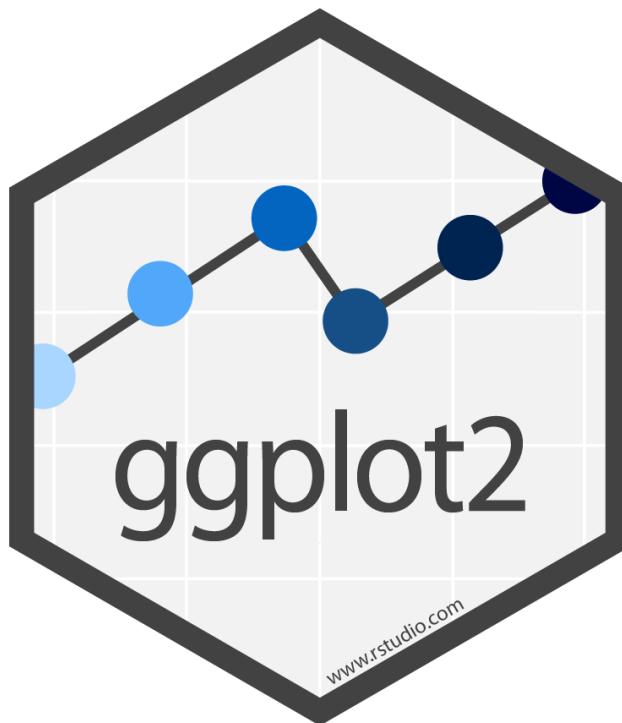


Visualising Data with



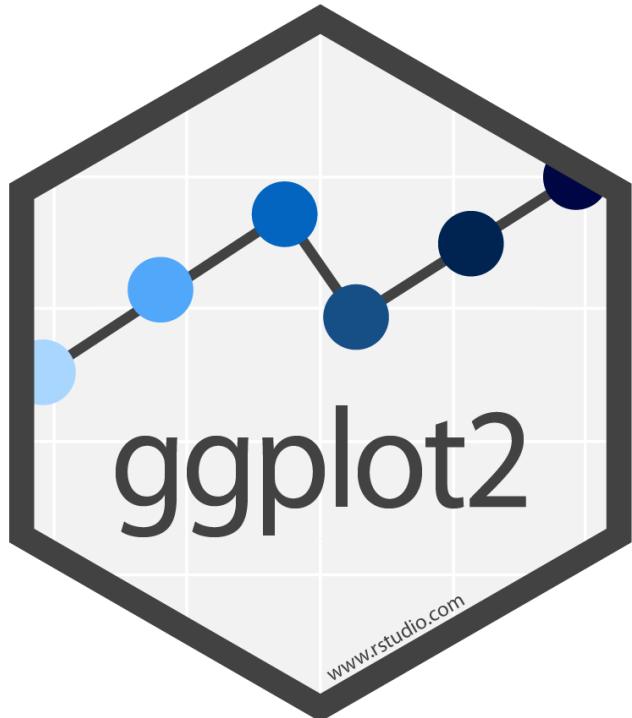
[Adapted from “Explore the Tidyverse” CC by Hadley Wickham](#)

Why Visualise?

“The simple graph has brought more information to the data analyst’s mind than any other device.”

- John Tukey

ggplot2



One of the earliest tidyverse packages.

Complex plots, by layering simple components.

Your Turn

Open 01-Visualise.Rmd



I'm working on it



I'm stuck!

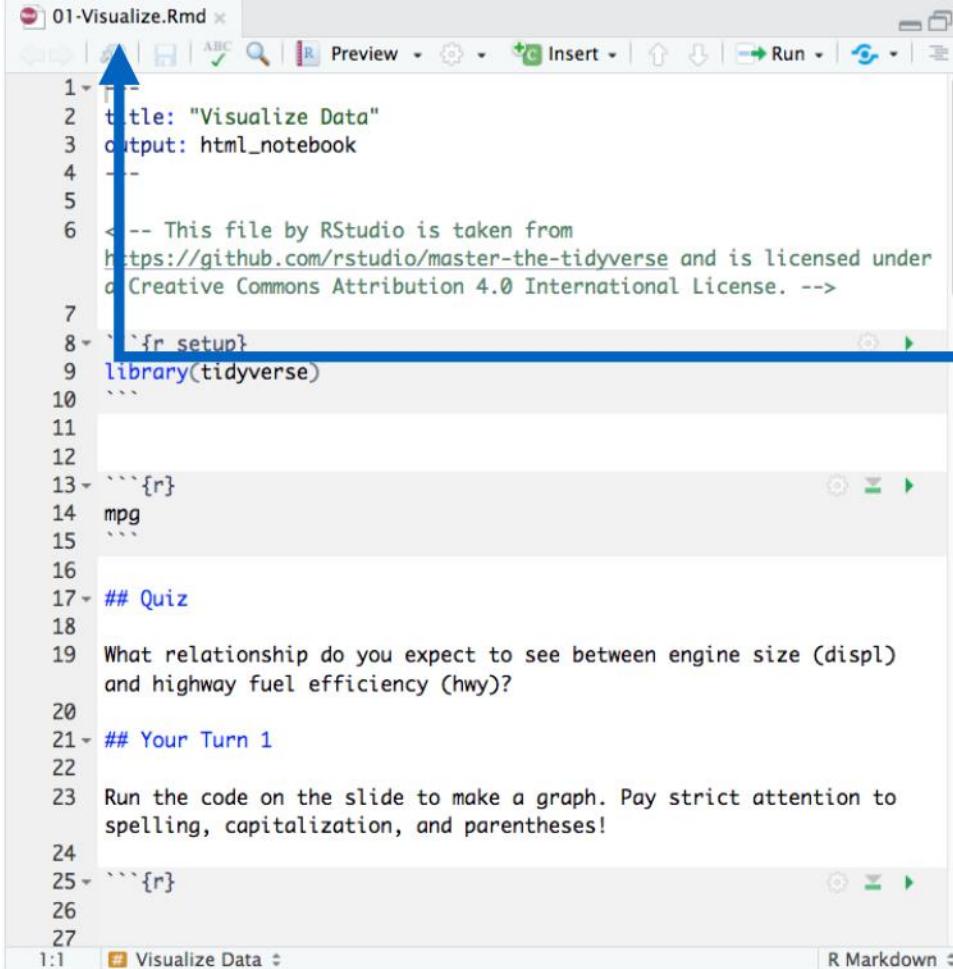


I'm done!

The screenshot shows the RStudio interface with the 'Files' tab selected. The current directory is 'Home > Desktop > data-science-in-tidyverse'. The file list includes:

	Name	Size	Modified
	..		
	.gitignore	47 B	Mar 6, 2018, 9:48 AM
	00-Getting-started.Rmd	1.3 KB	Mar 6, 2018, 10:28 AM
	01-Visualize.Rmd	1.6 KB	Mar 6, 2018, 10:28 AM
	02-Transform.Rmd	3.7 KB	Mar 6, 2018, 10:28 AM
	03-Tidy.Rmd	2.6 KB	Mar 6, 2018, 10:28 AM
	04-Case-Study.Rmd	5 KB	Mar 6, 2018, 10:28 AM
	05-Data-Types.Rmd	3.4 KB	Mar 6, 2018, 10:28 AM
	06-Iterate.Rmd	2.9 KB	Mar 6, 2018, 10:28 AM
	07-Model.Rmd	2.5 KB	Mar 6, 2018, 10:28 AM
	08-Organize.Rmd	2 KB	Mar 6, 2018, 10:28 AM
	99-Setup.md	1.4 KB	Mar 6, 2018, 9:48 AM
	cheatsheets		
	data-science-in-the-tidyverse.Rproj	205 B	Mar 6, 2018, 9:49 AM
	email-to-participants.md	2.7 KB	Mar 6, 2018, 9:48 AM
	README.md	2.3 KB	Mar 6, 2018, 9:59 AM

If you get lost or need to restart

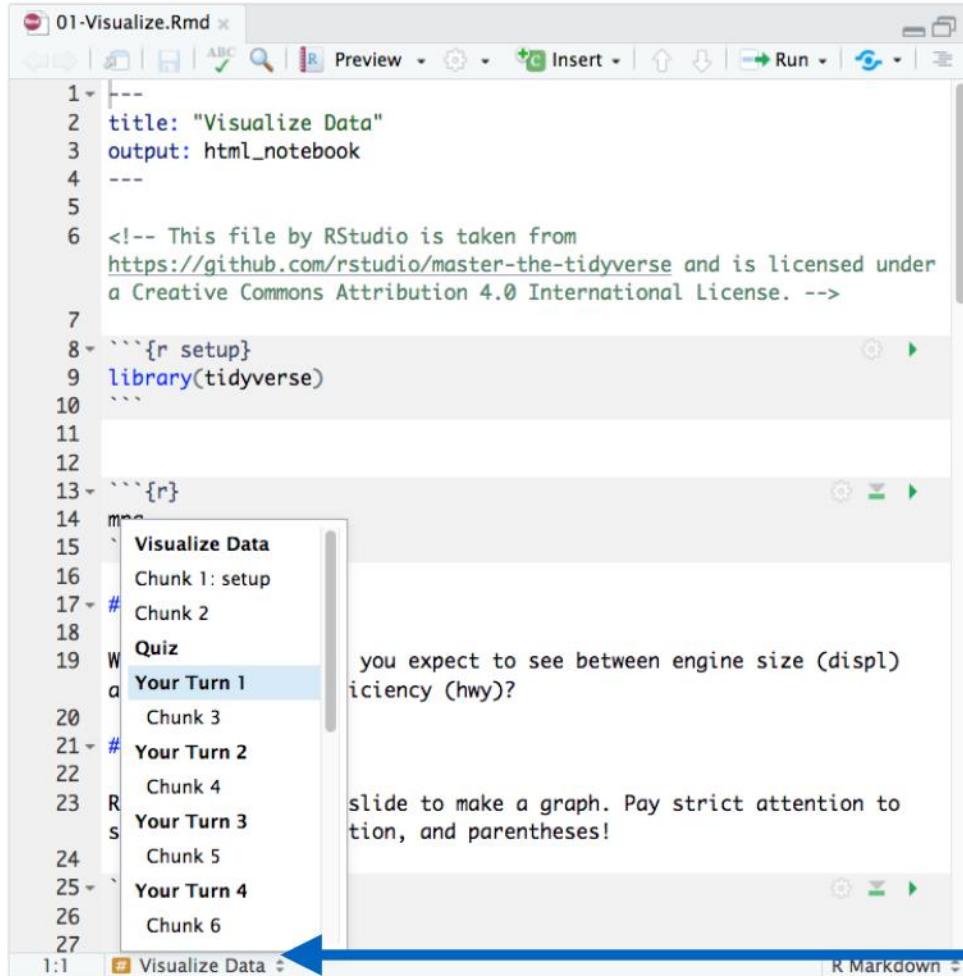


```
01-Visualize.Rmd ×
ABC 🔎 Preview ⚙️ Insert ⚡ Run ⚡

1 - title: "Visualize Data"
2 - output: html_notebook
3 -
4 -
5 -
6 <-- This file by RStudio is taken from
https://github.com/rstudio/master-the-tidyverse and is licensed under
the Creative Commons Attribution 4.0 International License. -->
7
8 `r setup!
9 library(tidyverse)
10 ...
11
12
13 ``{r}
14 mpg
15 ...
16
17 ## Quiz
18
19 What relationship do you expect to see between engine size (displ)
and highway fuel efficiency (hwy)?
20
21 ## Your Turn 1
22
23 Run the code on the slide to make a graph. Pay strict attention to
spelling, capitalization, and parentheses!
24
25 ``{r}
26
27
1:1 # Visualize Data ▾ R Markdown ▾
```

Check you are in the right file

If you get lost or need to restart



```
1 ---  
2 title: "Visualize Data"  
3 output: html_notebook  
4 ---  
5  
6 <!-- This file by RStudio is taken from  
https://github.com/rstudio/master-the-tidyverse and is licensed under  
a Creative Commons Attribution 4.0 International License. -->  
7  
8 ```{r setup}  
9 library(tidyverse)  
10```  
11  
12  
13 ```{r}  
14 mma  
15 ` Visualize Data  
16 Chunk 1: setup  
17 # Chunk 2  
18 Quiz  
19 W  
a Your Turn 1  
20 Chunk 3  
21 # Your Turn 2  
22 Chunk 4  
23 R  
s Your Turn 3  
24 Chunk 5  
25 # Your Turn 4  
26 Chunk 6  
27  
1:1 # Visualize Data
```

Use the section browser to quickly navigate to the right *Your Turn*

If you get lost or need to restart

```
01-Visualize.Rmd x
ABC Preview Insert Run
15 ``
16
17 ## Quiz
18
19 What relationship do you expect to see between engine size (displ)
and highway fuel efficiency (hwy)?
20
21 ## Your Turn 1
22
23 Run the code on the slide to make a graph. Pay strict attention to
spelling, capitalization, and parentheses!
24
25 ``{r}
26
27 ``
28
29
30 ## Your Turn 2
31
32 Add `color`, `size`, `alpha`, and `shape` aesthetics to your graph.
Experiment.
33
34 ``{r}
35 ggplot(data = mpg) +
36   geom_point(mapping = aes(x = displ, y = hwy))
37 ``
38
39 ## Your Turn 3
40
41 Replace this scatterplot with one that draws boxplots. Use the
cheatsheet. Try your best guess.
21:1 # Your Turn 1 R Markdown
```

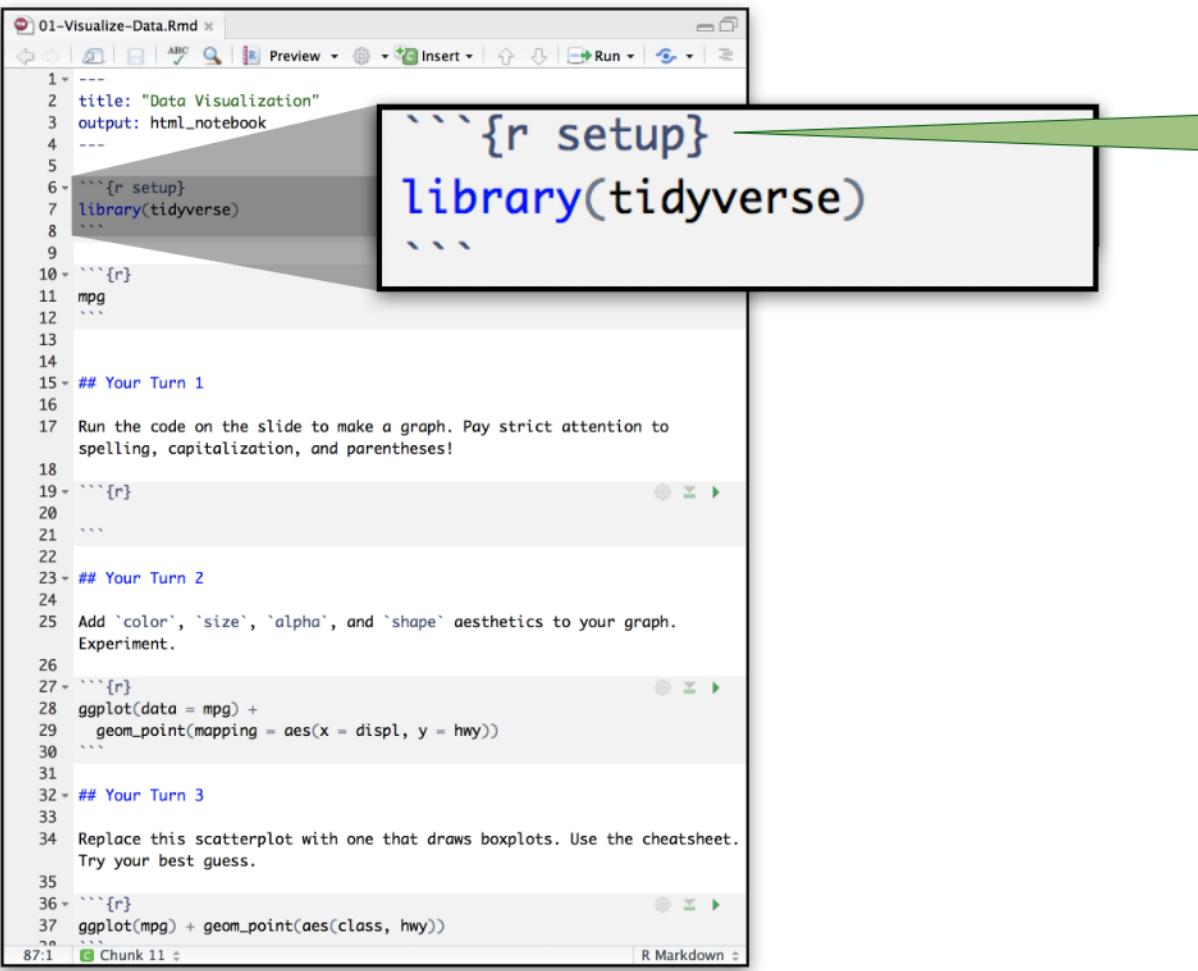
Click to run all
chunks before this
one.



You should be ready
to go.

Setup

The setup chunk is always run once before anything else



A screenshot of the RStudio interface showing an R Markdown document titled "01-Visualize-Data.Rmd". The code editor displays the following R code:

```
1 ---  
2 title: "Data Visualization"  
3 output: html_notebook  
4 ---  
5  
6 ```{r setup}  
7 library(tidyverse)  
8 ```  
9  
10 ````{r}  
11 mpg  
12  
13  
14  
15 ## Your Turn 1  
16  
17 Run the code on the slide to make a graph. Pay strict attention to  
spelling, capitalization, and parentheses!  
18  
19 ````{r}  
20  
21  
22  
23 ## Your Turn 2  
24  
25 Add `color`, `size`, `alpha`, and `shape` aesthetics to your graph.  
Experiment.  
26  
27 ````{r}  
28 ggplot(data = mpg) +  
29   geom_point(mapping = aes(x = displ, y = hwy))  
30  
31  
32 ## Your Turn 3  
33  
34 Replace this scatterplot with one that draws boxplots. Use the cheatsheet.  
Try your best guess.  
35  
36 ````{r}  
37 ggplot(mpg) + geom_point(aes(class, hwy))
```

A callout bubble with a green border and white text points to the line `library(tidyverse)` with the label "(optional) label for chunk".



mpg

Fuel economy data for 38 models of car.

```
mpg
```

```
?mpg
```



Quiz

Confer with your neighbours.

What relationship do you expect to see between engine size (displ) and highway fuel efficiency (hwy)?

No peeking ahead!



Your Turn 1

Run this code in your notebook to make a graph.

Pay strict attention to spelling, capitalization, and parentheses!

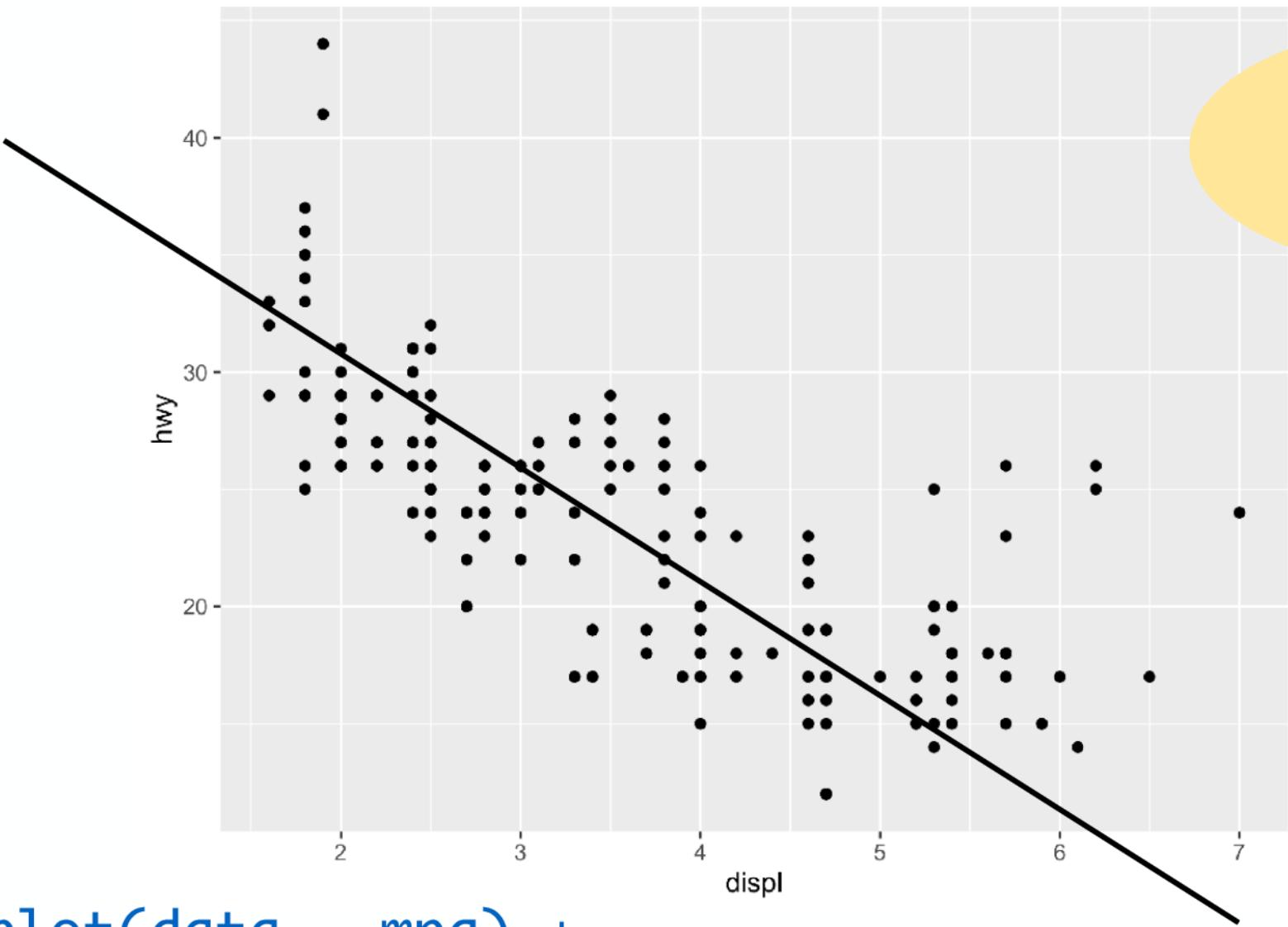
```
ggplot(data = mpg) +  
  geom_point(mapping = aes(x = displ, y = hwy))
```

I'm working on it

I'm stuck!

