MATH 6350 – HOMEWORK #1

Brian Le, Basel Najjar, Soo Jong Cho

12 September, 2021

Table of Contents

I. Scope 2
II. Preliminary Statistical Analysis 2
1. Basic Descriptive Statistics 2
2. Graphical Description of Variables 3
3. Graphical Comparison of Variables 6
4. Feature Correlation Matrix 9
5. Response Variable Quantile Curve 10
6. Linear Regression Models 10
Summary of Regression Parameters 10
Graphical Representation of Regression Models 11
III. Automatic Classification of Data by kNN 15
8. Data Subsetting 15
9. Comparison of Low vs. High MPG Classes 15
10. Basic Descriptive Statistics for Low and High MPG Classes 18
11. Application of the k-Nearest Neighbor (kNN) Automatic Classifier 19
12. Determining the Best K-Value 21
13. Effect of Feature Normalization kNN Accuracy 22
IV. Appendix – R Code 23

I. Scope

This report analyzes a data set containing vehicle performance information. Exploratory analysis of this data set was completed using basic descriptive statistics and graphing tools. In addition, an automatic classification model was implemented using the k Nearest Neighbor (kNN) algorithm. The *R* statistical software package was utilized for this analysis. After removing entries containing missing data and eliminating categorical and irrelevant features, this data set contains 392 cases with 5 features and 1 response variable.

The response variable of this data set is:

- MPG - Miles per Gallon

The features of this data set are:

- Cylinders - Number of Cylinders (#)
- Displacement - Engine Displacement (in^3)
- Horsepower - Engine Horsepower (hp)
- Weight - Vehicle Weight (lbs)
- Acceleration - Time to Accelerate from 0 to 60 mph (sec.)

Several variables were eliminated from this analysis due to their irrelevance in description and/or prediction. These variables include:

- Year - Model Year (modulo 100)
- Origin - National Origin of Vehicle (1. American, 2. European, 3. Japanese)
- Name - Vehicle Make/Model

II. Preliminary Statistical Analysis

1. Basic Descriptive Statistics

Response Variable (Y) - Miles per Gallon

The mean of the response variable Y (vehicle fuel economy measured by miles per gallon) is 23.4 mpg. The standard deviation of Y is 7.8 mpg. The minimum value of response variable Y is 9 mpg and the maximum value is 46.6 mpg. The range is 37.6 mpg.

Feature Variable 1 (F1) - Number of Cylinders

The mean of feature variable F1 (number of cylinders) is 5.5 cylinders However, it should be noted that feature F1 is a discrete variable with whole number values (the number of cylinders in a vehicle engine must be a whole number). The median of F1 is 4 cylinders. The standard deviation of F1 is 1.7 cylinders. The minimum value of feature variable F1 is 3 cylinders and the maximum value is 8 cylinders. The range is 5 cylinders.

Feature Variable 2 (F2) - Engine Displacement (Cubic inches)

The mean of feature variable F2 (engine displacement measured in cubic inches) is 194.4 in^3. The standard deviation of F2 is 104.6 in^3. The minimum value of feature variable F2 is 68 in^3 and the maximum value is 455 in^3. The range is 387 in^3.

Feature Variable 3 (F3) - Engine Horsepower (hp)

The mean of feature variable F3 (engine power output measured in horsepower) is 104.5 hp. The standard deviation of F2 is 38.5 hp. The minimum value of feature variable F3 is 46 hp and the maximum value is 230 hp. The range is 184 hp.

Feature Variable 4 (F4) - Vehicle Weight (lb)

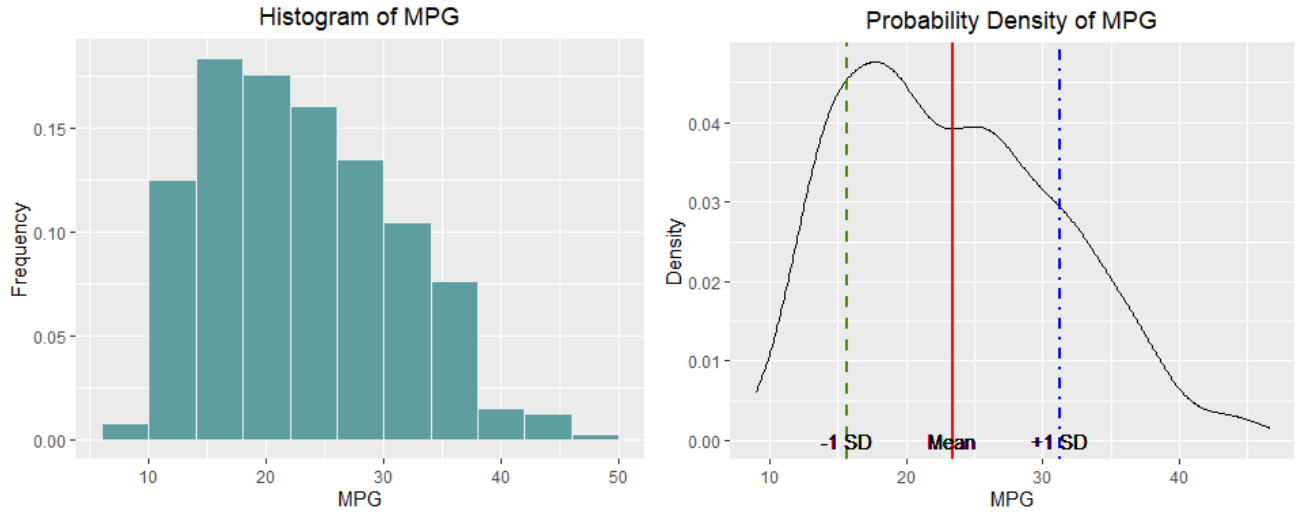
The mean of feature variable F4 (vehicle weight measured in pounds) is 2977.6 lb. The standard deviation of F2 is 849.4 lb. The minimum value of feature variable F4 is 1613 lb and the maximum value is 5140. The range is 3527 lb.

Feature Variable 5 (F5) - Vehicle Acceleration (seconds)

The mean of feature variable F5 (vehicle acceleration measured in time from 0 to 60 mph in seconds) is 15.5 s. The standard deviation of F2 is 2.8 s. The minimum value of feature variable F5 is 8 s and the maximum value is 24.8 s. The range is 16.8 s.

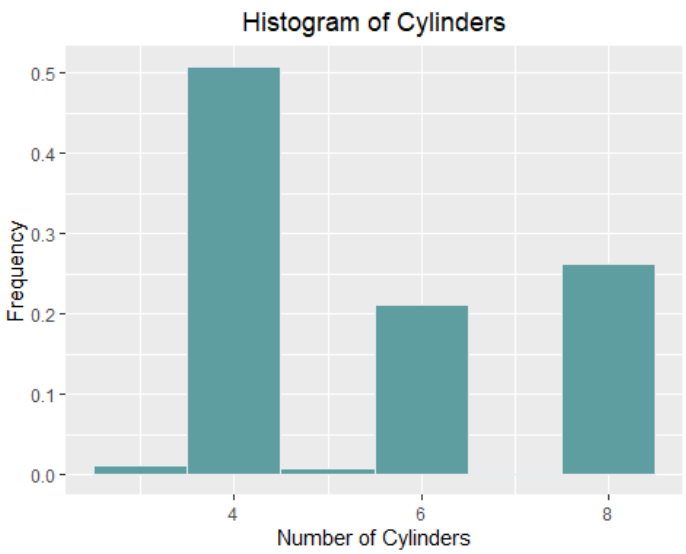
2. Graphical Description of Variables

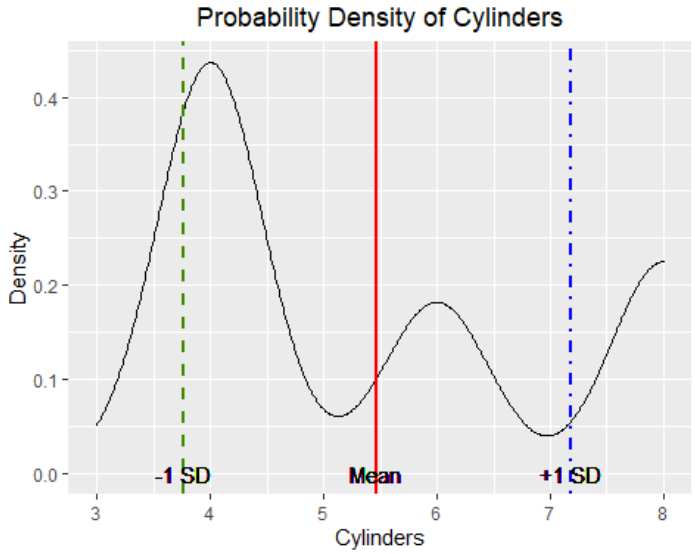
Response Variable (Y) - Miles per Gallon



MPG is a right-skewed distribution. This histogram shows that a majority of cases appear between 10 and 30 MPG. In fact, 76% of cases occur between 10 and 30 MPG. A similar conclusion can be drawn from the probability density plot of MPG. The PDF shows the same right skew with a relatively small standard deviation about the mean.

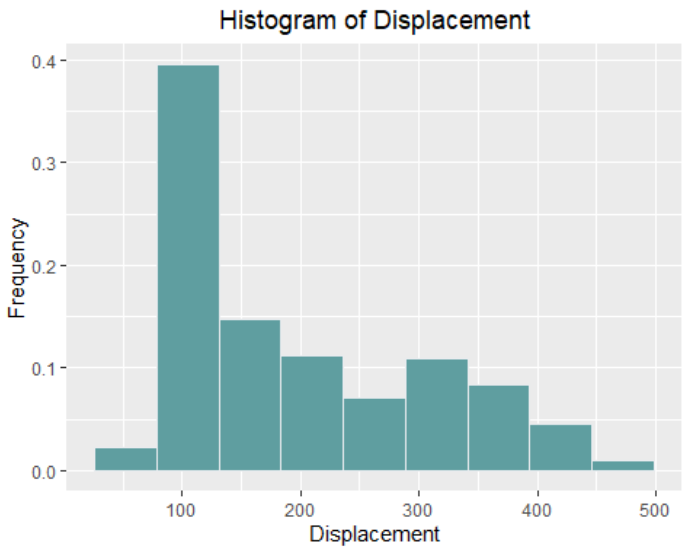
Feature Variable 1 (F1) - Number of Cylinders

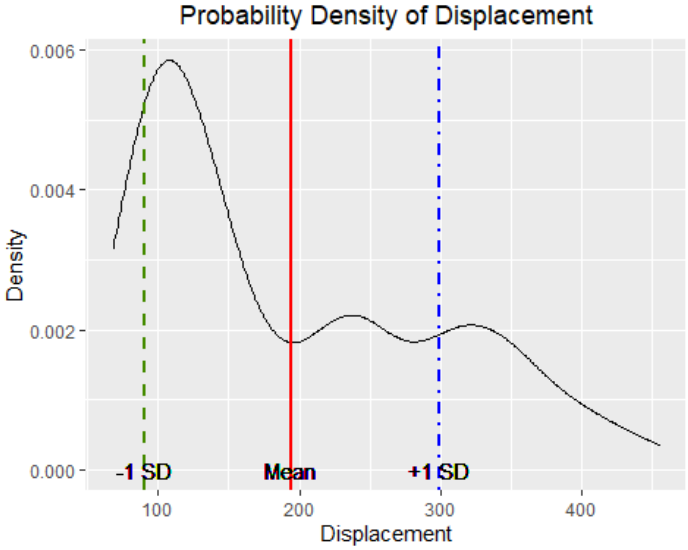




Based on the histogram above, the most common vehicle engine design in the data set is a 4-cylinder engine, followed by the 8 and 6-cylinder designs. The probability density plot shows the same bias towards 4-cylinder engines, and also shows the relatively large spread about the mean. The number of cylinders is a discrete variable, so the curve represented by the PDF is somewhat misleading as only whole integer numbers are possible for this variable.

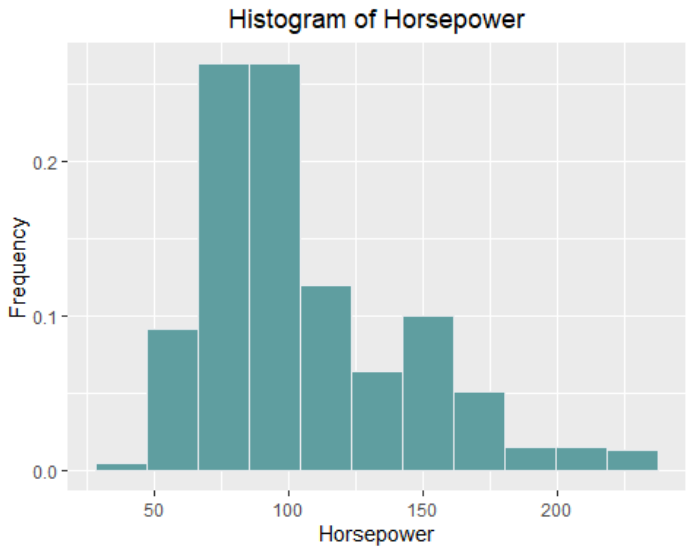
Feature Variable 2 (F2) - Engine Displacement (in^3)

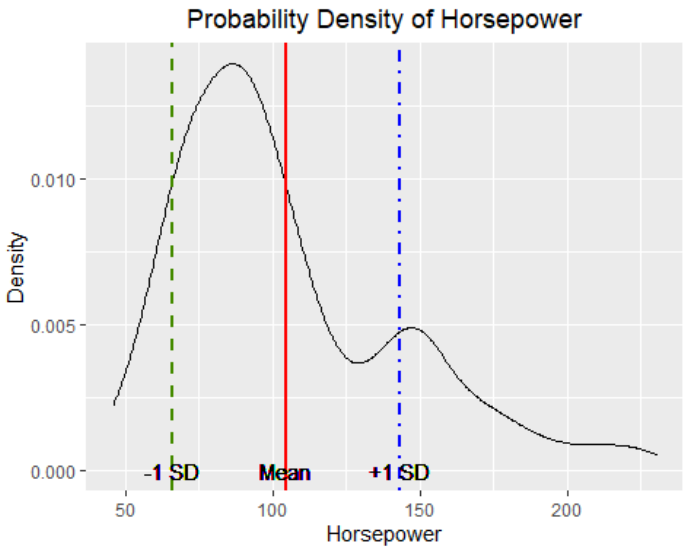




Displacement shows a strong right skew with a peak around 100 in^3. The PDF shows a similar peak slightly beyond 100 in^3. Very few vehicle engines displace more than 400 in^3 or less than 75 in^3.

