# Stats 101A Final Project

Aryan Mistry

7/22/2022

This notebook is a final draft of my final project for my Stats 101A class. The problem statement involves working with Geely Auto, a Chinese automobile company that wishes to enter the American market, to understand the various factors affecting US car prices. Specifically, the company wishes to know which variables are significant in predicting car prices and how well these variables describe the prices of cars.

The provided data describes features of various cars on the American market, such as their year of manufacture, drivetrain type, passenger capacity, horsepower, etc. The aim is to use these predictors to create a multiple linear regression model that will be able to accurately predict car prices based on these features.

The project is assessed by two metrics: how well it explains variance, and how complex it is. The first criterion is measured using the model's R-squared value. In order to hedge against overly complicated models using the maximum number of predictors, a penalty was imposed for every predictor the model used after the 10th one. For example, 10 predictors would earn a 100% complexity score, while 27 predictors would earn a 73% complexity score.

```
library(dplyr)
```

```
##
## Attaching package: 'dplyr'
```

```
## The following objects are masked from 'package:stats':
##
##     filter, lag
```

```
## The following objects are masked from 'package:base':
##
##     intersect, setdiff, setequal, union
```

```
library(ggplot2)
```

```
#Reading in the data
cars_train <- read.csv("SummercarsTrain.csv")
cars_test <- read.csv("SummercarsTestNoY.csv")
head(cars_train)
```

```
##   Ob   Type MPG.highway     AirBags DriveTrain Cylinders EngineSize Horsepower
## 1  1 Sporty          25 Driver only       Rear         8        5.7        300
## 2  2  Small          29        None      Front         4        1.8        124
## 3  3  Small          36        None      Front         4        1.8        103
## 4  4 Sporty          26 Driver only      Front         4        1.6        100
## 5  5  Small          43        None      Front         3        1.3         70
## 6  6  Large          26 Driver only       Rear         8        5.0        170
##    RPM Rev.per.mile Man.trans.avail Fuel.tank.capacity Passengers Length
## 1 5000         1450             Yes               20.0          2    179
## 2 6000         2745             Yes               13.7          5    172
## 3 5500         2220             Yes               14.5          5    172
```

```
## 4 5750         2475             Yes               11.1          4    166
## 5 6000         3360             Yes               10.6          4    161
## 6 4200         1350              No               23.0          6    214
##   Wheelbase Width Turn.circle Rear.seat.room Luggage.room Weight  Origin
## 1        96    74          43           24.5           16   3380     USA
## 2        98    66          36           28.0           12   2620 non-USA
## 3        98    66          36           26.5           13   2440 non-USA
## 4        95    65          36           19.0            6   2450     USA
## 5        93    63          34           27.5           10   1965 non-USA
## 6       116    77          42           29.5           20   3910     USA
##                 Make  PriceNew
## 1 Chevrolet Corvette 39263.269
## 2    Hyundai Elantra 12758.759
## 3      Mazda Protege 12938.756
## 4      Mercury Capri 15676.605
## 5       Suzuki Swift  8361.677
## 6  Chevrolet Caprice 21141.481
```

One predetermined goal was to make use of the Make variable, which describes the make and model of every car in the dataset. Using this variable would lead to an extremely accurate model (since the model can just use the prices of makes in the training set to predict the prices of makes in the testing set). However, doing this would cause the model to have over 100 predictors, which would lead to a complexity score of 0%.

My idea was to cluster the various makes into smaller groups, such as categorizing cars by their country of manufacture (Japan, Germany, USA, etc.). Doing this would decrease the number of predictors.

I first made lists of German, Asian, and non-German European brands that were found in the dataset. I also made a list of which brands were considered luxurious.

```
other_euro <- c("Saab", "Volvo", "Geo")
asian <- c("Toyota", "Honda", "Suzuki", "Mitsubishi", "Nissan", "Mazda", "Acura", "Subaru", "Scion", "L
german <- c("Audi", "Porsche", "Volkswagen", "Mercedes-Benz", "BMW", "Mercedes")
luxury <- c("Audi", "Porsche", "Mercedes-Benz", "BMW", "Lexus", "Infiniti", "Acura", "Genesis", "Cadill
```

I used gsub to isolate the brand from the model.

```
cars_train$Brand <- gsub("([A-Za-z]+).*", "\\1", cars_train$Make)
cars_test$Brand <- gsub("([A-Za-z]+).*", "\\1", cars_test$Make)
unique(cars_train$Brand)
```

```
##  [1] "Chevrolet"  "Hyundai"    "Mazda"      "Mercury"    "Suzuki"
##  [6] "Lexus"      "Lincoln"    "Saturn"     "Cadillac"   "Mitsubishi"
## [11] "Honda"      "Buick"      "Dodge"      "Eagle"      "Nissan"
## [16] "Saab"       "Ford"       "Acura"      "Oldsmobile" "BMW"
## [21] "Subaru"     "Audi"       "Mercedes"   "Volkswagen" "Pontiac"
## [26] "Chrysler"   "Plymouth"   "Volvo"      "Chrylser"   "Toyota"
## [31] "Geo"        "Infiniti"
```

There was a typo in the dataset where "Chrysler" and "Chrylser" were two different brands.

```
cars_train$Brand[cars_train$Brand == "Chrylser"] <- "Chrysler"
cars_test$Brand[cars_test$Brand == "Chrylser"] <- "Chrysler"
```

