3.Data Visualisation

HaiXiao Lu

3/5/2021

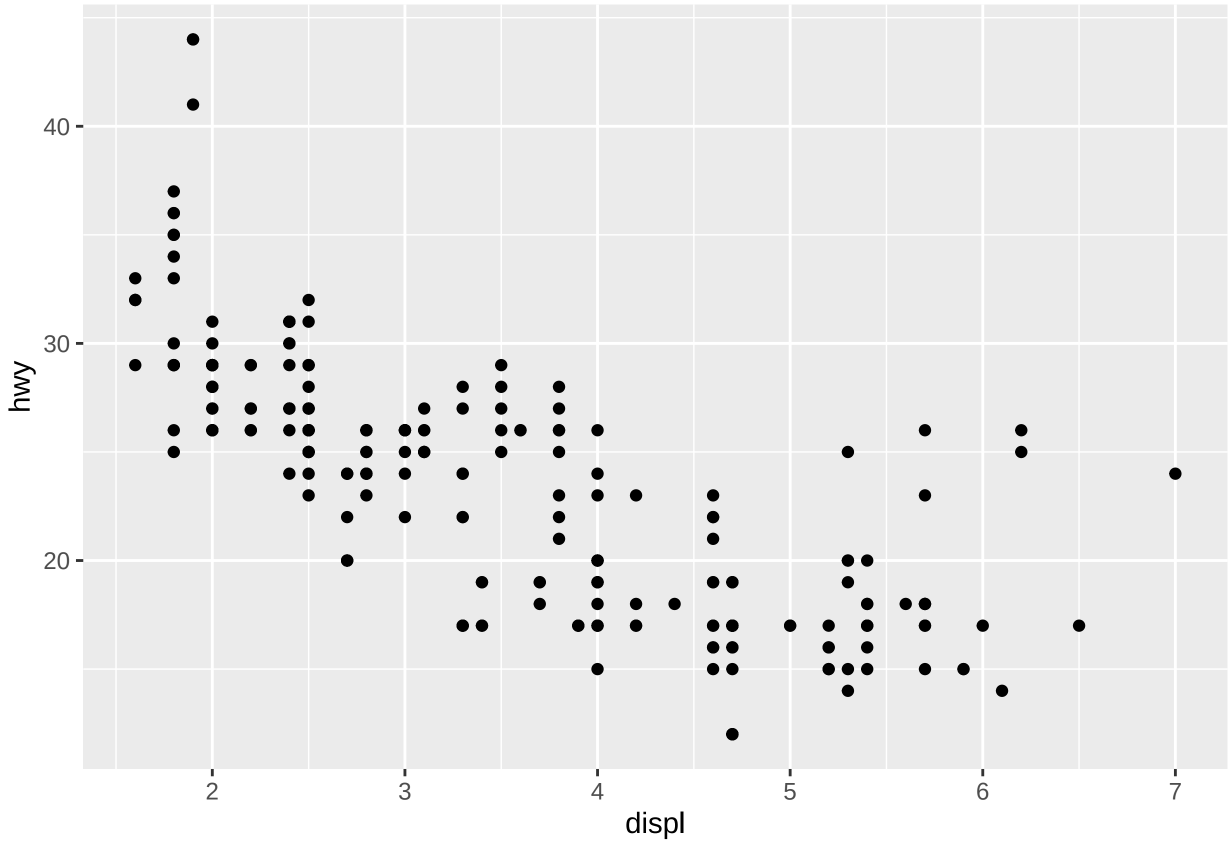
Contents

load data

```
## # 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
```

Creating a ggplot

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



Note: The plot shows a negative relationship between engine size(displ) and fuel efficiency (hwy). In other words, cars with big engines use more fuel.

With ggplot2, ggplot() creates a coordinate system that we can add layers to. We complete our graph by adding one or more layers to ggplot(). The function geom_point() adds a layer of points to our plot, which creates a scatterplot.

A graphing template

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

Exercises

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

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

```r
dim(mpg)
```

How many rows are in mpg? How many columns?

```
## [1] 234  11
```

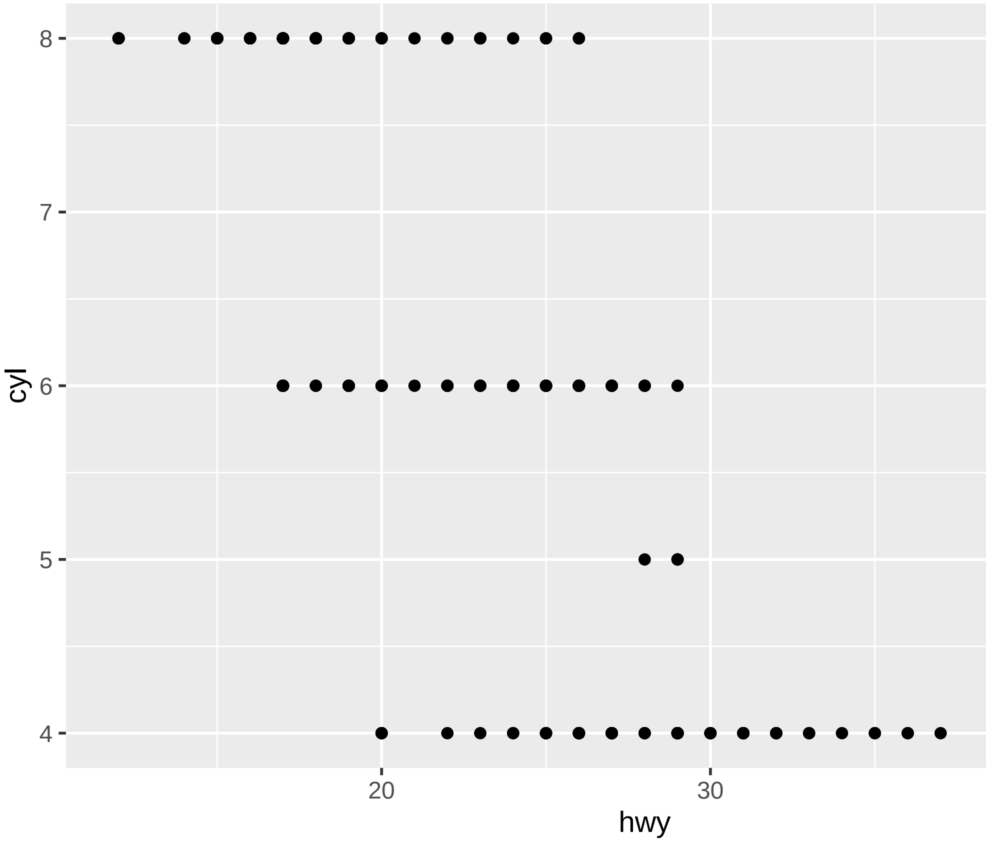
```r
glimpse(mpg)
```

```
## Rows: 234
## Columns: 11
## $ manufacturer <chr> "audi", "audi", "audi", "audi", "audi", "audi", "audi"...
## $ model        <chr> "a4", "a4", "a4", "a4", "a4", "a4", "a4", "a4 quattro"...
## $ displ        <dbl> 1.8, 1.8, 2.0, 2.0, 2.8, 2.8, 3.1, 1.8, 1.8, 2.0, 2.0,...
## $ year         <int> 1999, 1999, 2008, 2008, 1999, 1999, 2008, 1999, 1999, ...
## $ cyl          <int> 4, 4, 4, 4, 6, 6, 6, 4, 4, 4, 4, 6, 6, 6, 6, 6, 6, 8, ...
## $ trans        <chr> "auto(l5)", "manual(m5)", "manual(m6)", "auto(av)", "a...
## $ drv          <chr> "f", "f", "f", "f", "f", "f", "f", "4", "4", "4", "4",...
## $ cty          <int> 18, 21, 20, 21, 16, 18, 18, 18, 16, 20, 19, 15, 17, 17...
## $ hwy          <int> 29, 29, 31, 30, 26, 26, 27, 26, 25, 28, 27, 25, 25, 25...
## $ fl           <chr> "p", "p", "p", "p", "p", "p", "p", "p", "p", "p", "p",...
## $ class        <chr> "compact", "compact", "compact", "compact", "compact",...
```

```r
?mpg
```

What does the drv variable describe? drv: the type of drive train, where f = front-wheel drive, r = rear wheel drive, 4 = 4wd

