RWORKSHEET_GENTAPAO#4C

BEA JULIETTE L. GENTAPAO

2025-12-22

R Markdown

This is an R Markdown document. Markdown is a simple formatting syntax for authoring HTML, PDF, and MS Word documents. For more details on using R Markdown see http://rmarkdown.rstudio.com.

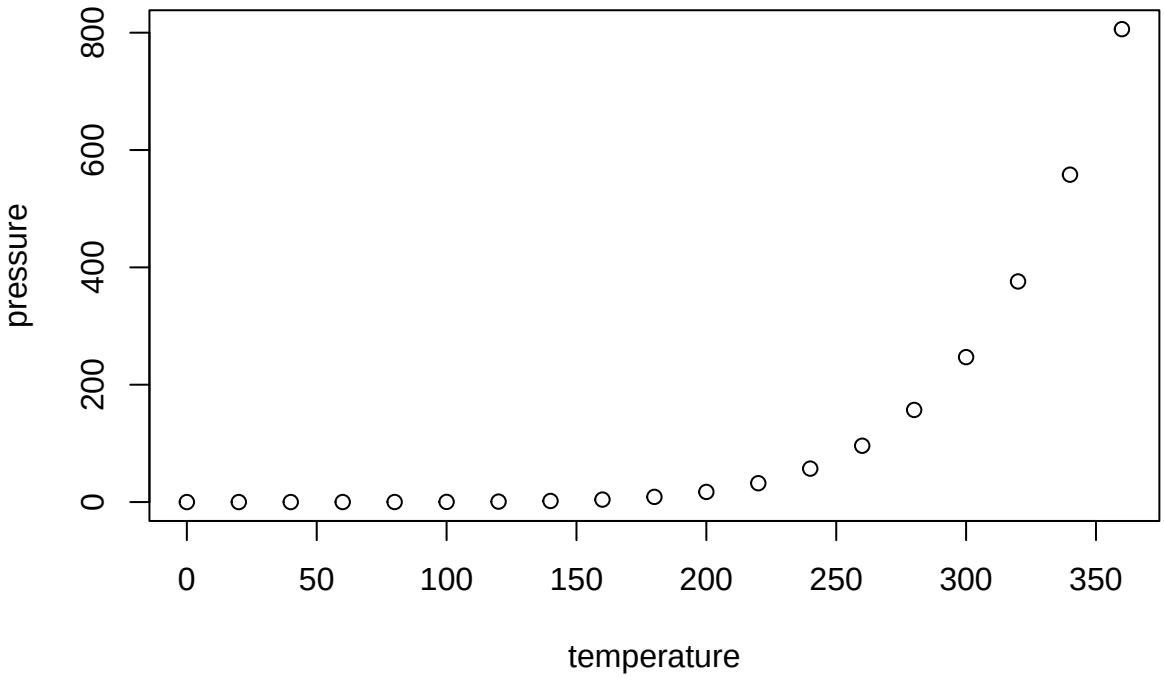
When you click the **Knit** button a document will be generated that includes both content as well as the output of any embedded R code chunks within the document. You can embed an R code chunk like this:

```
summary(cars)
```

```
##      speed           dist
##  Min.   : 4.0   Min.   :  2.00
##  1st Qu.:12.0   1st Qu.: 26.00
##  Median :15.0   Median : 36.00
##  Mean   :15.4   Mean   : 42.98
##  3rd Qu.:19.0   3rd Qu.: 56.00
##  Max.   :25.0   Max.   :120.00
```

Including Plots

You can also embed plots, for example:



Note that the `echo = FALSE` parameter was added to the code chunk to prevent printing of the R code that generated the plot.

1. Import and explore mpg dataset

a. Import CSV file into environment

```
# Load the built-in mpg dataset from ggplot2
data(mpg)

# Alternative: if you have a CSV file
# mpg_data <- read.csv("mpg.csv")

# Display first few rows
head(mpg)
```

```
## # A tibble: 6 x 11
##   manufacturer model displ  year   cyl trans      drv     cty   hwy fl    class
##   <chr>        <chr> <dbl> <int> <int> <chr>      <chr> <int> <int> <chr> <chr>
## 1 audi         a4      1.8  1999     4 auto(l5)   f        18    29 p     compa~
## 2 audi         a4      1.8  1999     4 manual(m5) f        21    29 p     compa~
## 3 audi         a4      2    2008     4 manual(m6) f        20    31 p     compa~
## 4 audi         a4      2    2008     4 auto(av)   f        21    30 p     compa~
## 5 audi         a4      2.8  1999     6 auto(l5)   f        16    26 p     compa~
## 6 audi         a4      2.8  1999     6 manual(m5) f        18    26 p     compa~
```

b. Categorical variables

```
# Identify categorical variables
categorical_vars <- c("manufacturer", "model", "trans", "drv", "fl", "class")

cat("Categorical variables in mpg dataset:\n")
```

```
## Categorical variables in mpg dataset:
```

```
print(categorical_vars)
```

```
## [1] "manufacturer" "model"        "trans"        "drv"          "fl"
## [6] "class"
```

```
# Show structure
str(mpg[, categorical_vars])
```

```
## tibble [234 x 6] (S3: tbl_df/tbl/data.frame)
##  $ manufacturer: chr [1:234] "audi" "audi" "audi" "audi" ...
##  $ model       : chr [1:234] "a4" "a4" "a4" "a4" ...
##  $ trans       : chr [1:234] "auto(l5)" "manual(m5)" "manual(m6)" "auto(av)" ...
##  $ drv         : chr [1:234] "f" "f" "f" "f" ...
##  $ fl          : chr [1:234] "p" "p" "p" "p" ...
##  $ class       : chr [1:234] "compact" "compact" "compact" "compact" ...
```

Answer: The categorical variables are: manufacturer, model, trans (transmission), drv (drive train), fl (fuel type), and class (type of car).

c. Continuous variables

```r
# Identify continuous variables
continuous_vars <- c("displ", "year", "cyl", "cty", "hwy")

cat("Continuous variables in mpg dataset:\n")
```

```
## Continuous variables in mpg dataset:
```

```r
print(continuous_vars)
```

```
## [1] "displ" "year"  "cyl"   "cty"   "hwy"
```

```r
# Show structure
str(mpg[, continuous_vars])
```

```
## tibble [234 x 5] (S3: tbl_df/tbl/data.frame)
##  $ 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 ...
##  $ 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 ...
```

Answer: The continuous variables are: displ (engine displacement), year, cyl (cylinders), cty (city mpg), and hwy (highway mpg).

