

Chapter 4

An Excel-Based Data Mining Tool

Chapter Objectives

- ▶ Understand the structure of the iData Analyzer.
- ▶ Recognize how predictability and predictiveness scores help determine categorical attribute significance.
- ▶ Understand how attribute mean and standard deviation scores are used to compute numerical attribute significance.
- ▶ Know how to use ESX for building supervised learner models.
- ▶ Know how to perform unsupervised clustering with ESX.
- ▶ Know how to create production rules with RuleMaker.
- ▶ Understand how instance typicality is computed.

In this chapter we introduce the iData Analyzer (iDA) and show you how to use two of the learner models contained in your iDA software suite of data mining tools. Section 4.1 overviews the iDA Model for Knowledge Discovery. In Section 4.2 we introduce ESX, an exemplar-based data mining tool capable of both supervised learning and unsupervised clustering. In Section 4.3 we present the standard way of representing datasets for all iDA data mining tools. Section 4.4 shows you how to use ESX to perform unsupervised clustering. You will also learn how to generate production rules with RuleMaker. In Section 4.5 we show you how to build supervised learner models with ESX. Section 4.6 details RuleMaker's rule generation options. Section 4.7 introduces the concept of instance typicality. Section 4.8 offers additional information about features of the ESX learner model. The end-of-chapter exercises help you better understand how ESX and RuleMaker can create generalized models from data.

4.1 The iData Analyzer

The iData Analyzer (iDA) provides support for the business or technical analyst by offering a visual learning environment, an integrated tool set, and data mining process support. iDA consists of a preprocessor, three data mining tools, and a report generator. As iDA is an Excel add-on, the user interface is Microsoft Excel. Figure 4.1 shows the component parts of iDA. The following is a brief description of each component:

- **Preprocessor.** Before the data in a file is presented to one of the iDA mining engines, the file is scanned for several types of errors, including illegal numeric values, blank lines, and missing items. The preprocessor corrects several types of errors but does not attempt to fix numerical data errors. The preprocessor outputs a data file ready for data mining as well as a document informing us about the nature and location of unresolved errors.
- **Heuristic agent.** The heuristic agent responds to the presentation of data files containing several thousand instances. The heuristic agent allows us to decide if we wish to extract a representative subset of the data for analysis or if we desire to process the entire dataset.
- **ESX.** This component is an exemplar-based data mining tool that builds a concept hierarchy to generalize data.
- **Neural networks.** iDA contains two neural network architectures: a backpropagation neural network for supervised learning and a self-organizing feature map for unsupervised clustering. These neural network architectures are the topic of Chapter 9.

- **RuleMaker.** iDA's production rule generator provides several rule generating options.
- **Report generator.** The report generator offers several sheets of summary information for each data mining session. The contents of the individual sheets created by the report generator are detailed in this chapter.

As a final point, a major concern with neural network architectures is their lack of explanation about what has been discovered. Fortunately, a primary strength seen with the ESX learner model is its ability to output useful explanations about patterns within the data. Figure 4.1 illustrates that we can use ESX to help explain the output of an iDA neural network data mining session.

Installing iDA

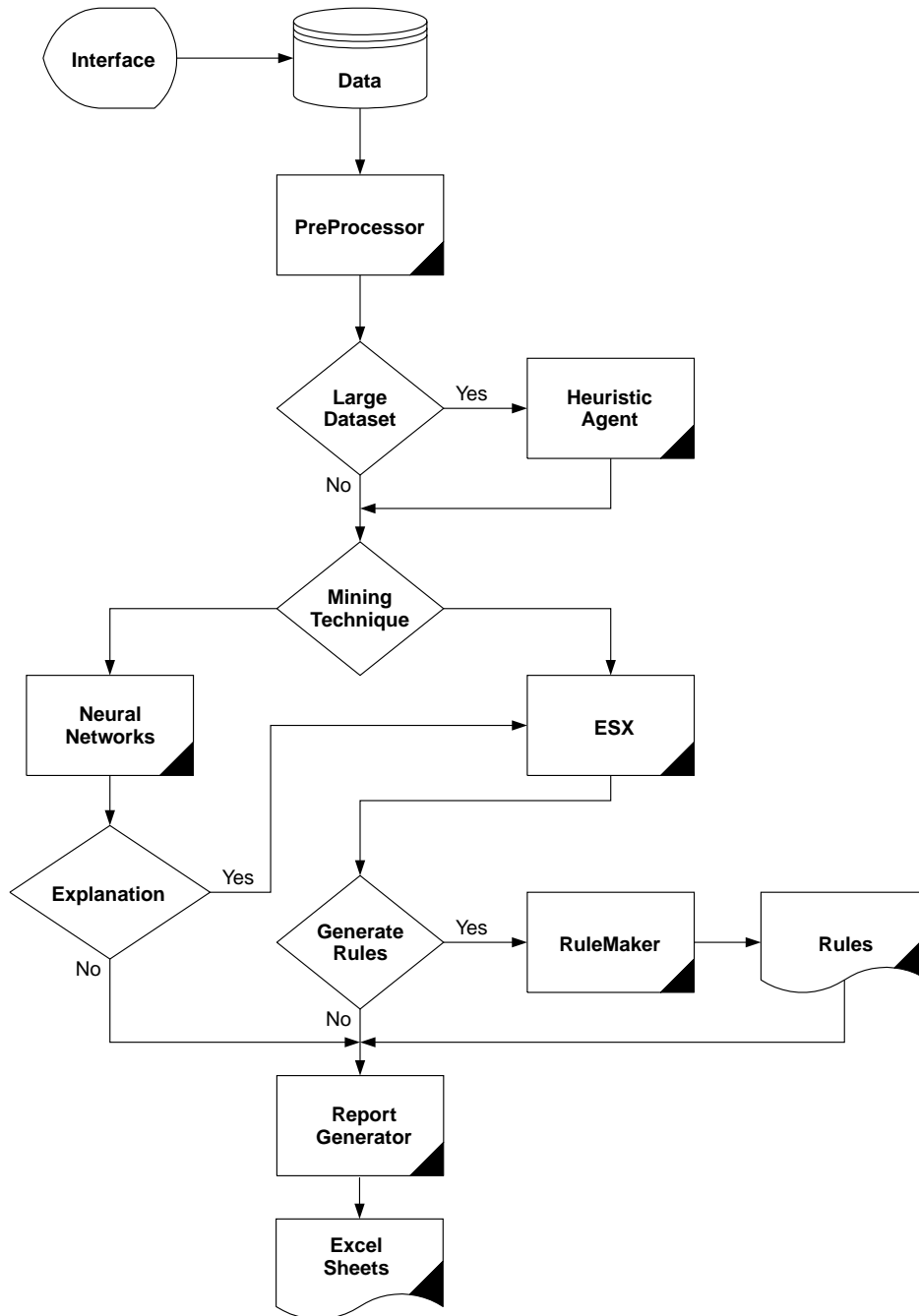
In this section we show you how to install iDA. The software should install correctly with all current versions of MS Excel. However, if you are using MS Office 2000 or Office XP, the macro security level of your implementation may be set at *high*. If this is the case, the software will not install. Perform the following steps to check and possibly reset the security level:

- Open Excel and select *Tools*.
- Mouse to *Macro* and then select *Security*.
- If the security setting is *high*, change the setting to *medium* and click *OK*.
- Exit Excel and close all other applications.

The steps for installing iDA are as follows:

- Insert the CD that comes with the text into your CD drive.
- Mouse to *Start* and left-click on *Run*.
- Specify your CD drive and type *installiDA.exe*. For example, if your CD drive is D, type: *d:\installiDA.exe*.
- Press *Enter*.
- Select *Next*, read the license agreement, and select *Agree*.
- Select *Finish*, then *OK*.
- Answer *Yes* if asked "Would you like to enable macros?"

Figure 4.1 • The iDA system architecture

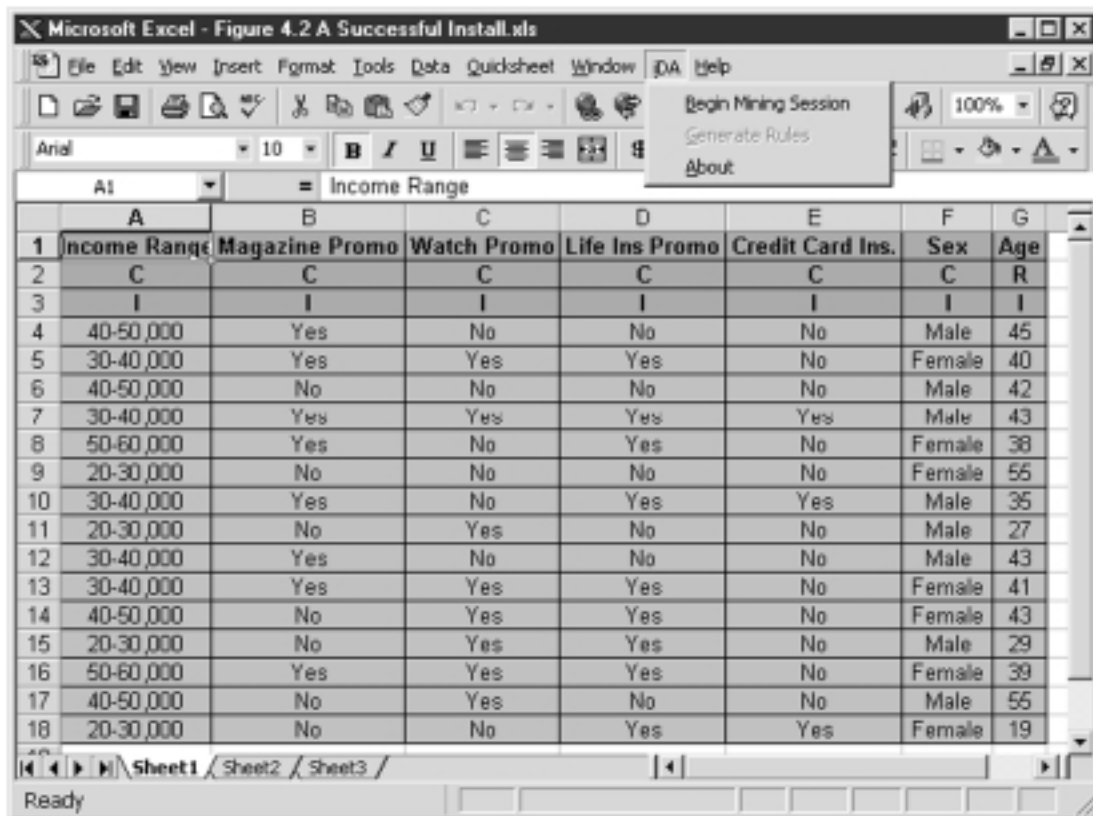


- Upon completion, a ReadMe file will appear on your screen. Once you close the file the installation is complete.

