Chapter Nine

Classification Models: The Most Used Models in Analytics

"The key to big data is whether it's going to give you actionable insights that you can then grow your business on."

Xavier Amatriain, Netflix's director of algorithms engineering[1]

LEARNING OBJECTIVES

After reading this chapter, you should be able to:

1. Explain the use of classification algorithms.
2. Construct and employ four different classification algorithms.
3. Interpret the standard diagnostic statistics for these algorithms.
4. Explain how better decisions can be made through the use of classification algorithms.
5. Relate the pros and cons of each of the classification algorithms.
6. Apply classification algorithms to make business predictions.

INTRODUCTION

The statistical forecasting models we have introduced previously in this text, have been applied to continuous, numeric target variables. For instance, we used exponential smoothing and regression to predict sales at The Gap. This chapter introduces the situation in which we wish to assign a category to each instance of the target. These classification algorithms will be data driven in the sense that we will not use assumptions about the structure of the data as we have used prior

[1] James Willhite, "Getting Started in 'Big Data,'" *Wall Street Journal*, February 4, 2014. (https://blogs.wsj.com/cfo/2014/02/04/getting-started-in-big-data/).

to chapter eight. So these are not model driven algorithms; they are referred to as data driven algorithms.

Classification algorithms are the most used techniques in predictive analytics.

Classification algorithms are the most used techniques in predictive analytics. That fact is probably due to the value to firms in making correct classification predictions. The use of classification algorithms is not limited to any single industry; consumer products, entertainment, health care, and the fast food industry all use classification effectively. These industries (and many others) mine mountains of data trying to smooth supply chains, improve product design, enhance customer service, and add to the company's bottom line. Classification models are not all there is to predictive analytics, but there is a reason why they are the most used of the techniques.

Douglas Laney of Gartner Inc. reported an astounding finding that should probably convince any firm that there is real value in using predictive analytics.[2] Laney and a colleague, Somendra Tripathi, compared firms that heavily used predictive analytics and those that did not. What they measured was the Tobin's q of each firm. Tobin's q is the market-to-book value of a firm; it represents the market's estimate of the value of the firm compared to the accounting or book value. If one firm's Tobin's q is higher than another's, it indicates that the market (comprised of thousands, perhaps hundreds of thousands, of investors) has identified the one firm as more valuable than the other firm regardless of accounting (i.e., historical) value. What Laney and Tripathi reported was that the Tobin's q of firms heavily using predictive analytics was 200 to 300 percent higher than the norm. The Tobin's q for information-centered firms (those in the business of selling data) using predictive analytics were higher yet: 400 to 500 percent above the average. Apparently, the market recognizes the value of firms using predictive analytics. The market value of these firms is substantially enhanced as a result of the leverage they gain by using predictive analytics. This information, with the knowledge that the most used of the data mining algorithms are in the classification category, should provide an incentive to discover how the classification algorithms presented in this chapter work and how they are employed by firms.

The Tobin's q of firms heavily using predictive analytics was 200 to 300 percent higher than the norm.

Classification is a form of supervised learning in which we wish to predict the class (classification) of each record in our data: our statistical forecasts were also a form of supervised learning, but in those forecasts, the prediction took the form of a continuous outcome variable.

When Vermont Country Store (a largely nostalgic catalog mail order company) sends a catalog to a prospective customer, they would like to know that they are mailing the costly catalog to a person or family likely to purchase from the catalog; they are classifying prospective customers as "likely to purchase" or "not likely to purchase" and contacting only the likely purchasers. John Wanamaker, a Philadelphia retailer who was also the U.S. Postmaster General, once is reputed

[2] Douglas Laney, "The Hidden Shareholder Boost from Information Assets," *Forbes*, July 21, 2014, (https://www.forbes.com/sites/gartnergroup/2014/07/21/the-hidden-shareholder-boost-from-information-assets/#698bdd397628).

to have said, "Half the money I spend on advertising is wasted; the trouble is I don't know which half." If Wanamaker had been able to use the classification algorithms used by Vermont Country Store, he might have known which half of his advertising was wasted.

Amazon also uses classification algorithms; up to 60 percent of Amazon sales results from up-selling or cross-selling, according to *Fortune*.[3] Amazon is predicting which customers would likely be susceptible to an offer of a more expensive version of a product or a compatible product; apparently, they are correct more often than not.

eHarmony helps people meet each other; their mantra is "Beat the odds, bet on love." The implication is that the online dating site can suggest individuals to you who have a high probability of being compatible. They have attributes gleaned from a lengthy questionnaire that each of their users has filled out, and they are classifying which individuals in their database may likely be compatible with you. You are scored as being either compatible or not compatible with another individual by the eHarmony classification algorithm. eHarmony CEO Neil Clark Warren believes the site is a better way of meeting possible partners than the random process of letting luck determine your fate.

Whirlpool Corporation uses classification algorithms to predict which dishwashers coming down the assembly line are likely to fail in some manner within one year. Warranty repairs on these failing machines is very costly and erodes profit. By using a series of test results performed along the assembly line as attributes and classifying the dishwashers as either "likely to fail" or "unlikely to fail," the company is able to significantly reduce warranty claims and has probably increased customer satisfaction as well.

Each of these examples has a common feature: what is being predicted is a "class," or one of a few categories. The prediction does not involve the forecast of a continuous variable but it is still prediction. As such, the tools you learned in earlier chapters will be of little use here; classification models are, like those earlier models, a form of supervised learning, but they will require algorithms that are quite different than you learned in earlier chapters.

The types of things for which categories are predicted vary widely. Firms will often gain value by making predictions for individuals: they may make a decision who to send a mailing to; they may decide who to lend to; they may choose who to investigate for crime or fraud; and they may decide to treat one medical patient differently than another. In each case, however, the prediction that is being made is to place an individual into one category or another; it is not to predict a numeric value such as dollars of sales.

The value of being able to make such categorizations is quite important. If I send an advertising mailer to an individual likely to use my services or purchase my product, the outcome will probably be better than if I sent the same

[3] J. P. Mangalindan, "Amazon's Recommendation Secret," *Fortune*, July 30, 2012, http://fortune.com/2012/07/30/amazons-recommendation-secret/.

The types of things for which categories are predicted vary widely. Firms will often gain value by making predictions for individuals.

information to an individual who was unlikely to ever take advantage of the offer. I would rather lend to an individual who is likely to repay the loan than to an individual likely to default. If I am going to spend time and effort to investigate someone for crime or fraud, I would like to know that the individual is likely to be guilty and my efforts will have been worthwhile. If I choose to treat a patient with a drug known to be effective in curing a certain disease, I would like to know that the probability is high that the person I am considering treating actually has the disease. When correct category decisions are made, firms sell more product, banks suffer fewer defaults, crime fighting is toughened, and health care is made more robust.

Firms and their customers, suppliers, and financiers benefit when these correct categories are predicted. eHarmony tries to predict whether you will be compatible with another individual; if they get the category (i.e., compatible or not compatible) predicted correctly, you might benefit for the remainder of your life. Linkedin makes suggestions of people you may know in the hopes that a formal connection on their social media platform may prove useful to both of you. It is a category that Linkedin is predicting; either you know someone or you don't. Amazon will often display products you may wish to purchase; it is making a category prediction and placing you in the "might purchase" category. Remember that estimates indicate that a significant portion of Amazon sales result from such predicted categories.[4] Target predicted the pregnancy of some of its customers in 2011 and acted upon the information by sending coupons appropriate for a mother-to-be to individuals identified.[5] In 2017, the BBC announced that the biggest killer you may not know was sepsis; it kills more people each year than bowel, breast, and prostate cancer combined.[6] The Sisters of Mercy Health Systems uses patient vital signs to accurately categorize patients with sepsis and those without sepsis so that "likely sepsis" patients can be treated earlier than otherwise was available and to improve their chances of survival. Early detection would seem to be a priority since sepsis costs hospitals between $28 and $33 billion dollars per year, according to Booz Allen Hamilton.[7] The Chicago Police Department tried to determine which individuals would be shot in the near future.[8] An analytics algorithm assigned a category based on arrests, shootings, affiliations with gang members, and other attributes. It produced a list predicting who was most likely to

[4] Ibid.

[5] Charles Duhigg, "How Companies Learn Your Secrets," *New York Times*, February 16, 2012 (http://www.nytimes.com/2012/02/19/magazine/shopping-habits.html).

[6] James Gallagher, "The Biggest Killer You May Not Know," *BBC*, March 10, 2017 (http://www.bbc.com/news/health-39219765).

[7] Mark Adams, Reechlyk Chatterjee, and Sharma Yugal, "Improving Patient Outcomes Through Advanced Biomedical Analytics," Booz Allen Hamilton, March 2012 (https://www.boozallen.com/content/dam/boozallen/media/file/leveraging-advanced-data-collection-cs.PDF).

[8] "Monica Davey, "Chicago Police Try to Predict Who May Shoot or Be Shot," *New York Times*, May 23, 2016, (https://www.nytimes.com/2016/05/24/us/armed-with-data-chicago-police-try-to-predict-who-may-shoot-or-be-shot.html?_r=0).

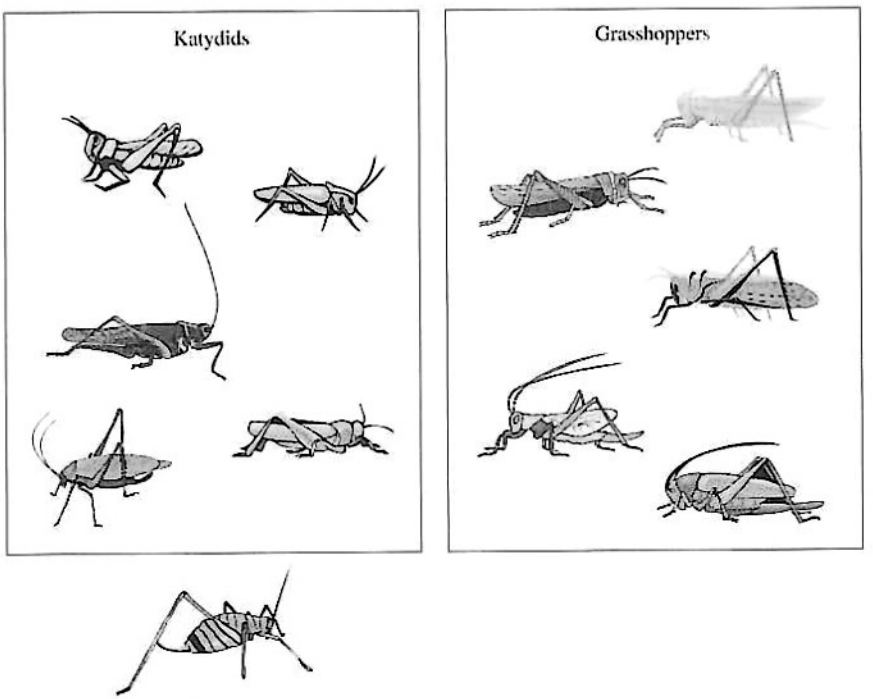
be shot soon or to shoot someone. The algorithm proved all too predictive for one unfortunate man named Shaquon Thomas. His name appeared on the list as likely to be shot right after the department began to use the classification algorithm. The 19-year-old was fatally shot shortly afterward on May 29, 2015.

A DATA MINING CLASSIFICATION EXAMPLE: k-NEAREST-NEIGHBOR (kNN)

Many data mining techniques are able to be shown in graphic form, and this makes them easier to understand.

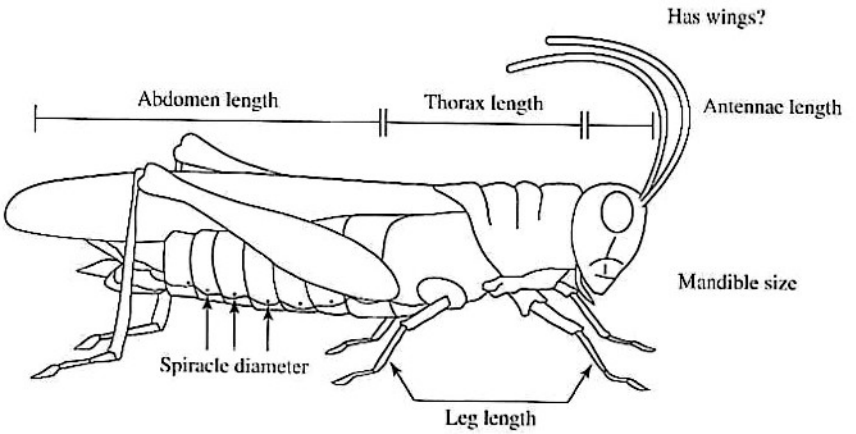
Consider the following data mining example from Eamonn Keogh; while it is not business related, it is easy to see the technique unfold visually. Many data mining techniques are able to be shown in graphic form, and this makes them easier to understand. We will attempt to use a graphical approach for explanation when possible. Suppose you are a researcher attempting to classify insects you have found into one of two groups (i.e., you are attempting to predict the correct classification for new insects found). The insects you find may be either katydids or grasshoppers. These insects look quite a bit alike, but there are subtle differences. They are much like ducks and geese: many similarities but some important differences as well.

You have five examples of insects that you know are katydids and five examples that you know are grasshoppers. These 10 insects will comprise our known data set. The unknown is thought to be either a katydid or a grasshopper. Could we



Courtesy of Eamonn Keogh.

use this known data set to come up with a set of rules that would allow us to classify any unknown insect as either a katydid or a grasshopper? By examining how this might be done by hand through trial and error, we can begin to understand one general process that classification data mining algorithms use.

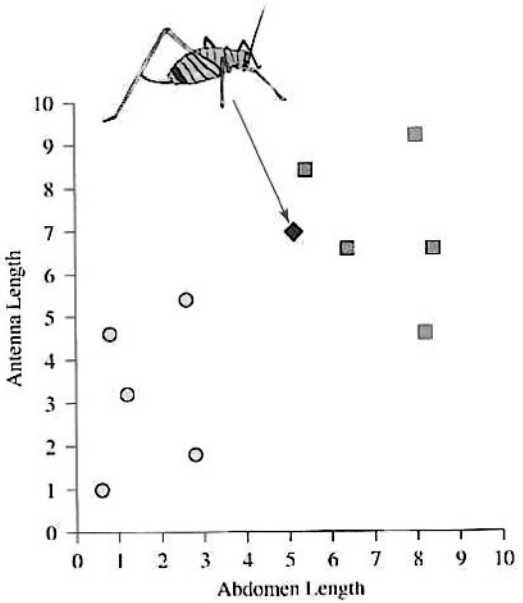


There are many characteristics we could use to aid in our classification. Some of them would include abdomen length, thorax length, leg length, antenna length, the presence of wings, and so on. The 10 insects we have in our known database have the following values for the attributes titled abdomen length and antenna length.

