RWorksheet_Tupaz#4c

Lorie Mae Tupaz

2024-11-07

```
# 1.a
library(ggplot2)
```

```
Warning: package 'ggplot2' was built under R version 4.4.2
```

```
data(mpg)
str(mpg)
```

```
tibble [234 x 11] (S3: tbl_df/tbl/data.frame)
 $ 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        : int [1:234] 1999 1999 2008 2008 1999 1999 2008 1999 1999 2008 ...
 $ cyl         : int [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         : int [1:234] 18 21 20 21 16 18 18 18 16 20 ...
 $ hwy         : int [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" ...
```

```
# 1.b
#b. Categorical variables: manufacturer, model, trans, drv, fl, class, cyl


# 1.c
# Continuous variables: displ, year, cty, hwy
```

```r
#2.a
library(dplyr)
```

```
Warning: package 'dplyr' was built under R version 4.4.2
```

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

```
The following objects are masked from 'package:stats':

    filter, lag
```

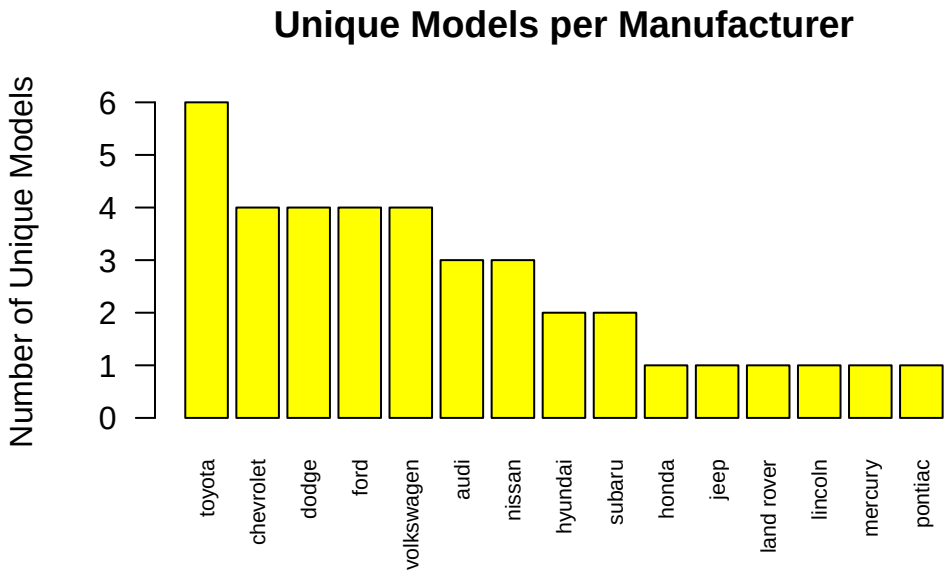
```
The following objects are masked from 'package:base':

    intersect, setdiff, setequal, union
```

```r
# Group by manufacturer and count unique models
manufacturer_models <- mpg %>%
  group_by(manufacturer) %>%
  summarise(unique_models = n_distinct(model)) %>%
  arrange(desc(unique_models))

# Display the result
print(manufacturer_models)
```

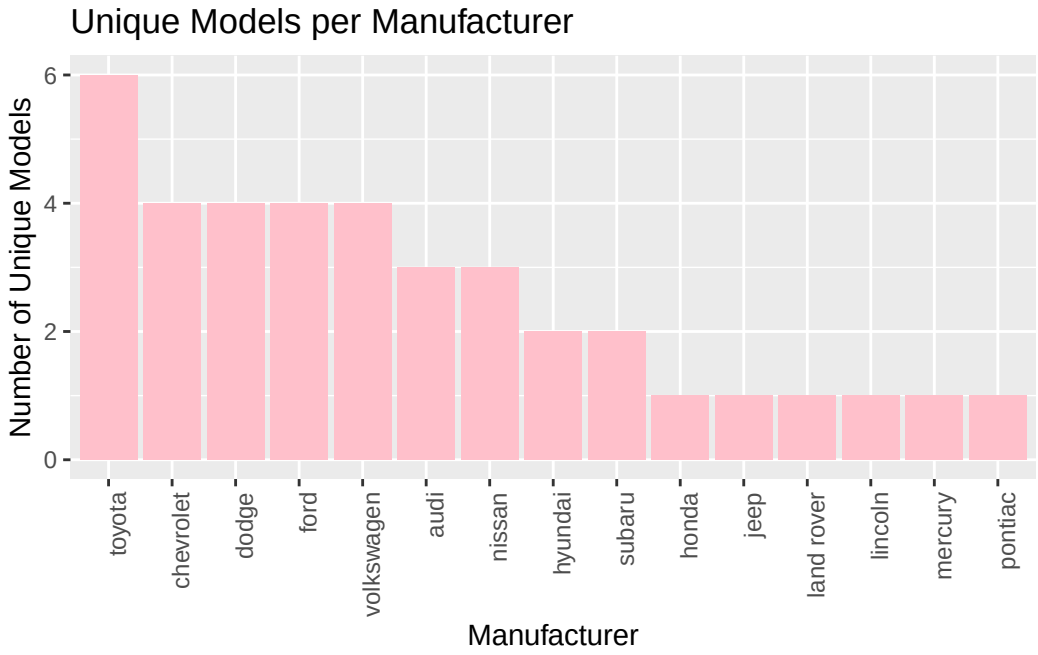
```
# A tibble: 15 x 2
   manufacturer unique_models
   <chr>                <int>
 1 toyota                   6
 2 chevrolet                4
 3 dodge                    4
 4 ford                     4
 5 volkswagen               4
 6 audi                     3
 7 nissan                   3
 8 hyundai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

