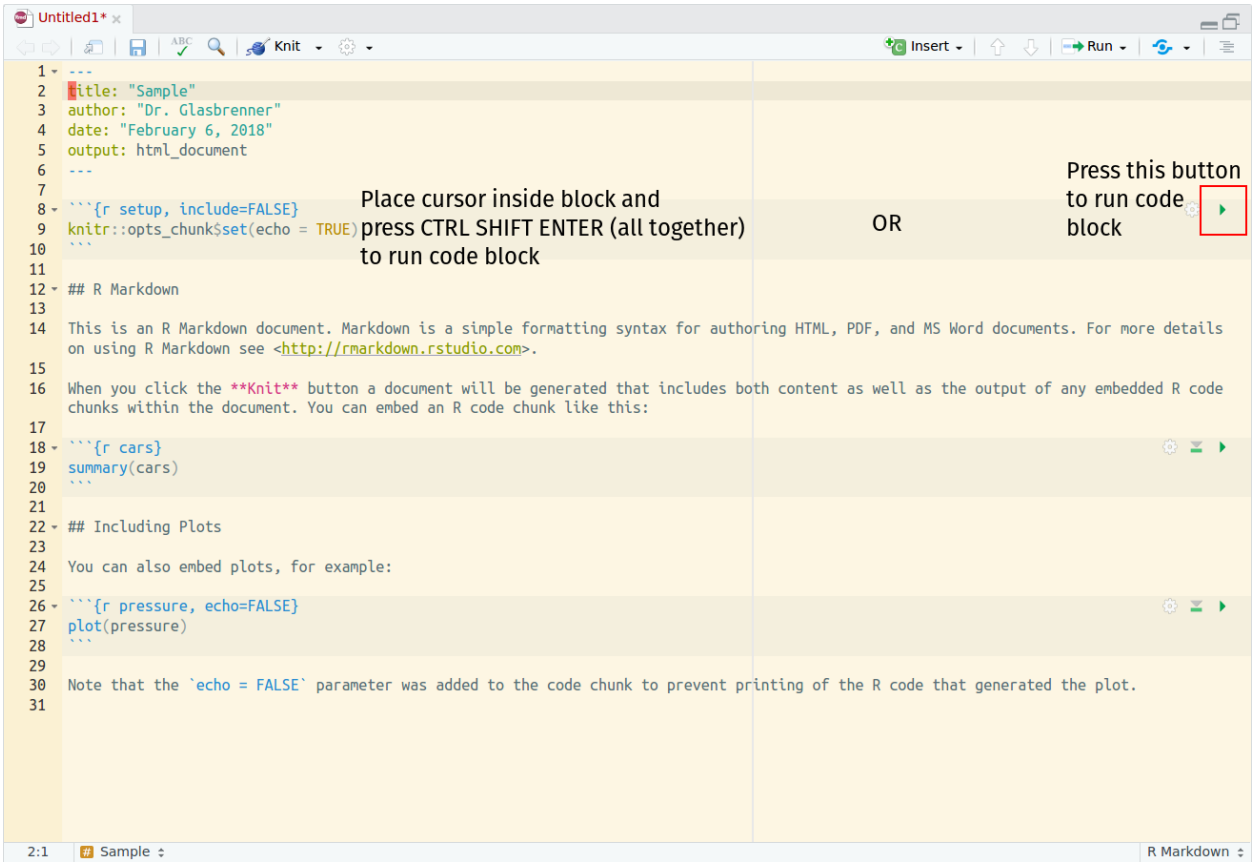
Mini-homework 2b: Visualization practice

Spencer Retcher

February 5, 2019

Interactive RStudio

RStudio can be used to run code blocks interactively and display the results, allowing you to preview what your knitted file will look like. You have two easy methods available to you for running the code in a code block. The graphical way is to click the green arrow located on the far-right side of a code block. The keyboard shortcut is place your cursor *inside* the block and then press `CTRL SHIFT ENTER` all together. The image below illustrates:



Use one of these methods whenever you want to run the code within a block.

Setup chunks

When you are given an RMarkdown template file, it will frequently have a *setup* code block at the top of the page, much like the one below. These chunks will configure the knitting procedure, which controls how your output documents will look. Often, they will also load the libraries you will need to complete an exercise or assignment. **If you reopen a file after restarting RStudio or switching projects, you should always run this block first.**

Give it a try, run the block below.

```
# DO NOT ALTER THIS CHUNK
knitr::opts_chunk$set(
  echo = TRUE,
  eval = TRUE,
  fig.width = 5,
  fig.asp = 0.618,
  out.width = "70%",
  dpi = 120,
  fig.align = "center",
  cache = FALSE
)
# Load required packages
suppressPackageStartupMessages(library(tidyverse))
```

If you didn't get an error, then excellent! If so, then you need to complete the RStudio Server Initial Configuration first. If you need help, you can leave a post on Slack.

