descent shuffle the dataset once and then pass through it multiple times. On the first pass, each minibatch is used to compute an unbiased estimate of the true generalization error. On the second pass, the estimate becomes biased because it is formed by re-sampling values that have already been used, rather than obtaining new fair samples from the data generating distribution.

The fact that stochastic gradient descent minimizes generalization error is easiest to see in the online learning case, where examples or minibatches are drawn from a **stream** of data. In other words, instead of receiving a fixed-size training set, the learner is similar to a living being who sees a new example at each instant, with every example $(\boldsymbol{x}, y)$ coming from the data generating distribution $p_{\text{data}}(\boldsymbol{x}, y)$. In this scenario, examples are never repeated; every experience is a fair sample from p_{data}.

The equivalence is easiest to derive when both $\boldsymbol{x}$ and y are discrete. In this case, the generalization error (equation 8.2) can be written as a sum

$$J^*(\boldsymbol{\theta}) = \sum_{\boldsymbol{x}} \sum_{y} p_{\text{data}}(\boldsymbol{x}, y) L(f(\boldsymbol{x}; \boldsymbol{\theta}), y), \tag{8.7}$$

with the exact gradient

$$\boldsymbol{g} = \nabla_{\boldsymbol{\theta}} J^*(\boldsymbol{\theta}) = \sum_{\boldsymbol{x}} \sum_{y} p_{\text{data}}(\boldsymbol{x}, y) \nabla_{\boldsymbol{\theta}} L(f(\boldsymbol{x}; \boldsymbol{\theta}), y). \tag{8.8}$$

We have already seen the same fact demonstrated for the log-likelihood in equation 8.5 and equation 8.6; we observe now that this holds for other functions L besides the likelihood. A similar result can be derived when $\boldsymbol{x}$ and y are continuous, under mild assumptions regarding p_{data} and L.

Hence, we can obtain an unbiased estimator of the exact gradient of the generalization error by sampling a minibatch of examples $\{\boldsymbol{x}^{(1)}, \dots \boldsymbol{x}^{(m)}\}$ with corresponding targets $y^{(i)}$ from the data generating distribution p_{data}, and computing the gradient of the loss with respect to the parameters for that minibatch:

$$\hat{\boldsymbol{g}} = \frac{1}{m} \nabla_{\boldsymbol{\theta}} \sum_{i} L(f(\boldsymbol{x}^{(i)}; \boldsymbol{\theta}), y^{(i)}). \tag{8.9}$$

Updating $\boldsymbol{\theta}$ in the direction of $\hat{\boldsymbol{g}}$ performs SGD on the generalization error.

Of course, this interpretation only applies when examples are not reused. Nonetheless, it is usually best to make several passes through the training set, unless the training set is extremely large. When multiple such epochs are used, only the first epoch follows the unbiased gradient of the generalization error, but