R for Data Science - Solutions Manual

Jonathan Mendes de Almeida

Solutions to the exercises in **R for Data Science (Garrett Grolemund & Hadley Wickham).**

Chapter 1

No exercises in this chapter.

Chapter 2

No exercises in this chapter.

Chapter 3

Prerequisites

```r
library(tidyverse)
```

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

```
## √ ggplot2 2.2.1     √ purrr   0.2.4
## √ tibble  1.4.2     √ dplyr   0.7.4
## √ tidyr   0.8.0     √ stringr 1.3.0
## √ readr   1.1.1     √ forcats 0.3.0
```

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

To install tidyverse library:

```r
install.packages('tidyverse')
```

mpg data frame

```r
mpg = ggplot2::mpg
mpg
```

```
## # A tibble: 234 x 11
##    manufacturer model    displ  year   cyl trans   drv     cty   hwy fl
##    <chr>        <chr>    <dbl> <int> <int> <chr>   <chr> <int> <int> <chr>
##  1 audi         a4        1.80  1999     4 auto(l~ f        18    29 p
##  2 audi         a4        1.80  1999     4 manual~ f        21    29 p
##  3 audi         a4        2.00  2008     4 manual~ f        20    31 p
##  4 audi         a4        2.00  2008     4 auto(a~ f        21    30 p
##  5 audi         a4        2.80  1999     6 auto(l~ f        16    26 p
##  6 audi         a4        2.80  1999     6 manual~ f        18    26 p
##  7 audi         a4        3.10  2008     6 auto(a~ f        18    27 p
##  8 audi         a4 quat~  1.80  1999     4 manual~ 4        18    26 p
##  9 audi         a4 quat~  1.80  1999     4 auto(l~ 4        16    25 p
## 10 audi         a4 quat~  2.00  2008     4 manual~ 4        20    28 p
## # ... with 224 more rows, and 1 more variable: class <chr>
```

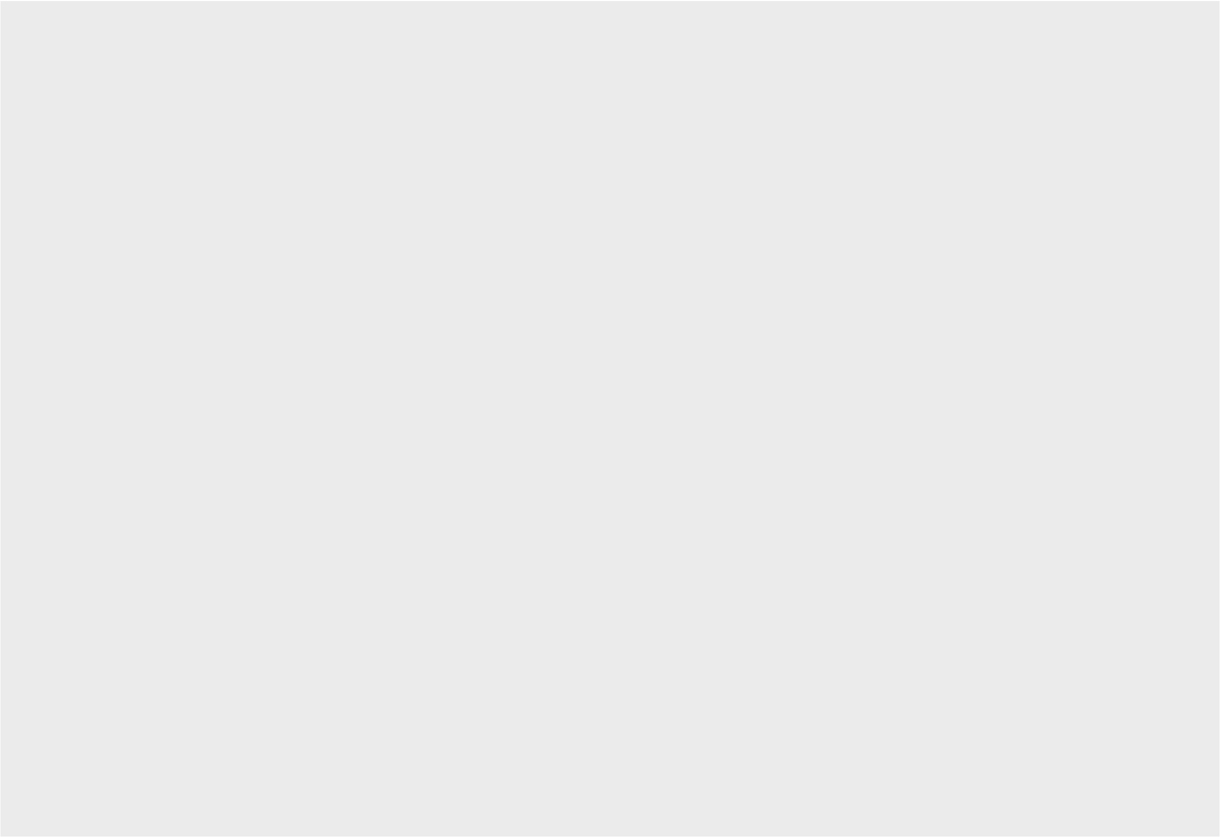
- **displ**: car's engine size (in litres)
- **hwy**: car's fuel efficiency on the highway (in miles per gallon - mpg).

3.2.4 Exercises

Exercise 1

> Run **ggplot(data = mpg)**. What do you see?

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



R) An empty plot. To see some nice plots we should add some geom_function to map some points. Add a good caption for each axis is great to make your plot easier to read and understand!