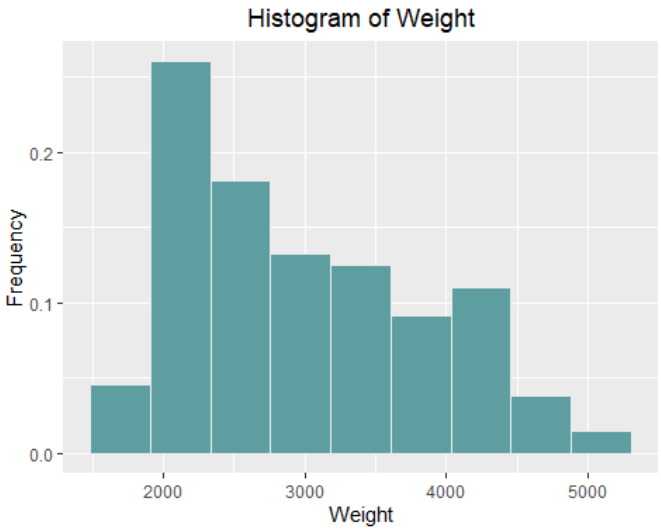
Feature Variable 3 (F3) - Engine Horsepower (hp)

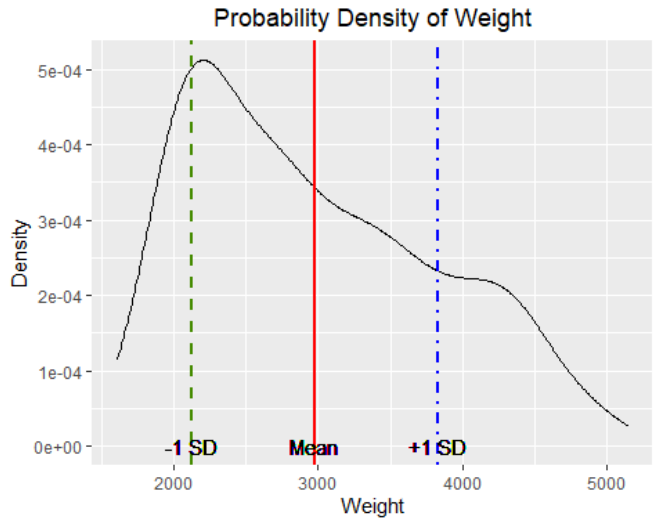




The distribution of horsepower is right-skewed. The histogram of horsepower shows that a majority of vehicles are rated at less than 125 hp. In fact, this represents approximately 74% of cases. The probability density plot of horsepower also displays a slight bi-modality with a second, smaller peak around 150 hp.

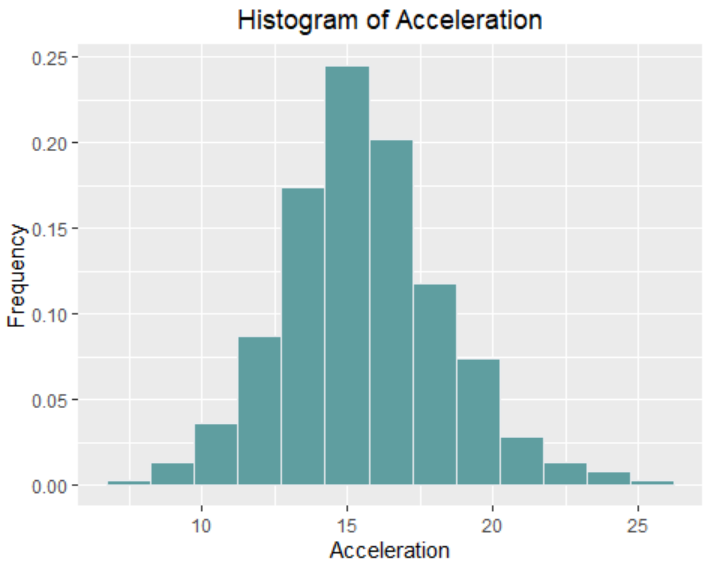
Feature Variable 4 (F4) - Vehicle Weight (lb)

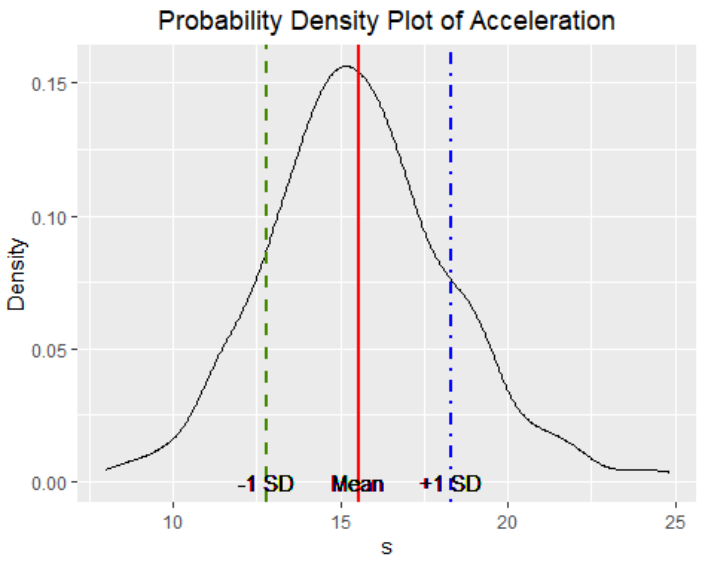




The distribution of weight is right-skewed. Both the histogram and PDF show a peak between 2,000 to 2,500 lb. Despite a concentration of vehicle weights in this range, there a relatively large standard deviation about the mean in this distribution. There are very few vehicles less than 2,000 lb or greater than 4,500 lb.

Feature Variable 5 (F5) - Vehicle Acceleration (seconds)

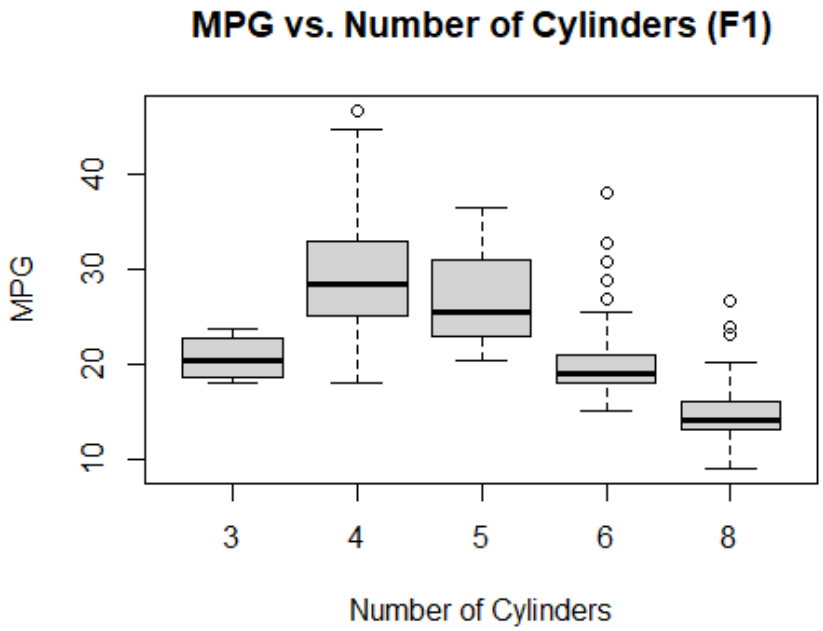




Acceleration appears to be normally distributed in this data set, with a bell-shaped curve about the mean. As shown in the probability density plot, the mean sits very close to peak frequency. The standard deviation is relatively small and the curve is “tall and thin”, as opposed to “short and wide”, Clearly, the majority of vehicles are rated between 12 to 18 s acceleration. This range represents 70% of cases.

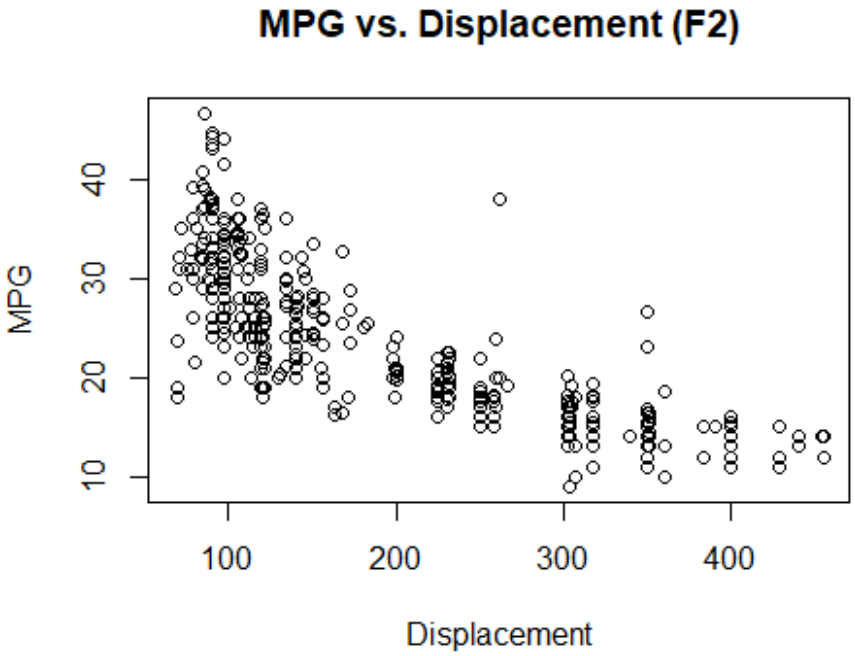
3. Graphical Comparison of Variables

The following figures provide a visual comparison of the response variable (Y) with each feature variable.

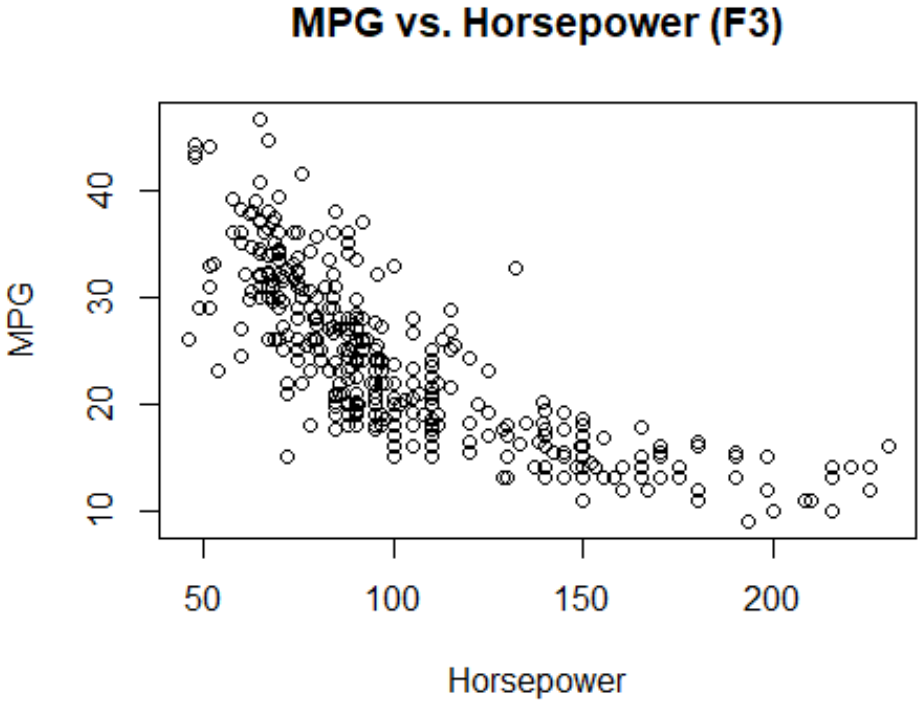


Based on the boxplot of MPG vs. Cylinders, we can conclude that 4-cylinder engines achieve better fuel economy (higher MPG) than any other engine design. The vehicle with the highest fuel economy in the data set has 4 cylinders. 3-cylinder engines achieve relatively poor fuel economy with few outliers. As the number of cylinders

increases beyond 4, vehicle fuel economy decreases in a somewhat linear fashion. 8-cylinder engines achieve the worst fuel economy of any engine design, and the vehicle with the lowest MPG rating in the data set has an 8-cylinder engine.

