Aprendizagem 2021/22

Homework III – Group 35

I. Pen-and-paper

1) **a)** Do enunciado retiramos os valores de $W^{[1]}$, $b^{[1]}$, $W^{[2]}$, $b^{[2]}$, $W^{[3]}$, $b^{[3]}$. Temos também que a learning rate $\eta = 0.1$, $\mathbf{x}^{[0]} = (1 \quad 1 \quad 1 \quad 1 \quad 1)^T$ e a função de ativação $f(z) = \frac{e^x - e^{-x}}{e^x + e^{-x}} = \tanh(z)$.

$$\mathbf{z}^{[1]} = W^{[1]} \cdot x^{[0]} + b^{[1]} = \begin{pmatrix} 1 & 1 & 1 & 1 & 1 \\ 0 & 0 & 0 & 0 & 0 \\ 1 & 1 & 1 & 1 & 1 \end{pmatrix} \begin{pmatrix} 1 \\ 1 \\ 1 \\ 1 \\ 1 \end{pmatrix} + \begin{pmatrix} 1 \\ 1 \\ 1 \end{pmatrix} = \begin{pmatrix} 6 \\ 1 \\ 6 \end{pmatrix}$$

$$\mathbf{x}^{[1]} = \tanh\left(\begin{pmatrix} 6 \\ 1 \\ 6 \end{pmatrix}\right) = f\left(\begin{pmatrix} 6 \\ 1 \\ 6 \end{pmatrix}\right) = \begin{pmatrix} f(6) \\ f(1) \\ f(6) \end{pmatrix} = \begin{pmatrix} 0.9999 \\ 0.7616 \\ 0.9999 \end{pmatrix}$$

$$\mathbf{z}^{[2]} = W^{[2]} \cdot x^{[1]} + b^{[2]} = \begin{pmatrix} 1 & 1 & 1 \\ 1 & 1 & 1 \end{pmatrix} \begin{pmatrix} 0.9999 \\ 0.7616 \\ 0.9999 \end{pmatrix} + \begin{pmatrix} 1 \\ 1 \end{pmatrix} = \begin{pmatrix} 2f(6) + f(1) + 1 \\ 2f(6) + f(1) + 1 \end{pmatrix} = \begin{pmatrix} 3.7614 \\ 3.7614 \end{pmatrix}$$

$$\mathbf{x}^{[2]} = \tanh\left(\begin{pmatrix} 2f(6) + f(1) + 1 \\ 2f(6) + f(1) + 1 \end{pmatrix}\right) = \begin{pmatrix} f(3.7614) \\ f(3.7614) \end{pmatrix} = \begin{pmatrix} 0.9989 \\ 0.9989 \end{pmatrix}$$

$$\mathbf{z}^{[3]} = W^{[3]} \cdot x^{[2]} + b^{[3]} = \begin{pmatrix} 0 & 0 \\ 0 & 0 \end{pmatrix} \begin{pmatrix} f(2f(6) + f(1) + 1) \\ f(2f(6) + f(1) + 1) \end{pmatrix} + \begin{pmatrix} 0 \\ 0 \end{pmatrix} = \begin{pmatrix} 0 & 0 \\ 0 & 0 \end{pmatrix} \begin{pmatrix} 0.9989 \\ 0.9989 \end{pmatrix} + \begin{pmatrix} 0 \\ 0 \end{pmatrix} = \begin{pmatrix} 0 \\ 0 \end{pmatrix}$$

$$\mathbf{x}^{[3]} = \tanh\left(\begin{pmatrix} 0 \\ 0 \end{pmatrix}\right) = \begin{pmatrix} f(0) \\ f(0) \end{pmatrix} = \begin{pmatrix} 0 \\ 0 \end{pmatrix}$$

Agora queremos fazer a Backwards Phase usando o Erro Quadrático Médio:

$$E\left(x^{[3]}, z\right) = \frac{1}{n} \sum_{i=1}^{n} \left(x_i^{[3]} - z_i\right)^2 = \frac{1}{2}\left(x^{[3]} - z\right)^2$$

Derivadas das funções da nossa rede:

$$\frac{\partial E(x^{[l]}, z)}{\partial x^{[l]}} = x^{[l]} - z$$

$$\frac{\partial x^{[l]}}{\partial z^{[l]}} = 1 - \tanh^2\left(z^{[l]}\right)$$

$$\frac{\partial z^{[l]}}{\partial W^{[l]}} = x^{[l-1]} \qquad \frac{\partial z^{[l]}}{\partial x^{[l-1]}} = W^{[l]} \qquad \frac{\partial z^{[l]}}{\partial b^{[l]}} = 1$$

