Programming for Data Analytics

4. ggplot2

Dr. Jim Duggan,

School of Engineering & Informatics

National University of Ireland Galway.

https://twitter.com/_jimduggan

Course Overview

Lectures l-3	**R Fundamentals** *Atomic Vectors – Functions – Lists – Matrices – Data Frames*
Lectures 4-9	**Data Science with R** ***ggplot2*** *– dplyr – tidyr – stringr – lubridate - purrr*
Lectures 10-11	**Advanced Programming with R** *Environments – Closures – S3 Object System*
Lectures 12	**Machine Learning with R – Case Studies** *Electricity Generation, Health*

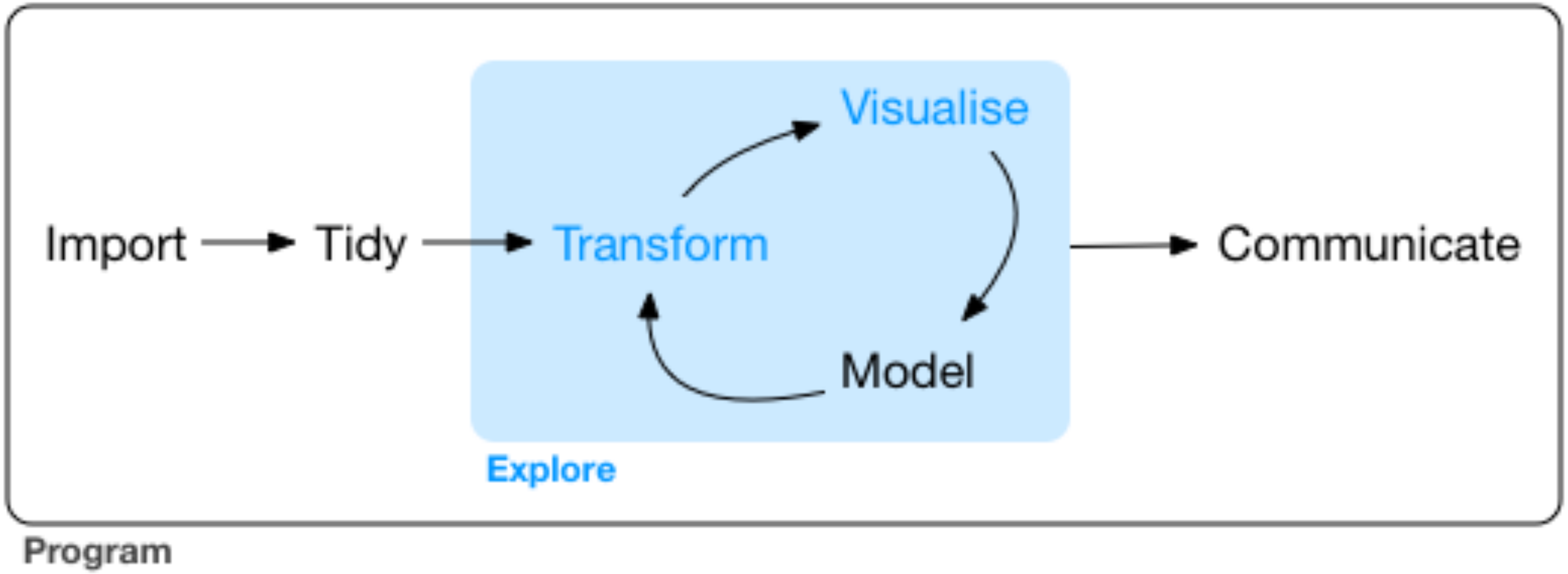
Lecture Overview

- Data Exploration
- Aesthetic Mappings
- Common Problems
- Facets
- Geometric Objects
- Statistical Transformations
- Coordinate Systems
- Layered Grammar of Graphics

Lectures l-3	**R Fundamentals** *Atomic Vectors – Functions – Lists – Matrices – Data Frames*
Lectures 4-8	**Data Science with R** ***ggplot2*** *– dplyr – tidyr – stringr – lubridate –forcats - purrr*
Lectures 9-10	**Advanced Programming with R** *Environments – Closures – S3 Object System*
Lectures 11-12	**Machine Learning with R – Case Studies** *Electricity Generation, Marketing, Epidemiology*

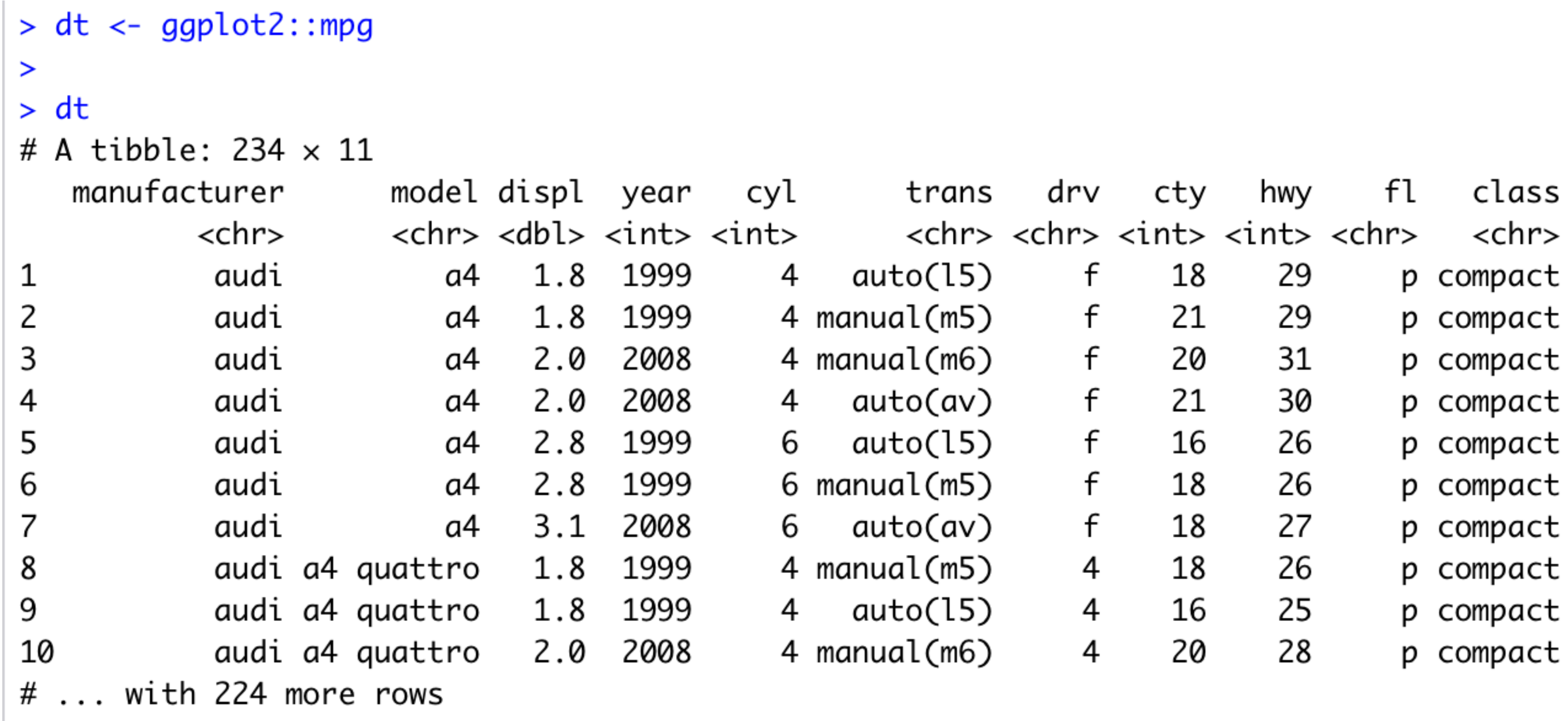
(1) Data Exploration

"Data exploration is the art of looking at your data, rapidly generating hypotheses, quickly testing them, then repeating again and again and again." (Wickham and Grolemund 2017).



Data Visualisation with **ggplot2**

> “The simple graph has brought more information to the data analyst’s mind that any other device.” – John Tukey



```
> dt <- ggplot2::mpg
>
> dt
# A tibble: 234 × 11
   manufacturer      model displ  year   cyl      trans   drv   cty   hwy    fl    class
          <chr>      <chr> <dbl> <int> <int>      <chr> <chr> <int> <int> <chr>    <chr>
1          audi         a4   1.8  1999     4   auto(l5)     f    18    29     p  compact
2          audi         a4   1.8  1999     4 manual(m5)     f    21    29     p  compact
3          audi         a4   2.0  2008     4 manual(m6)     f    20    31     p  compact
4          audi         a4   2.0  2008     4   auto(av)     f    21    30     p  compact
5          audi         a4   2.8  1999     6   auto(l5)     f    16    26     p  compact
6          audi         a4   2.8  1999     6 manual(m5)     f    18    26     p  compact
7          audi         a4   3.1  2008     6   auto(av)     f    18    27     p  compact
8          audi a4 quattro   1.8  1999     4 manual(m5)     4    18    26     p  compact
9          audi a4 quattro   1.8  1999     4   auto(l5)     4    16    25     p  compact
10         audi a4 quattro   2.0  2008     4 manual(m6)     4    20    28     p  compact
# ... with 224 more rows
```

Fuel Economy Data Set (ggplot2::mpg)

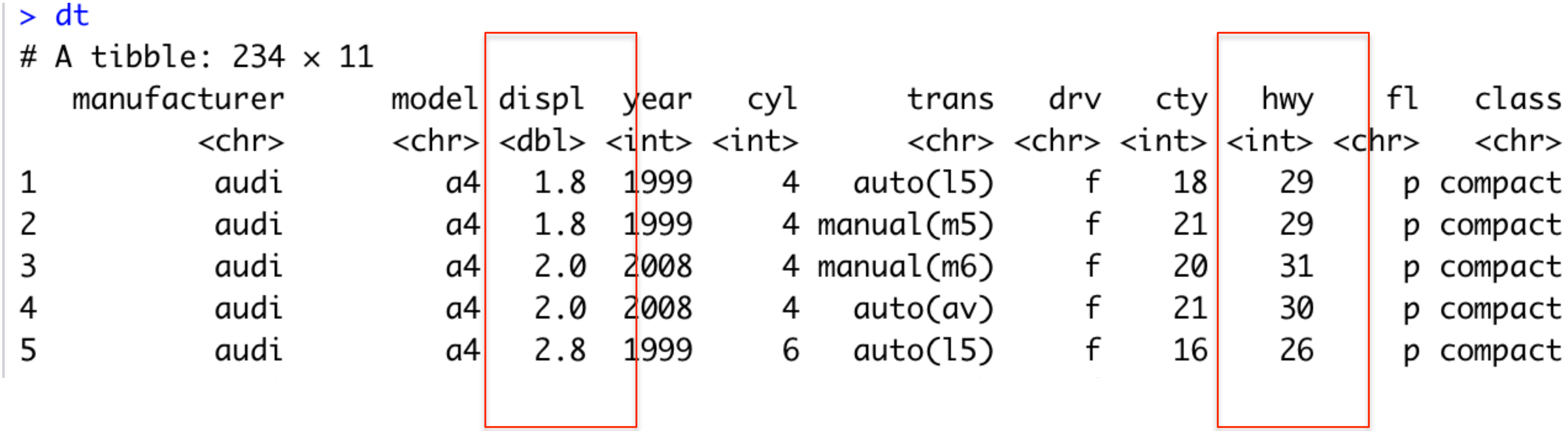
This dataset contains a subset of the fuel economy data that the EPA makes available on http://fueleconomy.gov. It contains only models which had a new release every year between 1999 and 2008 - this was used as a proxy for the popularity of the car.

manufacturer	manufacturer	**drv**	f = front-wheel drive, r = rear wheel drive, 4 = 4wd
model	model name	**cty**	city miles per gallon
displ	engine displacement, in litres	**hwy**	highway miles per gallon
year	year of manufacture	**fl**	fuel type
cyl	number of cylinders	**class**	“type” of car
trans	type of transmission		

First Steps

- Generate a first graph to help answer the following question:
  - *Do cars with big engines use more fuel than cars with small engines*
- What might the relationship between engine size and fuel efficiency look like?
  - Positive or negative?
  - Linear or non-linear?

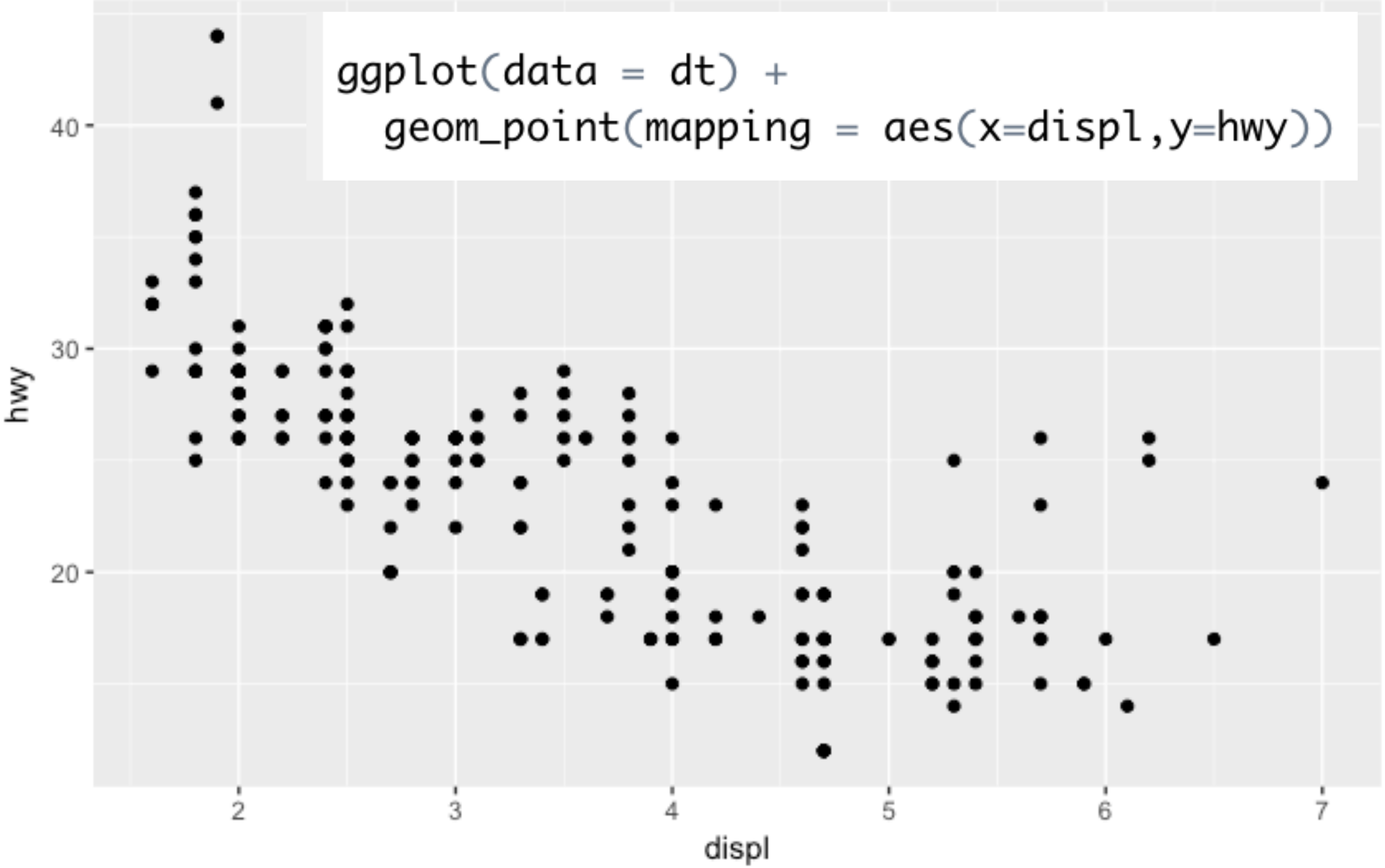
Selecting data



```
> dt
# A tibble: 234 × 11
   manufacturer      model displ  year   cyl      trans   drv   cty   hwy    fl   class
          <chr>      <chr> <dbl> <int> <int>      <chr> <chr> <int> <int> <chr>   <chr>
1          audi         a4   1.8  1999     4   auto(l5)     f    18    29     p compact
2          audi         a4   1.8  1999     4 manual(m5)     f    21    29     p compact
3          audi         a4   2.0  2008     4 manual(m6)     f    20    31     p compact
4          audi         a4   2.0  2008     4   auto(av)     f    21    30     p compact
5          audi         a4   2.8  1999     6   auto(l5)     f    16    26     p compact
```