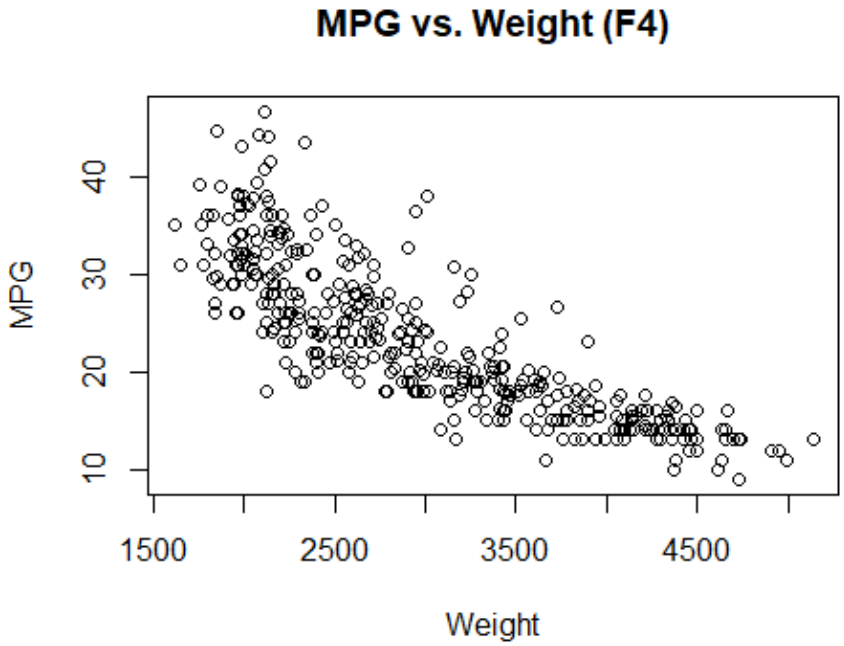


The scatter plot displays a negative correlation between displacement and MPG. The relationship is non-linear at low displacement (high MPG) values with small changes in displacement causing significant changes in MPG. The relationship becomes increasingly linear for moderate displacement (moderate MPG) cases. For high displacement (low MPG) cases, there appears to be minimal effect of displacement on MPG.

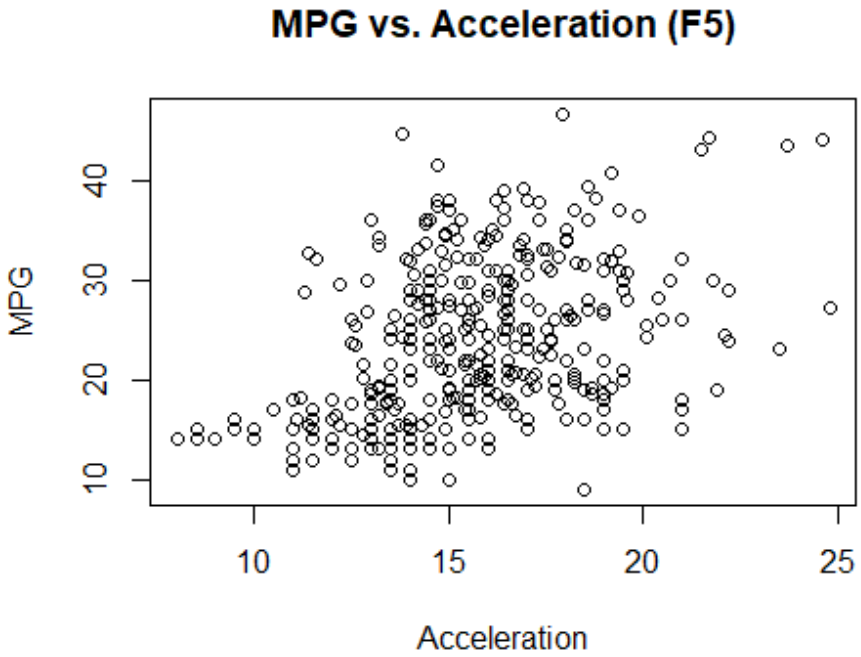


The scatter plot displays a negative correlation between horsepower and MPG. The relationship is non-linear at low horsepower (high MPG) values with small changes in horsepower causing significant changes in MPG. The

relationship becomes increasingly linear for moderate horsepower (moderate MPG) cases. For high horsepower (low MPG) cases, there appears to be minimal effect of weight on MPG.



The scatter plot displays a negative correlation between weight and MPG. The relationship appears to be non-linear at low horsepower (high MPG) values with small changes in horsepower causing significant changes in MPG. The relationship becomes increasingly linear for moderate horsepower (moderate MPG) cases. For high horsepower (low MPG) cases, there appears to be minimal effect of horsepower on MPG.



The scatter plot displays a positive correlation between acceleration and MPG. The relationship appears to be linear for cases with low acceleration (low MPG). The relationship becomes less linear and less predictable for moderate and high acceleration cases. Acceleration does not appear to be an effective sole discriminator for MPG, as there is a wide variety of MPG ratings for vehicles with similar acceleration.

The correlation coefficient for each of the five relationships (Y vs. F1, …, Y vs. F5) is provided in the table below. The correlation coefficient describes the "strength" and "direction" of the linear relationship, with -1 describing a perfectly negative linear relationship, 0 describing no linear relationship, and +1 describing a perfectly positive linear relationship. It should be noted that variables whose coefficients are close to 0 may still exhibit a non-linear relationship. The formula for correlation coefficient between two variables X,Y is also described below.

	Correlation Coefficient Value
MPG vs. Cylinders	-0.78
MPG vs. Displacement	-0.81
MPG vs. Horsepower	-0.78
MPG vs. Weight	-0.83
MPG vs. Acceleration	0.42

$$cor(X,Y) = \frac{\sum(x_i - \bar{x})(y_i - \bar{y})}{\sqrt{\sum(x_i - \bar{x})^2 \sum(y_i - \bar{y})^2}}$$

Where:

- x_i represents the ith entry of feature variable X.
- $\bar{x}$ represents the mean of the values of the feature variable
- Similar notation is used for the response variable Y.

4. Feature Correlation Matrix

The following table represents the 5x5 feature correlation matrix. This outlines correlation coefficient values for all features compared to one another.

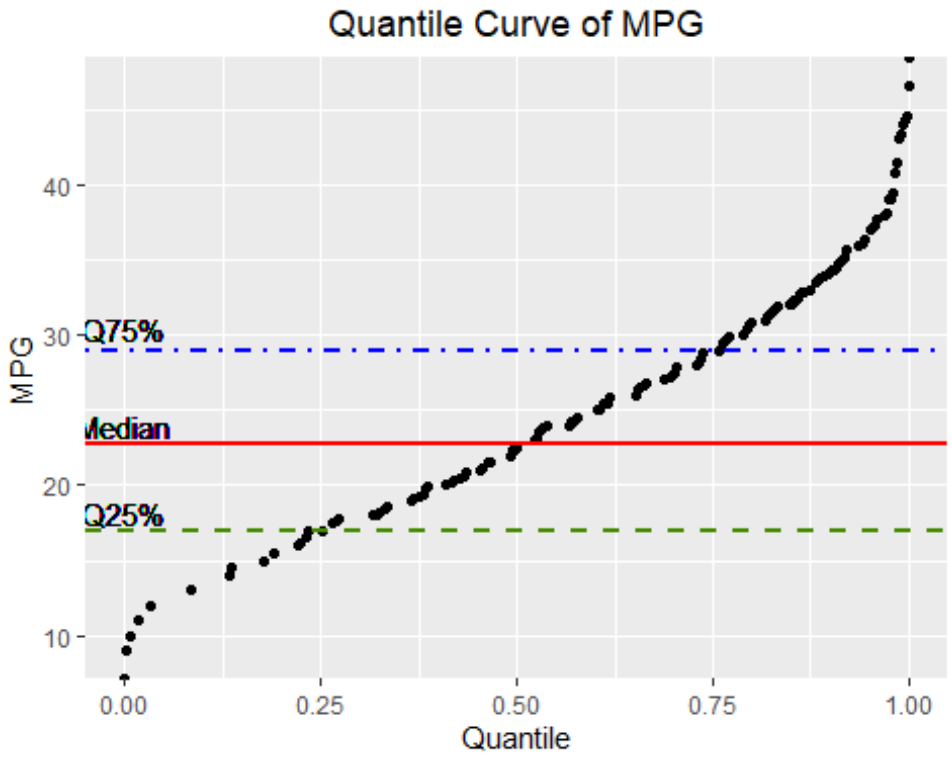
	Cylinders	Displacement	Horsepower	Weight	Acceleration
Cylinders	1.00	0.95	0.84	0.90	-0.50
Displacement	0.95	1.00	0.90	0.93	-0.54
Horsepower	0.84	0.90	1.00	0.86	-0.69
Weight	0.90	0.93	0.86	1.00	-0.42
Acceleration	-0.50	-0.54	-0.69	-0.42	1.00

Based on the 5x5 feature correlation matrix, all features except for acceleration exhibit strong positive correlation. Acceleration shows a weaker negative relationship to all other features. The weakest correlation is between weight and acceleration (-0.42). The strongest relationship is between cylinders and displacement (+0.95), and is nearly perfectly linear. Displacement measures the total volume that is displaced by the piston within the cylinders of the engine. Therefore, this relationship is not surprising. The weakest correlation among features (excluding

acceleration) is between horsepower and cylinders (0.84). In the context of automatic classification, a strong relationship (either positive or negative) between feature variables typically indicates that these features are interchangeable or redundant. Several methods exist in literature and in practice for eliminating unnecessary features. One commonly used method is principle component analysis (PCA). Discussion and implementation of PCA or other dimensionality reduction methods is beyond the scope of this report.

5. Response Variable Quantile Curve

The quantile curve of the response variable shows the 25%, median (50%), and 75% quantiles for MPG.



6. Linear Regression Models

Summary of Regression Parameters

The table below outlines key linear regression parameters for each feature variable (F1, …, F5) vs. the response variable (Y).

- Slope represents the rate of change in the value of MPG for a unit change in the value of the feature.
- Intercept represents the MPG value when the feature variable value set to zero; the interpretation of the intercept is meaningless in many real-world examples (including this one) and serves to adjust the “starting point” for the slope in the regression model.
- RMSE is the root mean squared error and represents the square root of the sum of distances between each actual value and the regression line divided by degrees of freedom (in this case n-2, where n represents the

number of cases). This is essentially the average model prediction error. The RMSE formula is described below.

- Relative accuracy represents the RMSE divided by the mean of response variable Y. This re-scales RMSE to the scale of Y.

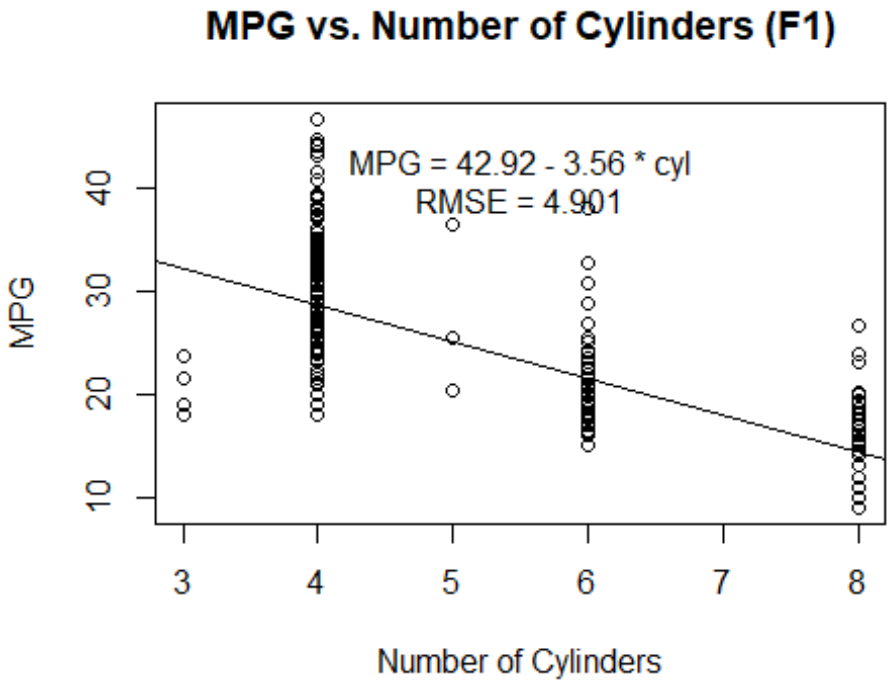
	Slope	Intercept	RMSE	Relative Accuracy
F1	-3.56	42.92	4.90	0.21
F2	-0.06	35.12	4.62	0.20
F3	-0.16	39.94	4.89	0.21
F4	-0.01	46.22	4.32	0.18
F5	1.20	4.83	7.06	0.30

$$RMSE = \sqrt{\frac{\sum(yi - \hat{y})^2}{n}}$$

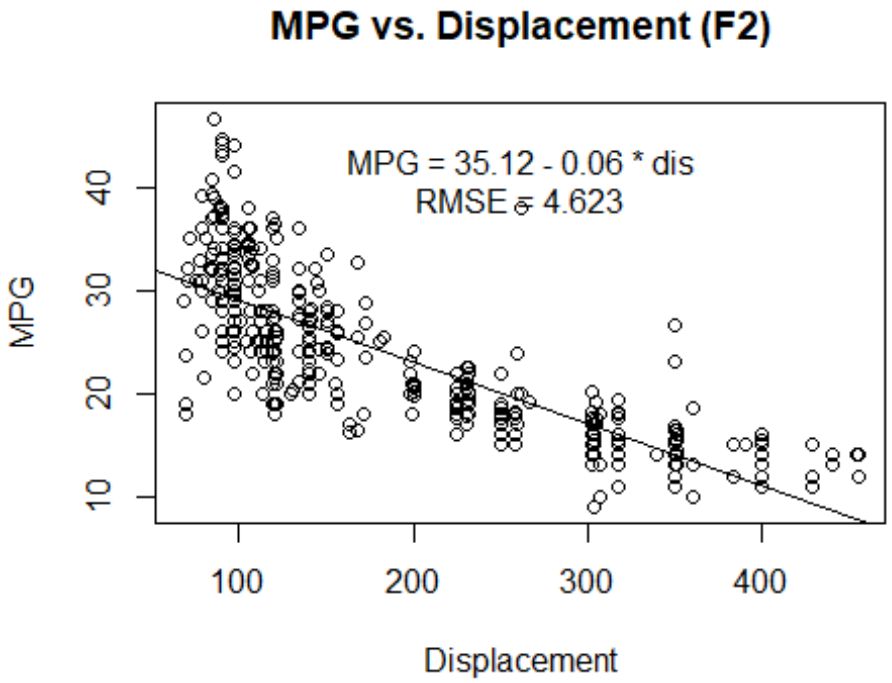
Where:

- yi represents the actual value of response variable y for case i.
- $\hat{y}$ represents the predicted value of response variable y for case i, using the linear model.
- *n* represents the total number of cases (i = 1, …, n)

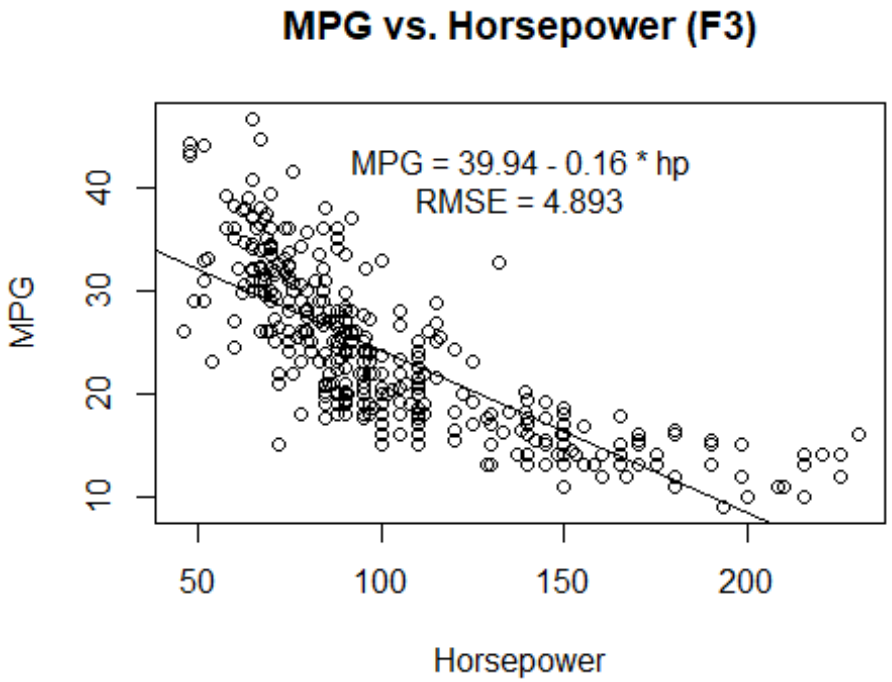
Graphical Representation of Regression Models



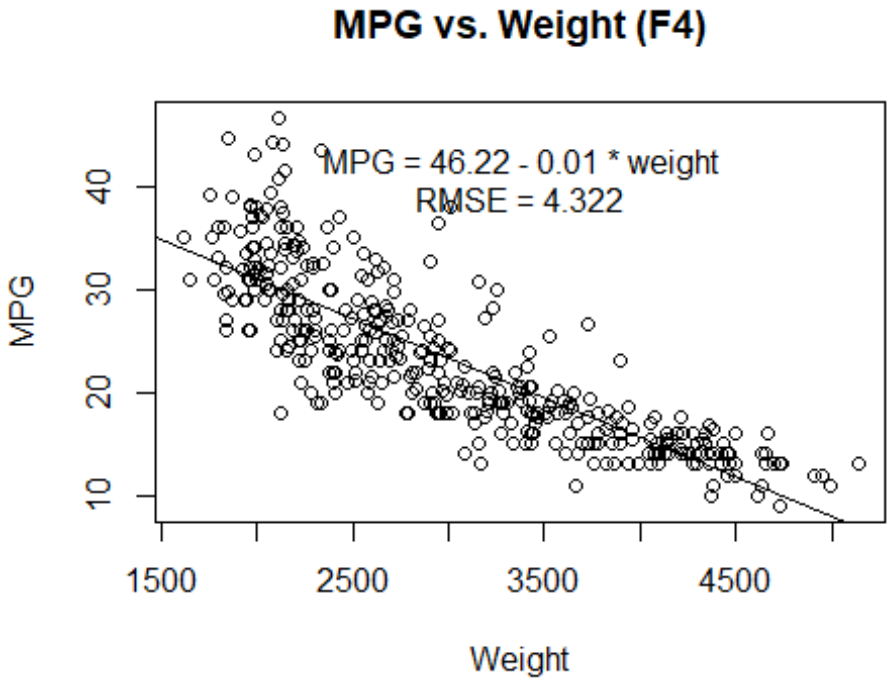
For every unit increase in the number of cylinders, there is a change of -3.56 in the MPG rating of the vehicle. The linear relationship is negative; as the number of cylinders increases, MPG rating decreases. The regression line appears to explain the central tendency of MPG for a given number of cylinders relatively well, with the exception of 3-cylinder vehicles. However, this single-feature linear model does not capture the variety of MPG values observed for a given number of cylinders. The RMSE is 4.901 MPG for this linear model.



For every unit increase in displacement, there is a change of -0.06 in the MPG rating of the vehicle. As described in previous sections, the linear relationship is negative. As the displacement increases, MPG rating decreases. The linear model appears to have reasonably good prediction power for moderate displacement, but fails to capture the non-linearity in the low displacement regime and under-predicts MPG for high displacement vehicles. The RMSE is 4.623 MPG for this linear model.

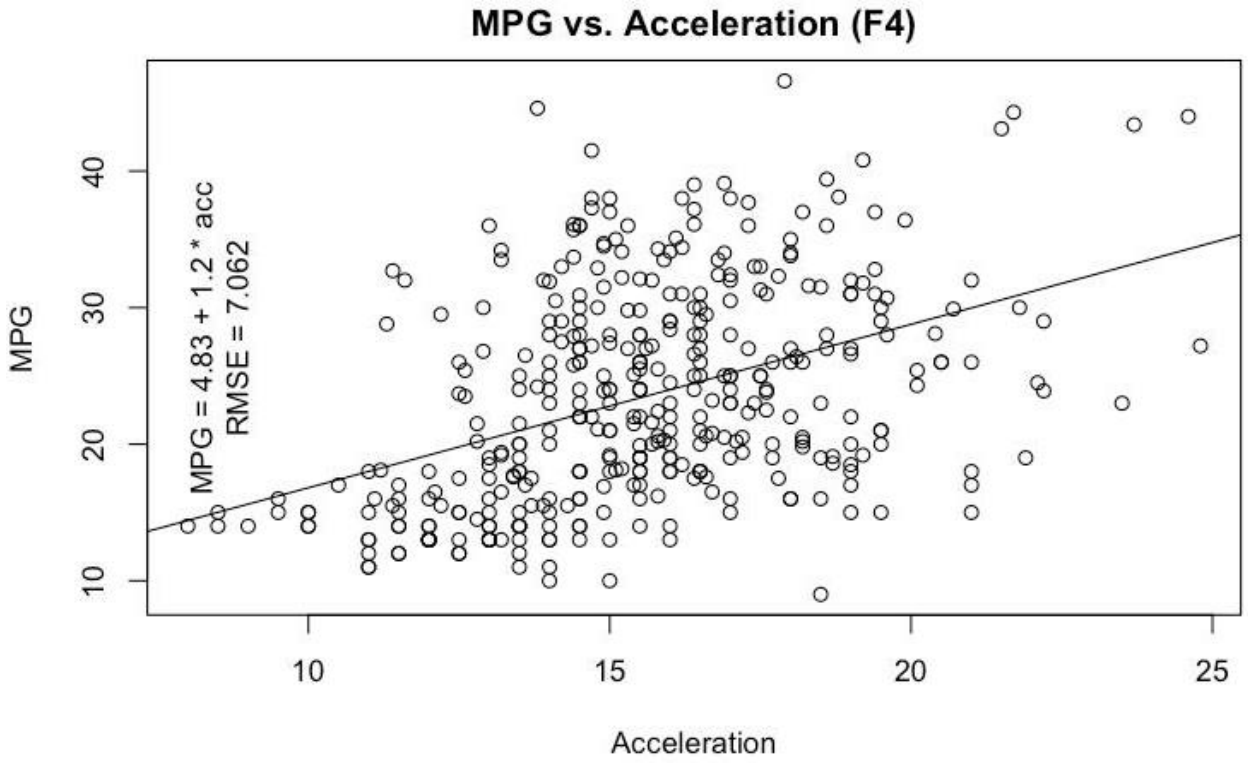


For every unit increase in horsepower, there is a change of -0.16 in the MPG rating of the vehicle. As described in previous sections, the linear relationship is negative. As the horsepower increases, MPG rating decreases. The linear model appears to have reasonably good prediction power for moderate horsepower, but fails to capture the non-linearity in the low horsepower regime and under-predicts MPG for high horsepower vehicles. The RMSE is 4.893 MPG for this linear model.



For every unit increase in weight, there is a change of -0.01 in the MPG rating of the vehicle. As described in previous sections, the linear relationship is negative. As the weight increases, MPG rating decreases. The linear

model appears to have reasonably good prediction power for moderate weight, but fails to capture the non-linearity in the low weight regime and under-predicts MPG for heavy vehicles. The RMSE is 4.893 MPG for this linear model.



For every unit increase in acceleration, there is a change of +1.2 in the MPG rating of the vehicle. As described in previous sections, the linear relationship is negative. As the acceleration increases, MPG rating decreases. The linear model appears to have reasonably good prediction power for low acceleration vehicles (<10 s). However, this single-feature linear model does not capture the variety of MPG values observed at higher acceleration ratings. The RMSE is 4.893 MPG for this linear model, the highest RMSE of the five single-feature linear models described in this section.

III. Automatic Classification of Data by kNN

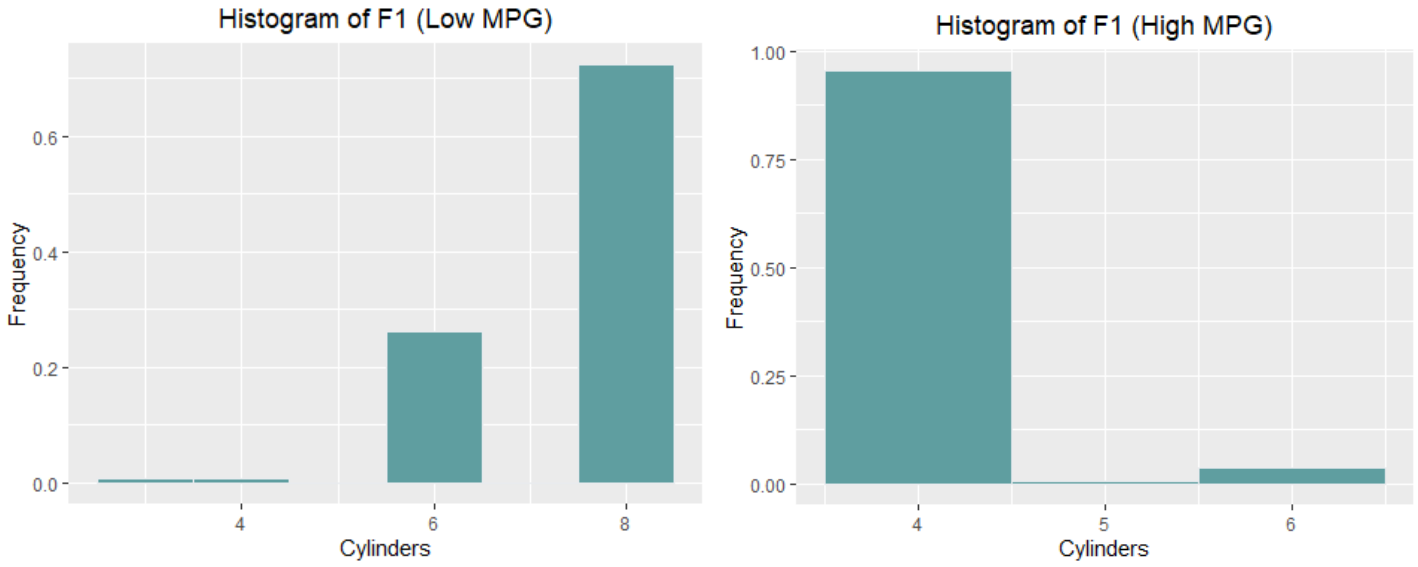
8. Data Subsetting

Three subsets of data were extracted from the cleaned 'auto' data frame. These three subsets are:

- Low MPG: 9 - 18.5 MPG. This subset contains 130 entries.
- Med MPG: 18.6 - 26.6 MPG. This subset contains 130 entries.
- High MPG: 26.8 - 46.6 MPG. This subset contains 132 entries.

