unit pre-activations denoted by a_k^{π} that determine the mixing coefficients $\pi_k(\mathbf{x})$, K outputs denoted by a_k^{σ} that determine the Gaussian standard deviations $\sigma_k(\mathbf{x})$, and $K \times L$ outputs denoted by a_{kj}^{μ} that determine the components $\mu_{kj}(\mathbf{x})$ of the Gaussian means $\boldsymbol{\mu}_k(\mathbf{x})$. The total number of network outputs is given by $(L+2)K$, unlike the usual L outputs for a network that simply predicts the conditional means of the target variables.

The mixing coefficients must satisfy the constraints

$$\sum_{k=1}^{K} \pi_k(\mathbf{x}) = 1, \qquad 0 \leqslant \pi_k(\mathbf{x}) \leqslant 1, \tag{6.39}$$

which can be achieved using a set of softmax outputs:

$$\pi_k(\mathbf{x}) = \frac{\exp(a_k^{\pi})}{\sum_{l=1}^{K} \exp(a_l^{\pi})}. \tag{6.40}$$

Similarly, the variances must satisfy $\sigma_k^2(\mathbf{x}) \geqslant 0$ and so can be represented in terms of the exponentials of the corresponding network pre-activations using

$$\sigma_k(\mathbf{x}) = \exp(a_k^{\sigma}). \tag{6.41}$$

Finally, because the means $\boldsymbol{\mu}_k(\mathbf{x})$ have real components, they can be represented directly by the network outputs:

$$\mu_{kj}(\mathbf{x}) = a_{kj}^{\mu} \tag{6.42}$$

in which the output-unit activation functions are given by the identity $f(a) = a$.

The learnable parameters of the mixture density network comprise the vector $\mathbf{w}$ of weights and biases in the neural network, which can be set by maximum likelihood or equivalently by minimizing an error function defined to be the negative logarithm of the likelihood. For independent data, this error function takes the form

$$E(\mathbf{w}) = -\sum_{n=1}^{N} \ln \left\{ \sum_{k=1}^{K} \pi_k(\mathbf{x}_n, \mathbf{w}) \mathcal{N}\left(\mathbf{t}_n | \boldsymbol{\mu}_k(\mathbf{x}_n, \mathbf{w}), \sigma_k^2(\mathbf{x}_n, \mathbf{w})\right) \right\} \tag{6.43}$$

where we have made the dependencies on $\mathbf{w}$ explicit.

6.5.3 Gradient optimization

To minimize the error function, we need to calculate the derivatives of the error $E(\mathbf{w})$ with respect to the components of $\mathbf{w}$. We will see later how to compute these derivatives automatically. It is instructive, however, to derive suitable expressions for the derivatives of the error with respect to the output-unit pre-activations explicitly as this highlights the probabilistic interpretation of these quantities. Because the error function (6.43) is composed of a sum of terms, one for each training data point, we can consider the derivatives for a particular input vector $\mathbf{x}_n$ with associated target vector $\mathbf{t}_n$. The derivatives of the total error E are obtained by summing over all

Chapter 8