DS6372 Project 1

Carl Keusseyan, John Olan, Feby Thomas

1/31/2021

MAIN OBJECTIVES

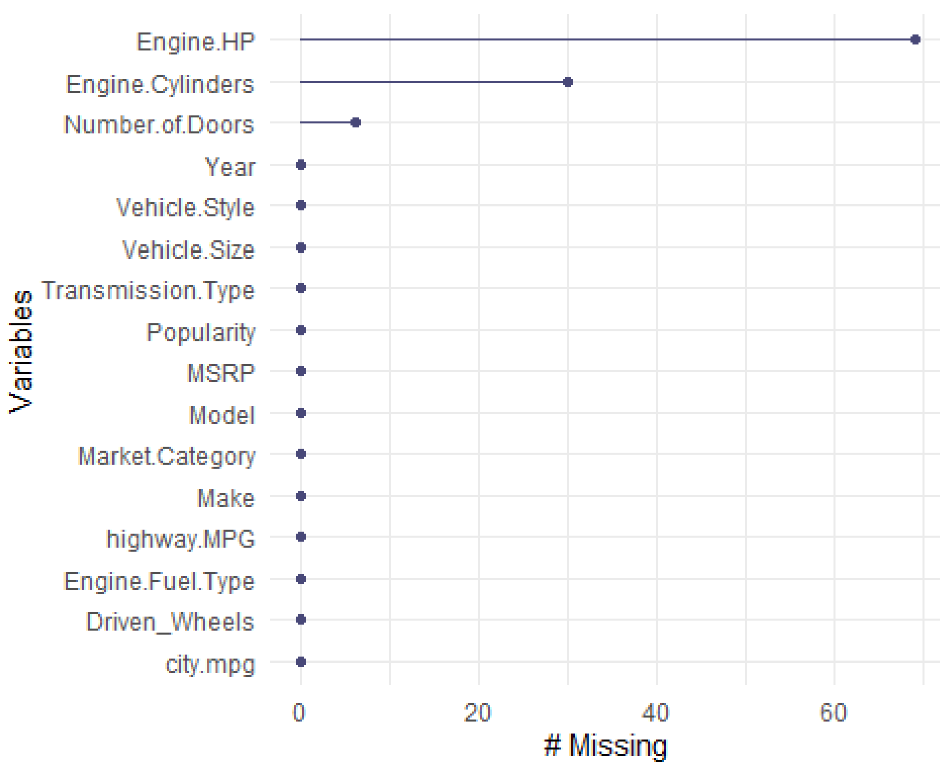
1. Read in the data
2. EDA
    a. Deal with missing data - analyze and document how we deal with it
    b. Examine the data - boxplots, Min, Max, look for outliers
    c. deal with columns that are Factors - turn them into levels
    d. Scatter plot matrix to see if there is a relationship between any 2 variables This gives us an idea on the model
3. Fit a preliminary linear model to the entire data set.
4. If residual plot shows non-constant variance, use weighted linear regression
5. Split the data into 80% train, 10% test and 10% validation First go for a simple model (no interaction terms or quadratics) using the train set, predict and compare to the test set, and then validate.
6. Fit the model using Single tree, prune as it may.
7. Fit to random forest. Grid search parameters.
8. Fit using lasso regularization. Grid search parameters.
9. Tabulate all the Test RMSEs to see which is lowest.

The codes as adequately clearly commented upon.

EXPLORATORY DATA ANALYSES

- Deal with missing data - analyze and document how we deal with it

```r
#let us plot what variable are missing
gg_miss_var(car.new)
```



As we can see from the above figure. The Engine.HP (70) and Engine.Cylinders (30) have some missing values.

Using the View() we realized that The Market.Category contained a substantial amount of 'N/As' (3600 missing values) which were not captured by `gg_miss_var(car.new) function.`

This values in the Market Category were subjectively inputed and could not help with prediction.

- Examine the data - boxplots, Min, Max, look for outliers
- deal with columns that are Factors - turn them into levels
- Scatter plot matrix to see if there is a relationship between any 2 variables
  This gives us an idea on the model

KNITTED FILE FROM R-MARKDOWN

```
#set Working directory
setwd("C:/Users/olani/OneDrive/Documents/Data Science/SMU-Data Science/Applie
d Statistics/Project1Details_2021/Project1Details_2021")

#REad in the CSV file
car.new<-read.csv("data1.csv")
attach(car.new)

#examine the newly rad in dataset car.new
head(car.new)
```

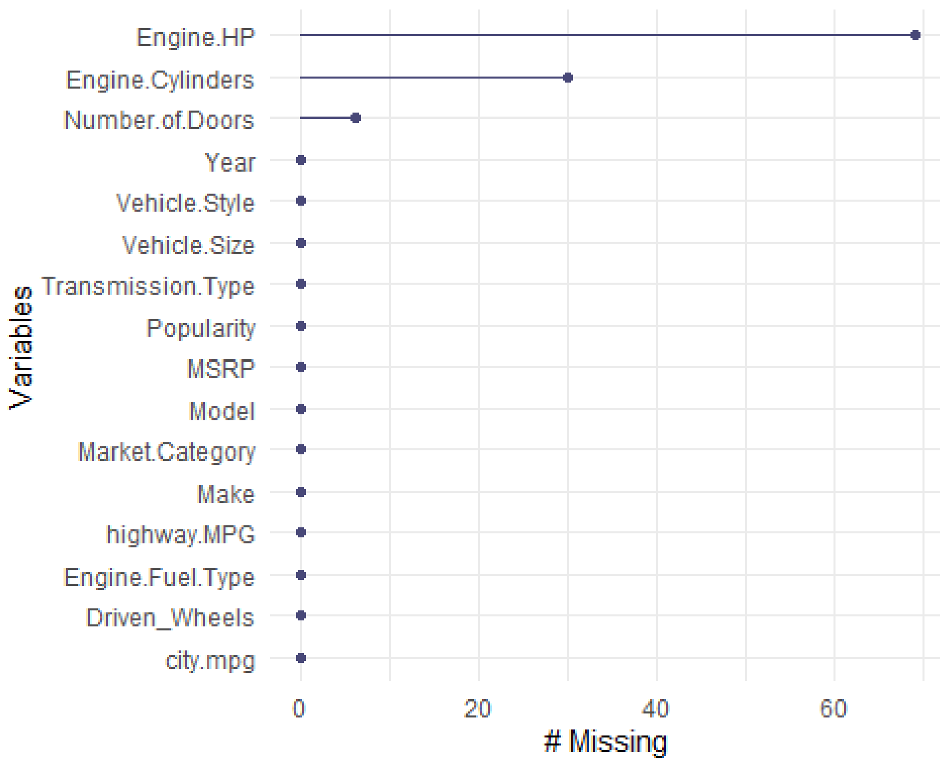
```
##   Make        Model Year              Engine.Fuel.Type Engine.HP Engine.Cylind
ers
## 1  BMW 1 Series M 2011 premium unleaded (required)       335
6
## 2  BMW   1 Series 2011 premium unleaded (required)       300
6
## 3  BMW   1 Series 2011 premium unleaded (required)       300
6
## 4  BMW   1 Series 2011 premium unleaded (required)       230
6
## 5  BMW   1 Series 2011 premium unleaded (required)       230
6
## 6  BMW   1 Series 2012 premium unleaded (required)       230
6
##   Transmission.Type    Driven_Wheels Number.of.Doors
## 1            MANUAL rear wheel drive               2
## 2            MANUAL rear wheel drive               2
## 3            MANUAL rear wheel drive               2
## 4            MANUAL rear wheel drive               2
## 5            MANUAL rear wheel drive               2
## 6            MANUAL rear wheel drive               2
##                         Market.Category Vehicle.Size Vehicle.Style highway
.MPG
## 1 Factory Tuner,Luxury,High-Performance      Compact         Coupe
26
## 2                    Luxury,Performance      Compact   Convertible
28
## 3               Luxury,High-Performance      Compact         Coupe
28
## 4                    Luxury,Performance      Compact         Coupe
28
## 5                                Luxury      Compact   Convertible
28
## 6                    Luxury,Performance      Compact         Coupe
28
##   city.mpg Popularity  MSRP
```

```
## 1       19       3916 46135
## 2       19       3916 40650
## 3       20       3916 36350
## 4       18       3916 29450
## 5       18       3916 34500
## 6       18       3916 31200
```

```r
str(car.new)
```

```
## 'data.frame':    11914 obs. of  16 variables:
##  $ Make             : chr  "BMW" "BMW" "BMW" "BMW" ...
##  $ Model            : chr  "1 Series M" "1 Series" "1 Series" "1 Series" .
..
##  $ Year             : int  2011 2011 2011 2011 2011 2012 2012 2012 2012 20
13 ...
##  $ Engine.Fuel.Type : chr  "premium unleaded (required)" "premium unleaded
(required)" "premium unleaded (required)" "premium unleaded (required)" ...
##  $ Engine.HP        : int  335 300 300 230 230 230 300 300 230 230 ...
##  $ Engine.Cylinders : int  6 6 6 6 6 6 6 6 6 6 ...
##  $ Transmission.Type: chr  "MANUAL" "MANUAL" "MANUAL" "MANUAL" ...
##  $ Driven_Wheels    : chr  "rear wheel drive" "rear wheel drive" "rear whe
el drive" "rear wheel drive" ...
##  $ Number.of.Doors  : int  2 2 2 2 2 2 2 2 2 2 ...
##  $ Market.Category  : chr  "Factory Tuner,Luxury,High-Performance" "Luxury
,Performance" "Luxury,High-Performance" "Luxury,Performance" ...
##  $ Vehicle.Size     : chr  "Compact" "Compact" "Compact" "Compact" ...
##  $ Vehicle.Style    : chr  "Coupe" "Convertible" "Coupe" "Coupe" ...
##  $ highway.MPG      : int  26 28 28 28 28 28 26 28 28 27 ...
##  $ city.mpg         : int  19 19 20 18 18 18 17 20 18 18 ...
##  $ Popularity       : int  3916 3916 3916 3916 3916 3916 3916 3916 3916 39
16 ...
##  $ MSRP             : int  46135 40650 36350 29450 34500 31200 44100 39300
36900 37200 ...
```

```r
### Deal with Missing Data

#let us plot what variable are missing
gg_miss_var(car.new)
```




```
# we can see the missing data
#     Engine HP around about 70 values
#     Engine Cylinders about 30 values
#      Number of doors about 5

# Display the number of rows anc columns
dim(car.new)  # we know we have 11914 rows of data with 16 columns

## [1] 11914    16

# removing the Market.category column because it has too many N/As  - 3600 ro
ws are missing this value, and this amount of lack of data will compromise ou
r analysis and the model.  At this point we are going to take it out.

car.new <- subset(car.new, select =-c(Market.Category))

str(car.new)

## 'data.frame':    11914 obs. of  15 variables:
##  $ Make             : chr  "BMW" "BMW" "BMW" "BMW" ...
##  $ Model            : chr  "1 Series M" "1 Series" "1 Series" "1 Series" .
..
##  $ Year             : int  2011 2011 2011 2011 2011 2012 2012 2012 2012 20
13 ...
##  $ Engine.Fuel.Type : chr  "premium unleaded (required)" "premium unleaded
(required)" "premium unleaded (required)" "premium unleaded (required)" ...
##  $ Engine.HP        : int  335 300 300 230 230 230 300 300 230 230 ...
```

```
##  $ Engine.Cylinders : int  6 6 6 6 6 6 6 6 6 6 ...
##  $ Transmission.Type: chr  "MANUAL" "MANUAL" "MANUAL" "MANUAL" ...
##  $ Driven_Wheels    : chr  "rear wheel drive" "rear wheel drive" "rear whe
el drive" "rear wheel drive" ...
##  $ Number.of.Doors  : int  2 2 2 2 2 2 2 2 2 2 ...
##  $ Vehicle.Size     : chr  "Compact" "Compact" "Compact" "Compact" ...
##  $ Vehicle.Style    : chr  "Coupe" "Convertible" "Coupe" "Coupe" ...
##  $ highway.MPG      : int  26 28 28 28 28 28 26 28 28 27 ...
##  $ city.mpg         : int  19 19 20 18 18 18 17 20 18 18 ...
##  $ Popularity       : int  3916 3916 3916 3916 3916 3916 3916 3916 3916 39
16 ...
##  $ MSRP             : int  46135 40650 36350 29450 34500 31200 44100 39300
36900 37200 ...
```

###Feature Engineering ####convert features to factors

```
##Convert relevant features to factors
car.new1 <- car.new %>%
    mutate(Make = as.factor(Make),
           Popularity = as.factor(Popularity),
           Model = as.factor(Model),
           Vehicle.Size = as.factor(Vehicle.Size),
           Vehicle.Style = as.factor(Vehicle.Style),
           Number.of.Doors = as.factor(Number.of.Doors),
           Driven_Wheels = as.factor(Driven_Wheels),
           Transmission.Type=as.factor(Transmission.Type),
           Engine.Cylinders = as.factor(Engine.Cylinders),
           Engine.Fuel.Type = as.factor(Engine.Fuel.Type))

str(car.new1)

## 'data.frame':    11914 obs. of  15 variables:
##  $ Make             : Factor w/ 48 levels "Acura","Alfa Romeo",..: 6 6 6 6
6 6 6 6 6 6 ...
##  $ Model            : Factor w/ 915 levels "1 Series","1 Series M",..: 2 1
1 1 1 1 1 1 1 1 ...
##  $ Year             : int  2011 2011 2011 2011 2011 2012 2012 2012 2012 20
13 ...
##  $ Engine.Fuel.Type : Factor w/ 11 levels "","diesel","electric",..: 10 10
10 10 10 10 10 10 10 10 ...
##  $ Engine.HP        : int  335 300 300 230 230 230 300 300 230 230 ...
##  $ Engine.Cylinders : Factor w/ 9 levels "0","3","4","5",..: 5 5 5 5 5 5 5
5 5 5 ...
##  $ Transmission.Type: Factor w/ 5 levels "AUTOMATED_MANUAL",..: 4 4 4 4 4
4 4 4 4 4 ...
##  $ Driven_Wheels    : Factor w/ 4 levels "all wheel drive",..: 4 4 4 4 4 4
4 4 4 4 ...
##  $ Number.of.Doors  : Factor w/ 3 levels "2","3","4": 1 1 1 1 1 1 1 1 1 1
...
##  $ Vehicle.Size     : Factor w/ 3 levels "Compact","Large",..: 1 1 1 1 1 1
```

```
1 1 1 1 ...
##  $ Vehicle.Style    : Factor w/ 16 levels "2dr Hatchback",..: 9 7 9 9 7 9
7 9 7 7 ...
##  $ highway.MPG      : int  26 28 28 28 28 28 26 28 28 27 ...
##  $ city.mpg         : int  19 19 20 18 18 18 17 20 18 18 ...
##  $ Popularity       : Factor w/ 48 levels "2","21","26",..: 47 47 47 47 47
47 47 47 47 47 ...
##  $ MSRP             : int  46135 40650 36350 29450 34500 31200 44100 39300
36900 37200 ...
```

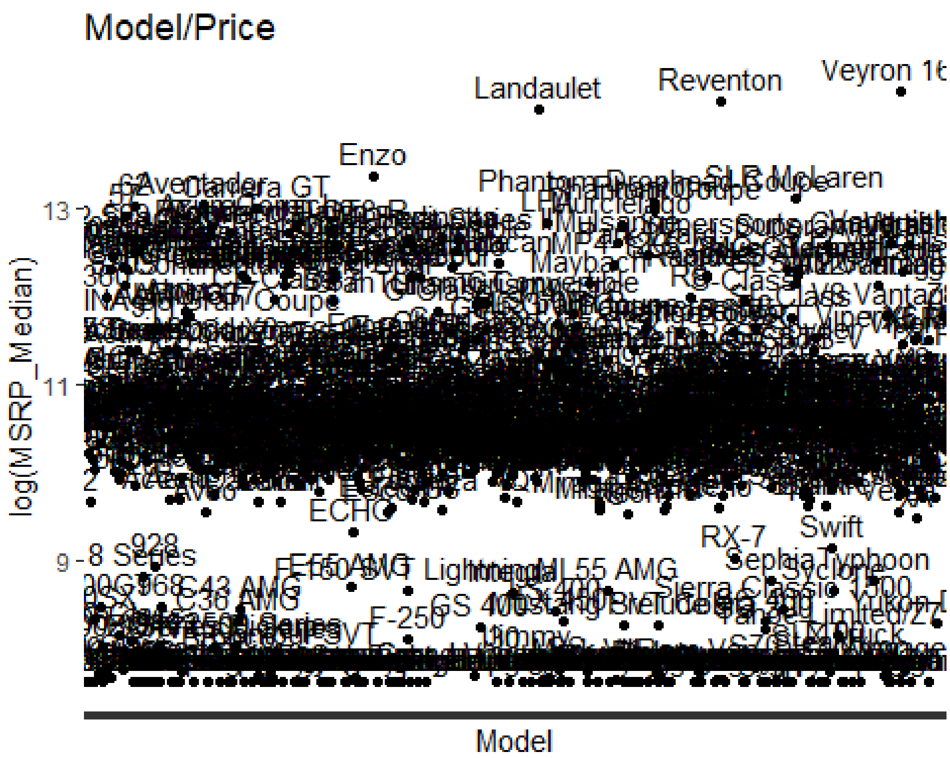
```
#list of level in Make. I can't merge the make to reduce the levels. They are
different.
make <- unique(car.new1$Make)
make
```

```
##  [1] BMW           Audi          FIAT          Mercedes-Benz Chrysler
##  [6] Nissan        Volvo         Mazda         Mitsubishi    Ferrari
## [11] Alfa Romeo    Toyota        McLaren       Maybach       Pontiac
## [16] Porsche       Saab          GMC           Hyundai       Plymouth
## [21] Honda         Oldsmobile    Suzuki        Ford          Cadillac
## [26] Kia           Bentley       Chevrolet     Dodge         Lamborghini
## [31] Lincoln       Subaru        Volkswagen    Spyker        Buick
## [36] Acura         Rolls-Royce   Maserati      Lexus         Aston Martin
## [41] Land Rover    Lotus         Infiniti      Scion         Genesis
## [46] HUMMER        Tesla         Bugatti
## 48 Levels: Acura Alfa Romeo Aston Martin Audi Bentley BMW Bugatti ... Volv
o
```

