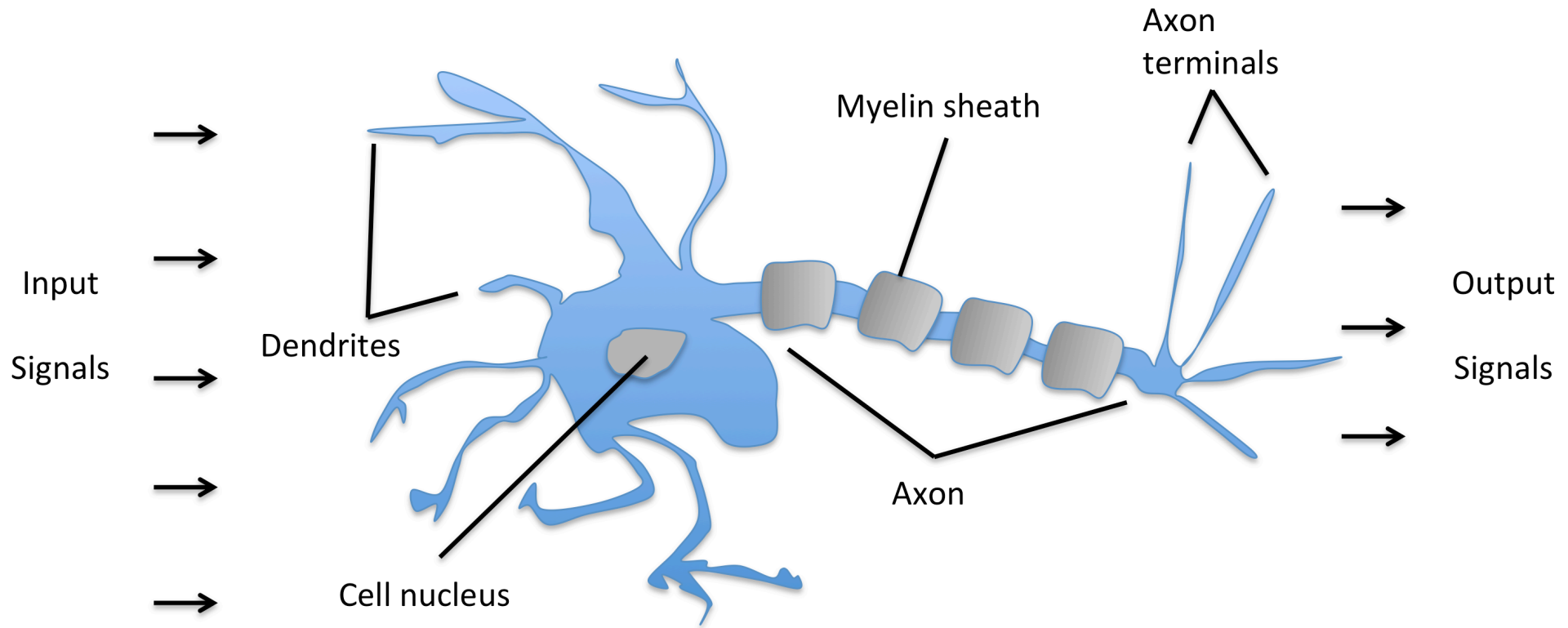
Modelos preditivos

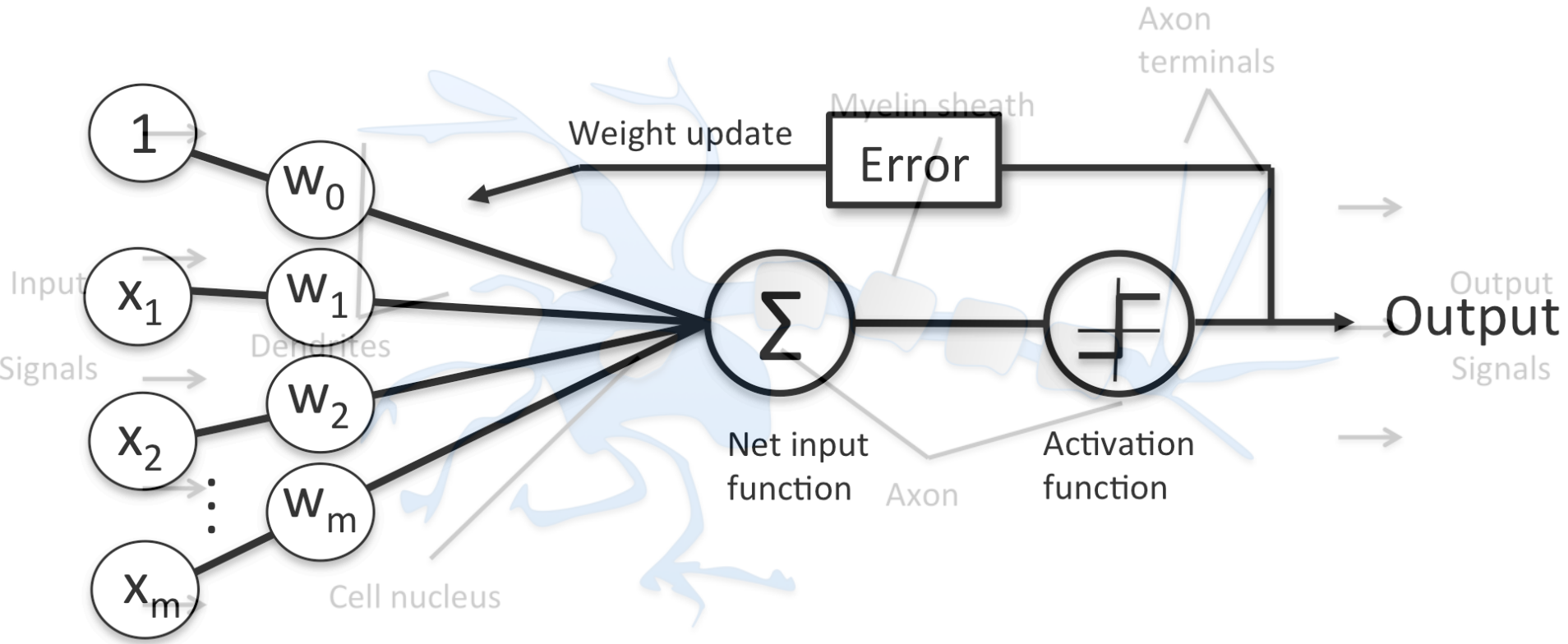
Classificação

predizer categoria

Métodos baseados em otimização



McCulloch, W. S., & Pitts, W. (1943). A logical calculus of the ideas immanent in nervous activity. *The bulletin of mathematical biophysics*, *5*(4), 115-133.



