UNIVERSITY OF LEEDS

Class: Machine Learning

Neural Networks: Perceptron

Instructor: Matteo Leonetti

Learning outcomes

UNIVERSITY OF LEEDS

- Define linear separability.
- Justify whether a given error function is suitable for gradient descent.

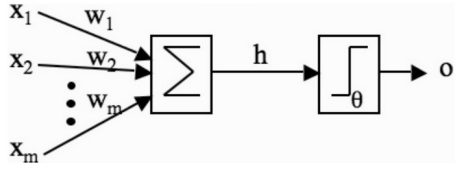
Recap

UNIVERSITY OF LEEDS

We want to apply gradient descent:

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

To the parameters of a perceptron:

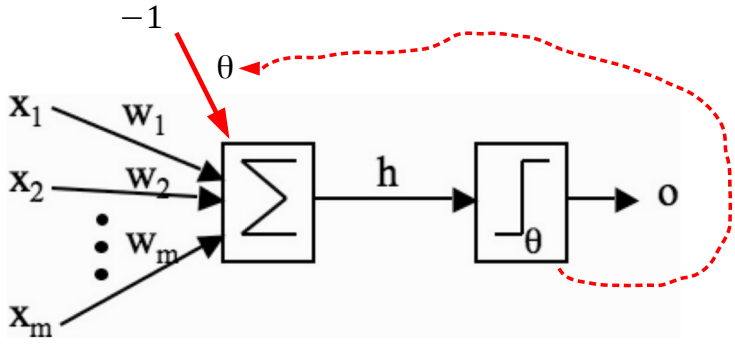


So as to minimise an error (or loss) function, such as:

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

Bias input

UNIVERSITY OF LEEDS



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

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

$$w_1 x_1 + w_2 x_2 + \cdots + w_n x_n - 1 \cdot \theta > 0$$

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

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

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

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

We want to optimise the parameters of the perceptron through gradient descent.

For this purpose, we will need to compute the derivative with respect to the parameters. With the weights this is simple (as we will see shortly). However, the threshold of the activation function is in an awkward position.

Nonetheless, we can simply move the threshold on the other side of the inequality of the activation function, and treat it as it if were just a normal weight, associated with an additional input, called the *bias* input.

The bias input must be fixed at a constant. The value of the constant is not important, since the parameter theta can adjust accordingly. In books, you will usually find either 1 or -1.

Question

UNIVERSITY OF LEEDS

The decision boundary of the perceptron is the function below:

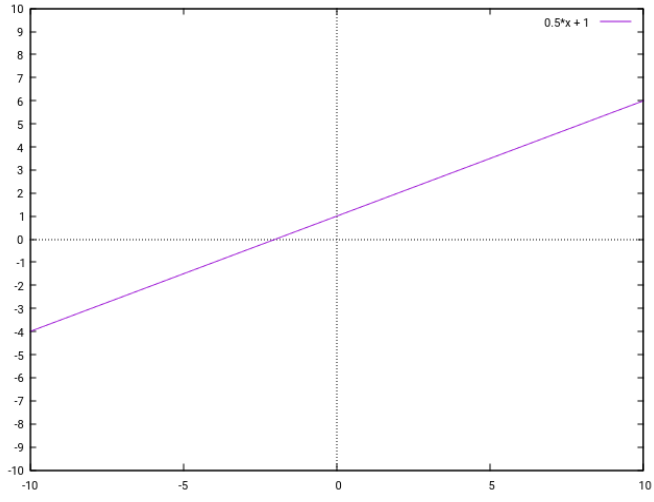
$$f(\boldsymbol{x}) = \boldsymbol{w} \cdot \boldsymbol{x} = 0$$

