6.2. Multilayer Networks

In the previous section, we saw that to apply linear models of the form (6.1) to problems involving large-scale data sets and high-dimensional spaces, we need to find a set of basis functions that is tuned to the problem being solved. The key idea behind neural networks is to choose basis functions $\phi_j(\mathbf{x})$ that themselves have learnable parameters and then allow these parameters to be adjusted, along with the coefficients $\{w_j\}$, during training. We then optimize the whole model by minimizing an error function using gradient-based optimization methods, such as stochastic gradient descent, where the error function is defined jointly across all the parameters in the model.

Chapter 7

There are, of course, many ways to construct parametric nonlinear basis functions. One key requirement is that they must be differentiable functions of their learnable parameters so that we can apply gradient-based optimization. The most successful choice has been to use basis functions that follow the same form as (6.1), so that each basis function is itself a nonlinear function of a linear combination of the inputs, where the coefficients in the linear combination are learnable parameters. Note that this construction can clearly be extended recursively to give a hierarchical model with many layers, which forms the basis for deep neural networks.

Section 6.3

Consider a basic neural network model having two layers of learnable parameters. First, we construct M linear combinations of the input variables $x_1, \ldots, x_D$ in the form

$$a_j^{(1)} = \sum_{i=1}^{D} w_{ji}^{(1)} x_i + w_{j0}^{(1)} \tag{6.7}$$

where $j = 1, \ldots, M$, and the superscript (1) indicates that the corresponding parameters are in the first 'layer' of the network. We will refer to the parameters $w_{ji}^{(1)}$ as *weights* and the parameters $w_{j0}^{(1)}$ as *biases*, while the quantities $a_j^{(1)}$ are called *pre-activations*. Each of the quantities a_j is then transformed using a differentiable, nonlinear *activation function* $h(\cdot)$ to give

Chapter 4

$$z_j^{(1)} = h(a_j^{(1)}), \tag{6.8}$$

which represent the outputs of the basis functions in (6.1). In the context of neural networks, these basis functions are called *hidden units*. We will explore various choices for the nonlinear function $h(\cdot)$ shortly, but here we note that provided the derivative $h'(\cdot)$ can be evaluated, then the overall network function will be differentiable. Following (6.1), these values are again linearly combined to give

$$a_k^{(2)} = \sum_{j=1}^{M} w_{kj}^{(2)} z_j^{(1)} + w_{k0}^{(2)} \tag{6.9}$$

where $k = 1, \ldots, K$, and K is the total number of outputs. This transformation corresponds to the second layer of the network, and again the $w_{k0}^{(2)}$ are bias parameters. Finally, the $\{a_k^{(2)}\}$ are transformed using an appropriate output-unit activation