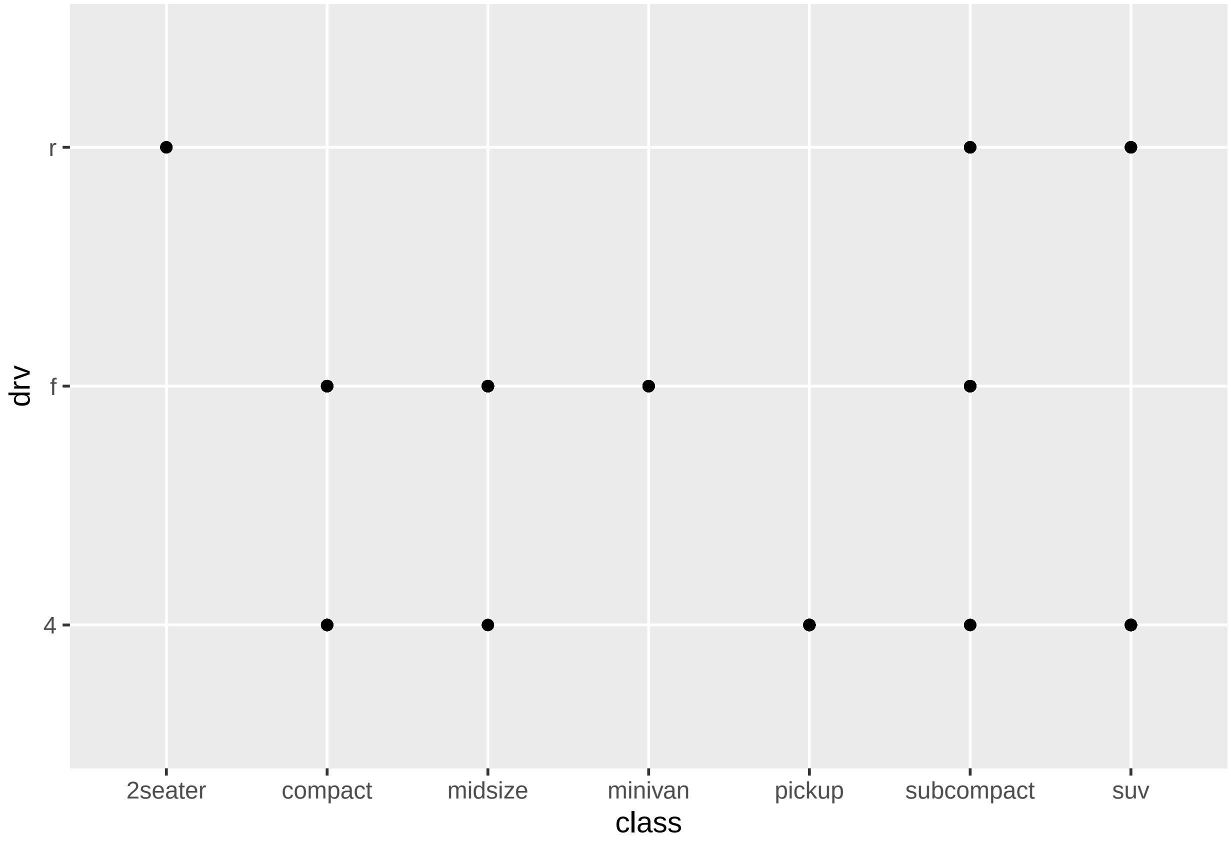
```r
ggplot(data = mpg) +
    geom_point(mapping = aes(x = hwy, y = cyl))
```



Make a scatterplot of hwy vs cyl

```
ggplot(data = mpg) +
    geom_point(mapping = aes(x = class, y = drv))
```

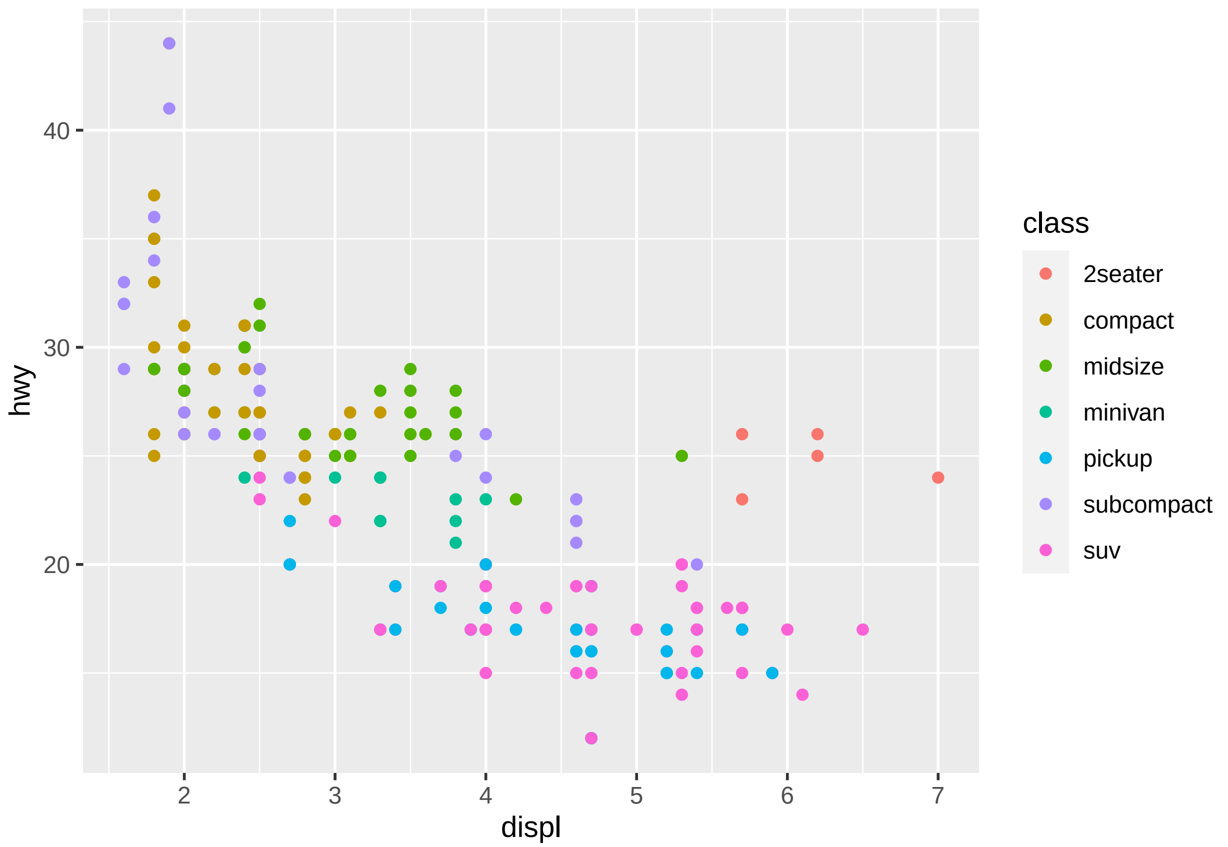
What happens if you make a scatterplot of `class` vs `drv`? why is the plot not useful?



Aesthetic mappings

You can convey information about your data by mapping the aesthetics in your plot to the variables in your dataset.

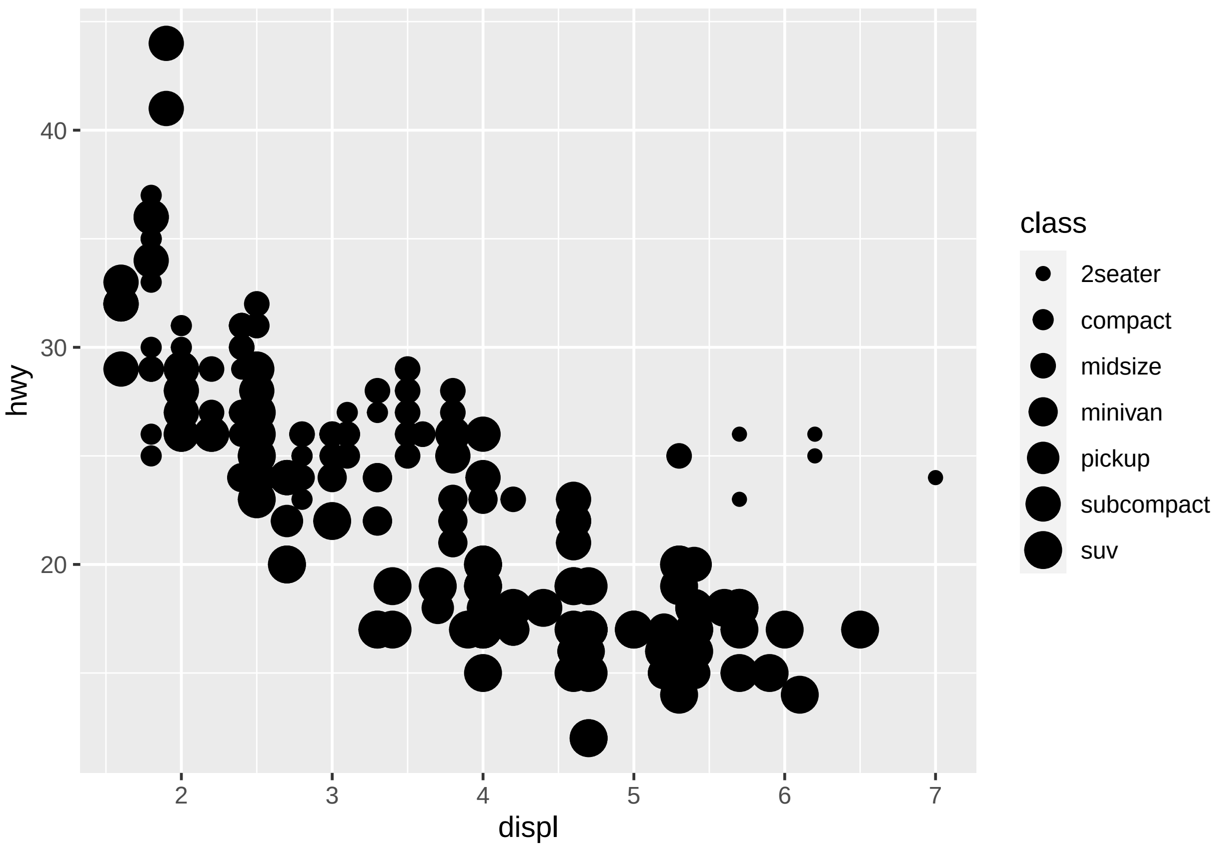
```
ggplot(data = mpg) +
    geom_point(mapping = aes(x = displ, y = hwy, color = class))
```



We could have mapped class to the size aesthetic in the same way.

