

ALGORITMOS GENÉTICOS E REDES NEURAIS

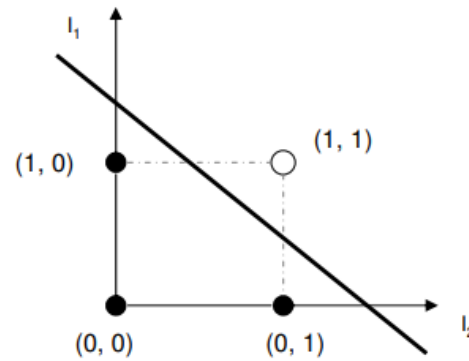
MLP – MULTILAYER PERCEPTRON

Prof. Flávio Belizário da Silva Mota
Universidade do Vale do Sapucaí – UNIVAS
Sistemas de Informação

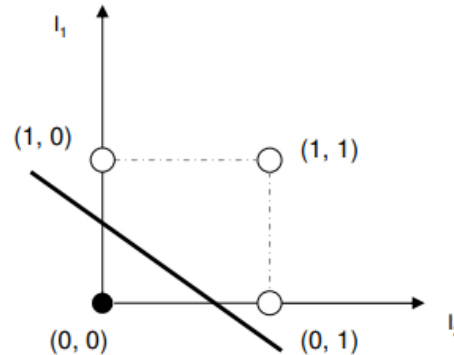
A LIMITAÇÃO DO PERCEPTRON

- O Perceptron foi duramente criticado por Minsky e Papert (1969) pelo fato de não serem capazes de resolver problemas simples (ou não lineares), como a porta XOR.

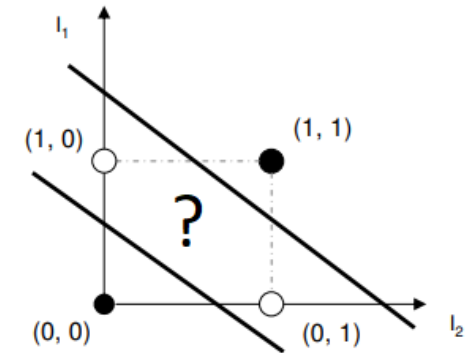
AND		
I_1	I_2	out
0	0	0
0	1	0
1	0	0
1	1	1



