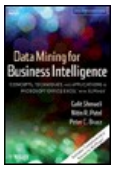

Data Mining for Business Intelligence: Concepts, Techniques, and Applications in Microsoft Office Excel with XLMiner, Second Edition

by Galit Shmueli, Nitin R. Patel and Peter C. Bruce
John Wiley & Sons (US). (c) 2010. Copying Prohibited.

Chapter 2: Overview of the Data Mining Process

In Kare this chapter we give an overview of the steps involved in data mining, starting from a clear goal definition and ending with model deployment. The general steps are shown schematically in Figure 2.1. We also discuss issues related to data collection, cleaning, and preprocessing. We explain the notion of data partitioning, where methods are trained on a set of training data and then their performance is evaluated on a separate set of validation data, and how this practice helps avoid overfitting. Finally, we illustrate the steps of model building by applying them to data.

FIGURE 2.1: SCHEMATIC OF THE DATA MODELING PROCESS

2.1 Introduction

In Chapter 1 we saw some very general definitions of data mining. In this chapter we introduce the variety of methods sometimes referred to as *data mining*. The core of this book focuses on what has come to be called *predictive analytics*, the tasks of classification and prediction that are becoming key elements of a "business intelligence" function in most large firms. These terms are described and illustrated below.

Not covered in this book to any great extent are two simpler database methods that are sometimes considered to be data mining techniques: (1) OLAP (online analytical processing) and (2) SQL (structured query language). OLAP and SQL searches on databases are descriptive in nature ("find all credit card customers in a certain zip code with annual charges > $20, 000, who own their own home and who pay the entire amount of their monthly bill at least 95% of the time") and do not involve statistical modeling.

2.2 Core Ideas in Data Mining

Classification

Classification is perhaps the most basic form of data analysis. The recipient of an offer can respond or not respond. An applicant for a loan can repay on time, repay late, or declare bankruptcy. A credit card transaction can be normal or fraudulent. A packet of data traveling on a network can be benign or threatening. A bus in a fleet can be available for service or unavailable. The victim of an illness can be recovered, still be ill, or be deceased.

A common task in data mining is to examine data where the classification is unknown or will occur in the future, with the goal of predicting what that classification is or will be. Similar data where the classification is known are used to develop rules, which are then applied to the data with the unknown classification.

Prediction

Prediction is similar to classification, except that we are trying to predict the value of a numerical variable (e.g., amount of purchase) rather than a class (e.g., purchaser or nonpurchaser). Of course, in classification we are trying to predict a class, but the term *prediction* in this book refers to the prediction of the value of a continuous variable. (Sometimes in the data mining literature, the term *estimation* is used to refer to the prediction of the value of a continuous variable, and *prediction* may be used for both continuous and categorical data.)

Association Rules

Large databases of customer transactions lend themselves naturally to the analysis of associations among items purchased, or "what goes with what" *Association rules, or affinity analysis*, can then be used in a variety of ways. For example, grocery stores can use such information after a customer's purchases have all been scanned to print discount coupons, where the items being discounted are determined by mapping the customer's purchases onto the association rules. Online merchants such as Amazon.com and Netflix.com use these methods as the heart of a "recommender" system that suggests new purchases to customers.

Predictive Analytics

Classification, prediction, and to some extent, affinity analysis constitute the analytical methods employed in *predictive analytics*.

Data Reduction

Sensible data analysis often requires distillation of complex data into simpler data. Rather than dealing with thousands of product types, an analyst might wish to group them into a smaller number of groups. This process of consolidating a large number of variables (or cases) into a smaller set is termed *data reduction*.

Data Exploration

Unless our data project is very narrowly focused on answering a specific question determined in advance (in which case it has drifted more into the realm of statistical analysis than of data mining), an essential part of the job is to review and examine the data to see what messages they hold, much as a detective might survey a crime scene. Here, full understanding of the data may require a reduction in its scale or dimension to allow us to see the forest without getting lost in the trees. Similar variables (i.e., variables that supply similar information) might be aggregated into a single variable incorporating all the similar variables. Analogously, records might be aggregated into groups of similar records.

Data Visualization

Another technique for exploring data to see what information they hold is through graphical analysis. This includes looking at each variable separately as well as looking at relationships between variables. For numerical variables, we use histograms and boxplots to learn about the distribution of their values, to detect outliers (extreme observations), and to find other information that is relevant to the analysis task. Similarly, for categorical variables we use bar charts. We can also look at scatterplots of pairs of numerical variables to learn about possible relationships, the type of relationship, and again, to detect outliers. Visualization can be greatly enhanced by adding features such as color, zooming, and interactive navigation.

2.3 Supervised and Unsupervised Learning

A fundamental distinction among data mining techniques is between supervised and unsupervised methods. *Supervised learning algorithms* are those used in classification and prediction. We must have data available in which the value of the outcome of interest (e.g., purchase or no purchase) is known. These *training data* are the data from which the classification or prediction algorithm "learns," or is "trained," about the relationship between predictor variables and the outcome variable. Once the algorithm has learned from the training data, it is then applied to another sample of data (the *validation data)* where the outcome is known, to see how well it does in comparison to other models. Ifmany different models are being tried out, it is prudent to save a third sample of known outcomes (the *test data)* to use with the model finally selected to predict how well it will do. The model can then be used to classify or predict the outcome of interest in new cases where the outcome is unknown. Simple linear regression analysis is an example of supervised learning (although rarely called that in the introductory statistics course where you probably first encountered it). The *Y* variable is the (known) outcome variable and the *X* variable is a predictor variable. A regression line is drawn to minimize the sum of squared deviations between the actual *Y* values and the values predicted by this line. The regression line can now be used to predict *Y* values for new values of *X* for which we do not know the *Y* value.

Unsupervised learning algorithms are those used where there is no outcome variable to predict or classify. Hence, there is no "learning" from cases where such an outcome variable is known. Association rules, dimension reduction methods, and clustering techniques are all unsupervised learning methods.

2.4 Steps in Data Mining

This book focuses on understanding and using data mining algorithms (steps 4-7 below). However, some of the most serious errors in data analysis result from a poor understanding of the problem—an understanding that must be developed before we get into the details of algorithms to be used. Here is a list of steps to be taken in a typical data mining effort:

1. *Develop an understanding of the purpose of the data mining project* (if it is a one-shot effort to answer a question or questions) or application (if it is an ongoing procedure).

2. *Obtain the dataset to be used in the analysis*. This often involves random sampling from a large database to capture records to be used in an analysis. It may also involve pulling together data from different databases. The databases could be internal (e.g., past purchases made by customers) or external (credit ratings). While data mining deals with very large databases, usually the analysis to be done requires only thousands or tens of thousands of records.

3. *Explore, clean, and preprocess the data*. This involves verifying that the data are in reasonable condition. How should missing data be handled? Are the values in a reasonable range, given what you would expect for each variable? Are there obvious outliers? The data are reviewed graphically: for example, a matrix of scatterplots showing the relationship of each variable with every other variable. We also need to ensure consistency in the definitions of fields, units of measurement, time periods, and so on.

4. *Reduce the data, if necessary, and* (where supervised training is involved) *separate them into training, validation, and test datasets*. This can involve operations such as eliminating unneeded variables, transforming variables (e.g., turning "money spent" into "spent > $100" vs. "spent ≤ $100"), and creating new variables (e.g., a variable that records whether at least one of several products was purchased). Make sure that you know what each variable means and whether it is sensible to include it in the model.

5. *Determine the data mining task* (classification, prediction, clustering, etc.). This involves translating the general question or problem of step 1 into a more specific statistical question.

6. *Choose the data mining techniques to be used* (regression, neural nets, hierarchical clustering, etc.).

7. *Use algorithms to perform the task*. This is typically an iterative process— trying multiple variants, and often using multiple variants of the same algorithm (choosing different variables or settings within the algorithm). Where appropriate, feedback from the algorithm's performance on validation data is used to refine the settings.

8. *Interpret the results of the algorithms*. This involves making a choice as to the best algorithm to deploy, and where possible, testing the final choice on the test data to get an idea as to how well it will perform. (Recall that each algorithm may also be tested on the validation data for tuning purposes; in this way the validation data become a part of the fitting process and are likely to underestimate the error in the deployment of the model that is finally chosen.)