Insect ID	Abdomen Length (mm)	Antenna Length (mm)	Insect Class
1	2.7	5.5	Grasshopper
2	8.0	9.1	Katydid
3	0.9	4.7	Grasshopper
4	1.1	3.1	Grasshopper
5	5.4	8.5	Katydid
6	2.9	1.9	Grasshopper
7	6.1	6.6	Katydid
8	0.5	1.0	Grasshopper
9	8.3	6.6	Katydid
10	8.1	4.7	Katydid
Unknown	5.1	7.0	?

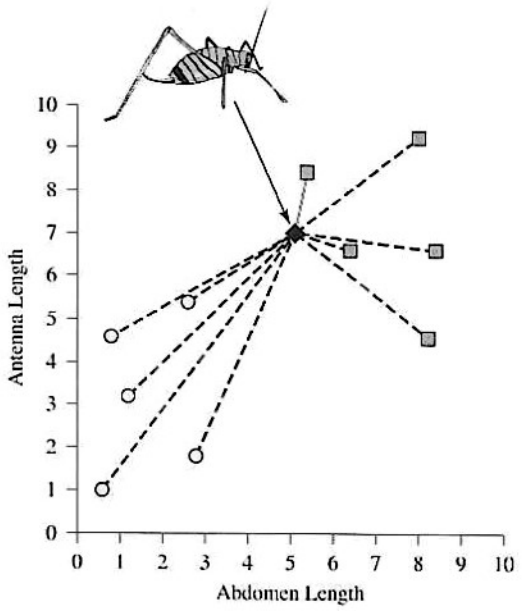
The unknown insect is represented by the last row in the table. We have only included two insect attributes in our table for demonstration purposes because this makes the classification method easy to replicate graphically. As we have seen in

discussing business forecasting techniques, it is usually a good idea to first graph the data in order to look for obvious relationships. We can do the same here by arbitrarily placing abdomen length on one axis and antenna length on the other, thus creating a scatterplot of the data.



The resulting plot is quite informative; the katydids (shown as squares) cluster in the upper right-hand corner of our plot, while the grasshoppers (shown as circles) cluster in the lower left-hand corner of the plot. It is important to note that neither characteristic by itself would do well in making a perfect classification; the combination of the two attributes, however, might more accurately define unknown insects. This unknown insect appears to fall closest to the katydids. But can we come up with a mechanistic (i.e., a rules-based algorithm) way of classifying the unknown as a katydid rather than as a grasshopper? One method would be to look at the geographical neighbors of the unknown insect. Which neighbors are the closest to the unknown? We could describe this process by drawing distance lines between the unknown insect and its neighbors.

If the distance to the unknown insect is closest to the katydids (as measured by summing the distance to katydid neighbors and comparing this to the sum of the distances to grasshopper neighbors), then the unknown is likely a katydid. In essence, the k-Nearest-Neighbor model of data mining works in a similar manner. In actual practice, it is not necessary to calculate the distance to every neighbor; only a small subset of the neighbors is used. The "k" in k-Nearest-Neighbor refers to the number of nearest neighbors used in determining a category correctly.



When using k-Nearest-Neighbor, we use a subset of the total data we have available (called the *training data* set) to attempt to identify observations in the training data set that are similar to the unknown. Scoring (or predicting) new unknowns is assigning the unknowns to the same class as their nearest neighbors. While Euclidian distance is shown in the diagrams here, there are other metrics possible that can be used to define neighbors, and they are at times used in the various commercial data mining packages.

What we are truly interested in is classifying future unknown insects, not the past classification performance on known data. We already know the classifications of the insects in the training data set; that's why we call it a training data set. It trains the model to correctly classify the unknowns by selecting closeness to the k-nearest-neighbors. So, the error rate on known data will not be very helpful in determining if we have a good classification model. An error rate on a training set is not the best indicator of future performance.

The real test of a business forecast was the "out-of-sample" test; the real test of a data mining algorithm will be the test statistics on the validation data.

To predict how well this model might do in the real world at classifying unknowns, we need to use it to classify some data that the model has not previously had access to (the unseen data); we need to use data that was not part of the training data set. This separate data set is called the *validation data* (sometimes called the testing data). In one sense, this separation of data into a training data set and a validation data set is much like the difference between "in-sample" test statistics and "out-of-sample" test statistics in standard forecasting. The real test of a business forecast was the "out-of-sample" test; the real test of a data mining algorithm will be the test statistics on the validation data, not the statistics calculated from the training data.

In order to produce reliable measures of the effectiveness of a data mining tool, researchers *partition* a data set before building a data mining model. It is standard practice to divide the data set into partitions using some random procedure. We could, for instance, assign each instance in our data set a number and then randomly partition the data set into two parts called the training data and the validation data. If there is a great deal of data (unlike the simple example of the katydids and grasshoppers), there is little trouble in using, for example, 60 percent of the records as a training set and the remaining 40 percent as a validation data set. This will ensure that no effectiveness statistics are drawn from the data used to create the model. Thus, an early step in any real data mining procedure is to partition the data.

A BUSINESS DATA MINING CLASSIFICATION EXAMPLE: k-NEAREST-NEIGHBOR (kNN)

What would such a model look like in a business situation? We now turn again to examining a data set used by Shmueli, Patel, and Bruce.[9] The Universal Bank data is also included as an example data set with the Frontline Systems Inc. Solver software. This data set represents information on the customers a bank has in its data warehouse. These individuals have been customers of the bank at some time in the past; perhaps many remain current customers in one dimension or another. The type of information the bank has on each of these 5,000 customers is represented in Tables 9.1 and 9.2.

TABLE 9.1
Universal Bank (Fictitious) Data
The bank has data on a customer-by-customer basis for 5,000 customers in these categories.

Variable Name	Explanation
Age	Customer's age in completed years
Experience	No. of years of professional experience
Income	Annual income of the customer ($000)
ZIP code	Home address, ZIP code
Family	Family size of the customer
CC Avg.	Average spending on credit cards per month ($000)
Education	Education level (1) Undergrad; (2) Graduate; (3) Advanced/Professional
Mortgage	Value of house mortgage if any ($000)
Personal loan	Did this customer accept the personal loan offered in the last campaign?
Securities account	Does the customer have a securities account with the bank?
CD account	Does the customer have a certificate of deposit (CD) account with the bank?
Online	Does the customer use Internet banking facilities?
Credit card	Does the customer use a credit card issued by Universal Bank?

[9] Galit Shmueli, Nitin Patel, and Peter Bruce, *Data Mining for Business Intelligence*. New York: John Wiley & Sons, 2007.

TABLE 9.2 Universal Bank Customer Profiles
The data includes both continuous variables such as income as well as dummy variables such as personal loan. (C9T2)

ID	Age	Experience	Income	ZIP Code	Family	CCAvg	Education	Mortgage	Personal Loan	Securities Account	CD Account	Online	CreditCard
1	25	1	49	91107	4	1.60	1	0	0	1	0	0	0
2	45	19	34	90089	3	1.50	1	0	0	1	0	0	0
3	39	15	11	94720	1	1.00	1	0	0	0	0	0	0
4	35	9	100	94112	1	2.70	2	0	0	0	0	0	0
5	35	8	45	91330	4	1.00	2	0	0	0	0	0	1
6	37	13	29	92121	4	0.40	2	155	0	0	0	1	0
7	53	27	72	91711	2	1.50	2	0	0	0	0	1	0
8	50	24	22	93943	1	0.30	3	0	0	0	0	0	1
9	35	10	81	90089	3	0.60	2	104	0	0	0	1	0
10	34	9	180	93023	1	8.90	3	0	1	0	0	0	0
11	65	39	105	94710	4	2.40	3	0	0	0	0	0	0
12	29	5	45	90277	3	0.10	2	0	0	0	0	1	0
13	48	23	114	93106	2	3.80	3	0	0	1	0	0	0
14	59	32	40	94920	4	2.50	2	0	0	0	0	1	0
15	67	41	112	91741	1	2.00	1	0	0	1	0	0	0
16	60	30	22	95054	1	1.50	3	0	0	0	0	1	1
17	38	14	130	95010	4	4.70	3	134	1	0	0	0	0
18	42	18	81	94305	4	2.40	1	0	0	0	0	0	0

Universal Bank would like to know which customers are likely to accept a personal loan. The bank is considering a promotion to customers that will offer personal loans. What attributes would predict which specific customers would be likely to respond positively to such an offer? If the bank were to consider expending advertising efforts to contact customers who would be likely to consider a personal loan, which customers should the bank contact first? By answering this question correctly, the bank will be able to optimize its advertising effort by directing its attention to the highest-yield customers.

This is an iconic classification problem not unlike the situation of deciding in what class to place an unknown insect. The two classes in this example would be: (1) those with a high probability of accepting a personal loan (*acceptors*), and (2) those with a low probability of accepting a personal loan (*nonacceptors*). We will be unable to classify customers with certainty about whether they will accept a personal loan, but we may be able to classify the customers better than a naive model into one of these two mutually exclusive categories if we estimate a kNN model. Naive model here means to always assign new customers to the most prevalent category in the training data (in this case that would be nonacceptor).

Figure 9.1 shows that there were only 480 acceptors among the 5,000 records in the full data set. Less than 10 percent of the bank's customers had accepted a personal loan. The naive model would then be to classify every new customer as a nonacceptor, and the naive model would be correct more than 90 percent of the time. The goal of the data scientist here would be to make better classifications than the naive model.

The researcher would begin by first partitioning the Universal Bank data. Recall that partitioning the data set is the first step in any data mining technique. Since each row, or record, is a different customer, we could assign a number to each row and use a random selection process to choose 60 percent of the data as

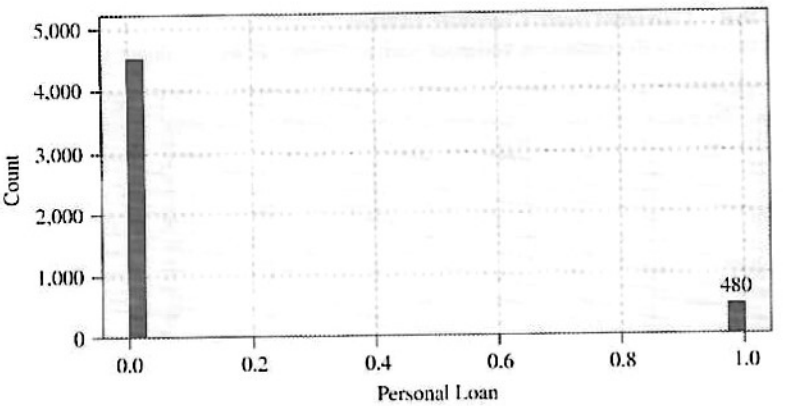


FIGURE 9.1
Universal Bank Personal Loan Variable Distribution ("0" Indicates a Nonacceptor; "1" Indicates an Acceptor) (C9T2)

a training set. The remaining 40 percent of the data set would be the validation partition. All data mining software has such an option available. Once the data is selected into a training set, it would look that in Table 9.3. This partial rendition of the table is produced using the XLMiner® software.

Examining the Partition Summary at the top left of Table 9.3, you will note that there were 5,000 customers in the original data set that have now been divided into a training data set of 3,000 customers and a validation data set of 2,000 customers.

When we instruct the software to perform a k-Nearest-Neighbor analysis of the training data, the real data mining analysis takes place. Just as in the insect classification example, the software will compare each customer's personal loan experience with the selected attributes. This example is, of course, much more multidimensional since we have many attributes for each customer (as opposed to only the two attributes we used in the insect example). For this analysis, we have used all the attributes available in the data set except ID and zip code (zip code is left out arbitrarily in this example; it could be included). The program will compute the distance associated with each attribute. For attributes that are measured as continuous variables, the software will normalize the distance and then measure it

TABLE 9.3 Training Data (Only the First Few Records Shown)
This is a subset of the complete data set. (C9T2)

Partition Summary

	3000
	2000

Partitioned Data

	3	39	15	11	1	1	1	0	0	0	0	0	0
	7	53	27	72	2	1.5	2	0	0	0	0	1	0
	8	50	24	22	1	0.3	3	0	0	0	0	0	1
	9	35	10	81	3	0.6	2	104	0	0	0	1	0
	10	34	9	180	1	8.9	3	0	1	0	0	0	0
	13	48	23	114	2	3.8	3	0	0	1	0	0	0
	14	59	32	40	4	2.5	2	0	0	0	0	1	0
	16	60	30	22	1	1.5	3	0	0	0	0	1	1

Source: Frontline Systems Inc.

TABLE 9.4 Validation Confusion (or Classification) Matrix for the Universal Bank Data The number of nearest neighbors chosen by the XLMiner® software is 4 (not shown in this table). (C9T2)

Validation: Classification Summary

Confusion Matrix		
Actual\Predicted	0	1
0	1794	18
1	52	136

Source: Frontline Systems Inc.

(because different continuous attributes are measured in different scales). For the dummy type or categorical attributes, most programs use a weighting mechanism that is beyond the scope of this treatment. XLMiner® allows the user to normalize distances by using the "rescale" procedure.

The diagnostic statistics for the estimated model will tell if we have possibly found a useful classification scheme.

The diagnostic statistics for the estimated model will tell if we have possibly found a useful classification scheme. In this instance, we want to find a way to classify customers as likely to accept a personal loan. How accurately can we do that by considering the range of customer attributes in our data? Are there some attributes that could lead us to classify some customers as much more likely to accept a loan and other customers as quite unlikely? While the accuracy measures are often produced by the software for both the training data set and the validation data set, our emphasis should clearly be on those measures pertaining to the validation data. There are two standard accuracy measures we will examine: the *classification matrix* (also called the *confusion matrix*) and the *lift chart*. The validation classification matrix for the Universal Bank data training data is shown in Table 9.4.

When our task is classification, accuracy is often measured in terms of error rate, the percentage of records we have classified incorrectly (the converse, of course, would be the percentage of records correctly classified). The error rate is often displayed for both the training data set and the validation data set in separate tables. Table 9.4 is the confusion matrix for the validation data set in the Universal Bank case. The misclassification rate is the number misclassified ($52 + 18 = 70$) divided by the total records ($52 + 18 + 1794 + 136 = 2{,}000$). This gives a misclassification in this instance of 3.5% ($70/2{,}000 = 0.035$).

The table is correctly called either a *confusion matrix* or a *classification matrix*. In Table 9.4, there were 136 records that were correctly classified as "class 1" (i.e., probable personal loan candidates). They were correctly classified because these records represented individuals that did indeed take out a personal loan. However, 18 records were classified as class 1 incorrectly; these were individuals that the model expected to take out a personal loan when, in fact, they did not historically do so. In addition, the table shows 1,794 records predicted to be class 0 (i.e., not probable loan candidates). These records were classified correctly since

TABLE 9.5
Validation Classification Matrix (Confusion Matrix) for the Universal Bank Data
The number of nearest neighbors selected was 4 (not shown in this table). (C9T2)

Source: Frontline Systems Inc.

