Multiple Linear Regression in R

The following is a multi-linear regression assessment of a given 1985 auto imports database developed by Jeffrey C. Schlimmer and maintained at the University of California, School of Information and Computer Science data repository. The main goal during this exercise is to explore this dataset, develop a multi-linear regression model using R, and identify unique characteristics of this machine learning modeling algorithm.

This multivariate dataset was consolidated from various sources including the 1985 Ward's Automotive Yearbook, Personal Auto Manuals and Insurance Services Office, New York, and Insurance Collision Report – Insurance Institute for Highway Safety of Washington, D.C. a variety of attributes including, continuous and nominal values as well as missing values. The dataset consists of 26 attributes and 205 observations with a mix of categorical, integers, numeric and factor types of variables including missing values (Dua, D. and Graff, C. (2019).

Per the original repository, the below list describes each of the attributes of this dataset.

- symboling: -3, -2, -1, 0, 1, 2, 3.
- normalized-losses: continuous from 65 to 256.
- make: alfa-romero, audi, bmw, chevrolet, dodge, honda, isuzu, jaguar, mazda, mercedes-benz, mercury, mitsubishi, nissan, peugot, plymouth, porsche, renault, saab, subaru, toyota, volkswagen, volvo
- fuel-type: diesel, gas.
- aspiration: std, turbo.
- num-of-doors: four, two.
- body-style: hardtop, wagon, sedan, hatchback, convertible.
- drive-wheels: 4wd, fwd, rwd.
- engine-location: front, rear.
- wheel-base: continuous from 86.6 120.9.
- length: continuous from 141.1 to 208.1.
- width: continuous from 60.3 to 72.3.
- height: continuous from 47.8 to 59.8.
- curb-weight: continuous from 1488 to 4066.
- engine-type: dohc, dohcv, l, ohc, ohcf, ohcv, rotor.
- num-of-cylinders: eight, five, four, six, three, twelve, two.
- engine-size: continuous from 61 to 326.
- fuel-system: 1bbl, 2bbl, 4bbl, idi, mfi, mpfi, spdi, spfi.
- bore: continuous from 2.54 to 3.94.
- stroke: continuous from 2.07 to 4.17.
- compression-ratio: continuous from 7 to 23.
- horsepower: continuous from 48 to 288.
- peak-rpm: continuous from 4150 to 6600.
- city-mpg: continuous from 13 to 49.
- highway-mpg: continuous from 16 to 54.
- **price**: continuous from 5118 to 45400.

The dependent variable (response) of this dataset is the price of the vehicle, while the other remaining 25 variables will be considered independents or predictors. My goal is to illustrate how a multi-linear regression analysis can enable or empower auto sales and insurance services with immediate quotes and/or valuations of new or used vehicle based on some particular characteristics of the car.

Preview of the data set. The first step was to set and load the data to be analyzed. The imports_85.csv dataframe was loaded into the console using the read.csv( ) command and saved as a vector named *import*. The image below shows a preview of the set.

```
> #1 Set the working directory/load the data
> setwd("~/OneDrive/UMGC/DATA630/Week4/")
> # Read the csv file
> import <- read.csv("imports_85.csv", header=TRUE, sep = ",",as.is=FALSE)
> head(import)          #Preview the dataframe
```

```
> head(import)
   symboling normalized_losses fuel_type aspiration num_doors body_style drive_wheels wheel_base length width height curb_weight
4          2               164       gas        std      four      sedan          fwd       99.8  176.6  66.2   54.3        2337
5          2               164       gas        std      four      sedan          4wd       99.4  176.6  66.4   54.3        2824
7          1               158       gas        std      four      sedan          fwd      105.8  192.7  71.4   55.7        2844
9          1               158       gas      turbo      four      sedan          fwd      105.8  192.7  71.4   55.9        3086
11         2               192       gas        std       two      sedan          rwd      101.2  176.8  64.8   54.3        2395
12         0               192       gas        std      four      sedan          rwd      101.2  176.8  64.8   54.3        2395
   engine_size bore stroke compression_ratio horsepower peak_rpm city_mpg highway_mpg Price
4          109 3.19    3.4              10.0        102     5500       24          30 13950
5          136 3.19    3.4               8.0        115     5500       18          22 17450
7          136 3.19    3.4               8.5        110     5500       19          25 17710
9          131 3.13    3.4               8.3        140     5500       17          20 23875
11         108 3.50    2.8               8.8        101     5800       23          29 16430
12         108 3.50    2.8               8.8        101     5800       23          29 16925
```

Data preprocessing. During the data preprocessing phase, and as directed, I excluded four of the original dataframe (df) variables using the NULL command. Those variables were engine_type, make, num_of_cylinders, and fuel_system.

```
> #2 Data Pre Processing
> import$engine_type<-NULL          #Remove the variable
> import$make<-NULL                 #Remove the variable
> import$num_of_cylinders<-NULL     #Remove the variable
> import$fuel_system<-NULL          #Remove the variable
```

