RWorskheet_TAN#6

John Kenneth D. Tan

2022-11-25

Worksheet #6

```
library(ggplot2)
library(dplyr)
```

```
##
## Attaching package: 'dplyr'
```

```
## The following objects are masked from 'package:stats':
##
##     filter, lag
```

```
## The following objects are masked from 'package:base':
##
##     intersect, setdiff, setequal, union
```

```
data(mpg)
datampg <- 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, 2.~
## $ year         <int> 1999, 1999, 2008, 2008, 1999, 1999, 2008, 1999, 1999, 200~
## $ cyl          <int> 4, 4, 4, 4, 6, 6, 6, 4, 4, 4, 4, 6, 6, 6, 6, 6, 6, 8, 8, ~
## $ trans        <chr> "auto(l5)", "manual(m5)", "manual(m6)", "auto(av)", "auto~
## $ drv          <chr> "f", "f", "f", "f", "f", "f", "f", "4", "4", "4", "4", "4~
## $ cty          <int> 18, 21, 20, 21, 16, 18, 18, 18, 16, 20, 19, 15, 17, 17, 1~
## $ hwy          <int> 29, 29, 31, 30, 26, 26, 27, 26, 25, 28, 27, 25, 25, 25, 2~
## $ fl           <chr> "p", "p", "p", "p", "p", "p", "p", "p", "p", "p", "p", "p~
## $ class        <chr> "compact", "compact", "compact", "compact", "compact", "c~
```

1. How many columns are in mpg dataset? How about the number of rows? Show the codes and its result.

Number of columns in mpg data set

```
ncol(mpg)
```

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

Number of rows in mpg data set

```
nrow(mpg)
```

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

2. Which manufacturer has the most models in this data set? Which model has the most variations?

```
manufac<-mpg%>%
group_by(manufacturer)%>%count()
manufac
```

```
## # A tibble: 15 x 2
## # Groups:   manufacturer [15]
##    manufacturer     n
##    <chr>        <int>
##  1 audi            18
##  2 chevrolet       19
##  3 dodge           37
##  4 ford            25
##  5 honda            9
##  6 hyundai         14
##  7 jeep             8
##  8 land rover       4
##  9 lincoln          3
## 10 mercury          4
## 11 nissan          13
## 12 pontiac          5
## 13 subaru          14
## 14 toyota          34
## 15 volkswagen      27
```

```
colnames(manufac)<-c("Manufacturer", "Counts")
```

#Answer: Dodge is the manufacturer that has the most models, for it has 37 models.

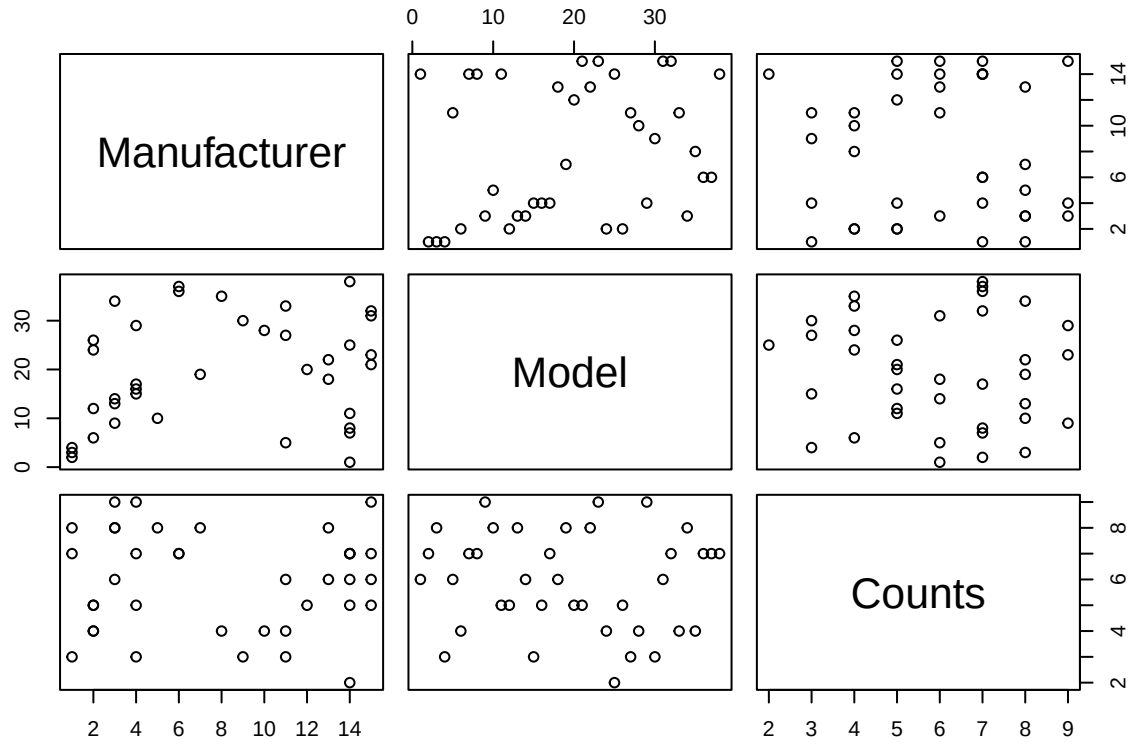
a. Group the manufacturers and find the unique models. Copy the codes and result.

```
unique_data <- mpg %>% group_by(manufacturer, model) %>%
distinct() %>% count()
colnames(unique_data) <- c("Manufacturer", "Model","Counts")
unique_data
```