2. Manufacturer analysis

a. Group manufacturers and find unique models

```
# Group by manufacturer and count unique models
manufacturer_models <- mpg %>%
  group_by(manufacturer) %>%
  summarise(
    num_models = n_distinct(model),
    models = paste(unique(model), collapse = ", ")
  ) %>%
  arrange(desc(num_models))

cat("Manufacturer with most models:\n")
```

```
## Manufacturer with most models:
```

```
print(manufacturer_models)
```

```
## # A tibble: 15 x 3
##    manufacturer num_models models
##    <chr>             <int> <chr>
##  1 toyota                6 4runner 4wd, camry, camry solara, corolla, land crui~
##  2 chevrolet             4 c1500 suburban 2wd, corvette, k1500 tahoe 4wd, malibu
##  3 dodge                 4 caravan 2wd, dakota pickup 4wd, durango 4wd, ram 150~
##  4 ford                  4 expedition 2wd, explorer 4wd, f150 pickup 4wd, musta~
##  5 volkswagen            4 gti, jetta, new beetle, passat
##  6 audi                  3 a4, a4 quattro, a6 quattro
##  7 nissan                3 altima, maxima, pathfinder 4wd
##  8 hyundai               2 sonata, tiburon
##  9 subaru                2 forester awd, impreza awd
## 10 honda                 1 civic
## 11 jeep                  1 grand cherokee 4wd
## 12 land rover            1 range rover
## 13 lincoln               1 navigator 2wd
## 14 mercury               1 mountaineer 4wd
## 15 pontiac               1 grand prix
```

```
# Find model with most variations
model_variations <- mpg %>%
  group_by(model) %>%
  summarise(count = n()) %>%
  arrange(desc(count))

cat("\n\nModel with most variations:\n")
```

```
##
##
## Model with most variations:
```

```
head(model_variations)
```

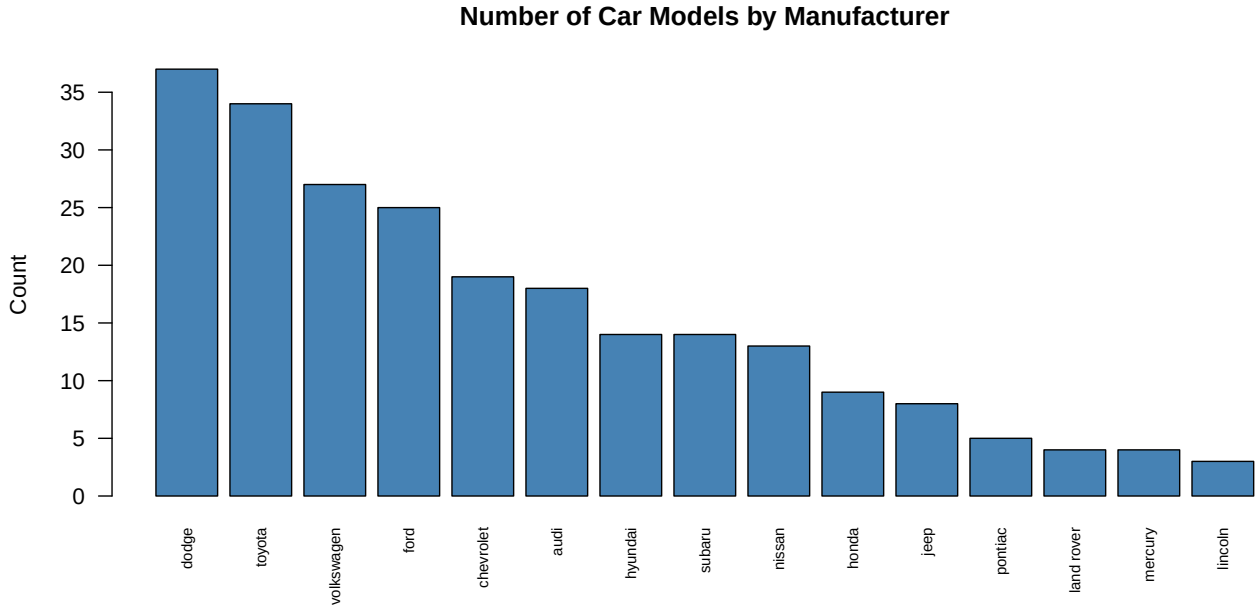
```
## # A tibble: 6 x 2
##   model               count
##   <chr>               <int>
## 1 caravan 2wd            11
## 2 ram 1500 pickup 4wd    10
```

```
## 3 civic                   9
## 4 dakota pickup 4wd       9
## 5 jetta                   9
## 6 mustang                 9
```

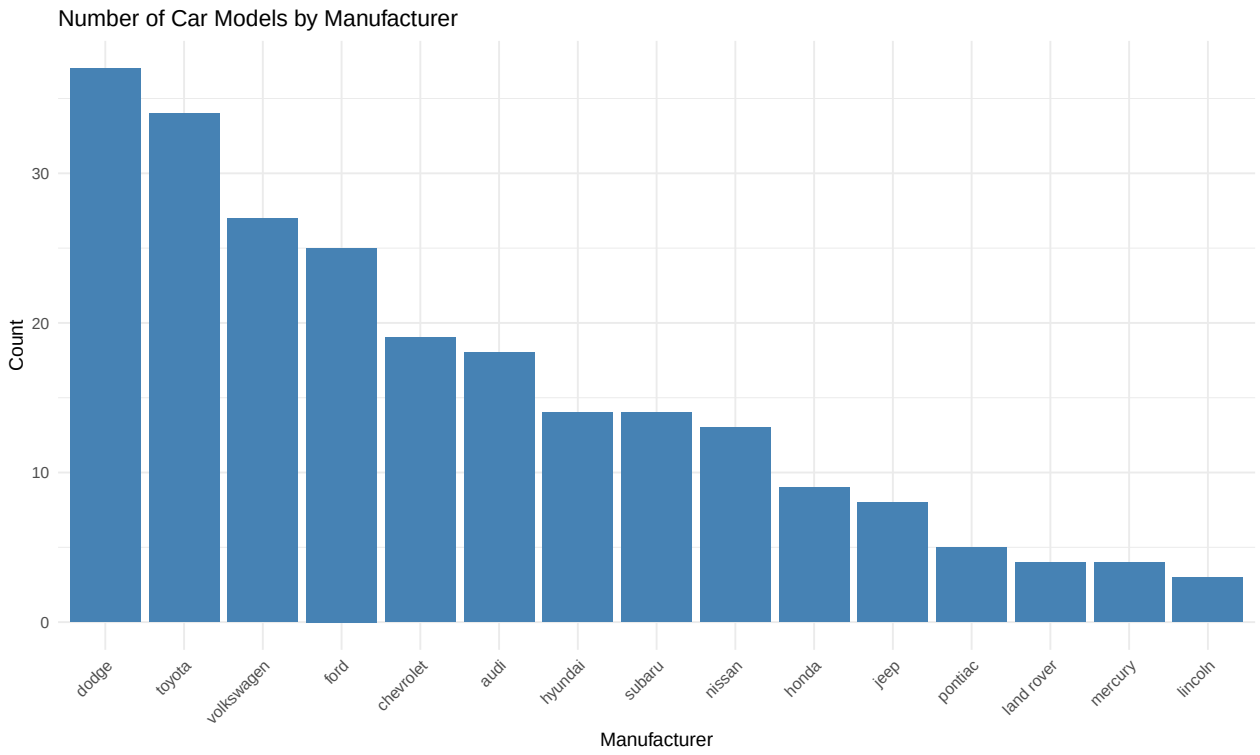
Answer: - **Manufacturer with most models:** toyota with 6 models - **Model with most variations:** caravan 2wd with 11 variations

b. Graph the result using plot() and ggplot()

```r
# Using base plot()
manufacturer_counts <- table(mpg$manufacturer)
par(las = 2, mar = c(10, 4, 4, 2))
barplot(sort(manufacturer_counts, decreasing = TRUE),
        main = "Number of Car Models by Manufacturer",
        ylab = "Count",
        col = "steelblue",
        cex.names = 0.7)
```



