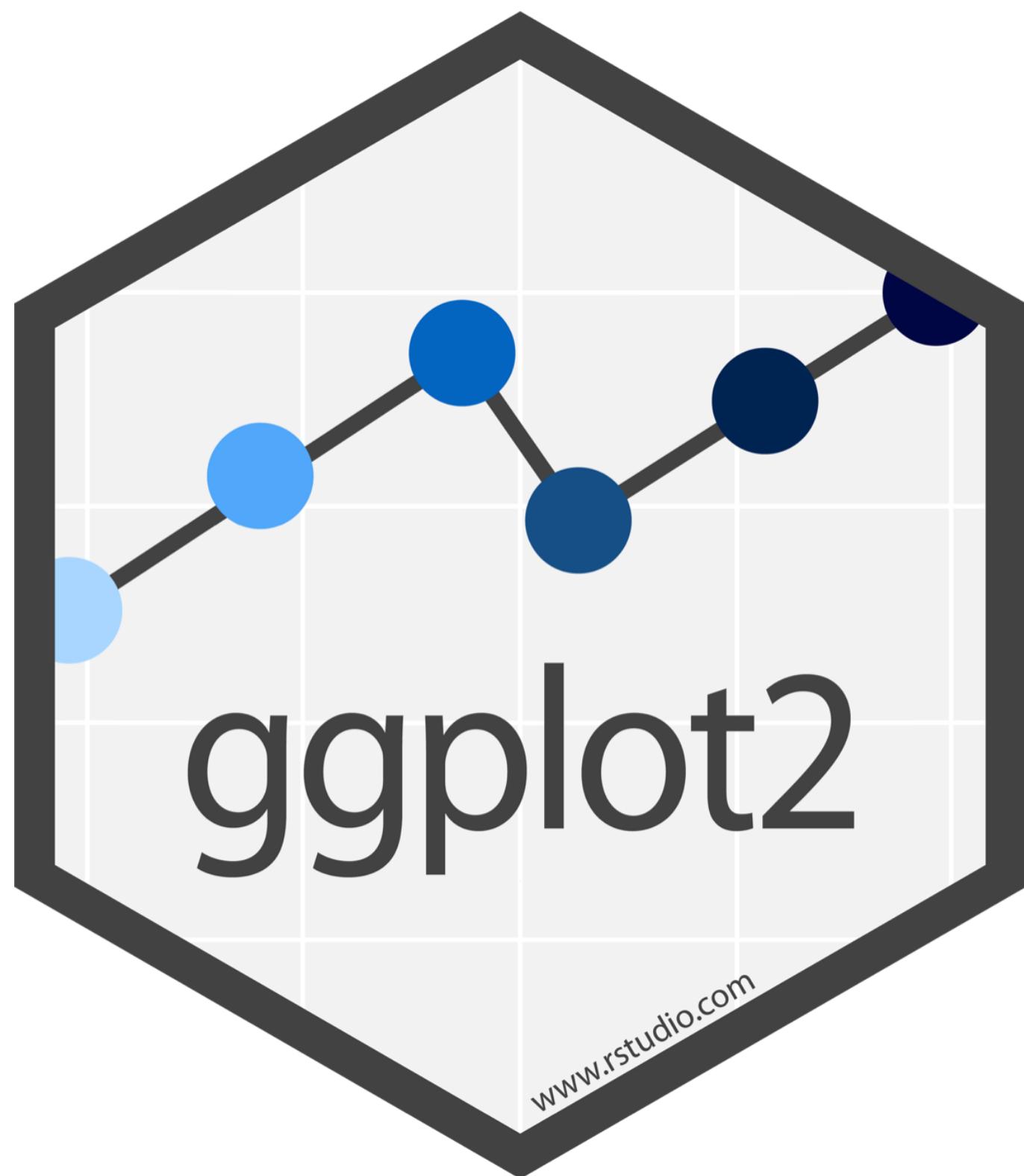


Visualize Data with



Setup

The setup chunk is always run once before anything else

```
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))  
38  
87:1 | Chunk 11 | R Markdown
```

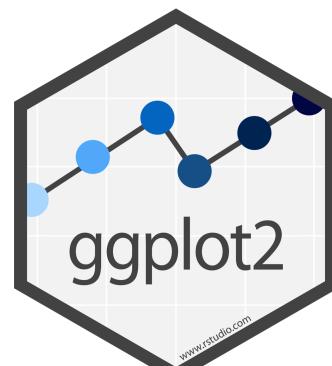
chunk labels are optional,
the setup label is special

income

Reported employment income of
graduates (2010, 2011) two years and
five years after graduation

income

<http://github.com/paleolimbot/unienrollment>



Quiz

Confer with your group.

What relationship do you expect to see between how much a graduate makes after 2 years (income_2yr) and 5 years (income_5yr)?

No peeking ahead!



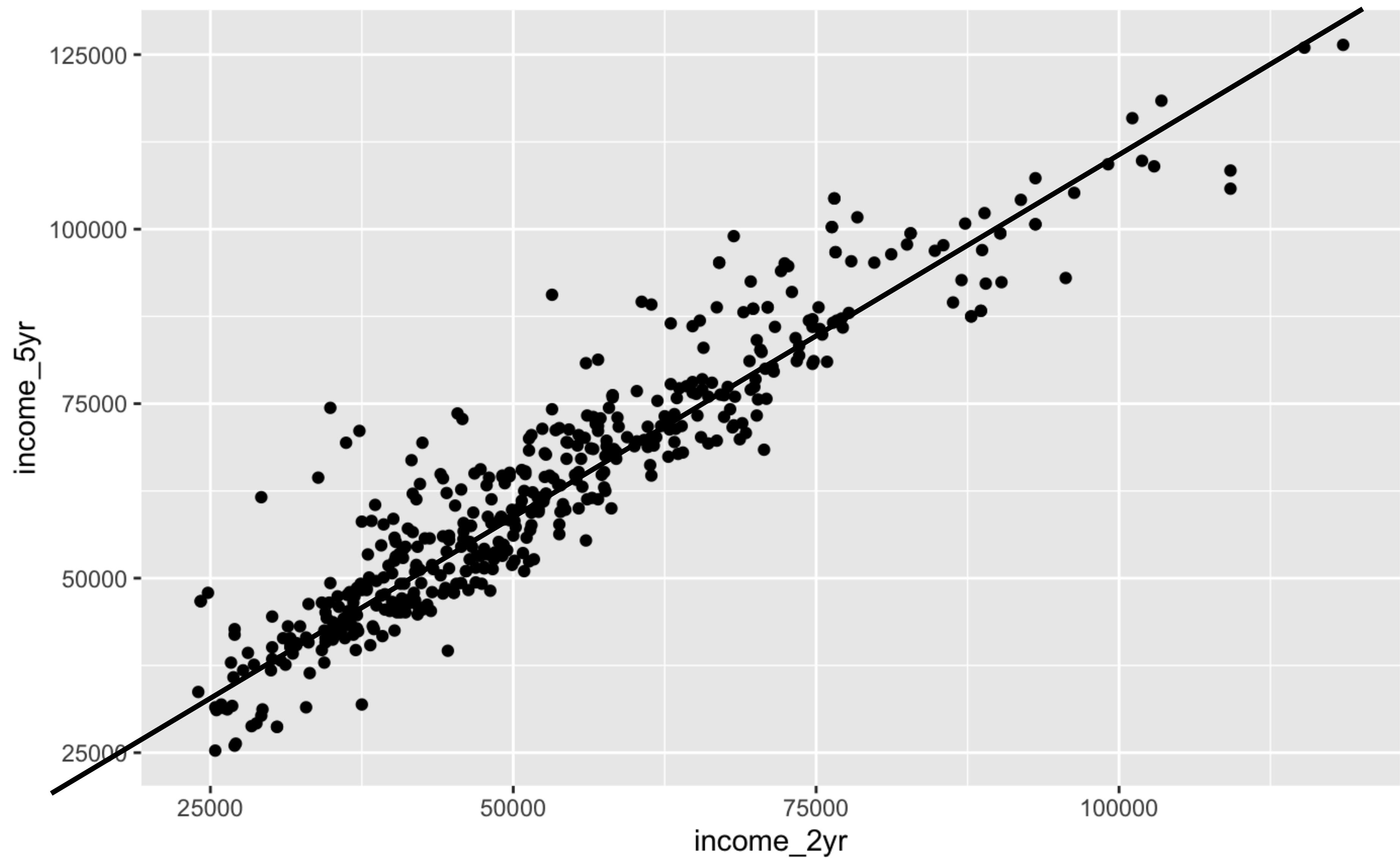
Your Turn 1

Run this code in your notebook to make a graph.

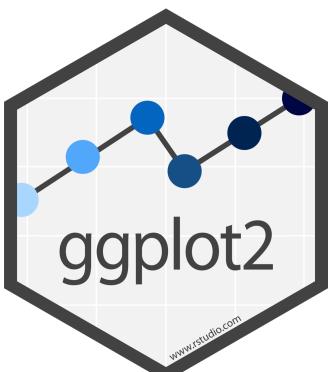
Pay strict attention to spelling, capitalization, and parentheses!

```
ggplot(data = income) +  
  geom_point(mapping = aes(x = income_5yr, y = income_2yr))
```



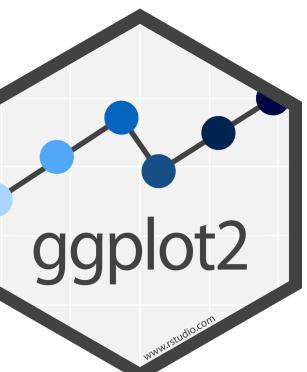


```
ggplot(data = income) +  
  geom_point(mapping = aes(x = income_2yr, y = income_5yr))
```



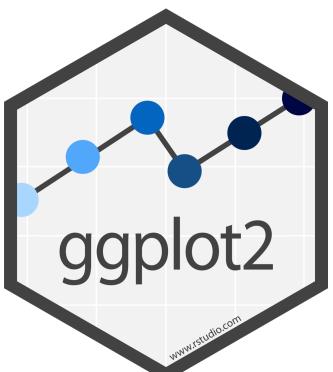
1. "Initialize" a plot with `ggplot()`
2. Add layers with `geom_` functions

```
ggplot(data = income) +  
  geom_point(mapping = aes(x = income_2yr, y = income_5yr))
```



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

```
ggplot(data = income) +  
  geom_point(mapping = aes(x = income_2yr, y = income_5yr))
```



data

+ before new line

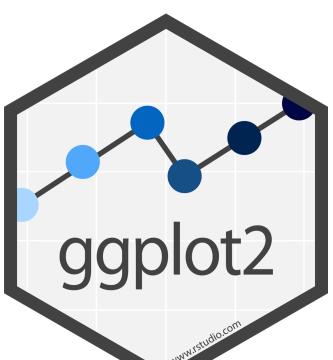
```
ggplot(data = income) +  
  geom_point(mapping = aes(x = income_2yr, y = income_5yr))
```

type of layer

aes()

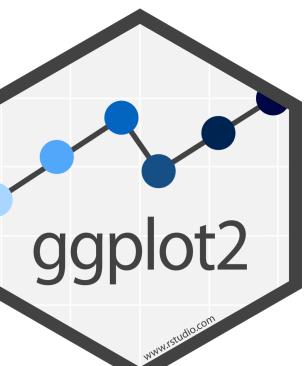
x variable

y variable



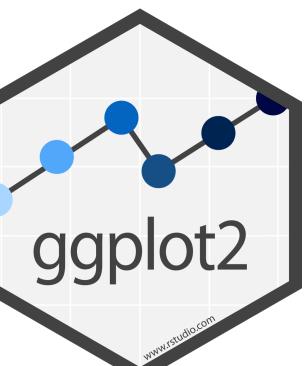
A template

```
ggplot(data = <DATA>) +  
<GEOM_FUNCTION>(mapping = aes(<MAPPINGS>))  
  
geom_point(mapping = aes(x = income_2yr, y = income_5yr))
```



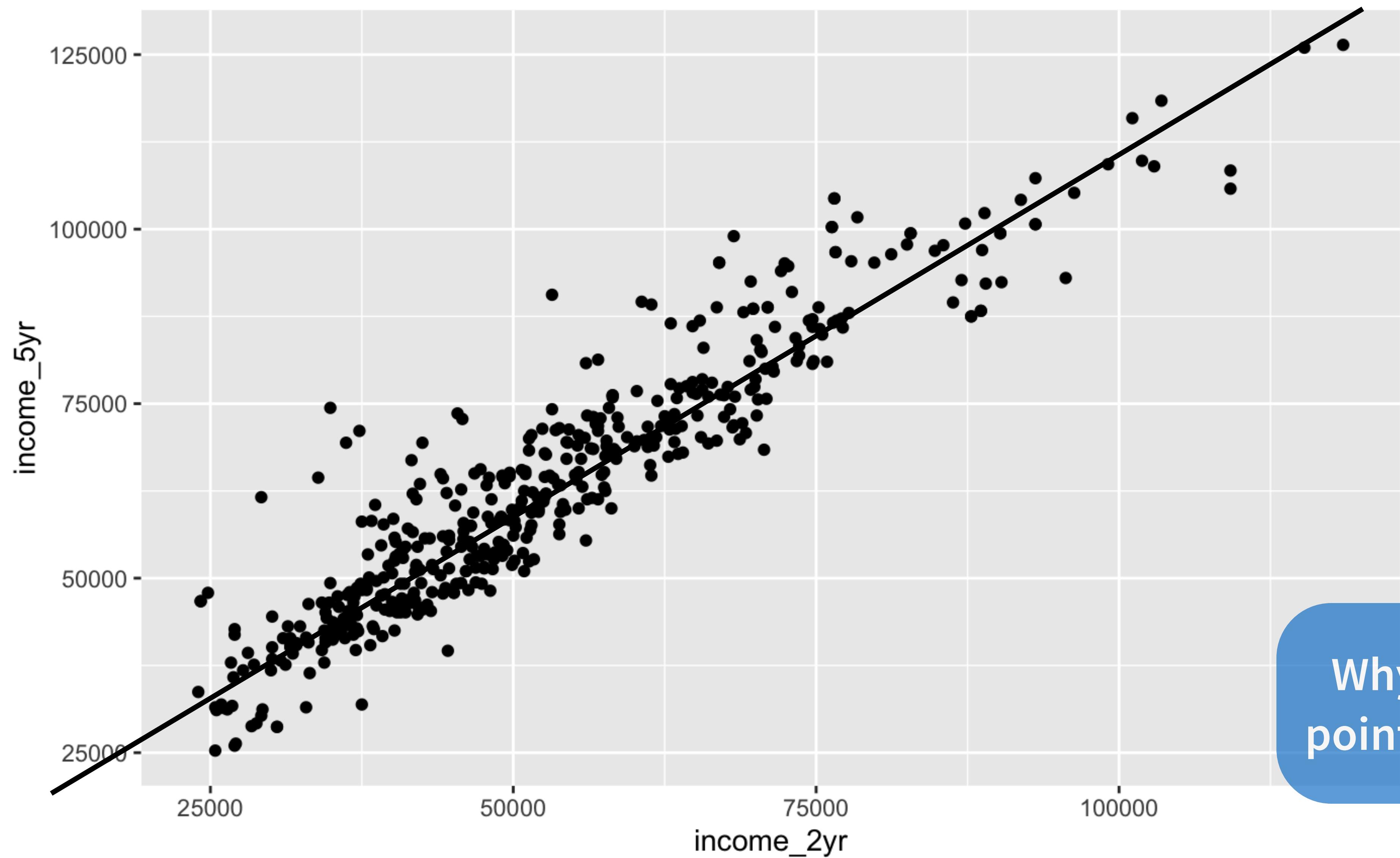
A template

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



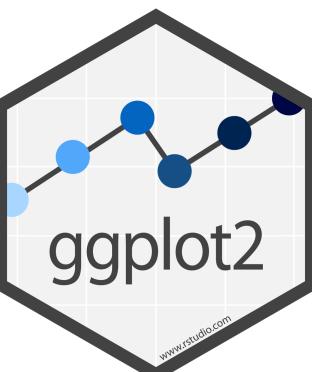
Mappings

R

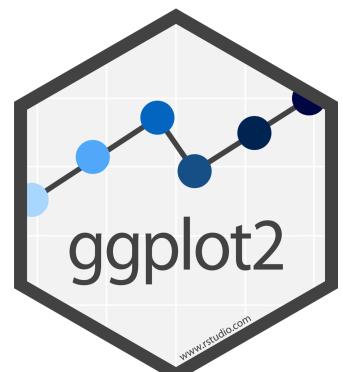
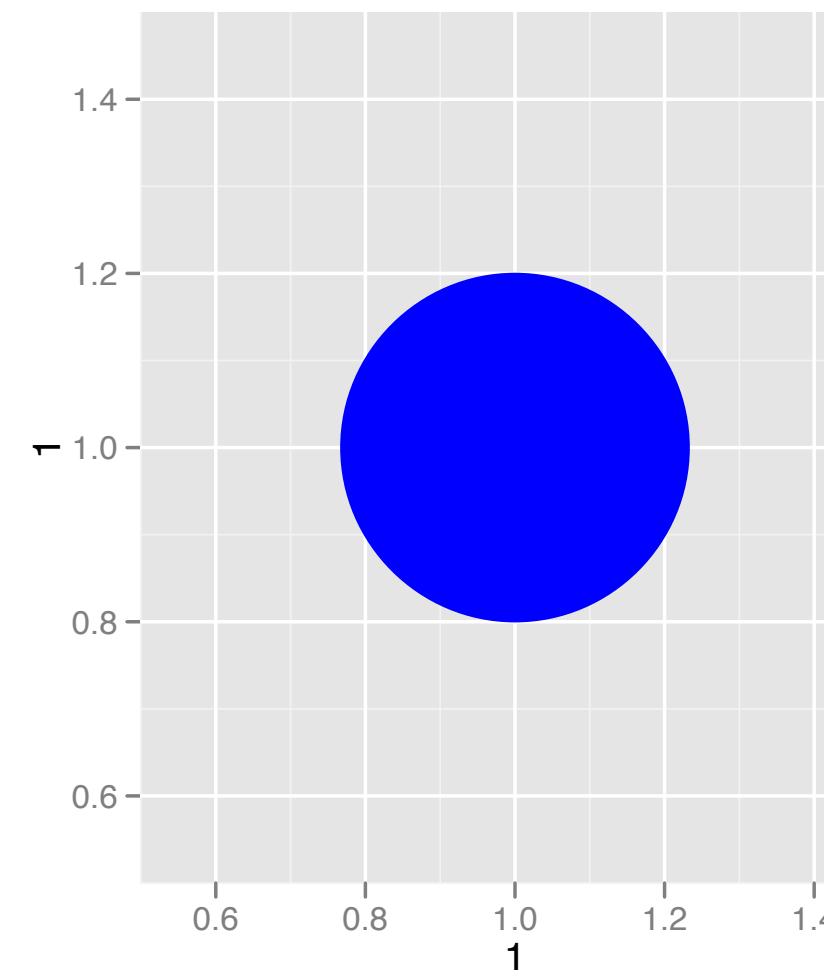
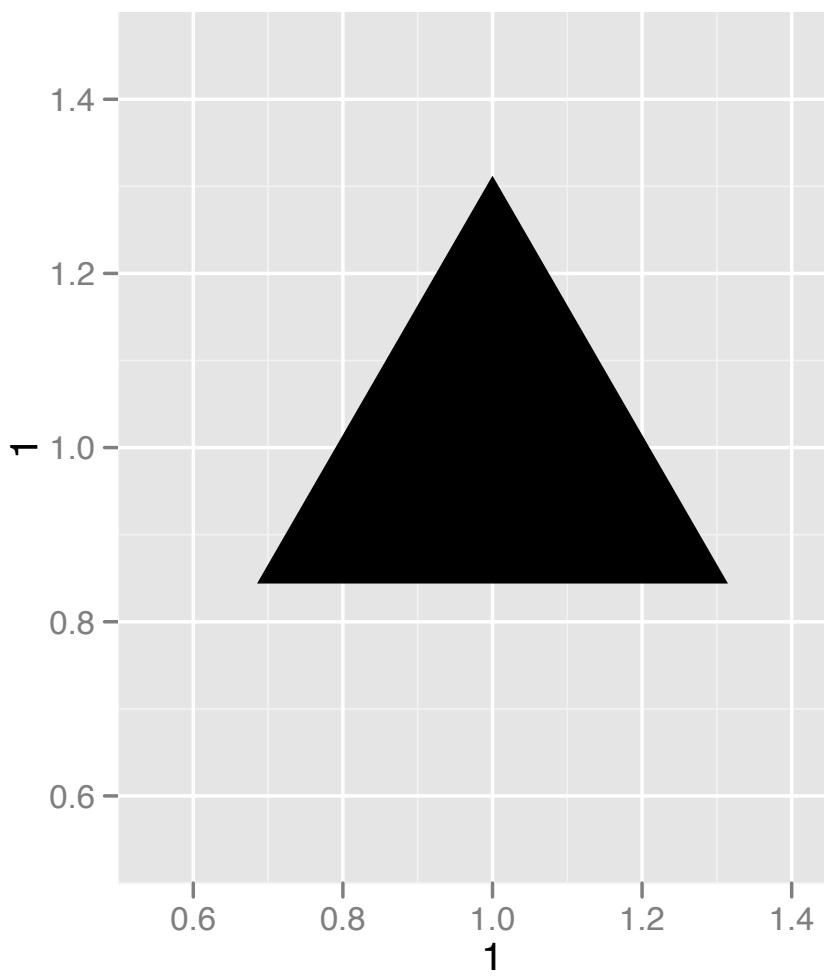
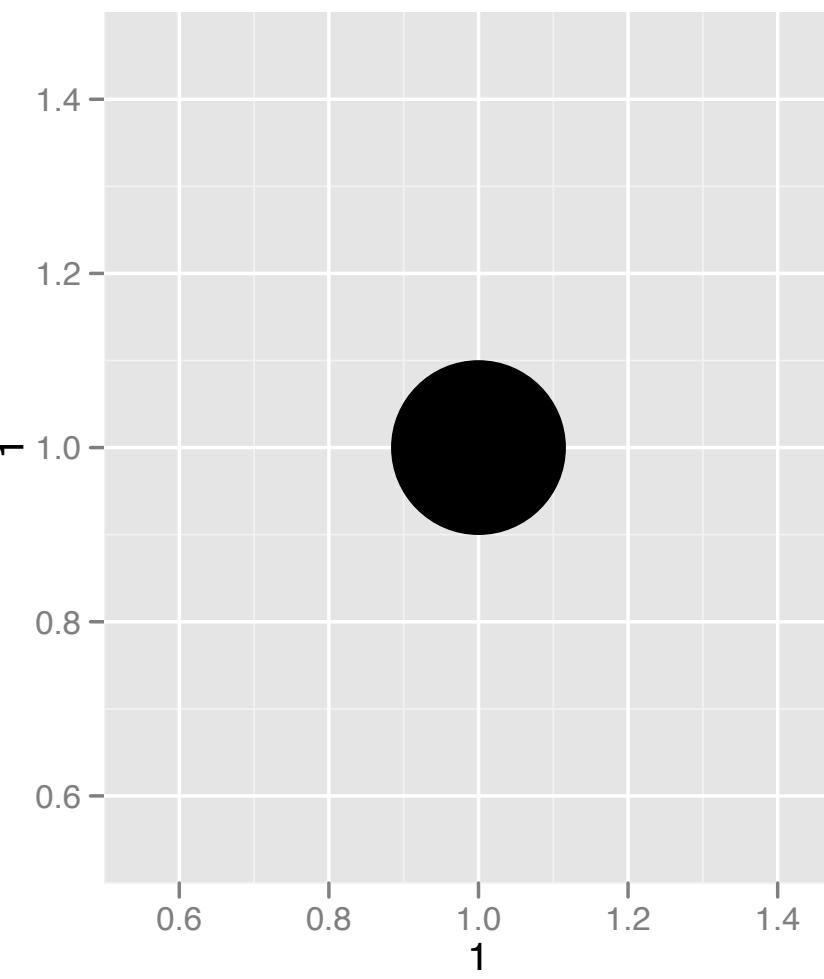
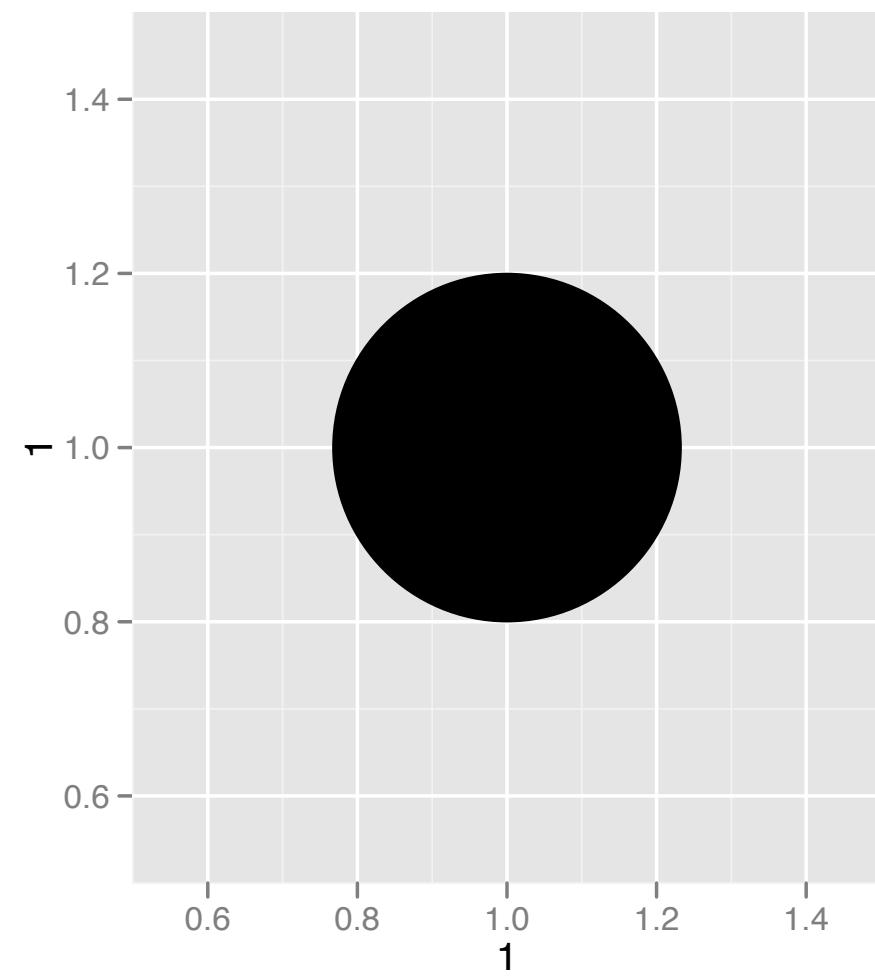


Why are some
points different?

```
ggplot(data = income) +  
  geom_point(mapping = aes(x = income_2yr, y = income_5yr))
```