Following this action, I checked the data structure with str( ) and noticed NA values across some of the attributes (figure 1.1). After familiarizing myself with these missing values, I decided to ignore these observations containing NA values with the na.omit( ) command. In retrospect, this data preparation step proved to be critical in preparation of the modeling of the data. The newly updated df was reassigned to the "import" dataset. Then, to confirm my previous actions, and confirmed the content of the df, I pulled a summary( ) command as shown in figure 1.2.

```
str(import)                      #Check data structure
```

```
> str(import)                 #Check data structure
'data.frame':   205 obs. of  22 variables:
 $ symboling        : int  3 3 1 2 2 2 1 1 1 0 ...
 $ normalized_losses: int  NA NA NA 164 164 NA 158 NA 158 NA ...
 $ fuel_type        : Factor w/ 2 levels "diesel","gas": 2 2 2 2 2 2 2 2 2 2 ...
 $ aspiration       : Factor w/ 2 levels "std","turbo": 1 1 1 1 1 1 1 1 2 2 ...
 $ num_doors        : Factor w/ 2 levels "four","two": 2 2 2 1 1 2 1 1 1 2 ...
 $ body_style       : Factor w/ 5 levels "convertible",..: 1 1 3 4 4 4 4 5 4 3 ...
 $ drive_wheels     : Factor w/ 3 levels "4wd","fwd","rwd": 3 3 3 2 1 2 2 2 2 1 ...
 $ engine_location  : Factor w/ 2 levels "front","rear": 1 1 1 1 1 1 1 1 1 1 ...
 $ wheel_base       : num  88.6 88.6 94.5 99.8 99.4 ...
 $ length           : num  169 169 171 177 177 ...
 $ width            : num  64.1 64.1 65.5 66.2 66.4 66.3 71.4 71.4 71.4 67.9 ...
 $ height           : num  48.8 48.8 52.4 54.3 54.3 53.1 55.7 55.7 55.9 52 ...
 $ curb_weight      : int  2548 2548 2823 2337 2824 2507 2844 2954 3086 3053 ...
 $ engine_size      : int  130 130 152 109 136 136 136 136 131 131 ...
 $ bore             : num  3.47 3.47 2.68 3.19 3.19 3.19 3.19 3.19 3.13 3.13 ...
 $ stroke           : num  2.68 2.68 3.47 3.4 3.4 3.4 3.4 3.4 3.4 3.4 ...
 $ compression_ratio: num  9 9 9 10 8 8.5 8.5 8.5 8.3 7 ...
 $ horsepower       : int  111 111 154 102 115 110 110 110 140 160 ...
 $ peak_rpm         : int  5000 5000 5000 5500 5500 5500 5500 5500 5500 5500 ...
 $ city_mpg         : int  21 21 19 24 18 19 19 19 17 16 ...
 $ highway_mpg      : int  27 27 26 30 22 25 25 25 20 22 ...
 $ Price            : int  13495 16500 16500 13950 17450 15250 17710 18920 23875 23870 ...
>
```

Figure 1.1 – The import data frame contains 205 observations across 22 variables.

```
import<-na.omit(import)          #Exclude rows with NA values
summary(import)                  #Check the descriptive statistics
```

```
> import<-na.omit(import)          #Exclude variables w/ NA values in prep. for model
> summary(import)                  #Check the descriptive statistics
   symboling       normalized_losses  fuel_type   aspiration  num_doors        body_style drive_wheels engine_location
 Min.   :-2.0000   Min.   : 65.0     diesel: 15   std  :132   four:96    convertible: 2   4wd:  8      front:160
 1st Qu.: 0.0000   1st Qu.: 94.0     gas   :145   turbo: 28   two :64    hardtop    : 5   fwd:106      rear :  0
 Median : 1.0000   Median :114.0                                         hatchback  :56   rwd: 46
 Mean   : 0.7375   Mean   :121.3                                         sedan      :80
 3rd Qu.: 2.0000   3rd Qu.:148.0                                         wagon      :17
 Max.   : 3.0000   Max.   :256.0
   wheel_base        length          width           height         curb_weight    engine_size         bore           stroke
 Min.   : 86.60   Min.   :141.1   Min.   :60.3   Min.   :49.40   Min.   :1488   Min.   : 61.0   Min.   :2.540   Min.   :2.070
 1st Qu.: 94.50   1st Qu.:165.5   1st Qu.:64.0   1st Qu.:52.00   1st Qu.:2073   1st Qu.: 97.0   1st Qu.:3.050   1st Qu.:3.107
 Median : 96.90   Median :172.2   Median :65.4   Median :54.10   Median :2338   Median :110.0   Median :3.270   Median :3.270
 Mean   : 98.24   Mean   :172.3   Mean   :65.6   Mean   :53.88   Mean   :2459   Mean   :119.1   Mean   :3.298   Mean   :3.237
 3rd Qu.:100.60   3rd Qu.:177.8   3rd Qu.:66.5   3rd Qu.:55.50   3rd Qu.:2809   3rd Qu.:134.5   3rd Qu.:3.550   3rd Qu.:3.410
 Max.   :115.60   Max.   :202.6   Max.   :71.7   Max.   :59.80   Max.   :4066   Max.   :258.0   Max.   :3.940   Max.   :4.170
 compression_ratio   horsepower        peak_rpm       city_mpg       highway_mpg        Price
 Min.   : 7.00     Min.   : 48.00   Min.   :4150   Min.   :15.00   Min.   :18.00   Min.   : 5118
 1st Qu.: 8.70     1st Qu.: 69.00   1st Qu.:4800   1st Qu.:23.00   1st Qu.:28.00   1st Qu.: 7384
 Median : 9.00     Median : 88.00   Median :5200   Median :26.00   Median :32.00   Median : 9164
 Mean   :10.15     Mean   : 95.88   Mean   :5116   Mean   :26.51   Mean   :32.07   Mean   :11428
 3rd Qu.: 9.40     3rd Qu.:114.00   3rd Qu.:5500   3rd Qu.:31.00   3rd Qu.:37.00   3rd Qu.:14559
 Max.   :23.00     Max.   :200.00   Max.   :6600   Max.   :49.00   Max.   :54.00   Max.   :35056
>
```

Figure 1.2 – Descriptive statistics of the dataset. The engine_location variable appears as a one-value variable.