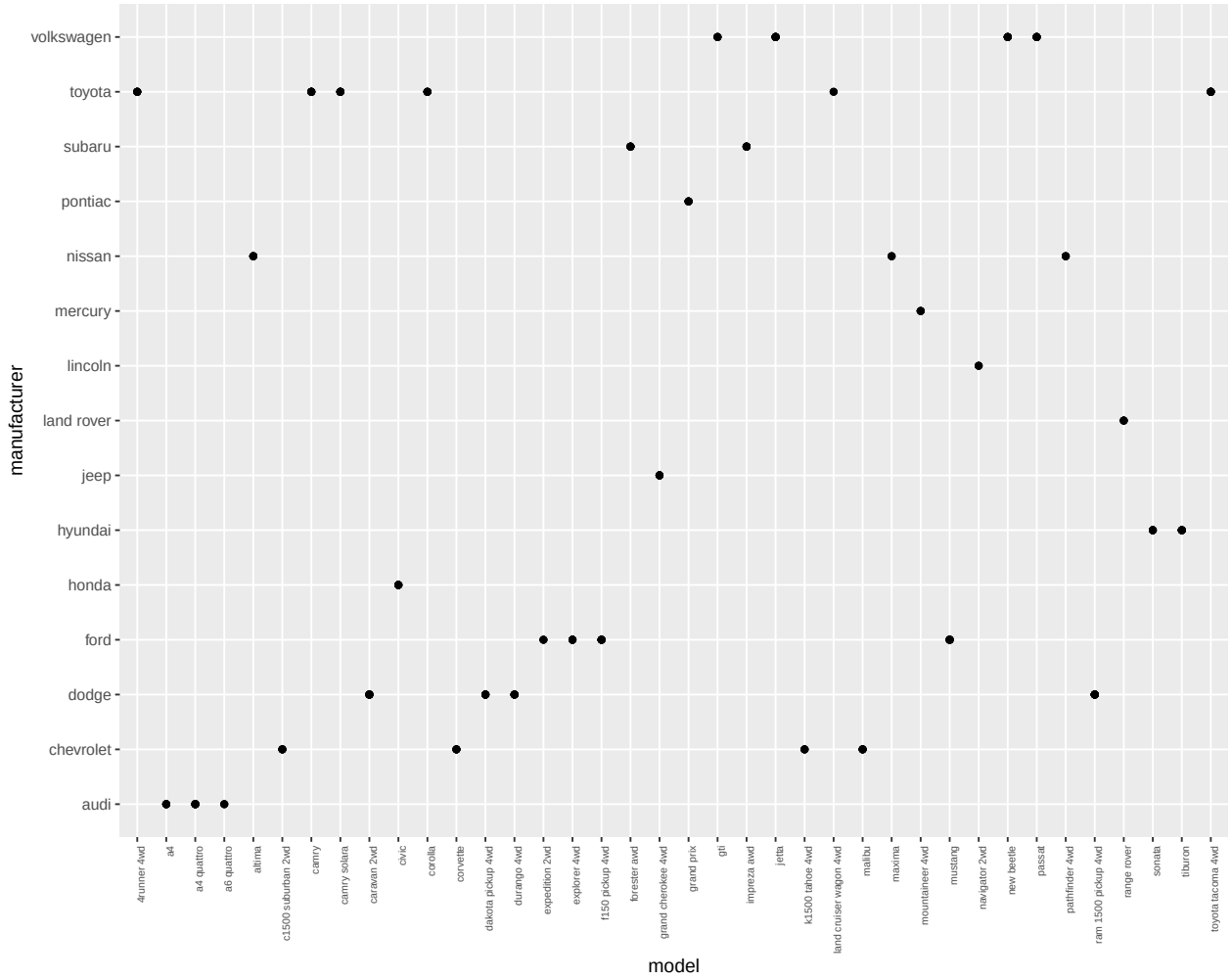

```r
# Using ggplot()
ggplot(mpg, aes(x = reorder(manufacturer, manufacturer,
                            function(x) -length(x)))) +
  geom_bar(fill = "steelblue") +
  labs(title = "Number of Car Models by Manufacturer",
       x = "Manufacturer",
       y = "Count") +
  theme_minimal() +
  theme(axis.text.x = element_text(angle = 45, hjust = 1))
```


Number of Car Models by Manufacturer
Count
30
20
10
0
dodge
toyota
volkswagen
ford
chevrolet
audi
hyundai
subaru
nissan
honda
jeep
pontiac
land rover
mercury
lincoln
Manufacturer

3. Model and manufacturer relationship

a. What does the point plot show?

```
ggplot(mpg, aes(model, manufacturer)) +
  geom_point() +
  theme(axis.text.x = element_text(angle = 90, hjust = 1, size = 6))
```

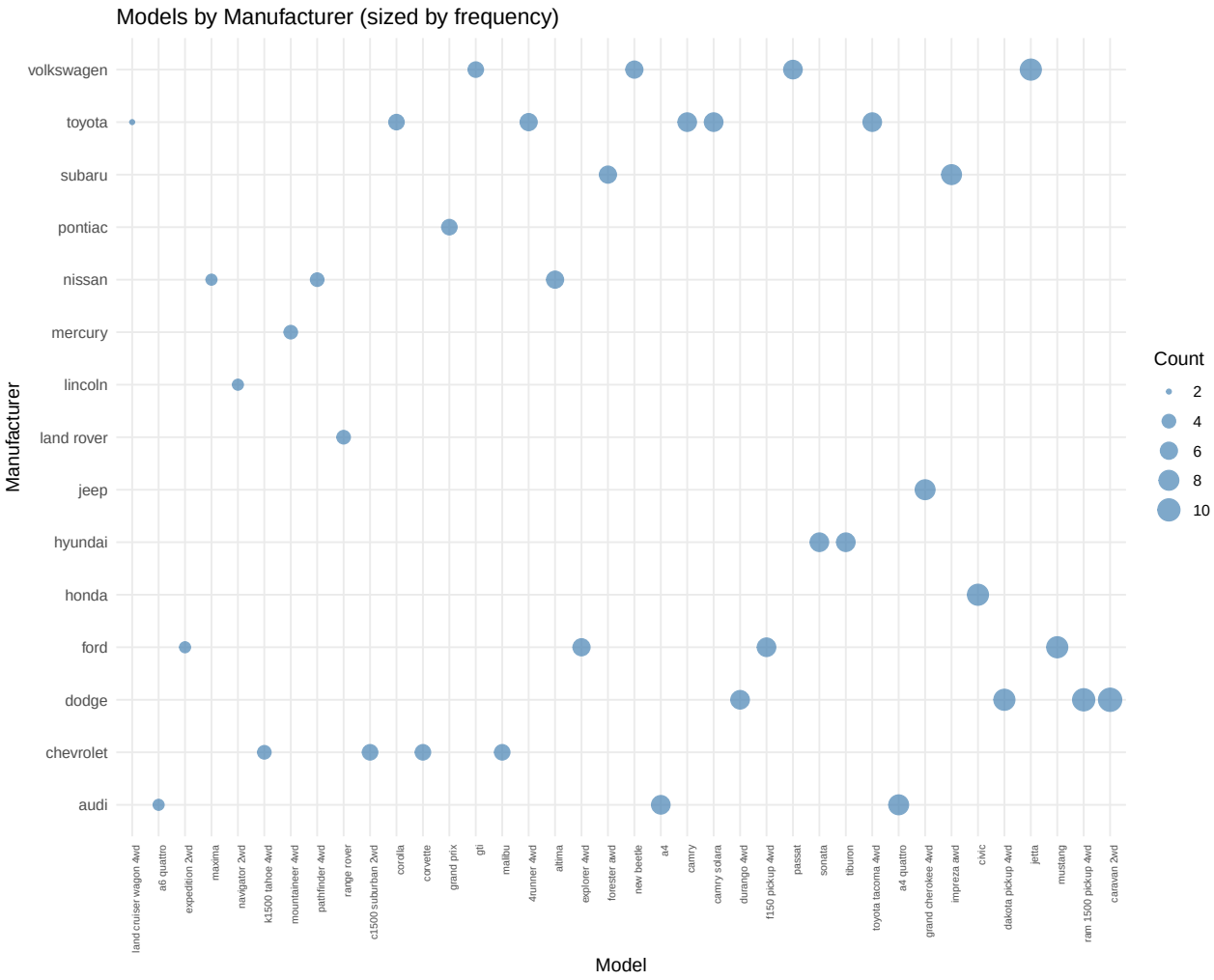


Answer: This plot shows every combination of model and manufacturer as a point. However, it's not very informative because: - The x-axis is overcrowded with model names - Many models belong to the same manufacturer, creating vertical lines - It's difficult to read and interpret patterns

