Assignment 6

Nikita Pai & Abhilash Hemaraj

4/18/2020

```
library(readxl)
data <- read_excel('18 Toyota Corolla.xlsx')
str(data)
```

```
## Classes 'tbl_df', 'tbl' and 'data.frame':    1436 obs. of  38 variables:
##  $ Id              : num  1 2 3 4 5 6 7 8 9 10 ...
##  $ Model           : chr  "TOYOTA Corolla 2.0 D4D HATCHB TERRA 2/3-Doors" "TOYOTA Corolla 2.0 D4D HA
##  $ Price           : num  13500 13750 13950 14950 13750 ...
##  $ Age_08_04       : num  23 23 24 26 30 32 27 30 27 23 ...
##  $ Mfg_Month       : num  10 10 9 7 3 1 6 3 6 10 ...
##  $ Mfg_Year        : num  2002 2002 2002 2002 2002 ...
##  $ KM              : num  46986 72937 41711 48000 38500 ...
##  $ Fuel_Type       : chr  "Diesel" "Diesel" "Diesel" "Diesel" ...
##  $ HP              : num  90 90 90 90 90 90 90 90 192 69 ...
##  $ Met_Color       : num  1 1 1 0 0 0 1 1 0 0 ...
##  $ Color           : chr  "Blue" "Silver" "Blue" "Black" ...
##  $ Automatic       : num  0 0 0 0 0 0 0 0 0 0 ...
##  $ cc              : num  2000 2000 2000 2000 2000 2000 2000 2000 1800 1900 ...
##  $ Doors           : num  3 3 3 3 3 3 3 3 3 3 ...
##  $ Cylinders       : num  4 4 4 4 4 4 4 4 4 4 ...
##  $ Gears           : num  5 5 5 5 5 5 5 5 5 5 ...
##  $ Quarterly_Tax   : num  210 210 210 210 210 210 210 210 100 185 ...
##  $ Weight          : num  1165 1165 1165 1165 1170 ...
##  $ Mfr_Guarantee   : num  0 0 1 1 1 0 0 1 0 0 ...
##  $ BOVAG_Guarantee : num  1 1 1 1 1 1 1 1 1 1 ...
##  $ Guarantee_Period: num  3 3 3 3 3 3 3 3 3 3 ...
##  $ ABS             : num  1 1 1 1 1 1 1 1 1 1 ...
##  $ Airbag_1        : num  1 1 1 1 1 1 1 1 1 1 ...
##  $ Airbag_2        : num  1 1 1 1 1 1 1 1 0 1 ...
##  $ Airco           : num  0 1 0 0 1 1 1 1 1 1 ...
##  $ Automatic_airco : num  0 0 0 0 0 0 0 0 0 0 ...
##  $ Boardcomputer   : num  1 1 1 1 1 1 1 1 0 1 ...
##  $ CD_Player       : num  0 1 0 0 0 0 0 1 0 0 ...
##  $ Central_Lock    : num  1 1 0 0 1 1 1 1 1 0 ...
##  $ Powered_Windows : num  1 0 0 0 1 1 1 1 1 0 ...
##  $ Power_Steering  : num  1 1 1 1 1 1 1 1 1 1 ...
##  $ Radio           : num  0 0 0 0 0 0 0 0 1 0 ...
##  $ Mistlamps       : num  0 0 0 0 1 1 0 0 0 0 ...
##  $ Sport_Model     : num  0 0 0 0 0 0 1 0 0 0 ...
##  $ Backseat_Divider: num  1 1 1 1 1 1 1 1 0 1 ...
##  $ Metallic_Rim    : num  0 0 0 0 0 0 0 0 1 0 ...
```

```
##  $ Radio_cassette  : num  0 0 0 0 0 0 0 0 1 0 ...
##  $ Tow_Bar         : num  0 0 0 0 0 0 0 0 0 0 ...
```

```r
#Factor variables which is equivalent to creating dummy variables
data$Fuel_Type <- factor(data$Fuel_Type)
data$Color <- factor(data$Color)
```

```r
data$Price <- (data$Price - min(data$Price))/(max(data$Price) - min(data$Price))
data$Age_08_04 <- (data$Age_08_04 - min(data$Age_08_04))/(max(data$Age_08_04) - min(data$Age_08_04))
data$Mfg_Month <- (data$Mfg_Month - min(data$Mfg_Month))/(max(data$Mfg_Month) - min(data$Mfg_Month))
data$Mfg_Year <- (data$Mfg_Year - min(data$Mfg_Year))/(max(data$Mfg_Year)-min(data$Mfg_Year))
data$Guarantee_Period <- (data$Guarantee_Period - min(data$Guarantee_Period))/(max(data$Guarantee_Perio
data$Weight <- (data$Weight - min(data$Weight))/(max(data$Weight) - min(data$Weight))