9. *Deploy the model*. This involves integrating the model into operational systems and running it on real records to produce decisions or actions. For example, the model might be applied to a purchased list of possible customers, and the action might be "include in the mailing ifthe predicted amount of purchase is > $10."

The foregoing steps encompass the steps in SEMMA, a methodology developed by SAS:

Sample Take a sample from the dataset; partition into training, validation, and test datasets.

Explore Examine the dataset statistically and graphically.

Modify Transform the variables and impute missing values.

Model Fit predictive models (e.g., regression tree, collaborative filtering).

Assess Compare models using a validation dataset.

IBM Modeler (previously SPSS Clementine) has a similar methodology, termed CRISP-DM (cross-industry standard process for data mining).

2.5 Preliminary Steps

Organization of Datasets

Datasets are nearly always constructed and displayed so that variables are in columns and records are in rows. In the example shown in Section 2.6 (the Boston housing data), the values of 14 variables are recorded for a number of census tracts. The spreadsheet is organized such that each row represents a census tract—the first tract had a per capital crime rate (CRIM) of 0.00632, had 18% of its residential lots zoned for over 25, 000 square feet (ZN), and so on. In supervised learning situations, one of these variables will be the outcome variable, typically listed at the end or the beginning (in this case it is median value, MEDV, at the end).

Samplingfrom a Database

Quite often, we want to perform our data mining analysis on less than the total number of records that are available. Data mining algorithms will have varying limitations on what they can handle in terms of the numbers of records and variables, limitations that may be specific to computing power and capacity as well as software limitations. Even within those limits, many algorithms will execute faster with smaller datasets.

From a statistical perspective, accurate models can often be built with as few as several hundred records (see below). Hence, we will often want to sample a subset of records for model building.

Oversampling Rare Events

If the event we are interested in is rare, however (e.g., customers purchasing a product in response to a mailing), sampling a subset of records may yield so few events (e.g., purchases) that we have little information on them. We would end up with lots of data on nonpurchasers but little on which to base a model that distinguishes purchasers from nonpurchasers. In such cases we would want our sampling procedure to overweight the purchasers relative to the nonpurchasers so that our sample would end up with a healthy complement of purchasers. This issue arises mainly in classification problems because those are the types of problems in which an overwhelming number of 0's is likely to be encountered in the response variable. Although the same principle could be extended to prediction, any prediction problem in which most responses are 0 is likely to raise the question of what distinguishes responses from nonresponses (i.e., a classification question). (For convenience below, we speak of responders and nonresponders as to a promotional offer, but we are really referring to any binary—0/1—outcome situation.)

Assuring an adequate number of responder or "success" cases to train the model is just part of the picture. A more important factor is the costs of misclas-sification. Whenever the response rate is extremely low, we are likely to attach more importance to identifying a responder than to identifying a nonresponder. In direct-response advertising (whether by traditional mail or via the Internet), we may encounter only one or two responders for every hundred records—the value of finding such a customer far outweighs the costs of reaching him or her. In trying to identify fraudulent transactions, or customers unlikely to repay debt, the costs of failing to find the fraud or the nonpaying customer are likely to exceed the cost of more detailed review of a legitimate transaction or customer.

If the costs of failing to locate responders were comparable to the costs of misidentifying responders as nonresponders, our models would usually be at their best if they identified everyone (or almost everyone, if it is easy to pick off a few responders without catching many nonresponders) as a nonresponder. In such a case, the misclassification rate is very low—equal to the rate of responders—but the model is of no value.

More generally, we want to train our model with the asymmetric costs in mind so that the algorithm will catch the more valuable responders, probably at the cost of "catching" and misclassifying more nonresponders as responders than would be the case if we assume equal costs. This subject is discussed in detail in Chapter 5.

Preprocessing and Cleaning the Data

Types of Variables There are several ways of classifying variables. Variables can be numerical or text (character). They can be continuous (able to assume any real numerical value, usually in a given range), integer (assuming only integer values), or categorical (assuming one of a limited number of values). Categorical variables can be either numerical (1, 2, 3) or text (payments current, payments not current, bankrupt). Categorical variables can also be unordered (called *nominal variables)* with categories such as North America, Europe, and Asia; or they can be ordered (called *ordinal variables)* with categories such as high value, low value, and nil value.

Continuous variables can be handled by most data mining routines. In XLMiner, all routines take continuous variables, with the exception of the naive Bayes classifier, which deals exclusively with categorical variables. The machine learning roots of data mining grew out of problems with categorical outcomes; the roots of statistics lie in the analysis of continuous variables. Sometimes, it is desirable to convert continuous variables to categorical variables. This is done most typically in the case of outcome variables, where the numerical variable is mapped to a decision (e.g., credit scores above a certain level mean "grant credit," a medical test result above a certain level means "start treatment"). XLMiner has a facility for this type of conversion.

Handling Categorical Variables Categorical variables can also be handled by most routines but often require special handling. If the categorical variable is ordered (age category, degree of creditworthiness, etc.), we can often use it as is, as if it were a continuous variable. The smaller the number of categories, and the less they represent equal increments of value, the more problematic this procedure becomes, but it often works well enough.

Categorical variables, however, often cannot be used as is. In those cases they must be decomposed into a series of dummy binary variables. For example, a single variable that can have possible values of "student," "unemployed," "employed," or "retired" would be split into four separate variables:

Student—Yes/No

Unemployed—Yes/No

Employed—Yes/No

Retired—Yes/No

Note that only three of the variables need to be used; if the values of three are known, the fourth is also known. For example, given that these four values are the only possible ones, we can know that if a person is neither student, unemployed, nor employed, he or she must be retired. In some routines (e.g., regression and logistic regression), you should not use all four variables—the redundant information will cause the algorithm to fail. XLMiner has a utility to convert categorical variables to binary dummies.

Variable Selection More is not necessarily better when it comes to selecting variables for a model. Other things being equal, parsimony, or compactness, is a desirable feature in a model. For one thing, the more variables we include, the greater the number of records we will need to assess relationships among the variables. Fifteen records may suffice to give us a rough idea of the relationship between *Y* and a single predictor variable *X*. If we now want information about the relationship between *Y* and 15 predictor variables $X_1 \cdots X_{15}$, 15 records will not be enough (each estimated relationship would have an average of only one record's worth of information, making the estimate very unreliable).

Overfitting The more variables we include, the greater the risk of over-fitting the data. What is overfitting?

In Table 2.1 we show hypothetical data about advertising expenditures in one time period and sales in a subsequent time period (a scatterplot of the data is shown in Figure 2.2). We could connect up these points with a smooth but complicated function, one that explains all these data points perfectly and leaves no error (residuals). This can be seen in Figure 2.3. However, we can see that such a curve is unlikely to be accurate, or even useful, in predicting future sales on the basis of advertising expenditures (e.g., it is hard to believe that increasing expenditures from $400 to $500 will actually decrease revenue).