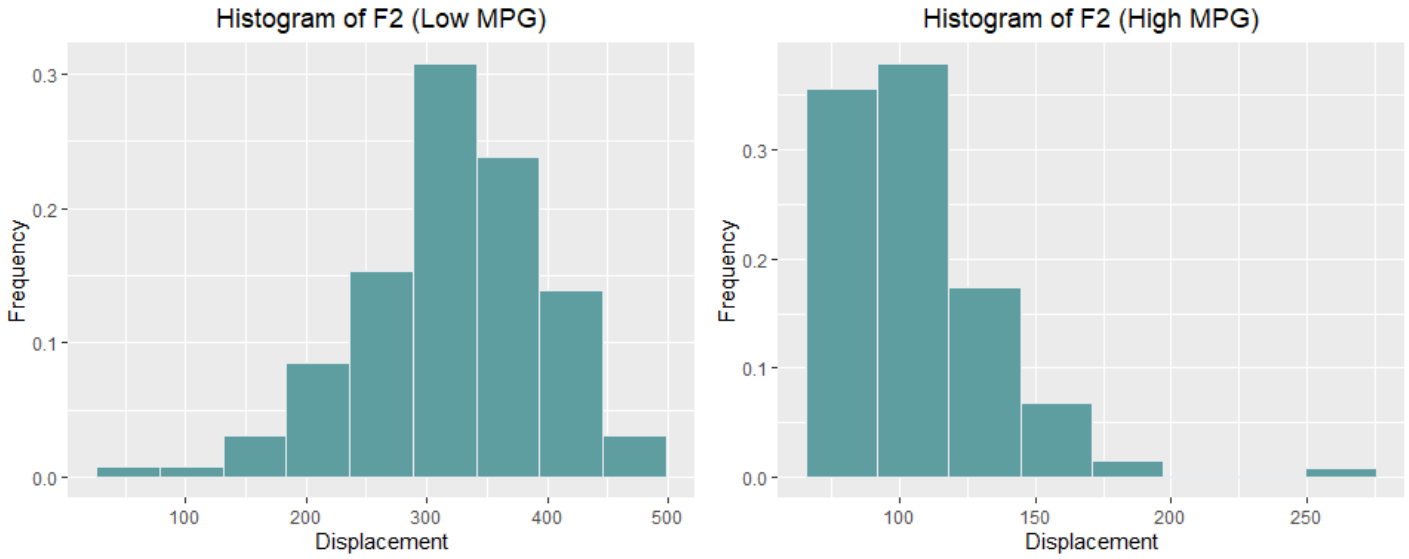
9. Comparison of Low vs. High MPG Classes

Feature Variable 1 (F1) - Number of Cylinders



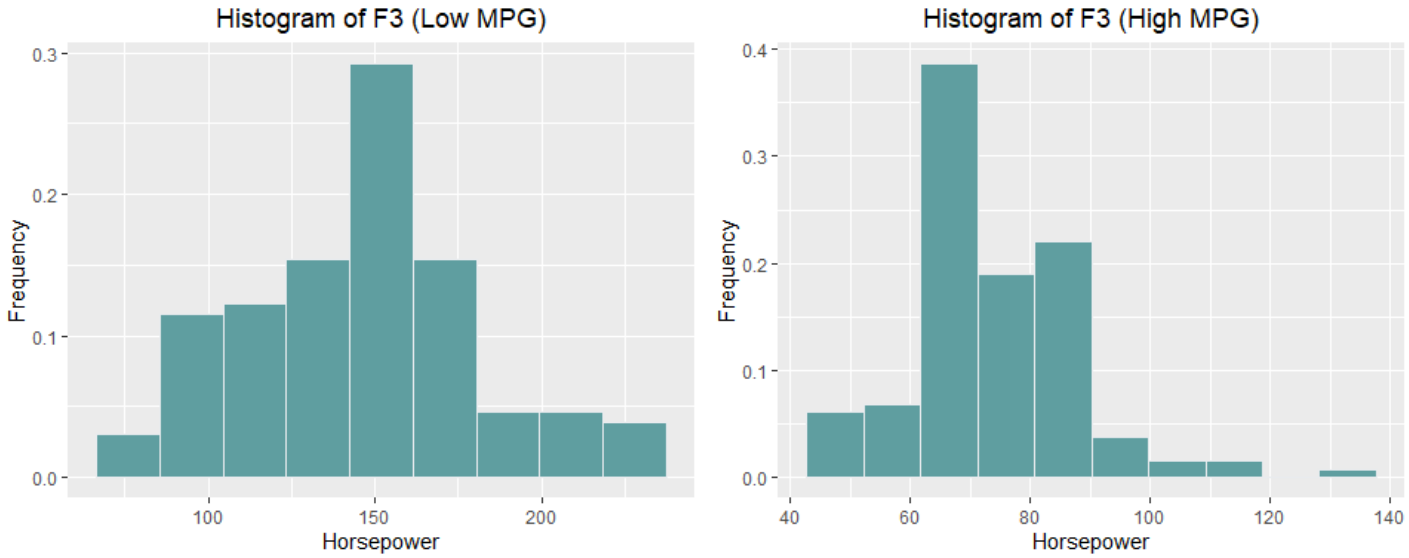
Based on the comparison of the above histograms, there is minimal intersection between the two subsets. Low MPG vehicles typically have more cylinders, with only a small fraction of low MPG vehicles having 4 cylinders or fewer. Similarly, high MPG vehicles typically have fewer cylinders; nearly all the vehicles in the high MPG subset have 4 cylinders. Several high MPG vehicles have 6 cylinders: this represents the primary intersection between low and high MPG subsets with respect to number of cylinders.

Feature Variable 2 (F2) - Engine Displacement (Cubic inches)



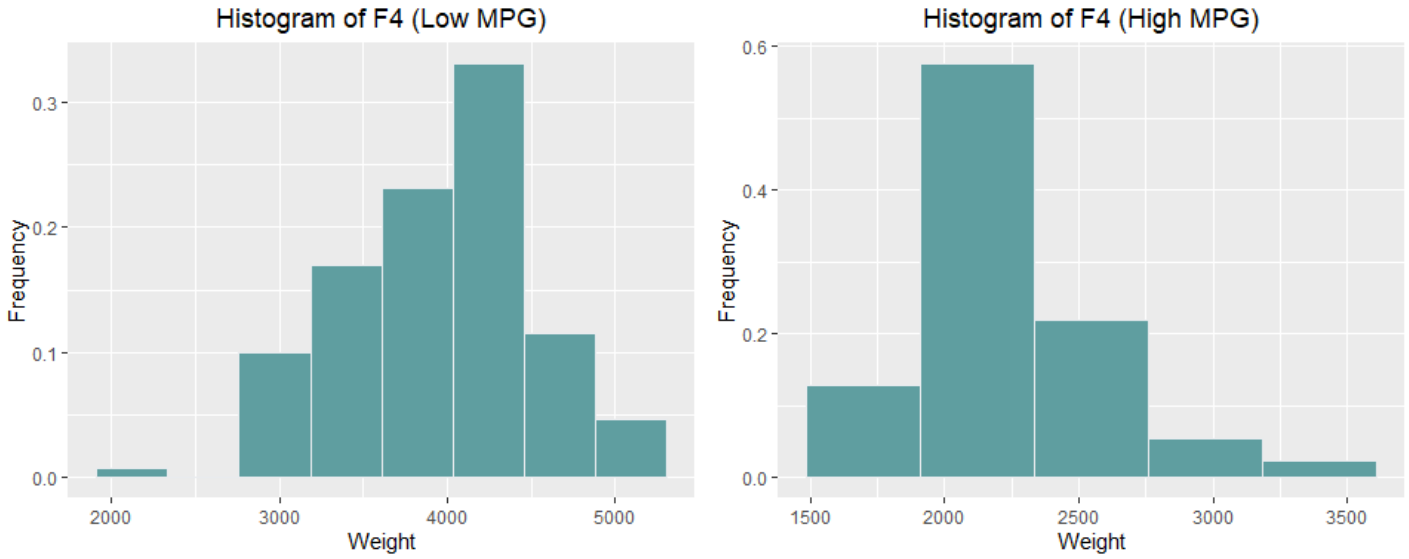
There is some intersection between the low MPG and high MPG subsets with respect to displacement. High MPG vehicles typically have smaller displacement; nearly all the vehicles in the high MPG subset have displacement less than 200 in^3. Low MPG vehicles, on the other hand, typically have a larger displacement. A relatively small (but non-negligible) fraction of low MPG vehicles have displacement values less than 200 in^3. This represents the primary intersection between low and high MPG subsets with respect to displacement.

Feature Variable 3 (F3) - Engine Horsepower (hp)



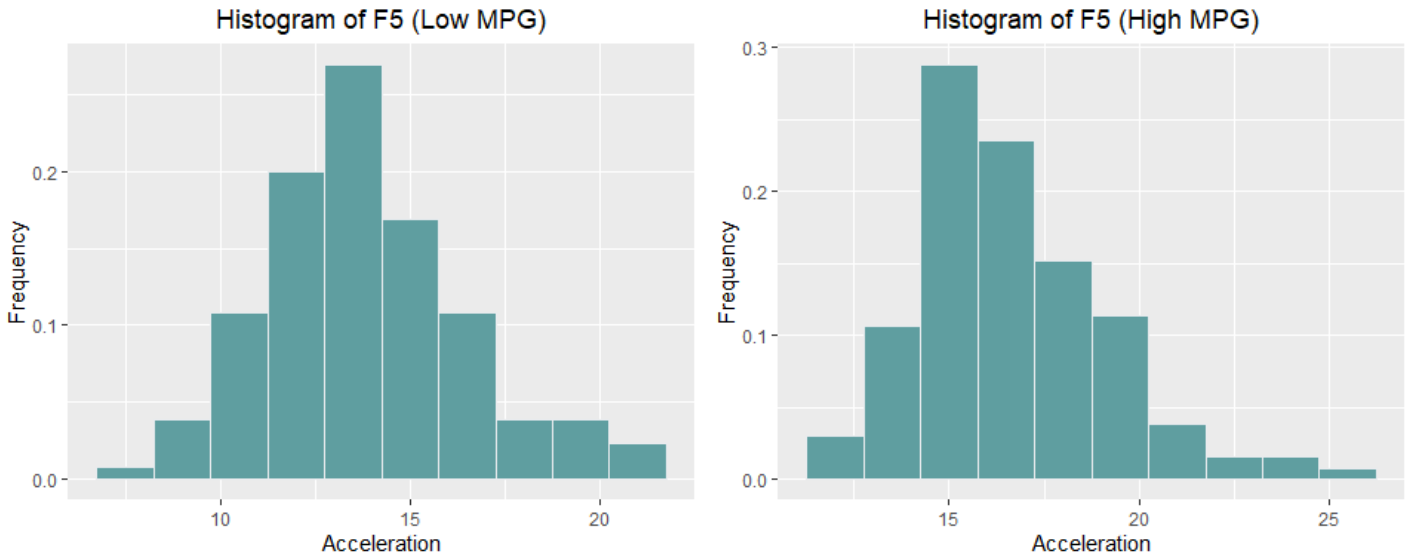
There is some intersection between the low MPG and high MPG subsets with respect to horsepower High MPG vehicles typically have lower horsepower ratings; nearly all the vehicles in the high MPG subset have less than 100 hp. Low MPG vehicles, on the other hand, typically have a higher horsepower ratings. A relatively small (but non-negligible) fraction of low MPG vehicles have less than 100 hp. This represents the primary intersection between low and high MPG subsets with respect to horsepower.

Feature Variable 4 (F4) - Vehicle Weight (lb)



There is some intersection between the low MPG and high MPG subsets with respect to vehicle weight. Low MPG vehicles are typically heavier, with nearly all low MPG vehicles weighing 3000 lb or more. High MPG vehicles, on the other hand, are typically lighter. Nearly all the vehicles in the high MPG subset weigh less than 2500 lb, though there are several heavy (3000+ lb) high MPG vehicles. This represents the primary intersection between low and high MPG subsets with respect to horsepower.

Feature Variable 5 (F5) - Vehicle Acceleration (seconds)



There is significant intersection between the low MPG and high MPG subsets with respect to vehicle acceleration. Low MPG vehicles tend to be faster (higher acceleration) and high MPG vehicles tend to be slower (lower acceleration). However, the intersection between the two subsets suggests that there are many slow low MPG vehicles and many fast high MPG vehicles. This is detrimental in the context of automatic classification, as ideally class subsets should be disjoint or minimally intersecting with respect to each feature.

10. Basic Descriptive Statistics for Low and High MPG Classes

Feature Variable 1 (F1) - Number of Cylinders

The mean of feature variable F1 (number of cylinders) is 7.4 for the low MPG class, and 4.1 for the high MPG class. As previously stated, F1 is a discrete numeric variable with positive integer values (the number of cylinders in a vehicle engine is a whole number). The median of F1 is 8 for the low MPG class, and 4 for the high MPG class. The standard deviation of F1 is 1.01 for the low MPG class, and 0.39 for the high MPG class.

At the 90% confidence level, the margin of error is 0.142 for the low MPG class, and 0.055 for the high MPG class. This essentially means that with 90% confidence, the number of cylinders for vehicles in the low MPG class will be between 7.27 and 7.55. Similarly, the number of cylinders for vehicles in the high MPG class will be between 4.03 and 4.14. The 90% confidence intervals for the low MPG and high MPG classes are disjoint. In fact, the difference ("the gap") between the upper bound of the high MPG class and the lower bound of the low MPG class is 3.13. This indicates good discriminating power when used as a classification feature.

Feature Variable 2 (F2) - Engine Displacement (Cubic inches)

The mean of feature variable F2 (engine displacement) is 315.3 for the low MPG class, and 106.4 for the high MPG class. The standard deviation of F2 is 71.11 for the low MPG class, and 26.42 for the high MPG class.

At the 90% confidence level, the margin of error is 9.979 for the low MPG class, and 3.68 for the high MPG class. This essentially means that with 90% confidence, the engine displacement for vehicles in the low MPG class will be between 305.33 and 325.29. Similarly, the engine displacement for vehicles in the high MPG class will be between 102.72 and 110.08. The 90% confidence intervals for the low MPG and high MPG classes are disjoint. The difference between the upper bound of the high MPG class and the lower bound of the low MPG class is 195.25. This indicates good discriminating power when used as a classification feature.

Feature Variable 3 (F3) - Engine Horsepower (hp)

The mean of feature variable F3 (engine horsepower) is 145.6 for the low MPG class, and 74.4 for the high MPG class. The standard deviation of F3 is 35.84 for the low MPG class, and 13.88 for the high MPG class.

At the 90% confidence level, the margin of error is 5.03 for the low MPG class, and 1.934 for the high MPG class. This essentially means that with 90% confidence, the engine horsepower for vehicles in the low MPG class will be between 140.59 and 150.65. Similarly, the engine horsepower for vehicles in the high MPG class will be between 72.46 and 76.33. The 90% confidence intervals for the low MPG and high MPG classes are disjoint. The difference between the upper bound of the high MPG class and the lower bound of the low MPG class is 64.26. This indicates good discriminating power when used as a classification feature.

