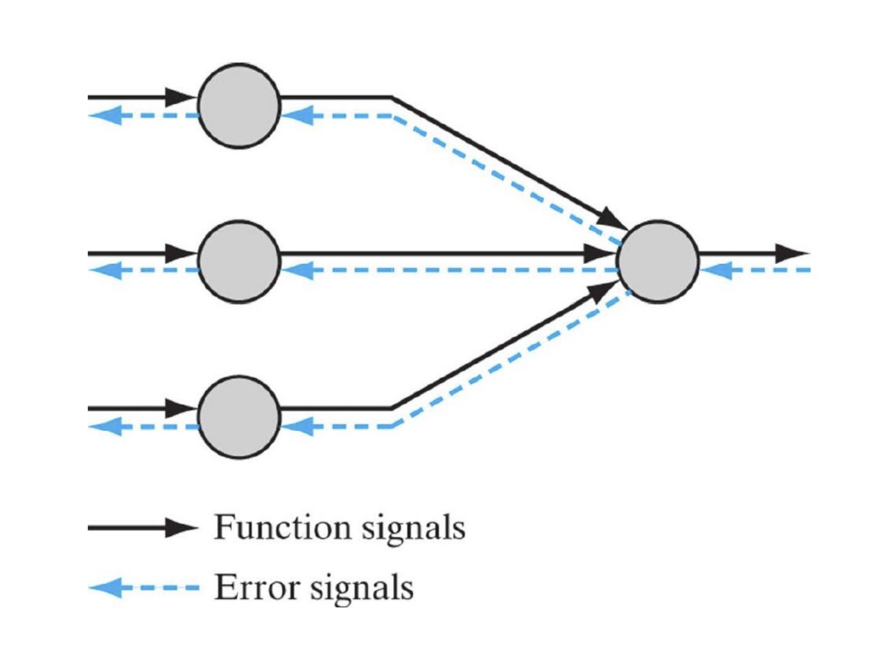
Back-Propagation + Example

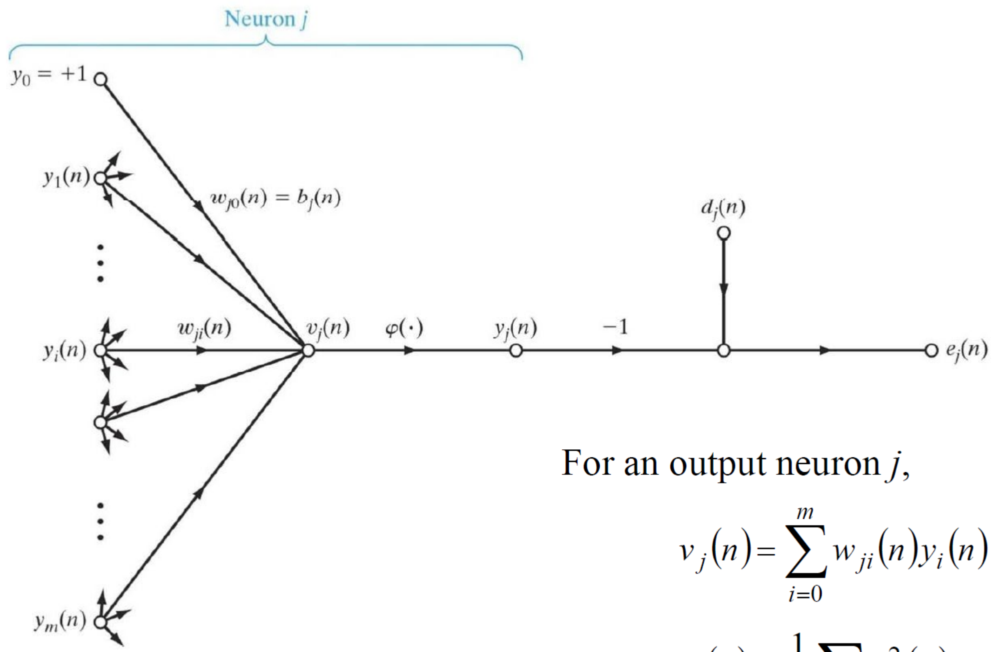
Training Procedure

- Forward Pass
- Error Function
- Backward Pass



- Reference: Simon Haykin, *Neural Networks and Learning Machines*

Back-Propagation Algorithm for Output Neuron *j*



For an output neuron *j*,

$$v_j(n) = \sum_{i=0}^{m} w_{ji}(n) y_i(n) \qquad y_j(n) = \varphi(v_j(n))$$

$$\mathcal{E}(n) = \frac{1}{2}\sum_j e_j^2(n) = \frac{1}{2}\sum_j \left[d_j(n) - y_j(n)\right]^2$$

$$\frac{\partial \mathcal{E}(n)}{\partial w_{ji}(n)} = \frac{\partial \mathcal{E}(n)}{\partial e_j(n)} \frac{\partial e_j(n)}{\partial y_j(n)} \frac{\partial y_j(n)}{\partial v_j(n)} \frac{\partial v_j(n)}{\partial w_{ji}(n)}$$