I'm done!





What do you think
about the line of
best fit?

```
ggplot(data = mpg) +  
  geom_point(mapping = aes(x = displ, y = hwy))
```



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

```
ggplot(data = mpg) +  
  geom_point(mapping = aes(x = displ, y = hwy))
```



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

```
ggplot(data = mpg) +  
  geom_point(mapping = aes(x = displ, y = hwy))
```



```
ggplot(data = mpg) +  
  geom_point(mapping = aes(x = displ, y = hwy))
```

data

+ before new line

type of layer

aes()

x variable

y variable

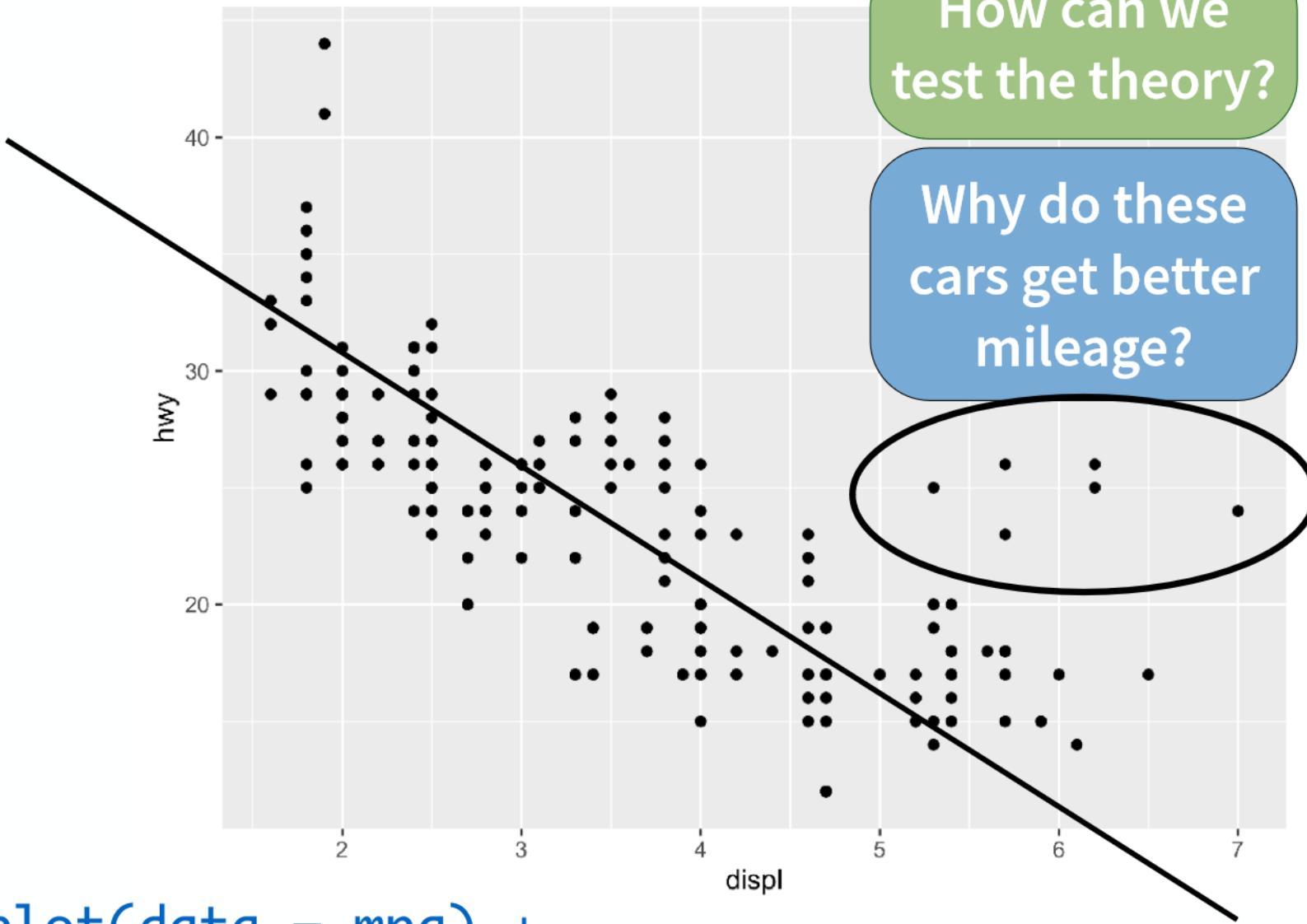


A Template

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



Mappings



```
ggplot(data = mpg) +  
  geom_point(mapping = aes(x = displ, y = hwy))
```

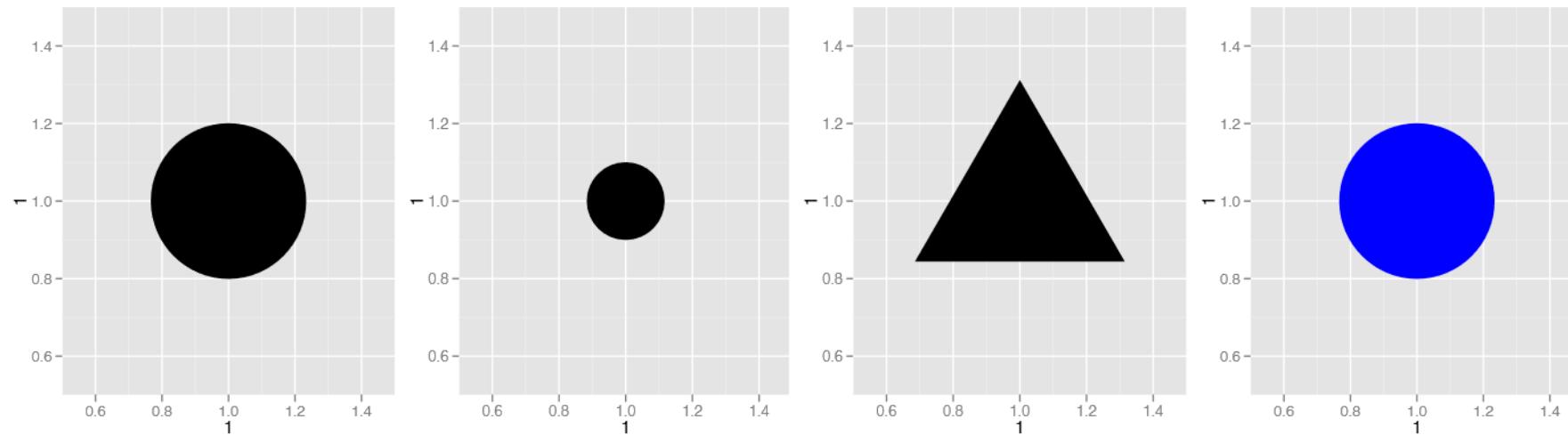
How can we
test the theory?

Why do these
cars get better
mileage?



Aesthetics

Visual properties of a geometric object



How do the appearance of these points vary?



Mappings describe how aesthetics should relate to variables in the data.



Aesthetics

```
ggplot(data = mpg) +  
  geom_point(mapping = aes(x = displ, y = hwy, color = class))
```

aesthetic
property

Variable to
map it to

```
ggplot(data = mpg) +  
  geom_point(mapping = aes(x = displ, y = hwy, size = class))
```



Aesthetics

```
ggplot(data = mpg) +  
  geom_point(mapping = aes(x = displ, y = hwy, color = class))
```

aesthetic
property

Variable to
map it to

```
ggplot(data = mpg) +  
  geom_point(mapping = aes(x = displ, y = hwy, shape = class))
```



Aesthetics

```
ggplot(data = mpg) +  
  geom_point(mapping = aes(x = displ, y = hwy, color = class))
```

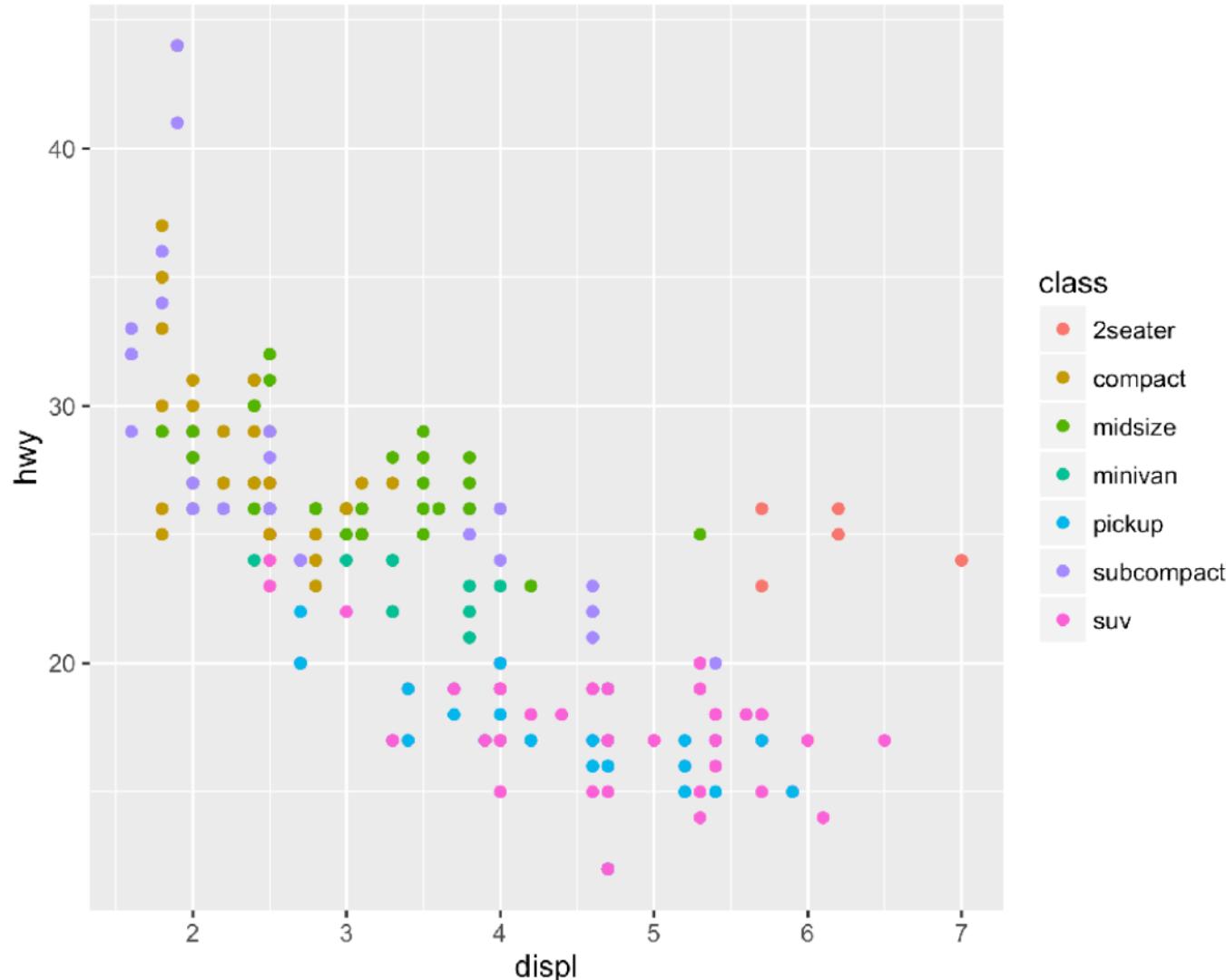
aesthetic
property

Variable to
map it to

```
ggplot(data = mpg) +  
  geom_point(mapping = aes(x = displ, y = hwy, alpha = class))
```



```
ggplot(data = mpg) +  
  geom_point(mapping = aes(x = displ, y = hwy, color = class))
```



Legend
added
automatically



Your Turn 2

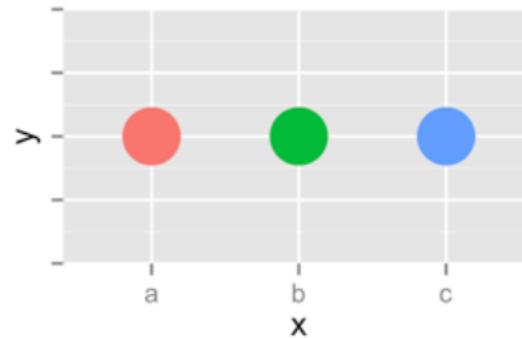
In the next chunk, add color, size, alpha, and shape aesthetics to your graph. Experiment.

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



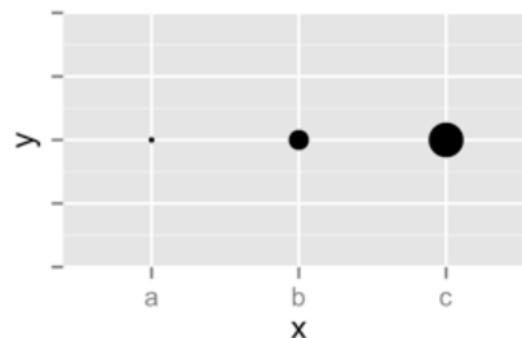
Color

Discrete

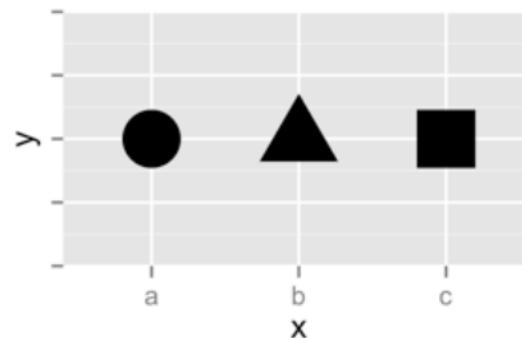


Size

Continuous

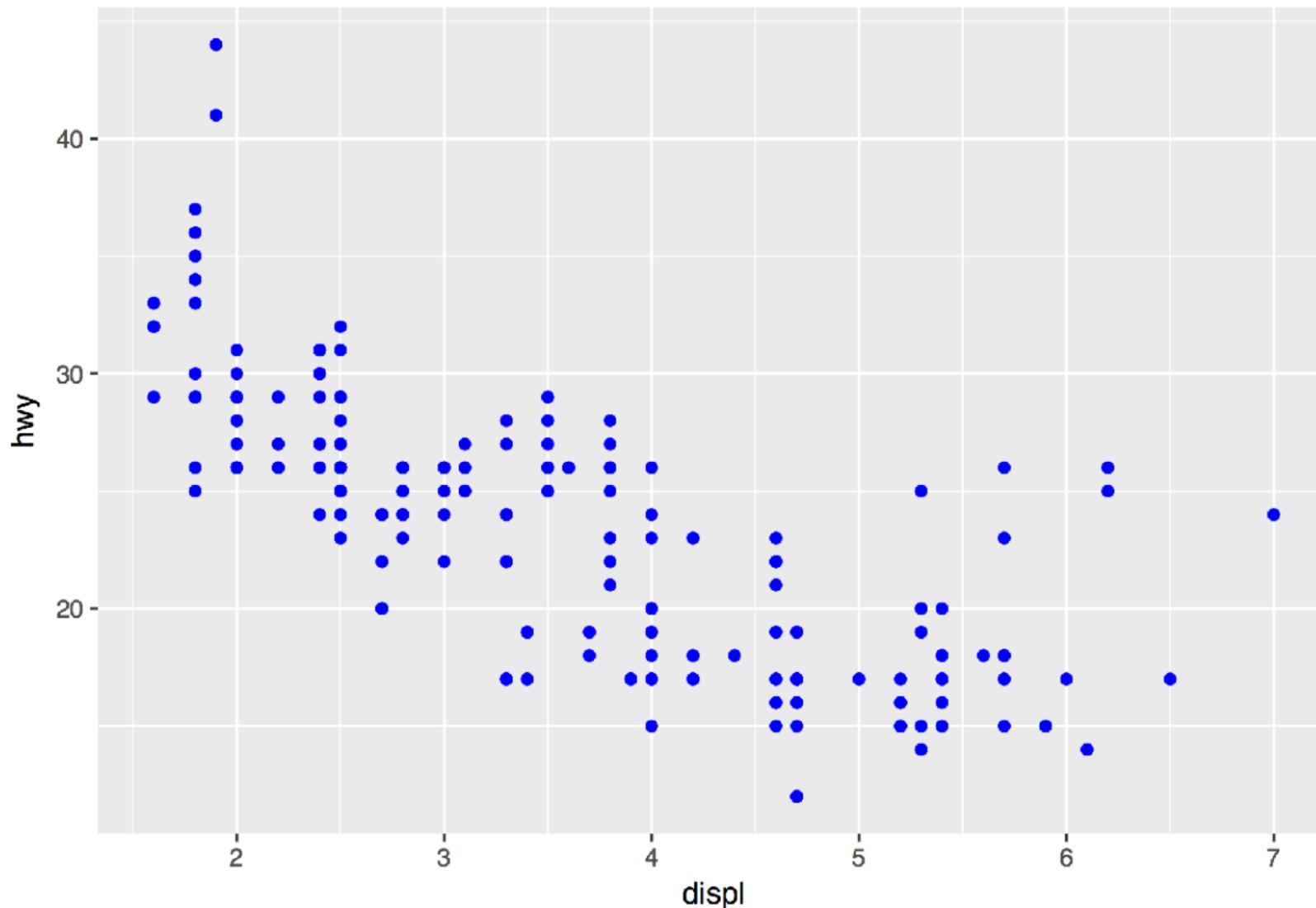


Shape

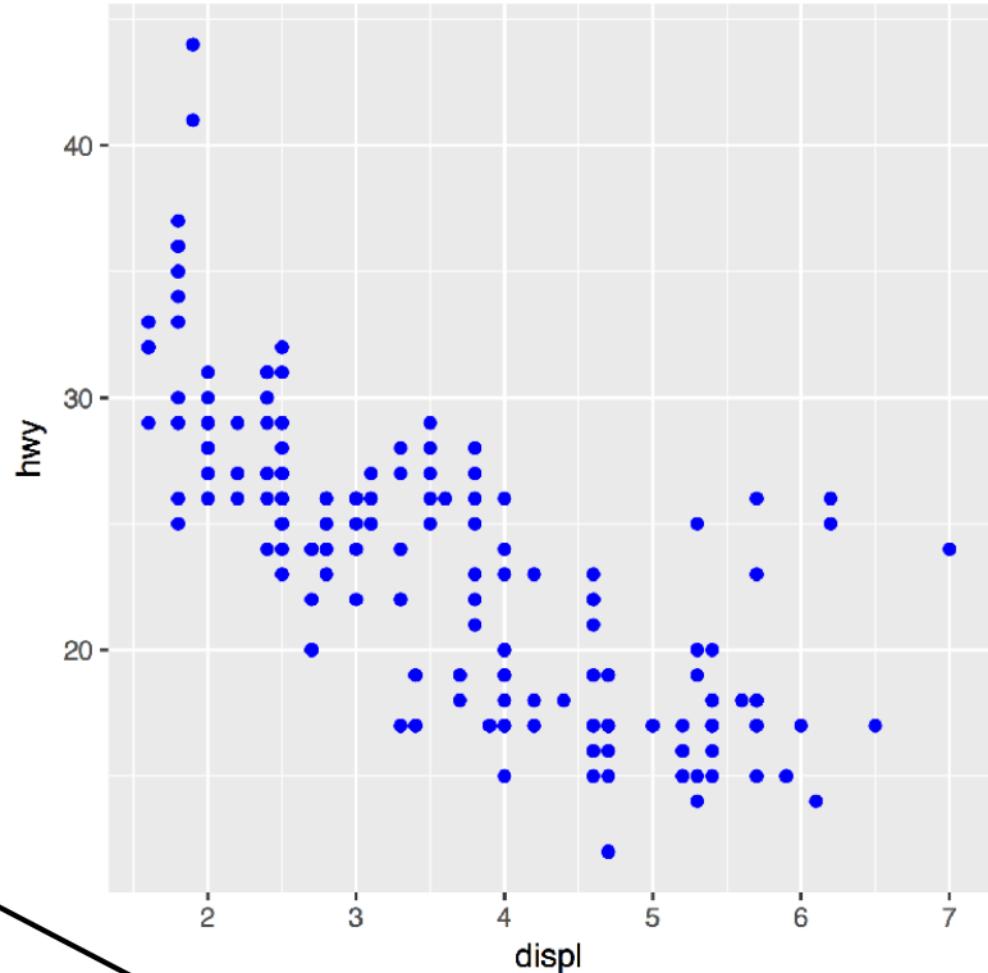


Setting vs. Mapping an aesthetic

How would you make this plot?