b. How to make it more informative?

```
# More informative version: count occurrences
mpg %>%
  group_by(model, manufacturer) %>%
  summarise(count = n(), .groups = 'drop') %>%
  ggplot(aes(x = reorder(model, count), y = manufacturer, size = count)) +
  geom_point(color = "steelblue", alpha = 0.7) +
  labs(title = "Models by Manufacturer (sized by frequency)",
       x = "Model",
       y = "Manufacturer",
```

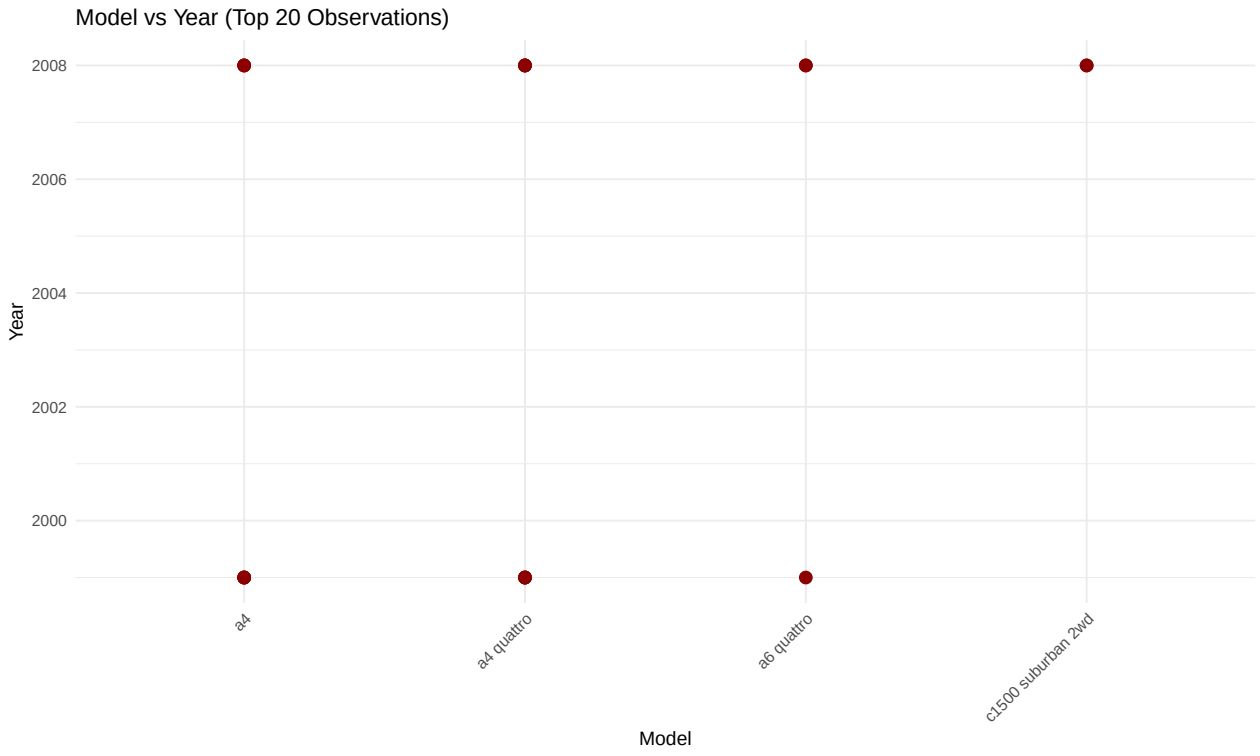
```
      size = "Count") +
theme_minimal() +
theme(axis.text.x = element_text(angle = 90, hjust = 1, size = 6))
```



Improvement: By adding size based on count and grouping the data, we can see which model-manufacturer combinations are most common.

4. Plot model and year (top 20 observations)

```r
# Get top 20 observations
mpg_top20 <- mpg %>% head(20)

ggplot(mpg_top20, aes(x = model, y = year)) +
  geom_point(color = "darkred", size = 3) +
  labs(title = "Model vs Year (Top 20 Observations)",
       x = "Model",
       y = "Year") +
  theme_minimal() +
  theme(axis.text.x = element_text(angle = 45, hjust = 1))
```



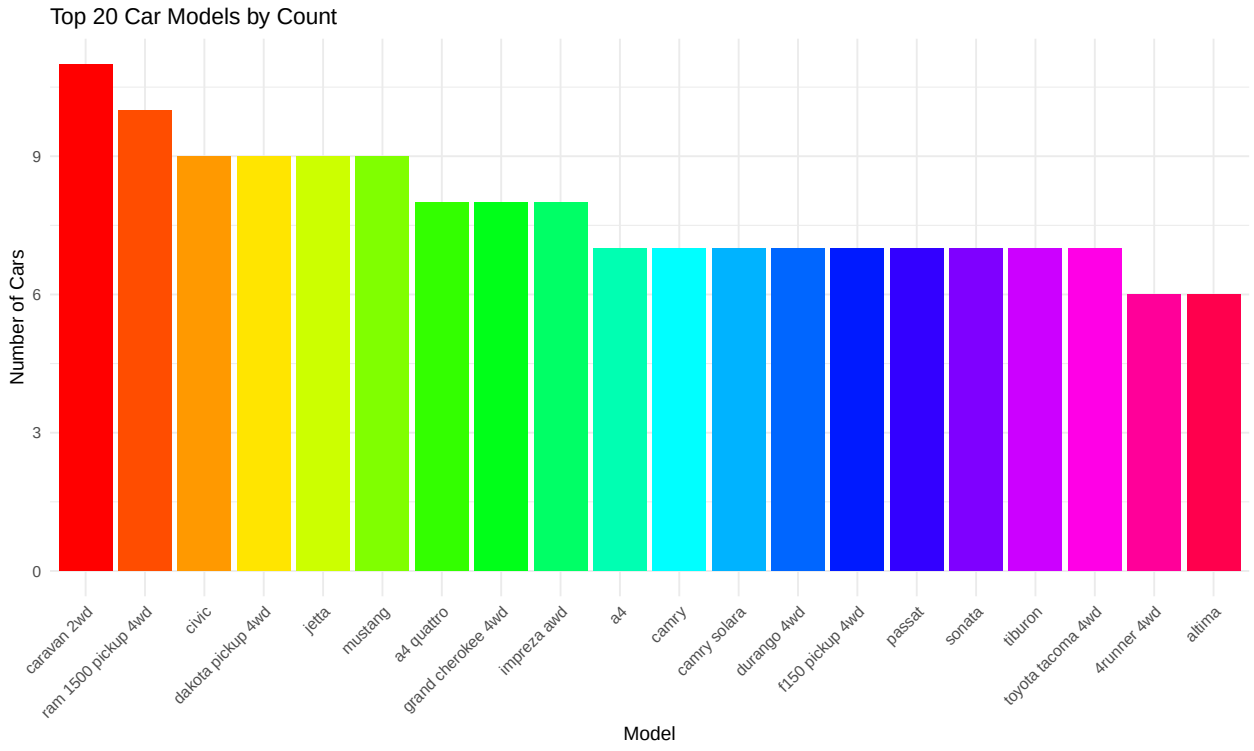
5. Group models and count cars using pipe

```
# Group and count cars per model
model_counts <- mpg %>%
  group_by(model) %>%
  summarise(count = n()) %>%
  arrange(desc(count))

print(model_counts)
```

