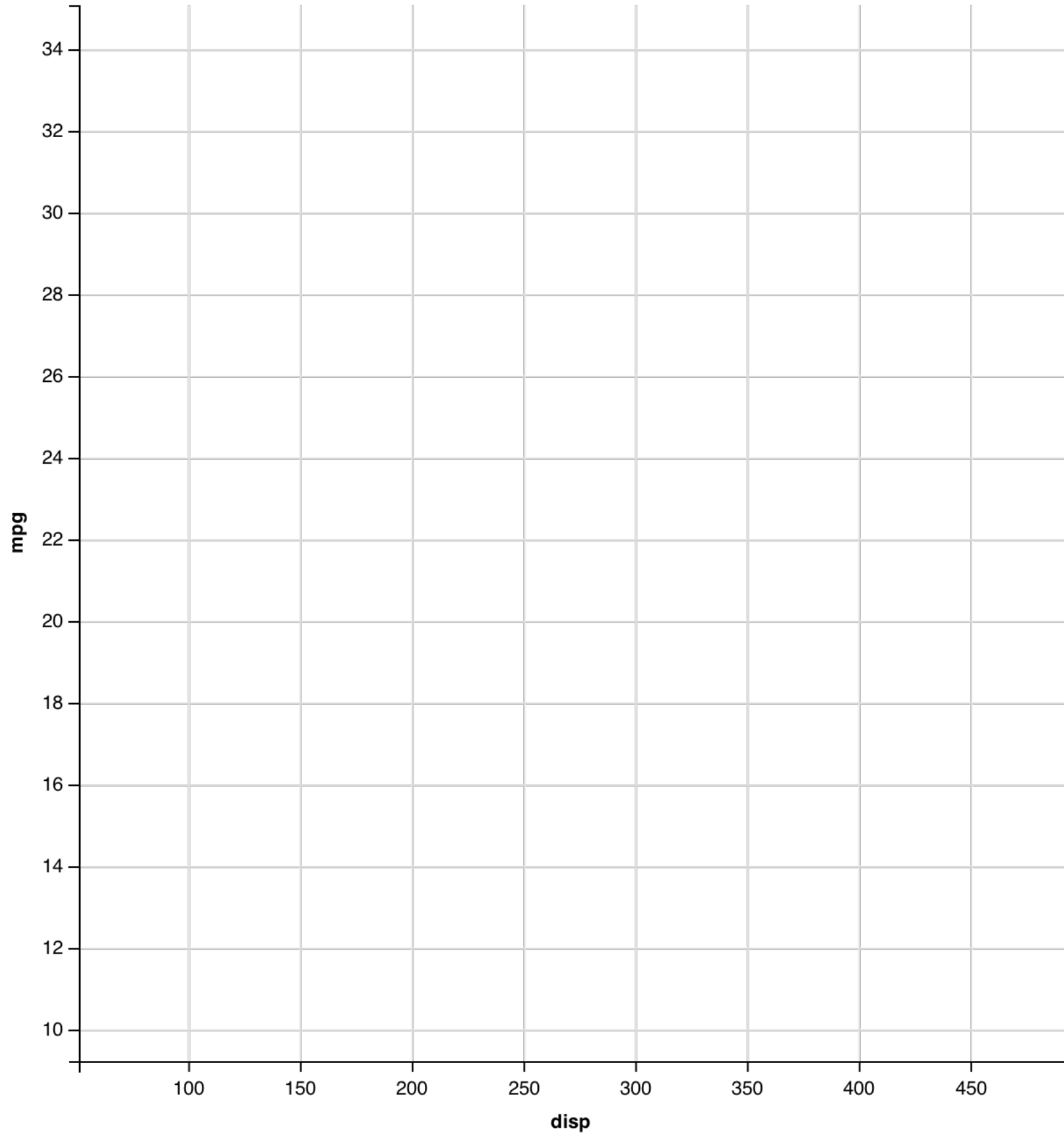
data

geom



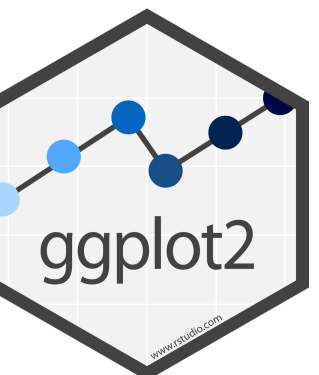
mappings

| mpg | shape | | hp | fill |
|------|-------|-------|----|------|
| | cyl | disp | | |
| 21.0 | 6 + | 160.0 | 2 | + |
| 21.0 | 6 + | 160.0 | 2 | + |
| 22.8 | 4 ● | 108.0 | 1 | ● |
| 21.4 | 6 + | 258.0 | 2 | + |
| 18.7 | 8 ◆ | 360.0 | 3 | ◆ |
| 18.1 | 6 + | 225.0 | 2 | + |
| 14.3 | 8 ◆ | 360.0 | 5 | ◆ |
| 24.4 | 4 ● | 146.7 | 1 | ● |
| 22.8 | 4 ● | 140.8 | 1 | ● |
| 19.2 | 6 + | 167.6 | 2 | + |
| 17.8 | 6 + | 167.6 | 2 | + |
| 16.4 | 8 ◆ | 275.8 | 3 | ◆ |
| 17.3 | 8 ◆ | 275.8 | 3 | ◆ |
| 15.2 | 8 ◆ | 275.8 | 3 | ◆ |
| 10.4 | 8 ◆ | 472.0 | 4 | ◆ |
| 10.4 | 8 ◆ | 460.0 | 4 | ◆ |
| 14.7 | 8 ◆ | 440.0 | 4 | ◆ |
| 32.4 | 4 ● | 78.7 | 1 | ● |
| 30.4 | 4 ● | 75.7 | 1 | ● |
| 33.9 | 4 ● | 71.1 | 1 | ● |



