Predicting Vehicle MPG

Presented to

CSUF

COLLEGE OF

Natural Sciences and Mathematics

California State University, Fullerton

Math 533 Fall 2023

Prepared by

Emilio VASQUEZ

September 23, 2023

Contents

1 Overview

Using the provided data from the Auto dataset, the task at hand requires us to build a classification model that classifies if a vehicle gets high gas mileage or low gas mileage. The way a vehicle was deemed having high gas mileage was if the mile-per-gallon (MPG) was above the median 22.75 MPG. Within the dataset predictive models will be built around the following predictor variables

- mpg - Miles per gallon
- cylinders - Number of cylinders between 4 and 8
- displacement - Engine displacement (cu. inches)
- horsepower - Engine horsepower
- weight - Vehicle weight (lbs.)
- acceleration - Time to accelerate from 0 to 60 mph (sec.)
- year - Model year (MY) between 1970 and 1982
- origin - Origin of car (1. American, 2. European, 3. Japanese)
- name - Vehicle name with brand
- brand - Extraction of brand from name variable

2 Data Exploration

2.1 MPG Analysis - Qualitative

Given that the task requires exploring classifying vehicles with high and low gas mileage, exploring MPG will be the jumping off point for this analysis.

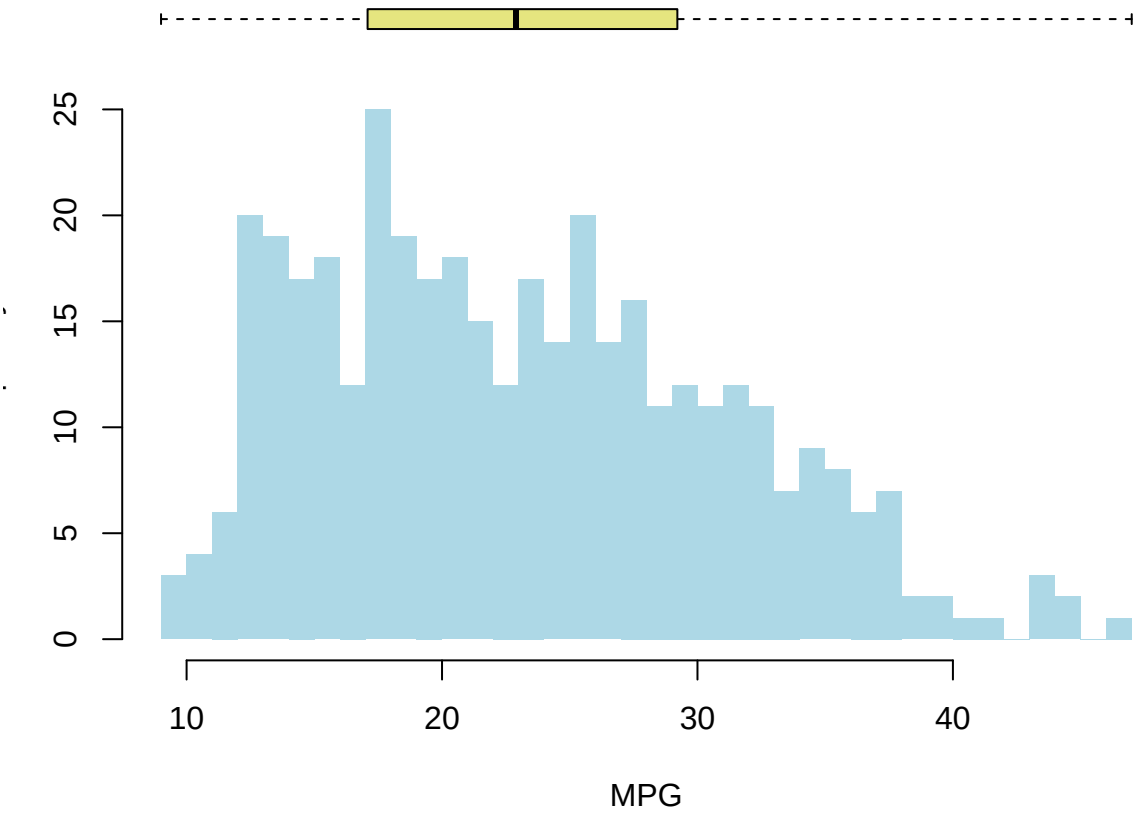

Figure 1. Distribution of MPG via histogram and boxplot directly above.

