

chapter-one-exercise

Anyanwu Chinedu

Chapter one exercise

Package(s) & dataset(s) used in this exercise includes:

- tidyverse
- mpg data set

mpg dataset represent the Fuel economy data from the 1999 to 2008 for 38 popular models of cars

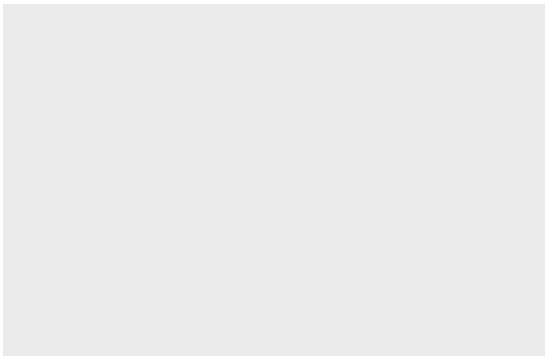
Question: Import the tidyverse package

Answer:

```
library(tidyverse)
```

Question: Run the ggplot(data=mpg). What do you see?

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



Answer:

The visual shows a blank output. This is because no mapping variables were specified

Question: What does the drv variable describe?

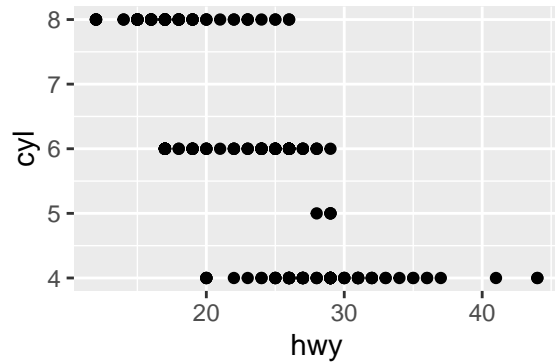
Answer:

This is the type of drive train, comprising f=front-wheel r=rear wheel drive, 4=4wd. This is according to the documentation on executing ?mpg

Question: Make a scatterplot of hwy versus cyl

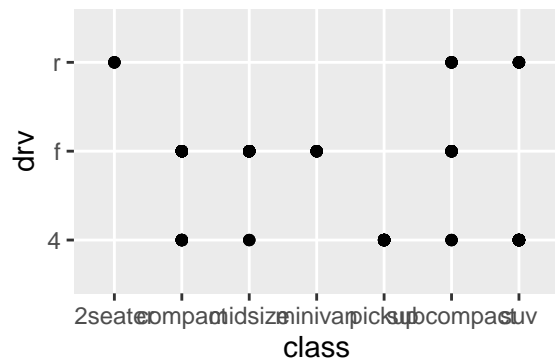
Answer:

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



Question: What happens if you make a scatterplot of class versus drv? Why is the plot not useful?

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

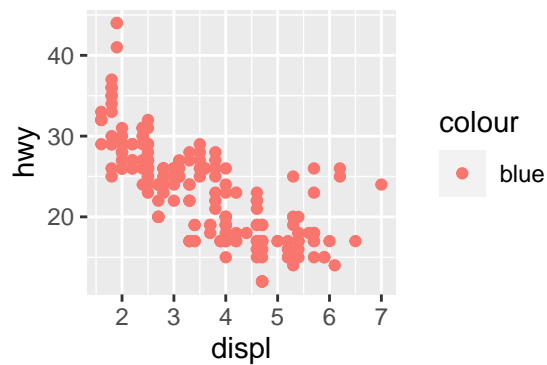


Answer:

Both `drv` & `class` are categorical variables taking small number of values. Hence the scatterplot is not useful

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

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

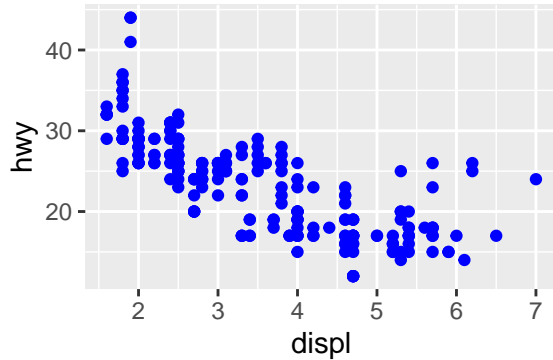


Answer:

The variable to the `color` argument as used here must be a known variable to the data set.

However, to make the points blue, the argument must be specified outside of `aes()` as shown below

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



Question: Which variables in mpg are categorical? Which variables are continuous?

