

Rworksheet_lago6

Jisielle Mae Lago

2022-12-22

```
install.packages("dplyr")
```

```
## Installing package into '/cloud/lib/x86_64-pc-linux-gnu-library/4.2'  
## (as 'lib' is unspecified)
```

```
install.packages("ggplot2")
```

```
## Installing package into '/cloud/lib/x86_64-pc-linux-gnu-library/4.2'  
## (as 'lib' is unspecified)
```

```
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
```

```
library(ggplot2)
```

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

```
data(mpg)
```

```
nrow(mpg)
```

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

```
ncol(mpg)
```

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

```
#There are 11 columns and 234 rows in mpg data set.
```

```
#2. Which manufacturer has the most models in this data set? Which model has the most  
#variations? Ans:
```

```
manu_mpg <- mpg$manufacturer
```

```
table(manu_mpg)
```

```
## manu_mpg
```

```
##      audi  chevrolet    dodge      ford    honda  hyundai    jeep  
##      18       19       37       25       9       14       8
```

```
## land rover      lincoln      mercury      nissan      pontiac      subaru      toyota
##              4              3              4              13              5              14              34
## volkswagen
##              27
```

#dodge has the most models and variations in this data set.

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

```
g1 <- mpg
unimod <- g1 %>% group_by(manufacturer, model) %>%
  distinct() %>% count()
unimod
```

```
## # A tibble: 38 x 3
## # Groups:   manufacturer, model [38]
##   manufacturer model          n
##   <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
```

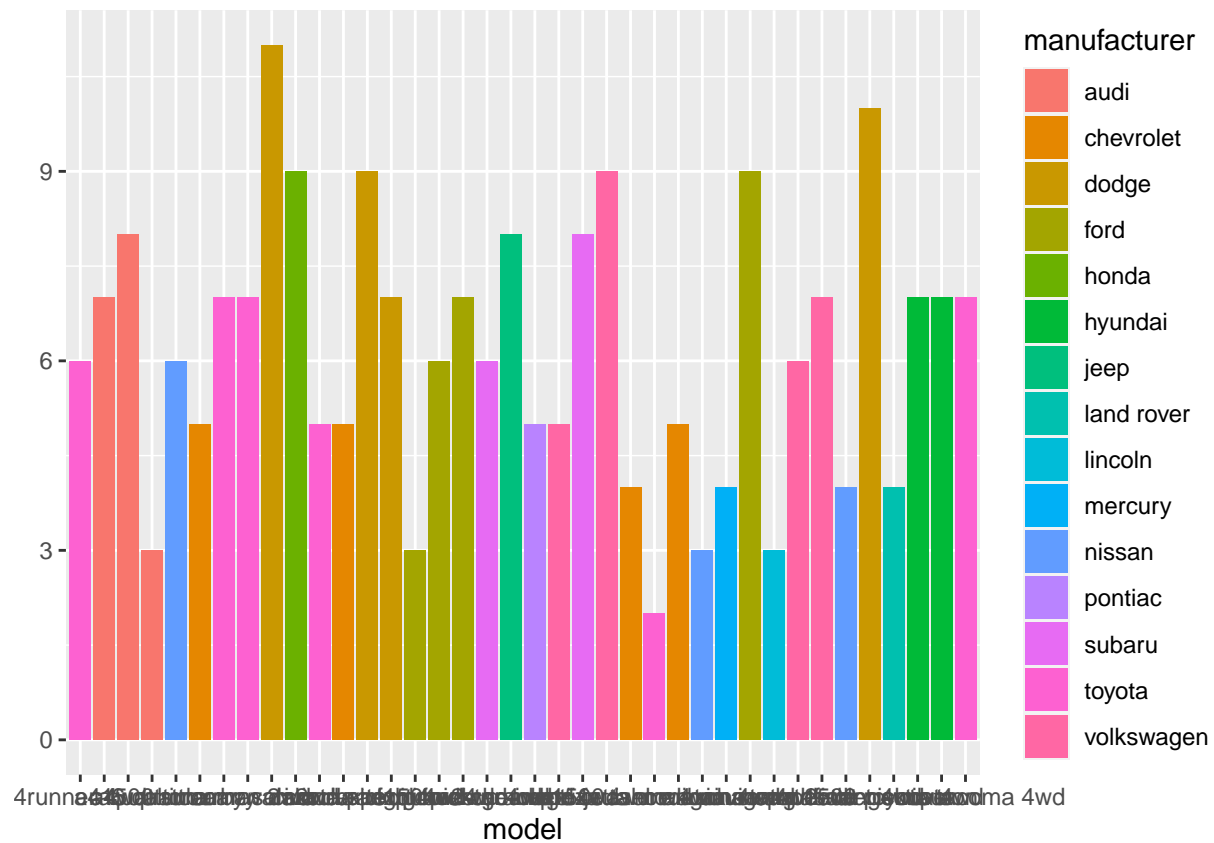
```
colnames(unimod) <- c("Manufacturer", "Model", "Counts")
unimod
```

```
## # 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
```

