Neural Networks

2. The Perceptron

Center for Cognitive Science
Department of Applied Informatics
Faculty of Mathematics, Physics and Informatics
Comenius University in Bratislava

Thursday 29th February, 2024

Types of Learning (recap)

- **supervised** - learning with teacher
- **unsupervised** - self organisation
- **reinforcement** - no immediate feedback

- + hybrid methods

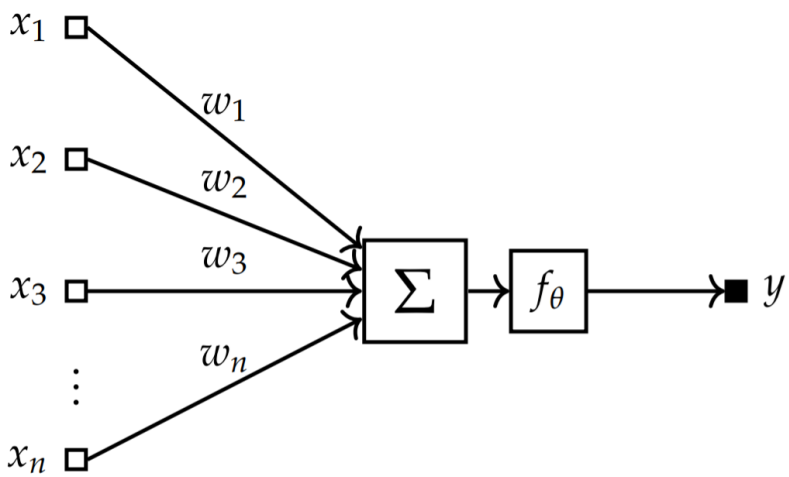
Error-Driven Supervised Learning

General scheme:

- repeat until some criterion (# of epochs, success, ...):
  - for each input $\boldsymbol{x}$ and desired output $\boldsymbol{d}$ (random order):
    - ▷ compute output: $\boldsymbol{y} = ?(\boldsymbol{W}\boldsymbol{x})$
    - ▷ compute error: $\boldsymbol{e} = \boldsymbol{d} - \boldsymbol{y}$
    - ▷ compute adjustment: $\Delta \boldsymbol{W} = ?(\boldsymbol{W}, \boldsymbol{x}, \boldsymbol{y}, \boldsymbol{e})$
    - ▷ adjust weights: $\boldsymbol{W} := \boldsymbol{W} + \alpha \Delta \boldsymbol{W}$

$$\boldsymbol{W}(t+1) = \boldsymbol{W}(t) + \alpha \Delta \boldsymbol{W}$$

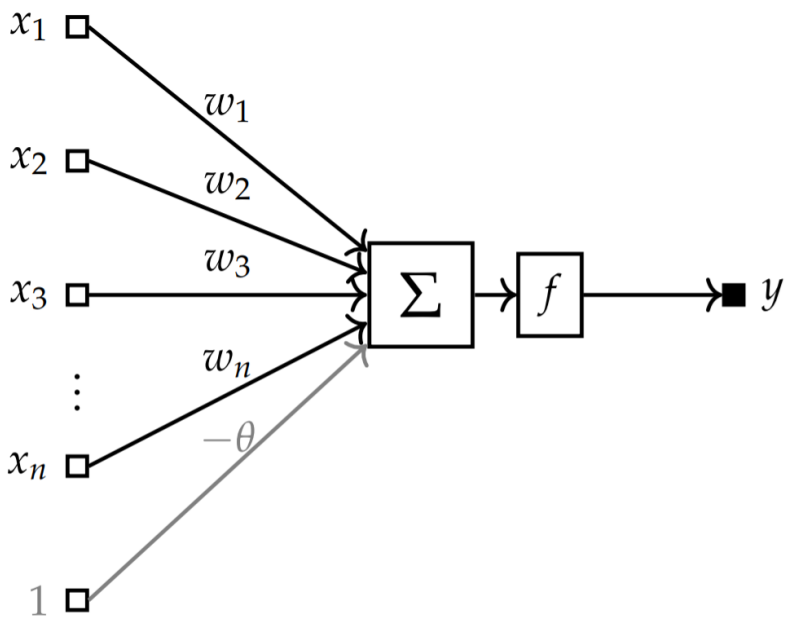
The Perceptron



- net input: $net = \boldsymbol{w} \cdot \boldsymbol{x} = \sum_{i=1}^{n} x_i w_i$
- output: $y = f_\theta(net)$
- activation function for discrete perceptron (θ = treshold)

$$f_\theta(net) = \begin{cases} 1, & net \geq \theta \\ 0, & net < \theta \end{cases}$$

The Perceptron: Threshold "Input" – Bias



- virtual input for the threshold term: $x'_{n+1} = 1$
- threshold as weight: $w'_{n+1} = -\theta$
- simplified activation - "step function":

$$f(net) = \begin{cases} 1, & net \geq 0 \\ 0, & net < 0 \end{cases}$$

The Perceptron: Error-Driven Learning

- input: $\boldsymbol{x} \in \mathbb{R}^n$
  - augmented input $\boldsymbol{x}' \in \mathbb{R}^{n+1}$
  - $\boldsymbol{x}' = \text{add\_bias}(\boldsymbol{x})$
- weights: $\boldsymbol{w} \in \mathbb{R}^n$
  - including threshold $\boldsymbol{w}' \in \mathbb{R}^{n+1}$
  - are initialized randomly
- output $y \in \{0, 1\}$

  $$y = f(\boldsymbol{w}' \cdot \boldsymbol{x}') = f(\sum_{i=1}^{n+1} x_i' w_i')$$
- target $d \in \{0, 1\}$ is given for all data points
- weight adjustment
  - $\boldsymbol{w}'(t+1) = \boldsymbol{w}'(t) + \alpha(d - y)\boldsymbol{x}'$

Task: Linear separation using perceptron

- complete the missing lines in perceptron.py according to theory in these slides.
- C02.py is the main file, algorithm starts after running this file.
- resulting perceptron should be trained to separate two classes of dots with a single line (because the classes are linearly separable).