After omitting the NA values and running the summary command, I noticed the engine_location variable with only one-type of value (front = 160); however, I decided to press forward without any additional changes. Further in the process, as I was building the model, RStudio gave me the below "error in contrasts", which urged me to return to this step and remove the one-value variable of engine_location using the NULL command.

```
model<-lm(Price~., train.data)
```

```
Error in `contrasts<-`(`*tmp*`, value = contr.funs[1 +
isOF[nn]]):contrasts can be applied only to factors with 2 or more
levels
```

```
> import$engine_location<-NULL     # Remove the one-value variable>
summary(import)                    #Check the descriptive statistics
```

```
> import$engine_location<-NULL    # Remove one-value variable in prep. for model
> summary(import)                 #Check the descriptive statistics
   symboling       normalized_losses  fuel_type   aspiration  num_doors        body_style drive_wheels   wheel_base    
 Min.   :-2.0000   Min.   : 65.0     diesel: 15   std  :132   four:96    convertible: 2   4wd:  8      Min.   : 86.60  
 1st Qu.: 0.0000   1st Qu.: 94.0     gas   :145   turbo: 28   two :64    hardtop    : 5   fwd:106      1st Qu.: 94.50  
 Median : 1.0000   Median :114.0                                         hatchback  :56   rwd: 46      Median : 96.90  
 Mean   : 0.7375   Mean   :121.3                                         sedan      :80                Mean   : 98.24  
 3rd Qu.: 2.0000   3rd Qu.:148.0                                         wagon      :17                3rd Qu.:100.60  
 Max.   : 3.0000   Max.   :256.0                                                                       Max.   :115.60  
     length          width           height        curb_weight    engine_size         bore           stroke      compression_ratio
 Min.   :141.1   Min.   :60.3   Min.   :49.40   Min.   :1488   Min.   : 61.0   Min.   :2.540   Min.   :2.070   Min.   : 7.00    
 1st Qu.:165.5   1st Qu.:64.0   1st Qu.:52.00   1st Qu.:2073   1st Qu.: 97.0   1st Qu.:3.050   1st Qu.:3.107   1st Qu.: 8.70    
 Median :172.2   Median :65.4   Median :54.10   Median :2338   Median :110.0   Median :3.270   Median :3.270   Median : 9.00    
 Mean   :172.3   Mean   :65.6   Mean   :53.88   Mean   :2459   Mean   :119.1   Mean   :3.298   Mean   :3.237   Mean   :10.15    
 3rd Qu.:177.8   3rd Qu.:66.5   3rd Qu.:55.50   3rd Qu.:2809   3rd Qu.:134.5   3rd Qu.:3.550   3rd Qu.:3.410   3rd Qu.: 9.40    
 Max.   :202.6   Max.   :71.7   Max.   :59.80   Max.   :4066   Max.   :258.0   Max.   :3.940   Max.   :4.170   Max.   :23.00    
   horsepower        peak_rpm        city_mpg      highway_mpg        Price      
 Min.   : 48.00   Min.   :4150   Min.   :15.00   Min.   :18.00   Min.   : 5118  
 1st Qu.: 69.00   1st Qu.:4800   1st Qu.:23.00   1st Qu.:28.00   1st Qu.: 7384  
 Median : 88.00   Median :5200   Median :26.00   Median :32.00   Median : 9164  
 Mean   : 95.88   Mean   :5116   Mean   :26.51   Mean   :32.07   Mean   :11428  
 3rd Qu.:114.00   3rd Qu.:5500   3rd Qu.:31.00   3rd Qu.:37.00   3rd Qu.:14559  
 Max.   :200.00   Max.   :6600   Max.   :49.00   Max.   :54.00   Max.   :35056  
> 
```

Figure 1.3 – Summary of the dataset after completion of preprocessing.

At this point, I was comfortable with the structure and content of the dataset and advanced to the transformation phase of splitting the original dataset into a training and test data subsets.

Training and test data. The first step in creating the subsets was establishing a seed. The set.seed command was used to ensure the results were reproducible. More specifically, the created seed ensures that the data would get divided similarly every time (Bansal, 2020).

In this model, the training set was set to hold 70% of the observations, while the test data, the remaining 30% . The main reason we split the dataset between training and data is for evaluating the performance and accuracy of the generated algorithm. This technique is vital when handling large data sets, or when developing a time-intensive model, and/or accuracy or specific threshold is expected of the model performance (Brownlee, 2020). Below (see figure 1.4A, & figure 1.4B,) are the used commands to set the seed and divide the data into the training and test subsets, followed by a quick glimpse of each subset by the head( ) command.

```
# Set the seed value to ensure that result were reproducible
set.seed(1234)
> # Divide the data into training and test data
```

```
> smpl <- sample(2, nrow(import), replace = TRUE, prob = c(0.7, 0.3))
> train.data <- import [smpl == 1, ]     #Training sample
> head(train.data)
```

```
> train.data <- import [smpl == 1, ]     #Training sample
> head(train.data)
   symboling normalized_losses fuel_type aspiration num_doors body_style drive_wheels engine_location wheel_base length width
4          2               164       gas        std      four      sedan          fwd           front       99.8  176.6  66.2
5          2               164       gas        std      four      sedan          4wd           front       99.4  176.6  66.4
7          1               158       gas        std      four      sedan          fwd           front      105.8  192.7  71.4
9          1               158       gas      turbo      four      sedan          fwd           front      105.8  192.7  71.4
12         0               192       gas        std      four      sedan          rwd           front      101.2  176.8  64.8
13         0               188       gas        std       two      sedan          rwd           front      101.2  176.8  64.8
   height curb_weight engine_size bore stroke compression_ratio horsepower peak_rpm city_mpg highway_mpg Price
4    54.3        2337         109 3.19   3.40              10.0        102     5500       24          30 13950
5    54.3        2824         136 3.19   3.40               8.0        115     5500       18          22 17450
7    55.7        2844         136 3.19   3.40               8.5        110     5500       19          25 17710
9    55.9        3086         131 3.13   3.40               8.3        140     5500       17          20 23875
12   54.3        2395         108 3.50   2.80               8.8        101     5800       23          29 16925
13   54.3        2710         164 3.31   3.19               9.0        121     4250       21          28 20970
>
```

Figure 1.4A – Heading of training dataset.

