Problem Set 1

Meredith Gavin

04/01/2022

Front matter This submission is my work alone and complies with the 30535 integrity policy.

Add your initials to indicate your agreement: **MG**

Add your collaborators: **___**

Late coins used this pset: 0. Late coins left: 5.

```
library(tidyverse)
```

SET-UP

```
library("readr")
```

```
library("devtools")
```

```
devtools::install_github("hadley/r4ds")
```

#11

```
list.of.packages <- c("ggplot2", "Rcpp")
new.packages <- list.of.packages[!(list.of.packages %in% installed.packages()[,"Package"])]
if(length(new.packages)) install.packages(new.packages)

print(new.packages)
```

```
## character(0)
```

#12

github ID: meredithgavin

#13 #14 #15 Revert Practice

2.1 FIRST STEPS

1

The data frame mpg has 234 rows and 11 columns. The columns are manufacturer (maker of the vehicle), model (name of the car), displ (engine displacement in liters), year (the year the car was manufactured), cyl

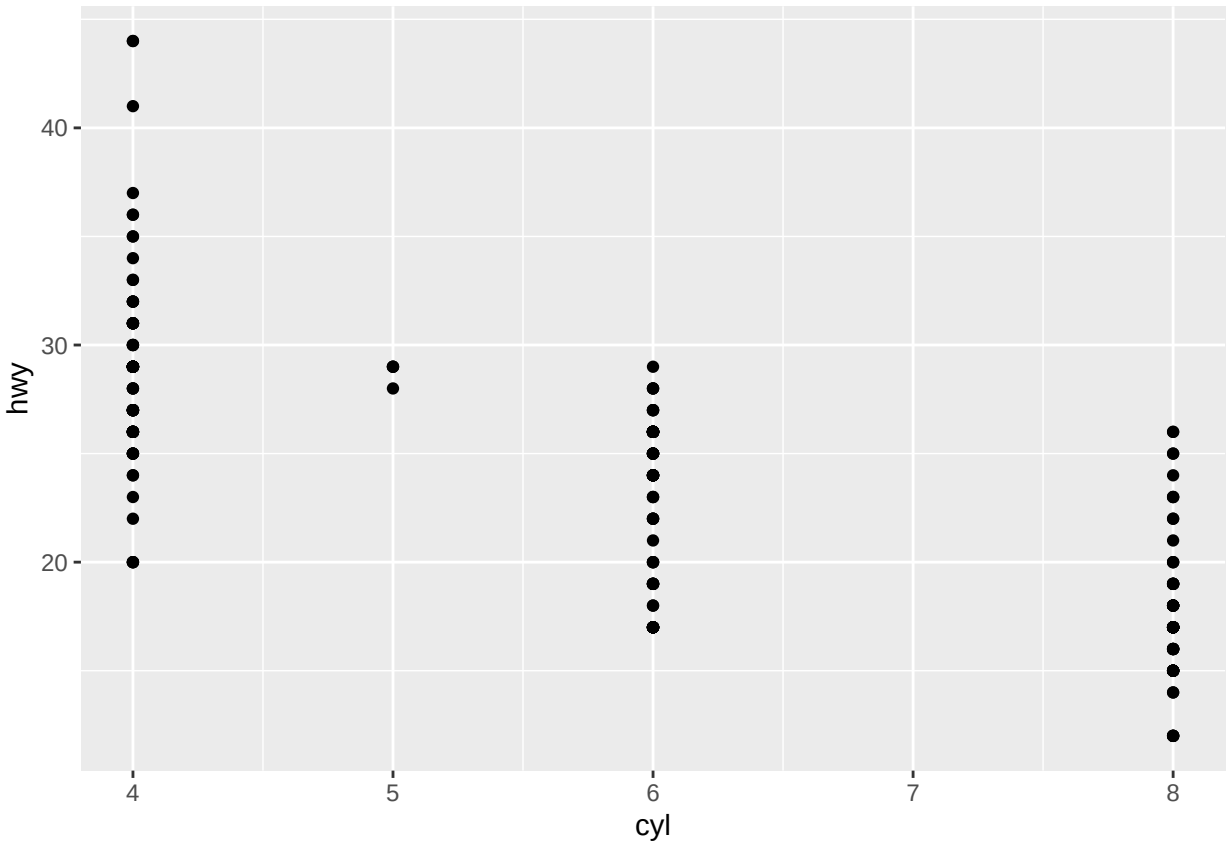
(number of cylinders), trans (transmission type), drv (the type of drive train - f is front-wheel, r is rear-wheel, 4 is four-wheel drive), cty (city mpg), hwy (highway mpg), fl (fuel type), and class ("type" of car).The rows are the observations below each variable.

```
?mpg

print(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~ f        18    29 p     comp~
##  2 audi         a4           1.8  1999     4 manu~ f        21    29 p     comp~
##  3 audi         a4           2    2008     4 manu~ f        20    31 p     comp~
##  4 audi         a4           2    2008     4 auto~ f        21    30 p     comp~
##  5 audi         a4           2.8  1999     6 auto~ f        16    26 p     comp~
##  6 audi         a4           2.8  1999     6 manu~ f        18    26 p     comp~
##  7 audi         a4           3.1  2008     6 auto~ f        18    27 p     comp~
##  8 audi         a4 quattro   1.8  1999     4 manu~ 4        18    26 p     comp~
##  9 audi         a4 quattro   1.8  1999     4 auto~ 4        16    25 p     comp~
## 10 audi         a4 quattro   2    2008     4 manu~ 4        20    28 p     comp~
## # ... with 224 more rows
```

2

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



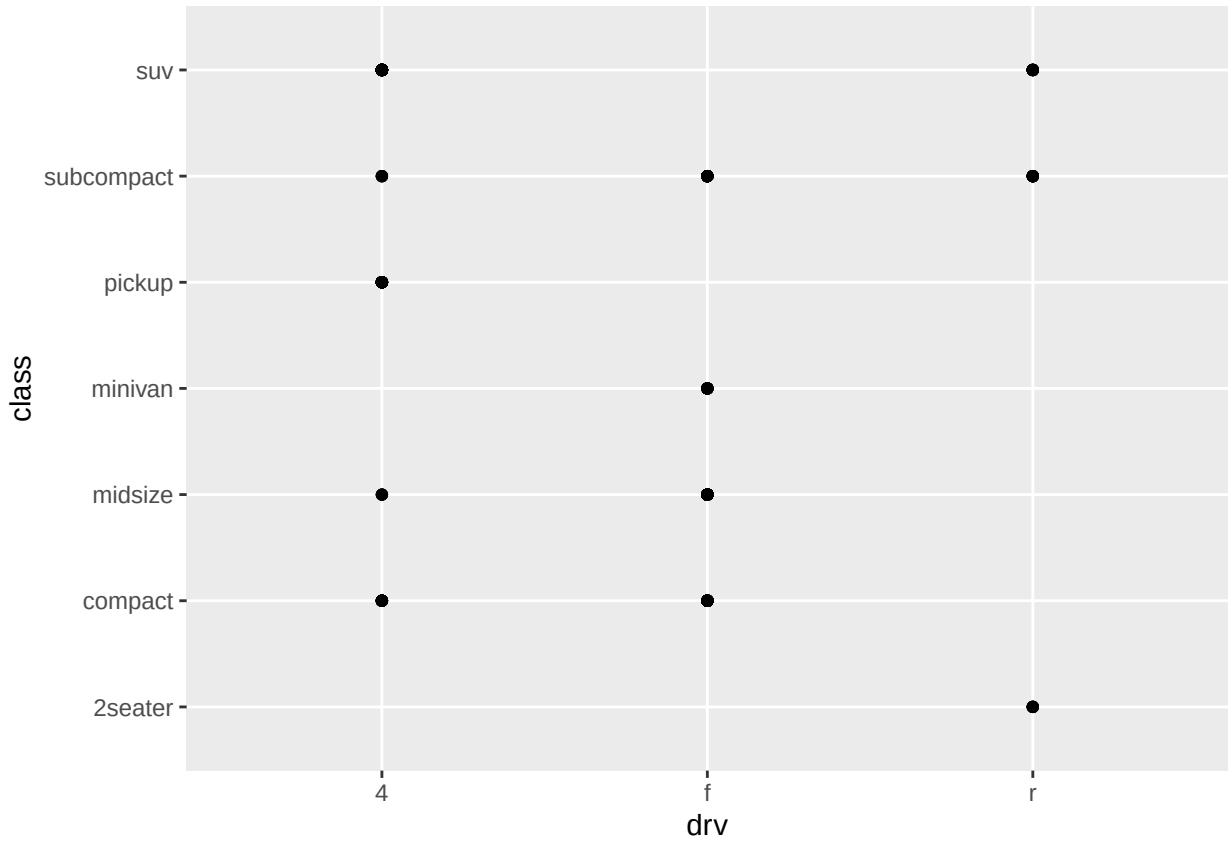
3

The variable drv is the vehicle's type of drive train. The observation can take the value f for front-wheel drive, r for rear-wheel drive, or 4 for four-wheel drive.

4

A plot of drv vs. cyl is not especially helpful because both drv and cyl are character variables. Neither takes a numerical value.

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



2.2: Grammar of Graphics: Mapping Data to Aesthetics

1

```
?mpg
summary(mpg)
```

```
##  manufacturer          model               displ           year
##  Length:234         Length:234         Min.   :1.600   Min.   :1999
##  Class :character   Class :character   1st Qu.:2.400   1st Qu.:1999
##  Mode  :character   Mode  :character   Median :3.300   Median :2004
##                                        Mean   :3.472   Mean   :2004
##                                        3rd Qu.:4.600   3rd Qu.:2008
##                                        Max.   :7.000   Max.   :2008
##       cyl           trans               drv                 cty
##  Min.   :4.000   Length:234         Length:234         Min.   : 9.00
##  1st Qu.:4.000   Class :character   Class :character   1st Qu.:14.00
##  Median :6.000   Mode  :character   Mode  :character   Median :17.00
##  Mean   :5.889                                         Mean   :16.86
##  3rd Qu.:8.000                                         3rd Qu.:19.00
##  Max.   :8.000                                         Max.   :35.00
##       hwy             fl                class
##  Min.   :12.00   Length:234         Length:234
##  1st Qu.:18.00   Class :character   Class :character
```

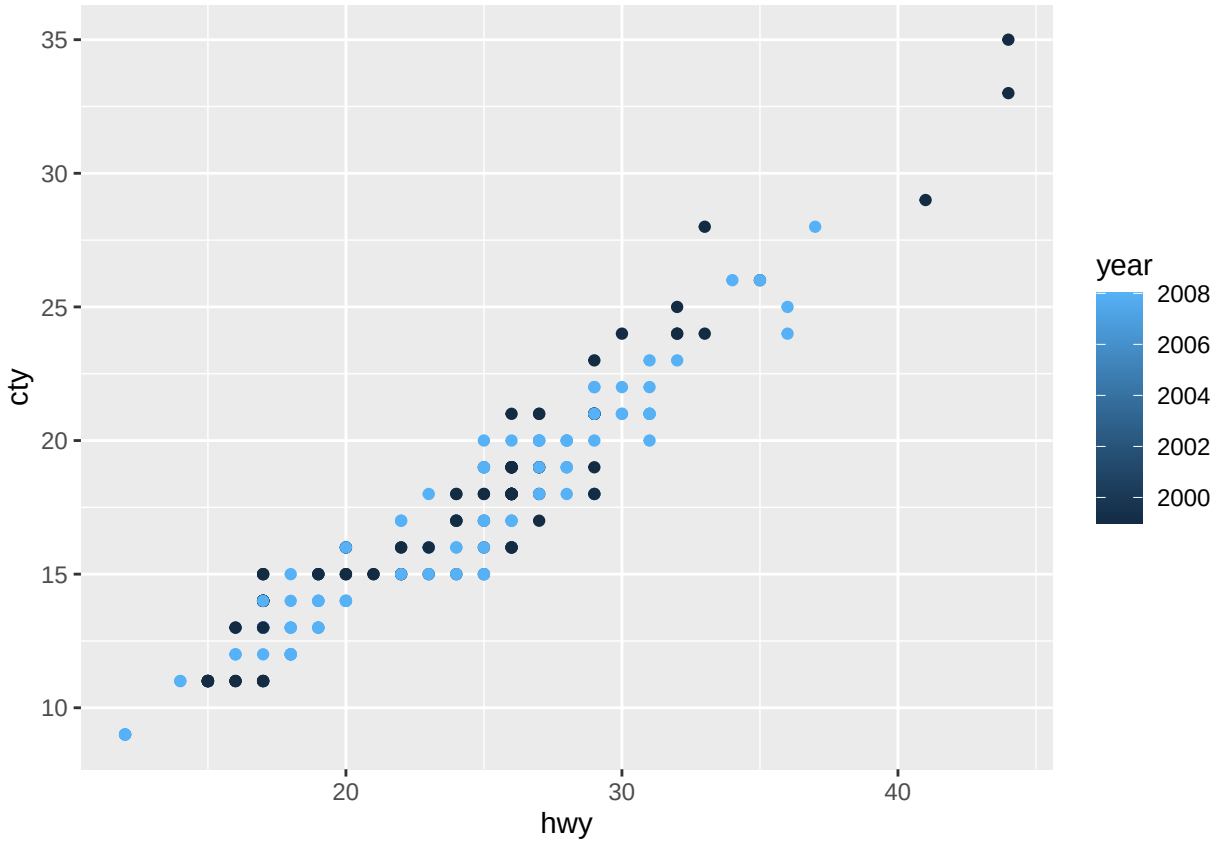
```
##  Median :24.00   Mode  :character   Mode  :character
##  Mean   :23.44
##  3rd Qu.:27.00
##  Max.   :44.00
```

```
print(mpg)
```