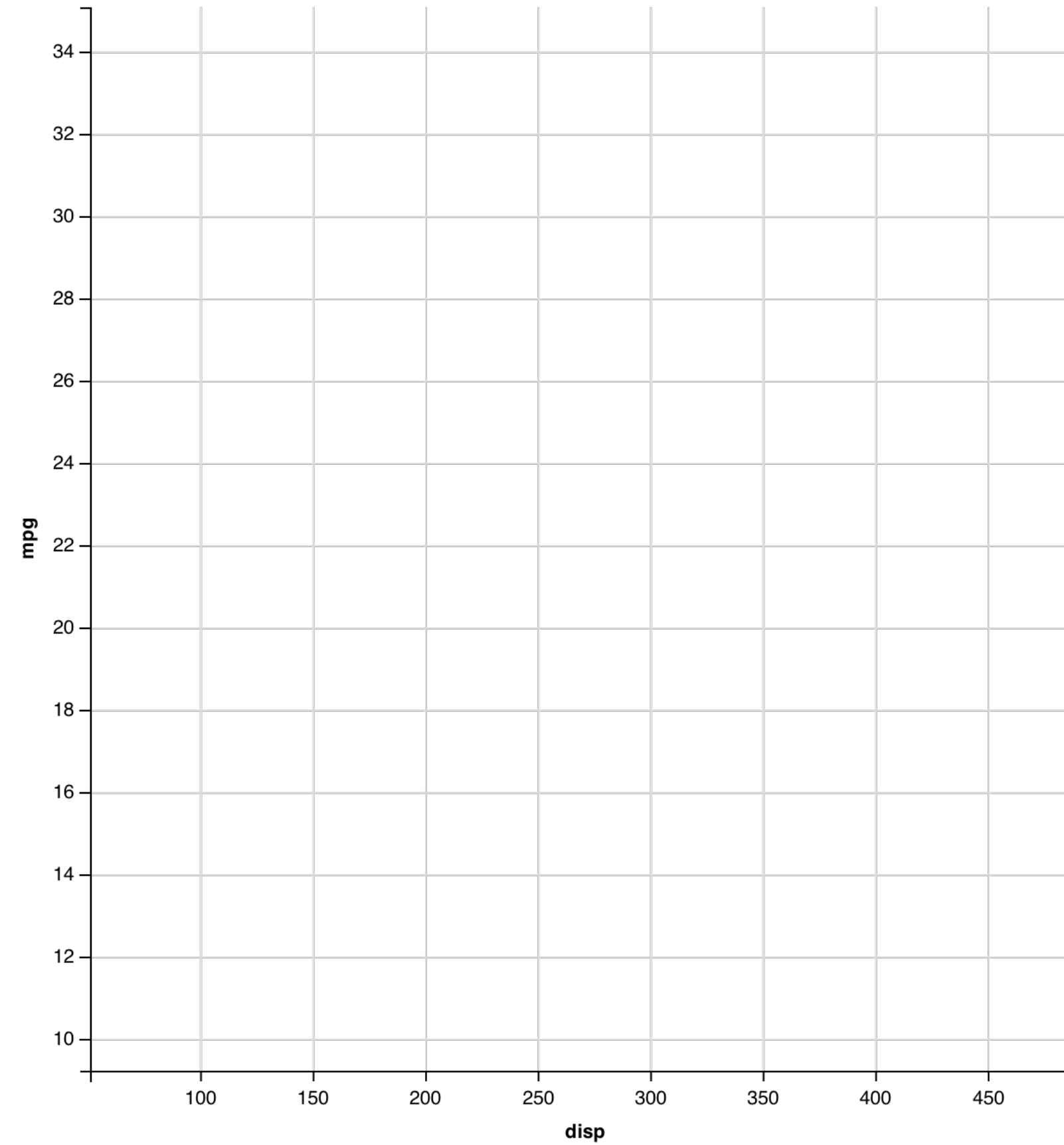
data

geom



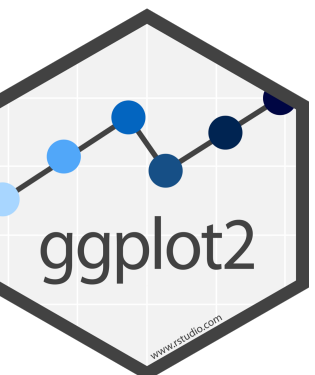
mappings

| mpg | shape
↑↑
cyl | x
↑↑
disp | fill
↑↑
hp | |
|------|---------------------------|------------------------|-------------------------|---|
| 21.0 | 6 | 160.0 | 2 | + |
| 21.0 | 6 | 160.0 | 2 | + |
| 22.8 | 4 | 108.0 | 1 | ● |
| 21.4 | 6 | 258.0 | 2 | + |
| 18.7 | 8 | 360.0 | 3 | ◆ |
| 18.1 | 6 | 225.0 | 2 | + |
| 14.3 | 8 | 360.0 | 5 | ◆ |
| 24.4 | 4 | 146.7 | 1 | ● |
| 22.8 | 4 | 140.8 | 1 | ● |
| 19.2 | 6 | 167.6 | 2 | + |
| 17.8 | 6 | 167.6 | 2 | + |
| 16.4 | 8 | 275.8 | 3 | ◆ |
| 17.3 | 8 | 275.8 | 3 | ◆ |
| 15.2 | 8 | 275.8 | 3 | ◆ |
| 10.4 | 8 | 472.0 | 4 | ◆ |
| 10.4 | 8 | 460.0 | 4 | ◆ |
| 14.7 | 8 | 440.0 | 4 | ◆ |
| 32.4 | 4 | 78.7 | 1 | ● |
| 30.4 | 4 | 75.7 | 1 | ● |
| 33.9 | 4 | 71.1 | 1 | ● |



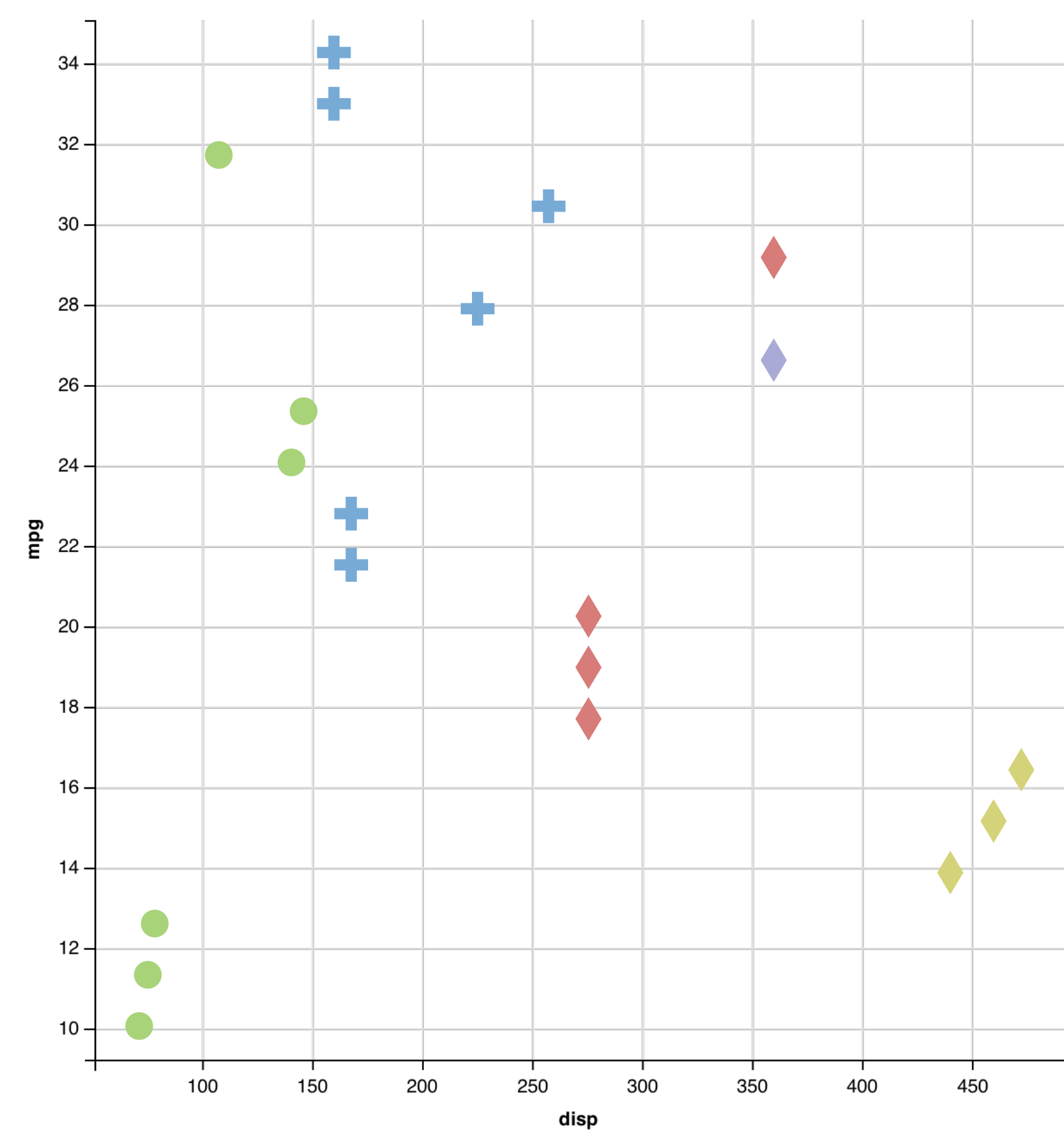
data

geom

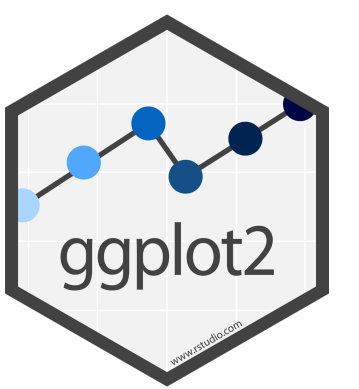


mappings

| y | shape | x | fill |
|------|-------|-------|------|
| mpg | cyl | disp | hp |
| 21.0 | 6 | 160.0 | 2 |
| 21.0 | 6 | 160.0 | 2 |
| 22.8 | 4 | 108.0 | 1 |
| 21.4 | 6 | 258.0 | 2 |
| 18.7 | 8 | 360.0 | 3 |
| 18.1 | 6 | 225.0 | 2 |
| 14.3 | 8 | 360.0 | 5 |
| 24.4 | 4 | 146.7 | 1 |
| 22.8 | 4 | 140.8 | 1 |
| 19.2 | 6 | 167.6 | 2 |
| 17.8 | 6 | 167.6 | 2 |
| 16.4 | 8 | 275.8 | 3 |
| 17.3 | 8 | 275.8 | 3 |
| 15.2 | 8 | 275.8 | 3 |
| 10.4 | 8 | 472.0 | 4 |
| 10.4 | 8 | 460.0 | 4 |
| 14.7 | 8 | 440.0 | 4 |
| 32.4 | 4 | 78.7 | 1 |
| 30.4 | 4 | 75.7 | 1 |
| 33.9 | 4 | 71.1 | 1 |