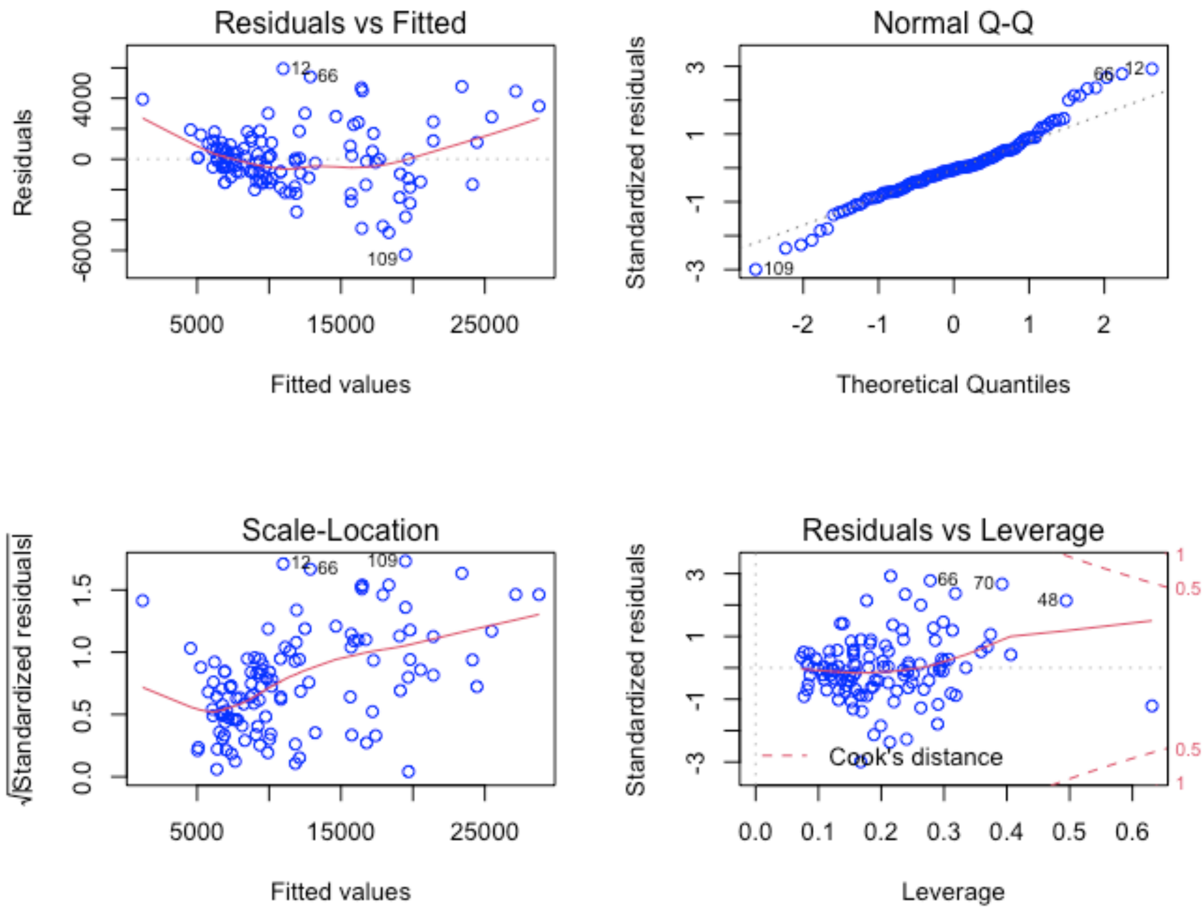
```
> test.data <- import [smpl == 2, ]      #Test sample
> head(test.data)
```

```
> test.data <- imp [ind == 2, ]
> head(test.data)
   symboling normalized_losses fuel_type aspiration num_doors body_style drive_wheels wheel_base length width height curb_weight
11         E               192       gas        std       two      sedan          rwd      101.2  176.8  64.8   54.3        2395
24         D               118       gas      turbo       two  hatchback          fwd       93.7  157.3  63.8   50.8        2128
26         D               148       gas        std      four      sedan          fwd       93.7  157.3  63.8   50.6        1989
36         C               110       gas        std      four      sedan          fwd       96.5  163.4  64.0   54.5        2010
38         C               106       gas        std       two  hatchback          fwd       96.5  167.5  65.2   53.3        2236
39         C               106       gas        std       two  hatchback          fwd       96.5  167.5  65.2   53.3        2289
   engine_size     bore      stroke compression_ratio horsepower peak_rpm city_mpg highway_mpg Price
11         108 (3,3.5] (2.49,2.91]               8.8        101     5800       23          29 16430
24          98 (3,3.5] (3.33,3.75]               7.6        102     5500       24          30  7957
26          90 (2.5,3] (2.91,3.33]               9.4         68     5500       31          38  6692
36          92 (2.5,3] (3.33,3.75]               9.2         76     6000       30          34  7295
38         110 (3,3.5] (3.33,3.75]               9.0         86     5800       27          33  7895
39         110 (3,3.5] (3.33,3.75]               9.0         86     5800       27          33  9095
>
```

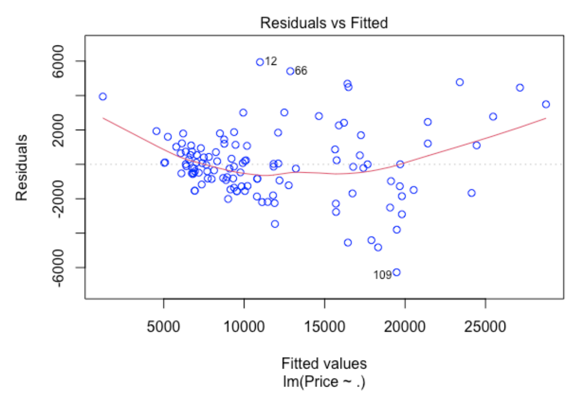
Figure 1.4B – Heading of testing dataset .

After splitting the data, I continued to create the model. I used the lm( ) command to fit the regression model with the training data. Here, price was set as the dependable variable and the rest of the variables – using the any-character wildcard (i.e. Price ~ . , …)—as predictors.

```
# Build the model w/ lm command.
> model<-lm(Price~., train.data)
> plot(model, col = "blue")
```



The plot ( ) command generated four unique plots of the model. The first one displayed the model's residuals vs. fitted values. As observed, the model reflects positive residuals (above the dotted line), and negative residuals (below the dotted line). The curved red line highlights the pattern followed by the error residuals.



The second graphic shows a normal q-q plot displaying the distribution of the residuals against the quartiles. Per this graphic we can argue that the majority of the residuals are normally distributed as they run close to the dotted line. Some exceptions, observations 66, 109, and 120.

Next, the scale-location plot. This plot indicates how spread were the points along the predicted range. Per Shantanu Deo, one of the assumptions for regression is that the points' variance should be within a reasonable range of the predictor (2016). In this model, we may say that between 5000 and 20000 the variance shows a good degree of uniformity.

Finally, the residuals and leverage plot. This graphic displays how much leverage, and therefore, influence a point may exerts onto the regression model's variable of price, should a given observation be removed. In terms of the Cook's distance red dotted-line reference, any point falling above or below this line are considered of high leverage. In this case, no point was found within the mentioned regions.