data$Quarterly_Tax <- (data$Quarterly_Tax - min(data$Quarterly_Tax))/(max(data$Quarterly_Tax)-min(data$
data$cc <- (data$cc - min(data$cc))/(max(data$cc) - min(data$cc))
data$Doors <- (data$Doors - min(data$Doors))/(max(data$Doors) - min(data$Doors))
data$Cylinders <- (data$Cylinders - min(data$Cylinders))/(max(data$Cylinders)-min(data$Cylinders))
data$Gears <- (data$Gears - min(data$Gears))/(max(data$Gears)-min(data$Gears))
data$KM <- (data$KM - min(data$KM))/(max(data$KM) - min(data$KM))
data$HP <- (data$HP - min(data$HP))/(max(data$HP) - min(data$HP))
```

```r
#creating data set
set.seed(222)
ind <- sample(2, nrow(data), replace = TRUE, prob = c(0.75, 0.25))
training <- data[ind==1,]
testing <- data[ind==2,]
str(training)
```

```
## Classes 'tbl_df', 'tbl' and 'data.frame':    1096 obs. of  38 variables:
##  $ Id              : num  2 3 4 7 8 9 10 11 12 13 ...
##  $ Model           : chr  "TOYOTA Corolla 2.0 D4D HATCHB TERRA 2/3-Doors" " TOYOTA Corolla 2.0 D4D H
##  $ Price           : num  0.334 0.341 0.377 0.446 0.506 ...
##  $ Age_08_04       : num  0.278 0.291 0.316 0.329 0.367 ...
##  $ Mfg_Month       : num  0.818 0.727 0.545 0.455 0.182 ...
##  $ Mfg_Year        : num  0.667 0.667 0.667 0.667 0.667 ...
##  $ KM              : num  0.3 0.172 0.198 0.389 0.312 ...
##  $ Fuel_Type       : Factor w/ 3 levels "CNG","Diesel",..: 2 2 2 2 2 3 2 3 3 3 ...
##  $ HP              : num  0.171 0.171 0.171 0.171 0.171 ...
##  $ Met_Color       : num  1 1 0 1 1 0 0 0 0 0 ...
##  $ Color           : Factor w/ 10 levels "Beige","Black",..: 7 3 2 5 5 6 3 7 6 6 ...
##  $ Automatic       : num  0 0 0 0 0 0 0 0 0 0 ...
##  $ cc              : num  0.0476 0.0476 0.0476 0.0476 0.0476 ...
##  $ Doors           : num  0.333 0.333 0.333 0.333 0.333 ...
##  $ Cylinders       : num  NaN NaN NaN NaN NaN NaN NaN NaN NaN NaN ...
##  $ Gears           : num  0.667 0.667 0.667 0.667 0.667 ...
##  $ Quarterly_Tax   : num  0.723 0.723 0.723 0.723 0.723 ...
##  $ Weight          : num  0.268 0.268 0.268 0.398 0.398 ...
##  $ Mfr_Guarantee   : num  0 1 1 0 1 0 0 1 1 1 ...
##  $ BOVAG_Guarantee : num  1 1 1 1 1 1 1 1 1 1 ...
##  $ Guarantee_Period: num  0 0 0 0 0 ...
##  $ ABS             : num  1 1 1 1 1 1 1 1 1 1 ...
##  $ Airbag_1        : num  1 1 1 1 1 1 1 1 1 1 ...
##  $ Airbag_2        : num  1 1 1 1 1 0 1 1 1 1 ...
```

```
##  $ Airco           : num  1 0 0 1 1 1 1 1 1 1 ...
##  $ Automatic_airco : num  0 0 0 0 0 0 0 1 1 1 ...
##  $ Boardcomputer   : num  1 1 1 1 1 0 1 0 1 1 ...
##  $ CD_Player       : num  1 0 0 0 1 0 0 1 0 0 ...
##  $ Central_Lock    : num  1 0 0 1 1 1 0 1 1 1 ...
##  $ Powered_Windows : num  0 0 0 1 1 1 0 1 1 1 ...
##  $ Power_Steering  : num  1 1 1 1 1 1 1 1 1 1 ...
##  $ Radio           : num  0 0 0 0 0 1 0 0 0 0 ...
##  $ Mistlamps       : num  0 0 0 0 0 0 0 0 1 1 ...
##  $ Sport_Model     : num  0 0 0 1 0 0 0 0 1 1 ...
##  $ Backseat_Divider: num  1 1 1 1 1 0 1 0 1 1 ...
##  $ Metallic_Rim    : num  0 0 0 0 0 1 0 1 1 1 ...
##  $ Radio_cassette  : num  0 0 0 0 0 1 0 0 0 0 ...
##  $ Tow_Bar         : num  0 0 0 0 0 0 0 0 0 0 ...
```