```
13 lincoln                  1
14 mercury                  1
15 pontiac                  1
```

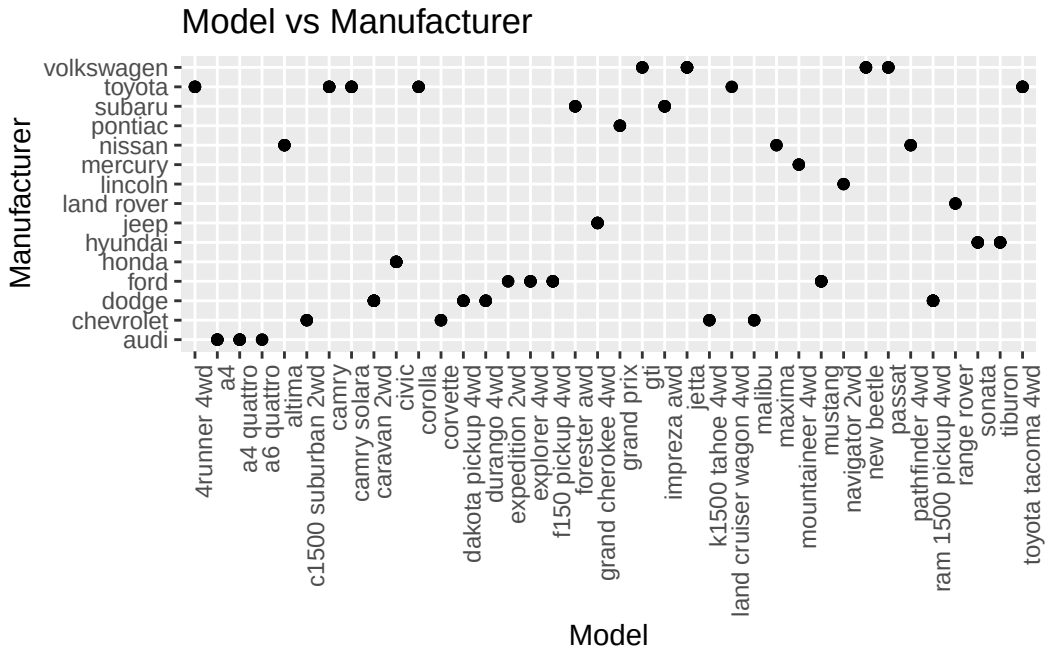
```r
#2.b
# plot()
manufacturer_models$manufacturer <- factor(manufacturer_models$manufacturer,
                                           levels = manufacturer_models$manufacturer[order(-m

# Bar plot using base R plot function
barplot(manufacturer_models$unique_models, names.arg = manufacturer_models$manufacturer,
        las = 2, col = "yellow", main = "Unique Models per Manufacturer",
        ylab = "Number of Unique Models", cex.names = 0.7)
```



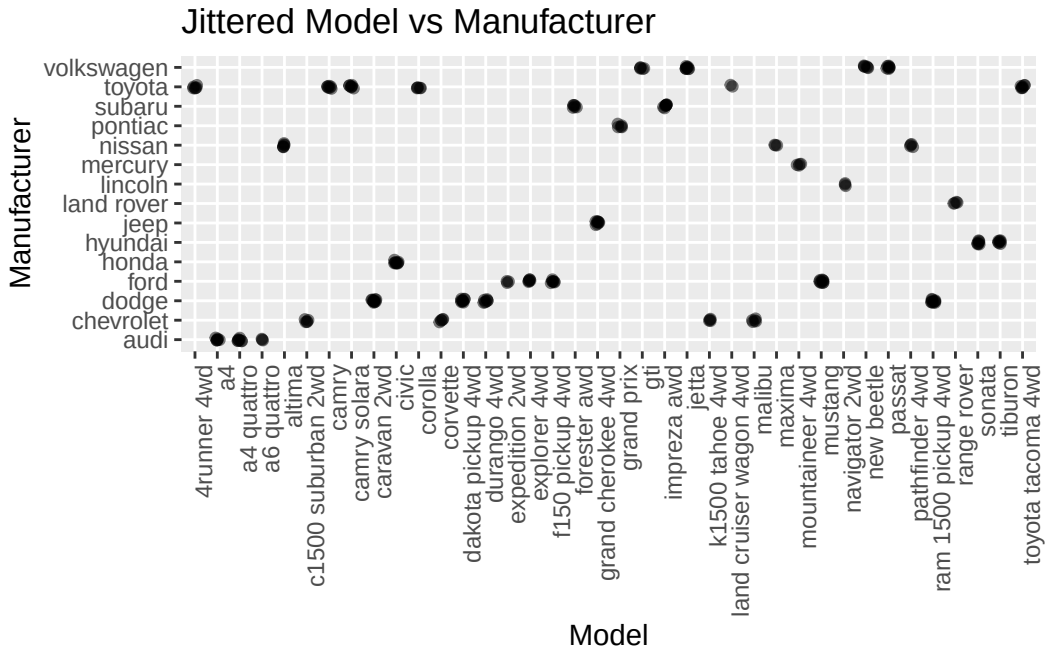

