eters. These priors must not be improper as this will lead to a degenerate posterior, with all the mixing weight on one component.

Gibbs sampling is just one of a number of recently developed procedures for sampling from posterior distributions. It uses conditional sampling of each parameter given the rest, and is useful when the structure of the problem makes this sampling easy to carry out. Other methods do not require such structure, for example the *Metropolis–Hastings* algorithm. These and other computational Bayesian methods have been applied to sophisticated learning algorithms such as Gaussian process models and neural networks. Details may be found in the references given in the Bibliographic Notes at the end of this chapter.

8.7 Bagging

Earlier we introduced the bootstrap as a way of assessing the accuracy of a parameter estimate or a prediction. Here we show how to use the bootstrap to improve the estimate or prediction itself. In Section 8.4 we investigated the relationship between the bootstrap and Bayes approaches, and found that the bootstrap mean is approximately a posterior average. Bagging further exploits this connection.

Consider first the regression problem. Suppose we fit a model to our training data $\mathbf{Z} = \{(x_1, y_1), (x_2, y_2), \ldots, (x_N, y_N)\}$, obtaining the prediction $\hat{f}(x)$ at input x. Bootstrap aggregation or *bagging* averages this prediction over a collection of bootstrap samples, thereby reducing its variance. For each bootstrap sample $\mathbf{Z}^{*b}$, $b = 1, 2, \ldots, B$, we fit our model, giving prediction $\hat{f}^{*b}(x)$. The bagging estimate is defined by

$$\hat{f}_{\text{bag}}(x) = \frac{1}{B}\sum_{b=1}^{B} \hat{f}^{*b}(x). \tag{8.51}$$

Denote by $\hat{\mathcal{P}}$ the empirical distribution putting equal probability $1/N$ on each of the data points (x_i, y_i). In fact the "true" bagging estimate is defined by $\mathrm{E}_{\hat{\mathcal{P}}}\hat{f}^*(x)$, where $\mathbf{Z}^* = (x_1^*, y_1^*), (x_2^*, y_2^*), \ldots, (x_N^*, y_N^*)$ and each $(x_i^*, y_i^*) \sim \hat{\mathcal{P}}$. Expression (8.51) is a Monte Carlo estimate of the true bagging estimate, approaching it as $B \to \infty$.

The bagged estimate (8.51) will differ from the original estimate $\hat{f}(x)$ only when the latter is a nonlinear or adaptive function of the data. For example, to bag the B-spline smooth of Section 8.2.1, we average the curves in the bottom left panel of Figure 8.2 at each value of x. The B-spline smoother is linear in the data if we fix the inputs; hence if we sample using the parametric bootstrap in equation (8.6), then $\hat{f}_{\text{bag}}(x) \to \hat{f}(x)$ as $B \to \infty$ (Exercise 8.4). Hence bagging just reproduces the original smooth in the

top left panel of Figure 8.2. The same is approximately true if we were to bag using the nonparametric bootstrap.

A more interesting example is a regression tree, where $\hat{f}(x)$ denotes the tree's prediction at input vector x (regression trees are described in Chapter 9). Each bootstrap tree will typically involve different features than the original, and might have a different number of terminal nodes. The bagged estimate is the average prediction at x from these B trees.

Now suppose our tree produces a classifier $\hat{G}(x)$ for a K-class response. Here it is useful to consider an underlying indicator-vector function $\hat{f}(x)$, with value a single one and $K-1$ zeroes, such that $\hat{G}(x) = \arg\max_k \hat{f}(x)$. Then the bagged estimate $\hat{f}_{\text{bag}}(x)$ (8.51) is a K-vector $[p_1(x), p_2(x), \ldots, p_K(x)]$, with $p_k(x)$ equal to the proportion of trees predicting class k at x. The bagged classifier selects the class with the most "votes" from the B trees, $\hat{G}_{\text{bag}}(x) = \arg\max_k \hat{f}_{\text{bag}}(x)$.

Often we require the class-probability estimates at x, rather than the classifications themselves. It is tempting to treat the voting proportions $p_k(x)$ as estimates of these probabilities. A simple two-class example shows that they fail in this regard. Suppose the true probability of class 1 at x is 0.75, and each of the bagged classifiers accurately predict a 1. Then $p_1(x) = 1$, which is incorrect. For many classifiers $\hat{G}(x)$, however, there is already an underlying function $\hat{f}(x)$ that estimates the class probabilities at x (for trees, the class proportions in the terminal node). An alternative bagging strategy is to average these instead, rather than the vote indicator vectors. Not only does this produce improved estimates of the class probabilities, but it also tends to produce bagged classifiers with lower variance, especially for small B (see Figure 8.10 in the next example).

