Redes Neurais Artificiais

Exercício 10 - MLP

Vítor Gabriel Reis Caitité - 2016111849

3/17/2021

Enunciado Exercício 10

O objetivo dos exercícios desta semana é utilizar redes MLP para resolver problemas multidimensionais, a partir de bases de dados reais. Podem ser empregados pacotes de treinamento de redes neurais, como RSNNS (mostrado em vídeo aula) ou Scikit-Learn (para aqueles que preferem Python). As bases de dados devem ser baixadas do repositório UCI Machine Learning Repository.

As bases de dados devem ser baixadas do repositório UCI Machine Learning Repository:

• https://archive.ics.uci.edu/ml/datasets.php (https://archive.ics.uci.edu/ml/datasets.php)

A primeira base de dados a ser estudada é a base Boston Housing, disponível no link:

• https://archive.ics.uci.edu/ml/machine-learning-databases/housing/ (https://archive.ics.uci.edu/ml/machine-learning-databases/housing/)

Para esta base, o objetivo é prever o valor da variável M EDV . Os dados devem ser separados de forma aleatória entre treinamento e teste. Devem ser estudadas pelo menos 3 arquiteturas diferentes de rede neural (variando o número de neurônios e funções de ativação). Os valores de erro devem ser apresentados na forma de media ± desvio_padrao para, pelo menos, cinco execuções diferentes.

Uma breve discussão sobre o desempenho dos modelos deve ser apresentada.

O mesmo deve ser feito para o problema de classificação na base Statlog (Heart), disponível no link:

• https://archive.ics.uci.edu/ml/machine-learning-databases/statlog/heart/ (https://archive.ics.uci.edu/ml/machine-learning-databases/statlog/heart/)

Solução - Base de dados Boston Housing:

Para resolver esse exercício, utilizou-se o pacote de treinamento de redes neurais RSNNS, em R. Diversas arquiteturas de rede foram testadas e os respectivos códigos, bem como os resultados, são mostrados abaixo.

MLP com 3, 9 e 15 neurônios na camada escondida e função de ativação Act_Logistic

```
rm(list=ls())
library(caret)
```

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

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

```
library(RSNNS)
```

```
## Loading required package: Rcpp
```

```
##
## Attaching package: 'RSNNS'
```

```
## The following objects are masked from 'package:caret':
##
##     confusionMatrix, train
```

```
source("~/Documents/UFMG/9/Redes Neurais/exemplos/escalonamento_matrix.R")

# Carregando base de dados:
data <- read.table("~/Documents/UFMG/9/Redes Neurais/listas/lista 10/databases/housing.d
ata",
           sep = "", dec = ".", header = FALSE)

executions <- 5
for (p in c(3,9,15)) {
  executions <- 5
  for(exec in 1:executions){
    # Separando dados de entrada e saída e treino e teste aleatoriamente:
    partition <- createDataPartition(1:dim(data)[1],p=.7)
    train <- as.matrix(data[partition$Resample1,])
    test <- as.matrix(data[- partition$Resample1,])
    x_train <- as.matrix(train[, 1:(ncol(train)-1)])
    y_train <- as.matrix(train[, ncol(train)])
    x_test <- as.matrix(test[, 1:(ncol(train)-1)])
    y_test <- as.matrix(test[, ncol(train)])

    # Escalonando os valores dos atributos para que fiquem restritos entre 0 e 1
    x_all <- rbind(x_train, x_test)
    x_all <- staggeringMatrix(x_all, nrow(x_all), ncol(x_all))
    x_train <- x_all[1:nrow(x_train), ]
    x_test <- x_all[(nrow(x_train)+1):(nrow(x_train)+nrow(x_test)), ]

    # Criando modelo:
    model <- mlp(x_train, y_train, size = p, maxit = 1000, initFunc = "Randomize_Weight
s",
           initFuncParams = c(-0.3, 0.3), learnFunc = "Std_Backpropagation",
           learnFuncParams = c(0.01, 0.05), updateFunc = "Topological_Order",
           updateFuncParams = 0.0, hiddenActFunc = "Act_Logistic",
           shufflePatterns = TRUE, linOut = TRUE, inputsTest = NULL,
           targetsTest = NULL)

    # Testando:
    yhat <- predict(model, as.matrix(x_test))
    mean_error <- mean(abs(yhat - y_test))
    sd <- sd(yhat - y_test)
    mse <- sum((yhat - y_test)^2)/length(y_test)
    print(paste("Para p = ", p, ", o erro médio para a execução ", exec, "foi: ", round
(mean_error, 2), " +/- ", round(sd, 2)))
