Data Mining In Excel: Lecture Notes and Cases

Draft December 30, 2005

Galit Shmueli
Nitin R. Patel
Peter C. Bruce

Distributed by:
Resampling Stats, Inc.
612 N. Jackson St.
Arlington, VA 22201
USA
info@xlminer.com
www.xlminer.com

Contents

Chapter 1

Introduction

1.1 Who Is This Book For?

This book arose out of a data mining course at MIT's Sloan School of Management. Preparation for the course revealed that there are a number of excellent books on the business context of data mining, but their coverage of the statistical and machine-learning algorithms that underlie data mining is not sufficiently detailed to provide a practical guide if the instructor's goal is to equip students with the skills and tools to implement those algorithms. On the other hand, there are also a number of more technical books about data mining algorithms, but these are aimed at the statistical researcher, or more advanced graduate student, and do not provide the case-oriented business focus that is successful in teaching business students.

Hence, this book is intended for the business student (and practitioner) of data mining techniques, and its goal is threefold:

1. To provide both a theoretical and practical understanding of the key methods of classification, prediction, reduction and exploration that are at the heart of data mining;

2. To provide a business decision-making context for these methods;

3. Using real business cases, to illustrate the application and interpretation of these methods.

An important feature of this book is the use of Excel, an environment familiar to business analysts. All required data mining algorithms (plus illustrative datasets) are provided in an Excel add-in, XLMiner. XLMiner offers a variety of data mining tools: neural nets, classification and regression trees, k-nearest neighbor classification, naive Bayes, logistic regression, multiple linear regression, and discriminant analysis, all for predictive modeling. It provides for automatic partitioning of data into training, validation and test samples, and for the deployment of the model to new data. It also offers association rules, principal components analysis, k-means clustering and hierarchical clustering, as well as visualization tools, and data handling utilities. With its short learning curve, affordable price, and reliance on the familiar Excel platform, it is an ideal companion to a book on data mining for the business student.

The presentation of the cases in the book is structured so that the reader can follow along and implement the algorithms on his or her own with a very low learning hurdle.

Just as a natural science course without a lab component would seem incomplete, a data mining course without practical work with actual data is missing a key ingredient. The MIT data mining course that gave rise to this book followed an introductory quantitative course that relied on Excel – this made its practical work universally accessible. Using Excel for data mining seemed a natural progression.

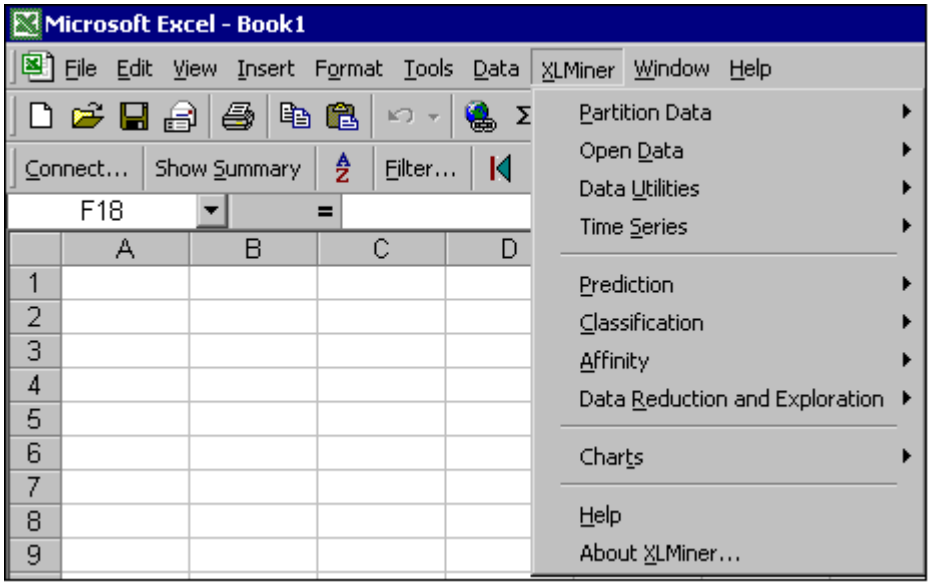
While the genesis for this book lay in the need for a case-oriented guide to teaching data mining, analysts and consultants who are considering the application of data mining techniques in contexts where they are not currently in use will also find this a useful, practical guide.

Using XLMiner Software

This book is based on using the XLMiner software. The illustrations, exercises, and cases are written with relation to this software. XLMiner is a comprehensive data mining add-in for Excel, which is easy to learn for users of Excel. It is a tool to help you get quickly started on data mining, offering a variety of methods to analyze data. It has extensive coverage of statistical and data mining techniques for classification, prediction, affinity analysis, and data exploration and reduction.

Installation: Click on setup.exe and installation dialog boxes will guide you through the installation procedure. After installation is complete, the XLMiner program group appears under Start→ Programs→ XLMiner. You can either invoke XLMiner directly or select the option to register XLMiner as an Excel Add-in.

Use: Once opened, XLMiner appears as another menu in the top toolbar in Excel, as shown in the figure below. By choosing the appropriate menu item, you can run any of XLMiner's procedures on the dataset that is open in the Excel worksheet.



1.2 What Is Data Mining?

The field of data mining is still relatively new, and in a state of evolution. The first International Conference on Knowledge Discovery and Data Mining ("KDD") was held in 1995, and there are a variety of definitions of data mining.

A concise definition that captures the essence of data mining is:

"Extracting useful information from large datasets" (Hand et al., 2001).

A slightly longer version is:

> "Data mining is the process of exploration and analysis, by automatic or semi-automatic means, of large quantities of data in order to discover meaningful patterns and rules" (Berry and Linoff: 1997 and 2000).

Berry and Linoff later had cause to regret the 1997 reference to "automatic and semi-automatic means," feeling it shortchanged the role of data exploration and analysis.

Another definition comes from the Gartner Group, the information technology research firm (from their web site, Jan. 2004):

> "Data mining is the process of discovering meaningful new correlations, patterns and trends by sifting through large amounts of data stored in repositories, using pattern recognition technologies as well as statistical and mathematical techniques."

A summary of the variety of methods encompassed in the term "data mining" is found at the beginning of Chapter 2 (Core Ideas).

1.3 Where Is Data Mining Used?

Data mining is used in a variety of fields and applications. The military use data mining to learn what roles various factors play in the accuracy of bombs. Intelligence agencies might use it to determine which of a huge quantity of intercepted communications are of interest. Security specialists might use these methods to determine whether a packet of network data constitutes a threat. Medical researchers might use them to predict the likelihood of a cancer relapse.

Although data mining methods and tools have general applicability, most examples in this book are chosen from the business world. Some common business questions one might address through data mining methods include:

1. From a large list of prospective customers, which are most likely to respond? We can use classification techniques (logistic regression, classification trees or other methods) to identify those individuals whose demographic and other data most closely matches that of our best existing customers. Similarly, we can use prediction techniques to forecast how much individual prospects will spend.

2. Which customers are most likely to commit, for example, fraud (or might already have committed it)? We can use classification methods to identify (say) medical reimbursement applications that have a higher probability of involving fraud, and give them greater attention.

3. Which loan applicants are likely to default? We can use classification techniques to identify them (or logistic regression to assign a "probability of default" value).

4. Which customers are more likely to abandon a subscription service (telephone, magazine, etc.)? Again, we can use classification techniques to identify them (or logistic regression to assign a "probability of leaving" value). In this way, discounts or other enticements can be proffered selectively.

1.4 The Origins of Data Mining

Data mining stands at the confluence of the fields of statistics and machine learning (also known as artificial intelligence). A variety of techniques for exploring data and building models have been around for a long time in the world of statistics - linear regression , logistic regression, discriminant

analysis and principal components analysis, for example. But the core tenets of classical statistics-computing is difficult and data are scarce - do not apply in data mining applications where both data and computing power are plentiful.

This gives rise to Daryl Pregibon's description of data mining as "statistics at scale and speed" (Pregibon, 1999). A useful extension of this is "statistics at scale, speed, and simplicity." Simplicity in this case refers not to simplicity of algorithms, but rather to simplicity in the logic of inference. Due to the scarcity of data in the classical statistical setting, the same sample is used to make an estimate, and also to determine how reliable that estimate might be. As a result, the logic of the confidence intervals and hypothesis tests used for inference may seem elusive for many, and their limitations are not well appreciated. By contrast, the data mining paradigm of fitting a model with one sample and assessing its performance with another sample is easily understood.

Computer science has brought us "machine learning" techniques, such as trees and neural networks , that rely on computational intensity and are less structured than classical statistical models. In addition, the growing field of database management is also part of the picture.

The emphasis that classical statistics places on inference (determining whether a pattern or interesting result might have happened by chance) is missing in data mining. In comparison to statistics, data mining deals with large datasets in open-ended fashion, making it impossible to put the strict limits around the question being addressed that inference would require.

As a result, the general approach to data mining is vulnerable to the danger of "overfitting," where a model is fit so closely to the available sample of data that it describes not merely structural characteristics of the data, but random peculiarities as well. In engineering terms, the model is fitting the noise, not just the signal.

1.5 The Rapid Growth of Data Mining

Perhaps the most important factor propelling the growth of data mining is the growth of data. The mass retailer Walmart in 2003 captured 20 million transactions per day in a 10-terabyte database (a terabyte is 1,000,000 megabytes). In 1950, the largest companies had only enough data to occupy, in electronic form, several dozen megabytes. Lyman and Varian (2003) estimate that 5 exabytes of information were produced in 2002, double what was produced in 1999 (an exabyte is one million terabytes). 40% of this was produced in the U.S.

The growth of data is driven not simply by an expanding economy and knowledge base, but by the decreasing cost and increasing availability of automatic data capture mechanisms. Not only are more events being recorded, but more information per event is captured. Scannable bar codes, point of sale (POS) devices, mouse click trails, and global positioning satellite (GPS) data are examples.

The growth of the internet has created a vast new arena for information generation. Many of the same actions that people undertake in retail shopping, exploring a library or catalog shopping have close analogs on the internet, and all can now be measured in the most minute detail.

In marketing, a shift in focus from products and services to a focus on the customer and his or her needs has created a demand for detailed data on customers.

The operational databases used to record individual transactions in support of routine business activity can handle simple queries, but are not adequate for more complex and aggregate analysis. Data from these operational databases are therefore extracted, transformed and exported to a *data warehouse* - a large integrated data storage facility that ties together the decision support systems of an enterprise. Smaller *data marts* devoted to a single subject may also be part of the system. They may include data from external sources (e.g., credit rating data).

Many of the exploratory and analytical techniques used in data mining would not be possible without today's computational power. The constantly declining cost of data storage and retrieval has made it possible to build the facilities required to store and make available vast amounts of data. In short, the rapid and continuing improvement in computing capacity is an essential enabler of the

growth of data mining.

1.6 Why are there so many different methods?

As can be seen in this book or any other resource on data mining, there are many different methods for prediction and classification. You might ask yourself why they coexist, and whether some are better than others. The answer is that each method has its advantages and disadvantages. The usefulness of a method can depend on factors such as the size of the dataset, the types of patterns that exist in the data, whether the data meet some underlying assumptions of the method, how noisy the data are, the particular goal of the analysis, etc. A small illustration is shown in Figure 1.1, where the goal is to find a combination of *household income level* and *household lot size* that separate buyers (solid circles) from non-buyers (hollow circles) of riding mowers. The first method (left panel) looks only for horizontal and vertical lines to separate buyers from non-buyers, whereas the second method (right panel) looks for a single diagonal line.

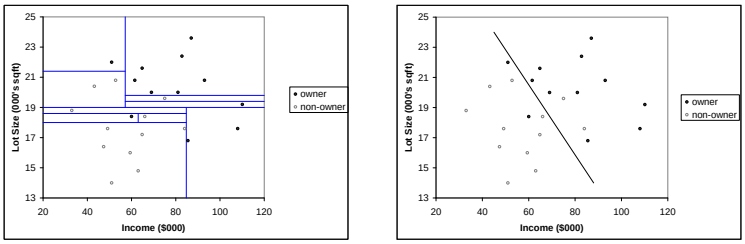


Figure 1.1: Two different methods for separating buyers from non-buyers

Different methods can lead to different results, and their performance can vary. It is therefore customary in data mining to apply several different methods and select the one that is most useful for the goal at hand.

1.7 Terminology and Notation

Because of the hybrid parentry of data mining, its practitioners often use multiple terms to refer to the same thing. For example, in the machine learning (artificial intelligence) field, the variable being predicted is the output variable or the target variable. To a statistician, it is the dependent variable or the response. Here is a summary of terms used:

Algorithm refers to a specific procedure used to implement a particular data mining technique-classification tree, discriminant analysis, etc.

Attribute - see **Predictor**.

Case - see **Observation**.

Confidence has a specific meaning in association rules of the type "If A and B are purchased, C is also purchased." Confidence is the conditional probability that C will be purchased, IF A and B are purchased.

Confidence also has a broader meaning in statistics ("confidence interval"), concerning the degree of error in an estimate that results from selecting one sample as opposed to another.

Dependent variable - see **Response**.

Estimation - see **Prediction**.

Feature - see **Predictor**.

Holdout sample is a sample of data not used in fitting a model, used to assess the performance of that model; this book uses the terms *validation set* or, if one is used in the problem, *test set* instead of holdout sample.

Input variable - see **Predictor**.

Model refers to an algorithm as applied to a dataset, complete with its settings (many of the algorithms have parameters which the user can adjust).

Observation is the unit of analysis on which the measurements are taken (a customer, a transaction, etc.); also called *case*, *record*, *pattern* or *row*. (each row typically represents a record, each column a variable)

Outcome variable - see **Response**.

Output variable - see **Response**.

P(A|B) is the conditional probability of event A occurring given that event B has occurred. Read as "the probability that A will occur, given that B has occurred."

Pattern is a set of measurements on an observation (e.g., the height, weight, and age of a person)

Prediction means the prediction of the value of a continuous output variable; also called *estimation.*

Predictor usually denoted by X, is also called a *feature*, *input variable*, *independent variable*, or, from a database perspective, a *field*.

Record - see **Observation**.

Response , usually denoted by Y, is the variable being predicted in supervised learning; also called *dependent variable*, *output variable*, *target variable* or *outcome variable*.

Score refers to a predicted value or class. "Scoring new data" means to use a model developed with training data to predict output values in new data.

Success class is the class of interest in a binary outcome (e.g., "purchasers" in the outcome "purchase/no-purchase")

Supervised learning refers to the process of providing an algorithm (logistic regression, regression tree, etc.) with records in which an output variable of interest is known and the algorithm "learns" how to predict this value with new records where the output is unknown.

Test data (or test set) refers to that portion of the data used only at the end of the model building and selection process to assess how well the final model might perform on additional data.

Training data (or training set) refers to that portion of data used to fit a model.

Unsupervised learning refers to analysis in which one attempts to learn something about the data other than predicting an output value of interest (whether it falls into clusters, for example).

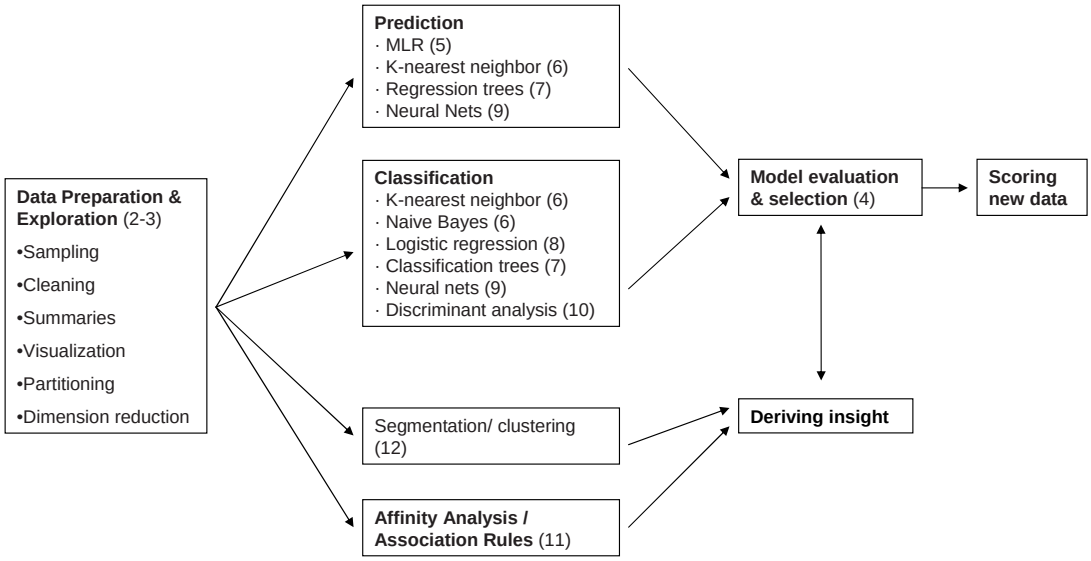


Figure 1.2: Data Mining From A Process Perspective

Validation data (or validation set) refers to that portion of the data used to assess how well the model fits, to adjust some models, and to select the best model from among those that have been tried.

Variable is any measurement on the records, including both the input (X) variables and the output (Y) variable.

1.8 Road Maps to This Book

The book covers many of the widely-used predictive and classification methods, as well as other data mining tools. Figure 1.2 outlines data mining from a process perspective, and where the topics in this book fit in. Chapter numbers are indicated beside the topic.

Table 1.1 provides a different perspective - what type of data we have, and what that says about the data mining procedures available.

Order of Topics

The chapters are generally divided into three parts: Chapters 1-3 cover general topics, Chapters 4-10 cover prediction and classification methods, and Chapters 11-12 discuss association rules and cluster analysis. Within the prediction and classification group of chapters, the topics are generally organized according to the level of sophistication of the algorithms, their popularity, and ease of understanding. Although the topics in the book can be covered in the order of the chapters, each chapter (aside from Chapters 1-4) stands alone so that it can be dropped or covered at a different time without loss in comprehension.

Table 1.1: Organization Of Data Mining Methods In This Book, According To The Nature Of The Data

	Continuous Response	Categorical Response	No Response
Continuous Predictors	Linear Reg (5) Neural Nets (9) KNN (6)	Logistic Reg (8) Neural Nets (9) Discriminant Analysis (10) KNN (6)	Principal Components (3) Cluster Analysis (12)
Categorical Predictors	Linear Reg (5) Neural Nets (9) Reg Trees (7)	Neural Nets (9) Classification Trees (7) Logistic Reg (8) Naive Bayes (6)	Association Rules (11)

Note: Chapter 3 (Data Exploration and Dimension Reduction) also covers principal components analysis as a method for dimension reduction. Instructors may wish to defer covering PCA to a later point.

Chapter 2

Overview of the Data Mining Process

2.1 Introduction

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

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

2.2 Core Ideas in Data Mining

2.2.1 Classification

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

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

2.2.2 Prediction

Prediction is similar to classification, except we are trying to predict the value of a numerical variable (e.g., amount of purchase), rather than a class (e.g. purchaser or nonpurchaser).

Of course, in classification we are trying to predict a class, but the term "prediction" in this book refers to the prediction of the value of a continuous variable. (Sometimes in the data mining literature, the term "estimation" is used to refer to the prediction of the value of a continuous variable, and "prediction" may be used for both continuous and categorical data.)

2.2.3 Association Rules

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

2.2.4 Predictive Analytics

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

2.2.5 Data Reduction

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

2.2.6 Data Exploration

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

2.2.7 Data Visualization

Another technique for exploring data to see what information they hold is through graphical analysis. This includes looking at each variable separately as well as looking at relationships between variables. For numeric variables we use histograms and boxplots to learn about the distribution of their values, to detect outliers (extreme observations), and to find other information that is relevant to the analysis task. Similarly, for categorical variables we use bar charts and pie charts. We can also look at scatter plots of pairs of numeric variables to learn about possible relationships, the type of relationship, and again, to detect outliers.

2.3 Supervised and Unsupervised Learning

A fundamental distinction among data mining techniques is between supervised methods and unsupervised methods.

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

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

2.4 The Steps in Data Mining

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

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

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

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

4. Reduce the data, if necessary, and (where supervised training is involved) separate them into training, validation and test datasets. This can involve operations such as eliminating unneeded variables, transforming variables (for example, turning "money spent" into "spent $>$ \$100" vs. "spent $\leq$ \$100"), and creating new variables (for example, a variable that records whether at

least one of several products was purchased). Make sure you know what each variable means, and whether it is sensible to include it in the model.

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

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

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

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

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

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

Sample: from datasets, partition into training, validation and test datasets

Explore: dataset statistically and graphically

Modify: transform variables, impute missing values

Model: fit predictive models, e.g. regression, tree, collaborative filtering

Assess: compare models using validation dataset

SPSS-Clementine also has a similar methodology, termed CRISP-DM (CRoss-Industry Standard Process for Data Mining).

2.5 Preliminary Steps

2.5.1 Organization of Datasets

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

2.5.2 Sampling from a Database

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

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

2.5.3 Oversampling Rare Events

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

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

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

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

2.5.4 Pre-processing and Cleaning the Data

Types of Variables

There are several ways of classifying variables. Variables can be numeric or text (character). They can be continuous (able to assume any real numeric value, usually in a given range), integer (assuming only integer values), or categorical (assuming one of a limited number of values). Categorical variables can be either numeric $(1, 2, 3)$ or text (payments current, payments not current, bankrupt). Categorical variables can also be unordered (called "nominal variables") with categories

such as North America, Europe, and Asia; or they can be ordered (called "ordinal variables") with categories such as high value, low value, and nil value.

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

Handling Categorical Variables

Categorical variables can also be handled by most routines, but often require special handling.

If the categorical variable is ordered (age category, degree of creditworthiness, etc.), then we can often use it as is, as if it were a continuous variable. The smaller the number of categories, and the less they represent equal increments of value, the more problematic this procedure becomes, but it often works well enough.

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

Student - yes/no

Unemployed - yes/no

Employed - yes/no

Retired - yes/no

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

XLMiner has a utility to convert categorical variables to binary dummies.

Variable Selection

More is not necessarily better when it comes to selecting variables for a model. Other things being equal, parsimony, or compactness, is a desirable feature in a model.

For one thing, the more variables we include, the greater the number of records we will need to assess relationships among the variables. 15 records may suffice to give us a rough idea of the relationship between Y and a single predictor variable X. If we now want information about the relationship between Y and fifteen predictor variables $X_1 \cdots X_{15}$, fifteen records will not be enough (each estimated relationship would have an average of only one record's worth of information, making the estimate very unreliable).

Overfitting

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

Consider the following hypothetical data about advertising expenditures in one time period, and sales in a subsequent time period: (a scatter plot of the data is shown in Figure 2.1)

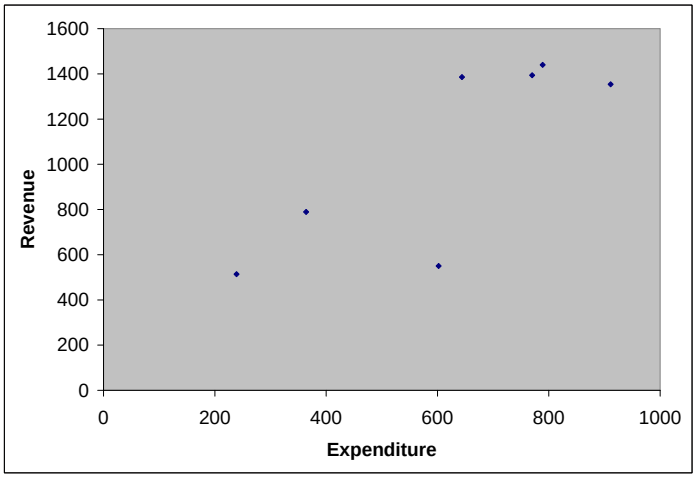


Figure 2.1: X-Y Scatterplot For Advertising And Sales Data

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

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

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

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

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

For example, one of the variables might be height. We have no basis in theory to suppose that tall people might contribute more or less to charity, but if there are several tall people in our sample

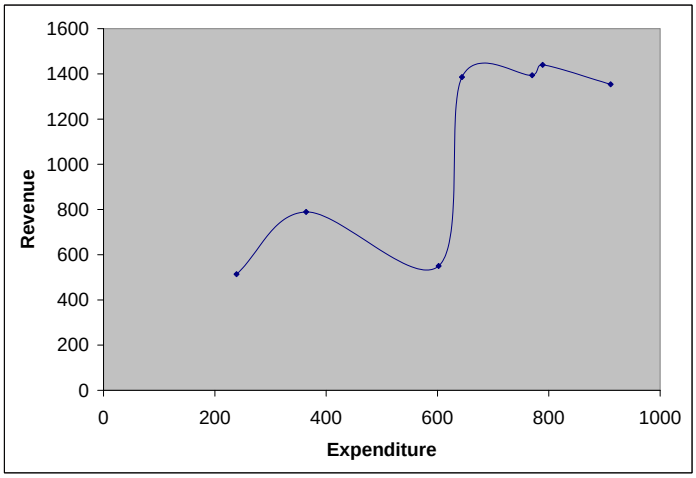


Figure 2.2: X-Y Scatterplot, Smoothed

and they just happened to contribute heavily to charity, our model might include a term for height - the taller you are, the more you will contribute. Of course, when the model is applied to additional data, it is likely that this will not turn out to be a good predictor.

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

Somewhat surprisingly, even if we know for a fact that a higher degree curve is the appropriate model, if the model-fitting dataset is not large enough, a lower degree function (that is not as likely to fit the noise) is likely to perform better.

Overfitting can also result from the application of many different models, from which the best performing is selected (see below).

How Many Variables and How Much Data?

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

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

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

anything.

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

Outliers

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

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

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

If manual review is feasible, some outliers may be identified and corrected. In any case, if the number of records with outliers is very small, they might be treated as missing data.

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

Missing Values

Typically, some records will contain missing values. If the number of records with missing values is small, those records might be omitted.

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

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

and indeed the performance of our data mining technique, using the validation data, this need not present a major problem.

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

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

Normalizing (Standardizing) the Data

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

To consider why this might be necessary, consider the case of clustering. Clustering typically involves calculating a distance measure that reflects how far each record is from a cluster center, or from other records. With multiple variables, different units will be used - days, dollars, counts, etc. If the dollars are in the thousands and everything else is in the 10's, the dollar variable will come to dominate the distance measure. Moreover, changing units from (say) days to hours or months could completely alter the outcome.

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

2.5.5 Use and Creation of Partitions

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

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

- A superior model
- Chance aspects of the data that happen to match the chosen model better than other models.

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

To address this problem, we simply divide (partition) our data and develop our model using only one of the partitions. After we have a model, we try it out on another partition and see how

it does. We can measure how it does in several ways. In a classification model, we can count the proportion of held-back records that were misclassified. In a prediction model, we can measure the residuals (errors) between the predicted values and the actual values.

We will typically deal with two or three partitions: a training set, a validation set, and sometimes an additional test set. Partitioning the data into training, validation and test sets is done either randomly according to predetermined proportions, or by specifying which records go into which partitioning according to some relevant variable (e.g., in time series forecasting, the data are partitioned according to their chronological order). In most cases the partitioning should be done randomly to avoid getting a biased partition. It is also possible (though cumbersome) to divide the data into more than 3 partitions by successive partitioning - e.g., divide the initial data into 3 partitions, then take one of those partitions and partition it further.

Training Partition

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

Validation Partition

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

Test Partition

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

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

The random features of the validation data that enhance the apparent performance of the chosen model will not likely be present in new data to which the model is applied. Therefore, we may have overestimated the accuracy of our model. The more models we test, the more likely it is that one of them will be particularly effective in explaining the noise in the validation data. Applying the model to the test data, which it has not seen before, will provide an unbiased estimate of how well it will do with new data. The diagram in Figure 2.3 shows the three partitions and their use in the data mining process.

When we are concerned mainly with finding the best model and less with exactly how well it will do, we might use only training and validation partitions.

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

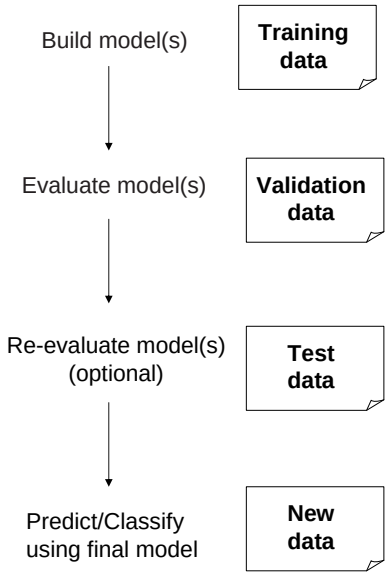


Figure 2.3: The Three Data Partitions and Their Role in the Data Mining Process

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

2.6 Building a Model - An Example with Linear Regression

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

The Boston Housing Data

The Boston Housing data contains information on neighborhoods in Boston for which several measurements are taken (crime rate, pupil/teacher ratio, etc.). The outcome variable of interest is the median value of a housing unit in the neighborhood. This dataset has 14 variables and a description of each variable is given in Table 2.1. The data themselves are shown in Figure 2.4. The first row in this figure represents the first neighborhood, which had an average per capita crime rate of .006, had 18% of the residential land zoned for lots over 25,000 square feet, 2.31% of the land devoted to non-retail business, no border on the Charles River, etc.

The modeling process

We now describe in detail the different model stages using the Boston Housing example.

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

	A	B	C	D	E	F	G	H	I	J	K	L	M	N
	CRIM	ZN	INDUS	CHAS	NOX	RM	AGE	DIS	RAD	TAX	PTRATIO	B	LSTAT	MEDV
1	0.006	18	2.31	0	0.54	6.58	65.2	4.09	1	296	15.3	397	5	24
2	0.027	0	7.07	0	0.47	6.42	78.9	4.97	2	242	17.8	397	9	21.6
3	0.027	0	7.07	0	0.47	7.19	61.1	4.97	2	242	17.8	393	4	34.7
4	0.032	0	2.18	0	0.46	7.00	45.8	6.06	3	222	18.7	395	3	33.4
5	0.069	0	2.18	0	0.46	7.15	54.2	6.06	3	222	18.7	397	5	36.2
6	0.030	0	2.18	0	0.46	6.43	58.7	6.06	3	222	18.7	394	5	28.7
7	0.088	12.5	7.87	0	0.52	6.01	66.6	5.56	5	311	15.2	396	12	22.9
8	0.145	12.5	7.87	0	0.52	6.17	96.1	5.95	5	311	15.2	397	19	27.1
9	0.211	12.5	7.87	0	0.52	5.63	100	6.08	5	311	15.2	387	30	16.5
10	0.170	12.5	7.87	0	0.52	6.00	85.9	6.59	5	311	15.2	387	17	18.9

Figure 2.4: Boston Housing Data

Table 2.1: Description Of Variables in Boston Housing Dataset

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

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

3. Explore, clean, and preprocess the data.

   Let's look first at the description of the variables (crime rate, number of rooms per dwelling, etc.) to be sure we understand them all. These descriptions are available on the "description" tab on the worksheet, as is a web source for the dataset. They all seem fairly straightforward, but this is not always the case. Often variable names are cryptic and their descriptions may be unclear or missing.

   It is useful to pause and think about what the variables mean, and whether they should be included in the model. Consider the variable TAX. At first glance, we consider that tax on a home is usually a function of its assessed value, so there is some circularity in the model - we want to predict a home's value using TAX as a predictor, yet TAX itself is determined by a home's value. TAX might be a very good predictor of home value in a numerical sense, but would it be useful if we wanted to apply our model to homes whose assessed value might not be

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

Figure 2.5: Outlier in Boston Housing Data

known? Reflect, though, that the TAX variable, like all the variables, pertains to the average in a neighborhood, not to individual homes. While the purpose of our inquiry has not been spelled out, it is possible that at some stage we might want to apply a model to individual homes and, in such a case, the neighborhood TAX value would be a useful predictor. So, we will keep TAX in the analysis for now.

In addition to these variables, the dataset also contains an additional variable, CATMEDV, which has been created by categorizing median value (MEDV) into two categories – high and low. The variable CATMEDV is actually a categorical variable created from MEDV. If MEDV $\geq \$30,000$, CATV = 1. If MEDV $\leq \$30,000$, CATV = 0. If we were trying to categorize the cases into high and low median values, we would use CAT MEDV instead of MEDV. As it is, we do not need CAT MEDV so we will leave it out of the analysis.

There are a couple of aspects of MEDV − the median house value − that bear noting. For one thing, it is quite low, since it dates from the 1970's. For another, there are a lot of 50's, the top value. It could be that median values above $50,000 were recorded as $50,000.

We are left with 13 independent (predictor) variables, which can all be used.

It is also useful to check for outliers that might be errors. For example, suppose the RM (# of rooms) column looked like the one in Figure 2.5, after sorting the data in descending order based on rooms. We can tell right away that the 79.29 is in error - no neighborhood is going to have houses that have an average of 79 rooms. All other values are between 3 and 9. Probably, the decimal was misplaced and the value should be 7.929. (This hypothetical error is not present in the dataset supplied with XLMiner.)

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

   In Excel, select XLMiner → Partition and the dialog box shown in Figure 2.6 appears. Here we specify which data range is to be partitioned, and which variables are to be included in the partitioned dataset.

   The partitioning can be handled in one of two ways:

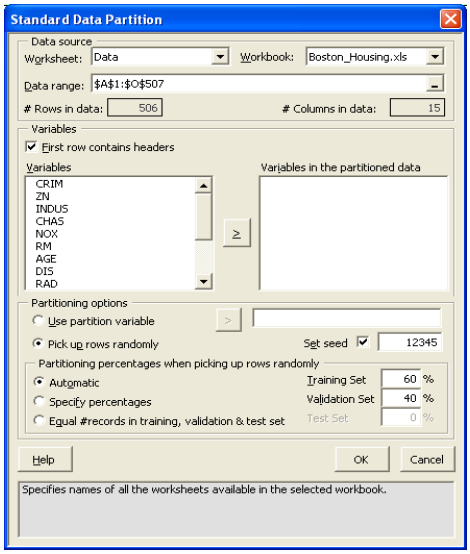


Figure 2.6: Partitioning the Data. The Default in XLMiner Partitions the Data into 60% Training Data, 40% Validation Data, and 0% Test Data

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

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

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

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

Typically, a data mining endeavor involves testing multiple models, perhaps with multiple settings on each model. When we train just one model and try it out on the validation data, we can get an unbiased idea of how it might perform on more such data.

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

The test data, which should not be used either in the model building or model selection process, can give a better estimate of how well the chosen model will do with fresh data. Thus, once

we have selected a final model, we will apply it to the test data to get an estimate of how well it will actually perform.

5. Determine the data mining task. In this case, as noted, the specific task is to predict the value of MEDV using the 13 predictor variables.

6. Choose the technique. In this case, it is multiple linear regression.

   Having divided the data into training and validation partitions, we can use XLMiner to build a multiple linear regression model with the training data - we want to predict median house price on the basis of all the other values.

7. Use the algorithm to perform the task. In XLMiner, we select Prediction → Multiple Linear Regression, as shown in Figure 2.7.

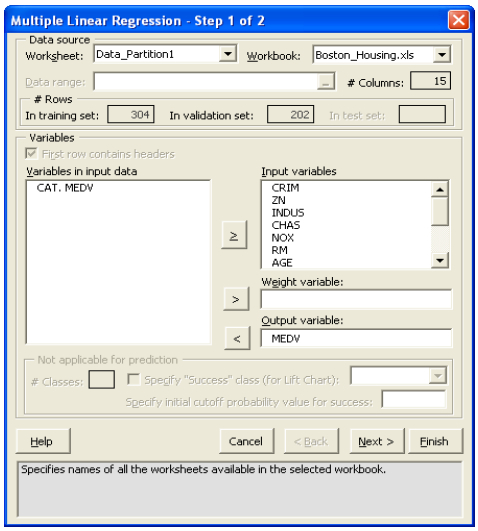


Figure 2.7: Using XLMiner for Multiple Linear Regression

The variable MEDV is selected as the output (dependent) variable, the variable CAT.MEDV is left unused, and the remaining variables are all selected as input (independent or predictor) variables. We will ask XLMiner to show us the fitted values on the training data, as well as the predicted values (scores) on the validation data, as shown in Figure 2.8. XLMiner produces standard regression output, but we will defer that for now, as well as the more advanced options displayed above. (See the chapter on multiple linear regression (Chapter 5), or the user documentation for XLMiner, for more information.) Rather, we will review the predictions themselves. Figure **??** shows the predicted values for the first few records in the training data, along with the actual values and the residual (prediction error). Note that these predicted values would often be called the fitted values, since they are for the records that the model was fit to. The results for the validation data are shown in Figure 2.10. The prediction error for the training and validation data are compared in Figure 2.11. Prediction error can be measured in several ways. Three measures produced by XLMiner are shown in Figure 2.11. On the right is the "average error" - simply the average of the residuals (errors). In both cases, it is quite small, indicating that, on balance, predictions average about right - our predictions are "unbiased." Of course, this simply means that the positive errors and

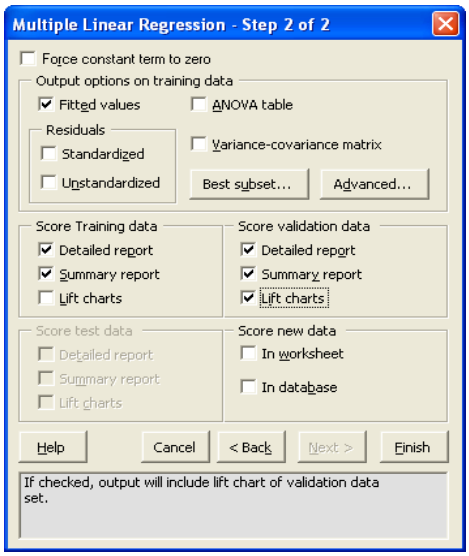


Figure 2.8: Specifying the Output

negative errors balance each other out. It tells us nothing about how large those positive and negative errors are.

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

The "RMS error" or root mean squared error is perhaps the most useful term of all. It takes the square root of the average squared error, and so it gives an idea of the typical error (whether positive or negative) in the same scale as the original data.

As we might expect, the RMS error for the validation data ($5,337), which the model is seeing for the first time in making these predictions, is larger than for the training data ($4,518), which were used in training the model.

8. Interpret the results.

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

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

XLMiner : Multiple Linear Regression - Prediction of Training Data

Data range	['Boston_Housing.xls']'Data_Partition1'!$C$19:$P$322	Back to Navigator

Row Id.	Predicted Value	Actual Value	Residual	CRIM	ZN	INDUS	CHAS	NOX
1	30.24690555	24	-6.246905549	0.00632	18	2.31	0	0.538
4	28.61652272	33.4	4.783477282	0.03237	0	2.18	0	0.458
5	27.76434086	36.2	8.435659135	0.06905	0	2.18	0	0.458
6	25.6204032	28.7	3.079596801	0.02985	0	2.18	0	0.458
9	11.54583087	16.5	4.954169128	0.21124	12.5	7.87	0	0.524
10	19.13566187	18.9	-0.235661871	0.17004	12.5	7.87	0	0.524
12	21.95655773	18.9	-3.05655773	0.11747	12.5	7.87	0	0.524
17	20.80054199	23.1	2.299458015	1.05393	0	8.14	0	0.538
18	16.94685562	17.5	0.553144385	0.7842	0	8.14	0	0.538
19	16.68387738	20.2	3.516122619	0.80271	0	8.14	0	0.538

Figure 2.9: Predictions for the Training Data

XLMiner : Multiple Linear Regression - Prediction of Validation Data

Data range	['Boston_Housing.xls']'Data_Partition1'!$C$323:$P$524	Back to Navigator

Row Id.	Predicted Value	Actual Value	Residual	CRIM	ZN	INDUS	CHAS	NOX
2	25.03555247	21.6	-3.435552468	0.02731	0	7.07	0	0.469
3	30.1845219	34.7	4.515478101	0.02729	0	7.07	0	0.469
7	23.39322259	22.9	-0.493222593	0.08829	12.5	7.87	0	0.524
8	19.58824389	27.1	7.511756109	0.14455	12.5	7.87	0	0.524
11	18.83048747	15	-3.830487466	0.22489	12.5	7.87	0	0.524
13	21.20113865	21.7	0.498861352	0.09378	12.5	7.87	0	0.524
14	19.81376359	20.4	0.586236414	0.62976	0	8.14	0	0.538
15	19.42217211	18.2	-1.222172107	0.63796	0	8.14	0	0.538
16	19.63108414	19.9	0.268915856	0.62739	0	8.14	0	0.538

Figure 2.10: Predictions for the Validation Data

Training Data scoring - Summary Report

Total sum of squared errors	RMS Error	Average Error
6977.106	4.790720883	3.11245E-07

Validation Data scoring - Summary Report

Total sum of squared errors	RMS Error	Average Error
4251.582211	4.587748542	-0.011138034

Figure 2.11: Error Rates for Training and Validation Data

2.7 Using Excel For Data Mining

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

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

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

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

Data Mining Software Tools: The State of the Market
by Herb Edelstein[1]

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

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

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

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

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

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

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

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

Data Mining Software Tools: The State of the Market - cont.

Stand-alone data mining tools can be based on a single algorithm or a collection of algorithms called a suite. Target users include both statisticians and analysts. Well-known single-algorithm products include KXEN; RuleQuest Research C5.0; and Salford Systems CART, MARS and Treenet. Most of the top single-algorithm tools have also been licensed to suite vendors. The leading suites include SAS Enterprise Miner; SPSS Clementine; and Insightful Miner. Suites are characterized by providing a wide range of functionality and an interface designed to enhance model-building productivity. Many suites have outstanding visualization tools and links to statistical packages that extend the range of tasks they can perform, and most provide a procedural scripting language for more complex transformations. They use a graphical workflow interface to outline the entire data mining process. The suite vendors are working to link their tools more closely to underlying DBMS's; for example, data transformations might be handled by the DBMS. Data mining models can be exported to be incorporated into the DBMS either through generating SQL, procedural language code (e.g., C++ or Java), or a standardized data mining model language called Predictive Model Markup Language (PMML).
Application-specific tools, in contrast to the other types, are intended for particular analytic applications such as credit scoring, customer retention, or product marketing. Their focus may be further sharpened to address the needs of certain markets such as mortgage lending or financial services. The target user is an analyst with expertise in the applications domain. Therefore the interfaces, the algorithms, and even the terminology are customized for that particular industry, application, or customer. While less flexible than general-purpose tools, they offer the advantage of already incorporating domain knowledge into the product design, and can provide very good solutions with less effort. Data mining companies including SAS and SPSS offer vertical market tools, as do industry specialists such as Fair Isaac.
The tool used in this book, XLMiner, is a suite with both sampling and scoring capabilities. While Excel itself is not a suitable environment for dealing with thousands of columns and millions of rows, it is a familiar workspace to business analysts and can be used as a work platform to support other tools. An Excel add-in such as XLMiner (which uses non-Excel computational engines) is user-friendly and can be used in conjunction with sampling techniques for prototyping, small-scale and educational applications of data mining.

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

2.8 Exercises

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

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

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

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

   (d) Identifying segments of similar customers.

   (e) Predicting whether a company will go bankrupt, based on comparing its financial data to similar bankrupt and non-bankrupt firms.

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

   (g) Automated sorting of mail by zip code scanning.

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

2. In Figure 2.2, locate the plot of the crime rate vs. median value, and interpret it.

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

4. Consider the sample from a database of credit applicants in Figure 2.12. Comment on the likelihood that it was sampled randomly, and whether it is likely to be a useful sample.

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

Figure 2.12: Sample from a Database of Credit Applicants

5. Consider the sample from a bank database shown in Figure 2.13; it was selected randomly from a larger database to be the training set. "Personal loan" indicates whether a solicitation for a personal loan was accepted and is the response variable. A campaign is planned for a similar solicitation in the future and the bank is looking for a model that will identify likely responders. Examine the data carefully and indicate what your next step would be.

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

Figure 2.13: Sample from a Bank Database

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

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

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

9. Normalize the data in the following table, showing calculations:

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

Statistical distance between records can be measured in several ways. Consider Euclidean distance, measured as the square route of the sum of the squared differences. For the first two records above it is:

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

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

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

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

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

    (b) We plan to analyze the data using various data mining techniques to be covered in future chapters. Prepare the data for use as follows:

        i. The dataset has two categorical attributes, *Fuel_Type(3)* and *Color(10).*
        – Describe how you would convert these to binary variables.
        – Confirm this using XLMiner's utility to transform categorical data into dummies.
        – How would you work with these new variables to avoid including redundant information in models?

        ii. Prepare the dataset (as factored into dummies) for data mining techniques of supervised learning by creating partitions using XLMiner's data partitioning utility. Select all the variables and use default values for the random seed and partitioning percentages for training (50%), validation (30%) and test (20%) sets. Describe the roles that these partitions will play in modeling.

Chapter 3

Data Exploration and Dimension Reduction

3.1 Introduction

In data mining one often encounters situations where there are a large number of variables in the database. In such situations it is very likely that subsets of variables are highly correlated with each other. Including highly correlated variables, or variables that are unrelated to the outcome of interest, in a classification or prediction model can lead to overfitting, and accuracy and reliability can suffer. Large numbers of variables also pose computational problems for some models (aside from questions of correlation.) In model deployment, superfluous variables can increase costs due to collection and processing of these variables.

The "dimensionality" of a model is the number of independent or input variables used by the model. One of the key steps in data mining, therefore, is finding ways to reduce dimensionality without sacrificing accuracy.

3.2 Practical Considerations

Although data mining prefers automated methods over domain knowledge, it is important at the first step of data exploration to make sure that the measured variables are reasonable for the task at hand. The integration of expert knowledge through a discussion with the data provider (or user) will most likely lead to better results. Practical considerations include: Which variables are most important and which are most likely to be useless for the task at hand? Which variables are likely to contain much error? Which variables will be available for measurement (and their cost feasibility) in the future, if the analysis will be repeated? Which variables can actually be measured before the outcome occurs? (for example, if we want to predict the closing price of an auction, we cannot use the number of bids as a predictor, because this is unknown until the auction closes).

Example 1: House Prices in Boston

We return to the Boston housing example introduced in Chapter **??** on housing-related characteristics in different neighborhoods in Boston. For each neighborhood, a number of variables are given such as the crime rate, the student/teacher ratio, and the median value of a housing unit in the neighborhood. A description of the complete 14 variables is given in Table 3.1. The first 10 records of the data are shown in Figure 3.1.

Table 3.1: Description of Variables in Boston Housing Dataset

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

	A	B	C	D	E	F	G	H	I	J	K	L	M	N
	CRIM	ZN	INDUS	CHAS	NOX	RM	AGE	DIS	RAD	TAX	PTRATIO	B	LSTAT	MEDV
1	0.006	18	2.31	0	0.54	6.58	65.2	4.09	1	296	15.3	397	5	24
2	0.027	0	7.07	0	0.47	6.42	78.9	4.97	2	242	17.8	397	9	21.6
3	0.027	0	7.07	0	0.47	7.19	61.1	4.97	2	242	17.8	393	4	34.7
4	0.032	0	2.18	0	0.46	7.00	45.8	6.06	3	222	18.7	395	3	33.4
5	0.069	0	2.18	0	0.46	7.15	54.2	6.06	3	222	18.7	397	5	36.2
6	0.030	0	2.18	0	0.46	6.43	58.7	6.06	3	222	18.7	394	5	28.7
7	0.088	12.5	7.87	0	0.52	6.01	66.6	5.56	5	311	15.2	396	12	22.9
8	0.145	12.5	7.87	0	0.52	6.17	96.1	5.95	5	311	15.2	397	19	27.1
9	0.211	12.5	7.87	0	0.52	5.63	100	6.08	5	311	15.2	387	30	16.5
10	0.170	12.5	7.87	0	0.52	6.00	85.9	6.59	5	311	15.2	387	17	18.9

Figure 3.1: The First 10 Records in The Boston Housing Dataset

The first row in this figure represents the first neighborhood, which had an average per capita crime rate of .006, 18% of the residential land zoned for lots over 25,000 square feet, 2.31% of the land devoted to non-retail business, no border on the Charles River, etc.

3.3 Data Summaries

The first step in data analysis is data exploration. This means getting familiar with the data and their characteristics through summaries and graphs. The importance of this step cannot be overstated. The better you understand your data, the better the results from the modeling or mining process.

Excel has several functions and facilities that assist in summarizing data. The functions *average*, *stdev*, *min*, *max*, *median*, and *count* are very helpful for learning about the characteristics of each variable. First, they give us information about the scale and type of values that the variable takes. The min and max functions can be used to detect extreme values that might be errors. The average and median give a sense of the central values of that variable, and a large deviation between the two

Table 3.2: Correlation Matrix for a Subset of the Boston Housing Variables

	PTRATIO	*B*	*LSTAT*	*MEDV*
PTRATIO	1			
B	-0.17738	1		
LSTAT	0.374044	-0.36609	1	
MEDV	-0.50779	0.333461	-0.73766	1

also indicates skew. The standard deviation (relative to the mean) gives a sense of how dispersed the data are. Other functions such as *countblank*, which give the number of empty cells, can tell us about missing values. It is also possible to use Excel's *DescriptiveStatistics* facility in the *Tool* > *Data Analysis* menu. This will generate a set of 13 summary statistics for each of the variables.

Figure 3.2 shows six summary statistics for the Boston Housing example. We immediately see

	Average	Median	Min	Max	Std	Count	Countblank
CRIM	3.61	0.26	0.01	88.98	8.60	506	0
ZN	11.36	0.00	0.00	100.00	23.32	506	0
INDUS	11.14	9.69	0.46	27.74	6.86	506	0
CHAS	0.07	0.00	0.00	1.00	0.25	506	0
NOX	0.55	0.54	0.39	0.87	0.12	506	0
RM	6.28	6.21	3.56	8.78	0.70	506	0
AGE	68.57	77.50	2.90	100.00	28.15	506	0
DIS	3.80	3.21	1.13	12.13	2.11	506	0
RAD	9.55	5.00	1.00	24.00	8.71	506	0
TAX	408.24	330.00	187.00	711.00	168.54	506	0
PTRATIO	18.46	19.05	12.60	22.00	2.16	506	0
B	356.67	391.44	0.32	396.90	91.29	506	0
LSTAT	12.65	11.36	1.73	37.97	7.14	506	0
MEDV	22.53	21.20	5.00	50.00	9.20	506	0

Figure 3.2: Summary Statistics for the Boston Housing Data

that the different variables have very different ranges of values. We will see soon how variation in scale across variables can distort analyses if not properly treated. Another observation that can be made is that the first variable CRIM (as well as several others) has an average that is much larger than the median, indicating right skew. None of the variables have empty cells. There also do not appear to be indications of extreme values that might result from typing errors.

Next, we summarize relationships between two or more variables. For numeric variables, we can compute pairwise correlations (using the Excel function *correl*). We can also obtain a complete matrix of correlations between each pair of variables in the data using Excel's *Correlation* facility in the *Tools* > *DataAnalysis* menu. Table 3.2 shows the correlation matrix for a subset of the Boston Housing variables. We see that overall the correlations are not very strong, and that all are negative except for the correlation between LSAT and PTRATIO and between MEDV and B. We will return to the importance of the correlation matrix soon, in the context of correlation analysis.

Another very useful tool is Excel's *pivot tables* (in the *Data* menu). These are interactive tables that can combine information from multiple variables and compute a range of summary statistics (count, average, percentage, etc.). A simple example is the average MEDV for neighborhoods that

bound the Charles river vs. those that do not (the variable CHAS is chosen as the column area). This is shown in the top panel of Figure 3.3. It appears that the majority of neighborhoods (471 of 506) do not bound the river. By double-clicking on a certain cell, the complete data for records in that cell are shown on a new worksheet. For instance, double-clicking on the cell containing 471 will display the complete records of neighborhoods that do not bound the river.

Pivot tables can be used for multiple variables. For categorical variables we obtain a breakdown of the records by the combination of categories. For instance, the bottom panel of Figure 3.3 shows the average MEDV by CHAS (row) and RM (column). Notice that the numerical variable RM (the average number of rooms per dwelling in the neighborhood) is grouped into bins of 3-4, 5-6, etc. Notice also the empty cells denoting that there are no neighborhoods in the dataset with those combinations (e.g., bounding the river and having on average 3-4 rooms).

Count of MEDV	
CHAS	Total
0	471
1	35
Grand Total	506

Average of MEDV	CHAS		
RM	0	1	Grand Total
3-4	25.3		25.3
4-5	16.023077		16.02307692
5-6	17.133333	22.21818182	17.48734177
6-7	21.76917	25.91875	22.01598513
7-8	35.964444	44.06666667	36.91764706
8-9	45.7	35.95	44.2
Grand Total	22.093843	28.44	22.53280632

Figure 3.3: Pivot Tables for the Boston Housing Data

There are many more possibilities and options for using Excel's pivot tables. We leave it to the reader to explore these using Excel's documentation.

In classification tasks, where the goal is to find predictor variables that separate well between two classes, a good exploratory step is to produce summaries for each class. This can assist in detecting useful predictors that indeed display some separation between the two classes. Data summaries are useful for almost any data mining task, and are therefore an important preliminary step for cleaning and understanding the data before carrying out further analyses.

3.4 Data Visualization

Another powerful exploratory analysis approach is to examine graphs and plots of the data. For single numerical variables we can use histograms and boxplots to display their distribution. For categorical variables we use bar charts (and to a lesser degree, pie charts). Figure 3.4 shows a

histogram of MEDV (top panel) and a side-by-side boxplot of MEDV for river-bound vs. non-river-bound neighborhoods (bottom panel). The histogram shows the skewness of MEDV, with a concentration of MEDV values around 20-25. The boxplots show that neighborhoods that are river bound tend to be pricier.

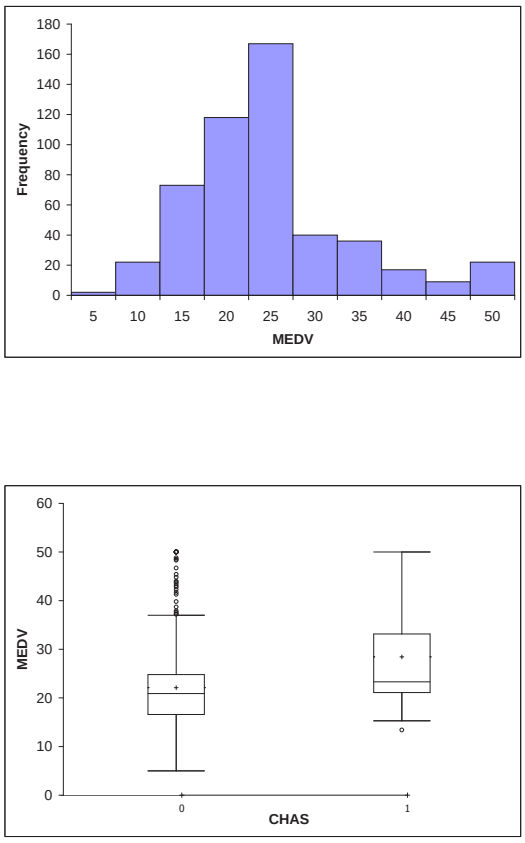


Figure 3.4: Histogram (Top) And Side-By-Side Boxplots (Bottom) for MEDV

Scatterplots are very useful for displaying relationships between numerical variables. They are also good for detecting patterns and outliers. Like the correlation matrix, we can examine multiple scatter plots at once by combining all possible scatterplots between a set of variables on a single page. This is called a "matrix plot", and it allows us to quickly visualize relationships among the many variables.

Figure 3.5 displays a matrix plot for four variables from the Boston Housing dataset. In the lower left, for example, the crime rate (CRIM) is plotted on the x-axis and the median value (MEDV) on the y-axis. In the upper right, the same two variables are plotted on opposite axes. From the scatterplots in the lower right quadrant, we see that, unsurprisingly, the more lower economic status residents a neighborhood has, the lower the median house value. From the upper right and lower left corners we see (again, unsurprisingly) that higher crime rates are associated with lower median values. An interesting result can be seen in the upper left quadrant. All the very high crime rates seem to be associated with a specific, mid-range value of INDUS (proportion of non-retail businesses

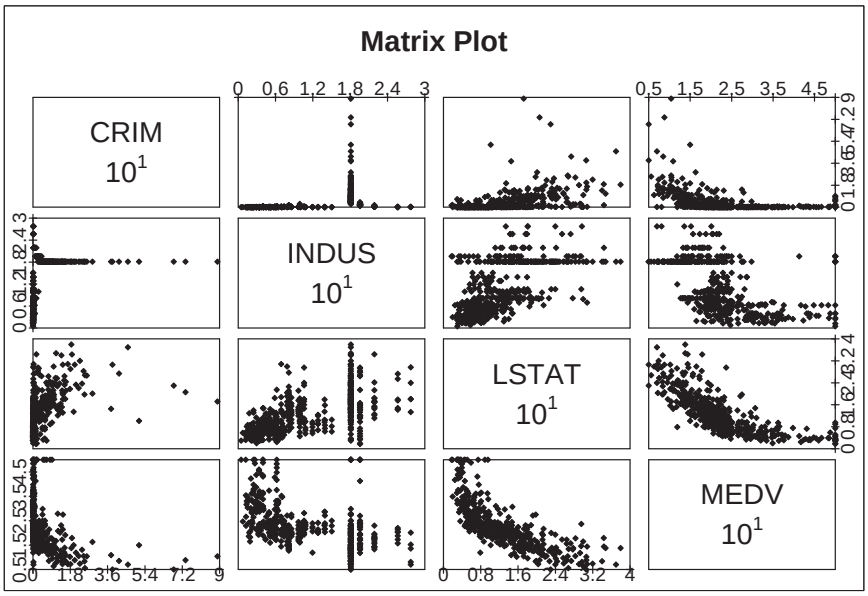


Figure 3.5: Matrix Scatterplot For Four Variables from the Boston Housing Data

per neighborhood). It seems dubious that a specific, middling level of INDUS is really associated with high crime rates. A closer examination of the data reveals that each specific value of INDUS is shared by a number of neighborhoods, indicating that INDUS is measured for a broader area than that of the census tract neighborhood. The high crime rate associated so markedly with a specific value of INDUS indicates that the few neighborhoods with extremely high crime rates fall mainly within one such broader area.

Of course, with a huge dataset, it might be impossible to compute or even to generate usual plots. For instance, a scatterplot of one million points is likely to be unreadable. A solution is to draw a random sample from the data and use it to generate visualizations.

Finally, it is always more powerful to use interactive visualization over static plots. Software packages for interactive visualization (such as Spotfire, www.spotfire.com) allow the user to select and change variables on the plots interactively; zoom-in and zoom-out of different areas on the plot; and generally empower the user to better navigate in the sea of data.

3.5 Correlation Analysis

In datasets with a large number of variables (that are likely to serve as predictors), there is usually much overlap in the information covered by the set of variables. One simple way to find redundancies is to look at a correlation matrix. This shows all the pairwise correlations between the variables. Pairs that have a very strong (positive or negative) correlation contain a lot of overlap in information and are good candidates for data reduction by removing one of the variables. Removing variables that are strongly correlated to others is useful for avoiding multicollinearity problems that can arise in various models (Multicollinearity is the presence of two or more predictors sharing the same linear

relationship with the outcome variable.) This is also a good method to find variable duplications in the data: sometimes the same variable accidently appears more than once in the dataset (under a different name) because the dataset was merged from multiple sources, the same phenomenon is measured in different units, etc. Using color to encode the correlation magnitude in the correlation matrix can make the task of identifying strong correlations easier.

3.6 Reducing the Number of Categories in Categorical Variables

When a categorical variable has many categories, and this variable is destined to be a predictor, it will result in many dummy variables. In particular, a variable with m categories will be transformed into variables $m-1$ dummy variables when used in an analysis. This means that even if we have very few original categorical variables, they can greatly inflate the dimension of the dataset. One way to handle this is to reduce the number of categories by binning close bins together. This requires incorporating expert knowledge and common sense. Pivot tables are useful for this task: we can examine the sizes of the different categories and how the response behaves at each category. Generally, bins that contain very few observations are good candidates for combining with other categorie+s. Use only the categories that are most relevant to the analysis, and label the rest as "other."

3.7 Principal Components Analysis

Principal components analysis (PCA) is a useful procedure for reducing the number of predictors in the model by analyzing the input variables. It is especially valuable when we have subsets of measurements that are measured on the same scale and are highly correlated. In that case it provides a few variables (often as few as three) that are weighted linear combinations of the original variables that retain the explanatory power of the full original set. PCA is intended for use with quantitative variables. For categorical variables, other methods, such as correspondence analysis, are more suitable.

3.7.1 Example 2: Breakfast Cereals

Data were collected on the nutritional information and consumer rating of 77 breakfast cereals[1]. For each cereal the data include 13 numerical variables, and we are interested in reducing this dimension.

For each cereal the information is based on a bowl of cereal rather than a serving size, because most people simply fill a cereal bowl (resulting in constant volume, but not weight). A snapshot of these data is given in Figure 3.6, and the description of the different variables is given in Table 3.3.

We focus first on two variables: Calories and Consumer Rating. These are given in Table 3.4. The average calories across the 75 cereals is 106.88 and the average consumer rating is 42.67. The estimated covariance matrix between the two variables is $S = \begin{bmatrix} 379.63 & -188.68 \\ -188.68 & 197.32 \end{bmatrix}$.

It can be seen that the two variables are strongly correlated with a negative correlation of $-0.69 = \frac{-188.68}{\sqrt{(379.63)(197.32)}}$. This means that there is redundancy in the information that the two variables contain, so it might be possible to reduce the two variables into a single one without losing too much information. The idea in PCA is to find a linear combination of the two variables that contains most of the information, even if not all of it, so that this new variable can replace the two original variables. Information here is in the sense of variability: what can explain the most variability *among* the 77 cereals? The total variability here is the sum of the variances of the

[1]The data are available at http://lib.stat.cmu.edu/DASL/Stories/HealthyBreakfast.html

Cereal Name	mfr	type	calories	protein	fat	sodium	fiber	carbo	sugars	potass	vitamins	shelf	weight	cups	rating
100% Bran	N	cold	70	4	1	130	10	5	6	280	25	3	1	0.33	68.40297
100% Natural Bran	Q	cold	120	3	5	15	2	8	8	135	0	3	1	1	33.98368
All-Bran	K	cold	70	4	1	260	9	7	5	320	25	3	1	0.33	59.42551
All-Bran with Extra Fiber	K	cold	50	4	0	140	14	8	0	330	25	3	1	0.5	93.70491
Almond Delight	R	cold	110	2	2	200	1	14	8		25	3	1	0.75	34.38484
Apple Cinnamon Cheerios	G	cold	110	2	2	180	1.5	10.5	10	70	25	1	1	0.75	29.50954
Apple Jacks	K	cold	110	2	0	125	1	11	14	30	25	2	1	1	33.17409
Basic 4	G	cold	130	3	2	210	2	18	8	100	25	3	1.33	0.75	37.03856
Bran Chex	R	cold	90	2	1	200	4	15	6	125	25	1	1	0.67	49.12025
Bran Flakes	P	cold	90	3	0	210	5	13	5	190	25	3	1	0.67	53.31381
Cap'n'Crunch	Q	cold	120	1	2	220	0	12	12	35	25	2	1	0.75	18.04285
Cheerios	G	cold	110	6	2	290	2	17	1	105	25	1	1	1.25	50.765
Cinnamon Toast Crunch	G	cold	120	1	3	210	0	13	9	45	25	2	1	0.75	19.82357
Clusters	G	cold	110	3	2	140	2	13	7	105	25	3	1	0.5	40.40021
Cocoa Puffs	G	cold	110	1	1	180	0	12	13	55	25	2	1	1	22.73645
Corn Chex	R	cold	110	2	0	280	0	22	3	25	25	1	1	1	41.44502
Corn Flakes	K	cold	100	2	0	290	1	21	2	35	25	1	1	1	45.86332
Corn Pops	K	cold	110	1	0	90	1	13	12	20	25	2	1	1	35.78279
Count Chocula	G	cold	110	1	1	180	0	12	13	65	25	2	1	1	22.39651
Cracklin' Oat Bran	K	cold	110	3	3	140	4	10	7	160	25	3	1	0.5	40.44877

Figure 3.6: Sample from the 77 Breakfast Cereal Dataset

Table 3.3: Description of the Variables in the Breakfast Cereals Dataset

Variable	Description
mfr	Manufacturer of cereal (American Home Food Products, General Mills, Kelloggs, etc.)
type	Cold or hot
calories	Calories per serving
protein	Grams of protein
fat	Grams of fat
sodium	Milligrams of sodium
fiber	Grams of dietary fiber
carbo	Grams of complex carbohydrates
sugars	Grams of sugars
potass	Milligrams of potassium
vitamins	Vitamins and minerals - 0, 25, or 100, Indicating the typical percentage of FDA recommended
shelf	Display shelf (1, 2, or 3, counting from the floor)
weight	Weight in ounces of one serving
cups	Number of cups in one serving
rating	A rating of the cereal calculated by Consumer Reports

Table 3.4: Cereal Calories and Ratings

Cereal	Calories	Rating	Cereal	Calories	Rating
100% Bran	70	68.40297	Just Right Fruit & Nut	140	36.471512
100% Natural Bran	120	33.98368	Kix	110	39.241114
All-Bran	70	59.42551	Life	100	45.328074
All-Bran with Extra Fiber	50	93.70491	Lucky Charms	110	26.734515
Almond Delight	110	34.38484	Maypo	100	54.850917
Apple Cinnamon Cheerios	110	29.50954	Muesli Raisins, Dates & Almonds	150	37.136863
Apple Jacks	110	33.17409	Muesli Raisins, Peaches & Pecans	150	34.139765
Basic 4	130	37.03856	Mueslix Crispy Blend	160	30.313351
Bran Chex	90	49.12025	Multi-Grain Cheerios	100	40.105965
Bran Flakes	90	53.31381	Nut&Honey Crunch	120	29.924285
Cap'n'Crunch	120	18.04285	Nutri-Grain Almond-Raisin	140	40.69232
Cheerios	110	50.765	Nutri-grain Wheat	90	59.642837
Cinnamon Toast Crunch	120	19.82357	Oatmeal Raisin Crisp	130	30.450843
Clusters	110	40.40021	Post Nat. Raisin Bran	120	37.840594
Cocoa Puffs	110	22.73645	Product 19	100	41.50354
Corn Chex	110	41.44502	Puffed Rice	50	60.756112
Corn Flakes	100	45.86332	Puffed Wheat	50	63.005645
Corn Pops	110	35.78279	Quaker Oat Squares	100	49.511874
Count Chocula	110	22.39651	Quaker Oatmeal	100	50.828392
Cracklin' Oat Bran	110	40.44877	Raisin Bran	120	39.259197
Cream of Wheat (Quick)	100	64.53382	Raisin Nut Bran	100	39.7034
Crispix	110	46.89564	Raisin Squares	90	55.333142
Crispy Wheat & Raisins	100	36.1762	Rice Chex	110	41.998933
Double Chex	100	44.33086	Rice Krispies	110	40.560159
Froot Loops	110	32.20758	Shredded Wheat	80	68.235885
Frosted Flakes	110	31.43597	Shredded Wheat 'n'Bran	90	74.472949
Frosted Mini-Wheats	100	58.34514	Shredded Wheat spoon size	90	72.801787
Fruit & Fibre Dates, Walnuts & Oats	120	40.91705	Smacks	110	31.230054
Fruitful Bran	120	41.01549	Special K	110	53.131324
Fruity Pebbles	110	28.02577	Strawberry Fruit Wheats	90	59.363993
Golden Crisp	100	35.25244	Total Corn Flakes	110	38.839746
Golden Grahams	110	23.80404	Total Raisin Bran	140	28.592785
Grape Nuts Flakes	100	52.0769	Total Whole Grain	100	46.658844
Grape-Nuts	110	53.37101	Triples	110	39.106174
Great Grains Pecan	120	45.81172	Trix	110	27.753301
Honey Graham Ohs	120	21.87129	Wheat Chex	100	49.787445
Honey Nut Cheerios	110	31.07222	Wheaties	100	51.592193
Honey-comb	110	28.74241	Wheaties Honey Gold	110	36.187559
Just Right Crunchy Nuggets	110	36.52368			

two variables, which in this case is $379.63 + 197.32 = 577$. This means that Calories account for $66\% = 379.63/577$ of the total variability, and Rating for the remaining 44%. If we drop one of the variables for the sake of dimension reduction, then we lose at least 44% of the total variability. Can we redistribute the total variability between two new variables in a more polarized way? If so, then it might be possible to keep only the one new variable that accounts for (hopefully) a large portion of the total variation.