Exercise 2

> How many rows are in **mpg**? How many columns?

```r
nrow(mpg)
```

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

```r
ncol(mpg)
```

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

R) 234 rows and 11 columns

Alternative method to check the number of rows and columns of a data frame:

```r
glimpse(mpg)
```

```
## Observations: 234
## Variables: 11
## $ manufacturer <chr> "audi", "audi", "audi", "audi", "audi", "audi", "...
## $ model        <chr> "a4", "a4", "a4", "a4", "a4", "a4", "a4", "a4 qua...
## $ displ        <dbl> 1.8, 1.8, 2.0, 2.0, 2.8, 2.8, 3.1, 1.8, 1.8, 2.0,...
## $ year         <int> 1999, 1999, 2008, 2008, 1999, 1999, 2008, 1999, 1...
## $ cyl          <int> 4, 4, 4, 4, 6, 6, 6, 4, 4, 4, 4, 6, 6, 6, 6, 6, 6...
## $ trans        <chr> "auto(l5)", "manual(m5)", "manual(m6)", "auto(av)...
## $ drv          <chr> "f", "f", "f", "f", "f", "f", "f", "4", "4", "4",...
## $ cty          <int> 18, 21, 20, 21, 16, 18, 18, 18, 16, 20, 19, 15, 1...
## $ hwy          <int> 29, 29, 31, 30, 26, 26, 27, 26, 25, 28, 27, 25, 2...
## $ fl           <chr> "p", "p", "p", "p", "p", "p", "p", "p", "p", "p",...
## $ class        <chr> "compact", "compact", "compact", "compact", "comp...
```

Exercise 3

> What does the **drv** variable describe? Read the help for ?mpg to find out. *run ?mpg in RStudio console and check the 'help' tab)*

```r
mpg['drv']
```

```
## # A tibble: 234 x 1
##    drv
##    <chr>
##  1 f
##  2 f
##  3 f
##  4 f
##  5 f
##  6 f
##  7 f
##  8 4
##  9 4
## 10 4
## # ... with 224 more rows
```

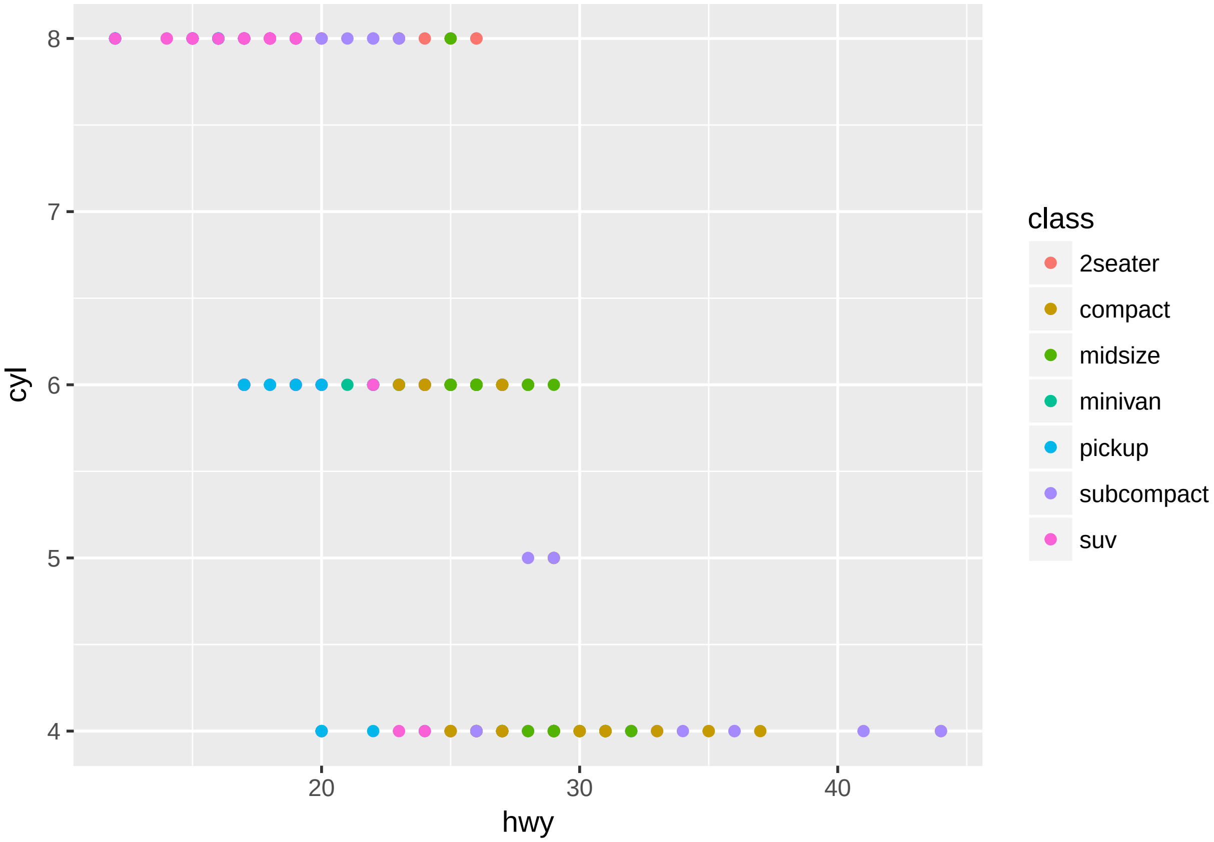
R) There are 3 possible values for **drv** variable (**variable** : *description*):

