Neural Networks: Perceptron

Learning outcomes

UNIVERSITY OF LEEDS

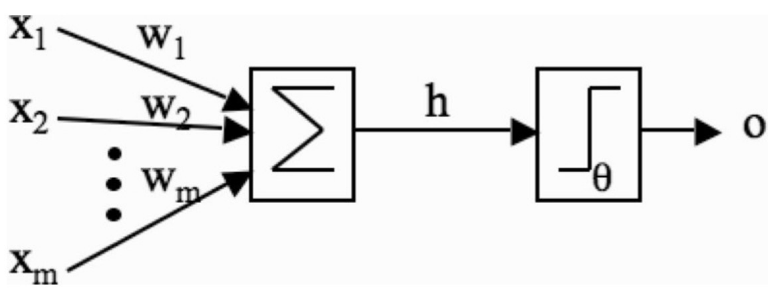
- Define linear separability.
- Justify whether a given error function is suitable for gradient descent.
- Define an appropriate error function for the perceptron.
- Derive the corresponding update algorithm.
- Describe the difference between gradient descent and stochastic gradient descent.

Recap

We want to apply gradient descent:

$$x_{t+1} = x_t - \eta \nabla f(x_t)$$

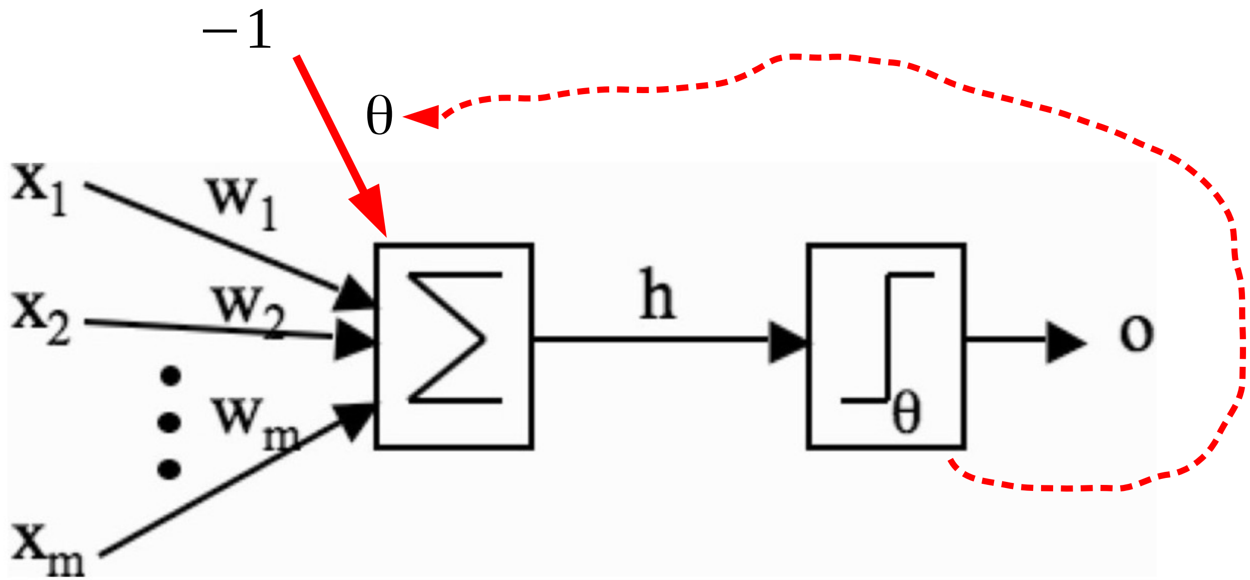
To the parameters of a perceptron:



So as to minimise an error (or loss) function, such as:

$$E(\boldsymbol{X}) = \sum_{\vec{x}_n \in \boldsymbol{X}} |y_n - t_n|$$

Bias input



$$h_w(\boldsymbol{x}) = \sum_i w_i x_i = \boldsymbol{w} \cdot \boldsymbol{x}$$

$$o(h_w) = \begin{cases} 1 & \text{if} \quad h_w > \theta \\ 0 & \text{if} \quad h_w \leq \theta \end{cases}$$

$$h_w - 1 \cdot \theta > 0$$

$$h_w - 1 \cdot \theta \leq 0$$

$$h_w(\boldsymbol{x}) = \sum_i w_i x_i - \theta$$

$$\boldsymbol{x}_{new} = \langle \boldsymbol{x}, -1 \rangle \qquad \boldsymbol{w}_{new} = \langle \boldsymbol{w}, \theta \rangle$$

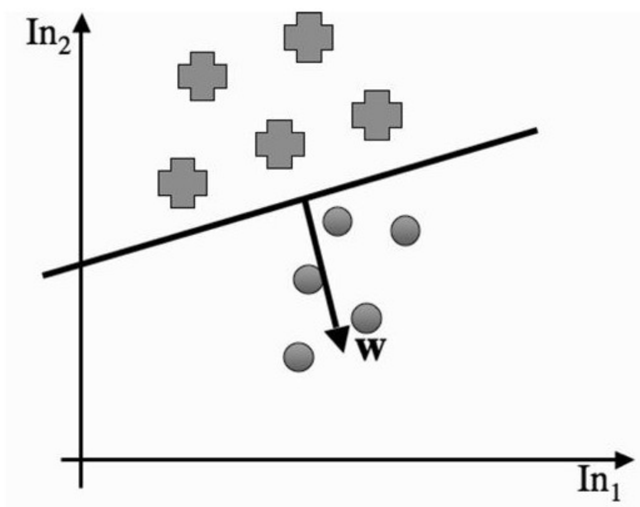
$$h_w(\boldsymbol{x}_{\boldsymbol{new}}) = \boldsymbol{w}_{\boldsymbol{new}} \cdot \boldsymbol{x}_{\boldsymbol{new}}$$

$$o(h_w) = \begin{cases} 1 & \text{if} \quad h_w > 0 \\ 0 & \text{if} \quad h_w \leq 0 \end{cases}$$

Linear separability

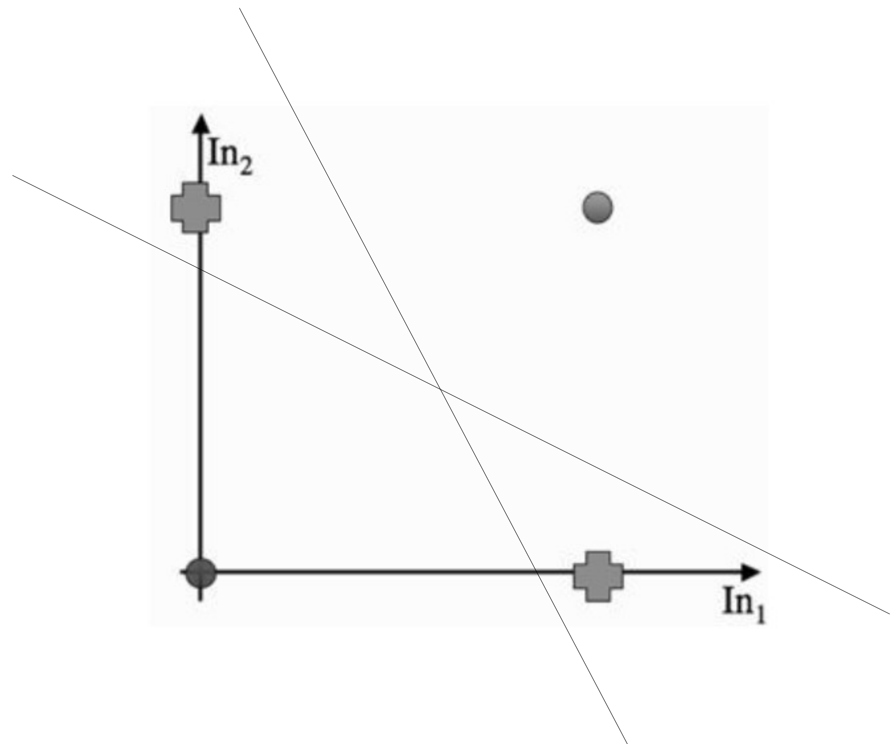
We have established that the decision boundary is a hyperplane.

$$h_w(\boldsymbol{x}) = \boldsymbol{w}^T \boldsymbol{x} + w_0 = 0$$



XOR

Not linearly separable!