A successful installation adds the iDA drop-down menu item to the MS Excel spreadsheet environment, as shown in Figure 4.2.

The iDA neural networks are written in the Java programming language. Therefore to run the neural network software you must have Java installed on your machine. Java is free software, and the installation file for Java is contained on your CD. The instructions for installing Java are given in Appendix A. If you need to install Java, you can do so now or you can wait until you are ready to study Chapter 9. If you have any other problems with the installation, refer to Appendix A for additional assistance.

Figure 4.2 • A successful installation



Limitations

The commercial version of iDA is bound by the size of a single MS Excel spreadsheet, which allows a maximum of 65,536 rows and 256 columns. The iDA input format uses the first three rows of a spreadsheet to house information about individual attributes. Therefore a maximum of 65,533 data instances in attribute-value format can be mined. The student version that comes with your text allows a maximum of 7000 data instances (7003 rows).

As each MS Excel column holds a single attribute, the maximum number of attributes allowed is 256. The maximum size of an attribute name or attribute value is 250 characters. Also, RuleMaker will not generate rules for more than 20 classes. Although not required, it is best to close all other applications while you use the iDA software suite of tools. This is especially true for applications containing more than a few hundred data instances.

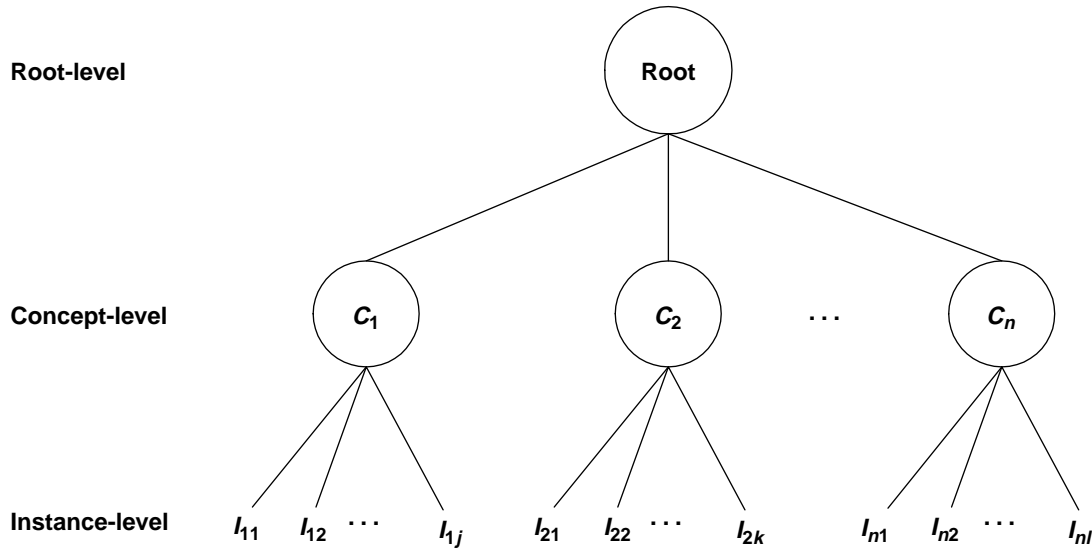
4.2 ESX: A Multipurpose Tool for Data Mining

ESX can help create target data, find irregularities in data, perform data mining, and offer insight about the practical value of discovered knowledge. The following is a partial list of features seen with the ESX learner model:

- It supports both supervised learning and unsupervised clustering.
- It does not make statistical assumptions about the nature of data to be processed.
- It supports an automated method for dealing with missing attribute values.
- It can be applied in domains containing both categorical and numerical data.
- It can point out inconsistencies and unusual values in data.
- For supervised classification, ESX can determine those instances and attributes best able to classify new instances of unknown origin.
- For unsupervised clustering, ESX incorporates a globally optimizing evaluation function that encourages a best instance clustering.

The primary data structure used by ESX is a three-level concept hierarchy. Figure 4.3 shows the general form of this tree structure. The nodes at the instance level of the tree represent the individual instances that define the concept classes given at the concept level. The concept-level nodes store summary statistics about the attribute values found within their respective instance-level children. The root-level tree node stores summary information about all instances within the domain. Concept- and root-level

Figure 4.3 • An ESX concept hierarchy



summary information is given to the report generator, which in turn outputs a summary report in spreadsheet format.

Class resemblance scores are stored within the root node and each concept-level node. Class resemblance scores form the basis of ESX's evaluation function. Class resemblance scores provide a measure of overall similarity for the exemplars making up individual concept classes. As ESX is commercial software, details about the class resemblance computation are not available. However, we can state the evaluation rule ESX uses for both supervised learning and unsupervised clustering.

1. Given:
 - a. A set of existing concept-level nodes C_1, C_2, \dots, C_n .
 - b. An average class resemblance score S , computed by summing the resemblance scores for each class C_1, C_2, \dots, C_n and dividing by n .
 - c. A new instance I to be classified.
2. Make I an instance of the concept node that results in the largest average increase or smallest average decrease for S .
3. When learning is unsupervised:

If a better score is achieved by creating a new concept node, then create new node C_{n+1} with I as its only member.

That is, for each existing concept-level node C_k , a new instance to be classified is temporarily placed in the concept class represented by the node. A new class resemblance score is computed for C_k , as well as a new average class resemblance score. The winning concept class is the one that creates the largest increase (or smallest decrease) in average class resemblance. When learning is unsupervised, a score for creating a new concept-level node is also computed. As you can see, unlike the K-Means algorithm, we are not required to make a determination about the total number of clusters to be formed.

So much for theory! Let's work through a few simple examples so you can learn how the features of ESX help you create your own data models.

4.3 iDAV Format for Data Mining

ESX can build both unsupervised and supervised learner models. For the first example we use ESX and unsupervised clustering. Our example is based on the fictitious data defined in Chapter 2 about individuals who own credit cards issued by the Acme Credit Card Company. The data is displayed in Table 4.1 in *iDAV format* (iDA attribute-value format). This is the format required by all iDA data mining tools. Let's begin by taking a closer look at the structure of iDAV formatted files.

Setting up a data file in iDAV format is easy! The columns in the first row contain attribute names. Although Excel allows us to place multiple lines of data in a single cell, iDA does not. Be sure to reformat any data file cells with multiple lines before attempting a data mining session.

Table 4.1 shows a *C* or an *R* in each column of the second row. We place a *C* in a second-row column if the corresponding attribute data type is categorical (nominal). We place an *R* in a second-row column if the entered data is real-valued (numerical). Integers, decimal values, numbers formatted with scientific notation, as well as values containing dollar signs (\$57.30) or percent symbols (57.3%) are all valid numeric data items.

Categorical data includes all data not represented by numbers. We can sometimes treat numbers as categorical data. As a general rule, if there are but a few differing attribute values, we can and should treat numerical data as categorical. For example, an attribute representing the total number of household vehicles for individual families should be considered categorical.

The third row of Table 4.1 informs ESX about attribute usage. Table 4.2 displays the possibilities. To perform an unsupervised clustering we place an *I* in the third-row columns of attributes we wish to have ESX use as input attributes. If learning is supervised, exactly one attribute must contain an *O* in its third-row column. With ESX, an output attribute must be categorical. Categorical attributes having several unique values are of little predictive value and should be marked

Table 4.1 • Credit Card Promotion Database: iDAV Format

Income Range	Magazine Promotion	Watch Promotion	Life Insurance Promotion	Credit Card Insurance	Sex	Age
C	C	C	C	C	C	R
I	I	I	I	I	I	I
40–50K	Yes	No	No	No	Male	45
30–40K	Yes	Yes	Yes	No	Female	40
40–50K	No	No	No	No	Male	42
30–40K	Yes	Yes	Yes	Yes	Male	43
50–60K	Yes	No	Yes	No	Female	38
20–30K	No	No	No	No	Female	55
30–40K	Yes	No	Yes	Yes	Male	35
20–30K	No	Yes	No	No	Male	27
30–40K	Yes	No	No	No	Male	43
30–40K	Yes	Yes	Yes	No	Female	41
40–50K	No	Yes	Yes	No	Female	43
20–30K	No	Yes	Yes	No	Male	29
50–60K	Yes	Yes	Yes	No	Female	39
40–50K	No	Yes	No	No	Male	55
20–30K	No	No	Yes	Yes	Female	19

with a *U*. Attributes falling into this category include *last name*, *sequence number*, and *birthdate*.

