Data Visualisation

Yifan Jin

05/02/2020

Introduction

Prerequisites

```
library(tidyverse)
```

```
## -- Attaching packages --------------------------------------------- tidyverse 1.3.0 --
```

```
## v ggplot2 3.2.1     v purrr   0.3.3
## v tibble  2.1.3     v dplyr   0.8.3
## v tidyr   1.0.0     v stringr 1.4.0
## v readr   1.3.1     v forcats 0.4.0
```

```
## -- Conflicts ------------------------------------------------ tidyverse_conflicts() --
## x dplyr::filter() masks stats::filter()
## x dplyr::lag()    masks stats::lag()
```

```
# install.packages("maps")
library(maps)
```

```
##
## Attaching package: 'maps'
```

```
## The following object is masked from 'package:purrr':
##
##     map
```

We can employ function by using `package::function()` eg. `ggplot2::ggplot()` or directly using function eg. `ggplot()`

First steps

Think about the relationship between variables, think about the mean of one group is smaller or greater than other, think about one variable is positively or negatively correlated with other.

Example: The data frame `mpg`

```
mpg
```

```
## # A tibble: 234 x 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(l~ f        18    29 p     comp~
##   2 audi         a4         1.8  1999     4 manual~ f        21    29 p     comp~
##   3 audi         a4         2    2008     4 manual~ f        20    31 p     comp~
##   4 audi         a4         2    2008     4 auto(a~ f        21    30 p     comp~
##   5 audi         a4         2.8  1999     6 auto(l~ f        16    26 p     comp~
##   6 audi         a4         2.8  1999     6 manual~ f        18    26 p     comp~
##   7 audi         a4         3.1  2008     6 auto(a~ f        18    27 p     comp~
##   8 audi         a4 quat~   1.8  1999     4 manual~ 4        18    26 p     comp~
##   9 audi         a4 quat~   1.8  1999     4 auto(l~ 4        16    25 p     comp~
##  10 audi         a4 quat~   2    2008     4 manual~ 4        20    28 p     comp~
## # ... with 224 more rows
```

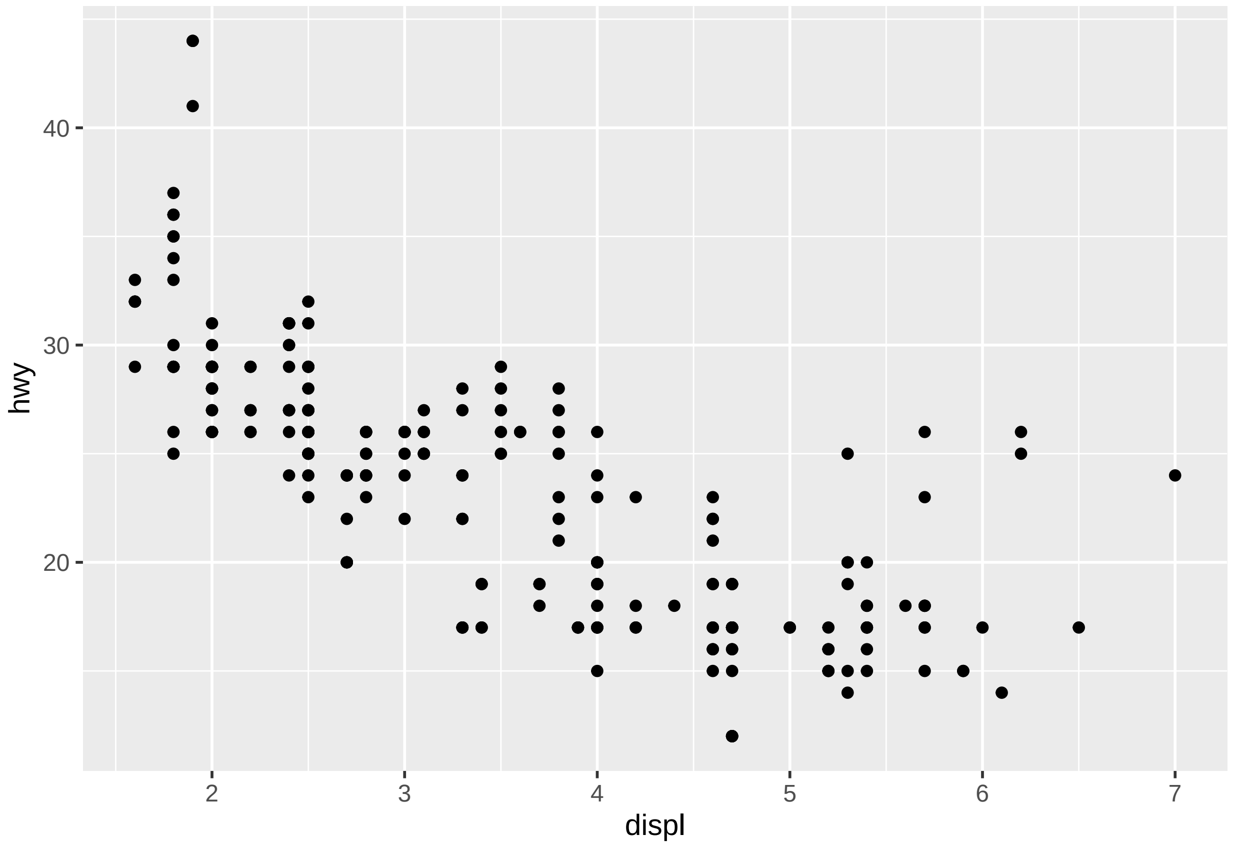
Variables descriptions:

1. `displ`, a car's engine size, in litres
2. `hwy`, a car's fuel efficiency on the highway, in miles per gallon (mpg). A car with low fuel efficiency consumes more fuel than a car with a high fuel efficiency when they travel the same distance

To learn more information about `mpg`, use `?mpg` to get more.

Creating a ggplot

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



We can see that there is a negative relationship between engine size (`displ`) and fuel efficiency (`hwy`). We can form a hypothesis test to formally test the relationship.

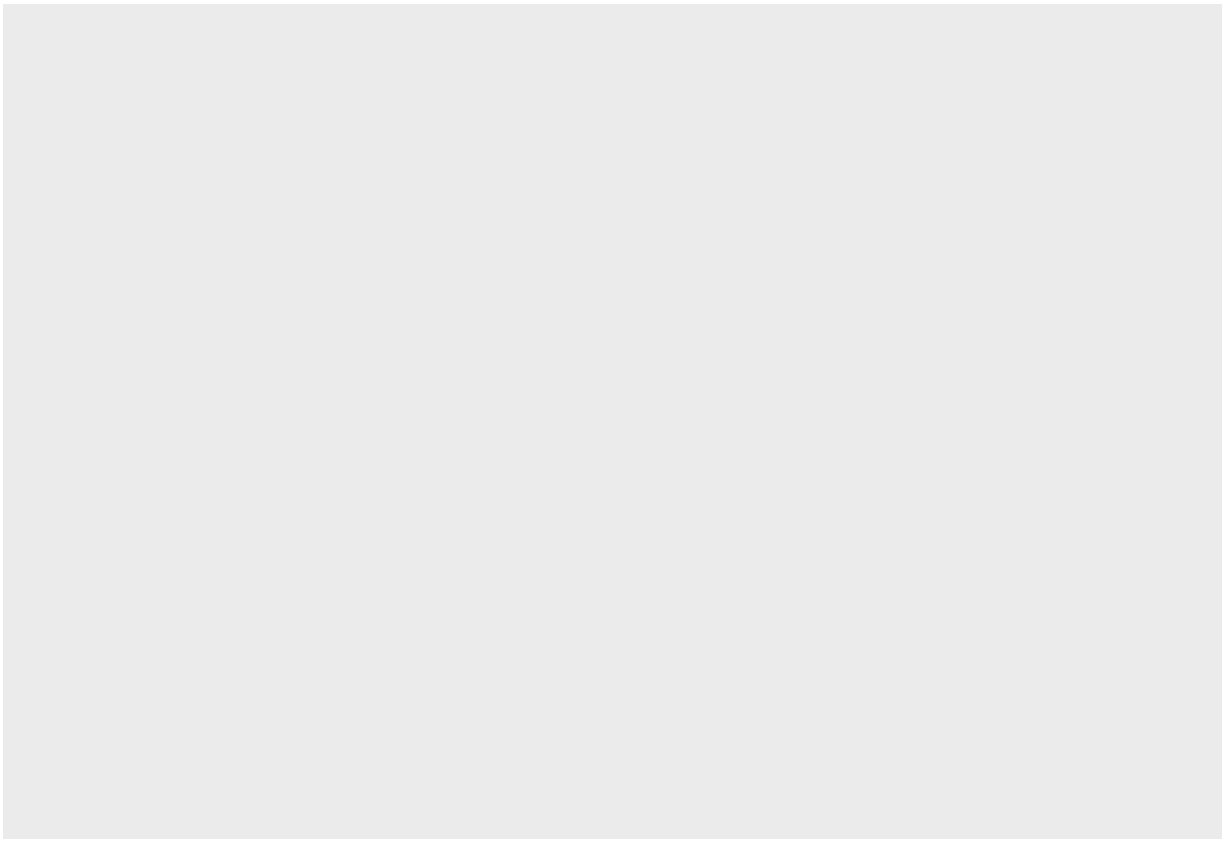
A graphing template

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

Exercise

1. Run `ggplot(data=mpg)`. What do you see?

```
ggplot(data=mpg)
```



Answer: We get nothing

2. How many rows are in `mpg`? How many columns

```
str(mpg)
```

```
## Classes 'tbl_df', 'tbl' and 'data.frame':    234 obs. of  11 variables:
##  $ manufacturer: chr  "audi" "audi" "audi" "audi" ...
##  $ model       : chr  "a4" "a4" "a4" "a4" ...
##  $ displ       : num  1.8 1.8 2 2 2.8 2.8 3.1 1.8 1.8 2 ...
##  $ year        : int  1999 1999 2008 2008 1999 1999 2008 1999 1999 2008 ...
##  $ cyl         : int  4 4 4 4 6 6 6 4 4 4 ...
##  $ trans       : chr  "auto(l5)" "manual(m5)" "manual(m6)" "auto(av)" ...
##  $ drv         : chr  "f" "f" "f" "f" ...
##  $ cty         : int  18 21 20 21 16 18 18 18 16 20 ...
##  $ hwy         : int  29 29 31 30 26 26 27 26 25 28 ...
##  $ fl          : chr  "p" "p" "p" "p" ...
##  $ class       : chr  "compact" "compact" "compact" "compact" ...
```

Answer:

We have 234 rows of observations and 11 variables

3. What does the `drv` variable describe? Read the help for out

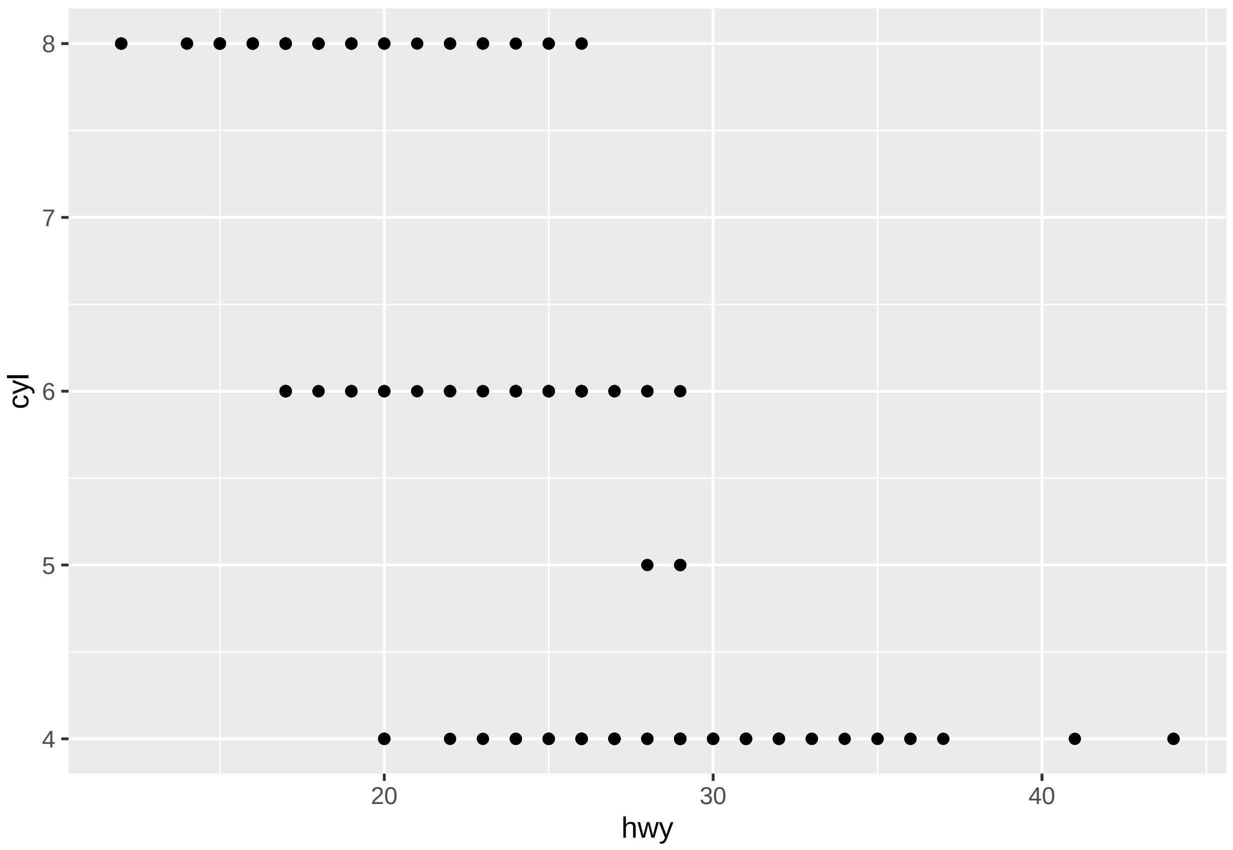
```
?mpg
```

`drv`: f = front-wheel drive, r = rear wheel drive, 4 = 4wd

4. Make a scatterplot of `hwy` vs `cyl`

Answer:

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



5. What happens if you make scatterplot of `class` vs `drv`? Why is not useful?

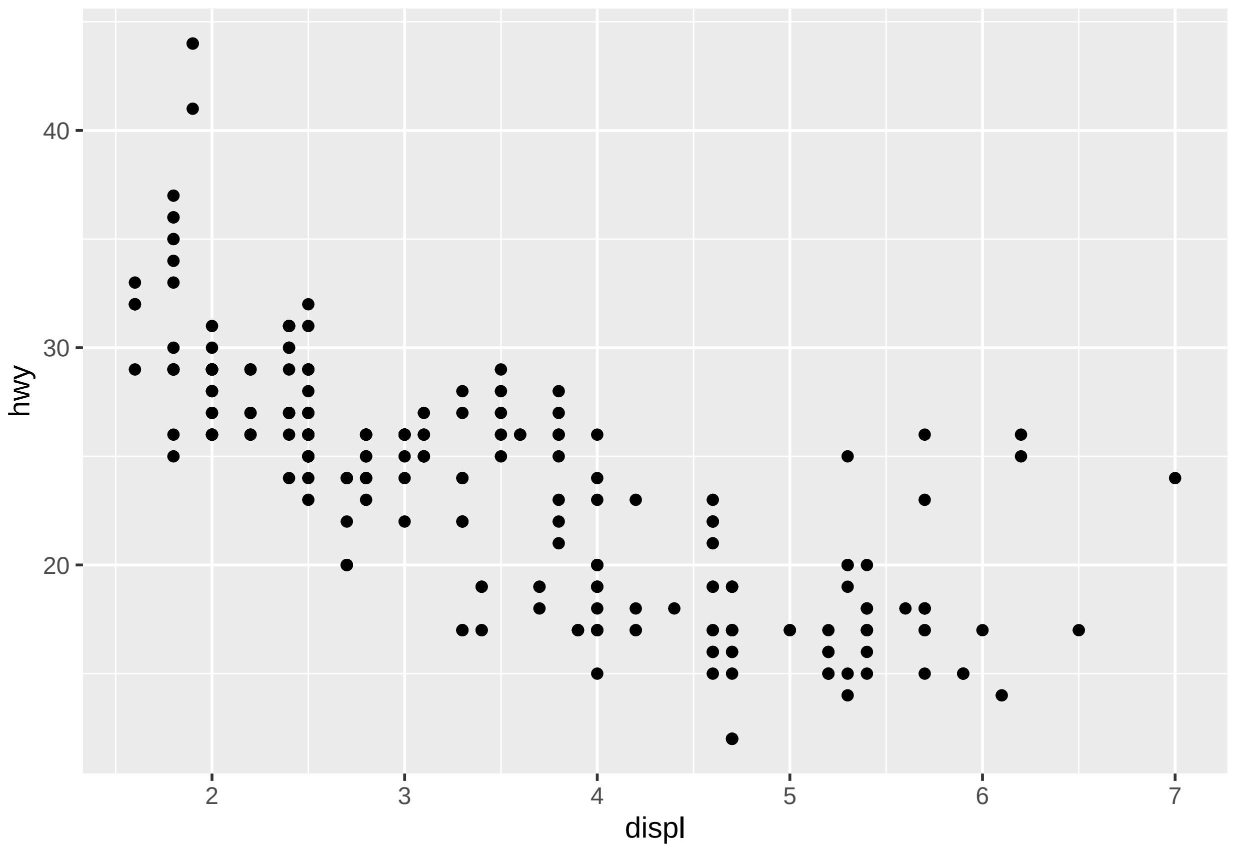
Answer:

It can be seen that `cyl` only take certain values which is discrete variable. Therefore, it looks strange. We can not see any information about the realtionship between two variables, hence, it doesn't make much sense.

Aesthetic mappings

The greatest value of a picture is when it forces us to notice what we never expected to see

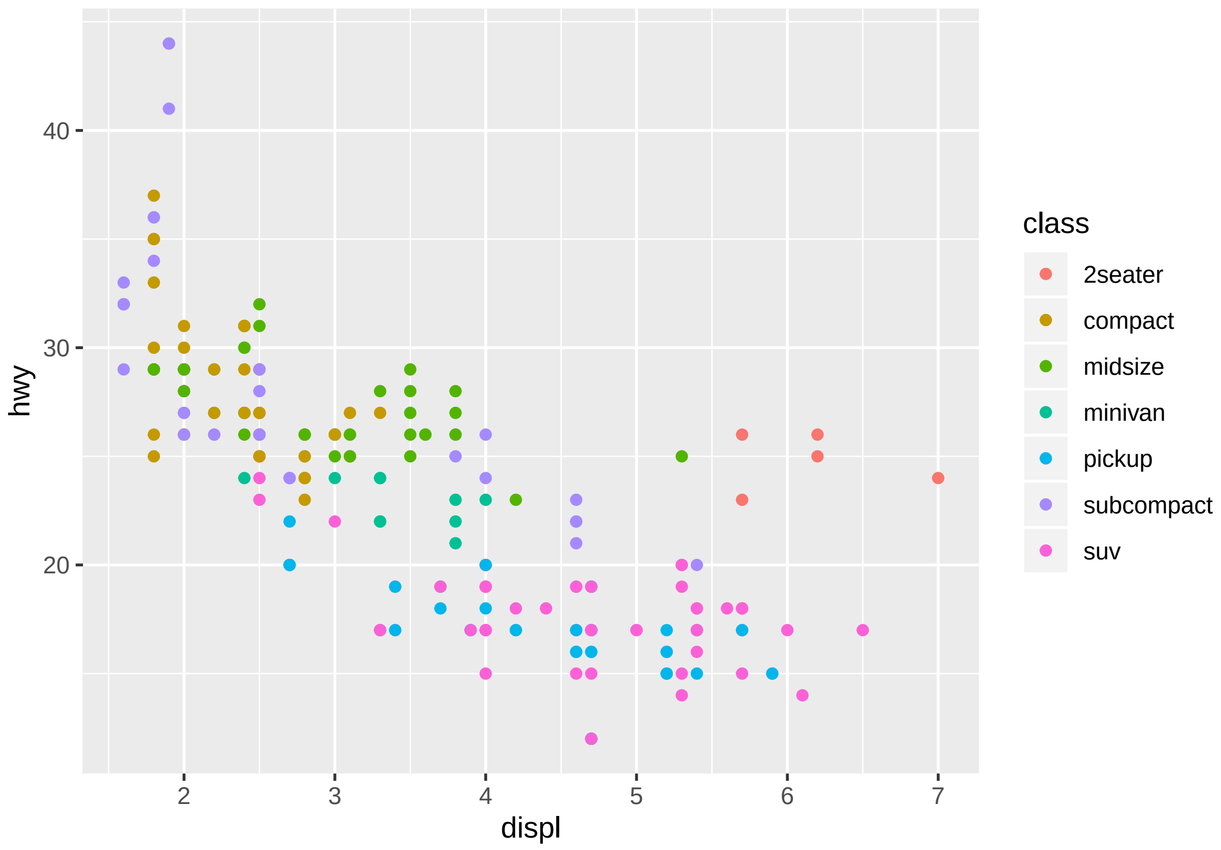
```
ggplot(data=mpg)+
  geom_point(mapping=aes(x=displ,y=hwy))
```



There are some points on the right which is outside the general trend. There may be other factors that affect hwy.

We can add `class` variable

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

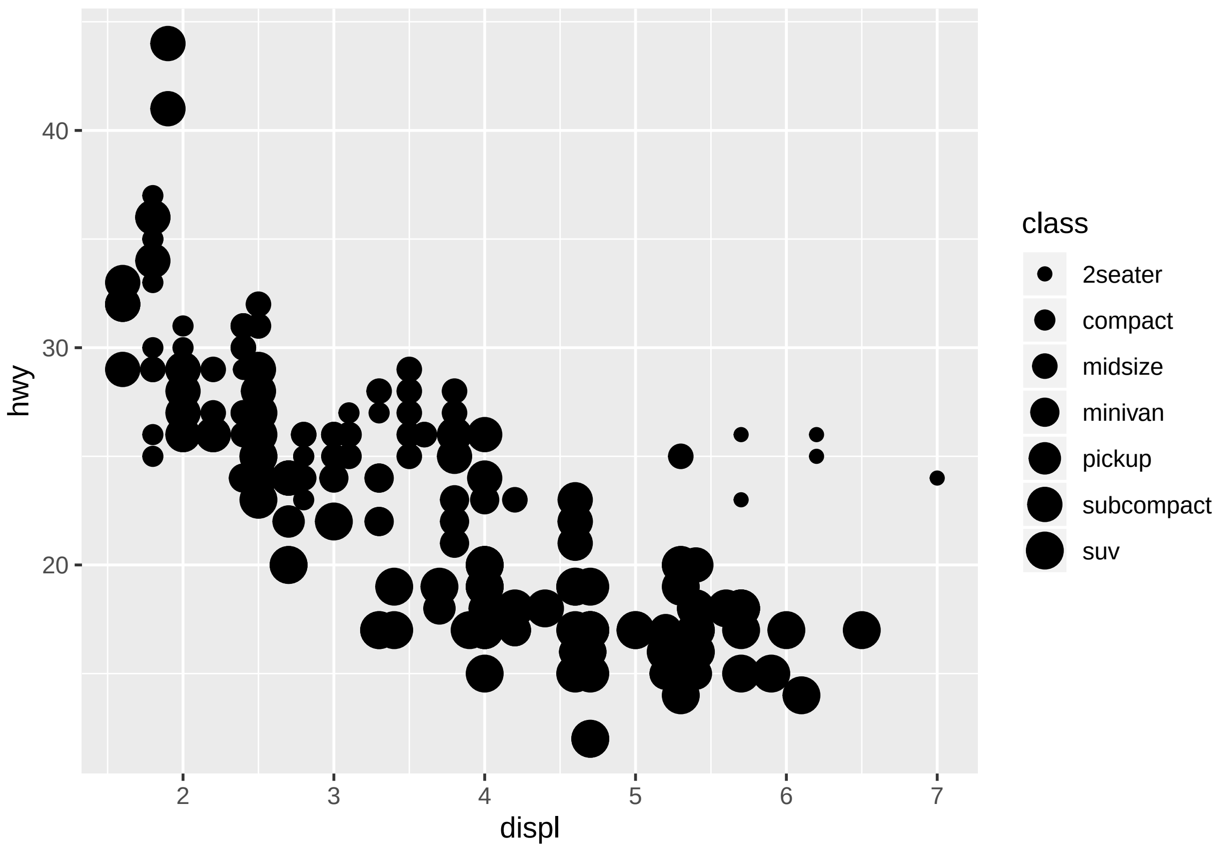


The colors reveal that many of the unusual points are two-seater cars.

Not only the color can be used for categorical variable, but also the size.

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

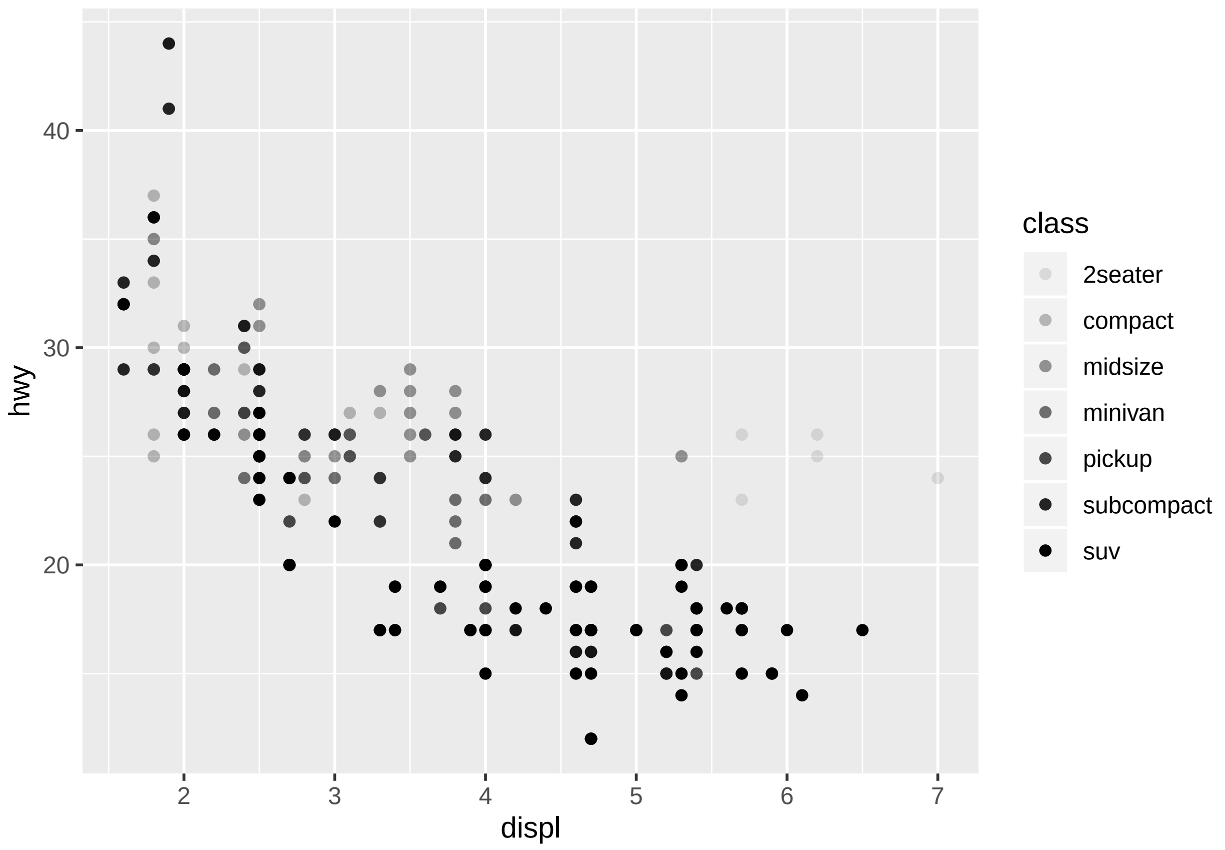
```
## Warning: Using size for a discrete variable is not advised.
```



We could also mapped `class` to the alpha aesthetic, which controls the transparency of the points, or to the shape aesthetic, which controls the shape of the points.

