# Worksheet#4C

Marivic Amuan

2023-11-22

```
#1. Use the dataset mpg

#1A. Show your solutions on how to import a csv file into the environment.

library(readr)

mpg <- read_csv("mpg.csv")
```

```
## New names:
## Rows: 234 Columns: 12
## -- Column specification
## -------------------------------------------------------- Delimiter: "," chr
## (6): manufacturer, model, trans, drv, fl, class dbl (6): ...1, displ, year,
## cyl, cty, hwy
## i Use `spec()` to retrieve the full column specification for this data. i
## Specify the column types or set `show_col_types = FALSE` to quiet this message.
## * `` -> `...1`
```

```
head(mpg)
```

```
## # A tibble: 6 x 12
##    ...1 manufacturer model displ  year   cyl trans drv     cty   hwy fl    class
##   <dbl> <chr>        <chr> <dbl> <dbl> <dbl> <chr> <chr> <dbl> <dbl> <chr> <chr>
## 1     1 audi         a4      1.8  1999     4 auto~ f        18    29 p     comp~
## 2     2 audi         a4      1.8  1999     4 manu~ f        21    29 p     comp~
## 3     3 audi         a4      2    2008     4 manu~ f        20    31 p     comp~
## 4     4 audi         a4      2    2008     4 auto~ f        21    30 p     comp~
## 5     5 audi         a4      2.8  1999     6 auto~ f        16    26 p     comp~
## 6     6 audi         a4      2.8  1999     6 manu~ f        18    26 p     comp~
```

```
#1B. Which variables from mpg dataset are categorical?

str(mpg)
```

```
## spc_tbl_ [234 x 12] (S3: spec_tbl_df/tbl_df/tbl/data.frame)
##  $ ...1        : num [1:234] 1 2 3 4 5 6 7 8 9 10 ...
##  $ manufacturer: chr [1:234] "audi" "audi" "audi" "audi" ...
##  $ model       : chr [1:234] "a4" "a4" "a4" "a4" ...
##  $ displ       : num [1:234] 1.8 1.8 2 2 2.8 2.8 3.1 1.8 1.8 2 ...
##  $ year        : num [1:234] 1999 1999 2008 2008 1999 ...
##  $ cyl         : num [1:234] 4 4 4 4 6 6 6 4 4 4 ...
##  $ trans       : chr [1:234] "auto(l5)" "manual(m5)" "manual(m6)" "auto(av)" ...
##  $ drv         : chr [1:234] "f" "f" "f" "f" ...
##  $ cty         : num [1:234] 18 21 20 21 16 18 18 18 16 20 ...
```

```
##  $ hwy          : num [1:234] 29 29 31 30 26 26 27 26 25 28 ...
##  $ fl           : chr [1:234] "p" "p" "p" "p" ...
##  $ class        : chr [1:234] "compact" "compact" "compact" "compact" ...
##  - attr(*, "spec")=
##   .. cols(
##   ..   ...1 = col_double(),
##   ..   manufacturer = col_character(),
##   ..   model = col_character(),
##   ..   displ = col_double(),
##   ..   year = col_double(),
##   ..   cyl = col_double(),
##   ..   trans = col_character(),
##   ..   drv = col_character(),
##   ..   cty = col_double(),
##   ..   hwy = col_double(),
##   ..   fl = col_character(),
##   ..   class = col_character()
##   .. )
##  - attr(*, "problems")=<externalptr>
```

```
#ANSWER: Manufacturer,model,year,cyl,trans,drv,fl, and class are the categorical variables.
```

```
#1C. Which are continuous variables?
#The continious variablesa are the , manufacturer, model, disply, year , cyl, cty ,hwy, fl, trans, drv

summary(mpg)
```

```
##       ...1        manufacturer          model               displ
##  Min.   :  1.00   Length:234         Length:234         Min.   :1.600
##  1st Qu.: 59.25   Class :character   Class :character   1st Qu.:2.400
##  Median :117.50   Mode  :character   Mode  :character   Median :3.300
##  Mean   :117.50                                         Mean   :3.472
##  3rd Qu.:175.75                                         3rd Qu.:4.600
##  Max.   :234.00                                         Max.   :7.000
##       year           cyl           trans               drv
##  Min.   :1999   Min.   :4.000   Length:234         Length:234
##  1st Qu.:1999   1st Qu.:4.000   Class :character   Class :character
##  Median :2004   Median :6.000   Mode  :character   Mode  :character
##  Mean   :2004   Mean   :5.889
##  3rd Qu.:2008   3rd Qu.:8.000
##  Max.   :2008   Max.   :8.000
##       cty             hwy             fl               class
##  Min.   : 9.00   Min.   :12.00   Length:234         Length:234
##  1st Qu.:14.00   1st Qu.:18.00   Class :character   Class :character
##  Median :17.00   Median :24.00   Mode  :character   Mode  :character
##  Mean   :16.86   Mean   :23.44
##  3rd Qu.:19.00   3rd Qu.:27.00
##  Max.   :35.00   Max.   :44.00
```

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

#ANSWER: THE MOST MODEL MANUFACTURER IS DODGE.THE MOST VARIATION IS CARAVAN 2WD.
#ANSWER: #The model with most variation is caravan 2wd.
library(magrittr)
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
```

```
model <- mpg %>%
 group_by(manufacturer) %>%
 summarise(count = n()) %>%
 arrange(desc(count))

print(model)
```

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

```
count <- mpg %>%
 group_by(model) %>%
 summarise(variation = n()) %>%
 arrange(desc(variation))

print(count)
```

```
## # A tibble: 38 x 2
##    model               variation
##    <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
## # i 28 more rows
```

```
#2A Group the manufacturers and find the unique models. Show your codes and result.
library(dplyr)

manumodel <- mpg %>%
 group_by(manufacturer) %>%
 summarise(unique_models = n_distinct(model))

print(manumodel)
```

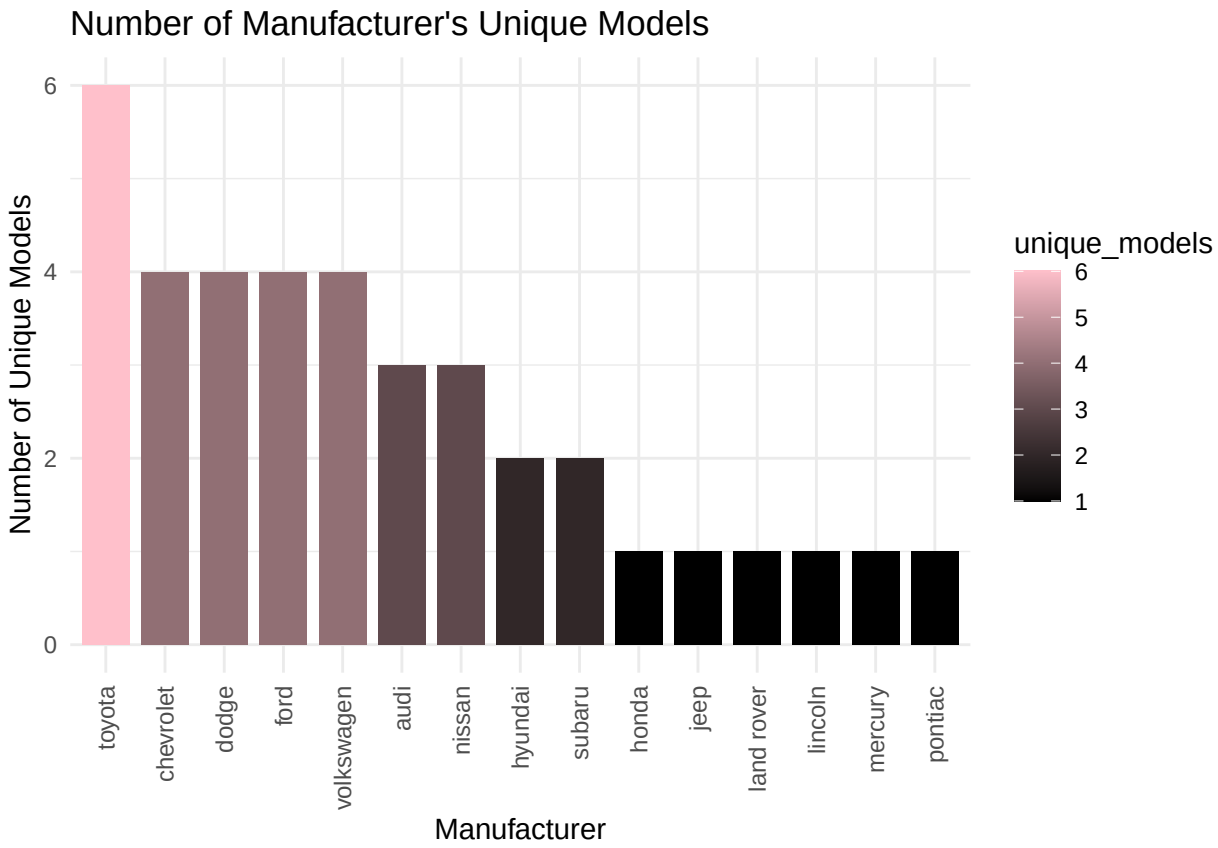
```
## # A tibble: 15 x 2
##    manufacturer unique_models
##    <chr>                <int>
##  1 audi                     3
##  2 chevrolet                4
##  3 dodge                    4
##  4 ford                     4
##  5 honda                    1
##  6 hyundai                  2
##  7 jeep                     1
##  8 land rover               1
##  9 lincoln                  1
## 10 mercury                  1
## 11 nissan                   3
## 12 pontiac                  1
## 13 subaru                   2
## 14 toyota                   6
## 15 volkswagen               4
```

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

library(ggplot2)
```