Figure 3.7 shows the scatter plot of Rating vs. Calories. The line z_1 is the direction in which the variability of the points is largest. It is the line that captures the most variation in the data if we decide to reduce the dimensionality of the data from two to one. Among all possible lines, it is the line for which, if we project the points in the dataset orthogonally to get a set of 77 (one dimensional) values, the variance of the z_1 values will be maximum. This is called the *first principal component*. It is also the line that minimizes the sum of squared perpendicular distances from the line. The z_2 axis is chosen to be perpendicular to the z_1 axis. In the case of two variables there is only one line that is perpendicular to z_1, and it has the second largest variability, but its information is uncorrelated with z_1. This is called the *second principal component*. In general, when we have more than two variables, once we find the direction z_1 with the largest variability, we search among all the orthogonal directions to z_1 for the one with the next highest variability. That is z_2. The idea is then to find the coordinates of these lines, and to see how they redistribute the variability.

Figure 3.8 shows the XLMiner output from running PCA on these two variables. The Principal Components table gives the weights that are used to project the original points onto the two new directions. The weights for z_1 are given by (-0.847, 0.532), and for z_2 they are given by (0.532, 0.847). The table below gives the new reallocated variation: z_1 accounts for 86% of the total variability and the remaining 14% are accounted by z_2. Therefore, if we drop z_2 we still maintain 88% of the total variability. The weights are used to compute principal component scores, which are the projected values of Calories and Rating onto the new axes (after subtracting the means). Figure 11.4 shows the scores for the two dimensions. The first column is the projection onto z_1 using the weights (-0.847, 0.532). The second column is the projection onto z_2 using the weights (0.532, 0.847). For instance, the first score for the 100% Bran cereal (with 70 calories and rating of 68.4) is

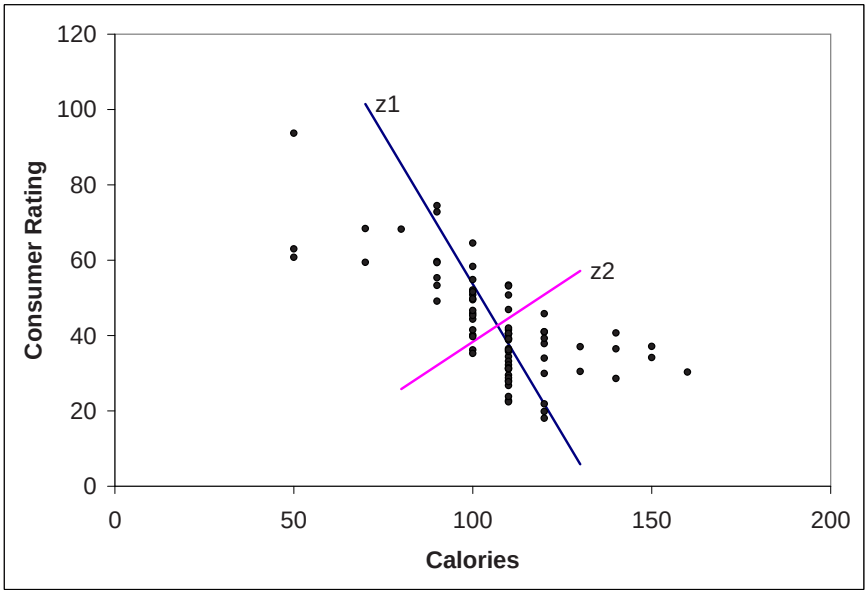


Figure 3.7: Scatterplot of Consumer Rating Vs. Calories for 77 Breakfast Cereals, With the Two Principal Component Directions

Principal Components

	Components	
Variable	1	2
calories	-0.84705347	0.53150767
rating	0.53150767	0.84705347
Variance	498.0244751	78.932724
Variance%	86.31913757	13.68086338
Cum%	86.31913757	100
P-value	0	1

Figure 3.8: Output from Principal Components Analysis of Calories and Ratings

XLMiner : Principal Components Analysis - Scores

Row Id.	1	2
100% Bran	44.92152786	2.19717932
100% Natural Bran	-15.7252636	-0.38241446
All-Bran	40.14993668	-5.40721178
All-Bran with Extra Fiber	75.31076813	12.99912071
Almond Delight	-7.04150867	-5.35768652
Apple Cinnamon Cheerios	-9.63276863	-9.48732758
Apple Jacks	-7.68502998	-6.38325357
Basic 4	-22.57210541	7.52030993
Bran Chex	17.7315464	-3.50615811
Bran Flakes	19.96045494	0.04600986
Cap'n'Crunch	-24.19793701	-13.88514996
Cheerios	1.66467071	8.5171833
Cinnamon Toast Crunch	-23.25147057	-12.37678337
Clusters	-3.84429598	-0.26235023
Cocoa Puffs	-13.23272038	-15.2244997
Corn Chex	-3.28897071	0.62266076
Corn Flakes	7.5299263	-0.94987571

Figure 3.9: Principal Scores from Principal Components Analysis of Calories and Ratings for the First 17 Cereals

$(-0.847)(70 - 106.88) + (0.532)(68.4 - 42.67) = 44.92$.

Notice that the means of the new variables z_1 and z_2 are zero (because we've subtracted the mean of each variable). The sum of the variances $var(z_1) + var(z_2)$ is equal to the sum of the variances of the original variables, Calories and Rating. Furthermore, the variances of z_1 and z_2 are 498 and 79 respectively, so the first principal component, z_1, accounts for 86% of the total variance. Since it captures most of the variability in the data, it seems reasonable to use one variable, the first principal score, to represent the two variables in the original data.

We will now generalize these ideas to more than two variables.

3.7.2 The Principal Components

Let us formalize the above procedure, so that it can be easily generalized to $p > 2$ variables. Denote by $X_1, X_2, \ldots, X_p$ the original p variables. In PCA we are looking for a set of new variables $Z_1, Z_2, \ldots, Z_p$ that are weighted averages of the original variables (after subtracting their mean):

$$Z_i = a_{i,1}(X_1 - \bar{X}_1) - +a_{i,2}(X_2 - \bar{X}_2) + \cdots + a_{i,p}(X_p - \bar{X}_p) \quad i = 1, \ldots, p,$$

where each pair of Z's has correlation =0. We then order the resulting Z's by their variance, with Z_1 having the largest variance and Z_p having the smallest variance. The software computes the weights $a_{i,j}$ that are then used for computing the principal component scores.

A further advantage of the principal components compared to the original data is that they are uncorrelated (correlation coefficient = 0). If we construct regression models using these principal components as independent variables, we will not encounter problems of multicollinearity.

Let us return to the breakfast cereal dataset with all 15 variables, and apply PCA to the 13 numerical variables. The resulting output is shown in Figure 3.10. For simplicity, we removed three cereals that contained missing values. Notice that the first three components account for more than

Variable	1	2	3	4	5
calories	0.07798425	-0.00931156	0.62920582	-0.60102159	0.45495847
protein	-0.00075678	0.00880103	0.00102611	0.00319992	0.05617596
fat	-0.00010178	0.00269915	0.01619579	-0.02526222	-0.01609845
sodium	0.98021454	0.14089581	-0.13590187	-0.00096808	0.01394816
fiber	-0.00541276	0.03068075	-0.01819105	0.0204722	0.01360502
carbo	0.01724625	-0.0167833	0.01736996	0.02594825	0.34926692
sugars	0.00298888	-0.00025348	0.09770504	-0.11548097	-0.29906642
potass	-0.13490002	0.98656207	0.03678251	-0.0421758	-0.04715054
vitamins	0.09429332	0.01672884	0.69197786	0.714118	-0.03700861
shelf	-0.00154142	0.0043604	0.01248884	0.00564718	-0.00787646
weight	0.000512	0.00099922	0.00380597	-0.00254643	0.00302211
cups	0.00051012	-0.00159098	0.00069433	0.00098539	0.00214846
rating	-0.07529629	0.07174215	-0.30794701	0.33453393	0.75770795
Variance	7016.42041	5028.831543	512.7391968	367.9292603	70.95076752
Variance%	53.95025635	38.66740417	3.94252491	2.82906055	0.54555058
Cum%	53.95025635	92.61766052	96.56018829	99.38924408	99.93479919
P-value					

Figure 3.10: PCA Output Using All 13 Numerical Variables in The Breakfast Cereals Dataset. The Table Gives Results for the First Five Principal Components

96% of the total variation associated with all 13 of the original variables. This suggests that we can capture most of the variability in the data with less than 25% of the number of original dimensions in the data. In fact, the first two principal components alone capture 92.6% of the total variation. However, these results are influenced by the scales of the variables, as we describe next.

3.7.3 Normalizing the Data

A further use of PCA is to understand the structure of the data. This is done by examining the weights to see how the original variables contribute to the different principal components. In our example, it is clear that the first principal component is dominates by the sodium content of the cereal: it has the highest (in this case positive) weight. This means that the first principal component is measuring how much sodium is in the cereal. Similarly, the second principal component seems to be measuring the amount of potassium. Since both these variables are measured in milligrams whereas the other nutrients are measured in grams, the scale is obviously leading to this result. The variances of potassium and sodium are much larger than the variances of the other variables, and thus the total variance is dominated by these two variances. A solution is to normalize the data before performing the PCA. Normalization (or standardization) means replacing each original variable by a standardized version of the variable that has unit variance. This is easily accomplished by dividing each variable by its standard deviation. The effect of this normalization (standardization) is to give all variables equal importance in terms of the variability.

When should we normalize the data like this? It depends on the nature of the data. When the units of measurement are common for the variables (e.g. dollars), and when their scale reflects their importance (sales of jet fuel, sales of heating oil), it is probably best not to normalize (i.e., not to rescale the data so that it has unit variance). If the variables are measured in quite differing units so that it is unclear how to compare the variability of different variables (e.g. dollars for some, parts per million for others) or if, for variables measured in the same units, scale does not reflect

Variable	1	2	3	4	5	6	7	8
calories	0.2995424	0.39314792	0.11485746	0.20435865	0.20389892	-0.25590625	-0.02559552	-0.0024775
protein	-0.30735639	0.16532333	0.27728197	0.30074316	0.319749	0.120752	0.28270504	-0.42663196
fat	0.03991544	0.34572428	-0.20489009	0.18683317	0.58689332	0.34796733	-0.05115468	0.06305054
sodium	0.18339655	0.13722059	0.38943109	0.12033724	-0.33836424	0.66437215	-0.28370309	0.17672044
fiber	-0.45349041	0.17981192	0.06976604	0.03917367	-0.255119	0.0642436	0.11232537	0.21621555
carbo	0.19244903	-0.14944831	0.56245244	0.0878355	0.18274252	-0.32639283	-0.26046798	0.16743632
sugars	0.22806853	0.35143444	-0.35540518	-0.02270711	-0.31487244	-0.15208226	0.22798519	-0.06308819
potass	-0.40196434	0.30054429	0.06762024	0.09087842	-0.14836049	0.02515389	0.14880823	0.26222241
vitamins	0.11598022	0.1729092	0.38785872	-0.6041106	-0.04928682	0.12948574	0.29427618	-0.45704079
shelf	-0.17126338	0.26505029	-0.00153102	-0.63887852	0.32910112	-0.05204415	-0.17483434	0.41414571
weight	0.05029929	0.45030847	0.24713831	0.15342878	-0.22128329	-0.39877367	0.01392053	0.07524765
cups	0.29463556	-0.21224795	0.13999969	0.04748911	0.12081645	0.09946091	0.74856687	0.49895892
rating	-0.43837839	-0.25153893	0.1818424	0.0383162	0.05758421	-0.18614525	0.06344455	0.01494502
Variance	3.63360572	3.1480546	1.90934956	1.01947618	0.98935974	0.72206175	0.67151642	0.4162229
Variance%	27.95081329	24.21580505	14.6873045	7.84212446	7.61045933	5.55432129	5.16551113	3.20171452
Cum%	27.95081329	52.16661835	66.85391998	74.69604492	82.3065033	87.86082458	93.02633667	96.22805023
P-value								

Figure 3.11: PCA Output Using All *Normalized* 13 Numerical Variables in The Breakfast Cereals Dataset. The Table Gives Results for the First Eight Principal Components

importance (earnings per share, gross revenues) it is generally advisable to normalize. In this way, the changes in units of measurement do not change the principal components' weights. In the rare situations where we can give relative weights to variables, we multiply the normalized variables by these weights before doing the principal components analysis.

When we perform PCA we are operating on the covariance matrix. Therefore, an alternative to normalizing and then performing PCA is to perform PCA on the correlation matrix instead of the covariance matrix. Most software programs allow the user to choose between the two. Remember that using the correlation matrix means that you are operating on the normalized data.

Returning to the breakfast cereals data, we normalize the 13 variables due to the different scales of the variables, and then perform PCA (or equivalently, we use PCA applied to the correlation matrix). The output is shown in Figure 11.6. Now we find that we need seven principal components to account for more than 90% of the total variability. The first two principal components account for only 52% of the total variability, and thus reducing the number of variables to two would mean loosing a lot of information. Examining the weights, we see that the first principal component measures the balance between two quantities: (1) calories and cups (large positive weights) vs. (2) protein, fiber, potassium, and consumer rating (large negative weights). High scores on principal component 1 mean that the cereal is high in calories and the amount per bowl, and low in protein, fiber, and potassium. Unsurprisingly, this type of cereal is associated with low consumer rating. The second principal component is most affected by the weight of a serving, and the third principal component by the carbohydrate content. We can continue labeling the next principal components in a similar fashion to learn about the structure of the data.

When the data can be reduced to two dimensions, a useful plot is a scatterplot of the first vs. second principal scores with labels for the observations (if the dataset is not too large). To illustrate this, Figure 3.12 displays the first two principal components scores for the breakfast cereals. We can see that as we move from left (bran cereals) to right, the cereals are less "healthy" in the sense of high calories, low protein and fiber, etc. Also, moving from bottom to top we get heavier cereals (moving from puffed rice to raisin bran). These plots are especially useful if interesting clusterings of observations can be found. For instance, we see here that children's cereals are close together on the middle-right part of the plot.

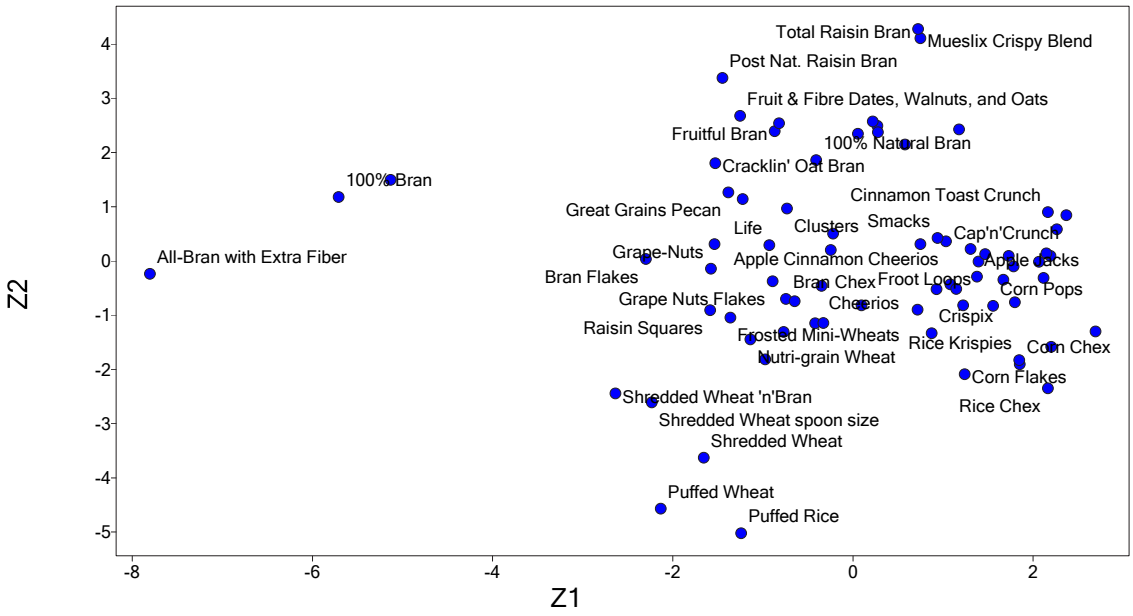


Figure 3.12: Scatterplot of the Second Vs. First Principal Components Scores for the Normalized Breakfast Cereal Output

3.7.4 Using Principal Components for Classification and Prediction

When the goal of the data reduction is to have a smaller set of variables that will serve as predictors, we can proceed as following: Apply PCA to the training data. Use the output to determine the number of principal components to be retained. The predictors in the model now use the (reduced number of) principal scores columns. For the validation set we can use the weights computed from the training data to obtain a set of principal scores, by applying the weights to the variables in the validation set. These new variables are then treated as the predictors.

3.8 Exercises

Breakfast cereals: Use the data for the breakfast cereals example in section 3.7.1 to explore and summarize the data as follows:

1. Which variables are quantitative/numeric? Which are ordinal? Which are nominal?
2. Create a table with the average, median, min, max, and standard deviation for each of the quantitative variables. This can be done through Excel's functions or Excel's *Tools > Data Analysis > Descriptive Statistics* menu.
3. Use XLMiner to plot a histogram for each of the quantitative variables. Based on the histograms and summary statistics, answer the following questions:
   (a) Which variables have the largest variability?
   (b) Which variables seem skewed?
   (c) Are there any values that seem extreme?
4. Use XLMiner to plot a side-by-side boxplot comparing the calories in hot vs. cold cereals. What does this plot show us?
5. Use XLMiner to plot a side-by-side boxplot of consumer rating as a function of the shelf height. If we were to predict consumer rating from shelf height, does it appear that we need to keep all three categories (1,2,3) of shelf height?
6. Compute the correlation table for the quantitative variable (use Excel's *Tools > Data Analysis > Correlation* menu). In addition, use XLMiner to generate a matrix plot for these variables.
   (a) Which pair of variables is most strongly correlated?
   (b) How can we reduce the number of variables based on these correlations?
   (c) How would the correlations change if we normalized the data first?

Consider Figure 3.10, PCA Output for the breakfast cereal data, the first column on the left.

7. Describe briefly what this column represents.

Chemical Features of Wine: The following table (Table 3.5) represents PCA output on data (non-normalized) in which the variables represent chemical characteristics of wine, and each case is a different wine.

1. The data are in Wine.xls. Consider the row near the bottom labeled "Variance." Explain why column 1's variance is so much greater than any other column's variance.
2. Comment on the use of normalization (standardization) in the above case.

Jobs in Europe: Using the file EuropeanJobs.xls, conduct a principal components analysis on the data, and comment on the results. Should the data be normalized? Discuss what characterizes the components you consider key.

Sales of Toyota Corolla cars: The file "ToyotaCorolla.xls" contains the data on used cars (Toyota Corollas) on sale during late summer of 2004 in the Netherlands. It has 1436 records containing details on 38 attributes including Price, Age, Kilometers, Horsepower, and other specifications. The goal will be to predict the price of a used Toyota Corolla based on its specifications.

1. Identify the categorical variables.
2. Explain the relationship between a categorical variable, and the series of binary dummy variables derived from it.

Table 3.5: Principal Components of Non-Normalized Wine Data

	Principal Components					Std. Dev.
	1	2	3	4	5	
Alcohol	0.001	0.013	0.014	-0.030	0.129	0.8
MalicAcid	-0.001	0.009	0.167	-0.427	-0.402	1.2
Ash	0.000	-0.002	0.054	-0.009	0.006	0.3
Ash_Alcalinity	-0.004	-0.045	0.976	0.176	0.060	3.6
Magnesium	0.014	-0.998	-0.040	-0.031	0.006	14.7
Total Phenols	0.001	0.002	-0.015	0.164	0.316	0.7
Flavanoids	0.002	0.000	-0.049	0.214	0.545	1.1
Nonflavanoid_Phenols	0.000	0.002	0.004	-0.025	-0.040	0.1
Proanthocyanins	0.001	-0.007	-0.031	0.082	0.244	0.7
Color Intensity	0.002	0.022	0.097	-0.804	0.536	1.6
Hue	0.000	-0.002	-0.021	0.096	0.064	0.2
OD280/OD315	0.001	-0.002	-0.022	0.220	0.261	0.7
Proline	1.000	0.014	0.004	0.001	-0.004	351.5
Variance	123594.453	194.345	11.424	2.388	1.391	
% Variance	99.830%	0.157%	0.009%	0.002%	0.001%	
Cumulative %	99.830%	99.987%	99.996%	99.998%	99.999%	

3. How many dummy binary variables are required to capture the information in a categorical variable with N categories?
4. Using XLMiner's data utilities, convert the categorical variables in this dataset into dummy binaries, and explain in words the values in the derived binary dummies for one record.
5. Use Excel's correlation command (*Tools* > *Data Analysis* > *Correlation* menu) to produce a correlation matrix, and XLMiner's matrix plot to obtain a matrix of all scatterplots. Comment on the relationships among variables.

Chapter 4

Evaluating Classification and Predictive Performance

4.1 Introduction

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

4.2 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.

4.2.1 Accuracy Measures

A natural criterion for judging the performance of a classifier is the probability for 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 precisely classify cases. Is there a maximal probability of misclassification we should require of a classifier?

At a minimum, we hope to do better than the *naive rule* "classify everything as belonging to the most prevalent class." This rule does not incorporate any predictor information and relies only on the percent of items in each class.

If the classes are well separated by the predictor information, then 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 4.1 illustrates this for a two-class case. The top panel includes a small dataset (n=24 observations) where two predictors (income and lot size) are used for separating owners from non-owners. Here the predictor information seems useful in that it separates between the two classes (owners/non-owners). The bottom panel shows a much larger

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

Table 4.1: Classification Matrix: Meaning of Each Cell

dataset (n=5000 observations) where the two predictors (income and average credit card spending) do not separate the two classes well (loan acceptors/non-acceptors).

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 confusion matrix correspond to the true and predicted classes respectively. Figure 4.2 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 misclassification 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 census data to estimate these proportions. However, in most practical business settings we will not know them. 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 predicted classifications for the observations in the validation set. 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 due to the danger of overfitting.

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/non-buyer). The schematic classification matrix in Table 4.1 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 $i \neq j$ then these are counts of misclassifications. The total number of observations is $n = n_{0,0} + n_{0,1} + n_{1,0} + n_{1,1}$.

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

$$Err = (n_{0,1} + n_{1,0})/n$$

where n is the total number of cases in the validation dataset. In the example in Figure 4.2 we get Err = (25+85)/3000 = 3.67%.

If n is reasonably large, our estimate of the misclassification rate is probably reasonably accurate. We can compute a confidence interval using the standard formula for estimating a population proportion from a random sample. Table 4.2 gives an idea of how the accuracy of the estimate varies

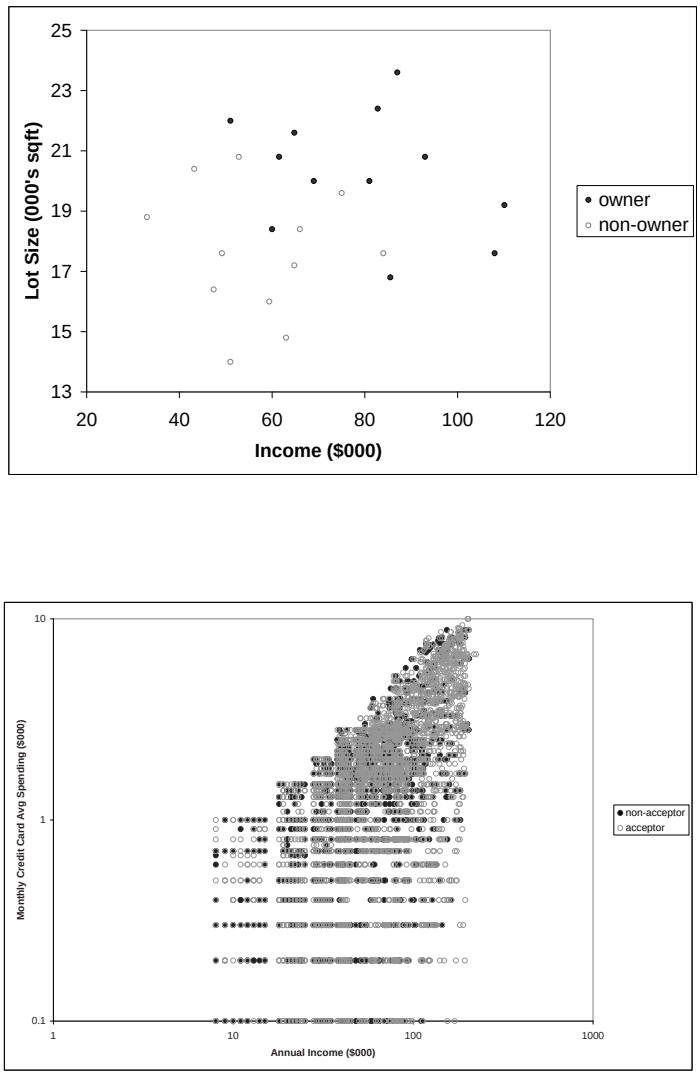


Figure 4.1: High (Top) and Low (Bottom) Levels of Separation Between Two Classes, Using Two Predictors

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

Figure 4.2: Classification Matrix Based on 3000 Observations and Two Classes

	Err							
	0.01	0.05	0.10	0.15	0.20	0.30	0.40	0.50
± 0.025	250	504	956	1,354	1,699	2,230	2,548	2,654
± 0.010	657	3,152	5,972	8,461	10,617	13,935	15,926	16,589
± 0.005	2,628	12,608	23,889	33,842	42,469	55,741	63,703	66,358

Table 4.2: Accuracy of Estimated Misclassification Rate (Err) as a Function of n

with n. The column headings are values of the misclassification rate and the rows give the desired accuracy in estimating the misclassification rate as measured by the half-width of the confidence interval at the 99% confidence level. For example, if we think that the true misclassification rate is likely to be around 0.05 and we want to be 99% confident that Err is within ±0.01 of the true misclassification rate, we need to have a validation dataset with 3,152 cases.

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

$$Accuracy = 1 - Err = (n_{0,0} + n_{1,1})/n$$

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

4.2.2 Cutoff For Classification

Many data mining algorithms classify a case in a two-step manner: first they estimate its probability of belonging to class 1, and then they compare this probability to a threshold called a *cutoff value*. If the probability is above the cutoff, the case is classified as belonging to class 1, and otherwise to class 0. When there are more than two classes, a popular rule is to assign the case to the class to which it has the highest probability of belonging.

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 misclassification rate will rise in either case.

Consider the data in Figure 4.3, 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 adopt instead 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 4.4.

To see the whole range of cutoff values and how the accuracy or misclassification rates change as a function of the cutoff, we can use one-way tables in Excel (see box), and then plot the performance measure of interest vs. the cutoff. The results for the above data are shown in Figure 4.6. We can see that the accuracy level is pretty stable around 0.8 for cutoff values between 0.2 and 0.8.

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

Figure 4.3: 24 Records with Their Actual Class and the Probability of Them Being Class 1 Members, as Estimated by a Classifier

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

Classification Confusion Matrix		
	Predicted Class	
Actual Class	owner	non-owner
owner	11	1
non-owner	2	10

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

Classification Confusion Matrix		
	Predicted Class	
Actual Class	owner	non-owner
owner	11	1
non-owner	4	8

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

Classification Confusion Matrix		
	Predicted Class	
Actual Class	owner	non-owner
owner	7	5
non-owner	1	11

Figure 4.4: Classification Matrices Based on Cutoffs of 0.5 (Top), 0.25 (Middle), and 0.75 (Bottom)

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 confusion matrix. To create a one-variable data table (see Figure 4.5):

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, C11). The left-most column should be titled "cutoff" (A11).
2. In the row below add formulas, using references to the relevant confusion matrix cells (the formula in B12 is =(B6+C7)/(B6+C6+B7+C7)).
3. In the left-most column, list the cutoff values you want to evaluate (we chose $0, 0.05, \ldots, 1$ in B13-B33.)
4. Select the range excluding the first row (B12:C33) and in the *Data* menu select *Table*
5. In "column input cell" select the cell that changes (here, the cell with the cutoff value, $D1$)

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 properly classify 1's 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.

4.2.3 Performance in Unequal Importance of Classes

Suppose the two classes are asymmetric in that it is more important to correctly predict membership in class 0 than in class 1. An example is predicting the financial status (bankrupt/solvent) of firms. It may be more important to correctly predict a firm that is going bankrupt than to correctly predict 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_0. Popular accuracy measures are:

Sensitivity of a classifier is its ability to correctly detect the important class members. This is measured by $n_{0,0}/(n_{0,0} + n_{0,1})$, the % of C_0 members correctly classified.

Specificity of a classifier is its ability to correctly rule out C_1 members. This is measured by $n_{1,1}/(n_{1,0} + n_{1,1})$, the % of C_1 members correctly classified.

False positive rate is $n_{1,0}/(n_{0,0} + n_{1,0})$. 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.

False negative rate is $n_{0,1}/(n_{0,1} + n_{1,1})$. 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.

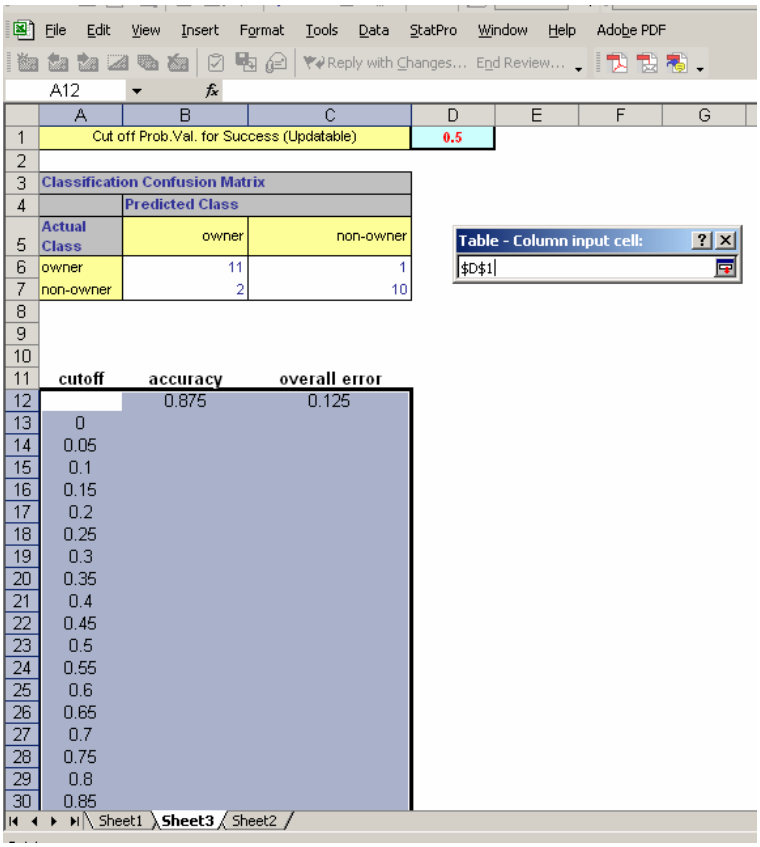


Figure 4.5: Creating One-Way Tables in Excel. Accuracy and Overall Error are Computed for Different Values of the Cutoff

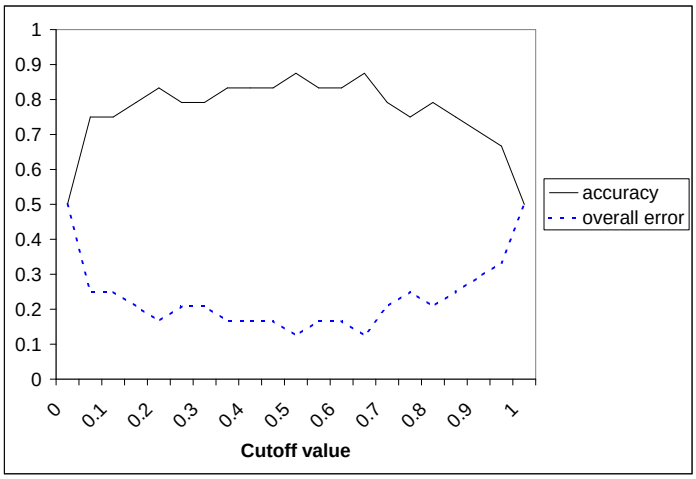


Figure 4.6: Plotting Results From One-Way Table: Accuracy and Overall Error as a Function of The Cutoff Value

It is sometimes useful to plot these measures vs. the cutoff value (using one-way tables in Excel, as described above), in order to find a cutoff value that balances these measures.

A graphical method that is very useful for evaluating the ability of a classifier to "catch" observations of a class of interest is the *lift chart*. We describe this in further detail next.

Lift Charts

Let's 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, etc. We can then make more informed decisions. For example, we can decide how many 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 non-cheaters as we proceed through the sorted data. 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 rather than actual 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 Figure 4.3. We've shown that different choices of a cutoff value lead to different confusion matrices (as in Figure 4.4). 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 as shown below. True positives are those observations from the important class (here class 1) that are classified correctly. Figure 4.7 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 positives we would predict if we 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 we choose to act on. A good classifier will give us a high lift when we act on only a few cases (i.e. use the prediction for the ones 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, continue with a slope of 1 until it reached 12 successes (all the successes), then continue horizontally to the right.

The same information can be portrayed as a "decile" chart, shown in Figure 4.8, 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. 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.

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

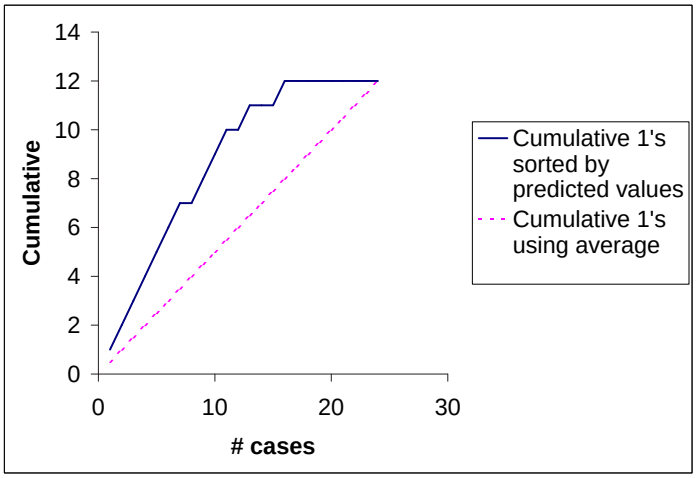


Figure 4.7: Lift Chart and Table Showing the Cumulative True Positives

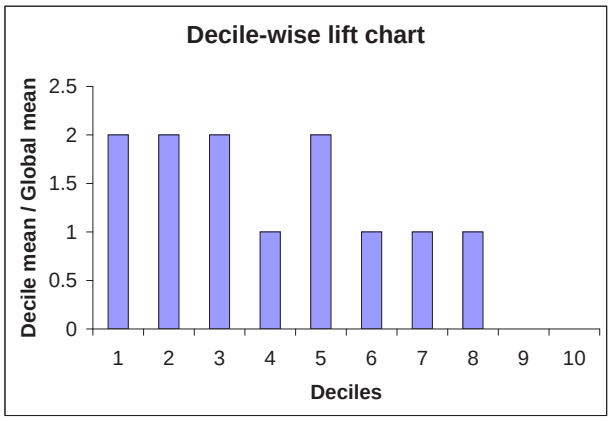


Figure 4.8: Decile Lift Chart

XLMiner automatically 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.

ROC Curve

It is worth mentioning that a curve that captures the same information as the lift curve in a slightly different manner is also popular in data mining applications. This is the ROC (short for Receiver Operating Characteristic) curve. It uses the same variable on the y-axis as the lift curve (but expressed as a percentage of the maximum) and on the x-axis it shows the true negatives (the number of unimportant class members correctly classified, also expressed as a percentage of the maximum) for differing cutoff levels. The ROC curve for our 24 cases example above is shown in Figure 4.9.

4.2.4 Asymmetric Misclassification Costs

Up to this point we have been using the misclassification rate as the criterion for judging the efficacy of a classifier. However, there are circumstances when this measure is not appropriate. Sometimes 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 opportunity cost 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 correctly classifying a buyer 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

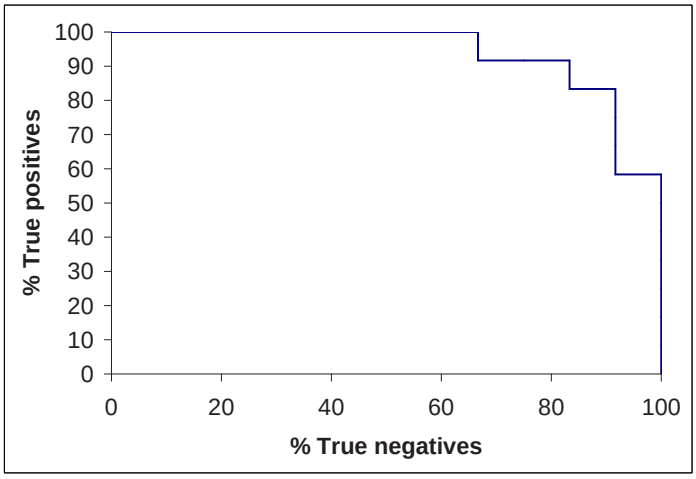


Figure 4.9: ROC Curve For The Example

benefit of correctly classifying a respondent household, 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 non-responder, it will have an error rate of only 1% but it will be useless in practice. A classifier that misclassifies 30% of buying households as non-buyers and 2% of the non-buyers 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 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 we can produce these classifications:

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

The classifications above have an error rate of $100 \times (20 + 2)/1000 = 2.2\%$ – 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 1	Predict class 0
Actual 1	\$80	0
Actual 0	− \$20	0

Looked at purely in terms of costs, when everyone 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 10 1's = \$100. The costs (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 are only \$50, as follows:

COSTS	Predict class 1	Predict class 0
Actual 1	0	\$30
Actual 0	\$20	0

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 previous section, to reflect the asymmetric costs.

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

$$\frac{q_0 n_{0,1} + q_1 n_{1,0}}{n} = \left(\frac{n_{0,1}}{n_{0,0} + n_{0,1}}\right)\left(\frac{n_{0,0} + n_{0,1}}{n}\right) q_0 + \left(\frac{n_{1,0}}{n_{1,0} + n_{1,1}}\right)\left(\frac{n_{1,0} + n_{1,1}}{n}\right) q_1$$

Minimizing this expression is equivalent to minimizing the same expression divided by a constant. If we divide by q_0, then 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 hard 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 we use stratified sampling, when one class is oversampled (as described in the next section), then we can use external/prior information on the proportions of observations belonging to each class, denoted by $p(C_0)$ and $p(C_1)$, and incorporate them into the cost structure:

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

This is called the *expected misclassification cost*. 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 we have m classes $C_0, C_1, C_2, \cdots C_{m-1}$. The confusion 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 different 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.

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 above value is the y-coordinate of the first point on the lift chart. The x coordinate is the index # 1.
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 the index # 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).

Notice that this is similar to plotting the lift as a function of the cutoff. The only difference is the scale on the x-axis. When the goal is to select the top records based on a certain budget, then the lift vs. number of records is preferable. In contrast, when the goal is to find a cutoff that distinguishes well between the two classes, then the lift vs. the cutoff value is more useful.

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%, then the expected net value of mailing to a list of 10,000 is $(0.02 \times \$25 * 10,000)(\$0.65 \times 10,000) = \$5000 - \$6500 = -\$1500$. 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 4.10.

4.2.5 Oversampling and Asymmetric Costs

As we saw briefly in Chapter 2, when classes are present in very unequal proportions, 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, etc.

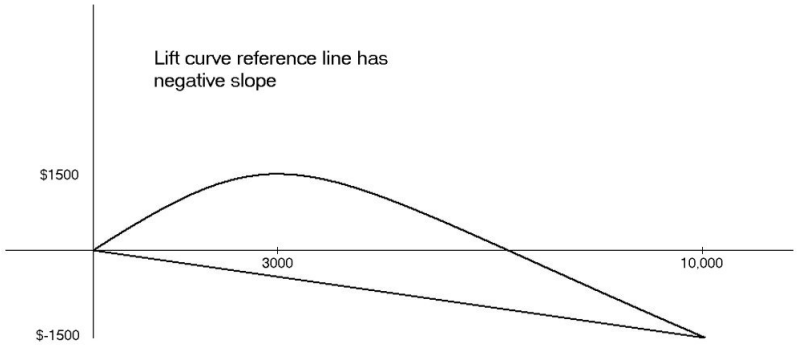


Figure 4.10: Lift Curve Incorporating Costs

In all discussion 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 4.11: "x" represents non-responders 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 "x"). If we incorporate more realistic misclassification costs − lets say that failing to catch a "o" is 5 times as costly as failing to catch an "x" − then the costs of misclassification jump to 5. In such a case, a horizontal line, as shown in Figure 4.12, does a better job: it results in misclassification costs of just 2.

Oversampling is one way of incorporating these costs into the training process. In Figure 4.13, 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" four times over.

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 non-responders 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 non-responders, but the exact ratio of costs is difficult to determine.

Practitioners, when faced with very low response rates in a classification problem, often sample equal numbers of responders and non-responders 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 oversampling (i.e., via simple random sampling).

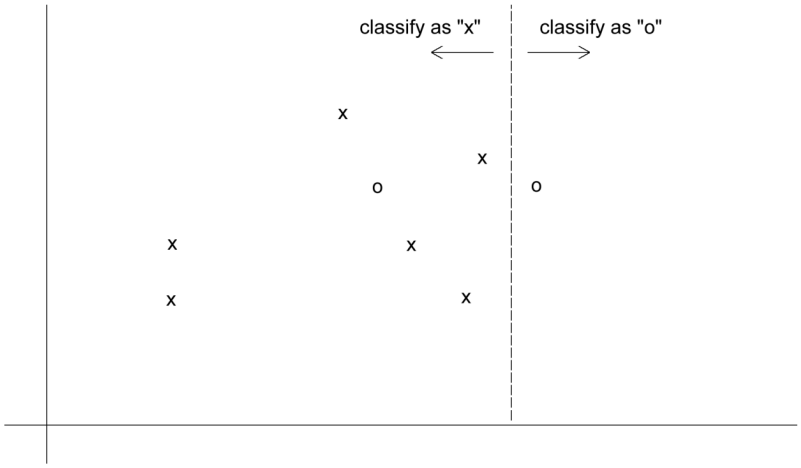


Figure 4.11: Classification Assuming Equal Costs of Misclassification

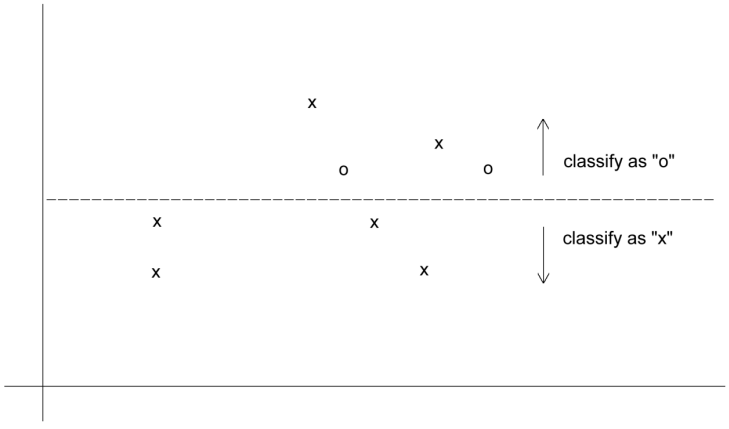


Figure 4.12: Classification, Assuming Unequal Costs of Misclassification

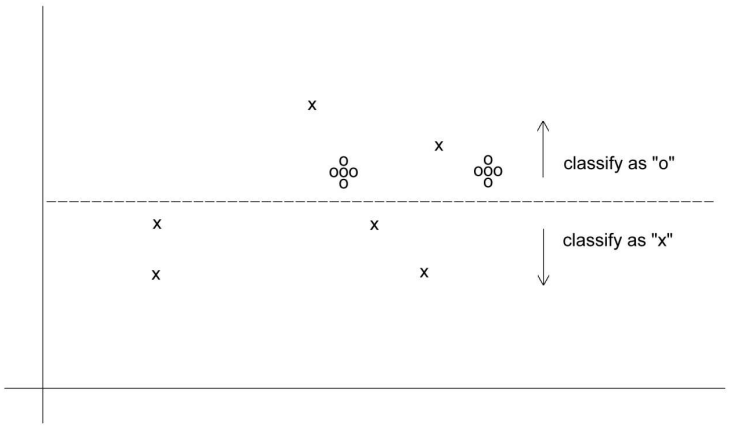


Figure 4.13: Classification, Using Oversampling to Account for Unequal Costs

2. Score the model to an oversampled validation set, and reweight the results to remove the effects of oversampling.

The first method is the most straightforward and easiest to implement. We describe how to oversample, and also how to evaluate performance for each of the two methods.

In all discussion 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.

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 the 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 non-response data are separated into two distinct sets, or "strata."
2. Records are then randomly selected for the training set from each strata. Typically, one might select half the (scarce) responders for the training set, then an equal number of non-responders.
3. The remaining responders are put in the validation set.
4. Non-responders are randomly selected for the validation set in sufficient number, so as to maintain the original ratio of responders to non-responders.

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 non-oversampled 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 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 using oversampled validation set

In some cases, very low response rates (perhaps combined with software limitations or lack of access to original data) may make it more practical to use oversampled data not only for the training data, but also for the validation data. In such circumstances, the only way to predict model performance is to re-weight the sample to restore the non-responders that were under-represented 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's 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 confusion matrix looks like this:

	Actual 1	Actual 0	
Predicted 1	420	110	530
Predicted 0	80	390	470
	500	500	1000

Confusion Matrix, Oversampled Data (Validation)

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 only constitute 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 confusion matrix by augmenting the number of (actual) non-responders, assigning them to the appropriate cells in the same ratio in which they appear in the above confusion table (3.545 predicted 1's for every predicted 0):

	Actual 1	Actual 0	
Predicted 1	420	5,390	5,810
Predicted 0	80	19,110	19,190
	500	24,500	25,000

Confusion Table, Reweighted

The adjusted misclassification rate is (80+5390)/25,000 = 21.9%, and the model ends up classifying 5,810/25,000 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 correctly classifying all the records, as in finding a model that guides us towards 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 benefit/cost element. The following procedure can be used (and easily implemented in Excel):

1. Sort the validation 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. 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 above value is the y-coordinate of the first point on the lift chart. The x coordinate is the index #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 the index #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).

4.2.6 Classification Using a Triage Strategy

In some cases it is useful to have a "can't 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 confidently pick C_0 or C_1. 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 that 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 medically treated 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 the 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.

4.3 Evaluating Predictive Performance

When the response variable is continuous, the evaluation of model performance is slightly different than 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 goodness-of-fit measures in classical regression modeling, 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 to assess predictive accuracy than the training set. This is because records in the validation set are not used to select predictors or 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). The prediction error for observation 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) $= \frac{1}{n}\sum_{i=1}^{n}|e_i|$. This gives the magnitude of the average absolute error.
- Average Error $= \frac{1}{n}\sum_{i=1}^{n} e_i$. This measure is similar to MAD, except it retains the sign of the errors, so that negative errors cancel out positive errors of the same magnitude. It therefore gives an indication whether the predictions are on average over-predicting or under-predicting the response.
- MAPE (Mean Absolute Percentage Error) $= 100\% \times \frac{1}{n}\sum_{i=1}^{n}\frac{e_i}{y_i}$. This measure gives a percentage score of how predictions deviate (on average) from the actual values.
- RMSE (Root Mean Squared Error) $= \sqrt{\frac{1}{n}\sum_{i=1}^{n} e_i^2}$. This is similar to the standard error of estimate, except that it is computed on the validation data rather than the training data. It has the same units of the predicted variable.
- 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. In order 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 renew 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

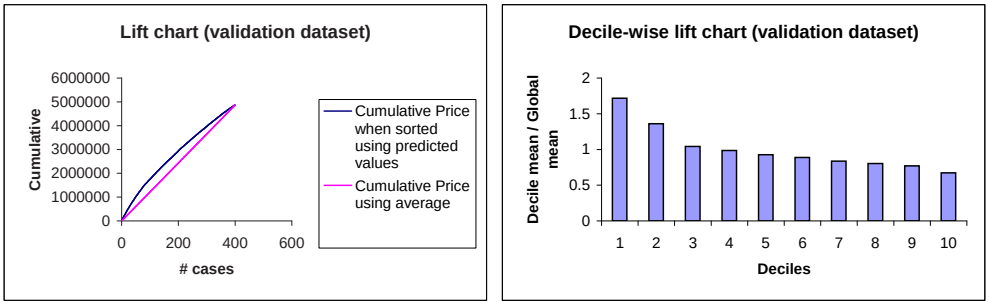


Figure 4.14: Lift Chart for Continuous Response (Sales)

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 4.14 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 the car's features (mileage, color, etc.) 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 above lift (and decile-wise) chart would be useful in the following scenario: For example, choosing the top 10% of the cars that gave the highest predicted sales 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. This number can also be computed from the lift chart by comparing the predicted sales from 40 random cars of $486,871 (= sum of the predictions of the 400 validation set cars divided by 10) with the 40 cars with the highest predicted values according to the model, giving $885,883. The ratio between these numbers is 1.7.

4.4 Exercises

1. A data mining routine has been applied to a transaction data set and has classified 88 records as fraudulent (30 correctly so), and 952 as non-fraudulent (920 correctly so). Construct the confusion matrix and calculate the error rate.
2. Suppose 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 classification error rate for records that are truly fraudulent

   - the classification error rate for records that are truly non-fraudulent
3. Consider Figure 4.15: Lift chart for the transaction data model, applied to new data:

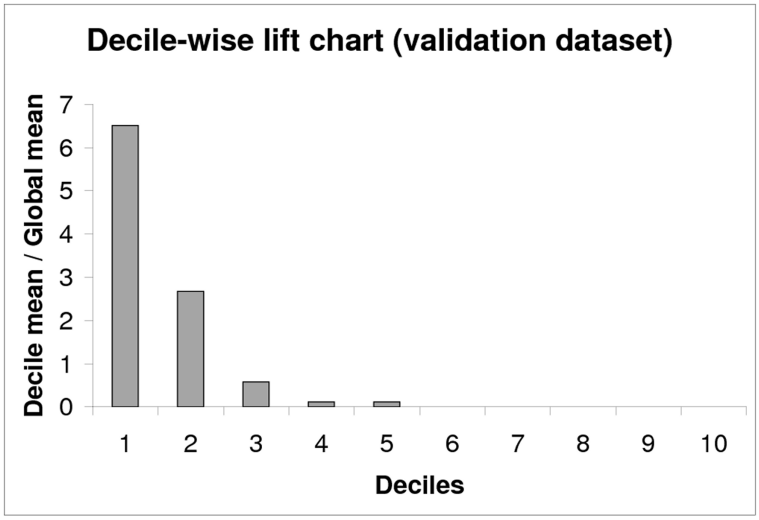


Figure 4.15: Lift Chart for Transaction Data

   (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 non-fraudulent. If you 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).
4. A large number of insurance records are to be examined to develop a model for predicting fraudulent claims. 1% of the claims in the historical database were judged as 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 non-frauds. It missed 90 frauds, and incorrectly classified 130 records as frauds when they were not.

(a) Produce the confusion matrix for the sample as it stands.

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

(c) What percent of new records would you expect to be classified as frauds?

Chapter 5

Multiple Linear Regression

5.1 Introduction

The most popular model for making predictions is the *multiple linear regression* model encountered in most introductory statistics classes and textbooks. This model is used to fit a linear relationship between a quantitative dependent variable Y (also called the outcome or response variable), and a set of predictors $X_1, X_2, \cdots, X_p$ (also referred to as independent variables, input variables, regressors or covariates). The assumption is that in the population of interest the following relationship holds:

$$Y = \beta_0 + \beta_1 x_1 + \beta_2 x_2 + \cdots + \beta_p x_p + \epsilon \tag{5.1}$$

where $\beta_0, \ldots, \beta_p$ are coefficients, and ϵ is the "noise", or the "unexplained" part. The data, which are a sample from this population, are then used to estimate the coefficients and the variability of the "noise."

The two popular objectives behind fitting a model that relates a quantitative outcome with predictors are for understanding the relationship between these factors, and for predicting outcomes of new cases. The classical statistical approach has focused on the first objective, namely, fitting the best model to the data in an attempt to learn about the underlying relationship in the population. In data mining, however, the focus is typically on the second goal, i.e. predicting new observations. Important differences between the approaches stem from the fact that in the classical statistical world we are interested in drawing conclusions from a limited supply of data, and in learning how reliable those conclusions might be. In data mining, by contrast, data are typically plentiful, so the performance and reliability of our model can easily be established by applying it to fresh data.

Multiple linear regression is applicable to numerous data mining situations. Examples are: predicting customer activity on credit cards from their demographics and historical activity patterns, predicting the time to failure of equipment based on utilization and environment conditions, predicting expenditures on vacation travel based on historical frequent flyer data, predicting staffing requirements at help desks based on historical data and product and sales information, predicting sales from cross selling of products from historical information, and predicting the impact of discounts on sales in retail outlets. Although a linear regression model is used for both goals, the modelling step and performance assessment differ depending on the goal. Therefore, the choice of model is closely tied to whether the goal is explanatory or predictive.

5.2 Explanatory Vs. Predictive Modeling

Both explanatory modeling and predictive modeling involve using a dataset to fit a model (i.e. to estimate coefficients), checking model validity, assessing its performance, and comparing to other

models. However, there are several major differences between the two:

1. A good explanatory model is one that fits the data closely, whereas a good predictive model is one that accurately predicts new cases.

2. In explanatory models (classical statistical world, scarce data) the entire dataset is used for estimating the best-fit model, in order to maximize the amount of information that we have about the hypothesized relationship in the population. When the goal is to predict outcomes of new cases (data mining, plentiful data), the data are typically split into training set and validation set. The training set is used to estimate the model, and the holdout set is used to assess this model's performance on new, unobserved data.

3. Performance measures for explanatory models measure how close the data fit the model (how well the model approximates the data), whereas in predictive models performance is measured by predictive accuracy (how well the model predicts new cases).

For these reasons it is extremely important to know the goal of the analysis before beginning the modelling process. A good predictive model can have a looser fit to the data it is based on, and a good explanatory model can have low prediction accuracy. In the remainder of this chapter we focus on predictive models, because these are more popular in data mining, and because most textbooks focus on explanatory modelling.

5.3 Estimating the Regression Equation and Prediction

The coefficients $\beta_0, \ldots, \beta_p$ and the standard deviation of the noise (σ) determine the relationship in the population of interest. Since we only have a sample from that population, these coefficients are unknown. We therefore estimate them from the data using a method called *Ordinary Least Squares* (OLS). This method finds values $\hat{\beta}_0, \hat{\beta}_1, \hat{\beta}_2, \cdots, \hat{\beta}_p$ that minimize the sum of squared deviations between the actual values (Y) and their predicted values based on that model ($\hat{Y}$).

To predict the value of the dependent value from known values of the predictors, $x_1, x_2, \cdots, x_p$, we use sample estimates for $\beta_0, \ldots, \beta_p$ in the linear regression model (1), since $\beta_0, \ldots, \beta_p$ cannot be directly observed unless we have available the entire population of interest. The predicted value, $\hat{Y}$, is computed from the equation $\hat{Y} = \hat{\beta}_0 + \hat{\beta}_1 x_1 + \hat{\beta}_2 x_2 + \cdots + \hat{\beta}_p x_p$. Predictions based on this equation are the best predictions possible in the sense that they will be unbiased (equal to the true values on average) and will have the smallest average squared error compared to any unbiased estimates *if* we make the following assumptions:

1. The noise ϵ (or equivalently the dependent variable) follows a normal distribution

2. The linear relationship is correct

3. The cases are independent of each other

4. The variability in Y values for a given set of predictors is the same, regardless of the values of the predictors ("homoskedasticity".)

An important and interesting fact for the predictive goal is that *even if we drop the first assumption and allow the noise to follow an arbitrary distribution, these estimates are very good for prediction,* in the sense that among all linear models, as defined by equation (1) above, the model using the least squares estimates, $\hat{\beta}_0, \hat{\beta}_1, \hat{\beta}_2, \cdots, \hat{\beta}_p$, will have the smallest average squared errors. The Normal distribution assumption is required in the classical implementation of multiple linear regression to derive confidence intervals for predictions. In this classical world, data are scarce and the same data are used to fit the regression model and to assess its reliability (with confidence

limits). In data mining applications we have two distinct sets of data: the training dataset and the validation dataset both are representative of the relationship between the dependent and independent variables. The training data is used to fit the model and estimate the regression coefficients $\beta_0, \beta_1, \cdots, \beta_p$. The validation dataset constitutes a "hold-out" sample and is not used in computing the coefficient estimates. The estimates are then used to make predictions for each case in the validation data. This enables us to estimate the error in our predictions by using the validation set without having to assume that the noise follows a Normal distribution. The prediction for each case is then compared to the value of the dependent variable that was actually observed in the validation data. The average of the square of this error enables us to compare different models and to assess the prediction accuracy of the model.

5.3.1 Example: Predicting the Price of Used Toyota Corolla Automobiles

A large Toyota car dealer offers purchasers of a new Toyota cars the option to buy from them their used car. In particular, a new promotion promises to pay high prices for used Toyota Corolla cars for purchasers of a new car. The dealer then sells the used cars for a small profit. In order to ensure a reasonable profit, the dealer needs to be able to predict the price that the dealership will get for the used cars. For that reason data were collected on all previous sales of used Toyota Corolla's at their dealership. The data include the sales price and information on the car such as its age, mileage, fuel type, engine size, etc. A description of each of these variables is given in Table 5.1. A sample of this dataset is shown in Table 5.2.

Table 5.1: Description of The Variables for Toyota Corolla Example

Variable	**Description**
Price	Offer Price in EUROs
Age	Age in months as in August 2004
Mileage	Accumulated Kilometers on odometer
Fuel Type	Fuel Type (Petrol, Diesel, CNG)
HP	Horse Power
Metallic Color	Metallic Color? (Yes=1, No=0)
Color	Color (Blue, Red, Grey, Silver, Black, etc.)
Automatic	Automatic (Yes=1, No=0)
CC	Cylinder Volume in cubic centimeters
Doors	Number of doors
Quarterly Tax	Quarterly road tax in EUROs
Weight	Weight in Kilograms

The total number of records in the dataset is 1000 cars. After partitioning the data into training and validation sets (at a 60%-40% ratio), we fit a multiple linear regression model between price (the dependent variable) and the other variables (as predictors) using the training set only. Figure 5.1 shows the estimated coefficients (as computed by XLMiner). Notice that the "Fuel Type" predictor has three categories (Petrol, Diesel, and CNG) and we therefore have two dummy variables in the model (e.g., Petrol (0/1) and Diesel (0/1); the third CNG (0/1) is redundant given the information on the first two dummies). These coefficients are then used to predict prices of used Toyota Corolla cars based on their age, mileage, etc. Figure 5.2 shows a sample of 20 of the predicted prices for cars in the validation set, using the estimated model. It gives the predictions and their errors (relative to the actual prices) for these 20 cars. On the right we get overall measures of predictive accuracy. Note that the average error is \$111. A boxplot of the residuals (Figure 5.3) shows that 50% of the errors are approximately between ±\$850. This might be small relative to the car price, but should be taken into account when considering the profit. Such measures are used to assess predictive performance of a model and also to compare different models. We discuss such measures in the next section. This

Table 5.2: Prices and Attributes for a Sample of 30 Used Toyota Corolla Cars

Price	Age	Mileage	Fuel Type	HP	Metallic Color	Automatic	CC	Doors	Quart Tax	Weight
13500	23	46986	Diesel	90	1	0	2000	3	210	1165
13750	23	72937	Diesel	90	1	0	2000	3	210	1165
13950	24	41711	Diesel	90	1	0	2000	3	210	1165
14950	26	48000	Diesel	90	0	0	2000	3	210	1165
13750	30	38500	Diesel	90	0	0	2000	3	210	1170
12950	32	61000	Diesel	90	0	0	2000	3	210	1170
16900	27	94612	Diesel	90	1	0	2000	3	210	1245
18600	30	75889	Diesel	90	1	0	2000	3	210	1245
21500	27	19700	Petrol	192	0	0	1800	3	100	1185
12950	23	71138	Diesel	69	0	0	1900	3	185	1105
20950	25	31461	Petrol	192	0	0	1800	3	100	1185
19950	22	43610	Petrol	192	0	0	1800	3	100	1185
19600	25	32189	Petrol	192	0	0	1800	3	100	1185
21500	31	23000	Petrol	192	1	0	1800	3	100	1185
22500	32	34131	Petrol	192	1	0	1800	3	100	1185
22000	28	18739	Petrol	192	0	0	1800	3	100	1185
22750	30	34000	Petrol	192	1	0	1800	3	100	1185
17950	24	21716	Petrol	110	1	0	1600	3	85	1105
16750	24	25563	Petrol	110	0	0	1600	3	19	1065
16950	30	64359	Petrol	110	1	0	1600	3	85	1105
15950	30	67660	Petrol	110	1	0	1600	3	85	1105
16950	29	43905	Petrol	110	0	1	1600	3	100	1170
15950	28	56349	Petrol	110	1	0	1600	3	85	1120
16950	28	32220	Petrol	110	1	0	1600	3	85	1120
16250	29	25813	Petrol	110	1	0	1600	3	85	1120
15950	25	28450	Petrol	110	1	0	1600	3	85	1120
17495	27	34545	Petrol	110	1	0	1600	3	85	1120
15750	29	41415	Petrol	110	1	0	1600	3	85	1120
11950	39	98823	CNG	110	1	0	1600	5	197	1119

The Regression Model

Input variables	Coefficient	Std. Error	p-value	SS
Constant term	-2327.28149	1622.56287	0.15210986	8.1482E+10
Age	-134.137619	4.77474403	0	5888770000
Mileage	-0.0199055	0.00236949	0	172544200
Fuel_Type_Diesel	129.241013	536.766052	0.80982733	2427870
Fuel_Type_Petrol	2670.87329	520.021179	0.0000004	670008.438
Horse_Power	33.9551201	5.37533283	0	339071900
Metalic_Color	-38.049099	120.321022	0.75196105	716922.5
Automatic	224.9384	269.069672	0.40356547	10970180
CC	0.0209207	0.0959821	0.8275463	1553226
Doors	-3.00326943	61.7951851	0.96125734	17263280
Quarterly_Tax	22.9035111	2.48583364	0	221851400
Weight	12.9385519	1.51249933	0	136067800

Residual df	588
Multiple R-squared	0.86134457
Std. Dev. estimate	1363.60046
Residual SS	1093331000

Figure 5.1: Estimated Coefficients for Regression Model of Price Vs. Car Attributes

Validation Data scoring

Predicted Value	Actual Value	Residual
16199	13750	-2449
16686	13950	-2736
16266	16900	634
16236	18600	2364
20534	20950	416
20520	19600	-920
19860	21500	1640
19504	22500	2996
20385	22000	1615
16993	16950	-43
16106	16950	844
16099	16250	151
15789	15750	-39
15590	15950	360
15660	14950	-710
15668	14750	-918
15300	16750	1450
17919	19000	1081
17242	17950	708
19148	21950	2802

Validation Data scoring - Summary Report

Total sum of squared errors	RMS Error	Average Error
795600925.2	1410.31993	110.914571

Figure 5.2: Predicted Prices (and Errors) For 20 Cars in Validation Set, and Summary Predictive Measures for Entire Validation Set

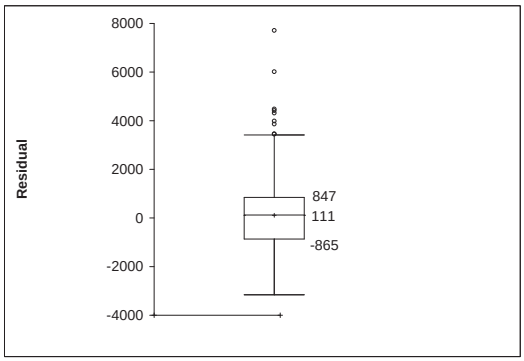


Figure 5.3: Boxplot of Model Residuals (Based on Validation Set)

example also illustrates the point about the relaxation of the Normality assumption. A histogram or probability plot of prices shows a right-skewed distribution. In a classical modeling case where the goal is to obtain a good fit to the data, the dependent variable would be transformed (e.g., by taking a natural log) to achieve a more "Normal" variable. Although the fit of such a model to the training data is expected to be better, it will not necessarily yield a significant predictive improvement. In this example the average error in a model of log(price) is -$160, compared to $111 in the original model for price.