Aesthetics



Visual Space

color

Red

Brown

Green

Blue

Violet

Data Space

degree_type

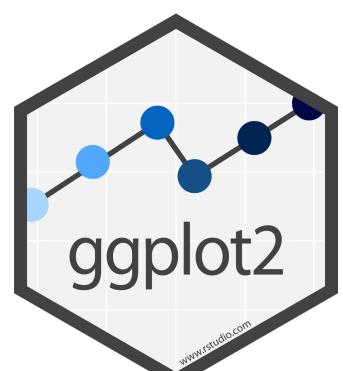
Associate

Doctoral

Master's

Professional

Undergraduate

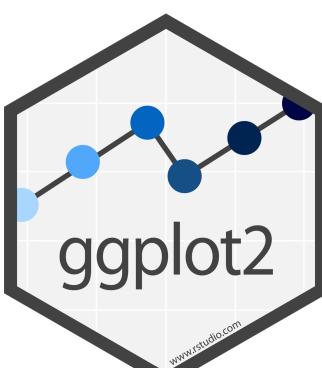


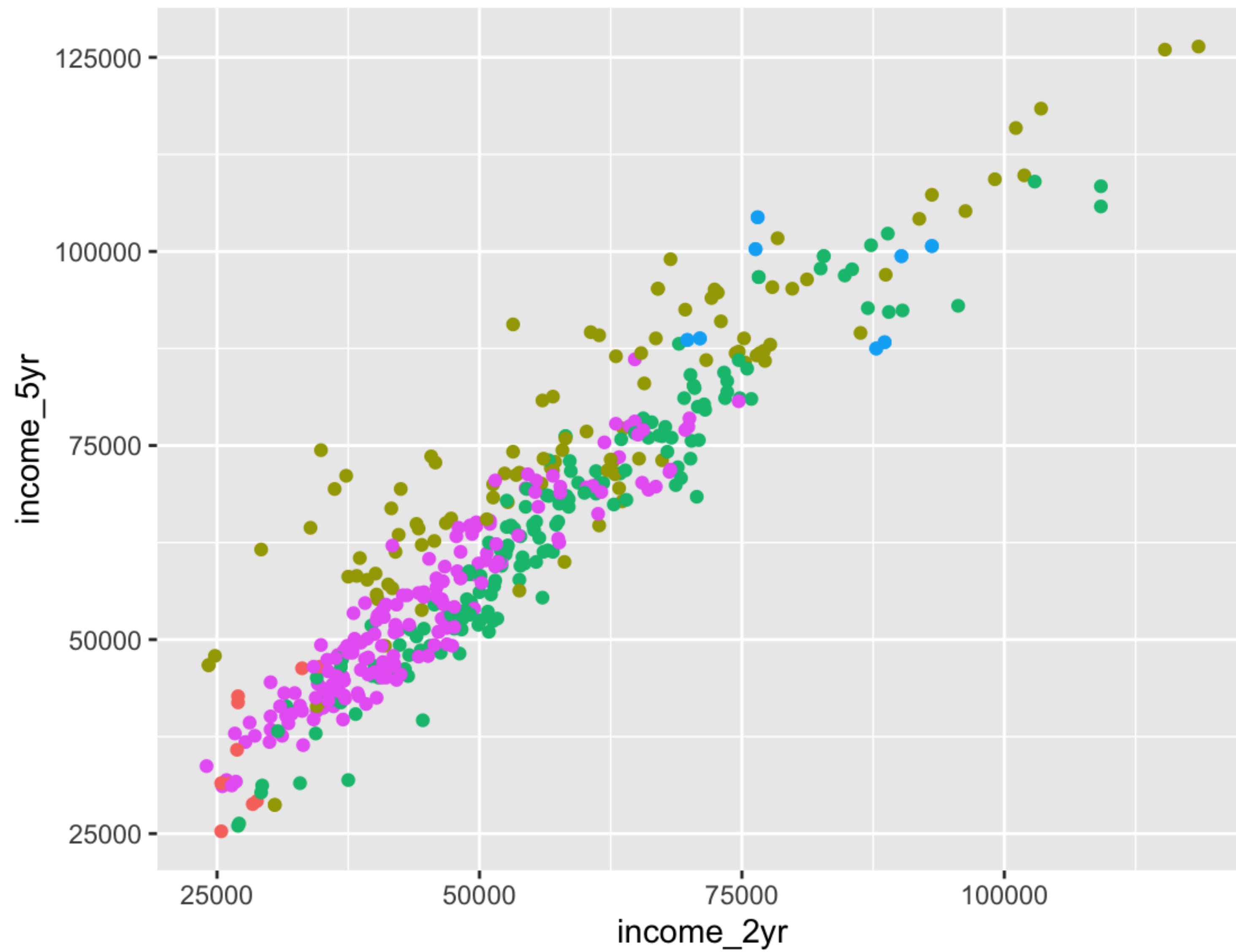
Aesthetics

aesthetic
property

Variable to
map it to

```
ggplot(income) + geom_point(aes(x = income_2yr, y = income_5yr, color = degree_type))  
ggplot(income) + geom_point(aes(x = income_2yr, y = income_5yr, size = degree_type))  
ggplot(income) + geom_point(aes(x = income_2yr, y = income_5yr, shape = degree_type))  
ggplot(income) + geom_point(aes(x = income_2yr, y = income_5yr, alpha = degree_type))
```

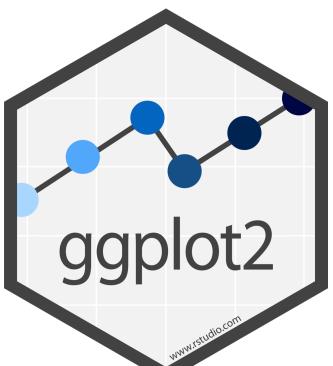




Legend added automatically

degree_type
Associate
Doctoral
Master's
Professional
Undergraduate

```
ggplot(data = income) +  
  geom_point(mapping = aes(x = income_2yr, y = income_5yr, color = degree_type))
```



Your Turn 2

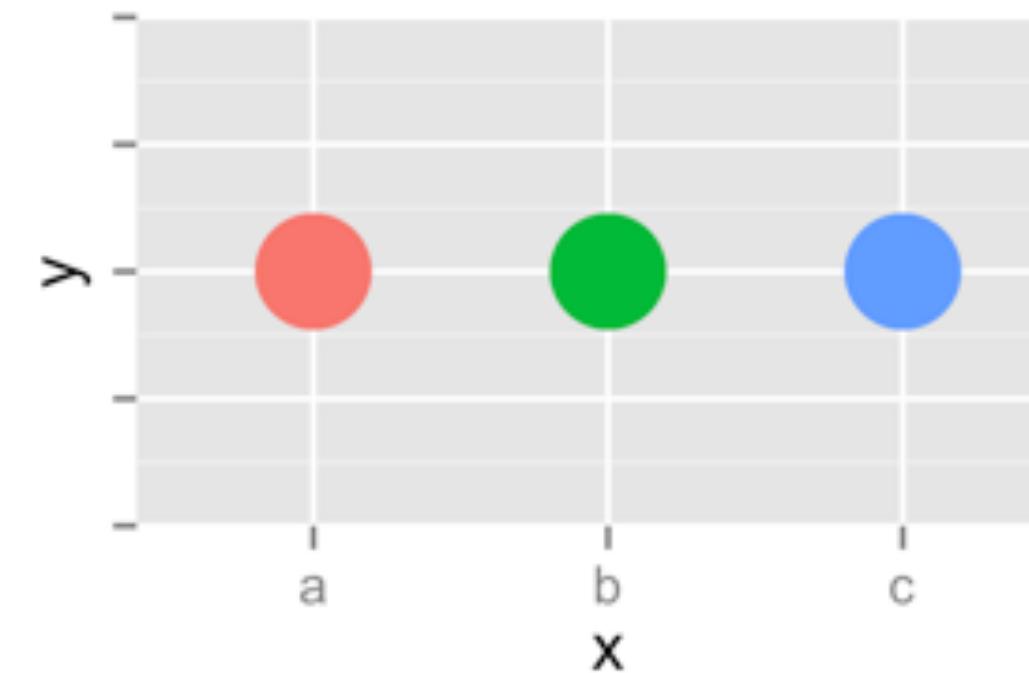
In the next chunk, add color, size, alpha, and shape aesthetics to your graph. Experiment.

Do different things happen when you map aesthetics to discrete and continuous variables?

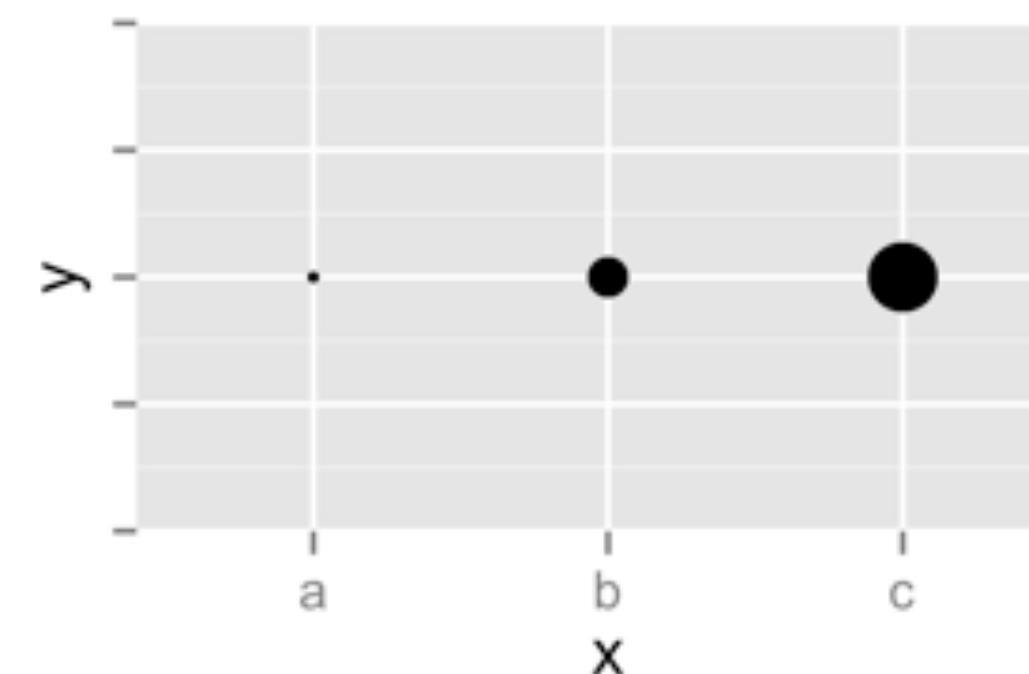
What happens when you use more than one aesthetic?



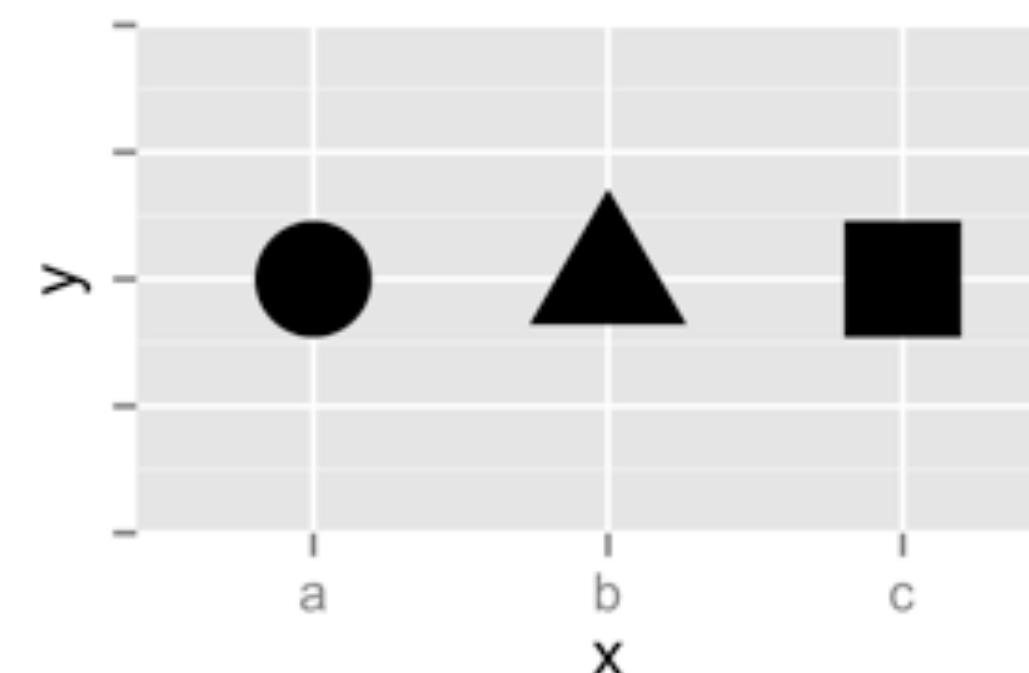
Color



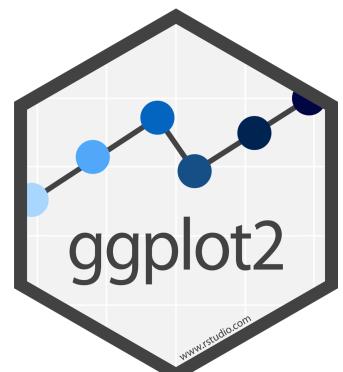
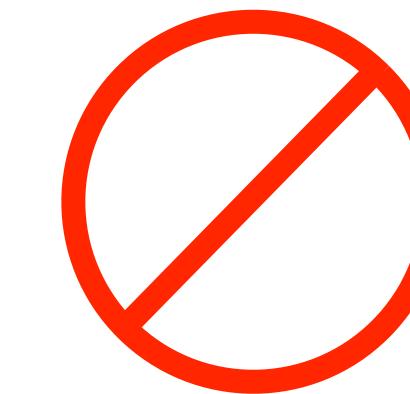
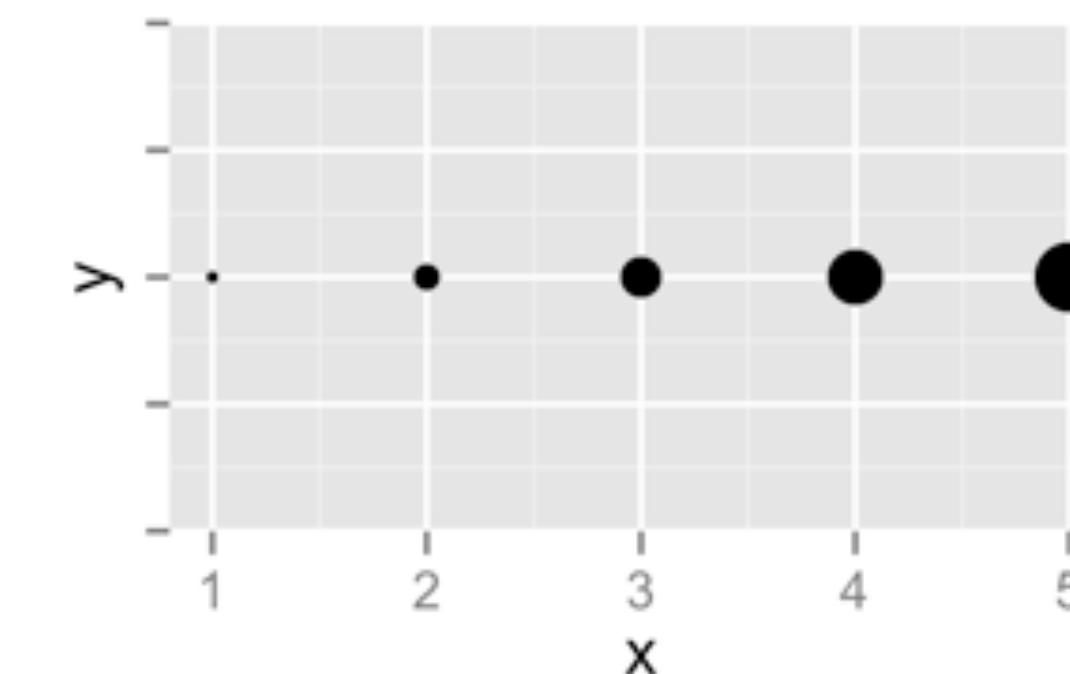
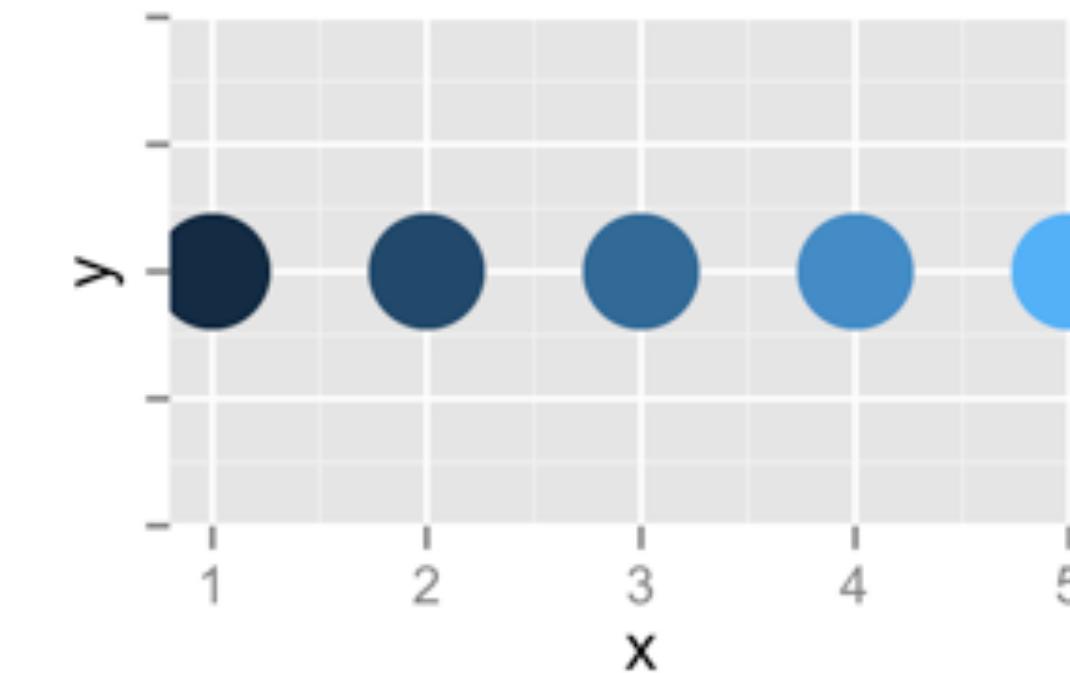
Size



Shape



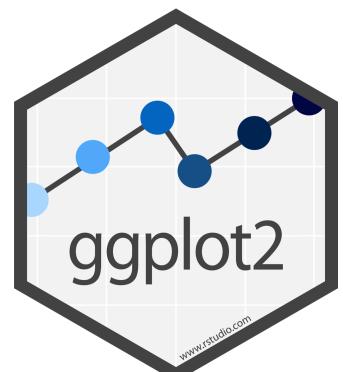
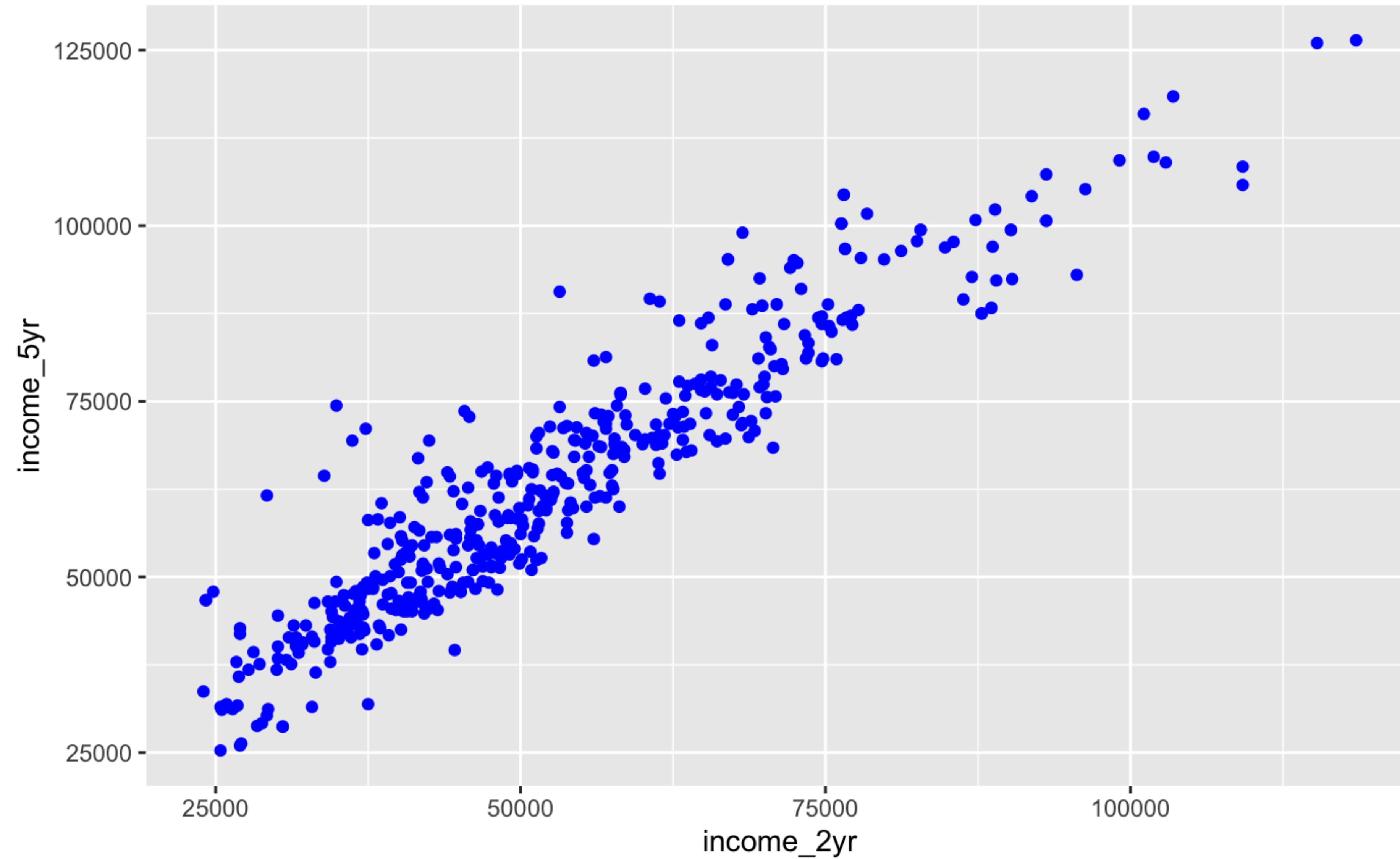
Continuous

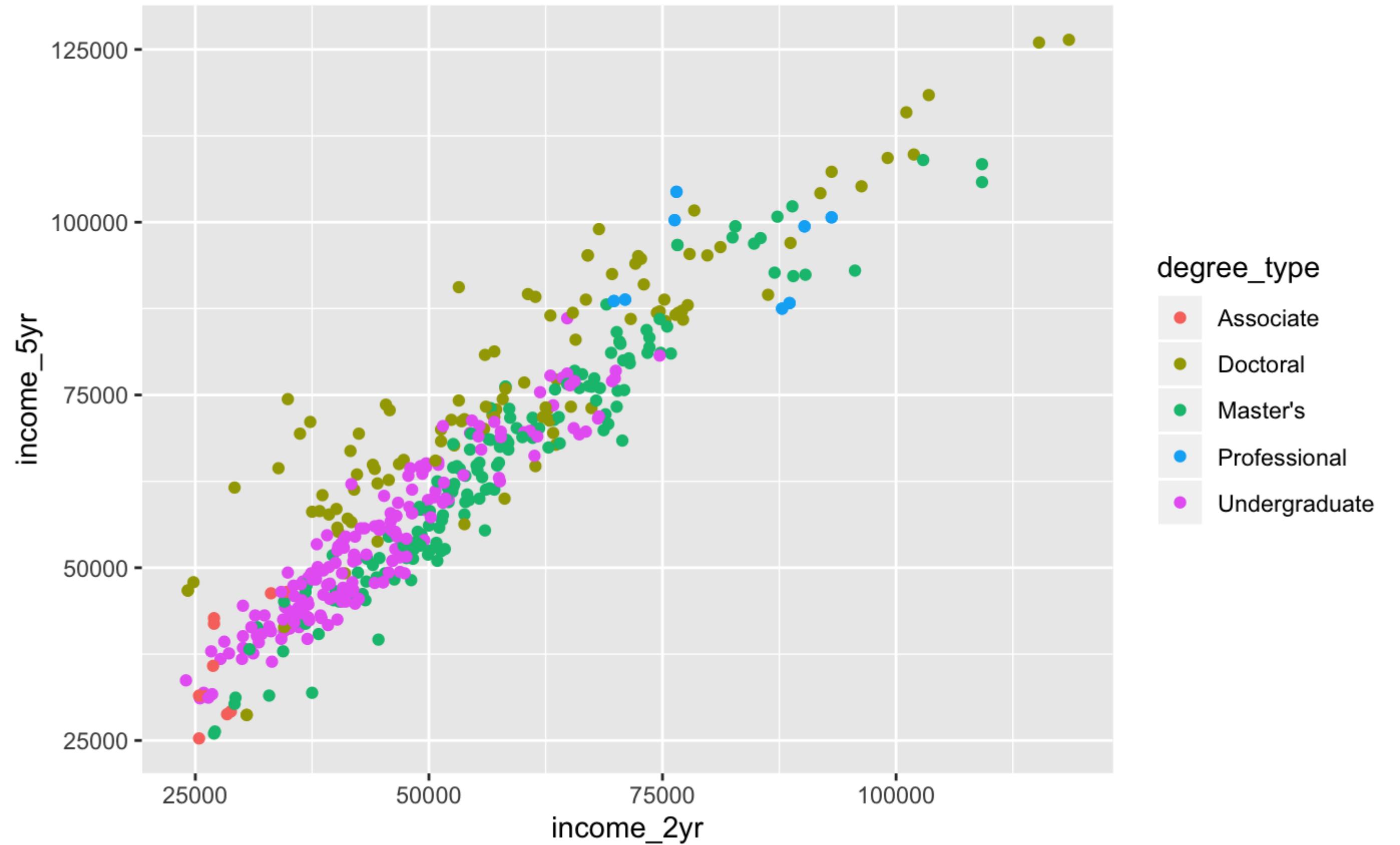


set vs. map

A large, semi-transparent watermark of the R logo is positioned in the bottom right corner. The logo consists of a circular emblem with the letters "R" inside.

How would you make this plot?