Starting with the fourth row, we enter actual data. Each new row contains one data instance. There are two cautions when entering data to be mined:

- Each data item, whether it is an attribute name, value, or data type, must be entered in a single cell on one line. The preprocessor will flag any row containing multiple lines as an error.
- The preprocessor flags any data instance containing an illegal numeric value as an error. This is true even if the attribute is declared as unused. To be safe, place a *C* in the second-row column corresponding to an unused numeric attribute that is suspect of containing illegal values. In this way, the preprocessor will think the attribute is categorical and ignore any illegal numeric characters.



Table 4.2 • Values for Attribute Usage

Character	Usage
I	The attribute is used as an input attribute.
U	The attribute is not used.
D	The attribute is not used for classification or clustering, but attribute value summary information is displayed in all output reports.
O	The attribute is used as an output attribute. For supervised learning with ESX, exactly one categorical attribute is selected as the output attribute.

The next section offers an example using a five-step approach for unsupervised clustering with ESX. The example is written as a tutorial. We encourage you to use your iDA software to work through the example with us.

4.4 A Five-Step Approach for Unsupervised Clustering

Here is a five-step procedure for performing unsupervised clustering with ESX:

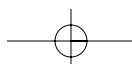
1. Enter data to be mined into a new Excel spreadsheet.
2. Perform a data mining session.
3. Read and interpret summary results.
4. Read and interpret results for individual clusters.
5. Visualize and interpret rules defining the individual clusters.

Let's apply the five-step approach to the credit card promotion database shown in Table 4.1!

Applying the Five-Step Approach for Unsupervised Clustering

STEP 1: ENTER THE DATA TO BE MINED

Two forms of the credit card promotion database are located in the iDA samples directory—CreditCardPromotion.xls and CreditCardPromotionNet.xls. We are interested in the first version of the dataset. The second version has been transformed by mapping categorical attribute values to numeric equivalents. We will use the second version of the dataset when we study neural networks in Chapters 8 and 9. Here's how to open Excel and load the CreditCardPromotion dataset:





4.4 • A Five-Step Approach for Unsupervised Clustering

115

1. Open Microsoft Windows Explorer.
2. Double-click on the *iDA* directory.
3. Double-click on the *Samples* directory.
4. Double-click on the file *CreditCardPromotion.xls*.

As an alternative, you may choose to mouse to *Start*, then to *Run* and type `c:\iDA\Samples\CreditCardPromotion.xls`. In either case, your spreadsheet should look like the one shown in Figure 4.4.

STEP 2: PERFORM A DATA MINING SESSION

Before mining the data make sure all attribute usage values show an *I*. The procedure to perform the data mining session is as follows:

1. Select the *iDA* menu item from the menu bar at the top of your MS Excel workbook.
2. Select *Begin Mining Session* from the drop-down list. If this does not start a data mining session, click on any cell and try a second time.

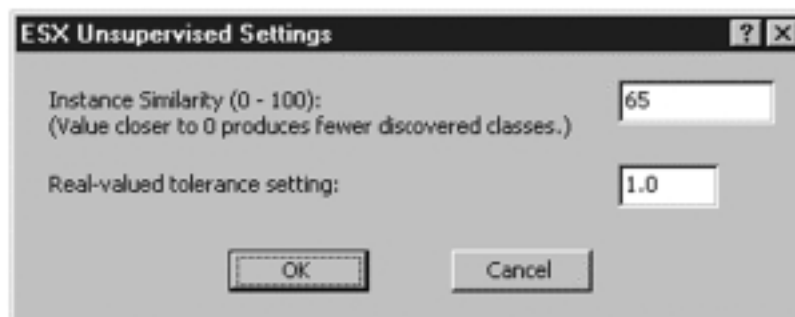
Figure 4.4 • The credit card promotion database in MS Excel

	A	B	C	D	E	F	G
1	Income Range	Magazine Promo	Watch Promo	Life Ins Promo	Credit Card Ins.	Sex	Age
2	C	C	C	C	C	C	R
3	I	I	I	I	I	I	I
4	40-50,000	Yes	No	No	No	Male	45
5	30-40,000	Yes	Yes	Yes	No	Female	40
6	40-50,000	No	No	No	No	Male	42
7	30-40,000	Yes	Yes	Yes	Yes	Male	43
8	50-60,000	Yes	No	Yes	No	Female	38
9	20-30,000	No	No	No	No	Female	55
10	30-40,000	Yes	No	Yes	Yes	Male	35
11	20-30,000	No	Yes	No	No	Male	27
12	30-40,000	Yes	No	No	No	Male	43
13	30-40,000	Yes	Yes	Yes	No	Female	41
14	40-50,000	No	Yes	Yes	No	Female	43
15	20-30,000	No	Yes	Yes	No	Male	29
16	50-60,000	Yes	Yes	Yes	No	Female	39
17	40-50,000	No	Yes	No	No	Male	55
18	20-30,000	No	No	Yes	Yes	Female	19

3. When the registration information appears, enter your name. Enter student as your company name. The registration code is given on your installation CD. Enter the registration code exactly as it appears on the CD. Click *OK*. You need only enter this information the first time you use iDA.
4. A message box appears asking you to select an unsupervised learner model. Select *ESX* and click *OK*.
5. You will then see the message box shown in Figure 4.5. You are asked to enter two values:
 - The value for instance similarity encourages or discourages the creation of new clusters. A value closer to *100* encourages the formation of new clusters. A value closer to *0* favors new instances to enter existing classes. The default value is *65*. Notice that with *ESX* the terms *class* and *cluster* are used interchangeably.
 - The real-valued tolerance setting helps determine the similarity criteria for real-valued attributes. A setting of *1.0* is usually appropriate. However, there are exceptions to this rule.

For our example, use the defaults for both parameters and click *OK*.
6. Next you will see a message box indicating that eight clusters were formed. This tells you the data has been successfully mined. The box asks if we wish to generate the output report. As a general rule, an unsupervised clustering of more than five or six clusters is likely to be less than optimal. There are always exceptions. However, let's apply this rule and click *No*.
7. A message box reading "Dataset not mined" will appear. Click *OK*.
8. Let's try again. Repeat steps 1–4. For step 5, set the similarity value to *55*, leave the real-tolerance value at *1.0*, and click *OK*.

Figure 4.5 • **Unsupervised settings for ESX**





4.4 • A Five-Step Approach for Unsupervised Clustering

117

9. A message appears indicating three classes have been formed. Click *Yes* to generate the output report.
10. After a minute or two you will be presented with the message box shown in Figure 4.6. The message box asks you to choose settings for the rule generator. You can also bypass rule generation. For our example, we will generate rules. Set the minimum rule coverage at *30*, use the default settings for all other parameters, and click *OK*.

Once rule and report generation is complete you will see the output of the rule generator as displayed in Figure 4.7. By clicking inside the production rules window you can scroll the window and examine the rules for each cluster. Notice that the clusters have been respectively named: *Class 1*, *Class 2*, and *Class 3*. We examine each sheet of the output report in detail in the sections that follow.

STEP 3: READ AND INTERPRET SUMMARY RESULTS

The output of an unsupervised mining session appears in four new sheets. Sheet names are based on the name of the original sheet (in our case, Sheet1) as well as on their purpose. You can move from one sheet to the next by clicking on the sheet name given in the bottom tray of the current spreadsheet. Any sheet not appearing in the tray can be summoned by clicking on the left or right arrows found in the left-bottom portion of the spreadsheet. The description box titled

Figure 4.6 • RuleMaker options

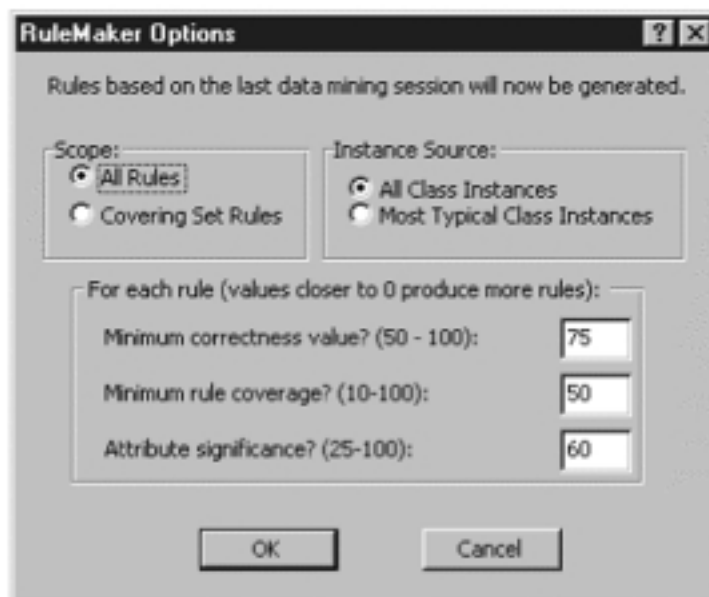


Figure 4.7 • Rules for the credit card promotion database

Rules for Class 1		Scope:	
3 instances		Instance Source:	All Rules
-----		Minimum Correctness:	0.75
45.00 <= Age <= 45.00	rule accuracy 100.00%	Percent Covered:	0.3
	rule coverage 33.33%	Attribute Significance:	0.6
42.00 <= Age <= 42.00	rule accuracy 100.00%		
	rule coverage 33.33%		
**Total Percent Coverage = 66.67%			

Rules for Class 2			
5 instances			

Output Reports: Unsupervised Clustering offers a brief explanation of the contents found within each sheet. We begin by examining the overall summary results of the data mining session.

To display the summary results on your screen left-click on the tray sheet labeled *Sheet1 RES SUM*. The first part of the output for the result summary sheet is displayed in Figure 4.8. We first examine the class resemblance statistics.

