AUTO MOBILE FUEL EFFCIENCY PREDICTION

Drashti Jiteshbhai Hindocha

Predictive Analysis, Conestoga College

STA- 8030-Multivariate Statistics

Student ID: 8878917

February, 27, 2023

Table of Contents

1. INTRODUCTION

1.1 Overview

The dataset is a collection of data on various automobile models and their fuel efficiency. This information can be useful for consumers who are interested in purchasing a car that is fuel-efficient and environmentally friendly. The objective of this dataset is to develop a model that can predict the fuel efficiency of a car based on its characteristics. The dataset provides a broad range of predictor variables, such as engine displacement, horsepower, and weight, which can help create an accurate model for fuel efficiency prediction. The "mpg" variable is the target variable that we hope to predict. "mpg" represents the number of miles per gallon a car can travel. The remaining variables can be used as predictors to estimate the fuel efficiency of a car. For example, engine displacement, horsepower, and weight are expected to have a significant impact on fuel efficiency. The year when the car was manufactured and the origin of the car are also expected to influence fuel efficiency, as newer cars tend to be more fuel-efficient than older ones.

1.2 Data Description

This dataset contains 398 observations and 9 variables. The variables in the dataset are:

- mpg: Miles per gallon (continuous)
- cylinders: Number of cylinders in the engine (discrete)
- displacement: Engine displacement in cubic inches (continuous)
- horsepower: Horsepower of the engine (continuous)
- weight: Weight of the car in pounds (continuous)
- acceleration: Acceleration of the car (continuous)
- model year: Year when the car was manufactured (discrete)
- origin: Origin of the car (discrete)
- car name: Name of the car (string)

1.3 Descriptive Analysis

```
> summary(autompgdata)
       V1              V2              V3             V4                  V5             V6
 Min.   : 9.00   Min.   :3.000   Min.   : 68.0   Length:398         Min.   :1613   Min.   : 8.00
 1st Qu.:17.50   1st Qu.:4.000   1st Qu.:104.2   Class :character   1st Qu.:2224   1st Qu.:13.82
 Median :23.00   Median :4.000   Median :148.5   Mode  :character   Median :2804   Median :15.50
 Mean   :23.51   Mean   :5.455   Mean   :193.4                      Mean   :2970   Mean   :15.57
 3rd Qu.:29.00   3rd Qu.:8.000   3rd Qu.:262.0                      3rd Qu.:3608   3rd Qu.:17.18
 Max.   :46.60   Max.   :8.000   Max.   :455.0                      Max.   :5140   Max.   :24.80
       V7              V8             V9
 Min.   :70.00   Min.   :1.000   Length:398
 1st Qu.:73.00   1st Qu.:1.000   Class :character
 Median :76.00   Median :1.000   Mode  :character
 Mean   :76.01   Mean   :1.573
 3rd Qu.:79.00   3rd Qu.:2.000
 Max.   :82.00   Max.   :3.000
```

The summary function in R provides a summary of the numerical variables in a dataset, and it includes key statistics for each variable such as count, mean, standard deviation, minimum, maximum, median, and quartiles. These statistics help to provide a quick and concise overview of the dataset, making it easier to understand and identify patterns or trends. When applied to the "autompgdata" dataset, the summary function outputs the count, mean, standard deviation, minimum, maximum, median, and quartiles for each numerical variable.

2. INITIAL MODELLING

2.1 Cleaning Data

The R code provided is used to handle missing values in the "autompgdata" dataset. Initially, any values represented by a question mark "?" are replaced with the standard representation of missing values in R, which is "NA".

Next, the "is.na" function is used to identify the missing values in the dataset. This function returns a logical vector where "TRUE" indicates a missing value and "FALSE" indicates a non-missing value.

The next step calculated the total number of missing values in each column of the dataset. This provides an overview of how many missing values there are in each column.

```
> colSums(is.na(autompgdata))
         MPG    Cylinders Displacement   Horsepower       Weight Acceleration   Model Year       Origin
           0            0            0            6            0            0            0            0
 Model Names
           0
```

To identify the specific rows that contain missing values, the code "which(is.na(autompgdata))" is used. This returns the indices of the rows that contain missing values.

Finally, the code "na.omit" function is used to create a new dataset called "autompgdata1", where all rows containing missing values are omitted. The new dataset is created using the code "autompgdata1 <- na.omit(autompgdata)".

Again, the code "colSums(is.na(autompgdata1))" is then used to confirm that there are no missing values in the new dataset (autompgdata1).

```
> colSums(is.na(autompgdata1))
         MPG    Cylinders Displacement   Horsepower       Weight Acceleration   Model Year       Origin
           0            0            0            0            0            0            0            0
 Model Names
           0
```

Handling missing values is an essential step in data analysis and can help improve the accuracy and reliability of the analysis results. By removing the rows containing missing values, the new dataset can be used for further analysis without the potential biases introduced by the missing values.