Rosenblatt, F. (1957). *The perceptron, a perceiving and recognizing automaton Project Para*. Cornell Aeronautical Laboratory.

pesos características

$$\mathbf{w} = \begin{bmatrix} w_1 \\ w_2 \\ \vdots \\ w_m \end{bmatrix}, \quad \mathbf{x} = \begin{bmatrix} x_1 \\ x_2 \\ \vdots \\ x_m \end{bmatrix}.$$

Entrada da rede

$$z = w_1 x_1 + \cdots + w_m x_m$$

Função de ativação (*Heaviside step function*)

$$\phi(z) = \begin{cases} 1 & \text{if } z \geq \theta \\ -1 & \text{otherwise .} \end{cases}$$

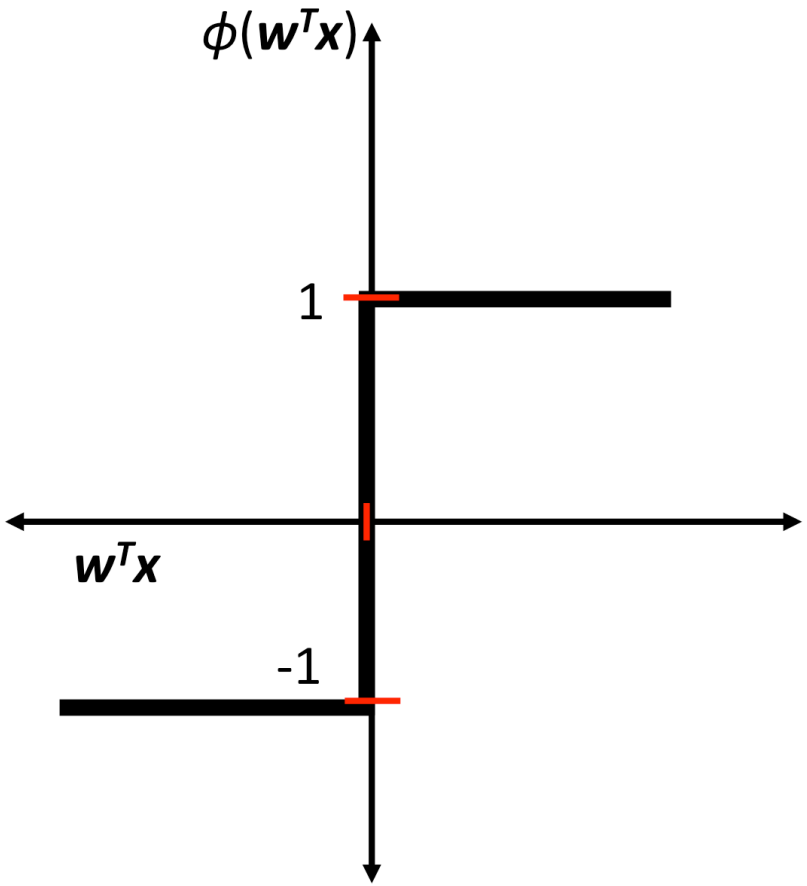
classes

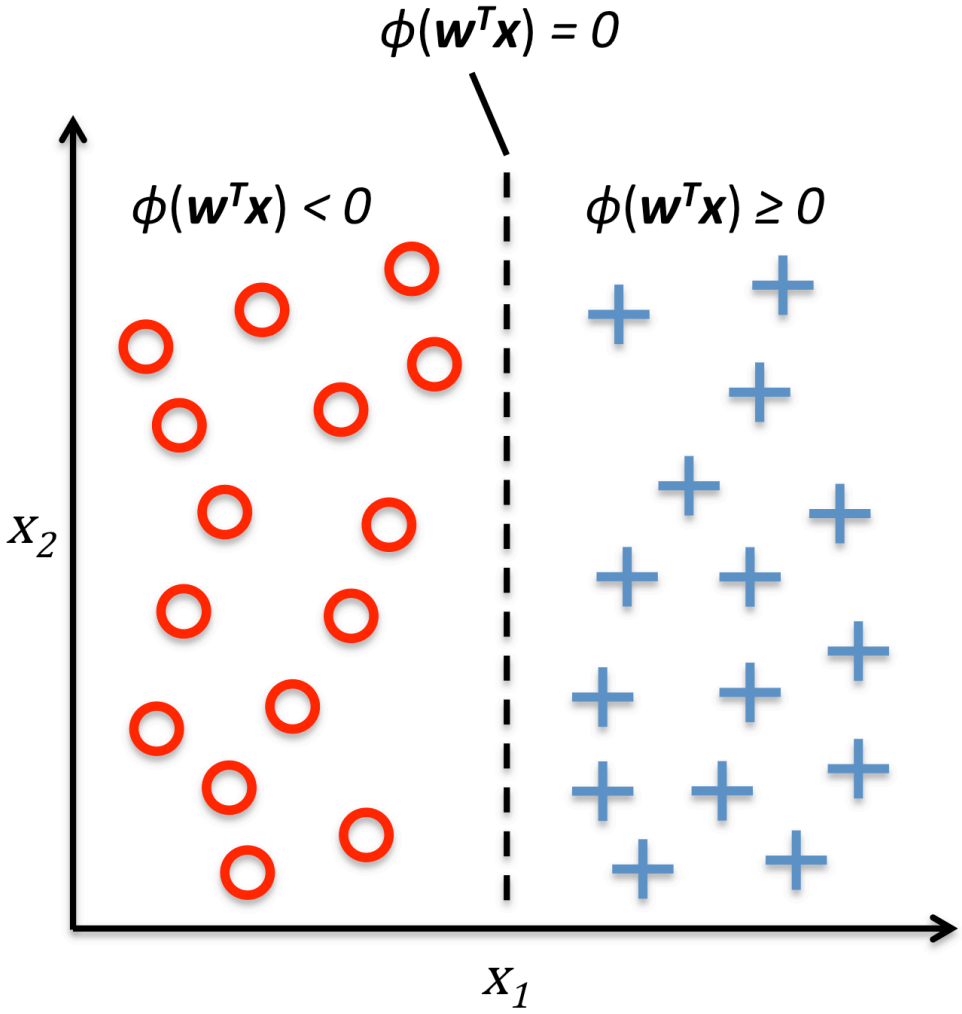
$$w_0 = -\theta \text{ and } x_0 = 1$$

$$\phi(z) = \begin{cases} 1 & \text{if } z \geq 0 \\ -1 & \text{otherwise .} \end{cases}$$

$$z = w_0 x_0 + w_1 x_1 + \cdots + w_m x_m = \mathbf{w^T x} = \sum_{j=0}^{m} \mathbf{w_j x_j} = \mathbf{w}^T \mathbf{x}.$$

Rosenblatt, F. (1957). *The* **perceptron**, *a perceiving and recognizing automaton Project Para*. Cornell Aeronautical Laboratory.