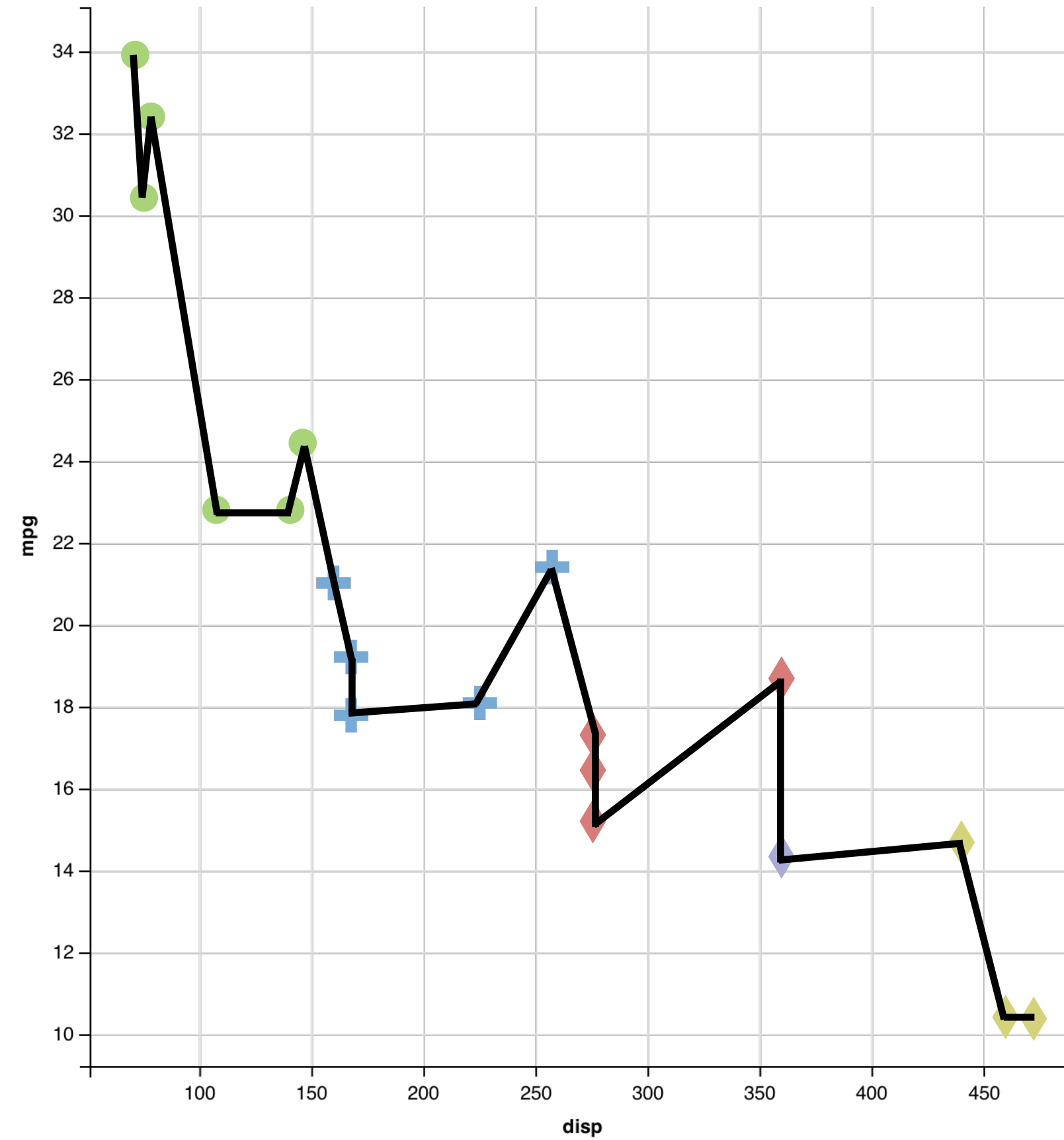


data geom



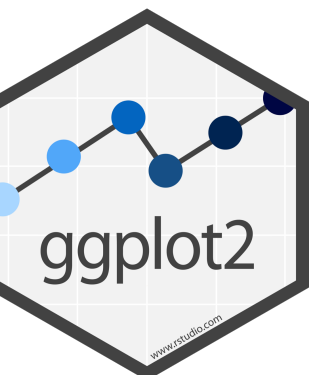
mappings

| ↑↑ y | ↑↑ shape | ↑↑ x | ↑↑ fill | |
|------|----------|-------|---------|---|
| mpg | cyl | disp | hp | |
| 21.0 | 6 | 160.0 | 2 | — |
| 21.0 | 6 | 160.0 | 2 | — |
| 22.8 | 4 | 108.0 | 1 | — |
| 21.4 | 6 | 258.0 | 2 | — |
| 18.7 | 8 | 360.0 | 3 | — |
| 18.1 | 6 | 225.0 | 2 | — |
| 14.3 | 8 | 360.0 | 5 | — |
| 24.4 | 4 | 146.7 | 1 | — |
| 22.8 | 4 | 140.8 | 1 | — |
| 19.2 | 6 | 167.6 | 2 | — |
| 17.8 | 6 | 167.6 | 2 | — |
| 16.4 | 8 | 275.8 | 3 | — |
| 17.3 | 8 | 275.8 | 3 | — |
| 15.2 | 8 | 275.8 | 3 | — |
| 10.4 | 8 | 472.0 | 4 | — |
| 10.4 | 8 | 460.0 | 4 | — |
| 14.7 | 8 | 440.0 | 4 | — |
| 32.4 | 4 | 78.7 | 1 | — |
| 30.4 | 4 | 75.7 | 1 | — |
| 33.9 | 4 | 71.1 | 1 | — |


























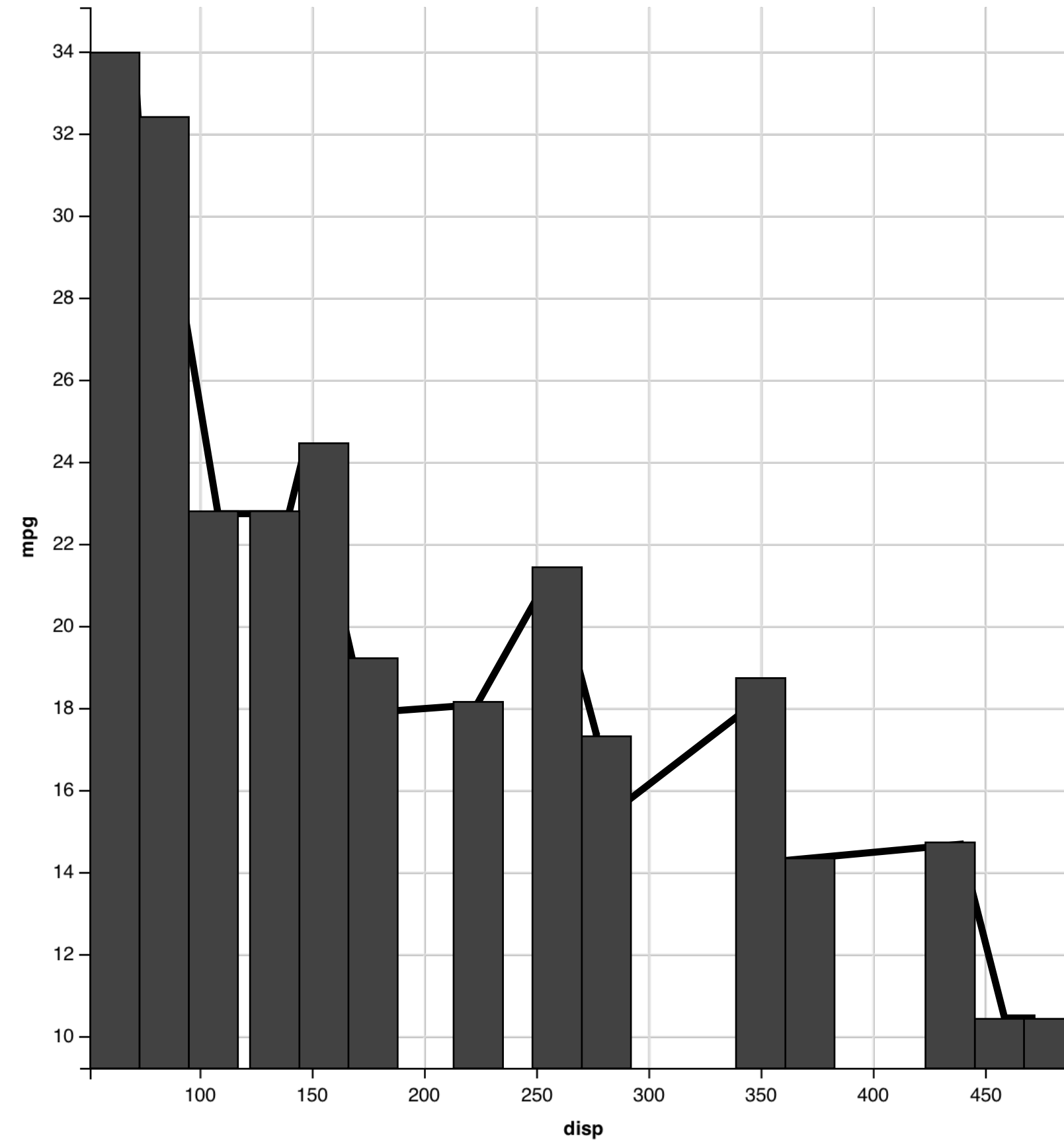
data

geom
points
lines



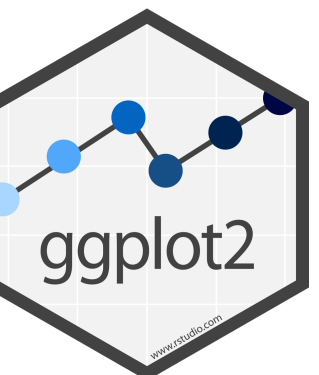
mappings

|  mpg | cyl |  disp | hp |  |
|---|-----|--|----|---|
| 21.0 | 6 | 160.0 | 2 |  |
| 21.0 | 6 | 160.0 | 2 |  |
| 22.8 | 4 | 108.0 | 1 |  |
| 21.4 | 6 | 258.0 | 2 |  |
| 18.7 | 8 | 360.0 | 3 |  |
| 18.1 | 6 | 225.0 | 2 |  |
| 14.3 | 8 | 360.0 | 5 |  |
| 24.4 | 4 | 146.7 | 1 |  |
| 22.8 | 4 | 140.8 | 1 |  |
| 19.2 | 6 | 167.6 | 2 |  |
| 17.8 | 6 | 167.6 | 2 |  |
| 16.4 | 8 | 275.8 | 3 |  |
| 17.3 | 8 | 275.8 | 3 |  |
| 15.2 | 8 | 275.8 | 3 |  |
| 10.4 | 8 | 472.0 | 4 |  |
| 10.4 | 8 | 460.0 | 4 |  |
| 14.7 | 8 | 440.0 | 4 |  |
| 32.4 | 4 | 78.7 | 1 |  |
| 30.4 | 4 | 75.7 | 1 |  |
| 33.9 | 4 | 71.1 | 1 |  |