Para começar a recursão, precisamos do δ da última *layer*:

$$\delta^{[3]} = \frac{\partial E}{\partial x^{[3]}} \circ \frac{\partial x^{[3]}}{\partial z^{[3]}} = \left(x^{[3]} - z\right) \circ \left(1 - \tanh^2\left(z^{[3]}\right)\right)$$
$$= \left(\begin{pmatrix} 0 \\ 0 \end{pmatrix} - \begin{pmatrix} 1 \\ -1 \end{pmatrix}\right) \circ \left(\begin{pmatrix} 1 \\ 1 \end{pmatrix} - \begin{pmatrix} \tanh^2(0) \\ \tanh^2(0) \end{pmatrix}\right) = \begin{pmatrix} -1 \\ 1 \end{pmatrix} \circ \begin{pmatrix} 1 \\ 1 \end{pmatrix}$$
$$= \begin{pmatrix} -1 \\ 1 \end{pmatrix}$$

Agora podemos a recursão para computar o delta das *hidden layers*:

$$\begin{aligned}
\delta^{[2]} &= \frac{\partial z^{[3]}}{\partial x^{[2]}}^T \cdot \delta^{[3]} \circ \frac{\partial x^{[2]}}{\partial z^{[2]}} = \left(W^{[3]}\right)^T \cdot \delta^{[3]} \circ \left(1 - \tanh^2\left(z^{[2]}\right)\right) \\
&= \begin{pmatrix} 0 & 0 \\ 0 & 0 \end{pmatrix} \begin{pmatrix} -1 \\ 0 \end{pmatrix} \circ \left(\begin{pmatrix} 1 \\ 1 \end{pmatrix} - \begin{pmatrix} \tanh^2(0) \\ \tanh^2(0) \end{pmatrix} \right) = \begin{pmatrix} 0 \\ 0 \end{pmatrix} \circ \begin{pmatrix} 1 \\ 1 \end{pmatrix} \\
&= \begin{pmatrix} 0 \\ 0 \end{pmatrix}
\end{aligned}$$

$$\begin{aligned}
\delta^{[1]} &= \frac{\partial z^{[2]}}{\partial x^{[1]}}^T \cdot \delta^{[2]} \circ \frac{\partial x^{[1]}}{\partial z^{[1]}} = \left(W^{[2]}\right)^T \cdot \delta^{[2]} \circ \left(1 - \tanh^2\left(z^{[1]}\right)\right) \\
&= \begin{pmatrix} 1 & 1 \\ 1 & 1 \\ 1 & 1 \end{pmatrix} \begin{pmatrix} 0 \\ 0 \end{pmatrix} \circ \left(\begin{pmatrix} 1 \\ 1 \\ 1 \end{pmatrix} - \begin{pmatrix} \tanh^2(6) \\ \tanh^2(1) \\ \tanh^2(6) \end{pmatrix} \right) = \begin{pmatrix} 0 \\ 0 \\ 0 \end{pmatrix} \circ \begin{pmatrix} 1 - \tanh^2(6) \\ 1 - \tanh^2(1) \\ 1 - \tanh^2(6) \end{pmatrix} \\
&= \begin{pmatrix} 0 \\ 0 \\ 0 \end{pmatrix}
\end{aligned}$$

A última fase consiste no aplicar das atualizações. Começando pela primeira *layer:*

$$\frac{\partial E}{\partial W^{[1]}} = \delta^{[1]} \cdot \frac{\partial z^{[1]}}{\partial W^{[1]}} = \delta^{[1]} \cdot \left(x^{[0]}\right)^T = \begin{pmatrix} 0 \\ 0 \\ 0 \end{pmatrix} \cdot \begin{pmatrix} 1 & 1 & 1 & 1 & 1 \end{pmatrix} = \begin{pmatrix} 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 \end{pmatrix}$$

$$W^{[1]} = W^{[1]} - \eta \frac{\partial E}{\partial W^{[1]}} = \begin{pmatrix} 1 & 1 & 1 & 1 & 1 \\ 0 & 0 & 0 & 0 & 0 \\ 1 & 1 & 1 & 1 & 1 \end{pmatrix} - 0.1 \begin{pmatrix} 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 \end{pmatrix} = \begin{pmatrix} 1 & 1 & 1 & 1 & 1 \\ 0 & 0 & 0 & 0 & 0 \\ 1 & 1 & 1 & 1 & 1 \end{pmatrix}$$

$$\frac{\partial E}{\partial b^{[1]}} = \delta^{[1]} \cdot \frac{\partial z^{[1]}}{\partial b^{[1]}} = \delta^{[1]}$$

$$b^{[1]} = b^{[1]} - \eta \frac{\partial E}{\partial b^{[1]}} = b^{[1]} - \eta \cdot \delta^{[1]} = \begin{pmatrix} 1 \\ 1 \\ 1 \end{pmatrix} - 0.1 \begin{pmatrix} 0 \\ 0 \\ 0 \end{pmatrix} = \begin{pmatrix} 1 \\ 1 \\ 1 \end{pmatrix}$$