5.4 Variable Selection in Linear Regression

5.4.1 Reducing the Number of Predictors

A frequent problem in data mining is that of using a regression equation to predict the value of a dependent variable when we have many variables available to choose as predictors in our model. Given the high speed of modern algorithms for multiple linear regression calculations, it is tempting in such a situation to take a kitchen-sink approach: why bother to select a subset, just use all the variables in the model. There are several reasons why this could be undesirable.

- It may be expensive or not feasible to collect the full complement of predictors for future predictions.
- We may be able to measure fewer predictors more accurately (e.g., in surveys).
- The more predictors, the higher chance of missing values in the data. If we delete or impute cases with missing values, then multiple predictors will lead to a higher rate of case deletion or imputation.
- *Parsimony* is an important property of good models. We obtain more insight into the influence of predictors in models with few parameters.
- Estimates of regression coefficients are likely to be unstable due to *multicollinearity* in models with many variables. (Multicollinearity is the presence of two or more predictors sharing the same linear relationship with the outcome variable.) Regression coefficients are more stable for parsimonious models. One very rough rule of thumb is to have a number of cases n larger than $5(p + 2)$, where p is the number of predictors.
- It can be shown that using predictors that are uncorrelated with the dependent variable increases the variance of predictions.
- It can be shown that dropping predictors that are actually correlated with the dependent variable can increase the average error (bias) of predictions.

The last two points mean that there is a tradeoff between too few and too many predictors. In general, accepting some bias can reduce the variance in predictions. This *bias-variance tradeoff* is particularly important for large numbers of predictors, because in that case it is very likely that there are variables in the model that have small coefficients relative to the standard deviation of the noise and also exhibit at least moderate correlation with other variables. Dropping such variables will improve the predictions as it will reduce the prediction variance. This type of bias-variance tradeoff is a basic aspect of most data mining procedures for prediction and classification. In light of this methods for reducing the number of predictors p to a smaller set are often used.

5.4.2 How to Reduce the Number of Predictors

The first step in trying to reduce the number of predictors should always be to use domain knowledge. It is important to understand what the different predictors are measuring, and why they are relevant for predicting the response. With this knowledge the set of predictors should be reduced to a sensible set that reflects the problem at hand. Some practical reasons of predictor elimination are expense of collecting this information in the future, inaccuracy, high correlation with another predictor, many missing values, or simply irrelevance. Summary statistics and graphs are also helpful in examining potential predictors. Useful ones are frequency and correlation tables, predictor-specific summary statistics and plots, and missing values counts.

The next step makes use of computational power and statistical significance. In general there are two types of methods for reducing the number of predictors in a model. The first is an exhaustive search for the "best" subset of predictors by fitting regression models with all the possible different combinations of predictors. The second is to search through a partial set of models. We describe these two approaches next.

Exhaustive search

The idea here is to evaluate all subsets. Since the number of subsets for even moderate values of p is very large we need some way to examine the most promising subsets and to select from them. Criteria for evaluating and comparing models are based on the fit to the training data. One popular criterion is the adjusted-R^2 which is defined as

$$R^2_{adj} = 1 - \frac{n-1}{n-p-1}(1-R^2),$$

where R^2 is the proportion of explained variability in the model (in a model with a single predictor this is the squared correlation). Like R^2, higher values of adjusted-R^2 indicate better fit. Unlike R^2, which does not account for the number of predictors used, the adjusted-R^2 uses a penalty on the number of predictors. This avoids the artificial increase in R^2 which can result from simply increasing the number of predictors but not the amount of information. It can be shown that using R^2_{adj} to choose a subset is equivalent to picking the subset that minimizes $\hat{\sigma}^2$.

Another criterion that is often used for subset selection is known as Mallow's C_p. This criterion assumes that the full model (with all predictors) is unbiased although it may have predictors that, if dropped, would reduce prediction variability. With this assumption we can show that if a subset model is unbiased, then the average C_p equals the number of parameters $p+1$ (=number of predictors + 1), the size of the subset. So a reasonable approach to identifying subset models with small bias is to examine those with values of C_p that are near $p+1$. C_p is also an estimate of the error[1] for predictions at the x-values observed in the training set. Thus good models are those that have values of C_p near $p+1$ and that have small p (i.e. are of small size). C_p is computed from the formula:

$$C_p = \frac{SSR}{\hat{\sigma}^2_{Full}} + 2(p+1) - n,$$

where $\hat{\sigma}^2_{Full}$ is the estimated value of σ^2 in the full model that includes all predictors. It is important to remember that the usefulness of this approach depends heavily on the reliability of the estimate of σ^2 for the full model. This requires that the training set contain a large number of observations relative to the number of predictors. Finally, a useful point to note is that for a fixed size of subset, R^2, R^2_{adj} and C_p all select the same subset. In fact there is no difference between them in the order of merit they ascribe to subsets of a fixed size.

Figure 5.4 gives the results of applying an exhaustive search on the Toyota Corolla price data (with the 11 predictors). It reports the best model with a single predictor, two predictors, etc. It

[1] In particular, it is the sum of MSE standardized by dividing by σ^2.

#Coeffs	RSS	Cp	R-Sq	Adj. R Sq	Model (Constant present in all models)											
					1	2	3	4	5	6	7	8	9	10	11	12
2	1996467712	477.71	0.75	0.75	Constant	Age	*	*	*	*	*	*	*	*	*	*
3	1672546432	305.51	0.79	0.79	Constant	Age	HP	*	*	*	*	*	*	*	*	*
4	1438242432	181.50	0.82	0.82	Constant	Age	HP	Weight	*	*	*	*	*	*	*	*
5	1258062976	86.59	0.84	0.84	Constant	Age	Mileage	HP	Weight	*	*	*	*	*	*	*
6	1181816320	47.59	0.85	0.85	Constant	Age	Mileage	Petrol	QuarTax	Weight	*	*	*	*	*	*
7	1095153024	2.98	0.86	0.86	Constant	Age	Mileage	Petrol	HP	QuarTax	Weight	*	*	*	*	*
8	1093753344	4.23	0.86	0.86	Constant	Age	Mileage	Petrol	HP	Automatic	QuarTax	Weight	*	*	*	*
9	1093557120	6.12	0.86	0.86	Constant	Age	Mileage	Petrol	HP	Metalic	Automatic	QuarTax	Weight	*	*	*
10	1093422592	8.05	0.86	0.86	Constant	Age	Mileage	Diesel	Petrol	HP	Metalic	Automatic	QuarTax	Weight	*	*
11	1093335424	10.00	0.86	0.86	Constant	Age	Mileage	Diesel	Petrol	HP	Metalic	Automatic	CC	QuarTax	Weight	*
12	1093331072	12.00	0.86	0.86	Constant	Age	Mileage	Diesel	Petrol	HP	Metalic	Automatic	CC	Doors	QuarTax	Weight

Figure 5.4: Exhaustive Search Result for Reducing Predictors in Toyota Corolla Prices Example

can be seen that the R^2_{adj} increases until 6 predictors are used (#coeffs = 7) and then stabilizes. The C_p indicates that a model with 9-11 predictors is good. The dominant predictor in all models is the age of the car, with horsepower and mileage playing important roles as well.

Popular subset selection algorithms

The second method for finding the best subset of predictors relies on a partial, iterative search through the space of all the possible regression models. The end product is one best subset of predictors (although there do exist variations of these methods that identify several close to best choices for different sizes of predictor subsets). This approach is computationally cheaper, but it has the potential of missing "good" combinations of predictors. None of the methods guarantee that they yield the best subset for any criterion such as adjusted-R^2. They are reasonable methods for situations with large numbers of predictors, but for moderate numbers of predictors the exhaustive search is preferable.

Three popular iterative search algorithms are *forward selection*, *backward elimination*, and *stepwise regression*. In "forward selection" we start with no predictors, and then add predictors one by one. Each added predictor is the one (among all predictors) that has the largest contribution to the R^2 on top of the predictors that are already in it. The algorithm stops when the contribution of additional predictors is not statistically significant. The main disadvantage of this method is that the algorithm will miss pairs or groups of predictors that perform very well together, but perform poorly as single predictors. This is similar to interviewing job candidates for a team project one by one, thereby missing groups of candidates who perform superiorly together, but poorly on their own.

In "backward elimination" we start with all predictors, and then at each step eliminate the least useful predictor (according to statistical significance). The algorithm stops when all the remaining predictors have significant contributions. The weakness of this algorithm is that computing the initial model with all predictors can be time-consuming and unstable. Finally, "stepwise" is like forward selection except that at each step we consider dropping predictors that are not statistically significant as in backward elimination.

Note: In XLMiner, unlike other popular software packages (SAS, Minitab, etc.) these three algorithms yield a table similar to the one that the exhaustive search yields rather than a single model. This allows the user to decide on the subset size after reviewing all the possible sizes, based on criteria such as R^2_{adj} and C_p.

For the Toyota Corolla prices example, forward selection yields the exact same results as the

#Coeffs	RSS	Cp	R-Sq	Adj. R Sq	Probability	Model (Constant present in all models)											
						1	2	3	4	5	6	7	8	9	10	11	12
2	1996467712	477.712	0.747	0.746	0	Constant	Age	*	*	*	*	*	*	*	*	*	*
3	1780184064	363.394	0.774	0.773	0	Constant	Age	Weight	*	*	*	*	*	*	*	*	*
4	1482806272	205.462	0.812	0.811	0	Constant	Age	Petrol	Weight	*	*	*	*	*	*	*	*
5	1310214400	114.641	0.834	0.833	0	Constant	Age	Petrol	terly_Tax	Weight	*	*	*	*	*	*	*
6	1181816320	47.5879	0.85	0.849	8E-08	Constant	Age	Mileage	Petrol	QuarTax	Weight	*	*	*	*	*	*
7	1095153024	2.97989	0.861	0.86	0.962122	Constant	Age	Mileage	Petrol	HP	QuarTax	Weight	*	*	*	*	*
8	1093753344	4.22713	0.861	0.86	0.993999	Constant	Age	Mileage	Petrol	HP	Automatic	QuarTax	Weight	*	*	*	*
9	1093557120	6.1216	0.861	0.859	0.989111	Constant	Age	Mileage	Petrol	HP	Metalic	Automatic	QuarTax	Weight	*	*	*
10	1093422592	8.04925	0.861	0.859	0.975677	Constant	Age	Mileage	Diesel	Petrol	HP	Metalic	Automatic	QuarTax	Weight	*	*
11	1093335424	10.0024	0.861	0.859	0.961197	Constant	Age	Mileage	Diesel	Petrol	HP	Metalic	Automatic	CC	QuarTax	Weight	*
12	1093331072	12	0.861	0.859	1	Constant	Age	Mileage	Diesel	Petrol	HP	Metalic	Automatic	CC	Doors	QuarTax	Weight

Figure 5.5: Backward Elimination Result for Reducing Predictors in Toyota Corolla Prices Example

#Coeffs	RSS	Cp	R-Sq	Adj. R Sq	Probability	Model (Constant present in all models)											
						1	2	3	4	5	6	7	8	9	10	11	12
2	1996467712	477.712	0.747	0.746	0	Constant	Age	*	*	*	*	*	*	*	*	*	*
3	1672546432	305.506	0.788	0.787	0	Constant	Age	HP	*	*	*	*	*	*	*	*	*
4	1438242560	181.495	0.818	0.817	0	Constant	Age	HP	Weight	*	*	*	*	*	*	*	*
5	1258062976	86.5938	0.84	0.839	0	Constant	Age	Mileage	HP	Weight	*	*	*	*	*	*	*
6	1188944640	51.4216	0.849	0.848	2E-08	Constant	Age	Mileage	HP	QuarTax	Weight	*	*	*	*	*	*
7	1095153024	2.97989	0.861	0.86	0.962122	Constant	Age	Mileage	Petrol	HP	QuarTax	Weight	*	*	*	*	*
8	1093753344	4.22713	0.861	0.86	0.993999	Constant	Age	Mileage	Petrol	HP	Automatic	QuarTax	Weight	*	*	*	*
9	1468513408	207.775	0.814	0.811	0	Constant	Age	Mileage	Diesel	Petrol	HP	Metalic	Automatic	CC	*	*	*
10	1451250000	200.491	0.816	0.813	0	Constant	Age	Mileage	Diesel	Petrol	HP	Metalic	Automatic	CC	Doors	*	*
11	1229398784	83.1781	0.844	0.841	0	Constant	Age	Mileage	Diesel	Petrol	HP	Metalic	Automatic	CC	Doors	QuarTax	*
12	1093331072	12	0.861	0.859	1	Constant	Age	Mileage	Diesel	Petrol	HP	Metalic	Automatic	CC	Doors	QuarTax	Weight

Figure 5.6: Stepwise Selection Result for Reducing Predictors in Toyota Corolla Prices Example

exhaustive search: for each number of predictors the same subset is chosen (it therefore gives a table identical to the one in Figure 5.4). Notice that this not always be the case. In comparison, backward elimination starts with the full model, and then drops predictors one-by-one in this order: *Doors, CC, Diesel, Metalic, Automatic, Quart Tax, Petro, Weight,* and *Age* (see Figure 5.5). The R^2_{adj} and C_p measures indicate the exact same subsets that the exhaustive search suggested. In other words it correctly identifies *Doors, CC, Diesel, Metalic,* and *Automatic* as the least useful predictors. Backward elimination would yield a different model than the exhaustive search only if we decided to use less than 6 predictors. For instance, if we were limited to two predictors, then backward elimination would choose *Age* and *Weight*, whereas the exhaustive search shows that the best pair of predictors is actually *Age* and *Horsepower*. The results for stepwise regression can be seen in Figure 5.6. It chooses the same subsets as forward selection for subset sizes of 1-7 predictors. However, for 8-10 predictors, it chooses a different subset than the other methods: It decides to drop $Doors$, $QuartTax$, and $Weight$. This means that it fails to detect the best subsets for 8-10 predictors. R^2_{adj} is largest at 6 predictors (the same 6 that were selected by the other models), but C_p indicates that the full model with 11 predictors is the best fit.

This example shows that the search algorithms yield fairly good solutions, but we need to carefully determine the number of predictors to retain. It also shows the merits of running a few searches, and using the combined results to conclude on the subset to choose. There is a popular

(but false) notion that stepwise regression is superior to backward elimination and forward selection because of its ability to add and to drop predictors. This example clearly shows that it is not always so.

Finally, additional ways to reduce the dimension of the data are by using principal components (Chapter 3), and using regression trees (Chapter 7).

5.5 Exercises

Predicting Boston Housing Prices: The file BostonHousing.xls contains information collected by the US Census Service concerning housing in the area of Boston, Massachusetts. The dataset includes information on 506 census housing tracts in the Boston area. The goal is to predict the median house price in new tracts based on information such as crime rate, pollution, number of rooms, etc. The dataset contains 14 predictors, and the response is the median house price (MEDV). The table below describes each of the predictors and the response.

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

1. Why should the data be partitioned into training and validation sets? What will the training set be used for? What will the validation set be used for?

Fit a multiple linear regression model to the median house price (MEDV) as a function of CRIM, CHAS, and RM.

2. Write the equation for predicting the median house price from the predictors in the model.
3. What is the predicted median house price for a tract in the Boston area that does not bound the Charles river, has a crime rate of 0.1, and the average number of rooms is 3? What is the prediction error?
4. Reduce the number of predictors:
   (a) Which predictors are likely to be measuring the same thing among the 14 predictors? Discuss the relationship between INDUS, NOX, and TAX.
   (b) Compute the correlation table between the 13 numerical predictors and search for highly correlated pairs. These have potential redundancy and can cause multicollinearity. Choose which ones to remove based on this table.
   (c) Use exhaustive search to reduce the remaining predictors as follows: First, choose the top three models. Then run each of these models separately on the training set, and compare their predictive accuracy for the validation set. Compare RMSE and average error, as well as lift charts. Finally, give the best model.

Predicting Software Reselling Profits: Tayko Software is a software catalog firm that sells games and educational software. It started out as a software manufacturer, and added third party titles to its offerings. It has recently put together a revised collection of items in a new catalog, which it mailed out to its customers. This mailing yielded 1000 purchases. Based on these data, Tayko wants to devise a model for predicting the spending amount that a purchasing customer will yield.

The file Tayko.xls contains the following information on 1000 purchases:

FREQ	Number of transactions in last year
LAST_UPDATE	Number of days since last update to customer record
WEB	Whether customer purchased by web order at least once
GENDER	Male/Female
ADDRESS_RES	whether it is a residential address
ADDRESS_US	whether it is a US address
SPENDING (response)	Amount spent by customer in test mailing (in $)

1. Explore the spending amount by creating a pivot table for the categorical variables, and computing the average and standard deviation of spending in each category.
2. Explore the relationship between spending and each of the two continuous predictors by creating two scatter plots (SPENDING vs. FREQ, and SPENDING vs. LAST_UPDATE). Does there seem to be a linear relationship there?
3. In order to fit a predictive model for SPENDING,
   (a) Partition the 1000 records into training and validation sets. Pre-process the 4 categorical variables by creating dummy variables.
   (b) Run a multiple linear regression model for SPENDING vs. all 6 predictors. Give the estimate predictive equation.
   (c) Based on this model, what type of purchaser is most likely to spend a large amount of money?
   (d) If we used backward elimination to reduce the number of predictors, which predictor would be dropped first from the model?
   (e) Show how the prediction and the prediction error are computed for the first purchase in the validation set.
   (f) Evaluate the predictive accuracy of the model by examining its performance on the validation set.
   (g) Create a histogram of the model residuals. Do they appear to follow a normal distribution? How doest this affect the predictive performance of the model?

Predicting airfares on new routes: Several new airports have opened in major cities, opening the market for new routes (a route refers to a pair of airports), and Southwest has not announced whether it will cover routes to/from these cities. In order to price flights on these routes, a major airline collected information on 638 air routes in the United States. Some factors are known about these new routes: the distance traveled, demographics of the city where the new airport is located, and whether this city is a vacation destiny. Other factors are yet unknown (e.g., the number of passengers that will travel this route). A major unknown factor is whether Southwest or another discount airline will travel on these new routes. Southwest's strategy (point-to-point routes covering only major cities, use of secondary airports, standardized fleet, low fares) has been very different from the model followed by the older and bigger airlines (hub-and-spoke model extending to even smaller cities, presence in primary airports, variety in fleet, pursuit of high-end business travelers). The presence of discount airlines is therefore believed to to reduce the fares greatly.

The file Airfares.xls contains real data that were collected for the third quarter of 1996. They consist of the following predictors and response:

S_CODE	Starting airport's code
S_CITY	Starting city
E_CODE	Ending airport's code
E_CITY	Ending city
COUPON	Average number of coupons (a one-coupon flight is a non-stop flight, A two-coupon flight is a one stop flight, etc.) for that route
NEW	Number of new carriers entering that route between Q3-96 and Q2-97
VACATION	Whether a vacation route (Yes) or not (No).
SW	Whether Southwest Airlines serves that route (Yes) or not (No)
HI	Herfindel Index - measure of market concentration
S_INCOME	Starting city's average personal income
E_INCOME	Ending city's average personal income
S_POP	Starting city's population
E_POP	Ending city's population
SLOT	Whether either endpoint airport is slot controlled or not; This is a measure of airport congestion
GATE	Whether either endpoint airport has gate constraints or not; This is another measure of airport congestion
DISTANCE	Distance between two endpoint airports in miles
PAX	Number of passengers on that route during period of data collection
FARE (the response)	Average fare on that route

Note that some cities are served by more than one airport, and in those cases the airports are distinguished by their 3-letter code.

1. Explore the numerical predictors and response (FARE) by creating a correlation table and examining some scatter plots between FARE and those predictors. What seems to be the best single predictor of FARE?
2. Explore the categorical predictors (excluding the first 4) by computing the percentage of flights in each category. Create a pivot table that gives the average fare in each category. Which categorical predictor seems best for predicting FARE?
3. Find a model for predicting the average fare on a new route:
   (a) Partition the data into training and validation sets. The model will be fit to the training data and evaluated on the validation set.
   (b) Use stepwise regression to reduce the number of predictors. You can ignore the first four predictors (S_CODE, S_CITY, E_CODE, E_CITY). Remember to turn categorical variables (e.g., SW) into dummy variables first. Report the selected estimated model.
   (c) Repeat (b) using exhaustive search instead of stepwise regression. Compare the resulting best model to the one you obtained in (b) in terms of the predictors that are in the model.
   (d) Compare the predictive accuracy of both models (b),(c) using measures such as RMSE and average error and lift charts.
   (e) Using model (c), predict the average fare on a route with the following characteristics: COUPON=1.202, NEW=3, VACATION=No, SW=No, HI=4442.141, S_INCOME = $28,760, E_INCOME=$27,664, S_POP=4,557,004, E_POP=3,195,503, SLOT=Free, GATE=Free,
   PAX=12782, DISTANCE=1976 miles. What is a 95% predictive interval?
   (f) Predict the reduction in average fare on the above route if Southwest decides to cover this route (using model (c)).

(g) In reality, which of the factors will not be available for predicting the average fare from a new airport? (i.e., before flights start operating on those routes) Which ones can be estimated? How?

(h) Select a model that includes only factors that are available before flights begin to operate on the new route. Use exhaustive search to find such a model.

(i) Use this model to predict the average fare on the route from (10e): COUPON=1.202, NEW=3, VACATION=No, SW=No, HI=4442.141, S_INCOME=$28,760, E_INCOME = $27,664, S_POP=4,557,004, E_POP=3,195,503, SLOT=Free, GATE=Free, PAX=12782, DISTANCE=1976 miles. What is a 95% predictive interval?

(j) Compare the predictive accuracy of this model with model (c). Is this model good enough, or is it worth while re-evaluating the model once flights commence on the new route?

4. In competitive industries, a new entrant with a novel business plan can have a disruptive effect on existing firms. If the new entrants' business model is sustainable, other players are forced to respond by changing their business practices. If the goal of the analysis was to evaluate the effect of Southwest Airlines' presence on the airline industry, rather than predicting fares on new routes, how would the analysis be different? Describe technical and conceptual aspects.

Predicting Prices of Used Cars (Regression Trees): The file ToyotaCorolla.xls contains the data on used cars (Toyota Corolla) on sale during late summer of 2004 in the Netherlands. It has 1436 records containing details on 38 attributes including Price, Age, Kilometers, Horsepower, and other specifications. The goal is to predict the price of a used Toyota Corolla based on its specifications. (The example in section 4.2.1 is a subset of this dataset).

Data pre-processing: Create dummy variables for the categorical predictors (Fuel Type and Color). Split the data into training (50%), validation (30%), and test (20%) datasets.

Run a multiple linear regression using the Prediction menu in XLMiner with the output variable *Price* and input variables *Age_08_04, KM, Fuel_Types, HP, Automatic, Doors, Quarterly_Tax, Mfg_Guarantee, Guarantee_Period, Airco, Automatic_Airco, CD_Player, Powered_Windows, Sport_Model,* and *Tow_Bar*.

1. What appear to be the 3-4 most important car specifications for predicting the car's price?
2. Using metrics you consider useful, assess the performance of the model in predicting prices.

Chapter 6

Three Simple Classification Methods

6.1 Introduction

We start by introducing three methods that are simple and intuitive. The first is used mainly as a baseline for comparison with more advanced methods. It is similar to a "no-predictor" model. The last two (Naive Bayes and k-nearest neighbor) are very widely used in practice. All methods share the property that they make nearly no assumptions about the structure of the data, and are therefore very data-driven as opposed to model-driven. In the following chapters we will move to more model-driven methods. There is, of course, a tradeoff between simplicity and power, but in the presence of large datasets the simple methods often work surprisingly well.

We start with two examples which include categorical predictors. These are used to illustrate the first two methods (Naive rule and Naive Bayes). We use a third example for illustrating the k-nearest neighbor method, which has continuous predictors and therefore is more suitable for illustrative purposes.

6.1.1 Example 1: Predicting Fraudulent Financial Reporting

An auditing firm has many large companies as customers. Each customer submits an annual financial report to the firm, which is then audited. To avoid being involved in any legal charges against it, the firm wants to detect whether a company submitted a fraudulent financial report . In this case each company (customer) is a record, and the response of interest, $Y = \{fraudulent, truthful\}$, has two classes that a company can be classified into: C_1=fraudulent and C_2=truthful. The only other piece of information that the auditing firm has on its customers is whether or not legal charges were filed against them. The firm would like to use this information to improve its estimates of fraud. Thus "X=legal charges" is a single (categorical) predictor with two categories: whether legal charges were filed (1) or not (0).

The auditing firm has data on 1500 companies which it investigated in the past. For each company they have information on whether it is fraudulent or truthful and whether legal charges were filed against it. After partitioning the data into a training set (1000 firms) and a validation set (500 firms), the following counts were obtained from the training set:

	legal charges ($X = 1$)	no legal charges ($X = 0$)	total
fraudulent (C_1)	50	50	100
truthful (C_2)	180	720	900
total	230	770	1000

How can this information be used to classify a certain customer as fraudulent or truthful?

6.1.2 Example 2: Predicting Delayed Flights

Predicting flight delays would be useful to a variety of organizations – airport authorities, airlines, aviation authorities. At times, joint task forces have been formed to address the problem. Such an organization, if it were to provide ongoing real-time assistance with flight delays, would benefit from some advance notice about flights that are likely to be delayed.

In this simplified illustration, we look at six predictors (see table below). The outcome of interest is whether the flight is delayed or not (delayed means arrive more than 15 minutes late). Our data consist of all flights from the Washington, DC area into the New York City area during January 2004. The percent of delayed flights among these 2346 flights is 18%. The data were obtained from the Bureau of Transportation Statistics (available on the web at `www.transtats.bts.gov`).

The goal is to accurately predict whether a new flight, not in this dataset, will be delayed or not.

A record is a particular flight. The response is whether the flight was delayed, and thus it has two classes (1="Delayed", and 0="On time"). In addition, there is information collected on the following predictors:

Day of Week	Coded as: 1=Monday, 2=Tuesday,..., 7=Sunday
Departure time	Broken down into 18 intervals between 6:00AM and 10:00PM.
Origin	Three airport codes: DCA (Reagan National), IAD (Dulles), BWI (Baltimore-Washington Intl)
Destination	Three airport codes: JFK (Kennedy), LGA (LaGuardia), EWR (Newark)
Carrier	8 airline codes: CO (Continental), DH (Atlantic Coast), DL (Delta), MQ (American Eagle), OH (Comair), RU (Continental Express), UA (United), and US (USAirways).
Weather	coded as 1 if there was a weather-related delay.

6.2 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. For instance, in the auditing example above, the naive rule would classify all customers as being truthful, because 90% of the investigated companies in the training set were found to be truthful. Similarly, all flights would be classified as being on-time, because the majority of the flights in the dataset (82%) were not delayed. The naive rule is used mainly as a baseline for evaluating the performance of more complicated classifiers. Clearly, a classifier that uses external predictor information (on top of the class membership allocation), should out-perform the naive rule. There are various performance measures based on the naive rule, which measure how much better a certain classifier performs compared to the naive rule. One example is the "multiple-R-squared" 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 "Evaluating Goodness-of- Fit" in Chapter 8).

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 for a new record. In both cases the predictions rely solely on the y information and exclude any additional predictor information.

6.3 Naive Bayes

The Naive Bayes classifier is a more sophisticated method than the naive rule. The main idea is to integrate the information given in a set of predictors into the naive rule to obtain more accurate classifications. In other words, the probability of a record belonging to a certain class is now evaluated not only based on the prevalence of that class but also on the additional information that is given on that record in term of its X information.

In contrast to the other classifiers discussed here, Naive Bayes works only with predictors that are categorical. Numerical predictors must be binned and converted to categorical variables before the Naive Bayes classifier can use them. In the two examples above all predictors are categorical. Notice that the originally-continuous variable "departure time" in the flight delay example was binned into 18 categories.

The naive Bayes method is very useful when very large datasets are available. For instance, web-search companies like Google use naive Bayes classifiers to correct misspellings that users type in. When you type a phrase that includes a misspelled word into Google it suggests a spelling correction for the phrase. The suggestion(s) are based on information not only on the frequencies of similarly-spelled words that were typed by millions of other users, but also on the other words in the phrase.

6.3.1 Bayes Theorem

The naive Bayes method is based on conditional probabilities, and in particular on a fundamental theorem in probability theory, called Bayes Theorem. This clever theorem provides the probability of a prior event, given that a certain subsequent event has occurred. For instance, what is the probability that a firm submitted a fraudulent financial report if we know it is sued for such fraud? Clearly the fraudulent act precedes the legal charges.

A conditional probability of event A given event B (denoted by $P(A|B)$) represents the chances of event A occurring *only* under the scenario that event B occurs. In the auditing example we are interested in P(fraudulent financial report | legal charges). Since conditioning on an event means that we have additional information (e.g., we know that legal charges were filed against them), the uncertainty is reduced.

In the context of classification, Bayes theorem provides a formula for updating the probability that a given record belongs to a class, given the record's attributes. Suppose that we have m classes, $C_1, C_2, \ldots, C_m$ and we know that the proportion of records in these classes are $P(C_1), P(C_2), \ldots, P(C_m)$. We want to classify a new record with a set of predictor values $x_1, x_2, \ldots, x_p$ on the basis of these values. If we know the probability of occurrence of the predictor values $X_1, X_2, \ldots, X_p$ within each class[1], Bayes theorem gives us the following formula to compute the probability that the record belongs to class C_i.

$$P(C_i|X_1, \ldots, X_p) = \frac{P(X_1, \ldots X_p|C_i)P(C_i)}{P(X_1, \ldots, X_p|C_1)P(C_1) + \cdots + P(X_1, \ldots, X_p|C_m)P(C_m)} \tag{6.1}$$

[1]We use the notation $P(X_1, X_2, \ldots, X_p)$ to denote the probability that event X_1 occurs AND event X_2 occurs, ... AND X_p occurs.

This is known as the posterior probability of belonging to class C_i (which includes the predictor information). This is in contrast to the prior probability, $P(C_i)$, of belonging to class C_i in the absence of any information about its attributes.

To classify a record, we compute its chance of belonging to each of the classes by computing $P(C_i|X_1, \ldots, X_p)$ for each class i. We then classify the record to the class that has the highest probability. In practice, we only need to compute the numerator of (6.1), since the denominator is the same for all classes. To compute the numerator we need two pieces of information:

1. The proportions of each class in the population $(P(C_1), \ldots P(C_m))$
2. The probability of occurrence of the predictor values $X_1, X_2 ... X_p$ within each class.

Assuming that our dataset is a representative sample of the population, we can estimate the population proportions of each class and the predictor occurrence within each class from the training set. To estimate $P(X_1, X_2 ... X_p|C_i)$, we count the number of occurrences of the values $x_1, x_2 ... x_p$ in class C_i and divide them by the total number of observations in that class.

Returning to the financial reporting fraud example we estimate the class proportions from the training set, using the count table, by $\hat{P}(C_1) = 100/1000 = 0.1$ and $\hat{P}(C_2) = 900/1000 = 0.9$. Recall that the naive rule classifies every firm as being truthful (the majority class). The additional information on whether or not legal charges were filed gives us the probability of occurrence of the predictor within each class:

$$\begin{aligned}
\hat{P}(X = 1|\, C_1) &= \hat{P}(legal\ charges \mid fraudulent) = 50/100 = 0.5 \\
\hat{P}(X = 1|\, C_2) &= \hat{P}(legal\ charges \mid truthful) = 180/900 = 0.2 \\
\hat{P}(X = 0|\, C_1) &= \hat{P}(no\ charges \mid fraudulent) = 50/100 = 0.5 \\
\hat{P}(X = 0|\, C_2) &= \hat{P}(no\ charges \mid truthful) = 720/900 = 0.8
\end{aligned}$$

Now, consider a company that has just been charged with fraudulent financial reporting. To classify this company as fraudulent or truthful, we compute the probabilities of belonging to each of the two classes:

$$\begin{aligned}
\hat{P}(fraudulent|legal\ charges\ filed) &\propto \hat{P}(legal\ charges\ filed \mid fraudulent) P(fraudulent) \\
&= (0.5)(0.1) = 0.05 \\
\hat{P}(truthful|legal\ charges\ filed) &\propto \hat{P}(legal\ charges\ filed \mid truthful) P(truthful) \\
&= (0.2)(0.9) = 0.018
\end{aligned}$$

This indicates that this firm is more likely to be fraudulent. Notice that this is in contrast to the Naive rule classification which ignores completely the legal charges information and classifies the firm as truthful.

6.3.2 A Practical Difficulty and a Solution: From Bayes to Naive Bayes

The difficulty with using this formula is that if the number of predictors, p, is even modestly large, say, 20, and the number of classes, m is 2, even if all predictors are binary, we would need a large dataset with several million observations to get reasonable estimates for $P(X_1, X_2, \cdots, X_p|C_i)$, the probability of observing a record with the attribute vector $(x_1, x_2, \ldots, x_p)$. In fact the vector may not even be present in our training set for all classes, as required by the formula; it may even be missing in our entire dataset! Consider the flight delays example where we have 6 (categorical) predictors, most of them having more than two categories, and a binary response (delayed/on-time). What are the chances of finding delayed and on-time flights within our 2346 flights database for a combination such as a Delta flight from DCA to LGA between 10AM-11AM on a Sunday with good weather? (In fact, there is no such flight in the training data). Another example is in predicting voting, where

even a sizeable dataset may not contain many individuals who are male hispanics with high income from the midwest who voted in the last election, did not vote in the prior election, have 4 children, are divorced, etc.

A solution that has been widely used is based on making the simplifying assumption of predictor independence. If it is reasonable to assume that the predictors are all mutually independent within each class, we can considerably simplify the expression and make it useful in practice. Independence of the predictors within each class gives us the following simplification which follows from the product rule for probabilities of independent events (the probability of occurrence of multiple events is the product of the probabilities of the individual event occurrences):

$$P(X_1, X_2, \cdots, X_m|C_i) = P(X_1|C_i)P(X_2|C_i)P(X_3|C_i)\cdots P(X_m|C_i) \tag{6.2}$$

The terms on the right are estimated from frequency counts in the training data, with the estimate of $P(X_j|C_i)$ being equal to the number of occurrences of the value x_j in class C_i divided by the total number of records in that class.

For example, instead of estimating the probability of delay on a Delta flight from DCA to LGA on Sunday from 10-11AM with good weather by tallying such flights (which do not exist), we multiply the probability that a Delta flight is delayed times the probability that a DCA to LGA flight is delayed times the probability that a flight in the Sunday 10-11AM slot is delayed times the probability that a good weather flight is delayed.

We would like to have each possible value for each predictor to be available in the training data. If this is not true for a particular predictor value for a certain class, the estimated probability will be zero for the class for records with that predictor value. For example, if there are no delayed flights from DCA airport in our dataset, we estimate the probability of delay for new flights from DCA to be zero. Often this is reasonable so we can relax our requirement of having every possible value for every predictor being present in the training data.

In any case the number of required records will be far fewer than would be required if we did not make the independence assumption. This is a very simplistic assumption since the predictors are very likely to be correlated. Surprisingly this "Naive Bayes" approach, as it is called, does work well in practice where there are many variables and they are binary or categorical with a few discrete levels.

Let us consider a small numerical example to illustrate how the exact Bayes calculations differ from the Naive Bayes. Consider the following 10 companies. For each company we have information on whether charges were filed against it or not, whether it is a small or large company, and whether (after investigation) it turned out to be fraudulent or truthful in its financial reporting.

Charges filed?	Company size	Status
y	small	truthful
n	small	truthful
n	large	truthful
n	large	truthful
n	small	truthful
n	small	truthful
y	small	fraudulent
y	large	fraudulent
n	large	fraudulent
y	large	fraudulent

We first compute the conditional probabilities of fraud, given each of the four possible combinations $\{y, small\}, \{y, large\}, \{n, small\}, \{n, large\}$. For the combination $Charges = y$, $Size = small$, the numerator is the proportion of $\{y, small\}$ pairs among the fraudulent companies multiplied by the proportion of fraudulent companies $((1/4)(4/10))$. The denominator is the proportion of $\{y, small\}$

pairs among all 10 companies (2/10). A similar argument is used to construct the 3 other conditional probabilities:

$$\begin{aligned} P(fraud|Charges = y, Size = small) &= \frac{(1/4)(4/10)}{2/10} = 0.5 \\ P(fraud|Charges = y, Size = large) &= \frac{(2/4)(4/10)}{2/10} = 1 \\ P(fraud|Charges = n, Size = small) &= \frac{(0/4)(4/10)}{3/10} = 0 \\ P(fraud|Charges = n, Size = large) &= \frac{(1/4)(4/10)}{3/10} = 0.33 \end{aligned}$$

Now, we compute the Naive Bayes probabilities. For the conditional probability of fraud given $Charges = y$, $Size = small$, the numerator is a multiplication of the proportion of $Charges = y$ instances among the fraudulent companies, times the proportion of $Size = small$ instances among the fraudulent companies, times the proportion of fraudulent companies: (3/4)(1/4)(4/10)=0.075. However, to get the actual probabilities we need also compute the numerator for the conditional probability of truth given $Charges = y$, $Size = small$: (1/6)(4/6)(6/10)=0.067. The denominator is then the sum of these two conditional probabilities (0.075+0.067=0.14). The conditional probability of fraud given $Charges = y$, $Size = small$ is therefore $0.075/0.14 = 0.53$. In a similar fashion we compute all four conditional probabilities:

$$\begin{aligned} P_{NB}(fraud|Charges = y, Size = small) &= \frac{(3/4)(1/4)(4/10)}{(3/4)(1/4)(4/10) + (1/6)(4/6)(6/10)} = 0.53 \\ P_{NB}(fraud|Charges = y, Size = large) &= 0.87 \\ P_{NB}(fraud|Charges = n, Size = small) &= 0.07 \\ P_{NB}(fraud|Charges = n, Size = large) &= 0.31 \end{aligned}$$

Notice how close these Naive Bayes probabilities are to the exact Bayes probabilities. Although they are not equal, both would lead to the exact same classification for a cutoff of 0.5 (and many other values). It is often the case that the rank ordering of probabilities are even closer to the exact Bayes method than are the probabilities themselves, and it is the rank orderings that matter, for classification purposes.

Figure 6.1 shows the estimated prior probabilities of a delayed flight and an on-time flight, and also the corresponding conditional probabilities as a function of the predictor values. The data were first partitioned into training and validation sets (with a 60%-40% ratio), and then a Naive Bayes classifier was applied to the training set. Notice that the conditional probabilities in the output can be computed simply by using pivot tables in Excel, looking at the percent of records in a cell relative to the entire class. This is illustrated in Table 6.1, which displays the percent of delayed (or on-time) flights by destination airport, as a percentage of the total delayed (or on-time) flights.

Note that in this example there are no predictor values that were not represented in the training data except for on-time flights (Class=0) when the weather was bad (weather=1). When the weather was bad, all flights in the training set were delayed.

To classify a new flight, we compute the probability that it will be delayed and the probability that it will be on-time. Recall that since both will have the same denominator, we can just compare the numerators. Each numerator is computed by multiplying all the conditional probabilities of the relevant predictor values and finally multiplying by the proportion of that class (in this case $\hat{P}(delay) = 0.19$). For example, to classify a Delta flight from DCA to LGA between 10AM-11AM

Prior class probabilities

According to relative occurrences in training data

Class	Prob.	
1	0.193792581	<-- Success Class
0	0.806207419	

Conditional probabilities

	Classes-->			
Input Variables	1		0	
	Value	Prob	Value	Prob
CARRIER	CO	0.06640625	CO	0.038497653
	DH	0.33984375	DH	0.243192488
	DL	0.109375	DL	0.2
	MQ	0.1796875	MQ	0.112676056
	OH	0.01171875	OH	0.017840376
	RU	0.21484375	RU	0.170892019
	UA	0.0078125	UA	0.016901408
	US	0.0703125	US	0.2
DAY_OF_WEEK	1	0.203125	1	0.128638498
	2	0.16015625	2	0.139906103
	3	0.12890625	3	0.152112676
	4	0.12890625	4	0.159624413
	5	0.1640625	5	0.181220657
	6	0.0703125	6	0.131455399
	7	0.14453125	7	0.107042254
DEP_TIME_BLK	0600-0659	0.03515625	0600-0659	0.061971831
	0700-0759	0.05078125	0700-0759	0.060093897
	0800-0859	0.0546875	0800-0859	0.071361502
	0900-0959	0.0234375	0900-0959	0.053521127
	1000-1059	0.01953125	1000-1059	0.057276995
	1100-1159	0.01953125	1100-1159	0.038497653
	1200-1259	0.0546875	1200-1259	0.062910798
	1300-1359	0.05078125	1300-1359	0.068544601
	1400-1459	0.15234375	1400-1459	0.110798122
	1500-1559	0.08203125	1500-1559	0.064788732
	1600-1659	0.07421875	1600-1659	0.078873239
	1700-1759	0.15625	1700-1759	0.094835681
	1800-1859	0.03125	1800-1859	0.043192488
	1900-1959	0.08984375	1900-1959	0.040375587
	2000-2059	0.01953125	2000-2059	0.030985915
	2100-2159	0.0859375	2100-2159	0.061971831
DEST	EWR	0.38671875	EWR	0.283568075
	JFK	0.1875	JFK	0.176525822
	LGA	0.42578125	LGA	0.539906103
ORIGIN	BWI	0.09375	BWI	0.068544601
	DCA	0.484375	DCA	0.635680751
	IAD	0.421875	IAD	0.295774648
Weather	0	0.92578125	0	1
	1	0.07421875	1	0

Figure 6.1: Output from Naive Bayes Classifier Applied to Flight Delays (Training) Data

	Status		
Destination	Delayed	On-time	Total
EWR	38.67%	28.36%	30.36%
JFK	18.75%	17.65%	17.87%
LGA	42.58%	53.99%	51.78%
Total	100%	100%	100%

Table 6.1: Pivot Table of Delayed and On-Time Flights By Destination Airport. Cell Numbers are Column Percentages

on a Sunday with good weather, we compute the numerators:

$$\begin{aligned}\hat{P}(delayed| & \quad Carrier = DL,\ Day = 7,\ Time = 1000-1059,\ Dest = LGA,\ Origin = DCA,\ Weather = 0\\ & \propto (0.11)(0.14)(0.020)(0.43)(0.48)(0.93)(0.19) = 0.000011\\ \hat{P}(on-time| & \quad Carrier = DL,\ Day = 7,\ Time = 1000-1059,\ Dest = LGA,\ Origin = DCA,\ Weather = 0\\ & \propto (0.2)(0.11)(0.06)(0.54)(0.64)(1)(0.81) = 0.00034\end{aligned}$$

It is therefore more likely that the above flight will be on-time. Notice that a record with such a combination of predictors does not exist in the training set, and therefore we use the naive Bayes rather than the exact Bayes.

To compute the actual probability, we divide each of the numerators by their sum:

$$\begin{aligned}\hat{P}(delayed| & \quad Carrier = DL,\ Day = 7,\ Time = 1000-1059,\ Dest = LGA,\ Origin = DCA,\ Weather = 0\\ & = 0.000011/(0.000011+0.00034) = 0.03\\ \hat{P}(on-time| & \quad Carrier = DL,\ Day = 7,\ Time = 1000-1059,\ Dest = LGA,\ Origin = DCA,\ Weather = 0\\ & = 0.00034/(0.000011+0.00034) = 0.97\end{aligned}$$

Of course we rely on software to compute these probabilities for any records of interest (in the training set, the validation set, or for scoring new data). Figure 6.2 shows the estimated probabilities and classifications for a sample of flights in the validation set.

Finally, to evaluate the performance of the naive Bayes classifier for our data, we use the classification matrix, lift charts, and all the measures that were described in Chapter 4. For our example, the classification matrices for the training and validation sets are shown in Figure 6.3. We see that the overall error level is around 18% for both the training and validation data. In comparison, a naive rule which would classify all 880 flights in the validation set as on-time, would have missed the 172 delayed flights resulting in a 20% error level. In other words, the Naive Bayes is only slightly less accurate. However, examining the lift chart (Figure 6.4) shows the strength of the Naive Bayes in capturing the delayed flights well.

6.3.3 Advantages and Shortcomings of the Naive Bayes Classifier

The Naive Bayes classifier's beauty is in its simplicity, computational efficiency, and its good classification performance. In fact, it often outperforms more sophisticated classifiers even when the underlying assumption of independent predictors is far from true. This advantage is especially pronounced when the number of predictors is very large. There are three main issues that should be kept in mind however.

First, the Naive Bayes classifier requires a very large number of records to obtain good results.

Second, where a predictor category is not present in the training data, Naive Bayes assumes that a new record with that category of the predictor has zero probability. This can be a problem if this rare predictor value is important. For example, assume the target variable is "bought high value

XLMiner : Naive Bayes - Classification of Validation Data

Data range	[Flight Delays data, figs, analysis for NB example.xls]'Data_Partition1'!$C$1340:$H$2219	Back to Navigator

Cut off Prob.Val. for Success (Updatable)	0.5	(Updating the value here will NOT update value in summary report)

Row Id.	Predicted Class	Actual Class	Prob. for 1 (success)	CARRIER	DAY_OF_WEEK	DEP_TIME_BLK	DEST	ORIGIN	Weather
2	0	0	0.160552079	DH	4	1600-1659	JFK	DCA	0
3	0	0	0.197147877	DH	4	1200-1259	LGA	IAD	0
7	0	0	0.248536067	DH	4	1200-1259	JFK	IAD	0
8	0	0	0.263631618	DH	4	1600-1659	JFK	IAD	0
11	0	0	0.281467602	DH	4	2100-2159	LGA	IAD	0
13	0	0	0.025209812	DL	4	0900-0959	LGA	DCA	0
14	0	0	0.048830719	DL	4	1200-1259	LGA	DCA	0
15	0	0	0.07510312	DL	4	1400-1459	LGA	DCA	0
16	0	0	0.088673655	DL	4	1700-1759	LGA	DCA	0
22	0	0	0.113149152	MQ	4	1300-1359	LGA	DCA	0
24	0	0	0.179013723	MQ	4	1500-1559	LGA	DCA	0
25	0	0	0.277045255	MQ	4	1900-1959	LGA	DCA	0
28	0	0	0.018897189	US	4	1100-1159	LGA	DCA	0
33	0	0	0.366861013	RU	4	1400-1459	EWR	BWI	0
34	0	0	0.409792076	RU	4	1700-1759	EWR	BWI	0
40	0	0	0.44593539	DH	4	1700-1759	EWR	IAD	0
42	0	0	0.403842858	DH	4	2100-2159	EWR	IAD	0
46	0	0	0.343153176	RU	4	1900-1959	EWR	DCA	0
47	0	0	0.244033602	RU	4	1400-1459	EWR	DCA	0
50	0	0	0.18094682	RU	4	1600-1659	EWR	DCA	0
57	0	1	0.097462126	DH	5	1000-1059	LGA	IAD	0

Figure 6.2: Estimated Probability of Delay for a Sample of the Validation Set

Training Data scoring - Summary Report

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

Classification Confusion Matrix		
	Predicted Class	
Actual Class	1	0
1	43	213
0	35	1030

Error Report			
Class	# Cases	# Errors	% Error
1	256	213	83.20
0	1065	35	3.29
Overall	1321	248	18.77

Validation Data scoring - Summary Report

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

Classification Confusion Matrix		
	Predicted Class	
Actual Class	1	0
1	30	142
0	15	693

Error Report			
Class	# Cases	# Errors	% Error
1	172	142	82.56
0	708	15	2.12
Overall	880	157	17.84

Figure 6.3: Classification Matrices for Flight Delays Using a Naive Bayes Classifier

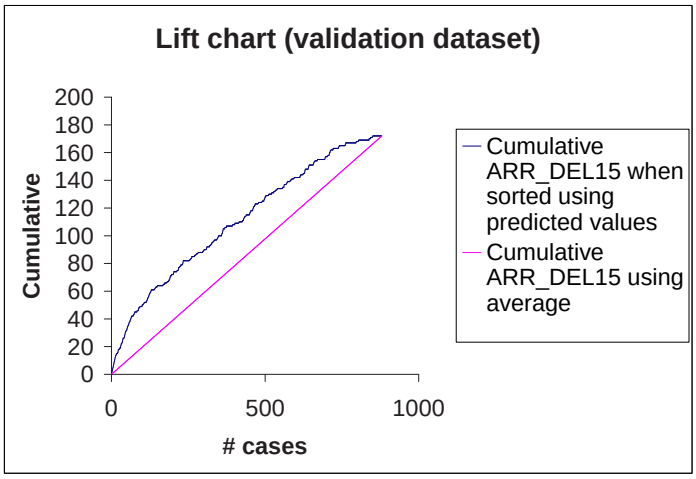


Figure 6.4: Lift Chart of Naive Bayes Classifier Applied to Flight Delays Data

life insurance" and a predictor category is "own yacht". If the training data have no records with "owns yacht"=1, for any new records where "owns yacht"=1, Naive Bayes will assign a probability of 0 to the target variable "bought high value life insurance". With no training records with "owns yacht"=1, of course, no data mining technique will be able to incorporate this potentially important variable into the classification model - it will be ignored. With Naive Bayes, however, the absence of this predictor actively "outvotes" any other information in the record to assign a 0 to the target value (when, in this case, it has a relatively good chance of being a 1). The presence of a large training set (and judicious binning of continuous variables, if required) help mitigate this effect.

Finally, the good performance is obtained when the goal is classification or ranking of records according to their probability of belonging to a certain class. However, when the goal is to actually *estimate* the probability of class membership, this method provides very biased results. For this reason the Naive Bayes method is rarely used in credit scoring[2].

6.4 k-Nearest Neighbor (k-NN)

The idea in k-Nearest Neighbor methods is to identify k observations in the training dataset that are similar to a new record that we wish to classify. We then use these similar (neighboring) records to classify the new record into a class, assigning the new record to the predominant class among these neighbors. Denote by $(x_1, x_2, \cdots, x_p)$ the values of the predictors for this new record. We look for records in our training data that are similar or "near" to the record to be classified in the predictor space, i.e., records that have values close to $x_1, x_2, \ldots, x_p$. Then, based on the classes to which those proximate records belong, we assign a class to the record that we want to classify.

The k-Nearest Neighbor algorithm is a classification method that does not make assumptions about the form of the relationship between the class membership (Y) and the predictors $X_1, X_2, \cdots, X_p$. This is a non-parametric method because it does not involve estimation of parameters in an assumed function form such as the linear form that we encountered in linear regression. Instead, this method draws information from similarities between the predictor values of the records in the data set.

[2]Larsen, K. "Generalized Naive Bayes Classifiers", (2005), *SIGKDD Explorations, vol 7(1), pp. 76-81.*

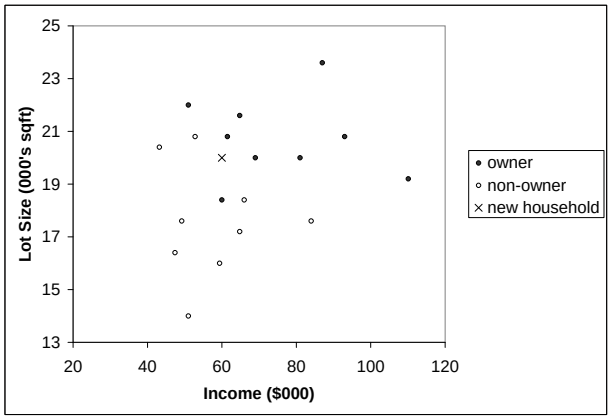


Figure 6.5: Scatterplot of Lot Size Vs. Income for the 18 Households in the Training Set and the New Household To Be Classified

The central issue here is how to measure the distance between records based on their predictor values. The most popular measure of distance is the Euclidean distance. The Euclidean distance between two records $(x_1, x_2, \cdots, x_p)$ and $(u_1, u_2, \cdots, u_p)$ is $\sqrt{(x_1 - u_1)^2) + (x_2 - u_2)^2 + \cdots + (x_p - u_p)^2}$. For simplicity, we continue here only with the Euclidean distance, but you will find a host of other distance metrics in Chapters 12 (Cluster Analysis) and 10 (Discriminant Analysis) for both numerical and categorical variables. Note that in most cases predictors should first be standardized before computing Euclidean distance, to equalize the scales that the different predictors may have.

After computing the distances between the record to be classified and existing records, we need a rule to assign a class to the record to be classified, based on the classes of its neighbors. The simplest case is $k = 1$ where we look for the record that is closest (the nearest neighbor) and classify the new record as belonging to the same class as its closest neighbor. It is a remarkable fact that this simple, intuitive idea of using a single nearest neighbor to classify records can be very powerful when we have a large number of records in our training set. It is possible to prove that the misclassification error of the 1-Nearest Neighbor scheme has a misclassification rate that is no more than twice the error when we know exactly the probability density functions for each class.

The idea of the 1-Nearest Neighbor can be extended to $k > 1$ neighbors as follows:

1. Find the nearest k neighbors to the record to be classified

2. Use a majority decision rule to classify the record, where the record is classified as a member of the majority class of the k neighbors.

6.4.1 Example 3: Riding Mowers

A riding-mower manufacturer would like to find a way of classifying families in a city into those likely to purchase a riding mower and those not likely to buy one. A pilot random sample of 12 owners and 12 non-owners in the city is undertaken. The data are shown and plotted in Table 6.2. We first partition the data into training data (18 households) and validation data (6 households). Obviously this dataset is too small for partitioning, but we continue with this for illustration purposes. The training set is shown in Figure 6.5. Now consider a new household with $60,000 income and lot size 20,000 ft (also shown in Figure 6.5). Among the households in the training set, the closest one

Table 6.2: Lot Size, Income, and Ownership of a Riding Mower for 24 Households

Household number	Income ($ 000's)	Lot Size (000's ft^2)	Ownership of, riding mower
1	60	18.4	Owner
2	85.5	16.8	Owner
3	64.8	21.6	Owner
4	61.5	20.8	Owner
5	87	23.6	Owner
6	110.1	19.2	Owner
7	108	17.6	Owner
8	82.8	22.4	Owner
9	69	20	Owner
10	93	20.8	Owner
11	51	22	Owner
12	81	20	Owner
13	75	19.6	Non-Owner
14	52.8	20.8	Non-Owner
15	64.8	17.2	Non-Owner
16	43.2	20.4	Non-Owner
17	84	17.6	Non-Owner
18	49.2	17.6	Non-Owner
19	59.4	16	Non-Owner
20	66	18.4	Non-Owner
21	47.4	16.4	Non-Owner
22	33	18.8	Non-Owner
23	51	14	Non-Owner
24	63	14.8	Non-Owner

to the new household (in Euclidean distance after normalizing income and lot size) is household #4, with $61,500 income and lot size 20,800 ft. If we use a 1-NN classifier, we would classify the new household as an owner, like household #4. If we use $k = 3$, then the three nearest households are #4, #9, and #14. The first two are owners of riding mowers, and the last is a non-owner. The majority vote is therefore "owner", and the new household would be classified as an owner.

6.4.2 Choosing k

The advantage of choosing $k > 1$ is that higher values of k provide smoothing that reduces the risk of overfitting due to noise in the training data. Generally speaking, if k is too low, we may be fitting to the noise in the data. However, if k is too high, we will miss out on the method's ability to capture the local structure in the data, one of its main advantages. In the extreme, $k = n$ = the number of records in the training dataset. In that case we simply assign all records to the majority class in the training data irrespective of the values of $(x_1, x_2, \cdots, x_p)$, which coincides with the Naive Rule!

This is clearly a case of over-smoothing in the absence of useful information in the predictors about the class membership. In other words, we want to balance between overfitting to the predictor information and ignoring this information completely. A balanced choice depends greatly on the nature of the data. The more complex and irregular the structure of the data, the lower the optimum value of k. Typically, values of k fall in the range between 1 and 20. Often an odd number is chosen, to avoid ties.

So how is k chosen? Answer: we choose that k which has the best classification performance. We use the training data to classify the records in the validation data, then compute error rates

Validation error log for different k

Value of k	% Error Training	% Error Validation	
1	0.00	33.33	
2	16.67	33.33	
3	11.11	33.33	
4	22.22	33.33	
5	11.11	33.33	
6	27.78	33.33	
7	22.22	33.33	
8	22.22	16.67	<--- Best k
9	22.22	16.67	
10	22.22	16.67	
11	16.67	33.33	
12	16.67	16.67	
13	11.11	33.33	
14	11.11	16.67	
15	5.56	33.33	
16	16.67	33.33	
17	11.11	33.33	
18	50.00	50.00	

Figure 6.6: Misclassification Rate of Validation Set for Different Choices of k

for various choices of k. For our example, if we choose $k = 1$ we will classify in a way that is very sensitive to the local characteristics of the training data. On the other hand, if we choose a large value of k such as $k = 18$ we would simply predict the most frequent class in the dataset in all cases. This is a very stable prediction but it completely ignores the information in the predictors. To find a balance we examine the misclassification rate (of the validation set) that results for different choices of k between 1-18. This is shown in Figure 6.6. We would choose $k = 8$, which minimizes the misclassification rate in the validation set. Note, however, that now the validation set is used as an addition to the training set and does not reflect a "hold-out" set as before. Ideally, we would want a third test set to evaluate the performance of the method on data that it did not see.

6.4.3 k-NN for a Quantitative Response

The idea of k-NN can be readily extended to predicting a continuous value (as is our aim with multiple linear regression models). Instead of taking a majority vote of the neighbors to determine class, we take the average response value of the k nearest neighbors to determine the prediction. Often this average is a weighted average with the weight decreasing with increasing distance from the point at which the prediction is required.

6.4.4 Advantages and Shortcomings of k-NN Algorithms

The main advantage of k-NN methods is their simplicity and lack of parametric assumptions. In the presence of a large enough training set, these methods perform surprisingly well, especially when each class is characterized by multiple combinations of predictor values. For instance, in the flight delays example there are likely to be multiple combinations of carrier-destination-arrival-time etc.

that characterize delayed flights vs. on-time flights.

There are two difficulties with the practical exploitation of the power of the k-NN approach. First, while there is no time required to estimate parameters from the training data (as would be the case for parametric models such as regression), the time to find the nearest neighbors in a large training set can be prohibitive. A number of ideas have been implemented to overcome this difficulty. The main ideas are:

- Reduce the time taken to compute distances by working in a reduced dimension using dimension reduction techniques such as principal components analysis (Chapter 3).
- Use sophisticated data structures such as search trees to speed up identification of the nearest neighbor. This approach often settles for an "almost nearest" neighbor to improve speed.
- Edit the training data to remove redundant or "almost redundant" points to speed up the search for the nearest neighbor. An example is to remove records in the training set that have no effect on the classification because they are surrounded by records that all belong to the same class.

Second, the number of records required in the training set to qualify as large increases exponentially with the number of predictors p. This is because the expected distance to the nearest neighbor goes up dramatically with p unless the size of the training set increases exponentially with p. This phenomenon is knows as "the curse of dimensionality". The curse of dimensionality is a fundamental issue pertinent to all classification, prediction and clustering techniques. This is why we often seek to reduce the dimensionality of the space of predictor variables through methods such as selecting subsets of the predictors for our model or by combining them using methods such as principal components analysis, singular value decomposition, and factor analysis. In the artificial intelligence literature dimension reduction is often referred to as factor selection or feature extraction.

6.5 Exercises

Personal loan acceptance: Universal Bank is a relatively young bank growing rapidly in terms of overall customer acquisition. The majority of these customers are liability customers (depositors) with varying sizes of relationship with the bank. The customer base of asset customers (borrowers) is quite small, and the bank is interested in expanding this base rapidly to bring in more loan business. In particular, it wants to explore ways of converting its liability customers to personal loan customers (while remaining as depositors).

A campaign the bank ran for liability customers last year showed a healthy conversion rate of over 9% success. This has encouraged the retail marketing department to devise smarter campaigns with better target marketing. The goal of our analysis is to model the previous campaign's customer behavior to analyze what combination of factors make a customer more likely to accept a personal loan. This will serve as the basis for the design of a new campaign.

The file UniversalBank.xls contains data on 5000 customers. The data include customer demographic information (age, income, etc.), the customer's relationship with the bank (mortgage, securities account, etc.), and the customer response to the last personal loan campaign (Personal Loan). Among these 5000 customers only 480 (= 9.6%) accepted the personal loan that was offered to them in the previous campaign.

Partition the data into training (60%) and validation (40%) sets.

1. Using the naive rule on the training set, classify a customer with the following characteristics: Age=40, Experience=10, Income=84, Family=2, CCAvg=2, Education=2, Mortgage=0, Securities Account=0, CD Account=0, Online=1 CreditCard=1.
2. Compute the confusion matrix for the validation set based on the naive rule.
3. Perform a k-nearest neighbor classification with all predictors except zipcode using $k = 1$. Remember to transform categorical predictors with more than 2 categories into dummy variables first. Specify the "success" class as 1 (loan acceptance), and use the default cutoff value of 0.5. How would the above customer be classified?
4. What is a choice of k that balances between overfitting and ignoring the predictor information?
5. Show the classification matrix for the validation data that results from using the "best" k.
6. Classify the above customer using the "best" k.
7. Re-partition the data, this time into training, validation, and test sets (50%-30%-20%). Apply the k-NN method with the k chosen above. Compare the confusion matrix of the test set with that of the training and validation sets. Comment on the differences and their reason.

Automobile accidents: The file Accidents.xls contains information on 42,183 actual automobile accidents in 2001 in the US that involved one of three levels of injury: "no injury," "injury," or "fatality." For each accident, additional information is recorded such as day of week, weather conditions, and road type. A firm might be interested in developing a system for quickly classifying the severity of an accident, based upon initial reports and associated data in the system (some of which rely on GPS-assisted reporting).

Partition the data into validation and training sets (60%, 40%).

1. Using the information in the training set, if an accident has just been reported and no further information is available, what injury level is predicted? Why?

2. Select the first 10 records in the dataset and look only at the response (MAX_SEV_IR) and the two predictors WEATHER_R and TRAF_CON_R.
   - Compute the exact Bayes conditional probabilities of an injury (MAX_SEV_IR=1) given the 4 possible combinations of the predictors.
   - Classify the 10 accidents using these probabilities.
   - Compute the Naive Bayes conditional probability of an injury given WEATHER_R=1 and TRAF_CON_R=1. Compare this with the exact Bayes calculation.
   - Run a Naive Bayes classifier on the 10 records and two predictors using XLMiner. Check "detailed report" to obtain probabilities and classifications for all 10 records. Compare this to the exact Bayes classification.
3. Run a Naive Bayes classifier on the data with all predictors. Make sure that all predictors are categorical, and bin any continuous predictor into reasonable bins. How well does the classifier perform on the validation set? Show the classification matrix. How much better does it perform relative to the naive rule?
4. Examine the conditional probabilities output. List the predictor values which do not exist for a class in the training set.
5. Show how the probability of "no injury" is computed for the first record in the training set. Use the prior and conditional probabilities in the output.

Chapter 7

Classification and Regression Trees

7.1 Introduction

If one had to choose a classification technique that performs well across a wide range of situations without requiring much effort from the analyst while being readily understandable by the consumer of the analysis, a strong contender would be the tree methodology developed by Breiman, Friedman, Olshen and Stone (1984). We will discuss this classification procedure first, then in later sections we will show how the procedure can be extended to prediction of a continuous dependent variable. The program that Breiman et. al. created to implement these procedures was called CART (Classification And Regression Trees). A related procedure is called C4.5 .

What is a classification tree? Figure 7.1 describes a tree for classifying bank customers who receive a loan offer as either acceptors or non-acceptors, as a function of information such as their income, education level, and average credit card expenditure. One of the reasons tree classifiers are very popular is that they provide easily understandable classification rules (at least if the trees are not too large). Consider the tree in the example. The square "terminal nodes" are marked with 0 or 1 corresponding to a non-acceptor (0) or acceptor (1). The values in the circle nodes give the splitting value on a predictor. This tree can easily be translated into a set of rules for classifying a bank customer. For example, the middle left square node in this tree gives us the rule:

IF(Income > 92.5) AND (Education < 1.5) AND (Family ≤ 2.5) THEN Class $= 0$ (non-acceptor).

In this following we show how trees are constructed and evaluated.

7.2 Classification Trees

There are two key ideas underlying classification trees. The first is the idea of recursive partitioning of the space of the independent variables. The second is of pruning using validation data. In the next few sections we describe recursive partitioning, and subsequent sections explain the pruning methodology.

