INSTITUTO FEDERAL
Ceará

Programa de Pós-Graduação
em Ciência da Computação

03 ADAptive LINear Element

APRENDIZAGEM PROFUNDA

PPGCC – 2023.1

Prof. Saulo Oliveira <saulo.oliveira@ifce.edu.br>

ADALINE

- Em 1960 , o Elemento Linear Adaptativo (ADAptive LINear Element — ADALINE) surgiu na literatura quase que simultaneamente com o Perceptron, no trabalho:

  B. Widrow and M. E. Hoff. Adaptive switching circuits. In IRE WESCON Convention Record-Part 4, pages 96-104, 1960.

- A operação é não-linear do tipo degrau para o Perceptron. Enquanto, a operação é puramente linear para o ADALINE.

$$\hat{f}(\mathbf{x}) = \mathbf{w}^{\intercal}\mathbf{x} = u.$$

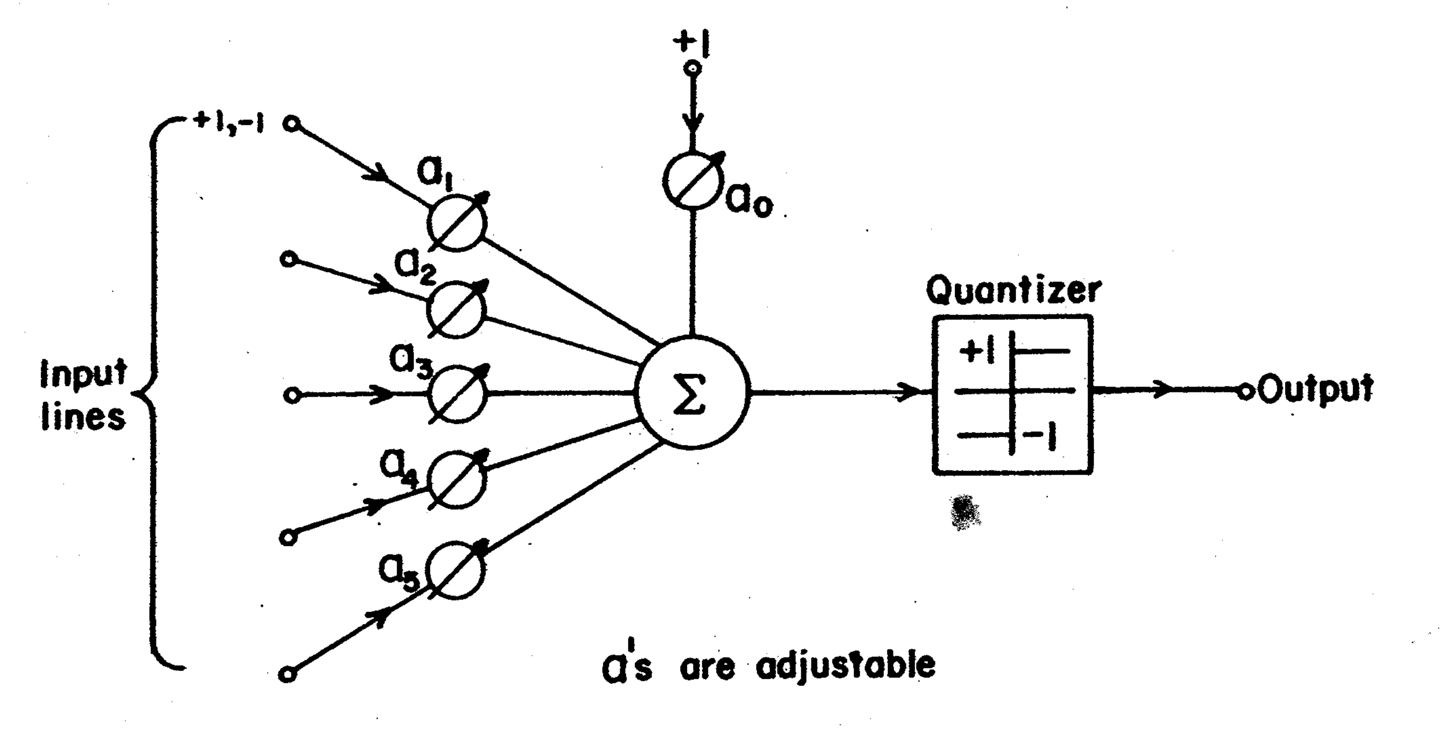
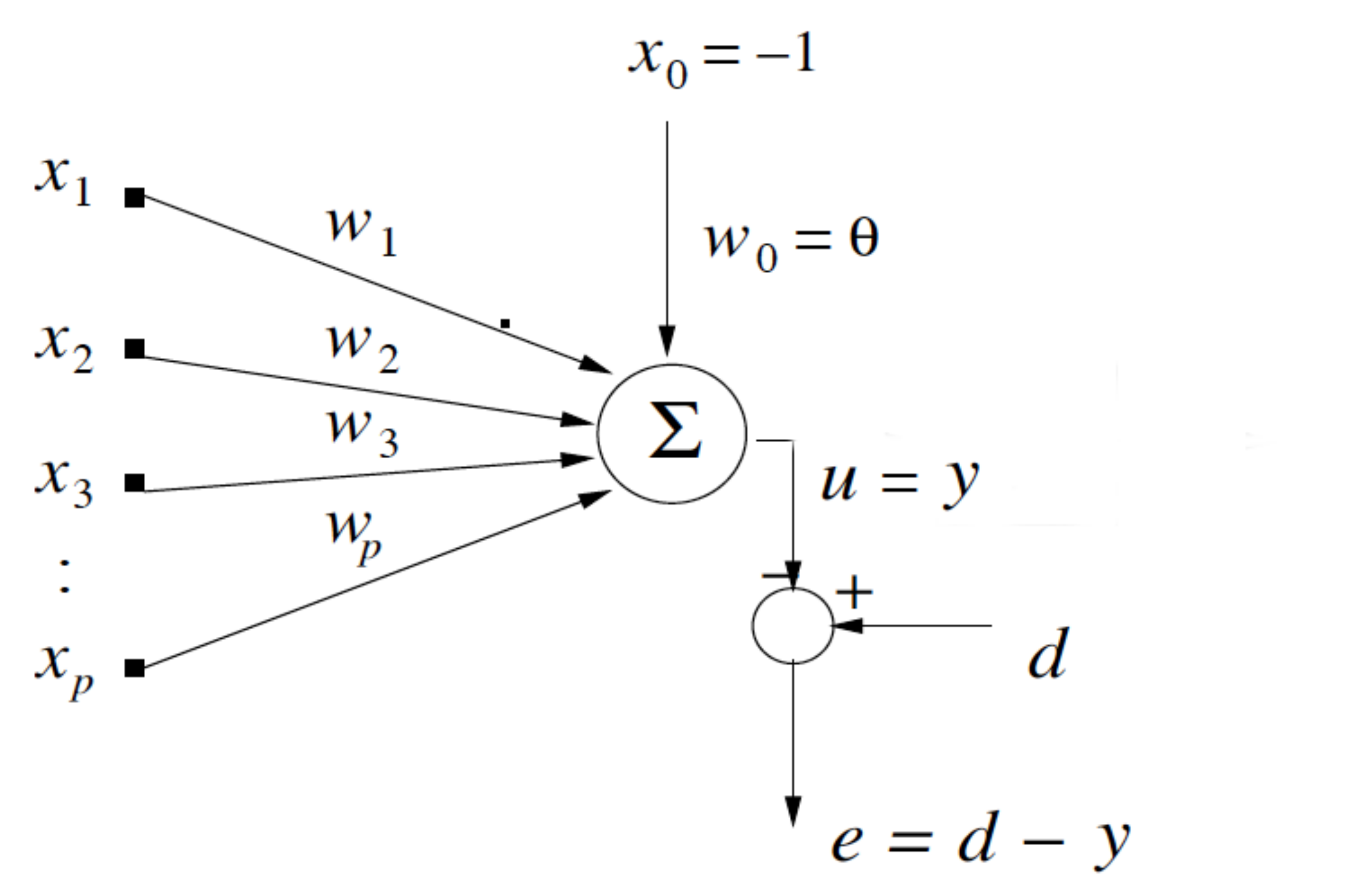