Class Resemblance Statistics

The *Class Resemblance Statistics* show us that ESX has partitioned the 15 instances into 3 clusters denoted as *Class 1*, *Class 2*, and *Class 3*. Row three lists the resemblance score (*Res. Score*) for each class as well as the resemblance score for the entire domain of instances. The class resemblance score offers a first indication about how well the instances within each cluster fit together. This score is a similarity value and should not be interpreted as a probability. Notice that the instances of Class 1 have a best within-class fit.

4.4 • A Five-Step Approach for Unsupervised Clustering

119

Figure 4.8 • Summary statistics for the Acme credit card promotion database

CLASS RESEMBLANCE STATISTICS				
	Class 1	Class 2	Class 3	Domain
Res. Score:	0.81	0.529	0.66	0.46
No. of Inst.:	3	5	7	15
Cluster Quality:	0.75	0.14	0.43	
DOMAIN STATISTICS FOR CATEGORICAL ATTRIBUTES				
Number of Classes:	3			
Domain Res. Score:	0.46			
Categorical Attribute Summary:				
	Name	Value	Frequency	Predictability
	Income Range	"40-50,000"	4	0.27
		"20-30,000"	4	0.27
		"30-40,000"	5	0.33
		"50-60,000"	2	0.13
	Magazine Proms	Yes	8	0.53
		No	7	0.47
	Watch Proms	No	7	0.47
		Yes	8	0.53
	Life Ins Promo	No	6	0.40
		Yes	9	0.60
	Credit Card Ins.	No	12	0.80
		Yes	3	0.20

The **domain resemblance** represents the overall similarity of all instances within the dataset. As a rule, it is important to see within-class resemblance scores that are higher than the domain resemblance value. This rule need not be true in all cases, but should apply to the larger classes. When learning is unsupervised, if within-class similarity scores are not generally higher than the domain similarity score, there are at least three possibilities:

1. The chosen attributes do a poor job of representing the instances. One way to test this is to choose other attributes and see if the results improve.
2. The instances chosen for the unsupervised clustering are not representative of typical instances found within the domain.
3. The domain does not contain definable classes and is not appropriate for unsupervised clustering.

Whether learning is supervised or unsupervised, poor class resemblance scores will likely result in poor model performance.

Finally, we see **cluster quality** scores listed directly below the instance count for each class. These values are percentages. The cluster quality for an individual class is simply the percent increase (or decrease) of the class resemblance score relative to the domain resemblance. Higher class resemblance scores relative to the domain resemblance result in higher values for cluster quality.

Domain Statistics for Categorical Attributes

Returning to Figure 4.8, the next section of the RES SUM sheet is titled *Domain Statistics for Categorical Attributes*. You will need to use the scroll bar to see all categorical attribute values. By examining the domain statistics for categorical data, we can locate duplicate categorical attribute names and values. For example, the attribute *income range* might be referenced in some instances as “Income Range” and in other instances as “income range.” In addition, low frequency counts can help identify incorrect categorical attribute values.

We can make several initial deductions about categorical attributes by examining attribute-value *predictability scores*. Predictability scores are simple probabilities, but they can be confusing. Attribute-value predictability scores can be computed relative to the entire domain of instances. They can also be computed for each class or cluster found within a dataset. First we investigate **domain predictability**.

Given categorical attribute A with values $v_1, v_2, v_3 \dots v_n$, the domain predictability of v_i tells us the percent of domain instances showing v_i as the value for A .

To illustrate, here are two potentially interesting domain predictability scores from Figure 4.8.

- Eighty percent of all credit card holders do not have credit card insurance.
- Sixty percent of all card holders took advantage of the life insurance promotion.

Output Reports: Unsupervised Clustering

Sheet1 RES SUM: This sheet contains summary statistics about attribute values and offers several heuristics to help us determine the quality of a data mining session.

Sheet1 RES CLS: This sheet has information about the clusters formed as a result of an unsupervised mining session.

Sheet1 RUL TYP: Instances are listed by their cluster number. The last column of this sheet shows a typicality value for each instance. The typicality of instance I is the average similarity of I to the other members of its cluster.

Sheet1 RES RUL: The production rules generated for each cluster are contained in this sheet.

4.4 • A Five-Step Approach for Unsupervised Clustering

A predictability score near 100% for a domain-level categorical attribute value indicates that the attribute is not likely to be useful for supervised learning or unsupervised clustering. This is true because a majority of instances will have an identical value for the attribute. However, if we are specifically looking for outliers within the dataset, we must exercise caution when eliminating attributes displaying large attribute-value predictability scores.

Domain Statistics for Numerical Attributes

Moving down the RES SUM sheet, we see the section labeled *Domain Statistics for Numerical Attributes*. Scroll the RES SUM sheet until your screen is similar to the output displayed in Figure 4.9. The summary provides useful information about class mean and standard deviation scores. The *attribute significance* value measures the predictive value of each numerical attribute. The computation of attribute significance is best illustrated by example. Here's how we compute the attribute significance score for attribute *age*.

- Subtract the smallest class mean (*Class 2 age* = 37.00) from the largest mean value (*Class 1 age* = 43.33).
- Divide this result by the domain standard deviation (sd = 9.51).

The result gives a final attribute significance of 0.67. Dividing by the standard deviation normalizes mean difference values, thus allowing us to compare the attribute significance scores for all numeric attributes. Numeric attributes with lower significance values (usually less than 0.25) will likely be of little value in differentiating one class from another.

Commonly Occurring Categorical Attributes

The final output for the RES SUM sheet displays the most commonly occurring categorical attribute values found within each cluster. Figure 4.9 shows the most commonly occurring categorical attribute summary for our application. This information offers initial insight about which categorical attributes are best able to differentiate the individual clusters.

STEP 4: READ AND INTERPRET INDIVIDUAL CLASS RESULTS

Sheet1 RES CLS displays statistics about the individual classes. We often find valuable knowledge within the class summary sheet that cannot be seen in the rules generated for each class. We limit our discussion to the output for Class 3.

Figure 4.10 shows a portion of the summary results for Class 3. To open this sheet click on *Sheet1 RES CLS* and scroll down until your spreadsheet matches the output shown in Figure 4.10. The initial information found within the RES CLS sheet repeats facts about the number of class instances and the class resemblance score. You will also see attribute values for the two most and two least typical class instances. **Typicality** is defined as the average similarity of an instance to all other members of its cluster or class. Scroll to the right if the typicality scores are not

Figure 4.9 • **Statistics for numerical attributes and common categorical attribute values**

The screenshot shows an Excel spreadsheet with the following data:

DOMAIN STATISTICS FOR NUMERICAL ATTRIBUTES					
	Class 1	Class 2	Class 3	Domain	Attribute Significance
Age (mean)	43.33	37.00	39.86	39.60	0.67
Age (std)	1.53	16.85	2.85	9.51	
MOST COMMONLY OCCURRING CATEGORICAL ATTRIBUTE VALUES					
	Class 1	Class 2	Class 3		
Income Range	"40-50,000"	"20-30,000"	"30-40,000"		
Magazine Promo	Yes	No	Yes		
Watch Promo	No	Yes	Yes		
Life Ins Promo	No	No	Yes		
Credit Card Ins.	No	No	No		
Sex	Male	Male	Female		

displayed on your spreadsheet. As you can see, the typicality score for each of the most typical instances is 0.76. Examining the most and least typical class instances gives us a first impression about the structure of the class.

Class Predictability and Predictiveness

Class predictiveness and predictability scores shown in Figure 4.10 for Class 3 merit detailed examination. Let's first consider the attribute *income range*. **Class predictability** can be defined as follows:

Given categorical attribute A with values $v_1, v_2, v_3, \dots, v_n$, the class C predictability score for v_i tells us the percent of instances within class C showing v_i as the value for A . Class predictability is a within-class measure. The sum of the predictability scores for attribute A with values $v_1, v_2, v_3, \dots, v_n$ within class C is always equal to 1.

Class predictability tells us the percent of class instances having a particular value for a categorical attribute. To illustrate, Figure 4.10 shows the class predictability score for *income range* = 30–40K as 0.57. This predictability score informs us that 57% of all Class 3 instances make between \$30,000 and \$40,000 per year. Class predictability scores measure the degree of *necessity* for class membership.

4.4 • A Five-Step Approach for Unsupervised Clustering

123

Figure 4.10 • Class 3 summary results

Class	3							
Total Number of Instances:	7							
Class Reemblance Score:	0.66							
Most Typical Instance:	Income Range	Magazine Promo	Watch Promo	Life Ins Promo	Credit Card Ins.	Sex	Age	Typicality
	"30-40,000"	Yes	Yes	Yes	No	Female	40	0.76
	"30-40,000"	Yes	Yes	Yes	No	Female	41	0.76
Least Typical Instance:	Income Range	Magazine Promo	Watch Promo	Life Ins Promo	Credit Card Ins.	Sex	Age	Typicality
	"40-50,000"	No	Yes	Yes	No	Female	45	0.57
	"40-50,000"	Yes	No	Yes	Yes	Male	36	0.55
Categorical Attribute Summary:	Name	Value	Frequency	Predictability	Predictiveness			
	Income Range	"30-40,000"	4	0.57	0.60			
		"50-60,000"	2	0.29	1.00			
		"40-50,000"	1	0.14	0.26			
	Magazine Promo	Yes	6	0.86	0.75			
		No	1	0.14	0.14			
	Watch Promo	Yes	5	0.71	0.63			
		No	2	0.29	0.29			

Class predictiveness scores are more difficult to understand. An attribute-value predictiveness score is defined as the probability an instance resides in a specified class given the instance has the value for the chosen attribute. A formal definition for **class predictiveness** follows:

Given class C and attribute A with values $v_1, v_2, v_3, \dots, v_n$, an attribute-value predictiveness score for v_j is defined as the probability an instance resides in C given the instance has value v_j for A . Predictiveness scores are between-class measures. The sum of the predictiveness scores for categorical attribute A with value v_j is always equal to 1.