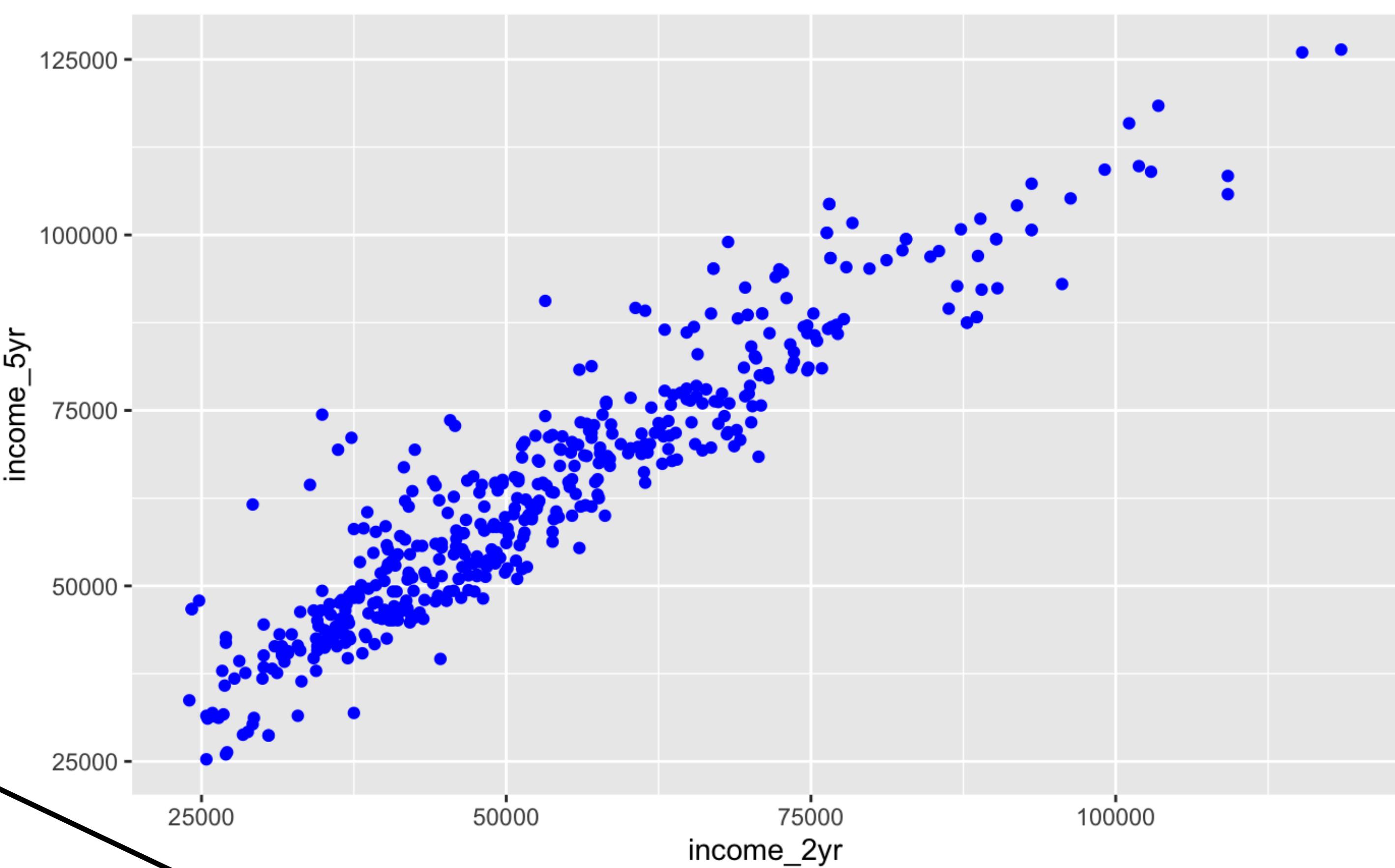




Inside of aes(): maps an aesthetic to a variable

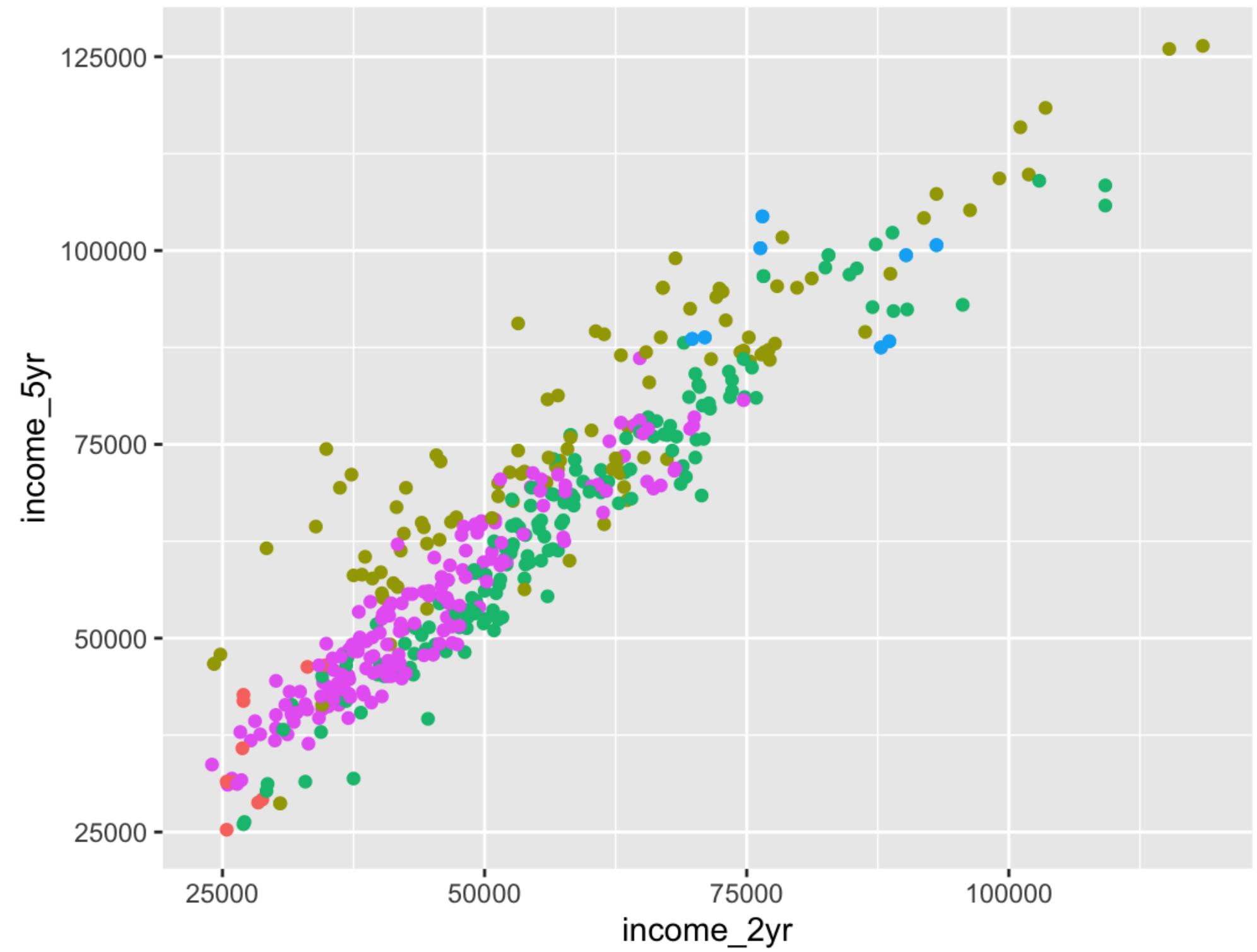
```
ggplot(income) + geom_point(aes(x = income_2yr, y = income_5yr, color = degree_type))
```

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



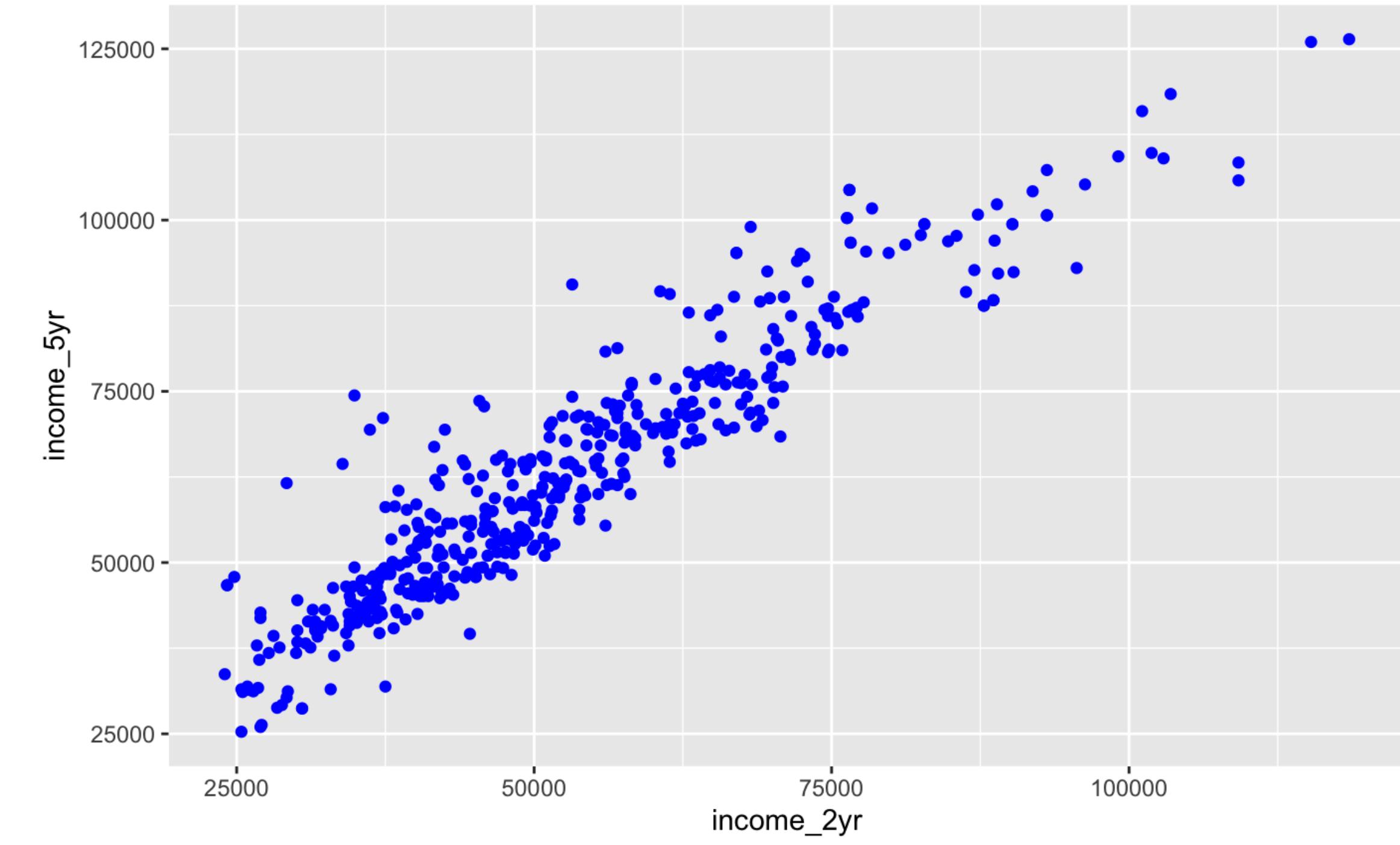
```
ggplot(income) + geom_point(aes(x = income_2yr, y = income_5yr, color = degree_type))
```

```
ggplot(income) + geom_point(aes(x = income_2yr, y = income_5yr), color = "blue")
```



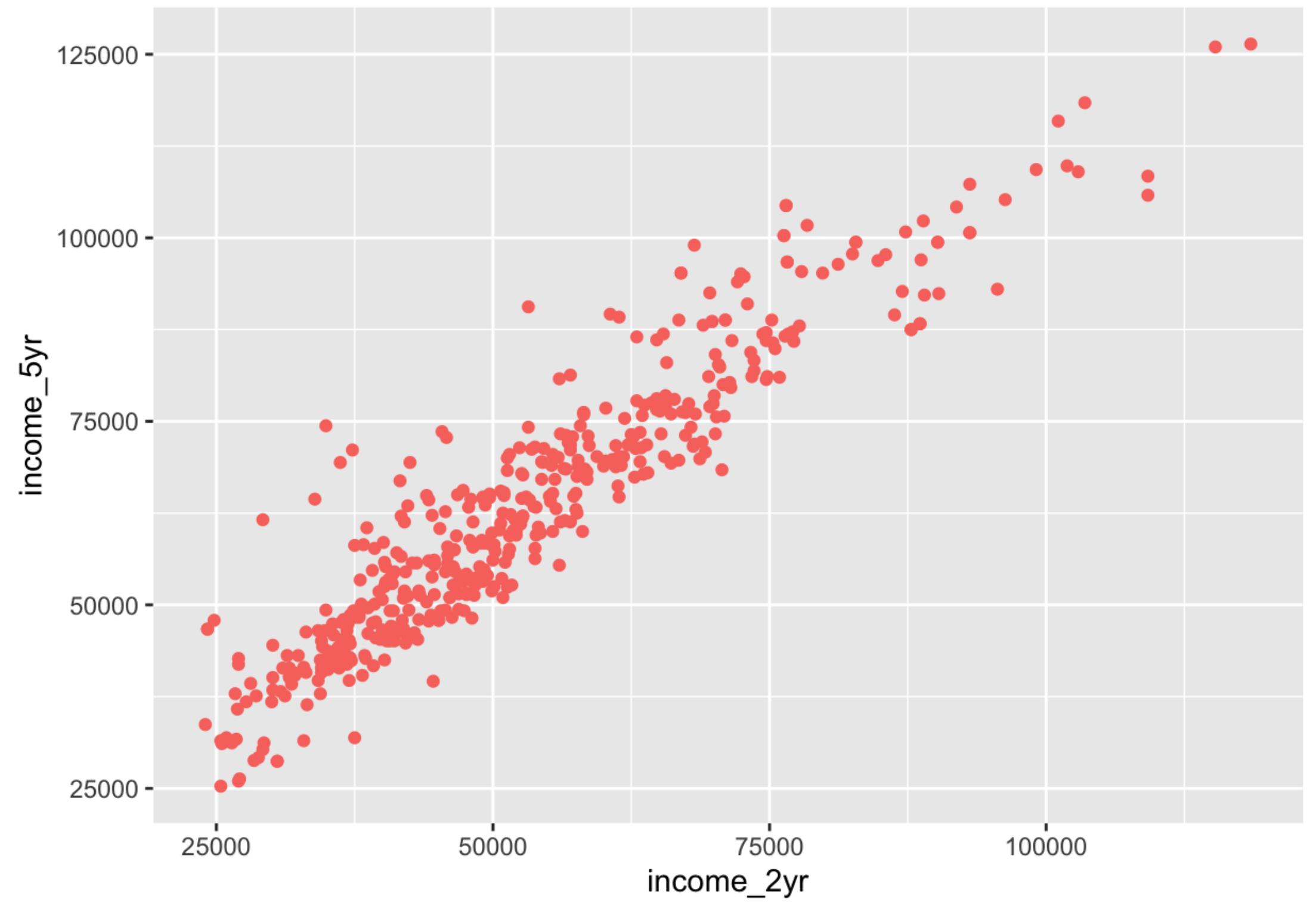
degree_type

- Associate
- Doctoral
- Master's
- Professional
- Undergraduate

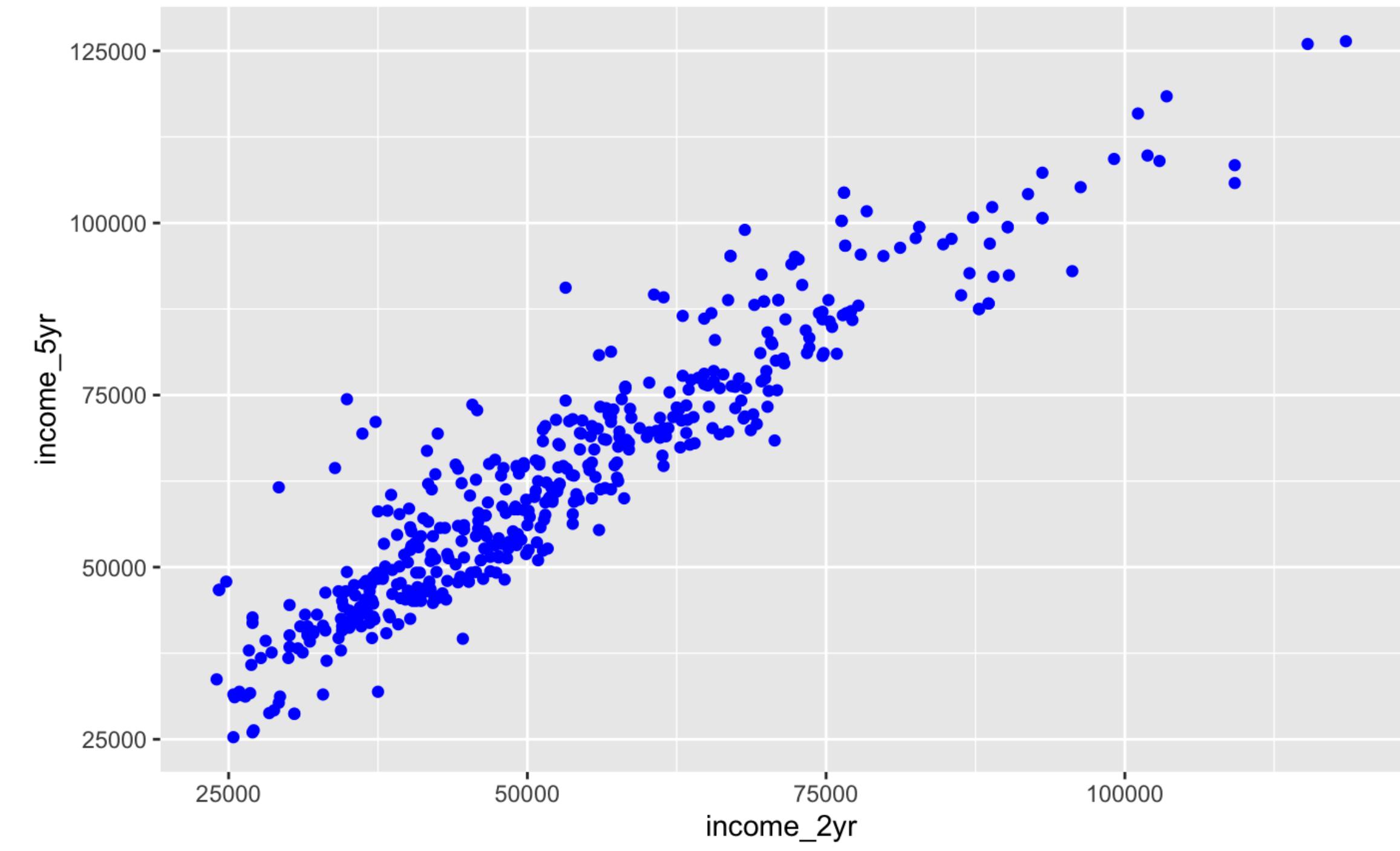


```
ggplot(income) + geom_point(aes(x = income_2yr, y = income_5yr, color = degree_type))
```

```
ggplot(income) + geom_point(aes(x = income_2yr, y = income_5yr), color = "blue")
```



colour
● blue



```
ggplot(income) + geom_point(aes(x = income_2yr, y = income_5yr), color = "blue")
```

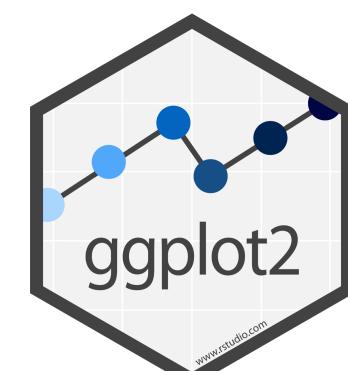
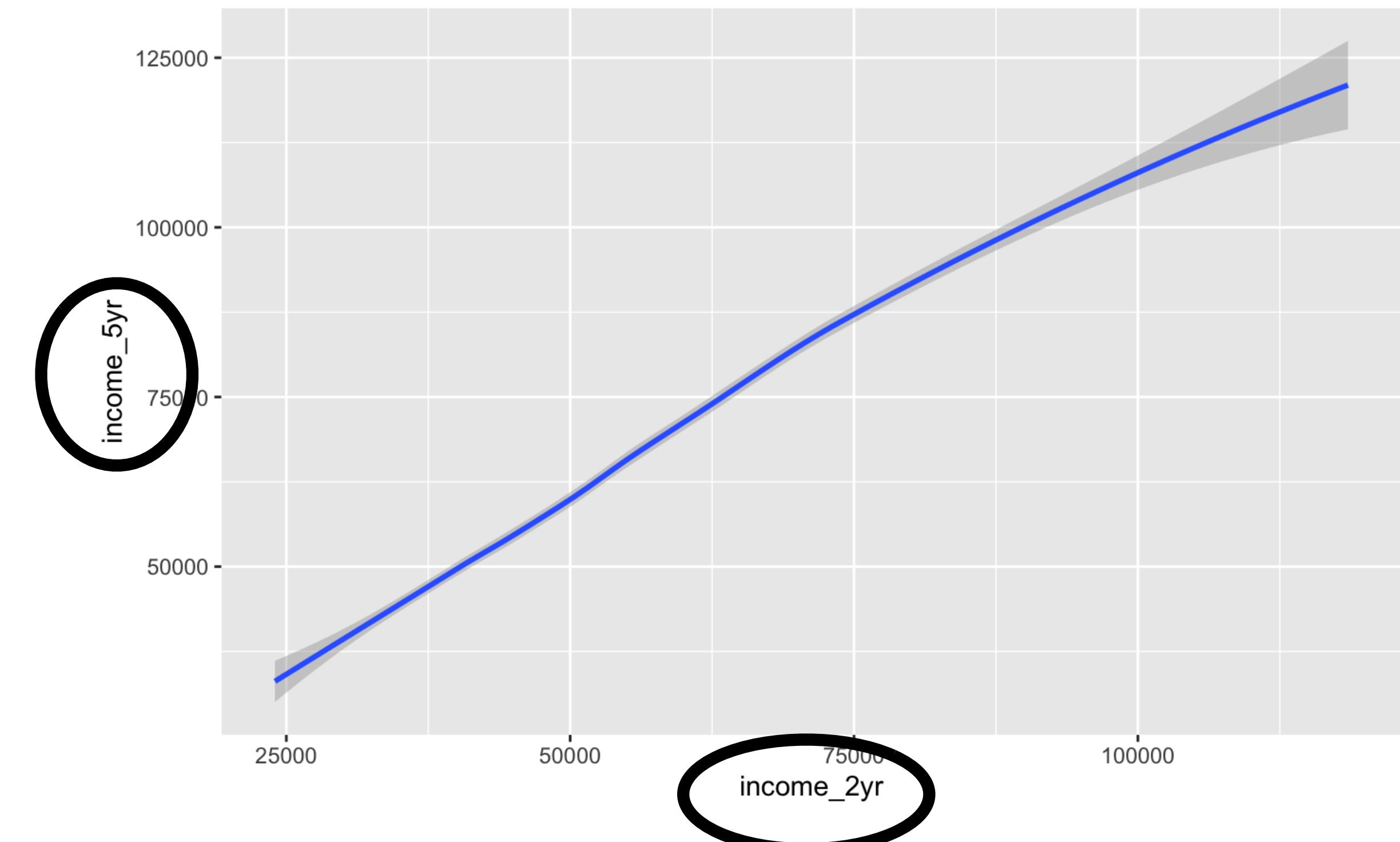
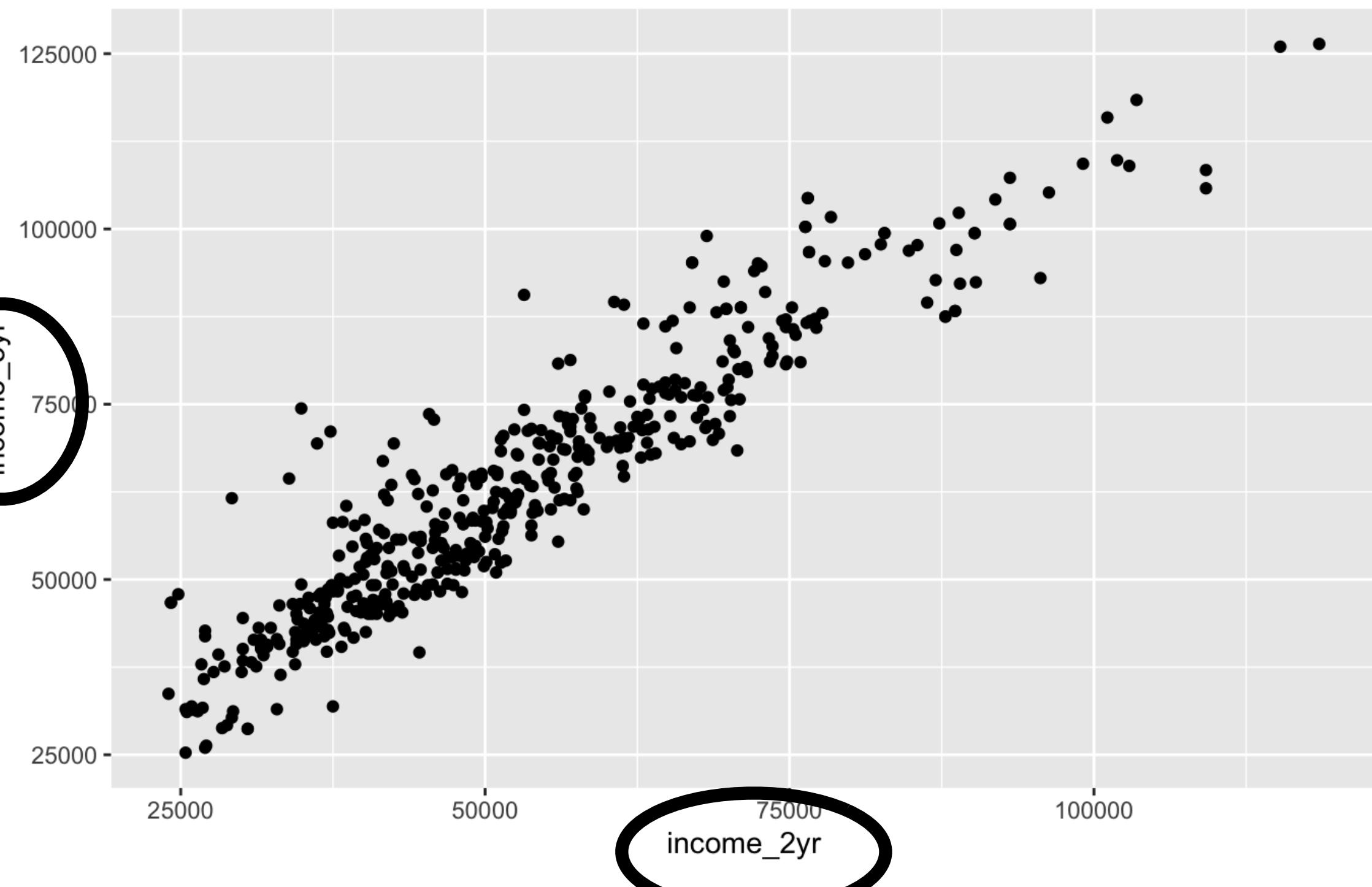
```
ggplot(income) + geom_point(aes(x = income_2yr, y = income_5yr), color = "blue")
```

Geoms

R

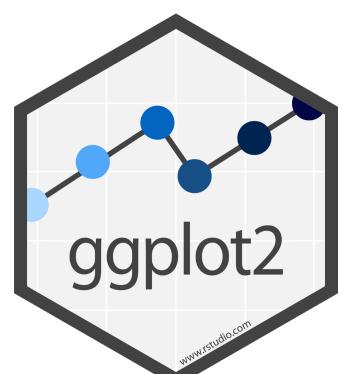
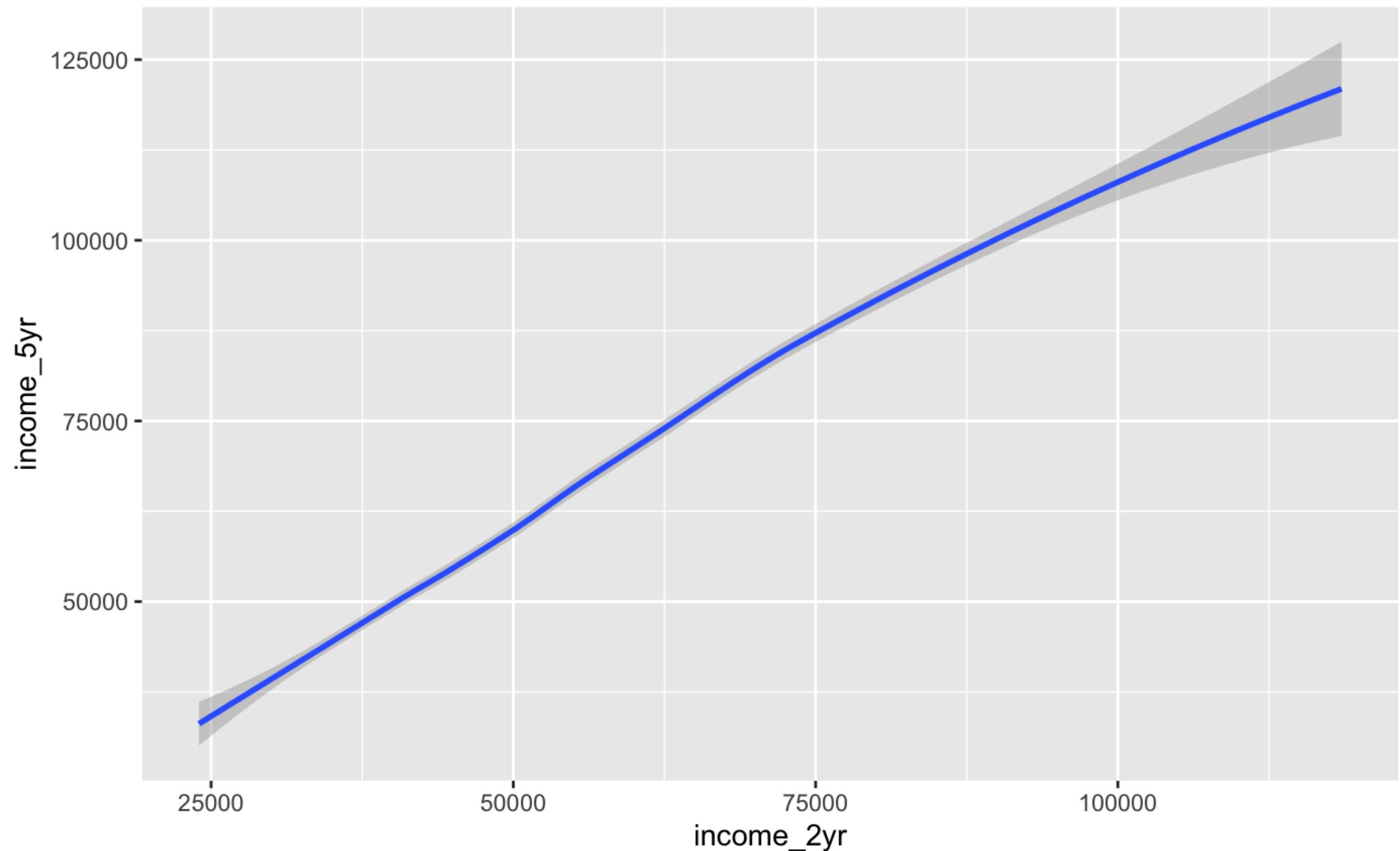
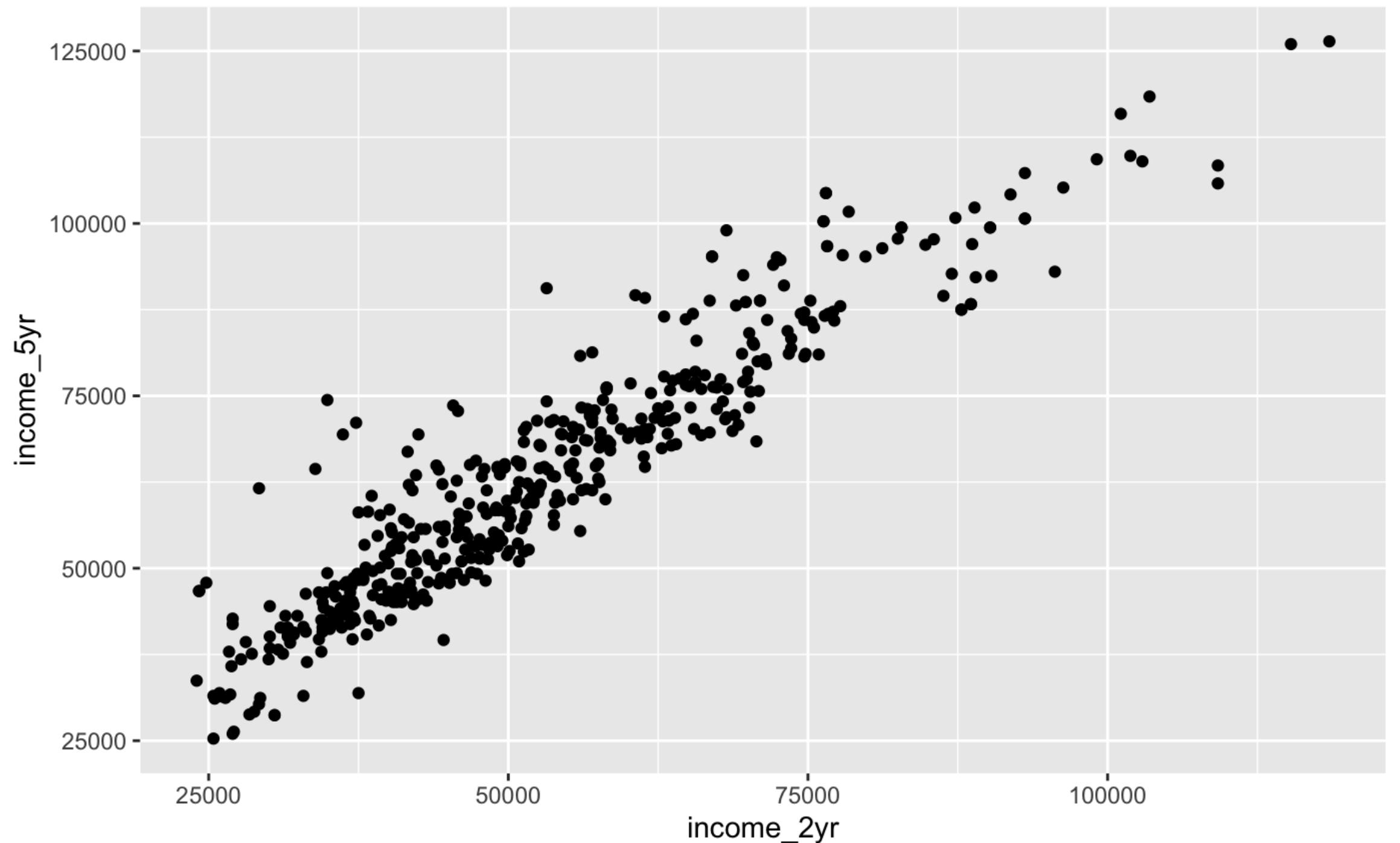
How are these plots similar?