- Among the variables are:
  - **displ**, a car’s engine size in litres
  - **hwy**, a car’s fuel efficiency on the highway in miles per gallon

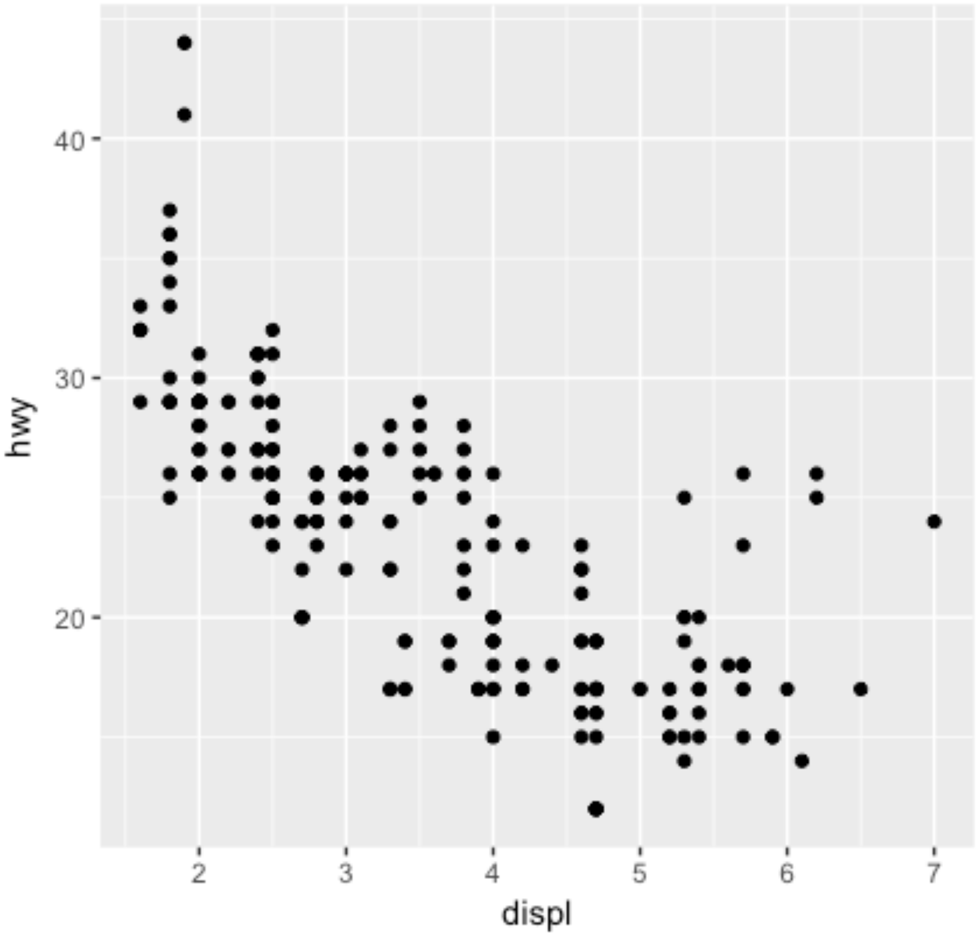
Creating a ggplot



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

Interpreting the plot

- The plot shows a negative relationship between engine size (displ) and fuel efficiency (hwy)
- Cars with big engines use more fuel
- Does this confirm or refute your hypothesis about fuel efficiency and engine size?

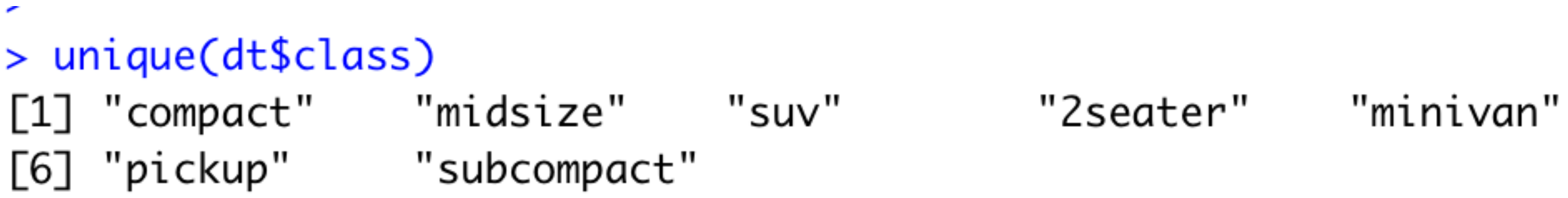


Challenge 4.1

- Explore the hypothesis that city driving is less fuel efficient that highway driving
- Use ggplot to present the points on the same graph, and colour each data set differently
- Does the data confirm or refute your initial hypothesis?

(2) Aesthetic Mappings

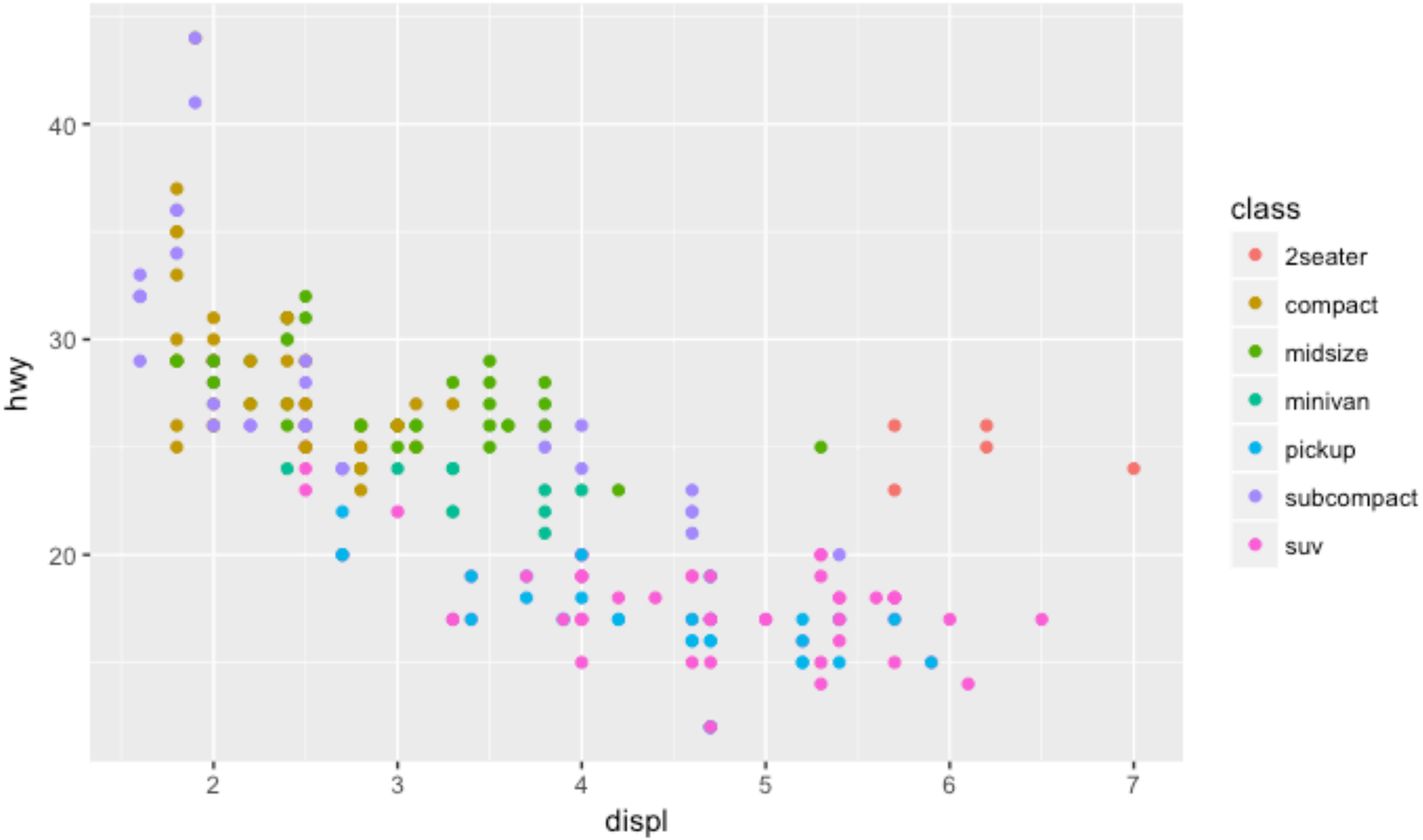
> "The greatest value of a picture is when it forces us to notice what we never expected to see" – John Tukey



```
> unique(dt$class)
[1] "compact"    "midsize"    "suv"        "2seater"    "minivan"
[6] "pickup"     "subcompact"
```

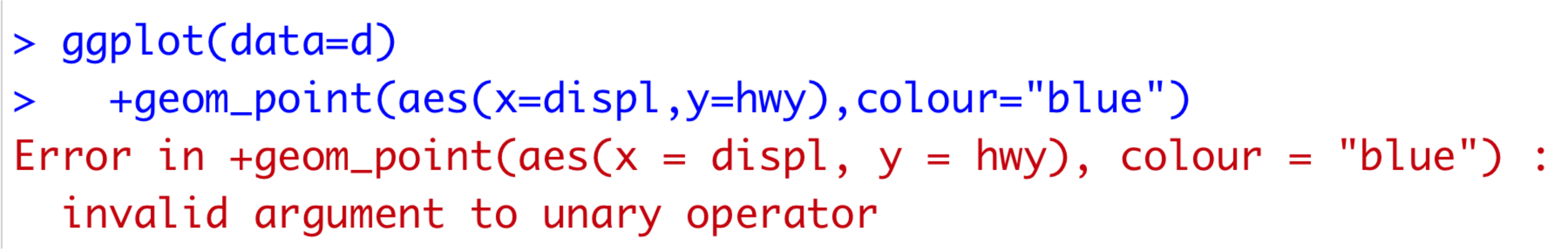
- A third variable can be added to a 2-D plot by mapping it to an aesthetic.
- An aesthetic is a visual property of the plot's objects.
- An aesthetic's *level* could be colour, size or shape.

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



(3) Common Problems

- R can be “extremely picky, and a misplaced character can make all the difference”
- Make sure every (is matched with a)
- For ggplot calls, the + must come at the end of the line, not at the start (see below)
- You can get help about any function by running ?function_name

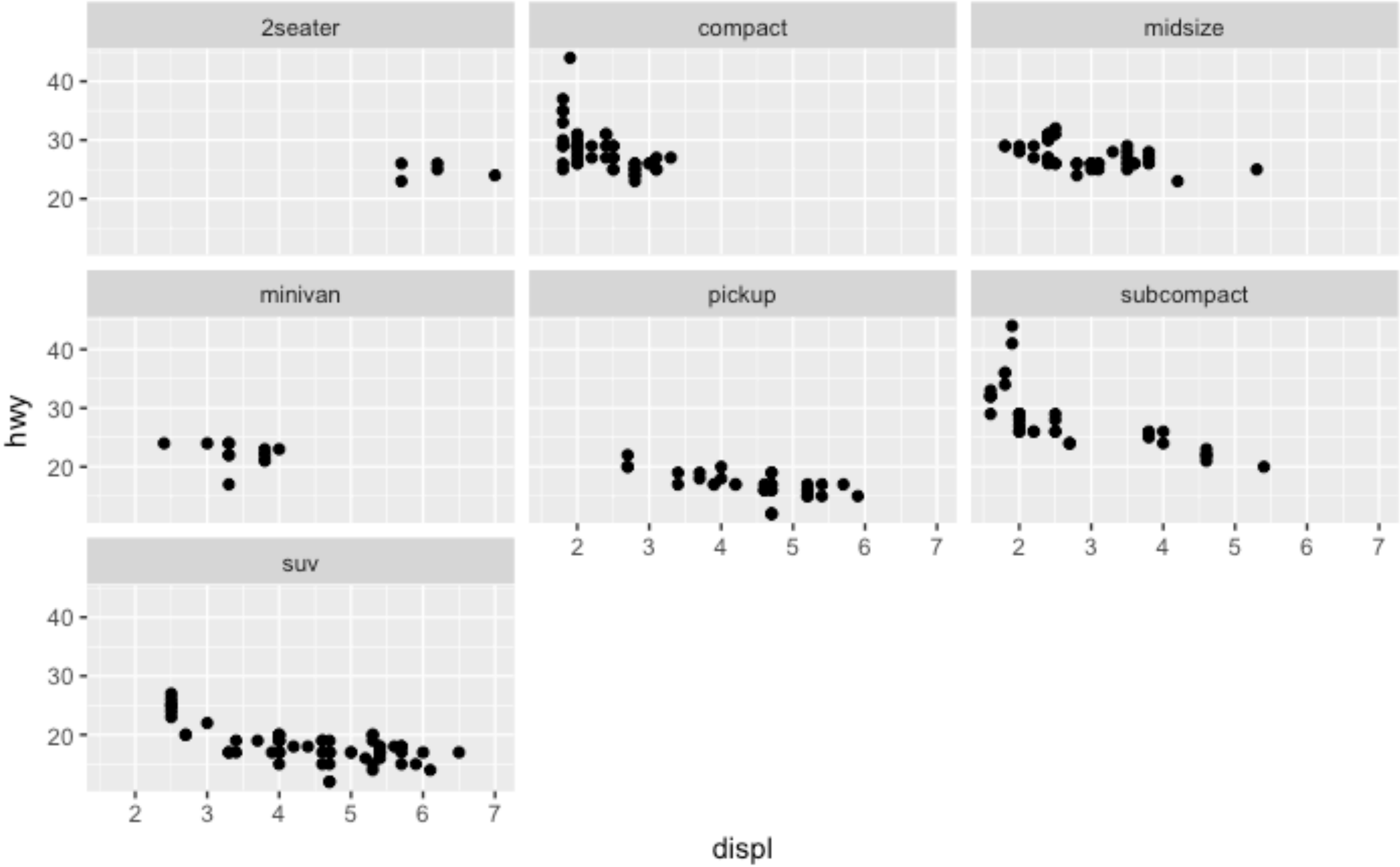


```
> ggplot(data=d)
>   +geom_point(aes(x=displ,y=hwy),colour="blue")
Error in +geom_point(aes(x = displ, y = hwy), colour = "blue") :
  invalid argument to unary operator
```