Table 2.1

Advertising	Sales
239	514
364	789
602	550
644	1386
770	1394
789	1440
911	1354

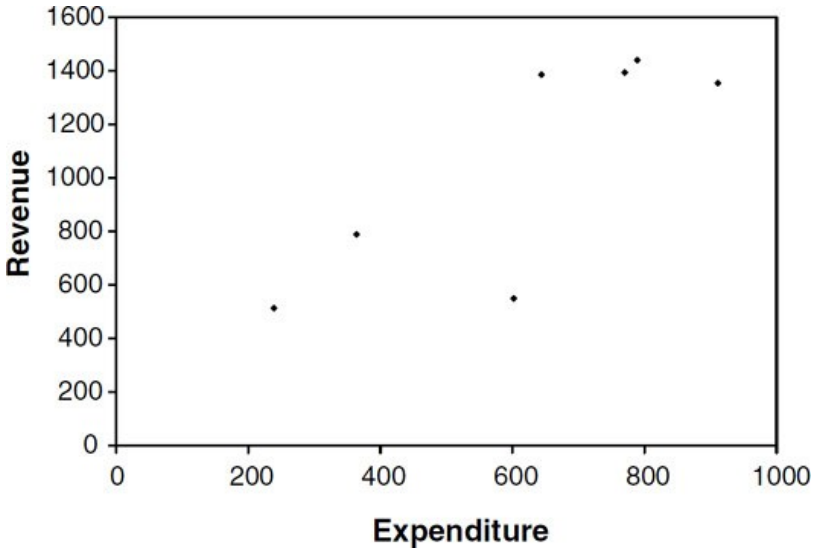

FIGURE 2.2: SCATTERPLOT FOR ADVERTISING AND SALES DATA

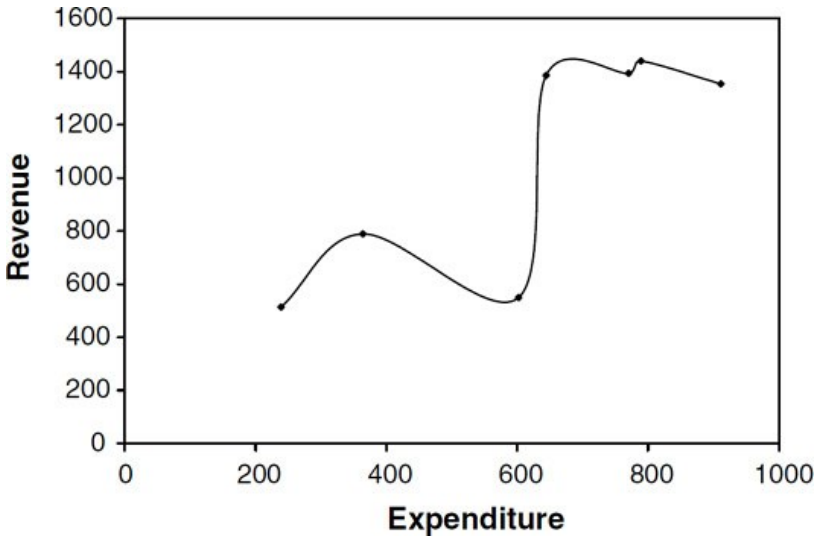

FIGURE 2.3: SCATTERPLOT SMOOTHED

A basic purpose of building a model is to describe relationships among variables in such a way that this description will do a good job of predicting future outcome (dependent) values on the basis of future predictor (independent) values. Of course, we want the model to do a good job of describing the data we have, but we are more interested in its performance with future data.

In the example above, a simple straight line might do a better job than the complex function does of predicting future sales on the basis of advertising. Instead, we devised a complex function that fit the data perfectly, and in doing so, we overreached. We ended up "explaining" some variation in the data that was nothing more than chance variation. We mislabeled the noise in the data as if it were a signal.

Similarly, we can add predictors to a model to sharpen its performance with the data at hand. Consider a database of 100 individuals, half of whom have contributed to a charitable cause. Information about income, family size, and zip code might do a fair job of predicting whether or not someone is a contributor. Ifwe keep adding additional predictors, we can improve the performance of the model with the data at hand and reduce the misclassification error to a negligible level. However, this low error rate is misleading because it probably includes spurious "explanations."

For example, one of the variables might be height. We have no basis in theory to suppose that tall people might contribute more or less to charity, but if there are several tall people in our sample and they just happened to contribute heavily to charity, our model might include a term for height—the taller you are, the more you will contribute. Of course, when the model is applied to additional data, it is likely that this will not turn out to be a good predictor.

If the dataset is not much larger than the number of predictor variables, it is very likely that a spurious relationship like this will creep into the model. Continuing with our charity example, with a small sample just a few of whom are tall, whatever the contribution level of tall people may be, the algorithm is tempted to attribute it to their being tall. If the dataset is very large relative to the number of predictors, this is less likely. In such a case, each predictor must help predict the outcome for a large number of cases, so the job it does is much less dependent on just a few cases, which might be flukes.

Somewhat surprisingly, even ifwe know for a fact that a higher degree curve is the appropriate model, if the model-fitting dataset is not large enough, a lower degree function (that is not as likely to fit the noise) is likely to perform better. Overfitting can also result from the application of many different models, from which the best performing is selected (see below).

How Many Variables and How Much Data? Statisticians give us procedures to learn with some precision how many records we would need to achieve a given degree of reliability with a given dataset and a given model. Data miners' needs are usually not so precise, so we can often get by with rough rules of thumb. A good rule of thumb is to have 10 records for every predictor variable. Another, used by Delmaster and Hancock (2001, p. 68) for classification procedures, is to have at least 6 β*m* *βp* records, where *m* is the number of outcome classes and *p* is the number of variables.

Even when we have an ample supply of data, there are good reasons to pay close attention to the variables that are included in a model. Someone with domain knowledge (i.e., knowledge of the business process and the data) should be consulted, as knowledge of what the variables represent can help build a good model and avoid errors.

For example, the amount spent on shipping might be an excellent predictor of the total amount spent, but it is not a helpful one. It will not give us any information about what distinguishes high-paying from low-paying customers that can be put to use with future prospects because we will not have the information on the amount paid for shipping for prospects that have not yet bought anything.

In general, compactness or parsimony is a desirable feature in a model. A matrix of scatterplots can be useful in variable selection. In such a matrix, we can see at a glance scatterplots for all variable combinations. A straight line would be an indication that one variable is exactly correlated with another. Typically, we would want to include only one of them in our model. The idea is to weed out irrelevant and redundant variables from our model.

Outliers The more data we are dealing with, the greater the chance of encountering erroneous values resulting from measurement error, data entry error, or the like. If the erroneous value is in the same range as the rest of the data, it may be harmless. Ifit is well outside the range of the rest of the data (e.g., a misplaced decimal), it may have a substantial effect on some of the data mining procedures we plan to use.

Values that lie far away from the bulk of the data are called *outliers*. The term *far away* is deliberately left vague because what is or is not called an outlier is basically an arbitrary decision. Analysts use rules of thumb such as "anything over 3 standard deviations away from the mean is an outlier," but no statistical rule can tell us whether such an outlier is the result of an error. In this statistical sense, an outlier is not necessarily an invalid data point; it is just a distant data point.

The purpose of identifying outliers is usually to call attention to values that need further review. We might come up with an explanation looking at the data— in the case of a misplaced decimal, this is likely. We might have no explanation but know that the value is wrong—a temperature of 178°F for a sick person. Or, we might conclude that the value is within the realm of possibility and leave it alone. All these are judgments best made by someone with *domain knowledge*, knowledge of the particular application being considered: direct mail, mortgage finance, and so on, as opposed to technical knowledge of statistical or data mining procedures. Statistical procedures can do little beyond identifying the record as something that needs review.

If manual review is feasible, some outliers may be identified and corrected. In any case, if the number of records with outliers is very small, they might be treated as missing data. How do we inspect for outliers? One technique in Excel is to sort the records by the first column, then review the data for very large or very small values in that column. Then repeat for each successive column. Another option is to examine the minimum and maximum values of each column using Excel's min and max functions. For a more automated approach that considers each record as a unit, clustering techniques could be used to identify clusters of one or a few records that are distant from others. Those records could then be examined.

Missing Values Typically, some records will contain missing values. If the number of records with missing values is small, those records might be omitted. However, if we have a large number of variables, even a small proportion of missing values can affect a lot of records. Even with only 30 variables, if only 5% of the values are missing (spread randomly and independently among cases and variables), almost 80% of the records would have to be omitted from the analysis. (The chance that a given record would escape having a missing value is $0.95^{30} = 0.215$.)

An alternative to omitting records with missing values is to replace the missing value with an imputed value, based on the other values for that variable across all records. For example, ifamong 30 variables, household income is missing for a particular record, we might substitute the mean household income across all records. Doing so does not, of course, add any information about how household income affects the outcome variable. It merely allows us to proceed with the analysis and not lose the information contained in this record for the other 29 variables. Note that using such a technique will understate the variability in a dataset. However, we can assess variability and the performance of our data mining technique, using the validation data, and therefore this need not present a major problem.