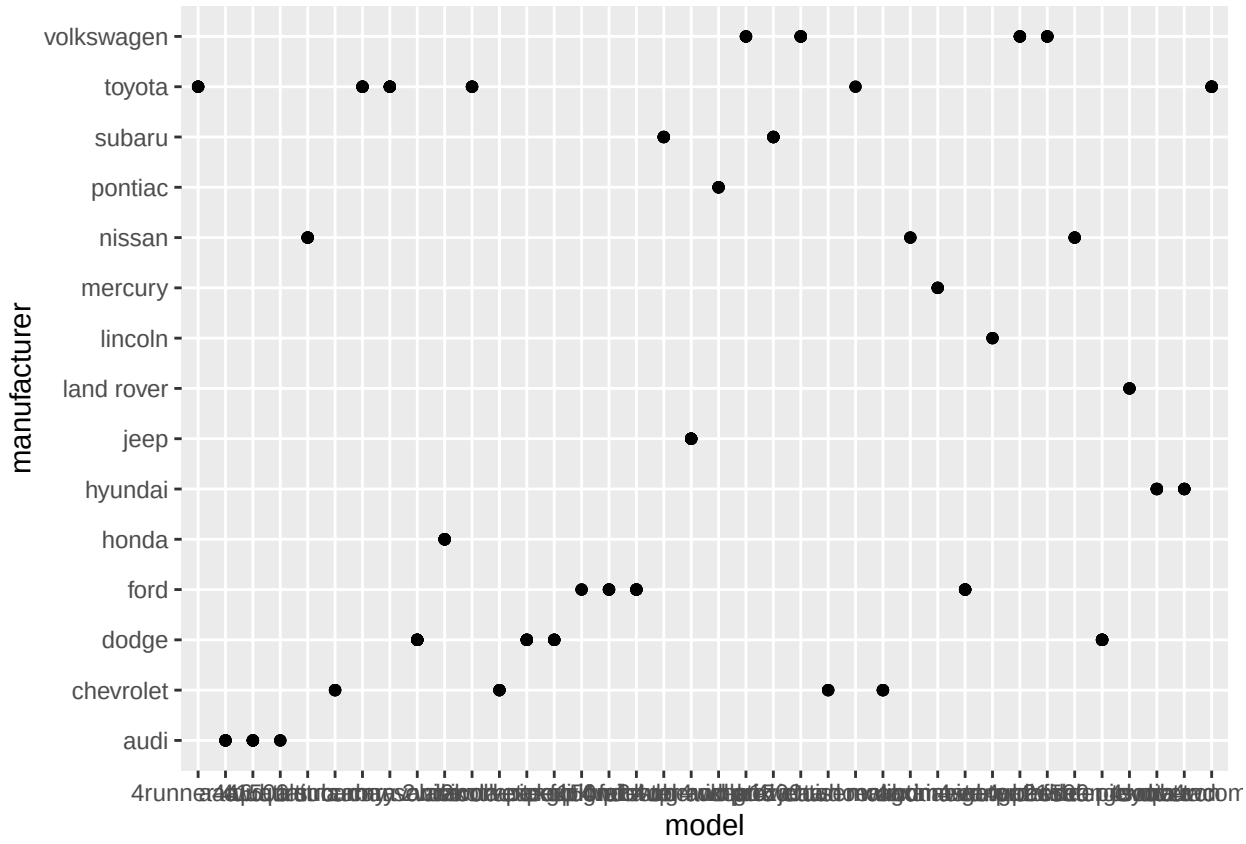
```
##
## Attaching package: 'ggplot2'
```

```
## The following object is masked _by_ '.GlobalEnv':
##
##     mpg
```

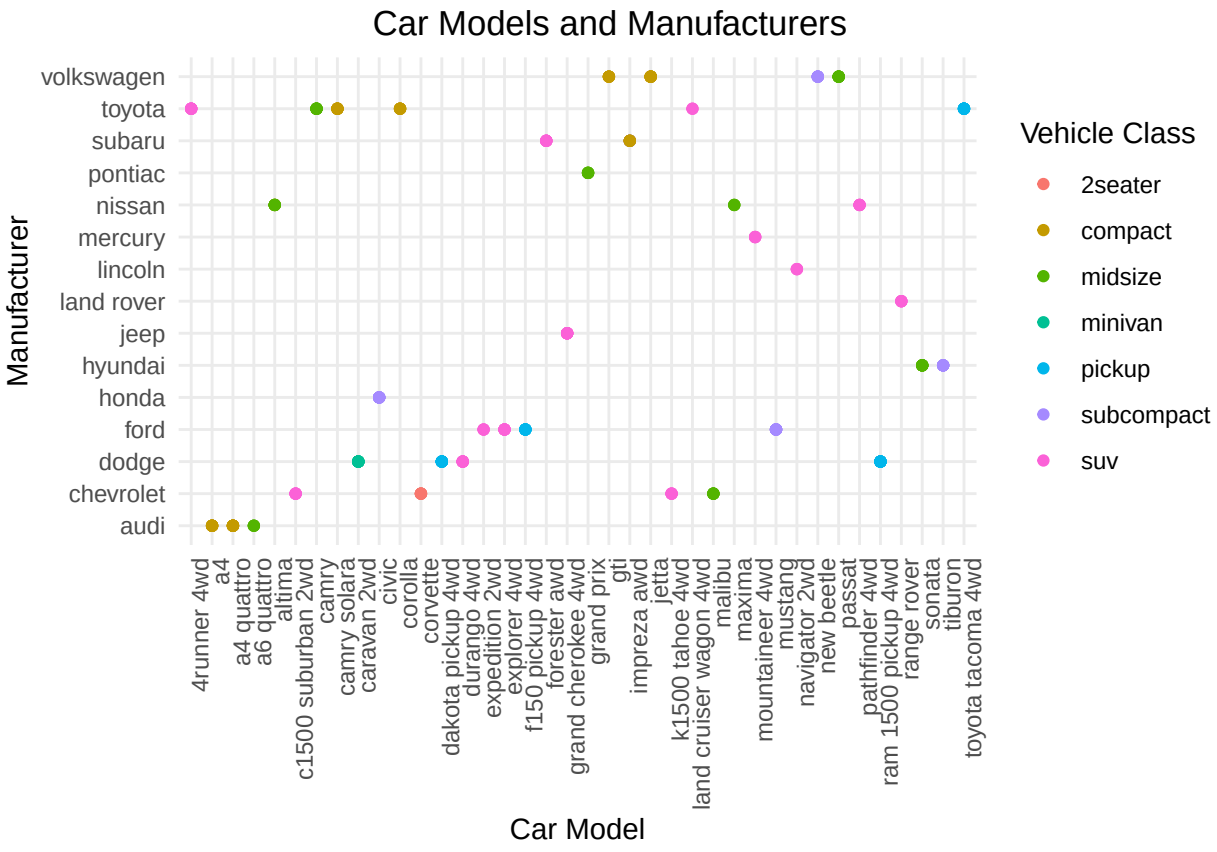
```
plot(ggplot(manumodel, aes(x = reorder(manufacturer, -unique_models), y = unique_models, fill = unique_
 geom_bar(stat = "identity", width = 0.8) +
 labs(title = "Number of Manufacturer's Unique Models",
      x = "Manufacturer",
      y = "Number of Unique Models") +

 theme_minimal() +
 scale_fill_gradient(low = "black", high = "pink") +
 theme(axis.text.x = element_text(angle = 90, vjust = 0.5, hjust=1)))
```




```
#2.2 A What does ggplot(mpg, aes(model, manufacturer)) + geom_point() show?

#This plot displays the relationship between car models on the x-axis and manufacturers (on the y-axis)

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




```
#2B.b. For you, is it useful? If not, how could you modify the data to make it more informative?

#To enhance its informativeness, I plan to introduce color coding to distinguish points according to va

ggplot(mpg, aes(x = model, y = manufacturer, color = class)) +
  geom_point() +
  labs(title = "Car Models and Manufacturers",
       cex = 3,
       x = "Car Model",
       y = "Manufacturer",
       color = "Vehicle Class") +
  theme_minimal() +
  theme(legend.position = "right", axis.text.x = element_text(angle = 90, hjust = 1),
  plot.title = element_text(hjust  = 0.6))
```




```
#3. Plot the model and the year using ggplot(). Use only the top 20 observations. Write the codes and i
```

```
library(ggplot2)
library(dplyr)

data(mpg)


mean_displ_df <- mpg %>%
 group_by(year, model) %>%
 summarise(mean_displ = mean(displ)) %>%
 arrange(desc(mean_displ)) %>%
 filter(row_number() < 20)
```

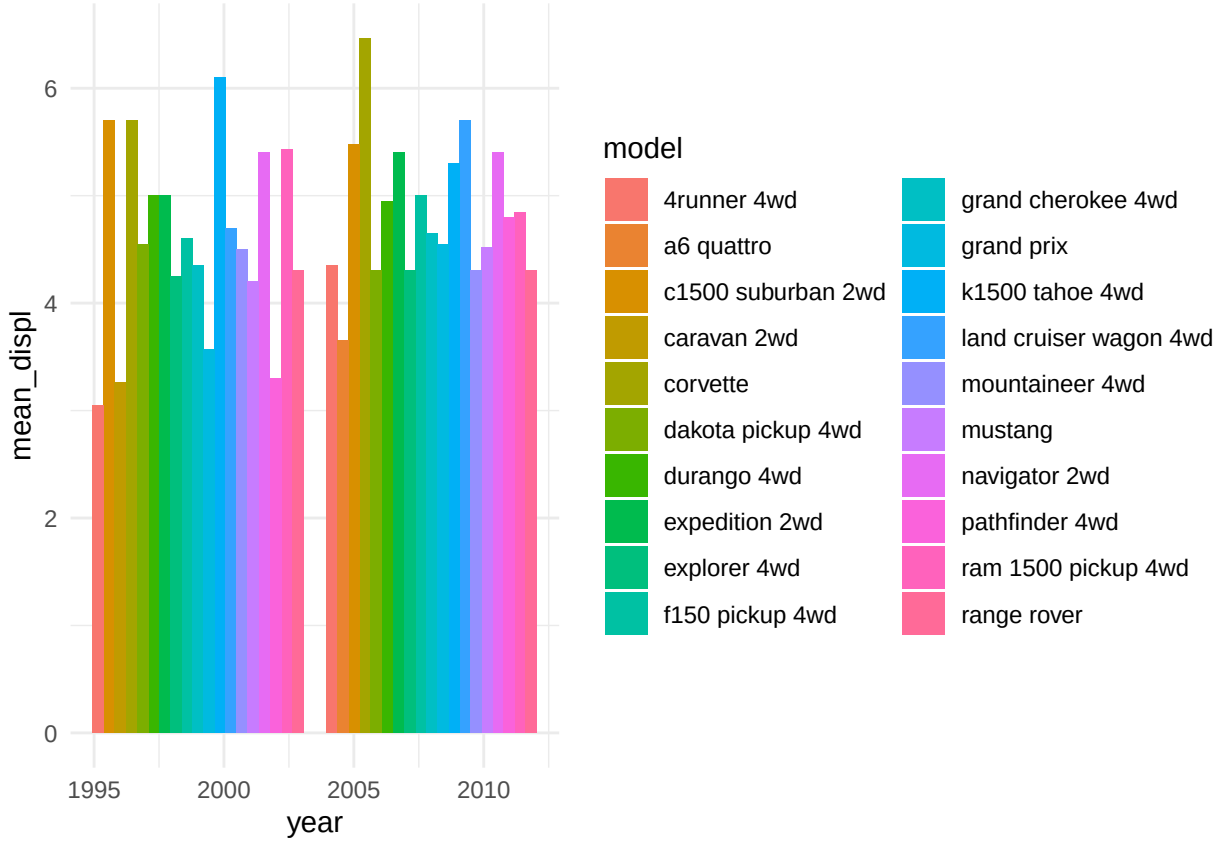
```
## `summarise()` has grouped output by 'year'. You can override using the
## `.groups` argument.
```

