Final Project DSC 520

Reenie Christudass

2022-08-13

Contents

Introduction - MULTIPLE LINEAR REGRESSION 2
Research questions: 2
Load libraries as needed 2
Identify data set and import 5
Summarize the variable (univariate analysis) 5
Identify potential predictor variable and dependent variable 6
Convert categorical variables 6
Data transformation and cleaning 7
Check for number of nulls in each column and fill in NA values by finding mean 8
Correlations between numeric variables in our new data frame 10
plotting the scatter plot 13
Perform Linear Regression with one Predictors 14
Check for multicollinearity 19
Identifying Outliers in Linear Regression — Cook's Distance 20
Dataset - Selected Vehicle data 24
1.Medical insurance (https://www.kaggle.com/datasets/mirichoi0218/insurance) 24
2. Death rate from Cancer (https://data.world/nrippner/cancer-trials) 24
3. Vehicle data (https://www.kaggle.com/datasets/nehalbirla/vehicle-dataset-from-cardekho) 24
References 24

Introduction - MULTIPLE LINEAR REGRESSION

Any organization can make better decisions by using regression techniques for predictive analysis. This statistical technique can help organizations to make better decisions. Relationships between data can transform analysis into actionable information. Linear regression can fit in one dependent and one independent variable and has significant limits. In comparison, Multiple regression can overcome and fit in single dependent and multiple independent variables. This statistical techniques is used for forecasting, time series modelling and find relationship between variable.

In real life scenario, I am interested in finding if the selling price of car is solely based on the miles driven of them or it depends on the other predictor factors like (owner, fuel, mileage , transmission etc)

Research questions:

Load libraries as needed

```
if(!require('foreign')) {
  install.packages('foreign')
  library('foreign')
}
```

```
## Loading required package: foreign
```

```
if(!require('tidyr')) {
  install.packages('tidyr')
  library('tidyr')
}
```

```
## Loading required package: tidyr

## Warning: package 'tidyr' was built under R version 4.2.1
```

```
if(!require('dplyr')) {
  install.packages('dplyr')
  library('dplyr')
}
```

```
## Loading required package: dplyr

## Warning: package 'dplyr' was built under R version 4.2.1

##
## 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
```

```r
if(!require('corrr')) {
  install.packages('corrr')
  library('corrr')
}
```

```
## Loading required package: corrr

## Warning: package 'corrr' was built under R version 4.2.1
```

```r
install.packages("MASS", repos="http://cran.us.r-project.org")
```

```
## Installing package into 'C:/Users/chris/AppData/Local/R/win-library/4.2'
## (as 'lib' is unspecified)

## package 'MASS' successfully unpacked and MD5 sums checked

## Warning: cannot remove prior installation of package 'MASS'

## Warning in file.copy(savedcopy, lib, recursive = TRUE): problem copying C:
## \Users\chris\AppData\Local\R\win-library\4.2\00LOCK\MASS\libs\x64\MASS.dll to C:
## \Users\chris\AppData\Local\R\win-library\4.2\MASS\libs\x64\MASS.dll: Permission
## denied

## Warning: restored 'MASS'

##
## The downloaded binary packages are in
##  C:\Users\chris\AppData\Local\Temp\RtmpaakjBi\downloaded_packages
```

```r
library(MASS)
```

```
## Warning: package 'MASS' was built under R version 4.2.1

##
## Attaching package: 'MASS'

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

```r
install.packages("corrplot ", repos="http://cran.us.r-project.org")
```

```
## Installing package into 'C:/Users/chris/AppData/Local/R/win-library/4.2'
## (as 'lib' is unspecified)

## Warning: package 'corrplot ' is not available for this version of R
##
## A version of this package for your version of R might be available elsewhere,
## see the ideas at
## https://cran.r-project.org/doc/manuals/r-patched/R-admin.html#Installing-packages
```

```
library(corrplot )
```

```
## Warning: package 'corrplot' was built under R version 4.2.1
```

```
## corrplot 0.92 loaded
```

```
install.packages("olsrr", repos="http://cran.us.r-project.org")
```

```
## Installing package into 'C:/Users/chris/AppData/Local/R/win-library/4.2'
## (as 'lib' is unspecified)
```

```
## package 'olsrr' successfully unpacked and MD5 sums checked
```

```
## Warning: cannot remove prior installation of package 'olsrr'
```

```
## Warning in file.copy(savedcopy, lib, recursive = TRUE): problem copying C:
## \Users\chris\AppData\Local\R\win-library\4.2\00LOCK\olsrr\libs\x64\olsrr.dll
## to C:\Users\chris\AppData\Local\R\win-library\4.2\olsrr\libs\x64\olsrr.dll:
## Permission denied
```

```
## Warning: restored 'olsrr'
```

```
##
## The downloaded binary packages are in
##  C:\Users\chris\AppData\Local\Temp\RtmpaakjBi\downloaded_packages
```

```
library(olsrr)
```

```
## Warning: package 'olsrr' was built under R version 4.2.1
```

```
##
## Attaching package: 'olsrr'
```

```
## The following object is masked from 'package:MASS':
##
##     cement
```

```
## The following object is masked from 'package:datasets':
##
##     rivers
```

```
install.packages("ggcorrplot", repos="http://cran.us.r-project.org")
```

```
## Installing package into 'C:/Users/chris/AppData/Local/R/win-library/4.2'
## (as 'lib' is unspecified)
```

```
## package 'ggcorrplot' successfully unpacked and MD5 sums checked
##
## The downloaded binary packages are in
##  C:\Users\chris\AppData\Local\Temp\RtmpaakjBi\downloaded_packages
```

```
library(ggcorrplot)
```

```
## Warning: package 'ggcorrplot' was built under R version 4.2.1
```

```
## Loading required package: ggplot2
```

```
## Warning: package 'ggplot2' was built under R version 4.2.1
```

Identify data set and import

```
## Set the working directory to the root of your DSC 520 directory
setwd("C:/Users/chris/dsc520/data")
```

```
## Load the `data/r4ds/heights.csv` to
df <- read.csv("C:/Users/chris/dsc520/Final_Project_Data/Car_details_v3.csv")
head(df)
```

```
##                           name year selling_price km_driven   fuel seller_type
## 1        Maruti Swift Dzire VDI 2014        450000    145500 Diesel  Individual
## 2  Skoda Rapid 1.5 TDI Ambition 2014        370000    120000 Diesel  Individual
## 3      Honda City 2017-2020 EXi 2006        158000    140000 Petrol  Individual
## 4     Hyundai i20 Sportz Diesel 2010        225000    127000 Diesel  Individual
## 5        Maruti Swift VXI BSIII 2007        130000    120000 Petrol  Individual
## 6 Hyundai Xcent 1.2 VTVT E Plus 2017        440000     45000 Petrol  Individual
##   transmission        owner    mileage  engine  max_power
## 1       Manual  First Owner  23.4 kmpl 1248 CC     74 bhp
## 2       Manual Second Owner 21.14 kmpl 1498 CC 103.52 bhp
## 3       Manual  Third Owner  17.7 kmpl 1497 CC     78 bhp
## 4       Manual  First Owner  23.0 kmpl 1396 CC     90 bhp
## 5       Manual  First Owner  16.1 kmpl 1298 CC   88.2 bhp
## 6       Manual  First Owner 20.14 kmpl 1197 CC  81.86 bhp
##                     torque seats
## 1           190Nm@ 2000rpm     5
## 2      250Nm@ 1500-2500rpm     5
## 3    12.7@ 2,700(kgm@ rpm)     5
## 4 22.4 kgm at 1750-2750rpm     5
## 5    11.5@ 4,500(kgm@ rpm)     5
## 6        113.75nm@ 4000rpm     5
```

Summarize the variable (univariate analysis)

```
## Summarize the variable (univariate analysis)
summary(df)
```

```
##      name                year      selling_price        km_driven
##  Length:8128        Min.   :1983   Min.   :   29999   Min.   :      1
##  Class :character   1st Qu.:2011   1st Qu.:  254999   1st Qu.:  35000
##  Mode  :character   Median :2015   Median :  450000   Median :  60000
```