OR		
I_1	I_2	out
0	0	0
0	1	1
1	0	1
1	1	1

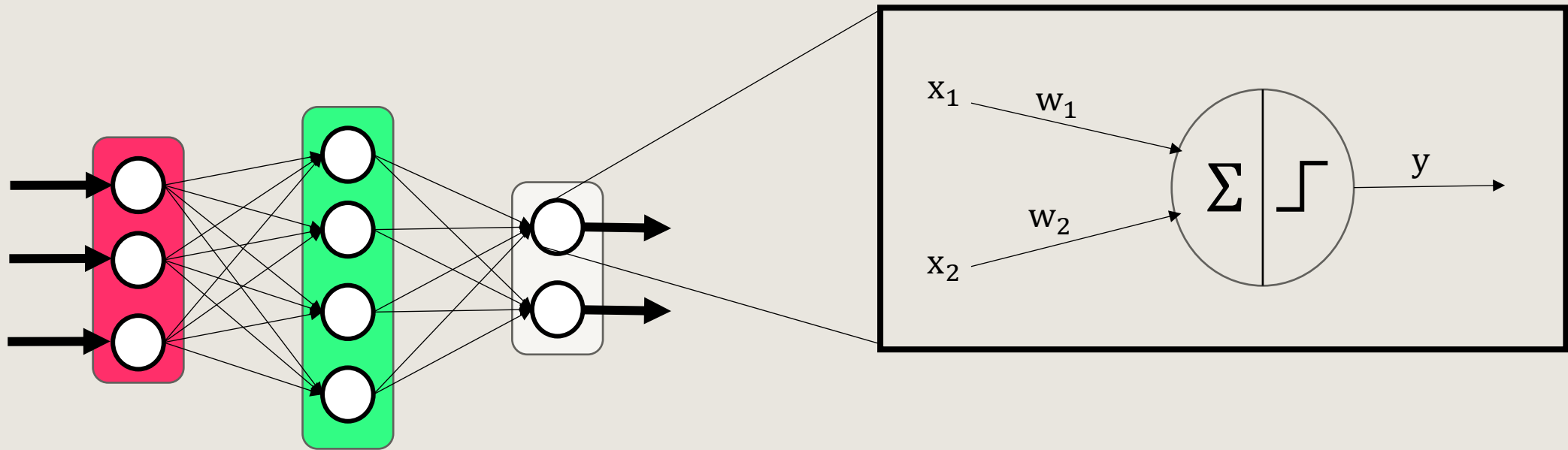


XOR		
I_1	I_2	out
0	0	0
0	1	1
1	0	1
1	1	0



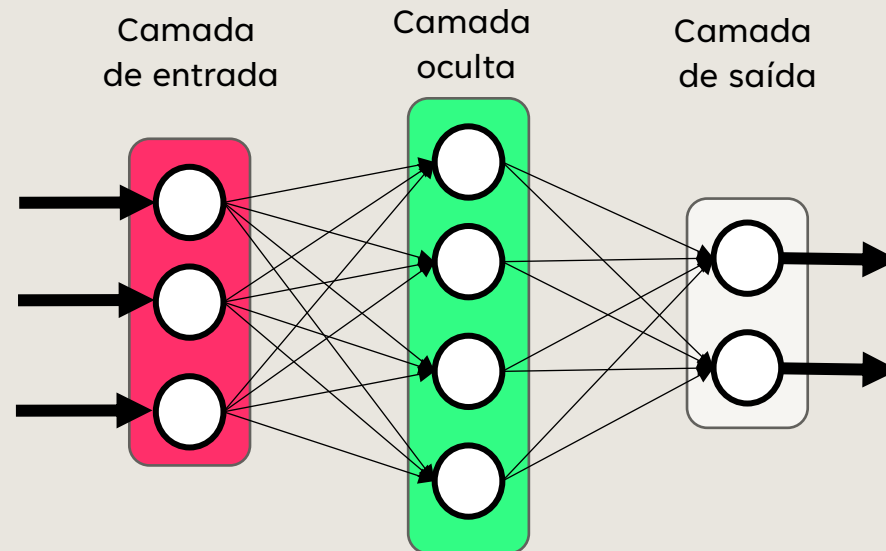
O PERCEPTRON MULTICAMADA (MLP)

- A limitação inicial foi superada quando os pesquisadores perceberam que, assim como os neurônios biológicos se conectam, os neurônios artificiais também poderiam se conectar se vários Perceptrons fossem empilhados. Surge o MLP (Multilayer Perceptron – Perceptron Multicamada).



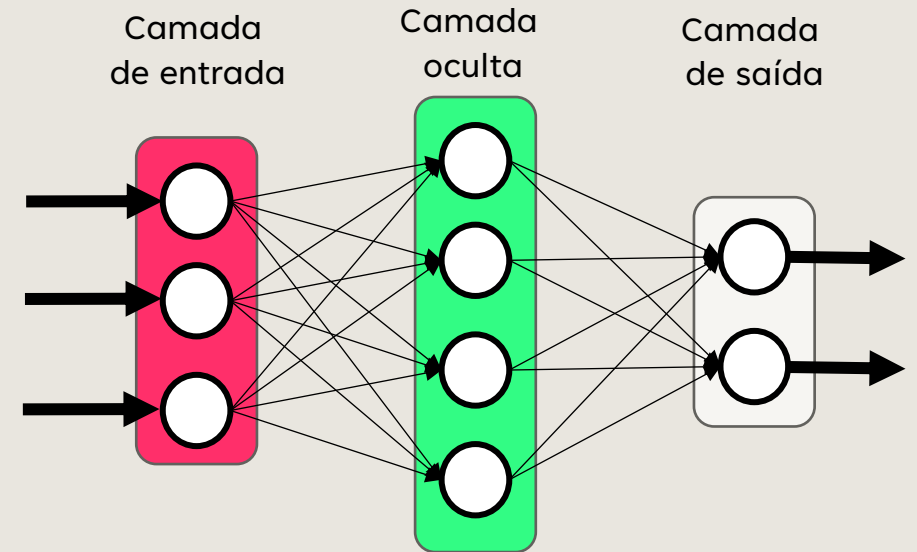
O PERCEPTRON MULTICAMADA (MLP)

- O MLP é composto de uma camada de entrada, uma ou várias camadas ocultas e uma cada de saída. Essas camadas são compostas por vários neurônios que estão totalmente conectados na camada seguinte.



