RWorksheet_Condag#4c

Angel Blase

2024-11-02

1. Use the dataset mpg

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

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

b-c. Among the 11 variables in the mpg dataset from the ggplot2 package, the categorical variables include manufacturer, model, year, cyl, trans, drv, fl, and class. While the continuous variables are displ, hwy, and cty.

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

```
library(ggplot2)
```

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

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

```
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)

manufacturer_models <- summarise(group_by(mpg, manufacturer), unique_models = n_distinct(model))
most_models <- arrange(manufacturer_models, desc(unique_models))
most_models <- slice(most_models, 1)
print(most_models)
```

```
## # A tibble: 1 x 2
##   manufacturer unique_models
##   <chr>                <int>
## 1 toyota                   6
```

```
model_variations <- summarise(group_by(mpg, model), variations = n())
most_variations <- arrange(model_variations, desc(variations))
most_variations <- slice(most_variations, 1)
print(most_variations)
```

```
## # A tibble: 1 x 2
##   model       variations
##   <chr>            <int>
## 1 caravan 2wd         11
```

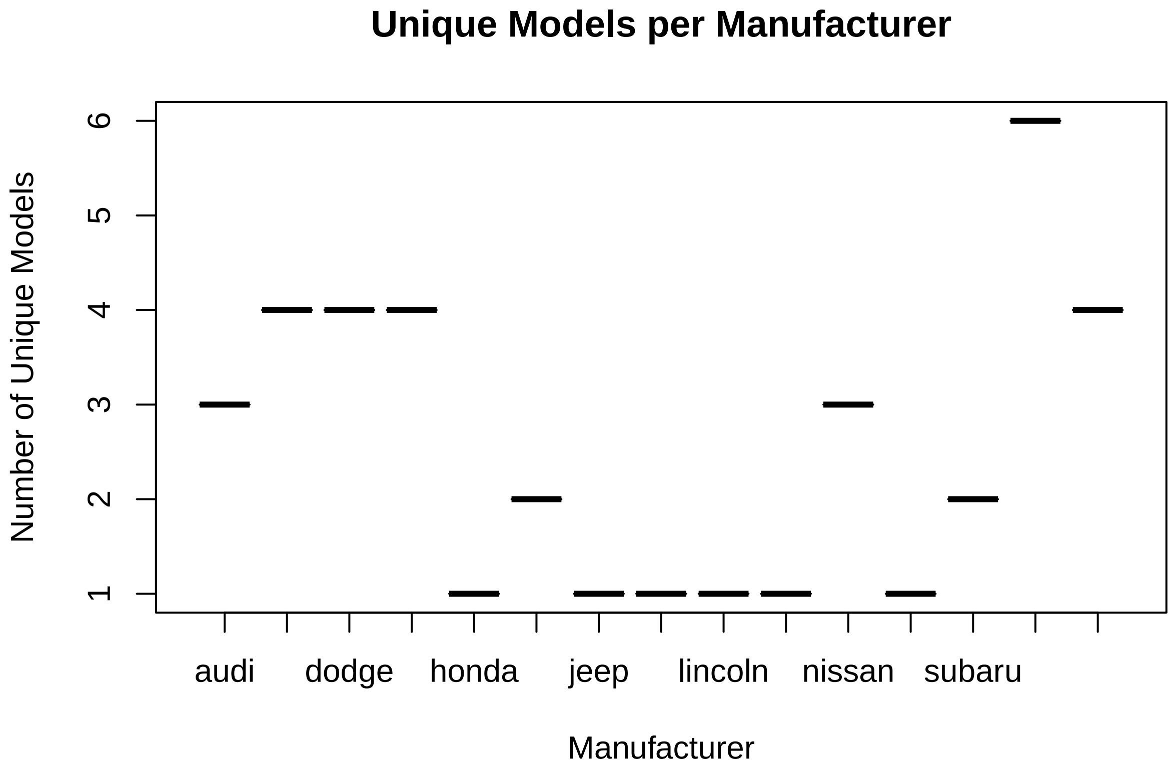
a. Group the manufacturers and find the unique models. Show your codes and result.

```
manufacturer_models <- summarise(group_by(mpg, manufacturer), unique_models = n_distinct(model))
manufacturer_models
```