Validation: Classification Summary

Confusion Matrix

Actual\Predicted	0	1
0	1794	18
1	52	136

Error Report

Class	# Cases	# Errors	% Error
0	1812	18	0.993377483
1	188	52	27.65957447
Overall	2000	70	3.5

historically these individuals did not take out personal loans. Finally, 52 records were incorrectly classified as class 0 when they actually were loan acceptors. The table can then be used to compute a *misclassification rate*. This calculation simply shows the percentage of the records that the model has placed in the incorrect category. In this case, we have 2,000 records in the validation data set, and we have correctly classified 1,930 of them (1,794 + 136). But we have incorrectly classified 18 records as class 1 when they were actually in class 0. We have also incorrectly classified 52 records as class 0 when they were actually in class 1. Thus, we have incorrectly classified 70 records (18 + 52). The misclassification rate is the total number of misclassifications divided by the total records classified (and is usually reported as a percentage). Most packages show the calculation and report it.

In Table 9.5, the misclassification rate is shown in the lower right-hand corner as 3.5 percent (calculated as 70/2,000 and expressed as a percentage). It should be noted that there are two ways in which the error occurred in our example, and although some errors may be worse than others, the misclassification rate groups these two types of errors together.

While this may not be an ideal reporting mechanism, it is commonly used and displayed in the same manner by all data mining software. Some software programs allow placing different costs on the various types of errors as a way of differentiating their impacts to the firm. While the overall error rate of 3.5 percent in the validation data is low in this example, the error of classifying an actual loan acceptor incorrectly as a nonacceptor (52 cases) is much greater than that of incorrectly classifying an actual nonacceptor as a loan acceptor (only 18 cases).

Notice that in both Tables 9.4 and 9.5, the summary report is for the k=4 case, meaning that we have used four neighbors (***not*** four attributes) to classify the records. The number 4 for the k value is chosen by the algorithm. The algorithm has taken a "vote" of the four nearest neighbors in order to classify each record as either a loan acceptor or a nonacceptor. The algorithm actually varied the number of nearest neighbors used from a small number to a large number and selected and reported the best value of k to use. Usually the researcher may specify the range of k values over which the program searches, and the program will respond

TABLE 9.6
Search Log for the Universal Bank Data
The best number of nearest neighbors has been chosen to be 4 because this provides the lowest validation misclassification rate. (C9T2)

Source: Frontline Systems Inc.

Search Log

K	Training: % Incorrect	Validation: % Incorrect
1	0	4.1
2	1.833333333	4.6
3	2.5	3.65
Best: 4	2.166666667	3.5
5	3.166666667	4.3
6	2.8	3.85
7	3.266666667	4.15
8	2.933333333	4
9	3.8	4.2
10	3.466666667	4.15
11	4.1	4.6
12	3.666666667	4.55

Note: Scoring will be done using K=4

by choosing the number of neighbors that optimizes the results (in this situation, XLMiner® minimized the validation misclassification error rate).

In Table 9.6, the XLMiner® program provides an easy way to visualize how the number of nearest neighbors has been chosen. The validation misclassification error rate of 3.5 percent is lowest for four neighbors.

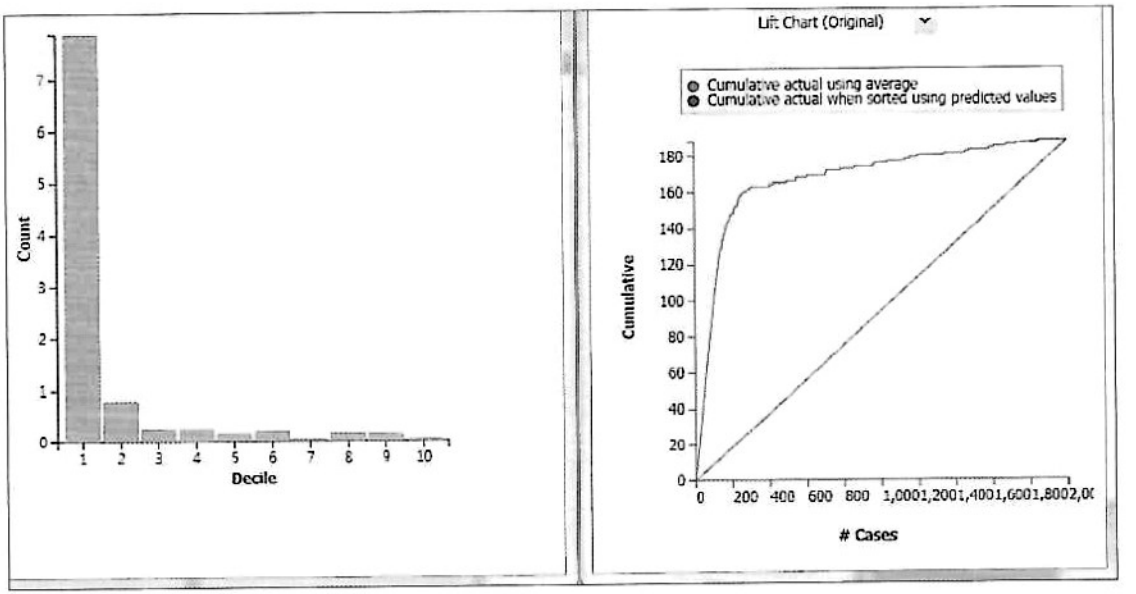
A second way of examining the predictive capability and usefulness of a data mining model can be demonstrated with our Universal Bank example. All data mining software will display a lift chart for any calculated solution; the one for the Universal Bank k-Nearest-Neighbor model is displayed two ways in Figure 9.2.

Lift charts are the most common way (and perhaps the most important way) to compare different classification models.

Lift charts are the most common way (and perhaps the most important way) to compare different classification models. *Lift* is actually a ratio. Lift measures the change in concentration of a particular class when the model is used to select a group from a portion of the general population. Recall from Chapter 8 that lift charts use only reordered data for display; that is, the original order of the data has been replaced by ordering the data from those individuals most likely to accept a personal loan to those least likely to accept a personal loan. Recall that understanding that lift charts are drawn with "reordered" data is key to interpreting correctly what they mean. The most likely acceptors then are represented on the left in each of the charts and the least likely acceptors are represented towards the right-hand side of the chart.

Consider why Universal Bank is attempting to classify the records in its database into *acceptors* and *nonacceptors*. Perhaps Universal Bank is considering a direct solicitation to individuals in the database in order to obtain new personal loan applications. Based on previous experience, the percentage of individuals who respond favorably and take out a personal loan is slightly less than 10 percent

FIGURE 9.2 Decile-Wise Lift Chart and Cumulative Gains Lift Chart for the Universal Bank Validation Data Set (C9T2)



Source: Frontline Systems Inc.

(480 out of 5,000 persons in the data set took out a personal loan). But what if the bank could identify (i.e., predict), before sending a personal loan offer solicitation, the most likely *acceptors*? And what if the number of these likely *acceptors* was quite small relative to the size of the entire database? If the bank could successfully classify the database and identify these likely acceptors, then it might pay for the bank to restrict the solicitation to only those most "likely to respond" individuals. Preparation and delivery costs would be saved, and the bank would receive a *lift* in the percentage of recipients actually accepting a loan. What we may be able to help the bank do is to offer only to those customers with a high probability of loan acceptance, as opposed to offering to everyone in the database. Remember, over 90 percent of the people represented in the database are not likely loan acceptors. Only a relatively small number of the records in the database represent acceptors.

The lift curve is drawn from information about what the k-Nearest-Neighbor model predicted in each case and what actually took place. The lift chart shown on the right-hand side in Figure 9.2 is sometimes called a cumulative gains chart. It is constructed with the records arranged on the x-axis *left to right from the highest probability to the lowest probability of accepting a loan.* The y-axis reports the number of true positives at every point (i.e., the y-axis counts the number of records that represent loan acceptors).

Looking at the decile-wise lift chart on the left-hand side in Figure 9.2, we can see that if we were to choose the top 10 percent of the records classified by our model (i.e., the 10 percent that the algorithm predicts are most likely to accept a personal loan), our selection would include more than seven times as many correct classifications than if we were to select a random 10 percent of records from the database. That's a dramatic lift provided by the model when compared to a random selection.

The same information is displayed in a different manner in the lift chart on the right-hand side of Figure 9.2. This lift (or cumulative gains) chart represents the cumulative records correctly classified (on the *y*-axis), with *the records arranged in descending probability order* on the *x*-axis. Since the curve inclines steeply upward over the first few hundred cases displayed on the *x*-axis, the model appears to provide significant lift relative to a random or naïve selection of records (the naive model selection is depicted by the 45 degree line). Generally, a better model will display higher lift than other candidate models. Lift can be used to compare the performance of different kinds of algorithms (e.g., the k-Nearest-Neighbor algorithm compared with other classification algorithms) and is a good tool for relating the performance of two or more data mining algorithms using the same or comparable data. Notice carefully the straight line rising at a 45-degree angle in the lift chart in Figure 9.2: this could be called a reference line. The line represents how well you might do by classifying as a result of random selection (called a naive model in this context). If the calculated lift line is significantly above this reference line at any point, you may expect the model to outperform a random selection. In the Universal Bank case, the k-Nearest-Neighbor model outperforms a random selection by a very large margin.

CLASSIFICATION TREES: A SECOND CLASSIFICATION TECHNIQUE

Our second data mining technique is variously called a classification tree, a decision tree, or a regression tree. As the name implies, it is, like k-Nearest Neighbor, a way of classifying or dividing up a large number of records into successively smaller sets in which the members become similar to one another. Data miners commonly use a tree metaphor to explain (and to display results from) this technique. Because the term *regression* is most often used to forecast a numeric quantity, when this classification technique is predicting numeric quantities, it is called a *regression tree*. When the technique is classifying by category, it is usually called either a *classification tree* or a *decision tree*. For this reason, the general technique is often called a CART model; CART stands for classification and regression tree.

The general technique is often called a CART model; CART stands for regression and classification tree.

As a child you may have played a game called "Animal, Mineral, or Vegetable." The origin of the game's name, some believe, arises from the 15th-century belief that all living organisms were either animal or vegetable, while all inanimate objects were mineral. Thus, the three categories could effectively separate all matter

into three neat classes. In the game, as you may recall, one player picks any object and the other players must try to guess what it is by asking a limited number of yes or no questions. The object is to ask the least number of questions before correctly guessing the item. In a sense, classification trees are like the game: we begin by knowing virtually nothing about the items we are sorting, but we make up rules along the way that allow us to place the records into different bins, with each bin containing like objects. In "Animal, Mineral, or Vegetable," the set of questions you successfully used to determine the object's name would be the set of rules you could again use to correctly guess a class if the same object were to be chosen by another participant. In the same manner, we create a set of rules from our successful classification attempts, and these rules become the solution to the classification problem and allow the prediction of the class of any unknown.

We now return to our insect classification problem.

In Figure 9.3, we ask first if the abdomen length is greater than 7.1. The vertical line drawn at a value of 7.1 is the graphical representation of this question (or rule). Note that when we draw this line, all the known instances to the right of the line are katydids—we have a uniform classification on that side of the line. To the left of the line, however, we have a mix of katydids and grasshoppers in the known instances. Reasonable questions to ask are why did we choose to split on the attribute "abdomen length" and why did we choose a split value of exactly "7.1?" Would a different attribute to split on or a different value for the split have given better results?

A CART algorithm chooses the attribute to split on (and the particular value for the split) by using the concept of "information entropy." Entropy, or information entropy, is actually a mathematical concept hijacked (from Claude

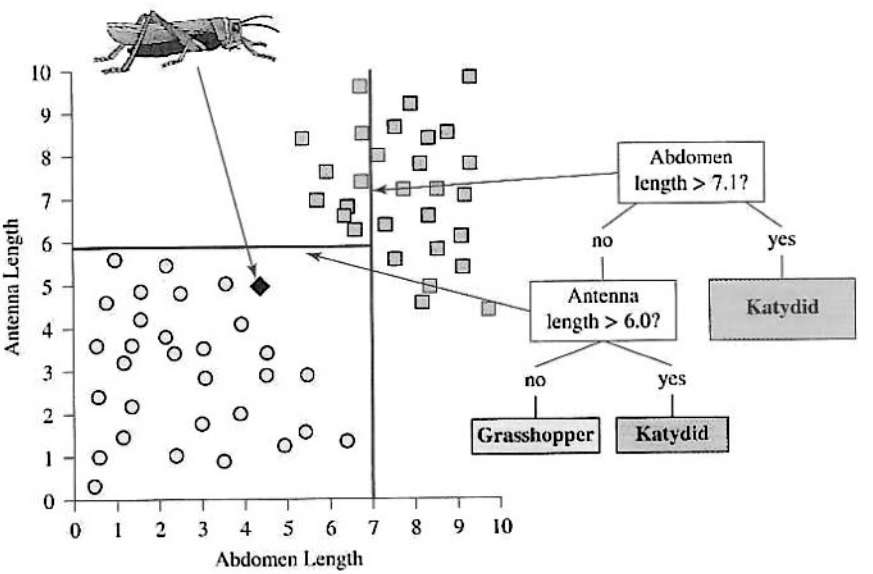


FIGURE 9.3
The Insect Classification Problem Viewed as a Classification Tree Exercise

Courtesy of Eamonn Keogh.

Shannon) by data scientists to aid in the tree-building process. Entropy is a measure of uncertainty associated with a random variable. It is a measure of disorder or, more precisely, unpredictability. The lower the entropy value, the less the uncertainty; the higher the entropy variable, the greater the uncertainty. We are attempting to lower uncertainty, and so the algorithm is directed to look for which particular attribute, and which particular value of that attribute, should be chosen to lower entropy by the greatest value; that attribute and that particular split value are then chosen as the first split. In Figure 9.3, we see the result of the entropy calculation as choosing to split on the attribute abdomen length at a value of 7.1. All records with values greater than 7.1 fall on the right side of the line drawn, and the remainder of records fall to the left. Note that the remaining two "sets" (i.e., all the records to the right or all the records to the left) have more homogeneity (less entropy) than the entire set of records we had at the beginning of the process.

A further question (or split) is necessary to continue the classification. But again we choose the next split attribute and split value by using an entropy calculation. Which attribute and which split value will now offer us the greatest reduction in entropy? The entropy reduction calculation suggests that we next split on "antenna length" and at a value of 6 for this second split. This time, we ask whether the antenna length is greater than 6. The horizontal line drawn at a value of 6 in Figure 9.3 is the graphical representation of this split (or rule). An examination now of the entire set of known instances shows that there is homogeneity in each region (or minimum entropy in each region) defined by our two splits. The right-hand region contains *only* katydids, as does the topmost region in the upper left-hand corner. The bottommost region in the lower left-hand corner, however, contains *only* grasshoppers. Thus, we have divided the geometric attribute space into three regions, each containing only a single class of insect.

In performing two splits to create the three regions, we have also *created* the rules necessary to perform further classifications on unknown insects. Take the unknown insect shown in the diagram with an antenna length of 5 and an abdominal length of 4.5. By asking whether the unknown has an abdominal length of greater than 7.1 (answer no) and then asking whether the antenna length is greater than 6 (answer no), the insect is correctly classified as a grasshopper.