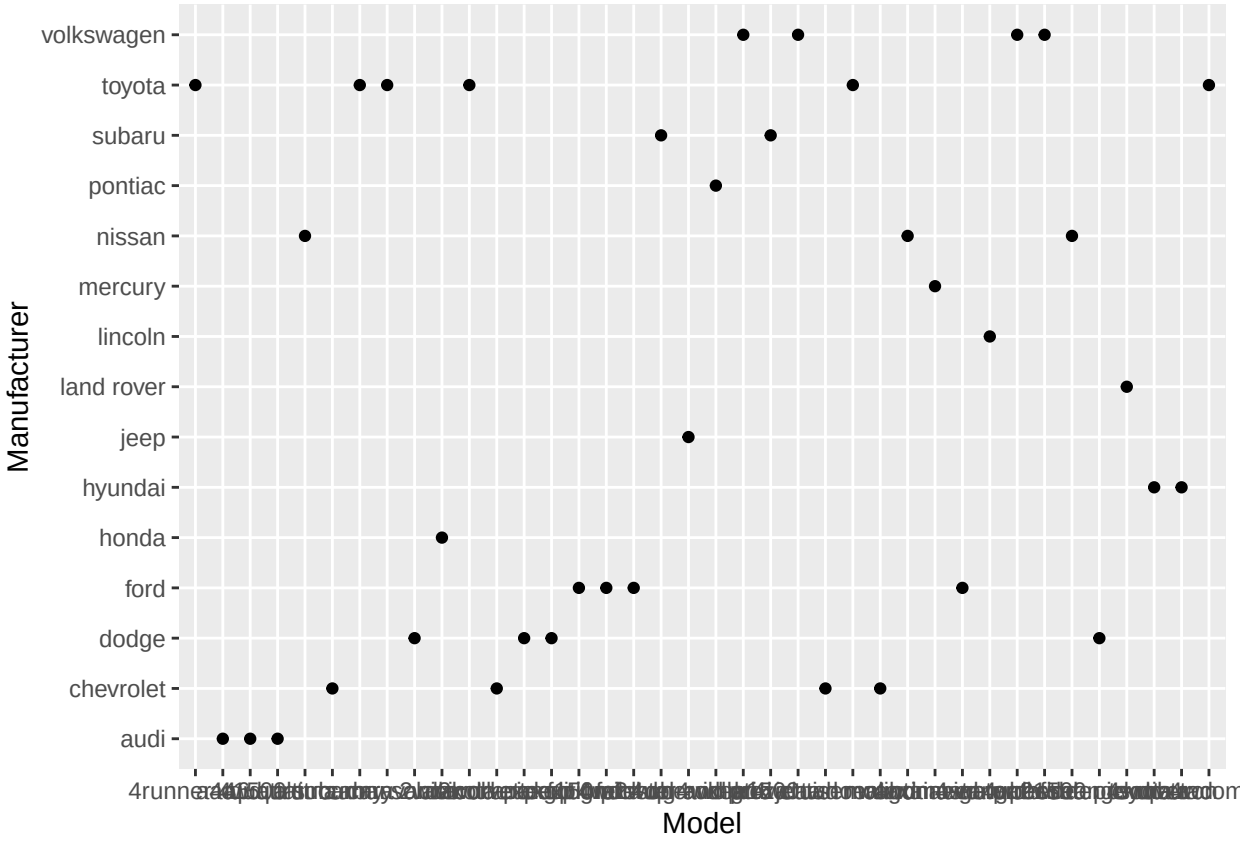
```
## # A tibble: 38 x 3
## # Groups:   Manufacturer, Model [38]
##    Manufacturer Model              Counts
##    <chr>        <chr>               <int>
##  1 audi         a4                      7
##  2 audi         a4 quattro              8
##  3 audi         a6 quattro              3
##  4 chevrolet    c1500 suburban 2wd      4
##  5 chevrolet    corvette                5
##  6 chevrolet    k1500 tahoe 4wd         4
##  7 chevrolet    malibu                  5
##  8 dodge        caravan 2wd             9
##  9 dodge        dakota pickup 4wd       8
## 10 dodge        durango 4wd             6
## # ... with 28 more rows
```

b. Graph the result by using plot() and ggplot(). Write the codes and its result.

```
plot(unique_data)
```



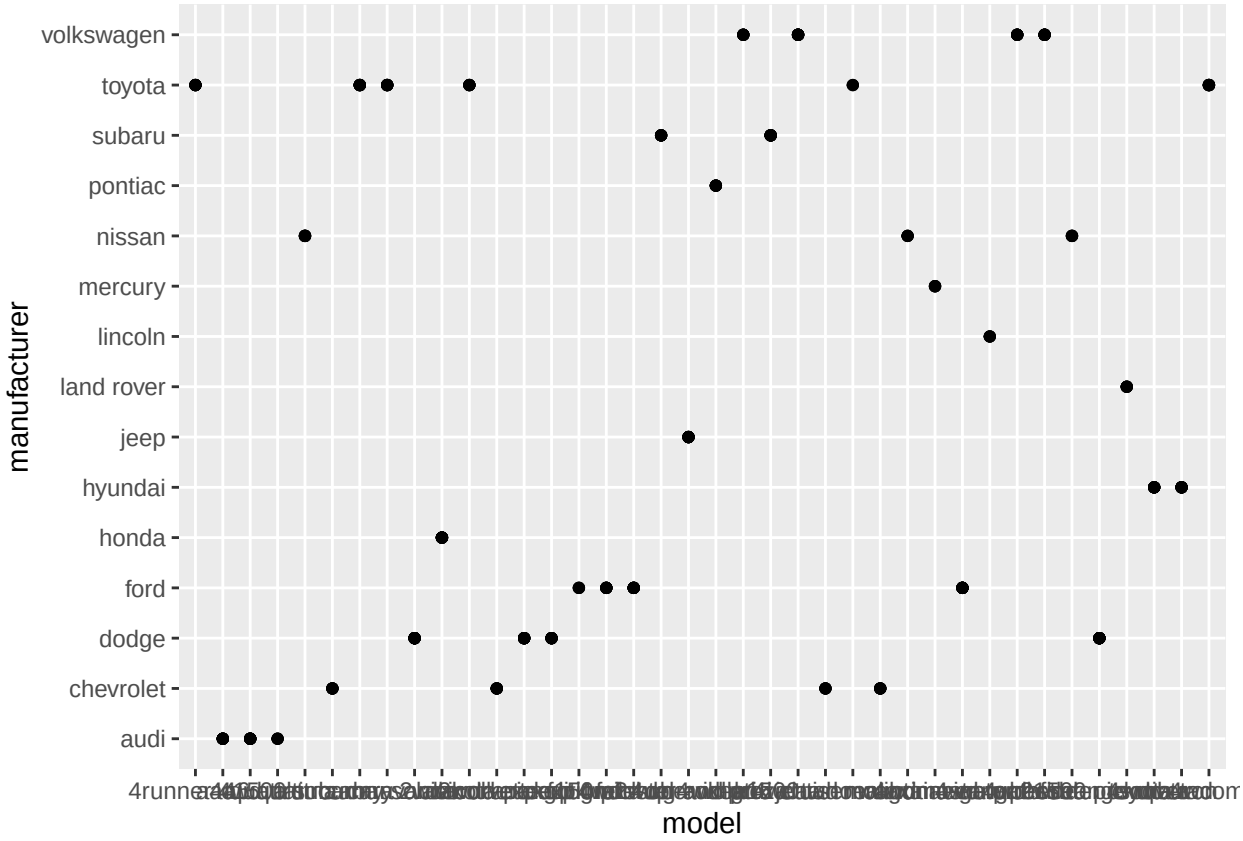

```
ggplot(unique_data, aes(Model, Manufacturer)) + geom_point()
```