- **f**: *front-wheel drive*
- **r**: *rear wheel drive*
- **4**: *4wd*

Exercise 4

Make a scatterplot of **hwy** vs **cyl**

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

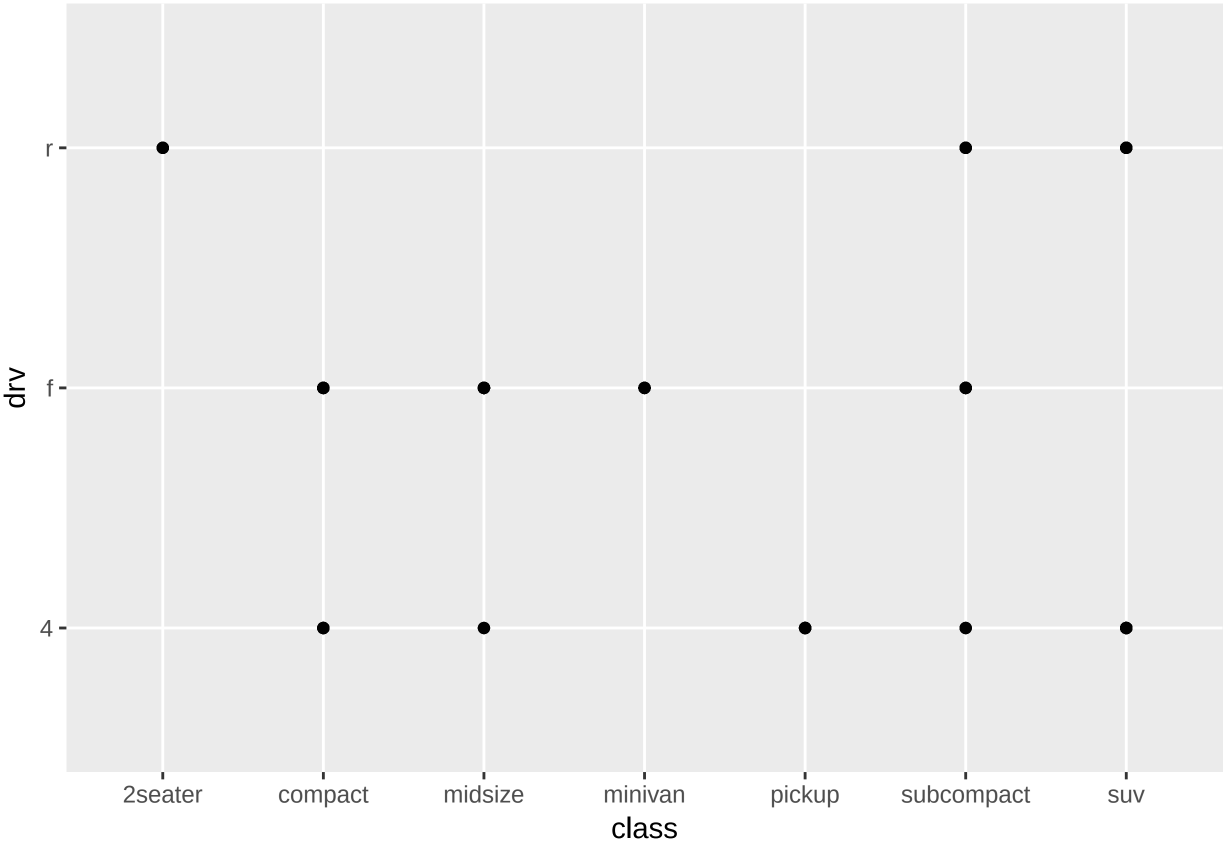


R) We add a definition of the data used in x,y axis and add geom_point function mapping these points. Here we are using aes(colour = class) (to associate the name of the aesthetic with a variable to display) to plot using a different colour for each class present in our data frame (car type: 2seater, compact, midsize, minivan, pickup, subcompact, suv).

Exercise 5

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

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



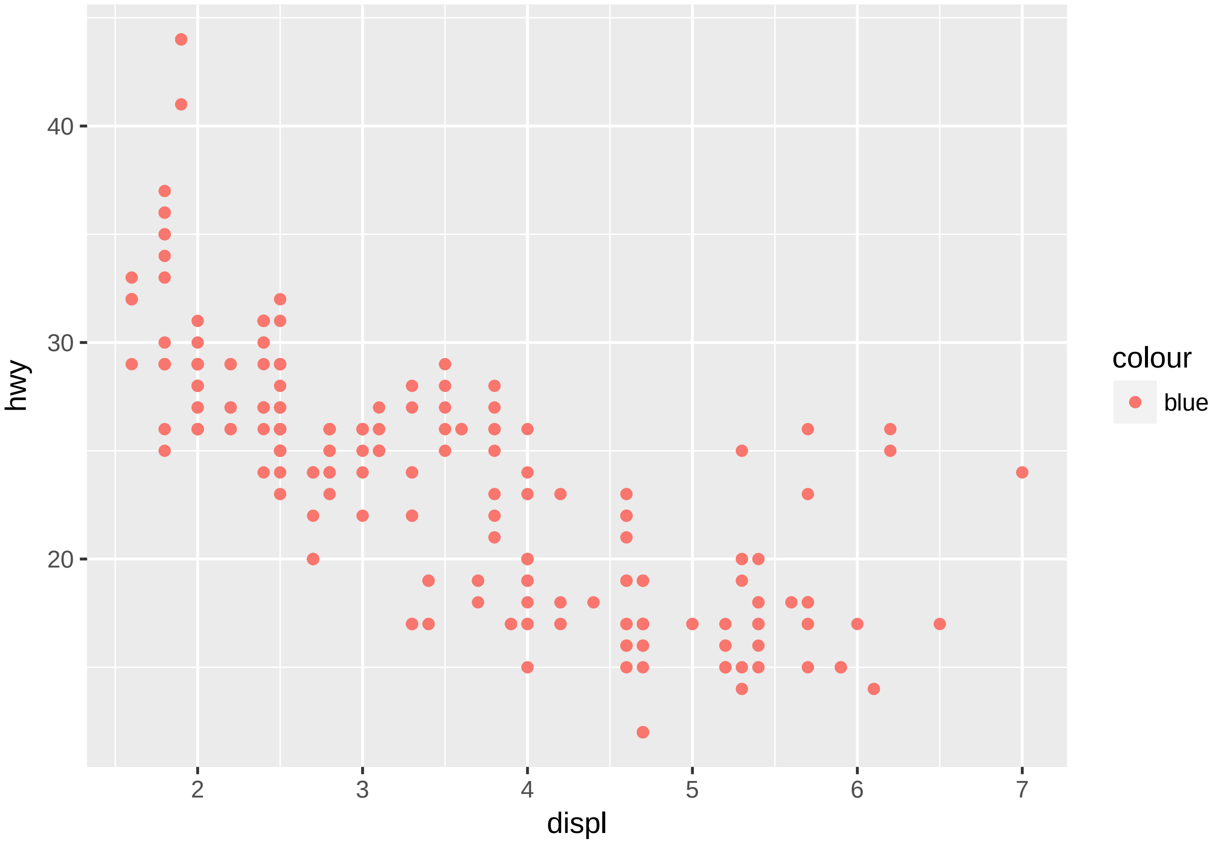
R) This plot is not useful because **class** and **drv** are factor variables. Each possible value of these two variables is limited by a set (r, f and 4 are the possible values for **drv** and 2seater, compact, midsize, minivan, pickup, subcompact and suv are the possible values for **class**). This plot is pretty useless to perform a data analysis.