```
##                     Mean   :2014   Mean   :  638272   Mean   :  69820
##                     3rd Qu.:2017   3rd Qu.:  675000   3rd Qu.:  98000
##                     Max.   :2020   Max.   :10000000   Max.   :2360457
##
##      fuel           seller_type        transmission          owner
##  Length:8128        Length:8128        Length:8128        Length:8128
##  Class :character   Class :character   Class :character   Class :character
##  Mode  :character   Mode  :character   Mode  :character   Mode  :character
##
##
##
##
##    mileage            engine           max_power            torque
##  Length:8128        Length:8128        Length:8128        Length:8128
##  Class :character   Class :character   Class :character   Class :character
##  Mode  :character   Mode  :character   Mode  :character   Mode  :character
##
##
##
##
##      seats
##  Min.   : 2.000
##  1st Qu.: 5.000
##  Median : 5.000
##  Mean   : 5.417
##  3rd Qu.: 5.000
##  Max.   :14.000
##  NA's   :221
```

Identify potential predictor variable and dependent variable

```
    ## One dependent variable - Price of the car
    ## Multiple Independent variable - year, selling_price, km_driven, fuel, seller_type, transmission,
```

Convert categorical variables

```
df <- transform(df,fuel = as.numeric(as.factor(fuel)))
df <- transform(df,seller_type = as.numeric(as.factor(seller_type)))
df <- transform(df,transmission = as.numeric(as.factor(transmission)))
df <- transform(df,owner = as.numeric(as.factor(owner)))
head(df)
```

```
##                           name year selling_price km_driven fuel seller_type
## 1        Maruti Swift Dzire VDI 2014        450000    145500    2           2
## 2  Skoda Rapid 1.5 TDI Ambition 2014        370000    120000    2           2
## 3      Honda City 2017-2020 EXi 2006        158000    140000    4           2
## 4     Hyundai i20 Sportz Diesel 2010        225000    127000    2           2
## 5        Maruti Swift VXI BSIII 2007        130000    120000    4           2
## 6 Hyundai Xcent 1.2 VTVT E Plus 2017        440000     45000    4           2
##   transmission owner    mileage  engine  max_power                  torque
```

```
## 1           2     1  23.4 kmpl 1248 CC     74 bhp           190Nm@ 2000rpm
## 2           2     3 21.14 kmpl 1498 CC 103.52 bhp      250Nm@ 1500-2500rpm
## 3           2     5  17.7 kmpl 1497 CC     78 bhp    12.7@ 2,700(kgm@ rpm)
## 4           2     1  23.0 kmpl 1396 CC     90 bhp 22.4 kgm at 1750-2750rpm
## 5           2     1  16.1 kmpl 1298 CC   88.2 bhp    11.5@ 4,500(kgm@ rpm)
## 6           2     1 20.14 kmpl 1197 CC  81.86 bhp        113.75nm@ 4000rpm
##   seats
## 1     5
## 2     5
## 3     5
## 4     5
## 5     5
## 6     5
```

Data transformation and cleaning

```
# Converting column km/kg to kmpl. created new column in dataframe with modififed values
df$mileage_modified <- df$mileage
df$mileage_modified <- gsub("kmpl","",as.character(df$mileage_modified))
df$mileage_modified <- gsub("km/kg","",as.character(df$mileage_modified))
df$mileage_modified <- as.double(df$mileage_modified)
df$mileage_modified[endsWith(df$mileage, "km/kg")] <- as.numeric(df$mileage_modified) * 1.40
```

```
## Warning in df$mileage_modified[endsWith(df$mileage, "km/kg")] <-
## as.numeric(df$mileage_modified) * : number of items to replace is not a multiple
## of replacement length
```

```
# created new column in dataframe with modififed values for engine
df$engine_modified <- gsub("CC","",as.character(df$engine))
df$engine_modified <- as.double(df$engine_modified)
# created new column in dataframe with modififed values for max_power
df$max_power_modified <-gsub("bhp","",as.character(df$max_power))
df$max_power_modified <- as.double(df$max_power_modified)
head(df)
```

```
##                           name year selling_price km_driven fuel seller_type
## 1        Maruti Swift Dzire VDI 2014        450000    145500    2           2
## 2  Skoda Rapid 1.5 TDI Ambition 2014        370000    120000    2           2
## 3      Honda City 2017-2020 EXi 2006        158000    140000    4           2
## 4     Hyundai i20 Sportz Diesel 2010        225000    127000    2           2
## 5        Maruti Swift VXI BSIII 2007        130000    120000    4           2
## 6 Hyundai Xcent 1.2 VTVT E Plus 2017        440000     45000    4           2
##   transmission owner    mileage  engine  max_power                  torque
## 1            2     1  23.4 kmpl 1248 CC     74 bhp           190Nm@ 2000rpm
## 2            2     3 21.14 kmpl 1498 CC 103.52 bhp      250Nm@ 1500-2500rpm
## 3            2     5  17.7 kmpl 1497 CC     78 bhp    12.7@ 2,700(kgm@ rpm)
## 4            2     1  23.0 kmpl 1396 CC     90 bhp 22.4 kgm at 1750-2750rpm
## 5            2     1  16.1 kmpl 1298 CC   88.2 bhp    11.5@ 4,500(kgm@ rpm)
## 6            2     1 20.14 kmpl 1197 CC  81.86 bhp        113.75nm@ 4000rpm
##   seats mileage_modified engine_modified max_power_modified
## 1     5            23.40            1248              74.00
```

```
## 2     5            21.14            1498             103.52
## 3     5            17.70            1497              78.00
## 4     5            23.00            1396              90.00
## 5     5            16.10            1298              88.20
## 6     5            20.14            1197              81.86
```

Check for number of nulls in each column and fill in NA values by finding mean

```
colSums(is.na(df))
```

```
##               name               year      selling_price          km_driven
##                  0                  0                  0                  0
##               fuel        seller_type       transmission              owner
##                  0                  0                  0                  0
##            mileage             engine          max_power             torque
##                  0                  0                  0                  0
##              seats   mileage_modified    engine_modified max_power_modified
##                221                225                221                216
```

```
# Return the column names containing missing observations
list_na <- colnames(df)[ apply(df, 2, anyNA) ]
list_na
```

```
## [1] "seats"              "mileage_modified"   "engine_modified"
## [4] "max_power_modified"
```

```
# Exclude the missing observations
library(dplyr)
# Exclude the missing observations
df_drop <-df %>%
na.omit()

head(df)
```

```
##                           name year selling_price km_driven fuel seller_type
## 1        Maruti Swift Dzire VDI 2014        450000    145500    2           2
## 2  Skoda Rapid 1.5 TDI Ambition 2014        370000    120000    2           2
## 3      Honda City 2017-2020 EXi 2006        158000    140000    4           2
## 4     Hyundai i20 Sportz Diesel 2010        225000    127000    2           2
## 5        Maruti Swift VXI BSIII 2007        130000    120000    4           2
## 6 Hyundai Xcent 1.2 VTVT E Plus 2017        440000     45000    4           2
##   transmission owner    mileage  engine  max_power                  torque
## 1            2     1  23.4 kmpl 1248 CC     74 bhp           190Nm@ 2000rpm
## 2            2     3 21.14 kmpl 1498 CC 103.52 bhp      250Nm@ 1500-2500rpm
## 3            2     5  17.7 kmpl 1497 CC     78 bhp    12.7@ 2,700(kgm@ rpm)
## 4            2     1  23.0 kmpl 1396 CC     90 bhp 22.4 kgm at 1750-2750rpm
## 5            2     1  16.1 kmpl 1298 CC   88.2 bhp    11.5@ 4,500(kgm@ rpm)
## 6            2     1 20.14 kmpl 1197 CC  81.86 bhp        113.75nm@ 4000rpm
##   seats mileage_modified engine_modified max_power_modified
## 1     5            23.40            1248              74.00
## 2     5            21.14            1498             103.52
```

```
## 3     5            17.70            1497             78.00
## 4     5            23.00            1396             90.00
## 5     5            16.10            1298             88.20
## 6     5            20.14            1197             81.86
```

```
# Create mean
average_missing <- apply(df[,colnames(df) %in% list_na], 2, mean,na.rm =  TRUE)