MPG itself has a wide spread with vehicles having as low as 9 MPG to vehicles as high as 46.60 MPG. This is quite a huge range and it is impressive to see that cars from the 70s and 80s had MPG as high as the 40s as this was long before the advent of hybrid vehicles. The median as shown in the boxplot above the histogram is 22.75. So vehicles higher than this will be classified as high mpg and vehicles to the left of this will be classified as low mpg.

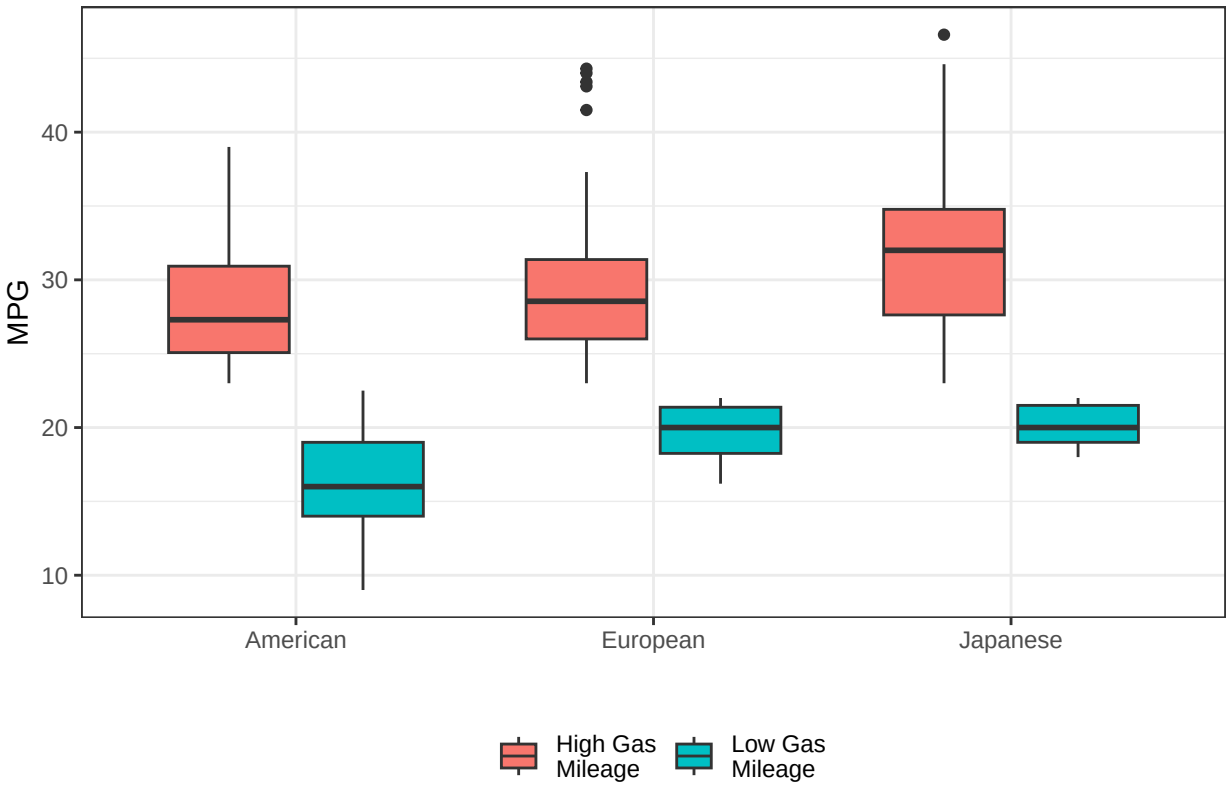

Figure 2. Boxplots of MPG by origin of vehicle

In the figure above, one can see the distribution of MPG based off where the vehicle was manufactured. Cars that were of Japanese origin had the best in class high gas mileage. They had the largest spread of high MPG while simultaneously having the lowest spread in low MPG as denoted by the large and small IQR boxes respectively. Interestingly enough, this sentiment still holds largely true within the automotive industry.

Between American and European there is not much difference between the high MPG groups, but there is most certainly a difference between the low MPG. American cars have a large range of low MPG while European is small. This might potentially be attributed to a smaller overall sample size for European cars (n=68) when compared to American (n=245).

