Project 1 Group Code

Ely, Kalen, Rayon

# Libraries  
library("tidyverse")

## ── Attaching packages ─────────────────────────────────────── tidyverse 1.3.1 ──

## ✓ ggplot2 3.3.5 ✓ purrr 0.3.4  
## ✓ tibble 3.1.6 ✓ dplyr 1.0.7  
## ✓ tidyr 1.1.4 ✓ stringr 1.4.0  
## ✓ readr 2.1.1 ✓ forcats 0.5.1

## ── Conflicts ────────────────────────────────────────── tidyverse\_conflicts() ──  
## x dplyr::filter() masks stats::filter()  
## x dplyr::lag() masks stats::lag()

library("ggplot2")  
library("GGally")

## Registered S3 method overwritten by 'GGally':  
## method from   
## +.gg ggplot2

library("naniar")  
library("caret")

## Loading required package: lattice

##   
## Attaching package: 'caret'

## The following object is masked from 'package:purrr':  
##   
## lift

library("ggpubr")  
library("glmnet")

## Loading required package: Matrix

##   
## Attaching package: 'Matrix'

## The following objects are masked from 'package:tidyr':  
##   
## expand, pack, unpack

## Loaded glmnet 4.1-3

library(car)

## Loading required package: carData

## Warning: package 'carData' was built under R version 4.1.2

##   
## Attaching package: 'car'

## The following object is masked from 'package:dplyr':  
##   
## recode

## The following object is masked from 'package:purrr':  
##   
## some

#Read Data  
car\_data = read.csv("~/Desktop/applied\_stats/data1.csv")  
  
head(car\_data)

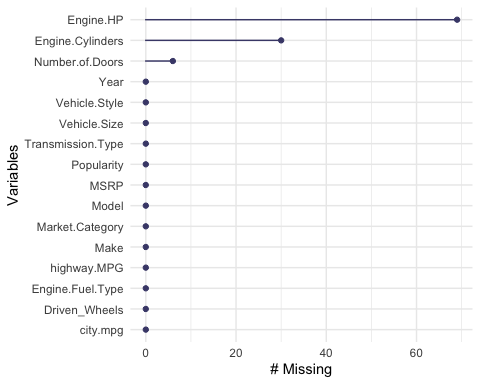
## Make Model Year Engine.Fuel.Type Engine.HP Engine.Cylinders  
## 1 BMW 1 Series M 2011 premium unleaded (required) 335 6  
## 2 BMW 1 Series 2011 premium unleaded (required) 300 6  
## 3 BMW 1 Series 2011 premium unleaded (required) 300 6  
## 4 BMW 1 Series 2011 premium unleaded (required) 230 6  
## 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

# Cleaning Data

# Change categorical attributes to factors  
  
car\_data$Make = as.factor(car\_data$Make)  
car\_data$Model = as.factor(car\_data$Model)  
car\_data$Engine.Fuel.Type = as.factor(car\_data$Engine.Fuel.Type)  
car\_data$Transmission.Type = as.factor(car\_data$Transmission.Type)  
car\_data$Driven\_Wheels = as.factor(car\_data$Driven\_Wheels)  
car\_data$Vehicle.Size = as.factor(car\_data$Vehicle.Size)  
car\_data$Market.Category = as.factor(car\_data$Market.Category)  
car\_data$Vehicle.Style = as.factor(car\_data$Vehicle.Style)  
car\_data$Engine.Cylinders = as.factor(car\_data$Engine.Cylinders)

# Evaluate if there is missing information - NA  
gg\_miss\_var(car\_data)

## Warning: It is deprecated to specify `guide = FALSE` to remove a guide. Please  
## use `guide = "none"` instead.



# Visualize the missing data. DOES NOT WORK  
  
#vis\_miss(new\_AutoM) # Where is new\_AutoM?

# Missing Doors  
  
car\_data %>%  
 filter(is.na(Number.of.Doors))

## Make Model Year Engine.Fuel.Type Engine.HP Engine.Cylinders  
## 1 Ferrari FF 2013 premium unleaded (required) 651 12  
## 2 Tesla Model S 2016 electric NA 0  
## 3 Tesla Model S 2016 electric NA 0  
## 4 Tesla Model S 2016 electric NA 0  
## 5 Tesla Model S 2016 electric NA 0  
## 6 Tesla Model S 2016 electric NA 0  
## Transmission.Type Driven\_Wheels Number.of.Doors Market.Category  
## 1 AUTOMATED\_MANUAL all wheel drive NA Exotic,High-Performance  
## 2 DIRECT\_DRIVE all wheel drive NA Exotic,Performance  
## 3 DIRECT\_DRIVE all wheel drive NA Exotic,Performance  
## 4 DIRECT\_DRIVE all wheel drive NA Exotic,High-Performance  
## 5 DIRECT\_DRIVE rear wheel drive NA Exotic,Performance  
## 6 DIRECT\_DRIVE all wheel drive NA Exotic,Performance  
## Vehicle.Size Vehicle.Style highway.MPG city.mpg Popularity MSRP  
## 1 Large Coupe 16 11 2774 295000  
## 2 Large Sedan 105 102 1391 79500  
## 3 Large Sedan 101 98 1391 66000  
## 4 Large Sedan 105 92 1391 134500  
## 5 Large Sedan 100 97 1391 74500  
## 6 Large Sedan 107 101 1391 71000

# Other Tesla Model S 2016 have 4 doors - manual fix  
car\_data %>%  
 filter(Make == 'Tesla' & Model == 'Model S' & Year == 2016)

## Make Model Year Engine.Fuel.Type Engine.HP Engine.Cylinders  
## 1 Tesla Model S 2016 electric NA 0  
## 2 Tesla Model S 2016 electric NA 0  
## 3 Tesla Model S 2016 electric NA 0  
## 4 Tesla Model S 2016 electric NA 0  
## 5 Tesla Model S 2016 electric NA 0  
## 6 Tesla Model S 2016 electric NA 0  
## 7 Tesla Model S 2016 electric NA 0  
## 8 Tesla Model S 2016 electric NA 0  
## 9 Tesla Model S 2016 electric NA 0  
## Transmission.Type Driven\_Wheels Number.of.Doors Market.Category  
## 1 DIRECT\_DRIVE all wheel drive NA Exotic,Performance  
## 2 DIRECT\_DRIVE all wheel drive NA Exotic,Performance  
## 3 DIRECT\_DRIVE all wheel drive NA Exotic,High-Performance  
## 4 DIRECT\_DRIVE rear wheel drive NA Exotic,Performance  
## 5 DIRECT\_DRIVE all wheel drive NA Exotic,Performance  
## 6 DIRECT\_DRIVE all wheel drive 4 Exotic,Performance  
## 7 DIRECT\_DRIVE all wheel drive 4 Exotic,High-Performance  
## 8 DIRECT\_DRIVE all wheel drive 4 Exotic,High-Performance  
## 9 DIRECT\_DRIVE rear wheel drive 4 Exotic,Performance  
## Vehicle.Size Vehicle.Style highway.MPG city.mpg Popularity MSRP  
## 1 Large Sedan 105 102 1391 79500  
## 2 Large Sedan 101 98 1391 66000  
## 3 Large Sedan 105 92 1391 134500  
## 4 Large Sedan 100 97 1391 74500  
## 5 Large Sedan 107 101 1391 71000  
## 6 Large Sedan 102 101 1391 75000  
## 7 Large Sedan 107 101 1391 89500  
## 8 Large Sedan 100 91 1391 112000  
## 9 Large Sedan 90 88 1391 70000

car\_data[(is.na(car\_data$Number.of.Doors)&car\_data$Make=='Tesla'),9] = 4  
  
# There are no other 2013 FF Ferraris in our data set, but the other years have 2 doors, and some online research shows thee 2013 does as well - manual fix  
car\_data %>%  
 filter(Make == 'Ferrari' & Model =='FF')

## Make Model Year Engine.Fuel.Type Engine.HP Engine.Cylinders  
## 1 Ferrari FF 2013 premium unleaded (required) 651 12  
## 2 Ferrari FF 2014 premium unleaded (required) 651 12  
## 3 Ferrari FF 2015 premium unleaded (required) 651 12  
## Transmission.Type Driven\_Wheels Number.of.Doors Market.Category  
## 1 AUTOMATED\_MANUAL all wheel drive NA Exotic,High-Performance  
## 2 AUTOMATED\_MANUAL all wheel drive 2 Exotic,High-Performance  
## 3 AUTOMATED\_MANUAL all wheel drive 2 Exotic,High-Performance  
## Vehicle.Size Vehicle.Style highway.MPG city.mpg Popularity MSRP  
## 1 Large Coupe 16 11 2774 295000  
## 2 Large Coupe 16 11 2774 295000  
## 3 Large Coupe 16 11 2774 295000

car\_data[(is.na(car\_data$Number.of.Doors)&car\_data$Make=='Ferrari'),9] = 2

# Engine Cylinders  
  
head(car\_data %>%  
 filter(is.na(Engine.Cylinders)))

## Make Model Year Engine.Fuel.Type Engine.HP Engine.Cylinders  
## 1 Chevrolet Bolt EV 2017 electric 200 <NA>  
## 2 Chevrolet Bolt EV 2017 electric 200 <NA>  
## 3 Volkswagen e-Golf 2015 electric 115 <NA>  
## 4 Volkswagen e-Golf 2015 electric 115 <NA>  
## 5 Volkswagen e-Golf 2016 electric 115 <NA>  
## 6 Volkswagen e-Golf 2016 electric 115 <NA>  
## Transmission.Type Driven\_Wheels Number.of.Doors Market.Category  
## 1 DIRECT\_DRIVE front wheel drive 4 Hatchback  
## 2 DIRECT\_DRIVE front wheel drive 4 Hatchback  
## 3 DIRECT\_DRIVE front wheel drive 4 Hatchback  
## 4 DIRECT\_DRIVE front wheel drive 4 Hatchback  
## 5 DIRECT\_DRIVE front wheel drive 4 Hatchback  
## 6 DIRECT\_DRIVE front wheel drive 4 Hatchback  
## Vehicle.Size Vehicle.Style highway.MPG city.mpg Popularity MSRP  
## 1 Compact 4dr Hatchback 110 128 1385 40905  
## 2 Compact 4dr Hatchback 110 128 1385 36620  
## 3 Compact 4dr Hatchback 105 126 873 33450  
## 4 Compact 4dr Hatchback 105 126 873 35445  
## 5 Compact 4dr Hatchback 105 126 873 28995  
## 6 Compact 4dr Hatchback 105 126 873 35595

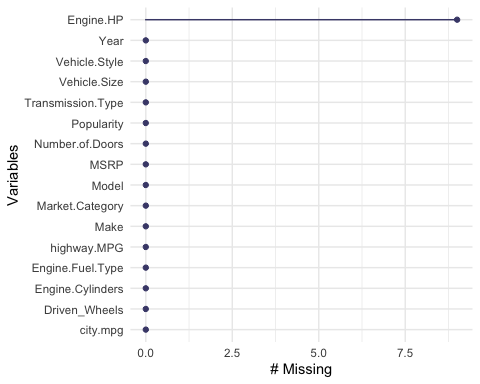
# Mazda RX Models - These have a rotary engine, so have 0 cylinders  
  
car\_data[(is.na(car\_data$Engine.Cylinders) & car\_data$Make == 'Mazda'),6]= 0  
  
# Other missing cylinder cars are all electric  
  
car\_data[(is.na(car\_data$Engine.Cylinders) & car\_data$Engine.Fuel.Type == 'electric'),6]= 0

# Engine Horsepower  
  
head(car\_data %>%  
 filter(is.na(Engine.HP)))

## Make Model Year Engine.Fuel.Type Engine.HP  
## 1 FIAT 500e 2015 electric NA  
## 2 FIAT 500e 2016 electric NA  
## 3 FIAT 500e 2017 electric NA  
## 4 Lincoln Continental 2017 premium unleaded (recommended) NA  
## 5 Lincoln Continental 2017 premium unleaded (recommended) NA  
## 6 Lincoln Continental 2017 premium unleaded (recommended) NA  
## Engine.Cylinders Transmission.Type Driven\_Wheels Number.of.Doors  
## 1 0 DIRECT\_DRIVE front wheel drive 2  
## 2 0 DIRECT\_DRIVE front wheel drive 2  
## 3 0 DIRECT\_DRIVE front wheel drive 2  
## 4 6 AUTOMATIC all wheel drive 4  
## 5 6 AUTOMATIC front wheel drive 4  
## 6 6 AUTOMATIC front wheel drive 4  
## Market.Category Vehicle.Size Vehicle.Style highway.MPG city.mpg Popularity  
## 1 Hatchback Compact 2dr Hatchback 108 122 819  
## 2 Hatchback Compact 2dr Hatchback 103 121 819  
## 3 Hatchback Compact 2dr Hatchback 103 121 819  
## 4 Luxury Large Sedan 25 17 61  
## 5 Luxury Large Sedan 27 18 61  
## 6 Luxury Large Sedan 27 18 61  
## MSRP  
## 1 31800  
## 2 31800  
## 3 31800  
## 4 55915  
## 5 62915  
## 6 53915

# Using the base horsepower I can find online for these Some versions have higher horsepower, but this data set does not give us the attributes needed to identify these.  
  