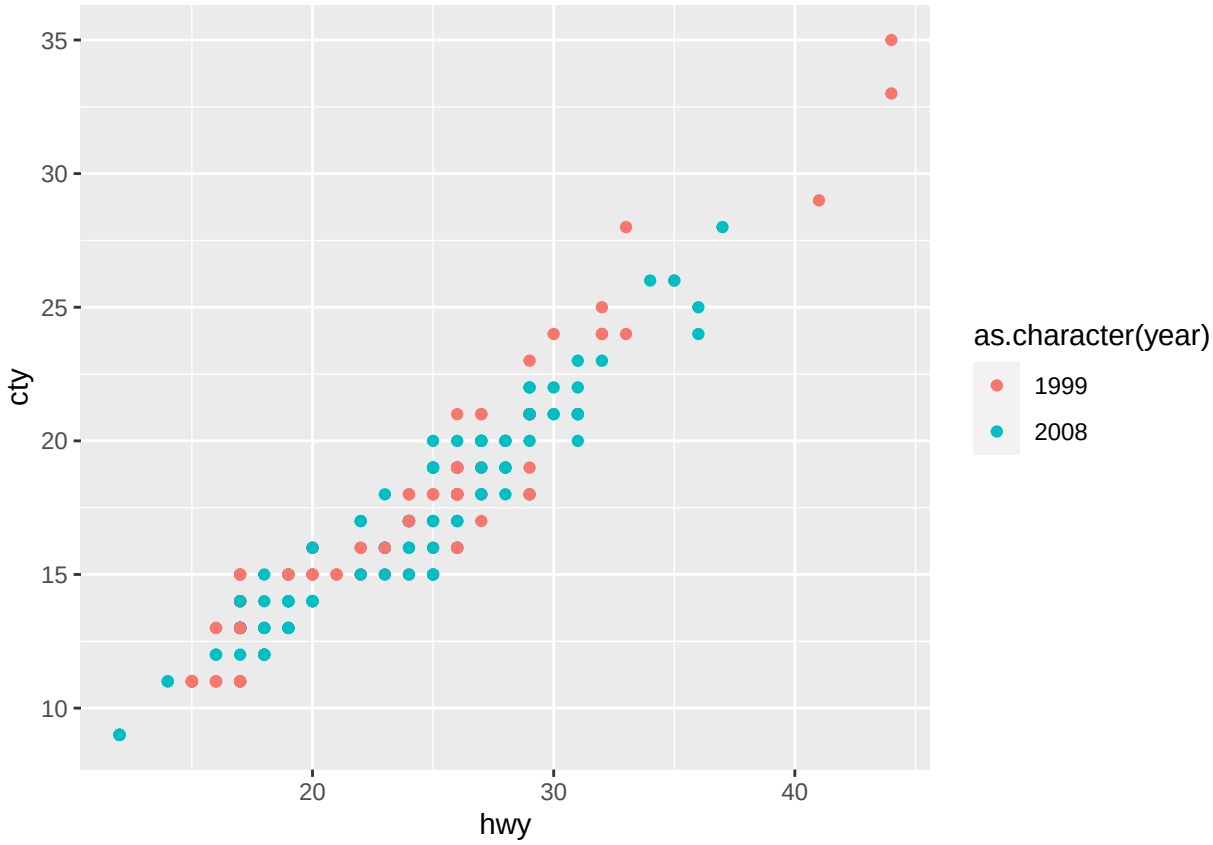
```
## # A tibble: 234 x 11
##    manufacturer model      displ  year   cyl trans drv     cty   hwy fl    class
##    <chr>        <chr>      <dbl> <int> <int> <chr> <chr> <int> <int> <chr> <chr>
##  1 audi         a4           1.8  1999     4 auto~ f        18    29 p     comp~
##  2 audi         a4           1.8  1999     4 manu~ f        21    29 p     comp~
##  3 audi         a4           2    2008     4 manu~ f        20    31 p     comp~
##  4 audi         a4           2    2008     4 auto~ f        21    30 p     comp~
##  5 audi         a4           2.8  1999     6 auto~ f        16    26 p     comp~
##  6 audi         a4           2.8  1999     6 manu~ f        18    26 p     comp~
##  7 audi         a4           3.1  2008     6 auto~ f        18    27 p     comp~
##  8 audi         a4 quattro   1.8  1999     4 manu~ 4        18    26 p     comp~
##  9 audi         a4 quattro   1.8  1999     4 auto~ 4        16    25 p     comp~
## 10 audi         a4 quattro   2    2008     4 manu~ 4        20    28 p     comp~
## # ... with 224 more rows
```

#2

```
# Graph 1
ggplot(data = mpg) +
geom_point(mapping = aes(x = hwy, y = cty, color = year))
```



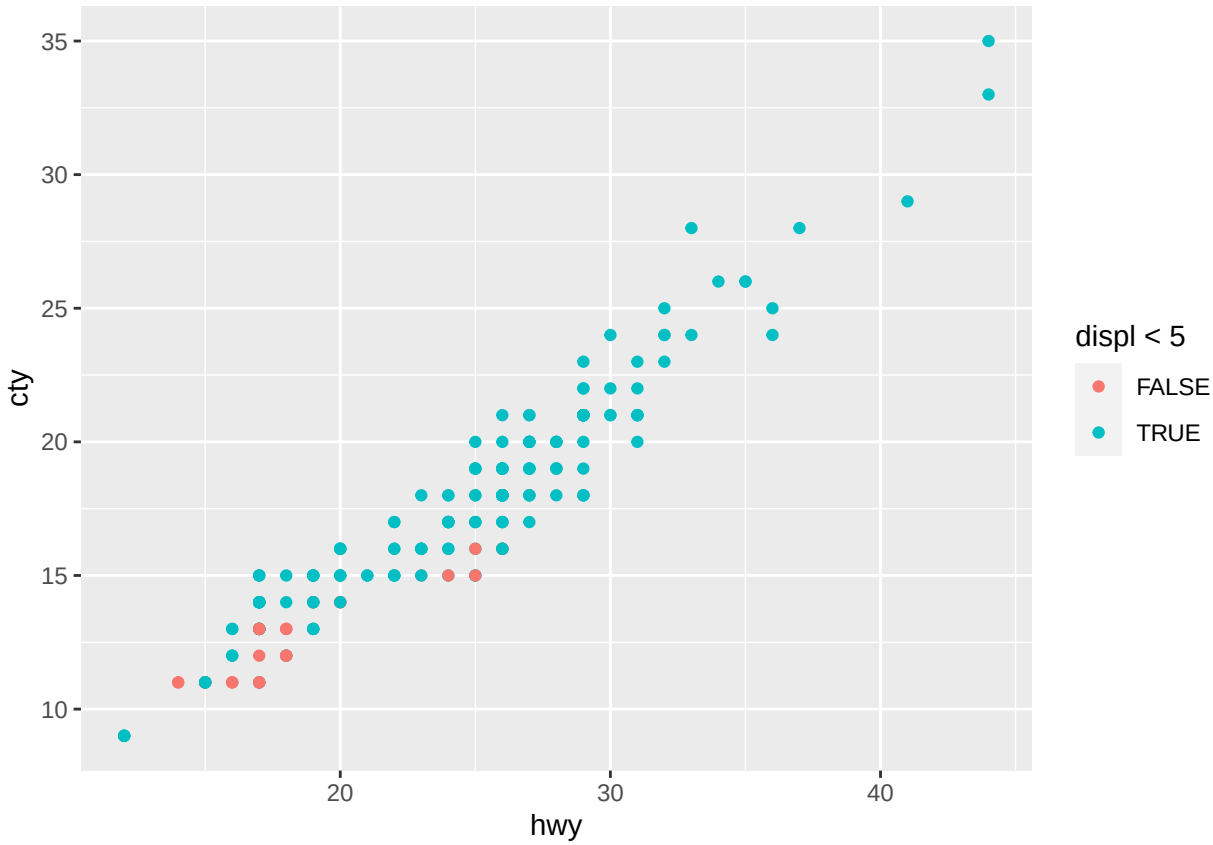
```
# Graph 2
ggplot(data = mpg) +
geom_point(mapping = aes(x = hwy, y = cty, color = as.character(year)))
```



The first graph uses year as a continuous variable, whereas the second graph uses year as a categorical variable. The second graph is easier to read because the colors contrast with one another and it is easier to see the difference between the two times.

3

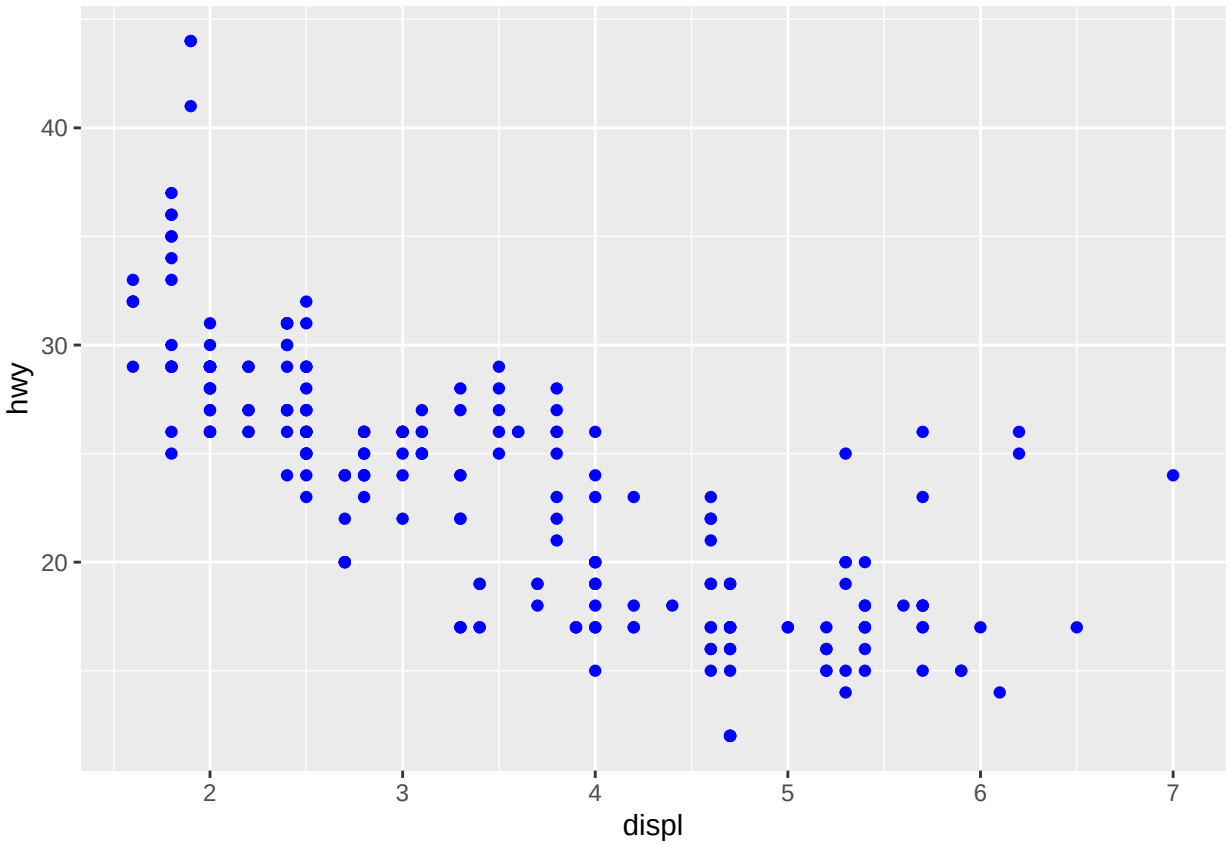
```
ggplot(data = mpg) +
geom_point(mapping = aes(x = hwy, y = cty, color = displ < 5))
```



In this scenario, all vehicles with displacement less than 5 are colored blue, whereas all vehicles with displacement not less than 5 are colored coral.