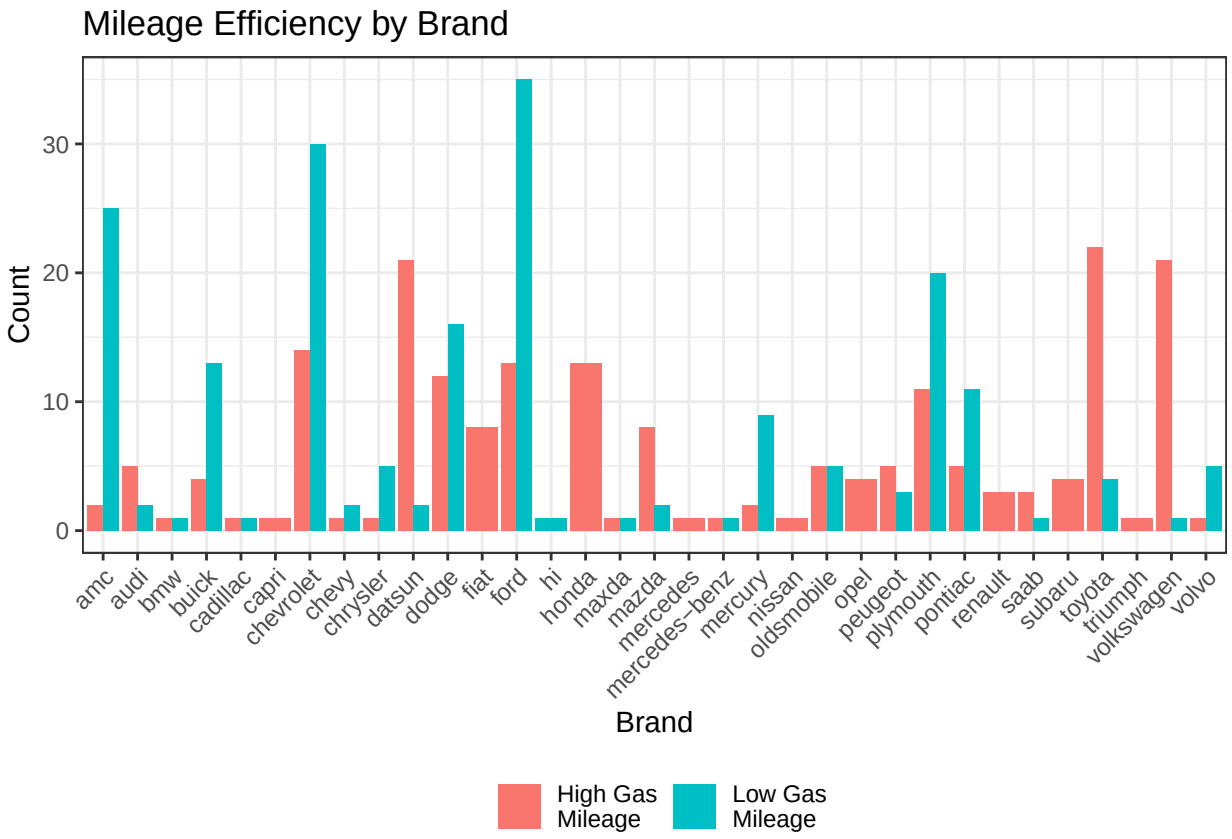

Figure 3. Brand overview of mileage efficiency

Above are all the auto manufacturers that were provided in the dataset and a count of how many of their vehicles had low and high gas mileage. The best in class brand for high gas mileage was Toyota, followed closely by Volkswagen and Datsun. Some brands in this time frame based off the dataset may have only supplied high mileage vehicles such as Honda, Fiat, and Renault. Ford, Chevrolet, and AMC which all happen to be of American origin had the most vehicles with low gas mileage.