3.3.1 Exercises

Exercise 1

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

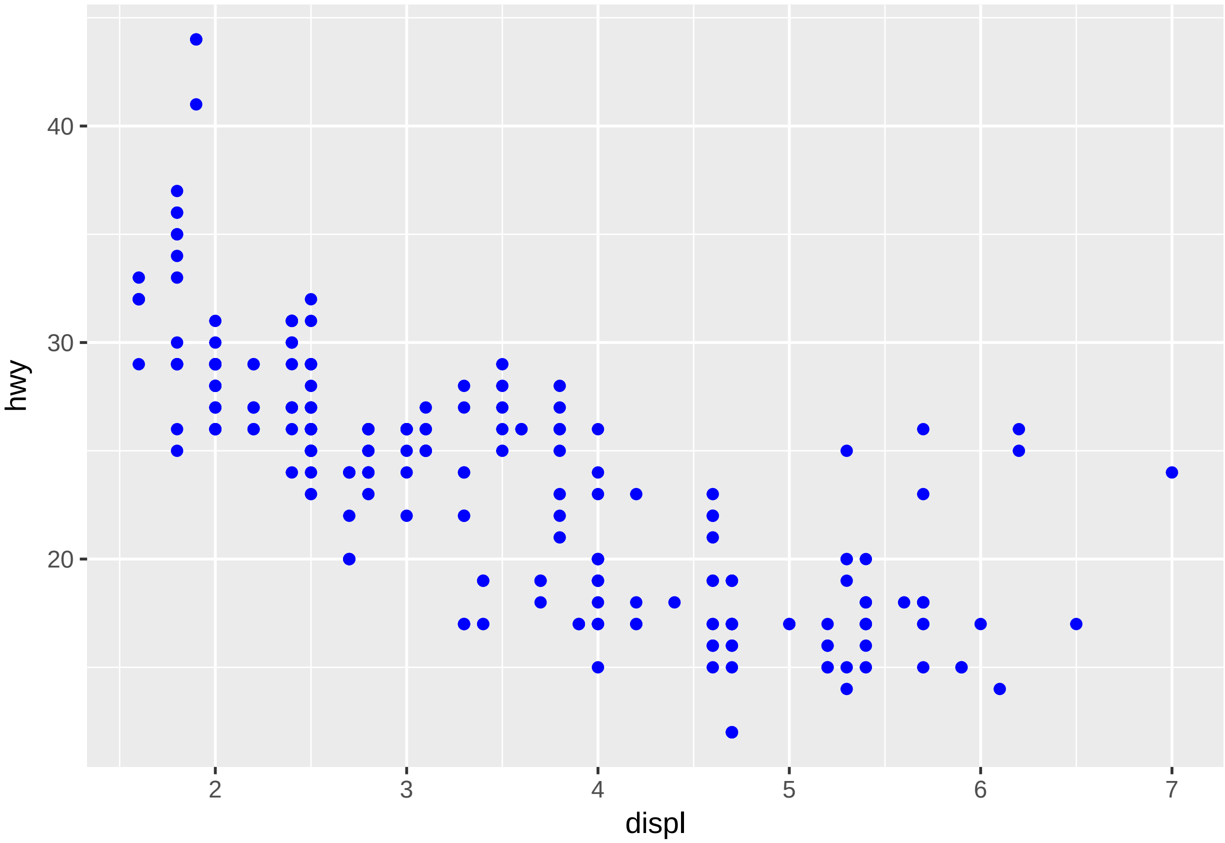
```
ggplot(data = mpg) + geom_point(mapping = aes(x = displ, y = hwy, color = "blue"))
```



R) The **color** argument is not in the correct place. The **color** argument is included inside the **mapping** argument so it is treated as an aesthetic, which receives a variable (like we used **class** as argument in previous exercise). In this case, the **color** argument is interpreted as a variable with only one value (which is "blue" in this case).

If the goal is to plot all these points using blue, the correct code is:

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



Exercise 2

> Which variables in **mpg** are categorical? Which variables are continuous? (**Hint:** *type* ***?mpg*** *(using RStudio console) to read the documentation for the dataset*). How can you see this information when you run **mpg**?