To better understand class predictiveness, consider the attribute *income range* with value 50–60K. Figure 4.10 shows that only 29% of all Class 3 instances have this value for income range. However, the predictiveness score of 1.00 tells us that all instances with *income range* = 50–60K reside in Class 3. We say that *income range* = 50–60K is a *sufficient* condition for Class 3 membership. Therefore if someone tells us that the value of *income range* for a particular instance is 50–60K, we can predict with 100% certainty that the instance is from Class 3. Notice that *life insurance promotion* = yes has a predictability score of 1.00 and a predictiveness score of 0.78, making *life insurance promotion* = yes a best defining attribute value for

Class 3. Here are four useful observations relating to class predictability and predictiveness scores. For a given concept class *C*:

1. If an attribute value has a predictability and predictiveness score of 1.0, the attribute value is said to be *necessary and sufficient* for membership in *C*. That is, all instances within class *C* have the specified value for the attribute and all instances with this value for the attribute reside in class *C*.
2. If an attribute value has a predictiveness score of 1.0 and a predictability score less than 1.0, we conclude that all instances with the value for the attribute reside in class *C*. However, there are some instances in *C* that do not have the value for the attribute in question. We call the attribute value *sufficient but not necessary* for class membership.
3. If an attribute value has a predictability score of 1.0 and a predictiveness score less than 1.0, we conclude that all instances in class *C* have the same value for the chosen attribute. However, some instances outside of class *C* also have this same value for the given attribute. The attribute is said to be *necessary but not sufficient* for class membership.

In general, any categorical attribute with at least one highly predictive value should be designated as an input attribute. Also, a categorical attribute with little predictive value ought to be flagged as unused or display-only.

Necessary and Sufficient Attribute Values

The report generator lists attribute values with predictiveness scores greater than or equal to 0.80 as highly sufficient. Likewise, attribute values showing a predictability greater than or equal to 0.80 are listed as necessary attributes. Figure 4.11 presents a summary of necessary and sufficient attribute-value information for Class 3. Scroll the current RES CLS spreadsheet until you see the information displayed in the figure. As you can see, there are no categorical attribute values necessary and sufficient for Class 3 membership. Also, income range values 30–40K and 50–60K are highly sufficient for membership in Class 3. Finally, *magazine promotion = yes* and *life insurance promotion = yes* are highly necessary Class 3 attribute values.

STEP 5: VISUALIZE INDIVIDUAL CLASS RULES

In Chapter 2 you were introduced to RuleMaker, the production rule generator that comes with your iDA software package. In this section we review RuleMaker's output format by examining the production rules generated for Class 3. In Section 4.6 we will take a closer look at how to make optimal use of RuleMaker's features.

To examine the rules generated for Class 3, click on *Sheet1 RES RUL*. Figure 4.7 displays the initial output for this sheet. Click inside the rules window and scroll until you encounter the list of rules for Class 3. Here is an interesting Class 3 rule with one precondition:

4.4 • A Five-Step Approach for Unsupervised Clustering

Figure 4.11 • Necessary and sufficient attribute values for Class 3

	A	B	C	D
115	Attribute Values Necessary and Sufficient for Class Membership:	<u>Name</u>	<u>Value</u>	
116				
117	Attribute Values Highly Sufficient for Class Membership:	<u>Name</u>	<u>Value</u>	
118		Income Range	"30-40,000"	
119		Income Range	"50-60,000"	
120				
121	Attribute Values Highly Necessary for Class Membership:	<u>Name</u>	<u>Value</u>	
122		Magazine Promo	Yes	
123		Life Ins Promo	Yes	
124				
125	Numerical Value Attribute Summary:	<u>Name</u>	<u>Mean</u>	<u>Standard Deviation</u>
126		Age	39.857	2.854
127				
128				

Life Ins Promo = Yes
 :rule accuracy 77.78%
 :rule coverage 100.00%

Notice the rule format is a variation of what we saw in Chapter 2. Each rule simply declares the precondition(s) necessary for an instance to be covered by the rule. The abbreviated rule format is preferred, as RuleMaker is designed to generate a wealth of rules for each concept class. An explicit form of the same rule is:

IF *Life Ins Promo = Yes*
THEN *Class = 3*
 :rule accuracy 77.78%
 :rule coverage 100.00%

Rule accuracy tells us the rule is accurate in 77.78% of all cases where it applies. That is, the rule will cause a misclassification with 22.22% of the training set instances. Rule coverage

informs us that the rule applies to 100% of the Class 3 instances. In other words, everyone in Class 3 accepted the life insurance promotion.

Rarely are single conditionals able to cover a significant number of class instances. For this reason, RuleMaker is able to generate rules with multiple conditions. Rules with multiple conditions are interesting because they help uncover possible dependencies between the attributes. The following is a Class 3 rule with multiple requirements for class membership:

```
35.00 <= Age <=43.00  
and Life Insurance Promo = Yes  
:rule accuracy 100.00%  
:rule coverage 100.00%
```

This rule shows a relationship between the values of the attributes *age* and *life insurance promotion*. The rule covers all Class 3 instances and is 100% correct. For a real application, this result is strong supportive evidence for targeting the 35-to-43 age group for any new life insurance promotional offerings.

4.5 A Six-Step Approach for Supervised Learning

We now turn our attention toward supervised learning with ESX. For our example we once again employ the hypothetical database about credit card promotions. To follow our discussion, open the CreditCardPromotion.xls file and copy the dataset to a new spreadsheet file. Here is a six-step procedure for supervised learning with ESX.

1. Enter data to be mined into an Excel spreadsheet and choose an output attribute.
2. Perform a data mining session.
3. Read and interpret summary results.
4. Read and interpret test set results.
5. Read and interpret results for individual classes.
6. Visualize and interpret class rules.

Let's work through the credit card promotion example using the six-step approach just described.



Applying the Six-Step Approach for Supervised Learning

STEP 1: CHOOSE AN OUTPUT ATTRIBUTE

The output attribute forms the basis for supervised classification. Our understanding of the data and what we are trying to achieve determines the choice of an output attribute. Let's follow the theme of Chapters 2 and 3 and assume we are about to launch a fresh life insurance promotion. Therefore our goal is to build a model able to predict customers likely to take advantage of the new promotional offering. Since we want our model to be applicable to new card holders, we limit the input attribute selection to *income range*, *credit card insurance*, *sex*, and *age*.

STEP 2: PERFORM THE MINING SESSION

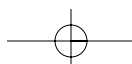
To begin, change the *I* in the third-row column for *life insurance promotion* to an *O*. Also, change the *I* in the third-row column for *magazine promotion* as well as *watch promotion* to *D*, indicating these will be display-only attributes. Here is the procedure for mining the data.

1. From the dropdown iDA menu, click on *Begin Mining Session*.
2. Click *OK* when your registration information appears.
3. A message box will appear asking you to choose a supervised learner model. Select *ESX* and click *OK*.
4. You will see a message box asking you to select the number of instances for training as well as a real-valued tolerance setting. Select *10* instances for training and *1.0* for the real-valued tolerance. Click *OK*.
5. Finally, use the default settings for the rule generator and click *OK*.

The output of a supervised mining session appears in either four or six sheets. The description box titled *Output Reports: Supervised Learning* gives a brief description of the output for each sheet. The contents of four of these sheets are similar to those described for unsupervised clustering. The two additional sheets are created only when test data are present. We will examine the contents of the new sheets in step 4. We begin our analysis of the data with summary results. To open the summary results sheet, click on *Sheet1 RES SUM* located in the bottom tray of your spreadsheet.

STEP 3: READ AND INTERPRET SUMMARY RESULTS

The output summary sheet gives us results based on the training data. As a first test, we examine the class resemblance scores for each class. The highest class resemblance score is seen with the class represented by *life insurance promotion = yes* (0.575). The class for *life insurance promotion = no* displays a resemblance score of 0.525, and the domain resemblance



shows 0.48. As the class resemblance scores are higher than the domain resemblance, we have some indication that differences exist between the two classes. *Domain Statistics for Categorical Attributes* tell us that 80% of the training instances represent individuals without credit card insurance.

Domain Statistics for Numerical Attributes show attribute *age* with a significance score of 0.42. Finally, upon examining the *Most Commonly Occurring Categorical Attribute Values* for both classes, we see differences between the values for all categorical attributes except *credit card insurance*.

STEP 4: READ AND INTERPRET TEST SET RESULTS

To determine test set model performance, locate and open the sheet labeled *Sheet1 RES TST*. Finding the RES TST sheet may involve manipulating the arrows in the lower left of the Excel spreadsheet. This sheet lists each test set instance together with its computed class.

The RES TST output sheet for our experiment is shown in Figure 4.12. The last two columns are of interest. The last column offers the choices made by the model. The second-to-last column displays the correct classification for each instance. A star is placed to the right of those test instances correctly classified by the model. Notice three of the five test set instances have been correctly classified.

Next, find and open the sheet labeled *Sheet1 RES MTX*. This sheet displays the confusion matrix for the test data. RES MTX shows that three test set instances were correctly classified. Also, two instances from the class represented by *life insurance promotion = yes* were incorrectly classified as unlikely candidates for the promotional offer. Finally, the RES MTX sheet

Output Reports: Supervised Learning

Sheet1 RES MTX: This sheet shows the confusion matrix for test set data.

Sheet1 RES TST: This sheet contains individual test set instance classifications and is only seen when a test set is applied.

Sheet1 RES SUM: This sheet contains summary statistics about attribute values and offers several heuristics to help us determine the quality of a data mining session.

Sheet1 RES CLS: This sheet has information about the classes formed as a result of a supervised mining session.

