Table 5.9 Performance Measures for Four Numeric Prediction Models

	A	B	C	D
Root mean-squared error	67.8	91.7	63.3	57.4
Mean absolute error	41.3	38.5	33.4	29.2
Root relative squared error	42.2%	57.2%	39.4%	35.8%
Relative absolute error	43.1%	40.1%	34.8%	30.4%
Correlation coefficient	0.88	0.88	0.89	0.91

errors, and the reverse is true for absolute and relative absolute error. It is likely that the extra emphasis that the squaring operation gives to outliers accounts for the differences in this case.

When comparing two different learning schemes that involve numeric prediction, the methodology developed in Section 5.5 still applies. The only difference is that success rate is replaced by the appropriate performance measure (e.g., root mean-squared error) when performing the significance test.

5.10 THE MDL PRINCIPLE

What is learned by a machine learning scheme is a kind of "theory" of the domain from which the examples are drawn, a theory that is predictive in that it is capable of generating new facts about the domain—in other words, the class of unseen instances. Theory is rather a grandiose term: we are using it here only in the sense of a predictive model. Thus theories might comprise decision trees, or sets of rules—they don't have to be any more "theoretical" than that.

There is a long-standing tradition in science that, other things being equal, simple theories are preferable to complex ones. This is known as *Occam's Razor* after the medieval philosopher William of Occam (or Ockham). Occam's Razor shaves philosophical hairs off a theory. The idea is that the best scientific theory is the smallest one that explains all the facts. As Einstein is reputed to have said, "Everything should be made as simple as possible, but no simpler." Of course, quite a lot is hidden in the phrase "other things being equal," and it can be hard to assess objectively whether a particular theory really does "explain" all the facts on which it is based—that's what controversy in science is all about.

In our case, in machine learning, most theories make errors. And if what is learned is a theory, then the errors it makes are like *exceptions* to the theory. One way to ensure that other things *are* equal is to insist that the information embodied in the exceptions is included as part of the theory when its "simplicity" is judged.

Imagine an imperfect theory for which there are a few exceptions. Not all the data is explained by the theory, but most is. What we do is simply adjoin the exceptions to the theory, specifying them explicitly as exceptions. This new theory is larger: i.e., a price that, quite justifiably, has to be paid for its inability

to explain all the data. However, it may be that the simplicity—Is it too much to call it *elegance*?—of the original theory is sufficient to outweigh the fact that it does not quite explain everything compared with a large, baroque theory that is more comprehensive and accurate.

For example, if Kepler's three laws of planetary motion did not at the time account for the known data quite so well as Copernicus's latest refinement of the Ptolemaic theory of epicycles, they had the advantage of being far less complex, and that would have justified any slight apparent inaccuracy. Kepler was well aware of the benefits of having a theory that was compact, despite the fact that his theory violated his own esthetic sense because it depended on "ovals" rather than pure circular motion. He expressed this in a forceful metaphor: "I have cleared the Augean stables of astronomy of cycles and spirals, and left behind me only a single cartload of dung."

The MDL principle takes the stance that the best theory for a body of data is one that minimizes the size of the theory plus the amount of information necessary to specify the exceptions relative to the theory—the smallest cartload of dung. In statistical estimation theory, this has been applied successfully to various parameter-fitting problems. It applies to machine learning as follows: given a set of instances, a learning scheme infers a theory—be it ever so simple; unworthy, perhaps, to be called a "theory"—from them. Using a metaphor of communication, imagine that the instances are to be transmitted through a noiseless channel. Any similarity that is detected among them can be exploited to give a more compact coding. According to the MDL principle, the best theory is the one that minimizes the number of bits required to communicate the theory, along with the labels of the examples from which it was made.

Now the connection with the informational loss function introduced in Section 5.7 should be starting to emerge. That function measures the error in terms of the number of bits required to transmit the instances' class labels, given the probabilistic predictions made by the theory. According to the MDL principle we need to add to this the "size" of the theory in bits, suitably encoded, to obtain an overall figure for complexity. However, the MDL principle refers to the information required to transmit the examples from which the theory was formed, i.e., the *training* instances—not a test set. The overfitting problem is avoided because a complex theory that overfits will be penalized relative to a simple one by virtue of the fact that it takes more bits to encode. At one extreme is a very complex, highly overfitted theory that makes no errors on the training set. At the other is a very simple theory—the null theory—which does not help at all when transmitting the training set. And in between are theories of intermediate complexity, which make probabilistic predictions that are imperfect and need to be corrected by transmitting some information about the training set. The MDL principle provides a means of comparing all these possibilities on an equal footing to see which is the best. We have found the Holy Grail: an evaluation scheme that works on the training set alone and does not need a separate test set. But the devil is in the details, as we will see.