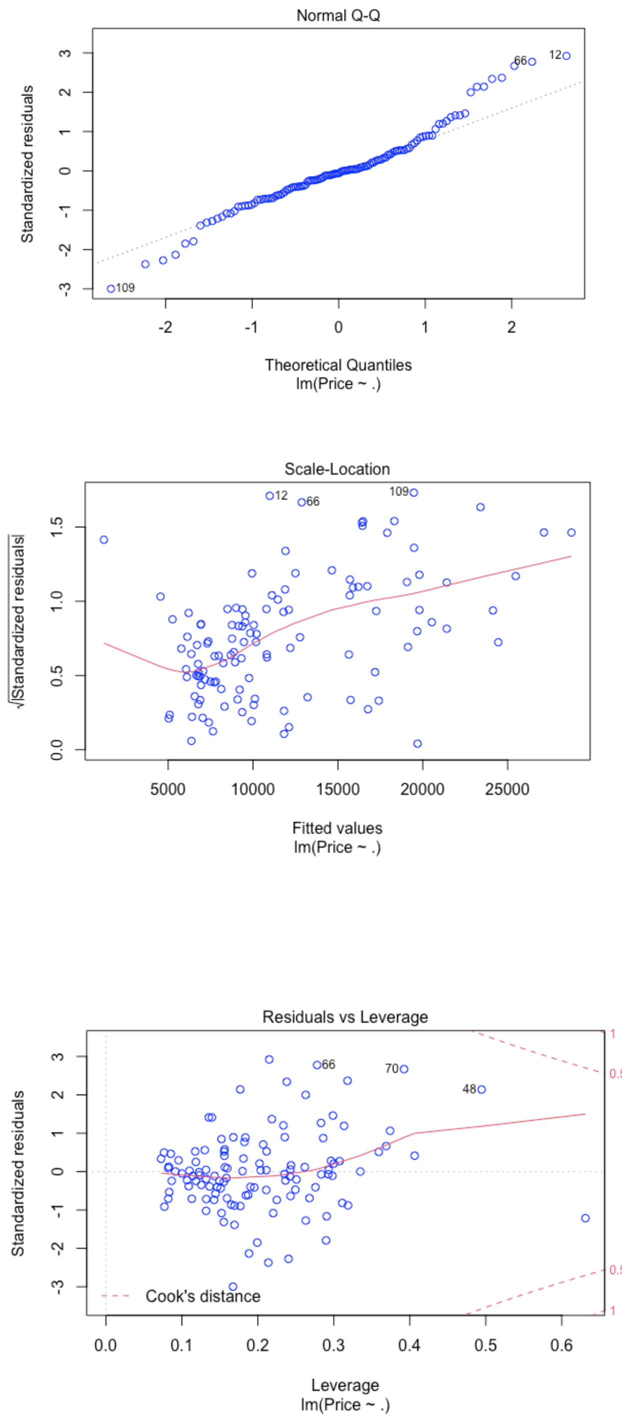


Figure 1.5 – Model's residual plots .

The next step was reviewing the model's descriptive statistics using the summary( ) command. Displayed below, this command produced particulars of the model including the formula used, residuals information, and coefficients.

```
> summary(model)      # Run summary( ) to see descriptive statistics
Call:
lm(formula = Price ~ ., data = train.data)

Residuals:
    Min      1Q  Median      3Q     Max
-6273.9 -1238.0  -106.9  1045.8  5942.4

Coefficients:
                      Estimate Std. Error t value Pr(>|t|)
(Intercept)         -6.447e+04  2.177e+04  -2.962  0.00387 **
fuel_typegas         1.773e+04  8.003e+03   2.215  0.02919 *
aspirationturbo      3.180e+03  1.162e+03   2.736  0.00744 **
wheel_base           2.279e+02  1.338e+02   1.704  0.09176 .
curb_weight          5.963e+00  3.115e+00   1.914  0.05861 .
bore                -3.647e+03  1.320e+03  -2.764  0.00688 **
compression_ratio    1.292e+03  5.847e+02   2.210  0.02950 *
---
... "See appendix for complete summary( ) results"

Residual standard error: 2294 on 94 degrees of freedom
Multiple R-squared:  0.8791,     Adjusted R-squared:  0.8482
F-statistic: 28.48 on 24 and 94 DF,  p-value: < 2.2e-16
```

Observations about the summary output

How does the model represent the relationships between dependent and independent variables in the auto import dataset?

The model represents the relationship between the predictors and the dependent variables. Each coefficients' estimated values underline the relationship between such variable and the targeted variable. The residuals range gives an example of how normally distributed the residuals are, important aspect of regression. Other values such as the adjusted r-squared, p-value and F-statistics give additional parameters to confirm the veracity of the model.

How does the method handle categorical variables?

Categorical variables are handled differently. Case in point, across the df the aspiration variable only shows to options, std or turbo. The model evaluates one of the two, in this case aspiration_turbo. Basically, when the value is zero this coefficient does not influence the total Price, but when it is equal to one (meaning the car has turbo), the price goes up by $3,180

What does the residuals section of the output mean?

The residuals section refers to the observed values of the error term for each of the given observations. In simple terms, residuals are the opposite side of the predictions, the distances between the observed and predicted values. In this model, the residuals' variation runs from -6273 to 5942 with a median around -100. As previously mentioned, these noted range and median illustrates the models' quasi-normal distribution and constant variance (Dietrich, Heller, & Yang, 2015).

What are coefficients, and what do they mean?

The coefficient is an estimated calculation for each variable, based on the ordinary least squares (OLS), which aims to minimize the error delta between the linear model and the actual observations, in order to estimate and trace the fit line of the model that best approximates the relationship between the outcome variable (Price) and the independent variables (Dietrich, Heller, & Yang, 2015). Consider the curb_weight value of 5.963. It means that for every single unit increase in a car's curb_weight the total price increases by $5,963.

What is an intercept, and what does it mean?