If you are not able to classify each variable as categorical or continuous by checking the description of each variable (by typing **?mpg**) you can print the data frame and R will answer this for you (another way to check this information is using the glimpse() function).

```
mpg
```

```
## # A tibble: 234 x 11
##    manufacturer model    displ  year   cyl trans   drv     cty   hwy fl
##    <chr>        <chr>    <dbl> <int> <int> <chr>   <chr> <int> <int> <chr>
##  1 audi         a4        1.80  1999     4 auto(l~ f        18    29 p
##  2 audi         a4        1.80  1999     4 manual~ f        21    29 p
##  3 audi         a4        2.00  2008     4 manual~ f        20    31 p
##  4 audi         a4        2.00  2008     4 auto(a~ f        21    30 p
##  5 audi         a4        2.80  1999     6 auto(l~ f        16    26 p
##  6 audi         a4        2.80  1999     6 manual~ f        18    26 p
##  7 audi         a4        3.10  2008     6 auto(a~ f        18    27 p
##  8 audi         a4 quat~  1.80  1999     4 manual~ 4        18    26 p
##  9 audi         a4 quat~  1.80  1999     4 auto(l~ 4        16    25 p
## 10 audi         a4 quat~  2.00  2008     4 manual~ 4        20    28 p
## # ... with 224 more rows, and 1 more variable: class <chr>
```

As you can see, the information is given at top of each column within '<>'. If the variable is categorical, it will have a class of ‘character’ (represented as). So, once you know where to find this is information is easy to anwser which variable is categorical and which is continuous.

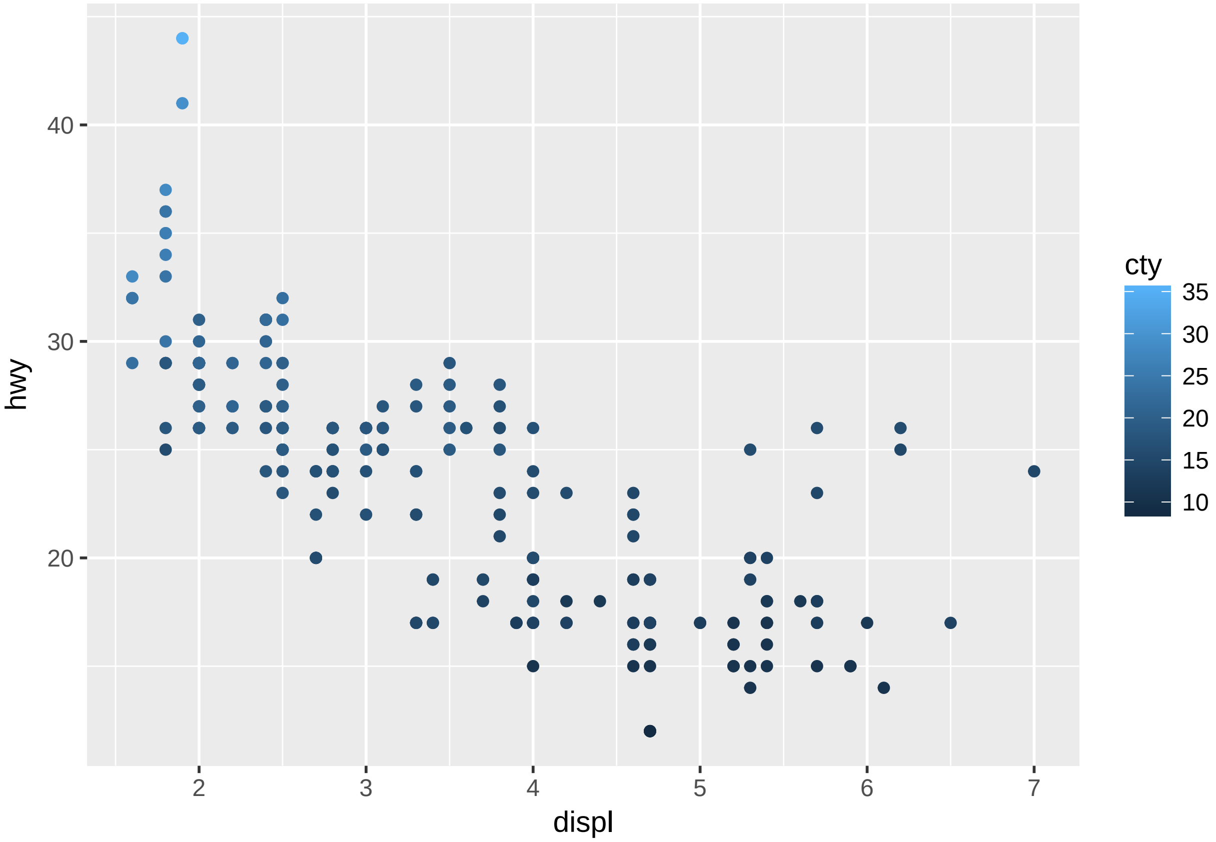
- **model**: categorical
- **displ**: continuous
- **year**: continuous
- **cyl**: continuous
- **trans**: categorical
- **drv**: categorical
- **cty**: continuous
- **hwy**: continuous
- **fl**: categorical
- **class**: categorical

