

## Trabalho de IA – 2º Bimestre

---

### Implementação da Rede Neural MLP (Multilayer Perceptron) usando Algoritmo Backpropagation

Deverá ser implementada uma ferramenta para realizar o treinamento de uma rede neural usando o algoritmo Backpropagation.

Para a realização de testes para o treinamento e avaliação da rede serão disponibilizados dois arquivos de dados: um para o treinamento da rede e outro para a realização dos testes da rede.

Este arquivo contém 6 entradas e possui 5 classes como saída. Vocês deverão adequar a estrutura da rede para a realização do treinamento usando um neurônio para a identificação de cada Classe.

**ATENÇÃO:** O programa deve aceitar a entrada de arquivos no padrão CSV (separados por vírgula) e identificar o número de neurônios para a camada oculta. Note que a última coluna sempre será a classe. Podem ser usadas outras bases de dados como entrada.

```
X1,X2,X3,X4,X5,X6,classe
1,19,35,28,17,4, CA
4,22,38,32,14,9, CB
-2,16,39,29,21,5, CB
-2,22,33,32,22,1, CE
:
.
```

A ferramenta deve permitir a definição dos critérios de parada para o treinamento:

- **Por Iteração** (número de épocas de treinamento)
- **Por Erro** (até o erro da rede atingir o limiar definido).

**Atenção:** O treinamento se encerra quando um dos dois critérios for atingido.

Outro parâmetro que deve ser configurado é a **Taxa de Aprendizagem (N)**, que pode ser maior do que 0 e menor ou igual a 1.

Além disso, deve possibilitar a escolha da **Função de Saída** (Função de Transferência), a qual poderá ser **Linear**, **Logística** ou **Tangente Hiperbólica** (vide livro de Inteligência Artificial, Almir Artero Olivette, disponível na biblioteca). No final deste documento tem um resumo dessas funções e respectivas derivadas.

A figura a seguir é apenas uma sugestão de interface.

	X1	X2	X3	X4	X5	X6	classe
▶	0,0286	0,1233	0,2987	0,1493	0,0435	0,0411	1
	0,0571	0,1644	0,3377	0,209	0,0109	0,1096	1
	0	0,0822	0,3506	0,1642	0,087	0,0548	1
	0	0,1644	0,2727	0,209	0,0978	0	1
	0,0095	0,0822	0,2727	0,1194	0,0326	0,0822	1
	0,019	0,1096	0,3117	0,1194	0,0761	0,1096	1
	0,0095	0,1644	0,3117	0,1343	0,0761	0,0959	1
	0,0095	0,1918	0,2727	0,1194	0,0761	0,0137	1
	0,0571	0,1096	0,2987	0,1045	0,0217	0,0137	1
	0,019	0,1096	0,2987	0,194	0,0978	0,0274	1
	0,0476	0,1781	0,2597	0,1642	0,0217	0,0411	1
	0,0286	0,1781	0,2987	0,209	0,0435	0,0137	1
	0,0286	0,1781	0,2987	0,1045	0,0326	0,0137	1
	0,0762	0,1644	0,3377	0,194	0,087	0,1096	1
	0	0,1507	0,3117	0,1045	0,087	0,0685	1
	0,0762	0,1918	0,3377	0,1642	0,0326	0,0137	1

Os valores de entrada devem ser normalizados entre 0 e 1. Os valores mínimos e máximos de cada atributo do treinamento devem ser guardados para a normalização dos valores de teste.

$$ValorNormalizado = (ValorAtributo - MenorValorAtrib) / (MaiorValorAtrib - MenorValorAtrib)$$

### Número de Neurônios na Camada Escondida

O número de neurônios nas camadas escondidas, geralmente, é definido EMPIRICAMENTE. Deve-se ter cuidado para não utilizar nem unidades demais, o que pode levar a rede a memorizar os dados de treinamento (*overfitting*), ao invés de extrair as características gerais que permitirão a generalização, nem um número muito pequeno, que pode forçar a rede a gastar tempo em excesso tentando encontrar uma representação ótima.

Devido a estas dificuldades, recomenda-se manter o número de neurônios escondidos baixo, mas não tão baixo quanto o estritamente necessário.

