Métodos Computacionais em Física: “Aprendizado de máquina”

Aula: Introdução a Redes Neurais.

“Machine Learning”

- Introdução às redes neurais e aprendizado de máquina.
- Exemplo com 1 neurônio: regressão linear.
- Exemplos de classificação: portas lógicas AND, OR.
- Reconhecimendo de padrões: “Modelo de Ising” + Monte Carlo.

Aula: Intro. Redes Neurais - Objetivos

Nesta aula, temos o objetivo de:

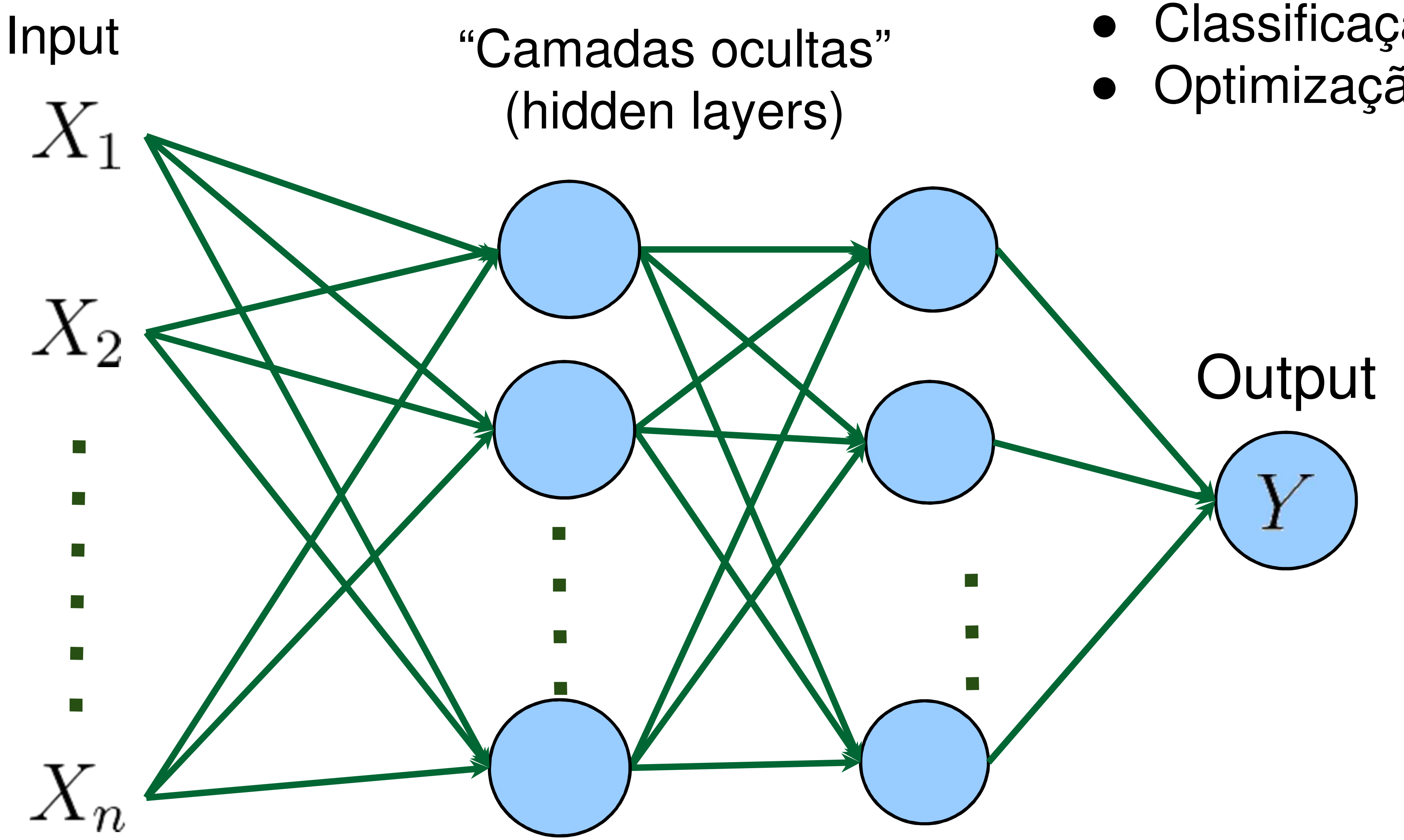
1) Introduzir alguns conceitos básicos de redes neurais: “neurônios”, função de ativação, função custo, classificação x optimização .