    print(paste("Para p = ", p, ", o erro quadrático médio para a execução ", exec, ": M
SE = ", round(mse, 2)))
    print("_____
_____")
  }
}
```

```
## [1] "Para p =  3 , o erro médio para a execução  1 foi:  2.9  +/-  4.29"
## [1] "Para p =  3 , o erro quadrático médio para a execução  1 : MSE =  20.69"
## [1] "_____________________________________________________________________________________________
__"
## [1] "Para p =  3 , o erro médio para a execução  2 foi:  3.54  +/-  4.69"
## [1] "Para p =  3 , o erro quadrático médio para a execução  2 : MSE =  22.84"
## [1] "_____________________________________________________________________________________________
__"
## [1] "Para p =  3 , o erro médio para a execução  3 foi:  3.58  +/-  4.7"
## [1] "Para p =  3 , o erro quadrático médio para a execução  3 : MSE =  25.07"
## [1] "_____________________________________________________________________________________________
__"
## [1] "Para p =  3 , o erro médio para a execução  4 foi:  3.5  +/-  5.2"
## [1] "Para p =  3 , o erro quadrático médio para a execução  4 : MSE =  26.94"
## [1] "_____________________________________________________________________________________________
__"
## [1] "Para p =  3 , o erro médio para a execução  5 foi:  3.29  +/-  4.19"
## [1] "Para p =  3 , o erro quadrático médio para a execução  5 : MSE =  17.85"
## [1] "_____________________________________________________________________________________________
__"
## [1] "Para p =  9 , o erro médio para a execução  1 foi:  2.52  +/-  3.4"
## [1] "Para p =  9 , o erro quadrático médio para a execução  1 : MSE =  12.51"
## [1] "_____________________________________________________________________________________________
__"
## [1] "Para p =  9 , o erro médio para a execução  2 foi:  2.4  +/-  3.54"
## [1] "Para p =  9 , o erro quadrático médio para a execução  2 : MSE =  12.48"
## [1] "_____________________________________________________________________________________________
__"
## [1] "Para p =  9 , o erro médio para a execução  3 foi:  2.46  +/-  3.25"
## [1] "Para p =  9 , o erro quadrático médio para a execução  3 : MSE =  10.99"
## [1] "_____________________________________________________________________________________________
__"
## [1] "Para p =  9 , o erro médio para a execução  4 foi:  2.21  +/-  3.21"
## [1] "Para p =  9 , o erro quadrático médio para a execução  4 : MSE =  10.34"
## [1] "_____________________________________________________________________________________________
__"
## [1] "Para p =  9 , o erro médio para a execução  5 foi:  2.39  +/-  3.66"
## [1] "Para p =  9 , o erro quadrático médio para a execução  5 : MSE =  13.28"
## [1] "_____________________________________________________________________________________________
__"
## [1] "Para p =  15 , o erro médio para a execução  1 foi:  2.29  +/-  3.51"
## [1] "Para p =  15 , o erro quadrático médio para a execução  1 : MSE =  12.41"
## [1] "_____________________________________________________________________________________________
__"
## [1] "Para p =  15 , o erro médio para a execução  2 foi:  2.25  +/-  3.3"
## [1] "Para p =  15 , o erro quadrático médio para a execução  2 : MSE =  10.9"
## [1] "_____________________________________________________________________________________________
__"
## [1] "Para p =  15 , o erro médio para a execução  3 foi:  2.23  +/-  3.32"
## [1] "Para p =  15 , o erro quadrático médio para a execução  3 : MSE =  10.98"
## [1] "_____________________________________________________________________________________________
__"
## [1] "Para p =  15 , o erro médio para a execução  4 foi:  2.27  +/-  3.5"
## [1] "Para p =  15 , o erro quadrático médio para a execução  4 : MSE =  12.19"
## [1] "_____________________________________________________________________________________________
__"
## [1] "Para p =  15 , o erro médio para a execução  5 foi:  2.92  +/-  5.46"
## [1] "Para p =  15 , o erro quadrático médio para a execução  5 : MSE =  30.09"
## [1] "_____________________________________________________________________________________________
__"
```

