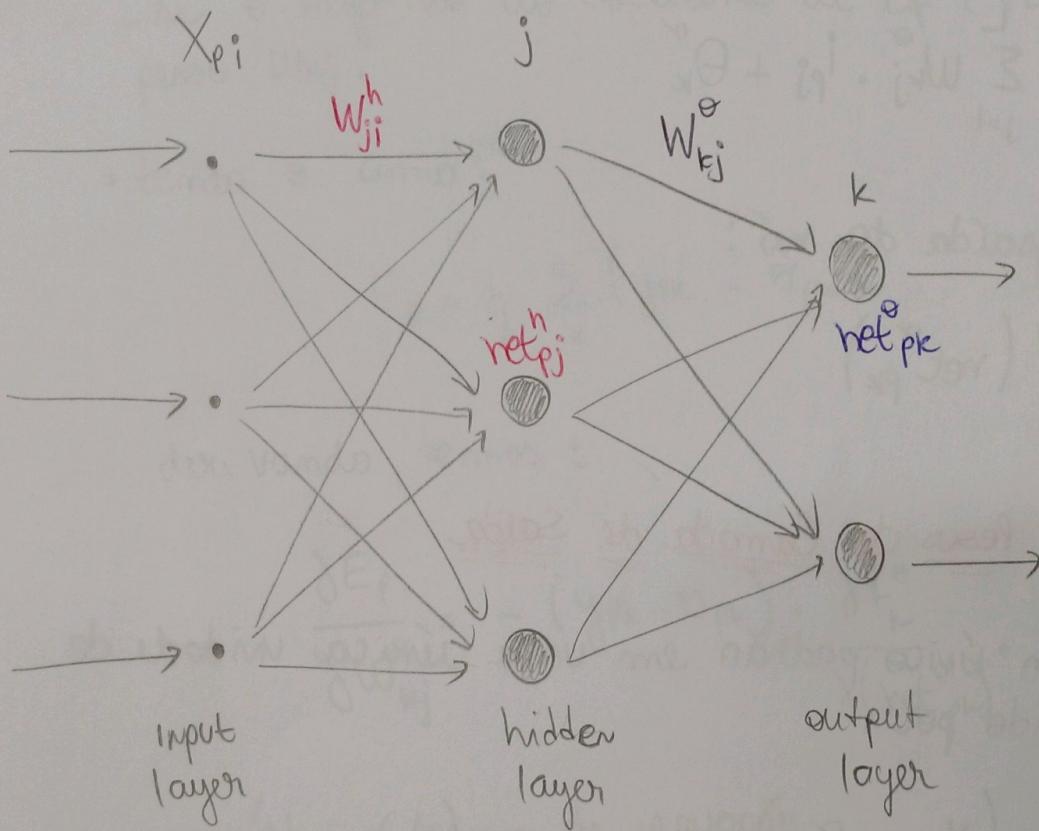


* Multilayer Perceptron

→ modelo



→ Definindo

- net da camada oculta

$$net^h_{pj} = \sum_i^N W_{ji}^h \cdot X_{pi} + \theta_j^h$$

a ativação desse nó é igual a:

$$i_{pj} = f_j^h (net^h_{pj})$$

- as equações para a camada de saída são:

$$\text{net}_{pk}^{\theta} = \sum_{j=1}^L w_{kj}^{\theta} \cdot f_j^h(\text{net}_{pj}^h) + \theta_k^{\theta}$$

como $f_j^h(\text{net}_{pj}^h) = i_{pj}$, logo:

$$\text{net}_{pk}^{\theta} = \sum_{j=1}^L w_{kj}^{\theta} \cdot i_{pj} + \theta_k^{\theta}$$

temos como saída do nó:

$$\theta_{pk} = f_k^{\theta}(\text{net}_{pk}^{\theta})$$

• obs:
 θ indica camada
 h indica camada
 oculta
 θ indica camada
 de saída

* Atualizações dos pesos da Camada de Saída

O erro de um único padrão em uma única unidade de saída é definido por:

$$\delta_{pk} = (y_{pk} - \theta_{pk}) \quad (1)$$

onde:

p refere-se ao padrão "p" do conjunto de treinamento
 y_{pk} é a saída desejada
 θ_{pk} é a saída obtida

- o erro que será minimizado pelo GDR é a soma dos quadrados dos erros para todas as unidades

$$E_p = \frac{1}{2} \sum_{k=1}^M \delta_{pk}^2 \quad (2)$$

o fator $\frac{1}{2}$ é usado para facilitar as derivadas.