car\_data[(is.na(car\_data$Engine.HP) & car\_data$Model=='Model S' & car\_data$Year == 2014),5] = 302  
car\_data[(is.na(car\_data$Engine.HP) & car\_data$Model=='Model S' & car\_data$Year == 2015),5] = 329  
car\_data[(is.na(car\_data$Engine.HP) & car\_data$Model=='Model S' & car\_data$ear == 2016),5] = 315  
car\_data[(is.na(car\_data$Engine.HP) & car\_data$Model=='500e' & car\_data$Year == 2015),5] = 111  
car\_data[(is.na(car\_data$Engine.HP) & car\_data$Model=='500e' & car\_data$Year == 2016),5] = 111  
car\_data[(is.na(car\_data$Engine.HP) & car\_data$Model=='500e' & car\_data$Year == 2017),5] = 111  
car\_data[(is.na(car\_data$Engine.HP) & car\_data$Model=='Continental' & car\_data$Year == 2017),5] =305  
car\_data[(is.na(car\_data$Engine.HP) & car\_data$Model=='Escape' & car\_data$Year == 2017),5] = 168  
car\_data[(is.na(car\_data$Engine.HP) & car\_data$Model=='Fit EV' & car\_data$Year == 2013),5] = 123  
car\_data[(is.na(car\_data$Engine.HP) & car\_data$Model=='Fit EV' & car\_data$Year == 2014),5] = 123  
car\_data[(is.na(car\_data$Engine.HP) & car\_data$Model=='Focus' & car\_data$Year == 2015),5] = 123  
car\_data[(is.na(car\_data$Engine.HP) & car\_data$Model=='Focus' & car\_data$Year == 2016),5] = 123  
car\_data[(is.na(car\_data$Engine.HP) & car\_data$Model=='Focus' & car\_data$Year == 2017),5] = 123  
car\_data[(is.na(car\_data$Engine.HP) & car\_data$Model=='Freestar' & car\_data$Year == 2005),5] = 193  
car\_data[(is.na(car\_data$Engine.HP) & car\_data$Model=='i-MiEV' & car\_data$Year == 2014),5] = 66  
car\_data[(is.na(car\_data$Engine.HP) & car\_data$Model=='Impala' & car\_data$Year == 2015),5] = 195  
car\_data[(is.na(car\_data$Engine.HP) & car\_data$Model=='Impala' & car\_data$Year == 2016),5] = 196  
car\_data[(is.na(car\_data$Engine.HP) & car\_data$Model=='Impala' & car\_data$Year == 2017),5] = 197  
car\_data[(is.na(car\_data$Engine.HP) & car\_data$Model=='Leaf' & car\_data$Year == 2014),5] = 107  
car\_data[(is.na(car\_data$Engine.HP) & car\_data$Model=='Leaf' & car\_data$Year == 2015),5] = 107  
car\_data[(is.na(car\_data$Engine.HP) & car\_data$Model=='Leaf' & car\_data$Year == 2016),5] = 107  
car\_data[(is.na(car\_data$Engine.HP) & car\_data$Model=='M-Class' & car\_data$Year == 2015),5] = 302  
car\_data[(is.na(car\_data$Engine.HP) & car\_data$Model=='MKZ' & car\_data$Year == 2017),5] = 240  
car\_data[(is.na(car\_data$Engine.HP) & car\_data$Model=='RAV4 EV' & car\_data$Year == 2013),5] = 154  
car\_data[(is.na(car\_data$Engine.HP) & car\_data$Model=='RAV4 EV' & car\_data$Year == 2014),5] = 154  
car\_data[(is.na(car\_data$Engine.HP) & car\_data$Model=='Soul EV' & car\_data$Year == 2015),5] = 109  
car\_data[(is.na(car\_data$Engine.HP) & car\_data$Model=='Soul EV' & car\_data$Year == 2016),5] = 109  
  
  
  
  
# Double Check NA Values - Looks good!  
gg\_miss\_var(car\_data)

## Warning: It is deprecated to specify `guide = FALSE` to remove a guide. Please  
## use `guide = "none"` instead.



sapply(car\_data, function(x) sum(is.na(x)))

## Make Model Year Engine.Fuel.Type   
## 0 0 0 0   
## Engine.HP Engine.Cylinders Transmission.Type Driven\_Wheels   
## 9 0 0 0   
## Number.of.Doors Market.Category Vehicle.Size Vehicle.Style   
## 0 0 0 0   
## highway.MPG city.mpg Popularity MSRP   
## 0 0 0 0

# Unnown Transmission Types  
  
head(car\_data %>%  
 filter(Transmission.Type == 'UNKNOWN'))

## Make Model Year Engine.Fuel.Type Engine.HP Engine.Cylinders  
## 1 Oldsmobile Achieva 1997 regular unleaded 150 4  
## 2 Oldsmobile Achieva 1997 regular unleaded 150 4  
## 3 Pontiac Firebird 2000 regular unleaded 305 8  
## 4 Pontiac Firebird 2000 regular unleaded 305 8  
## 5 Pontiac Firebird 2000 regular unleaded 305 8  
## 6 GMC Jimmy 1999 regular unleaded 190 6  
## Transmission.Type Driven\_Wheels Number.of.Doors  
## 1 UNKNOWN front wheel drive 2  
## 2 UNKNOWN front wheel drive 4  
## 3 UNKNOWN rear wheel drive 2  
## 4 UNKNOWN rear wheel drive 2  
## 5 UNKNOWN rear wheel drive 2  
## 6 UNKNOWN rear wheel drive 2  
## Market.Category Vehicle.Size Vehicle.Style highway.MPG  
## 1 N/A Midsize Coupe 29  
## 2 N/A Midsize Sedan 29  
## 3 Hatchback,Performance Midsize 2dr Hatchback 23  
## 4 Hatchback,Factory Tuner,Performance Midsize 2dr Hatchback 23  
## 5 Factory Tuner,Performance Midsize Convertible 23  
## 6 N/A Compact 2dr SUV 19  
## city.mpg Popularity MSRP  
## 1 19 26 2000  
## 2 19 26 2000  
## 3 15 210 6175  
## 4 15 210 8548  
## 5 15 210 9567  
## 6 14 549 2182

# Based off of other RAM 150s  
car\_data[(car\_data$Transmission.Type == 'UNKNOWN' & car\_data$Model=='RAM 150'),7] = 'MANUAL'  
  
# Based off of online research  
car\_data[(car\_data$Transmission.Type == 'UNKNOWN' & car\_data$Model=='Achieva'),7] = 'AUTOMATIC'  
  
# Other 3 models with unknown transmissions don't have enough identifiable info to fix, so we'll drop these 9 rows.  
  
car\_data = car\_data %>%  
 filter(Transmission.Type != 'UNKNOWN')  
  
# Refactor transmission type to exclude UNKNOWN  
car\_data$Transmission.Type = as.character(car\_data$Transmission.Type)  
car\_data$Transmission.Type = as.factor(car\_data$Transmission.Type)

summary(car\_data)

## Make Model Year   
## Chevrolet :1123 Silverado 1500 : 156 Min. :1990   
## Ford : 881 Tundra : 140 1st Qu.:2007   
## Volkswagen: 809 F-150 : 126 Median :2015   
## Toyota : 746 Sierra 1500 : 90 Mean :2010   
## Dodge : 626 Beetle Convertible: 89 3rd Qu.:2016   
## Nissan : 558 Tacoma : 80 Max. :2017   
## (Other) :7162 (Other) :11224   
## Engine.Fuel.Type Engine.HP Engine.Cylinders  
## regular unleaded :7163 Min. : 55.0 4 :4750   
## premium unleaded (required) :2009 1st Qu.: 170.0 6 :4485   
## premium unleaded (recommended):1523 Median : 227.0 8 :2028   
## flex-fuel (unleaded/E85) : 899 Mean : 249.1 12 : 230   
## diesel : 154 3rd Qu.: 300.0 5 : 225   
## electric : 66 Max. :1001.0 0 : 86   
## (Other) : 91 NA's :9 (Other): 101   
## Transmission.Type Driven\_Wheels Number.of.Doors  
## AUTOMATED\_MANUAL: 626 all wheel drive :2353 Min. :2.000   
## AUTOMATIC :8268 four wheel drive :1401 1st Qu.:2.000   
## DIRECT\_DRIVE : 68 front wheel drive:4785 Median :4.000   
## MANUAL :2943 rear wheel drive :3366 Mean :3.437   
## 3rd Qu.:4.000   
## Max. :4.000   
##   
## Market.Category Vehicle.Size Vehicle.Style   
## N/A :3736 Compact:4758 Sedan :3048   
## Crossover :1110 Large :2777 4dr SUV :2488   
## Flex Fuel : 872 Midsize:4370 Coupe :1210   
## Luxury : 855 Convertible : 791   
## Luxury,Performance: 673 4dr Hatchback : 702   
## Hatchback : 641 Crew Cab Pickup: 681   
## (Other) :4018 (Other) :2985   
## highway.MPG city.mpg Popularity MSRP   
## Min. : 12.00 Min. : 7.00 Min. : 2 Min. : 2000   
## 1st Qu.: 22.00 1st Qu.: 16.00 1st Qu.: 549 1st Qu.: 21040   
## Median : 26.00 Median : 18.00 Median :1385 Median : 29995   
## Mean : 26.64 Mean : 19.74 Mean :1556 Mean : 40622   
## 3rd Qu.: 30.00 3rd Qu.: 22.00 3rd Qu.:2009 3rd Qu.: 42250   
## Max. :354.00 Max. :137.00 Max. :5657 Max. :2065902   
##

# Seeing odd values for max HP, Engine Cylinders,MPG, and MSRP  
  
# HP and MSRP looks correct  
head(car\_data %>%  
 filter(Engine.HP > 700))

## Make Model Year Engine.Fuel.Type Engine.HP  
## 1 Lamborghini Aventador 2014 premium unleaded (required) 720  
## 2 Lamborghini Aventador 2014 premium unleaded (required) 720  
## 3 Lamborghini Aventador 2015 premium unleaded (required) 720  
## 4 Lamborghini Aventador 2015 premium unleaded (required) 720  
## 5 Lamborghini Aventador 2016 premium unleaded (required) 750  
## 6 Lamborghini Aventador 2016 premium unleaded (required) 750  
## Engine.Cylinders Transmission.Type Driven\_Wheels Number.of.Doors  
## 1 12 AUTOMATED\_MANUAL all wheel drive 2  
## 2 12 AUTOMATED\_MANUAL all wheel drive 2  
## 3 12 AUTOMATED\_MANUAL all wheel drive 2  
## 4 12 AUTOMATED\_MANUAL all wheel drive 2  
## 5 12 AUTOMATED\_MANUAL all wheel drive 2  
## 6 12 AUTOMATED\_MANUAL all wheel drive 2  
## Market.Category Vehicle.Size Vehicle.Style highway.MPG city.mpg  
## 1 Exotic,High-Performance Midsize Convertible 16 10  
## 2 Exotic,High-Performance Midsize Coupe 18 11  
## 3 Exotic,High-Performance Midsize Convertible 16 10  
## 4 Exotic,High-Performance Midsize Coupe 18 11  
## 5 Exotic,High-Performance Midsize Convertible 18 11  
## 6 Exotic,High-Performance Midsize Coupe 18 11  
## Popularity MSRP  
## 1 1158 548800  
## 2 1158 497650  
## 3 1158 548800  
## 4 1158 497650  
## 5 1158 535500  
## 6 1158 490700

# Cylinders looks correct based on car and HP  
head(car\_data %>%  
 filter(Engine.Cylinders > 10))

## Warning in Ops.factor(Engine.Cylinders, 10): '>' not meaningful for factors

## [1] Make Model Year Engine.Fuel.Type   
## [5] Engine.HP Engine.Cylinders Transmission.Type Driven\_Wheels   
## [9] Number.of.Doors Market.Category Vehicle.Size Vehicle.Style   
## [13] highway.MPG city.mpg Popularity MSRP   
## <0 rows> (or 0-length row.names)

# Highway MPG for this item is false - manual fix based on online research  
head(car\_data %>%  
 filter(highway.MPG > 300))

## Make Model Year Engine.Fuel.Type Engine.HP Engine.Cylinders  
## 1 Audi A6 2017 premium unleaded (recommended) 252 4  
## Transmission.Type Driven\_Wheels Number.of.Doors Market.Category  
## 1 AUTOMATED\_MANUAL front wheel drive 4 Luxury  
## Vehicle.Size Vehicle.Style highway.MPG city.mpg Popularity MSRP  
## 1 Midsize Sedan 354 24 3105 51600

car\_data[car\_data$highway.MPG == 354,13] = 32  
  
# All high city mpgs are electric, as expected  
head(car\_data %>%  
 filter(city.mpg > 100))

## Make Model Year Engine.Fuel.Type Engine.HP Engine.Cylinders  
## 1 FIAT 500e 2015 electric 111 0  
## 2 FIAT 500e 2016 electric 111 0  
## 3 FIAT 500e 2017 electric 111 0  
## 4 Chevrolet Bolt EV 2017 electric 200 0  
## 5 Chevrolet Bolt EV 2017 electric 200 0  
## 6 Volkswagen e-Golf 2015 electric 115 0  
## Transmission.Type Driven\_Wheels Number.of.Doors Market.Category  
## 1 DIRECT\_DRIVE front wheel drive 2 Hatchback  
## 2 DIRECT\_DRIVE front wheel drive 2 Hatchback  
## 3 DIRECT\_DRIVE front wheel drive 2 Hatchback  
## 4 DIRECT\_DRIVE front wheel drive 4 Hatchback  
## 5 DIRECT\_DRIVE front wheel drive 4 Hatchback  
## 6 DIRECT\_DRIVE front wheel drive 4 Hatchback  
## Vehicle.Size Vehicle.Style highway.MPG city.mpg Popularity MSRP  
## 1 Compact 2dr Hatchback 108 122 819 31800  
## 2 Compact 2dr Hatchback 103 121 819 31800  
## 3 Compact 2dr Hatchback 103 121 819 31800  
## 4 Compact 4dr Hatchback 110 128 1385 40905  
## 5 Compact 4dr Hatchback 110 128 1385 36620  
## 6 Compact 4dr Hatchback 105 126 873 33450

# N/A Strings in Market Category  
  
# These N/A Values range over a wide variety of car types. We'll choose to keep these values in the data set unless we need to create a model using market category  
  
head(  
 car\_data %>%  
 filter(Market.Category == 'N/A')  
 )

## Make Model Year Engine.Fuel.Type Engine.HP Engine.Cylinders  
## 1 Nissan 200SX 1996 regular unleaded 115 4  
## 2 Nissan 200SX 1996 regular unleaded 115 4  
## 3 Nissan 200SX 1997 regular unleaded 115 4  
## 4 Nissan 200SX 1997 regular unleaded 115 4  
## 5 Nissan 200SX 1998 regular unleaded 115 4  
## 6 Nissan 200SX 1998 regular unleaded 115 4  
## Transmission.Type Driven\_Wheels Number.of.Doors Market.Category  
## 1 MANUAL front wheel drive 2 N/A  
## 2 MANUAL front wheel drive 2 N/A  
## 3 MANUAL front wheel drive 2 N/A  
## 4 MANUAL front wheel drive 2 N/A  
## 5 MANUAL front wheel drive 2 N/A  
## 6 MANUAL front wheel drive 2 N/A  
## Vehicle.Size Vehicle.Style highway.MPG city.mpg Popularity MSRP  
## 1 Compact Coupe 36 26 2009 2000  
## 2 Compact Coupe 36 26 2009 2000  
## 3 Compact Coupe 35 25 2009 2000  
## 4 Compact Coupe 35 25 2009 2000  
## 5 Compact Coupe 35 25 2009 2000  
## 6 Compact Coupe 35 25 2009 2000

