Introduction to Neural Networks

Jeff Shelton Okang'a

21/07/2020

Introduction to Neural Networks

NN are inspired by the **human brain** to perform a particular task or function. NN perform computations through a process by learning. The neural network is a set of connected input/output units in which each connection has a weight associated with it. In the learning phase, the network learns by adjusting the weights to predict the correct label of the given input.

Implementation of a Neural Network in R

Data Creation

The data to be used for this example contains two types of attributes or columns i.e. **Feature** and **label**. The columns are

- Technical Knowledge
- Communication Skill Score
- Placement

The first two are the features and the third column is the class label.

```
TKS<-c(20,10,30,20,80,30)
CSS<-c(90,20,40,50,50,80)
Placed<-c(1,0,0,0,1,1)
myData<-data.frame(TKS,CSS,Placed)
myData
```

```
##   TKS CSS Placed
## 1  20  90      1
## 2  10  20      0
## 3  30  40      0
## 4  20  50      0
## 5  80  50      1
## 6  30  80      1
```

We create a NN classifier using the neuralnet library in R. First we need to install the neuralnet library then import it.

```
library(neuralnet)
```

```
## Warning: package 'neuralnet' was built under R version 4.0.2
```

Creating the model

The model is of the form of $Placed \sim TKS + CSS$. The model will have 3 hidden layers (single layer with 3 neurons). We will also act.fct = logistic for the smoothing of the result. When logisric is used we make the linear.output = FALSE.

```
nn<- neuralnet(Placed~TKS+CSS,
               data = myData, hidden = 3,
               act.fct = "logistic", linear.output = FALSE)
```

Plotting the Results

Next we plot the neural net model created above.

```
plot(nn)
```

Creating the Test dataset

Create test data using features TKS and CSS (**these are features**)

```
TKS<- c(30,43,85)
CSS<- c(85,50,40)
testData<- data.frame(TKS,CSS)
testData
```

```
##   TKS CSS
## 1  30  85
## 2  43  50
## 3  85  40
```

Predict the results for the test set

```
predict<- compute(nn, testData)
predict$net.result
```

```
##            [,1]
## [1,] 0.99299994
## [2,] 0.02414341
## [3,] 0.96961017
```

We now convert the probabilities into binary classes.

```
prob<- predict$net.result
pred<- ifelse(prob>0.5, 1,0)
```

Conclusion

Neural Networks are more flexible and can be used with both regression and classification problems. Neural networks are good for nonlinear data sets with large number of inputs such as images. The number of layers can be increased. NN are much more like a black box, require more time for development and more computation power is needed. A neural network is the second best way to solve any problem , the best way is to actually understand the problem.