4

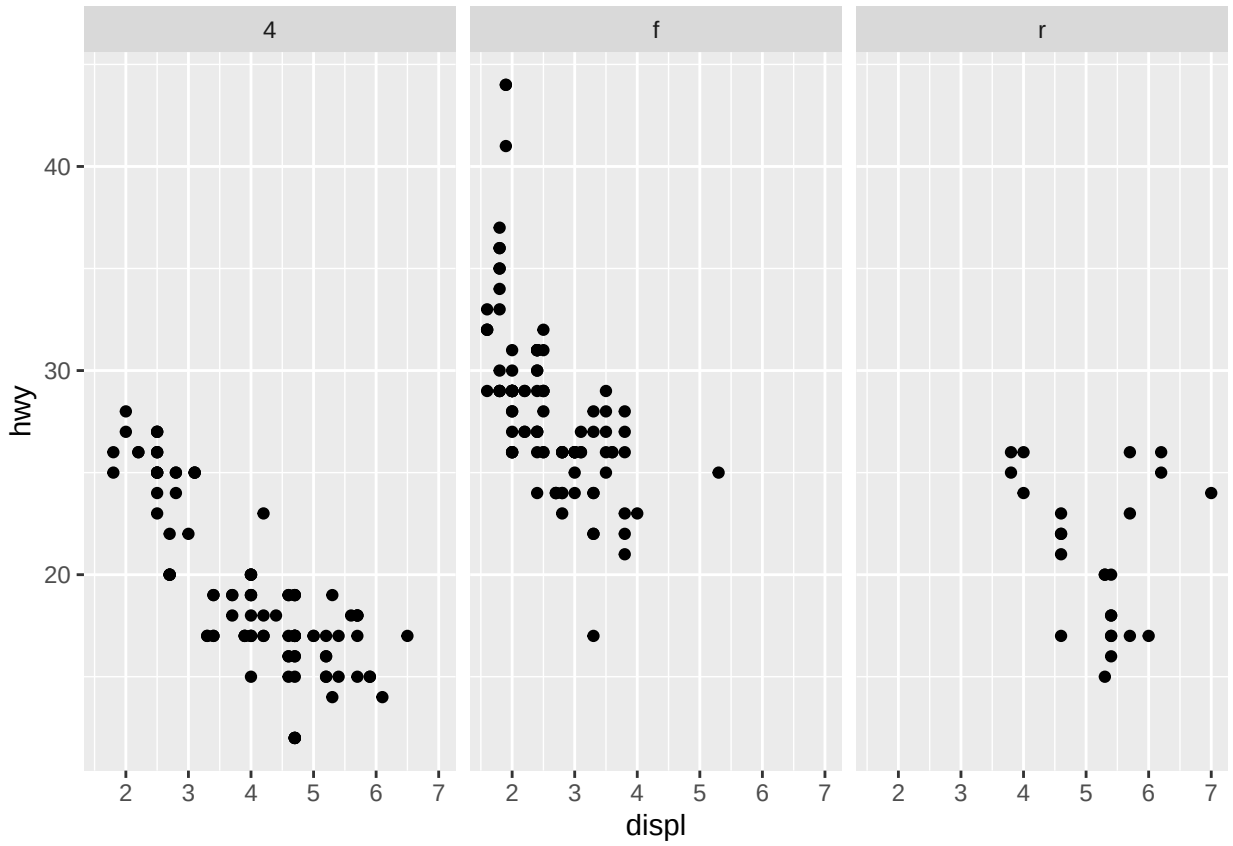
```
ggplot(data = mpg) +
geom_point(mapping = aes(x = displ, y = hwy), color = "blue")
```



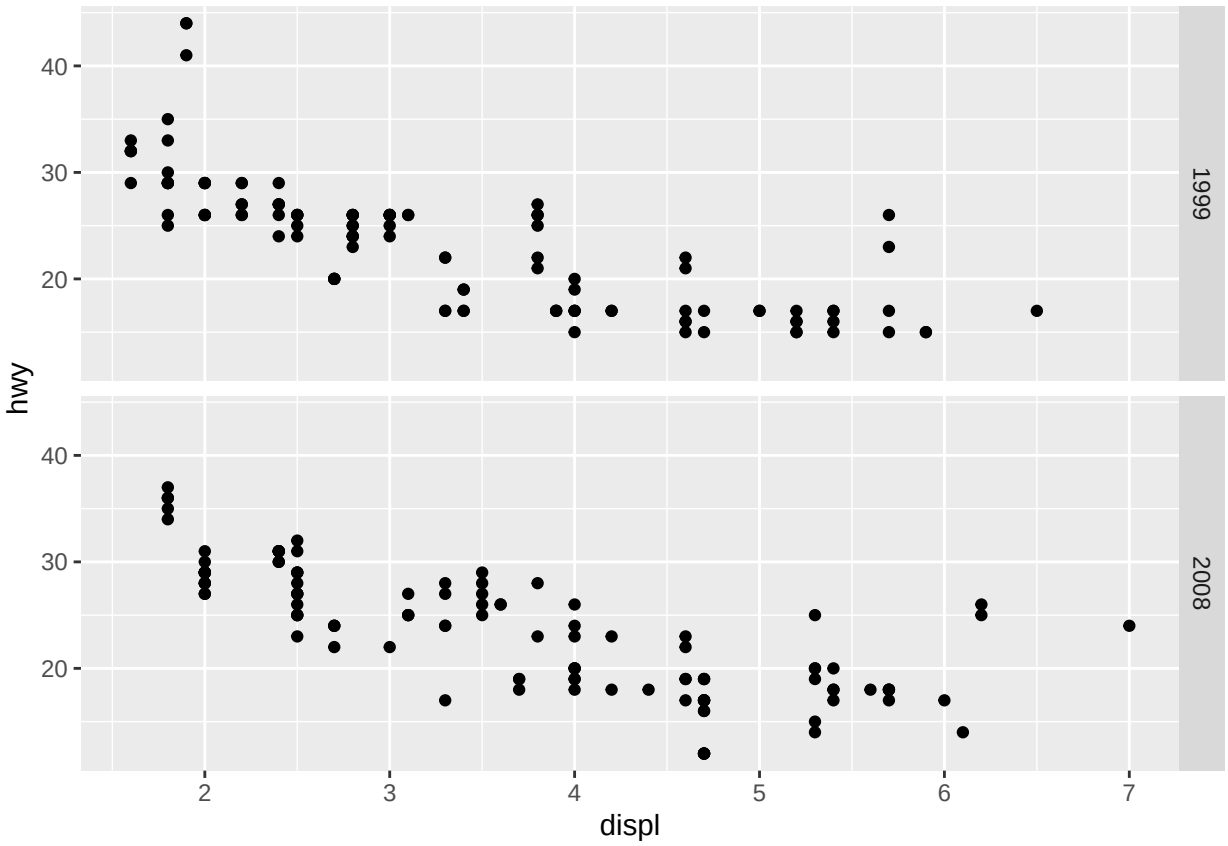
2.3: FACETS

1

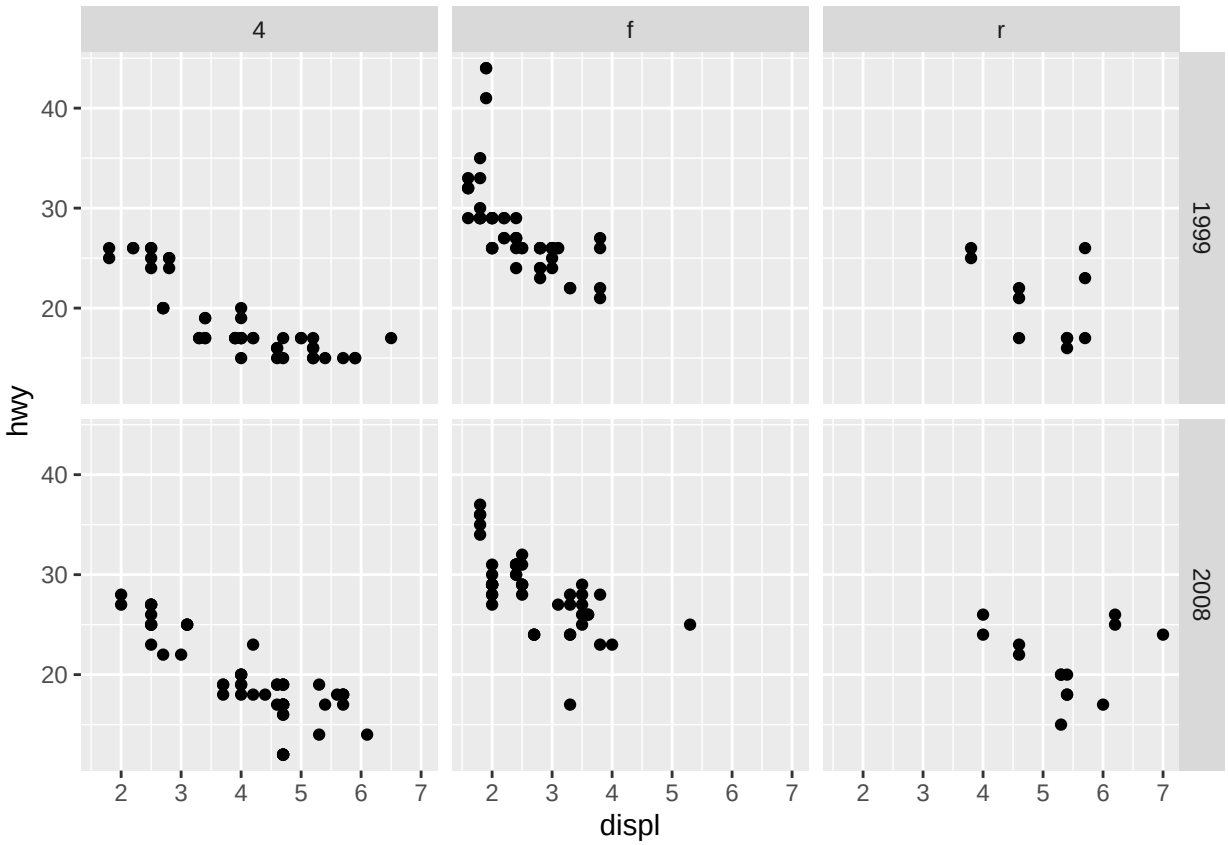
```
ggplot(data = mpg) +
geom_point(mapping = aes(x = displ, y = hwy)) +
facet_grid(cols = vars(drv))
```




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



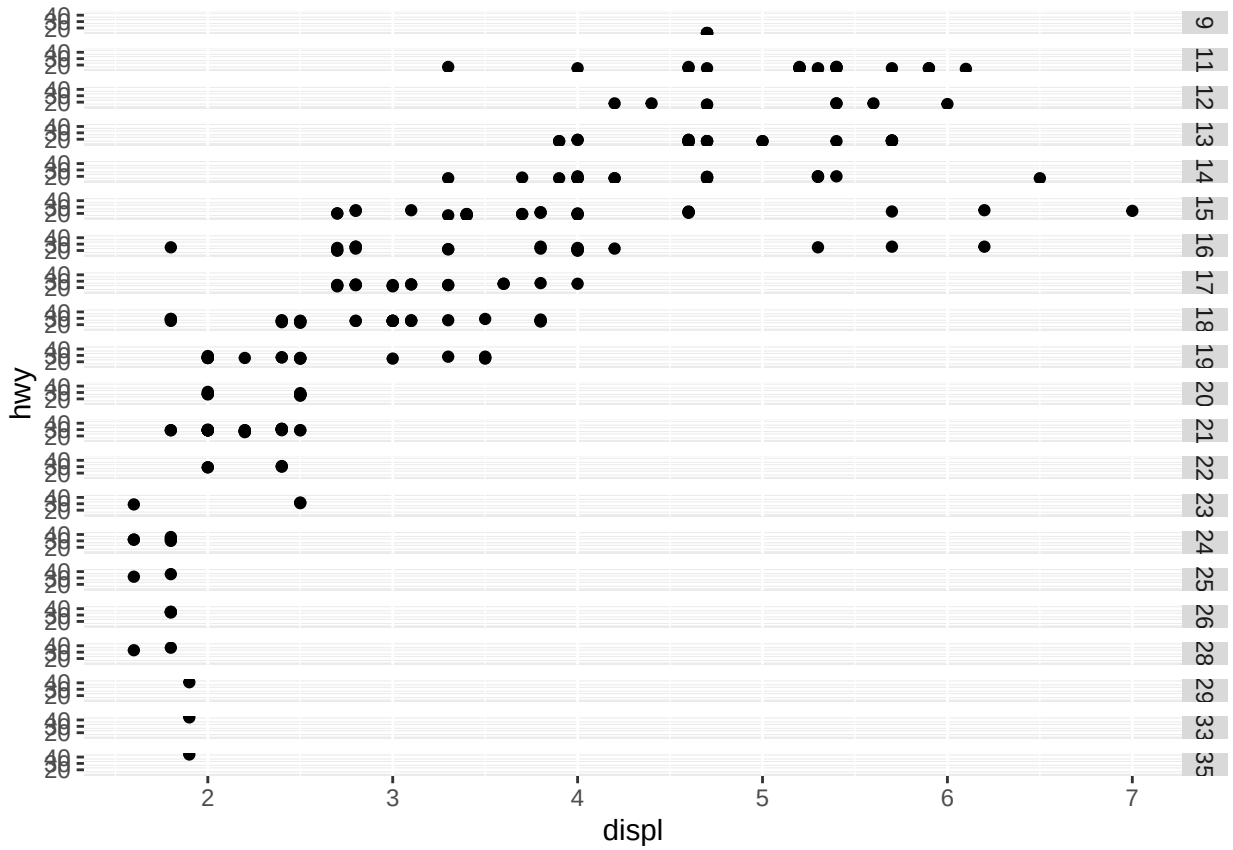

```
ggplot(data = mpg) +
geom_point(mapping = aes(x = displ, y = hwy)) +
facet_grid(rows = vars(year), cols = vars(drv))
```



The "facet-grid" code breaks down the data into smaller graphs based on a chosen variable. For example, the first graph set shows displacement compared to highway mileage broken down into three graphs based on drive type (4-wheel drive, front-wheel drive, and rear-wheel drive).