data

geom
points
lines
bars

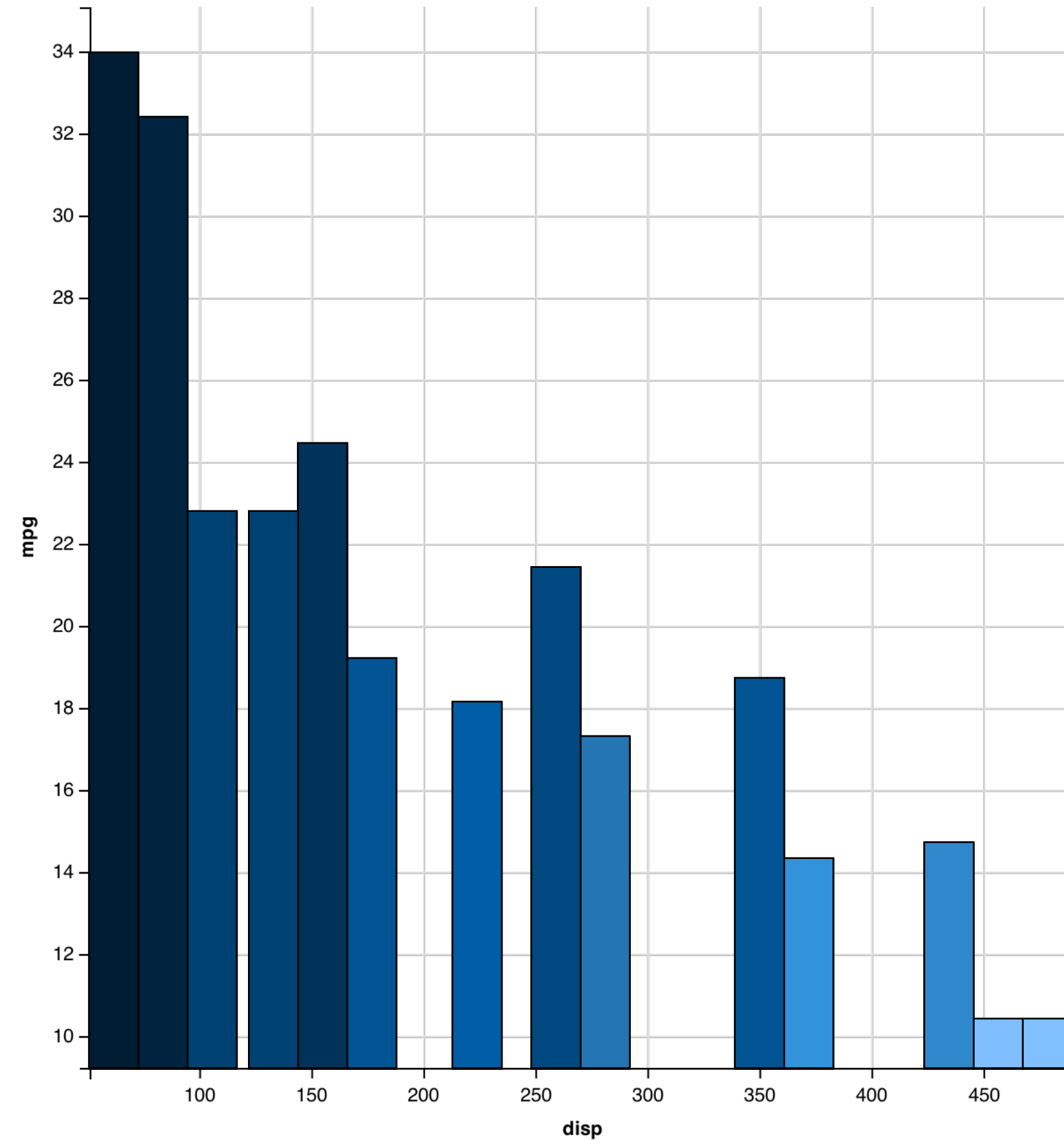



mappings

↑ y

↑↑ fill

| mpg | cyl | disp | hp |
|------|-----|-------|----|
| 21.0 | 6 | 160.0 | 2 |
| 21.0 | 6 | 160.0 | 2 |
| 22.8 | 4 | 108.0 | 1 |
| 21.4 | 6 | 258.0 | 2 |
| 18.7 | 8 | 360.0 | 3 |
| 18.1 | 6 | 225.0 | 2 |
| 14.3 | 8 | 360.0 | 5 |
| 24.4 | 4 | 146.7 | 1 |
| 22.8 | 4 | 140.8 | 1 |
| 19.2 | 6 | 167.6 | 2 |
| 17.8 | 6 | 167.6 | 2 |
| 16.4 | 8 | 275.8 | 3 |
| 17.3 | 8 | 275.8 | 3 |
| 15.2 | 8 | 275.8 | 3 |
| 10.4 | 8 | 472.0 | 4 |
| 10.4 | 8 | 460.0 | 4 |
| 14.7 | 8 | 440.0 | 4 |
| 32.4 | 4 | 78.7 | 1 |
| 30.4 | 4 | 75.7 | 1 |
| 33.9 | 4 | 71.1 | 1 |



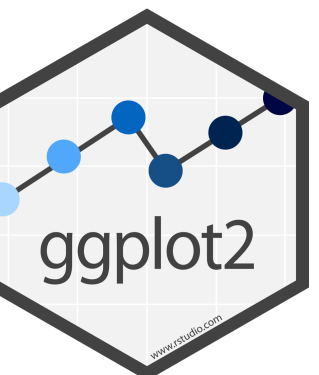
data

geom

points

lines

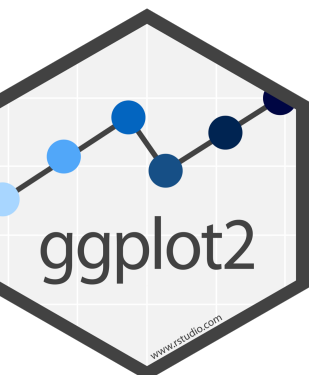
bars



To make a graph

[template]

```
ggplot(data = <DATA>) +  
  <GEOM_FUNCTION>(mapping = aes(<MAPPINGS>))
```



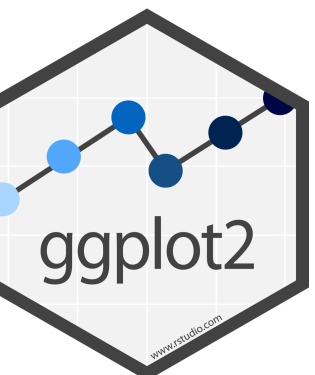
To make a graph

1. Pick a data set

| mpg | cyl | disp | hp |
|------|-----|-------|----|
| 21.0 | 6 | 160.0 | 2 |
| 21.0 | 6 | 160.0 | 2 |
| 22.8 | 4 | 108.0 | 1 |
| 21.4 | 6 | 258.0 | 2 |
| 18.7 | 8 | 360.0 | 3 |
| 18.1 | 6 | 225.0 | 2 |
| 14.3 | 8 | 360.0 | 5 |
| 24.4 | 4 | 146.7 | 1 |
| 22.8 | 4 | 140.8 | 1 |
| 19.2 | 6 | 167.6 | 2 |
| 17.8 | 6 | 167.6 | 2 |
| 16.4 | 8 | 275.8 | 3 |
| 17.3 | 8 | 275.8 | 3 |
| 15.2 | 8 | 275.8 | 3 |
| 10.4 | 8 | 472.0 | 4 |
| 10.4 | 8 | 460.0 | 4 |
| 14.7 | 8 | 440.0 | 4 |
| 32.4 | 4 | 78.7 | 1 |
| 30.4 | 4 | 75.7 | 1 |
| 33.9 | 4 | 71.1 | 1 |

data

```
ggplot(data = <DATA>) +  
  <GEOM_FUNCTION>(mapping = aes(<MAPPINGS>))
```



To make a graph

| mpg | cyl | disp | hp |
|------|-----|-------|----|
| 21.0 | 6 | 160.0 | 2 |
| 21.0 | 6 | 160.0 | 2 |
| 22.8 | 4 | 108.0 | 1 |
| 21.4 | 6 | 258.0 | 2 |
| 18.7 | 8 | 360.0 | 3 |
| 18.1 | 6 | 225.0 | 2 |
| 14.3 | 8 | 360.0 | 5 |
| 24.4 | 4 | 146.7 | 1 |
| 22.8 | 4 | 140.8 | 1 |
| 19.2 | 6 | 167.6 | 2 |
| 17.8 | 6 | 167.6 | 2 |
| 16.4 | 8 | 275.8 | 3 |
| 17.3 | 8 | 275.8 | 3 |
| 15.2 | 8 | 275.8 | 3 |
| 10.4 | 8 | 472.0 | 4 |
| 10.4 | 8 | 460.0 | 4 |
| 14.7 | 8 | 440.0 | 4 |
| 32.4 | 4 | 78.7 | 1 |
| 30.4 | 4 | 75.7 | 1 |
| 33.9 | 4 | 71.1 | 1 |