Some datasets contain variables that have a very large number of missing values. In other words, a measurement is missing for a large number of records. In that case, dropping records with missing values will lead to a large loss of data. Imputing the missing values might also be useless, as the imputations are based on a small number of existing records. An alternative is to examine the importance of the predictor. If it is not very crucial, it can be dropped. If it is important, perhaps a proxy variable with fewer missing values can be used instead. When such a predictor is deemed central, the best solution is to invest in obtaining the missing data.

Significant time may be required to deal with missing data, as not all situations are susceptible to automated solution. In a messy dataset, for example, a 0 might mean two things: (1) the value is missing, or (2) the value is actually zero. In the credit industry, a 0 in the "past due" variable might mean a customer who is fully paid up, or a customer with no credit history at all—two very different situations. Human judgment may be required for individual cases or to determine a special rule to deal with the situation.

Normalizing (Standardizing) the Data Some algorithms require that the data be normalized before the algorithm can be implemented effectively. To normalize the data, we subtract the mean from each value and divide by the standard deviation of the resulting deviations from the mean. In effect, we are expressing each value as the "number of standard deviations away from the mean," also called a *z-score*.

To consider why this might be necessary, consider the case of clustering. Clustering typically involves calculating a distance measure that reflects how far each record is from a cluster center or from other records. With multiple variables, different units will be used: days, dollars, counts, and so on. If the dollars are in the thousands and everything else is in the tens, the dollar variable will come to dominate the distance measure. Moreover, changing units from (say) days to hours or months could alter the outcome completely.

Data mining software, including XLMiner, typically has an option that normalizes the data in those algorithms where it may be required. It is an option rather than an automatic feature of such algorithms because there are situations where we want each variable to contribute to the distance measure in proportion to its scale.

Use and Creation of Partitions

In supervised learning, a key question presents itself: How well will our prediction or classification model perform when we apply it to new data? We are particularly interested in comparing the performance among various models so that we can choose the one we think will do the best when it is actually implemented.

At first glance, we might think it best to choose the model that did the best job of classifying or predicting the outcome variable of interest with the data at hand. However, when we use the same data both to develop the model and to assess its performance, we introduce bias. This is because when we pick the model that works best with the data, this model's superior performance comes from two sources:

A superior model

Chance aspects of the data that happen to match the chosen model better than they match other models

The latter is a particularly serious problem with techniques (such as trees and neural nets) that do not impose linear or other structure on the data, and thus end up overfitting it.

To address this problem, we simply divide (partition) our data and develop our model using only one of the partitions. After we have a model, we try it out on another partition and see how it performs, which we can measure in several ways. In a classification model, we can count the proportion of held-back records that were misclassified. In a prediction model, we can measure the residuals (errors) between the predicted values and the actual values. We typically deal with two or three partitions: a training set, a validation set, and sometimes an additional test set. Partitioning the data into training, validation, and test sets is done either randomly according to predetermined proportions or by specifying which records go into which partitioning according to some relevant variable (e.g., in time-series forecasting, the data are partitioned according to their chronological order). In most cases, the partitioning should be done randomly to avoid getting a biased partition. It is also possible (although cumbersome) to divide the data into more than three partitions by successive partitioning (e.g., divide the initial data into three partitions, then take one of those partitions and partition it further).

Training Partition The training partition, typically the largest partition, contains the data used to build the various models we are examining. The same training partition is generally used to develop multiple models.

Validation Partition This partition (sometimes called the *test partition)is* used to assess the performance of each model so that you can compare models and pick the best one. In some algorithms (e.g., classification and regression trees), the validation partition may be used in automated fashion to tune and improve the model.

Test Partition This partition (sometimes called the *holdout* or *evaluation partition)* is used if we need to assess the performance of the chosen model with new data.

Why have both a validation and a test partition? When we use the validation data to assess multiple models and then pick the model that does best with the validation data, we again encounter another (lesser) facet of the overfitting problem—chance aspects of the validation data that happen to match the chosen model better than they match other models.

The random features of the validation data that enhance the apparent performance of the chosen model will probably not be present in new data to which the model is applied. Therefore, we may have overestimated the accuracy of our model. The more models we test, the more likely it is that one of them will be particularly effective in explaining the noise in the validation data. Applying the model to the test data, which it has not seen before, will provide an unbiased estimate of how

well it will do with new data. Figure 2.4 shows the three partitions and their use in the data mining process. When we are concerned mainly with finding the best model and less with exactly how well it will do, we might use only training and validation partitions.

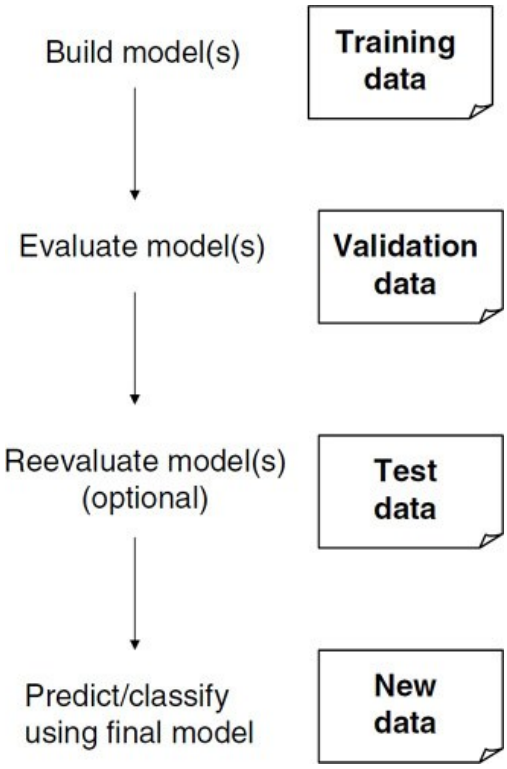

FIGURE 2.4: THREE DATA PARTITIONS AND THEIR ROLE IN THE DATA MINING PROCESS

Note that with some algorithms, such as nearest-neighbor algorithms, the training data itself is the model—records in the validation and test partitions, and in new data, are compared to records in the training data to find the nearest neighbor(s). As As *k*-nearest neighbors is implemented in XLMiner and as discussed in this book, the use of two partitions is an essential part of the classification or prediction process, not merely a way to improve or assess it. Nonetheless, we can still interpret the error in the validation data in the same way that we would interpret error from any other model.

XLMiner has a facility for partitioning a dataset randomly or according to a user-specified variable. For user-specified partitioning, a variable should be created that contains the value *t* (training), *v* (validation), or *s* (test), according to the designation of that record.

2.6 Building a Model: Example with Linear Regression

Let us go through the steps typical to many data mining tasks using a familiar procedure: multiple linear regression. This will help us understand the overall process before we begin tackling new algorithms. We illustrate the Excel procedure using XLMiner.

Boston Housing Data

The Boston housing data contain information on neighborhoods in Boston for which several measurements are taken (e.g., crime rate, pupil/teacher ratio). The outcome variable of interest is the median value of a housing unit in the neighborhood. This dataset has 14 variables, and a description of each variable is given in Table 2.2. A sample of the data is shown in Figure 2.5.

