ML\_REDES\_NEURAIS\_INTRO

Juliano Sarnes Longo

12/03/2022

## Redes Neurais Artificiais utilizando R

Para esse projeto introdutório utilizarei dois pacotes do CRAN para desenvolver as redes neurais artifiais. O primeiro pacote que utilizarei é o **caret** que possui diversas funções para treinamento e plotagem de modelos de classificação e regressão, o segundo pacote que utilizarei é o **neuralnet**, esse pacote será usado para o treinamento da rede neural.

### Entendo o objetivo

O objetivo é demonstrar o funcionamento das redes neurais e como aplicá-las em modelos clássicos de classificação e principalmente como saber percentual de acurária do modelo gerado usando uma matriz de confusão. Para esse mini projeto utilizei o dataset IRIS, que já faz parte dos pacotes do R Studio, a rede neural que será desenvolvida terá como objetivo conseguir classificar as três espécies da flor conhecida como “Iris”.

O dataset “Iris” foi criado a partir das características da flor Iris como descrito acima o objetivo é fazer a classificação “definir a classe ou rótulo” entre os três tipos que existem e que são: Setosa, Virginica e Versicolor.

O dataset “Iris” contém 50 observações para cada uma das espécies da flor Iris (Setosa, Virginica e Versicolor), sendo assim, nosso conjunto de dados possui um total de 150 observações e as variáveis morfológicas são: comprimento (Length) e largura (Width) da “pétala” e comprimento (Length) e largura (Width) da “sépala”. O objetivo original é quantificar a variação morfológica em relação a essas espécies com bases nas quatro variáveis de interesse.

head(iris)

## Sepal.Length Sepal.Width Petal.Length Petal.Width Species  
## 1 5.1 3.5 1.4 0.2 setosa  
## 2 4.9 3.0 1.4 0.2 setosa  
## 3 4.7 3.2 1.3 0.2 setosa  
## 4 4.6 3.1 1.5 0.2 setosa  
## 5 5.0 3.6 1.4 0.2 setosa  
## 6 5.4 3.9 1.7 0.4 setosa

Primeiro vou copiar os dados do dataset iris para um objeto do R que chamarei de ds\_iris.

ds\_iris <- iris

Agora irei criar 3 colunas (Setosa, Virginica e Versicolor) e setar valores TRUE ou FALSE. Essas colunas serão usadas na saída da rede neural e nos testes.

ds\_iris <- cbind(ds\_iris, ds\_iris$Species == 'setosa')  
ds\_iris <- cbind(ds\_iris, ds\_iris$Species == 'virginica')  
ds\_iris <- cbind(ds\_iris, ds\_iris$Species == 'versicolor')  
  
  
names(ds\_iris)[6] <- 'setosa'  
names(ds\_iris)[7] <- 'virginica'  
names(ds\_iris)[8] <- 'versicolor'  
  
head(ds\_iris)

## Sepal.Length Sepal.Width Petal.Length Petal.Width Species setosa virginica  
## 1 5.1 3.5 1.4 0.2 setosa TRUE FALSE  
## 2 4.9 3.0 1.4 0.2 setosa TRUE FALSE  
## 3 4.7 3.2 1.3 0.2 setosa TRUE FALSE  
## 4 4.6 3.1 1.5 0.2 setosa TRUE FALSE  
## 5 5.0 3.6 1.4 0.2 setosa TRUE FALSE  
## 6 5.4 3.9 1.7 0.4 setosa TRUE FALSE  
## versicolor  
## 1 FALSE  
## 2 FALSE  
## 3 FALSE  
## 4 FALSE  
## 5 FALSE  
## 6 FALSE

Agora irei construir as bases de treino e teste, sendo que a base de treino **ds\_iris\_treino** irá conter 70% das observações da base original, sendo assim, a base teste **ds\_iris\_teste** será composta de 30% das observações que restaram na base original.

#install.packages("caret", dependencies=T)  
library(caret)

## Warning: package 'caret' was built under R version 4.1.2

## Carregando pacotes exigidos: ggplot2

## Carregando pacotes exigidos: lattice

Criando as bases treino **ds\_iris\_treino** e teste **ds\_iris\_teste**.