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

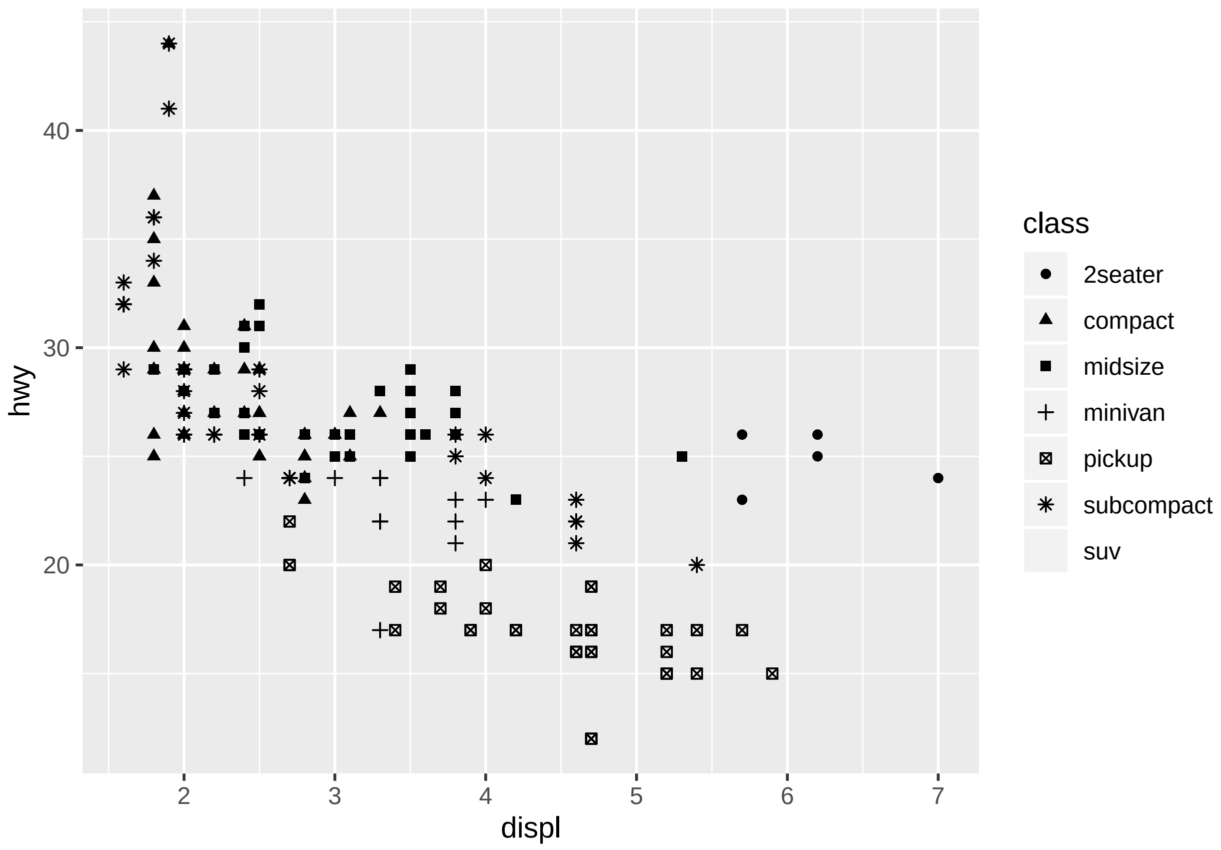
```
## Warning: Using alpha for a discrete variable is not advised.
```



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

```
## Warning: The shape palette can deal with a maximum of 6 discrete values because
## more than 6 becomes difficult to discriminate; you have 7. Consider
## specifying shapes manually if you must have them.
```

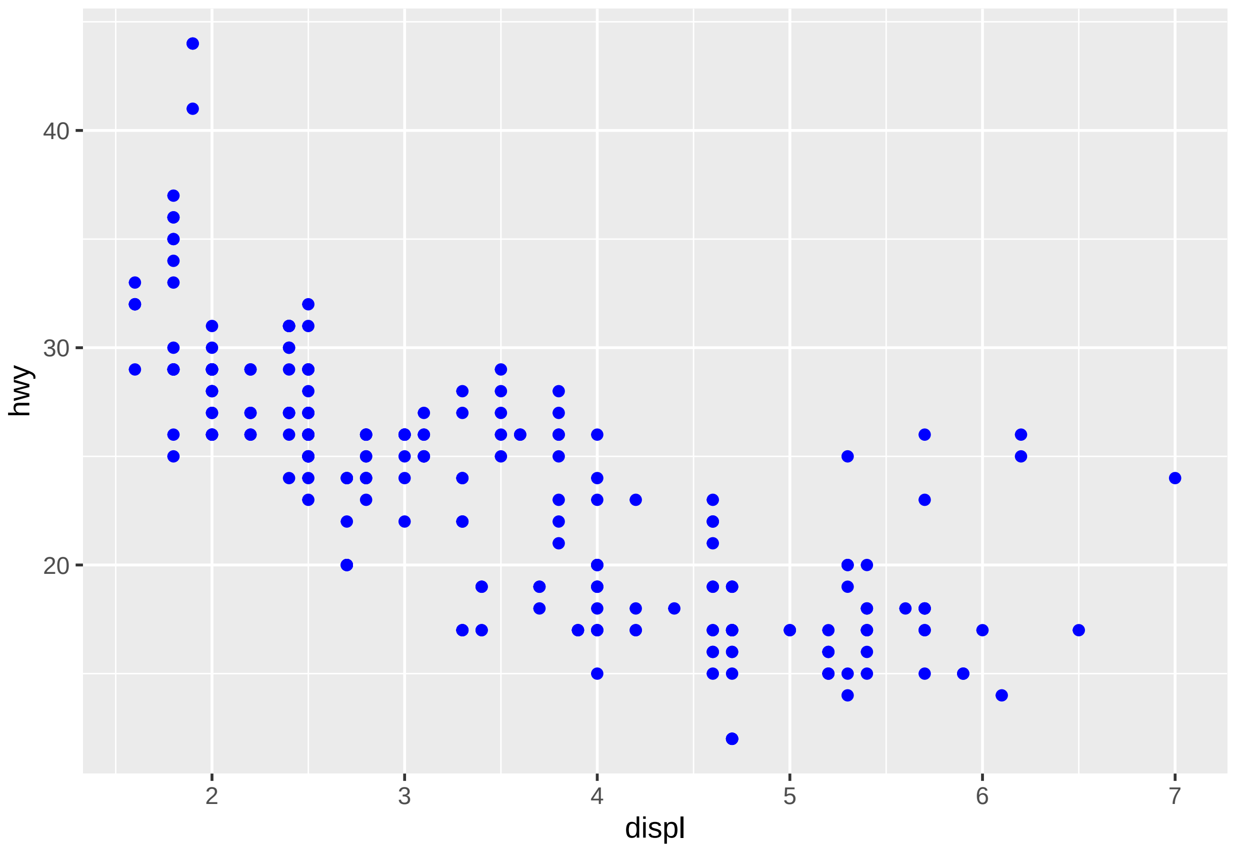
```
## Warning: Removed 62 rows containing missing values (geom_point).
```



However, for shape method, maximum shape in a graph is 6, there are seven. the additional groups are unplotted.

We also can directly add color to the plot.

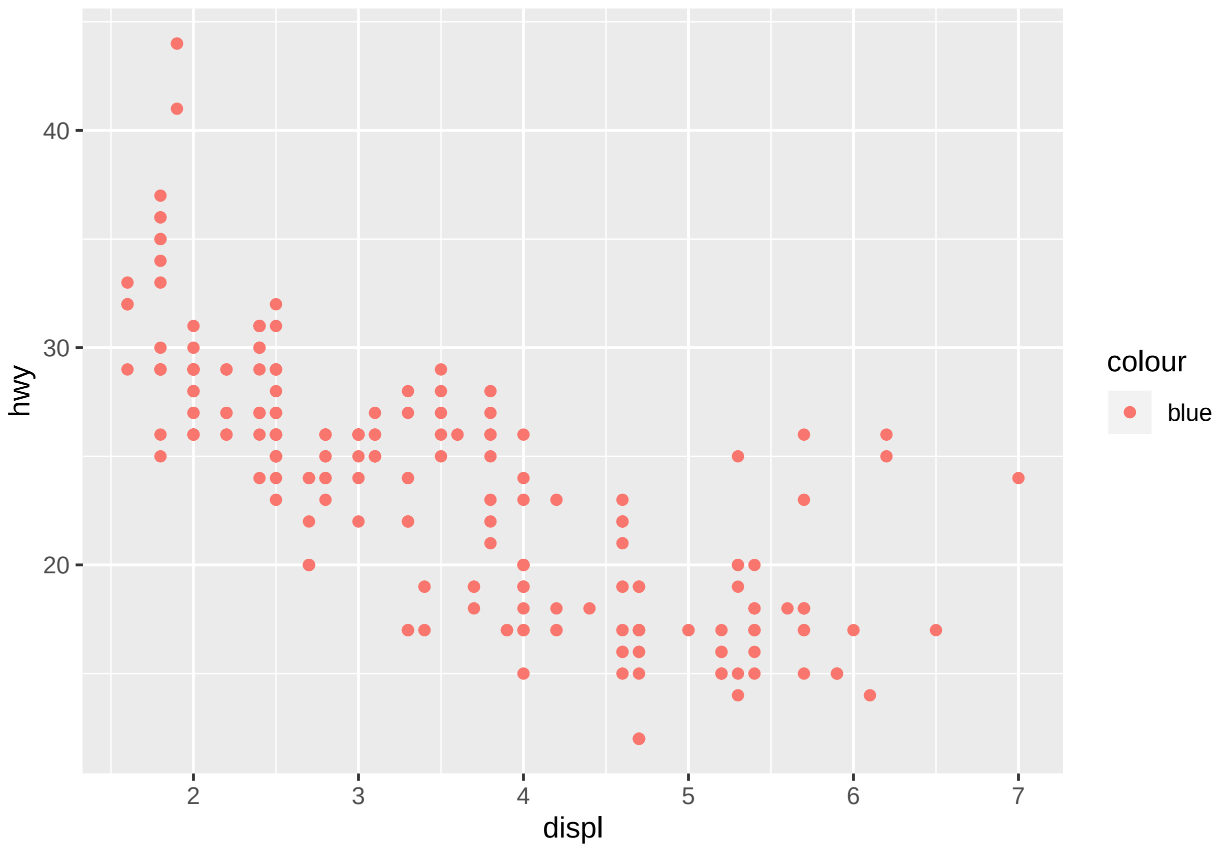
```
ggplot(data=mpg)+
  geom_point(mapping=aes(x=displ,y=hwy),color="blue")
```



Exercise

1. What is gone wrong with this code? Why are the points not blue?

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



Answer

It shouldn't inside the aes bracket.

2. Which variables in `mpg` are categorical? Which variables are continuous? use`?mpg`how can you see this information when you run `mpg`

```
mpg
```

```
## # A tibble: 234 x 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(l~ f        18    29 p     comp~
##  2 audi         a4         1.8  1999     4 manual~ f        21    29 p     comp~
##  3 audi         a4         2    2008     4 manual~ f        20    31 p     comp~
##  4 audi         a4         2    2008     4 auto(a~ f        21    30 p     comp~
##  5 audi         a4         2.8  1999     6 auto(l~ f        16    26 p     comp~
##  6 audi         a4         2.8  1999     6 manual~ f        18    26 p     comp~
##  7 audi         a4         3.1  2008     6 auto(a~ f        18    27 p     comp~
##  8 audi         a4 quat~   1.8  1999     4 manual~ 4        18    26 p     comp~
##  9 audi         a4 quat~   1.8  1999     4 auto(l~ 4        16    25 p     comp~
## 10 audi         a4 quat~   2    2008     4 manual~ 4        20    28 p     comp~
## # ... with 224 more rows
```

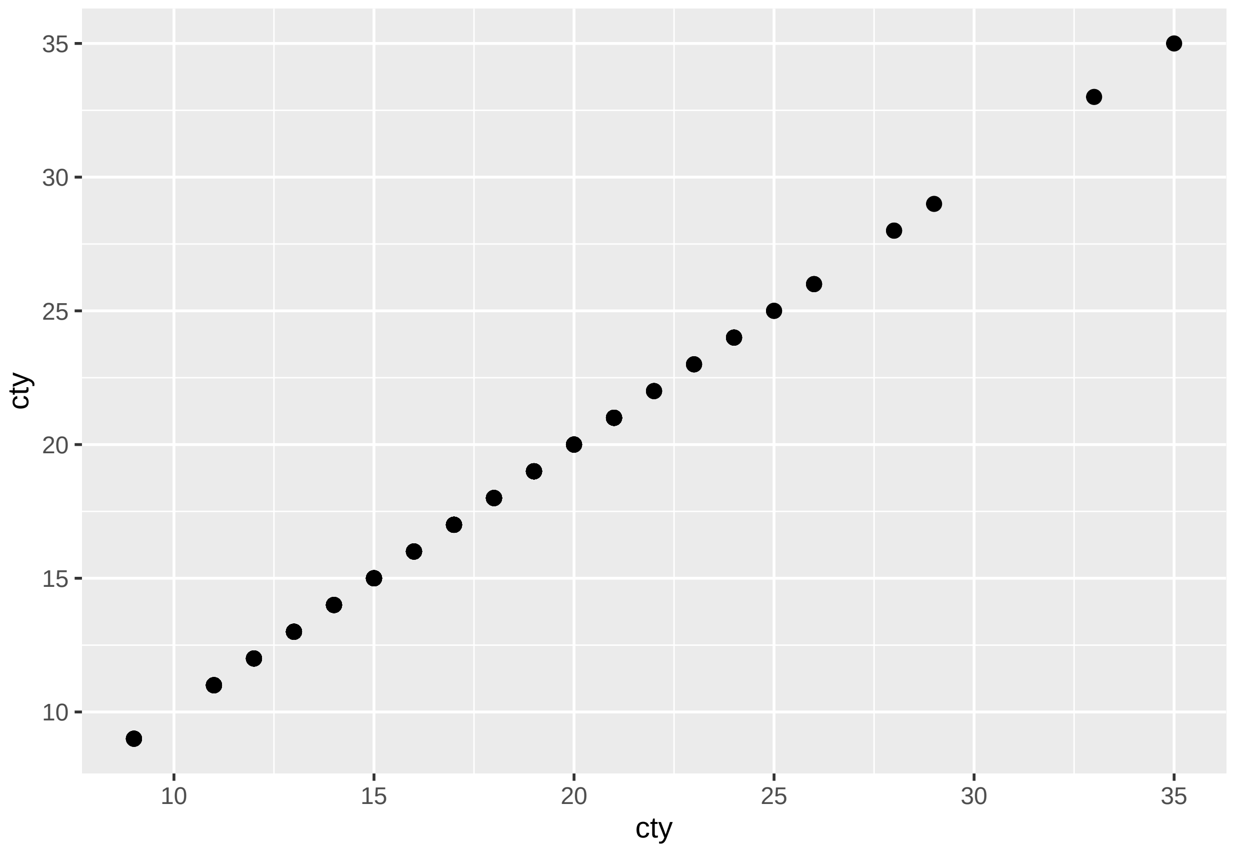
We can see this by seeing data structure

3. Map a continuous variable to `color`, `size`, and `shape`. How do these aesthetics behave differently for categorical vs. continuous variables?

Answer Nothing

4. What happens if you map the same variable to multiple aesthetics

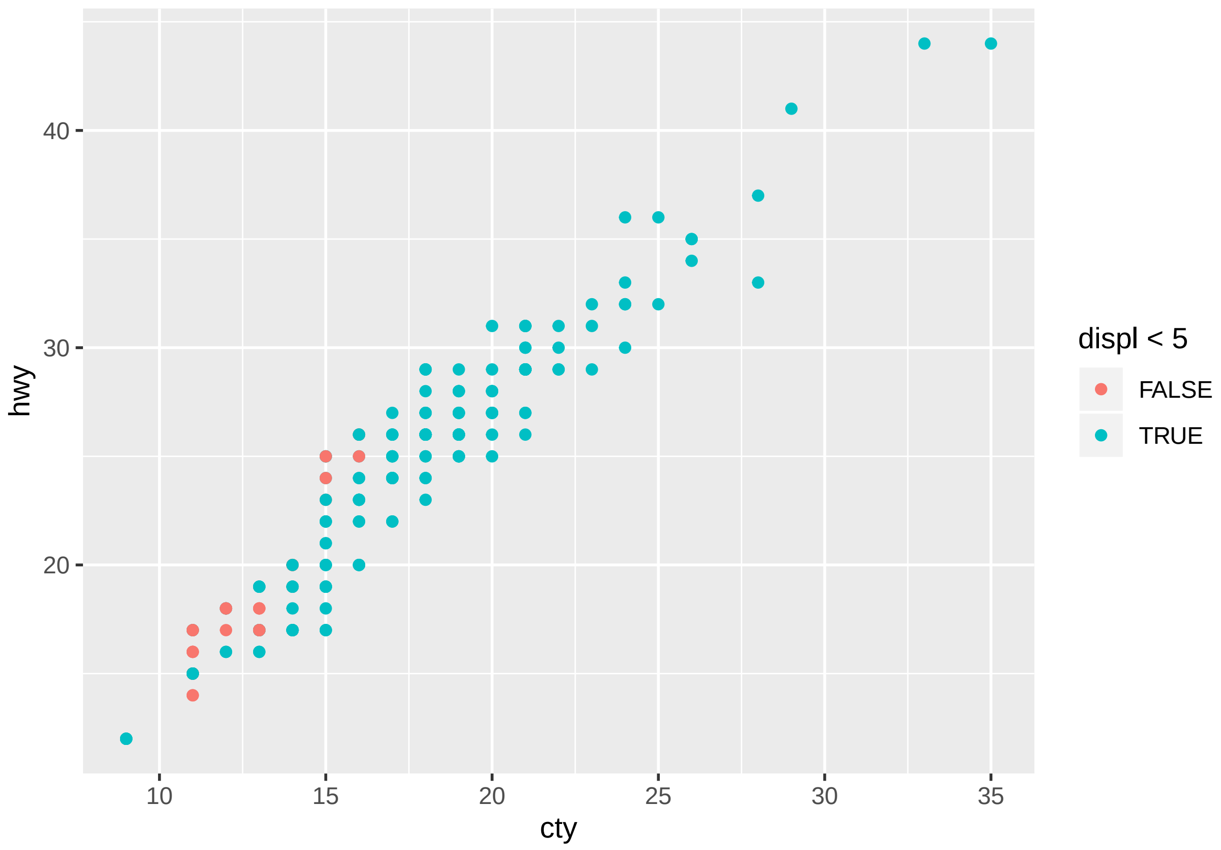
```
ggplot(data=mpg)+geom_point(aes(x=cty,y=cty),stroke=1)
```



Answer It will form a 45 degree line.

6. What happens if you map an aesthetic to something other than a variable name, like aes(colour = displ < 5)? Note, you'll also need to specify x and y.

```
ggplot(data=mpg)+geom_point(aes(x=cty,y=hwy,color=displ<5))
```



Answer

It distinguishs data point with the displ with <5 and >5.

Common problems

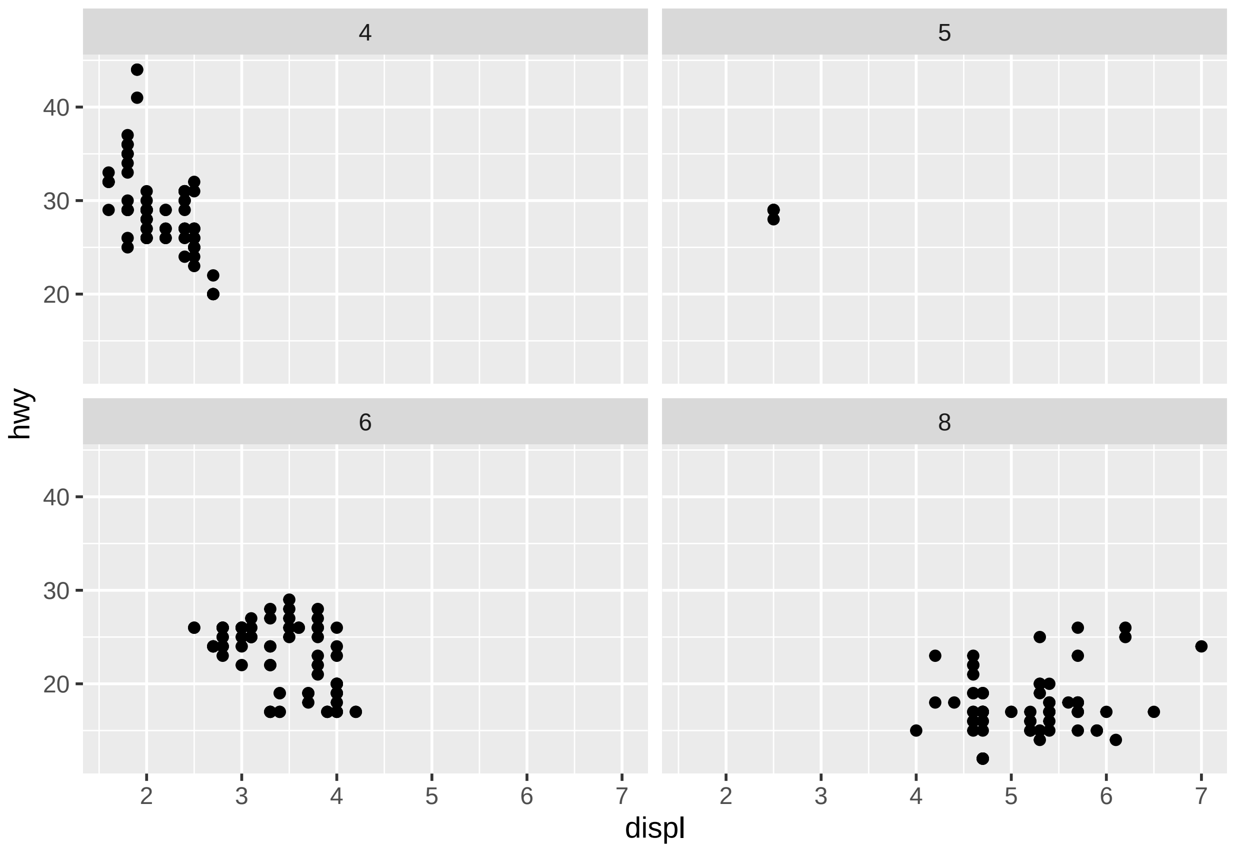
Remember to use pairness and put everything on th right position.

Facets

Facets is particularly useful for categorical variables, is to split your plot into facets.