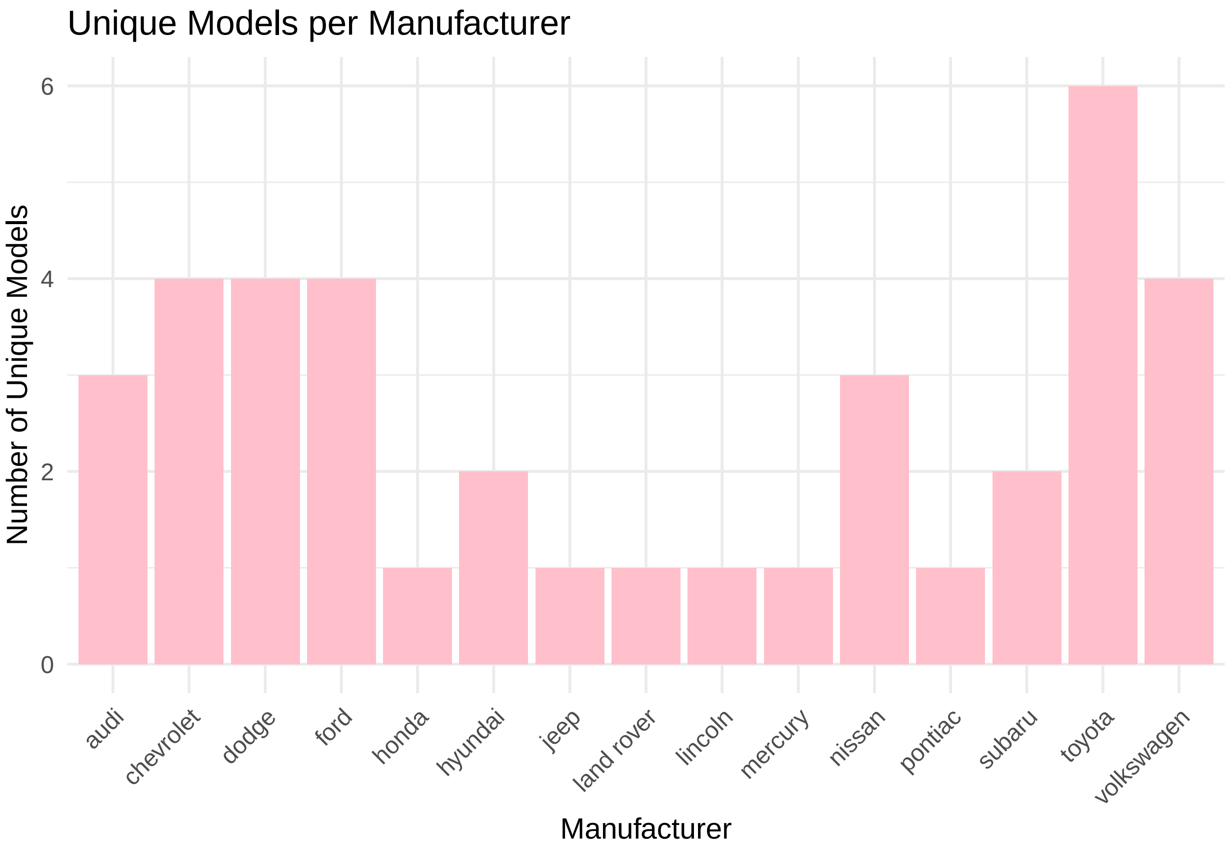
```
## # A tibble: 15 x 2
##    manufacturer unique_models
##    <chr>                <int>
##  1 audi                     3
##  2 chevrolet                4
##  3 dodge                    4
##  4 ford                     4
##  5 honda                    1
##  6 hyundai                  2
##  7 jeep                     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
```

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

```
# Using plot()

manufacturer_models$manufacturer <- as.factor(manufacturer_models$manufacturer)

plot(manufacturer_models$manufacturer, manufacturer_models$unique_models,
     main = "Unique Models per Manufacturer",
     xlab = "Manufacturer",
     ylab = "Number of Unique Models",
     col = "pink",
     pch = 19)
```



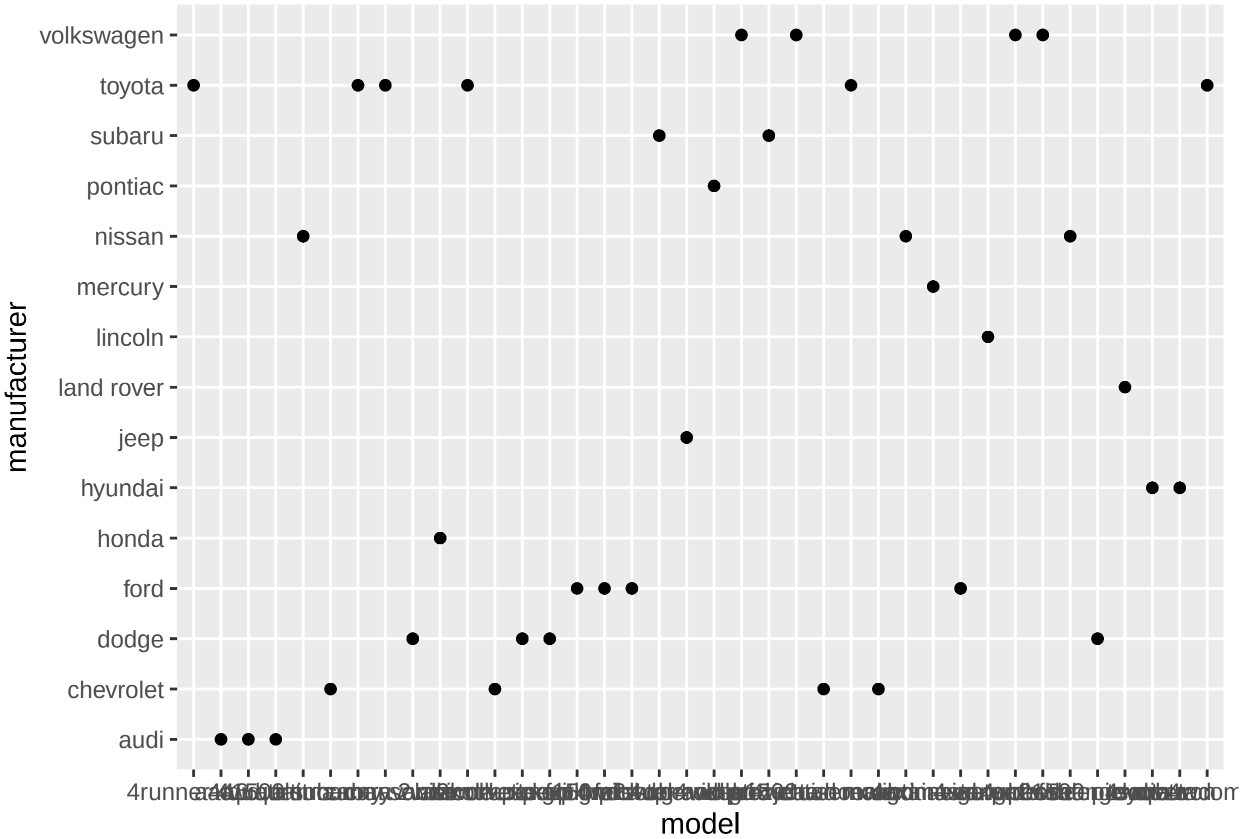
```r
# Using ggplot()

ggplot(manufacturer_models, aes(x = manufacturer, y = unique_models)) +
  geom_bar(stat = "identity", fill = "pink") +
  labs(title = "Unique Models per Manufacturer",
       x = "Manufacturer",
       y = "Number of Unique Models") +
  theme_minimal() +
  theme(axis.text.x = element_text(angle = 45, hjust = 1))
```