Sheet1 RUL TYP: Instances are listed by their class name. The last column of this sheet shows a typicality value for each instance.

Sheet1 RES RUL: The production rules generated for each class are contained in this sheet.



4.5 • A Six-Step Approach for Supervised Learning

129

Figure 4.12 • Test set instance classification

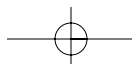
	A	B	C	D	E	F	G	H	I
1	Income Range	Magazine Promo	Watch Promo	Credit Card Ins.	Sex	Age	Life Ins Promo	computed class	
2	C	C	C	C	C	R	C		
3	I	D	D	I	I	I	O		
4	40-50,000	No	Yes	No	Female	43	Yes	No	
5	20-30,000	No	Yes	No	Male	29	Yes	No	
6	50-60,000	Yes	Yes	No	Female	39	Yes	Yes	*
7	40-50,000	No	Yes	No	Male	55	No	No	*
8	20-30,000	No	No	Yes	Female	19	Yes	Yes	*

gives the percent of test set instances classified correctly, as well as an upper and lower error bound for the test set classification. Notice that 60% of the test instances were correctly classified. The size of the test set must be relatively large ($n > 100$) for the upper and lower bound error rates to have meaning. For purposes of general explanation, the error bound gives us a confidence interval for the test set error rate. We offer a clear example of how the confidence interval is used in Section 4.8.

Lastly, a supervised model must be able to classify new instances of unknown origin. If we wish to classify instances whose classification is not known, we simply place the instances in the dataset as test set entries. As the instances are not really part of the test set per se, we leave the output attribute field blank. Once the data is mined the RES TST sheet will show the classification determined by the model for the unknown instances. Given that iDA denotes a missing field with an open square, the column specifying the correct class output will contain the open square symbol. In addition, since the unknown instance is without a predetermined class, the instance does not count as an incorrect classification.

STEP 5: READ AND INTERPRET RESULTS FOR INDIVIDUAL CLASSES

Section 4.4 showed you how to interpret individual class results with unsupervised clustering. The same information is relevant when learning is supervised. However, with supervised learning it is particularly important for us to examine both categorical attribute predictiveness scores and numeric attribute significance. In this way, when a wealth of available attributes exist, we can create new supervised models by employing only those attributes most predictive of class membership.



For our example the summary results for the two formed classes are given in the sheet labeled *Sheet1 RES CLS*. Notice that the attribute-value combination *credit card insurance = yes* as well as the attribute-value pair *income range = 50–60K* each show a predictiveness score of 1.00 for the class *life insurance promotion = yes*. Can you find any highly predictive categorical attribute-value combinations in the class *life insurance promotion = no*?

STEP 6: VISUALIZE AND INTERPRET CLASS RULES

Open the *Sheet1 RES RUL* sheet to visualize the rules for the two classes. By scrolling the rule window, we see that 100% of all instances from each class are covered by at least one rule. Although this may be an acceptable result, it often takes several iterations of rule generation parameter manipulation to create a satisfactory set of rules. For this reason, iDA has a rerule feature that allows us to generate a new rule set without repeating an entire mining session. Here is the procedure for generating a new set of rules.

1. Click on the *RUL TYP* sheet located in the lower tray of your Excel spreadsheet. Two MS Word document images appear on your screen. These documents contain information relevant to the rule generation process. There is no need to open either document.
2. Mouse to the *iDA Drop-Down Menu* and click on *Generate Rules*. Several windows appear and disappear as the rule generator prepares itself for another session.
3. When the rule dialog box appears, modify the rule generating parameters as desired and click *OK*.

To try this, follow the procedure and change the *Scope* setting from *All Rules* to *Covering Set Rules*. Also change the *Minimum Rule Coverage* parameter to *30*. When rule generation is complete, you will see two rules for the first listed class and one rule for the second class. Here is the first rule for class *life insurance promotion = no*:

```
Income Range = 40–50K
:rule accuracy 100.00%
# covered = 2
# remaining = 3
```

Notice the rule format has changed in that we are told the number of class instances covered by the rule and the number of instances remaining to be covered. In this way, instead of obtaining a list of all possible rules satisfying the parameter settings, we get a best set of covering rules.

The *Covering Set Rules* algorithm uses the following procedure to generate a set of rules.

1. Construct a best-covering rule for the current set of class instances.
2. Eliminate all class instances covered by the rule generated in step 1.



4.5 • A Six-Step Approach for Supervised Learning

3. If additional class instances remain to be covered, and the minimum rule coverage criterion can be satisfied, repeat step 1.

Notice that each additional rule generation gives a new rule sheet. The first rerule will show new rules in *Sheet1 RUL TYP 2 RUL*. The second rerule will create *Sheet1 RUL TYP 3 RUL*. The pattern continues with each new set of rules. After several rule generations, it is best to delete previous rule sets. Simply right-click the bottom tray sheet you wish to delete and left-click *Delete*.

Before we look more closely at the options offered by the rule generator, a word of caution about the interpretation of rule accuracy values is merited. Although useful, the accuracy score is optimistic, as the score is computed based on training rather than test data. As of this writing, iDA does not support test set rule accuracy computations. The next section details the rule generation parameter settings and offers helpful hints when generating rules.

4.6 Techniques for Generating Rules

Many traditional rule generation programs follow a simple procedure for creating rules.

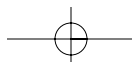
1. Choose an attribute that best differentiates all domain or subclass instances.
2. Use the attribute to further subdivide instances into classes.
3. For each subclass created in step 2:
 - If the instances in the subclass satisfy a predefined criteria, generate a defining rule for the subclass.
 - If the subclass does not satisfy the predefined criterion, repeat 1.

This algorithm works well for generating a best set of covering rules for a set of class instances. However, in a data mining environment we frequently wish to determine as many interesting class rules as possible. With RuleMaker we have this option. Alternatively, we can request RuleMaker to generate a best set of class covering rules. Let's look at all of the options available with RuleMaker.

Once a data mining session is complete, the dialog box shown in Figure 4.6 appears. We are asked to:

1. *Define the scope of the rules to be generated.*

If we select the *All Rules* radio button, RuleMaker will generate all rules for all classes that meet the criteria of the other parameters. If we select the *Covering Set Rules* radio button, RuleMaker will generate a set of best-defining rules for each class. With the covering set option, the rule coverage percent is replaced by two values: *#covered* and *#remaining*. In this way, we clearly see the covering effect each rule has on a set of class instances.



2. *Choose whether we wish to use all class instances or only the most typical instances for rule generation.*

If we choose the most typical instance option, we obtain a set of rules that best describe the most characteristic features of each class. However, some of the more atypical class instances may not be covered by the generated rules.

3. *Set a minimum correctness value.*

We must input a value between 50 and 100. If we enter 100, the only rules generated by RuleMaker will be rules that are 100% correct. If we enter the value 80, the rules generated must have an error rate less than or equal to 20%.

4. *Define a minimum percent of instances to be covered.*

We enter a value between 10 and 100. If we enter the value 10, RuleMaker will generate rules that cover 10% or more of the instances in each class. If we enter 80, only those rules covering 80% of the class instances will be covered.

5. *Choose an attribute significance value.*

Values close to 100 will allow RuleMaker to consider only those attribute values most highly predictive of class membership for rule generation. A good choice for this value is extremely important in domains containing more than a few attributes. As a general rule, in domains containing several attributes, we should start with a value between 80 and 90. If an interesting set of rules is not generated, we can always use the rerule feature and try lower attribute significance values.

Some helpful hints for generating rules:

1. If you spend time manipulating rule generating parameters you can often find a best set of class defining rules. As a first attempt, generate rules for a dataset by using the default parameters. If you are satisfied with these results you need not proceed further. However, if the rules appear to be less than optimal or if one or more classes are without rules you should alter the parameters and try again.
2. For categorical data, you can take advantage of attribute value predictability and predictiveness scores to help find rule sets. Suppose class resemblance scores indicate an interesting classification, but one or more of the formed classes is without a set of rules. For any class without rules and containing at least one categorical attribute, attribute-value predictiveness scores can be examined to determine a lower-bound setting for the minimum rule correctness value.
3. Generating a covering set of rules can be difficult at times. Frequently, we achieve a desired result by manipulating the minimum correctness, coverage, and attribute significance attributes. First, try setting the coverage and signifi-

cance parameters at their minimal values. If you are still unsuccessful, try decreasing the parameter value for classification correctness.

4.7 Instance Typicality

Whether learning is supervised or unsupervised, each instance has an associated typicality score. The typicality score for instance I represents the average similarity of I to all other instances within its class. We can observe the typicality scores for all instances by opening Sheet1 RUL TYP. This sheet gives us a listing of the data instances ordered by their typicality scores. With supervised learning the instances are limited to those found in the training data. Figure 4.13 shows the contents of the RUL TYP sheet for the supervised learning example just presented. Notice the sheet also contains two MS Word document icons. These documents hold instance and attribute information for the rule generator and are not of interest to us.

Let's examine the RUL TYP sheet more closely. The left-most column contains sequential instance numbers. The second column gives the class for each instance. The next few columns show the attribute values for the individual instances. The final column displays typicality scores. Typicality scores range between 0 and 1, with higher scores being more typical class instances. The instances are ordered within each class from most to least typical. The first listed instance for each class is considered the **class prototype** as it is a

Figure 4.13 • Instance typicality

Instance	Class	Income Range	Magazine	Promo	Watch	Promo	Life Ins	Promo	Credit Card	Sex	Age	Typicality
1	No	40-50,000	Yes	No	No	No	No	No	Male	45	0.625	
2	No	40-50,000	No	No	No	No	No	No	Male	42	0.625	
3	No	30-40,000	Yes	No	No	No	No	No	Male	43	0.605	
4	No	20-30,000	No	Yes	No	No	No	No	Male	27	0.5	
5	No	20-30,000	No	No	No	No	No	No	Female	55	0.3125	
6	Yes	30-40,000	Yes	Yes	Yes	Yes	No	No	Female	40	0.6875	
7	Yes	30-40,000	Yes	Yes	Yes	Yes	No	No	Female	41	0.6875	
8	Yes	50-60,000	Yes	No	Yes	Yes	No	No	Female	38	0.5	
9	Yes	30-40,000	Yes	No	Yes	Yes	Yes	Yes	Male	35	0.5	
10	Yes	30-40,000	Yes	Yes	Yes	Yes	Yes	Yes	Male	43	0.5	

best single representative of the class in general. The last few class instances are candidate *outliers* as they are atypical class members. The exposure of class outliers can frequently lead to a better understanding about the nature of the domain under study.