MLP com 3, 9 e 15 neurônios na camada escondida e função de ativação Act_TanH

Agora, trocou-se a função de ativação dos neurônios da camada escondida, optando-se pela função Act_TanH.

OBS: Para ver todas as funções de ativação disponíveis no pacote RSNNS, basta utilizar o comando: RSNNS::getSnnsRFunctionTable().

```
rm(list=ls())
library(caret)
library(RSNNS)
source("~/Documents/UFMG/9/Redes Neurais/exemplos/escalonamento_matrix.R")

# Carregando base de dados:
data <- read.table("~/Documents/UFMG/9/Redes Neurais/listas/lista 10/databases/housing.d
ata",
           sep = "", dec = ".", header = FALSE)

executions <- 5
for (p in c(3,9,15)) {
  executions <- 5
  for(exec in 1:executions){
    # Separando dados de entrada e saída e treino e teste aleatoriamente:
    partition <- createDataPartition(1:dim(data)[1],p=.7)
    train <- as.matrix(data[partition$Resample1,])
    test <- as.matrix(data[- partition$Resample1,])
    x_train <- as.matrix(train[, 1:(ncol(train)-1)])
    y_train <- as.matrix(train[, ncol(train)])
    x_test <- as.matrix(test[, 1:(ncol(train)-1)])
    y_test <- as.matrix(test[, ncol(train)])

    # Escalonando os valores dos atributos para que fiquem restritos entre 0 e 1
    x_all <- rbind(x_train, x_test)
    x_all <- staggeringMatrix(x_all, nrow(x_all), ncol(x_all))
    x_train <- x_all[1:nrow(x_train), ]
    x_test <- x_all[(nrow(x_train)+1):(nrow(x_train)+nrow(x_test)), ]

    # Criando modelo:
    model <- mlp(x_train, y_train, size = p, maxit = 1000, initFunc = "Randomize_Weight
s",
           initFuncParams = c(-0.3, 0.3), learnFunc = "Std_Backpropagation",
           learnFuncParams = c(0.01, 0.05), updateFunc = "Topological_Order",
           updateFuncParams = 0.0, hiddenActFunc = "Act_TanH",
           shufflePatterns = TRUE, linOut = TRUE, inputsTest = NULL,
           targetsTest = NULL)

    # Testando:
    yhat <- predict(model, as.matrix(x_test))
    mean_error <- mean(abs(yhat - y_test))
    sd <- sd(yhat - y_test)
    mse <- sum((yhat - y_test)^2)/length(y_test)
    print(paste("Para p = ", p, ", o erro médio para a execução ", exec, "foi: ", round
(mean_error, 2), " +/- ", round(sd, 2)))
    print(paste("Para p = ", p, ", o erro quadrático médio para a execução ", exec, ": M
SE = ", round(mse, 2)))
    print("_____
_____")
  }
}
```

```
## [1] "Para p =  3 , o erro médio para a execução  1 foi:  4.65  +/-  5.97"
## [1] "Para p =  3 , o erro quadrático médio para a execução  1 : MSE =  35.44"
## [1] "_____________________________________________________________________________________________
__"
## [1] "Para p =  3 , o erro médio para a execução  2 foi:  4.88  +/-  6.22"
## [1] "Para p =  3 , o erro quadrático médio para a execução  2 : MSE =  38.43"
## [1] "_____________________________________________________________________________________________
__"
## [1] "Para p =  3 , o erro médio para a execução  3 foi:  4.71  +/-  6.32"
## [1] "Para p =  3 , o erro quadrático médio para a execução  3 : MSE =  39.84"
## [1] "_____________________________________________________________________________________________
__"
## [1] "Para p =  3 , o erro médio para a execução  4 foi:  4.77  +/-  6.18"
## [1] "Para p =  3 , o erro quadrático médio para a execução  4 : MSE =  39.49"
## [1] "_____________________________________________________________________________________________
__"
## [1] "Para p =  3 , o erro médio para a execução  5 foi:  4.93  +/-  6.62"
## [1] "Para p =  3 , o erro quadrático médio para a execução  5 : MSE =  43.54"
## [1] "_____________________________________________________________________________________________
__"
## [1] "Para p =  9 , o erro médio para a execução  1 foi:  3.12  +/-  4.48"
## [1] "Para p =  9 , o erro quadrático médio para a execução  1 : MSE =  20.01"
## [1] "_____________________________________________________________________________________________
__"
## [1] "Para p =  9 , o erro médio para a execução  2 foi:  3.04  +/-  4.13"
## [1] "Para p =  9 , o erro quadrático médio para a execução  2 : MSE =  18.13"
## [1] "_____________________________________________________________________________________________
__"
## [1] "Para p =  9 , o erro médio para a execução  3 foi:  3.57  +/-  5.39"
## [1] "Para p =  9 , o erro quadrático médio para a execução  3 : MSE =  28.92"
## [1] "_____________________________________________________________________________________________
__"
## [1] "Para p =  9 , o erro médio para a execução  4 foi:  3.54  +/-  5.26"
## [1] "Para p =  9 , o erro quadrático médio para a execução  4 : MSE =  29.45"
## [1] "_____________________________________________________________________________________________
__"
## [1] "Para p =  9 , o erro médio para a execução  5 foi:  3.22  +/-  4.91"
## [1] "Para p =  9 , o erro quadrático médio para a execução  5 : MSE =  26.32"
## [1] "_____________________________________________________________________________________________
__"
## [1] "Para p =  15 , o erro médio para a execução  1 foi:  3.12  +/-  4.58"
## [1] "Para p =  15 , o erro quadrático médio para a execução  1 : MSE =  24.17"
## [1] "_____________________________________________________________________________________________
__"
## [1] "Para p =  15 , o erro médio para a execução  2 foi:  2.62  +/-  3.93"
## [1] "Para p =  15 , o erro quadrático médio para a execução  2 : MSE =  15.38"
## [1] "_____________________________________________________________________________________________
__"
## [1] "Para p =  15 , o erro médio para a execução  3 foi:  2.45  +/-  3.33"
## [1] "Para p =  15 , o erro quadrático médio para a execução  3 : MSE =  11.02"
## [1] "_____________________________________________________________________________________________
__"
## [1] "Para p =  15 , o erro médio para a execução  4 foi:  2.24  +/-  3.18"
## [1] "Para p =  15 , o erro quadrático médio para a execução  4 : MSE =  10.11"
## [1] "_____________________________________________________________________________________________
__"
## [1] "Para p =  15 , o erro médio para a execução  5 foi:  2.64  +/-  3.67"
## [1] "Para p =  15 , o erro quadrático médio para a execução  5 : MSE =  13.78"
## [1] "_____________________________________________________________________________________________
__"
```