2. Same dataset will be used. You are going to show the relationship of the modeland the manufacturer.

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

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

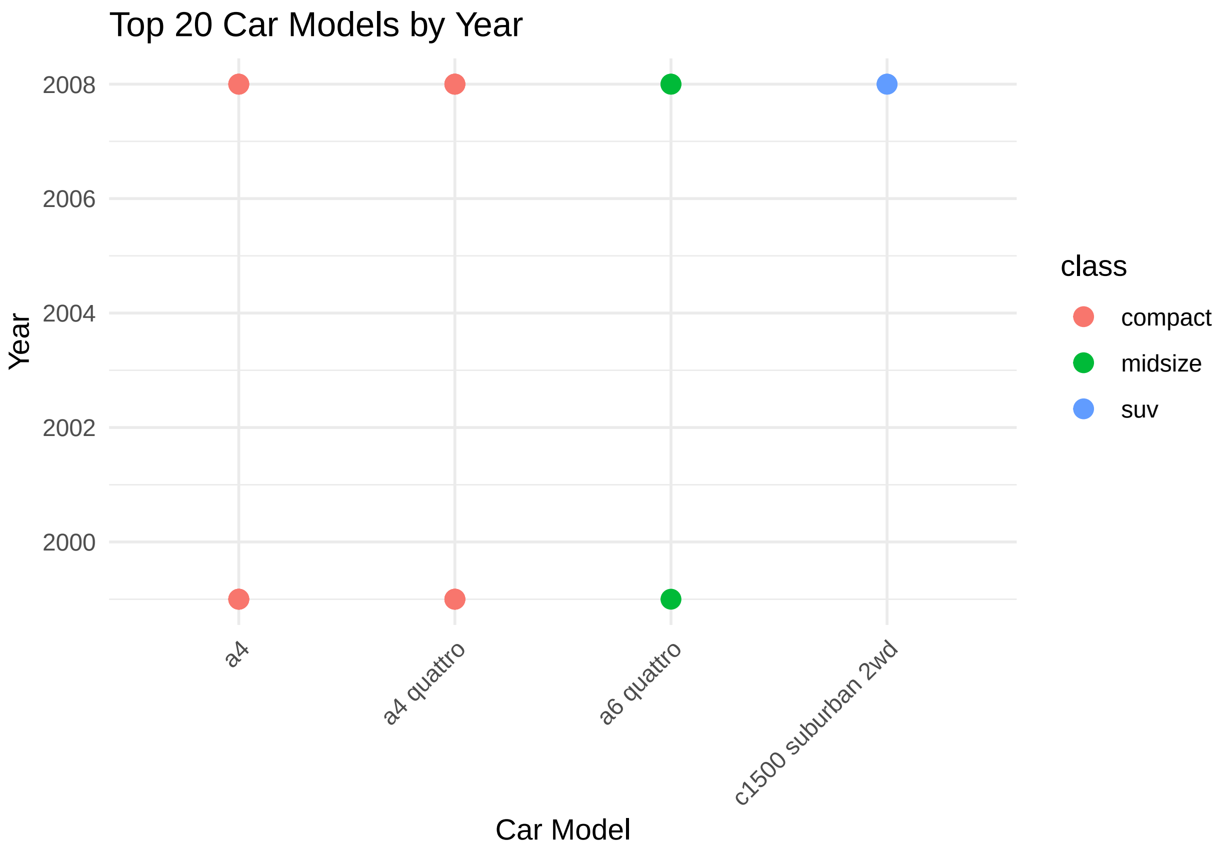


Using the code ggplot(mpg, aes(model, manufacturer)) + geom_point() creates a scatter plot with model on the x-axis and manufacturer on the y-axis. Each point on the plot represents a specific car model from the mpg dataset, plotted according to its associated manufacturer.

b. The initial scatter plot created with ggplot(mpg, aes(model, manufacturer)) + geom_point() is not very useful because it displays a lot of overlapping points, making it hard to distinguish between different car models and their manufacturers. This clutter can obscure patterns and relationships in the data. To make the plot more informative, it could be enhanced by adding color to represent another variable, like the car class, which would provide more context. Additionally, adjusting the size of the points based on factors like highway miles per gallon would help convey more information.

3. Plot the model and the year using ggplot(). Use only the top 20 observations. Write the codes and its results.

```r
library(ggplot2)

top_20 <- head(mpg, 20)

ggplot(top_20, aes(x = model, y = year)) +
  geom_point(aes(color = class), size = 3) +
  labs(title = "Top 20 Car Models by Year",
       x = "Car Model",
       y = "Year") +
  theme_minimal() +
  theme(axis.text.x = element_text(angle = 45, hjust = 1))
```