We now create a new column for the country of manufacture of a given car, as well as a binary variable for whether or not it is a luxury vehicle.

```
#Creates two new columns: country and luxury. Country sorts the car brands into American, Asian, German
cars_train$country <- as.factor(ifelse(cars_train$Origin == "USA", "american",
                                ifelse(cars_train$Brand %in% other_euro, "other_euro",
```

```
                                ifelse(cars_train$Brand %in% asian, "asian", "german"))))

cars_train$luxury <- as.numeric(ifelse(cars_train$Brand %in% luxury, 1, 0))


cars_test$country <- as.factor(ifelse(cars_test$Origin == "USA", "american",
                                ifelse(cars_test$Brand %in% other_euro, "other_euro",
                                ifelse(cars_test$Brand %in% asian, "asian", "german"))))

cars_test$luxury <- as.numeric(ifelse(cars_test$Brand %in% luxury, 1, 0))

head(cars_train)
```

```
##   Ob   Type MPG.highway     AirBags DriveTrain Cylinders EngineSize Horsepower
## 1  1 Sporty          25 Driver only       Rear         8        5.7        300
## 2  2  Small          29        None      Front         4        1.8        124
## 3  3  Small          36        None      Front         4        1.8        103
## 4  4 Sporty          26 Driver only      Front         4        1.6        100
## 5  5  Small          43        None      Front         3        1.3         70
## 6  6  Large          26 Driver only       Rear         8        5.0        170
##    RPM Rev.per.mile Man.trans.avail Fuel.tank.capacity Passengers Length
## 1 5000         1450             Yes               20.0          2    179
## 2 6000         2745             Yes               13.7          5    172
## 3 5500         2220             Yes               14.5          5    172
## 4 5750         2475             Yes               11.1          4    166
## 5 6000         3360             Yes               10.6          4    161
## 6 4200         1350              No               23.0          6    214
##   Wheelbase Width Turn.circle Rear.seat.room Luggage.room Weight  Origin
## 1        96    74          43           24.5           16   3380     USA
## 2        98    66          36           28.0           12   2620 non-USA
## 3        98    66          36           26.5           13   2440 non-USA
## 4        95    65          36           19.0            6   2450     USA
## 5        93    63          34           27.5           10   1965 non-USA
## 6       116    77          42           29.5           20   3910     USA
##                 Make  PriceNew     Brand  country luxury
## 1 Chevrolet Corvette 39263.269 Chevrolet american      0
## 2    Hyundai Elantra 12758.759   Hyundai    asian      0
## 3      Mazda Protege 12938.756     Mazda    asian      0
## 4      Mercury Capri 15676.605   Mercury american      0
## 5       Suzuki Swift  8361.677    Suzuki    asian      0
## 6  Chevrolet Caprice 21141.481 Chevrolet american      0
```

Now we further cluster the Brand column into luxury brands exclusive to certain countries.

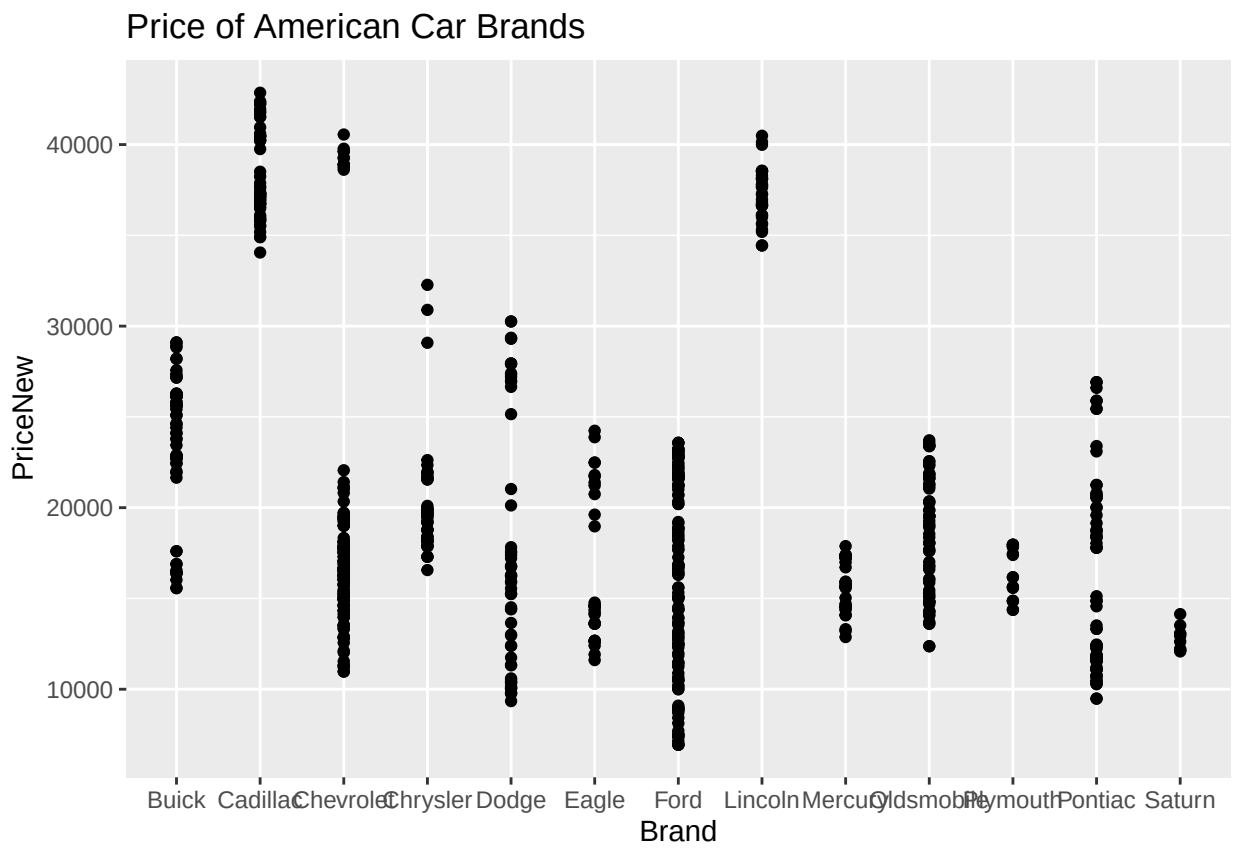
```
luxury_german <- c("Audi", "Porsche", "Mercedes-Benz", "BMW")
luxury_american <- c("Cadillac", "Lincoln")
luxury_japanese <- c("Lexus", "Infiniti", "Acura")
```

We now want to explore which car categories have similar distributions, so we can group them. We will do this with plots.