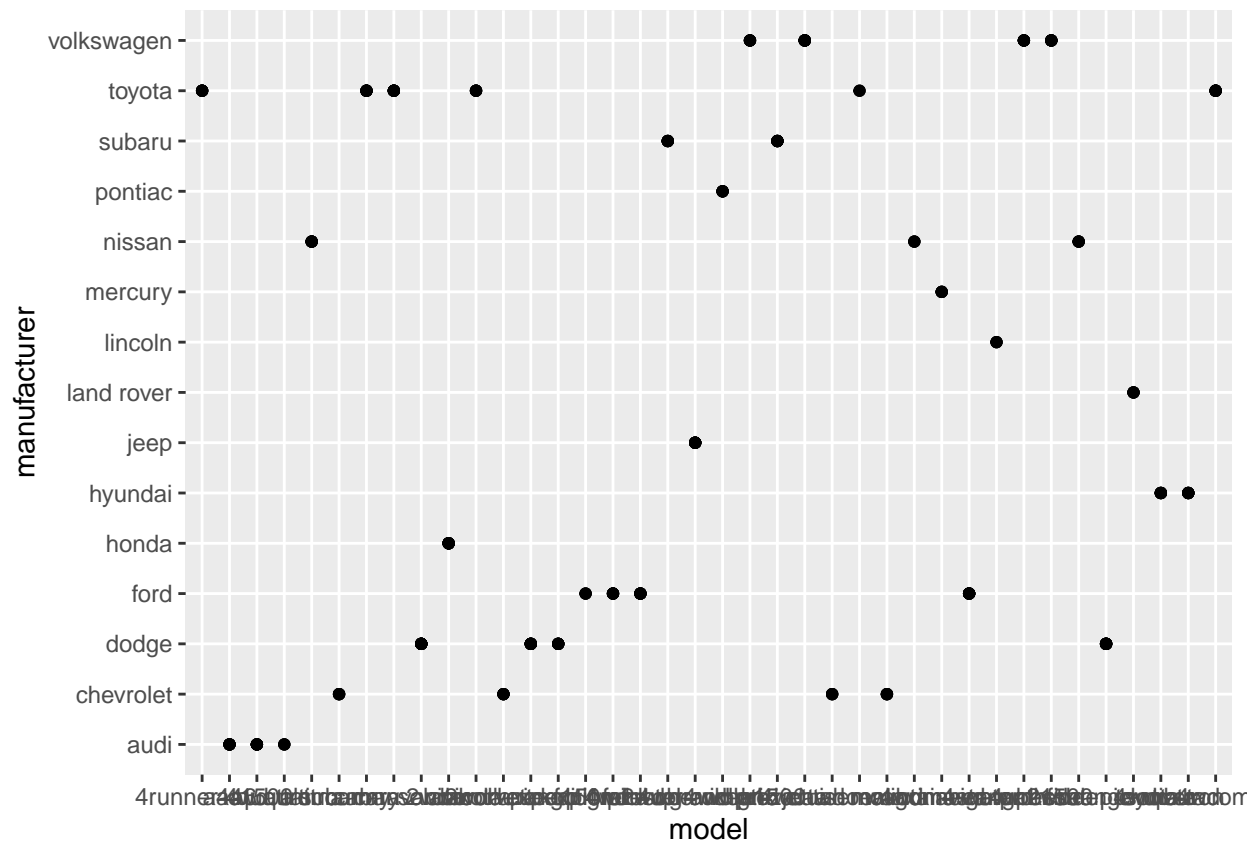
#2.b Graph the result by using plot() and ggplot(). Write the codes and its result.

```
qplot(model, data = mpg, geom = "bar", fill=manufacturer)
```

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



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



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

```
rdata <- mpg
models <- rdata %>% group_by(manufacturer, model) %>%
  distinct() %>% count()
models
```

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

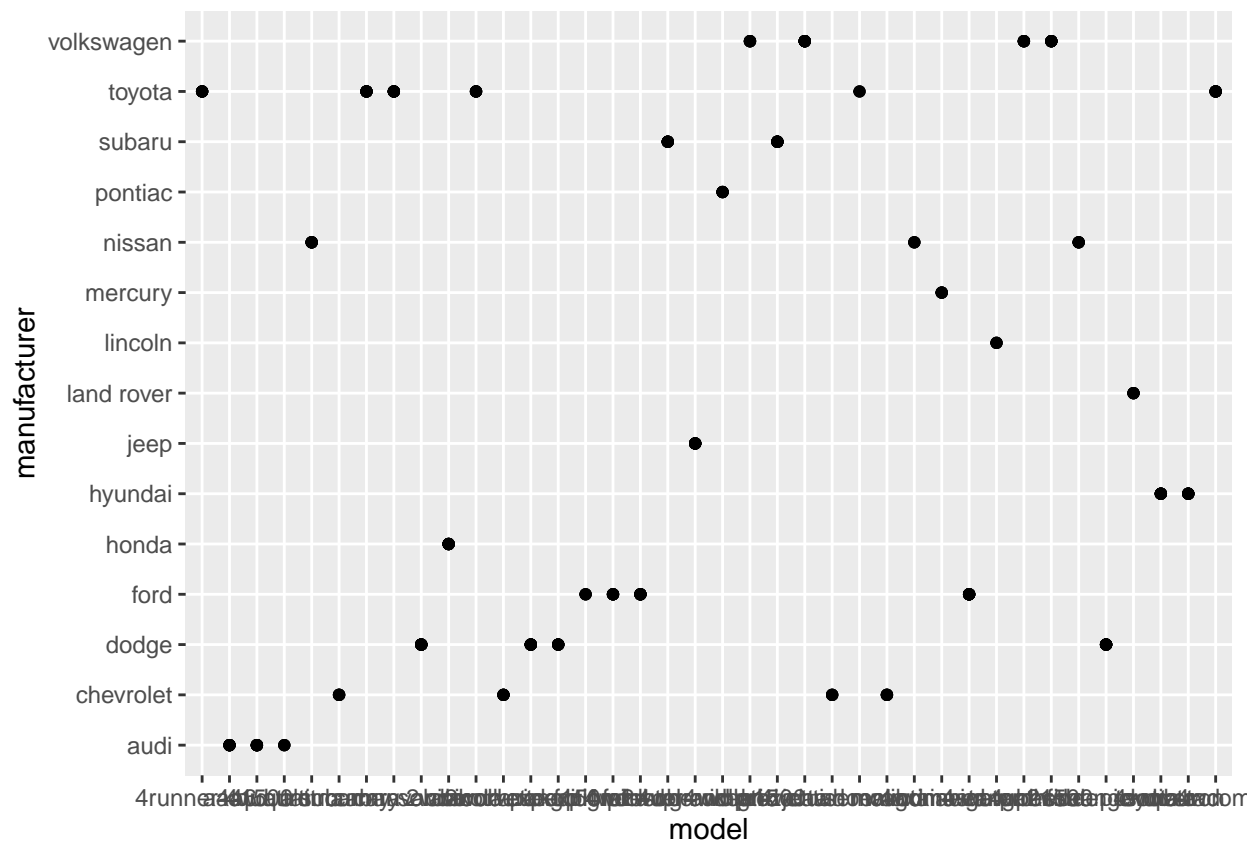
```
colnames(models) <- c("Manufacturer", "Model", "Counts")
models
```

```
## # 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
```