8.7.1 Example: Trees with Simulated Data

We generated a sample of size $N = 30$, with two classes and $p = 5$ features, each having a standard Gaussian distribution with pairwise correlation 0.95. The response Y was generated according to $\Pr(Y = 1|x_1 \le 0.5) = 0.2$, $\Pr(Y = 1|x_1 > 0.5) = 0.8$. The Bayes error is 0.2. A test sample of size 2000 was also generated from the same population. We fit classification trees to the training sample and to each of 200 bootstrap samples (classification trees are described in Chapter 9). No pruning was used. Figure 8.9 shows the original tree and eleven bootstrap trees. Notice how the trees are all different, with different splitting features and cutpoints. The test error for the original tree and the bagged tree is shown in Figure 8.10. In this example the trees have high variance due to the correlation in the predictors. Bagging succeeds in smoothing out this variance and hence reducing the test error.

Bagging can dramatically reduce the variance of unstable procedures like trees, leading to improved prediction. A simple argument shows why

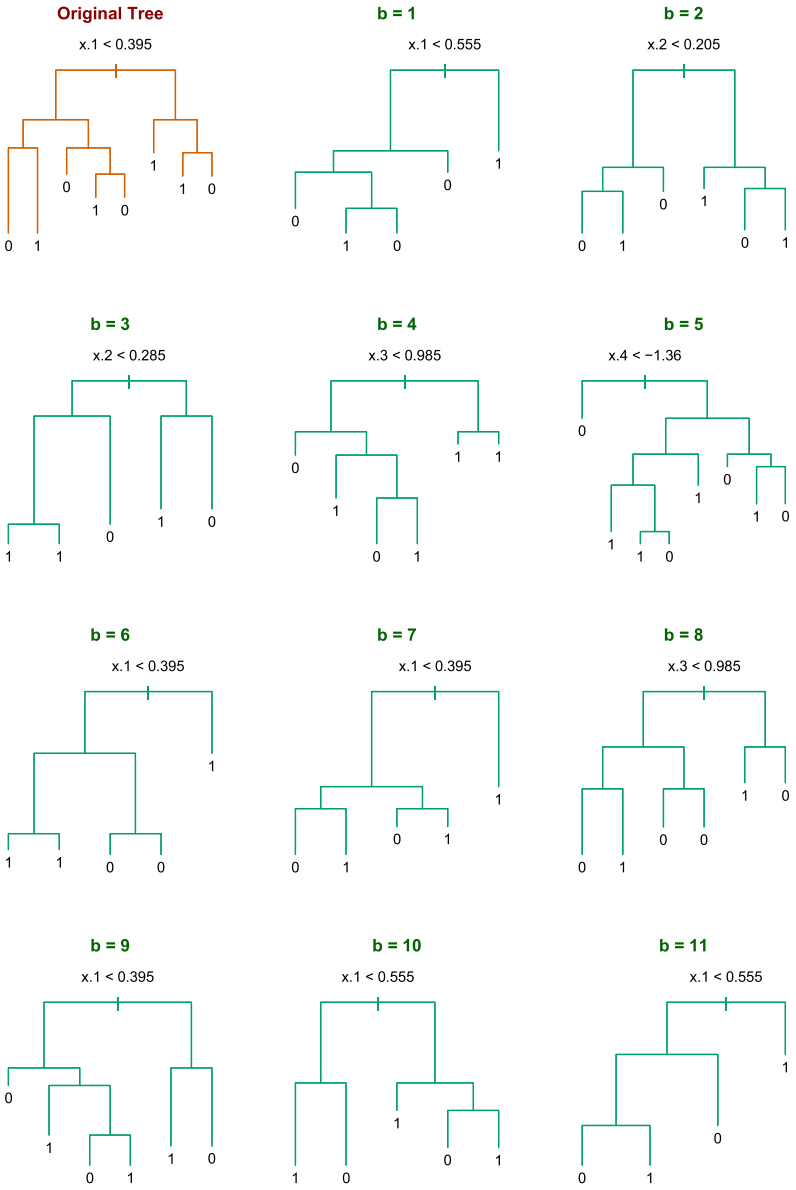

FIGURE 8.9. *Bagging trees on simulated dataset. The top left panel shows the original tree. Eleven trees grown on bootstrap samples are shown. For each tree, the top split is annotated.*