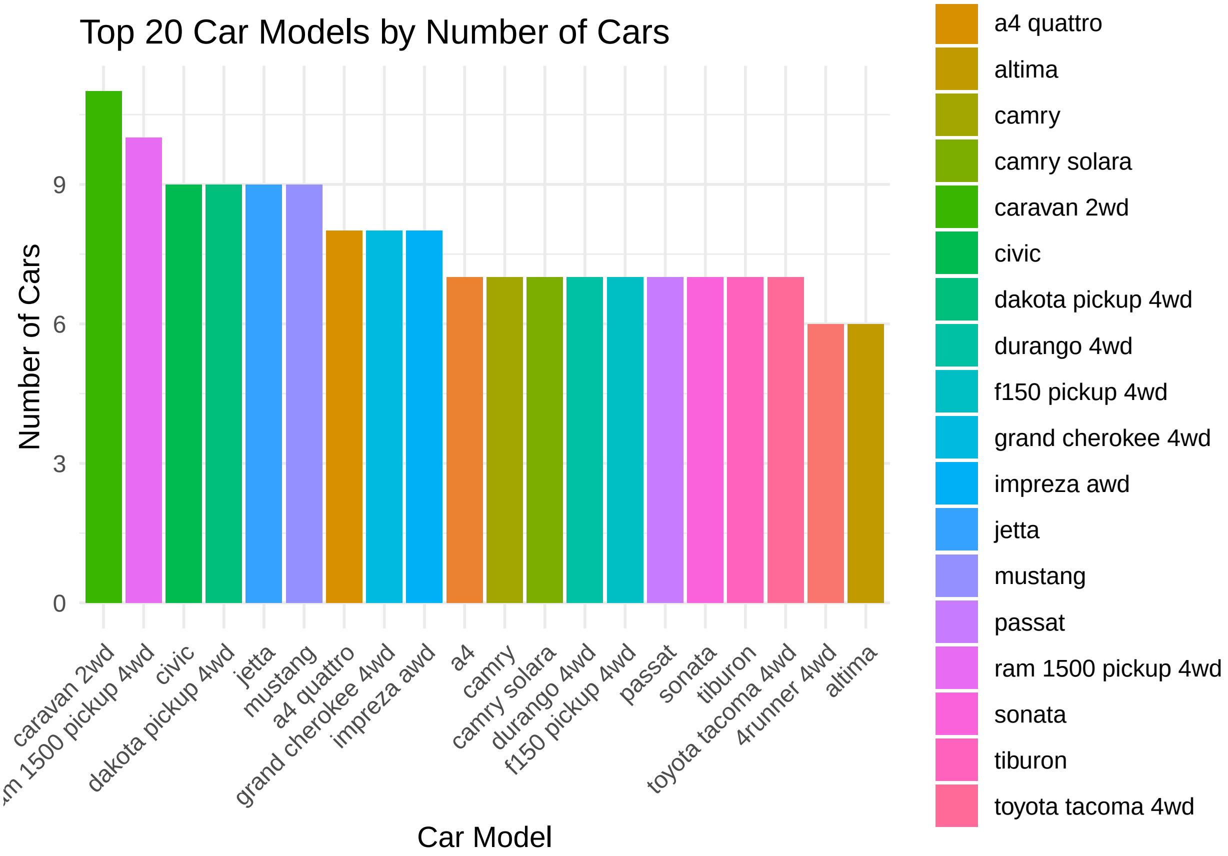
4. Using the pipe (%>%), group the model and get the number of cars per model. Show codes and its result

```
model_counts <- mpg %>%
  group_by(model) %>%
  summarise(num_cars = n()) %>%
  arrange(desc(num_cars))

print(model_counts)
```

```
## # A tibble: 38 x 2
##    model              num_cars
##    <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
```

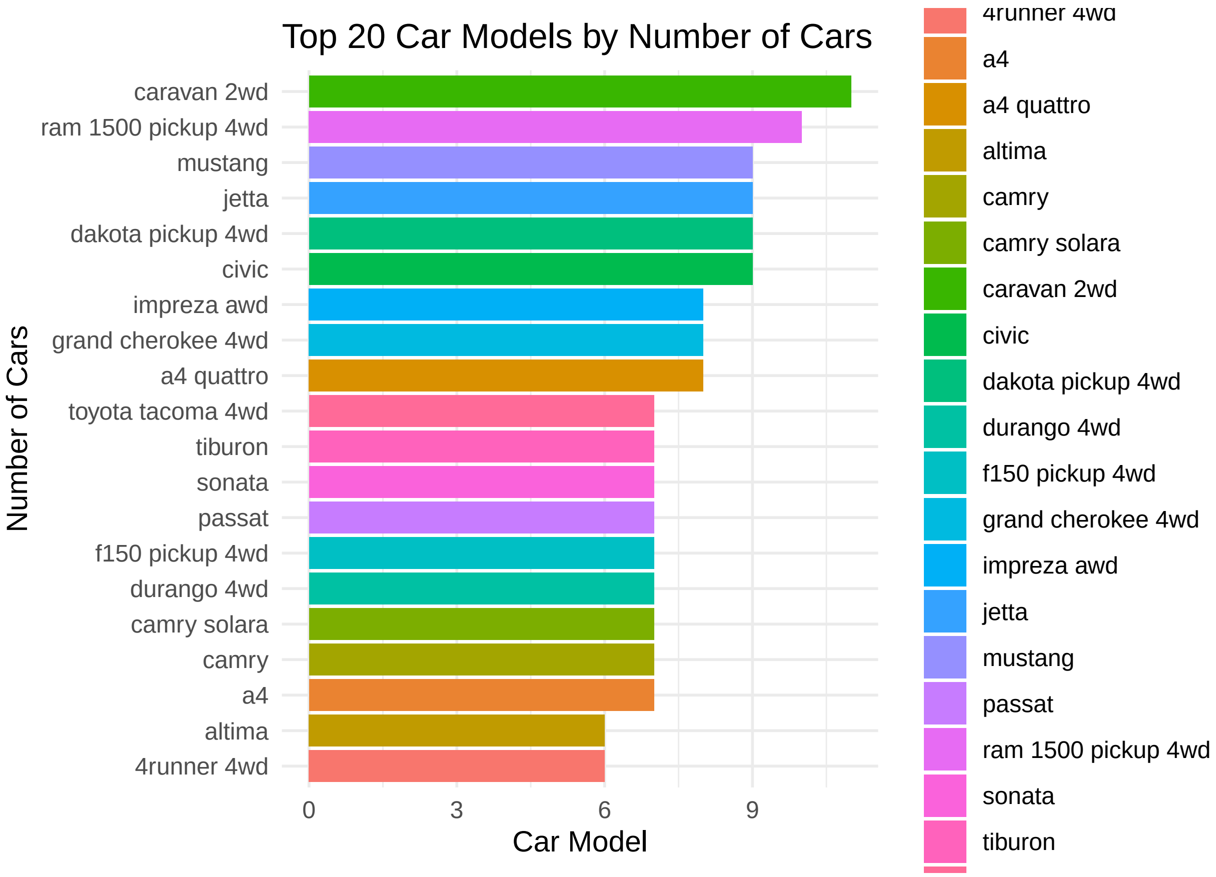
a. Plot using geom_bar() using the top 20 observations only. The graphs shoudl have a title, labels and colors. Show code and results.

```r
top_20_models <- model_counts %>% head(20)

ggplot(top_20_models, aes(x = reorder(model, -num_cars), y = num_cars)) +
  geom_bar(stat = "identity", aes(fill = model)) +
  labs(title = "Top 20 Car Models by Number of Cars",
       x = "Car Model",
       y = "Number of Cars") +
  theme_minimal() +
  theme(axis.text.x = element_text(angle = 45, hjust = 1))
```