# Create a new variable with the mean
df_replace <- df %>%
   mutate(replace_mean_seats  = ifelse(is.na(seats), average_missing[1], seats),
   replace_mean_mileage_modified = ifelse(is.na(mileage_modified), average_missing[2], mileage_modified
   replace_mean_engine_modified = ifelse(is.na(engine_modified), average_missing[3], engine_modified),
   replace_mean_max_power_modified = ifelse(is.na(max_power_modified), average_missing[4], max_power_mo
   )

head(df_replace)
```

```
##                           name year selling_price km_driven fuel seller_type
## 1        Maruti Swift Dzire VDI 2014        450000    145500    2           2
## 2  Skoda Rapid 1.5 TDI Ambition 2014        370000    120000    2           2
## 3      Honda City 2017-2020 EXi 2006        158000    140000    4           2
## 4     Hyundai i20 Sportz Diesel 2010        225000    127000    2           2
## 5        Maruti Swift VXI BSIII 2007        130000    120000    4           2
## 6 Hyundai Xcent 1.2 VTVT E Plus 2017        440000     45000    4           2
##   transmission owner    mileage  engine  max_power                 torque
## 1            2     1  23.4 kmpl 1248 CC     74 bhp           190Nm@ 2000rpm
## 2            2     3 21.14 kmpl 1498 CC 103.52 bhp      250Nm@ 1500-2500rpm
## 3            2     5  17.7 kmpl 1497 CC     78 bhp    12.7@ 2,700(kgm@ rpm)
## 4            2     1  23.0 kmpl 1396 CC     90 bhp 22.4 kgm at 1750-2750rpm
## 5            2     1  16.1 kmpl 1298 CC   88.2 bhp    11.5@ 4,500(kgm@ rpm)
## 6            2     1 20.14 kmpl 1197 CC  81.86 bhp        113.75nm@ 4000rpm
##   seats mileage_modified engine_modified max_power_modified replace_mean_seats
## 1     5            23.40            1248              74.00                  5
## 2     5            21.14            1498             103.52                  5
## 3     5            17.70            1497              78.00                  5
## 4     5            23.00            1396              90.00                  5
## 5     5            16.10            1298              88.20                  5
## 6     5            20.14            1197              81.86                  5
##   replace_mean_mileage_modified replace_mean_engine_modified
## 1                         23.40                          1248
## 2                         21.14                          1498
## 3                         17.70                          1497
## 4                         23.00                          1396
## 5                         16.10                          1298
## 6                         20.14                          1197
##   replace_mean_max_power_modified
## 1                           74.00
## 2                          103.52
## 3                           78.00
## 4                           90.00
## 5                           88.20
## 6                           81.86
```

```
df_new <- subset(df_replace, select = c(name,year,selling_price,km_driven,fuel,seller_type,transmission
head(df_new)
```

```
##                           name year selling_price km_driven fuel seller_type
## 1        Maruti Swift Dzire VDI 2014        450000    145500    2           2
## 2  Skoda Rapid 1.5 TDI Ambition 2014        370000    120000    2           2
## 3      Honda City 2017-2020 EXi 2006        158000    140000    4           2
## 4     Hyundai i20 Sportz Diesel 2010        225000    127000    2           2
## 5        Maruti Swift VXI BSIII 2007        130000    120000    4           2
## 6 Hyundai Xcent 1.2 VTVT E Plus 2017        440000     45000    4           2
##   transmission owner replace_mean_seats replace_mean_mileage_modified
## 1            2     1                  5                         23.40
## 2            2     3                  5                         21.14
## 3            2     5                  5                         17.70
## 4            2     1                  5                         23.00
## 5            2     1                  5                         16.10
## 6            2     1                  5                         20.14
##   replace_mean_engine_modified replace_mean_max_power_modified
## 1                         1248                           74.00
## 2                         1498                          103.52
## 3                         1497                           78.00
## 4                         1396                           90.00
## 5                         1298                           88.20
## 6                         1197                           81.86
```

Correlations between numeric variables in our new data frame

```
## Identify columns in dataframe that are numeric

df <- select_if(df_new, is.numeric)
head(df)
```

```
##   year selling_price km_driven fuel seller_type transmission owner
## 1 2014        450000    145500    2           2            2     1
## 2 2014        370000    120000    2           2            2     3
## 3 2006        158000    140000    4           2            2     5
## 4 2010        225000    127000    2           2            2     1
## 5 2007        130000    120000    4           2            2     1
## 6 2017        440000     45000    4           2            2     1
##   replace_mean_seats replace_mean_mileage_modified replace_mean_engine_modified
## 1                  5                         23.40                         1248
## 2                  5                         21.14                         1498
## 3                  5                         17.70                         1497
## 4                  5                         23.00                         1396
## 5                  5                         16.10                         1298
## 6                  5                         20.14                         1197
##   replace_mean_max_power_modified
## 1                           74.00
## 2                          103.52
## 3                           78.00
## 4                           90.00
```

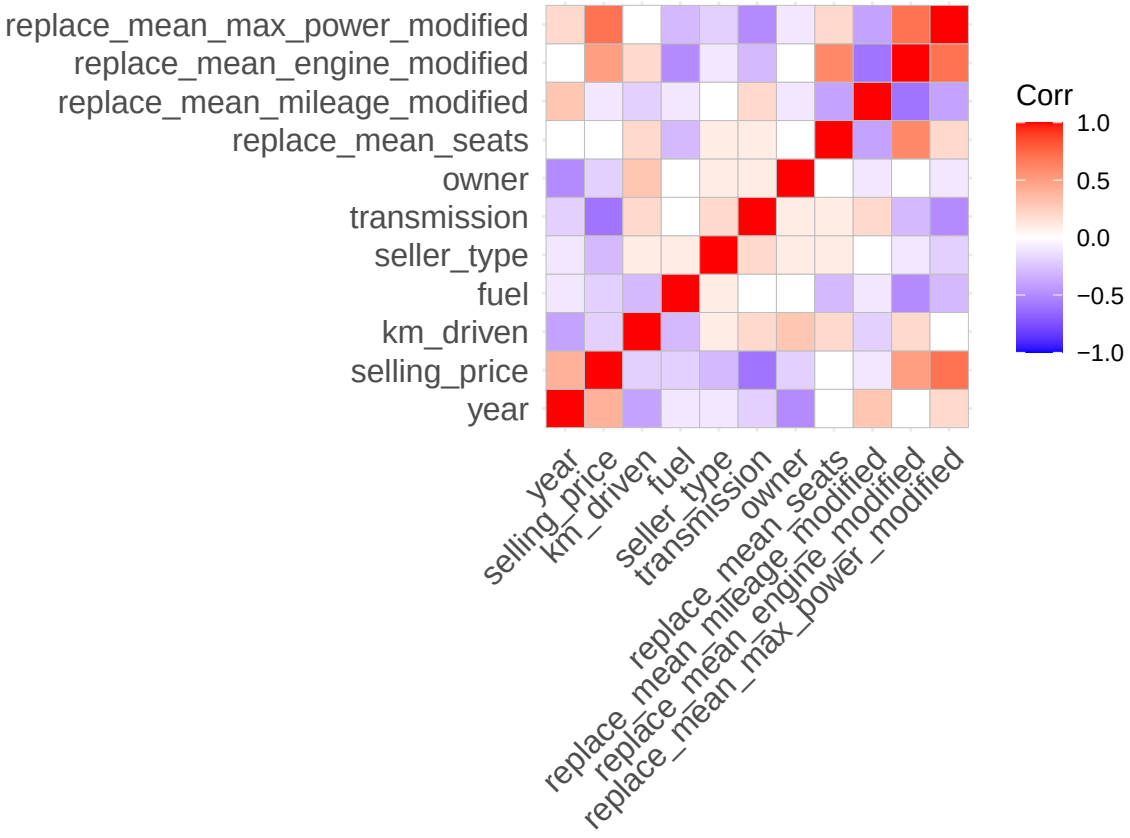
```
## 5                           88.20
## 6                           81.86
```