Here are some ways in which typicality scores can be used.

- Typicality scores can identify prototypical class instances as well as outliers.
- Instance typicality can help select a best set of instances for supervised classification.
- Typicality scores can be used to compute individual instance classification confidence values.

The end-of-chapter exercises help you learn more about how typicality is used to build supervised learner models. Here we take a closer look at how typicality can be employed to compute instance classification confidence scores.

As an example, assume we have built a supervised model to distinguish individuals likely to respond to a promotion from those who would not respond. We present our model with a set of 1,000,000 new instances to be classified. We have a budget allowing us to send the promotional package to 10,000 individuals.

Suppose our model classifies 20,000 of the instances as good candidates for the promotional offer. Given our restriction, we need a mechanism for choosing a best set of 10,000 individuals from the group of candidate instances. Several models associate confidence factors with individual rules, but few models allow us to easily compute test set confidence scores for individual instances. Fortunately, typicality scores allow us to associate confidence factors with new classifications. The general procedure for computing a classification confidence score for instance I is as follows:

1. Open the RES TST sheet to determine the classification given for unknown instance I .
2. Return to the original data sheet and replace the blank output attribute cell for instance I with the computed class.
3. Initiate a new supervised mining session, but this time include the previously unknown instance as part of the training data. By doing this, the instance will receive an associated typicality score.
4. Open the RESTYP sheet to see the typicality score associated with instance I .
5. Divide the instance I typicality score by the typicality score for the most typical class instance. This value represents the classification confidence score for instance I .

When you have time, test this procedure by adding one or more new unknown instances to the credit card promotion database.

4.8 Special Considerations and Features

This section offers information about making your data mining sessions more productive. We also provide a short discussion on erroneous and missing data.

Avoid Mining Delays

The data within an Excel spreadsheet can be mined several times. However, each mining session creates several other Excel spreadsheets. To avoid confusion each of these sheets are numbered. To illustrate, the first mining session will result in a sheet titled Sheet1 RES 1 SUM. If we mine the same data a second time, Sheet1 RES 2 SUM will be created. After a few mining sessions it is easy to get confused with the numbers as they increase. In addition, prior to each new mining session, Excel must save the contents of the current spreadsheets. Therefore at some point it is best to simply copy the original data into another Excel spreadsheet before starting a new mining session.

The Deer Hunter Dataset

The Deer Hunter dataset contains information about 6059 individual deer hunters who were asked whether they would have taken any hunting trips during 1991 if the total cost of their trips was a specified amount more than what they had paid for the current year. Although each instance contains only one increase amount, there are a total of 10 possible dollar-increase values ranging from as little as \$9.00 to as much as \$953.00. The original data was obtained from a survey conducted by the U.S. Fish and Wildlife Service. We obtained a cleaned form of the datafile from Dr. John Whitehead. The cleaned dataset contains 20 input attributes and 1 output attribute indicating whether the hunter responded posi-

tively to the aforementioned question.

The dataset is interesting because it represents real data that can be used to monitor and develop profiles of deer hunters. For example, states might use the results of discovered relationships in the data to help determine future changes in permit and licensing fees. The dataset is a difficult data mining application because of the variable amounts for total trip-increase values. See Appendix B for a complete description of the attributes in the dataset as well as more information about the original data. The dataset is found in the iDA samples directory under the title DeerHunter.xls. ■



The Quick Mine Feature

Datasets having thousands of instances may take several minutes to mine. If a dataset contains more than 2000 instances, the iDA quick mine feature allows us to significantly reduce the time it takes to mine the data. The quick mine feature is useful because the performance of a model built from a random subset of the data is likely to give us results similar to those seen with a model constructed from the entire dataset.

When learning is supervised and we have more than 2000 training set instances, iDA asks us if we wish to conduct a *quick mine* of the data. If we choose the quick mine feature, the ESX concept hierarchy is built using a random subset of 500 training set instances. The test set data remains unchanged. When learning is unsupervised and more than 2000 data instances are presented for clustering, ESX is given a random selection of 500 instances from which to cluster the data. We use the quick mine feature to:

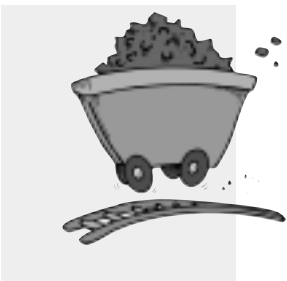
1. Obtain initial information about the predictive nature of the attributes within the domain.
2. Determine a setting for the similarity parameter (unsupervised learning).
3. Determine a value for the real-valued tolerance.
4. Determine settings for the rule generation parameters.

Two of the datasets that come with your software are quick mine candidates. The Deer Hunter dataset is an interesting dataset containing 6059 instances. Each instance of the dataset represents individual statistics gathered from a national survey conducted by the U.S. Fish and Wildlife Service (USFWS). The description box titled *The Deer Hunter Dataset* as well as a summary in Appendix B offers more details about this dataset.

The Titanic dataset is also a quick mine candidate. The dataset contains survival information for 2201 passengers and crew aboard the Titanic when it sank into the Atlantic ocean on April 15, 1912, while on its maiden voyage. The description box titled *The Titanic Dataset* as well Appendix B gives additional information about this dataset. Let's use the Titanic dataset to demonstrate iDA's quick mine feature.

Applying the Quick Mine Feature to the Titanic Dataset

1. Open Excel and load the Titanic.xls file.
2. Perform a supervised mining session with *class* as the output attribute. Use 2001 instances for training and the default real-tolerance setting.
3. When asked if you wish to perform a quick mine session, answer Yes.
4. Select *Cancel* when the rule generation menu is presented.



4.8 • *Special Considerations and Features*

137

As the training data is randomly selected, your results will be similar to but not identical to our findings. Please examine the contents of your RES SUM and RES MTX output sheets to compare your results with ours.

When the mining session is complete, check the RES SUM sheet for the total number of instances used for training. Our results show that 180 of the training data instances represent survivors of the tragedy and 320 of the training data instances did not survive. This gives us a 36% survivor rate. This is relatively close to the survivor rate of 32% seen within the entire population.

Finally, the test set confusion matrix is of interest because we can achieve a better understanding of the meaning associated with upper and lower bound error rates. By clicking on the RES MTX sheet, our results show that 99% of the test set instances were correctly classified. Stated another way, the model test set error rate is 1%. The test data is biased toward non-survivor instances, as only 5% of the test instances represent survivors. The upper and lower bound error values give us a 95% confidence limit for the test set error rate. Our results show an upper-bound error of 1.5% and a lower-bound error of 0.5%. Therefore, we can be 95% confident that the test set error rate of 1% is actually somewhere between 0.5% and 1.5%. Stated another way, given a pool of data instances for testing, if we perform this experiment with the same model by applying randomly selected test data from this pool 100 times, we will see a test set classification error rate somewhere between 0.5% and 1.5% in at least 95 of the 100 experiments. Although we cannot be sure of the exact test set accuracy of our model, we can be 95% confident that the model test set performance is within the computed lower and upper error bound. We will address the issue of how confidence intervals and error rates are computed in Chapter 7.

The Titanic Dataset

The Titanic dataset contains 2201 data instances. Each data instance represents information about one passenger or crew member aboard the Titanic when it sank into the Atlantic Ocean on April 15, 1912. An individual instance has three input attribute values: the class of the individual (first, second, third, or crew), the individual's age (adult or child), and the sex of the individual. The fourth attribute is an output attribute that tells whether the individual survived to tell the story of the tragic event. The dataset is of historic interest and allows us to develop a profile of a "typical" survivor of this unusual event. The dataset is found in the iDA samples directory as Titanic.xls.

Erroneous and Missing Data

The preprocessor lets you know if an input data file contains errors. Incorrectly formatted data can take many forms. Common mistakes include:

- Blank lines without any valid data.
- Instances containing values in cells beyond the last attribute column.
- Numeric attributes containing invalid characters.

A blank cell indicates a missing data item. Rows containing valid data and one or more blank cells are not flagged as errors. The iDA mining tools have a built-in facility to process missing data items.

When we begin a mining session with a file containing one or more errors that cannot be resolved by the preprocessor, a message box informing us about the total number of errors appears on our screen. We are offered the option to continue the mining session with the valid data. In any case, the rows containing one or more unresolved errors are highlighted in red. Also, an MS Word document offering a general explanation about the location of the errors appears in the upper left of the Excel spreadsheet. Clicking on the document opens the file. You can delete the file by placing the mouse pointer over the document and striking the delete key.

Many real-world datasets contain a wealth of missing and/or erroneous information. The Spine Clinic Dataset that comes with your iDA software package fits this

The Spine Clinic Dataset

The dataset contains three-month follow-up information collected by a metropolitan spine clinic on 171 of their patients who have had back surgery. The dataset consists of 31 attributes. Several attributes relate to the condition of the patient before and during surgery. Other attributes store information about current patient health as well as general patient characteristics. The attribute *return to work* is of particular interest as it indicates whether the patient has returned to work. Ninety-seven of

the patients had returned to work at the time of the survey.