Answer:

Categorical variables are discrete variables that take limited amount of values(eg - yes/no, dead/alive, male/female/binary, etc.), while continuous variables take fairly large number of values(eg - cities in Eastern Nigeria, species of Penguins etc.)

By running the command `View(data=mpg)`, we can discover that `class`, `drv`, `fl`, `hwy`, `cyl` are all categorical variables while the variables `manufacturer`, `model`, `displ`, `year`, `cty`, `hwy` are continuous

Question: What happens if you facet on a continuous variable?

Answer:

The visualization results in a more sub visuals with each continuous variable converting to a discrete variable

Question: What do empty cells in a plot with “`facet_grid(drv ~ cyl)`” mean? How do they relate to this plot?

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

Answer:

Empty cells in such as plot as this show points of no observations

Question: What geom would you use to draw a line a chat? A boxpplox? A histogram? An area chart?

`geom_line()`, `geom_boxplot()`, `geom_histogram` and `geom_area`

Will these two graphs look different? why/why not?

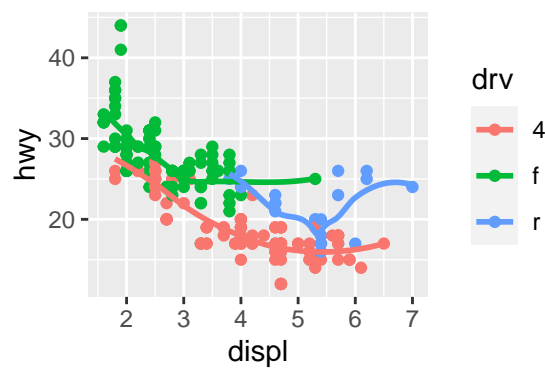
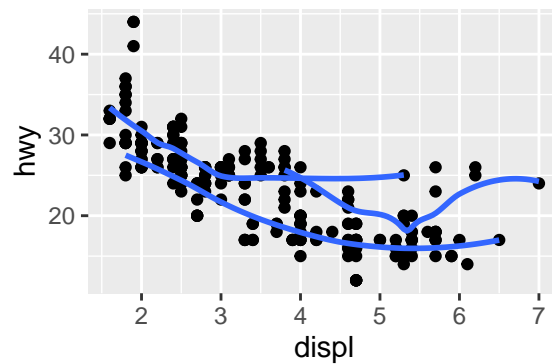
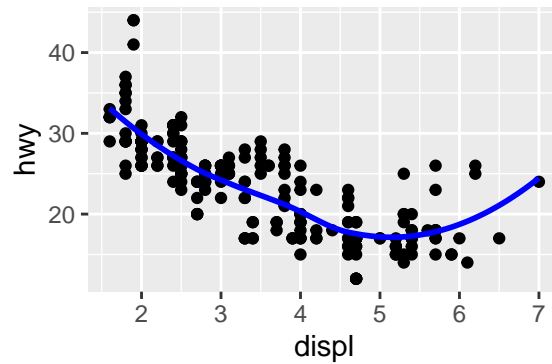
```
ggplot(data = mpg, mapping = aes(x = displ, y = hwy))+
  geom_point()+
  geom_smooth()
```

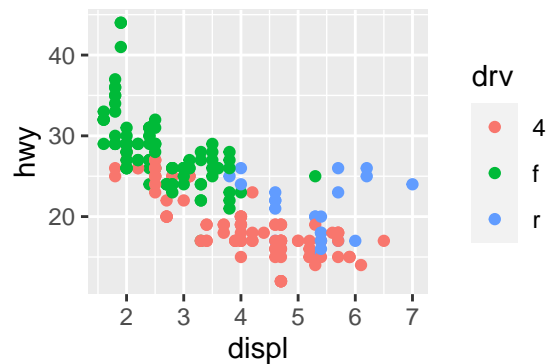
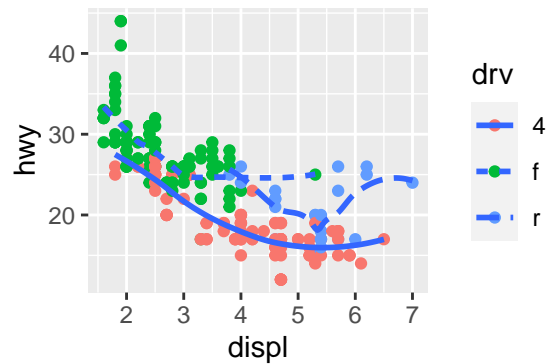
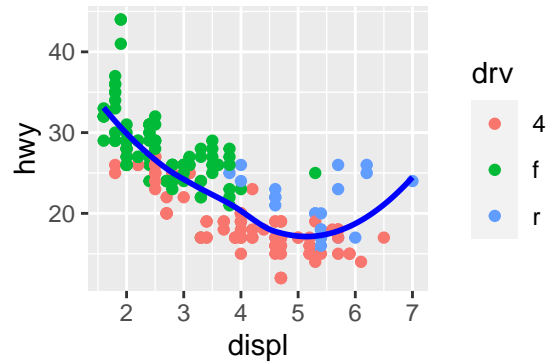
```
ggplot()+
  geom_point(data = mpg, mapping = aes(x = displ, y = hwy))+
  geom_smooth(data = mpg, mapping = aes(x = displ, y = hwy))
```