```
## Correlation
## Correlation in numbers

corr <- round(cor(df), 1)
corr
```

```
##                                year selling_price km_driven fuel seller_type
## year                            1.0           0.4      -0.4 -0.1        -0.1
## selling_price                   0.4           1.0      -0.2 -0.2        -0.3
## km_driven                      -0.4          -0.2       1.0 -0.3         0.1
## fuel                           -0.1          -0.2      -0.3  1.0         0.1
## seller_type                    -0.1          -0.3       0.1  0.1         1.0
## transmission                   -0.2          -0.6       0.2  0.0         0.2
## owner                          -0.5          -0.2       0.3  0.0         0.1
## replace_mean_seats              0.0           0.0       0.2 -0.3         0.1
## replace_mean_mileage_modified   0.3          -0.1      -0.2 -0.1         0.0
## replace_mean_engine_modified    0.0           0.5       0.2 -0.5        -0.1
## replace_mean_max_power_modified 0.2           0.7       0.0 -0.3        -0.2
##                                transmission owner replace_mean_seats
## year                                   -0.2  -0.5                0.0
## selling_price                          -0.6  -0.2                0.0
## km_driven                               0.2   0.3                0.2
## fuel                                    0.0   0.0               -0.3
## seller_type                             0.2   0.1                0.1
## transmission                            1.0   0.1                0.1
## owner                                   0.1   1.0                0.0
## replace_mean_seats                      0.1   0.0                1.0
## replace_mean_mileage_modified           0.2  -0.1               -0.4
## replace_mean_engine_modified           -0.3   0.0                0.6
## replace_mean_max_power_modified        -0.5  -0.1                0.2
##                                replace_mean_mileage_modified
## year                                                     0.3
## selling_price                                           -0.1
## km_driven                                               -0.2
## fuel                                                    -0.1
## seller_type                                              0.0
## transmission                                             0.2
## owner                                                   -0.1
## replace_mean_seats                                      -0.4
## replace_mean_mileage_modified                            1.0
## replace_mean_engine_modified                            -0.6
## replace_mean_max_power_modified                         -0.4
##                                replace_mean_engine_modified
## year                                                    0.0
## selling_price                                           0.5
## km_driven                                               0.2
## fuel                                                   -0.5
## seller_type                                            -0.1
## transmission                                           -0.3
## owner                                                   0.0
## replace_mean_seats                                      0.6
```

```
## replace_mean_mileage_modified                          -0.6
## replace_mean_engine_modified                            1.0
## replace_mean_max_power_modified                         0.7
##                                 replace_mean_max_power_modified
## year                                                        0.2
## selling_price                                               0.7
## km_driven                                                   0.0
## fuel                                                       -0.3
## seller_type                                                -0.2
## transmission                                               -0.5
## owner                                                      -0.1
## replace_mean_seats                                          0.2
## replace_mean_mileage_modified                              -0.4
## replace_mean_engine_modified                                0.7
## replace_mean_max_power_modified                             1.0
```

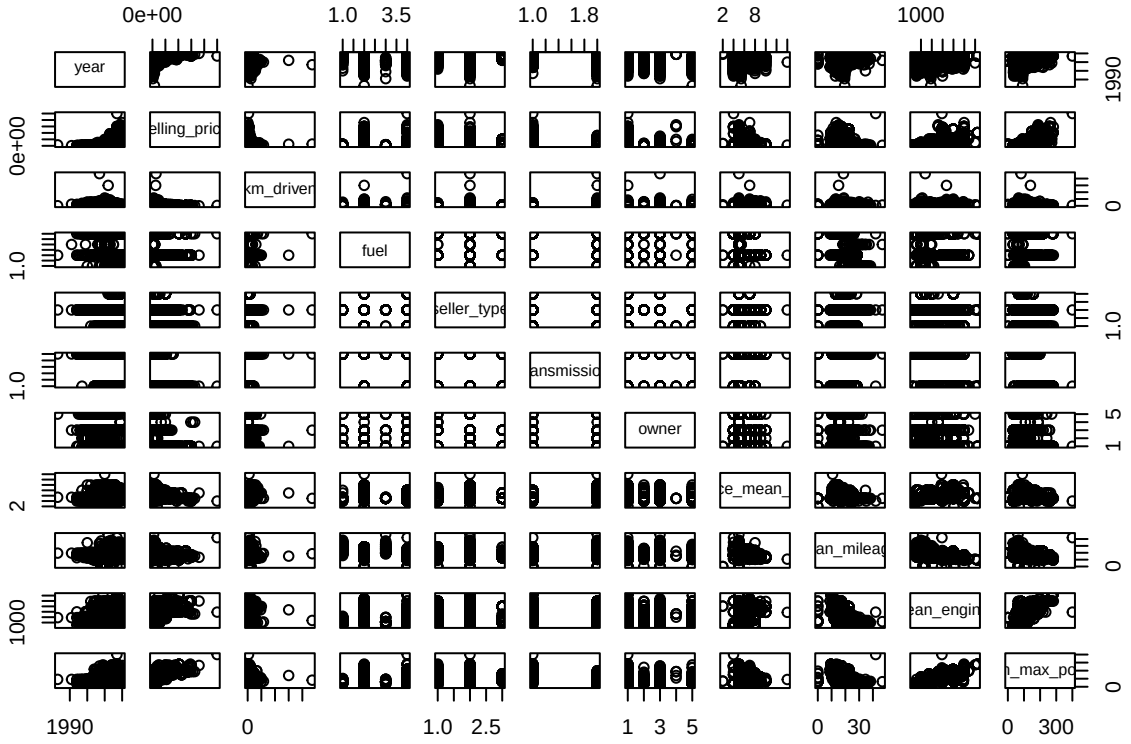
```
## Correlation in graph
ggcorrplot(corr)
```




```
#CONCLUSION:  Variable with highest correlation is a good predictor( > .5)
```

plotting the scatter plot

```
### Check the relationship using scatter plot

plot(df)
```



Scale the dataframe values

```
#scale values in each column of data frame
df_scaled <- scale(df)

#view scaled data frame
head(df_scaled)
```

```
##          year selling_price  km_driven       fuel seller_type transmission
## 1  0.04846121    -0.2335144  1.3382802 -0.8890607   0.2775901    0.3851347
## 2  0.04846121    -0.3327388  0.8873563 -0.8890607   0.2775901    0.3851347
## 3 -1.92965648    -0.5956834  1.2410221  1.1063240   0.2775901    0.3851347
## 4 -0.94059764    -0.5125830  1.0111393 -0.8890607   0.2775901    0.3851347
## 5 -1.68239177    -0.6304120  0.8873563  1.1063240   0.2775901    0.3851347
## 6  0.79025534    -0.2459175 -0.4388907  1.1063240   0.2775901    0.3851347
##        owner replace_mean_seats replace_mean_mileage_modified
## 1 -0.6665981         -0.4402971                     0.9628467
## 2  0.9705402         -0.4402971                     0.4072459
## 3  2.6076785         -0.4402971                    -0.4384473
```

```
## 4 -0.6665981         -0.4402971                     0.8645103
## 5 -0.6665981         -0.4402971                    -0.8317930
## 6 -0.6665981         -0.4402971                     0.1614049
##   replace_mean_engine_modified replace_mean_max_power_modified
## 1                  -0.42377786                     -0.49565110
## 2                   0.07922253                      0.33958626
## 3                   0.07721053                     -0.38247531
## 4                  -0.12600163                     -0.04294792
## 5                  -0.32317778                     -0.09387703
## 6                  -0.52638994                     -0.27326066
```

Perform Linear Regression with one Predictors