Same: x var , y var , data



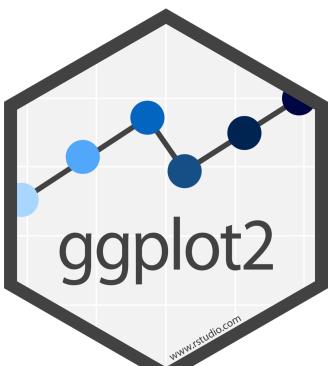
How are these plots different?

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



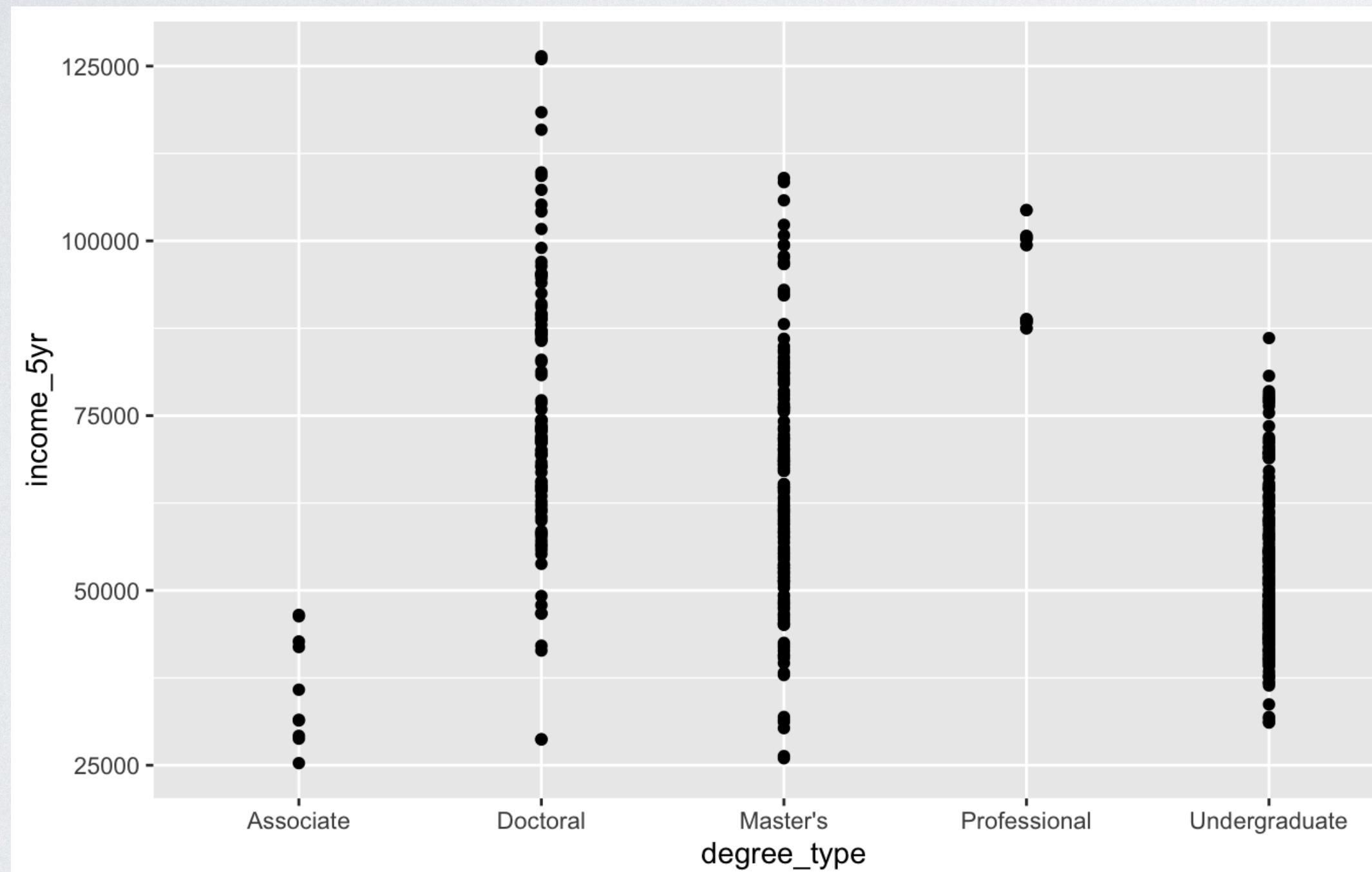
geoms

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

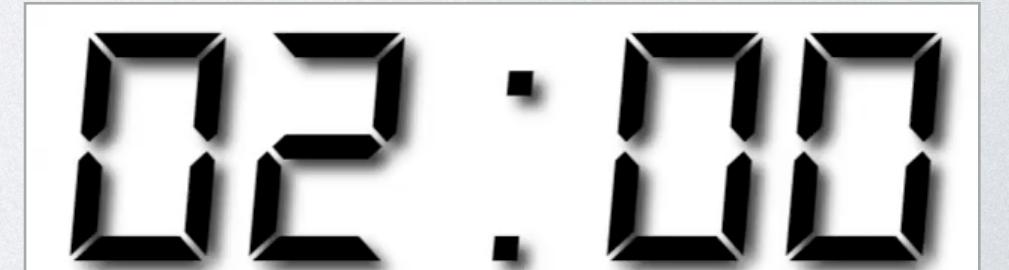
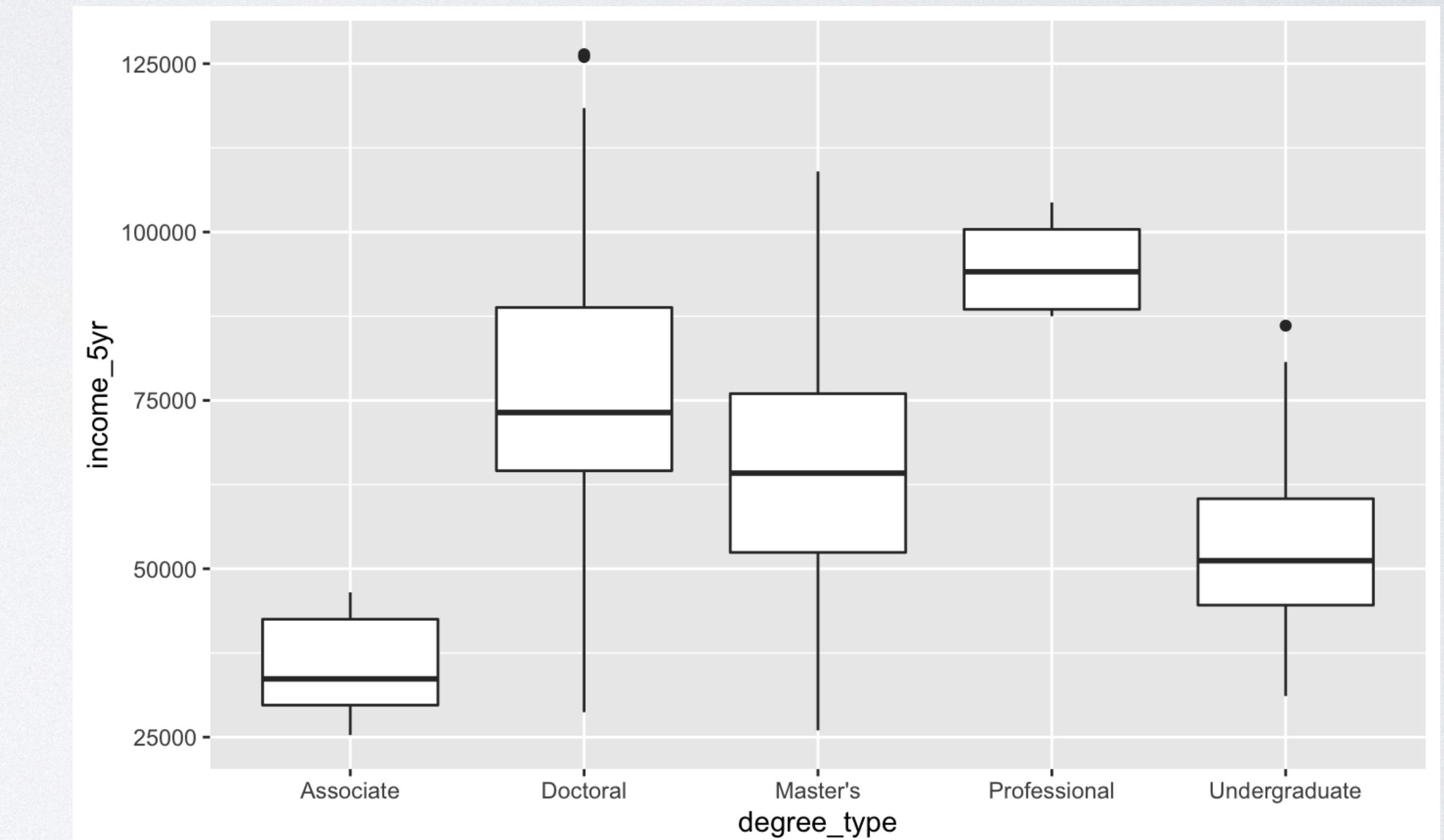


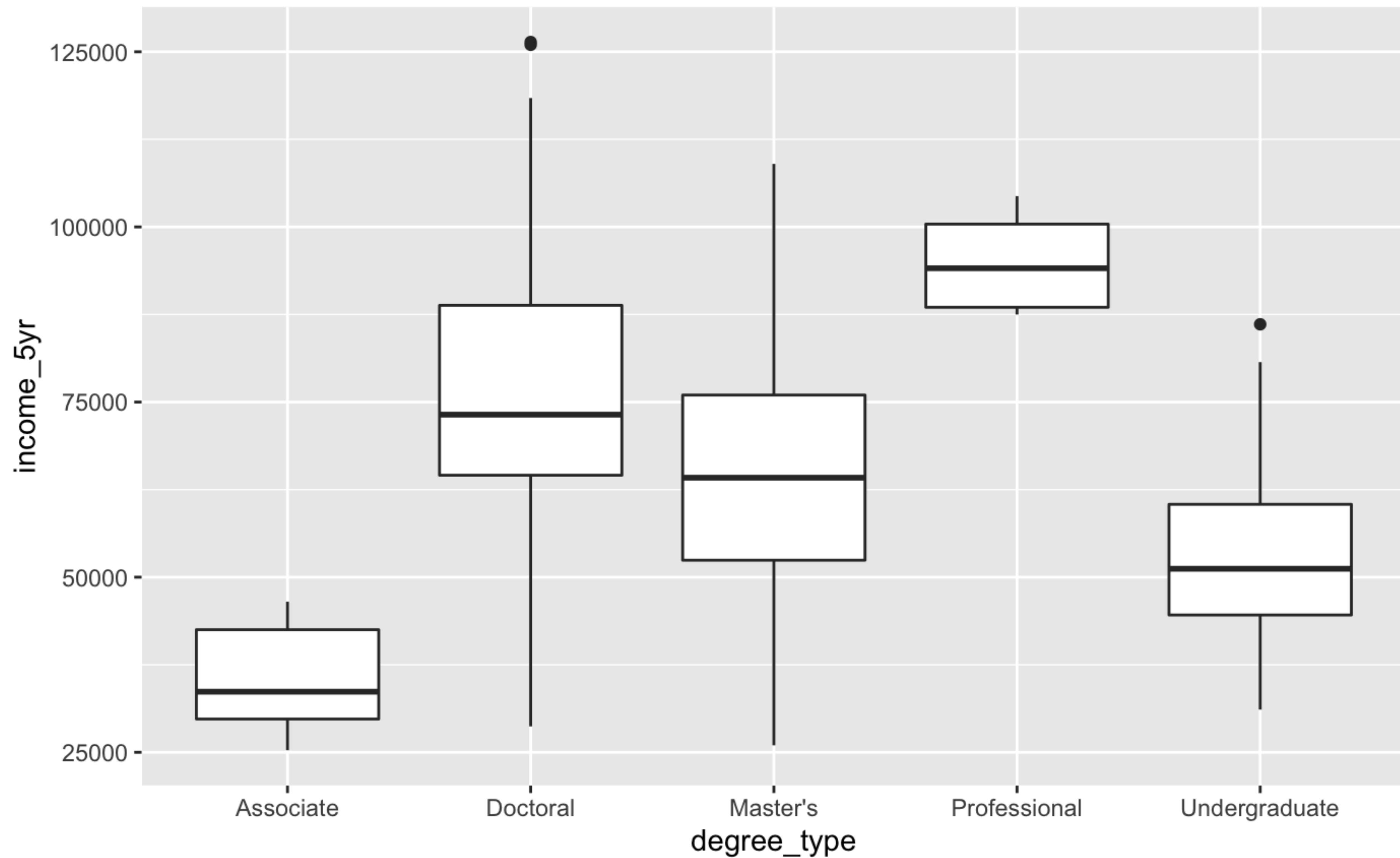
Your Turn 3

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

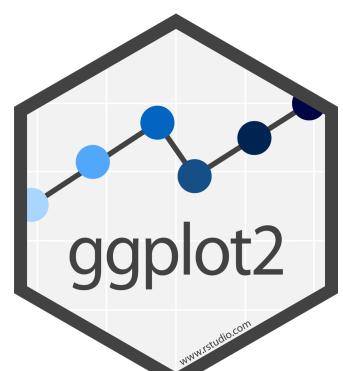


```
ggplot(income) +  
  geom_point(aes(degree_type, income_5yr))
```



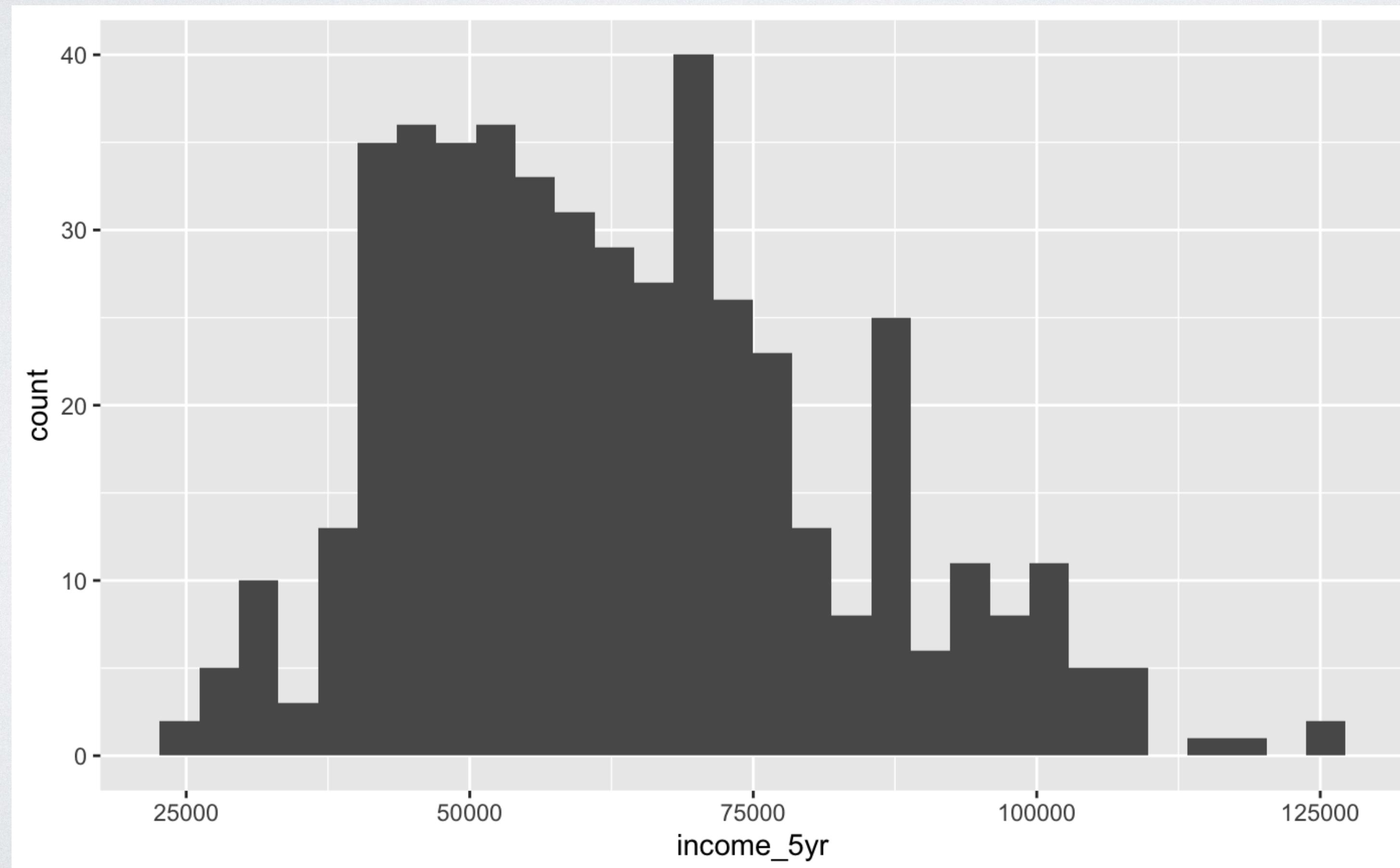


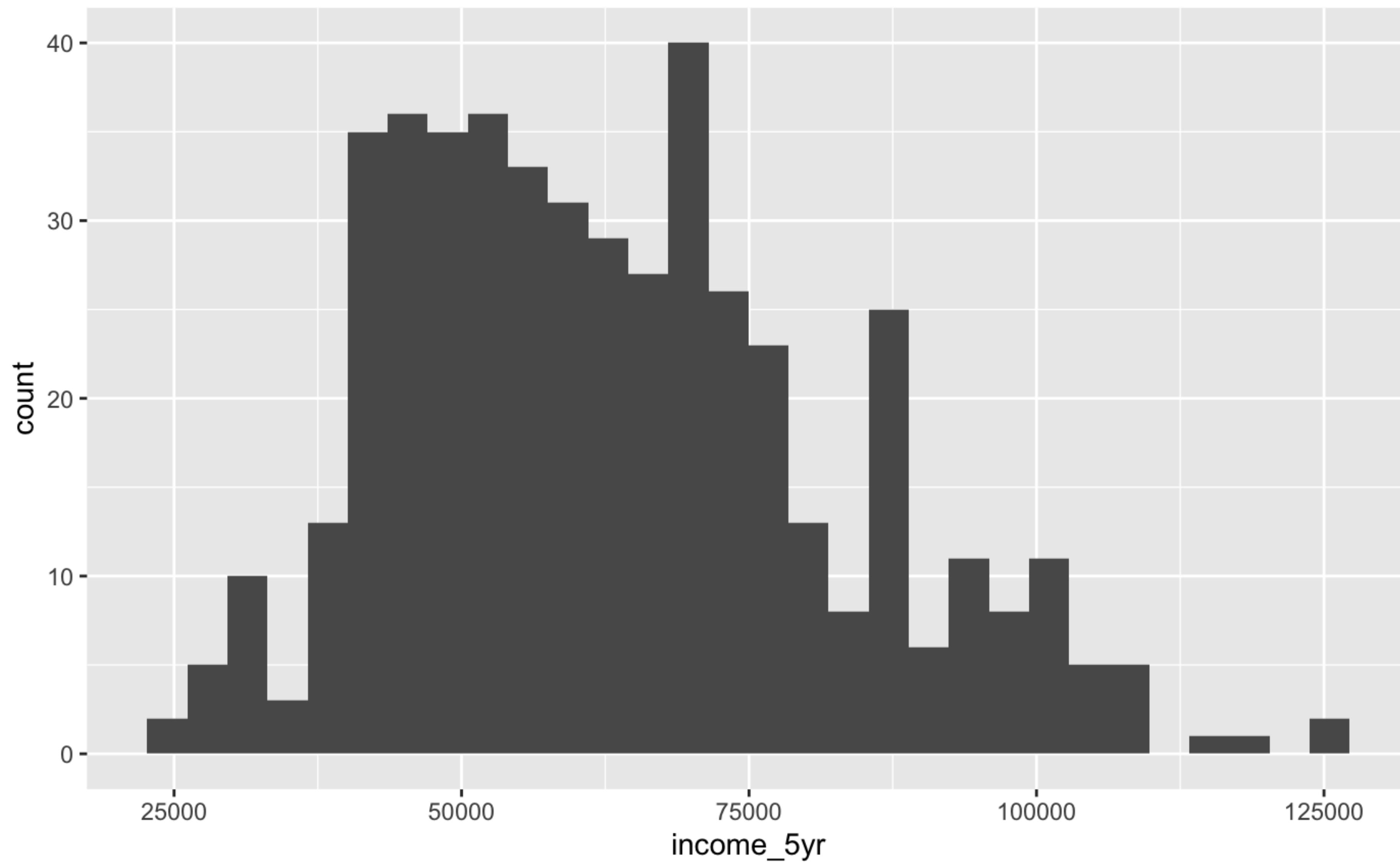
```
ggplot(income) +  
  geom_boxplot(aes(degree_type, income_5yr))
```



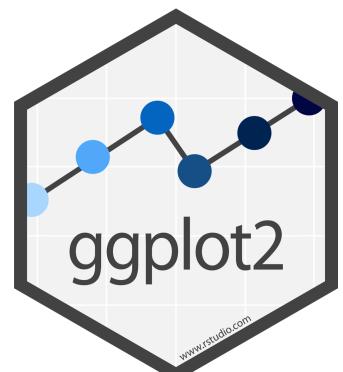
Your Turn 4

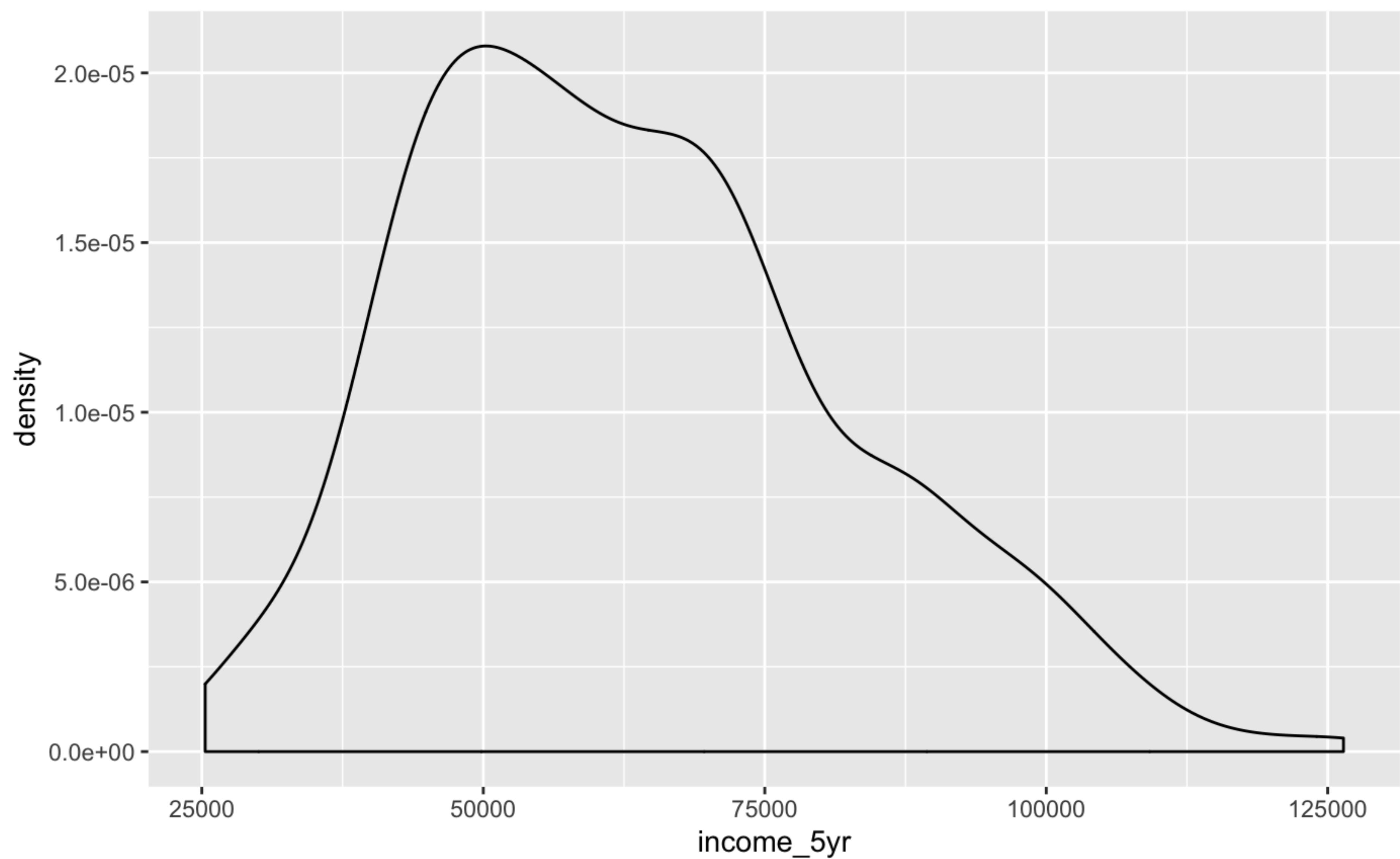
With your partner, make the histogram of **income_5yr** below.
Use the cheatsheet. Hint: do not supply a **y** variable.