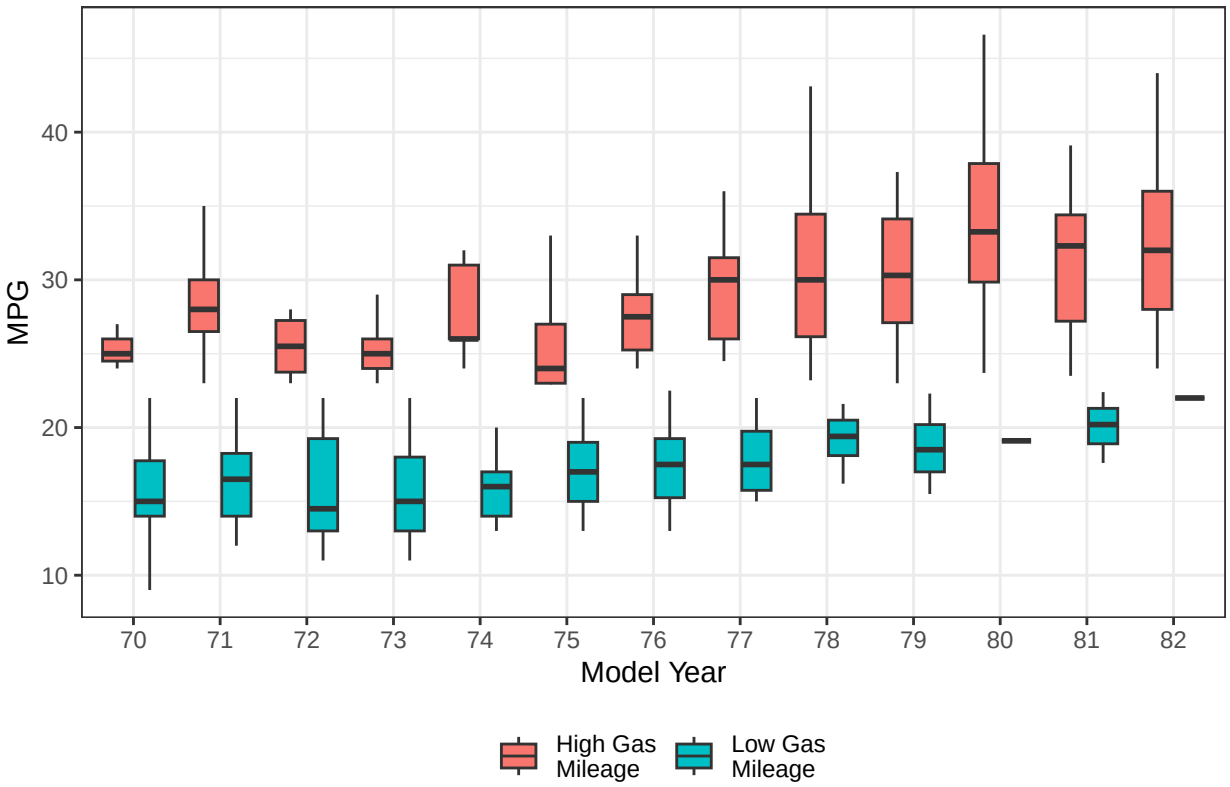

Figure 4. Model Year overview of mileage efficiency

Figure 4 shows how the distribution of MPG has changed with each model year since 1970 for vehicles considered low and high gas mileage. As time moved on, vehicles improved their gas mileage significantly based off the boxplot using the provided data. The range of high MPG vehicles was very small for 1970 but had a large range for 1982.