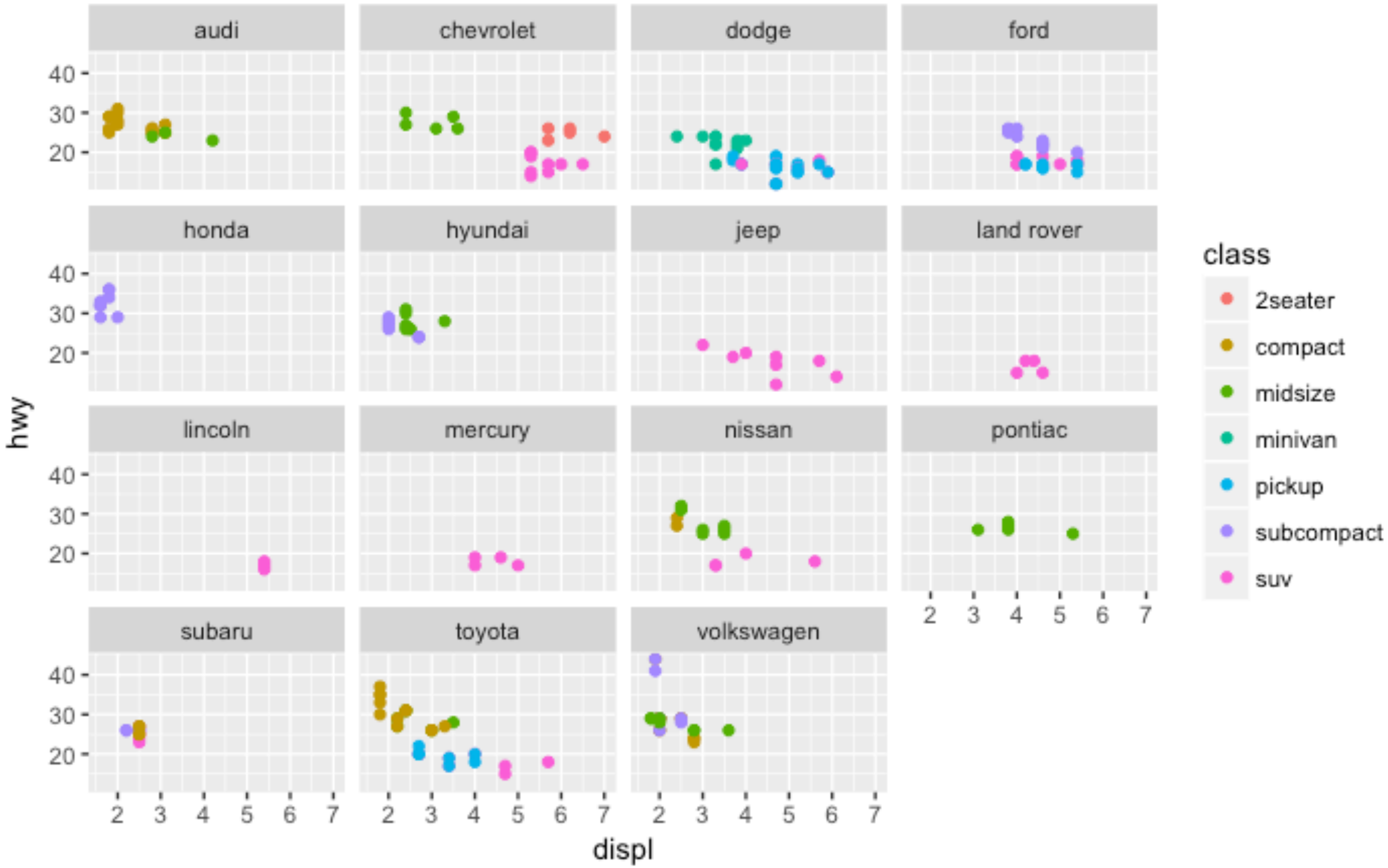
(4) Facets

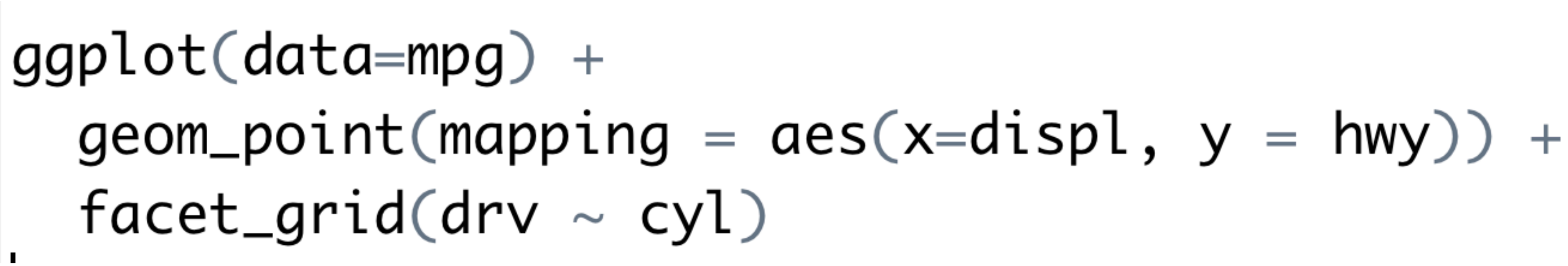
- Another way to add categorical variables is to split a plot into facets, subplots that display one subset of the data.
- To facet your plot by a single variable, use facet_wrap(), with ~ followed by the variable name
- To facet on the combination of two variables, used facet_grid()

```
ggplot(data = dt) +
  geom_point(mapping = aes(x=displ,y=hwy)) + facet_wrap(~class)
```

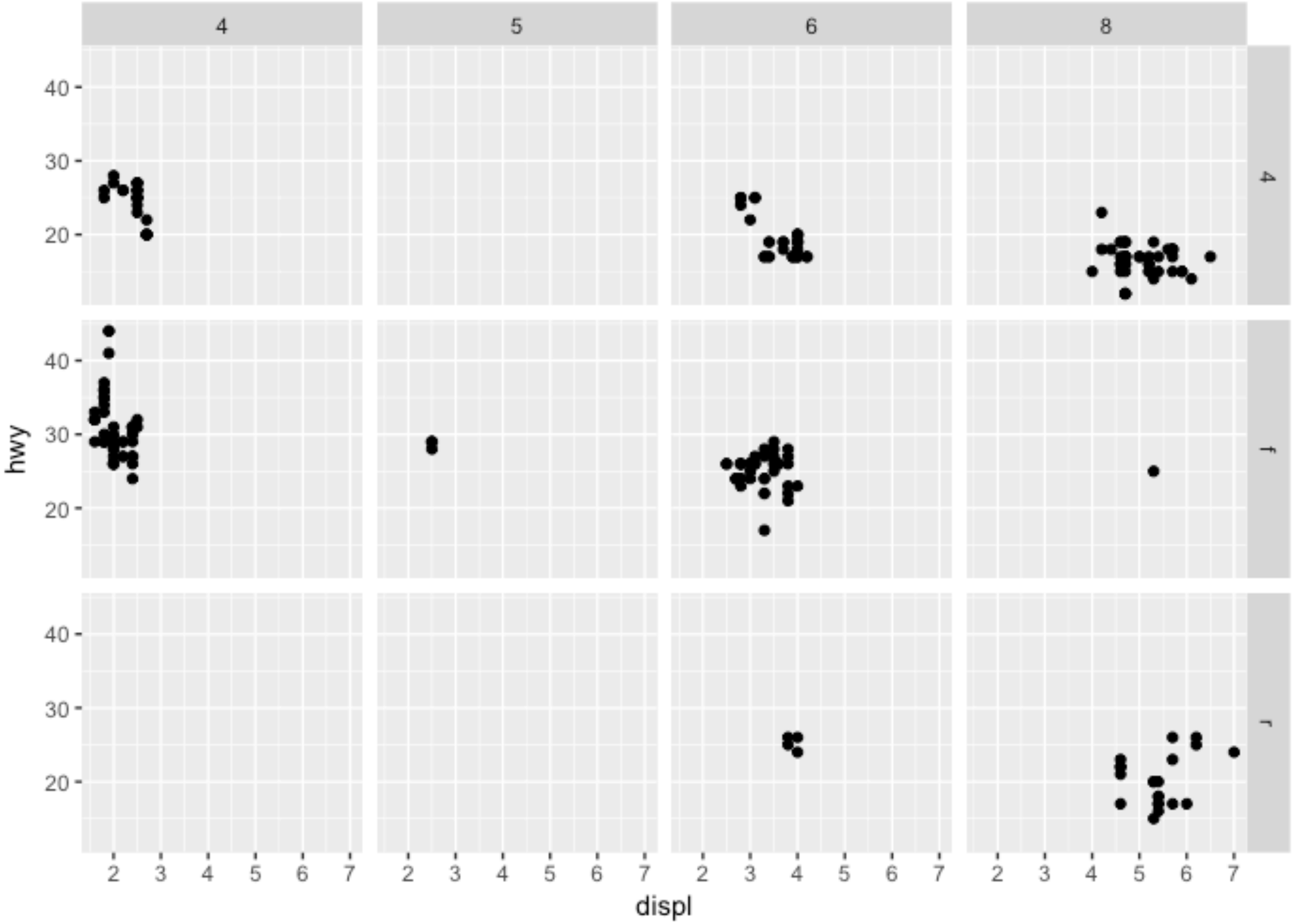


```
ggplot(data = dt) +
  geom_point(mapping = aes(x=displ,y=hwy,colour=class)) +
  facet_wrap(~manufacturer)
```

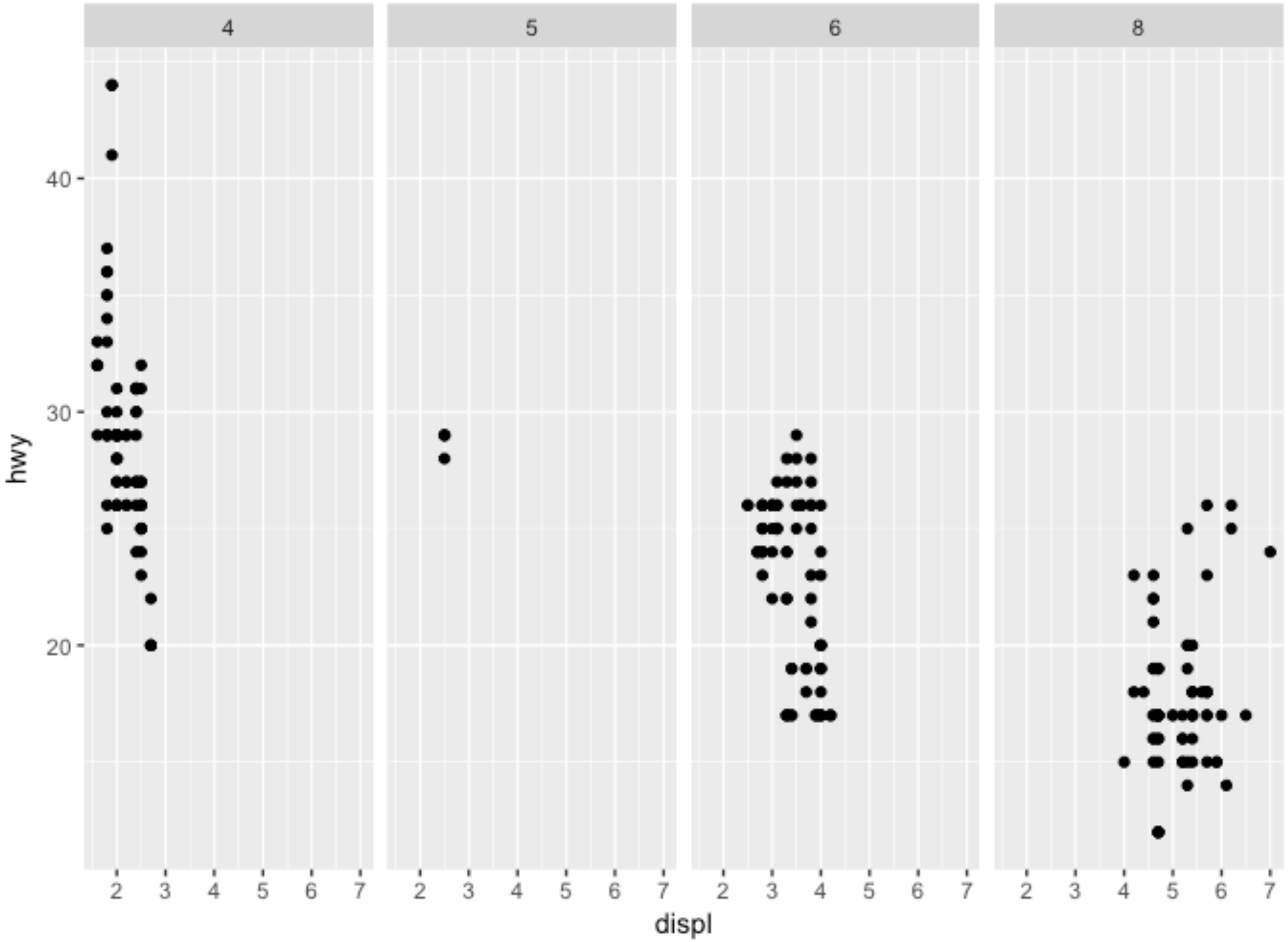




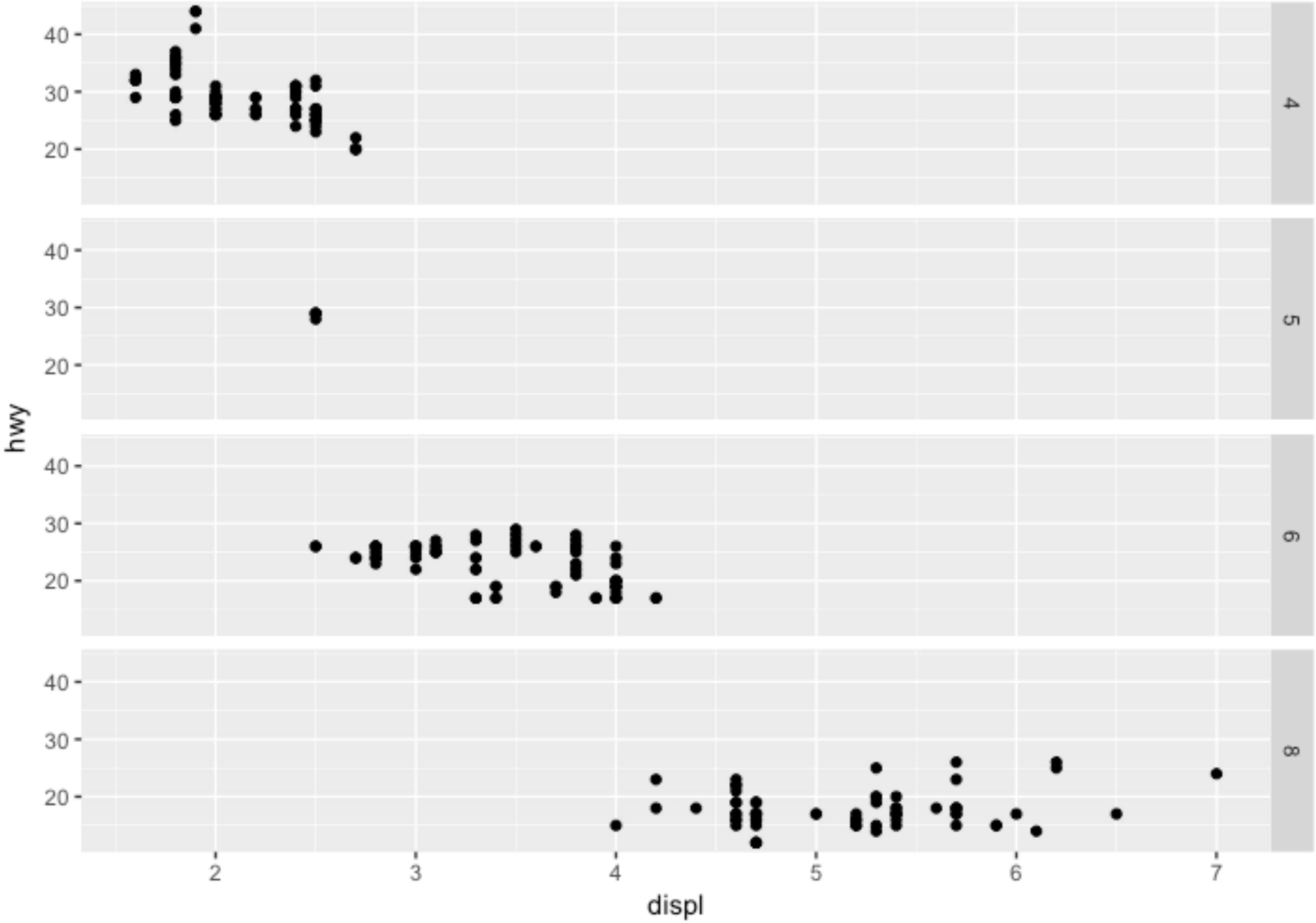
```
ggplot(data=mpg) +
  geom_point(mapping = aes(x=displ, y = hwy)) +
  facet_grid(drv ~ cyl)
```