Creating models

```r
library(neuralnet)
```

```
## Warning: package 'neuralnet' was built under R version 3.6.3
```

```r
nmodel1 <- neuralnet(Price~. -Cylinders -Color -Fuel_Type -Model-Id,
                    data = training,
                    hidden=1,
                    threshold = 1,
                    learningrate.limit = NULL,
                    algorithm = "rprop+")

plot(nmodel1)
```

```r
nmodel2 <- neuralnet(Price~. -Cylinders -Color -Fuel_Type -Model-Id,
                     data= training, hidden=1,
                    threshold = 0.1,
                    learningrate.limit = NULL,
                    algorithm = "rprop+")
plot(nmodel2)
```

```r
nmodel3 <- neuralnet(Price~. -Cylinders -Color -Fuel_Type -Model-Id, data = training, hidden=1,
                    threshold = 0.05,
                    learningrate.limit = NULL,
                    algorithm = "rprop+")
plot(nmodel3)
```

```r
nmodel4 <- neuralnet(Price~. -Cylinders -Color -Fuel_Type -Model-Id, data = training, hidden=1,
                    threshold = 0.01,
                    learningrate.limit = NULL,
                    algorithm = "rprop+")
plot(nmodel4)
```

```r
nmodel5 <- neuralnet(Price~. -Cylinders -Color -Fuel_Type -Model-Id, data = training, hidden=1,
                    threshold = 0.005,
```

```r
                    learningrate.limit = NULL,
                    algorithm = "rprop+")
plot(nmodel5)
```

```r
nmodel6 <- neuralnet(Price~. -Cylinders -Color -Fuel_Type -Model-Id, data = training, hidden=1,
                    threshold = 0.001,
                    learningrate.limit = NULL,
                    algorithm = "rprop+")
plot(nmodel6)
```

```r
nmodel7 <- neuralnet(Price~. -Cylinders -Color -Fuel_Type -Model-Id,
                    data = training, hidden=1,
                    threshold = 0.0001,
                    learningrate.limit = NULL,
                    algorithm = "rprop+")
```

```
## Warning: Algorithm did not converge in 1 of 1 repetition(s) within the stepmax.
```

```r
output1 <- compute(nmodel1, training[,-3])
head(output1$net.result)
```

```
##            [,1]
## [1,] 0.2261974
## [2,] 0.2261973
## [3,] 0.2261973
## [4,] 0.2261974
## [5,] 0.2261974
## [6,] 0.2261974
```

```r
head(training[3,])
```

```
## # A tibble: 1 x 38
##      Id Model Price Age_08_04 Mfg_Month Mfg_Year    KM Fuel_Type    HP Met_Color
##   <dbl> <chr> <dbl>     <dbl>     <dbl>    <dbl> <dbl> <fct>     <dbl>     <dbl>
## 1     4 TOYO~ 0.377     0.316     0.545    0.667 0.198 Diesel    0.171         0
## # ... with 28 more variables: Color <fct>, Automatic <dbl>, cc <dbl>,
## #   Doors <dbl>, Cylinders <dbl>, Gears <dbl>, Quarterly_Tax <dbl>,
## #   Weight <dbl>, Mfr_Guarantee <dbl>, BOVAG_Guarantee <dbl>,
## #   Guarantee_Period <dbl>, ABS <dbl>, Airbag_1 <dbl>, Airbag_2 <dbl>,
## #   Airco <dbl>, Automatic_airco <dbl>, Boardcomputer <dbl>, CD_Player <dbl>,
## #   Central_Lock <dbl>, Powered_Windows <dbl>, Power_Steering <dbl>,
## #   Radio <dbl>, Mistlamps <dbl>, Sport_Model <dbl>, Backseat_Divider <dbl>,
## #   Metallic_Rim <dbl>, Radio_cassette <dbl>, Tow_Bar <dbl>
```

```r
library(caret)
```

```
## Warning: package 'caret' was built under R version 3.6.3
```