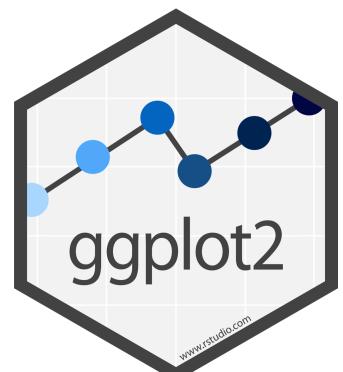


```
ggplot(data = income) +  
  geom_histogram(mapping = aes(x = income_5yr))
```



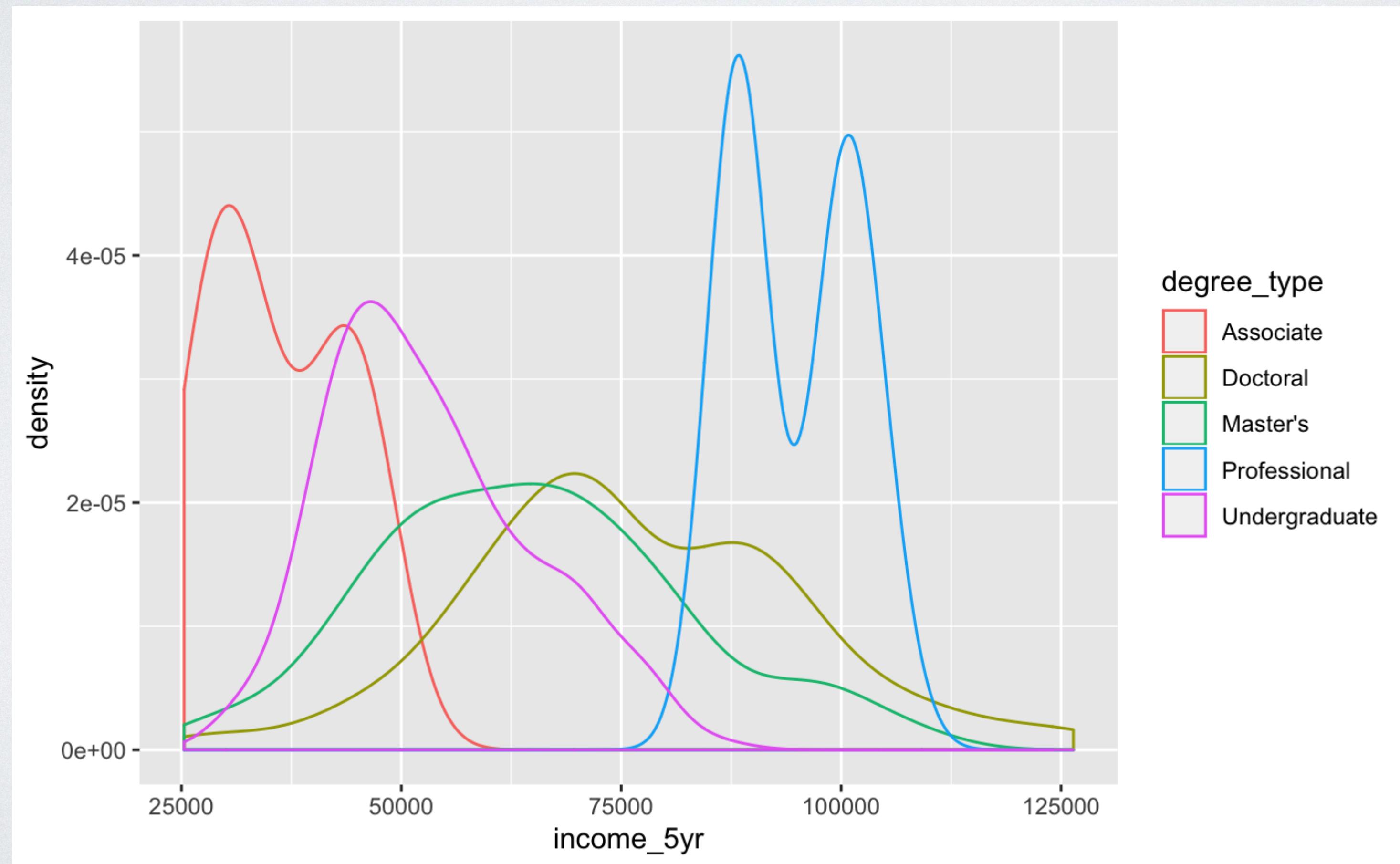


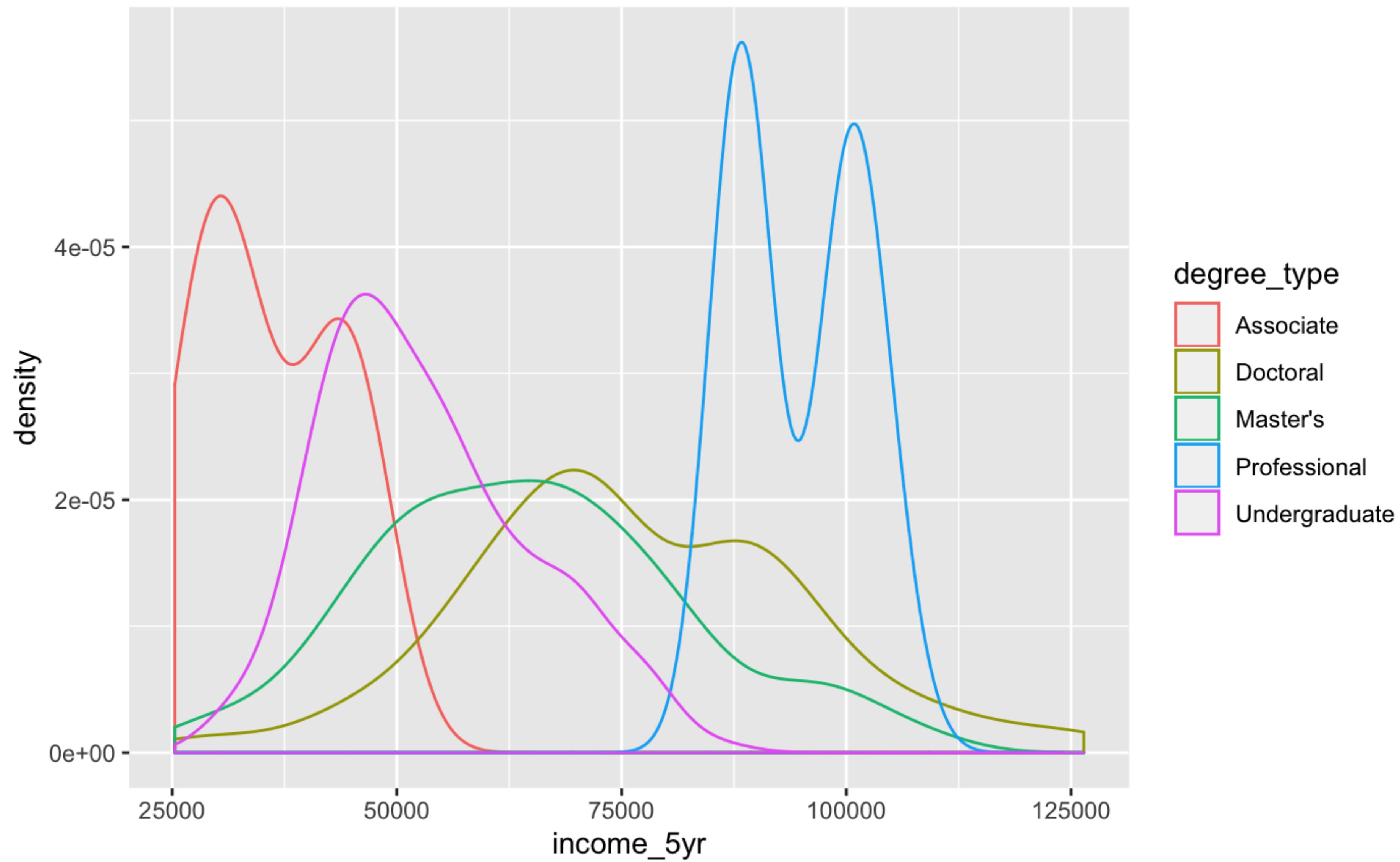
```
ggplot(data = income) +  
  geom_density(mapping = aes(x = income_5yr))
```



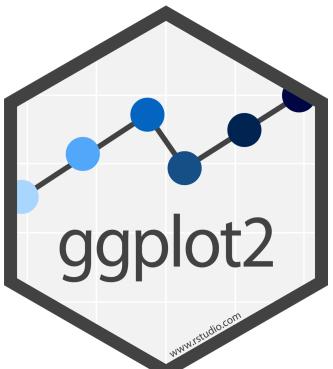
Your Turn 5

With your partner, make the density plot of **income_5yr** colored by **degree_type** below. Use the cheatsheet. Try your best guess.



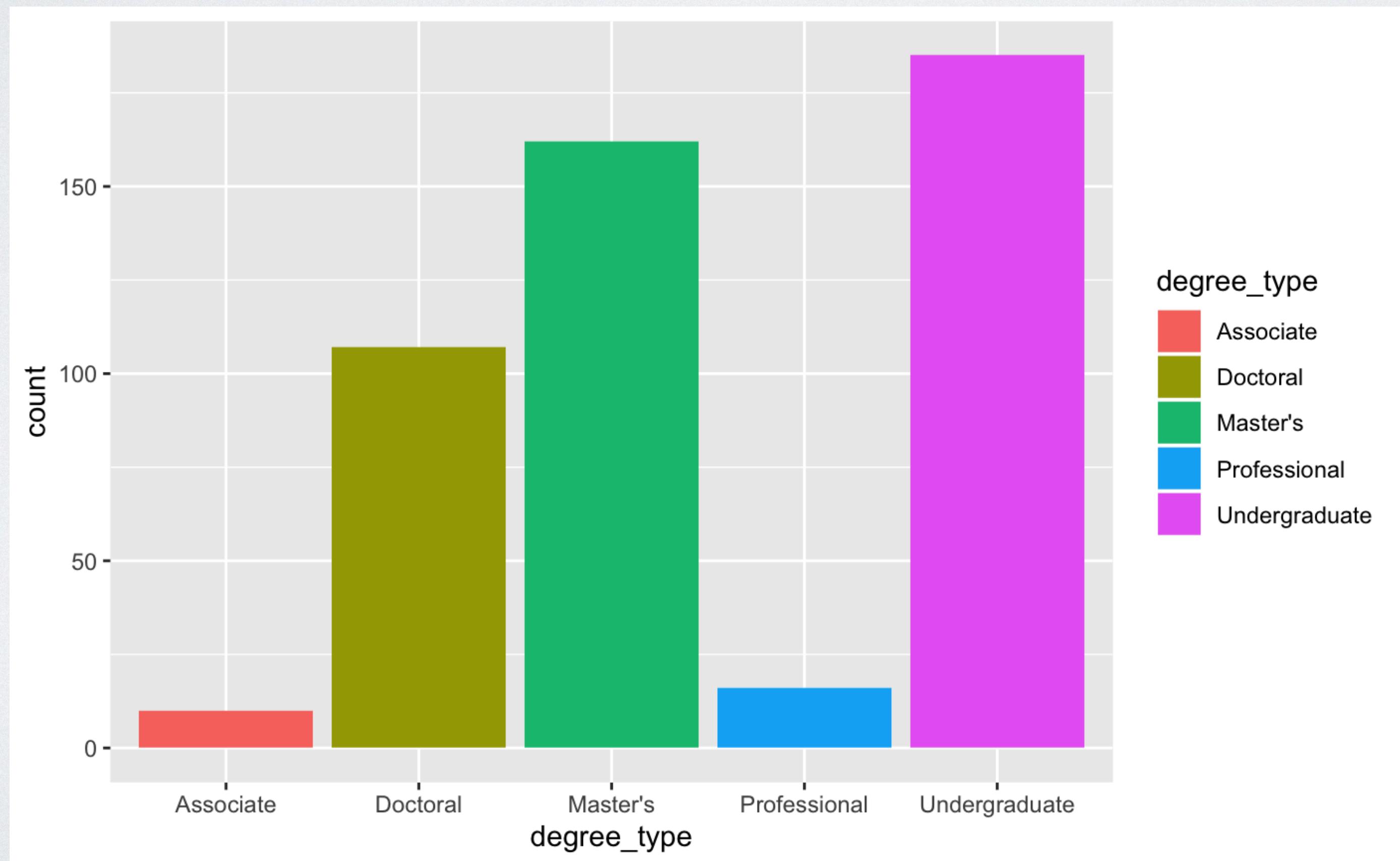


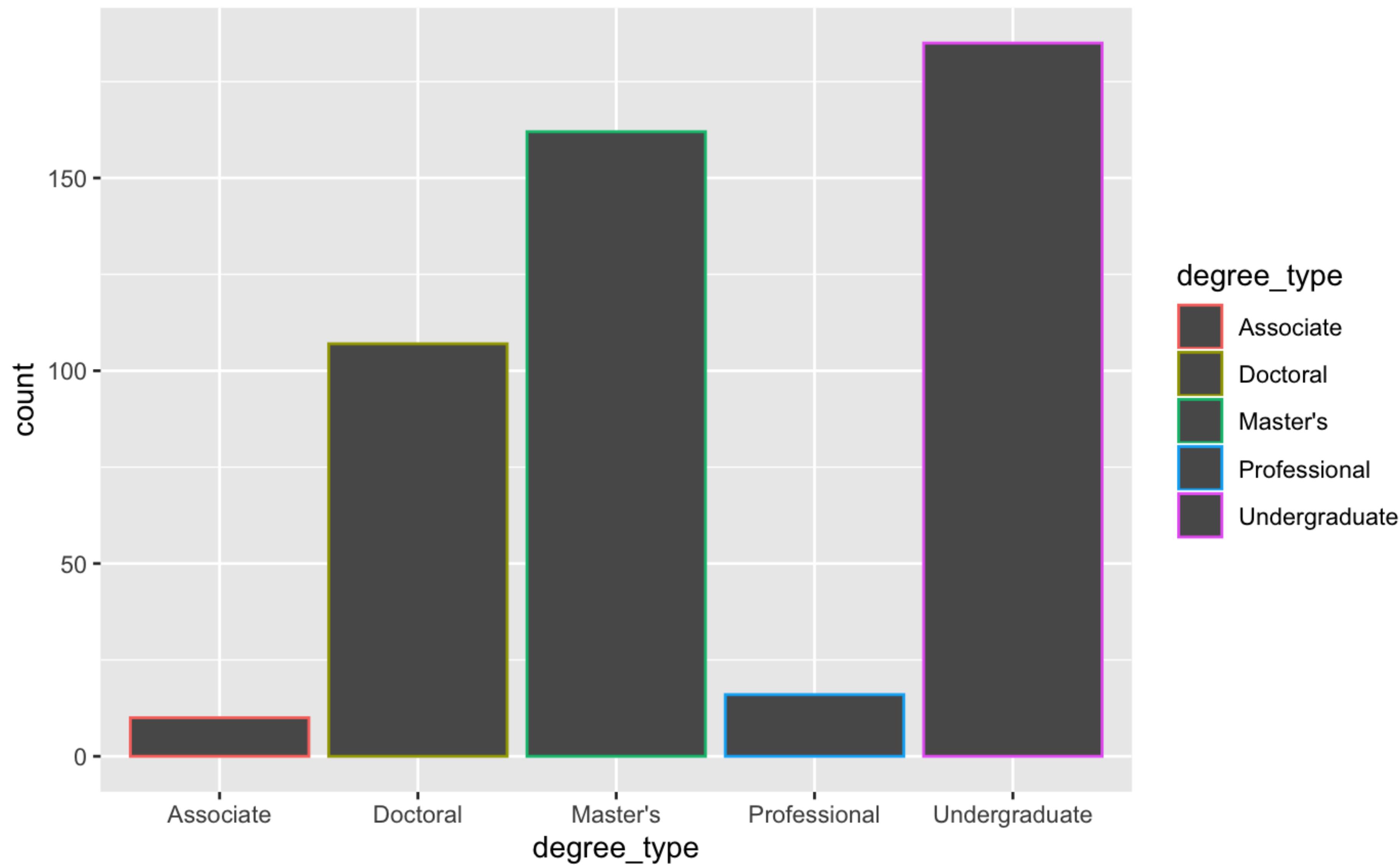
```
ggplot(data = income) +  
  geom_density(mapping = aes(x = income_5yr, color = degree_type))
```



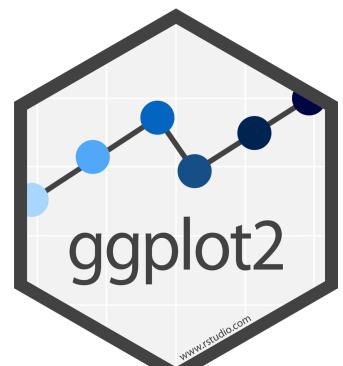
Your Turn 6

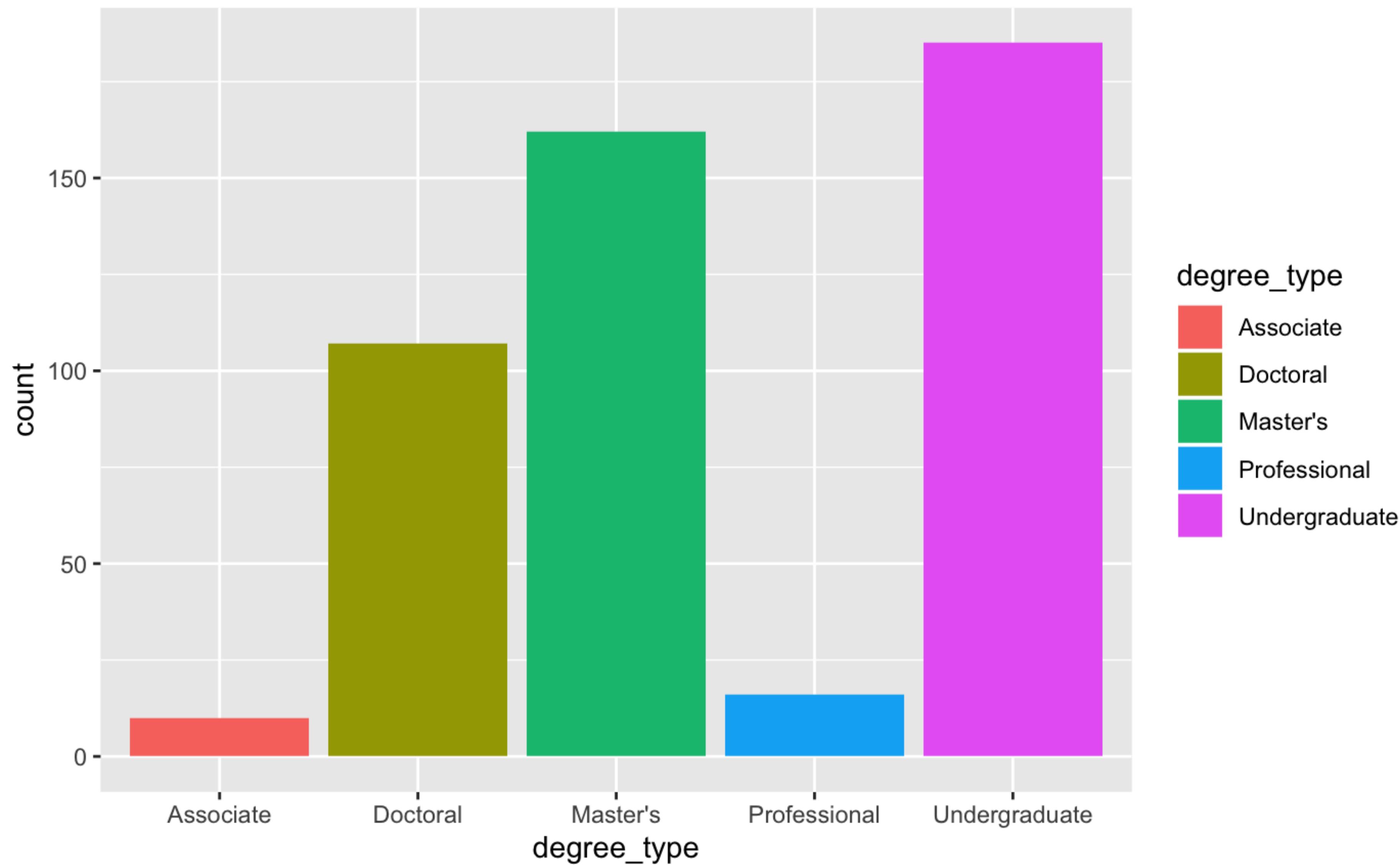
With your partner, make the bar chart of **degree_type** colored by **degree_type** below. Use the cheatsheet. Try your best guess.



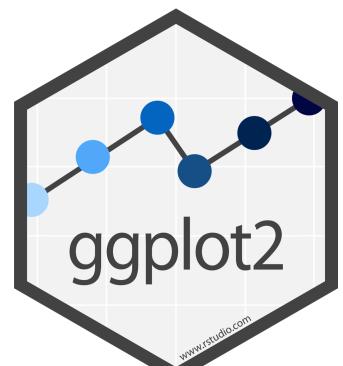


```
ggplot(data = income) +  
  geom_bar(mapping = aes(x = degree_type, color = degree_type))
```





```
ggplot(data = income) +  
  geom_bar(mapping = aes(x = degree_type, fill = degree_type))
```



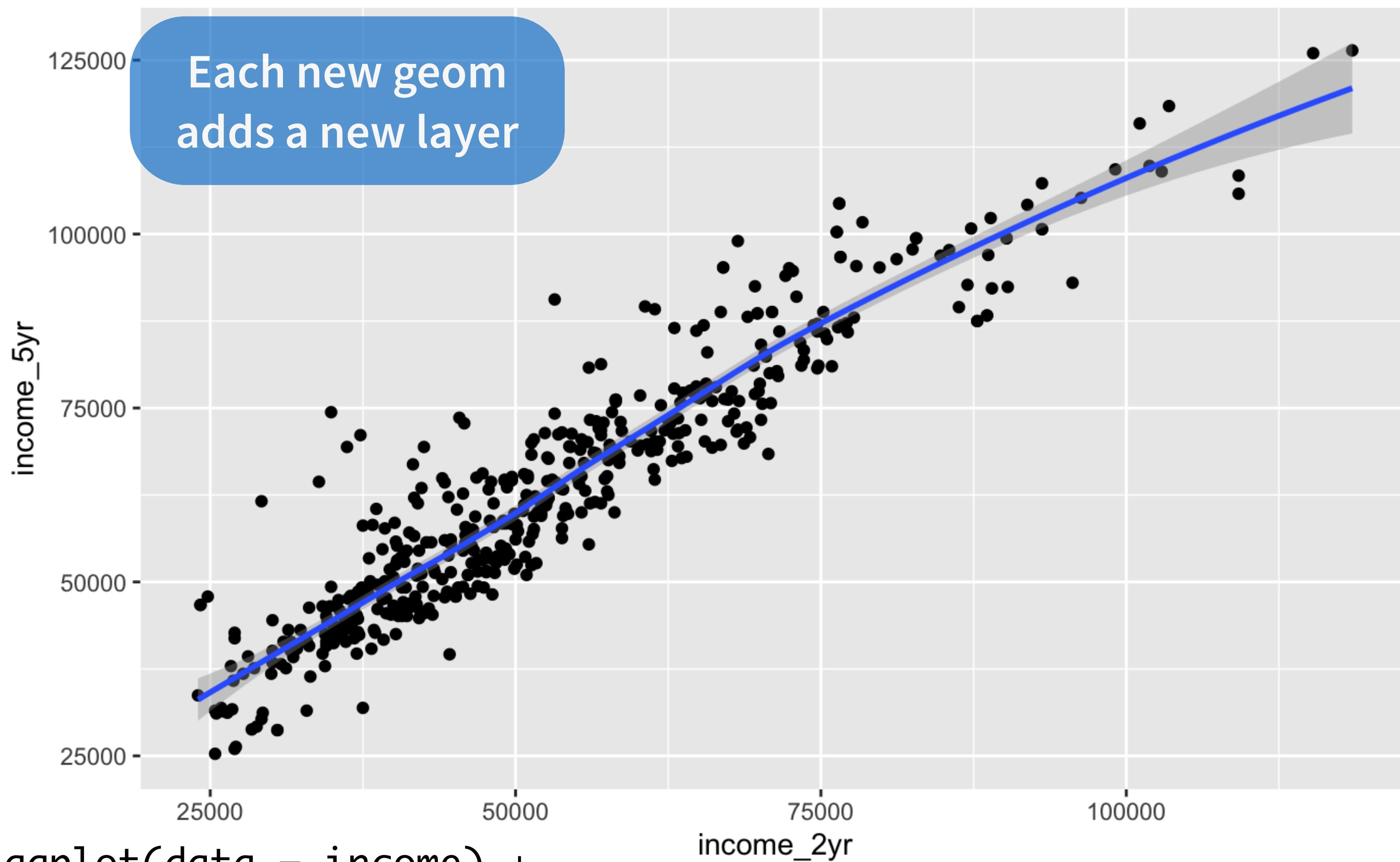
Your Turn 7

With your partner, predict what this code will do.

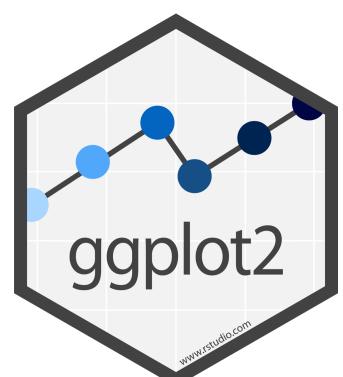
Then run it.