Feature Variable 4 (F4) - Vehicle Weight (lb)

The mean of feature variable F4 (vehicle weight) is 3937.3 for the low MPG class, and 2226.1 for the high MPG class. The standard deviation of F4 is 557.19 for the low MPG class, and 345.88 for the high MPG class.

At the 90% confidence level, the margin of error is 78.189 for the low MPG class, and 48.168 for the high MPG class. This essentially means that with 90% confidence, the vehicle weight for vehicles in the low MPG class will be between 3859.14 and 4015.52. Similarly, the vehicle weight for vehicles in the high MPG class will be between 2177.92 and 2274.26. The 90% confidence intervals for the low MPG and high MPG classes are disjoint. The difference between the upper bound of the high MPG class and the lower bound of the low MPG class is 1584.88. This indicates good discriminating power when used as a classification feature.

Feature Variable 5 (F5) - Vehicle Acceleration (seconds)

The mean of feature variable F5 (vehicle acceleration) is 13.8 for the low MPG class, and 16.6 for the high MPG class. The standard deviation of F5 is 2.65 for the low MPG class, and 2.52 for the high MPG class.

At the 90% confidence level, the margin of error is 0.372 for the low MPG class, and 0.351 for the high MPG class. This essentially means that with 90% confidence, the vehicle acceleration for vehicles in the low MPG class will be between 13.41 and 14.15. Similarly, the vehicle acceleration for vehicles in the high MPG class will be between 16.21 and 16.91. The 90% confidence intervals for the low MPG and high MPG classes are disjoint. The difference between the upper bound of the high MPG class and the lower bound of the low MPG class is 3.5. Despite the significant intersection observed in the histogram of acceleration for the two subsets, the 90% confidence interval suggests good discriminating power when acceleration is used as a classification feature.

11. Application of the k-Nearest Neighbor (kNN) Automatic Classifier

In this section, the k-Nearest Neighbor (kNN) automatic classification algorithm will be applied to the auto data set. The kNN algorithm is a supervised automatic classification algorithm that produces a categorization model based on input training data. The algorithm uses the distance between points to classify or categorize points into pre-defined classes or groups. With kNN, the MPG value of vehicles in the auto data set is classified as Low, Medium, or High MPG using the feature variables. A range of k-values was tested to determine the k-value that offers the best performance. Performance is measured using percent accuracy and is described in detail in a later section of this report. In general, lower values of k (<10) produced the best results. The algorithm was also tested with and without the use of standardization (normalization) of feature variables. The performance of kNN using normalized and un-normalized data is discussed in a later section of this report. In selecting k, it is worth noting that high k-values tend to result in model “overfit”. This means that the model performance is good when applied to the training set, but poor when applied to the test set or a new un-tested data set. Higher k-values force the model to classify based on a pool of training data that is too large.

As with the use of any model, there are benefits and drawbacks. The kNN algorithm is fast, simple, and can produce a relatively high percent of correct classifications without the need for tedious model training and parameter-fitting. The algorithm and close variations of the it can be used for both classification and regression tasks, and its applications are broad. One drawback of the kNN algorithm is the complexity introduced by many feature variables. For this data set with only five features, however, kNN is adequate as an automatic classification algorithm.

In previous sections, data sub-setting based on MPG quantiles was described in detail. Here, the data is further modified before implementation into the algorithm. First, every case was labeled with its true classification (Low, Med, High) by appending a label column to each subset. Next, the subsets were randomly split into training and test subsets; the 80%/20% heuristic for training/test set size was used. Finally, the six subsets (three training, three test) were re-grouped into a global training set and a global test set. Following this random partitioning and recombination, set size and composition is outlined below:

- Training Set Size: 313 cases - 79.85% of total
    - Low MPG: 104 cases
    - Med MPG: 104 cases
    - High MPG: 105 cases
- Test Set Size: 79 cases - 20.15% of total
    - Low MPG: 26 cases
    - Med MPG: 26 cases
    - High MPG: 27 cases

Using a k-value equal to 5 (classification based on 5 nearest neighbors), the training set accuracy was 85%. As is typical with most classification models, the test set accuracy was slightly lower, at 72%. Accuracy is calculated by totaling the number of cases that were correctly classified and dividing by the total number of cases.

$$\%\ Accuracy = \frac{\#\ of\ Cases\ Classified\ Correctly}{Total\ \#\ of\ Cases}$$

The confusion matrix for k-value equal to 5 is described below. The classes on the top horizontal are the predicted classes, the classes on the left vertical are actual classes. Diagonal entries are correctly classified cases.

kNN Applied to Training Set (k = 5)			
N = 313	**High**	**Low**	**Medium**
High	86%	0%	13%
Low	0%	89%	7%
Medium	14%	11%	80%

kNN Applied to Test Set (k = 5)			
N = 79	**High**	**Low**	**Medium**
High	74%	0%	23%
Low	0%	96%	31%
Medium	26%	4%	46%

Example (training set): 89% of low MPG cases were correctly classified as low MPG. 7% of low MPG cases were incorrectly classified as medium MPG.

12. Determining the Best K-Value

Multiple k-values were tested iteratively, and the percent accuracy was calculated for each iteration. Significant differences in model accuracy are observed between the training and test data sets. The best performance overall was given by k = 9, with 73% test set accuracy and 76% training set accuracy. However, the best performance for the training set was given by k = 3 with 85% accuracy; this value of k gave the worst performance for the test set with 63% accuracy. Therefore, using the best k value from the training data will not always give the best performance in the test set.

13. Effect of Feature Normalization kNN Accuracy

When an automatic classification algorithm is implemented, features are typically normalized (centered and re-scaled). This allows the kNN algorithm to compute a more meaningful distance between neighboring points and should result in better classification performance. The normalization formula is described below. It should be noted that only feature variables (F1, …, F5) were normalized, not the response variable (Y).

$$V_i = \frac{U_i - \overline{U}}{stdev(U)}$$

Where:

- V_i represents the normalized entry i (i = 1 to 392) for a given feature variable
- U_i represents raw entry i of a given feature variable
- $\overline{U}$ represents the mean of a given feature variable
- $stdev(U)$ represents the standard deviation of feature variable U

Using a k-value equal to 5 (classification based on 5 nearest neighbors), normalized training set accuracy was 82.1%. As expected, the test set accuracy was slightly lower, at 79.7%. Compare this to the raw data set with 82.7% training set accuracy and 75.9% test set accuracy. Normalization of features had minimal impact on training set accuracy, but a 3.8% difference can be observed for test set accuracy. The confusion matrix for k-value equal to 5 is described below. The classes on the top horizontal are the predicted classes, the classes on the left vertical are actual classes. Diagonal entries are correctly classified cases.

kNN Applied to Feature-Normalized Training Set (k = 5)			
N = 313	**High**	**Low**	**Medium**
High	87%	1%	20%
Low	1%	89%	10%
Medium	12%	10%	70%

kNN Applied to Feature-Normalized Test Set (k = 5)			
N = 79	**High**	**Low**	**Medium**
High	74%	0%	15%
Low	0%	81%	12%
Medium	26%	19%	73%

IV. Appendix – R Code

```
knitr::opts_chunk$set(echo = TRUE)
#install.packages("ggplot2")
#install.packages("lattice")
#install.packages("knitr")
#install.packages("rsample")
#install.packages("caret")
#install.packages("patchwork")
library(ggplot2)
library(lattice)
library(knitr)
library(rsample)
library(caret)
library(class)
library(patchwork)

#Importing cleaned CSV file as a data frame called "auto"
auto = read.csv("DataAuto.csv")
auto_features = auto[2:6]

Y = auto$mpg #declare response variable MPG as "Y"
# Declaration of all feature variables Fi (i=1 to 5)
F1 = auto$cyl
F2 = auto$displacement
F3 = auto$horsepower
F4 = auto$weight
F5 = auto$acceleration
```

II. Preliminary Statistical Data Analysis

1. Basic Descriptive Statistics

Response Variable (Y) - Miles per Gallon

```
mY = mean(Y)      #mean for Response
stdY = sd(Y)      #standard deviation for Response
rangeY = range(Y) #range for Response
```

Feature Variable 1 (F1) - Number of Cylinders

```
mF1 = mean(F1)          #mean for cylinders
medF1 = median(F1)      #median for cylinders
```

```
stdF1 = sd(F1)            #standard deviation for cylinders
rangeF1 = range(F1)       #range for cylinders
```

Feature Variable 2 (F2) - Engine Displacement (Cubic inches)

```
mF2 = mean(F2)            #mean for displacement
stdF2 = sd(F2)            #standard deviation for displacement
rangeF2 = range(F2)       #range for displacement
```

Feature Variable 3 (F3) - Engine Horsepower (hp)

```
mF3 = mean(F3)            #mean for horsepower
stdF3 = sd(F3)            #standard deviation for horsepower
rangeF3 = range(F3)       #range for horsepower
```

Feature Variable 4 (F4) - Vehicle Weight (lb)

```
mF4 = mean(F4)            #mean for Weight
stdF4 = sd(F4)            #standard deviation for Weight
rangeF4 = range(F4)       #range for Weight
```

Feature Variable 5 (F5) - Vehicle Acceleration (seconds)

```
mF5 = mean(F5)            #mean for Weight
stdF5 = sd(F5)            #standard deviation for Weight
rangeF5 = range(F5)       #range for Weight
```

2. Graphical Description of Variables

Response Variable (Y) - Miles per Gallon

```
#histogram showing Auto MPG
Y_hist = ggplot(auto,aes(mpg)) +
  geom_histogram(aes(y=stat(count)/sum(count)),binwidth = round(stdY)/2, fill = "cadetblu
e", color="#e9ecef") +
  labs(title = "Histogram of MPG")+ xlab("MPG")+ylab("Frequency")+
  theme(plot.title = element_text(hjust = 0.5))
#PDF plot showing Auto MPG
Y_pdf = ggplot(auto, aes(x=mpg)) + geom_density() +
  geom_vline(xintercept = mY - stdY, linetype = "dashed", color = "chartreuse4", size = 1
) +
  geom_text(aes(x = mY-1*stdY, y = 0, label = "-1 SD")) +
  geom_vline(xintercept = mY, linetype = "solid", color = "red", size = 1) +
```

```
  geom_text(aes(x = mY, y = 0, label = "Mean")) +
  geom_vline(xintercept = mY+stdY, linetype = "dotdash", color = "blue", size = 1) +
  geom_text(aes(x = mY+1*stdY, y = 0, label = "+1 SD")) +
  labs(title = "Probability Density of MPG")+ xlab("MPG")+ylab("Density")+
  theme(plot.title = element_text(hjust = 0.5))
Y_hist + Y_pdf
```

Feature Variable 1 (F1) - Number of Cylinders

```
#histogram showing Auto Cylinders
F1_hist = ggplot(auto,aes(cylinders)) +
  geom_histogram(aes(y=stat(count)/sum(count)),binwidth = round(stdF1)/2, fill = "cadetbl
ue", color="#e9ecef") +
  labs(title = "Histogram of Cylinders")+ xlab("Number of Cylinders")+ylab("Frequency")+
  theme(plot.title = element_text(hjust = 0.5))
#PDF plot showing Auto Cylinders
F1_pdf = ggplot(auto, aes(x=cylinders)) + geom_density() +
  geom_vline(xintercept = mF1 - stdF1, linetype = "dashed", color = "chartreuse4", size =
 1) +
  geom_text(aes(x = mF1-1*stdF1, y = 0, label = "-1 SD")) +
  geom_vline(xintercept = mF1, linetype = "solid", color = "red", size = 1) +
  geom_text(aes(x = mF1, y = 0, label = "Mean")) +
  geom_vline(xintercept = mF1+stdF1, linetype = "dotdash", color = "blue", size = 1) +
  geom_text(aes(x = mF1+1*stdF1, y = 0, label = "+1 SD")) +
  labs(title = "Probability Density of Cylinders")+ xlab("Cylinders")+ylab("Density")+
  theme(plot.title = element_text(hjust = 0.5))
F1_hist + F1_pdf
```

Feature Variable 2 (F2) - Engine Displacement (Cubic inches)

```
#histogram showing Auto Displacement
F2_hist = ggplot(auto,aes(displacement)) +
  geom_histogram(aes(y=stat(count)/sum(count)),binwidth = round(stdF2)/2, fill = "cadetbl
ue", color="#e9ecef") +
  labs(title = "Histogram of Displacement")+ xlab("Displacement")+ylab("Frequency")+
  theme(plot.title = element_text(hjust = 0.5))
#PDF plot showing Auto Displacement
F2_pdf = ggplot(auto, aes(x=displacement)) + geom_density() +
  geom_vline(xintercept = mF2 - stdF2, linetype = "dashed", color = "chartreuse4", size =
 1) +
  geom_text(aes(x = mF2-1*stdF2, y = 0, label = "-1 SD")) +
```

```
  geom_vline(xintercept = mF2, linetype = "solid", color = "red", size = 1) +
  geom_text(aes(x = mF2, y = 0, label = "Mean")) +
  geom_vline(xintercept = mF2+stdF2, linetype = "dotdash", color = "blue", size = 1) +
  geom_text(aes(x = mF2+1*stdF2, y = 0, label = "+1 SD")) +
  labs(title = "Probability Density of Displacement")+ xlab("Displacement")+ylab("Density
")+
  theme(plot.title = element_text(hjust = 0.5))
F2_hist + F2_pdf
```