2.2 MPG Analysis - Quantitative

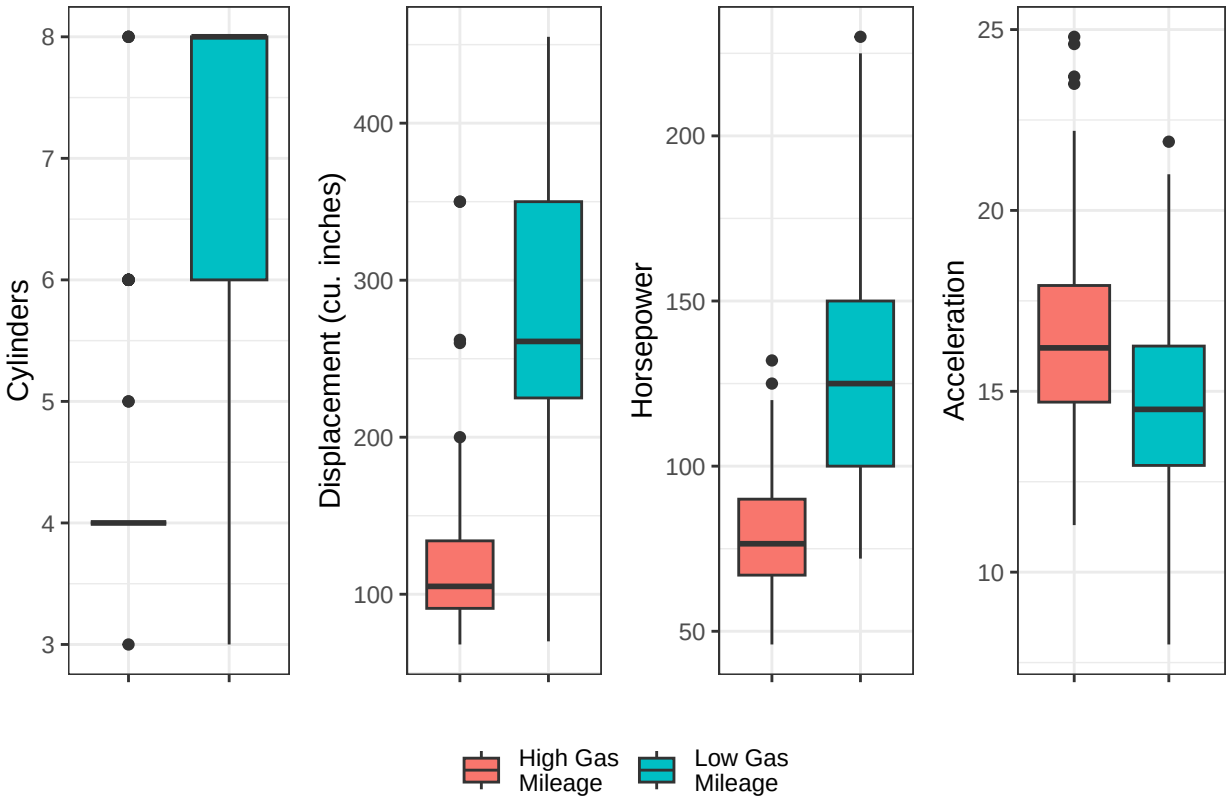

Figure 5. Model Year overview of mileage efficiency

Above are boxplots of how the high/low MPG fares with more quantitative variables relating to engine.

Starting from the left, it appears that pretty much all vehicles with 4 cylinders will have high MPG with a few exceptions of course. There is a pretty clear cut division between displacement as well. If a vehicle has displacement under 200 cubic inches, the vehicle might be considered high mileage.

When viewing horsepower and acceleration is where things are not as clear cut. There is some overlap between the high/low MPG when looking at horsepower. Looking at the far most right graph for acceleration, there is not much variation relatively speaking between the high/low MPG groups.