7.3 Recursive Partitioning

Let us denote the dependent (response) variable by y and the independent (predictor) variables by $x_1, x_2, x_3, \cdots, x_p$. In classification, the outcome variable will be a categorical variable. Recursive partitioning divides up the p dimensional space of the x variables into non-overlapping multi-dimensional rectangles. The X variables here are considered to be continuous, binary or ordinal. This division is accomplished recursively (i.e., operating on the results of prior divisions). First, one of the variables

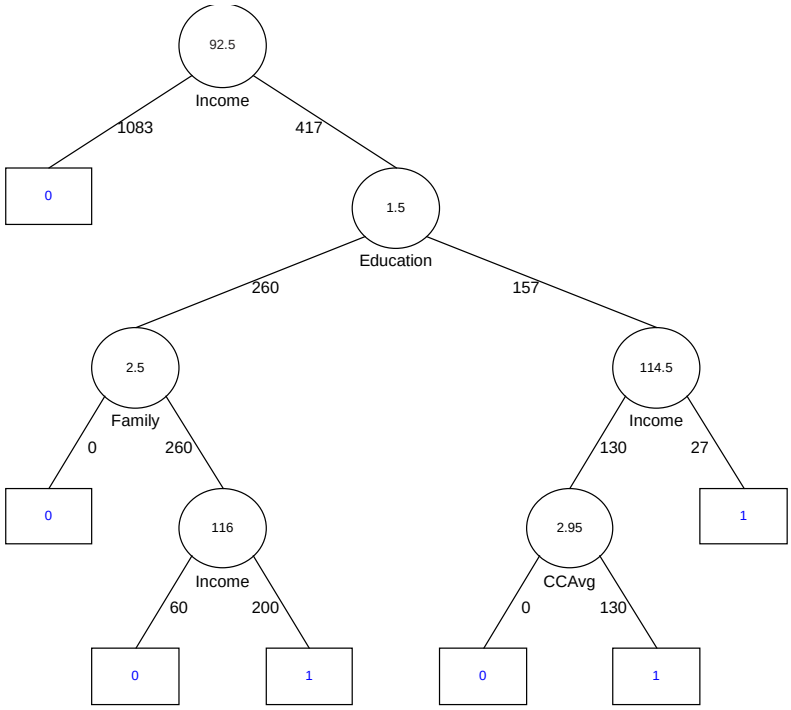


Figure 7.1: The Best Pruned Tree Obtained By Fitting a Full Tree to the Training Data and Pruning it Using the Validation Data

is selected, say x_i, and a value of x_i, say s_i, is chosen to split the p dimensional space into two parts: one part that contains all the points with $x_i \leq s_i$ and the other with all the points with $x_i > s_i$. Then one of these two parts is divided in a similar manner by choosing a variable again (it could be x_i or another variable) and a split value for the variable. This results in three (multi-dimensional) rectangular regions. This process is continued so that we get smaller and smaller rectangular regions. The idea is to divide the entire x-space up into rectangles such that each rectangle is as homogeneous or 'pure' as possible. By 'pure' we mean containing points that belong to just one class. (Of course, this is not always possible, as there may be points that belong to different classes but have exactly the same values for every one of the independent variables.) Let us illustrate recursive partitioning with an example.

7.4 Example 1: Riding Mowers

We use again the riding-mower example presented in Chapter **??**. A riding-mower manufacturer would like to find a way of classifying families in a city into those likely to purchase a riding mower and those not likely to buy one. A pilot random sample of 12 owners and 12 non-owners in the city is undertaken. The data are shown and plotted in Table 7.1 and Figure 7.2.

If we apply the classification tree procedure to these data, the procedure will choose $LotSize$ for the first split with a splitting value of 19. The (x_1, x_2) space is now divided into two rectangles, one with $LotSize \leq 19$ and the other with $LotSize > 19$. This is illustrated in Figure 7.3.

Notice how the split into two rectangles has created two rectangles each of which is much more homogeneous than the rectangle before the split. The upper rectangle contains points that are mostly owners (9 owners and 3 non-owners) while the lower rectangle contains mostly non-owners (9 non-owners and 3 owners).

How was this particular split selected? The algorithm examined each variable (in this case,

Table 7.1: Lot Size, Income, and Ownership of a Riding Mower for 24 Households

Household number	Income ($ 000's)	Lot Size (000's ft^2)	Ownership of, riding mower
1	60	18.4	Owner
2	85.5	16.8	Owner
3	64.8	21.6	Owner
4	61.5	20.8	Owner
5	87	23.6	Owner
6	110.1	19.2	Owner
7	108	17.6	Owner
8	82.8	22.4	Owner
9	69	20	Owner
10	93	20.8	Owner
11	51	22	Owner
12	81	20	Owner
13	75	19.6	Non-Owner
14	52.8	20.8	Non-Owner
15	64.8	17.2	Non-Owner
16	43.2	20.4	Non-Owner
17	84	17.6	Non-Owner
18	49.2	17.6	Non-Owner
19	59.4	16	Non-Owner
20	66	18.4	Non-Owner
21	47.4	16.4	Non-Owner
22	33	18.8	Non-Owner
23	51	14	Non-Owner
24	63	14.8	Non-Owner

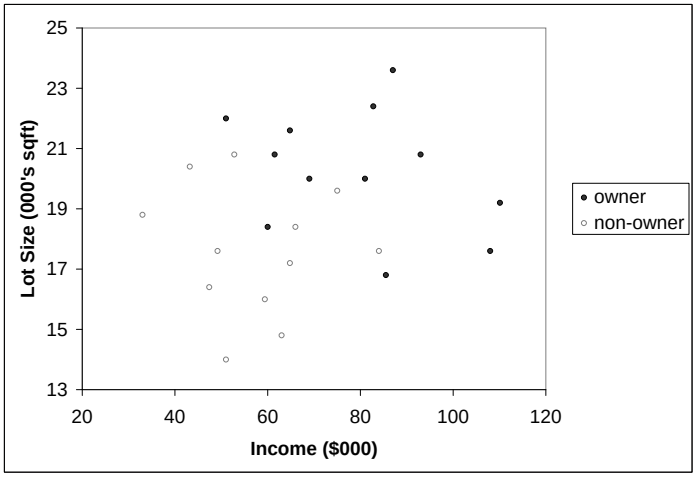


Figure 7.2: Scatterplot of Lot Size Vs. Income for 24 Owners and Non-Owners of Riding Mowers

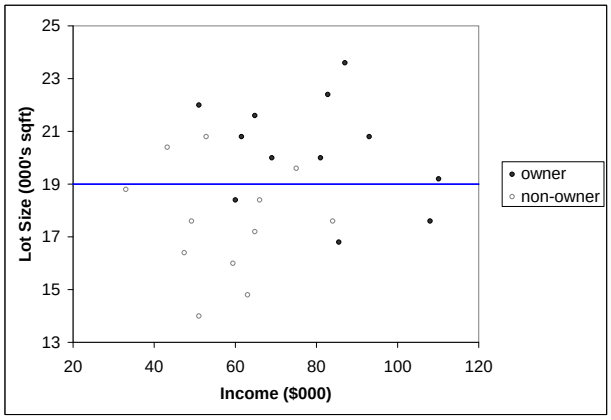


Figure 7.3: Splitting the 24 Observations By Lot Size Value of 19

Income and *Lot Size*) and all possible split values for each variable to find the best split. What are the possible split values for a variable? They are simply the mid-points between pairs of consecutive values for the variable. The possible split points for *Income* are $\{38.1, 45.3, 50.1, \cdots, 109.5\}$ and those for *Lot Size* are $\{14.4, 15.4, 16.2, \cdots, 23\}$. These split points are ranked according to how much they reduce impurity (heterogeneity) in the resulting rectangle. A pure rectangle is one that is composed of a single class (e.g., owners). The reduction in impurity is defined as overall impurity before the split minus the sum of the impurities for the two rectangles that result from a split.

7.4.1 Measures of Impurity

There are a number of ways to measure impurity. The two most popular measures are the *Gini index* and an *entropy measure*. We describe both next. Denote the m classes of the response variable by $k = 1, 2, \cdots, m$.

The Gini impurity index for a rectangle A is defined by

$$I(A) = 1 - \sum_{k=1}^{m} p_k^2$$

where p_k is the proportion of observations in rectangle A that belong to class k. This measure takes values between 0 (if all the observations belong to the same class) and $(m-1)/m$ (when all m classes are equally represented). Figure 7.4 shows the values of the Gini Index for a two-class case, as a function of p_k. It can be seen that the impurity measure is at its peak when $p_k = 0.5$, i.e. when the rectangle contains 50% of each of the two classes.[1]

A second impurity measure is the entropy measure. The entropy for a rectangle A is defined by

$$Entropy(A) = -\sum_{k=1}^{m} p_k \log_2(p_k)$$

(to compute $\log_2(x)$ in Excel, use the function $= log(x, 2)$).
This measure ranges between 0 (most pure, all observations belong to the same class) and $\log_2(m)$ (when all m classes are equally represented). In the two-class case, the entropy measure is maximized (like the Gini index) at $p_k = 0.5$.

[1] XLMiner uses a variant of the Gini Index called the delta splitting rule; for details, see XLMiner documentation.

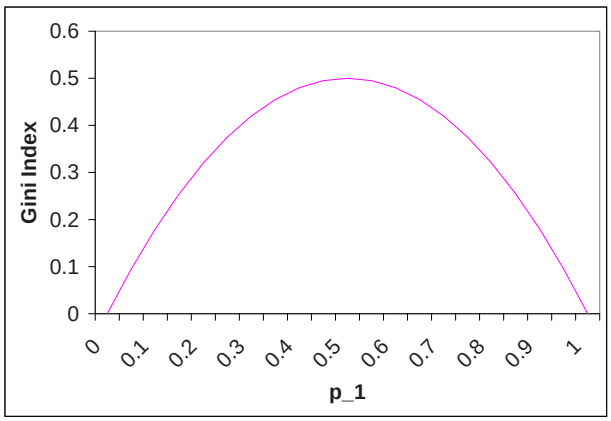


Figure 7.4: Values of the Gini Index for a Two-Class Case, As a Function of the Proportion of Observations in Class 1 (p_1)

Let us compute the impurity in the riding mower example, before and after the first split (using *Lot Size* with the value of 19). The unsplit dataset contains 12 owners and 12 non-owners. This is a two-class case with an equal number of observations from each class. Both impurity measures are therefore at their maximum value: Gini=0.5 and entropy=$\log_2(2) = 1$. After the split, the upper rectangle contains 9 owners and 3 non-owners. The impurity measures for this rectangle are Gini= $1-0.25^2-0.75^2 = .375$ and entropy= $-0.25\log_2(0.25)-0.75\log_2(0.75) = 0.811$. The lower rectangle contains 3 owners and 9 non-owners. Since both impurity measures are symmetric, they obtain the same values as for the upper rectangle.

The combined impurity of the two rectangles that were created by the split is a weighted average of the two impurity measures, weighted by the number of observations in each (in this case we ended up with 12 observations in each rectangle, but in general the number of observations need not be equal): $Gini = (12/24)(.375)+(12/24)(0.375) = .375$ and $entropy = (12/24)(.811)+(12/24)(.811) = .811$. Thus the Gini impurity index reduced from 0.5 before the split to .375 after the split. Similarly, the entropy impurity measure reduced from 1 before the split to .811 after the split.

By comparing the reduction in impurity across all possible splits in all possible predictors, the next split is chosen. If we continue splitting the mower data, the next split is on the Income variable at the value 84.75. Figure 7.5 shows that once again the tree procedure has astutely chosen to split a rectangle to increase the purity of the resulting rectangles. The left lower rectangle, which contains data points with $Income \leq 84.75$ and $Lot\ Size \leq 19$, has all points that are non-owners (with one exception); while the right lower rectangle, which contains data points with $Income > 84.75$ and $Lot\ Size \leq 19$, consists exclusively of owners. We can see how the recursive partitioning is refining the set of constituent rectangles to become purer as the algorithm proceeds. The final stage of the recursive partitioning is shown in Figure 7.6.

Notice that now each rectangle is pure: it contains data points from just one of the two classes.

The reason the method is called a classification tree algorithm is that each split can be depicted as a split of a node into two successor nodes. The first split is shown as a branching of the root node of a tree in Figure 7.7. The tree representing the first three splits is shown in Figure 7.5.

The full grown tree is shown in Figure 7.9.

We represent the nodes that have successors by circles. The numbers inside the circle are the splitting values and the name of the variable chosen for splitting at that node is shown below the node. The numbers on the left fork at a decision node shows the number of points in the decision node that had values less than or equal to the splitting value while the number on the right fork

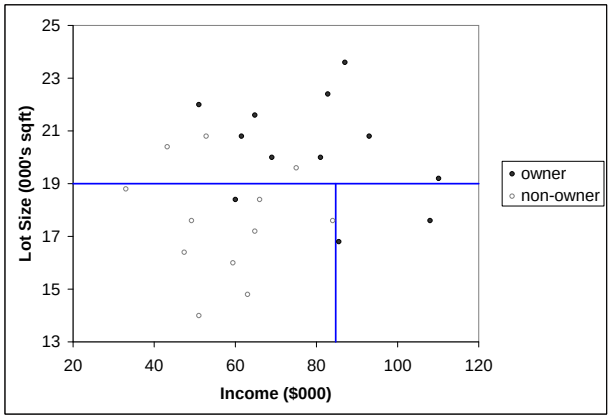


Figure 7.5: Splitting the 24 Observations By Lot Size Value of 19K, and then Income Value of 84.75K

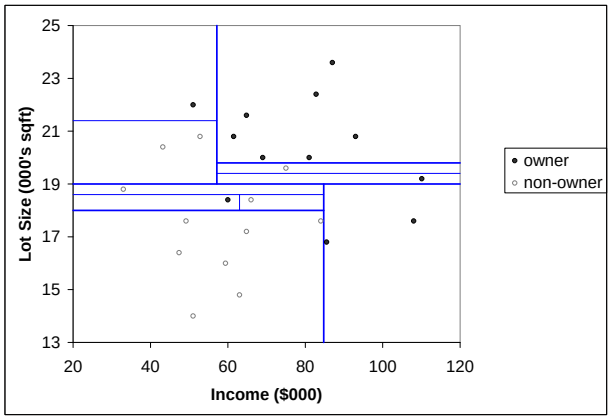


Figure 7.6: Final Stage of Recursive Partitioning: Each Rectangle Consists of a Single Class (Owners or Non-Owners)

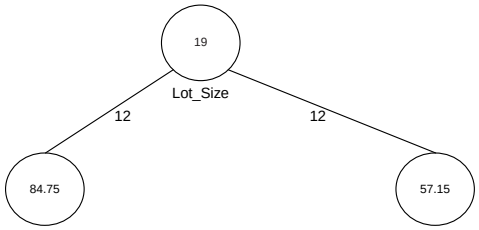


Figure 7.7: Tree Representation of First Split (Corresponds to Figure 7.3)

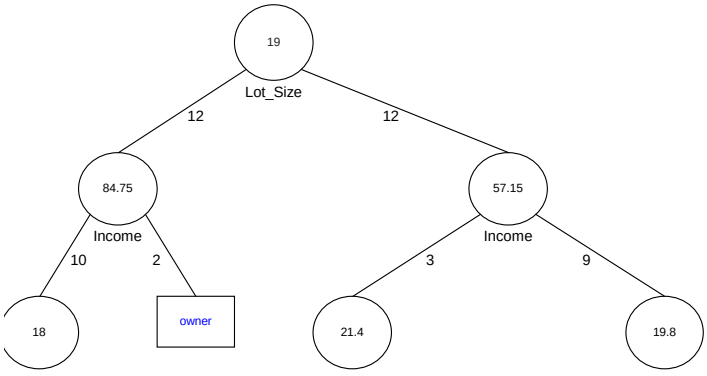


Figure 7.8: Tree Representation of First Three Splits (Corresponds to Figure 7.5)

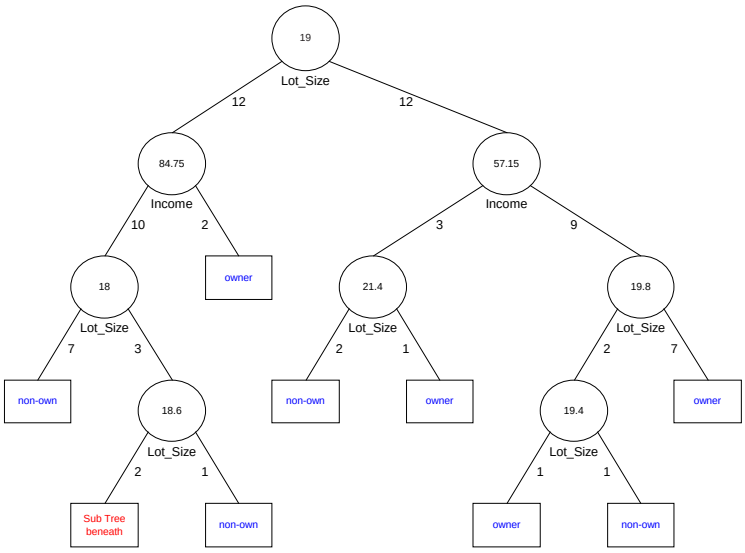


Figure 7.9: Tree Representation of First Three Splits (Corresponds to Figure 7.6)

shows the number that had a greater value. These are called decision nodes because if we were to use a tree to classify a new observation for which we knew only the values of the independent variables, we would "drop" the observation down the tree in such a way that at each decision node the appropriate branch is taken until we get to a node that has no successors. Such terminal nodes are called the leaves of the tree. Each leaf node is depicted with a rectangle, rather than a circle, and corresponds to one of the final rectangles into which the x-space is partitioned. When the observation has dropped all the way down to a leaf, we can predict a class for it by simply taking a "vote" of all the training data that belonged to the leaf when the tree was grown. The class with the highest vote is the class that we would predict for the new observation. The name of this class appears in the leaf nodes. For instance, the rightmost leaf node in Figure 7.9 has a majority of observations that belong to the owner group. It is therefore labeled "owner". It is useful to note that the type of trees grown by CART (called binary trees) have the property that the number of leaf nodes is exactly one more than the number of decision nodes.

To handle categorical predictors, the split choices for a categorical predictor are all ways in which the set of categorical values can be divided into two subsets. For example a categorical variable with 4 categories, say {1,2,3,4} can be split in 7 ways into two subsets: {1} and {2,3,4}; {2} and {1,3,4}; {3} and {1,2,4}; {4} and {1,2,3}; {1,2} and {3,4}; {1,3} and {2,4}; {1,4} and {2,3}. When the number of categories is large the number of splits becomes very large. XLMiner supports only binary categorical variables (coded as numbers). If you have a categorical predictor that takes more than two values, you will need to replace the variable with several dummy variables each of which is binary in a manner that is identical to the use of dummy variables in regression.[2]

[2]This is a difference between CART and C4.5: the former performs only binary splits, leading to binary trees, whereas the latter performs splits that are as large as the number of categories, leading to "bush-like" structures.

7.5 Evaluating the Performance of a Classification Tree

In order to assess the accuracy of the tree in classifying new cases, we use the same tools and criteria that were discussed in Chapter 4. We start by partitioning the data into training and validation sets. The training set is used to grow the tree, and the validation set is used to assess its performance. In the next section we will discuss an important step in constructing trees that involves using the validation data. In that case, a third set of test data is preferable for assessing the accuracy of the final tree.

Each observation in the validation (or test) data is "dropped down" the tree and classified according to the leaf node it reaches. These predicted classes can then be compared to the actual memberships via a confusion matrix. When there is a particular class that is of interest, then a lift chart is useful for assessing the model's ability to capture those members. We use the following example to illustrate this.

7.5.1 Example 2: Acceptance of Personal Loan

Universal Bank is a relatively young bank which is growing rapidly in terms of overall customer acquisition. The majority of these customers are liability customers with varying sizes of relationship with the bank. The customer base of asset customers is quite small, and the bank is interested in growing this base rapidly to bring in more loan business. In particular, it wants to explore ways of converting its liability customers to Personal Loan customers.

A campaign the bank ran for liability customers showed a healthy conversion rate of over 9% successes. This has encouraged the Retail Marketing department to devise smarter campaigns with better target marketing. The goal of our analysis is to model the previous campaign's customer behavior to analyze what combination of factors make a customer more likely to accept a personal loan. This will serve as the basis for the design of a new campaign.

The bank's dataset includes data on 5000 customers. The data include customer demographic information (Age, Income, etc.), customer response to the last personal loan campaign (Personal Loan), and the customer's relationship with the bank (mortgage, securities account, etc.). Among these 5000 customers only 480 (= 9.6%) accepted the personal loan that was offered to them in the previous campaign.

Table 7.2 contains a sample of the bank's customer database for 20 customers, to illustrate the structure of the data.

ID	Age	Professional Experience	Income	Family Size	CC Avg	Education	Mortgage	**Personal Loan**	Securities Account	CD Account	Online Banking	Credit Card
1	25	1	49	4	1.60	UG	0	No	Yes	No	No	No
2	45	19	34	3	1.50	UG	0	No	Yes	No	No	No
3	39	15	11	1	1.00	UG	0	No	No	No	No	No
4	35	9	100	1	2.70	Grad	0	No	No	No	No	No
5	35	8	45	4	1.00	Grad	0	No	No	No	No	Yes
6	37	13	29	4	0.40	Grad	155	No	No	No	Yes	No
7	53	27	72	2	1.50	Grad	0	No	No	No	Yes	No
8	50	24	22	1	0.30	Prof	0	No	No	No	No	Yes
9	35	10	81	3	0.60	Grad	104	No	No	No	Yes	No
10	34	9	180	1	8.90	Prof	0	Yes	No	No	No	No
11	65	39	105	4	2.40	Prof	0	No	No	No	No	No
12	29	5	45	3	0.10	Grad	0	No	No	No	Yes	No
13	48	23	114	2	3.80	Prof	0	No	Yes	No	No	No
14	59	32	40	4	2.50	Grad	0	No	No	No	Yes	No
15	67	41	112	1	2.00	UG	0	No	Yes	No	No	No
16	60	30	22	1	1.50	Prof	0	No	No	No	Yes	Yes
17	38	14	130	4	4.70	Prof	134	Yes	No	No	No	No
18	42	18	81	4	2.40	UG	0	No	No	No	No	No
19	46	21	193	2	8.10	Prof	0	Yes	No	No	No	No
20	55	28	21	1	0.50	Grad	0	No	Yes	No	No	Yes

Table 7.2: Sample of Data for 20 Customers of Universal Bank

After randomly partitioning the data into training (2500 observations), validation (1500 observations), and test (1000 observations) sets, we use the training data to construct a full-grown tree.

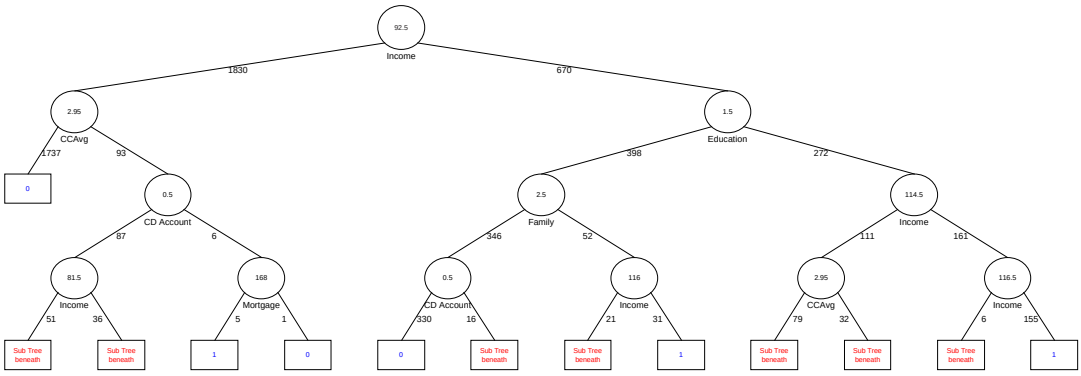


Figure 7.10: First Seven Levels of the Full-Grown Tree for the Loan Acceptance Data, Using the Training Set (2500 Observations)

The first 7 levels of the tree are shown in figure 7.10, and the complete results are given in a form of a table in figure 7.11

Even with just 7 levels it is hard to see the complete picture. A look at the first row of the table reveals that the first predictor that is chosen to split the data is *Income* with the value of 92.5 ($000).

Since the full grown tree leads to completely pure terminal leaves, it is 100% accurate in classifying the training data. This can be seen in Figure 7.12. In contrast, the confusion matrix for the validation and test data (which were not used to construct the full grown tree) show lower classification accuracy of the tree. The main reason is that the full-grown tree overfits the training data (to complete accuracy!) This motivates the next section which describes ways to avoid overfitting by either stopping the growth of the tree before it is fully grown, or by pruning the full-grown tree.

7.6 Avoiding Overfitting

As the last example illustrated, using a full-grown tree (based on the training data) leads to complete overfitting of the data. As discussed in Chapter 4, overfitting will lead to poor performance on new data. If we look at the overall error at the different levels of the tree, it is expected to decrease as the number of levels grows until the point of overfitting. Of course, for the training data the overall error decreases more and more until it is zero at the maximum level of the tree. However, for new data, the overall error is expected to decrease until the point where the tree models the relationship between class and the predictors. After that, the tree starts to model the noise in the training set, and we expect the overall error for the validation set to start increasing. This is depicted in Figure 7.13. One intuitive reason for the overfitting at the high levels of the tree is that these splits are based on very small numbers of observations. In such cases class difference is likely to be attributed to noise rather than predictor information.

Two ways to try and avoid exceeding this level, thereby limiting overfitting, are by setting rules to stop the tree growth, or alternatively, by pruning the full-grown tree back to a level where it does not overfit. These solutions are discussed below.

Full Tree Rules (Using Training Data)

#Decision Nodes	41

#Terminal Nodes	42

Level	NodeID	ParentID	SplitVar	SplitValue	Cases	LeftChild	RightChild	Class	Node Type
0	0	N/A	Income	92.5	2500	1	2	0	Decision
1	1	0	CCAvg	2.95	1830	3	4	0	Decision
1	2	0	Education	1.5	670	5	6	0	Decision
2	3	1	N/A	N/A	1737	N/A	N/A	0	Terminal
2	4	1	CD Account	0.5	93	7	8	0	Decision
2	5	2	Family	2.5	398	9	10	0	Decision
2	6	2	Income	114.5	272	11	12	1	Decision
3	7	4	Income	81.5	87	13	14	0	Decision
3	8	4	Mortgage	168	6	15	16	1	Decision
3	9	5	CD Account	0.5	346	17	18	0	Decision
3	10	5	Income	116	52	19	20	1	Decision
3	11	6	CCAvg	2.95	111	21	22	0	Decision
3	12	6	Income	116.5	161	23	24	1	Decision
4	13	7	Age	28	51	25	26	0	Decision
4	14	7	CCAvg	3.75	36	27	28	0	Decision
4	15	8	N/A	N/A	5	N/A	N/A	1	Terminal
4	16	8	N/A	N/A	1	N/A	N/A	0	Terminal
4	17	9	N/A	N/A	330	N/A	N/A	0	Terminal
4	18	9	Mortgage	350.5	16	29	30	0	Decision
4	19	10	CCAvg	1.7	21	31	32	0	Decision
4	20	10	N/A	N/A	31	N/A	N/A	1	Terminal
4	21	11	Income	106.5	79	33	34	0	Decision
4	22	11	EducProf	0.5	32	35	36	1	Decision
4	23	12	CCAvg	1.1	6	37	38	1	Decision
4	24	12	N/A	N/A	155	N/A	N/A	1	Terminal
5	25	13	N/A	N/A	1	N/A	N/A	1	Terminal
5	26	13	N/A	N/A	50	N/A	N/A	0	Terminal
5	27	14	CCAvg	3.35	16	39	40	0	Decision
5	28	14	Mortgage	93.5	20	41	42	0	Decision
5	29	18	N/A	N/A	15	N/A	N/A	0	Terminal
5	30	18	N/A	N/A	1	N/A	N/A	1	Terminal
5	31	19	N/A	N/A	13	N/A	N/A	0	Terminal
5	32	19	Income	109.5	8	43	44	0	Decision
5	33	21	N/A	N/A	49	N/A	N/A	0	Terminal
5	34	21	CCAvg	1.75	30	45	46	0	Decision
5	35	22	Age	60	17	47	48	1	Decision
5	36	22	CCAvg	3.7	15	49	50	0	Decision
5	37	23	Online	0.5	2	51	52	1	Decision
5	38	23	N/A	N/A	4	N/A	N/A	1	Terminal
6	39	27	N/A	N/A	7	N/A	N/A	0	Terminal
6	40	27	CCAvg	3.65	9	53	54	1	Decision
6	41	28	N/A	N/A	16	N/A	N/A	0	Terminal
6	42	28	Mortgage	104.5	4	55	56	0	Decision
6	43	32	Experience	6.5	4	57	58	1	Decision
6	44	32	N/A	N/A	4	N/A	N/A	0	Terminal
6	45	34	Family	1.5	13	59	60	0	Decision
6	46	34	ZIP Code	94206.5	17	61	62	0	Decision
6	47	35	N/A	N/A	13	N/A	N/A	1	Terminal
6	48	35	Age	64	4	63	64	1	Decision
6	49	36	N/A	N/A	3	N/A	N/A	1	Terminal
6	50	36	Family	2.5	12	65	66	0	Decision
6	51	37	N/A	N/A	1	N/A	N/A	1	Terminal
6	52	37	N/A	N/A	1	N/A	N/A	0	Terminal
7	53	40	Age	61.5	5	67	68	1	Decision
7	54	40	EducProf	0.5	4	69	70	0	Decision
7	55	42	N/A	N/A	1	N/A	N/A	1	Terminal
7	56	42	N/A	N/A	3	N/A	N/A	0	Terminal
7	57	43	N/A	N/A	1	N/A	N/A	0	Terminal
7	58	43	N/A	N/A	3	N/A	N/A	1	Terminal
7	59	45	EducProf	0.5	7	71	72	0	Decision
7	60	45	ZIP Code	91409	6	73	74	1	Decision
7	61	46	Income	111	7	75	76	0	Decision
7	62	46	N/A	N/A	10	N/A	N/A	0	Terminal
7	63	48	N/A	N/A	2	N/A	N/A	0	Terminal
7	64	48	N/A	N/A	2	N/A	N/A	1	Terminal
7	65	50	N/A	N/A	9	N/A	N/A	0	Terminal
7	66	50	N/A	N/A	3	N/A	N/A	1	Terminal
8	67	53	N/A	N/A	4	N/A	N/A	1	Terminal
8	68	53	N/A	N/A	1	N/A	N/A	0	Terminal
8	69	54	N/A	N/A	1	N/A	N/A	1	Terminal
8	70	54	N/A	N/A	3	N/A	N/A	0	Terminal
8	71	59	Online	0.5	2	77	78	1	Decision
8	72	59	N/A	N/A	5	N/A	N/A	0	Terminal
8	73	60	Family	2.5	3	79	80	0	Decision
8	74	60	N/A	N/A	3	N/A	N/A	1	Terminal
8	75	61	Mortgage	54.5	3	81	82	1	Decision
8	76	61	N/A	N/A	4	N/A	N/A	0	Terminal
9	77	71	N/A	N/A	1	N/A	N/A	0	Terminal
9	78	71	N/A	N/A	1	N/A	N/A	1	Terminal
9	79	73	N/A	N/A	1	N/A	N/A	1	Terminal
9	80	73	N/A	N/A	2	N/A	N/A	0	Terminal
9	81	75	N/A	N/A	2	N/A	N/A	1	Terminal
9	82	75	N/A	N/A	1	N/A	N/A	0	Terminal

Figure 7.11: Description of Each Splitting Step of the Full-Grown Tree for the Loan Acceptance Data

Training Data scoring - Summary Report (Using Full Tree)

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

Classification Confusion Matrix		
	Predicted Class	
Actual Class	1	0
1	235	0
0	0	2265

Error Report			
Class	# Cases	# Errors	% Error
1	235	0	0.00
0	2265	0	0.00
Overall	2500	0	0.00

Validation Data scoring - Summary Report (Using Full Tree)

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

Classification Confusion Matrix		
	Predicted Class	
Actual Class	1	0
1	128	15
0	17	1340

Error Report			
Class	# Cases	# Errors	% Error
1	143	15	10.49
0	1357	17	1.25
Overall	1500	32	2.13

Test Data scoring - Summary Report (Using Full Tree)

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

Classification Confusion Matrix		
	Predicted Class	
Actual Class	1	0
1	88	14
0	8	890

Error Report			
Class	# Cases	# Errors	% Error
1	102	14	13.73
0	898	8	0.89
Overall	1000	22	2.20

Figure 7.12: Confusion Matrix and Error Rates for the Training Data and the Validation Data

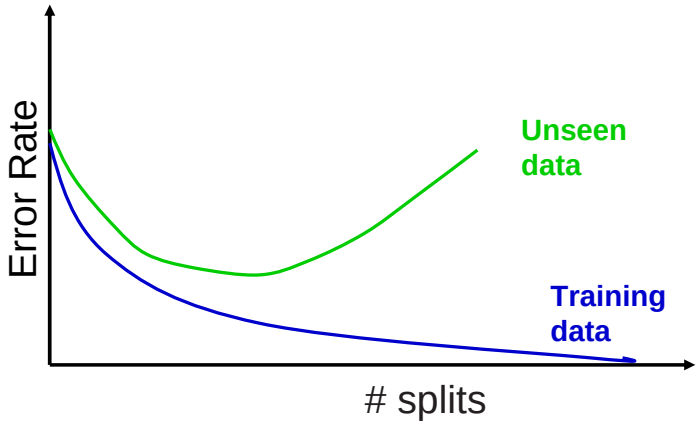


Figure 7.13: Error Rate as a Function of the Number of Splits for Training Vs. Validation Data: Overfitting

7.6.1 Stopping Tree Growth: CHAID

One can think of different criteria for stopping the tree growth before it starts overfitting the data. Examples are tree depth (i.e. number of splits), minimum number of records in a node, and minimum reduction in impurity. The problem is that it is not simple to determine what is a good stopping point using such rules.

Previous methods developed were based on the idea of recursive partitioning, using rules to prevent the tree from growing excessively and overfitting the training data. One popular method called CHAID (Chi-Squared Automatic Interaction Detection) is a recursive partitioning method that predates classification and regression tree (CART) procedures by several years and is widely used in database marketing applications to this day. It uses a well-known statistical test (the chi-square test for independence) to assess whether splitting a node improves the purity by a statistically significant amount. In particular, at each node we split on the predictor that has the strongest association with the response variable. The strength of association is measured by the p-value of a chi-squared test of independence. If for the best predictor the test does not show a significant improvement the split is not carried out, and the tree is terminated. This method is more suitable for categorical predictors, but it can be adapted to continuous predictors by binning the continuous values into categorical bins.

7.6.2 Pruning the Tree

An alternative solution which has proven to be more successful than stopping tree growth is pruning the full-grown tree. This is the basis of methods such as CART (developed by Breiman et al., implemented in multiple data mining software packages such as SAS Enterprise Miner, CART, MARS, and in XLMiner) and C4.5 (developed by Quinlan, and implemented in packages such as Clementine by SPSS). In C4.5 the training data are used both for growing the tree and for pruning it. In CART the innovation is to use the validation data to prune back the tree that is grown from training data. CART and CART-like procedures use validation data to prune back the tree that has

been deliberately overgrown using the training data. This approach is also used by XLMiner.

The idea behind pruning is to recognize that a very large tree is likely to be overfitting the training data, and that the weakest branches, which hardly reduce the error rate, should be removed. In the mower example the last few splits resulted in rectangles with very few points (indeed four rectangles in the full tree had just one point). We can see intuitively that these last splits are likely to be simply capturing noise in the training set rather than reflecting patterns that would occur in future data such as the validation data. Pruning consists of successively selecting a decision node and re-designating it as a leaf node (lopping off the branches extending beyond that decision node (its "subtree") and thereby reducing the size of the tree). The pruning process trades off misclassification error in the validation dataset against the number of decision nodes in the pruned tree to arrive at a tree that captures the patterns but not the noise in the training data. Returning to Figure 7.13, we would like to find the point where the curve for the unseen data begins to increase.

To find this point, the CART algorithm uses a criterion called the "cost complexity" of a tree to generate a sequence of trees that are successively smaller to the point of having a tree with just the root node. (What is the classification rule for a tree with just one node?). This means that the first step is to find the best subtree of each size $(1, 2, 3, \ldots)$. Then, to chose among these, we want the tree that minimizes the error rate of the validation set. We then pick as our best tree the one tree in the sequence that gives the smallest misclassification error in the validation data.

Constructing the "best tree" of each size is based on the cost complexity (CC) criterion, which is equal to the misclassification error of a tree (based on the training data) plus a penalty factor for the size of the tree. For a tree T that has $L(T)$ leaf nodes, the cost complexity can be written as

$$CC(T) = Err(T) + \alpha L(T)$$

where Err(T) is the fraction of training data observations that are misclassified by tree T and α is a "penalty factor" for tree size. When $\alpha = 0$ there is no penalty for having too many nodes in a tree and the best tree using the cost complexity criterion is the full-grown unpruned tree. When we increase α to a very large value the penalty cost component swamps the misclassification error component of the cost complexity criterion function and the best tree is simply the tree with the fewest leaves, namely the tree with simply one node. The idea is therefore to start with the full-grown tree and then increase the penalty factor α gradually until the cost complexity of the full tree exceeds that of a subtree. Then, the same procedure is repeated using the subtree. Continuing in this manner we generate a succession of trees with diminishing number of nodes all the way to the trivial tree consisting of just one node.

From this sequence of trees it seems natural to pick the one that gave the minimum misclassification error on the validation dataset. We call this the Minimum Error Tree.

To illustrate this, Figure 7.14 shows the error rate for both the training and validation data as a function of the tree size. It can be seen that the training set error steadily decreases as the tree grows, with a noticeable drop in error rate between 2 and 3 nodes. The validation set error rate, however, reaches a minimum at 11 nodes, and then starts to increase as the tree grows. At this point the tree is pruned and we obtain the "Minimum Error Tree".

A further enhancement is to incorporate the sampling error which might cause this minimum to vary if we had a different sample. The enhancement uses the estimated standard error of the error to prune the tree even further (to the validation error rate that is one standard error above the minimum.) In other words, the "Best Pruned Tree" is the smallest tree in the pruning sequence that has an error within one standard error of the Minimum Error Tree. The best pruned tree for the loan acceptance example is shown in Figure 7.15.

Returning to the loan acceptance example, we expect that the classification accuracy of the validation set using the pruned tree would be higher than using the full-grown tree (compare Figure 7.12 with 7.16). However, the performance of the pruned tree on the validation data is not fully reflective of the performance on completely new data, because the validation data were actually used for the pruning. This is a situation where it is particularly useful to evaluate the performance

# Decision Nodes	% Error Training	% Error Validation			
41	0	2.133333			
40	0.04	2.2			
39	0.08	2.2			
38	0.12	2.2			
37	0.16	2.066667			
36	0.2	2.066667			
35	0.2	2.066667			
34	0.24	2.066667			
33	0.28	2.066667			
32	0.4	2.066667			
31	0.48	2.133333			
30	0.48	2.133333			
29	0.56	2.133333			
28	0.6	1.866667			
27	0.64	1.866667			
26	0.72	1.866667			
25	0.76	1.866667			
24	0.88	1.866667			
23	0.88	1.733333			
22	0.88	1.733333			
21	0.96	1.733333			
20	0.96	1.733333			
19	1	1.733333			
18	1	1.733333			
17	1.12	1.733333			
16	1.12	1.533333			
15	1.12	1.533333			
14	1.16	1.533333			
13	1.16	1.6			
12	1.2	1.6			
11	1.2	1.466667	<-- Min. Err. Tree	Std. Err.	0.003103929
10	1.6	1.666667			
9	2.2	1.666667			
8	2.2	1.866667			
7	2.24	1.866667			
6	2.24	1.6	<-- Best Pruned Tree		
5	4.44	1.8			
4	5.08	2.333333			
3	5.24	3.466667			
2	9.4	9.533333			
1	9.4	9.533333			
0	9.4	9.533333			

Figure 7.14: Error Rate as a Function of the Number of Splits for Training Vs. Validation Data for Loan Example

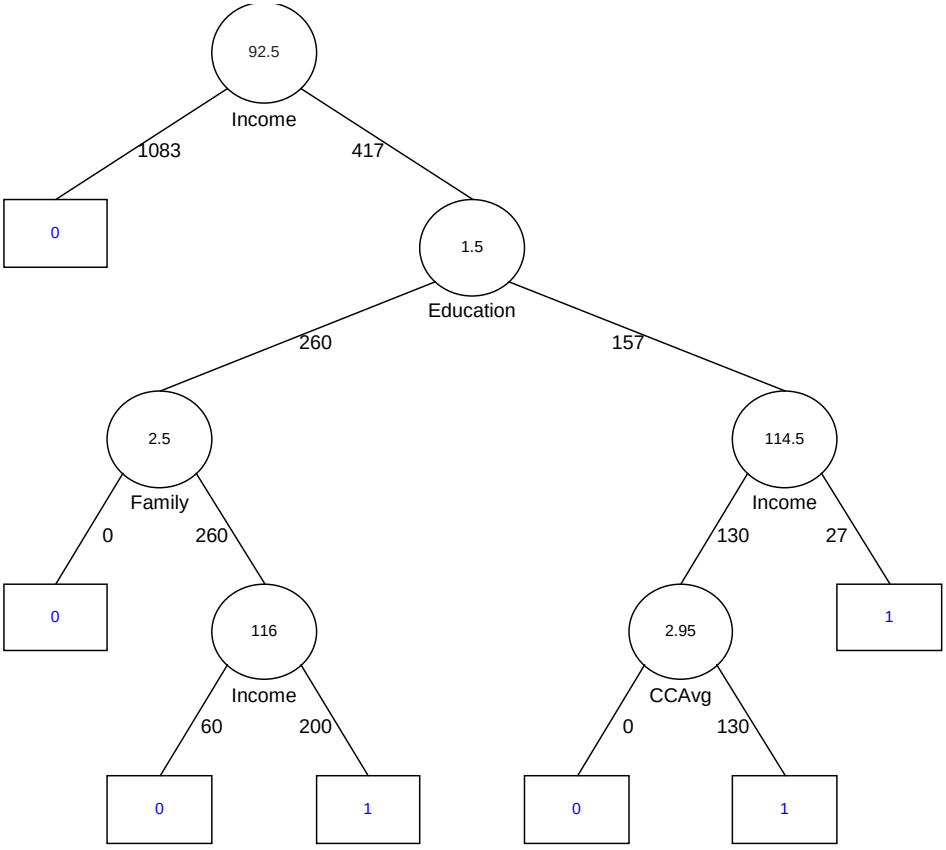


Figure 7.15: The Best Pruned Tree Obtained by Fitting a Full Tree to the Training Data and Pruning it Using the Validation Data

of the chosen model, whatever it may be, on a third set of data, the test set, which has not been used at all. In our example, the pruned tree applied to the test data yields an overall error rate of 1.7% (compared to 0% for the training data and 1.6% for the validation data). Although in this example the performance on the validation and test sets is similar, the difference can be larger for other datasets.

7.7 Classification Rules from Trees

As described in the introduction, classification trees provide easily understandable *classification rules* (if the trees are not too large). Each leaf is equivalent to a classification rule. Returning to the example, the middle left leaf in the best pruned tree, gives us the rule:

$$\text{IF(Income} > 92.5) \text{ AND (Education} < 1.5) \text{ AND (Family} \leq 2.5) \text{ THEN Class} = 0.$$

However, in many cases the number of rules can be reduced by removing redundancies. For example, the rule

$$\text{IF(Income} > 92.5) \text{ AND (Education} > 1.5) \text{ AND (Income} > 114.5) \text{ THEN Class} = 1.$$

can be simplified to

$$\text{IF(Income} > 114.5) \text{ AND (Education} > 1.5) \text{ THEN Class} = 1.$$

This transparency in the process and understandability of the algorithm that leads to classifying a record as belonging to a certain class is very advantageous in settings where the final classification is not solely of interest. Berry & Linoff (2000) give the example of health insurance underwriting, where the insurer is required to show that coverage denial is not based on discrimination. By showing rules that led to denial, e.g., income $< 20K$ AND low credit history, the company can avoid law suits. Compared to the output of other classifiers such as discriminant functions, tree-based classification rules are easily explained to managers and operating staff. Their logic is certainly far more transparent than that of weights in neural networks!

7.8 Regression Trees

The CART method can also be used for continuous response variables. Regression trees for prediction operate in much the same fashion as classification trees. The output variable, Y, is a continuous variable in this case, but both the principle and the procedure are the same: many splits are attempted and, for each, we measure "impurity" in each branch of the resulting tree. The tree procedure then selects the split that minimizes the sum of such measures. To illustrate a regression tree, consider the example of predicting prices of Toyota Corolla automobiles (from Chapter 5). The dataset includes information on 1000 sold Toyota Corolla cars. The goal is to find a predictive model of price as a function of 10 predictors (including mileage, horsepower, number of doors, etc.). A regression tree for these data was built using a training set of 600. The best pruned tree is shown in Figure 7.17.

It can be seen that only two predictors show up as useful for predicting price: the age of the car and its horsepower. There are three details that are different in regression trees than in classification trees: prediction, impurity measures, and evaluating performance. We describe these next.

7.8.1 Prediction

Predicting the value of the response Y for an observation is performed in a similar fashion to the classification case: the predictor information is used for "dropping" down the tree until reaching a leaf node. For instance, to predict the price of a Toyota Corolla with Age=55 and Horsepower=86, we drop it down the tree and reach the node that has the value \$8842.65. This is the price prediction for this car according to the tree. In classification trees the value of the leaf node (which is one of the categories) is determined by the "voting" of the training data that were in that leaf. In regression trees the value of the leaf node is determines by the average of the training data in that leaf. In the above example, the value \$8842.6 is the average of the 56 cars in the training set that fall in the category of Age > 52.5 AND Horsepower < 93.5.

Training Data scoring - Summary Report (Using Full Tree)

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

Classification Confusion Matrix		
	Predicted Class	
Actual Class	1	0
1	235	0
0	0	2265

Error Report			
Class	# Cases	# Errors	% Error
1	235	0	0.00
0	2265	0	0.00
Overall	2500	0	0.00

Validation Data scoring - Summary Report (Using Best Pruned Tree)

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

Classification Confusion Matrix		
	Predicted Class	
Actual Class	1	0
1	127	16
0	8	1349

Error Report			
Class	# Cases	# Errors	% Error
1	143	16	11.19
0	1357	8	0.59
Overall	1500	24	1.60

Test Data scoring - Summary Report (Using Best Pruned Tree)

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

Classification Confusion Matrix		
	Predicted Class	
Actual Class	1	0
1	88	14
0	3	895

Error Report			
Class	# Cases	# Errors	% Error
1	102	14	13.73
0	898	3	0.33
Overall	1000	17	1.70

Figure 7.16: Confusion Matrix And Error Rates For The Training, Validation, and Test Data Based on the Pruned Tree

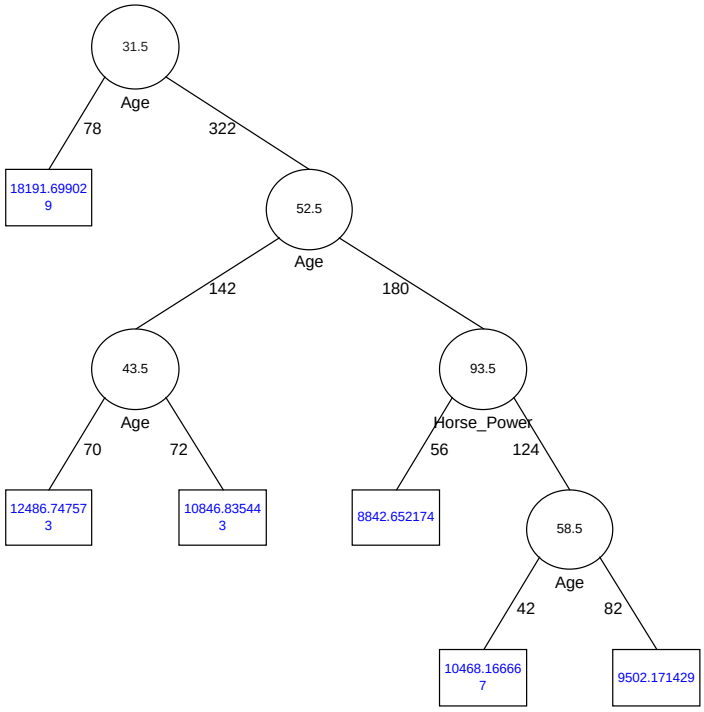


Figure 7.17: Best Pruned Regression Tree for Toyota Corolla Prices

7.8.2 Measuring Impurity

We described two types of impurity measures for nodes in classification trees: the Gini index and the entropy-based measure. In both cases the index is a function of the ratio between the categories of the observations in that node. In regression trees a typical impurity measure is the sum of the squared deviations from the mean of the leaf. This is equivalent to the squared errors, because the mean of the leaf is exactly the prediction. In the example above, the impurity of the node with the value \$8842.6 is computed by subtracting \$8842.6 from the price of each of the 56 cars in the training set that fell in that leaf, then squaring these deviations, and summing them up. The lowest impurity possible is zero, when all values in the node are equal.

7.8.3 Evaluating Performance

As stated above, predictions are obtained by averaging the values of the responses in the nodes. We therefore have the usual definition of predictions and errors. The predictive performance of regression trees can be measured in the same way that other predictive methods are evaluated, using summary measures such as RMSE and charts such as lift charts.

7.9 Advantages, Weaknesses, and Extensions

Tree methods are a good off-the-shelf classifiers and predictors. They are also useful for variable selection, with the most important predictors usually showing up at the top of the tree. Trees require relatively little effort from users in the following senses: First, there is no need for transformation of variables (any monotone transformation of the variables will give the same trees). And second, variable subset selection is automatic since it is part of the split selection. In the loan example notice that the best pruned tree has automatically selected just four variables (*Income, Education, Family, and CCAvg)* out of the set 14 variables available.

Trees are also intrinsically robust to outliers, since the choice of a split depends on the *ordering* of observation values and not on the absolute *magnitudes* of these values. However, the are sensitive to changes in the data, and even a slight change can cause very different splits!

Unlike models that assume a particular relationship between the response and predictors (e.g., a linear relationship such as in linear regression and linear discriminant analysis), classification and regression trees are non-linear and non-parametric. This allows for a wide range of relationships between the predictors and the response. However, this can also be a weakness: since the splits are done on single predictors rather than on combinations of predictors, the tree is likely to miss relationships between predictors, and in particular linear structures like those in linear or logistic regression models. Classification trees are useful classifiers in cases where horizontal and vertical splitting of the predictor space adequately divides the classes. But consider, for instance, a dataset with two predictors and two classes, where separation between the two classes is most obviously achieved by using a diagonal line (as shown in Figure 7.18). A classification tree is therefore expected to have lower performance than methods like discriminant analysis. One way to improve performance is to create new predictors that are derived from existing predictors, which can capture hypothesized relationships between predictors (similar to interactions in regression models).

Another performance issue with classification trees is that they require a large dataset in order to construct a good classifier. Recently, Breiman & Cutler introduced "Random Forests"[3], an extension to classification trees that tackles these issues. The basic idea is to create multiple classification trees from the data (and thus obtain a "forest") and combine their output to obtain a better classifier.

An Appealing feature of trees is that they handle missing data without having to impute values or delete observations with missing values. The method can also be extended to incorporate an importance ranking for the variables in terms of their impact on quality of the classification.

[3]For further details on Random Forests see `http://www.stat.berkeley.edu/users/breiman/RandomForests/cc_home.htm`

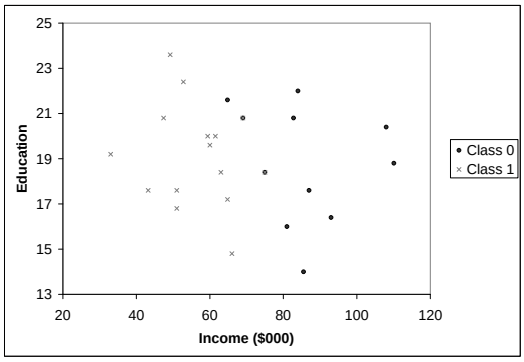


Figure 7.18: Scatterplot Describing a Two Predictor Case with Two Classes

From a computational aspect, trees can be relatively expensive to grow, because of the multiple sorting involved in computing all possible splits on every variable. Pruning the data using the validation set further adds computation time.

Finally, a very important practical advantage of trees is the transparent rules that they generate. Such transparency is often useful in managerial applications.

7.10 Exercises

Competitive auctions on eBay.com: The file eBayAuctions.xls contains information on 1972 auctions that transacted on eBay.com during May-June in 2004. The goal is to use these data in order to build a model that will classify competitive auctions from non-competitive ones. A competitive auction is defined as an auction with at least 2 bids placed on the auctioned item. The data include variables that describe the auctioned item (auction category), the seller (his/her eBay rating) and the auction terms that the seller selected (auction duration, opening price, currency, day-of-week of auction close). In addition, we have the price that the auction closed at. The goal is to predict whether the auction will be competitive or not.

Data pre-processing: Create dummy variables for the categorical predictors. These include Category (18 categories), Currency (USD, GBP, EURO), EndDay (Mon-Sun), and Duration (1,3,5,7, or 10 days). Split the data into training and validation datasets using a 60%-40% ratio.

1. Fit a classification tree using all predictors, using the best pruned tree. To avoid overfitting, set the minimum number of records in a leaf node to 50. Also, set the maximum number of levels to be displayed at 7 (the maximum allowed in XLminer). To remain within the limitation of 30 predictors, combine some of the categories of categorical predictors. Write down the results in terms of rules.
2. Is this model practical for predicting the outcome of a new auction?
3. Describe the interesting and un-interesting information that these rules provide.
4. Fit another classification tree (using the best pruned tree, with a minimum number of records per leaf node = 50, and maximum allowed number of displayed levels), this time only with predictors that can be used for predicting the outcome of a new auction. Describe the resulting tree in terms of rules. Make sure to report the smallest set of rules required for classification.
5. Plot the resulting tree on a scatterplot: use the two axes for the two best (quantitative) predictors. Each auction will appear as a point, with coordinates corresponding to its values on those two predictors. Use different colors or symbols to separate competitive and non-competitive auctions. Draw lines (you can sketch these by hand or use Excel) at the values that create splits. Does this splitting seem reasonable with respect to the meaning of the two predictors? Does it seem to do a good job of separating the two classes?
6. Examine the lift chart and the classification table for the above tree. What can you say about the predictive performance of this model?
7. Based on this last tree, what can you conclude from these data about the chances of an auction transacting and its relationship to the auction settings set by the seller (duration, opening price, ending day, currency)? What would you recommend for a seller as the strategy that will most likely lead to a competitive auction?

Predicting Delayed Flights: The file FlightDelays.xls contains information on all commercial flights departing the Washington DC area and arriving at New York during January 2004. For each flight there is information on the departure and arrival airports, the distance of the route, the scheduled time and date of the flight, etc. The variable that we are trying to predict is whether or not a flight is delayed. A delay is defined as an arrival that is at least 15 minutes later than scheduled.

1. Preprocessing: create dummies for day of week, carrier, departure airport, and arrival airport. This will give you 17 dummies. Bin the scheduled departure time into 2-hour

bins (in XLMiner use Data Utilities¿Bin Continuous Data and select 8 bins with equal width). This will avoid treating the departure time as a continuous predictor, because it is reasonable that delays are related to rush-hour times. Partition the data into training and validation sets.

2. Fit a classification tree to the flight delay variable, using all the relevant predictors. Use the best pruned tree without a limitation on the minimum number of observations in the final nodes. Express the resulting tree as a set of rules.
3. If you needed to fly between DCA and EWR on a Monday at 7:00 AM, would you be able to use this tree? What other information would you need? Is it available in practice? What information is redundant?
4. Fit another tree, this time excluding the day-of-month predictor (Why?). Select the option of seeing both the full tree and the best pruned tree. You will find that the best pruned tree contains a single terminal node.
   (a) How is this tree used for classification? (What is the rule for classifying?)
   (b) What is this rule equivalent to?
   (c) Examine the full tree. What are the top three predictors according to this tree?
   (d) Why, technically, does the pruned tree result in a tree with a single node?
   (e) What is the disadvantage of using the top levels of the full tree as opposed to the best pruned tree?
   (f) Compare this general result to that from logistic regression in the example in Chapter 8. What are possible reasons for the classification tree's failure to find a good predictive model?

Predicting Prices of Used Cars (Regression Trees): The file ToyotaCorolla.xls contains the data on used cars (Toyota Corolla) on sale during late summer of 2004 in the Netherlands. It has 1436 records containing details on 38 attributes including Price, Age, Kilometers, Horsepower, and other specifications. The goal is to predict the price of a used Toyota Corolla based on its specifications. (The example in section 7.8 is a subset of this dataset).

Data pre-processing: Create dummy variables for the categorical predictors (Fuel Type and Color). Split the data into training (50%), validation (30%), and test (20%) datasets.

1. Run a Regression Tree (RT) using Prediction menu in XLMiner with the output variable *Price* and input variables *Age_08_04, KM, Fuel_Types, HP, Automatic, Doors, Quarterly_Tax, Mfg_Guarantee, Guarantee_Period, Airco, Automatic_Airco, CD_Player, Powered_Windows, Sport_Model, and Tow_Bar*. Normalize the variables. Keep the minimum #records in a terminal node to 1 and scoring option to Full Tree, to make the run least restrictive.
   (a) What appear to be the 3-4 most important car specifications for predicting the car's price?
   (b) Compare the prediction errors of the training, validation, and test sets by examining their RMS Error and by plotting the three boxplots. What is happening with the training set predictions? How does the predictive performance of the test set compare to the other two? Why does this occur?
   (c) How can we achieve predictions for the training set that are not equal to the actual prices?
   (d) If we used the best pruned tree instead of the full tree, how would this affect the predictive performance for the validation set? (Hint: does the full tree use the validation data?)

2. Let us see the effect of turning the price variable into a categorical variable. First, create a new variable that categorizes price into 20 bins. Use *DataUtilities Bincontinuousdata* to categorize *Price* into 20 bins of equal intervals (leave all other options at their default). Now repartition the data keeping *Binned_Price* instead of *Price*. Run a Classification Tree (CT) using Classification menu of the XLMiner with the same set of input variables as in the RT, and with *Binned_Price* as the output variable. Keep the minimum #records in a terminal node to 1 and uncheck the Prune Tree option, to make the run least restrictive.
   (a) Compare the tree generated by CT with the one generated by the RT. Are they different? (look at structure, the top predictors, size of tree, etc.) Why?
   (b) Predict the price, using the RT and the CT, of a used Toyota Corolla with the following specifications:

Age_08_04	77
KM	117,000
Fuel_Type	Petrol
HP	110
Automatic	No
Doors	5
Quarterly_Tax	100
Mfg_Guarantee	No
Guarantee_Period	3
Airco	Yes
Automatic_Airco	No
CD_Player	No
Powered_Windows	No
Sport_Model	No
Tow_Bar	Yes

   (c) Compare the predictions in terms of the variables that were used, the magnitude of the difference between the two predictions, and the advantages/disadvantages of the two methods.

Chapter 8

Logistic Regression

8.1 Introduction

Logistic regression extends the ideas of linear regression to the situation where the dependent variable, Y, is categorical. We can think of a categorical variable as dividing the observations into classes. For example, if Y denotes a recommendation on holding /selling / buying a stock, then we have a categorical variable with 3 categories. We can think of each of the stocks in the dataset (the observations) as belonging to one of three classes: the "hold" class, the "sell" class, and the "buy" class. Logistic regression can be used for classifying a new observation, where the class is unknown, into one of the classes, based on the values of its predictor variables (called "classification"). It can also be used in data (where the class is known) to find similarities between observations within each class in terms of the predictor variables (called "profiling"). Logistic regression is used in applications such as:

1. Classifying customers as returning or non-returning (classification)
2. Finding factors that differentiate between male and female top executives (profiling)
3. Predicting the approval or disapproval of a loan based on information such as credit scores (classification).

In this chapter we focus on the use of logistic regression for classification. We deal only with a binary dependent variable, having two possible classes. At the end we show how the results can be extended to the case where Y assumes more than two possible outcomes. Popular examples of binary response outcomes are success/failure, yes/no, buy/don't buy, default/don't default, and survive/die. For convenience we often code the values of a binary response Y as 0 and 1.

Note that in some cases, we may choose to convert continuous data or data with multiple outcomes into binary data for purposes of simplification, reflecting the fact that decision-making may be binary (approve the loan / don't approve, make an offer/ don't make an offer). As with multiple linear regression, the independent variables $X_1, X_2, \cdots, X_k$ may be categorical or continuous variables or a mixture of these two types. While in multiple linear regression the aim is to predict the value of the continuous Y for a new observation, in logistic regression the goal is to predict which class a new observation will belong to, or simply to *classify* the observation into one of the classes. In the stock example, we would want to classify a new stock into one of the three recommendation classes: sell, hold, or buy.

In logistic regression we take two steps: the first step yields estimates of the **probabilities** of belonging to each class. In the binary case we get an estimate of $P(Y = 1)$, the probability of belonging to class 1 (which also tells us the probability of belonging to class 0). In the next

step we use a cutoff value on these probabilities in order to classify each case to one of the classes. For example, in a binary case, a cutoff of 0.5 means that cases with an estimated probability of $P(Y = 1) > 0.5$ are classified as belonging to class 1, whereas cases with $P(Y = 1) < 0.5$ are classified as belonging to class 0. This cutoff need not be set at 0.5. When the event in question is a low probability event, a higher-than-average cutoff value, though still below 0.5, may be sufficient to classify a case as belonging to class 1.

8.2 The Logistic Regression Model

The logistic regression model is used in a variety of fields – whenever a structured model is needed to explain or predict categorical (and in particular binary) outcomes. One such application is in describing choice behavior in econometrics which is useful in the context of the above example (see box).

Logistic Regression and Consumer Choice Theory

In the context of choice behavior, the logistic model can be shown to follow from the *random utility theory* developed by Manski (1977) as an extension of the standard economic theory of consumer behavior. In essence, the consumer theory states that when faced with a set of choices a consumer makes a choice which has the highest utility (a numeric measure of worth with arbitrary zero and scale). It assumes that the consumer has a preference order on the list of choices that satisfies reasonable criteria such as transitivity. The preference order can depend on the individual (e.g., socioeconomic characteristics) as well as attributes of the choice. The random utility model considers the utility of a choice to incorporate a random element. When we model the random element as coming from a "reasonable" distribution, we can logically derive the logistic model for predicting choice behavior.

The idea behind logistic regression is straightforward: instead of using Y as the dependent variable, we use a function of it, which is called the *logit*. To understand the logit, we take two intermediate steps: First, we look at p, the probability of belonging to class 1 (as opposed to class 0). In contrast to Y, the class number, which only takes the values 0 and 1, p can take any value in the interval $[0, 1]$. However, if we express p as a linear function of the q predictors[1] in the form:

$$p = \beta_0 + \beta_1 x_1 + \beta_2 x_2 + \cdots + \beta_q x_q \tag{8.1}$$

It is not guaranteed that the right hand side will lead to values within the interval $[0, 1]$. The fix is to use a non-linear function of the predictors in the form

$$p = \frac{1}{1 + e^{-(\beta_0 + \beta_1 x_1 + \beta_2 x_2 + \cdots + \beta_q x_q)}} \tag{8.2}$$

This is called the *logistic response function*. For any values of $x_1, \ldots, x_q$ the right hand side will always lead to values in the interval $[0, 1]$. Although this form solves the problem mathematically, sometimes we prefer to look at a different measure of belonging to a certain class, known as *odds*. The *odds* of belonging to class 1 ($Y = 1$) is defined as the ratio of the probability of belonging to class 1 to the probability of belonging to class 0:

$$odds = \frac{p}{1 - p} \tag{8.3}$$

[1] Unlike elsewhere in the book where p denotes the number of predictors, in this chapter by q, to avoid confusion with the probability p.

This metric is very popular in horse races, sports, gambling in general, epidemiology, and many other areas. Instead of talking about the *probability* of winning or contacting a disease, people talk about the *odds* of winning or contacting a disease. How are these two different? If, for example, the probability of winning is 0.5, then the odds of winning are $0.5/0.5 = 1$. We can also perform the reverse calculation: Given the odds of an event, we can compute its probability by manipulating equation (8.3):

$$p = \frac{odds}{1 + odds} \tag{8.4}$$

We can write the relation between the odds and the predictors as:

$$odds = e^{\beta_0 + \beta_1 x_1 + \beta_2 x_2 + \cdots + \beta_q x_q} \tag{8.5}$$

Now, if we take a log on both sides, we get the standard formulation of a logistic model:

$$\log(odds) = \beta_0 + \beta_1 x_1 + \beta_2 x_2 + \cdots + \beta_q x_q \tag{8.6}$$

The $\log(odds)$ is called the *logit*, and it takes values from $-\infty$ to ∞. Thus, our final formulation of the relation between the response and the predictors uses the logit as the dependent variable, and models it as a **linear** function of the q predictors.

To see the relation between the probability, odds, and logit of belonging to class 1, look at Figure 8.1, which shows the odds (top) and logit (bottom) as a function of p. Notice that the odds can take any non-negative value, and that the logit can take any real value.

Let us examine some data to illustrate the use of logistic regression.

8.2.1 Example: Acceptance of Personal Loan

Recall the example described in Chapter 7, of acceptance of a personal loan by Universal Bank. The bank's dataset includes data on 5000 customers. The data include customer demographic information (Age, Income, etc.), customer response to the last personal loan campaign (Personal Loan), and the customer's relationship with the bank (mortgage, securities account, etc.). Among these 5000 customers only 480 (= 9.6%) accepted the personal loan that was offered to them in a previous campaign. The goal is to find characteristics of customers who are most likely to accept the loan offer in future mailings.

Data Preprocessing

We start by partitioning the data randomly, using a standard 60%-40% rate, into training and validation sets. We will use the training set to fit a model and the validation set to assess the model's performance.