Plot the following function: $\frac{1}{2}x - y + 1 = 0$

Solution

UNIVERSITY OF LEEDS

Plot the following function: $\frac{1}{2}x - y + 1 = 0$

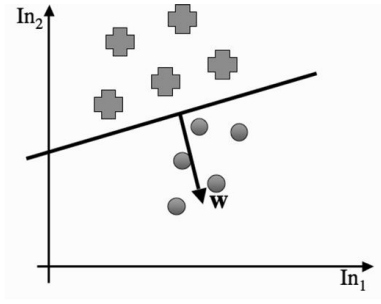


Linear separability

UNIVERSITY OF LEEDS

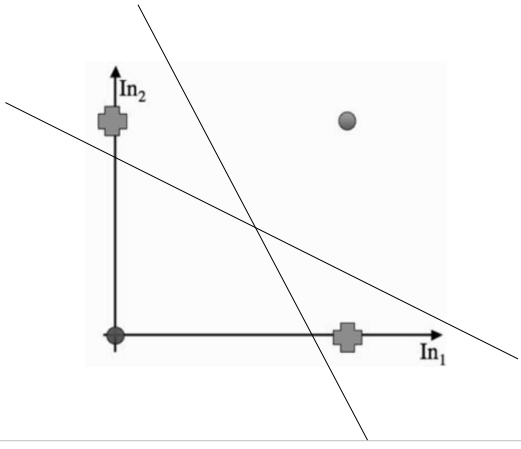
We have established that the decision boundary is a hyperplane.

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



XOR

Not linearly separable!



The decision boundary of the perceptron is a hyperplane (in 2D, a straight line).

The vector of weights (excluding the bias weight) is a vector orthogonal to the hyperplane, and plays an important role.

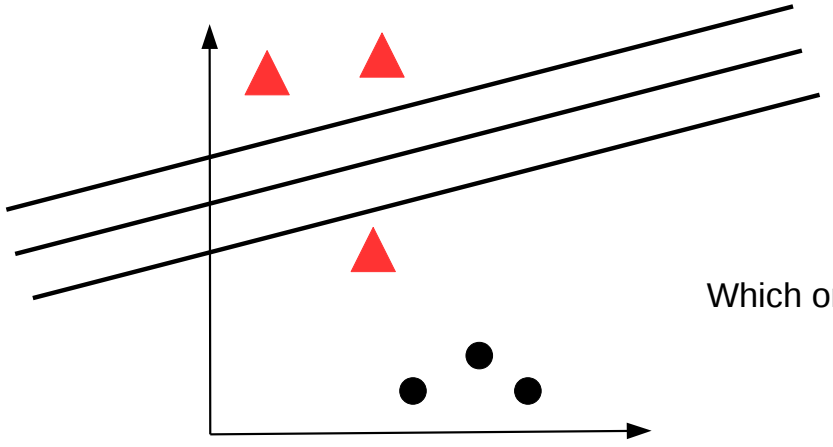
If the dataset is such that an hyperplane that separates the two classes exists, it is said to be linearly separable.

The logic function XOR over points in {0,1}x{0,1} is a well-known example of a non-linearly separable dataset.

Number of mistakes as Error

UNIVERSITY OF LEEDS

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



Which one is *better*?

If we use the number of mistakes as an error function, all the lines drawn here will have the same error (since they all make the same number of mistakes).

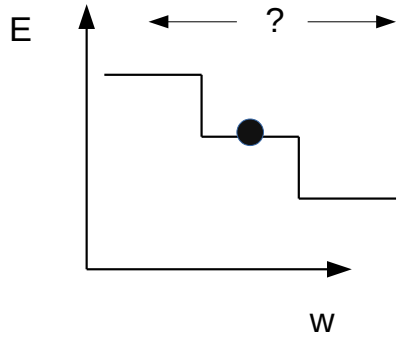
Therefore, it is not possible to use the number of mistakes to inform the search with a local optimization method. Most local steps achieve no difference in the objective (in our case the error) function, and it is impossible to know whether the algorithm is progressing towards a minimum or not.

Number of mistakes

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.

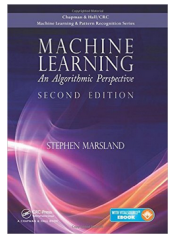


There is no local information on the direction of improvement

Rather than this function, which is piecewise constant, we would like an error function in which the error decreases as the current solution gets closer to a misclassified point.

Conclusion

UNIVERSITY OF LEEDS



Section 3.3 & 3.4