MLP com 3, 9 e 15 neurônios na camada escondida e função de ativação Act_Signum

```
rm(list=ls())
library(caret)
library(RSNNS)
source("~/Documents/UFMG/9/Redes Neurais/exemplos/escalonamento_matrix.R")

# Carregando base de dados:
data <- read.table("~/Documents/UFMG/9/Redes Neurais/listas/lista 10/databases/housing.d
ata",
           sep = "", dec = ".", header = FALSE)

executions <- 5
for (p in c(6,9,12)) {
  executions <- 5
  for(exec in 1:executions){
    # Separando dados de entrada e saída e treino e teste aleatoriamente:
    partition <- createDataPartition(1:dim(data)[1],p=.7)
    train <- as.matrix(data[partition$Resample1,])
    test <- as.matrix(data[- partition$Resample1,])
    x_train <- as.matrix(train[, 1:(ncol(train)-1)])
    y_train <- as.matrix(train[, ncol(train)])
    x_test <- as.matrix(test[, 1:(ncol(train)-1)])
    y_test <- as.matrix(test[, ncol(train)])

    # Escalonando os valores dos atributos para que fiquem restritos entre 0 e 1
    x_all <- rbind(x_train, x_test)
    x_all <- staggeringMatrix(x_all, nrow(x_all), ncol(x_all))
    x_train <- x_all[1:nrow(x_train), ]
    x_test <- x_all[(nrow(x_train)+1):(nrow(x_train)+nrow(x_test)), ]

    # Criando modelo:
    model <- mlp(x_train, y_train, size = p, maxit = 1000, initFunc = "Randomize_Weight
s",
           initFuncParams = c(-0.3, 0.3), learnFunc = "Std_Backpropagation",
           learnFuncParams = c(0.01, 0.05), updateFunc = "Topological_Order",
           updateFuncParams = 0.0, hiddenActFunc = "Act_Signum",
           shufflePatterns = TRUE, linOut = TRUE, inputsTest = NULL,
           targetsTest = NULL)

    # Testando:
    yhat <- predict(model, as.matrix(x_test))
    mean_error <- mean(abs(yhat - y_test))
    sd <- sd(yhat - y_test)
    mse <- sum((yhat - y_test)^2)/length(y_test)
    print(paste("Para p = ", p, ", o erro médio para a execução ", exec, "foi: ", round
(mean_error, 2), " +/- ", round(sd, 2)))
    print(paste("Para p = ", p, ", o erro quadrático médio para a execução ", exec, ": M
SE = ", round(mse, 2)))
    print("_____
_____")
  }
}
```

```
## [1] "Para p =  6 , o erro médio para a execução  1 foi:  6.76  +/-  8.23"
## [1] "Para p =  6 , o erro quadrático médio para a execução  1 : MSE =  69.11"
## [1] "_____________________________________________________________________________________________
__"
## [1] "Para p =  6 , o erro médio para a execução  2 foi:  4.99  +/-  6.96"
## [1] "Para p =  6 , o erro quadrático médio para a execução  2 : MSE =  48.63"
## [1] "_____________________________________________________________________________________________
__"
## [1] "Para p =  6 , o erro médio para a execução  3 foi:  5.33  +/-  7.98"
## [1] "Para p =  6 , o erro quadrático médio para a execução  3 : MSE =  66.31"
## [1] "_____________________________________________________________________________________________
__"
## [1] "Para p =  6 , o erro médio para a execução  4 foi:  6.05  +/-  8.39"
## [1] "Para p =  6 , o erro quadrático médio para a execução  4 : MSE =  71.65"
## [1] "_____________________________________________________________________________________________
__"
## [1] "Para p =  6 , o erro médio para a execução  5 foi:  6.63  +/-  9.9"
## [1] "Para p =  6 , o erro quadrático médio para a execução  5 : MSE =  98.4"
## [1] "_____________________________________________________________________________________________
__"
## [1] "Para p =  9 , o erro médio para a execução  1 foi:  5.56  +/-  8.11"
## [1] "Para p =  9 , o erro quadrático médio para a execução  1 : MSE =  70.72"
## [1] "_____________________________________________________________________________________________
__"
## [1] "Para p =  9 , o erro médio para a execução  2 foi:  6.31  +/-  9.15"
## [1] "Para p =  9 , o erro quadrático médio para a execução  2 : MSE =  83.36"
## [1] "_____________________________________________________________________________________________
__"
## [1] "Para p =  9 , o erro médio para a execução  3 foi:  6.32  +/-  9.18"
## [1] "Para p =  9 , o erro quadrático médio para a execução  3 : MSE =  86.38"
## [1] "_____________________________________________________________________________________________
__"
## [1] "Para p =  9 , o erro médio para a execução  4 foi:  6.38  +/-  9.09"
## [1] "Para p =  9 , o erro quadrático médio para a execução  4 : MSE =  82.19"
## [1] "_____________________________________________________________________________________________
__"
## [1] "Para p =  9 , o erro médio para a execução  5 foi:  6.08  +/-  7.65"
## [1] "Para p =  9 , o erro quadrático médio para a execução  5 : MSE =  59.7"
## [1] "_____________________________________________________________________________________________
__"
## [1] "Para p =  12 , o erro médio para a execução  1 foi:  6.38  +/-  10.22"
## [1] "Para p =  12 , o erro quadrático médio para a execução  1 : MSE =  105.35"
## [1] "_____________________________________________________________________________________________
__"
## [1] "Para p =  12 , o erro médio para a execução  2 foi:  5.31  +/-  8.07"
## [1] "Para p =  12 , o erro quadrático médio para a execução  2 : MSE =  67.96"
## [1] "_____________________________________________________________________________________________
__"
## [1] "Para p =  12 , o erro médio para a execução  3 foi:  5.24  +/-  7.47"
## [1] "Para p =  12 , o erro quadrático médio para a execução  3 : MSE =  56.09"
## [1] "_____________________________________________________________________________________________
__"
## [1] "Para p =  12 , o erro médio para a execução  4 foi:  6.03  +/-  9.22"
## [1] "Para p =  12 , o erro quadrático médio para a execução  4 : MSE =  84.67"
## [1] "_____________________________________________________________________________________________
__"
## [1] "Para p =  12 , o erro médio para a execução  5 foi:  5.18  +/-  7.57"
## [1] "Para p =  12 , o erro quadrático médio para a execução  5 : MSE =  56.97"
## [1] "_____________________________________________________________________________________________
__"
```