Fig. 1. An adjustable neuron.



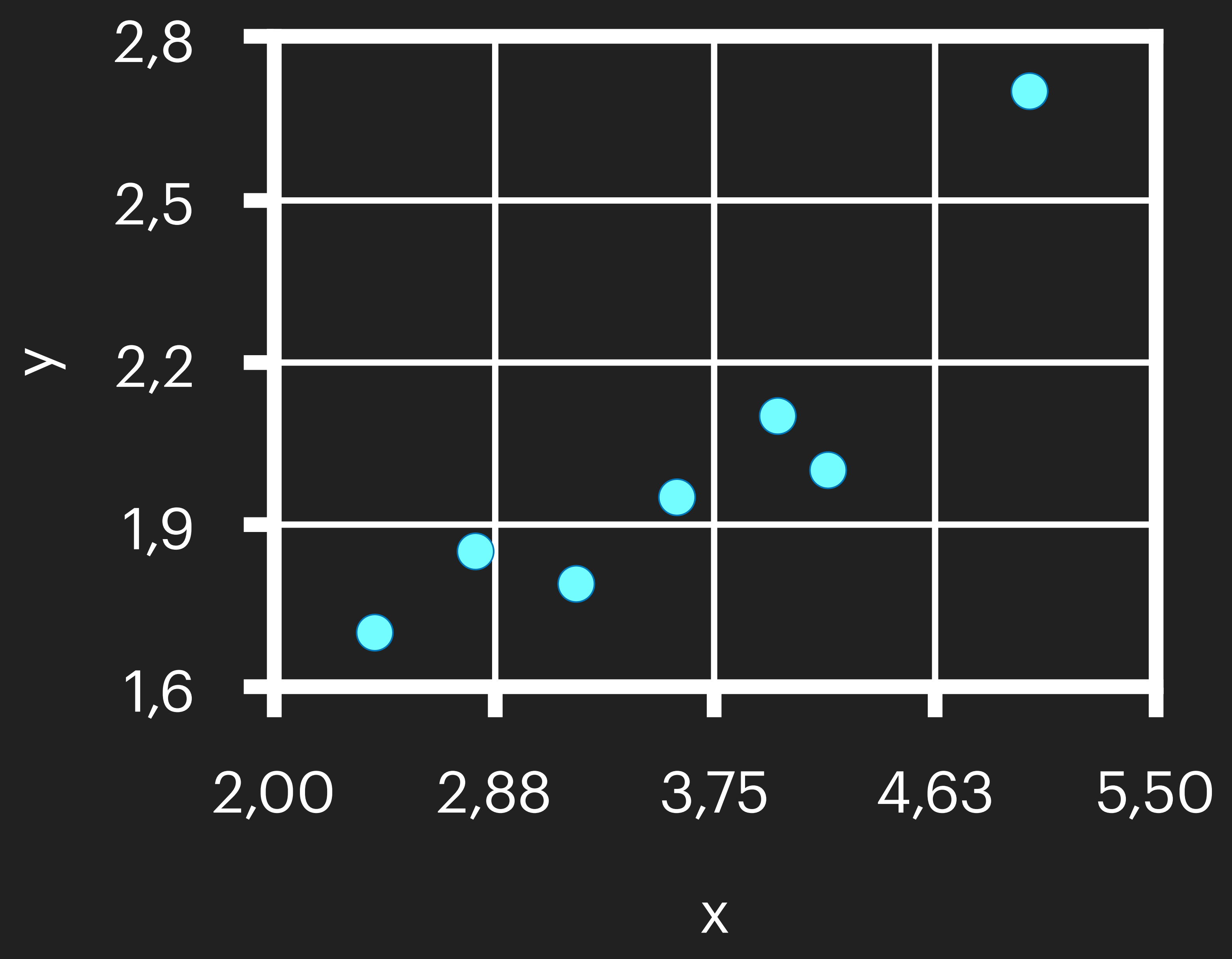
COMO QUE TREINA UM ADALINE?

TREINANDO UM ADALINE

- Premissa básica: achar o vetor de pesos que minimize o TODOS os erros;
- Forma de calcular o erro continua a mesma, i.e., $e = d - y = d - \mathbf{x}^\top \mathbf{w}$.
- Como agregar TODOS os erros em uma função só?