```
df <- data.frame(df_scaled)    # Convert array to data frame
head(df)
```

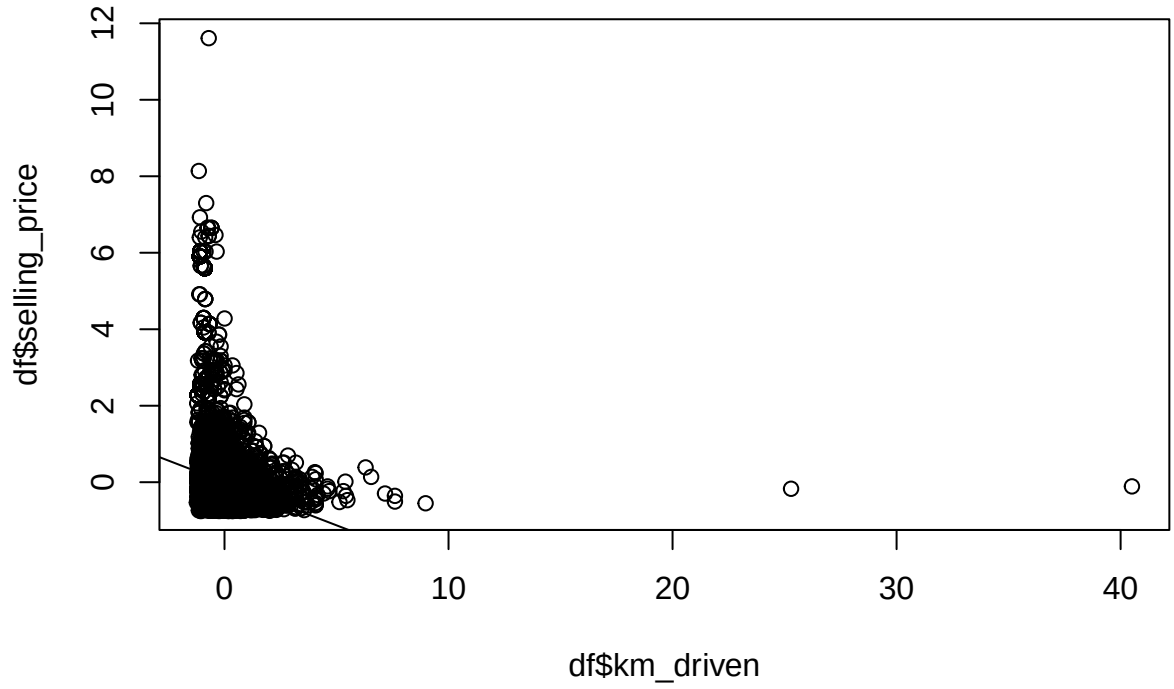
```
##          year selling_price  km_driven       fuel seller_type transmission
## 1  0.04846121    -0.2335144  1.3382802 -0.8890607   0.2775901    0.3851347
## 2  0.04846121    -0.3327388  0.8873563 -0.8890607   0.2775901    0.3851347
## 3 -1.92965648    -0.5956834  1.2410221  1.1063240   0.2775901    0.3851347
## 4 -0.94059764    -0.5125830  1.0111393 -0.8890607   0.2775901    0.3851347
## 5 -1.68239177    -0.6304120  0.8873563  1.1063240   0.2775901    0.3851347
## 6  0.79025534    -0.2459175 -0.4388907  1.1063240   0.2775901    0.3851347
##        owner replace_mean_seats replace_mean_mileage_modified
## 1 -0.6665981         -0.4402971                     0.9628467
## 2  0.9705402         -0.4402971                     0.4072459
## 3  2.6076785         -0.4402971                    -0.4384473
## 4 -0.6665981         -0.4402971                     0.8645103
## 5 -0.6665981         -0.4402971                    -0.8317930
## 6 -0.6665981         -0.4402971                     0.1614049
##   replace_mean_engine_modified replace_mean_max_power_modified
## 1                  -0.42377786                     -0.49565110
## 2                   0.07922253                      0.33958626
## 3                   0.07721053                     -0.38247531
## 4                  -0.12600163                     -0.04294792
## 5                  -0.32317778                     -0.09387703
## 6                  -0.52638994                     -0.27326066
```

```
##Linear Regression model
ln_model <- lm(df,formula=selling_price ~km_driven)
summary(ln_model)
```

```
##
## Call:
## lm(formula = selling_price ~ km_driven, data = df)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -0.9943 -0.4553 -0.2262  0.0468 11.4526
##
## Coefficients:
```

```
##               Estimate Std. Error t value Pr(>|t|)
## (Intercept) -4.121e-16  1.081e-02    0.00        1
## km_driven   -2.255e-01  1.081e-02  -20.87   <2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 0.9743 on 8126 degrees of freedom
## Multiple R-squared:  0.05087,    Adjusted R-squared:  0.05075
## F-statistic: 435.5 on 1 and 8126 DF,  p-value: < 2.2e-16
```

```
# Basic scatterplot
plot(x=df$km_driven, y=df$selling_price)
abline(ln_model)
```



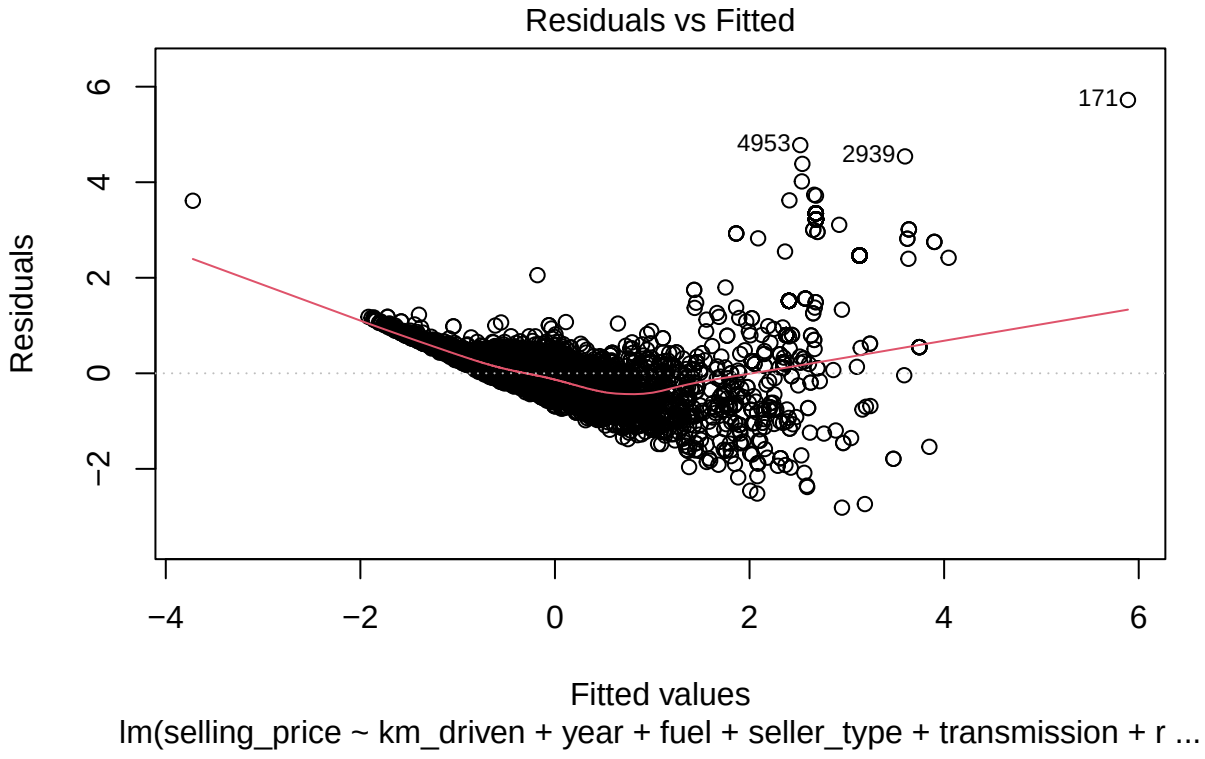
Perform Multiple Linear Regression with All Predictors

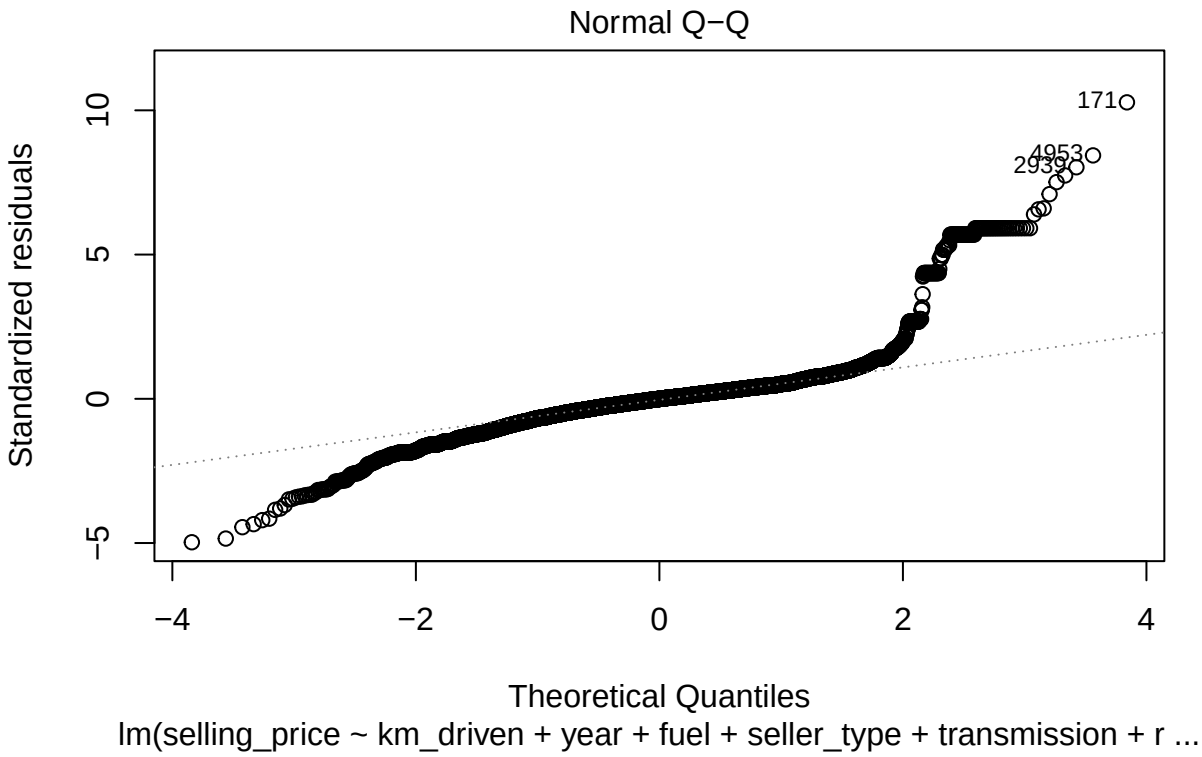
```
##Multiple regression model