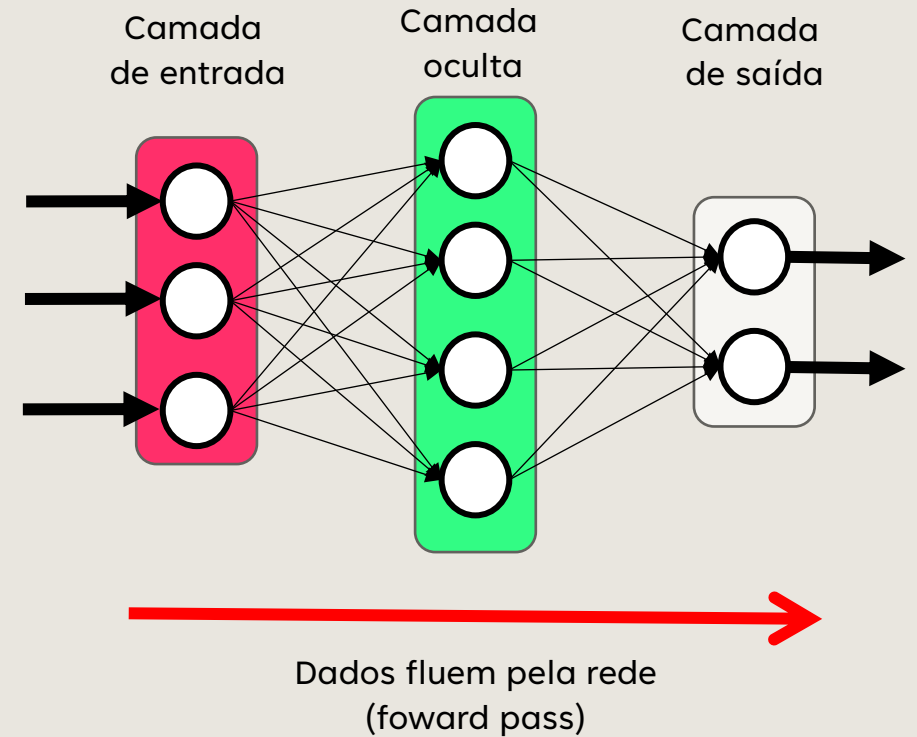
O PERCEPTRON MULTICAMADA (MLP)

- As conexões entre os neurônios das camadas têm pesos associados.
- O processo de treinamento da rede agora se torna mais complexo, pois o ajuste dos pesos não acontece apenas em poucas conexões, mas na rede inteira.
- Pesos dos neurônios das camadas finais influenciam menos no erro do que o das camadas iniciais, então não faz sentido que eles recebam a mesma forma de atualização.



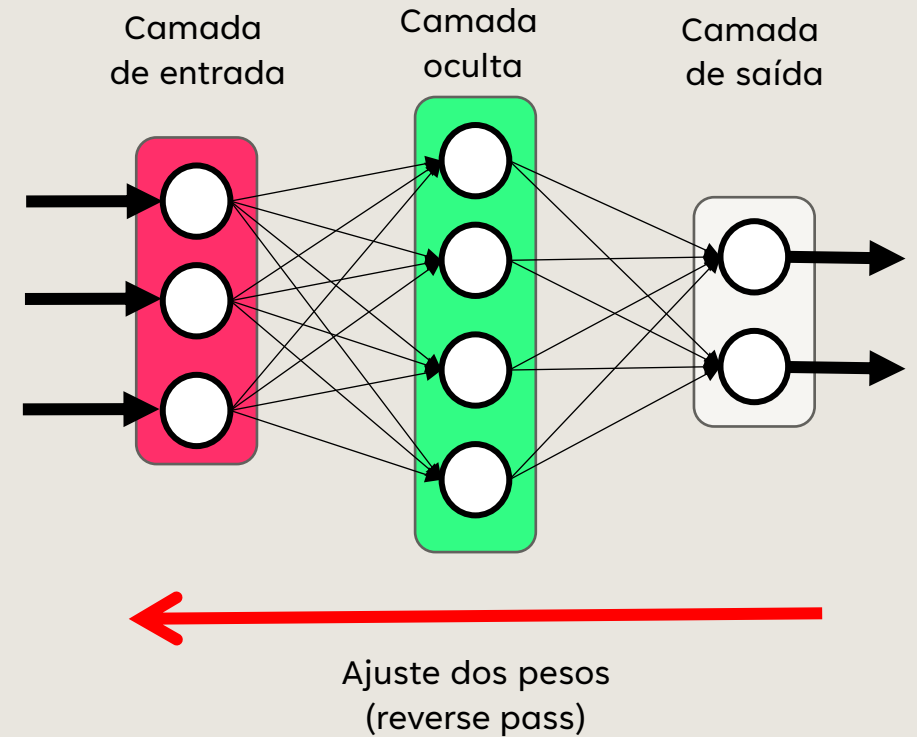
RETROPROPAGAÇÃO (BACKPROPAGATION)