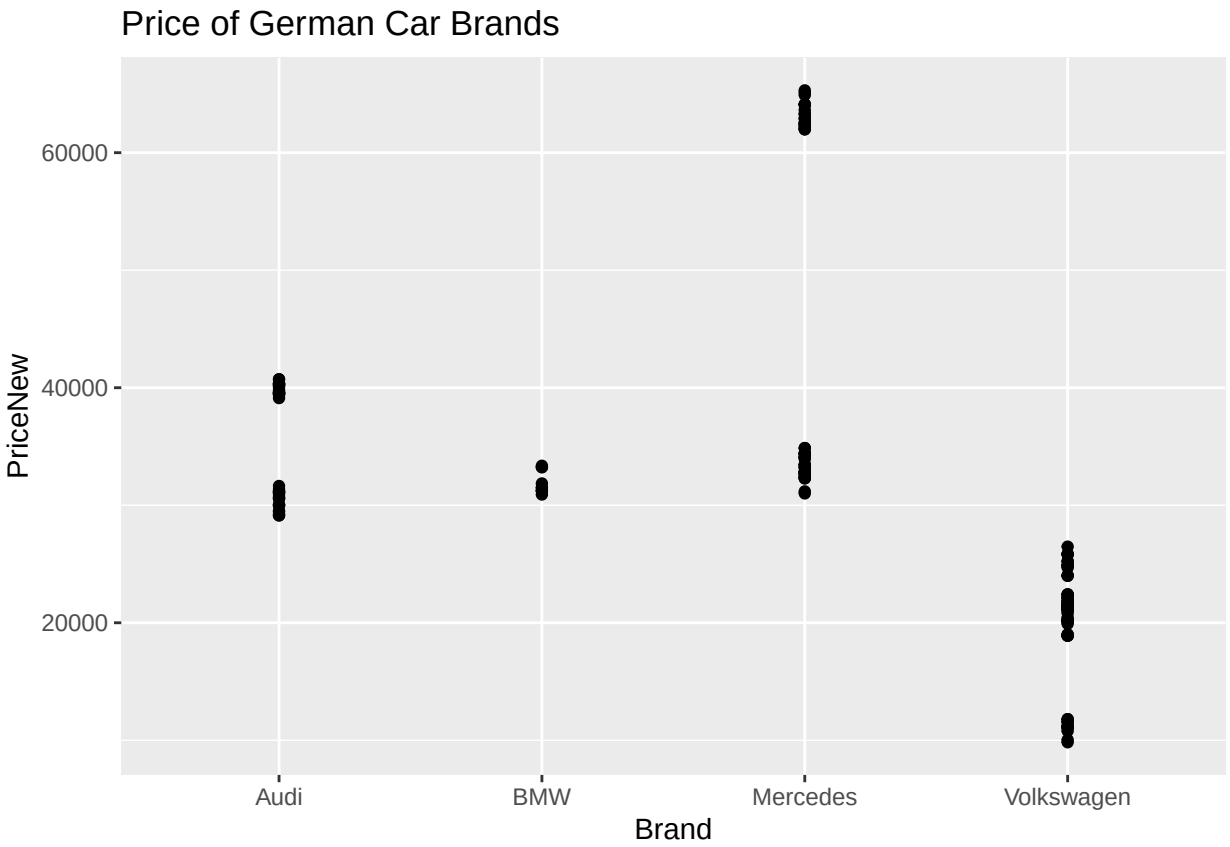
Price distribution of American cars

```
am <- cars_train %>% filter(Origin == "USA")

ggplot(am, aes(Brand, PriceNew)) +
```

```
  geom_point() +
  ggtitle("Price of American Car Brands")
```




```
eu <- cars_train %>% filter(country == "german")

ggplot(eu, aes(Brand, PriceNew)) +
  geom_point() +
  ggtitle("Price of German Car Brands")
```

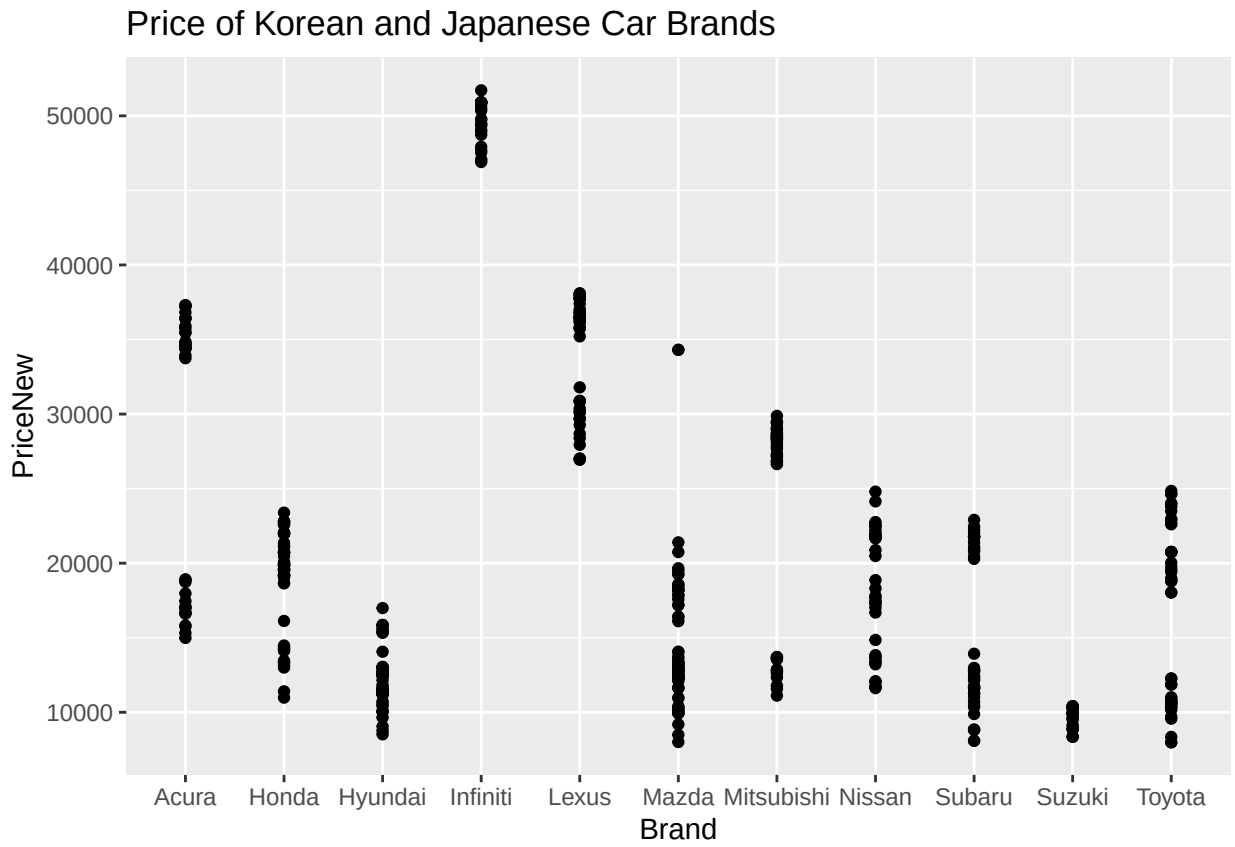



We can group the non-luxury American cars into one category, since their means don't differ too drastically. Mercedes cars should be a group of their own, since they are far more expensive than other German brands.

```
standard_american <- c("Chevrolet", "Mercury", "Saturn", "Buick", "Dodge", "Eagle", "Ford", "Oldsmobile
```