2

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

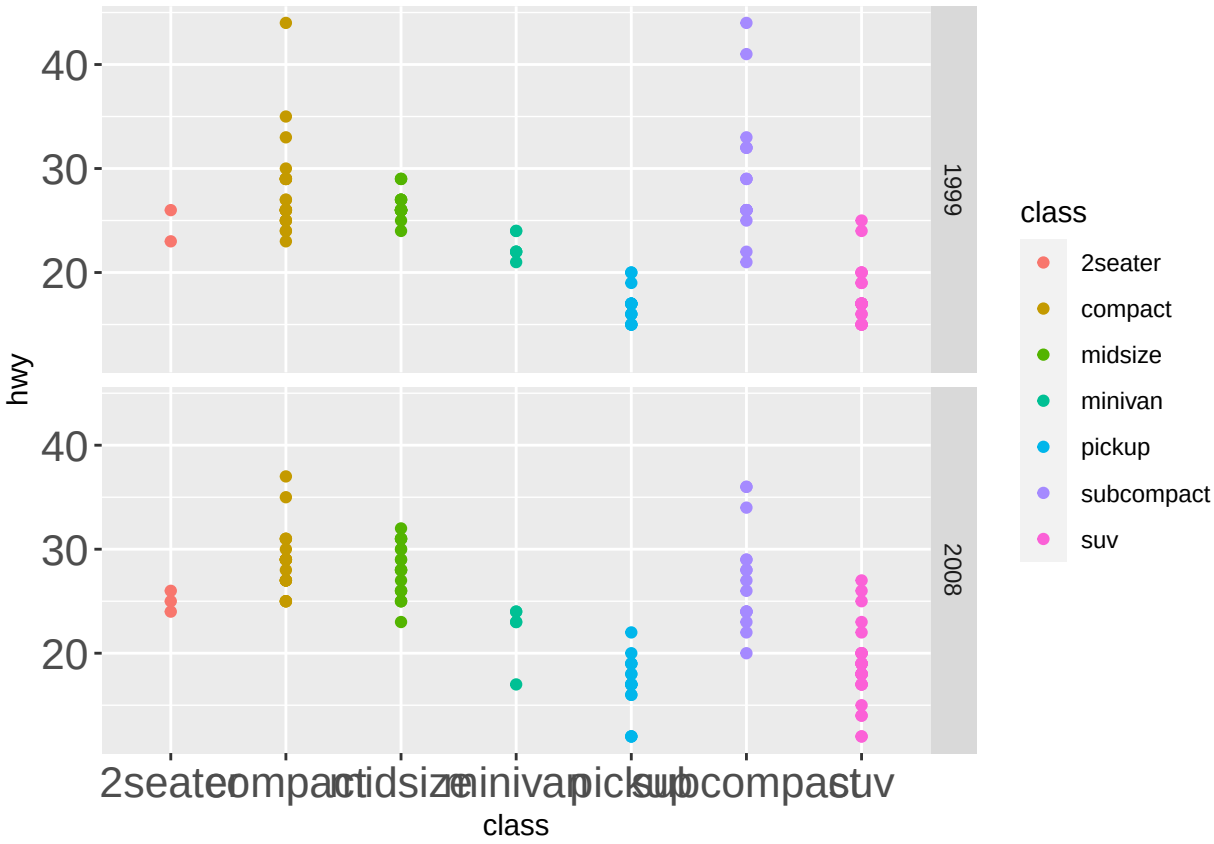


When using a continuous variable for "facet_grid," for example city mpg, the graph set will be broken down into so many individual graphs to a point where they are too small to read or interpret.

3

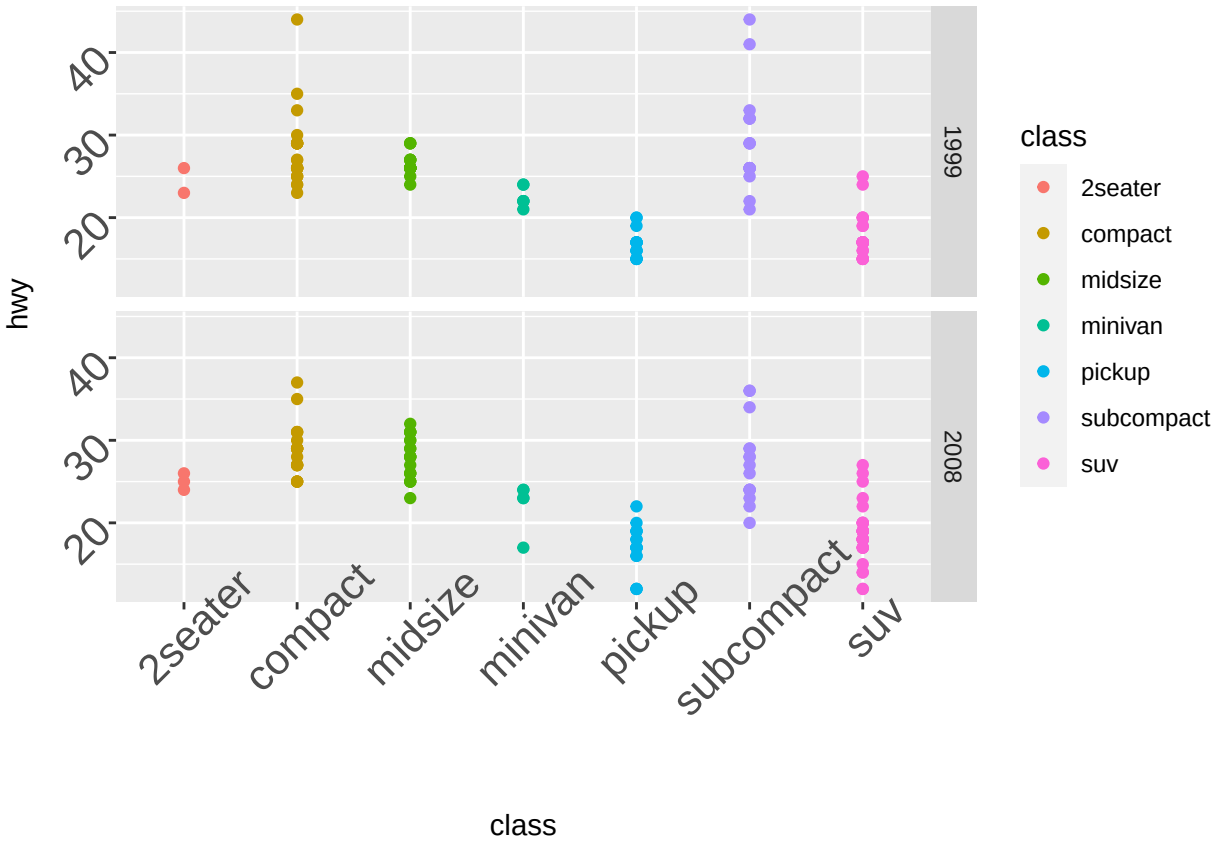
#Resource: https://stackoverflow.com/questions/1330989/rotating-and-spacing-axis-labels-in-ggplot2

```
ggplot(data = mpg) +
  geom_point (mapping = aes(x = class, y = hwy, color = class)) +
  facet_grid(rows = vars(year)) +
  theme(axis.text = element_text(size = 16))
```



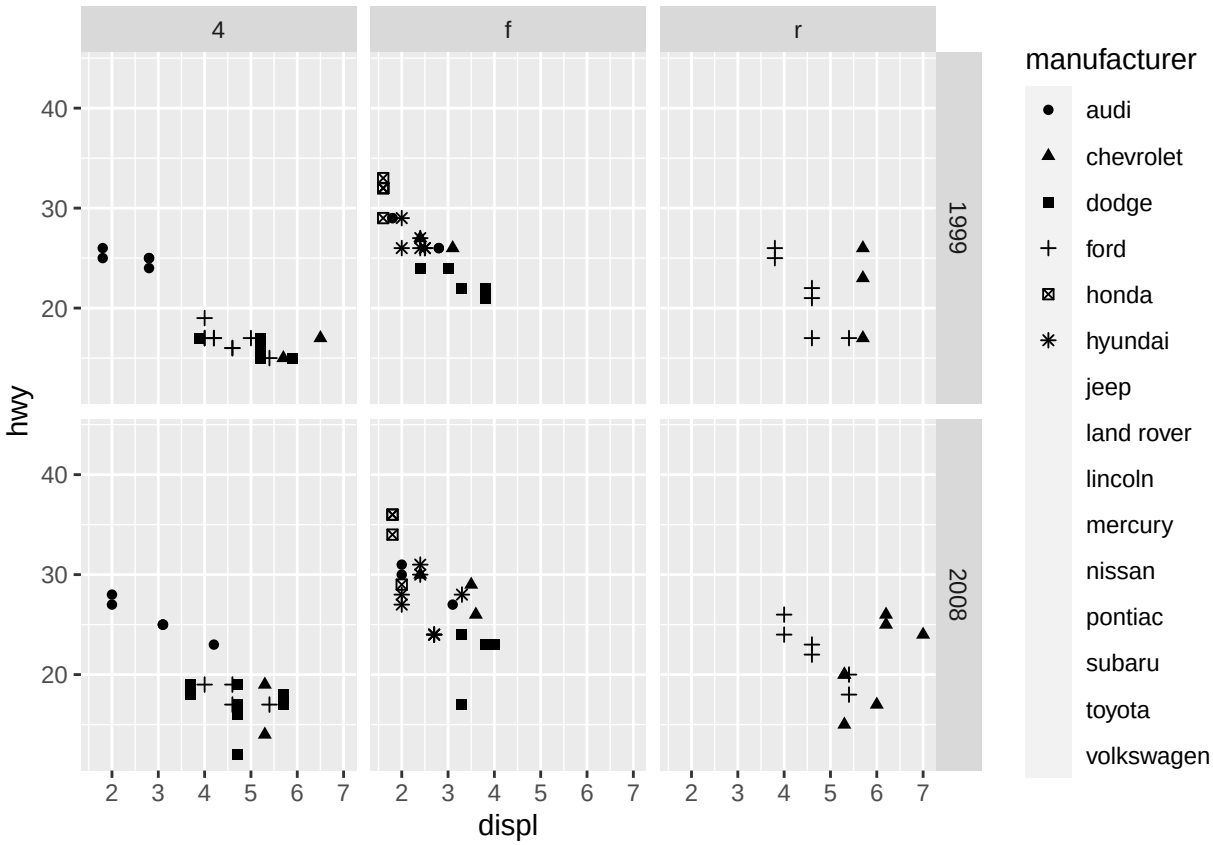
4

```
ggplot(data = mpg) +
  geom_point (mapping = aes(x = class, y = hwy, color = class)) +
  facet_grid(rows = vars(year)) +
  theme(axis.text = element_text(size = 16, angle = 45))
```



5

```
ggplot(data = mpg) +
  geom_point (mapping = aes(x = displ, y = hwy, shape = manufacturer)) +
  facet_grid(rows = vars(year), cols = vars(drv))
```



There are too many manufacturers and not enough shapes. According to the error warning, the shape palette can handle no more than 6 different values. In this dataset, there are over 100 manufacturers. As a result only the first six manufacturers in the list are assigned a shape and the rest are ommitted.

2.4 GRAMMAR OF GRAPHICS: GEOMS

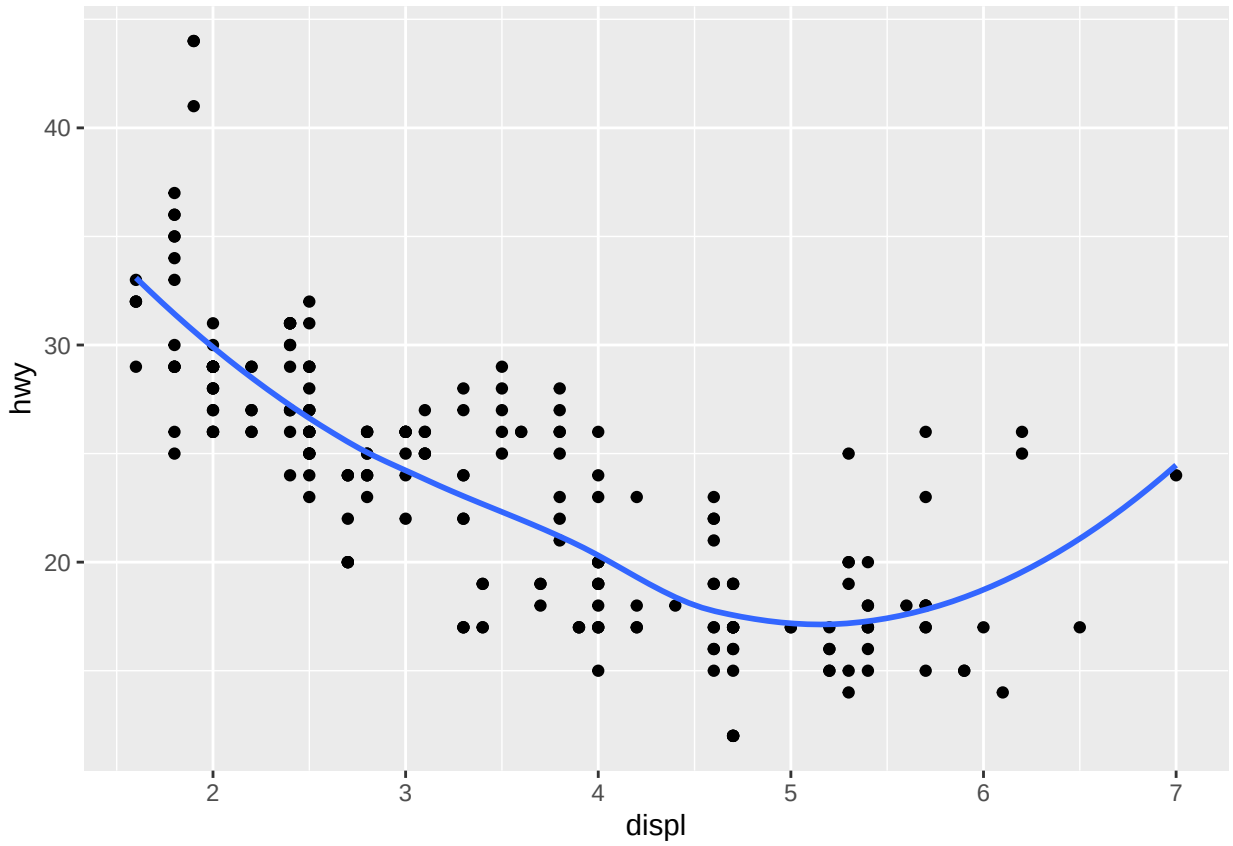
1

For a line graph, we can use geom_line. For a boxplot, we can use geom_boxplot. For a histogram, we can use geom_histogram. For an area chart, we can use geom_area.