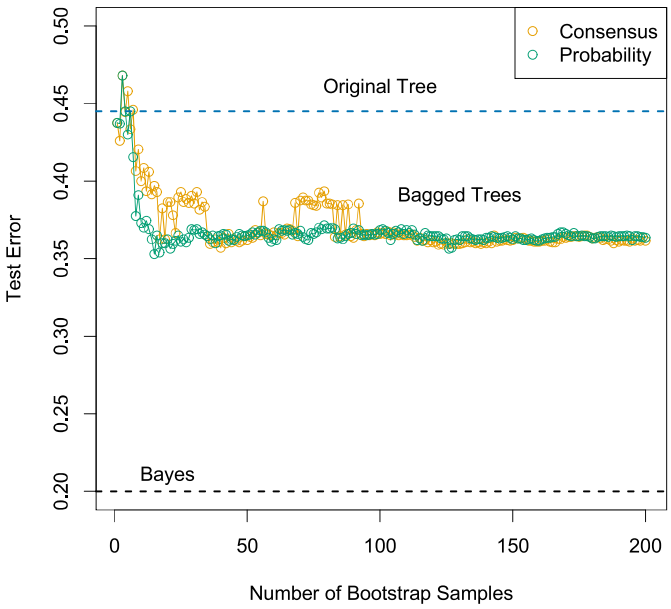

FIGURE 8.10. *Error curves for the bagging example of Figure 8.9. Shown is the test error of the original tree and bagged trees as a function of the number of bootstrap samples. The orange points correspond to the consensus vote, while the green points average the probabilities.*

bagging helps under squared-error loss, in short because averaging reduces variance and leaves bias unchanged.

Assume our training observations (x_i, y_i), $i = 1, \ldots, N$ are independently drawn from a distribution $\mathcal{P}$, and consider the ideal aggregate estimator $f_{\text{ag}}(x) = \text{E}_{\mathcal{P}}\hat{f}^*(x)$. Here x is fixed and the bootstrap dataset $\mathbf{Z}^*$ consists of observations x_i^*, y_i^*, $i = 1, 2, \ldots, N$ sampled from $\mathcal{P}$. Note that $f_{\text{ag}}(x)$ is a bagging estimate, drawing bootstrap samples from the actual population $\mathcal{P}$ rather than the data. It is not an estimate that we can use in practice, but is convenient for analysis. We can write

$$\begin{aligned} \text{E}_{\mathcal{P}}[Y - \hat{f}^*(x)]^2 &= \text{E}_{\mathcal{P}}[Y - f_{\text{ag}}(x) + f_{\text{ag}}(x) - \hat{f}^*(x)]^2 \\ &= \text{E}_{\mathcal{P}}[Y - f_{\text{ag}}(x)]^2 + \text{E}_{\mathcal{P}}[\hat{f}^*(x) - f_{\text{ag}}(x)]^2 \\ &\geq \text{E}_{\mathcal{P}}[Y - f_{\text{ag}}(x)]^2. \end{aligned} \tag{8.52}$$

The extra error on the right-hand side comes from the variance of $\hat{f}^*(x)$ around its mean $f_{\text{ag}}(x)$. Therefore true population aggregation never increases mean squared error. This suggests that bagging—drawing samples from the training data— will often decrease mean-squared error.

The above argument does not hold for classification under 0-1 loss, because of the nonadditivity of bias and variance. In that setting, bagging a

good classifier can make it better, but bagging a bad classifier can make it worse. Here is a simple example, using a randomized rule. Suppose $Y = 1$ for all x, and the classifier $\hat{G}(x)$ predicts $Y = 1$ (for all x) with probability 0.4 and predicts $Y = 0$ (for all x) with probability 0.6. Then the misclassification error of $\hat{G}(x)$ is 0.6 but that of the bagged classifier is 1.0.