FUNÇÃO DE PERDA/CUSTO/ OBJETIVO

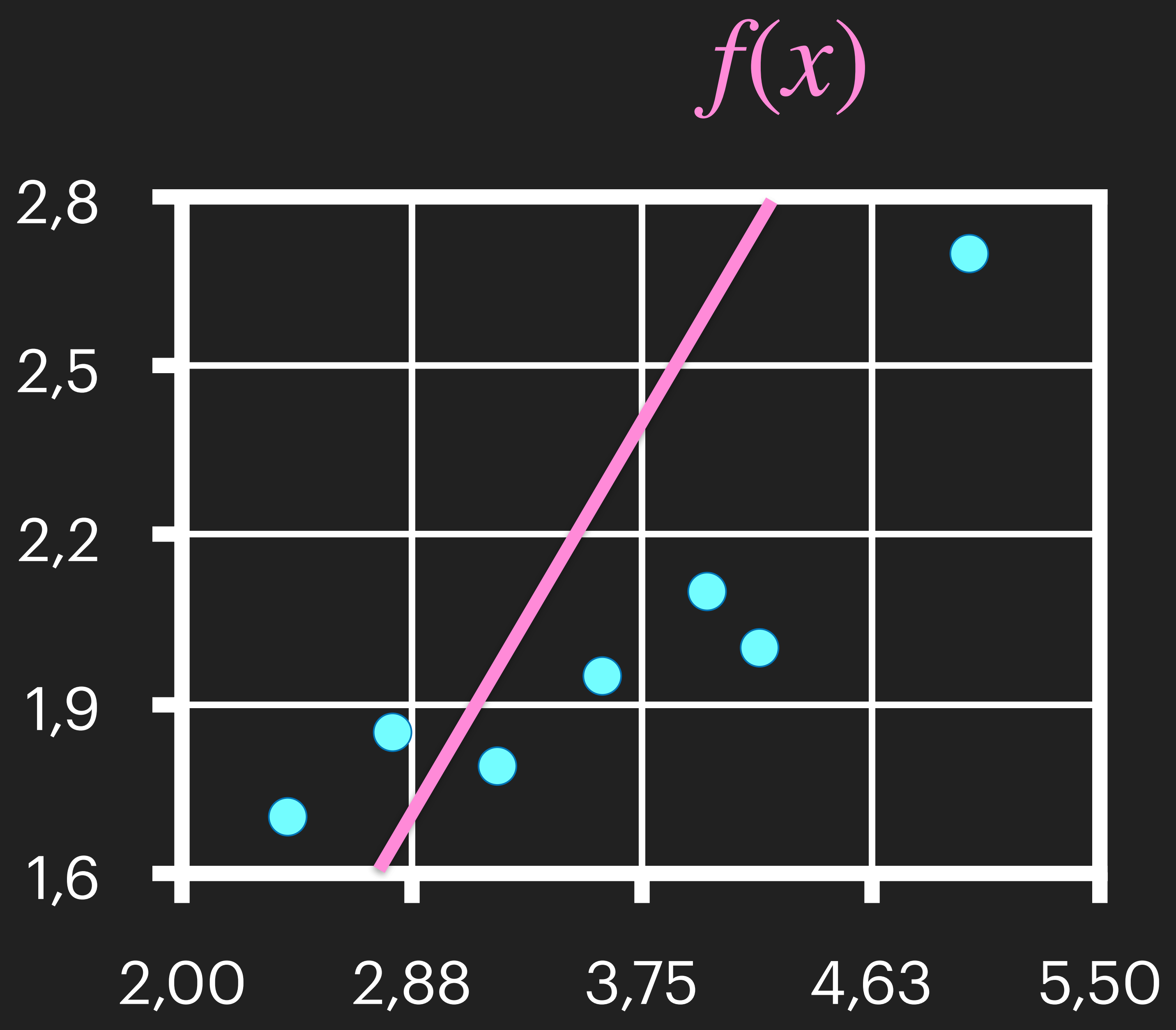
TREINANDO UM ADALINE



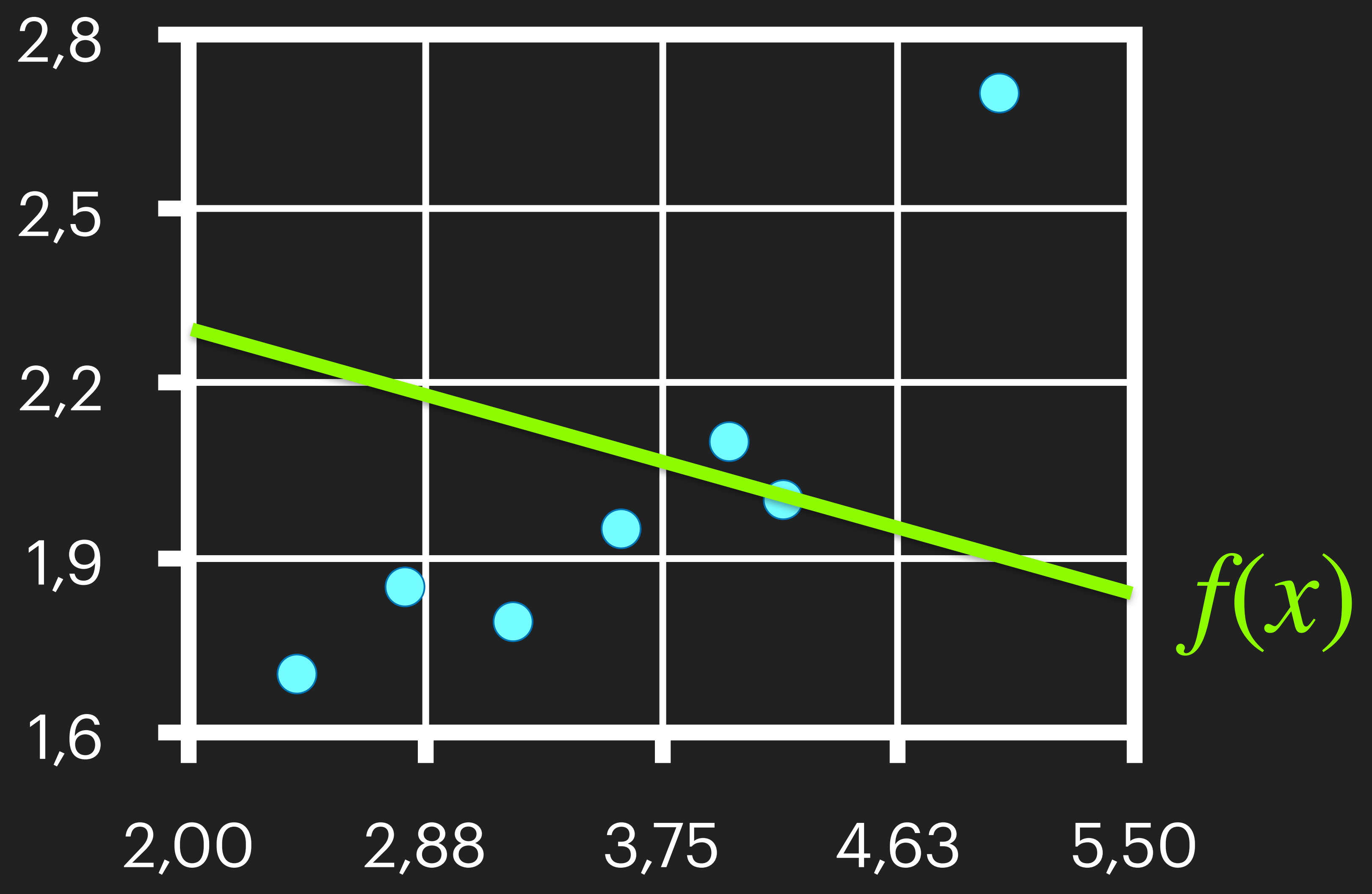
x	y
2,40	1,70
2,80	1,85
3,20	1,79
3,60	1,95
4,00	2,10
4,20	2,00
5,00	2,70

Qual a relação entre x e y?