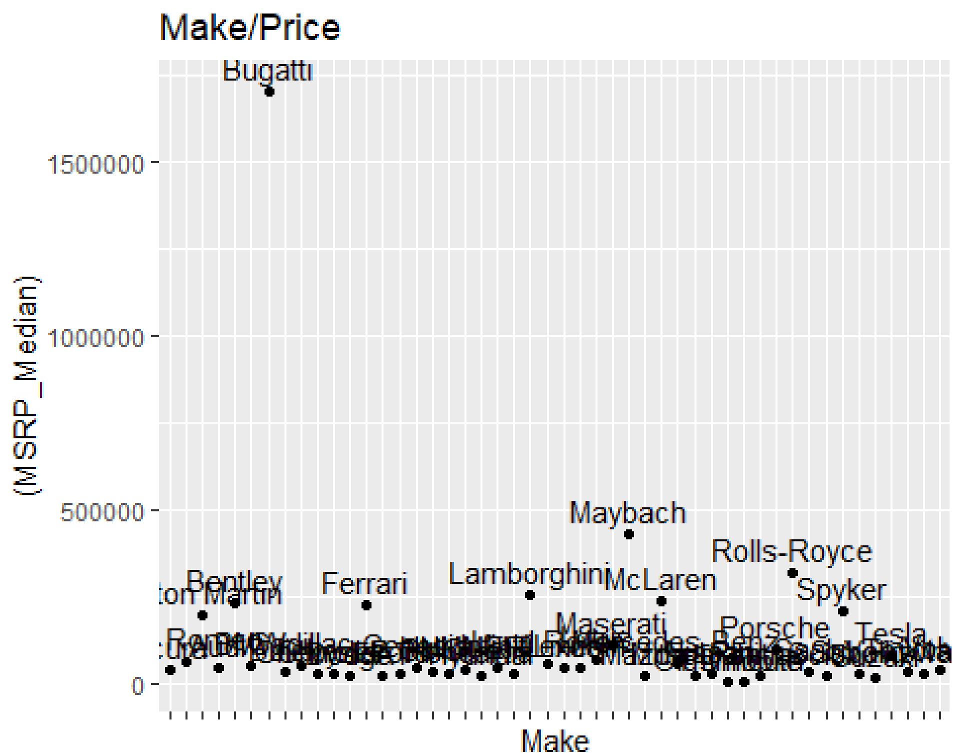
###EDA

```
#plot some viz
#Model versus MSRP. No trend transformed or not
##

car.new1 %>% group_by(Model) %>% summarise(MSRP_Median = median(MSRP)) %>%
  arrange(desc(MSRP_Median)) %>%
  ggplot(aes(x=Model, y = log(MSRP_Median))) +
  geom_point() +
  geom_text(aes(label=Model),hjust=0.5, vjust=-0.5) + #this line adds label t
o the datapoints so I can where the outliers come from
  theme(axis.text.x = element_blank()) +
  xlab("Model") +
  ggtitle("Model/Price")
```

```
## `summarise()` ungrouping output (override with `.groups` argument)
```




```
#Make and MSRP
#High leverage from the very expensive vehicles
#remove or keep @ MSRP> USD$100,000?

#The primary objective for this EDA section is to identify the varibles that
have association with MSRP (the response)
#The code below serves as template. So I just changed out the variables as ne
eded, to view their relationship.

car.new1 %>% group_by(Make) %>% summarise(MSRP_Median = median(MSRP)) %>%
  arrange(desc(MSRP_Median)) %>%
  ggplot(aes(x = Make, y = (MSRP_Median))) +
  geom_point() +
  geom_text(aes(label=Make),hjust=0.5, vjust=-0.5) + #this line adds label to
the datapoints so I can where the outliers come from
  theme(axis.text.x = element_blank()) +
  xlab("Make") +
  ggtitle("Make/Price")

## `summarise()` ungrouping output (override with `.groups` argument)
```



###Post-EDA 1

```
#I subset out the highly expensive cars that have high leverage and are 'outl
iers'.

car_nonEx <-  car.new %>%
    filter(MSRP< 100000)
```

###exclude missing values

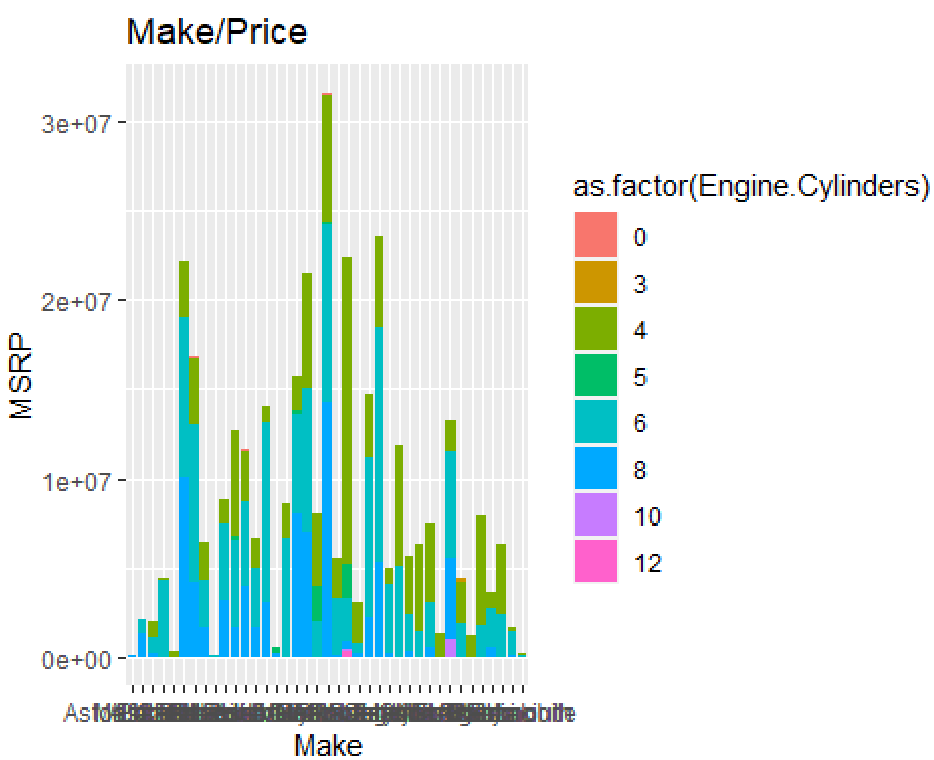
```
car_nonEx <- car_nonEx[complete.cases(car_nonEx),]  #exclude NA and missing d
ata
```

###Feature engineering and EDA 2

```
car_nonEx <- car_nonEx %>%
    mutate(Make = as.factor(Make),
           Popularity = as.factor(Popularity),
           Model = as.factor(Model),
           Vehicle.Size = as.factor(Vehicle.Size),
           Vehicle.Style = as.factor(Vehicle.Style),
           Number.of.Doors = as.factor(Number.of.Doors),
           Driven_Wheels = as.factor(Driven_Wheels),
           Transmission.Type=as.factor(Transmission.Type),
           Engine.Cylinders = as.factor(Engine.Cylinders),
           Engine.Fuel.Type = as.factor(Engine.Fuel.Type))
```

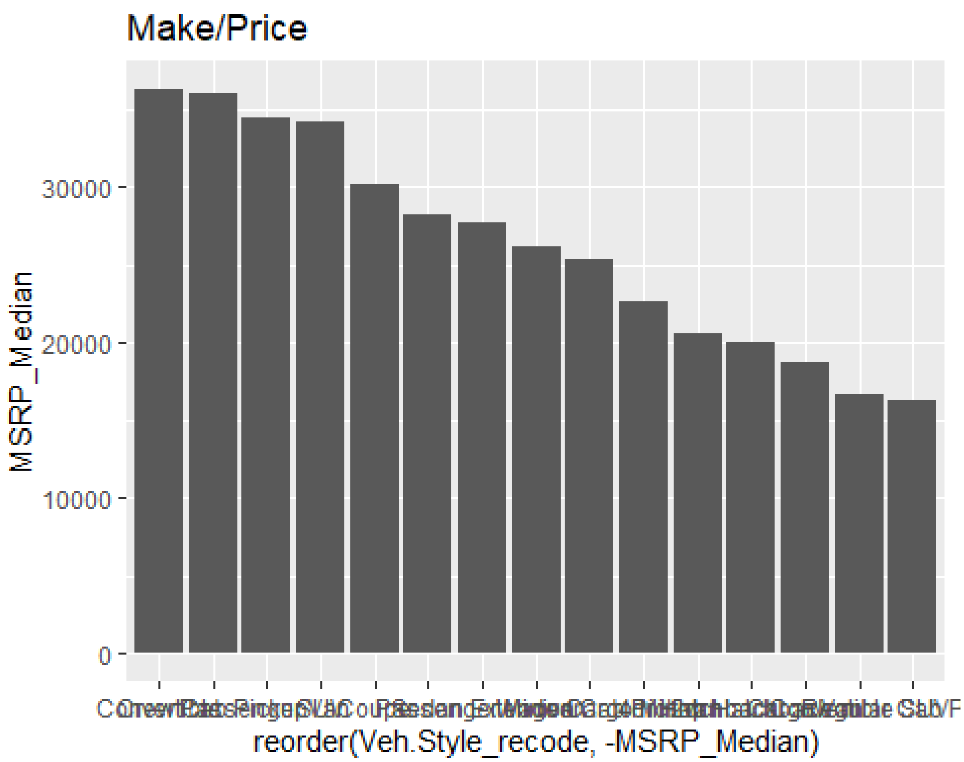
```
#Vehicle style has some levels that can be merged without losing info.
#Below is to replace "4Dr SUV" and "2Dr SUV" with "SUV", given that 4Dr and 2
Dr are already captured in Number.of.Doors column.

car_nonEx.new <- car_nonEx %>%
    mutate(Veh.Style_recode1 = Vehicle.Style)



car_nonEx.new$Veh.Style_recode = str_replace_all(as.character(car_nonEx.new$V
eh.Style_recode), "2dr SUV", "SUV")
car_nonEx.new$Veh.Style_recode = str_replace_all(as.character(car_nonEx.new$V
eh.Style_recode), "4dr SUV", "SUV")

car_nonEx.new$Veh.Style_recode = as.factor(car_nonEx.new$Veh.Style_recode)

#how does it look now.
str(car_nonEx.new)

## 'data.frame':    11182 obs. of  17 variables:
##  $ Make             : Factor w/ 39 levels "Acura","Alfa Romeo",..: 5 5 5 5
5 5 5 5 5 5 ...
##  $ Model            : Factor w/ 811 levels "1 Series","1 Series M",..: 2 1
1 1 1 1 1 1 1 1 ...
##  $ Year             : int  2011 2011 2011 2011 2011 2012 2012 2012 2012 20
13 ...
##  $ Engine.Fuel.Type : Factor w/ 10 levels "","diesel","electric",..: 9 9 9
9 9 9 9 9 9 9 ...
##  $ Engine.HP        : int  335 300 300 230 230 230 300 300 230 230 ...
##  $ Engine.Cylinders : Factor w/ 8 levels "0","3","4","5",..: 5 5 5 5 5 5 5
5 5 5 ...
##  $ Transmission.Type: Factor w/ 5 levels "AUTOMATED_MANUAL",..: 4 4 4 4 4
4 4 4 4 4 ...
##  $ Driven_Wheels    : Factor w/ 4 levels "all wheel drive",..: 4 4 4 4 4 4
4 4 4 4 ...
##  $ Number.of.Doors  : Factor w/ 3 levels "2","3","4": 1 1 1 1 1 1 1 1 1 1
...
##  $ Vehicle.Size     : Factor w/ 3 levels "Compact","Large",..: 1 1 1 1 1 1
1 1 1 1 ...
##  $ Vehicle.Style    : Factor w/ 16 levels "2dr Hatchback",..: 9 7 9 9 7 9
7 9 7 7 ...
##  $ highway.MPG      : int  26 28 28 28 28 28 26 28 28 27 ...
##  $ city.mpg         : int  19 19 20 18 18 18 17 20 18 18 ...
##  $ Popularity       : Factor w/ 39 levels "21","26","61",..: 38 38 38 38 3
8 38 38 38 38 38 ...
##  $ MSRP             : int  46135 40650 36350 29450 34500 31200 44100 39300
36900 37200 ...
##  $ Veh.Style_recode1: Factor w/ 16 levels "2dr Hatchback",..: 9 7 9 9 7 9
```

```
7 9 7 7 ...
##  $ Veh.Style_recode : Factor w/ 15 levels "2dr Hatchback",..: 7 5 7 7 5 7
5 7 5 5 ...

#View result

#reorder argument with arrange the levels in ascending other, which makes it
easy to view the trend.
#This is also a utility code chunk that I was adapting for different plots of
different variables.


car_nonEx %>%
  #group_by(Make) %>% summarise(MSRP_Median = median(MSRP)) %>%
  #arrange(desc(MSRP_Median)) %>%
  #ggplot(aes(x = reorder(Make,-MSRP_Median), y = MSRP_Median, fill = Engine.
Cylinders)) +
  ggplot(aes(x = reorder(Make,-MSRP), y = MSRP, fill = as.factor(Engine.Cylin
ders))) +
  geom_bar(stat="identity") +
  #geom_text(aes(label=Make),hjust=0.5, vjust=-0.5) +
  #theme(axis.text.x = element_blank(angle = 45)) +
  xlab("Make") +
  ggtitle("Make/Price")
```




```
#Vehicle.Style had 16 variables with some that a redundant.
car_nonEx.new %>%
```

```r
  group_by(Veh.Style_recode) %>% summarise(MSRP_Median = median(MSRP)) %>%
  #arrange(desc(MSRP_Median)) %>%
  #ggplot(aes(x = reorder(Make,-MSRP_Median), y = MSRP_Median, fill = Engine.
Cylinders)) +
  ggplot(aes(x = reorder(Veh.Style_recode,-MSRP_Median), y = MSRP_Median)) +
  geom_bar(stat="identity") +
  #geom_text(aes(label=Make),hjust=0.5, vjust=-0.5) + #this line adds label t
o the datapoints so I can where the outliers come from
  #theme(axis.text.x = element_blank(angle = 45)) +
  #xlab("Make") +
  ggtitle("Make/Price")

## `summarise()` ungrouping output (override with `.groups` argument)
```

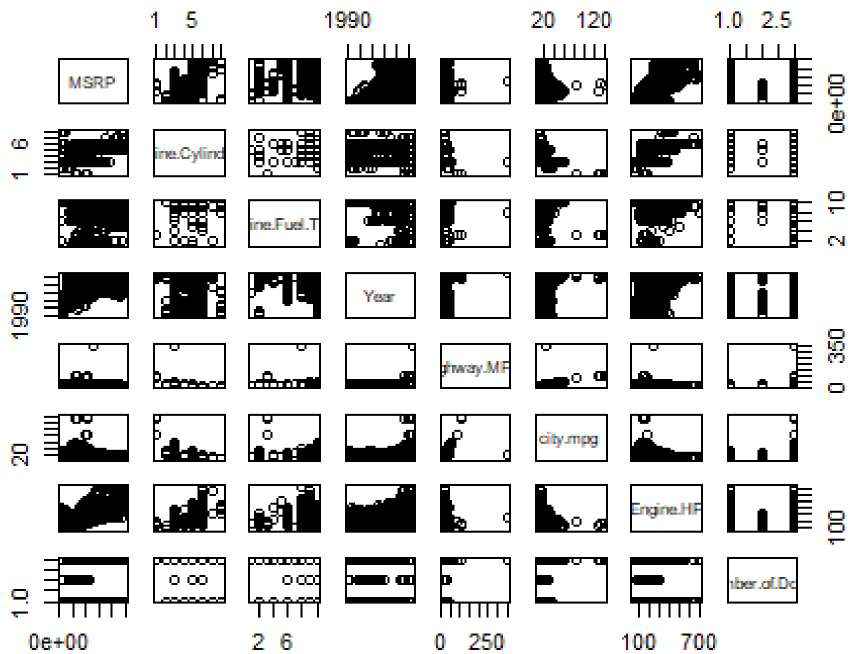



```r
#View names of levels that makes up the factor
unique(car_nonEx.new$Veh.Style_recode)

##  [1] Coupe               Convertible         Sedan
##  [4] Wagon               4dr Hatchback       2dr Hatchback
##  [7] SUV                 Passenger Minivan   Cargo Minivan
## [10] Crew Cab Pickup     Regular Cab Pickup  Extended Cab Pickup
## [13] Cargo Van           Convertible SUV     Passenger Van
## 15 Levels: 2dr Hatchback 4dr Hatchback Cargo Minivan Cargo Van ... Wagon

colSums(is.na(car_nonEx.new))

##              Make             Model              Year  Engine.Fuel.Type
##                 0                 0                 0                 0
```

```
##         Engine.HP  Engine.Cylinders Transmission.Type     Driven_Wheels
##                 0                 0                 0                 0
##   Number.of.Doors      Vehicle.Size     Vehicle.Style       highway.MPG
##                 0                 0                 0                 0
##          city.mpg        Popularity              MSRP Veh.Style_recode1
##                 0                 0                 0                 0
##  Veh.Style_recode
##                 0
```