3. Same dataset will be used. You are going to show the relationship of the model and the manufacturer.

a. What does ggplot(mpg, aes(model, manufacturer)+ geom_point() show?

```
ggplot(mpg, aes(model, manufacturer)) + geom_point()
```



Answer: It connects the models to their appropriate manufacturers through plotting it.

b. For you, is it useful? If not, how could you modify the data to make it more informative?

Answer: It is useful because it arranges and organizes the data, by that, it is easy to distinguish the relationship between manafacturers and models.

4. Using the pipe (%>%), group the model and get the number of cars per model. Show codes and its result.

```
group_model <- unique_data %>% group_by(Model) %>% count()
group_model
```

```
## # A tibble: 38 x 2
## # Groups:   Model [38]
##    Model                  n
##    <chr>              <int>
##  1 4runner 4wd            1
##  2 a4                     1
##  3 a4 quattro             1
##  4 a6 quattro             1
##  5 altima                 1
##  6 c1500 suburban 2wd     1
##  7 camry                  1
##  8 camry solara           1
##  9 caravan 2wd            1
```

```
## 10 civic                   1
## # ... with 28 more rows
```

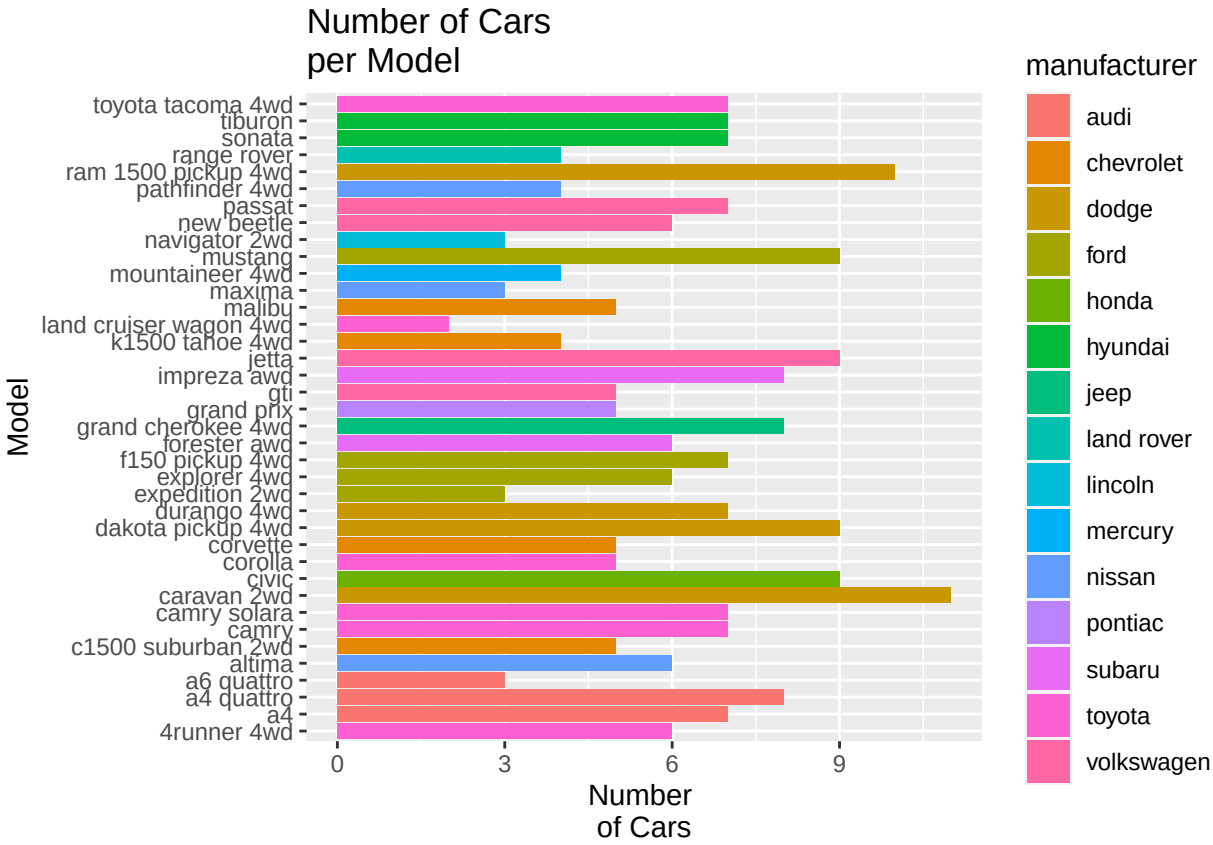
```
  colnames(group_model) <- c("Model","Counts")
  group_model
```

```
## # A tibble: 38 x 2
## # Groups:   Model [38]
##    Model              Counts
##    <chr>               <int>
##  1 4runner 4wd             1
##  2 a4                      1
##  3 a4 quattro              1
##  4 a6 quattro              1
##  5 altima                  1
##  6 c1500 suburban 2wd      1
##  7 camry                   1
##  8 camry solara            1
##  9 caravan 2wd             1
## 10 civic                   1
## # ... with 28 more rows
```