Let's do a similar exploration for Japanese and Korean cars.

```
as <- cars_train %>% filter(country == "asian")

ggplot(as, aes(Brand, PriceNew)) +
  geom_point() +
  ggtitle("Price of Korean and Japanese Car Brands")
```




Japanese and Korean non-luxury brands (Acura, Hyundai, Nissan, etc.) have a very similar price distribution, so we can group them into one category, which we call "standard_asian". The only exception is Suzuki, which has a much cheaper average price than the other brands. We will leave this out of our "standard_asian" group.

```
standard_asian <- c("Honda", "Mazda", "Nissan", "Subaru", "Toyota", "Hyundai", "Mitsubishi")
```

We will do the same for non-luxury European car brands.

```
standard_euro <- c("Volkswagen", "Saab", "Volvo")
```

We now overwrite our "Brand" variable to categorize the different makes into the below categories. Note that though the variable is called Brand, we are not actually categorizing the cars into brands, but rather categories of our own choosing using the above plots and analyses. We will do this for both the training and testing data.

```
cars_train$Brand[cars_train$Brand %in% luxury_american] <- "lux_american"
cars_train$Brand[cars_train$Brand %in% luxury_german] <- "lux_german"
cars_train$Brand[cars_train$Brand %in% luxury_japanese] <- "lux_japanese"
cars_train$Brand[cars_train$Brand %in% standard_american] <- "standard_american"
cars_train$Brand[cars_train$Brand %in% standard_asian] <- "standard_asian"
cars_train$Brand[cars_train$Brand %in% standard_euro] <- "standard_euro"
```

```
cars_test$Brand[cars_test$Brand %in% luxury_american] <- "lux_american"
cars_test$Brand[cars_test$Brand %in% luxury_german] <- "lux_german"
cars_test$Brand[cars_test$Brand %in% luxury_japanese] <- "lux_japanese"
cars_test$Brand[cars_test$Brand %in% standard_american] <- "standard_american"
cars_test$Brand[cars_test$Brand %in% standard_asian] <- "standard_asian"
cars_test$Brand[cars_test$Brand %in% standard_euro] <- "standard_euro"
```