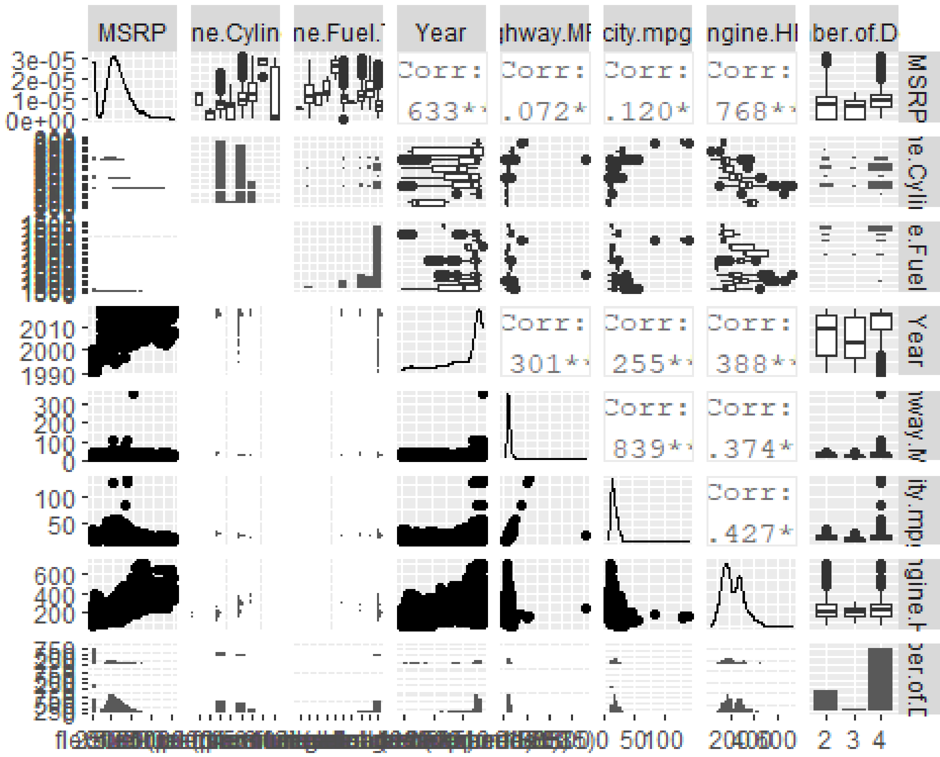
##EDA-3

```r
#For a pairwise view of plots
#My plan was to view a set 5 or 6 features at a time instead of the whole thi
ng.
#I played with log, squared of the variables as I.
#these plots show that city.MPG and Highway MPG do not show any considerable
association with MSRP.
#Make and model do not have meaningful association with the MSRP below 100000
.

library(GGally)
#Pairwise plots
df2.new <- car_nonEx.new %>%
    mutate(log.MSRP = log(MSRP),
          sq.Engine.HP=(Engine.HP)^2) %>%
    dplyr::select(c(MSRP, Driven_Wheels, Transmission.Type, Number.of.Doors,
Vehicle.Size, Veh.Style_recode))

df2.raw <- car_nonEx %>%
        dplyr::select(c(MSRP, Engine.Cylinders, Engine.Fuel.Type, Year,highwa
y.MPG, city.mpg, Engine.HP, Number.of.Doors))


pairs(df2.raw)
```




```r
ggpairs(df2.raw)
```

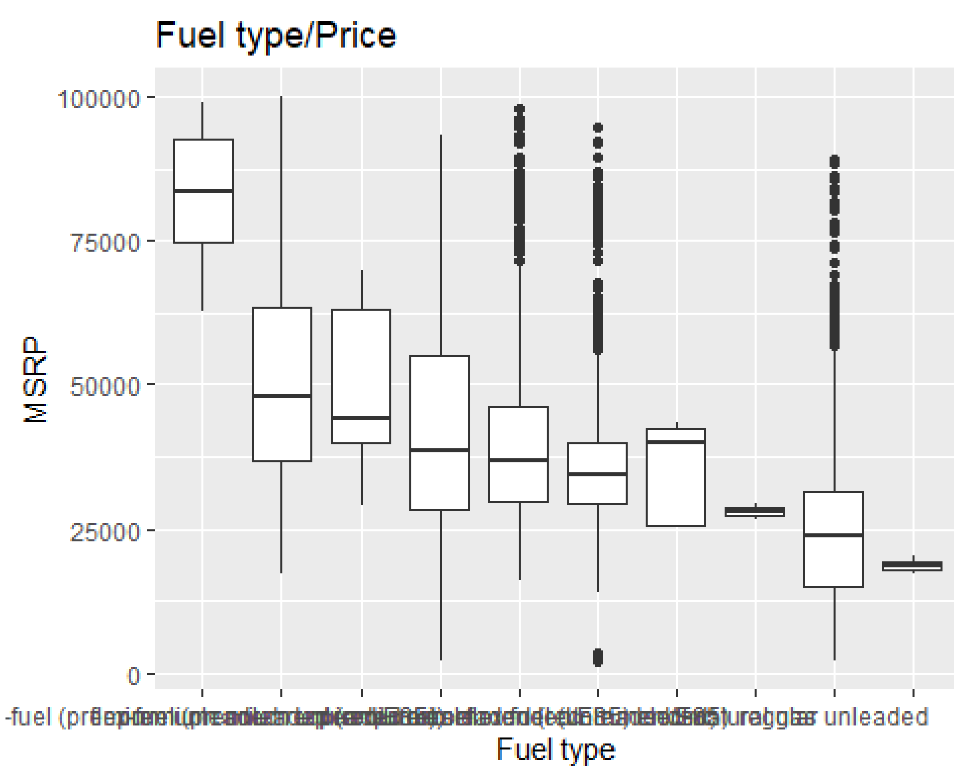
```
## `stat_bin()` using `bins = 30`. Pick better value with `binwidth`.
## `stat_bin()` using `bins = 30`. Pick better value with `binwidth`.
## `stat_bin()` using `bins = 30`. Pick better value with `binwidth`.
## `stat_bin()` using `bins = 30`. Pick better value with `binwidth`.
## `stat_bin()` using `bins = 30`. Pick better value with `binwidth`.
## `stat_bin()` using `bins = 30`. Pick better value with `binwidth`.
## `stat_bin()` using `bins = 30`. Pick better value with `binwidth`.
## `stat_bin()` using `bins = 30`. Pick better value with `binwidth`.
## `stat_bin()` using `bins = 30`. Pick better value with `binwidth`.
## `stat_bin()` using `bins = 30`. Pick better value with `binwidth`.
## `stat_bin()` using `bins = 30`. Pick better value with `binwidth`.
## `stat_bin()` using `bins = 30`. Pick better value with `binwidth`.
## `stat_bin()` using `bins = 30`. Pick better value with `binwidth`.
## `stat_bin()` using `bins = 30`. Pick better value with `binwidth`.
## `stat_bin()` using `bins = 30`. Pick better value with `binwidth`.
```




```
#what levels are in fuel type
unique(car_nonEx$Engine.Cylinders)

## [1] 6  4  5  8  12 0  3  10
## Levels: 0 3 4 5 6 8 10 12

str(car_nonEx)

## 'data.frame':    11182 obs. of  15 variables:
##  $ Make             : Factor w/ 39 levels "Acura","Alfa Romeo",..: 5 5 5 5
5 5 5 5 5 5 ...
##  $ Model            : Factor w/ 811 levels "1 Series","1 Series M",..: 2 1
1 1 1 1 1 1 1 1 ...
##  $ Year             : int  2011 2011 2011 2011 2011 2012 2012 2012 2012 20
13 ...
##  $ Engine.Fuel.Type : Factor w/ 10 levels "","diesel","electric",..: 9 9 9
9 9 9 9 9 9 9 ...
##  $ Engine.HP        : int  335 300 300 230 230 230 300 300 230 230 ...
##  $ Engine.Cylinders : Factor w/ 8 levels "0","3","4","5",..: 5 5 5 5 5 5 5
5 5 5 ...
##  $ Transmission.Type: Factor w/ 5 levels "AUTOMATED_MANUAL",..: 4 4 4 4 4
4 4 4 4 4 ...
##  $ Driven_Wheels    : Factor w/ 4 levels "all wheel drive",..: 4 4 4 4 4 4
4 4 4 4 ...
##  $ Number.of.Doors  : Factor w/ 3 levels "2","3","4": 1 1 1 1 1 1 1 1 1 1
...
##  $ Vehicle.Size     : Factor w/ 3 levels "Compact","Large",..: 1 1 1 1 1 1
```

```
1 1 1 1 ...
##  $ Vehicle.Style    : Factor w/ 16 levels "2dr Hatchback",..: 9 7 9 9 7 9
7 9 7 7 ...
##  $ highway.MPG      : int  26 28 28 28 28 28 26 28 28 27 ...
##  $ city.mpg         : int  19 19 20 18 18 18 17 20 18 18 ...
##  $ Popularity       : Factor w/ 39 levels "21","26","61",..: 38 38 38 38 3
8 38 38 38 38 38 ...
##  $ MSRP             : int  46135 40650 36350 29450 34500 31200 44100 39300
36900 37200 ...
```

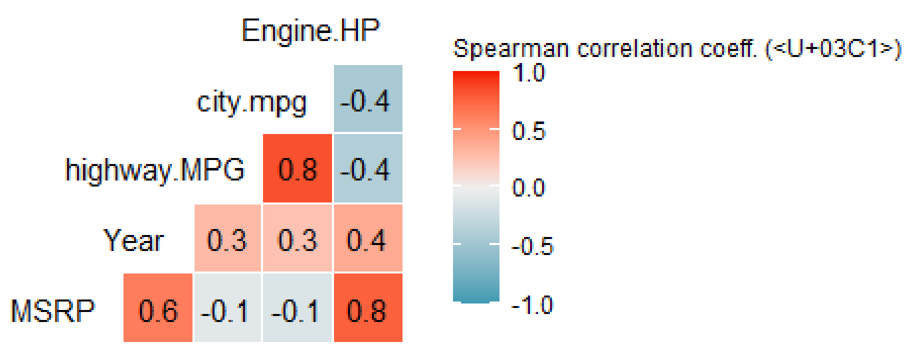
```
view(car_nonEx$Engine.Fuel.Type)

#Grouped plot
car_nonEx %>%
  group_by(Engine.Fuel.Type) %>%
  #summarise(MSRP_Median = median(MSRP)) %>%
  #arrange(desc(MSRP_Median)) %>%
  ggplot(aes(x = reorder(Engine.Fuel.Type,-MSRP), y = MSRP)) +
  geom_boxplot() +
  #geom_text(aes(label=Make),hjust=0.5, vjust=-0.5) + #this line adds label t
o the datapoints so I can where the outliers come from
  #theme(axis.text.x = element_blank(angle = 45)) +
  xlab("Fuel type") +
  ggtitle("Fuel type/Price")
```




```
#correlation of the selected variables
```

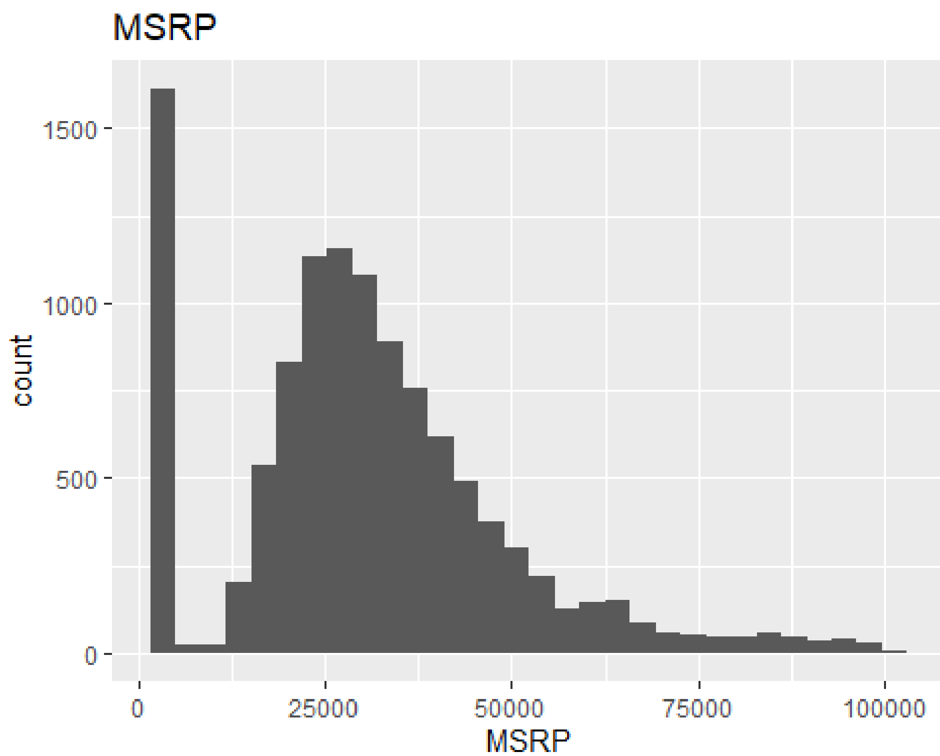
```r
df2.raw %>% ggcorr(palette = "RdBu", label = TRUE, hjust= 0.9, layout.exp = 1
.2, name = "Spearman correlation coeff. (ρ)")
```

```
## Warning in ggcorr(., palette = "RdBu", label = TRUE, hjust = 0.9, layout.e
xp
## = 1.2, : data in column(s) 'Engine.Cylinders', 'Engine.Fuel.Type',
## 'Number.of.Doors' are not numeric and were ignored
```




```r
#No association with City MPG/highway MPG so drop from model.


#MSRP histogram to see MSRP distribution. There was some very very cheap cars
0 to 5000 that are seriously skewing the result.
car_nonEx.new %>%
  ggplot(aes(x =MSRP)) +
  geom_histogram() +
  #xlab("Fuel type") +
  ggtitle("MSRP")
```

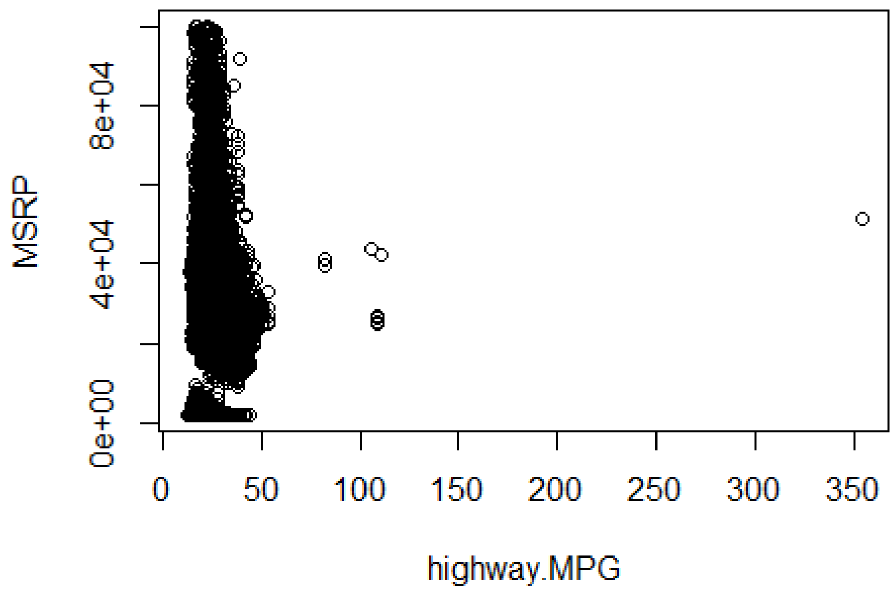
```
## `stat_bin()` using `bins = 30`. Pick better value with `binwidth`.
```




```
car_v.cheap = car_nonEx.new %>% filter(MSRP<5000)
car_nonEx.new = car_nonEx.new %>% filter(MSRP >=5000)
```

There is no association of MSRP with City MPG nor Highway MPG

```
plot( MSRP~(highway.MPG), data = car_nonEx)
```



##Multiple linear regression

```
##Fit a preliminary model using the variables that have some association with
MSRP


Model1<- lm(log(MSRP)~Year+(Engine.HP)^2+Transmission.Type+Driven_Wheels+Numb
er.of.Doors+Vehicle.Size+Veh.Style_recode+Engine.Fuel.Type, data=car_nonEx.ne
w)
summary(Model1)
```

```
##
## Call:
## lm(formula = log(MSRP) ~ Year + (Engine.HP)^2 + Transmission.Type +
##     Driven_Wheels + Number.of.Doors + Vehicle.Size + Veh.Style_recode +
##     Engine.Fuel.Type, data = car_nonEx.new)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -1.8284 -0.1124 -0.0061  0.1092  1.2706
##
## Coefficients:
##                                                               Estimate
## (Intercept)                                                 -2.109e+01
## Year                                                         1.526e-02
## Engine.HP                                                    2.481e-03
## Transmission.TypeAUTOMATIC                                  -2.049e-02
```