a. Plot using the geom_bar() + coord_flip() just like what is shown below. Show codes and its result

```
qplot(model,data = mpg,main = "Number of Cars
per Model", xlab = "Model",ylab = "Number
of Cars", geom = "bar", fill = manufacturer) +
coord_flip()
```

```
## Warning: `qplot()` was deprecated in ggplot2 3.4.0.
```

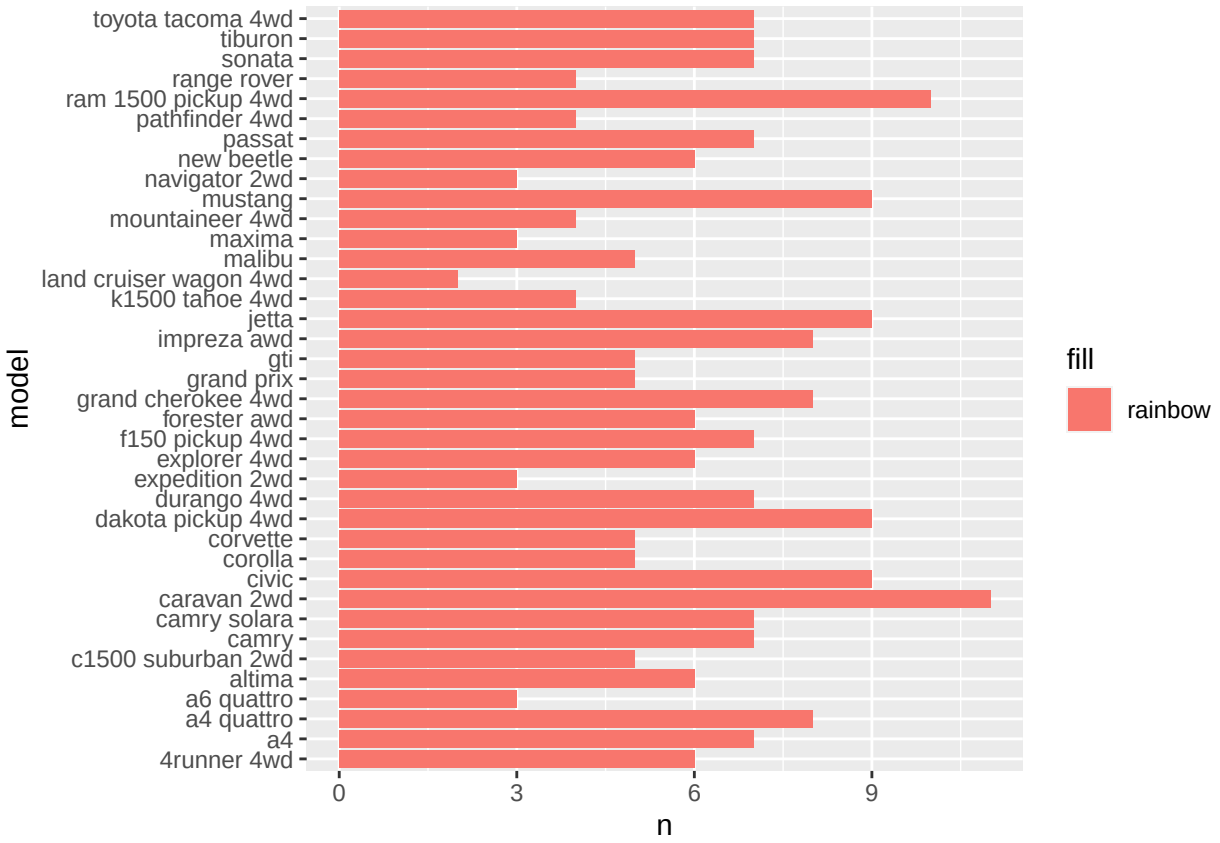


b. Use only the top 20 observations. Show code and results

```
top_obs <- mpg %>%
  group_by(model) %>%
  tally(sort = TRUE)
top_obs
```

```
## # A tibble: 38 x 2
##    model                  n
##    <chr>              <int>
##  1 caravan 2wd           11
##  2 ram 1500 pickup 4wd   10
##  3 civic                  9
##  4 dakota pickup 4wd      9
##  5 jetta                  9
##  6 mustang                9
##  7 a4 quattro             8
##  8 grand cherokee 4wd     8
##  9 impreza awd            8
## 10 a4                     7
## # ... with 28 more rows
```