We now use regsubsets to select the best subset of predictors to use for our model via forward selection. We choose nvmax (the max number of predictors to use) to be 14. This should make for a good balance between complexity and accuracy.

```
library(leaps)
fwd <- regsubsets(PriceNew ~ .-Make, data = cars_train, method = "forward", nvmax = 14)
```

```
## Warning in leaps.setup(x, y, wt = wt, nbest = nbest, nvmax = nvmax, force.in =
## force.in, : 4 linear dependencies found
```

```
## Reordering variables and trying again:
```

```
summary(fwd)
```

```
## Subset selection object
## Call: regsubsets.formula(PriceNew ~ . - Make, data = cars_train, method = "forward",
##     nvmax = 14)
## 43 Variables  (and intercept)
##                          Forced in Forced out
## Ob                           FALSE      FALSE
## TypeLarge                    FALSE      FALSE
## TypeMidsize                  FALSE      FALSE
## TypeSmall                    FALSE      FALSE
## TypeSporty                   FALSE      FALSE
## TypeVan                      FALSE      FALSE
## MPG.highway                  FALSE      FALSE
## AirBagsDriver only           FALSE      FALSE
## AirBagsNone                  FALSE      FALSE
## DriveTrainFront              FALSE      FALSE
## DriveTrainRear               FALSE      FALSE
## Cylinders4                   FALSE      FALSE
## Cylinders5                   FALSE      FALSE
## Cylinders6                   FALSE      FALSE
## Cylinders8                   FALSE      FALSE
## Cylindersrotary              FALSE      FALSE
## EngineSize                   FALSE      FALSE
## Horsepower                   FALSE      FALSE
## RPM                          FALSE      FALSE
## Rev.per.mile                 FALSE      FALSE
## Man.trans.availYes           FALSE      FALSE
## Fuel.tank.capacity           FALSE      FALSE
## Passengers                   FALSE      FALSE
## Length                       FALSE      FALSE
## Wheelbase                    FALSE      FALSE
## Width                        FALSE      FALSE
## Turn.circle                  FALSE      FALSE
## Rear.seat.room               FALSE      FALSE
## Luggage.room                 FALSE      FALSE
## Weight                       FALSE      FALSE
## OriginUSA                    FALSE      FALSE
## Brandlux_american            FALSE      FALSE
## Brandlux_german              FALSE      FALSE
## Brandlux_japanese            FALSE      FALSE
## BrandMercedes                FALSE      FALSE
## Brandstandard_asian          FALSE      FALSE
## Brandstandard_euro           FALSE      FALSE
```

```
## BrandSuzuki                FALSE      FALSE
## countrygerman              FALSE      FALSE
## Brandstandard_american     FALSE      FALSE
## countryasian               FALSE      FALSE
## countryother_euro          FALSE      FALSE
## luxury                     FALSE      FALSE
## 1 subsets of each size up to 15
## Selection Algorithm: forward
##           Ob  TypeLarge TypeMidsize TypeSmall TypeSporty TypeVan MPG.highway
## 1  ( 1 )  " " " "        " "          " "       " "         " "     " "
## 2  ( 1 )  " " " "        " "          " "       " "         " "     " "
## 3  ( 1 )  " " " "        " "          " "       " "         " "     " "
## 4  ( 1 )  " " " "        " "          " "       " "         " "     " "
## 5  ( 1 )  " " " "        "*"          " "       " "         " "     " "
## 6  ( 1 )  " " " "        "*"          " "       " "         " "     " "
## 7  ( 1 )  " " " "        "*"          " "       " "         " "     " "
## 8  ( 1 )  " " " "        "*"          " "       " "         " "     " "
## 9  ( 1 )  " " " "        "*"          " "       " "         " "     " "
## 10  ( 1 ) " " " "        "*"          " "       " "         " "     " "
## 11  ( 1 ) " " " "        "*"          " "       " "         " "     " "
## 12  ( 1 ) " " " "        "*"          " "       " "         " "     " "
## 13  ( 1 ) " " " "        "*"          " "       "*"         " "     " "
## 14  ( 1 ) " " " "        "*"          " "       "*"         " "     " "
## 15  ( 1 ) " " "*"        "*"          " "       "*"         " "     " "
##           AirBagsDriver only AirBagsNone DriveTrainFront DriveTrainRear
## 1  ( 1 )  " "                " "          " "              " "
## 2  ( 1 )  " "                " "          " "              " "
## 3  ( 1 )  " "                " "          " "              " "
## 4  ( 1 )  " "                " "          " "              " "
## 5  ( 1 )  " "                " "          " "              " "
## 6  ( 1 )  " "                "*"          " "              " "
## 7  ( 1 )  " "                "*"          " "              " "
## 8  ( 1 )  " "                "*"          " "              " "
## 9  ( 1 )  " "                "*"          " "              " "
## 10  ( 1 ) " "                "*"          " "              " "
## 11  ( 1 ) " "                "*"          " "              " "
## 12  ( 1 ) " "                "*"          " "              " "
## 13  ( 1 ) " "                "*"          " "              " "
## 14  ( 1 ) " "                "*"          " "              " "
## 15  ( 1 ) " "                "*"          " "              " "
##           Cylinders4 Cylinders5 Cylinders6 Cylinders8 Cylindersrotary
## 1  ( 1 )  " "        " "        " "        " "        " "
## 2  ( 1 )  " "        " "        " "        " "        " "
## 3  ( 1 )  " "        " "        " "        " "        " "
## 4  ( 1 )  " "        " "        " "        " "        " "
## 5  ( 1 )  " "        " "        " "        " "        " "
## 6  ( 1 )  " "        " "        " "        " "        " "
## 7  ( 1 )  " "        " "        " "        " "        " "
## 8  ( 1 )  " "        " "        " "        " "        " "
## 9  ( 1 )  " "        " "        " "        " "        " "
## 10  ( 1 ) " "        " "        " "        " "        " "
## 11  ( 1 ) "*"        " "        " "        " "        " "
## 12  ( 1 ) "*"        " "        " "        " "        " "
## 13  ( 1 ) "*"        " "        " "        " "        " "
```

```
## 14  ( 1 ) "*"         " "          " "          " "          " "
## 15  ( 1 ) "*"         " "          " "          " "          " "
##           EngineSize Horsepower RPM Rev.per.mile Man.trans.availYes
## 1  ( 1 )  " "        "*"        " " " "          " "
## 2  ( 1 )  " "        "*"        " " " "          " "
## 3  ( 1 )  " "        "*"        " " " "          " "
## 4  ( 1 )  " "        "*"        " " " "          " "
## 5  ( 1 )  " "        "*"        " " " "          " "
## 6  ( 1 )  " "        "*"        " " " "          " "
## 7  ( 1 )  " "        "*"        " " " "          " "
## 8  ( 1 )  " "        "*"        " " " "          " "
## 9  ( 1 )  " "        "*"        " " " "          " "
## 10  ( 1 ) " "        "*"        " " " "          "*"
## 11  ( 1 ) " "        "*"        " " " "          "*"
## 12  ( 1 ) " "        "*"        " " " "          "*"
## 13  ( 1 ) " "        "*"        " " " "          "*"
## 14  ( 1 ) " "        "*"        " " " "          "*"
## 15  ( 1 ) " "        "*"        " " " "          "*"
##           Fuel.tank.capacity Passengers Length Wheelbase Width Turn.circle
## 1  ( 1 )  " "                " "        " "    " "       " "   " "
## 2  ( 1 )  " "                " "        " "    " "       " "   " "
## 3  ( 1 )  " "                " "        " "    " "       " "   " "
## 4  ( 1 )  " "                " "        " "    " "       " "   " "
## 5  ( 1 )  " "                " "        " "    " "       " "   " "
## 6  ( 1 )  " "                " "        " "    " "       " "   " "
## 7  ( 1 )  "*"                " "        " "    " "       " "   " "
## 8  ( 1 )  "*"                " "        " "    " "       " "   " "
## 9  ( 1 )  "*"                " "        " "    " "       "*"   " "
## 10  ( 1 ) "*"                " "        " "    " "       "*"   " "
## 11  ( 1 ) "*"                " "        " "    " "       "*"   " "
## 12  ( 1 ) "*"                " "        " "    " "       "*"   " "
## 13  ( 1 ) "*"                " "        " "    " "       "*"   " "
## 14  ( 1 ) "*"                " "        " "    " "       "*"   " "
## 15  ( 1 ) "*"                " "        " "    " "       "*"   " "
##           Rear.seat.room Luggage.room Weight OriginUSA Brandlux_american
## 1  ( 1 )  " "            " "          " "    " "       " "
## 2  ( 1 )  " "            " "          " "    " "       " "
## 3  ( 1 )  " "            " "          " "    " "       " "
## 4  ( 1 )  " "            " "          " "    " "       " "
## 5  ( 1 )  " "            " "          " "    " "       " "
## 6  ( 1 )  " "            " "          " "    " "       " "
## 7  ( 1 )  " "            " "          " "    " "       " "
## 8  ( 1 )  " "            " "          " "    " "       " "
## 9  ( 1 )  " "            " "          " "    " "       " "
## 10  ( 1 ) " "            " "          " "    " "       " "
## 11  ( 1 ) " "            " "          " "    " "       " "
## 12  ( 1 ) " "            " "          " "    " "       " "
## 13  ( 1 ) " "            " "          " "    " "       " "
## 14  ( 1 ) " "            "*"          " "    " "       " "
## 15  ( 1 ) " "            "*"          " "    " "       " "
##           Brandlux_german Brandlux_japanese BrandMercedes
## 1  ( 1 )  " "              " "               " "
## 2  ( 1 )  " "              " "               "*"
## 3  ( 1 )  " "              " "               "*"
```

```
## 4  ( 1 )  " "             " "               "*"
## 5  ( 1 )  " "             " "               "*"
## 6  ( 1 )  " "             " "               "*"
## 7  ( 1 )  " "             " "               "*"
## 8  ( 1 )  " "             "*"               "*"
## 9  ( 1 )  " "             "*"               "*"
## 10  ( 1 ) " "             "*"               "*"
## 11  ( 1 ) " "             "*"               "*"
## 12  ( 1 ) " "             "*"               "*"
## 13  ( 1 ) " "             "*"               "*"
## 14  ( 1 ) " "             "*"               "*"
## 15  ( 1 ) " "             "*"               "*"
##           Brandstandard_american Brandstandard_asian Brandstandard_euro
## 1  ( 1 )  " "                    " "                 " "
## 2  ( 1 )  " "                    " "                 " "
## 3  ( 1 )  " "                    " "                 " "
## 4  ( 1 )  " "                    " "                 "*"
## 5  ( 1 )  " "                    " "                 "*"
## 6  ( 1 )  " "                    " "                 "*"
## 7  ( 1 )  " "                    " "                 "*"
## 8  ( 1 )  " "                    " "                 "*"
## 9  ( 1 )  " "                    " "                 "*"
## 10  ( 1 ) " "                    " "                 "*"
## 11  ( 1 ) " "                    " "                 "*"
## 12  ( 1 ) " "                    " "                 "*"
## 13  ( 1 ) " "                    " "                 "*"
## 14  ( 1 ) " "                    " "                 "*"
## 15  ( 1 ) " "                    " "                 "*"
##           BrandSuzuki countryasian countrygerman countryother_euro luxury
## 1  ( 1 )  " "         " "          " "           " "               " "
## 2  ( 1 )  " "         " "          " "           " "               " "
## 3  ( 1 )  " "         " "          " "           " "               "*"
## 4  ( 1 )  " "         " "          " "           " "               "*"
## 5  ( 1 )  " "         " "          " "           " "               "*"
## 6  ( 1 )  " "         " "          " "           " "               "*"
## 7  ( 1 )  " "         " "          " "           " "               "*"
## 8  ( 1 )  " "         " "          " "           " "               "*"
## 9  ( 1 )  " "         " "          " "           " "               "*"
## 10  ( 1 ) " "         " "          " "           " "               "*"
## 11  ( 1 ) " "         " "          " "           " "               "*"
## 12  ( 1 ) " "         " "          "*"           " "               "*"
## 13  ( 1 ) " "         " "          "*"           " "               "*"
## 14  ( 1 ) " "         " "          "*"           " "               "*"
## 15  ( 1 ) " "         " "          "*"           " "               "*"
```

```
res_sum <- summary(fwd)
data.frame(
Adj.R2 = which.max(res_sum$adjr2),
CP = which.min(res_sum$cp),
BIC = which.min(res_sum$bic))
```

```
##   Adj.R2 CP BIC
## 1     15 15  15
```