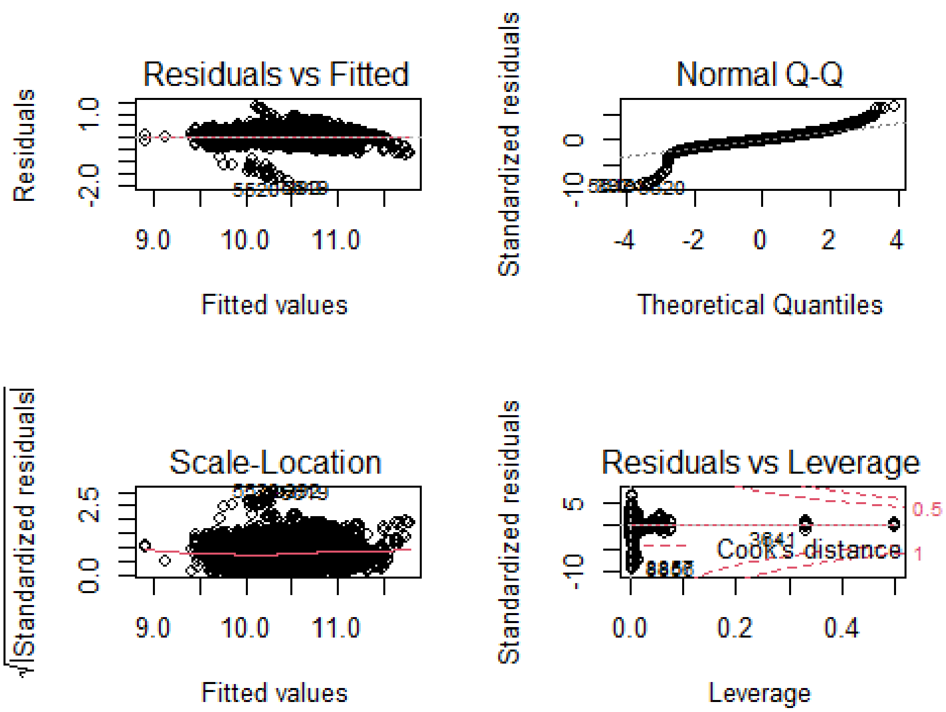
```
## Transmission.TypeDIRECT_DRIVE                                   5.550e-02
## Transmission.TypeMANUAL                                         -1.549e-01
## Transmission.TypeUNKNOWN                                        -1.339e+00
## Driven_Wheelsfour wheel drive                                    4.778e-02
## Driven_Wheelsfront wheel drive                                  -1.047e-01
## Driven_Wheelsrear wheel drive                                   -4.550e-02
## Number.of.Doors3                                                -2.647e-02
## Number.of.Doors4                                                -5.210e-02
## Vehicle.SizeLarge                                                1.405e-01
## Vehicle.SizeMidsize                                              3.847e-02
## Veh.Style_recode4dr Hatchback                                    2.990e-02
## Veh.Style_recodeCargo Minivan                                    5.535e-02
## Veh.Style_recodeCargo Van                                       -1.554e-02
## Veh.Style_recodeConvertible                                      2.227e-01
## Veh.Style_recodeConvertible SUV                                  1.218e-01
## Veh.Style_recodeCoupe                                            4.841e-02
## Veh.Style_recodeCrew Cab Pickup                                  2.844e-02
## Veh.Style_recodeExtended Cab Pickup                             -4.492e-02
## Veh.Style_recodePassenger Minivan                                1.853e-01
## Veh.Style_recodePassenger Van                                    9.029e-02
## Veh.Style_recodeRegular Cab Pickup                              -1.877e-01
## Veh.Style_recodeSedan                                            9.574e-02
## Veh.Style_recodeSUV                                              1.562e-01
## Veh.Style_recodeWagon                                            1.251e-01
## Engine.Fuel.Typediesel                                           4.612e-01
## Engine.Fuel.Typeelectric                                         4.157e-01
## Engine.Fuel.Typeflex-fuel (premium unleaded recommended/E85)  3.884e-01
## Engine.Fuel.Typeflex-fuel (premium unleaded required/E85)     4.718e-01
## Engine.Fuel.Typeflex-fuel (unleaded/E85)                      6.578e-02
## Engine.Fuel.Typenatural gas                                      3.929e-01
## Engine.Fuel.Typepremium unleaded (recommended)                   2.252e-01
## Engine.Fuel.Typepremium unleaded (required)                      3.407e-01
## Engine.Fuel.Typeregular unleaded                                 6.938e-02
##                                                                 Std. Error t
value
## (Intercept)                                                      9.827e-01 -2
1.465
## Year                                                             4.877e-04  3
1.291
## Engine.HP                                                        3.718e-05  6
6.718
## Transmission.TypeAUTOMATIC                                       1.044e-02  -
1.962
## Transmission.TypeDIRECT_DRIVE                                    1.376e-01
0.403
## Transmission.TypeMANUAL                                          1.085e-02 -1
4.278
## Transmission.TypeUNKNOWN                                         1.131e-01 -1
1.834
## Driven_Wheelsfour wheel drive                                    8.865e-03
```

```
5.389
## Driven_Wheelsfront wheel drive                              6.293e-03 -1
6.632
## Driven_Wheelsrear wheel drive                               7.042e-03  -
6.461
## Number.of.Doors3                                            2.570e-02  -
1.030
## Number.of.Doors4                                            2.206e-02  -
2.362
## Vehicle.SizeLarge                                           7.602e-03  1
8.480
## Vehicle.SizeMidsize                                         5.727e-03
6.718
## Veh.Style_recode4dr Hatchback                               2.448e-02
1.221
## Veh.Style_recodeCargo Minivan                               3.278e-02
1.689
## Veh.Style_recodeCargo Van                                   3.875e-02  -
0.401
## Veh.Style_recodeConvertible                                 1.388e-02  1
6.052
## Veh.Style_recodeConvertible SUV                             4.739e-02
2.570
## Veh.Style_recodeCoupe                                       1.354e-02
3.576
## Veh.Style_recodeCrew Cab Pickup                             2.552e-02
1.114
## Veh.Style_recodeExtended Cab Pickup                         2.492e-02  -
1.802
## Veh.Style_recodePassenger Minivan                           2.534e-02
7.312
## Veh.Style_recodePassenger Van                               3.343e-02
2.701
## Veh.Style_recodeRegular Cab Pickup                          1.858e-02 -1
0.103
## Veh.Style_recodeSedan                                       2.371e-02
4.039
## Veh.Style_recodeSUV                                         2.370e-02
6.590
## Veh.Style_recodeWagon                                       2.503e-02
4.996
## Engine.Fuel.Typediesel                                      1.134e-01
4.068
## Engine.Fuel.Typeelectric                                    1.854e-01
2.242
## Engine.Fuel.Typeflex-fuel (premium unleaded recommended/E85)  1.185e-01
3.278
## Engine.Fuel.Typeflex-fuel (premium unleaded required/E85)     1.225e-01
3.852
## Engine.Fuel.Typeflex-fuel (unleaded/E85)                    1.124e-01
```

```
0.585
## Engine.Fuel.Typenatural gas                                  1.771e-01
2.218
## Engine.Fuel.Typepremium unleaded (recommended)                1.123e-01
2.005
## Engine.Fuel.Typepremium unleaded (required)                   1.123e-01
3.034
## Engine.Fuel.Typeregular unleaded                              1.121e-01
0.619
##                                                              Pr(>|t|)
## (Intercept)                                                   < 2e-16 ***
## Year                                                          < 2e-16 ***
## Engine.HP                                                     < 2e-16 ***
## Transmission.TypeAUTOMATIC                                   0.049734 *
## Transmission.TypeDIRECT_DRIVE                                0.686600
## Transmission.TypeMANUAL                                       < 2e-16 ***
## Transmission.TypeUNKNOWN                                      < 2e-16 ***
## Driven_Wheelsfour wheel drive                                7.24e-08 ***
## Driven_Wheelsfront wheel drive                                < 2e-16 ***
## Driven_Wheelsrear wheel drive                                1.09e-10 ***
## Number.of.Doors3                                             0.303204
## Number.of.Doors4                                             0.018219 *
## Vehicle.SizeLarge                                             < 2e-16 ***
## Vehicle.SizeMidsize                                          1.94e-11 ***
## Veh.Style_recode4dr Hatchback                                0.221987
## Veh.Style_recodeCargo Minivan                                0.091333 .
## Veh.Style_recodeCargo Van                                    0.688412
## Veh.Style_recodeConvertible                                   < 2e-16 ***
## Veh.Style_recodeConvertible SUV                              0.010198 *
## Veh.Style_recodeCoupe                                        0.000351 ***
## Veh.Style_recodeCrew Cab Pickup                              0.265215
## Veh.Style_recodeExtended Cab Pickup                          0.071513 .
## Veh.Style_recodePassenger Minivan                            2.85e-13 ***
## Veh.Style_recodePassenger Van                                0.006923 **
## Veh.Style_recodeRegular Cab Pickup                            < 2e-16 ***
## Veh.Style_recodeSedan                                        5.42e-05 ***
## Veh.Style_recodeSUV                                          4.64e-11 ***
## Veh.Style_recodeWagon                                        5.95e-07 ***
## Engine.Fuel.Typediesel                                       4.77e-05 ***
## Engine.Fuel.Typeelectric                                     0.025013 *
## Engine.Fuel.Typeflex-fuel (premium unleaded recommended/E85) 0.001048 **
## Engine.Fuel.Typeflex-fuel (premium unleaded required/E85)    0.000118 ***
## Engine.Fuel.Typeflex-fuel (unleaded/E85)                     0.558473
## Engine.Fuel.Typenatural gas                                  0.026572 *
## Engine.Fuel.Typepremium unleaded (recommended)               0.044975 *
## Engine.Fuel.Typepremium unleaded (required)                  0.002420 **
## Engine.Fuel.Typeregular unleaded                             0.536046
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
```

```
## Residual standard error: 0.1939 on 9534 degrees of freedom
## Multiple R-squared:  0.7855, Adjusted R-squared:  0.7847
## F-statistic: 969.9 on 36 and 9534 DF,  p-value: < 2.2e-16
```

```r
par(mfrow=c(2,2))
plot(Model1)
```




```r
plot(cooks.distance(Model1))
#Cooks D shows outliers but they don't have strong leverage. 90 % of the resi
duals are within -2 to +2 and have random cloud  and relative constant varian
ce.
```

```r
car::vif(Model1) # Variance Inflation Factor (anything above 10 is a problem)
```

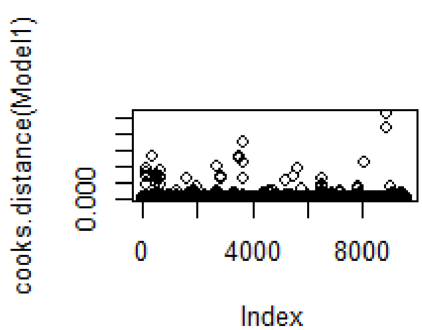
```
##                          GVIF Df GVIF^(1/(2*Df))
## Year                 1.275534  1        1.129395
## Engine.HP            2.678279  1        1.636545
## Transmission.Type   12.525099  4        1.371585
## Driven_Wheels        4.038226  3        1.261920
## Number.of.Doors     58.612240  2        2.766923
## Vehicle.Size         2.831413  2        1.297182
## Veh.Style_recode   265.665935 14        1.220628
## Engine.Fuel.Type    21.813260  9        1.186789
```

```r
#there is high multicollinearity with Vehicle style and number of doors.
# even though removing it does not have any significant effect on the model,
they should be removed in other to see the actual predictors contributing to
the effect sizes.


#below the revised model following removal of multicollinear predictors



Model1.new<- lm(log(MSRP)~Year+(Engine.HP)^2+Transmission.Type+Driven_Wheels+
Vehicle.Size+Engine.Fuel.Type, data=car_nonEx.new, )
summary(Model1.new)

##
## Call:
## lm(formula = log(MSRP) ~ Year + (Engine.HP)^2 + Transmission.Type +
##     Driven_Wheels + Vehicle.Size + Engine.Fuel.Type, data = car_nonEx.new)
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -1.84652 -0.12007 -0.00888  0.11533  1.27863
##
## Coefficients:
##                                                              Estimate
## (Intercept)                                                -1.884e+01
## Year                                                        1.413e-02
## Engine.HP                                                   2.668e-03
## Transmission.TypeAUTOMATIC                                 -4.690e-03
## Transmission.TypeDIRECT_DRIVE                               5.050e-02
## Transmission.TypeMANUAL                                    -1.596e-01
## Transmission.TypeUNKNOWN                                   -1.329e+00
## Driven_Wheelsfour wheel drive                               9.843e-04
## Driven_Wheelsfront wheel drive                             -1.016e-01
## Driven_Wheelsrear wheel drive                              -6.800e-02
## Vehicle.SizeLarge                                           1.043e-01
## Vehicle.SizeMidsize                                         5.084e-02
## Engine.Fuel.Typediesel                                      5.274e-01
## Engine.Fuel.Typeelectric                                    4.035e-01
## Engine.Fuel.Typeflex-fuel (premium unleaded recommended/E85)  4.345e-01
## Engine.Fuel.Typeflex-fuel (premium unleaded required/E85)     5.678e-01
## Engine.Fuel.Typeflex-fuel (unleaded/E85)                    6.163e-02
## Engine.Fuel.Typenatural gas                                 4.262e-01
## Engine.Fuel.Typepremium unleaded (recommended)              2.769e-01
## Engine.Fuel.Typepremium unleaded (required)                 3.899e-01
## Engine.Fuel.Typeregular unleaded                            9.928e-02
##                                                            Std. Error t
value
## (Intercept)                                                 1.000e+00 -1
8.830
```

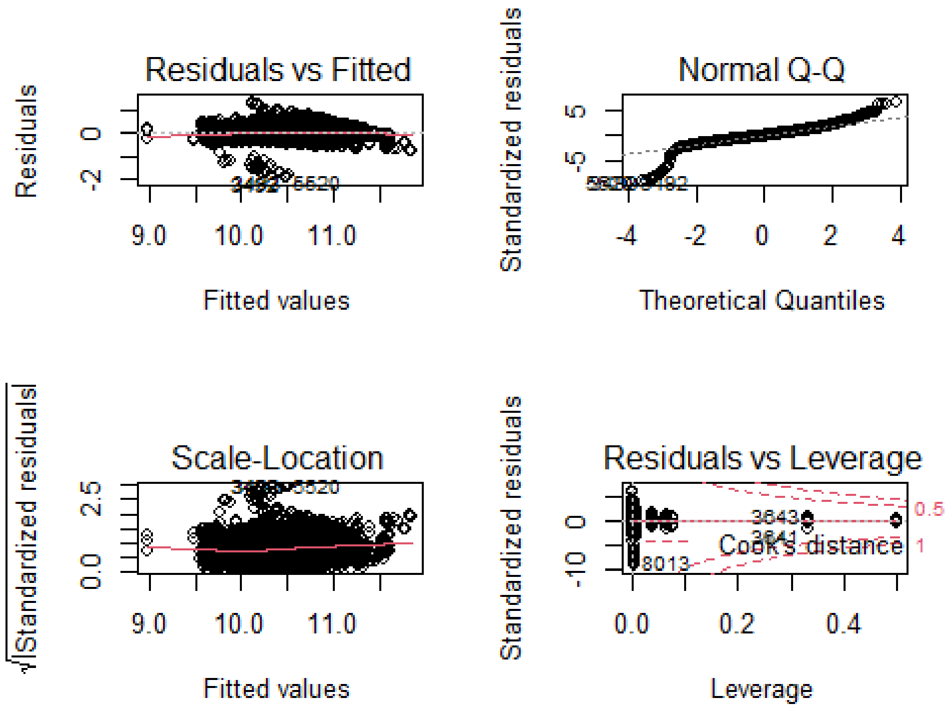
```
## Year                                                    4.959e-04  2
8.487
## Engine.HP                                               3.746e-05  7
1.227
## Transmission.TypeAUTOMATIC                              1.093e-02  -
0.429
## Transmission.TypeDIRECT_DRIVE                           1.459e-01
0.346
## Transmission.TypeMANUAL                                 1.140e-02 -1
4.006
## Transmission.TypeUNKNOWN                                1.196e-01 -1
1.109
## Driven_Wheelsfour wheel drive                           8.492e-03
0.116
## Driven_Wheelsfront wheel drive                          6.176e-03 -1
6.448
## Driven_Wheelsrear wheel drive                           6.784e-03 -1
0.023
## Vehicle.SizeLarge                                       7.493e-03  1
3.927
## Vehicle.SizeMidsize                                     5.496e-03
9.249
## Engine.Fuel.Typediesel                                  1.202e-01
4.388
## Engine.Fuel.Typeelectric                                1.965e-01
2.054
## Engine.Fuel.Typeflex-fuel (premium unleaded recommended/E85)  1.256e-01
3.459
## Engine.Fuel.Typeflex-fuel (premium unleaded required/E85)     1.298e-01
4.375
## Engine.Fuel.Typeflex-fuel (unleaded/E85)                1.192e-01
0.517
## Engine.Fuel.Typenatural gas                             1.879e-01
2.268
## Engine.Fuel.Typepremium unleaded (recommended)          1.191e-01
2.325
## Engine.Fuel.Typepremium unleaded (required)             1.191e-01
3.274
## Engine.Fuel.Typeregular unleaded                        1.189e-01
0.835
##                                                        Pr(>|t|)
## (Intercept)                                             < 2e-16 ***
## Year                                                    < 2e-16 ***
## Engine.HP                                               < 2e-16 ***
## Transmission.TypeAUTOMATIC                             0.667903
## Transmission.TypeDIRECT_DRIVE                          0.729253
## Transmission.TypeMANUAL                                 < 2e-16 ***
## Transmission.TypeUNKNOWN                                < 2e-16 ***
## Driven_Wheelsfour wheel drive                          0.907731
## Driven_Wheelsfront wheel drive                          < 2e-16 ***
```

```
## Driven_Wheelsrear wheel drive                                    < 2e-16 ***
## Vehicle.SizeLarge                                                < 2e-16 ***
## Vehicle.SizeMidsize                                              < 2e-16 ***
## Engine.Fuel.Typediesel                                          1.16e-05 ***
## Engine.Fuel.Typeelectric                                        0.040019 *
## Engine.Fuel.Typeflex-fuel (premium unleaded recommended/E85) 0.000544 ***
## Engine.Fuel.Typeflex-fuel (premium unleaded required/E85)    1.23e-05 ***
## Engine.Fuel.Typeflex-fuel (unleaded/E85)                        0.605051
## Engine.Fuel.Typenatural gas                                     0.023356 *
## Engine.Fuel.Typepremium unleaded (recommended)                  0.020098 *
## Engine.Fuel.Typepremium unleaded (required)                     0.001064 **
## Engine.Fuel.Typeregular unleaded                                0.403728
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 0.2057 on 9550 degrees of freedom
## Multiple R-squared:  0.7582, Adjusted R-squared:  0.7577
## F-statistic:  1497 on 20 and 9550 DF,  p-value: < 2.2e-16
```

```r
par(mfrow=c(2,2))
```