- Para solucionar essa questão o treinamento funciona da seguinte forma:
 - A rede inicia todos os pesos de forma aleatória
 - Para cada amostra do dado, o algoritmo faz a previsão enviando os dados para a frente da rede (forward pass).
 - A classificação (saída) se dá em função da aplicação das somas ponderadas de todos os neurônios envolvidos na rede.



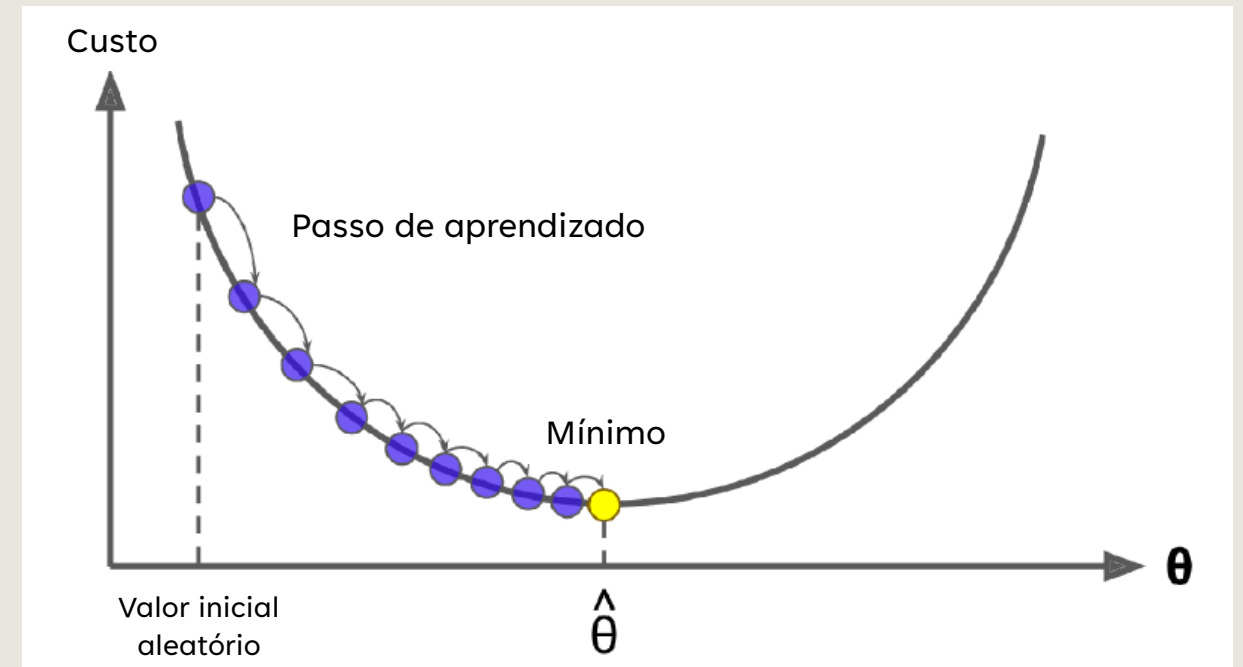
RETROPROPAGAÇÃO (BACKPROPAGATION)

- Para solucionar essa questão o treinamento funciona da seguinte forma:
 - Quando a saída é gerada (classificação), o erro é medido.
 - Inicia-se um processo reverso para medir a contribuição do erro em cada conexão.
 - Quando essa contribuição é calculada, ajusta os pesos da conexão para reduzir o erro.
 - Esses ajustes baseiam-se no Gradiente Descendente.