Suppose a learning scheme comes up with a theory T, based on a training set E of examples, that requires a certain number of bits $L(T)$ to encode (L for *length*). We are only interested in predicting class labels correctly, so we assume that E stands for the collection of class labels in the training set. Given the theory, the training set itself can be encoded in a certain number of bits, $L(E|T)$. $L(E|T)$ is in fact given by the informational loss function summed over all members of the training set. Then the total description length of theory plus training set is

$$L(T) + L(E|T)$$

and the MDL principle recommends choosing the theory T that minimizes this sum.

There is a remarkable connection between the MDL principle and basic probability theory. Given a training set E, we seek the "most likely" theory T, i.e., the theory for which the a posteriori probability $P(T|E)$—the probability after the examples have been seen—is maximized. Bayes' rule of conditional probability, the very same rule that we encountered in Section 4.2, dictates that

$$P(T|E) = \frac{P(E|T)P(T)}{P(E)}.$$

Taking negative logarithms,

$$-\log P(T|E) = -\log P(E|T) - \log P(T) + \log P(E).$$

Maximizing the probability is the same as minimizing its negative logarithm. Now (as we saw in Section 5.7) the number of bits required to code something is just the negative logarithm of its probability. Furthermore, the final term, $\log P(E)$, depends solely on the training set and not on the learning method. Thus choosing the theory that maximizes the probability $P(T|E)$ is tantamount to choosing the theory that minimizes.

$$\mathrm{L}(E|T) + \mathrm{L}(T)$$

—in other words, the MDL principle!

This astonishing correspondence with the notion of maximizing the a posteriori probability of a theory after the training set has been taken into account gives credence to the MDL principle. But it also points out where the problems will sprout when the principle is applied in practice. The difficulty with applying Bayes' rule directly is in finding a suitable prior probability distribution $P(T)$ for the theory. In the MDL formulation, that translates into finding how to code the theory T into bits in the most efficient way. There are many ways of coding things, and they all depend on presuppositions that must be shared by encoder and decoder. If you know in advance that the theory is going to take a certain form, you can use that information to encode it more efficiently. How are you going to actually encode T? The devil is in the details.

Encoding E with respect to T to obtain $L(E|T)$ seems a little more straightforward: we have already met the informational loss function. But actually, when

you encode one member of the training set after another, you are encoding a *sequence* rather than a *set*. It is not necessary to transmit the training set in any particular order, and it ought to be possible to use that fact to reduce the number of bits required. Often, this is simply approximated by subtracting $\log n!$ (where n is the number of elements in E), which is the number of bits needed to specify a particular permutation of the training set (and because this is the same for all theories, it doesn't actually affect the comparison between them). But one can imagine using the frequency of the individual errors to reduce the number of bits needed to code them. Of course, the more sophisticated the method that is used to code the errors, the less the need for a theory in the first place—so whether a theory is justified or not depends to some extent on how the errors are coded. The details, the details.

We end this section as we began, on a philosophical note. It is important to appreciate that Occam's Razor, the preference of simple theories over complex ones, has the status of a philosophical position or "axiom" rather than something that can be proven from first principles. While it may seem self-evident to us, this is a function of our education and the times we live in. A preference for simplicity is—or may be—culture specific rather than absolute.

The Greek philosopher Epicurus (who enjoyed good food and wine and supposedly advocated sensual pleasure—in moderation—as the highest good) expressed almost the opposite sentiment. His *principle of multiple explanations* advises "if more than one theory is consistent with the data, keep them all" on the basis that if several explanations are equally in agreement, it may be possible to achieve a higher degree of precision by using them together—and anyway, it would be unscientific to discard some arbitrarily. This brings to mind instance-based learning, in which all the evidence is retained to provide robust predictions, and resonates strongly with decision combination methods such as bagging and boosting (described in Chapter 12: Ensemble learning) that actually do gain predictive power by using multiple explanations together.

5.11 APPLYING THE MDL PRINCIPLE TO CLUSTERING

One of the nice things about the MDL principle is that unlike other evaluation criteria, it can be applied under widely different circumstances. Although in some sense equivalent to Bayes' rule in that, as we have seen, devising a coding scheme for theories is tantamount to assigning them a prior probability distribution, schemes for coding are somehow far more tangible and easier to think about in concrete terms than intuitive prior probabilities. To illustrate this we will briefly describe—without entering into coding details—how you might go about applying the MDL principle to clustering.