$$\frac{\partial E}{\partial W^{[2]}} = \delta^{[2]} \cdot \frac{\partial z^{[2]}}{\partial W^{[2]}} = \delta^{[2]} \cdot \left(x^{[1]}\right)^T = \begin{pmatrix} 0 \\ 0 \end{pmatrix} \cdot \begin{pmatrix} 0.9999 & 0.7616 & 0.9999 \end{pmatrix} = \begin{pmatrix} 0 & 0 & 0 \\ 0 & 0 & 0 \end{pmatrix}$$

$$W^{[2]} = W^{[2]} - \eta \frac{\partial E}{\partial W^{[2]}} = \begin{pmatrix} 1 & 1 & 1 \\ 1 & 1 & 1 \end{pmatrix} - 0.1 \begin{pmatrix} 0 & 0 & 0 \\ 0 & 0 & 0 \end{pmatrix} = \begin{pmatrix} 1 & 1 & 1 \\ 1 & 1 & 1 \end{pmatrix}$$

$$\frac{\partial E}{\partial b^{[2]}} = \delta^{[2]} \cdot \frac{\partial z^{[2]}}{\partial b^{[2]}} = \delta^{[2]}$$

$$b^{[2]} = b^{[2]} - \eta \frac{\partial E}{\partial b^{[2]}} = b^{[2]} - \eta \cdot \delta^{[2]} = \begin{pmatrix} 1 \\ 1 \end{pmatrix} - 0.1 \begin{pmatrix} 0 \\ 0 \end{pmatrix} = \begin{pmatrix} 1 \\ 1 \end{pmatrix}$$

$$\frac{\partial E}{\partial W^{[3]}} = \delta^{[3]} \cdot \frac{\partial z^{[3]}}{\partial W^{[3]}} = \delta^{[3]} \cdot \left(x^{[2]}\right)^T = \begin{pmatrix} -1 \\ 1 \end{pmatrix} \cdot \begin{pmatrix} 0.9989 & 0.9989 \end{pmatrix} = \begin{pmatrix} 0.9989 & 0.9989 \\ -0.9989 & -0.9989 \end{pmatrix}$$

$$W^{[3]} = W^{[3]} - \eta \frac{\partial E}{\partial W^{[3]}} = \begin{pmatrix} 0 & 0 \\ 0 & 0 \end{pmatrix} - 0.1 \begin{pmatrix} 0.9989 & 0.9989 \\ -0.9989 & -0.9989 \end{pmatrix} = \begin{pmatrix} -0.09989 & -0.09989 \\ 0.09989 & 0.09989 \end{pmatrix}$$

$$\frac{\partial E}{\partial b^{[3]}} = \delta^{[3]} \cdot \frac{\partial z^{[3]}}{\partial b^{[3]}} = \delta^{[3]}$$

$$b^{[3]} = b^{[3]} - \eta \frac{\partial E}{\partial b^{[3]}} = b^{[3]} - \eta \cdot \delta^{[3]} = \begin{pmatrix} 0 \\ 0 \end{pmatrix} - 0.1 \begin{pmatrix} -1 \\ 1 \end{pmatrix} = \begin{pmatrix} 0.1 \\ -0.1 \end{pmatrix}$$

b)

$$softmax((z_1 \quad z_2 \quad \dots \quad z_n)^T) = (x_1 \quad x_2 \quad \dots \quad x_n)^T \qquad onde\ x_i = \frac{\exp(z_i)}{\sum_{k=1}^{n} \exp(z_k)}$$

Temos que, segundo a equação (5.57) do livro da cadeira:

$$\frac{\partial x_i}{\partial z_j} = \frac{\partial}{\partial z_j} \frac{\exp(z_i)}{\sum_{k=1}^{n} \exp(z_k)} = \begin{cases} x_i(1 - x_i), & i = j \\ -x_j x_i, & i \neq j \end{cases}$$

Como a função de ativação apenas muda na última layer, da alínea anterior vêm os seguintes resultados:

$$\mathbf{z}^{[1]} = \begin{pmatrix} 6 \\ 1 \\ 6 \end{pmatrix} \qquad \mathbf{x}^{[1]} = \begin{pmatrix} f(6) \\ f(1) \\ f(6) \end{pmatrix} = \begin{pmatrix} 0.9999 \\ 0.7616 \\ 0.9999 \end{pmatrix} \qquad \mathbf{z}^{[2]} = \begin{pmatrix} 2f(6) + f(1) + 1 \\ 2f(6) + f(1) + 1 \end{pmatrix} = \begin{pmatrix} 3.7614 \\ 3.7614 \end{pmatrix}$$

$$\mathbf{x}^{[2]} = \begin{pmatrix} f(3.7614) \\ f(3.7614) \end{pmatrix} = \begin{pmatrix} 0.9989 \\ 0.9989 \end{pmatrix} \qquad \mathbf{z}^{[3]} = \begin{pmatrix} 0 \\ 0 \end{pmatrix}$$

