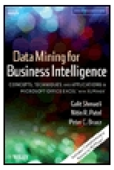


Data Mining for Business Intelligence: Concepts, Techniques, and Applications in Microsoft Office Excel with XLMiner, Second Edition

by Galit Shmueli, Nitin R. Patel and Peter C. Bruce

John Wiley & Sons (US). (c) 2010. Copying Prohibited.

Reprinted for Ana Maria TUTA OSMAN, SAP

ANA.MARIA.TUTA.OSMAN@SAP.COM

Reprinted with permission as a subscription benefit of **Skillport**,
http://skillport.books24x7.com/

All rights reserved. Reproduction and/or distribution in whole or in part in electronic,paper or other forms without written permission is prohibited.

Chapter 5: Evaluating Classification and Predictive Performance

In this chapter we discuss how the predictive performance of data mining methods can be assessed. We point out the danger of overfitting to the training data and the need for testing model performance on data that were not used in the training step. We discuss popular performance metrics. For prediction, metrics include Average Error, MAPE, and RMSE (based on the validation data). For classification tasks, metrics include the classification matrix, specificity and sensitivity, and metrics that account for misclassification costs. We also show the relation between the choice of cutoff value and method performance, and present the receiver operating characteristic (ROC) curve, which is a popular plot for assessing method performance at different cutoff values. When the goal is to accurately classify the top tier of a new sample rather than accurately classify the entire sample (e.g., the 10% of customers most likely to respond to an offer), lift charts are used to assess performance. We also discuss the need for oversampling rare classes and how to adjust performance metrics for the oversampling. Finally, we mention the usefulness of comparing metrics based on the validation data to those based on the training data for the purpose of detecting overfitting. While some differences are expected, extreme differences can be indicative of overfitting.

5.1 Introduction

In supervised learning we are interested in predicting the class (classification) or continuous value (prediction) of an outcome variable. In Chapter 2 we worked through a simple example. Let us now examine the questions of how to judge the usefulness of a classifier or predictor and how to compare different ones.

5.2 Judging Classification Performance

The need for performance measures arises from the wide choice of classifiers and predictive methods. Not only do we have several different methods, but even within a single method there are usually many options that can lead to completely different results. A simple example is the choice of predictors used within a particular predictive algorithm. Before we study these various algorithms in detail and face decisions on how to set these options, we need to know how we will measure success.

A natural criterion forjudging the performance of a classifier is the probability of making a *misclassification error*. Misclassification means that the observation belongs to one class but the model classifies it as a member of a different class. A classifier that makes no errors would be perfect, but we do not expect to be able to construct such classifiers in the real world due to "noise" and to not having all the information needed to classify cases precisely. Is there a minimal probability of misclassification that we should require of a classifier?

Benchmark: The Naive Rule

A very simple rule for classifying a record into one of m classes, ignoring all predictor information (X_1, X_2, … X_p) that we may have, is to classify the record as a member of the majority class. In other words, "classify as belonging to the most prevalent class." The *naive rule* is used mainly as a baseline or benchmark for evaluating the performance of more complicated classifiers. Clearly, a classifier that uses external predictor information (on top of the class membership allocation) should outperform the naive rule. There are various performance measures based on the naive rule that measure how much better than the naive rule a certain classifier performs. One example is the multiple R^2 reported by XLMiner, which measures the distance between the fit of the classifier to the data and the fit of the naive rule to the data (for further details, see Section 10.5.

The equivalent of the naive rule for classification when considering a quantitative response is to use $\hat{y}$, the sample mean, to predict the value of y foranew record. In both cases the predictions rely solely on the y information and exclude any additional predictor information.

Class Separation

If the classes are well separated by the predictor information, even a small dataset will suffice in finding a good classifier, whereas if the classes are not separated at all by the predictors, even a very large dataset will not help. Figure 5.1 illustrates this for a two-class case. Figure 5.1(a) includes a small dataset (n = 24 observations) where two predictors (income and lot size) are used for separating owners from nonowners [we thank Dean Wichern for this example, described in Johnson and Wichern (2002)]. Here, the predictor information seems useful in that it separates the two classes (owners/nonowners). Figure 5.1(b) shows a much larger dataset (n = 5000 observations) where the two predictors (income

John Wiley & Sons (US), John Wiley & Sons, Inc. (c) 2010, Copying Prohibited

and average credit card spending) do not separate the two classes well in most of the higher ranges (loan acceptors/nonacceptors).

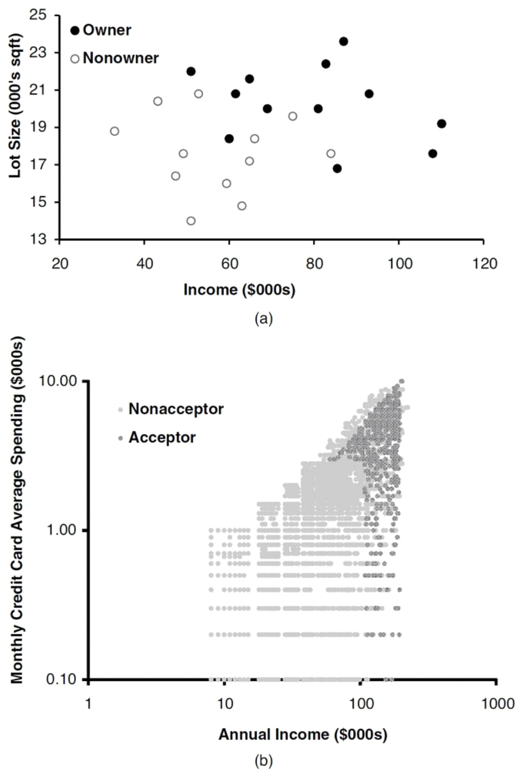
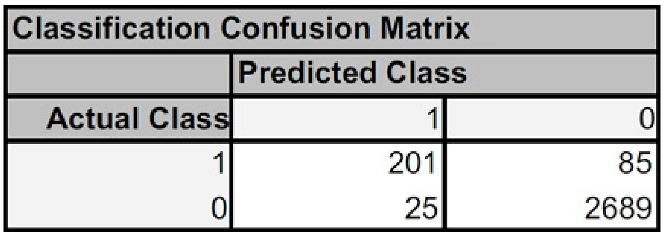


FIGURE 5.1: *(A)* HIGH AND *(B)* LOW LEVELS OF SEPARATION BETWEEN TWO CLASSES, USING TWO PREDICTORS



Classification Confusion Matrix		
	Predicted Class	
Actual Class	1	0
1	201	85
0	25	2689

FIGURE 5.2: CLASSIFICATION MATRIX BASED ON 3000 OBSERVATIONS AND TWO CLASSES

Classification Matrix

In practice, most accuracy measures are derived from the *classification matrix* (also called the *confusion matrix)*. This matrix summarizes the correct and incorrect classifications that a classifier produced for a certain dataset. Rows and columns of the classification matrix correspond to the true and predicted classes, respectively. Figure 5.2 shows an example of a classification (confusion) matrix for a two-class (0/1) problem resulting from applying a certain classifier to 3000 observations. The two diagonal cells (upper left, lower right) give the number of correct classifications, where the predicted class coincides with the actual class of the observation. The off-diagonal cells give counts of misclassification. The top right cell gives the number of class 1 members that were misclassified as 0's (in this example, there were 85 such misclassifications). Similarly, the lower left cell gives the number of class 0 members that were misclassified as 1's (25 such observations).

The classification matrix gives estimates of the true classification and mis-classification rates. Of course, these are estimates and they can be incorrect, but if we have a large enough dataset and neither class is very rare, our estimates will be reliable. Sometimes, we may be able to use public data such as U.S. Census data to estimate these proportions. However, in most practical business settings, we will not know them.

Using the Validation Data

Reprinted for WRFSM/I300634, SAP John Wiley & Sons (US), John Wiley & Sons, Inc. (c) 2010, Copying Prohibited

To obtain an honest estimate of classification error, we use the classification matrix that is computed from the *validation data*. In other words, we first partition the data into training and validation sets by random selection of cases. We then construct a classifier using the training data and apply it to the validation data. This will yield the predicted classifications for observations in the validation set (see Figure 2.4). We then summarize these classifications in a classification matrix. Although we can summarize our results in a classification matrix for training data as well, the resulting classification matrix is not useful for getting an honest estimate of the misclassification rate for new data due to the danger of overfitting.