Feature Variable 3 (F3) - Engine Horsepower (hp)

```
#histogram showing Auto Horsepower
F3_hist = ggplot(auto,aes(horsepower)) +
  geom_histogram(aes(y=stat(count)/sum(count)),binwidth = round(stdF3)/2, fill = "cadetbl
ue", color="#e9ecef") +
  labs(title = "Histogram of Horsepower")+ xlab("Horsepower")+ylab("Frequency")+
  theme(plot.title = element_text(hjust = 0.5))
#PDF plot showing Auto Horsepower
F3_pdf = ggplot(auto, aes(x=horsepower)) + geom_density() +
  geom_vline(xintercept = mF3 - stdF3, linetype = "dashed", color = "chartreuse4", size =
 1) +
  geom_text(aes(x = mF3-1*stdF3, y = 0, label = "-1 SD")) +
  geom_vline(xintercept = mF3, linetype = "solid", color = "red", size = 1) +
  geom_text(aes(x = mF3, y = 0, label = "Mean")) +
  geom_vline(xintercept = mF3+stdF3, linetype = "dotdash", color = "blue", size = 1) +
  geom_text(aes(x = mF3+1*stdF3, y = 0, label = "+1 SD")) +
  labs(title = "Probability Density of Horsepower")+ xlab("Horsepower")+ylab("Density")+
  theme(plot.title = element_text(hjust = 0.5))
F3_hist + F3_pdf
```

Feature Variable 4 (F4) - Vehicle Weight (lb)

```
#histogram showing Auto Weight
F4_hist = ggplot(auto,aes(weight)) +
  geom_histogram(aes(y=stat(count)/sum(count)),binwidth = round(stdF4)/2, fill = "cadetbl
ue", color="#e9ecef") +
  labs(title = "Histogram of Weight")+ xlab("Weight")+ylab("Frequency")+
  theme(plot.title = element_text(hjust = 0.5))
#PDF plot showing Auto Weight
F4_pdf = ggplot(auto, aes(x=weight)) + geom_density() +
  geom_vline(xintercept = mF4 - stdF4, linetype = "dashed", color = "chartreuse4", size =
 1) +
  geom_text(aes(x = mF4-1*stdF4, y = 0, label = "-1 SD")) +
```

```
  geom_vline(xintercept = mF4, linetype = "solid", color = "red", size = 1) +
  geom_text(aes(x = mF4, y = 0, label = "Mean")) +
  geom_vline(xintercept = mF4+stdF4, linetype = "dotdash", color = "blue", size = 1) +
  geom_text(aes(x = mF4+1*stdF4, y = 0, label = "+1 SD")) +
  labs(title = "Probability Density of Weight")+ xlab("Weight")+ylab("Density")+
  theme(plot.title = element_text(hjust = 0.5))
F4_hist + F4_pdf
```

Feature Variable 5 (F5) - Vehicle Acceleration (seconds)

```
#histogram showing Auto Acceleration
F5_hist = ggplot(auto,aes(acceleration)) +
  geom_histogram(aes(y=stat(count)/sum(count)),binwidth = round(stdF5)/2, fill = "cadetbl
ue", color="#e9ecef") +
  labs(title = "Histogram of Acceleration")+ xlab("Acceleration")+ylab("Frequency")+
  theme(plot.title = element_text(hjust = 0.5))
#PDF plot showing Auto Acceleration
F5_pdf = ggplot(auto, aes(x=acceleration)) + geom_density() +
  geom_vline(xintercept = mF5 - stdF5, linetype = "dashed", color = "chartreuse4", size =
 1) +
  geom_text(aes(x = mF5-1*stdF5, y = 0, label = "-1 SD")) +
  geom_vline(xintercept = mF5, linetype = "solid", color = "red", size = 1) +
  geom_text(aes(x = mF5, y = 0, label = "Mean")) +
  geom_vline(xintercept = mF5+stdF5, linetype = "dotdash", color = "blue", size = 1) +
  geom_text(aes(x = mF5+1*stdF5, y = 0, label = "+1 SD")) +
  labs(title = "Probability Density Plot of Acceleration")+ xlab("s")+ylab("Density")+
  theme(plot.title = element_text(hjust = 0.5))
F5_hist + F5_pdf
```

3. Graphical Comparison of Variables

```
#Box plot of cylinders compared to MPG
plot(as.factor(F1),Y,main = "MPG vs. Number of Cylinders (F1)", xlab = "Number of Cylinde
rs", ylab = "MPG")

#Box plot of Displacement compared to MPG
plot(F2,Y,main = "MPG vs. Displacement (F2)", xlab = "Displacement", ylab = "MPG")

#Box plot of Horsepower compared to MPG
plot(F3,Y,main = "MPG vs. Horsepower (F3)", xlab = "Horsepower", ylab = "MPG")

#Box plot of Weight compared to MPG
plot(F4,Y,main = "MPG vs. Weight (F4)", xlab = "Weight", ylab = "MPG")
```

```
#Box plot of Acceleration compared to MPG
plot(F5,Y,main = "MPG vs. Acceleration (F5)", xlab = "Acceleration", ylab = "MPG")

#creation of correlation table of features and the responce
cor_values = c(cor(F1,Y), cor(F2,Y), cor(F3,Y), cor(F4,Y), cor(F5,Y))
names(cor_values) = c("MPG vs. Cylinders", "MPG vs. Displacement", "MPG vs. Horsepower",
"MPG vs. Weight", "MPG vs. Acceleration")
kable(cor_values, digits = 2, row.names = TRUE, col.names = "Correlation Coefficient Valu
e", align = "c")
```

4. Feature Correlation Matrix

```
#creation of correlation table of features with other features
names(auto_features) = c("Cylinders", "Displacement", "Horsepower", "Weight", "Accelerati
on")
feature_cor = cor(auto_features)
kable(feature_cor, digits = 2, row.names = TRUE, col.names = names(auto_features),align =
 "c")
```

5. Response Variable Quantile Curve

The quantile curve of the response variable shows the 25%, median (50%), and 75% quantiles for MPG.

```
#quantile curve of MPG data
ggplot(auto,aes(mpg)) + stat_ecdf(geom = "point") + labs(title = "Quantile Curve of MPG",
 x = "MPG" ,y = "Quantile") +
  theme(plot.title = element_text(hjust = 0.5)) +
  geom_vline(xintercept = quantile(Y,.25), linetype = "dashed", color = "chartreuse4", si
ze = 1) +
  geom_text(aes(quantile(Y,.32), y = 0, label = "Q25%")) +
  geom_vline(xintercept = quantile(Y,.5), linetype = "solid", color = "red", size = 1) +
  geom_text(aes(x = quantile(Y,.55), y = 0, label = "Median")) +
  geom_vline(xintercept = quantile(Y,.75), linetype = "dotdash", color = "blue", size = 1
) +
  geom_text(aes(x = quantile(Y,.79), y = 0, label = "Q75%")) +
  coord_flip()
```

6. Linear Regression Models

Summary of Regression Parameters

```
#create Linear models of for each variable to MPG
F1.LM <- lm(Y~F1)            #Linear model of Cylinder with response
F2.LM <- lm(Y~F2)            #Linear model of displacement with response
F3.LM <- lm(Y~F3)            #Linear model of horsepower with response
```

```
F4.LM <- lm(Y~F4)            #Linear model of weight with response
F5.LM <- lm(Y~F5)            #Linear model of acceleration  with response
F1_RMSE = sqrt(c(crossprod(F1.LM$residuals))/length(F1.LM$residuals))        #RMSE of Cyl
inder
F2_RMSE = sqrt(c(crossprod(F2.LM$residuals))/length(F2.LM$residuals))        #RMSE of dis
placement
F3_RMSE = sqrt(c(crossprod(F3.LM$residuals))/length(F3.LM$residuals))        #RMSE of hor
sepower
F4_RMSE = sqrt(c(crossprod(F4.LM$residuals))/length(F4.LM$residuals))        #RMSE of wei
ght
F5_RMSE = sqrt(c(crossprod(F5.LM$residuals))/length(F5.LM$residuals))        #RMSE of acc
eleration
LM_param = data.frame("Slope" = c(F1.LM$coefficients[2],F2.LM$coefficients[2],F3.LM$coeff
icients[2],F4.LM$coefficients[2],F5.LM$coefficients[2]),
                      "Intercept" = c(F1.LM$coefficients[1],F2.LM$coefficients[1],F3.LM$c
oefficients[1],F4.LM$coefficients[1],F5.LM$coefficients[1]),
                      "RMSE" = c(F1_RMSE,F2_RMSE,F3_RMSE,F4_RMSE,F5_RMSE),
                      "Relative Accuracy" = c(F1_RMSE/mY,F2_RMSE/mY,F3_RMSE/mY,F4_RMSE/mY
,F5_RMSE/mY))
kable(LM_param, digits = 2, align = "c")
```

Graphical Representation of Regression Models

```
#linear models of MPG and Cylinders
plot(F1,Y,main = "MPG vs. Number of Cylinders (F1)", xlab = "Number of Cylinders", ylab =
 "MPG")
abline(F1.LM)
cf1 <- round(coef(F1.LM), 2)
eq1 <- paste0("MPG = ", cf1[1],
              ifelse(sign(cf1[2])==1, " + ", " - "), abs(cf1[2]), " * cyl")
eq1_2 <- paste0("RMSE = ", round(F1_RMSE, 3))
mtext(eq1,3,line = -2)
mtext(eq1_2,3,line = -3)
```

```
#linear models of MPG and displacement
plot(F2,Y,main = "MPG vs. Displacement (F2)", xlab = "Displacement", ylab = "MPG")
abline(F2.LM)
cf2 <- round(coef(F2.LM), 2)
eq2 <- paste0("MPG = ", cf2[1],
              ifelse(sign(cf2[2])==1, " + ", " - "), abs(cf2[2]), " * dis")
eq2_2 <- paste0("RMSE = ", round(F2_RMSE, 3))
mtext(eq2,3,line = -2)
mtext(eq2_2,3,line = -3)
```

```r
#linear models of MPG and Horsepower
plot(F3,Y,main = "MPG vs. Horsepower (F3)", xlab = "Horsepower", ylab = "MPG")
abline(F3.LM)
cf3 <- round(coef(F3.LM), 2)
eq3 <- paste0("MPG = ", cf3[1],
              ifelse(sign(cf3[2])==1, " + ", " - "), abs(cf3[2]), " * hp")
eq3_2 <- paste0("RMSE = ", round(F3_RMSE, 3))
mtext(eq3,3,line = -2)
mtext(eq3_2,3,line = -3)

#linear models of MPG and Weight
plot(F4,Y,main = "MPG vs. Weight (F4)", xlab = "Weight", ylab = "MPG")
abline(F4.LM)
cf4 <- round(coef(F4.LM), 2)
eq4 <- paste0("MPG = ", cf4[1],
              ifelse(sign(cf4[2])==1, " + ", " - "), abs(cf4[2]), " * weight")
eq4_2 <- paste0("RMSE = ", round(F4_RMSE, 3))
mtext(eq4,3,line = -2)
mtext(eq4_2,3,line = -3)

#linear models of MPG and acceleration
plot(F5,Y,main = "MPG vs. Acceleration (F4)", xlab = "Acceleration", ylab = "MPG")
abline(F5.LM)
cf5 <- round(coef(F5.LM), 2)
eq5 <- paste0("MPG = ", cf5[1],
              ifelse(sign(cf5[2])==1, " + ", " - "), abs(cf5[2]), " * acc")
eq5_2 <- paste0("RMSE = ", round(F5_RMSE, 3))
mtext(eq5,3,line = -2)
mtext(eq5_2,3,line = -3)
```

III. Automatic Classification of Data by KNN

8. Data Subsetting

```r
#Define new data frames for low, mid, high MPG classes. Using the quantiles of .33, .66,
and 1.
auto_lowMPG = subset(auto, mpg < quantile(mpg,0.33))
auto_medMPG = subset(auto, mpg >= quantile(mpg,0.33) & mpg <= quantile(mpg,0.66))
auto_highMPG = subset(auto, mpg>quantile(mpg,0.66))
#Define features datafame (excluding MPG)
auto_features = auto[-1]
#count the number of rows for LOW, MED, and HIGH subsets
n_low = nrow(auto_lowMPG)
```

```
n_med = nrow(auto_medMPG)
n_high = nrow(auto_highMPG)

#Define response and feature vectors for each subset
Y_low = auto_lowMPG$mpg
Y_med = auto_medMPG$mpg
Y_high = auto_highMPG$mpg
#Feature 1
F1_low = auto_lowMPG$cylinders
F1_med = auto_medMPG$cylinders
F1_high = auto_highMPG$cylinders
#Feature 2
F2_low = auto_lowMPG$displacement
F2_med = auto_medMPG$displacement
F2_high = auto_highMPG$displacement
#Feature 3
F3_low = auto_lowMPG$horsepower
F3_med = auto_medMPG$horsepower
F3_high = auto_highMPG$horsepower
#Feature 4
F4_low = auto_lowMPG$weight
F4_med = auto_medMPG$weight
F4_high = auto_highMPG$weight
#Feature 5
F5_low = auto_lowMPG$acceleration
F5_med = auto_medMPG$acceleration
F5_high = auto_highMPG$acceleration
```