# Accounting for duplicate entries with differing MSRP values

# Collaborated with Braden on this code chunk.  
  
avg\_multiple\_priced\_duplicates <- function(df){  
   
 count <- 0  
   
 # Create a copy of the dataframe to edit  
 new\_df <- df  
   
 for(index in 1:nrow(df)){  
   
 # Grab a dataframe row, which we will use to search for rows that are  
 # exactly the same, other than having a different MSRP  
 df\_row <- df[index,]  
   
 # Get a set of rows that are the same as this one (other than different MSRP)  
 filtered\_data <- filter\_by\_example(df=df, row\_example=df\_row)  
   
 # Check how many rows matched df\_row  
 num\_matching\_rows <- nrow(filtered\_data)  
   
 # If more than one (itself) matched... there are duplicates  
 if(num\_matching\_rows > 1){  
   
 # Get the rownames for the duplicates  
 row\_names <- rownames(filtered\_data)  
   
 # Grab the first row name, we will keep this one (arbitrary) and drop the others  
 first\_row\_name <- row\_names[1]  
   
 # Calculate the average price for these duplicate rows  
 average\_price <- mean(filtered\_data[,"MSRP"])  
   
 # Set the MSRP for the first instance of these duplicates to the average  
 new\_df[rownames(new\_df) == first\_row\_name, "MSRP"] <- average\_price  
   
 # Remove the rest of the duplicates  
 removal\_row\_names <- row\_names[row\_names != first\_row\_name]  
 new\_df <- new\_df[!(rownames(new\_df) %in%removal\_row\_names),]  
   
 # Just for keeping track of how many things we remove, in case of troubleshooting.  
 count <- count + 1  
   
 }  
 }  
   
 return(new\_df)  
}  
  
filter\_by\_example <- function(df, row\_example){  
   
 filtered\_df <- filter\_all\_columns(df=df,  
 Make=row\_example$Make,   
 Model=row\_example$Model,   
 Year=row\_example$Year,  
 Engine.Fuel.Type=row\_example$Engine.Fuel.Type,  
 Engine.HP=row\_example$Engine.HP,  
 Engine.Cylinders=row\_example$Engine.Cylinders,  
 Transmission.Type=row\_example$Transmission.Type,  
 Driven\_Wheels=row\_example$Driven\_Wheels,  
 Number.of.Doors=row\_example$Number.of.Doors,  
 Market.Category=row\_example$Market.Category,  
 Vehicle.Size=row\_example$Vehicle.Size,  
 Vehicle.Style=row\_example$Vehicle.Style,  
 highway.MPG=row\_example$highway.MPG,  
 city.mpg=row\_example$city.mpg,  
 Popularity=row\_example$Popularity)  
   
 return(filtered\_df)  
   
}  
  
filter\_all\_columns <- function(df, Make, Model, Year, Engine.Fuel.Type, Engine.HP, Engine.Cylinders,   
 Transmission.Type, Driven\_Wheels, Number.of.Doors, Market.Category, Vehicle.Size,   
 Vehicle.Style, highway.MPG, city.mpg, Popularity){  
   
   
 f1 <- df[,"Make"] == Make  
 f2 <- df[,"Model"] == Model  
 f3 <- df[,"Year"] == Year  
 f4 <- df[,"Engine.Fuel.Type"] == Engine.Fuel.Type  
 f5 <- df[,"Engine.HP"] == Engine.HP  
 f6 <- df[,"Engine.Cylinders"] == Engine.Cylinders  
 f7 <- df[,"Transmission.Type"] == Transmission.Type  
 f8 <- df[,"Driven\_Wheels"] == Driven\_Wheels  
 f9 <- df[,"Number.of.Doors"] == Number.of.Doors  
 f10 <- df[,"Market.Category"] == Market.Category  
 f11 <- df[,"Vehicle.Size"] == Vehicle.Size  
 f12 <- df[,"Vehicle.Style"] == Vehicle.Style  
 f13 <- df[,"highway.MPG"] == highway.MPG  
 f14 <- df[,"city.mpg"] == city.mpg  
 f15 <- df[,"Popularity"] == Popularity  
   
 full\_filter <- (f1& f2& f3 & f4 & f5 & f6 & f7 & f8 & f9 & f10 & f11 & f12 & f13 & f14 & f15)  
   
 filtered\_df <- df[full\_filter,]  
   
 return(filtered\_df)  
   
}  
  
  
car\_data1 <- avg\_multiple\_priced\_duplicates(df=car\_data)

# Removing Electric Vehicles and outlier MSRP values

car\_data\_clean = car\_data1 %>%  
 filter(Engine.Fuel.Type != 'electric') %>%  
 filter(MSRP > 10000) %>%  
 filter(MSRP < 150000)

# Transforming Response and Predictor Variables

car\_data\_clean =   
 car\_data\_clean %>%  
 mutate(logYear = log(Year)) %>%  
 mutate(logEngineHP = log(Engine.HP)) %>%  
 mutate(logMSRP = log(MSRP)) %>%  
 mutate(loghighwayMPG = log(highway.MPG)) %>%  
 mutate(logcitympg = log(city.mpg)) %>%  
 mutate(sqrtMSRP=sqrt(MSRP)) %>%   
 mutate(invertedMSRP=1/MSRP) %>%   
 mutate(Observation = n()) # Created to merge the data set to find leverage points

# Training / Test Split

rows = count(car\_data\_clean)[1]  
train\_rows = as.numeric(round(0.8\*rows,0))  
test\_validate\_rows = as.numeric((rows - train\_rows)/2)  
  
set.seed(1234)  
index = sample(1:dim(car\_data\_clean)[1],train\_rows,replace=F)  
  
train = car\_data\_clean[index,]  
test\_validate = car\_data\_clean[-index,]  
test\_index = sample(1:dim(test\_validate)[1],test\_validate\_rows,replace=F)  
  
train = car\_data\_clean[index,]  
test = test\_validate[test\_index,]  
validate = test\_validate[-test\_index,]  
  
head(train)

## Make Model Year Engine.Fuel.Type Engine.HP  
## 1004 Volkswagen CC 2015 premium unleaded (recommended) 200  
## 623 Nissan Armada 2015 regular unleaded 317  
## 2693 Nissan Juke 2016 premium unleaded (required) 215  
## 934 Toyota Camry 2017 regular unleaded 268  
## 4496 Chevrolet TrailBlazer 2007 regular unleaded 291  
## 2948 Acura MDX 2015 premium unleaded (recommended) 290  
## Engine.Cylinders Transmission.Type Driven\_Wheels Number.of.Doors  
## 1004 4 AUTOMATED\_MANUAL front wheel drive 4  
## 623 8 AUTOMATIC rear wheel drive 4  
## 2693 4 AUTOMATIC all wheel drive 4  
## 934 6 AUTOMATIC front wheel drive 4  
## 4496 6 AUTOMATIC rear wheel drive 4  
## 2948 6 AUTOMATIC all wheel drive 4  
## Market.Category Vehicle.Size Vehicle.Style  
## 1004 Performance Midsize Sedan  
## 623 N/A Large 4dr SUV  
## 2693 Crossover,Hatchback,Factory Tuner,Performance Compact 4dr Hatchback  
## 934 Performance Midsize Sedan  
## 4496 N/A Midsize 4dr SUV  
## 2948 Crossover,Luxury Midsize 4dr SUV  
## highway.MPG city.mpg Popularity MSRP logYear logEngineHP logMSRP  
## 1004 31 22 873 35851.67 7.608374 5.298317 10.48715  
## 623 19 13 2009 45692.50 7.608374 5.758902 10.72969  
## 2693 29 25 2009 30020.00 7.608871 5.370638 10.30962  
## 934 30 21 2031 31370.00 7.609367 5.590987 10.35361  
## 4496 20 14 1385 26507.50 7.604396 5.673323 10.18518  
## 2948 27 18 204 50256.25 7.608374 5.669881 10.82489  
## loghighwayMPG logcitympg sqrtMSRP invertedMSRP Observation  
## 1004 3.433987 3.091042 189.3454 2.789271e-05 5050  
## 623 2.944439 2.564949 213.7580 2.188543e-05 5050  
## 2693 3.367296 3.218876 173.2628 3.331113e-05 5050  
## 934 3.401197 3.044522 177.1158 3.187759e-05 5050  
## 4496 2.995732 2.639057 162.8112 3.772517e-05 5050  
## 2948 3.295837 2.890372 224.1791 1.989802e-05 5050

head(test)

## Make Model Year Engine.Fuel.Type Engine.HP  
## 2141 Infiniti G Convertible 2011 premium unleaded (required) 325  
## 3698 Acura RSX 2005 regular unleaded 160  
## 4326 Toyota Tacoma 2016 regular unleaded 278  
## 1109 Mercedes-Benz CLK-Class 2007 premium unleaded (required) 475  
## 4440 Acura TLX 2016 premium unleaded (recommended) 290  
## 4359 Ford Taurus X 2009 regular unleaded 263  
## Engine.Cylinders Transmission.Type Driven\_Wheels Number.of.Doors  
## 2141 6 AUTOMATIC rear wheel drive 2  
## 3698 4 MANUAL front wheel drive 2  
## 4326 6 MANUAL four wheel drive 4  
## 1109 8 AUTOMATIC rear wheel drive 2  
## 4440 6 AUTOMATIC front wheel drive 4  
## 4359 6 AUTOMATIC all wheel drive 4  
## Market.Category Vehicle.Size Vehicle.Style  
## 2141 Luxury,Performance Midsize Convertible  
## 3698 Hatchback,Luxury,Performance Compact 2dr Hatchback  
## 4326 N/A Compact Crew Cab Pickup  
## 1109 Factory Tuner,Luxury,High-Performance Compact Convertible  
## 4440 Luxury Midsize Sedan  
## 4359 Crossover Large Wagon  
## highway.MPG city.mpg Popularity MSRP logYear logEngineHP logMSRP  
## 2141 25 17 190 58000.00 7.606387 5.783825 10.968198  
## 3698 31 24 204 20812.50 7.603399 5.075174 9.943309  
## 4326 20 17 2031 32460.00 7.608871 5.627621 10.387764  
## 1109 18 12 617 89200.00 7.604396 6.163315 11.398636  
## 4440 34 21 204 39098.33 7.608871 5.669881 10.573835  
## 4359 22 15 5657 32491.67 7.605392 5.572154 10.388739  
## loghighwayMPG logcitympg sqrtMSRP invertedMSRP Observation  
## 2141 3.218876 2.833213 240.8319 1.724138e-05 5050  
## 3698 3.433987 3.178054 144.2654 4.804805e-05 5050  
## 4326 2.995732 2.833213 180.1666 3.080715e-05 5050  
## 1109 2.890372 2.484907 298.6637 1.121076e-05 5050  
## 4440 3.526361 3.044522 197.7330 2.557654e-05 5050  
## 4359 3.091042 2.708050 180.2544 3.077712e-05 5050

head(validate)

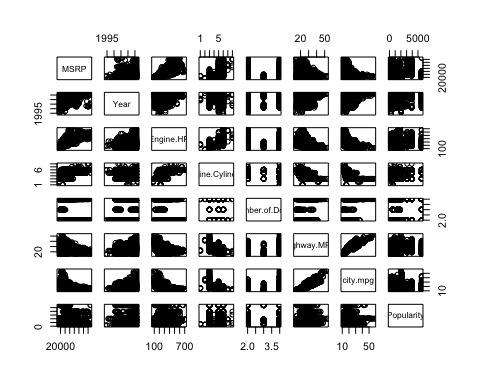
## Make Model Year Engine.Fuel.Type Engine.HP  
## 6 BMW 1 Series 2012 premium unleaded (required) 230  
## 18 BMW 2 Series 2016 premium unleaded (required) 240  
## 29 BMW 2 Series 2017 premium unleaded (recommended) 335  
## 48 BMW 3 Series Gran Turismo 2015 premium unleaded (required) 300  
## 60 BMW 3 Series 2015 premium unleaded (required) 180  
## 70 BMW 3 Series 2016 premium unleaded (required) 240  
## Engine.Cylinders Transmission.Type Driven\_Wheels Number.of.Doors  
## 6 6 MANUAL rear wheel drive 2  
## 18 4 AUTOMATIC rear wheel drive 2  
## 29 6 AUTOMATIC all wheel drive 2  
## 48 6 AUTOMATIC all wheel drive 4  
## 60 4 AUTOMATIC all wheel drive 4  
## 70 4 MANUAL rear wheel drive 4  
## Market.Category Vehicle.Size Vehicle.Style highway.MPG  
## 6 Luxury,Performance Compact Coupe 28  
## 18 Luxury,Performance Compact Coupe 35  
## 29 Factory Tuner,Luxury,High-Performance Compact Convertible 32  
## 48 Hatchback,Luxury,Performance Midsize 4dr Hatchback 30  
## 60 Luxury Midsize Sedan 35  
## 70 Luxury,Performance Midsize Sedan 34  
## city.mpg Popularity MSRP logYear logEngineHP logMSRP loghighwayMPG  
## 6 18 3916 31200 7.606885 5.438079 10.34817 3.332205  
## 18 23 3916 32850 7.608871 5.480639 10.39971 3.555348  
## 29 21 3916 51050 7.609367 5.814131 10.84056 3.465736  
## 48 20 3916 47250 7.608374 5.703782 10.76321 3.401197  
## 60 23 3916 34950 7.608374 5.192957 10.46167 3.555348  
## 70 22 3916 38350 7.608871 5.480639 10.55451 3.526361  
## logcitympg sqrtMSRP invertedMSRP Observation  
## 6 2.890372 176.6352 3.205128e-05 5050  
## 18 3.135494 181.2457 3.044140e-05 5050  
## 29 3.044522 225.9425 1.958864e-05 5050  
## 48 2.995732 217.3707 2.116402e-05 5050  
## 60 3.135494 186.9492 2.861230e-05 5050  
## 70 3.091042 195.8316 2.607562e-05 5050