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

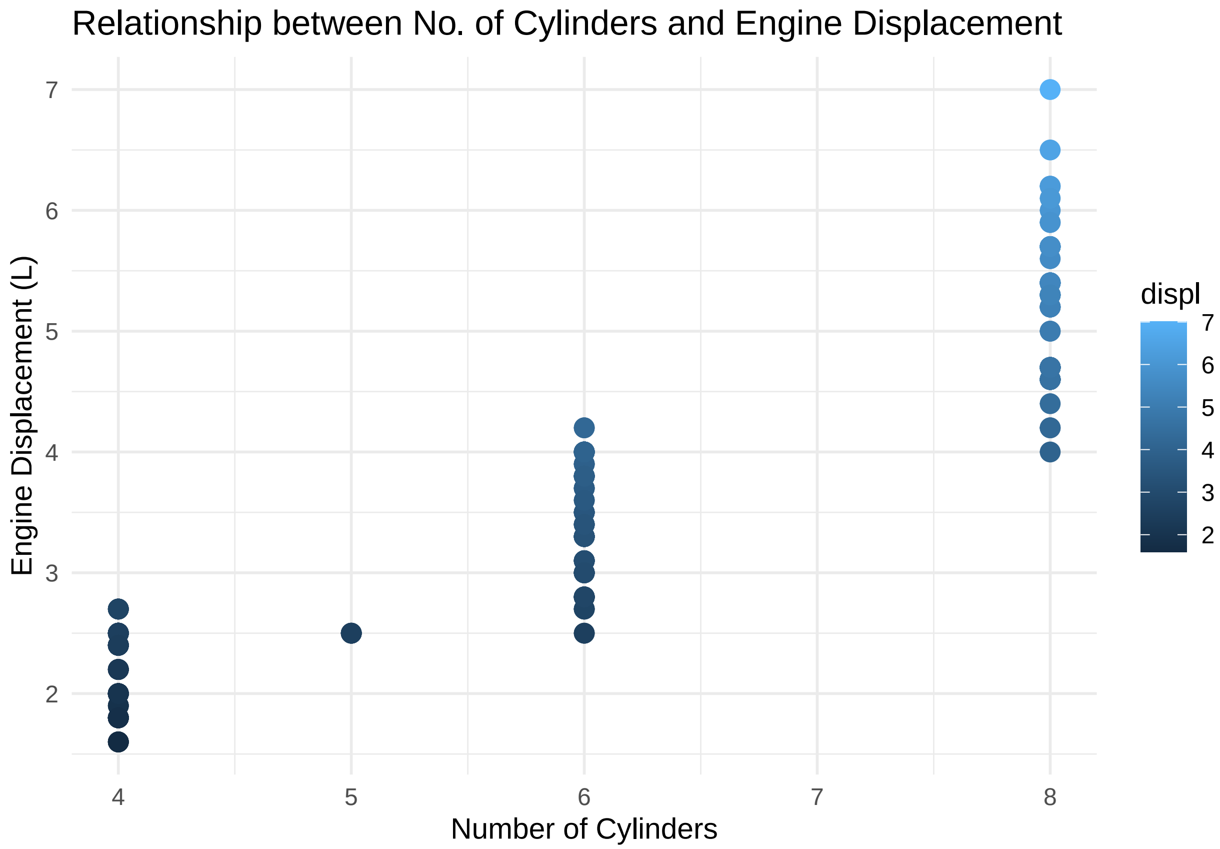
```r
ggplot(top_20_models, aes(x = reorder(model, num_cars), y = num_cars)) +
  geom_bar(stat = "identity", aes(fill = model)) +
  labs(title = "Top 20 Car Models by Number of Cars",
       x = "Number of Cars",
       y = "Car Model") +
  theme_minimal() +
  coord_flip()
```



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

a. How would you describe its relationship? Show the codes and its result.