```r
#ggplot()
library(ggplot2)

# Plotting with ggplot2
ggplot(manufacturer_models, aes(x = manufacturer, y = unique_models)) +
  geom_bar(stat = "identity", fill = "pink") +
  theme(axis.text.x = element_text(angle = 90, hjust = 1)) +
  labs(title = "Unique Models per Manufacturer", x = "Manufacturer", y = "Number of Unique Mo
```



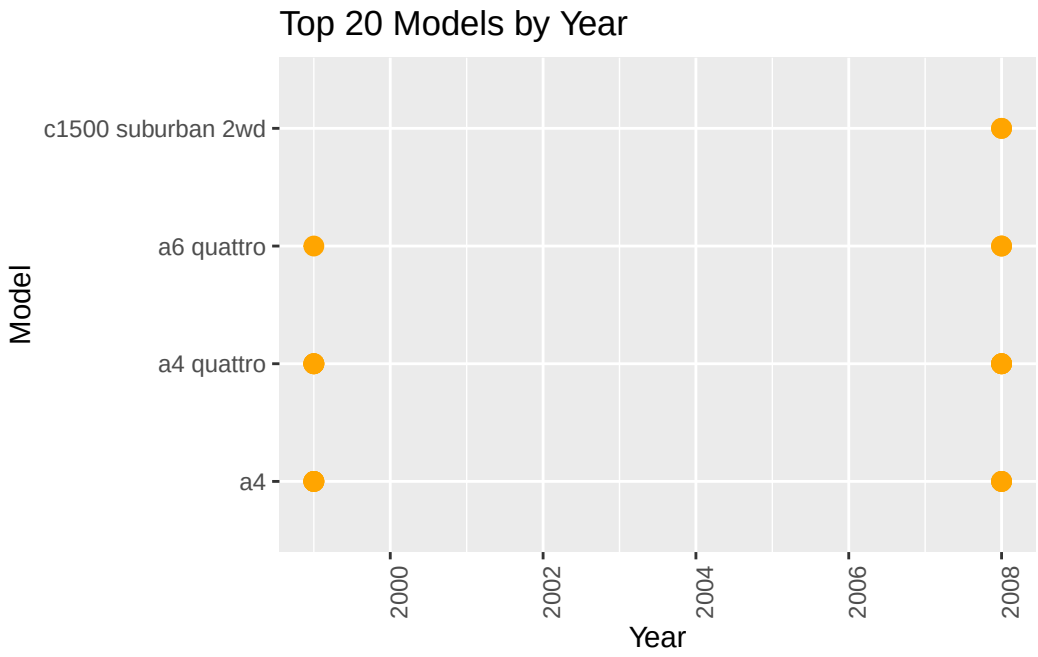

```
# 2. Showing the Relationship Between the Model and Manufacturer

#2.a
# This plot shows the relationship between the model and the manufacturer. Each point repres
ggplot(mpg, aes(x = model, y = manufacturer)) +
  geom_point() +
  theme(axis.text.x = element_text(angle = 90, hjust = 1)) +
  labs(title = "Model vs Manufacturer",
       x = "Model",
       y = "Manufacturer")
```




```
# 2.b
# This plot might not be very useful if there are many models and manufacturers, as the point
#Using geom_jitter() to spread the points slightly and reduce overlap.
#Filtering the data to show only the most common manufacturers or models.
#Adding more informative aesthetics, such as color or size, based on another variable like di

ggplot(mpg, aes(x = model, y = manufacturer)) +
  geom_jitter(width = 0.1, height = 0.1, alpha = 0.5) +
  theme(axis.text.x = element_text(angle = 90, hjust = 1)) +
  labs(title = "Jittered Model vs Manufacturer",
       x = "Model",
       y = "Manufacturer")
```




```r
#3
# Load necessary libraries
library(ggplot2)
library(dplyr)

# Select the top 20 observations from the mpg dataset
top_20_mpg <- mpg %>%
  slice_head(n = 20)  # or you can use head(mpg, 20)