In our example, we have used only two attributes (abdomen length and antenna length) to construct the classification routine so that we could represent the results in Cartesian coordinate two-space. The rule we used to select split attributes and split values was to "reduce entropy." In a real-world situation, however, we need not confine ourselves to only two attributes. In fact, we can use many attributes. The geometric picture might be difficult (or impossible) to draw, but the decision tree (shown on the right-hand side of Figure 9.3) would look much the same as it does in our simple example. In data mining terminology, the two decision points in Figure 9.3 (shown as "abdomen length 7.1" and "antenna length 6") are called *decision nodes*. Nodes in XLMiner® are shown as circles with the decision value shown inside. They are called decision nodes because we classify unknowns by "dropping" them through the tree structure and letting the splitting rules sort them down different branches.

The bottom of our classification tree in Figure 9.3 has three leaves. Each *leaf* is a terminal node in the classification process: it represents the situation in which all the instances that follow that *branch* result in uniformity. The three leaves in Figure 9.3 are represented by the shaded boxes in the diagram. Data mining classification trees are *upside-down* in that the leaves are at the bottom, while the root of the tree is at the top; this is the convention in data mining circles. To begin a *scoring* process, all the instances are at the root (i.e., top) of the tree; these instances are partitioned by the rules we have determined with the known instances. The result is that the unknown instances move downward through the tree until reaching a leaf node, at which point they are (hopefully) successfully classified. In analytics software, the tree is "drawn" using rules from information theory that point the direction toward creating leaves that contain only homogeneous objects or instances. The tree representations of the solution can be drawn in most software (XLMiner® will do this), but that is primarily so the user can interpret and explain the result. It is the set of rules that the tree represents that allows rapid classifications of new unknowns.

Classification trees can become quite large and ungainly, but, more importantly, they will tend to overfit the data.

At times, the classification trees can become quite large and ungainly, but, more importantly, they will tend to overfit the data, much like the ARIMA models. It is common for data mining programs to *prune* the trees to remove branches to prevent this overfitting. An unpruned tree that was constructed using the training data set will sometimes match that data perfectly. Does that mean that this unpruned tree will do the best job in classifying new unknown instances? Probably not.

A good classification tree algorithm will make the best split (at the first decision node) first followed by decision rules that are made up with successively smaller and smaller numbers of training records. These later decision rules will become more and more idiosyncratic. The result may be (and usually will be) an overfit tree that will not do well in classifying new instances. Thus, the need for pruning. Each data mining package uses a proprietary pruning algorithm that usually takes into account for any branch the added drop in the misclassification rate versus the additional tree complexity. XLMiner® and other data mining programs use candidate tree formulations with the validation data set to find the lowest validation data set misclassification rate—that tree is selected as the final best-pruned tree. The actual process is more complicated than we have described here.

The best pruned tree is often used in actual practice.

In Figure 9.4, you can see that the error rate (misclassification rate) decreases on the training data as the number of splits increases, until finally each leaf contains only like items and the error rate is zero. But if we used this tree to classify unseen data (i.e., the validation data), the error rate bottoms out well before the error rate on the training data reaches zero. We use this property of decision trees as a basis for pruning. If we construct a tree with just enough splits to minimize the error rate on the unseen data, the resulting tree is called a "minimum error tree." If we choose the smallest tree within one standard error of the minimum error, the resulting tree is called the "best pruned tree." The best pruned tree is often used in actual practice: it is very similar to using the Akaike or Bayesian Information Criterions because it balances the complexity of the selected model

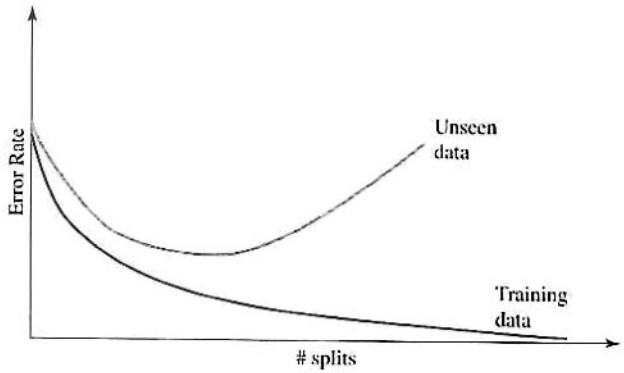


FIGURE 9.4
Pruning Is Often Accomplished by Minimizing the Error Rate on the Unseen Data (Called a Minimum Error Tree) or Choosing a Tree within One Standard Deviation of That Point (Called a Best Pruned Tree)

with its predictiveness. A data scientist who fails to use some form of pruning when employing a CART algorithm will almost certainly overfit the data and select a model that will perform poorly in the field.

Classification trees are very popular in actual practice because the decision rules are easily generated and, more importantly, because the trees themselves are easy to understand and explain to others. There are disadvantages as well, however. The classification trees can suffer from overfitting, and if they are not pruned well, these trees may not result in good classifications of new data (i.e., they will not score new data well). Attributes that are correlated will also cause this technique serious problems. It is somewhat similar to multicollinearity in a regression model. Be careful not to use attributes that are very closely correlated one with another. The more data you have available to build the model, the less chance the correlated attributes will cause problems.

A Business Data Mining Example: Classification Trees

We can once again use the Universal Bank data from Table 9.2 in an attempt to classify customers into likely or unlikely personal loan clients. The first step, as always, would be to partition the data into training and validation data sets; the training data set was displayed in Table 9.3. Note that while the data scientist selects the attributes that are to be used, the CART algorithm selects the decision rules and the order in which they are executed using Shannon's information entropy approach. Table 9.7 displays a portion of the classification tree output from XLMiner® for the Universal Bank data. We have used a number of attributes to help in making up the decision rules; most of the attributes can be seen to intuitively affect whether a person is a likely personal loan candidate. Among the attributes used are:

- Customer's age
- Individual's average spending per month on credit cards
- Value of the individual's house mortgage
- Individual's annual income
- And others (but excluding ID and zip code).

TABLE 9.7 Validation Classification Using the Best Pruned Tree on the Validation Data Set of the Universal Bank Data (C9T2)

Validation: Classification Summary

Confusion Matrix

Actual\Predicted	0	1
0	1783	29
1	16	172

Error Report

Class	# Cases	# Errors	% Error
0	1812	29	1.600441501
1	188	16	8.510638298
Overall	2000	45	2.25

Source: Frontline Systems Inc.

The scoring summary format is identical to the one we saw with the k-Nearest-Neighbor technique. For the classification tree technique, the misclassification rate is just 2.25 percent; this is even lower than the 3.5 percent achieved with the k-Nearest-Neighbor model. A scant 29 individuals were expected to be likely to accept personal loans and yet did not do so.

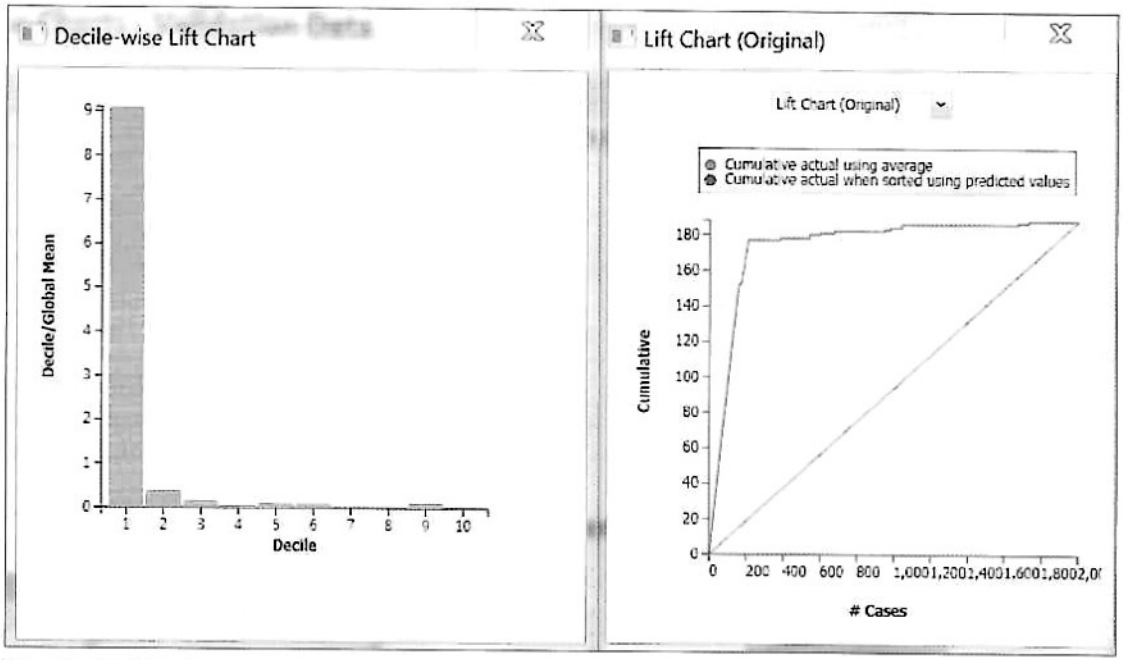
Looking at the decile-wise lift chart on the left-hand side of Figure 9.5, we can see that if we were to choose the top 10 percent of the records classified by our best pruned classification tree model (i.e., the 10 percent most likely to accept a personal loan), our selection would include almost nine times as many correct classifications than if we were to select a random 10 percent of the database. That result is even more striking than the one we obtained with the k-Nearest-Neighbor model.

The lift chart on the right-hand side of Figure 9.5 is a cumulative gains chart. Recall that it is constructed with the records arranged on the x-axis *left to right from the highest probability of accepting a loan to the lowest probability of accepting a loan.* The y-axis reports the number of true positives at every point (i.e., the y-axis counts the number of records that represent actual loan acceptors). The fact that the cumulative personal loan line jumps sharply above the average beginning on the left side of the chart shows that our model does significantly better than choosing likely loan applicants at random. In other words, there is considerable lift associated with this model.

The actual topmost part of the classification tree that was produced by XLMiner® is displayed in Figure 9.6.

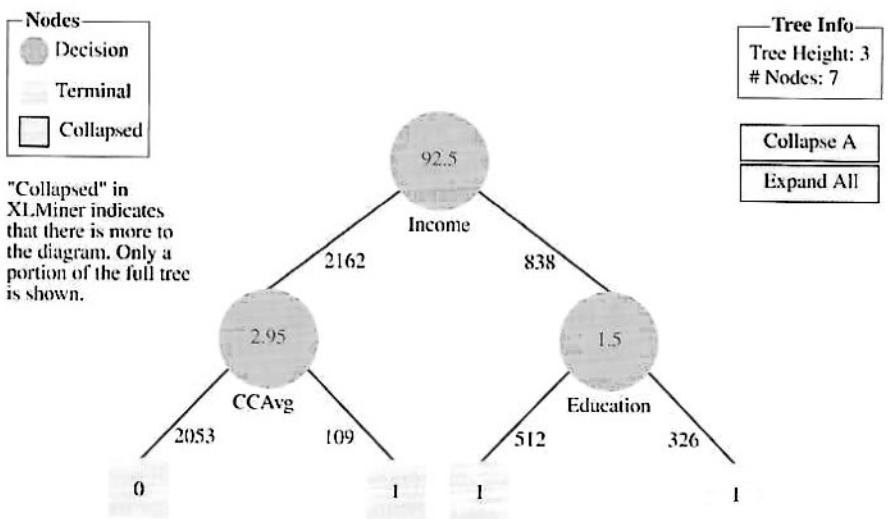
The classification tree first divides on the income variable. Is income greater than 92.5? That results in 2,162 of the records being sorted to the "less than 92.5" and 838 records sorted to the "greater than 92.5" side. XLMiner® then sorted on the basis of educational level on the "greater than" side and credit

FIGURE 9.5 **Decile-Wise Lift Chart and Lift Chart (Cumulative Gains Chart) Using the Best Pruned Tree on the Validation Data Set of the Universal Bank** (C9T2)



Source: Frontline Systems Inc.

FIGURE 9.6 **A Portion of the Classification Tree Using the Best Pruned Tree on the Validation Data Set of the Universal Bank** (C9T2)



Source: Frontline Systems Inc.

TABLE 9.8 A Portion of the Tree Rules Using the Best Pruned Tree on the Validation Data Set of the Universal Bank (C9T2)

Fully Grown Tree Rules (Using Training Data)

Node ID	Parent ID	Left Child ID	Right Child ID	Split Var	Split Value/Set	Training Cases
1	N/A	2	3	Income	92.5	3000
2	1	4	5	CCAvg	2.95	2162
3	1	6	7	Education	1.5	838
4	2	N/A	N/A	N/A	N/A	2053
5	2	N/A	N/A	N/A	N/A	109
6	3	N/A	N/A	N/A	N/A	512
7	3	N/A	N/A	N/A	N/A	326

Source: Frontline Systems Inc

card average balance on the "less than" side and so on. While examining the partially drawn tree in Figure 9.6 is useful, it may be more instructive to examine the rules that are exemplified by the tree. Some of those rules are displayed in Table 9.8.

The rules displayed in Table 9.8 represent the same information shown in the tree diagram in Figure 9.6. Examining the first row of the table shows the split value as 92.5 for the split variable of income. It is called a decision node because there are two branches extending downward from this node (i.e., it is not a terminal node or leaf). The split on credit card average balance uses a split value of 2.95, while on education a split value of 1.5 is used. Row 2 shows that 2,162 cases are classified as going down the left branch, while 838 records travel down the right branch. It is the rules displayed in this table that the program uses to score new data, and they provide a concise and exact way to score new data in a speedy manner.

If actual values are predicted (as opposed to categories) for each case, then the tree is called a regression tree.

If actual values are predicted (as opposed to categories) for each case, then the tree is called a regression tree. For instance, we could attempt to predict the selling price of a used car by examining a number of attributes of the car. The relevant attributes might include the age of the car, the mileage the car had been driven to date, the original selling price of the car when new, and so on. The prediction would be expected to be an actual number, not simply a category. The process we have described could, however, still be used in this case. The result would then be a set of rules that would determine the predicted price.