9. Comparison of Low vs. High MPG Classes

```
#F1 low and high plots for cylinders
F1_low_hist = ggplot(auto_lowMPG,aes(cylinders)) +
  geom_histogram(aes(y=stat(count)/sum(count)),binwidth = round(stdF1)/2, fill = "cadetbl
ue", color="#e9ecef") +
  labs(title = "Histogram of F1 (Low MPG)")+ xlab("Cylinders")+ylab("Frequency")+
  theme(plot.title = element_text(hjust = 0.5))
F1_high_hist = ggplot(auto_highMPG,aes(cylinders)) +
  geom_histogram(aes(y=stat(count)/sum(count)),binwidth = round(stdF1)/2, fill = "cadetbl
ue", color="#e9ecef") +
  labs(title = "Histogram of F1 (High MPG)")+ xlab("Cylinders")+ylab("Frequency")+
  theme(plot.title = element_text(hjust = 0.5))
F1_low_hist + F1_high_hist
```

```r
#F1 low and high plots for displacement
F2_low_hist = ggplot(auto_lowMPG,aes(displacement)) +
  geom_histogram(aes(y=stat(count)/sum(count)),binwidth = round(stdF2)/2, fill = "cadetbl
ue", color="#e9ecef") +
  labs(title = "Histogram of F2 (Low MPG)")+ xlab("Displacement")+ylab("Frequency")+
  theme(plot.title = element_text(hjust = 0.5))
F2_high_hist = ggplot(auto_highMPG,aes(displacement)) +
  geom_histogram(aes(y=stat(count)/sum(count)),binwidth = round(stdF2)/4, fill = "cadetbl
ue", color="#e9ecef") +
  labs(title = "Histogram of F2 (High MPG)")+ xlab("Displacement")+ylab("Frequency")+
  theme(plot.title = element_text(hjust = 0.5))

F2_low_hist + F2_high_hist

#F1 low and high plots for horsepower
F3_low_hist = ggplot(auto_lowMPG,aes(horsepower)) +
  geom_histogram(aes(y=stat(count)/sum(count)),binwidth = round(stdF3)/2, fill = "cadetbl
ue", color="#e9ecef") +
  labs(title = "Histogram of F3 (Low MPG)")+ xlab("Horsepower")+ylab("Frequency")+
  theme(plot.title = element_text(hjust = 0.5))
F3_high_hist = ggplot(auto_highMPG,aes(horsepower)) +
  geom_histogram(aes(y=stat(count)/sum(count)),binwidth = round(stdF3)/4, fill = "cadetbl
ue", color="#e9ecef") +
  labs(title = "Histogram of F3 (High MPG)")+ xlab("Horsepower")+ylab("Frequency")+
  theme(plot.title = element_text(hjust = 0.5))

F3_low_hist + F3_high_hist

#F1 low and high plots for weight
F4_low_hist = ggplot(auto_lowMPG,aes(weight)) +
  geom_histogram(aes(y=stat(count)/sum(count)),binwidth = round(stdF4)/2, fill = "cadetbl
ue", color="#e9ecef") +
  labs(title = "Histogram of F4 (Low MPG)")+ xlab("Weight")+ylab("Frequency")+
  theme(plot.title = element_text(hjust = 0.5))
F4_high_hist = ggplot(auto_highMPG,aes(weight)) +
  geom_histogram(aes(y=stat(count)/sum(count)),binwidth = round(stdF4)/2, fill = "cadetbl
ue", color="#e9ecef") +
  labs(title = "Histogram of F4 (High MPG)")+ xlab("Weight")+ylab("Frequency")+
  theme(plot.title = element_text(hjust = 0.5))

F4_low_hist + F4_high_hist
```

```
#F1 low and high plots accleeration
F5_low_hist = ggplot(auto_lowMPG,aes(acceleration)) +
  geom_histogram(aes(y=stat(count)/sum(count)),binwidth = round(stdF5)/2, fill = "cadetbl
ue", color="#e9ecef") +
  labs(title = "Histogram of F5 (Low MPG)")+ xlab("Acceleration")+ylab("Frequency")+
  theme(plot.title = element_text(hjust = 0.5))
F5_high_hist = ggplot(auto_highMPG,aes(acceleration)) +
  geom_histogram(aes(y=stat(count)/sum(count)),binwidth = round(stdF5)/2, fill = "cadetbl
ue", color="#e9ecef") +
  labs(title = "Histogram of F5 (High MPG)")+ xlab("Acceleration")+ylab("Frequency")+
  theme(plot.title = element_text(hjust = 0.5))

F5_low_hist + F5_high_hist
```

10. Basic Descriptive Statistics for Low and High MPG Classes

Feature Variable 1 (F1) - Number of Cylinders

```
#basic statistics for cylinders
mF1_low = mean(F1_low)                    #mean for low cylinders
medF1_low = median(F1_low)                #median for low cylinders
stdF1_low = sd(F1_low)                    #standard derivation for low cylinders
MEF1_low = 1.6*stdF1_low/sqrt(n_low)      #Confidence interval for low cylinders

mF1_high = mean(F1_high)                    #mean for high cylinders
medF1_high = median(F1_high)                #median for high cylinders
stdF1_high = sd(F1_high)                    #standard derivation for high cylinders
MEF1_high = 1.6*stdF1_high/sqrt(n_high)     #Confidence interval for high cylinders
```

Feature Variable 2 (F2) - Engine Displacement (Cubic inches)

```
mF2_low = mean(F2_low)                    #mean for low Displacement
stdF2_low = sd(F2_low)                    #median for low Displacement
MEF2_low = 1.6*stdF2_low/sqrt(n_low)      #Confidence interval for low Displacement


mF2_high = mean(F2_high)                    #mean for high Displacement
stdF2_high = sd(F2_high)                    #median for high Displacement
MEF2_high = 1.6*stdF2_high/sqrt(n_high)     #Confidence interval for high Displacement
```

Feature Variable 3 (F3) - Engine Horsepower (hp)

```
mF3_low = mean(F3_low)                    #mean for low Horsepower
stdF3_low = sd(F3_low)                    #median for low Horsepower
MEF3_low = 1.6*stdF3_low/sqrt(n_low)      #Confidence interval for low Horsepower

mF3_high = mean(F3_high)                    #mean for high Horsepower
stdF3_high = sd(F3_high)                    #median for high Horsepower
MEF3_high = 1.6*stdF3_high/sqrt(n_high)     #Confidence interval for high Horsepower
```

Feature Variable 4 (F4) - Vehicle Weight (lb)

```
mF4_low = mean(F4_low)                    #mean for low Weight
stdF4_low = sd(F4_low)                    #median for low Weight
MEF4_low = 1.6*stdF4_low/sqrt(n_low)      #Confidence interval for low Weight

mF4_high = mean(F4_high)                    #mean for high Weight
stdF4_high = sd(F4_high)                    #median for high Weight
MEF4_high = 1.6*stdF4_high/sqrt(n_high)     #Confidence interval for high Weight
```

Feature Variable 5 (F5) - Vehicle Acceleration (seconds)

```
mF5_low = mean(F5_low)                     #mean for low Acceleration
stdF5_low = sd(F5_low)                     #median for low Acceleration
MEF5_low = 1.6*stdF5_low/sqrt(n_low)       #Confidence interval for low Acceleration

mF5_high = mean(F5_high)                     #mean for high Acceleration
stdF5_high = sd(F5_high)                     #median for high Acceleration
MEF5_high = 1.6*stdF5_high/sqrt(n_high)     #Confidence interval for high Acceleration
```

11. Application of the k-Nearest Neighbor (kNN) Automatic Classifier

```
#Adding column with class label for each subset
label = c(rep("LOW",n_low))
auto_lowMPG = cbind(auto_lowMPG,label)

label = c(rep("MED",n_med))
auto_medMPG = cbind(auto_medMPG, label)

label = c(rep("HIGH",n_high))
auto_highMPG = cbind(auto_highMPG, label)

#Using rsample and caret packages.
#The initial_split function randomly partitions each normalized data set into 80/20 train
```

```
ing/test splits.
low_Sample <- initial_split(auto_lowMPG, prop = .8)
med_Sample <- initial_split(auto_medMPG, prop = .8)
high_Sample <- initial_split(auto_highMPG, prop = .8)

#Each partition is re-grouped into a global training/test data frame.
Train_data <- rbind(training(low_Sample),training(med_Sample),training(high_Sample))
Test_data <- rbind(testing(low_Sample),testing(med_Sample),testing(high_Sample))

#k = 5
#applied on the train dataset
k5_train = sum(Train_data[,7] == knn(Train_data[2:6], Train_data[2:6], Train_data[,7], k
= 5)) / nrow(Train_data[7])

#applied on the test dataset
k5_test = sum(Test_data[,7] == knn(Train_data[2:6], Test_data[2:6], Train_data[,7], k = 5
)) / nrow(Test_data[7])
```

12. Determining Best K-Value

```
#____________  graphs
K <- c(3,5,7,9,11,13,15,17,19,29,39)   #vector of K values that we may test on
TrainAccuracy <- c()                   #empty vector to full values of train accuracy
TestAccuracy <- c()                    #empty vector to full values of test accuracy
for (i in 1:length(K)){
  TrainAccuracy[i] <- sum(Train_data[,7] == knn(Train_data[2:6], Train_data[2:6], Train_d
ata[,7], k = K[i])) / nrow(Train_data[7])
  TestAccuracy[i] <- sum(Test_data[,7] == knn(Train_data[2:6], Test_data[2:6], Train_data
[,7], k = K[i])) / nrow(Test_data[7])
}
accuracy_df <- cbind(TrainAccuracy, TestAccuracy, K)    #combine back the vectors of trai
n accuracy, test accuracy, and k
accuracy_df <- as.data.frame(accuracy_df)               #mark the given list/dataframe as
 a dataframe

ggplot(accuracy_df, aes(K)) +                                          #using GGplot
 package for plotting accuracy of ks
  geom_line(aes(y = TrainAccuracy, color = "Training Set")) +
  geom_point(aes(y = TrainAccuracy),shape = 15) +
  geom_line(aes(y = TestAccuracy, color = "Test Set")) +
  geom_point(aes(y = TestAccuracy),shape = 16) +
  labs(title = "Training & Test Set Accuracy",x="k-value",y="% Accuracy") +
  theme(plot.title = element_text(hjust = 0.5)) +
```

```
  scale_color_manual("",
                     breaks = c("Training Set", "Test Set"),
                     values = c("Training Set" = "red", "Test Set" = "blue"))

#Get the Confusion Matrix for our KNN model
training_matrix <- knn(Train_data[2:6], Train_data[2:6], Train_data[,7], k = 3)        #a
t K = 3 KNN model
y <- table(training_matrix,Train_data[,7])                                             #c
reate a table of the correct labels from our train dataset
confusionMatrix(y)[2]                                                                  #c
reation of the confusion matrix of the train model

## $table
##
## training_matrix HIGH LOW MED
##            HIGH   90   1  14
##            LOW     0  93   6
##            MED    15  10  84

testing_matrix <-knn(Train_data[2:6], Test_data[2:6], Train_data[,7], k = 3)           #a
t K = 3 KNN model
y <- table(testing_matrix,Test_data[,7])                                               #c
reate a table of the correct labels from our test dataset
confusionMatrix(y)[2]                                                                  #c
reation of the confusion matrix of the test model

## $table
##
## testing_matrix HIGH LOW MED
##           HIGH   18   0   7
##           LOW     0  22   4
##           MED     9   4  15
```

13. Effect of Feature Normalization on kNN Accuracy

```
#normalize data function
normalize = function(x){
  return((x - mean(x))/sd(x))
}
#Creating new df "auto_norm" by normalizing feature data from original auto data set. Usi
ng the lapply function to
auto_norm = as.data.frame(lapply(auto[,2:6], normalize))
auto_norm = cbind(auto[1], auto_norm)                        #adding mpg column to normali
zed feature data.
```

```
#Subsetting auto_norm df into low, med, high classes.
auto_lowMPG_norm = subset(auto_norm, mpg < quantile(mpg,0.33))
    #LOWmpg normal data subset
auto_medMPG_norm = subset(auto_norm, mpg >= quantile(mpg,0.33) & mpg <= quantile(mpg,0.66
))  #MEDmpg normal data subset
auto_highMPG_norm = subset(auto_norm, mpg>quantile(mpg,0.66))
    #HIGHmpg normal data subset

#Adding column with class label for each subset
label = c(rep("LOW",n_low))         #creation of low labels
auto_lowMPG_norm = cbind(auto_lowMPG_norm,label) #combine back the labels to the dataset

label = c(rep("MED",n_med))     #creation of med labels
auto_medMPG_norm = cbind(auto_medMPG_norm, label)  #combine back the labels to the datase
t

label = c(rep("HIGH",n_high))  #creation of high labels
auto_highMPG_norm = cbind(auto_highMPG_norm, label)  #combine back the labels to the data
set

#Using rsample and caret packages.
#The initial_split function randomly partitions each normalized data set into 80/20 train
ing/test splits.
low_Sample_n <- initial_split(auto_lowMPG_norm, prop = .8)#LOW sample split at 80%
med_Sample_n <- initial_split(auto_medMPG_norm, prop = .8)#MED sample split at 80%
high_Sample_n <- initial_split(auto_highMPG_norm, prop = .8)#HIGH sample split at 80%

#Each partition is re-grouped into a global training/test data frame.
Train_data_n <- rbind(training(low_Sample_n),training(med_Sample_n),training(high_Sample_
n))  #combine the values that are in the train dataset
Test_data_n <- rbind(testing(low_Sample_n),testing(med_Sample_n),testing(high_Sample_n))
  #combine the values that are in the test dataset

#k = 5
#applied on the train dataset
k5_train_n = sum(Train_data_n[,7] == knn(Train_data_n[2:6], Train_data_n[2:6], Train_data
_n[,7], k = 5)) / nrow(Train_data_n[7])

#applied on the test dataset
k5_test_n = sum(Test_data_n[,7] == knn(Train_data_n[2:6], Test_data_n[2:6], Train_data_n[
,7], k = 5)) / nrow(Test_data_n[7])
```

```r
training_matrix_n <- knn(Train_data_n[2:6], Train_data_n[2:6], Train_data_n[,7], k = 3)
     #at K = 3 KNN model
y <- table(training_matrix_n,Train_data_n[,7])
     #create a table of the correct labels from our train dataset
confusionMatrix(y)[2]   #creation of the confusion matrix of the train model

testing_matrix_n <-knn(Train_data_n[2:6], Test_data_n[2:6], Train_data_n[,7], k = 3)
  #at K = 3 KNN model
y <- table(testing_matrix,Test_data_n[,7])
 #create a table of the correct labels from our test dataset
confusionMatrix(y)[2] #creation of the confusion matrix of the test model
```