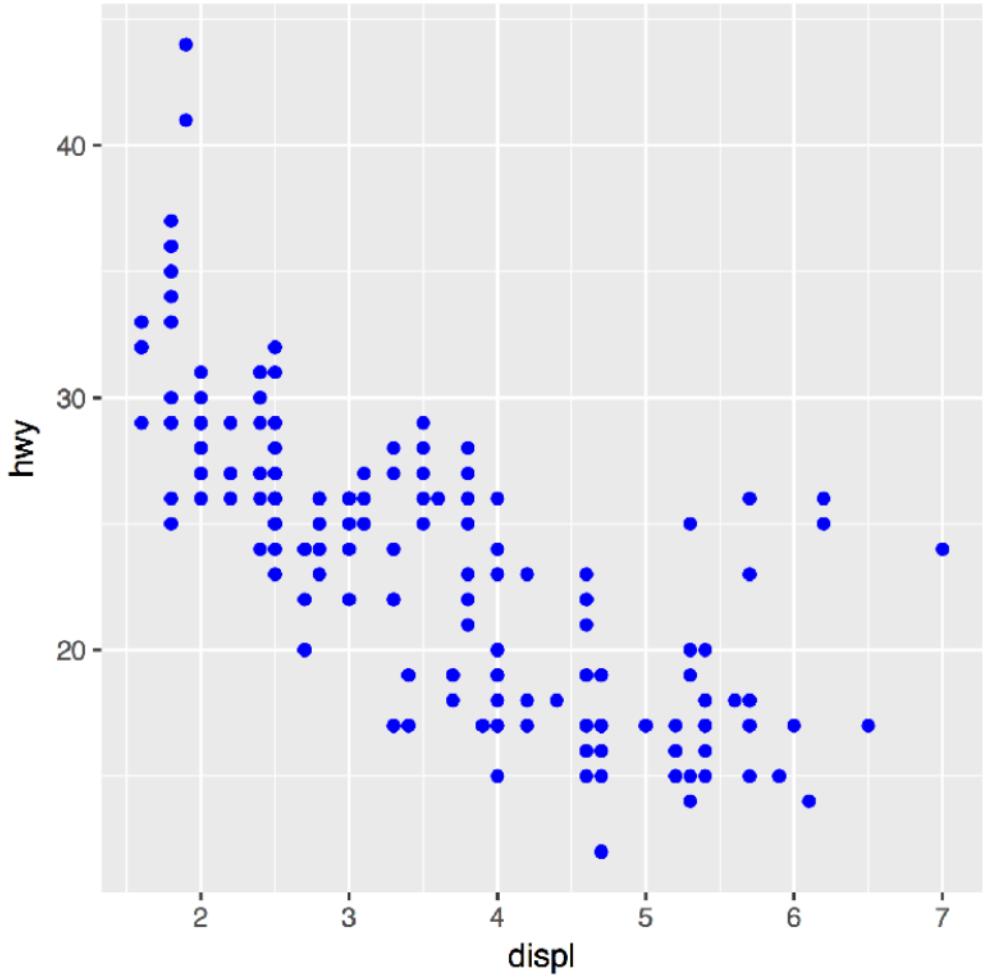
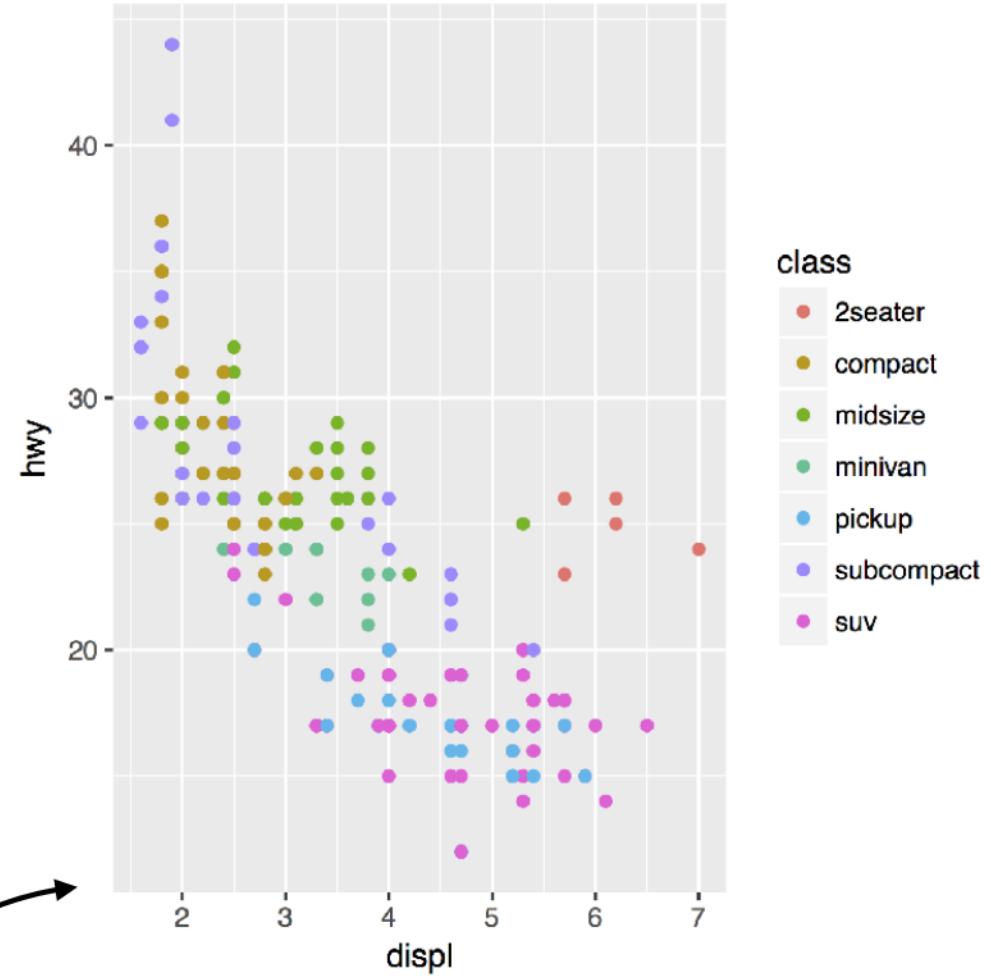


Outside of aes(): sets
an aesthetic to a value



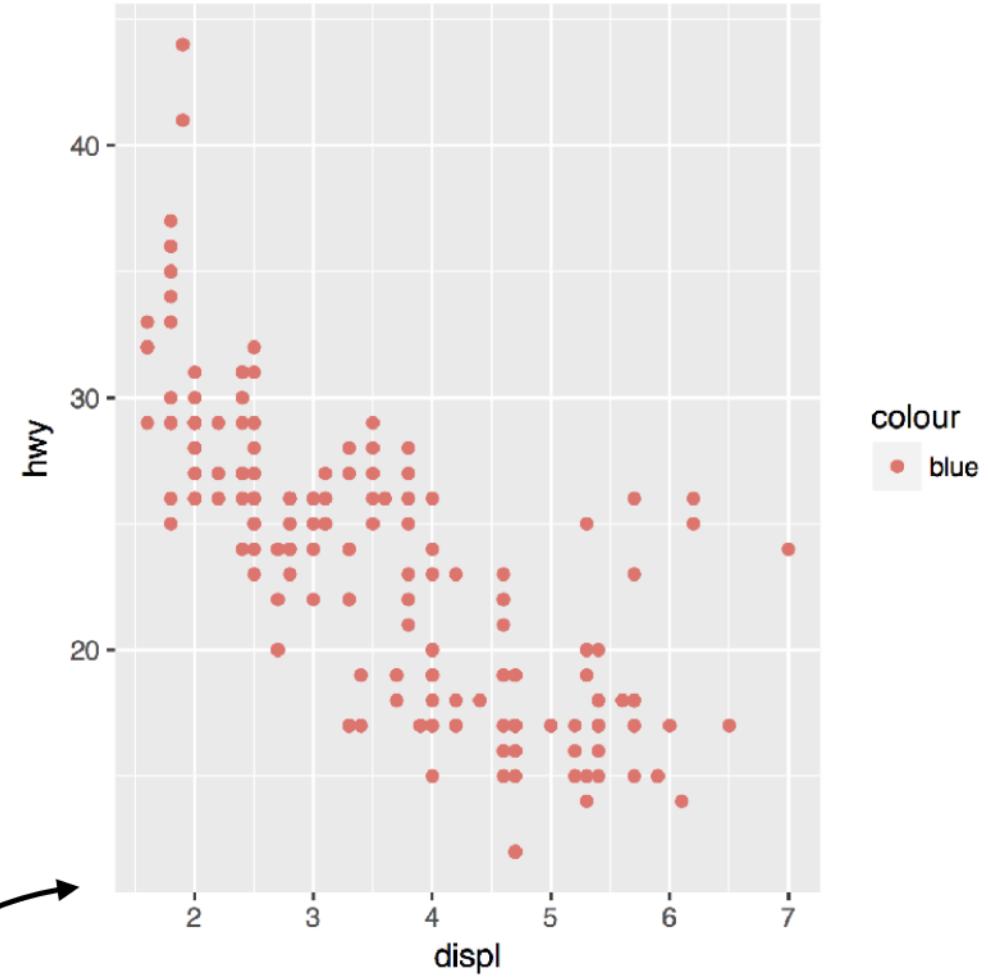
```
ggplot(mpg) + geom_point(aes(x = displ, y = hwy, color = class))
```

```
ggplot(mpg) + geom_point(aes(x = displ, y = hwy), color = "blue")
```

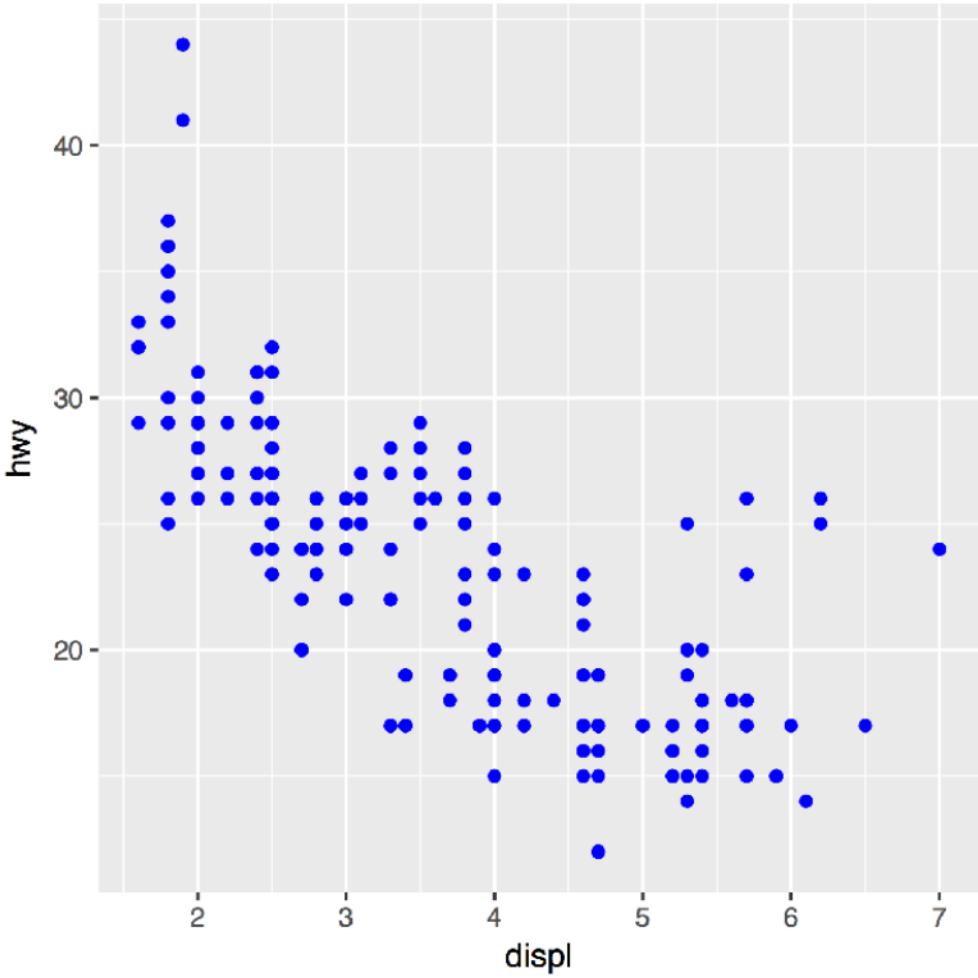


```
ggplot(mpg) + geom_point(aes(x = displ, y = hwy, color = class))
```

```
ggplot(mpg) + geom_point(aes(x = displ, y = hwy), color = "blue")
```



colour
● blue



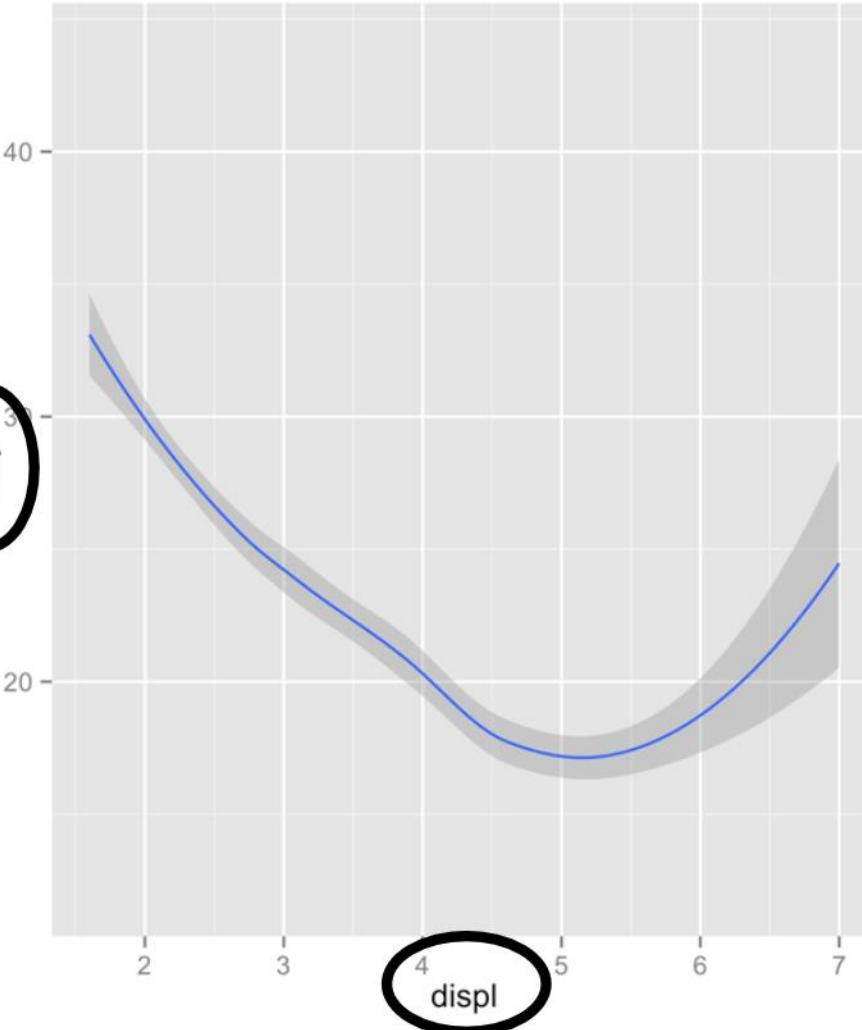
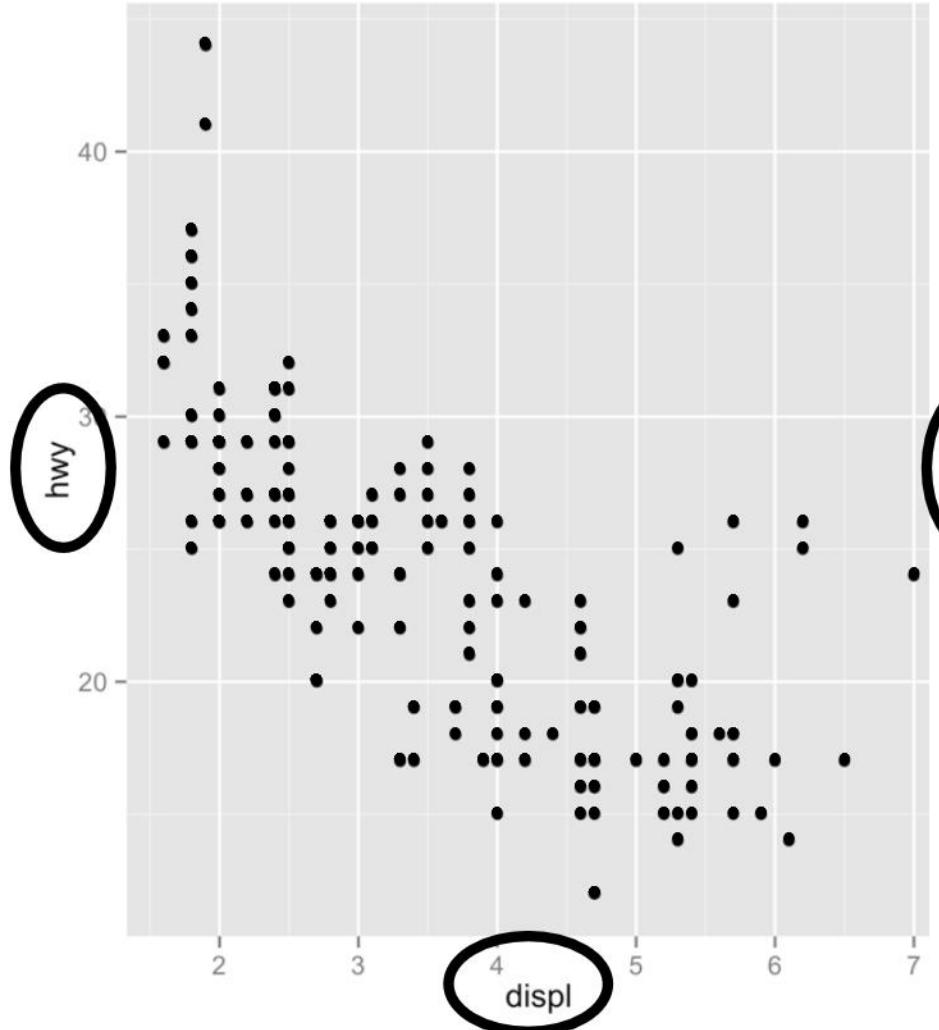
```
ggplot(mpg) + geom_point(aes(x = displ, y = hwy, color = "blue"))
```

```
ggplot(mpg) + geom_point(aes(x = displ, y = hwy), color = "blue")
```

Geoms

How are these plots similar?

How are they different?



geoms

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



Every geom requires a mapping argument.