# Create a scatter plot of model vs. year for the top 20 observations
ggplot(top_20_mpg, aes(x = year, y = model)) +
  geom_point(color = "orange", size = 3) +
  labs(title = "Top 20 Models by Year",
       x = "Year",
       y = "Model") +
  theme(axis.text.x = element_text(angle = 90, hjust = 1))
```

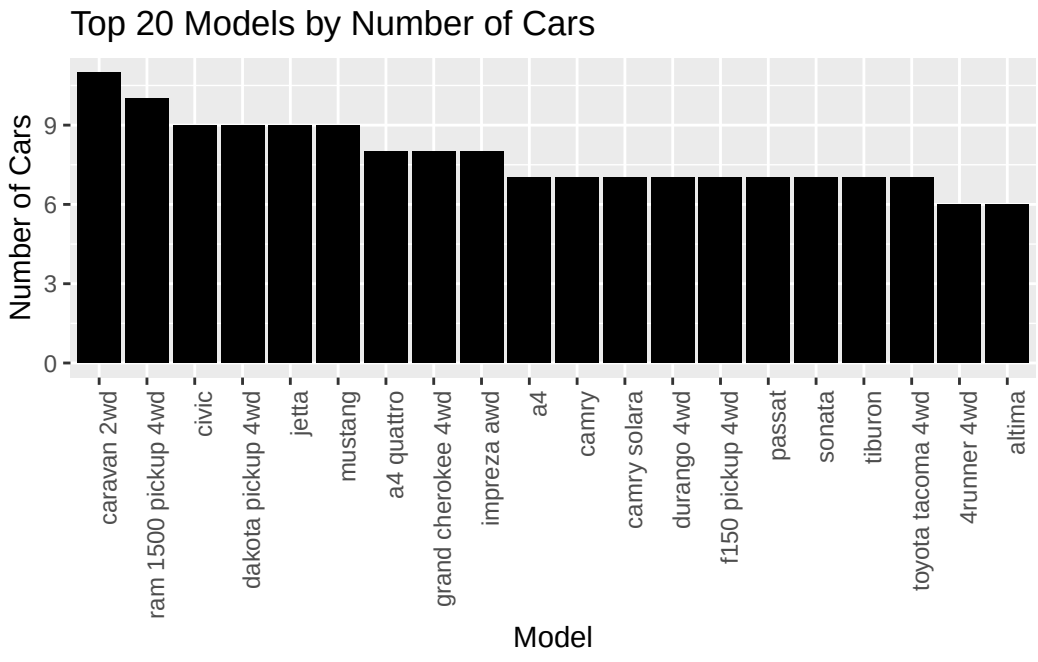



```
#4
# Load necessary libraries
library(ggplot2)
library(dplyr)

# Group by model and count the number of cars per model
model_count <- mpg %>%
  group_by(model) %>%
  summarise(num_cars = n()) %>%
  arrange(desc(num_cars))

# Display the result
print(model_count)
```

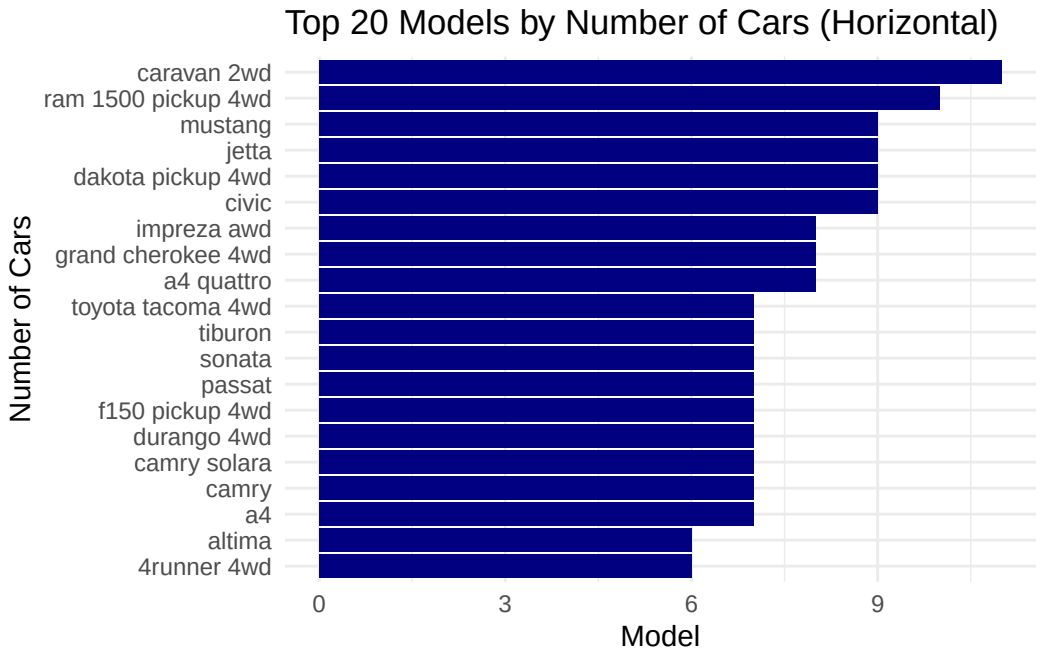
```
# A tibble: 38 x 2
   model               num_cars
   <chr>                  <int>
 1 caravan 2wd               11
 2 ram 1500 pickup 4wd       10
 3 civic                      9
 4 dakota pickup 4wd          9
 5 jetta                      9
```