# EDA

head(train)

## Make Model Year Engine.Fuel.Type Engine.HP  
## 1004 Volkswagen CC 2015 premium unleaded (recommended) 200  
## 623 Nissan Armada 2015 regular unleaded 317  
## 2693 Nissan Juke 2016 premium unleaded (required) 215  
## 934 Toyota Camry 2017 regular unleaded 268  
## 4496 Chevrolet TrailBlazer 2007 regular unleaded 291  
## 2948 Acura MDX 2015 premium unleaded (recommended) 290  
## Engine.Cylinders Transmission.Type Driven\_Wheels Number.of.Doors  
## 1004 4 AUTOMATED\_MANUAL front wheel drive 4  
## 623 8 AUTOMATIC rear wheel drive 4  
## 2693 4 AUTOMATIC all wheel drive 4  
## 934 6 AUTOMATIC front wheel drive 4  
## 4496 6 AUTOMATIC rear wheel drive 4  
## 2948 6 AUTOMATIC all wheel drive 4  
## Market.Category Vehicle.Size Vehicle.Style  
## 1004 Performance Midsize Sedan  
## 623 N/A Large 4dr SUV  
## 2693 Crossover,Hatchback,Factory Tuner,Performance Compact 4dr Hatchback  
## 934 Performance Midsize Sedan  
## 4496 N/A Midsize 4dr SUV  
## 2948 Crossover,Luxury Midsize 4dr SUV  
## highway.MPG city.mpg Popularity MSRP logYear logEngineHP logMSRP  
## 1004 31 22 873 35851.67 7.608374 5.298317 10.48715  
## 623 19 13 2009 45692.50 7.608374 5.758902 10.72969  
## 2693 29 25 2009 30020.00 7.608871 5.370638 10.30962  
## 934 30 21 2031 31370.00 7.609367 5.590987 10.35361  
## 4496 20 14 1385 26507.50 7.604396 5.673323 10.18518  
## 2948 27 18 204 50256.25 7.608374 5.669881 10.82489  
## loghighwayMPG logcitympg sqrtMSRP invertedMSRP Observation  
## 1004 3.433987 3.091042 189.3454 2.789271e-05 5050  
## 623 2.944439 2.564949 213.7580 2.188543e-05 5050  
## 2693 3.367296 3.218876 173.2628 3.331113e-05 5050  
## 934 3.401197 3.044522 177.1158 3.187759e-05 5050  
## 4496 2.995732 2.639057 162.8112 3.772517e-05 5050  
## 2948 3.295837 2.890372 224.1791 1.989802e-05 5050

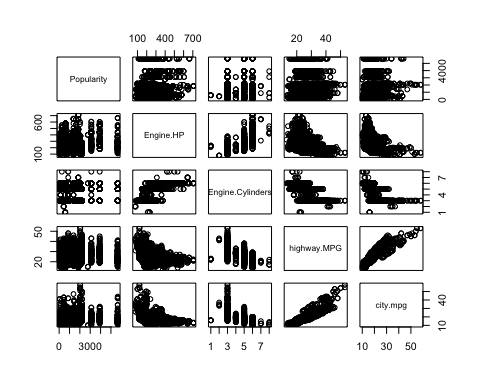
pairs(train[,c(16,3,5,6,9,13,14,15)])



**Initial Findings**

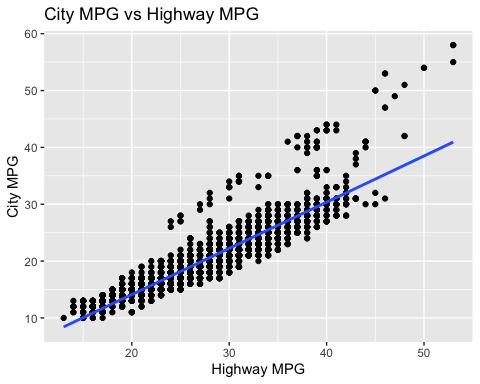
* City and highway mpg are highly correlated.
* Seeing some correlation between HP and Cylinders as well

pairs(train[,c(15,5,6,13,14)])

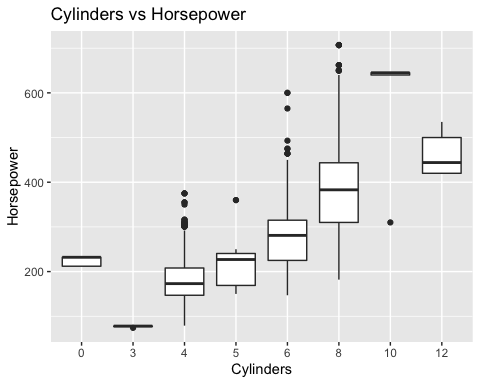


# City vs Highway MPG  
train %>%  
 ggplot(aes(x = highway.MPG, y = city.mpg)) +  
 geom\_point() +  
 geom\_smooth(method = "lm") +  
 xlab("Highway MPG") +  
 ylab("City MPG") +  
 ggtitle("City MPG vs Highway MPG")

## `geom\_smooth()` using formula 'y ~ x'



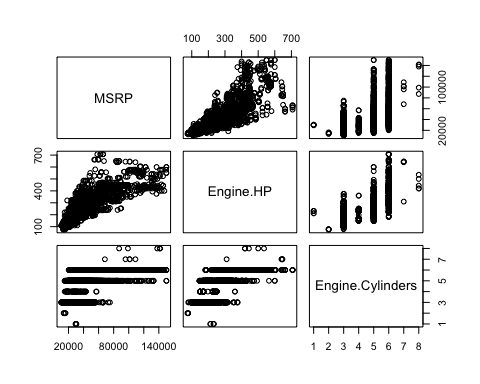
# Horsepower vs Cylinders  
train %>%  
 ggplot(aes(x = Engine.Cylinders, y = Engine.HP)) +  
 geom\_boxplot() +  
 xlab("Cylinders") +  
 ylab("Horsepower") +  
 ggtitle("Cylinders vs Horsepower")



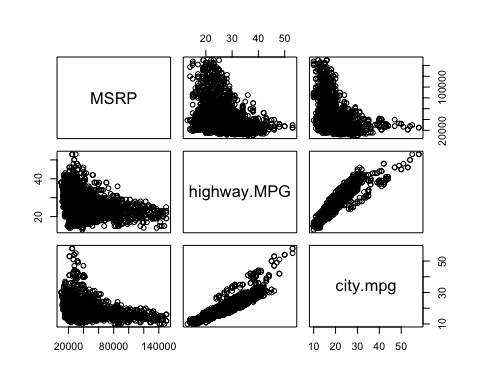
**Findings**

* We’ll need to choose between HP vs Cylinders and Highway mpg vs City mpg

pairs(train[,c(16,5,6)])



pairs(train[,c(16,13,14)])



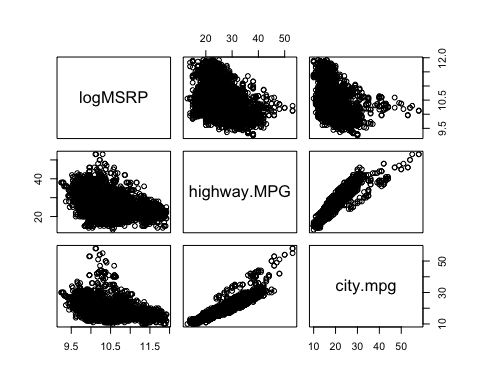
# highway mpg is statistically significant when used as a predictor for popularity (did you mean MSRP?) while city mpg is not. We'll go with highway mpg  
summary(lm(MSRP ~ highway.MPG, data = train))[4]

## $coefficients  
## Estimate Std. Error t value Pr(>|t|)  
## (Intercept) 71958.564 1578.23089 45.59445 0.000000e+00  
## highway.MPG -1222.082 57.77197 -21.15355 2.801735e-94

summary(lm(MSRP ~ city.mpg, data = train))[4]

## $coefficients  
## Estimate Std. Error t value Pr(>|t|)  
## (Intercept) 69012.082 1289.06867 53.53639 0.000000e+00  
## city.mpg -1517.986 63.69272 -23.83295 1.410862e-117

# Checking against Log MSRP  
pairs(train[,c(19,13,14)])



summary(lm(logMSRP ~ highway.MPG, data = train))[4]

## $coefficients  
## Estimate Std. Error t value Pr(>|t|)  
## (Intercept) 11.24299823 0.033036352 340.32203 0.000000e+00  
## highway.MPG -0.02976946 0.001209313 -24.61683 8.648528e-125

summary(lm(logMSRP ~ city.mpg, data = train))[4]

## $coefficients  
## Estimate Std. Error t value Pr(>|t|)  
## (Intercept) 11.16066560 0.026929603 414.4386 0.000000e+00  
## city.mpg -0.03643669 0.001330588 -27.3839 1.385997e-151

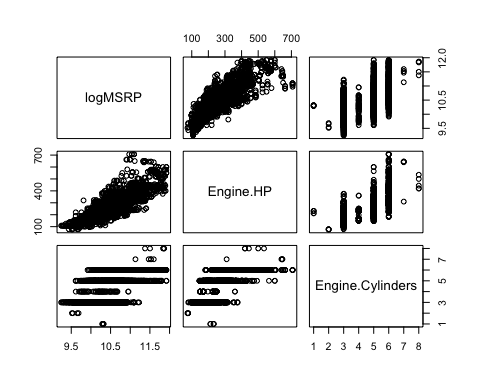
# Will stick with highway mpg  
  
  
# Both are statistically significant - next we'll test to see how they hold up when paired with highway mpg  
  
  
summary(lm(MSRP ~ Engine.HP, data = train))[4]

## $coefficients  
## Estimate Std. Error t value Pr(>|t|)  
## (Intercept) -8936.3602 591.364735 -15.11142 3.143431e-50  
## Engine.HP 186.6939 2.130789 87.61724 0.000000e+00

summary(lm(MSRP ~ Engine.Cylinders, data = train))[4]

## $coefficients  
## Estimate Std. Error t value Pr(>|t|)  
## (Intercept) 29931.5833 8387.536 3.56857894 3.631009e-04  
## Engine.Cylinders3 -15144.9167 12581.304 -1.20376371 2.287516e-01  
## Engine.Cylinders4 -3140.2018 8400.730 -0.37380106 7.085720e-01  
## Engine.Cylinders5 -756.2593 8714.020 -0.08678649 9.308455e-01  
## Engine.Cylinders6 10748.6840 8400.542 1.27952272 2.007867e-01  
## Engine.Cylinders8 33165.0256 8415.117 3.94112457 8.247471e-05  
## Engine.Cylinders10 65182.4167 11861.767 5.49516930 4.144477e-08  
## Engine.Cylinders12 91488.4167 11861.767 7.71288277 1.539766e-14

# Checking against Log MSRP  
pairs(train[,c(19,5,6)])



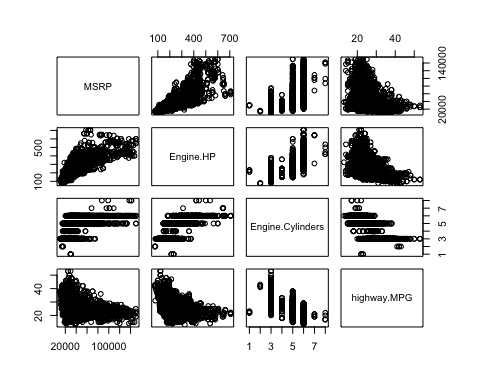
summary(lm(logMSRP ~ Engine.HP, data = train))[4]

## $coefficients  
## Estimate Std. Error t value Pr(>|t|)  
## (Intercept) 9.367631310 1.126880e-02 831.2889 0  
## Engine.HP 0.004179939 4.060345e-05 102.9454 0

summary(lm(logMSRP ~ Engine.Cylinders, data = train))[4]

## $coefficients  
## Estimate Std. Error t value Pr(>|t|)  
## (Intercept) 10.30657588 0.1733003 59.4723656 0.000000e+00  
## Engine.Cylinders3 -0.70768299 0.2599504 -2.7223773 6.509288e-03  
## Engine.Cylinders4 -0.17381470 0.1735729 -1.0013933 3.166968e-01  
## Engine.Cylinders5 -0.07479268 0.1800460 -0.4154088 6.778647e-01  
## Engine.Cylinders6 0.22693943 0.1735690 1.3074885 1.911214e-01  
## Engine.Cylinders8 0.63566123 0.1738701 3.6559541 2.594940e-04  
## Engine.Cylinders10 1.14370777 0.2450836 4.6666033 3.161607e-06  
## Engine.Cylinders12 1.38008259 0.2450836 5.6310695 1.912926e-08

pairs(train[,c(16,5,6,13)])



summary(lm(Popularity ~ Engine.HP + highway.MPG, data = train))[4]

## $coefficients  
## Estimate Std. Error t value Pr(>|t|)  
## (Intercept) 239.844344 160.7708046 1.491840 1.358192e-01  
## Engine.HP 2.256954 0.2574994 8.764890 2.707442e-18  
## highway.MPG 26.546665 4.3200620 6.144973 8.771538e-10

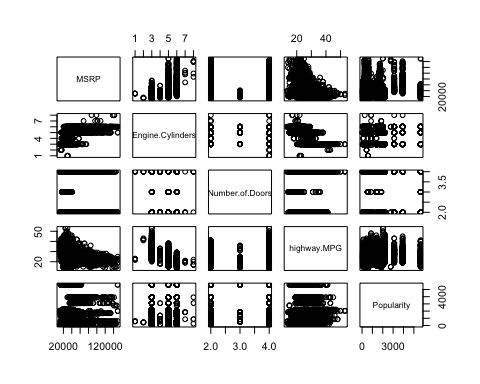
summary(lm(Popularity ~ Engine.Cylinders + highway.MPG, data = train))[4]