Table 2.2: DESCRIPTION OF VARIABLES IN BOSTON HOUSING DATASET

CRIM	Crime rate

ZN	Percentage of residential land zoned for lots over 25, 000 ft^2
INDUS	Percentage of land occupied by nonretail business
CHAS	Charles River dummy variable (= 1 if tract bounds river; = 0 otherwise)
NOX	Nitric oxide concentration (parts per 10 million)
RM	Average number of rooms per dwelling
AGE	Percentage of owner-occupied units built prior to 1940
DIS	Weighted distances to five Boston employment centers
RAD	Index of accessibility to radial highways
TAX	Full-value property tax rate per $10, 000
PTRATIO	Pupil/teacher ratio by town
B	1000(Bk minus 0.63)^2, where Bk is the proportion of blacks by town
LSTAT	% Lower status of the population
MEDV	Median value of owner-occupied homes in $1000s

CRIM	ZN	INDUS	CHAS	NOX	RM	AGE	DIS	RAD	TAX	PTRATIO	B	LSTAT	MEDV	CAT. MEDV
0.00632	18	2.31	0	0.538	6.58	65.2	4.09	1	296	15.3	396.9	4.98	24	0
0.02731	0	7.07	0	0.469	6.42	78.9	4.97	2	242	17.8	396.9	9.14	21.6	0
0.02729	0	7.07	0	0.469	7.19	61.1	4.97	2	242	17.8	392.83	4.03	34.7	1
0.03237	0	2.18	0	0.458	7	45.8	6.06	3	222	18.7	394.63	2.94	33.4	1
0.06905	0	2.18	0	0.458	7.15	54.2	6.06	3	222	18.7	396.9	5.33	36.2	1
0.02985	0	2.18	0	0.458	6.43	58.7	6.06	3	222	18.7	394.12	5.21	28.7	0
0.08829	13	7.87	0	0.524	6.01	66.6	5.56	5	311	15.2	395.6	12.43	22.9	0
0.14455	13	7.87	0	0.524	6.17	96.1	5.95	5	311	15.2	396.9	19.15	27.1	0
0.21124	13	7.87	0	0.524	5.63	100	6.08	5	311	15.2	386.63	29.93	16.5	0

FIGURE 2.5: FIRST NINE RECORDS IN THE BOSTON HOUSING DATA

The first row in the data represents the first neighborhood, which had an average per capita crime rate of 0.006, had 18% of the residential land zoned for lots over 25, 000 square feet (ft^2), 2.31% of the land devoted to nonretail business, no border on the Charles River, and so on.

Modeling Process

We now describe in detail the various model stages using the Boston housing example.

1. *Purpose*. Let us assume that the purpose of our data mining project is to predict the median house value in small Boston area neighborhoods.

2. *Obtain the Data*. We will use the Boston housing data. The dataset in question is small enough that we do not need to sample from it—we can use it in its entirety.

3. *Explore, Clean, and Preprocess the Data*. Let us look first at the description of the variables (e.g., crime rate, number of rooms per dwelling) to be sure that we understand them all. These descriptions are available on the "description" tab on the worksheet, as is a Web source for the dataset. They all seem fairly straightforward, but this is not always the case. Often, variable names are cryptic and their descriptions may be unclear or missing.

   It is useful to pause and think about what the variables mean and whether they should be included in the model. Consider the variable TAX. At first glance, we consider that the tax on a home is usually a function of its assessed value, so there is some circularity in the model— we want to predict a home's value using TAX as a predictor, yet TAX itself is determined by a home's value. TAX might be a very good predictor of home value in a numerical sense, but would it be useful if we wanted to apply our model to homes whose assessed value might not be known? Reflect, though, that the TAX variable, like all the variables, pertains to the average in a neighborhood, not to individual homes. Although the purpose of our inquiry has not been spelled out, it is possible that at some stage we might want to apply a model to individual homes, and in such a case, the neighborhood TAX value would be a useful predictor. So we will keep TAX in the analysis for now.

   In addition to these variables, the dataset also contains an additional variable, CAT.MEDV, which has been created

by categorizing median value (MEDV) into two categories, high and low. (There are a couple of aspects of MEDV, the median house value, that bear noting. For one thing, it is quite low, since it dates from the 1970s. For another, there are a lot of 50s, the top value. It could be that median values above $50, 000 were recorded as $50, 000.) The variable CAT.MEDV is actually a categorical variable created from MEDV. If MEDV $\geq$ $30, 000, CAT.MEDV = 1. If MEDV $\leq$ $30, 000, CAT.MEDV = 0. If we were trying to categorize the cases into high and low median values, we would use CAT.MEDV instead of MEDV. As it is, we do not need CAT.MEDV, so we leave it out of the analysis. We are left with 13 independent (predictor) variables, which can all be used.

It is also useful to check for outliers that might be errors. For example, suppose that the RM (number of rooms) column looked like the one in Figure 2.6, after sorting the data in descending order based on rooms. We can tell right away that the 79.29 is in error—no neighborhood is going to have houses that have an average of 79 rooms. All other values are between 3 and 9. Probably, the decimal was misplaced and the value should be 7.929. (This hypothetical error is not present in the dataset supplied with XLMiner.)

RM	AGE	DIS
79.29	96.2	2.04
8.78	82.9	1.90
8.75	83	2.89
8.70	88.8	1.00

FIGURE 2.6: OUTLIER IN BOSTON HOUSING DATA

4. *Reduce the Data and Partition Them into Training, Validation, and Test Partitions*. Our dataset has only 13 variables, so data reduction is not required. If we had many more variables, at this stage we might want to apply a variable reduction technique such as principal components analysis to consolidate multiple similar variables into a smaller number of variables. Our task is to predict the median house value and then assess how well that prediction does. We will partition the data into a training set to build the model and a validation set to see how well the model does. This technique is part of the "supervised learning" process in classification and prediction problems. These are problems in which we know the class or value of the outcome variable for some data, and we want to use those data in developing a model that can then be applied to other data where that value is unknown.

   In Excel, select *XLMiner* > *Partition* and the dialog box shown in Figure 2.7 appears. Here we specify which data range is to be partitioned and which variables are to be included in the partitioned dataset. The partitioning can be handled in one of two ways:

   a. The dataset can have a partition variable that governs the division into training and validation partitions (e.g., 1 = training, 2 = validation).

   b. The partitioning can be done randomly. If the partitioning is done randomly, we have the option of specifying a seed for randomization (which has the advantage of letting us duplicate the same random partition later should we need to). In this example, a seed of 54 is used.

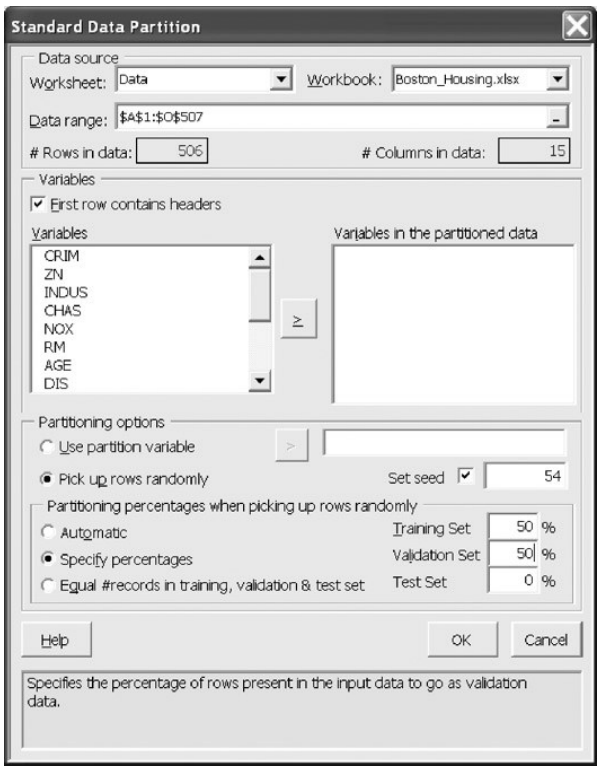

FIGURE 2.7: PARTITIONING THE DATA. THE DEFAULT IN XLMINER PARTITIONS THE DATA INTO 6O% TRAINING DATA, 40% VALIDATION DATA, AND O% TEST DATA. IN THIS EXAMPLE, A PARTITION OF 50% TRAINING AND 50% VALIDATION IS USED

In this case we divide the data into two partitions: training and validation. The training partition is used to build the model, and the validation partition is used to see how well the model does when applied to new data. We need to specify the percent of the data used in each partition.

Note Although we are not using it here, a test partition might also be used.

Typically, a data mining endeavor involves testing multiple models, perhaps with multiple settings on each model. When we train just one model and try it out on the validation data, we can get an unbiased idea of how it might perform on more such data. However, when we train many models and use the validation data to see how each one does, then choose the best-performing model, the validation data no longer provide an unbiased estimate of how the model might do with more data. By playing a role in choosing the best model, the validation data have become part of the model itself. In fact, several algorithms (e.g., classification and regression trees) explicitly factor validation data into the model-building algorithm itself (e.g., in pruning trees). Models will almost always perform better with the data they were trained on than with fresh data. Hence, when validation data are used in the model itself, or when they are used to select the best model, the results achieved with the validation data, just as with the training data, will be overly optimistic.