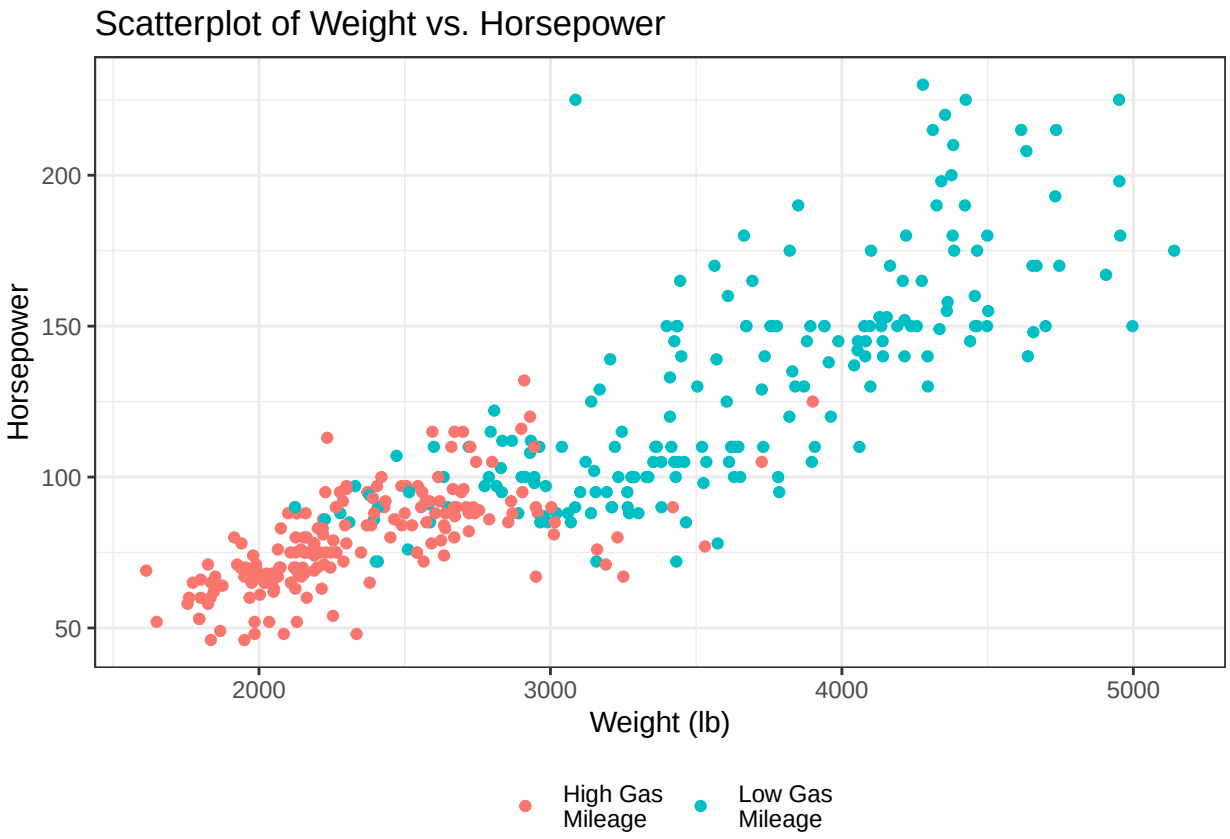

Figure 6. Scatterplot of Weight Vs Horsepower

Above one can see a scatterplot of horsepower and weight with the point color coded by high/low MPG. There is an almost clean-cut decision boundary made somewhere around the weight being 3000 and the horsepower being around 90. One thing is clear from this graph: the higher the horsepower and weight, the lower the MPG.

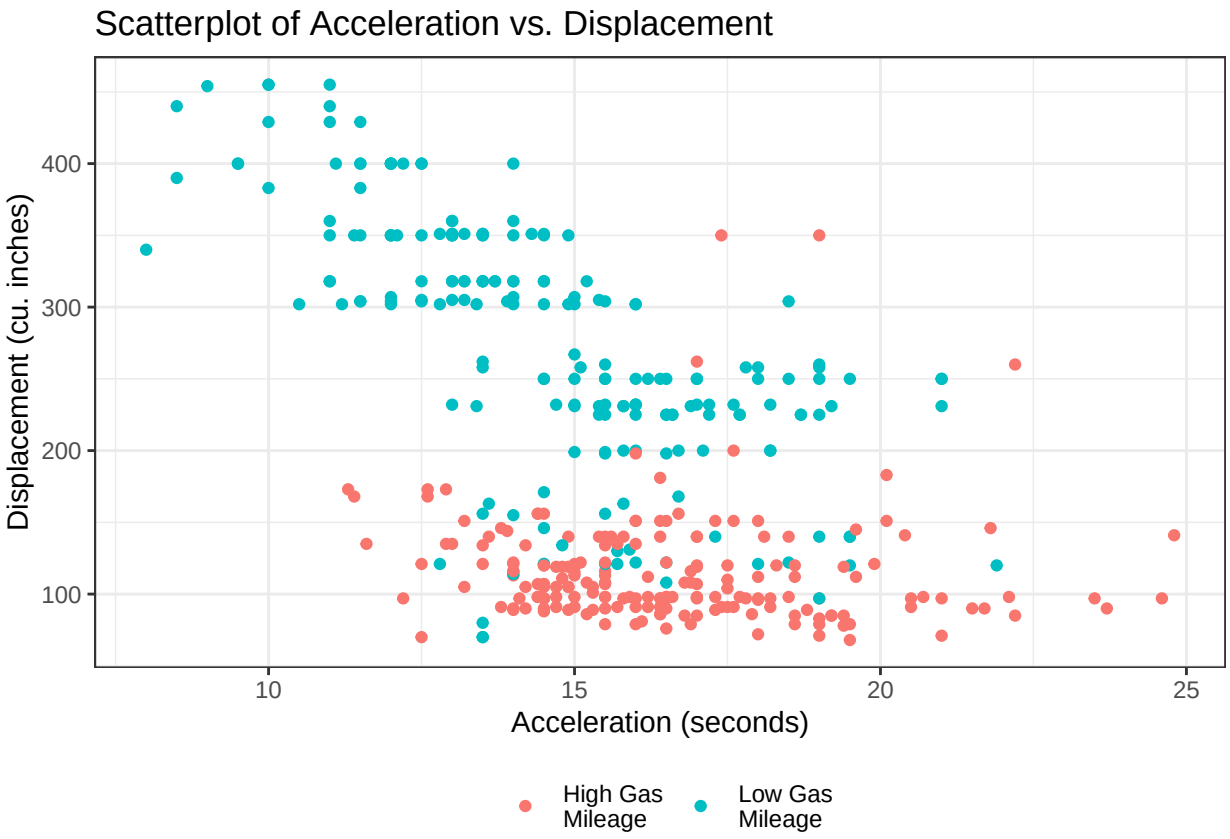

Figure 7. Scatterplot of Displacement Vs Acceleration

Above in figure 7, there is some separation of points when viewing high/low MPG. There is a downward trend in the points overall which makes sense because engines with a larger displacement tend to produce more power and torque compared to engines with smaller displacements which in turn produces a faster 0-60 acceleration time. Vehicles under 200 cu. inches in displacement have overwhelmingly high gas mileage, but there is a mix of low MPG vehicles as well.