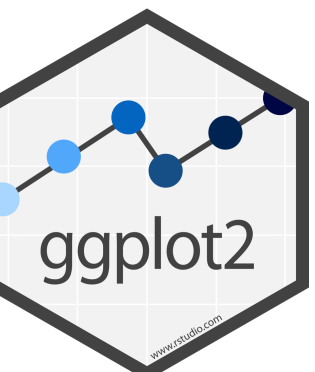
data

geom

1. Pick a data set

```
ggplot(data = <DATA>) +  
  <GEOM_FUNCTION>(mapping = aes(<MAPPINGS>))
```

2. Choose a geom to display cases



To make a graph

mappings

| mpg | cyl | disp | hp | fill |
|------|-----|-------|----|------|
| 21.0 | 6 | 160.0 | 2 | ● |
| 21.0 | 6 | 160.0 | 2 | ● |
| 22.8 | 4 | 108.0 | 1 | ● |
| 21.4 | 6 | 258.0 | 2 | ● |
| 18.7 | 8 | 360.0 | 3 | ● |
| 18.1 | 6 | 225.0 | 2 | ● |
| 14.3 | 8 | 360.0 | 5 | ● |
| 24.4 | 4 | 146.7 | 1 | ● |
| 22.8 | 4 | 140.8 | 1 | ● |
| 19.2 | 6 | 167.6 | 2 | ● |
| 17.8 | 6 | 167.6 | 2 | ● |
| 16.4 | 8 | 275.8 | 3 | ● |
| 17.3 | 8 | 275.8 | 3 | ● |
| 15.2 | 8 | 275.8 | 3 | ● |
| 10.4 | 8 | 472.0 | 4 | ● |
| 10.4 | 8 | 460.0 | 4 | ● |
| 14.7 | 8 | 440.0 | 4 | ● |
| 32.4 | 4 | 78.7 | 1 | ● |
| 30.4 | 4 | 75.7 | 1 | ● |
| 33.9 | 4 | 71.1 | 1 | ● |

data

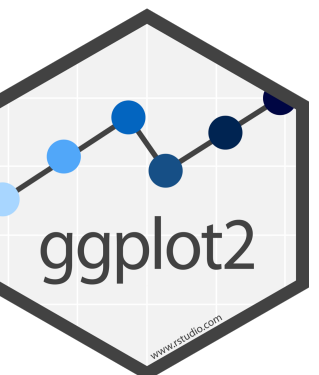
geom

1. Pick a data set

```
ggplot(data = <DATA>) +  
  <GEOM_FUNCTION>(mapping = aes(<MAPPINGS>))
```

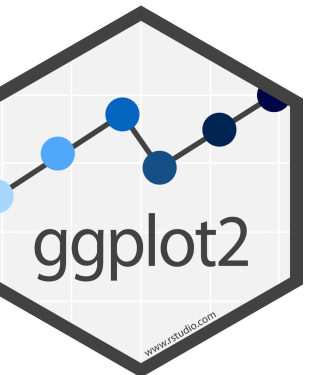
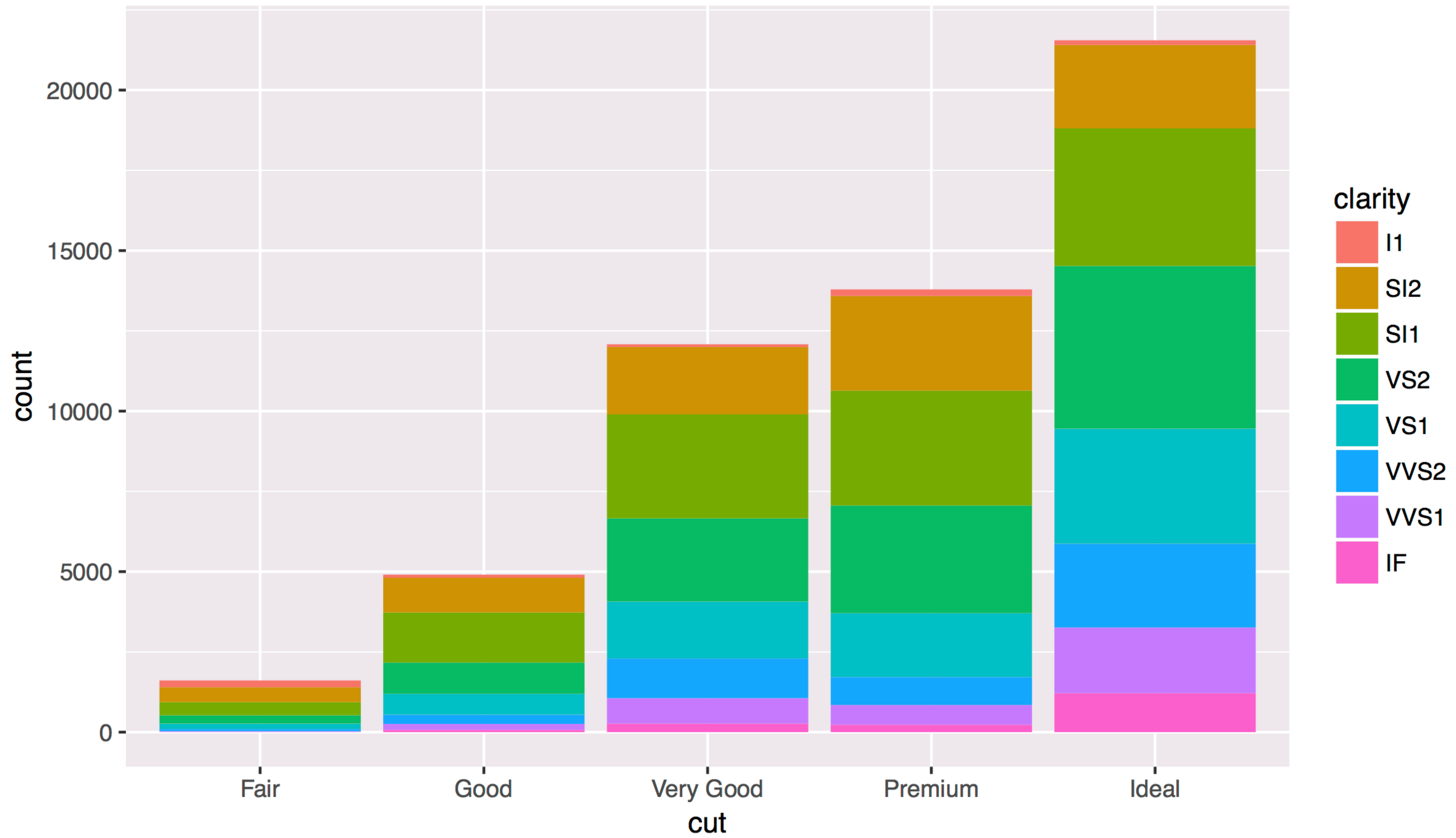
2. Choose a geom
to display cases

3. Map aesthetic
properties to
variables



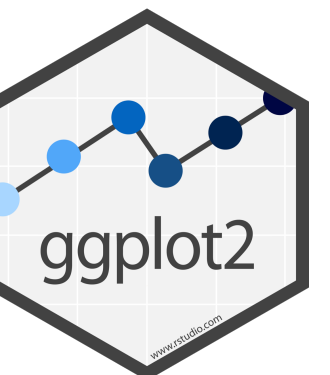
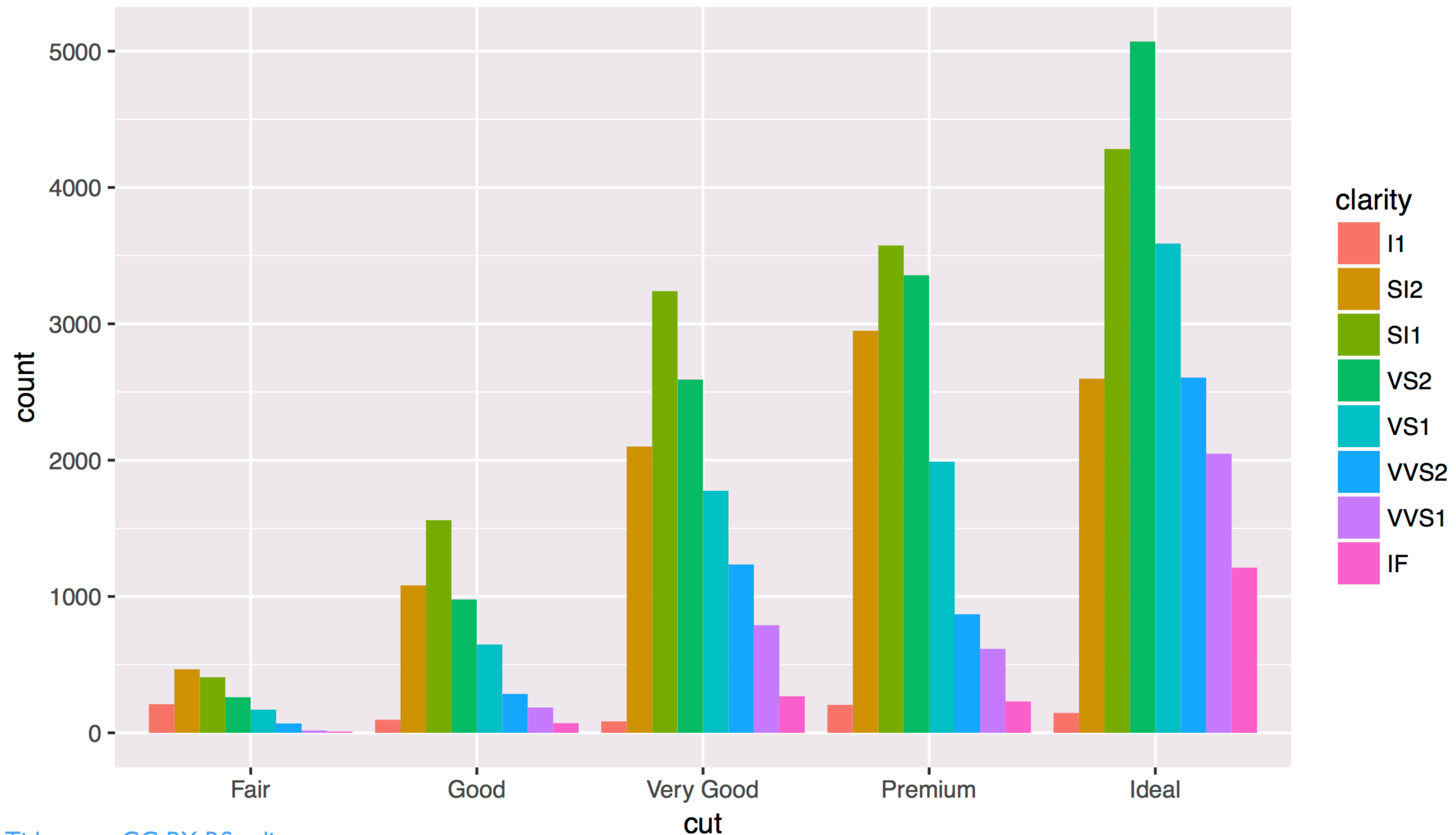
What else?