2) Entender como a rede “aprende”: minimização da função-custo.

Tarefa: Usar uma “rede neural de um neurônio” para estimar os coeficientes de uma regressão linear de um conjunto de dados.

Tempo aproximado: 10 a 20 min (lembrando que o *debug* é a maior parte disso!).

Uma “rede neural” genérica

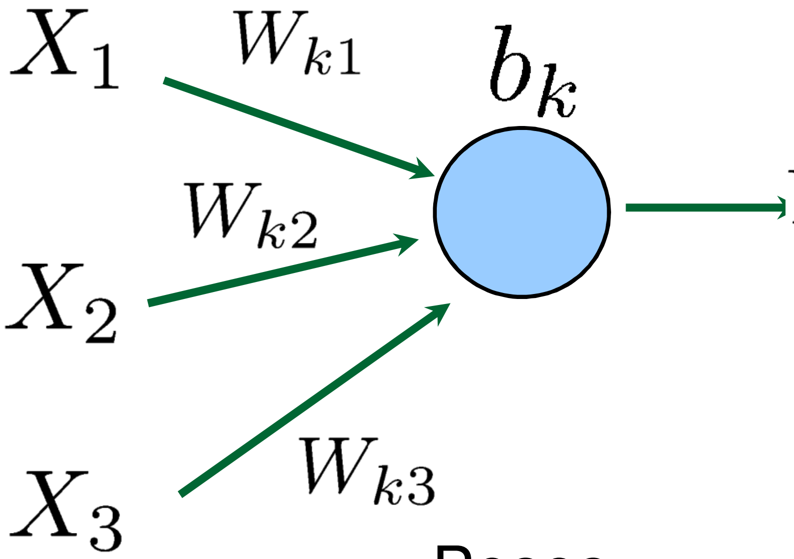


- Classificação
- Optimização

O que é um “neurônio”?

Essencialmente é uma unidade de processamento:

Gera um output que é uma função do vetor de inputs.



$$Y_k = F\left(\sum_i W_{ki} X_i + b_k\right)$$

Output

Input

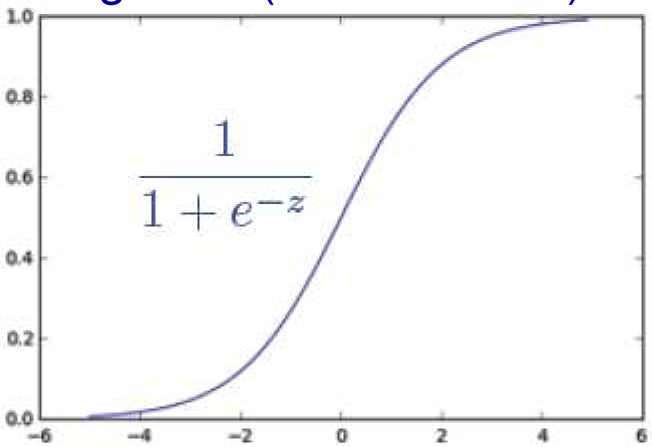
Pesos
(“weights”)

A função F(**W.X** +b) é chamada *função de ativação*.

