Assignment 4: Predicting Automobile Pricing Using Neural Networks

Nabeel Khan, Nha Nguyen, Razzaq

2022-12-02

R Markdown

This is an R Markdown document. Markdown is a simple formatting syntax for authoring HTML, PDF, and MS Word documents. For more details on using R Markdown see http://rmarkdown.rstudio.com.

When you click the **Knit** button a document will be generated that includes both content as well as the output of any embedded R code chunks within the document. You can embed an R code chunk like this:

```
library(readxl)
library(ISLR)
library(caret)
```

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

```
## Loading required package: lattice
```

```
library(pROC)
```

```
## Type 'citation("pROC")' for a citation.
```

```
##
## Attaching package: 'pROC'
```

```
## The following objects are masked from 'package:stats':
##
##     cov, smooth, var
```

```
library(ROSE)
```

```
## Loaded ROSE 0.0-4
```

```
library(robustbase)
library(smotefamily)
library(ROCR)
library(rpart)
library(rpart.plot)
library(ggplot2)
library(neuralnet)
```

```
##
## Attaching package: 'neuralnet'
```

```
## The following object is masked from 'package:ROCR':
##
##     prediction
```

```
library(GGally)
```

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

```
library(corrr)
library(tidyverse)
```

```
## -- Attaching packages --------------------------------------- tidyverse 1.3.2 --
```

```
## v tibble  3.1.8      v dplyr   1.0.10
## v tidyr   1.2.1      v stringr 1.4.1
## v readr   2.1.2      v forcats 0.5.2
## v purrr   0.3.4
## -- Conflicts ------------------------------------------ tidyverse_conflicts() --
## x dplyr::compute() masks neuralnet::compute()
## x dplyr::filter()  masks stats::filter()
## x dplyr::lag()     masks stats::lag()
## x purrr::lift()    masks caret::lift()
```

```
library(ROSE)

#importing the dataset
Assi4 <- read_excel("Assi4.xlsx")
#Summary of the data present in Assi4
summary(Assi4)
```

```
##      Price            Age              KM              HP
##  Min.   :13000   Min.   :23.00   Min.   : 9752   Min.   :100.0
##  1st Qu.:15800   1st Qu.:26.00   1st Qu.:25448   1st Qu.:100.0
##  Median :16300   Median :28.00   Median :32222   Median :113.0
##  Mean   :17251   Mean   :27.94   Mean   :33628   Mean   :126.9
##  3rd Qu.:18000   3rd Qu.:30.00   3rd Qu.:42210   3rd Qu.:113.0
##  Max.   :22800   Max.   :33.00   Max.   :67662   Max.   :195.0
##        MC            Colour               Auto              CC
##  Min.   :0.0000   Length:31          Min.   :0.00000   Min.   :1400
##  1st Qu.:0.0000   Class :character   1st Qu.:0.00000   1st Qu.:1400
##  Median :1.0000   Mode  :character   Median :0.00000   Median :1600
##  Mean   :0.6452                      Mean   :0.03226   Mean   :1574
##  3rd Qu.:1.0000                      3rd Qu.:0.00000   3rd Qu.:1600
##  Max.   :1.0000                      Max.   :1.00000   Max.   :1800
##       Grs             Wght           G_P             Mfr_G
##  Min.   :5.000   Min.   :1069   Min.   : 4.000   Min.   :0.0000
```

```
##  1st Qu.:5.000   1st Qu.:1104   1st Qu.: 4.000   1st Qu.:1.0000
##  Median :5.000   Median :1124   Median : 4.000   Median :1.0000
##  Mean   :5.194   Mean   :1130   Mean   : 4.903   Mean   :0.9032
##  3rd Qu.:5.000   3rd Qu.:1149   3rd Qu.: 4.000   3rd Qu.:1.0000
##  Max.   :6.000   Max.   :1189   Max.   :20.000   Max.   :1.0000
##      Abag_2             AC              Comp              CD
##  Min.   :0.0000   Min.   :0.0000   Min.   :0.0000   Min.   :0.0000
##  1st Qu.:1.0000   1st Qu.:0.0000   1st Qu.:1.0000   1st Qu.:0.0000
##  Median :1.0000   Median :1.0000   Median :1.0000   Median :1.0000
##  Mean   :0.9355   Mean   :0.5484   Mean   :0.9032   Mean   :0.6129
##  3rd Qu.:1.0000   3rd Qu.:1.0000   3rd Qu.:1.0000   3rd Qu.:1.0000
##  Max.   :1.0000   Max.   :1.0000   Max.   :1.0000   Max.   :1.0000
##      Clock              Pw             Radio              SpM
##  Min.   :0.0000   Min.   :0.0000   Min.   :0.00000   Min.   :0.0000
##  1st Qu.:1.0000   1st Qu.:1.0000   1st Qu.:0.00000   1st Qu.:1.0000
##  Median :1.0000   Median :1.0000   Median :0.00000   Median :1.0000
##  Mean   :0.9032   Mean   :0.9032   Mean   :0.06452   Mean   :0.8065
##  3rd Qu.:1.0000   3rd Qu.:1.0000   3rd Qu.:0.00000   3rd Qu.:1.0000
##  Max.   :1.0000   Max.   :1.0000   Max.   :1.00000   Max.   :1.0000
##      M_Rim           Tow_Bar
##  Min.   :0.0000   Min.   :0.0000
##  1st Qu.:0.0000   1st Qu.:0.0000
##  Median :0.0000   Median :0.0000
##  Mean   :0.3871   Mean   :0.1613
##  3rd Qu.:1.0000   3rd Qu.:0.0000
##  Max.   :1.0000   Max.   :1.0000
```

```
# Check missing values
sapply(Assi4,function(x) sum(is.na(x))) #There are no missing values in all columns.
```

```
##   Price     Age      KM      HP      MC  Colour    Auto      CC     Grs    Wght
##       0       0       0       0       0       0       0       0       0       0
##     G_P   Mfr_G  Abag_2      AC    Comp      CD   Clock      Pw   Radio     SpM
##       0       0       0       0       0       0       0       0       0       0
##   M_Rim Tow_Bar
##       0       0
```

```
# Factor categorical variables
Assi4$Colour <- as.factor(Assi4$Colour)
#Attach Assi4 to data1
data1 <- Assi4
# Calculate correlation coefficient. #Factors that influence a customer's decision
#to buy a car are Horsepower (HP), Cylinder voloume in cubic cms (CC) and Weight (Wght)
#whose correlation with car price are 0.923, 0.890 and 0.856 accordingly.

correlate(data1)
```

```
## Non-numeric variables removed from input: ‘Colour‘
## Correlation computed with
## * Method: ’pearson’
## * Missing treated using: ’pairwise.complete.obs’

## # A tibble: 21 x 22
```

```
##    term    Price     Age      KM       HP      MC    Auto      CC     Grs
##    <chr>   <dbl>   <dbl>   <dbl>    <dbl>   <dbl>   <dbl>   <dbl>   <dbl>
##  1 Price NA       0.273  -0.0733  0.923   -0.117  -0.0183  0.890   0.760
##  2 Age    0.273  NA       0.0356  0.164    0.347   0.127   0.285   0.0381
##  3 KM    -0.0733  0.0356 NA      -0.0601  -0.0701  0.137   0.0738 -0.111
##  4 HP     0.923   0.164  -0.0601 NA       -0.223  -0.0682  0.892   0.896
##  5 MC    -0.117   0.347  -0.0701 -0.223   NA      -0.246  -0.127  -0.319
##  6 Auto  -0.0183  0.127   0.137  -0.0682  -0.246  NA       0.0314 -0.0894
##  7 CC     0.890   0.285   0.0738  0.892   -0.127   0.0314 NA       0.737
##  8 Grs    0.760   0.0381 -0.111   0.896   -0.319  -0.0894  0.737  NA
##  9 Wght   0.856   0.248  -0.0122  0.918   -0.208   0.226   0.846   0.815
## 10 G_P    0.0369 -0.160  -0.261   0.00685  0.0849 -0.0528  0.0497  0.0152
## # ... with 11 more rows, and 13 more variables: Wght <dbl>, G_P <dbl>,
## #   Mfr_G <dbl>, Abag_2 <dbl>, AC <dbl>, Comp <dbl>, CD <dbl>, Clock <dbl>,
## #   Pw <dbl>, Radio <dbl>, SpM <dbl>, M_Rim <dbl>, Tow_Bar <dbl>
```

```
# Test relationship between categorical and numerical variables using Chi-squared test
chisq.test(data1$Price, data1$Colour)
```

```
## Warning in chisq.test(data1$Price, data1$Colour): Chi-squared approximation may
## be incorrect
```

```
##
##  Pearson's Chi-squared test
##
## data:  data1$Price and data1$Colour
## X-squared = 99.338, df = 90, p-value = 0.235
```