ML_model <- lm(df,formula=selling_price ~ km_driven +  year + fuel + seller_type + transmission + repla
summary(ML_model)
```

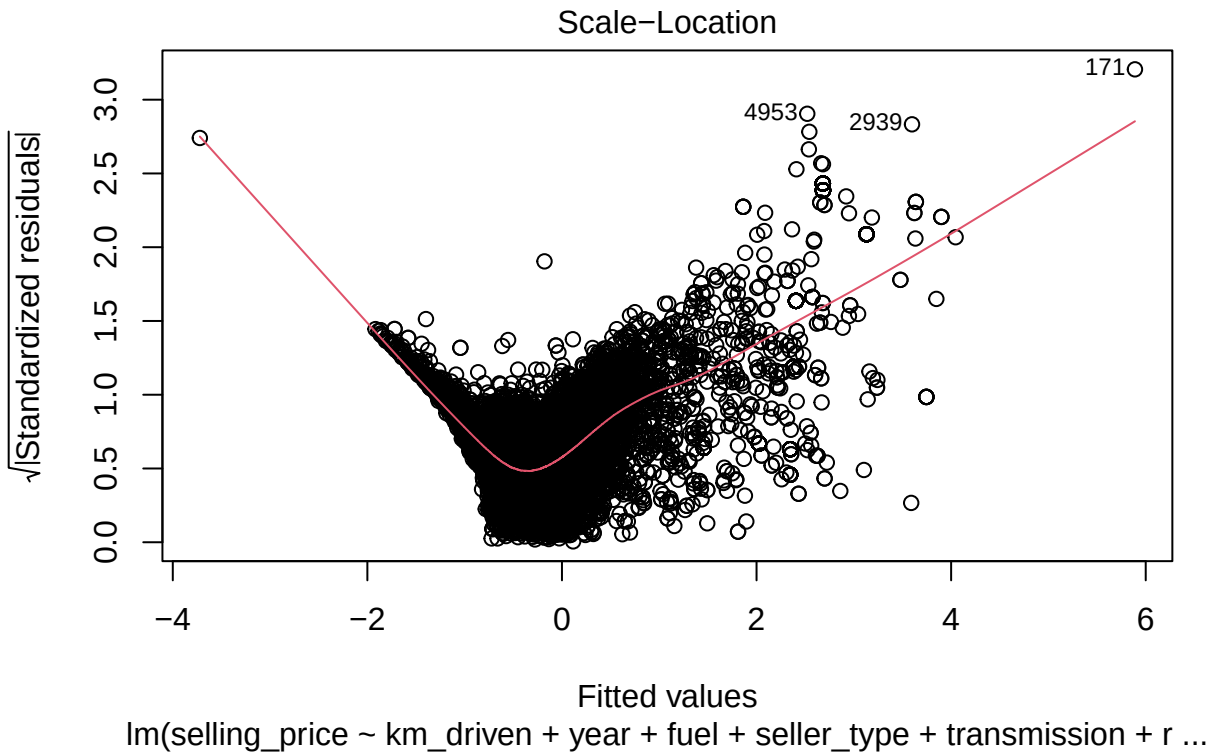
```
##
## Call:
## lm(formula = selling_price ~ km_driven + year + fuel + seller_type +
##     transmission + replace_mean_seats + replace_mean_mileage_modified +
##     replace_mean_engine_modified + replace_mean_max_power_modified,
```

```
##     data = df)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -2.8117 -0.2367 -0.0017  0.1941  5.7221
##
## Coefficients:
##                                 Estimate Std. Error t value Pr(>|t|)
## (Intercept)                    5.415e-16  6.287e-03   0.000  1.00000
## km_driven                     -7.662e-02  7.465e-03 -10.263  < 2e-16 ***
## year                           1.807e-01  7.956e-03  22.707  < 2e-16 ***
## fuel                          -2.906e-02  8.859e-03  -3.280  0.00104 **
## seller_type                   -1.082e-01  6.645e-03 -16.277  < 2e-16 ***
## transmission                  -1.999e-01  7.948e-03 -25.148  < 2e-16 ***
## replace_mean_seats            -4.753e-02  9.221e-03  -5.155 2.60e-07 ***
## replace_mean_mileage_modified  5.755e-02  1.008e-02   5.711 1.16e-08 ***
## replace_mean_engine_modified   6.337e-02  1.435e-02   4.416 1.02e-05 ***
## replace_mean_max_power_modified  5.478e-01  1.118e-02  49.001  < 2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 0.5668 on 8118 degrees of freedom
## Multiple R-squared:  0.6791, Adjusted R-squared:  0.6787
## F-statistic:  1909 on 9 and 8118 DF,  p-value: < 2.2e-16
```

```
plot(ML_model)
```

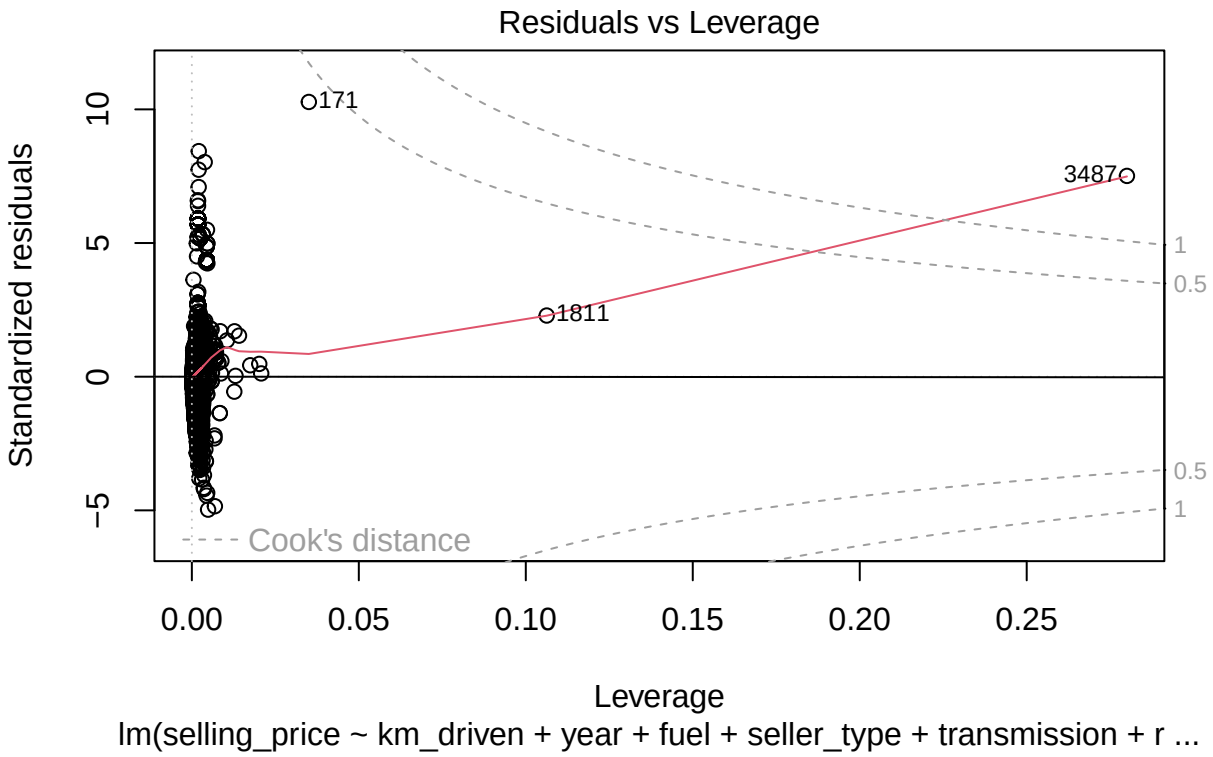



Normal Q−Q
Standardized residuals
10
5
0
−5
171
4953
2939
−4
−2
0
2
4
Theoretical Quantiles
lm(selling_price ~ km_driven + year + fuel + seller_type + transmission + r ...