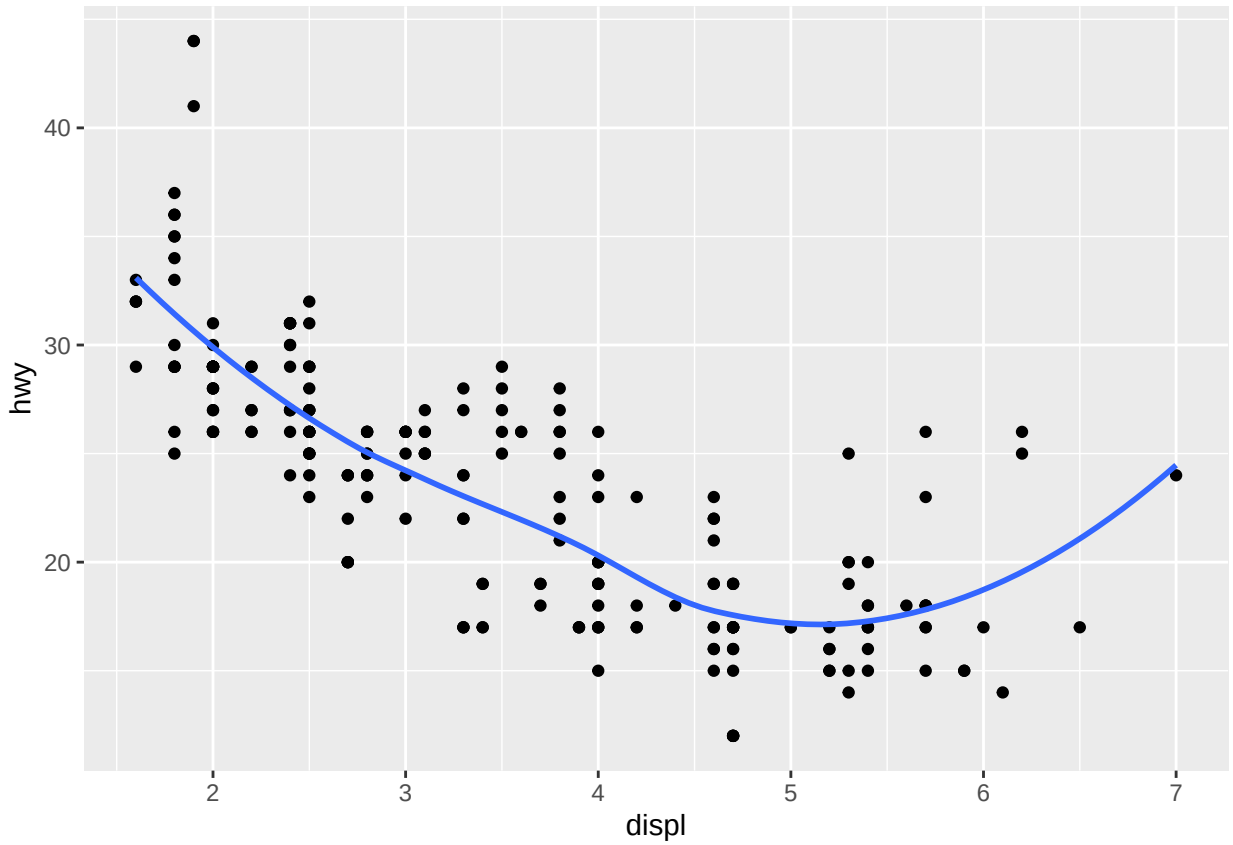
2

Yes, the two graphs will look alike because they are saying the same thing. The only difference is that the first graph uses a more condensed code set-up whereas the second graph is written out completely.

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

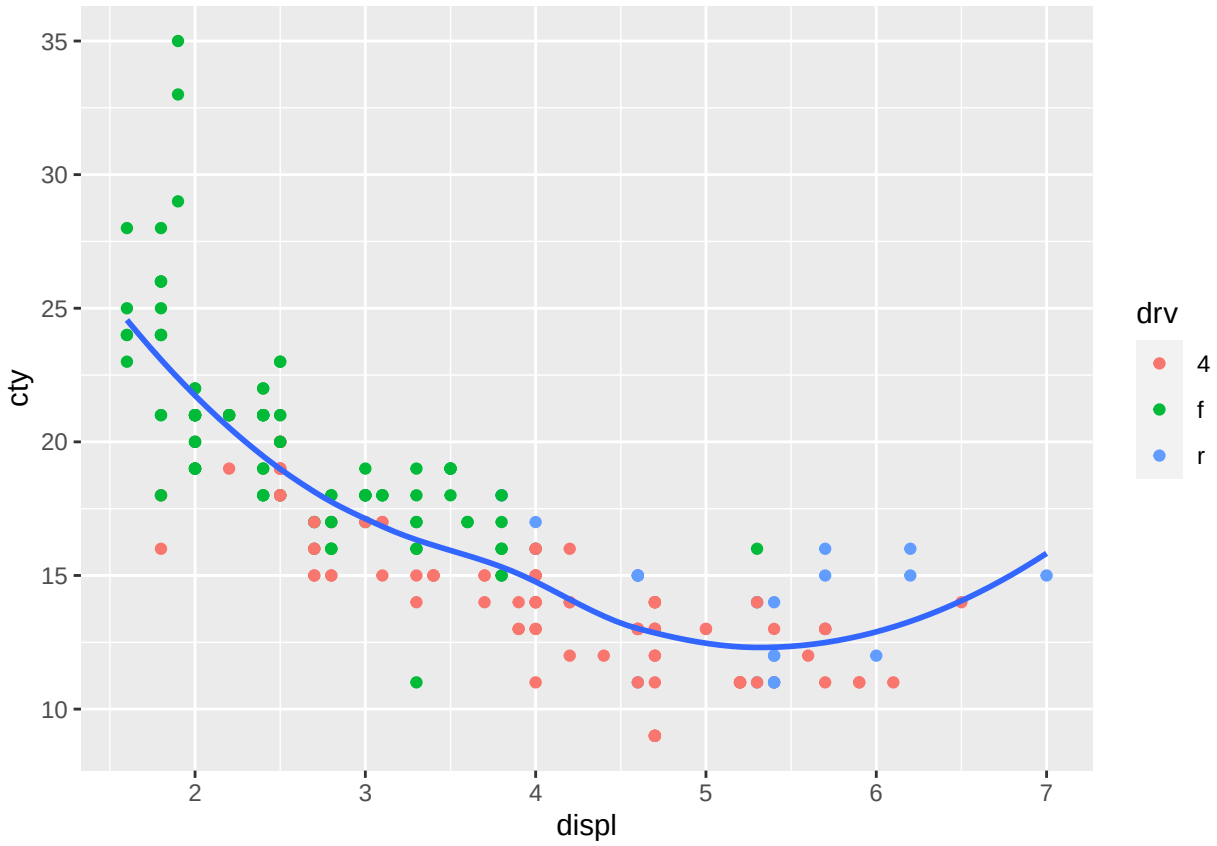



```
ggplot() +
geom_point(data = mpg, mapping = aes(x = displ, y = hwy)) +
geom_smooth(data = mpg, mapping = aes(x = displ, y = hwy), se = FALSE)
```



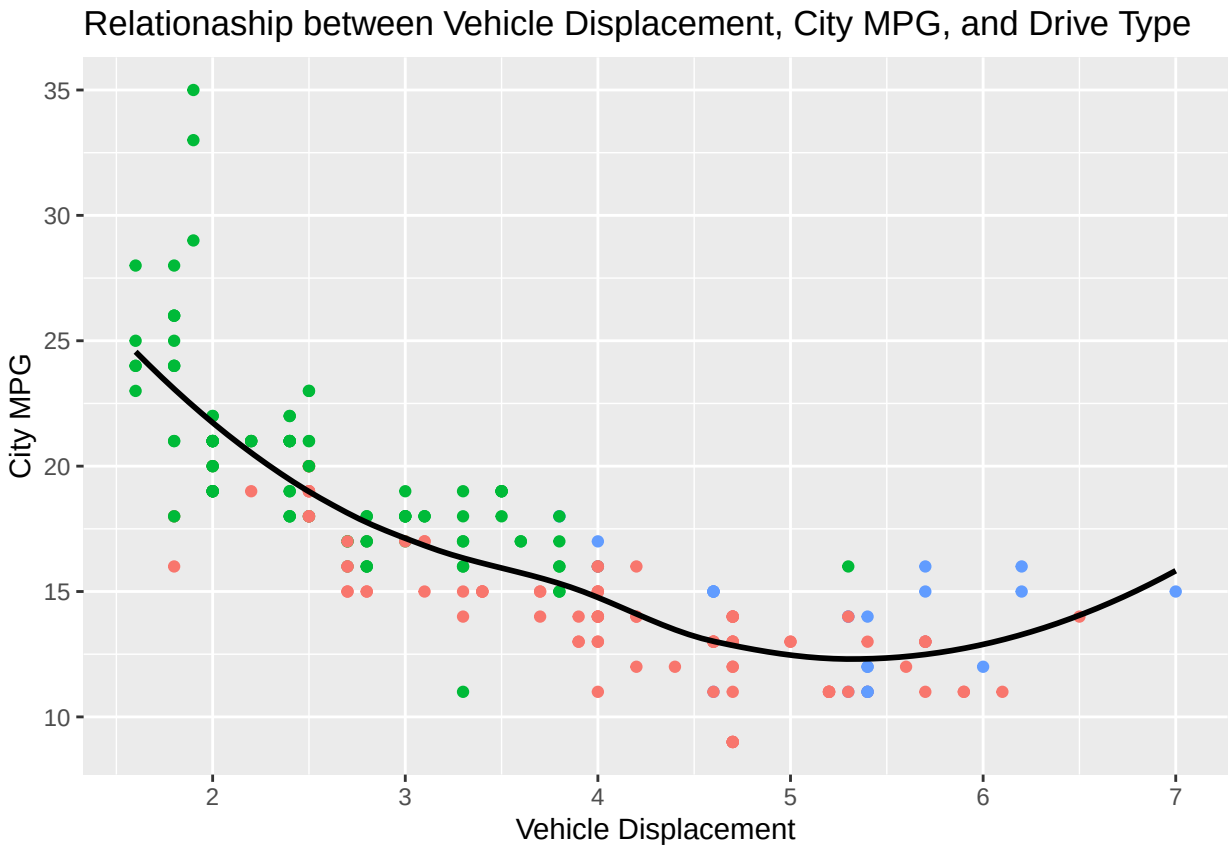
3

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



4

```
ggplot(data = mpg) +
  geom_point(mapping = aes (x = displ, y = cty, color = drv)) +
  #https://stackoverflow.com/questions/49358100/colour-black-in-geom-smooth-changes-lm-line-with-r-gglp
  geom_smooth(data = mpg,
              mapping = aes(x = displ, y = cty),color="black", se = FALSE) +
  xlab("Vehicle Displacement") +
  ylab("City MPG") +
  ggtitle("Relationaship between Vehicle Displacement, City MPG, and Drive Type") +
  #https://www.statology.org/remove-legend-ggplot2/#:~:text=By%20specifying%20legend.,all%20legends%20f
  theme(legend.position = "none")
```



Changing the Color of the line is an improvement because it no longer blends in with the dots. Adding the labels and title improve the overall readability of the graph. However, removing the key made the graph worse. Without it, the different colors of the points are meaningless.