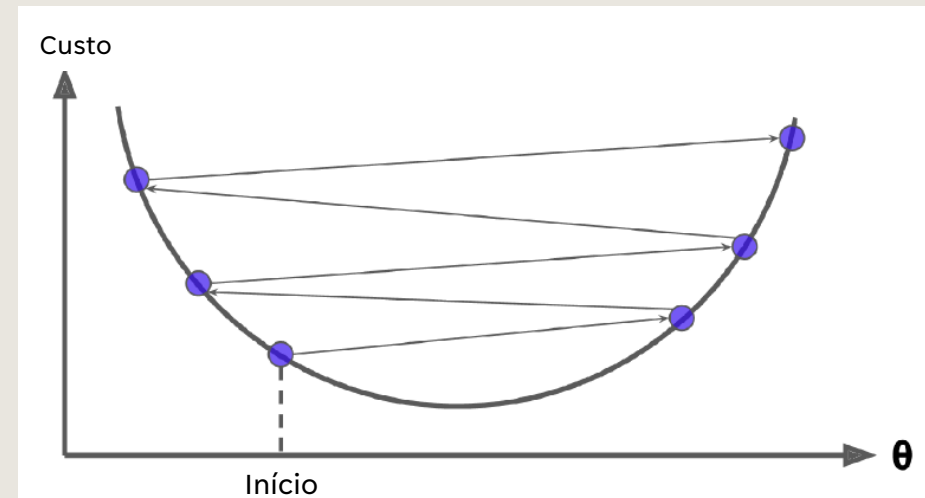
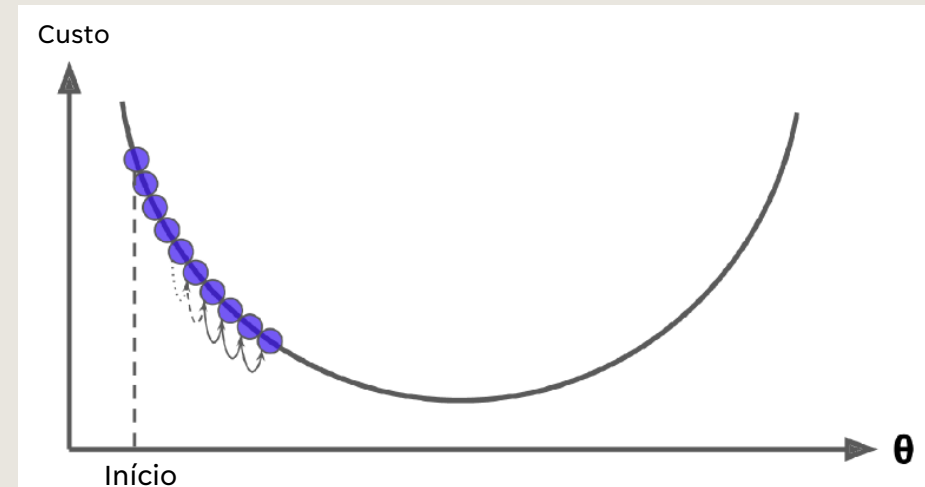
GRADIENTE DESCENDENTE

- A ideia do gradiente descendente é minimizar uma função de custo, no caso da rede, o erro.
- Uma abstração desse método é imaginar que você está perdido em uma montanha na neblina. A única informação que tem é da inclinação do terreno sob seus pés.
- Uma estratégia para para descer a montanha é ir em direção ao lado mais íngreme do terreno.
- O que o gradiente descendente faz é isso: mede qual o gradiente do erro (se aumenta ou diminui) e vai em direção ao que diminui o erro.