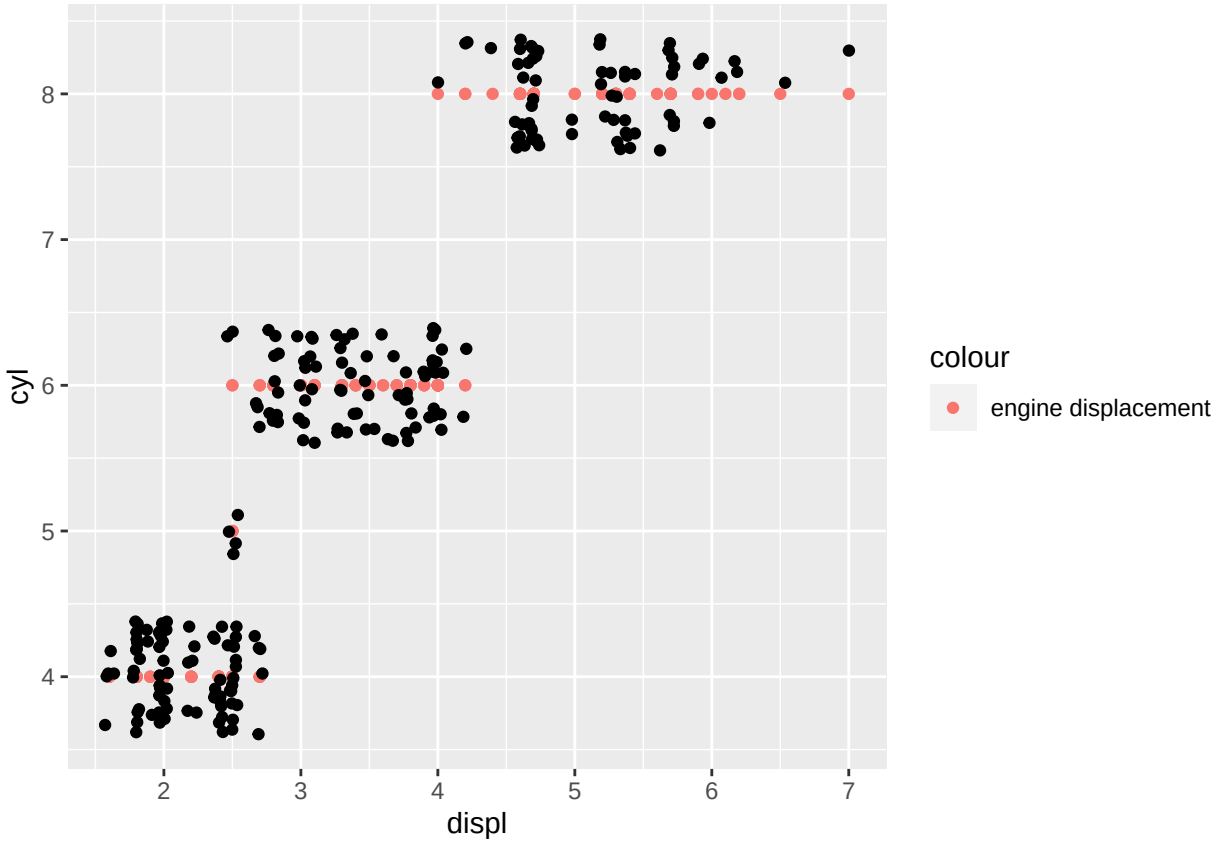
```
ggplot(top_obs, aes(x = model, y = n, fill = "rainbow")) +
  geom_bar(stat = "identity") + coord_flip()
```



5. Plot the relationship between cyl - number of cylinders and displ - engine displacement using geom_point with aesthetic colour = engine displacement. Title should be "Relationship between No. of Cylinders and Engine Displacement".

a. Show the codes and its result.

```
ggplot(data = mpg , mapping = aes(x = displ,
y = cyl, main = "Relationship between No of
Cylinders and Engine Displacement")) + geom_point(mapping=aes(colour =
"engine displacement")) + geom_jitter()
```



b. How would you describe its relationship?

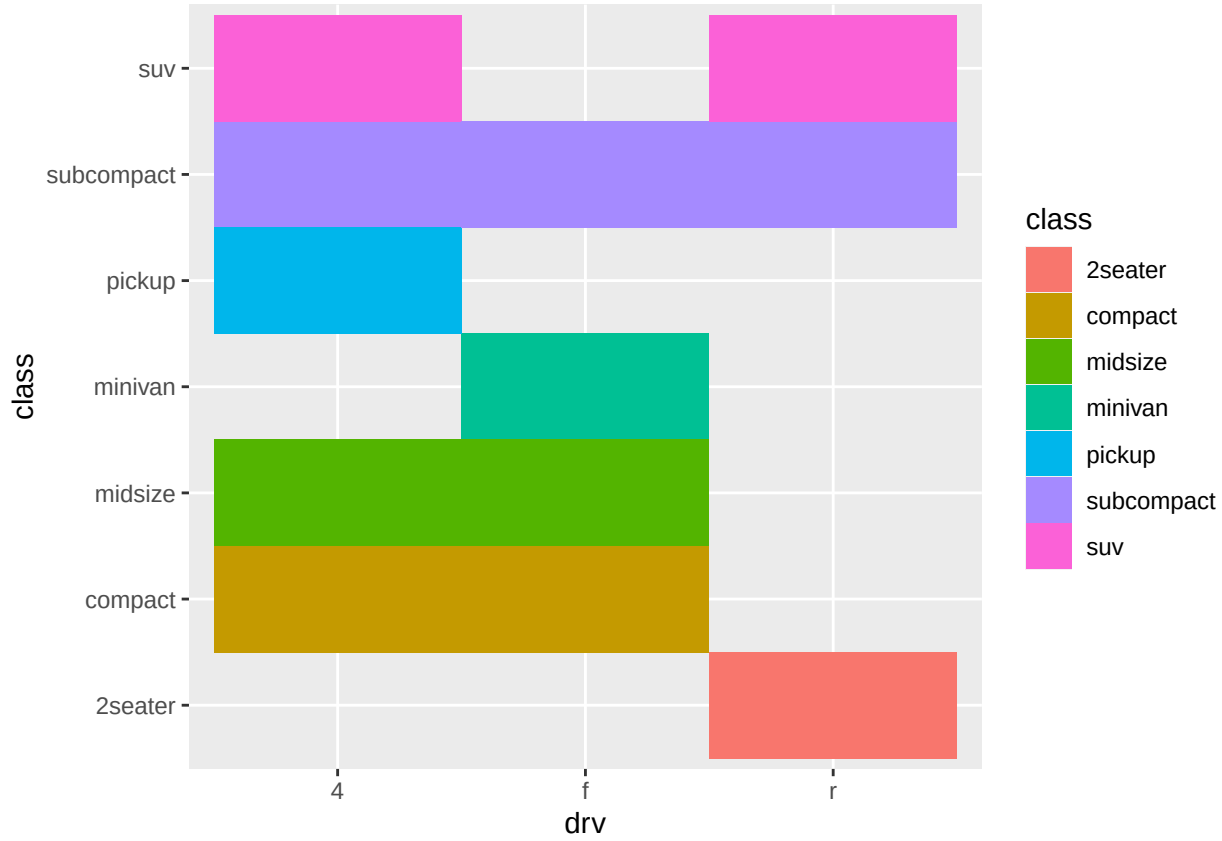
Answer: Less cylinders means less inertia in the engine. That is why as the number of cylinders increases, the dots that represents the engine displacement also increased.