Existem várias propostas de como determinar a quantidade adequada de neurônios nas camadas escondidas de uma rede neural. Uma das mais utilizada é definir o número de neurônios em função da dimensão das camadas de entrada e saída da rede. Pode-se definir o número de neurônios na camada escondida pela média aritmética ou, ainda, pela média geométrica entre tamanho da entrada e da saída da rede.

$$\text{Média Aritmética} = \frac{N^{\circ}\text{Entradas} + N^{\circ}\text{Saídas}}{2}$$

$$\text{Média Geométrica} = \sqrt[2]{N^{\circ}\text{Entradas} \cdot N^{\circ}\text{Saídas}}$$

## Matriz de Confusão

A matriz de confusão quantifica quantos exemplos da base de dados para teste são classificados corretamente pelo modelo construído (representado na diagonal principal) sendo que as demais colunas representam os exemplos de classificados incorretamente pela rede.

Por exemplo: Suponha que sua rede deva classificar **Mamífero**, **Ave** ou **Peixe**. Nas colunas você indica a resposta obtida pela Rede e nas linhas, a classe correta em relação ao exemplo apresentado.

Vejamos:

Matriz de Confusão			
	Mamíferos	Aves	Peixes
Mamíferos	<b>13</b>	2	0
Aves	2	<b>10</b>	3
Peixes	0	3	<b>12</b>

Para interpretar a matriz de confusão temos o seguinte:

Foram apresentados à rede para teste 15 exemplos de cada classe. A quantidade que aparece na diagonal principal (em negrito) representa a quantidade de acertos da rede, enquanto os valores fora da diagonal principal representam os erros ocorridos. No caso dos exemplos de **Mamíferos**, veja que dos 15 apresentados, 13 foram classificados corretamente, enquanto dois foram classificados incorretamente como **Aves**.

## Funções de Transferência (Saída)

Linear:

$$f(net) = \frac{net}{10}$$

$$f'(net) = \frac{1}{10}$$

Logística:

$$f(net) = \frac{1}{1 + e^{-net}}$$

$$f'(net) = f(net) \cdot (1 - f(net))$$

Lembre-se que  **$f(net)$**  é a saída  $i$  para os neurônios da camada oculta ou as saídas da camada de saída.

Dessa forma para calcular o gradiente do erro do neurônio da camada de saída, tendo como base o exemplo dos slides, temos:

$$ErroG = (DesejadoG - iG) \cdot f'(netG)$$

$$ErroG = (Desejado - iG) \cdot (iG \cdot (1 - iG))$$

O mesmo vale para os neurônios da camada oculta

$$ErroC = (ErroG \cdot w_{g,c}) \cdot f'(net)$$

$$ErroC = (ErroG \cdot w_{g,c}) \cdot (iC \cdot (1 - iC))$$

Tangente Hiperbólica (função  $\tanh()$ ):

$$f(net) = \frac{1 - e^{-2 \cdot net}}{1 + e^{-2 \cdot net}}$$

$$f'(net) = 1 - (f(net))^2$$

Dessa forma para calcular o gradiente do erro do neurônio da camada de saída, tendo como base o exemplo dos slides, temos:

$$ErroG = (DesejadoG - iG) \cdot f'(netG)$$

$$ErroG = (Desejado - iG) \cdot (1 - iG^2)$$

O mesmo vale para os neurônios da camada oculta

$$ErroC = (ErroG \cdot w_{g,c}) \cdot f'(net)$$

$$ErroC = (ErroG \cdot w_{g,c}) \cdot (1 - iC^2)$$

### Funções Adicionais:

- 1) Permitir que o usuário opte por carregar o arquivo de dados e dividi-lo em conjunto de Treino e Teste, ou possibilitar a entrada de dois arquivos separados, sendo um para Treino e outro para Teste.
- 2) Identificar durante treinamento, com base na média dos erros da Rede em cada época se o treinamento estabilizou (ocorreu um platô):
  - a. Dar a opção para o usuário interromper, continuar com ou sem a redução da Taxa de Aprendizado. Caso opte pela redução, diminua em 10%.
  - b. Se mesmo assim permanecer o platô, apresentar novamente a opção acima.
  - c. Para identificar o platô, avalia pelo menos 10 épocas e o desvio padrão da média do erro da Rede. Se ele estiver entre 0 e 0,00001, por exemplo, considere o treinamento estagnado.