$\phi(\boldsymbol{w}^T\boldsymbol{x})$
1
$\boldsymbol{w}^T\boldsymbol{x}$
-1


$\phi(\boldsymbol{w}^T\boldsymbol{x}) = 0$
$\phi(\boldsymbol{w}^T\boldsymbol{x}) < 0$
$\phi(\boldsymbol{w}^T\boldsymbol{x}) \geq 0$
x_2
x_1

Algoritmo Perceptron

1. Initialize the weights to 0 or small random numbers.
2. For each training sample $\mathbf{x}^{(i)}$, perform the following steps:
   (a) Compute the output value $\hat{y}$.
   (b) Update the weights.

$$w_j := w_j + \Delta w_j$$

$$\Delta w_j = \eta \left(y^{(i)} - \hat{y}^{(i)} \right) x_j^{(i)}$$

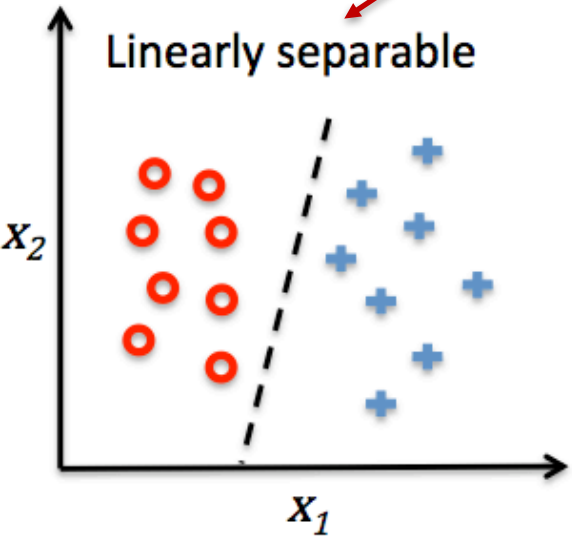
Taxa de aprendizado [0.0, 1.0]

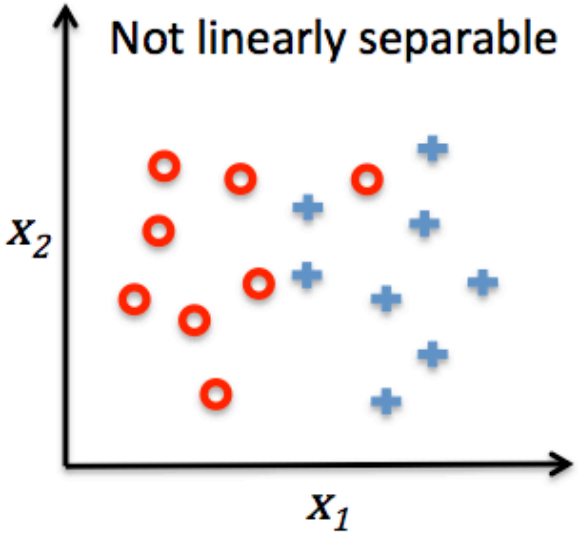
Rótulo de classe real do exemplo de treinamento

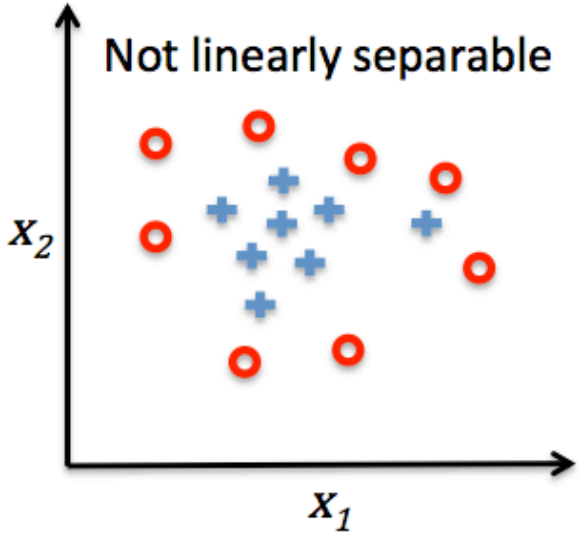
Rótulo de classe prevista

Taxa de aprendizado pequena

Garantia de convergência do Perceptron...