(1) After your EDA, what factors do you think influence a customer's decision to buy a care? What are the objectives of the model that Farid plans to build?

Question 1 Answer

There are some variables present in the excel sheet where the variables take the same value for all the observations. These variables can directly be removed from the excel sheet as they do not make any difference to our models as they have the same value for all observations. These varaibles are - Fuel: (Fuel type (Petrol, Diesel, CNG)), Drs(Number of doors), Cyl(Number of cylinders), ABS(Anti-lock brake system (Yes=1, No=0)), Abag1(Driver airbag (Yes=1, No=0)), PStr(Power steering (Yes=1, No=0)). Remaining variables are the ones that influence a customer's decision to buy a car but it can be noted that some variables are more correlated to variable 'Price' than other variables. We have shown this in our code. Although some variables are more correlate than other to price, we have decided to use all the variables except for the ones removed in the first paragraph

We have then performed normalization to the data after importing it bringing it to a scale between 0 and 1. This makes it easiier for neural network model.

Objectives of the model that farid plans to build: For predicting the prices, Farid decided that he would try various machine-learning methods, including linear regression. He also knew that neural networks excelled in predicting price, so he decided to use a feed-forward neural network to train data and accurately predict the price. He wondered how neural networks would compare to linear regression. For long-term marketing, he wanted to decide on one particular computing system to determine prices.

```r
# (2) Construct a neural network model. Validate and interpret the model
# using a different number of hidden neurons.

#Create a function to normalize data between 0 and 1
mynormalization <- function(x)
{
  (x - min(x))/ (max(x)-min(x))
}
#Apply normalization to only those coloumns that are numerical
data1 <- Assi4%>% mutate_if(is.numeric, mynormalization)
summary(data1)
```

```
##      Price              Age               KM               HP
##  Min.   :0.0000   Min.   :0.0000   Min.   :0.0000   Min.   :0.0000
##  1st Qu.:0.2857   1st Qu.:0.3000   1st Qu.:0.2710   1st Qu.:0.0000
##  Median :0.3367   Median :0.5000   Median :0.3880   Median :0.1368
##  Mean   :0.4338   Mean   :0.4935   Mean   :0.4123   Mean   :0.2832
##  3rd Qu.:0.5102   3rd Qu.:0.7000   3rd Qu.:0.5605   3rd Qu.:0.1368
##  Max.   :1.0000   Max.   :1.0000   Max.   :1.0000   Max.   :1.0000
##        MC            Colour        Auto               CC
##  Min.   :0.0000   Black : 5   Min.   :0.00000   Min.   :0.0000
##  1st Qu.:0.0000   Blue  : 9   1st Qu.:0.00000   1st Qu.:0.0000
##  Median :1.0000   Green : 1   Median :0.00000   Median :0.5000
##  Mean   :0.6452   Grey  :12   Mean   :0.03226   Mean   :0.4355
##  3rd Qu.:1.0000   Red   : 2   3rd Qu.:0.00000   3rd Qu.:0.5000
##  Max.   :1.0000   Silver: 2   Max.   :1.00000   Max.   :1.0000
##       Grs              Wght              G_P              Mfr_G
##  Min.   :0.0000   Min.   :0.0000   Min.   :0.00000   Min.   :0.0000
##  1st Qu.:0.0000   1st Qu.:0.2917   1st Qu.:0.00000   1st Qu.:1.0000
##  Median :0.0000   Median :0.4583   Median :0.00000   Median :1.0000
##  Mean   :0.1935   Mean   :0.5081   Mean   :0.05645   Mean   :0.9032
##  3rd Qu.:0.0000   3rd Qu.:0.6667   3rd Qu.:0.00000   3rd Qu.:1.0000
##  Max.   :1.0000   Max.   :1.0000   Max.   :1.00000   Max.   :1.0000
##      Abag_2             AC               Comp              CD
##  Min.   :0.0000   Min.   :0.0000   Min.   :0.0000   Min.   :0.0000
##  1st Qu.:1.0000   1st Qu.:0.0000   1st Qu.:1.0000   1st Qu.:0.0000
##  Median :1.0000   Median :1.0000   Median :1.0000   Median :1.0000
##  Mean   :0.9355   Mean   :0.5484   Mean   :0.9032   Mean   :0.6129
##  3rd Qu.:1.0000   3rd Qu.:1.0000   3rd Qu.:1.0000   3rd Qu.:1.0000
##  Max.   :1.0000   Max.   :1.0000   Max.   :1.0000   Max.   :1.0000
##      Clock              Pw              Radio              SpM
##  Min.   :0.0000   Min.   :0.0000   Min.   :0.00000   Min.   :0.0000
##  1st Qu.:1.0000   1st Qu.:1.0000   1st Qu.:0.00000   1st Qu.:1.0000
##  Median :1.0000   Median :1.0000   Median :0.00000   Median :1.0000
##  Mean   :0.9032   Mean   :0.9032   Mean   :0.06452   Mean   :0.8065
##  3rd Qu.:1.0000   3rd Qu.:1.0000   3rd Qu.:0.00000   3rd Qu.:1.0000
##  Max.   :1.0000   Max.   :1.0000   Max.   :1.00000   Max.   :1.0000
##      M_Rim          Tow_Bar
##  Min.   :0.0000   Min.   :0.0000
##  1st Qu.:0.0000   1st Qu.:0.0000
##  Median :0.0000   Median :0.0000
##  Mean   :0.3871   Mean   :0.1613
##  3rd Qu.:1.0000   3rd Qu.:0.0000
```