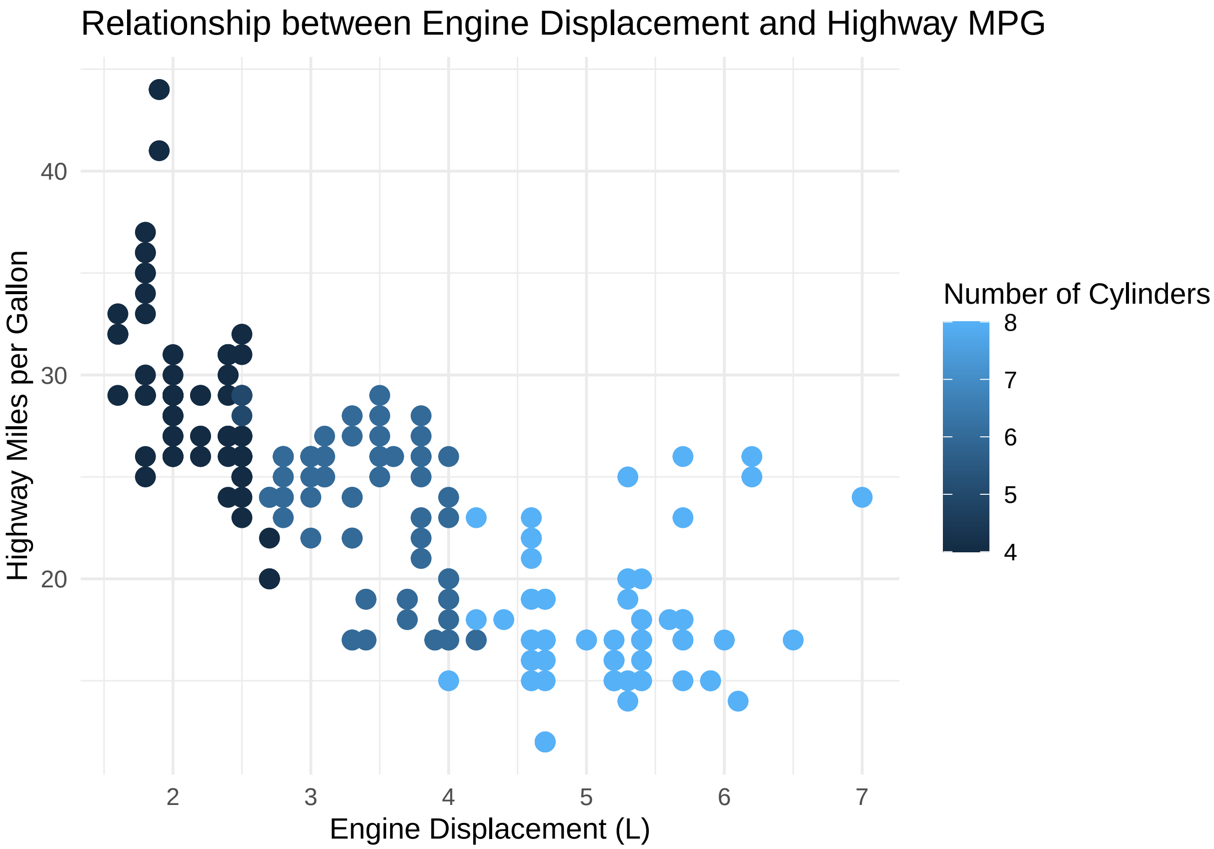
```
ggplot(mpg, aes(x = cyl, y = displ, color = displ)) +
  geom_point(size = 3) +
  labs(title = "Relationship between No. of Cylinders and Engine Displacement",
       x = "Number of Cylinders",
       y = "Engine Displacement (L)") +
  theme_minimal()
```



Discription: In the plot, I can see that as the number of cylinders goes up, the engine displacement also increases. This means cars with more cylinders tend to have bigger engines. The colors in the plot help show this pattern clearly.

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?

```
ggplot(mpg, aes(x = displ, y = hwy, color = cyl)) +
  geom_point(size = 3) +
  labs(title = "Relationship between Engine Displacement and Highway MPG",
       x = "Engine Displacement (L)",
       y = "Highway Miles per Gallon",
       color = "Number of Cylinders") +
  theme_minimal()
```



Description: The color gradient shows city MPG (cty). Usually, cars with bigger engines (higher displacement) get lower fuel efficiency, meaning they have lower highway (hwy) and city (cty) MPG. In other words, bigger engines use more fuel, so they show lower MPG on highways. This pattern shows a negative relationship between engine size (displ) and fuel efficiency (hwy and cty).

6. Import the traffic.csv onto your R environment.

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

a. How many numbers of observation does it have? What are the variables of the traffic dataset the Show your answer.

```
num_obs <- nrow(traffic)
variables <- colnames(traffic)

num_obs
```

```
## [1] 48120
```

```
variables
```

```
## [1] "DateTime" "Junction" "Vehicles" "ID"
```

b. subset the traffic dataset into junctions. What is the R codes and its output?

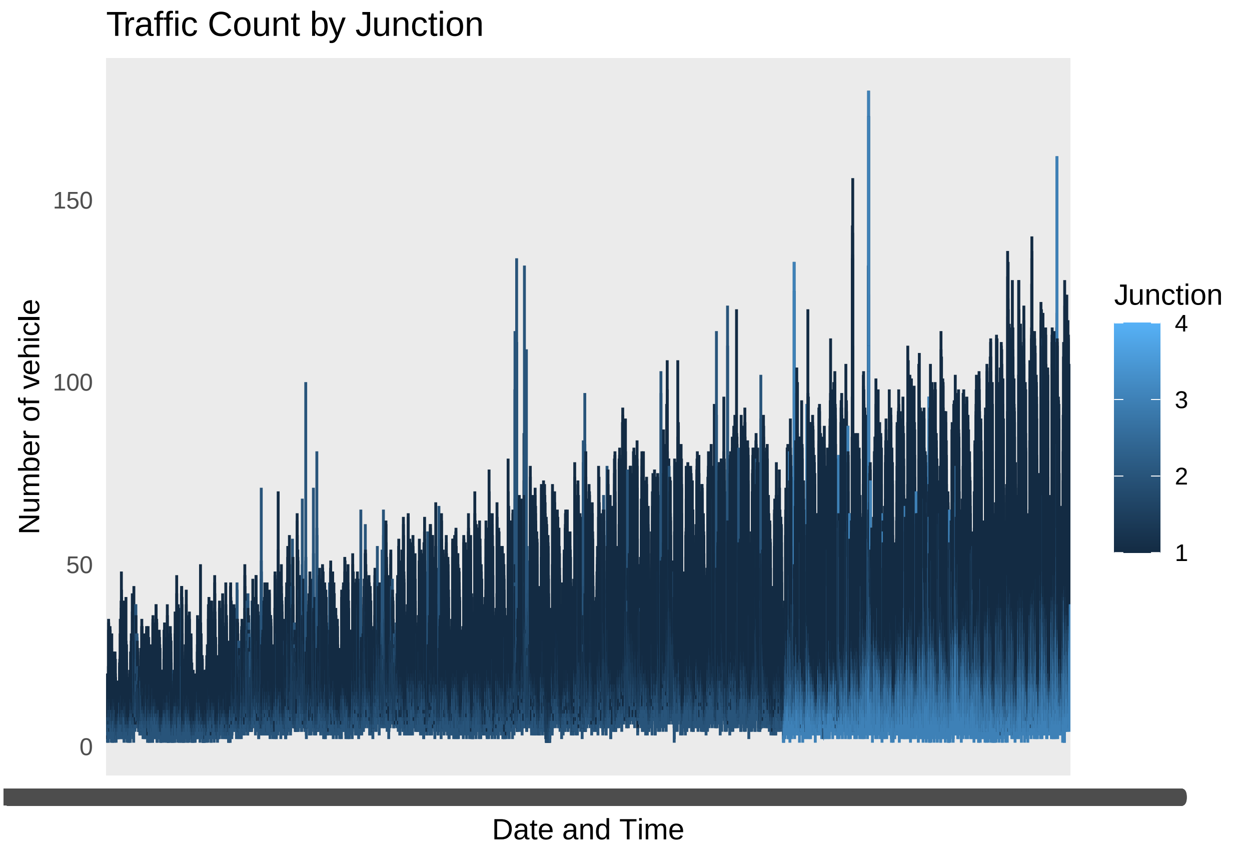
```
traffic_by_junction <- traffic %>%
  group_by(Junction) %>%
  group_split()

traffic_by_junction
```

```
## <list_of<
##   tbl_df<
##     DateTime: character
##     Junction: integer
##     Vehicles: integer
##     ID      : double
##   >
## >[4]>
## [[1]]
## # A tibble: 14,592 x 4
##    DateTime            Junction Vehicles          ID
##    <chr>                  <int>    <int>       <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
##
## [[2]]
## # A tibble: 14,592 x 4
##    DateTime            Junction Vehicles          ID
##    <chr>                  <int>    <int>       <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
##
## [[3]]
## # A tibble: 14,592 x 4
##    DateTime            Junction Vehicles          ID
##    <chr>                  <int>    <int>       <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
```

```
##
## [[4]]
## # A tibble: 4,344 x 4
##    DateTime            Junction Vehicles          ID
##    <chr>                  <int>    <int>       <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
```