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

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

```r
nmodel1.pred1 <- compute(nmodel1, training[,-3])
RMSE(nmodel1.pred1$net.result, training$Price)
```

```
## [1] 0.125237
```

```r
nmodel2.pred2 <- compute(nmodel2, training[,-3])
RMSE(nmodel2.pred2$net.result, training$Price)
```

```
## [1] 0.03889802
```

```r
nmodel3.pred3 <- compute(nmodel3, training[,-3])
RMSE(nmodel3.pred3$net.result, training$Price)
```

```
## [1] 0.03649143
```

```r
nmodel4.pred4 <- compute(nmodel4, training[,-3])
RMSE(nmodel4.pred4$net.result, training$Price)
```

```
## [1] 0.03527904
```

```r
nmodel4.pred4 <- compute(nmodel4, training[,-3])
RMSE(nmodel4.pred4$net.result, training$Price)
```

```
## [1] 0.03527904
```

```r
nmodel5.pred5 <- compute(nmodel5, training[,-3])
RMSE(nmodel5.pred5$net.result, training$Price)
```

```
## [1] 0.03524061
```

```r
nmodel6.pred6 <- compute(nmodel6, training[,-3])
RMSE(nmodel6.pred6$net.result, training$Price)
```

```
## [1] 0.03520978
```

#What happens to the RMS error (or Sum of Squares Error) for the training data as the value of threshold decreases?

The value of RMSE error decreases for the training data as the the value of threshold decreases from 1 to 0.0001

```r
nmodel1.red1 <- compute(nmodel1, testing[,-3])
RMSE(nmodel1.red1$net.result, testing$Price)
```

```
## [1] 0.1396461
```

```r
nmodel2.red2 <- compute(nmodel2, testing[,-3])
RMSE(nmodel2.red2$net.result, testing$Price)
```

```
## [1] 0.04920632
```

```r
nmodel3.red3 <- compute(nmodel3, testing[,-3])
RMSE(nmodel3.red3$net.result, testing$Price)
```

```
## [1] 0.04388745
```

```r
nmodel4.red4 <- compute(nmodel4, testing[,-3])
RMSE(nmodel4.red4$net.result, testing$Price)
```

```
## [1] 0.04209523
```

```r
nmodel5.red5 <- compute(nmodel5, testing[,-3])
RMSE(nmodel5.red5$net.result, testing$Price)
```

```
## [1] 0.04173473
```

```r
nmodel6.red6 <- compute(nmodel6, testing[,-3])
RMSE(nmodel6.red6$net.result, testing$Price)
```

```
## [1] 0.04124369
```

#What happens to the RMS error Sum of Squares Error for the validation data?