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



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

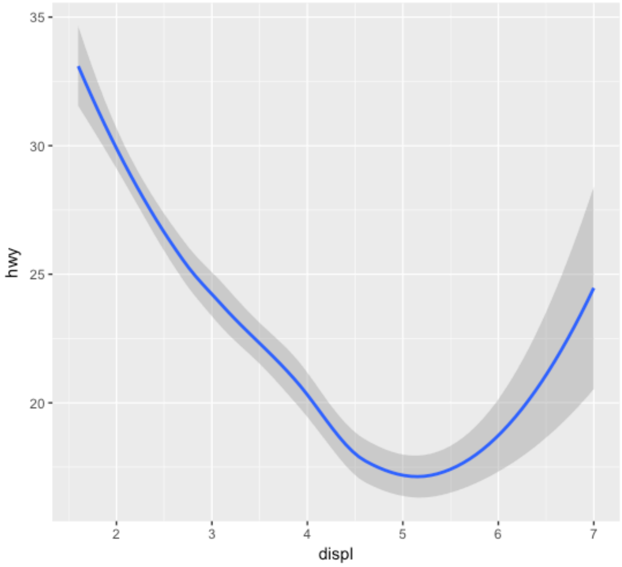


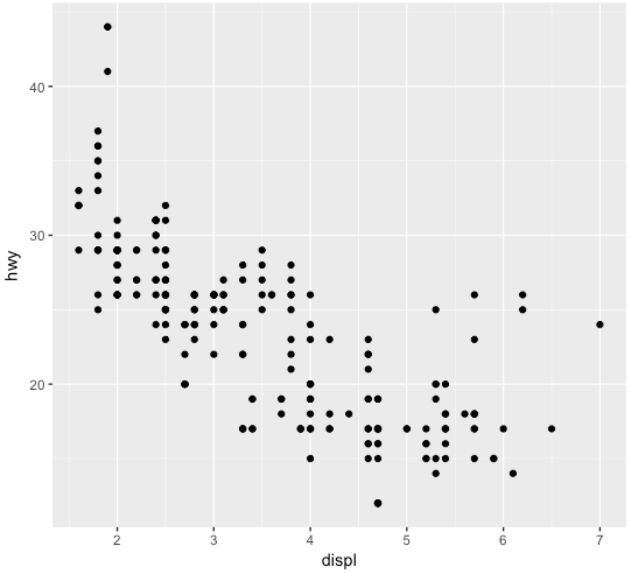
Challenge 4.2

- When using facet_grid() you should usually put the variable with more unique levels in the columns. Why?

(5) Geometric Objects

- Both of these plots contain the same x and y variable, and describe the same data
- The plots are not identical, they use a different visual object to represent the data
- In ggplot2 syntax, we say the use different ***geoms***



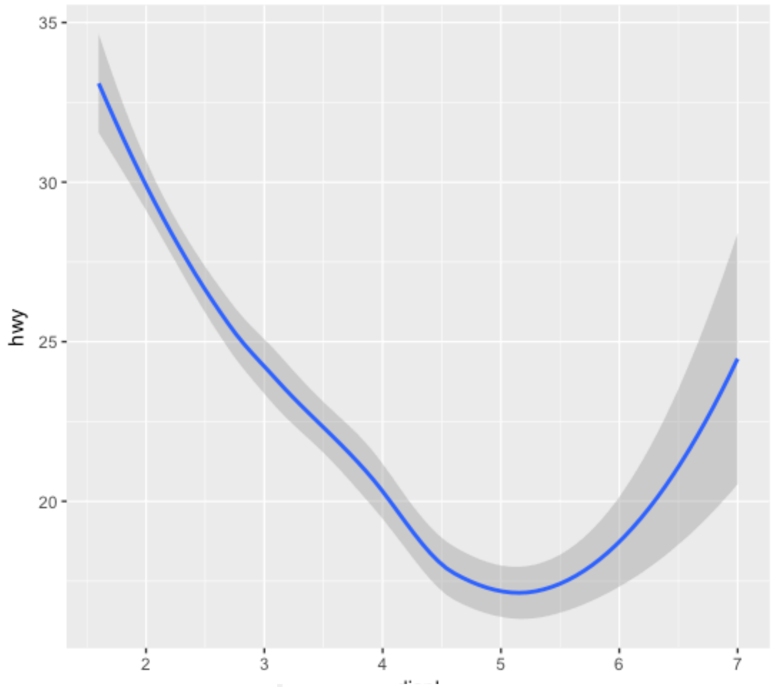


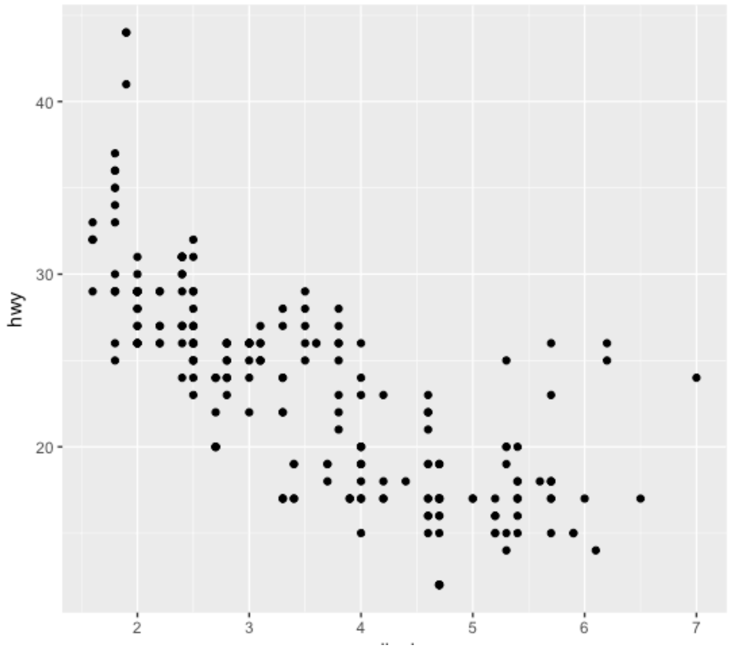
geom

- A geom is a geometrical object that a plot uses to represent data
- Bar charts use bar geoms, line charts use line geoms, and scatter plots use the point geom.
- To change the geom in your plot, simply change the geom function that is added to the ggplot call.

Examples of using different geoms

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

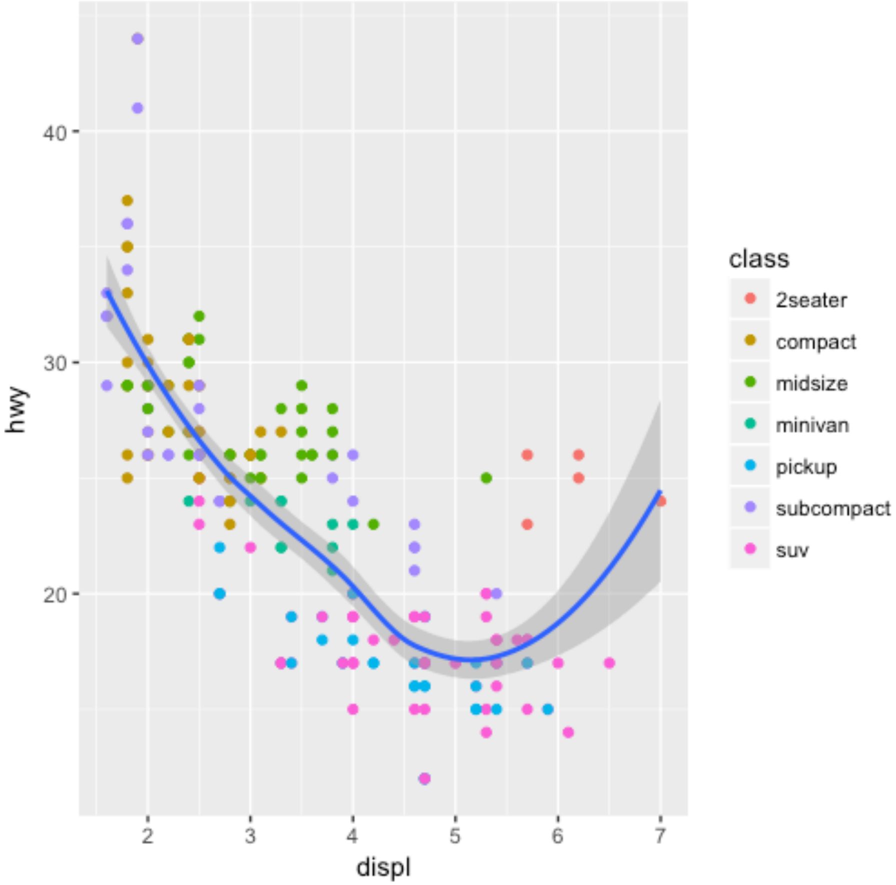


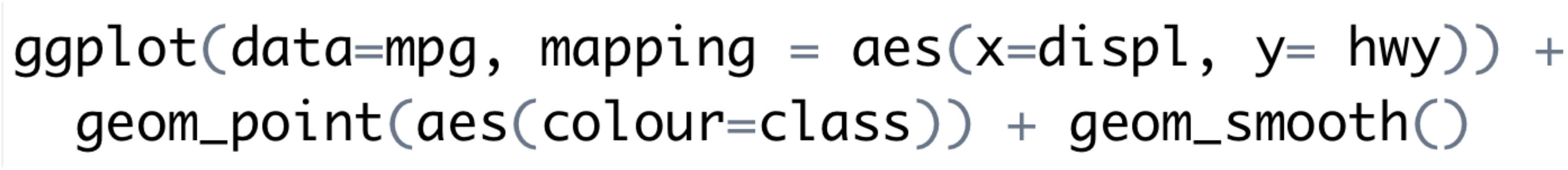


```
ggplot(data=mpg)+
  geom_point(mapping=aes(x=displ,y=hwy))
```

Displaying Multiple geoms

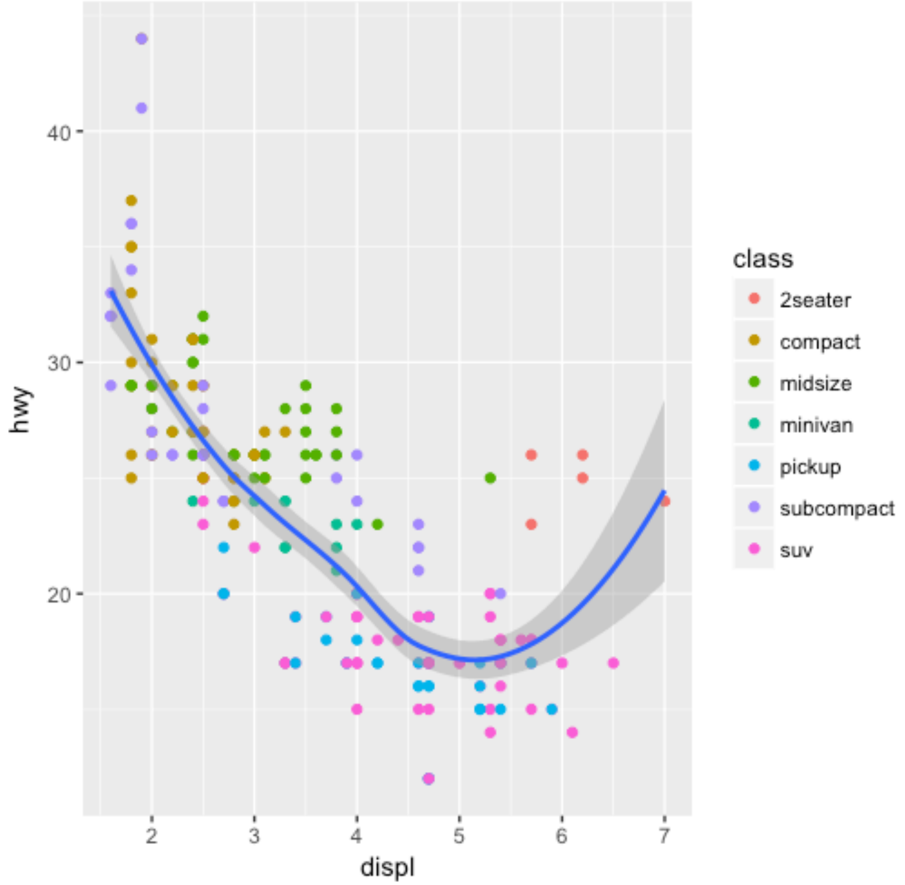
- Multiple geoms can be displayed on the same plot
- Data can be specified in first ggplot() call, and shared by all geoms
- Also, different geoms can have their own data