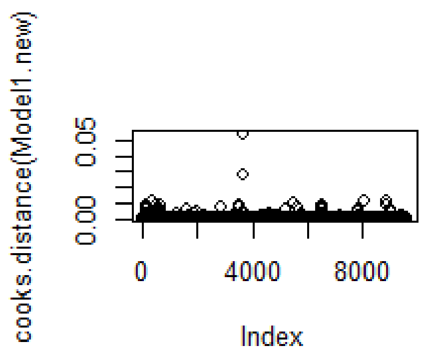



```r
plot(Model1.new)
```




```r
plot(cooks.distance(Model1.new))


str(car_nonEx.new)

## 'data.frame':    9571 obs. of  17 variables:
##  $ Make             : Factor w/ 39 levels "Acura","Alfa Romeo",..: 5 5 5 5
5 5 5 5 5 5 ...
##  $ Model            : Factor w/ 811 levels "1 Series","1 Series M",..: 2 1
1 1 1 1 1 1 1 1 ...
##  $ Year             : int  2011 2011 2011 2011 2011 2012 2012 2012 2012 20
13 ...
##  $ Engine.Fuel.Type : Factor w/ 10 levels "","diesel","electric",..: 9 9 9
9 9 9 9 9 9 9 ...
##  $ Engine.HP        : int  335 300 300 230 230 230 300 300 230 230 ...
##  $ Engine.Cylinders : Factor w/ 8 levels "0","3","4","5",..: 5 5 5 5 5 5 5
5 5 5 ...
##  $ Transmission.Type: Factor w/ 5 levels "AUTOMATED_MANUAL",..: 4 4 4 4 4
4 4 4 4 4 ...
##  $ Driven_Wheels    : Factor w/ 4 levels "all wheel drive",..: 4 4 4 4 4 4
4 4 4 4 ...
##  $ Number.of.Doors  : Factor w/ 3 levels "2","3","4": 1 1 1 1 1 1 1 1 1 1
...
##  $ Vehicle.Size     : Factor w/ 3 levels "Compact","Large",..: 1 1 1 1 1 1
1 1 1 1 ...
##  $ Vehicle.Style    : Factor w/ 16 levels "2dr Hatchback",..: 9 7 9 9 7 9
7 9 7 7 ...
```

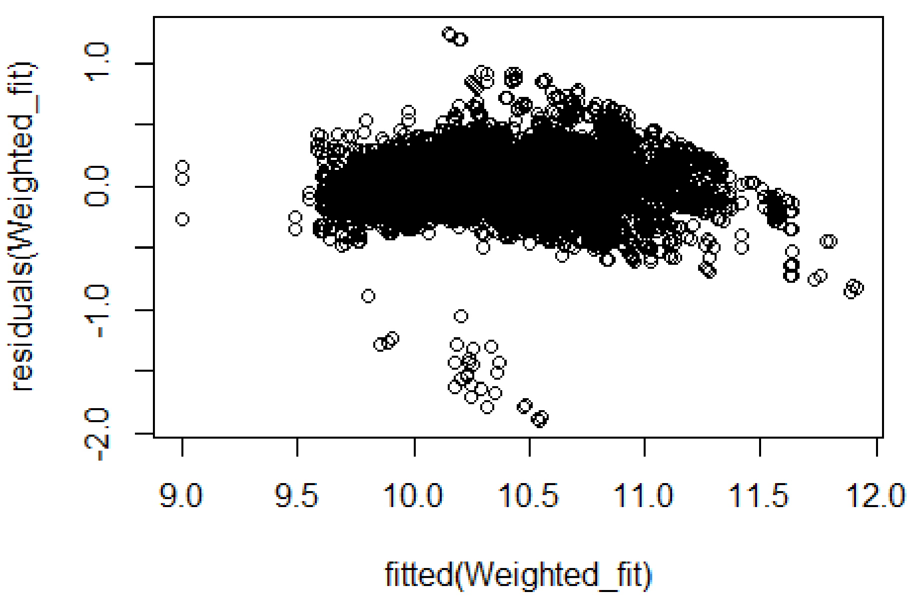
```
##  $ highway.MPG      : int  26 28 28 28 28 28 26 28 28 27 ...
##  $ city.mpg         : int  19 19 20 18 18 18 17 20 18 18 ...
##  $ Popularity       : Factor w/ 39 levels "21","26","61",..: 38 38 38 38 3
8 38 38 38 38 38 ...
##  $ MSRP             : int  46135 40650 36350 29450 34500 31200 44100 39300
36900 37200 ...
##  $ Veh.Style_recode1: Factor w/ 16 levels "2dr Hatchback",..: 9 7 9 9 7 9
7 9 7 7 ...
##  $ Veh.Style_recode : Factor w/ 15 levels "2dr Hatchback",..: 7 5 7 7 5 7
5 7 5 5 ...

#Evaluation metrics for the model:
#Root Mean Squared Error
Root_MSE = sqrt(mean(Model1$residuals^2)) #0.19 which is very low and very go
od
Root_MSE_B = sqrt(mean(Model1.new$residuals^2)) #0.205
```



##Observations: 1. The vif output reveals high multicollinearity in Veh.Style_recode, Number.of.Doors and Engine.Fuel.Type predictors. The first two were the most severe (266 and 58 respectively). 2. When these 3 predictors were removed from the model the adjusted Rsqured dropped from 0.78 to 0.69 and the RMSE increased from 0.205 to 0.235. 3. Given this poor performance, I returned the Engine.Fuel.Type predictor given its mildly multicollinear value. Then the adjusted r-squared increased to 0.76 and RMSE shows 0.205 which is similar to what was obtained for the original full model. But now we can explain that the predictors in this model are contributing to the effect size seeing in the response.

#Weighted least squares regression Calculate fitted values from a regression of absolute residuals vs fitted values.

```
library (MASS)

Weighted_fit <- rlm(log(MSRP)~Year+(Engine.HP)^2+Transmission.Type+Driven_Whe
els+Vehicle.Size+Engine.Fuel.Type, data=car_nonEx.new)


plot(fitted(Weighted_fit), residuals(Weighted_fit))
```




```
##A bit of non-constant variance here so let use weighted to see if there wil
l be improvement.

wts <- 1/fitted(lm(abs(residuals(Weighted_fit)) ~ fitted(Weighted_fit)))^2

Weighted_fit1 <- rlm(log(MSRP)~Year+(Engine.HP)^2+Transmission.Type+Driven_Wh
eels+Vehicle.Size+Engine.Fuel.Type, data=car_nonEx.new, weights = wts)

Root_MSE_B1 = sqrt(mean(Weighted_fit$residuals^2))
```

Transform the MSRP (response) to log for better distribution. It showed better correlation with Engine.HP

```
car_nonEx.newB = car_nonEx.new %>%
  mutate(MSRP.log = log(MSRP))
car_nonEx.newB = subset(car_nonEx.newB, select=-c(MSRP))
```

```
car_nonEx.newB = subset(car_nonEx.newB, select=-c(Veh.Style_recode1,MSRP))


car_nonEx.newB =car_nonEx.newB %>%
  mutate(Year = as.factor(Year))

#verify the result
ncol(car_nonEx.newB)

## [1] 16

str(car_nonEx.newB)

## 'data.frame':    9571 obs. of  16 variables:
##  $ Make             : Factor w/ 39 levels "Acura","Alfa Romeo",..: 5 5 5 5
5 5 5 5 5 5 ...
##  $ Model            : Factor w/ 811 levels "1 Series","1 Series M",..: 2 1
1 1 1 1 1 1 1 1 ...
##  $ Year             : Factor w/ 27 levels "1991","1992",..: 21 21 21 21 21
22 22 22 22 23 ...
##  $ Engine.Fuel.Type : Factor w/ 10 levels "","diesel","electric",..: 9 9 9
9 9 9 9 9 9 9 ...
##  $ Engine.HP        : int  335 300 300 230 230 230 300 300 230 230 ...
##  $ Engine.Cylinders : Factor w/ 8 levels "0","3","4","5",..: 5 5 5 5 5 5 5
5 5 5 ...
##  $ Transmission.Type: Factor w/ 5 levels "AUTOMATED_MANUAL",..: 4 4 4 4 4
4 4 4 4 4 ...
##  $ Driven_Wheels    : Factor w/ 4 levels "all wheel drive",..: 4 4 4 4 4 4
4 4 4 4 ...
##  $ Number.of.Doors  : Factor w/ 3 levels "2","3","4": 1 1 1 1 1 1 1 1 1 1
...
##  $ Vehicle.Size     : Factor w/ 3 levels "Compact","Large",..: 1 1 1 1 1 1
1 1 1 1 ...
##  $ Vehicle.Style    : Factor w/ 16 levels "2dr Hatchback",..: 9 7 9 9 7 9
7 9 7 7 ...
##  $ highway.MPG      : int  26 28 28 28 28 28 26 28 28 27 ...
##  $ city.mpg         : int  19 19 20 18 18 18 17 20 18 18 ...
##  $ Popularity       : Factor w/ 39 levels "21","26","61",..: 38 38 38 38 3
8 38 38 38 38 38 ...
##  $ Veh.Style_recode : Factor w/ 15 levels "2dr Hatchback",..: 7 5 7 7 5 7
5 7 5 5 ...
##  $ MSRP.log         : num  10.7 10.6 10.5 10.3 10.4 ...
```

#Split section for the linear model

```
attach(car_nonEx.newB)
```

```
## The following objects are masked from car.new:
##
##     city.mpg, Driven_Wheels, Engine.Cylinders, Engine.Fuel.Type,
##     Engine.HP, highway.MPG, Make, Model, Number.of.Doors, Popularity,
##     Transmission.Type, Vehicle.Size, Vehicle.Style, Year

set.seed(123)
splitPerc = .80
splitPerc2 = .50
trainIndices1 = sample(1:dim(car_nonEx.newB)[1],round(splitPerc * dim(car_non
Ex.newB)[1]))
train1 = car_nonEx.newB[trainIndices1,]
test_val1 = car_nonEx.newB[-trainIndices1,]

trainIndices1 = sample(1:dim(test_val1)[1],round(splitPerc2 * dim(test_val1)[
1]))
test1 = test_val1[trainIndices1,]
validation1 = test_val1[-trainIndices1,]
dim(car_nonEx.newB)

## [1] 9571   16

dim(train1)

## [1] 7657   16

dim(test1)

## [1] 957  16

dim(validation1)

## [1] 957  16
```

#Fit the regression model using the “train” split. This model used the selected predictors used in the full dataset above

```
model_2 = lm(MSRP.log~(Engine.HP)^2+Transmission.Type+Driven_Wheels+Vehicle.S
ize+Engine.Fuel.Type+Year, data=train1)



Root_MSE_2.train = sqrt(mean(model_2$residuals^2)) #0.19 which is very low an
d very good



MSE_2.test = mean((test1$MSRP.log - predict.lm(model_2, test1))^2)
```

```r
Root_MSE_2.test = sqrt(MSE_2.test) #This test RMSE is the best technique for evaluating a model. But it looks high on this model.


#For weighted Linear Regression

Weighted_fit <- rlm(log(MSRP)~Year+(Engine.HP)^2+Transmission.Type+Driven_Wheels+Vehicle.Size+Engine.Fuel.Type, data=car_nonEx.new)
wts <- 1/fitted(lm(abs(residuals(Weighted_fit)) ~ fitted(Weighted_fit)))^2

Weighted_fit1 <- rlm(log(MSRP)~Year+(Engine.HP)^2+Transmission.Type+Driven_Wheels+Vehicle.Size+Engine.Fuel.Type, data=car_nonEx.new, weights = wts)
```

Given the huge disparity between Train RMSE and Test RMSE, it seems the model overfit the data. But how come?

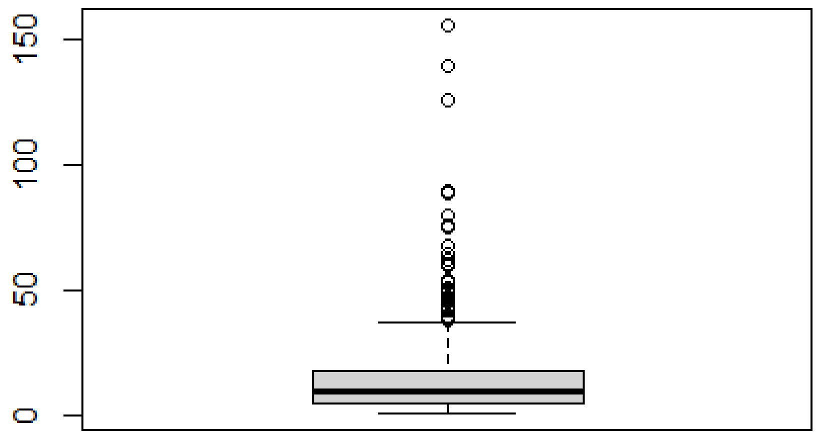
#Complex model section

Feature engineering:

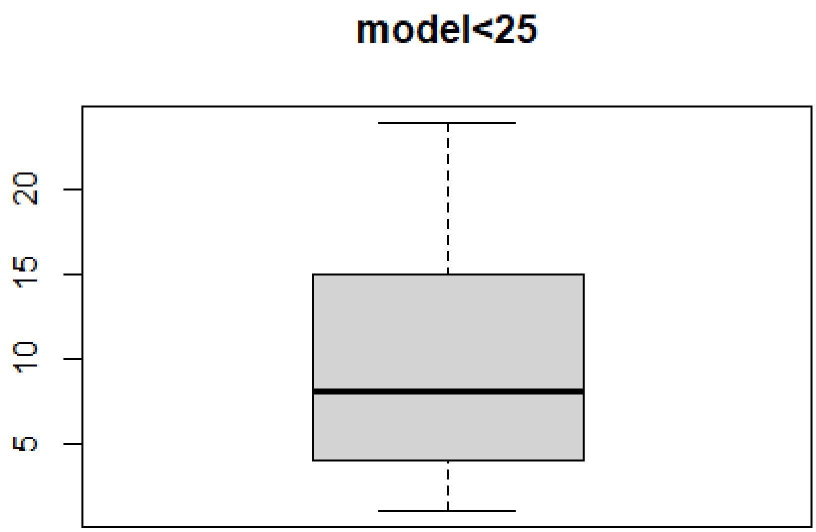
Model currently has 811 levels and random forest wants a maximum of 32. It turns out most levels have 1 to 10 observations. These are too few so I will collapse these levels and rename them.

```r
#Grouped plot
# Most levels have less than 50 observations.
model.count3 = car_nonEx.new %>%
  group_by(Model) %>%
  summarise(count=n())

## `summarise()` ungrouping output (override with `.groups` argument)

attach(model.count3)

## The following object is masked from car_nonEx.newB:
##
##     Model

## The following object is masked from car.new:
##
##     Model

#Most of the model recorded has less than 25 observations
```

```r
model.25 = model.count3 %>% filter(model.count3$count<25)
model.25$Model = as.character(model.25$Model)


#View the new distribution
boxplot(model.count3$count, data=model.count3)
```

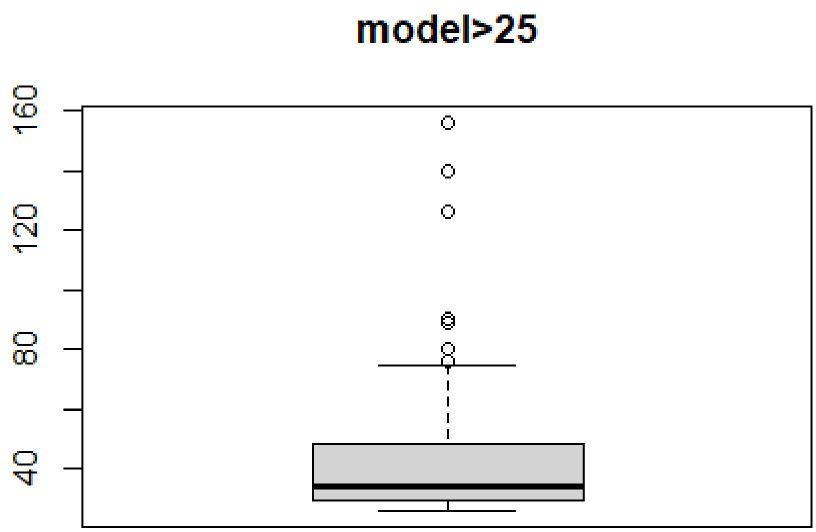



```r
boxplot(model.25$count, data=model.25, main = "model<25")
```




```
#Recode the model variables that have low counts. Threshold of 25 counts (obs
ervations) is used here)

car_nonEx.New2 <- car_nonEx.new %>%
  mutate(Model_recode = as.character(Model))

car_nonEx.New2 <- car_nonEx.New2 %>%
  mutate(Model_recode = ifelse(Model_recode %in% as.character(model.25$Model)
, "Model<25", Model_recode)
         )

car_nonEx.New2 <- car_nonEx.New2 %>%
  mutate(Model_recode = as.factor(Model_recode)
         )
unique(car_nonEx.New2$Model_recode)

##  [1] Model<25                3 Series                300
##  [4] 350Z                    370Z                    3
##  [7] 4 Series                4Runner                 500
## [10] 9-3                     A3                      A4
## [13] Acadia                  Accord                  Aerio
## [16] ATS Coupe               ATS                     B9 Tribeca
## [19] Beetle Convertible      Beetle                  C-Class
## [22] Camaro                  Camry Solara            Canyon
## [25] CC                      Challenger              Charger
## [28] Civic                   Colorado                Corolla
```

```
## [31] Corvette              CR-V                  Cruze
## [34] CTS                   CX-5                  Dakota
## [37] Durango               E-Class               Encore
## [40] Escalade ESV          Escalade              Esteem
## [43] Expedition            Explorer Sport Trac   F-150
## [46] Forenza               Forte                 Frontier
## [49] Golf GTI              Golf                  GTI
## [52] Impreza               Jetta GLI             Jetta SportWagen
## [55] Jetta                 Journey               Juke
## [58] Kizashi               MDX                   Murano
## [61] MX-5 Miata            New Beetle            Outlander Sport
## [64] Passat                Pathfinder            Pilot
## [67] Q50                   Ram Pickup 1500       Range Rover Evoque
## [70] Ranger                S-10                  S60
## [73] Sequoia               Sienna                Sierra 1500 Classic
## [76] Sierra 1500           Silverado 1500 Classic Silverado 1500
## [79] Sonata                Sonic                 Sorento
## [82] SX4                   Tacoma                Terrain
## [85] Tiguan                Titan                 TrailBlazer
## [88] Transit Connect       Transit Wagon         Traverse
## [91] Tribute               Tundra                Veloster
## [94] Venza                 WRX                   XC60
## [97] XL-7                  XL7                   XTS
## 99 Levels: 3 3 Series 300 350Z 370Z 4 Series 4Runner 500 9-3 A3 A4 ... XTS

#model.count3$Model_recode.new = model.count$Model[model.count$count < 30] =
"Model<25"

model.25more = model.count3 %>% filter(model.count3$count>25)
boxplot(model.25more$count, data=model.25more, main = "model>25")
```




```
#Model factor has now ben reduced from 810 levels to 99 levels. More work nee
ds to be done though.

#Remove columns that had been recoded
car_nonEx.New2 = subset(car_nonEx.New2, select=-c(Veh.Style_recode1, Model, P
opularity))

#re-order column to make response the last column.
car_nonEx.New2b = car_nonEx.New2[,c(1,15,14,2,3,4:13)]


#verify the result
ncol(car_nonEx.New2b)

## [1] 15

str(car_nonEx.New2b)

## 'data.frame':    9571 obs. of  15 variables:
##  $ Make             : Factor w/ 39 levels "Acura","Alfa Romeo",..: 5 5 5 5
5 5 5 5 5 5 ...
##  $ Model_recode     : Factor w/ 99 levels "3","3 Series",..: 59 59 59 59 5
9 59 59 59 59 59 ...
##  $ Veh.Style_recode : Factor w/ 15 levels "2dr Hatchback",..: 7 5 7 7 5 7
5 7 5 5 ...
##  $ Year             : int  2011 2011 2011 2011 2011 2012 2012 2012 2012 20
13 ...
```