```
plot <- ggplot(mean_displ_df, aes(x = year, y = mean_displ, fill = model)) +
 geom_bar(stat = "identity", position = "dodge") +
 theme_minimal() +
 guides(fill = guide_legend(ncol = 2))
 labs(title = "Average Engine Displacement over the years for the top 20 models",
       x = "Year",
       y = "Engine Displacement",
       fill = "Model")
```

```
## $x
## [1] "Year"
```

```
##
## $y
## [1] "Engine Displacement"
##
## $fill
## [1] "Model"
##
## $title
## [1] "Average Engine Displacement over the years for the top 20 models"
##
## attr(,"class")
## [1] "labels"
```

```
print(plot)
```



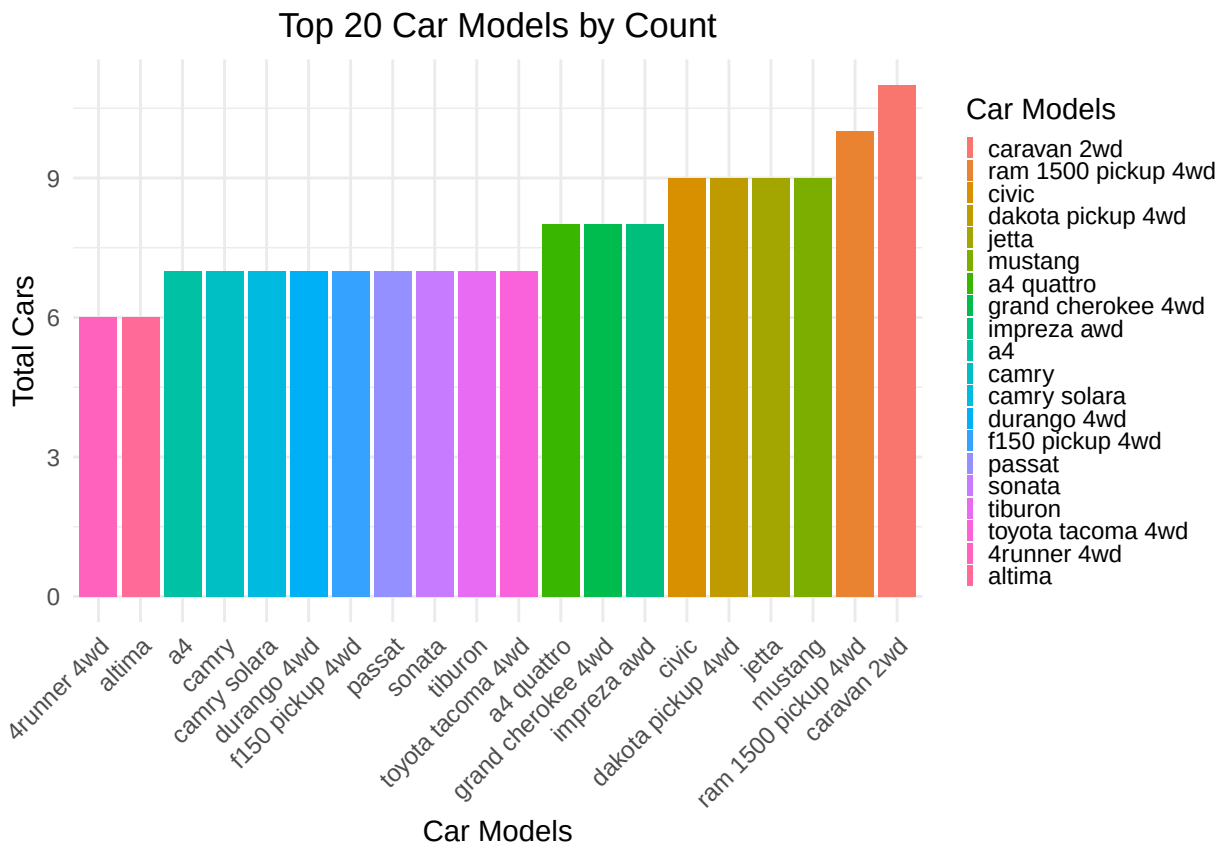

```
#4.  Using the pipe (%>%), group the model and get the number of cars per model. Show codes and its res
```

```
library(dplyr)
data(mpg)


countmodel <- mpg %>%
  group_by(model) %>%
  summarise(num_cars = n())


print(countmodel)
```

```
## # A tibble: 38 x 2
##    model               num_cars
##    <chr>                  <int>
##  1 4runner 4wd                6
##  2 a4                         7
##  3 a4 quattro                 8
##  4 a6 quattro                 3
##  5 altima                     6
##  6 c1500 suburban 2wd         5
##  7 camry                      7
##  8 camry solara               7
##  9 caravan 2wd               11
## 10 civic                      9
## # i 28 more rows
```

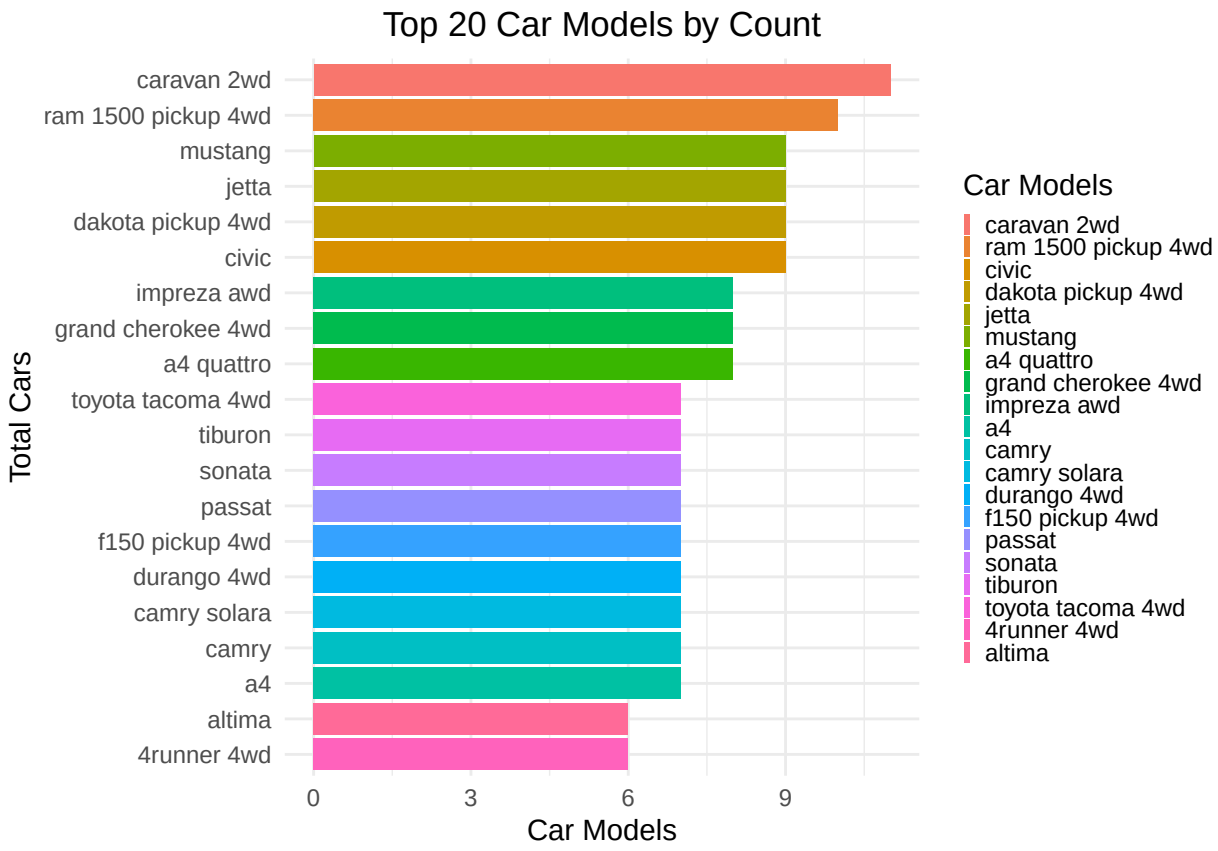
```
#4A. Plot using geom_bar() using the top 20 observations only. The graphs should have a title, labels a
```