Next, we create dummy variables for each of the categorical predictors. Except for Education which has three categories, the remaining four categorical variables have two categories. We therefore need $6 = 2+1+1+1+1$ dummy variables to describe these five categorical predictors. In XLMiner's classification functions the response can remain in text form ('yes', 'no', etc.), but the predictor

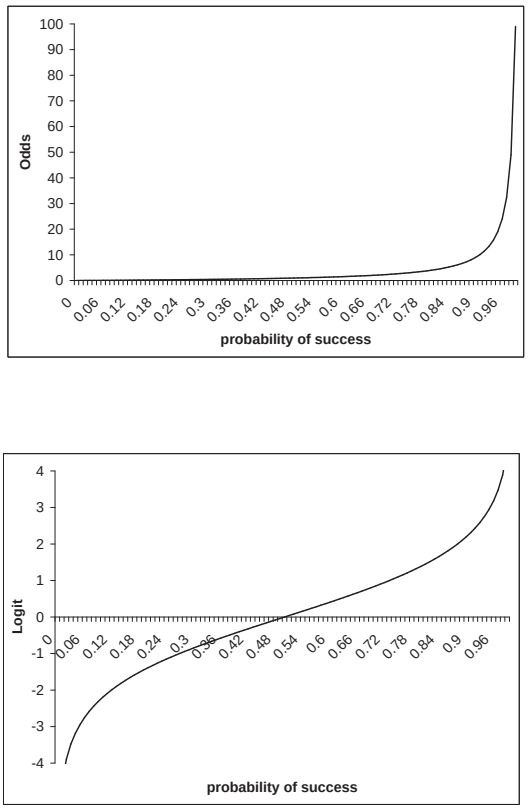


Figure 8.1: Odds (Top Panel) and Logit (Bottom Panel) as a Function of p

variables must be coded into dummy variables. We use the following coding:

$$\text{EducProf} = \begin{cases} 1 & \text{if education is "Professional"} \\ 0 & \text{otherwise} \end{cases}$$

$$\text{EducGrad} = \begin{cases} 1 & \text{if education is at "Graduate" level} \\ 0 & \text{otherwise} \end{cases}$$

$$\text{Securities} = \begin{cases} 1 & \text{if customer has securities account in bank} \\ 0 & \text{otherwise} \end{cases}$$

$$\text{CD} = \begin{cases} 1 & \text{if customer has CD account in bank} \\ 0 & \text{otherwise} \end{cases}$$

$$\text{Online} = \begin{cases} 1 & \text{if customer uses online banking} \\ 0 & \text{otherwise} \end{cases}$$

$$\text{CreditCard} = \begin{cases} 1 & \text{if customer holds Universal Bank credit card} \\ 0 & \text{otherwise} \end{cases}$$

8.2.2 A Model with a Single Predictor

Consider first a simple logistic regression model with just one independent variable. This is analogous to the simple linear regression model in which we fit a straight line to relate the dependent variable, Y, to a single independent variable, X.

Let us construct a simple logistic regression model for classification of customers using the single predictor *Income*. The equation relating the dependent variable to the explanatory variable in terms of probabilities is:

$$\text{Prob}\ (PersonalLoan = \text{`}yes'|Income = x) = \frac{1}{1 + e^{-(\beta_0+\beta_1 x)}}$$

or, equivalently, in terms of odds

$$\text{Odds}(PersonalLoan = \text{`}yes') = e^{\beta_0+\beta_1 x}. \tag{8.7}$$

The maximum likelihood estimates (more on this below) of the coefficients for the model are: $b_0 = -6.3525$, and $b_1 = 0.0392$. So the fitted model is:

$$\text{Prob}\ (PersonalLoan = \text{`}yes'|Income = x) = \frac{1}{1 + e^{6.3525-0.0392x}}. \tag{8.8}$$

Although logistic regression can be used for prediction in the sense that we predict the *probability* of a categorical outcome, it is most often used for classification. To see the difference between the two, think about predicting the probability of a customer accepting the loan offer as opposed to classifying the customer as an accepter/non-accepter. From Figure 8.2 it can be seen that the loan acceptance can yield numbers between 0 and 1. In order to end up with classifications into either 0 or 1 (e.g., a customer either accepts the loan offer or not) we need a cutoff value. This is true in the case of multiple predictor variables as well.

The Cutoff Value

Given the values for a set of predictors, we can predict the probability that each observation belongs to class 1. The next step is to set a cutoff on these probabilities so that each observation is classified into one of the two classes. This is done by setting a cutoff value, c, such that observations with probabilities above c are classified as belonging to class 1, and observations with probabilities below c are classified as belonging to class 0.

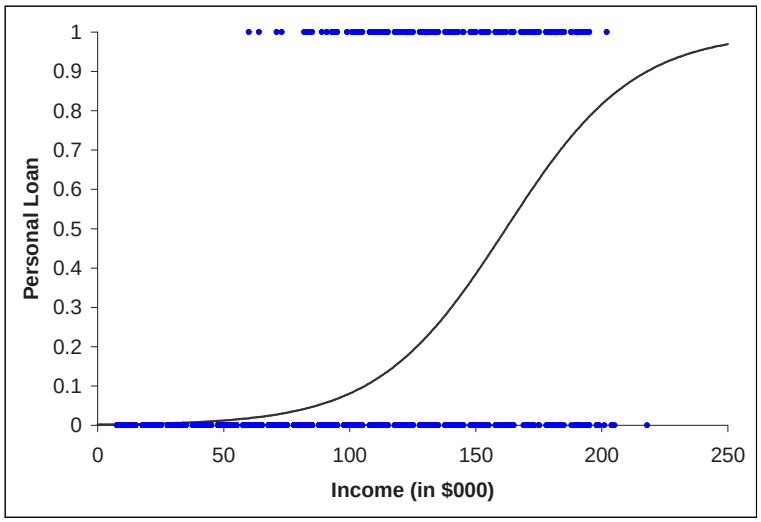


Figure 8.2: Plot of Data Points (Personal Loan as a Function of Income) and the Fitted Logistic Curve

In the Universal Bank example, in order to classify a new customer as an acceptor/non-acceptor of the loan offer, we use the information on his/her income by plugging it into the fitted equation in (8.8). This yields an estimated probability of accepting the loan offer. We then compare it to the cutoff value. The customer is classified as an acceptor if the probability of his/her accepting the offer is above the cutoff. If we prefer to look at *odds* of accepting rather than the probability, an equivalent method is to use the equation in (8.7) and compare the odds to $c/(1-c)$. If the odds are higher than this number, the customer is classified as an acceptor. If it is lower, we classify the customer as a non-acceptor. For example, the odds of acceptance for a customer with a $50K annual income are estimated from the model as

$$\text{Odds}(PersonalLoan = \text{`}yes\text{'}) = e^{-6.5325} = 0.0017 \tag{8.9}$$

For a cutoff value of 0.5, we compare the odds to 1. Alternatively, we can compute the probability of acceptance as

$$\frac{odds}{1+odds} = 0.0017$$

and compare it directly with the cutoff value of 0.5. In both cases we would classify this customer as a non-acceptor of the loan offer.

Different cutoff values lead to different classifications and consequently, different confusion matrices. There are several approaches to determining the "optimal" cutoff probability: A popular cutoff value for a two-class case is 0.5. The rationale is to assign an observation to the class in which its probability of membership is highest. A cutoff can also be chosen to maximize overall accuracy. This can be determined using a one-way data table in Excel (see Chapter 4). The overall accuracy is computed for various values of the cutoff value, and the cutoff value that yields maximum accuracy is chosen. The danger is, of course, overfitting. Alternatives to maximizing accuracy are to maximize sensitivity subject to some minimum level of specificity, or minimize false positives subject to some maximum level of false negatives etc. Finally, a cost- approach is to find a cutoff value that

minimizes the expected cost of misclassification. In this case one must specify the misclassification costs and the prior probabilities of belonging to each class.

8.2.3 Estimating the Logistic Model From Data: Computing Parameter Estimates

In logistic regression, the relation between Y and the beta parameters is nonlinear. For this reason the beta parameters are not estimated using the method of least squares (as in multiple regression). Instead, a method called Maximum Likelihood is used. The idea, in brief, is to find the estimates that maximize the chance of obtaining the data that we have. This requires iterations using a computer program. The method of maximum likelihood ensures good asymptotic (large sample) properties for the estimates. Under very general conditions maximum likelihood estimators are:

- *Consistent* : the probability of the estimator differing from the true value approaches zero with increasing sample size
- *Asymptotically Efficient* : the variance is the smallest possible among consistent estimators
- *Asymptotically Normally-Distributed*: This allows us to compute confidence intervals and perform statistical tests in a manner analogous to the analysis of linear multiple regression models, provided the sample size is 'large.'

Algorithms to compute the coefficient estimates and confidence intervals are iterative and less robust than algorithms for linear regression. Computed estimates are generally reliable for well-behaved datasets where the number of observations with dependent variable values of both 0 and 1 are 'large'; their ratio is 'not too close' to either zero or one; and when the number of coefficients in the logistic regression model is small relative to the sample size (say, no more than 10%). As with linear regression, collinearity (strong correlation amongst the independent variables) can lead to computational difficulties. Computationally intensive algorithms have been developed recently that circumvent some of these difficulties. For technical details on the Maximum Likelihood estimation in logistic regression see Hosmer & Lemeshow (2000).

To illustrate a typical output from such a procedure, look at the output in Figure 8.3 for the logistic model fitted to the training set of 3000 Universal Bank customers. The dependent variable is $PersonalLoan$ with 'Yes' defined as the 'success' (this is equivalent to setting the variable to 1 for an acceptor and 0 for a non-acceptor). Here we use all 12 predictors.

Ignoring p-values for the coefficients, a model based on all 12 predictors would have the estimated logistic equation

$$\begin{aligned} logit \;=\; & -13.201 - 0.045\ \text{Age} + 0.057\ \text{Experience} + 0.066\ \text{Income} + 0.572\ \text{Family} \\ & +0.18724874\ \text{CCAvg} + 0.002\ \text{Mortgage} - 0.855\ \text{Securities} + 3.469\ \text{CD} \\ & -0.844\ \text{Online} - 0.964\ \text{Credit Card} + 4.589\ \text{EducGrad} + 4.523\ \text{EducProf} \end{aligned} \tag{8.10}$$

The positive coefficients for the dummy variables CD, EducGrad, and EducProf mean that holding a CD account, having graduate education or professional education (all marked by "1" in the dummy variables) are associated with higher probabilities of accepting the loan offer. On the other hand, having a securities account, using online banking, and owning a Universal Bank credit card are associated with lower acceptance rates. For the continuous predictors, positive coefficients indicate that a higher value on that predictor is associated with a higher probability of accepting the loan offer (e.g., income: higher income customers tend more to accept the offer). Similarly, negative coefficients indicate that a higher value on that predictor is associated with a lower probability of accepting the loan offer (e.g., Age :older customers tend less to accept the offer).

The Regression Model

Input variables	Coefficient	Std. Error	p-value	Odds
Constant term	-13.20165825	2.46772742	0.00000009	*
Age	-0.04453737	0.09096102	0.62439483	0.95643985
Experience	0.05657264	0.09005365	0.5298661	1.05820346
Income	0.0657607	0.00422134	0	1.06797111
Family	0.57155931	0.10119002	0.00000002	1.77102649
CCAvg	0.18724874	0.06153848	0.00234395	1.20592725
Mortgage	0.00175308	0.00080375	0.02917421	1.00175464
Securities Account	-0.85484785	0.41863668	0.04115349	0.42534789
CD Account	3.46900773	0.44893095	0	32.10486984
Online	-0.84355801	0.22832377	0.00022026	0.43017724
CreditCard	-0.96406376	0.28254223	0.00064463	0.38134006
EducGrad	4.58909273	0.38708162	0	98.40509796
EducProf	4.52272701	0.38425466	0	92.08635712

Figure 8.3: Logistic Regression Coefficient Table for Personal Loan Acceptance as a Function of 12 Predictors

If we want to talk about the *odds* of offer acceptance, we can use the last column (entitled "odds") to obtain the equation:

$$\begin{aligned}\text{Odds (PersonalLoan='yes')} = \; & e^{-13.201}(0.956)^{\text{Age}} \; (1.058)^{\text{Experience}} \; (1.068)^{\text{Income}} \\ & (1.771)^{\text{Family}} \; (1.206)^{\text{CCAvg}} \; (1.002)^{\text{Mortgage}} \\ & (0.425)^{\text{Securities}} \; (32.105)^{\text{CD}} \; (0.430)^{\text{Online}} \\ & (0.381)^{\text{CreditCard}} (98.405)^{\text{EducGrad}} \; (92.086)^{\text{EducProf}}\end{aligned} \tag{8.11}$$

Notice how positive coefficients in the logit model translate into coefficients larger than 1 in the odds model, and negative positive coefficients in the logit translate into coefficients smaller than 1 in the odds.

A third option is to look directly at an equation for the probability of acceptance, using equation (8.2). This is useful for estimating the probability of accepting the offer for a customer with given values of the 12 predictors[2]

[2]If all q predictors are categorical, each having m_q categories, then we need not compute probabilities/odds for each of the n observations. The number of different probabilities/odds is exactly $m_1 \times m_2 \times \ldots \times m_q$.

Odds and Odds Ratios

A common confusion is between odds and odds ratios. Since the odds are in fact a ratio (between the probability of belonging to class 1 and the probability of belonging to class 0), they are sometimes termed, erroneously, "odds ratios". However, odds ratios refer to the ratio of two odds! These are used to compare different classes of observations. For a categorical predictor, odds ratios are used to compare two categories. For example, we could compare loan offer acceptance for customers with professional education vs. graduate education by looking at the ratio of odds of loan acceptance for customers with professional education divided by the odds of acceptance for customers with graduate education. This would yield an *odds ratio*. Ratios above 1 would indicate that the odds of acceptance for professionally educated customers are higher than for customers with graduate level education.

8.2.4 Interpreting Results in Terms of Odds

Recall that the odds are given by

$$\text{odds} \quad = \exp(\beta_0 + \beta_1 x_1 + \beta_2 x_2 + \cdots + \beta_k x_k)$$

At first let us return to the single predictor example, where we model a customer's acceptance of a personal loan offer as a function of his/her income.

$$\text{Odds} \quad (PersonalLoan = \text{`}yes\text{'}) = \exp(\beta_0 + \beta_1 \quad Income)$$

We can think of the model as a multiplicative model of odds. The odds that a customer with income zero will accept the loan is estimated by $\exp(-6.535 + (0.039)(0)) = 0.0017$. These are the *base case odds*. In this example it is obviously economically meaningless to talk about a zero income; the value zero and the corresponding base case odds could be meaningful, however, in the context of other predictors. The odds of accepting the loan with an income of 100K will increase by a multiplicative factor of $\exp(0.039 \times 100) = 50.5$ over the base case, so the odds that such a customer will accept the offer are $\exp(-6.535 + (0.039)(100)) = 0.088$

To generalize this to the multiple predictor case, consider the 12 predictors in the personal loan offer example. The odds of a customer accepting the offer as a function of the 12 predictors are given in equation (8.11).

Suppose the value of $Income$, or in general x_1, is increased by one unit from x_1 to $x_1 + 1$, while the other predictors (denoted $x_2, \ldots, x_{12}$) are held at their current value. We get the odds ratio

$$\frac{odds(x_1, \ldots x_{12})}{odds(x_1 + 1, x_2, \ldots, x_{12})} = \frac{\exp(\beta_0 + \beta_1(x_1 + 1) + \beta_2 x_2) + \ldots + \beta_{12} x_{12}}{\exp(\beta_0 + \beta_1 x_1 + \beta_2 x_2 + \ldots + \beta_{12} x_{12})} = \exp(\beta_1)$$

This tells us that a single unit increase in x_1, holding $x_2, \ldots, x_{12}$ constant, is associated with an increase in the odds that a customer accepts the offer by a factor of $\exp(\beta_1)$. In other words, β_1 is the multiplicative factor by which the odds (of belonging to class 1) increase when the value of x_1 is increased by 1 unit, and *holding all other predictors constant*. If $\beta_1 < 0$ an increase in x_1 is associated with a decrease in the odds of belonging to class 1, whereas a positive value of β_1 is associated with an increase in the odds.

When a predictor is a dummy variable, the interpretation is technically the same, but has a different practical meaning. For instance, the coefficient for CD was estimated from the data to be 3.469. Recall that the reference group is customers not holding a CD account. We interpret this coefficient as follows: $\exp(3.469) = 32.105$ are the odds that a customer who has a CD account will

accept the offer relative to a customer who does not have a CD account, holding all other factors constant. This means that customers who have CD accounts in Universal Bank are more likely to accept the offer than customers without a CD account (holding all other variables constant).

The advantage of reporting results in odds as opposed to probabilities, is that statements such as those above are true for any value of x_1. Unless x_1 is a dummy variable, we cannot apply such statements about the effect of increasing x_1 by a single unit to probabilities. This is because the result depends on the actual value of x_1. So if we increase x_1 from, say, 3 to 4, the effect on p, the probability of belonging to class 1, will be different than if we increase x_1 from 30 to 31. In short, the change in the probability, p, for a unit increase in a particular predictor variable, while holding all other predictors constant, is not a constant it depends on the specific values of the predictor variables. We therefore talk about probabilities only in the context of specific observations.

8.3 Why Linear Regression is Inappropriate for a Categorical Response

Now that you have seen how logistic regression works, we explain why linear regression is not suitable. Technically, one can apply a multiple linear regression model to this problem, treating the dependent variable Y as continuous. Of course, Y must be coded numerically (e.g., "1" for customers that did accept the loan offer and "0" for customers who did not accept it). Although software will yield an output that at first glance may seem usual, a closer look will reveal several anomalies:

1. Using the model to predict Y for each of the observations (or classify them) yields predictions that are not necessarily 0 or 1.

2. A look at the histogram or probability plot of the residuals reveals that the assumption that the dependent variable (or residuals) follows a normal distribution is violated. Clearly, if Y takes only the values 0 and 1 it cannot be normally distributed. In fact, a more appropriate distribution for the number of 1's in the dataset is the binomial distribution with $p = P(Y = 1)$.

3. The assumption that the variance of Y is constant across all classes is violated. Since Y follows a binomial distribution, its variance is $np(1 - p)$. This means that the variance will be higher for classes where the probability of adoption, p, is near 0.5 than where it is near 0 or 1.

Below you will find partial output from running a multiple linear regression of PersonalLoan (coded as 1 for customers who accepted the loan offer and 0 otherwise) on 3 of the predictors.

The estimated model is:

$$\widehat{PersonalLoan} = -0.2346 + 0.0032Income + 0.0329Family + 0.27016363CD$$

To predict whether a new customer will accept the personal loan offer ($Y = 1$) or not ($Y = 0$), we input the information on its values for these three predictors. For example, we would predict that the loan offer acceptance of a customer with an annual income of 50K, with two family members who does not hold CD accounts in Universal Bank to be $-0.2346 + (0.0032)(50) + (0.0329)(2) = -0.009$. Clearly, this is not a valid "loan acceptance" value. Furthermore, the histogram of the residuals (Figure 8.5) reveals that the residuals are most likely not normally distributed. Therefore, our estimated model is based on violated assumptions.

8.4 Evaluating Classification Performance

The general measures of performance that were described in Chapter 4 are used to assess how well the logistic model does. Recall that there are several performance measures, the most popular ones

The Regression Model

Input variables	Coefficient	Std. Error	p-value	SS
Constant term	-0.23462872	0.01328709	0	27.26533127
Income	0.00318939	0.00009888	0	67.95861816
Family	0.03294198	0.00383914	0	4.53180361
CD Account	0.27016363	0.01788521	0	13.18045044

ANOVA

Source	df	SS	MS	F-statistic	p-value
Regression	3	85.67087221	28.5569574	494.364771	5.9883E-261
Error	2996	173.063797	0.057764952		
Total	2999	258.7346692			

Figure 8.4: Output for Multiple Linear Regression Model of *Personalloan* on Three Predictors

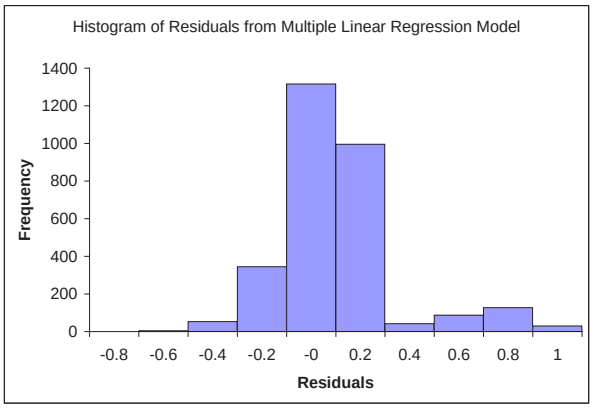


Figure 8.5: Histogram of Residuals from Multiple Linear Regression Model of Loan Acceptance on The Three Predictors. This Shows that the Residuals Do Not Follow a Normal Distribution, as the Model Assumes

XLMiner : Logistic Regression - Classification of Validation Data

Data range	['Universal BankGS.xls']'Data_Partition1'!$C$3019:$Q$5018	Back to Navigator

Cut off Prob.Val. for Success (Updatable)	0.5	(Updating the value here will NOT update value in summary report)

Row Id.	Predicted Class	Actual Class	Prob. for - 1 (success)	Log odds	Age	Experience	Income	Family
2	0	0	2.1351E-05	-10.75439275	45	19	34	3
3	0	0	3.34564E-06	-12.60785033	39	15	11	1
7	0	0	0.015822384	-4.13038073	53	27	72	2
8	0	0	0.000216511	-8.437650808	50	24	22	1
11	1	0	0.567824439	0.272980386	65	39	105	4

Figure 8.6: Scoring the Validation Data: XLMiner's Output for The First 5 Customers of Universal Bank (Based on 12 Predictors)

being measures based on the confusion matrix (accuracy alone or combined with costs) and the lift chart. The goal, like in other classification methods, is to find a model that accurately classifies observations to their class, using only the predictor information. A variant of this goal is to find a model that does a superior job of identifying the members of a particular class of interest (which might come at some cost to overall accuracy). Since the training data is used for selecting the model, we expect the model to perform quite well for those data, and therefore prefer to test its performance on the validation set. Recall that the data in the validation set were not involved in the model building, and thus we can use them to test the model's ability to classify data that it has not "seen" before.

To obtain the confusion matrix from a logistic regression analysis, we use the estimated equation to predict the probability of class membership for each observation in the validation set, and use the cutoff value to decide on the class assignment of these observations. We then compare these classifications to the actual class memberships of these observations. In the Universal Bank case we use the estimated model in (8.11) to predict the probability of adoption in a validation set that contains 2000 customers (these data were not used in the modeling step). Technically this is done by predicting the logit using the estimated model in (8.11) and then obtaining the probabilities p through the relation $p = \frac{e^{logit}}{1+e^{logit}}$. We then compare these probabilities to our chosen cutoff value in order to classify each of the 2000 validation observations as acceptors or non-acceptors. XLMiner automatically created the validation confusion matrix, and it is possible to obtain the detailed probabilities and classification for each observation. For example, Figure 8.6 shows a partial XLMiner output of scoring the validation set. It can be seen that the first 4 customers have a probability of accepting the offer that is lower than the cutoff of 0.5, and therefore they are classified as non-acceptors ("0"). The fifth customer's probability of acceptance is estimated by the model to exceed 0.5, and s/he is therefore classified as an acceptor ("1"), which in fact is a misclassification.

Another useful tool for assessing model classification performance is the lift (gains) chart (see Chapter 4). The left panel in Figure 8.7 illustrates the lift chart obtained for the personal loan offer model, using the validation set. The "lift" over the base curve indicates, for a given number of cases (read on the x-axis), the additional responders that you can identify by using the model.

The same information is portrayed in the "decile" chart (right panel in Figure 8.7): taking the 10% of the records that are ranked by the model as "most probable 1's" yields 7.7 times as many

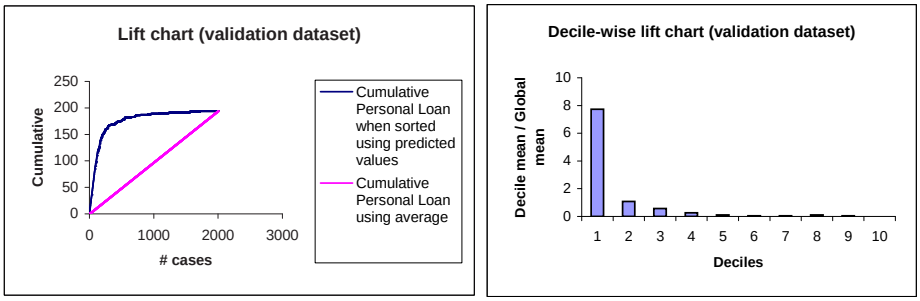


Figure 8.7: Lift and Decile Charts of Validation Data for Universal Bank Loan Offer: Comparing Logistic Model Classification (Blue) with Classification by Naive Model (Pink)

1's as would simply selecting 10% of the records at random.

8.4.1 Variable Selection

The next step includes searching for alternative models. As with multiple linear regression, we can build more complex models that reflect interactions between independent variables by including factors that are calculated from the interacting factors. For example if we hypothesize that there is an interactive effect between income and Family size we should add an interaction term of the form $Income \times Family$. The choice among the set of alternative models is guided primarily by performance on the validation data. For models that perform roughly equally well, simpler models are generally preferred over more complex ones. Note, also, that performance on validation data may be overly optimistic when it comes to predicting performance on data that have not been exposed to the model at all. This is because when the validation data are used to select a final model, we are selecting for how well the model performs with those data and therefore may be incorporating some of the random idiosyncracies of those data into the judgement about the best model. The model still may be the best among those considered, but it will probably not do as well with the unseen data. Therefore one must consider practical issues such as costs of collecting variables, error-proneness, and model complexity in the selection of the final model.

8.5 Evaluating Goodness-of-Fit

Assessing how well the model fits the data is important mainly when the purpose of the analysis is profiling, i.e., explaining the difference between classes in terms of predictor variables, and less when the aim is accurate classification. For example, if we are interested in characterizing loan offer acceptors vs. non-acceptors in terms of income, education, etc. we want to find a model that fits the data best. However, since over-fitting is a major danger in classification, a "too good" fit of the model to the training data should raise suspect. In addition, questions regarding the usefulness of specific predictors can arise even in the context of classification models. We therefore mention some of the popular measures that are used to assess how well the model fits the data. Clearly, we look at the training set in order to evaluate goodness-of-fit.

Residual df	2987
Std. Dev. Estimate	652.5175781
% Success in training data	9.533333333
# Iterations used	11
Multiple R-squared	0.65443069

Figure 8.8: Measures of Goodness-Of-Fit for Universal Bank Training Data with 12 Predictor Model

Overall Fit

As in multiple linear regression, we first evaluate the overall fit of the model to the data before looking at single predictors. We ask: Is this group of predictors better for explaining the different classes than a simple naive model[3]?

The deviance D is a statistic that measure overall goodness of fit. It is similar to the concept of sum-of-squared-errors (SSE) in the case of least squares estimation (used in linear regression). We compare the deviance of our model, D, (called *Std Dev Estimate* in XLMiner) to the deviance of the naive model, D_0. If the reduction in deviance is statistically significant (as indicated by a low p-value[4], or in XLMiner, by a high *multiple-R-squared*),, we consider our model to provide a good overall fit. XLMiner's Multiple-R-Squared measure is computed as $\frac{D_0-D}{D_0}$. Given the model deviance and the Multiple R^2 we can compute the null deviance by $D_0 = \frac{D}{1-R^2}$.

Finally, the confusion matrix and lift chart for the *training data* give a sense of how accurately the model classifies the data. If the model fits the data well, we expect it to classify these data accurately into their actual classes. Recall, however, that this does *not* provide a measure of future performance, since these confusion matrix and lift chart are based on the same data that were used for creating the best model! The confusion matrix and lift chart for the training set are therefore useful for the purpose of detecting over-fitting (manifested by "too good" results) and technical problems (manifested by "extremely bad" results) such as data entry, or even errors as basic as wrong choice of spreadsheet.

The Impact of Single Predictors

As in multiple linear regression, for each predictor X_i we have an estimated coefficient b_i and an associated standard error σ_i. The associated p-value indicates the statistical significance of the predictor X_i, or the significance of the contribution of this predictor beyond the other predictors. More formally, the ratio b_i/σ_i is used to test the hypotheses:

$$\begin{aligned} H_0 &: \beta_i = 0 \\ H_a &: \beta_i \neq 0 \end{aligned}$$

[3] A naive model is where no explanatory variables exist and each observation is classified as belonging to the majority class.

[4] The difference between the deviance of the naive model and deviance of the model at hand approximately follows a Chi-Square distribution with k degrees of freedom, where k is the number of predictors in the model at hand. Therefore, to get the p-value, compute the difference between the deviances (d) and then look up the probability that a Chi-square variable with k degrees of freedom is larger than d. This can be done using $= CHIDIST(d, k)$ in Excel.

Training Data scoring - Summary Report

Cut off Prob.Val. for Success (Updatable)	0.5	(Updating the value here will NOT update value in detailed report)

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

Error Report			
Class	# Cases	# Errors	% Error
1	286	85	29.72
0	2714	25	0.92
Overall	3000	110	3.67

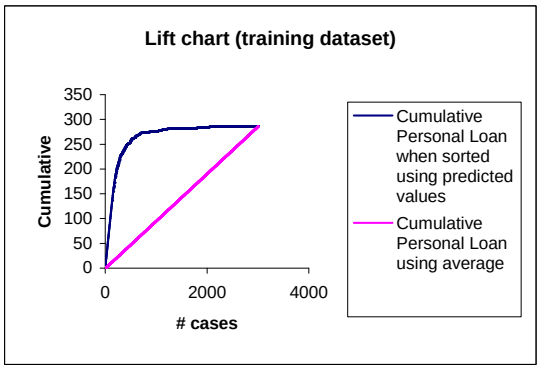


Figure 8.9: Confusion Matrix and Lift Chart for Training Data for Universal Bank Training Data with 12 Predictors

An equivalent set of hypotheses in terms of odds is

$$H_0 \quad : \quad \exp(\beta_i) = 1$$
$$H_a \quad : \quad \exp(\beta_i) \neq 1$$

Clearly if the sample is very large, the p-values will all be very small. If a predictor is judged to be informative, then we can look at the actual (estimated) impact that it has on the odds. Also, by comparing the odds of the different predictors we can see immediately which predictors have the most impact (given that the other predictors are accounted for), and which have the least impact.

8.6 Example of Complete Analysis: Predicting Delayed Flights

Predicting flight delays would be useful to a variety of organizations – airport authorities, airlines, aviation authorities. At times, joint task forces have been formed to address the problem. Such an organization, if it were to provide ongoing real-time assistance with flight delays, would benefit from some advance notice about flights that are likely to be delayed.

In this simplified illustration, we look at six predictors (see table below). The outcome of interest is whether the flight is delayed or not (delayed means more than 15 minutes late). Our data consist

of all flights from the Washington DC area into the NY city area during January 2004. The percent of delayed flights among these 2346 flights is 18%. The data were obtained from the Bureau of Transportation Statistics (available on the web at `www.transtats.bts.gov`).

The goal is to accurately predict whether a new flight, not in this dataset, will be delayed or not. Our dependent variable is a binary variable called "Delayed", coded as 1 for a delayed flight and 0 otherwise. We collected information on the following predictors:

Day of Week	Coded as: 1=Monday, 2=Tuesday,..., 7=Sunday
Departure time	Broken down into 18 intervals between 6:00AM and 10:00PM.
Origin	Three airport codes: DCA (Reagan National), IAD (Dulles), BWI (Baltimore-Washington Intl)
Destination	Three airport codes: JFK (Kennedy), LGA (LaGuardia), EWR (Newark)
Carrier	8 airline codes: CO (Continental), DH (Atlantic Coast), DL (Delta), MQ (American Eagle), OH (Comair), RU (Continental Express), UA (United), and US (USAirways).
Weather	coded as 1 if there was a weather-related delay.

Other information that is available on the website, such as distance and arrival time is irrelevant because we are looking at a certain route (distance, flight time, etc. should be approximately equal).

Below is a sample of the data for 20 flights:

Delayed	**Carrier**	**Day of week**	**Dep Time**	**Destination**	**Origin**	**Weather**
0	DL	2	728	LGA	DCA	0
1	US	3	1600	LGA	DCA	0
0	DH	5	1242	EWR	IAD	0
0	US	2	2057	LGA	DCA	0
0	DH	3	1603	JFK	IAD	0
0	CO	6	1252	EWR	DCA	0
0	RU	6	1728	EWR	DCA	0
0	DL	5	1031	LGA	DCA	0
0	RU	6	1722	EWR	IAD	0
1	US	1	627	LGA	DCA	0
1	DH	2	1756	JFK	IAD	0
0	MQ	6	1529	JFK	DCA	0
0	US	6	1259	LGA	DCA	0
0	DL	2	1329	LGA	DCA	0
0	RU	2	1453	EWR	BWI	0
0	RU	5	1356	EWR	DCA	0
1	DH	7	2244	LGA	IAD	0
0	US	7	1053	LGA	DCA	0
0	US	2	1057	LGA	DCA	0
0	US	4	632	LGA	DCA	0

The number of flights in each cell for Thur-Sun flights is approximately double that of the number of Mon-Wed flights. The data set includes four categorical variables: X_1 = Departure Airport, X_2 = Carrier, X_3 = Day Group (whether the flight was on Monday-Wednesday or Thursday-Sunday), and the response variable Y = Flight Status (delayed or not delayed) . In this example we have a binary response variable, or two classes. We start by looking at the pivot table for initial insight into the data: It appears that more flights departing on Thur-Sun are delayed than those leaving on Mon-Wed. Also, the worst airport (in term of delays) seems to be IAD. The worst Carrier, it appears,

Carrier	Continental	Delta	Northwest	US Airways	Total
Airport					
BWI	50 38	11 54	27 71	0 59	22 56
DCA	46 47	22 74	43 57	18 54	26 60
IAD	0 80	18 71	57 58	60 67	34 68
Total	47 49	18 69	41 60	20 56	27 60

Table 8.1: Number of Delayed Flights Out of Washington, DC Airports for Four Carriers by Day Group (Red Numbers are Delayed Flights on Mon-Wed, Black Numbers are Delayed Flights on Thur-Sun)

depends on the Day Group: on Mon-Wed Continental seems to have the most delays, whereas on Thur-Sun Delta has the most delays.

Our main goal is to find a model that can obtain accurate classifications of new flights, based on their predictor information. In some cases we might be interested in finding a certain % of flights that are most/least likely to get delayed. In other cases we may be interested in finding out which factors are associated with a delay (not only in this sample but in the entire population of flights on this route), and for those factors we would like to quantify these effects. A logistic regression model can be used for all these goals.

Data preprocessing

We first create dummy variables for each of the categorical predictors: 2 dummies for the Departure Airport (with IAD as the reference airport), 2 for arrival airport (with JFK as reference), 7 dummies for the Carrier (with USAirways as the reference carrier), 6 dummies for Day (with Sunday as reference group), 15 for departure hour (hourly intervals between 6AM and 10PM), blocked into hours. This yields a total of 33 dummies. In addition we have a single dummy for weather delays. This is a very large number of predictors. Some initial investigation and knowledge from airline experts led us to aggregate the day of week in a more compact way: it is known that delays are much more prevalent on this route on Sundays and Mondays. We therefore use a single dummy signifying whether it is a Sunday or a Monday (denoted by '1'), or not.

We then partition the data using a 60%-40% ratio into training and validation sets. We will use the training set to fit a model and the validation set to assess the model's performance.

Model fitting and estimation

The estimated model with 28 predictors is given in Figure 8.10. Notice how negative coefficients in the logit model (the "coefficient" column) translate into odds coefficients lower than 1, and positive logit coefficients translate into odds coefficients larger than 1.

Model Interpretation

The coefficient for Arrival Airport JFK is estimated from the data to be -0.67. Recall that the reference group is LGA. We interpret this coefficient as follows: $e^{-0.67} = 0.51$ are the odds of a flight arriving at JFK being delayed relative to a flight to LGA being delayed (=the base case odds), holding all other factors constant. This means that flights to LGA are more likely to be delayed than those to JFK (holding everything else constant). If we take into account statistical significance of the coefficients, we see that in general the departure airport is not associated with the chance of delays. For carriers, it appears that 4 carriers are significantly different from the base carrier (USAirways), with odds of 3.5-6.6 of delays relative to the other airlines. Weather has an enormous coefficient, which is not statistically significant. This is due to the fact that weather delays occurred

The Regression Model

Input variables	Coefficient	Std. Error	p-value	Odds
Constant term	-2.76648855	0.60903645	0.00000556	*
Weather	16.94781685	472.3040772	0.97137541	22926812
ORIGIN_BWI	0.31663841	0.407509	0.43715307	1.37250626
ORIGIN_DCA	-0.52621925	0.37920129	0.1652271	0.59083456
DEP_TIME_BLK_0700-0759	0.17635399	0.52038968	0.73469388	1.19286025
DEP_TIME_BLK_0800-0859	0.37122276	0.4879483	0.44678667	1.44950593
DEP_TIME_BLK_0900-0959	-0.2891154	0.61024719	0.6356656	0.74892575
DEP_TIME_BLK_1000-1059	-0.84254718	0.65849793	0.20072155	0.4306123
DEP_TIME_BLK_1100-1159	0.26919952	0.62188113	0.66510242	1.30891633
DEP_TIME_BLK_1200-1259	0.39577994	0.47712085	0.40681183	1.48554242
DEP_TIME_BLK_1300-1359	0.23689635	0.49711299	0.63368666	1.26730978
DEP_TIME_BLK_1400-1459	0.94953001	0.4257178	0.02571949	2.58449459
DEP_TIME_BLK_1500-1559	0.81428736	0.47320139	0.08528619	2.25756645
DEP_TIME_BLK_1600-1659	0.73656398	0.46096623	0.11007198	2.08874631
DEP_TIME_BLK_1700-1759	0.80683631	0.42013136	0.05480258	2.24080753
DEP_TIME_BLK_1800-1859	0.65816337	0.56922781	0.2475834	1.93124211
DEP_TIME_BLK_1900-1959	1.40413988	0.47974923	0.00342446	4.07202291
DEP_TIME_BLK_2000-2059	0.94785261	0.63308424	0.1343417	2.580163
DEP_TIME_BLK_2100-2159	0.76115495	0.45146817	0.09180449	2.14074731
DEST_EWR	-0.33785093	0.31752595	0.28732395	0.7133016
DEST_JFK	-0.66931868	0.2657896	0.01179471	0.5120573
CARRIER_CO	1.81500936	0.53502011	0.0006928	6.14113379
CARRIER_DH	1.25616693	0.52265555	0.016242	3.51193428
CARRIER_DL	0.41380161	0.33544913	0.21736139	1.51255703
CARRIER_MQ	1.73093832	0.32989427	0.00000015	5.64594936
CARRIER_OH	0.15529965	0.85175836	0.8553251	1.16800785
CARRIER_RU	1.27398086	0.51098496	0.01266023	3.57505608
CARRIER_UA	-0.59911883	1.17384589	0.60977846	0.54929543
Sun-Mon	0.53890741	0.16421914	0.00103207	1.71413302

Figure 8.10: Estimated Logistic Regression Model for Delayed Flights (Based on Training Set)

Validation Data scoring - Summary Report

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

Classification Confusion Matrix		
	Predicted Class	
Actual Class	delayed	non-delayed
delayed	18	154
non-delayed	3	705

Error Report			
Class	# Cases	# Errors	% Error
delayed	172	154	89.53
non-delayed	708	3	0.42
Overall	880	157	17.84

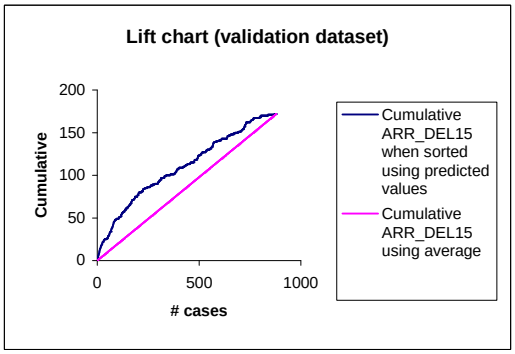


Figure 8.11: Confusion Matrix, Error Rates, and Lift Chart for the Flight Delay Validation Data

only on two days (Jan 26, 27), and those affected only some of the flights. Flights leaving on Sunday or Monday have, on average, odds of 1.7 of delays relative to other days of the week. Also, odds of delays appear to change over the course of the day, with the most noticeable difference between 7PM-8PM and the reference category 6AM-7AM.

Model Performance

How should we measure the performance of models? One possible measure is "percent of flights correctly classified." Accurate classification can be obtained from the confusion matrix for the validation data. The confusion matrix gives a sense of the classification accuracy and what type of misclassification is more frequent. From the confusion matrix and error rates in Figure 8.11 it can be seen that the model does better in classifying non-delayed flights correctly, and is less accurate in classifying flights that were delayed. (Note: the same pattern appears in the confusion matrix for the training data, so it is not surprising to see it emerge for new data). If there is a non-symmetric cost structure such that one type of misclassification is more costly than the other, then the cutoff value can be selected to minimize the cost. Of course, this tweaking should be carried out on the training data, and only assessed using the validation data.

In most conceivable situations, it is likely that the purpose of the model will be to identify those flights most likely to be delayed so that resources can be directed towards either reducing the delay or mitigating its effects. Air traffic controllers might work to open up additional air routes, or allocate more controllers to a specific area for a short time. Airlines might bring on personnel to rebook passengers, and activate standby flight crews and aircraft. Hotels might allocate space for stranded travellers. In all cases, the resources available are going to be limited, and might vary over time and from organization to organization. In this situation, the most useful model would provide an ordering of flights by their probability of delay, letting the model users decide how far down that list to go in taking action. Therefore, model lift is a useful measure of performance – as you move down that list of flights, ordered by their delay probability, how much better does the model do in predicting delay than would a naive model which is simply the average delay rate for all flights? From the lift curve for the validation data (Figure 8.11) we see that our model is superior to the baseline (simple random selection of flights).

Residual df	1292
Std. Dev. Estimate	1124.323608
% Success in training data	19.37925814
# Iterations used	16
Multiple R-squared	0.13447483

Figure 8.12: Goodness-of-Fit Measures for Flight Delay Training Data

Training Data scoring - Summary Report

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

Classification Confusion Matrix		
	Predicted Class	
Actual Class	delayed	non-delayed
delayed	24	232
non-delayed	6	1059

Error Report			
Class	# Cases	# Errors	% Error
delayed	256	232	90.63
non-delayed	1065	6	0.56
Overall	1321	238	18.02

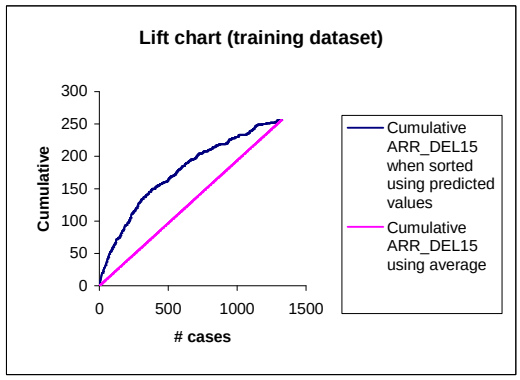


Figure 8.13: Confusion Matrix, Error Rates, and Lift Chart for Flight Delay Training Data

Goodness-of-fit

To see how closely the estimated logistic model fits the training data we look at "goodness-of-fit" measures such as the deviance and at the confusion matrix and lift chart that are computed from the training data. Figures 8.12 and 8.13 show some goodness-of-fit statistics that are part of the logistic regression output in XLMiner.

The naive model, in this case, would classify all flights as not delayed, since our training data contain more than 80% flights that were not delayed. How much better does our 6-predictor model perform? This can be assessed through the lift chart, confusion matrix, and goodness-of-fit measures based on the training data (Figures 8.12 and 8.13).

The model deviance is given by *Std. Dev. Estimate* as 1124. The low Multiple R^2 (13.45%) might lead us to believe that the model is not useful. To test this statistically, we can recover the naive model's deviance given by $D_0 = \frac{D}{1-R^2} = 1299$, and test whether the reduction from 1299 to 1124 (which measures the usefulness of our 28-predictor model over the naive model) is statistically significant we use Excel's CHIDIST function and obtain $CHIDIST(1299 - 1124, 28) = 0.00$. This tells us that our model is significantly better than the naive model, statistically speaking. But what does this mean from a practical point of view? The confusion matrix for the training data is different from what a naive model would yield. Since the majority of flights in the training set are not delayed, the naive model would classify all flights as not delayed. This would yield an overall error rate of 19.4%, whereas the 28 predictor model yields an error rate of 18.02%. This means that our model is indeed a better fit for the data than the naive model.

Variable Selection

From the coefficient table for the flights delay model, it appears that several of the variables might be dropped or coded differently. We further explore alternative models by examining pivot tables and charts and using variable selection procedures. First, we find that most carriers depart from a single airport: for those that depart from all three airports, the delay rates are similar regardless of airport. This means that there are multiple combinations of carrier and departure airport that do not include any flights. We therefore drop the departure airport distinction and find that the model performance and fit is not harmed. We also drop destination airport for a practical reason: not all carriers fly to all airports. Our model would then be invalid for prediction in non-existent combinations of carrier and destination airport. We also try re-grouping the carriers and hour of day into less categories that are more distinguishable with respect to delays. Finally we apply subset selection. Our final model includes only 12 predictors, and has the advantage of being more parsimonious. It does, however, include coefficients that are not statistically significant because our goal is prediction accuracy rather than model fit. Also, some of these variables have a practical importance (e.g., weather) and are therefore retained. Figure 8.14 displays the estimated model, with its goodness-of-fit measures, the training and validation confusion matrices and error rates, and the lift charts. It can be seen that this model competes well with the larger model in terms of accurate classification and lift.

We therefore conclude with a 7-predictor model that required only the knowledge of carrier, Day of week, hour of day, and whether it is likely that there will be a delay due to weather. The last piece of information is of course not known in advance, but we kept it in our model for purposes of interpretation. The impact of the other factors is estimated while "holding weather constant", i.e. (approximately) comparing days with weather delays to days without weather delays. If the aim is to predict in advance whether a particular flight will be delayed, a model without $Weather$ should be used. To conclude, we can summarize that the highest chance of a non-delayed flight from DC to NY, based on the data from January 2004, would be a flight on Mon-Fri during the late morning hours, on Delta, United, USAirways, or Atlantic Coast Airlines. And clearly, good weather is advantageous!

The Regression Model

Input variables	Coefficient	Std. Error	p-value	Odds
Constant term	-1.76942575	0.11373349	0	*
Weather	16.77862358	479.4146118	0.97208124	19358154
DEP_TIME_BLK_0600-0659	-0.62896502	0.36761174	0.08709048	0.53314334
DEP_TIME_BLK_0900-0959	-1.26741421	0.47863296	0.00809724	0.28155872
DEP_TIME_BLK_1000-1059	-1.37123489	0.52464402	0.00895813	0.25379336
DEP_TIME_BLK_1300-1359	-0.6303032	0.3188065	0.04803356	0.53243035
Sun-Mon	0.52237105	0.15871418	0.00099736	1.68602061
Carrier_CO_OH_MQ_RU	0.68775123	0.15049717	0.00000488	1.98923719

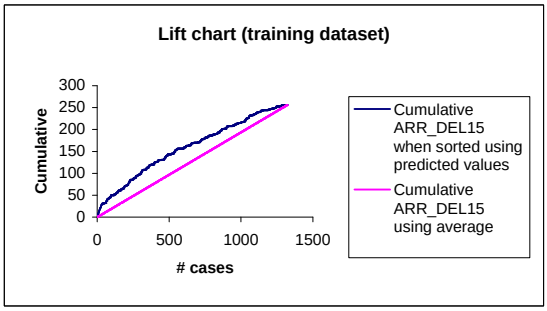
Residual df	
Std. Dev. Estimate	
% Success in training data	
# Iterations used	
Multiple R-squared	

Training Data scoring - Summary Report

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

Classification Confusion Matrix		
	Predicted Class	
Actual Class	1	0
1	19	237
0	0	1065

Error Report			
Class	# Cases	# Errors	% Error
1	256	237	92.58
0	1065	0	0.00
Overall	1321	237	17.94



Validation Data scoring - Summary Report

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

Classification Confusion Matrix		
	Predicted Class	
Actual Class	1	0
1	13	159
0	0	708

Error Report			
Class	# Cases	# Errors	% Error
1	172	159	92.44
0	708	0	0.00
Overall	880	159	18.07

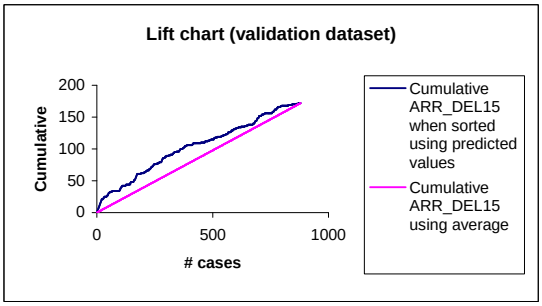


Figure 8.14: Output for Logistic Regression Model with Only 7 Predictors

8.7 Logistic Regression for More than 2 Classes

The logistic model for a binary response can be extended for more than two classes. Suppose there are m classes. Using a logistic regression model, for each observation we would have m probabilities of belonging to each of the m classes. Since the m probabilities must add up to 1, we need estimate only $m-1$ probabilities.

8.7.1 Ordinal Classes

Ordinal classes are classes that have a meaningful order. For example, in stock recommendations the there classes "buy", "hold", and "sell" can be treated as ordered. As a simple rule, if classes can be numbered in a meaningful way, we consider them ordinal. When the number of classes is large (typically more than 5), we can treat the dependent variable as continuous and perform multiple linear regression. When $m = 2$ the logistic model described above is used. We therefore need an extension of the logistic regression for a small number of ordinal classes ($3 \leq m \leq 5$). There are several ways to extend the binary-class case. Here we describe the *cumulative logit* method. For other methods see (Hosmer & Lemeshow, 2000).

For simplicity of interpretation and computation, we look at *cumulative* probabilities of class membership. For example, in the stock recommendations we have $m = 3$ classes. Let us denote them by 1="buy," 2="hold,", and 3="sell". The probabilities that are estimated by the model are $P(Y \leq 1)$, (the probability of a "buy" recommendation) and $P(Y \leq 2)$ (the probability of a "buy" or "hold" recommendation). The three non-cumulative probabilities of class membership can be easily recovered from the two cumulative probabilities:

$$\begin{aligned} P(Y=1) &= P(Y \leq 1) \\ P(Y=2) &= P(Y \leq 2) - P(Y \leq 1) \\ P(Y=3) &= 1 - P(Y \leq 2) \end{aligned}$$

Next, we want to model each logit as a function of the predictors. Corresponding to each of the $m-1$ cumulative probabilities is a logit. In our example we would have

$$\begin{aligned} logit(\text{buy}) &= \log \frac{P(Y \leq 1)}{1 - P(Y \leq 1)} \\ logit(\text{buy or hold}) &= \log \frac{P(Y \leq 2)}{1 - P(Y \leq 2)} \end{aligned}$$

Each of the logits is then modelled as a linear function of the predictors (like in the 2-class case). If in the stock recommendations we have a single predictor x then we have two equations:

$$\begin{aligned} logit(\text{buy}) &= \alpha_0 + \beta_1 x \\ logit(\text{buy or hold}) &= \beta_0 + \beta_1 x \end{aligned}$$

This means that both lines have the same slope (β_1) but different intercepts. Once the coefficients $\alpha_0, \beta_0, \beta_1$ are estimated, we can compute the class membership probabilities by rewriting the logit equations in terms of probabilities. For the 3-class case, for example, we would have

$$\begin{aligned} P(Y=1) &= P(Y \leq 1) = \frac{1}{1+e^{-(a_0+b_1 x)}} \\ P(Y=2) &= P(Y \leq 2) - P(Y \leq 1) = \frac{1}{1+e^{-(b_0+b_1 x)}} - \frac{1}{1+e^{-(a_0+b_1 x)}} \\ P(Y=3) &= 1 - P(Y \leq 2) = 1 - \frac{1}{1+e^{-(b_0+b_1 x)}} \end{aligned}$$

where a_0, b_0, and b_1 are the estimates obtained from the training set.

For each observation, we now have the estimated probabilities that it belongs to each of the classes. In our example, each stock would have three probabilities: for a "buy" recommendation, a "hold" recommendation, and a "sell" recommendation. The last step is to classify the observation into one of the classes. This is done by assigning it to the class with the highest membership probability. So if a stock had estimated probabilities $P(Y = 1) = 0.2$, $P(Y = 2) = 0.3$, and $P(Y = 3) = 0.5$, we would classify it as getting a "sell" recommendation.

This procedure is currently not implemented in XLMiner. Other non-Excel based packages that do have such an implementation are Minitab and SAS.

8.7.2 Nominal Classes

When the classes cannot be ordered and are simply different from one another, we are in the case of nominal classes. An example is the choice between several brands of cereal. A simple way to verify that the classes are nominal is when it makes sense to tag them as $A, B, C, ...$, and the assignment of letters to classes does not matter. For simplicity, let us assume there are $m = 3$ brands of cereal that consumers can choose from (assuming that each consumer chooses one). Then we estimate the probabilities $P(Y = A)$, $P(Y = B)$, and $P(Y = C)$. As before, if we know 2 of the probabilities, the third probability is determined. We therefore use one of the classes as the reference class. Let us use C as the reference brand.

The goal, once again, is to model the class membership as a function of predictors. So in the cereals example we might want to predict which cereal will be chosen if we know the cereal's price x.

Next, we form $m - 1$ pseudo-logit equations that are linear in the predictors. In our example we would have:

$$\begin{aligned} logit(\text{A}) &= \log \frac{P(Y = A)}{P(Y = C)} = \alpha_0 + \alpha_1 x \\ logit(\text{B}) &= \log \frac{P(Y = B)}{P(Y = C)} = \beta_0 + \beta_1 x \end{aligned}$$

Once the four coefficients are estimated from the training set we can estimate the class membership probabilities[5]:

$$\begin{aligned} P(Y = A) &= \frac{e^{a_0 + a_1 x}}{1 + e^{a_0 + a_1 x} + e^{b_0 + b_1 x}} \\ P(Y = B) &= \frac{e^{b_0 + b_1 x}}{1 + e^{a_0 + a_1 x} + e^{b_0 + b_1 x}} \\ P(Y = C) &= 1 - P(Y = A) - P(Y = B) \end{aligned}$$

where a_0, a_1, b_0, and b_1 are the coefficient estimates obtained from the training set.

Finally, an observation is assigned to the class that has the highest probability.

[5] From the two logit equations we see that

$$\begin{aligned} P(Y = A) &= P(Y = C)e^{\alpha_0 + \alpha_1 x} \\ P(Y = B) &= P(Y = C)e^{\beta_0 + \beta_1 x} \end{aligned}$$

and since $P(Y = A) + P(Y = B) + P(Y = C) = 1$ we have

$$\begin{aligned} & P(Y = C) = 1 - P(Y = C)e^{\alpha_0 + \alpha_1 x} - P(Y = C)e^{\beta_0 + \beta_1 x} \\ \rightarrow \quad & P(Y = C) = \frac{1}{e^{\alpha_0 + \alpha_1 x} + e^{\beta_0 + \beta_1 x}} \end{aligned}$$

By plugging in this form into the two equations above it we also obtain the membership probabilities in classes A and B.

8.8 Exercises

Financial condition of banks: The file Banks.xls includes data on a sample of 20 banks. The "Financial Condition" column records the judgment of an expert on the financial condition of each bank. This dependent variable takes one of two possible values - "weak" or "strong" - according to the financial condition of the bank. The predictors are two ratios used in financial analysis of banks: $TotExp/Assets$ is the ratio of Total Loans & Leases to Total Assets and $TotLns\&Lses/Assets$ is the ratio of Total Expenses to Total Assets. The target is to use the two ratios for classifying the financial condition of a new bank.

Run a logistic regression model (on the entire dataset) that models the status of a bank as a function of the two financial measures provided. Specify the "success" class as "weak" (this is similar to creating a dummy that is 1 for financially weak banks and 0 otherwise), and use the default cutoff value of 0.5.

1. Write the estimated equation that associates the financial condition of a bank with its two predictors in three formats:
   (a) The logit as a function of the predictors
   (b) The odds as a function of the predictors
   (c) The probability as a function of the predictors
2. Consider a new bank whose total loans-&-leases-to-assets ratio = 0.6 and total expenses-to-assets ratio = 0.11. From your logistic regression model, estimate the following four quantities for this bank (use Excel to do all the intermediate calculations; show your final answers to 4 decimal places): The logit, the odds, the probability of being financially weak, and the classification of the bank.
3. The cutoff value of 0.5 is used in conjunction with the probability of being financially weak. Compute the threshold that should be used if we want to make a classification based on the odds of being financially weak, and the threshold for the corresponding logit.
4. Interpret the estimated coefficient for the total expenses-to-assets ratio in terms of the odds of being financially weak.
5. When a bank that is in poor financial condition is misclassified as financially "strong," the misclassification cost is much higher than when a financially strong bank is misclassified as weak. In order to minimize the expected cost of misclassification, should the cutoff value for classification (which is currently at 0.5) be increased or decreased?

Identifying good system administrators: A management consultant is studying the roles played by experience and training in a system administrator's ability to complete a set of tasks in a specified amount of time. In particular, she is interested in discriminating between administrators who are able to complete given tasks within a specified time and those who are not. Data are collected on the performance of 75 randomly selected administrators. They are stored in the file SystemAdministrators.xls.

Using these data, the consultant performs a discriminant analysis. The variable "Experience" measures months of full time system administrator experience, while "Training" measures number of relevant training credits. The dependent variable "Completed" is either "yes" or "no", according to whether the administrator completed the tasks or not.

1. Create a scatterplot of Experience vs. Education using color or symbol to differentiate programmers that complete the task from those who did not complete it. Which predictor/s appear/s potentially useful for classifying task completion?

2. Run a logistic regression model with both predictors using the entire dataset as training data. Among those who complete the task, what is the percentage of programmers who are incorrectly classified as failing to complete the task?
3. In order to decrease this percentage above, should the cutoff probability be increased or decreased?
4. How much experience must be accumulated by a programmer with 4 years of education before his/her estimated probability of completing the task exceeds 50%?

Sales of Riding Mowers: A company that manufactures riding mowers wants to identify the best sales prospects for an intensive sales campaign. In particular, the manufacturer is interested in classifying households as prospective owners or non-owners on the basis of INCOME (in \$1000s) and LOT SIZE (in 1000 ft^2). The marketing expert looked at a random sample of 24 households, given in the file RidingMowers.xls. Use all the data to fit a logistic regression of ownership on the two predictors.

1. What percentage of households in the study were owners of a riding mower?
2. Create a scatterplot of income vs. lot size using color or symbol to differentiate owners from non-owners. From the scatterplot, which class seems to have higher average income? Owners or non-owners?
3. Among non-owners, what is the percentage of households correctly classified?
4. In order to increase the percentage of correctly classified household, should the cutoff probability be increased or decreased?
5. What are the odds that a household with a \$60K income and lot size of 20,000 ft^2 is an owner?
6. What is the classification of a household with a \$60K income and lot size of 20,000 ft^2?
7. What is the minimum income that a household with 16,000 ft^2 lot size should have before it is classified as an owner?

Competitive auctions on eBay.com: The file eBayAuctions.xls contains information on 1972 auctions that transacted on eBay.com during May-June in 2004. The goal is to use these data to build a model that will distinguish competitive auctions from non-competitive ones. A competitive auction is defined as an auction with at least 2 bids placed on the auctioned item. The data include variables that describe the auctioned item (auction category), the seller (his/her eBay rating) and the auction terms that the seller selected (auction duration, opening price, currency, day-of-week of auction close). In addition, we have the price that the auction closed at. The goal is to predict whether the auction will be competitive or not. **Data pre-processing**: Create dummy variables for the categorical predictors. These include Category (18 categories), Currency (USD, GBP, EURO), EndDay (Mon-Sun), and Duration (1,3,5,7, or 10 days). Split the data into training and validation datasets using a 60%-40% ratio.

1. Create pivot tables for the average of the binary dependent variable ("Competitive?") as a function of the various categorical variables (use the original variables, not the dummies). Use the information in the tables to reduce the number of dummies that will be used in the model. For example, categories that appear most similar with respect to the distribution of competitive auctions could be combined.
2. Run a logistic model with all predictors with a cutoff of 0.5. To remain within the limitation of 30 predictors, combine some of the categories of categorical predictors.

3. If we want to predict at the start of an auction whether it will be competitive, we cannot use the information on the closing price. Run a logistic model with all predictors as above excluding price. How does this model compare to the full model with respect to accurate prediction?
4. Interpret the meaning of the coefficient for closing price. Does closing price have a practical significance? Is it statistically significant for predicting competitiveness of auctions? (Use a 10% significance level.)
5. Use stepwise selection and an exhaustive search to find the model with the best fit to the training data. Which predictors are used?
6. Use stepwise selection and an exhaustive search to find the model with the lowest predictive error rate (use the validation data). Which predictors are used?
7. What is the danger in the best predictive model that you found?
8. Explain why the best-fitting model and the best predictive models are the same/ different.
9. If the major objective is accurate classification, what cutoff value should be used?
10. Based on these data, what auction settings set by the seller (duration, opening price, ending day, currency) would you recommend as the most likely to lead to a competitive auction?

Chapter 9

Neural Nets

9.1 Introduction

Neural networks, also called artificial neural networks, are models for classification and prediction. The neural network is based on a model of biological activity in the brain, where neurons are interconnected and learn from experience. Neural networks mimic the way that human experts learn. The learning and memory properties of neural networks resemble the properties of human learning and memory, and they also have a capacity to generalize from particulars.

A number of successful applications have been reported in financial applications (see Trippi and Turban, 1996) such as bankruptcy predictions, currency market trading, picking stocks and commodity trading, detecting fraud in credit card and monetary transactions, and in customer relationship management (CRM). There have also been a number of very successful applications of neural nets in engineering applications. One of the well known ones is ALVINN, an autonomous vehicle driving application for normal speeds on highways. Using as input a 30×32 grid of pixel intensities from a fixed camera on the vehicle, the classifier provides the direction of steering. The response variable is a categorical one with 30 classes such as "sharp left," "straight ahead," and "bear right."

The main strength of neural networks is their high predictive performance. Their structure supports capturing very complex relationships between predictors and a response, which is often not possible with other classifiers.

9.2 Concept and Structure of a Neural Network

The idea behind neural networks is to combine the input information in a very flexible way that captures complicated relationships among these variables and between them and the response variable. For instance, recall that in linear regression models the form of the relationship between the response and the predictors is assumed to be linear. In many cases the exact form of the relationship is much more complicated, or is generally unknown. In linear regression modeling we might try different transformations of the predictors, interactions between predictors, etc. In comparison, in neural networks the user is not required to specify the correct form. Instead, the network tries to learn from the data about such relationships. In fact, linear regression and logistic regression can be thought of as special cases of very simple neural networks that have only input and output layers and no hidden layers.

While researchers have studies numerous different neural network architectures, the most successful applications in data mining of neural networks have been *multilayer feedforward networks*. These are networks in which there is an *input layer* consisting of nodes that simply accept the input

values, and successive layers of nodes that receive input from the previous layers. The outputs of nodes in a layer are inputs to nodes in the next layer. The last layer is called the *output layer*. Layers between the input and output layers are known as *hidden layers*. A feedforward network is a fully connected network with a one-way flow and no cycles. Figure 9.2 shows a diagram for this architecture (two hidden layers are shown in this example).

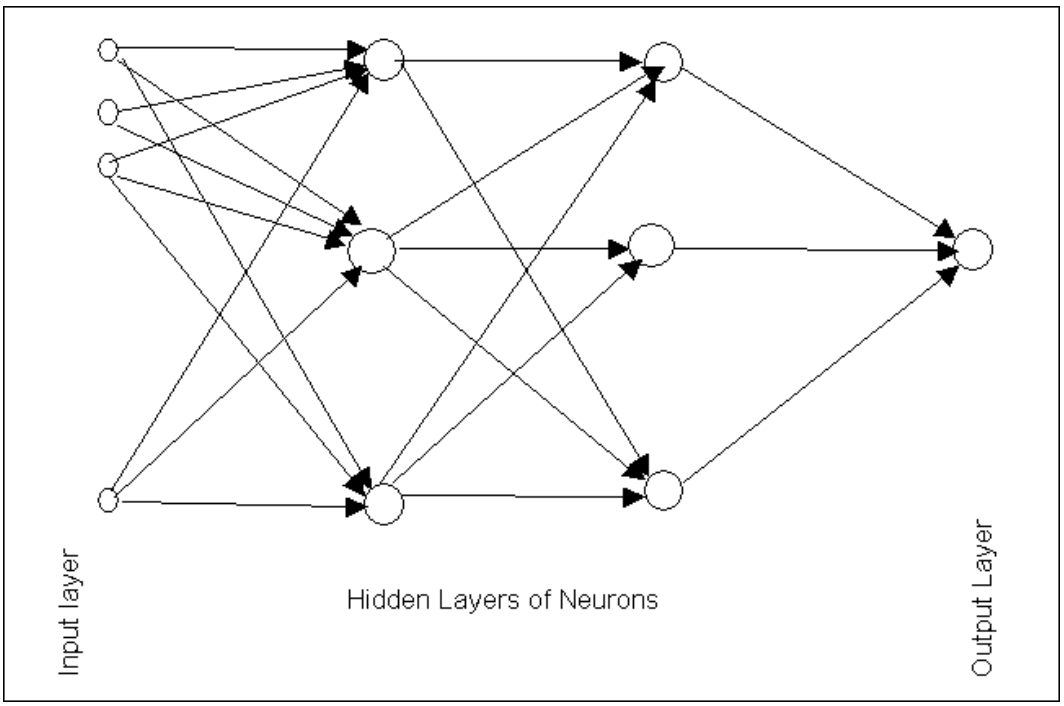


Figure 9.1: Multilayer Feed-Forward Neural Network

9.3 Fitting a Network to Data

To illustrate how a neural network is fitted to data, we start with a very small illustrative example. Although the method is by no means operational in such a small example, it is useful for explaining the main steps and operations, for showing how computations are done, and for integrating all the different aspects of neural network data fitting. We later discuss a more realistic setting.