- para determinar a direção onde ajustar os pesos, calcula-se o negativo do Gradiante de E_p (∇E_p), com relação aos pesos w_{kj} .
- tendo o como:

$$E_p = \frac{1}{2} \sum_k (y_{pk} - \sigma_{pk})^2 \quad (3)$$

derivando, temos:

$$\frac{\partial E_p}{\partial w_{kj}} = - (y_{pk} - \sigma_{pk}) \cdot \frac{\partial \sigma_k}{\partial (\text{net}_{pk}^\circ)} \cdot \frac{\partial (\text{net}_{pk}^\circ)}{\partial w_{kj}} \quad (4)$$

o último termo da equação

$$\frac{\partial (\text{net}_{pk}^\circ)}{\partial w_{kj}} = \left(\frac{\partial}{\partial w_{kj}} \sum_{j=1}^L w_{kj} \cdot i_{pj} + \theta_{kj}^\circ \right) \quad (5)$$

$$= i_{pj}$$

combinando as equações, temos

$$-\frac{\delta E_p}{\delta w_{kj}^{\sigma}} = (y_{pk} - \sigma_{pk}) \cdot \frac{\delta f_k^{\sigma}}{\delta (\text{net}_{pk}^{\sigma})} \cdot i_{pj} \quad (6)$$

- Sempre assim, os pesos da camada de saída são atualizados de acordo a:

$$w_{kj}^{\sigma}(t+1) = w_{kj}^{\sigma}(t) + \Delta_p w_{kj}^{\sigma}(t) \quad (7)$$

onde:

$$\Delta_p w_{kj}^{\sigma} = \eta (y_{pk} - \sigma_{pk}) \cdot f_k^{\sigma'}(\text{net}_{pk}^{\sigma}) \cdot i_{pj} \quad (8)$$

e

- η é uma taxa de aprendizado, positiva, e comumente menor que 1.

- Deixando em uma única fórmula

$$w_{kj}^{\sigma}(t+1) = w_{kj}^{\sigma}(t) + \eta (y_{pk} - \sigma_{pk}) \cdot f_k^{\sigma'}(\text{net}_{pk}^{\sigma}) \cdot i_{pj} \quad (9)$$

onde:

- w_{kj}^{σ} : são os pesos da camada de saída
- y_{pk} : é a saída esperada para o padrão p
- σ_{pk} : é a saída obtida para o padrão p
- i_{pj} : é a saída de ativação da unidade j da camada oculta
- f_k^{σ} : é a derivada da função de ativação

- Alguns questões sobre f_k^o . Primeiramente f_k^o deve ser diferenciável, o que elimina o uso de uma função degrau como a usada pelo perceptron simples. Dois exemplos de funções de saída:

$$\begin{aligned} \bullet f_k^o(\text{net}_{kj}^o) &= \text{net}_{kj}^o & \left. \right\} \text{saída linear} \\ \bullet f_k^o(\text{net}_{kj}^o) &= \frac{1}{(1 + e^{-\text{net}_{kj}^o})} & \left. \right\} \text{função sigmoidal, logística} \end{aligned}$$

- a) para o primeiro caso, se $f_k^o(\text{net}_{kj}^o) = \text{net}_{kj}^o$, então $f_k^o = 1$, assim a regra de atualização dos pesos fica como:

$$W_{kj}^o(t+1) = W_{kj}^o(t) + \eta (y_{pk} - \sigma_{pk}) \cdot 1 \cdot i_{pj} \quad (10)$$

- b) para a segunda função,

$$f_k^o = f_k^o(1 - f_k^o) = \sigma_{pk}(1 - \sigma_{pk})$$

$$W_{kj}^o(t+1) = W_{kj}^o(t) + \eta (y_{pk} - \sigma_{pk}) \cdot \sigma_{pk}(1 - \sigma_{pk}) \cdot i_{pj} \quad (11)$$

* Atualizações dos Pesos da Camada Oculta

↳ Repete-se o processo como é realizado na camada de saída, porém um problema surge: não há como saber a saída obtida nestas unidades. Intuitivamente, o erro total, E_p , pode ser relacionado à saída das unidades ocultas:

$$\begin{aligned}
 E_p &= \frac{1}{2} \sum_k (y_{pk} - \sigma_{pk})^2 \\
 &= \frac{1}{2} \sum_k (y_{pk} - f_k^\circ(\text{net}_{pk}^\circ))^2 \\
 &= \frac{1}{2} \sum_k (y_{pk} - f_k^\circ \left(\sum_j w_{kj}^\circ \cdot i_j + \theta_k^\circ \right))^2 \quad (1)
 \end{aligned}$$

Sabendo que i_j depende dos valores dos pesos da camada oculta.

• Dessa maneira, definimos a variação do erro em função da camada escondida:

$$\begin{aligned}
 \frac{\delta E_p}{\delta w_{ji}^h} &= \frac{1}{2} \sum_k \frac{\delta}{\delta w_{ij}^h} (y_{pk} - \sigma_{pk})^2 \quad (2) \\
 &= \frac{1}{2} \cdot 2 - \sum_k (y_{pk} - \sigma_{pk}) \cdot \frac{\delta \sigma_{pk}}{\delta \text{net}_{pk}^\circ} \cdot \frac{\delta \text{net}_{pk}^\circ}{\delta i_{pj}} \cdot \frac{\delta i_{pj}}{\delta \text{net}_{pj}^h} \cdot \frac{\delta \text{net}_{pj}^h}{\delta w_{ji}^h}
 \end{aligned}$$