```
library(ggplot2)
library(dplyr)

data(mpg)

summary_data <- mpg %>%
  count(model) %>%
  arrange(desc(n)) %>%
  slice(1:20)

top_models <- summary_data$model
palette <- scales::hue_pal()(length(top_models))

summary_data <- summary_data %>%
  mutate(color = palette[match(model, top_models)])

ggplot(summary_data, aes(x = reorder(model, n), y = n, fill = model)) +
  geom_bar(stat = "identity") +
  labs(
    title = "Top 20 Car Models by Count",
    x = "Car Models",
    y = "Total Cars"
  ) +
  scale_fill_manual(values = palette, name = "Car Models", breaks = summary_data$model) +
  theme_minimal() +
  theme(
    axis.text.x = element_text(angle = 45, hjust = 1),
    legend.key.size = unit(0.1, "cm"),
    plot.title = element_text(hjust = 0.5)
  )
```

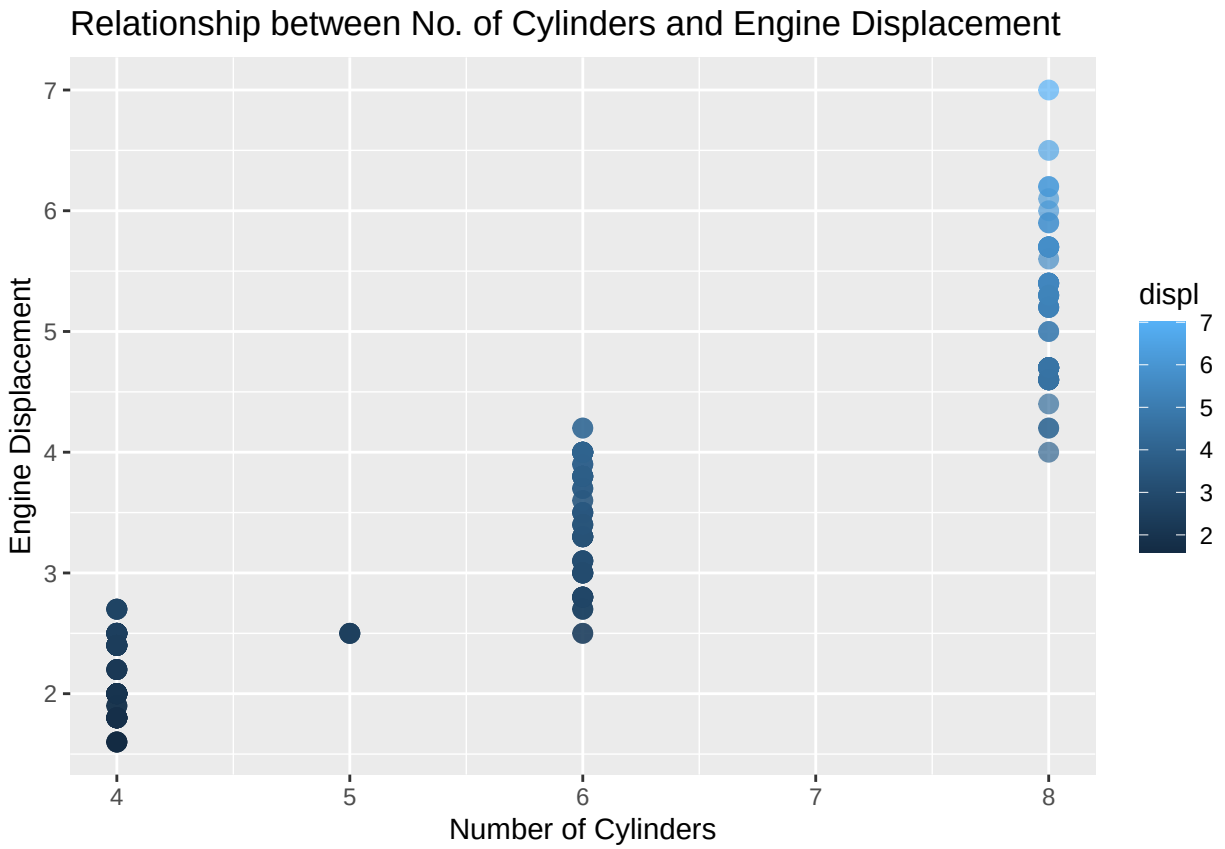



```
#4B. Plot using the geom_bar() + coord_flip() just like what is shown below. Show codes and its result.


library(ggplot2)
library(dplyr)

data(mpg)

summary_data <- mpg %>%
  count(model) %>%
  arrange(desc(n)) %>%
  slice(1:20)


top_models <- summary_data$model
palette <- scales::hue_pal()(length(top_models))

summary_data <- summary_data %>%
  mutate(color = palette[match(model, top_models)])

ggplot(summary_data, aes(x = reorder(model, n), y = n, fill = model)) +
  geom_bar(stat = "identity") +
  labs(
    title = "Top 20 Car Models by Count",
    y = "Car Models",
    x = "Total Cars"
  ) +
```

```
  scale_fill_manual(values = palette, name = "Car Models", breaks = summary_data$model) +
  coord_flip() +
  theme_minimal() +
  theme(
    legend.key.size = unit(0.1, "cm"),
    plot.title = element_text(hjust = 0.5)
  )
```

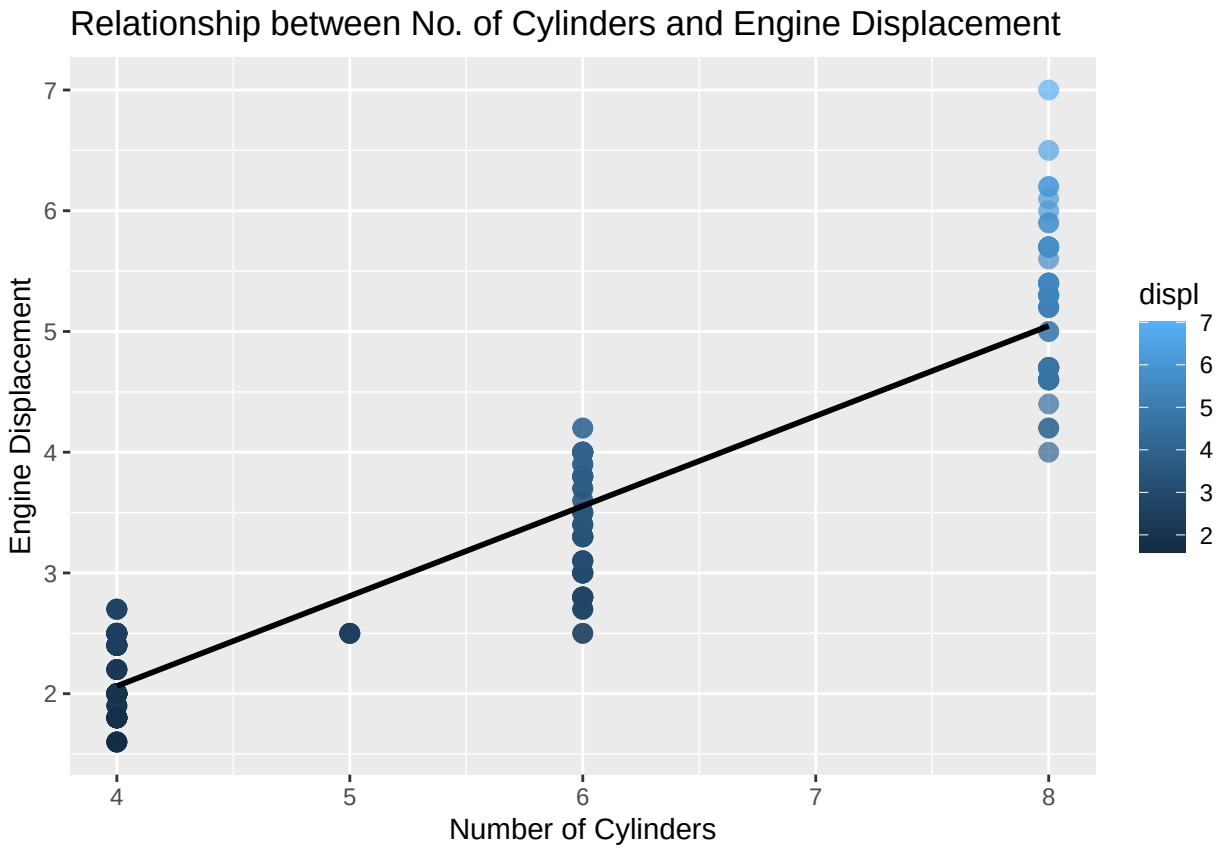



```
#5. Plot the relationship between cyl - number of cylinders and displ - engine displacement using#geom_

#No. of Cylinders and Engine Displacement".