c. Plot each junction in a using geom_line(). Show your solution and output.

```
ggplot(traffic, aes(x = DateTime, y = Vehicles, color = Junction)) +
  geom_line() +
  labs(title = "Traffic Count by Junction",
       x = "Date and Time",
       y = "Number of vehicle") +
  theme_minimal() +
  theme(legend.position = "right")
```



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

```r
library(readxl)

alexa <- read_excel("alexa_file.xlsx")
```

a. How many observations does alexa_file has? What about the number of columns? Show your solution and answer.

```r
num_obs <- nrow(alexa)
num_cols <- ncol(alexa)

num_obs
```

```
## [1] 3150
```

```r
num_cols
```

```
## [1] 5
```

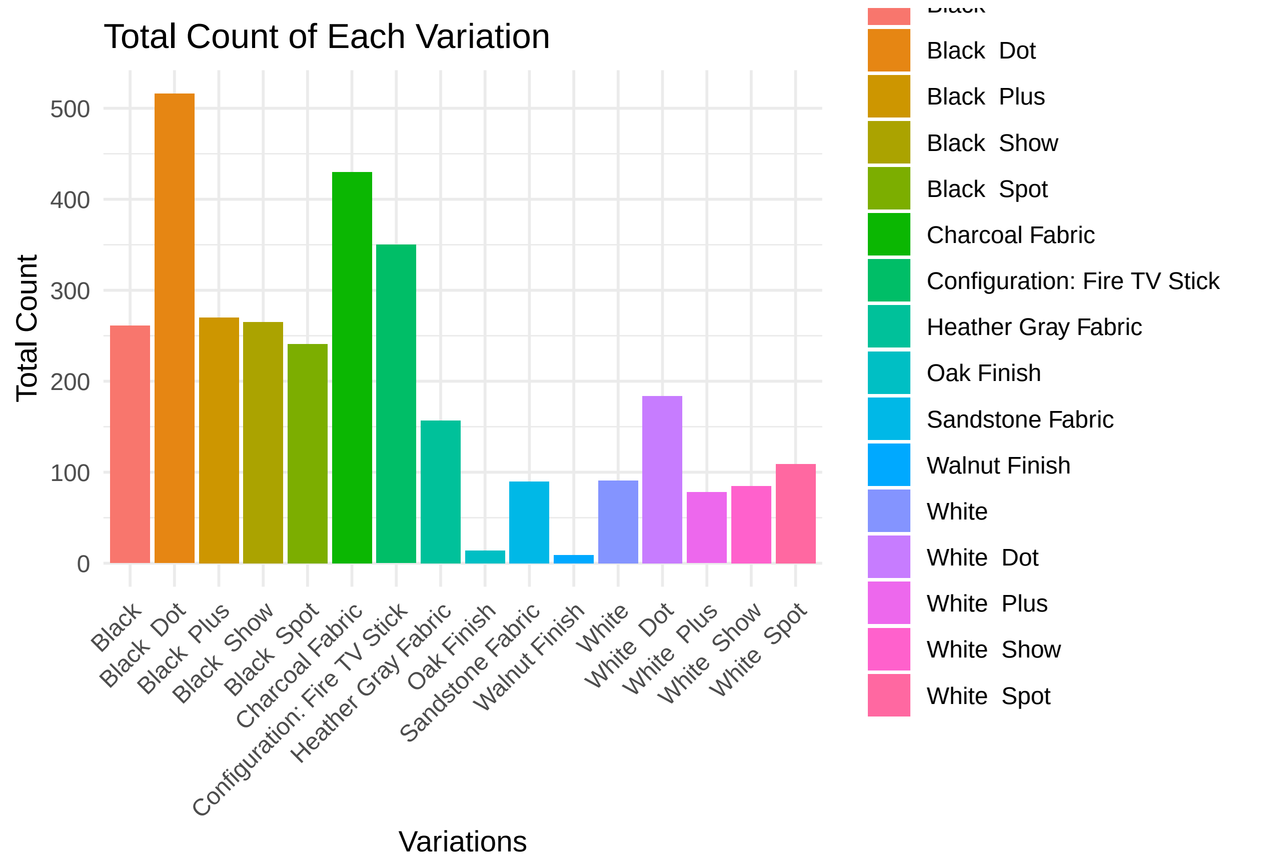
b. group the variations and get the total of each variations. Use dplyr package. Show solution and answer.

```r
var_totals <- alexa %>%
  group_by(variation) %>%
  summarise(total = n())

var_totals
```

```
## # A tibble: 16 x 2
##    variation                    total
##    <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
```

c. Plot the variations using the ggplot() function. What did you observe? Complete the details of the graph. Show solution and answer.

