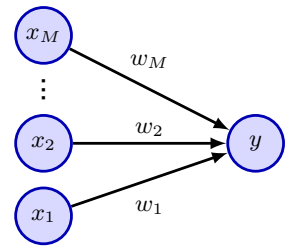

Figure 1.13 A simple neural network diagram representing the transformations (1.5) and (1.6) describing a single neuron. The polynomial function (1.1) can be seen as a special case of this model.

can be expressed mathematically in the form

$$a = \sum_{i=1}^{M} w_i x_i \tag{1.5}$$

$$y = f(a) \tag{1.6}$$

where $x_1, \ldots, x_M$ represent M inputs corresponding to the activities of other neurons that send connections to this neuron, and $w_1, \ldots, w_M$ are continuous variables, called *weights*, which represent the strengths of the associated synapses. The quantity a is called the *pre-activation*, the nonlinear function $f(\cdot)$ is called the *activation function*, and the output y is called the *activation*. We can see that the polynomial (1.1) can be viewed as a specific instance of this representation in which the inputs x_i are given by powers of a single variable x, and the function $f(\cdot)$ is just the identity $f(a) = a$. The simple mathematical formulation given by (1.5) and (1.6) has formed the basis of neural network models from the 1960s up to the present day, and can be represented in diagram form as shown in Figure 1.13.

1.3.1 Single-layer networks

The history of artificial neural networks can broadly be divided into three distinct phases according to the level of sophistication of the networks as measured by the number of 'layers' of processing. A simple neural model described by (1.5) and (1.6) can be viewed as having a single layer of processing corresponding to the single layer of connections in Figure 1.13. One of the most important such models in the history of neural computing is the *perceptron* (Rosenblatt, 1962) in which the activation function $f(\cdot)$ is a step function of the form

$$f(a) = \begin{cases} 0, & \text{if } a \leqslant 0, \\ 1, & \text{if } a > 0\,. \end{cases} \tag{1.7}$$

This can be viewed as a simplified model of neural firing in which a neuron fires if, and only if, the total weighted input exceeds a threshold of 0. The perceptron was pioneered by Rosenblatt (1962), who developed a specific training algorithm that has the interesting property that if there exists a set of weight values for which the perceptron can achieve perfect classification of its training data then the algorithm is guaranteed to find the solution in a finite number of steps (Bishop, 2006). As well as a learning algorithm, the perceptron also had a dedicated analogue hardware