O cálculo do $\mathbf{x}^{[3]}$ difere pelo que este é calculado da seguinte forma:

$$\mathbf{x}^{[3]} = softmax(\mathbf{z}^{[3]}) = softmax\left(\begin{pmatrix} 0 \\ 0 \end{pmatrix}\right) = \begin{pmatrix} 0.5 \\ 0.5 \end{pmatrix}$$

Temos:

$$E(x^{[3]}, z) = -\sum_{k=1}^{n} z_k \cdot \log x_k^{[3]}$$

Assim, o delta da terceira layer vai diferir pelo que teremos que o calcular novamente. Dado um $z_i^{[3]}$ obtemos que:

$$\begin{aligned}
\delta_i^{[3]} = \frac{\partial E(x^{[3]}, z)}{\partial z_i} &= \frac{\partial}{\partial z_i}\left(-\sum_{k=1}^{n} z_k \cdot \log x_k^{[3]}\right) = -\sum_{k=1}^{n} z_k \frac{\partial}{\partial z_i} \log x_k^{[3]} \\
&= -\sum_{k=1}^{n} z_k \frac{1}{x_k^{[3]}} \frac{\partial x_k^{[3]}}{\partial z_i} = -\sum_{k=i} z_k \frac{1}{x_k^{[3]}} \frac{\partial x_k^{[3]}}{\partial z_i} - \sum_{k \neq i} z_k \frac{1}{x_k^{[3]}} \frac{\partial x_k^{[3]}}{\partial z_i} \\
&= -z_i \frac{1}{x_i^{[3]}} \left(x_i^{[3]}\left(1 - x_i^{[3]}\right)\right) - \sum_{k \neq i} z_k \frac{1}{x_k^{[3]}} \left(-x_k^{[3]} x_i^{[3]}\right) \\
&= -z_i\left(1 - x_i^{[3]}\right) + \sum_{k \neq i} z_k x_i^{[3]} \\
&= -z_i + z_i x_i^{[3]} + \sum_{k \neq i} z_k x_i^{[3]} = -z_i + x_i^{[3]}\left(z_i + \sum_{k \neq i} z_k\right) \\
&= -z_i + x_i^{[3]}\left(\sum_{k=1}^{n} z_k\right) \qquad onde\ \left(\sum_{k=1}^{n} z_k\right) \text{ é } igual\ a\ 1\ pois\ z = \begin{pmatrix} 1 \\ 0 \end{pmatrix} \\
&= -z_i + x_i^{[3]} \\
&= x_i^{[3]} - z_i
\end{aligned}$$

As restantes derivadas das funções da nossa rede mantêm-se:

$$\frac{\partial x^{[l]}}{\partial z^{[l]}} = 1 - \tanh^2(z^{[l]}) \qquad \frac{\partial z^{[l]}}{\partial W^{[l]}} = x^{[l-1]} \qquad \frac{\partial z^{[l]}}{\partial x^{[l-1]}} = W^{[l]} \qquad \frac{\partial z^{[l]}}{\partial b^{[l]}} = 1$$

Para começar a recursão, precisamos do δ da última *layer*:

$$\delta^{[3]} = \begin{pmatrix} \delta_1^{[3]} \\ \delta_2^{[3]} \end{pmatrix} = \begin{pmatrix} \dfrac{\partial E\left(x^{[3]}, z\right)}{\partial z_1} \\ \dfrac{\partial E\left(x^{[3]}, z\right)}{\partial z_2} \end{pmatrix}$$

$$= \begin{pmatrix} x_1^{[3]} - z_1 \\ x_2^{[3]} - z_2 \end{pmatrix}$$

$$= \mathbf{x}^{[3]} - \mathbf{z} = \begin{pmatrix} 0.5 \\ 0.5 \end{pmatrix} - \begin{pmatrix} 1 \\ 0 \end{pmatrix} = \begin{pmatrix} -0.5 \\ 0.5 \end{pmatrix}$$