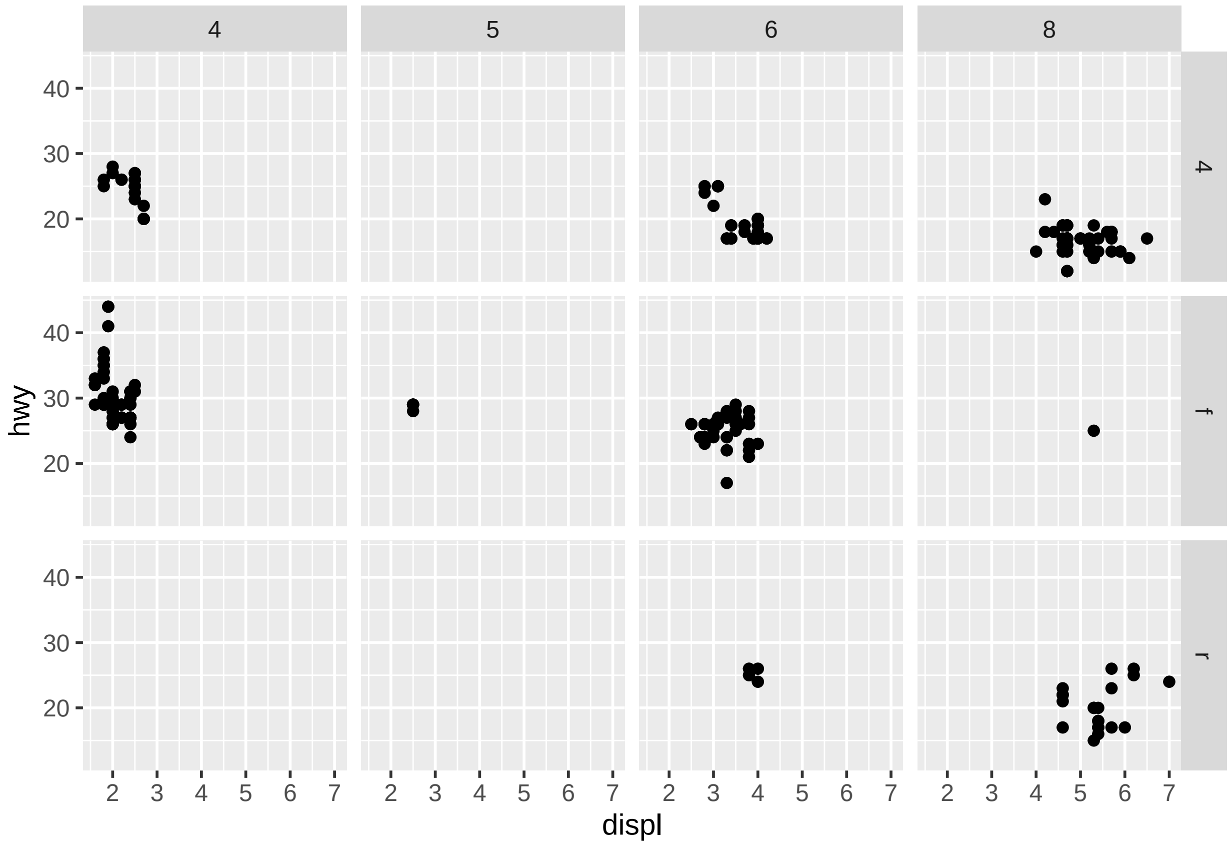
`facet_wrap`: first argument is `~`, which is followed by a variable name. the variable you pass to `facet_wrap` must be discrete

```r
ggplot(data=mpg)+
  geom_point(mapping = aes(x=displ,y=hwy))+
  facet_wrap(~cyl,nrow=2)
```



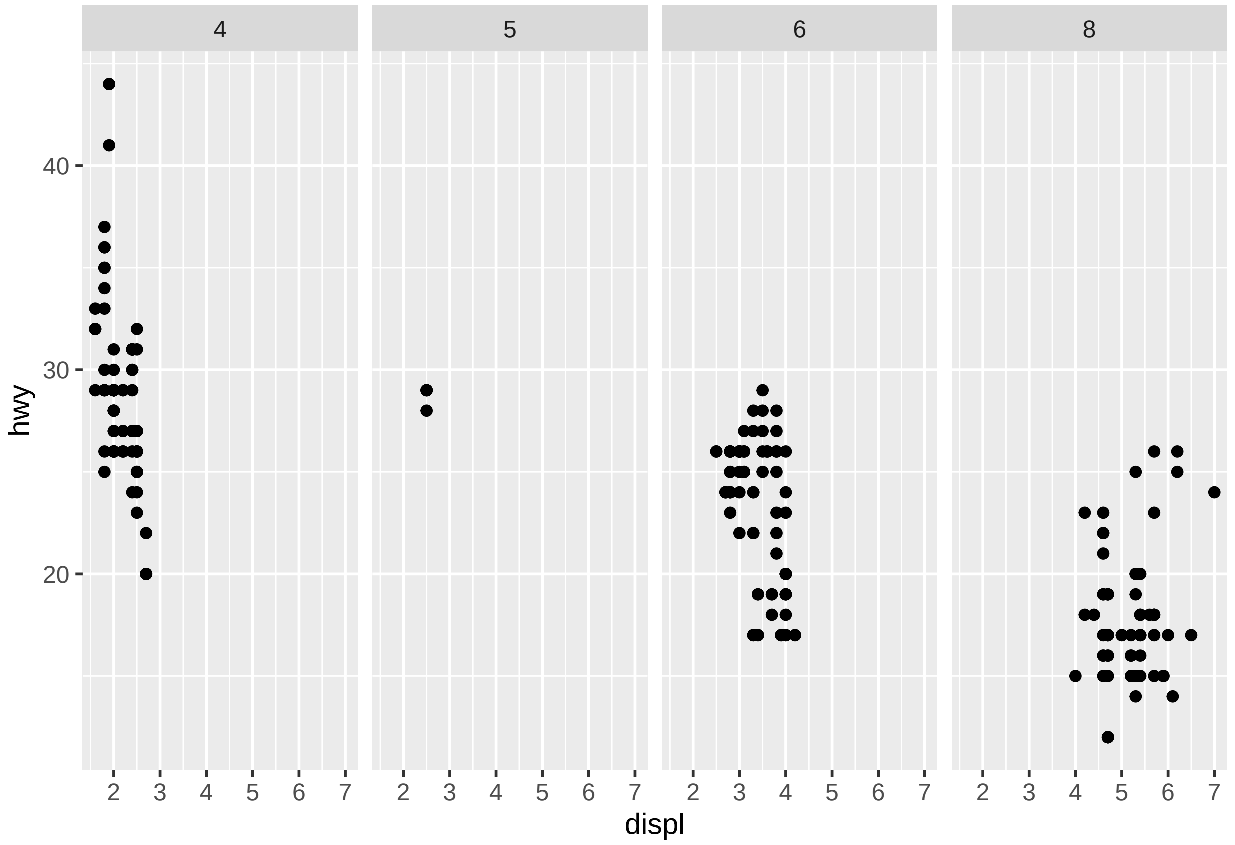
`facet_grid()` is contain two discrete variables ~

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



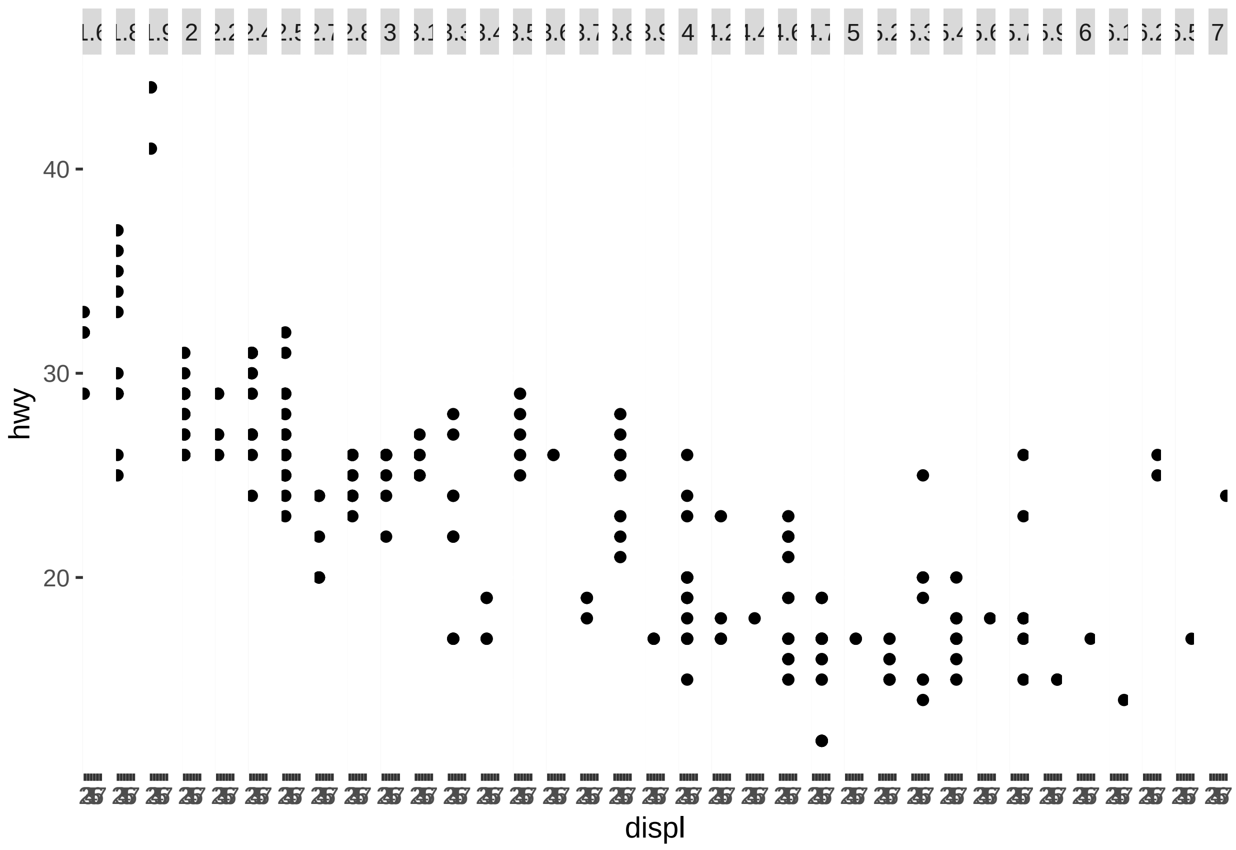
If you prefer to not facet in the rows or columns dimension.

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



Exercises

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

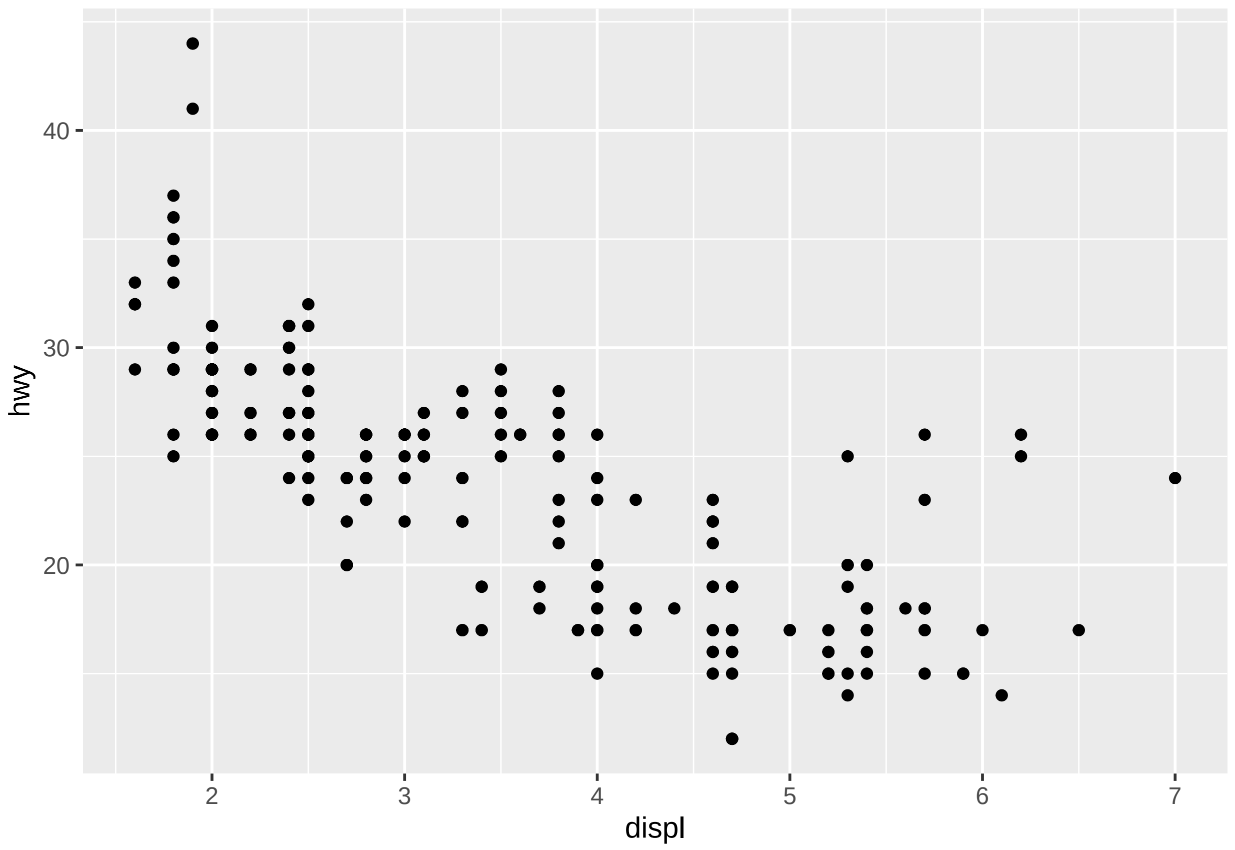


It actually will generate each category for each number of continuous variable

Geometric objects

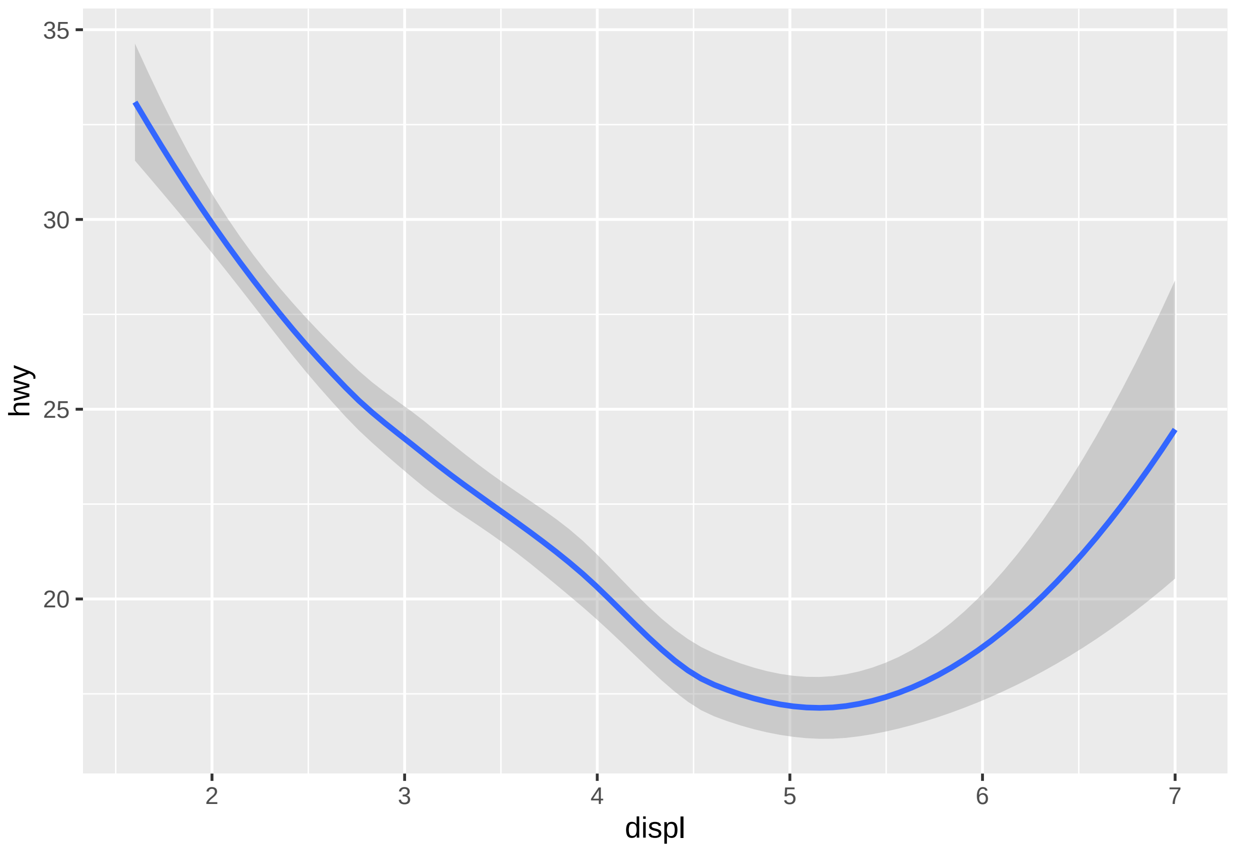
Geom

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



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

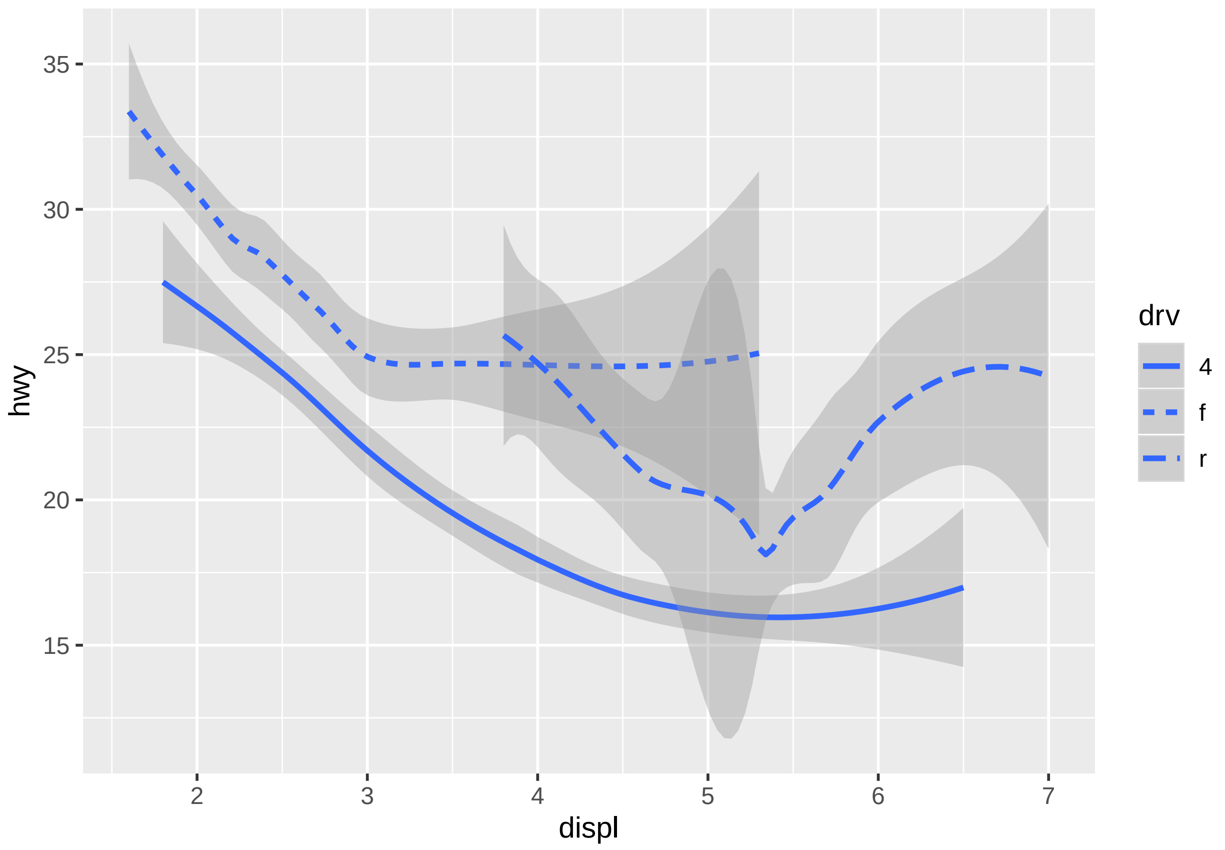
```
## `geom_smooth()` using method = 'loess' and formula 'y ~ x'
```



Every geom function in ggplot2 takes a `mapping` argument. However, not every aesthetic works with every geom.You could set shape of the points but not shape of the line. On the other hand, you could set the linetype of a line. `geom_smooth()` will draw a different line, with different linetype.

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

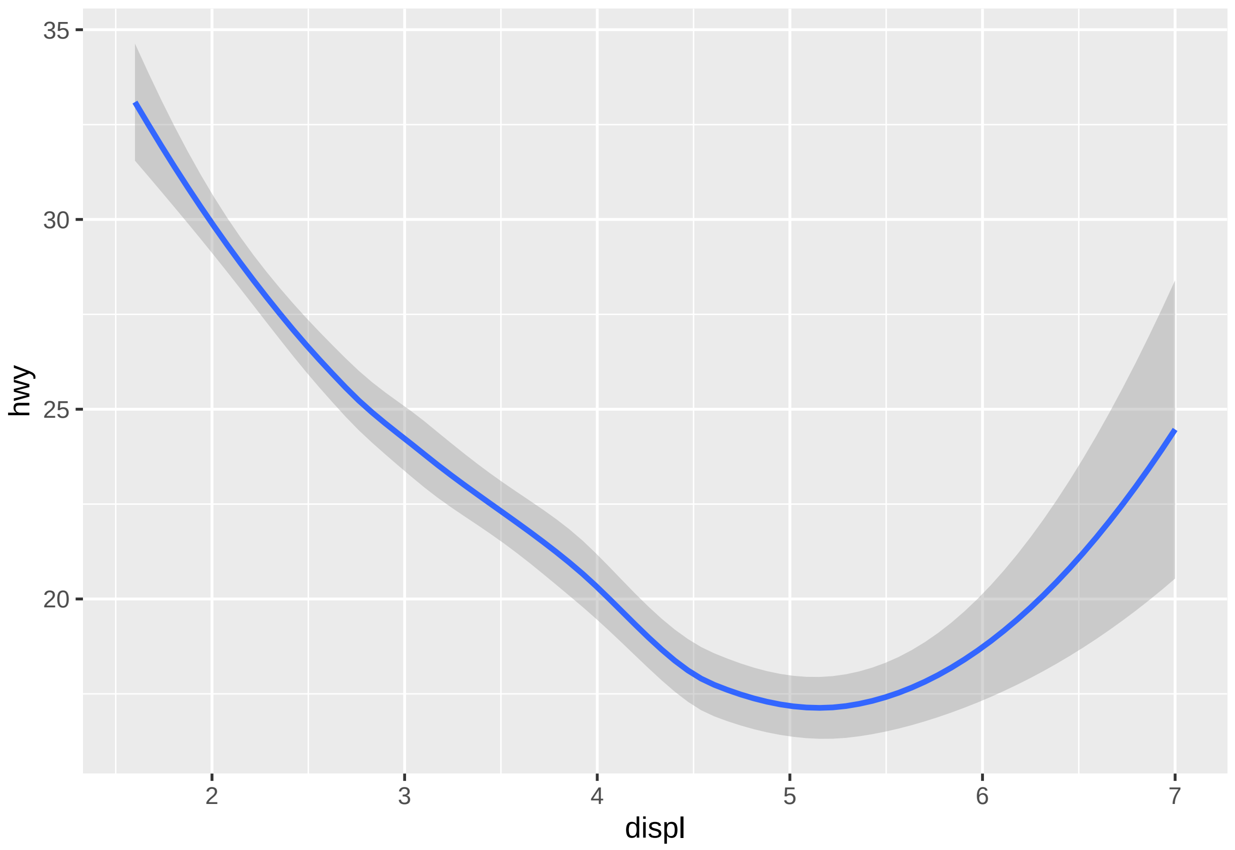
```
## `geom_smooth()` using method = 'loess' and formula 'y ~ x'
```



It separates the cars into three lines based on their `drv` value, which describes a car's drivetrain.

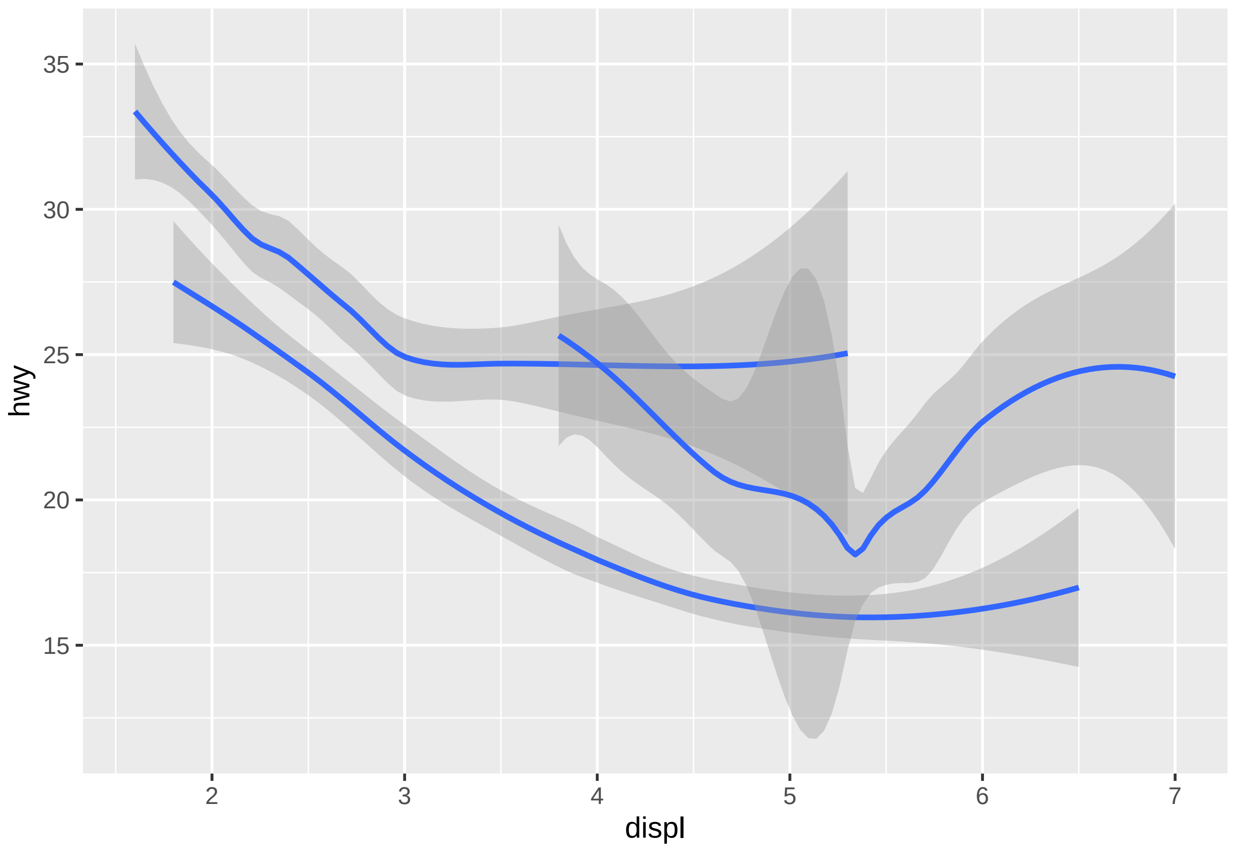
```
ggplot(data = mpg) +
  geom_smooth(mapping = aes(x = displ, y = hwy))
```

```
## `geom_smooth()` using method = 'loess' and formula 'y ~ x'
```