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

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



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

```
rdata2 <- models %>% group_by(Model) %>% count()
colnames(rdata2) <- c("Model", "Counts")
rdata2
```

```
## # A tibble: 38 x 2
## # Groups:   Model [38]
##   Model      Counts
```

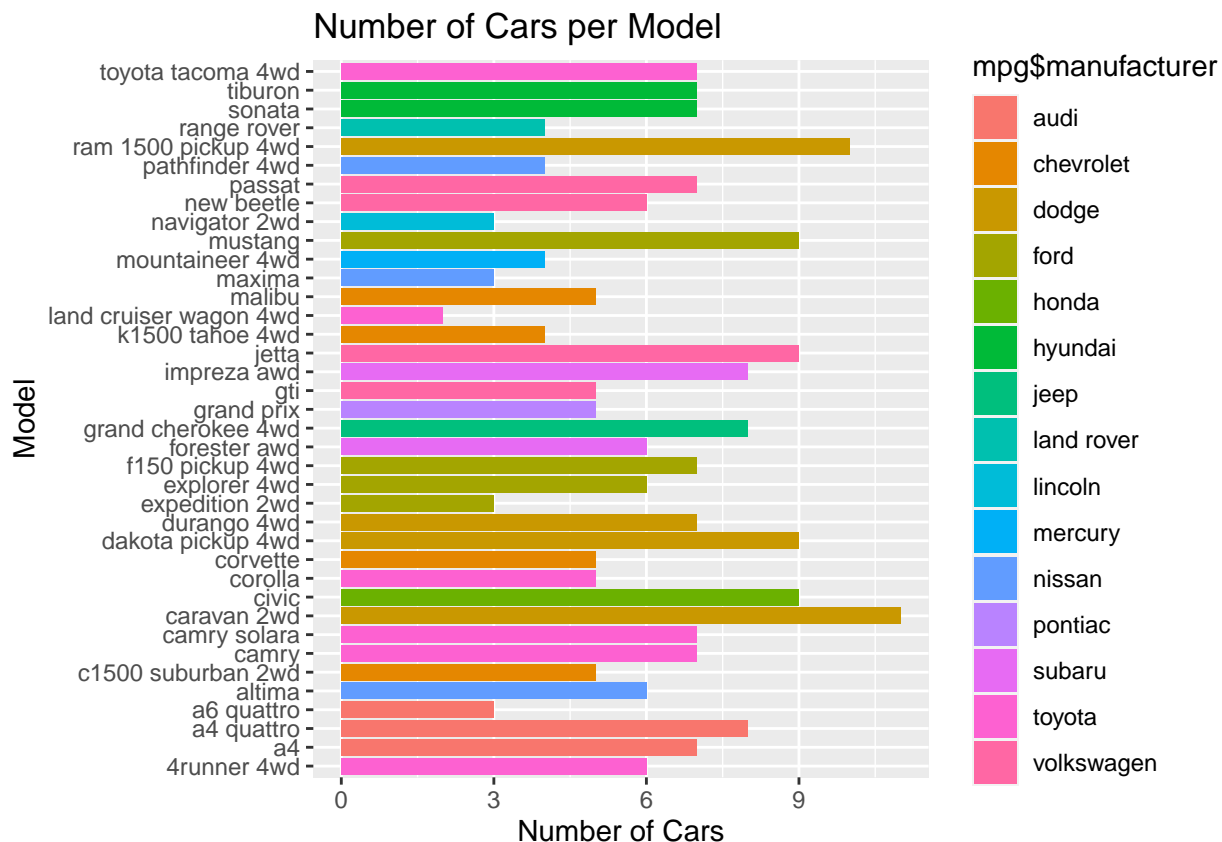
```
##      <chr>                <int>
## 1 4runner 4wd              1
## 2 a4                      1
## 3 a4 quattro              1
## 4 a6 quattro              1
## 5 altima                  1
## 6 c1500 suburban 2wd      1
## 7 camry                   1
## 8 camry solara            1
## 9 caravan 2wd             1
## 10 civic                  1
## # ... with 28 more rows
```