The test data, which should not be used in either the model-building or model selection process, can give a better estimate of how well the chosen model will do with fresh data. Thus, once we have selected a final model, we apply it to the test data to get an estimate of how well it will actually perform.

5. *Determine the Data Mining Task*. In this case, as noted, the specific task is to predict the value of MEDV using the 13 predictor variables.

6. *Choose the Technique*. In this case, it is multiple linear regression. Having divided the data into training and validation partitions, we can use XLMiner to build a multiple linear regression model with the training data. We want to predict median house price on the basis of all the other values.

7. *Use the Algorithm to Perform the Task*. In XLMiner, we select *Prediction > Multiple Linear Regression*, as shown in Figure 2.8. The variable MEDV is selected as the output (dependent) variable, the variable CAT.MEDV is left unused, and the remaining variables are all selected as input (independent or predictor) variables. We ask XLMiner to show us the fitted values on the training data as well as the predicted values (scores) on the validation data, as shown in Figure 2.9. XLMiner produces standard regression output, but for now we defer that as well as the more advanced

options displayed above. (See Chapter 6 or the user documentation for XLMiner for more information.) Rather, we review the predictions themselves. Figure 2.10 shows the predicted values for the first few records in the training data along with the actual values and the residual (prediction error). Note that the predicted values would often be called the *fitted values* since they are for the records to which the model was fit. The results for the validation data are shown in Figure 2.11. The prediction error for the training and validation data are compared in Figure 2.12.

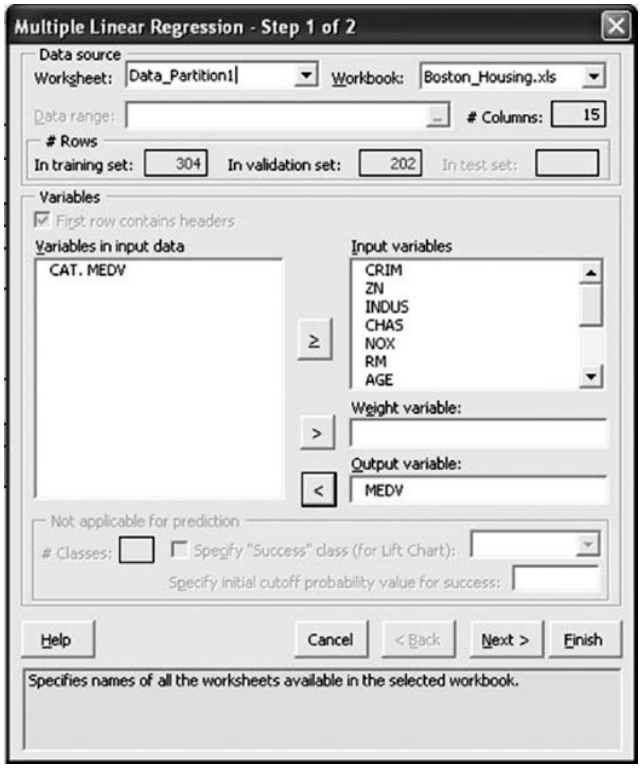

FIGURE 2.8: USING XLMINER FOR MULTIPLE LINEAR REGRESSION

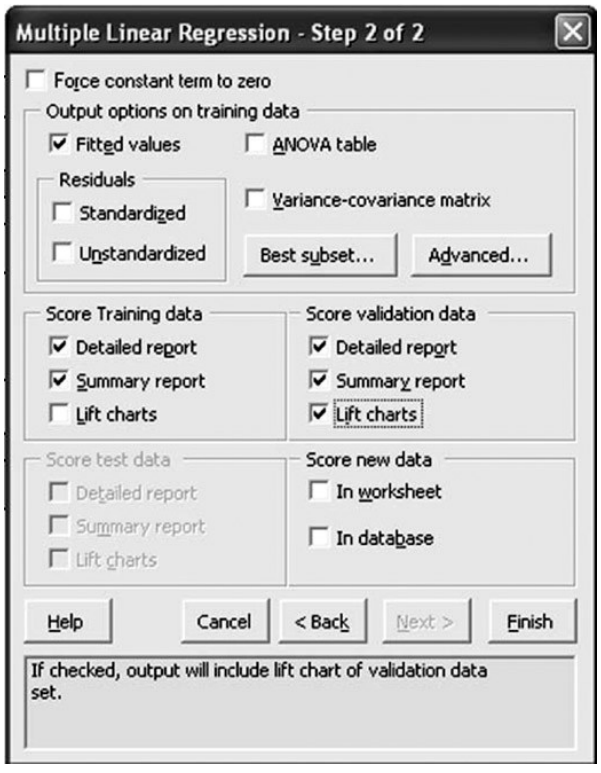

FIGURE 2.9: SPECIFYING THE OUTPUT

XLMiner : Multiple Linear Regression - Prediction of Training Data

Data range ['Boston_Housing']'Data_Partition2'!$C$19:$P$271 Back to

Row Id.	Predicted Value	Actual Value	Residual	CRIM	ZN	INDUS	CHAS	NOX	RM	AGE	DIS	RAD	TAX	PTRATIO	B	LSTAT
1	30.02788078	24	-6.027880779	0.00632	18	2.31	0	0.538	6.575	65.2	4.09	1	296	15.3	396.9	4.98
2	24.90910941	21.6	-3.309109407	0.02731	0	7.07	0	0.469	6.421	78.9	4.9671	2	242	17.8	396.9	9.14
3	30.9549987	34.7	3.745001299	0.02729	0	7.07	0	0.469	7.185	61.1	4.9671	2	242	17.8	392.83	4.03
4	28.07549961	33.4	5.324500385	0.03237	0	2.18	0	0.458	6.998	45.8	6.0622	3	222	18.7	394.63	2.94
5	27.9091436	36.2	8.290856402	0.06905	0	2.18	0	0.458	7.147	54.2	6.0622	3	222	18.7	396.9	5.33
7	23.65328843	22.9	-0.753288432	0.08829	12.5	7.87	0	0.524	6.012	66.6	5.5605	5	311	15.2	395.6	12.43
8	21.11420949	27.1	5.98579051	0.14455	12.5	7.87	0	0.524	6.172	96.1	5.9505	5	311	15.2	396.9	19.15
11	21.09801414	15	-6.09801414	0.22489	12.5	7.87	0	0.524	6.377	94.3	6.3467	5	311	15.2	392.52	20.45
12	22.12589464	18.9	-3.225894637	0.11747	12.5	7.87	0	0.524	6.009	82.9	6.2267	5	311	15.2	396.9	13.27
14	17.61047844	20.4	2.789521563	0.62976	0	8.14	0	0.538	5.949	61.8	4.7075	4	307	21	396.9	8.26
21	11.62929772	13.6	1.970702281	1.25179	0	8.14	0	0.538	5.57	98.1	3.7979	4	307	21	376.57	21.02