```
 6 mustang                    9
 7 a4 quattro                 8
 8 grand cherokee 4wd         8
 9 impreza awd                8
10 a4                         7
# i 28 more rows
```

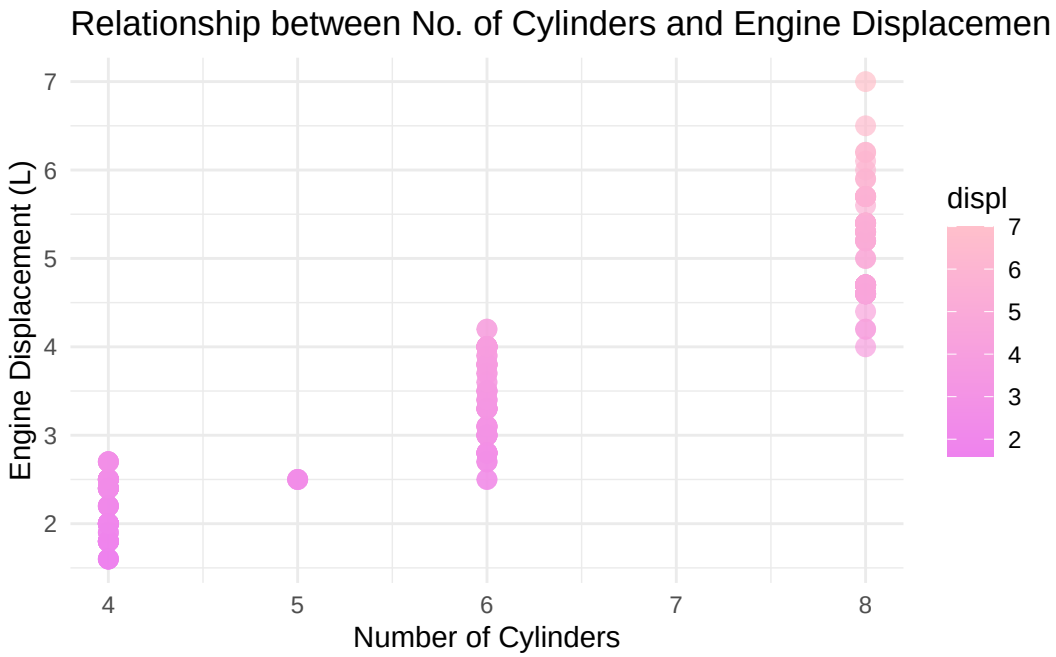
```r
# Create a dataset for the top 20 models by number of cars
top_20_mpg <- model_count %>%
  head(20)



# 4.a - Plot using geom_bar() for the top 20 observations (Vertical Bar Plot)
ggplot(top_20_mpg, aes(x = reorder(model, -num_cars), y = num_cars)) +
  geom_bar(stat = "identity", fill = "black") +
  labs(title = "Top 20 Models by Number of Cars",
       x = "Model",
       y = "Number of Cars") +
  theme(axis.text.x = element_text(angle = 90, hjust = 1))
```



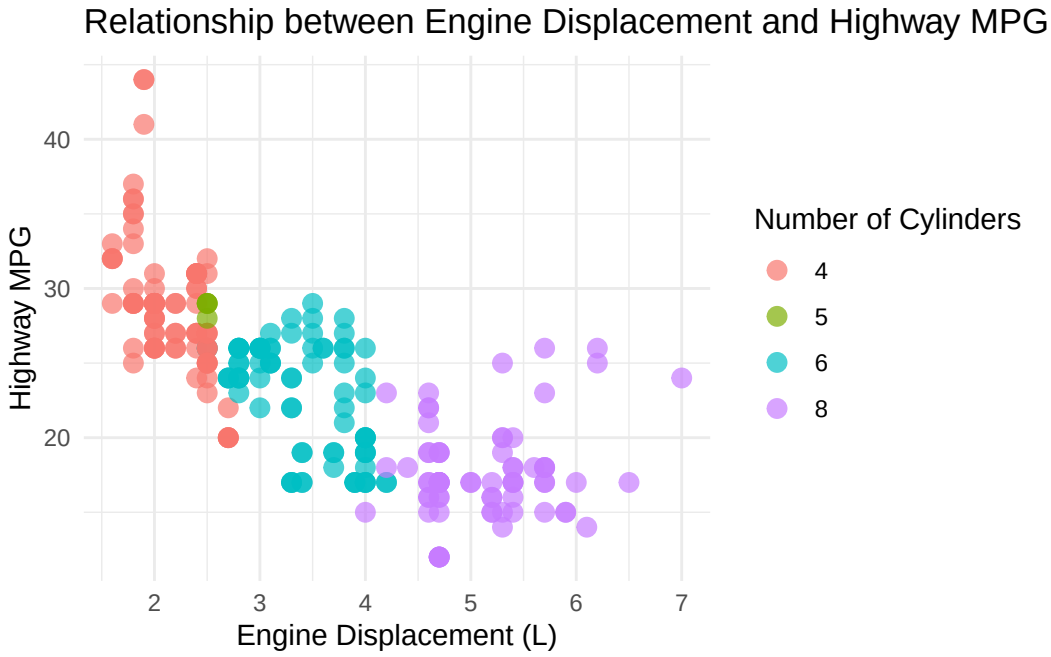
```r
# 4.b - Plot using geom_bar() + coord_flip() for the top 20 observations (Horizontal Bar Plot
ggplot(top_20_mpg, aes(x = reorder(model, num_cars), y = num_cars)) +
  geom_bar(stat = "identity", fill = "navy") +
  labs(title = "Top 20 Models by Number of Cars (Horizontal)",
       x = "Number of Cars",
       y = "Model") +
  coord_flip() +
  theme_minimal()
```