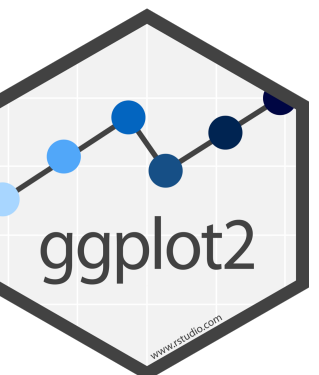
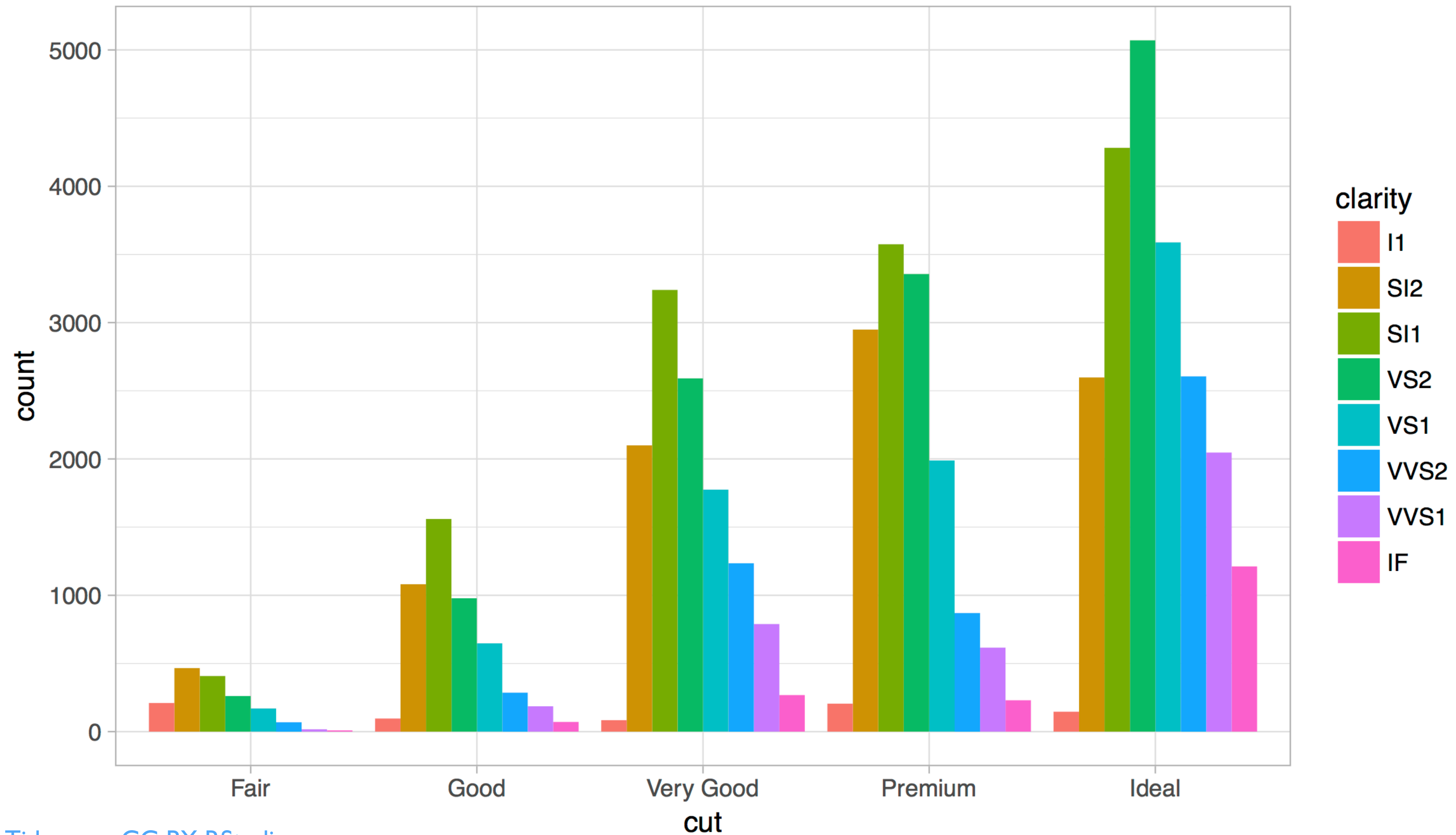
Position Adjustments

How overlapping objects are arranged



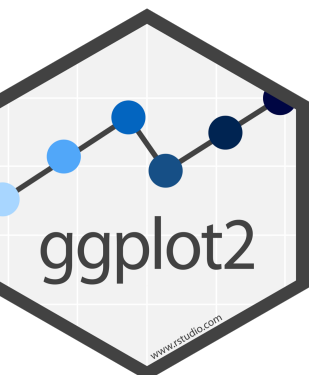
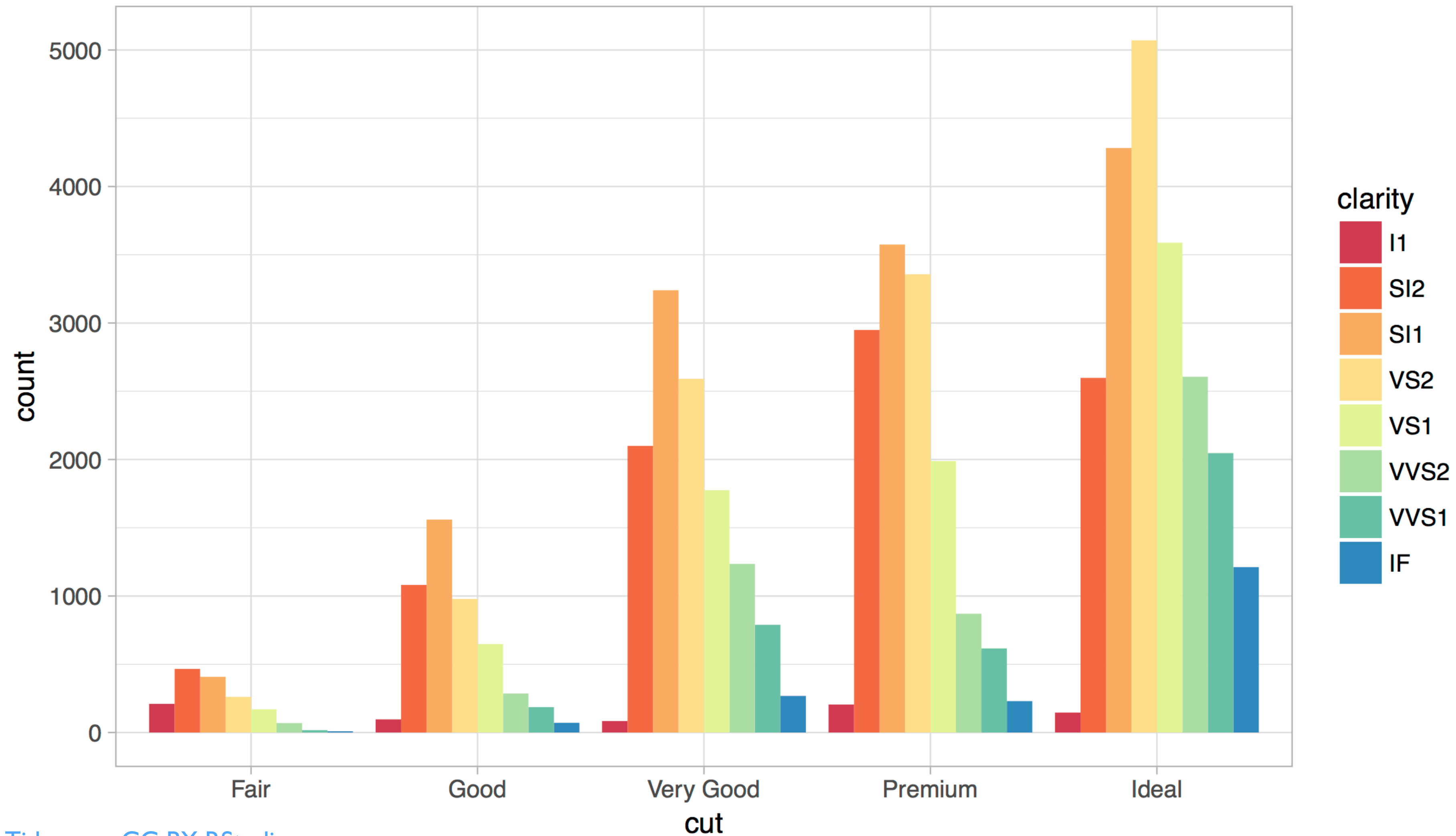
Themes