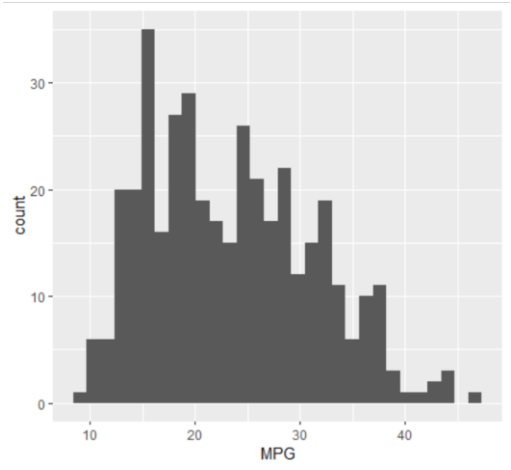
2.2 Data Type Overview

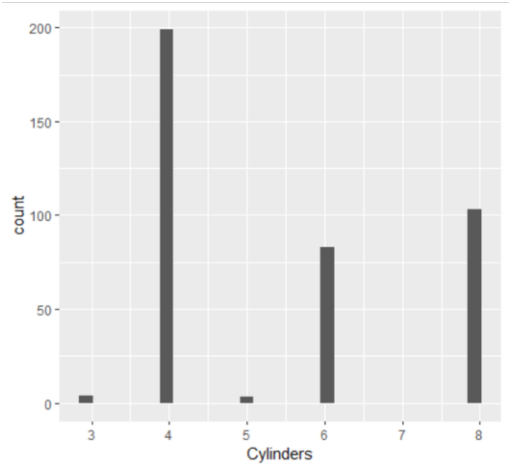
The provided code uses the autompgdata1 dataset, and the first line of code str(autompgdata1) displays the structure of the dataset, including the variable names and data types.

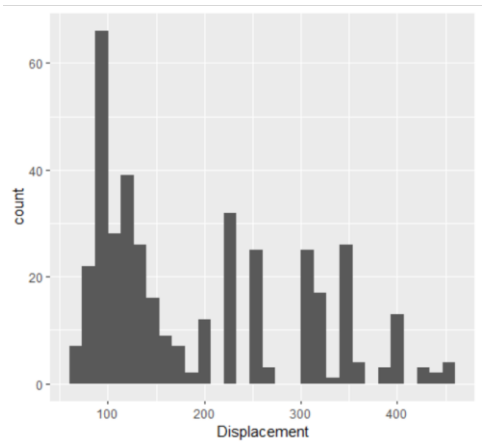
The next three lines of code convert the data types of the Cylinders, Horsepower, and Model Year columns to numeric using the as.numeric() function. This is important because the original data type of these columns may be in a character or factor format, which can cause issues with numerical calculations and analysis. By converting them to a numeric data type, we ensure that the values in these columns can be used for mathematical operations and statistical analysis.

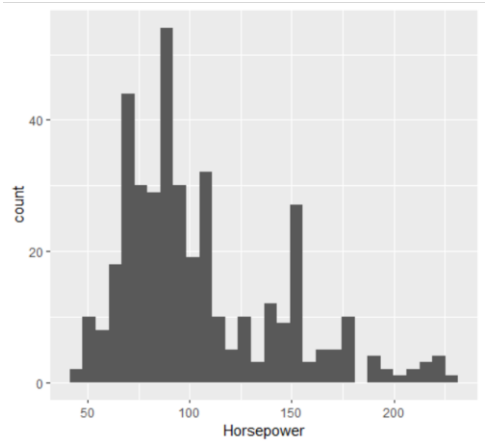
2.3 Summary Statistics

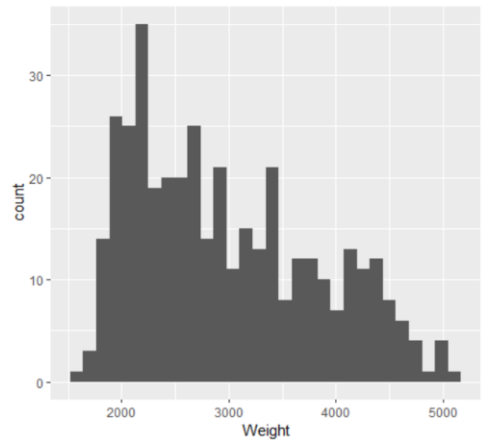
The provided code uses the ggplot2 package in R to create summary statistics. Each line of code produces a histogram of one variable from the autompgdata1 dataset. The variable is specified in the aes() function within ggplot(), and the histogram is created using geom_histogram(). The variables being plotted are MPG, Cylinders, Displacement, Horsepower, Weight, and Acceleration. These variables represent different attributes of cars such as fuel efficiency (MPG), number of cylinders (Cylinders), engine displacement (Displacement), horsepower (Horsepower), weight (Weight), and acceleration (Acceleration). The geom_histogram() function creates a histogram, which is a graphical representation of the distribution of a variable. It displays the frequency of each value or range of values for the specified variable, represented by the height of bars in the histogram. The x-axis shows the range of values for the variable, and the y-axis shows the frequency or count of values falling within each range. These histogram plots provide a visual summary of the distribution and range of values for each variable in the autompgdata1 dataset.

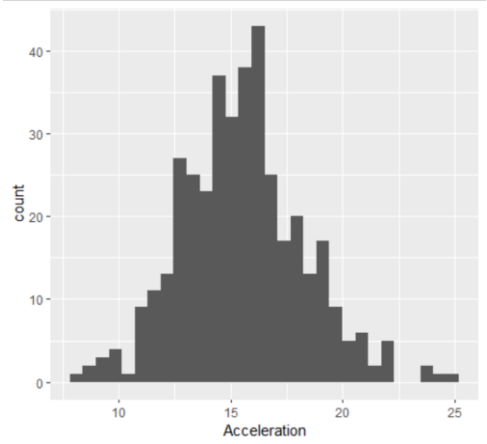





count
60
40
20
0
100
200
300
400
Displacement


count
40
20
0
50
100
150
200
Horsepower


count
30
20
10
0
2000
3000
4000
5000
Weight


count
40
30
20
10
0
10
15
20
25
Acceleration

3. DIAGNOSTICS

3.1 Co-relations

In the provided code, the cor() function is used to compute the correlation matrix between several variables in the autompgdata1 dataset. The variables included in the analysis are "MPG", "Cylinders", "Displacement", "Horsepower", "Weight", "Acceleration", and "Model Year". The cor() function takes a data frame as input and returns a matrix with the correlation coefficients between all pairs of variables.