Solução - Base de dados Statlog (Heart):

MLP com 3, 9 e 15 neurônios na camada escondida e função de ativação Act_Logistic

```
rm(list=ls())
library(caret)
library(RSNNS)
source("~/Documents/UFMG/9/Redes Neurais/exemplos/escalonamento_matrix.R")

# Carregando base de dados:
data <- read.table("~/Documents/UFMG/9/Redes Neurais/listas/lista 10/databases/heart.dat",
                   sep = "", dec = ".", header = FALSE)

executions <- 5
for (p in c(6,9,12)) {
  error <- rep(0, 5)
  executions <- 5
  for(exec in 1:executions){
    # Separando dados de entrada e saída e treino e teste aleatoriamente:
    partition <- createDataPartition(1:dim(data)[1],p=.7)
    train <- as.matrix(data[partition$Resample1,])
    test <- as.matrix(data[- partition$Resample1,])
    x_train <- as.matrix(train[, 1:(ncol(train)-1)])
    y_train <- as.matrix(train[, ncol(train)])
    y_train <- ifelse(y_train == 2, -1, 1)
    x_test <- as.matrix(test[, 1:(ncol(train)-1)])
    y_test <- as.matrix(test[, ncol(train)])
    y_test <- ifelse(y_test == 2, -1, 1)

    # Escalonando os valores dos atributos para que fiquem restritos entre 0 e 1
    x_all <- rbind(x_train, x_test)
    x_all <- staggeringMatrix(x_all, nrow(x_all), ncol(x_all))
    x_train <- x_all[1:nrow(x_train), ]
    x_test <- x_all[(nrow(x_train)+1):(nrow(x_train)+nrow(x_test)), ]

    # Criando modelo:
    model <- mlp(x_train, y_train, size = p, maxit = 1000, initFunc = "Randomize_Weights",
                 initFuncParams = c(-0.3, 0.3), learnFunc = "Std_Backpropagation",
                 learnFuncParams = c(0.01, 0.05), updateFunc = "Topological_Order",
                 updateFuncParams = 0.0, hiddenActFunc = "Act_Logistic",
                 shufflePatterns = TRUE, linOut = TRUE, inputsTest = NULL,
                 targetsTest = NULL)

    # Testando:
    yhat <- predict(model, as.matrix(x_test))
    accuracy<-((sum(abs(y_test + yhat)))/2)/length(y_test)
    error[exec] <- 1 - accuracy
    print(paste("Para p = ", p, ", o erro para a execução ", exec, "foi: ", round(error[exec], 2)))
    print(paste("Para p = ", p, ", a acurácia para a execução ", exec, "foi: ", round(accuracy, 2)))
    print("_____")
  }
  print(paste(">>> Para p = ", p, ", a média do erro foi: ", round(mean(error), 2), "+/- ", round(sd(error), 2)))
  print("_____")
}
```

```
## [1] "Para p =  6 , o erro para a execução  1 foi:  0.18"
## [1] "Para p =  6 , a acurácia para a execução  1 foi:  0.82"
## [1] "_____________________________________________________________________________________________
__"
## [1] "Para p =  6 , o erro para a execução  2 foi:  0.25"
## [1] "Para p =  6 , a acurácia para a execução  2 foi:  0.75"
## [1] "_____________________________________________________________________________________________
__"
## [1] "Para p =  6 , o erro para a execução  3 foi:  0.23"
## [1] "Para p =  6 , a acurácia para a execução  3 foi:  0.77"
## [1] "_____________________________________________________________________________________________
__"
## [1] "Para p =  6 , o erro para a execução  4 foi:  0.23"
## [1] "Para p =  6 , a acurácia para a execução  4 foi:  0.77"
## [1] "_____________________________________________________________________________________________
__"
## [1] "Para p =  6 , o erro para a execução  5 foi:  0.24"
## [1] "Para p =  6 , a acurácia para a execução  5 foi:  0.76"
## [1] "_____________________________________________________________________________________________
__"
## [1] ">>> Para p =  6 , a média do erro foi:  0.22 +/- 0.03"
## [1] "_____________________________________________________________________________________________
__"
## [1] "Para p =  9 , o erro para a execução  1 foi:  0.24"
## [1] "Para p =  9 , a acurácia para a execução  1 foi:  0.76"
## [1] "_____________________________________________________________________________________________
__"
## [1] "Para p =  9 , o erro para a execução  2 foi:  0.18"
## [1] "Para p =  9 , a acurácia para a execução  2 foi:  0.82"
## [1] "_____________________________________________________________________________________________
__"
## [1] "Para p =  9 , o erro para a execução  3 foi:  0.28"
## [1] "Para p =  9 , a acurácia para a execução  3 foi:  0.72"
## [1] "_____________________________________________________________________________________________
__"
## [1] "Para p =  9 , o erro para a execução  4 foi:  0.25"
## [1] "Para p =  9 , a acurácia para a execução  4 foi:  0.75"
## [1] "_____________________________________________________________________________________________
__"
## [1] "Para p =  9 , o erro para a execução  5 foi:  0.21"
## [1] "Para p =  9 , a acurácia para a execução  5 foi:  0.79"
## [1] "_____________________________________________________________________________________________
__"
## [1] ">>> Para p =  9 , a média do erro foi:  0.23 +/- 0.04"
## [1] "_____________________________________________________________________________________________
__"
## [1] "Para p =  12 , o erro para a execução  1 foi:  0.22"
## [1] "Para p =  12 , a acurácia para a execução  1 foi:  0.78"
## [1] "_____________________________________________________________________________________________
__"
## [1] "Para p =  12 , o erro para a execução  2 foi:  0.26"
## [1] "Para p =  12 , a acurácia para a execução  2 foi:  0.74"
## [1] "_____________________________________________________________________________________________
__"
## [1] "Para p =  12 , o erro para a execução  3 foi:  0.23"
## [1] "Para p =  12 , a acurácia para a execução  3 foi:  0.77"
## [1] "_____________________________________________________________________________________________
__"
## [1] "Para p =  12 , o erro para a execução  4 foi:  0.23"
## [1] "Para p =  12 , a acurácia para a execução  4 foi:  0.77"
## [1] "_____________________________________________________________________________________________
__"
## [1] "Para p =  12 , o erro para a execução  5 foi:  0.32"
## [1] "Para p =  12 , a acurácia para a execução  5 foi:  0.68"
## [1] "_____________________________________________________________________________________________
__"
## [1] ">>> Para p =  12 , a média do erro foi:  0.25 +/- 0.04"
## [1] "_____________________________________________________________________________________________
__"
```