## $coefficients  
## Estimate Std. Error t value Pr(>|t|)  
## (Intercept) -186.66984 648.773674 -0.2877272 7.735704e-01  
## Engine.Cylinders3 -826.08611 961.959323 -0.8587537 3.905275e-01  
## Engine.Cylinders4 547.26614 640.364438 0.8546167 3.928142e-01  
## Engine.Cylinders5 131.67557 662.683998 0.1987004 8.425071e-01  
## Engine.Cylinders6 944.57706 638.511114 1.4793432 1.391268e-01  
## Engine.Cylinders8 1078.73339 639.574741 1.6866416 9.174966e-02  
## Engine.Cylinders10 1152.18605 901.541523 1.2780177 2.013168e-01  
## Engine.Cylinders12 1350.27792 901.681611 1.4975108 1.343387e-01  
## highway.MPG 34.49419 5.394686 6.3941056 1.799013e-10

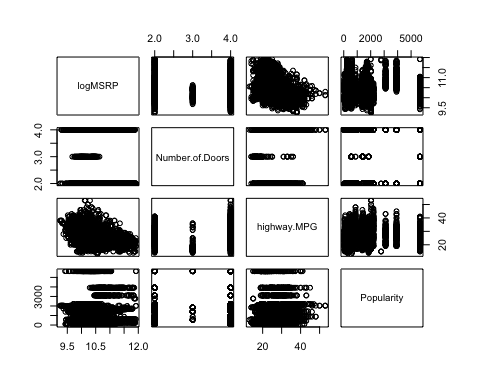
# Seeing some collinearity between mpg, horsepower, and cylinders. We'll move forward with caution using highway mpg and cylinders, understanding that we may ultimately need to decide between the two.  
  
head(train)

## Make Model Year Engine.Fuel.Type Engine.HP  
## 1004 Volkswagen CC 2015 premium unleaded (recommended) 200  
## 623 Nissan Armada 2015 regular unleaded 317  
## 2693 Nissan Juke 2016 premium unleaded (required) 215  
## 934 Toyota Camry 2017 regular unleaded 268  
## 4496 Chevrolet TrailBlazer 2007 regular unleaded 291  
## 2948 Acura MDX 2015 premium unleaded (recommended) 290  
## Engine.Cylinders Transmission.Type Driven\_Wheels Number.of.Doors  
## 1004 4 AUTOMATED\_MANUAL front wheel drive 4  
## 623 8 AUTOMATIC rear wheel drive 4  
## 2693 4 AUTOMATIC all wheel drive 4  
## 934 6 AUTOMATIC front wheel drive 4  
## 4496 6 AUTOMATIC rear wheel drive 4  
## 2948 6 AUTOMATIC all wheel drive 4  
## Market.Category Vehicle.Size Vehicle.Style  
## 1004 Performance Midsize Sedan  
## 623 N/A Large 4dr SUV  
## 2693 Crossover,Hatchback,Factory Tuner,Performance Compact 4dr Hatchback  
## 934 Performance Midsize Sedan  
## 4496 N/A Midsize 4dr SUV  
## 2948 Crossover,Luxury Midsize 4dr SUV  
## highway.MPG city.mpg Popularity MSRP logYear logEngineHP logMSRP  
## 1004 31 22 873 35851.67 7.608374 5.298317 10.48715  
## 623 19 13 2009 45692.50 7.608374 5.758902 10.72969  
## 2693 29 25 2009 30020.00 7.608871 5.370638 10.30962  
## 934 30 21 2031 31370.00 7.609367 5.590987 10.35361  
## 4496 20 14 1385 26507.50 7.604396 5.673323 10.18518  
## 2948 27 18 204 50256.25 7.608374 5.669881 10.82489  
## loghighwayMPG logcitympg sqrtMSRP invertedMSRP Observation  
## 1004 3.433987 3.091042 189.3454 2.789271e-05 5050  
## 623 2.944439 2.564949 213.7580 2.188543e-05 5050  
## 2693 3.367296 3.218876 173.2628 3.331113e-05 5050  
## 934 3.401197 3.044522 177.1158 3.187759e-05 5050  
## 4496 2.995732 2.639057 162.8112 3.772517e-05 5050  
## 2948 3.295837 2.890372 224.1791 1.989802e-05 5050

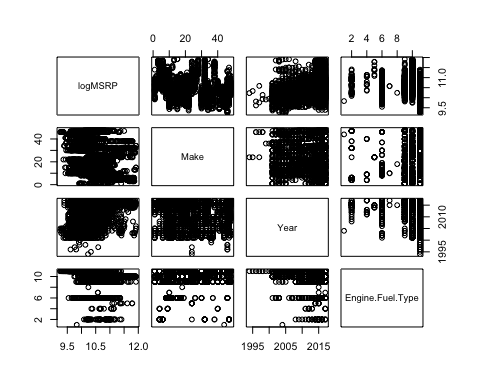
#Numerical  
pairs(train[,c(16,6,9,13,15)])



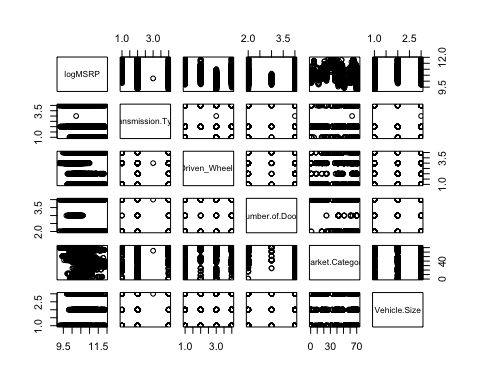
# Numerical with log msrp  
pairs(train[,c(19,9,13,15)]) # Log MSRP tends to show a more linear relationship



#Seeing some correlation with MSRP vs Year  
pairs(train[,c(19,1,3,4)])

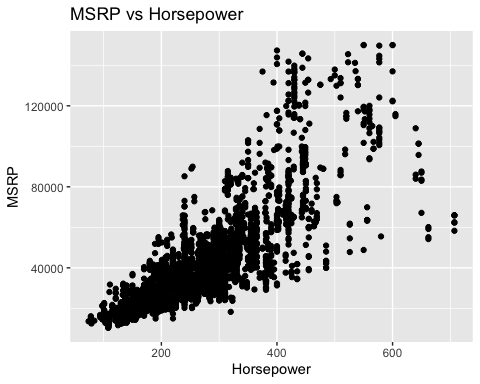


pairs(train[,c(19,7:11)])

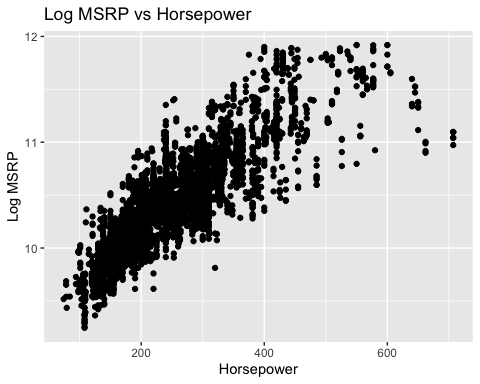


# Engine Horsepower and MSRP Comparisons

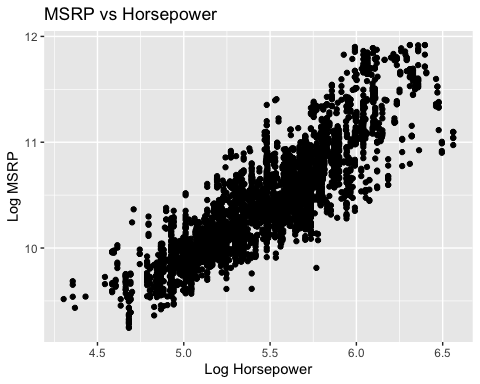
train %>%  
 ggplot(aes(x = Engine.HP, y = MSRP)) +  
 geom\_point() +  
 ylab("MSRP") +  
 xlab("Horsepower") +  
 ggtitle("MSRP vs Horsepower")



train %>%  
 ggplot(aes(x = Engine.HP, y = logMSRP)) +  
 geom\_point() +  
 ylab("Log MSRP") +  
 xlab("Horsepower") +  
 ggtitle("Log MSRP vs Horsepower")

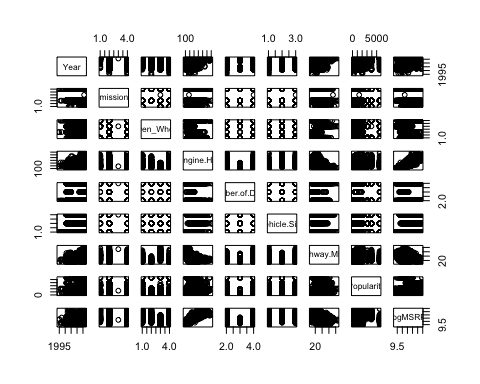


train %>%  
 ggplot(aes(x = logEngineHP, y = logMSRP)) +  
 geom\_point() +  
 ylab("Log MSRP") +  
 xlab("Log Horsepower") +  
 ggtitle("MSRP vs Horsepower")



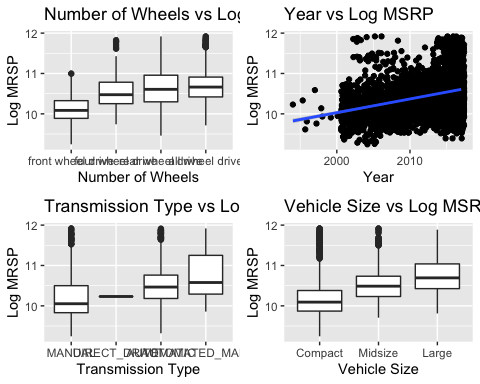
# Potential Predictors

attributes = subset(train, select = c("Year","Transmission.Type","Driven\_Wheels","Engine.HP","Number.of.Doors","Vehicle.Size","highway.MPG","Popularity","logMSRP"))  
  
pairs(attributes)



p1 = attributes %>%  
 ggplot(aes(x = reorder(Driven\_Wheels, logMSRP), y = logMSRP)) +  
 geom\_boxplot() +  
 ylab("Log MRSP") +  
 xlab("Number of Wheels") +  
 ggtitle("Number of Wheels vs Log MSRP")  
  
p2 = attributes %>%  
 ggplot(aes(x = Year, y = logMSRP)) +  
 geom\_point(position = "jitter") +  
 geom\_smooth(method = "lm") +  
 ylab("Log MRSP") +  
 xlab("Year") +  
 ggtitle("Year vs Log MSRP")  
  
  
p3 = attributes %>%  
 ggplot(aes(x = reorder(Transmission.Type, logMSRP), y = logMSRP)) +  
 geom\_boxplot() +  
 ylab("Log MRSP") +  
 xlab("Transmission Type") +  
 ggtitle("Transmission Type vs Log MSRP")  
  
p4 = attributes %>%  
 ggplot(aes(x = reorder(Vehicle.Size, logMSRP), y = logMSRP)) +  
 geom\_boxplot() +  
 ylab("Log MRSP") +  
 xlab("Vehicle Size") +  
 ggtitle("Vehicle Size vs Log MSRP")  
  
ggarrange(p1,p2,p3,p4)

## `geom\_smooth()` using formula 'y ~ x'



## Importance of Popularity:

#### Popularity is broken down by make - not by model. Also, no two makes have the same popularity. This means that popularity has a 1-1 relationship with Make, so we only need one of them

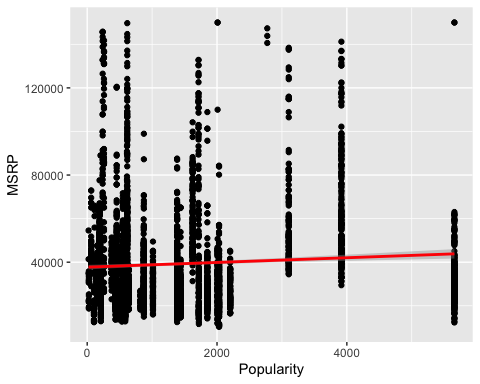
car\_data %>%  
 select(Make, Model, Popularity) %>%  
 group\_by(Make) %>%  
 summarize(count = n(), max = max(Popularity), min = min(Popularity)) %>%  
 arrange(max)

## # A tibble: 48 × 4  
## Make count max min  
## <fct> <int> <int> <int>  
## 1 Spyker 3 2 2  
## 2 Genesis 3 21 21  
## 3 Oldsmobile 150 26 26  
## 4 Lincoln 164 61 61  
## 5 Maybach 16 67 67  
## 6 Rolls-Royce 31 86 86  
## 7 Scion 60 105 105  
## 8 Alfa Romeo 5 113 113  
## 9 HUMMER 17 130 130  
## 10 Buick 196 155 155  
## # … with 38 more rows

#### Popularity vs MSRP

train %>%  
 ggplot(aes(x = Popularity, y = MSRP)) +  
 geom\_point() +  
 geom\_smooth(method = 'lm', color = 'red')

## `geom\_smooth()` using formula 'y ~ x'



summary(lm(MSRP~Popularity, data = train))

##   
## Call:  
## lm(formula = MSRP ~ Popularity, data = train)  
##   
## Residuals:  
## Min 1Q Median 3Q Max   
## -31494 -15227 -6476 6790 111306   
##   
## Coefficients:  
## Estimate Std. Error t value Pr(>|t|)   
## (Intercept) 3.773e+04 5.306e+02 71.110 < 2e-16 \*\*\*  
## Popularity 1.079e+00 2.527e-01 4.271 1.99e-05 \*\*\*  
## ---  
## Signif. codes: 0 '\*\*\*' 0.001 '\*\*' 0.01 '\*' 0.05 '.' 0.1 ' ' 1  
##   
## Residual standard error: 23070 on 4038 degrees of freedom  
## Multiple R-squared: 0.004498, Adjusted R-squared: 0.004251   
## F-statistic: 18.24 on 1 and 4038 DF, p-value: 1.989e-05

**Findings**

* There exists a negative correlation between popularity score and MSRP
  + This may indicate that more affordable cars are more popular
* The relationship between popularity and MRSP is statistically significant

# Model Creation Work

## Investigation of potential predictors noted in EDA

eda\_attributes = subset(train, select = c("Year","Transmission.Type","Driven\_Wheels","Engine.HP","Vehicle.Size","highway.MPG","Popularity","logMSRP"))  
  
  
full.model<-lm(logMSRP~.,data=eda\_attributes)  
vif(full.model)[,3]^2

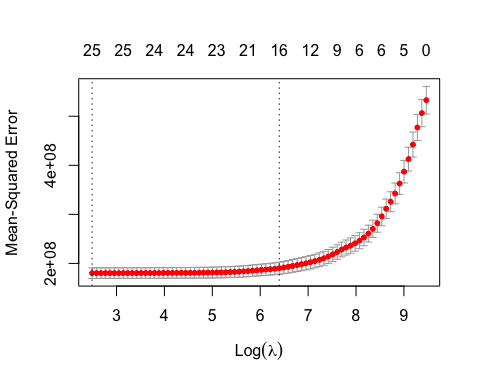
## Year Transmission.Type Driven\_Wheels Engine.HP   
## 1.811154 1.122663 1.304032 2.386983   
## Vehicle.Size highway.MPG Popularity   
## 1.329896 2.843927 1.051725

**Findings**

We’re not seeing much evidence of collinearity between these attributes, which is a good sign. We’ll likely include these in our final model. Other attributes will be found through variable selection techniques.