```
abline(ML_model)
```

```
## Warning in abline(ML_model): only using the first two of 10 regression
## coefficients
```




```
## CONCLUSION: Most of the variable can be participate in the multiple linear
## regression other than owner
```

Check for multicollinearity

```
##TOLERANCE & VARIANCE INFLATION FACTOR (VIF)
library("olsrr")
ols_vif_tol(ML_model)
```

```
##                         Variables Tolerance      VIF
## 1                       km_driven 0.7093337 1.409774
## 2                            year 0.6244896 1.601308
## 3                            fuel 0.5037578 1.985081
## 4                     seller_type 0.8954044 1.116814
## 5                    transmission 0.6258800 1.597750
## 6              replace_mean_seats 0.4649557 2.150742
## 7   replace_mean_mileage_modified 0.3892989 2.568720
## 8    replace_mean_engine_modified 0.1920493 5.206995
## 9 replace_mean_max_power_modified 0.3163419 3.161137
```

```
## CONCLUSION: VIF above 10 shows multicollinearity. I did not notice anything in the dataset
```

Identifying Outliers in Linear Regression — Cook's Distance

```
cooksD <- cooks.distance(ML_model)
influential <- cooksD[(cooksD > (3 * mean(cooksD, na.rm = TRUE)))]
head(influential)
```

```
##         133         136         137         140         149         169
## 0.008524006 0.006694408 0.009305409 0.007988492 0.007681526 0.002957490
```

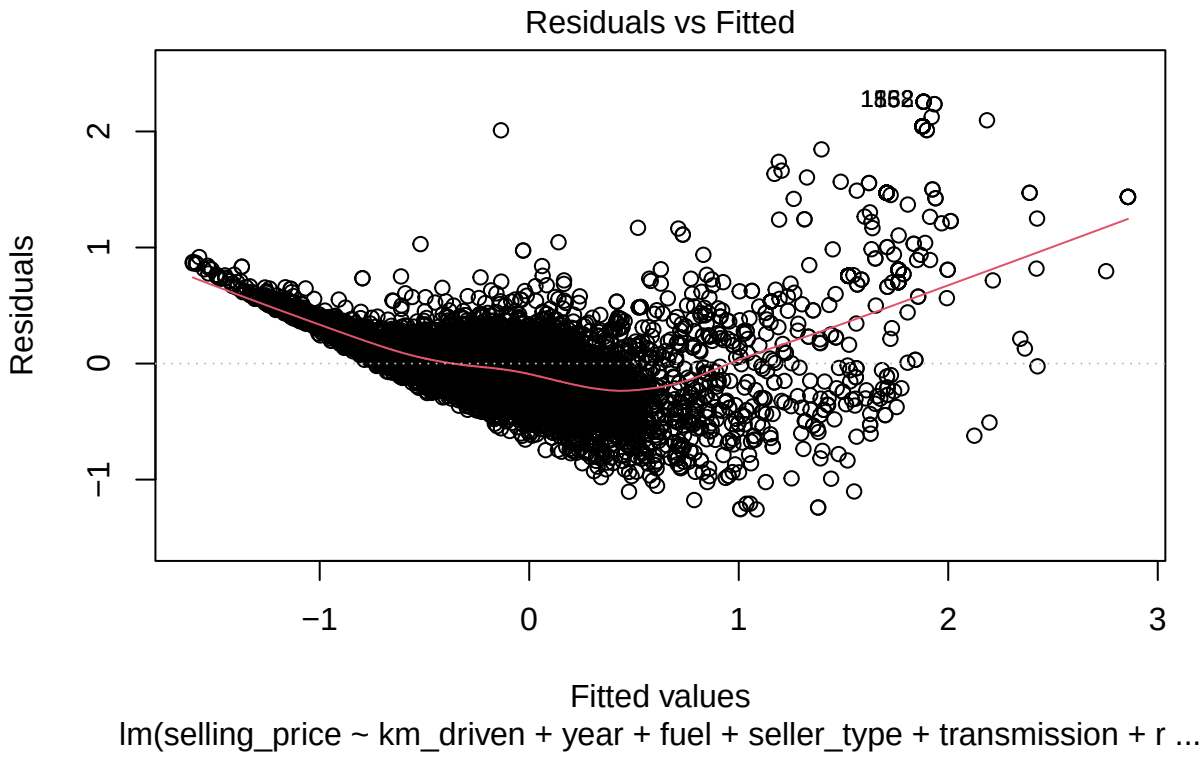
```
names_of_influential <- names(influential)
outliers <- df[names_of_influential,]
hitters_without_outliers <- df %>% anti_join(outliers)
```

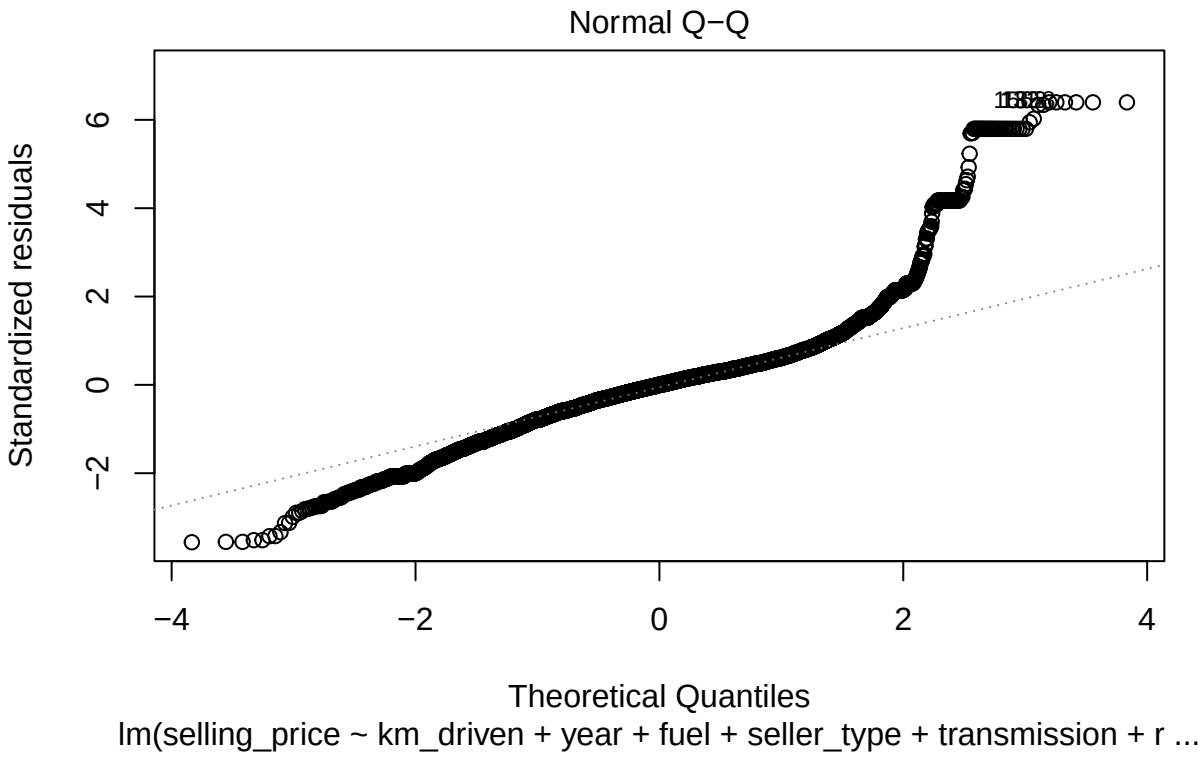
```
## Joining, by = c("year", "selling_price", "km_driven", "fuel", "seller_type",
## "transmission", "owner", "replace_mean_seats", "replace_mean_mileage_modified",
## "replace_mean_engine_modified", "replace_mean_max_power_modified")
```

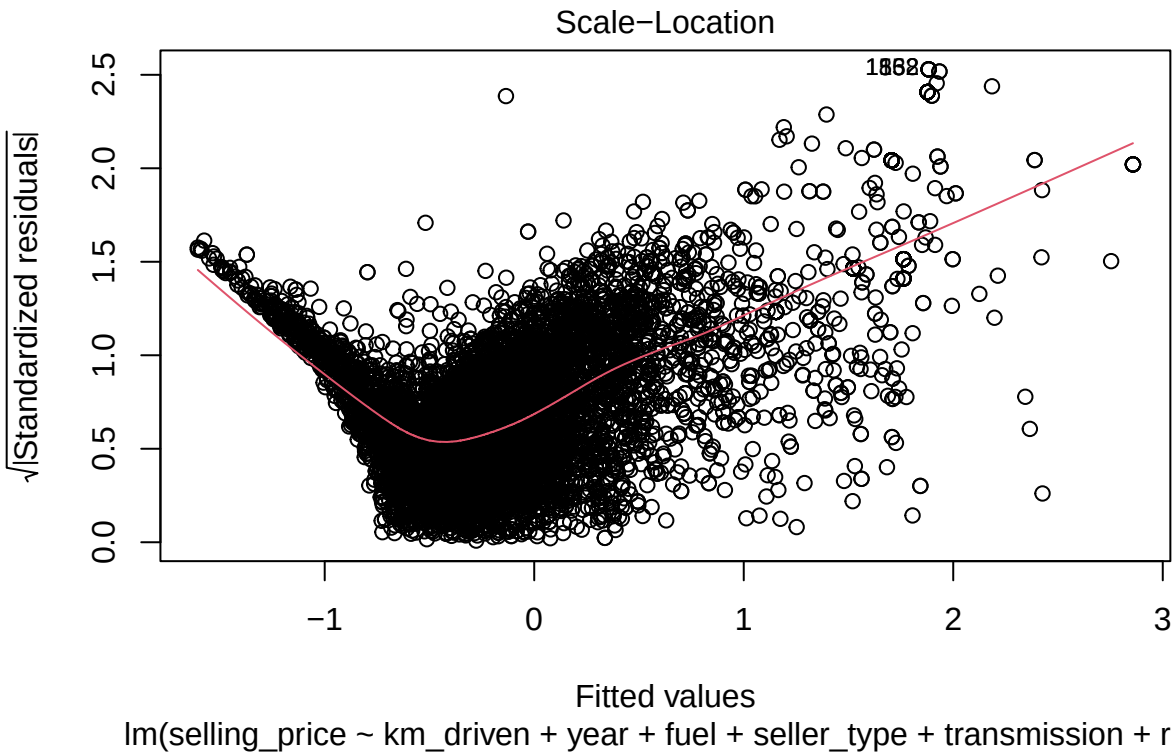
```
ML_model2 <- lm(formula=selling_price ~ km_driven +  year + fuel + seller_type + transmission + replace
summary(ML_model2)
```