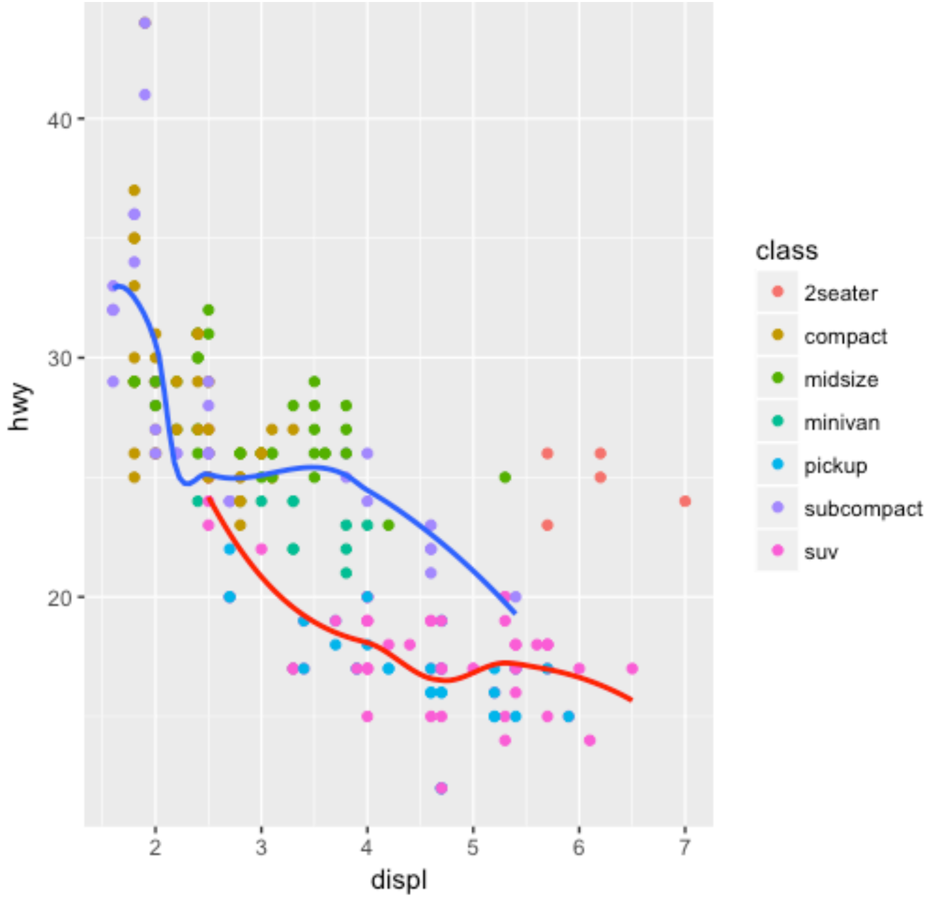
```
ggplot(data=mpg, mapping = aes(x=displ, y= hwy)) +
  geom_point(aes(colour=class)) + geom_smooth()
```

- Data and x,y can be defined in the first call, and then used by the different geoms
- Additional attributes can then be added for geoms (i.e. for specific layers)
- *This makes it possible to display different aesthetics in different layers*



```
ggplot(data=mpg,mapping=aes(x=displ,y=hwy))+
  geom_point(mapping=aes(colour=class))+
  geom_smooth(data=filter(mpg,class=="subcompact"),
              se=F)+
  geom_smooth(data=filter(mpg,class=="suv"),
              se=F,colour="red")
```

- Different data can be specified for each layer
- A local data argument can override a global data argument for a specific layer
- filter() will be explained in a subsequent lecture, it is part of dplyr()



Sample plot geoms

Geom	Purpose
geom_smooth()	Fits a smoother to data and displays the smooth and its standard error
geom_boxplot()	Produces a box-and-whisker plot to summarise the distribution of a set of points

geom_histogram()
geom_freqpoly()
	Shows the distribution of continuous variables

geom_bar()	Shows the distribution of categorical variables

geom_path()
geom_line()
	Draws lines between data points

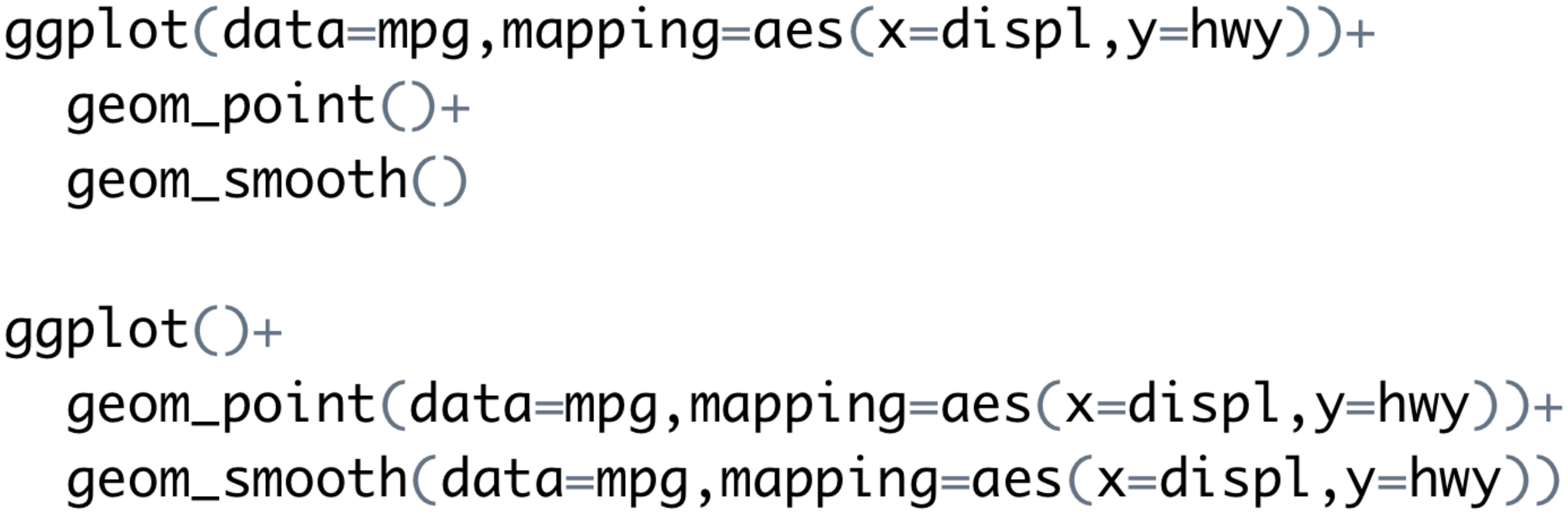
geom_area()	Draws an area plot, which is a line plot filled to the y-axis. Multiple groups will be stacked upon each other

geom_rect()
geom_tile()
geom_raster()
	Draw rectangles

geom_polygon()	Draws polygons, which are filled paths.

Challenge 4.2.2

- Will these two graphs look different. Why/why not?

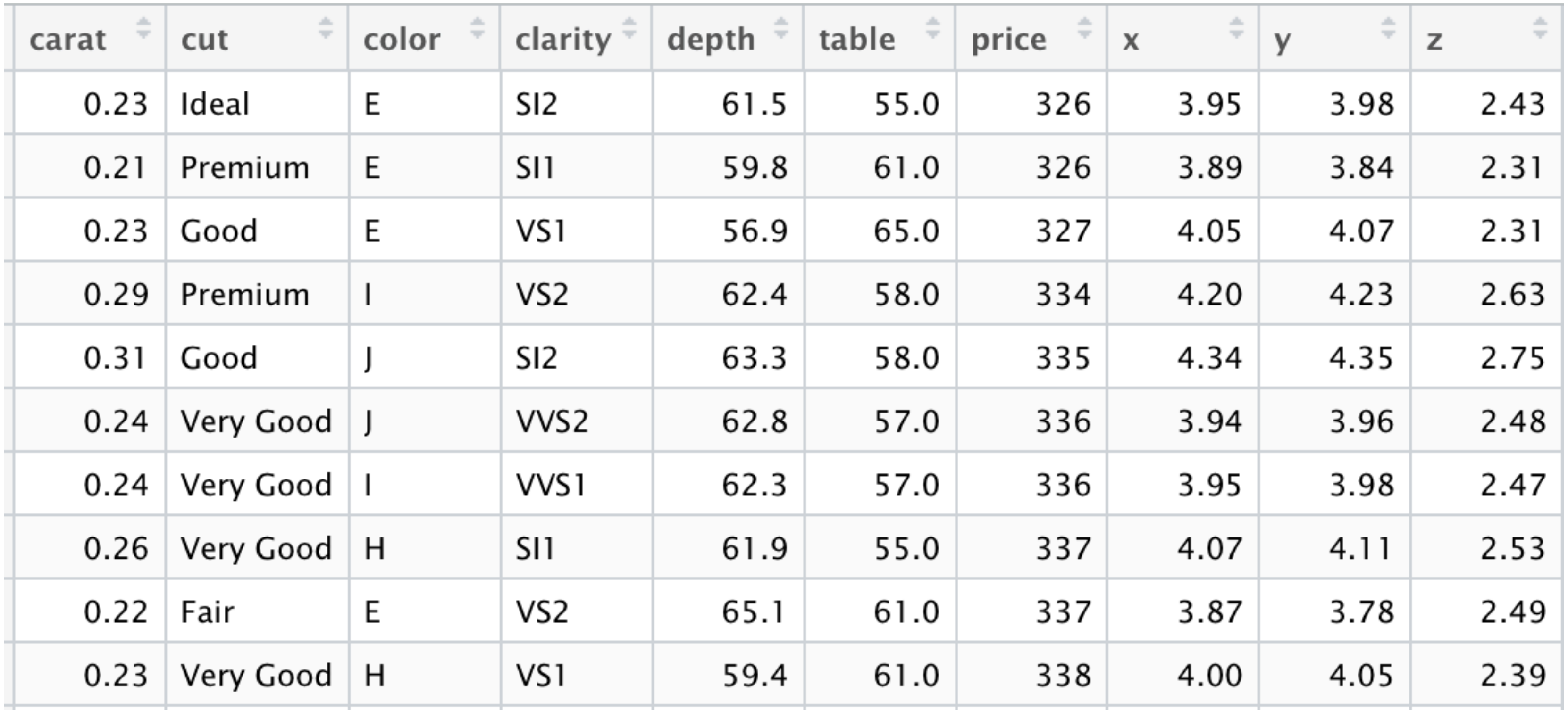


```
ggplot(data=mpg,mapping=aes(x=displ,y=hwy))+
  geom_point()+
  geom_smooth()

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

diamonds data set (ggplot2)

A dataset containing the prices and other attributes of almost 54,000 diamonds.



carat	cut	color	clarity	depth	table	price	x	y	z
0.23	Ideal	E	SI2	61.5	55.0	326	3.95	3.98	2.43
0.21	Premium	E	SI1	59.8	61.0	326	3.89	3.84	2.31
0.23	Good	E	VS1	56.9	65.0	327	4.05	4.07	2.31
0.29	Premium	I	VS2	62.4	58.0	334	4.20	4.23	2.63
0.31	Good	J	SI2	63.3	58.0	335	4.34	4.35	2.75
0.24	Very Good	J	VVS2	62.8	57.0	336	3.94	3.96	2.48
0.24	Very Good	I	VVS1	62.3	57.0	336	3.95	3.98	2.47
0.26	Very Good	H	SI1	61.9	55.0	337	4.07	4.11	2.53
0.22	Fair	E	VS2	65.1	61.0	337	3.87	3.78	2.49
0.23	Very Good	H	VS1	59.4	61.0	338	4.00	4.05	2.39

Explanation of variables

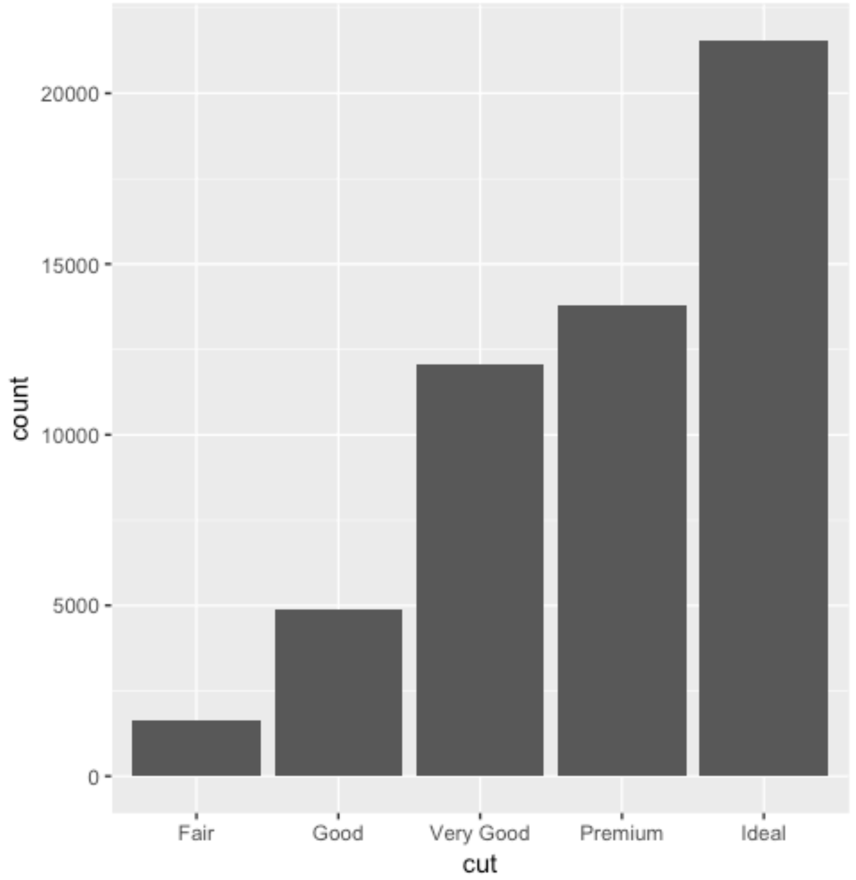
Feature	Explanation
price	price in US dollars \$326–\$18,823
carat	weight of the diamond (0.2–5.01)
cut	quality of the cut (Fair, Good, Very Good, Premium, Ideal)
color	diamond colour, from J (worst) to D (best)
clarity	a measurement of how clear the diamond is (I1 (worst), SI1, SI2, VS1, VS2, VVS1, VVS2, IF (best))
x	length in mm (0–10.74)
y	width in mm (0–58.9)
z	depth in mm (0–31.8)
depth	total depth percentage = z / mean(x, y) = 2 * z / (x + y) (43–79)
table	width of top of diamond relative to widest point (43–95)

Summary of dataset