A Business Data Mining Example: Regression Trees

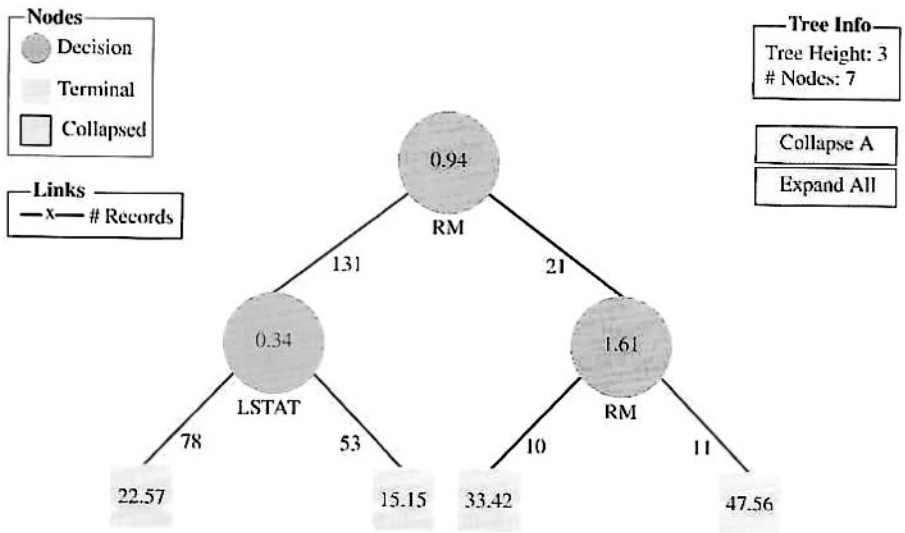
Regression trees (part of the CART family of algorithms) are used to predict actual numeric values rather than the category of a particular record. Consider the case of using Boston housing data to predict the median value of a home (i.e., the target is a variable denominated in thousands of dollars); the prediction will be made on a continuous variable, not a categorical variable as in the Universal Bank case. The attributes used will be items that are thought to affect housing prices in the area. For example, one attribute is the weighted distance to five

Boston employment centers (travel time to work is thought to affect housing prices). Another attribute is pupil-teacher ratio by town, which may be a proxy for school attractiveness, also thought to affect housing prices.

A "Regression Tree" algorithm is chosen for the prediction rather than a "Classification Tree" because the target is a continuous variable, but the algorithm works in much the same manner in constructing a tree that will probably require some pruning in order to prevent the disease of overfitting. The Regression Tree algorithm produces lift charts similar to a Classification Tree, but the confusion matrix will be replaced by the following diagnostic statistics: the root mean square error (RMSE as described on p. 30) and an R-squared calculation.

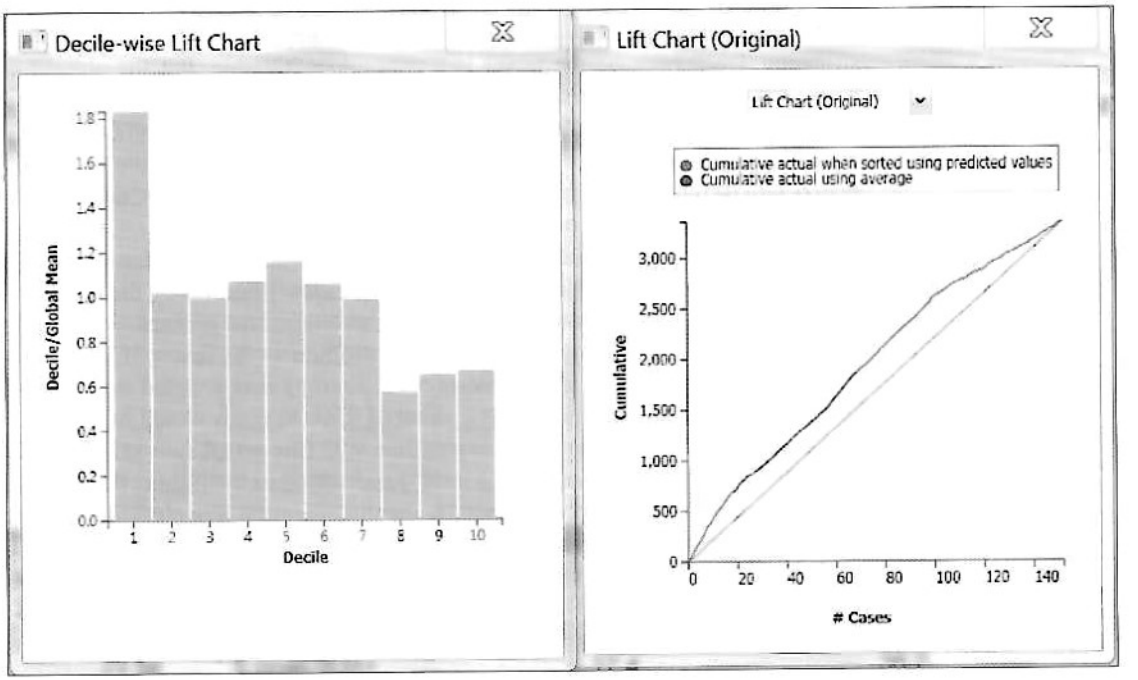
Consider Figure 9.7, which shows the upper portion of the regression tree as drawn by XLMiner®. The first split chosen by the algorithm was on the LSAT attribute. "LSAT" is the "% lower status of the population;" the greatest reduction in entropy was apparently achieved by using this attribute for the first split. Figure 9.7 shows that the split had a split value of 9.73. Entropy was reduced the most by splitting on the RM attribute and at a value of 9.73. Records would be sorted to the left if the RM value of a record is less than 9.73 (the actual split value is truncated in the diagrams drawn by XLMiner®) and sorted to the right if the value is greater. The next two splits suggested by the classification tree algorithm are on "RM" (number of rooms per dwelling) and on "CRIM" (per capita crime rate). The remainder of the tree is not shown in Figure 9.7, which is confined to displaying only the first three levels of the tree.

FIGURE 9.7 **The Regression Tree (Only a Portion of the Full Tree) for the Boston Housing Data** (C9F7)



Source: Frontline Systems Inc.

FIGURE 9.8 The Decile-Wise Lift Chart and The Cumulative Gains Chart for the Boston Housing Data (C9F7)



Source: Frontline Systems Inc.

We can, however, judge the appropriateness of the tree by examining the diagnostic statistics.

Figure 9.8 displays both the decile-wise lift chart as well as the cumulative gains chart for the best pruned regression tree on the Boston housing data. Both charts tell the story that there is some lift associated with the model. XLMiner® also makes available for regression tree estimates the RMSE and the R-squared calculations.

NAIVE BAYES: A THIRD CLASSIFICATION TECHNIQUE

A third and somewhat different approach to classification uses statistical classifiers. This technique will also predict the probability that an instance is a member of a certain class. This technique is based on Bayes' theorem; we will describe the theorem below. In actual practice, these Naive Bayes algorithms have been found to be comparable in performance to the decision trees we have examined. One hallmark of the Naive Bayes model is speed, along with its high accuracy. Bayesian analytics techniques have been used successfully in many real-world situations. The Google self-driving car was based upon a Bayesian model predicting

the state space of an unknown location. E-mail spam filters used at many universities and businesses are often based upon a Bayesian classification algorithm. The World War II codebreaking at Bletchley Park depicted in the movie *Enigma* used a technique titled Banburismus, which is a highly intensive Bayesian technique that allowed Alan Turing and his colleagues to guess a stretch of letters in a German Enigma-encoded message and measure their belief in the accuracy of these guesses (or classifications).

World War II codebreaking at Bletchley Park depicted in the movie *Enigma* used a technique titled Banburismus, which is a highly intensive Bayesian technique.

This model is called *naive* because it assumes (perhaps naively) that each of the attributes is independent of the values of the other attributes. Of course, this will never be strictly true, but in actual practice, the assumption (although somewhat incorrect) allows the rapid determination of a classification scheme and the accuracy does not seem to suffer appreciably when such an assumption is made.

To explain the basic procedure, we return to Eamonn Keogh's insect classification example. Our diagram may be of the same data we have used before, but we will examine it in a slightly different manner.

The known instances of katydids and grasshoppers are again shown in Figure 9.9, but only a single attribute of interest is labeled on the *y*-axis: antenna length. On the right-hand side of Figure 9.9, we have drawn a histogram of the antenna lengths for grasshoppers and a separate histogram representing the antenna lengths of katydids.

Now assume we wish to use this information about a single attribute to classify an unknown insect. Our unknown insect has a measured antenna length of 3 (as shown in Figure 9.10). Look on the problem as an entirely statistical problem. Is this unknown more likely to be in the katydid distribution or the grasshopper distribution? A length of 3 would be in the far-right tail of the katydid distribution (and therefore unlikely to be a part of that distribution). But a length of 3

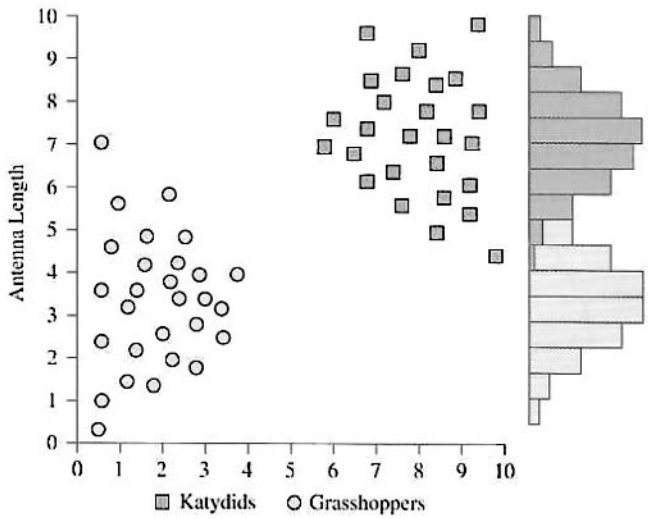


FIGURE 9.9
Insect Example
This has only a single attribute displayed: antenna length. Abdomen length is still measured on the *x*-axis.

Courtesy of Eamonn Keogh.

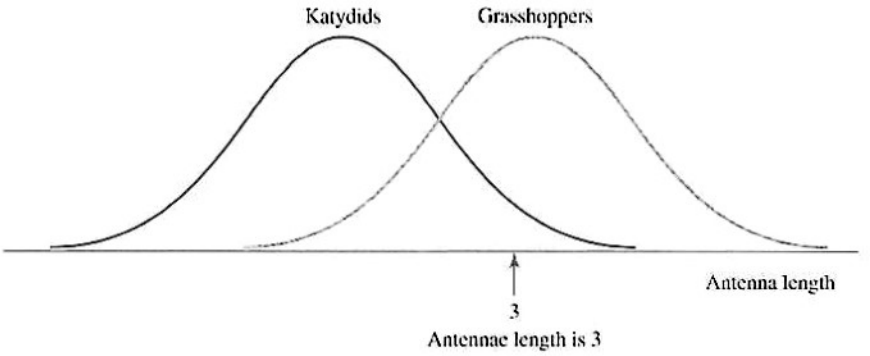


FIGURE 9.10
Histograms Representing Antenna Lengths
Katydids are on the left and grasshoppers on the right.

Courtesy of Eamonn Keogh.

is squarely in the center of the grasshopper distribution (and therefore it is more likely to be a member of that distribution). Of course, there is the possibility that the unknown with an antenna length of 3 is actually part of the katydid distribution (and therefore is actually a katydid), but that probability is small, as evidenced by a length of 3 being in the small tail of the distribution. It is far more likely that our unknown is part of the grasshopper distribution (and is therefore truly a grasshopper). So far we have used only a single attribute. What if we consider an additional attribute? Would that perhaps help our accuracy in making classifications?

Figure 9.11 represents two attributes (antenna length on the y-axis and abdomen length on the x-axis) for the known katydids and grasshoppers. By using the two attributes together, we effectively create a quadratic boundary between the

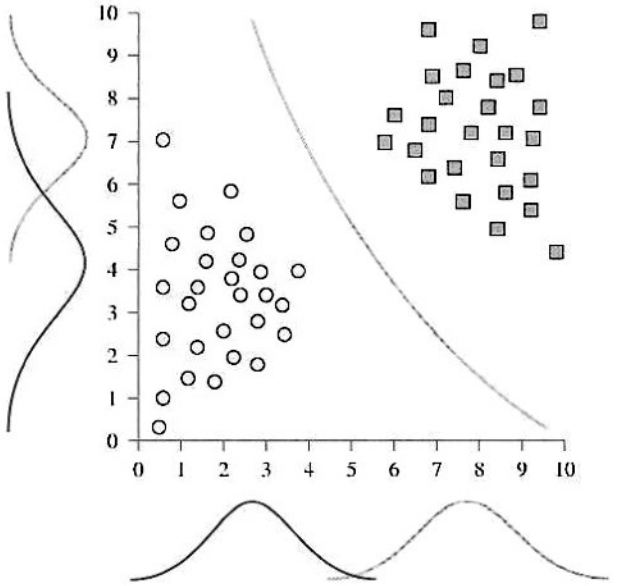


FIGURE 9.11
Two Sets of Histograms
These represent the antenna lengths of katydids on the y-axis, and abdomen lengths on the x-axis.

Courtesy of Eamonn Keogh.

two classes of known insects. An unknown would be classified by its location above or below the boundary. One of the important features of the Naive Bayes model is that it handles irrelevant features quite well. If an irrelevant feature is included in the attributes list, it has little effect on the classifications the model makes (and thus introduces little error).

To examine this technique, we will use actual data from the passenger list of the HMS *Titanic*. On Sunday evening, April 14, 1912, the *Titanic* struck an iceberg. The ship sank a scant two hours and 40 minutes later. We have somewhat complete information on the 2,201 souls on the ship at the time of the accident. We say the information is "somewhat" complete because the data are based on a report made shortly after the event and the White Star line (the owners of the *Titanic*) kept their records in a peculiar manner.[10] For instance, boys are classified by the title "Master," but girls are not clearly distinguished from women. The data are not without some ambiguity, but we can still attempt to ascertain characteristics of the survivors. We are attempting to classify individuals as survivors of the disaster or nonsurvivors (i.e., those who perished). Our data looks like the following:

Age	Sex	Class	Survived
Adult	Male	First	Alive
Adult	Male	First	Alive
Adult	Male	First	Alive
Adult	Male	First	Alive
Adult	Male	First	Alive
Adult	Male	First	Alive
Adult	Male	First	Alive
Adult	Male	First	Alive
Adult	Male	First	Alive

The data set contains information on each of the individuals on the *Titanic*. We know whether they were adult or child, whether they were male or female, the class of their accommodations (first class passenger, second class, third class, or crew), and whether they survived that fateful night. In our list, 711 are listed as alive while 1,490 are listed as dead; thus, only 32 percent of the people on board survived.

What if we wished to examine the probability that an individual with certain characteristics (say, an adult, male crew member) were to survive? Could we use the Naive Bayes method to determine the probability that this person survived? The answer is yes; that is precisely what a Naive Bayes model will do. In this case, we are attempting to classify the adult, male crew member into one of two categories: alive or dead.

The Naive Bayes process begins like our two previous techniques; the data set is divided into a training data set and a validation data set. In Table 9.9, we

[10] The *Titanic* data set is used by permission of Professor Robert J. MacG. Dawson of Saint Mary's University, Halifax, Nova Scotia. See Dawson, "The Unusual Episode, Data Revisited," *Journal of Statistics Education*, vol. 3, no. 3 (1995).

