UNIVERSIDADE FEDERAL DE MINAS GERAIS

DISCIPLINA: Redes Neurais Artificial
PROFESSORES: Frederico Gualberto Ferreira Coelho
ALUNO: Antonio Carlos da Anunciação – 2018019443

TRABALHO PRATICO 6
Aplicação do Modelo Multilayer Perceptron para Problemas de Regressão:

Exercícios 1: Dada distribuições normais no espaço **R2**, ou seja, duas distribuições com duas variáveis cada, **(X_1, X_2)**, gerando um conjunto de dados com duas classes, caracterizadas como $\mathcal{N}$**([2, 2],** σ **= 0.4)** e $\mathcal{N}$**([4, 4],** σ **= 0.4)**, como pode ser visualizado na **FIG01**. Com o tamanho das amostras ***nc*** igual a 100 para cada classe.

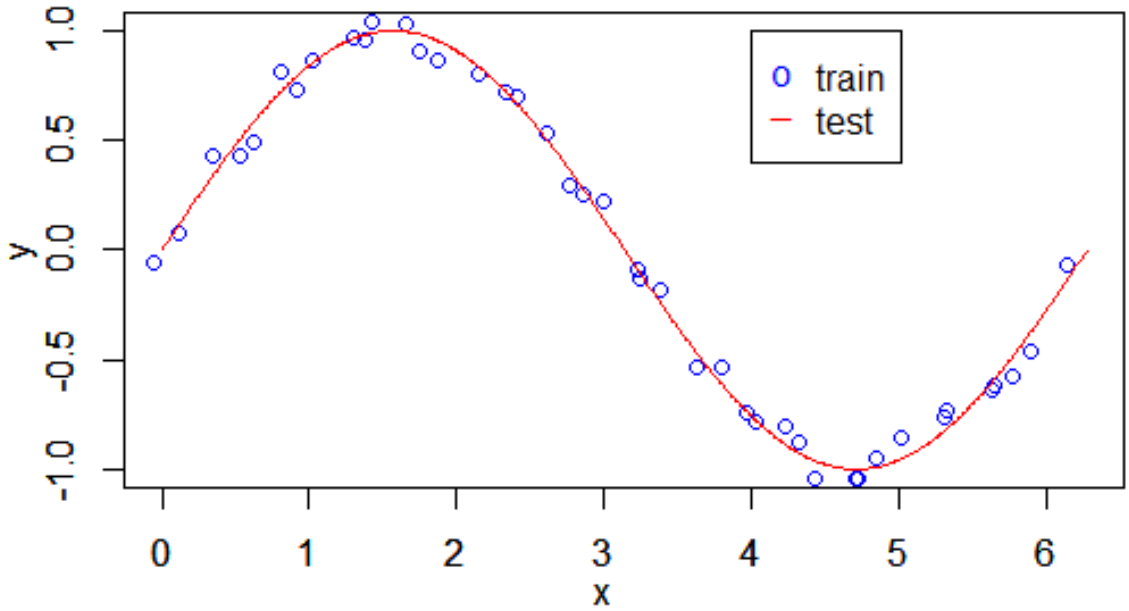

FIG01: Distribuição dos Dados

Neste trabalho iremos criar um modelo para regressão utilizando uma rede de 2 camadas, uma camada de entrada com três neurônios com ***Tangente hiperbólica*** como saída e uma com saída linear. Para tal deverá ser implementado o código do backpropagation para uma rede de duas camadas.

Implementação do Código para Backpropagation na MLP:

```
MLPerceptron <- function(xin, yd, eta, tol, maxepocas, neuronios) {
 dimxin <- dim(xin)
 N <- dimxin[1]
 n <- dimxin[2]
 wo <- matrix( runif( (n+1)*neuronios, -0.5, 0.5), nrow =neuronios, ncol=n+1 )
 wt <- matrix(runif(neuronios+1)-0.5, nrow = 1)
 xin <- cbind(1, xin)
 nepocas <- 0
 eepoca <- tol + 1
 evec <- matrix(0, nrow = 1, ncol = maxepocas)
 while((nepocas < maxepocas) && (eepoca > tol)) {
  erro <- 0
  xseq <- sample(N)

  for(i in 1:N) {
   irand <- xseq[i]

   z1 <- wo %*% xin[irand, ]
   a1 <- rbind(1, tanh(z1))

   z2 <- wt %*% a1
   #yhati <- tanh(z2)
   yhati <- z2

   e <- yd[irand]-yhati
   deltaE2 <- -1*e
   dwt <- eta*deltaE2 %*% t(a1)

   dwo <- matrix(0,dim(wo)[1], dim(wo)[2])
   for(i in 1:dim(wo)[1]) {
    dwo[i,] <- ( eta*deltaE2*wt[,i+1]*( 1/cosh(z1[i,])^2 ) ) %*% t(xin[irand, ])
   }

   wt <- wt - dwt
   wo <- wo - dwo
   erro <- erro + e*e
  }
  nepocas <- nepocas + 1
  evec[nepocas] <- erro/N

  eepoca <- evec[nepocas]
  cat("Erro[", nepocas, "]: ", evec[nepocas], "\n")
 }
 retlist <- list(wo, wt, evec[1:nepocas])
 return(retlist)
}
```