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

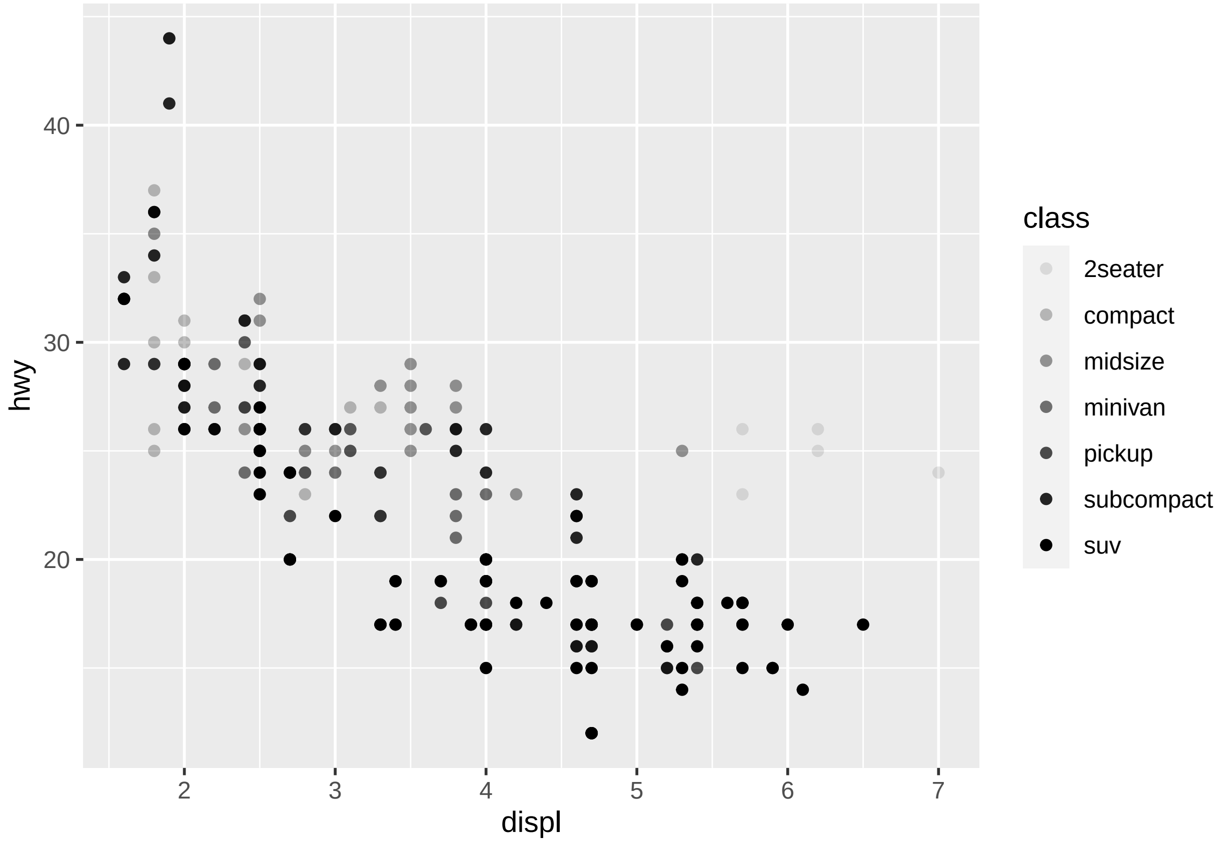
```
## Warning: Using size for a discrete variable is not advised.
```



Or we could have 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

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

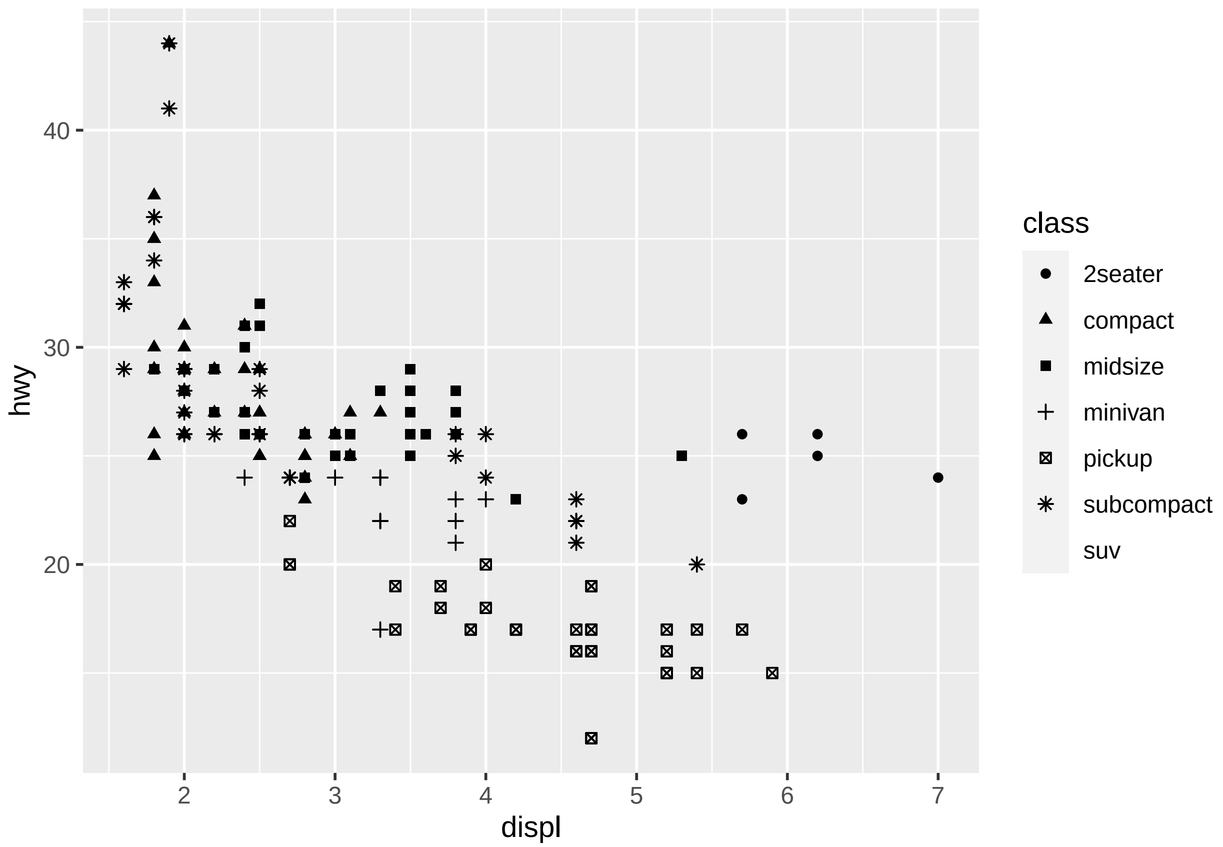
```
## Warning: Using alpha for a discrete variable is not advised.
```



```
# Right
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.
```

