R_Deep_Learning_Iris

Ulises Jose Bustamante Mora

2023-11-27

Importing all necessary libraries

```
library(neuralnet)
library(mltools)
library(data.table)
library(caret)

## Loading required package: ggplot2

## Loading required package: lattice
```

Scaling the data to have a better results

```
scaled_iris = scale(iris[,1:4])
scaled_iris = as.data.frame(scaled_iris)
scaled_iris$Species = iris$Species

head(scaled_iris)

##   Sepal.Length Sepal.Width Petal.Length Petal.Width Species
## 1   -0.8976739  1.01560199    -1.335752   -1.311052  setosa
## 2   -1.1392005 -0.13153881    -1.335752   -1.311052  setosa
## 3   -1.3807271  0.32731751    -1.392399   -1.311052  setosa
## 4   -1.5014904  0.09788935    -1.279104   -1.311052  setosa
## 5   -1.0184372  1.24503015    -1.335752   -1.311052  setosa
## 6   -0.5353840  1.93331463    -1.165809   -1.048667  setosa
```

Splitting the data into training and testing

```
ratio = createDataPartition(1:dim(scaled_iris)[1], p = .7)
scaled_iris_train = scaled_iris[ratio$Resample1,]
scaled_iris_test = scaled_iris[-ratio$Resample1,]

dim(scaled_iris_train)

## [1] 106   5

dim(scaled_iris_test)

## [1] 44  5
```

Coverting the y variables into numeric by one hot coding and gather it with the original dataset

```
scaled_iris_train = cbind(scaled_iris_train[,1:4],
one_hot(as.data.table(scaled_iris_train[,5])))

head(scaled_iris_train)

##   Sepal.Length Sepal.Width Petal.Length Petal.Width V1_setosa
V1_versicolor
## 1   -0.8976739  1.01560199    -1.335752   -1.311052         1
0
## 2   -1.1392005 -0.13153881    -1.335752   -1.311052         1
0
## 3   -1.3807271  0.32731751    -1.392399   -1.311052         1
0
## 4   -1.5014904  0.09788935    -1.279104   -1.311052         1
0
## 5   -1.0184372  1.24503015    -1.335752   -1.311052         1
0
## 6   -0.5353840  1.93331463    -1.165809   -1.048667         1
0
##   V1_virginica
## 1            0
## 2            0
## 3            0
## 4            0
## 5            0
## 6            0

names(scaled_iris_train)

## [1] "Sepal.Length"  "Sepal.Width"   "Petal.Length"  "Petal.Width"
## [5] "V1_setosa"     "V1_versicolor" "V1_virginica"
```

Training the Deep Learning model

```
nn = neuralnet(V1_setosa +V1_versicolor + V1_virginica~
                 Sepal.Length+Sepal.Width+Petal.Length+Petal.Width,
               data = scaled_iris_train, hidden = c(4,3))
```

Plotting the model

```
plot(nn)
```

Making predictions

```
nn_pre = compute(nn, scaled_iris_test[,1:4])
nn_pre = as.data.frame(nn_pre$net.result)
head(nn_pre)
```

```
##           V1            V2           V3
## 11 0.9999595 -5.002961e-05 3.121817e-05
## 12 0.9999590 -4.959283e-05 3.090478e-05
## 14 0.9999595 -5.010129e-05 3.126960e-05
## 20 0.9999591 -4.965668e-05 3.095060e-05
## 22 0.9999576 -4.821078e-05 2.991318e-05
## 24 0.9999212 -1.213085e-05 4.026045e-06
```

Reverting the one hot coding to make it easier to understand

```
names(nn_pre)[1] = "setosa"
names(nn_pre)[2] = "versicolor"
names(nn_pre)[3] = "virginica"
head(nn_pre)
```

```
##       setosa    versicolor    virginica
## 11 0.9999595 -5.002961e-05 3.121817e-05
## 12 0.9999590 -4.959283e-05 3.090478e-05
## 14 0.9999595 -5.010129e-05 3.126960e-05
## 20 0.9999591 -4.965668e-05 3.095060e-05
## 22 0.9999576 -4.821078e-05 2.991318e-05
## 24 0.9999212 -1.213085e-05 4.026045e-06
```

Adding a new column with the predicted value

```
nn_pre$class = colnames(nn_pre[,1:3])[max.col(nn_pre[,1:3], ties.method =
"first")]
head(nn_pre)
```

```
##       setosa    versicolor    virginica  class
## 11 0.9999595 -5.002961e-05 3.121817e-05 setosa
## 12 0.9999590 -4.959283e-05 3.090478e-05 setosa
## 14 0.9999595 -5.010129e-05 3.126960e-05 setosa
## 20 0.9999591 -4.965668e-05 3.095060e-05 setosa
## 22 0.9999576 -4.821078e-05 2.991318e-05 setosa
## 24 0.9999212 -1.213085e-05 4.026045e-06 setosa
```

Checking by a confusion matrix the performance of the model

```
confu_matrix = table(nn_pre$class, scaled_iris_test$Species)
confu_matrix

##
##               setosa versicolor virginica
##   setosa          14          0         0
##   versicolor       0         14         2
##   virginica        0          1        13
```

Getting a result of 93% of accurate from our model

```
acc_rate = sum(diag(confu_matrix)*100/sum(confu_matrix))
acc_rate

## [1] 93.18182
```