# o comando dim retorna as dimensões do dataset iris nesse caso 150 observaçoes e 8 variáveis.  
dim(ds\_iris)

## [1] 150 8

#[1] 150 8  
  
# Como preciso criar um dataset de treino 'ds\_iris\_treino' com 70% das observações existentes   
# na base original e os 30% restantes usarei na base de teste 'ds\_iris\_teste' aplicarei a função  
# createDataPartition para criar as partições de treino e teste.  
part = createDataPartition(1:dim(ds\_iris)[1],p=.7)  
ds\_iris\_treino <- ds\_iris[part$Resample1,]  
ds\_iris\_teste <- ds\_iris[- part$Resample1,]

## Criando a Rede Neural Artifical ou simplesmente RNA.

As redes neurais artificais de treino e teste serão criadas usando o pacote **neuralnet** conforme abaixo:

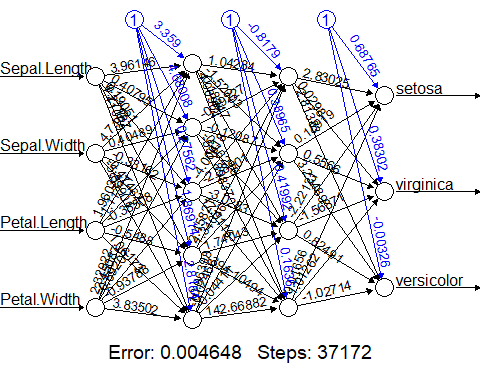
#install.packages("neuralnet")  
library(neuralnet)

## Warning: package 'neuralnet' was built under R version 4.1.2

# criei o objeto chamado de model\_treino para armazenar nossa rede neural de 'treino'.  
model\_treino = neuralnet( setosa + virginica + versicolor ~ Sepal.Length + Sepal.Width + Petal.Length + Petal.Width,   
 ds\_iris\_treino,   
 hidden=c(5,4), # hidden: são as camadas escondidas, estamos   
 # dizendo pro R colocar 5 neurônios na primeira camada  
 # escondida, e colocar 4 neurônios na segunda camada   
 # escondida.   
 act.fct = "logistic" # nessa RNA estou aplicando a função de ativação logística.  
 )

*A função de ativação*  ***logística*** *transforma a saída de um neurônio ANTES que o sinal seja passado para a camada seguinte.*

# plotando a RNA de treino.  
plot(model\_treino, rep = "best")



## Realizando o teste com base *ds\_iris\_teste*, gerando a *matriz de confusão* e exibindo o percentual de acertos (acurácia do modelo).

# executando a predição com a função "compute" do pacote neuralnet.  
test <- compute(model\_treino, ds\_iris\_teste[,1:4])  
  
# recuperando o resultado da predição a partir do atributo "net.result".  
resultado\_test <- as.data.frame(test$net.result)  
  
names(resultado\_test)[1] <- 'setosa'  
names(resultado\_test)[2] <- 'virginica'  
names(resultado\_test)[3] <- 'versicolor'  
  
resultado\_test$class <- colnames(resultado\_test[,1:3])[max.col(resultado\_test[,1:3],   
 ties.method = 'first')]  
  
# gerando a matriz de confusão para determinar a acurácia da RNA.  
matriz\_confusao = table(resultado\_test$class, ds\_iris\_teste$Species)  
matriz\_confusao

##   
## setosa versicolor virginica  
## setosa 14 0 0  
## versicolor 0 16 2  
## virginica 0 0 12

## Matriz de confusão (Acurácia do modelo).

setosa versicolor virginica

setosa 15 0 0 versicolor 0 15 2 virginica 0 1 11

A matriz de confusão acima indica que o modelo da Rede Neural Artificial teve 41 acertos, como que nossa amostra de teste possui 44 observações chegamos em uma acurácia de 93,18%.

\*\* acuracia <- (15+15+11)/(15+15+2+1+11) = 93,18 \*\*

**Percentual de acertos: 93,18 %**