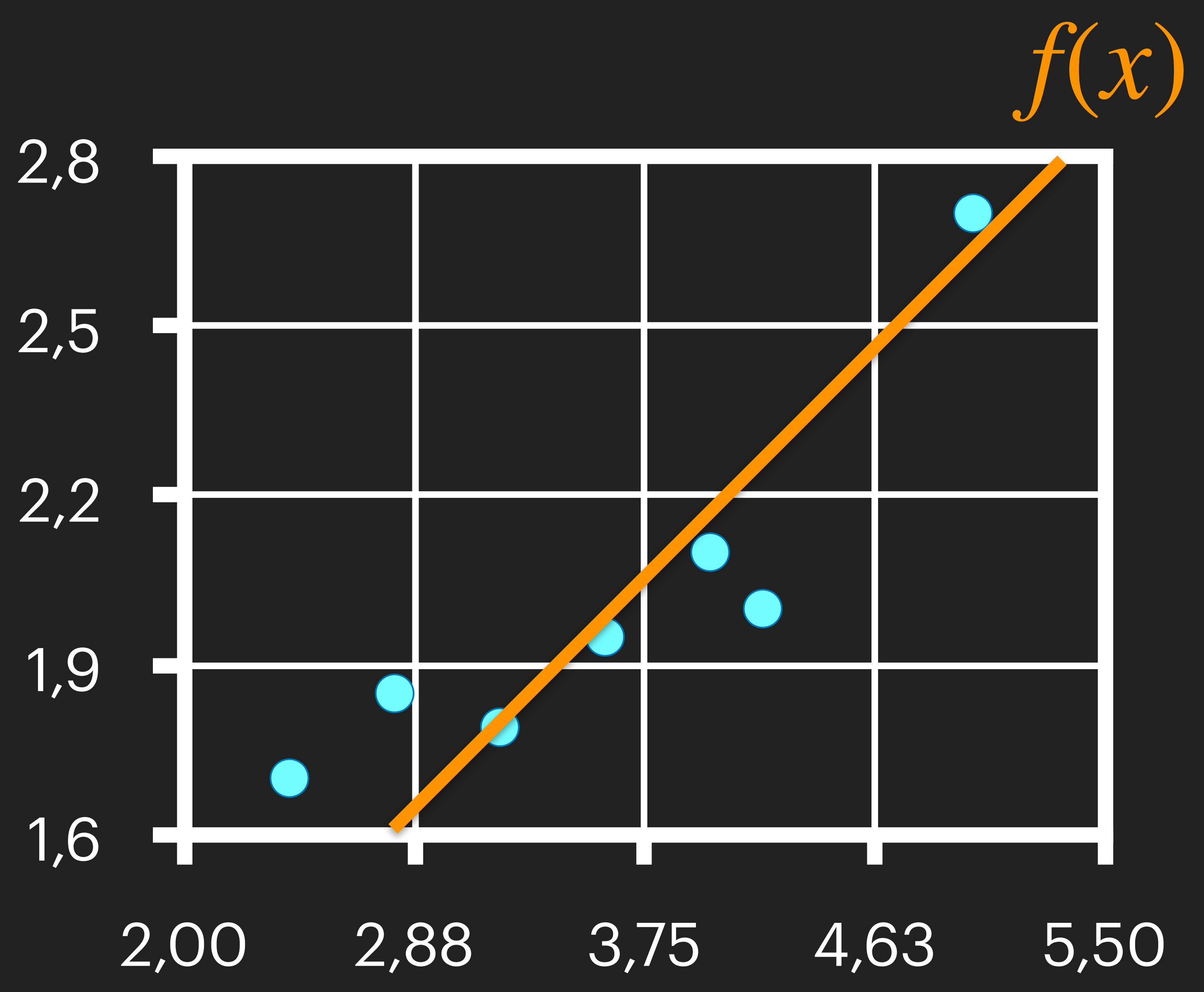
TREINANDO UM ADALINE



$f(x) = 0{,}92x - 1$, i.e.,
$\mathbf{w} = [-1,\ 0.92]$

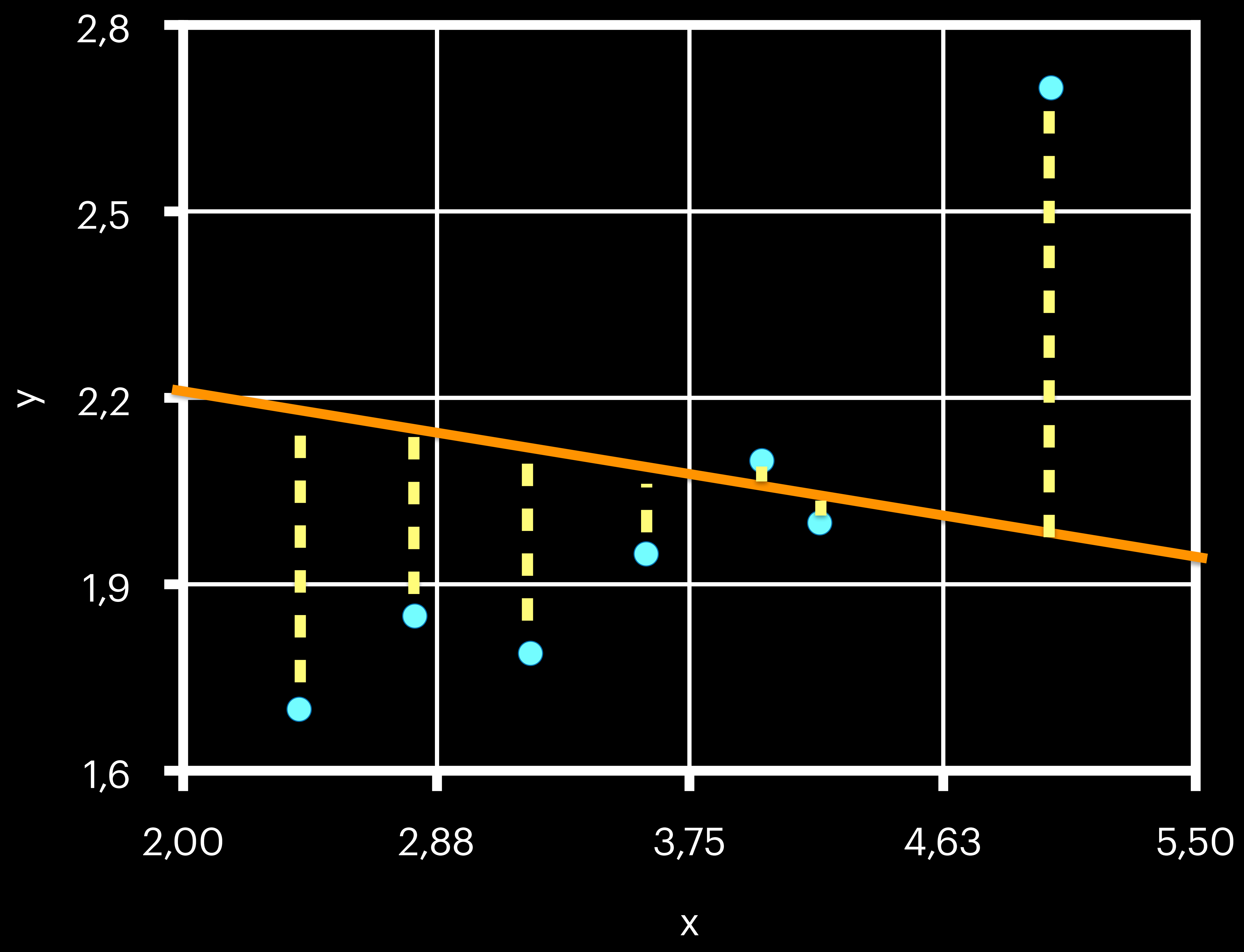


$f(x) = 0{,}11x + 2{,}5$, i.e.,
$\mathbf{w} = [2.5,\ 0.11]$



$f(x) = 0{,}52x - 0{,}1$, i.e.,
$\mathbf{w} = [0.10,\ 0.52]$

FUNÇÃO DE PERDA/CUSTO/OBJETIVO