```
> summary(diamonds)
     carat               cut        color        clarity          depth
 Min.   :0.2000   Fair     : 1610   D: 6775   SI1    :13065   Min.   :43.00
 1st Qu.:0.4000   Good     : 4906   E: 9797   VS2    :12258   1st Qu.:61.00
 Median :0.7000   Very Good:12082   F: 9542   SI2    : 9194   Median :61.80
 Mean   :0.7979   Premium  :13791   G:11292   VS1    : 8171   Mean   :61.75
 3rd Qu.:1.0400   Ideal    :21551   H: 8304   VVS2   : 5066   3rd Qu.:62.50
 Max.   :5.0100                     I: 5422   VVS1   : 3655   Max.   :79.00
                                    J: 2808   (Other): 2531
     table           price             x                y                z
 Min.   :43.00   Min.   :  326   Min.   : 0.000   Min.   : 0.000   Min.   : 0.000
 1st Qu.:56.00   1st Qu.:  950   1st Qu.: 4.710   1st Qu.: 4.720   1st Qu.: 2.910
 Median :57.00   Median : 2401   Median : 5.700   Median : 5.710   Median : 3.530
 Mean   :57.46   Mean   : 3933   Mean   : 5.731   Mean   : 5.735   Mean   : 3.539
 3rd Qu.:59.00   3rd Qu.: 5324   3rd Qu.: 6.540   3rd Qu.: 6.540   3rd Qu.: 4.040
 Max.   :95.00   Max.   :18823   Max.   :10.740   Max.   :58.900   Max.   :31.800
```

(6) Statistical Transformations

- Lets explore the *bar chart*: appears simple, yet reveals a subtle feature of plots
- The bar chart geom_bar() shows the total number of diamonds, grouped by cut
- **But where does the count come from?**



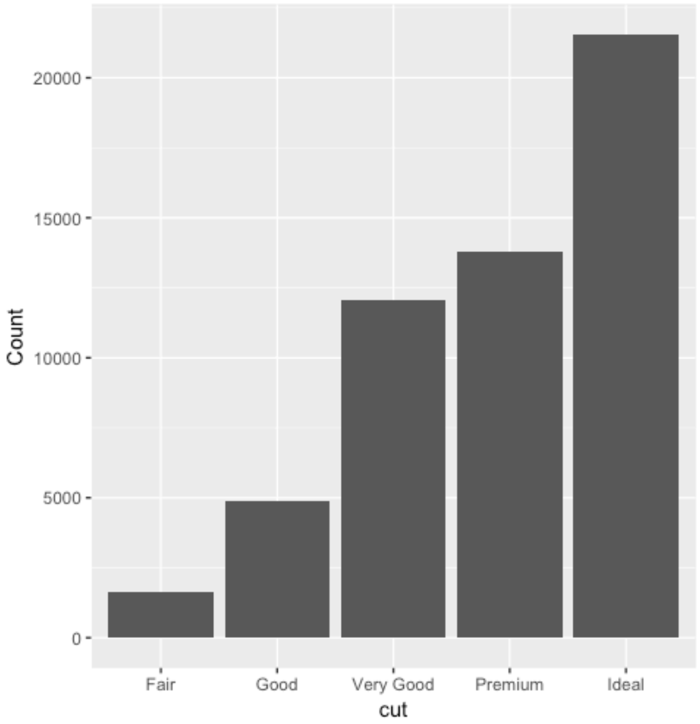
```
ggplot(data=diamonds) +
  geom_bar(mapping = aes(x = cut))
```

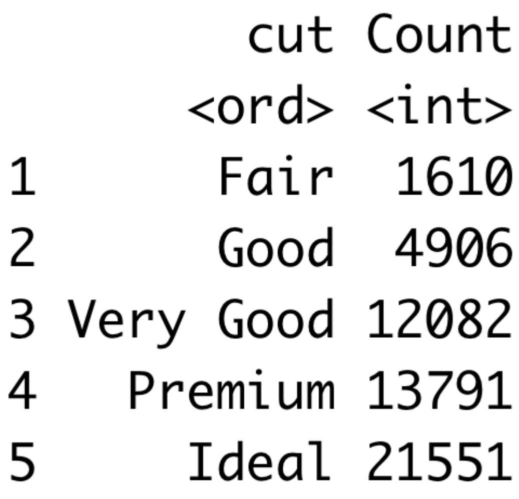
Explanation

- Many graphs, like scatterplots, plot the raw values of the dataset
- However, other graphs (e.g. bar charts) calculate new values to plot
  - **Bar charts, histograms and frequency polygons** bin your data and plot bin counts, the number of points that fall in each bin
  - **Smoothers** fit a model to your data and the plot predictions from the model
  - **Boxplots** compute a robust summary of the distribution and display a specially formatted box

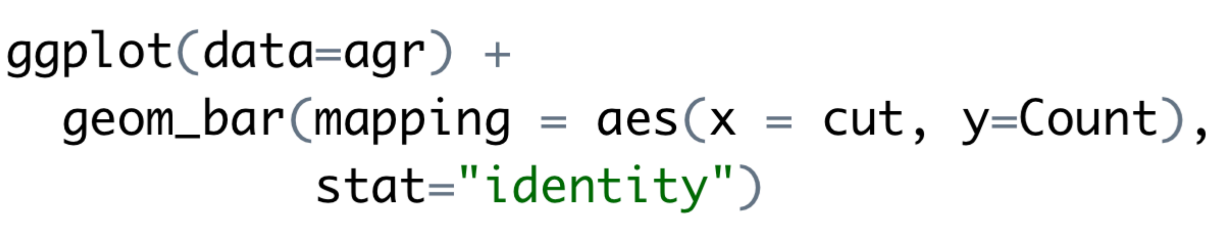
Overriding the default stat

- Every geom has a default stat, and every stat has a default geom.
- What is the aggregated data was already contained in 5 rows?
- **Use stat="identity"**





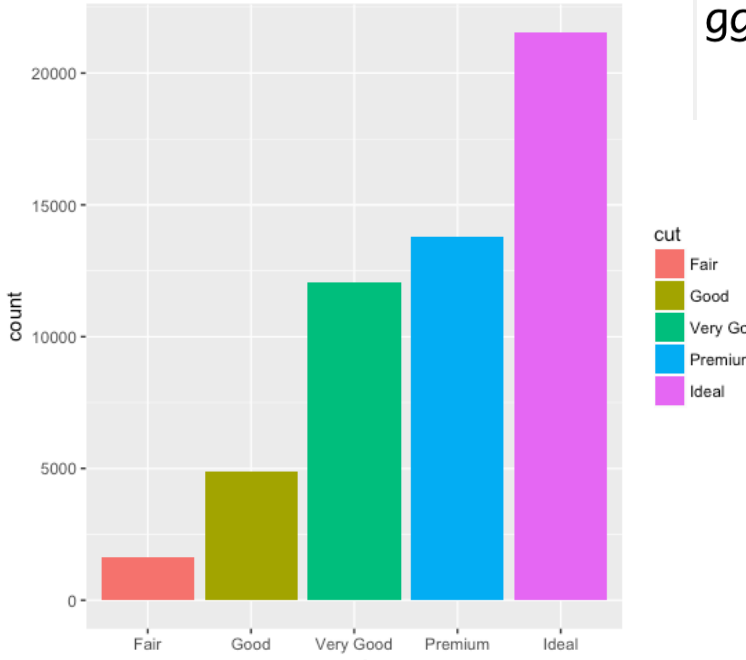
```
         cut Count
       <ord> <int>
1       Fair  1610
2       Good  4906
3 Very Good 12082
4    Premium 13791
5      Ideal 21551
```



```
ggplot(data=agr) +
  geom_bar(mapping = aes(x = cut, y=Count),
                stat="identity")
```

fill aesthetic for bar charts

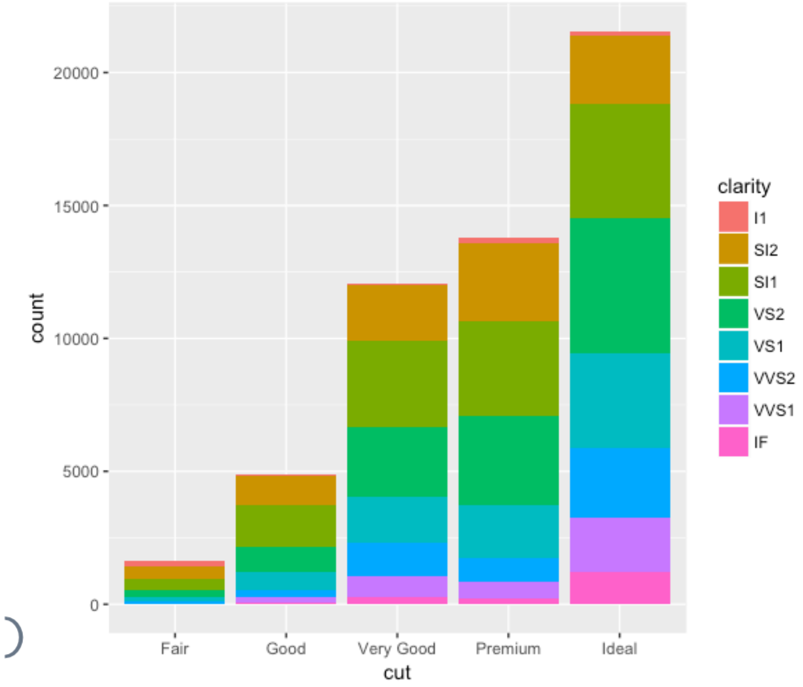
- Bar charts can be coloured using the fill aesthetic



```
ggplot(data=diamonds) +
  geom_bar(mapping=aes(x=cut,fill=cut))
```

- When a different variable is used, the graph has further detail

```
ggplot(data=diamonds) +
  geom_bar(mapping=aes(x=cut,fill=clarity))
```

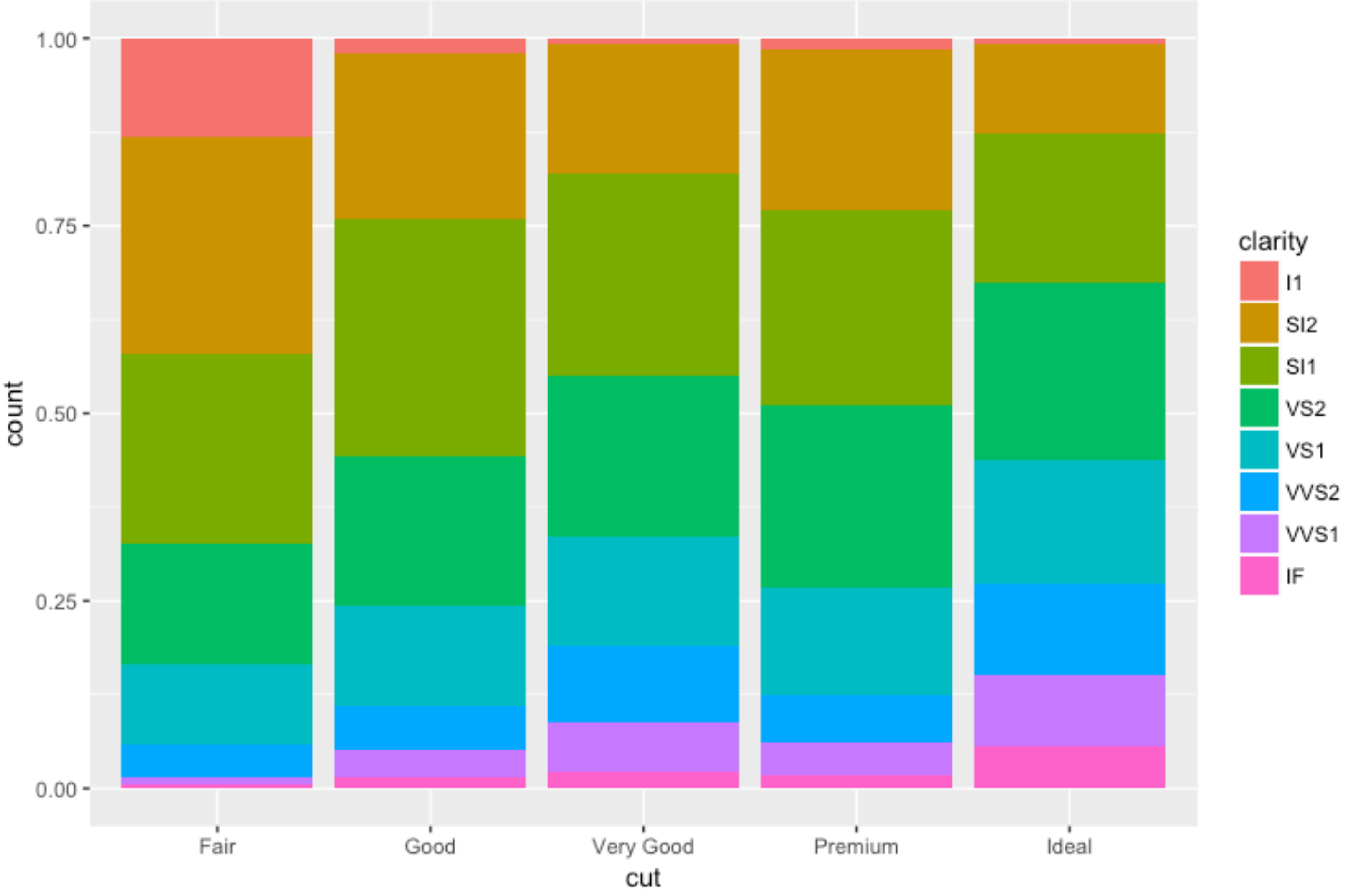


Stacking options

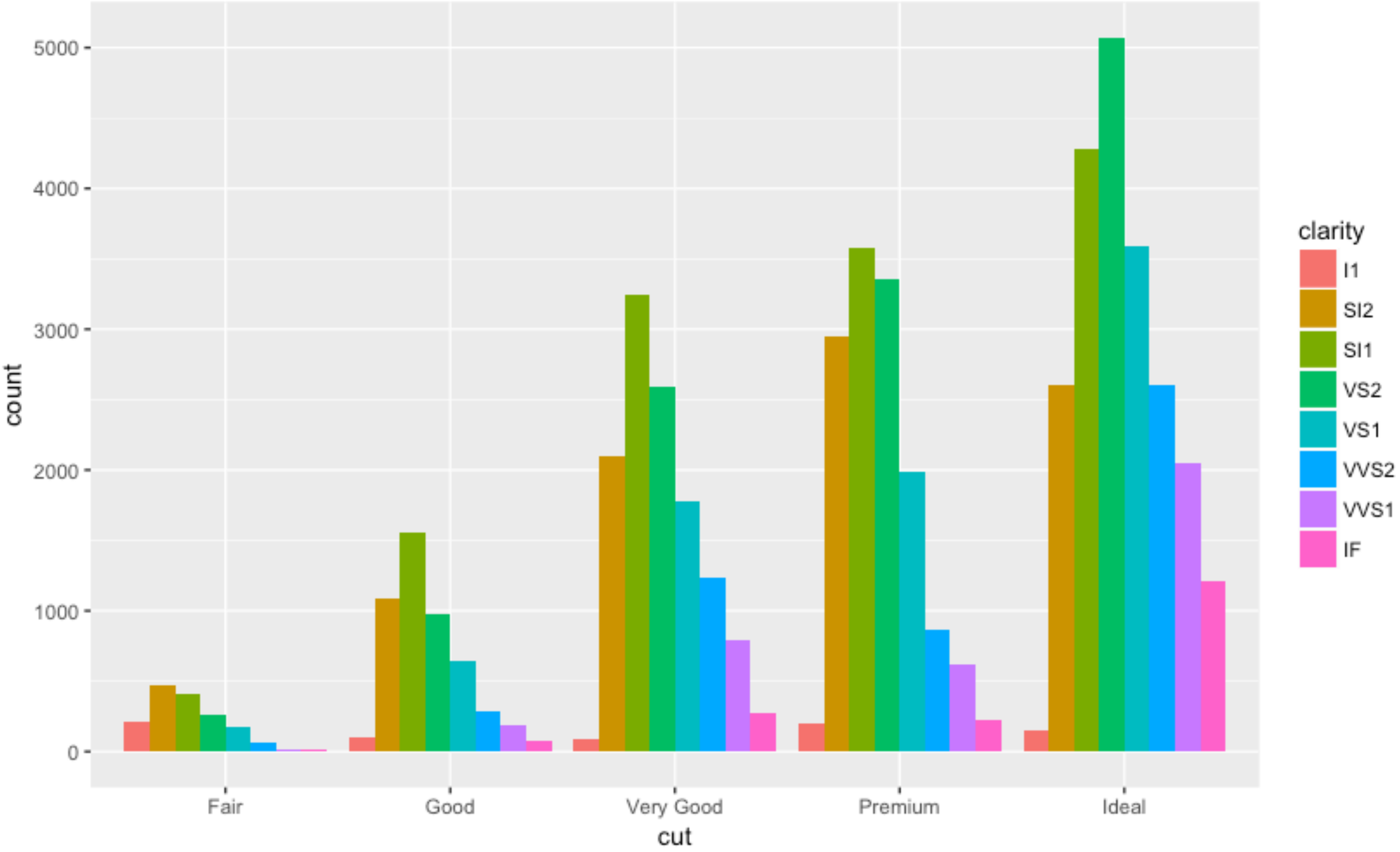
- Stacking is performed automatically by the position adjustment specified by the **position** argument
- Examples include "identity", "fill" and "dodge"

- "fill"
  - Works like stacking, but each stacked bar is the same height
  - Makes it easier to compare proportions
- "dodge"
  - Places objects directly beside one another
  - Makes it easier to compare individual values

