Assign4-2_Small

R Markdown

```
library(caret)
library(neuralnet)
library(nnet)

toyota.df <- read.csv("/Users/Nick/Desktop/toyota.csv", stringsAsFactors=FALS
E)
toyota.df$Fuel_Diesel <- ifelse(toyota.df$Fuel_Type=="Diesel", 1, 0)
toyota.df$Fuel_Petrol <- ifelse(toyota.df$Fuel_Type=="Petrol", 1, 0)
columns <- c('Powered_Windows', 'Power_Steering', 'ABS', 'KM', 'Radio', 'Cent
ral_Lock', 'Price')

toyota.df <- toyota.df[,(names(toyota.df) %in% columns)]


str(toyota.df)
```

```
## 'data.frame':    1436 obs. of  7 variables:
##  $ Price          : int  13500 13750 13950 14950 13750 12950 16900 18600 2
1500 12950 ...
##  $ KM             : int  46986 72937 41711 48000 38500 61000 94612 75889 1
9700 71138 ...
##  $ ABS            : int  1 1 1 1 1 1 1 1 1 1 ...
##  $ Central_Lock   : int  1 1 0 0 1 1 1 1 1 0 ...
##  $ Powered_Windows: int  1 0 0 0 1 1 1 1 1 0 ...
##  $ Power_Steering : int  1 1 1 1 1 1 1 1 1 1 ...
##  $ Radio          : int  0 0 0 0 0 0 0 0 1 0 ...
```

Normalization of the Data

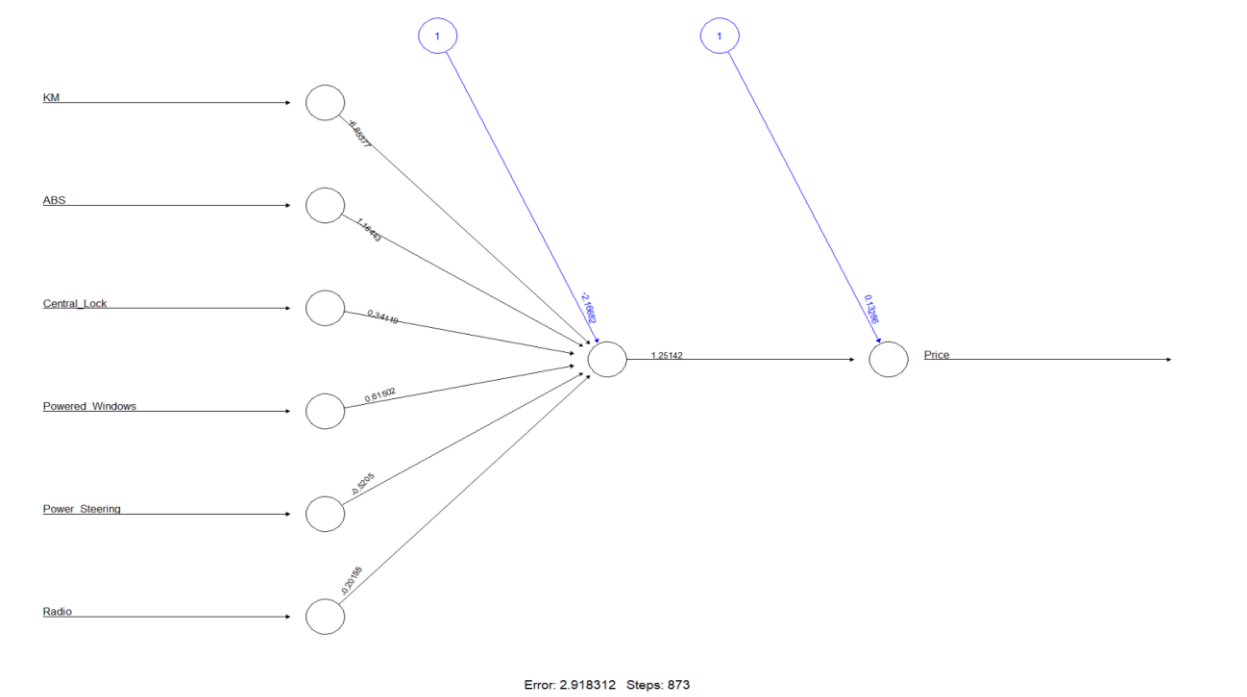
```
normalize <- function(x){return ((x-min(x))/(max(x)-min(x)))}

toyota.df <- as.data.frame(lapply(toyota.df, normalize))
```