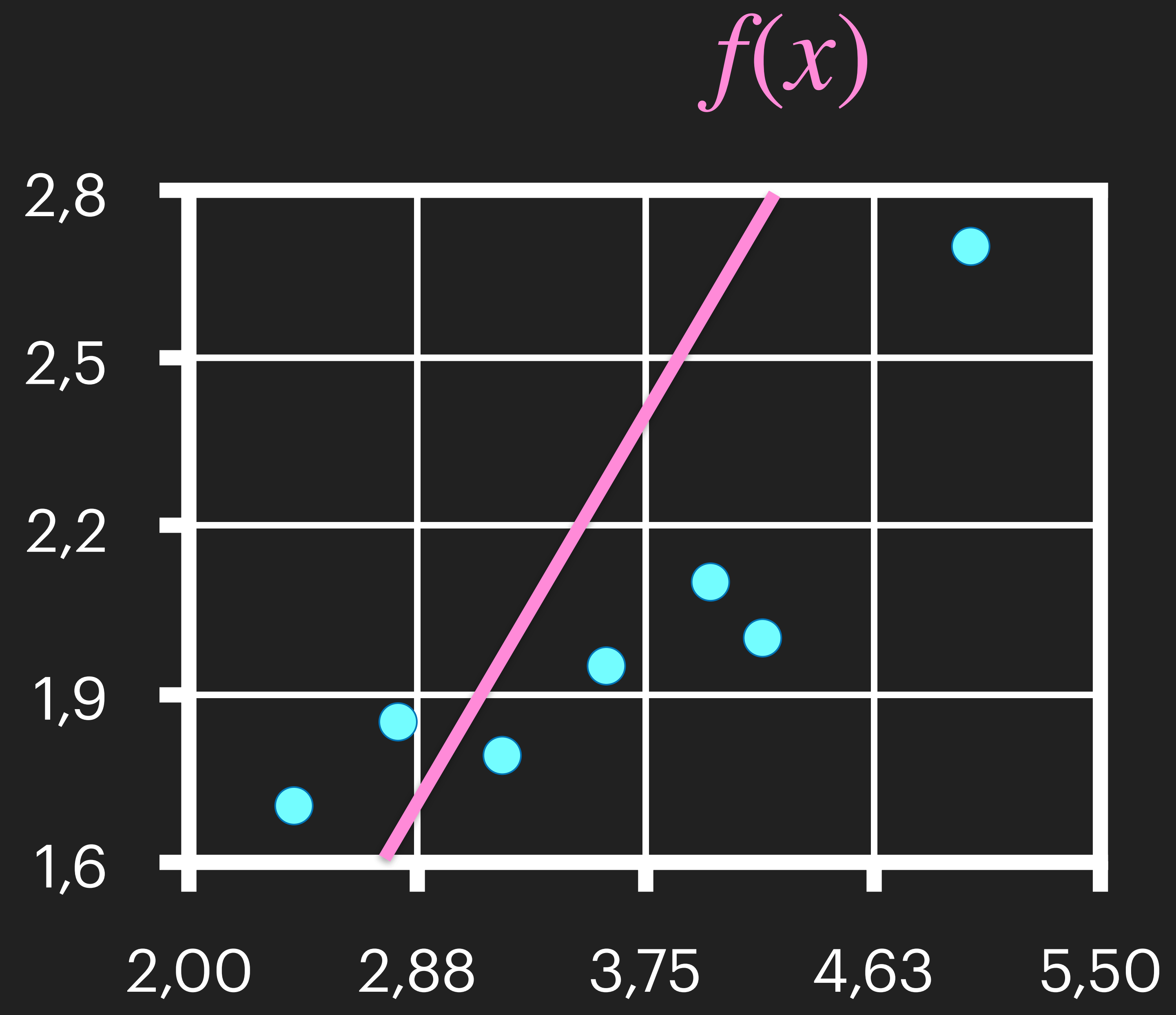


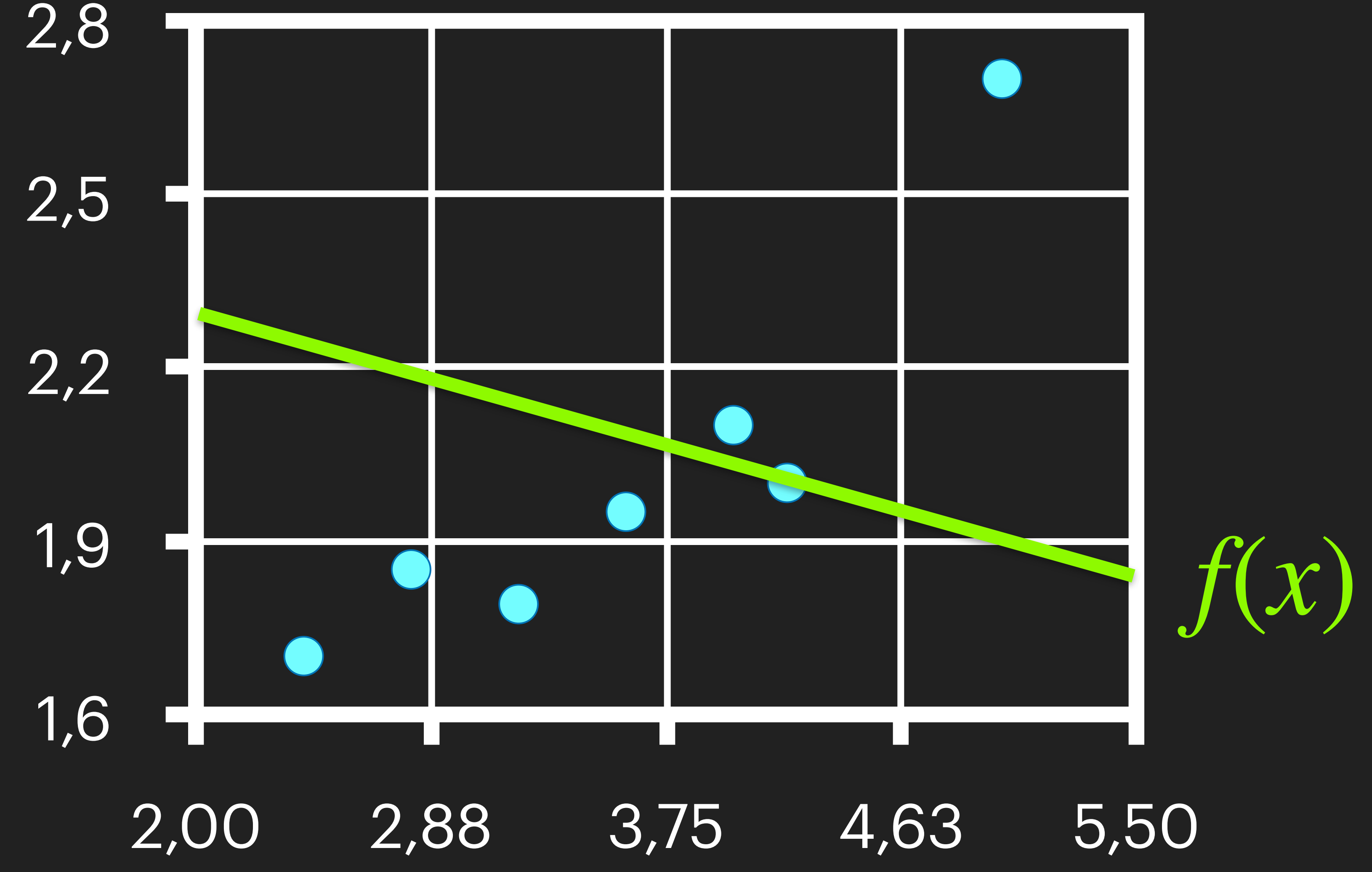
Embora existam muitas funções de perda, todas elas essencialmente computam o desempenho do modelo com base na dissimilaridade entre o valor previsto e seu valor real em um conjunto de dados, ou seja, o ERRO!