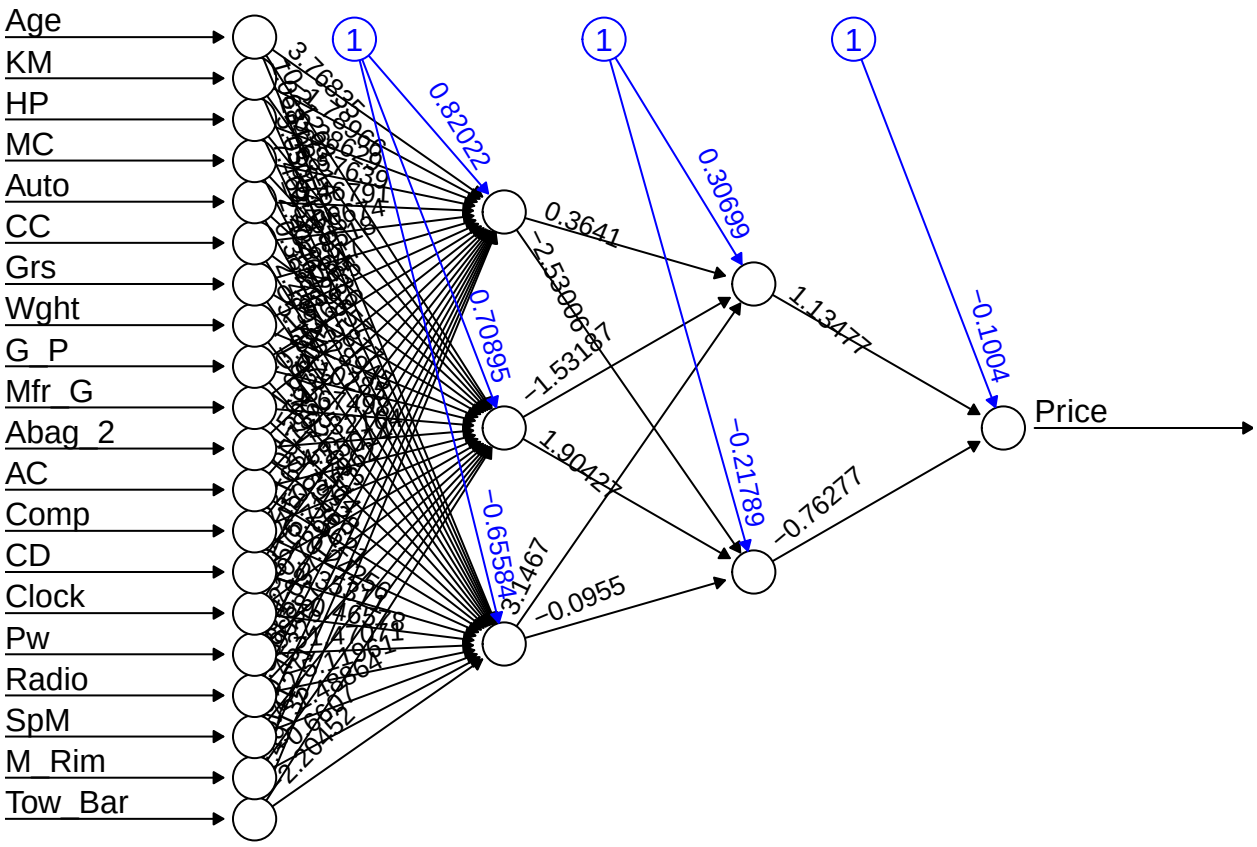
```
##  Max.   :1.0000   Max.   :1.0000
```

```
#Create training and test data
set.seed(124)
indx <- sample(2, nrow(data1), replace = T, prob = c(0.7, 0.3))
train <- data1[indx == 1, ]
test <- data1[indx == 2, ]

#Create Neural Network Model
set.seed(12321)
nnModel1 <- neuralnet(Price ~ Age + KM + HP + MC + Auto + CC + Grs + Wght + G_P + Mfr_G + Abag_2 +
                        AC + Comp + CD + Clock + Pw + Radio + SpM + M_Rim + Tow_Bar, data = train,
                      err.fct = 'sse', hidden = c(3,2), threshold = 0.01, linear.output = T)


#Check for error and other parameters using result.matrix
nnModel1$result.matrix
```

```
##                                [,1]
## error                   0.011562574
## reached.threshold       0.004938142
## steps                 111.000000000
## Intercept.to.1layhid1   0.820224401
## Age.to.1layhid1         3.768349990
## KM.to.1layhid1         -1.789664164
## HP.to.1layhid1          0.386294119
## MC.to.1layhid1          0.576393249
## Auto.to.1layhid1        0.467906031
## CC.to.1layhid1          1.066735548
## Grs.to.1layhid1        -0.851800712
## Wght.to.1layhid1        0.803473099
## G_P.to.1layhid1        -1.882545614
## Mfr_G.to.1layhid1      -0.864612567
## Abag_2.to.1layhid1     -1.644053347
## AC.to.1layhid1         -1.569773474
## Comp.to.1layhid1       -1.007856615
## CD.to.1layhid1          0.231672799
## Clock.to.1layhid1       1.042445234
## Pw.to.1layhid1         -1.585747574
## Radio.to.1layhid1       1.002211915
## SpM.to.1layhid1        -0.388048569
## M_Rim.to.1layhid1       1.705002655
## Tow_Bar.to.1layhid1    -0.371200763
## Intercept.to.1layhid2   0.708950973
## Age.to.1layhid2        -0.212681901
## KM.to.1layhid2         -3.668135300
## HP.to.1layhid2         -3.979322733
## MC.to.1layhid2         -0.450550977
## Auto.to.1layhid2       -0.666357889
## CC.to.1layhid2         -3.296567317
## Grs.to.1layhid2        -5.158770432
## Wght.to.1layhid2       -1.289448615
## G_P.to.1layhid2         0.102845964
## Mfr_G.to.1layhid2       1.674982481
```

```
## Abag_2.to.1layhid2     -1.321643395
## AC.to.1layhid2         -4.150138038
## Comp.to.1layhid2       -0.379616557
## CD.to.1layhid2         -6.717130530
## Clock.to.1layhid2      -0.729597629
## Pw.to.1layhid2         -1.037170271
## Radio.to.1layhid2       3.968872090
## SpM.to.1layhid2        -0.453253314
## M_Rim.to.1layhid2      -6.412707410
## Tow_Bar.to.1layhid2    -2.491349543
## Intercept.to.1layhid3  -0.655843480
## Age.to.1layhid3         1.068465592
## KM.to.1layhid3         -5.050540248
## HP.to.1layhid3          7.462586006
## MC.to.1layhid3         -2.434637303
## Auto.to.1layhid3        0.387411557
## CC.to.1layhid3          2.489236595
## Grs.to.1layhid3         5.192727882
## Wght.to.1layhid3        1.825247555
## G_P.to.1layhid3        -7.026036540
## Mfr_G.to.1layhid3      -1.515640751
## Abag_2.to.1layhid3      0.693283328
## AC.to.1layhid3          1.256206324
## Comp.to.1layhid3       -0.242563934
## CD.to.1layhid3          0.353113843
## Clock.to.1layhid3      -0.465777167
## Pw.to.1layhid3         -1.470714639
## Radio.to.1layhid3      -5.119614668
## SpM.to.1layhid3        -1.488639744
## M_Rim.to.1layhid3      -0.660696697
## Tow_Bar.to.1layhid3    -2.204517686
## Intercept.to.2layhid1   0.306985563
## 1layhid1.to.2layhid1    0.364099760
## 1layhid2.to.2layhid1   -1.531865567
## 1layhid3.to.2layhid1    3.146702685
## Intercept.to.2layhid2  -0.217887538
## 1layhid1.to.2layhid2   -2.530063358
## 1layhid2.to.2layhid2    1.904271813
## 1layhid3.to.2layhid2   -0.095498817
## Intercept.to.Price     -0.100395726
## 2layhid1.to.Price       1.134773107
## 2layhid2.to.Price      -0.762774759
```

```
#Error on training set for this model is 0.011562574
#Plot the model
plot(nnModel1, rep = 'best')
```




```
# Predict on test data
pr.nn <- neuralnet::compute(nnModel1, test[,1:22])

# Compute mean squared error
pr.nn_ <- pr.nn$net.result * (max(data1$Price) - min(data1$Price)) + min(data1$Price)
pr.nn_
```

```
##            [,1]
##  [1,] 0.6309953
##  [2,] 0.3488842
##  [3,] 0.4015929
##  [4,] 0.3336059
##  [5,] 0.3020155
##  [6,] 0.4583227
##  [7,] 0.2756727
##  [8,] 0.2611084
##  [9,] 0.5216279
## [10,] 0.1725715
```