The RMSE value decreases from model 1 to model 7. Hence, suggesting that there is a strong dependent correlation between lowering threshold with that of RMSE value

```r
nmodel5.1 <- neuralnet(Price~. -Cylinders -Color -Fuel_Type -Model-Id, data = training, hidden=1,
                    threshold = 0.005,
                    learningrate.limit = NULL,
                    algorithm = "rprop+")
plot(nmodel5.1)
```

```r
nmodel5.2 <- neuralnet(Price~. -Cylinders -Color -Fuel_Type -Model-Id, data = training, hidden=2,
                    threshold = 0.005,
                    learningrate.limit = NULL,
                    algorithm = "rprop+")
plot(nmodel5)
```

```r
nmodel5.3 <- neuralnet(Price~. -Cylinders -Color -Fuel_Type -Model-Id, data = training, hidden=4,
                    threshold = 0.005,
                    learningrate.limit = NULL,
                    algorithm = "rprop+")
plot(nmodel5)
```

```r
nmodel5.4 <- neuralnet(Price~. -Cylinders -Color -Fuel_Type -Model-Id, data = training, hidden=8,
                    threshold = 0.005,
                    learningrate.limit = NULL,
                    algorithm = "rprop+")
plot(nmodel5)
```

```r
nmodel5.1.pred5 <- compute(nmodel5.1, training[,-3])
RMSE(nmodel5.1.pred5$net.result, training$Price)
```

```
## [1] 0.03523865
```

```r
nmodel5.1.red5 <- compute(nmodel5.1, testing[,-3])
RMSE(nmodel5.1.red5$net.result, testing$Price)
```

```
## [1] 0.04172574
```

```r
nmodel5.2.pred5 <- compute(nmodel5.2, training[,-3])
RMSE(nmodel5.2.pred5$net.result, training$Price)
```

```
## [1] 0.03330954
```

```r
nmodel5.2.red5 <- compute(nmodel5.2, testing[,-3])
RMSE(nmodel5.2.red5$net.result, testing$Price)
```

```
## [1] 0.04231841
```

```r
nmodel5.3.pred5 <- compute(nmodel5.3, training[,-3])
RMSE(nmodel5.3.pred5$net.result, training$Price)
```

```
## [1] 0.03069817
```

```r
nmodel5.3.red5 <- compute(nmodel5.3, testing[,-3])
RMSE(nmodel5.3.red5$net.result, testing$Price)
```

```
## [1] 0.0434862
```

```r
nmodel5.4.pred5 <- compute(nmodel5.4, training[,-3])
RMSE(nmodel5.4.pred5$net.result, training$Price)
```

```
## [1] 0.02605389
```

```r
nmodel5.4.red5 <- compute(nmodel5.4, testing[,-3])
RMSE(nmodel5.4.red5$net.result, testing$Price)
```

```
## [1] 0.04761447
```