Data Visualization with ggplot2 :: CHEAT SHEET



R Studio

geom_functions

Geoms

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

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 = 2)) -> x, xend, y, yend,
alpha, angle, color, curvature, linetype, size
a + geom_path(lineend = "butt", linejoin = "round",
linemiter = 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)) -> x, ymax, xmin, ymax,
ymin, alpha, color, fill, linetype, size
a + geom_ribbon(aes(ymin = unemploy - 900,
ymax = unemploy + 900))
x, ymax, ymn, 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(aesthetics = list(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(binaxis = "y", stackdir =
"center")
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 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))
l <- ggplot(seals, aes(long, lat))
l + geom_point(aes(group = seal))
l + geom_text(aes(label = seal, nudge_x = 1,
nudge_y = 1, check_overlap = TRUE))
x, y, 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, alpha, color, family, fontface, hjust,
lineheight, size, vjust
e + geom_linerp(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, alpha, angle, color, family, fontface, hjust,
lineheight, size, vjust
```

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

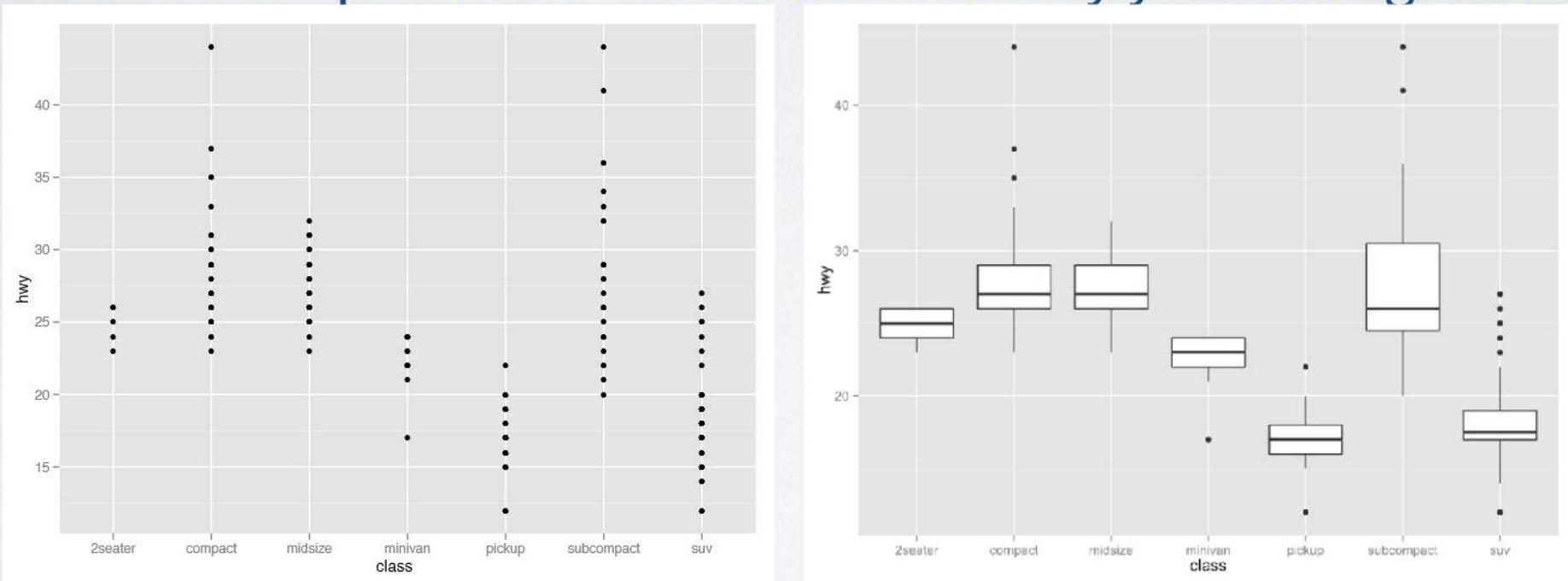
```
df <- data.frame(grp = c("A", "B"), fit = 4.5, se = 1.2)
j <- ggplot(df, aes(grp, fit, ymin = fit - se, ymax = fit + se))
j + geom_crossbar(fatten = 2)
x, y, ymin, ymax, alpha, color, fill, group, linetype, size
j + geom_errorbar()
x, y, ymin, ymax, alpha, color, group, linetype, size, width (also
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, size
```

maps

```
data <- data.frame(murder = USArrests$Murder,
state = tolower(rownames(USArrests)))
map <- map_data("state")
k <- ggplot(data, aes(fill = murder))
k + geom_map(aes(map_id = state), map = map)
+ expand_limits(x = map$long, y = map$lat),
map_id, alpha, color, fill, linetype, size
```

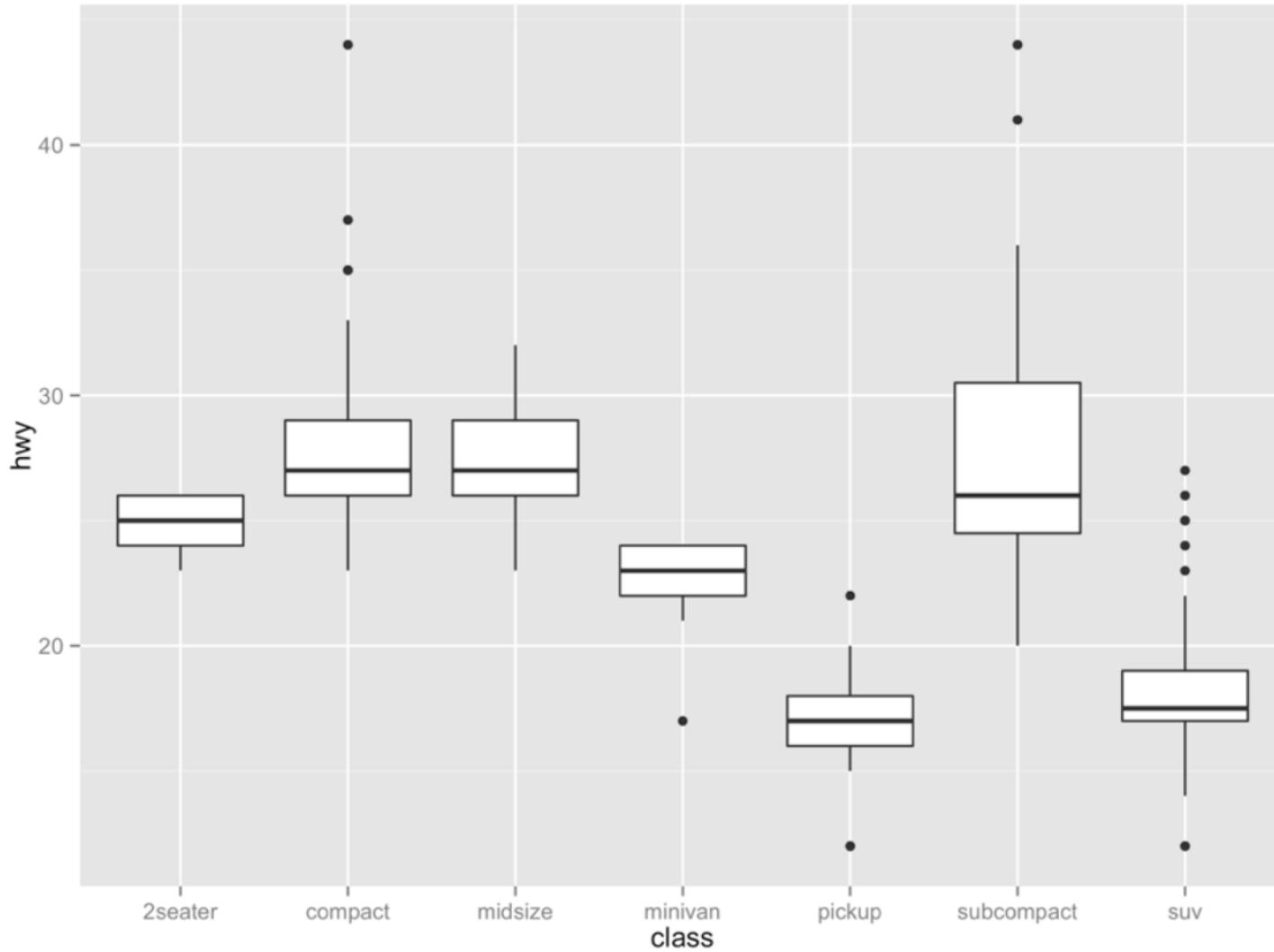
Your Turn 3

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



`ggplot(mpg) + geom_point(aes(class, hwy))`

02 : 00

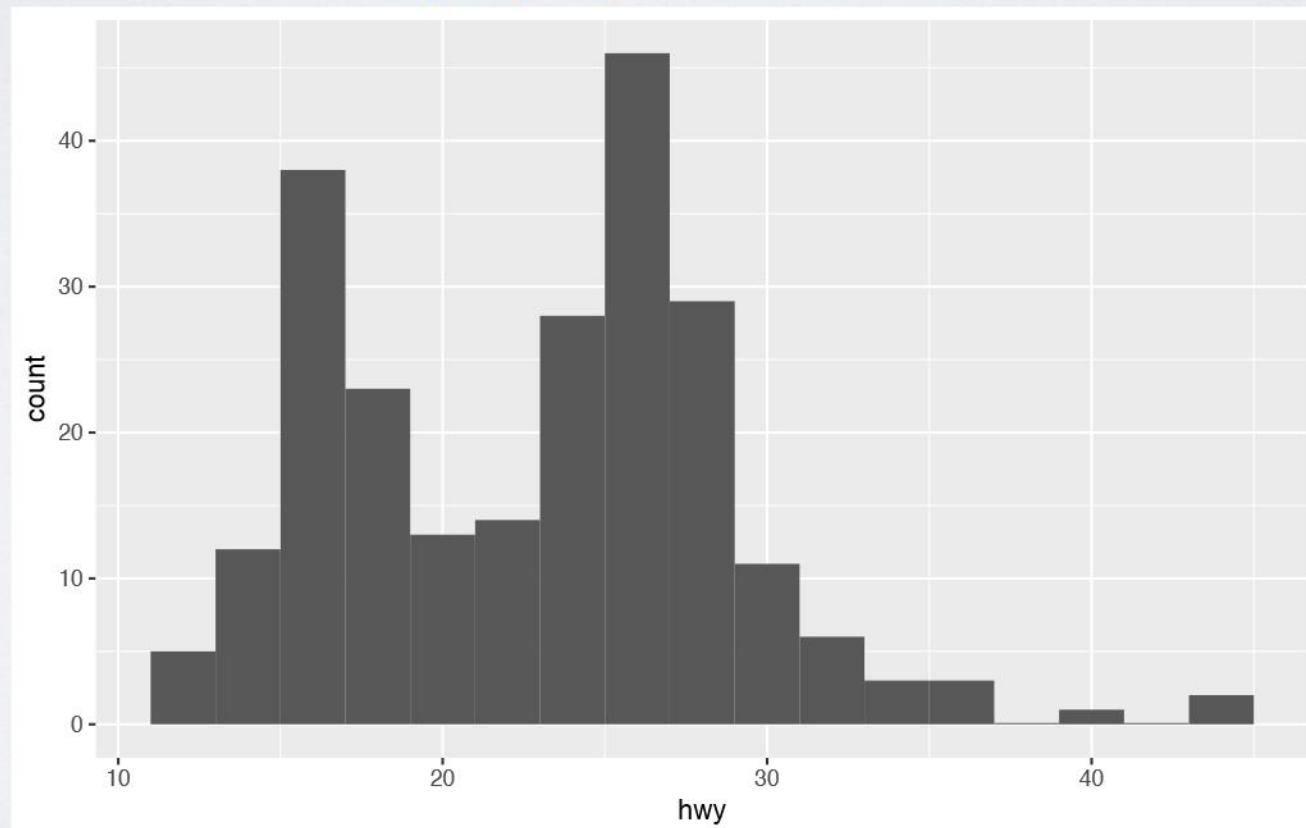


```
ggplot(data = mpg) +  
  geom_boxplot(mapping = aes(x = class, y = hwy))
```

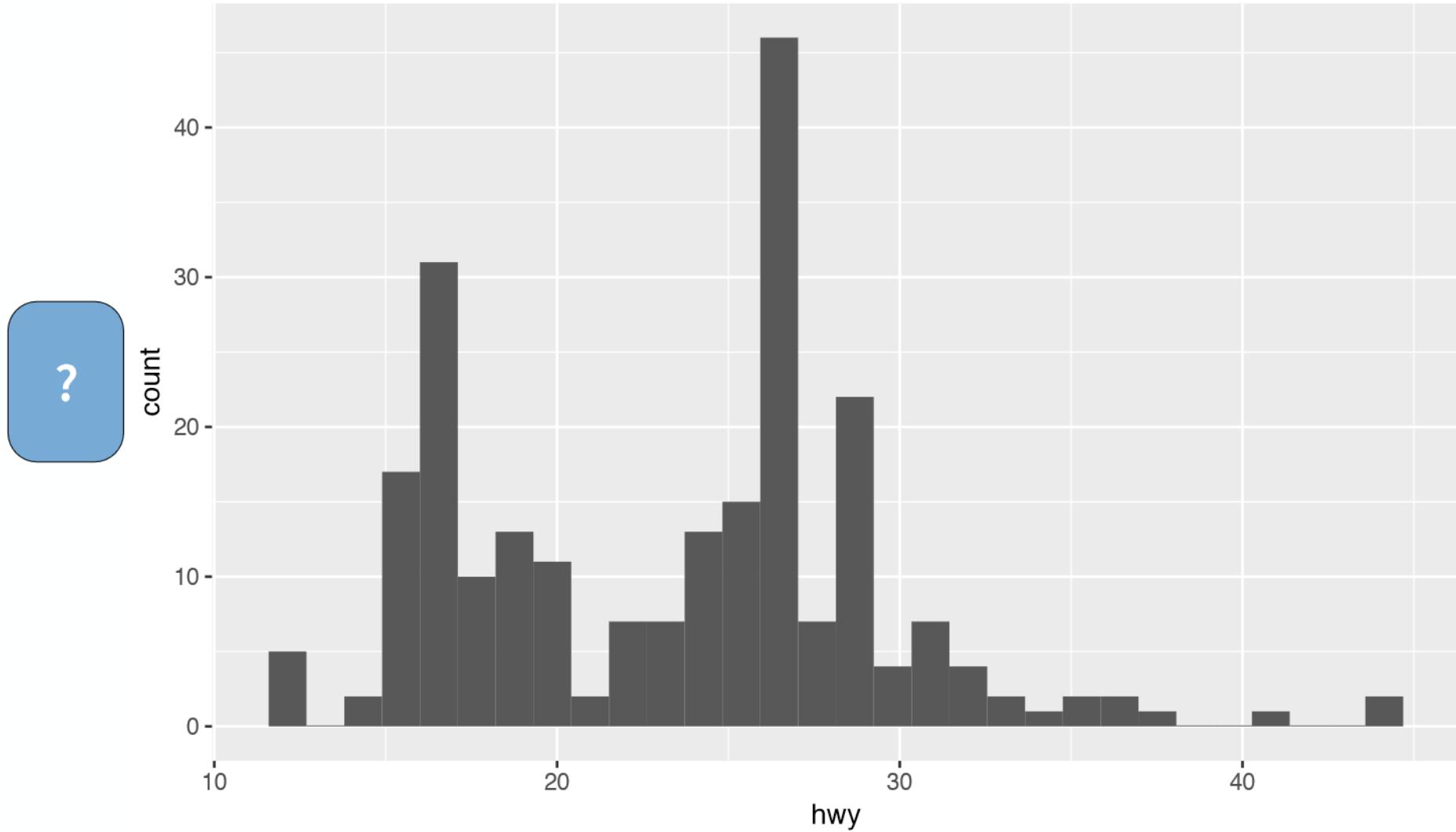


Your Turn 4

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



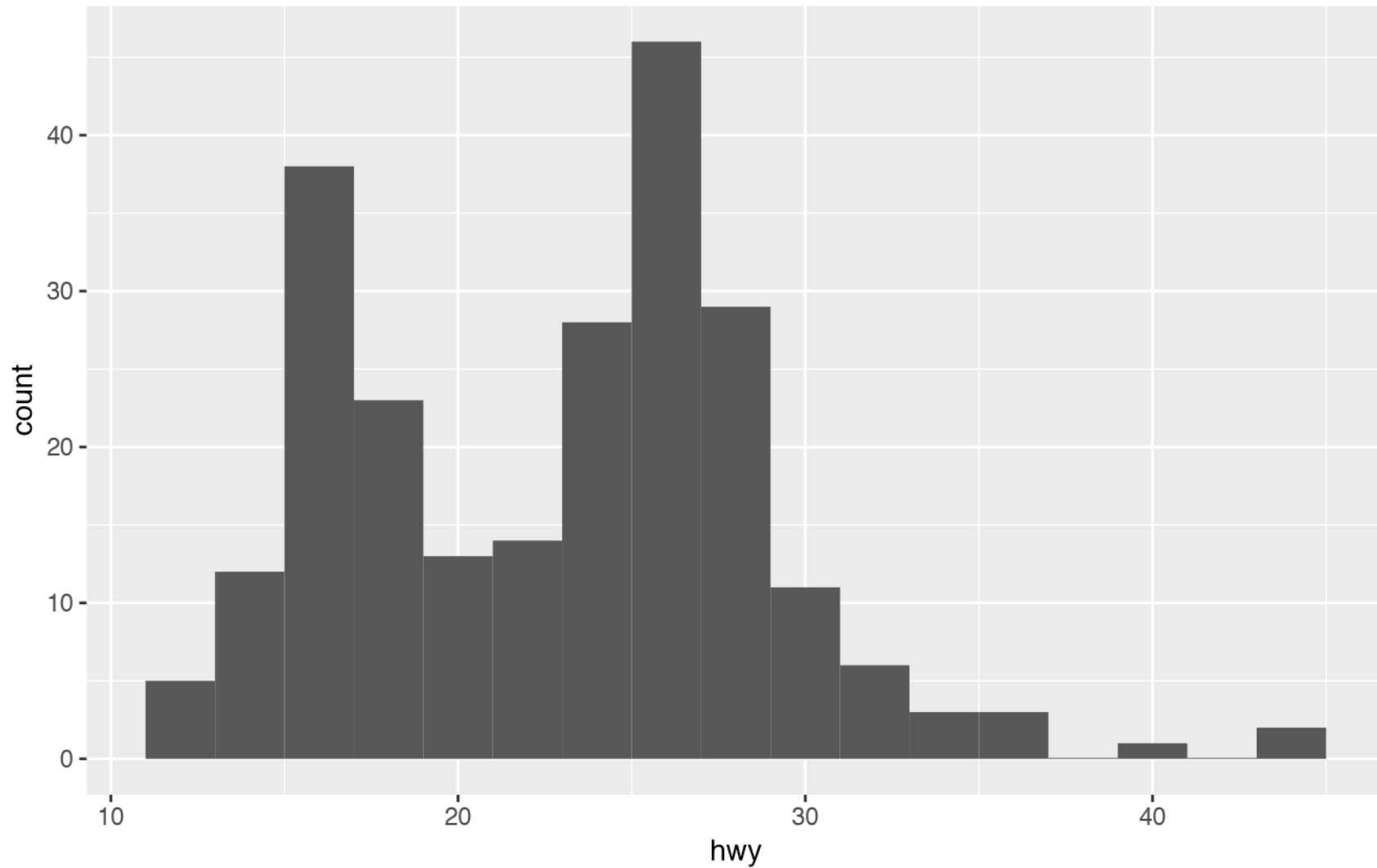
02 : 00



```
ggplot(data = mpg) +  
  geom_histogram(mapping = aes(x = hwy))
```

No y aesthetic

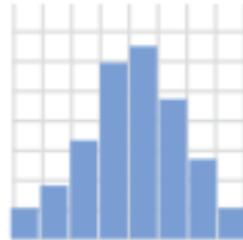