Training the Data

```
set.seed(1234)
intrain <- createDataPartition(y=toyota.df$Price, p=0.6, list=FALSE)
train <- toyota.df[intrain,]
test <- toyota.df[-intrain,]
```

Default Neural Net Model

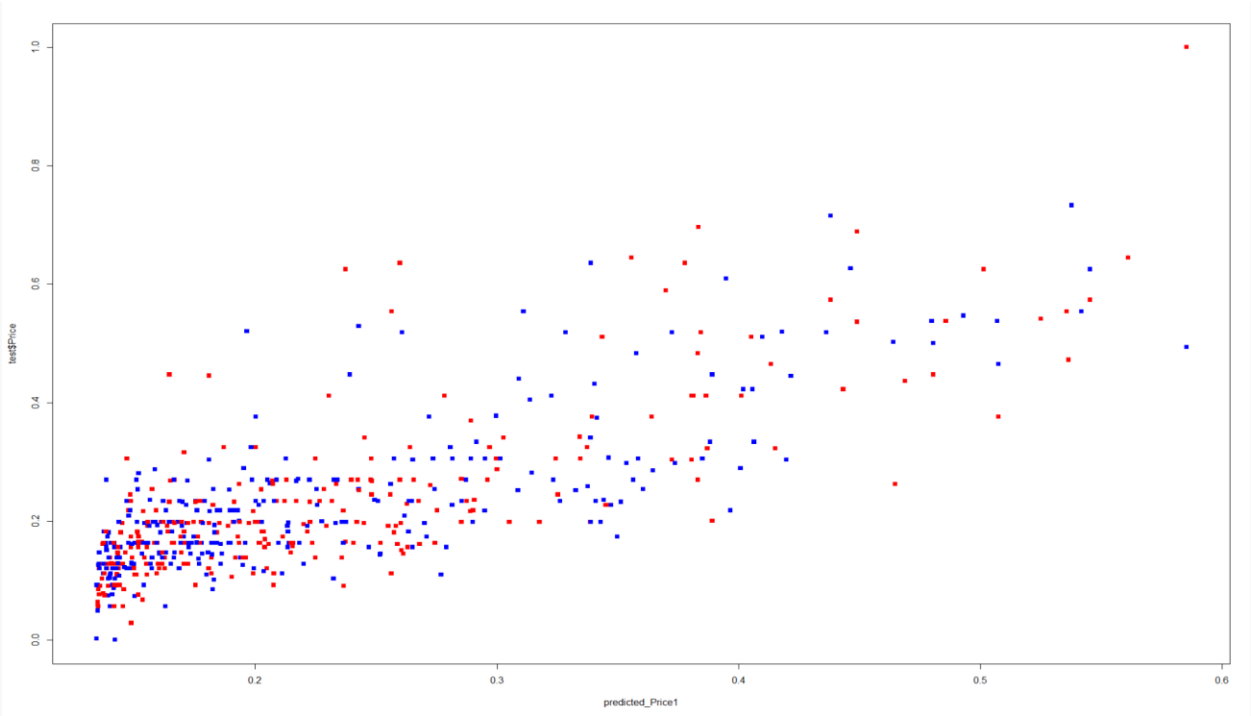
```
model1 <- neuralnet(Price~., data = train)
plot(model1)
model1_Results <- compute(model1, test[2:7])
predicted_Price1 <- model1_Results$net.result
```




```
cor(predicted_Price1, test$Price)
```