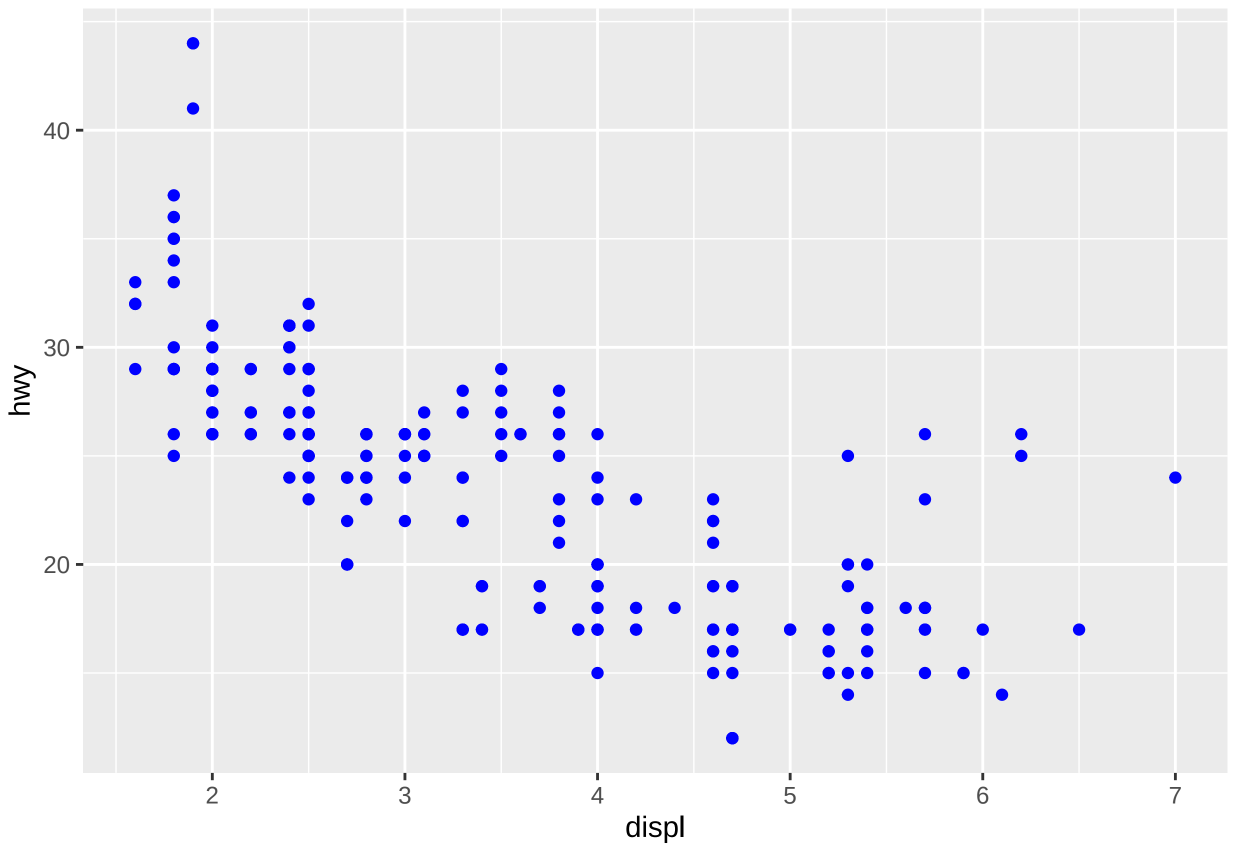
```
## Warning: Removed 62 rows containing missing values (geom_point).
```



The `aes()` function gathers together each of the aesthetic mappings used by a layer and passes them to the layer's mapping argument. The syntax highlights a useful insight about x and y

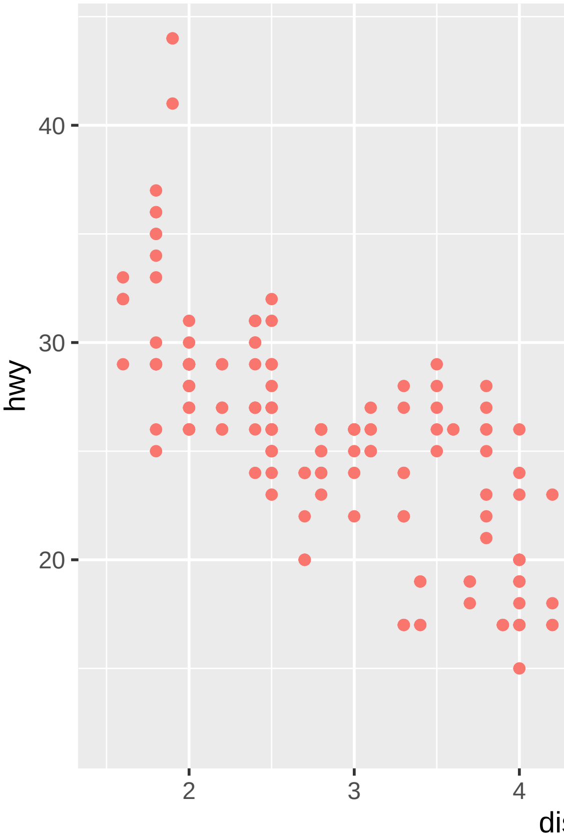
We can also set the aesthetic properties of our geom manualy. For example, we can make all of the points in our **plot blue**

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



Exercises

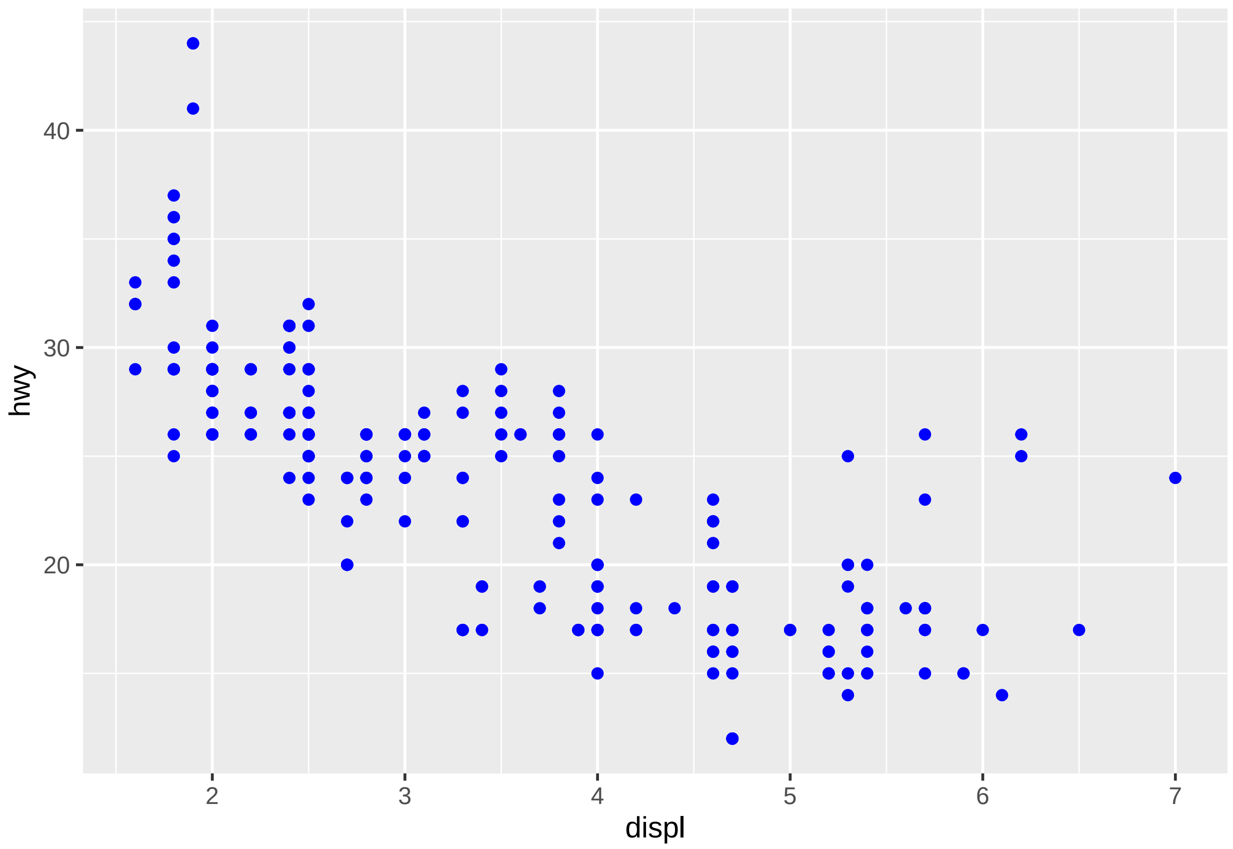
```
ggplot(data = mpg) +
    geom_point(mapping = aes(x = displ, y = hwy, color = 'blue'))
```



What's gone wrong with this code? why are the points not blue?

we can change it to:

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



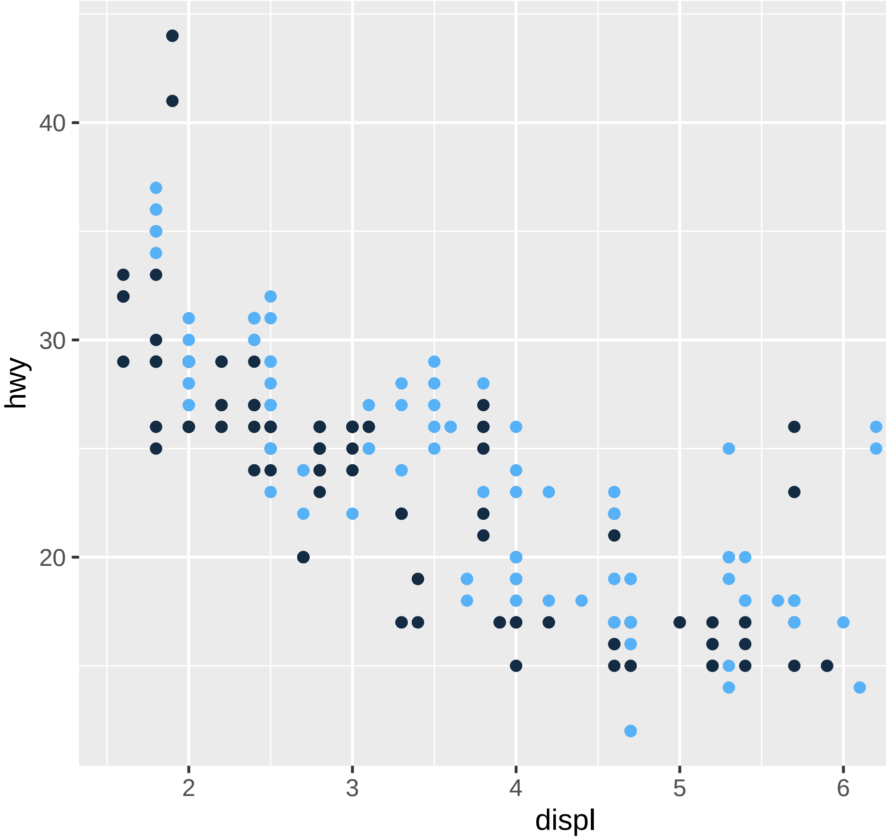
Which variables are continuous? How can you see this information when you run mpg?

```
glimpse(mpg)
```

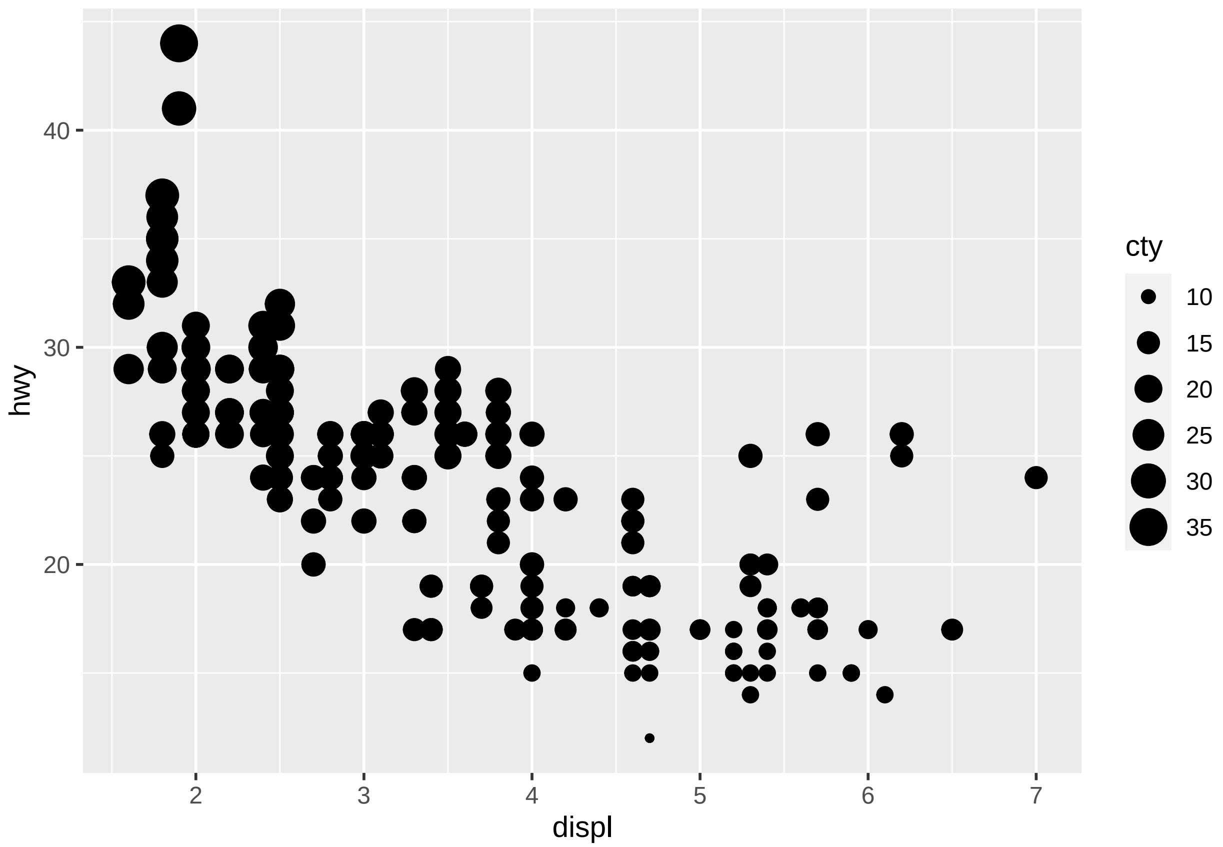
```
## Rows: 234
## Columns: 11
## $ manufacturer <chr> "audi", "audi", "audi", "audi", "audi", "audi", "audi"...
## $ model        <chr> "a4", "a4", "a4", "a4", "a4", "a4", "a4", "a4 quattro"...
## $ displ        <dbl> 1.8, 1.8, 2.0, 2.0, 2.8, 2.8, 3.1, 1.8, 1.8, 2.0, 2.0,...
## $ year         <int> 1999, 1999, 2008, 2008, 1999, 1999, 2008, 1999, 1999, ...
## $ cyl          <int> 4, 4, 4, 4, 6, 6, 6, 4, 4, 4, 4, 6, 6, 6, 6, 6, 6, 8, ...
## $ trans        <chr> "auto(l5)", "manual(m5)", "manual(m6)", "auto(av)", "a...
## $ drv          <chr> "f", "f", "f", "f", "f", "f", "f", "4", "4", "4", "4",...
## $ cty          <int> 18, 21, 20, 21, 16, 18, 18, 18, 16, 20, 19, 15, 17, 17...
## $ hwy          <int> 29, 29, 31, 30, 26, 26, 27, 26, 25, 28, 27, 25, 25, 25...
## $ fl           <chr> "p", "p", "p", "p", "p", "p", "p", "p", "p", "p", "p",...
## $ class        <chr> "compact", "compact", "compact", "compact", "compact",...
```

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

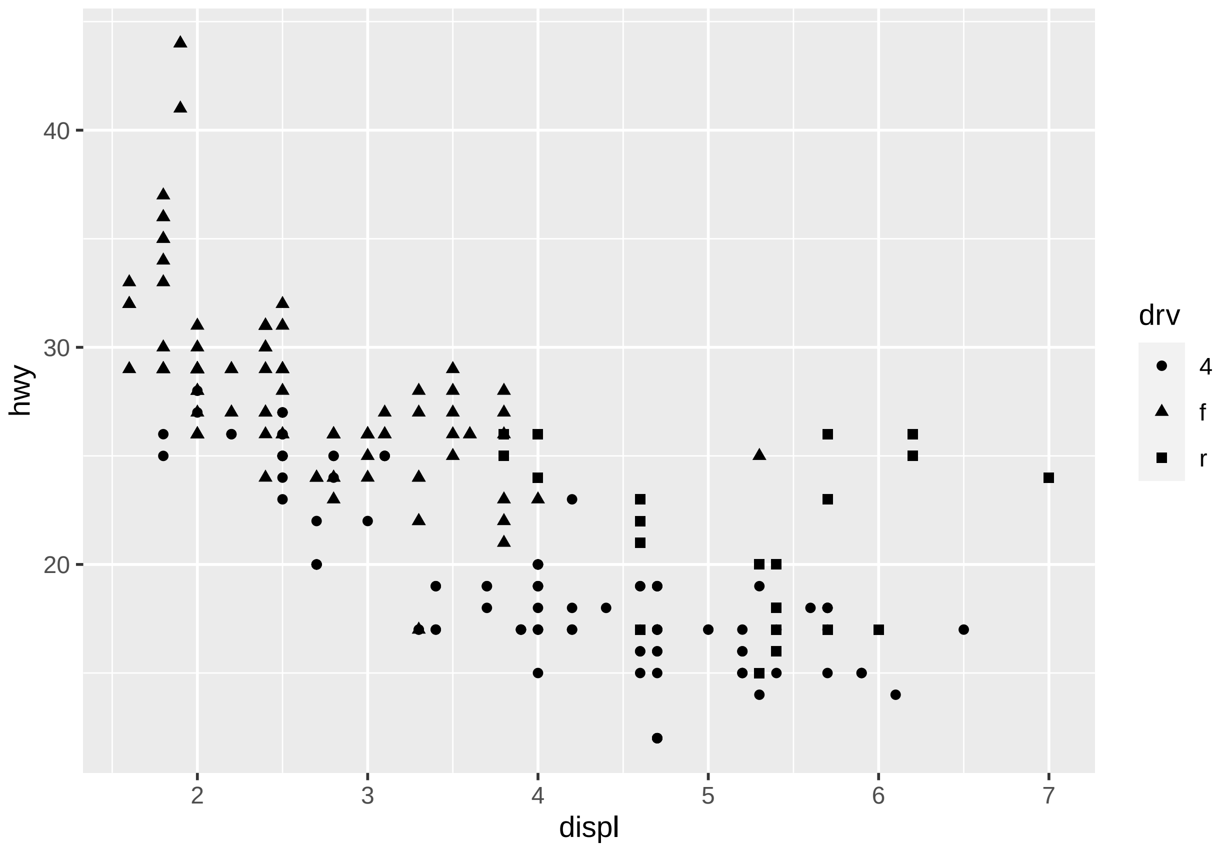
Map a continuous variable to color, size and shape, How do these aesthetics behave differently for categorical vs. continuous variables?