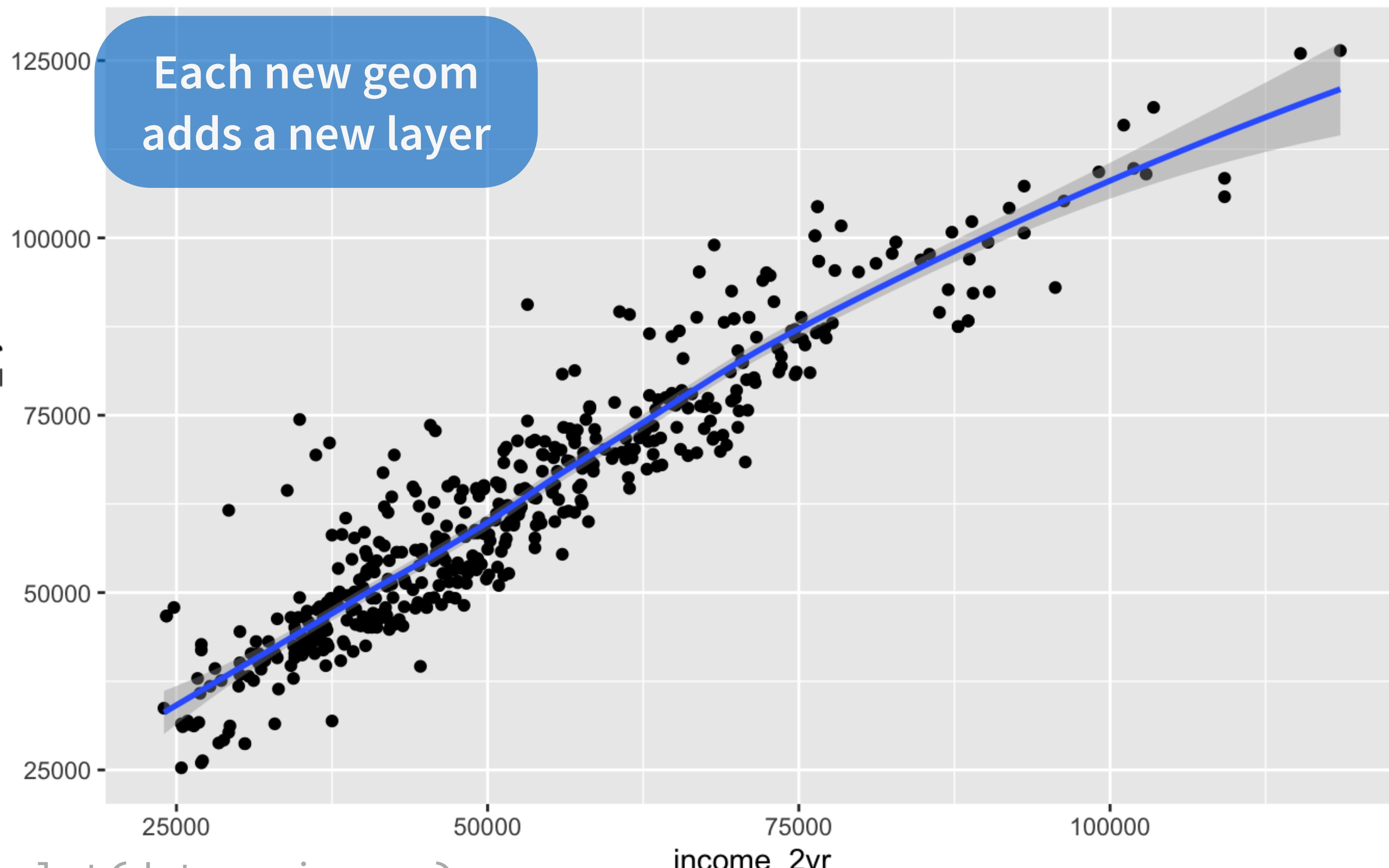
```
ggplot(income) +  
  geom_point(aes(income_2yr, income_5yr)) +  
  geom_smooth(aes(income_2yr, income_5yr))
```



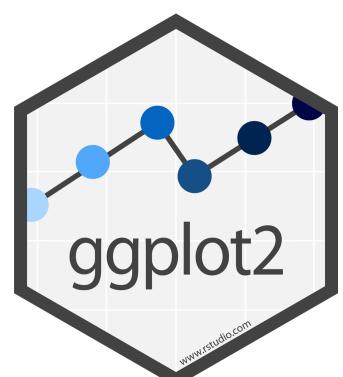


```
ggplot(data = income) +  
  geom_point(mapping = aes(x = income_2yr, y = income_5yr)) +  
  geom_smooth(mapping = aes(x = income_2yr, y = income_5yr))
```



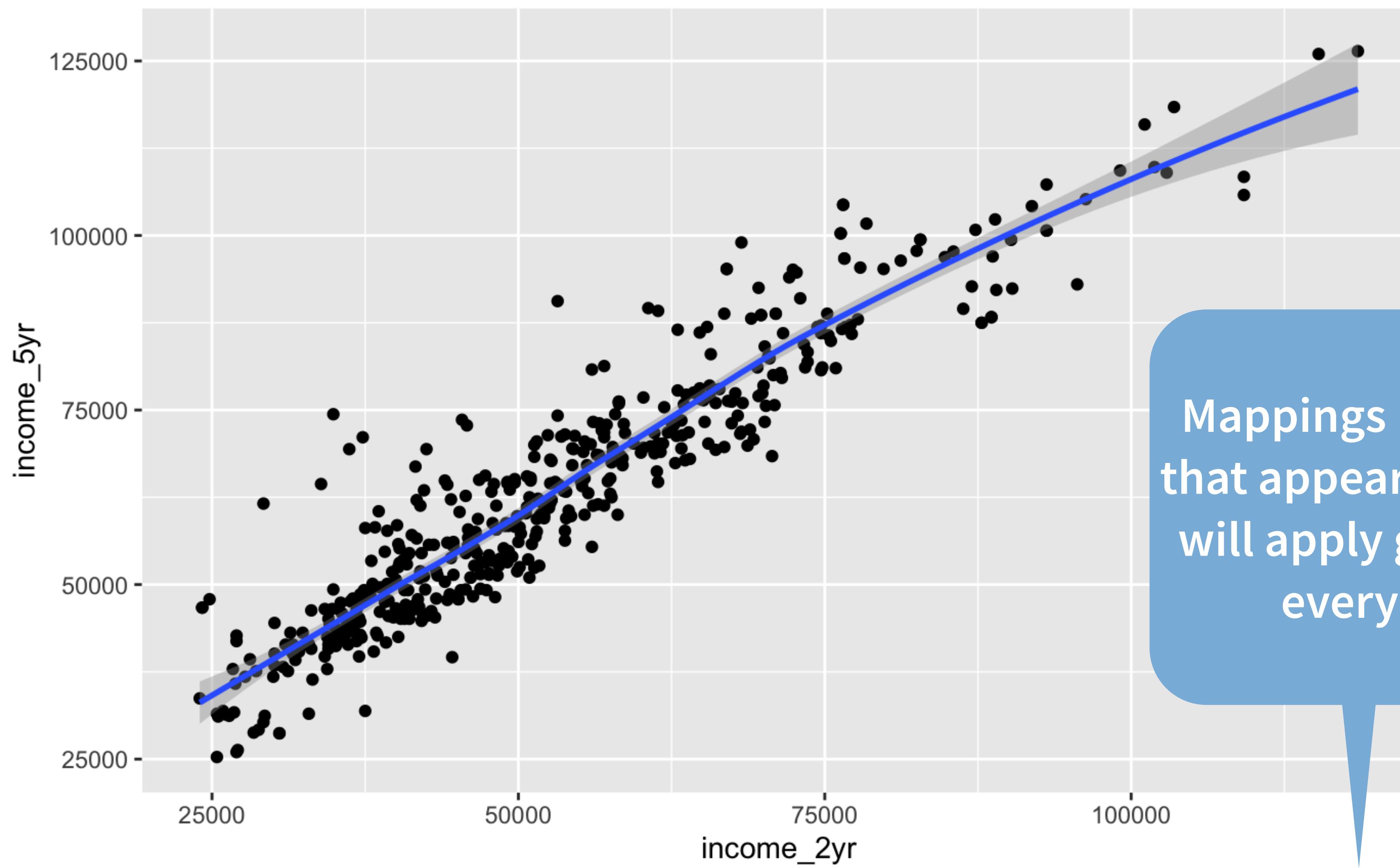


```
ggplot(data = income) +  
  geom_point(mapping = aes(x = income_2yr, y = income_5yr)) +  
  geom_smooth(mapping = aes(x = income_2yr, y = income_5yr))
```

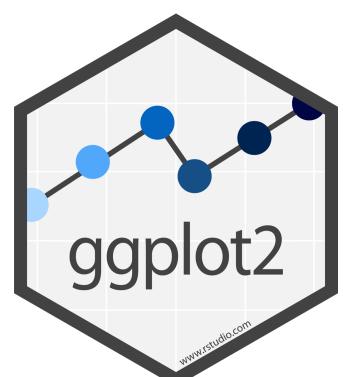


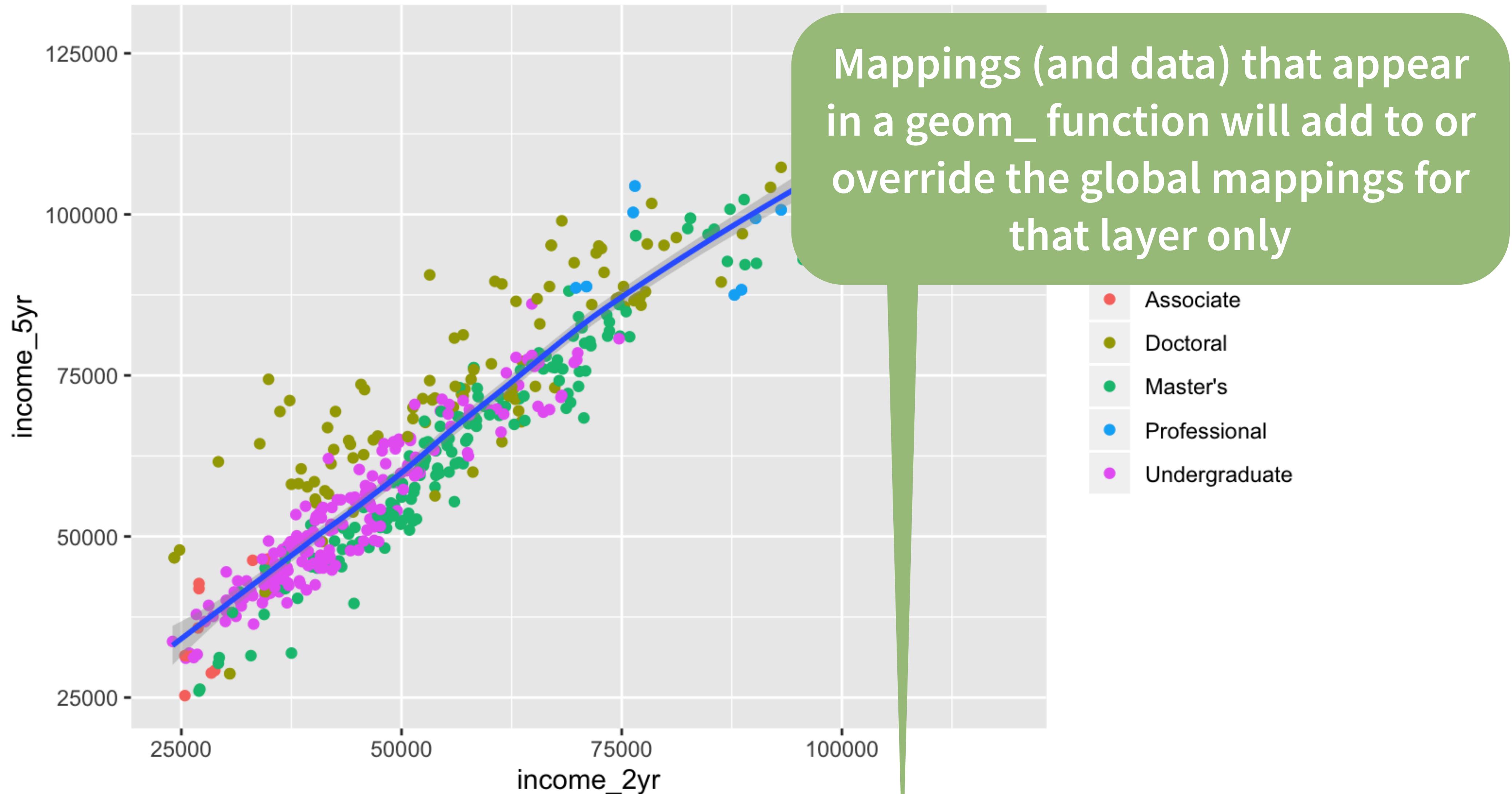
global vs. local

R

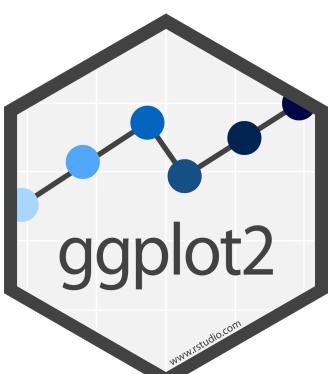


```
ggplot(data = income, mapping = aes(x = income_2yr, y = income_5yr)) +  
  geom_point() +  
  geom_smooth()
```





```
ggplot(data = income, mapping = aes(x = income_2yr, y = income_5yr)) +
  geom_point(mapping = aes(color = degree_type)) +
  geom_smooth()
```



Saving graphs



Your Turn 8

What does this command return?

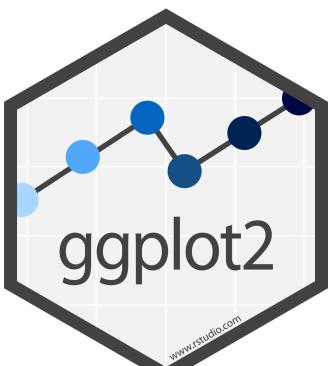
getwd()



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
- When using an R Notebook, the working directory is the same as the directory that the R Notebook is saved in



Saving plots

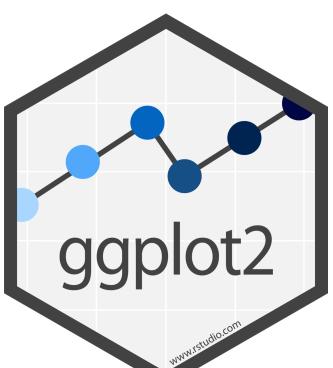
ggsave() saves the last plot.

Uses size on screen:

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

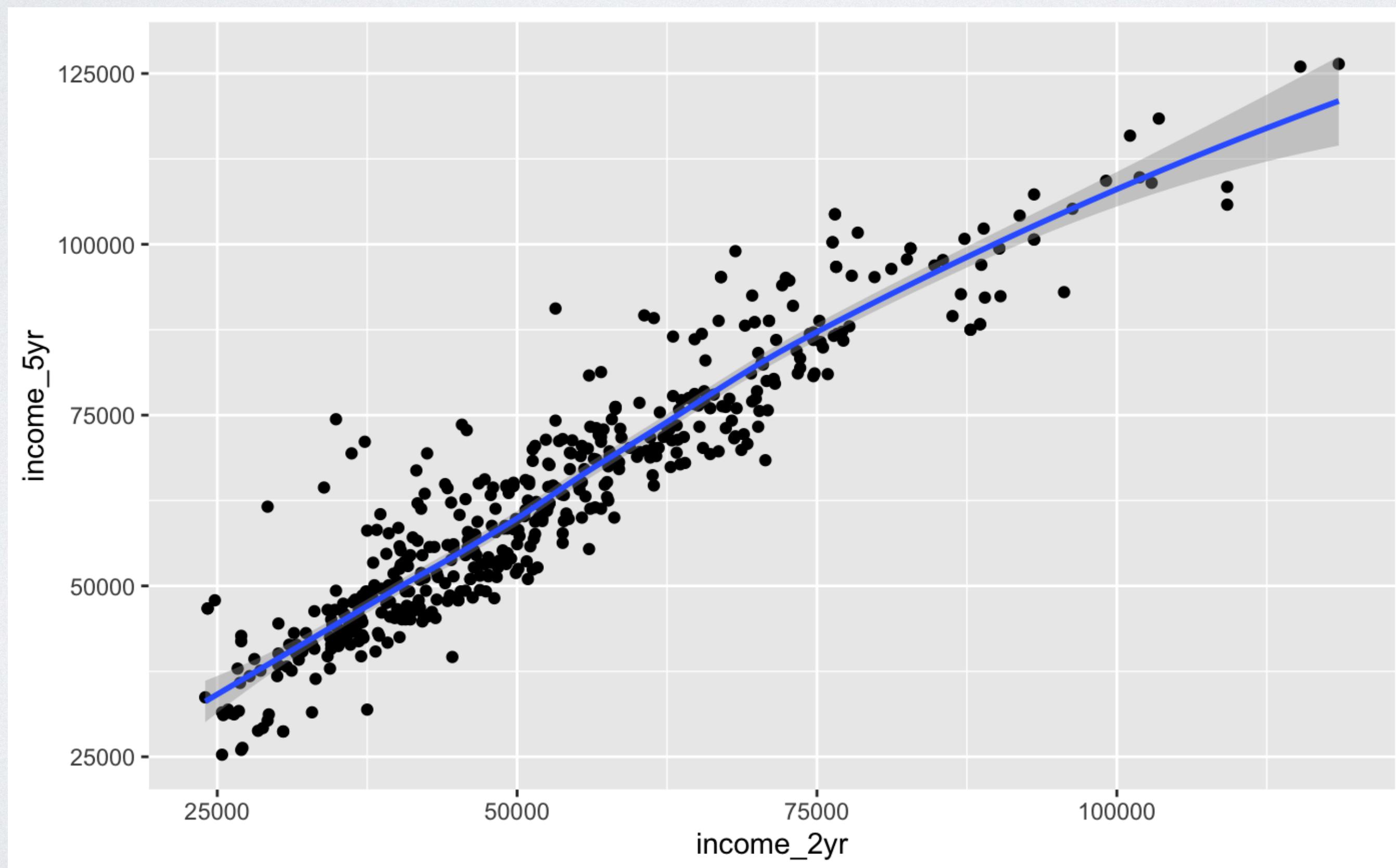
Specify size in inches

```
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).



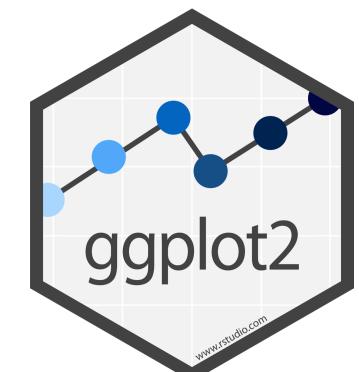
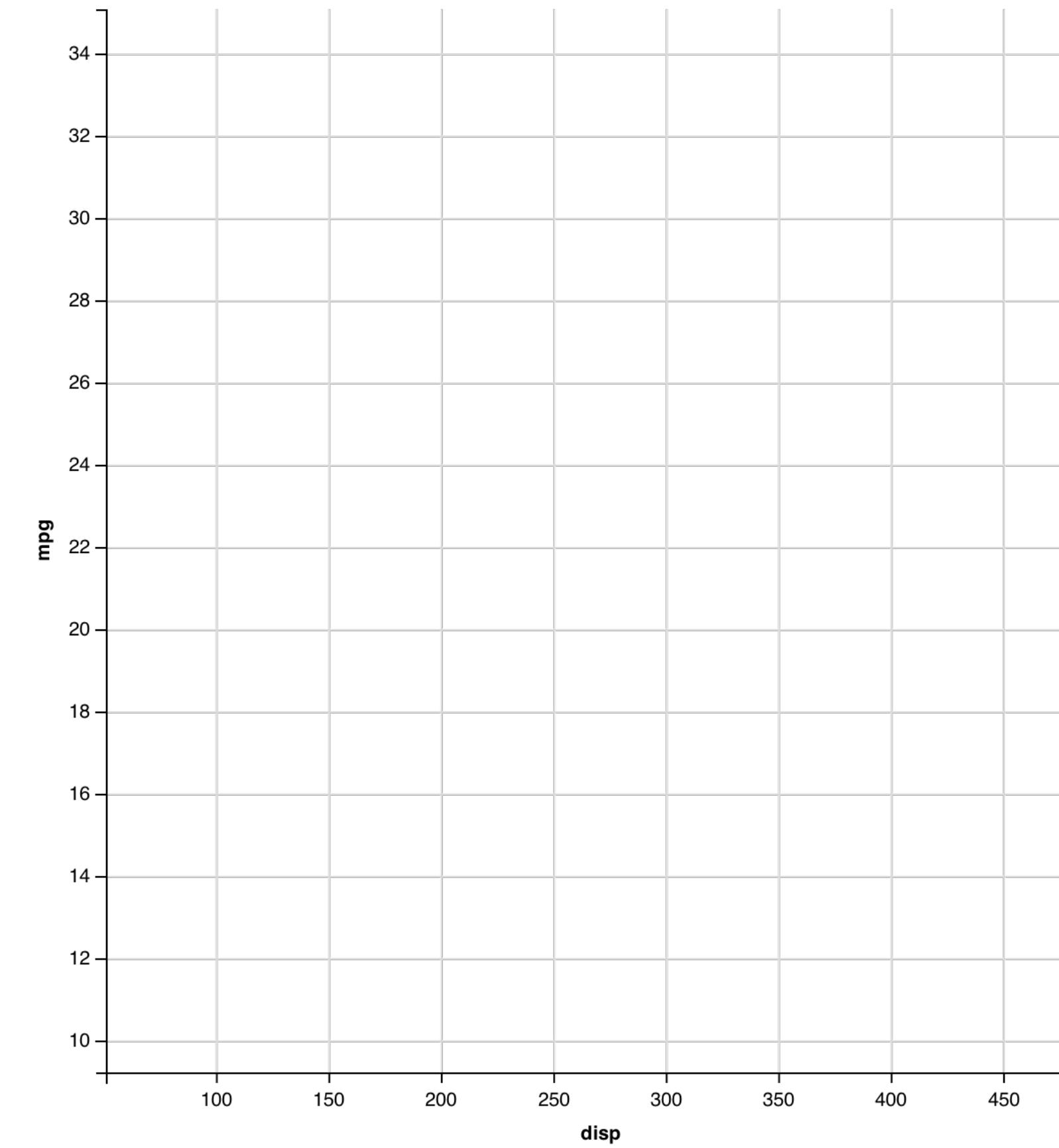
Grammar of Graphics



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

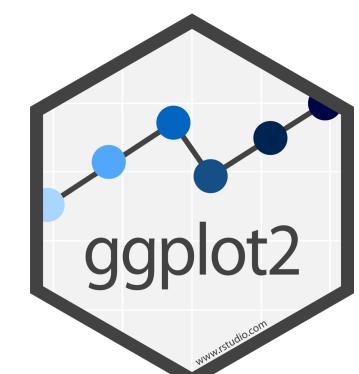
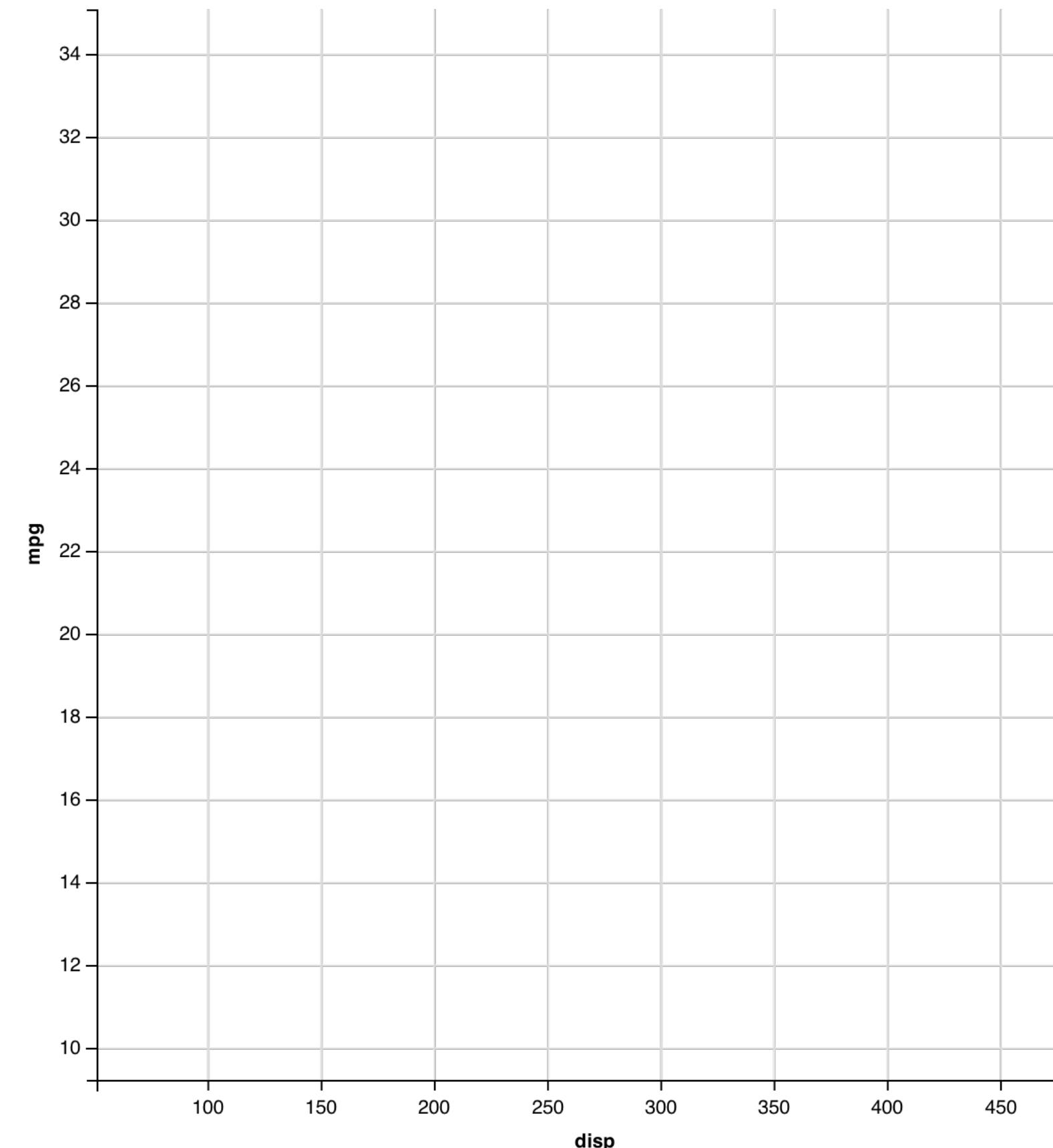
mpg	cyl	disp	hp
21.0	6	160.0	2
21.0	6	160.0	2
22.8	4	108.0	1
21.4	6	258.0	2
18.7	8	360.0	3
18.1	6	225.0	2
14.3	8	360.0	5
24.4	4	146.7	1
22.8	4	140.8	1
19.2	6	167.6	2
17.8	6	167.6	2
16.4	8	275.8	3
17.3	8	275.8	3
15.2	8	275.8	3
10.4	8	472.0	4
10.4	8	460.0	4
14.7	8	440.0	4
32.4	4	78.7	1
30.4	4	75.7	1
33.9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

fill



data

geom

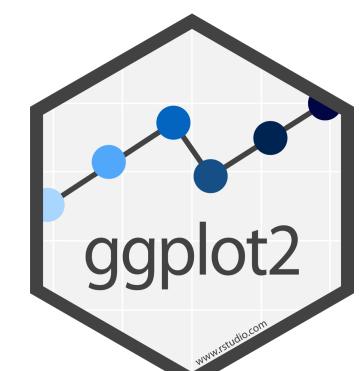
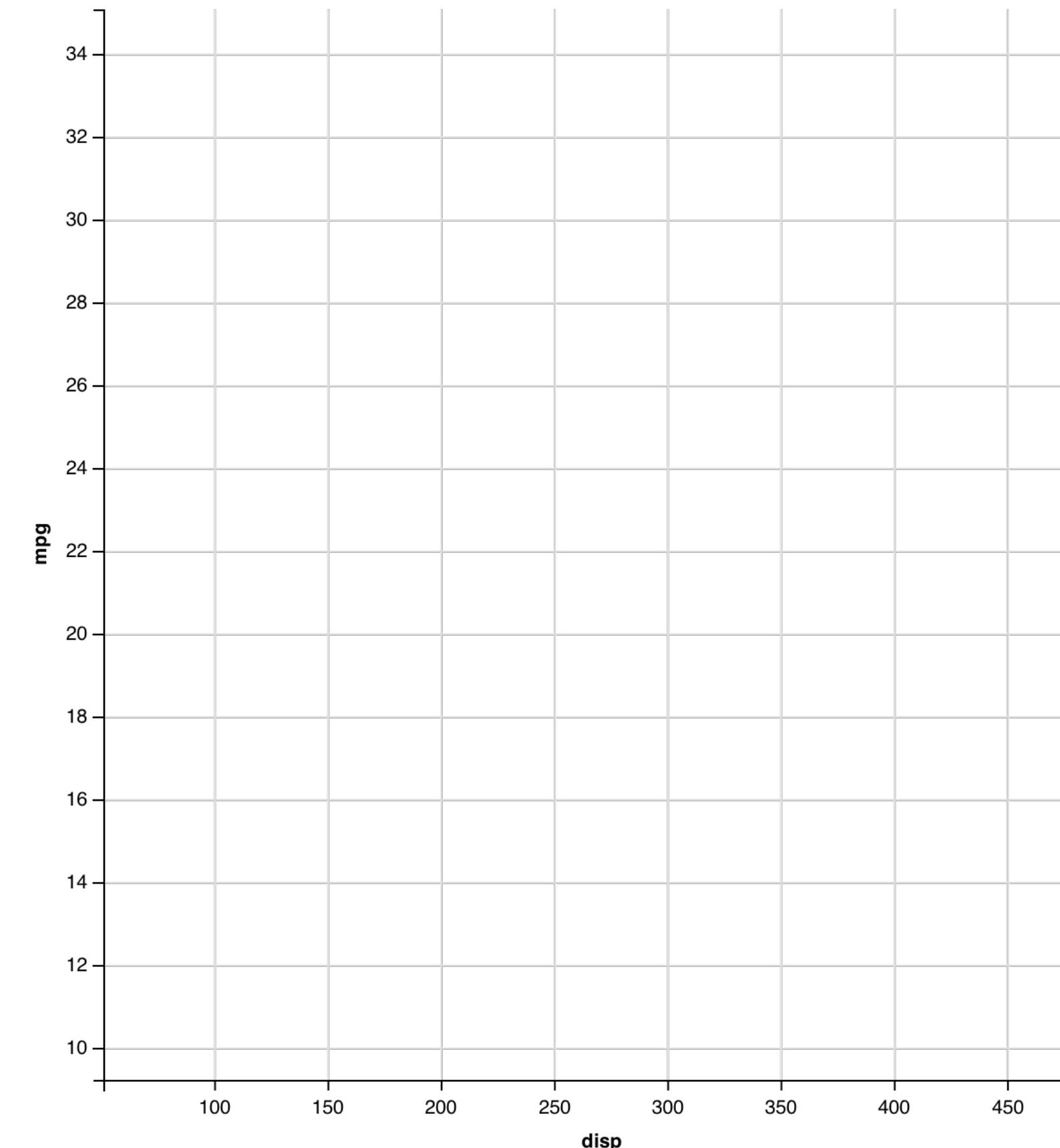


mappings

	shape		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

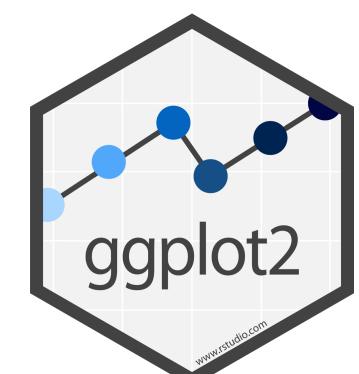
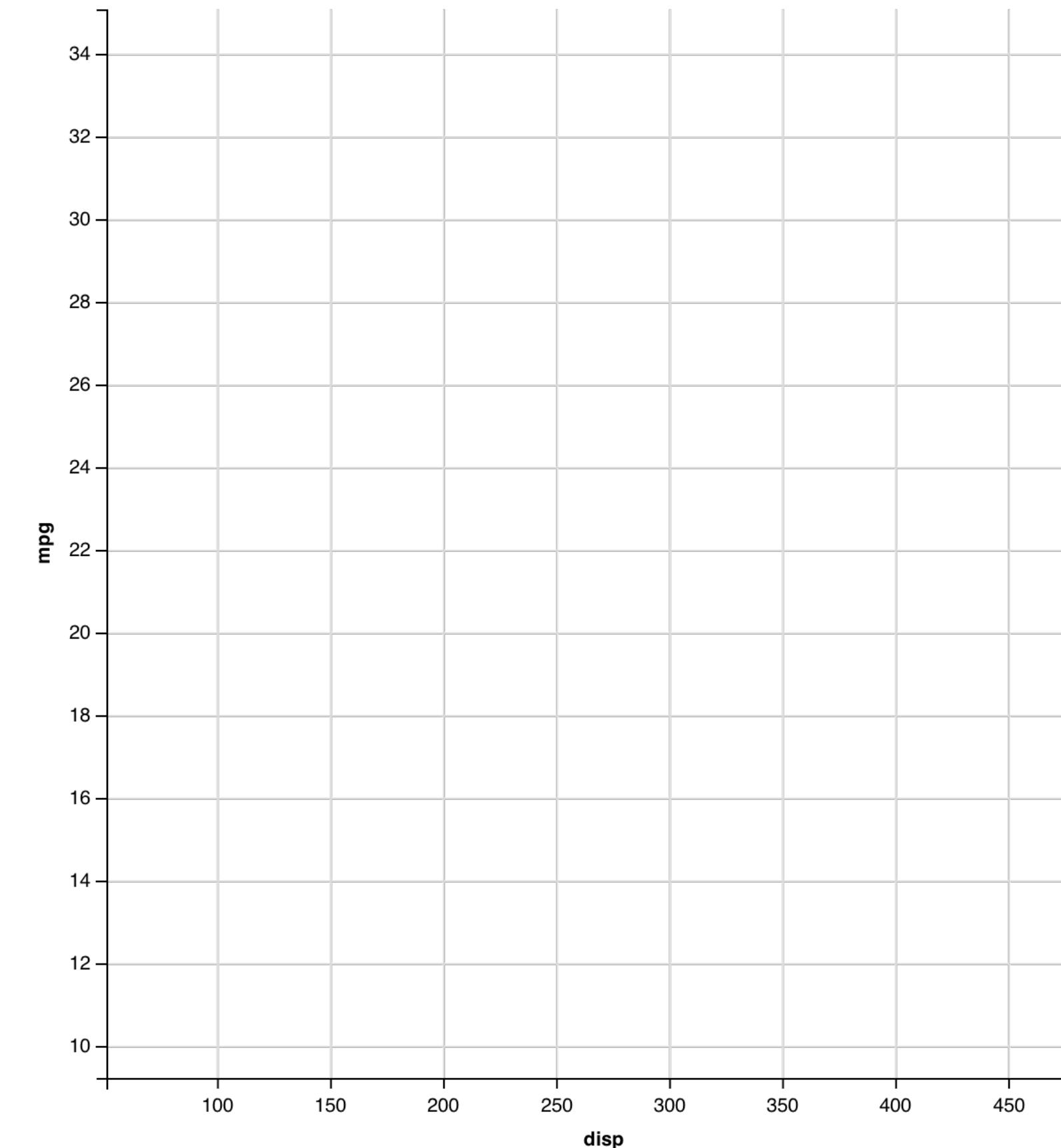


mappings

	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

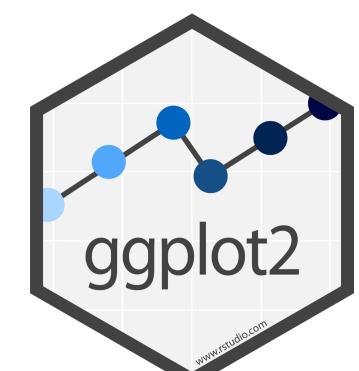
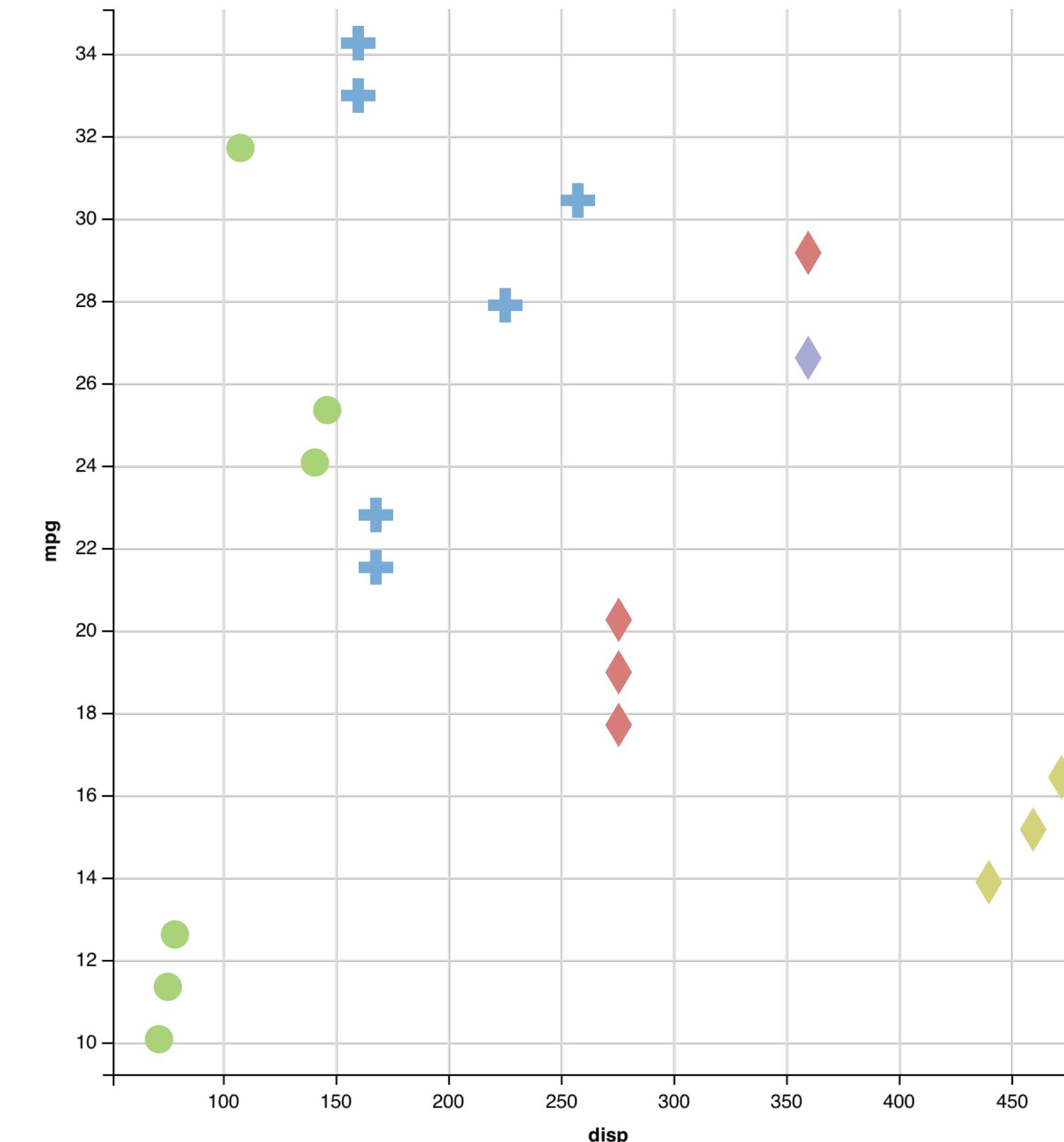


mappings

	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

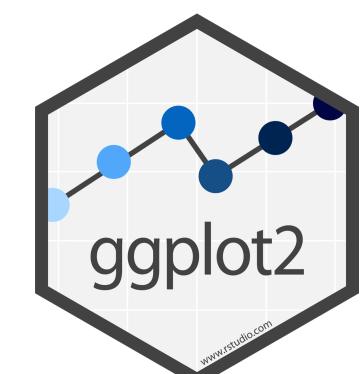
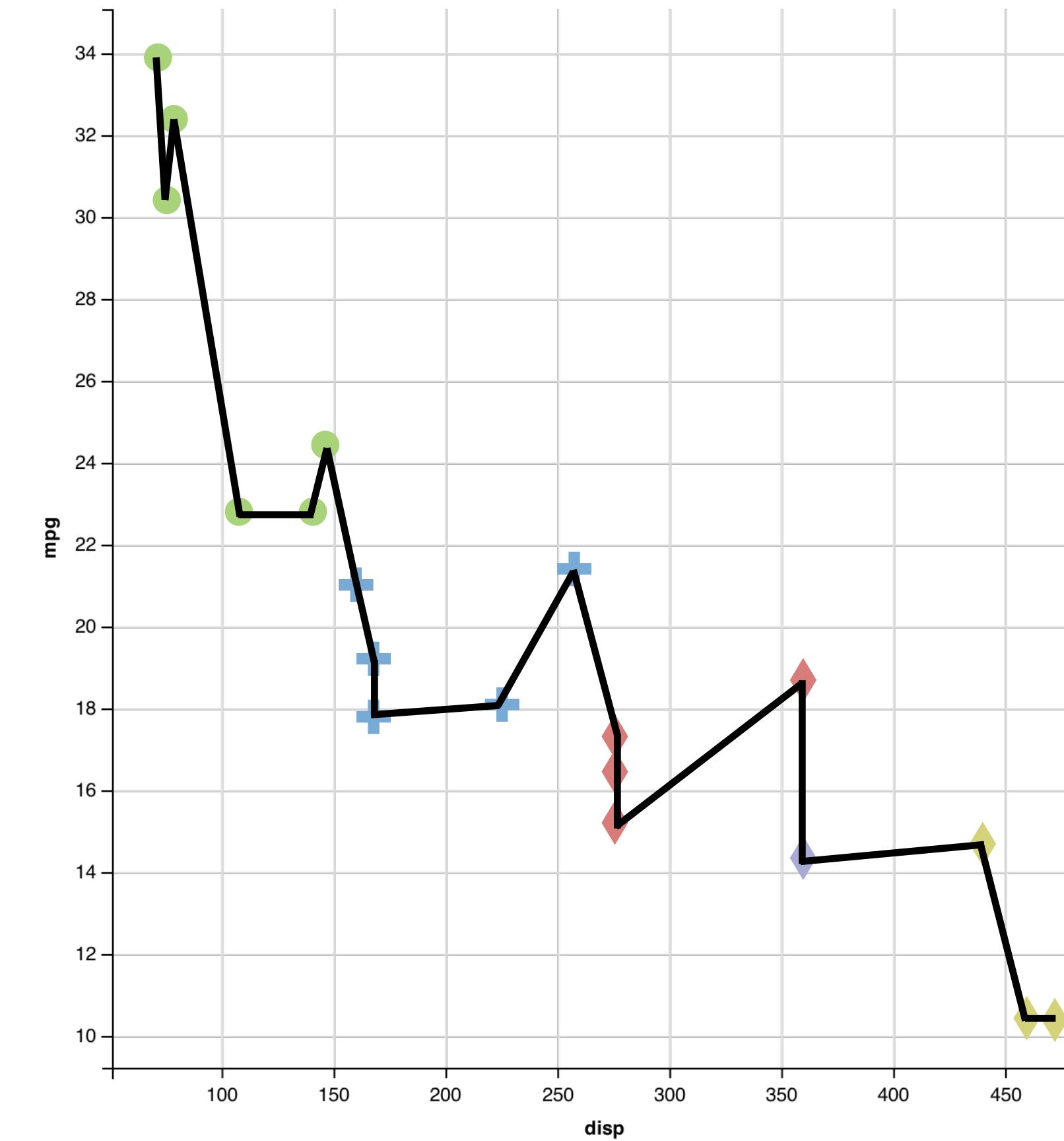


mappings

	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

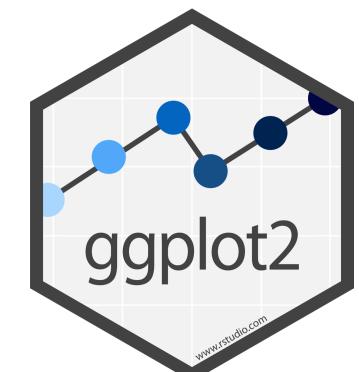
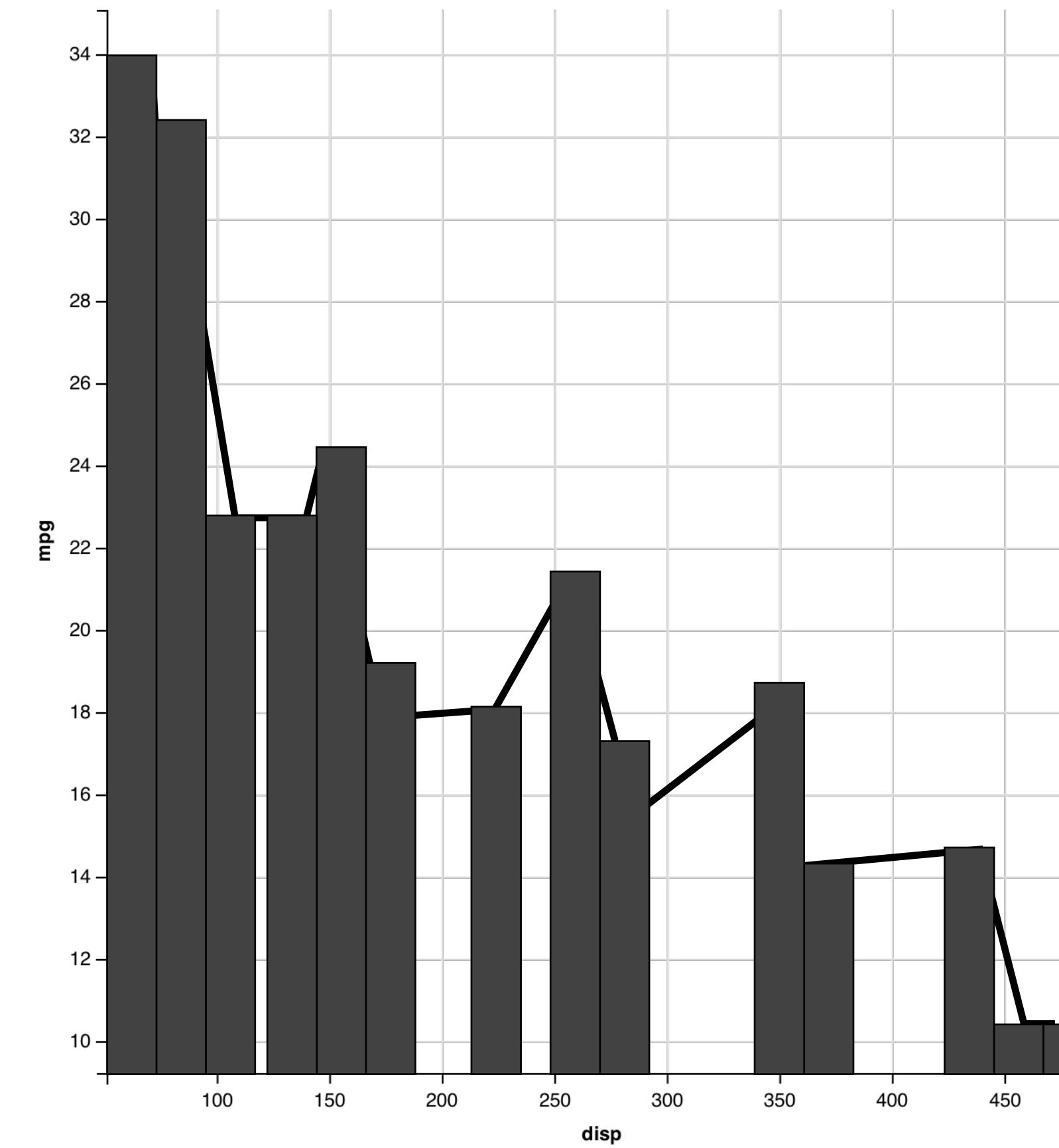


mappings

	y	x		
	mpg	cyl	disp	hp
1	21.0	6	160.0	2
2	21.0	6	160.0	2
3	22.8	4	108.0	1
4	21.4	6	258.0	2
5	18.7	8	360.0	3
6	18.1	6	225.0	2
7	14.3	8	360.0	5
8	24.4	4	146.7	1
9	22.8	4	140.8	1
10	19.2	6	167.6	2
11	17.8	6	167.6	2
12	16.4	8	275.8	3
13	17.3	8	275.8	3
14	15.2	8	275.8	3
15	10.4	8	472.0	4
16	10.4	8	460.0	4
17	14.7	8	440.0	4
18	32.4	4	78.7	1
19	30.4	4	75.7	1
20	33.9	4	71.1	1

data

geom
points
lines
bars

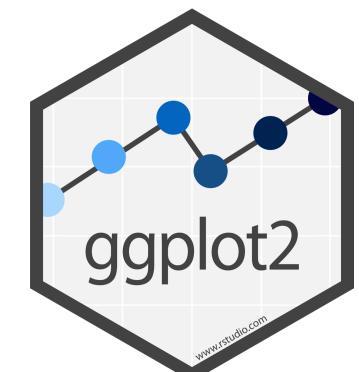
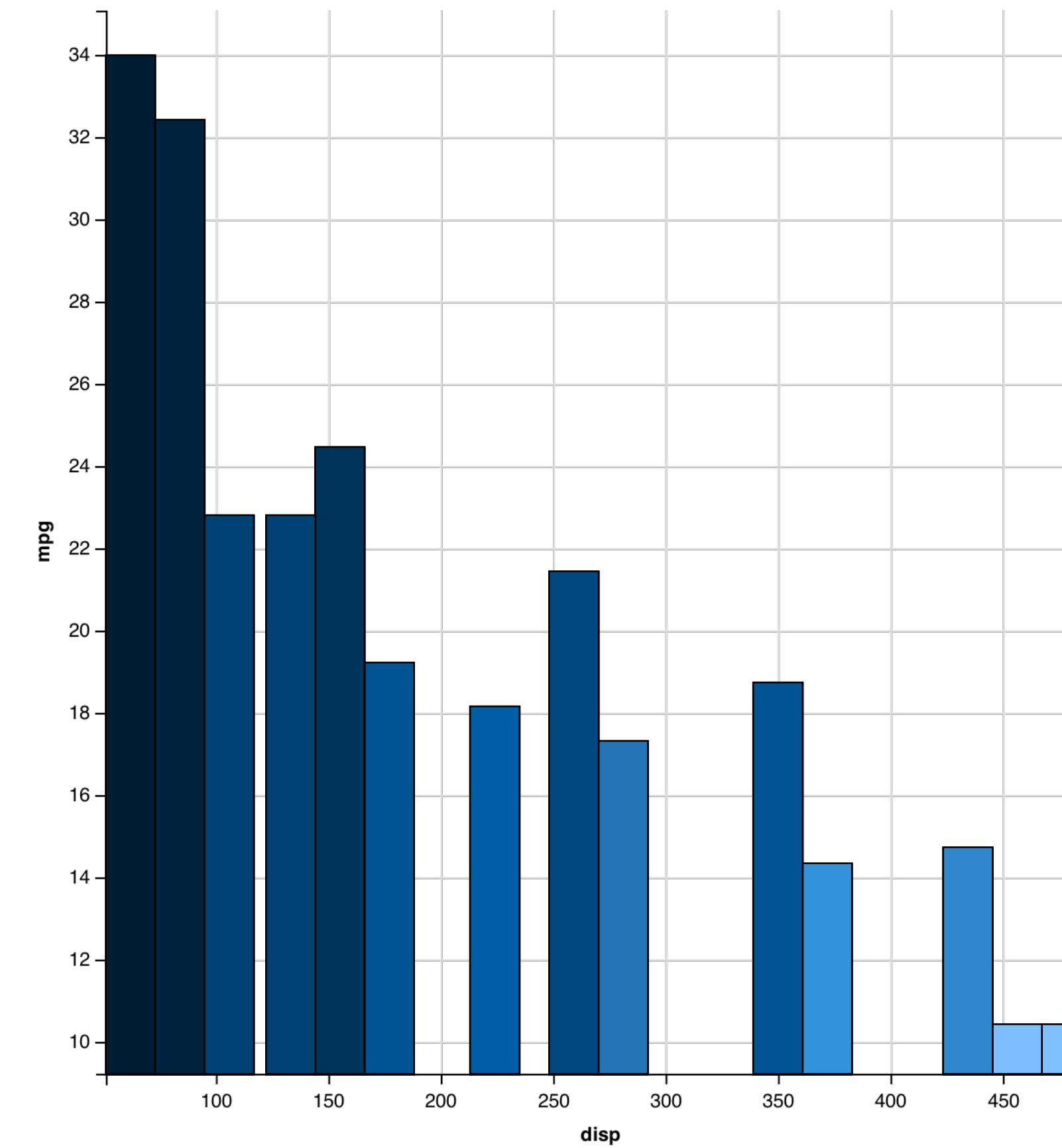


mappings

	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

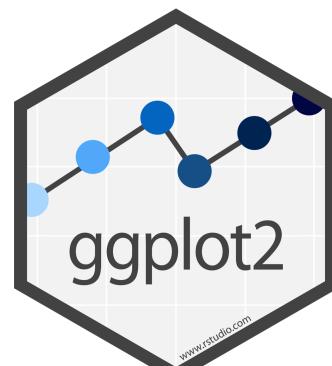
geom
points
lines
bars



To make a graph

[template]

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



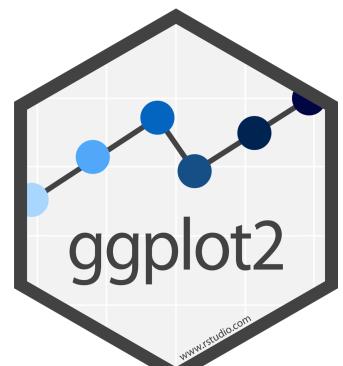
To make a graph

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>) +  
<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	●

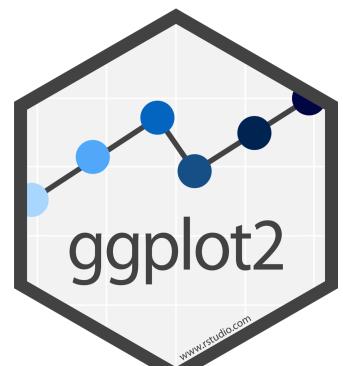
data

geom

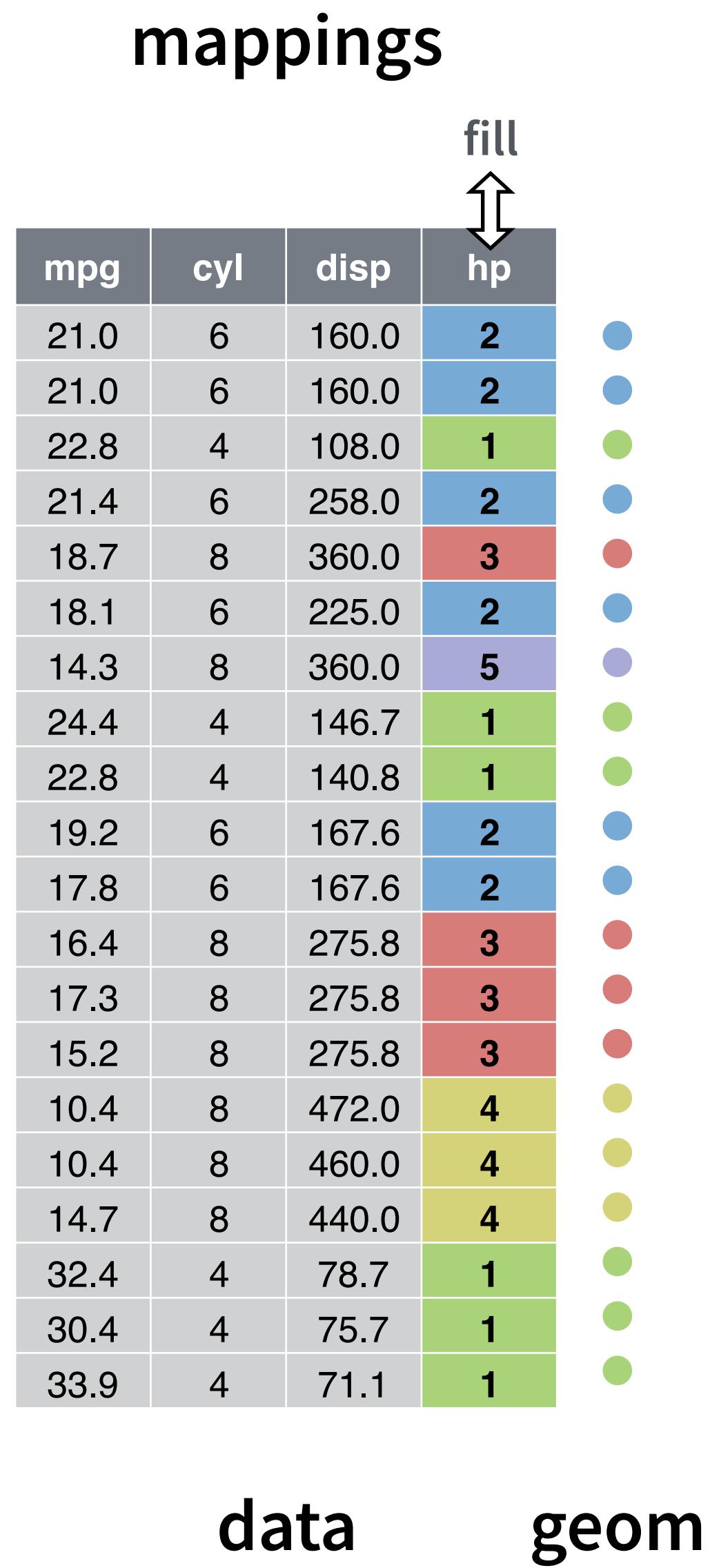
1. Pick a **data** set

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

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



To make a graph

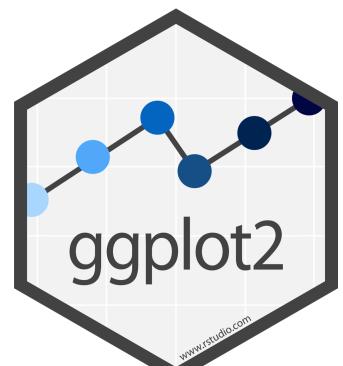


1. Pick a **data** set

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

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

3. **Map** aesthetic
properties to
variables

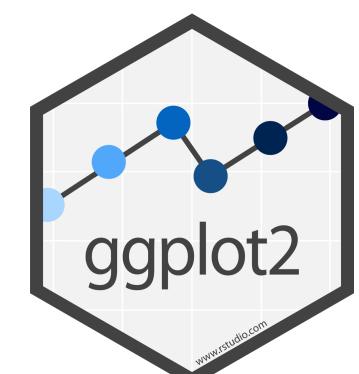
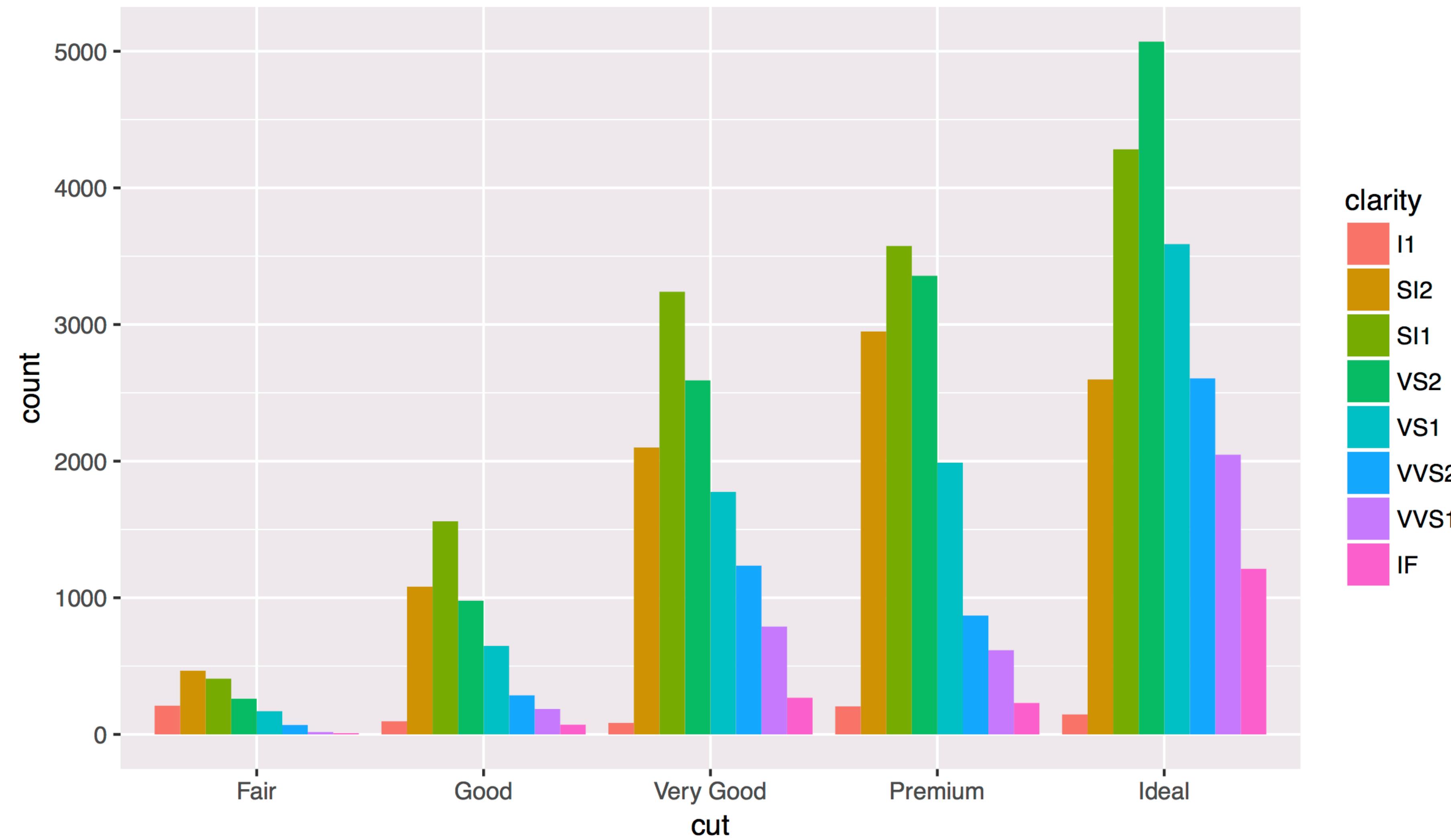


what else?

R

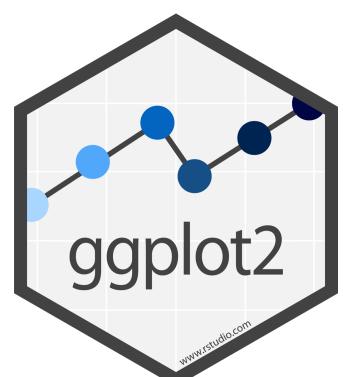
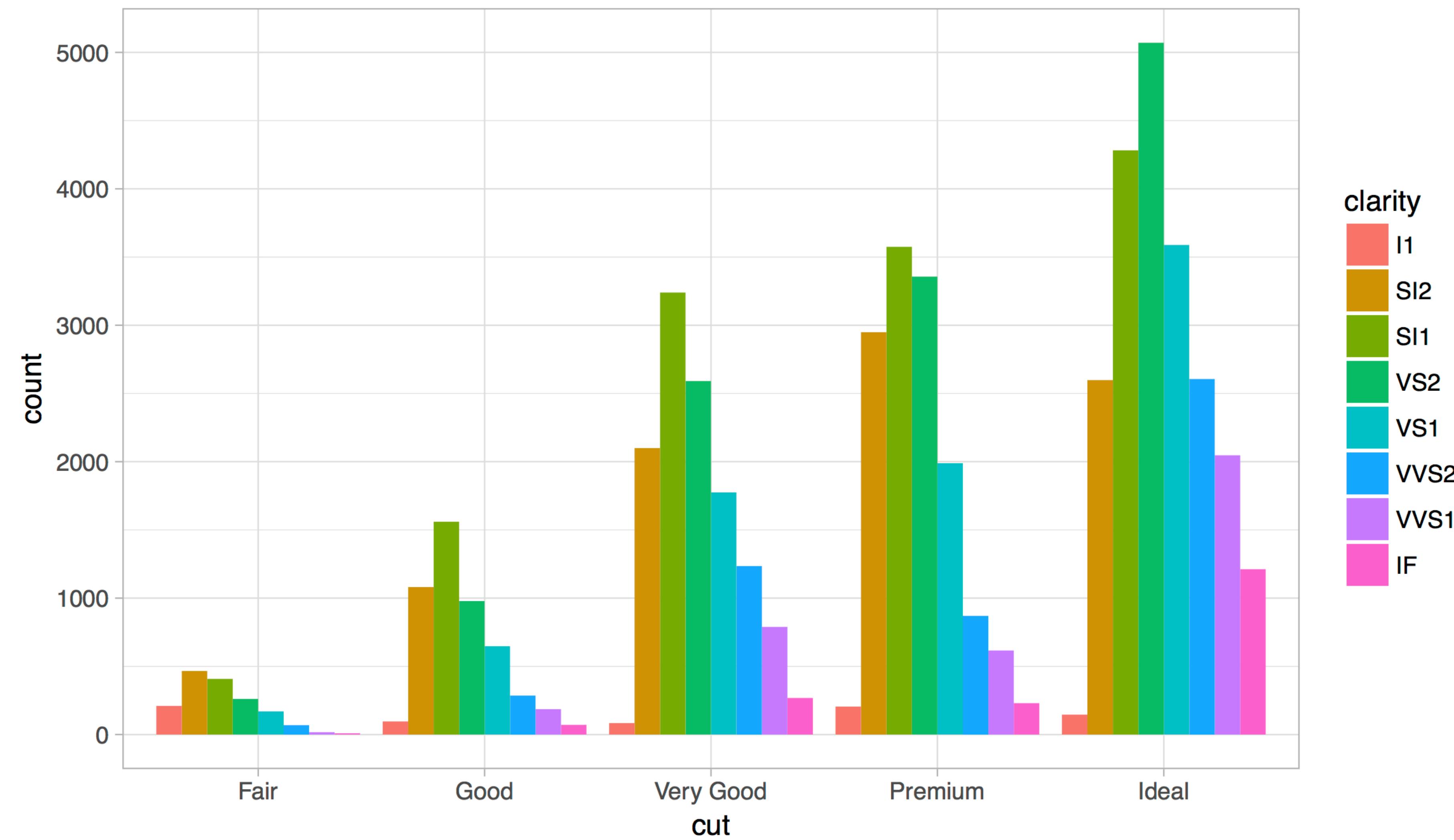
Position Adjustments

How overlapping objects are arranged



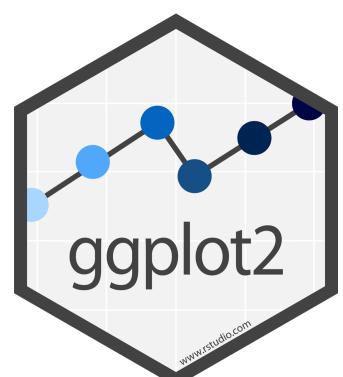
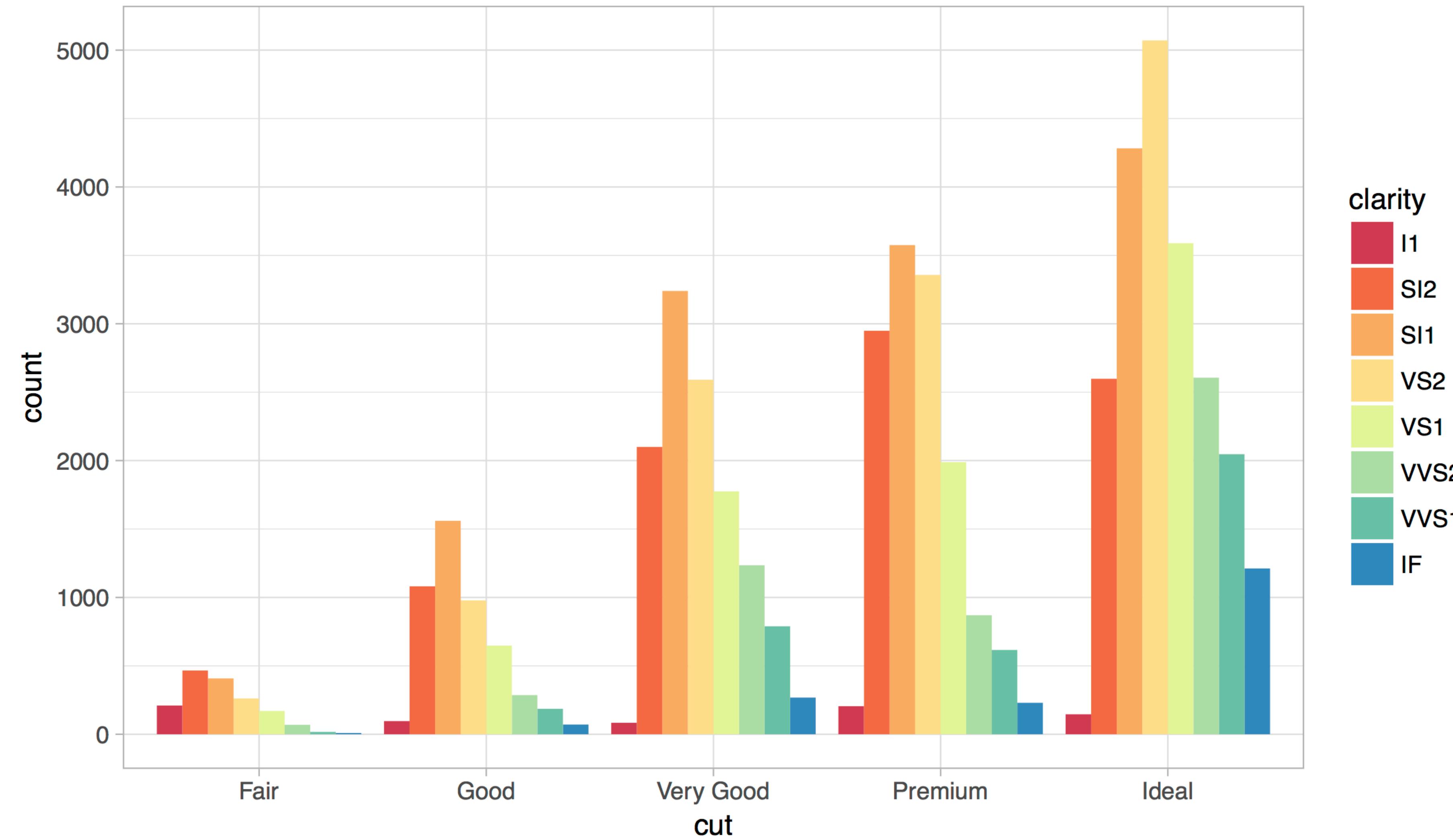
Themes

Visual appearance of non-data elements



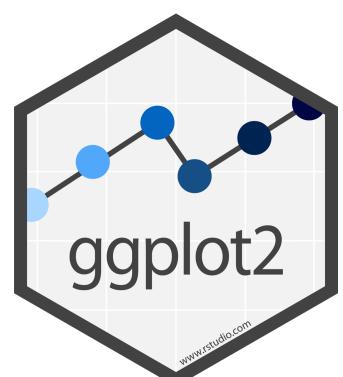
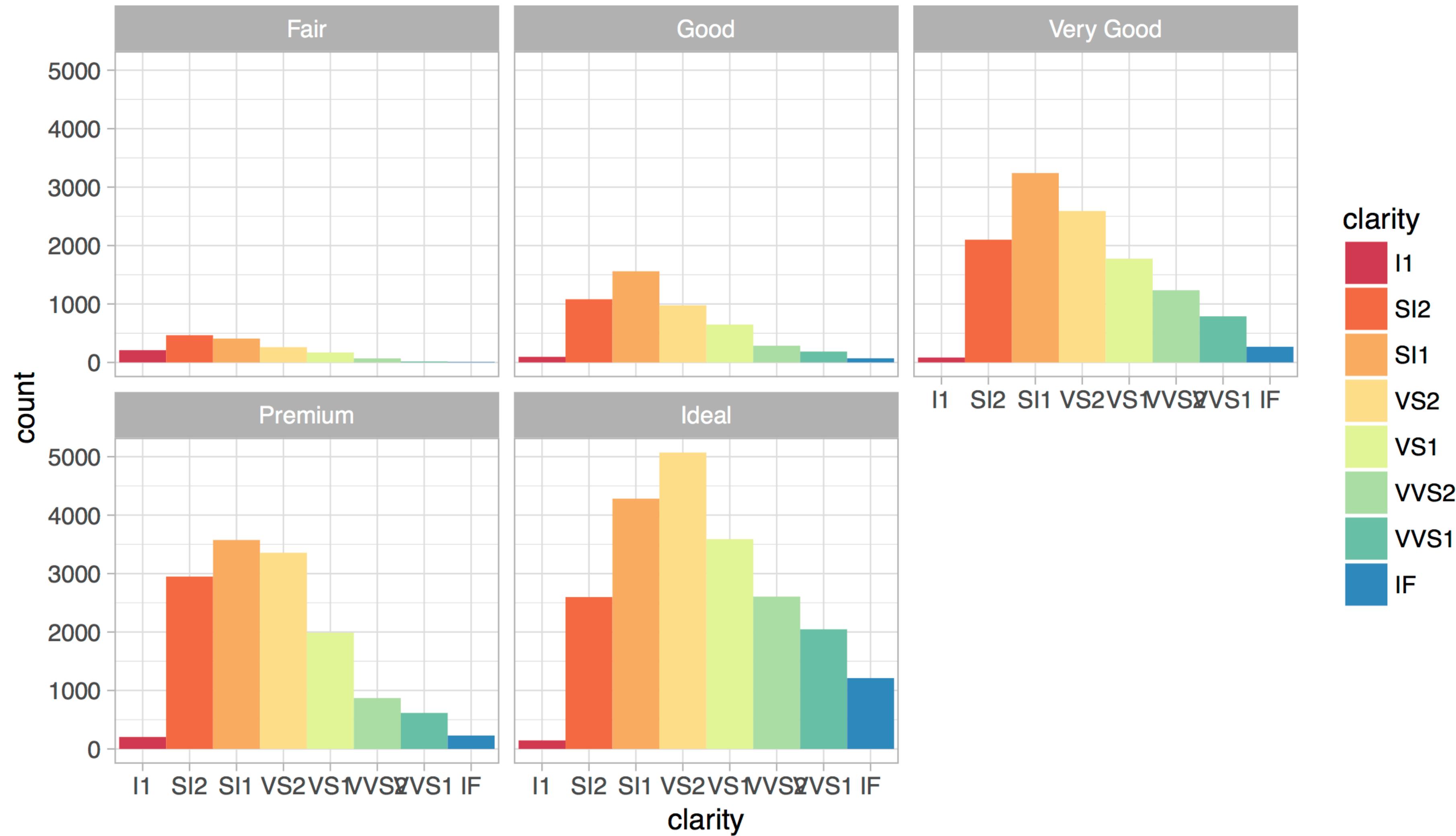
Scales

Customize color scales, other mappings

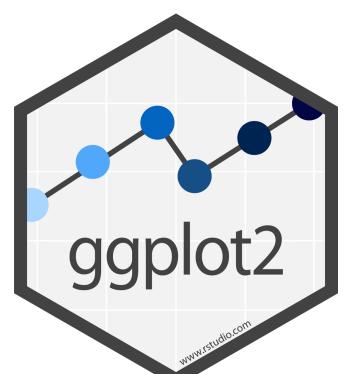


Facets

Subplots that display subsets of the data.



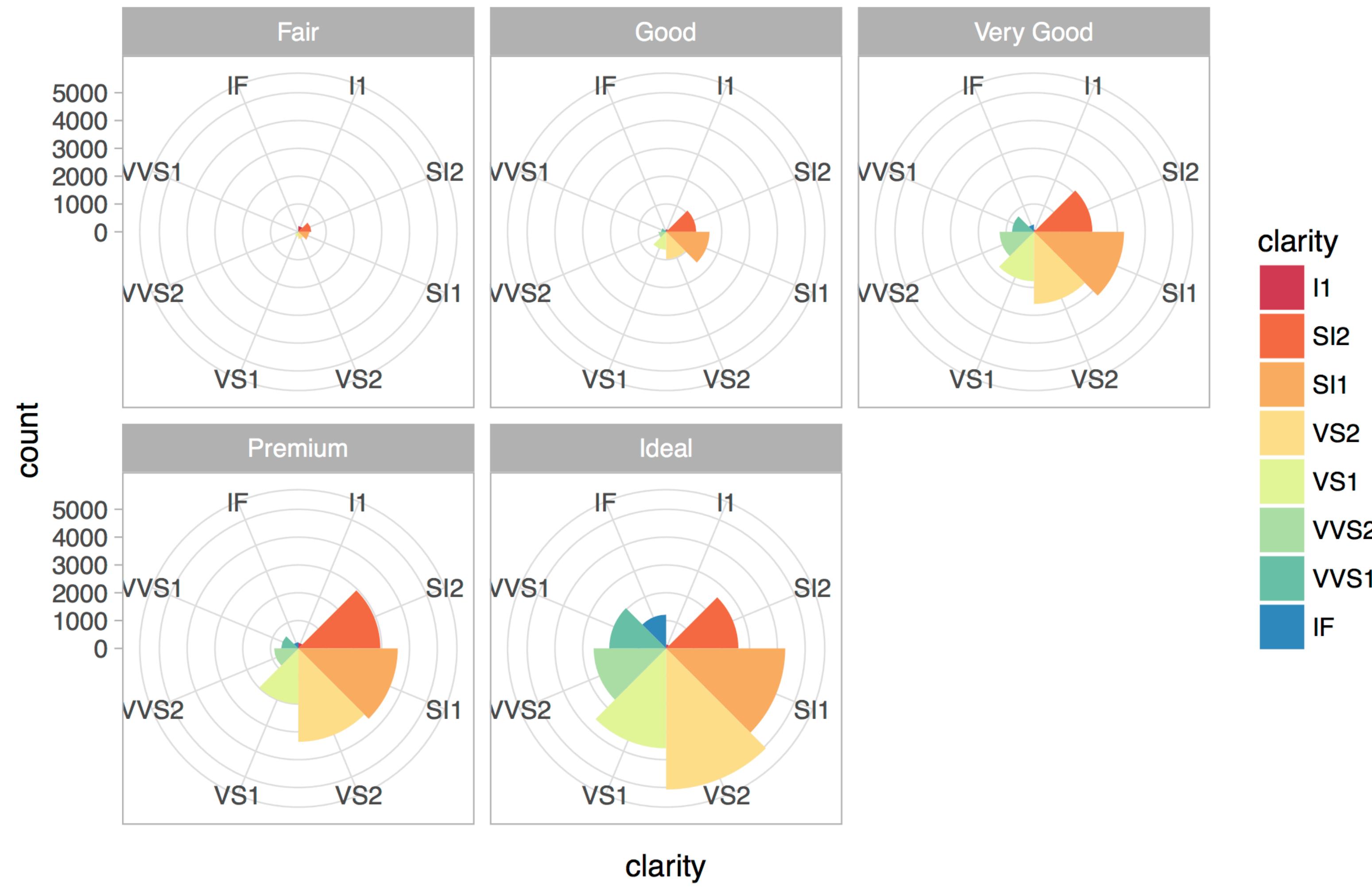
Coordinate systems



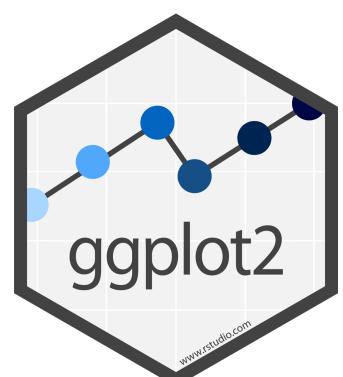
Titles and captions

Diamonds data

The data set is skewed towards ideal cut diamonds



Data by Hadley Wickham

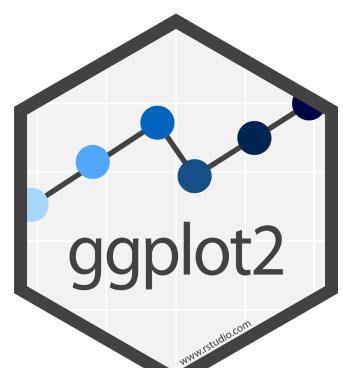
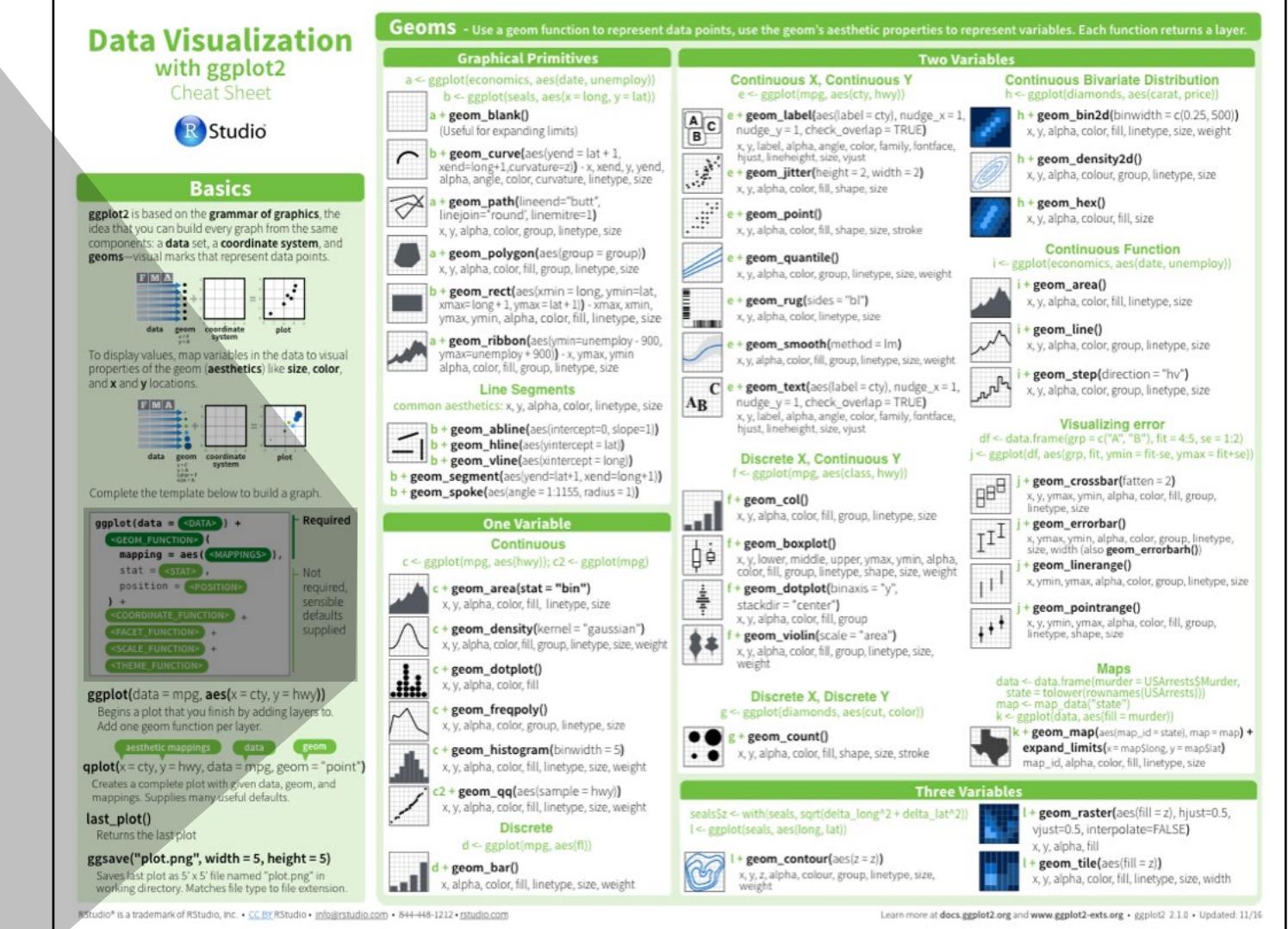


A ggplot2 template

Make any plot by filling in the parameters of this template