In addition to examining the validation data classification matrix to assess the classification performance on new data, we compare the training data classification matrix to the validation data classification matrix, in order to detect overfitting: although we expect inferior results on the validation data, a large discrepancy in training and validation performance might be indicative of over-fitting.

Accuracy Measures

Different accuracy measures can be derived from the classification matrix. Consider a two-class case with classes C_0 and C_1 (e.g., buyer/nonbuyer). The schematic classification matrix in Table 5.1 uses the notation $n_{i,J}$ to denote the number of cases that are class C_i members and were classified as C_j members. Of course, if $i \neq j$, these are counts of misclassifications. The total number of observations is $n = n_{0,0} + n_{0,1} + n_{1,0} + n_{1,1}$.

Table 5.1: CLASSIFICATION MATRIX: MEANING OF EACH CELL

	Predicted Class	
Actual Class	**C_0**	**C_1**
C_0	$n_{0,0}$ = number of C_0 cases classified correctly	$n_{0,1}$ = number of C_0 cases classified incorrectly as C_1
C_1	$n_{1,0}$ = number of C_1 cases classified incorrectly as C_0	$n_{1,1}$ = number of C_1 cases classified correctly

A main accuracy measure is the *estimated misclassification rate*, also called the *overall error* rate. It is given by

$$\text{err} = \frac{n_{0,1} + n_{1,0}}{n},$$

where n is the total number of cases in the validation dataset. In the example in Figure 5.2, we get err = (25 + 85)/3000 = 3.67%.

We can measure accuracy by looking at the correct classifications instead of the misclassifications. The *overall accuracy* of a classifier is estimated by

$$\text{Accuracy} = 1 - \text{err} = \frac{n_{0,0} + n_{1,1}}{n}.$$

In the example we have (201 + 2689)/3000 = 96.33.

Cutoff for Classification

The first step in most classification algorithms is to estimate the probability that a case belongs to each of the classes. If overall classification accuracy (involving all the classes) is of interest, the case can be assigned to the class with the highest probability. In many cases, a single class is of special interest, so we will focus on that particular class and compare the estimated probability of belonging to that class to a *cutoff value*. This approach can be used with two classes or more than two classes, though it may make sense in such cases to consolidate classes so that you end up with two: the class of interest and all other classes. If the probability of belonging to the class of interest is above the cutoff, the case is assigned to that class.

The default cutoff value in two-class classifiers is 0.5. Thus, if the probability of a record being a class 1 member is greater than 0.5, that record is classified as a 1. Any record with an estimated probability of less than 0.5 would be classified as a 0. It is possible, however, to use a cutoff that is either higher or lower than 0.5. A cutoff greater than 0.5 will end up classifying fewer records as 1's, whereas a cutoff less than 0.5 will end up classifying more records as 1. Typically, the

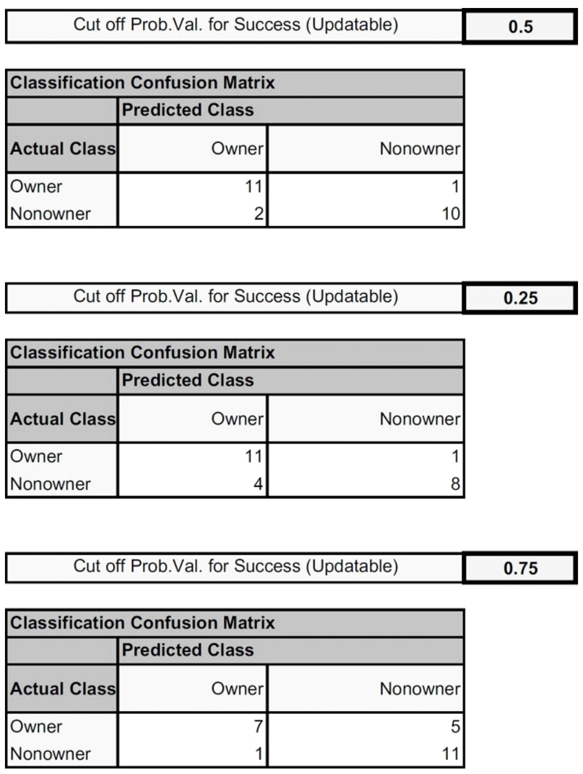
Reprinted for WRFSM/I300634, SAP John Wiley & Sons (US), John Wiley & Sons, Inc. (c) 2010, Copying Prohibited

misclassification rate will rise in either case.

Consider the data in Table 5.2, showing the actual class for 24 records, sorted by the probability that the record is a 1 (as estimated by a data mining algorithm). If we adopt the standard 0.5 as the cutoff, our misclassification rate is 3/24, whereas if we instead adopt a cutoff of 0.25, we classify more records as 1's and the misclassification rate goes up (comprising more 0's misclassified as 1's) to 5/24. Conversely, if we adopt a cutoff of 0.75, we classify fewer records as 1's. The misclassification rate goes up (comprising more 1's misclassified as 0's) to 6/24. All this can be seen in the classification tables in Figure 5.3.

Table 5.2: 24 RECORDS WITH THEIR ACTUAL CLASS AND PROBABILITY OF BEING CLASS 1 MEMBERS, AS ESTIMATED BY A CLASSIFIER

Actual Class	Probability of Class 1	Actual Class	Probability of Class 1
1	0.995976726	1	0.505506928
1	0.987533139	0	0.47134045
1	0.984456382	0	0.337117362
1	0.980439587	1	0.21796781
1	0.948110638	0	0.199240432
1	0.889297203	0	0.149482655
1	0.847631864	0	0.047962588
0	0.762806287	0	0.038341401
1	0.706991915	0	0.024850999
1	0.680754087	0	0.021806029
1	0.656343749	0	0.016129906
0	0.622419543	0	0.003559986



Cut off Prob.Val. for Success (Updatable)	0.5

Classification Confusion Matrix		
	Predicted Class	
Actual Class	Owner	Nonowner
Owner	11	1
Nonowner	2	10

Cut off Prob.Val. for Success (Updatable)	0.25

Classification Confusion Matrix		
	Predicted Class	
Actual Class	Owner	Nonowner
Owner	11	1
Nonowner	4	8

Cut off Prob.Val. for Success (Updatable)	0.75

Classification Confusion Matrix		
	Predicted Class	
Actual Class	Owner	Nonowner
Owner	7	5
Nonowner	1	11

FIGURE 5.3: CLASSIFICATION MATRICES BASED ON CUTOFFS OF 0.5, 0.25, AND 0.75

To see the entire range of cutoff values and how the accuracy or misclassifi-cation rates change as a function of the cutoff, we can use one-variable tables in Excel (see the accompanying box), and then plot the performance measure of interest

Reprinted for WRFSM/I300634, SAP John Wiley & Sons (US), John Wiley & Sons, Inc. (c) 2010, Copying Prohibited

versus the cutoff. The results for the data above are shown in Figure 5.4.

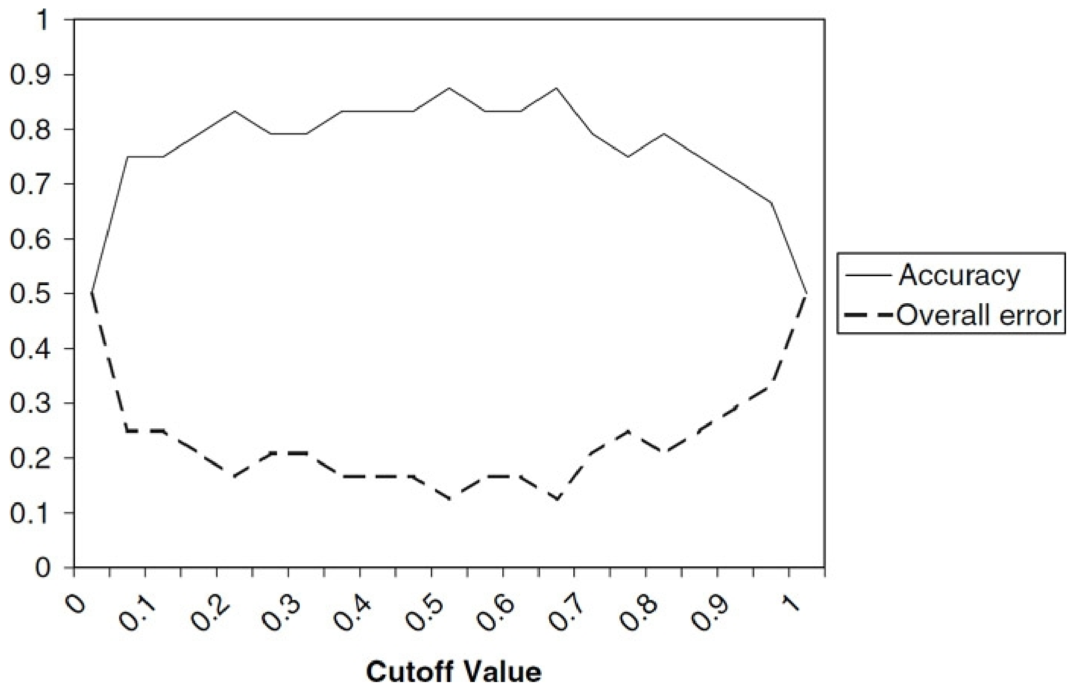


FIGURE 5.4: PLOTTING RESULTS FROM ONE-WAYTABLE: ACCURACY AND OVERALL ERROR AS A FUNCTION OF THE CUTOFF VALUE

We can see that the accuracy level is pretty stable around 0.8 for cutoff values between 0.2 and 0.8.

