Data Mining in Engineering HW-6 Group 7

R Markdown

```
toyota_cars <- read_excel("1B Toyota Corolla.xlsx")

toyota_cars <- toyota_cars[c(3,4,7,9,12,13,19)]
```

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

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

Divide the data set in training dataset of 75% and test dataset of 15%

```
training_set = sample(row.names(toyota_cars), dim(toyota_cars)[1]*0.75)
validation_set = setdiff(row.names(toyota_cars), training_set)
train_Dataset <- toyota_cars[training_set, c(1,2,3,4,9)]
valid_Dataset <- toyota_cars[validation_set, c(1,2,3,4,9)]
```

We are using neuralnet library and rprop+ algorithm

Neural Network with threshold = 1

```
nn1 <- neuralnet(Price ~ Age_08_04 + KM + HP + cc, data = train_Dataset, linear.output = FALSE, threshold = 1, al
gorithm = 'rprop+')
training_set.prediction1 <- compute(nn1, train_Dataset[,-1])
trainprednn1 <- training_set.prediction1$net.result
rmse_train1 <- sqrt(mean((trainprednn1 - train_Dataset$Price)^2))
validation_set.prediction1 <- compute(nn1, valid_Dataset[,-1])
validprednn1 <- validation_set.prediction1$net.result
rmse_valid1 <- sqrt(mean((validprednn1 - valid_Dataset$Price)^2))
```

Neural Network with threshold = 0.1

```
nn2 <- neuralnet(Price ~ Age_08_04 + KM + HP + cc, data = train_Dataset, linear.output = FALSE, threshold = 0.1,
 algorithm = 'rprop+')
training_set.prediction2 <- compute(nn2, train_Dataset[,-1])
trainprednn2 <- training_set.prediction2$net.result
rmse_train2 <- sqrt(mean((trainprednn2 - train_Dataset$Price)^2))
validation_set.prediction2 <- compute(nn2, valid_Dataset[,-1])
validprednn2 <- validation_set.prediction2$net.result
rmse_valid2 <- sqrt(mean((validprednn2 - valid_Dataset$Price)^2))
```

Neural Network with threshold = 0.05

```
nn3 <- neuralnet(Price ~ Age_08_04 + KM + HP + cc, data = train_Dataset, linear.output = FALSE, threshold = 0.05,
algorithm = 'rprop+')
training_set.prediction3 <- compute(nn3, train_Dataset[,-1])
trainprednn3 <- training_set.prediction3$net.result
rmse_train3 <- sqrt(mean((trainprednn3-train_Dataset$Price)^2))
validation_set.prediction3 <- compute(nn3, valid_Dataset[,-1])
validprednn3 <- validation_set.prediction3$net.result
rmse_valid3 <- sqrt(mean((validprednn3-valid_Dataset$Price)^2))
```

Neural Network with threshold = 0.01

```
nn4 <- neuralnet(Price ~ Age_08_04 + KM + HP + cc, data = train_Dataset, linear.output = FALSE,threshold = 0.01,
 algorithm = 'rprop+')
training_set.prediction4 <- compute(nn4, train_Dataset[,-1])
trainprednn4 <- training_set.prediction4$net.result
rmse_train4 <- sqrt(mean((trainprednn4-train_Dataset$Price)^2))
validation_set.prediction4 <- compute(nn4, valid_Dataset[,-1])
validprednn4<- validation_set.prediction4$net.result
rmse_valid4 <- sqrt(mean((validprednn4-valid_Dataset$Price)^2))
```

Neural Network with threshold = 0.005

```
nn5 <- neuralnet(Price ~ Age_08_04 + KM + HP + cc, data = train_Dataset, linear.output = FALSE, threshold = 0.005
, algorithm = 'rprop+')
training_set.prediction5 <- compute(nn5, train_Dataset[,-1])
trainprednn5 <- training_set.prediction5$net.result
rmse_train5 <- sqrt(mean((trainprednn5 - train_Dataset$Price)^2))
validation_set.prediction5 <- compute(nn5, valid_Dataset[,-1])
validprednn5 <- validation_set.prediction5$net.result
rmse_valid5 <- sqrt(mean((validprednn5 - valid_Dataset$Price)^2))
```