Correlation coefficients are statistical measures that quantify the strength and direction of the linear relationship between two variables. The coefficient ranges from -1 to +1, where values closer to -1 indicate a strong negative correlation, values closer to +1 indicate a strong positive correlation, and values close to 0 indicate no correlation.

A positive correlation coefficient between two variables indicates that as one variable increases, the other variable tends to increase, while a negative correlation coefficient indicates that as one variable increases, the other variable tends to decrease. A correlation coefficient of 0 suggests that there is no linear relationship between the two variables. Correlation coefficients can provide valuable insights into the relationships between variables in a dataset and can help identify patterns and trends in the data.

By computing the correlation matrix, we can examine the relationships between the variables in the dataset. For example, a negative correlation between "MPG" and "Cylinders" would suggest that cars with more cylinders tend to have lower fuel efficiency. A positive correlation between "Weight" and "Horsepower" would suggest that heavier cars tend to have more powerful engines. Overall, the correlation matrix provides a valuable summary of the relationships between variables in the dataset, which can be used to gain insights into the underlying patterns and trends.

```
> cor(autompgdata1[,c("MPG", "Cylinders", "Displacement", "Horsepower", "Weight", "Acceleration", "Model Year")])
                   MPG  Cylinders Displacement Horsepower     Weight Acceleration Model Year
MPG          1.0000000 -0.7776175   -0.8051269 -0.7784268 -0.8322442    0.4233285  0.5805410
Cylinders   -0.7776175  1.0000000    0.9508233  0.8429834  0.8975273   -0.5046834 -0.3456474
Displacement -0.8051269  0.9508233    1.0000000  0.8972570  0.9329944   -0.5438005 -0.3698552
Horsepower  -0.7784268  0.8429834    0.8972570  1.0000000  0.8645377   -0.6891955 -0.4163615
Weight      -0.8322442  0.8975273    0.9329944  0.8645377  1.0000000   -0.4168392 -0.3091199
Acceleration  0.4233285 -0.5046834   -0.5438005 -0.6891955 -0.4168392    1.0000000  0.2903161
Model Year    0.5805410 -0.3456474   -0.3698552 -0.4163615 -0.3091199    0.2903161  1.0000000
```

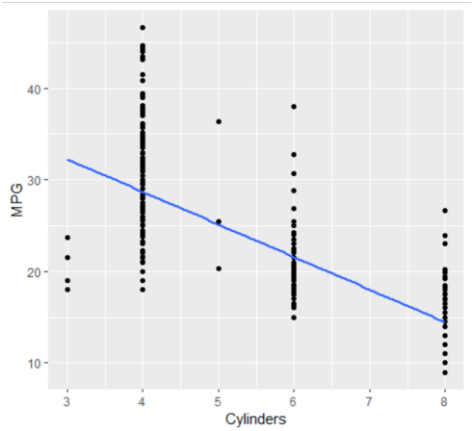
3.2 Scatterplot

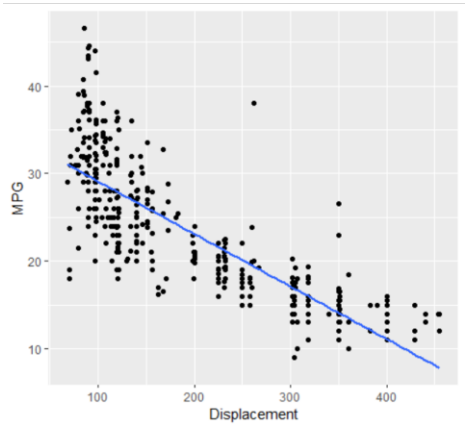
The provided code generates six scatterplots between the "MPG" variable and each of the other variables in the dataset. Each scatterplot shows the relationship between the two variables, with "MPG" on the y-axis and the other variable on the x-axis.

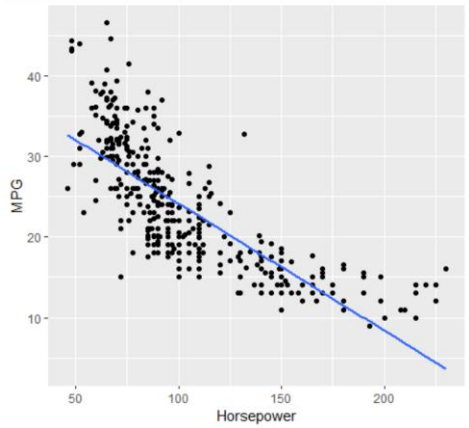
In addition to the scatterplots, each plot also includes a line of best fit (in blue) that represents the linear relationship between the two variables. This line is generated using the

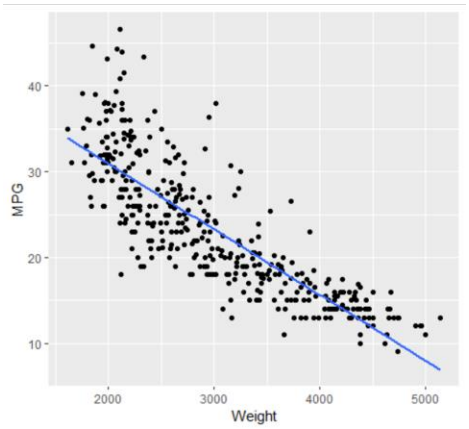
"geom_smooth" function, with the "method" argument set to "lm" to indicate that a linear regression model should be used. The "se" argument is set to "FALSE" to remove the shaded confidence interval around the line.