# Lasso Variable Selectioin

x=model.matrix(logMSRP~.,train[,c(3,4,6:9,11,13,15,19)])[,-1]  
y=train$MSRP  
  
xtest=model.matrix(logMSRP~.,test[,c(3,4,6:9,11,13,15,19)])[,-1]  
ytest=test$MSRP  
  
grid=10^seq(10,-2, length =100)  
lasso.mod=glmnet(x,y,alpha=1, lambda =grid)  
  
cv.out=cv.glmnet(x,y,alpha=1) #alpha=1 performs LASSO  
plot(cv.out)



bestlambda<-cv.out$lambda.min #Optimal penalty parameter. You can make this call visually.  
lasso.pred=predict (lasso.mod ,s=bestlambda ,newx=xtest)  
  
testMSE\_LASSO<-mean((ytest-lasso.pred)^2)  
testMSE\_LASSO

## [1] 202397211

coef(lasso.mod,s=bestlambda)

## 31 x 1 sparse Matrix of class "dgCMatrix"  
## s1  
## (Intercept) -2.532158e+06  
## Year 1.286656e+03  
## Engine.Fuel.Typediesel 8.639188e+03  
## Engine.Fuel.Typeelectric .   
## Engine.Fuel.Typeflex-fuel (premium unleaded recommended/E85) 3.029588e+03  
## Engine.Fuel.Typeflex-fuel (premium unleaded required/E85) 4.161823e+04  
## Engine.Fuel.Typeflex-fuel (unleaded/E85) -1.396693e+04  
## Engine.Fuel.Typeflex-fuel (unleaded/natural gas) -7.881280e+03  
## Engine.Fuel.Typenatural gas .   
## Engine.Fuel.Typepremium unleaded (recommended) .   
## Engine.Fuel.Typepremium unleaded (required) 1.505618e+04  
## Engine.Fuel.Typeregular unleaded -7.145002e+03  
## Engine.Cylinders3 -6.416869e+03  
## Engine.Cylinders4 -1.249596e+02  
## Engine.Cylinders5 5.617499e+03  
## Engine.Cylinders6 1.096751e+04  
## Engine.Cylinders8 2.741568e+04  
## Engine.Cylinders10 4.631071e+04  
## Engine.Cylinders12 7.744989e+04  
## Engine.Cylinders16 .   
## Transmission.TypeAUTOMATIC -7.232169e+03  
## Transmission.TypeDIRECT\_DRIVE 3.150667e+02  
## Transmission.TypeMANUAL -8.362362e+03  
## Driven\_Wheelsfour wheel drive -4.658937e+03  
## Driven\_Wheelsfront wheel drive -4.883857e+03  
## Driven\_Wheelsrear wheel drive -5.736228e+03  
## Number.of.Doors -2.587532e+03  
## Vehicle.SizeLarge 3.622278e+03  
## Vehicle.SizeMidsize -1.170293e+03  
## highway.MPG -2.441252e+02  
## Popularity -1.031915e-01

# LASSO Model - to be used as a reference for final model selection

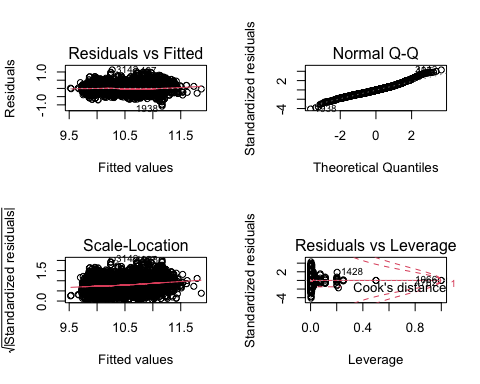
model = lm(logMSRP~ Engine.Fuel.Type + Engine.Cylinders+Transmission.Type+Driven\_Wheels+Number.of.Doors+Vehicle.Size+highway.MPG+Popularity, data = train)  
  
summary(model)

##   
## Call:  
## lm(formula = logMSRP ~ Engine.Fuel.Type + Engine.Cylinders +   
## Transmission.Type + Driven\_Wheels + Number.of.Doors + Vehicle.Size +   
## highway.MPG + Popularity, data = train)  
##   
## Residuals:  
## Min 1Q Median 3Q Max   
## -1.07800 -0.17572 -0.01908 0.14978 1.12487   
##   
## Coefficients:  
## Estimate  
## (Intercept) 9.631e+00  
## Engine.Fuel.Typediesel 7.050e-01  
## Engine.Fuel.Typeflex-fuel (premium unleaded recommended/E85) 8.029e-01  
## Engine.Fuel.Typeflex-fuel (premium unleaded required/E85) 1.227e+00  
## Engine.Fuel.Typeflex-fuel (unleaded/E85) 3.378e-01  
## Engine.Fuel.Typeflex-fuel (unleaded/natural gas) 6.430e-01  
## Engine.Fuel.Typenatural gas 6.093e-01  
## Engine.Fuel.Typepremium unleaded (recommended) 6.883e-01  
## Engine.Fuel.Typepremium unleaded (required) 8.658e-01  
## Engine.Fuel.Typeregular unleaded 3.565e-01  
## Engine.Cylinders3 -2.756e-01  
## Engine.Cylinders4 7.758e-02  
## Engine.Cylinders5 2.759e-01  
## Engine.Cylinders6 3.551e-01  
## Engine.Cylinders8 6.503e-01  
## Engine.Cylinders10 1.136e+00  
## Engine.Cylinders12 1.175e+00  
## Transmission.TypeAUTOMATIC -1.376e-01  
## Transmission.TypeDIRECT\_DRIVE -5.185e-02  
## Transmission.TypeMANUAL -2.647e-01  
## Driven\_Wheelsfour wheel drive -4.367e-02  
## Driven\_Wheelsfront wheel drive -2.654e-01  
## Driven\_Wheelsrear wheel drive -1.207e-01  
## Number.of.Doors -2.543e-02  
## Vehicle.SizeLarge 1.705e-01  
## Vehicle.SizeMidsize 7.836e-02  
## highway.MPG 1.075e-02  
## Popularity 9.548e-06  
## Std. Error t value  
## (Intercept) 2.918e-01 32.999  
## Engine.Fuel.Typediesel 2.655e-01 2.656  
## Engine.Fuel.Typeflex-fuel (premium unleaded recommended/E85) 2.726e-01 2.945  
## Engine.Fuel.Typeflex-fuel (premium unleaded required/E85) 2.777e-01 4.419  
## Engine.Fuel.Typeflex-fuel (unleaded/E85) 2.638e-01 1.281  
## Engine.Fuel.Typeflex-fuel (unleaded/natural gas) 3.223e-01 1.995  
## Engine.Fuel.Typenatural gas 3.722e-01 1.637  
## Engine.Fuel.Typepremium unleaded (recommended) 2.634e-01 2.613  
## Engine.Fuel.Typepremium unleaded (required) 2.634e-01 3.287  
## Engine.Fuel.Typeregular unleaded 2.632e-01 1.355  
## Engine.Cylinders3 1.782e-01 -1.546  
## Engine.Cylinders4 1.190e-01 0.652  
## Engine.Cylinders5 1.233e-01 2.238  
## Engine.Cylinders6 1.187e-01 2.992  
## Engine.Cylinders8 1.189e-01 5.468  
## Engine.Cylinders10 1.669e-01 6.810  
## Engine.Cylinders12 1.671e-01 7.032  
## Transmission.TypeAUTOMATIC 2.128e-02 -6.464  
## Transmission.TypeDIRECT\_DRIVE 2.644e-01 -0.196  
## Transmission.TypeMANUAL 2.207e-02 -11.994  
## Driven\_Wheelsfour wheel drive 1.791e-02 -2.439  
## Driven\_Wheelsfront wheel drive 1.292e-02 -20.544  
## Driven\_Wheelsrear wheel drive 1.268e-02 -9.516  
## Number.of.Doors 5.896e-03 -4.313  
## Vehicle.SizeLarge 1.539e-02 11.079  
## Vehicle.SizeMidsize 1.128e-02 6.948  
## highway.MPG 1.158e-03 9.278  
## Popularity 2.998e-06 3.185  
## Pr(>|t|)   
## (Intercept) < 2e-16 \*\*\*  
## Engine.Fuel.Typediesel 0.00795 \*\*   
## Engine.Fuel.Typeflex-fuel (premium unleaded recommended/E85) 0.00325 \*\*   
## Engine.Fuel.Typeflex-fuel (premium unleaded required/E85) 1.02e-05 \*\*\*  
## Engine.Fuel.Typeflex-fuel (unleaded/E85) 0.20032   
## Engine.Fuel.Typeflex-fuel (unleaded/natural gas) 0.04610 \*   
## Engine.Fuel.Typenatural gas 0.10169   
## Engine.Fuel.Typepremium unleaded (recommended) 0.00900 \*\*   
## Engine.Fuel.Typepremium unleaded (required) 0.00102 \*\*   
## Engine.Fuel.Typeregular unleaded 0.17563   
## Engine.Cylinders3 0.12219   
## Engine.Cylinders4 0.51453   
## Engine.Cylinders5 0.02527 \*   
## Engine.Cylinders6 0.00279 \*\*   
## Engine.Cylinders8 4.82e-08 \*\*\*  
## Engine.Cylinders10 1.12e-11 \*\*\*  
## Engine.Cylinders12 2.38e-12 \*\*\*  
## Transmission.TypeAUTOMATIC 1.14e-10 \*\*\*  
## Transmission.TypeDIRECT\_DRIVE 0.84452   
## Transmission.TypeMANUAL < 2e-16 \*\*\*  
## Driven\_Wheelsfour wheel drive 0.01479 \*   
## Driven\_Wheelsfront wheel drive < 2e-16 \*\*\*  
## Driven\_Wheelsrear wheel drive < 2e-16 \*\*\*  
## Number.of.Doors 1.65e-05 \*\*\*  
## Vehicle.SizeLarge < 2e-16 \*\*\*  
## Vehicle.SizeMidsize 4.30e-12 \*\*\*  
## highway.MPG < 2e-16 \*\*\*  
## Popularity 0.00146 \*\*   
## ---  
## Signif. codes: 0 '\*\*\*' 0.001 '\*\*' 0.01 '\*' 0.05 '.' 0.1 ' ' 1  
##   
## Residual standard error: 0.2629 on 4012 degrees of freedom  
## Multiple R-squared: 0.7167, Adjusted R-squared: 0.7148   
## F-statistic: 376 on 27 and 4012 DF, p-value: < 2.2e-16

par(mfrow=c(2,2))  
plot(model)

## Warning: not plotting observations with leverage one:  
## 2257

## Warning in sqrt(crit \* p \* (1 - hh)/hh): NaNs produced  
  
## Warning in sqrt(crit \* p \* (1 - hh)/hh): NaNs produced



# Final Model

final\_model = lm(logMSRP~Year+Transmission.Type+Driven\_Wheels+Engine.HP+Number.of.Doors+Vehicle.Size+highway.MPG+Popularity, data = train)  
summary(final\_model)

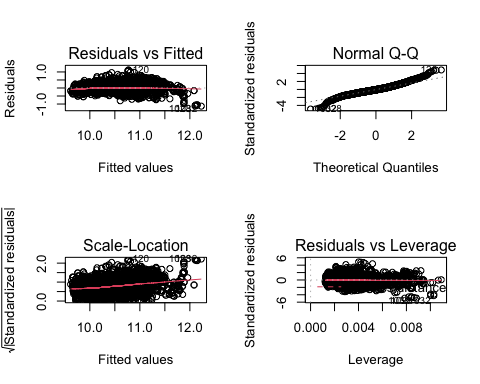
##   
## Call:  
## lm(formula = logMSRP ~ Year + Transmission.Type + Driven\_Wheels +   
## Engine.HP + Number.of.Doors + Vehicle.Size + highway.MPG +   
## Popularity, data = train)  
##   
## Residuals:  
## Min 1Q Median 3Q Max   
## -1.13205 -0.13710 -0.01697 0.12051 1.12650   
##   
## Coefficients:  
## Estimate Std. Error t value Pr(>|t|)   
## (Intercept) -3.653e+00 2.062e+00 -1.772 0.0765 .   
## Year 6.647e-03 1.037e-03 6.413 1.60e-10 \*\*\*  
## Transmission.TypeAUTOMATIC -1.370e-01 1.840e-02 -7.443 1.20e-13 \*\*\*  
## Transmission.TypeDIRECT\_DRIVE -1.463e-01 2.324e-01 -0.630 0.5290   
## Transmission.TypeMANUAL -2.634e-01 1.926e-02 -13.675 < 2e-16 \*\*\*  
## Driven\_Wheelsfour wheel drive -7.284e-02 1.512e-02 -4.816 1.52e-06 \*\*\*  
## Driven\_Wheelsfront wheel drive -2.223e-01 1.116e-02 -19.920 < 2e-16 \*\*\*  
## Driven\_Wheelsrear wheel drive -1.230e-01 1.082e-02 -11.366 < 2e-16 \*\*\*  
## Engine.HP 3.774e-03 5.725e-05 65.932 < 2e-16 \*\*\*  
## Number.of.Doors -4.676e-02 5.147e-03 -9.084 < 2e-16 \*\*\*  
## Vehicle.SizeLarge 3.226e-02 1.282e-02 2.516 0.0119 \*   
## Vehicle.SizeMidsize 2.529e-02 9.635e-03 2.625 0.0087 \*\*   
## highway.MPG 7.501e-03 1.029e-03 7.287 3.79e-13 \*\*\*  
## Popularity -1.351e-05 2.607e-06 -5.180 2.33e-07 \*\*\*  
## ---  
## Signif. codes: 0 '\*\*\*' 0.001 '\*\*' 0.01 '\*' 0.05 '.' 0.1 ' ' 1  
##   
## Residual standard error: 0.2313 on 4026 degrees of freedom  
## Multiple R-squared: 0.7801, Adjusted R-squared: 0.7793   
## F-statistic: 1098 on 13 and 4026 DF, p-value: < 2.2e-16

confint(final\_model)