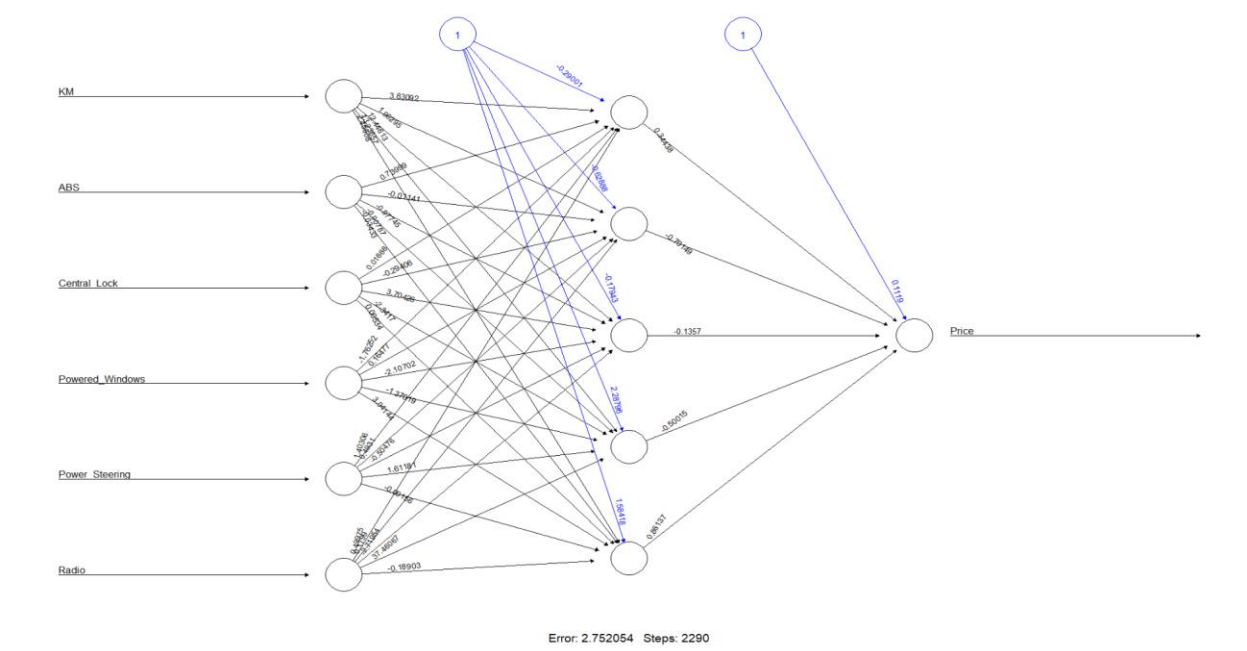
```
##          [,1]
## [1,] 0.781925
```

We observe a relatively good correlation between predicted and test price by 78.19%.


test$Price
predicted_Price1
0.0
0.2
0.4
0.6
0.8
1.0
0.2
0.3
0.4
0.5
0.6

Neural Net model with Hidden 5

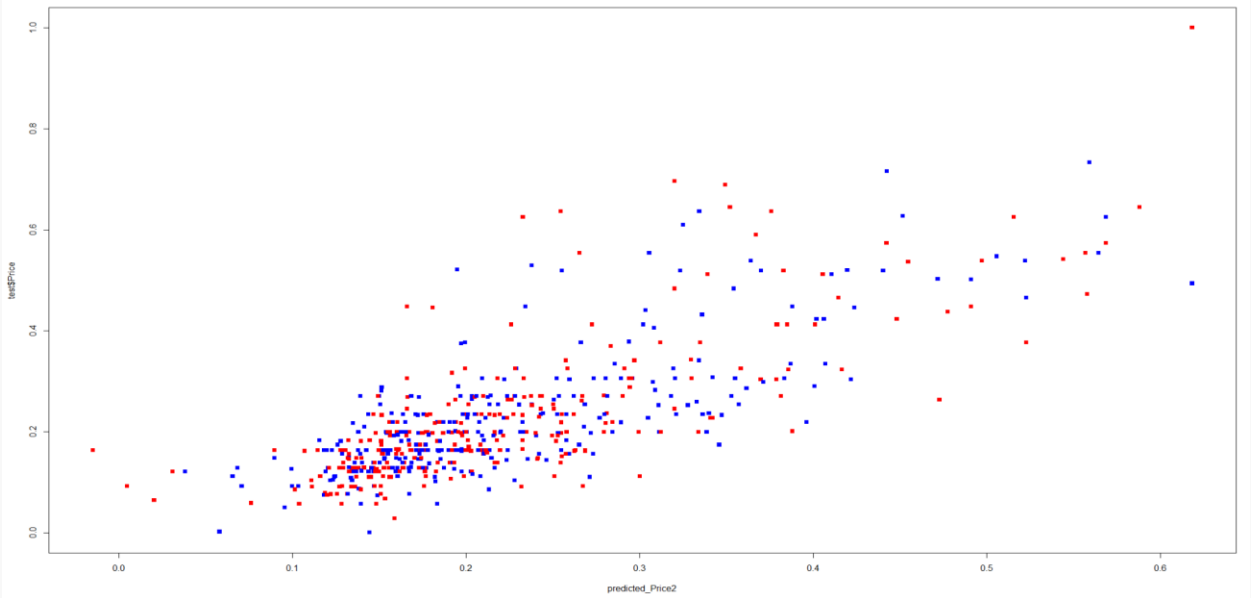
```
model2 <- neuralnet(Price~., data = train, hidden = 5)
plot(model2)
model2_Results <- compute(model2, test[2:7])
predicted_Price2 <- model2_Results$net.result
```




```
cor(predicted_Price2, test$Price)
```

```
##           [,1]
## [1,] 0.7713379
```