TABLE 9.9 Validation Data Scoring for the Naive Bayes Model of *Titanic* Passengers and Crew (C9T9)

Validation: Classification Summary

Confusion Matrix

Actual\Predicted	Alive	Dead
Alive	130	149
Dead	50	551

Error Report

Class	# Cases	# Errors	% Error
Alive	279	149	53.40501792
Dead	601	50	8.319467554
Overall	880	199	22.61363636

Source: Frontline Systems Inc.

present the validation summary report for the Naive Bayes model as computed in XLMiner®.

The misclassification rate on the validation data set that included 880 souls is computed by XLMiner® as 22.61 percent, but the lift chart and the decile-wise lift chart in Figure 9.12 show that the model does improve on naively selecting a class at random for the result. Note that the naive model would have us predict every individual as a casualty since that is the predominant result. Doing so, we would incorrectly classify the 279 survivors in the validation data set as dead; thus, the naive model validation data set misclassification rate is 31.70 percent (significantly higher than the Naive Bayes misclassification rate).

Bayes' theorem predicts the probability of a prior event (called a posterior probability), given that a certain subsequent event has taken place.

The Naive Bayes model rests on Bayes' theorem. Simply stated, *Bayes' theorem* predicts the probability of a prior event (called a posterior probability), given that a certain subsequent event has taken place. For instance, what is the probability that a credit card transaction is fraudulent, given that the card has been reported lost? Note that the reported loss preceded the current attempted use of the credit card.

The posterior probability is written as $P(A \mid B)$. Thus , $P(A \mid B)$ is the probability that the credit card use is fraudulent, given that we know the card has been reported lost. $P(A)$ would be called the prior probability of A and is the probability that any credit card transaction is fraudulent, regardless of whether the card is reported lost.

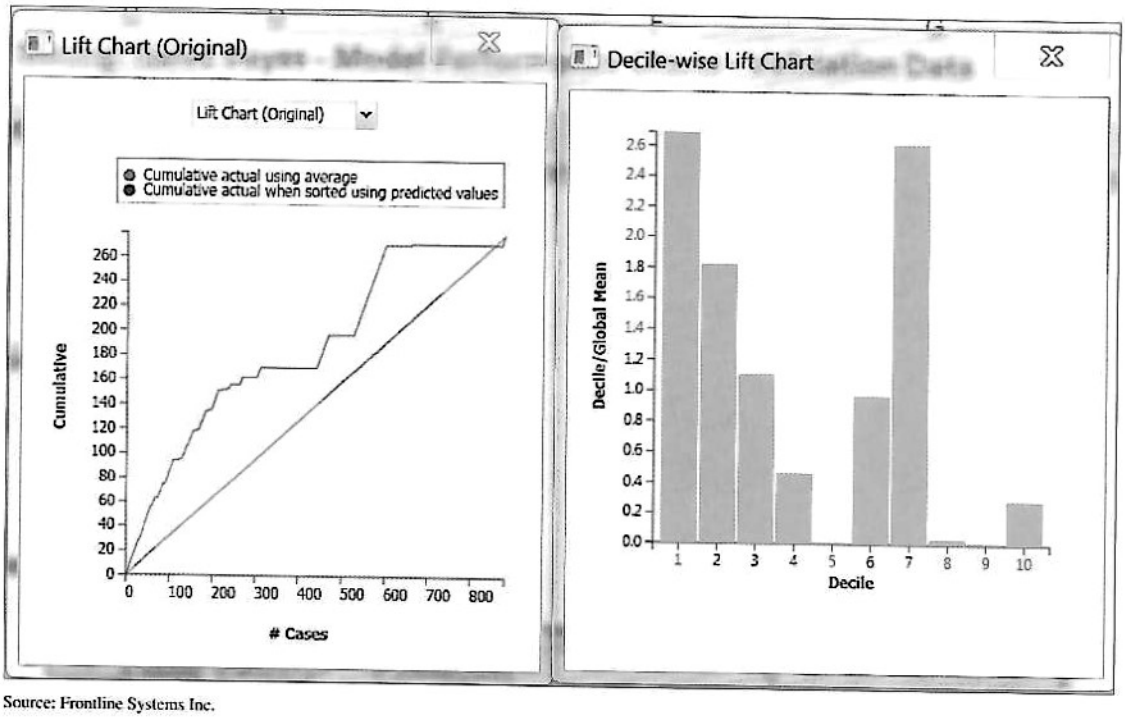
The Bayesian theorem is stated in the following manner:

$$P(A \mid B) = \frac{P(B \mid A)P(A)}{P(B)}$$

where:

$P(A)$ is the prior probability of A. It is *prior* in the sense that it does not take into account any information about B.

FIGURE 9.12 **Cumulative Gains Lift Chart and Decile-Wise Lift Chart for the Naive Bayes *Titanic* Model** (C9T9)



Source: Frontline Systems Inc.

$P(A \mid B)$ is the conditional probability of A, given B. It is also called the *posterior probability* because it is derived from or depends upon the specified value of B. This is the probability we are usually seeking to determine.

$P(B \mid A)$ is the conditional probability of B, given A.

$P(B)$ is the prior probability of B.

An additional example will perhaps make the use of Bayes' theorem clearer. Consider that we have the following data set showing eight credit card transactions. For each transaction, we have information about whether the transaction was fraudulent and whether the card used was previously reported lost (see Table 9.10).

Applying Bayes' theorem:

$$P(Fraud \mid Card\ Reported\ Lost) = \frac{P(Lost \mid Fraud)\ P(Fraud)}{P(Lost)}$$

$$= \frac{\left(\frac{2}{3}\right)\left(\frac{3}{8}\right)}{\frac{3}{8}} = 0.667$$

TABLE 9.10
Credit Card Transaction Data Set

Transaction No.	Fraudulent?	Reported Lost?
1	Yes	Yes
2	No	No
3	No	No
4	No	No
5	Yes	Yes
6	No	No
7	No	Yes
8	Yes	No

and

$$P(NonFraud \mid Card\ Reported\ Lost) = \frac{P(Lost \mid NonFraud)P(NonFraud)}{P(Lost)}$$

$$= \frac{\left(\frac{1}{5}\right)\left(\frac{5}{8}\right)}{\frac{3}{8}} = 0.333$$

Thus, the probability of a fraudulent transaction if the card has been reported lost is 66.7 percent. The probability of a nonfraudulent transaction if the card has been reported lost is 33.3 percent.

Returning to the *Titanic* data and the Naive Bayes model calculated by XLMiner®, we may now demonstrate the calculation of the posterior probabilities of interest. These are the answers to our question concerning the probability that an adult, male crew member would survive the disaster. XLMiner® produces an additional output for the Naive Bayes model displaying the prior class probabilities and the calculated conditional probabilities. These are displayed in Table 9.11.

To answer our question concerning the survival probability of an adult, male crew member we need once again to apply Bayes' theorem.

The statement of Bayes theorem would be:

$$P(alive \mid age = adult,\ sex = male,\ class = crew) = \frac{P(adult,\ male,\ crew \mid alive)P(alive)}{P(adult,\ male,\ crew)}$$

We first need to calculate the conditional probabilities required in the Bayes' theorem:

Conditional probability of "alive" if you were a crew member, male, and adult:

$$\text{P(alive)} = (0.28440367)(0.495391705)(0.912442396)(0.327024981) = 0.042040736$$

TABLE 9.11
Prior Probabilities and Prior Conditional Probabilities Calculated in XLMiner® for the *Titanic* Data (C9T9)

Source: Frontline Systems Inc.

Prior Probability

Class	Probability
Alive	0.327024981
Dead	0.672975019

Prior Conditional Probability: Training

Prior Conditional Probability: Training-Age		
Value/Class	Alive	Dead
Adult	0.912442396	0.970819304
Child	0.087557604	0.029180696

Prior Conditional Probability: Training-Sex		
Value/Class	Alive	Dead
Male	0.495391705	0.912457912
Female	0.504608295	0.087542088

Prior Conditional Probability: Training-Class		
Value/Class	Alive	Dead
First	0.279816514	0.085106383
Second	0.172018349	0.10862262
Third	0.263761468	0.356103024
Crew	0.28440367	0.450167973

Note that we are now multiplying probabilities, assuming they are independent. In like manner, we calculate the conditional "dead" probability:

Conditional probability of "dead" if you were a crew member, male, and adult:

$$P(\text{dead}) = (0.450167973)(0.912457912)(0.970819304)(0.672975019) = 0.268364325$$

To compute the actual (or posterior) probabilities, we divide each of these conditional probabilities by their sum:

Posterior probability of "alive" if you were a crew member, male, and adult:

$$= (0.042040736)/(0.042040736 + 0.268364325) = 0.135438307$$

And

Posterior probability of "dead" if you were a crew member, male, and adult:

$$= (0.268364325)/(0.268364325 + 0.042040736) = 0.864561693$$

There are only two possible outcomes here ("dead" or "alive"), and the posterior probabilities should (and do) sum to 1 (0.135438307 + 0.864561693 = 1). The Bayes theorem calculation in this instance includes a sum in the denominator because the denominator includes all individuals who are adult, male, and crew, whether they are dead or alive (hence, the sum of two probabilities).

Naive Bayes has assumed the attributes have independent distributions. While this is not strictly true, the model seems to work well in situations where the assumption is not grossly violated. Use of larger data sets will all but eliminate the problem of including irrelevant attributes in the model. The effects of these irrelevant attributes are minimized as the data set becomes larger.

We can again use the Universal Bank data and apply the Naive Bayes model in order to predict customers who will accept a personal loan. Figure 9.13 displays the Naive Bayes results from XLMiner® for the Universal Bank data.

Once again, it is clear that the model performs much better than a naive selection of individuals when we try to select possible loan acceptors. Looking at the decile-wise lift chart on the right-hand side of Figure 9.13, we can see that if we were to choose the top 10 percent of the records reordered by our classification tree model (i.e., the 10 percent most likely to accept a personal loan), our selection would include approximately seven times as many correct classifications than if we were to select a random 10 percent of the database and count the number of acceptors included (i.e., the naive model). While Naive Bayes models do extremely well on training data, in real-world applications these models tend not to do quite as well as other classification models in some situations. This is likely due to the disregard of the model for attribute interdependence. In many real-world situations, however, Naive Bayes models do just as well as other classification models. While the Naive Bayes model is relatively simple, it makes sense to try the simplest models first and to use them

FIGURE 9.13 The Naive Bayes Model Applied to the Universal Bank Data
Included are the confusion matrix, misclassification rate, and lift charts for the validation data set (C9F13)

Validation: Classification Summary

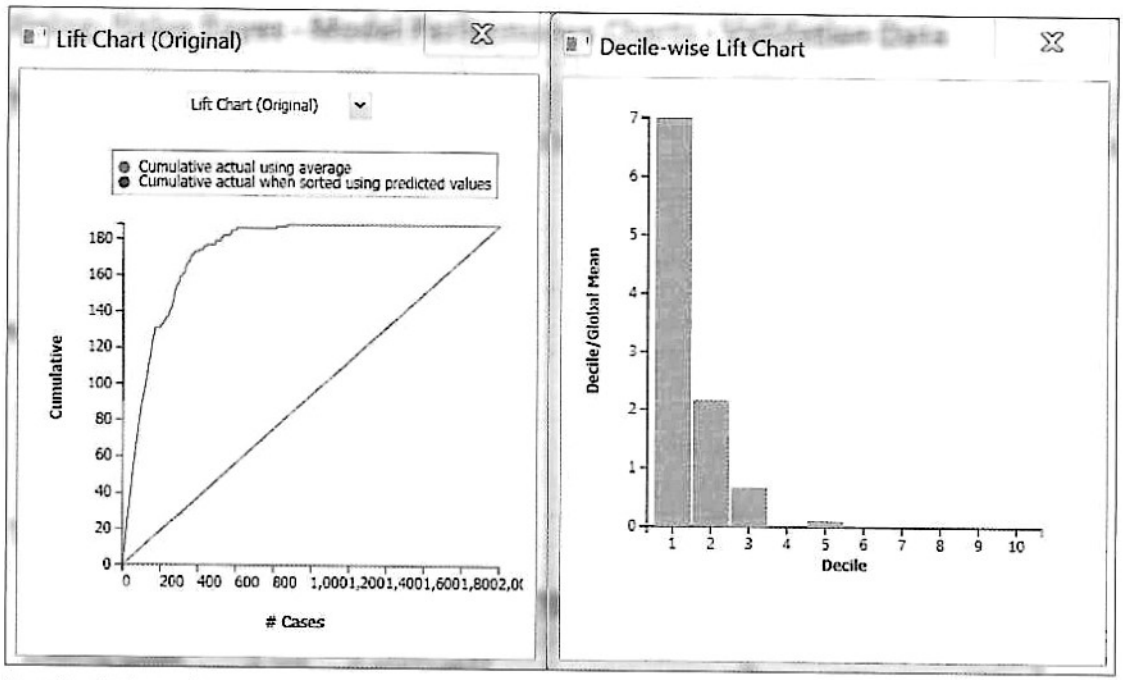
Confusion Matrix

Actual\Predicted	0	1
0	1762	50
1	57	131

Error Report

Class	# Cases	# Errors	% Error
0	1812	50	2.759381898
1	188	57	30.31914894
Overall	2000	107	5.35

FIGURE 9.13 (continued)



Source: Frontline Systems Inc.

if they provide sufficient results. Data sets (especially small data sets) that contain highly interdependent attributes may fare poorly with Naive Bayes.

LOGIT: A FOURTH CLASSIFICATION TECHNIQUE

Economist Daniel McFadden won the Nobel Memorial Prize in 2000 for his development of a discrete choice model, that is, a particular form of logit.

Logistic regression (or logit) is perhaps the most famous of the classification algorithms and is used in a variety of fields. It is the only classification algorithm to be associated with a Nobel Prize. Economist Daniel McFadden won the Nobel Memorial Prize in 2000 for his development of a discrete choice model, that is, a particular form of logit. The logit technique is a natural complement to linear least-squares regression. It has something in common with the ordinary linear regression models we examined in Chapters 4 and 5. Ordinary linear regression provides a universal framework for much of economic analysis; its simplified manner of looking at data has proven useful to researchers and forecasters for decades. But ordinary linear regression assumes a continuous numeric target variable and tries to estimate the functional relationship between the predictors and the target variable.

Logistic regression serves much the same purpose for categorical data. The single most important distinction between logistic regression and ordinary regression is that the dependent variable in logistic regression is categorical (and not continuous numerical). The explanatory variables, or attributes, may be either continuous or

categorical (as they were in linear least-squares models). Just like the ordinary linear regression model, logistic regression is able to use all sorts of extensions and sophisticated variants. Logistic regression has found its way into the toolkits of not only forecasters and economists but also of, for example, toxicologists and epidemiologists.

In situations where the target variable is categorical, ordinary linear regression models become inadequate. Thus, a different approach involving a very different estimating procedure is required. What logit does is to transform the concept of an ordinary linear regression into an equation able to predict the probabilities of the possible outcomes, not just the numeric value of the target variable itself.