## 2.5 % 97.5 %  
## (Intercept) -7.694572e+00 3.893582e-01  
## Year 4.615146e-03 8.679843e-03  
## Transmission.TypeAUTOMATIC -1.730609e-01 -1.008978e-01  
## Transmission.TypeDIRECT\_DRIVE -6.019012e-01 3.093032e-01  
## Transmission.TypeMANUAL -3.011518e-01 -2.256286e-01  
## Driven\_Wheelsfour wheel drive -1.024864e-01 -4.318436e-02  
## Driven\_Wheelsfront wheel drive -2.441980e-01 -2.004363e-01  
## Driven\_Wheelsrear wheel drive -1.441847e-01 -1.017614e-01  
## Engine.HP 3.662252e-03 3.886728e-03  
## Number.of.Doors -5.684539e-02 -3.666462e-02  
## Vehicle.SizeLarge 7.121798e-03 5.739301e-02  
## Vehicle.SizeMidsize 6.400998e-03 4.418243e-02  
## highway.MPG 5.483238e-03 9.519510e-03  
## Popularity -1.861835e-05 -8.394403e-06

par(mfrow=c(2,2))  
plot(final\_model)

## Warning: not plotting observations with leverage one:  
## 2257

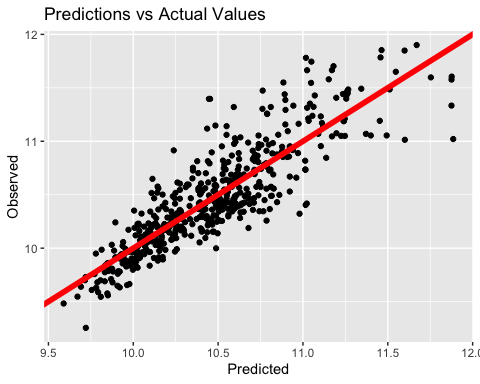


summary(train)

## Make Model Year   
## Chevrolet: 360 Tundra : 37 Min. :1994   
## Ford : 246 E-Class : 32 1st Qu.:2009   
## BMW : 237 F-150 : 32 Median :2015   
## Toyota : 220 Silverado 1500: 30 Mean :2012   
## Infiniti : 217 Sierra 1500 : 29 3rd Qu.:2016   
## Dodge : 196 911 : 27 Max. :2017   
## (Other) :2564 (Other) :3853   
## Engine.Fuel.Type Engine.HP   
## regular unleaded :2138 Min. : 74.0   
## premium unleaded (required) : 958 1st Qu.:180.8   
## premium unleaded (recommended) : 596 Median :245.0   
## flex-fuel (unleaded/E85) : 256 Mean :258.8   
## diesel : 65 3rd Qu.:310.0   
## flex-fuel (premium unleaded recommended/E85): 14 Max. :707.0   
## (Other) : 13   
## Engine.Cylinders Transmission.Type Driven\_Wheels   
## 6 :1611 AUTOMATED\_MANUAL: 183 all wheel drive : 977   
## 4 :1588 AUTOMATIC :2968 four wheel drive : 380   
## 8 : 759 DIRECT\_DRIVE : 1 front wheel drive:1495   
## 5 : 63 MANUAL : 888 rear wheel drive :1188   
## 0 : 5   
## 10 : 5   
## (Other): 9   
## Number.of.Doors Market.Category Vehicle.Size Vehicle.Style   
## Min. :2.000 N/A :1055 Compact:1528 Sedan :1154   
## 1st Qu.:3.000 Crossover : 358 Large : 886 4dr SUV : 941   
## Median :4.000 Luxury : 304 Midsize:1626 Coupe : 399   
## Mean :3.524 Luxury,Performance: 304 Convertible : 285   
## 3rd Qu.:4.000 Flex Fuel : 244 4dr Hatchback: 249   
## Max. :4.000 Performance : 218 Wagon : 215   
## (Other) :1557 (Other) : 797   
## highway.MPG city.mpg Popularity MSRP   
## Min. :13.00 Min. :10.00 Min. : 21 Min. : 10355   
## 1st Qu.:23.00 1st Qu.:16.00 1st Qu.: 481 1st Qu.: 24040   
## Median :26.00 Median :18.00 Median :1385 Median : 32929   
## Mean :26.66 Mean :19.52 Mean :1532 Mean : 39381   
## 3rd Qu.:30.00 3rd Qu.:22.00 3rd Qu.:2009 3rd Qu.: 46041   
## Max. :53.00 Max. :58.00 Max. :5657 Max. :149995   
##   
## logYear logEngineHP logMSRP loghighwayMPG   
## Min. :7.598 Min. :4.304 Min. : 9.245 Min. :2.565   
## 1st Qu.:7.605 1st Qu.:5.197 1st Qu.:10.087 1st Qu.:3.135   
## Median :7.608 Median :5.501 Median :10.402 Median :3.258   
## Mean :7.607 Mean :5.484 Mean :10.449 Mean :3.258   
## 3rd Qu.:7.609 3rd Qu.:5.737 3rd Qu.:10.737 3rd Qu.:3.401   
## Max. :7.609 Max. :6.561 Max. :11.918 Max. :3.970   
##   
## logcitympg sqrtMSRP invertedMSRP Observation   
## Min. :2.303 Min. :101.8 Min. :6.667e-06 Min. :5050   
## 1st Qu.:2.773 1st Qu.:155.0 1st Qu.:2.172e-05 1st Qu.:5050   
## Median :2.890 Median :181.5 Median :3.037e-05 Median :5050   
## Mean :2.939 Mean :191.8 Mean :3.237e-05 Mean :5050   
## 3rd Qu.:3.091 3rd Qu.:214.6 3rd Qu.:4.160e-05 3rd Qu.:5050   
## Max. :4.060 Max. :387.3 Max. :9.657e-05 Max. :5050   
##

# Test Final Model Against Test Data

data\_mod = data.frame(Predicted = predict(final\_model,test), Observed = test$logMSRP)  
  
data\_mod %>%  
 ggplot(aes(x = Predicted, y = Observed)) +  
 geom\_point() +  
 geom\_abline(intercept = 0, slope = 1, color = 'red', size = 2) +  
 ggtitle("Predictions vs Actual Values")

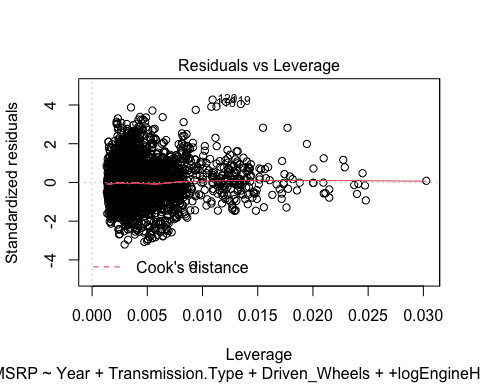
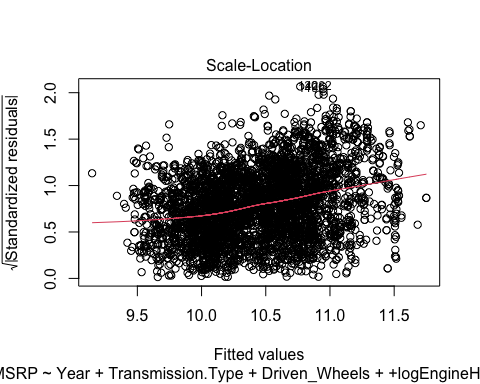
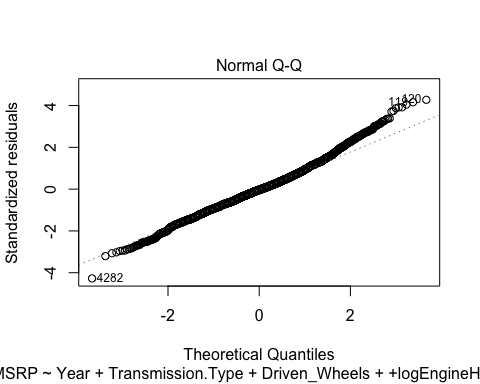
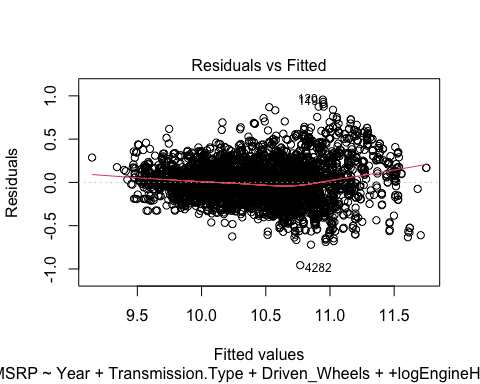


# Part 2

## Complex model creation:

complex\_model = lm(logMSRP~Year+Transmission.Type+Driven\_Wheels + +logEngineHP+Number.of.Doors+Vehicle.Size+highway.MPG+Popularity+ I(Number.of.Doors^2) + logEngineHP\*highway.MPG + Year\*highway.MPG + Year\*Popularity, data = train)  
  
plot(complex\_model)

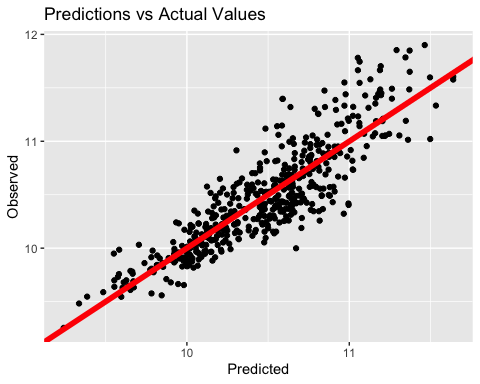
## Warning: not plotting observations with leverage one:  
## 2257



summary(complex\_model)

##   
## Call:  
## lm(formula = logMSRP ~ Year + Transmission.Type + Driven\_Wheels +   
## +logEngineHP + Number.of.Doors + Vehicle.Size + highway.MPG +   
## Popularity + I(Number.of.Doors^2) + logEngineHP \* highway.MPG +   
## Year \* highway.MPG + Year \* Popularity, data = train)  
##   
## Residuals:  
## Min 1Q Median 3Q Max   
## -0.95613 -0.14500 -0.00819 0.12907 0.95623   
##   
## Coefficients:  
## Estimate Std. Error t value Pr(>|t|)   
## (Intercept) 2.578e+01 7.670e+00 3.362 0.000782 \*\*\*  
## Year -1.100e-02 3.808e-03 -2.889 0.003886 \*\*   
## Transmission.TypeAUTOMATIC -1.276e-01 1.795e-02 -7.106 1.40e-12 \*\*\*  
## Transmission.TypeDIRECT\_DRIVE -2.222e-01 2.260e-01 -0.983 0.325718   
## Transmission.TypeMANUAL -2.256e-01 1.892e-02 -11.925 < 2e-16 \*\*\*  
## Driven\_Wheelsfour wheel drive -8.935e-02 1.501e-02 -5.952 2.87e-09 \*\*\*  
## Driven\_Wheelsfront wheel drive -1.920e-01 1.113e-02 -17.251 < 2e-16 \*\*\*  
## Driven\_Wheelsrear wheel drive -1.253e-01 1.059e-02 -11.835 < 2e-16 \*\*\*  
## logEngineHP 1.386e+00 4.662e-02 29.728 < 2e-16 \*\*\*  
## Number.of.Doors -5.043e-01 1.466e-01 -3.440 0.000588 \*\*\*  
## Vehicle.SizeLarge -1.348e-02 1.286e-02 -1.048 0.294573   
## Vehicle.SizeMidsize -3.208e-02 9.823e-03 -3.265 0.001102 \*\*   
## highway.MPG -1.161e+00 3.001e-01 -3.869 0.000111 \*\*\*  
## Popularity 5.521e-04 1.094e-03 0.505 0.613738   
## I(Number.of.Doors^2) 7.687e-02 2.430e-02 3.164 0.001570 \*\*   
## logEngineHP:highway.MPG -1.067e-02 1.681e-03 -6.346 2.46e-10 \*\*\*  
## Year:highway.MPG 6.106e-04 1.488e-04 4.103 4.17e-05 \*\*\*  
## Year:Popularity -2.799e-07 5.433e-07 -0.515 0.606419   
## ---  
## Signif. codes: 0 '\*\*\*' 0.001 '\*\*' 0.01 '\*' 0.05 '.' 0.1 ' ' 1  
##   
## Residual standard error: 0.2249 on 4022 degrees of freedom  
## Multiple R-squared: 0.7923, Adjusted R-squared: 0.7914   
## F-statistic: 902.4 on 17 and 4022 DF, p-value: < 2.2e-16

data\_mod = data.frame(Predicted = predict(complex\_model,test), Observed = test$logMSRP)  
  
data\_mod %>%  
 ggplot(aes(x = Predicted, y = Observed)) +  
 geom\_point() +  
 geom\_abline(intercept = 0, slope = 1, color = 'red', size = 2) +  
 ggtitle("Predictions vs Actual Values")



set.seed(1)  
  
colSums(is.na(train))

## 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 Market.Category Vehicle.Size Vehicle.Style   
## 0 0 0 0   
## highway.MPG city.mpg Popularity MSRP   
## 0 0 0 0   
## logYear logEngineHP logMSRP loghighwayMPG   
## 0 0 0 0   
## logcitympg sqrtMSRP invertedMSRP Observation   
## 0 0 0 0

colSums(is.na(test))