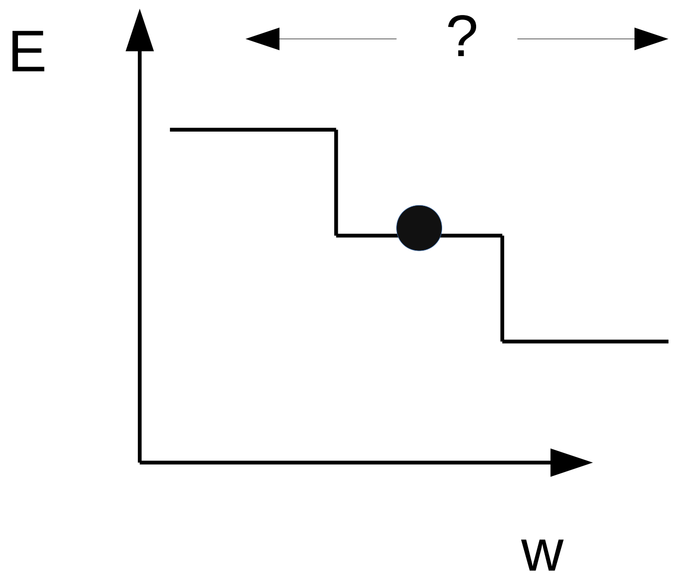


Number of mistakes as error

UNIVERSITY OF LEEDS

$$E(\boldsymbol{X}) = \sum_{\vec{x}_n \in \boldsymbol{X}} |y_n - t_n|$$

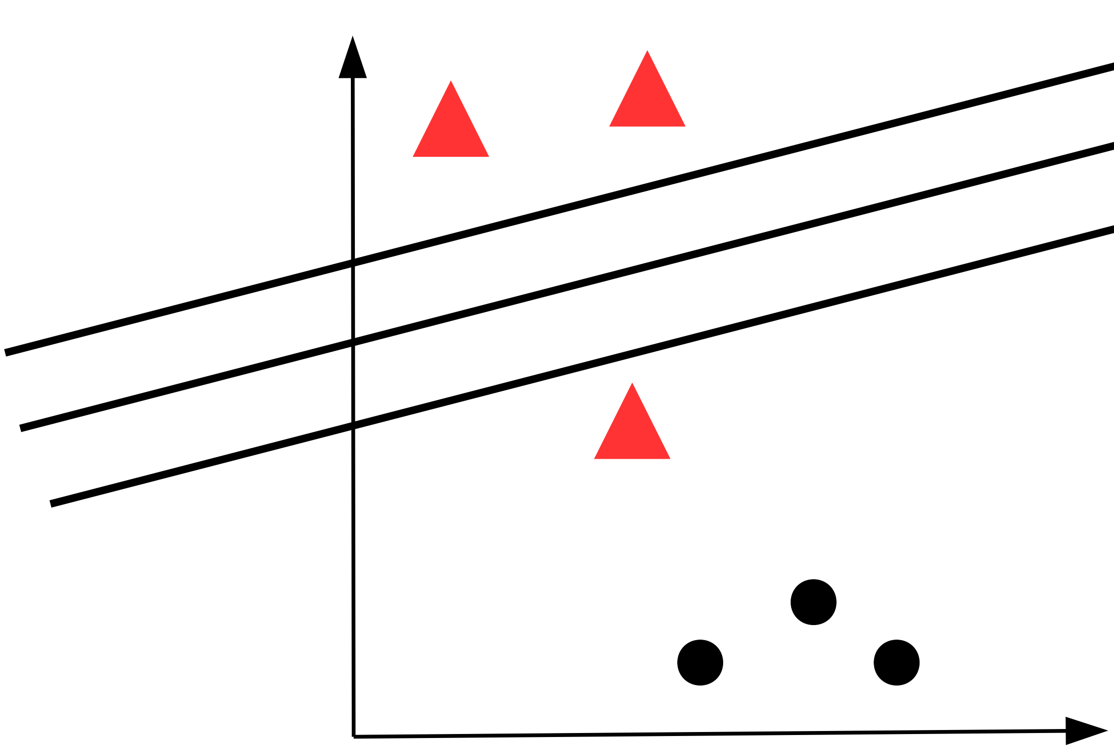
Number of mistakes on the dataset. Piecewise constant → no gradient.