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

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

```
## Warning: Use of `mpg$model` is discouraged.
## i Use `model` instead.

## Warning: Use of `mpg$manufacturer` is discouraged.
## i Use `manufacturer` instead.
```



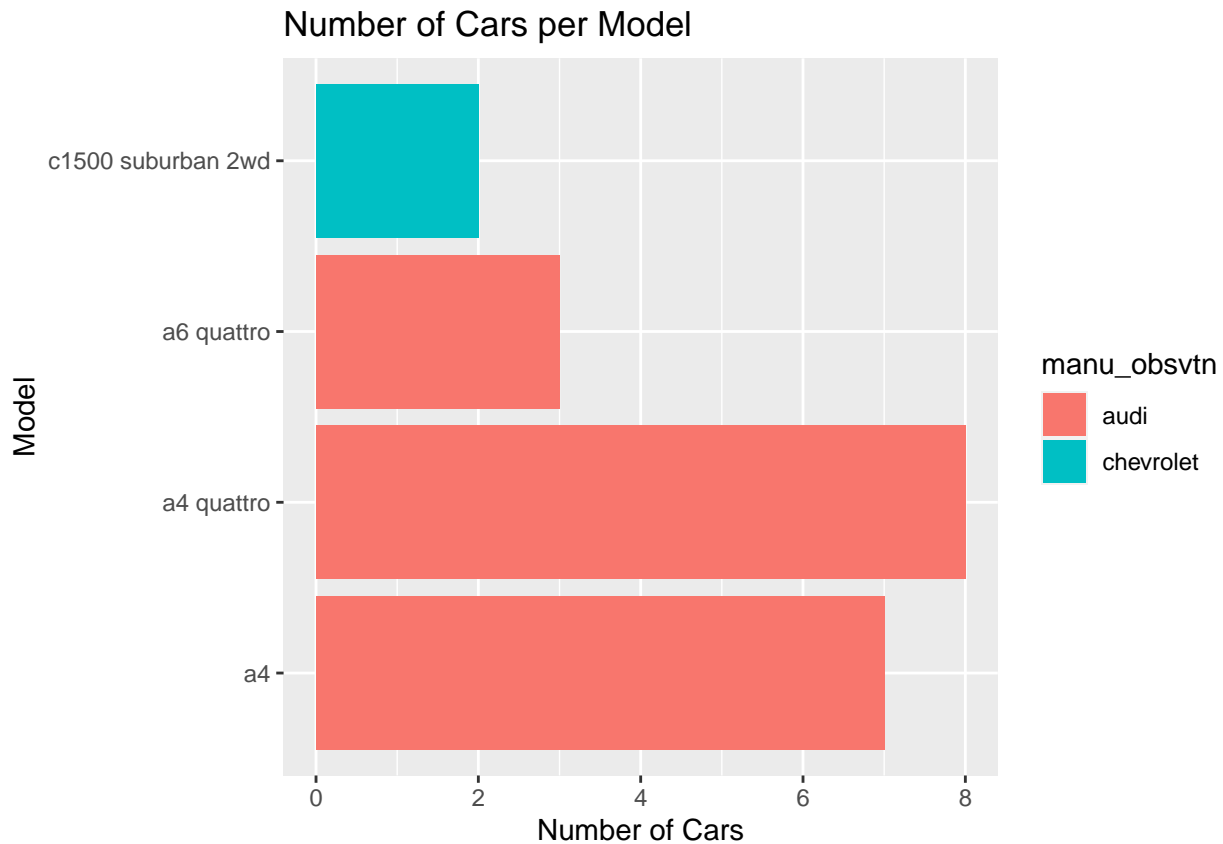
#4.b Use only the top 20 observations. Show code and results.

```

data_obsvtm <- subset(mpg[c(1:20),c(1:11)])
manu_obsvtm <- mpg$manufacturer[1:20]
model_obsvtm <- mpg$model[1:20]

qplot(model_obsvtm, data = data_obsvtm, main = "Number of Cars per Model",
      xlab = "Model", ylab = "Number of Cars", geom = "bar", fill = manu_obsvtm) + coord_flip()