Agora podemos a recursão para computar o delta das *hidden layers*:

$$\delta^{[2]} = \frac{\partial z^{[3]^T}}{\partial x^{[2]}} \cdot \delta^{[3]} \circ \frac{\partial x^{[2]}}{\partial z^{[2]}} = \left(W^{[3]}\right)^T \cdot \delta^{[3]} \circ \left(1 - \tanh^2\left(z^{[2]}\right)\right)$$

$$= \begin{pmatrix} 0 & 0 \\ 0 & 0 \end{pmatrix} \begin{pmatrix} -0.5 \\ 0.5 \end{pmatrix} \circ \left(\begin{pmatrix} 1 \\ 1 \end{pmatrix} - \begin{pmatrix} \tanh^2(0) \\ \tanh^2(0) \end{pmatrix} \right) = \begin{pmatrix} 0 \\ 0 \end{pmatrix}$$

$$\delta^{[1]} = \frac{\partial z^{[2]^T}}{\partial x^{[1]}} \cdot \delta^{[2]} \circ \frac{\partial x^{[1]}}{\partial z^{[1]}} = \left(W^{[2]}\right)^T \cdot \delta^{[2]} \circ \left(1 - \tanh^2\left(z^{[1]}\right)\right)$$

$$= \begin{pmatrix} 1 & 1 \\ 1 & 1 \\ 1 & 1 \end{pmatrix} \begin{pmatrix} 0 \\ 0 \end{pmatrix} \circ \left(\begin{pmatrix} 1 \\ 1 \\ 1 \end{pmatrix} - \begin{pmatrix} \tanh^2(6) \\ \tanh^2(1) \\ \tanh^2(6) \end{pmatrix} \right) = \begin{pmatrix} 0 \\ 0 \\ 0 \end{pmatrix} \circ \begin{pmatrix} 1 - \tanh^2(6) \\ 1 - \tanh^2(1) \\ 1 - \tanh^2(6) \end{pmatrix} = \begin{pmatrix} 0 \\ 0 \\ 0 \end{pmatrix}$$

Passamos ao realizar das atualizações. Começando pela primeira *layer:*

$$\frac{\partial E}{\partial W^{[1]}} = \delta^{[1]} \cdot \frac{\partial z^{[1]}}{\partial W^{[1]}} = \delta^{[1]} \cdot \left(x^{[0]}\right)^T = \begin{pmatrix} 0 \\ 0 \\ 0 \end{pmatrix} \cdot \begin{pmatrix} 1 & 1 & 1 & 1 & 1 \end{pmatrix} = \begin{pmatrix} 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 \end{pmatrix}$$

$$W^{[1]} = W^{[1]} - \eta \frac{\partial E}{\partial W^{[1]}} = \begin{pmatrix} 1 & 1 & 1 & 1 & 1 \\ 0 & 0 & 0 & 0 & 0 \\ 1 & 1 & 1 & 1 & 1 \end{pmatrix} - 0.1 \begin{pmatrix} 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 \end{pmatrix} = \begin{pmatrix} 1 & 1 & 1 & 1 & 1 \\ 0 & 0 & 0 & 0 & 0 \\ 1 & 1 & 1 & 1 & 1 \end{pmatrix}$$

$$\frac{\partial E}{\partial b^{[1]}} = \delta^{[1]} \qquad b^{[1]} = b^{[1]} - \eta \frac{\partial E}{\partial b^{[1]}} = b^{[1]} - \eta \cdot \delta^{[1]} = \begin{pmatrix} 1 \\ 1 \\ 1 \end{pmatrix} - 0.1 \begin{pmatrix} 0 \\ 0 \\ 0 \end{pmatrix} = \begin{pmatrix} 1 \\ 1 \\ 1 \end{pmatrix}$$

$$\frac{\partial E}{\partial W^{[2]}} = \delta^{[2]} \cdot \frac{\partial z^{[2]}}{\partial W^{[2]}} = \delta^{[2]} \cdot \left(x^{[1]}\right)^T = \begin{pmatrix} 0 \\ 0 \end{pmatrix} \cdot \begin{pmatrix} 0.9999 & 0.7616 & 0.9999 \end{pmatrix} = \begin{pmatrix} 0 & 0 & 0 \\ 0 & 0 & 0 \end{pmatrix}$$

$$W^{[2]} = W^{[2]} - \eta \frac{\partial E}{\partial W^{[2]}} = \begin{pmatrix} 1 & 1 & 1 \\ 1 & 1 & 1 \end{pmatrix} - 0.1 \begin{pmatrix} 0 & 0 & 0 \\ 0 & 0 & 0 \end{pmatrix} = \begin{pmatrix} 1 & 1 & 1 \\ 1 & 1 & 1 \end{pmatrix}$$