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



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

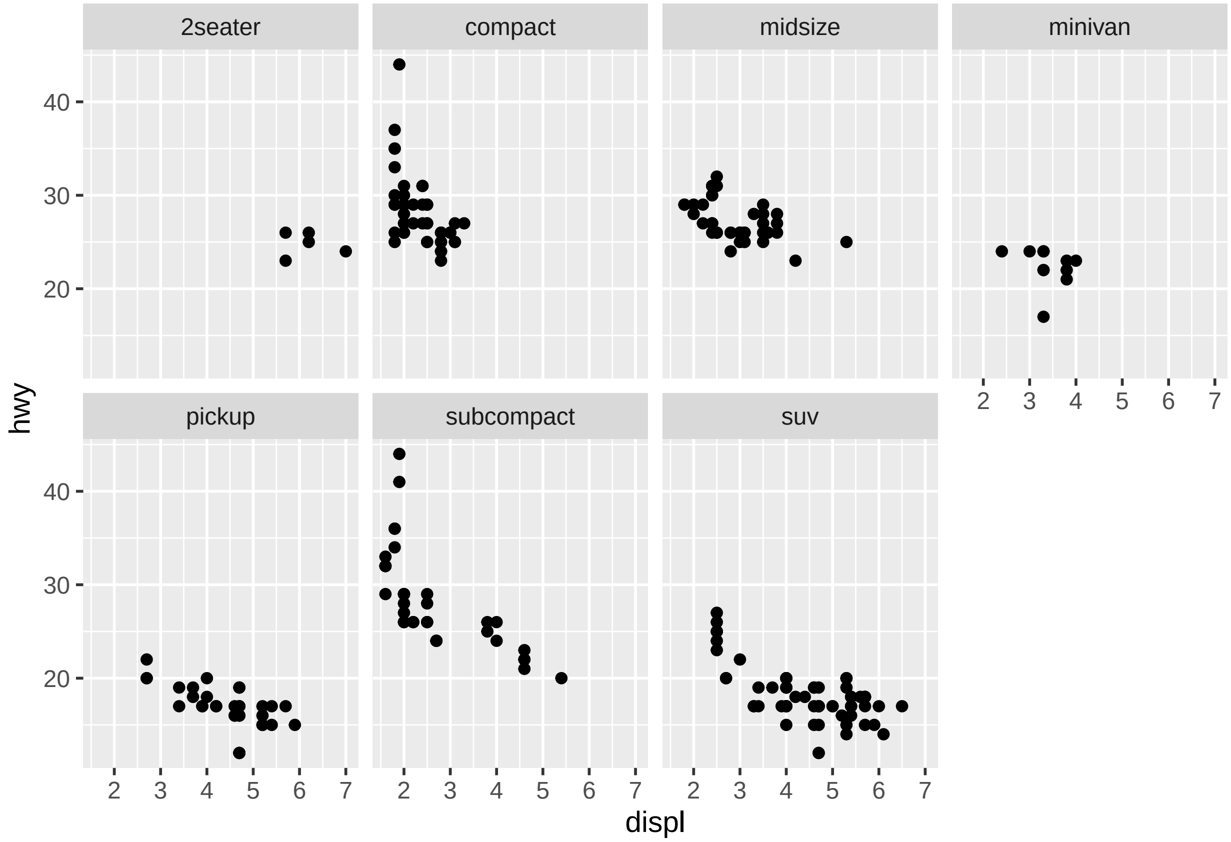


Facets

One way to add additional variables is with aesthetics. Another way, particularly useful for categorical variables, is to split your plot into **facets**, subplots that each display one subset of the data

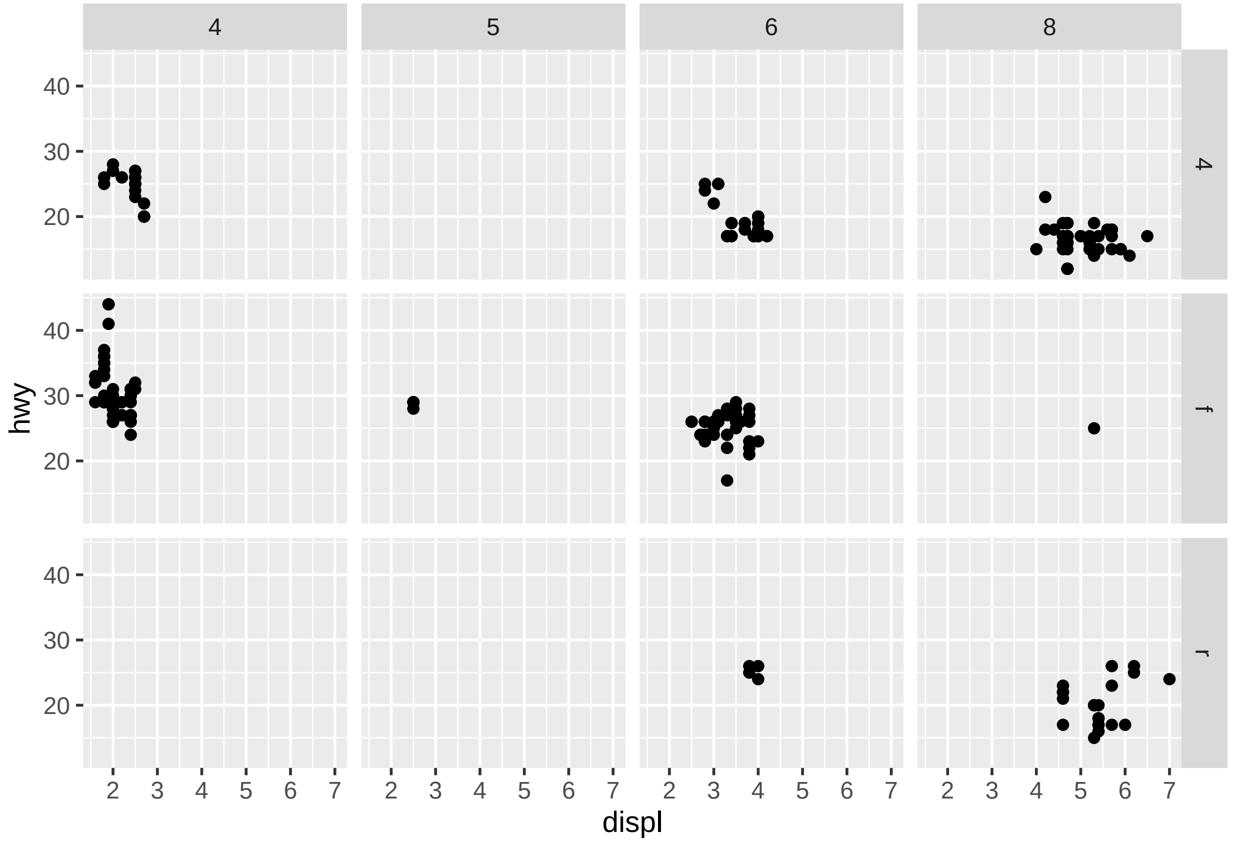
The facet your plot by a single variable, use `facet_wrap()`. The first argument of `facet_wrap()` should be a formular, which you create with ~ followed by a variable name. The variable that you pass to `facet_wrap()` should be discrete

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



To facet your plot on the combination of two varaibles, add `facet_grid()` to your plot call. The first argument of `facet_grid()` is also a formula. This time the formula should contain two variable names separated by a ~