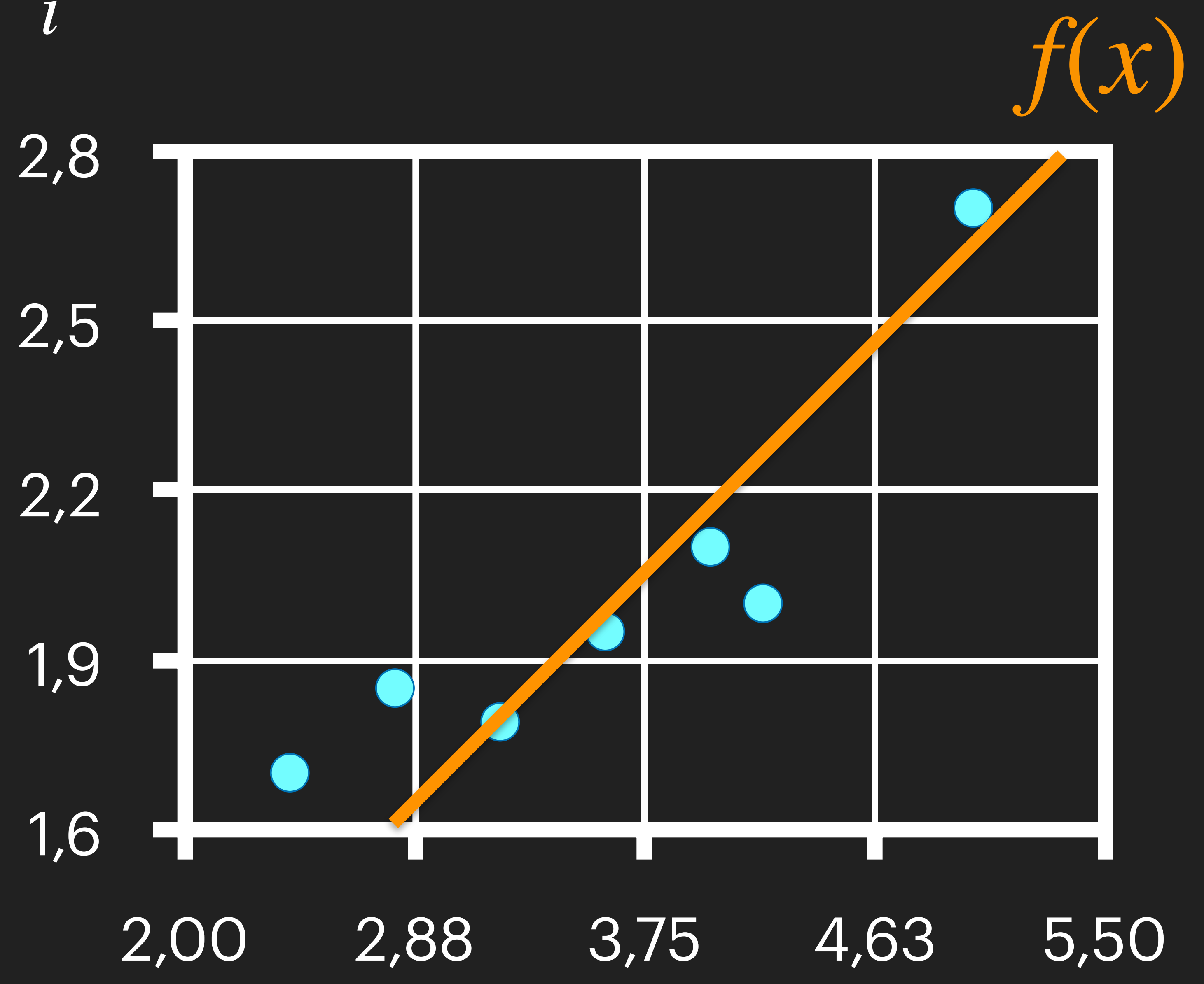
$$MSE(f) = \sum_i e_i^2 \,.$$

ERRO QUADRÁTICO MÉDIO

$$MSE(f) = \frac{1}{n}\sum_i e_i^2 = \frac{1}{n}\sum_i \left(y_i - f(x_i)\right)^2 = \frac{1}{n}\sum_i \left(y_i - \mathbf{x}_i^\intercal \mathbf{w}\right)^2$$







MSE$(f) = 0{,}17$

$f(x) = 0{,}92x - 1$, i.e., $\mathbf{w} = [-1,\ 0.92]$

MSE$(f) = 0{,}08$

$f(x) = 0{,}11x + 2{,}5$, i.e., $\mathbf{w} = [2.5,\ 0.11]$

MSE$(f) = 0{,}02$

$f(x) = 0{,}52x - 0{,}1$, i.e., $\mathbf{w} = [0.10,\ 0.52]$

TREINANDO UM ADALINE (Recap)