MLP com 3, 9 e 15 neurônios na camada escondida e função de ativação Act_TanH

```
rm(list=ls())
library(caret)
library(RSNNS)
source("~/Documents/UFMG/9/Redes Neurais/exemplos/escalonamento_matrix.R")

# Carregando base de dados:
data <- read.table("~/Documents/UFMG/9/Redes Neurais/listas/lista 10/databases/heart.da
t",
                   sep = "", dec = ".", header = FALSE)

executions <- 5
for (p in c(3,9,15)) {
  error <- rep(0, 5)
  executions <- 5
  for(exec in 1:executions){
    # Separando dados de entrada e saída e treino e teste aleatoriamente:
    partition <- createDataPartition(1:dim(data)[1],p=.7)
    train <- as.matrix(data[partition$Resample1,])
    test <- as.matrix(data[- partition$Resample1,])
    x_train <- as.matrix(train[, 1:(ncol(train)-1)])
    y_train <- as.matrix(train[, ncol(train)])
    y_train <- ifelse(y_train == 2, -1, 1)
    x_test <- as.matrix(test[, 1:(ncol(train)-1)])
    y_test <- as.matrix(test[, ncol(train)])
    y_test <- ifelse(y_test == 2, -1, 1)

    # Escalonando os valores dos atributos para que fiquem restritos entre 0 e 1
    x_all <- rbind(x_train, x_test)
    x_all <- staggeringMatrix(x_all, nrow(x_all), ncol(x_all))
    x_train <- x_all[1:nrow(x_train), ]
    x_test <- x_all[(nrow(x_train)+1):(nrow(x_train)+nrow(x_test)), ]

    # Criando modelo:
    model <- mlp(x_train, y_train, size = p, maxit = 1000, initFunc = "Randomize_Weight
s",
                 initFuncParams = c(-0.3, 0.3), learnFunc = "Std_Backpropagation",
                 learnFuncParams = c(0.01, 0.1), updateFunc = "Topological_Order",
                 updateFuncParams = 0.0, hiddenActFunc = "Act_TanH",
                 shufflePatterns = TRUE, linOut = TRUE, inputsTest = NULL,
                 targetsTest = NULL)

    # Testando:
    yhat <- predict(model, as.matrix(x_test))
    accuracy<-((sum(abs(y_test + yhat)))/2)/length(y_test)
    error[exec] <- 1 - accuracy
    print(paste("Para p = ", p, ", o erro para a execução ", exec, "foi: ", round(error
[exec], 2)))
    print(paste("Para p = ", p, ", a acurácia para a execução ", exec, "foi: ", round(ac
curacy, 2)))
    print("_____
_____")
  }
  print(paste(">>> Para p = ", p, ", a média do erro foi: ", round(mean(error), 2), "+/-
", round(sd(error), 2)))
  print("_____
___")

}
```

```
## [1] "Para p =  3 , o erro para a execução  1 foi:  0.27"
## [1] "Para p =  3 , a acurácia para a execução  1 foi:  0.73"
## [1] "_____________________________________________________________________________________________
__"
## [1] "Para p =  3 , o erro para a execução  2 foi:  0.26"
## [1] "Para p =  3 , a acurácia para a execução  2 foi:  0.74"
## [1] "_____________________________________________________________________________________________
__"
## [1] "Para p =  3 , o erro para a execução  3 foi:  0.25"
## [1] "Para p =  3 , a acurácia para a execução  3 foi:  0.75"
## [1] "_____________________________________________________________________________________________
__"
## [1] "Para p =  3 , o erro para a execução  4 foi:  0.25"
## [1] "Para p =  3 , a acurácia para a execução  4 foi:  0.75"
## [1] "_____________________________________________________________________________________________
__"
## [1] "Para p =  3 , o erro para a execução  5 foi:  0.25"
## [1] "Para p =  3 , a acurácia para a execução  5 foi:  0.75"
## [1] "_____________________________________________________________________________________________
__"
## [1] ">>> Para p =  3 , a média do erro foi:  0.26 +/- 0.01"
## [1] "_____________________________________________________________________________________________
__"
## [1] "Para p =  9 , o erro para a execução  1 foi:  0.2"
## [1] "Para p =  9 , a acurácia para a execução  1 foi:  0.8"
## [1] "_____________________________________________________________________________________________
__"
## [1] "Para p =  9 , o erro para a execução  2 foi:  0.2"
## [1] "Para p =  9 , a acurácia para a execução  2 foi:  0.8"
## [1] "_____________________________________________________________________________________________
__"
## [1] "Para p =  9 , o erro para a execução  3 foi:  0.13"
## [1] "Para p =  9 , a acurácia para a execução  3 foi:  0.87"
## [1] "_____________________________________________________________________________________________
__"
## [1] "Para p =  9 , o erro para a execução  4 foi:  0.11"
## [1] "Para p =  9 , a acurácia para a execução  4 foi:  0.89"
## [1] "_____________________________________________________________________________________________
__"
## [1] "Para p =  9 , o erro para a execução  5 foi:  0.21"
## [1] "Para p =  9 , a acurácia para a execução  5 foi:  0.79"
## [1] "_____________________________________________________________________________________________
__"
## [1] ">>> Para p =  9 , a média do erro foi:  0.17 +/- 0.05"
## [1] "_____________________________________________________________________________________________
__"
## [1] "Para p =  15 , o erro para a execução  1 foi:  0.16"
## [1] "Para p =  15 , a acurácia para a execução  1 foi:  0.84"
## [1] "_____________________________________________________________________________________________
__"
## [1] "Para p =  15 , o erro para a execução  2 foi:  0.28"
## [1] "Para p =  15 , a acurácia para a execução  2 foi:  0.72"
## [1] "_____________________________________________________________________________________________
__"
## [1] "Para p =  15 , o erro para a execução  3 foi:  0.23"
## [1] "Para p =  15 , a acurácia para a execução  3 foi:  0.77"
## [1] "_____________________________________________________________________________________________
__"
## [1] "Para p =  15 , o erro para a execução  4 foi:  0.14"
## [1] "Para p =  15 , a acurácia para a execução  4 foi:  0.86"
## [1] "_____________________________________________________________________________________________
__"
## [1] "Para p =  15 , o erro para a execução  5 foi:  0.25"
## [1] "Para p =  15 , a acurácia para a execução  5 foi:  0.75"
## [1] "_____________________________________________________________________________________________
__"
## [1] ">>> Para p =  15 , a média do erro foi:  0.21 +/- 0.06"
## [1] "_____________________________________________________________________________________________
__"
```