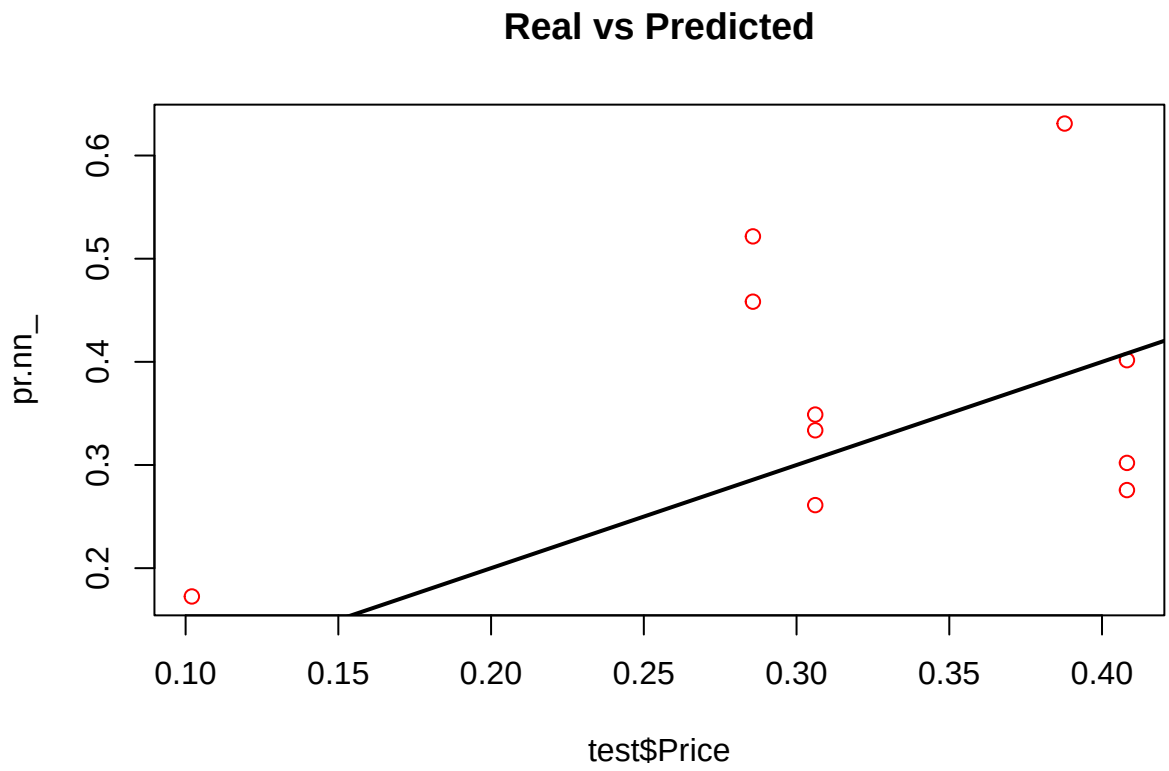
```
test.r <- (test$Price) * (max(data1$Price) - min(data1$Price)) +
  min(data1$Price)
test.r
```

```
##  [1] 0.3877551 0.3061224 0.4081633 0.3061224 0.4081633 0.2857143 0.4081633
##  [8] 0.3061224 0.2857143 0.1020408
```

```
#MSE.nn is the Mean Squared Error on the test data
MSE.nn <- sum((test.r - pr.nn_)^2) / nrow(test)
MSE.nn
```

```
## [1] 0.01830637
```

```
#MSE.nn is the mean squared error for test data which is 0.01830637

# Plot regression line
plot(test$Price, pr.nn_, col = "red",
     main = 'Real vs Predicted')
abline(0, 1, lwd = 2)
```