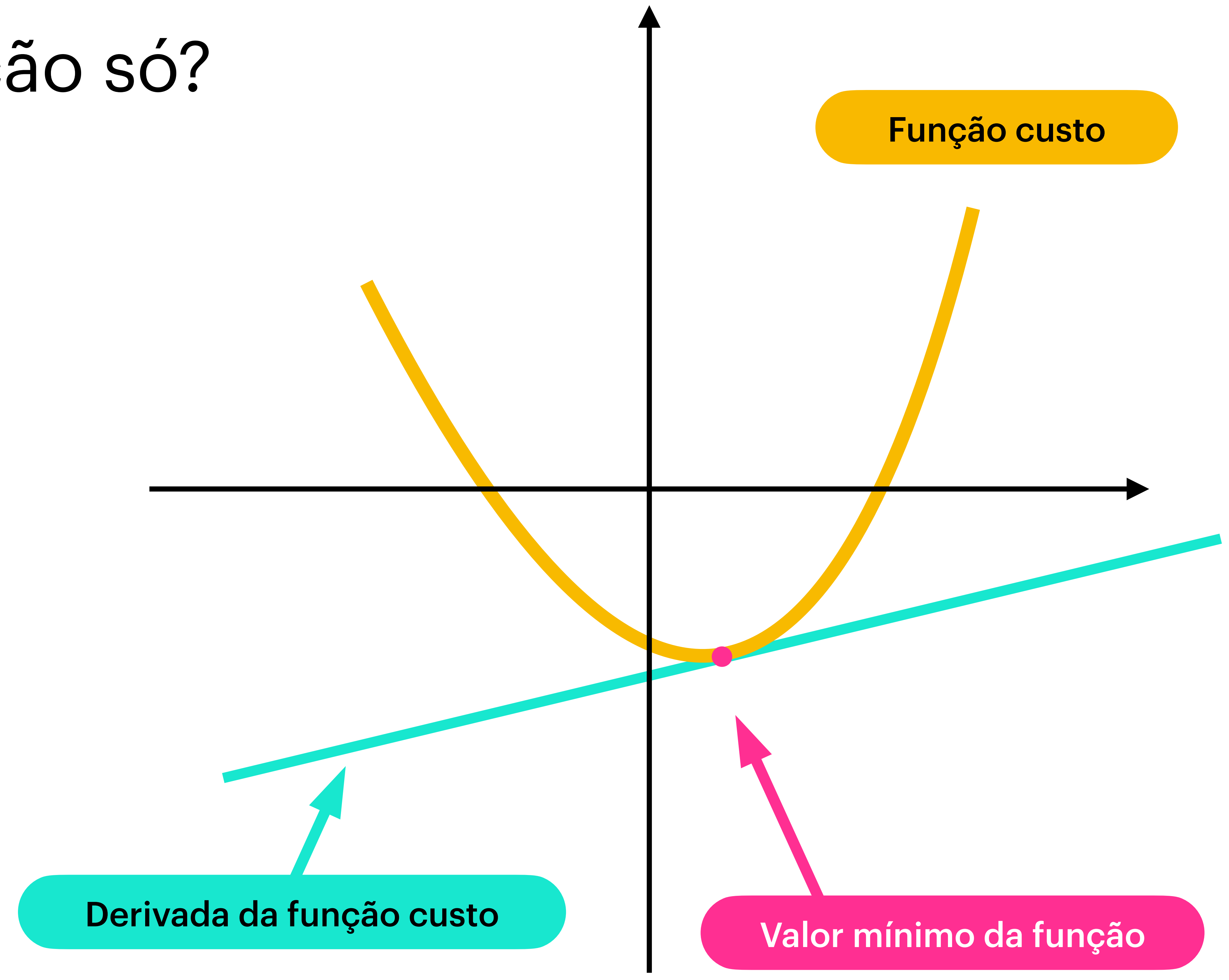
- Premissa básica: achar o vetor de pesos que minimize o TODOS os erros;
- Forma de calcular o erro continua a mesma, i.e., $e = d - y = d - \mathbf{x}^\intercal \mathbf{w}$.
- Como agregar TODOS os erros em uma função só?

$$J(\mathbf{e}) = \sum_i e_i^2.$$

$$J(\mathbf{w}) = \sum_i (d_i - \mathbf{x}_i^\intercal \mathbf{w})^2.$$

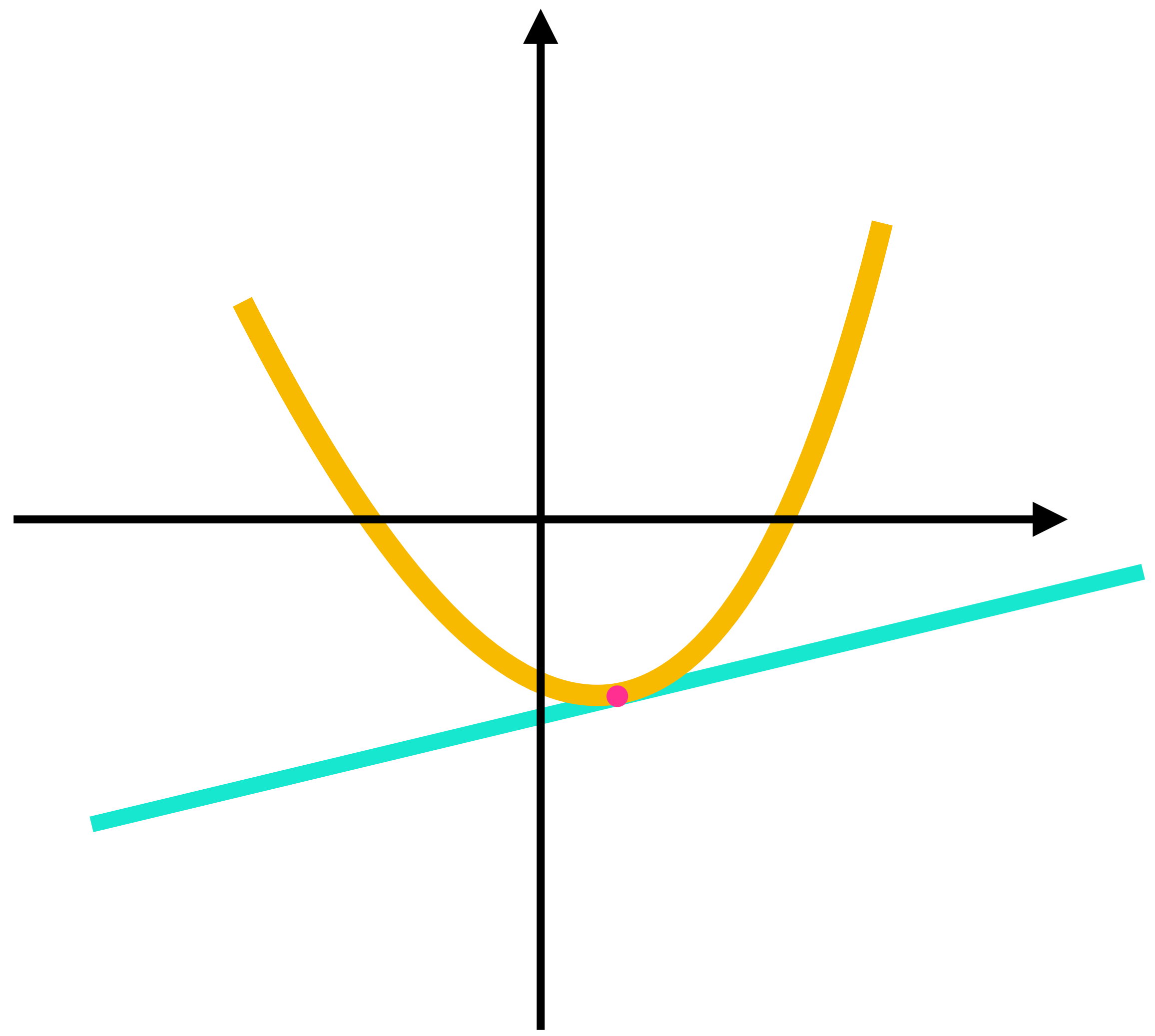
$$J(\mathbf{w}) = \sum_i d_i^2 - 2d_i\mathbf{x}_i^\intercal \mathbf{w} + (\mathbf{x}_i^\intercal \mathbf{w})^2.$$

$$\boxed{\mathbf{w}^\star = \min_{\mathbf{w}} J(\mathbf{w})}$$



TREINANDO UM ADALINE (Recap)