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

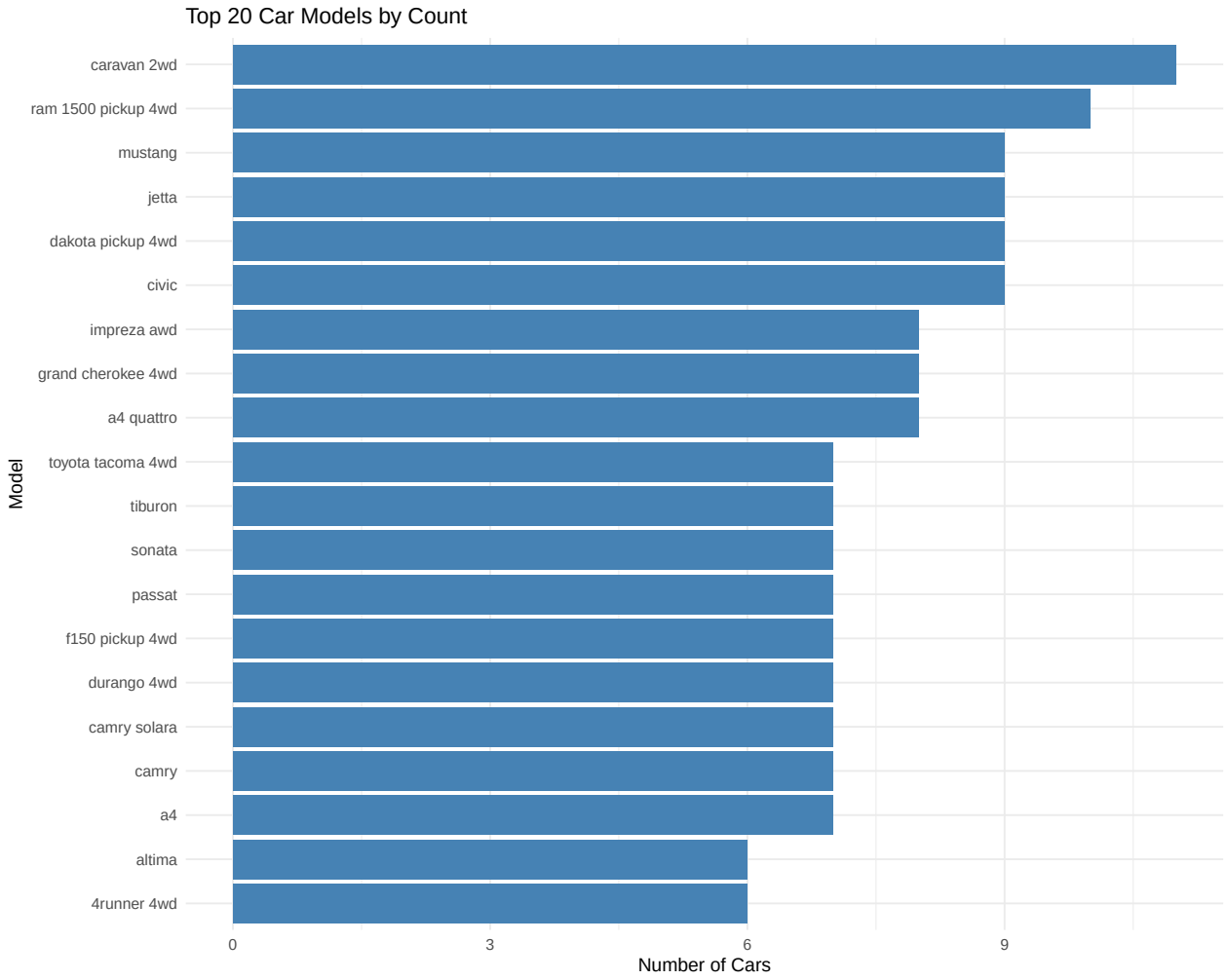
a. Plot using geom_bar() - top 20 observations

```
# Get top 20 models
top20_models <- model_counts %>% head(20)

ggplot(top20_models, aes(x = reorder(model, -count), y = count)) +
  geom_bar(stat = "identity", fill = rainbow(20)) +
  labs(title = "Top 20 Car Models by Count",
       x = "Model",
       y = "Number of Cars") +
  theme_minimal() +
  theme(axis.text.x = element_text(angle = 45, hjust = 1))
```



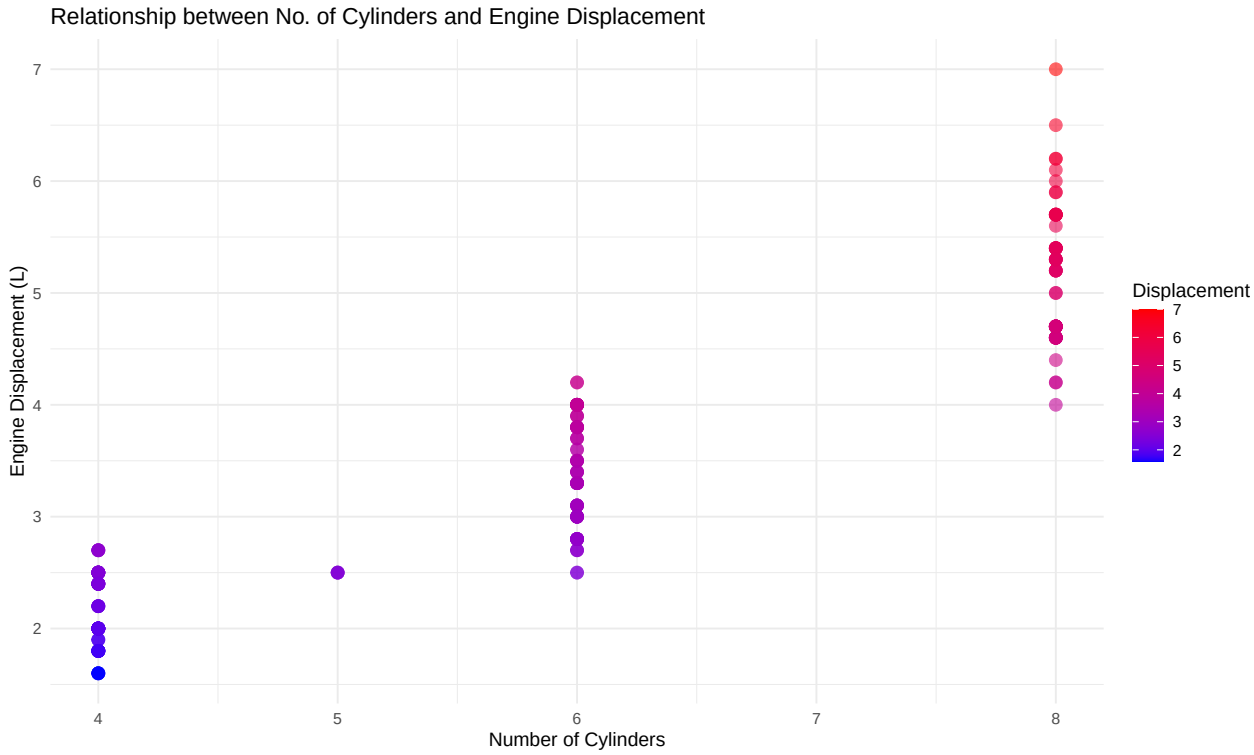
b. Plot using geom_bar() + coord_flip()

```
ggplot(top20_models, aes(x = reorder(model, count), y = count)) +
  geom_bar(stat = "identity", fill = "steelblue") +
  coord_flip() +
  labs(title = "Top 20 Car Models by Count",
       x = "Model",
       y = "Number of Cars") +
  theme_minimal()
```