library(ggplot2)
library(dplyr)

data(mpg)

ggplot(mpg, aes(x = cyl, y = displ, color = displ)) +
  geom_point(size = 3, alpha = 0.7) +
  labs(
    title = "Relationship between No. of Cylinders and Engine Displacement",
    x = "Number of Cylinders",
    y = "Engine Displacement")
```

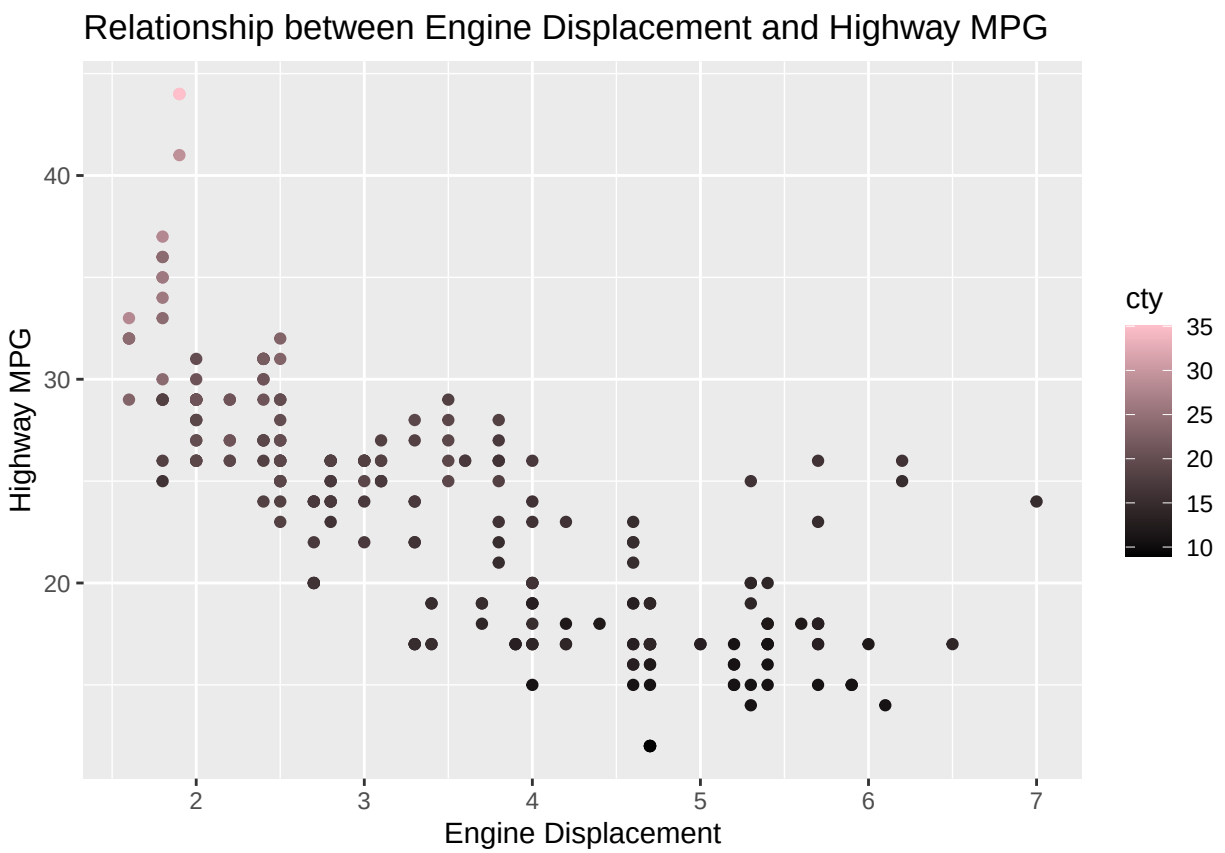



```
#5A. How would you describe its relationship? Show the codes and its result.


#Utilizing a regression line, showcase the correlation between the number of cylinders and engine displ
#As the cylinder count rises, there's a tendency for the engine size to also increase.

library(ggplot2)
library(dplyr)

data(mpg)

ggplot(mpg, aes(x = cyl, y = displ, color = displ)) +
  geom_point(size = 3, alpha = 0.7) +
  geom_smooth(method = "lm", se = FALSE, color = "black") +
  labs(
    title = "Relationship between No. of Cylinders and Engine Displacement",
    x = "Number of Cylinders",
    y = "Engine Displacement"
  )
```

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




```
#6. Plot the relationship between displ (engine displacement) and hwy(highway miles per gallon). Mapped
#it with a continuous variable you have identified in #1-c. What is its result? Why it produced such
#output?

library(ggplot2)
library(dplyr)

data(mpg)

ggplot(mpg, aes(x = displ, y = hwy, color = cty)) +
  geom_point() +
  labs(
    title = "Relationship between Engine Displacement and Highway MPG",
    x = "Engine Displacement",
    y = "Highway MPG"
  ) +
  scale_color_gradient(low = "black", high = "pink")
```




The scatter plot illustrates the relationship between engine displacement (displ) and highway miles per gallon (hwy). It employs a color gradient derived from city miles per gallon (cty) to represent a continuous variable across the plotted points. The color gradient, reflective of city miles per gallon (cty), doesn't directly correlate with engine displacement or highway miles per gallon (displ and hwy). Rather, it showcases the diversity in city MPG across the entire scatter plot.

```
#6. Import the traffic.csv onto your R environment.
#A. How many numbers of observation does it have? What are the variables of the traffic dataset the Sho
#your answer.

traffic <- read_csv("traffic.csv")
```

```
## Rows: 48120 Columns: 4
## -- Column specification --------------------------------------------------------
## Delimiter: ","
## dbl  (3): Junction, Vehicles, ID
## dttm (1): DateTime
##
## i Use `spec()` to retrieve the full column specification for this data.
## i Specify the column types or set `show_col_types = FALSE` to quiet this message.
```

```
head(traffic)
```

```
## # A tibble: 6 x 4
##   DateTime            Junction Vehicles          ID
##   <dttm>                 <dbl>    <dbl>       <dbl>
## 1 2015-11-01 00:00:00        1       15 20151101001
## 2 2015-11-01 01:00:00        1       13 20151101011
## 3 2015-11-01 02:00:00        1       10 20151101021
```

```
## 4 2015-11-01 03:00:00        1        7 20151101031
## 5 2015-11-01 04:00:00        1        9 20151101041
## 6 2015-11-01 05:00:00        1        6 20151101051
```

```
observations <- nrow(traffic)
variables <- names(traffic)

cat("Number of observations:", observations, "\n")
```

```
## Number of observations: 48120
```

```
cat("The variables are:", variables, "\n")
```

```
## The variables are: DateTime Junction Vehicles ID
```

```
#6B. subset the traffic dataset into junctions. What is the R codes and its output?

junctions1 <- subset(traffic, Junction == 1)
junctions2 <- subset(traffic, Junction == 2)
junctions3 <- subset(traffic, Junction == 3)
junctions4 <- subset(traffic, Junction == 4)

#The output are:
junctions1
```

```
## # A tibble: 14,592 x 4
##    DateTime            Junction Vehicles          ID
##    <dttm>                 <dbl>    <dbl>       <dbl>
##  1 2015-11-01 00:00:00        1       15 20151101001
##  2 2015-11-01 01:00:00        1       13 20151101011
##  3 2015-11-01 02:00:00        1       10 20151101021
##  4 2015-11-01 03:00:00        1        7 20151101031
##  5 2015-11-01 04:00:00        1        9 20151101041
##  6 2015-11-01 05:00:00        1        6 20151101051
##  7 2015-11-01 06:00:00        1        9 20151101061
##  8 2015-11-01 07:00:00        1        8 20151101071
##  9 2015-11-01 08:00:00        1       11 20151101081
## 10 2015-11-01 09:00:00        1       12 20151101091
## # i 14,582 more rows
```

```
junctions2
```

```
## # A tibble: 14,592 x 4
##    DateTime            Junction Vehicles          ID
##    <dttm>                 <dbl>    <dbl>       <dbl>
##  1 2015-11-01 00:00:00        2        6 20151101002
##  2 2015-11-01 01:00:00        2        6 20151101012
##  3 2015-11-01 02:00:00        2        5 20151101022
##  4 2015-11-01 03:00:00        2        6 20151101032
##  5 2015-11-01 04:00:00        2        7 20151101042
##  6 2015-11-01 05:00:00        2        2 20151101052
##  7 2015-11-01 06:00:00        2        4 20151101062
##  8 2015-11-01 07:00:00        2        4 20151101072
##  9 2015-11-01 08:00:00        2        3 20151101082
## 10 2015-11-01 09:00:00        2        3 20151101092
## # i 14,582 more rows
```

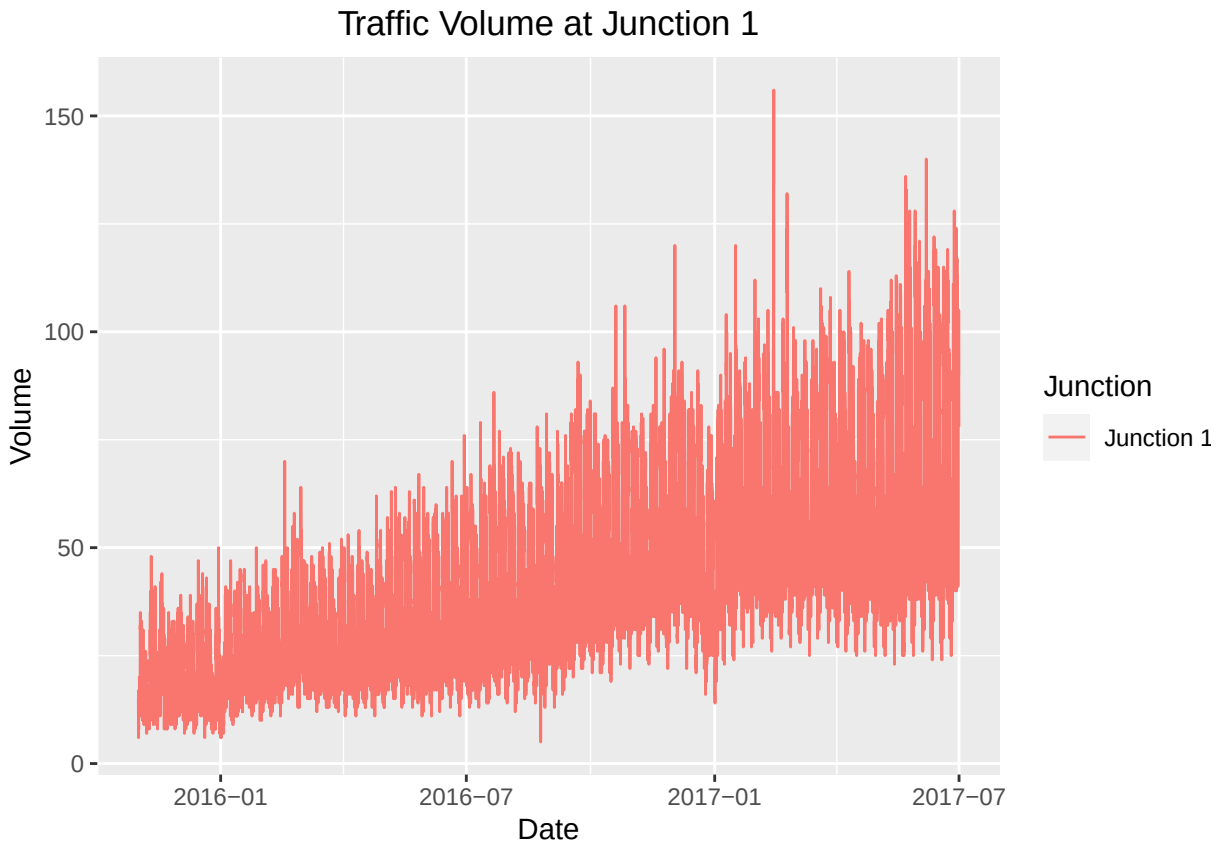
```
junctions3
```