The Universal Bank situation we have been examining provides a case in point. The dependent variable, the item we are attempting to forecast, is dichotomous—either a person accepts a loan or rejects the loan. There is no continuous variable here; it is more like an on/off switch. But why are we unable to use linear least-squares models on this data?

Consider Table 9.12: it contains information about 20 students, the hours they spent studying for a qualifying exam, and their results. If they passed the exam, the table shows a 1; if they failed the exam, the table shows a 0.

If we graph this data as a scatterplot (Figure 9.14), we see there are two possible outcomes: pass (shown as 1) and fail (shown as 0).

It appears from the scatterplot that students who spent more time studying for the exam did have a better chance of passing. We might seek to quantify this perception by running an ordinary least-squares regression using "hours of study" as the single independent variable (i.e., attribute) and "pass/fail" as the

TABLE 9.12
Data on 20 Students and Their Test Performance and Hours of Study (C10T12)

Student No.	Hours of Study	Pass/Fail
1	2.5	0
2	22.6	1
3	17.8	0
4	5.4	0
5	14	0
6	13.3	1
7	26	1
8	33.1	1
9	13.6	0
10	45.3	1
11	1.9	0
12	31.4	1
13	27	1
14	10.1	0
15	2.7	0
16	16.3	1
17	14.5	1
18	4.5	0
19	22.6	1
20	17.1	1

FIGURE 9.14
Scatterplot of Student Performance (C10T12)

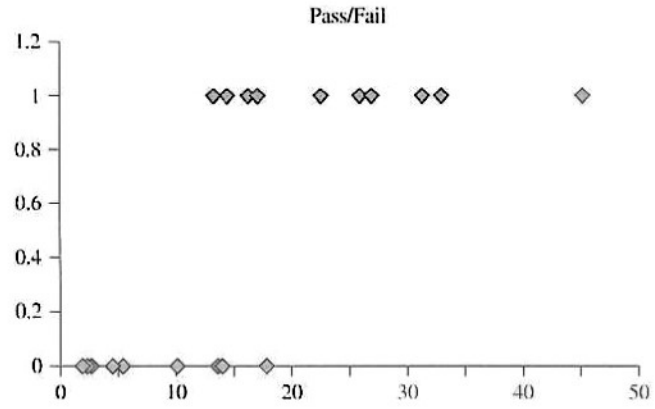


TABLE 9.13
Linear Least-Squares Regression (C10T12)

	Coefficients	Standard Error	T-test	P-value
Intercept	0.002053203	0.15	0.01	0.99
Hours of Study	0.032071805	0.01	4.47	0.00

dependent variable or target. Running this regression results in the output in Table 9.13.

Since the "hours of study" coefficient is positive (0.03), it appears to indicate that more study leads to a higher probability of passing. But is the relationship correctly quantified? Suppose a student studies for 100 hours. How well will this student do on the exam? Substituting into the regression equation, we have:

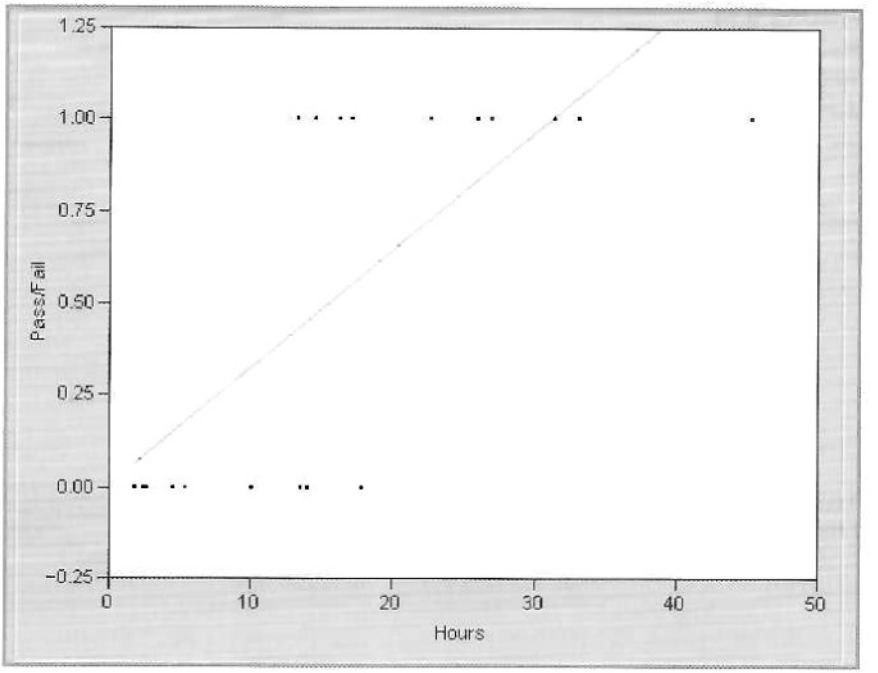
$$\text{Pass/fail} = 0.002053 + (0.032072)\ \text{X}\ (100)$$

$$\mathbf{3.209253} = 0.002053 + (0.032072)\ \text{X}\ (100)$$

What does this mean? Is the predicted grade 3.21 percent? That doesn't seem to make sense. Examining the regression line estimated and superimposing it on the data scatter may make the problem clear (see Figure 9.15).

The difficulty becomes clearer when examining the diagram. There are only two states of nature for the dependent variable (pass and fail). However, the regression line plotted in Figure 9.13 is a straight line sloping upward to the right and predicting values all along its path. When predicting the outcome from 100 hours of study, the regression chooses a number (i.e., 3.209253) that is much greater than the maximum value of 1 exhibited in the data set. Does this mean the individual has passed the test 3.21 times? Or does this mean that the expected score is 3.21 percent? Or does this have any meaningful explanation at all? This rather confusing result indicates that we have used an inappropriate tool in attempting to find the answer to our question. In earlier chapters, we assumed the dependent variable was continuous; this one is not. What we are attempting to estimate is

FIGURE 9.15
Linear Least-Squares Regression Plot



This function of the dependent variable is limited to values between zero and one. The function we use is called a ***logit***.

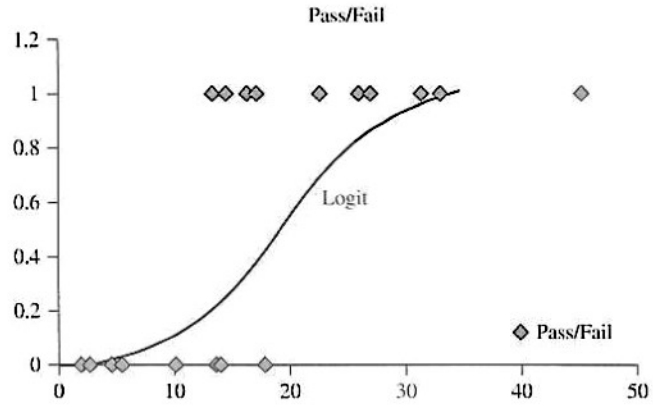
actually the probability of passing the exam; that will be a number between zero and one. Linear least-squares regression does not restrict the predictions of the dependent variable to a range of zero to one as we would like in this case.

We would like to use this same data but predict the probability that an individual would pass the test given a certain number of hours of study, but that will require a different algorithm. To accomplish this, we modify the linear least-squares model by modifying what we use as the target variable. Ordinarily, we simply use *Y, a numeric variable*, as the target variable; in logit, we use a function of *Y* as the dependent variable instead. This function of the dependent variable is limited to values between zero and one. The function we use is called a *logit*, and that is the reason the technique is called logistic regression.

The logit is $\text{Log}(e^{\alpha+\beta_1 X_1+\beta_2 X_2+\cdots+\beta_p X_p})$. You will recognize this as being similar to our explanation of the logistic curve in Chapter 3. In fact, the concepts are one and the same. We are going to use some knowledge of how growth works in the real world just as we did in Chapter 3. Recall that the diffusion models' assumed growth proceeded along an s-curve. When we used these models to predict new product sales, we did so in the knowledge that real-world new products almost always follow such a path. We now make the same assumption that real-world probabilities will behave in a similar manner. This assumption has withstood the test of time, as logistic regression has proven very predictive and accurate in actual practice.

The logistic regression model will estimate a value for pass/fail as a probability with zero as the minimum and one as the maximum. If we were to look

FIGURE 9.16
The Logit Estimated for the Student Data



at the entire range of values that a logistic regression would estimate for the student data, it would appear like the s-curve in Figure 9.16. Recall that "hours of study" are represented on the *x*-axis, while "pass/fail" is represented on the *y*-axis. If for instance, an individual had studied for a scant 10 hours, the model would predict a probability of passing somewhere near 10 percent (since the *y*-axis represents the values from zero to one, it can be read directly as the probability of occurrence of the dependent event). However, if the individual in question were to have studied for 30 hours, the probability of passing is predicted to be near 90 percent.

BANK DISTRESS

One example of the use of logit would be for a banking commission to predict the probability that any particular bank is likely to face distress in the near future. Bank distress is a condition where the bank cannot meet, or has difficulty paying off, its financial obligations to its creditors. Financial analysts wishing to assess the performance of a bank look for ways to measure the financial and economic consequences of past management decisions that have shaped the investments, assets, expenses, and financing of the bank. While there are many tools that address measurement in very specific and often narrow ways, analysts often "run the numbers" by calculating illuminating ratios. Ratios are easy to calculate, and a few selected ratios will often yield the information the analyst is seeking.

Ratios, to be useful, should have clear meaning and be predictive of the item being forecasted. In the past, analysts have looked at individual ratios as simple tests to be examined one at a time, but logit allows the ratios that have been useful in the past to be used together in a single predictive model with much more predictive power than any single ratio alone. While there are differences in accounting methods among various industries, there are often rigid standards applied by the government to the banking community.

Bank regulators need a way to identify banks that are in distress; the costs of a bank failing are quite costly, not only to the bank's investors and owners but also to the bank's customers. The effect on a single bank failure may also lead to failures in related banks. Because of this, the European Banking Authority (EBA) does stress testing on banks routinely (you can even see some of their methodology online[11]). Let's use a simplified example of how logit may be used with the ratios the EBA uses.

In our data set are 25 banks for which we have the values of two ratios; we also know whether each of these banks is in distress. The banks are shown in Table 9.14.

We will begin our analysis by using only a single ratio (total loans and leases divided by assets) as the attribute and using the "Distress" variable as the target. The goal is to create a model that is able to predict which banks are in distress by using ratio analysis. Running a logit on the entire data set of 25 banks with that variable gives the result shown in Table 9.15.

TABLE 9.14
Stress Data. Stressed Banks (Ones in Distress) Are Labeled as "1," While Banks That Are Strong and Not Stressed (I.e., Strong Banks) Are Labeled as "0." (C9T14)

Obs	Stress	Total Loans and Leases to Assets	Total Expenses to Assets
1	1	0.65	0.12
2	1	0.7	0.12
3	1	0.66	0.11
4	1	0.92	0.09
5	1	0.69	0.11
6	1	0.74	0.14
7	1	0.75	0.12
8	1	0.75	0.12
9	1	0.7	0.16
10	1	0.64	0.13
11	1	0.64	0.11
12	1	1.01	0.11
13	0	1.04	0.1
14	0	0.51	0.09
15	0	0.3	0.08
16	0	0.55	0.1
17	0	0.6	0.13
18	0	0.54	0.08
19	0	0.43	0.08
20	0	0.52	0.07
21	0	0.54	0.08
22	0	0.3	0.09
23	0	0.67	0.07
24	0	0.79	0.12
25	0	0.46	0.08

[11] http://www.eba.europa.eu/risk-analysis-and-data

TABLE 9.15 Logit Model for Bank Distress Given Only One Ratio as a Predictor: Total Loans and Leases to Assets (C9T14)

Predictor	Estimate	Confidence Interval: Lower	Confidence Interval: Upper	Odds	Standard Error	Chi2-Statistic	P-Value
Intercept	−5.1354956	4.884E-05	0.708943	0.005884	2.444695953	4.41280786	0.03567
Total Loans and Leases to Assets	7.84907599	1.7020223	3860606	2563.365	3.733363923	4.420143959	0.035517

Source: Frontline Systems Inc.

The P-value on the ratio is 0.035, indicating that there is a significant effect of the ratio on bank distress. To interpret the result, however, is probably best done by graphing the resulting logit.

The logit itself is not the probability. However, we are able to convert the logit (or log odds) into the probability by using the formula:

$$Probability = \frac{Odds}{1 + Odds}$$

Let's examine two different banks with different Total Loans and Leases to Assets ratios and predict the probability that each bank is in distress by calculating the probability that the bank is in the "1" class as opposed to the "0" class.

Bank number one has a Total Loans and Leases to Assets ratio of 0.6; Bank number two has a Total Loans and Leases to Assets ratio of 0.8. Note that the two financial ratios for the banks are quite close; they differ only by 0.2.

For bank number one, the probability of being in the "1" class (i.e., stressed) is:

$$\text{Probability} = \frac{e^{(-5.1355 + (7.8491\ X\ 0.6))}}{(1 + -5.1355 + (7.8491\ X\ 0.6))}$$

$$\text{Probability} = 0.3951 \text{ or } \mathbf{39.51\%}$$

For bank number two, the probability of being in the "1" class (i.e., stressed) is:

$$\text{Probability} = \frac{e^{(-5.1355 + (7.8491\ X\ 0.8))}}{(1 + -5.1355 + (7.8491\ X\ 0.8))}$$

$$\text{Probability} = 0.7584 \text{ or } \mathbf{75.84\%}$$

Notice that as the ratio increased only slightly, the probability that the bank was in distress increased by a much larger amount. Probabilities are not linear (they follow a logit pattern), and that is a compelling reason to use logit instead of ordinary least-squares regression in this case. Figure 9.17 makes this clear by examining the two different financial ratios for the two different banks; bank number one with only a 39.51 percent probability of being in distress would be classified by the logit

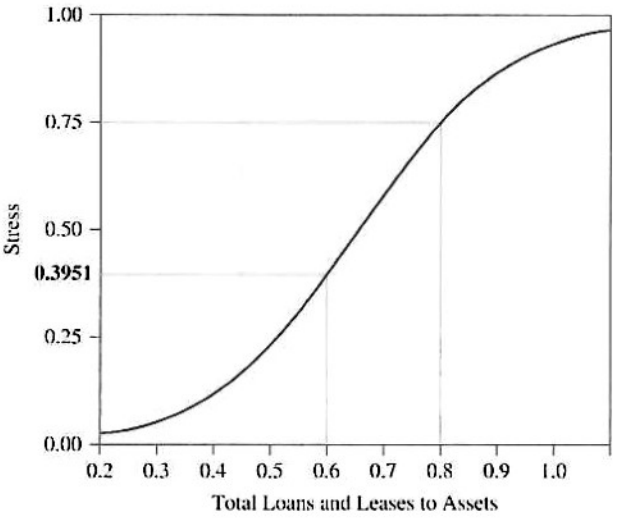


FIGURE 9.17
Logit Plot Using the Attributes Total Loans and Leases to Total Assets

algorithm as "nonstressed." But bank number two with a 75.84 percent probability of being in distress would be classified by the logit algorithm as "stressed," assuming we used a cutoff value of 0.5 to separate stressed from nonstressed.