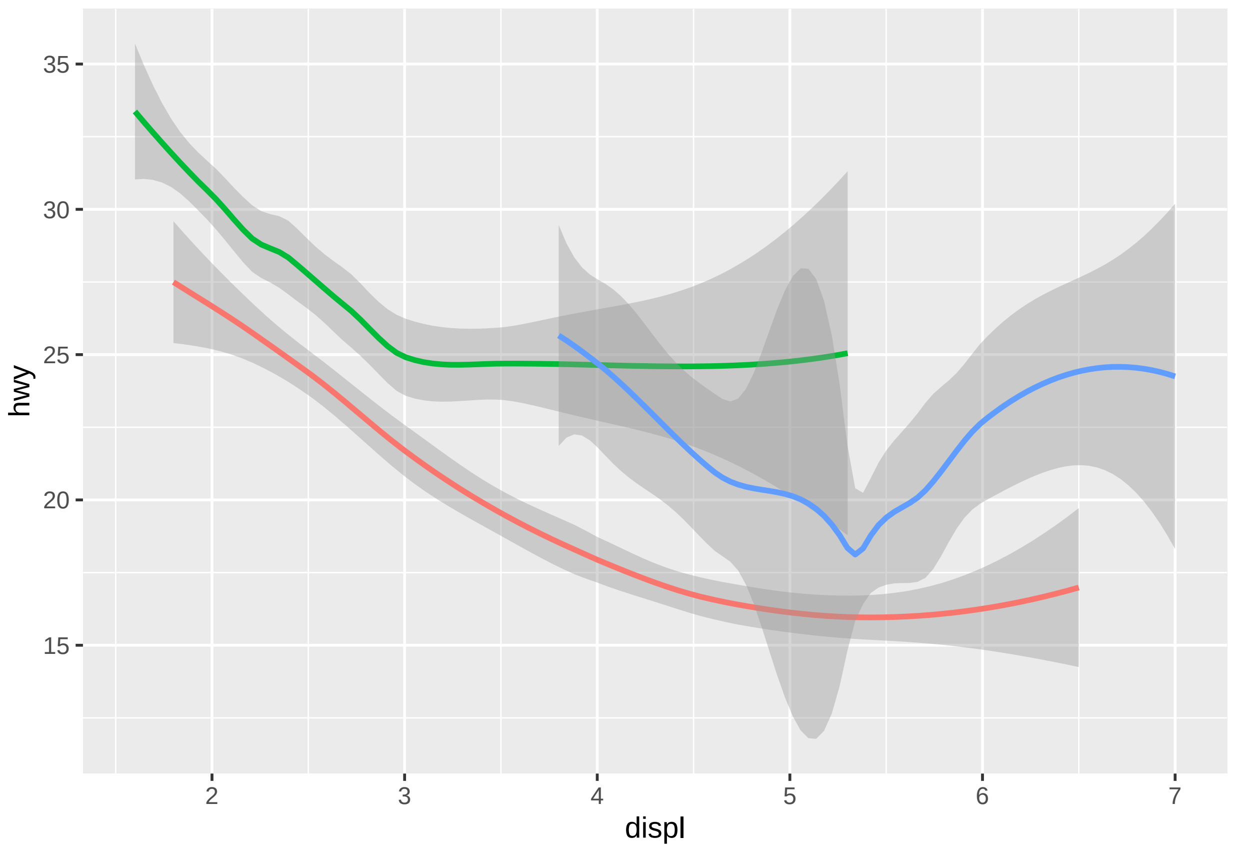
```
ggplot(data = mpg) +
  geom_smooth(mapping = aes(x = displ, y = hwy, group = drv))
```

```
## `geom_smooth()` using method = 'loess' and formula 'y ~ x'
```



```
ggplot(data = mpg) +
  geom_smooth(
    mapping = aes(x = displ, y = hwy, color = drv),
    show.legend = FALSE
  )
```

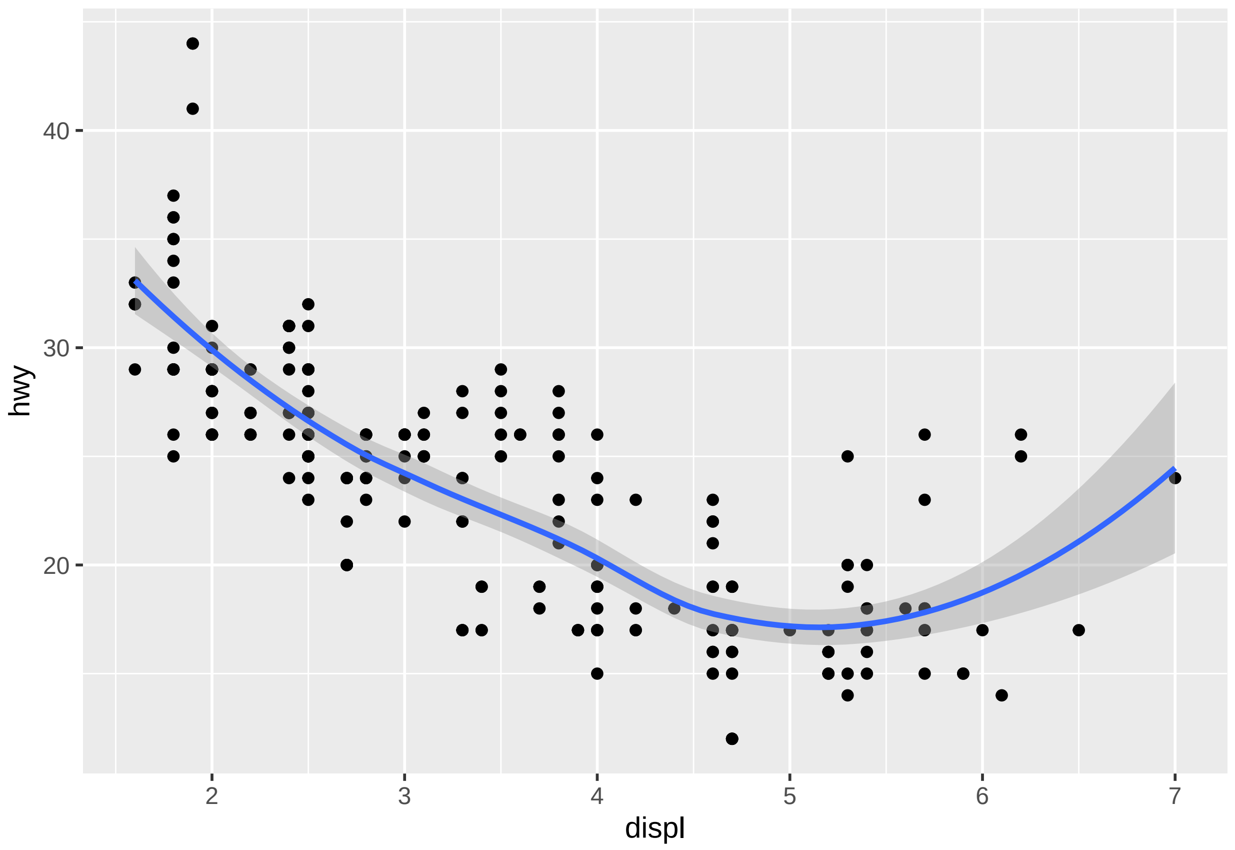
```
## `geom_smooth()` using method = 'loess' and formula 'y ~ x'
```



To display multiple geoms in the same plot, add multiple geom functions to `ggplot()`:

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

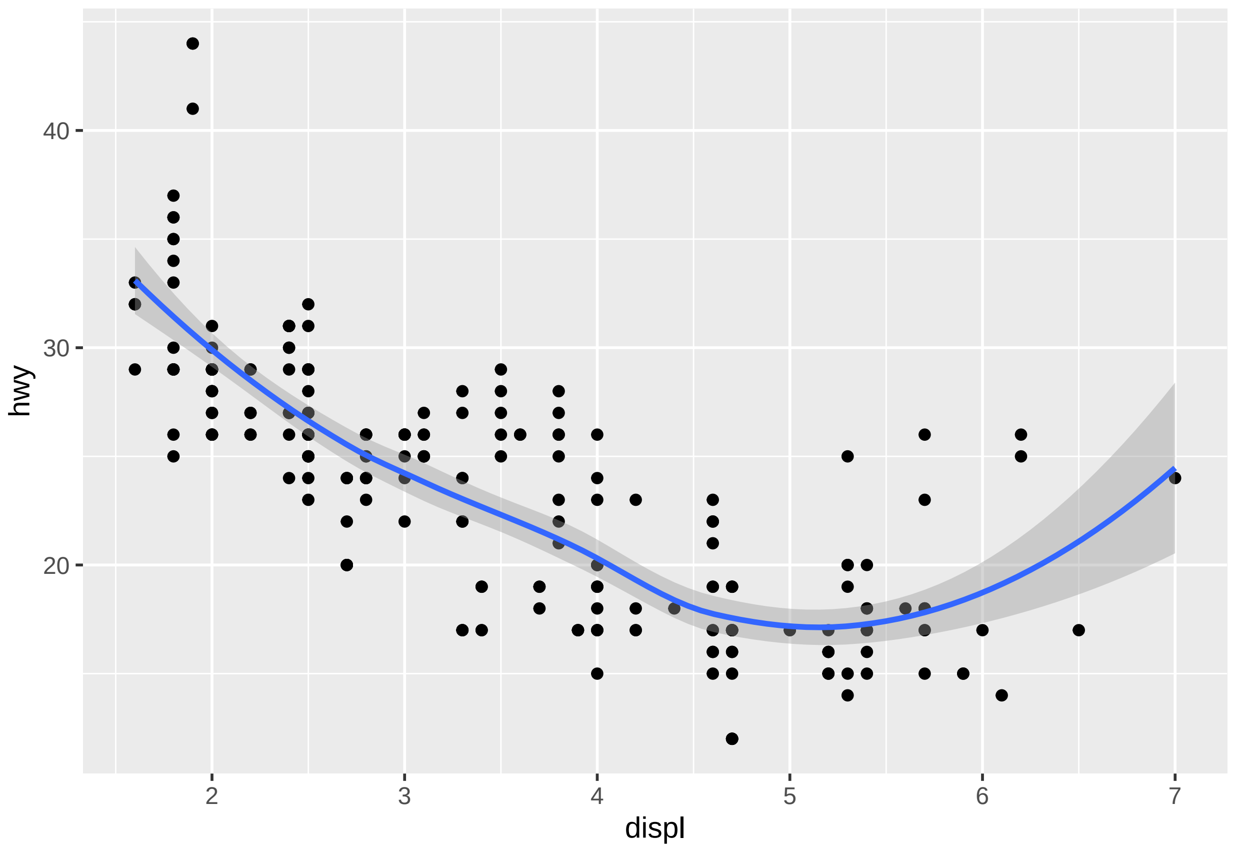
```
## `geom_smooth()` using method = 'loess' and formula 'y ~ x'
```



Or equivalently,

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

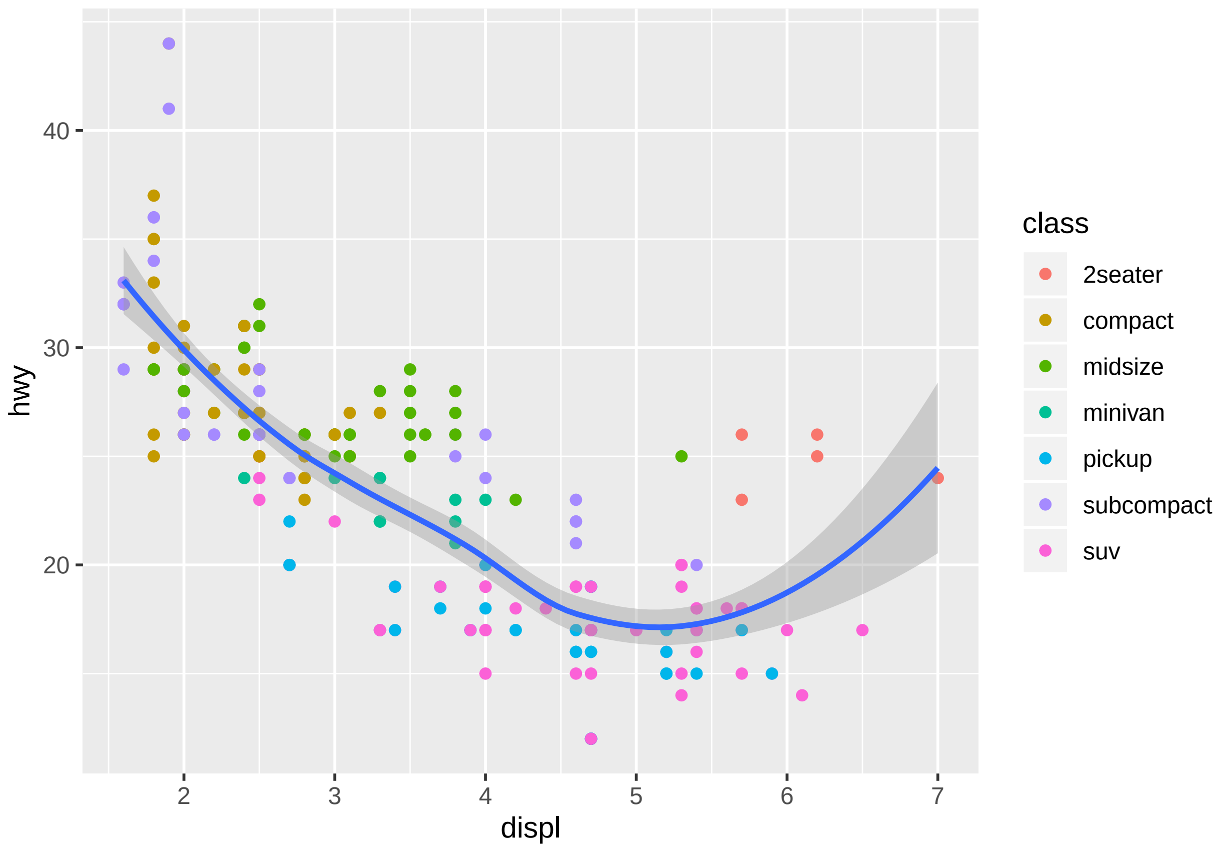
```
## `geom_smooth()` using method = 'loess' and formula 'y ~ x'
```



We can also add the color

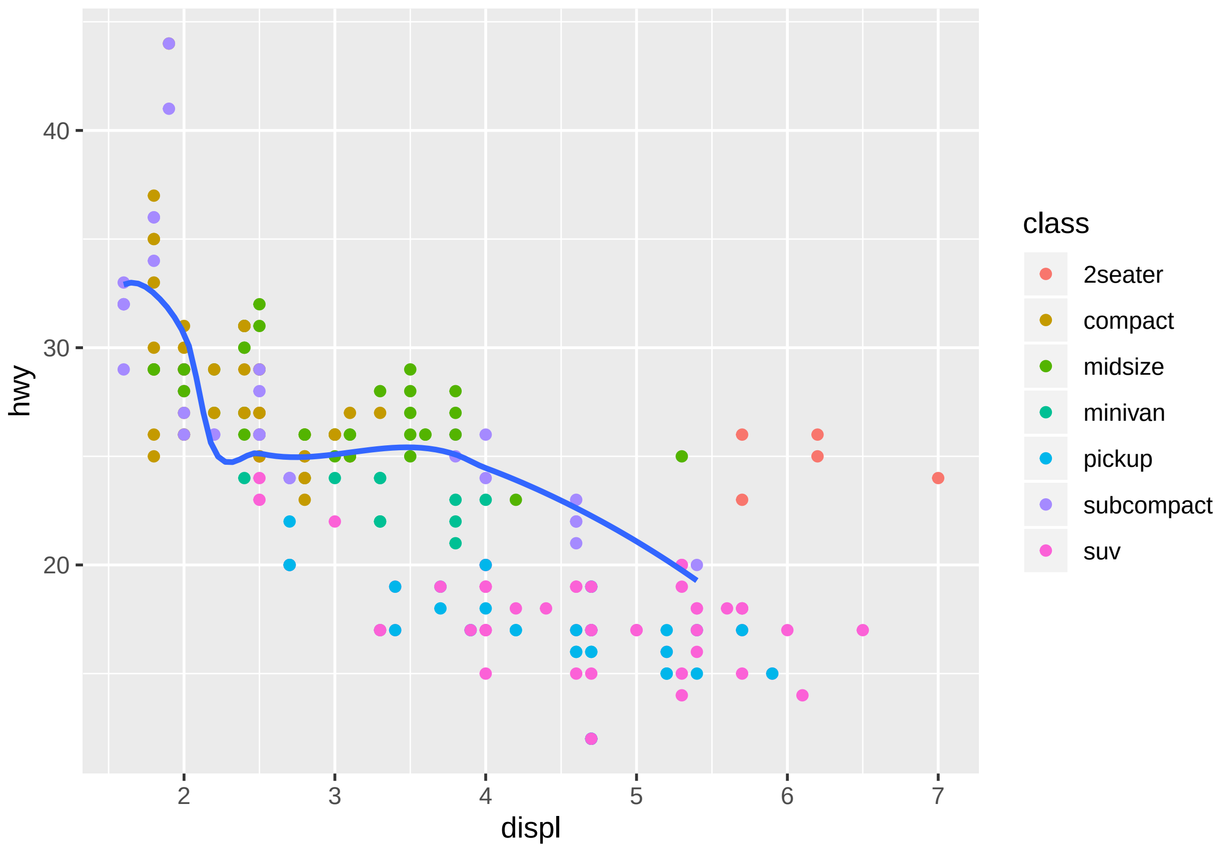
```
ggplot(data=mpg,mapping=aes(x=displ,y=hwy))+
  geom_point(mapping = aes(color=class))+
  geom_smooth()
```

```
## `geom_smooth()` using method = 'loess' and formula 'y ~ x'
```



```
ggplot(data=mpg,mapping=aes(x=displ,y=hwy))+
  geom_point(mapping=aes(color=class))+
  geom_smooth(data=filter(mpg,class=="subcompact"),se=FALSE)
```

```
## `geom_smooth()` using method = 'loess' and formula 'y ~ x'
```



Just plot the data which class=cubcompact

Exercises

1. What geom would you use to draw a line chart? A boxplot? A histogram? An area chart?

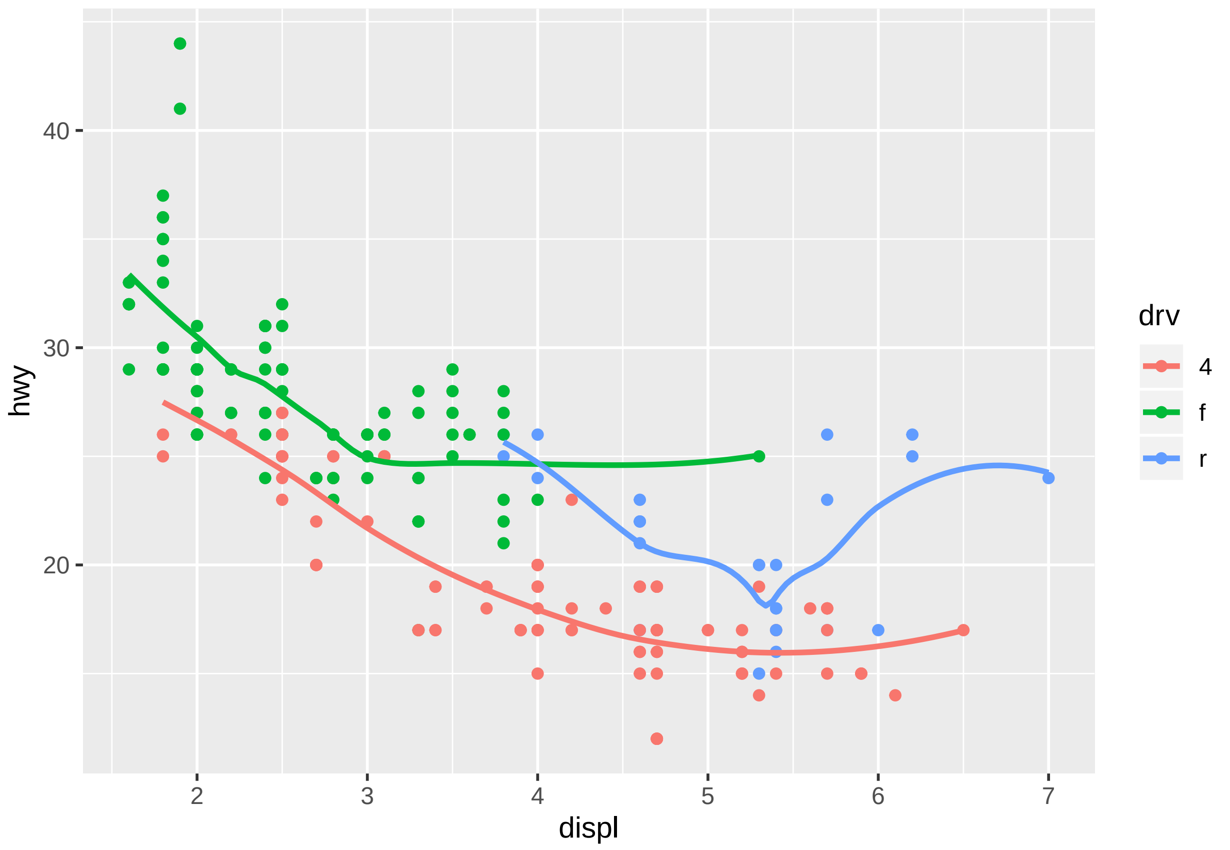
Answer

geom_line(), geom_box(), geom_histogram()

2. Running this code predict what the output will look like. Then, run the code in R and check your predictions

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

```
## `geom_smooth()` using method = 'loess' and formula 'y ~ x'
```



Answer

se=false means no shaded areas.

3. What does `show.legend=FALSE` do? What happens if you remove it? Why do you think I used it ealier in the chapter?

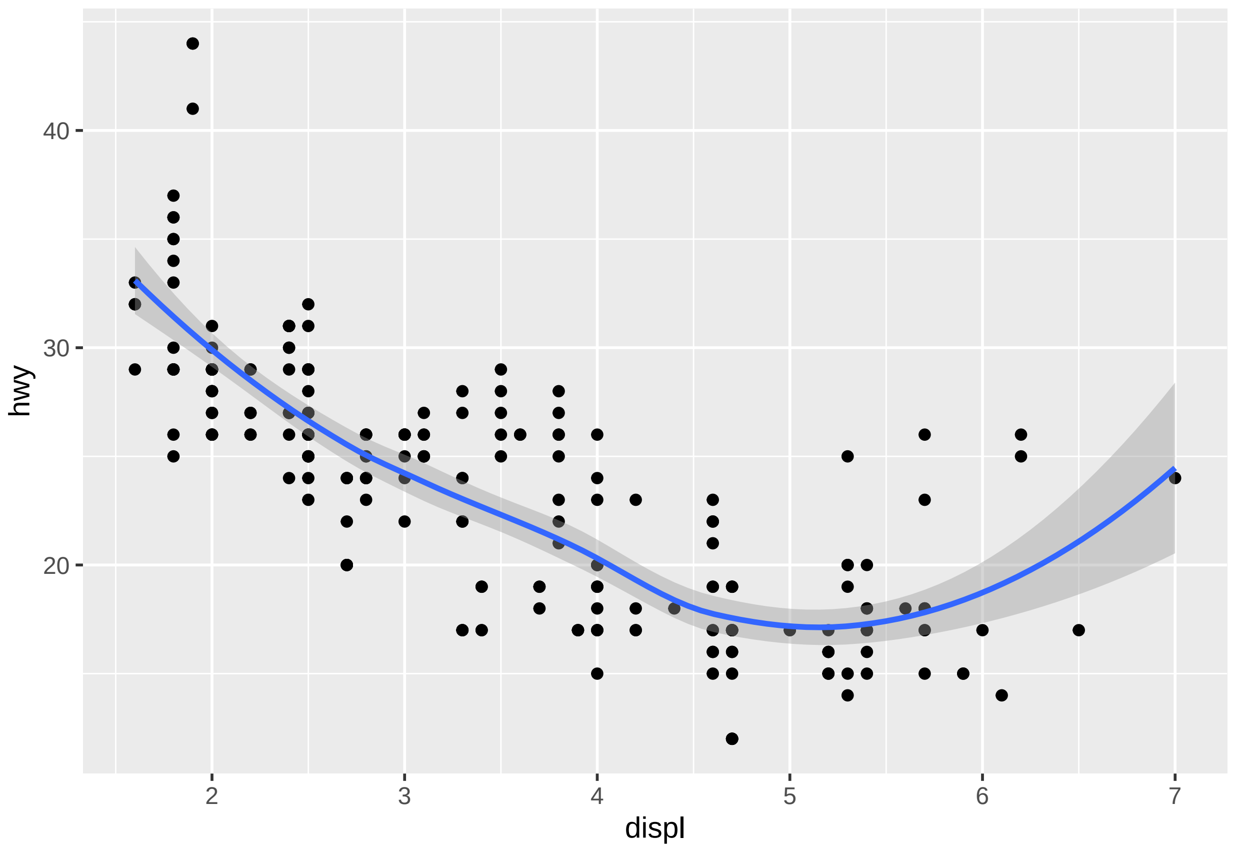
Answer

It means the graph do not show the upper right corner of the tag.