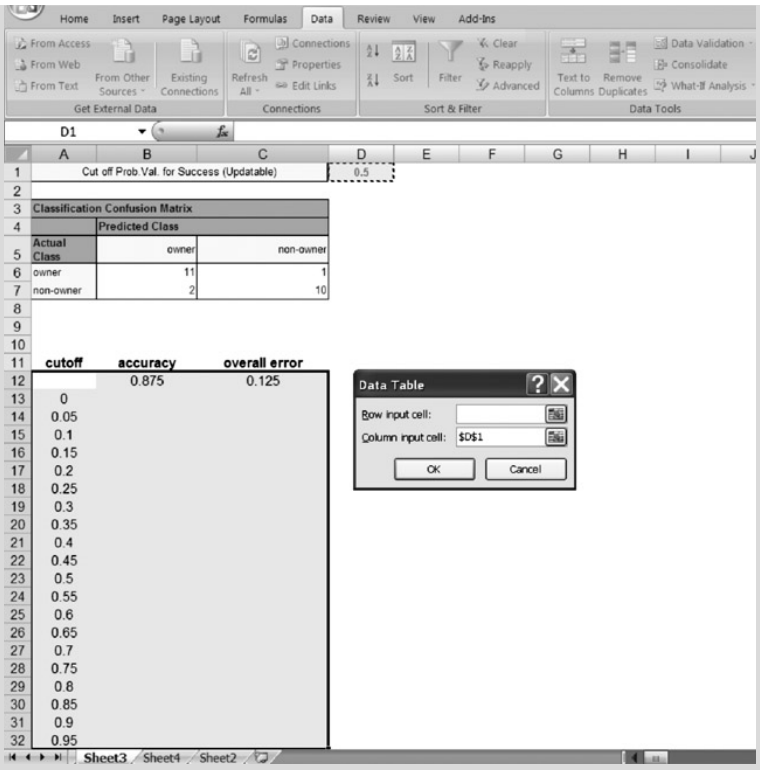
Why would we want to use cutoffs different from 0.5 if they increase the misclassification rate? The answer is that it might be more important to classify 1's properly than 0's, and we would tolerate a greater misclassification of the latter. Or the reverse might be true; in other words, the costs of misclassification might be asymmetric. We can adjust the cutoff value in such a case to classify more records as the high-value class (in other words, accept more misclassifications where the misclassification cost is low). Keep in mind that we are doing so after the data mining model has already been selected—we are not changing that model. It is also possible to incorporate costs into the picture before deriving the model. These subjects are discussed in greater detail below.

ONE-VARIABLE TABLES IN EXCEL

Excel's one-variable data tables are very useful for studying how the cutoff affects different performance measures. It will change the cutoff values to values in a user-specified column and calculate different functions based on the corresponding classification matrix. To create a one-variable data table (see Figure 5.5):

1. In the top row, create column names for each of the measures you wish to compute. (We created "overall error" and "accuracy" in B11 and C11.) The leftmost column should be titled "cutoff" (A11).
2. In the row below, add formulas, using references to the relevant classification matrix cells. [The formula in B12 is = (B6 + C_7)/(B6 + C6 + B_7 + C_7).]
3. In the leftmost column, list the cutoff values that you want to evaluate. (We chose 0, 0.05, …, 1 in B_{13} to B_{33}.)
4. Select the range excluding the first row (Bi2:C33). In Excel 2007 go to *Data* > *WhatifAnalysis* > *Data Table* (in Excel 2003 select *Table* from the *Data* menu).
5. In "column input cell," select the cell that changes (here, the cell with the cutoff value, Di). Click OK.
6. The table will now be automatically completed.

Reprinted for WRFSM/I300634, SAP
John Wiley & Sons (US), John Wiley & Sons, Inc. (c) 2010, Copying Prohibited



	A	B	C	D
1	Cut off Prob.Val. for Success (Updatable)			0.5
2				
3	Classification Confusion Matrix			
4		Predicted Class		
5	Actual Class	owner	non-owner	
6	owner	11	1	
7	non-owner	2	10	
8				
9				
10				
11	cutoff	accuracy	overall error	
12		0.875	0.125	
13	0			
14	0.05			
15	0.1			
16	0.15			
17	0.2			
18	0.25			
19	0.3			
20	0.35			
21	0.4			
22	0.45			
23	0.5			
24	0.55			
25	0.6			
26	0.65			
27	0.7			
28	0.75			
29	0.8			
30	0.85			
31	0.9			
32	0.95			

Data Table

Row input cell:

Column input cell: D1

OK Cancel

FIGURE 5.5: CREATING ONE-VARIABLE TABLES IN EXCEL. ACCURACY AND OVERALL ERROR ARE COMPUTED FOR DIFFERENT VALUES OF THE CUTOFF

Performance in Unequal Importance of Classes

Suppose that it is more important to predict membership correctly in class 1 than in class 0. An example is predicting the financial status (bankrupt/solvent) of firms. It may be more important to predict correctly a firm that is going bankrupt than to predict correctly a firm that is going to stay solvent. The classifier is essentially used as a system for detecting or signaling bankruptcy. In such a case, the overall accuracy is not a good measure for evaluating the classifier. Suppose that the important class is C_1. The following pair of accuracy measures are the most popular:

The sensitivity of a classifier is its ability to detect the important class members correctly. This is measured by $n_{1,1}/(n_{1,0} + n_{1,1})$, the percentage of C_1 members classified correctly.

The specificity of a classifier is its ability to rule out C_0 members correctly. This is measured by $n_{0,0}/(n_{0,0} + n_{0,1})$, the percentage of C_0 members classified correctly.

It can be useful to plot these measures versus the cutoff value (using one-variable tables in Excel, as described above) in order to find a cutoff value that balances these measures.

ROC Curve A more popular method for plotting the two measures is through *ROC* (receiver operating characteristic) *curves*. The ROC curve plots the pairs {sensitivity, 1-specificity} as the cutoff value increases from 0 and 1. Better performance is reflected by curves that are closer to the top left corner. The comparison curve is the diagonal, which reflects the performance of the naive rule, using varying cutoff values (i.e., setting different thresholds on the level of majority used by the majority rule). The ROC curve for our 24-case example above is shown in Figure 5.6.

FALSE-POSITIVE AND FALSE-NEGATIVE RATES

Sensitivity and specificity measure the performance of a classifier from the point of view of the "classifying agency" (e.g., a company classifying customers or a hospital classifying patients). They answer the question: How well does the classifier segregate the important class members? It is also possible to measure accuracy from the perspective of the entity that is being predicted (e.g., the customer or the patient), who asks: What is my chance of belonging to the important class? This question, however, is usually less relevant in a data mining application. The terms *false-positive*

Reprinted for WRFSM/I300634, SAP

John Wiley & Sons (US), John Wiley & Sons, Inc. (c) 2010, Copying Prohibited

rate and *false-negative rate*, which are sometimes used erroneously to describe 1-sensitivity and 1-specificity, are measures of performance from the perspective of the individual entity. They are defined as:

The false-positive rate is the proportion of C_1 predictions that are wrong: $n_{0,1}/(n_{0,1} + n_{1,1})$. Notice that this is a ratio within the column of C_1 predictions (i.e., it uses only records that were classified as C_1).