6. Get the total number of observations for drv - type of drive train (f = front-wheel drive, r = rear wheel drive, 4 = 4wd) and class - type of class (Example: suv, 2seater, etc.).

Plot using the geom_tile() where the number of observations for class be used as a fill for aesthetics.

a. Show the codes and its result for the narrative in #6.

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

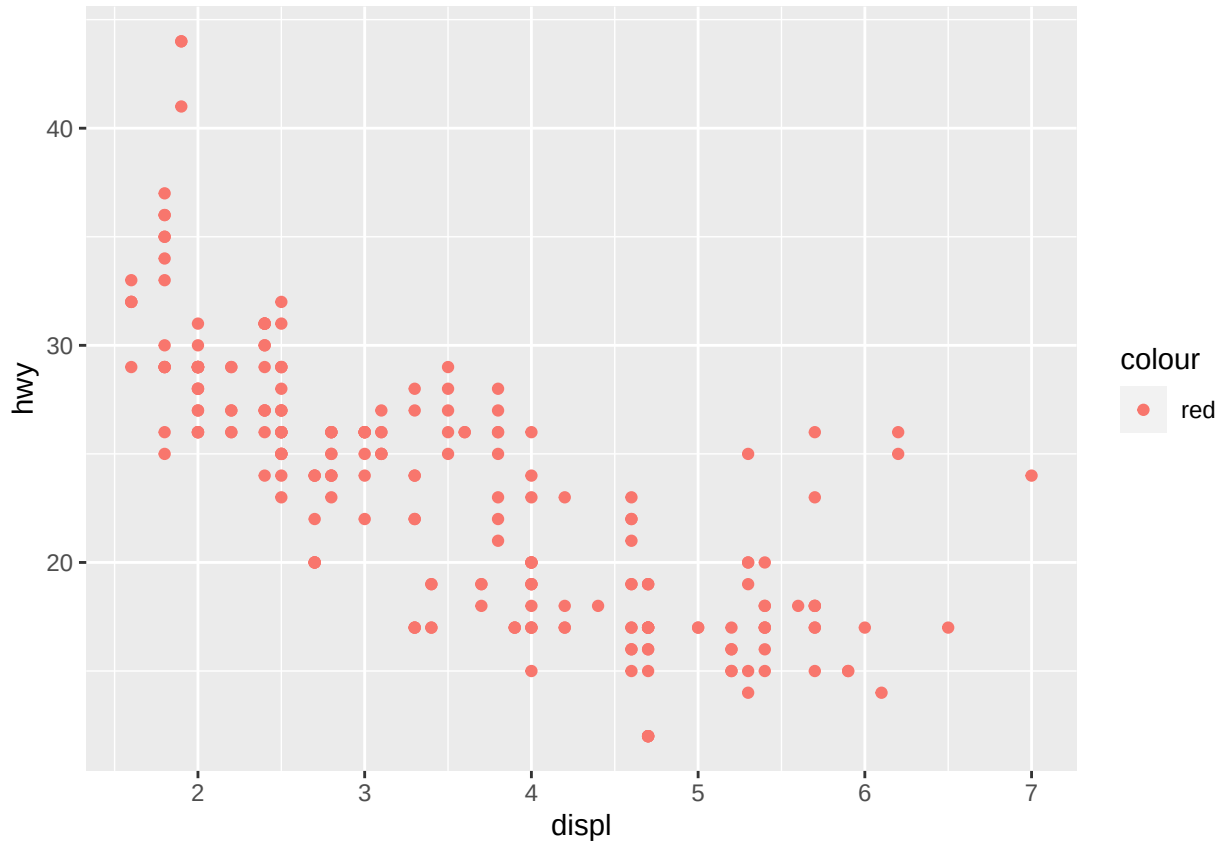


b. Interpret the result.

Answer: Under the 4 wheel drive are the suv, subcompact, pick-up, midsize and compact. Then Mini van is only a front wheel drive while 2seater is only a rear wheel drive.