Clustering seems intrinsically difficult to evaluate. Whereas classification or association learning has an objective criterion of success—predictions made on

test cases are either right or wrong—this is not so with clustering. It seems that the only realistic evaluation is whether the result of learning—the clustering—proves useful in the application context. (It is worth pointing out that really this is the case for all types of learning, not just clustering.)

Despite this, clustering can be evaluated from a description length perspective. Suppose a cluster-learning technique divides the training set E into k clusters. If these clusters are natural ones, it should be possible to use them to encode E more efficiently. The best clustering will support the most efficient encoding.

One way of encoding the instances in E with respect to a given clustering is to start by encoding the cluster centers—the average value of each attribute over all instances in the cluster. Then, for each instance in E, transmit which cluster it belongs to (in $\log_2 k$ bits) followed by its attribute values with respect to the cluster center—perhaps as the numeric difference of each attribute value from the center. Couched as it is in terms of averages and differences, this description presupposes numeric attributes and raises thorny questions of how to code numbers efficiently. Nominal attributes can be handled in a similar manner: for each cluster there is a probability distribution for the attribute values, and the distributions are different for different clusters. The coding issue becomes more straightforward: attribute values are coded with respect to the relevant probability distribution, a standard operation in data compression.

If the data exhibits extremely strong clustering, this technique will result in a smaller description length than simply transmitting the elements of E without any clusters. However, if the clustering effect is not so strong, it will likely increase rather than decrease the description length. The overhead of transmitting cluster-specific distributions for attribute values will more than offset the advantage gained by encoding each training instance relative to the cluster it lies in. This is where more sophisticated coding techniques come in. Once the cluster centers have been communicated, it is possible to transmit cluster-specific probability distributions adaptively, in tandem with the relevant instances: the instances themselves help to define the probability distributions, and the probability distributions help to define the instances. We will not venture further into coding techniques here. The point is that the MDL formulation, properly applied, may be flexible enough to support the evaluation of clustering. But actually doing it satisfactorily in practice is not easy.

5.12 USING A VALIDATION SET FOR MODEL SELECTION

The MDL principle is an example of a so-called model selection criterion, which can be used to determine the appropriate complexity of a model for a given dataset. Adding unnecessary structure to a model can result in overfitting and a consequent drop in predictive performance. Conversely, insufficient model complexity implies that the information in the training data cannot be completely

exploited: the model will underfit. Model selection criteria such as the MDL principle can be used as tools for guessing the right complexity.

A classic model selection problem in statistics is to determine, for a given dataset, what subset of attributes to use in a linear regression model for the data. (Even a simple technique such as linear regression can overfit!) However, the problem is ubiquitous in machine learning because learning algorithms inevitably need to choose how much structure to add to a model. Examples include pruning subtrees in a decision tree, determining the number of instances to retain in a nearest-neighbor classifier, and picking the number and size of the layers in an artificial neural network.

Many model selection strategies similar to the MDL principle exist, based on various theoretical approaches and corresponding underlying assumptions. They all follow the same strategy: the predictive performance on the training data is balanced with the complexity of the model. The aim is to find a sweet spot. Whether they succeed depends on whether the underlying assumptions are appropriate for the problem at hand. This is difficult to know in practice. The good news is that there is a simple alternative approach to guessing what model complexity will maximize predictive performance on new data: we can simply use a validation set for model selection, just as we did in Section 5.5 for tuning hyperparameters. Alternatively, if the dataset is small, we can use cross-validation, or maybe the bootstrap.

5.13 FURTHER READING AND BIBLIOGRAPHIC NOTES

The statistical basis of confidence tests is well covered in most statistics texts, which also gives tables of the normal distribution and Student's distribution. (We use an excellent course text by Wild and Seber (1995), which we recommend very strongly if you can get hold of it). "Student" is the nom de plume of a statistician called William Gosset, who obtained a post as a chemist in the Guinness brewery in Dublin, Ireland, in 1899 and invented the *t*-test to handle small samples for quality control in brewing. The corrected resampled *t*-test was proposed by Nadeau and Bengio (2003). Cross-validation is a standard statistical technique, and its application in machine learning has been extensively investigated and compared with the bootstrap by Kohavi (1995a). The bootstrap technique itself is thoroughly covered by Efron and Tibshirani (1993).

The Kappa statistic was introduced by Cohen (1960). Ting (2002) has investigated a heuristic way of generalizing to the multiclass case the algorithm given in Section 5.8 to make two-class learning schemes cost sensitive. Lift charts are described by Berry and Linoff (1997). The use of ROC analysis in signal detection theory is covered by Egan (1975); this work has been extended for visualizing and analyzing the behavior of diagnostic systems (Swets, 1988) and is also used in medicine (Beck & Schultz, 1986). Provost and Fawcett (1997) brought