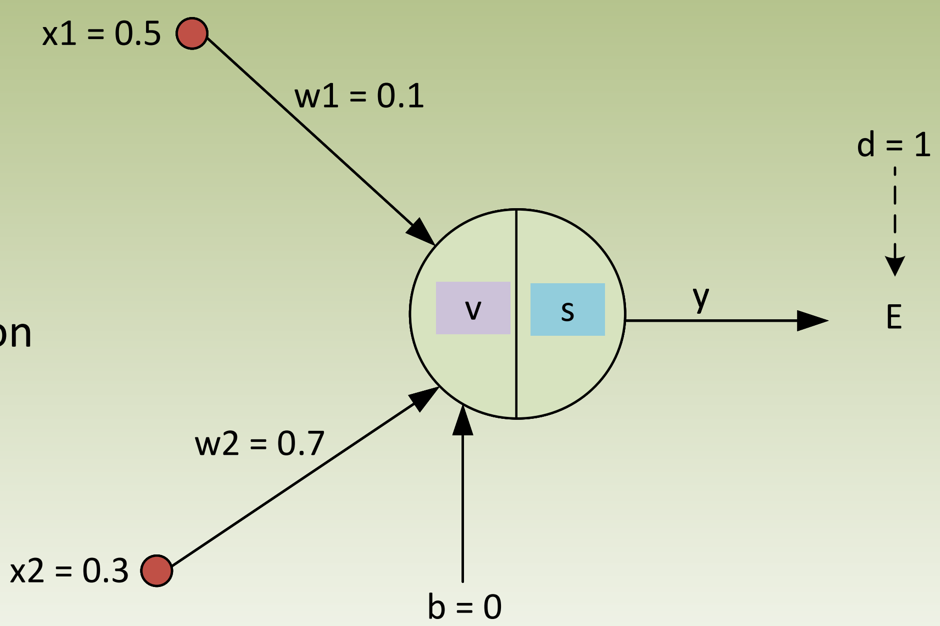
$$= -e_j(n)\varphi'(v_j(n)) y_i(n)$$

$$\Delta w_{ji}(n) = -\eta \frac{\partial \mathcal{E}(n)}{\partial w_{ji}(n)} = \eta \delta_j(n) y_i(n) \quad \text{where } \delta_j(n) = e_j(n)\varphi'(v_j(n))$$

Numerical Example

- Inputs: x_1, x_2
- Summing Junction:
  - $v = w_1 x_1 + w_2 x_2 + b$
- Output of Activation Function
  - $y = s(v) = 1/(1 + e^{-v})$

    Where $s(v)$ is a sigmoid function



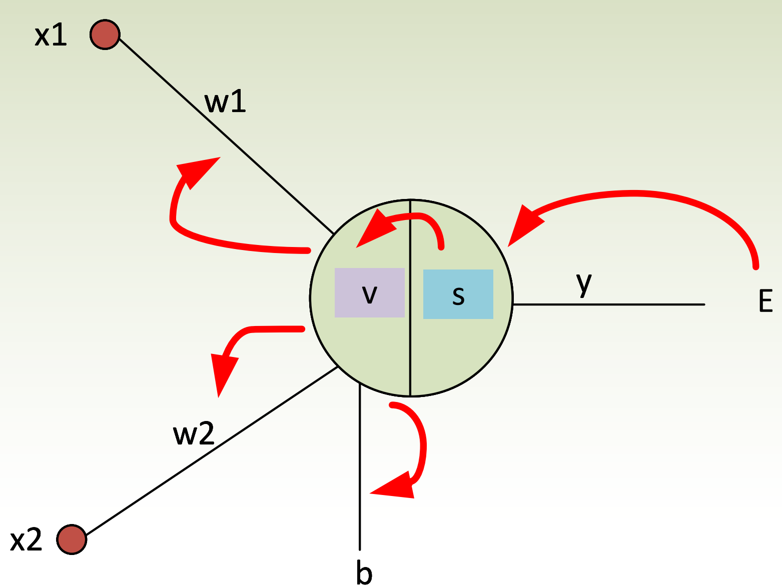
- Error: $E = \frac{1}{2}(d - y)^2$
  - Where d is the true class label