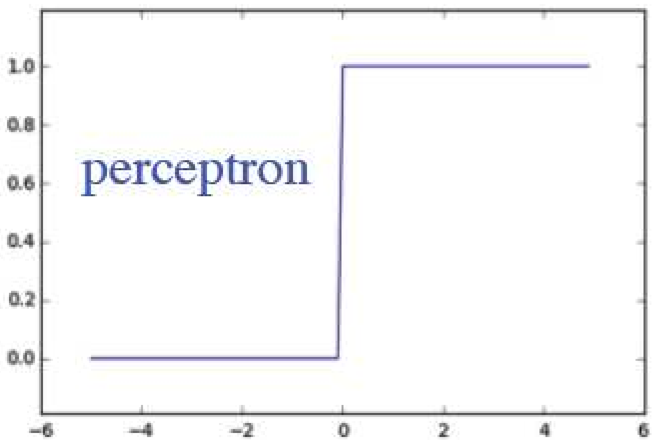
Escolha da função de ativação

Para muitas aplicações, escolhe-se F(z) que retornem outputs entre 0 e 1.
Exemplos:

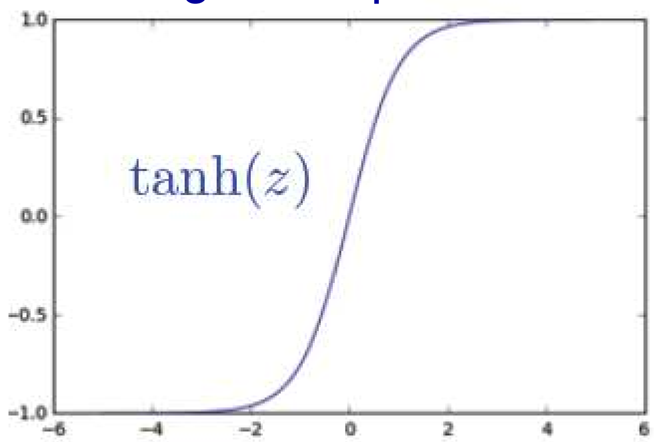
Sigmoid (muito usada)