```
ggplot(data = mpg) +  
  geom_histogram(mapping = aes(x = hwy), binwidth = 2)
```



On the cheatsheat:



**c + geom_histogram(binwidth = 5) x, y, alpha,
color, fill, linetype, size, weight**

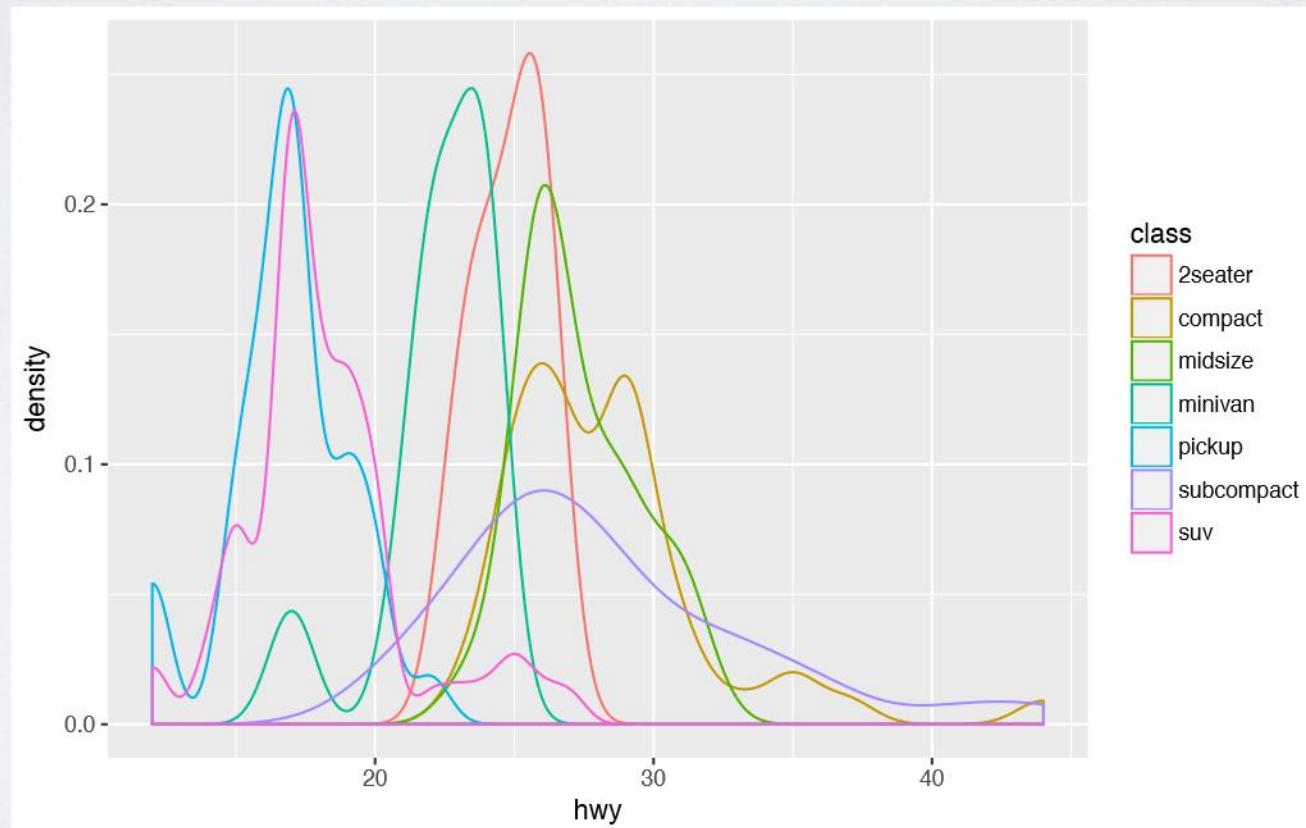
Arguments inside (), are
geom specific options

Outside the (), are
aesthetics that can be
mapped or set.

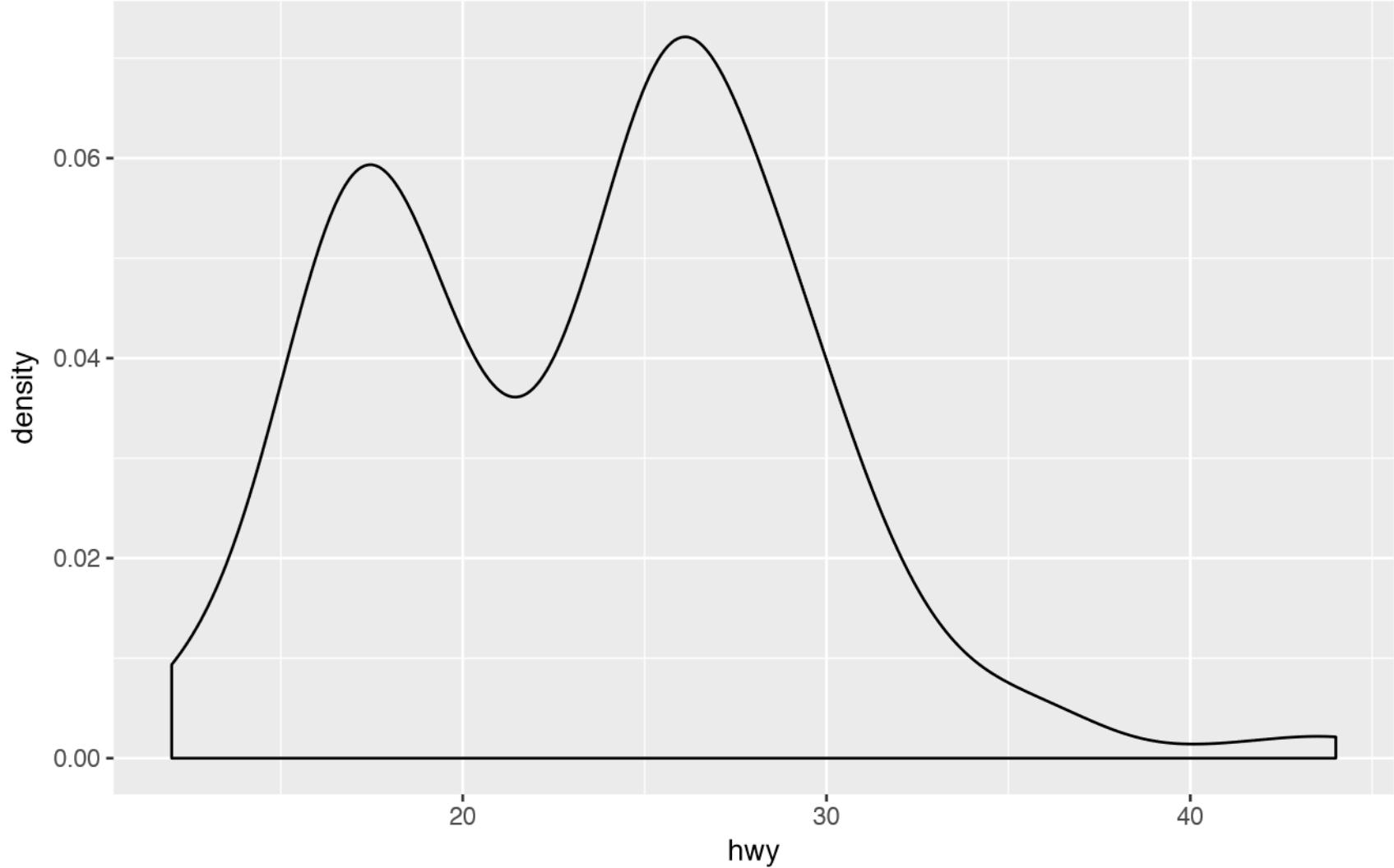


Your Turn 5

With your partner, make the density plot of `hwy` colored by `class` below. Use the cheatsheet. Try your best guess.

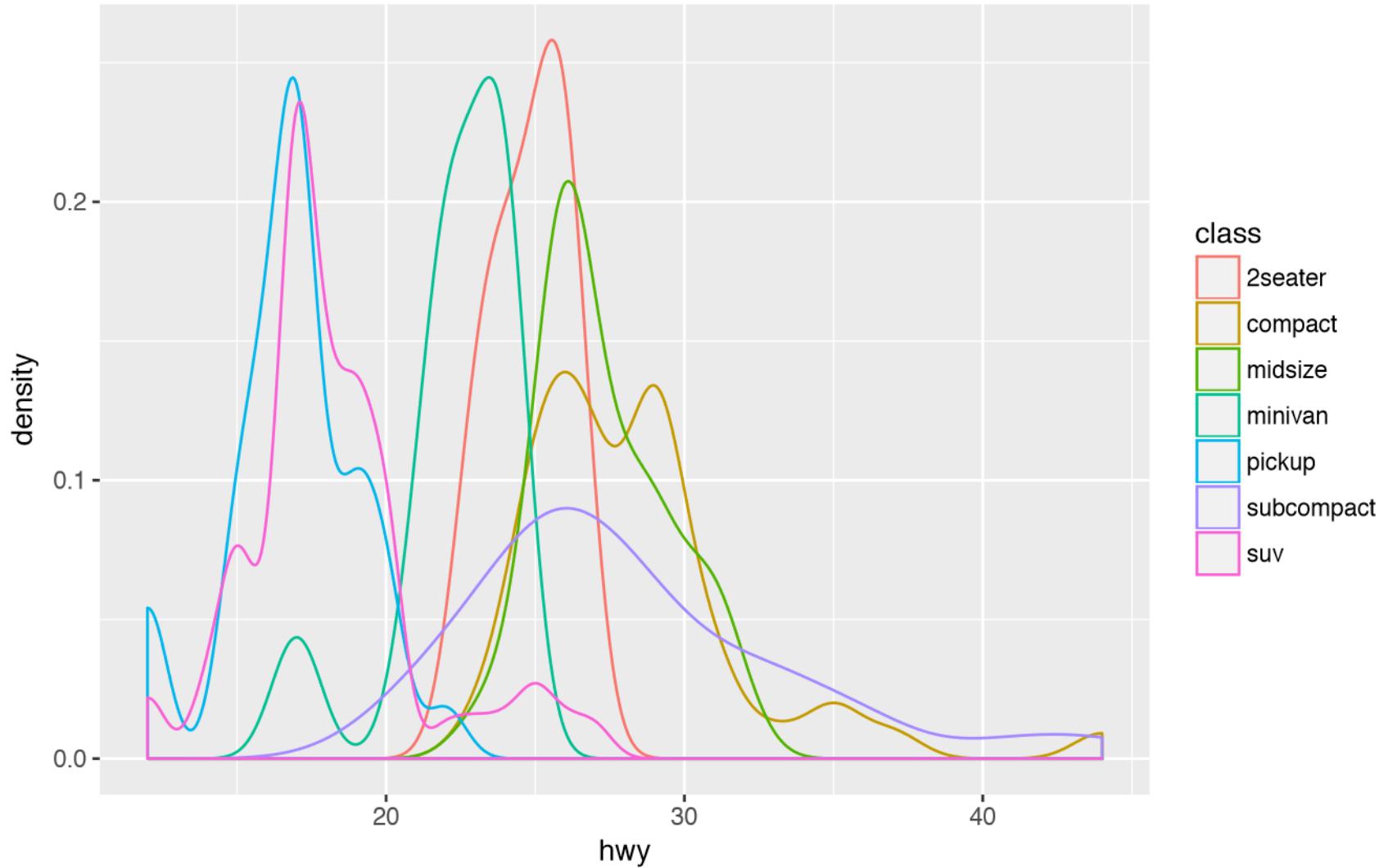


02 : 00

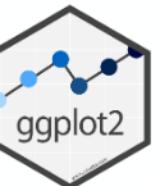


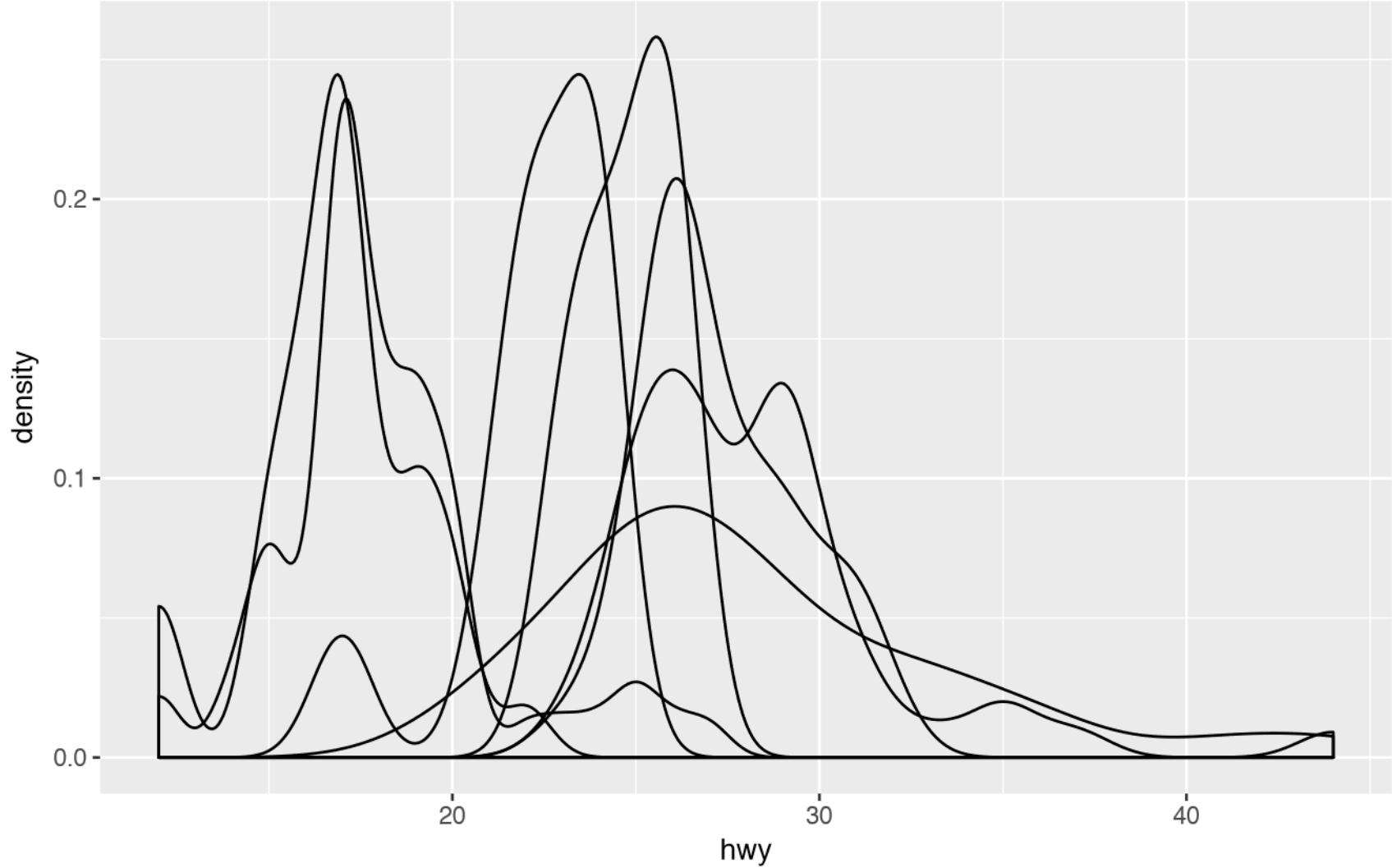
```
ggplot(data = mpg) +  
  geom_density(mapping = aes(x = hwy))
```





```
ggplot(data = mpg) +  
  geom_density(mapping = aes(x = hwy, color = class))
```



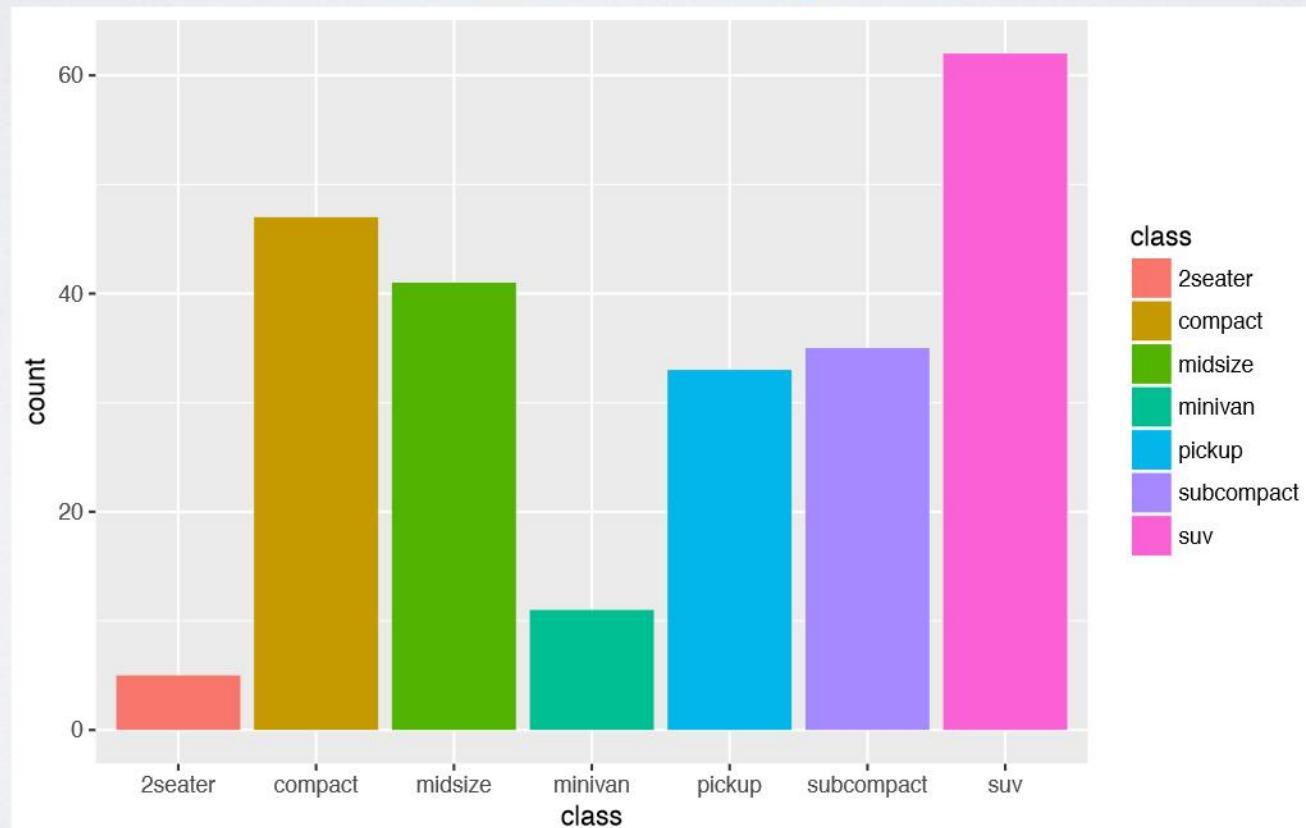


```
ggplot(data = mpg) +  
  geom_density(mapping = aes(x = hwy, group = class))
```

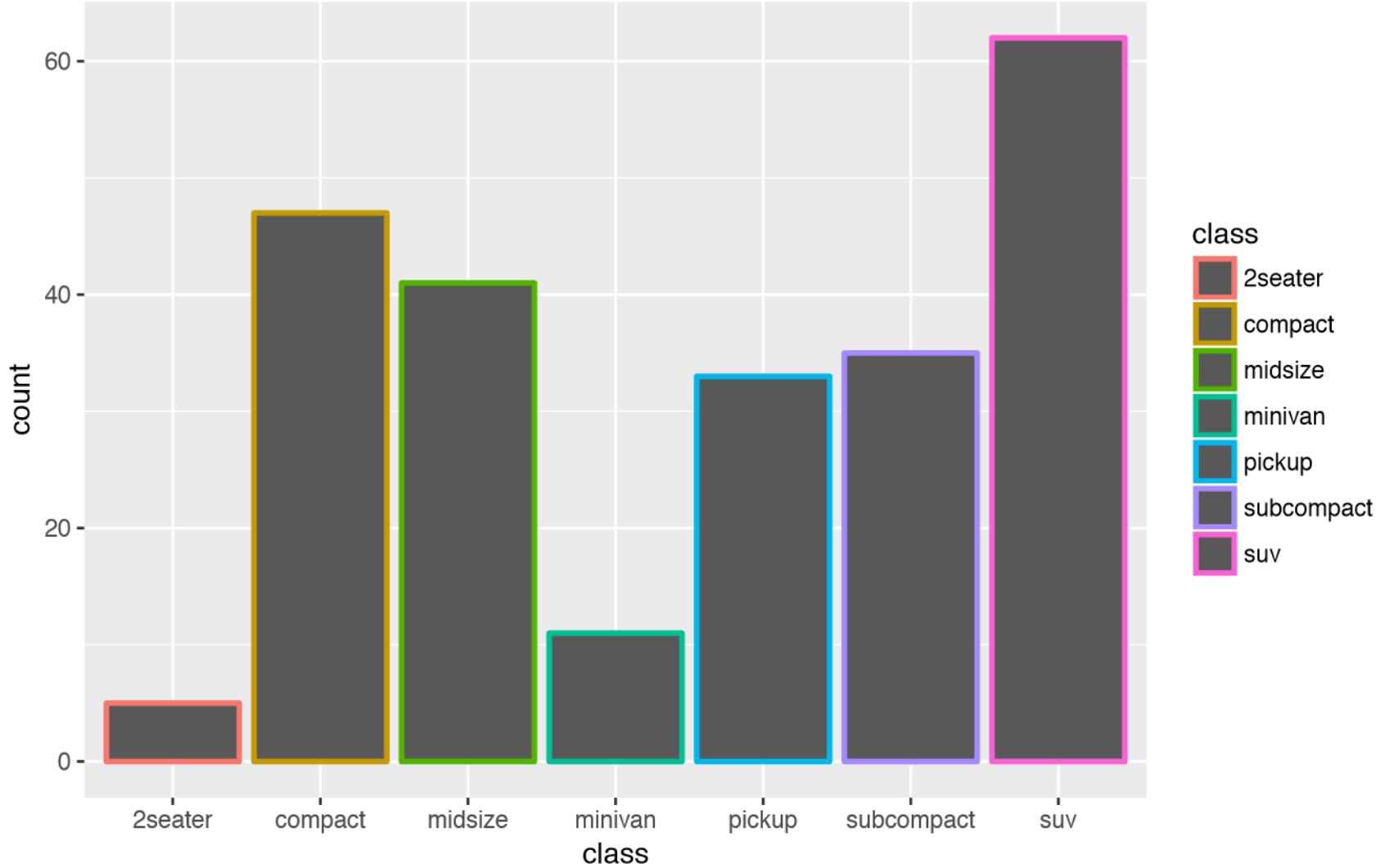


Your Turn 6

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

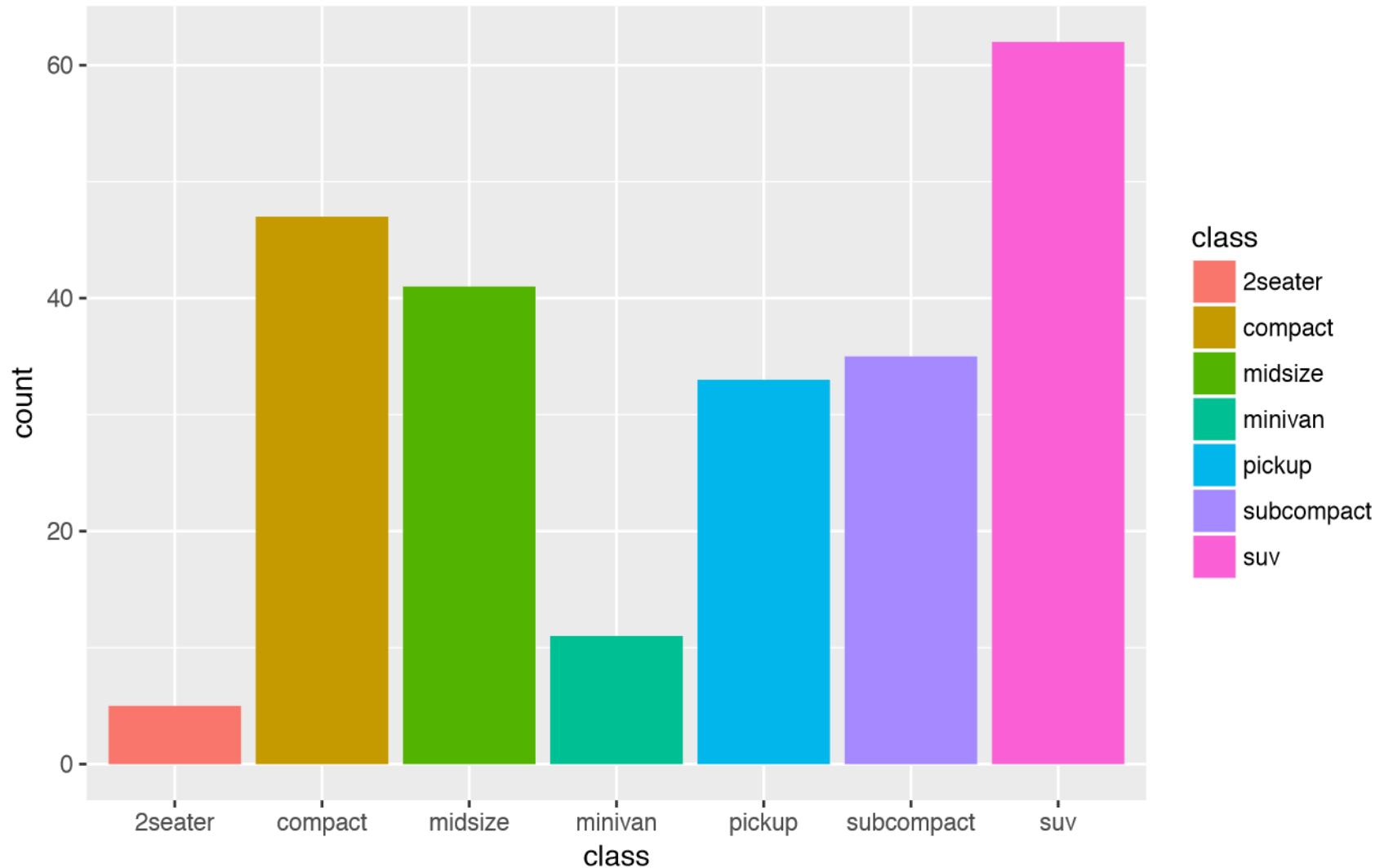


02 : 00