Demo: The `mpg` dataset

Viewing the dataset

When loading `tidyverse`, several practice datasets are automatically loaded, one of which is the `mpg` dataset. It's good practice to start by looking at the dataset and getting familiar with the different columns and rows. You can do this within RStudio, try running the code block below:

```
mpg
```

You can also read more about the dataset by running:

```
?mpg
```

in the *Console* window.

Question

What is the data contained within the `mpg` dataset?

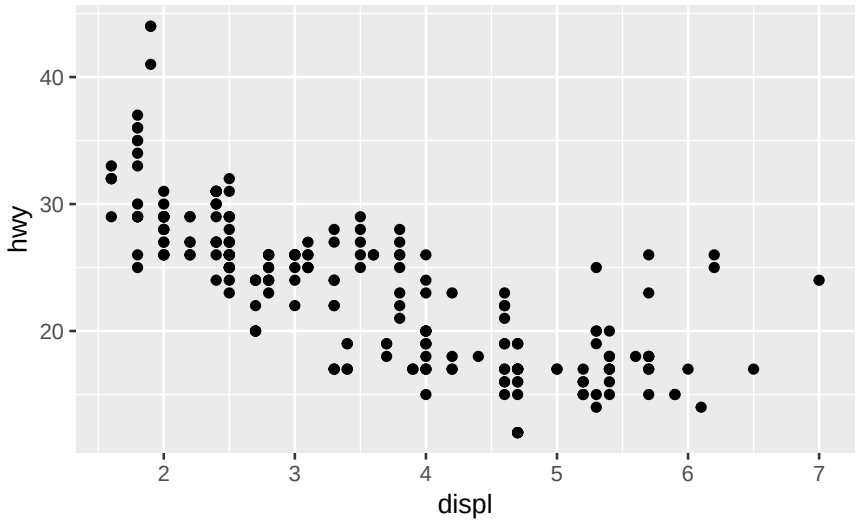
Answer

The mpg dataset contains characteristics about a group of cars. These characteristics include things like the car's model, manufacturer, and other important information.

Make a scatterplot

It's very easy to make a scatterplot in R using the `ggplot2` library. This is the library you will read about in Chapter 3 of *R for Data Science*. Let's use it to plot each car's highway fuel efficiency (`hwy`) as a function of the engine size (`displ`). The code block below will make this plot, try running it!

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



Question

Using your intuition, what part of the above code block tells R to put the `displ` variable on the horizontal axis and the `hwy` variable on the vertical axis?

Answer

The part of the code which does this is `(x = displ, y = hwy)`

Make a slight change

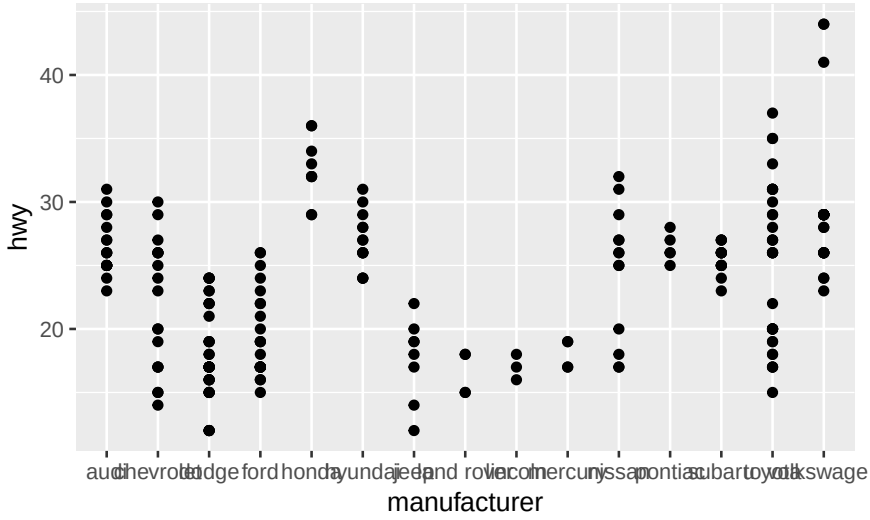
Exercise

Take a look at the `mpg` table again (in the section Viewing the dataset). Choose a **categorical variable** that isn't `displ`. Then, change the visualization code below so that your chosen variable replaces `displ` on the horizontal axis (the vertical should still be `hwy`).