9.3.1 Example 1: Tiny Dataset

Consider the following very small dataset. Table 9.3.1 includes information on a tasting score for a certain processed cheese. The two predictors are scores for fat and salt indicating the relative presence of fat and salt in the particular cheese sample (where 0 is the minimum amount possible in the manufacturing process, and 1 the maximum). The output variable is the cheese sample's consumer taste acceptance, where "1" indicates that a taste test panel likes the cheese and "0" that it does not like it. The diagram in Figure 9.2 describes an example of a typical neural net that could be used for predicting the acceptance for these data. We numbered the nodes in the example from 1-6. Nodes 1,2 belong to the input layer, nodes 3,4,5 belong to the hidden layer, and node 6 belongs to the output layer. The values on the connecting arrows are called *weights*, and the weight on the

obs	fat score	salt score	acceptance
1	0.2	0.9	1
2	0.1	0.1	0
3	0.2	0.4	0
4	0.2	0.5	0
5	0.4	0.5	1
6	0.3	0.8	1

Table 9.1: Tiny Example on Tasting Scores for Six Individuals and Two Predictors

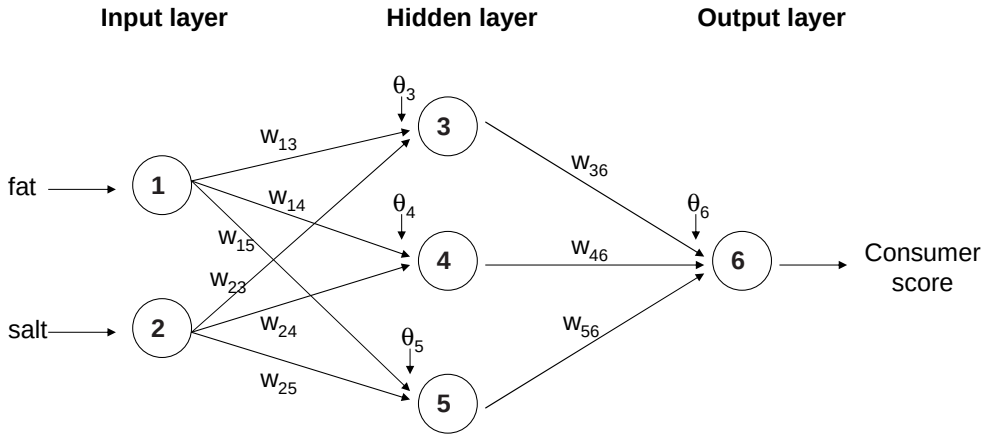


Figure 9.2: Diagram of a Neural Network for the Tiny Example. Circles Represent Nodes, $w_{i,j}$ on Arrows are Weights, and θ_j are Node "Bias" Values

arrow from node i to node j is denoted by $w_{i,j}$. The additional it bias nodes, denoted by θ_j, serve as an intercept for the output from node j. These are all explained in further detail below.

9.3.2 Computing Output of Nodes

We discuss the input and output of the nodes separately for each of the three types of layers (input, hidden, and output). The main difference is the function used to map from the input to the output of the node.

Input nodes take as input the values of the predictors. Their output is the same as the input. If we have p predictors, then the input layer will usually include p nodes. In our example there are two predictors and therefore the input layer (shown in Figure 9.2) includes two nodes. Each of these nodes feed into each of the nodes of the hidden layer. Consider the first observation: the input into the input layer is $fat = 0.2$ and $salt = 0.9$, and the output of this layer is also $x_1 = 0.2$ and $x_2 = 0.9$.

Hidden layer nodes take as input the output values from the input layer. The hidden layer in this example consists of 3 nodes. Each of the nodes receives input from all the input nodes. To compute the output of a hidden layer node, we compute a weighted sum of the inputs, and then apply a certain function to it. More formally, for a set of input values $x_1, x_2, \ldots, x_p$, we compute the output of node j by taking the weighted sum[1] $\theta_j + \sum_{i=1}^{p} w_{ij}x_i$, where $\theta_j, w_{1,j}, \ldots, w_{p,j}$ are weights

[1]Other options exist for combining inputs, like taking the maximum or minimum of the weighted inputs rather

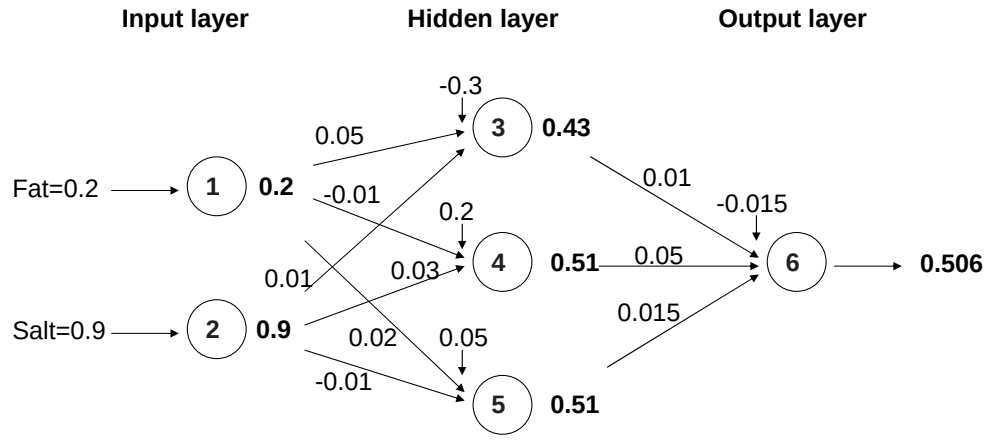


Figure 9.3: Computing Node Outputs (in **Bold**) Using the First Observation in the Tiny Example and a Logistic Function

that are initially set randomly, then adjusted as the network "learns." Note that θ_j, also called the "bias" of node j, is a constant that controls the level of contribution of node j. In the next step we take a function g of this sum. The function g, also called a *transfer function*, is some monotone function and examples are a linear function ($g(s) = bs$), an exponential function ($g(s) = \exp(bs)$), and a logistic/sigmoidal function ($g(s) = \frac{1}{1+e^{-s}}$. This last function is by far the most popular one in neural networks. Its practical value arises from the fact that it has a squashing effect on very small or very large values, but is almost linear in the range where the value of the function is between 0.1 and 0.9.

If we use a logistic function, then we can write the output of node j in the hidden layer as

$$output_j = g(\theta_j + \sum_{i=1}^{p} w_{ij}x_i) = \frac{1}{1 + e^{-(\theta_j + \sum_{i=1}^{p} w_{ij}x_i)}}. \tag{9.1}$$

Initializing the weights

The values of θ_j and w_{ij} are typically initialized to small (generally random) numbers in the range 0.00 ± 0.05. Such values represent a state of no-knowledge by the network, similar to a model with no predictors. The initial weights are used in the first round of training.

Returning to our example, suppose that the initial weights for node 3 are $\theta_3 = -0.3$, $w_{1,3} = 0.05$ and $w_{2,3} = 0.01$ (as shown in Figure 9.3). Using the logistic function we can compute the output of node 3 in the hidden layer (using the first observation) as

$$Output_3 = \frac{1}{1 + e^{-\{-0.3+(0.05)(0.2)+(0.01)(0.9)\}}} = 0.43$$

The diagram in Figure 9.3 shows the initial weights, inputs, and outputs for observation #1 in our tiny example.

If there is more than one hidden layer, the same calculation applies - except that the input values for the the second, third, etc. hidden layers would be the output of the previous hidden layer.

than their sum, but they are much less popular.

This means that the number of input values into a certain node is equal to the number of nodes in the previous layer. (If, in our example there was an additional hidden layer, then its nodes would receive input from the 3 nodes in the first hidden layer.)

Finally, the output layer obtains input values from the (last) hidden layer. It applies the same function as above to create the output. In other words, it takes a weighted average of its input values and then applies the function g. This is the prediction of the model. For classification we use a cutoff value (for a binary response), or the output node with the largest value (for more than 2 classes).

Returning to our example, the single output node (node 6) receives input from the 3 hidden layer nodes. We can compute the prediction (the output of node 6) by

$$Output_6 = \frac{1}{1+e^{-\{-0.015+(0.01)(0.430)+(0.05)(0.507)+(0.015)(0.511)\}}} = 0.506.$$

To classify this record, we use the cutoff of 0.5 and obtain the classification into class "1" (because $0.506 > 0.5$).

The relation with linear and logistic regression

Consider a neural network with a single output node and no hidden layers. For a dataset with p predictors, the output node receives $x_1, x_2, \ldots, x_p$, takes a weighted sum of these and applies the g function. The output of the neural network is therefore $g\left(\theta + \sum_{i=1}^{p} w_i x_i\right)$.

First, consider a numerical output variable y. If g is the identity function ($g(s) = s$), then the output is simply

$$\hat{y} = \theta + \sum_{i=1}^{p} w_i x_i.$$

This is exactly equivalent to the formulation of a multiple linear regression! This means that a neural network with no hidden layers, a single output node, and an identity function g searches only for linear relationships between the response and the predictors.

Now consider a binary output variable y. If g is the logistic function, then the output is simply

$$\hat{P}(Y=1) = \frac{1}{1+e^{\theta+\sum_{i=1}^{p} w_i x_i}},$$

which is equivalent to the logistic regression formulation!

In both cases, although the formulation is equivalent to the linear and logistic regression models, the resulting estimates for the weights ("coefficients" in linear and logistic regression) can differ, because the estimation method is different. The neural net estimation method is different than least squares, the method used to calculate coefficients in linear regression, or the maximum likelihood method used in logistic regression. We explain the method by which the neural network "learns" below.

9.3.3 Preprocessing the Data

Neural networks perform best when the predictors and response variables are on a scale of [0,1]. For this reason all variables should be scaled to a [0,1] scale before entering them into the network.

For a numerical variable X which takes values in the range $[a, b]$ ($a < b$), we normalize the measurements by subtracting a and dividing by $b - a$. The normalized measurement is then

$$X_{norm} = \frac{X-a}{b-a}.$$

Note that if $[a, b]$ is within the [0,1] interval, the original scale will be stretched.

If a and b are not known, we can estimate them from the minimal and maximal values of X in the data. Even if new data exceed this range by a small amount, yielding normalized values slightly lower than 0 or larger than 1, this will not affect the results much.

For binary variables no adjustment needs to be made besides creating dummy variables. For categorical variables with m categories, if they are ordinal in nature, a choice of m fractions in [0,1] should reflect their perceived ordering. For example, if four ordinal categories are equally distant from each other, we can map them to [0, 0.25, 0.5, 1]. If the categories are nominal, transforming into $m-1$ dummies is a good solution.

Another operation that improves performance of the network is to transform highly skewed predictors. In business applications there tend to be many highly right-skewed variables (such as income). Taking a log transform of a right-skewed variable will usually spread out the values more symmetrically.

9.3.4 Training the Model

Training the model means estimating the weights θ_j and w_{ij} that lead to the best predictive results. The process that we described earlier (Section 9.3.2) for computing the neural network output for an observation is repeated for all the observations in the training set. For each observation the model produces a prediction which is then compared with the actual response value. Their difference is the error for the output node. However, unlike least squares or maximum likelihood where a global function of the errors (e.g., sum of squared errors) is used for estimating the coefficients, in neural networks the estimation process uses the errors iteratively to update the estimated weights.

In particular, the error for the output node is distributed across all the hidden nodes that led to it, so that each node is assigned "responsibility" for part of the error. Each of these node-specific errors is then used for updating the weights.

Backpropagation of error

The most popular method for using model errors to update weights ("learning") is an algorithm called *back propagation*. As the name implies, errors are computed from the last layer (the output layer) back to the hidden layers.

Let us denote by $\hat{y}_k$ the output from output node k . The error associated with output node k is computed by

$$Err_k = \hat{y}_k(1-\hat{y}_k)(y_k-\hat{y}_k).$$

Notice that this is similar to the ordinary definition of an error $(y_k-\hat{y}_k)$ multiplied by a correction factor. The weights are then updated as follows:

$$\begin{aligned} \theta_j^{new} &= \theta_j^{old} + lErr_j \\ w_{i,j}^{new} &= w_{i,j}^{old} + lErr_j \end{aligned} \tag{9.2}$$

where l is a *learning rate* or *weight decay* parameter, a constant ranging typically between 0 and 1, which controls the amount of change in weights from one iteration to the other.

In our example, the error associated with the output node for the first observation is $(0.506)(1-0.506)(1-0.506) = 0.123$. This error is then used to compute the errors associated with the hidden layer nodes, and those weights are updated accordingly using a formula similar to (9.2).

Two methods for updating the weights are case updating and batch updating. In case updating the weights are updated after each observation is run through the network (called a trial). For example, if we used case updating in the tiny example, the weights would first be updated after running observation #1 as follows: Using a learning rate of 0.5, the weights θ_6 and $w_{3,6}, w_{4,6}, w_{5,6}$

are updated to

$$\begin{aligned} \theta_6 &= -0.015 + (0.5)(0.123) = 0.047 \\ w_{3,6} &= 0.01 + (0.5)(0.123) = 0.072 \\ w_{4,6} &= 0.05 + (0.5)(0.123) = 0.112 \\ w_{5,6} &= 0.015 + (0.5)(0.123) = 0.077 \end{aligned}$$

These new weights are next updated after the second observation is run through the network, etc. until all observations are used. This is called one "epoch", "sweep" or "iteration" through the data. Typically there are many epochs.

In batch updating, the entire training set is run through the network before each updating of weights takes place. In that case the errors Err_k in the updating equation is the sum of the errors from all observations. Case updating tends to yield more accurate results than batch updating in practice, but it requires a longer runtime. This is a serious consideration, since even in batch updating hundreds or even thousands of sweeps through the training data are executed.

When does the updating stop? The most common conditions are either

1. when the new weights are only incrementally different than those from the last iteration, or
2. when the misclassification rate reached a required threshold, or
3. when the limit on the number of runs is reached.

Let us examine the output from running a neural network on the tiny data. Following the diagram in Figures 9.2 and 9.3 we used a single hidden layer with three nodes. The weights and classification matrix are shown in Figure 9.4. We can see that the network misclassifies all "1" observations and correctly classifies all "0" observations. This is not surprising, since the number of observations is too small for estimating the 13 weights. However, for purposes of illustration we discuss the remainder of the output.

> To run a neural net in XLMiner, choose the $NeuralNetwork$ option either in the $Prediction$ or in the $Classification$ menu, depending on whether your response variable is quantitative or categorical. There are several options that the user can choose, which are described in the software guide. Notice, however, two points: 1. *Normalize input data* applies standard normalization (subtracting the mean and dividing by the standard deviation). Scaling to a [0,1] interval should be done beforehand. 2. XLMiner employs case updating. The user can specify the number of epochs to run.

The final network weights are shown in two tables in the XLMiner output. Figure 9.5 shows these weights in a similar format to our previous diagrams.

The first table shows the weights that connect the input layer and the hidden layer. The *bias nodes* are the weights $\theta_3, \theta_4, \theta_5$. The weights in this table are used to compute the output of the hidden layer nodes. They were computed iteratively after choosing a random initial set of weights (like the set we chose in Figure 9.3). We use the weights in the same way described earlier to compute the hidden layer's output. For instance, for the first observation, the output of our previous node 3 (denoted *Node #1* in XLMiner) is

$$Output_3 = \frac{1}{1 + e^{-\{0.0115 + (-0.11)(0.2) + (-0.08)(0.9)\}}} = 0.48$$

Training Data scoring - Summary Report

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

Classification Confusion Matrix		
	Predicted Class	
Actual Class	1	0
1	0	3
0	0	3

Error Report			
Class	# Cases	# Errors	% Error
1	3	3	100.00
0	3	0	0.00
Overall	6	3	50.00

Inter-layer connections weights

	Input Layer		
Hidden Layer # 1	fat	salt	Bias Node
Node # 1	-0.110424	-0.0800683	0.011531
Node # 2	-0.10581	-0.10347	-0.0447816
Node # 3	-0.0400986	0.128012	-0.0534663

	Hidden Layer # 1			
Output Layer	Node # 1	Node # 2	Node # 3	Bias Node
1	-0.0964131	-0.1029	0.0763853	0.0470577
0	-0.00586047	0.100234	-0.0960382	0.0130296

Row Id.	Predicted Class	Actual Class	Prob. for 1 (success)	fat	salt
1	1	1	0.498658971	0.2	0.9
2	1	0	0.497477278	0.1	0.1
3	1	0	0.497954285	0.2	0.4
4	1	0	0.498202273	0.4	0.5
5	1	1	0.49800783	0.3	0.4
6	1	1	0.498571499	0.3	0.8

Figure 9.4: Output for Neural Network with a Single Hidden Layer with 3 Nodes, for Tiny Data Example

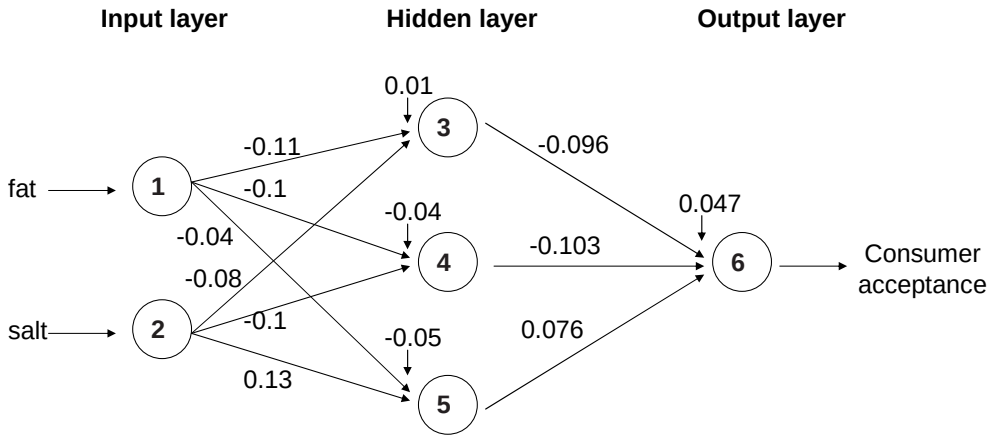


Figure 9.5: Diagram of a Neural Network for The Tiny Example with Weights from XLMiner Output (Figure 9.4)

Similarly we can compute the output from the two other hidden nodes for the same observation and get $Output_4 = 0.46$, and $Output_5 = 0.51$.

The second table gives the weights connecting the hidden and output layer nodes. Notice that XLMiner uses two output nodes even though we have a binary response. This is equivalent to using a single node with a cutoff value. To compute the output from the (first) output node for the first observation, we use the outputs from the hidden layer that we computed above, and get

$$Output_6 = \frac{1}{1 + e^{-\{0.047 + (-0.096)(0.48) + (-0.103)(0.46) + (0.076)(0.51)\}}} = 0.498$$

This is the probability that is shown in the bottom-most table (for observation #1), which gives the neural network's predicted probabilities and the classifications based on these values. The probabilities for the other 5 observations are computed equivalently, replacing the input value in the computation of the hidden layer outputs, and then plugging in these outputs into the computation for the output layer.

9.3.5 Example 2: Classifying Accident Severity

Let's apply the network training process to some real data - U.S. automobile accidents that have been classified by their level of severity as "no injury," "injury," or "fatality." A firm might be interested in developing a system for quickly classifying the severity of an accident, based upon initial reports and associated data in the system (some of which rely on GPS-assisted reporting). Such a system could be used to assign emergency response team priorities. Figure 9.3.5 shows a small extract (999 records, 4 predictor variables) from a US government database. The explanation of the four predictor variables and response is given below:

Row Id.	Selected variables				
	ALCHL_I	PROFIL_I_R	SUR_COND	VEH_INVL	MAX_SEV_IR
1	2	0	1	1	0
4	2	0	2	2	1
5	2	1	1	2	1
6	2	0	1	1	0
9	2	1	1	1	1
10	2	0	1	1	0
12	1	1	1	2	1
17	2	1	1	2	1
18	2	1	4	1	1
19	2	0	1	1	0
20	2	1	1	2	2
21	2	1	1	2	1
23	2	1	3	2	1
26	1	0	1	1	2
27	1	1	1	2	2

Figure 9.6: Subset from the Accident Data

ALCHL_I	presence (1) or absence (2) of alcohol
PROFIL_I_R	profile of the roadway - level (1), grade (2), hillcrest (3), other (4), unknown (5)
SUR_COND	surface condition of the road - dry (1), wet (2), snow/slush (3), ice (4), sand/dirt (5), other (6), unknown (7)
VEH_INVL	the number of vehicles involved
MAX_SEV_IR	presence of injuries/fatalities - no injuries (0), injury (1), fatality (2)

With the exception of alcohol involvement, and a few other variables in the larger database, most of the variables are ones that we might reasonably expect to be available at the time of initial accident report, before accident details and severity have been determined by first responders. A data mining model that could predict accident severity on the basis of these initial reports would have value in allocating first responder resources.

To use a neural net architecture for this classification problem we use 4 nodes in the input layer, one for each of the 4 predictors, and 3 neurons (one for each class) in the output layer. We use a single hidden layer, and experiment with the number of nodes. Increasing the number of nodes from 4 to 8 and examining the resulting confusion matrices, we find that 5 nodes gave a good balance between improving the predictive performance on the training set without deteriorating the performance on the validation set. In fact, networks with more than 5 nodes in the hidden layer performed equally well as the 5-node network.

Note that there is a total of 5 connections from each node in the input layer to each node in the hidden layer, a total of $4 \times 5 = 20$ connections between the input layer and the hidden layer. In addition there is a total of 3 connections from each node in the hidden layer to each node in the

output layer, a total of $5 \times 3 = 15$ connections between the hidden layer and the output layer.

We train the network on the training partition of 600 records. Each iteration in the neural network process consists of presentation to the input layer of the predictors in a case, followed by successive computations of the outputs of the hidden layer nodes and the output layer nodes using the appropriate weights. The output values of the output layer nodes are used to compute the error. This error is used to adjust the weights of all the connections in the network using the backward propagation algorithm to complete the iteration. Since the training data has 600 cases, one sweep through the data, termed an epoch, consists of 600 iterations. We will train the network using 30 epochs, so there will be a total of 18,000 iterations. The resulting classification results, error rates, and weights following the last epoch of training the neural net on this data are shown in Figure 9.7.

Note that had we stopped after only one pass of the data (150 iterations) the error would have been much worse, and none of fatal accidents ("2") would have been spotted as can be seen in Figure 9.8. Our results can depend on how we set the different parameters, and there are a few pitfalls to avoid. We discuss these next.

Avoiding overfitting

A weakness of the neural network is that it can easily overfit the data, causing the error rate on validation data (and most importantly, on new data) to be too large. It is therefore important to limit the number of training epochs and not to overtrain the data. As in classification and regression trees, overfitting can be detected by examining the performance on the validation set and seeing when it starts deteriorating (while the training set performance is still improving). This approach is used in some algorithms (but not in XLMiner) to limit the number of training epochs: periodically the error rate on the validation dataset is computed while the network is being trained. The validation error decreases in the early epochs of the training but after a while it begins to increase. The point of minimum validation error is a good indicator of the best number of epochs for training and the weights at that stage are likely to provide the best error rate in new data.

To illustrate the effect of overfitting, compare the confusion matrices of the 5-node network (Figure 9.7) with those from a 25-node network (Figure 9.9). Both networks perform similarly on the training set, but the 25-node network does worse on the validation set.

9.3.6 Using the Output for Prediction and Classification

When the neural network is used for predicting a numerical response, the resulting output needs to be scaled back to the original units of that response. Recall that numerical variables (both predictor and response variables) are usually rescaled to a [0,1] scale before being used by the network. The output therefore will also be on a [0,1] scale. To transform the prediction back to the original y units, which were in the range $[a, b]$, we multiply the network output by $b - a$ and add a.

When the neural net is used for classification and we have m classes, we will obtain an output from each of the m output nodes. How do we translate these m outputs into a classification rule? Usually, the output node with the largest value determines the net's classification.

In the case of a binary response ($m = 2$), we can use just one output node with a cutoff value to map a numerical output value to one of the two classes. Although we typically use a cutoff of 0.5 with other classifiers, in neural networks there is a tendency for values to cluster around 0.5 (from above and below). An alternative is to use the validation set to determine a cutoff that produces reasonable predictive performance.

Training Data scoring - Summary Report

Classification Confusion Matrix			
	Predicted Class		
Actual Class	0	1	2
0	316	1	12
1	0	173	7
2	20	41	30

Error Report			
Class	# Cases	# Errors	% Error
0	329	13	3.95
1	180	7	3.89
2	91	61	67.03
Overall	600	81	13.50

Validation Data scoring - Summary Report

Classification Confusion Matrix			
	Predicted Class		
Actual Class	0	1	2
0	215	0	7
1	0	114	5
2	17	21	20

Error Report			
Class	# Cases	# Errors	% Error
0	222	7	3.15
1	119	5	4.20
2	58	38	65.52
Overall	399	50	12.53

Inter-layer connections weights

	Input Layer				
Hidden Layer # 1	ALCHL_I	PROFIL_I_R	SUR_COND	VEH_INVL	Bias Node
Node # 1	0.752536	-1.56418	-1.87962	-1.08218	-1.12008
Node # 2	1.00038	-1.59099	-0.545948	-0.818684	-0.835187
Node # 3	0.215221	-1.33309	-2.80806	-0.0157571	-0.646288
Node # 4	0.07074	-2.35661	-3.43757	-2.49196	-1.71293
Node # 5	1.00251	1.52574	4.30335	0.56337	-0.793611

	Hidden Layer # 1					
Output Layer	Node # 1	Node # 2	Node # 3	Node # 4	Node # 5	Bias Node
0	1.50246	1.27632	1.65596	3.12193	-2.7042	-2.87614
1	-1.26542	-0.211778	-2.13848	-2.71368	2.94968	-0.562839
2	-0.873338	-1.21267	0.601146	-2.78924	-2.79458	0.997683

Figure 9.7: XLMiner Output for Neural Network for Accident Data, with 5 Nodes in the Hidden Layer, After 30 Epochs

Training Data scoring - Summary Report

Classification Confusion Matrix			
	Predicted Class		
Actual Class	0	1	2
0	328	1	0
1	17	163	0
2	44	47	0

Error Report			
Class	# Cases	# Errors	% Error
0	329	1	0.30
1	180	17	9.44
2	91	91	100.00
Overall	600	109	18.17

Validation Data scoring - Summary Report

Classification Confusion Matrix			
	Predicted Class		
Actual Class	0	1	2
0	221	1	0
1	16	103	0
2	29	29	0

Error Report			
Class	# Cases	# Errors	% Error
0	222	1	0.45
1	119	16	13.45
2	58	58	100.00
Overall	399	75	18.80

Inter-layer connections weights

	Input Layer				
Hidden Layer # 1	ALCHL_I	PROFIL_I_R	SUR_COND	VEH_INVL	Bias Node
Node # 1	-0.0352883	-0.668327	-0.425221	-0.595552	-0.124272
Node # 2	0.0467488	-0.612836	-0.290081	-0.491289	0.00871788
Node # 3	-0.0386277	-0.660557	-0.621354	-0.683005	-0.0677495
Node # 4	0.0813384	-1.28064	-0.996499	-1.13877	-0.09437
Node # 5	0.206675	0.492427	0.477908	0.501264	0.0936535

	Hidden Layer # 1					
Output Layer	Node # 1	Node # 2	Node # 3	Node # 4	Node # 5	Bias Node
0	0.526532	0.347701	0.593953	1.57821	-0.97179	-0.871603
1	-0.57228	-0.463982	-0.75034	-1.32693	0.711036	0.283258
2	-0.33211	-0.456262	-0.303628	-0.509042	-0.417585	-0.646354

Figure 9.8: XLMiner Output for Neural Network for Accident Data, with 5 Nodes in the Hidden Layer, After Only One Epoch

Training Data scoring - Summary Report

Classification Confusion Matrix			
	Predicted Class		
Actual Class	0	1	2
0	316	1	12
1	0	174	6
2	20	43	28

Error Report			
Class	# Cases	# Errors	% Error
0	329	13	3.95
1	180	6	3.33
2	91	63	69.23
Overall	600	82	13.67

Validation Data scoring - Summary Report

Classification Confusion Matrix			
	Predicted Class		
Actual Class	0	1	2
0	215	0	7
1	0	114	5
2	17	23	18

Error Report			
Class	# Cases	# Errors	% Error
0	222	7	3.15
1	119	5	4.20
2	58	40	68.97
Overall	399	52	13.03

Figure 9.9: XLMiner Output for Neural Network for Accident Data with 25 Nodes in The Hidden Layer

9.4 Required User Input

One of the time consuming and complex aspects of training a model using backpropagation is that we first need to decide on a network architecture. This means specifying the number of hidden layers and the number of nodes in each layer. The usual procedure is to make intelligent guesses using past experience and to do several trial and error runs on different architectures. Algorithms exist that grow the number of nodes selectively during training or trim them in a manner analogous to what is done in classification and regression trees (see Chapter 7). Research continues on such methods. As of now, no automatic method seems clearly superior to the trial and error approach. Below are a few general guidelines for choosing an architecture.

Number of hidden layers: The most popular choice for the number of hidden layers is one. A single hidden layer is usually sufficient to capture even very complex relationships between the predictors.

Size of hidden layer: The number of nodes in the hidden layer also determines the level of complexity of the relationship between the predictors that the network captures. The tradeoff is between under- and over-fitting. Using too few nodes might not be sufficient to capture complex relationships (recall the special cases of a linear relationship such as in linear and logistic regression, in the extreme case of zero nodes, or no hidden layer). On the other hand, too many nodes might lead to overfitting. A rule of thumb is to start with p (= number of predictors) nodes and gradually decrease/increase a bit while checking for overfitting.

Number of output nodes: For a binary response a single node is sufficient, and a cutoff is used for classification. For a categorical response with $m > 2$ classes, the number of nodes should equal the number of classes. Finally, for a numerical response, typically a single output node is used, unless we are interested in predicting more than one function.

In addition to the choice of architecture, the user should pay attention to the **choice of predictors**. Since neural networks are highly dependent on the quality of their input, the choice of predictors should be done carefully, using domain knowledge, variable selection, and dimension reduction techniques before using the network. We return to this point in the discussion of advantages and weaknesses below.

Other parameters that the user can control are the *learning rate* (aka weight decay), l, and the *momentum* . The first is used primarily to avoid overfitting, by down-weighting new information. This helps to tone down the effect of outliers on the weights, and avoids getting stuck in local optima. This parameter typically takes a value in the range $[0, 1]$. Berry & Linoff (2000) suggest starting with a large value (moving away from the random initial weights, thereby "learning quickly" from the data) and then slowly decreasing it as the iterations progress and the weights are more reliable. Han & Kamber (2001) suggest the more concrete rule of thumb of setting l=1/(current # iterations). This means that at the start $l = 1$, during the second iteration it is $l = 0.5$, and then it keeps decaying towards $l = 0$. Notice that in XLMiner the default is $l = 0$, which means that the weights do not decay at all.

The second parameter, called momentum, is used to "keep the ball rolling" (hence the term momentum) in the convergence of the weights to the optimum. The idea is to keep the weights changing in the same direction that they did in the previous iteration. This helps avoiding getting stuck in some local optimum. High values of momentum mean that the network will be "reluctant" to learn from data that want to change the direction of the weights, especially when we consider case updating. In general, values in the range 0 to 2 are used.

9.5 Exploring the Relationship Between Predictors and Response

Neural networks are known to be "black boxes" in the sense that their output does not shed light on the patterns in the data that it models (like our brains). In fact, that is one of the biggest criticism of the method. However, in some cases it is possible to learn more about the relationships that the network captures, by conducting a sensitivity analysis on validation set. This is done by setting all predictor values to their mean and obtaining the network's prediction. Then, the process is repeated by setting each predictor sequentially to its minimum (and then maximum) value. By comparing the predictions from different levels of the predictors we can get a sense of which predictors affect predictions more and in what way.

9.6 Advantages and Weaknesses of Neural Networks

As mentioned in the introduction, the most prominent advantage of neural networks is their good predictive performance. They are known to have high tolerance to noisy data, and be able to capture highly complicated relationships between the predictors and a response.

Their weakest point is in providing insight into the structure of the relationship, and hence their "black-box" reputation.

Several considerations and dangers should be kept in mind when using neural networks. First, although they are capable of generalizing from a set of examples, extrapolation is still a serious danger. If the network sees only cases in a certain range, then its predictions outside this range can be completely invalid.

Second, neural networks do not have a built-in variable selection mechanism. This means that there is need for careful consideration of predictors. Combination with classification and regression trees (see Chapter 7) and other dimension reduction techniques (e.g., principal components analysis in Chapter 3) is often used to identify key predictors.

Third, the extreme flexibility of the neural network relies heavily on having sufficient data for training purposes. As our tiny example shows, a neural network performs poorly when the training set size is insufficient, even with the relationship between the response and predictors is very simple. A related issue is that in classification problems, the network requires sufficient records of the minority class in order to learn it. This is achieved by oversampling, as explained in Chapter 2.

Fourth, a technical problem is the risk of obtaining weights that lead to a local optimum rather than the global optimum, in the sense that the weights converge to values that do not provide the best fit to the training data. We described several parameters that are used to try and avoid this situation (such as controlling the learning rate and slowly reducing the momentum). However, there is no guarantee that the resulting weights are indeed the optimal ones.

Finally, a practical consideration that can determine the usefulness of a neural network is the timeliness of computation. Neural networks are relatively heavy on computation time, requiring longer runtime than other classifiers. This run time grows greatly when the number of predictors is increased (as there will be many more weights to compute). In applications where real-time or near-real-time prediction is required, run time should be measured to make sure it does not cause unacceptable delay in the decision making.

9.7 Exercises

Credit card usage: Consider the following hypothetical bank data on consumers' use of credit card credit facilities:

Years	Salary	Used credit
4	43	0
18	65	1
1	53	0
3	95	0
15	88	1
6	112	1

Years	Number of years that the customer has been with the bank
Salary	Customer's salary (in thousands of $)
Used credit	"1" = customer has left an unpaid credit card balance at the end of at least one month in the prior year, "0" = the balance was paid off at the end of each month.

Create a small worksheet in Excel, like that used in Example 1, to illustrate one pass through a simple neural network.

Neural net evolution: A neural net typically starts out with random coefficients, and, hence, produces essentially random predictions when presented with its first case. What is the key ingredient by which the net evolves to produce a more accurate prediction?

Car sales: Consider again the data on used cars (ToyotaCorolla.xls) with 1436 records and details on 38 attributes including price, age, kilometers, horsepower, and other specifications. The goal is to predict the price of a used Toyota Corolla based on its specifications.

1. Use XLMiner's neural network routine to fit a model using the XLMiner default values for the neural net parameters, except normalizing the data. Record the RMS error for the training data and the validation data. Repeat the process, changing the number of epochs (and only this) to 300, 3000, and 10,000.
   (a) What happens to the RMS error for the training data as the number of epochs increases?
   (b) For the validation data?
   (c) Comment on the appropriate number of epochs for the model.
2. Conduct a similar experiment to assess the effect of changing the number of layers in the network, as well as the gradient descent step size.

Direct mailing to airline customers: East-West Airlines has entered into a partnership with the wireless phone company Telcon to sell the latter's service via direct mail. The file East-WestAirlinesNN.xls contains a subset of a data sample who has already received a test offer. About 13% accepted.

You are asked to develop a model to classify East-West customers as to whether they purchased a wireless phone service contract (target variable $Phone_sale$), a model that can be used to predict classifications for additional customers.

1. Using XLMiner, run a neural net model on these data, using the option to normalize the data, setting the number of epochs at 3000, and requesting lift charts for both the training and validation data. Interpret the meaning (in business terms) of the left most bar of the validation lift chart (the bar chart).
2. Comment on the difference between the training and validation lift charts.

3. Run a second neural net model on the data, this time setting the number of epochs at 100. Comment now on the difference between this model and the model you ran earlier, and how overfitting might have affected results.
4. What sort of information, if any, is provided about the effects of the various variables?

Chapter 10

Discriminant Analysis

10.1 Introduction

Discriminant analysis is another classification method. Like logistic regression, it is a classical statistical technique that can be used for classification and profiling. It uses continuous variable measurements on different classes of items to classify new items into one of those classes ("classification"). Common uses of the method have been in classifying organisms into species and sub-species, classifying applications for loans, credit cards and insurance into low risk and high risk categories, classifying customers of new products into early adopters, early majority, late majority and laggards, classifying bonds into bond rating categories, classifying skulls of human fossils, as well as in research studies involving disputed authorship, decision on college admission, medical studies involving alcoholics and non-alcoholics, and methods to identify human fingerprints. Discriminant analysis can also be used to highlight aspects that distinguish the classes ("profiling").

We return to two examples that were described in previous chapters, and use them to illustrate discriminant analysis.

10.2 Example 1: Riding Mowers

We return to the example from Chapter 6, where a riding-mower manufacturer would like to find a way of classifying families in a city into those likely to purchase a riding mower and those not likely to buy one. A pilot random sample of 12 owners and 12 non-owners in the city is undertaken. The data are given in Chapter 6 (Table 6.2) and a scatterplot is shown in Figure 10.1 below. We can think of a linear classification rule as a line that separates the two-dimensional region into two parts where most of the owners are in one half-plane and most of the non-owners are in the complementary half-plane. A good classification rule would separate the data so that the fewest points are misclassified: the line shown in Figure 10.1 seems to do a good job in discriminating between the two classes as it makes 4 misclassifications out of 24 points. Can we do better?

10.3 Example 2: Personal Loan Acceptance

The riding mowers example is a classic example and is useful in describing the concept and goal of discriminant analysis. However, in today's business applications, the number of records is much larger and their separation into classes is much less distinct. To illustrate this we return to the Universal Bank example described in Chapter 7, where the bank's goal is to find which factors make a customer more likely to accept a personal loan. For simplicity, we consider only two variables: the customer's annual income (*Income*, in $000), and the average monthly credit-cards spending

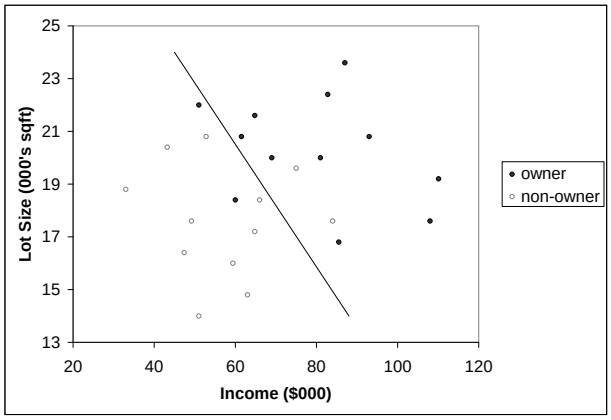


Figure 10.1: Scatterplot of Lot Size Vs. Income for 24 Owners and Non-Owners of Riding Mowers. The (Ad-Hoc) Line Tries to Separate Owners from Non-Owners

($CCAvg$, in \$000). The first part of Figure 10.2 shows the acceptance of a personal loan by a subset of 200 customers from the bank's database as a function of $Income$ and $CCAvg$). We use log-scale on both axes to enhance visibility, because there are many points condensed in the low income, low CC spending area. Even for this small subset the separation is not clear. The second figure shows all 5,000 customers and the added complexity from dealing with large numbers of observations.

10.4 Distance of an Observation from a Class

Finding the best separation between items involves measuring their distance from their class. The general idea is to classify an item to the class that it is closest to. Suppose we are required to classify a new customer of Universal Bank as being an acceptor/non-acceptor of their personal loan offer, based on an income of x. From the bank's database we find that the average income for loan acceptors was \$144.75K and for non-acceptors \$66.24K. We can use $Income$ as a predictor of loan acceptance via a simple *Euclidean distance* Rule: If x is closer to the average income of the acceptor class than to the average income of the non-acceptor class, then classify the customer as an acceptor; otherwise, classify the customer as a non-acceptor. In other words, if $|x - 144.75| < |x - 66.24|$, then classification=acceptor; otherwise, non-acceptor. Moving from a single variable (income) to two or more variables, the equivalent of the mean of a class is the *centroid* of a class. This is simply the vector of means $\overline{\mathbf{x}} = [\overline{x}_1, \ldots, \overline{x}_p]$. The Euclidean distance between an item with p measurements $\mathbf{x} = [x_1, \ldots, x_p]$ and the centroid $\overline{\mathbf{x}}$ is defined as the root of the sum of the squared differences between the individual values and the means:

$$D_{\mathrm{E}uclidean}(\mathbf{x}, \overline{\mathbf{x}}) = \sqrt{(x_1 - \overline{x}_1)^2 + \ldots + (x_p - \overline{x}_1)^p}. \tag{10.1}$$

Using the Euclidean distance has two drawbacks. First, the distance depends on the units we choose to measure the variables. We will get different answers if we decide to measure income, for instance, in \$ rather than thousands of \$. Second, Euclidean distance does not take into account the variability of the different variables. For example, if we compare the variability in income in the two classes, we find that for acceptors the standard deviation is lower than for non-acceptors! (\$31.6K vs. \$40.6K). Therefore the income of a new customer might be closer to the acceptors' average

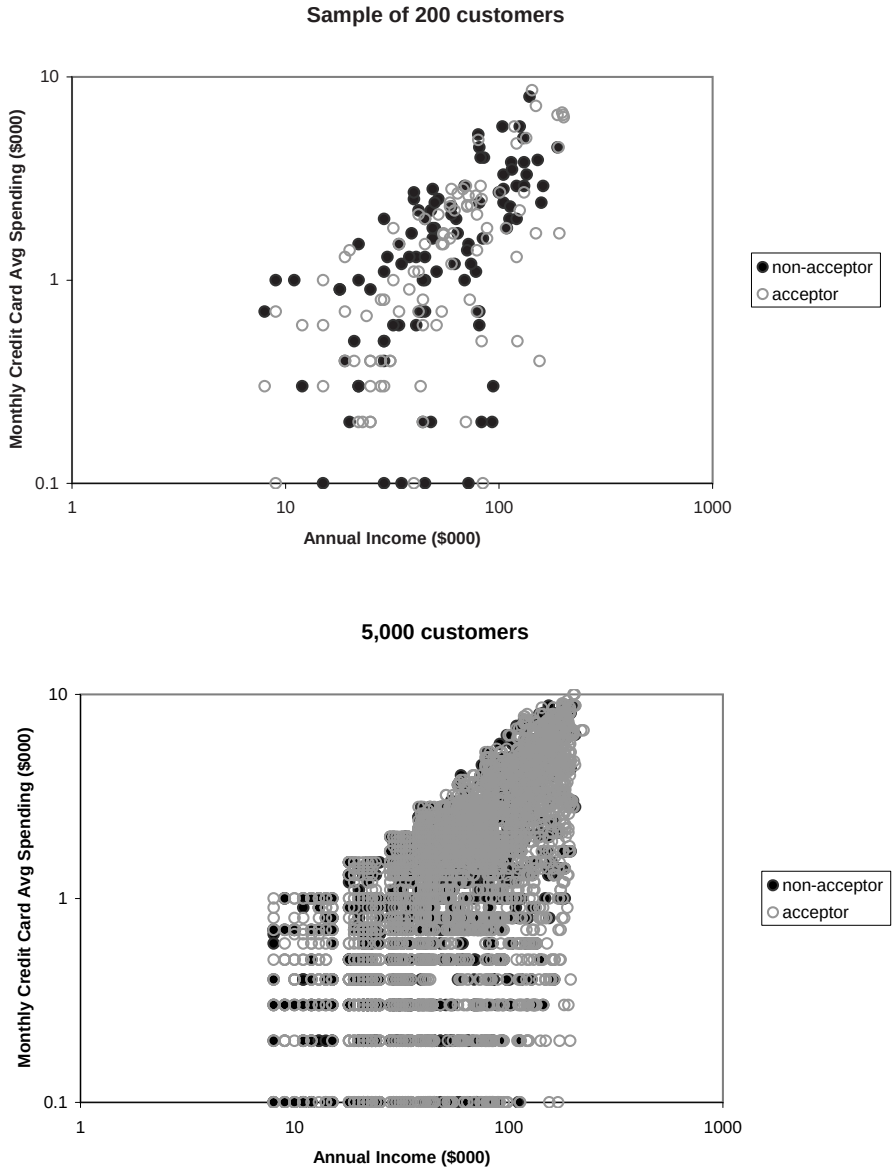


Figure 10.2: Personal Loan Acceptance as a Function of Income and Credit Card Spending For 5000 Customers of Universal Bank (in Log Scale)

income in $ figures, but because of the large variability in income for non-acceptors, this customer is just as likely to be a non-acceptor. We therefore want the distance measure to take into account the variance of the different variables, and measure a distance in "standard deviations" rather than in the original units. This is equivalent to "z-scores".

Third, Euclidean distance ignores the correlation between the variables. This is often a very important consideration, especially when we are using many variables to separate classes. In this case there will often be variables which, by themselves, are useful discriminators between classes but in the presence of other variables, are practically redundant as they capture the same effects as the other variables.

A solution to these drawbacks is to use a measure called **statistical distance** (or, Mahalanobis distance). Let us denote by **S** the covariance matrix between the p variables. The definition of a statistical distance is:

$$D_{\mathrm{S}statistical}(\mathbf{x}, \overline{\mathbf{x}}) = [\mathbf{x} - \overline{\mathbf{x}}]' S^{-1} [\mathbf{x} - \overline{\mathbf{x}}] \tag{10.2}$$

$$= [(x_1 - \overline{x}_1), (x_2 - \overline{x}_2), \ldots, (x_p - \overline{x}_p)] S^{-1} \begin{bmatrix} x_1 - \overline{x}_1 \\ x_2 - \overline{x}_2 \\ \cdots \\ x_p - \overline{x}_p \end{bmatrix} \tag{10.3}$$

($'$ is called a transpose operation, and simply turns the column vector into a row vector). S^{-1} is the inverse matrix of S, which is the p-dimension extension to division. When there is a single predictor ($p = 1$), this reduces to a "z-score", since we subtract the mean and divide by the standard deviation. The statistical distance takes into account not only the predictor averages, but also the spread of the predictor values and the correlations between the different predictors. To compute a statistical distance between an observation and a class we must compute the predictor averages (the centroid) and the covariances between each pair of variables. These are used to construct the distances. The method of discriminant analysis uses statistical distance as the basis for finding a separating line (or, if there are more than two variables, a separating hyper-plane) that is equally distant from the different class means[1]. It is based on measuring the statistical distances of an observation to each of the classes, and allocating it to the closest class. This is done through *classification functions*, which are explained next.

10.5 Fisher's Linear Classification Functions

Linear classification functions were suggested by the noted statistician R. A. Fisher in 1936 as the basis for improved separation of observations into classes. The idea is to find linear functions of the measurements that maximize the ratio of between-class variability to within-class variability. In other words, we would obtain classes that are very homogeneous and differ the most from each other. For each observation, these functions are used to compute scores that measure the proximity of that observation to each of the classes. An observation is classified as belonging to the class for which it has the highest classification score (equivalent to the smallest statistical distance).

Using Classification Function "Scores" to Classify

For each record, we calculate the value of the classification function (one for each class); whichever class's function has the highest value (=score) is the class assigned to that record.

[1] An alternative approach finds a separating line or hyperplane that is "best" at separating the different clouds of points. In the case of two classes the two methods coincide.

Classification Function

Variables	Classification Function	
	owner	non-owner
Constant	-73.16020203	-51.42144394
Income	0.42958561	0.32935533
Lot Size	5.46674967	4.68156528

Figure 10.3: Discriminant Analysis Output for Riding Mower Data, Displaying the Estimated Classification Functions

The classification functions are estimated using software (see Figure 10.5). Note that the number of classification functions is equal to the number of classes (in this case 2).

To classify a family into the class of owners or non-owners, we use the above functions to compute the family's classification scores: A family is classified into the class of "owners" if the "owners" function is higher than the "non-owner" function, and into "non-owners" if the reverse is the case. These functions are specified in a way that can be easily generalized to more than two classes. The values given for the functions are simply the weights to be associated with each variable in the linear function in a manner analogous to multiple linear regression. For instance, the first household has an income of \$60K and lot size of 18.4K sqft. Their "owner" score is therefore -73.16 + (0.43)(60)+(5.47)(18.4)=53.2, and their "non-owner" score is -51.42+(0.33)(60)+(4.68)(18.4)=54.48. Since the second score is higher, the household is (mis)classified by the model as a non-owner. The scores for all 24 households are given in Figure 10.5.

An alternative way for classifying an observation into one of the classes is to compute the probability of belonging to each of the classes and assigning the observation to the most likely class. If we have two classes, we need only compute a single probability for each observation (of belonging to "owners", for example). Using a cutoff of 0.5 is equivalent to assigning the observation to the class with the highest classification score. The advantage of this approach is that we can sort the records in order of descending probabilities and generate lift curves. Let us assume that there are m classes. To compute the probability of belonging to a certain class k, for a certain observation i, we need to compute all the classification scores $c_1(i), c_2(i), \ldots, c_g(i)$ and combine them using the following formula:

$$\text{P \{ observation i (with measurements } x_1, x_2, ..., x_p) \text{ belongs to class k \} } = \frac{e^{c_k(i)}}{e^{c_1(i)} + e^{c_2(i)} + \ldots + e^{c_g(i)}}$$

In XLMiner these probabilities are automatically computed, as can be seen in Figure 10.5.

We now have 3 misclassifications, compared to 4 in our original (ad-hoc) classifications. This can be seen in Figure 10.6 which includes the line resulting from the discriminant model.[2]

[2]The slope of the line is given by $-\frac{a_1}{a_2}$ and the intercept is $\frac{a_1}{a_2}\overline{x}_1 + \overline{x}_2$, where a_i is the difference between the ith classification function coefficients of owners and non-owners (e.g. here $a_{income} = 0.43 - 0.33$).

				classification scores	
Row Id.	Actual Class	Income	Lot Size	owners	non-owners
1	owner	60	18.4	53.2031285	54.48067701
2	owner	85.5	16.8	55.41076208	55.38873348
3	owner	64.8	21.6	72.75873837	71.04259149
4	owner	61.5	20.8	66.96770612	66.21046668
5	owner	87	23.6	93.22903825	87.71741038
6	owner	110.1	19.2	79.0987673	74.72663127
7	owner	108	17.6	69.44983804	66.54448063
8	owner	82.8	22.4	84.86467909	80.71623966
9	owner	69	20	65.81619846	64.93537943
10	owner	93	20.8	80.49965284	76.58515957
11	owner	51	22	69.01715682	68.37011405
12	owner	81	20	70.97122578	68.88764339
13	non-owner	75	19.6	66.20701225	65.0388853
14	non-owner	52.8	20.8	63.23031131	63.34507531
15	non-owner	64.8	17.2	48.70503982	50.44370426
16	non-owner	43.2	20.4	56.91958959	58.31063803
17	non-owner	84	17.6	59.1397834	58.63995271
18	non-owner	49.2	17.6	44.19020417	47.17838722
19	non-owner	59.4	16	39.82517792	43.04730714
20	non-owner	66	18.4	55.78064216	56.45680899
21	non-owner	47.4	16.4	36.85685047	40.96766929
22	non-owner	33	18.8	43.7910169	47.46070921
23	non-owner	51	14	25.28315946	30.91759181
24	non-owner	63	14.8	34.81158652	38.61510799

Figure 10.4: Classification Scores for Riding Mower Data

Cut off Prob.Val. for Success (Updatable)	0.5	(Updating the value here will NOT update value in s

Row Id.	Predicted Class	Actual Class	Prob. for - owner (success)	Income	Lot Size
1	non-owner	owner	0.21796781	60	18.4
2	owner	owner	0.505506928	85.5	16.8
3	owner	owner	0.847631864	64.8	21.6
4	owner	owner	0.680754087	61.5	20.8
5	owner	owner	0.995976726	87	23.6
6	owner	owner	0.987533139	110.1	19.2
7	owner	owner	0.948110638	108	17.6
8	owner	owner	0.984456382	82.8	22.4
9	owner	owner	0.706991915	69	20
10	owner	owner	0.980439587	93	20.8
11	owner	owner	0.656343749	51	22
12	owner	owner	0.889297203	81	20
13	owner	non-owner	0.762806287	75	19.6
14	non-owner	non-owner	0.47134045	52.8	20.8
15	non-owner	non-owner	0.149482655	64.8	17.2
16	non-owner	non-owner	0.199240432	43.2	20.4
17	owner	non-owner	0.622419543	84	17.6
18	non-owner	non-owner	0.047962588	49.2	17.6
19	non-owner	non-owner	0.038341401	59.4	16
20	non-owner	non-owner	0.337117362	66	18.4
21	non-owner	non-owner	0.016129906	47.4	16.4
22	non-owner	non-owner	0.024850999	33	18.8
23	non-owner	non-owner	0.003559986	51	14
24	non-owner	non-owner	0.021806029	63	14.8

Figure 10.5: Discriminant Analysis Output for Riding Mower Data, Displaying the Estimated Probability of Ownership for Each Family

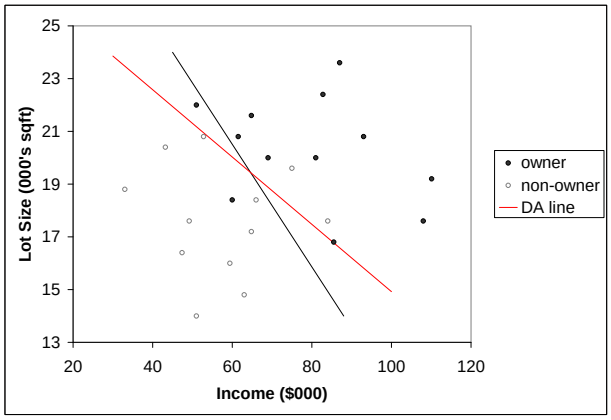


Figure 10.6: The Class Separation Obtained from The Discriminant Model (Compared to Ad-Hoc Line from Figure 10.1)

10.6 Classification Performance of Discriminant Analysis

The discriminant analysis method relies on two main assumptions to arrive at the classification scores: First, it assumes that the measurements in all classes come from a multivariate normal distribution. When this assumption is reasonably met, discriminant analysis is a more powerful tool than other classification methods such as logistic regression. In fact, it is 30% more efficient than logistic regression if the data are multivariate normal, in the sense that we require 30% less data to arrive at the same results. In practice, it has been shown that this method is relatively robust to departures from normality in the sense that predictors can be non-normal and even dummy variables. This is true as long as the smallest class is sufficiently large (approximately more than 20 cases). This method is also known to be sensitive to outliers both in the univariate space of single predictors and also in the multivariate space. Exploratory analysis should therefore be used to locate extreme cases and determine whether they can be eliminated.

The second assumption behind discriminant analysis is that the correlation structure between the different measurements within a class is the same across classes. This can be roughly checked by estimating the correlation matrix for each class and comparing matrices. If the correlations differ substantially across classes, then the classifier will tend to classify cases into the class with largest variability. When the correlation structure differs significantly and the dataset is very large, then an alternative is to use "quadratic discriminant analysis."[3]

With respect to the evaluation of classification accuracy we once again use the general measures of performance that were described in Chapter 4 (judging the performance of a classifier), with the main ones being based on the confusion matrix (accuracy alone or combined with costs) and the lift chart. The same argument for using the validation set for evaluating performance still holds. For example, in the riding mowers example, families 1, 13 and 17 are misclassified. This means that the model yields an error rate of 12.5% for these data. However, this rate is a biased estimate - it is overly optimistic, because we have used the same data for fitting the classification parameters and

[3]In practice, quadratic discriminant analysis has not been found useful except when the difference in the correlation matrices is large and the number of observations available for training and testing is large. The reason is that the quadratic model requires estimating many more parameters that are all subject to error (for c classes and p variables, the total number of parameters to be estimated for all the different correlation matrices is $cp(p+1)/2$).

for estimating the error. Therefore, as with all other models, we test performance on a validation set that includes data that were not involved in estimating the classification functions.

To obtain the confusion matrix from a discriminant analysis, we use either the classification scores directly or else the probabilities of class membership that are computed from the classification scores. In both cases we decide on the class assignment of each observation based on the highest score or probability. We then compare these classifications to the actual class memberships of these observations. This yields the confusion matrix. In the Universal Bank case (exercises 1-6) we use the estimated classification functions in Figure 10.5 to predict the probability of loan acceptance in a validation set that contains 2000 customers (these data were not used in the modeling step).

10.7 Prior Probabilities

So far we have assumed that our objective is to minimize the classification error. The method presented above assumes that the chances of encountering an item from either class requiring classification is the same. If the probability of encountering an item for classification in the future is not equal for the different classes, we should modify our functions to reduce our expected (long run average) error rate. The modification is done as follows: Let us denote by p_j the prior/future probability of membership in class j (in the two-class case we have p_1 and $p_2 = 1 - p_1$). We modify the classification function for each class by adding $\log(p_j)$[4]. To illustrate this, suppose that the percentage of riding mower owners in the population is 15% (compared to 50% in the sample). This means that the model should classify less households as owners. To account for this we adjust the constants in the classification functions from Figure 10.5 and obtain the adjusted constants $-73.16 + \log(0.15) = -75.06$ for owners and $-51.42 + \log(0.85) = -50.58$ for non-owners. To see how this can affect classifications, consider family 13 that was misclassified as an owner in the equal probability of class membership case. When we account for the lower probability of owning a mower in the population, family 13 is properly classified as a non-owner (its owner classification score exceeds the non-owner score.)

10.8 Unequal Misclassification Costs

A second practical modification is needed when misclassification costs are not symmetrical. If the cost of misclassifying a class 1 item is very different from the cost of misclassifying a class 2 item, we may want to minimize the expected cost of misclassification rather than the simple error rate (which does not take cognizance of unequal misclassification costs.) In the two-class case it is easy to manipulate the classification functions to account for differing misclassification costs (in addition to prior probabilities). We denote by C_1 the cost of misclassifying a class 1 member (into class 2). Similarly, C_2 denotes the cost of misclassifying a class 2 member (into class 1). These costs are integrated into the constants of the classification functions by adding $\log(C_1)$ to the constant for class 1 and $\log(C_2)$ to the constant of class 2. To incorporate both prior probabilities and misclassification costs, add $\log(p_1 C_1)$ to the constant of class 1 and $\log(p_2 C_2)$ to that of class 2.

In practice, it is not always simple to come up with misclassification costs C_1, C_2 for each of the classes. It is usually much easier to estimate the ratio of costs C_2/C_1 (e.g., the cost of misclassifying a credit defaulter is 10 times more expensive than that of misclassifying a non-defaulter). Luckily, the relationship between the classification functions depends only on this ratio. Therefore we can set $C_1 = 1$ and $C_2 = ratio$ and simply add $\log(C_2/C_1)$ to the constant for class 2.

[4]XLMiner also has the option to set the prior probabilities as the ratios that are encountered in the dataset. This is based on the assumption that a random sample will yield a reasonable estimate of membership probabilities. However, for other prior probabilities the classification functions should be modified manually.

Accident #	RushHour	WRK_ZONE	WKDY	INT_HWY	LGTCON	LEVEL	SPD_LIM	SUR_COND	TRAF_WAY	WEATHER	MAX_SEV
1	1	0	1	1	dark_light	1	70	ice	one_way	adverse	no-injury
2	1	0	1	0	dark_light	0	70	ice	divided	adverse	no-injury
3	1	0	1	0	dark_light	0	65	ice	divided	adverse	non-fatal
4	1	0	1	0	dark_light	0	55	ice	two_way	not_adverse	non-fatal
5	1	0	0	0	dark_light	0	35	snow	one_way	adverse	no-injury
6	1	0	1	0	dark_light	1	35	wet	divided	adverse	no-injury
7	0	0	1	1	dark_light	1	70	wet	divided	adverse	non-fatal
8	0	0	1	0	dark_light	1	35	wet	two_way	adverse	no-injury
9	1	0	1	0	dark_light	0	25	wet	one_way	adverse	non-fatal
10	1	0	1	0	dark_light	0	35	wet	divided	adverse	non-fatal
11	1	0	1	0	dark_light	0	30	wet	divided	adverse	non-fatal
12	1	0	1	0	dark_light	0	60	wet	divided	not_adverse	no-injury
13	1	0	1	0	dark_light	0	40	wet	two_way	not_adverse	no-injury
14	0	0	1	0	day	1	65	dry	two_way	not_adverse	fatal
15	1	0	0	0	day	0	55	dry	two_way	not_adverse	fatal
16	1	0	1	0	day	0	55	dry	two_way	not_adverse	non-fatal
17	1	0	0	0	day	0	55	dry	two_way	not_adverse	non-fatal
18	0	0	1	0	dark	0	55	ice	two_way	not_adverse	no-injury
19	0	0	0	0	dark	0	50	ice	two_way	adverse	no-injury
20	0	0	0	0	dark	1	55	snow	divided	adverse	no-injury

Figure 10.7: Sample of 20 Automobile Accidents from the 2001 Dept of Transportation Database. Each Accident is Classified as One of Three Injury Types (No-Injury, Non-Fatal, or Fatal), and Has 10 Measurements (Extracted from a Larger Set of Measurements)

10.9 Classifying More Than Two Classes

10.9.1 Example 3: Medical Dispatch to Accident Scenes

Ideally, every automobile accident call to 911 results in an immediate dispatch of an ambulance to the accident scene. However, in some cases the dispatch might be delayed (e.g., at peak accident hours, or in some resource-strapped towns or shifts). In such cases, the 911 dispatchers must make decisions about which units to send based on sketchy information. It is useful to augment the limited information provided in the initial call with additional information in order to classify the accident as minor injury, serious injury, or death. For this purpose, we can use data that were collected on automobile accidents that involved some type of injury in 2001 in the US. For each accident, additional information is recorded such as day of week, weather conditions, and road type. Figure 10.7 shows a small sample of records with 10 measurements of interest.

The goal is to see how well the predictors can be used to correctly classify injury type. To evaluate this, a sample of 1000 records was drawn and partitioned into training and validation sets, and a discriminant analysis was performed on the training data. The output structure is very similar to the two-class case. The only difference is that each observation now has three classification functions (one for each injury type), and the confusion and error matrices are 3×3 to account for all the combinations of correct and incorrect classifications (see Figure 10.8). The rule for classification is still to classify an observation to the class that has the highest corresponding classification score. The classification scores are computed, as before, using the classification function coefficients. This can be seen in Figure 10.5. For instance, the "no-injury" classification score for the first accident in the training set is $-24.51 + 1.95(1) + 1.19(0) + \cdots + 16.36(1) = 30.93$. The "non-fatal" score is similarly computed as 31.42 and the "fatal" score as 25.94. Since the "non-fatal" score is highest, this accident is (correctly) classified as having non-fatal injuries.

We can also compute for each accident the estimated probabilities of belonging to each of the three classes using the same relationship between classification scores and probabilities as in the two-class case. For instance, the probability of the above accident involving non-fatal injuries is

Classification Function

Variables	Classification Function		
	fatal	no-injury	non-fatal
Constant	-25.59584999	-24.51432228	-24.2336216
RushHour	0.92256236	1.95240343	1.9031992
WRK_ZONE	0.51786095	1.19506037	0.77056831
WKDY	4.78014898	6.41763353	6.11652184
INT_HWY	-1.84187829	-2.67303801	-2.53662229
LGTCON_day	3.70701218	3.66607523	3.7276206
LEVEL	2.62689376	1.56755066	1.71386576
SPD_LIM	0.50513172	0.46147966	0.45208475
SUR_COND_dry	9.99886131	15.8337965	16.25656509
TRAF_WAY_two_way	7.10797691	6.34214783	6.35494375
WEATHER_adverse	9.68802357	16.36388016	16.31727791

Training Data scoring - Summary Report

Classification Confusion Matrix			
	Predicted Class		
Actual Class	fatal	no-injury	non-fatal
fatal	1	1	3
no-injury	6	114	172
non-fatal	6	95	202

Error Report			
Class	# Cases	# Errors	% Error
fatal	5	4	80.00
no-injury	292	178	60.96
non-fatal	303	101	33.33
Overall	600	283	47.17

Figure 10.8: XLMiner's Discriminant Analysis Output for the Three-Class Injury Example: Classification Functions and Confusion Matrix for Training Set

Predicted Class	Actual Class	Score for fatal	Score for non-fatal	Score for no-injury	Prob. for class fatal	Prob. for class no-injury	Prob. for class non-fatal	RushHour	WRK_ZONE	WKDY	INT_HWY	LGTCON_day	LEVEL	SPD_LIM	SUR_COND_dry	TRAF_two_way	WEATHER_adverse
no-injury	no-injury	25.94	31.42	30.93	0.002583566	0.618769909	0.378646525	1	0	1	1	0	1	70	0	0	1
no-injury	non-fatal	15.00	15.58	15.01	0.263257586	0.471205318	0.265537095	1	0	1	0	0	0	55	0	1	0
no-injury	no-injury	2.69	9.95	9.81	0.000376892	0.535717942	0.463905165	1	0	0	0	0	0	35	0	0	1
no-injury	no-injury	10.10	17.94	17.64	0.000226522	0.574000564	0.425772914	1	0	1	0	0	1	35	0	0	1
no-injury	non-fatal	2.42	11.76	11.41	5.18896E-05	0.586851481	0.413096629	1	0	1	0	0	0	25	0	0	1
no-injury	non-fatal	7.47	16.37	15.93	8.33756E-05	0.609408333	0.390508291	1	0	1	0	0	0	35	0	0	1
no-injury	no-injury	10.41	11.54	10.91	0.174287408	0.539388146	0.286324446	1	0	1	0	0	0	60	0	0	0
non-fatal	non-fatal	22.57	28.37	28.46	0.001452278	0.476947897	0.521599825	1	0	1	0	0	1	45	1	1	0
non-fatal	no-injury	15.12	20.45	20.47	0.002408934	0.493295725	0.504295341	0	0	1	1	0	0	55	1	0	0
no-injury	non-fatal	23.62	29.32	29.15	0.00181542	0.540936208	0.457248372	1	0	1	1	0	0	70	1	0	0
no-injury	no-injury	20.17	25.06	24.99	0.003898532	0.515946916	0.480154552	0	0	1	1	0	0	65	1	0	0
no-injury	non-fatal	10.31	18.15	18.13	0.00019948	0.505559651	0.494240868	1	0	1	0	0	0	40	1	0	0
no-injury	non-fatal	12.84	20.46	20.39	0.000253881	0.517266017	0.482480102	1	0	1	0	0	0	45	1	0	0
non-fatal	non-fatal	15.16	20.38	20.62	0.002378591	0.43818467	0.559436739	1	0	0	0	0	0	45	1	1	0
no-injury	no-injury	19.94	26.80	26.74	0.000542513	0.513922638	0.485534849	1	0	1	0	0	0	45	1	1	0
non-fatal	non-fatal	12.37	19.88	19.96	0.000262636	0.478861093	0.520876272	1	0	1	0	0	0	30	1	1	0
no-injury	non-fatal	25.00	31.41	31.26	0.000877376	0.537149759	0.461972865	1	0	1	0	0	0	55	1	1	0

Figure 10.9: Classification Scores, Membership Probabilities, and Classifications for Three-Class Injury Training Dataset

estimated by the model as

$$\frac{e^{31.42}}{e^{31.42} + e^{30.93} + e^{25.94}} = 0.38. \tag{10.4}$$

The probabilities of an accident involving no injuries or fatal injuries are computed in a similar manner. For this accident, the highest probability is that of involving no-injuries, and therefore it is classified as a "no-injury" accident. In general, membership probabilities can be obtained directly from XLMiner for the training set, the validation set, or for new observations.