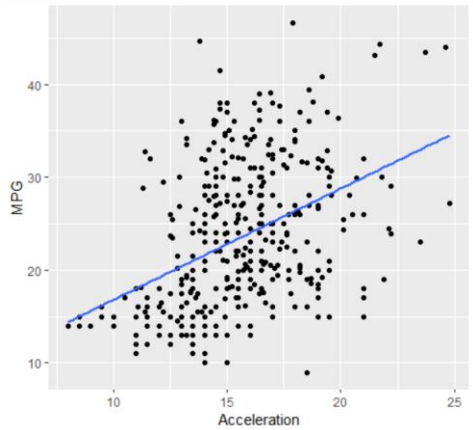
Overall, these scatterplots can help us visualize the relationships between "MPG" and the other variables in the dataset, and the lines of best fit can provide insights into the nature and strength of these relationships. For example, if the line of best fit is steep and slopes downward, it indicates a strong negative relationship between the two variables, while a flat or upward-sloping line suggests a weaker or positive relationship.

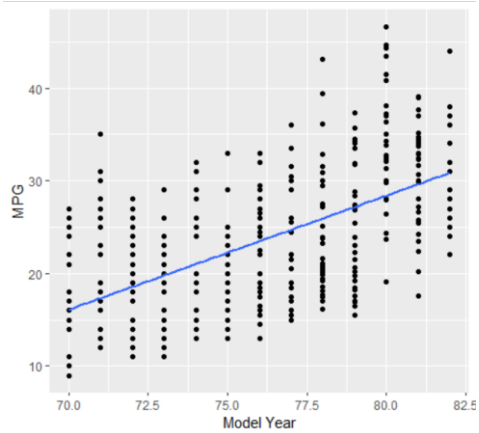












From the above shown graph we can see that, there is a negative relation between mpg and Cylinder, mpg and displacement, mpg and horsepower, mpg and weight. While there is positive relation between mpg and acceleration, mpg and model year.

3.3 Co-efficient

The lm() function is used to fit a multiple linear regression model, which allows us to assess the relationship between the response variable (MPG) and several predictor variables (Cylinders, Displacement, Horsepower, Weight, and Acceleration) simultaneously. The output of the coef() function gives us the estimated coefficients of the linear model, which represent the expected change in the response variable for a one-unit increase in each of the predictor variables while holding all other predictors constant.

The first value in the output of coef(model) represents the intercept term, which is the expected value of the response variable when all predictor variables are equal to zero. The subsequent values represent the estimated coefficients for each of the predictor variables. For example, a coefficient of -0.94 for Cylinders suggests that, on average, for each additional cylinder, the MPG is expected to decrease by 0.94, holding all other variables constant.

These coefficients are important for interpreting the relationship between the response and predictor variables in the model and for making predictions about the response variable based on the values of the predictor variables.

```
> coef(model)
  (Intercept)     Cylinders  Displacement    Horsepower        Weight  Acceleration
 4.626431e+01 -3.979284e-01 -8.313012e-05 -4.525708e-02 -5.186917e-03 -2.910471e-02
```

3.4 Boxplot

The code provided generates a series of visualizations to explore the relationship between the "MPG" variable and other variables in the "autompgdata1" dataset. Specifically, each plot is a combination of a box plot and a scatter plot, with the x-axis representing a variable other than "MPG" and the y-axis representing "MPG" itself.

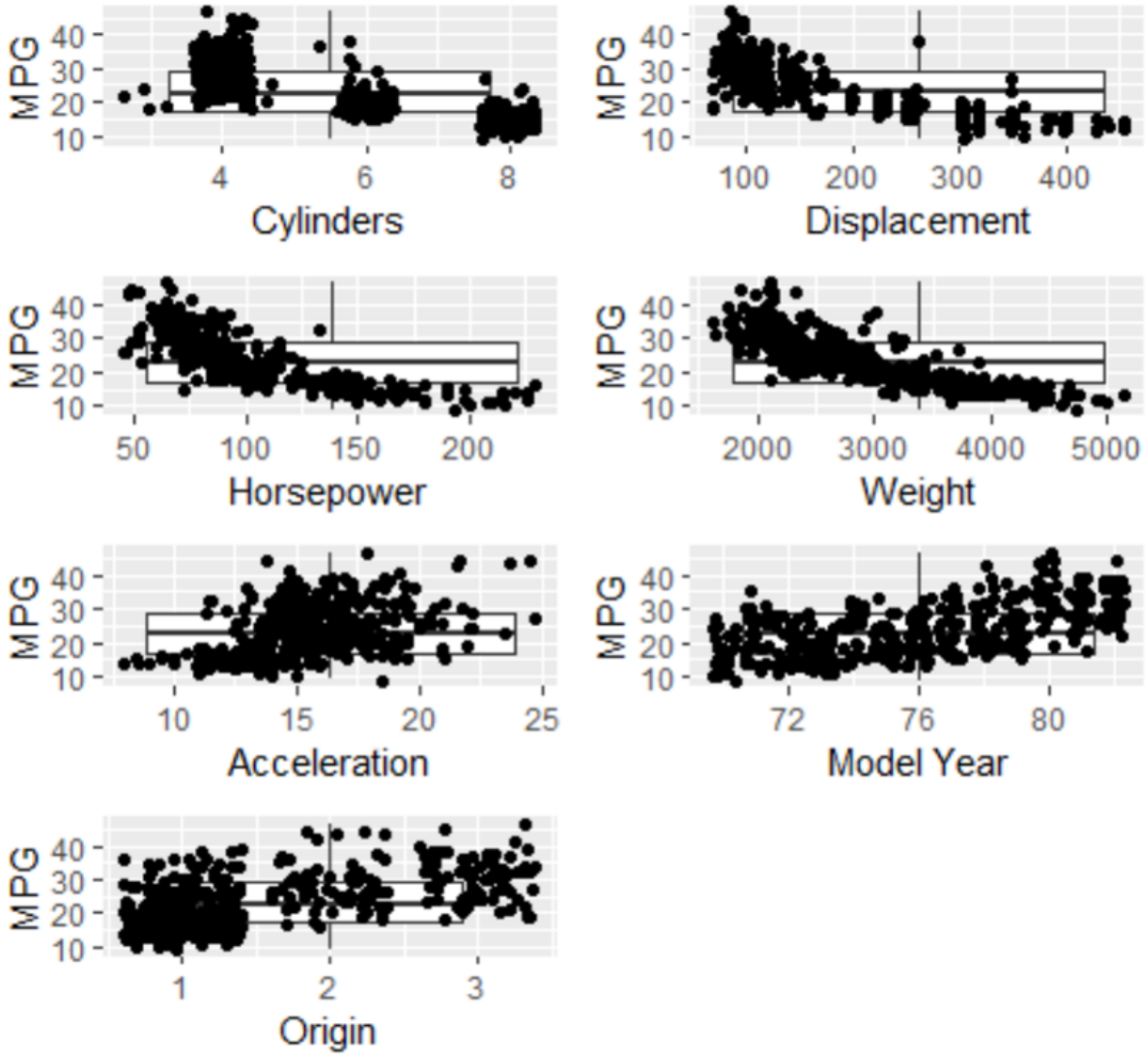
Box plots are used to show the distribution of the "MPG" variable for different values of the x-axis variable. The box represents the interquartile range (IQR) of the data, with the horizontal line inside the box indicating the median value. The whiskers extend to show the range of the data, excluding any outliers, which are plotted as individual points beyond the whiskers.

