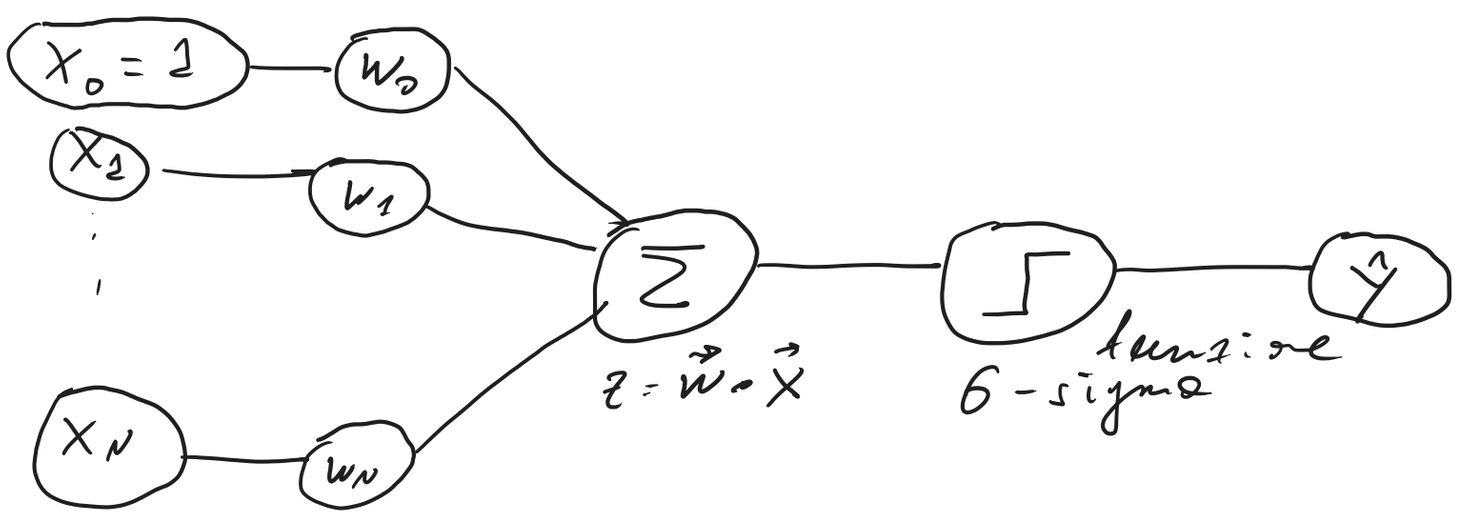
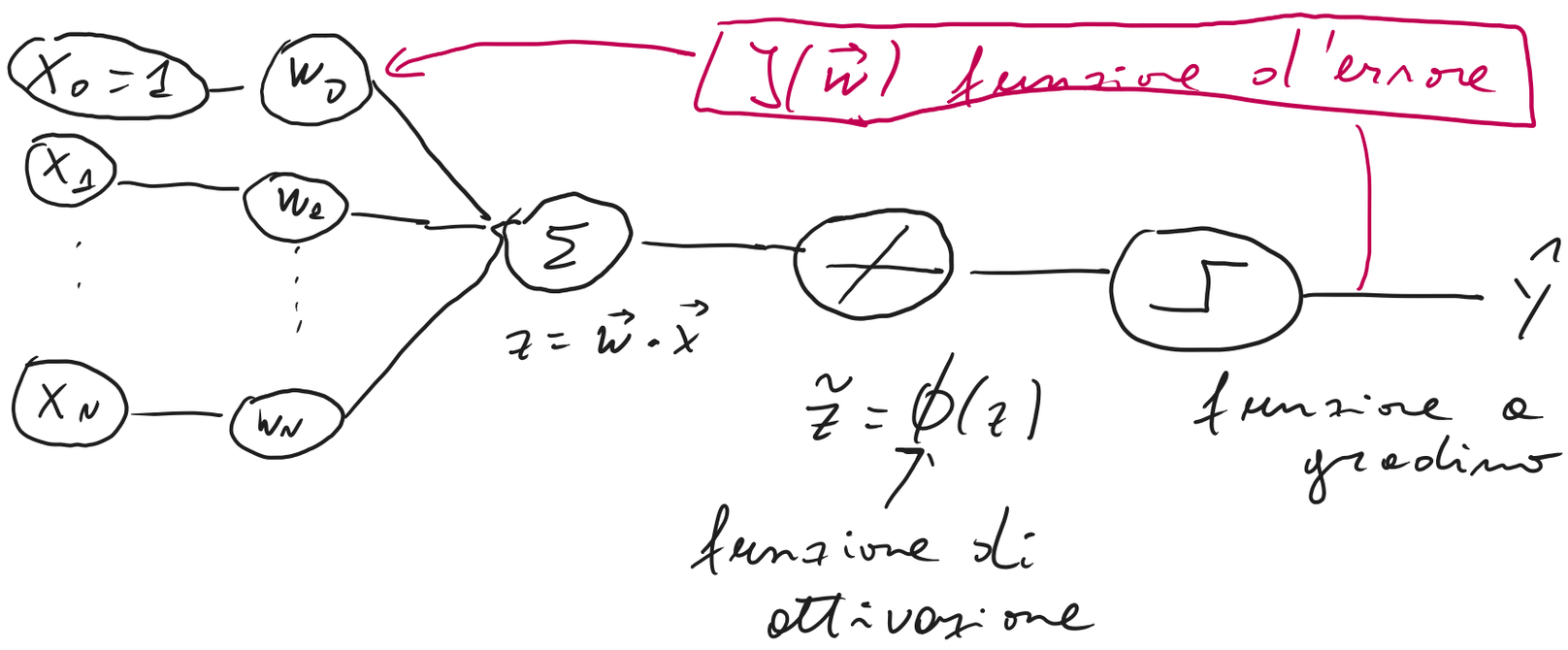
Perceptron



ADECINE (Adaptive linear neuron)



NB in adaline

$$\phi(z) = z$$

$$J(\vec{w}) = \frac{1}{2} \sum_{i=1, N_{sample}} \left(y^i - \phi(\vec{w} \cdot \vec{x}^i) \right)^2$$

con $\vec{x}^i$ features dell'
i-esimo
sample

$$\frac{\partial J}{\partial w_j} = \frac{\partial}{\partial w_j} \frac{1}{2} \sum_{i=1, N_{samples}} \left(y^i - \vec{w} \cdot \vec{x}^i \right)^2$$

$$= \sum_{i=1, N_{samples}} -\left(y^i - \vec{w} \cdot \vec{x}^i \right) x_j^i$$

Algoritmo iterativo — piccolo intero positivo

$$\vec{w} = \vec{w} - \eta \vec{\nabla} J$$

Stochastic gradient descent (uso ad ogni ciclo set ridotto di samples: determinato casualmente.

Se ho più classi uso la strategia OVE VERSUS ALL (REST) OVA o OVR

STANDARDIZZAZIONE

Se la feature j-esima ha valore medio μ_j e deviazione standard σ_j

$$x'_j = \frac{x_j - \mu_j}{\sigma_j}$$