10.10 Advantages and Weaknesses

Discriminant analysis tends to be considered more of a statistical classification method than a data mining one. This is reflected in its absence or short mention in many data mining resources. However, it very popular in social sciences, and has shown good performance.

The use and performance of discriminant analysis are similar to those of multiple linear regression. The two methods therefore share several advantages and weaknesses.

Like linear regression, discriminant analysis searches for the optimal weighting of predictors. In linear regression the weighting is with relation to the response, whereas in discriminant analysis it is with relation to separating the classes. Both use the same estimation method of least squares, and the resulting estimates are robust to local optima.

In both methods an underlying assumption is normality. In discriminant analysis we assume that the predictors are approximately from a multivariate normal distribution. Although this assumption is in many practical violated (such as with commonly used binary predictors), the method is surprisingly robust. According to Hastie et al. (2001), the reason might be that data can usually only support simple separation boundaries, such as linear boundaries. However, for continuous variables that are found to be very skewed (e.g., through a histogram), transformations such as the log-transform can improve performance. In addition, the method's sensitivity to outliers commands exploring the data for extreme values and removing those records from the analysis.

An advantage of discriminant analysis as a classifier (and similar to linear regression as a predictive method), is that it provides estimates of single predictor contributions. This is useful for obtaining a ranking of the importance of predictors, and for variable selection.

Finally, the method is computationally simple, is parsimonious, and is especially useful for small datasets. With its parametric form, discriminant analysis "makes the most out of the data" and is therefore especially useful where the data are few (as explained in Section 10.6).

10.11 Exercises

Personal loan acceptance: Universal Bank is a relatively young bank growing rapidly in terms of overall customer acquisition. The majority of these customers are liability customers with varying sizes of relationship with the bank. The customer base of asset customers is quite small, and the bank is interested in expanding this base rapidly to bring in more loan business. In particular, it wants to explore ways of converting its liability customers to personal loan customers.

A campaign the bank ran for liability customers last year showed a healthy conversion rate of over 9% successes. This has encouraged the retail marketing department to devise smarter campaigns with better target marketing. The goal of our analysis is to model the previous campaign's customer behavior to analyze what combination of factors make a customer more likely to accept a personal loan. This will serve as the basis for the design of a new campaign.

The file UniversalBank.xls contains data on 5000 customers. The data include customer demographic information (age, income...), the customer's relationship with the bank (mortgage, securities account, etc.), and the customer response to the last personal loan campaign (Personal Loan). Among these 5000 customers only 480 (= 9.6%) accepted the personal loan that was offered to them in the previous campaign.

Partition the data (60% training and 40% validation) and then perform a discriminant analysis that models Personal Loan as a function of the remaining predictors (excluding zipcode). Remember to turn categorical predictors with more than 2 categories into dummy variables first. Specify the "success" class as 1 (loan acceptance), and use the default cutoff value of 0.5.

1. Compute summary statistics for the predictors separately for loan acceptors and non-acceptors. For continuous predictors compute the mean and standard deviation. For categorical predictors compute percentages. Are there predictors where the two classes differ substantially?
2. Examine the model performance on the validation set.
   (a) What is the misclassification rate?
   (b) Is there one type of misclassification that is more likely than the other?
   (c) Select 3 customers who were misclassified as acceptors and 3 that were misclassified as non-acceptors. The goal is to determine why they are misclassified. First, examine their probability of being classified as acceptors: is it close to the threshold of 0.5? If not, compare their predictor values to the summary statistics of the two classes in order to determine why they were misclassified.
3. As in many marketing campaigns, it is more important to identify customers who will accept the offer rather than customers who will not accept it. Therefore, a good model should be especially accurate at detecting acceptors. Examine the lift chart and decile chart for the validation set and interpret them in light of this goal.
4. Compare the results from the discriminant analysis with those from a logistic regression (both with cutoff 0.5 and the same predictors). Examine the confusion matrices, the lift charts, and the decile charts. Which method performs better on your validation set in detecting the acceptors?
5. The bank is planning to continue its campaign by sending its offer to 1000 additional customers. Suppose the cost of sending the offer is $1 and the profit from an accepted offer is $50. What is the expected profitability of this campaign?
6. The cost of misclassifying a "loan acceptor" customer as a "non-acceptor" is much higher than the opposite misclassification cost. In order to minimize the expected cost of misclassification, should the cutoff value for classification (which is currently at 0.5) be increased or decreased?

Identifying good system administrators: A management consultant is studying the roles played by experience and training in a system administrator's ability to complete a set of tasks in a specified amount of time. In particular, she is interested in discriminating between administrators who are able to complete given tasks within a specified time and those who are not. Data are collected on the performance of 75 randomly selected administrators. They are stored in the file SystemAdministrators.xls.

Using these data, the consultant performs a discriminant analysis. The variable "Experience" measures months of full time system administrator experience, while "Training" measures number of relevant training credits. The dependent variable "Completed" is either "yes" or "no", according to whether the administrator completed the tasks or not.

7. Create a scatterplot of Experience vs. Training using color or symbol to differentiate administrators who completed the tasks from those who did not complete them. See if you can identify a line that separates the two classes with minimum misclassification.
8. Run a discriminant analysis with both predictors using the entire dataset as training data. Among those who completed the tasks, what is the percentage of administrators who are incorrectly classified as failing to complete the tasks?
9. Compute the two classification scores for an administrator with 4 years of higher education and 6 credits of training. Based on these, how would you classify this administrator?
10. How much experience must be accumulated by a administrator with 4 training credits before his/her estimated probability of completing the tasks exceeds 50%?
11. Compare the classification accuracy of this model to that resulting from a logistic regression with cutoff 0.5.
12. Compute the correlation between Experience and Training for administrators that completed the tasks and compare it to the correlation of administrators who did not complete the tasks. Does the equal correlation assumption seem reasonable?

Detecting Spam Email (from the UCI Machine Learning Repository): A team at Hewlett Packard collected data on a large amount of email messages from their postmaster and personal email for the purpose of finding a classifier that can separate email messages that are spam vs. non-spam (AKA "ham"). The spam concept is diverse: it includes advertisements for products or websites, "make money fast" schemes, chain letters, pornography, etc. The definition used here is "unsolicited commercial e-mail". The file Spambase.xls contains information on 4601 email messages, among which 1813 are tagged "spam". The predictors include 57 attributes, most of them are the average number of times a certain word (e.g., "mail", "George") or symbol (e.g., #, !) appears in the email. A few predictors are related to the number and length of capitalized words.

13. In order to reduce the number of predictors to a manageable size, examine how each predictor differs between the spam and non-spam emails by comparing the spam-class average and non-spam-class average. Which are the 11 predictors that appear to vary the most between spam and non-spam emails? From these 11, which words/signs occur more often in spam?
14. Partition the data into training and validation sets, then perform a discriminant analysis on the training data using only the 11 predictors.
15. If we are interested mainly in detecting spam messages, is this model useful? Use the confusion matrix, lift chart, and decile chart for the validation set for the evaluation.
16. In the sample, almost 40% of the email messages were tagged as spam. However, suppose that the actual proportion of spam messages in these email accounts is 10%. Compute the constants of the classification functions to account for this information.

17. A spam filter that is based on your model is used, so that only messages that are classified as non-spam are delivered, while messages that are classified as spam are quarantined. In this case misclassifying a non-spam email (as spam) has much heftier results. Suppose that the cost of quarantining a non-spam email is 20 times that of not detecting a spam message. Compute the constants of the classification functions to account for these costs (assume that the proportion of spam is reflected correctly by the sample proportion).

Chapter 11

Association Rules

11.1 Introduction

Put simply, association rules, or *affinity analysis* are the study of "what goes with what." For example, a medical researcher wants to learn what symptoms go with what confirmed diagnoses. These methods are also called "market basket analysis," because they originated with the study of customer transactions databases in order to determine dependencies between purchases of different items.

11.2 Discovering Association Rules in Transaction Databases

The availability of detailed information on customer transactions has led to the development of techniques that automatically look for associations between items that are stored in the database. An example is data collected using bar-code scanners in supermarkets. Such 'market basket' databases consist of a large number of transaction records. Each record lists all items bought by a customer on a single purchase transaction. Managers would be interested to know if certain groups of items are consistently purchased together. They could use this data for store layouts to place items optimally with respect to each other, or they could use such information for cross-selling, for promotions, for catalog design and to identify customer segments based on buying patterns. Association rules provide information of this type in the form of "if-then" statements. These rules are computed from the data; Unlike the if-then rules of logic, association rules are probabilistic in nature.

Rules like this are commonly encountered in online "recommendation systems" (or "recommender systems"), where customers examining an item or items for possible purchase are shown other items that are often purchased in conjunction with the first item(s). The display from Amazon.com's online shopping system illustrates the application of rules like this. In the example shown in Figure 11.2, a purchaser of Last Train Home's "Bound Away" audio CD is shown the other CD's most frequently purchased by other Amazon purchasers of this CD.

We introduce a simple artificial example and use it throughout the chapter to demonstrate the concepts, computations, and steps of affinity analysis. We end by applying affinity analysis to a more realistic example of book purchases.

Figure 11.1: Recommendations Based on Association Rules

11.3 Example 1: Synthetic Data on Purchases of Phone Faceplates

A store that sells accessories for cellular phones runs a promotion on faceplates. Customers who purchase multiple faceplates from a choice of six different colors get a discount. The store managers, who would like to know what colors of faceplates customers are likely to purchase together, collected the following transaction database:

Table 11.1: Transactions for Purchases of Different Colored Cellular Phone Faceplates

Transaction #	Faceplate colors purchased			
1	red	white	green	
2	white	orange		
3	white	blue		
4	red	white	orange	
5	red	blue		
6	white	blue		
7	white	orange		
8	red	white	blue	green
9	red	white	blue	
10	yellow			

11.4 Generating Candidate Rules

The idea behind associations rules is to examine all the possible rules between items in an "if-then" format, and select only those that are most likely to be indicators of true dependence.

We use the term *antecedent* to describe the "if" part, and *consequent* to describe the "then" part. In association analysis the antecedent and consequent are sets of items (called *item sets*) that are disjoint (do not have any items in common).

Returning to the phone faceplate purchase example, one example of a possible rule is "if red then white", meaning if a red faceplate is purchased, then a white one is too. Here the antecedent is "red" and the consequent is "white". The antecedent and consequent each contain a single item in this case. Another possible rule is "if red and white, then green". Here the antecedent includes the item set {red, white} and the consequent is {green}.

The first step in affinity analysis is to generate all the rules that would be candidates for indicating associations between items. Ideally, we might want to look at all possible combinations of items in a database with p distinct items (in the phone faceplate example $p = 6$). This means finding all combinations of single items, pairs of items, triplets of items, etc. in the transactions database. However, generating all these combinations requires long computation time that grows exponentially in k. A practical solution is to consider only combinations that occur with higher frequency in the database. These are called *frequent item sets*.

Determining what consists of a frequent item set is related to the concept of *support*. The support of a rule is simply the number of transactions that include both the antecedent and consequent item sets. It is called a support because it measures the degree to which the data "support" the validity of the rule. The support is sometimes expressed as a percentage of the total number of records in the database. For example, the support for the item set {red,white} in the phone faceplate example is 4 ($100 \times \frac{4}{10} = 40\%$).

What constitutes a frequent item set is therefore defined as an item set that has a support that exceeds a selected minimum support, determined by the user.

11.4.1 The Apriori Algorithm

Several algorithms have been proposed for generating frequent item sets, but the classic algorithm is the Apriori algorithm of Agrawal and Srikant (1993). The key idea of the algorithm is to begin by generating frequent item sets with just one item (1-item sets) and to recursively generate frequent item sets with 2 items, then with 3 items, and so on until we have generated frequent item sets of all sizes.

It is easy to generate frequent 1-item sets. All we need to do is to count, for each item, how many transactions in the database include the item. These transaction counts are the supports for the 1-item sets. We drop 1-item sets that have support below the desired minimum support to create a list of the frequent 1-item sets.

To generate frequent 2-item sets, we use the frequent 1-item sets. The reasoning is that if a certain 1-item set did not exceed the minimum support, then any larger size item set that includes it will not exceed the minimum support. In general, generating k-item sets uses the frequent $k-1$-item sets that were generated in the previous step. Each step requires a single run through the database, and therefore the Apriori algorithm is very fast even for a large number of unique items in a database.

11.5 Selecting Strong Rules

From the abundance of rules generated, the goal is to find only the rules that indicate a strong dependence between the antecedent and consequent item sets. To measure the strength of association implied by a rule, we use the measures of *confidence* and *lift ratio* , as described below.

11.5.1 Support and Confidence

In addition to support, which we described earlier, there is another measure that expresses the degree of uncertainty about the "if-then" rule. This is known as the *confidence*[1] of the rule. This measure compares the co-occurrence of the antecedent and consequent item sets in the database to the occurrence of the antecedent item sets. Confidence is defined as the ratio of the number of transactions that include all antecedent and consequent item sets (namely, the support) to the number of transactions that include all the antecedent item sets:

$$Confidence = \frac{\text{\# Transactions with both antecedent and consequent item sets}}{\text{\# Transactions with antecedent item set}}$$

For example, suppose a supermarket database has 100,000 point-of-sale transactions. Of these transactions, 2000 include both orange juice and (over-the-counter) flu medication, and 800 of these include soup purchases. The association rule "IF orange juice and flu medication are purchased THEN soup is purchased on the same trip" has a support of 800 transactions (alternatively 0.8% = 800/100,000) and a confidence of 40% (= 800/2000).

To see the relationship between support and confidence, let us think about what each is measuring (estimating). One way to think of support is that it is the (estimated) probability that a randomly selected transaction from the database will contain all items in the antecedent and the consequent:

$$\text{P(antecedent AND consequent)}$$

In comparison, the confidence is the (estimated) *conditional* probability that a randomly selected transaction will include all the items in the consequent *given* that the transaction includes all the

[1] The concept of confidence is different from and unrelated to the ideas of confidence intervals and confidence levels used in statistical inference.

items in the antecedent:

$$\frac{\text{P(antecedent AND consequent)}}{\text{P(antecedent)}} = \text{P(consequent} \mid \text{antecedent)}$$

A high value of confidence suggests a strong association rule (in which we are highly confident). However, this can be deceptive because if the antecedent and/or the consequent have a high support, we can have a high value for confidence even when they are independent! For example, if nearly all customers buy bananas and nearly all customers buy ice cream, then the confidence level will be high regardless of whether there is an association between the items.

11.5.2 Lift Ratio

A better way to judge the strength of an association rule is to compare the confidence of the rule with a benchmark value, where we assume that the occurrence of the consequent item set in a transaction is independent of the occurrence of the antecedent for each rule. In other words, if the antecedent and consequent item sets are independent, what confidence values would we expect to see? Under independence, the support would be:

$$\text{P(antecedent AND consequent)} = \text{P(antecedent)} \times \text{P(consequent)},$$

and the benchmark confidence would be

$$\frac{\text{P(antecedent)} \times \text{P(consequent)}}{\text{P(antecedent)}} = \text{P(consequent)}$$

The estimate of this benchmark from the data, called the *benchmark confidence value* for a rule is computed by

$$Benchmark\ confidence = \frac{\text{\# Transactions with consequent item set}}{\text{\# Transactions in database}}$$

We compare the confidence to the benchmark confidence by looking at their ratio: this is called the *lift ratio* of a rule. The lift ratio is the confidence of the rule divided by the confidence, assuming independence of consequent from antecedent.

$$lift\ ratio = \frac{confidence}{benchmark\ confidence}$$

A lift ratio greater than 1.0 suggests that there is some usefulness to the rule. In other words, the level of association between the antecedent and consequent item sets is higher than would be expected if they were independent. The larger the lift ratio, the greater the strength of the association.

To illustrate the computation of support, confidence, and lift ratio for the cellular phone faceplates example, we introduce a presentation of the data better suited to this purpose.

11.5.3 Data Format

Transaction data are usually displayed in one of two formats: a list of items purchased (each row representing a transaction), or a binary matrix in which columns are items, rows again represent transactions, and each cell has either a "1" or a "0," indicating the presence or absence of an item in the transaction. For example, Table 11.1 displays the data for the cellular faceplate purchases in item list format. We can translate these into a binary matrix format:

Transaction #	red	white	blue	orange	green	yellow
1	1	1	0	0	1	0
2	0	1	0	1	0	0
3	0	1	1	0	0	0
4	1	1	0	1	0	0
5	1	0	1	0	0	0
6	0	1	1	0	0	0
7	1	0	1	0	0	0
8	1	1	1	0	1	0
9	1	1	1	0	0	0
10	0	0	0	0	0	1

Now, suppose that we want association rules between items for this database that have a support count of at least 2 (equivalent to a percentage support of 2/10=20%). In other words, rules based on items that were purchased together in at least 20% of the transactions. By enumeration we can see that only the following item sets have a count of at least 2:

item set	support (count)
{red}	6
{white}	7
{blue}	6
{orange}	2
{green}	2
{red, white}	4
{red, blue}	4
{red, green}	2
{white, blue}	4
{white, orange}	2
{white, green}	2
{red, white, blue}	2
{red, white, green}	2

The first item set {red} has a support of 6, because 6 of the transactions included a red faceplate. Similarly the last item set {red, white, green} has a support of 2, because only 2 transactions included red, white, and green faceplates.

In XLMiner the user can choose to input data using the Affinity→Association Rules facility in either item-list format or in binary matrix format.

11.5.4 The Process of Rule Selection

The process of selecting strong rules is based on generating all association rules that meet stipulated support and confidence requirements. This is done in two stages. The first stage, described in Section 11.4 consists of finding all "frequent" item sets, those item sets that have a requisite support. In the second stage we generate, from the frequent item sets, association rules that meet a confidence requirement. The first step is aimed at removing item combinations that are rare in the database. The second stage then filters the remaining rules and selects only those with high

confidence. For most association analysis data, the computational challenge is the first stage, as described in discussion of the Apriori algorithm.

The computation of confidence in the second stage is simple. Since any subset (e.g., {red} in the phone faceplate example) must occur at least as frequently as the set it belongs to (e.g. {red, white}), each subset will also be in the list. It is then straightforward to compute the confidence as the ratio of the support for the item set to the support for each subset of the item set. We retain the corresponding association rule only if it exceeds the desired cutoff value for confidence. For example, from the item set {red,white,green} in the phone faceplate purchases we get the following association rules:

Rule 1: $\{\text{red, white}\} \Rightarrow \{\text{green}\}$ with $confidence = \frac{\text{support of } \{\text{red, white, green}\}}{\text{support of } \{\text{red, white}\}} = 2/4 = 50\%;$

Rule 2: $\{\text{red, green}\} \Rightarrow \{\text{white}\}$ with $confidence = \frac{\text{support of } \{\text{red, white, green}\}}{\text{support of } \{\text{red, green}\}} = 2/2 = 100\%;$

Rule 3: $\{\text{white, green}\} \Rightarrow \{\text{red}\}$ with $confidence = \frac{\text{support of } \{\text{red, white, green}\}}{\text{support of } \{\text{white, green}\}} = 2/2 = 100\%;$

Rule 4: $\{\text{red}\} \Rightarrow \{\text{white, green}\}$ with $confidence = \frac{\text{support of } \{\text{red, white, green}\}}{\text{support of } \{\text{red}\}} = 2/6 = 33\%;$

Rule 5: $\{\text{white}\} \Rightarrow \{\text{red, green}\}$ with $confidence = \frac{\text{support of } \{\text{red, white, green}\}}{\text{support of } \{\text{white}\}} = 2/7 = 29\%;$

Rule 6: $\{\text{green}\} \Rightarrow \{\text{red, white}\}$ with $confidence = \frac{\text{support of } \{\text{red, white, green}\}}{\text{support of } \{\text{green}\}} = 2/2 = 100\%;$

If the desired minimum confidence is 70%, we would report only the second, third, and last rules.

We can generate association rules in XLMiner by specifying the minimum support count (2) and minimum confidence level percentage (70%). Figure 11.2 shows the output. Note that here we consider all possible item sets, not just {red, white, green} as above.

XLMiner : Association Rules

Data	
Input Data	Faceplates!$B$1:$G$11
Data Format	Binary Matrix
Minimum Support	2
Minimum Confidence %	70
# Rules	6
Overall Time (secs)	2

Rule 1: If item(s) green= is / are purchased, then this implies item(s) red, white is / are also purchased. This rule has confidence of 100%.

Rule #	Conf. %	Antecedent (a)	Consequent (c)	Support(a)	Support(c)	Support(a U c)	Lift Ratio
1	**100**	**green=>**	**red, white**	**2**	**4**	**2**	**2.5**
2	100	green=>	red	2	6	2	1.666667
3	100	green, white=>	red	2	6	2	1.666667
4	100	green=>	white	2	7	2	1.428571
5	100	green, red=>	white	2	7	2	1.428571
6	100	orange=>	white	2	7	2	1.428571

Figure 11.2: Association Rules for Phone Faceplates Transactions: XLMiner Output

The output includes information on the support of the antecedent, the support of the consequent, and the support of the combined set (denoted by $Support(a \cup c)$). It also gives the confidence of the rule (in %) and the lift ratio. In addition, XLMiner has an "interpreter" that translates the rule from a certain row into English. In the snapshot shown in Figure 11.2, the first rule is highlighted (by clicking), and the corresponding English rule appears in the yellow box:

Rule 1: If item(s) green is/are purchased, then this implies item(s) red, white is/are also purchased. This rule has confidence of 100%

11.5.5 Interpreting the Results

In interpreting results, it is useful to look at the different measures. The support for the rule indicates its impact in terms of overall size – what proportion of transactions is affected? If only a small number of transactions are affected, the rule may be of little use (unless the consequent is very valuable and/or the rule is very efficient in finding it).

The lift ratio indicates how efficient the rule is in finding consequents, compared to random selection. A very efficient rule is preferred to an inefficient rule, but we must still consider support – a very efficient rule that has very low support may not be as desirable as a less efficient rule with much larger support.

The confidence tells us at what rate consequents will be found, and is useful in determining the business or operational viability of a rule: a rule with low confidence may find consequents at too low a rate to be worth the cost of (say) promoting the consequent in all the transactions that involve the antecedent.

11.5.6 Statistical Significance of Rules

What about "confidence" in the non-technical sense? How sure can we be that the rules we develop are meaningful? Considering the matter from a statistical perspective, we can ask "Are we finding associations that are really just chance occurrences?"

Let us examine the output from an application of this algorithm to a small database of 50 transactions, where each of the 9 items is randomly assigned to each transaction. The data are shown in Table 11.2, and the generated association rules are shown in Table 11.3.

In this example, the lift ratios highlight Rule 6 as most interesting in that it suggests purchase of item 4 is almost 5 times as likely when items 3 and 8 are purchased than if item 4 was not associated with the item set {3,8}. Yet we know there is no fundamental association underlying these data – they were randomly generated.

Two principles can guide us in assessing rules for possible spuriousness due to chance effects:

1. The more records the rule is based on, the more solid the conclusion. The key evaluative statistics are based on ratios and proportions, and we can look to statistical confidence intervals on proportions, such as political polls, for a rough preliminary idea of how variable rules might be owing to chance sampling variation. Polls based on 1500 respondents, for example, yield margins of error in the range of plus-or-minus 1.5%.

2. The more distinct rules we consider seriously (perhaps consolidating multiple rules that deal with the same items), the more likely that at least some will be based on chance sampling results. For one person to toss a coin 10 times and get 10 heads would be quite surprising. If 1000 people toss a coin 10 times apiece, it would not be nearly so surprising to have one get 10 heads. Formal adjustment of "statistical significance" when multiple comparisons are made is a complex subject in its own right, and beyond the scope of this book. A reasonable approach is to consider rules from the top down in terms of business or operational applicability, and not consider more than can be reasonably incorporated in a human decision-making process. This will impose a rough constraint on the dangers that arise from an automated review of hundreds or thousands of rules in search of "something interesting."

We now consider a more realistic example, using a larger database and real transactional data.

Table 11.2: 50 Transactions of Randomly Assigned Items

Tr#	Items				
1	8				
2	3	4	8		
3	8				
4	3	9			
5	9				
6	1	8			
7	6	9			
8	3	5	7	9	
9	8				
10	8				
11	1	7	9		
12	1	4	5	8	9
13	5	7	9		
14	6	7	8		
15	3	7	9		
16	1	4	9		
17	6	7	8		
18	8				
19	8				
20	9				
21	2	5	6	8	
22	4	6	9		
23	4	9			
24	8	9			
25	6	8			

Tr#	Items				
26	1	6	8		
27	5	8			
28	4	8	9		
29	9				
30	8				
31	1	5	8		
32	3	6	9		
33	7	9			
34	7	8	9		
35	3	4	6	8	
36	1	4	8		
37	4	7	8		
38	8	9			
39	4	5	7	9	
40	2	8	9		
41	2	5	9		
42	1	2	7	9	
43	5	8			
44	1	7	8		
45	8				
46	2	7	9		
47	4	6	9		
48	9				
49	9				
50	6	7	8		

11.6 Example 2: Rules for Similar Book Purchases

The following example (drawn from the Charles Book Club case) examines associations among transactions involving various types of books. The database includes 2000 transactions, and there are 11 different types of books. The data, in binary matrix form, are shown in Figure 11.6. For instance, the first transaction included $YouthBks$ (youth books) $DoItBks$ (do-it-yourself books) and $GeogBks$ (geography books). Figure 11.6 shows (part of) the rules generated by XLMiner's Association Rules on these data. We specified a minimal support of 200 transactions and a minimal confidence of 50%. This resulted in 49 rules (the first 26 rules are shown in Figure 11.6).

In reviewing these rules, we can see that the the information can be compressed. First, rule #1, which appears from the confidence level to be a very promising rule, is probably meaningless. It says "if Italian cooking books have been purchased, then cookbooks are purchased." It seems likely that Italian cooking books are simply a subset of cookbooks. Rules 2 and 7 involve the same trio of books, with different antecedents and consequents. The same is true of rules 14 and 15 and rules 9 and 10. (Pairs and groups like this are easy to track down by looking for rows that share the same support.) This does not mean the rules are not useful - on the contrary, it can reduce the number of item sets to be considered for possible action from a business perspective.

Table 11.3: Association Rules Output for Random Data

Input	Data:	$A$5:$E$54							
Min.	Support:	2	=	4%					
Min.	Conf. % :	70							
Rule #	Confidence %	Anteced. (a)		Conseq. (c)	Support (a)	Support (c)	Support (a ∪ c)	Confidence If pr(c—a) = pr(c) %	Lift Ratio (conf/ prev.col.)
1	80	2	⇒	9	5	27	4	54	1.5
2	100	5, 7	⇒	9	3	27	3	54	1.9
3	100	6, 7	⇒	8	3	29	3	58	1.7
4	100	1, 5	⇒	8	2	29	2	58	1.7
5	100	2, 7	⇒	9	2	27	2	54	1.9
6	100	3, 8	⇒	4	2	11	2	22	4.5
7	100	3, 4	⇒	8	2	29	2	58	1.7
8	100	3, 7	⇒	9	2	27	2	547	1.9
9	100	4, 5	⇒	9	2	27	2	54	1.9

ChildBks	YouthBks	CookBks	DoItYBks	RefBks	ArtBks	GeogBks	ItalCook	ItalAtlas
0	1	0	1	0	0	1	0	0
1	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0
1	1	1	0	1	0	1	0	0
0	0	1	0	0	0	1	0	0
1	0	0	0	0	1	0	0	0
0	1	0	0	0	0	0	0	0
0	1	0	0	1	0	0	0	0
1	0	0	1	0	0	0	0	0
1	1	1	0	0	0	1	0	0
0	0	0	0	0	0	0	0	0

Figure 11.3: Subset of Book Purchase Transactions in Binary Matrix Format)

11.7 Summary

Affinity analysis (also called "market basket analysis") is a method for deducing rules on associations between purchased items from databases of transactions. The main advantage of this method is that it generates clear, simple rules of the form "IF X is purchased then Y is also likely to be purchased". The method is very transparent and easy to understand.

The process of creating association rules is two-staged. First, a set of candidate rules based on "frequent item sets" is generated (with the Apriori algorithm being the most popular rule generating algorithm). Then, from these candidate rules the ones indicating the strongest association between items are selected. We use the measures of support and confidence to evaluate the uncertainty in a rule. The user also specifies minimal support and confidence values to be used in the rule generation and selection process. A third measure, the lift ratio, compares the efficiency of the rule to detect a real association compared to a random combination.

One shortcoming of association rules is the profusion of rules that are generated. There is therefore a need for ways to reduce these to a small set of useful and strong rules. An important

XLMiner : Association Rules

Data	
Input Data	Assoc_binary!$A$1:$K$2001
Data Format	Binary Matrix
Minimum Support	200
Minimum Confidence %	50
# Rules	49
Overall Time (secs)	2

Rule 1: If item(s) ItalCook= is / are purchased, then this implies item(s) CookBks is / are also purchased. This rule has confidence of 100%.

Rule #	Conf. %	Antecedent (a)	Consequent (c)	Support(a)	Support(c)	Support(a U c)	Lift Ratio
1	100	ItalCook=>	CookBks	227	862	227	2.320186
2	62.77	ArtBks, ChildBks=>	GeogBks	325	552	204	2.274247
3	54.13	CookBks, DoItYBks=>	ArtBks	375	482	203	2.246196
4	61.98	ArtBks, CookBks=>	GeogBks	334	552	207	2.245509
5	53.77	CookBks, GeogBks=>	ArtBks	385	482	207	2.230964
6	57.11	RefBks=>	ChildBks, CookBks	429	512	245	2.230842
7	52.31	ChildBks, GeogBks=>	ArtBks	390	482	204	2.170444
8	60.78	ArtBks, CookBks=>	DoItYBks	334	564	203	2.155264
9	58.4	ChildBks, CookBks=>	GeogBks	512	552	299	2.115885
10	54.17	GeogBks=>	ChildBks, CookBks	552	512	299	2.115885
11	57.87	CookBks, DoItYBks=>	GeogBks	375	552	217	2.096618
12	56.79	ChildBks, DoItYBks=>	GeogBks	368	552	209	2.057735
13	52.49	ArtBks=>	ChildBks, CookBks	482	512	253	2.050376
14	52.12	YouthBks=>	ChildBks, CookBks	495	512	258	2.035985
15	50.39	ChildBks, CookBks=>	YouthBks	512	495	258	2.035985
16	57.03	ChildBks, CookBks=>	DoItYBks	512	564	292	2.022385
17	51.77	DoItYBks=>	ChildBks, CookBks	564	512	292	2.022385
18	56.36	CookBks, GeogBks=>	DoItYBks	385	564	217	1.998711
19	52.9	ArtBks=>	GeogBks	482	552	255	1.916832
20	82.19	ArtBks, DoItYBks=>	CookBks	247	862	203	1.906873
21	53.59	ChildBks, GeogBks=>	DoItYBks	390	564	209	1.900346
22	81.89	DoItYBks, GeogBks=>	CookBks	265	862	217	1.899926
23	80.33	CookBks, RefBks=>	ChildBks	305	846	245	1.899004
24	80	ArtBks, GeogBks=>	ChildBks	255	846	204	1.891253
25	81.18	ArtBks, GeogBks=>	CookBks	255	862	207	1.883445
26	79.63	CookBks, YouthBks=>	ChildBks	324	846	258	1.882497

Figure 11.4: Association Rules for Book Purchase Transactions: XLMiner Output

non-automated method to condense the information involves examining the rules for non-informative and trivial rules, as well as for rules that share the same support.

Another issue that needs to be kept in mind is that rare combinations tend to be ignored, because they do not meet the minimum support requirement. For this reason it is better to have items that are approximately equally frequent in the data. This can be achieved by using higher level hierarchies as the items. An example is to use types of audio CDs rather than names of individual audio CDs in deriving association rules from a database of music store transactions.

11.8 Exercises

Satellite Radio Customers: An analyst at a subscription-based satellite radio company has been given a sample of data from their customer database, with the goal of finding groups of customers that are associated with one another. The data consist of company data, together with purchased demographic data that are mapped to the company data - see Figure 11.5. The analyst decides to apply Association Rules to learn more about the associations between customers. Comment on this approach.

Row Id.	zipconvert_2	zipconvert_3	zipconvert_4	zipconvert_5	homeowner dummy	NUMCHLD	INCOME	gender dummy
17	0	1	0	0	1	1	5	1
25	1	0	0	0	1	1	1	0
29	0	0	0	1	0	2	5	1
38	0	0	0	1	1	1	3	0
40	0	1	0	0	1	1	4	0
53	0	1	0	0	1	1	4	1
58	0	0	0	1	1	1	4	1
61	1	0	0	0	1	1	1	0
71	0	0	1	0	1	1	4	0
87	1	0	0	0	1	1	4	1
100	0	0	0	1	1	1	4	1
104	1	0	0	0	1	1	1	1
121	0	0	1	0	1	1	4	1
142	1	0	0	0	0	1	5	0

Figure 11.5: Sample of Data on Satellite Radio Customers

Cosmetics purchases: The data shown in Figure 11.6 are a subset of a dataset on cosmetic purchases, given in binary matrix form. The complete dataset (in the file Cosmetics.xls) contains data on the purchases of different cosmetic items at a large chain drugstore. The store wants to analyze associations among purchase of these items, for purposes of point of sale display, guidance to sales personnel in promoting cross sales, and for piloting an eventual time-of-purchase electronic recommender system to boost cross sales. Consider first only the subset shown in Figure 11.6.

1. Select several values in the matrix and explain their meaning.

Consider the results of the Association Rules analysis shown in Figure 11.7, and:

2. For the first row, explain the "Conf. %" output and how it is calculated.
3. For the first row, explain the "Support(a), Support(c) and Support($a \cup c$)" output and how it is calculated.
4. For the first row, explain the "Lift Ratio" and how it is calculated.
5. For the first row, explain the rule that is represented there in words.

Now, use the complete dataset on the cosmetics purchases, which is given in the file Cosmetics.xlsindex.

Transaction #	Bag	Blush	Nail Polish	Brushes	Concealer	Eyebrow Pencils	Bronzer
1	0	1	1	1	1	0	1
2	0	0	1	0	1	0	1
3	0	1	0	0	1	1	1
4	0	0	1	1	1	0	1
5	0	1	0	0	1	0	1
6	0	0	0	0	1	0	0
7	0	1	1	1	1	0	1
8	0	0	1	1	0	0	1
9	0	0	0	0	1	0	0
10	1	1	1	1	0	0	0
11	0	0	1	0	0	0	1
12	0	0	1	1	1	0	1

Figure 11.6: Data on Cosmetics Purchases in Binary Matrix Form

6. Using XLMiner, apply Association Rules to these data.
7. Interpret the first three rules in the output in words.
8. Reviewing the first couple of dozen rules, comment on their redundancy, and how you would assess their utility.

Rule #	Conf. %	Antecedent (a)	Consequent (c)	Support(a)	Support(c)	Support(a U c)	Lift Ratio
2	60.19	Bronzer, Nail Polish=>	Brushes, Concealer	103	77	62	3.908713
1	80.52	Brushes, Concealer=>	Bronzer, Nail Polish	77	103	62	3.908713
4	56.36	Brushes=>	Bronzer, Concealer, N	110	76	62	3.708134
3	81.58	Bronzer, Concealer, Nail Polish=>	Brushes	76	110	62	3.708134
6	76.36	Brushes=>	Bronzer, Nail Polish	110	103	84	3.706973
5	81.55	Bronzer, Nail Polish=>	Brushes	103	110	84	3.706973
8	56.88	Concealer, Nail Polish=>	Bronzer, Brushes	109	84	62	3.385758
7	73.81	Bronzer, Brushes=>	Concealer, Nail Polish	84	109	62	3.385758
10	70	Brushes=>	Concealer, Nail Polish	110	109	77	3.211009
9	70.64	Concealer, Nail Polish=>	Brushes	109	110	77	3.211009
12	50	Brushes=>	Blush, Nail Polish	110	82	55	3.04878
11	67.07	Blush, Nail Polish=>	Brushes	82	110	55	3.04878

Figure 11.7: Association Rules for Cosmetics Purchases Data

Online Statistics Courses: Consider the data in CourseTopics.xls, the first few rows of which are shown in Figure 11.8. These data are for purchases of online statistics courses at statistics.com. Each row represents the courses attended by a single customer.

The firm wishes to assess alternative sequencings and combinations of courses. Use Association Rules to analyze these data, and interpret several of the resulting rules.

Course Topics

Intro	DataMining	Survey	Cat Data	Regression	Forecast	DOE	Meta
1	1	0	0	0	0	0	0
0	0	1	0	0	0	0	0
0	1	0	1	1	0	0	1
1	0	0	0	0	0	0	0
1	1	0	0	0	0	0	0
0	1	0	0	0	0	0	0
1	0	0	0	0	0	0	0
0	0	0	1	0	1	1	1
1	0	0	0	0	0	0	0
0	0	0	1	0	0	0	0
1	0	0	0	0	0	0	0

Figure 11.8: Data on Purchases of Online Statistics Courses

Chapter 12

Cluster Analysis

12.1 Introduction

Cluster analysis is used to form groups or clusters of similar records based on several measurements made on these records. The key idea is to characterize these clusters in ways that would be useful for the aims of the analysis. This idea has been applied in many areas, including astronomy, archaeology, medicine, chemistry, education, psychology, linguistics and sociology. Biologists, for example, have made extensive use of classes and sub-classes to organize species. A spectacular success of the clustering idea in chemistry was Mendelev's periodic table of the elements.

One popular use of cluster analysis in marketing is for *market segmentation*: customers are segmented based on demographic and transaction history information, and a marketing strategy is tailored for each segment. Another use is for *market structure analysis*: identifying groups of similar products according to competitive measures of similarity. In marketing and political forecasting, clustering of neighborhoods using US postal zip codes has been used successfully to group neighborhoods by lifestyles. Claritas, a company that pioneered this approach, grouped neighborhoods into 40 clusters using various measures of consumer expenditure and demographics. Examining the clusters enabled Claritas to come up with evocative names, such as "Bohemian Mix," "Furs and Station Wagons" and "Money and Brains," for the groups that captured the dominant lifestyles. Knowledge of lifestyles can be used to estimate the potential demand for products (such as sports utility vehicles) and services (such as pleasure cruises).

In finance, cluster analysis can be used for creating *balanced portfolios*: given data on numerous different investment opportunities (e.g., stocks) one may find clusters based on financial performance variables such as return (daily, weekly, or monthly), volatility, beta, and other characteristics such as industry and market capitalization. Selecting securities from different clusters can help create a balanced portfolio. Another application of cluster analysis in finance is for *industry analysis*: for a given industry, we are interested in finding groups of similar firms based on measures such as growth rate, profitability, market size, product range, presence in various international markets, etc. These groups can then be analyzed in order to understand industry structure and to determine, for instance, who is a competitor.

An interesting and unusual application of cluster analysis, described in Berry & Linoff (1997), is the design of a new set of sizes for army uniforms for women in the US army. The study came up with a new clothing size system with only 20 sizes, where different "sizes" fit different body types. The 20 sizes are combinations of six measurements: Chest, neck, and shoulder circumference, sleeve outseam, and neck to buttock length (for further details see McCulloch, Paal and Ashdown (1998)). This example is important because it shows how a completely new insightful view can be gained by examining clusters of records.

Cluster analysis can be applied to huge amounts of data. For instance, internet search engines

use clustering techniques to cluster queries that users submit. These can then be used for improving search algorithms.

The objective of this chapter is to describe the key ideas underlying the most commonly used techniques for cluster analysis and to lay out their strengths and weaknesses.

Typically, the basic data used to form clusters are a table of measurements on several variables where each column represents a variable and a row represents a record. Our goal is to form groups of records so that similar records are in the same group. The number of clusters may be pre-specified or determined from the data.

12.2 Example: Public Utilities

Table 12.1 gives corporate data on 22 US public utilities (the definition of each variable is given in the bottom table).

We are interested in forming groups of similar utilities. The records to be clustered are the utilities, and the clustering will be based on the 8 measurements on each utility. An example where clustering would be useful is a study to predict the cost impact of deregulation. To do the requisite analysis, economists would need to build a detailed cost model of the various utilities. It would save a considerable amount of time and effort if we could cluster similar types of utilities and build detailed cost models for just one "typical" utility in each cluster and then scale up from these models to estimate results for all utilities.

For simplicity, let us consider only two of the measurements: Sales and Fuel Cost. Figure 12.1 shows a scatterplot of these two variables, with labels marking each company. At first glance, there appear to be two or three clusters of utilities: one with utilities that have high fuel costs; a second with utilities that have lower fuel costs and relatively low sales; and a third with utilities with low fuel costs but high sales. We can therefore think of cluster analysis as a more formal algorithm that measures the distance between records, and according to these distances (here, two-dimensional distances) forms clusters.

There are two general types of clustering algorithms for a dataset of n records:

Hierarchical methods: can be either agglomerative or divisive. Agglomerative methods begin with n clusters and sequentially merge similar clusters until a single cluster is left. Divisive methods work in the opposite direction, starting with one cluster that includes all observations. Hierarchical methods are especially useful when the goal is to arrange the clusters into a natural hierarchy.

Non-hierarchical methods , such as k-means: Using a pre-specified number of clusters, the method assigns cases to each of the clusters. These methods are generally less computationally intensive and are therefore preferred with very large datasets.

We concentrate here on the two most popular methods: hierarchical agglomerative clustering and k-means clustering. In both cases we need to define two types of distances: distance between two records, and distance between two clusters. In both cases there are multiple different metrics that can be used.

12.3 Measuring Distance Between Two Records

We denote by d_{ij} a distance metric, or dissimilarity measure, between records i and j. For record i we have the vector of p measurements $(x_{i1}, x_{i2}, \cdots, x_{ip})$, while for record j we have the vector of measurements $(x_{j1}, x_{j2}, \cdots, x_{jp})$. For example, we can write the measurement vector for *Arizona Public Service* as $[1.06, 9.2, 151, 54.4, 1.6, 9077, 0, 0.628]$.

Distances can be defined in multiple ways, but in general, the following properties are required:

Company	Fixed	RoR	Cost	Load	Demand	Sales	Nuclear	Fuel
Arizona Public Service	1.06	9.2	151	54.4	1.6	9077	0	0.628
Boston Edison Co.	0.89	10.3	202	57.9	2.2	5088	25.3	1.555
Central Louisiana Co.	1.43	15.4	113	53	3.4	9212	0	1.058
Commonwealth Edison Co.	1.02	11.2	168	56	0.3	6423	34.3	0.7
Consolidated Edison Co. (NY)	1.49	8.8	192	51.2	1	3300	15.6	2.044
Florida Power & Light Co.	1.32	13.5	111	60	-2.2	11127	22.5	1.241
Hawaiian Electric Co.	1.22	12.2	175	67.6	2.2	7642	0	1.652
Idaho Power Co.	1.1	9.2	245	57	3.3	13082	0	0.309
Kentucky Utilities Co.	1.34	13	168	60.4	7.2	8406	0	0.862
Madison Gas & Electric Co.	1.12	12.4	197	53	2.7	6455	39.2	0.623
Nevada Power Co.	0.75	7.5	173	51.5	6.5	17441	0	0.768
New England Electric Co.	1.13	10.9	178	62	3.7	6154	0	1.897
Northern States Power Co.	1.15	12.7	199	53.7	6.4	7179	50.2	0.527
Oklahoma Gas & Electric Co.	1.09	12	96	49.8	1.4	9673	0	0.588
Pacific Gas & Electric Co.	0.96	7.6	164	62.2	-0.1	6468	0.9	1.4
Puget Sound Power & Light Co.	1.16	9.9	252	56	9.2	15991	0	0.62
San Diego Gas & Electric Co.	0.76	6.4	136	61.9	9	5714	8.3	1.92
The Southern Co.	1.05	12.6	150	56.7	2.7	10140	0	1.108
Texas Utilities Co.	1.16	11.7	104	54	-2.1	13507	0	0.636
Wisconsin Electric Power Co.	1.2	11.8	148	59.9	3.5	7287	41.1	0.702
United Illuminating Co.	1.04	8.6	204	61	3.5	6650	0	2.116
Virginia Electric & Power Co.	1.07	9.3	174	54.3	5.9	10093	26.6	1.306

Fixed:	Fixed-charge covering ratio (income/debt)
RoR:	Rate of return on capital
Cost:	Cost per KW capacity in place
Load:	Annual Load Factor
Demand:	Peak KWH demand growth from 1974 to 1975
Sales:	Sales (KWH use per year)
Nuclear:	Percent Nuclear
Fuel:	Total fuel costs (cents per KWH)

Table 12.1: Data on 22 Public Utilities Firms

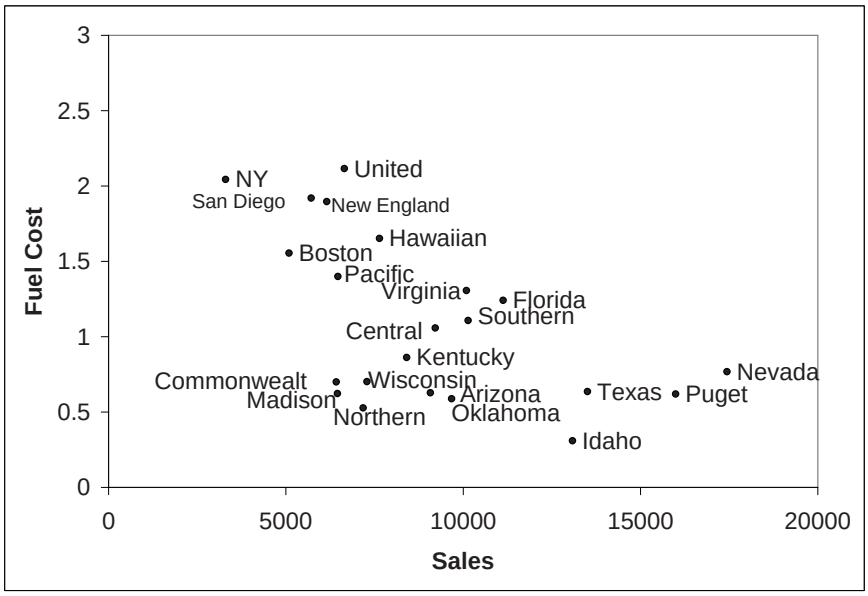


Figure 12.1: Scatterplot of Sales vs. Fuel Cost for the 22 Utilities

Non-negative $d_{ij} \geq 0$

Self-Proximity $d_{ii} = 0$ (the distance from a record to itself is zero)

Symmetry $d_{ij} = d_{ji}$

Triangle inequality $d_{ij} \leq d_{ik} + d_{kj}$ (the distance between any pair cannot exceed the sum of distances between the other two pairs.)

12.3.1 Euclidean Distance

The most popular distance measure is the *Euclidean distance*. The Euclidean distance d_{ij} between two cases, i and j is defined by:

$$d_{ij} = \sqrt{(x_{i1} - x_{j1})^2 + (x_{i2} - x_{j2})^2 + \cdots + (x_{ip} - x_{jp})^2}.$$

For instance, the Euclidean distance between *Arizona Public Service* and *Boston Edison Co.* can be computed from the raw data by:

$$\begin{aligned} d_{12} &= \sqrt{(1.06 - 0.89)^2 + (9.2 - 10.3)^2 + (151 - 202)^2 + \cdots + (0.628 - 1.555)^2} = \\ &= 3989.408 \end{aligned}$$

12.3.2 Normalizing Numerical Measurements

The measure computed above is highly influenced by the scale of each variable, so that variables with larger scales (like Sales) have a much higher influence over the total distance. It is

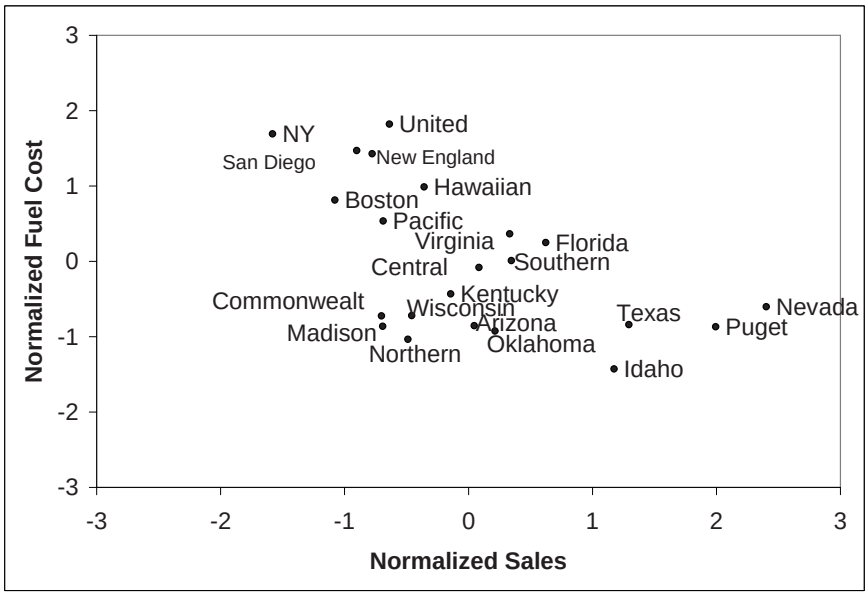


Figure 12.2: Scatterplot of Normalized Sales Vs. Fuel Cost for the 22 Utilities

therefore customary to *normalize* (or, *standardize*) continuous measurements before computing the Euclidean distance. This converts all measurements to the same scale. Normalizing a measurement means subtracting the average and dividing by the standard deviation (normalized values are also called z-scores). For instance, the average Sales across the 22 utilities is 8914.045 and the standard deviation is 3549.984. The normalized sales for *Arizona Public Service* is therefore $(9077 - 8914.045)/3549.984) = 0.046$. Figure 12.2 shows the 22 utilities in the normalized space. You can see that now both Sales and Fuel Cost are on a similar scale. Notice how *Texas* and *Puget* are farther apart in the normalized space compared to the original units space.

Returning to the simplified utilities data with only two measurements (Sales and Fuel Cost), we first normalize the measurements (see Table 12.2), and then compute the Euclidean distance between each pair. Table 12.3 gives these pairwise distanced for the first 5 utilities. A similar table can be constructed for all 22 utilities

12.3.3 Other Distance Measures for Numerical Data

It is important to note that the choice of the distance measure plays a major role in cluster analysis. The main guideline is domain-dependent: What exactly is being measured? How are the different measurements related? What scale should it be treated as (numerical, ordinal, or nominal)? Are there outliers? Finally, depending on the goal of the analysis, should the clusters be distinguished mostly by a small set of measurements or should they be separated by multiple measurements that weight moderately?

Although Euclidean distance is the most widely used distance, it has three main features that need to be kept in mind. First, as mentioned above, it is highly scale-dependent. Changing the units of one variable (e.g., from cents to dollars) can have a huge influence on the results. Standardizing is

Company	Sales	Fuel Cost	NormSales	NormFuel
Arizona Public Service	9077	0.628	0.0459	-0.8537
Boston Edison Co.	5088	1.555	-1.0778	0.8133
Central Louisiana Co.	9212	1.058	0.0839	-0.0804
Commonwealth Edison Co.	6423	0.7	-0.7017	-0.7242
Consolidated Edison Co. (NY)	3300	2.044	-1.5814	1.6926
Florida Power & Light Co.	11127	1.241	0.6234	0.2486
Hawaiian Electric Co.	7642	1.652	-0.3583	0.9877
Idaho Power Co.	13082	0.309	1.1741	-1.4273
Kentucky Utilities Co.	8406	0.862	-0.1431	-0.4329
Madison Gas & Electric Co.	6455	0.623	-0.6927	-0.8627
Nevada Power Co.	17441	0.768	2.4020	-0.6019
New England Electric Co.	6154	1.897	-0.7775	1.4283
Northern States Power Co.	7179	0.527	-0.4887	-1.0353
Oklahoma Gas & Electric Co.	9673	0.588	0.2138	-0.9256
Pacific Gas & Electric Co.	6468	1.4	-0.6890	0.5346
Puget Sound Power & Light Co.	15991	0.62	1.9935	-0.8681
San Diego Gas & Electric Co.	5714	1.92	-0.9014	1.4697
The Southern Co.	10140	1.108	0.3453	0.0095
Texas Utilities Co.	13507	0.636	1.2938	-0.8393
Wisconsin Electric Power Co.	7287	0.702	-0.4583	-0.7206
United Illuminating Co.	6650	2.116	-0.6378	1.8221
Virginia Electric & Power Co.	10093	1.306	0.3321	0.3655
Mean	8914.05	1.10	0.00	0.00
Std	3549.98	0.56	1.00	1.00

Table 12.2: Original and Normalized Measurements for Sales and Fuel Cost

	Arizona	Boston	Central	Commonwealth	Consolidated
Arizona	0				
Boston	2.01	0			
Central	0.77	1.47	0		
Commonwealth	0.76	1.58	1.02	0	
Consolidated	3.02	1.01	2.43	2.57	0

Table 12.3: Distance Matrix Between Pairs of First 5 Utilities, Using Euclidean Distance and Normalized Measurements

therefore a common solution. But unequal weighting should be considered if we want the clusters to depend more on certain measurements and less on others. The second feature of Euclidean distance is that it completely ignores the relationship between the measurements. Thus, if the measurements are in fact strongly correlated a different distance (such as statistical distance, described below) is likely to be a better choice. Third, Euclidean distance is sensitive to outliers. If the data are believed to contain outliers and careful removal is not a choice, the use of more robust distances (such as the Manhattan distance described below) is preferred.

Additional popular distance metrics often used (for reasons such as the ones above):

Correlation-based similarity : Sometimes it is more natural or convenient to work with a similarity measure between records rather than distance which measures dissimilarity. A popular similarity measure is the square of the correlation coefficient, r_{ij}^2, defined by

$$r_{ij}^2 \equiv \frac{\sum_{m=1}^{p}(x_{im}-\overline{x}_m)(x_{jm}-\overline{x}_m)}{\sqrt{\sum_{m=1}^{p}(x_{im}-\overline{x}_m)^2 \sum_{m=1}^{p}(x_{jm}-\overline{x}_m)^2}}$$

Such measures can always be converted to distance measures. In the above example we could define a distance measure $d_{ij} = 1 - r_{ij}^2$.

Statistical distance (also called Mahalanobis distance): this metric has an advantage over the other mentioned metrics in that it takes into account the correlation across the different measurements. With this metric, measurements that are highly correlated with other measurements do not contribute as much as those that are uncorrelated or mildly correlated.

The statistical distance between record i and j is defined as:

$$d_{i,j} = \sqrt{(\mathbf{x_i} - \mathbf{x_j})' S^{-1} (\mathbf{x_i} - \mathbf{x_j})}$$

where $\mathbf{x_i}$ and $\mathbf{x_j}$ are p-dimensional vectors of the measurements values for records i and j respectively; and S is the covariance matrix for these vectors. ($'$ is called a transpose operation, and simply turns a column vector into a row vector). S^{-1} is the inverse matrix of S, which is the p-dimension extension to division. For further information on statistical distance, see Chapter 10.

Manhattan distance ("city-block") : This distance looks at the absolute differences rather than squared differences, and is defined by

$$d_{ij} = \sum_{m=1}^{p} \mid x_{im} - x_{jm} \mid$$

Maximum co-ordinate distance : This distance looks only at the measurement on which records i and j deviate the most. It is defined by

$$d_{ij} = \max_{m=1,2,\cdots,p} \mid x_{im} - x_{jm} \mid$$

12.3.4 Distance Measures for Categorical Data

In the case of measurements with binary values, it is more intuitively appealing to use similarity measures than distance measures. Suppose we have binary values for all the x_{ij}'s, and for records i and j we have the following 2×2 table:

		record j		
		0	1	
record i	0	a	b	$a+b$
	1	c	d	$c+d$
		$a+c$	$b+d$	p

The most useful similarity measures in this situation are:

The matching coefficient , $(a+d)/p$

Jaquard's coefficient , $d/(b+c+d)$. This coefficient ignores zero matches. This is desirable when we do not want to consider two individuals to be similar simply because they both do not have a large number of characteristics. A simple example is when two people who own a Corvette are considered similar whereas two people who do *not* own a Corvette are not considered similar.

12.3.5 Distance Measures for Mixed Data

When the measurements are mixed (some continuous and some binary), a similarity coefficient suggested by Gower is very useful. Gower's similarity measure is a weighted average of the distances computed for each variable, after scaling each variable to a [0,1] scale. It is defined as

$$s_{ij} = \frac{\sum_{m=1}^{p} w_{ijm} s_{ijm}}{\sum_{m=1}^{p} w_{ijm}}$$

with $w_{ijm} = 1$ subject to the following rules:

1. $w_{ijm} = 0$ when the value of the measurement is not known for one of the pair of records
2. For non-binary categorical measurements $s_{ijm} = 0$ unless the records are in the same category in which case $s_{ijm} = 1$
3. For continuous measurements $s_{ijm} = 1 - \frac{|x_{im} - x_{jm}|}{\max(x_m) - \min(x_m)}$

12.4 Measuring Distance Between Two Clusters

We define a cluster as a set of one or more records. How do we measure distance between clusters? The idea is to extend measures of **distance between records** into **distances between clusters**. Consider cluster A which includes the m records $A_1, A_2, \cdots A_m$ and cluster B, which includes n records $B_1, B_2, \cdots B_n$. The most widely used measures of distance between clusters are:

Minimum distance (single linkage) - the distance between the pair of records A_i and B_j that are closest:

$$\min(distance(A_i, B_j)) \quad i = 1, 2 \ldots m; j = 1, 2 \ldots n$$

Maximum distance (complete linkage) - the distance between the pair of records A_i and B_j that are farthest:

$$\max(distance(A_i, B_j)) \quad i = 1, 2 \ldots m; j = 1, 2 \ldots n$$

Average distance (average linkage) - the average distance of all possible distances between records in one cluster and records in the other cluster:

$$Average(distance(A_i, B_j)) \quad i = 1, 2 \ldots m; j = 1, 2 \ldots n$$

Centroid distance - the distance between the two cluster centroids. A cluster centroid is the vector of measurement averages across all the records in that cluster. For cluster A, this is the vector $\overline{x}_A = (\frac{1}{m}\sum_{i=1}^{m} x_{1i}, \ldots \frac{1}{m}\sum_{i=1}^{m} x_{pi})$. The centroid distance between clusters A and B is:

$$|\overline{x}_A - \overline{x}_B|.$$

For instance, consider the first two utilities (Arizona, Boston) as cluster A, and the next 3 utilities (Central, Commonwealth, Consolidated) as cluster B. Using the normalized scores in Table 12.2 and the distance matrix in Table 12.3 we can compute each of the above distances:

- The closest pair is Arizona and Commonwealth, and therefore the minimum distance between clusters A and B is 0.76.
- The farthest pair is Arizona and Consolidated, and therefore the maximum distance between clusters A and B is 3.02.
- The average distance is $(0.77 + 0.76 + 3.02 + 1.47 + 1.58 + 1.01)/6 = 1.44$.
- The centroid of cluster A is

$$[(0.0459 - 1.0778)/2, -0.8537 + 0.8133)/2] = [-0.516, -0.020]$$

and the centroid of cluster B is

$$[(0.0839 - 0.7017 - 1.5814)/3, (-.0804 - 0.7242 + 1.6926)/3] = [-0.733, 0.296]$$

The distance between the two centroids is then

$$\sqrt{(-0.516 + 0.733)^2 + (-0.020 + 0.296)^2} = 0.38$$

In deciding among clustering methods, domain knowledge is key. If you have good reason to believe that the clusters might be chain- or sausage-like, minimum distance (single linkage) would be a good choice. This method does not require that cluster members all be close to one another, only that the new members being added be close to one of the existing members. An example of an application where this might be the case would be characteristics of crops planted in long rows, or disease outbreaks along navigable waterways that are the main areas of settlement in a region. Another example is laying and finding mines (land or marine). Single linkage is also fairly robust to small deviations in the distances. However, adding or removing data can influence it greatly.

Complete and average linkage are better choices if you know that the clusters are more likely to be spherical (for example, customers clustered on the basis of numerous attributes). If you do not know the likely nature of the cluster, these are good default choices, since most clusters tend to be spherical in nature.

We now move to a more detailed description of the two major types of clustering algorithms: hierarchical (agglomerative) and non-hierarchical.

	Arizona,Commonwealth	Boston	Central	Consolidated
Arizona,Commonwealth	0			
Boston	min(2.01,1.58)	0		
Central	min(0.77,1.47)	1.47	0	
Consolidated	min(3.02,2.57)	1.01	2.43	0

Table 12.4: Distance Matrix after Arizona, Commonwealth Consolidation Cluster Together, Using Single Linkage

12.5 Hierarchical (Agglomerative) Clustering

The idea behind hierarchical agglomerative clustering is to start with each cluster comprising of exactly one record and then progressively agglomerating (combining) the two nearest clusters until there is just one cluster left at the end, which consists of all the records.

The hierarchical agglomerative clustering algorithm:

1. Start with n clusters (each observation = cluster).
2. The two closest observations are merged into one cluster.
3. At every step, the two clusters with smallest distance are merged. This means that either single observations are added to existing clusters or two existing clusters are combined.

Returning to the small example of 5 utilities and two measures (Sales and Fuel Cost) and using the distance matrix (Table 12.3), the first step in the hierarchical clustering would join *Arizona* and *Commonwealth*, which are the closest (using normalized measurements and Euclidean distance). Next, we would recalculate a 4×4 distance matrix that would have the distances between these 4 clusters: $\{Arizona, Commonwealth\}, \{Boston\}, \{Central\}$, and $\{Consolidated\}$. At this point we use measure of distance between clusters, such as the ones described in the previous section. Each of these distances (minimum, maximum, average, and centroid distance) can be implemented in the hierarchical scheme as follows:

12.5.1 Minimum Distance (Single Linkage)

In minimum distance clustering the distance between two clusters that is used is the minimum distance (the distance between the nearest pair of records in the two clusters, one record in each cluster). In our small utilities example, we would compute the distances between each of $\{Boston\}$, $\{Central\}$,$\{consolidated\}$ with $\{Arizona, Commonwealth\}$ to create the 4×4 distance matrix shown in Table 12.4

The next step would consolidate $\{Central\}$ with $\{Arizona, Commonwealth\}$ because these two clusters are closest. The distance matrix will again be recomputed (this time it will be 3×3), etc.

This method has a tendency to cluster together at an early stage records that are distant from each other because of a chain of intermediate records in the same cluster. Such clusters have elongated sausage-like shapes when visualized as objects in space.

12.5.2 Maximum Distance (Complete Linkage)

In maximum distance clustering (also called complete linkage) the distance between two clusters is the maximum distance (between the farthest pair of records). If we used complete linkage with the 5 utilities example, the recomputed distance matrix would equivalent to Table 12.4, except that the "min" function would be replaced with a "max".

This method tends to produce clusters at the early stages with records that are within a narrow range of distances from each other. If we visualize them as objects in space, the records in such clusters would have roughly spherical shapes.

12.5.3 Group Average (Average Linkage)

Group average clustering is based on the average distance between clusters (between all possible pairs of records). If we used average linkage with the 5 utilities example, the recomputed distance matrix would equivalent to Table 12.4, except that the "min" function would be replaced with an "average."

Note that the results of the single linkage and the complete linkage methods depend only on the order of the inter-record distances and so are invariant to monotonic transformations of the inter-record distances.

12.5.4 Dendrograms: Displaying Clustering Process and Results

A dendrogram is a tree-like diagram that summarizes the process of clustering. At the bottom are the records. Similar records are joined by lines whose vertical length reflects the distance between the records. Figure 12.3 shows the dendrogram that results from clustering all 22 utilities using the 8 normalized measurements, Euclidean distance, and single linkage.

For any given number of clusters we can determine the records in the clusters by sliding a horizontal line up and down until the number of vertical intersections of the horizontal line equals the desired number of clusters. For example, if we wanted to form 6 clusters we would find that the clusters are:
$\{1, 2, 4, 10, 13, 20, 7, 12, 21, 15, 14, 19, 18, 22, 9, 3\}$
$\{8, 16\} = \{Idaho, Puget\}$
$\{6\} = \{Florida\}$
$\{17\} = \{SanDiego\}$
$\{11\} = \{Nevada\}$
and $\{5\} = \{NY\}$.

Note that if we wanted five clusters they would be identical to the six with the exception that the first two clusters would be merged into one cluster. In general, all hierarchical methods have clusters that are nested within each other as we decrease the number of clusters. This is a valuable property for interpreting clusters and is essential in certain applications, such as taxonomy of varieties of living organisms.

The average linkage dendrogram is shown in Figure 12.4. If we want six clusters using average linkage, they would be:

$$\{1, 14, 19, 18, 3, 6\}; \{2, 4, 10, 13, 20, 22\}; \{5\}; \{7, 12, 9, 15, 21\}; \{17\}; \{8, 16, 11\}$$

12.5.5 Validating Clusters

The goal of cluster analysis is to come up with **meaningful clusters**. Since there are many variations that can be chosen, it is important to make sure the the resulting clusters are valid, in the sense that they really create some insight.