FIGURE 2.10: PREDICTIONS FOR THE TRAINING DATA

XLMiner : Multiple Linear Regression - Prediction of Validation Data

Data range ['Boston_Housing']'Data_Partition2'!$C$272:$P$524 Back to

Row Id.	Predicted Value	Actual Value	Residual	CRIM	ZN	INDUS	CHAS	NOX	RM	AGE	DIS	RAD	TAX	PTRATIO	B	LSTAT
6	24.09753481	28.7	4.602465185	0.02985	0	2.18	0	0.458	6.43	58.7	6.0622	3	222	18.7	394.12	5.21
9	14.04293006	16.5	2.457069944	0.21124	12.5	7.87	0	0.524	5.631	100	6.0821	5	311	15.2	386.63	29.93
10	20.02731265	18.9	-1.127312654	0.17004	12.5	7.87	0	0.524	6.004	85.9	6.5921	5	311	15.2	386.71	17.1
13	22.31053056	21.7	-0.610530557	0.09378	12.5	7.87	0	0.524	5.889	39	5.4509	5	311	15.2	390.5	15.71
15	17.56538825	18.2	0.634611749	0.63796	0	8.14	0	0.538	6.096	84.5	4.4619	4	307	21	380.02	10.26
16	17.30239527	19.9	2.597604726	0.62739	0	8.14	0	0.538	5.834	56.5	4.4986	4	307	21	395.62	8.47
17	18.71477307	23.1	4.385226932	1.05393	0	8.14	0	0.538	5.935	29.3	4.4986	4	307	21	386.85	6.58
18	15.81653483	17.5	1.683465174	0.7842	0	8.14	0	0.538	5.99	81.7	4.2579	4	307	21	386.75	14.67
19	14.38144028	20.2	5.818559715	0.80271	0	8.14	0	0.538	5.456	36.6	3.7965	4	307	21	288.99	11.69
20	16.554546	18.2	1.645453996	0.7258	0	8.14	0	0.538	5.727	69.5	3.7965	4	307	21	390.95	11.28
22	16.30306818	19.6	3.296931816	0.85204	0	8.14	0	0.538	5.965	89.2	4.0123	4	307	21	392.53	13.83
25	14.58925181	15.6	1.010748192	0.75026	0	8.14	0	0.538	5.924	94.1	4.3996	4	307	21	394.33	16.3

FIGURE 2.11: PREDICTIONS FOR THE VALIDATION DATA

Training Data scoring - Summary Report

Total sum of squared errors	RMS Error	Average Error
4136.091425	4.043289187	-1.12501E-06

(a)

Validation Data scoring - Summary Report

Total sum of squared errors	RMS Error	Average Error
8117.85953	5.66448597	0.961240934

(b)

FIGURE 2.12: ERROR RATES FOR (*A*) TRAINING AND (*B*) VALIDATION DATA (ERROR FIGURES ARE IN THOUSANDS OF $)

Prediction error can be measured in several ways. Three measures produced by XLMiner are shown in Figure 2.12. On the right is the *average error*, simply the average of the residuals (errors). In both cases it is quite small relative to the units of MEDV, indicating that, on balance, predictions average about right—our predictions are "unbiased." Of course, this simply means that the positive and negative errors balance out. It tells us nothing about how large these errors are.

The *total sum of squared errors* on the left adds up the squared errors, so whether an error is positive or negative, it contributes just the same. However, this sum does not yield information about the size of the typical error.

The *RMS error* (root-mean-squared error) is perhaps the most useful term of all. It takes the square root of the average squared error; thus, it gives an idea of the typical error (whether positive or negative) in the same scale as

that used for the original data. As we might expect, the RMS error for the validation data (5.66 thousand $), which the model is seeing for the first time in making these predictions, is larger than for the training data (4.04 thousand $), which were used in training the model.

8. *Interpret the Results*. At this stage we would typically try other prediction algorithms (e.g., regression trees) and see how they do errorwise. We might also try different "settings" on the various models (e.g., we could use the *best subsets* option in multiple linear regression to chose a reduced set of variables that might perform better with the validation data). After choosing the best model (typically, the model with the lowest error on the validation data while also recognizing that "simpler is better"), we use that model to predict the output variable in fresh data. These steps are covered in more detail in the analysis of cases.

9. *Deploy the Model*. After the best model is chosen, it is applied to new data to predict MEDV for records where this value is unknown. This was, of course, the overall purpose.

2.7 Using Excel for Data Mining

An important aspect of this process to note is that the heavy-duty analysis does not necessarily require huge numbers of records. The dataset to be analyzed may have millions of records, of course, but in doing multiple linear regression or applying a classification tree, the use of a sample of 20, 000 is likely to yield as accurate an answer as that obtained when using the entire dataset. The principle involved is the same as the principle behind polling: Ifsampled judiciously, 2000 voters can give an estimate of the entire population's opinion within one or two percentage points. (See How Many Variables and How Much Data in Section 2.5 for further discussion.)

Therefore, in most cases, the number of records required in each partition (training, validation, and test) can be accommodated within the rows allowed by Excel. Of course, we need to get those records into Excel, and for this purpose the standard version of XLMiner provides an interface for random sampling of records from an external database.

Similarly, we need to apply the results of our analysis to a large database, and for this purpose the standard version of XLMiner has a facility for storing models and scoring them to an external database. For example, XLMiner would write an additional column (variable) to the database consisting of the predicted purchase amount for each record.

XLMiner has a facility fordrawinga sample from an external database. The sample can be drawn at random or it can be stratified. It also has a facility to score data in the external database using the model that was obtained from the training data.

DATA MINING SOFTWARE TOOLS: THE STATE OF THE MARKET BY HERB EDELSTEIN[1]

Data mining uses a variety of tools to discover patterns and relationships in data that can be used to explain the data or make meaningful predictions. The need for ever more powerful tools is driven by the increasing breadth and depth of analytical problems. In order to deal with tens of millions of cases (rows) and hundreds or even thousands of variables (columns), organizations need scalable tools. A carefully designed GUI (graphical user interface) also makes it easier to create, manage, and apply predictive models.

Data mining is a complete process, not just a particular technique or algorithm. Industrial-strength tools support all phases of this process, handle all sizes of databases, and manage even the most complex problems.

The software must first be able to pull all the data together. The data mining tool may need to access multiple databases across different database management systems. Consequently, the software should support joining and subsetting of data from a range of sources. Because some of the data may be a terabyte or more, the software also needs to support a variety of sampling methodologies.

Next, the software must facilitate exploring and manipulating the data to create understanding and suggest a starting point for model building. When a database has hundreds or thousands of variables, it becomes an enormous task to select the variables that best describe the data and lead to the most robust predictions. Visualization tools can make it easier to identify the most important variables and find meaningful patterns in very large databases. Certain algorithms are particularly suited to guiding the selection of the most relevant variables. However, often the best predictors are not the variables in the database themselves, but some mathematical combination of these variables. This not only increases the number of variables to be evaluated, but the more complex transformations require a scripting language.

Frequently, the data access tools use the DBMS language itself to make transformations directly on the underlying database.

Because building and evaluating models is an iterative process, a dozen or more exploratory models may be built before settling on the best model. While any individual model may take only a modest amount of time for the software to construct, computer usage can really add up unless the tool is running on powerful hardware. Although some people consider this phase to be what data miningis all about, it usually represents a relatively small part of the total effort.

Finally, after building, testing, and selecting the desired model, it is necessary to deploy it. A model that was built using a small subset of the data may now be applied to millions of cases or integrated into a real-time application, processing hundreds of transactions each second. For example, the model may be integrated into credit scoring or fraud detection applications. Over time the model should be evaluated and refined as needed.

Data mining tools can be general purpose (either embedded in a DBMS or stand-alones) orthey can be application specific.

All the major database management system vendors have incorporated data mining capabilities into their products. Leading products include IBM DB2 Intelligent Miner, Microsoft SQL Server 2005, Oracle Data Mining, and Teradata Warehouse Miner. The target user for embedded data mining is a database professional. Not surprisingly, these products take advantage of database functionality, including using the DBMS to transform variables, storing models in the database, and extending the data access language to include model building and scoring the database. A few products also supply a separate graphical interface for building data mining models. Where the DBMS has parallel processing capabilities, embedded data mining tools will generally take advantage of it, resulting in better performance. As with the data mining suites described below, these tools offer an assortment of algorithms.

Stand-alone data miningtools can be based on a single algorithm ora collection of algorithms called a suite. Target users include both statisticians and analysts. Well-known single-algorithm products include KXEN; RuleQuest Research C5.0; and Salford Systems CART, MARS, and Treenet. Most of the top single-algorithm tools have also been licensed to suite vendors. The leading suites include SAS Enterprise Miner, IBM Modeler (previously SPSS Clementine), and Spotfire Miner (previously Insightful Miner). Suites are characterized by providing a wide range of functionality and an interface designed to enhance model-building productivity. Many suites have outstandingvisualization tools and links to statistical packages that extend the range of tasks they can perform, and most provide a procedural scripting language for more complex transformations. They use a graphical workflow interface to outline the entire data mining process. The suite vendors are working to link their tools more closely to underlying DBMSs; for example, data transformations might be handled by the DBMS. Data mining models can be exported to be incorporated into the DBMS either through generating SQL, procedural language code (e.g., C++ or Java), ora standardized data mining model language called Predictive Model Markup Language (PMML).

