parameters. Adding a single parameter that has a random value for all objects (so that it does not separate the classes) can cause these methods to fail miserably.

The best choice of $k$ depends largely on the data. In general, larger values of $k$ tend to create larger classes in terms of the range of values included in them. This effect reduces the noise in the classification but makes the classification more choppy (relatively large distinctions between classes). A suitable $k$ for a given data set can be estimated by a decision rule or by using a resampling method (like cross-validation) to assign the mean value among samples. The accuracy of the $k$-NN algorithm can be severely degraded by the presence of noisy or irrelevant features, or if the feature scales are not consistent with their importance.

Analyzing Imbalanced Data Sets with Machine Learning Programs

Imbalanced Data Sets

Many problems in data mining involve the analysis of rare patterns of occurrence. For example, responses from a sales campaign can be very rare (typically, about 1%). You can just adjust the classification threshold to account for imbalanced data sets. Models built with many neural net and decision tree algorithms are very sensitive to imbalanced data sets. This imbalance between the rare category (customer response) and the common category (no response) can cause significant bias toward the common category in resulting models.

A neural net learns one case at a time. The error minimization routine (e.g., backpropagation, described in Chapter 7) adjusts the weights one case at a time. This adjustment process will be dominated by the most frequent class. If the most frequent class is “0” 99% of the time, the learning process will be 99% biased toward recognition of any data pattern as a “0.” Balancing data sets is necessary to balance the bias in the learning process. Clementine (for example) provides the ability to change the classification threshold in the Expert Options, and it provides the ability to balance the learning bias in the Balance node. If you have Clementine, run the same data set through the net with the threshold set appropriately. Then run the data through a Balance node (easily generated by a Distribution node), using the default threshold. Compare the results.

Another way to accomplish this balance is to weight the input cases appropriately. If the neural net can accept weights and use them appropriately in the error minimization process, the results can be comparable to using balanced data sets. STATISTICA does this. A final way to balance the learning bias is to adjust the prior probabilities of the “0” and “1” categories. SAS-Enterprise Miner and STATISTICA Data Miner use this approach.

If your data mining tool doesn’t have one of the preceding methods for removing the learning bias, you may have to physically balance the data by resampling. But resampling can be done two ways: by increasing the sample rate of the rare category (oversampling) or reducing the sample rate of the common category (undersampling). Undersampling the common category eliminates some of the common signal pattern. If the data set is not large, it is better to oversample the rare category. That approach retains all of the signal pattern of the common category and just duplicates the signal pattern of the rare category.