```

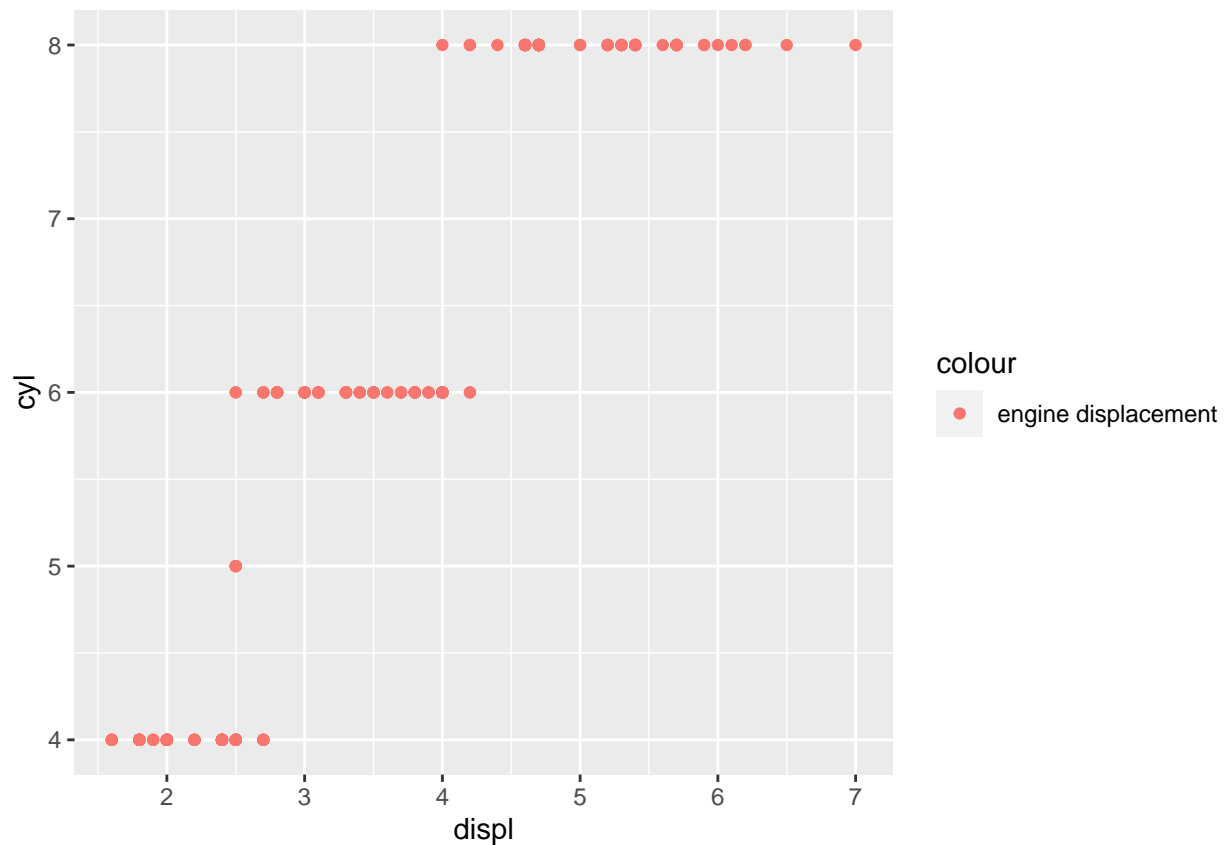


#5. Plot the relationship between cyl - number of cylinders and displ - engine displacement using geom_point. # "Relationship between No. of Cylinders and Engine Displacement". #5.a Show the codes and its result.

```

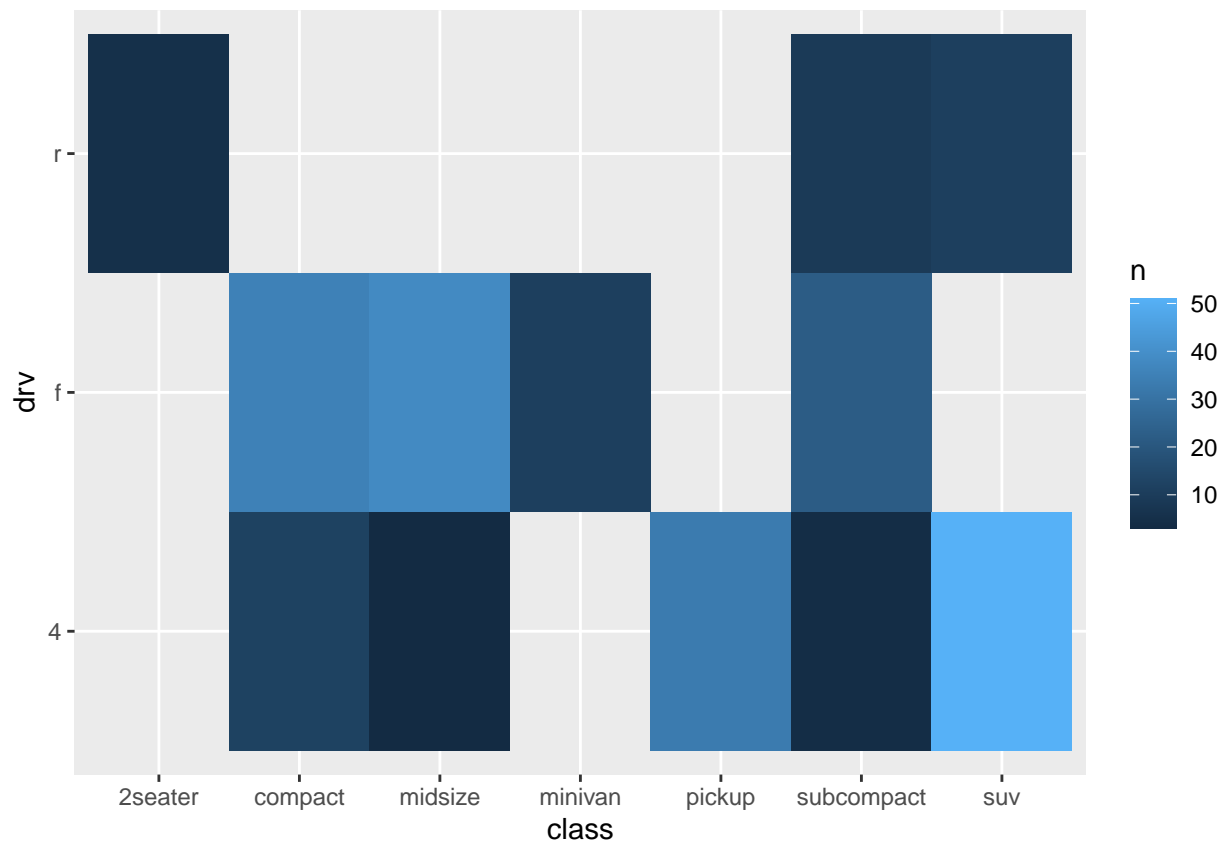
ggplot(data = mpg, mapping = aes(x = displ,
                                y = cyl,
                                main = "Relationship between No. of Cylinders and Engine Displacement")) +
  geom_point(mapping = aes(colour = "engine displacement"))