```r
# 5
library(ggplot2)

# Create the scatter plot
ggplot(mpg, aes(x = cyl, y = displ, color = displ)) +
  geom_point(size = 3, alpha = 0.7) +  # Adjust size and transparency of points
  labs(title = "Relationship between No. of Cylinders and Engine Displacement",
       x = "Number of Cylinders",
       y = "Engine Displacement (L)") +
  scale_color_gradient(low = "violet", high = "pink") +  # Color gradient from blue to red
  theme_minimal()
```




```
#6
library(ggplot2)

# Create the scatter plot for displ vs hwy, mapping color to cyl
ggplot(mpg, aes(x = displ, y = hwy, color = as.factor(cyl))) +
  geom_point(size = 3, alpha = 0.7) +  # Adjust size and transparency of points
  labs(title = "Relationship between Engine Displacement and Highway MPG",
       x = "Engine Displacement (L)",
       y = "Highway MPG",
       color = "Number of Cylinders") +
  theme_minimal()
```




```
#6.a
setwd("C:/Users/Juralin/OneDrive/Documents")

# Use the absolute file path directly
traffic <- read.csv("C:/Users/Juralin/OneDrive/Documents/traffic.csv")


# Check if the file loaded successfully
head(traffic)
```

```
      Time Vehicles
1  8:00 AM       25
2  9:00 AM       30
3 10:00 AM       15
```

```
# Check the number of rows (observations) and the column names of the dataset
num_observations <- nrow(traffic)
variables <- colnames(traffic)

# Print the results
cat("Number of observations:", num_observations, "\n")
```

```
Number of observations: 3
```

```
cat("Variables in the dataset:", paste(variables, collapse = ", "), "\n")
```

```
Variables in the dataset: Time, Vehicles
```

```
#6.b
traffic <- read.csv("C:/Users/Juralin/OneDrive/Documents/traffic.csv")

# Step 2: Check the first few rows of the dataset to understand its structure
head(traffic)
```

```
      Time Vehicles
1  8:00 AM       25
2  9:00 AM       30
3 10:00 AM       15
```

```
# Step 3: Check the column names to see which columns are available
colnames(traffic)
```

```
[1] "Time"     "Vehicles"
```

```
# Step 4: Subset the dataset based on an existing column
# Since there's no 'junction' column, let's try splitting by 'Time' or 'Vehicles'
# Let's first try splitting by 'Time' (or you can change to 'Vehicles' if needed)

# Check if 'Time' exists in the dataset
if("Time" %in% colnames(traffic)) {
  # Split the dataset by the 'Time' column
  time_subset <- split(traffic, traffic$Time)

  # Display a sample output of the subset for one of the time categories
  cat("Sample output for Time=1:\n")
  head(time_subset[[1]])  # Replace '1' with the relevant value if necessary
} else {
  cat("Column 'Time' does not exist in the dataset.\n")
}
```

```
Sample output for Time=1:
```

```
      Time Vehicles
3 10:00 AM       15
```

```
# Alternatively, you can try splitting by 'Vehicles' if 'Time' is not suitable
if("Vehicles" %in% colnames(traffic)) {
  # Split the dataset by the 'Vehicles' column
  vehicles_subset <- split(traffic, traffic$Vehicles)

  # Display a sample output of the subset for one of the vehicle categories
  cat("Sample output for Vehicles=50:\n")
  head(vehicles_subset[[1]])  # Replace '50' with the relevant value if necessary
} else {
  cat("Column 'Vehicles' does not exist in the dataset.\n")
}
```