Forward Pass & Error Function

- Forward Pass
  - $v = w_1 x_1 + w_2 x_2 + b = (0.1)(0.5) + (0.7)(0.3) + 0 = 0.26$
  - $y = s(v) = 1/(1 + e^{-v}) = 1/\left(1 + e^{-0.26}\right) = 0.5646$
- Error Function
  - $E = \frac{1}{2}(d - y)^2 = \frac{1}{2}(1 - 0.5646)^2 = 0.0948$

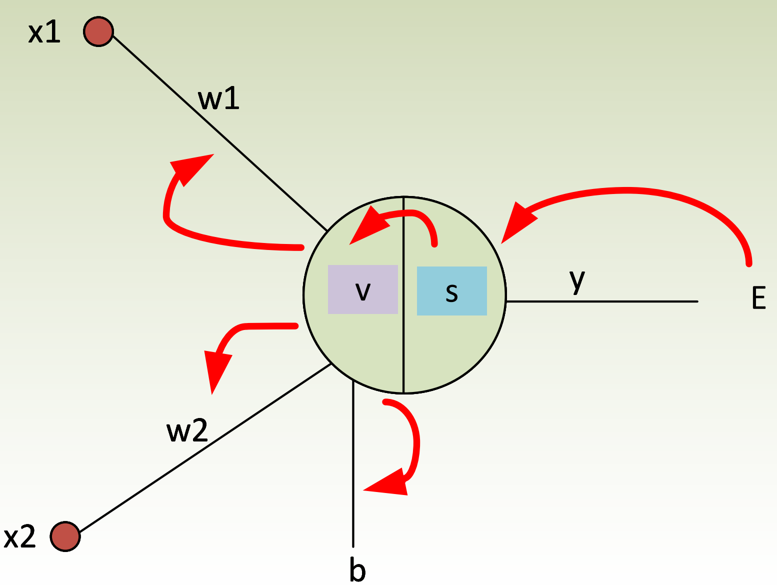
Backward Pass

- Propagate the error backward to correct the weight parameters
- $w_{ji} = w_{ji} + \Delta w_{ji}$
  - $\begin{pmatrix} weight \\ correction \\ \Delta w_{ji} \end{pmatrix} = \begin{pmatrix} learning \\ rate \\ \eta \end{pmatrix} \times \begin{pmatrix} local \\ gradient \end{pmatrix} \times \begin{pmatrix} input\ signal \\ of\ neuron\ j \end{pmatrix}$
  - a.k.a. Delta Rule



Backward Pass (continued)

- Specifically for our example:
  - $w_1 = w_1 + \Delta w_1$
  - $w_2 = w_2 + \Delta w_2$
  - $b = b + \Delta w_b$



Backward Pass (continued)

- Consider $w_1 = w_1 + \Delta w_1$
- First determine $\frac{\partial E}{\partial w_1}$ → use Chain Rule: $\frac{\partial E}{\partial w_1} = \frac{\partial E}{\partial y} . \frac{\partial y}{\partial v} . \frac{\partial v}{\partial w_1}$
- Where $\frac{\partial E}{\partial y} . \frac{\partial y}{\partial v}$ is the local gradient δ_1
  - $E = \frac{1}{2}(d-y)^2 \rightarrow \frac{\partial E}{\partial y} = -(d-y)$
  - $y = s(v) = 1/(1+e^{-v}) \rightarrow \frac{\partial y}{\partial v} = s(v)' = s(v)\big(1-s(v)\big) = y(1-y)$
  - $v = w_1 x_1 + w_2 x_2 + b \rightarrow \frac{\partial v}{\partial w_1} = x_1$

- $\Delta w_1 = -\eta \frac{\partial E}{\partial w_1} = \eta(d-y)\big(y(1-y)\big)x_1 = \eta \delta_1 x_1$
  - $\begin{pmatrix} weight \\ correction \\ \Delta w_1 \end{pmatrix} = \begin{pmatrix} learning \\ rate \\ \eta \end{pmatrix} \times \begin{pmatrix} local \\ gradient \\ \delta_1 \end{pmatrix} \times \begin{pmatrix} input\ signal \\ of\ output\ neuron \\ x_1 \end{pmatrix}$

Backward Pass (continued)

- Hence,
  - $w_1 = w_1 + \eta(d - y)\big(y(1 - y)\big)x_1$
  - $w_2 = w_2 + \eta(d - y)\big(y(1 - y)\big)x_2$
  - $b = b + \eta(d - y)\big(y(1 - y)\big)$

- Training example: $x_1 = 0.5, x_2 = 0.3, d = 1$
- Learning rate: $\eta = 0.9$
- Output at time t: $y = 0.5646$
- Weight parameters at time t:
  - $w_1 = 0.1, w_2 = 0.7, b = 0$

- Updated weight parameters at time $t + 1$:
  - $w_1 = 0.1482, w_2 = 0.7000, b = 0.0963$