The false-negative rate is the proportion of C_0 predictions that are wrong: $n_{1,0}/(n_{0,0} + n_{1,0})$. Notice that this is a ratio within the column of C_0 predictions (i.e., it uses only records that were classified as C_0).

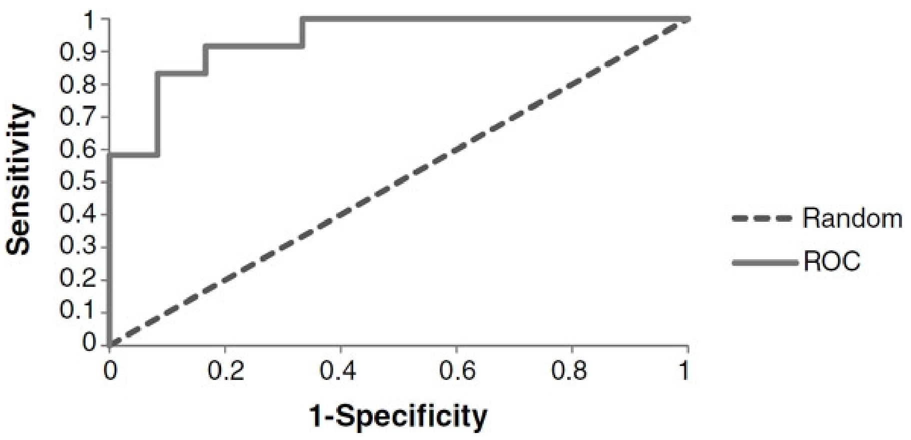


FIGURE 5.6: ROC CURVE FOR THE EXAMPLE

Lift Charts Let us continue further with the case in which a particular class is relatively rare and of much more interest than the other class: tax cheats, debt defaulters, or responders to a mailing. We would like our classification model to sift through the records and sort them according to which ones are most likely to be tax cheats, responders to the mailing, and so on. We can then make more informed decisions. For example, we can decide how many, and which tax returns to examine, looking for tax cheats. The model will give us an estimate of the extent to which we will encounter more and more noncheaters as we proceed through the sorted data starting with the records most likely to be tax cheats. Or we can use the sorted data to decide to which potential customers a limited-budget mailing should be targeted. In other words, we are describing the case when our goal is to obtain a rank ordering among the records according to their estimated probabilities of class membership.

In such cases, when the classifier gives a probability of belonging to each class and not just a binary classification to C_1 or C_0, we can use a very useful device known as the *lift curve*, also called a *gains curve* or *gains chart*. The lift curve is a popular technique in direct marketing. One useful way to think of a lift curve is to consider a data mining model that attempts to identify the likely responders to a mailing by assigning each case a "probability of responding" score. The lift curve helps us determine how effectively we can "skim the cream" by selecting a relatively small number of cases and getting a relatively large portion of the responders. The input required to construct a lift curve is a validation dataset that has been "scored" by appending to each case the estimated probability that it will belong to a given class.

Let us return to the example in Table 5.2. We have shown that different choices of a cutoff value lead to different classification matrices (as in Figure 5.3). Instead of looking at a large number of classification matrices, it is much more convenient to look at the *cumulative lift curve* (sometimes called a *gains chart)*, which summarizes all the information in these multiple classification matrices into a graph. The graph is constructed with the cumulative number of cases (in descending order of probability) on the *x* axis and the cumulative number of true positives on the *y* axis. Figure 5.7 gives the table of cumulative values of the class 1 classifications and the corresponding lift chart. The line joining the points (0, 0) to (24, 12) is a reference line. For any given number of cases (the *x*-axis value), it represents the expected number of C_1 predictions ifwe did not have a model but simply selected cases at random. It provides a benchmark against which we can see performance of the model. If we had to choose 10 cases as class 1 (the important class) members and used our model to pick the ones most likely to be 1's, the lift curve tells us that we would be right about 9 of them. If we simply select 10 cases at random, we expect to be right for $10 \times 12/24 = 5$ cases. The model gives us a "lift" in predicting class 1 of $9/5 = 1.8$. The lift will vary with the number of cases on which we choose to act. A good classifier will give us a high lift when we act on only a few cases (i.e., use the prediction for those at the top). As we include more cases, the lift will decrease. The lift curve for the best possible classifier—a classifier that makes no errors—would overlap the existing curve at the start,

John Wiley & Sons (US), John Wiley & Sons, Inc. (c) 2010, Copying Prohibited

continue with a slope of 1 until it reached 12 successes (all the successes), then continue horizontally to the right.

Serial no.	Predicted prob of 1	Actual Class	Cumulative Actual class
1	0.995976726	1	1
2	0.987533139	1	2
3	0.984456382	1	3
4	0.980439587	1	4
5	0.948110638	1	5
6	0.889297203	1	6
7	0.847631864	1	7
8	0.762806287	0	7
9	0.706991915	1	8
10	0.680754087	1	9
11	0.656343749	1	10
12	0.622419543	0	10
13	0.505506928	1	11
14	0.47134045	0	11
15	0.337117362	0	11
16	0.21796781	1	12
17	0.199240432	0	12
18	0.149482655	0	12
19	0.047962588	0	12
20	0.038341401	0	12
21	0.024850999	0	12
22	0.021806029	0	12
23	0.016129906	0	12
24	0.003559986	0	12

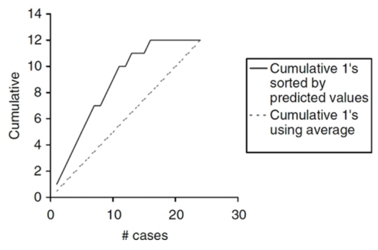


FIGURE 5.7: TABLE AND LIFT CHART FOR THE EXAMPLE

The same information can be portrayed as a *decile chart*, shown in Figure 5.8, which is widely used in direct marketing predictive modeling. The bars show the factor by which our model outperforms a random assignment of 0's and 1's, taking one decile at a time. Reading the first bar on the left, we see that taking the 10% of the records that are ranked by the model as "the most probable 1's" yields twice as many 1's as would a random selection of 10% of the records.

XLMinerautomatically creates lift (and decile) charts from probabilities predicted by classifiers for both training and validation data. Of course, the lift curve based on the validation data is a better estimator of performance for new cases.

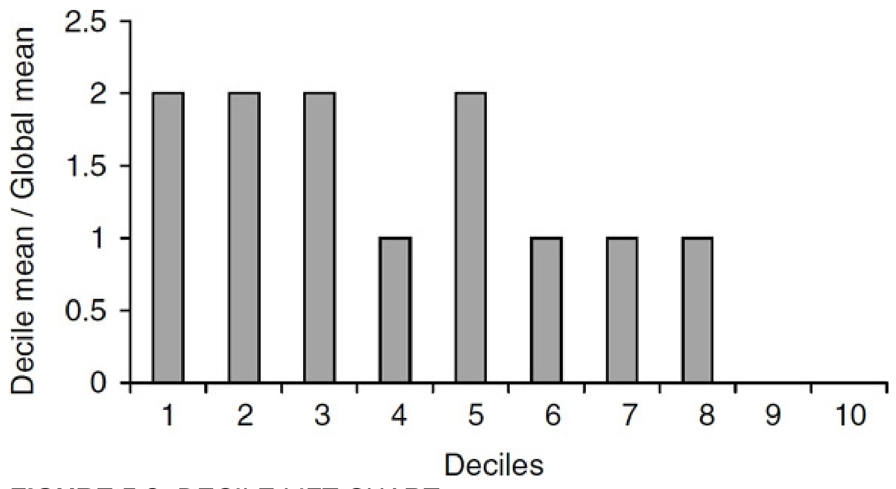


FIGURE 5.8: DECILE LIFT CHART

Asymmetric Misclassification Costs

Implicit in our discussion of the lift curve, which measures how effective we are in identifying the members of one particular

John Wiley & Sons (US), John Wiley & Sons, Inc. (c) 2010, Copying Prohibited

class, is the assumption that the error of misclassifying a case belonging to one class is more serious than for the other class. For example, misclassifying a household as unlikely to respond to a sales offer when it belongs to the class that would respond incurs a greater cost (the opportunity cost of the foregone sale) than the converse error. In the former case, you are missing out on a sale worth perhaps tens or hundreds of dollars. In the latter, you are incurring the costs of mailing a letter to someone who will not purchase. In such a scenario, using the misclassification rate as a criterion can be misleading.