The 15th subset is the best set of predictors in this case. We must now refer to the regsubsets output and

create dummy variables for these chosen predictors.

```
cars_train$isLarge <- as.numeric(ifelse(cars_train$Type == "Large", 1, 0))
cars_train$isMidsize <- as.numeric(ifelse(cars_train$Type == "Midsize", 1, 0))
cars_train$isSporty <- as.numeric(ifelse(cars_train$Type == "Sporty", 1, 0))

cars_train$noBags <- as.numeric(ifelse(cars_train$AirBags == "None", 1, 0))

cars_train$is4cyl <- as.numeric(ifelse(cars_train$Cylinders == '4', 1, 0))
cars_train$is8cyl <- as.numeric(ifelse(cars_train$Cylinders == '8', 1, 0))

cars_train$isLuxjapan <- as.numeric(ifelse(cars_train$Brand == "lux_japanese", 1, 0))

cars_train$isMerc <- as.numeric(ifelse(cars_train$Brand == "Mercedes", 1, 0))

cars_train$isSE <- as.numeric(ifelse(cars_train$Brand == "standard_euro", 1, 0))

cars_train$isGerman <- as.numeric(ifelse(cars_train$country == "german", 1, 0))
```

```
cars_test$isLarge <- as.numeric(ifelse(cars_test$Type == "Large", 1, 0))
cars_test$isMidsize <- as.numeric(ifelse(cars_test$Type == "Midsize", 1, 0))
cars_test$isSporty <- as.numeric(ifelse(cars_test$Type == "Sporty", 1, 0))

cars_test$noBags <- as.numeric(ifelse(cars_test$AirBags == "None", 1, 0))

cars_test$is4cyl <- as.numeric(ifelse(cars_test$Cylinders == '4', 1, 0))
cars_test$is8cyl <- as.numeric(ifelse(cars_test$Cylinders == '8', 1, 0))

cars_test$isLuxjapan <- as.numeric(ifelse(cars_test$Brand == "lux_japanese", 1, 0))

cars_test$isMerc <- as.numeric(ifelse(cars_test$Brand == "Mercedes", 1, 0))

cars_test$isSE <- as.numeric(ifelse(cars_test$Brand == "standard_euro", 1, 0))

cars_test$isGerman <- as.numeric(ifelse(cars_test$country == "german", 1, 0))
```

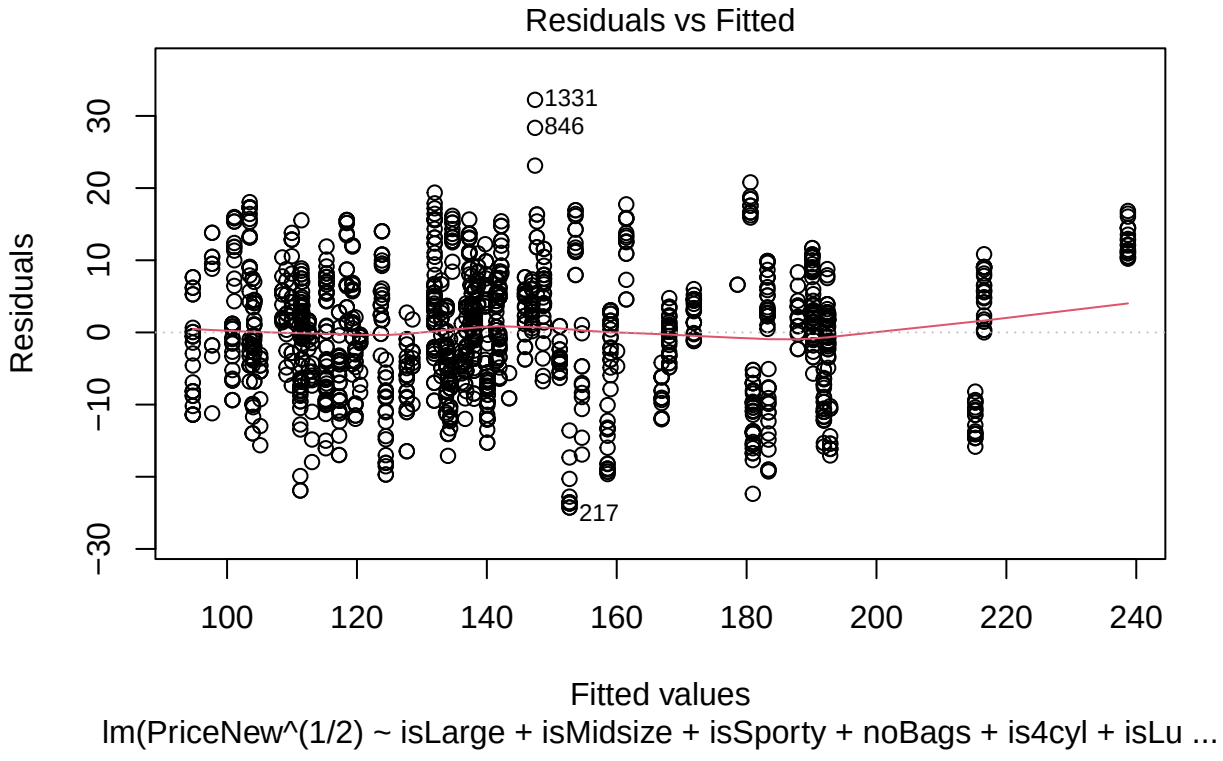
We create the model, performing a square-root transform on the response in order to mitigate non-constant variance and normalize the data.