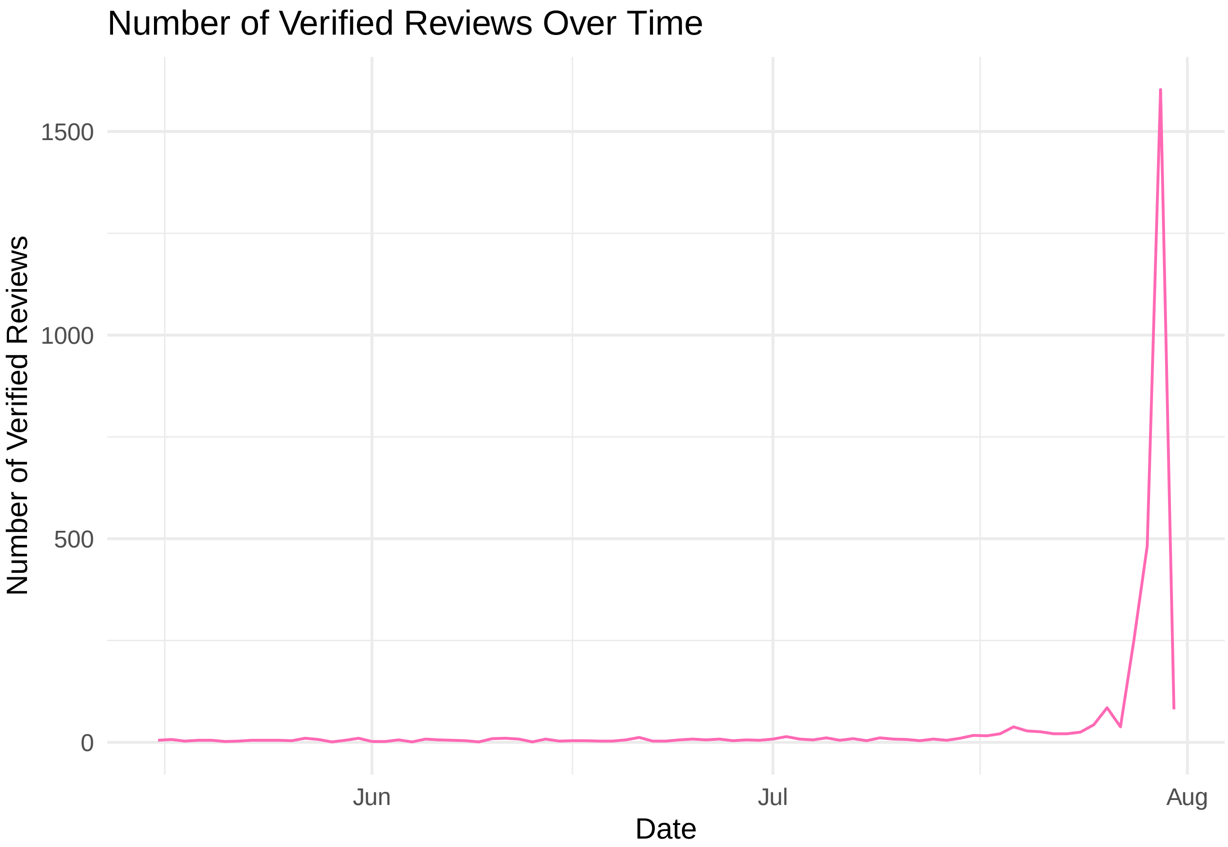
```r
ggplot(var_totals, aes(x = variation, y = total, fill = variation)) +
  geom_bar(stat = "identity") +
  labs(title = "Total Count of Each Variation",
       x = "Variations",
       y = "Total Count") +
  theme_minimal() +
  theme(axis.text.x = element_text(angle = 45, hjust = 1))
```



Observation: The bar chart shows counts of various product variations, but similar items like “Black Dot” and “Black Dot” appear separately due to inconsistent labeling, likely caused by extra spaces or slight name differences. This duplication makes the chart look cluttered, especially with many similar “Black” and “White” variations. The angled labels improve readability, and the color coding helps distinguish each variation. However, cleaning the data to remove extra spaces and group similar labels would create a clearer, more accurate representation of the main categories.

d. Plot a geom_line() with the date and the number of verified reviews. Complete the details of the graphs. Show your answer and solution.

```
verified_reviews_summary <- alexa %>%
  group_by(date) %>%
  summarise(verified_reviews_count = n())

ggplot(verified_reviews_summary, aes(x = date, y = verified_reviews_count)) +
  geom_line(color = "hotpink") +
  labs(title = "Number of Verified Reviews Over Time",
       x = "Date",
       y = "Number of Verified Reviews") +
  theme_minimal()
```



e. Get the relationship of variations and ratings. Which variations got the most highest in rating? Plot a graph to show its relationship. Show your solution and answer.

```
var_ratings <- alexa %>%
  group_by(variation) %>%
  summarise(avg_rating = mean(rating, na.rm = TRUE))

highest_var_rate <- var_ratings %>%
  arrange(desc(avg_rating)) %>%
  slice(1)

highest_var_rate
```

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