4. Will these graphs look different? Why/why not?

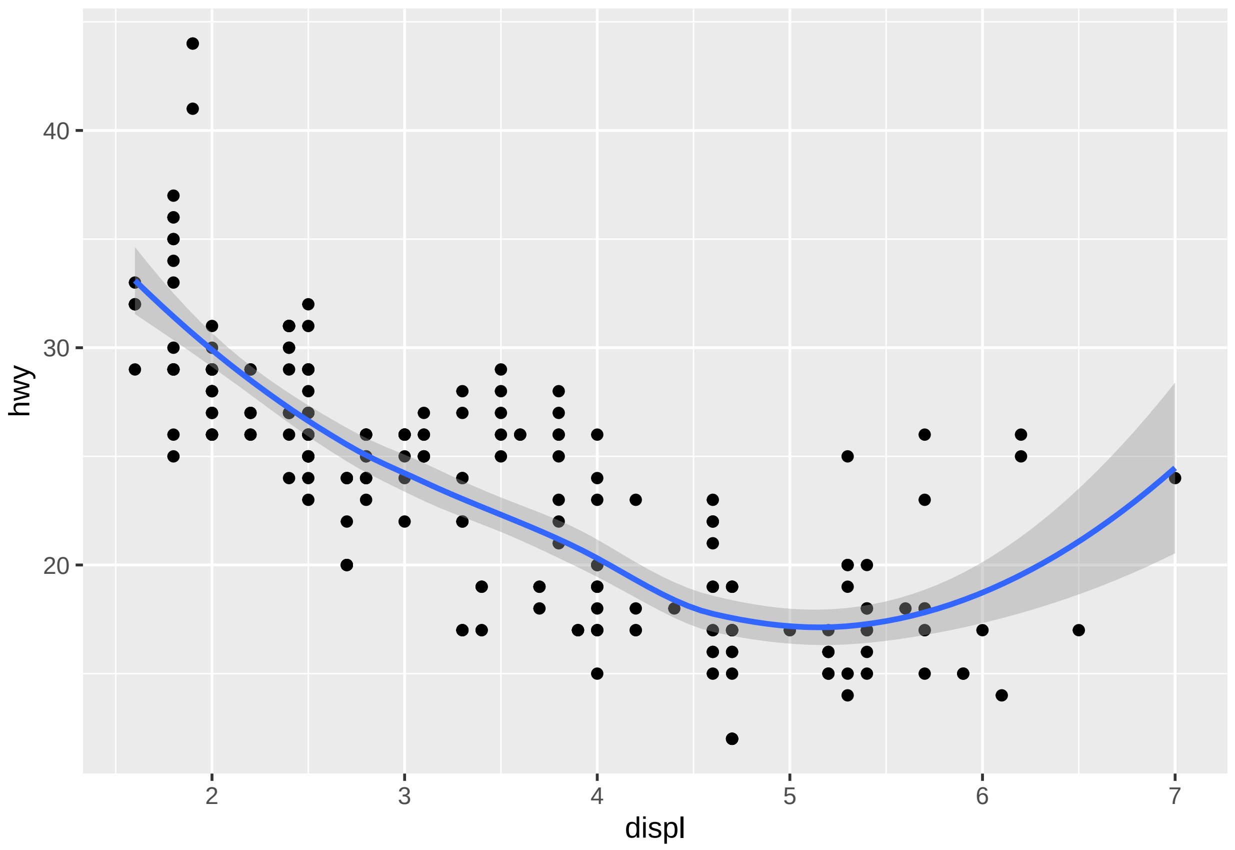
```r
ggplot(data = mpg, mapping = aes(x = displ, y = hwy)) +
  geom_point() +
  geom_smooth()
```

```
## `geom_smooth()` using method = 'loess' and formula 'y ~ x'
```



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

```
## `geom_smooth()` using method = 'loess' and formula 'y ~ x'
```



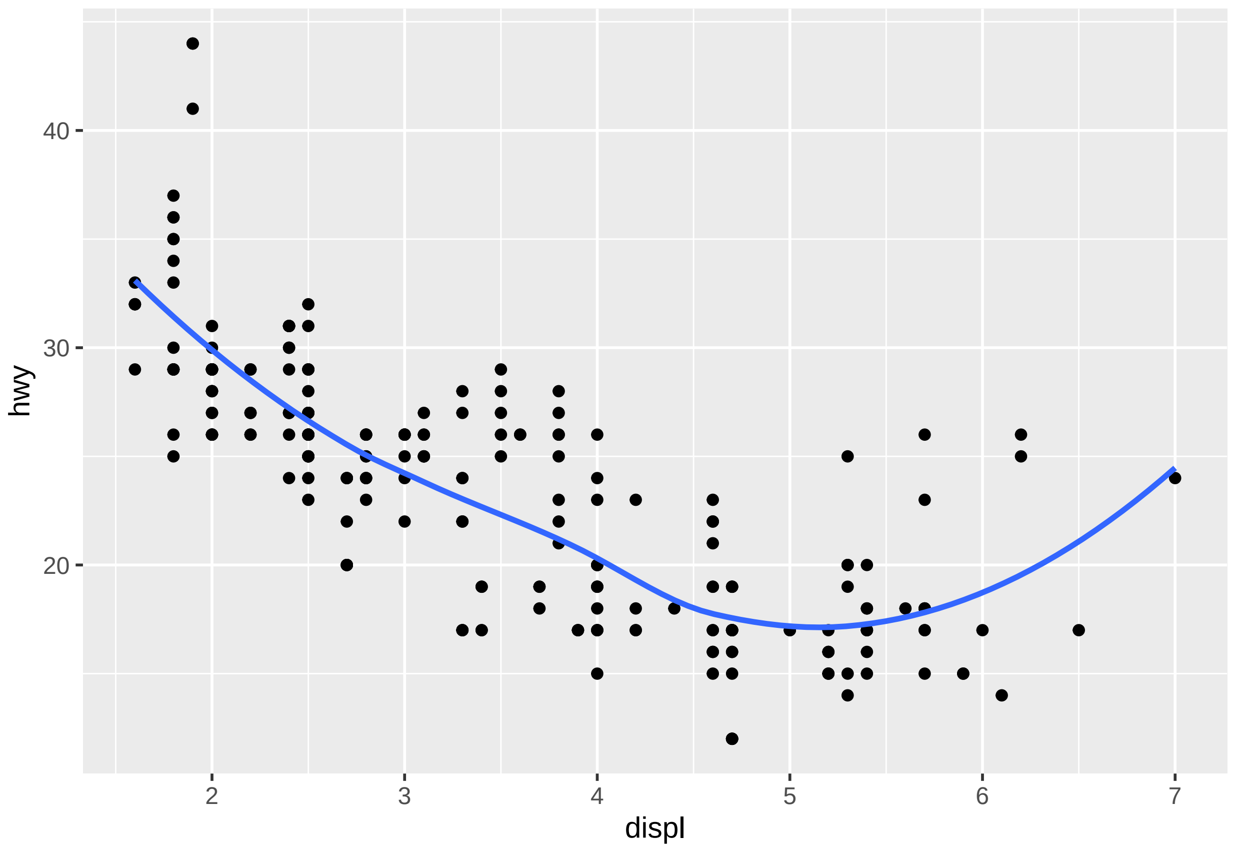
Answer

Nothing is different. Since it is same to put different positions.

6. Recreate the R code necessary to generate the following grpahs.

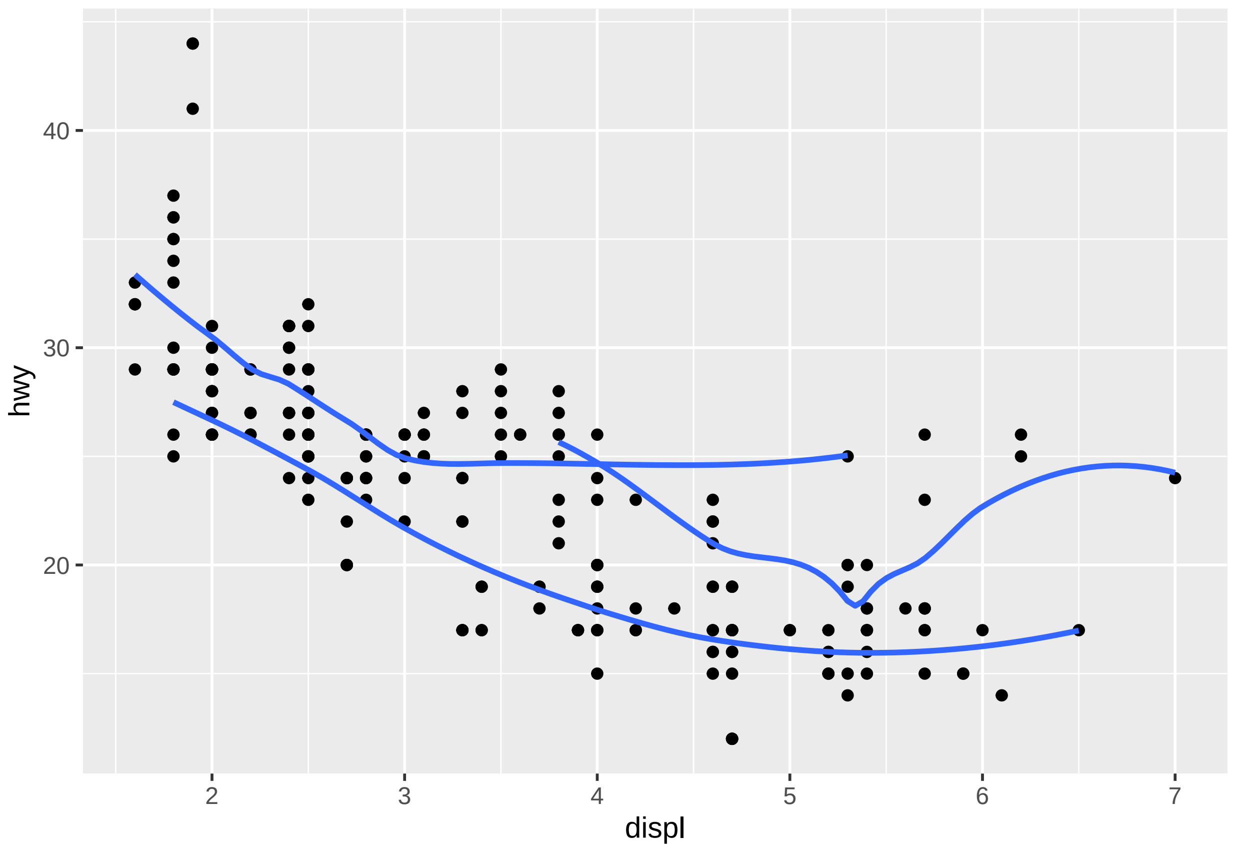
```r
ggplot(data=mpg,mapping = aes(x=displ,y=hwy))+
  geom_point()+
  geom_smooth(se=F)
```

```
## `geom_smooth()` using method = 'loess' and formula 'y ~ x'
```



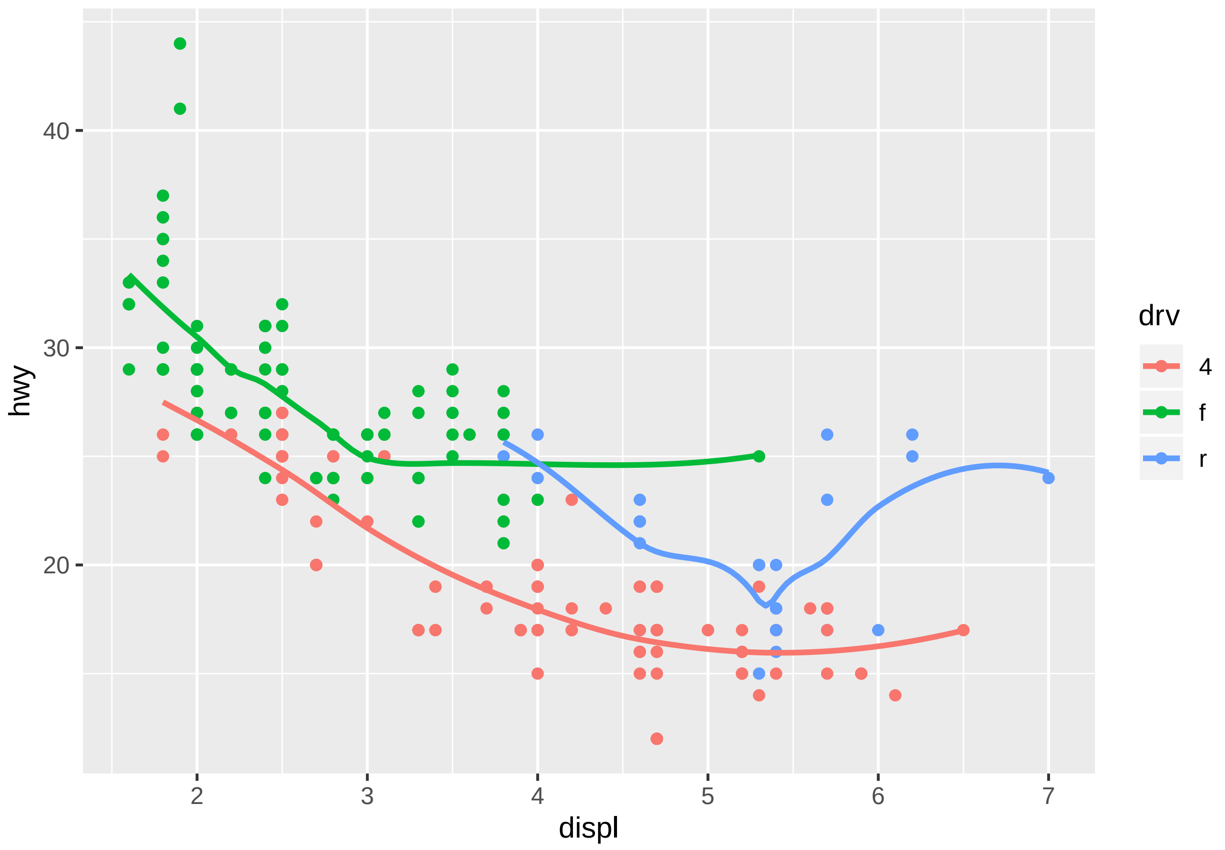
```r
ggplot(data=mpg,mapping=aes(x=displ,y=hwy,group=drv))+
  geom_point()+
  geom_smooth(se=F)
```

```
## `geom_smooth()` using method = 'loess' and formula 'y ~ x'
```



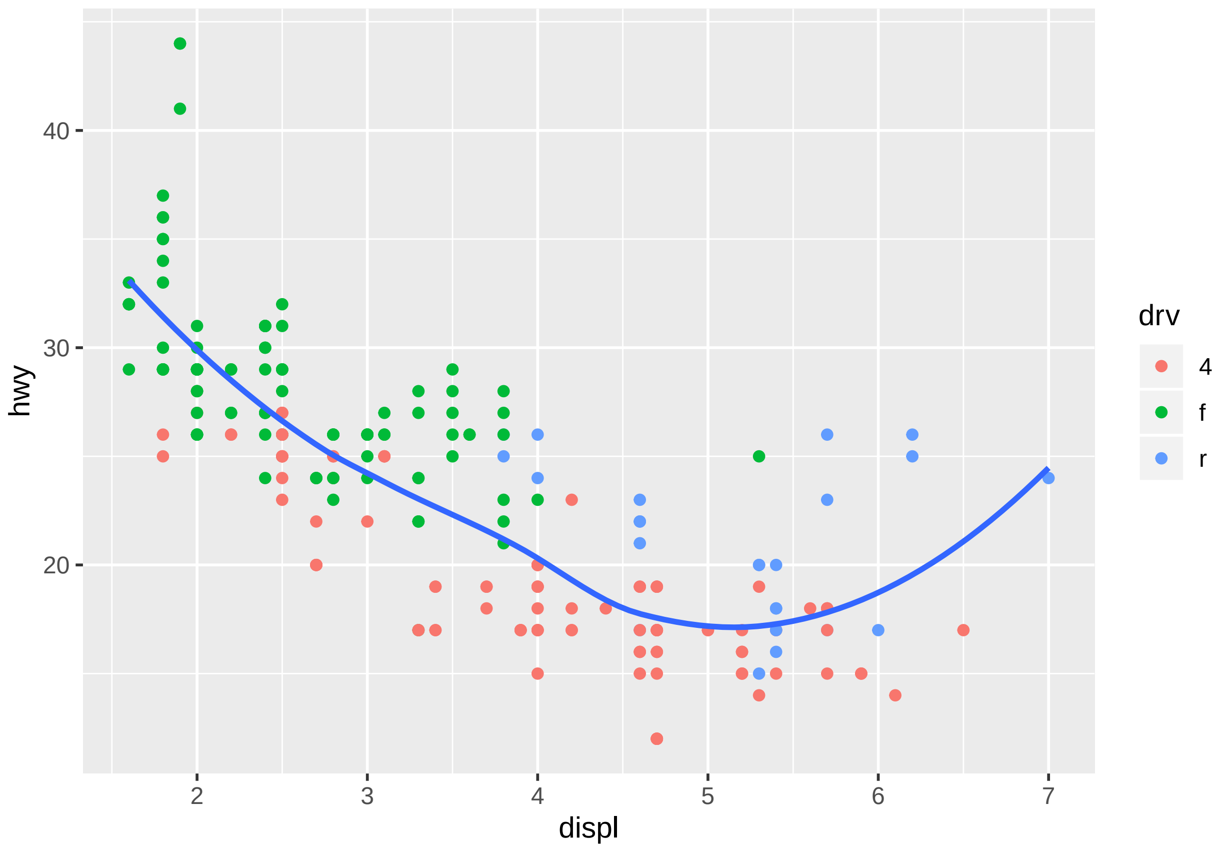
```
ggplot(data=mpg,mapping=aes(x=displ,y=hwy))+
  geom_point(mapping = aes(color=drv))+
  geom_smooth(se=F,mapping=aes(color=drv))
```

```
## `geom_smooth()` using method = 'loess' and formula 'y ~ x'
```



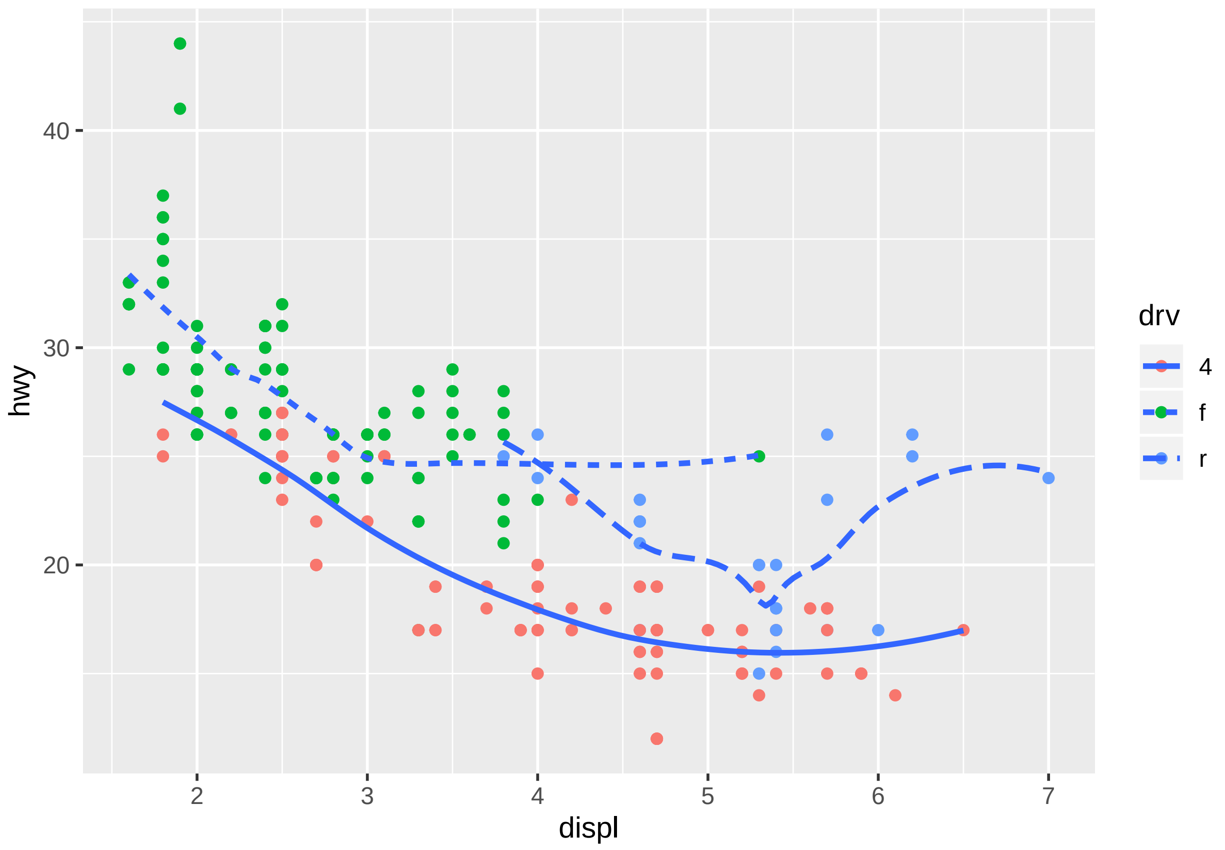
```r
ggplot(data=mpg,mapping=aes(x=displ,y=hwy))+
  geom_point(mapping = aes(color=drv))+
  geom_smooth(se=F)
```

```
## `geom_smooth()` using method = 'loess' and formula 'y ~ x'
```

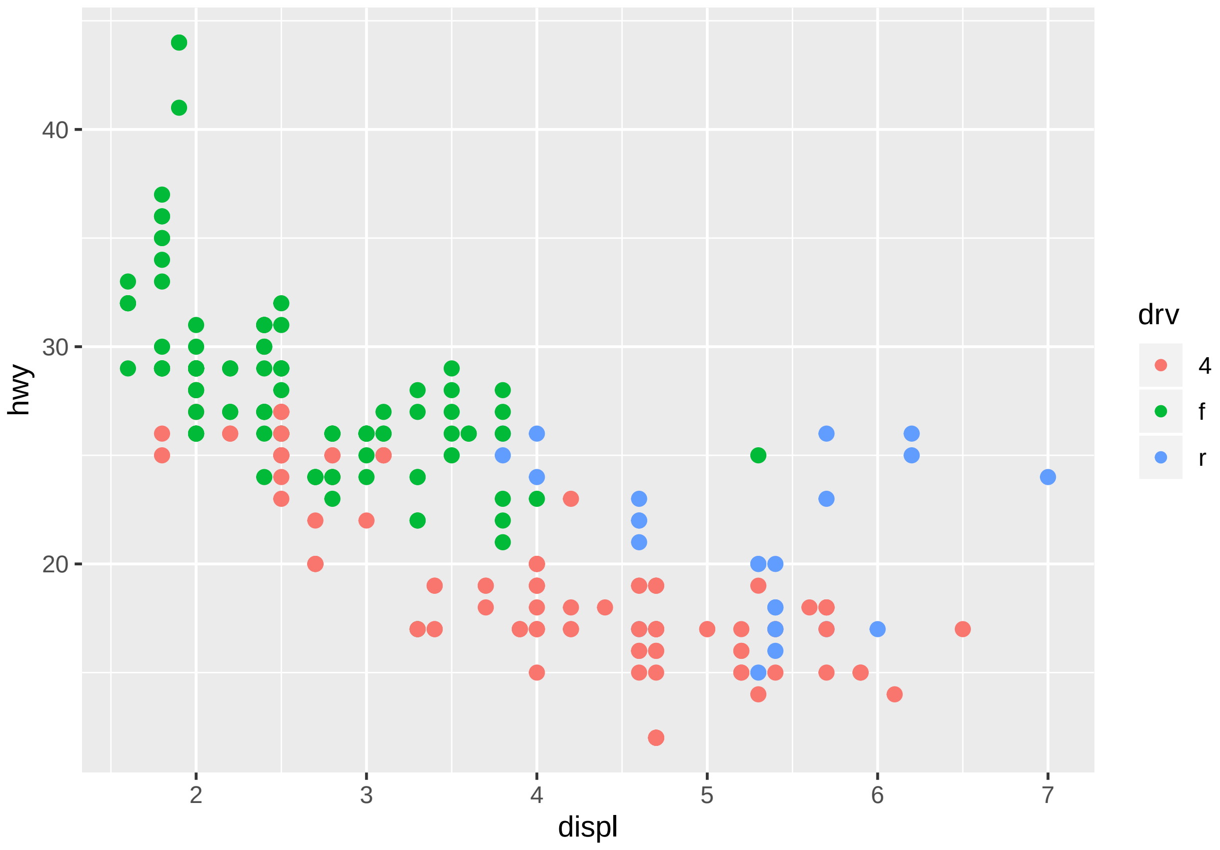


```r
ggplot(data=mpg,mapping=aes(x=displ,y=hwy,linetype=drv))+
  geom_point(mapping = aes(color=drv))+
  geom_smooth(se=F)
```

```
## `geom_smooth()` using method = 'loess' and formula 'y ~ x'
```



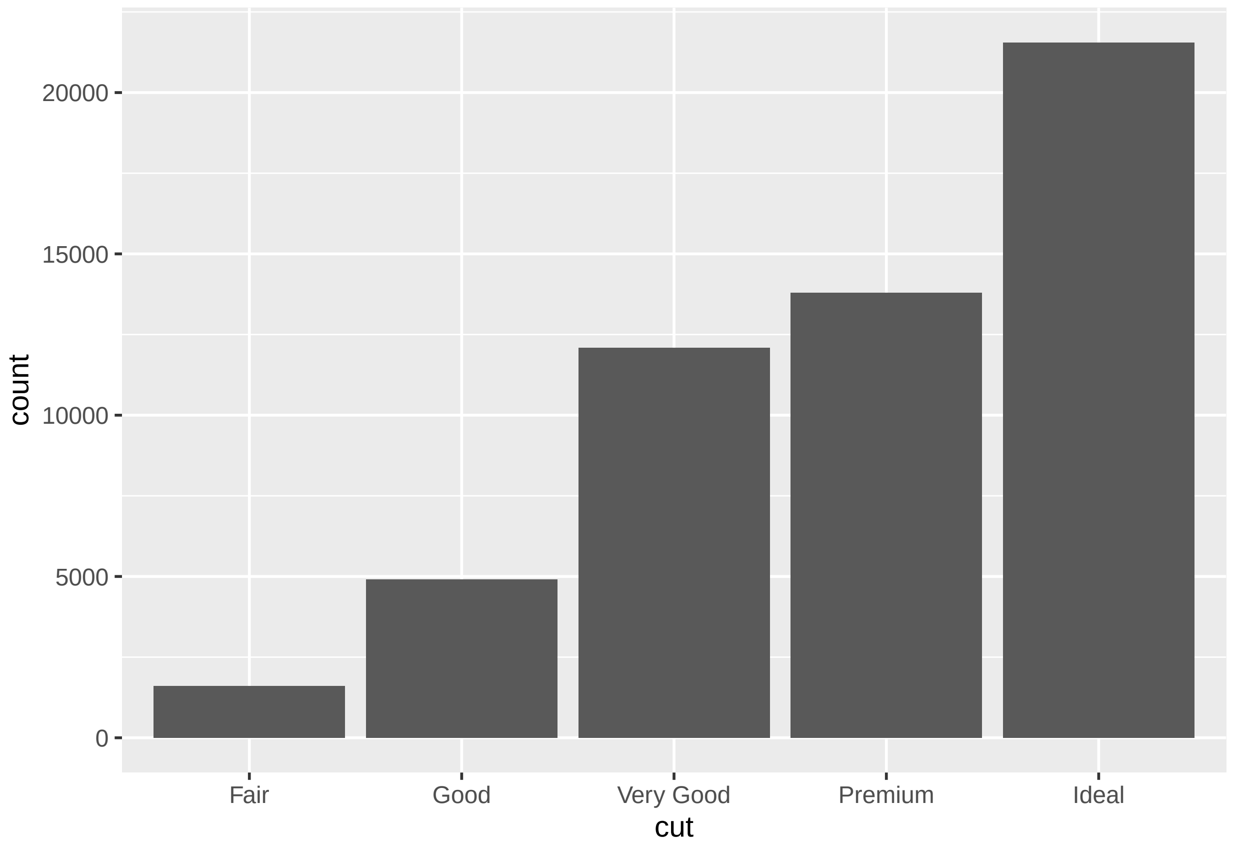
```r
ggplot(data=mpg,mapping=aes(x=displ,y=hwy))+
  geom_point(mapping = aes(color=drv,stroke=1))
```



Statistical transformations

Firstly look at the bar chart, `geom_bar()`. We can group diamonds by cut. The chart shows that more diamonds are available with high quality cuts than with low quality cuts.