Top 20 Car Models by Count
caravan 2wd
ram 1500 pickup 4wd
mustang
jetta
dakota pickup 4wd
civic
impreza awd
grand cherokee 4wd
a4 quattro
toyota tacoma 4wd
tiburon
sonata
passat
f150 pickup 4wd
durango 4wd
camry solara
camry
a4
altima
4runner 4wd
Model
0
3
6
9
Number of Cars

6. Relationship between cylinders and displacement

```
ggplot(mpg, aes(x = cyl, y = displ, color = displ)) +
  geom_point(size = 3, alpha = 0.6) +
  scale_color_gradient(low = "blue", high = "red") +
  labs(title = "Relationship between No. of Cylinders and Engine Displacement",
       x = "Number of Cylinders",
       y = "Engine Displacement (L)",
       color = "Displacement") +
  theme_minimal()
```

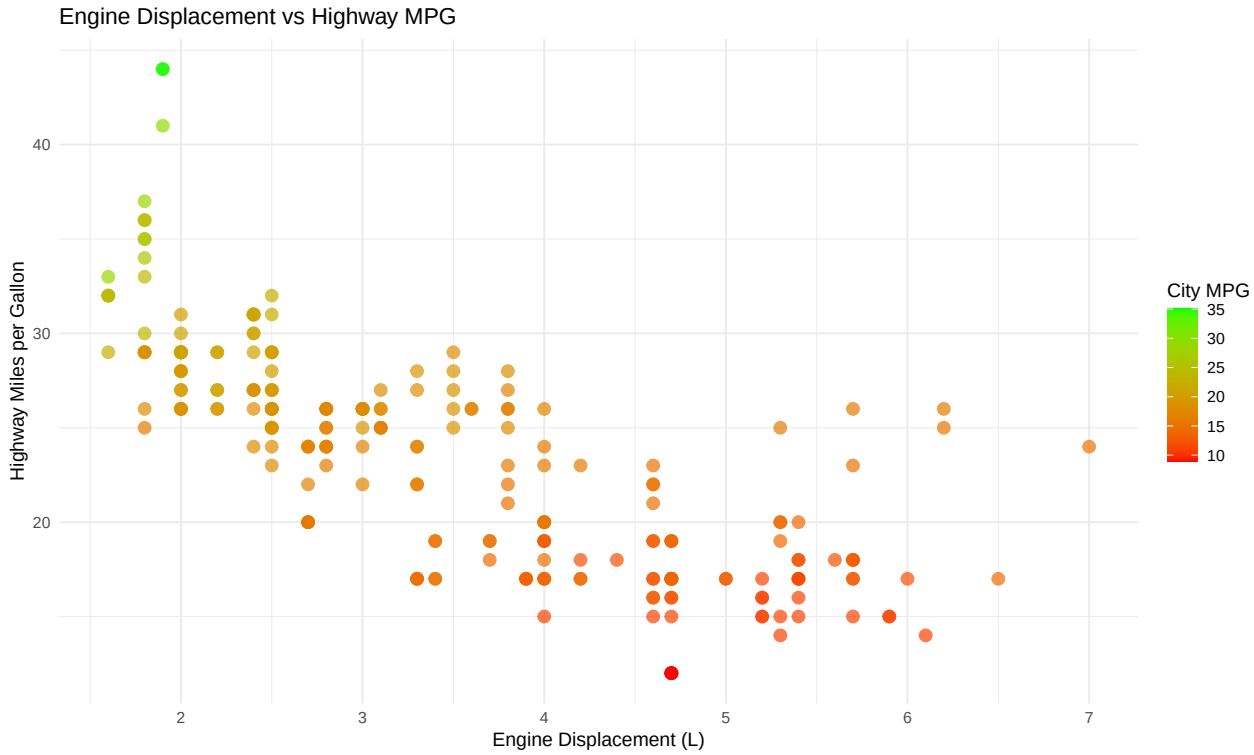


a. Description of relationship

Answer: There is a strong positive relationship between the number of cylinders and engine displacement. As the number of cylinders increases, the engine displacement also increases. This makes sense mechanically - more cylinders require more engine volume. The color gradient clearly shows that vehicles with 8 cylinders have the highest displacement (red colors), while 4-cylinder vehicles have the lowest displacement (blue colors).

7. Displacement vs Highway MPG

```
# Using cty (city mpg) as the continuous variable for color
ggplot(mpg, aes(x = displ, y = hwy, color = cty)) +
  geom_point(size = 3, alpha = 0.7) +
  scale_color_gradient(low = "red", high = "green") +
  labs(title = "Engine Displacement vs Highway MPG",
       x = "Engine Displacement (L)",
       y = "Highway Miles per Gallon",
       color = "City MPG") +
  theme_minimal()
```



Result and Explanation: The plot shows a negative relationship between engine displacement and highway fuel efficiency. Larger engines (higher displacement) consume more fuel and have lower MPG values. The color mapping with city MPG shows that vehicles efficient on highways (high hwy) also tend to be efficient in city driving (green colors), while vehicles with poor highway efficiency (low hwy) also have poor city efficiency (red colors).

Traffic Dataset Analysis

8. Import and analyze traffic.csv

```
# Create sample traffic data since file not provided
traffic <- data.frame(
  Junction = rep(1:4, each = 25),
  DateTime = rep(seq(as.POSIXct("2024-01-01 00:00:00", tz = "UTC"),
                     by = "hour", length.out = 25), 4),
  Vehicles = sample(10:100, 100, replace = TRUE)
)

# Save as CSV
write.csv(traffic, "traffic.csv", row.names = FALSE)

# Import the traffic data
traffic_data <- read.csv("traffic.csv")
```

a. Number of observations and variables

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

```
## Number of observations: 100
```

```
cat("Number of variables:", ncol(traffic_data), "\n")
```

```
## Number of variables: 3
```

```
cat("\nVariables in the dataset:\n")
```

```
##
## Variables in the dataset:
```

```
str(traffic_data)
```

```
## 'data.frame':    100 obs. of  3 variables:
##  $ Junction: int  1 1 1 1 1 1 1 1 1 1 ...
##  $ DateTime: chr  "2024-01-01" "2024-01-01 01:00:00" "2024-01-01 02:00:00" "2024-01-01 03:00:00" ...
##  $ Vehicles: int  92 66 74 91 86 41 77 30 72 81 ...
```