```
Sample output for Vehicles=50:
```

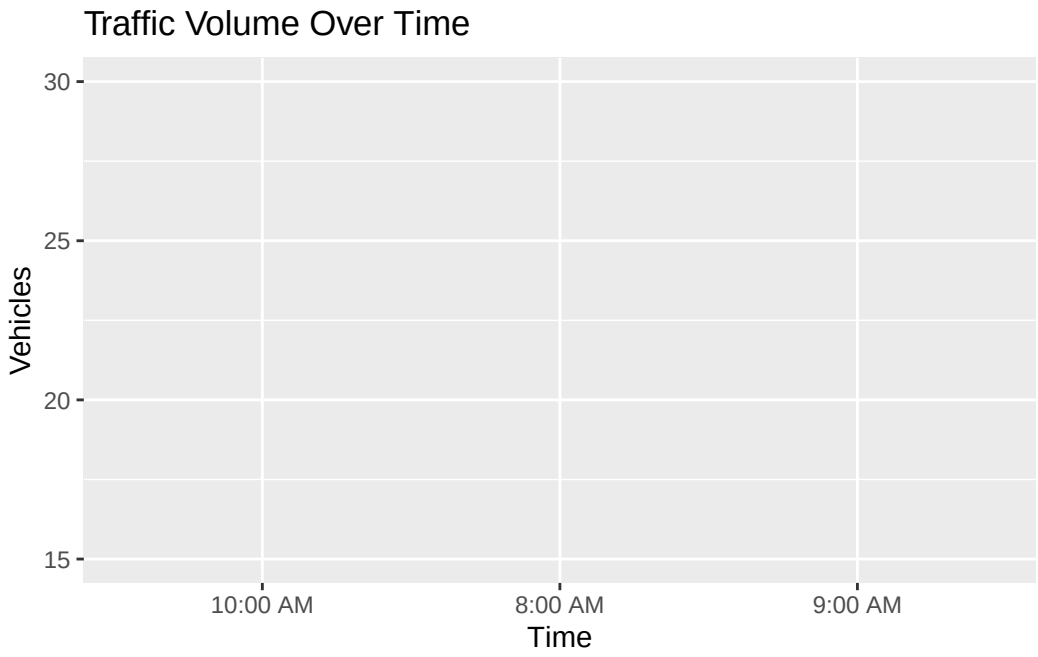
```
      Time Vehicles
3 10:00 AM       15
```

```
#6.c
library(ggplot2)

# Check the column names to ensure they are correct
colnames(traffic)
```

```
[1] "Time"     "Vehicles"
```

```
# Create a basic line plot for 'Time' vs. 'Vehicles'
ggplot(traffic, aes(x = Time, y = Vehicles)) +
  geom_line() +
  labs(title = "Traffic Volume Over Time", x = "Time", y = "Vehicles")
```

```
`geom_line()`: Each group consists of only one observation.
i Do you need to adjust the group aesthetic?
```




```
# Optional: If you want to explore the data further, you can create subsets by time intervals
# Example of creating a subset by time
# traffic_subset <- subset(traffic, Time >= 10 & Time <= 20)  # Adjust the time range as nee

# Example of plotting a subset of data
# ggplot(traffic_subset, aes(x = Time, y = Vehicles)) +
#   geom_line() +
#   labs(title = "Traffic Volume Over Time (Subset)", x = "Time", y = "Vehicles")
```

```
#7.a
# Load the necessary library
library(readxl)
```

```
Warning: package 'readxl' was built under R version 4.4.2
```

```
# Import the dataset from the specified path
alexa_data <- read_excel("C:/Users/Juralin/OneDrive/Documents/alexa_file.xlsx")

# Get the number of observations (rows) and columns
dimensions <- dim(alexa_data)
```

```r
# Display the result
cat("Number of observations:", dimensions[1], "\n")
```

```
Number of observations: 3150
```

```r
cat("Number of columns:", dimensions[2], "\n")
```

```
Number of columns: 5
```

```r
#6.b
# Load dplyr package
library(dplyr)

# Ensure 'verified_reviews' is numeric
alexa_data$verified_reviews <- as.numeric(alexa_data$verified_reviews)
```

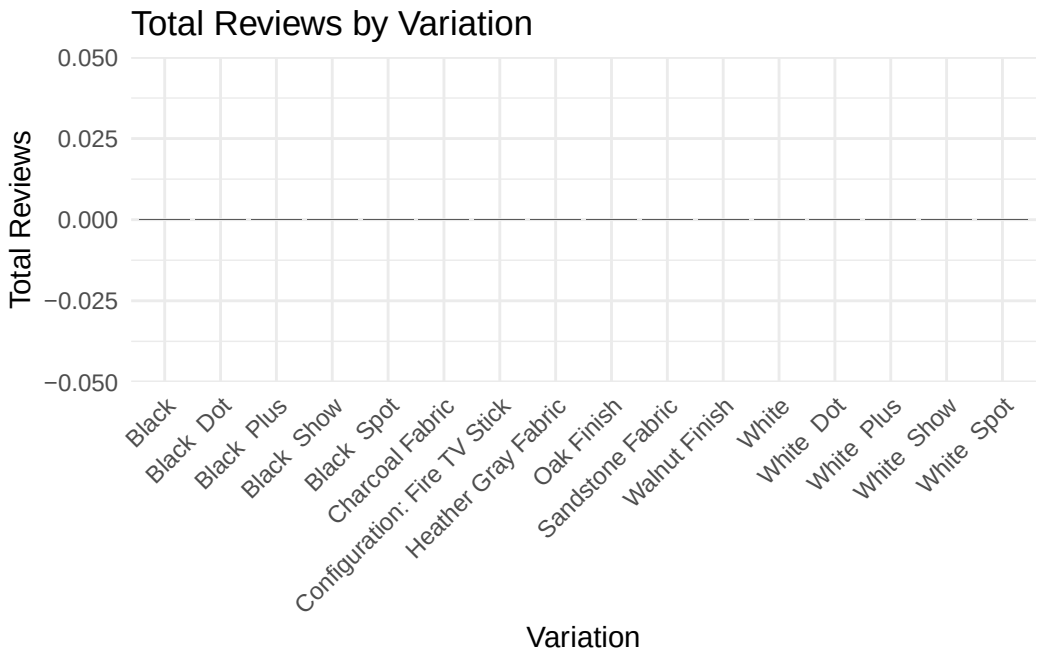
```
Warning: NAs introduced by coercion
```