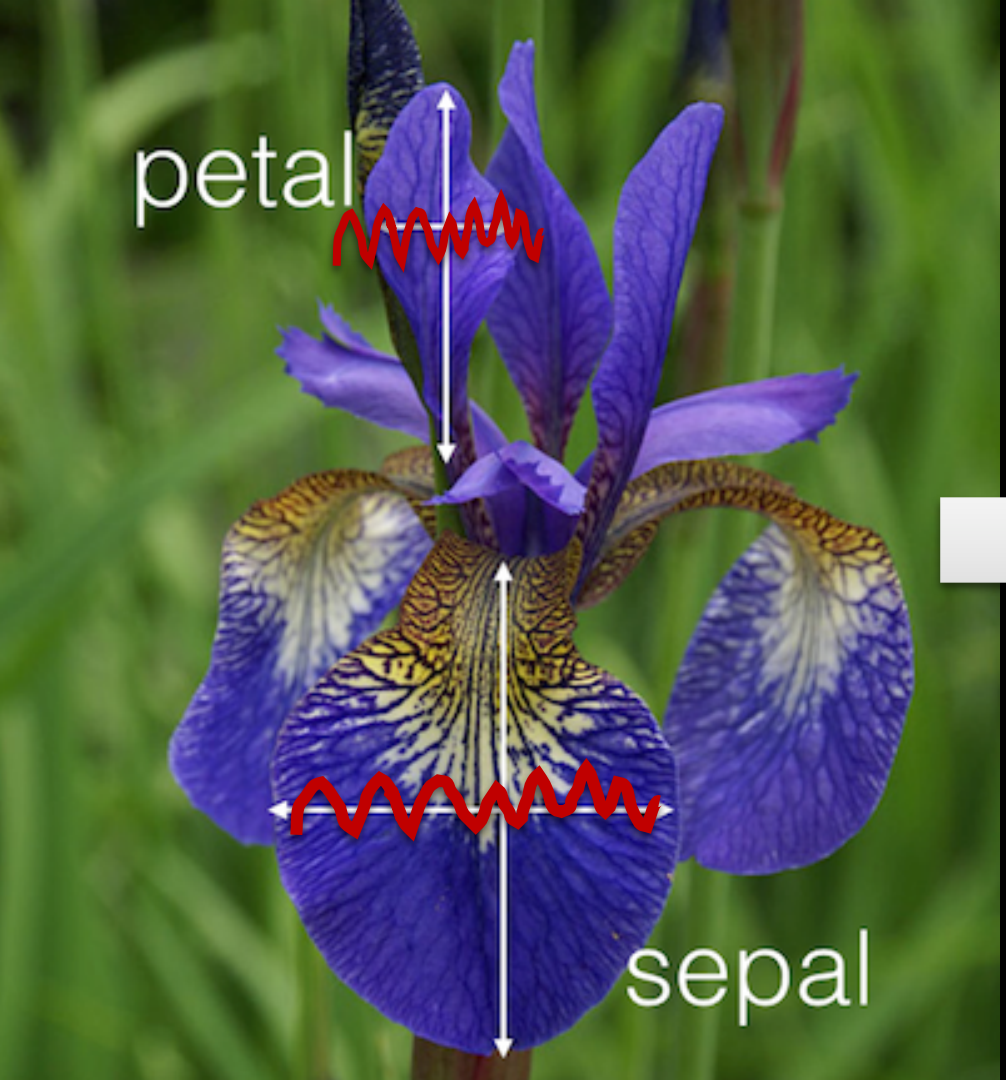






... caso contrário:

1) Estabelecer máximo de Épocas; ou
2) Valor de tolerância para exemplos mal classificados

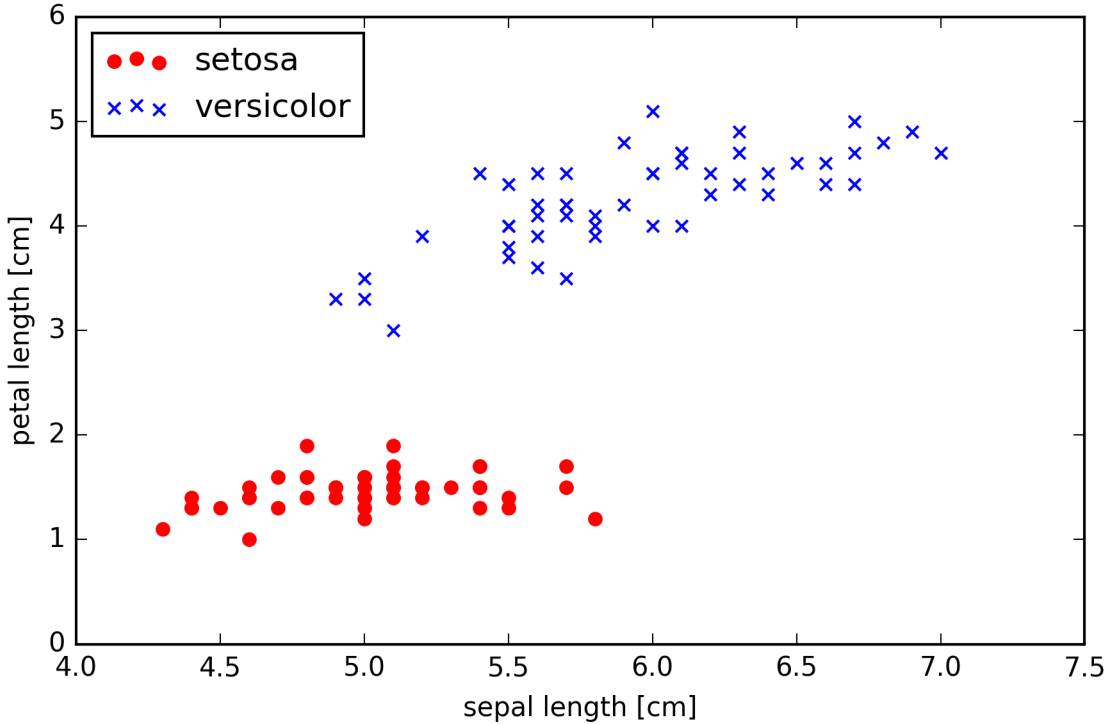


Iris Setosa?

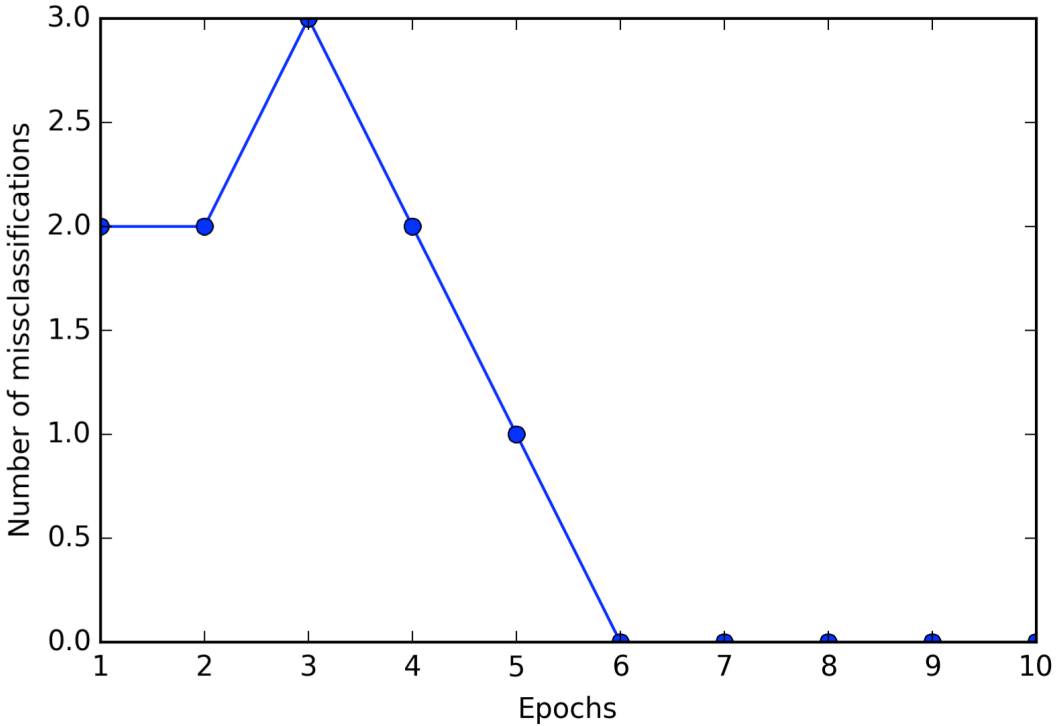
Iris Versicolor?