I have used various hidden neurons like (5,4,2), (4,2), (2,1) and (3,2) and have finally decided to use (3,2) after comparing the error for each on training and test data. For the Real vs pred graph, we can see that the predictions (red circles) made by the neural network are in general concentrated around the line (a perfect alignment with the line would indicate an MSE of 0 and thus an ideal prediction).

```
# (3) Compare your neural network models with linear regression model. Which one is better?

# Converting "Price" as numeric and factorize "Colour" variable
data1$Price <- as.numeric(data1$Price)
data1$Colour <- as.factor(data1$Colour)

# Split train test and test set for lm regression
set.seed(123)
```

```r
lm.indx <- sample(2,nrow(data1), replace = T, prob = c(0.7,0.3))
lm.train <- data1[lm.indx == 1,]
lm.test <- data1[lm.indx ==2,]

full <- lm(lm.train$Price ~ .,data = lm.train)
null <- lm(lm.train$Price ~ 1,data = lm.train)

step(null, scope = list(lower = null, upper = full), direction = "forward", trace = 0)
```

```
##
## Call:
## lm(formula = lm.train$Price ~ HP + Colour + SpM + Mfr_G + Clock,
##     data = lm.train)
##
## Coefficients:
##  (Intercept)            HP    ColourBlue   ColourGreen    ColourGrey
##     0.209212      0.604399     -0.005046      0.021214      0.043884
##    ColourRed  ColourSilver           SpM         Mfr_G         Clock
##    -0.190328     -0.145074     -0.092501      0.072009      0.075781
```

```r
#Create LM Model
#I have taken -Tow_Bar - Radio -Pw -Comp -Abag_2 -G_P -Colour variables out as when
#we split it into train and test data, that is because the datasets were imbalanced
#and linear regression failed. You might think this is wrong to remove variables like
#this but I can justify it as they did not have a high correlation with 'price' when
#we ended up doing EDA in question 1. Hence it is justified.

lmModel <- lm(formula = lm.train$Price ~. -Tow_Bar - Radio -Pw -Comp -Abag_2
              -G_P -Colour, data = lm.train)
summary(lmModel)
```

```
##
## Call:
## lm(formula = lm.train$Price ~ . - Tow_Bar - Radio - Pw - Comp -
##     Abag_2 - G_P - Colour, data = lm.train)
##
## Residuals:
##          1          2          3          4          5          6          7
## -1.681e-02 -8.041e-02  9.722e-02  2.373e-17  2.503e-17 -4.913e-17 -5.329e-18
##          8          9         10         11         12         13         14
## -6.306e-03  7.687e-02 -2.186e-02  1.681e-02 -7.564e-02  1.013e-02  6.514e-02
##         15         16         17         18         19         20
##  4.019e-02 -5.347e-17 -1.013e-02 -6.712e-02 -2.808e-02 -4.783e-17
##
## Coefficients:
##             Estimate Std. Error t value Pr(>|t|)
## (Intercept)  0.24447    0.13832   1.767    0.137
## Age          0.11052    0.12283   0.900    0.409
## KM          -0.14686    0.21309  -0.689    0.521
## HP           0.14364    0.58200   0.247    0.815
## MC          -0.07898    0.08579  -0.921    0.400
## Auto        -0.16712    0.24398  -0.685    0.524
```

```
## CC           0.35847    0.48523   0.739    0.493
## Grs         -0.19703    0.15351  -1.284    0.256
## Wght         0.39744    0.45526   0.873    0.423
## Mfr_G       -0.09667    0.14832  -0.652    0.543
## AC          -0.18595    0.19272  -0.965    0.379
## CD          -0.01800    0.06482  -0.278    0.792
## Clock        0.16634    0.12167   1.367    0.230
## SpM         -0.09030    0.08445  -1.069    0.334
## M_Rim        0.04233    0.09388   0.451    0.671
##
## Residual standard error: 0.08943 on 5 degrees of freedom
## Multiple R-squared:  0.9616, Adjusted R-squared:  0.8541
## F-statistic: 8.948 on 14 and 5 DF,  p-value: 0.01219
```

```
# LR prediction on train data and test data
train_MSE = mean(lmModel$residuals^2)
test_MSE = mean((lm.test$Price - predict.lm(lmModel, lm.test)) ^ 2)
train_MSE #0.001999398
```

```
## [1] 0.001999398
```

```
test_MSE  #0.01692538
```

```
## [1] 0.01692538
```

```
fitted(lmModel)
```

```
##         1         2         3         4         5         6         7         8
## 0.8331322 0.7589858 0.8262493 1.0000000 0.3877551 0.4081633 0.4081633 0.3124286
##         9        10        11        12        13        14        15        16
## 0.3312929 0.3585925 0.4469698 0.3613524 0.3980372 0.2205757 0.2659326 0.2602041
##        17        18        19        20
## 0.3162485 0.2712032 0.3137951 0.1836735
```

```
coefficients(lmModel)
```

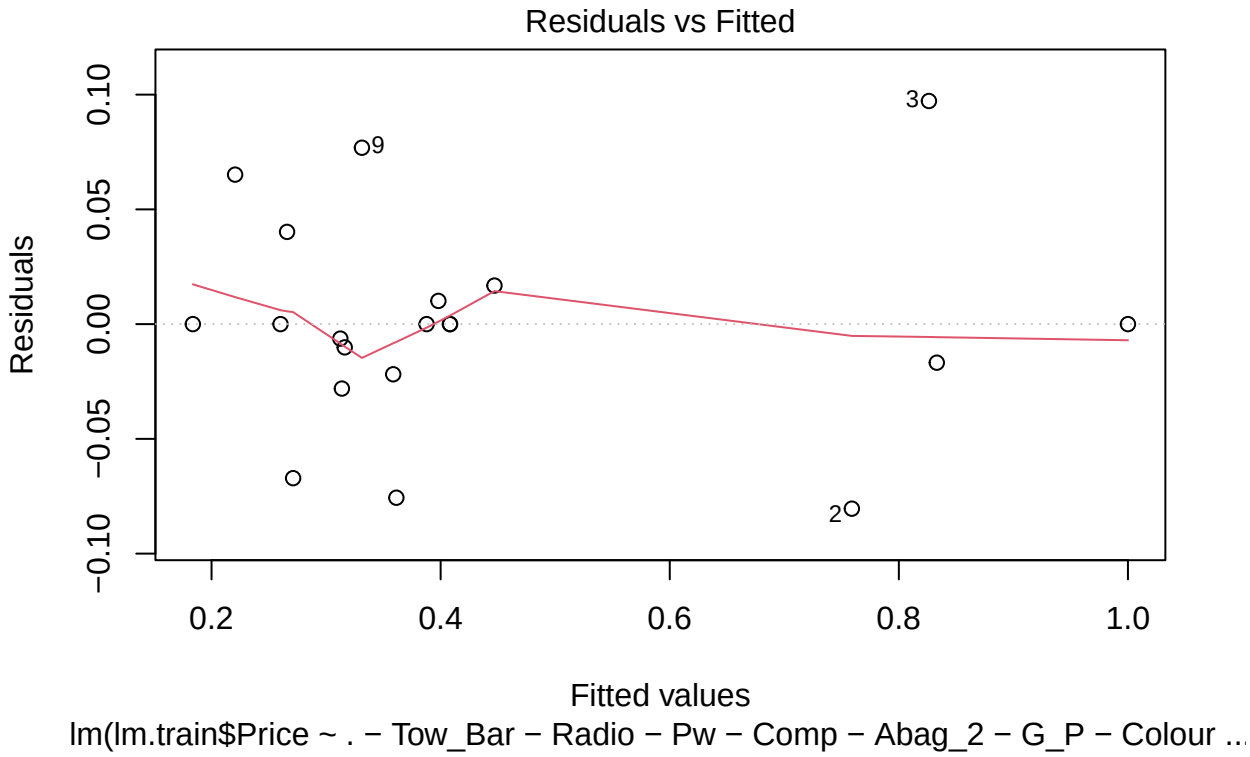
```
## (Intercept)         Age          KM          HP          MC        Auto
##  0.24447486  0.11051556 -0.14685827  0.14363655 -0.07897997 -0.16712447
##          CC         Grs        Wght       Mfr_G          AC          CD
##  0.35847007 -0.19702882  0.39743509 -0.09666580 -0.18595028 -0.01799818
##       Clock         SpM       M_Rim
##  0.16633640 -0.09029839  0.04232620
```