Exercise 3

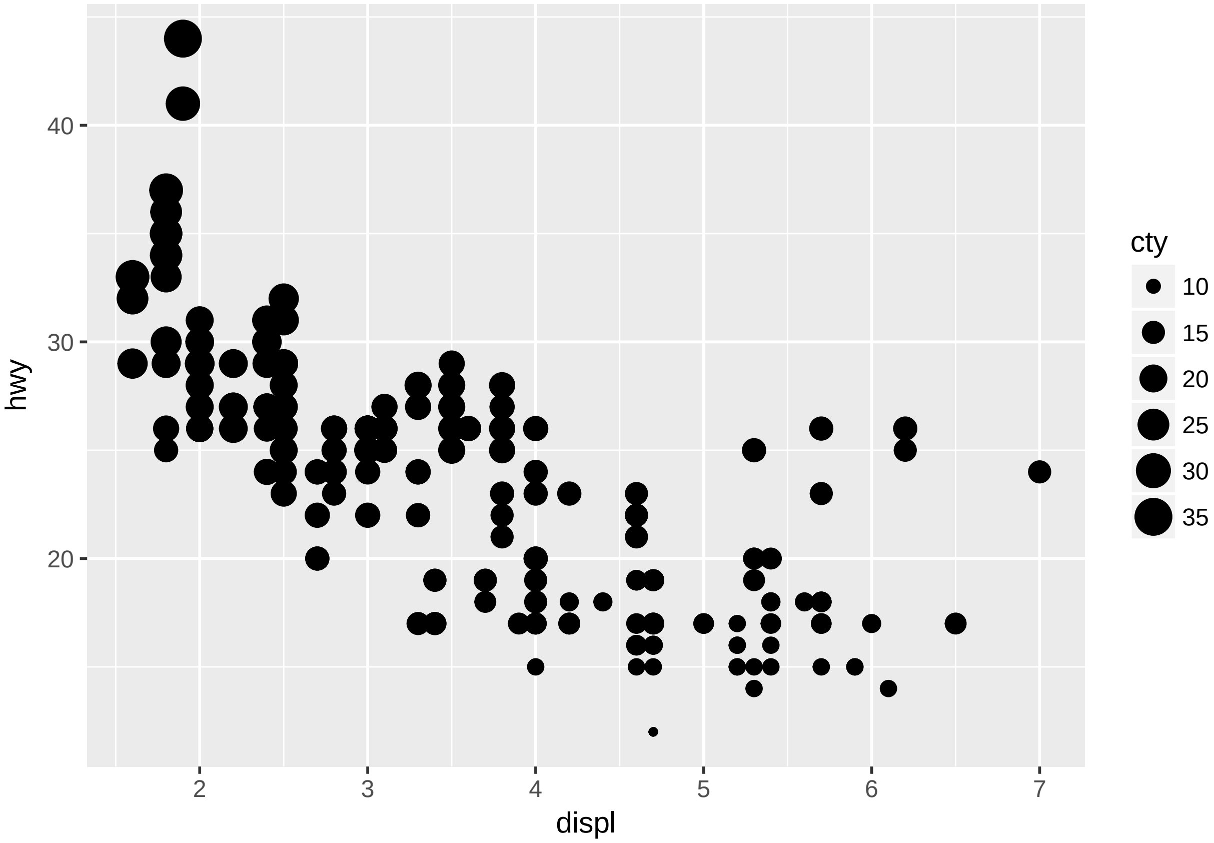
> Map a continuous variable to **color**, **size** and **shape**. How do these aesthetics behave differently for **categorical** *vs* **continuous** variables?

*Using the varible **cty** (city miles per gallon) - which is a continuous variable.*

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



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



R) When mapped to colour: the continuous variable uses a scale that varies using tons of blue (light to dark).

When mapped to size: the continuous variable uses a scale that varies using different sizes.

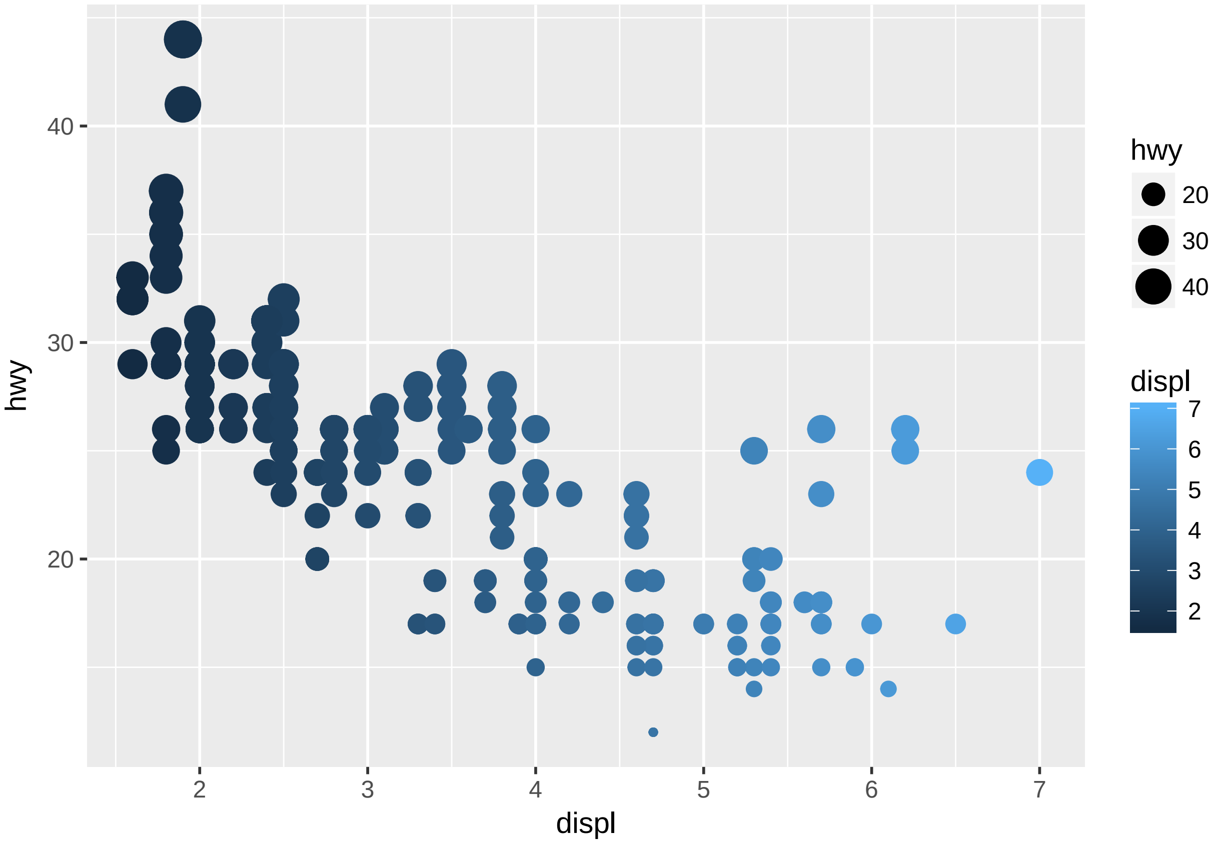
These information is easy to verify by checking these two previous plots.