The intercept refers to the value of y when x = 0. When everything still at zero, the point where the line touches the vertical or y axis, denoted in the below multiple linear regression equation as β_0 (Triola, 2017).

$$\gamma = \boldsymbol{\beta_0} + \beta_1\chi_1 + \beta_2\chi_2 + \beta_3\chi_3 + \cdots + \beta_{p-1}\chi_{p-1} + \varepsilon$$

What do the p-values tell us about the significance of each variable?

The p-values indicates how statistically significant a variable is in relation to either supporting or rejecting a null hypothesis. Normally a p-value less than 0.05 or 5% will be considered as a strong evidence to reject the null hypothesis, and support the other position, in this case the Price as the response.

What is the overall accuracy of the model?

The accuracy of the model can be measured in terms of the model's residual standard error (RSE). In other words, the RSE quantifies that threshold between the observations and the predicted regression line, thus describing the accuracy of the model. The lower the number the better accuracy the model has. For this model the RSE is 2294 on 94 degrees of freedom.

Evaluate the model on a test set. Once completed with the training data model, I moved to test the model with the 30% train data set. For this step, I used the predict( ) command which take in consideration the test.data argument and evaluates these new set of variables against the model created with the lm( ) command. Statistically, the prediction model displayed a minimum value of 5255, a max of 26997 and a median around 8850.

```
# Evaluate the model on test data
pred <- predict(model, newdata=test.data)
> summary(pred)
   Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
   5255    7432    8858   10585   12742   26997
```

Then I moved to display the results of predicted vs. observed values using the plot( ) command. Here I use the vectored prediction variable of pred to assess the test.data subset. Once generated, I added a fit line to compare actual vs. predicted values.

```
# Evaluate the model on test data
pred <- predict(model, newdata=test.data)
# Test data scatter plot
plot(test.data$Price, pred, xlab = "Observed", ylab = "Prediction",
     main = "Model Evaluation on Test Data", col = "dark red")
abline(a = 0, b = 1, col = "blue")
```

Importantly, this visual representation of the test data, confirms some of the details mentioned about the underpinning coefficients of low p-value and coefficient of determination of 84%. Per the graphic, the majority of the observations were along the prediction line, confirming that the residuals had a good distribution and minimal variance.

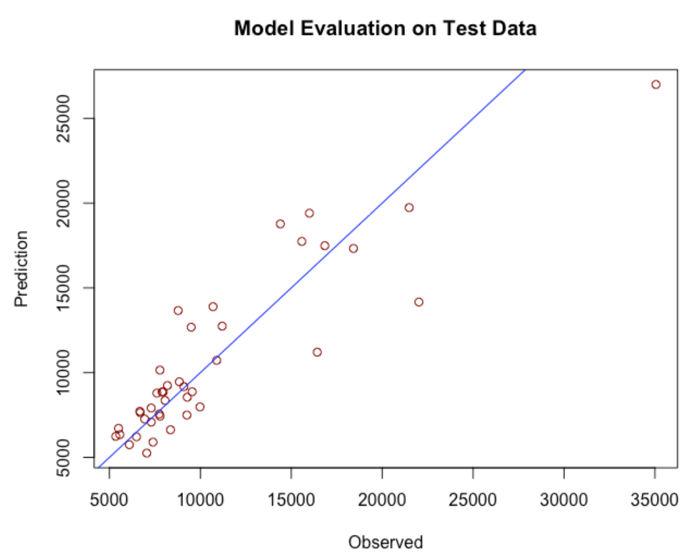


Figure 1.6 – Test set conforms to the model's prediction.

The close distance between the points and the fit line, measured by the ordinary least squares, talks about the small overall error and useflness of the developed model.

As previously mentioned on the residuals plot, I found the scale-location plot highly informative. The plot illustrates the residuals' variance along the prediction line. In addition, I created a histogram (figure 1.7) to better appreciate the residuals distribution, which once again confirmed the aforementioned values of an almost symmetrical distribution.

```
> model<-lm(Price~., train.data)
> par(mfrow = c(1,1))
> plot(model, col = "blue")
Hit <Return> to see next plot:
# Additional histogram to show residuals distribution
> hist(model$residual, col = blues9)
```

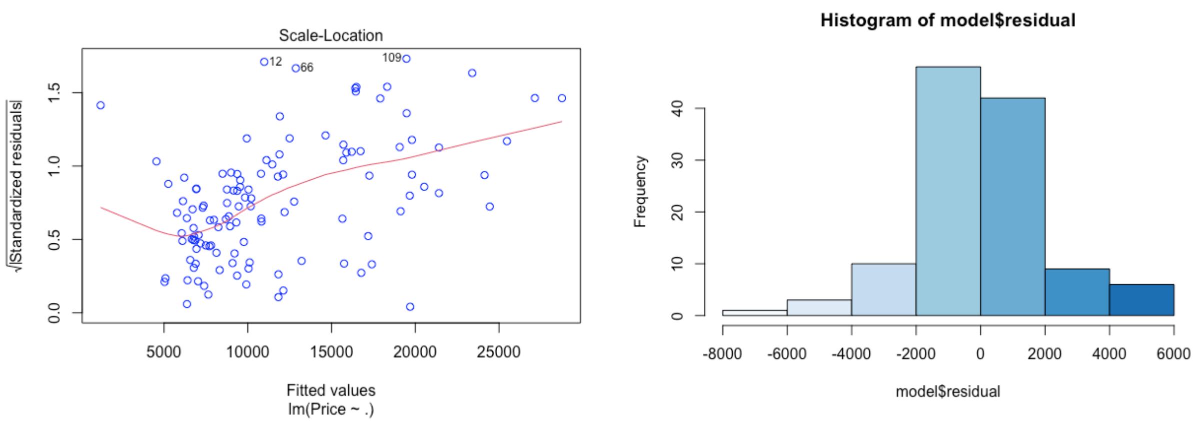


Figure 1.7 – Residuals distribution well-balanced.

Minimal adequate model. The minimal adequate model (MAM) involves the minimum set of variables or predictors needed to sustain the model at hand. This is a minimalist approach that follows the principle of parsimony (aka Occam's razor) which states that a model should be as simple as possible (Bruce, P & Bruce, A. 2017). This approach can result very useful when processing big datasets with numerous attributes and observations, by concentrating performance and processing efforts into particular variables with strong leverage.