- Com o intuito de minimizar $J(\mathbf{w})$, é necessário que seja obtido a direção do ajuste a ser aplicado no vetor de pesos para aproximar a solução do mínimo de $J(\mathbf{w})$;
- De modo iterativo, iremos realizar $\mathbf{w}^{(t+1)} = \mathbf{w}^{(t)} + \Delta\mathbf{w}$;
- Para tal, usa-se o **gradiente** da função $J(\mathbf{w})$ no ponto $\mathbf{w}^{(t)}$;
- Sabe-se que o gradiente possui a mesma direção da maior variação do erro (aponta na direção do crescimento da função). Portanto, o ajuste deve ocorrer na direção contrária do gradiente. Logo a variação dos pesos pode ser descrita como:



$$\Delta\mathbf{w} \propto -\nabla\mathbf{w}.$$

TREINANDO UM ADALINE (Pulo do 🐈)

- Sabe-se que a função J depende de e, bem como sabe-se que e depende de y e, ainda, sabe-se que y depende de $\mathbf{w}$;
- Cada componente do gradiente da função custo ∇J pode ser obtida com base na regra da cadeia, a saber,

$$\frac{\partial J}{\partial w_i} = \frac{\partial J}{\partial e}\frac{\partial e}{\partial y}\frac{\partial y}{\partial w_i}$$

- Quanto valem os termos abaixo?

a) $\frac{\partial J}{\partial e}$ b) $\frac{\partial e}{\partial y}$ c) $\frac{\partial y}{\partial w_i}$ d) $\frac{\partial J}{\partial w_i}$

TREINANDO UM ADALINE (Pulo do 🐈)

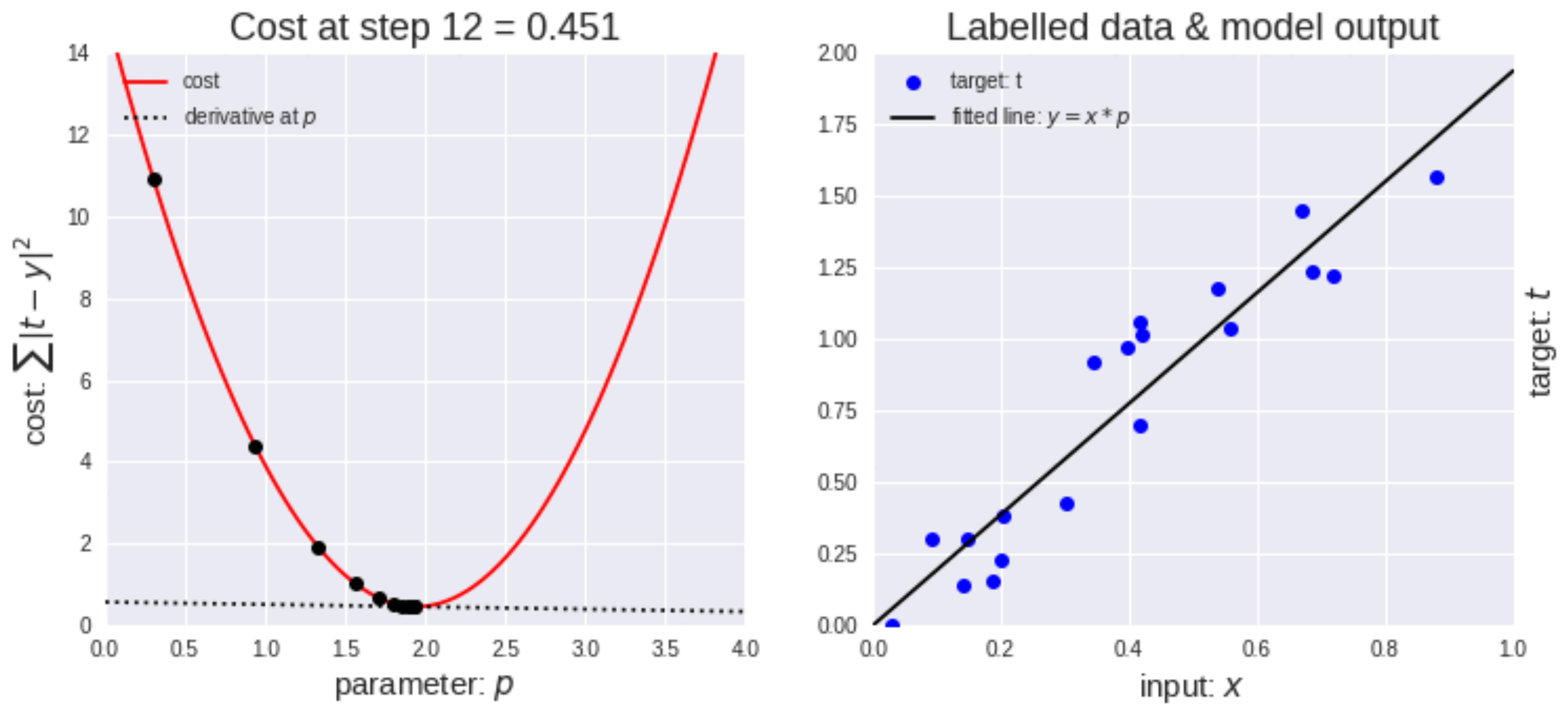
- Considerando que

a) $\frac{\partial J}{\partial e} = e$ b) $\frac{\partial e}{\partial y} = -1$ c) $\frac{\partial y}{\partial w_i} = x_i$ d) $\frac{\partial J}{\partial w_i} = e(-1)x_i = -ex_i$.

- Considerando também a regra de aprendizagem $\mathbf{w}^{(t+1)} = \mathbf{w}^{(t)} + \Delta\mathbf{w}$;
- Considerando que $\Delta\mathbf{w} \propto \nabla J$ e que $\frac{\partial J}{\partial \mathbf{w}} = -\mathbf{x}e$, podemos reescrever a regra de atualização da seguinte forma:

$$\mathbf{w}^{(t+1)} = \mathbf{w}^{(t)} + \eta e^{(t)}\mathbf{x}$$

TREINANDO UM ADALINE (Pulo do 🐈)



https://www.kdnuggets.com/2019/03/neural-networks-numpy-absolute-beginners-part-2-linear-regression.html/2

Referências

- Richard O. Duda, Peter E. Hart, David G. Stork. **Pattern Classification**. John Wiley & Sons, 2012.
- Guilherme A. Barreto. **Introdução à Classificação de Padrões**. Grupo de Aprendizado de Máquinas – GRAMA, 2021.
- KDNugets. **Neural Networks with Numpy for Absolute Beginners — Part 2: Linear Regression**. https://www.kdnuggets.com/2019/03/neural-networks-numpy-absolute-beginners-part-2-linear-regression.html/2, março de 2019. Acessado em Março de 2022.