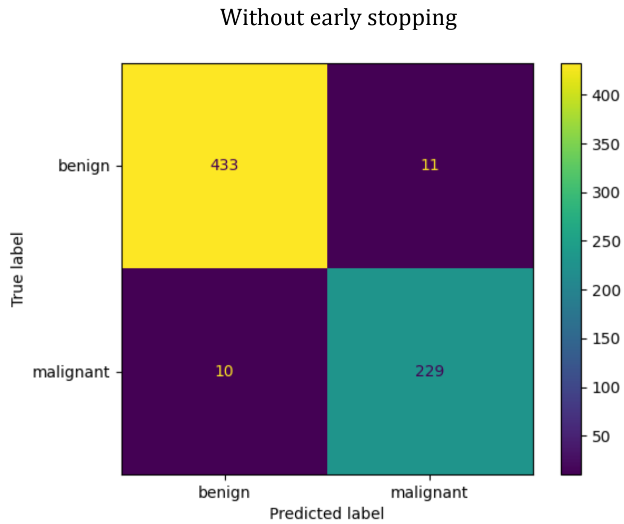
$$\frac{\partial E}{\partial b^{[2]}} = \delta^{[2]} \qquad b^{[2]} = b^{[2]} - \eta \frac{\partial E}{\partial b^{[2]}} = b^{[2]} - \eta \cdot \delta^{[2]} = \begin{pmatrix} 1 \\ 1 \end{pmatrix} - 0.1 \begin{pmatrix} 0 \\ 0 \end{pmatrix} = \begin{pmatrix} 1 \\ 1 \end{pmatrix}$$

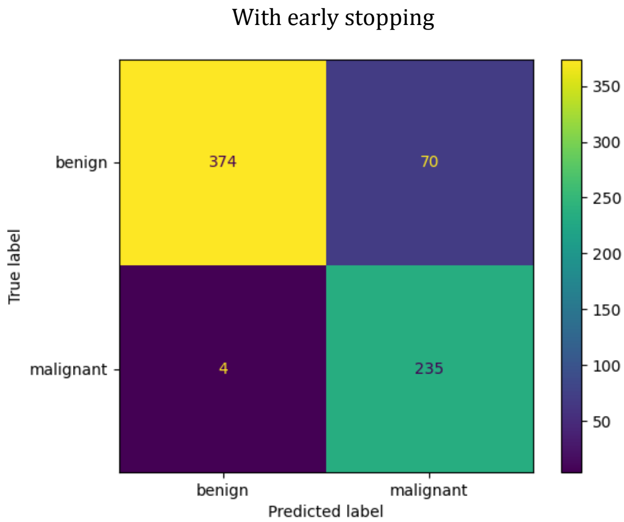
$$\frac{\partial E}{\partial W^{[3]}} = \delta^{[3]} \cdot \frac{\partial z^{[3]}}{\partial W^{[3]}} = \delta^{[3]} \cdot \left(x^{[2]}\right)^T = \begin{pmatrix} -0.5 \\ 0.5 \end{pmatrix} \cdot \begin{pmatrix} 0.9989 & 0.9989 \end{pmatrix} = \begin{pmatrix} -0.49945 & -0.49945 \\ 0.49945 & 0.49945 \end{pmatrix}$$

$$W^{[3]} = W^{[3]} - \eta \frac{\partial E}{\partial W^{[3]}} = \begin{pmatrix} 0 & 0 \\ 0 & 0 \end{pmatrix} - 0.1 \begin{pmatrix} -0.49945 & -0.49945 \\ 0.49945 & 0.49945 \end{pmatrix} = \begin{pmatrix} -0.04995 & -0.04995 \\ 0.04995 & 0.04995 \end{pmatrix}$$

$$\frac{\partial E}{\partial b^{[3]}} = \delta^{[3]} \qquad b^{[3]} = b^{[3]} - \eta \frac{\partial E}{\partial b^{[3]}} = b^{[3]} - \eta \cdot \delta^{[3]} = \begin{pmatrix} 0 \\ 0 \end{pmatrix} - 0.1 \begin{pmatrix} -0.5 \\ 0.5 \end{pmatrix} = \begin{pmatrix} 0.05 \\ -0.05 \end{pmatrix}$$

II. Programming and critical analysis





2) *Early stopping* é uma forma de regularização usada para evitar o *overfitting* ao treinar um modelo usando um método iterativo, por exemplo, o MLP. Estes métodos atualizam o modelo tendo em conta os dados de treino, no entanto, existe um ponto a partir do qual o modelo vai estar demasiado ajustado ao training set ao ponto de não ser eficaz a prever novos casos (*overfitting*). Assim, o *early stopping* é utilizado para prevenir tal acontecimento, terminando o treino do modelo se o *validation score* não estiver a melhorar e limitando o número de *epochs*.

Tendo em conta matrizes de confusão representadas acima (resultado do nosso Código 1, fixando o alpha igual a 1 após testagem com diversos valores), podemos inferir que a *accuracy* do modelo sem *early stopping* (96.93%) é superior ao modelo com *early stopping* (89.17%), o que não vai de encontro ao propósito do early stopping. Isto dá-se provavelmente devido ao facto de o Data Set ser relativamente pequeno e não podermos tirar proveito do *early stopping* dado que este ainda reduzirá mais o modelo, aumentado o risco de *underfitting*.