```



#6. Get the total number of observations for drv - type of drive train (f = front-wheel drive, #Plot using the geom_tile() where the number of observations for class be used as a fill for aesthetics #6.a Show the codes and its result for the narrative in #6.

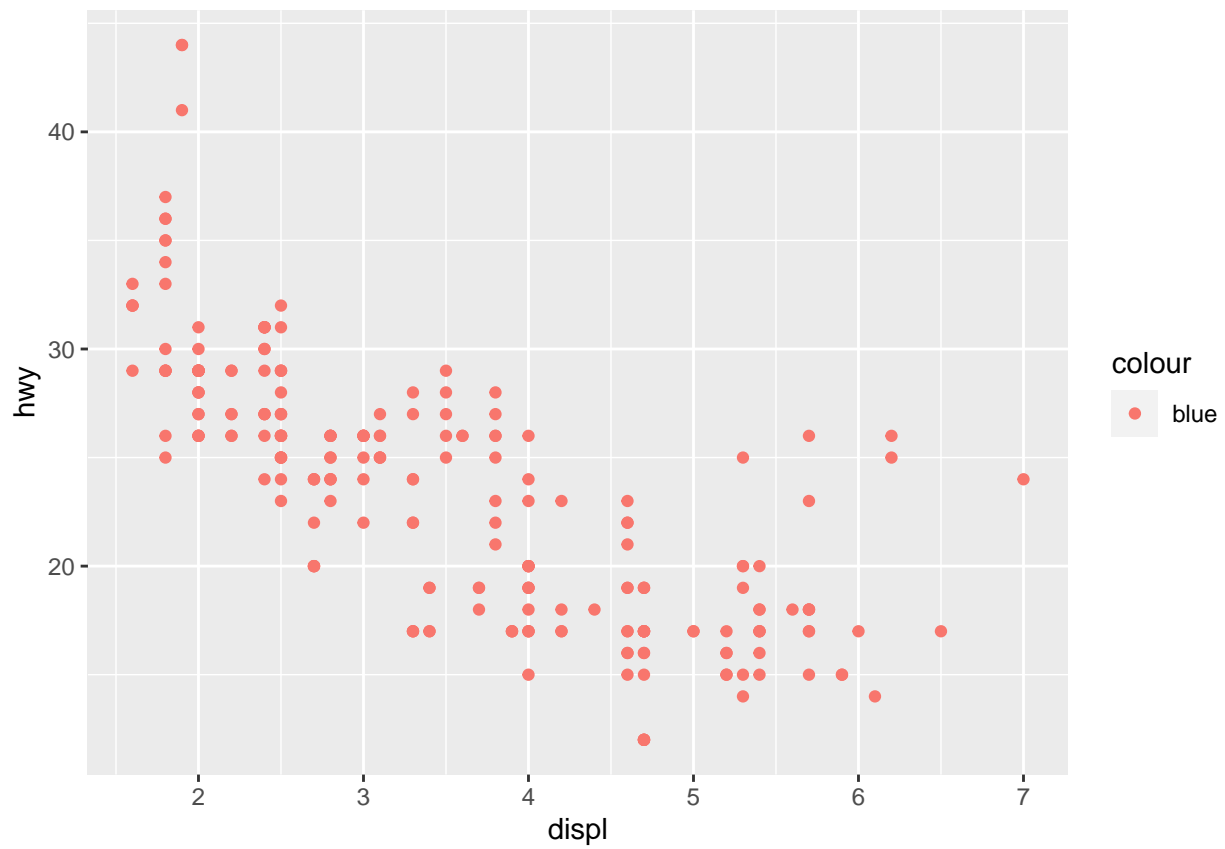
```
mpg %>%
  count(class, drv) %>%
  ggplot(aes(x = class, y = drv)) +
  geom_tile(mapping = aes(fill = n))
```