Answer

I chose the categorical variable manufacturer.

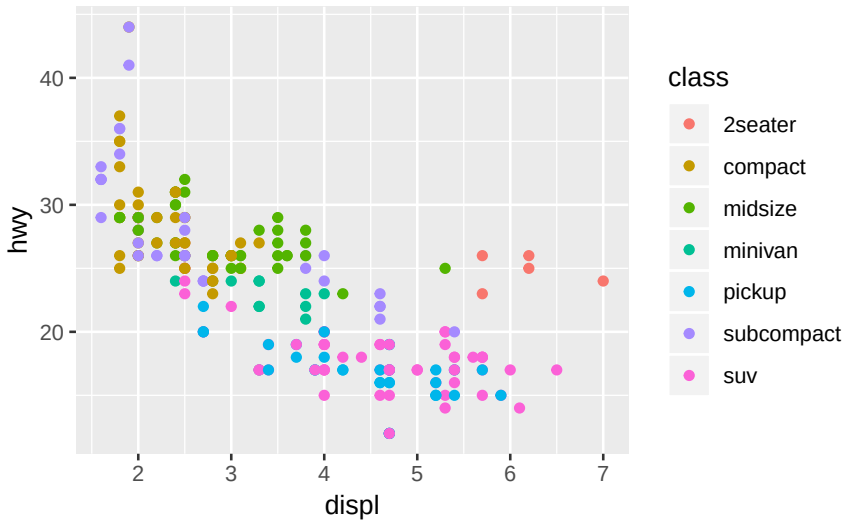
```
ggplot(data = mpg) +
  geom_point(mapping = aes(x =manufacturer, y = hwy))
```



Categories by color

The following code block extends the example with a new input, `color = class`. Try running it below:

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



Question

What does adding `color = class` inside the `aes()` code section do?

Answer

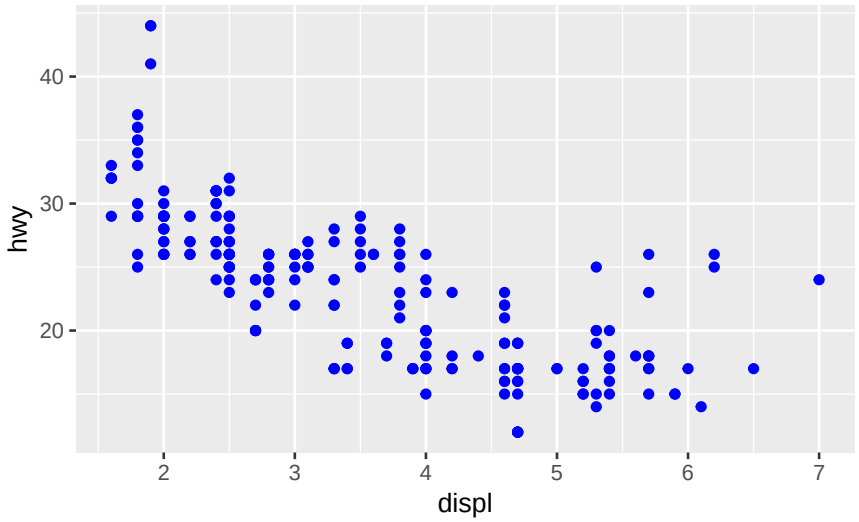
Changes the color of each data point on the graph

Other variations

Try running these other variations as well and observing their outputs.

First, we move `color` outside of `aes()` and set it equal to the *string* `"blue"`:

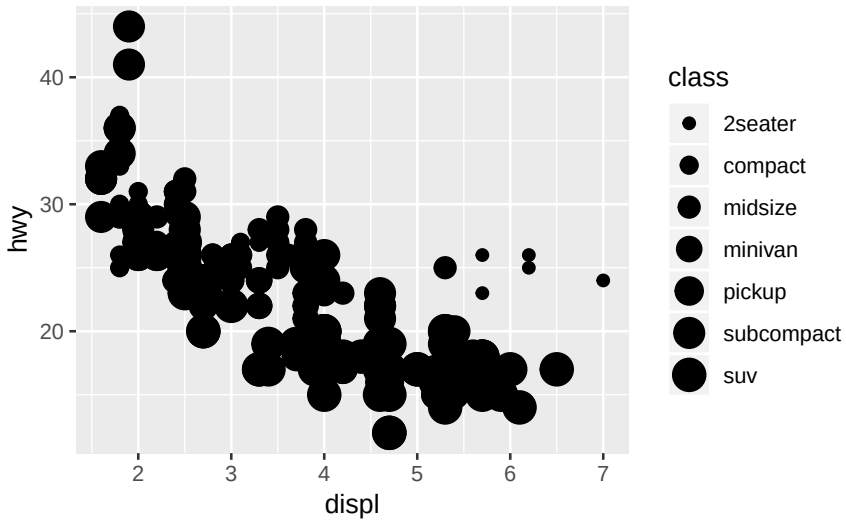
```
ggplot(data = mpg) +
  geom_point(mapping = aes(x = displ, y = hwy), color = "blue")
```