Visual appearance of non-data elements



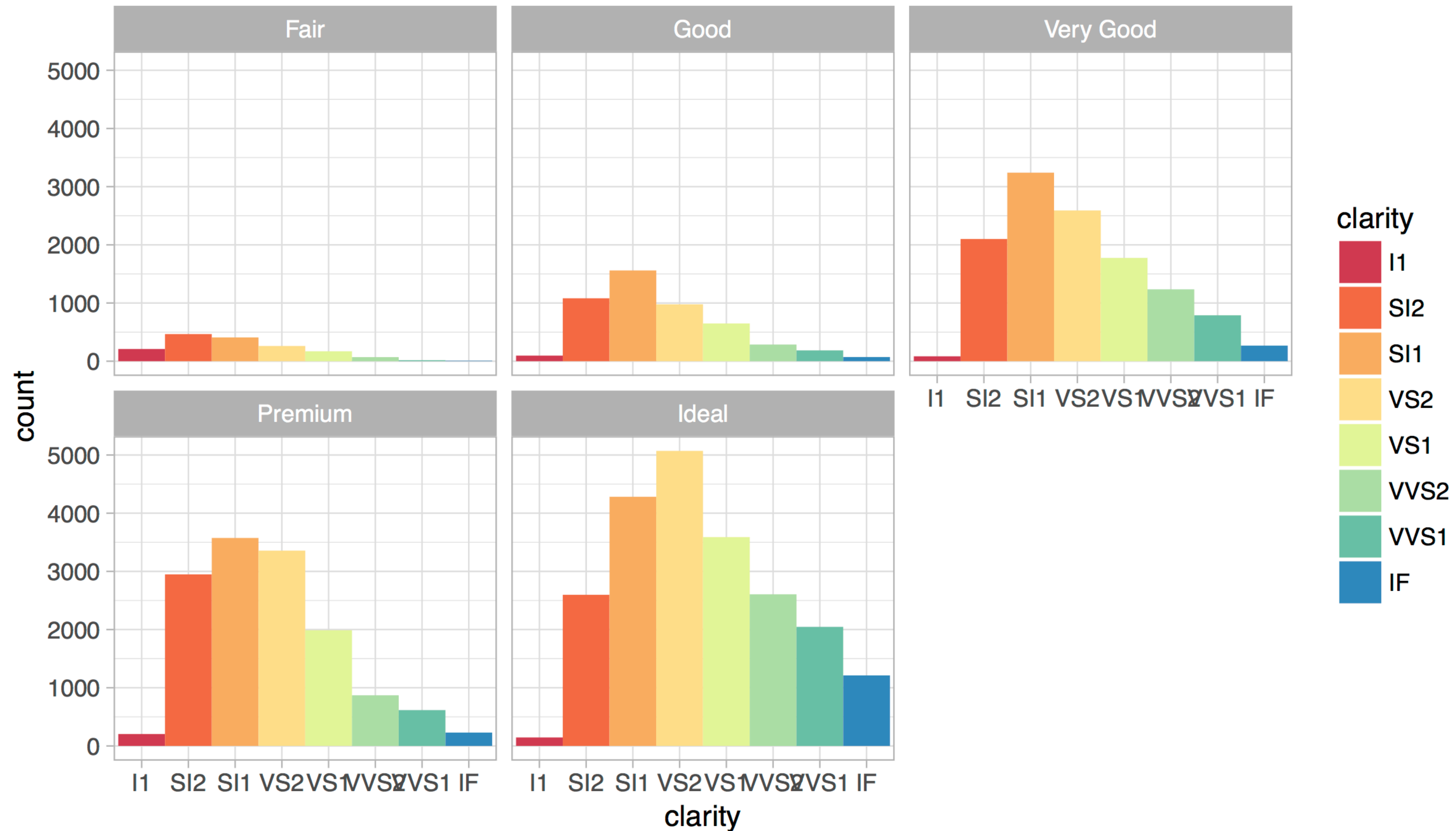
Scales

Customize color scales, other mappings

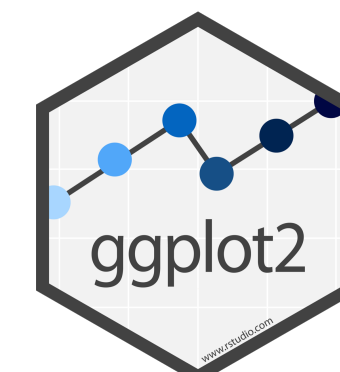
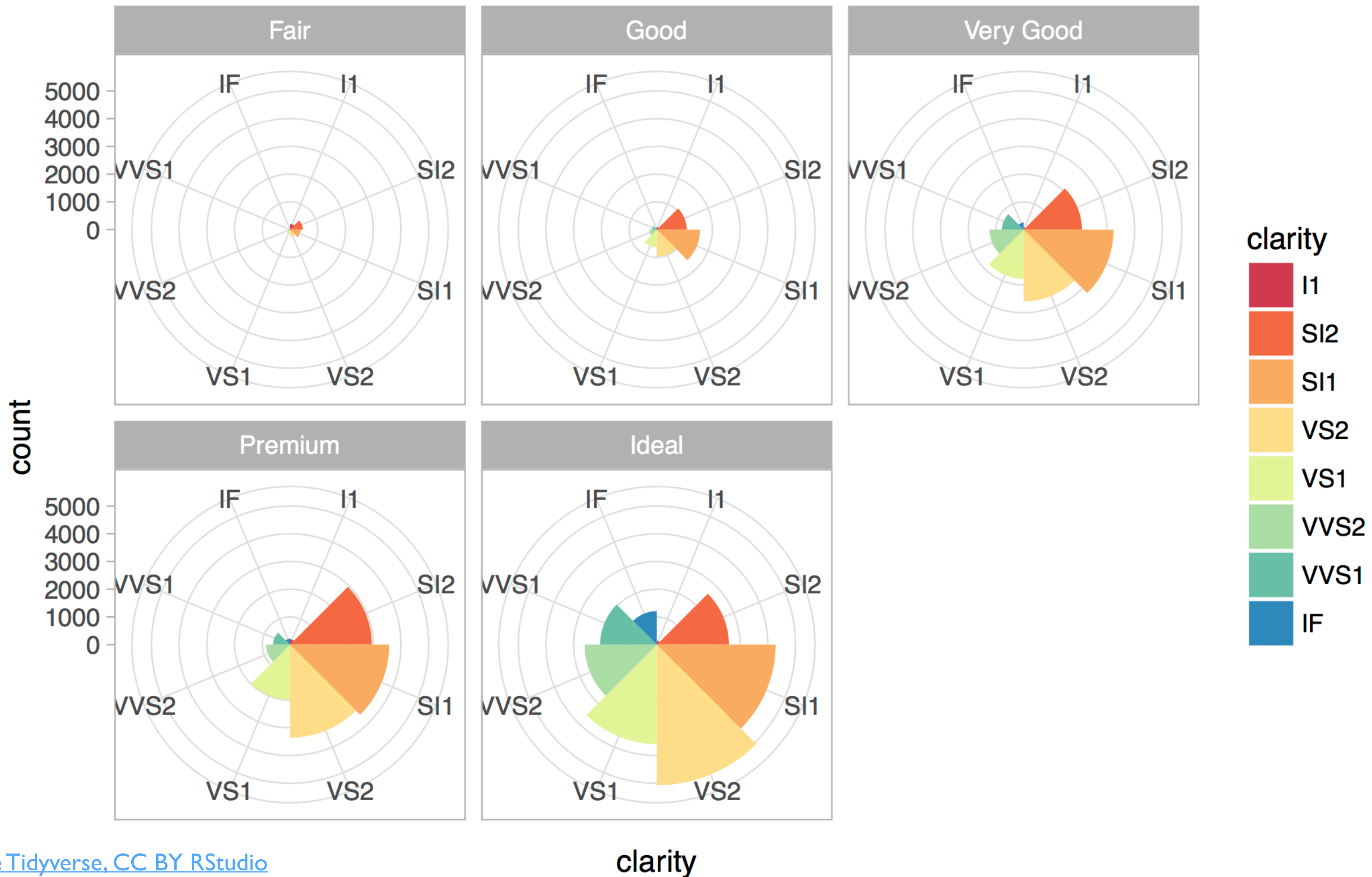


Facets

Subplots that display subsets of the data.