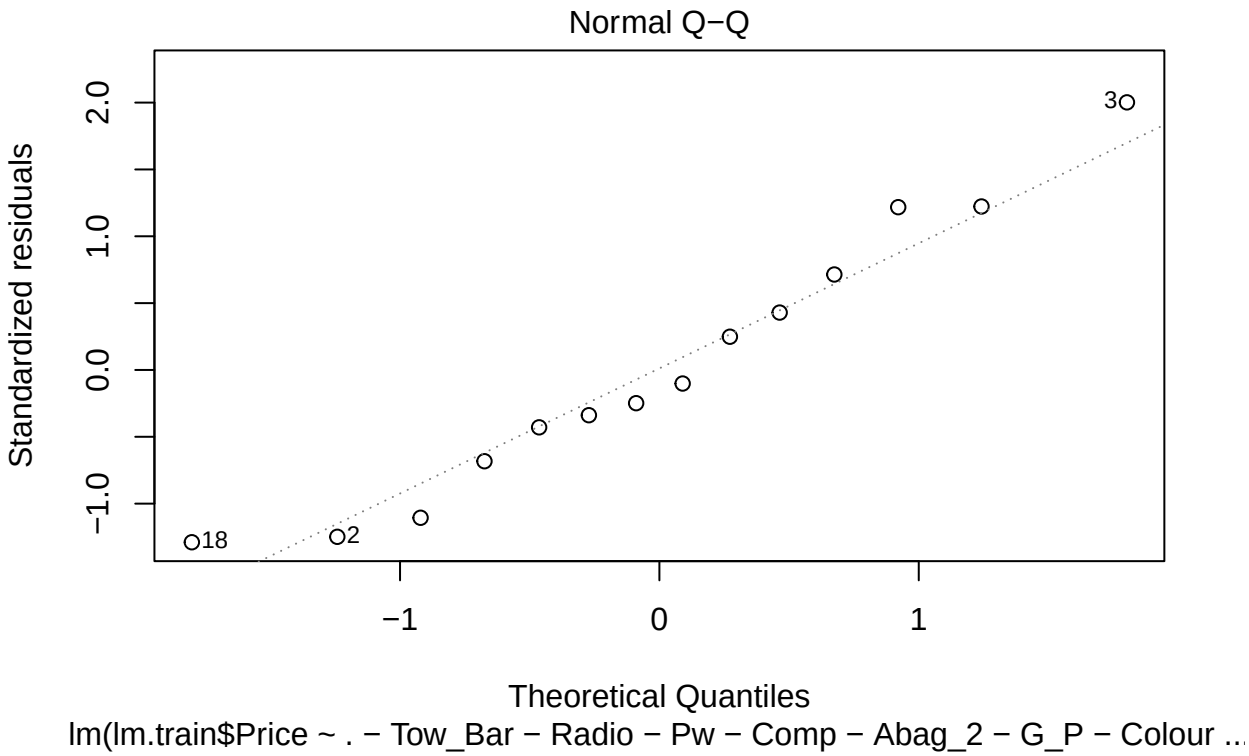
```
residuals((lmModel))
```

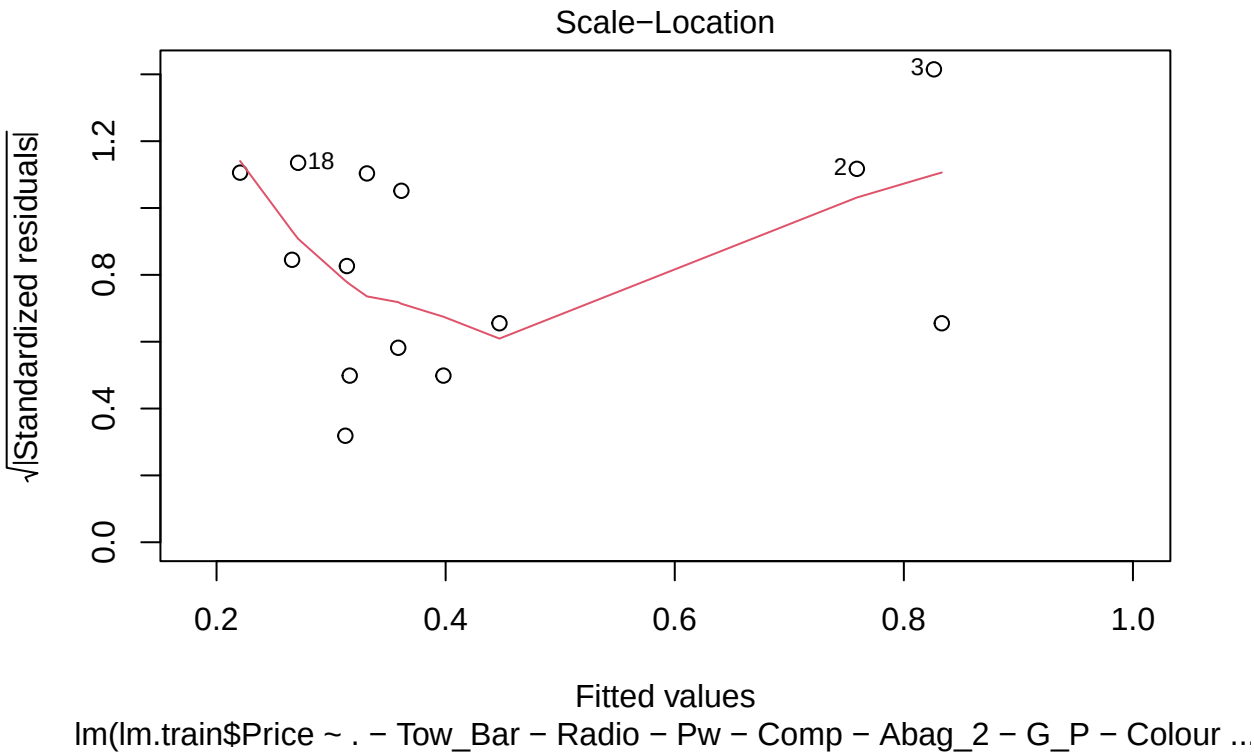
```
##             1             2             3             4             5
## -1.680568e-02 -8.041438e-02  9.722006e-02  2.372802e-17  2.502907e-17
##             6             7             8             9            10
## -4.913036e-17 -5.328593e-18 -6.306164e-03  7.687033e-02 -2.185778e-02
##            11            12            13            14            15
##  1.680568e-02 -7.563812e-02  1.012606e-02  6.513861e-02  4.018986e-02
##            16            17            18            19            20
## -5.346717e-17 -1.012606e-02 -6.712160e-02 -2.808081e-02 -4.782932e-17
```