Application-specific tools, in contrast to the other types, are intended for particular analytic applications such as credit scoring, customer retention, or product marketing. Their focus may be further sharpened to address the needs of certain markets such as mortgage lendingorfinancial services. The target user is an analyst with expertise in the applications domain. Therefore, the interfaces, the algorithms, and even the terminology are customized for that particular industry, application, or customer. While less flexible than general-purpose tools, they offer the advantage of already incorporating domain knowledge into the product design and can provide very good solutions with less effort. Data mining companies including SAS and SPSS offer vertical market tools, as do industry specialists such as Fair Isaac.

The tool used in this book, XLMiner, is asuite with both sampling and scoring capabilities. While Excel itself is not a suitable environment for dealing with thousands of columns and millions of rows, it is a familiar workspace to business analysts and can be used as a work platform to support othertools. An Excel add-in such as XLMiner (which uses non-Excel computational engines) is user friendly and can be used in conjunction with samplingtechniques for prototyping, small-scale, and educational applications of data mining.

---

[1]Herb Edelstein is president of Two Crows Consulting (www.twocrows.com), a leading data mining consulting firm near Washington, D.C. He is an internationally recognized expert in data mining and data warehousing, a widely published author on these topics, and a popular speaker. 2006 Herb Edelstein.

Problems

2.1 Assuming that data mining techniques are to be used in the following cases, identify whether the task required is supervised or unsupervised learning.

a. Deciding whether to issue a loan to an applicant based on demographic and financial data (with reference to a database of similar data on prior customers).

b. In an online bookstore, making recommendations to customers concerning additional items to buy based on the buying patterns in prior transactions.

c. Identifying a network data packet as dangerous (virus, hacker attack) based on comparison to other packets whose threat status is known.

d. Identifying segments of similar customers.

e. Predicting whether a company will go bankrupt based on comparing its financial data to those of similar bankrupt and nonbankrupt firms.

f. Estimating the repair time required for an aircraft based on a trouble ticket.

g. Automated sorting of mail by zip code scanning.

h. Printing of custom discount coupons at the conclusion of a grocery store checkout based on what you just bought and what others have bought previously.

2.2 Describe the difference in roles assumed by the validation partition and the test partition.

2.3 Consider the sample from a database of credit applicants in Figure 2.13. Comment on the likelihood that it was sampled randomly, and whether it is likely to be a useful sample.

OBS#	CHK_ACCT	DURATION	HISTORY	NEW_CAR	USED_CAR	FURNITURE	RADIO/TV	EDUCATION	RETRAINING	AMOUNT	SAV_ACCT	RESPONSE
1	0	6	4	0	0	0	1	0	0	1169	4	1
8	1	36	2	0	1	0	0	0	0	6948	0	1
16	0	24	2	0	0	0	1	0	0	1282	1	0
24	1	12	4	0	1	0	0	0	0	1804	1	1
32	0	24	2	0	0	1	0	0	0	4020	0	1
40	1	9	2	0	0	0	1	0	0	458	0	1
48	0	6	2	0	1	0	0	0	0	1352	2	1
56	3	6	1	1	0	0	0	0	0	783	4	1
64	1	48	0	0	0	0	0	0	1	14421	0	0
72	3	7	4	0	0	0	1	0	0	730	4	1
80	1	30	2	0	0	1	0	0	0	3832	0	1
88	1	36	2	0	0	0	0	1	0	12612	1	0
96	1	54	0	0	0	0	0	0	1	15945	0	0
104	1	9	4	0	0	1	0	0	0	1919	0	1
112	2	15	2	0	0	0	0	1	0	392	0	1

FIGURE 2.13: SAMPLE FROM A DATABASE OF CREDIT APPLICANTS

2.4 Consider the sample from a bank database shown in Figure 2.14; it was selected randomly from a larger database to be the training set. Personal Loan indicates whether a solicitation for a personal loan was accepted and is the response variable. A campaign is planned for a similar solicitation in the future, and the bank is looking for a model that will identify likely responders. Examine the data carefully and indicate what your next step would be.

ID	Age	Experience	Income	ZIP Code	Family	CCAvg	Educ.	Mortgage	Personal Loan	Securities Account
1	25	1	49	91107	4	1.60	1	0	0	1
4	35	9	100	94112	1	2.70	2	0	0	0
5	35	8	45	91330	4	1.00	2	0	0	0
6	37	13	29	92121	4	0.40	2	155	0	0
9	35	10	81	90089	3	0.60	2	104	0	0
11	65	39	105	94710	4	2.40	3	0	0	0
12	29	5	45	90277	3	0.10	2	0	0	0
18	42	18	81	94305	4	2.40	1	0	0	0
20	55	28	21	94720	1	0.50	2	0	0	1
23	29	5	62	90277	1	1.20	1	260	0	0
26	43	19	29	94305	3	0.50	1	97	0	0
27	40	16	83	95064	4	0.20	3	0	0	0
29	56	30	48	94539	1	2.20	3	0	0	0
31	59	35	35	93106	1	1.20	3	122	0	0
32	40	16	29	94117	1	2.00	2	0	0	0
35	31	5	50	94035	4	1.80	3	0	0	0
36	48	24	81	92647	3	0.70	1	0	0	0
37	59	35	121	94720	1	2.90	1	0	0	0
38	51	25	71	95814	1	1.40	3	198	0	0
40	38	13	80	94115	4	0.70	3	285	0	0
41	57	32	84	92672	3	1.60	3	0	0	1

FIGURE 2.14: SAMPLE FROM A BANK DATABASE

2.5 Using the concept of overfitting, explain why when a model is fit to training data, zero error with those data is not necessarily good.

2.6 In fitting a model to classify prospects as purchasers or nonpurchasers, a certain company drew the training data from internal data that include demographic and purchase information. Future data to be classified will be lists purchased from other sources, with demographic (but not purchase) data included. It was found that "refund issued" was a useful predictor in the training data. Why is this not an appropriate variable to include in the model?

2.7 A dataset has 1000 records and 50 variables with 5% of the values missing, spread randomly throughout the records and variables. An analyst decides to remove records that have missing values. About how many records would you expect would be removed?

2.8 Normalize the data in Table 2.3, showing calculations.

Table 2.3

Age	Income ($)
25	49, 000
56	156, 000
65	99, 000
32	192, 000
41	39, 000
49	57, 000

2.9 2.9 Statistical distance between records can be measured in several ways. Consider Euclidean distance, measured as the square root of the sum of the squared differences. For the first two records in Table 2.3, it is

$$\sqrt{(25-56)^2+(49,000-156,000)^2}.$$

Does normalizing the data change which two records are farthest from each other in terms of Euclidean distance?

2.10 Two models are applied to a dataset that has been partitioned. Model A is considerably more accurate than model B on the training data but slightly less accurate than model B on the validation data. Which model are you more likely to consider for final deployment?

2.11 The dataset ToyotaCorolla.xls contains data on used cars on sale during the late summer of 2004 in The Netherlands. It has 1436 records containing details on 38 attributes, including Price, Age, Kilometers, HP, and other specifications.

a. Explore the data using the data visualization (matrix plot) capabilities of XLMiner. Which of the pairs among the variables seem to be correlated?

b. We plan to analyze the data using various data mining techniques described in future chapters. Prepare the data for use as follows:
    i. The dataset has two categorical attributes, Fuel Type and Metallic.
        a. Describe how you would convert these to binary variables.
        b. Confirm this using XLMiner's utility to transform categorical data into dummies.
        c. How would you work with these new variables to avoid including redundant information in models?
    ii. Prepare the dataset (as factored into dummies) for data mining techniques of supervised learning by creating partitions using XLMiner's data partitioning utility. Select all the variables and use default values for the random seed and partitioning percentages for training (50%), validation (30%), and test (20%) sets. Describe the roles that these partitions will play in modeling.