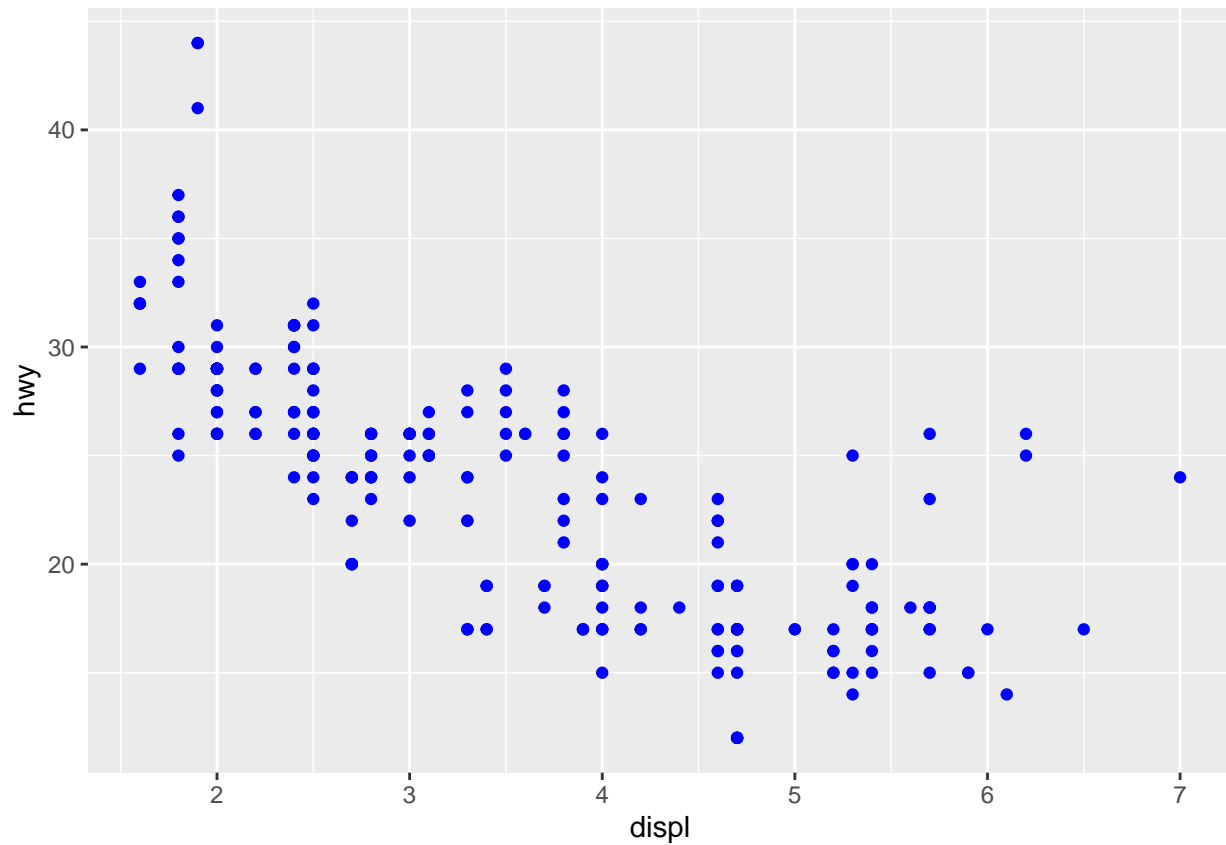
#7. Discuss the difference between these codes. Its outputs for each are shown below.

#Code 1

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



```
#Code 2  
ggplot(data = mpg) + geom_point(mapping = aes(x = displ, y = hwy), colour = "blue")
```