There is no local information on the direction of improvement

Number of mistakes as Error

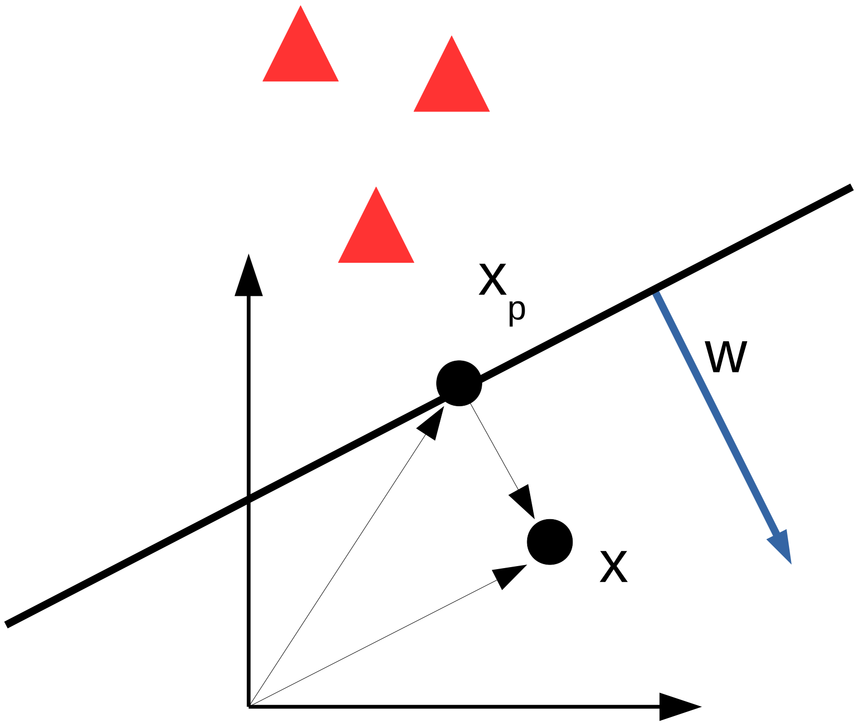
$$h_{\boldsymbol{w}}(\boldsymbol{x}) = \boldsymbol{w}^T \boldsymbol{x} + w_0 = 0$$



Which one is *better*?

Towards a better error function

$$h_{\mathbf{w}}(\mathbf{x}) = \mathbf{w}^T \mathbf{x} + w_0 = 0$$



Distance to the hyperplane

$$\mathbf{x} = \mathbf{x}_p + d\frac{\mathbf{w}}{\|\mathbf{w}\|}$$

$$h_{\mathbf{w}}(\mathbf{x}) = \mathbf{w}\left(\mathbf{x}_p + d\frac{\mathbf{w}}{\|\mathbf{w}\|}\right) + w_0$$

$$= \cancel{\mathbf{w}\,\mathbf{x}_p + w_0} + d\frac{\mathbf{w}^T\mathbf{w}}{\|\mathbf{w}\|} = d\|\mathbf{w}\|$$

Recall that:

$$\mathbf{w}^T\mathbf{w} = w_1^2 + w_2^2 + \cdots + w_n^2 = \|\mathbf{w}\|^2$$

The perceptron criterion

$h_w(\vec{x}) = \vec{w}^T \vec{x} + w_0 = 0$ apply the bias input

if $\boldsymbol{w}^T \boldsymbol{x} > 0$ then $y = 1$ In case of mistake: $t = 0$ $(y - t) = 1$

if $\boldsymbol{w}^T \boldsymbol{x} \leq 0$ then $y = 0$ In case of mistake: $t = 1$ $(y - t) = -1$