Note that we are assuming that the cost (or benefit) of making correct classifications is zero. At first glance, this may seem incomplete. After all, the benefit (negative cost) of classifying a buyer correctly as a buyer would seem substantial. And in other circumstances (e.g., scoring our classification algorithm to fresh data to implement our decisions), it will be appropriate to consider the actual net dollar impact of each possible classification (or misclassification). Here, however, we are attempting to assess the value of a classifier in terms of classification error, so it greatly simplifies matters if we can capture all cost-benefit information in the misclassification cells. So, instead of recording the benefit of classifying a respondent household correctly, we record the cost of failing to classify it as a respondent household. It amounts to the same thing and our goal becomes the minimization of costs, whether the costs are actual costs or missed benefits (opportunity costs).

Consider the situation where the sales offer is mailed to a random sample of people for the purpose of constructing a good classifier. Suppose that the offer is accepted by 1% of those households. For these data, if a classifier simply classifies every household as a nonresponder, it will have an error rate of only 1% but it will be useless in practice. A classifier that misclassifies 2% of buying households as nonbuyers and 20% of the nonbuyers as buyers would have a higher error rate but would be better if the profit from a sale is substantially higher than the cost of sending out an offer. In these situations, if we have estimates of the cost of both types of misclassification, we can use the classification matrix to compute the expected cost of misclassification for each case in the validation data. This enables us to compare different classifiers using overall expected costs (or profits) as the criterion.

Suppose that we are considering sending an offer to 1000 more people, 1% of whom respond (1), on average. Naively classifying everyone as a 0 has an error rate of only 1%. Using a data mining routine, suppose that we can produce these classifications:

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

These classifications have an error rate of 100 × (20 + 2)/1000 = 2.2%—higher than the naive rate.

Now suppose that the profit from a 1 is $10 and the cost of sending the offer is $1. Classifying everyone as a 0 still has a misclassification rate of only 1% but yields a profit of $0. Using the data mining routine, despite the higher misclassification rate, yields a profit of $60.

The matrix of profit is as follows (nothing is sent to the predicted 0's so there are no costs or sales in that column):

Profit	Predict Class 0	Predict Class 1
Actual 0	0	– $20
Actual 1	0	$80

Looked at purely in terms of costs, when everyone is classified as a 0, there are no costs of sending the offer; the only costs are the opportunity costs of failing to make sales to the ten 1's = $100. The cost (actual costs of sending the offer, plus the opportunity costs of missed sales) of using the data mining routine to select people to send the offer to is only $48, as follows:

Costs	Predict Class 0	Predict Class 1
Actual 0	0	$20
Actual 1	$20	$8

However, this does not improve the actual classifications themselves. A better method is to change the classification rules (and hence the misclassification rates), as discussed in the preceding section, to reflect the asymmetric costs.

Reprinted for WRFSM/I300634, SAP
John Wiley & Sons (US), John Wiley & Sons, Inc. (c) 2010, Copying Prohibited

A popular performance measure that includes costs is the *average misclassification cost*, which measures the average cost of misclassification per classified observation. Denote by q_0 the cost of misclassifying a class 0 observation (as belonging to class 1) and by q_1 the cost of misclassifying a class 1 observation (as belonging to class 0). The average misclassification cost is

$$\frac{q_0 n_{0,1} + q_1 n_{1,0}}{n}.$$

Thus, we are looking for a classifier that minimizes this quantity. This can be computed, for instance, for different cutoff values.

It turns out that the optimal parameters are affected by the misclassification costs only through the ratio of these costs. This can be seen if we write the foregoing measure slightly differently:

$$\frac{q_0 n_{0,1} + q_1 n_{1,0}}{n} = \frac{n_{0,1}}{n_{0,0} + n_{0,1}} \frac{n_{0,0} + n_{0,1}}{n} q_0 + \frac{n_{1,0}}{n_{1,0} + n_{1,1}} \frac{n_{1,0} + n_{1,1}}{n} q_1.$$

Minimizing this expression is equivalent to minimizing the same expression divided by a constant. If we divide by q_0, it can be seen clearly that the minimization depends only on q_1/q_0 and not on their individual values. This is very practical because in many cases it is difficult to assess the cost associated with misclassifying a 0 member and that associated with misclassifying a 1 member, but estimating the ratio is easier.

This expression is a reasonable estimate of future misclassification cost if the proportions of classes 0 and 1 in the sample data are similar to the proportions of classes 0 and 1 that are expected in the future. If instead of a random sample, we draw a sample such that one class is oversampled (as described in the next section), then the sample proportions of0's and 1's will be distorted compared to the future or population. We can then correct the average misclassification cost measure for the distorted sample proportions by incorporating estimates of the true proportions (from external data or domain knowledge), denoted by $p(C_0)$ and $p(C_1)$, into the formula:

$$\frac{n_{0,1}}{n_{0,0} + n_{0,1}} p(C_0)\, q_0 + \frac{n_{1,0}}{n_{1,0} + n_{1,1}} p(C_1)\, q_1.$$

Using the same logic as above, it can be shown that optimizing this quantity depends on the costs only through their ratio (q_1/q_0) and on the prior probabilities only through their ratio [$p(C_0)/p(C_1)$]. This is why software packages that incorporate costs and prior probabilities might prompt the user for ratios rather than actual costs and probabilities.

Generalization to More Than Two Classes All the comments made above about two-class classifiers extend readily to classification into more than two classes. Let us suppose that we have m classes $C_0, C_1, C_2, \ldots, C_{m-1}$. The classification matrix has m rows and m columns. The misclassification cost associated with the diagonal cells is, of course, always zero. Incorporating prior probabilities of the various classes (where now we have m such numbers) is still done in the same manner. However, evaluating misclassification costs becomes much more complicated: For an m-class case we have $m(m-1)$ types of misclassifications. Constructing a matrix of misclassification costs thus becomes prohibitively complicated.

A lift chart cannot be used with a multiclass classifier, unless a single "important class" is defined, and the classifications are reduced to "important" and "unimportant" classes.

Lift Charts Incorporating Costs and Benefits When the benefits and costs of correct and incorrect classification are known or can be estimated, the lift chart is still a useful presentation and decision tool. As before, a classifier is needed that assigns to each record a probability that it belongs to a particular class. The procedure is then as follows:

1. Sort the records in order of predicted probability of success (where *success* = belonging to the class of interest).
2. For each record, record the cost (benefit) associated with the actual outcome.
3. For the highest probability (i.e., first) record, the value in step 2 is the y coordinate of the first point on the lift chart. The x coordinate is index number 1.

Reprinted for WRFSM/I300634, SAP John Wiley & Sons (US), John Wiley & Sons, Inc. (c) 2010, Copying Prohibited

4. For the next record, again calculate the cost (benefit) associated with the actual outcome. Add this to the cost (benefit) for the previous record. This sum is the *y* coordinate of the second point on the lift curve. The *x* coordinate is index number 2.

5. Repeat step 4 until all records have been examined. Connect all the points, and this is the lift curve.

6. The reference line is a straight line from the origin to the point *y* = total net benefit and *x* = *N*(*N* = number of records).

Note It is entirely possible for a reference line that incorporates costs and benefits to have a negative slope if the net value for the entire dataset is negative. For example, if the cost of mailing to a person is $0.65, the value of a responder is $25, and the overall response rate is 2%, the expected net value of mailing to a list of 10, 000 is (0.02 3 $25 3 10, 000) - ($0.65 × 10, 000) = $5000 - $6500 = -$1500. Hence, the *y* value at the far right of the lift curve (*x* = 10, 000) is −1500, and the slope of the reference line from the origin will be negative. The optimal point will be where the lift curve is at a maximum (i.e., mailing to about 3000 people) in Figure 5.9.