```
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



ggplot2.tidyverse.org

The screenshot shows a web browser window displaying the ggplot2.tidyverse.org website. The title bar reads "Create Elegant Data Visualisati x" and "Garrett". The address bar shows the URL "ggplot2.tidyverse.org". The page itself has a header with the ggplot2 logo and the text "part of the tidyverse". It features a main content area with a section titled "Usage" containing R code examples and a plot, and a sidebar with "Links" and "License" sections.

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 showing fuel efficiency (mpg) on the y-axis versus engine displacement (displ) on the x-axis. The plot includes a legend indicating that purple dots represent the "2seater" class.

Links

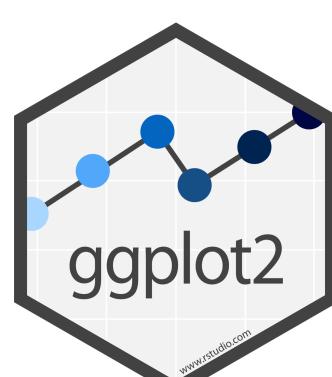
- Download from CRAN at <https://cran.r-project.org/package=ggplot2>
- Browse source code at <https://github.com/tidyverse/ggplot2>
- Report a bug at <https://github.com/tidyverse/ggplot2/issues>
- Learn more at <http://r4ds.had.co.nz/data-visualisation.html>

License

GPL-2 | file [LICENSE](#)

Developers

Hadley Wickham
Author, maintainer



Visualize Data with