```
##
## Call:
## lm(formula = selling_price ~ km_driven + year + fuel + seller_type +
##     transmission + replace_mean_seats + replace_mean_mileage_modified +
##     replace_mean_engine_modified + replace_mean_max_power_modified,
##     data = hitters_without_outliers)
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -1.25673 -0.17841  0.00065  0.14032  2.25610
##
## Coefficients:
##                                 Estimate Std. Error t value Pr(>|t|)
## (Intercept)                    -0.053424   0.003978 -13.428  < 2e-16 ***
## km_driven                      -0.105535   0.006161 -17.128  < 2e-16 ***
## year                            0.159721   0.005280  30.248  < 2e-16 ***
## fuel                           -0.071666   0.005911 -12.123  < 2e-16 ***
## seller_type                    -0.035820   0.004239  -8.451  < 2e-16 ***
## transmission                   -0.159592   0.005029 -31.733  < 2e-16 ***
## replace_mean_seats             -0.034626   0.005821  -5.949 2.82e-09 ***
## replace_mean_mileage_modified   0.007524   0.006644   1.132    0.258
## replace_mean_engine_modified    0.068135   0.009257   7.361 2.01e-13 ***
## replace_mean_max_power_modified  0.388229   0.007386  52.565  < 2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 0.3531 on 7935 degrees of freedom
## Multiple R-squared:  0.7241, Adjusted R-squared:  0.7238
## F-statistic:  2314 on 9 and 7935 DF,  p-value: < 2.2e-16
```

```
plot(ML_model2)
```

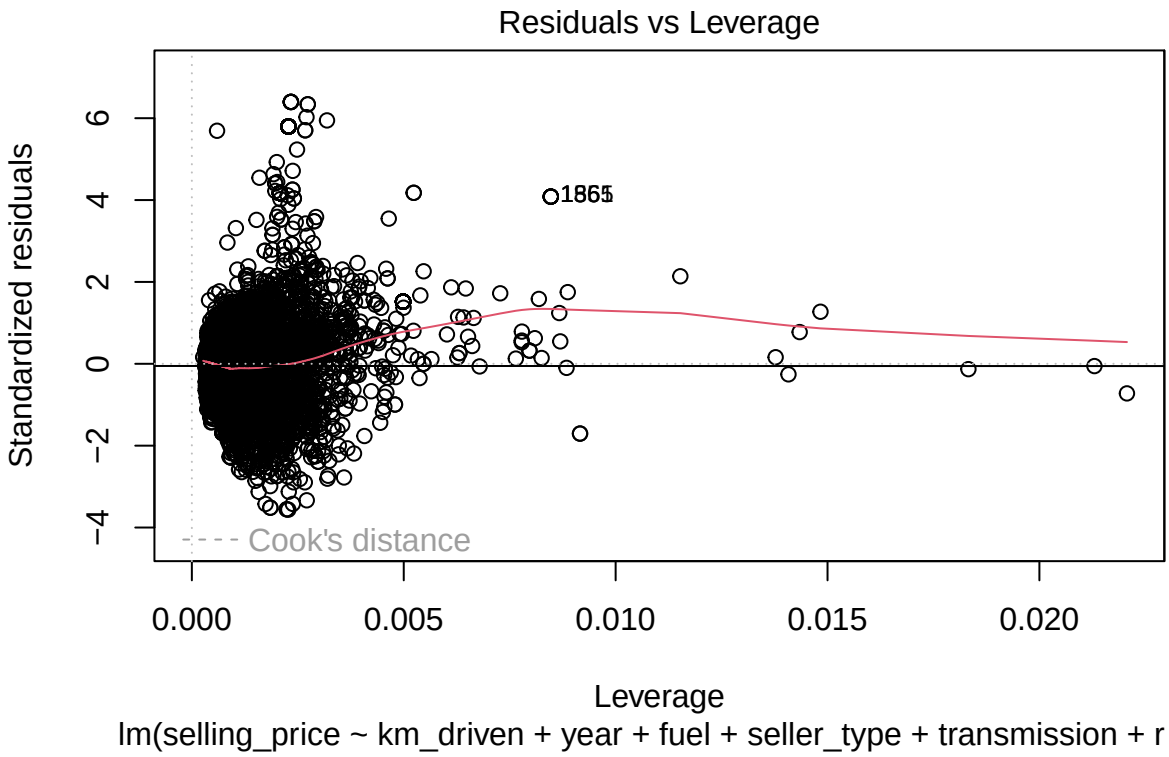



Normal Q−Q
Standardized residuals
−2
0
2
4
6
−4
−2
0
2
4
Theoretical Quantiles
lm(selling_price ~ km_driven + year + fuel + seller_type + transmission + r ...




```
abline(ML_model2)
```

```
## Warning in abline(ML_model2): only using the first two of 10 regression
## coefficients
```




```
## CONCLUSION: Compared the linear regression with multiple linear regression model.
## The Adjusted R-squared:  0.7238  values for the multiple linear regression model is better
## than Multiple R-squared 0.05087 of the linear model. This shows Multiple regression model
## works better for this dataset
```

Dataset - Selected Vehicle data

1.Medical insurance (https://www.kaggle.com/datasets/mirichoi0218/insurance)

Does the cost of the insurance is related with age, habits (smoking vs non smoking), BMI , region etc. Predict the cost increase / decrease on the cost of the insurance?

2. Death rate from Cancer (https://data.world/nrippner/cancer-trials)

Who is prevailing the trails? Is it been driven by the environment conditions or wealth

3. Vehicle data (https://www.kaggle.com/datasets/nehalbirla/vehicle-dataset-from-cardekho)

Is the price of the car increase or decrease as the mileage goes up or because of accident etc? Any other factor.

References