Função escada

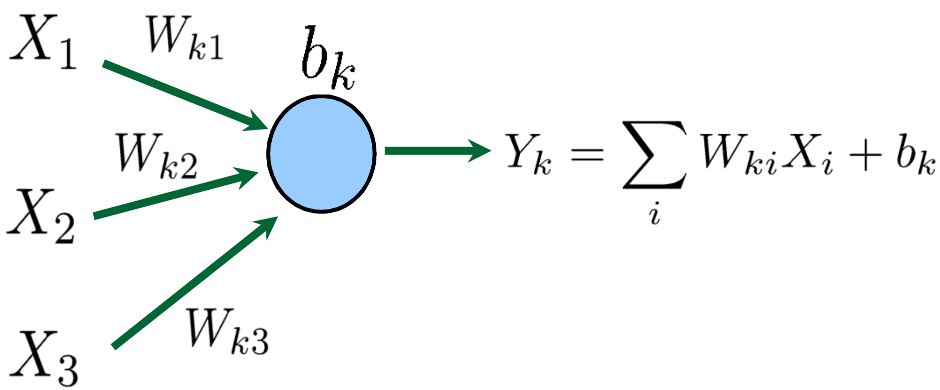


tangente hiperbólica

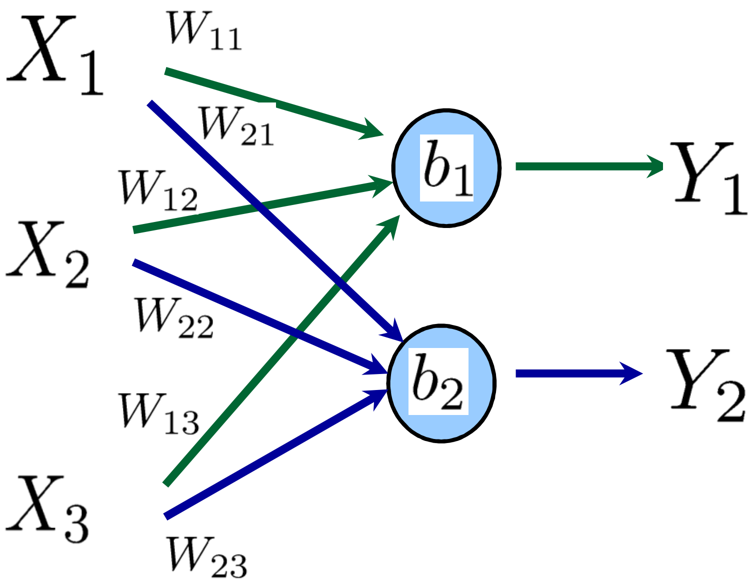


Outras funções são possíveis também.

Para o nosso primeiro exemplo, vamos usar a ***função linear*** *F(z)=z* :



"Aprendizado": Função custo (*cost function*).



- Escolhida a função de ativação, o "aprendizado" consiste na determinação dos parâmetros W_{ki} e b_k de *todos* os neurônios que minimizem uma determinada *função custo*.
- No caso de aprendizado supervisionado, teremos um conjunto de "dados de treinamento" (training set) $\{\mathbf{X}^{(s)}\} = \mathbf{X}^{(1)}, \mathbf{X}^{(2)}, \ldots, \mathbf{X}^{(N)}$
- A rede neural vai "devolver" um conjunto de outputs $\{\mathbf{Y}^{(s)}\} = \mathbf{Y}^{(1)}, \mathbf{Y}^{(2)}, \ldots, \mathbf{Y}^{(N)}$

que podem ser comparados com os valores esperados $\{\bar{\mathbf{y}}^{(s)}\} = \bar{\mathbf{y}}^{(1)}, \bar{\mathbf{y}}^{(2)}, \ldots, \bar{\mathbf{y}}^{(N)}$

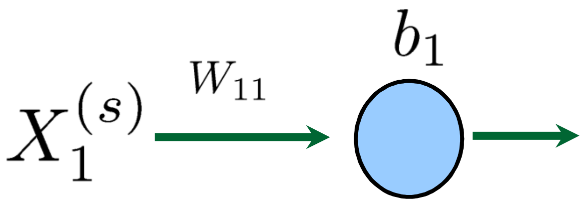
- A função custo $\mathcal{C}(\mathbf{W}, \mathbf{b})$ é uma espécie de "distância" entre o output obtido a partir dos inputs e o output desejado.

$$\mathcal{C}(\mathbf{W}, \mathbf{b}) = \frac{1}{N} \sum_{s=1}^{N} \left(\bar{\mathbf{y}}^{(s)} - \mathbf{Y}^{(s)} \right)^2$$

Exemplo: regressão linear

Para fazer uma regressão linear, precisamos de apenas um único neurônio.

Usaremos uma ***função linear*** *F(z)=z* como função de ativação:



$$Y_1^{(s)} = W_{11} X_1^{(s)} + b_1$$

Digamos que tenhamos um conjunto de N dados ($x^{(s)}$, $y^{(s)}$) a partir dos quais queremos calcular a regressão linear. A função-custo será:

$$\mathcal{C}(W_{11}, b_1) = \frac{1}{N} \sum_{s=1}^{N} \left[y^{(s)} - Y_1^{(s)} \right]^2 = \frac{1}{N} \sum_{s=1}^{N} \left[y^{(s)} - \left(W_{11} x^{(s)} + b_1 \right) \right]^2$$

O problema: encontrar W_{11} e b_1 que minimizem $\mathcal{C}(W_{11}, b_1)$!