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

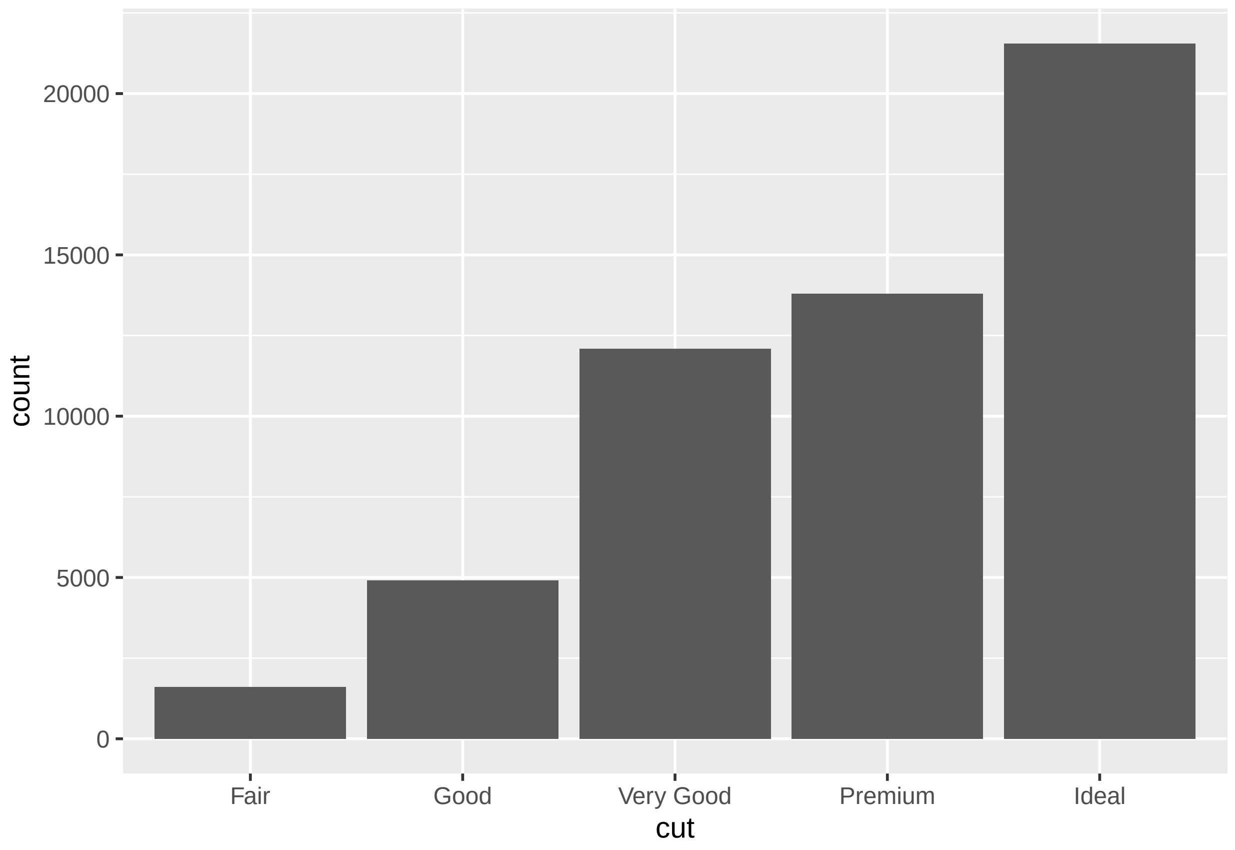


x-axis: `cut` y-axis: `count`

`stat_count()` is documented on the same page as `geom_bar`, and if you scroll down you can find a section called "Computed variables"

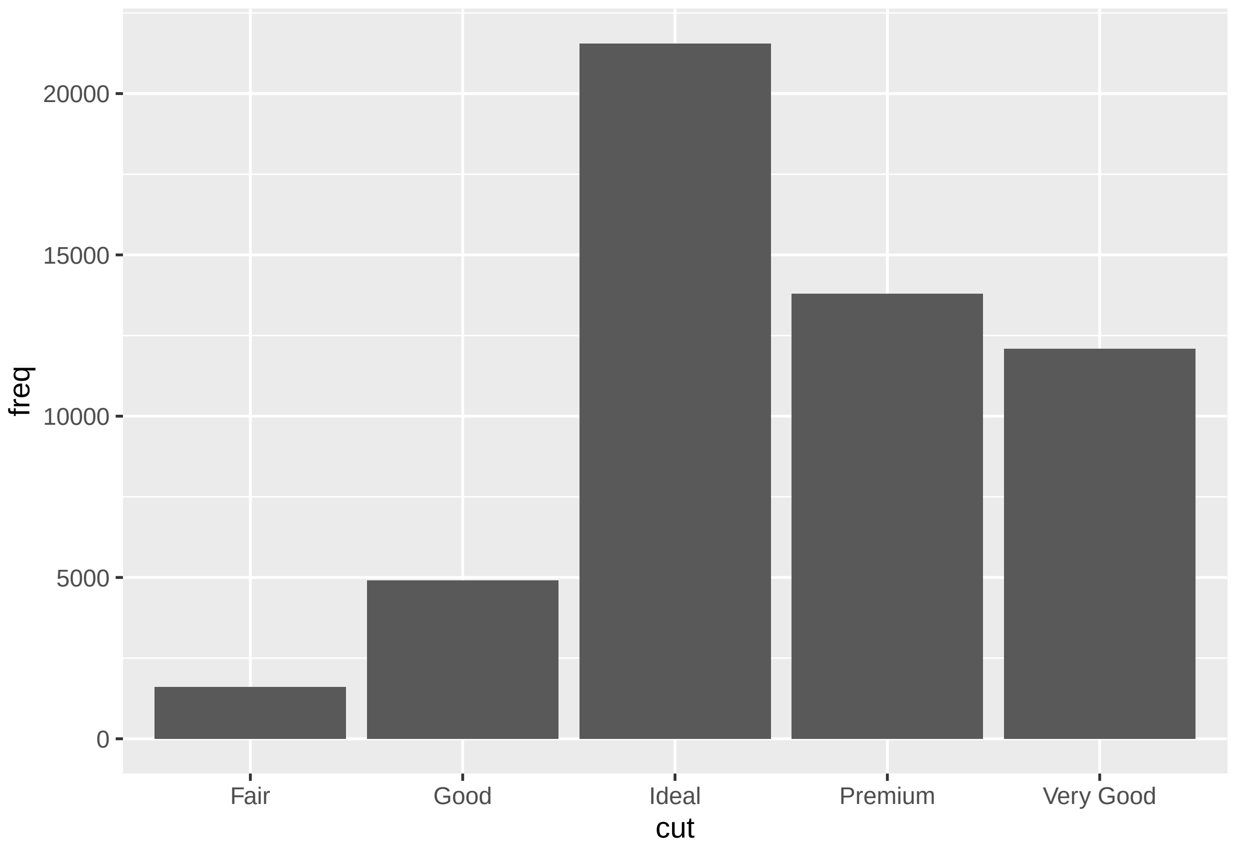
Generally, we can use `stat_count()` and `geom_bar()` interchangeably

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

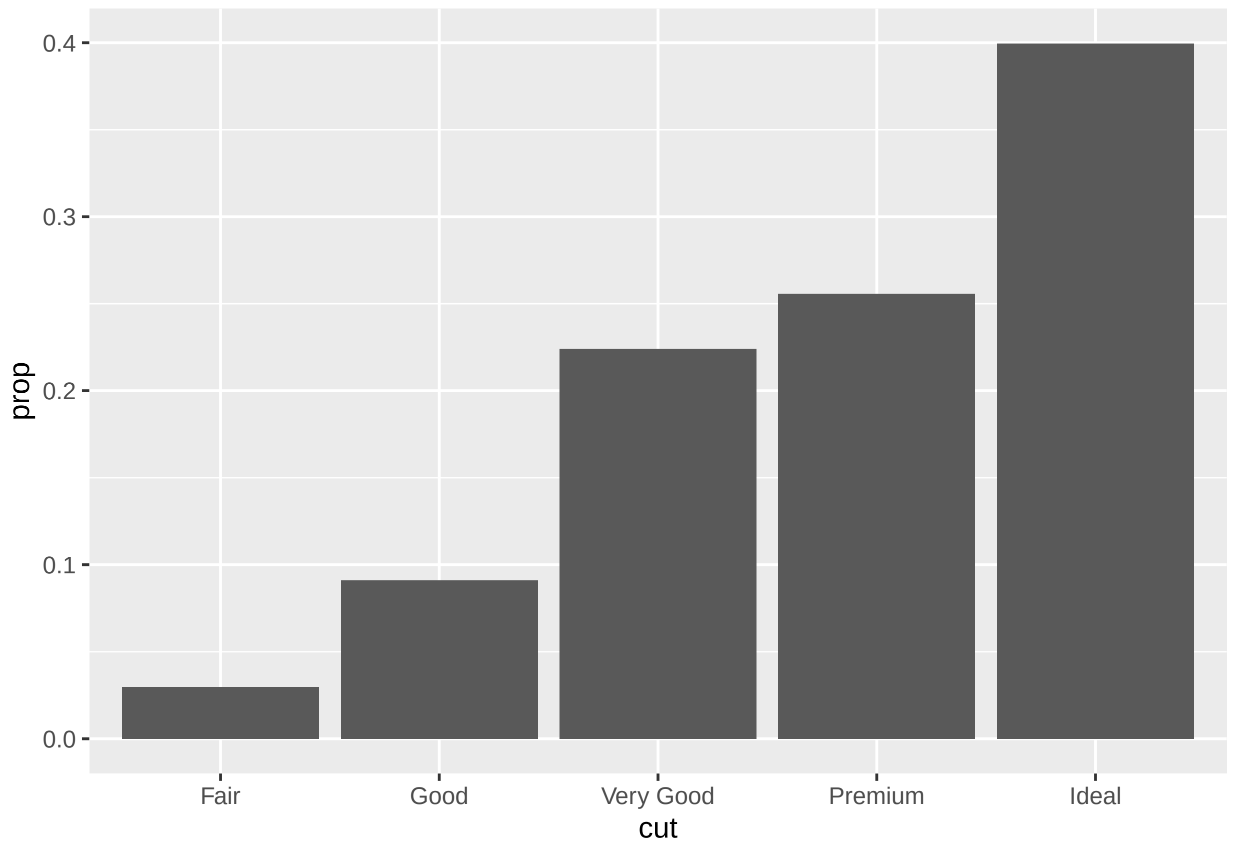


```r
demo <-tribble(
  ~cut,     ~freq,
  "Fair",     1610,
  "Good",     4906,
  "Very Good", 12082,
  "Premium",   13791,
  "Ideal",     21551
)

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

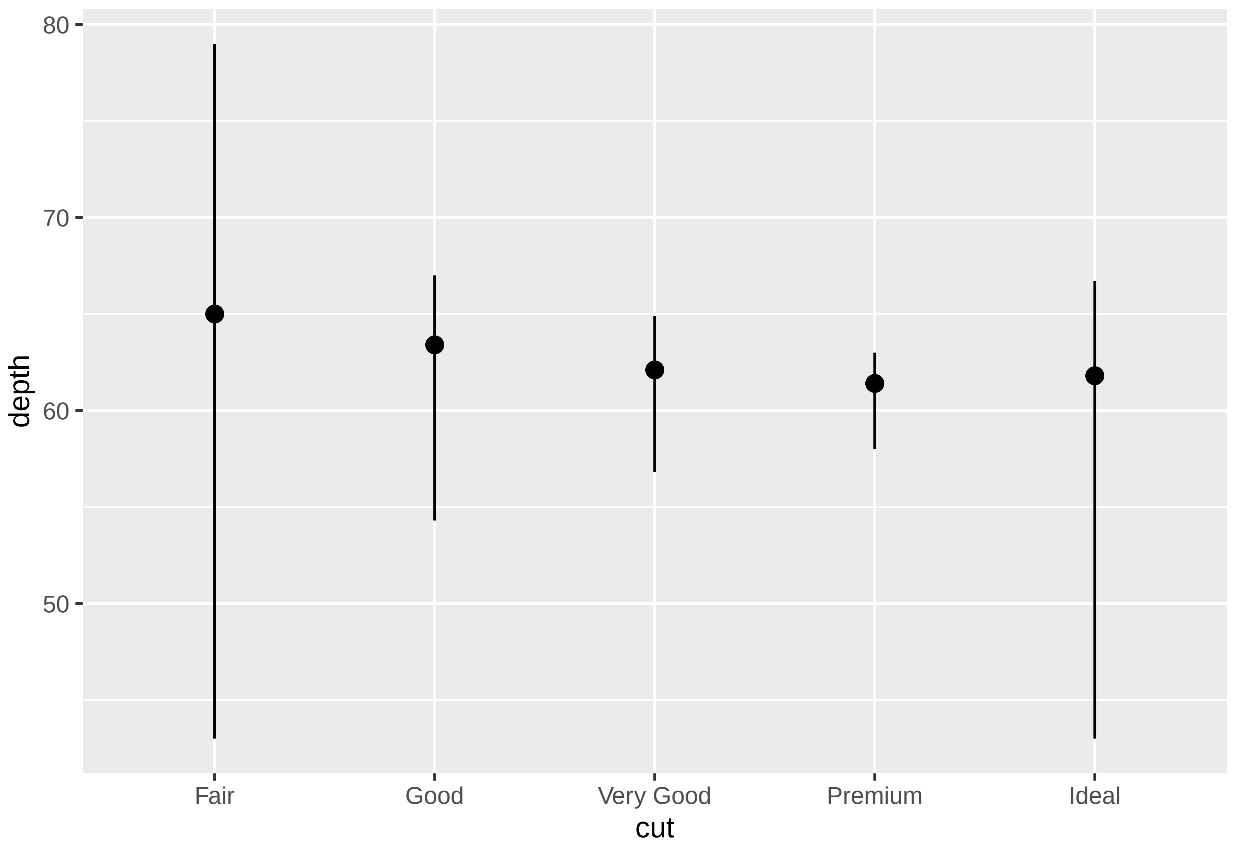


```
ggplot(data=diamonds)+
  geom_bar(mapping=aes(x=cut,y=stat(prop),group=1))
```



To draw greater attention to the statistical transformation in your code. `stat_summary()` which summarises the y values for each unique x value, to draw attention to the summary that you are computing

```
ggplot(data=diamonds)+
  stat_summary(
    mapping = aes(x=cut,y=depth),
    fun.ymin=min,
    fun.ymax=max,
    fun.y=median
  )
```

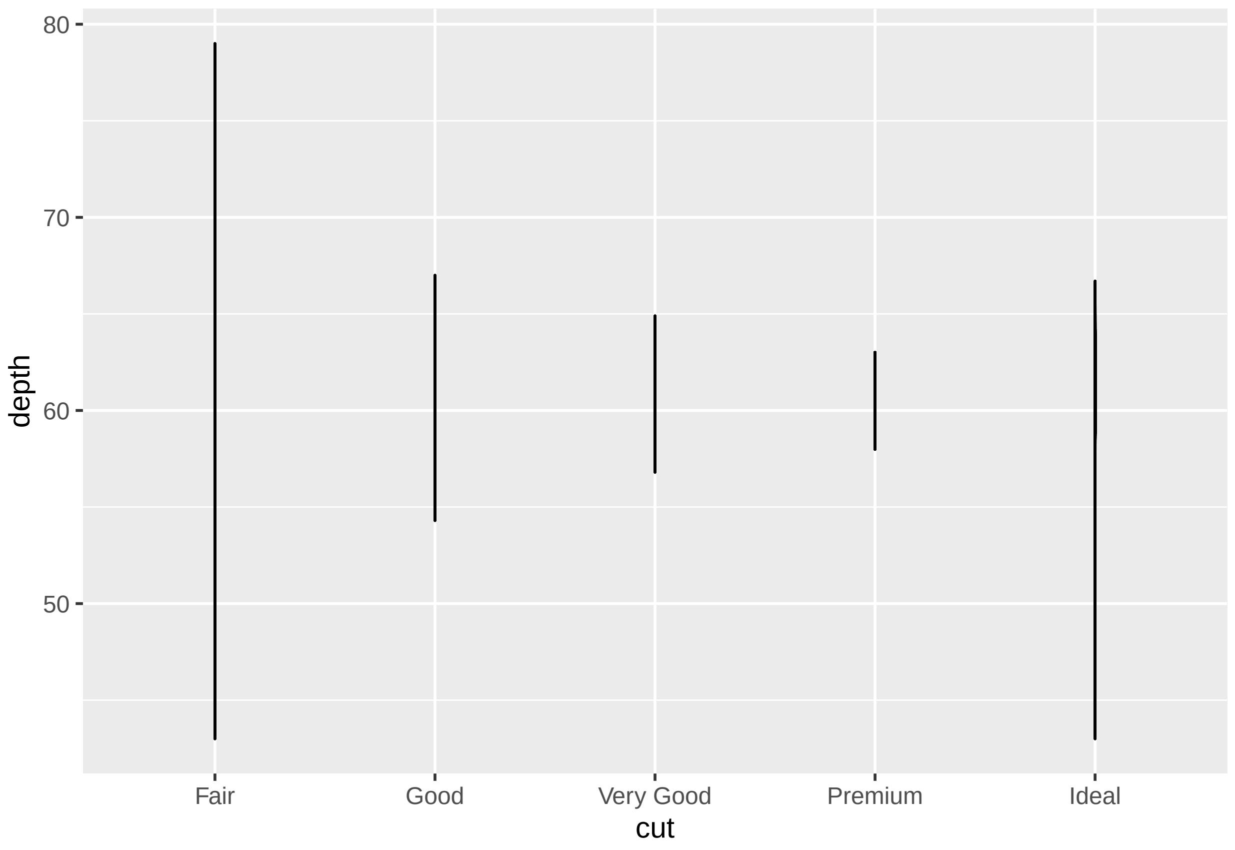


Exercise

1.What is the default geom associated with `stat_summary()`? How could you rewrite the previous plot to use that geom function instead of the stat function?

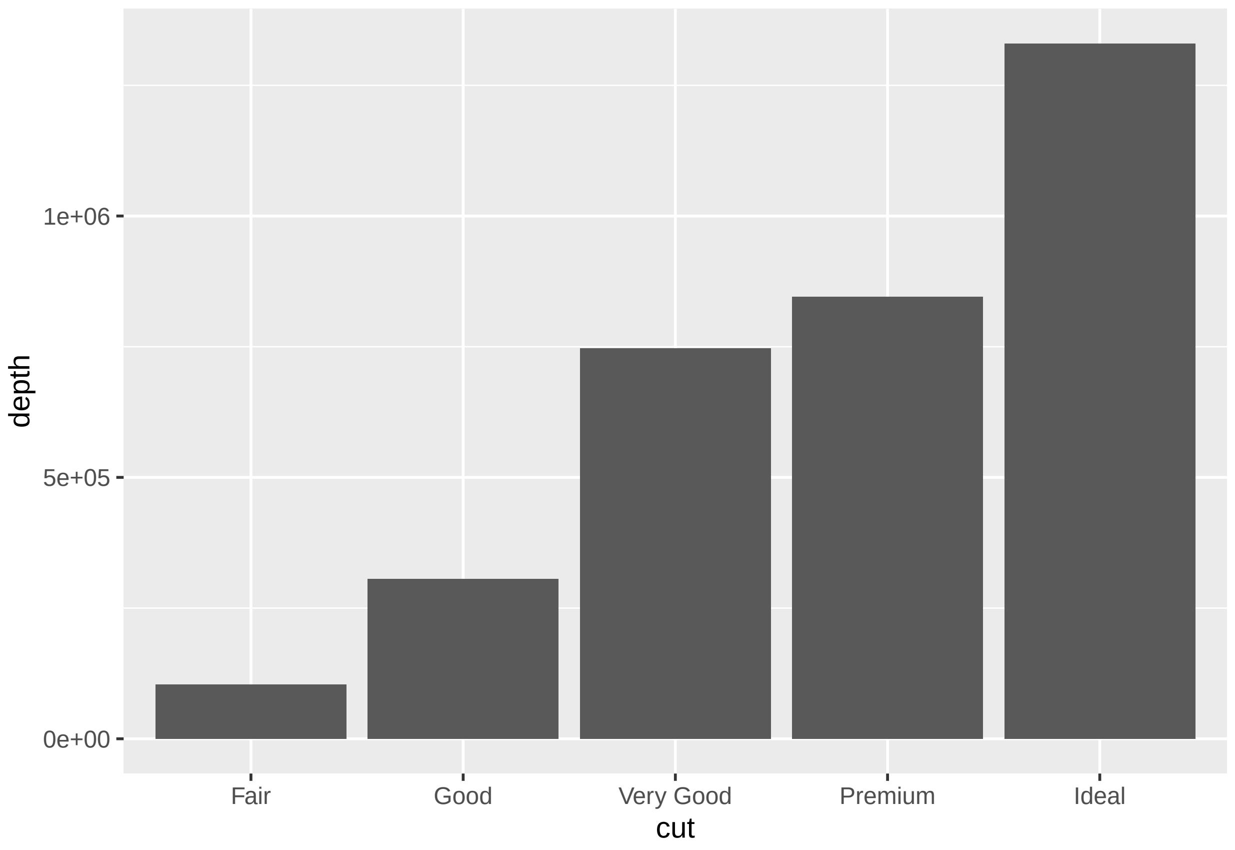
Answer

```
ggplot(data=diamonds)+
  geom_line(
    mapping = aes(x=cut,y=depth)
  )
```

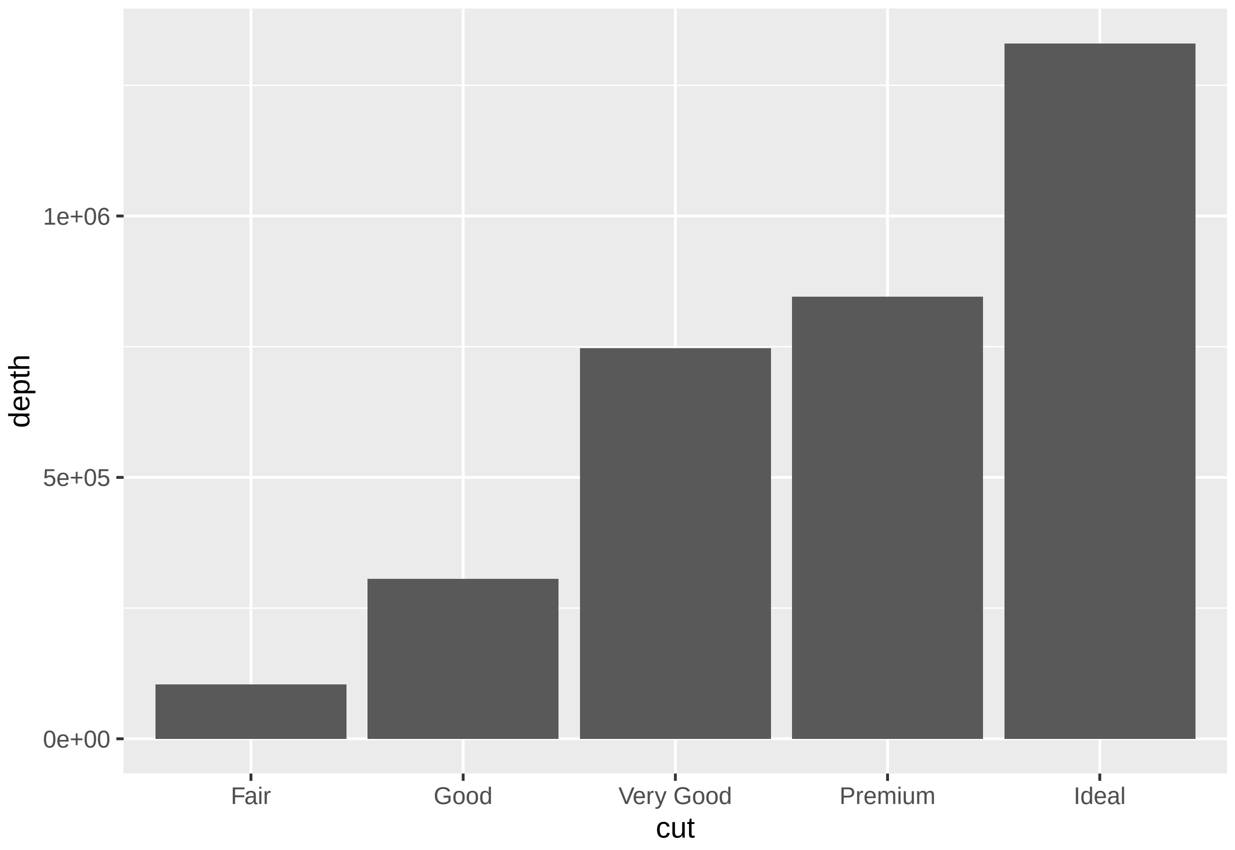


2. What does `geom_col()` do? How is it different to `geom_bar()`?