Answer:

The both plots will not look different. In the first plot, the mapping and variables were made global so each of the geoms has access to the same data. However, in the second plot, the same mapping and variables were made local to the respective geoms

Question: Re-create the R code necessary to generate the following graphs





Answers:

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

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

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

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

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

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

Question: What is the default geom associated with stat_summary?

Answer:

geom_point

Question: What does geom_col() do? How is it different from geom_bar()?

Answer:

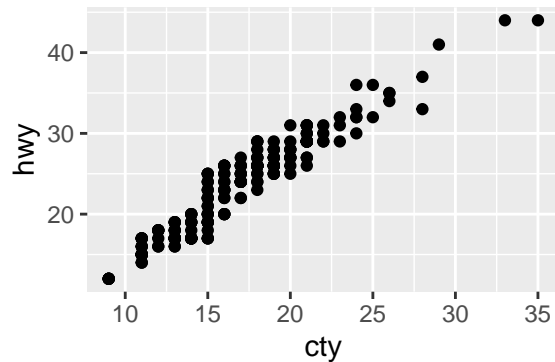
geom_col represents the heights of the bars with the values in the data set while geom_bar() makes the height of the bar proportional to the number of cases(that is occurrences) in each group

Question: What variables do stat_smooth() compute?

Answer: The predicted value, lower value of confidence interval and upper value confidence interval

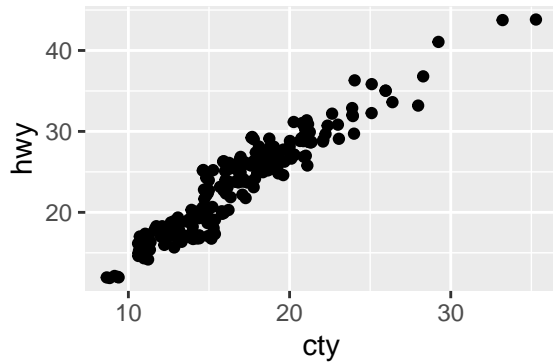
Question: What is the problem with this plot? How could you improve it?

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



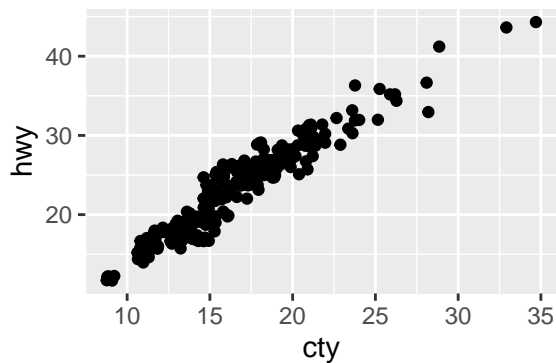
Answer: The outcome shows there is over-plotting on the visual due to the discreteness in smaller datasets. To improve this plot, the argument position = jitter should be supplied to the geom_point() to add random variations to the location of each point in the over-plotted visual as shown below

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



Alternative method: Use `geom_jitter()` instead of `geom_point()`

```
ggplot(data = mpg, mapping = aes(x = cty, y = hwy))+
  geom_jitter()
```



What parameters in `geom_jitter` control the amount of jittering?

Answer: The width which controls the amount of horizontal displacement and the height which controls the amount of vertical displacement

Compare and contrast `geom_jitter` and `geom_count`

Answer:

- `geom_jitter` is a convenient variation of `geom_point` which adds a random variation to the locations of each points in the dataset, useful in handling over-plotting
- `geom_count` is a variant of `geom_point` that counts the number of observations at each location, then maps the count to point area.
- Both `geom_jitter` and `geom_count` declare equal aesthetics arguments
- Both geom variants can declare `position` argument to override the defaults locations of datasets on the plots

To know more execute the commands: `?geom_jitter` and `?geom_count`