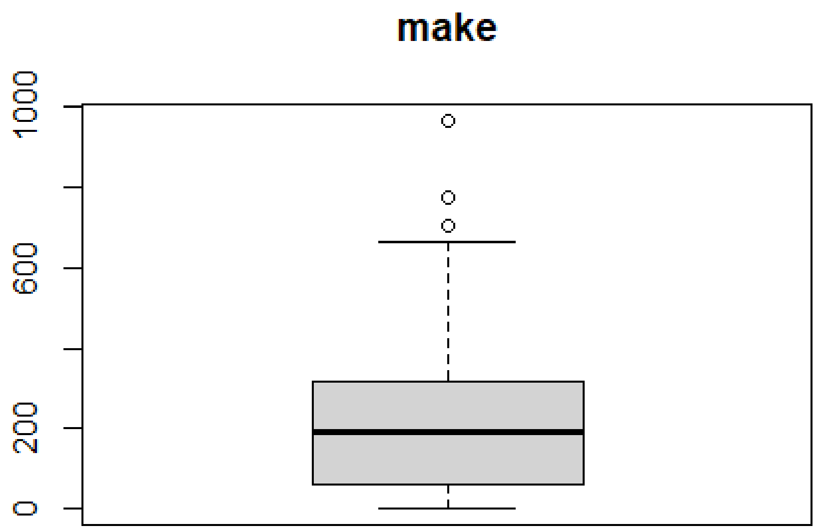
```
## $ Engine.Fuel.Type : Factor w/ 10 levels "","diesel","electric",..: 9 9 9
9 9 9 9 9 9 9 ...
## $ Engine.HP        : int 335 300 300 230 230 230 300 300 230 230 ...
## $ Engine.Cylinders : Factor w/ 8 levels "0","3","4","5",..: 5 5 5 5 5 5 5
5 5 5 ...
## $ Transmission.Type: Factor w/ 5 levels "AUTOMATED_MANUAL",..: 4 4 4 4 4
4 4 4 4 4 ...
## $ Driven_Wheels    : Factor w/ 4 levels "all wheel drive",..: 4 4 4 4 4 4
4 4 4 4 ...
## $ Number.of.Doors  : Factor w/ 3 levels "2","3","4": 1 1 1 1 1 1 1 1 1 1
...
## $ Vehicle.Size     : Factor w/ 3 levels "Compact","Large",..: 1 1 1 1 1 1
1 1 1 1 ...
## $ Vehicle.Style    : Factor w/ 16 levels "2dr Hatchback",..: 9 7 9 9 7 9
7 9 7 7 ...
## $ highway.MPG      : int 26 28 28 28 28 28 26 28 28 27 ...
## $ city.mpg         : int 19 19 20 18 18 18 17 20 18 18 ...
## $ MSRP             : int 46135 40650 36350 29450 34500 31200 44100 39300
36900 37200 ...
```

```
#Grouped plot for Make
# Most levels have less than 400 observations.
#remove the Make with fewer than 60 observations.
make.count = car_nonEx.New2b %>%
  group_by(Make) %>%
  summarise(count=n())
```

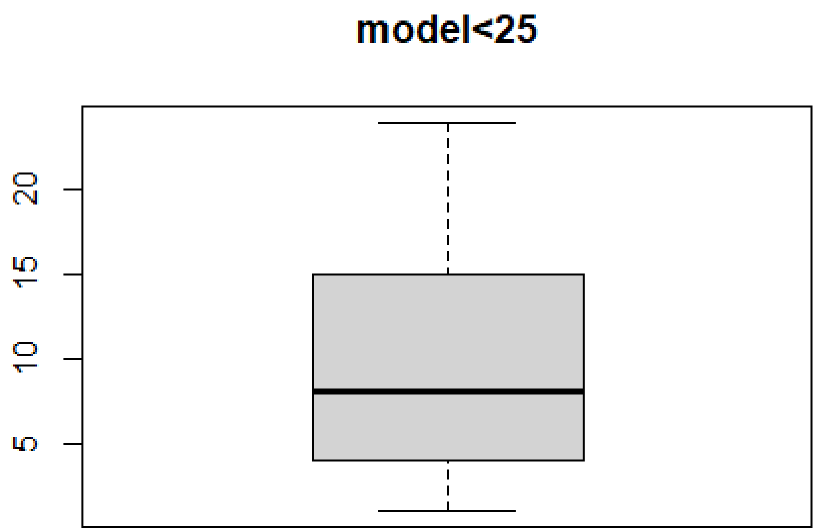
```
## `summarise()` ungrouping output (override with `.groups` argument)
```

```
#filter for Make level that has less than 80obs



make.60 = make.count %>% filter(make.count$count<60)
make.60 = as.character(make.60$Make) #the next set of recode steps worked bet
ter with as.character then as.factor steps.

make.60less= c("Alfa Romeo", "Aston Martin", "FIAT", "Genesis", "HUMMER", "Lo
tus", "Maserati", "Oldsmobile", "Plymouth")
car_nonEx.New2b <- car_nonEx.New2b %>%
  mutate(Make_recode = as.character(Make))

car_nonEx.New2b <- car_nonEx.New2b %>%
  mutate(Make_recode = ifelse(Make_recode %in% make.60less, "Make<60", Make_r
ecode)
         )

car_nonEx.New2b <- car_nonEx.New2b %>%
  mutate(Make_recode = as.factor(Make_recode)
         )
```

```r
boxplot(make.count$count, data=make.count, main = "make")
```




```r
boxplot(model.25$count, data=model.25, main = "model<25")
```




```
#Ensure MSRP is last column and remove make since it has now ben recoded.


car_nonEx.New3b = subset(car_nonEx.New2b, select=-c(Make))

#re-order column to make response the last column.
car_nonEx.New3b = car_nonEx.New3b[,c(15,1,2,3,4:14)]

#verify the result
ncol(car_nonEx.New3b)

## [1] 15

str(car_nonEx.New3b)

## 'data.frame':    9571 obs. of  15 variables:
##  $ Make_recode      : Factor w/ 31 levels "Acura","Audi",..: 3 3 3 3 3 3 3
3 3 3 ...
##  $ Model_recode     : Factor w/ 99 levels "3","3 Series",..: 59 59 59 59 5
9 59 59 59 59 59 ...
##  $ Veh.Style_recode : Factor w/ 15 levels "2dr Hatchback",..: 7 5 7 7 5 7
5 7 5 5 ...
##  $ Year             : int  2011 2011 2011 2011 2011 2012 2012 2012 2012 20
13 ...
##  $ Engine.Fuel.Type : Factor w/ 10 levels "","diesel","electric",..: 9 9 9
9 9 9 9 9 9 9 ...
##  $ Engine.HP        : int  335 300 300 230 230 230 300 300 230 230 ...
```

```
## $ Engine.Cylinders : Factor w/ 8 levels "0","3","4","5",..: 5 5 5 5 5 5 5
5 5 5 ...
## $ Transmission.Type: Factor w/ 5 levels "AUTOMATED_MANUAL",..: 4 4 4 4 4
4 4 4 4 4 ...
## $ Driven_Wheels    : Factor w/ 4 levels "all wheel drive",..: 4 4 4 4 4 4
4 4 4 4 ...
## $ Number.of.Doors  : Factor w/ 3 levels "2","3","4": 1 1 1 1 1 1 1 1 1 1
...
## $ Vehicle.Size     : Factor w/ 3 levels "Compact","Large",..: 1 1 1 1 1 1
1 1 1 1 ...
## $ Vehicle.Style    : Factor w/ 16 levels "2dr Hatchback",..: 9 7 9 9 7 9
7 9 7 7 ...
## $ highway.MPG      : int  26 28 28 28 28 28 26 28 28 27 ...
## $ city.mpg         : int  19 19 20 18 18 18 17 20 18 18 ...
## $ MSRP             : int  46135 40650 36350 29450 34500 31200 44100 39300
36900 37200 ...

#Result: MSRP in last column we have only Model currently as 99 levels that n
eed to be reduced further. I will use Lasso to reduce them.
```

##adjust the MSRP-HP relationship. ##Scale if needed

```
car_nonEx.New3b = car_nonEx.New3b %>%
  mutate(MSRP.log = log(MSRP))  #Transform MSRP to log MSRP and remove MSRP

car_nonEx.New3b = subset(car_nonEx.New3b, select=-c(MSRP))
```

VARIABLE SELECTION: LINEAR MODELS

```
#following lasso result of best lambdas on the full data set, remove Model_re
code predictor factor
#This was done backwards. I did lasso on full data set and came back here to
remove the variable prior to the train test split. I also did lasso on the tr
ain and test sets as well

car_nonEx.new4 = subset(car_nonEx.New3b, select=-c(Model_recode)) #not factor
s with >31 levels
car_nonEx.new4 =car_nonEx.new4 %>%
  mutate(Year = as.factor(Year)) #Year was being treated as continuous when i
t was not. It is an ordinal variable.
```

Par down variables using lasso to select variables

#Split section for complex model using the refined data set.

```
attach(car_nonEx.new4)

## The following objects are masked from car_nonEx.newB:
##
##     city.mpg, Driven_Wheels, Engine.Cylinders, Engine.Fuel.Type,
##     Engine.HP, highway.MPG, MSRP.log, Number.of.Doors,
##     Transmission.Type, Veh.Style_recode, Vehicle.Size, Vehicle.Style,
##     Year

## The following objects are masked from car.new:
##
##     city.mpg, Driven_Wheels, Engine.Cylinders, Engine.Fuel.Type,
##     Engine.HP, highway.MPG, Number.of.Doors, Transmission.Type,
##     Vehicle.Size, Vehicle.Style, Year

set.seed(123)
splitPerc = .80
splitPerc2 = .50
trainIndices = sample(1:dim(car_nonEx.new4)[1],round(splitPerc * dim(car_nonE
x.new4)[1]))
train = car_nonEx.new4[trainIndices,]
test_val = car_nonEx.new4[-trainIndices,]

trainIndices1 = sample(1:dim(test_val)[1],round(splitPerc2 * dim(test_val)[1]
))
test = test_val[trainIndices1,]
validation = test_val[-trainIndices1,]
dim(car_nonEx.new4)

## [1] 9571   14
```

```
dim(train)

## [1] 7657   14

dim(test)

## [1] 957  14

dim(validation)

## [1] 957  14
```

Search for the best lambdas used to shrink predictors.

```
#10^10 to 10^-2
grid = 10^seq(10, -2, length = 100)
```

Convert to train-test suitable for lasso regression

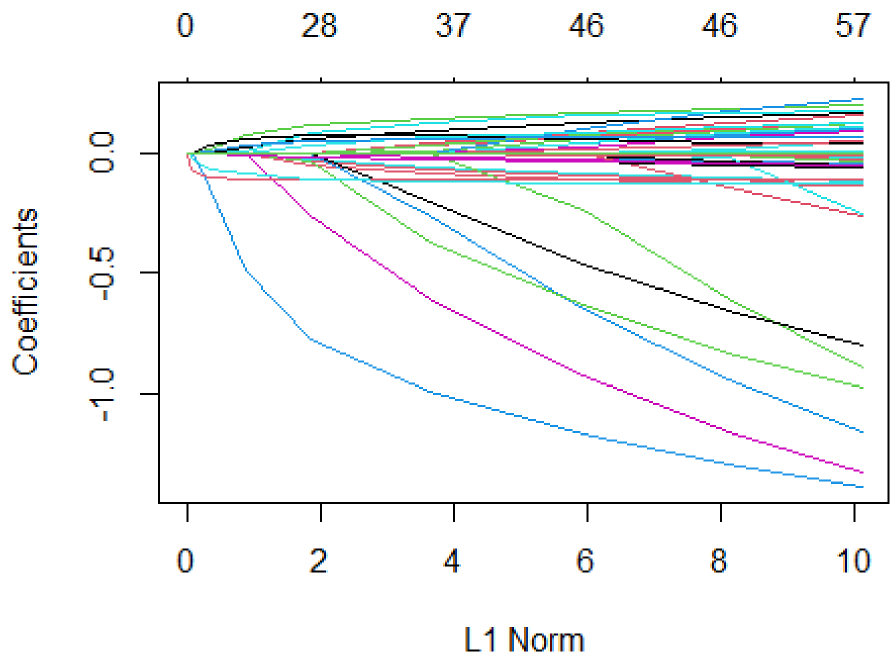
```
x_train = model.matrix(MSRP.log~., train)[,-1]
x_test = model.matrix(MSRP.log~., test)[,-1]

y_train = train %>%
  dplyr::select(MSRP.log) %>%
  unlist() %>%
  as.numeric()

y_test = test %>%
  dplyr::select(MSRP.log) %>%
  unlist() %>%
  as.numeric()
```

LASSO REGRESSION

Next we fit a lasso regression model on the training set, and evaluate its RMSE on the test set.