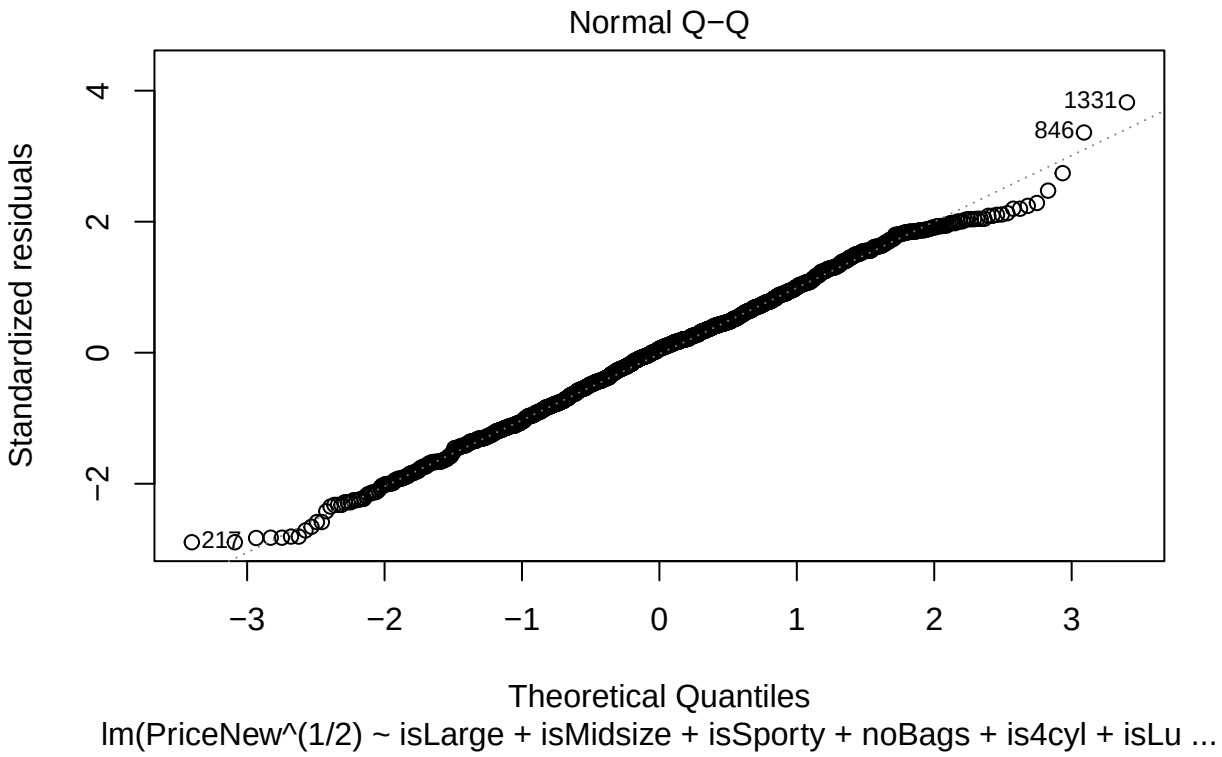
```
rgs_test <- lm(PriceNew^(1/2) ~ isLarge + isMidsize + isSporty + noBags + is4cyl + isLuxjapan + isMerc +

summary(rgs_test)
```

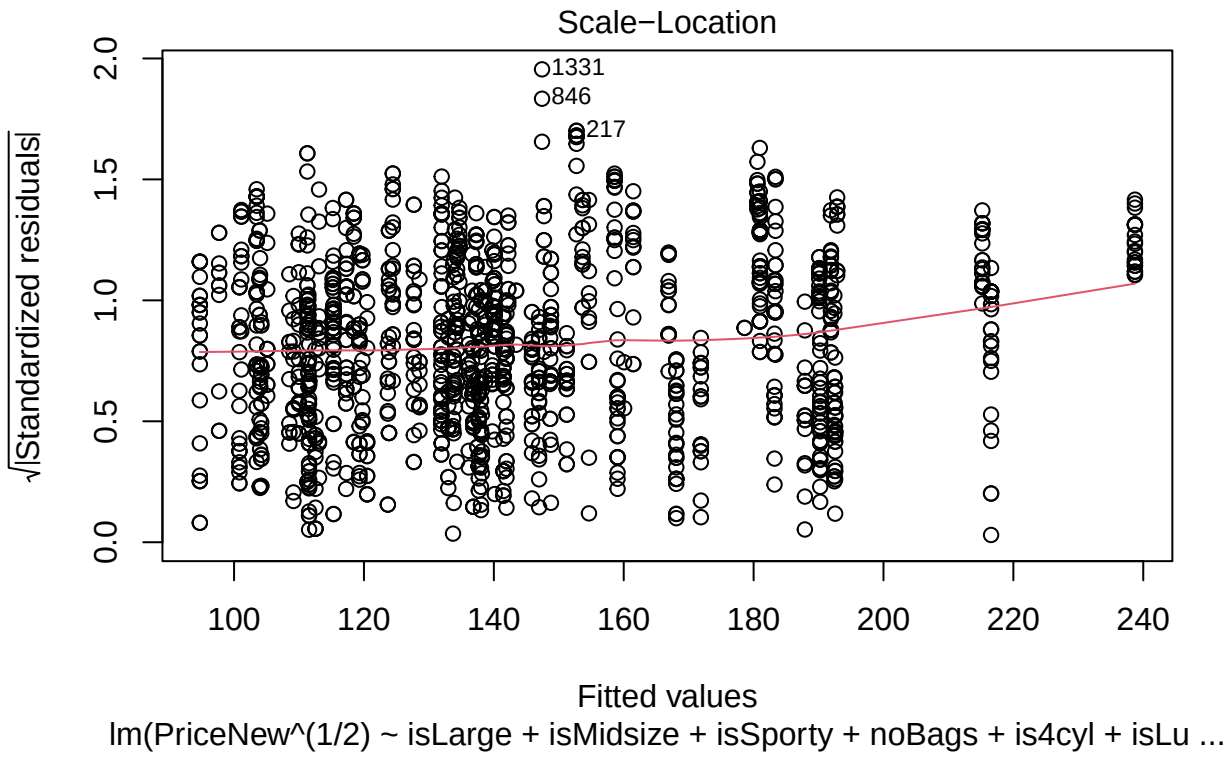
```
##
## Call:
## lm(formula = PriceNew^(1/2) ~ isLarge + isMidsize + isSporty +
##     noBags + is4cyl + isLuxjapan + isMerc + isSE + isGerman +
##     luxury + Luggage.room + Width + Man.trans.avail + Horsepower +
##     Fuel.tank.capacity, data = cars_train)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -24.262  -5.927   0.501   5.584  32.227
##
## Coefficients:
##                      Estimate Std. Error t value Pr(>|t|)
```