Neural Network with threshold = 0.001

```
nn6 <- neuralnet(Price ~ Age_08_04 + KM + HP + cc, data = train_Dataset, linear.output = FALSE, threshold = 0.001
, algorithm = 'rprop+')
training_set.prediction6 <- compute(nn6, train_Dataset[,-1])
trainprednn6 <- training_set.prediction6$net.result
rmse_train6 <- sqrt(mean((trainprednn6 - train_Dataset$Price)^2))
validation_set.prediction6 <- compute(nn6, valid_Dataset[,-1])
validprednn6 <- validation_set.prediction6$net.result
rmse_valid6 <- sqrt(mean((validprednn6 - valid_Dataset$Price)^2))
```

Neural Network with threshold = 0.0001

```
nn7 <- neuralnet(Price ~ Age_08_04 + KM + HP + cc, data = train_Dataset, linear.output = FALSE, threshold = 0.000
1, algorithm = 'rprop+')
training_set.prediction7 <- compute(nn7, train_Dataset[,-1])
trainprednn7 <- training_set.prediction7$net.result
rmse_train7 <- sqrt(mean((trainprednn7 - train_Dataset$Price)^2))
validation_set.prediction7 <- compute(nn7, valid_Dataset[,-1])
validprednn7 <- validation_set.prediction7$net.result
rmse_valid7 <- sqrt(mean((validprednn7 - valid_Dataset$Price)^2))
```

RMSE values for training and test dataset for Neural Networks with different threshold values

```
rmse_train <- rbind(rmse_train1, rmse_train2, rmse_train3, rmse_train4, rmse_train5, rmse_train6, rmse_train7)
rmse_valid <- rbind(rmse_valid1, rmse_valid2, rmse_valid3, rmse_valid4, rmse_valid5, rmse_valid6, rmse_valid7)
rmse_train
```

```
##                  [,1]
## rmse_train1 0.12697398
## rmse_train2 0.04925740
## rmse_train3 0.04766912
## rmse_train4 0.04886091
## rmse_train5 0.04740257
## rmse_train6 0.04737681
## rmse_train7 0.04737496
```

```
rmse_valid
```

```
##                  [,1]
## rmse_valid1 0.13372597
## rmse_valid2 0.04847269
## rmse_valid3 0.04632045
## rmse_valid4 0.04755198
## rmse_valid5 0.04633304
## rmse_valid6 0.04620736
## rmse_valid7 0.04617373
```

Neural Network with 2 nodes of hidden layers

```
nn8 <- neuralnet(Price ~ Age_08_04 + KM + HP + cc, data = train_Dataset, linear.output = FALSE, threshold = 1, al
gorithm = 'rprop+', hidden = 2)
training_set.prediction8 <- compute(nn8, train_Dataset[,-1])
trainprednn8 <- training_set.prediction8$net.result
rmse_train8 <- sqrt(mean((trainprednn8 - train_Dataset$Price)^2))
validation_set.prediction8 <- compute(nn8, valid_Dataset[,-1])
validprednn8 <- validation_set.prediction8$net.result
rmse_valid8 <- sqrt(mean((validprednn8 - valid_Dataset$Price)^2))
```

Neural Network with 4 nodes of hidden layers

```
nn9 <- neuralnet(Price ~ Age_08_04 + KM + HP + cc, data = train_Dataset, linear.output = FALSE, threshold = 1, al
gorithm = 'rprop+', hidden = 4)
training_set.prediction9 <- compute(nn9, train_Dataset[,-1])
trainprednn9 <- training_set.prediction9$net.result
rmse_train9 <- sqrt(mean((trainprednn9 - train_Dataset$Price)^2))
validation_set.prediction9 <- compute(nn9, valid_Dataset[,-1])
validprednn9 <- validation_set.prediction9$net.result
rmse_valid9 <- sqrt(mean((validprednn9 - valid_Dataset$Price)^2))
```

Neural Network with 8 nodes of hidden

```
nn10 <- neuralnet(Price ~ Age_08_04 + KM + HP + cc, data = train_Dataset, linear.output = FALSE, threshold = 1, a
lgorithm = 'rprop+', hidden = 8)
training_set.prediction10 <- compute(nn10, train_Dataset[,-1])
trainprednn10 <- training_set.prediction10$net.result
rmse_train10 <- sqrt(mean((trainprednn10 - train_Dataset$Price)^2))
validation_set.prediction10 <- compute(nn10, valid_Dataset[,-1])
validprednn10 <- validation_set.prediction10$net.result
rmse_valid10 <- sqrt(mean((validprednn10 - valid_Dataset$Price)^2))
```