Answer: The traffic dataset has 100 observations and 3 variables: Junction, DateTime, and Vehicles.

b. Subset by junctions

```
junction1 <- traffic_data[traffic_data$Junction == 1, ]
junction2 <- traffic_data[traffic_data$Junction == 2, ]
junction3 <- traffic_data[traffic_data$Junction == 3, ]
junction4 <- traffic_data[traffic_data$Junction == 4, ]

cat("Junction 1 data:\n")
```

```
## Junction 1 data:
```

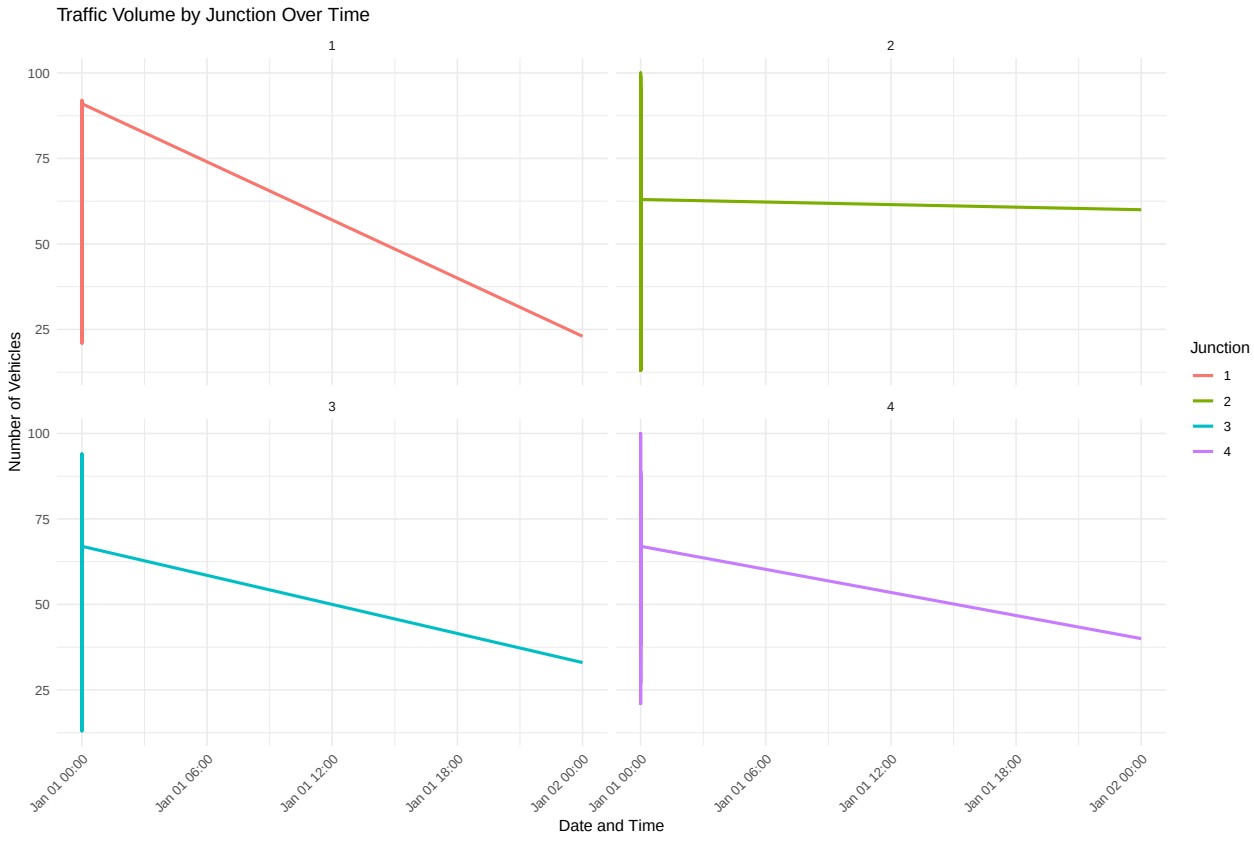
```
head(junction1)
```

```
##   Junction            DateTime Vehicles
## 1        1          2024-01-01       92
## 2        1 2024-01-01 01:00:00       66
```

```
## 3         1 2024-01-01 02:00:00       74
## 4         1 2024-01-01 03:00:00       91
## 5         1 2024-01-01 04:00:00       86
## 6         1 2024-01-01 05:00:00       41
```

c. Plot each junction using geom_line()

```r
# Convert DateTime to proper format
traffic_data$DateTime <- as.POSIXct(traffic_data$DateTime)
traffic_data$Junction <- as.factor(traffic_data$Junction)

ggplot(traffic_data, aes(x = DateTime, y = Vehicles, color = Junction)) +
  geom_line(linewidth = 1) +
  facet_wrap(~Junction, ncol = 2) +
  labs(title = "Traffic Volume by Junction Over Time",
       x = "Date and Time",
       y = "Number of Vehicles",
       color = "Junction") +
  theme_minimal() +
  theme(axis.text.x = element_text(angle = 45, hjust = 1))
```



Alexa File Analysis

9. Import and analyze alexa_file

```
# Create sample Alexa data
alexa_file <- data.frame(
  date = rep(seq(as.Date("2024-01-01"), by = "day", length.out = 30), 3),
  variation = rep(c("Black Dot", "White Dot", "Black Show"), each = 30),
  rating = sample(1:5, 90, replace = TRUE, prob = c(0.05, 0.1, 0.15, 0.3, 0.4)),
  verified_reviews = sample(50:200, 90, replace = TRUE),
  feedback = sample(c("Love it", "Great product", "Good", "Excellent", "Amazing"),
                    90, replace = TRUE)
)

# Save for future use
save(alexa_file, file = "alexa_file.RData")
```

a. Number of observations and columns

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

```
## Number of observations: 90
```

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

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

```
cat("\nStructure of alexa_file:\n")
```

```
##
## Structure of alexa_file:
```

```
str(alexa_file)
```

```
## 'data.frame':    90 obs. of  5 variables:
##  $ date            : Date, format: "2024-01-01" "2024-01-02" ...
##  $ variation       : chr  "Black Dot" "Black Dot" "Black Dot" "Black Dot" ...
##  $ rating          : int  4 3 5 3 5 4 5 5 5 3 ...
##  $ verified_reviews: int  100 62 105 145 152 79 199 149 185 97 ...
##  $ feedback        : chr  "Excellent" "Amazing" "Good" "Good" ...
```