MLP com 3, 9 e 15 neurônios na camada escondida e função de ativação Act_Signum

```
rm(list=ls())
library(caret)
library(RSNNS)
source("~/Documents/UFMG/9/Redes Neurais/exemplos/escalonamento_matrix.R")

# Carregando base de dados:
data <- read.table("~/Documents/UFMG/9/Redes Neurais/listas/lista 10/databases/heart.da
t",
                   sep = "", dec = ".", header = FALSE)

executions <- 5
for (p in c(3,9,15)) {
  error <- rep(0, 5)
  executions <- 5
  for(exec in 1:executions){
    # Separando dados de entrada e saída e treino e teste aleatoriamente:
    partition <- createDataPartition(1:dim(data)[1],p=.7)
    train <- as.matrix(data[partition$Resample1,])
    test <- as.matrix(data[- partition$Resample1,])
    x_train <- as.matrix(train[, 1:(ncol(train)-1)])
    y_train <- as.matrix(train[, ncol(train)])
    y_train <- ifelse(y_train == 2, -1, 1)
    x_test <- as.matrix(test[, 1:(ncol(train)-1)])
    y_test <- as.matrix(test[, ncol(train)])
    y_test <- ifelse(y_test == 2, -1, 1)

    # Escalonando os valores dos atributos para que fiquem restritos entre 0 e 1
    x_all <- rbind(x_train, x_test)
    x_all <- staggeringMatrix(x_all, nrow(x_all), ncol(x_all))
    x_train <- x_all[1:nrow(x_train), ]
    x_test <- x_all[(nrow(x_train)+1):(nrow(x_train)+nrow(x_test)), ]

    # Criando modelo:
    model <- mlp(x_train, y_train, size = p, maxit = 1000, initFunc = "Randomize_Weight
s",
                 initFuncParams = c(-0.3, 0.3), learnFunc = "Std_Backpropagation",
                 learnFuncParams = c(0.01, 0.1), updateFunc = "Topological_Order",
                 updateFuncParams = 0.0, hiddenActFunc = "Act_Signum",
                 shufflePatterns = TRUE, linOut = TRUE, inputsTest = NULL,
                 targetsTest = NULL)

    # Testando:
    yhat <- predict(model, as.matrix(x_test))
    accuracy<-((sum(abs(y_test + yhat)))/2)/length(y_test)
    error[exec] <- 1 - accuracy
    print(paste("Para p = ", p, ", o erro para a execução ", exec, "foi: ", round(error
[exec], 2)))
    print(paste("Para p = ", p, ", a acurácia para a execução ", exec, "foi: ", round(ac
curacy, 2)))
    print("_____
_____")
  }
  print(paste(">>> Para p = ", p, ", a média do erro foi: ", round(mean(error), 2), "+/-
", round(sd(error), 2)))
  print("_____
___")
}
```

```
## [1] "Para p =  3 , o erro para a execução  1 foi:  0.35"
## [1] "Para p =  3 , a acurácia para a execução  1 foi:  0.65"
## [1] "_____________________________________________________________________________________________
__"
## [1] "Para p =  3 , o erro para a execução  2 foi:  0.25"
## [1] "Para p =  3 , a acurácia para a execução  2 foi:  0.75"
## [1] "_____________________________________________________________________________________________
__"
## [1] "Para p =  3 , o erro para a execução  3 foi:  0.24"
## [1] "Para p =  3 , a acurácia para a execução  3 foi:  0.76"
## [1] "_____________________________________________________________________________________________
__"
## [1] "Para p =  3 , o erro para a execução  4 foi:  0.33"
## [1] "Para p =  3 , a acurácia para a execução  4 foi:  0.67"
## [1] "_____________________________________________________________________________________________
__"
## [1] "Para p =  3 , o erro para a execução  5 foi:  0.25"
## [1] "Para p =  3 , a acurácia para a execução  5 foi:  0.75"
## [1] "_____________________________________________________________________________________________
__"
## [1] ">>> Para p =  3 , a média do erro foi:  0.28 +/- 0.05"
## [1] "_____________________________________________________________________________________________
__"
## [1] "Para p =  9 , o erro para a execução  1 foi:  0.19"
## [1] "Para p =  9 , a acurácia para a execução  1 foi:  0.81"
## [1] "_____________________________________________________________________________________________
__"
## [1] "Para p =  9 , o erro para a execução  2 foi:  0.21"
## [1] "Para p =  9 , a acurácia para a execução  2 foi:  0.79"
## [1] "_____________________________________________________________________________________________
__"
## [1] "Para p =  9 , o erro para a execução  3 foi:  0.24"
## [1] "Para p =  9 , a acurácia para a execução  3 foi:  0.76"
## [1] "_____________________________________________________________________________________________
__"
## [1] "Para p =  9 , o erro para a execução  4 foi:  0.29"
## [1] "Para p =  9 , a acurácia para a execução  4 foi:  0.71"
## [1] "_____________________________________________________________________________________________
__"
## [1] "Para p =  9 , o erro para a execução  5 foi:  0.22"
## [1] "Para p =  9 , a acurácia para a execução  5 foi:  0.78"
## [1] "_____________________________________________________________________________________________
__"
## [1] ">>> Para p =  9 , a média do erro foi:  0.23 +/- 0.04"
## [1] "_____________________________________________________________________________________________
__"
## [1] "Para p =  15 , o erro para a execução  1 foi:  0.12"
## [1] "Para p =  15 , a acurácia para a execução  1 foi:  0.88"
## [1] "_____________________________________________________________________________________________
__"
## [1] "Para p =  15 , o erro para a execução  2 foi:  0.25"
## [1] "Para p =  15 , a acurácia para a execução  2 foi:  0.75"
## [1] "_____________________________________________________________________________________________
__"
## [1] "Para p =  15 , o erro para a execução  3 foi:  0.26"
## [1] "Para p =  15 , a acurácia para a execução  3 foi:  0.74"
## [1] "_____________________________________________________________________________________________
__"
## [1] "Para p =  15 , o erro para a execução  4 foi:  0.3"
## [1] "Para p =  15 , a acurácia para a execução  4 foi:  0.7"
## [1] "_____________________________________________________________________________________________
__"
## [1] "Para p =  15 , o erro para a execução  5 foi:  0.36"
## [1] "Para p =  15 , a acurácia para a execução  5 foi:  0.64"
## [1] "_____________________________________________________________________________________________
__"
## [1] ">>> Para p =  15 , a média do erro foi:  0.26 +/- 0.09"
## [1] "_____________________________________________________________________________________________
__"
```

