- $\mathbf{x}$ or $\mathbf{x_i}$ is a data point (with corresponding target $t$ or $t_i$ in the training and test sets). $i \in [1, N]$ or $i \in [0, N-1)$ ranges over data.
- $\mathbf{w}$ is a weight vector. When disambiguation is needed, $\mathbf{w_k}$ is the weight vector of the $k$th perceptron.
- $w_j$ (or $w_{kj}$) is the $j$th weight and $x_j$ (or $x_{ij}$) is the $j$th component in an input vector. $D$ is the dimensionality of the input and $j \in [0, D]$ for weights, but $j \in [1, D]$ for input vectors (or $x_0 = 1$).
- $a(\mathbf{x}) = \mathbf{w} \cdot \mathbf{x} = \sum_{j=0}^{D} w_j x_j = w_0 + \sum_{j=1}^{D} w_j x_j$ is an *unthresholded perceptron*, *linear unit*, or *activation* (see Bishop pg 227).
- $h$ is an *activation function*, which effectively provides a threshold for a perceptron.
- $z(\mathbf{x}) = h(a(\mathbf{x}))$ is a *perceptron*. (Bishop pg 227 calls this a *hidden unit*, which makes sense in the context of a MLP).
- $k \in [1, M]$ ranges over perceptrons in a hidden layer, hence $z_i$, $a_k$, and $\mathbf{w_k}$ and $w_{kj}$.
- $\eta$ is the *learning rate*.

**Perceptron rule**

Initialize $\mathbf{w}$ to random values

Repeat until all training data points are correctly classified

  For each data point $\mathbf{x_i}, t_i$

    Compute $z(\mathbf{x_i}) = h(\mathbf{w} \cdot \mathbf{x_i})$

    For each weight $w_j$

      $w_j += \eta(t_i - z(\mathbf{x_i}))x_{ij}$

**Gradient descent**

Initialize $\mathbf{w}$ to random values

Repeat until termination condition

  For each $\Delta w_j$

    $\Delta w_j = 0$

  For each data point $\mathbf{x_i}, t_i$

    Compute $a(\mathbf{x_i}) = \mathbf{w} \cdot \mathbf{x_i}$

    For each $\Delta w_j$

      $\Delta w_j += \eta(t_i - a(\mathbf{x_i}))x_{ij}$

  For each weight $w_j$

    $w_j += \Delta w_j$

**Stochastic gradient descent (delta rule)**

Initialize $\mathbf{w}$ to random values

Repeat until termination condition

For each data point $\mathbf{x_i}, t_i$

Compute $a(\mathbf{x_i}) = \mathbf{w} \cdot \mathbf{x_i}$

For each weight $w_j$

$w_j += \eta(t_i - a(\mathbf{x_i}))x_{ij}$

**Backpropagation**

Initialize all weights in all units to random value

Repeat until termination condition

For each data point $\mathbf{x_i}, t_i$

Compute $z_k$ and $y_\ell$ for every unit in the network

For each output unit $y_\ell$

$$\delta_{y_\ell} = y_\ell(\mathbf{x_i})(1 - y_\ell(\mathbf{x_i}))(t_i - y_\ell(\mathbf{x_i}))$$

For each hidden unit $z_k$

$$\delta_{z_k} = z_k(\mathbf{x_i})(1 - z_k(\mathbf{x_i})) \sum_{\ell=1}^{K} w_{\ell k}\delta_\ell$$

For each output unit $y_\ell$

For each weight $w_{y_\ell j}$

$$w_{y_\ell j} = \eta \delta_{y_\ell} x_{ij}$$

For each hidden unit $z_k$

For each weight $w_{z_k j}$

$$w_{z_k j} = \eta \delta_{z_k} x_{ij}$$