In building a reduced model I used the step( ) command with a backward direction. This command generated many different steps, each one with different AIC values. The Akaike's Information Criteria (AIC) is a metric that measures the level of observations and variables used for the model. Following the Occam's razor principle, the goal is to identify a model with the lowest AIC score (Bruce, P & Bruce, A. 2017). The below command output shows the step with the lowest AIC equal to 1839.57, and only using the normalized_losses, fuel_type, aspiration, wheel_base, curb_weight, engine_size, bore, and compression_ratio.

```
# Use the step function to build a reduced model
model2<-step(model, direction="backward")
# Preview model2 and identify lowest-value AIC
Step:  AIC=1839.57
Price ~ normalized_losses + fuel_type + aspiration + wheel_base +
    curb_weight + engine_size + bore + compression_ratio
                    Df Sum of Sq       RSS    AIC
<none>                           528983234 1839.6
- normalized_losses  1  15431693 544414927 1841.0
- engine_size        1  19872679 548855913 1842.0
- wheel_base         1  30890707 559873941 1844.3
- bore               1  45531578 574514812 1847.4
- compression_ratio  1  55921299 584904533 1849.5
- fuel_type          1  57462282 586445516 1849.8
- aspiration         1  83448857 612432091 1855.0
- curb_weight        1 108234656 637217890 1859.7
```

To further analyze this step, I used the summary ( ) command for this second model.

```
> summary(model2)
```

```
> summary(model2)

Call:
lm(formula = Price ~ normalized_losses + fuel_type + aspiration +
    wheel_base + curb_weight + engine_size + bore + compression_rati
    data = train.data)

Residuals:
    Min      1Q  Median      3Q     Max
-6447.9 -1090.4   -51.5   920.3  6502.8

Coefficients:
                   Estimate Std. Error t value Pr(>|t|)
(Intercept)      -57061.666  12507.494  -4.562 1.32e-05 ***
normalized_losses    11.719      6.542   1.791 0.075986 .
fuel_typegas      20628.663   5967.663   3.457 0.000778 ***
aspirationturbo    3380.538    811.522   4.166 6.20e-05 ***
wheel_base          201.644     79.560   2.534 0.012668 *
curb_weight           7.648      1.612   4.744 6.33e-06 ***
engine_size          36.753     18.079   2.033 0.044477 *
bore              -3137.874   1019.774  -3.077 0.002639 **
compression_ratio  1488.471    436.492   3.410 0.000909 ***
---
Signif. codes:  0 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 0.1 ‘ ’ 1

Residual standard error: 2193 on 110 degrees of freedom
Multiple R-squared:  0.8707,    Adjusted R-squared:  0.8613
F-statistic:  92.6 on 8 and 110 DF,  p-value: < 2.2e-16
```

```
> model2
```

```
Call:
lm(formula = Price ~ normalized_losses + fuel_type + aspiration +
    wheel_base + curb_weight + engine_size + bore + compression_ratio,
    data = train.data)

Coefficients:
      (Intercept)  normalized_losses       fuel_typegas    aspirationturbo         wheel_base        curb_weight
       -57061.666             11.719          20628.663           3380.538            201.644              7.648
      engine_size               bore  compression_ratio
           36.753          -3137.874           1488.471
```

What are the coefficients and the intercept, and what do they mean?

For the second model, the only displayed coefficients were the 8 identified ones during the step( ) command and criteria of minimum AIC.

Compare the prediction accuracy of the minimum adequate model with the prediction accuracy of the original model.

First, to compare both models, the original one and the MAM, I used the AIC( ) command as illustrated below. The second model, had the lowest AIC across 10 degrees of freedom, while the original training dataset, which included all the independent variables was scored higher than the MAM. This number confirms the fact that more variables does not necessarily means better accuracy of a model.

```
> AIC(model, model2) # Use AIC function to evaluate both models
       df      AIC
model  26 2203.309
model2 10 2179.281
```

Secondly, I compared both models statistic values as shown below. Notice the lower residual standard error (RSE), which relates to the accuracy of the models, between model 2 and model 1. Additionally, the adjusted r-squared, highlights the ratio of included observations part of the model increases from 84% to 86%. Lastly, the F-statistic values also increases from 28.4 to 92.6.

```
Residual standard error: 2294 on 94 degrees of freedom
Multiple R-squared:  0.8791, Adjusted R-squared:  0.8482     # Model 1 RSE
F-statistic: 28.48 on 24 and 94 DF,  p-value: < 2.2e-16

Residual standard error: 2193 on 110 degrees of freedom
Multiple R-squared:  0.8707, Adjusted R-squared:  0.8613    # Model 2 RSE
F-statistic:  92.6 on 8 and 110 DF,  p-value: < 2.2e-16
```

Suppose that we have a new car, and we know the values for the independent variables. How would you use the model to predict the value of the dependent variable for the new car?

The model gives you a baseline to anchor your deductions. If we have the value of those 8 independent variables identified as part of the step with the lowest AIC, we will be able to estimate the total price of the vehicle by substituting or comparing the new parameters against the model, all other factors remaining equal.

In conclusion, this exercised aimed to explain how a multi-linear regression model could result beneficial predicting or anticipating continues values based on the correlation between a targeted variable and its correlated predictors. The exercise provides the opportunity to utilize different commands with R to assess a dataset, build a regression model, and tested against a subset of the original dataset. As it relates to the auto dataset, in fact the model was able to predict, with a high level of confidence, the anticipated values in price. Along the assessment, it is noticed that the accuracy of the model could be impacted by the residuals' distribution of the tested population or sampled data. In other words, the model may not be as accurate assessing asymmetrical distributions.

References

Bansal, J. (2020). How to Use Random Seeds Effectively. https://towardsdatascience.com/how-to-use-random-seeds-effectively-54a4cd855a79

Brownlee, J. (2020). Train-test Split for Evaluating Machine Learning Algorithms. https://machinelearningmastery.com/train-test-split-for-evaluating-machine-learning-algorithms/

Deo, S. (2016). R Tutorial : Residual Analysis for Regression. http://analyticspro.org/2016/03/05/r-tutorial-residual-analysis-for-regression/

Dietrich, D., Heller, B., & B. Yang. (2015). Data science & big data analytics: Discovering, analyzing, visualizing and presenting data. John Wiley & Sons, Inc.