```
ggplot(data=diamonds)+
  geom_col(
    mapping = aes(x=cut,y=depth)
  )
```



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



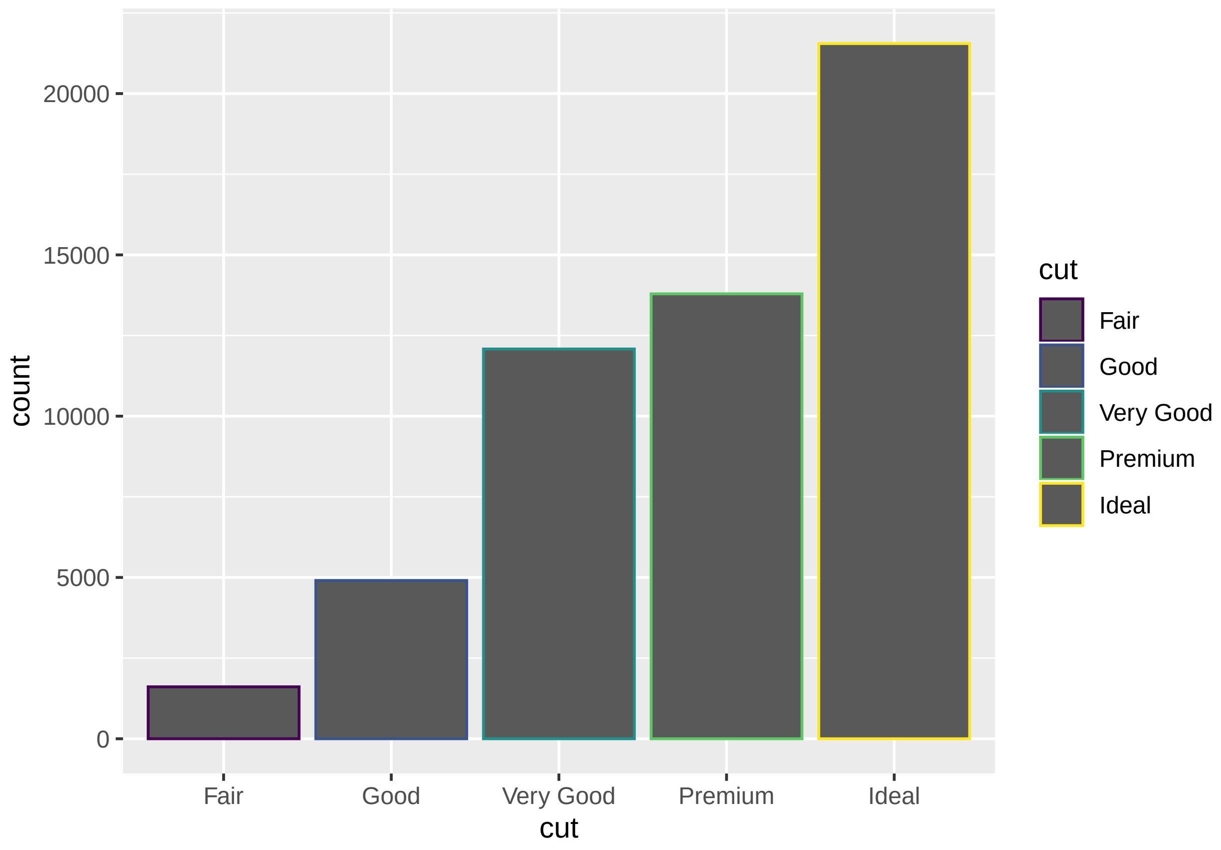
Answer

The geom_col directly use stat to transform our data, but geom_bar need stat="identity

Position adjustsment

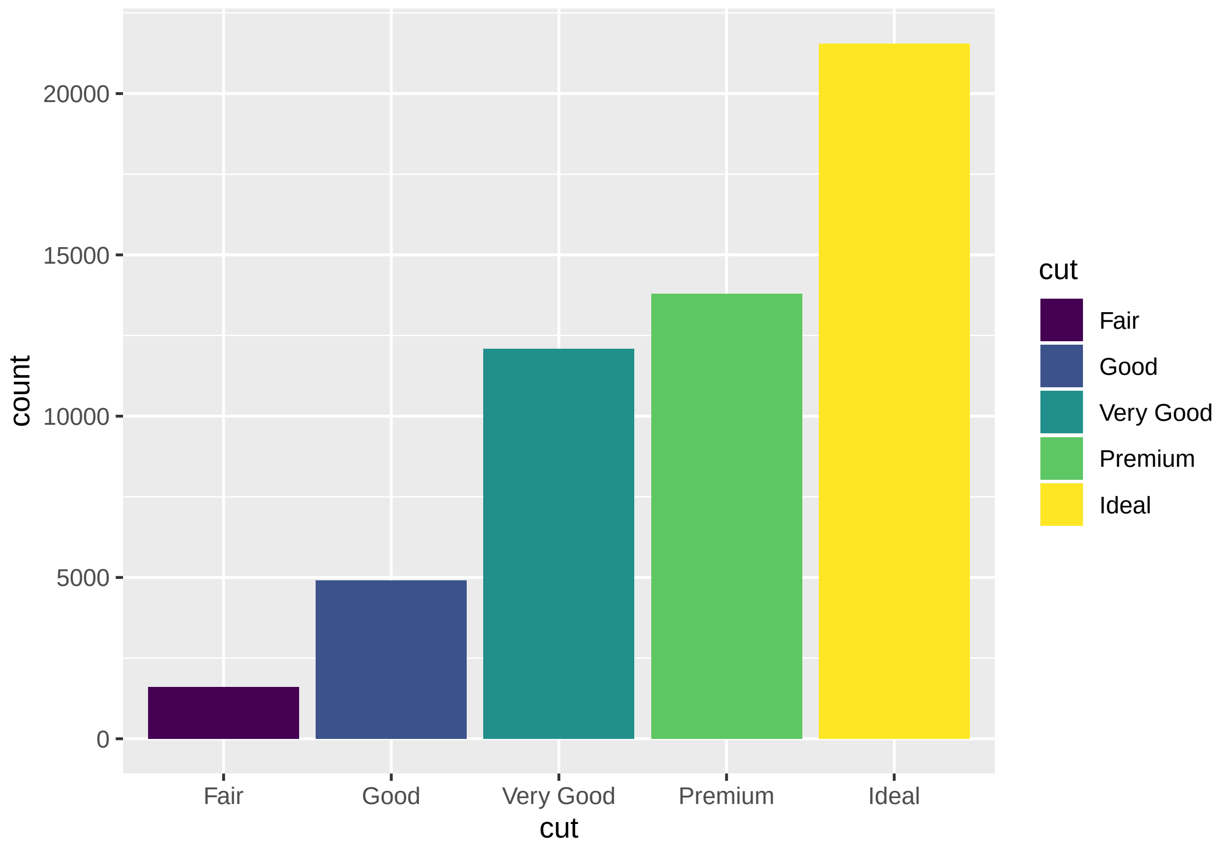
We can colour a bar chart using either the `colour` aesthetic, or, more usefully, `fill`:

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



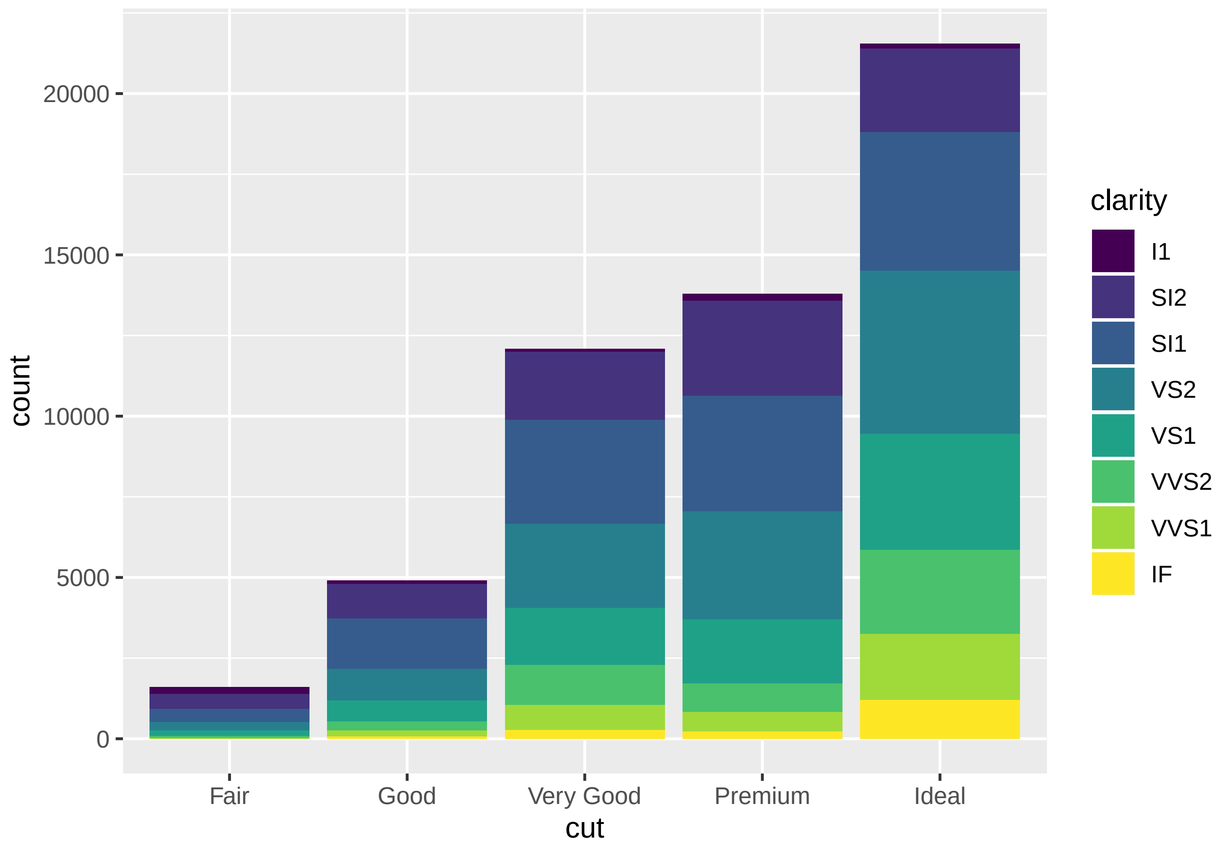
```
# Or alternatively,

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



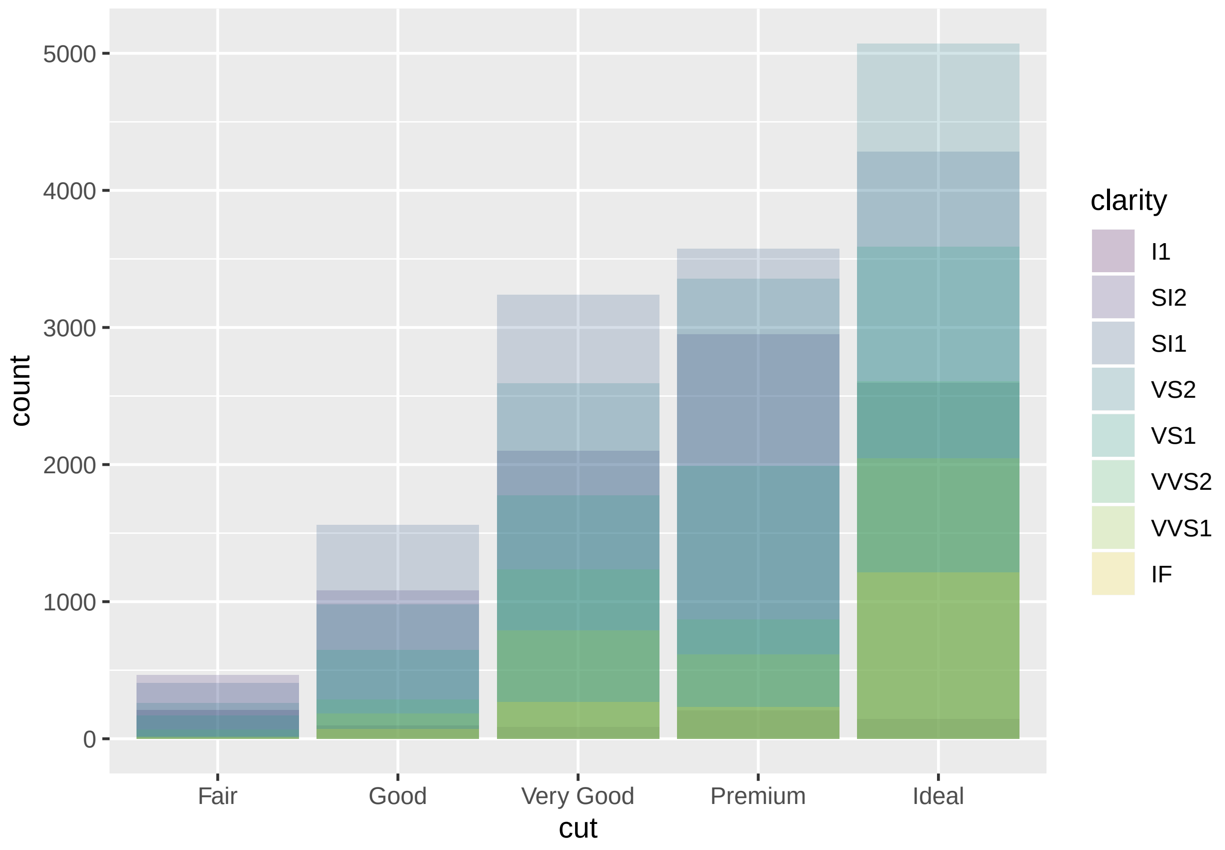
You can also map the fill aesthetic to another variable, like `clarity`: the bars are automatically stacked.

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

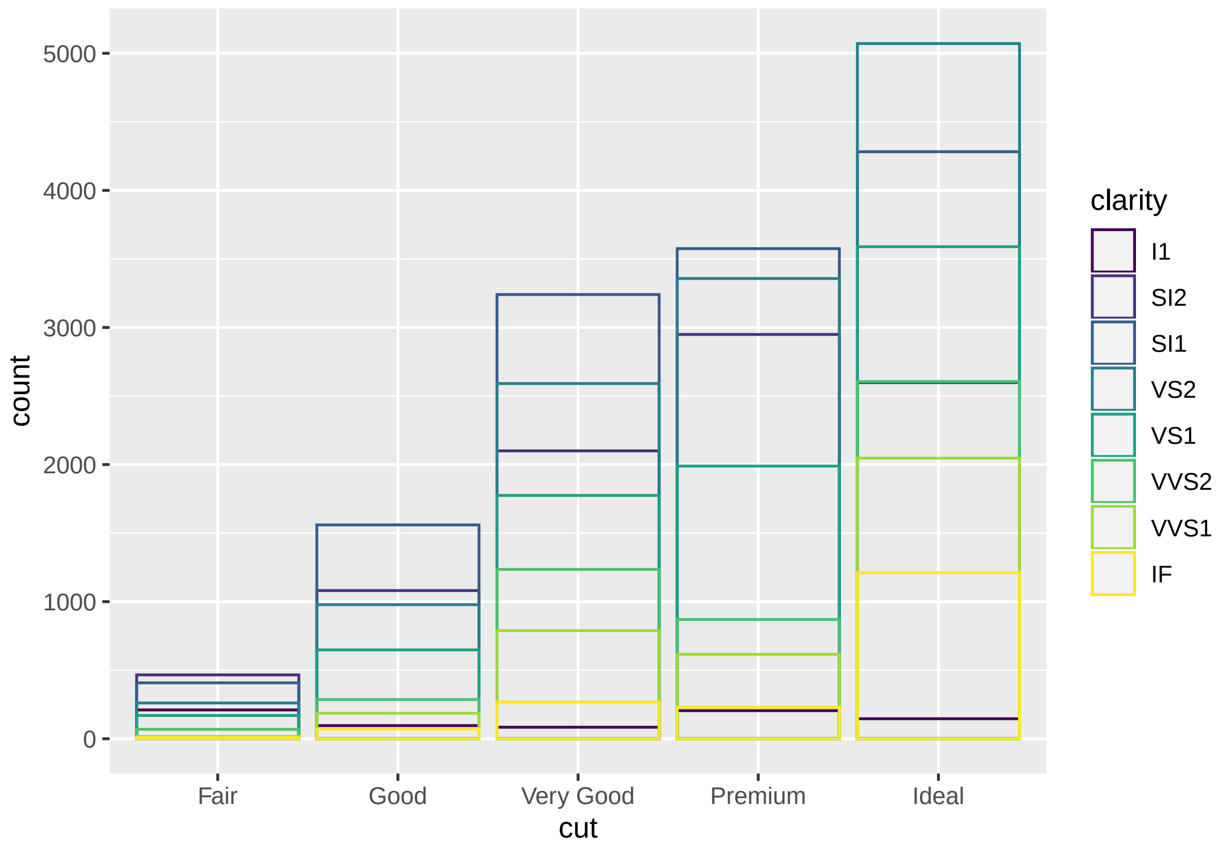


1. `position="identity"` will place each object exactly where it falls in the context of the graph

```
ggplot(data=diamonds,mapping = aes(x=cut,fill=clarity))+
  geom_bar(alpha=1/5,position="identity")
```

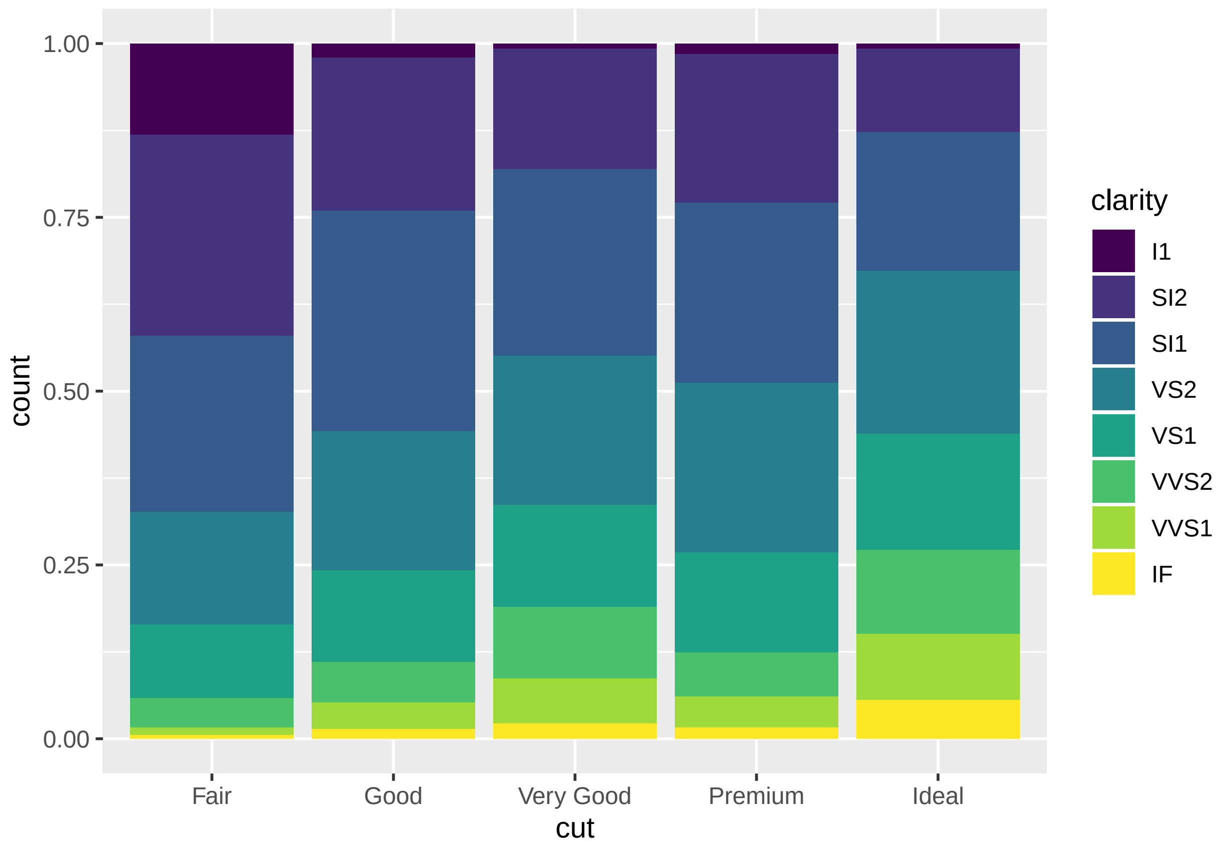


```r
ggplot(data=diamonds,mapping=aes(x=cut,color=clarity))+
  geom_bar(fill=NA,position="identity")
```



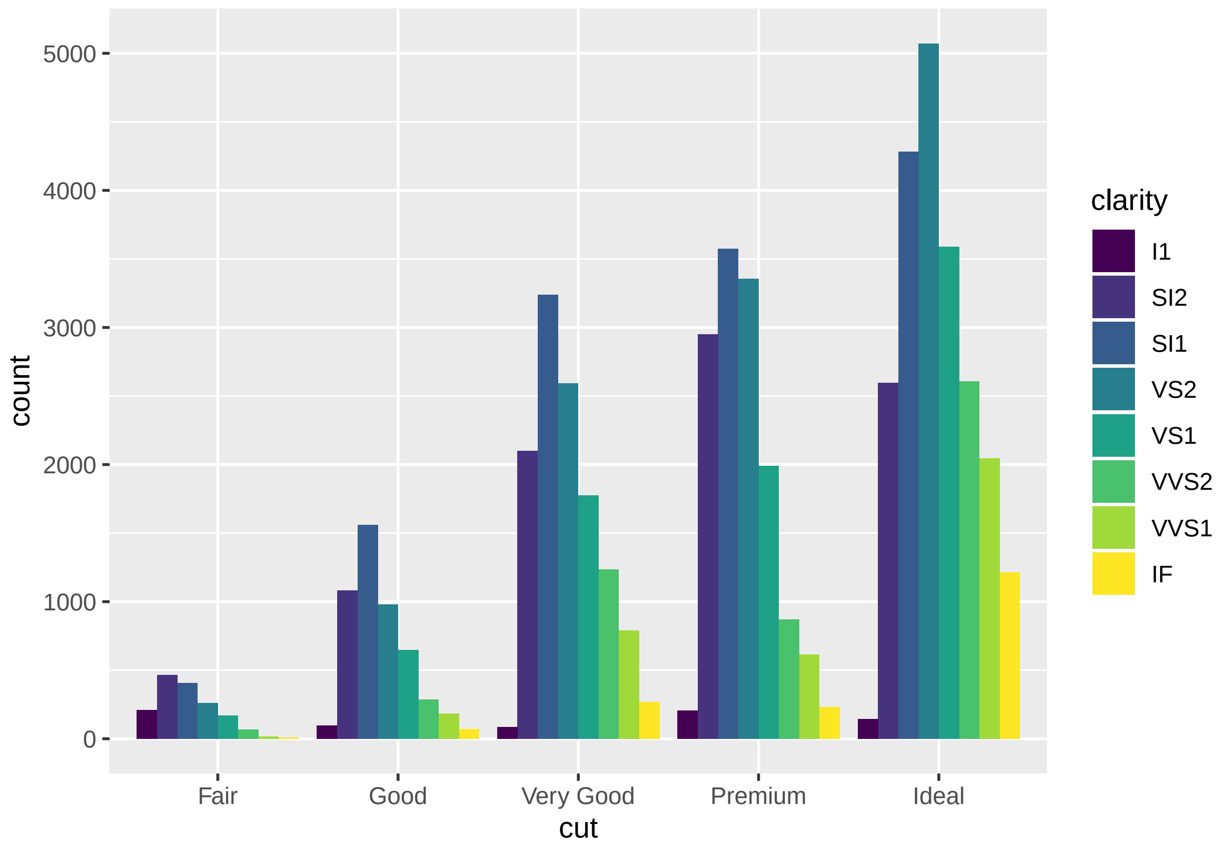
2. `position="fill"` works like stacking, but makes each set of stacked bars the same height. This makes it easier to compare proportions across groups.

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



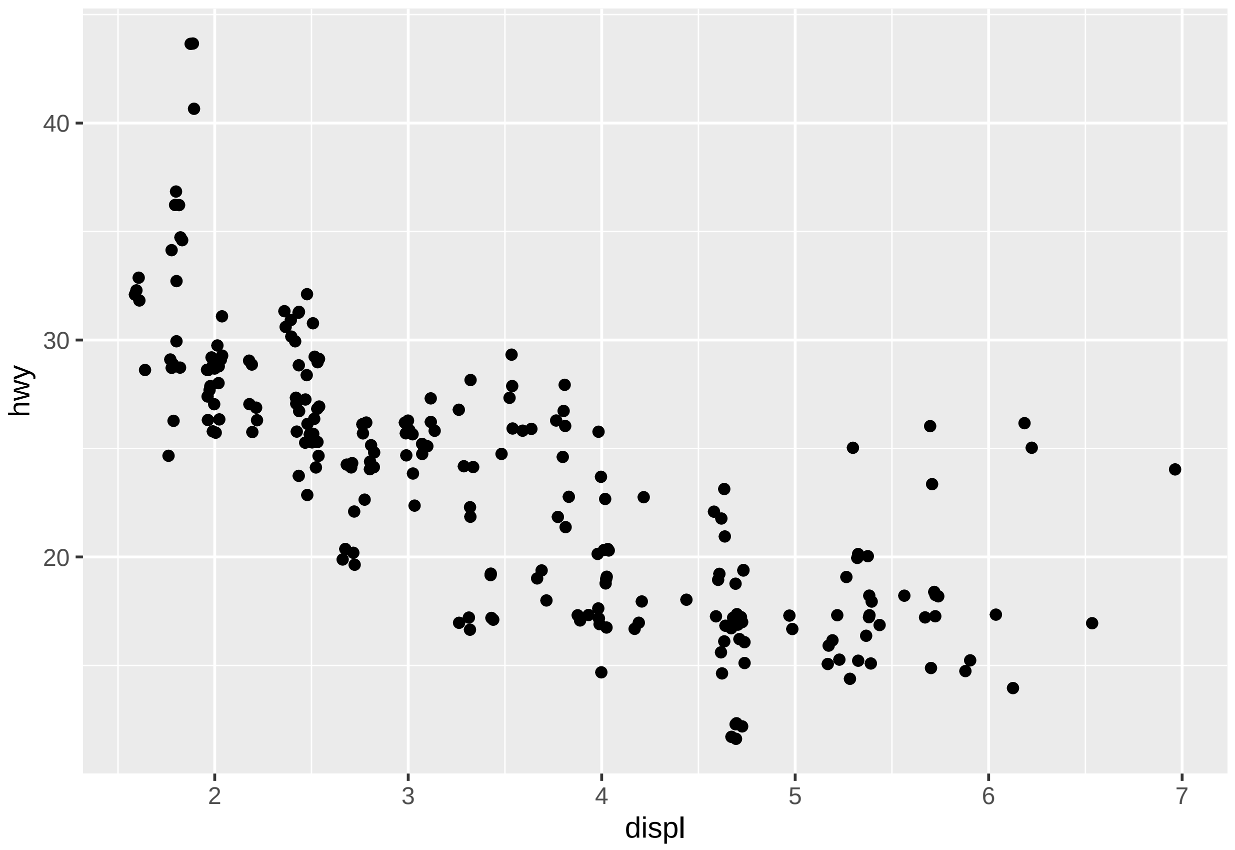
3. `position="dodge"` places overlapping objects directly beside one another. This makes it easier to compare individual values

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



4. `position="jitter"` adds a small amount of random noise to each point. This spreads the points out because no two points are likely to receive the same amount of random noise.