3 Building Predictive Models

The following models were created and evaluated using a 70/30 train/test split:

- LDA
- QDA
- Logistic regression
- Naive Bayes
- KNN (Most Optimal K = 10)

Of the available predictors, the following were used to predict if a vehicle has high/low MPG.

- cylinders
- displacement
- horsepower
- weight

Once the models were created using the train data, the test set was evaluated for each model to determine the test error. The test error is percentage of times the model guessed incorrectly. The model with the lowest test error rate could be deemed the best model if prediction is the goal of the individual.

Technique	Test Error Rate (%)
LDA	11.01
QDA	10.16
Logistic Regression	11.01
Naive Bayes	10.16
KNN (K = 10)	8.47

Figure 8. Table of test test error results

In the above figure, LDA and logistic regression performed the worst of the bunch at an error rate of both 11.01%. The best performing model happened to be KNN with K = 10 which had an error rate of 8.47%.

Although the final K was 10, below are various values tried for K. Overall the general trend is the higher the K chosen, the lower the test error rate.

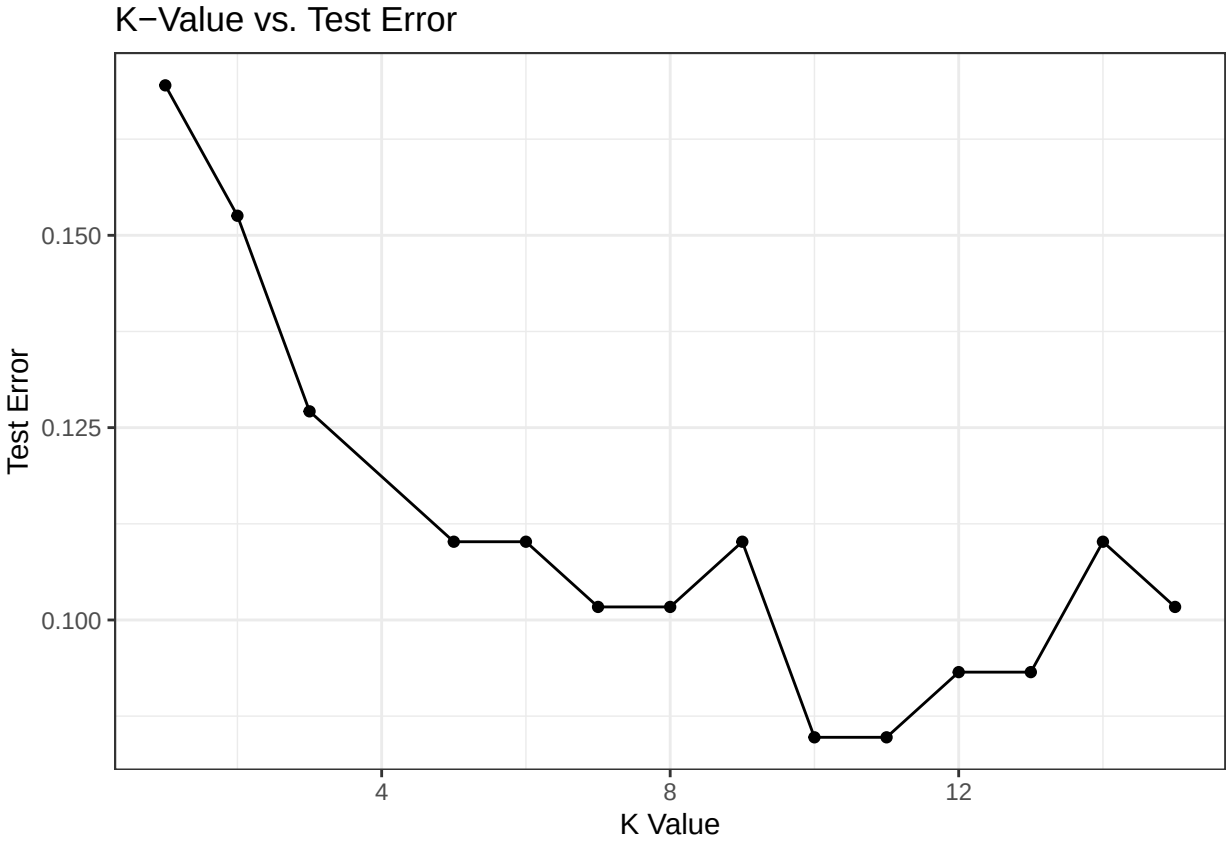

Figure 9. Test error rate as K value increases

4 Appendix

4.1 Part a code

```
data(Auto)
data <- Auto
str(data)
mpg_median <- median(Auto$mpg)
cat("mpg_median: ", mpg_median)
data$mpg01 <- as.factor(ifelse(Auto$mpg > mpg_median, 1, 0))
data$mpg_lab <- as.factor(ifelse(Auto$mpg > mpg_median, "High Gas\nMileage", "Low Gas\nMil
```

```r
# Define a function to extract characters to the left of the first space
extract_left_of_space <- function(input_string) {
  split_string <- unlist(strsplit(as.character(input_string), " ", fixed = TRUE))
  return(split_string[1])
}

# Apply the function to the entire column using sapply
data$brand <-  sapply(data$name, extract_left_of_space)

# Rename 1,2,3 as American, Euro, Japanese
data$origin_lab <- as.factor(ifelse(data$origin == 1, "American",
                         ifelse(data$origin == 2, "European",
                                ifelse(data$origin == 3, "Japanese", NA))))

# Create a vector of replacements
replacements <- c("vw" = "volkswagen",
                  "vokswagen" = "volkswagen",
                  "chevroelt" = "chevrolet",
                  "toyouta" = 'toyota')

# Use the replacements vector to clean up the brand names
data$brand <- ifelse(data$brand %in% names(replacements), replacements[data$brand], data$b
```

4.2 Part c code

```r
################################################################################
# Part c
################################################################################

# Set a random seed for reproducibility
set.seed(123)

# Select the relevant variables
selected_vars <- c("cylinders", "displacement", "horsepower", "weight")

# Create a subset of the data with only the selected variables
```

```r
train_data <- data[, c("mpg01", selected_vars)]

# Split the data into a training set (70%) and a test set (30%)
sample_size <- floor(0.7 * nrow(train_data))

train_indices <- sample(seq_len(nrow(train_data)), size = sample_size)