GRADIENTE DESCENDENTE

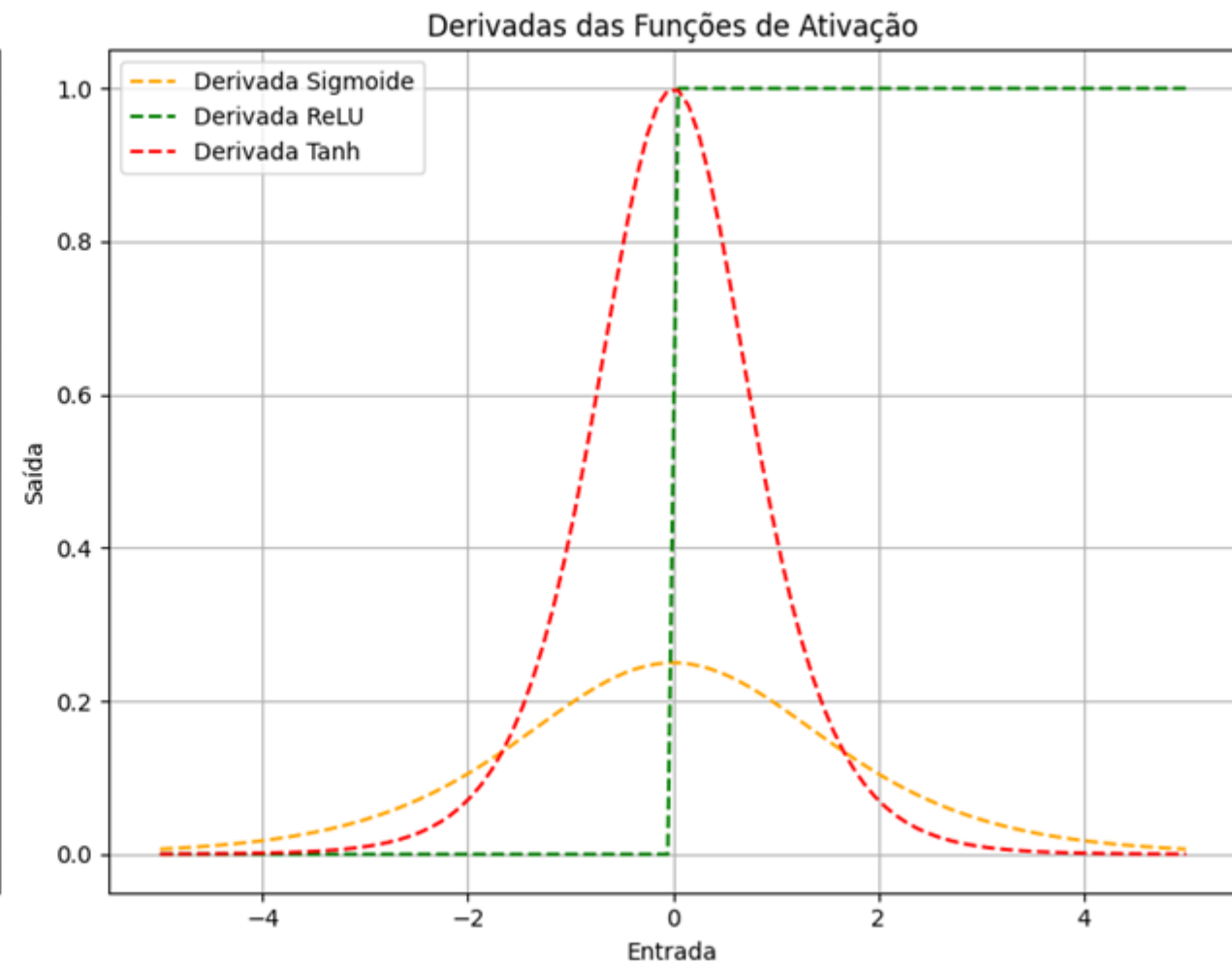
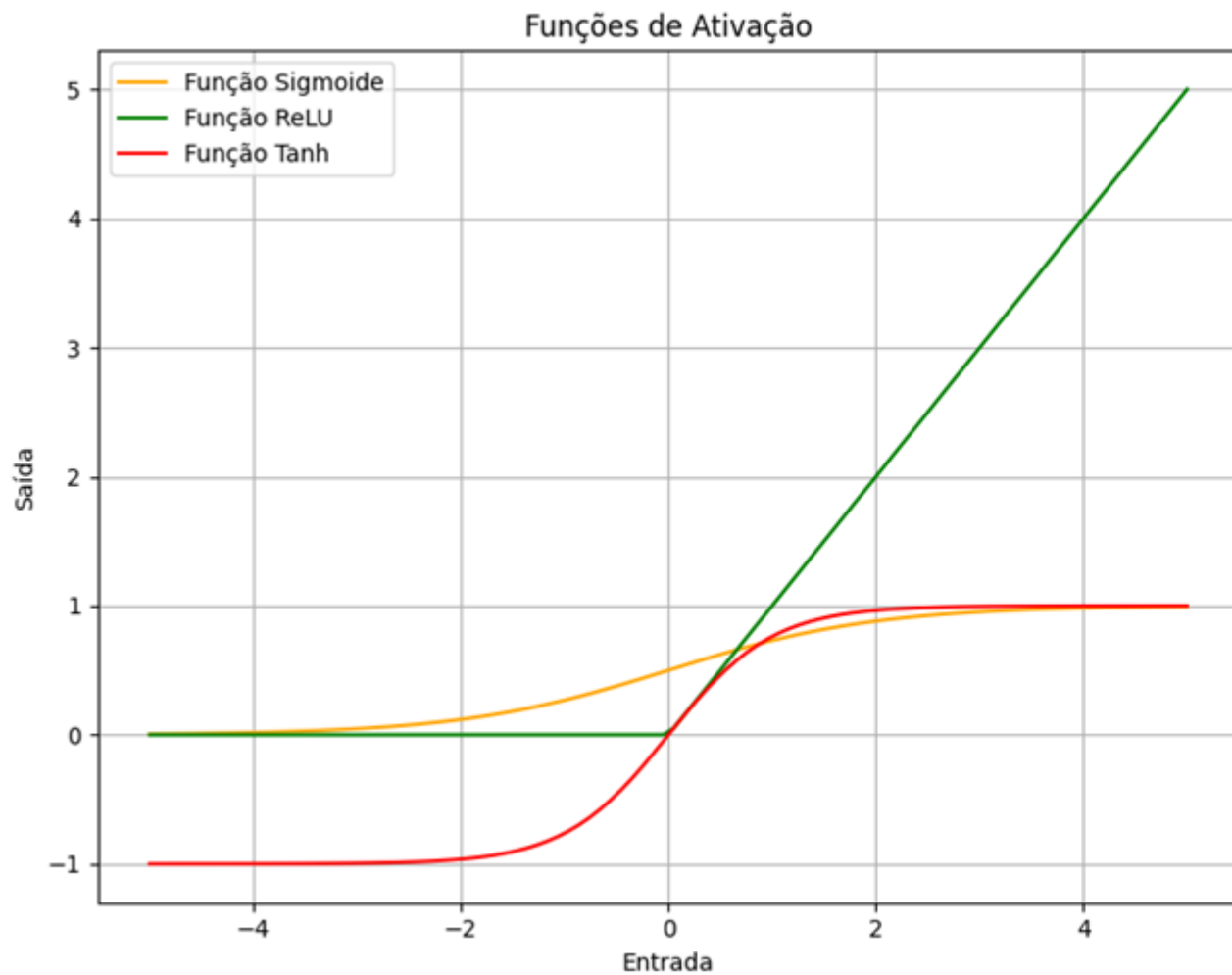
- Configurar a taxa de aprendizado nesse caso é importante, porque é ela quem ajuda o método a se mover no espaço.
- Se ela for muito pequena, o algoritmo vai levar muito tempo para reduzir o erro.
- Se ela for muito alta, o passo pode não ser suficiente para encontrar um valor mínimo e acabar atravessando o “vale”, o que leva a valores cada vez maiores e nenhuma boa solução.



FUNÇÕES DE ATIVAÇÃO

- Esses gradientes estão atrelados às funções de ativação, pois são elas que determinam os valores que podem conter erros.
- Para que o gradiente se mova, ele calcula uma derivada da função de ativação.
- Nesse caso, a função Degrau deixa de fazer sentido, pois tem apenas segmentos planos e qualquer derivação impede o gradiente de se mover.
- Então, as funções mais adotadas são:
 - Sigmoid
 - ReLU
 - Tanh (tangente hiperbólica)

FUNÇÕES DE ATIVAÇÃO



IMPLEMENTANDO UM MLP

- Vamos utilizar a biblioteca TensorFlow para implementar um MLP e treinar alguns dados simples de classificação.

<https://github.com/flavio-mota/si-rna-ag-2025>