```
ggplot(data = mpg) +  
  geom_bar(mapping = aes(x = class, color = class))
```





```
ggplot(data = mpg) +  
  geom_bar(mapping = aes(x = class, fill = class))
```

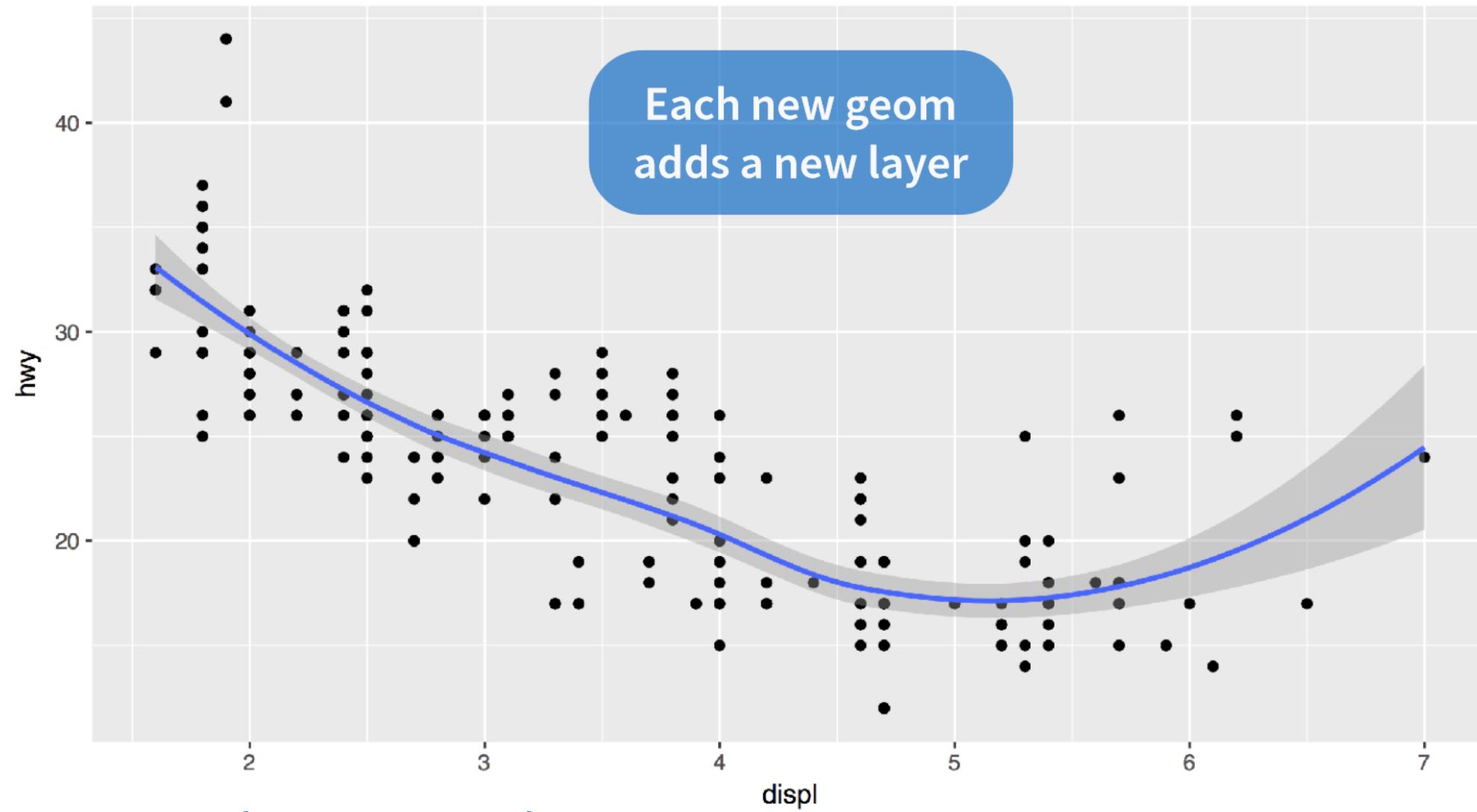


Your Turn 7

With your partner, predict what this code will do.
Then run it.

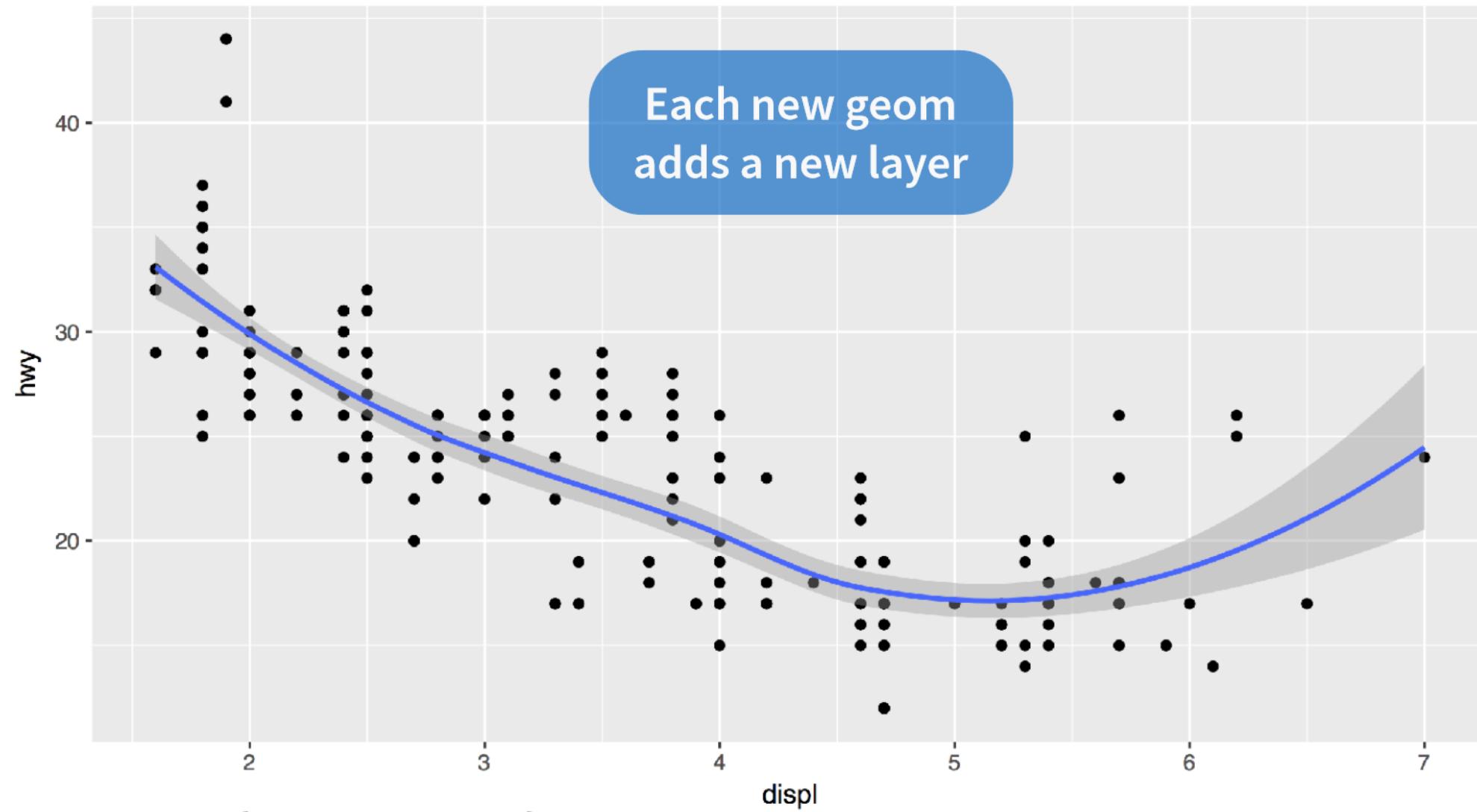
```
ggplot(mpg) +  
  geom_point(aes(displ, hwy)) +  
  geom_smooth(aes(displ, hwy))
```





```
ggplot(data = mpg) +  
  geom_point(mapping = aes(x = displ, y = hwy)) +  
  geom_smooth(mapping = aes(x = displ, y = hwy))
```

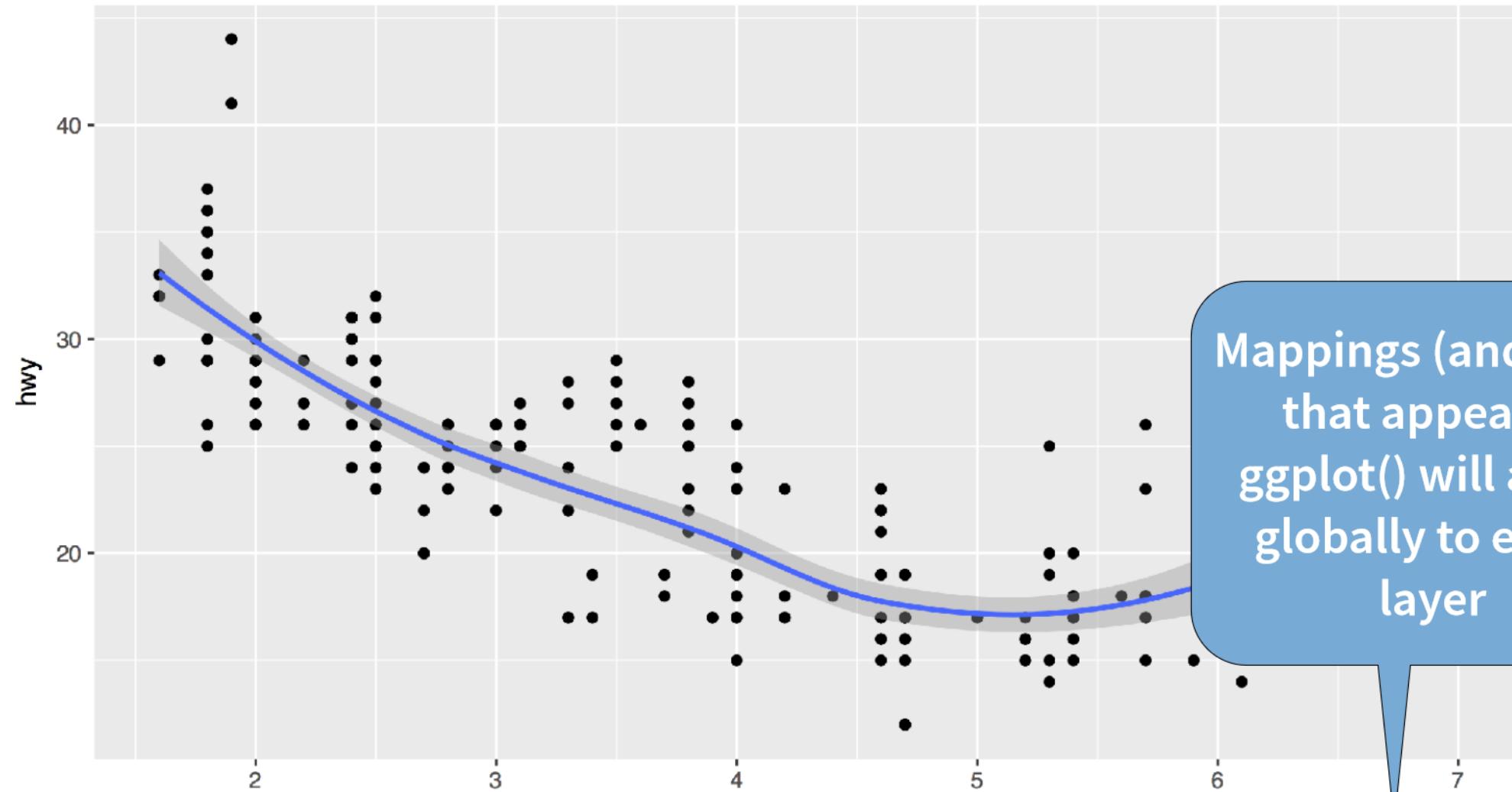




```
ggplot(data = mpg) +  
  geom_point(mapping = aes(x = displ, y = hwy)) +  
  geom_smooth(mapping = aes(x = displ, y = hwy))
```

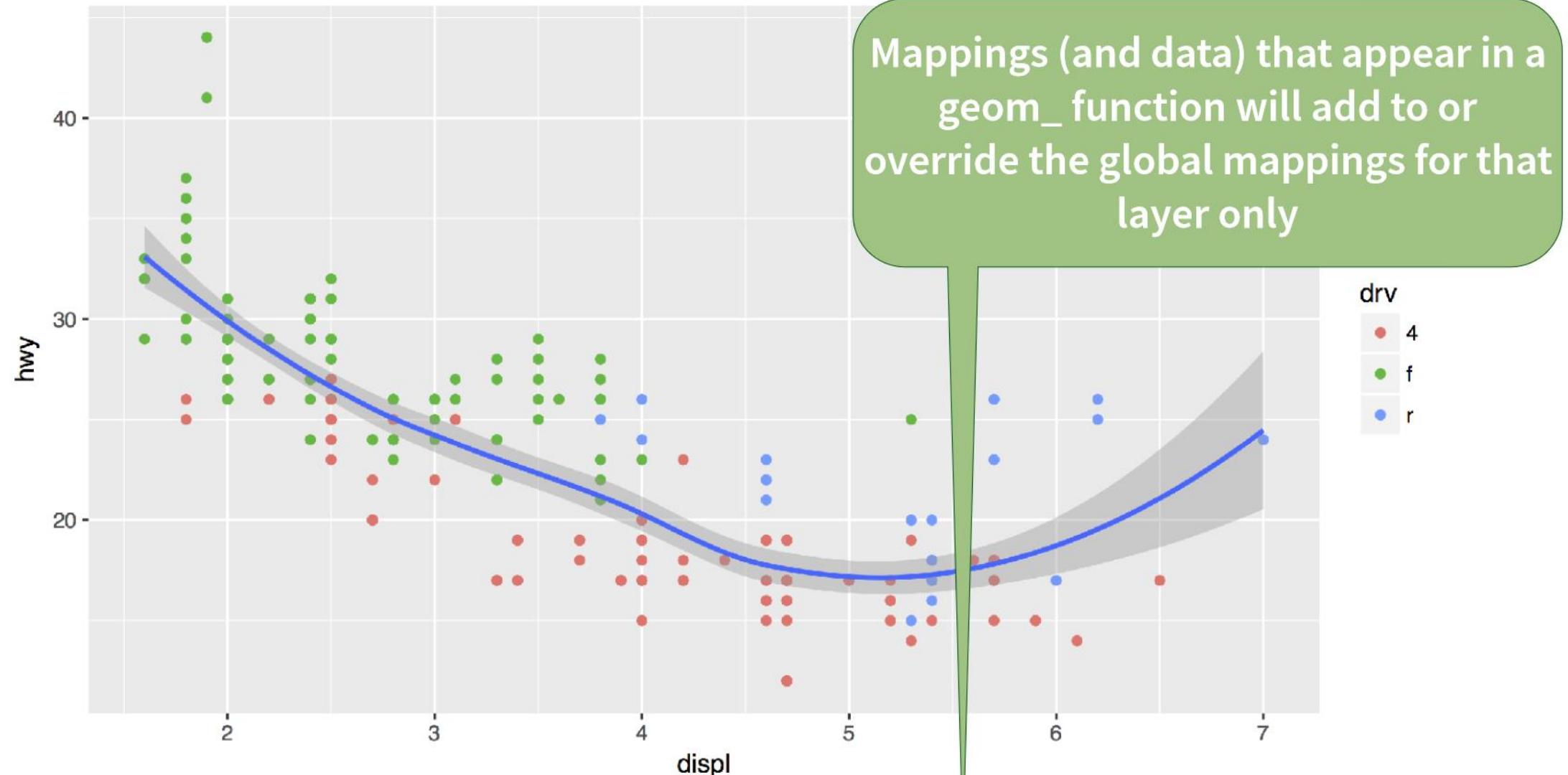


Global vs. Local



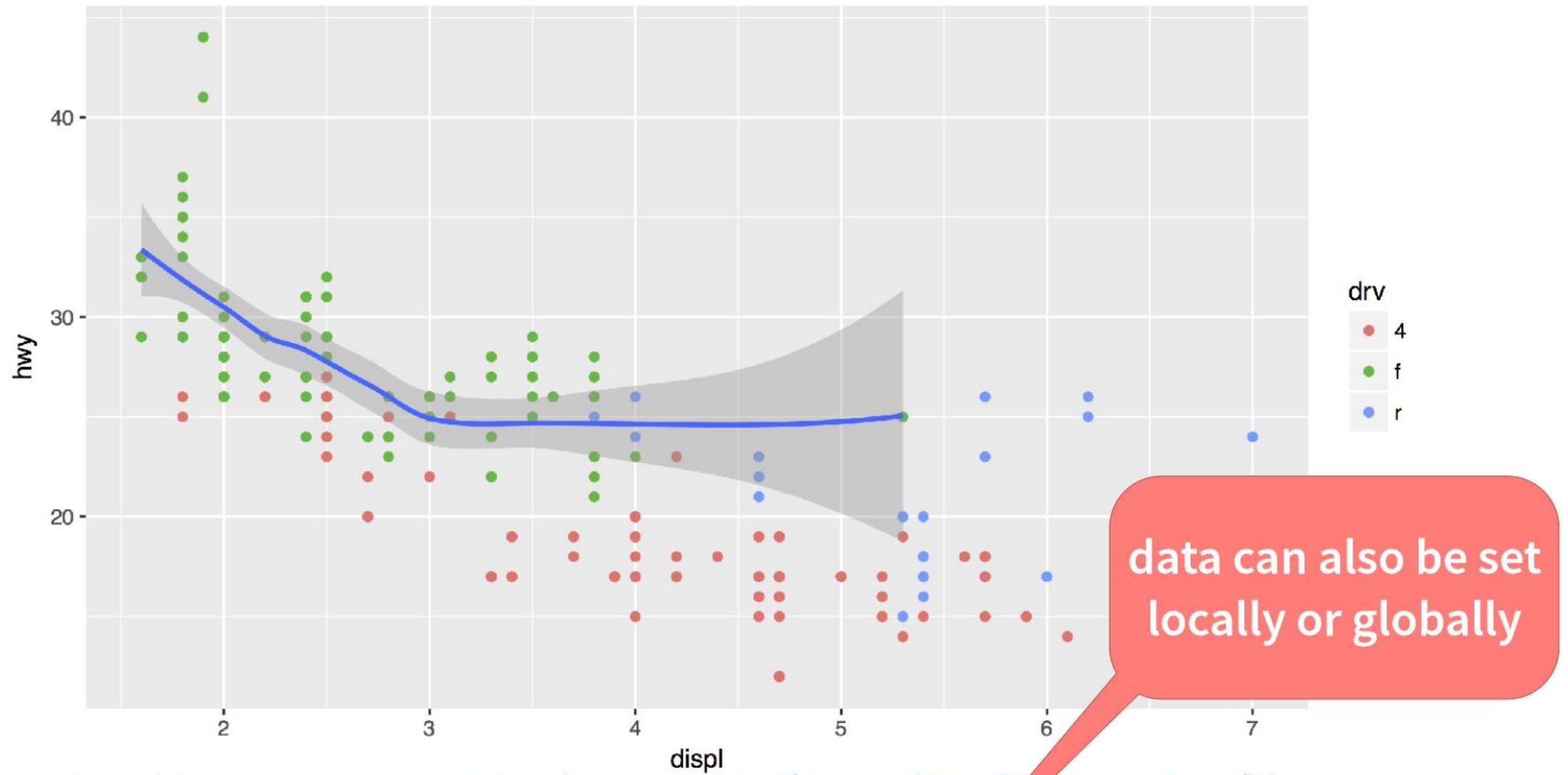
```
ggplot(data = mpg, mapping = aes(x = displ, y = hwy)) +  
  geom_point() +  
  geom_smooth()
```





```
ggplot(data = mpg, mapping = aes(x = displ, y = hwy)) +  
  geom_point(mapping = aes(color = drv)) +  
  geom_smooth()
```