```
ggplot(data=diamonds) +
  geom_bar(mapping=aes(x=cut,fill=clarity),
           position="fill")
```

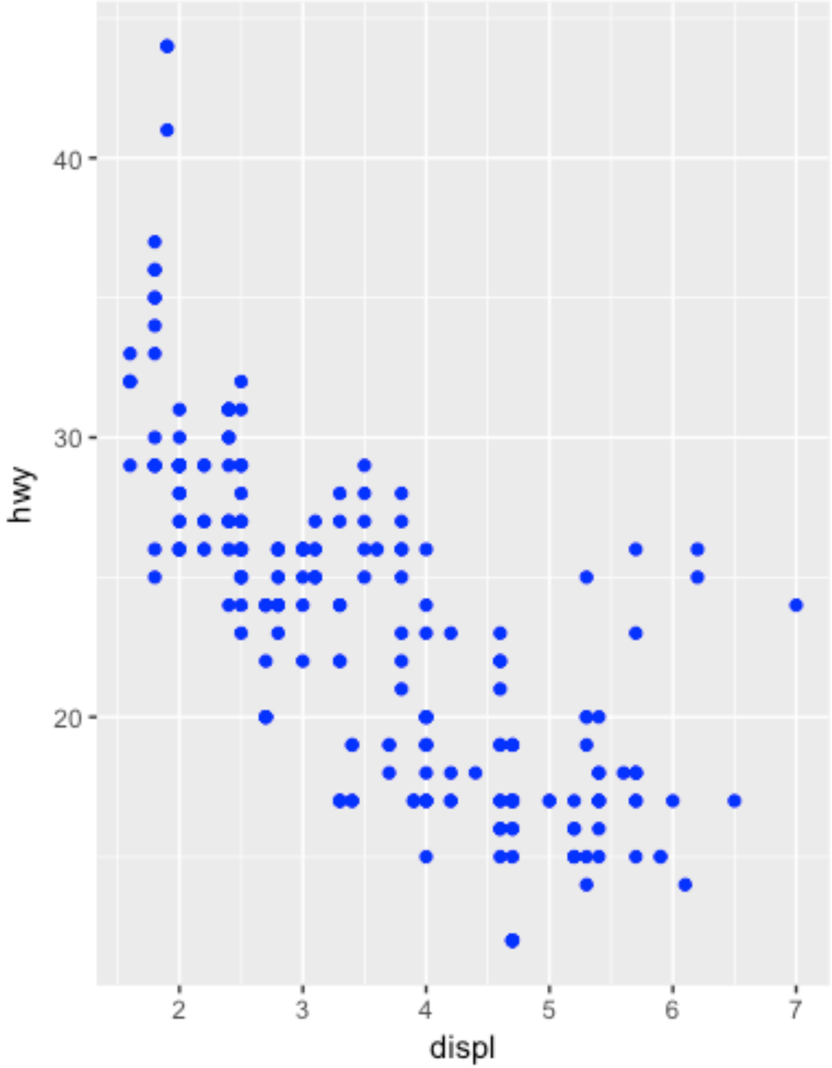


```
ggplot(data=diamonds) +
  geom_bar(mapping=aes(x=cut,fill=clarity),
                position="dodge")
```

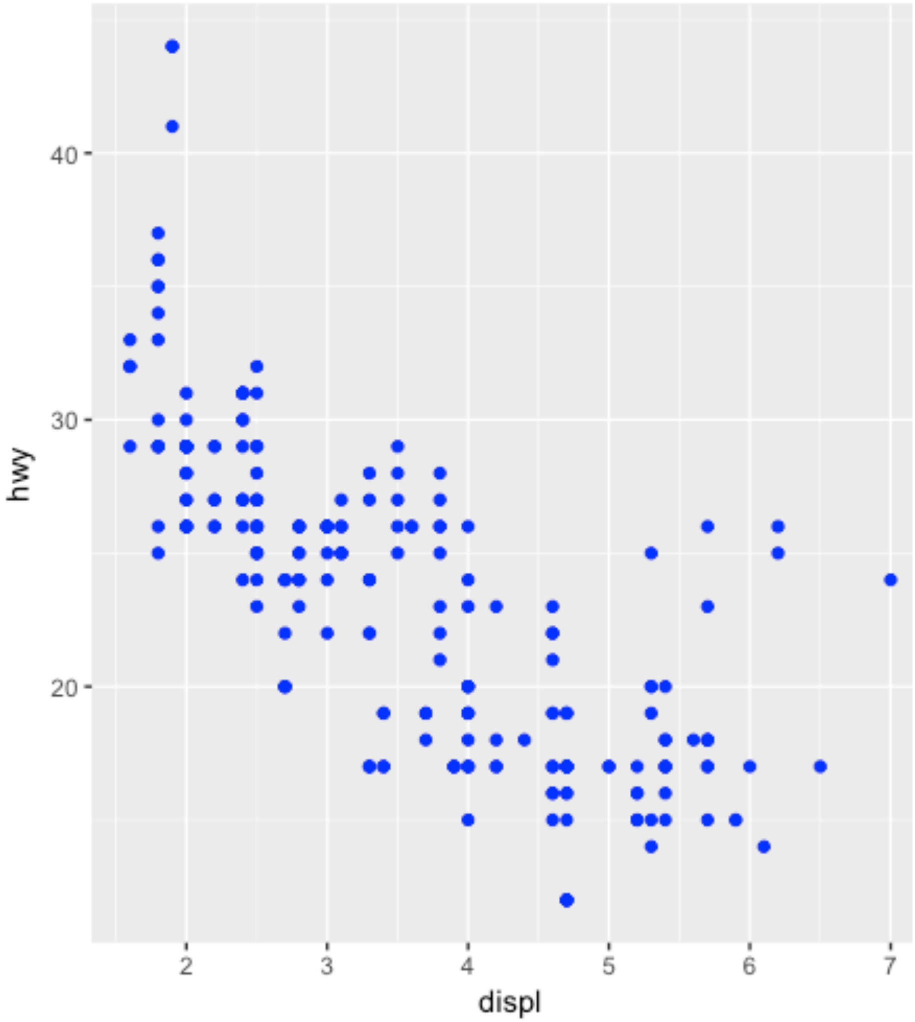


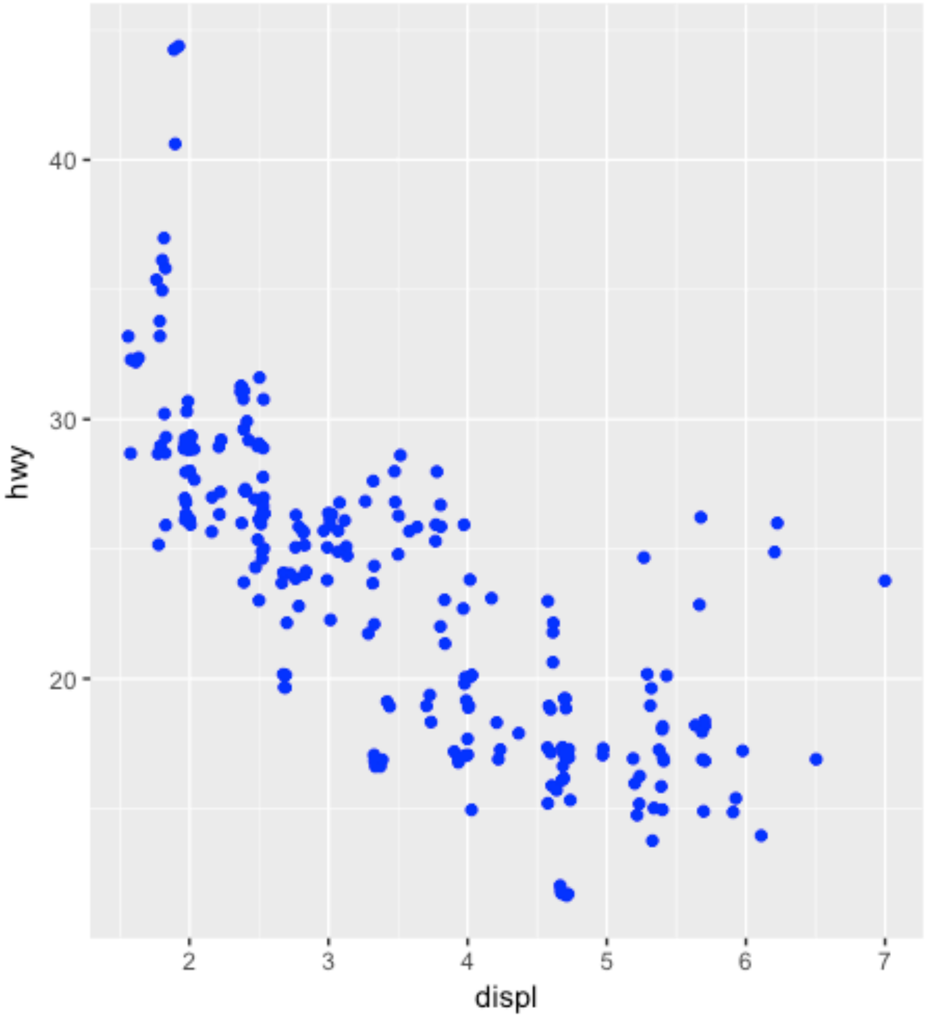
Additional adjustment

- Recall our first scatterplot
- 126 points displayed, yet there are 234 observations
- Many points can overlap, so it makes it hard to see where the mass of data is
- Are all points spread equally, or is there one special combination that contains 129 values?
- “jitter” adds random noise to each point.

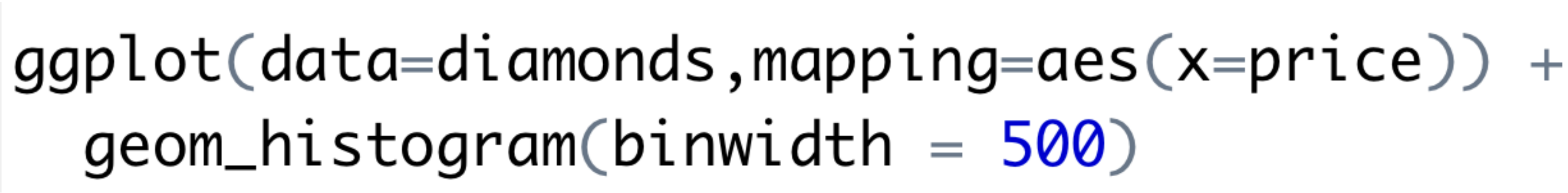


```
ggplot(data=mpg)+
  geom_point(mapping=aes(x=displ,y=hwy),
                           position="jitter",colour="blue")
```

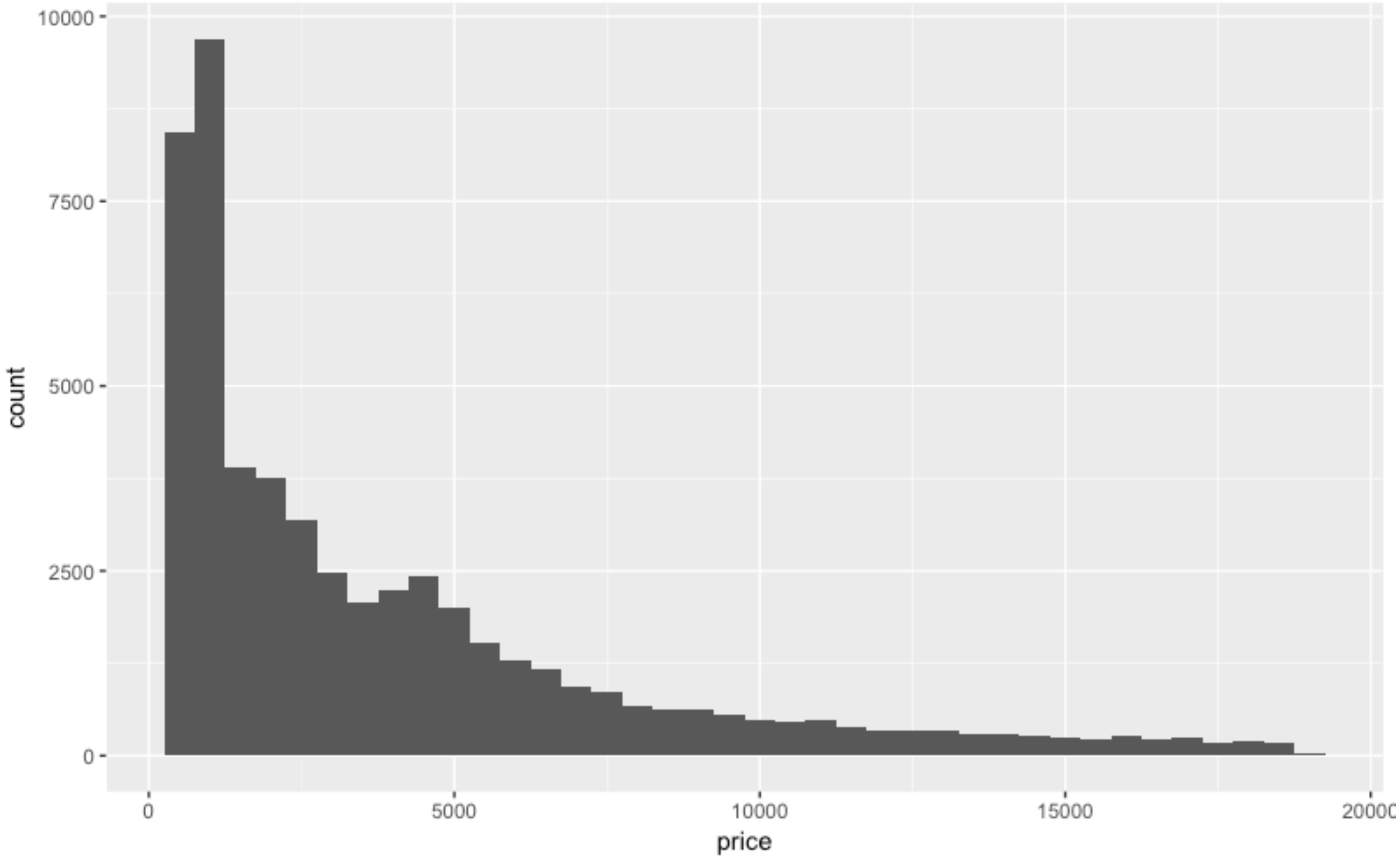




Histogram

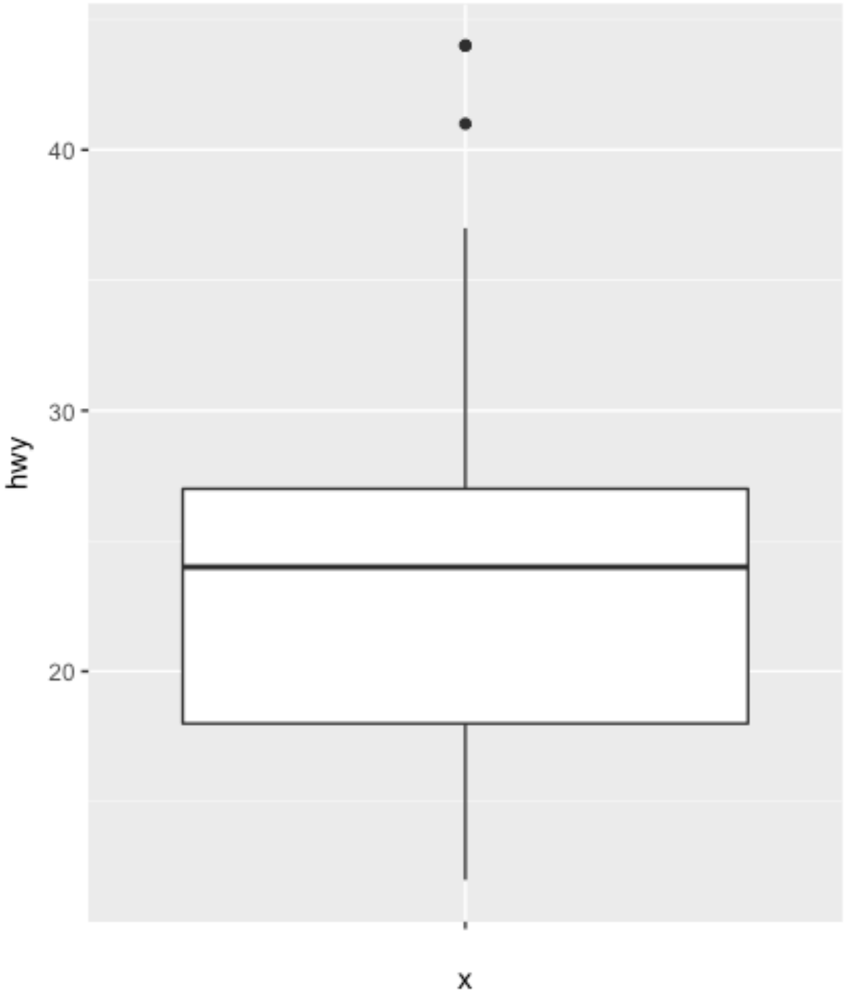