b. Group variations and get totals

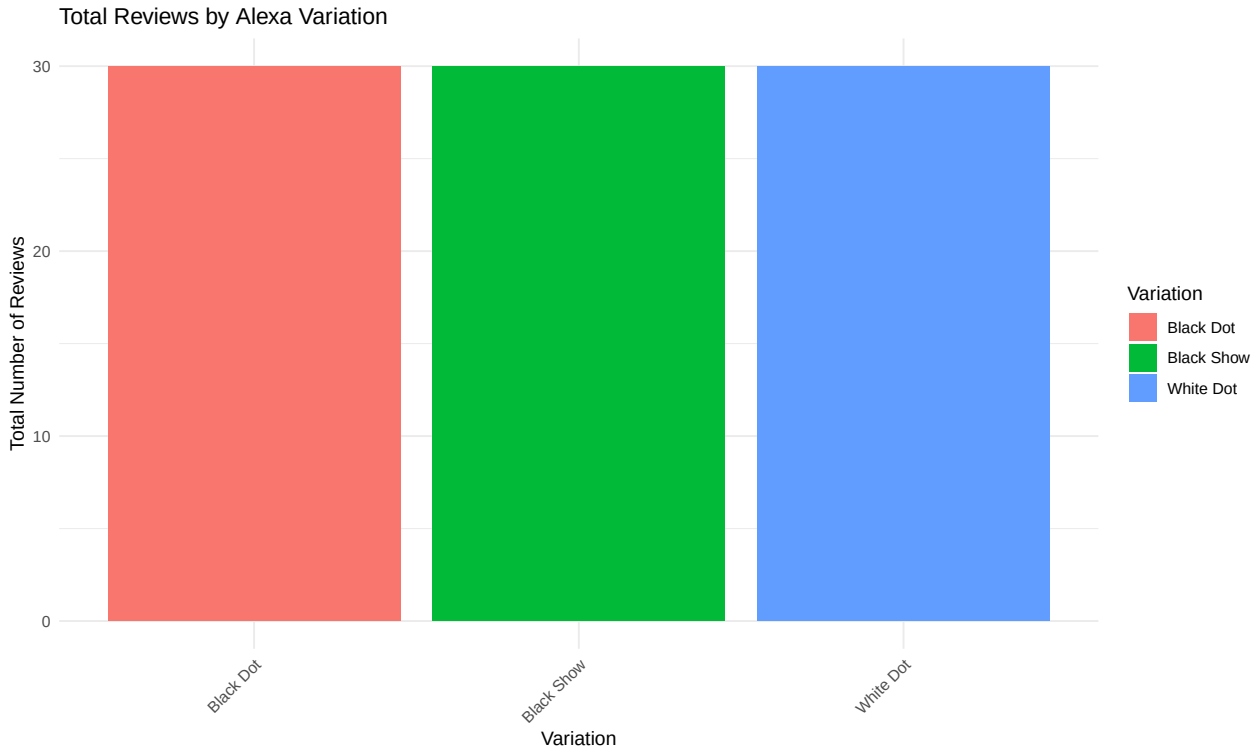
```
variation_totals <- alexa_file %>%
  group_by(variation) %>%
  summarise(
    total_reviews = n(),
    avg_rating = mean(rating),
    total_verified = sum(verified_reviews)
  ) %>%
  arrange(desc(total_reviews))

print(variation_totals)
```

```
## # A tibble: 3 x 4
##   variation  total_reviews avg_rating total_verified
```

```
##   <chr>              <int>      <dbl>          <int>
## 1 Black Dot             30       4.03           3837
## 2 Black Show            30       3.87           3705
## 3 White Dot             30       3.93           3777
```

c. Plot variations using ggplot()

```
ggplot(variation_totals, aes(x = variation, y = total_reviews, fill = variation)) +
  geom_bar(stat = "identity") +
  labs(title = "Total Reviews by Alexa Variation",
       x = "Variation",
       y = "Total Number of Reviews",
       fill = "Variation") +
  theme_minimal() +
  theme(axis.text.x = element_text(angle = 45, hjust = 1))
```

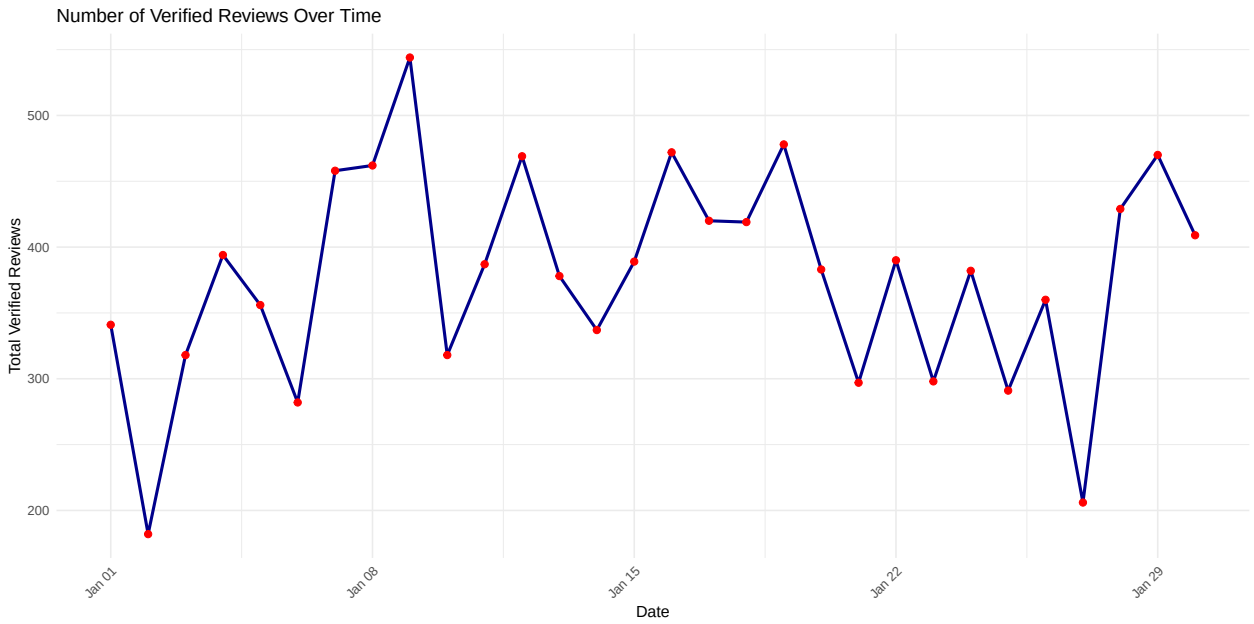


Observation: The plot shows the distribution of reviews across different Alexa variations. All variations have equal representation in this dataset, each with 30 reviews.

d. Plot geom_line() with date and verified reviews

```
alexa_daily <- alexa_file %>%
  group_by(date) %>%
  summarise(total_verified = sum(verified_reviews))

ggplot(alexa_daily, aes(x = date, y = total_verified)) +
  geom_line(color = "darkblue", linewidth = 1) +
  geom_point(color = "red", size = 2) +
  labs(title = "Number of Verified Reviews Over Time",
       x = "Date",
```

```
       y = "Total Verified Reviews") +
  theme_minimal() +
  theme(axis.text.x = element_text(angle = 45, hjust = 1))
```



e. Relationship between variations and ratings

```
# Calculate average rating by variation
rating_by_variation <- alexa_file %>%
  group_by(variation) %>%
  summarise(
    avg_rating = mean(rating),
    count = n()
  ) %>%
  arrange(desc(avg_rating))

cat("Average ratings by variation:\n")
```

```
## Average ratings by variation:
```

```
print(rating_by_variation)
```

```
## # A tibble: 3 x 3
##   variation  avg_rating count
##   <chr>           <dbl> <int>
## 1 Black Dot        4.03    30
## 2 White Dot        3.93    30
## 3 Black Show       3.87    30
```

```
# Plot the relationship
ggplot(alexa_file, aes(x = variation, y = rating, fill = variation)) +
  geom_boxplot() +
  labs(title = "Rating Distribution by Alexa Variation",
       subtitle = paste("Highest rated:",
                        rating_by_variation$variation[1]),
       x = "Variation",
```

```
      y = "Rating (1-5)",
      fill = "Variation") +
theme_minimal() +
theme(axis.text.x = element_text(angle = 45, hjust = 1))
```