For classification we can understand the bagging effect in terms of a consensus of independent *weak learners* (Dietterich, 2000a). Let the Bayes optimal decision at x be $G(x) = 1$ in a two-class example. Suppose each of the weak learners G_b^* have an error-rate $e_b = e < 0.5$, and let $S_1(x) = \sum_{b=1}^{B} I(G_b^*(x) = 1)$ be the consensus vote for class 1. Since the weak learners are assumed to be independent, $S_1(x) \sim \text{Bin}(B, 1 - e)$, and $\Pr(S_1 > B/2) \to 1$ as B gets large. This concept has been popularized outside of statistics as the "Wisdom of Crowds" (Surowiecki, 2004) — the collective knowledge of a diverse and independent body of people typically exceeds the knowledge of any single individual, and can be harnessed by voting. Of course, the main caveat here is "independent," and bagged trees are not. Figure 8.11 illustrates the power of a consensus vote in a simulated example, where only 30% of the voters have some knowledge.

In Chapter 15 we see how random forests improve on bagging by reducing the correlation between the sampled trees.

Note that when we bag a model, any simple structure in the model is lost. As an example, a bagged tree is no longer a tree. For interpretation of the model this is clearly a drawback. More stable procedures like nearest neighbors are typically not affected much by bagging. Unfortunately, the unstable models most helped by bagging are unstable because of the emphasis on interpretability, and this is lost in the bagging process.

Figure 8.12 shows an example where bagging doesn't help. The 100 data points shown have two features and two classes, separated by the gray linear boundary $x_1 + x_2 = 1$. We choose as our classifier $\hat{G}(x)$ a single axis-oriented split, choosing the split along either x_1 or x_2 that produces the largest decrease in training misclassification error.

The decision boundary obtained from bagging the 0-1 decision rule over $B = 50$ bootstrap samples is shown by the blue curve in the left panel. It does a poor job of capturing the true boundary. The single split rule, derived from the training data, splits near 0 (the middle of the range of x_1 or x_2), and hence has little contribution away from the center. Averaging the probabilities rather than the classifications does not help here. Bagging estimates the expected class probabilities from the single split rule, that is, averaged over many replications. Note that the expected class probabilities computed by bagging cannot be realized on any single replication, in the same way that a woman cannot have 2.4 children. In this sense, bagging increases somewhat the space of models of the individual base classifier. However, it doesn't help in this and many other examples where a greater enlargement of the model class is needed. "Boosting" is a way of doing this

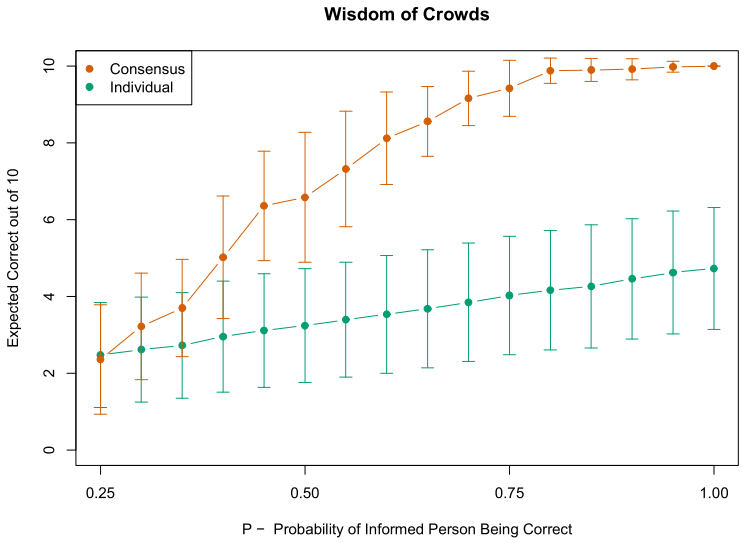

FIGURE 8.11. *Simulated academy awards voting.* 50 *members vote in 10 categories, each with* 4 *nominations. For any category, only* 15 *voters have some knowledge, represented by their probability of selecting the "correct" candidate in that category (so* $P = 0.25$ *means they have no knowledge). For each category, the* 15 *experts are chosen at random from the* 50*. Results show the expected correct (based on* 50 *simulations) for the consensus, as well as for the individuals. The error bars indicate one standard deviation. We see, for example, that if the* 15 *informed for a category have a* 50% *chance of selecting the correct candidate, the consensus doubles the expected performance of an individual.*