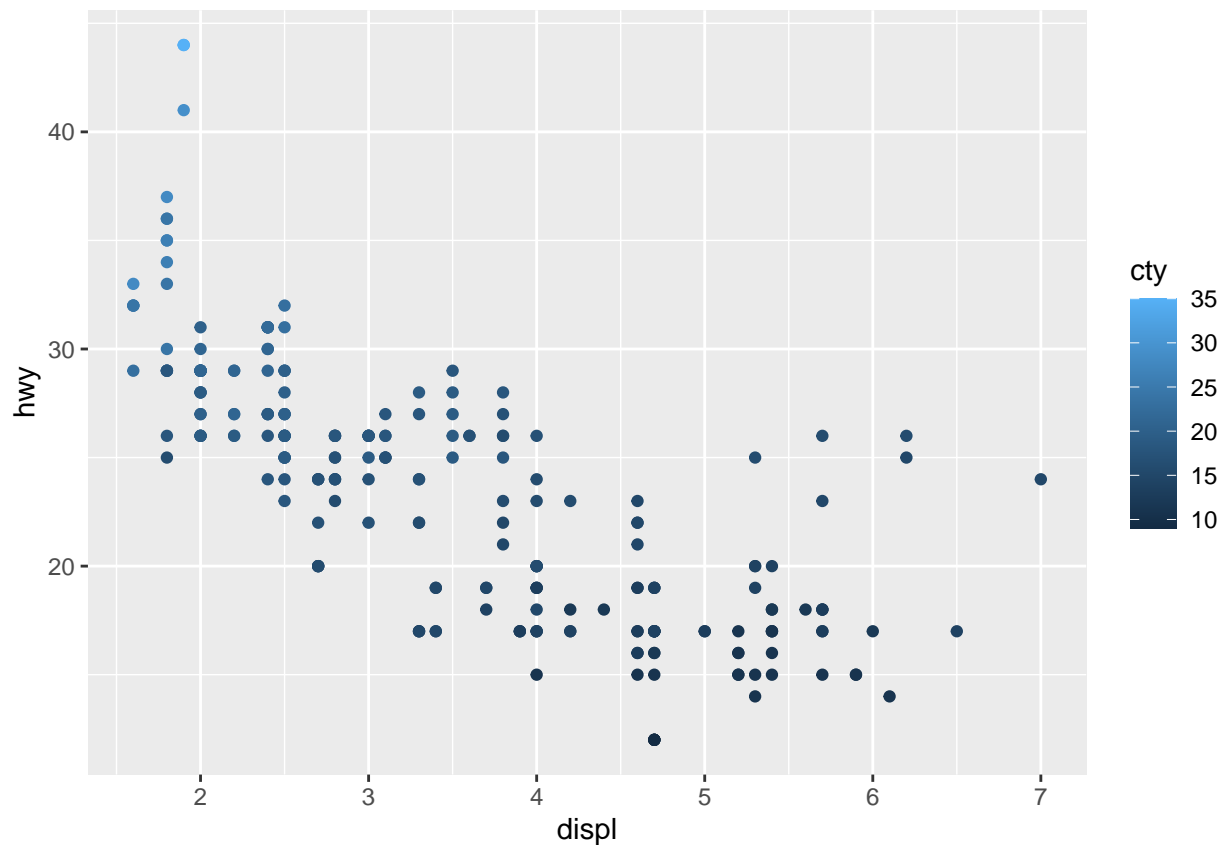


#8. Try to run the command `?mpg`. What is the result of this command?

`?mpg`

#8.c. Plot the relationship between `displ` (engine displacement) and `hwy` (highway miles per gallon). Map the result? Why it produced such output?

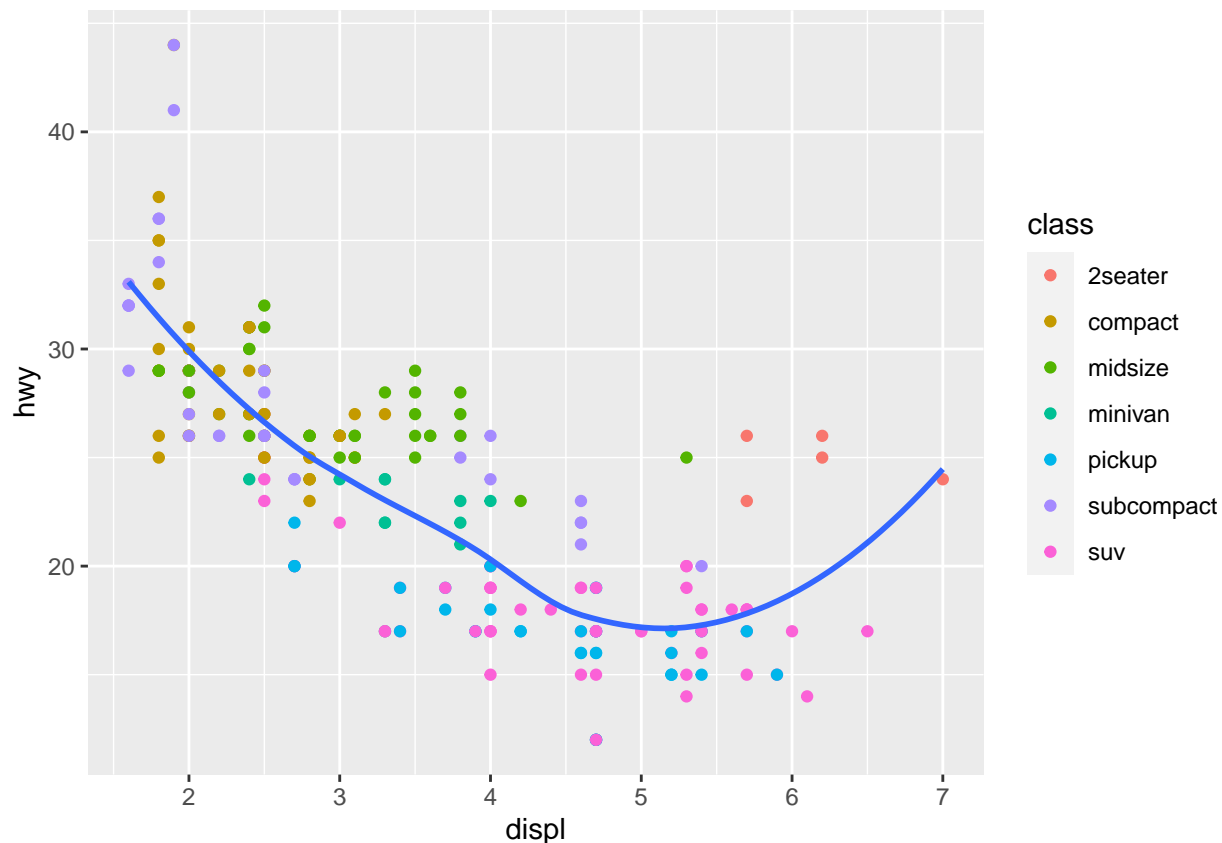
`ggplot(mpg, aes(x = displ, y = hwy, colour = cty)) + geom_point()`



#9. Plot the relationship between displ (engine displacement) and hwy(highway miles per gallon) 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 the `se = 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
```