Discussão:

Com essa lista pôde-se perceber que RSNNS é uma ferramenta eficiente para auxiliar a criação, treinamento e manutenção de redes neurais. Possui também facilidades para manipulação de arquivos de dados, de resultados, etc. O simulador funciona muito bem neste contexto. A criação e edição de redes no simulador é muito simples e rápida, possibilitandotrabalhar com um grande número de arquiteturas distintas.

O SNNS possui um grande número de algoritmos de aprendizagem. Entre esses algoritmos, estão presentes diversas variações de Backpropagation. Durante essa atividade optou-se por utilizar a versão padrão do Backpropagation, “Std_Backpropagation”.

Nos testes realizados, foram testadas 3 funções de ativação diferentes para os neurônio da camada escondida: Act_Logistic, Act_TanH e Act_Signum. Observou-se que no geral, as redes que utilizaram as 2 primeiras funções de ativação tiveram desempenhos semelhantes enquanto a rede que utilizou Act_Signum teve um desempenho pior. Essa discrepância ficou mais clara considerando a primeira base de dados. Com isso, pôde-se observar a importância em se escolher bem a função de ativação a ser utilizada dependendo do tipo de problema que se deseja resolver.

Com relação ao número de neurônios, notou-se que para os três valores de “p” testados o desempenho da rede foi parecido, com resultados suavemente melhores para p=9 e p=15, se comparados com p=3. Contudo, ao tentar extrapolar o valor de p para valores maiores observou-se um desempenho inferior. Com isso foi possível observar a importância de se buscar um número de neurônios grande o suficiente para não ocorrer *underfitting*, mas que também não seja excessivo para não causar um *overfitting*.