```
ggplot(data=diamonds,mapping=aes(x=price)) +
  geom_histogram(binwidth = 500)
```

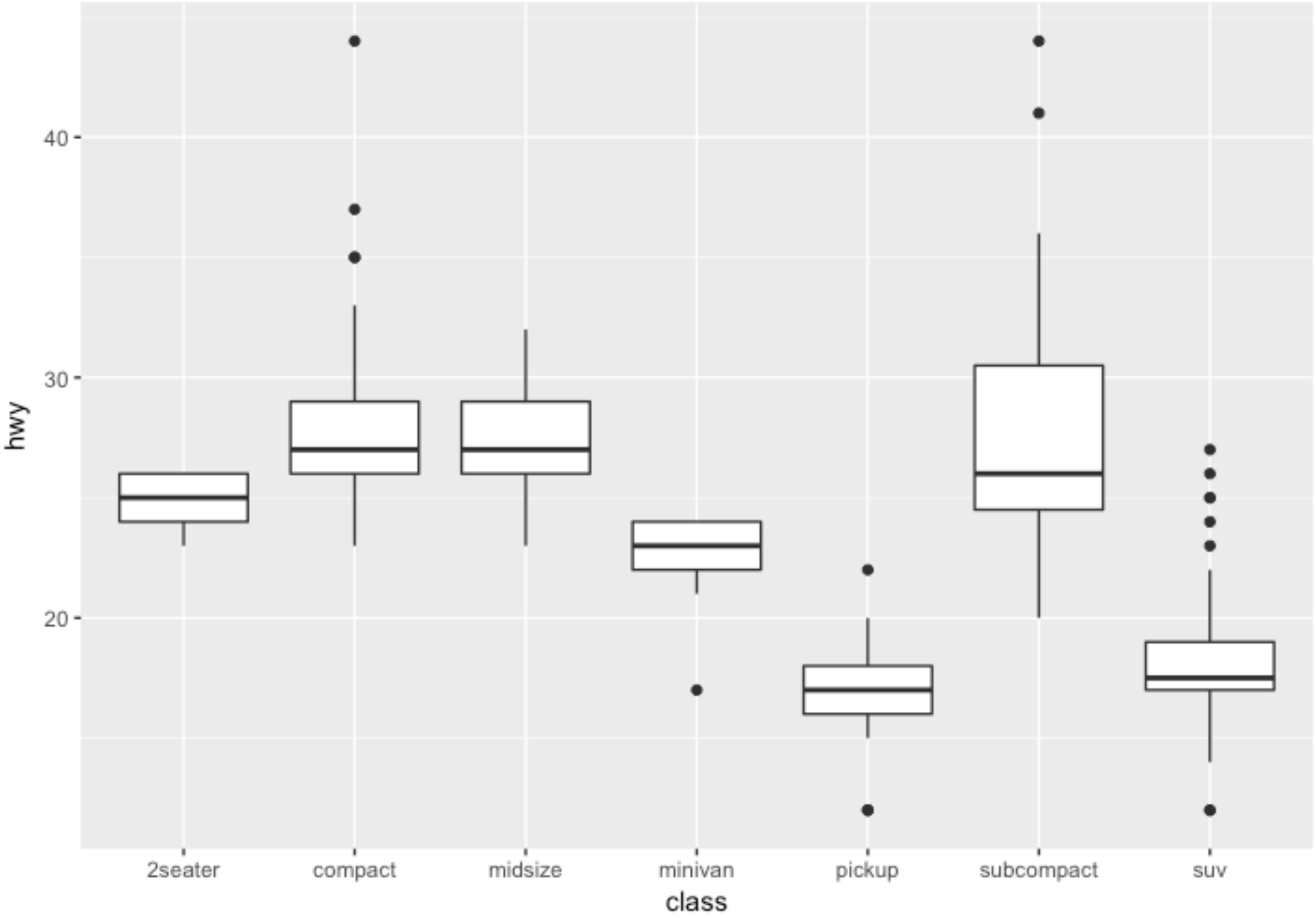


Boxplot

- Display the distribution of a continuous variable broken down by a categorical variable
- Box that stretches from the 25th to 75th percentile a distance known as the interquartile range (IRQ)
- Median in the middle of box
- Points outside more that 1.5 times the IQR from either edge of the box are displayed (outliers)
- Whisker extends to the farthest non-outlier point in the distribution



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

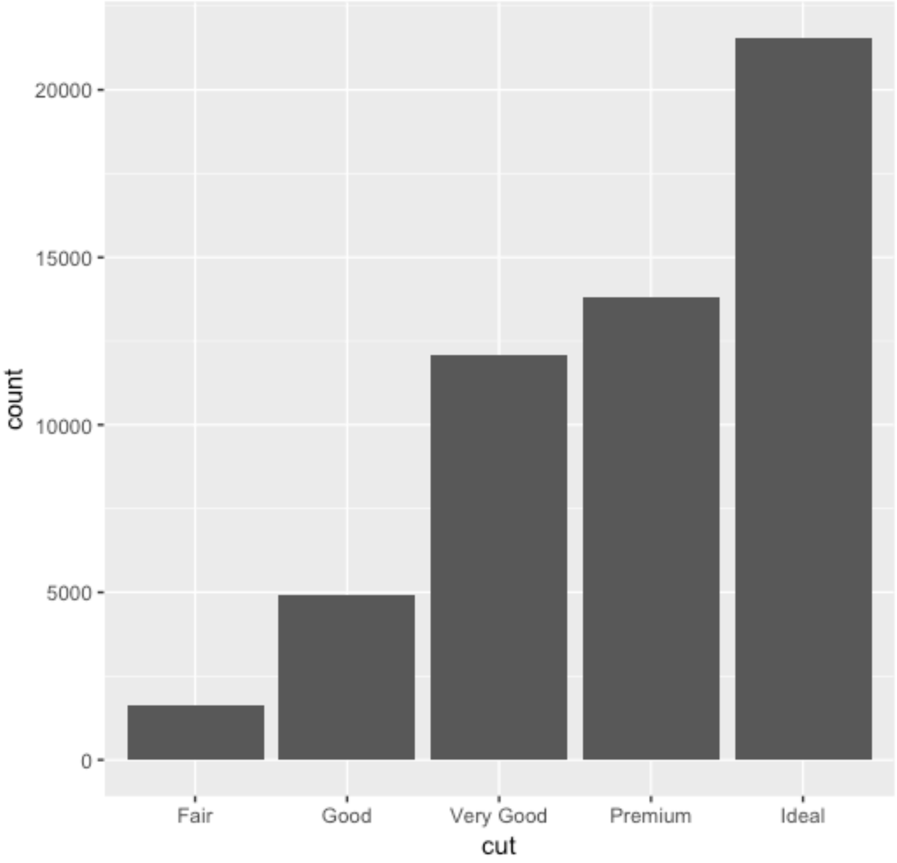


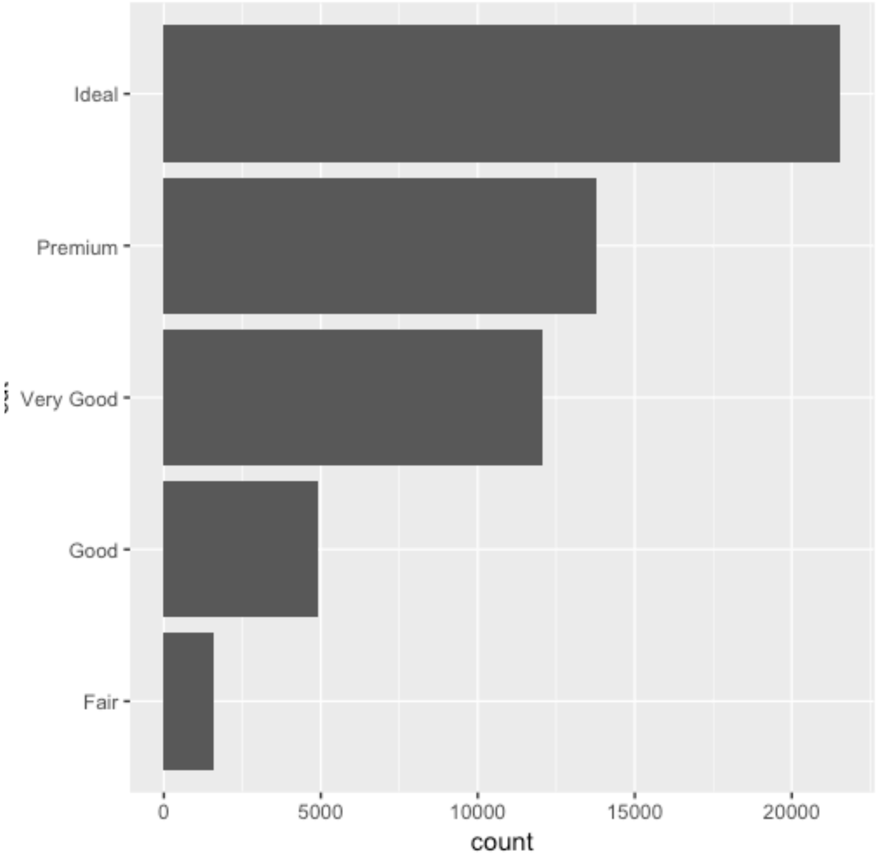
(7) Coordinate Systems

- Probably the most complicated part of ggplot2
- Default us the Cartesian coordinate system where the x and y position act independently to find the location of each point

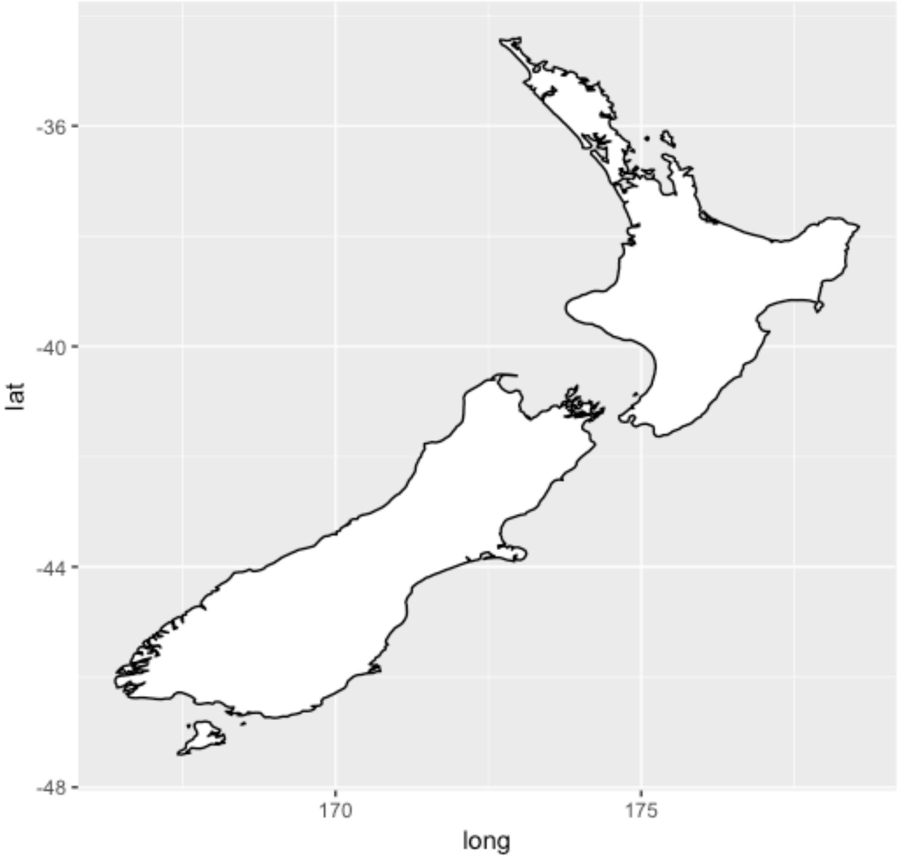
- A number of other coordinate systems can be helpful:
  - coord_flip(), switches the x and y axes
  - coord_quickmap() sets the aspect ration correctly for maps, important for plotting spatial data

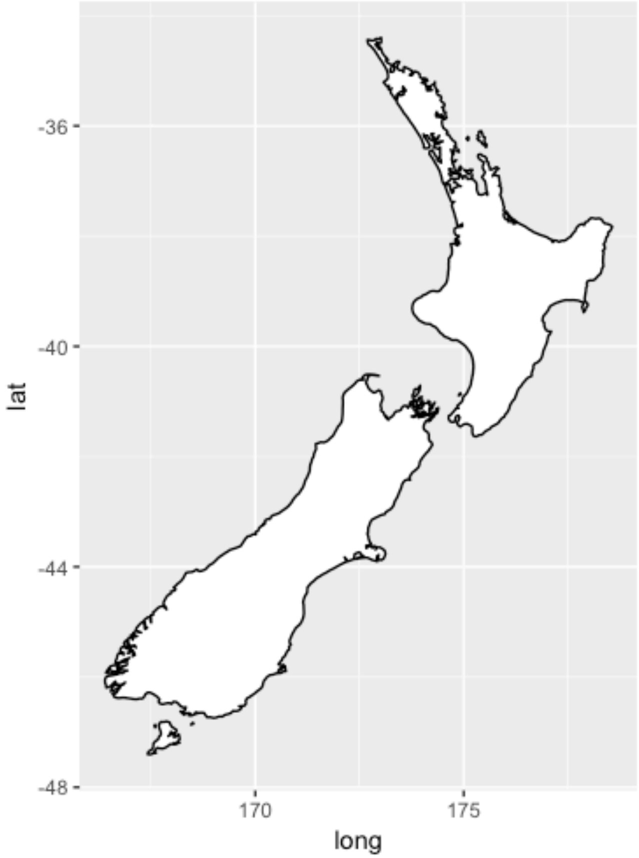
```
ggplot(data=diamonds) +
  geom_bar(mapping = aes(x = cut)) + coord_flip()
```





```
ggplot(nz, aes(long,lat, group=group)) +
  geom_polygon(fill="white",colour="black") +
  coord_quickmap()
```





(8) The Layered Grammar of Graphics

- The ggplot2 approach can be summarised by a template
- It can take seven parameters, but usually not all need to be applied (defaults used)
- These seven parameters compose the grammar of graphics

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