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



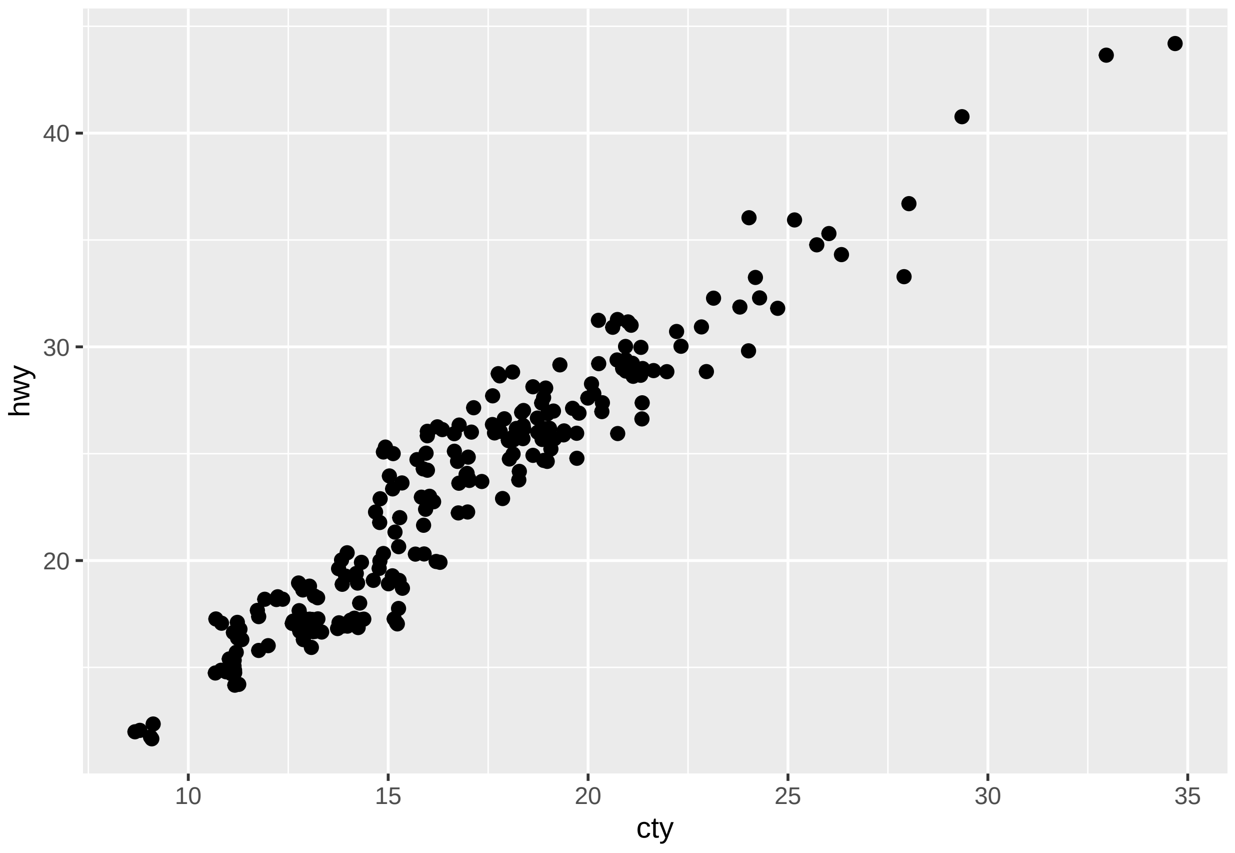
We can use `geom_point(position="jitter")` and `geom_jitter()` `?position_fill()` can find what is the structure of the function

Exercises

1. What is the problem with this plot? How could you improve it?

Answer

```
ggplot(data=mpg,mapping=aes(x=cty,y=hwy))+
  geom_point(position="jitter",size=2)
```



The data is seemly perfect, we should add random noise to each data point, to improve randomness.

2. What parameters to `geom_jitter()` control the amount of jittering?

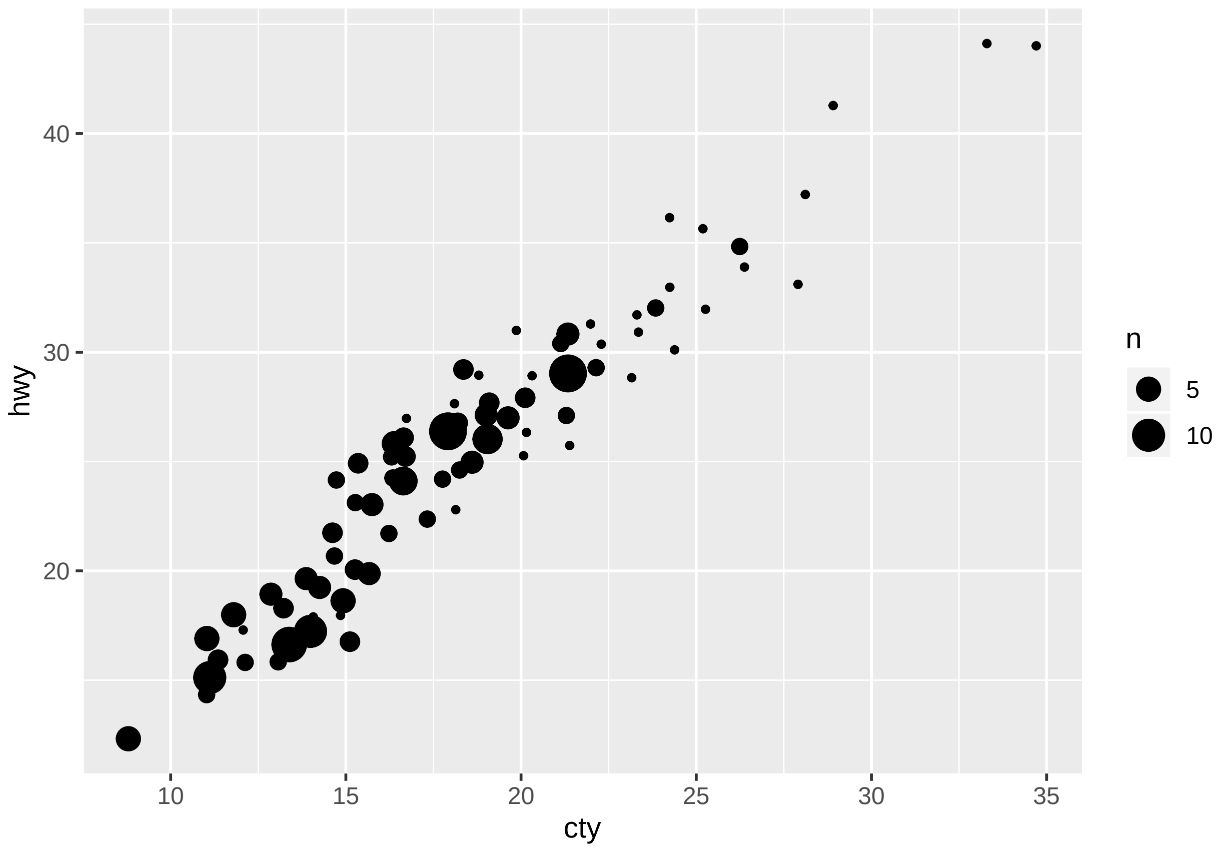
Answer

?geom_jitter

3. Compare and contrast `geom_jitter()` with `geom_count()`.

Answer

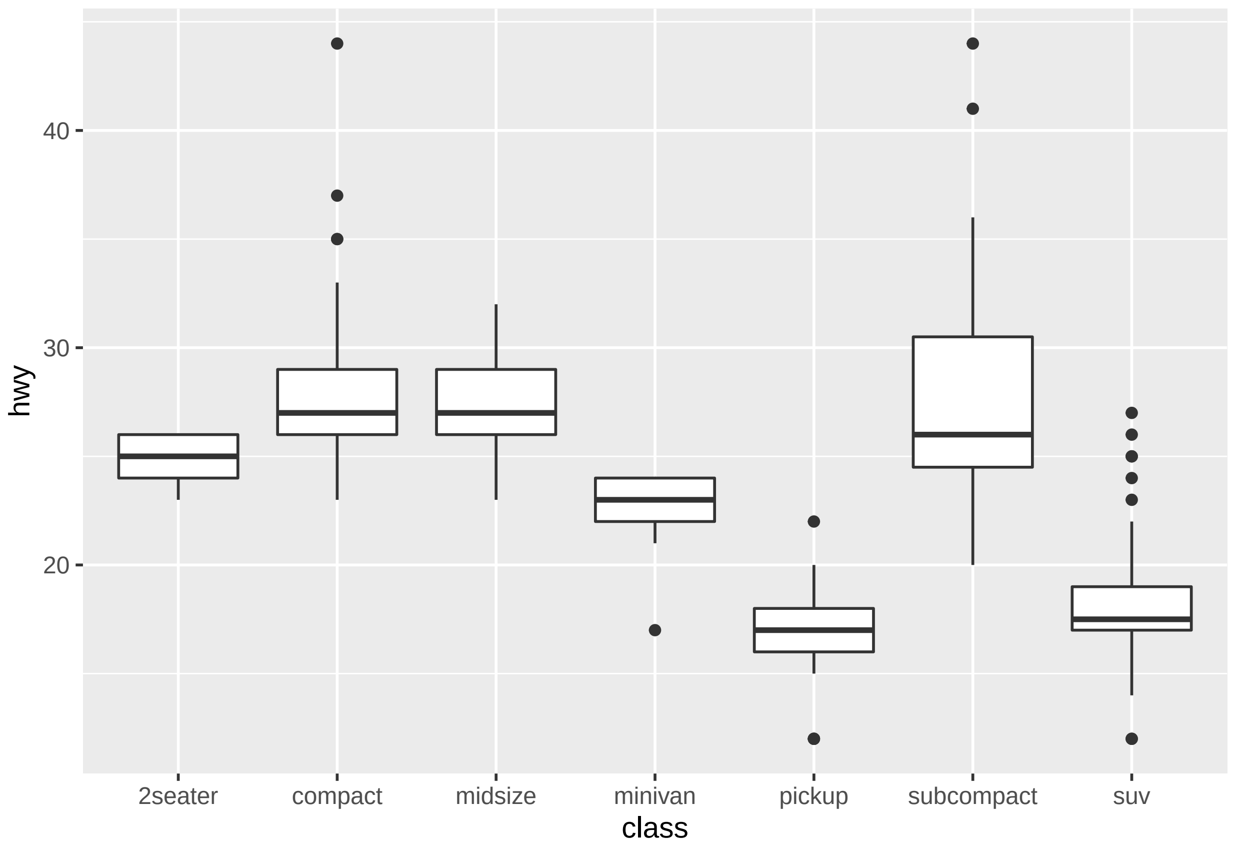
```
ggplot(data=mpg,mapping=aes(x=cty,y=hwy))+
  geom_count(position="jitter")
```



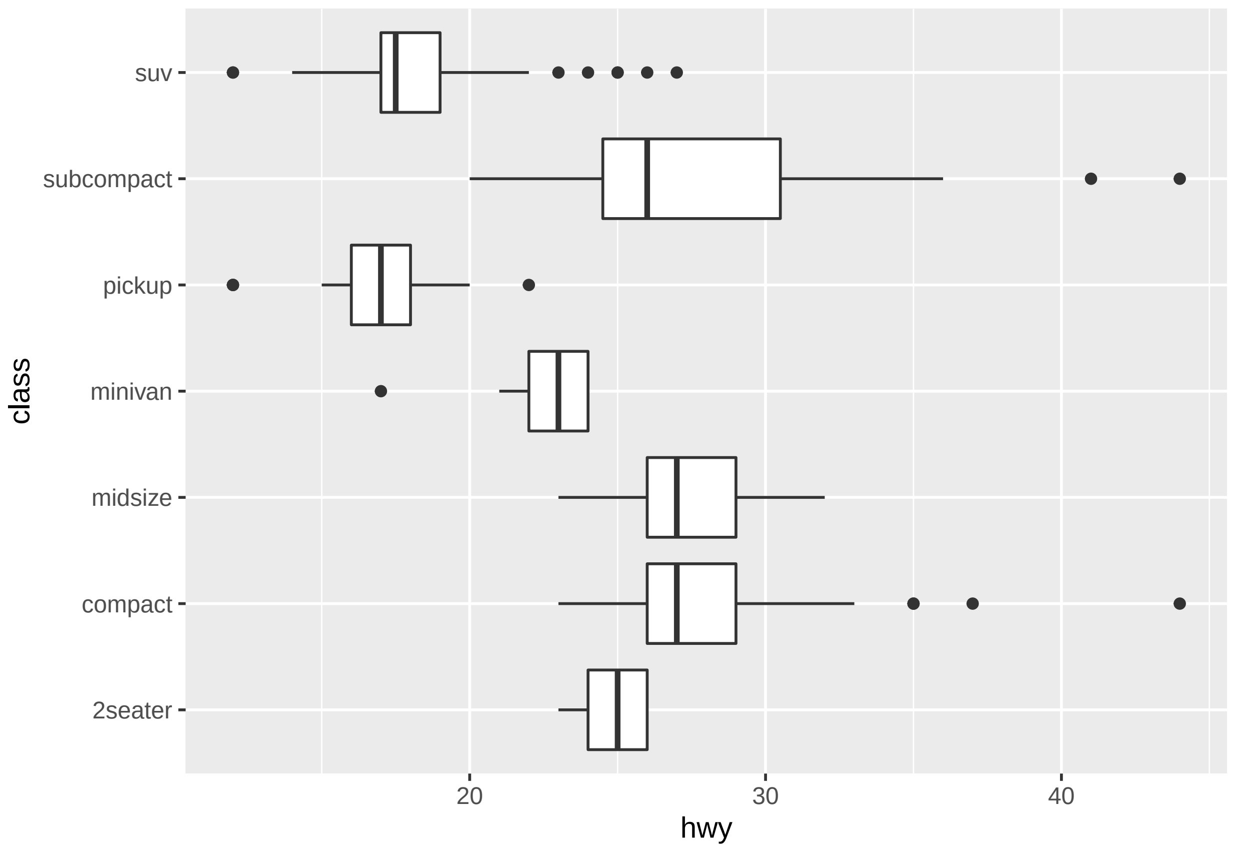
Coordinate systems

`coord_flip()` switches the x and y

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

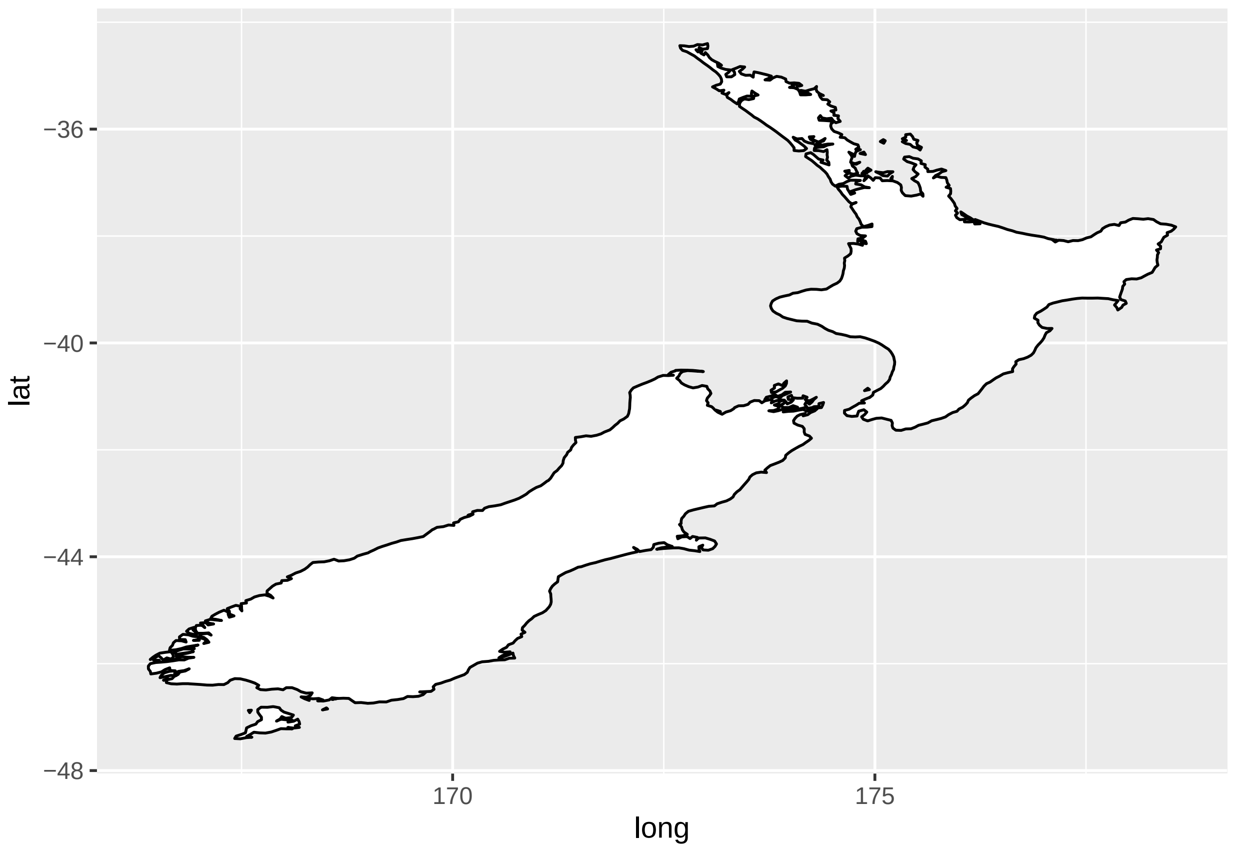


```
ggplot(data=mpg,mapping = aes(x=class,y=hwy))+
  geom_boxplot()+
  coord_flip()
```

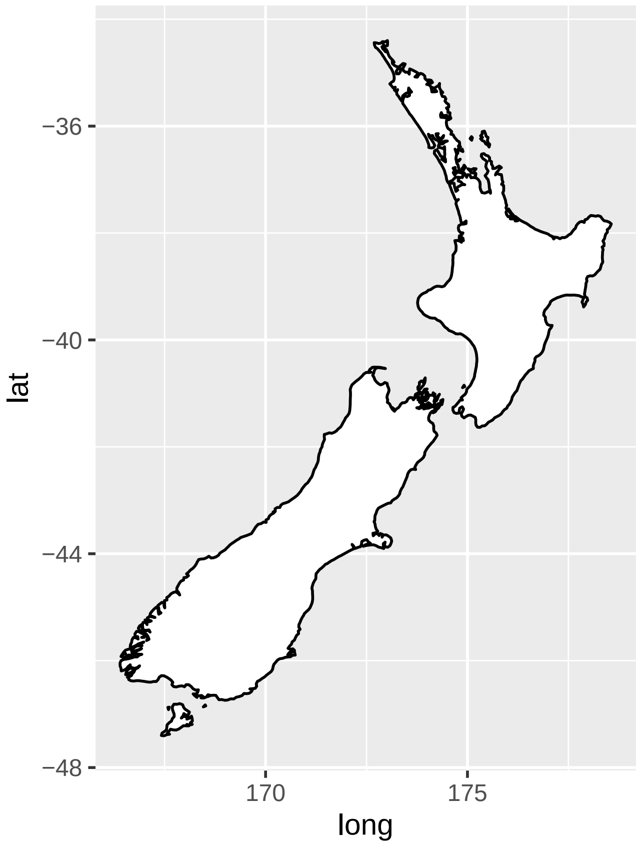


`coord_quickmap()` sets the aspect ratio correctly for maps. This is very important if you are plotting spatial data with ggplot2

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

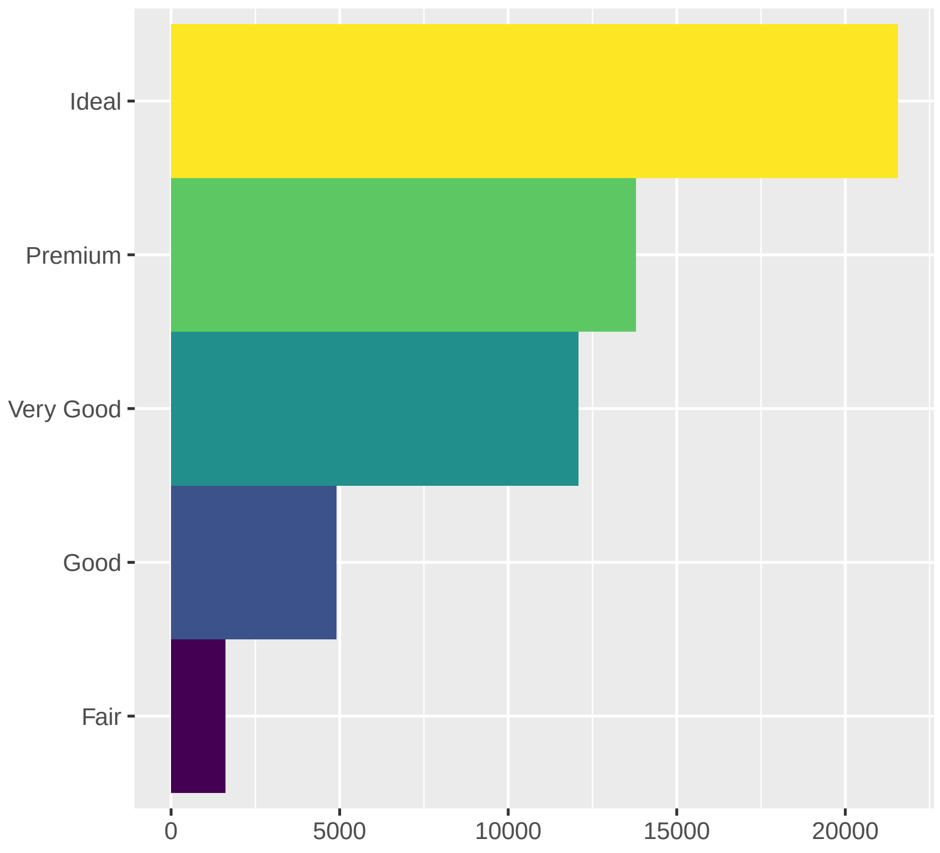


```r
ggplot(nz,aes(long,lat,group=group))+
  geom_polygon(fill="white",color="black")+
  coord_quickmap()
```

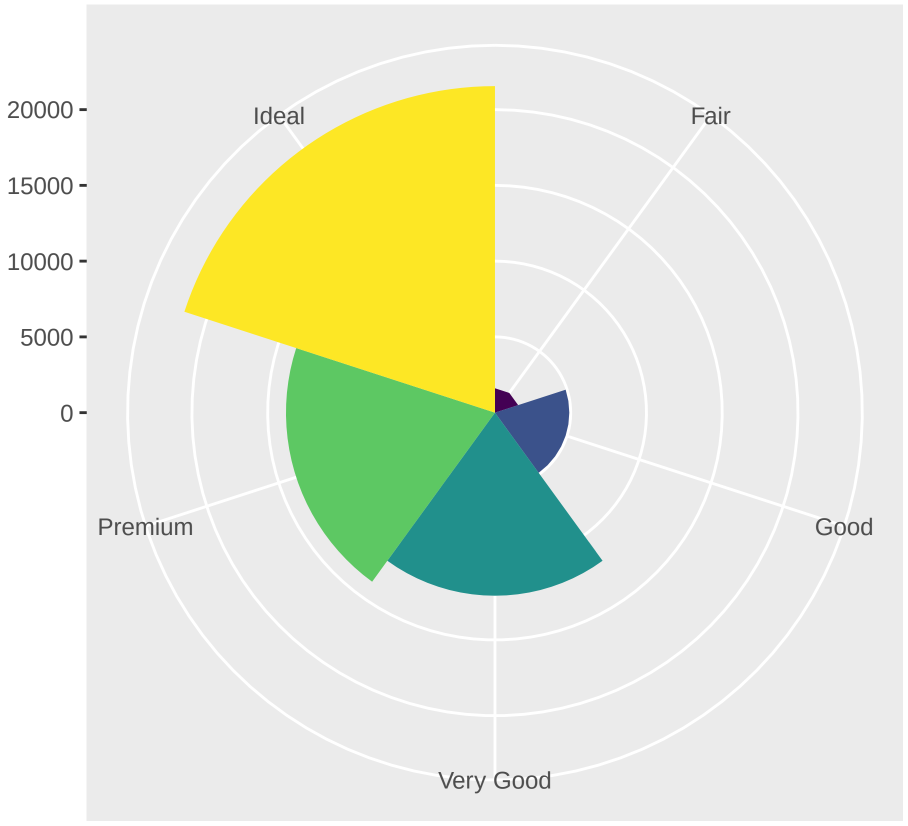


`coord_polar()` uses polar coordinates. Polar coordinates reveal an interestig connection between a bar chart and a Coxcomb chart

```
bar<-ggplot(data=diamonds)+
  geom_bar(
    mapping = aes(x=cut,fill=cut),
    show.legend = F,
    width=1
  )+
  theme(aspect.ratio = 1)+
  labs(x=NULL,y=NULL)
bar+coord_flip()
```



```
bar+coord_polar()
```

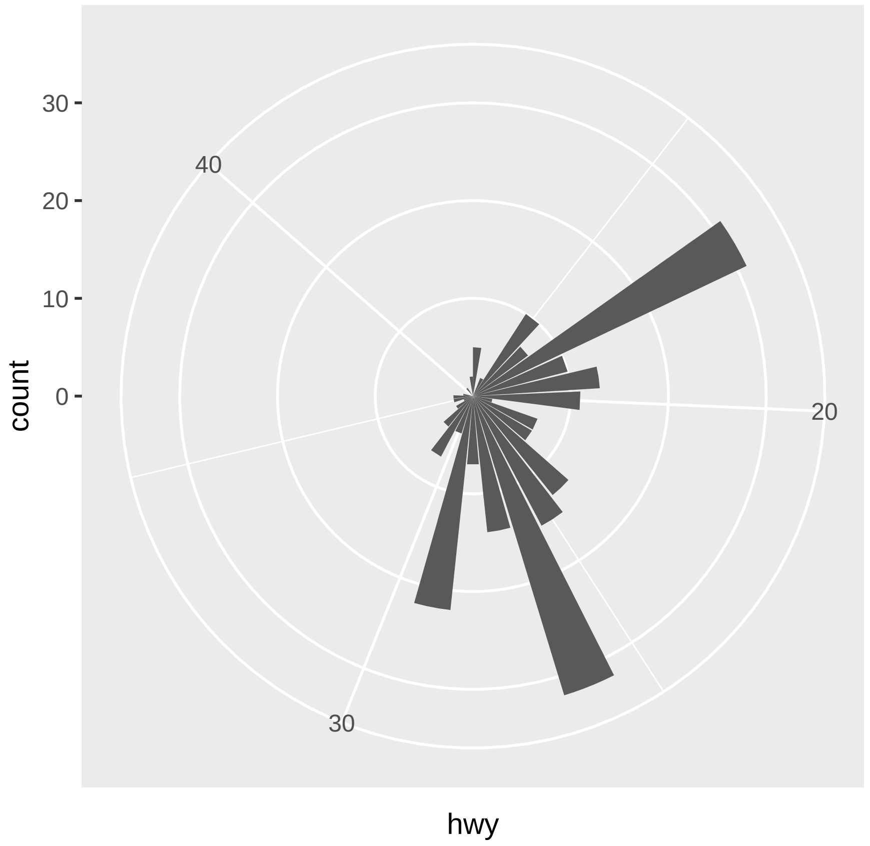


Exercises

1. Turn a stacked bar chart into a pie chart using 'coord_polar()'

Answer

```
bar<-ggplot(data=mpg,aes(x=hwy,fill=hwy))+
  geom_bar()
bar+coord_polar()
```



2. What does `labs()` do?

Answer

```
?labs
```

The layered grammar of graphics

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