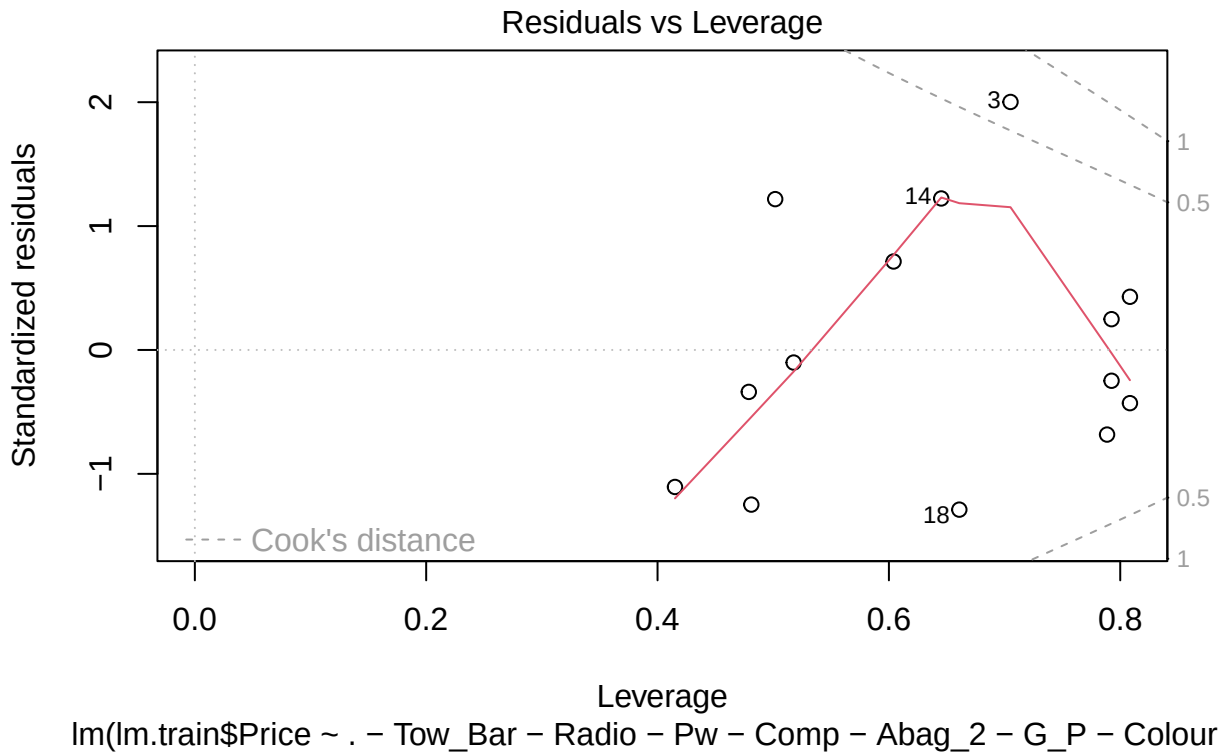
```
plot(lmModel)
```

```
## Warning: not plotting observations with leverage one:
##   4, 5, 6, 7, 16, 20
```




Normal Q−Q
Standardized residuals
2.0
1.0
0.0
−1.0
3
18
2
−1
0
1
Theoretical Quantiles
lm(lm.train$Price ~ . − Tow_Bar − Radio − Pw − Comp − Abag_2 − G_P − Colour ...


Scale−Location
√|Standardized residuals|
0.0
0.4
0.8
1.2
0.2
0.4
0.6
0.8
1.0
3
18
2
Fitted values
lm(lm.train$Price ~ . − Tow_Bar − Radio − Pw − Comp − Abag_2 − G_P − Colour ...



As we can see, I have got error MSE on train and test data of linear regression model in this question and also neural network model in the previous question. It is clearly seen that error is less for linear regression model when compared to neural network model but it is noteworthy to keep in mind that we considered all the variables for our neural network model. Also, the training and test error for neural network model were almost similar whereas for LM model we can clearly see that training set has very less error when compared to test error. This can mean that our LM model can be slightly overfitted. As to which one is better, I would say it depends on the individual deciding as we can prune both models to satisfy our needs with the question but I personally would go for the neural network model as it would lead to more accurate and faster results when we are working with huge and changing datasets.

Question 4 (4)Make a decision and offer your recommendations. Decision: Neural network is a better model Recommendations: The dataset used for this exercise is very small when compared to actual practical datasets for car prices. For practical use, neural network is the better model as it takes into account all the variables and can easily change accordingly if our needs from the model change in the future. I would still like to point out that as we are getting lower error for LM model, it would not be fair to completely neglect that option.

‘‘‘