~~Iris Virginica?~~

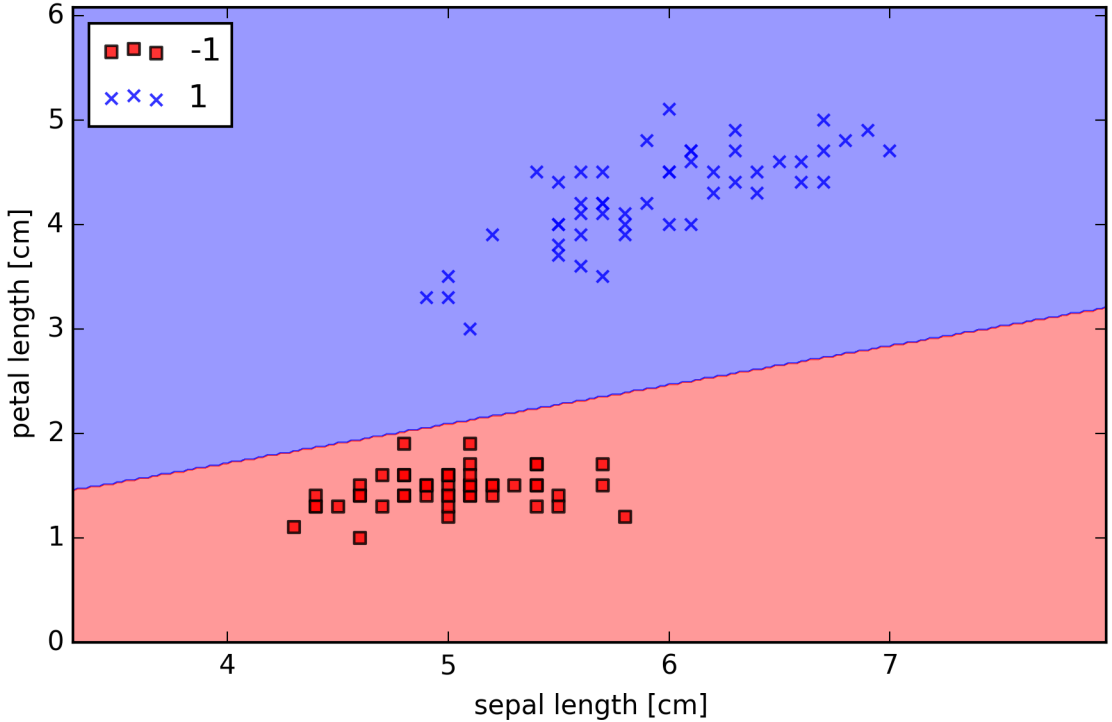
C: carregar base de dados Iris



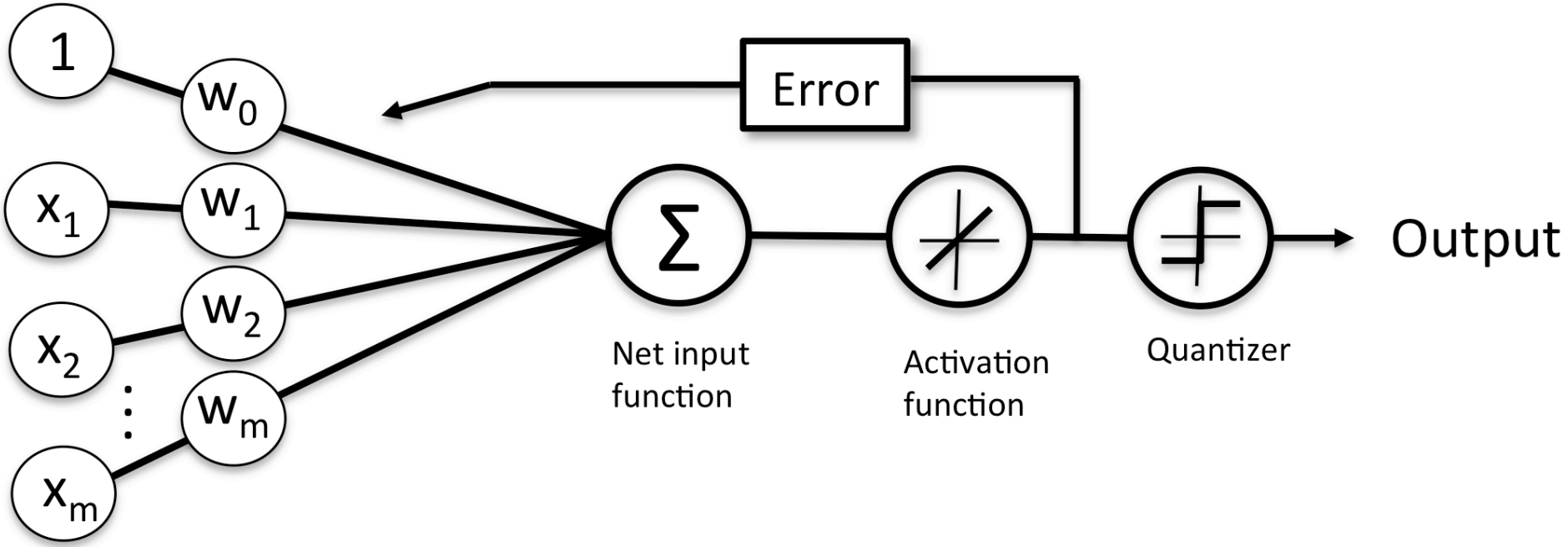
C: treinamento do Perceptron e curva de erro