```r
# Group by the variations and calculate the total verified reviews
variations_total <- alexa_data %>%
  group_by(variation) %>%
  summarize(total = sum(verified_reviews, na.rm = TRUE))

# Display the result
print(variations_total)
```

```
# A tibble: 16 x 2
   variation                    total
   <chr>                        <dbl>
 1 Black                            0
 2 Black  Dot                      0
 3 Black  Plus                     0
 4 Black  Show                     0
 5 Black  Spot                     0
 6 Charcoal Fabric                 0
 7 Configuration: Fire TV Stick    0
 8 Heather Gray Fabric             0
 9 Oak Finish                      0
10 Sandstone Fabric                0
```

```
11 Walnut Finish                    0
12 White                            0
13 White  Dot                       0
14 White  Plus                      0
15 White  Show                      0
16 White  Spot                      0
```

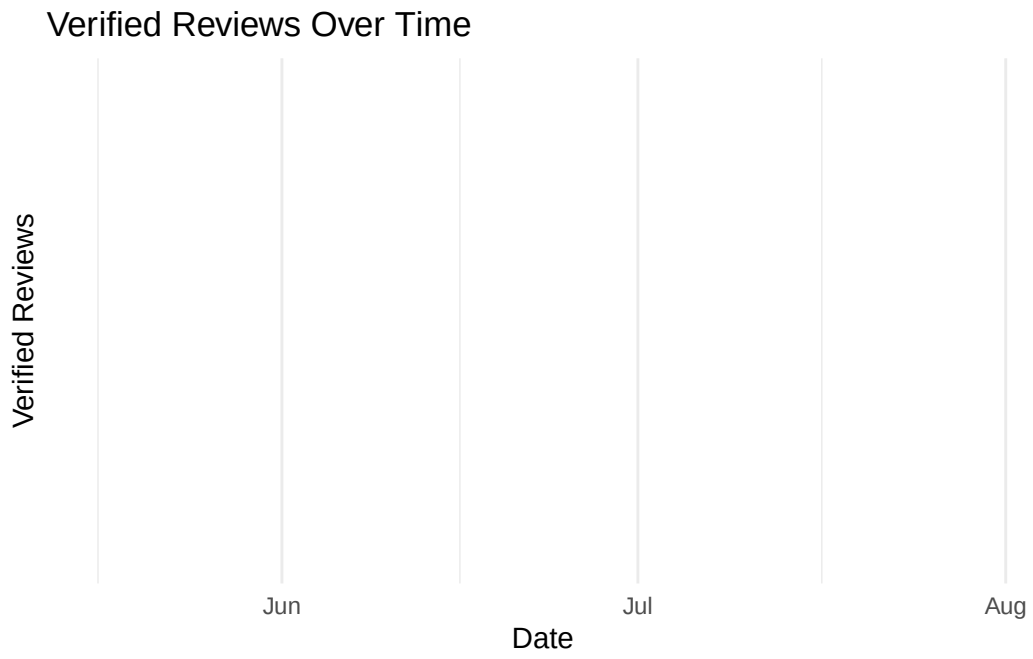
```r
#6.c
# Load ggplot2 package
library(ggplot2)

# Create a bar plot of variations
ggplot(variations_total, aes(x = variation, y = total)) +
  geom_bar(stat = "identity") +
  theme_minimal() +
  labs(title = "Total Reviews by Variation",
       x = "Variation",
       y = "Total Reviews") +
  theme(axis.text.x = element_text(angle = 45, hjust = 1))
```




```r
#6.d
# Ensure 'date' is in Date format
```

```r
alexa_data$date <- as.Date(alexa_data$date, format = "%Y-%m-%d")

# Plot the date vs. verified reviews with a line plot
ggplot(alexa_data, aes(x = date, y = verified_reviews)) +
  geom_line() +
  theme_minimal() +
  labs(title = "Verified Reviews Over Time",
       x = "Date",
       y = "Verified Reviews")
```

```
Warning: Removed 3150 rows containing missing values or values outside the scale range
(`geom_line()`).
```




```r
#6.e
# Get the mean rating for each variation
variation_ratings <- alexa_data %>%
  group_by(variation) %>%
  summarize(mean_rating = mean(rating, na.rm = TRUE))

# Find the variation with the highest rating
highest_rating_variation <- variation_ratings %>%
```

```r
  filter(mean_rating == max(mean_rating))

# Display the variation with the highest rating
print(highest_rating_variation)
```

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

```r
# Plot the relationship between variations and ratings
ggplot(variation_ratings, aes(x = variation, y = mean_rating)) +
  geom_bar(stat = "identity") +
  theme_minimal() +
  labs(title = "Mean Rating by Variation",
       x = "Variation",
       y = "Mean Rating") +
  theme(axis.text.x = element_text(angle = 45, hjust = 1))
```