```
## # A tibble: 14,592 x 4
##    DateTime            Junction Vehicles          ID
##    <dttm>                 <dbl>    <dbl>       <dbl>
##  1 2015-11-01 00:00:00        3        9 20151101003
##  2 2015-11-01 01:00:00        3        7 20151101013
##  3 2015-11-01 02:00:00        3        5 20151101023
##  4 2015-11-01 03:00:00        3        1 20151101033
##  5 2015-11-01 04:00:00        3        2 20151101043
##  6 2015-11-01 05:00:00        3        2 20151101053
##  7 2015-11-01 06:00:00        3        3 20151101063
##  8 2015-11-01 07:00:00        3        4 20151101073
##  9 2015-11-01 08:00:00        3        3 20151101083
## 10 2015-11-01 09:00:00        3        6 20151101093
## # i 14,582 more rows
```

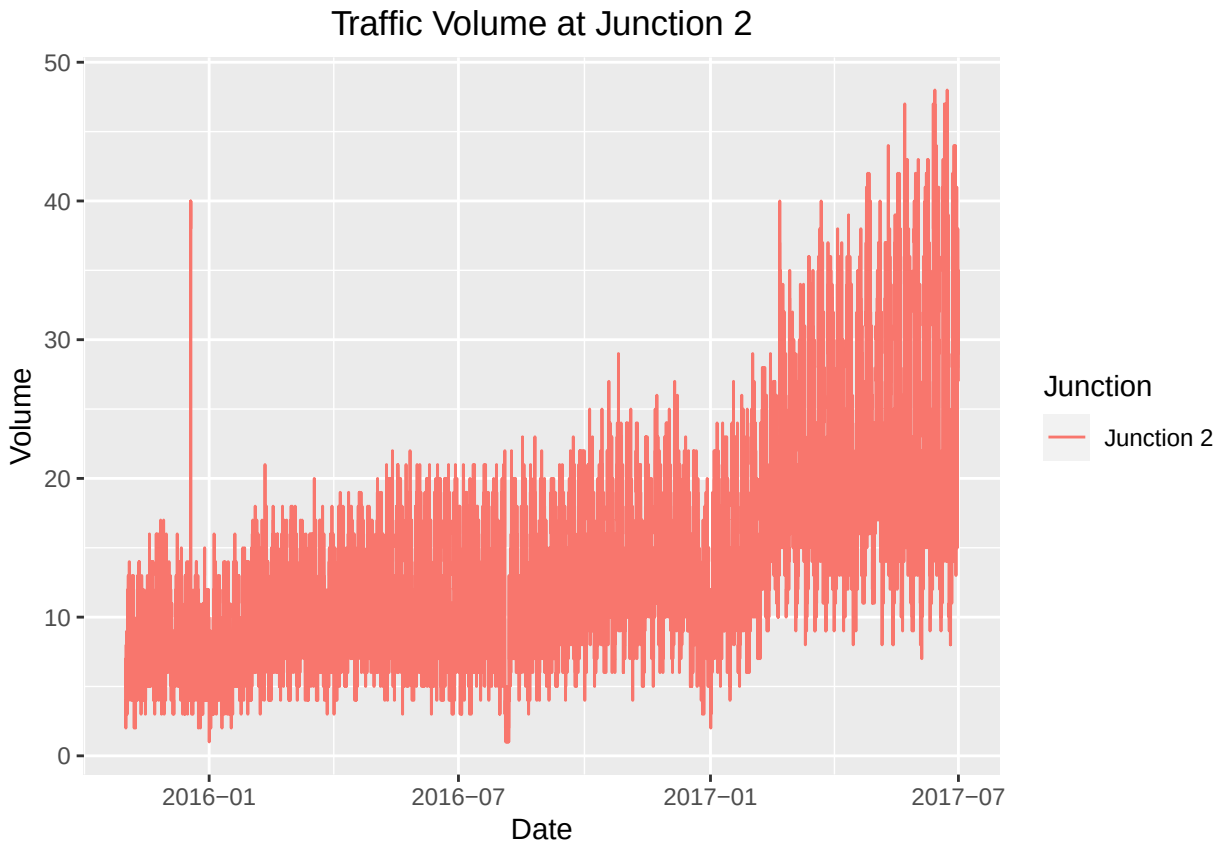
```
junctions4
```

```
## # A tibble: 4,344 x 4
##    DateTime            Junction Vehicles          ID
##    <dttm>                 <dbl>    <dbl>       <dbl>
##  1 2017-01-01 00:00:00        4        3 20170101004
##  2 2017-01-01 01:00:00        4        1 20170101014
##  3 2017-01-01 02:00:00        4        4 20170101024
##  4 2017-01-01 03:00:00        4        4 20170101034
##  5 2017-01-01 04:00:00        4        2 20170101044
##  6 2017-01-01 05:00:00        4        1 20170101054
##  7 2017-01-01 06:00:00        4        1 20170101064
##  8 2017-01-01 07:00:00        4        4 20170101074
##  9 2017-01-01 08:00:00        4        4 20170101084
## 10 2017-01-01 09:00:00        4        2 20170101094
## # i 4,334 more rows
```

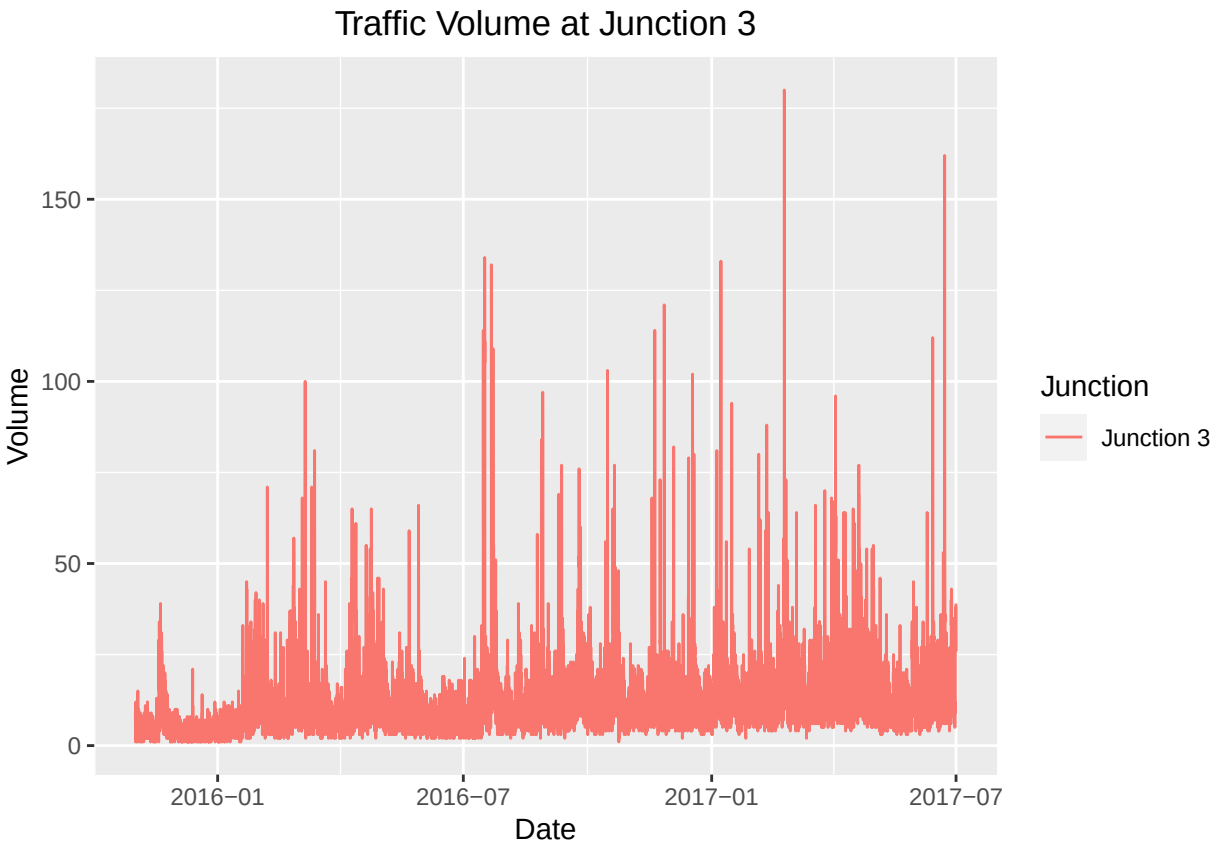
```
#6C. Plot each junction in a using geom_line(). Show your solution and output.