```
ggplot(data = mpg, mapping = aes(x = displ, y = hwy)) +  
  geom_point(mapping = aes(color = drv)) +  
  geom_smooth(data = filter(mpg, drv == "f"))
```



Exporting Graphs

Your Turn 8

What does this command return?

`getwd()`

00 : 30

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

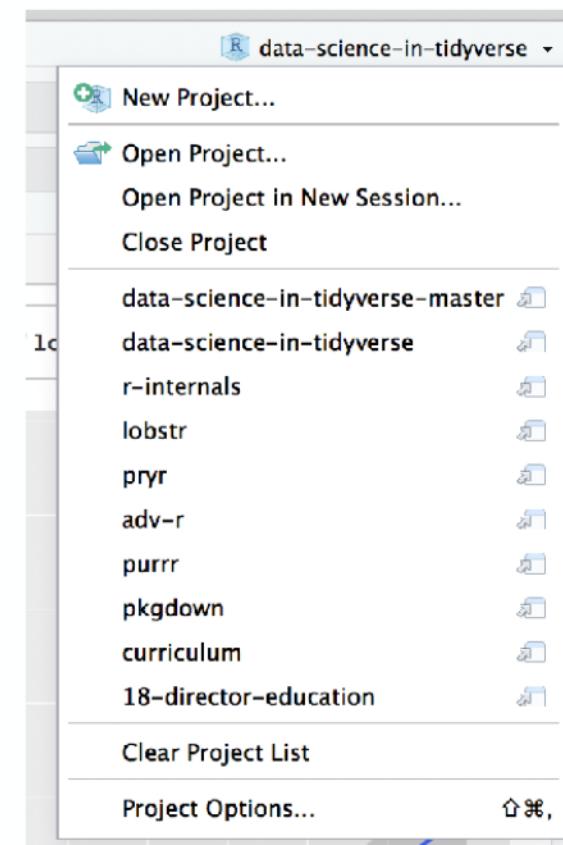


Projects

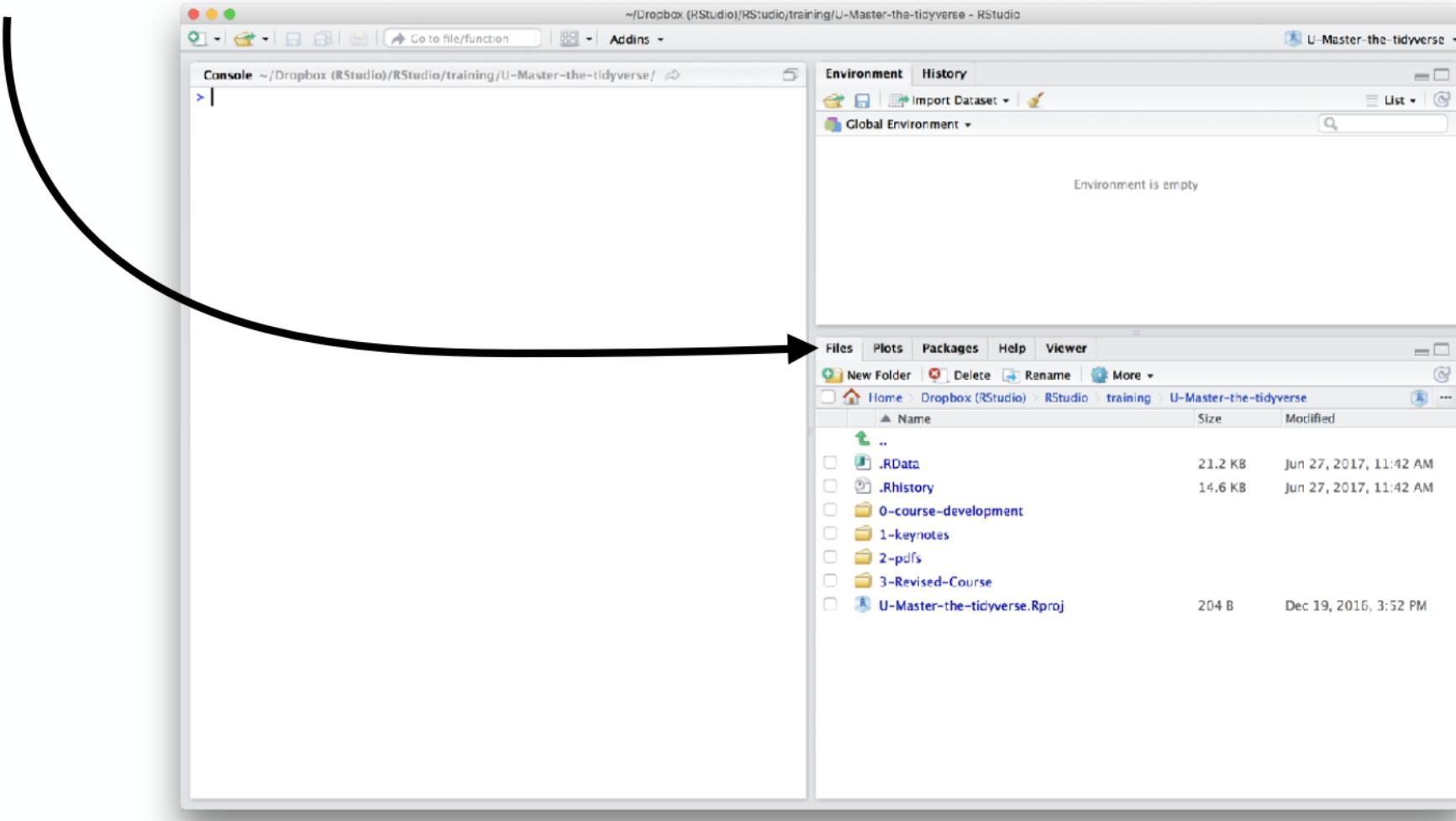
The best way of managing your working directory is with RStudio Projects.

One RStudio project = one real life project

One RStudio project = one directory



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



Saving plots

`ggsave()` saves the last plot.

Uses size on screen:

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

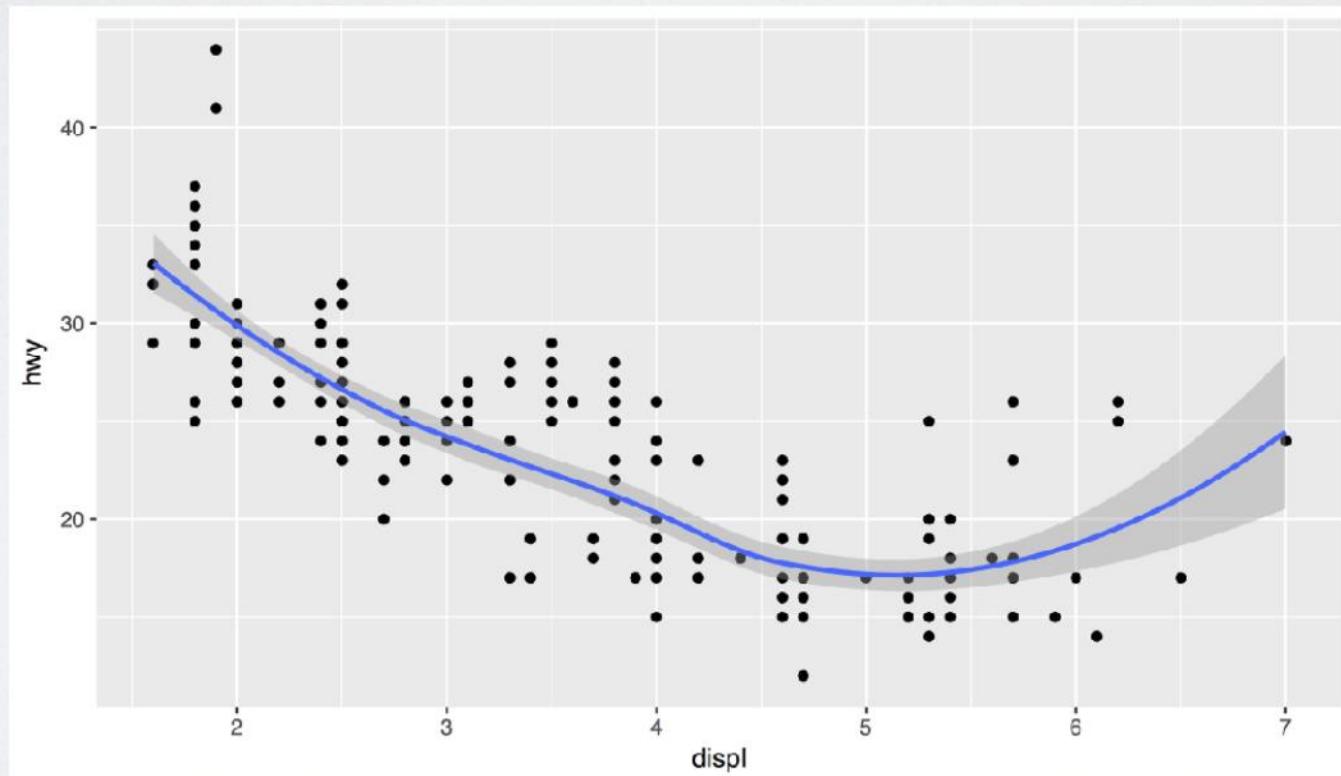
Specify size in inches

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



Your Turn 9

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



01 : 00

The Grammar of Graphics

To make a graph

[template]

```
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

1. Pick a **data** set

```
ggplot(data = <DATA>) +  
<GEO_M_FUNCTION>(mapping = aes(<MAPPIINGS>))
```



To make a graph

mpg	cyl	disp	hp	geom
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	•

1. Pick a **data** set

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

2. Choose a **geom**
to display cases

data

geom



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

To make a graph

1. Pick a **data** set

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

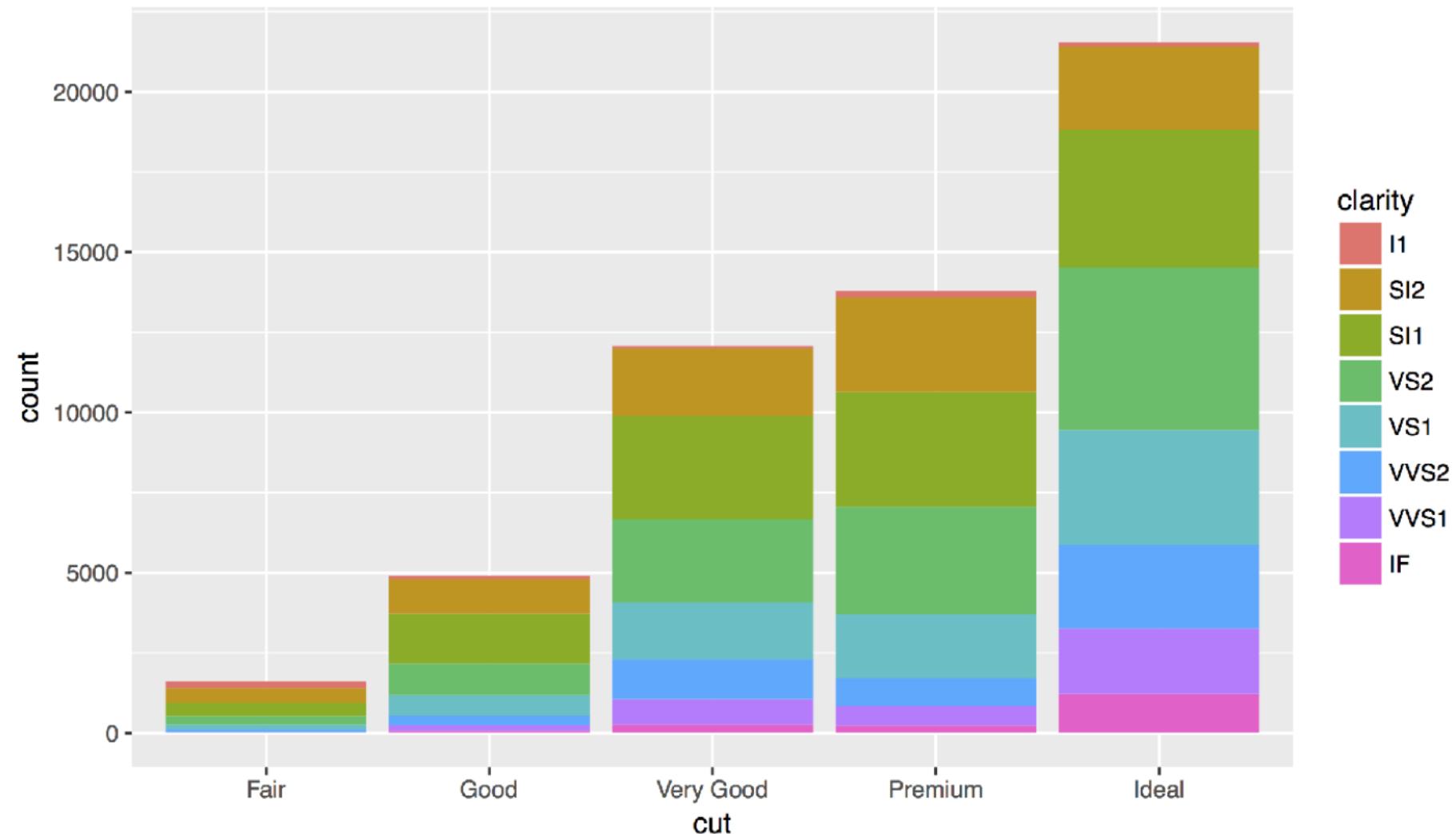
2. Choose a **geom**
to display cases

3. **Map** aesthetic
properties to
variables

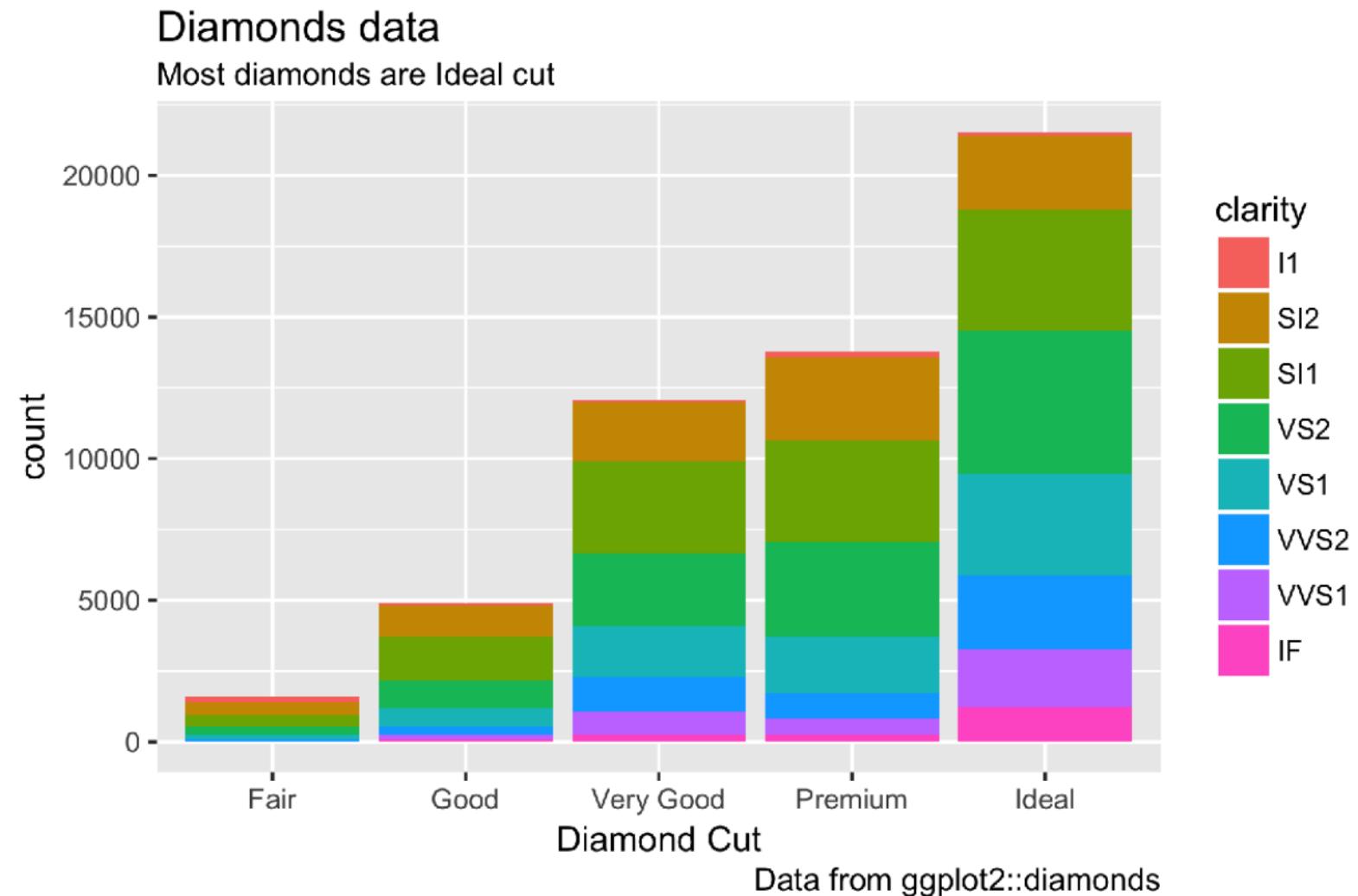


What else?

For example:

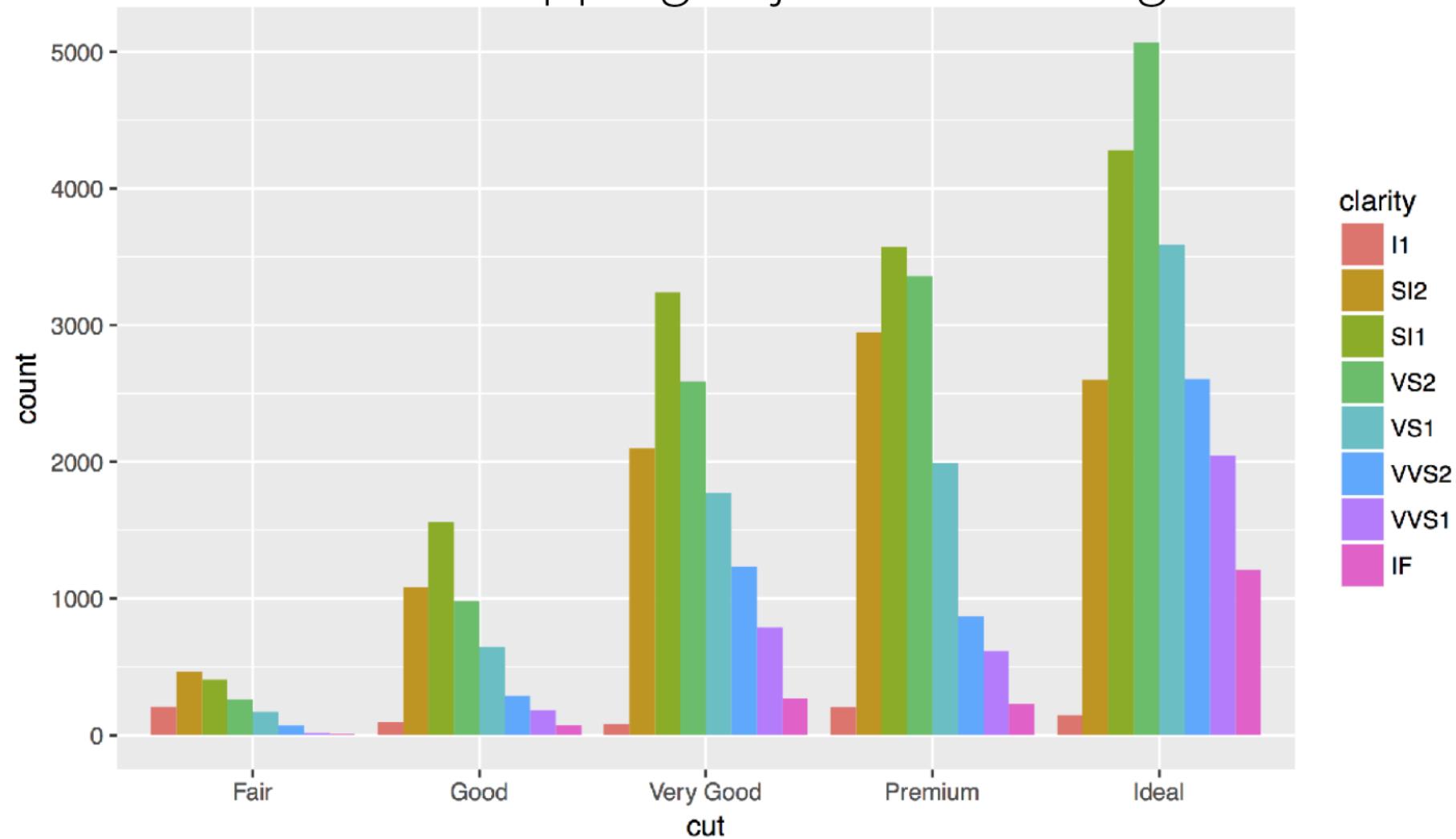


Titles and captions + labs()