C: visualizar fronteira de decisão



ADAptive LInear Neuron



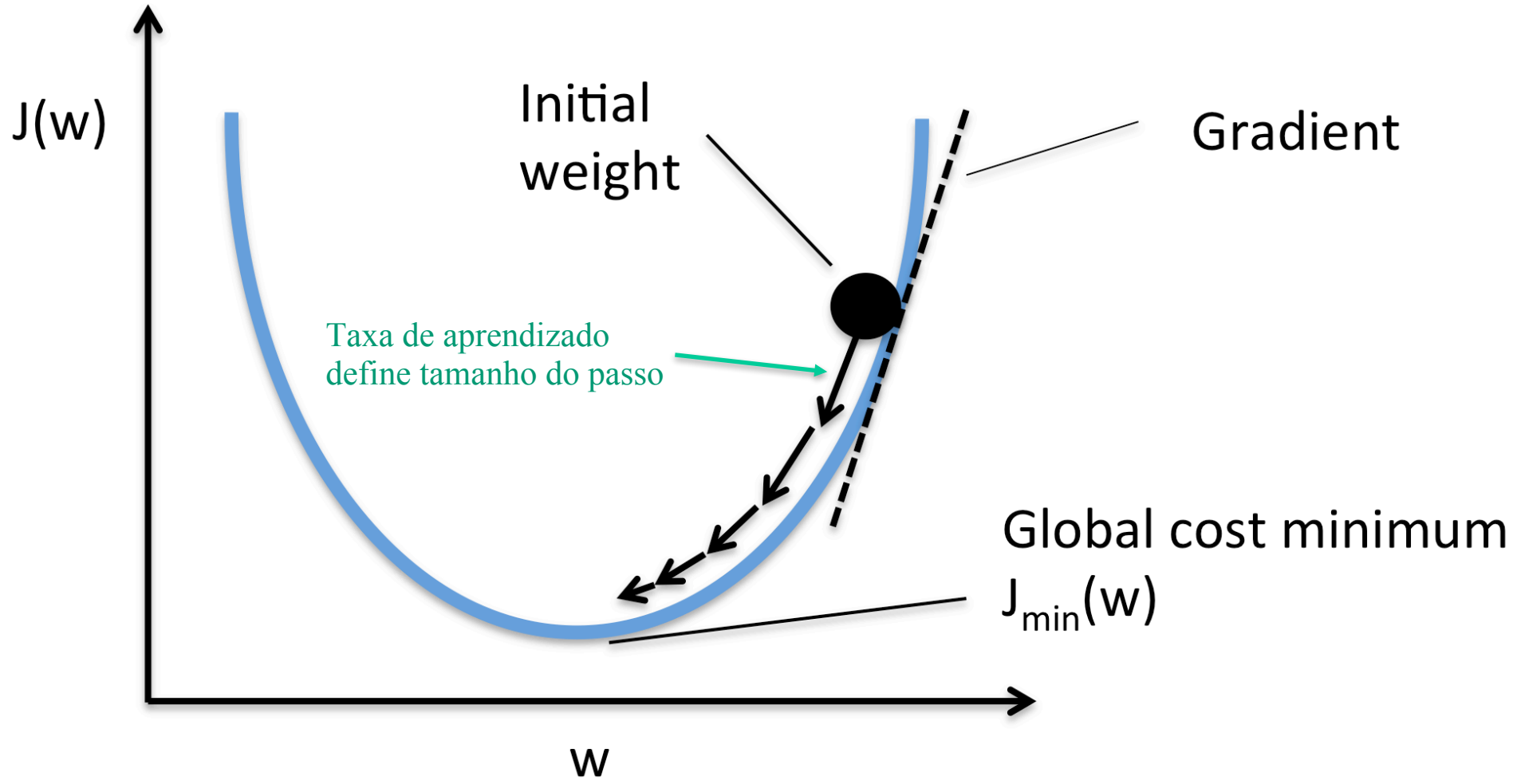
Widrow, B. (1960). *Adaptive "adaline" Neuron Using Chemical" memistors."*.

$$\phi(\mathbf{w}^T\mathbf{x}) = \mathbf{w}^T\mathbf{x}$$ ...o próprio valor: $\in \mathbb{R}$

$$J(\mathbf{w}) = \frac{1}{2}\sum_i \left(y^{(i)} - \phi(z^{(i)})\right)^2$$ Sum of Squared Errors (SSE) (função convexa)

$\nabla J(\mathbf{w})$ Gradient descent para encontrar os pesos

$$\frac{\partial J}{\partial w_j} = -\sum_i \left(y^{(i)} - \phi(z^{(i)})\right) x_j^{(i)}$$


J(w)
Initial
weight
Gradient
Taxa de aprendizado
define tamanho do passo
Global cost minimum
$J_{min}(w)$
w

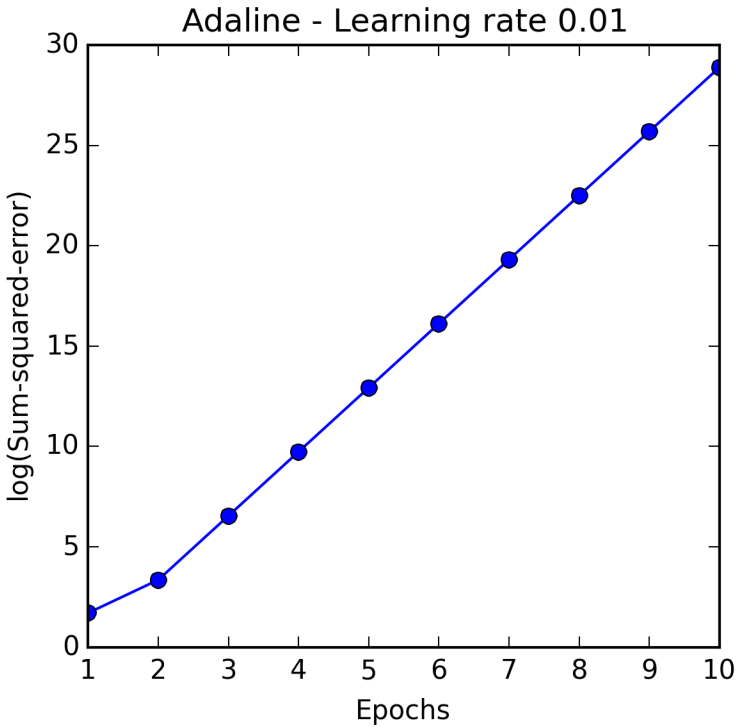
$$\frac{\partial J}{\partial w_j} = -\sum_i \Big(y^{(i)} - \phi\big(z^{(i)}\big) \Big) x_j^{(i)}$$

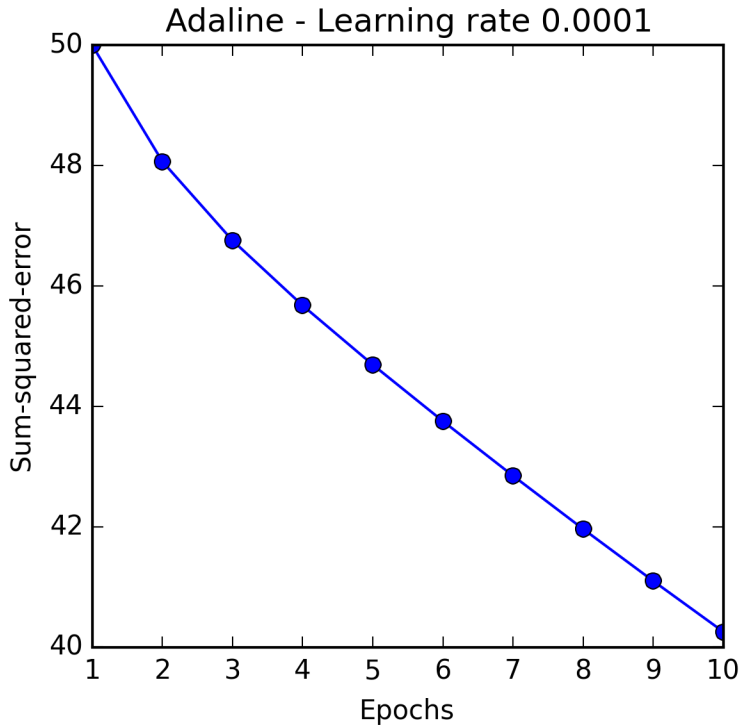
$$\Delta w_j = -\eta \frac{\partial J}{\partial w_j} = \eta \sum_i \Big(y^{(i)} - \phi\big(z^{(i)}\big) \Big) x_j^{(i)}$$