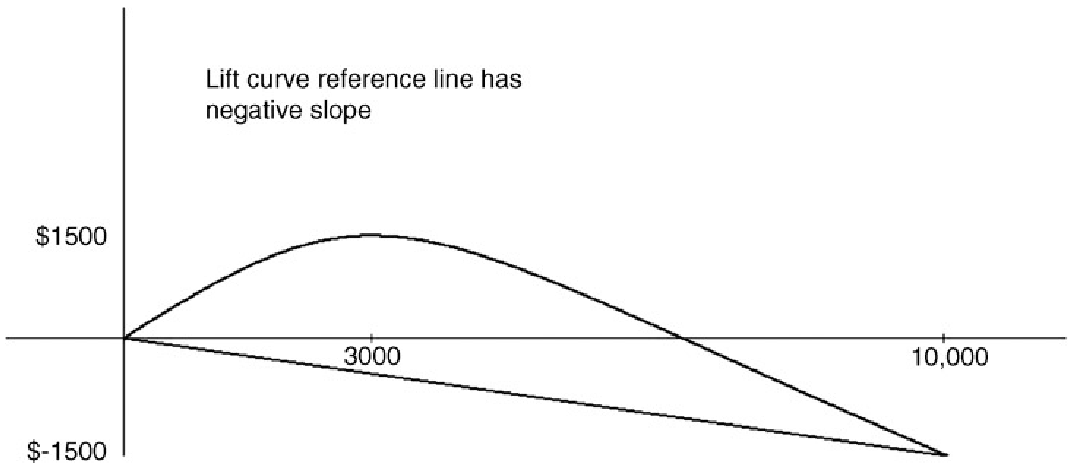


FIGURE 5.9: LIFT CURVE INCORPORATING COSTS

Lift as Function of Cutoff We could also plot the lift as a function of the cutoff value. The only difference is the scale on the *x* axis. When the goal is to select the top records based on a certain budget, the lift versus number of records is preferable. In contrast, when the goal is to find a cutoff that distinguishes well between the two classes, the lift versus cutoff value is more useful.

Oversampling and Asymmetric Costs

As we saw briefly in Chapter 2, when classes are present in very unequal proportions, simple random sampling may produce too few of the rare class to yield useful information about what distinguishes them from the dominant class. In such cases, stratified sampling is often used to oversample the cases from the more rare class and improve the performance of classifiers. It is often the case that the more rare events are the more interesting or important ones: responders to a mailing, those who commit fraud, defaulters on debt, and the like.

In all discussions of *oversampling* (also called *weighted sampling)*, we assume the common situation in which there are two classes, one of much greater interest than the other. Data with more than two classes do not lend themselves to this procedure.

Consider the data in Figure 5.10, where × represents nonresponders, and O, responders. The two axes correspond to two predictors. The dashed vertical line does the best job of classification under the assumption of equal costs: It results in just one misclassification (one O is misclassified as an ×). If we incorporate more realistic misclassification costs—let us say that failing to catch an O is five times as costly as failing to catch a 3—the costs of misclassification jump to 5. In such a case, a horizontal line as shown in Figure 5.11, does a better job: It results in misclassification costs of just 2.

John Wiley & Sons (US), John Wiley & Sons, Inc. (c) 2010, Copying Prohibited

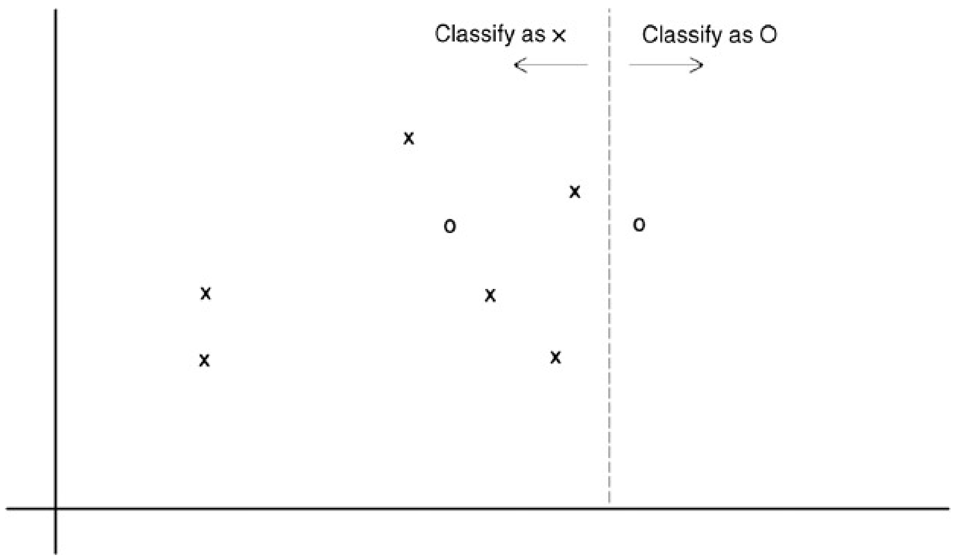


FIGURE 5.10: CLASSIFICATION ASSUMING EQUAL COSTS OF MISCLASSIFICATION

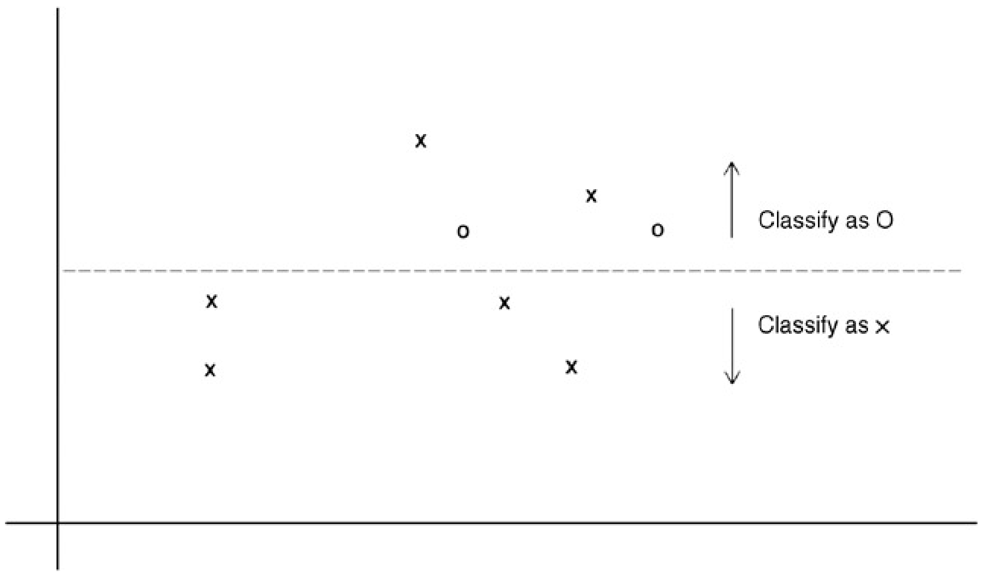


FIGURE 5.11: CLASSIFICATION ASSUMING UNEQUAL COSTS OF MISCLASSIFICATION

Oversampling is one way of incorporating these costs into the training process. In Figure 5.12, we can see that classification algorithms would automatically determine the appropriate classification line if four additional O's were present at each existing O. We can achieve appropriate results either by taking five times as many o's as we would get from simple random sampling (by sampling with replacement if necessary), or by replicating the existing o's fourfold.

John Wiley & Sons (US), John Wiley & Sons, Inc. (c) 2010, Copying Prohibited

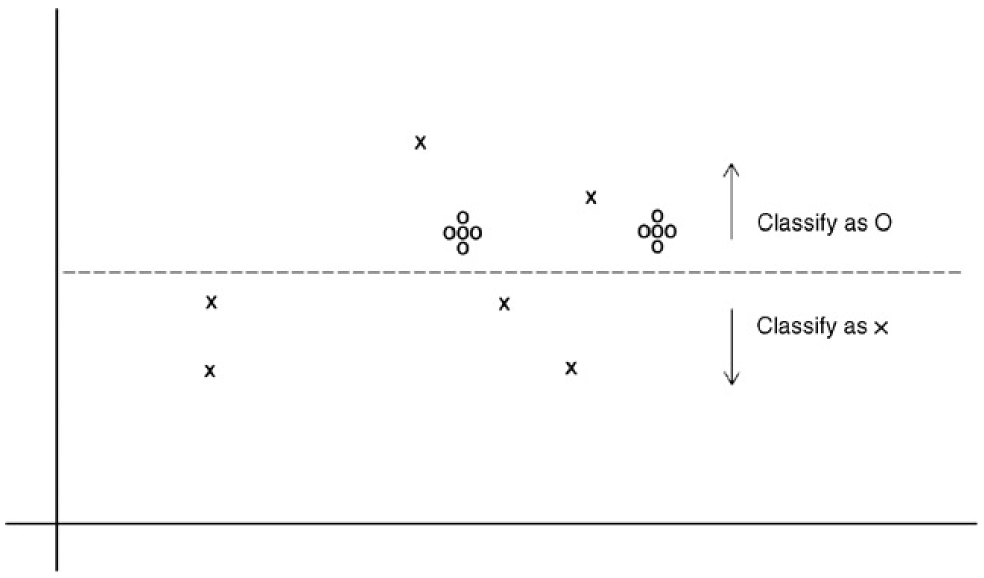


FIGURE 5.12: CLASSIFICATION USING OVERSAMPLING TO ACCOUNT FOR UNEQUAL COSTS

Oversampling without replacement in accord with the ratio of costs (the first option above) is the optimal solution but may not always be practical. There may not be an adequate number of responders to assure that there will be enough nonresponders to fit a model if the latter constitutes only a small proportion of the former. Also, it is often the case that our interest in discovering responders is known to be much greater than our interest in discovering nonresponders, but the exact ratio of costs is difficult to determine. When faced with very low response rates in a classification problem, practitioners often sample equal numbers of responders and nonresponders as a relatively effective and convenient approach. Whatever approach is used, when it comes time to assess and predict model performance, we will need to adjust for the oversampling in one of two ways:

1. Score the model to a validation set that has been selected without over-sampling (i.e., via simple random sampling).
2. Score the model to an oversampled validation set, and reweight the results to remove the effects of oversampling.

The first method is more straightforward and easier to implement. We describe how to oversample and how to evaluate performance for each of the two methods.

When classifying data with very low response rates, practitioners typically:

- Train models on data that are 50% responder, 50% nonresponder.
- Validate the models with an unweighted (simple random) sample from the original data.

Oversampling the Training Set How is weighted sampling done? One common procedure, where responders are sufficiently scarce that you will want to use all of them, follows:

1. First, the response and nonresponse data are separated into two distinct sets, or *strata*.
2. Records are then randomly selected for the training set from each stratum. Typically, one might select halfthe (scarce) responders for the training set, then an equal number of nonresponders.
3. The remaining responders are put in the validation set.
4. Nonresponders are randomly selected for the validation set in sufficient numbers to maintain the original ratio of responders to nonresponders.
5. If a test set is required, it can be taken randomly from the validation set.

XLMiner has a utility for this purpose.

Evaluating Model Performance Using a Nonoversampled Validation Set Although the oversampled data can be used to train models, they are often not suitable for predicting model performance because the number of responders will (of

Reprinted for WRFSM/I300634, SAP John Wiley & Sons (US), John Wiley & Sons, Inc. (c) 2010, Copying Prohibited

course) be exaggerated. The most straightforward way of gaining an unbiased estimate of model performance is to apply the model to regular data (i.e., data not oversampled). To recap: Train the model on oversampled data, but validate it with regular data.

Evaluating Model Performance If Only Oversampled Validation Set Exists In some cases, very low response rates may make it more practical to use oversampled data not only for the training data, but also for the validation data. This might happen, for example, if an analyst is given a data sample for exploration and prototyping, and it is more convenient to transfer and work with a smaller dataset in which a sizable proportion of cases are those with the rare response (typically the response of interest). In such cases it is still possible to assess how well the model will do with real data, but this requires the oversampled validation set to be reweighted, in order to restore the class of observations that were underrepresented in the sampling process. This adjustment should be made to the classification matrix and to the lift chart in order to derive good accuracy measures. These adjustments are described next.

I. Adjusting the Confusion Matrix for Oversampling Let us say that the response rate in the data as a whole is 2%, and that the data were oversampled, yielding a sample in which the response rate is 25 times as great = 50%. Assume that the validation classification matrix looks like this:

	CLASSIFICATION MATRIX, OVERSAMPLED DATA (VALIDATION)		
	Predicted 0	**Predicted 1**	**Total**
Actual 0	390	110	500
Actual 1	80	420	500
Total	470	530	1000

At this point, the (inaccurate) misclassification rate appears to be (80 + 110)/1000 = 19%, and the model ends up classifying 53% of the records as 1's.

There were 500 (actual) 1's in the sample and 500 (actual) 0's. If we had not oversampled, there would have been far fewer 1's. Put another way, there would be many more 0's for each 1. So we can either take away 1's or add 0's to reweight the sample. The calculations for the latter are shown: We need to add enough 0's so that the 1's constitute only 2% of the total, and the 0's, 98% (where *X* is the total):

$$500 + 0.98X = X.$$

Solving for *X*, we find that *X* = 25, 000.

The total is 25, 000, so the number of 0's is (0.98)(25, 000) = 24, 500. We can now redraw the classification matrix by augmenting the number of (actual) nonresponders, assigning them to the appropriate cells in the same ratio in which they appear in the classification table above (3.545 predicted 0's for every predicted 1):

	CLASSIFICATION MATRIX, REWEIGHTED		
	Predicted 0	**Predicted 1**	**Total**
Actual 0	19, 110	5, 390	24, 500
Actual 1	80	420	500
Total	19, 190	5, 810	25, 000

The adjusted misclassification rate is (80 + 5390)/25, 000 = 21.9%, and the model ends up classifying 5810/25, 000 of the records as 1's, or 21.4%.

II. Adjusting the Lift Curve for Oversampling The lift curve is likely to be a more useful measure in low-response situations, where our interest lies not so much in classifying all the records correctly as in finding a model that guides us toward those records most likely to contain the response of interest (under the assumption that scarce resources preclude examining or contacting all the records). Typically, our interest in such a case is in maximizing value or minimizing cost, so we will show the adjustment process incorporating the cost-benefit element. The following procedure can be used (and easily implemented in Excel):

1. Sort the validation records in order of the predicted probability of success (where success = belonging to the class of interest).

Reprinted for WRFSM/I300634, SAP John Wiley & Sons (US), John Wiley & Sons, Inc. (c) 2010, Copying Prohibited

2. For each record, record the cost (benefit) associated with the actual outcome.
3. Multiply that value by the proportion of the original data having this outcome; this is the adjusted value.
4. For the highest probability (i.e., first) record, the value above is the *y* coordinate of the first point on the lift chart. The *x* coordinate is index number 1.
5. For the next record, again calculate the adjusted value associated with the actual outcome. Add this to the adjusted cost (benefit) for the previous record. This sum is the *y* coordinate of the second point on the lift curve. The *x* coordinate is index number 2.
6. Repeat step 5 until all records have been examined. Connect all the points, and this is the lift curve.
7. The reference line is a straight line from the origin to the point *y* = total net benefit and *x* = *N*(*N* = number of records).

Classification Using a Triage Strategy

In some cases it is useful to have a "cannot say" option for the classifier. In a two-class situation, this means that for a case, we can make one of three predictions: The case belongs to C_0, or the case belongs to C_1, or we cannot make a prediction because there is not enough information to pick C_0 or C_1 confidently. Cases that the classifier cannot classify are subjected to closer scrutiny either by using expert judgment or by enriching the set of predictor variables by gathering additional information that is perhaps more difficult or expensive to obtain. This is analogous to the strategy of triage, which is often employed during retreat in battle. The wounded are classified into those who are well enough to retreat, those who are too ill to retreat even if treated medically under the prevailing conditions, and those who are likely to become well enough to retreat if given medical attention. An example is in processing credit card transactions, where a classifier may be used to identify clearly legitimate cases and obviously fraudulent ones while referring the remaining cases to a human decision maker who may look up a database to form a judgment. Since the vast majority of transactions are legitimate, such a classifier would substantially reduce the burden on human experts.

5.3 Evaluating Predictive Performance

When the response variable is continuous, the evaluation of model performance is slightly different from the categorical response case. First, let us emphasize that predictive accuracy is not the same as goodness of fit. Classical measures of performance are aimed at finding a model that fits the data well, whereas in data mining we are interested in models that have high predictive accuracy. Measures such as R^2 and standard error of estimate are very popular strength of fit measures in classical regression modeling, and residual analysis is used to gauge goodness of fit where the goal is to find the best fit for the data. However, these measures do not tell us much about the ability of the model to predict new cases.

For prediction performance, there are several measures that are used to assess the predictive accuracy of a regression model. In all cases, the measures are based on the validation set, which serves as a more objective ground than the training set to assess predictive accuracy. This is because records in the validation set are not used to select predictors or to estimate the model coefficients. Measures of accuracy use the prediction error that results from predicting the validation data with the model (that was trained on the training data).