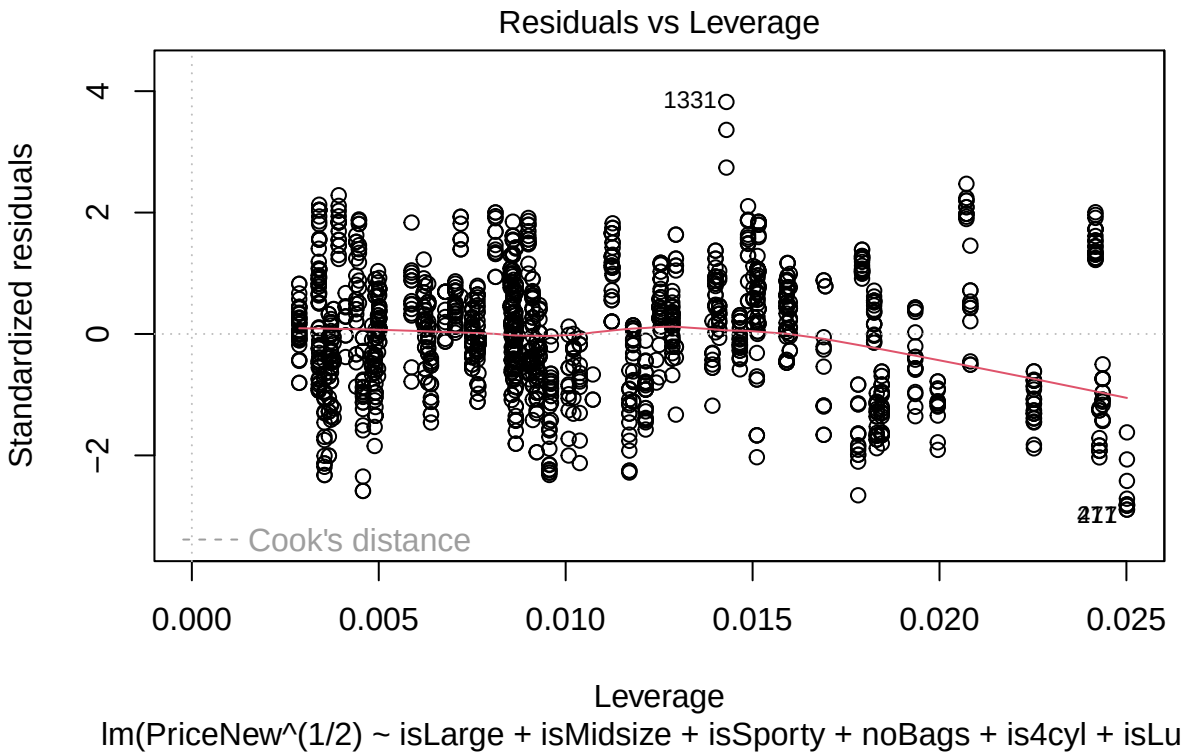
```
## (Intercept)        209.498931   9.473753  22.114  < 2e-16 ***
## isLarge              5.339511   1.137799   4.693 2.94e-06 ***
## isMidsize            9.248703   0.805670  11.480  < 2e-16 ***
## isSporty            10.637201   0.883264  12.043  < 2e-16 ***
## noBags             -11.800996   0.586772 -20.112  < 2e-16 ***
## is4cyl              -4.139443   0.700687  -5.908 4.30e-09 ***
## isLuxjapan          -8.841686   1.326581  -6.665 3.72e-11 ***
## isMerc              76.245411   1.830505  41.653  < 2e-16 ***
## isSE                20.486729   1.021848  20.049  < 2e-16 ***
## isGerman           -10.203619   1.214133  -8.404  < 2e-16 ***
## luxury              33.925308   1.048332  32.361  < 2e-16 ***
## Luggage.room         0.742155   0.130917   5.669 1.72e-08 ***
## Width               -2.369071   0.163329 -14.505  < 2e-16 ***
## Man.trans.availYes  -4.614912   0.790879  -5.835 6.59e-09 ***
## Horsepower           0.208289   0.008926  23.334  < 2e-16 ***
## Fuel.tank.capacity   3.300898   0.173573  19.017  < 2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 8.493 on 1481 degrees of freedom
## Multiple R-squared:  0.9348, Adjusted R-squared:  0.9342
## F-statistic:  1416 on 15 and 1481 DF,  p-value: < 2.2e-16
```

```
plot(rgs_test)
```




Normal Q−Q
Standardized residuals
Theoretical Quantiles
lm(PriceNew^(1/2) ~ isLarge + isMidsize + isSporty + noBags + is4cyl + isLu ...
1331
846
217
−3
−2
−1
0
1
2
3
4


Scale−Location
√|Standardized residuals|
0.0
0.5
1.0
1.5
2.0
1331
846
217
100
120
140
160
180
200
220
240
Fitted values
lm(PriceNew^(1/2) ~ isLarge + isMidsize + isSporty + noBags + is4cyl + isLu ...




We can see from our diagnostic plots that the model meets our assumptions of normality and constant variance for multiple linear regression, which means the model is valid. In addition, we have a very decent R-squared of 0.9348, a significant p-value, and a very large F-score, using only 15 predictors for a complexity score of 95%.