We observe a correlation of price with test price of hidden 5 nodes of 77.13% .




```
model4 <- train(Price~., data = train, method = "nnet")

print(model4)

## Neural Network
##
## 863 samples
##   6 predictor
##
## No pre-processing
## Resampling: Bootstrapped (25 reps)
## Summary of sample sizes: 863, 863, 863, 863, 863, 863, ...
## Resampling results across tuning parameters:
##
##   size  decay  RMSE        Rsquared   MAE
##   1     0e+00  0.18956589  0.5644208  0.15951113
##   1     1e-04  0.11282784  0.4847529  0.08639121
##   1     1e-01  0.09228066  0.4975282  0.06650569
##   3     0e+00  0.18211076  0.5832923  0.15267901
##   3     1e-04  0.12621994  0.4614552  0.09953073
##   3     1e-01  0.09217218  0.4952804  0.06652614
##   5     0e+00  0.14770551  0.5712595  0.11998224
##   5     1e-04  0.12568839  0.4752077  0.09887597
##   5     1e-01  0.09221865  0.4944304  0.06657133
##
## RMSE was used to select the optimal model using the smallest value.
## The final values used for the model were size = 3 and decay = 0.1.

model4_predict1 <- predict(model4, test[2:7], type = "raw")
summary(model4_predict1)
```

```
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
## 0.07035 0.16126 0.21533 0.22803 0.29082 0.41023
```

```
cor(model4_predict1, test$Price)
```

```
## [1] 0.7272289
```

We observe using the nnet model that the accuracy is 72.72% when predicted price based on the test set. The model selected size 3 has the best option.

```
model5 <- train(Price~., data = train, method = "nnet")
```

```
print(model5)
```

```
## Neural Network
##
## 863 samples
##   6 predictor
##
## No pre-processing
## Resampling: Bootstrapped (25 reps)
## Summary of sample sizes: 863, 863, 863, 863, 863, 863, ...
## Resampling results across tuning parameters:
##
##   size  decay  RMSE        Rsquared   MAE
##   1     0e+00  0.18953842  0.5771788  0.16040626
##   1     1e-04  0.16191019  0.3726560  0.13402070
##   1     1e-01  0.09134900  0.4981738  0.06667733
##   3     0e+00  0.21789295  0.5705701  0.18713026
##   3     1e-04  0.12066883  0.4467094  0.09461067
##   3     1e-01  0.09120233  0.4970842  0.06669602
##   5     0e+00  0.19742030  0.5507767  0.16711405
##   5     1e-04  0.12685742  0.4446409  0.10014576
##   5     1e-01  0.09128834  0.4957370  0.06678516
##
## RMSE was used to select the optimal model using the smallest value.
## The final values used for the model were size = 3 and decay = 0.1.
```

```
model5_predict1 <- predict(model5, test[2:7], type = "raw")
summary(model5_predict1)
```

```
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
## 0.07035 0.16126 0.21532 0.22803 0.29082 0.41023
```

```
cor(model5_predict1, test$Price)
```

```
## [1] 0.7272269
```

There I no way to actually change the node for the NNET package, but we can observe the size 5 and it is corresponding RMSE. The model specifies that 3 is the best option so there no need to factor in Hidden node 5.

In Conclusion we observe that the Default Neural Net model is the best option because it has the highest accuracy and the fewest number of hidden nodes. Neural Net model with Hidden 5 has a lower error rate when compared to the default Neural network, but the steps has increased tremendously. Which is normal. The remaining two NNET models have a less accuracy compared to the others and thus should not be chosen.