2.4.1

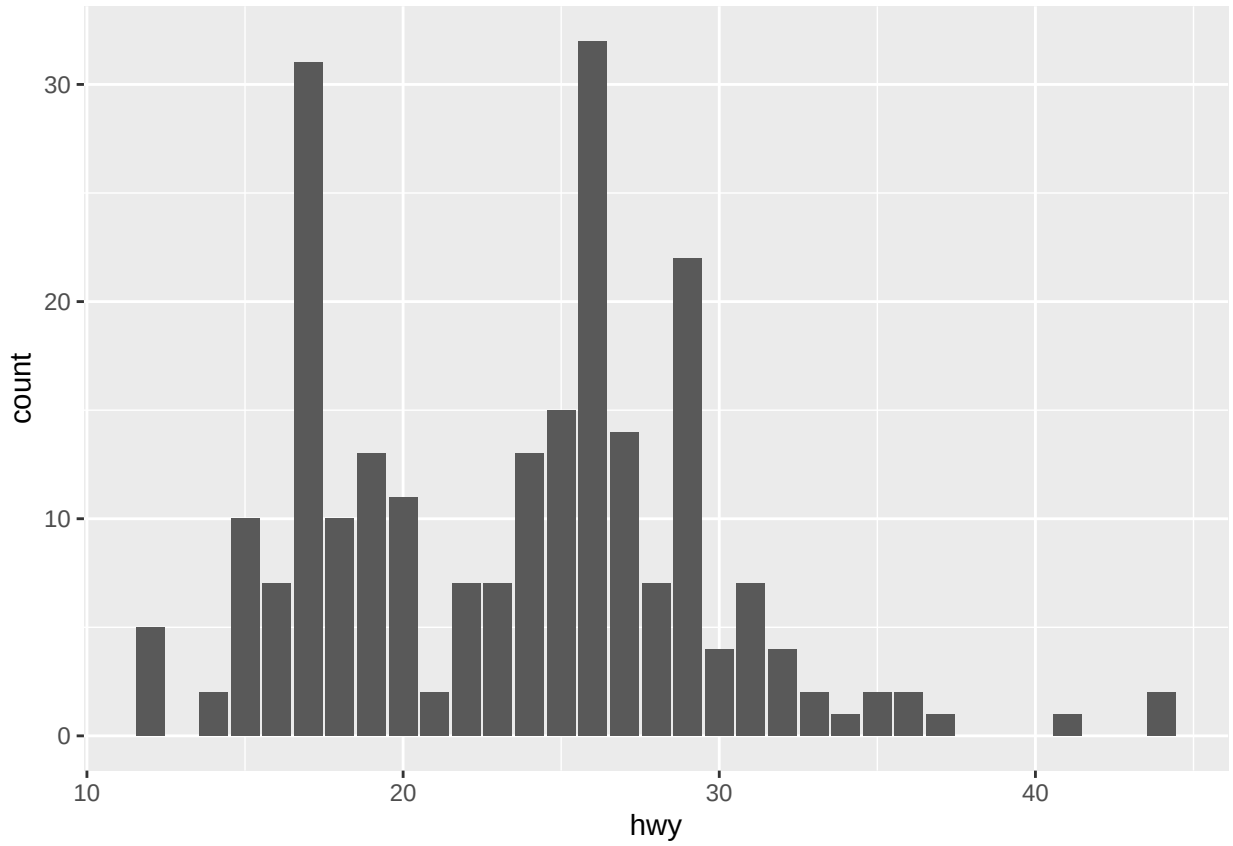
1

```
?geom_col
```

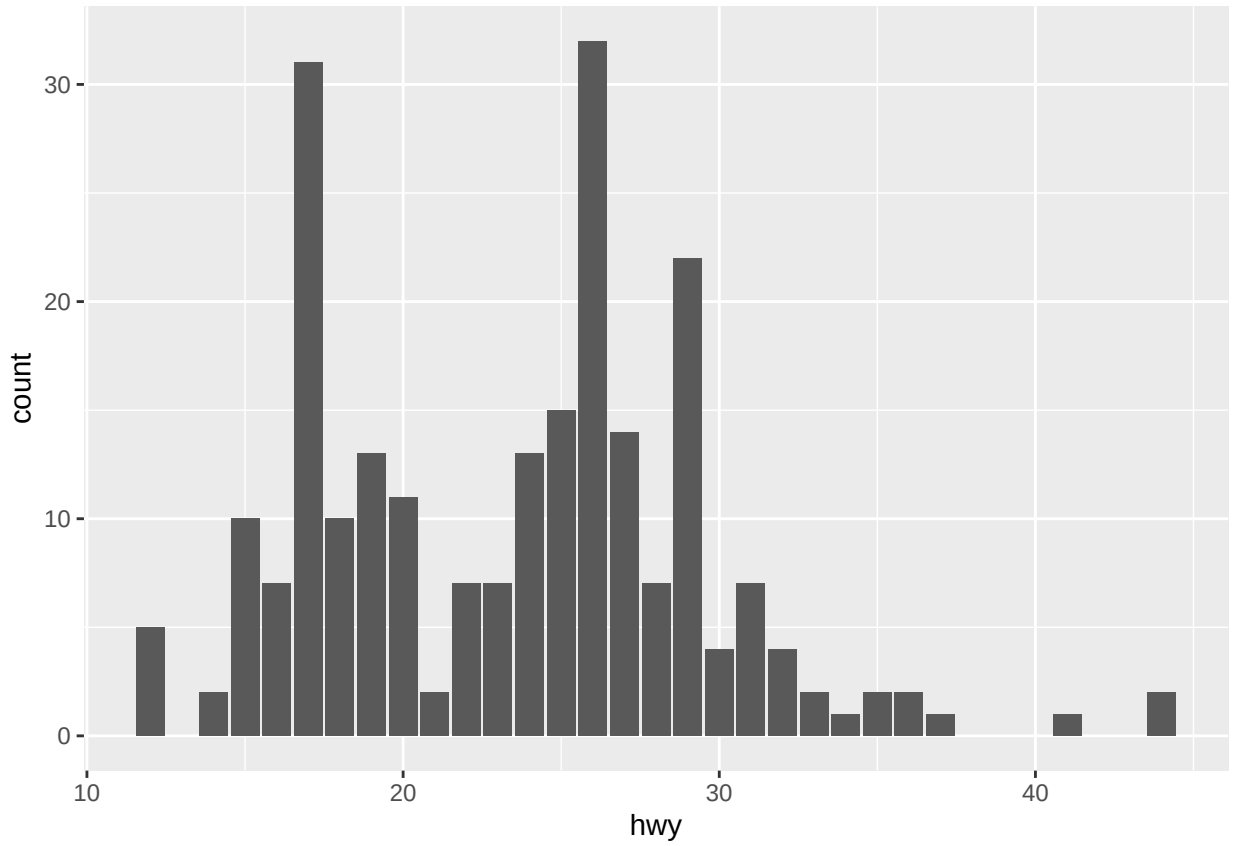
geom_col creates a bar chart of the data. Specifically, it shows columns keeping the

2

```
#original graph
ggplot(data=mpg, aes(x=hwy)) + geom_bar()
```




```
#geom replaced with stat
ggplot(data=mpg, aes(x=hwy)) + stat_count()
```

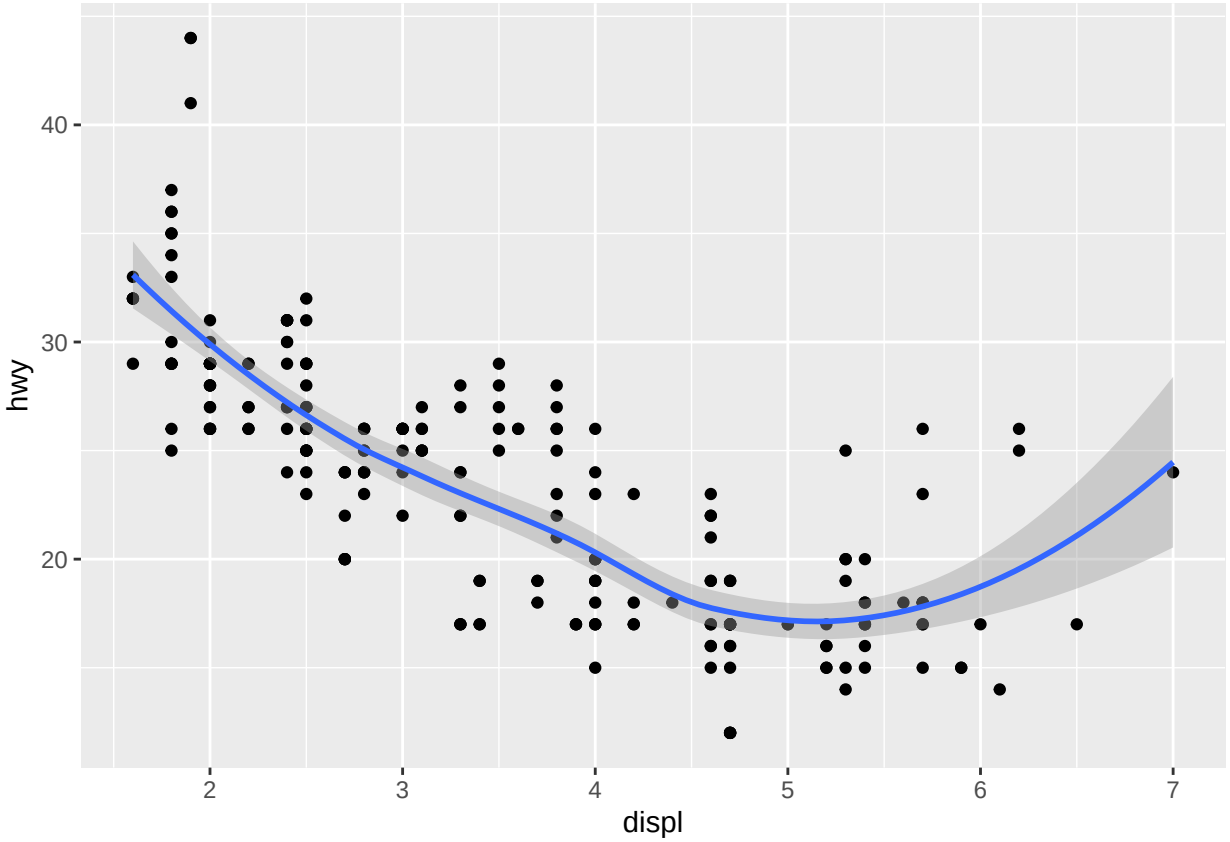


3

```
?stat_smooth
```

Stat_smooth computes the predicted value of y or x, ymin or xmin, ymax or xmax, and the standard error.

```
ggplot(mpg, aes(displ, hwy)) +
  geom_point() +
  geom_smooth()
```



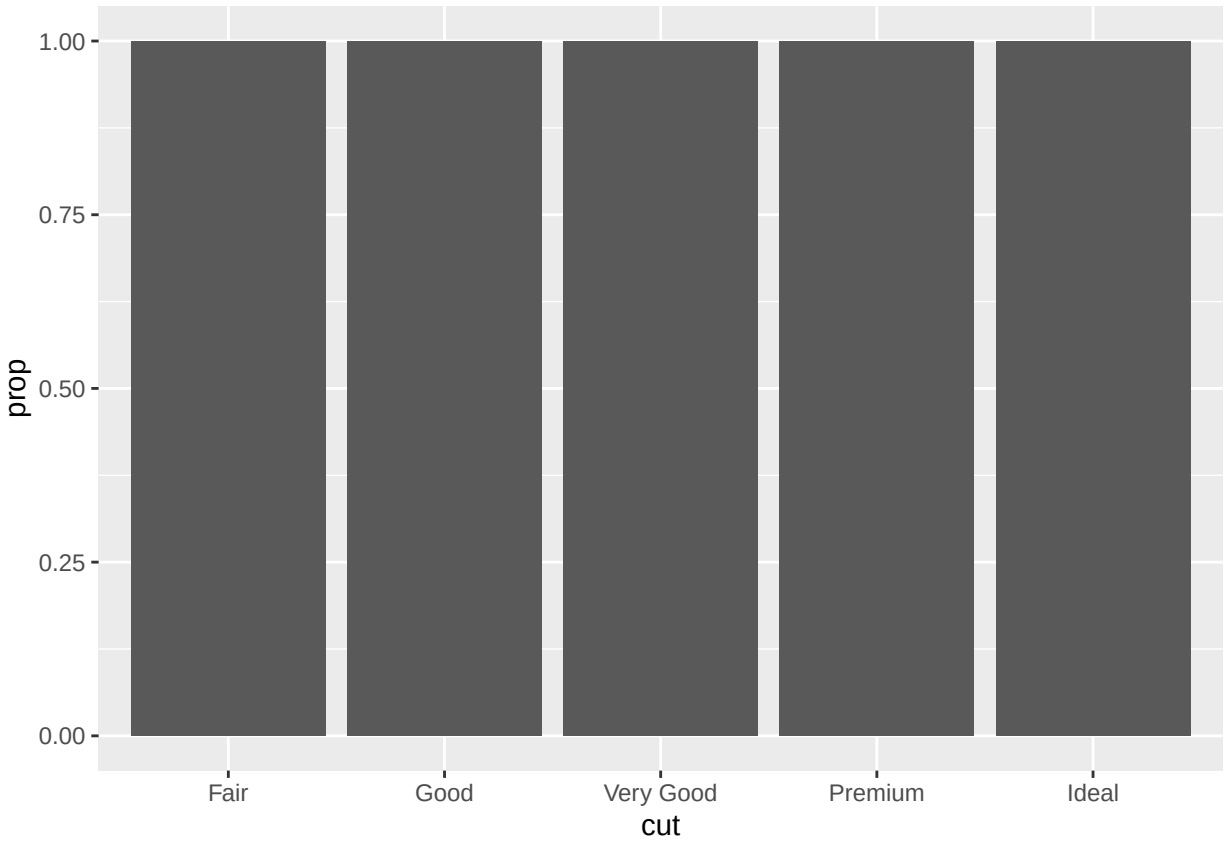
The computed variables are used to determine a confidence interval which is shown as the shaded area around the line on the graph.

The stat_smooth (aka geom_smooth) uses the variables from the x and y axes to determine orientation of the graph.