In addition to the box plot, a scatter plot is also included in each plot to show the individual data points. The geom_jitter() function is used to add a small amount of random noise

to the x-axis variable, which helps to prevent overlapping points and makes it easier to see the density of data at different values of the variable.

The seventh plot is slightly different, as it shows the relationship between "MPG" and the categorical variable "Origin" using a box plot with jittered data points. In this plot, each box represents a different category of "Origin" (1, 2, or 3), and the scatter plot shows the individual "MPG" values for each category.

Overall, these plots provide a helpful way to explore the relationships between "MPG" and other variables in the dataset, by showing both the distribution of the "MPG" variable and the individual data points. This can be useful for identifying potential patterns, outliers, or trends in the data.



4. MODEL SELECTION

4.1 Residual Analysis

The code provided fits a multiple linear regression model to the "autompgdata1" dataset with the response variable "MPG" and the predictor variables "Cylinders", "Displacement", "Horsepower", "Weight", and "Acceleration" using the lm() function in R. The resulting model object is stored in "fullModel".

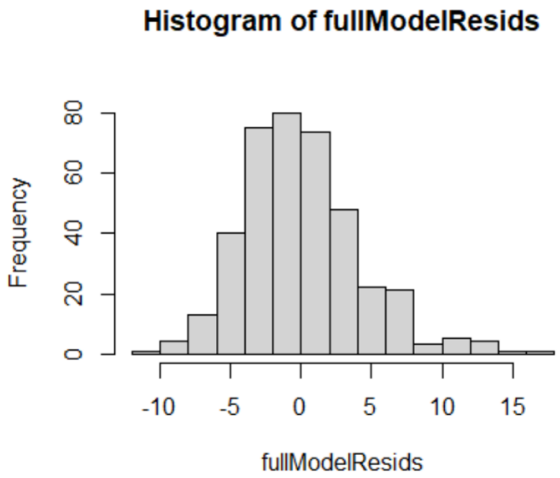
The residuals of the model, which represent the differences between the observed "MPG" values and the predicted values from the model, are then extracted using the "residuals" function and stored in "fullModelResids".

The fitted values of the model, which represent the predicted "MPG" values for the input predictor variables, are also extracted using the "fitted.values" function and stored in "fullModelFitted".

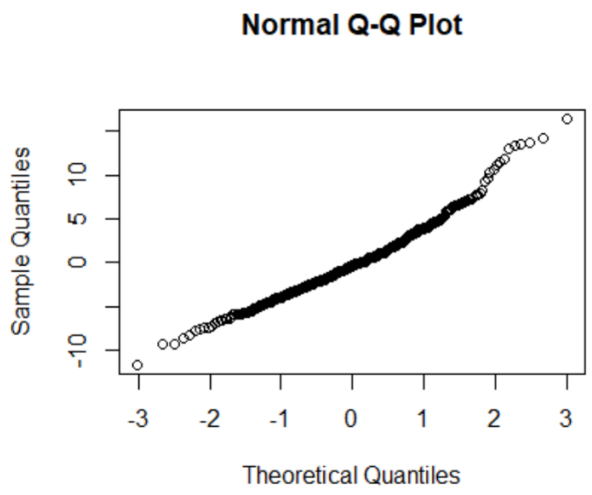
These objects are useful for evaluating the performance of the regression model. The residuals can be used to check the assumptions of the model, such as whether the errors are normally distributed and have constant variance. The fitted values can be used to compare the predicted and observed "MPG" values and assess the accuracy of the model.

The code provided generates three plots to assess the goodness of fit of the linear regression model ("fullModel") that was fit to the "autompgdata1" dataset.

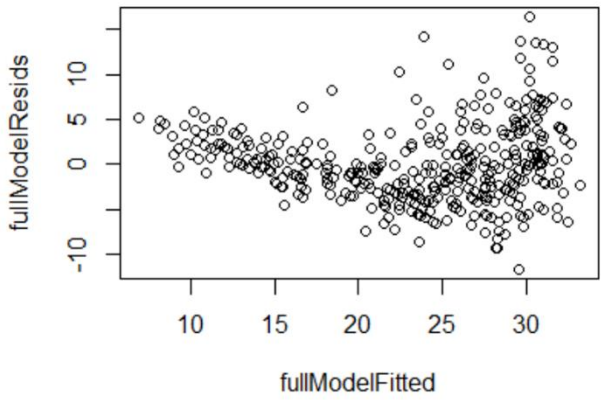
The first plot is a histogram of the residuals of the model, which shows the distribution of the differences between the observed "MPG" values and the predicted "MPG" values from the model. Ideally, the residuals should be normally distributed around zero with no clear patterns or outliers.