Here we replace `color` with `size`:

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

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

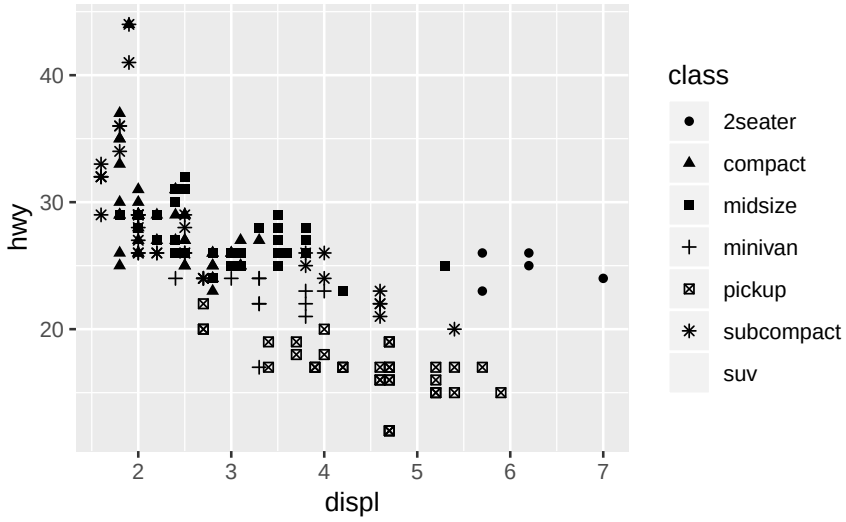


Here we replace `color` with `shape`:

```
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).
```

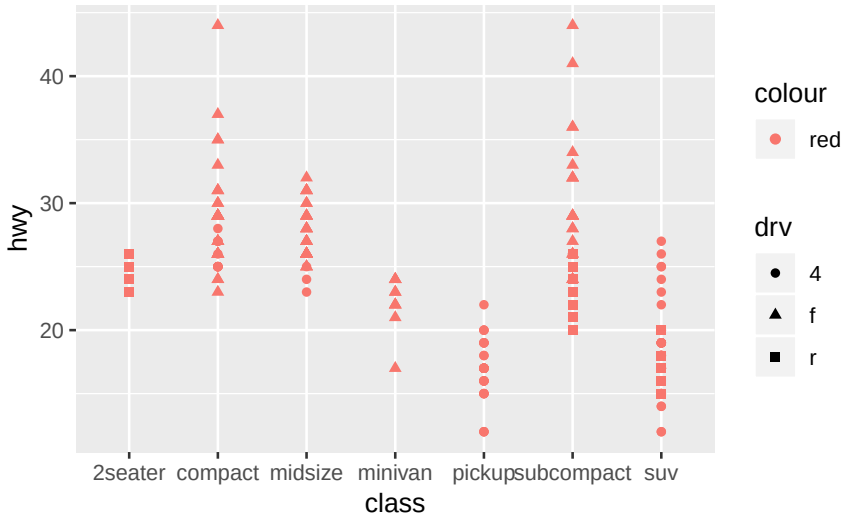


Exercise

Create your own figure using the `mpg` dataset! Follow the patterns you see in the above examples in order to build a new visualization. It should be different from the examples you've done so far, but otherwise, it's up to you.

Answer

```
# Put your code inside this block
ggplot(data = mpg) +
  geom_point(mapping = aes(x = class, y = hwy, shape = drv,color="red" ))
```



End

When you are done, save the file and try knitting it to PDF to make sure that it works. If you get an error message and it won't knit, then you will need to fix something in one of the code blocks. If the PDF formatting rendering looks funny, check through your RMarkdown file again and make sure you are following the proper RMarkdown syntax rules. Consult the RMarkdown Cheatsheet and RMarkdown Reference for help.