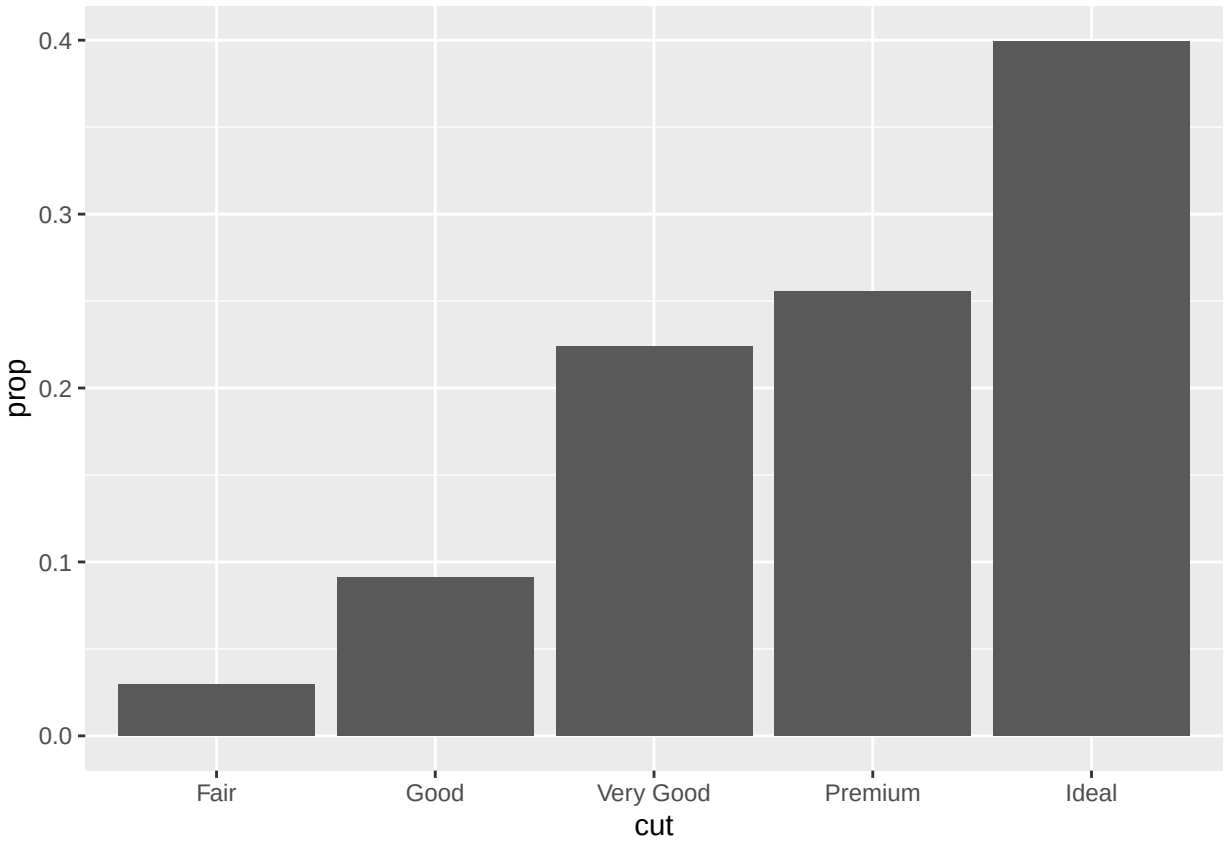
4

5

```
?geom_bar

#without group = 1
 ggplot(data = diamonds) +
  geom_bar(mapping = aes(x = cut, y = ..prop..))
```




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

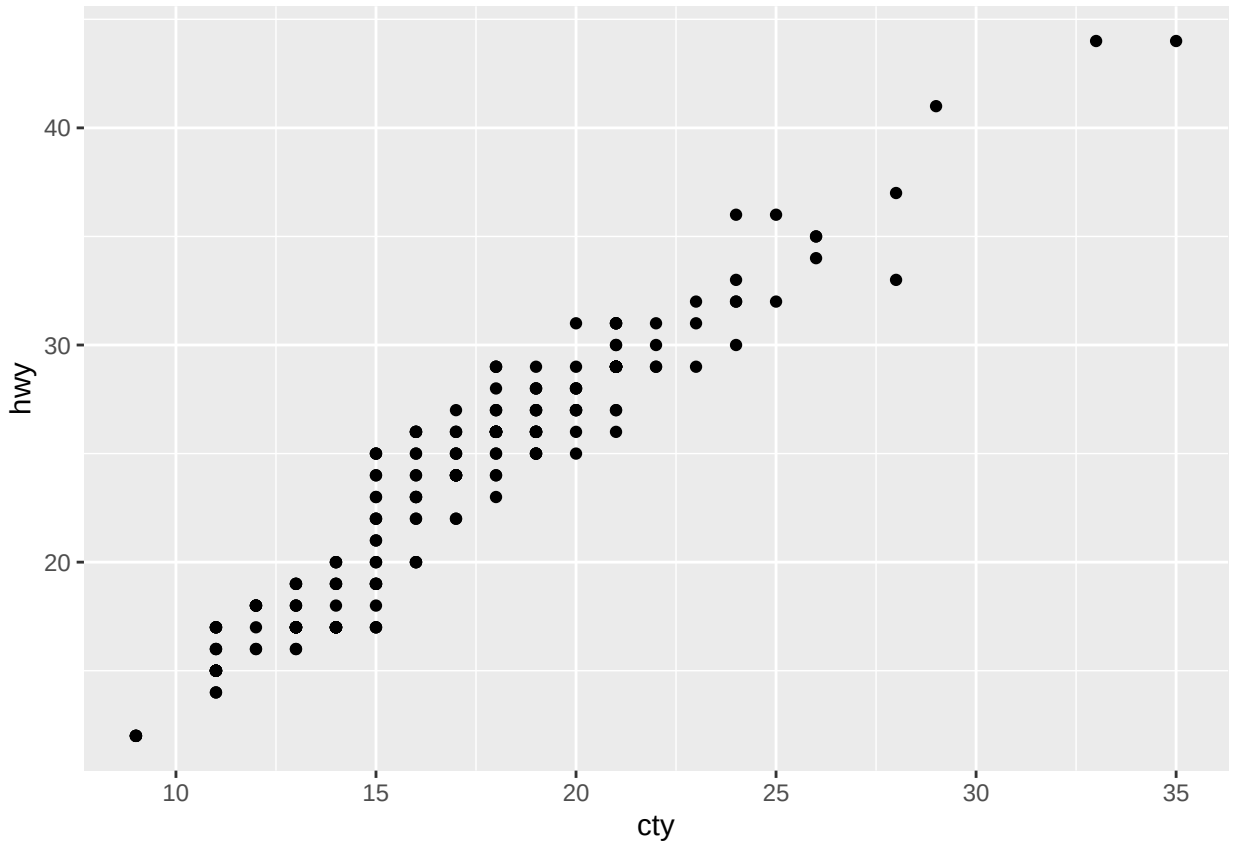


Failing to add "group = 1" creates a graph in which all bars are the same height. In that case, the proportion is determined by the proportion within the group (i.e. 100% of the Fair Diamonds are Fair). Adding "group = 1" looks at the proportion of the total data which fits the variable.

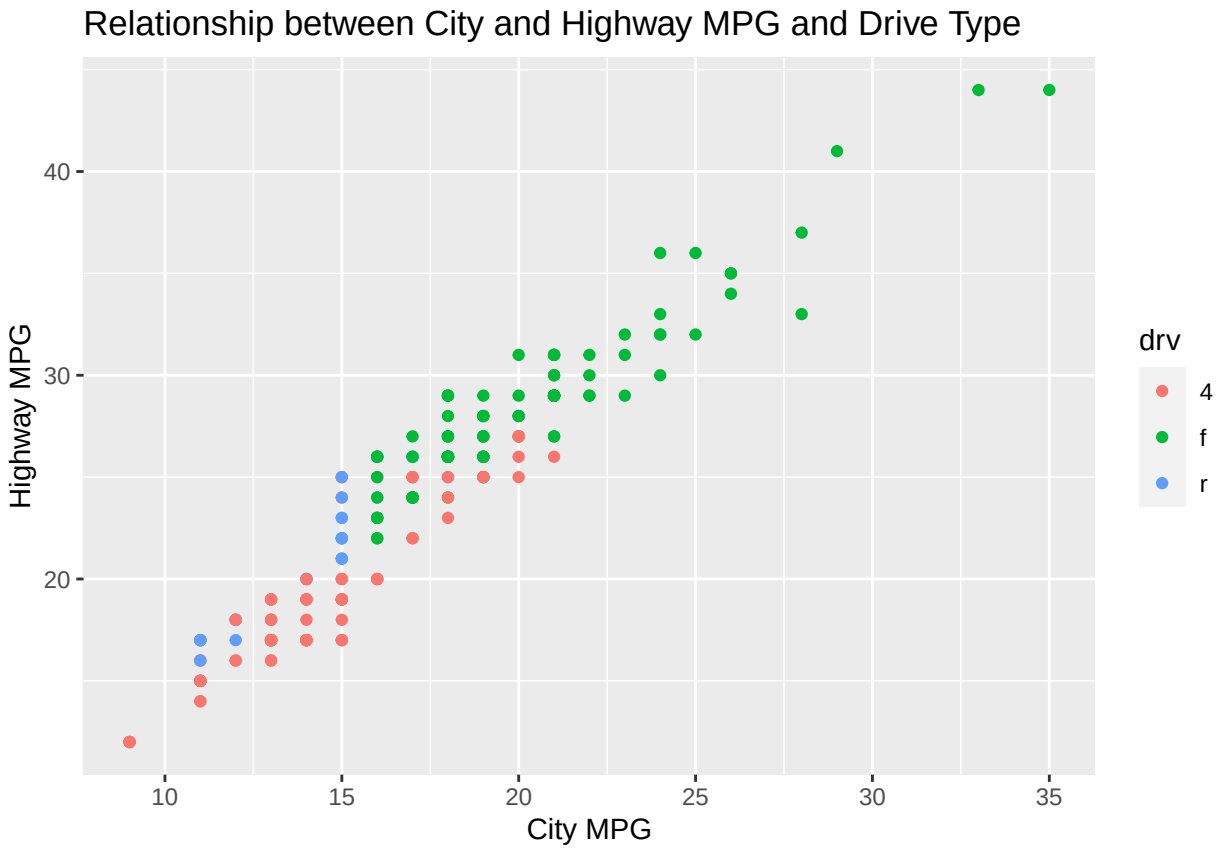
2.5: GRAMMAR OF GRAPHICS: POSITIONAL ADJUSTMENTS

1

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



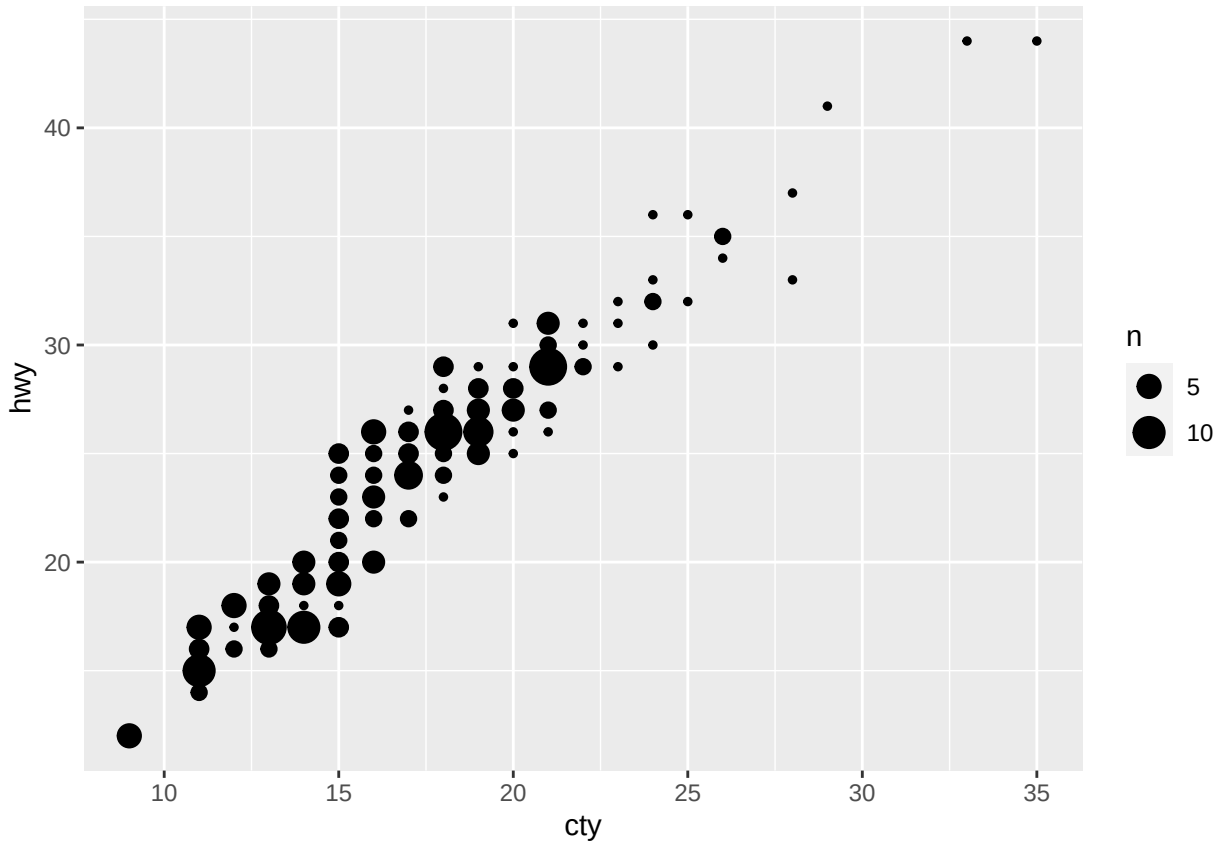

```
#Improved Graph
ggplot(data = mpg, mapping = aes(x = cty, y = hwy, color = drv)) +
  geom_point() +
  xlab("City MPG") +
  ylab("Highway MPG") +
  labs(title ="Relationship between City and Highway MPG and Drive Type")
```