However, when mapped to shape, R will give an error (a continuous variable can not be mapped to shape). This is because shapes does not have a natural order.

Exercise 4

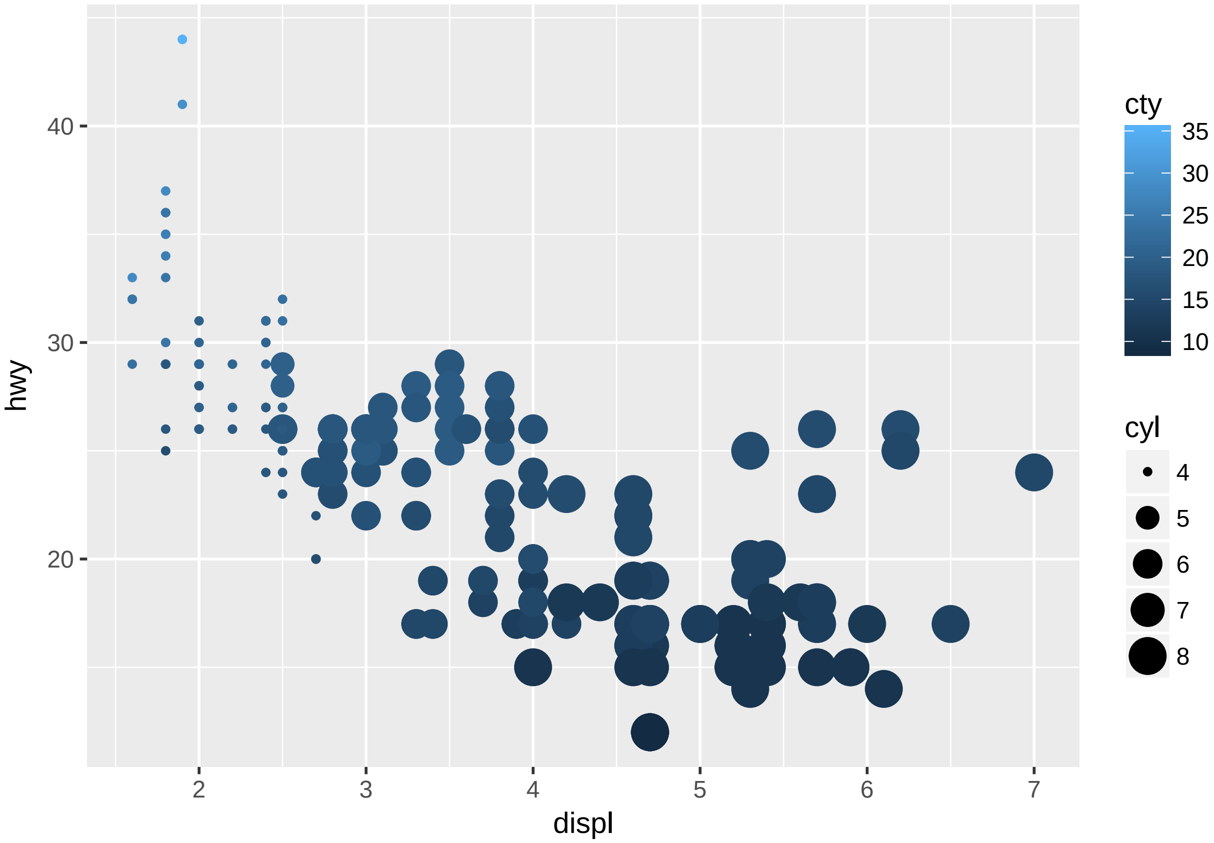
What happens if you map the same variable to multiple aesthetics?

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



Mapping one variable to multiple aesthetics is not a good idea because is redundant. Use different variables and the plot will show more information about your dataset. The next plot uses four different variables in aesthetics, which gives useful aditional information when compared to all the previous plots.

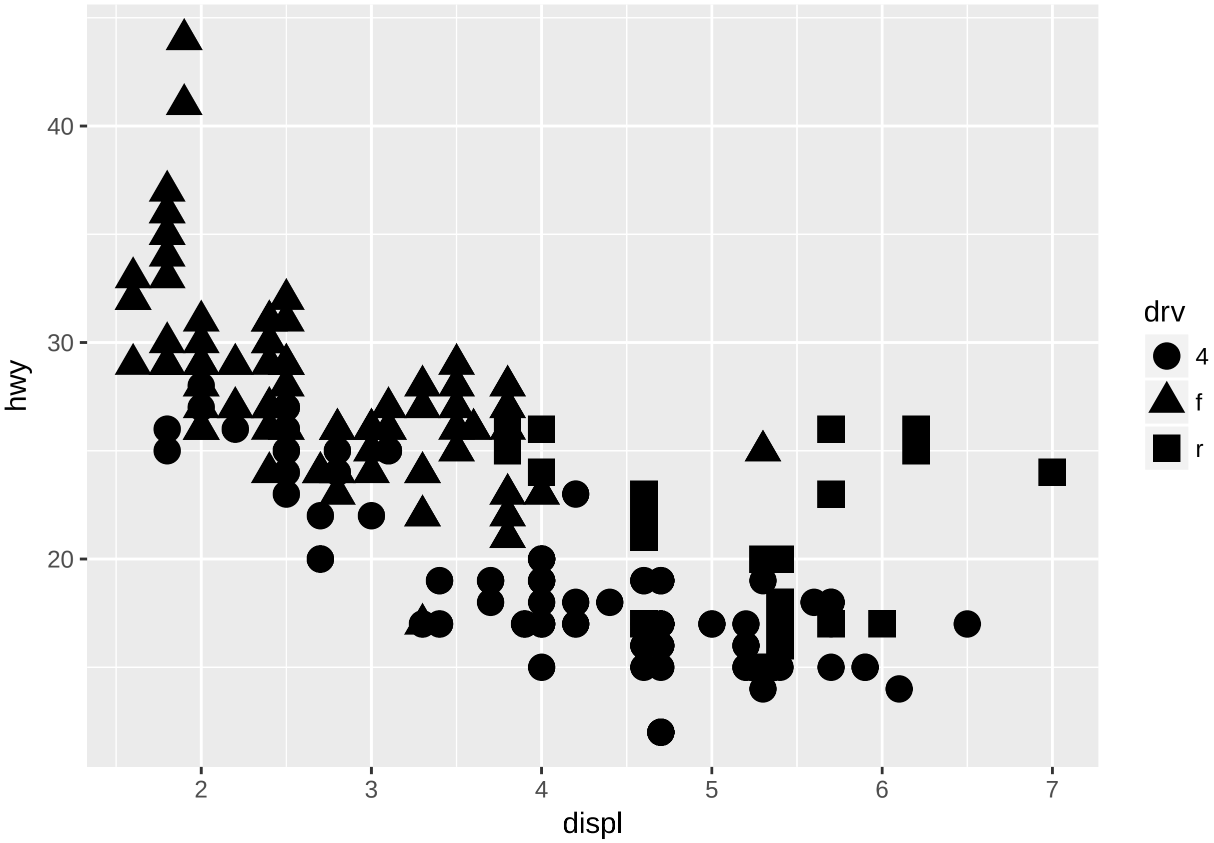
```
ggplot(data = mpg) + geom_point(mapping = aes(x = displ, y = hwy, colour = cty, size = cyl))
```