Dua, D. and Graff, C. (2019). UCI Machine Learning Repository [http://archive.ics.uci.edu/ml]. Irvine, CA: University of California, School of Information and Computer Science.

Bruce, B., & Bruce, A. (2017). Practical statistics for data science. O'Reilly Media, Inc.

Appendix

```
> summary(model)

Call:
lm(formula = Price ~ ., data = train.data)

Residuals:
    Min      1Q  Median      3Q     Max 
-4766.4 -1244.5  -130.1   774.3  5443.3 

Coefficients:
                      Estimate Std. Error t value Pr(>|t|)    
(Intercept)         -8.553e+04  2.195e+04  -3.896  0.00019 ***
symboling           -2.185e+02  3.514e+02  -0.622  0.53566    
normalized_losses    1.546e+01  9.721e+00   1.590  0.11534    
fuel_typegas         1.068e+04  8.225e+03   1.299  0.19738    
aspirationturbo      3.175e+03  1.282e+03   2.476  0.01519 *  
num_doorstwo         2.270e+02  8.070e+02   0.281  0.77916    
body_stylehatchback  6.605e+02  1.276e+03   0.517  0.60611    
body_stylesedan      1.098e+03  1.366e+03   0.803  0.42391    
body_stylewagon      1.284e+03  1.617e+03   0.794  0.42914    
drive_wheelsfwd     -6.753e+02  1.287e+03  -0.525  0.60107    
drive_wheelsrwd      1.050e+03  1.558e+03   0.674  0.50231    
wheel_base           1.068e+02  1.528e+02   0.699  0.48643    
length              -5.721e+01  6.571e+01  -0.871  0.38625    
width                8.691e+02  2.837e+02   3.064  0.00290 ** 
height               2.339e+02  1.956e+02   1.196  0.23494    
curb_weight          1.193e+00  2.228e+00   0.535  0.59366    
engine_size          6.308e+01  3.213e+01   1.963  0.05277 .  
bore                -1.377e+03  1.291e+03  -1.067  0.28902    
stroke              -1.478e+03  9.386e+02  -1.575  0.11892    
compression_ratio    9.262e+02  5.905e+02   1.568  0.12040    
horsepower           9.403e+00  2.910e+01   0.323  0.74739    
peak_rpm             7.211e-01  7.863e-01   0.917  0.36156    
city_mpg            -3.511e+02  2.086e+02  -1.683  0.09589 .  
highway_mpg          2.505e+02  1.903e+02   1.317  0.19137    
---
Signif. codes:  0 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 0.1 ‘ ’ 1

Residual standard error: 2282 on 88 degrees of freedom
Multiple R-squared:  0.8664,	Adjusted R-squared:  0.8314 
F-statistic:  24.8 on 23 and 88 DF,  p-value: < 2.2e-16
```

```
> coef(model)         # To display the model coefficients only
        (Intercept)           symboling   normalized_losses        fuel_typegas     aspirationturbo        num_doorstwo 
      -8.553431e+04       -2.185308e+02        1.546071e+01        1.068288e+04        3.174741e+03        2.269953e+02 
body_stylehatchback     body_stylesedan     body_stylewagon     drive_wheelsfwd     drive_wheelsrwd          wheel_base 
       6.605023e+02        1.097810e+03        1.284263e+03       -6.753284e+02        1.049831e+03        1.067810e+02 
             length               width              height         curb_weight         engine_size                bore 
      -5.721460e+01        8.690945e+02        2.338774e+02        1.192926e+00        6.308358e+01       -1.376596e+03 
             stroke   compression_ratio          horsepower            peak_rpm            city_mpg         highway_mpg 
      -1.477951e+03        9.261566e+02        9.402638e+00        7.211378e-01       -3.510797e+02        2.505175e+02 
```

```
> model$residuals      # Output the residuals
          4           5           7           9          11          12          13          14          20          22 
 1970.09796  4007.19252  -458.78623  2882.41221  3481.47928  3766.41304  4345.49435  4641.87876  1010.34770   615.48193 
         24          25          27          28          29          30          31          32          33          34 
-1112.16627  -371.78886   544.65866 -1213.67916 -2329.68476  -118.70446  3022.66683   296.91213  -349.04006  -141.46206 
         36          38          39          41          43          60          61          62          63          66 
   -3.63507 -1049.92001    86.85493  3913.65265   822.99533  -991.97835 -2005.17442   758.02165  -255.17442  3393.36884 
         69          70          71          77          78          79          81          86          88          90 
 2512.67317  5067.24967  5443.27784  -702.26288 -1288.20460  -879.78014  -796.32568 -1959.35907 -2279.02572 -1551.32416 
         91          92          93          94          95          97          98          99         101         102 
  747.66198  -435.91900    59.98205  1078.92006   174.71445   670.61550  1713.41203  1805.29245   246.61645 -3670.54003 
        104         105         107         108         112         113         116         117         119         121 
-3203.04427 -1557.63309 -1845.15616 -4766.35112 -2611.10823 -3868.27552  -101.96203 -2818.27552   559.17991  -464.55311 
        124         126         133         134         135         136         137         138         139         140 
-1759.33331  4162.80668  -473.36363    94.10868  -187.10254  3358.95437  1908.52476  2614.34965  -523.42598  1029.36644 
        142         143         144         145         146         148         149         151         153         155 
-1229.94050  -748.41575   579.33988   176.45678 -1192.14256   584.36713 -2400.17107   343.92832   722.32377  -169.26476 
        156         158         159         160         161         162         166         167         168         169 
 -422.07090   614.41153   640.95553  -336.33480   991.68425  1808.29666 -1494.09917  -858.54345 -3550.49905 -2355.72735 
        171         172         174         175         177         178         179         180         181         184 
 -966.31572 -1318.57045  -338.80642 -3060.30083  1305.09176  2273.78851 -2158.93538 -3117.73211 -1875.40616 -1844.85936 
        185         187         188         191         197         198         199         200         201         202 
-1865.28247  -743.69732 -2537.66474  1430.07232  -112.98476   465.77018   152.59192   725.38223 -1108.42318 -2384.21667 
        203         204 
  637.51348    74.31568 
```