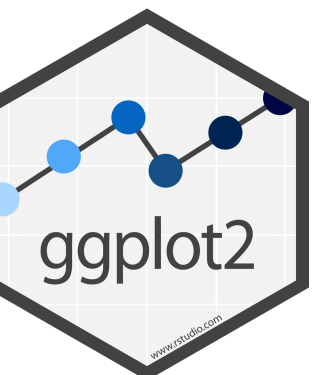
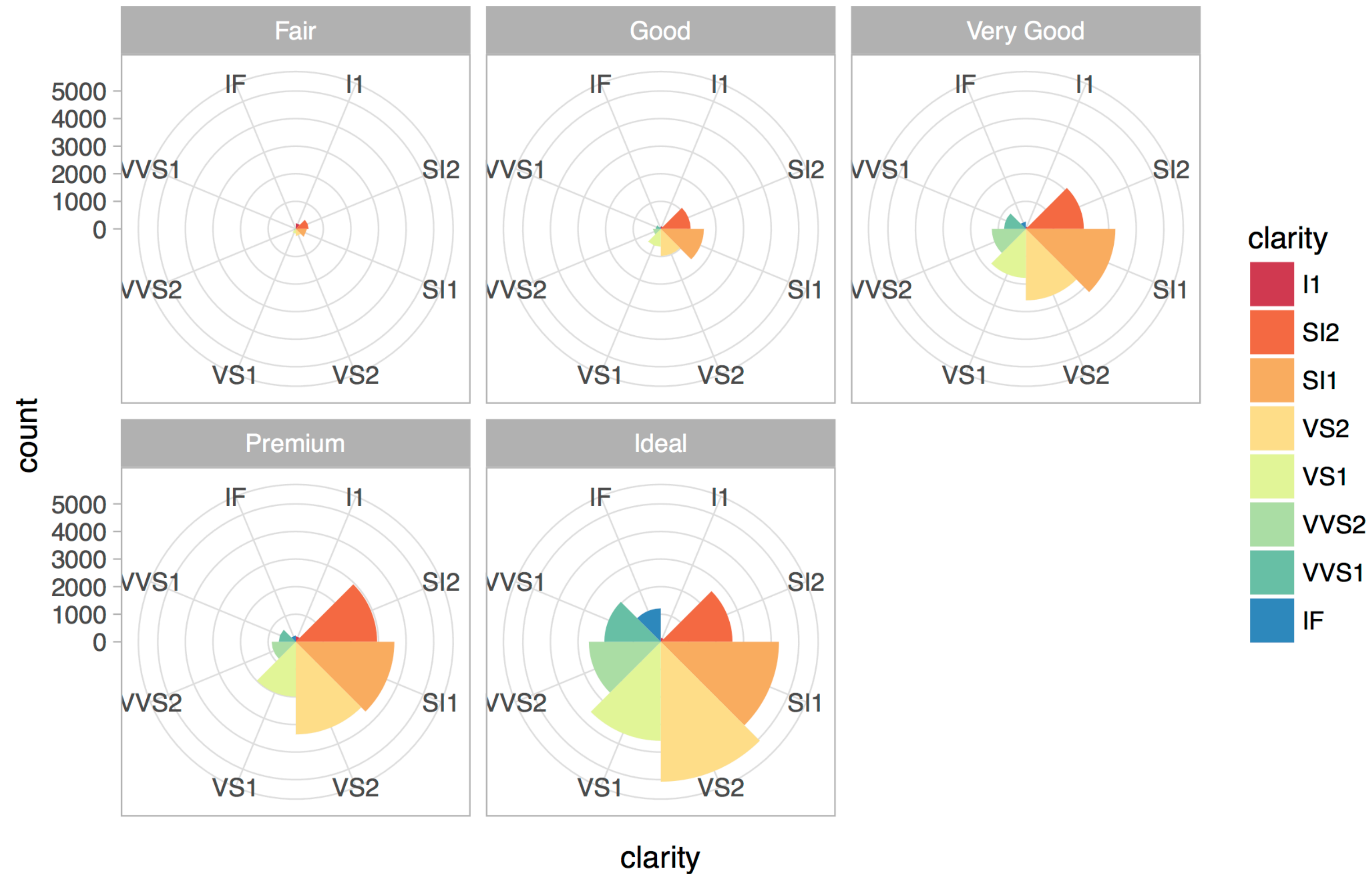
Coordinate systems



Titles and captions

Diamonds data

The data set is skewed towards ideal cut diamonds



A ggplot2 template

Make any plot by filling in the parameters of this template

Complete the template below to build a graph.

```
ggplot (data = <DATA>) +  
  <GEOM_FUNCTION> (mapping = aes(<MAPPINGS>),  
  stat = <STAT> , position = <POSITION>) +  
  <COORDINATE_FUNCTION> +  
  <FACET_FUNCTION> +  
  <SCALE_FUNCTION> +  
  <THEME_FUNCTION>
```

required

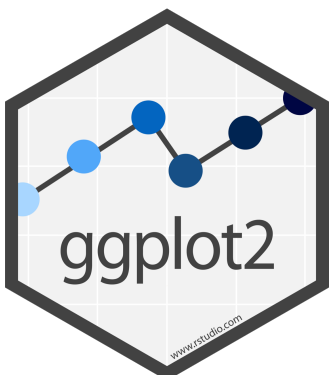
Not required, sensible defaults supplied

Data Visualization with ggplot2 :: CHEAT SHEET

The cheat sheet is titled "Data Visualization with ggplot2 :: CHEAT SHEET" and features the ggplot2 logo. It is organized into several sections:

- Basics:** Explains the grammar of graphics and provides a template for creating a plot: `ggplot(data = <DATA>) + <GEOM_FUNCTION>(mapping = aes(<MAPPINGS>), stat = <STAT>, position = <POSITION>) + <COORDINATE_FUNCTION> + <FACET_FUNCTION> + <SCALE_FUNCTION> + <THEME_FUNCTION>`. It also includes a diagram showing the relationship between data, geom, coordinate system, and plot.
- Geoms:** Lists graphical primitives such as `geom_blank`, `geom_curve`, `geom_path`, `geom_polygon`, `geom_rect`, and `geom_ribbon`.
- TWO VARIABLES:** Lists functions for continuous x and y, such as `geom_abline`, `geom_density`, `geom_histogram`, and `geom_smooth`.
- ONE VARIABLE:** Lists functions for continuous data, such as `geom_area`, `geom_bar`, `geom_boxplot`, and `geom_histogram`.
- Discrete x, discrete y:** Lists functions like `geom_crossbar`, `geom_errorbar`, and `geom_pointrange`.
- THREE VARIABLES:** Lists functions like `geom_count`, `geom_raster`, and `geom_tile`.

At the bottom, it includes the R logo and the text "RStudio is a trademark of RStudio, Inc. - CC BY SA RStudio - info@rstudio.com - 844-448-2212 - rstudio.com - Learn more at <http://ggplot2.tidyverse.org> - ggplot2 2.1.0 - Updated: 2016-11".



ggplot2.tidyverse.org

The screenshot shows a web browser window with the URL `ggplot2.tidyverse.org`. The page features a navigation bar with links for Reference, Articles, and News. The main content area is titled "Usage" and explains the ggplot2 workflow. It includes a code block showing the installation of the library and the creation of a plot. Below the code is a partial view of a scatter plot with points colored by class. A legend on the right indicates that red points represent "2seater" cars. To the right of the main text, there is a "Links" section with several helpful URLs for downloading the package, viewing source code, reporting bugs, and learning more about data visualization.

Create Elegant Data Visualisati x Garrett

ggplot2 part of the tidyverse

Reference Articles News

Usage

It's hard to succinctly describe how ggplot2 works because it embodies a deep philosophy of visualisation. However, in most cases you start with `ggplot()`, supply a dataset and aesthetic mapping (with `aes()`). You then add on layers (like `geom_point()` or `geom_histogram()`), scales (like `scale_colour_brewer()`), faceting specifications (like `facet_wrap()`) and coordinate systems (like `coord_flip()`).

```
library(ggplot2)

ggplot(mpg, aes(displ, hwy, colour = class)) +
  geom_point()
```

class
● 2seater

Links

Download from CRAN at <https://cran.r-project.org/package=ggplot2>

Browse source code at <https://github.com/tidyverse/ggplot2>

Report a bug at <https://github.com/tidyverse/ggplot2/issues>

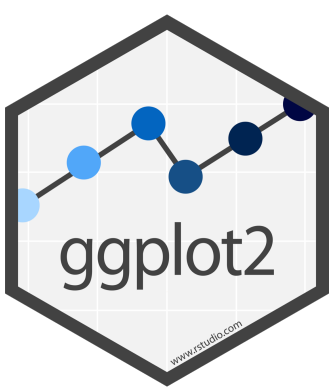
Learn more at <http://r4ds.had.co.nz/data-visualisation.html>

License

GPL-2 | file LICENSE

Developers

Hadley Wickham
Author, maintainer



Visualize Data with