Minimizando a função-custo.

$$\mathcal{C}(W_{11}, b_1) = \frac{1}{N}\sum_{s=1}^{N}\left[y^{(s)} - \left(W_{11}x^{(s)} + b_1\right)\right]^2$$

Derivando em relação aos parâmetros livres, temos:

$$\begin{cases} \dfrac{\partial \mathcal{C}}{\partial W_{11}} = -\dfrac{2}{N}\displaystyle\sum_{s=1}^{N}\left[y^{(s)} - \left(W_{11}x^{(s)} + b_1\right)\right]x^{(s)} = -\dfrac{2}{N}\displaystyle\sum_{s=1}^{N}\left[y^{(s)} - Y_1^{(s)}\right]x^{(s)} \\ \dfrac{\partial \mathcal{C}}{\partial b_1} = -\dfrac{2}{N}\displaystyle\sum_{s=1}^{N}\left[y^{(s)} - \left(W_{11}x^{(s)} + b_1\right)\right] = -\dfrac{2}{N}\displaystyle\sum_{s=1}^{N}\left[y^{(s)} - Y_1^{(s)}\right] \end{cases}$$

Método do gradiente descendente (gradient descent) para encontrar o mínimo.

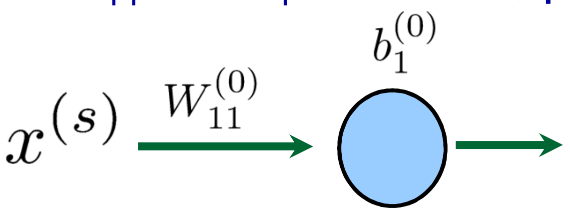
Escolhemos $W^{(0)}_{11}$ e $b^{(0)}_1$ iniciais e atualizamos na forma:

$$\begin{cases} W_{11}^{(1)} = W_{11}^{(0)} + \Delta W_{11} \\ b_1^{(1)} = b_1^{(0)} + \Delta b_1 \end{cases} \quad \text{onde} \quad \begin{cases} \Delta W_{11} = -\eta \frac{\partial \mathcal{C}}{\partial W_{11}} \\ \Delta b_1 = -\eta \frac{\partial \mathcal{C}}{\partial b_1} \end{cases}$$

$\eta \rightarrow$ “hiperparâmetro”

Algorítmo para minimizar a função-custo.

1) Escolhemos $W^{(0)}_{11}$ e $b^{(0)}_1$ iniciais e, para cada input $x^{(s)}$, calculamos $Y^{(s)}_1$



$$Y_1^{(s)} = W_{11}^{(0)} x^{(s)} + b_1^{(0)}$$

2) Calculamos ΔW_{11} e Δb_1 usando os N outputs $Y^{(s)}_1$ e os N dados $(x^{(s)}, y^{(s)})$:

$$\begin{cases} \Delta W_{11} = +\frac{2\eta}{N} \sum_{s=1}^{N} \left[y^{(s)} - Y_1^{(s)} \right] x^{(s)} \\ \Delta b_1 = +\frac{2\eta}{N} \sum_{s=1}^{N} \left[y^{(s)} - Y_1^{(s)} \right] \end{cases}$$

3) Fazermos o update em W_{11} e b_1 e re-calculamos $\mathcal{C}(W_{11}, b_1)$

$$\begin{cases} W_{11}^{(1)} = W_{11}^{(0)} + \Delta W_{11} \\ b_1^{(1)} = b_1^{(0)} + \Delta b_1 \end{cases} \Rightarrow \mathcal{C}^{(1)}(W_{11}, b_1) = \frac{1}{N} \sum_{s=1}^{N} \left[y^{(s)} - \left(W_{11}^{(1)} x^{(s)} + b_1^{(1)} \right) \right]^2$$

4) Repete-se o processo até convergir.

Intro a Redes neurais – Tarefa

Faça a regressão linear dos dados abaixo usando o método do gradiente descendente e um neurônio com função de ativação linear.

s	$x^{(s)}$	$y^{(s)}$
1	3	7.1
2	4	8.3
3	5	10.1
4	8	9.8
5	11	13.7
6	14	15.3
7	15	16.8

- Use o algoritmo anterior de minimização, escolhendo $W^{(0)}_{11}$ e $b^{(0)}_1$ iniciais (qual o critério que você usou?).
- Teste diferentes valores de η (0.00001, 0.001, 0.01) e veja qual funciona melhor.
- Rode o algorítmo por 1000 iterações OU até que

$$\left|\mathcal{C}^{(n+1)}(W_{11}^{(n+1)}, b_1^{(n+1)}) - \mathcal{C}^{(n)}(W_{11}^{(n)}, b_1^{(n)})\right| < 0.0001$$

- Faça um gráfico dos pontos ($x^{(s)}$, $y^{(s)}$) e da reta $Y_1 = W_{11}.x^{(s)} + b_1$ obtida após a convergência.