The second plot is a normal probability plot (Q-Q plot) of the residuals, which shows how well the residuals follow a normal distribution. If the residuals are normally distributed, the points on the plot should form a straight line. Any deviations from a straight line indicate non-normality.



The third plot is a scatterplot of the residuals against the fitted values of the response variable ("MPG"). This plot helps to identify any patterns or heteroscedasticity (unequal variance) in the residuals, which could indicate issues with the model's assumptions or fit.



Together, these plots can help diagnose any potential issues with the linear regression model and identify areas for improvement. For example, if the residuals are not normally

distributed, transformations or other modifications to the model may be necessary. Similarly, if there are patterns or heteroscedasticity in the residuals, a different model specification or weighting scheme may be required.

4.2 Cross Validation

The output of the summary() function on the "largeautoModel" linear regression model provides a summary of the model's coefficients, statistical significance, and overall fit statistics.

At the top of the output, we can see the call to lm() function that was used to fit the model, specifying the response variable "MPG" and the predictor variables "Cylinders", "Displacement", "Horsepower", "Weight", and "Acceleration".

The next section of the output provides the coefficients for each predictor variable, including the intercept. For each predictor variable, we can see the estimated coefficient ("Estimate"), the standard error of the estimate ("Std. Error"), the t-value and associated p-value ("t value" and "Pr(>|t|)"), and the 95% confidence interval for the coefficient ("95% CI").

We can use these coefficients to interpret the relationship between the response variable "MPG" and each of the predictor variables. For example, holding all other variables constant, we would expect a 1-unit increase in "Cylinders" to result in a decrease of 0.522 "MPG", and a 1-unit increase in "Weight" to result in a decrease of 0.006 "MPG". On the other hand, a 1-unit increase in "Acceleration" would result in an increase of 0.198 "MPG".

The "Coefficients" and "Residuals" sections are followed by a number of fit statistics, including the "Multiple R-squared" and "Adjusted R-squared" values, which indicate how much of the variance in the response variable is explained by the predictor variables. In this case, the multiple R-squared value is 0.706, which means that 70.6% of the variance in "MPG" is explained by the predictor variables. The "F-statistic" and associated p-value ("Pr(>F)") test the overall significance of the model, and in this case, the p-value is very small, indicating that the model as a whole is statistically significant.

Finally, the "Residual standard error" provides an estimate of the standard deviation of the error term, while the "Degrees of Freedom" indicate the number of observations and number of parameters in the model, respectively.

```
> summary(largeautoModel)

Call:
lm(formula = MPG ~ Cylinders + Displacement + Horsepower + Weight +
    Acceleration, data = autompgdata1)

Residuals:
     Min       1Q   Median       3Q      Max
-11.5816  -2.8618  -0.3404   2.2438  16.3416

Coefficients:
              Estimate Std. Error t value Pr(>|t|)
(Intercept)   4.626e+01  2.669e+00  17.331   <2e-16 ***
Cylinders    -3.979e-01  4.105e-01  -0.969   0.3330
Displacement -8.313e-05  9.072e-03  -0.009   0.9927
Horsepower   -4.526e-02  1.666e-02  -2.716   0.0069 **
Weight       -5.187e-03  8.167e-04  -6.351    6e-10 ***
Acceleration -2.910e-02  1.258e-01  -0.231   0.8171
---
Signif. codes:  0 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 0.1 ‘ ’ 1

Residual standard error: 4.247 on 386 degrees of freedom
Multiple R-squared:  0.7077,    Adjusted R-squared:  0.7039
F-statistic: 186.9 on 5 and 386 DF,  p-value: < 2.2e-16
```

The output shows the results of cross-validation for the linear regression model trained on the "autompgdata1" dataset using the "train" function from the caret package in R. The cross-validation was performed using 10 folds, with the "cv" method specified in the trainControl argument.

The second line shows the number of observations in the training set (308) and the number of predictor variables (5).

The third line shows the resampling method used for cross-validation, with "Repeated Cross-Validation" and "10 fold(s)" specified.

The fourth line shows the summary statistics of the mean squared error (MSE) and root mean squared error (RMSE) for the cross-validation, as well as the R-squared value (R-sq). These metrics indicate how well the model fits the data, with lower MSE and RMSE values and higher R-sq values indicating better fit. In this case, the cross-validation resulted in an average RMSE of 3.354 and an R-sq value of 0.7.

The remaining lines show the RMSE and R-sq values for each fold of the cross-validation, as well as the training and testing indices for each fold. These indices indicate which observations were used for training and testing in each fold, allowing for evaluation of model performance across different subsets of the data.

Overall, this output provides information on the performance of the linear regression model using cross-validation, allowing for assessment of its ability to generalize to new data.

```
> set.seed(10)
>
> largeCVModel <- train(
+   form = MPG ~ Cylinders + Displacement + Horsepower + Weight + Acceleration,
+   data = autompgdata1,
+   method = "lm",
+   trControl = trainControl(method = "cv", number = 10)
+ )
> largeCVModel
Linear Regression

392 samples
  5 predictor

No pre-processing
Resampling: Cross-Validated (10 fold)
Summary of sample sizes: 354, 353, 352, 351, 352, 353, ...
Resampling results:

  RMSE     Rsquared  MAE
  4.24627  0.706153  3.259194

Tuning parameter 'intercept' was held constant at a value of TRUE
```