Table 9.16 also offers a second financial ratio that could be used in testing; Total Expenses to Assets. Could this second financial ratio also be useful in predicting bank stress? Once again, we will run a logit model with the dichotomous stress variable as the target and the financial ratio as the attribute. The goal again is to classify banks as either "stressed" or "nonstressed."

Let's examine the results of estimating the logit for the same two banks. They have different Total Expenses to Total Assets ratios. We wish again to predict the probability that each bank is in distress by calculating the probability that the bank is in the "1" class (stressed) as opposed to the "0" class (nonstressed).

TABLE 9.16 Logit Model for Bank Distress, Given Only One Ratio as a Predictor: Total Expenses to Assets (C9T14)

Predictor	Estimate	Confidence Interval: Lower	Confidence Interval: Upper	Odds	Standard Error	Chi2-Statistic	P-Value
Intercept	−9.8348255	3.616E-08	0.079321	5.36E-05	3.724852799	6.971321055	0.008283
Total Expenses to Assets	93.2988115	5.666E+10	Inf	3.3E+40	34.96924312	7.11835689	0.00763

Source: Frontline Systems Inc.

Bank number one has a Total Expenses to Total Assets ratio of 0.10; Bank number two has a Total Expenses to Total Assets ratio of 0.14. Note again that the two financial ratios for the banks are quite close; they differ only by 0.04 in this case.

For bank number one, the probability of being in the "1" class (i.e., stressed) is:

$$\text{Probability} = \frac{e^{(-9.8348 + (93.2988\ X\ 0.10))}}{(1 + -9.8348 + (93.2988\ X\ 0.10))}$$

$$\text{Probability} = 0.3763 \text{ or } \mathbf{37.63\%}$$

For bank number two, the probability of being in the "1" class (i.e., stressed) is:

$$\text{Probability} = \frac{e^{(-9.8348 + (93.2988\ X\ 0.14))}}{(1 + -9.8348 + (93.2988\ X\ 0.14))}$$

$$\text{Probability} = 0.9618 \text{ or } \mathbf{96.18\%}$$

Figure 9.18 displays the results of using the logit estimate to plot the probability that either bank will be in the "stressed" category. Bank number one with a Total Expenses to Total Assets ratio of 0.10 has only a 37.63 percent chance of being in the "stressed" category and would be classified by the logit algorithm as a "nonstressed" bank. Bank number two, however, with a Total Expenses to Total Assets ratio of 0.14 has a 96.18 percent chance of being in the "stressed" class; this bank would be classified by the logit algorithm as being "stressed."

Logistic regression is not limited to using only a single attribute; in our bank stress example, we could use both financial ratios at the same time, probably with even better results. Could more than two financial ratios be used? Yes, much as with multiple linear regression, we may use multiple attributes.

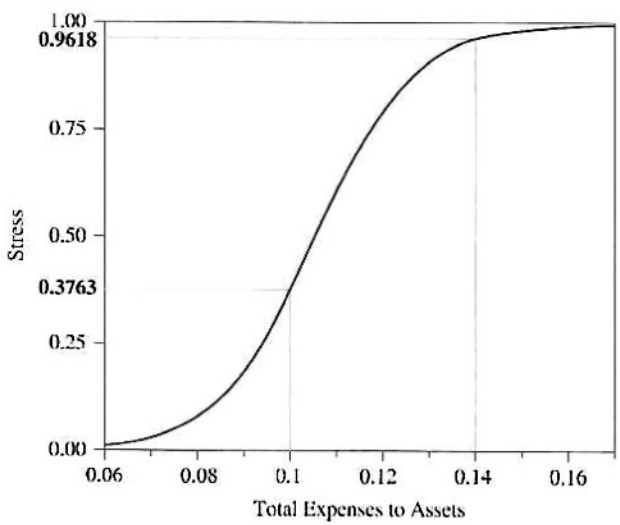


FIGURE 9.18
Logit Plot Using the Attribute Total Expenses to Total Assets

Summary

In this chapter, we have covered four classification techniques that are commonly used by real data miners. Classification, however, is only a single aspect of data mining. In general, there is no one best classification technique; the individual data in a particular situation will determine the best technique to use. The diagnostic statistics will lead the researcher to choose an appropriate model; there may be no single optimal model.

Data mining also uses other tools such as clustering analysis, and this will be covered in Chapter 10. The growth in the use of commercial data mining tools rivals the growth in business forecasting software sales; SAS Enterprise Miner, Frontline Solver, and IBM/SPSS Modeler have become important additions to the forecaster's toolkit in recent years.

Suggested Readings

Berry, Michael J. A.; and Gordon S. Linhoff. *Data Mining Techniques for Marketing, Sales, and Customer Relationship Management*. Indianapolis, IN: Wiley Publishing, Inc., 2004.

Cortada, James W. *The Digital Hand, Volume 2*. Oxford, U.K.: Oxford University Press, 2006.

Cramer, J. S. *Logit Models*. Cambridge, U.K.:Cambridge University Press, 2003.

De Ville, Barry. *Decision Trees for Business Intelligence and Data Mining*. Cary, NC: SAS Publishing, 2006.

Foreman, John W. *Data Smart*. Indianapolis, IN: John Wiley & Sons, Inc., 2014

Hilbe, Joseph M. *Logistic Regression Models*. Boca Raton, FL: Chapman & Hall/CRC, 2009.

Keating, Barry. "Analytics Off the Shelf." *Applied Marketing Analytics*, 2, no. 1 (Winter 2015–16), pp. 12–24.Keough, Eamonn. We would be remiss in not mentioning the tools Eamonn Keough has provided for this chapter to help explain the process of data mining; you may wish to examine his website (http://www.cs.ucr.edu/~eamonn/).

Mayer-Schonberger, Viktor; and Kenneth Cukier. *Big Data: A Revolution That Will Transform How We Live, Work, and Think*. New York: Houghton Mifflin Harcourt, 2013.

McGrayne, Sharon Bertsch. *The Theory That Would Not Die: How Bayes Rule Cracked the Enigma Code, Hunted Down Russian Submarines and Emerged Triumphant from Two Centuries of Controversy*. New Haven, CT.: Yale University Press, 2011.

SAS Institute. *Applied Analytics Using SAS Enterprise Miner*. Carey, NC.: SAS Institute, Inc., 2011.

Shannon, Claude. "A Mathematical Theory of Communication," *Bell System Techincal Journal*, 27 (July, October 1948), pp. 379–423, 623–656.

Shmueli, Galit; Nitin R. Patel; and Peter C. Bruce. *Data Mining for Business Analytics*. Hoboken, NJ: John Wiley & Sons, Inc., 2016.

Siegel, Eric. *Predictive Analytics*. Hoboken, NJ: John Wiley & Sons, Inc., 2013.

Singh, Simon. *The Code Book: The Science of Secrecy from Ancient Egypt to Quantum Cryptography*. New York: Anchor Books, 2000.

Wendler, Tito; and Soren Grottrup. *Data Mining with SPSS Modeler*. Switzerland: Springer International Publishing, 2016.

Witten, Ian H.; and Eibe Frank. *Data Mining: Practical Machine Learning Tools and Techniques*. Amsterdam: Elsevier, 2005.

Exercises

1. A data mining algorithm has been applied to a transaction dataset and has classified 88 records as fraudulent (30 correctly so) and 952 as nonfraudulent (920 correctly so). Which of the following situations represents the confusion matrix for the transactions data mentioned? Explain your reasoning.

A.

Classification Confusion Matrix		
	Predicted Class	
Actual Class	1	0
1	58	920
0	30	32

B.

Classification Confusion Matrix		
	Predicted Class	
Actual Class	1	0
1	32	30
0	58	920

C.

Classification Confusion Matrix		
	Predicted Class	
Actual Class	1	0
1	30	32
0	58	920

D.

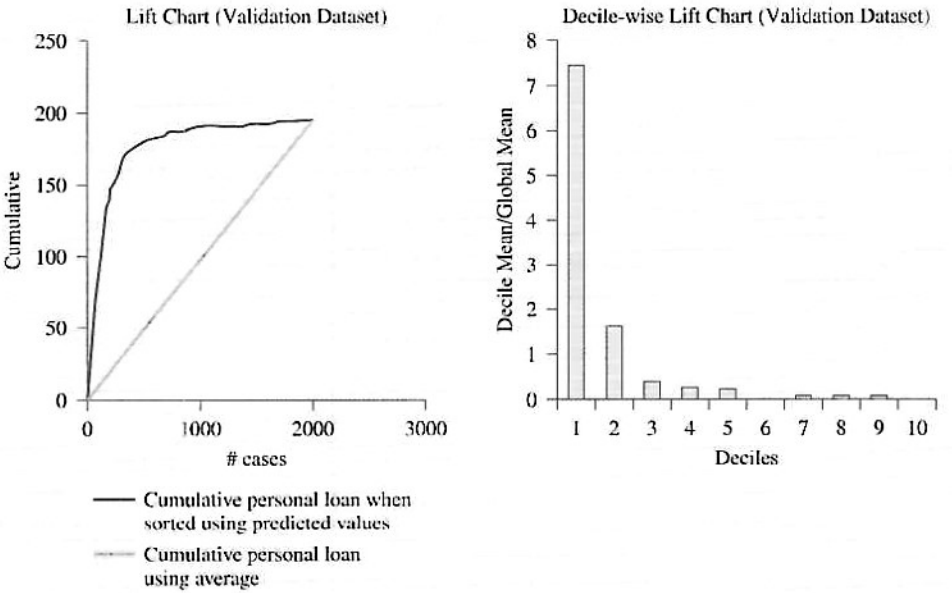
Classification Confusion Matrix		
	Predicted Class	
Actual Class	1	0
1	920	58
0	30	32

2. Calculate the classification error rate for the following confusion matrix. Comment on the pattern of misclassifications. How much better did this data mining technique do compared to a naive model? What is the misclassification rate for the naive model?

	Predict Class 1	Predict Class 0
Actual 1	8	2
Actual 0	20	970

3. Explain what is meant by Bayes' theorem as used in the Naive Bayes model.

4. Explain the difference between a training data set and a validation data set. Why are these data sets used routinely with data mining techniques in the XLMiner© program and not used in the ForecastX™ program? Is there, in fact, a similar technique presented in a previous chapter that is much the same as partitioning a data set?
5. For a data mining classification technique, the validation data set lift charts are shown below. What confidence in the model would you express, given this evidence?



Source: Frontline Systems Inc.

6. Wine (in this case, red wine) has been graded for many years by experts who actually taste a sample of the wine, examine its color and aroma, and assign a grade (in our case, high quality or lower quality). Would it be possible, however, to use attributes of the wine that are machine measurable such as fixed acidity and residual sugar to classify the wines?

   Partition the data into two partitions (60 percent and 40 percent, respectively), the training data and the validation data. Estimate a kNN algorithm for the training data, and examine the resulting estimation.

   Explain the overall misclassification rate for the validation partition and its calculation. What would have been the validation misclassification rate if you had used the naive model?

   What does the validation confusion matrix tell the data scientist? Is the algorithm more likely to make one type of error than the other?

   The lift chart is potentially the most important information provided by the algorithm. Examine either the cumulative gains chart, the lift chart proper, or the decile-wise lift chart. Do they each tell much the same story? How would you explain one of these lift charts to someone unfamiliar with predictive analytics?

7. Use the red wine data again with the same partition and estimate a classification model with Naive Bayes.

   Again explain the overall misclassification rate for the validation data. Is it different for the Naive Bayes algorithm and the kNN algorithm?

   Is the confusion matrix for the validation data different than the one obtained with the kNN algorithm?

   Finally, examine the lift chart and make a comparison with the kNN model.

8. The bank marketing data includes actual information for a direct marketing effort by a bank. We will attempt to construct a model with just a few of the available attributes. We are interested in classifying whether a customer will respond positively to the marketing effort offering a term deposit.

   The attributes you are to use are age, balance, duration, campaign, pdays, and previous; explanations of these numeric variables are in the file. After your initial analysis, you may wish to transform some of the remaining variables to attempt to estimate a more complete model.

   Use a logit model for the estimate, making sure to request an "analysis of coefficients" in XLMiner®. As usual, use a 60/40 split for the training and validation data sets, and request a full set of lift charts.

   Does this estimate, using only some of the available attributes, do better than a naive model in the overall misclassification rate? What if you examine the lift chart? Recall that the lift chart reorders the data from most likely to accept a marketing offer to least likely to accept such an offer. Now does the algorithm appear to have explanatory power (i.e., could you successfully use it to suggest who to market to in the first place)?

   Which of the attributes that you selected appear to have the greatest effect on the classification? How certain are you that these attributes have an effect on the classification?

   By creating dummy variables and categorical variables for the attributes that you did not use in this exercise already, you may extend the analysis in order to refine the algorithm. Evaluate the resulting output in the same manner described above and compare the two outputs. Did the addition of the extra attributes to the logit model add additional explanatory power?

9. The Boston housing data includes information from the 1970 U.S. census for the city of Boston and surrounding area. Note that there are two variables representing value; one is "Medv," which is a dollar value. The other is "Cat Medv," which is a binary variable indicating whether the house is of "high value" (signified by "1") or "lower value" (identified by "0").

   Estimate a classification tree using the Cat Medv variable as the target. You are trying to classify home as either high value or lower value by using the given attributes (such as the number of rooms in the house, the crime rate in the local area, and the age of the houses in the area). Use all of the attributes in the file to estimate the model requesting the "best pruned tree" to prevent overfitting. For display, however, request the "full tree."

   Evaluate the estimate for the best pruned tree using the confusion matrix, the misclassification rate, and, most importantly, the lift chart.

   By examining the full tree, you should be able to see how the CART algorithm will attempt to perfectly classify the records. In doing so, it may overfit the data, and that is the reason for using the pruning method.

   Now re-estimate the algorithm using the target Medv this time. In order to do so, you will have to use the "Prediction" menu in XLMiner® and select "Regression Tree."

The method is similar to the classification tree estimated earlier, but now an actual numerical prediction is being requested. Overfitting remains a possible problem, and so it will again be necessary to prune using either the best pruned tree or the minimum error tree.

10. The credit card fraud data is a small version (comprised of 14,240 records) of a much larger data set (containing 248,807 records); it is made up of 2013 European transactions. It is a very unbalanced data set in which there are only a few fraudulent transactions. Attempting to classify transactions as fraudulent will be difficult since there are very few instances of fraud.

    Use logit and a kNN model to create a predictive model for the credit card fraud data. Does either of these models have predictive power?

    Explain carefully the information provided by the lift chart or the decile-wise lift chart; how does this information differ from the information provided by the overall misclassification rate?

    What value to a firm could you see in creating such a model and using it in real time?