#Junction 1

ggplot(junctions1, aes(x = DateTime, y = Vehicles, color = "Junction 1")) +
  geom_line() +
  labs(
    title = "Traffic Volume at Junction 1",
    x = "Date",
    y = "Volume"
  ) +
  scale_color_discrete(name = "Junction") +
 theme(plot.title = element_text(hjust = 0.5))
```



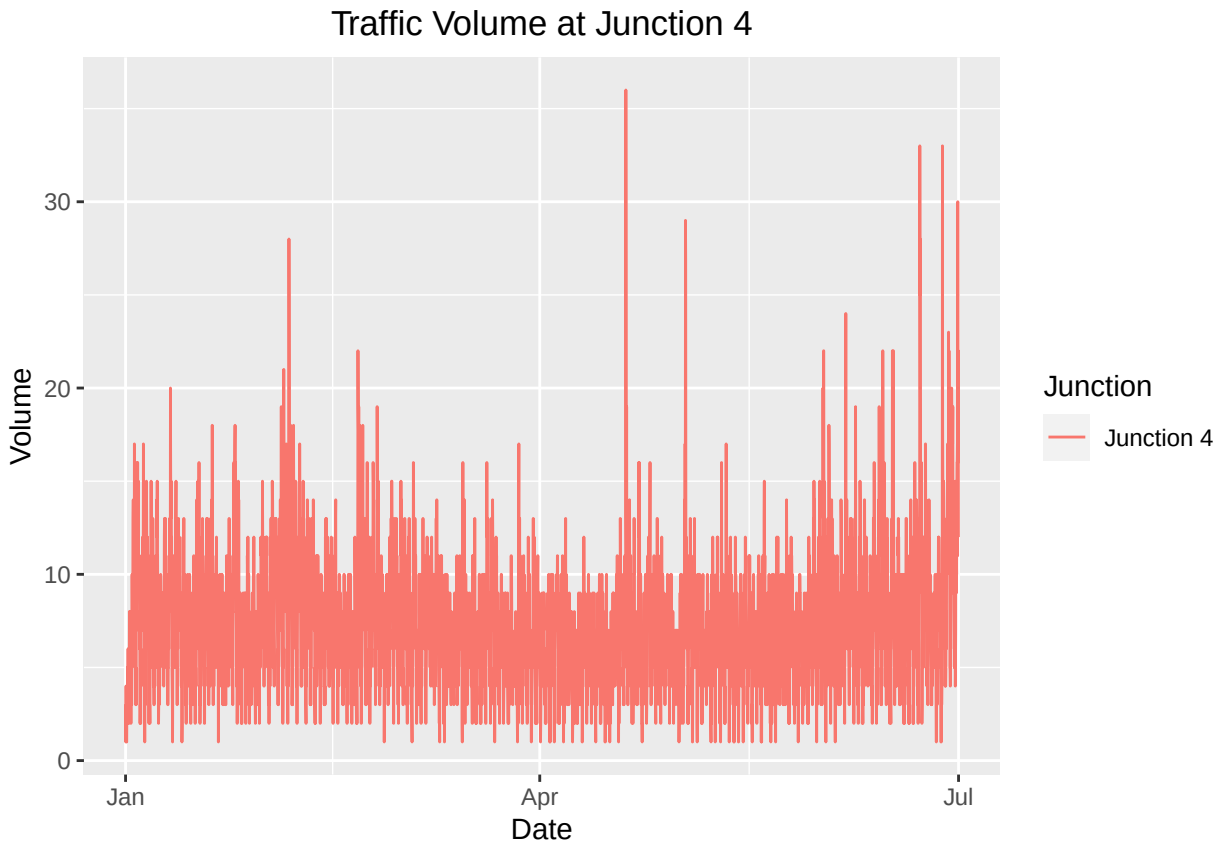

```
#Junction 2

ggplot(junctions2, aes(x = DateTime, y = Vehicles, color = "Junction 2")) +
  geom_line() +
  labs(
    title = "Traffic Volume at Junction 2",
    x = "Date",
    y = "Volume"
  ) +
  scale_color_discrete(name = "Junction") +
  theme(plot.title = element_text(hjust = 0.5))
```




```
#Junction 3

ggplot(junctions3, aes(x = DateTime, y = Vehicles, color = "Junction 3")) +
  geom_line() +
  labs(
    title = "Traffic Volume at Junction 3",
    x = "Date",
    y = "Volume"
  ) +
  scale_color_discrete(name = "Junction") +
 theme(plot.title = element_text(hjust = 0.5))
```




```
#Junction 4

ggplot(junctions4, aes(x = DateTime, y = Vehicles, color = "Junction 4")) +
  geom_line() +
  labs(
    title = "Traffic Volume at Junction 4",
    x = "Date",
    y = "Volume"
  ) +
  scale_color_discrete(name = "Junction") +
  theme(plot.title = element_text(hjust = 0.5))
```




```
#7. From alexa_file.xlsx, import it to your environment

library(readxl)
alexa_file <- read_excel("alexa_file.xlsx")
head(alexa_file)
```

```
## # A tibble: 6 x 5
##   rating date                variation          verified_reviews       feedback
##    <dbl> <dttm>              <chr>              <chr>                     <dbl>
## 1      5 2018-07-31 00:00:00 Charcoal Fabric    Love my Echo!                 1
## 2      5 2018-07-31 00:00:00 Charcoal Fabric    Loved it!                     1
## 3      4 2018-07-31 00:00:00 Walnut Finish      Sometimes while playi~        1
## 4      5 2018-07-31 00:00:00 Charcoal Fabric    I have had a lot of f~        1
## 5      5 2018-07-31 00:00:00 Charcoal Fabric    Music                         1
## 6      5 2018-07-31 00:00:00 Heather Gray Fabric I received the echo a~       1
```

```
#7A. How many observations does alexa_file has? What about the number of columns? Show your solution an

observations <- nrow(alexa_file)
columns <- ncol(alexa_file)

cat("Number of observations:", observations, "\n")
```

```
## Number of observations: 3150
```

```
cat("Number of columns:", columns, "\n")
```

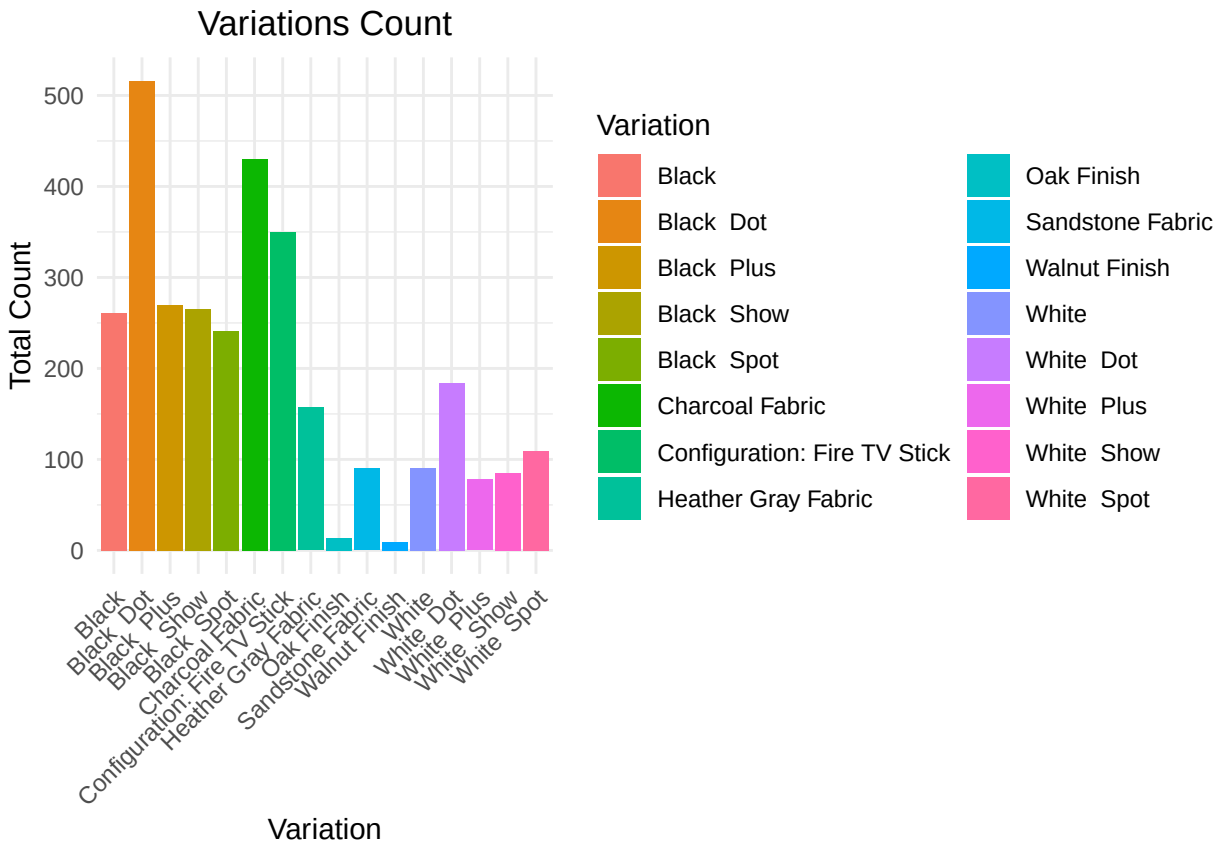
```
## Number of columns: 5
```

```r
#The number of observations is 3,150 and The number of columns is 5.
```

```r
#7B. group the variations and get the total of each variations. Use dplyr package. Show solution and an

library(dplyr)

result <- alexa_file %>%
  group_by(variation) %>%
  summarise(total_variations = n())

print(result)
```

```
## # A tibble: 16 x 2
##    variation                    total_variations
##    <chr>                                   <int>
##  1 Black                                     261
##  2 Black  Dot                               516
##  3 Black  Plus                              270
##  4 Black  Show                              265
##  5 Black  Spot                              241
##  6 Charcoal Fabric                          430
##  7 Configuration: Fire TV Stick             350
##  8 Heather Gray Fabric                      157
##  9 Oak Finish                                14
## 10 Sandstone Fabric                          90
## 11 Walnut Finish                              9
## 12 White                                     91
## 13 White  Dot                               184
## 14 White  Plus                               78
## 15 White  Show                               85
## 16 White  Spot                              109
```