train_set <- train_data[train_indices, ]
test_set <- train_data[-train_indices, ]
```

4.3 Part d code

```r
##############################################################################
# Part d
##############################################################################

# Perform Linear Discriminant Analysis (LDA)
lda_model <- lda(mpg01 ~ ., data = train_set)

# Predict mpg01 on the test set
lda_predictions <- predict(lda_model, newdata = test_set)

# Calculate the test error
test_error <- mean(lda_predictions$class != test_set$mpg01)

cat("Test Error:", test_error)
```

4.4 Part e code

```r
##############################################################################
# Part e
##############################################################################

# Perform Quadratic Discriminant Analysis (QDA)
qda_model <- qda(mpg01 ~ ., data = train_set)
```

```r
# Predict mpg01 on the test set
qda_predictions <- predict(qda_model, newdata = test_set)

# Calculate the test error
test_error <- mean(qda_predictions$class != test_set$mpg01)
cat("Test Error:", test_error)
```

4.5 Part f code

```r
##############################################################################
# Part f
##############################################################################

# Perform logistic regression
logistic_model <- glm(mpg01 ~ ., data = train_set, family = "binomial")

# Predict mpg01 on the test set
logistic_predictions <- predict(logistic_model, newdata = test_set, type = "response")

# Convert predicted probabilities to binary predictions (0 or 1)
logistic_predictions <- ifelse(logistic_predictions > 0.5, 1, 0)

# Calculate the test error
test_error <- mean(logistic_predictions != test_set$mpg01)
cat("Test Error:", test_error)
```

4.6 Part g code

```r
##############################################################################
# Part g
##############################################################################

# Perform Naive Bayes classification
naive_bayes_model <- naiveBayes(mpg01 ~ ., data = train_set)
```

```r
# Predict mpg01 on the test set
naive_bayes_predictions <- predict(naive_bayes_model, newdata = test_set)

# Calculate the test error
test_error <- mean(naive_bayes_predictions != test_set$mpg01)
cat("Test Error:", test_error)
```

4.7 Part h code

```r
##############################################################################
# Part h
##############################################################################

# Define a vector of K values to try
k_values <- c(1, 2, 3, 5, 6, 7, 8, 9, 10, 11, 12, 13, 14, 15)

# Initialize a vector to store test errors for each K
test_errors <- numeric(length(k_values))

# Perform KNN classification for each value of K
for (i in 1:length(k_values)) {
  k <- k_values[i]
  knn_predictions <- knn(train_set[, -1], test_set[, -1], train_set[, 1], k = k)

  # Calculate the test error for this K
  test_errors[i] <- mean(knn_predictions != test_set$mpg01)
}

# Print test errors for each K
for (i in 1:length(k_values)) {
  cat(paste("K =", k_values[i], "Test Error:", test_errors[i]), "\n")
}

k_values
test_errors
```