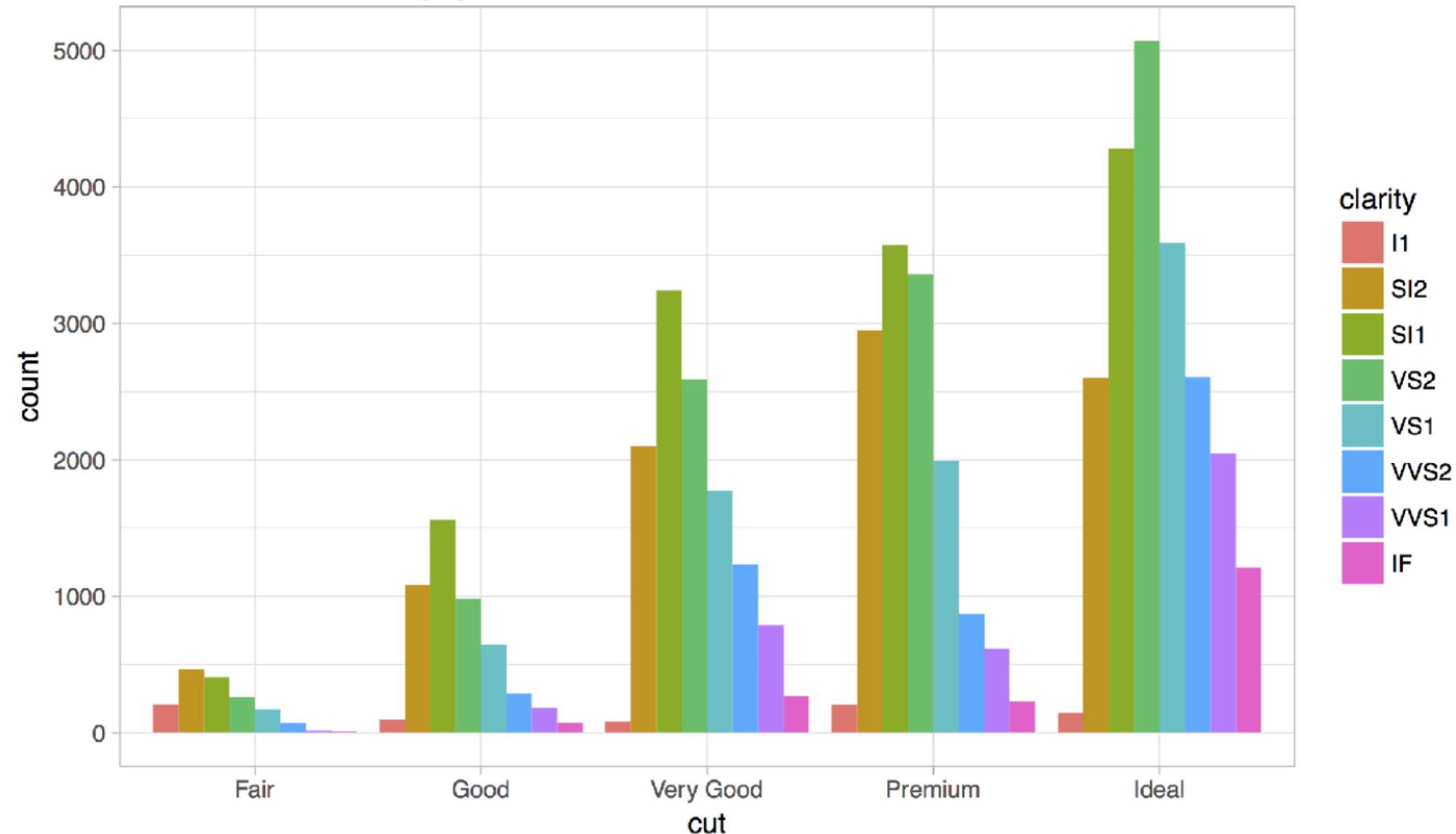
Position Adjustments

How overlapping objects are arranged



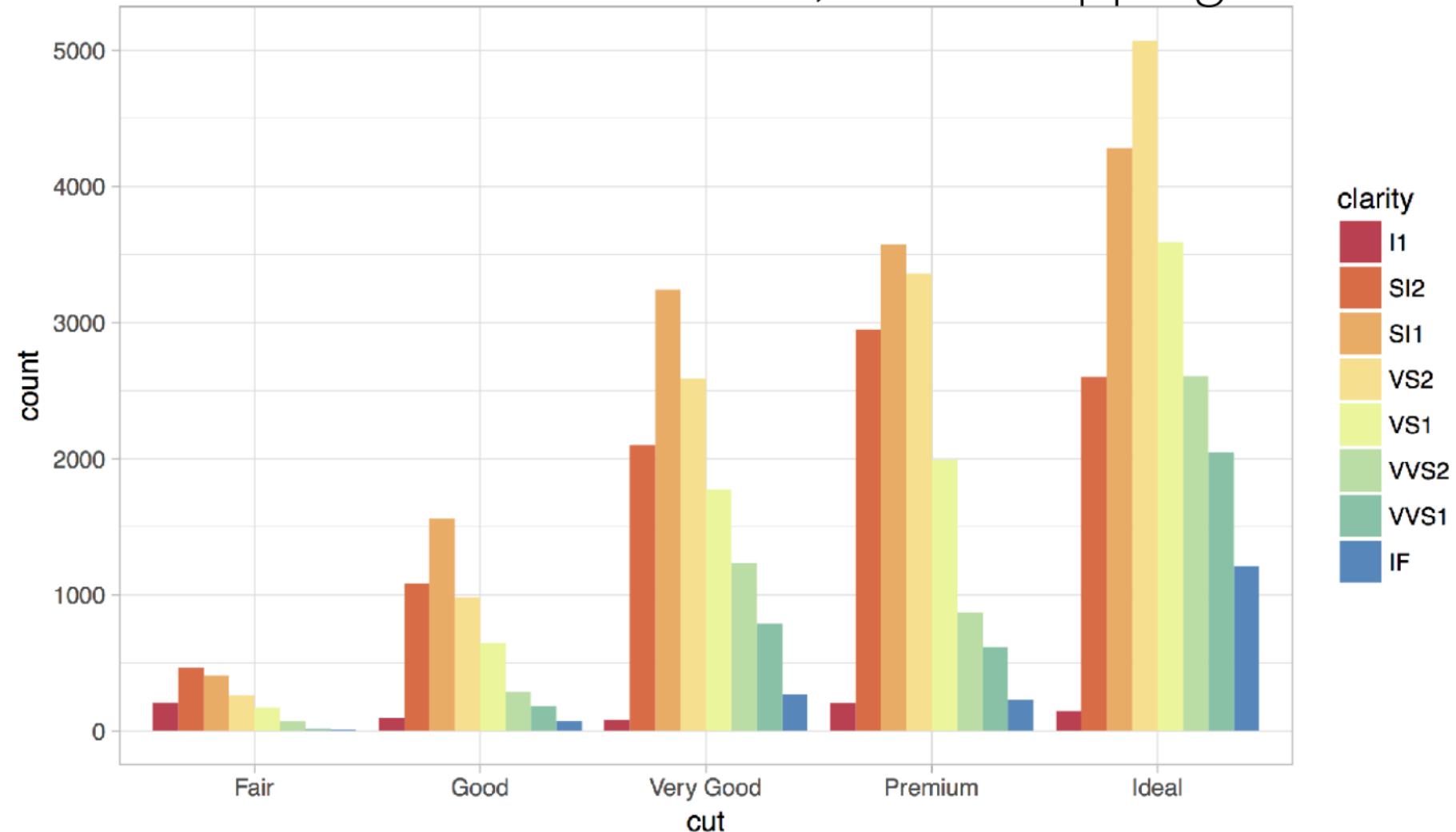
Themes

Visual appearance of non-data elements



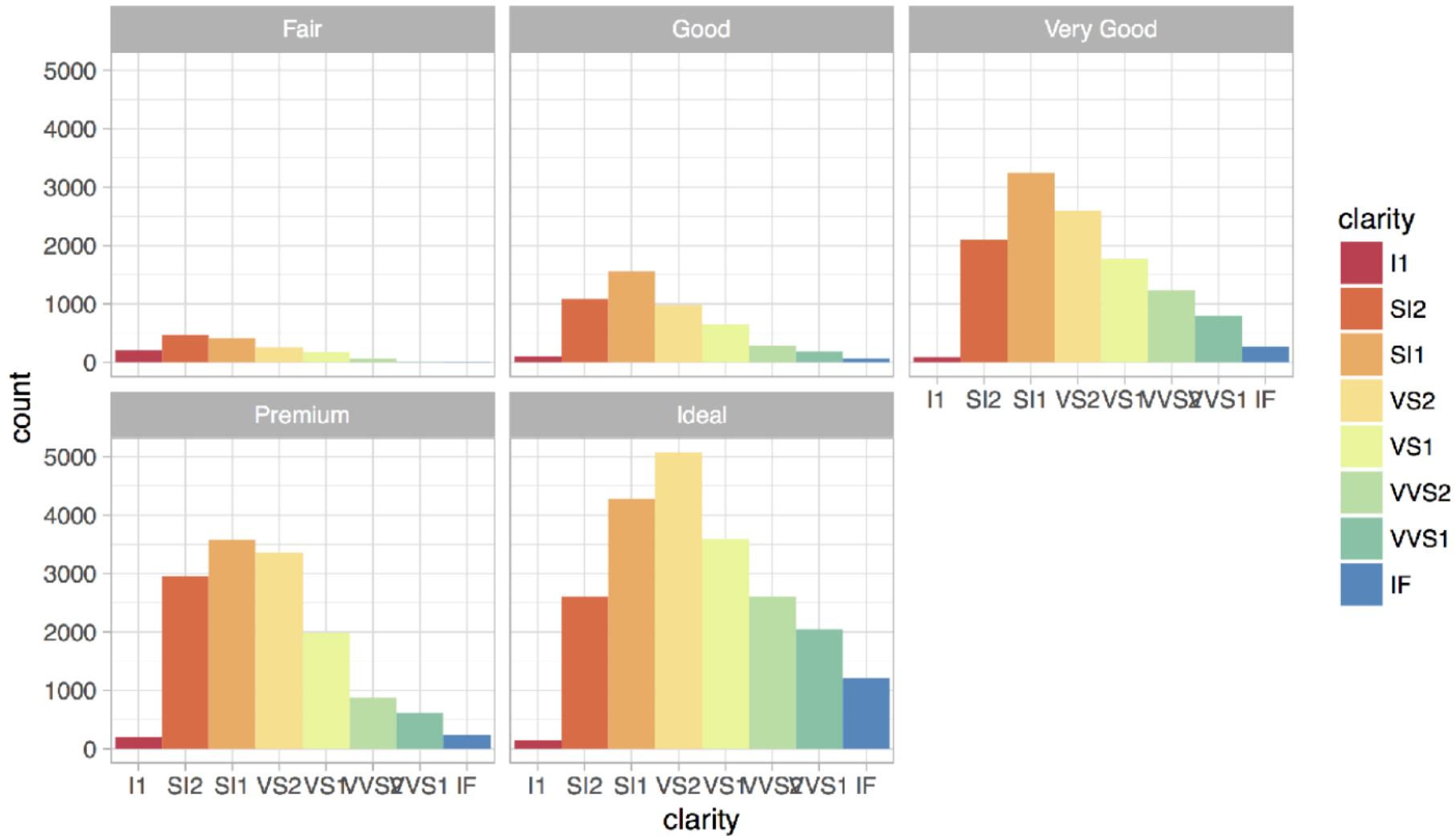
Scales

Customize color scales, other mappings

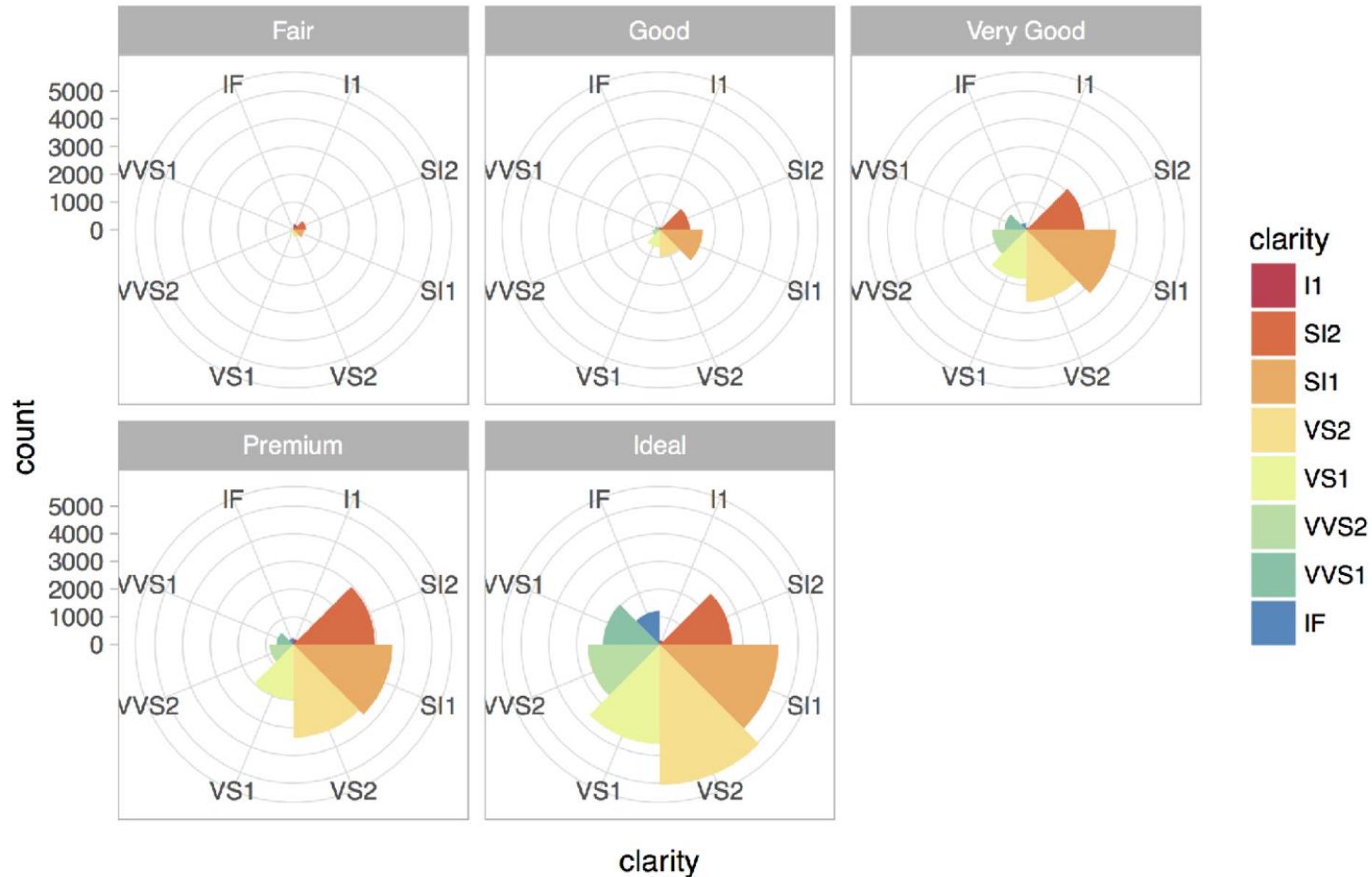


Facets

Subplots that display subsets of the data.



Coordinate systems



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

Visualization with ggplot2 :: CHEAT SHEET

Geoms	
Use a geom function to represent data points, use the geom's aesthetic properties to represent variables. Each function returns a layer.	
GRAPHICAL PRIMITIVES	
a + geom_blank()	(useful for expanding limits)
b + geom_curve(mapping = lat ~ long, y = lat)	b + geom_curve(mapping = lat ~ long, curveto = 2, x = 1, y = 1, check_overlap = TRUE)
c + geom_rect(mapping = x ~ y ~ xend ~ yend)	c + geom_rect(mapping = x ~ y ~ xend ~ yend, alpha = 0.5)
d + geom_path(mapping = "butt", linejoin = "round")	d + geom_path(mapping = "butt", linejoin = "round", lineheight = 1, x = 1, y = 1, xend = 2, yend = 2)
e + geom_polygon(mapping = group ~ group)	e + geom_polygon(mapping = group ~ group, alpha = 0.5)
f + geom_rect(mapping = long ~ ymin ~ ymax ~ xmax ~ xmin)	f + geom_rect(mapping = long ~ ymin ~ ymax ~ xmax ~ xmin, alpha = 0.5)
g + geom_ribbon(mapping = ymin ~ ymax ~ width = 900, ymax = max(y) + 900)	g + geom_ribbon(mapping = ymin ~ ymax ~ width = 900, ymax = max(y) + 900, alpha = 0.5)
LINE SEGMENTS	
common aesthetics: x, y, alpha, color, linetype, size	
b + geom_abline(mapping = intercept ~ slope ~ 1)	b + geom_abline(mapping = intercept ~ slope ~ 1, intercept = 0, slope = 1)
b + geom_hline(mapping = intercept ~ 1)	b + geom_hline(mapping = intercept ~ 1, intercept = 0)
b + geom_segment(mapping = x ~ y ~ xend ~ yend)	b + geom_segment(mapping = x ~ y ~ xend ~ yend, x = 1, y = 1, xend = 2, yend = 2)
b + geom_spoke(mapping = angle ~ r)	b + geom_spoke(mapping = angle ~ r, angle = 11155, r = 1)
ONE VARIABLE	continuous
c + geom_area(mapping = "bin")	c + geom_area(mapping = "bin", fill = "#F0E68C")
c + geom_dotplot(mapping = "dotplot", binwidth = 1)	c + geom_dotplot(mapping = "dotplot", binwidth = 1, center = "top", fill = "#F0E68C")
c + geom_freqpoly(mapping = "freqpoly", binwidth = 1)	c + geom_freqpoly(mapping = "freqpoly", binwidth = 1, fill = "#F0E68C")
c + geom_histogram(mapping = "histogram", binwidth = 5)	c + geom_histogram(mapping = "histogram", binwidth = 5, fill = "#F0E68C")
c + geom_dotplot(mapping = "dotplot", binwidth = 1)	c + geom_dotplot(mapping = "dotplot", binwidth = 1, center = "top", fill = "#F0E68C")
c + geom_hex(mapping = "hex", binwidth = 1)	c + geom_hex(mapping = "hex", binwidth = 1, fill = "#F0E68C")
c + geom_kde(mapping = "kde", fill = "#F0E68C")	c + geom_kde(mapping = "kde", fill = "#F0E68C")
c + geom_qq(mapping = "qq", sample = "hwy")	c + geom_qq(mapping = "qq", sample = "hwy", fill = "#F0E68C")
discrete x, continuous y	
f + geom_bar(mapping = "bar")	f + geom_bar(mapping = "bar", fill = "#F0E68C")
f + geom_boxplot(mapping = "boxplot", lower = "lower", middle = "middle", upper = "upper")	f + geom_boxplot(mapping = "boxplot", lower = "lower", middle = "middle", upper = "upper", fill = "#F0E68C")
f + geom_dotplot(mapping = "dotplot", fill = "#F0E68C")	f + geom_dotplot(mapping = "dotplot", fill = "#F0E68C")
f + geom_hex(mapping = "hex", fill = "#F0E68C")	f + geom_hex(mapping = "hex", fill = "#F0E68C")
f + geom_violin(mapping = "violin", scale = "area")	f + geom_violin(mapping = "violin", scale = "area", fill = "#F0E68C")
discrete x, discrete y	
g + geom_count(mapping = "count")	g + geom_count(mapping = "count", fill = "#F0E68C")
maps	
map + geom_map(mapping = "map", id = state, map = map)	map + geom_map(mapping = "map", id = state, map = map, fill = "#F0E68C")
map + geom_sf(mapping = "sf", id = state, map = map)	map + geom_sf(mapping = "sf", id = state, map = map, fill = "#F0E68C")
THREE VARIABLES	
sealsSz <- with(seals, sqrt(delta_long^2 + delta_lat^2))	sealsSz <- with(seals, sqrt(delta_long^2 + delta_lat^2))
i + geom_contour(mapping = "z ~ x ~ y")	i + geom_contour(mapping = "z ~ x ~ y", fill = "#F0E68C")
j + geom_raster(mapping = "fill ~ x ~ y", interpolate = FALSE)	j + geom_raster(mapping = "fill ~ x ~ y", fill = "#F0E68C")
k + geom_tile(mapping = "fill ~ x ~ y")	k + geom_tile(mapping = "fill ~ x ~ y", fill = "#F0E68C")



RStudio® is a trademark of RStudio, Inc. • CC BY RStudio • info@rstudio.com • 844-448-1232 • rstudio.com • Learn more at <http://ggplot2.tidyverse.org>

ggplot2.tidyverse.org

The screenshot shows a web browser window displaying the ggplot2.tidyverse.org website. The title bar reads "Create Elegant Data Visualisation" and "Garrett". The address bar shows the URL "ggplot2.tidyverse.org". The page header features the ggplot2 logo and the text "part of the tidyverse". Navigation links include "Reference", "Articles", "News", and a user icon.

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()
```

A scatter plot is shown with the x-axis labeled "displ" and the y-axis labeled "hwy". The legend indicates "class" with a red dot and "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

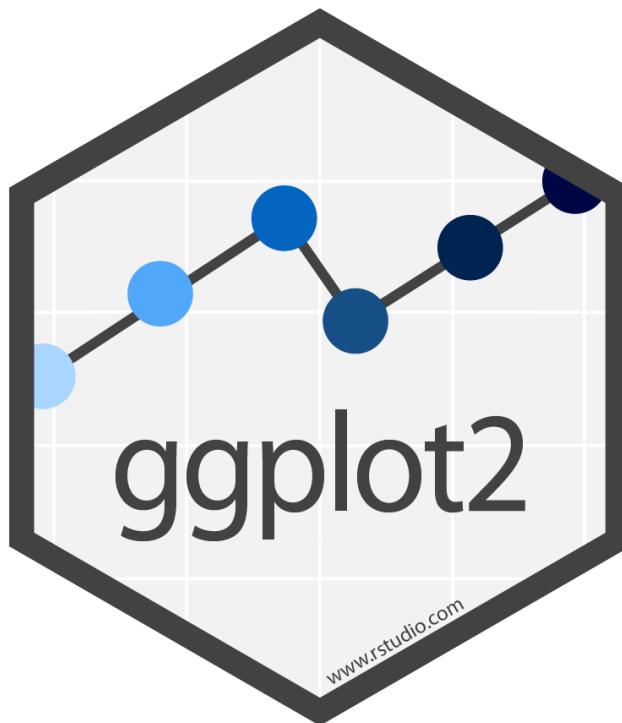
[GPL-2](#) | file [LICENSE](#)

Developers

Hadley Wickham
Author, maintainer



Visualising Data with



Adapted from “Explore the Tidyverse” CC by Hadley Wickham