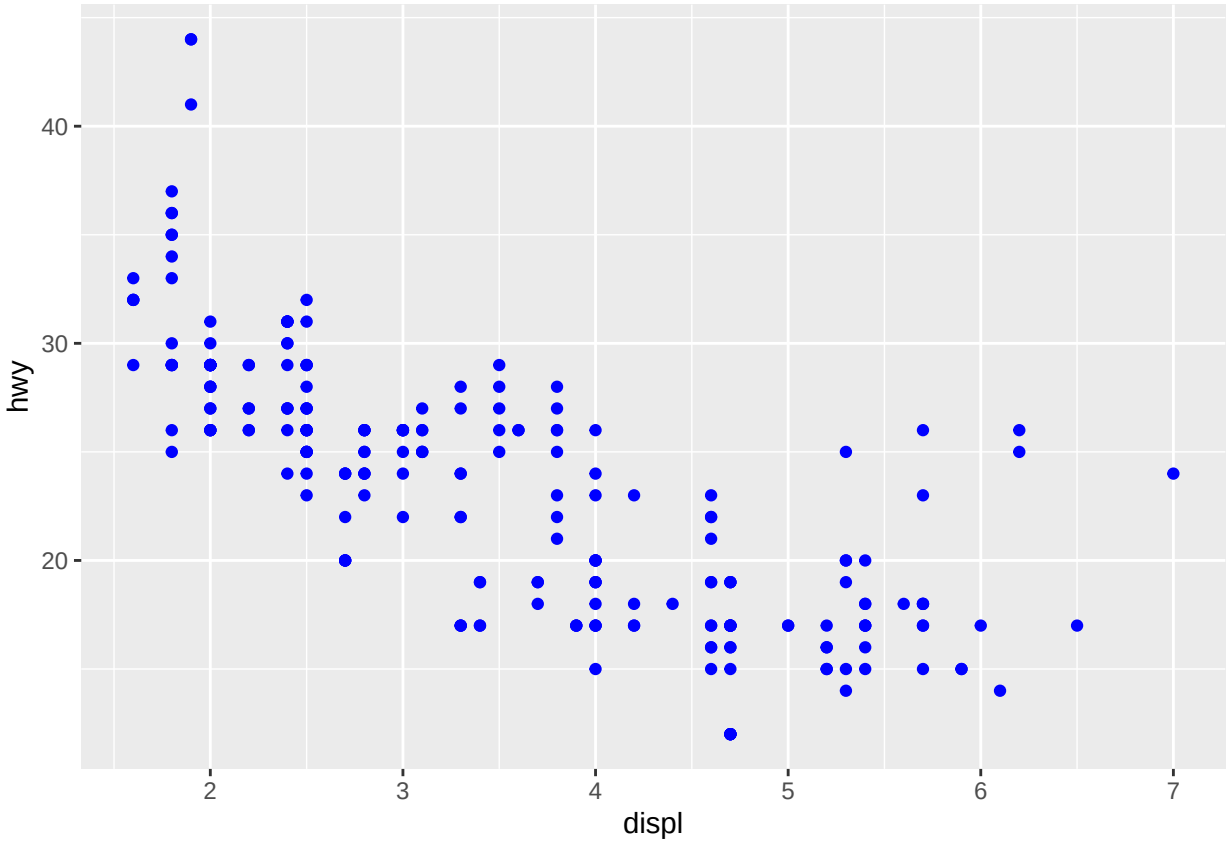
7. Discuss the difference between these codes. Its outputs for each are shown below. Code #1

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



Code #2

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



Answer: The outputs are closely the same except for some minor differences. Based on the output, they differ in color, one is red and the other one is blue. The first one also has a legend for its colour compared to the second one that has none.

8. Try to run the command ?mpg. What is the result of this command?

?mpg

a. Which variables from mpg dataset are categorical?

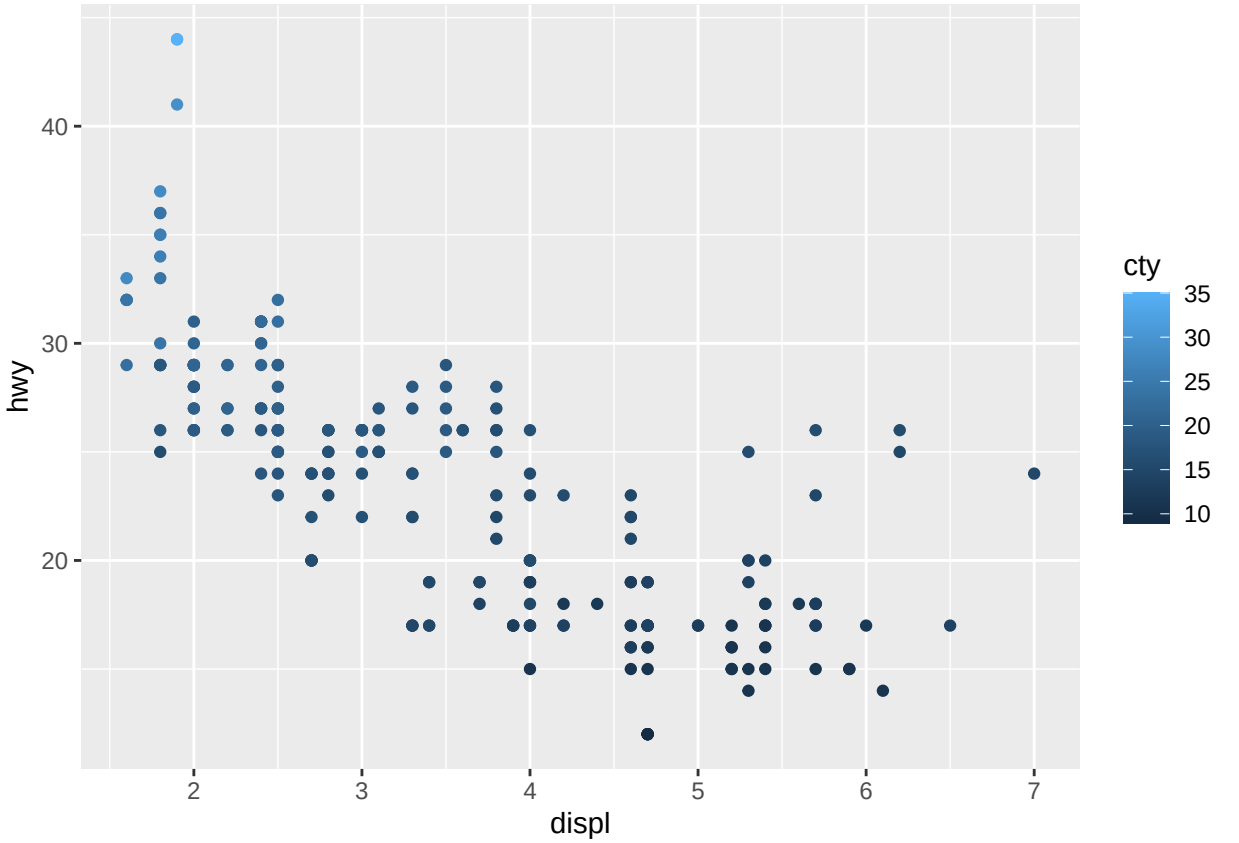
Answer: Categorical variables in mpg include: manufacturer, model, trans (type of transmission), drv (front-wheel drive, rear-wheel, 4wd), fl (fuel type), and class (type of car).

b. Which are continuous variables?