The dataset is interesting because it contains information about real patients who have had major surgery. Also, although the dataset has several missing and erroneous entries, useful patterns differentiating patients who have and have not returned to work can be found. The dataset is located in the iDA samples directory as *SpineData.xls*. ■





category. The Spine Clinic Dataset consists of three-month follow-up information for patients who have had back surgery. Many of the data entries for individual patients are missing, and some attribute values are erroneous. The description box titled *The Spine Clinic Dataset* as well as Appendix B provide additional information about this dataset. Data Mining Exercise 5 asks you to profile patients who have returned to work after back surgery.

4.9 Chapter Summary

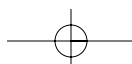
Data mining is an iterative process that involves a degree of art as well as science. Because of this, everyone develops their own special techniques for creating better models through data mining. In this chapter we offered a methodology for data mining using two of the data mining tools that are part of your iDA software package—ESX and RuleMaker. ESX forms a concept hierarchy from a set of instances to perform both supervised learning and unsupervised clustering. Whether learning is supervised or unsupervised, the following is a plausible strategy for analyzing a data mining session with ESX:

1. Examine class resemblance scores and individual class rules to see how well the input attributes define the formed classes.
2. Examine domain summary statistics to help locate attribute errors and to determine predictive numeric attributes.
3. Examine class summary statistics to locate predictive categorical attribute values.
4. Modify attribute or instance selections and parameter settings as appropriate and repeat the data mining process to create a best data model.

RuleMaker creates sets of production rules from instances of known classification. RuleMaker offers several rule generation parameters that include the option to obtain a listing of all possibly interesting rules. In the next chapter you will learn more about how ESX and RuleMaker create generalized models from data.

4.10 Key Terms

Class predictability. Given class C and categorical attribute A with values $v_1, v_2, v_3, \dots, v_n$, the class C predictability score for v_i tells us the percent of instances showing v_i as the value for A in C .



Class predictiveness. Given class C and attribute A with values $v_1, v_2, v_3 \dots v_n$. An attribute-value predictiveness score for v_i is defined as the probability an instance resides in C given the instance has value v_i for A .

Class prototype. The instance that best represents a class of instances.

Class resemblance. The average similarity of all instances within a class or cluster.

Cluster quality. For an individual class, cluster quality is the percent increase (or decrease) of the class resemblance score relative to the domain resemblance.

Domain predictability. Given attribute A with values $v_1, v_2, v_3 \dots v_n$, the domain predictability score for v_i tells us the percent of all dataset instances showing v_i as the value for A .

Domain resemblance. The overall similarity of all instances within a dataset.

Necessary attribute value. Value v_i for categorical attribute A is necessary for membership in class C if and only if all instances of C have v_i as their value for A .

Sufficient attribute value. Value v_i for categorical attribute A is sufficient for membership in class C if and only if any instance having v_i as its value for A resides in C .

Typicality. The typicality of instance I is the average similarity of I to the other members of its cluster or class.

4.11 Exercises

Review Questions

1. Differentiate between the following terms.
 - a. Domain resemblance and class resemblance
 - b. Class predictiveness and class predictability
 - c. Domain predictability and class predictability
 - d. Typicality and class resemblance
 - e. Within-class and between-class measure
2. Suppose you have used data mining to build a model able to differentiate between individuals likely and unlikely to default on a car loan. For each of the following, describe a categorical attribute value likely to display the stated characteristic.
 - a. A categorical attribute value that is necessary but not sufficient for class membership.

- b. A categorical attribute value that is sufficient but not necessary for class membership.
- c. A categorical attribute that is both necessary and sufficient for class membership.

Data Mining Questions

- LAB 1.** This exercise demonstrates how erroneous data is displayed in a spreadsheet file.
- a. Copy the CreditCardPromotion.xls dataset into a new spreadsheet.
 - b. Modify the instance on line 17 to contain one or more illegal characters for age.
 - c. Add one or more blank lines to the spreadsheet.
 - d. Remove values from one or more spreadsheet cells.
 - e. Using *life insurance promotion* as the output attribute, initiate a data mining session with ESX. When asked if you wish to continue mining the good instances, answer *No*. Open the Word document located in the upper-left corner of your spreadsheet to examine the error messages.
2. Suppose you suspect marked differences in promotional purchasing trends between female and male Acme credit card customers. You wish to confirm or refute your suspicion. Perform a supervised data mining session using the CreditCardPromotion.xls database with *sex* as the output attribute. Designate all other attributes as input attributes, and use all 15 instances for training. Because there is no test data, the RES TST and RES MTX sheets will not be created. Write a summary confirming or refuting your hypothesis. Base your analysis on:
- a. Class resemblance scores
 - b. Class predictability and predictiveness scores
 - c. Rules created for each class. You may wish to use the rerule feature.
3. Repeat the previous exercise with *income range* as the output attribute.
- LAB 4.** For this exercise you will use ESX to perform a data mining session with the cardiology patient data described in Chapter 2. Load the CardiologyCategorical.xls file into a new MS Excel spreadsheet. This is the mixed form of the dataset containing both categorical and numeric data. Recall that the data contains 303 instances representing patients who have a heart condition (*sick*) as well as those who do not.

LAB Denotes exercise appropriate for a laboratory setting.

Save the spreadsheet to a new file and perform a supervised mining session using *class* as the output attribute. Use 1.0 for the real-tolerance setting and select 203 instances as training data. The final 100 instances represent the test dataset. Generate rules using the default settings for the rule generator. Answer the following based on your results:

- a. Provide the domain resemblance score as well as the resemblance score for each class (sick and healthy).
 - b. What percent of the training data is female?
 - c. What is the most commonly occurring domain value for the attribute *slope*?
 - d. What is the average age of those individuals in the *healthy* class?
 - e. What is the most common *healthy* class value for the attribute *thal*?
 - f. Specify blood pressure values for the two most typical *sick* class instances.
 - g. What percent of the *sick* class is female?
 - h. What is the predictiveness score for the *sick* class attribute value *angina = true*? In your own words, explain what this value means.
 - i. List one *sick* class attribute value that is highly sufficient for class membership.
 - j. What percent of the test set instances were correctly classified?
 - k. Give the 95% confidence limits for the test set. State what the confidence limit values mean.
 - l. How many test set instances were classified as being sick when in reality they were from the *healthy* class?
 - m. List a rule with multiple conditions for the *sick* class that covers at least 50% of the instances and is accurate in at least 85% of all cases.
 - n. What is the most highly predictive numeric attribute?
5. The SpineClinic.xls data file has an attribute named *return to work*. A value of 1 for the attribute indicates the patient has returned to work. Load this dataset into an Excel spreadsheet and designate the *return to work* attribute as an output attribute. Perform a data mining session with ESX using all 171 patient records as training data. Generate rules in an attempt to find one or two interesting rules that differentiate individuals who have and have not returned to work. Use the rerule feature if necessary.
- LAB 6.** In this exercise you will use instance typicality to determine class prototypes. You will then employ the prototype instances to build and test a supervised learner model.
- a. Open the CardiologyCategorical.xls spreadsheet file.



- b. Save the file to a new spreadsheet.
 - c. Perform a supervised mining session on the *class* attribute. Use all instances for training.
 - d. When learning is complete, open the RESTYP sheet and copy the most typical sick class instance to a new spreadsheet. Save the spreadsheet and return to the RESTYP sheet.
 - e. Now, copy the most typical healthy class instance to the spreadsheet created in the previous step and currently holding the single most typical sick class instance. Save the updated spreadsheet file.
 - f. Delete columns A, B, and Q of the new spreadsheet file that now contains the most typical healthy class instance and the most typical sick class instance. Copy the two instances contained in the spreadsheet.
 - g. Return to the original sheet1 data sheet and insert two blank rows after row three. Paste the two instances copied in step f into sheet1.
 - h. Your sheet1 spreadsheet now contains 305 instances. The first instance in sheet1 (row 4) is the most typical sick class instance. The second instance is the most typical healthy class instance.
 - i. Perform a data mining session using the first *two* instances as training data. The remaining 303 instances will make up the test set.
 - j. Analyze the test set results by examining the confusion matrix. What can you conclude?
 - k. Repeat the above steps but this time extract the least typical instance from each class. How do your results compare with those of the first experiment?
7. Add one or more unknown data instances to the CreditCardPromotion.xls database. Use supervised learning on the *life insurance promotion* attribute together with the procedure described in Section 4.7 to compute classification confidence scores for the new instances.
- LAB 8.** Perform a supervised data mining session with the Titanic.xls dataset. Use 1500 instances for training and the remaining instances as test data.
- a. Are any of the input attributes highly predictive of class membership for either class?
 - b. What is the test set accuracy of the model?
 - c. Use the confidence limits to state the 95% test set accuracy range.
 - d. Why is the classification accuracy so much lower for this experiment than for the quick mine experiment given in Section 4.8?

Computational Questions

1. Concept class C_1 shows the following information for the categorical attribute *color*. Use this information and the information in the table to answer the following questions:

Name	Value	Frequency	Predictability	Predictiveness
Color	Red	30		0.4
	Black	20		1.0

- What percent of the instances in class C_1 have a value of *black* for the *color* attribute?
- Suppose that exactly one other concept class, C_2 , exists. In addition, assume all domain instances have a color value of either *red* or *black*. Given the information in the table, can you determine the predictability score of $color = red$ for class C_2 ? If your answer is yes, what is the predictability value?
- Using the same assumption as in part b, can you determine the predictiveness score for $color = red$ for class C_2 ? If your answer is yes, what is the predictiveness score?
- Once again, use the assumption stated in part b. How many instances reside in class C_2 ?