

# 1. Feedforward neural networks

## 1.1. The perceptron

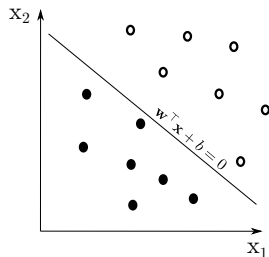
Manel Martínez-Ramón

Meenu Ajith

Aswathy Rajendra Kurup

# The perceptron

- A simplification of the perceptron is a binary classifier.
- Assume a set of observations in a column vector  $\mathbf{x} = \{x_1 \cdots x_D\}^\top$ , that can be arbitrarily labelled as “black” (-1) and “white” (+1) classes.



A classification function can be constructed from a *separating hyperplane* between both classes:

$$\mathbf{w}^\top \mathbf{x} + b = 0 \quad (1)$$

The classifier is defined as

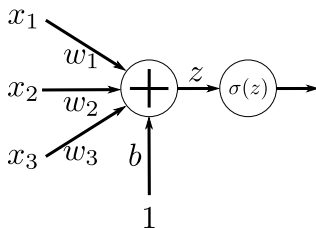
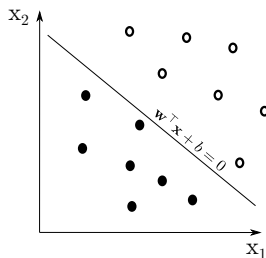
$$f(\mathbf{x}) = \text{sign} \left( \mathbf{w}^\top \mathbf{x} + b \right) \quad (2)$$

# The binary classification function

İç

The classifier is a function with weights  $\mathbf{w}$  and a bias  $b$ , and a generic *activation function*  $\sigma$ :

$$f(\mathbf{x}) = \sigma \left( \mathbf{w}^\top \mathbf{x} + b \right) = \sigma \left( \sum_d w_d x_d + b \right)$$



The separating hyperplane in an example of 2 dimensions (left), and a representation of the classification function for the case of 3 dimensions.

# The Minimum Mean Square Error criterion

- In order to illustrate the idea of MMSE we can take the simplest activation, which is the linear one:

$$f(\mathbf{x}) = \mathbf{w}^\top \mathbf{x} + b$$

- The training criterion is then (MMSE)

$$\min_{\mathbf{w}, b} \mathbb{E} [e_i^2] = \min \mathbb{E} \left[ \left( y_i - \mathbf{w}^\top \mathbf{x}_i - b \right)^2 \right] \quad (3)$$

Of course, the actual expectation cannot be computed, because the probability density functions of the random variables are not available, so the expectation will be approximated by a sample average.

# The Minimum Mean Square Error criterion

- Assuming that  $N$  labelled samples  $\mathbf{x}_i, y_i$  are available, we introduce here matrix  $\mathbf{X} = [\mathbf{x}_1, \dots, \mathbf{x}_N]$  and vector  $\mathbf{y} = [y_1, \dots, y_N]$  containing all training samples and labels.
- We extend both the input data and the weight vector to obtain a compact solution as follows:

$$\mathbf{x} \rightarrow \begin{bmatrix} \mathbf{x} \\ 1 \end{bmatrix}, \quad \mathbf{w} \rightarrow \begin{bmatrix} \mathbf{w} \\ b \end{bmatrix}$$

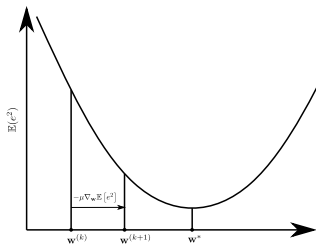
- In this case, nulling the gradient gives

$$\mathbf{w} = \left( \mathbf{X} \mathbf{X}^\top \right)^{-1} \mathbf{X} \mathbf{y} \quad (4)$$

(derivation as an exercise) which is a compact solution, i.e. no iterations needed.