Resultados:

Parâmetros da Rede, MLP:			
Camada de Entrada:	**Neurônio**	**bias**	**w**
	1	-2.8485	0.8965
	2	0.0925	-1.1022
	3	1.4746	-0.3045

Camada de Saída:	**bias**	**w_1**	**w_2**	**w_3**
	-0.2416	-1.7061	-0.9148	-1.5505

Precisão do Modelo:	**Acurácia**	**Desvio Padrão**
	0.9636	0.0507

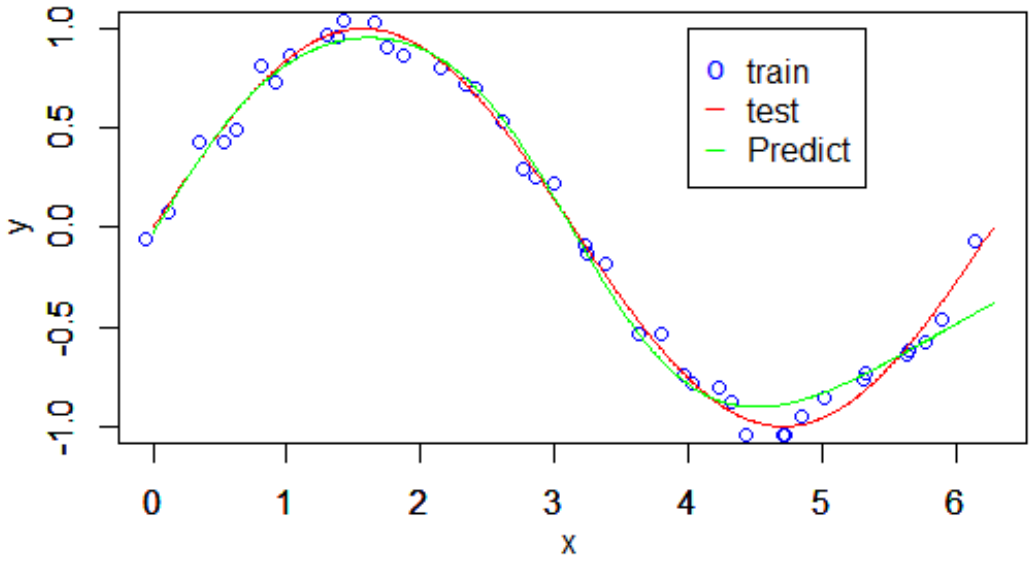

FIG02: Curva Saída do Modelo

Conclusões:

Durante a implementação do modelo foi verificado a influencia dos hiperparametos da rede em seu resultado final, ou seja, a acurácia da rede é fortemente influenciada pelo critérios de paradas e a taxa de aprendizado, altas taxas de aprendizados fez com que a rede convergisse rápido porém se fez necessário aumentar a quantidade de épocas de treinamento, conforme pode ser visto na **FIG03**, abaixo:

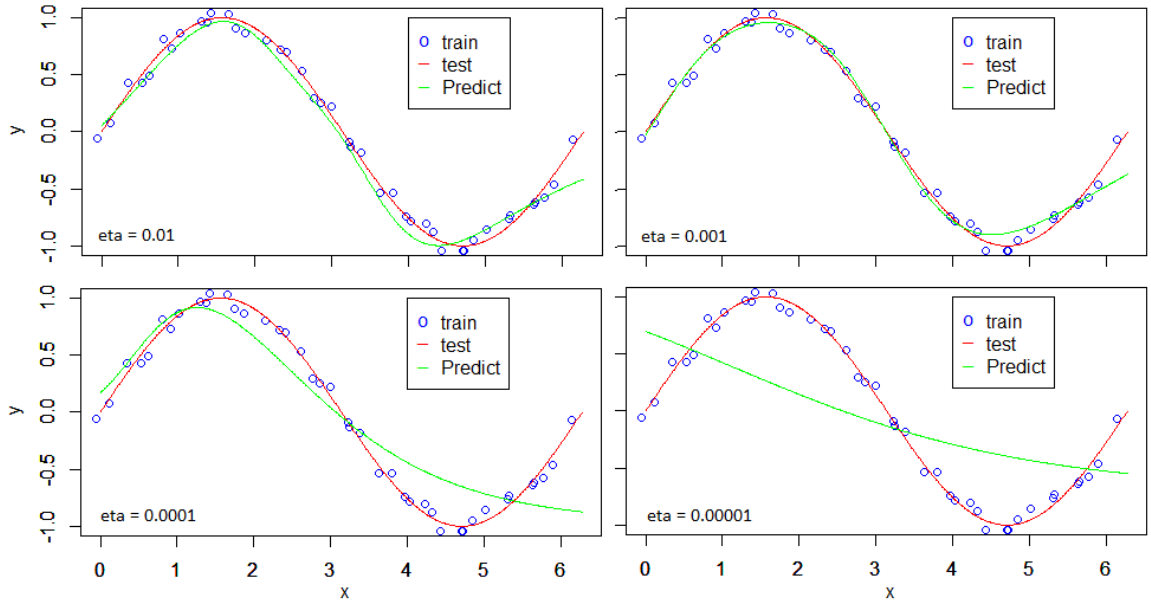

FIG03: Modelo x Taxa de Aprendizado, Maxepocas = 10000

A **FIG02** foi plotada com uma taxa de aprendizado de **eta = 0.00001**, a mesma da ultima figura da **FIG03**, provando a necessidade de um mento na quantidade de épocas de aprendizado, a primeira e segunda figuras da **FIG03** também mostrou que não houve mudanças significativas entre **eta = 0.01** e **0.001** para um **maxepocas = 10000**.

A **FIG04** mostra a influência da quantidade máxima de épocas de treinamento na acurácia do modelo para uma taxa de aprendizado de: **eta = 0.01**.

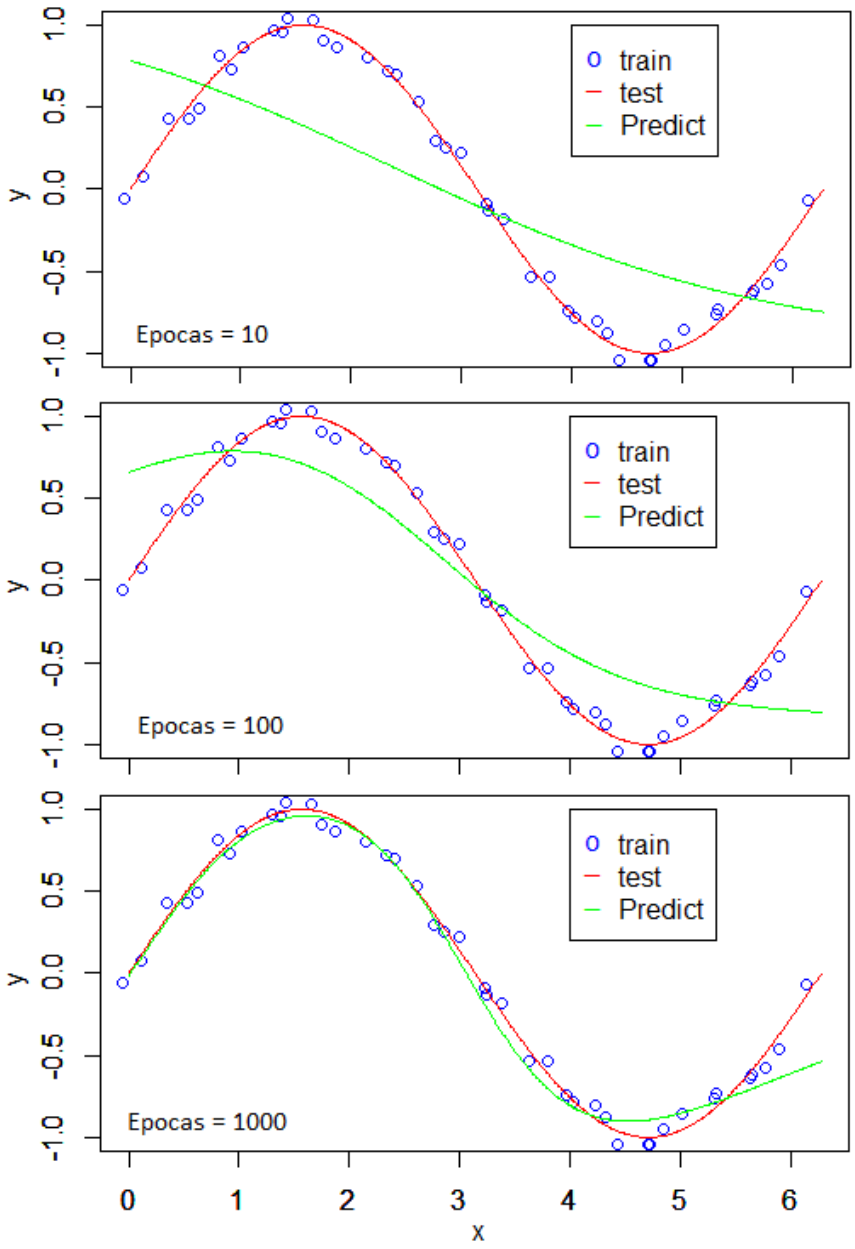

FIG04: Modelo x Época de Treinamento, eta = 0.01

Neste trabalho implementamos uma rede de duas camadas para regressão de uma função, e vimos a influência dos hiperparametros de treinamentos na acurácia da Rede.