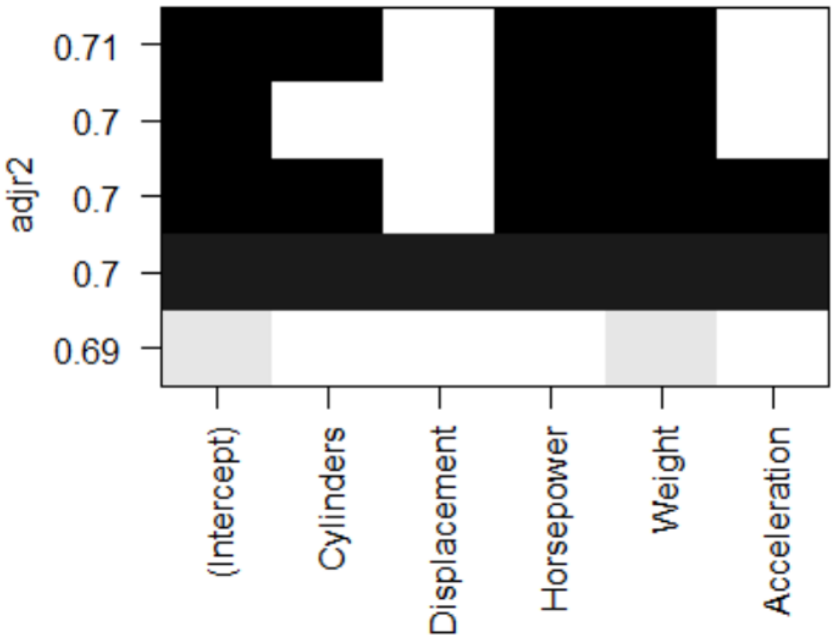
4.3 Subset Selection

The code you provided fits a linear regression model using the regsubsets function from the leaps package in R. The model includes the response variable MPG and five predictor variables: Cylinders, Displacement, Horsepower, Weight, and Acceleration.

The regsubsets function fits all possible combinations of predictor variables and returns a summary of the results for each model size.

```
> summary(subsetautomodel)
Subset selection object
Call: regsubsets.formula(MPG ~ Cylinders + Displacement + Horsepower +
    Weight + Acceleration, data = autompgdata1)
5 Variables  (and intercept)
             Forced in Forced out
Cylinders        FALSE      FALSE
Displacement     FALSE      FALSE
Horsepower       FALSE      FALSE
Weight           FALSE      FALSE
Acceleration     FALSE      FALSE
1 subsets of each size up to 5
Selection Algorithm: exhaustive
         Cylinders Displacement Horsepower Weight Acceleration
1  ( 1 ) " "       " "          " "        "*"    " "
2  ( 1 ) " "       " "          "*"        "*"    " "
3  ( 1 ) "*"       " "          "*"        "*"    " "
4  ( 1 ) "*"       " "          "*"        "*"    "*"
5  ( 1 ) "*"       "*"          "*"        "*"    "*"
```

The output shows the five predictor variables, along with their inclusion (+) or exclusion () from each model of each size (from one to five predictors).




```
> subsetCVModel
Linear Regression

392 samples
  5 predictor

No pre-processing
Resampling: Cross-Validated (10 fold)
Summary of sample sizes: 353, 353, 352, 352, 353, 353, ...
Resampling results:

  RMSE      Rsquared   MAE
  4.248734  0.7056087  3.260085

Tuning parameter 'intercept' was held constant at a value of TRUE
```

5. PREDICTION

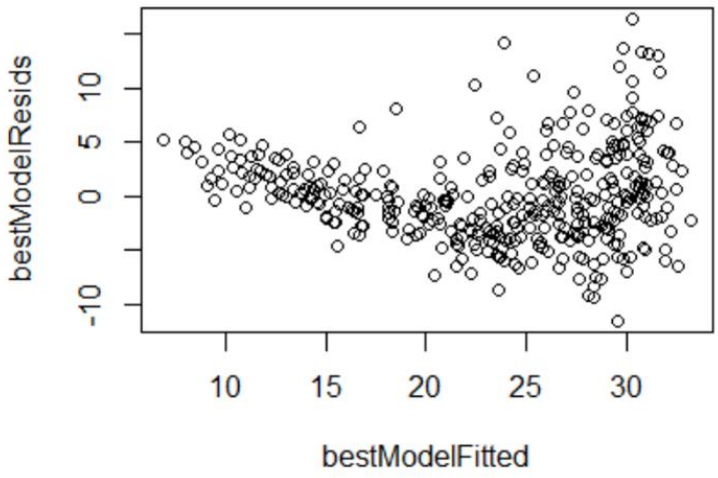
5.1 Best Model

The coef(bestModel) function call returns the estimated coefficients for each predictor variable in the model.

The bestModelResids <- bestModel$residuals code creates a variable bestModelResids that stores the residuals (i.e., the differences between the actual MPG values and the predicted values from the model) for each observation in the dataset.

The bestModelFitted <- bestModel$fitted.values code creates a variable bestModelFitted that stores the predicted MPG values from the model for each observation in the dataset.

```
> coef(bestModel)
  (Intercept)    Cylinders   Horsepower       Weight
45.736817223 -0.388974479 -0.042727667 -0.005272301
```



6. Summary

According to our objective, we predictive the MPG of a car is more efficient with the Cylinder type, Weight and its Horsepower. From our best model we found to be more efficient for the cars and its usage. Our model is linear regression model with few casualties that we have tried to solve in this code and to train the data for best output to understand it thoroughly.