```r
#7C. Plot the variations using the ggplot() function. What did you observe? Complete the details of the


library(ggplot2)

var <- ggplot(result, aes(x = variation, y = total_variations, fill = variation)) +
  geom_bar(stat = "identity") +
  labs(title = "Variations Count",
       x = "Variation",
       y = "Total Count") +
  theme_minimal() +
  theme(axis.text.x = element_text(angle = 45, hjust = 1)) +
  scale_fill_discrete(name = "Variation") +
  guides(fill = guide_legend(ncol = 2)) +
  theme(plot.title = element_text(hjust = 0.5))

print(var)
```

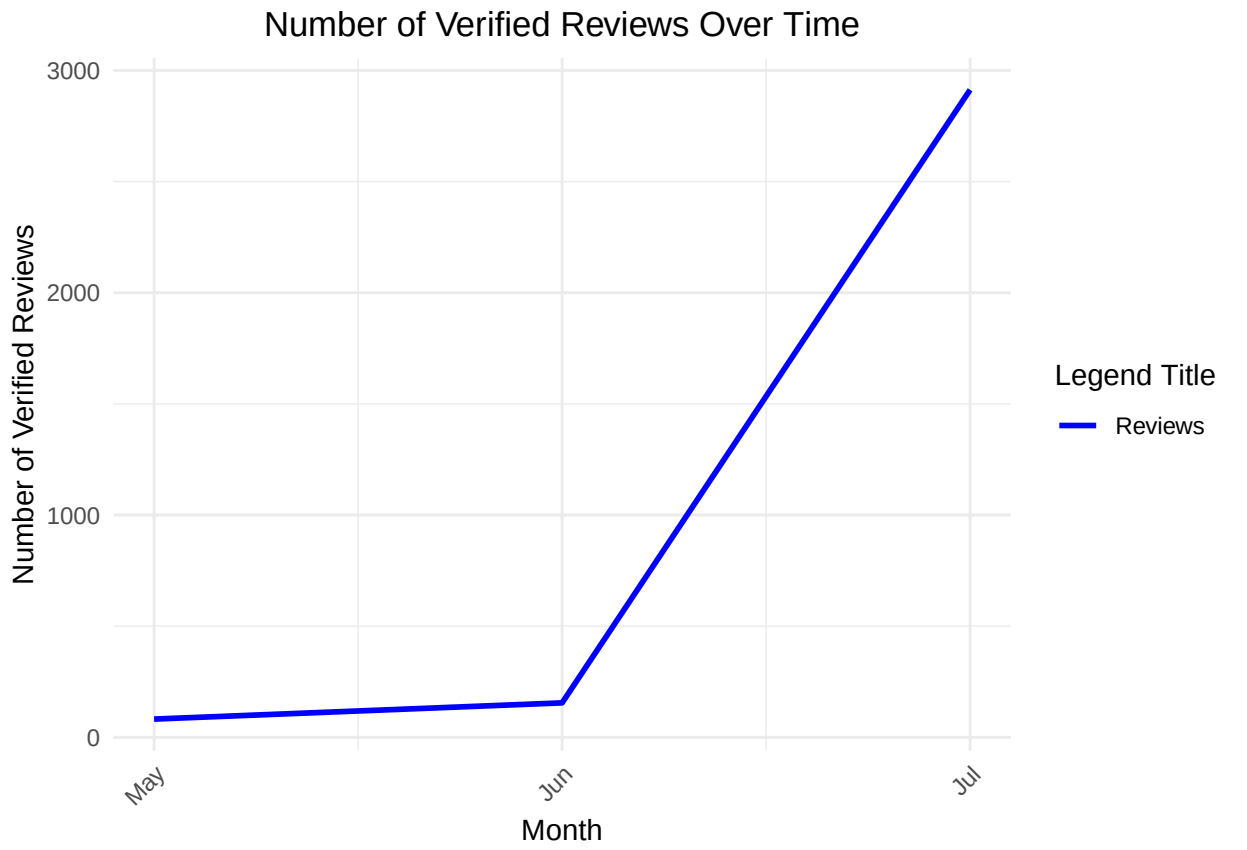



This visualization represents the different variations of the Alexa File, utilizing distinct colors to label each variation, aiding viewers in analysis. Additionally, it includes the total count for each variation. The variation labeled "Black Dot" notably appears more frequently than the others. The legend, conveniently organized into two columns, provides a clear reference to associate each color with its respective type of variation.

```
#7D. Plot a geom_line() with the date and the number of verified reviews. Complete the details of the g


library(dplyr)
library(ggplot2)

alexa_file$date <- as.Date(alexa_file$date)
alexa_file$month <- format(alexa_file$date, "%m")

monthcount <- alexa_file %>%
   count(month)

 p <- ggplot(monthcount, aes(x = as.integer(month), y = n, color = "Reviews")) +
  geom_line(size = 1) +
  labs(title = "Number of Verified Reviews Over Time",
       x = "Month",
       y = "Number of Verified Reviews",
       color = "Legend Title") +  # Change legend title
  scale_x_continuous(breaks = 1:12, labels = month.abb) +
  scale_color_manual(values = c("blue"), labels = c("Reviews")) +
  theme_minimal() +
  theme(plot.title = element_text(hjust = 0.5),
        axis.text.x = element_text(angle = 45, hjust = 1))
```

```
## Warning: Using `size` aesthetic for lines was deprecated in ggplot2 3.4.0.
## i Please use `linewidth` instead.
## This warning is displayed once every 8 hours.
## Call `lifecycle::last_lifecycle_warnings()` to see where this warning was
## generated.
```

```
print(p)
```




```
#7E. Get the relationship of variations and ratings. Which variations got the most highest in rating? P

library(dplyr)
library(ggplot2)

variation_ratings <- alexa_file %>%
  group_by(variation) %>%
  summarize(avg_rating = mean(rating))
print(variation_ratings)
```

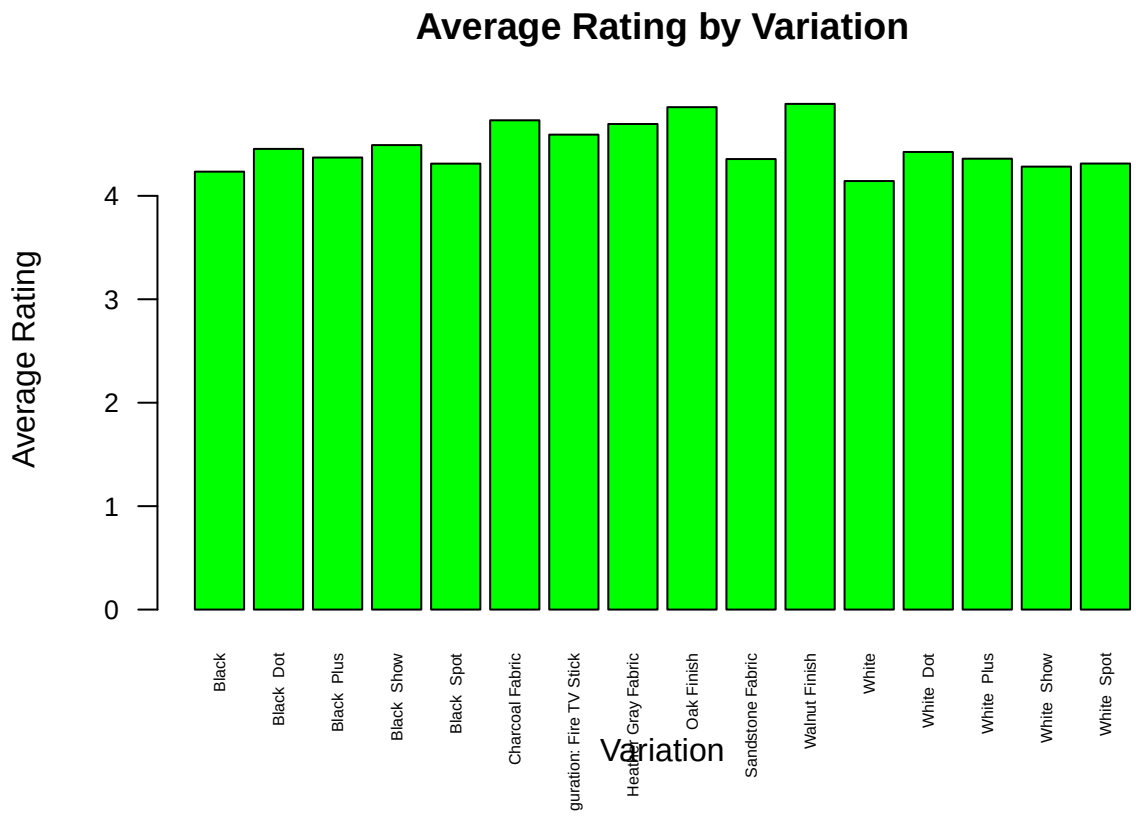
```
## # A tibble: 16 x 2
##    variation                    avg_rating
##    <chr>                             <dbl>
##  1 Black                              4.23
##  2 Black  Dot                         4.45
##  3 Black  Plus                        4.37
##  4 Black  Show                        4.49
##  5 Black  Spot                        4.31
##  6 Charcoal Fabric                    4.73
##  7 Configuration: Fire TV Stick       4.59
```

```
##  8 Heather Gray Fabric               4.69
##  9 Oak Finish                       4.86
## 10 Sandstone Fabric                 4.36
## 11 Walnut Finish                    4.89
## 12 White                            4.14
## 13 White  Dot                       4.42
## 14 White  Plus                      4.36
## 15 White  Show                      4.28
## 16 White  Spot                      4.31
```

```r
highest <- variation_ratings %>%
  filter(avg_rating == max(avg_rating))
print(highest)
```

```
## # A tibble: 1 x 2
##   variation     avg_rating
##   <chr>              <dbl>
## 1 Walnut Finish       4.89
```

```r
variation_names <- variation_ratings$variation
average_ratings <- variation_ratings$avg_rating

barplot(average_ratings, names.arg = variation_names, col = "green",
        main = "Average Rating by Variation",
        xlab = "Variation", ylab = "Average Rating",
        cex.axis = 0.8, cex.names = 0.5, las = 2)
```




```r
top_variation <- variation_names[which.max(average_ratings)]
top_rating <- max(average_ratings)
```

```r
cat("The variation with the highest average rating is:", top_variation, "with an average rating of", to
```

```
## The variation with the highest average rating is: Walnut Finish with an average rating of 4.888889
```