Answer: doubles or integers

c. Plot the relationship between displ (engine displacement) and hwy(highway miles per gallon). Mapped it with a continuous variable you have identified in #5-b. What is its result? Why it produced such output?

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

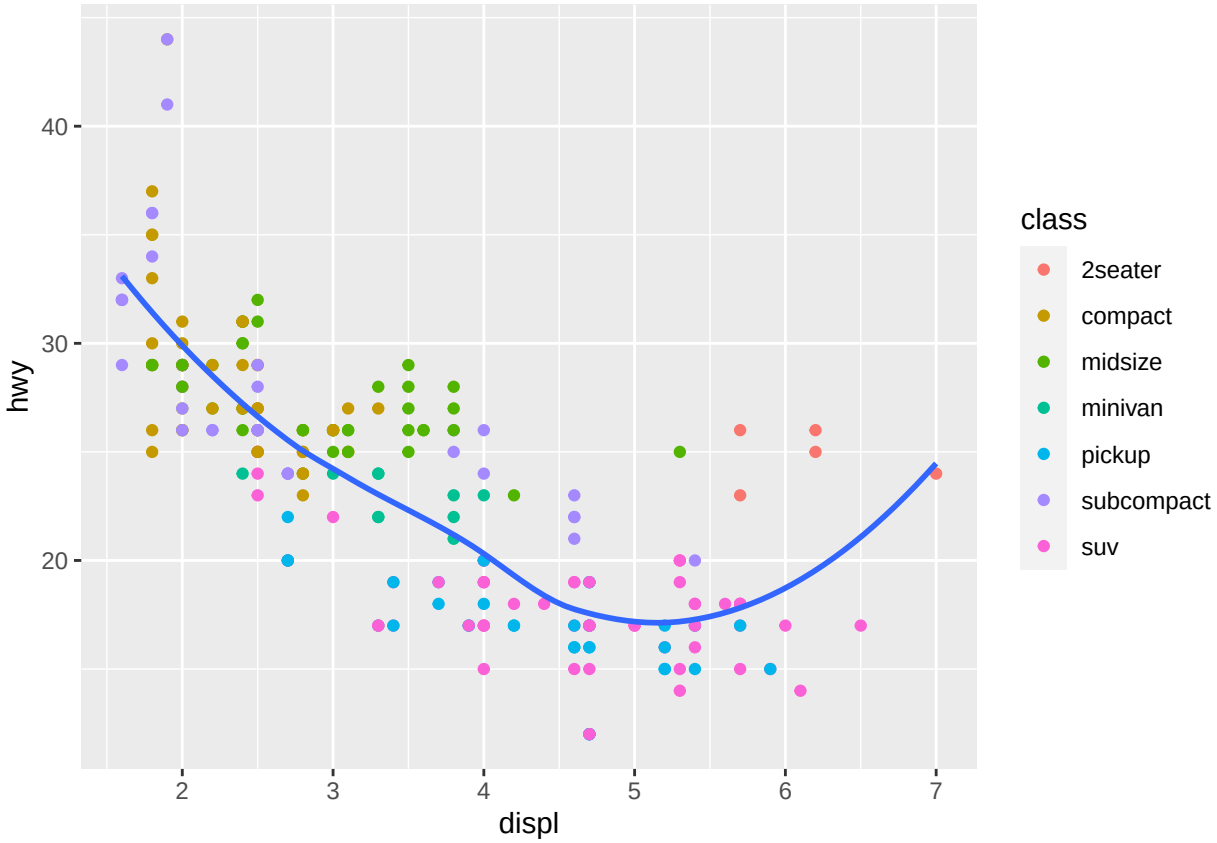


Answer: The displ(engine displacement) and hwy(highway miles per gallon) are matched together through mapping/plotting it then the values are represented by the different shade of color blue.

9. Plot the relationship between displ(engine displacement) and hwy(highway miles per gallon) using geom_point(). Add a trend line over the existing plot using geom_smooth() with se = FALSE. Default method is "loess".

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

```
## `geom_smooth()` using method = 'loess' and formula = 'y ~ x'
```



10. Using the relationship of displ and hwy, add a trend line over existing plot. Set these = FALSE to remove the confidence interval and method = lm to check for linear modeling.

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

```
## `geom_smooth()` using method = 'loess' and formula = 'y ~ x'

## Warning in simpleLoess(y, x, w, span, degree = degree, parametric =
## parametric, : span too small. fewer data values than degrees of freedom.

## Warning in simpleLoess(y, x, w, span, degree = degree, parametric =
## parametric, : pseudoinverse used at 5.6935

## Warning in simpleLoess(y, x, w, span, degree = degree, parametric =
## parametric, : neighborhood radius 0.5065

## Warning in simpleLoess(y, x, w, span, degree = degree, parametric =
## parametric, : reciprocal condition number 0

## Warning in simpleLoess(y, x, w, span, degree = degree, parametric =
## parametric, : There are other near singularities as well. 0.65044
```

```
## Warning in simpleLoess(y, x, w, span, degree = degree, parametric =
## parametric, : pseudoinverse used at 4.008

## Warning in simpleLoess(y, x, w, span, degree = degree, parametric =
## parametric, : neighborhood radius 0.708

## Warning in simpleLoess(y, x, w, span, degree = degree, parametric =
## parametric, : reciprocal condition number 0

## Warning in simpleLoess(y, x, w, span, degree = degree, parametric =
## parametric, : There are other near singularities as well. 0.25
```