We expect the coefficient estimates to be much smaller, in terms of l2 norm, when a large value of λ is used, as compared to when a small value of λ is used.

```
library(glmnet)
lasso_mod = glmnet(x_train,
                   y_train,
                   alpha = 1,
                   lambda = grid) # Fit lasso model on training data
par(mfrow=c(1,1))
plot(lasso_mod)    # Draw plot of coefficients

## Warning in regularize.values(x, y, ties, missing(ties), na.rm = na.rm):
## collapsing to unique 'x' values
```




```
#Snooping on the results of the lambdas

lasso_mod$lambda[100] #Display 100th lambda value

## [1] 0.01

coef(lasso_mod)[,100] # Display coefficients associated with 100th lambda val
ue

##                                                    (Intercept)
##                                                    9.715545224
##                                                Make_recodeAudi
##                                                    0.170245545
##                                                 Make_recodeBMW
##                                                    0.123164081
##                                               Make_recodeBuick
##                                                    0.000000000
##                                            Make_recodeCadillac
##                                                    0.200234095
##                                           Make_recodeChevrolet
##                                                   -0.039874229
##                                            Make_recodeChrysler
##                                                    0.000000000
##                                               Make_recodeDodge
##                                                   -0.114220574
##                                                Make_recodeFord
##                                                    0.000000000
```

```
##                                                     Make_recodeGMC
##                                                        0.000000000
##                                                   Make_recodeHonda
##                                                        0.000000000
##                                                 Make_recodeHyundai
##                                                       -0.017516740
##                                                Make_recodeInfiniti
##                                                        0.000000000
##                                                     Make_recodeKia
##                                                       -0.051059103
##                                              Make_recodeLand Rover
##                                                        0.159722893
##                                                   Make_recodeLexus
##                                                        0.122905953
##                                                 Make_recodeLincoln
##                                                        0.100950287
##                                                 Make_recodeMake<60
##                                                        0.108554399
##                                                   Make_recodeMazda
##                                                        0.000000000
##                                           Make_recodeMercedes-Benz
##                                                        0.091433017
##                                              Make_recodeMitsubishi
##                                                       -0.020484169
##                                                  Make_recodeNissan
##                                                       -0.024160764
##                                                 Make_recodePontiac
##                                                       -0.031300109
##                                                 Make_recodePorsche
##                                                        0.228824901
##                                                    Make_recodeSaab
##                                                        0.054453823
##                                                   Make_recodeScion
##                                                       -0.007559823
##                                                  Make_recodeSubaru
##                                                       -0.002854737
##                                                  Make_recodeSuzuki
##                                                       -0.135110956
##                                                  Make_recodeToyota
##                                                        0.000000000
##                                              Make_recodeVolkswagen
##                                                        0.000000000
##                                                   Make_recodeVolvo
##                                                        0.126104548
##                                     Veh.Style_recode4dr Hatchback
##                                                       -0.003436279
##                                     Veh.Style_recodeCargo Minivan
##                                                        0.000000000
##                                         Veh.Style_recodeCargo Van
##                                                        0.000000000
```

```
##                                Veh.Style_recodeConvertible
##                                                0.124408722
##                            Veh.Style_recodeConvertible SUV
##                                                0.000000000
##                                      Veh.Style_recodeCoupe
##                                                0.000000000
##                            Veh.Style_recodeCrew Cab Pickup
##                                                0.000000000
##                        Veh.Style_recodeExtended Cab Pickup
##                                               -0.052415751
##                          Veh.Style_recodePassenger Minivan
##                                                0.001307303
##                              Veh.Style_recodePassenger Van
##                                                0.000000000
##                         Veh.Style_recodeRegular Cab Pickup
##                                               -0.129984649
##                                      Veh.Style_recodeSedan
##                                                0.000000000
##                                        Veh.Style_recodeSUV
##                                                0.000000000
##                                      Veh.Style_recodeWagon
##                                                0.000000000
##                                                   Year1992
##                                               -0.893822356
##                                                   Year1993
##                                               -1.165239282
##                                                   Year1994
##                                               -0.252537998
##                                                   Year1995
##                                               -1.328415846
##                                                   Year1996
##                                               -0.801557940
##                                                   Year1997
##                                               -0.262951977
##                                                   Year1998
##                                                0.000000000
##                                                   Year1999
##                                               -0.979689573
##                                                   Year2000
##                                               -1.387973578
##                                                   Year2001
##                                               -0.013937202
##                                                   Year2002
##                                               -0.011181204
##                                                   Year2003
##                                                0.000000000
##                                                   Year2004
##                                                0.000000000
##                                                   Year2005
##                                                0.000000000
```

```
##                                                     Year2006
##                                                  0.000000000
##                                                     Year2007
##                                                 -0.058929983
##                                                     Year2008
##                                                 -0.005354655
##                                                     Year2009
##                                                  0.000000000
##                                                     Year2010
##                                                  0.000000000
##                                                     Year2011
##                                                  0.000000000
##                                                     Year2012
##                                                  0.000000000
##                                                     Year2013
##                                                  0.000000000
##                                                     Year2014
##                                                  0.000000000
##                                                     Year2015
##                                                  0.000000000
##                                                     Year2016
##                                                  0.009826981
##                                                     Year2017
##                                                  0.003028990
##                                      Engine.Fuel.Typediesel
##                                                  0.177538575
##                                    Engine.Fuel.Typeelectric
##                                                  0.000000000
## Engine.Fuel.Typeflex-fuel (premium unleaded recommended/E85)
##                                                  0.000000000
##    Engine.Fuel.Typeflex-fuel (premium unleaded required/E85)
##                                                  0.000000000
##                   Engine.Fuel.Typeflex-fuel (unleaded/E85)
##                                                 -0.051763670
##                                   Engine.Fuel.Typenatural gas
##                                                  0.000000000
##                  Engine.Fuel.Typepremium unleaded (recommended)
##                                                  0.000000000
##                    Engine.Fuel.Typepremium unleaded (required)
##                                                  0.038595806
##                               Engine.Fuel.Typeregular unleaded
##                                                 -0.106918090
##                                                    Engine.HP
##                                                  0.002861934
##                                           Engine.Cylinders3
##                                                  0.000000000
##                                           Engine.Cylinders4
##                                                  0.000000000
##                                           Engine.Cylinders5
##                                                  0.000000000
```

```
##                                      Engine.Cylinders6
##                                            0.009133605
##                                      Engine.Cylinders8
##                                            0.000000000
##                                     Engine.Cylinders10
##                                            0.000000000
##                                     Engine.Cylinders12
##                                            0.000000000
##                             Transmission.TypeAUTOMATIC
##                                            0.000000000
##                          Transmission.TypeDIRECT_DRIVE
##                                            0.000000000
##                                Transmission.TypeMANUAL
##                                           -0.122870454
##                               Transmission.TypeUNKNOWN
##                                            0.000000000
##                          Driven_Wheelsfour wheel drive
##                                            0.000000000
##                         Driven_Wheelsfront wheel drive
##                                           -0.052816796
##                          Driven_Wheelsrear wheel drive
##                                           -0.011937064
##                                       Number.of.Doors3
##                                            0.000000000
##                                       Number.of.Doors4
##                                            0.000000000
##                                      Vehicle.SizeLarge
##                                            0.047321511
##                                    Vehicle.SizeMidsize
##                                            0.000000000
##                                     Vehicle.Style2dr SUV
##                                            0.000000000
##                             Vehicle.Style4dr Hatchback
##                                           -0.001632665
##                                     Vehicle.Style4dr SUV
##                                            0.071305672
##                             Vehicle.StyleCargo Minivan
##                                            0.000000000
##                                 Vehicle.StyleCargo Van
##                                            0.000000000
##                               Vehicle.StyleConvertible
##                                            0.005300364
##                           Vehicle.StyleConvertible SUV
##                                            0.000000000
##                                     Vehicle.StyleCoupe
##                                            0.000000000
##                           Vehicle.StyleCrew Cab Pickup
##                                            0.000000000
##                       Vehicle.StyleExtended Cab Pickup
##                                           -0.004516003
```

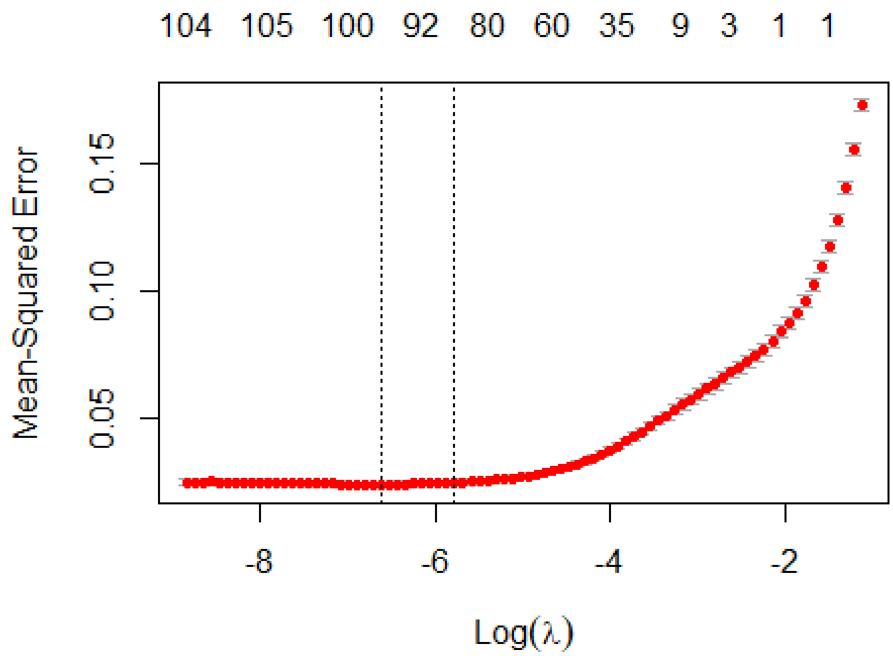
```
##                                    Vehicle.StylePassenger Minivan
##                                                       0.000243327
##                                        Vehicle.StylePassenger Van
##                                                       0.000000000
##                                   Vehicle.StyleRegular Cab Pickup
##                                                      -0.004961261
##                                               Vehicle.StyleSedan
##                                                       0.000000000
##                                               Vehicle.StyleWagon
##                                                       0.000000000
##                                                       highway.MPG
##                                                       0.000000000
##                                                          city.mpg
##                                                       0.001237789
```

```
sqrt(sum(coef(lasso_mod)[-1,100]^2)) # Calculate l1 norm. sqrt of l2 norm, ri
ght?

## [1] 2.821179
```

Notice that in the coefficient plot that depending on the choice of tuning parameter, some of the coefficients are exactly equal to zero.

Therefore, we use cross-validation to choose the tuning parameter λ . We can do this using the built-in cross-validation function, cv.glmnet(). By default, the function performs 10-fold cross-validation, though this can be changed using the argument folds. Note that we set a random seed first so our results will be reproducible, since the choice of the cross-validation folds is random.

```
set.seed(123)
cv.out = cv.glmnet(x_train, y_train, alpha = 1) # Fit lasso model on training
data


plot(cv.out) # Draw plot of training MSE as a function of lambda. At the elbo
w, my best lambda to minimize MSE is at log(-5ish)
```



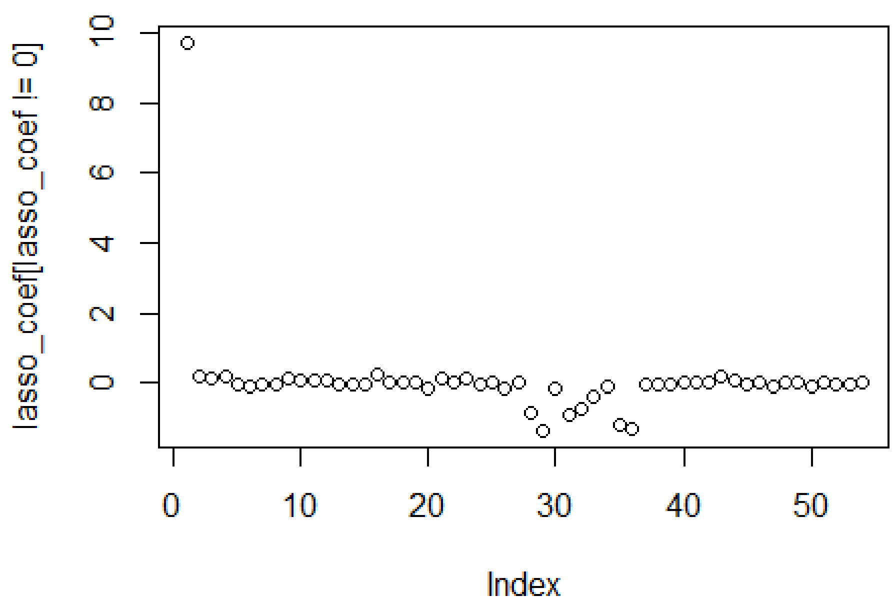

```
bestlam = cv.out$lambda.min # Select lamda that minimizes training MSE
lasso_pred = predict(lasso_mod, s = bestlam, newx = x_test) # Use best lambda
to predict test data
RMSE.lasso = sqrt(mean((lasso_pred - y_test)^2)) # Calculate test RMSE = #0.1
75

R_Squared =  1 - cv.out$cvm/var(y_train)
max(R_Squared) #= 0.864

## [1] 0.8642786

max(cv.out$glmnet.fit$dev.ratio) #= 0.88. good enough. The model can explain
88% of variation in my dataset. I used corss validation so this should be goo
d with test set.

## [1] 0.8801439

x = model.matrix(MSRP.log~., car_nonEx.new4)[,-15] # trim off the first colum
n
                                         # leaving only the predictors
y = car_nonEx.new4 %>%
  dplyr::select(MSRP.log) %>%
  unlist() %>%
  as.numeric()   #Vector for the target variable.
```

Here we see the number of coefficient estimates are exactly zero:

```
out = glmnet(x, y, alpha = 1, lambda = grid) # Fit lasso model on full datase
t
lasso_coef = predict(out, type = "coefficients", s = bestlam)[1:100,] # Displ
ay coefficients using lambda chosen by CV
```

Selecting only the predictors with non-zero coefficients, we see that the lasso model with λ chosen by cross-validation contains only 54 variables:

```
length(lasso_coef)

## [1] 100

length(lasso_coef[lasso_coef != 0]) # Display only non-zero coefficients

## [1] 54

#When 100 best was given, i got lasso_coef = 54


plot(lasso_coef[lasso_coef != 0]) # Display only non-zero coefficients
```




```
lasso_coef[lasso_coef != 0]

##                                                 (Intercept)
##                                                 9.733253872
##                                            Make_recodeAudi
##                                                 0.165285300
##                                             Make_recodeBMW
```

```
##                                                    0.123219297
##                                          Make_recodeCadillac
##                                                    0.200807909
##                                         Make_recodeChevrolet
##                                                   -0.042814961
##                                             Make_recodeDodge
##                                                   -0.114376507
##                                           Make_recodeHyundai
##                                                   -0.018196217
##                                               Make_recodeKia
##                                                   -0.047264320
##                                             Make_recodeLexus
##                                                    0.123369383
##                                           Make_recodeLincoln
##                                                    0.096809327
##                                           Make_recodeMake<60
##                                                    0.100785973
##                                     Make_recodeMercedes-Benz
##                                                    0.082975855
##                                        Make_recodeMitsubishi
##                                                   -0.025314496
##                                            Make_recodeNissan
##                                                   -0.024464029
##                                           Make_recodePontiac
##                                                   -0.035305006
##                                           Make_recodePorsche
##                                                    0.246710099
##                                              Make_recodeSaab
##                                                    0.042051657
##                                             Make_recodeScion
##                                                   -0.013308178
##                                            Make_recodeSubaru
##                                                   -0.006563362
##                                            Make_recodeSuzuki
##                                                   -0.134385537
##                                             Make_recodeVolvo
##                                                    0.124586489
##                                Veh.Style_recode4dr Hatchback
##                                                   -0.007187415
##                                  Veh.Style_recodeConvertible
##                                                    0.113148885
##                          Veh.Style_recodeExtended Cab Pickup
##                                                   -0.052587497
##                            Veh.Style_recodePassenger Minivan
##                                                    0.003531794
##                           Veh.Style_recodeRegular Cab Pickup
##                                                   -0.134250444
##                                          Veh.Style_recodeSUV
##                                                    0.033009338
##                                                     Year1992
```

```
##                                                 -0.826300319
##                                                     Year1993
##                                                 -1.366907628
##                                                     Year1994
##                                                 -0.130494509
##                                                     Year1995
##                                                 -0.926936012
##                                                     Year1996
##                                                 -0.748808315
##                                                     Year1997
##                                                 -0.371306719
##                                                     Year1998
##                                                 -0.129488021
##                                                     Year1999
##                                                 -1.184716563
##                                                     Year2000
##                                                 -1.311928880
##                                                     Year2001
##                                                 -0.023422674
##                                                     Year2002
##                                                 -0.016141716
##                                                     Year2007
##                                                 -0.058009798
##                                                     Year2008
##                                                 -0.005150472
##                                                     Year2016
##                                                  0.008379971
##                                                     Year2017
##                                                  0.002887764
##                                     Engine.Fuel.Typediesel
##                                                  0.173350815
## Engine.Fuel.Typeflex-fuel (premium unleaded required/E85)
##                                                  0.091588787
##                    Engine.Fuel.Typeflex-fuel (unleaded/E85)
##                                                 -0.061379238
##                 Engine.Fuel.Typepremium unleaded (required)
##                                                  0.037961584
##                            Engine.Fuel.Typeregular unleaded
##                                                 -0.113344214
##                                                   Engine.HP
##                                                  0.002845829
##                                          Engine.Cylinders6
##                                                  0.006996307
##                                    Transmission.TypeMANUAL
##                                                 -0.124110760
##                              Driven_Wheelsfour wheel drive
##                                                  0.006820672
##                             Driven_Wheelsfront wheel drive
##                                                 -0.057641514
##                              Driven_Wheelsrear wheel drive
```

```
##                                                       -0.016253497
##                                                  Vehicle.SizeLarge
##                                                        0.043698520
```

```
good.lasso = cbind(lasso_coef[lasso_coef != 0])# Display only non-zero coeffi
cients
good.lasso
```

```
##                                                [,1]
## (Intercept)                             9.733253872
## Make_recodeAudi                         0.165285300
## Make_recodeBMW                          0.123219297
## Make_recodeCadillac                     0.200807909
## Make_recodeChevrolet                   -0.042814961
## Make_recodeDodge                       -0.114376507
## Make_recodeHyundai                     -0.018196217
## Make_recodeKia                         -0.047264320
## Make_recodeLexus                        0.123369383
## Make_recodeLincoln                      0.096809327
## Make_recodeMake<60                      0.100785973
## Make_recodeMercedes-Benz                0.082975855
## Make_recodeMitsubishi                  -0.025314496
## Make_recodeNissan                      -0.024464029
## Make_recodePontiac                     -0.035305006
## Make_recodePorsche                      0.246710099
## Make_recodeSaab                         0.042051657
## Make_recodeScion                       -0.013308178
## Make_recodeSubaru                      -0.006563362
## Make_recodeSuzuki                      -0.134385537
## Make_recodeVolvo                        0.124586489
## Veh.Style_recode4dr Hatchback          -0.007187415
## Veh.Style_recodeConvertible             0.113148885
## Veh.Style_recodeExtended Cab Pickup    -0.052587497
## Veh.Style_recodePassenger Minivan       0.003531794
## Veh.Style_recodeRegular Cab Pickup     -0.134250444
## Veh.Style_recodeSUV                     0.033009338
## Year1992                               -0.826300319
## Year1993                               -1.366907628
## Year1994                               -0.130494509
## Year1995                               -0.926936012
## Year1996                               -0.748808315
## Year1997                               -0.371306719
## Year1998                               -0.129488021
## Year1999                               -1.184716563
## Year2000                               -1.311928880
## Year2001                               -0.023422674
## Year2002                               -0.016141716
## Year2007                               -0.058009798
## Year2008                               -0.005150472
## Year2016                                0.008379971
```

```
## Year2017                                                   0.002887764
## Engine.Fuel.Typediesel                                     0.173350815
## Engine.Fuel.Typeflex-fuel (premium unleaded required/E85)  0.091588787
## Engine.Fuel.Typeflex-fuel (unleaded/E85)                  -0.061379238
## Engine.Fuel.Typepremium unleaded (required)                0.037961584
## Engine.Fuel.Typeregular unleaded                          -0.113344214
## Engine.HP                                                  0.002845829
## Engine.Cylinders6                                          0.006996307
## Transmission.TypeMANUAL                                   -0.124110760
## Driven_Wheelsfour wheel drive                              0.006820672
## Driven_Wheelsfront wheel drive                            -0.057641514
## Driven_Wheelsrear wheel drive                             -0.016253497
## Vehicle.SizeLarge                                          0.043698520
```

```
#Write the result to a csv file for manipulation and view the result.

("C:/Users/olani/OneDrive/Documents/Data Science/SMU-Data Science/Applied Sta
tistics/Project1Details_2021/Project1Details_2021")

## [1] "C:/Users/olani/OneDrive/Documents/Data Science/SMU-Data Science/Appli
ed Statistics/Project1Details_2021/Project1Details_2021"

write.csv(good.lasso, "C:/Users/olani/OneDrive/Documents/Data Science/SMU-Dat
a Science/Applied Statistics/Project1Details_2021/Project1Details_2021/goodla
sso.csv")

goodlassos = read.csv("goodlasso_names.csv")
attach(goodlassos)
str(goodlassos)

## 'data.frame':    28 obs. of  1 variable:
##  $ Names: chr  "Make_recodeAudi" "Make_recodeBMW" "Make_recodeCadillac" "M
ake_recodeChevrolet" ...
```

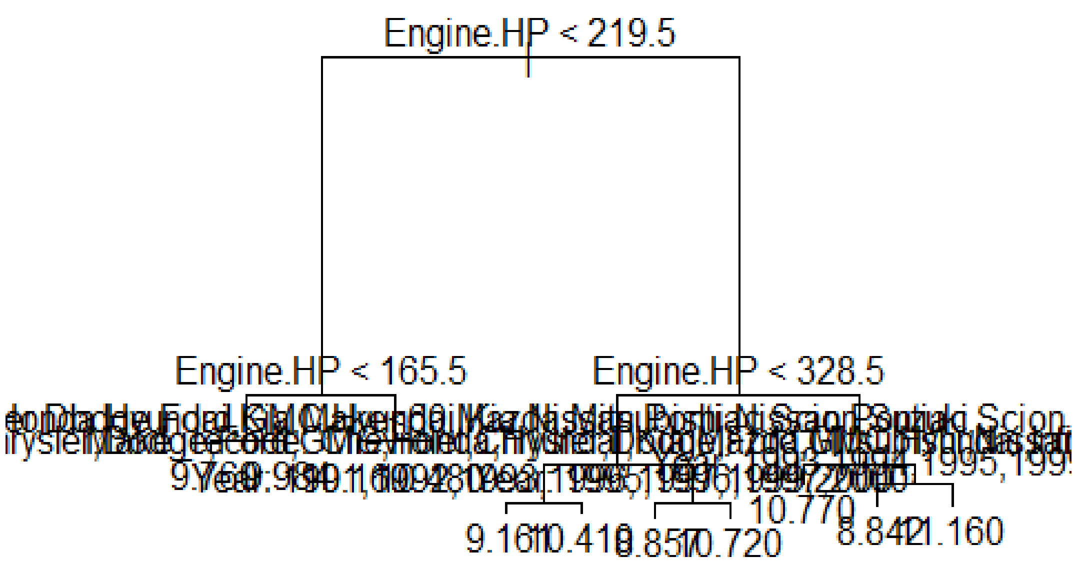
Lasso worked great. it lowered RMSE for regression a lot.