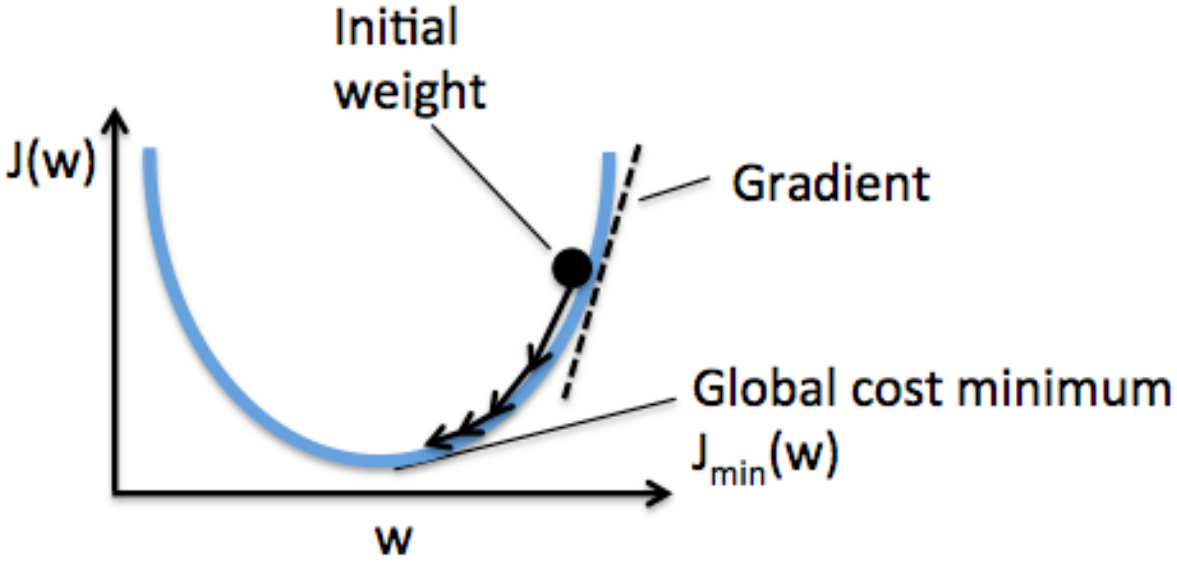
$$\mathbf{w} := \mathbf{w} + \Delta \mathbf{w}.$$