#Conduct an experiment to assess the effect of changing the number of hiddenlayer nodes (default 1), e.g., 1,2,4,8. *The increase in hidden layer does not lead to decrease in RMSE value in testing data which is quite contradictory than that of the training data. This shows that increasing the number of hidden layers much more than the sufficient number of layers will cause accuracy in the test set to decrease. It causes the network to overfit to the training set, i.e., it will learn the training data, but it won't be able to generalize to new unseen data(testing data).*

```
nmodel5.a <- neuralnet(Price~. -Cylinders -Color -Fuel_Type -Model-Id, data = training, hidden=c(2),
                    threshold = 0.005,
                    learningrate.limit = NULL,
                    learningrate.factor =
                         list(minus = 0.5, plus = 1.2),
                    algorithm = "rprop+")
plot(nmodel5)
```

```
nmodel5.b <- neuralnet(Price~. -Cylinders -Color -Fuel_Type -Model-Id, data = training, hidden=c(2,2),
                    threshold = 0.005,
                    learningrate.limit = NULL,
                    learningrate.factor =
                         list(minus = 0.5, plus = 1.2),
                    algorithm = "rprop+")
plot(nmodel5)
```

```
nmodel5.a.pred5 <- compute(nmodel5.a, training[,-3])
RMSE(nmodel5.a.pred5$net.result, training$Price)
```

```
## [1] 0.03359912
```

```
nmodel5.a.red5 <- compute(nmodel5.a, testing[,-3])
RMSE(nmodel5.a.red5$net.result, testing$Price)
```

```
## [1] 0.04041993
```

```
nmodel5.b.pred5 <- compute(nmodel5.b, training[,-3])
RMSE(nmodel5.b.pred5$net.result, training$Price)
```

```
## [1] 0.03245532
```

```
nmodel5.b.red5 <- compute(nmodel5.b, testing[,-3])
RMSE(nmodel5.b.red5$net.result, testing$Price)
```

```
## [1] 0.04073952
```

#Conduct a similar experiment to assess the effect of changing the number oflayers from 1 to 2 in the network.

#Neural network model capacity is controlled both by the number of nodes and the number of layers in the model.A model with a single hidden layer and sufficient number of nodes has the capability of learning any mapping function, but the chosen learning algorithm may or may not be able to realize this capability.Increasing the number of layers provides a short-cut to increasing the capacity of the model with fewer resources, and modern techniques allow learning algorithms to successfully train deep models. #Whether changing the number of layers from 1 to 2 in a network would be profitable or not depends on the data itself. As for this data, it is only decrementing the accuracy level sugguesting that the model has been overfitted.

```r
nmodel11 <- neuralnet(Price~. -Cylinders -Color -Fuel_Type -Model-Id, data = training, hidden=1,
                     threshold = 1,
                     learningrate = 0.1,
                     algorithm = "rprop+")
plot(nmodel11)
```

```r
nmodel12 <- neuralnet(Price~. -Cylinders -Color -Fuel_Type -Model-Id, data = training, hidden=1,
                     threshold = 1,
                     learningrate = 0.01,
                     algorithm = "rprop+")
plot(nmodel2)
```

```r
nmodel13 <- neuralnet(Price~. -Cylinders -Color -Fuel_Type -Model-Id, data = training, hidden=1,
                     threshold = 1,
                     learningrate = 0.001,
                     algorithm = "rprop+")
plot(nmodel13)
```

```r
nmodel11pred <- compute(nmodel11, training[,-3])
RMSE(nmodel11pred$net.result, training$Price)
```

```
## [1] 0.1252575
```

```r
nmodel11red <- compute(nmodel11, testing[,-3])
RMSE(nmodel11red$net.result, testing$Price)
```

```
## [1] 0.139653
```

```r
nmodel12pred <- compute(nmodel12, training[,-3])
RMSE(nmodel12pred$net.result, training$Price)
```

```
## [1] 0.1099406
```

```r
nmodel12red <- compute(nmodel12, testing[,-3])
RMSE(nmodel12red$net.result, testing$Price)
```

```
## [1] 0.1242203
```

```r
nmodel13pred <- compute(nmodel13, training[,-3])
RMSE(nmodel13pred$net.result, training$Price)
```

```
## [1] 0.08550215
```

```r
nmodel13red <- compute(nmodel13, testing[,-3])
RMSE(nmodel13red$net.result, testing$Price)
```

```
## [1] 0.09941224
```

“‘ #Study the effect of gradient descent step size (learningrate) on the training process and the network performance. #Dereasing learning rate means leaning to smaller steps (from model 1 to model 3), which is leading to decrement in RMSE value. A learning rate that is too large can cause the model to converge too quickly to a suboptimal solution, whereas a learning rate that is too small can cause the process to get stuck. Hence, the optimum solution lies in testing with sufficient number of epochs and then selecting the learning rate.