## 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 Market.Category Vehicle.Size Vehicle.Style   
## 0 0 0 0   
## highway.MPG city.mpg Popularity MSRP   
## 0 0 0 0   
## logYear logEngineHP logMSRP loghighwayMPG   
## 0 0 0 0   
## logcitympg sqrtMSRP invertedMSRP Observation   
## 0 0 0 0

head(train)

## Make Model Year Engine.Fuel.Type Engine.HP  
## 1004 Volkswagen CC 2015 premium unleaded (recommended) 200  
## 623 Nissan Armada 2015 regular unleaded 317  
## 2693 Nissan Juke 2016 premium unleaded (required) 215  
## 934 Toyota Camry 2017 regular unleaded 268  
## 4496 Chevrolet TrailBlazer 2007 regular unleaded 291  
## 2948 Acura MDX 2015 premium unleaded (recommended) 290  
## Engine.Cylinders Transmission.Type Driven\_Wheels Number.of.Doors  
## 1004 4 AUTOMATED\_MANUAL front wheel drive 4  
## 623 8 AUTOMATIC rear wheel drive 4  
## 2693 4 AUTOMATIC all wheel drive 4  
## 934 6 AUTOMATIC front wheel drive 4  
## 4496 6 AUTOMATIC rear wheel drive 4  
## 2948 6 AUTOMATIC all wheel drive 4  
## Market.Category Vehicle.Size Vehicle.Style  
## 1004 Performance Midsize Sedan  
## 623 N/A Large 4dr SUV  
## 2693 Crossover,Hatchback,Factory Tuner,Performance Compact 4dr Hatchback  
## 934 Performance Midsize Sedan  
## 4496 N/A Midsize 4dr SUV  
## 2948 Crossover,Luxury Midsize 4dr SUV  
## highway.MPG city.mpg Popularity MSRP logYear logEngineHP logMSRP  
## 1004 31 22 873 35851.67 7.608374 5.298317 10.48715  
## 623 19 13 2009 45692.50 7.608374 5.758902 10.72969  
## 2693 29 25 2009 30020.00 7.608871 5.370638 10.30962  
## 934 30 21 2031 31370.00 7.609367 5.590987 10.35361  
## 4496 20 14 1385 26507.50 7.604396 5.673323 10.18518  
## 2948 27 18 204 50256.25 7.608374 5.669881 10.82489  
## loghighwayMPG logcitympg sqrtMSRP invertedMSRP Observation  
## 1004 3.433987 3.091042 189.3454 2.789271e-05 5050  
## 623 2.944439 2.564949 213.7580 2.188543e-05 5050  
## 2693 3.367296 3.218876 173.2628 3.331113e-05 5050  
## 934 3.401197 3.044522 177.1158 3.187759e-05 5050  
## 4496 2.995732 2.639057 162.8112 3.772517e-05 5050  
## 2948 3.295837 2.890372 224.1791 1.989802e-05 5050

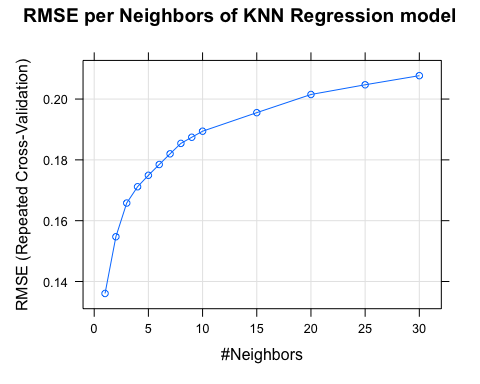
colSums(is.na(validate))

## 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 Market.Category Vehicle.Size Vehicle.Style   
## 0 0 0 0   
## highway.MPG city.mpg Popularity MSRP   
## 0 0 0 0   
## logYear logEngineHP logMSRP loghighwayMPG   
## 0 0 0 0   
## logcitympg sqrtMSRP invertedMSRP Observation   
## 0 0 0 0

train\_ = subset(train, select = c("Engine.HP", "Number.of.Doors", "city.mpg", "Popularity", "logMSRP"))  
train\_$city.mpg = log(train\_$city.mpg)  
train\_MSRP = train\_$logMSRP  
test\_ = subset(test, select = c("Engine.HP", "Number.of.Doors", "city.mpg", "Popularity", "logMSRP"))  
test\_$city.mpg = log(test\_$city.mpg)  
test\_$city.mpg = log(test\_$city.mpg)  
test\_MSRP = test\_$logMSRP  
validation\_ = subset(validate, select = c("Engine.HP", "Number.of.Doors", "city.mpg", "Popularity", "logMSRP"))  
validation\_$city.mpg = log(validation\_$city.mpg)  
validation\_$MSRP = validation\_$logMSRP

##KNN Model

fitControl = trainControl(method = "repeatedcv", number = 10, repeats = 10)  
  
knn\_model = train(logMSRP~., data = train\_, method = "knn", preProcess = c("center", "scale"), trControl = fitControl, tuneGrid = data.frame(k = c(1:10, 15, 20, 25, 30)))  
  
  
plot(knn\_model, main = "RMSE per Neighbors of KNN Regression model")

 # Testing Knn Model against test and validation sets

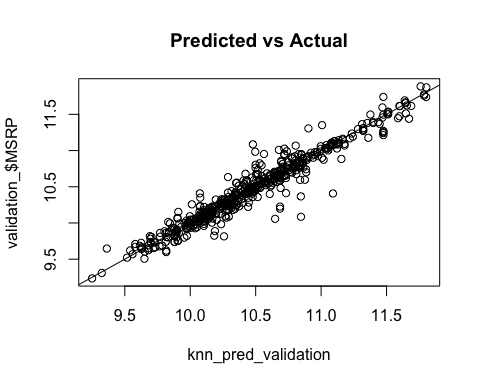
knn\_pred\_test = predict(knn\_model, test\_)  
  
test\_ase\_knn = mean(train\_$logMSRP - knn\_pred\_test)^2  
  
test\_ase\_knn

## [1] 0.1180802

knn\_pred\_validation = predict(knn\_model, validation\_)  
  
#Error metrics  
knn\_validate = postResample(pred = knn\_pred\_validation, obs = validation\_$MSRP)  
  
validation\_ase\_knn = mean(train\_$MSRP - knn\_pred\_validation)^2  
  
validation\_ase\_knn

## [1] NaN

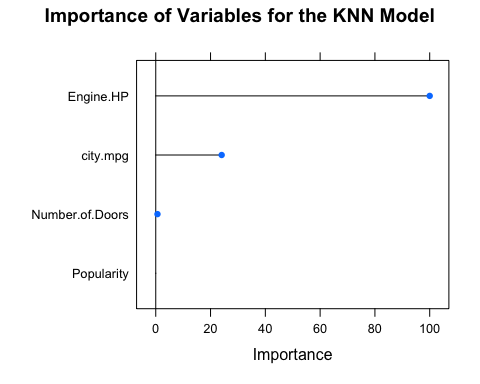
plot(knn\_pred\_validation, validation\_$MSRP, main = "Predicted vs Actual")  
lines(0:2000, 0:2000)



varImp(knn\_model)

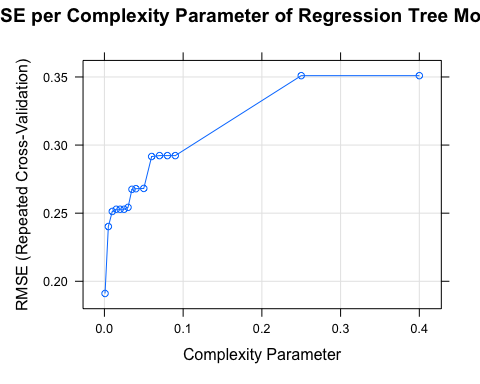
## loess r-squared variable importance  
##   
## Overall  
## Engine.HP 100.0000  
## city.mpg 24.0607  
## Number.of.Doors 0.6306  
## Popularity 0.0000

plot(varImp(knn\_model), main = "Importance of Variables for the KNN Model")



# Regression Tree

fitControl = trainControl(method = "repeatedcv", number = 10, repeats = 10)  
reg\_tree\_model = train(logMSRP~., data = train\_, method = "rpart", minsplit = 5, trControl = fitControl,  
 tuneGrid = data.frame(cp=c(.005,.0008,.01,.015,.02,.025,.03,.035,.04,.05,.06,.07,.08,.09,.25,.4)))  
  
  
  
plot(reg\_tree\_model, main = "RMSE per Complexity Parameter of Regression Tree Model")

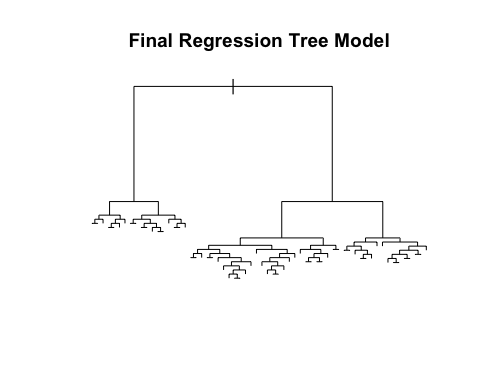


# Testing Regression Tree model against test and validation sets

test\_ase\_ref\_tree = mean(train\_$logMSRP - predict(reg\_tree\_model, test\_))^2  
  
  
test\_ase\_ref\_tree

## [1] 0.0009110068

plot(reg\_tree\_model$finalModel, main = "Final Regression Tree Model")



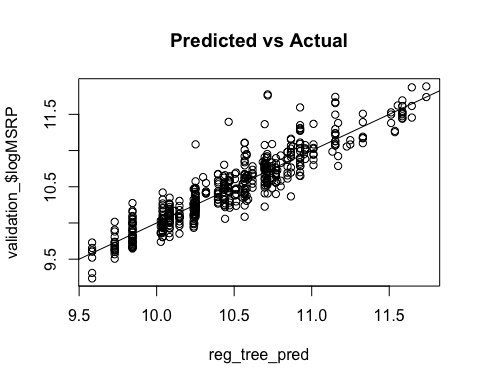
reg\_tree\_pred = predict(reg\_tree\_model, validation\_)  
  
reg\_tree\_validate = postResample(pred = reg\_tree\_pred, obs = validation\_$logMSRP)  
reg\_tree\_validate

## RMSE Rsquared MAE   
## 0.1931490 0.8500983 0.1399843

validation\_ase\_reg\_tree = mean(train\_$logMSRP - predict(reg\_tree\_model, validation\_))^2  
  
validation\_ase\_reg\_tree

## [1] 0.0003029807

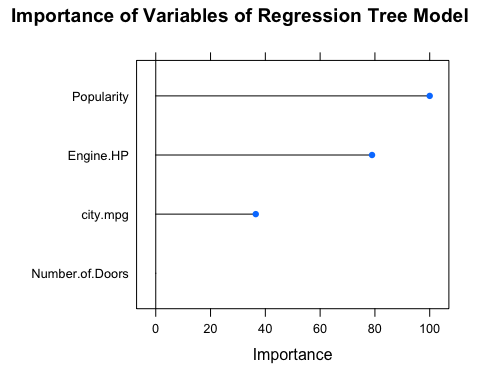
plot(reg\_tree\_pred, validation\_$logMSRP, main = "Predicted vs Actual")  
lines(0:2000, 0:2000)



varImp(reg\_tree\_model)

## rpart variable importance  
##   
## Overall  
## Popularity 100.00  
## Engine.HP 78.95  
## city.mpg 36.50  
## Number.of.Doors 0.00

plot(varImp(reg\_tree\_model), main = "Importance of Variables of Regression Tree Model")



# Model Comparisons

# Final Model  
  
test\_ase\_final\_model = mean(train$logMSRP - predict(final\_model, test))^2  
  
final\_model\_pred = predict(final\_model, validate)  
final\_model\_validate = postResample(pred = final\_model\_pred, obs = validate$logMSRP)  
  
validation\_ase\_final\_model = mean(train$logMSRP - predict(final\_model, validate))^2  
  
test\_ase\_final\_model

## [1] 0.000332322

validation\_ase\_final\_model

## [1] 0.0002448594

final\_model\_validate

## RMSE Rsquared MAE   
## 0.2327461 0.7824822 0.1732623

# Complex Model  
  
test\_ase\_complex\_model = mean(train$logMSRP - predict(complex\_model, test))^2  
  
complex\_model\_pred = predict(complex\_model, validate)  
complex\_model\_validate = postResample(pred = complex\_model\_pred, obs = validate$logMSRP)  
  
validation\_ase\_complex\_model = mean(train$logMSRP - predict(complex\_model, validate))^2  
  
test\_ase\_complex\_model

## [1] 0.0004900679

validation\_ase\_complex\_model

## [1] 0.0002903415

complex\_model\_validate

## RMSE Rsquared MAE   
## 0.2207695 0.8042274 0.1659148

# KNN Model  
  
test\_ase\_knn = mean(train\_$logMSRP - predict(reg\_tree\_model, test\_))^2  
  
knn\_pred = predict(knn\_model, validation\_)  
knn\_validate = postResample(pred = knn\_pred, obs = validation\_$logMSRP)  
  
validation\_ase\_knn = mean(train\_$logMSRP - predict(knn\_model, validation\_))^2  
  
test\_ase\_knn

## [1] 0.0009110068

validation\_ase\_knn

## [1] 0.001265711

knn\_validate

## RMSE Rsquared MAE   
## 0.12783558 0.93473733 0.08086537

#Regression Tree Model  
  
test\_ase\_ref\_tree = mean(train\_$logMSRP - predict(reg\_tree\_model, test\_))^2  
  
reg\_tree\_pred = predict(reg\_tree\_model, validation\_)  
reg\_tree\_validate = postResample(pred = reg\_tree\_pred, obs = validation\_$logMSRP)  
  
validation\_ase\_reg\_tree = mean(train\_$logMSRP - predict(reg\_tree\_model, validation\_))^2  
  
test\_ase\_ref\_tree

## [1] 0.0009110068

validation\_ase\_reg\_tree

## [1] 0.0003029807

reg\_tree\_validate

## RMSE Rsquared MAE   
## 0.1931490 0.8500983 0.1399843