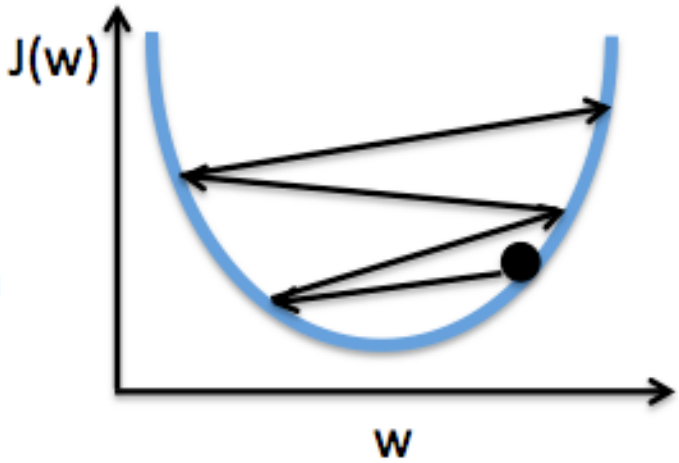
Batch gradient descent

C: treinamento e curva de erro vs. época





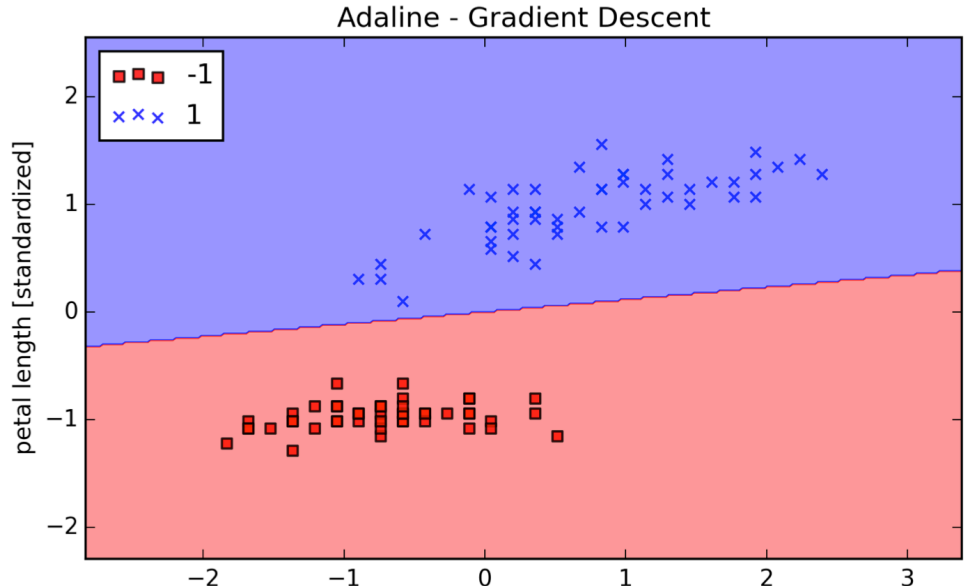

Initial
weight
J(w)
Gradient
Global cost minimum
$J_{min}(w)$
w

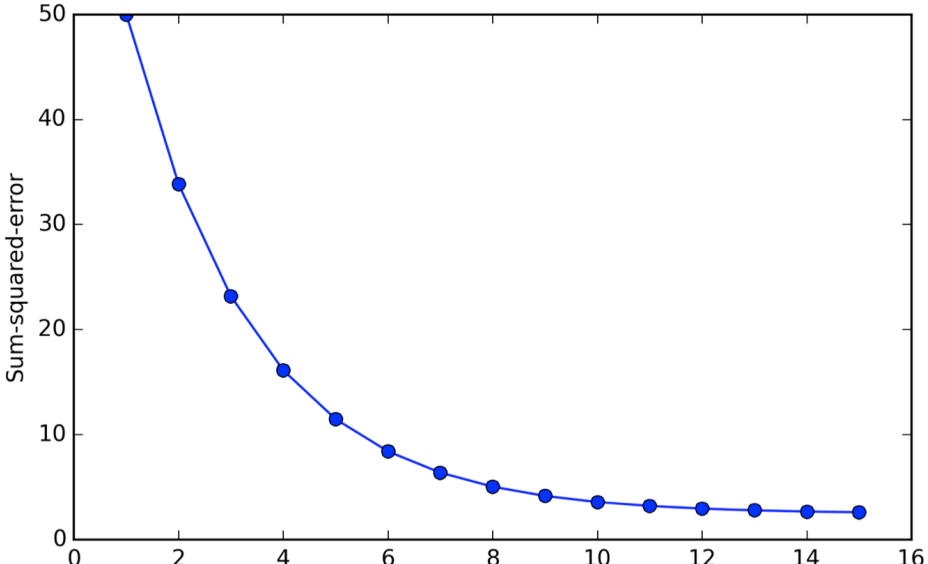

J(w)
w

C: reescala de valores de características

$$\mathbf{x'}_j = \frac{\mathbf{x} - \mu_\mathbf{j}}{\sigma_j}.$$

Dados passam a seguir
a Distribuição Normal





E quando lidamos com milhões de dados de treinamento?

$$\Delta \mathbf{w} = \eta \sum_i \left(y^{(i)} - \phi\left(z^{(i)}\right) \right) \mathbf{x}^{(i)}.$$

Batch gradient descent (BGD)

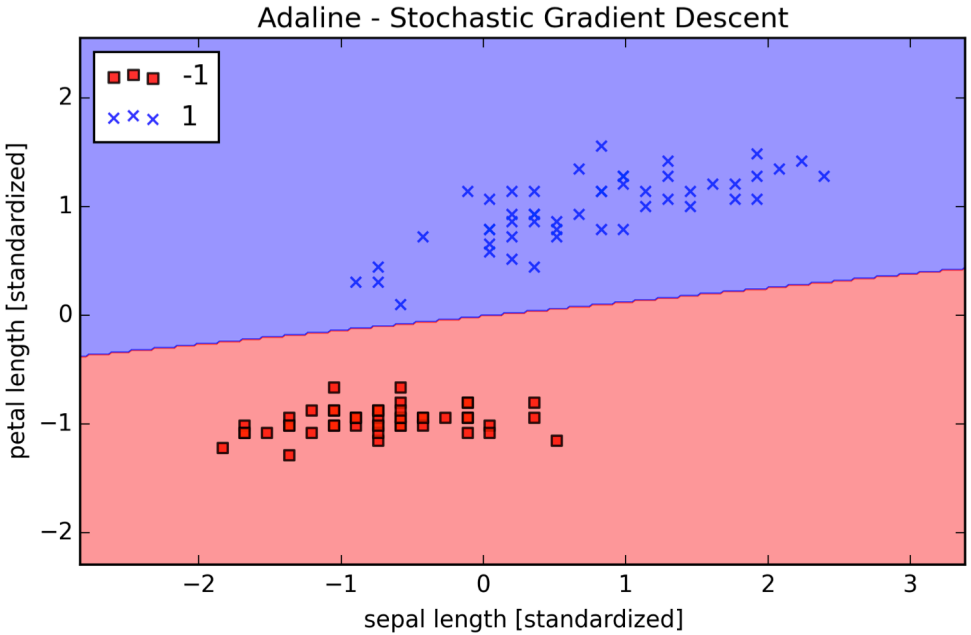
vs.

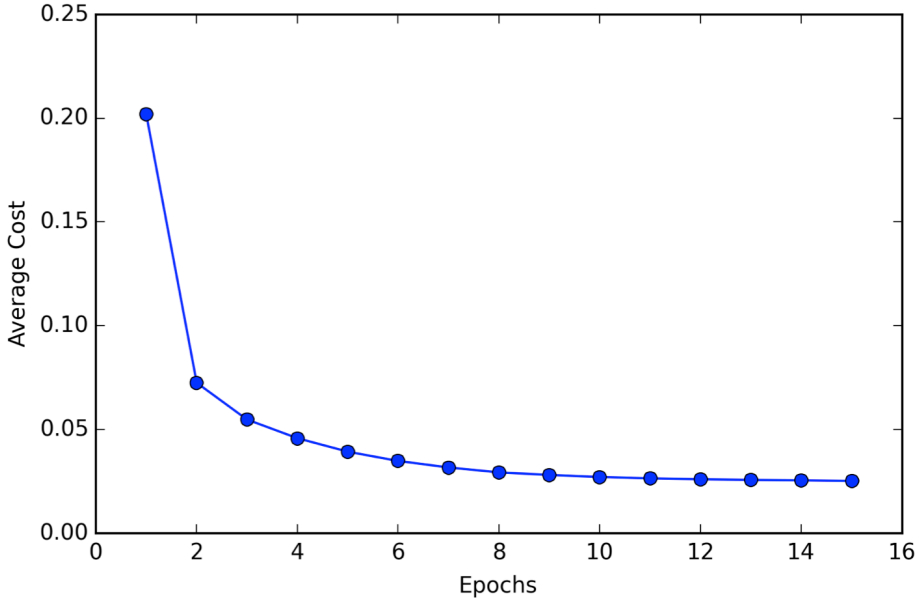
$$\Delta \mathbf{w} = \eta \left(y^{(i)} - \phi\left(z^{(i)}\right) \right) \mathbf{x}^{(i)}.$$

Stochastic gradient descent (SGD)
(atualização incremental de pesos)

Importante que os exemplos de treinamento estejam sempre embaralhados (a cada época)

C: treinamento SGD e curva de erro vs. época





Vantagens da versão *Stochastic*:

- Convergência mais rápida (por que?)
- Uso em aprendizado online (novos dados estão sempre chegando)

Desvantagem:

- Menor eficiência computacional em virtude da necessidade de loops vs. operações vetorizadas

Q: Você imagina alguma alternativa para ”resolver” a desvantagem do SGD e ao mesmo tempo evitar a desvantagem do BGD?

Divulgação científica

http://www.saense.com.br
https://www.facebook.com/saense/

Hendrik Macedo
Escreve sobre Inteligência Artificial no Saense.
http://www.saense.com.br/autores/artigos-publicados-por-hendrik-macedo/