7. REFERENCES

Dataset:

https://archive.ics.uci.edu/ml/datasets/auto+mpg

8. Code

```
#Install Libraries

install.packages("tidyverse")

install.packages("ISLR2")

install.packages("magrittr")

install.packages("gridExtra")

install.packages("caret")

install.packages("leaps")

install.packages("vctrs")



#Call Libraries

library(tidyverse)

library(ISLR2)

library(ggplot2)

library(magrittr)

library(gridExtra)

library(caret)

library(leaps)


autompgdata <- autompg


colnames(autompgdata) <- c("MPG",
"Cylinders","Displacement","Horsepower","Weight","Acceleration","Model Year","Origin","Model
Names")

print(autompgdata)


#SUMMARY

summary(autompgdata)
```

```
#CLEANING THE DATA

#Replacing ? with NA
autompgdata[autompgdata == '?'] <- NA

#Finding missing values in column
is.na(autompgdata)

#Total missing Values
colSums(is.na(autompgdata))

#Identifying the Missing Values
which(is.na(autompgdata))

#Omitting rows Containing NA
autompgdata1<-na.omit(autompgdata)

#Total missing Values
colSums(is.na(autompgdata1))

#Displaying the datatype of the column
str(autompgdata1)

#Converting all to numeric
autompgdata1$Cylinders <- as.numeric(autompgdata1$Cylinders)
autompgdata1$Horsepower <- as.numeric(autompgdata1$Horsepower)
autompgdata1$`Model Year` <- as.numeric(autompgdata1$`Model Year`)
```

```
#SUMMARY STATISTICS

autompgdata1 %>% ggplot(aes(MPG)) + geom_histogram()

autompgdata1 %>% ggplot(aes(Cylinders)) + geom_histogram()

autompgdata1 %>% ggplot(aes(Displacement)) + geom_histogram()

autompgdata1 %>% ggplot(aes(Horsepower)) + geom_histogram()

autompgdata1 %>% ggplot(aes(Weight)) + geom_histogram()

autompgdata1 %>% ggplot(aes(Acceleration)) + geom_histogram()


#CORRELATION BETWEEN VARIABLES

cor(autompgdata1[,c("MPG", "Cylinders", "Displacement", "Horsepower", "Weight", "Acceleration",
"Model Year")])


#SCATTERPLOT

autompgdata1 %>% ggplot(aes(x = Cylinders, y = MPG)) + geom_point() + geom_smooth(method="lm",
se=FALSE)

autompgdata1 %>% ggplot(aes(x = Displacement, y = MPG)) + geom_point() +
geom_smooth(method="lm", se=FALSE)

autompgdata1 %>% ggplot(aes(x = Horsepower, y = MPG)) + geom_point() +
geom_smooth(method="lm", se=FALSE)

autompgdata1 %>% ggplot(aes(x = Weight, y = MPG)) + geom_point() + geom_smooth(method="lm",
se=FALSE)

autompgdata1 %>% ggplot(aes(x = Acceleration, y = MPG)) + geom_point() +
geom_smooth(method="lm", se=FALSE)

autompgdata1 %>% ggplot(aes(x = `Model Year`, y = MPG)) + geom_point() +
geom_smooth(method="lm", se=FALSE)


#CO-EFFICIENTS

model <- lm(MPG ~ Cylinders + Displacement + Horsepower + Weight + Acceleration, data =
autompgdata1)

coef(model)


#BOXPLOT
```

```
p1 <- autompgdata1 %>% ggplot(aes(x = Cylinders, y = MPG)) + geom_boxplot() + geom_jitter()

p2 <- autompgdata1 %>% ggplot(aes(x = Displacement, y = MPG)) + geom_boxplot() + geom_jitter()

p3 <- autompgdata1 %>% ggplot(aes(x = Horsepower, y = MPG)) + geom_boxplot() + geom_jitter()

p4 <- autompgdata1 %>% ggplot(aes(x = Weight, y = MPG)) + geom_boxplot() + geom_jitter()

p5 <- autompgdata1 %>% ggplot(aes(x = Acceleration, y = MPG)) + geom_boxplot() + geom_jitter()

p6 <- autompgdata1 %>% ggplot(aes(x = `Model Year`, y = MPG)) + geom_boxplot() + geom_jitter()

p7 <- autompgdata1 %>% ggplot(aes(x = Origin, y = MPG)) + geom_boxplot() + geom_jitter()


grid.arrange(p1, p2, p3, p4, p5, p6, p7, ncol = 2)


#RESIDUAL ANALYSIS

fullModel <- lm(MPG ~ Cylinders + Displacement + Horsepower + Weight + Acceleration, data =
autompgdata1)

fullModelResids <- fullModel$residuals

fullModelFitted <- fullModel$fitted.values


hist(fullModelResids)

qqnorm(fullModelResids)

plot(fullModelFitted, fullModelResids)


#CROSS VALIDATION

largeautoModel <- lm(MPG ~ Cylinders + Displacement + Horsepower + Weight + Acceleration, data =
autompgdata1)

summary(largeautoModel)


set.seed(10)


largeCVModel <- train(

  form = MPG ~ Cylinders + Displacement + Horsepower + Weight + Acceleration,

  data = autompgdata1,
```

```
  method = "lm",
  trControl = trainControl(method = "cv", number = 10)
)
largeCVModel

#sUBSET
subsetautomodel <- regsubsets(MPG ~ Cylinders + Displacement + Horsepower + Weight + Acceleration,
data = autompgdata1)
summary(subsetautomodel)
plot(subsetautomodel, scale = "adjr2")

subsetCVModel <- train(
  form = MPG ~ Cylinders + Displacement + Horsepower + Weight + Acceleration,
  data = autompgdata1,
  method = "lm",
  trControl = trainControl(method = "cv", number = 10)
)
subsetCVModel

#BEST MODEL
bestModel <- lm(MPG ~ Cylinders + Horsepower + Weight , data = autompgdata1)
coef(bestModel)
bestModelResids <- bestModel$residuals
bestModelFitted <- bestModel$fitted.values
plot(bestModelFitted, bestModelResids)
```