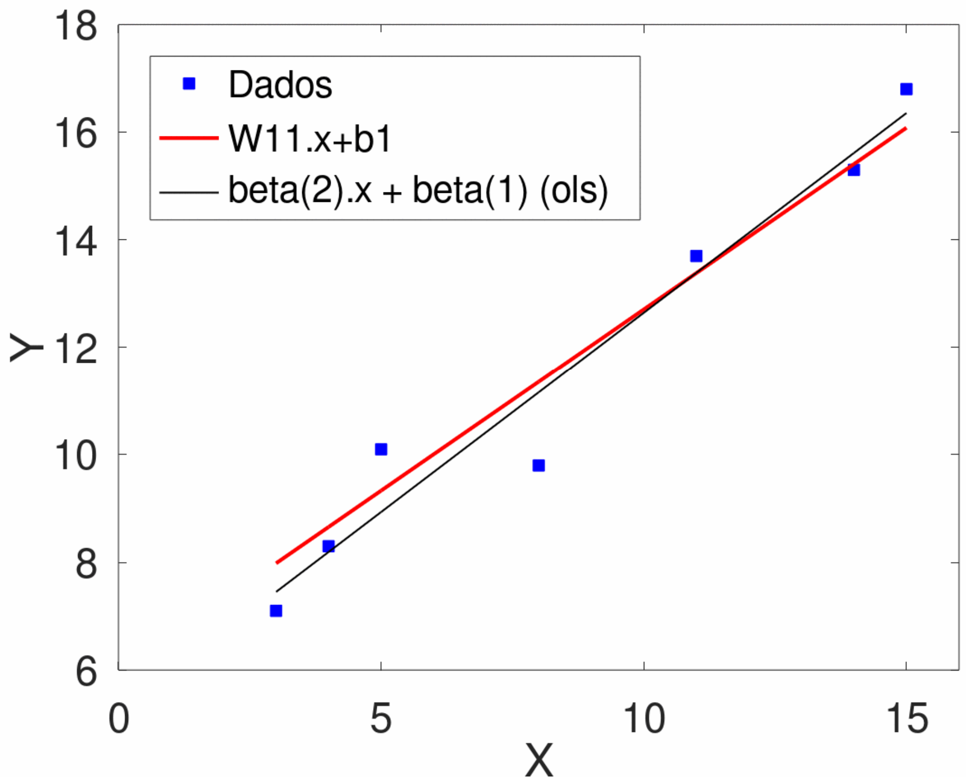
- **Mude os valores de $W^{(0)}_{11}$ e $b^{(0)}_1$ e η e responda:**

Os valores de W_{11} e b_1 que minimizam $\mathcal{C}(W_{11}, b_1)$ dependem destas escolhas?

Intro a Redes neurais – Tarefa - Dicas

s	$x^{(s)}$	$y^{(s)}$
1	3	7.1
2	4	8.3
3	5	10.1
4	8	9.8
5	11	13.7
6	14	15.3
7	15	16.8

- Exemplo ("chutes iniciais" $W^{(0)}_{11} = 1$ e $b^{(0)}_1 = 6$):



- Compare seus coeficientes com o output da função de mínimos quadrados do Octave: beta = ols(ys',MX) **Veja documentação de ols()!!**
- MX é uma matriz com 1 na primeira coluna e xs' na segunda.
- Ou melhor ainda: construa a sua própria rotina de regressão linear!