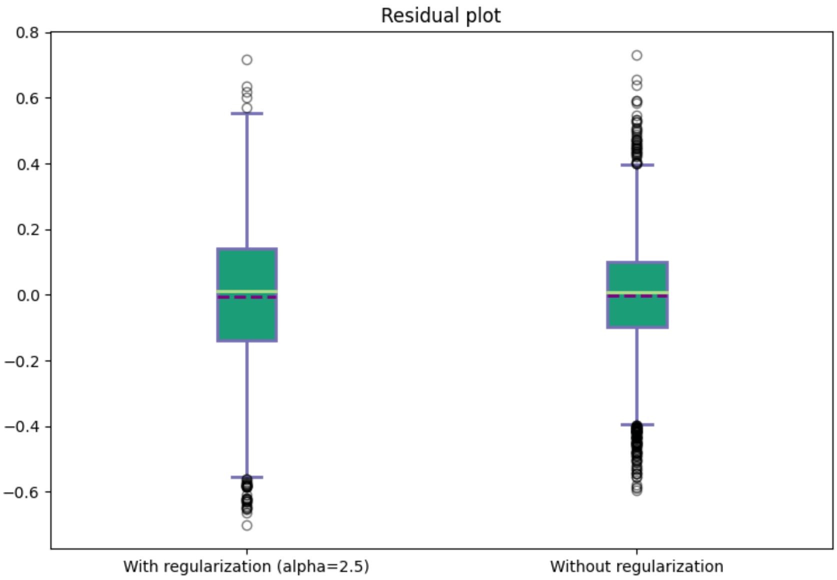
Outro fator é o facto do uso do *K-Fold cross-validation* não ser recomendado juntamente com o *early stopping*. Enquanto o primeiro método pretende estimar o erro de generalização, ajustando o modelo repetidamente e avaliando-o tendo em conta subsets do Data Set, o segundo tem como objetivo controlar o erro de generalização, parando caso este começa a degradar. O problema surge do facto de o *cross-validation* ser utilizado para estimar o erro de generalização enquanto o *early stopping* tenta otimizar o modelo tendo por base o conhecimento do erro.

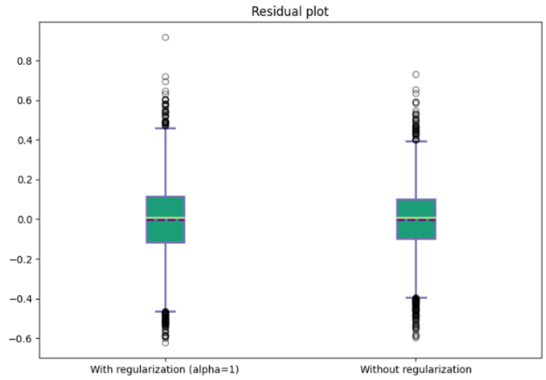
3) A regularização é usada de modo a tentar reduzir o *overfitting* de um modelo ao seu training set e aumentar a sua capacidade de generalização. Neste caso, a regularização penaliza os coeficientes de aprendizagem do modelo, alterando a *Cost Function*.

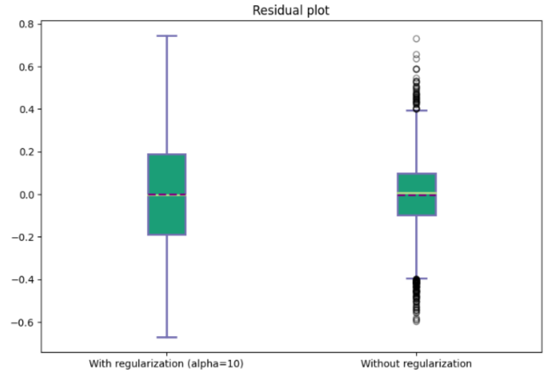
Neste regressor Multi-Layer Perceptron, o parâmetro *alpha* é responsável por definir o nível de regularização *l2*. Fixando um alpha = 2.5 obtivemos os resultados da primeira figura abaixo (Código 2). O gráfico mostra a distribuição dos resíduos usando *boxplots*. Fixámos também *alpha* igual a 1 e igual a 10 para comparar resultados e retirar conclusões.