# The Least Mean Square solution

Here we derive a recursive solution. The method is based on a *gradient descent* approach: compute the gradient of the error wrt  $\mathbf{w}$  and move the weights in its opposite direction.



$$\mathbf{w}^{(k+1)} = \mathbf{w}^{(k)} - \mu \nabla_{\mathbf{w}} \mathbb{E}[e^2] \quad (5)$$

where

$$\nabla_{\mathbf{w}} \mathbb{E}[e^2] = \mathbf{X}\mathbf{X}^T \mathbf{w} - \mathbf{X}\mathbf{y} \quad (6)$$

# The Least Mean Square solution

Now we approximate Eq. (6) by using a single sample:

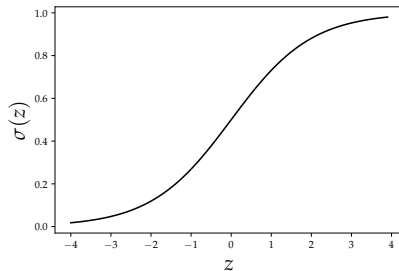
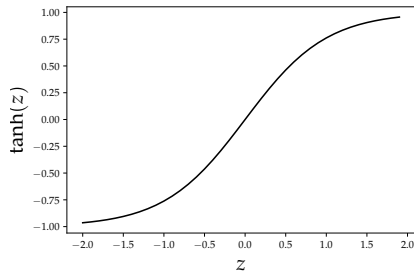
$$\begin{aligned}\nabla_{\mathbf{w}} \mathbb{E} [e^2] &= \mathbf{X}\mathbf{X}^\top \mathbf{w} - \mathbf{X}\mathbf{y} \\ &\approx \mathbf{x}_k \mathbf{x}_k^\top \mathbf{w} - \mathbf{x}_k y_k \\ &= \mathbf{x}_k \left( \mathbf{x}_k^\top \mathbf{w} - y_k \right) \\ &= -e_k \mathbf{x}_k\end{aligned}\tag{7}$$

Where  $e_k = y_k - \mathbf{x}_k^\top \mathbf{w}$ . This leads to the following update rule

$$\mathbf{w}^{(k+1)} = \mathbf{w}^{(k)} + \mu e_k \mathbf{x}_k\tag{8}$$

# Soft activations

- In neural networks the neuron includes an activation  $\sigma(\cdot)$ .
- It can be a *sigmoid* function that produces an output that can be interpreted as a soft state or a probability of a state.



Hyperbolic tangent function (left) and logistic function.



- The logistic function or the hyperbolic tangent are classic neural network activations, but nowadays, the logistic is used in combination with other functions that will be studied further.
- The hyperbolic tangent, the logistic function and their derivatives are:

$$\begin{aligned}\tanh(z) &= \frac{e^z - e^{-z}}{e^z + e^{-z}}, & \frac{d}{dz} \tanh(z) &= 1 - \tanh^2(z) \\ \sigma(z) &= \frac{1}{1 + e^{-z}}, & \frac{d}{dz} \sigma(z) &= \sigma(z) (1 - \sigma(z))\end{aligned}\tag{9}$$

# Soft activations

Now, if the output of the classifier is

$$f(\mathbf{x}) = \tanh \left( \mathbf{w}^\top \mathbf{x}_i + b \right) \quad (10)$$

then the criterion to optimize the parameters will be

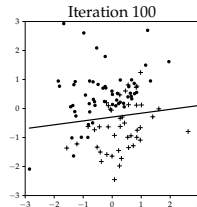
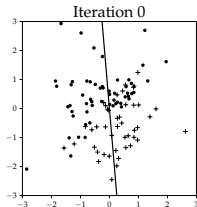
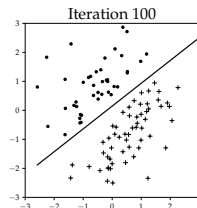
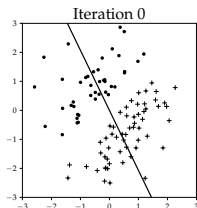
$$\min_{\mathbf{w}, b} \mathbb{E} [e_i^2] \approx \min_{\mathbf{w}, b} \sum_{i=1}^N \left( y_i - \tanh \left( \mathbf{w}^\top \mathbf{x}_i + b \right) \right)^2 \quad (11)$$

which leads to the update rule

$$\begin{aligned} \mathbf{w}^{(k+1)} &= \mathbf{w}^{(k)} + \mu \sum_{i=1}^N e_i (1 - f^2(\mathbf{x}_i)) \mathbf{x}_i \\ b^{(k+1)} &= b^{(k)} + \mu \sum_{i=1}^N e_i (1 - f^2(\mathbf{x}_i)) \end{aligned} \quad (12)$$

The proof is left as an exercise.

# Example of the MMSE applied to a perceptron



Example of the application of the MMSE criterion to a perceptron with hyperbolic tangent activation. The first row corresponds to a separable case and the second one to a non separable one. In both cases the algorithm converges to a solution.