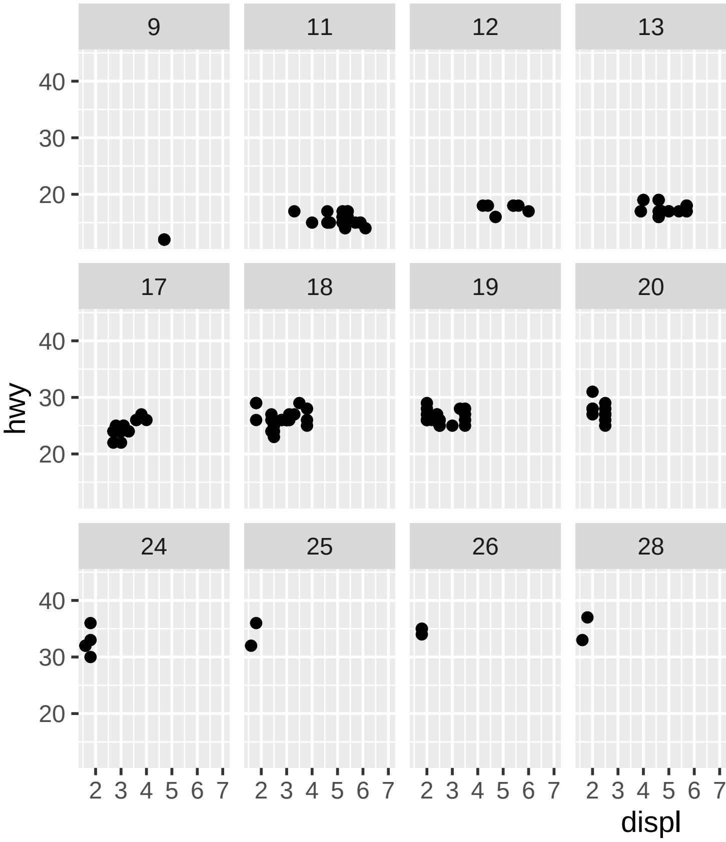
```
ggplot(data = mpg) +
    geom_point(mapping = aes(x = displ, y = hwy)) +
    facet_grid(drv ~ cyl)
```



If you prefer to not facet in the rows or columns dimension, use a. instead of a variable name, e.g. `+ facet_grid(. ~ cyl)`

Exercises

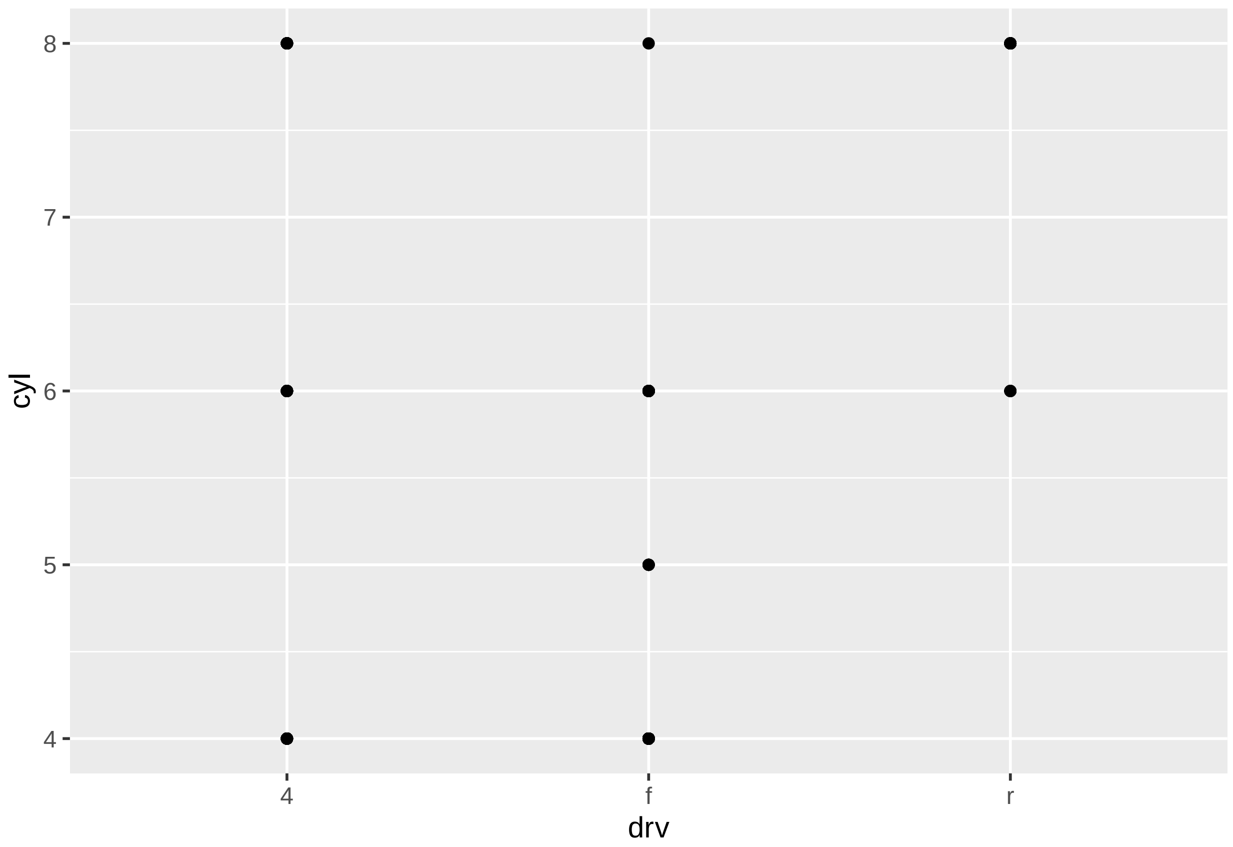
```
ggplot(data = mpg) +
    geom_point(mapping = aes(x = displ, y = hwy)) +
    facet_wrap(~cty, nrow = 3)
```



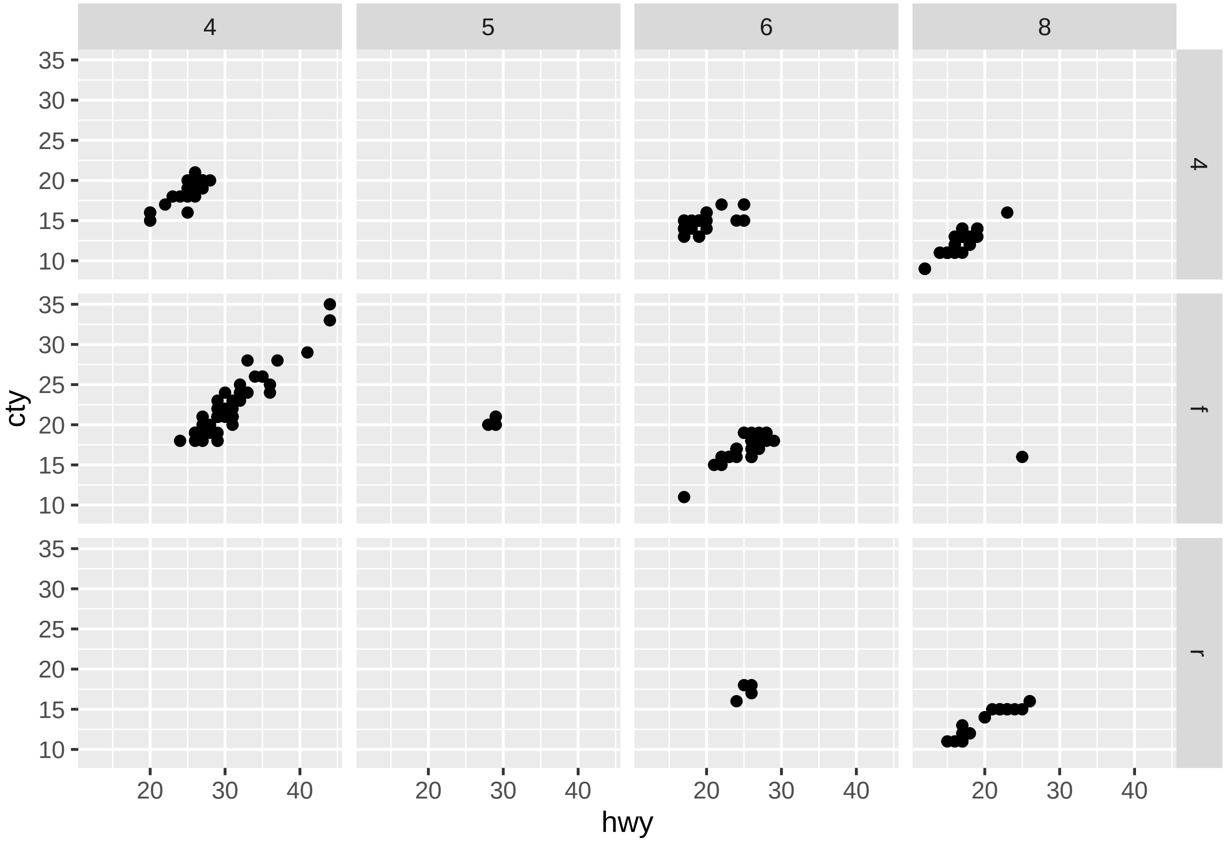
What happens if you facet on a continuous variable?
The continuous variable is converted to a categorical variable, and the plot contains a facet for each distinct value

What do the empty cells in plot with ‘facet_grid(drv ~ cyl) mean? How do they relate to this plot?