Resolvendo as equações parciais, sabemos que:

$$\bullet \frac{\partial \text{net}_{pk}^o}{\partial w_{pj}} = w_{kj}^o \quad \text{e} \quad \bullet \frac{\partial \text{net}_{pj}^h}{\partial w_{ij}^h} = x_{pi} \quad (3)$$

Logo:

$$\frac{\partial E_p}{\partial w_{ji}^h} = - \sum_k (y_{pk} - \sigma_{pk}) \cdot \frac{\partial \sigma_{pk}}{\partial \text{net}_{pk}^o} \cdot w_{kj}^o \cdot \frac{\partial w_{pj}}{\partial \text{net}_{pj}^h} \cdot x_{pi} \quad (4)$$

Simplificando:

$$\frac{\partial E_p}{\partial w_{ji}^h} = - \sum_k (y_{pk} - \sigma_{pk}) \cdot f'_k(\text{net}_{pk}^o) \cdot w_{kj}^o \cdot f'_j(\text{net}_{pj}^h) \cdot x_{pi} \quad (5)$$

onde:

- $f'_k(\text{net}_{pk}^o)$ é a derivada da saída da função de ativação da camada de saída pelo net

- $f'_j(\text{net}_{pj}^h)$ é a derivada da saída da função de ativação da camada oculta pelo net

- Podemos então calcular a atualizações dos pesos de neurônios da camada escondida como:

$$\Delta_p w_{ji}^h = \eta f_j^h(\text{net}_{pj}^h) x_{pi} \sum_k (y_{pk} - \sigma_{pk}) f_k^o(\text{net}_{pk}^o) w_{kj}^o \quad (6)$$

de seja,

$$w_{ji}^h(t+1) = w_{ji}^h(t) + \eta f_j^h(\text{net}_{pj}^h) x_{pi} \sum_k (y_{pk} - \sigma_{pk}) f_k^o(\text{net}_{pk}^o) w_{kj}^o \quad (7)$$

- Simplificando:

$$\delta_{pk}^o = (y_{pk} - \sigma_{pk}) f_k^o(\text{net}_{pk}^o) \quad (8)$$

Logo:

$$\Delta_p w_{ji}^h = \eta f_j^h(\text{net}_{pj}^h) x_{pi} \sum_k \delta_{pk}^o \cdot w_{kj}^o \quad (9)$$

os erros conhecidos na camada de saída são propagados para trás para a camada oculta, determinando as mudanças necessárias na camada

$$\delta_{pj}^h = f_j^h \cdot \sum_k \delta_{pk}^o \cdot w_{kj}^o$$

chega-se a uma equação análoga ao ajuste da camada de saída:

$$w_{ji}^h(t+1) = w_{ji}^h + \eta \cdot \delta_{pj}^h \cdot x_{pi} \quad (10)$$

• Backpropagation Summary

↳ passos para treinamento de um único vetor com padrão de entrada:

1. Aplicar o vetor de entrada $x_p = (x_{p1}, x_{p2}, \dots, x_{pN})'$ nas unidades de entrada

2. Calcular os valores de entrada na camada oculta

$$\text{net}_{pj}^h = \sum_{i=1}^N w_{ji}^h x_{pi} + \theta_j^h$$

3. Calcular as saídas da camada oculta

$$o_{pj} = f_j^h(\text{net}_{pj}^h)$$

4. Avance para a camada de saída. Calcular os valores net ^s de entrada para cada unidade

$$\text{net}_{pk}^s = \sum_{j=1}^L w_{kj}^s o_{pj} + \theta_k^s$$

5. Calcular as saídas

$$o_{pk} = f_k^s(\text{net}_{pk}^s)$$

6. Calcular os termos de erro para as unidades de saída

$$\delta_{pk}^s = (y_{pk} - o_{pk}) f_k'(\text{net}_{pk}^s)$$

7. Calcular os termos de erro para as unidades ocultas

$$\delta \text{ta}_{pj}^h = f_j^h(\text{net}_{pj}^h) \sum_k \delta \text{ta}_{pk}^o \cdot w_{kj}^o$$

obs: o erro das unidades da camada oculta é calculado ANTES que os pesos da camada de saída sejam ajustados.

8. Atualiza os pesos da camada de saída

$$w_{kj}^o(t+1) = w_{kj}^o(t) + \eta \delta \text{ta}_{pk}^o \cdot i_{pj}$$

9. Atualiza os pesos da camada de oculta

$$w_{ji}^h(t+1) = w_{ji}^h(t) + \eta \delta \text{ta}_{pj}^h \cdot x_{pi}$$

10. Calcular o erro total

$$E_p = \frac{1}{2} \sum_{k=1}^M (y_{pk} - \theta_{pk})^2$$

pois esta medida indica o quanto bem a rede está aprendendo. Quando o E_p (erro) for menor que o critério de parada depois de varrer todos os exemplos o algoritmo pode parar.