#Following feature engineering. We have a good dataset for tree.

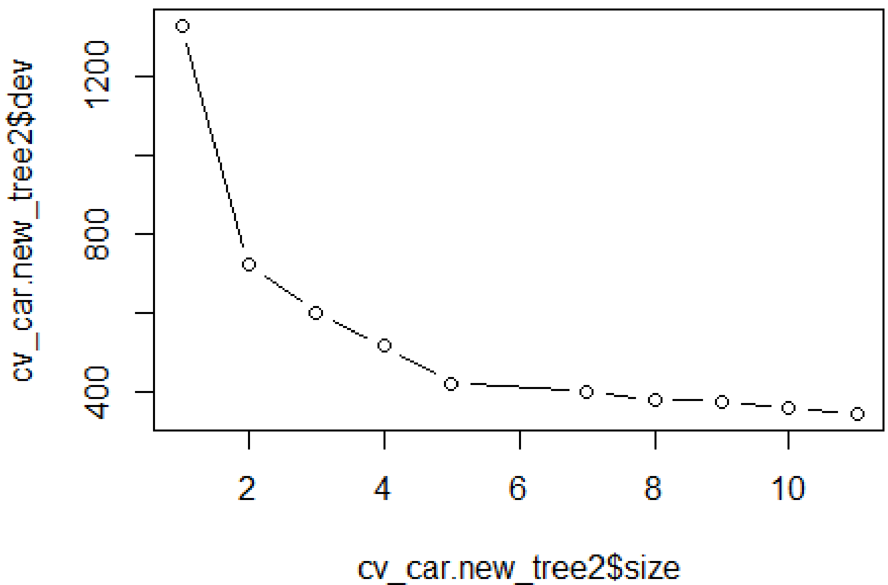
DECISION TREES MODEL

```
library(tree)

## Warning: package 'tree' was built under R version 4.0.3

## Registered S3 method overwritten by 'tree':
##   method     from
##   print.tree cli
```

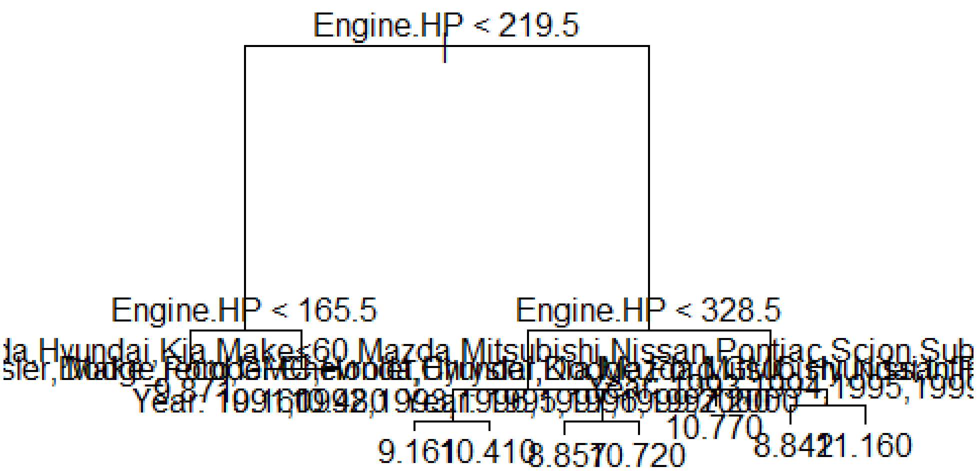
```r
set.seed(123)
#Decision tree. We allowed the tree to grow fully without any constraint and we will prune afterwards.


car.new_tree2 <- tree(MSRP.log~., data=train)

#Let us see the model
summary(car.new_tree2)

##
## Regression tree:
## tree(formula = MSRP.log ~ ., data = train)
## Variables actually used in tree construction:
## [1] "Engine.HP"    "Make_recode" "Year"
## Number of terminal nodes:  11
## Residual mean deviance:  0.03795 = 290.2 / 7646
## Distribution of residuals:
##      Min.   1st Qu.    Median      Mean   3rd Qu.      Max.
## -2.092000 -0.124500 -0.005558  0.000000  0.122400  1.090000

plot(car.new_tree2)
text(car.new_tree2, pretty=0)
```



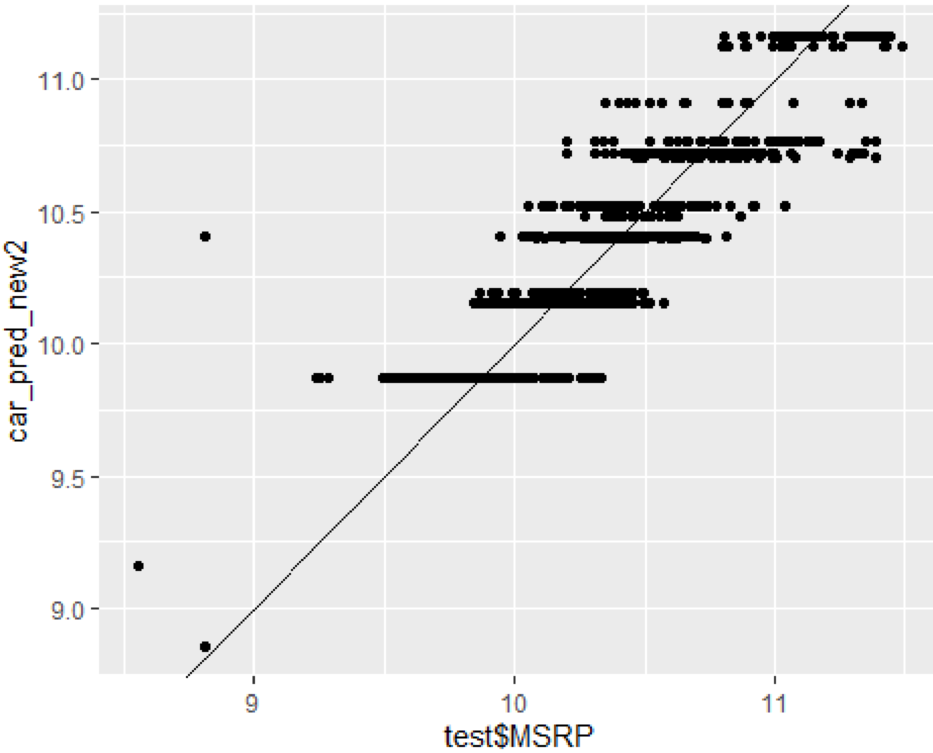
```r
#The unpruned tree found "Engine.HP"   "Make_recode" and "Year" as the most ‘important’ variables.

#Let us check if we need to prune the tree?

#We performed cross validation to estimate the cross-validated Mean Squared Error of the trees.

cv_car.new_tree2 = cv.tree(car.new_tree2)
plot(cv_car.new_tree2$size, cv_car.new_tree2$dev, type = 'b') #significant best is 10-node tree
```



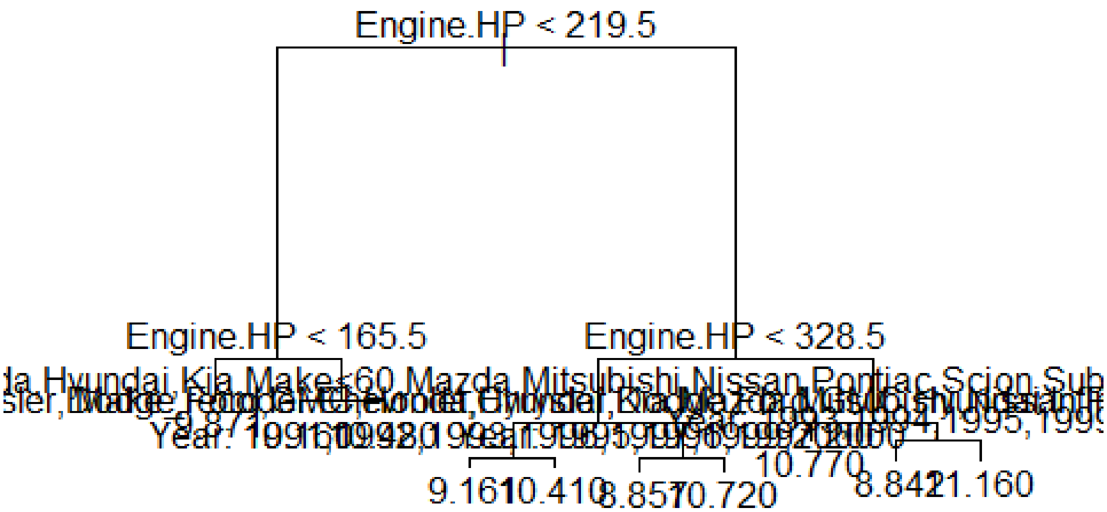

```r
#Yes we need to prune to the best 10 nodes.

The plot above indicates that 10-node tree is good enough.
prune.car.new_tree2 = prune.tree(car.new_tree2, best=10)
plot(prune.car.new_tree2)
text(prune.car.new_tree2, pretty=0)
```




```r
#let's use the pruned tree to make prediction
#prediction
car_pred_new2 <- predict(prune.car.new_tree2, newdata = test)



ggplot() +
    geom_point(aes(x = test$MSRP, y = car_pred_new2)) +
    geom_abline()
```




```
#The plot above shows the predicted value is correlated well with actual MSRP
which is a good thing. But the association is much better for higher MSRP.


Root_MSE_3 = sqrt(mean((car_pred_new2 - test$MSRP)^2)) #test RMSE at  9317 wh
ich is lower than the one obtained for linear model test RMSE.

plot(prune.car.new_tree2)
text(prune.car.new_tree2, pretty=0)
```



From the tree diagram above. It appears that the Engine.HP value is the most important variable that explains the MSRP of a vehicle. Followed by Make and then Year.

RANDOM FOREST MODEL

#1000 or more decision tree working together whereby majority decision is used to determine the value of the response.

```
library(randomForest)

## Warning: package 'randomForest' was built under R version 4.0.3

## randomForest 4.6-14

## Type rfNews() to see new features/changes/bug fixes.

##
## Attaching package: 'randomForest'
```

```
## The following object is masked from 'package:dplyr':
##
##     combine
```

```
## The following object is masked from 'package:ggplot2':
##
##     margin
```

```
set.seed(123)
car_new.rT <- randomForest(MSRP.log~., data=train, importance =TRUE, mtry=9,
ntree=1000)



#The mtry = 9 means we should use 9 predictors for each split of the tree.
```

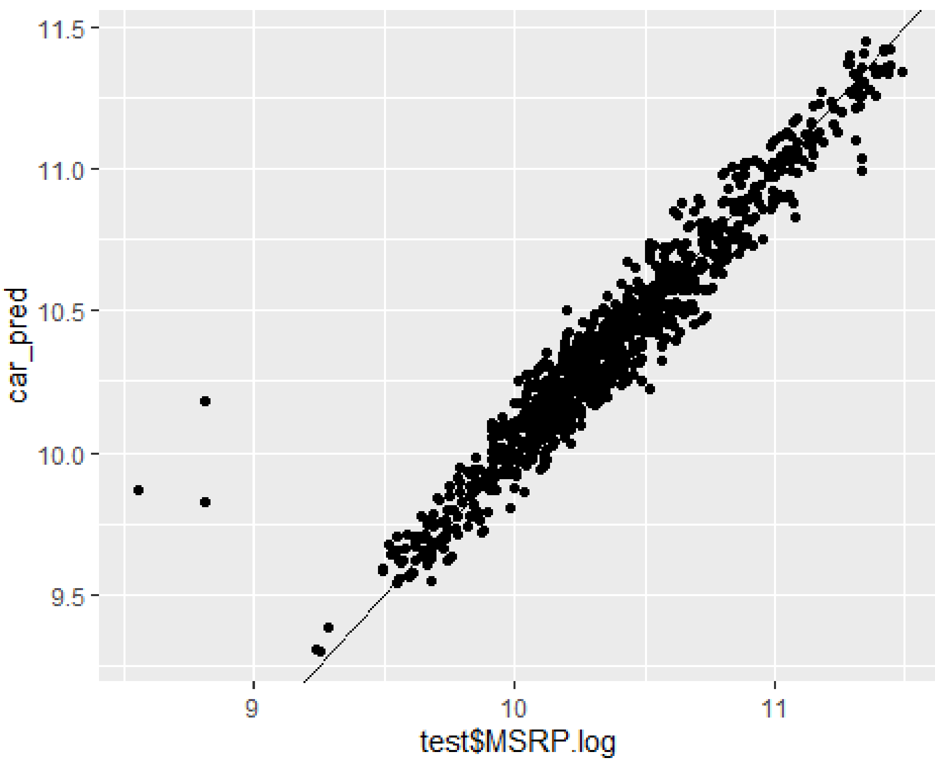
Importance=TRUE allows us to view the importance of each variable in the mode
l.

As we can see below Make, Year and Engine.HP account for most of the predicte
d MSRP of a vehicle.

```
> importance (car_new.rT)
                     %IncMSE IncNodePurity
Make_recode        209.56022    248.325918
Veh.Style_recode    63.28708     25.284694
Year               107.76909    109.678191
Engine.Fuel.Type    26.77641     35.136936
Engine.HP          234.63568    739.015771
Engine.Cylinders    14.94020     40.152504
Transmission.Type   81.12897     11.696606
Driven_Wheels       23.58750     16.511576
Number.of.Doors     27.89922      1.440921
Vehicle.Size        62.50075     16.734764
Vehicle.Style       59.20571     29.147054
highway.MPG         60.19749     14.425124
city.mpg            43.71163     15.988468
>
```

```
car_pred <- predict(car_new.rT, newdata = test)
```

```
ggplot() +
    geom_point(aes(x = test$MSRP.log, y = car_pred)) +
    geom_abline()
```




```
#The plot above offers the best correlation between predicted MSRP and actual
MSRP.Looks very strong.

Root_MSE_4 = sqrt(mean((car_pred-test$MSRP.log)^2)) #test RMSE is half of tha
t from decision tree which is very very nice.



summary(car_pred)

##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
##   9.302  10.120  10.348  10.385  10.608  11.448
```

TABULATING THE ERRORS

```
RMSE = c(Root_MSE, Root_MSE_B,  Root_MSE_B1, Root_MSE_2.train, Root_MSE_2.tes
t, RMSE.lasso,  Root_MSE_3, Root_MSE_4)
Model_type = c("Best8_Linear Full", "Best6_Linear Full", "Weighted_Linear Ful
l", "Best6_Linear Train", "Best6_Linear Test","Lasso_test RMSE", "Single Tree
_pruned", "Random Forest")

RMSE_car = cbind(Model_type, RMSE)

RMSE_car
```

```
##      Model_type             RMSE
## [1,] "Best8_Linear Full"    "0.193495275586965"
## [2,] "Best6_Linear Full"    "0.205453021109038"
## [3,] "Weighted_Linear Full" "0.206176980068281"
## [4,] "Best6_Linear Train"   "0.190171422915304"
## [5,] "Best6_Linear Test"    "0.195255029890895"
## [6,] "Lasso_test RMSE"      "0.175085056051979"
## [7,] "Single Tree_pruned Test"   "0.204902058529831"
## [8,] "Random Forest Test"        "0.117520686511865"
```

CONCLUSION

Based on the results obtained for the Root Mean Squared. Random Forest produced the lowest value and will generalize much better than the other models even though it scores low on explanability.

John's story ends here