RMSE values for training and testing dataset for Neural Networks with different number of hidden layer nodes

```
rmse_train1 <- rbind(rmse_train8, rmse_train9, rmse_train10)
rmse_valid1 <- rbind(rmse_valid8, rmse_valid9, rmse_valid10)
rmse_train1
```

```
##                  [,1]
## rmse_train8  0.06556738
## rmse_train9  0.05843847
## rmse_train10 0.05681545
```

```
rmse_valid1
```

```
##                  [,1]
## rmse_valid8  0.06956951
## rmse_valid9  0.06195549
## rmse_valid10 0.05891291
```

Neural Network with changing the number of layers from 1 to 2 in the network

```
nn11 <- neuralnet(Price ~ Age_08_04 + KM + HP + cc, data = train_Dataset, linear.output = FALSE, threshold = 1, a
lgorithm ='rprop+', hidden = c(1,2))
training_set.prediction11 <- compute(nn11, train_Dataset[,-1])
trainprednn11 <- training_set.prediction11$net.result
rmse_train11 <- sqrt(mean((trainprednn11-train_Dataset$Price)^2))
validation_set.prediction11 <- compute(nn11, valid_Dataset[,-1])
validprednn11 <- validation_set.prediction11$net.result
rmse_valid11 <- sqrt(mean((validprednn11-valid_Dataset$Price)^2))
rmse_train11
```

```
## [1] 0.1271689
```

```
rmse_valid11
```

```
## [1] 0.1339217
```

Neural Network with changing the number of layers from 2 to 1 in the network

```
nn12 <- neuralnet(Price ~ Age_08_04 + KM + HP + cc, data = train_Dataset, linear.output = FALSE, threshold = 1, a
lgorithm ='rprop+', hidden = c(2,1))
training_set.prediction12 <- compute(nn12, train_Dataset[,-1])
trainprednn12 <- training_set.prediction12$net.result
rmse_train12 <- sqrt(mean((trainprednn12 - train_Dataset$Price)^2))
validation_set.prediction12 <- compute(nn12, valid_Dataset[,-1])
validprednn12 <- validation_set.prediction12$net.result
rmse_valid12 <- sqrt(mean((validprednn12 - valid_Dataset$Price)^2))
rmse_train12
```

```
## [1] 0.127105
```

```
rmse_valid12
```

```
## [1] 0.1337948
```

Neural Network with learningrate

```
nn13 <- neuralnet(Price ~ Age_08_04 + KM + HP + cc, data = train_Dataset, linear.output = FALSE, threshold = 1, a
lgorithm = 'rprop+', learningrate = 1)
training_set.prediction13 <- compute(nn13, train_Dataset[,-1])
trainprednn13 <- training_set.prediction13$net.result
rmse_train13 <- sqrt(mean((trainprednn13 - train_Dataset$Price)^2))
validation_set.prediction13 <- compute(nn13, valid_Dataset[,-1])
validprednn13 <- validation_set.prediction13$net.result
rmse_valid13 <- sqrt(mean((validprednn13 - valid_Dataset$Price)^2))
rmse_train13
```

```
## [1] 0.05862504
```

```
rmse_valid13
```

```
## [1] 0.06669657
```

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

Answer: We can draw the conclusion from the output that as the threshold value decreases, the RMS error for training data set also decreases.

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

Answer: We can draw the conclusion from the output that as the threshold value decreases, the RMS error for the validation data set also decreases.

c. Conduct an experiment to assess the effect of changing the number of hidden layer nodes (default 1), e.g., 1,2,4,8.

Answer: There is a varying effect on the value of RMS error as per the number of hidden layer nodes. As we can observe from the output that the RMS error is highest for 2 hidden layer nodes following by 8 hidden layer nodes and lastly 4 hidden layer nodes.

d. Conduct a similar experiment to assess the effect of changing the number of hidden layers from 1 to 2 in the network.

Answer: After performing this experiment, we can observe that when the number of hidden layer increases, the value of RMS error decreases.