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

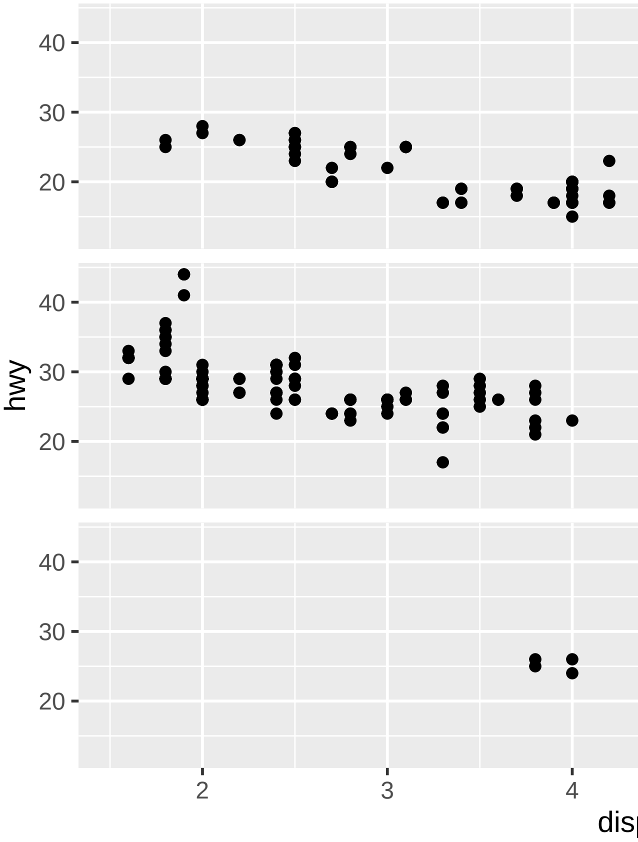


```
ggplot(data = mpg) +
    geom_point(mapping = aes(x = hwy, y = cty)) +
    facet_grid(drv ~ cyl)
```



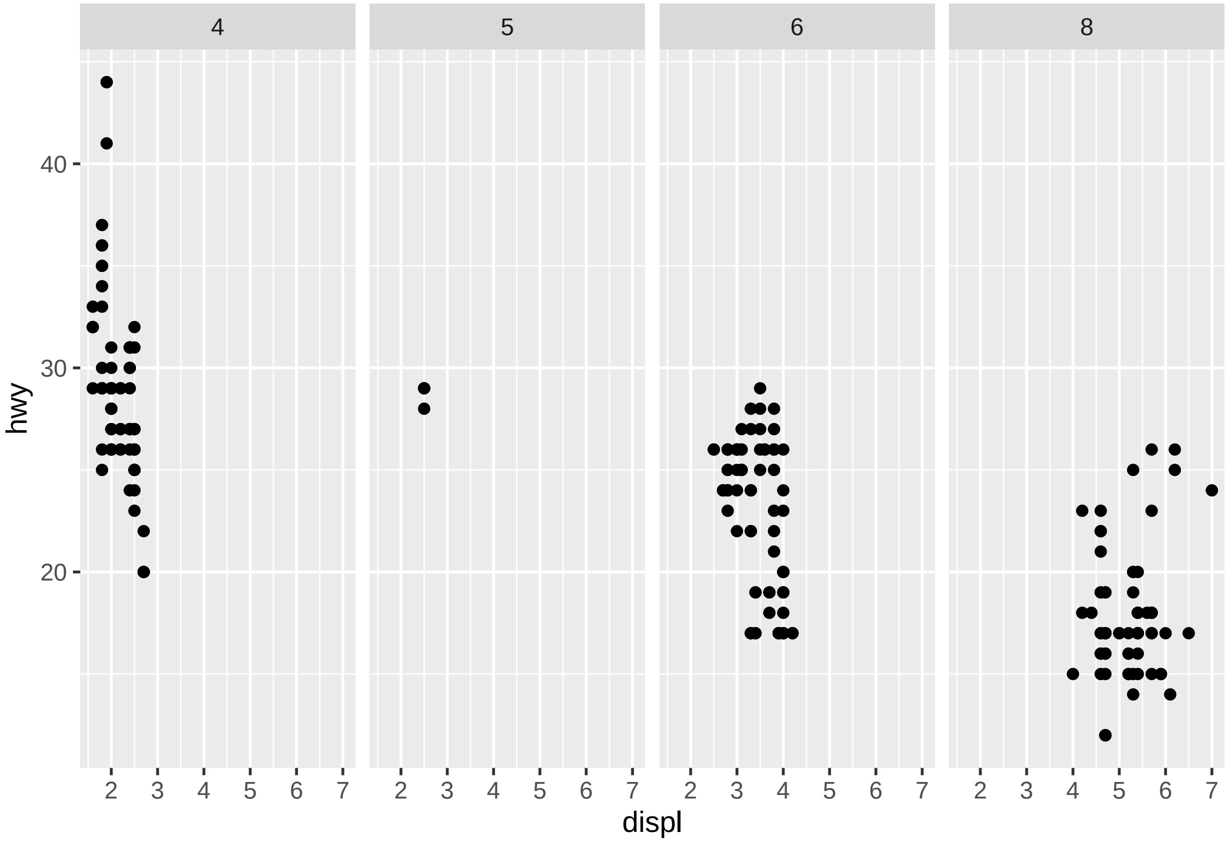
The empty cells (facets) in this plot are combination of `drv` and `cyl` that have no observations. These are the same locations in the scatter plot of `drv` and `cyl` that have no points

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



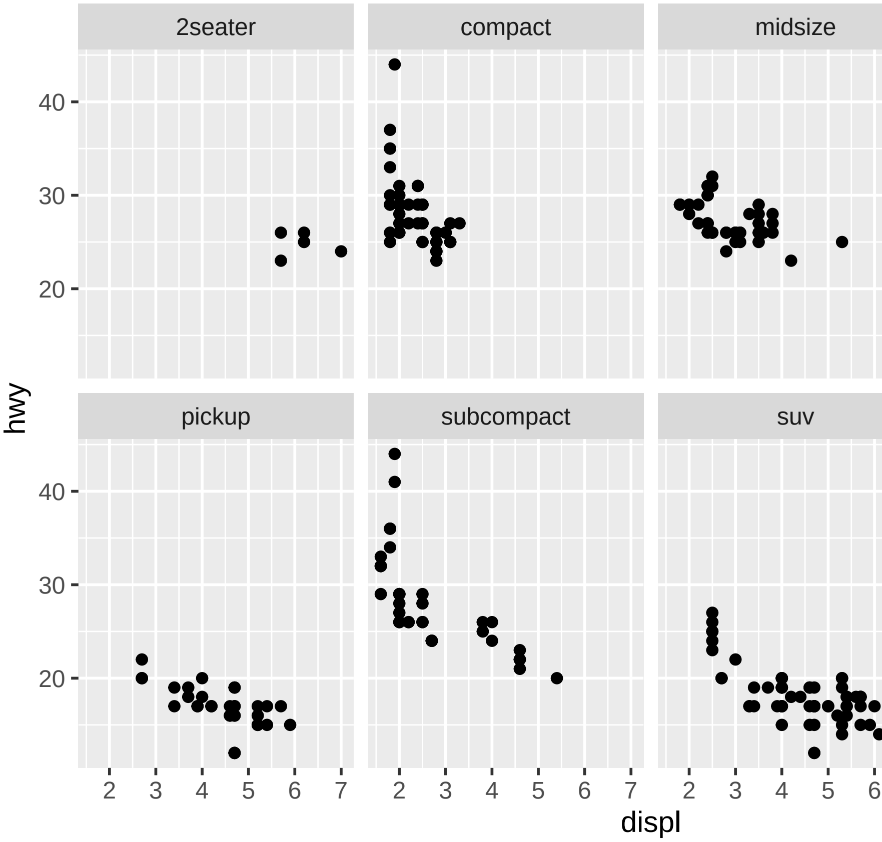
What plots does the following code make? what does . do?

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



The `drv ~ .` means facet by values of `drv` on the y-axis The `.~ drv` means facet by values of `drv` on the x-axis

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



Take the first faceted plot in this section

```
?facet_wrap
```

Read `?facet_wrap`. What does `nrow`,`ncol` do? what other options control the layout of the individual panels? Why doesn't facet_grid() have nrow and ncol argument? nrow, ncol : number of rows and columns

The `nrow` and `ncol` arguments are unnecessaary for `facet_grid()` since the number of unique values of the variables specified in the function determines the number of rows and columns

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

There will be more space for columns if the plot is laid out horizontally