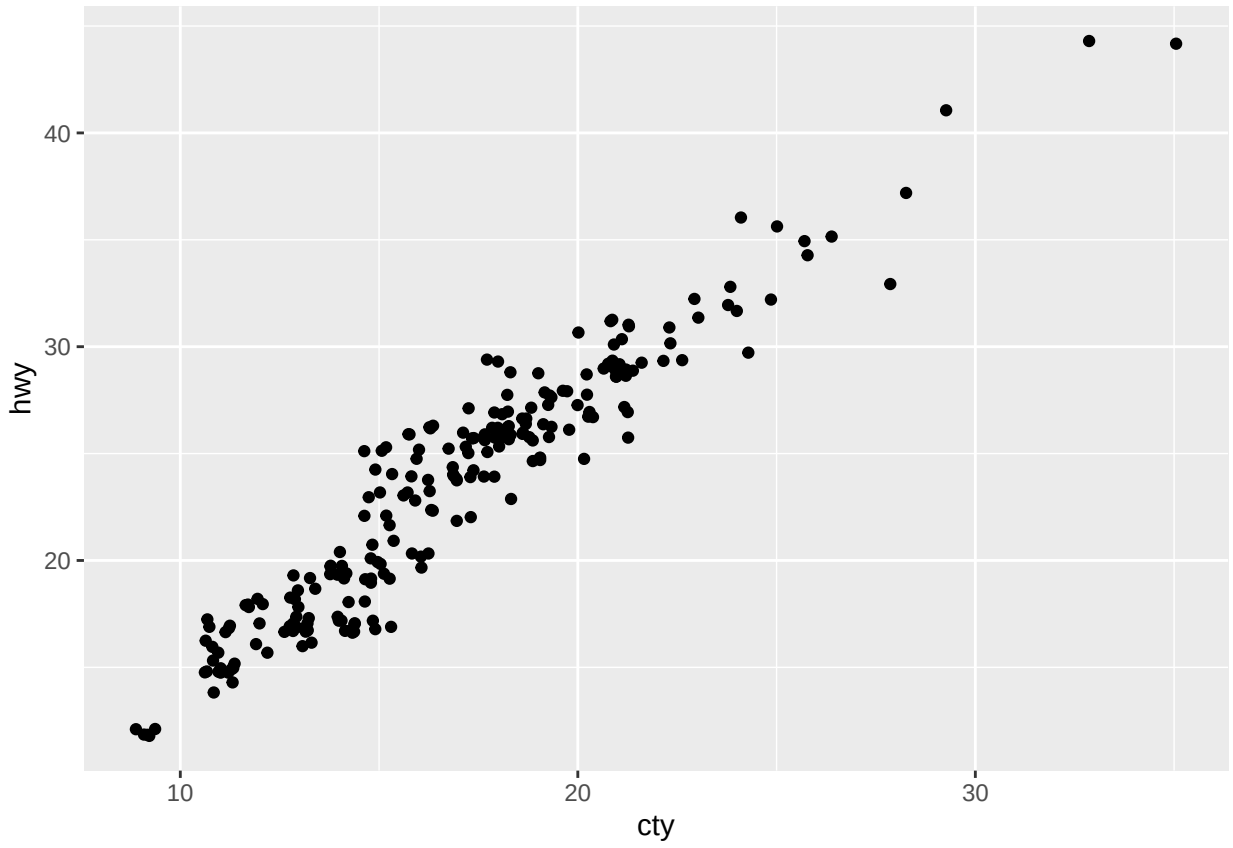
This original graph is missing labels and a title, so it is difficult to read. It shows a clear positive relationship between highway and city MPG, which is to be expected and doesn't provide much new/useful information. Color-coding the points adds some additional information as to the the types of cars with highest and lowest highway and city mpg.

2

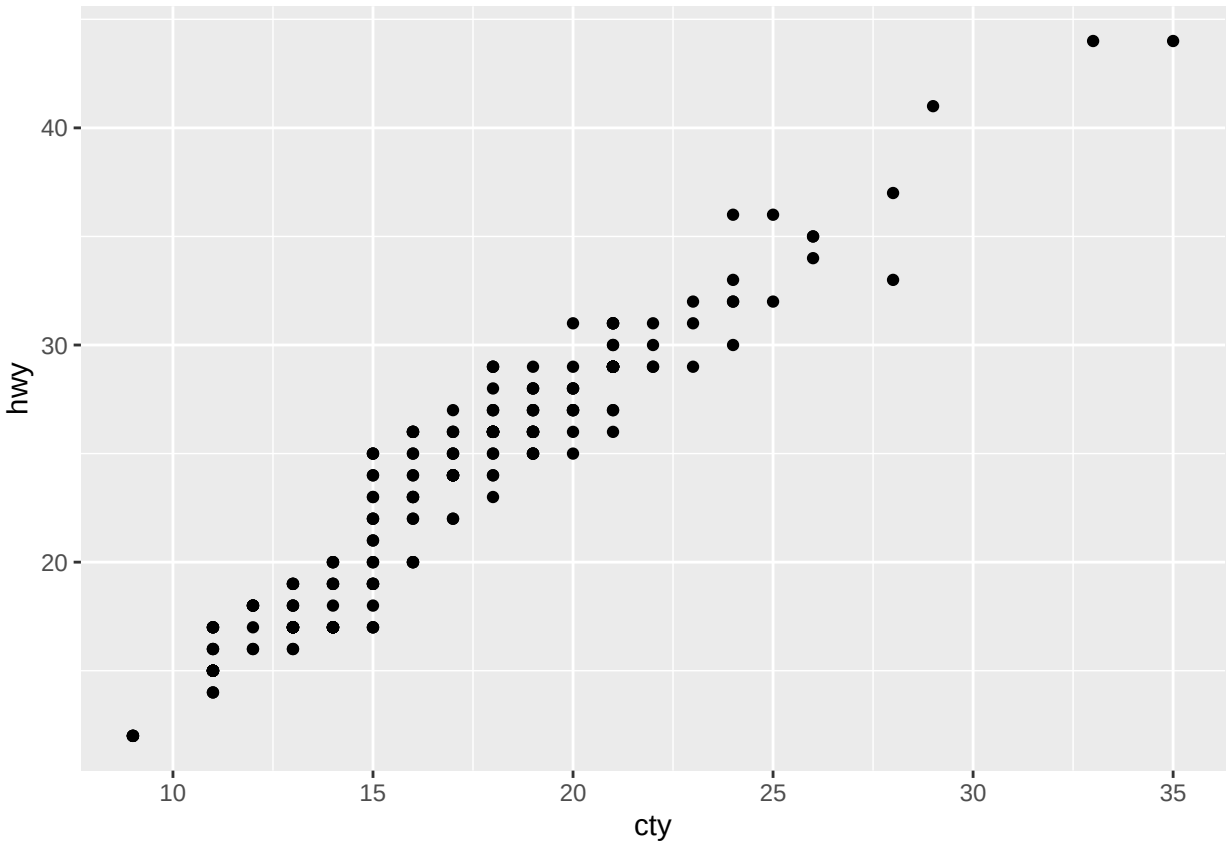
```
?geom_jitter
?geom_count

ggplot(mpg, aes(cty, hwy)) +
 geom_count()
```




```
ggplot(mpg, aes(cty, hwy)) +
 geom_jitter()
```




```
ggplot(mpg, aes(cty, hwy)) +
 geom_point()
```



When there are a number of points on a graph that would overlap, geom_jitter and geom_count modify the graph to show areas where points are more heavily concentrated. Geom_jitter creates a small amount of variation such that multiple points in the same area are visible. Geom_count creates points with different sizes to depict areas where multiple points overlap. Geom_jitter adjusts point positions while geom_count adjusts point size.

3

```
?geom_bar
```

By default, geom_bar grouped by fill() creates a stacked bar (each variable has one bar split into color-coded sections by proportions). In this question, the code "position ="dodge"" would be added to create multiple side-by-side bars as opposed to singular stacked bars.

2.6: GRAMMAR OF GRAPHICS: COORDINATE SYSTEMS

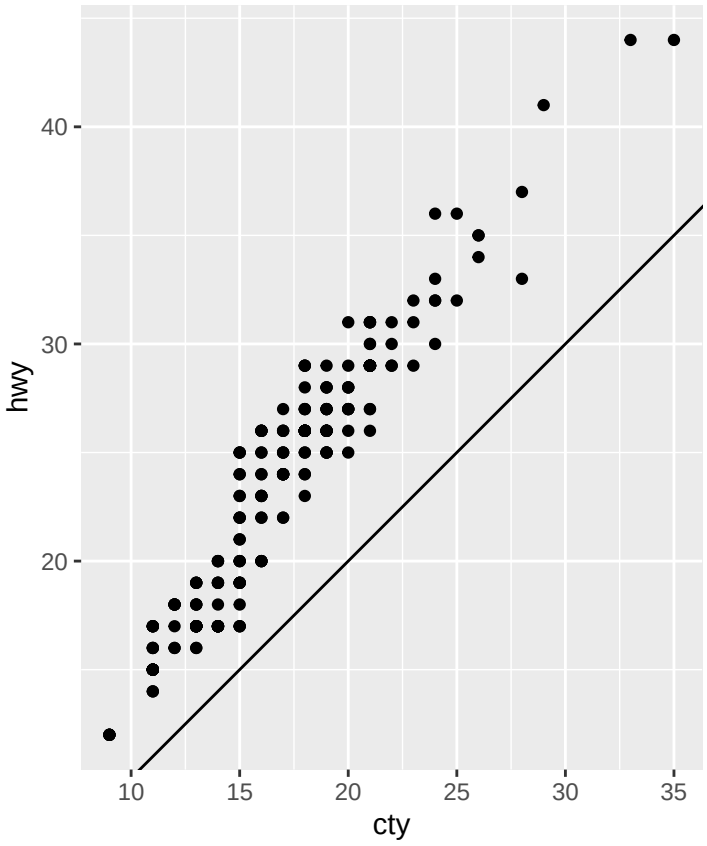
1

```
?coord_flip
```

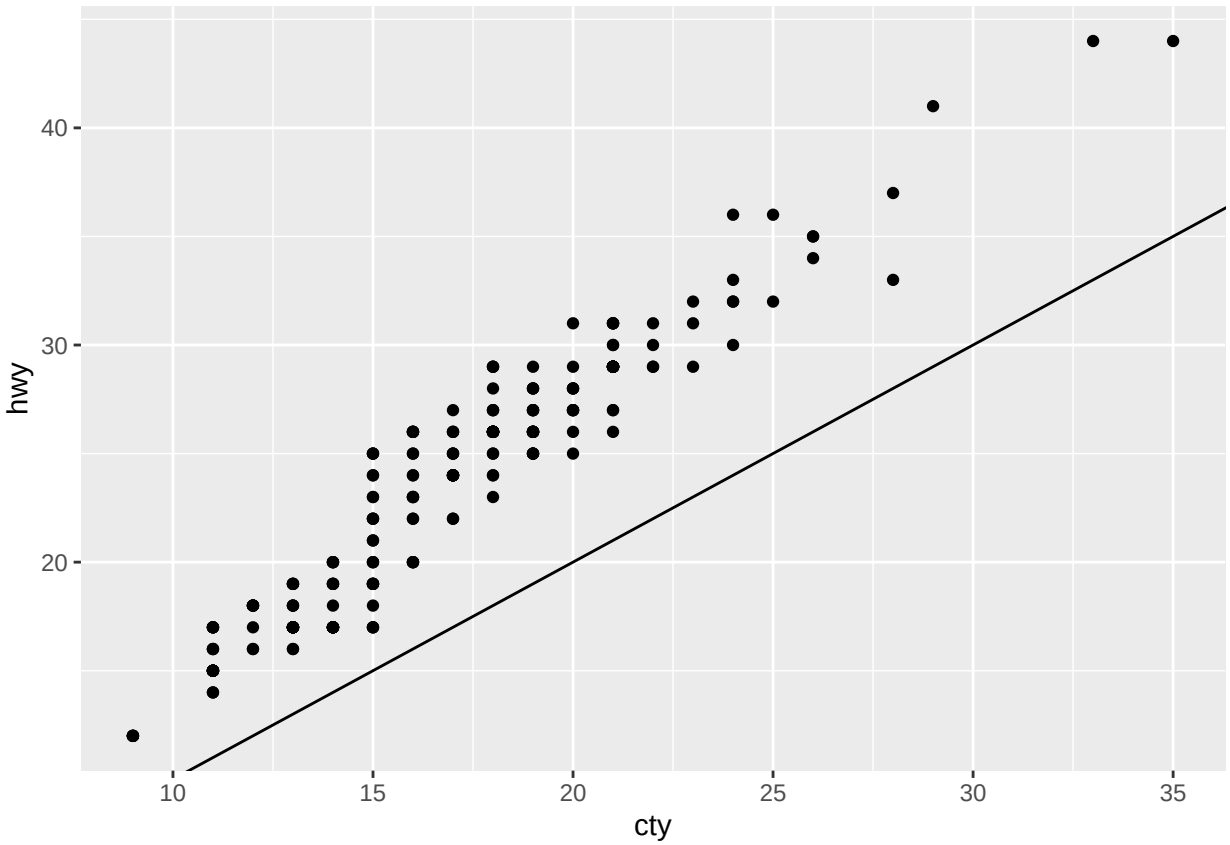
The function "coord_flip" flips the x and y axes of the graph.

#2

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




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



There is a positive, linear relationship between City MPG and Highway MPG. As one increases, so does the other. Geom_abline adds a reference line to the graph based on the data's slope and intercept. Its not a line a best fit. Coord_fixed adjusts the plot ratio. By default, the ratio is equal to one. In this case, it is easier to see the entire relationship between City and Highway MPG. CTY increases by units of five, and hwy increases by units of 10. Coord_fixed adjusts the plot to visually depict those units.