Therefore, if mistake: $\boldsymbol{w}^T \boldsymbol{x}(y - t) > 0$

$$E(\boldsymbol{X}) = \sum_{\boldsymbol{x}_n \in \boldsymbol{X}} |y_n - t_n|$$

$$E_p(\boldsymbol{X}) = \sum_{\boldsymbol{x}_n \in \boldsymbol{X}} \boldsymbol{w}^T \boldsymbol{x}_n (y_n - t_n)$$

Number of mistakes on the dataset. Piecewise constant → gradient useless.

Proportional to distance of misclassified points from surface. → gradient ok.

Question

UNIVERSITY OF LEEDS

Given the perceptron error (below), what is the gradient with respect to w?

$$E_p(\boldsymbol{X}) = \boldsymbol{w}^T \boldsymbol{x}(y - t)$$

Solution

$$E_p(\boldsymbol{x}) = \boldsymbol{w}^T \boldsymbol{x}(y-t)$$

$$= w_0 x_0 (y-t) + w_1 x_1 (y-t) + \cdots + w_m x_m (y-t)$$

$$\frac{\partial}{\partial w_0} E_p(\boldsymbol{x}) = \frac{\partial}{\partial w_0} w_0 x_0 (y-t) = x_0 (y-t)$$

$$\frac{\partial}{\partial w_1} E_p(\boldsymbol{x}) = x_1 (y-t)$$

...

$$\frac{\partial}{\partial w_m} E_p(\boldsymbol{x}) = x_m (y-t)$$

Gradient descent

$$\nabla E_p(\boldsymbol{X}) = \sum_{\boldsymbol{x}_n \in \boldsymbol{X}} \boldsymbol{x}_n (y_n - t_n)$$

Recall that gradient descent does the following update:

$$x_{t+1} = x_t - \eta \nabla f(x_t)$$

Which leads us to the update rule for the perceptron:

$$w_{t+1} = w_t - \eta \sum_{\boldsymbol{x}_n \in \boldsymbol{X}} \boldsymbol{x}_n (y_n - t_n)$$

Stochastic gradient descent

$$E_p(\boldsymbol{X}) = \frac{1}{N} \sum_{\boldsymbol{x}_n \in \boldsymbol{X}} \boldsymbol{w}^T \boldsymbol{x}_n (y_n - t_n) = \boldsymbol{E}[\boldsymbol{w}^T \boldsymbol{x}_n (y_n - t_n)]$$

Gradient:

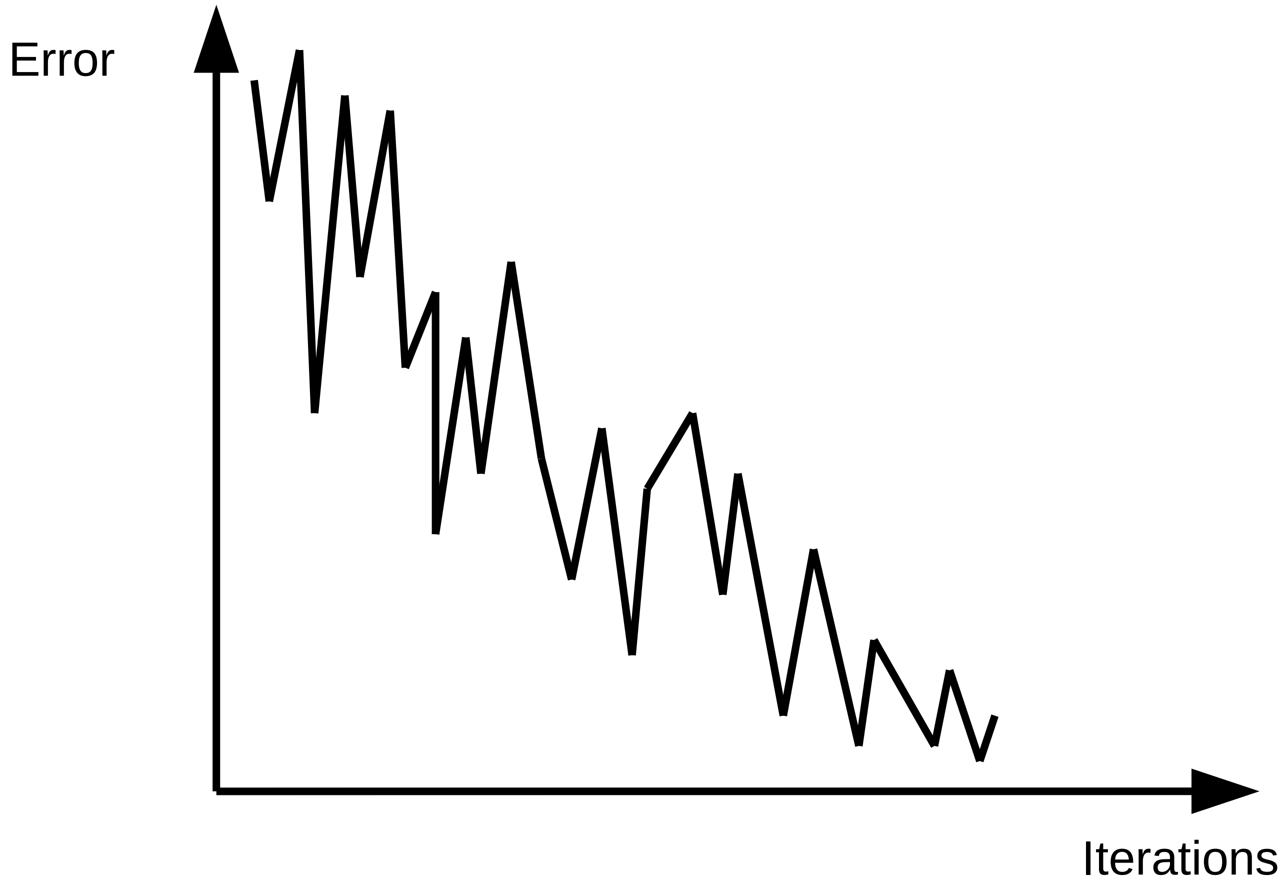
$$\boldsymbol{w} = \boldsymbol{w} - \eta \frac{1}{N} \sum_{\boldsymbol{x}_n \in \boldsymbol{X}} \boldsymbol{x}_n (y_n - t_n)$$

Stochastic Gradient Descent (SGD):

$$\boldsymbol{w} = \boldsymbol{w} - \eta \boldsymbol{x} (y - t)$$

SGD used only one(or a few) data points (x's), to compute the (hence noisy) gradient.

Stochastic gradient descent

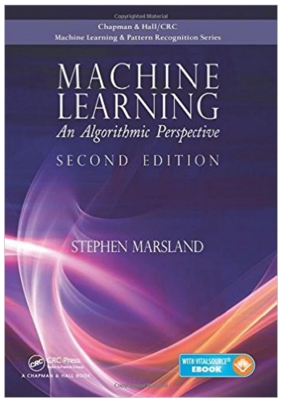


Conclusion

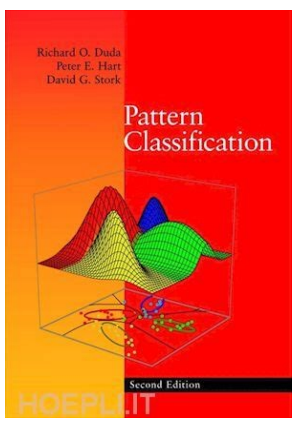
Learning outcomes

UNIVERSITY OF LEEDS

- Define linear separability.
- Justify whether a given error function is suitable for gradient descent.
- Define an appropriate error function for the perceptron.
- Derive the corresponding update algorithm.
- Describe the difference between gradient descent and stochastic gradient descent.



Section 3.4



Book in Minerva
in “ Online Course Readings Folder”

Section 5.2.1, 5.4. and 5.5
(without convergence proof)