more specifically, are most often improvements of the stochastic gradient descent algorithm, introduced in section 5.9.

We can of course, train models such as linear regression and support vector machines with gradient descent too, and in fact this is common when the training set is extremely large. From this point of view, training a neural network is not much different from training any other model. Computing the gradient is slightly more complicated for a neural network, but can still be done efficiently and exactly. Section 6.5 will describe how to obtain the gradient using the back-propagation algorithm and modern generalizations of the back-propagation algorithm.

As with other machine learning models, to apply gradient-based learning we must choose a cost function, and we must choose how to represent the output of the model. We now revisit these design considerations with special emphasis on the neural networks scenario.

6.2.1 Cost Functions

An important aspect of the design of a deep neural network is the choice of the cost function. Fortunately, the cost functions for neural networks are more or less the same as those for other parametric models, such as linear models.

In most cases, our parametric model defines a distribution $p(\boldsymbol{y} \mid \boldsymbol{x}; \boldsymbol{\theta})$ and we simply use the principle of maximum likelihood. This means we use the cross-entropy between the training data and the model's predictions as the cost function.

Sometimes, we take a simpler approach, where rather than predicting a complete probability distribution over $\boldsymbol{y}$, we merely predict some statistic of $\boldsymbol{y}$ conditioned on $\boldsymbol{x}$. Specialized loss functions allow us to train a predictor of these estimates.

The total cost function used to train a neural network will often combine one of the primary cost functions described here with a regularization term. We have already seen some simple examples of regularization applied to linear models in section 5.2.2. The weight decay approach used for linear models is also directly applicable to deep neural networks and is among the most popular regularization strategies. More advanced regularization strategies for neural networks will be described in chapter 7.

6.2.1.1 Learning Conditional Distributions with Maximum Likelihood

Most modern neural networks are trained using maximum likelihood. This means that the cost function is simply the negative log-likelihood, equivalently described