Exercise 5

What does the **stroke** aesthetic do? What shapes does it work with? (Hint: use **?geom_point** and check the 'help' tab)

```
ggplot(data = mpg) + geom_point(mapping = aes(x = displ, y = hwy, shape = drv), stroke = 5)
```



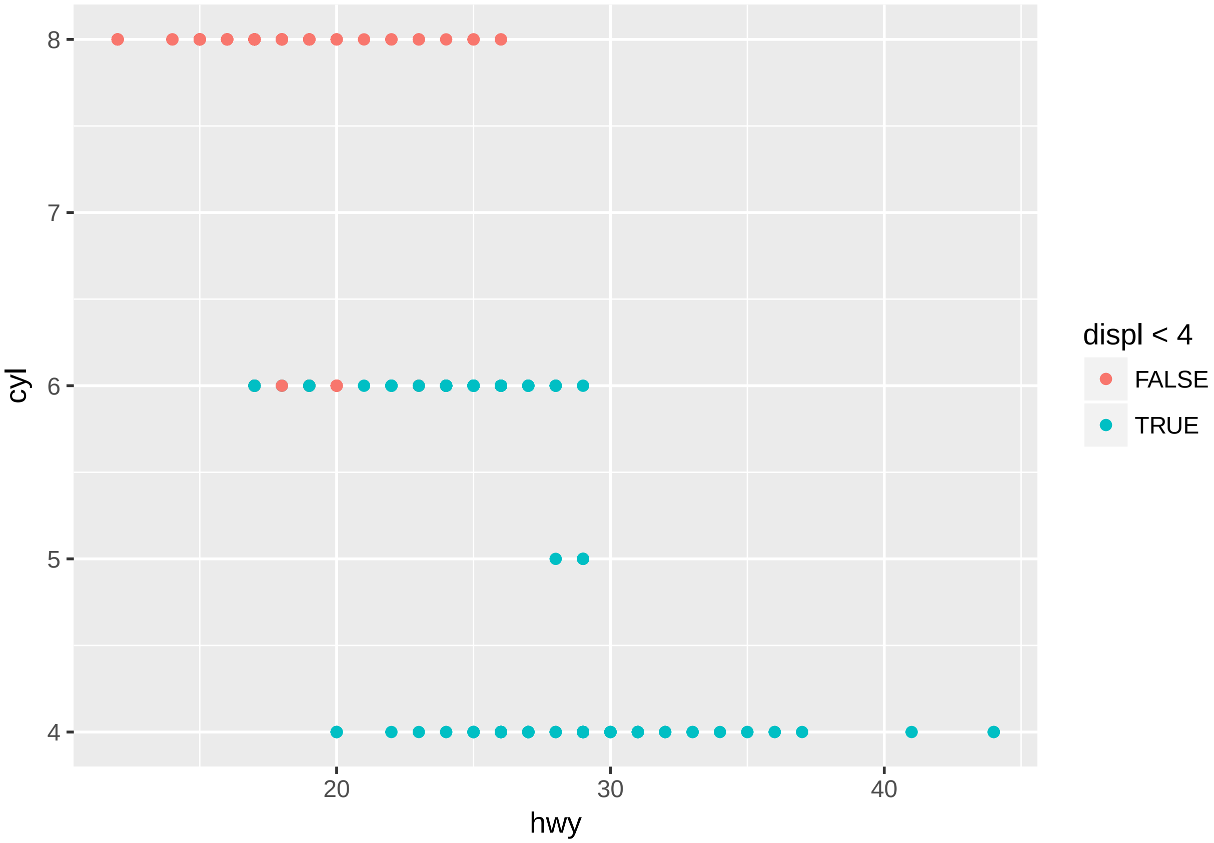
R) The **stroke** aesthetic is used to modify the width of the border.

Exercise 6

> What happens if you map an aesthetic to something other than a variable name, like **aes(colour = displ < 5)**?

R) The colour indicates if each **displ** value is less than 4 or not. The ggpplot function will assign the result of this expression (displ < 5, which is going to be true or false) to a temporary variable and then will assign a colour for values > 4 and a different colour for values < 4). This is easy visualize by checking the results of this code:

```
ggplot(data = mpg) + geom_point(mapping = aes(x = hwy, y = cyl, colour = displ < 4))
```


8
7
6
5
4
cyl
20
30
40
hwy
displ < 4
● FALSE
● TRUE