Benchmark: The Average

Recall that the benchmark criterion in prediction is using the average outcome (thereby ignoring all predictor information). In other words, the prediction for a new record is simply the average outcome of the records in the training set. A good predictive model should outperform the benchmark criterion in terms of predictive accuracy.

Prediction Accuracy Measures

The prediction error for record *i* is defined as the difference between its actual *y* value and its predicted *y* value: $e_i = y_i - \hat{y}i$. A few popular numerical measures of predictive accuracy are:

- *MAE* or *MAD* (mean absolute error/deviation) = $= 1/n \sum_{i=1}^{n} |e_i|$. This gives the magnitude of the average absolute error.
- *Average error* = $= 1/n \sum_{i=1}^{n} e_i$. This measure is similar to MAD except that it retains the sign of the errors, so that

John Wiley & Sons (US), John Wiley & Sons, Inc. (c) 2010, Copying Prohibited

negative errors cancel out positive errors of the same magnitude. It therefore gives an indication of whether the predictions are on average over- or underpredicting the response.

- *MAPE* (mean absolute percentage error) = $= 100\% \times 1/n \sum_{i=1}^{n} |e_i/y_i|$. This measure gives a percentage score of how predictions deviate (on average) from the actual values.
- *RRMSE* (root-mean-squared error) = $= \sqrt{1/n \sum_{i=1}^{n} e_i^2}$. This is similar to the standard error of estimate, except that it is computed on the validation data rather than on the training data. It has the same units as the variable predicted.
- Total *SSE* (total sum of squared errors) = $= \sum_{i=1}^{n} e_i^2$.

Such measures can be used to compare models and to assess their degree of prediction accuracy. Notice that all these measures are influenced by outliers. To check outlier influence, we can compute median-based measures (and compare to the mean-based measures) or simply plot a histogram or boxplot of the errors. It is important to note that a model with high predictive accuracy might not coincide with a model that fits the training data best.

Finally, a graphical way to assess predictive performance is through a lift chart. This compares the model's predictive performance to a baseline model that has no predictors. Predictions from the baseline model are simply the average $\bar{y}$. A lift chart for a continuous response is relevant only when we are searching for a set of records that gives the highest cumulative predicted values.

To illustrate this, consider a car rental firm that renews its fleet regularly so that customers drive late-model cars. This entails disposing of a large quantity of used vehicles on a continuing basis. Since the firm is not primarily in the used car sales business, it tries to dispose of as much of its fleet as possible through volume sales to used car dealers. However, it is profitable to sell a limited number of cars through its own channels. Its volume deals with the used car dealers leave it flexibility to pick and choose which cars to sell in this fashion, so it would like to have a model for selecting cars for resale through its own channels. Since all cars were purchased some time ago and the deals with the used car dealers are for fixed prices (specifying a given number of cars of a certain make and model class), the cars' costs are now irrelevant and the dealer is interested only in maximizing revenue. This is done by selecting for its own resale the cars likely to generate the most revenue. The lift chart in this case gives the predicted lift for revenue.

Figure 5.13 shows a lift chart based on fitting a linear regression model to a dataset that includes the car prices (y) and a set of predictor variables that describe a car's features (mileage, color, etc.) The lift chart is based on the validation data of400 cars. It can be seen that the model's predictive performance is better than the baseline model since its lift curve is higher than that of the baseline model. The lift (and decile-wise) charts in Figure 5.13 would be useful in the following scenario: Choosing the top 10% of the cars that gave the highest predicted sales, for example, we would gain 1.7 times the amount compared to choosing 10% of the cars at random. This can be seen from the decile chart (Figure 5.13). This number can also be computed from the lift chart by comparing the sales predicted for 40 random cars (the value of the baseline curve at *x* = 40), which is $486, 871 (= the sum of the predictions of the 400 validation set cars divided by 10) with the sales of the 40 cars that have the highest predicted values by the model (the value of the lift curve at *x* = 40), $835, 883. The ratio between these numbers is 1.7.

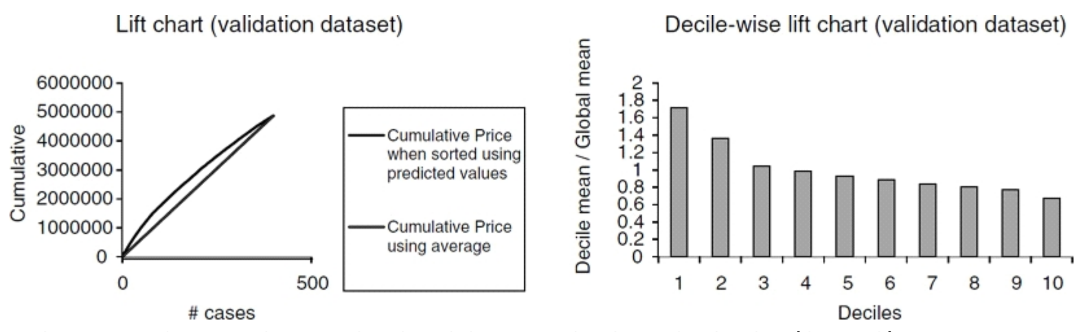


FIGURE 5.13: LIFT CHARTS FOR CONTINUOUS RESPONSE (SALES)

Problems

John Wiley & Sons (US), John Wiley & Sons, Inc. (c) 2010, Copying Prohibited

5.1 A data mining routine has been applied to a transaction dataset and has classified 88 records as fraudulent (30 correctly so) and 952 as nonfraudulent (920 correctly so). Construct the classification matrix and calculate the error rate.

5.2 Suppose that this routine has an adjustable cutoff (threshold) mechanism by which you can alter the proportion of records classified as fraudulent. Describe how moving the cutoff up or down would affect the following:

a. The classification error rate for records that are truly fraudulent

b. The classification error rate for records that are truly nonfraudulent

5.3 Consider Figure 5.14, the decile-wise lift chart for the transaction data model, applied to new data

a. Interpret the meaning of the first and second bars from the left.

b. Explain how you might use this information in practice.

c. Another analyst comments that you could improve the accuracy of the model by classifying everything as nonfraudulent. Ifyou do that, what is the error rate?

d. Comment on the usefulness, in this situation, of these two metrics of model performance (error rate and lift).

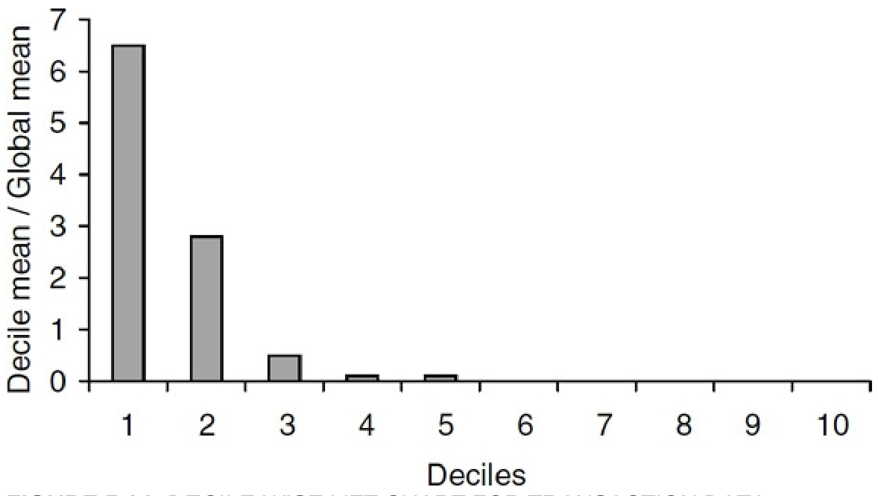


FIGURE 5.14: DECILE-WISE LIFT CHART FOR TRANSACTION DATA

5.4 A large number of insurance records are to be examined to develop a model for predicting fraudulent claims. Of the claims in the historical database, 1% were judged to be fraudulent. A sample is taken to develop a model, and oversampling is used to provide a balanced sample in light of the very low response rate. When applied to this sample *(N* = 800), the model ends up correctly classifying 310 frauds, and 270 nonfrauds. It missed 90 frauds, and classified 130 records incorrectly as frauds when they were not.

a. Produce the classification matrix for the sample as it stands.

b. Find the adjusted misclassification rate (adjusting for the oversampling).

c. What percentage of new records would you expect to be classified as fraudulent?

Reprinted for WRFSM/I300634, SAP John Wiley & Sons (US), John Wiley & Sons, Inc. (c) 2010, Copying Prohibited