Observando o erro do regressor MLP, conseguimos identificar algumas estratégias para o melhorar. Primeiramente, podemos aumentar o Data Set. Desta forma teremos mais dados para treinar o modelo e assim conseguimos reduzir o erro. Podemos também diminuir o número de *features* uma vez que isso simplifica o modelo, reduzindo a quantidade de dados necessários para o treinar eficazmente. Podemos ainda aumentar a regularização para reduzir o *overfitting* uma vez que os pesos são penalizados na aprendizagem de forma a que o modelo fique mais equilibrado o que evita uma grande quantidade de *outliers*. É de notar, tendo em conta das figuras representadas abaixo, que um elevado *alpha* também pode gerar *underfitting*.

Por fim, podemos ainda utilizar a estratégia de *early stopping* para limitar o número de *epochs* evitando *overfitting*, isto é, evitar que o modele se ajuste demasiado ao Training Set aumentando o seu erro de generalização para casos exteriores ao mesmo.







III. APPENDIX

(Código 1)

```
import matplotlib.pyplot as plt
import pandas as pd
from scipy.io import arff
from sklearn.metrics import ConfusionMatrixDisplay, confusion_matrix
from sklearn.model_selection import KFold, cross_val_predict
from sklearn.neural_network import MLPClassifier

data = arff.loadarff('breast.w.arff')
df = pd.DataFrame(data[0])
df = df.dropna()
data = df.drop(columns=["Class"]).values
results = df[df.keys()[-1]].astype('string').values

kfold = KFold(n_splits=5, shuffle=True, random_state=0)

class1 = MLPClassifier(hidden_layer_sizes=(3,2), max_iter=2000, activation='relu', alpha=1,
                        random_state=0, early_stopping=False)
class2 = MLPClassifier(hidden_layer_sizes=(3,2), max_iter=2000, activation='relu', alpha=1,
                        random_state=0, early_stopping=True)

y_pred1 = cross_val_predict(class1, data, results, cv=kfold)
y_pred2 = cross_val_predict(class2, data, results, cv=kfold)
cm1 = confusion_matrix(results, y_pred1)        ;       cm2 = confusion_matrix(results, y_pred2)
tn1, fp1, fn1, tp1 = cm1.ravel()                ;       tn2, fp2, fn2, tp2 = cm2.ravel()
```

```
print("Without early stopping:\n{}\nTN:{} FP:{} FN:{} TP:{}\nWith early stopping:\n{}\nTN:{} FP:{}
FN:{} TP:{}"
      .format(cm1, tn1, fp1, fn1, tp1, cm2, tn2, fp2, fn2, tp2))

ConfusionMatrixDisplay.from_predictions(results, y_pred1, display_labels=['benign','malignant'])
ConfusionMatrixDisplay.from_predictions(results, y_pred2, display_labels=['benign','malignant'])
plt.show()
```

(Código 2)

```
import pandas as pd, matplotlib.pyplot as plt
from scipy.io import arff
from sklearn.neural_network import MLPRegressor
from sklearn.model_selection import KFold, cross_val_predict

data = arff.loadarff('kin8nm.arff')
df = pd.DataFrame(data[0])
df = df.dropna()
data = df.drop(columns=["y"]).values
y = df[df.keys()[-1]].values
y_resid1, y_resid2 = [], []

kfold = KFold(n_splits=5, shuffle=True, random_state=0)

regr1 = MLPRegressor(hidden_layer_sizes=(3,2), max_iter=2000, activation='relu', alpha=2.5,
                        random_state=0)
regr2 = MLPRegressor(hidden_layer_sizes=(3,2), max_iter=2000, activation='relu', alpha=0,
                        random_state=0)

y_pred1 = cross_val_predict(regr1, data, y, cv=kfold)
y_pred2 = cross_val_predict(regr2, data, y, cv=kfold)

# residual = actual - expected
y_resid1 = [y-y_pred for y, y_pred in zip(y, y_pred1)]
y_resid2 = [y-y_pred for y, y_pred in zip(y, y_pred2)]

fig = plt.figure(1, figsize=(9,6))
ax = fig.add_subplot(111)
bp = ax.boxplot([y_resid1,y_resid2], showmeans=True, meanline=True, patch_artist=True)
ax.set_xticklabels(['With regularization (alpha=2.5)','Without regularization'])
ax.set_title('Residual plot')

#Customize boxplot
for box in bp['boxes']:
    box.set(color='#7570b3', linewidth=2)
    box.set(facecolor = '#1b9e77' )
for whisker in bp['whiskers']:
    whisker.set(color='#7570b3', linewidth=2)
for cap in bp['caps']:
    cap.set(color='#7570b3', linewidth=2)
for median in bp['medians']:
```

```python
    median.set(color='#b2df8a', linewidth=2)
for flier in bp['fliers']:
    flier.set(marker='o', color='green', alpha=0.5)
for mean in bp['means']:
    mean.set(linestyle='--', linewidth=2, color='purple')

fig.savefig('residual_plot.png', bbox_inches='tight')
plt.show()
```

END