To see whether the cluster analysis is useful, perform the following:

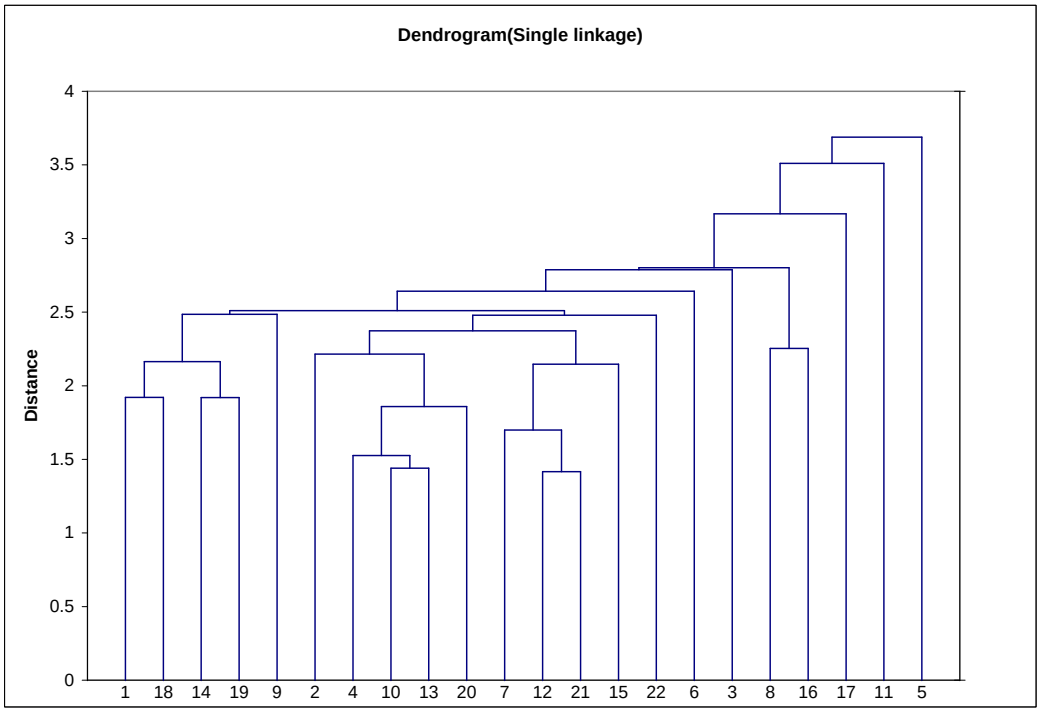


Figure 12.3: Dendrogram: Single Linkage for All 22 Utilities, Using All 8 Measurements

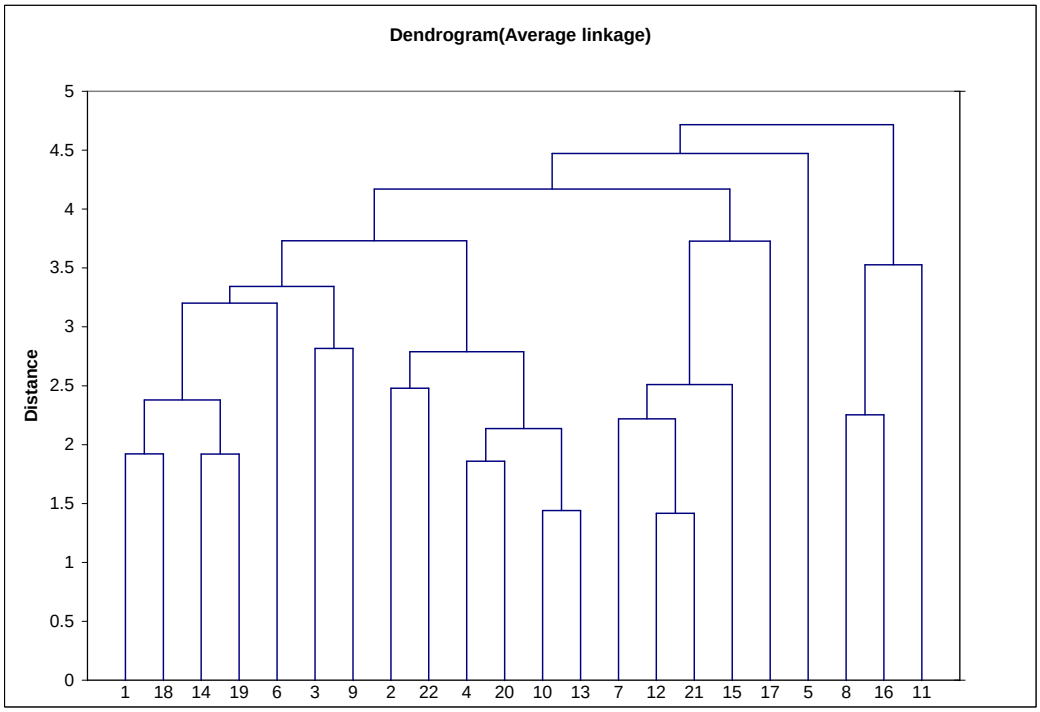


Figure 12.4: Dendrogram: Average Linkage for All 22 Utilities, Using All 8 Measurements

1. **Cluster interpretability:** Is the interpretation of the resulting clusters reasonable? In order to interpret the clusters, we explore the characteristics of each cluster by
   (a) Obtaining summary statistics (e.g., average, min, max) from each cluster on each measurement that was used in the cluster analysis
   (b) Examining the clusters for the presence of some common feature (variable) that was not used in the cluster analysis
   (c) Cluster labeling – based on the interpretation, trying to assign a name or label to each cluster.
2. **Cluster stability**: Do cluster assignments change significantly if some of the inputs were slightly altered? Another way to check stability is to partition the data, and see how well clusters that are formed based on one part apply to the other part. To do this:
   (a) Cluster partition A
   (b) Use the cluster centroids from A to assign each record in partition B (each record is assigned to the cluster with the closest centroid)
   (c) Assess how consistent the cluster assignments are compared to the assignments based on the entire data.
3. **Cluster separation**: Examine the ratio of between-cluster variation to within-cluster variation to see whether the separation is reasonable. There exist statistical tests for this task (an F-ratio), but their usefulness is somewhat controversial.

Returning to the utilities example, we notice that both methods (single and average linkage) identify {5} and {17} as singleton clusters. Also, both dendrograms imply that a reasonable number of clusters in this dataset is four. One insight that can be derived from this clustering is that clusters tend to group geographically: A southern group $\{1, 14, 19, 18, 3, 6\} = \{Arizona, Oklahoma, Southern, Texas, CentralLouisiana, Florida\}$, a northern group $\{2, 4, 10, 13, 20\} = \{Boston, Commonwealth, Madison, NorthernStates, Wisconsin\}$, and an east/west seaboard group: $\{7, 12, 21, 15\} = \{Hawaii, NewEngland, United, Pacific\}$. We can further characterize each of the clusters by examining the summary statistics of their measurements.

12.5.6 Limitations of Hierarchical Clustering

Hierarchical clustering is very appealing in that it does not require the specification of the number of clusters, and in that sense is purely data driven. The ability to represent the clustering process and results through dendrograms is also an advantage of this method, as it is easier to understand and interpret. There are, however, a few limitations to consider:

1. Hierarchical clustering requires the computation and storage of an $n \times n$ distance matrix. For very large datasets, this can be expensive and slow.
2. The hierarchical algorithm makes only one pass through the data. This means that records that are incorrectly allocated early on cannot be reallocated subsequently.
3. Hierarchical clustering also tends to have low stability. Reordering data or dropping a few records can lead to a very different solution.
4. With respect to the choice of distance between clusters, single and complete linkage are robust to changes in the distance metric (e.g., Euclidean, statistical distance) as long as the relative ordering is kept. Average linkage, on the other hand, is much more influenced by the choice of distance metric, and might lead to completely different clusters when the metric is changed.
5. Hierarchical clustering is sensitive to outliers.

12.6 Non-Hierarchical Clustering: The k-Means Algorithm

A non-hierarchical approach to forming good clusters is to pre-specify a desired number of clusters, k, and to assign each case to one of k clusters so as to minimize a measure of dispersion within the clusters. In other words, the goal is to to divide the sample into a predetermined number k, of non-overlapping clusters, so that clusters are as homogeneous as possible with respect to the measurements used.

A very common measure of within-cluster dispersion is the sum of distances (or sum of squared Euclidean distances) of records from their cluster centroid. The problem can be set up as an optimization problem involving integer programming, but because solving integer programs with a large number of variables is time consuming, clusters are often computed using a fast, heuristic method that produces good (although not necessarily optimal) solutions. The k-means algorithm is one such method.

The k-means algorithm starts with an initial partition of the cases into k clusters. Subsequent steps modify the partition to reduce the sum of the distances of each record from its cluster centroid. The modification consists of allocating each record to the nearest of the k centroids of the previous partition. This leads to a new partition for which the sum of distances is smaller than before. The means of the new clusters are computed and the improvement step is repeated until the improvement is very small.

The k-means clustering algorithm:

1. Start with k initial clusters (user chooses k).
2. At every step each record is re-assigned to the cluster with the "closest" centroid.
3. Recompute the centroids of clusters who lost or gained a record, and repeat step 2.
4. Stop when moving any more records between clusters increases cluster dispersion.

Returning to the example with the five utilities and two measurements, let us assume that $k = 2$ and that the initial clusters are $A = \{Arizona, Boston\}$ and $B = \{Central, Commonwealth, Consolidated\}$. The cluster centroids were computed in the previous section:

$\overline{x}_A = [-0.516, -0.020]$ and $\overline{x}_B = [-0.733, 0.296]$.

Now, we compute the distance of each record from each of these two centroids:

	Dist from centroid A	Dist from centroid B
Arizona	1.0052	1.3887
Boston	1.0052	0.6216
Central	0.6029	0.8995
Commonwealth	0.7281	1.0207
Consolidated	2.0172	1.6341

We see that *Boston* is closer to cluster B, and that *Central* and *Commonwealth* are each closer to cluster A. We therefore move each of these records to the other cluster and obtain: $A = \{Arizona, Central, Commonwealth\}$ and $B = \{Consolidated, Boston\}$. Recalculating the centroids gives

$\overline{x}_A = [-0.191, -0.553]$ and $\overline{x}_B = [-1.33, 1.253]$

Once again, we compute the distance of each record from each of the newly calculated centroids:

	Dist from centroid A	Dist from centroid B
Arizona	0.3827	2.5159
Boston	1.6289	0.5067
Central	0.5463	1.9432
Commonwealth	0.5391	2.0745
Consolidated	2.6412	0.5067

At this point we stop, because each record is allocated to its closest cluster.

12.6.1 Initial Partition Into k Clusters

The choice of the number of clusters can be driven either by external considerations (previous knowledge, practical constraints, etc.), or we can try a few different values for k and compare the resulting clusters.

After choosing k, the n records are partitioned into these initial clusters. If there is external reasoning that suggests a certain partitioning, this information should be used. Alternatively, if there exists external information on the centroids of the k clusters, this can be used to allocate the records.

In many cases, there is no information to be used for the initial partition. In these cases, the algorithm can be rerun with different randomly generated starting partitions to reduce the chances of the heuristic producing a poor solution. The number of clusters in the data is generally not known so it is a good idea to run the algorithm with different values for k that are near the number of clusters one expects from the data, to see how the sum of distances reduces with increasing values of k. Note that the clusters obtained using different values of k will not be nested (unlike those obtained by hierarchical methods).

The results of running the k-means algorithm for all 22 utilities and 8 measurement with $k = 6$ are shown in Figure 12.5. As in the results from the hierarchical clustering, we see once again that $\{5\}$ is a singleton cluster, and that some of the previous "geographic" clusters show up here as well.

In order to characterize the resulting clusters, we examine the cluster centroids (Figure 12.6). We can see, for instance, that cluster 1 has the highest average *Nuclear*, a very high *RoR*, and a slow demand growth. In contrast, cluster 3 has the highest *Sales*, with no *Nuclear*, a high *Demand Growth*, and the highest average *Cost*. We can also inspect the information on the within-cluster dispersion. From Figure 12.6 we see that cluster 2 has the highest average distance, and it includes only two records. In contract, cluster 1 which includes 5 records has the lowest within-cluster average distance. This is true for both normalized measurements (bottom left table) and original units (bottom right table). This means that cluster 1 is more homogeneous.

From the distances between clusters we can learn about the separation of the different clusters. For instance, we see that cluster 2 is very different from the other clusters except cluster 3. This might lead us to examine the possibility of merging the two. Cluster 5, which is a singleton cluster, appears to be very far from all the other clusters.

XLMiner : k-Means Clustering - Predicted Clusters

(Distance from Cluster Centers are in normalized Co-ordinates) Back to M

Row Id.	Cluster id	Dist clust-1	Dist clust-2	Dist clust-3	Dist clust-4	Dist clust-5	Dist clust-6
4	1	1.3452	4.3621	3.8339	2.7494	4.2245	3.1049
10	1	1.0029	4.8342	3.5693	3.2345	4.1405	3.7869
13	1	1.5118	5.032	3.7204	4.0716	4.87	4.4756
20	1	1.3849	4.6365	3.6625	2.9824	4.4891	3.3278
22	1	1.9086	2.6776	2.7989	3.0881	3.8605	2.8665
11	2	4.7561	2.4316	3.7481	4.5463	6.613	5.0225
17	2	4.8471	2.4316	5.0856	5.3699	5.761	3.4106
8	3	3.5856	3.9739	1.6141	3.8148	5.2768	3.8129
9	3	3.2057	4.5496	2.3468	2.9569	4.5955	3.2353
16	3	4.3344	4.0545	1.6392	4.7394	5.9536	4.7098
1	4	2.8046	3.2553	2.8973	1.9144	4.2202	2.9635
3	4	3.8904	5.7882	4.174	2.2184	4.5739	4.5489
6	4	3.551	5.6974	4.6489	2.2677	4.7084	3.8032
14	4	3.5124	4.4748	4.169	1.5343	4.9322	4.3896
18	4	2.7316	3.536	2.763	1.3952	4.4402	2.5216
19	4	3.9634	4.9218	4.213	1.4988	5.2546	4.3521
5	5	4.0768	5.7051	4.9534	4.3258	0.00002	3.786
2	6	2.4426	3.6051	3.9153	3.7564	3.9435	1.9151
7	6	4.0822	4.888	3.829	3.6938	4.7044	1.9664
12	6	3.5029	3.8347	3.4951	3.3834	3.6887	0.92426
15	6	3.8236	3.448	4.1034	3.5875	4.3554	1.633
21	6	3.9527	3.4058	3.7556	4.1582	3.7299	1.2034

Figure 12.5: Output for k-Means Clustering with $k = 6$ of 22 Utilities (After Sorting by Cluster ID)

Cluster centers

Cluster	Fixed	RoR	Cost	Load_factor	Demand	Sales	Nuclear	Fuel
Cluster-1	1.112	11.480001	177.200001	55.380002	3.76	7487.399702	38.280034	0.7716
Cluster-2	0.755001	6.949994	154.500005	56.700001	7.749996	11577.49951	4.149999	1.344
Cluster-3	1.2	10.7	221.666487	57.800002	6.566652	12493.01588	-0.000008	0.597
Cluster-4	1.185	12.400001	120.833197	54.650001	0.799999	10456.00045	3.750008	0.8765
Cluster-5	1.49	8.8	192.000002	51.20002	0.999999	3300.012277	15.600001	2.044
Cluster-6	1.048	9.920001	184.600002	62.14001	2.300001	6400.400459	5.240004	1.724

Distance between cluster centers	Cluster-1	Cluster-2	Cluster-3	Cluster-4	Cluster-5	Cluster-6
Cluster-1	0	4090.309917	5005.961483	2969.338323	4187.479063	1087.549967
Cluster-2	4090.309917	0	917.995763	1122.041163	8277.584943	5177.193266
Cluster-3	5005.961483	917.995763	0	2039.52432	9193.06908	6092.733626
Cluster-4	2969.338323	1122.041163	2039.52432	0	7156.353693	4056.109579
Cluster-5	4187.479063	8277.584943	9193.06908	7156.353693	0	3100.434146
Cluster-6	1087.549967	5177.193266	6092.733626	4056.109579	3100.434146	0

Data summary

Cluster	#Obs	Average distance in cluster
Cluster-1	5	1.431
Cluster-2	2	2.432
Cluster-3	3	1.867
Cluster-4	6	1.805
Cluster-5	1	0
Cluster-6	5	1.528
Overall	22	1.64

Data summary (In Original coordinates)

Cluster	#Obs	Average distance in cluster
Cluster-1	5	1042.936117
Cluster-2	2	5863.533146
Cluster-3	3	2724.981548
Cluster-4	6	1241.097807
Cluster-5	1	0.012277017
Cluster-6	5	624.4372161
Overall	22	1622.067124

Figure 12.6: Cluster Centroids and Distances for k-Means with $k = 6$

Finally, we can use the information on the distance between the final clusters to evaluate the cluster validity. The ratio of the sum of squared distances for a given k to the sum of squared distances to the mean of all the records ($k = 1$) is a useful measure for the usefulness of the clustering. If the ratio is near 1.0 the clustering has not been very effective, whereas if it is small we have well-separated groups.

12.7 Exercises

University Rankings: The dataset on American College and University Rankings (available from *www.xlminer.com/exercisedata*) contains information on 1302 American colleges and universities offering an undergraduate program. For each university there are 17 measurements that include continuous measurements (such as tuition and graduation rate) and categorical measurements (such as location by state and whether it is a private/public school).

Note the multiple records that are missing some measurements. Our first goal is to estimate these missing values from "similar" records. This will be done by clustering the complete records, and then finding the closest cluster for each of the partial records. The missing values will be imputed from the information in that cluster.

1. Remove all records with missing measurements from the dataset (by creating a new worksheet).
2. Run hierarchical clustering using all the continuous measurements, using complete linkage and Euclidean distance. Make sure to normalize the measurements. Examine the dendrogram: How many clusters seem reasonable for describing these data?
3. Compare the summary statistics for each cluster and describe each cluster in this context (e.g., "Universities with high tuition, low acceptance rate. . . ")
4. Use the categorical measurements that were not used in the analysis (State and Private/Public) to characterize the different clusters. Is there any relationship between the clusters and the categorical information?
5. Is there other external information that you can think of that explains the contents of some or all of these clusters?
6. Consider Tufts University, which is missing some information. Compute the Euclidean distance of this record from each of the clusters that you found above (using only the measurements that you have). Which cluster is it closest to? Impute the missing values for Tufts by taking the average of the cluster on those measurements.

Pharmaceutical Industry: An equities analyst is studying the pharmaceutical industry and would like your help in exploring and understanding the financial data collected by her firm. Her main objective is to understand the structure of the pharmaceutical industry using some basic financial measures.

Financial data were gathered on 21 firms in the pharmaceutical industry, and is available in the file Pharmaceuticals.xls. For each firm, the following variables are recorded:

(a) market capitalization (in $billion)
(b) beta
(c) price/earnings ratio
(d) return on equity
(e) return on assets
(f) asset turnover
(g) leverage
(h) estimated revenue growth
(i) net profit margin
(j) median recommendation (across major brokerages)
(k) location of firm's headquarters

(l) the stock exchange on which the firm is listed

Use cluster analysis to explore and analyze the given dataset as follows:

1. Use only the quantitative variables (a)-(i) to cluster the 21 firms. Justify the various choices made in conducting the cluster analysis such as weights accorded different variables, the specific clustering algorithm/s used, the number of clusters formed, etc.
2. Interpret the clusters with respect to the quantitative variables that were used in forming the clusters.
3. Is there a pattern in the clusters with respect to the qualitative variables (j)-(l) (that were not used in forming the clusters)?
4. Provide an appropriate name for each cluster using any/all of the variables in the dataset.

Customer Rating of Breakfast Cereals: The dataset Cereals.xls includes nutritional information, store display, and consumer ratings for 77 breakfast cereals.

1. Remove all cereals with missing values.
2. Apply hierarchical clustering to the data using Euclidean distance to the standardized measurements. Compare the dendrograms from single linkage, complete linkage, and cluster centroids. Comment on the structure of the clusters and on their stability.
3. Which method leads to the most insightful/meaningful clusters?
4. Choose one of the methods. How many clusters would you use? What is the distance used for this cutoff? (Look at the dendrogram).
5. The elementary public schools would like to choose a set of cereals to include in their daily cafeterias. Every day a different cereal is offered, but all cereals should support a healthy diet. For this goal you are requested to find a cluster of "healthy cereals." Should the data be standardized? If not, how should they be used in the cluster analysis?

Marketing to Frequent Fliers: The file EastWestAirlinesCluster.xls contains information on 4000 passengers who belong to an airline's frequent flier program. For each passenger the data include information on their mileage history and on different ways they accrued or spent miles in the last year. The goal is to try and identify clusters of passengers that have similar characteristics for the purpose of targeting different segments for different types of mileage offers.

1. Apply hierarchical clustering with Euclidean distance and average linkage. Make sure to standardize the data first. How many clusters appear?
2. What would happen if the data were not standardized?
3. Compare the cluster centroid to characterize the different clusters, and try to give each cluster a label.
4. To check the stability of the clusters, remove a random 5% of the data (by taking a random sample of 95% of the records), and repeat the analysis. Does the same picture emerge?
5. Use k-means clustering with the number of clusters that you found above. Does the same picture emerge?
6. Which clusters would you target for offers, and what type of offers would you target to customers in that cluster?

Chapter 13

Cases

13.1 Charles Book Club

Dataset: CharlesBookClub.xls

THE BOOK INDUSTRY

Approximately 50,000 new titles, including new editions, are published each year in the US, giving rise to a $25 billion industry in 2001[1]

In terms of percentage of sales, this industry may be segmented as follows:

16%	Textbooks
16%	Trade books sold in bookstores
21%	Technical, scientific and professional books
10%	Book clubs and other mail-order books
17%	Mass-market paperbound books
20%	All other books

Book retailing in the US in the 1970's was characterized by the growth of bookstore chains located in shopping malls. The 1980's saw increased purchases in bookstores stimulated through the widespread practice of discounting. By the 1990's, the superstore concept of book retailing gained acceptance and contributed to double-digit growth of the book industry. Conveniently situated near large shopping centers, superstores maintain large inventories of 30,000 to 80,000 titles, and employ well-informed sales personnel. Superstores applied intense competitive pressure on book clubs and mail-order firms as well on as traditional book retailers. (Association of American Publishers. Industry Statistics, 2002.) In response to these pressures, book clubs sought out alternative business models that were more responsive to their customers' individual preferences.

Historically, book clubs offered their readers different types of membership programs. Two common membership programs are "continuity" and "negative option" programs that were extended contractual relationships between the club and its members.

Under a continuity program, a reader would sign up by accepting an offer of several books for just a few dollars (plus shipping and handling) and an agreement to receive a shipment of one or two books each month thereafter at more standard pricing. The continuity program was most common

[1]This case was derived, with the assistance of Ms. Vinni Bhandari from *The Bookbinders Club, a Case Study in Database Marketing*, prepared by Nissan Levin and Jacob Zahavi, Tel Aviv University.

in the children's books market, where parents are willing to delegate the rights to the book club to make a selection, and much of the club's prestige depends on the quality of its selections.

In a negative option program, readers get to select which and how many additional books they would like to receive. However, the club's selection of the month will be automatically delivered to them unless they specifically mark "no' by a deadline date on their order form. Negative option programs sometimes result in customer dissatisfaction and always give rise to significant mailing and processing costs.

In an attempt to combat these trends, some book clubs have begun to offer books on a "positive option" basis, but only to specific segments of their customer base that are likely to be receptive to specific offers. Rather than expanding the volume and coverage of mailings, some book clubs are beginning to use database-marketing techniques to more accurately target customers. Information contained in their databases is used to identify who is most likely to be interested in a specific offer. This information enables clubs to carefully design special programs tailored to meet their customer segments' varying needs.

DATABASE MARKETING AT CHARLES

The club

The Charles Book Club ("CBC") was established in December of 1986, on the premise that a book club could differentiate itself through a deep understanding of its customer base and by delivering uniquely tailored offerings. CBC focused on selling specialty books by direct marketing through a variety of channels, including media advertising (TV, magazines, newspapers) and mailing. CBC is strictly a distributor and does not publish any of the books that it sells. In line with its commitment to understanding its customer base, CBC built and maintained a detailed database about its club members. Upon enrollment, readers were required to fill out an insert and mail it to CBC. Through this process, CBC created an active database of 500,000 readers; most were acquired through advertising in specialty magazines.

The problem

CBC sent mailings to its club members each month containing the latest offerings. On the surface, CBC appeared very successful: mailing volume was increasing, book selection was diversifying and growing, and their customer database was increasing. However, their bottom line profits were falling. The decreasing profits led CBC to revisit their original plan of using database marketing to improve mailing yields and stay profitable.

A possible solution

CBC embraced the idea of deriving intelligence from their data to allow them to know their customers better, and enable multiple targeted campaigns where each target audience would receive appropriate mailings.

CBC's management decided to focus its efforts on the most profitable customers and prospects, and to design targeted marketing strategies to best reach them. The two processes they had in place were:

1. Customer acquisition:
    - New members would be acquired by advertising in specialty magazines, newspapers and on TV.
    - Direct mailing and telemarketing would contact existing club members.

- Every new book would be offered to club members before general advertising.

2. Data collection:

   - All customer responses would be recorded and maintained in the database.
   - Any information not being collected that is critical would be requested from the customer.

For each new title, they decided to use a two-step approach:

(a) Conduct a market test, involving a random sample of 7,000 customers from the database to enable analysis of customer responses. The analysis would create and calibrate response models for the current book offering.

(b) Based on the response models, compute a score for each customer in the database. Use this score and a cutoff value to extract a target customer list for direct mail promotion.

Targeting promotions was considered to be of prime importance. Other opportunities to create successful marketing campaigns based on customer behavior data (returns, inactivity, complaints, compliments, etc.) would be addressed by CBC at a later stage.

Art History of Florence

A new title, "The Art History of Florence," is ready for release. CBC sent a test mailing to a random sample of 4,000 customers from its customer base. The customer responses have been collated with past purchase data. The dataset has been randomly partitioned into 3 parts: **Training Data** (1800 customers): initial data to be used to fit response models, **Validation Data** (1400 customers): hold-out data used to compare the performance of different response models, and **Test Data** (800 Customers): data only to be used after a final model has been selected to estimate the likely performance of the model when it is deployed. Each row (or case) in the spreadsheet (other than the header) corresponds to one market test customer. Each column is a variable with the header row giving the name of the variable. The variable names and descriptions are given in Table 13.1.

DATA MINING TECHNIQUES

Various data mining techniques can be used to mine the data collected from the market test. No one technique is universally better than another. The particular context and the particular characteristics of the data are the major factors in determining which techniques perform better in an application. For this assignment, we will focus on two fundamental techniques:

- K-Nearest Neighbor
- Logistic regression

We will compare them with each other as well as with a standard industry practice known as RFM segmentation.

RFM Segmentation

The segmentation process in database marketing aims to partition customers in a list of prospects into homogeneous groups (segments) that are similar with respect to buying behavior. The homogeneity criterion we need for segmentation is propensity to purchase the offering. But since we cannot measure this attribute, we use variables that are plausible indicators of this propensity.

In the direct marketing business the most commonly used variables are the 'RFM variables':

Table 13.1: List of Variables in Charles Book Club Dataset

Variable Name	Description
Seq# ID#	Sequence number in the partition Identification number in the full (unpartitioned) market test dataset
Gender	O=Male 1=Female
M	Monetary- Total money spent on books
R	Recency- Months since last purchase
F	Frequency - Total number of purchases
FirstPurch	Months since first purchase
ChildBks	Number of purchases from the category:Child books
YouthBks	Number of purchases from the category:Youth books
CookBks	Number of purchases from the category:Cookbooks
DoItYBks	Number of purchases from the category:Do It Yourself books I
RefBks	Number of purchases from the category:Reference books (Atlases, Encyclopedias,Dictionaries)
ArtBks	Number of purchases from the category:Art books
GeoBks	Number of purchases from the category:Geography books
ItalCook	Number of purchases of book title: "Secrets of Italian Cooking"
ItalAtlas	Number of purchases of book title: "Historical Atlas of Italy"
ItalArt	Number of purchases of book title: "Italian Art"
Florence	=1 "The Art History of Florence" was bought, = 0 if not
Related purchase	Number of related books purchased

R - Recency - time since last purchase

F - Frequency - the number of previous purchases from the company over a period

M - Monetary - the amount of money spent on the company's products over a period.

The assumption is that the more recent the last purchase, the more products bought from the company in the past, and the more money spent in the past buying the company's products, the more likely the customer is to purchase the product offered.

The 1800 observations in the training data and the 1400 observations in the validation data have been divided into Recency, Frequency and Monetary categories as follows:

Recency:

0-2 months (Rcode=1)

3-6 months (Rcode=2)

7-12 months (Rcode=3)

13 months and up (Rcode=4)

Frequency:

1 book (Fcode=l)

2 books (Fcode=2)

3 books and up (Fcode=3)

Monetary:

$0 - $25 (Mcode=1)

$26 - $50 (Mcode=2)

$51 - $100 (Mcode=3)

$101 - $200 (Mcode=4)

$201 and up (Mcode=5)

The tables below display the 1800 customers in the training data, cross tabulated by these categories. The buyers are summarized in the first five tables and the non-buyers in the next five

tables. These tables are available for Excel computations in the RFM spreadsheet in the data file.

Buyers

Sum of Florence	Mcode					
Fcode	1	2	3	4	5	Grand Total
1	2	2	10	7	17	38
2		3	5	9	17	34
3		1	1	15	62	79
Grand Total	2	6	16	31	96	151
						151
Rcode	1					

Sum of Florence	Mcode					
Fcode	1	2	3	4	5	Grand Total
1	0	0	0	2	1	3
2		1	0	0	1	2
3		1	0	0	5	6
Grand Total	0	2	0	2	7	11
Rcode	2					

Sum of Florence	Mcode					
Fcode	1	2	3	4	5	Grand Total
1	1	0	1	1	5	8
2		0	3	5	5	13
3			0	4	10	14
Grand Total	1	0	4	10	20	35
Rcode	3					

Sum of Florence	Mcode					
Fcode	1	2	3	4	5	Grand Total
1	1	0	1	2	5	9
2		1	1	2	4	8
3		0	0	4	31	35
Grand Total	1	1	2	8	40	52
Rcode	4					

Sum of Florence	Mcode					
Fcode	1	2	3	4	5	Grand Total
1	0	2	8	2	6	18
2		1	1	2	7	11
3			1	7	16	24
Grand Total	0	3	10	11	29	53

All customers (buyers and non-buyers)

Count of Florence	Mcode					
Fcode	1	2	3	4	5	Grand Total
1	20	40	93	166	219	538
2		32	91	180	247	550
3		2	33	179	498	712
Grand Total	20	74	217	525	964	1800
						1800
Rcode	1					

Count of Florence	Mcode					
Fcode	1	2	3	4	5	Grand Total
1	2	2	6	10	15	35
2		3	4	12	16	35
3		1	2	11	45	59
Grand Total	2	6	12	33	76	129
Rcode	2					

Count of Florence	Mcode					
Fcode	1	2	3	4	5	Grand Total
1	3	5	17	28	26	79
2		2	17	30	31	80
3			3	34	66	103
Grand Total	3	7	37	92	123	262
Rcode	3					

Count of Florence	Mcode					
Fcode	1	2	3	4	5	Grand Total
1	7	15	24	51	86	183
2		12	29	55	85	181
3		1	17	53	165	236
Grand Total	7	28	70	159	336	600
Rcode	4					

Count of Florence	Mcode					
Fcode	1	2	3	4	5	Grand Total
1	8	18	46	77	92	241
2		15	41	83	115	254
3			11	81	222	314
Grand Total	8	33	98	241	429	809

Assignment

(a) What is the response rate for the training data customers taken as a whole? What is the response rate for each of the $4 \times 5 \times 3 = 60$ combinations of RFM categories? Which combinations have response rates in the training data that are above the overall response in the training data?

(b) Suppose that we decide to send promotional mail only to the " above average" RFM combinations identified in part a. Compute the response rate in the validation data using these combinations.

(c) Rework parts (a) and (b) with three segments:

segment 1 consisting of RFM combinations that have response rates that exceed twice the overall response rate

segment 2 consisting of RFM combinations that exceed the overall response rate but do not exceed twice that rate, and

segment 3 consisting of the remaining RFM combinations.

Draw the cumulative lift curve (consisting of three points for these three segments) showing the number of customers in the validation dataset on the x axis and cumulative number of buyers in the validation dataset on the y axis.

k-Nearest Neighbor

The *k*-Nearest Neighbor technique can be used to create segments based on product proximity of the offered products to similar products as well as propensity to purchase (as measured by the RFM variables). For "The Art History of Florence," a possible segmentation by product proximity could be created using the following variables:

M: Monetary - Total money ($) spent on books

R: Recency - Months since last purchase

F: Frequency - Total number of past purchases

FirstPurch: Months since first purchase

RelatedPurch: Total number of past purchases of related books, i.e. sum of purchases from Art and Geography categories and of titles "Secrets of Italian Cooking," "Historical Atlas of Italy," and "Italian Art."

(d) Use the *k*-Nearest Neighbor option under the Classify menu choice in XLMiner to classify cases with $k = 1$, $k = 3$ and $k = 11$. Use normalized data (note the checkbox 'normalize input data' in the dialog box) and all five variables.

(e) Use the k-Nearest Neighbor option under the Prediction menu choice in XLMiner to compute a cumulative gains curve for the validation data for $k = 1, k = 3$ and $k = 11$. Use normalized data (note the checkbox 'normalize input data' in the dialog box) and all five variables. The k-NN prediction algorithm gives a numerical value, which is a weighted average of the values of the Florence variable for the k nearest neighbors with weights that are inversely proportional to distance.

Logistic Regression

The Logistic Regression model offers a powerful method for modeling response because it yields well-defined purchase probabilities. (The model is especially attractive in consumer choice settings because it can be derived from the random utility theory of consumer behavior, under the assumption that the error term in the customer's utility function follows a type I extreme value distribution.)

Use the training set data of 1800 observations to construct three logistic regression models with:

- the full set of 15 predictors in the dataset as independent variables and "Florence" as the dependent variable
- a subset that you judge the best

- only the R, F, and M variables.

(f) Score the customers in the validation sample and arrange them in descending order of purchase probabilities.

(g) Create a cumulative gains chart summarizing the results from the three logistic regression models created above, along with the expected cumulative gains for a random selection of an equal number of customers from the validation dataset.

(h) If the cutoff criterion for a campaign is a 30% likelihood of a purchase, find the customers in the validation data that would be targeted and count the number of buyers in this set.

13.2 German Credit

Dataset: GermanCredit.xls

The German Credit dataset[2] has 30 variables and 1000 records, each record being a prior applicant for credit. Each applicant was rated as "good credit" (700 cases) or "bad credit" (300 cases).

New applicants for credit can also be evaluated on these 30 "predictor" variables and classified as a good credit risk or a bad credit risk, based on the predictor variables. All the variables are explained in Tables 13.2-13.3.

(**Note** : The original dataset had a number of categorical variables, some of which have been transformed into a series of binary variables so that they can be appropriately handled by XLMiner. Several ordered categorical variables have been left as is, to be treated by XLMiner as numerical.)

Figure 13.1 shows the values of these variables for the first several records in the case.

OBS#	CHK_ACCT	DURATION	HISTORY	NEW_CAR	USED_CAR	FURNITURE	RADIO/TV	EDUCATION	RETRAINING	AMOUNT	SAV_ACCT	EMPLOYMENT	INSTALL_RATE	MALE_DIV
1	0	6	4	0	0	0	1	0	0	1169	4	4	4	0
2	1	48	2	0	0	0	1	0	0	5951	0	2	2	0
3	3	12	4	0	0	0	0	1	0	2096	0	3	2	0
4	0	42	2	0	0	1	0	0	0	7882	0	3	2	0

MALE_MAR_or_WID	CO-APPLICANT	GUARANTOR	PRESENT_RESIDENT	REAL_ESTATE	PROP_UNKN_NONE	AGE	OTHER_INSTALL	RENT	OWN_RES	NUM_CREDITS	JOB	NUM_DEPENDENTS	TELEPHONE	FOREIGN
0	0	0	4	1	0	67	0	0	1	2	2	1	1	0
0	0	0	2	1	0	22	0	0	1	1	2	1	0	0
0	0	0	3	1	0	49	0	0	1	1	1	2	0	0
0	0	1	4	0	0	45	0	0	0	1	2	2	0	0

Figure 13.1: The Data (First Several Rows)

[2]This is available from ftp.ics.uci.edu/pub/machine-learning-databases/statlog/

Table 13.2: Variables 1-15 for the German Credit Dataset

Var. #	Variable Name	Description	Variable Type	Code Description
1.	OBS#	Observation No.	Categorical	Sequence Number in dataset
2.	CHK_ACCT	Checking account status	Categorical	0 :< 0DM 1: $0 \Leftarrow \cdots < 200DM$ 2 :⇒ 200 DM 3: no checking account
3.	DURATION	Duration of credit in months	Numerical	
4.	HISTORY	Credit history	Categorical	0: no credits taken 1: all credits at this bank paid back duly 2: existing credits paid back duly till now 3: delay in paying off in the past 4: critical account
5.	NEW_CAR	Purpose of credit	Binary	car (new) 0: No, 1: Yes
6.	USED_CAR	Purpose of credit	Binary	car (used) 0: No, 1: Yes
7.	FURNITURE	Purpose of credit	Binary	furniture/equipment 0: No, 1: Yes
8.	RADIO/TV	Purpose of credit	Binary	radio/television 0: No, 1: Yes
9.	EDUCATION	Purpose of credit	Binary	education 0: No, 1: Yes
10.	RETRAINING	Purpose of credit	Binary	retraining 0: No, 1: Yes
11.	AMOUNT	Credit amount	Numerical	
12.	SAV_ACCT	Average balance in savings account	Categorical	0 :< 100 DM 1 : 100 <= $\cdots$ < 500 DM 2 : 500 <= $\cdots$ < 1000 DM 3 :⇒ 1000 DM 4 : unknown/ no savings account
13.	EMPLOYMENT	Present employment since	Categorical	0 : unemployed 1: < 1 year 2 : 1 <= $\cdots$ < 4 years 3 : 4 <= $\cdots$ < 7 years 4 : >= 7 years
14.	INSTALL_RATE	Installment rate as % of disposable income	Numerical	
15.	MALE_DIV	Applicant is male and divorced	Binary	0: No, 1:Yes

Table 13.3: Variables 16-30 for the German Credit Dataset

16.	MALE_SINGLE	Applicant is male and single	Binary	0: No, 1:Yes
17.	MALE_MAR_WID	Applicant is male and married or a widower	Binary	0: No, 1:Yes
18.	CO-APPLICANT	Application has a co-applicant	Binary	0: No, 1:Yes
19.	GUARANTOR	Applicant has a guarantor	Binary	0: No, 1:Yes
20.	PRESENT_RESIDENT	Present resident since - years	Categorical	0 :<= 1 year 1 < ··· <= 2 years 2 < ··· <= 3 years 3 :> 4 years
21.	REAL_ESTATE	Applicant owns real estate	Binary	0: No, 1:Yes
22.	PROP_UNKN_NONE	Applicant owns no property (or unknown)	Binary	0: No, 1:Yes
23.	AGE	Age in years	Numerical	
24.	OTHER_INSTALL	Applicant has other installment plan credit	Binary	0: No, 1:Yes
25.	RENT	Applicant rents	Binary	0: No, 1:Yes
26.	OWN_RES	Applicant owns residence	Binary	0: No, 1:Yes
27.	NUM_CREDITS	Number of existing credits at this bank	Numerical	
28.	JOB	Nature of job	Categorical	0 : unemployed/ unskilled - non-resident 1 : unskilled - resident 2 : skilled employee / official 3 : management/ self-employed/ highly qualified employee/ officer
29.	NUM_DEPENDENTS	Number of people for whom liable to provide maintenance	Numerical	
30.	TELEPHONE	Applicant has phone in his or her name	Binary	0: No, 1:Yes
31.	FOREIGN	Foreign worker	Binary	0: No, 1:Yes
32	RESPONSE	Credit rating is good	Binary	0: No, 1:Yes

The consequences of misclassification have been assessed as follows: the costs of a false positive (incorrectly saying an applicant is a good credit risk) outweigh the benefits of a true positive (correctly saying an applicant is a good credit risk) by a factor of five. This can be summarized in Table 13.4.

Table 13.4: Opportunity Cost Table (in Deutschemark)

		Predicted (Decision)	
Actual		Good (Accept)	Bad (Reject)
	Good	0	100 DM
	Bad	500 DM	0

The opportunity cost table was derived from the average net profit per loan as shown in Table 13.5 below.

Table 13.5: Average Net Profit

		Predicted (Decision)	
Actual		Good (Accept)	Bad (Reject)
	Good	100 DM	0
	Bad	- 500 DM	0

Because decision-makers are used to thinking of their decision in terms of net profits, we will use these table in assessing the performance of the various models.

Assignment

1. Review the predictor variables and guess what their role might be in a credit decision. Are there any surprises in the data?
2. Divide the data into training and validation partitions, and develop classification models using the following data mining techniques in XLMiner:
   - Logistic regression
   - Classification trees
   - Neural networks
3. Choose one model from each technique and report the confusion matrix and the cost/gain matrix for the validation data. Which technique has the most net profit?
4. Let us try and improve our performance. Rather than accept XLMiner's initial classification of all applicants' credit status, use the "predicted probability of success" in logistic regression (where "success" means "1") as a basis for selecting the best credit risks first, followed by poorer risk applicants.

   (a) Sort the validation on "predicted probability of success."
   (b) For each case, calculate the net profit of extending credit.
   (c) Add another column for cumulative net profit.
   (d) How far into the validation data do you go to get maximum net profit? (Often this is specified as a percentile or rounded to deciles.)
   (e) If this logistic regression model is scored to future applicants, what "probability of success" cutoff should be used in extending credit?

13.3 Tayko Software Cataloger

Dataset: Tayko.xls

Background

Tayko is a software catalog firm that sells games and educational software. It started out as a software manufacturer, and added third party titles to its offerings. It has recently put together a revised collection of items in a new catalog, which it is preparing to roll out in a mailing.

In addition to its own software titles, Tayko's customer list is a key asset. In an attempt to expand its customer base, it has recently joined a consortium of catalog firms that specialize in computer and software products.

The consortium affords members the opportunity to mail catalogs to names drawn from a pooled list of customers. Members supply their own customer lists to the pool, and can "withdraw" an equivalent number of names each quarter. Members are allowed to do predictive modeling on the records in the pool so they can do a better job of selecting names from the pool.

The Mailing Experiment

Tayko has supplied its customer list of 200,000 names to the pool, which totals over 5,000,000 names, so it is now entitled to draw 200,000 names for a mailing. Tayko would like to select the names that have the best chance of performing well, so it conducts a test - it draws 20,000 names from the pool and does a test mailing of the new catalog to them.

This mailing yielded 1065 purchasers - a response rate of 0.053. Average spending was $103 for each of the purchasers, or $5.46 per catalog mailed. To optimize the performance of the data mining techniques, it was decided to work with a stratified sample that contained equal numbers of purchasers and non-purchasers. For ease of presentation, the dataset for this case includes just 1000 purchasers and 1000 non-purchasers, an apparent response rate of 0.5. Therefore, after using the dataset to predict who will be a purchaser, we must adjust the purchase rate back down by multiplying each case's "probability of purchase" by 0.053/0.5 or 0.107.

Data

There are two response variables in this case. "Purchase" indicates whether or not a prospect responded to the test mailing and purchased something. "Spending" indicates, for those who made a purchase, how much they spent. The overall procedure in this case will be to develop two models. One will be used to classify records as "purchase" or "no purchase." The second will be used for those cases that are classified as "purchase," and will predict the amount they will spend.

Table 13.6 provides a description of the variables available in this case. A partition variable is used because we will be developing two different models in this case and want to preserve the same partition structure for assessing each model. Figure 13.2 shows the first few rows of data (the top shows the sequence number plus the first 14 variables, and the bottom shows the remaining 11 variables for the same rows).

Table 13.6: Description of Variables for Tayko Dataset

Var. #	Variable Name	Description	Variable Type	Code Description
1.	US	Is it a US address?	binary	1: yes 0: no
2 - 16	Source_*	Source catalog for the record (15 possible sources)	binary	1: yes 0: no
17.	Freq.	Number of transactions in last year at source catalog	numeric	
18.	last_update_days_ago	How many days ago was last update to cust. record	numeric	
19.	1st_update_days_ago	How many days ago was 1st update to cust. record	numeric	
20.	RFM%	Recency-frequency -monetary percentile, as reported by source catalog (see CBC case)	numeric	
21.	Web_order	Customer placed at least 1 order via web	binary	1: yes 0: no
22.	Gender=mal	Customer is male	binary	1: yes 0: no
23.	Address_is_res	Address is a residence	binary	1: yes 0: no
24.	Purchase	Person made purchase in test mailing	binary	1: yes 0: no
25.	Spending	Amount spent by customer in ($) test mailing	numeric	
26.	Partition	Variable indicating which partition the record will be assigned to	alpha	t: training v: validation s: test

sequence_number	US	source_a	source_c	source_b	source_d	source_e	source_m	source_o	source_h	source_r	source_s	source_t	source_u	source_p
1	1	0	0	1	0	0	0	0	0	0	0	0	0	0
2	1	0	0	0	0	1	0	0	0	0	0	0	0	0
3	1	0	0	0	0	0	0	0	0	0	0	1	0	0
4	1	0	1	0	0	0	0	0	0	0	0	0	0	0
5	1	0	1	0	0	0	0	0	0	0	0	0	0	0
6	1	0	0	0	0	0	0	0	0	1	0	0	0	0
7	1	0	0	0	0	0	0	0	0	0	0	0	0	0
8	1	0	0	1	0	0	0	0	0	0	0	0	0	0
9	1	1	0	0	0	0	0	0	0	0	0	0	0	0
10	1	1	0	0	0	0	0	0	0	0	0	0	0	0

source_x	source_w	Freq	last_update_days_ago	1st_update_days_ago	Web order	Gender=male	Address_is_res	Purchase	Spending	Partition
0	0	2	3662	3662	1	0	1	1	128	s
0	0	0	2900	2900	1	1	0	0	0	s
0	0	2	3883	3914	0	0	0	1	127	t
0	0	1	829	829	0	1	0	0	0	s
0	0	1	869	869	0	0	0	0	0	t
0	0	1	1995	2002	0	0	1	0	0	s
0	1	2	1498	1529	0	0	1	0	0	s
0	0	1	3397	3397	0	1	0	0	0	t
0	0	4	525	2914	1	1	0	1	489	t
0	0	1	3215	3215	0	0	0	1	174	v

Figure 13.2: Data for First 10 Records

Assignment

1. Each catalog costs approximately $2 to mail (including printing, postage and mailing costs). Estimate the gross profit that the firm could expect from the remaining 180,000 names if it randomly selected them from the pool.

2. Develop a model for classification a customer as a purchaser or non-purchaser

   (a) Partition the data into training on the basis of the partition variable, which has 800 "t's," 700 "v's" and 500 "s's" (training data, validation data and test data, respectively) randomly assigned to cases.

   (b) Using the "best subset" option in logistic regression, implement the full logistic regression model, select the best subset of variables, then implement a regression model with just those variables to classify the data into purchasers and non-purchasers. (Logistic regression is used because it yields an estimated "probability of purchase," which is required later in the analysis.)

3. Develop a model for predicting spending among the purchasers

   (a) Make a copy of the data sheet (call it data2), sort by the "Purchase" variable, and remove the records where Purchase = "0" (the resulting spreadsheet will contain only purchasers).

   (b) Partition this dataset into training and validation partitions on the basis of the partition variable.

   (c) Develop models for predicting spending, using:

      i. Multiple linear regression (use best subset selection)

      ii. Regression trees

   (d) Choose one model on the basis of its performance with the validation data.

4. Return to the original test data partition. Note that this test data partition includes both purchasers and non-purchasers. Note also that, although it contains the scoring of the chosen classification model, we have not used this partition our analysis up to this point thus it will thus give an unbiased estimate of the performance of our models. It is best to make a copy of the test data portion of this sheet to work with, since we will be adding analysis to it. This copy is called Score Analysis.

   (a) Copy the "predicted probability of success" (success = purchase) column from the classification of test data to this sheet.

   (b) Score the chosen prediction model to this data sheet.

   (c) Arrange the following columns so they are adjacent:

      i. Predicted probability of purchase (success)

      ii. Actual spending $

      iii. Predicted spending $

   (d) Add a column for "adjusted prob. of purchase" by multiplying "predicted prob. of purchase" by 0.107. *This is to adjust for oversampling the purchasers (see above).*

   (e) Add a column for expected spending [adjusted prob. of purchase × predicted spending].

   (f) Sort all records on the "expected spending" column.

   (g) Calculate cumulative lift (= cumulative "actual spending" divided by the average spending that would result from random selection [each adjusted by the .107]).

5. Using this cumulative lift curve, estimate the gross profit that would result from mailing to the 180,000 on the basis of your data mining models.

Note : Although Tayko is a hypothetical company, the data in this case (modified slightly for illustrative purposes) were supplied by a real company that sells software through direct sales. The concept of the a catalog consortium is based upon the Abacus Catalog Alliance. Details can be found at http://www.doubleclick.com/us/solutions/marketers/database/catalog/.

13.4 Segmenting Consumers of Bath Soap

Dataset: BathSoap.xls

Business Situation

The Indian Market Research Bureau (IMRB) is a leading market research agency that specializes in tracking consumer purchase behavior in consumer goods (both durable and non-durable).

IMRB tracks about 30 product categories (e.g. detergents, etc.) and, within each category, about 60 - 70 brands. To track purchase behavior, IMRB constituted about 50,000 household panels in 105 cities and towns in India, covering about 80% of the Indian urban market. (In addition to this, there are 25,000 sample households selected in rural areas; we are working, however, only with urban market data). The households are carefully selected using stratified sampling. The strata are defined on the basis of socio-economic status, and the market (a collection of cities).

IMRB has both transaction data (each row is a transaction) and household data (each row is a household), and, for the household data, maintains the following information:

- Demographics of the households (updated annually)
- Possession of durable goods (car, washing machine, etc.; updated annually); an "affluence index" is computed from this information
- Purchase data of product categories and brands (updated monthly).

IMRB has two categories of clients: (1) Advertising agencies who subscribe to the database services, obtain updated data every month, and use it to advise their clients on advertising and promotion strategies (2) Consumer goods manufacturers who monitor their market share using the IMRB database.

Key Problems

IMRB has traditionally segmented markets on the basis of purchaser demographics. They would like now to segment the market based on two key sets of variables more directly related to the purchase process and to brand loyalty:

1. Purchase behavior (volume, frequency, susceptibility to discounts, and brand loyalty), and
2. Basis of purchase (price, selling proposition)

Doing so would allow IMRB to gain information about what demographic attributes are associated with different purchase behaviors and degrees of brand loyalty, and more effectively deploy promotion budgets.

The better and more effective market segmentation would enable IMRB's clients to design more cost-effective promotions targeted at appropriate segments. Thus, multiple promotions could be launched, each targeted at different market segments at different times of the year. This would result in a more cost-effective allocation of the promotion budget to different market segments. It would also enable IMRB to design more effective customer reward systems and thereby increase brand loyalty.

Data

The data in this sheet profile each household - each row contains the data for one household.

Member Identification	**Member id**		Unique identifier for each household
Demographics	**SEC**	1 - 5 categories	Socio Economic Class (1=high, 5=low)
	FEH	1 - 3 categories	Food eating habits (1=vegetarian, 2=veg. but eat eggs, 3=non veg., 0=not specified)
	MT		Native language (see table in worksheet)
	SEX	1: male, 2: Female	Sex of homemaker
	AGE		Age of homemaker
	EDU	1 - 9 categories	Education of homemaker (1=minimum, 9 = maximum)
	HS	1 - 9	Number of members in household
	CHILD	1- 4 categories	Presence of children in the household
	CS	1 - 2	Television available (1=available, 2= not available)
	Affluence Index		Weighted value of durables possessed

Summarized Purchase Data

Purchase summary of the house hold over the period	**No. of Brands**	Number of brands purchased
	Brand Runs	Number of instances of consecutive purchase of brands
	Total Volume	Sum of volume
	No. of Trans	Number of purchase transactions; Multiple brands purchased in a month are counted as separate transactions
	Value	Sum of value
	Trans/ Brand Runs	Avg. transactions per brand run
	Vol/Tran	Avg. volume per transaction
	Avg. Price	Avg. price of purchase

Purchase within Promotion	**Pur Vol**	Percent of volume purchased
	No Promo - %	under no-promotion
	Pur Vol Promo 6 %	Percent of volume purchased under Promotion Code 6
	Pur Vol Other Promo %	Percent of volume purchased under other promotions

Brand wise purchase	Br. Cd. (57, 144), 55, 272, 286, 24, 481, 352, 5 and 999 (others)	Percent of volume purchased of the brand
Price category wise purchase	**Price** **Cat 1 to 4**	Percent of volume purchased under the price category

Selling proposition wise purchase	**Proposition** **Cat 5 to 15**	Percent of volume purchased under the product proposition category

Measuring Brand Loyalty

Several variables in this case deal measure aspects of brand loyalty. The number of different brands purchased by the customer is one measure. However, a consumer who purchases one or two brands in quick succession then settles on a third for a long streak is different from a consumer who constantly switches back and forth among three brands. How often customers switch from one brand to another is another measure of loyalty. Yet a third perspective on the same issue is the proportion of purchases that go to different brands - a consumer who spends 90% of his or her purchase money on one brand is more loyal than a consumer who spends more equally among several brands.

All three of these components can be measured with the data in the purchase summary worksheet.

Assignment

1. Use k-means clustering to identify clusters of households based on:

   (a) The variables that describe purchase behavior (including brand loyalty)

   (b) The variables that describe basis-for-purchase

   (c) The variables that describe both purchase behavior and basis of purchase.

   Note 1: How should k be chosen? Think about how the clusters would be used. It is likely that the marketing efforts would support 2-5 different promotional approaches.

   Note 2: How should the percentages of total purchases comprised by various brands be treated? Isn't a customer who buys all brand A just as loyal as a customer who buys all brand B? What will be the effect on any distance measure of using the brand share variables as is? Consider using a single derived variable.

2. Select what you think is the best segmentation and comment on the characteristics (demographic, brand loyalty and basis-for-purchase) of these clusters. (This information would be used to guide the development of advertising and promotional campaigns.)

3. Develop a model that classifies the data into these segments. Since this information would most likely be used in targeting direct mail promotions, it would be useful to select a market segment that would be defined as a "success" in the classification model.

APPENDIX

Although not used in the assignment, two additional data sets are provided that were used in the derivation of the summary data.

IMRB_Purchase_Data is a transaction database, where each row is a transaction. Multiple rows in this dataset corresponding to a single household were consolidated into a single household row in IMRD_Summary_Data.

The Durables sheet in IMRB_Summary_Data contains information used to calculate the affluence index. Each row is a household, and each column represents a durable consumer good. A "1" in the column indicates that the durable is possessed by the household; a "0" indicates it is not possessed. This value is multiplied by the weight assigned to the durable item. For example, a "5" indicates the weighted value of possessing the durable. The sum of all the weighted values of the durables possessed equals the Affluence Index.

13.5 Direct Mail Fundraising

Datasets: Fundraising.xls, FutureFundraising.xls

Background

A national veterans organization wishes to develop a data mining model to improve the cost-effectiveness of their direct marketing campaign. The organization, with its in-house database of over 13 million donors, is one of the largest direct mail fundraisers in the United States. According to their recent mailing records, the overall response rate is 5.1%. Out of those who responded (donated), the average donation is $13.00. Each mailing, which includes a gift of personalized address labels and assortments of cards and envelopes, costs $0.68 to produce and send. Using these facts, we take a sample of this dataset to develop a classification model that can effectively capture donors so that the expected net profit is maximized. Weighted sampling is used, under-representing the non-responders so that the sample has equal numbers of donors and non-donors.

Data

The file Fundraising.xls contains 3120 data points with 50% donors (TARGET_B = 1) and 50% non-donors (TARGET_B = 0). The amount of donation (TARGET_D) is also included but is not used in this case. The descriptions for the 25 variables (including two target variables) are listed in Table 13.7.

Assignment

Step 1: Partitioning - Partition the dataset into 60% training and 40% validation (set the seed to 12345).

Step 2: Model Building - follow these steps:

1. **Selecting classification tool and parameters**. Run the following classification tools on the data:
   - Logistic Regression
   - Classification Trees
   - Neural Networks

   Be sure to test different parameter values for each method. You may also want to run each method on a subset of the variables. Be sure NOT to include "TARGET_D" in your analysis.

2. **Classification under asymmetric response and cost:** What is the reasoning behind using weighted sampling to produce a training set with equal numbers of donors and non-donors? Why not use a simple random sample from the original dataset? (Hint: given the actual response rate of 5.1%, how do you think the classification models will behave under simple sampling)? In this case, is classification accuracy a good performance metric for our purposes of maximizing net profit? If not, how would you determine the best model? Explain your reasoning.
3. **Calculate Net Profit:** For each method, calculate the lift of net profit for both the training and validation set based on the actual response rate (5.1%). Again, the expected donation, given that they are donors, is $13.00, and the total cost of each mailing is $0.68. (Hint: to calculate estimated net profit, we will need to "undo" the effects of the weighted

Table 13.7: Description of Variables for the Fundraising Dataset

ZIP :	Zipcode group (zipcodes were grouped into 5 groups; only 4 are needed for analysis since if a potential donor falls into none of the four he or she must be in the other group. Inclusion of all five variables would be redundant and cause some modeling techniques to fail. A "1" indicates the potential donor belongs to this zip group.) 00000-19999 ⇒ 1 (omitted for above reason) 20000-39999 ⇒ zipconvert_2 40000-59999 ⇒ zipconvert_3 60000-79999 ⇒ zipconvert_4 80000-99999 ⇒ zipconvert_5
HOMEOWNER	1 = homeowner, 0 = not a homeowner
NUMCHLD	Number of children
INCOME	Household income
GENDER	Gender, 0 = Male, 1 = Female
WEALTH	Wealth Rating Wealth rating uses median family income and population statistics from each area to index relative wealth within each state The segments are denoted 0-9, with 9 being the highest wealth group and zero being the lowest. Each rating has a different meaning within each state.
HV	Average Home Value in potential donor's neighborhood in $ hundreds
ICmed	Median Family Income in potential donor's neighborhood in $ hundreds
ICavg	Average Family Income in potential donor's neighborhood in hundreds
IC15	Percent earning less than 15K in potential donor's neighborhood
NUMPROM	Lifetime number of promotions received to date
RAMNTALL	Dollar amount of lifetime gifts to date
MAXRAMNT	Dollar amount of largest gift to date
LASTGIFT	Dollar amount of most recent gift
TOTALMONTHS	Number of months from last donation to July 1998 (the last time the case was updated)
TIMELAG	Number of months between first and second gift
AVGGIFT	Average dollar amount of gifts to date
TARGET_B	Target Variable: Binary Indicator for Response 1 = Donor, 0 = Non-donor
TARGET_D	Target Variable: Donation Amount (in $). We will NOT be using this variable for this case.

sampling, and calculate the net profit that would reflect the actual response distribution of 5.1% donors and 94.9% non-donors.)

4. **Draw Lift Curves:** Draw each model's net profit lift curve for the validation set onto a single graph. Are there any models that dominate?

5. **Best Model:** From your answer in 2, what do you think is the "best" model?

Step 3: Testing -

The file FutureFundraising.xls contains the attributes for future mailing candidates. Using your "best" model from Step 2 (#5), which of these candidates do you predict as donors and non-donors? List them in descending order of probability of being a donor.

13.6 Catalog Cross-Selling

Dataset: CatalogCrossSell.xls

Background

Exeter, Inc. is a catalog firm that sells products in a number of different catalogs that it owns. The catalogs number in the dozens, but fall into nine basic categories:

1. Clothing
2. Housewares
3. Health
4. Automotive
5. Personal electronics
6. Computers
7. Garden
8. Novelty gift
9. Jewelry

The costs of printing and distributing catalogs are high - by far the biggest cost of operation is the cost of promoting products to people who buy nothing. Having invested so much in the production of artwork and printing of catalogs, Exeter wants to take every opportunity to use them effectively. One such opportunity is in cross selling - once a customer has "taken the bait" and purchases one product, try to sell them another while you have their attention.

Such cross promotion might take the form of enclosing a catalog in the shipment of the purchased product, along with a discount coupon to induce a purchase from that catalog. Or it might take the form of a similar coupon sent by email, with a link to the web version of that catalog.

But which catalog should be enclosed in the box, or included as a link in the email with the discount coupon? Exeter would like it to be an informed choice - a catalog that has a higher probability of inducing a purchase than simply choosing a catalog at random.

Assignment

Using the dataset CatalogCrossSell.xls, perform an Association Rules analysis, and comment on the results. Your discussion should provide interpretations in English of the meanings of the various output statistics (lift ratio, confidence, support), and include a very rough estimate (precise calculations not necessary) of the extent to which this will help Exeter make an informed choice about which catalog to cross-promote to a purchaser.

Acknowledgment:

The data for this case are adapted from the data in set of cases provided for educational purposes by the Direct Marketing Education Foundation ("DMEF Academic Data Set Two, Multi Division Catalog Company, Code: 02DMEF")

13.7 Predicting Bankruptcy

Dataset: Bankruptcy.xls

(Use Darden Case - UVA-QA-0371 - Duxbury to secure permission)
Do not use the material in the last 2 paragraphs. Instead:

Assignment

1. What data mining technique(s) would be appropriate in assessing whether there are groups of variables that convey the same information, and how important that information is? Conduct such an analysis.

2. Comment on the distinct goals of profiling the characteristics of bankrupt firms, versus simply predicting (black box style) whether a firm will go bankrupt, and whether both goals, or only one, might be useful. Also comment on the classification methods that would be appropriate in each circumstance.

3. Explore the data to gain a preliminary understanding of which variables might be important in distinguishing bankrupt from non-bankrupt firms. (Hint - as part of this analysis, use XLMiner's boxplot option, specifying the bankrupt/not bankrupt variable as the x variable)

4. Using your choice of classifiers, use XLMiner to produce several models to predict whether a firm goes bankrupt or not, assessing model performance on a validation partition.

5. Based on the above, comment on which variables are important in classification, and discuss their effect.

Bibliography

[1] Agrawal, R., Imielinski, T. and Swami, A. (1993). "Mining associations between sets of items in massive databases," *in Proceedings of the 1993 ACM-SIGMOD International Coference on Management of Data* (pp. 207-216), New York: ACM Press

[2] Berry, M. J. A. & Linoff, G. S. (1997), *Data Mining Techniques*. New York: Wiley.

[3] Berry, M. J. A. & Linoff, G. S. (2000), *Mastering Data Mining*. New York: Wiley.

[4] Bishop, C. (1995). *Neural Networks for Pattern Recognition*, Oxford: Clarendon Press.

[5] Breiman, L., Friedman, J., Olshen, R., & Stone, C. (1984). *Classification and Regression Trees*, Boca Raton: Chapman & Hall/CRC (orig. published by Wadsworth)

[6] Chaterjee, S., Hadi, A., and Price, B. (2000). *Regression Analysis by Example* (3rd ed.), New York, Wiley and Sons.

[7] Delmaster, R., and Hancock, M. (2001), *Data Mining Explained*, Boston: Digital Press.

[8] Elston and Grizzle (1962). "Estimation of Time Response Curves and Their Confidence Bands," *Biometrics*, **18**, pp 148-159.

[9] Flury, B., and Flury, B. D. (1997). *A First Course in Multivariate Statistics*, New York: Springer Verlag.

[10] Green, P., Carmone, F., and Wachspress, D. (1997). "On the Analysis of Qualitative Data in Marketing Research," *Journal of Marketing Research*, **XIV**, pp 52-59.

[11] Han, J., and Kamber, M. (2001). *Data Mining: Concepts and Techniques*. Academic Press.

[12] Hand, D., Mannila, H. & Smyth, P. (2001), *Principles of Data Mining*. Cambridge, MA: The MIT Press.

[13] Hastie, T. and Stuetzle, W. (1989). "Principal Curves," *Journal of the American Statistical Association*, **84**, 502-516.

[14] Hastie, T., Tibshirani, R., & Friedman, J. (2001). *The Elements of Statistical Learning*. New York, Springer-Verlag.

[15] Heckerman, D. (1996). "Bayesian Networks for Knowledge Discovery" *in Advances in Knowledge Discovery and Data Mining*, Fayyad, Piatetsky-Shapiro, Smyth and Uthurusamy (ed.), Cambridge: MIT Press

[16] Hosmer, D. W., and Lemeshow, S. (2000). *Applied Logistic Regression*. 2nd edition. Wiley-Interscience.

[17] Johnson, W., and Wichern, D. (2002). *Applied Multivariate Statistics*, Upper Saddle River, NJ: Prentice Hall.

[18] Labe, R. P. (1994), "Database Marketing Increases Prospecting Effectiveness at Merrill Lynch," *Interfaces*, **24-5**, 1-12.

[19] Lyman, P. and Varian, H. R. (2003). "How Much Information". Retrieved from http://www.sims.berkeley.edu/how-much-info-2003 on Nov. 29, 2005

[20] Manski, C. F. (1977), "The Structure of Random Utility Models," *Theory and Decision*, **8**, pp. 229-254.

[21] McCullugh, C. E., Paal, B., and Ashdown, S. P. (1998) "An optimisation approach to apparel sizing", *Journal of the Operational Research Society,* **49**(5), pp. 492-499.

[22] Pregibon, D. (2001), " A Statistical Odyssey", Invited talk at *The Fifth ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*

[23] Russell P. G. (1994). "Increases Prospecting Effectiveness at Merrill Lynch," *Interfaces*, **24:5**, pp. 1-12.

[24] Simon, J. (1975), *Applied Managerial Economics*, Englewood Cliffs, NJ: Prentice Hall.

[25] Trippi, R. and Turban, E. (ed.) (1996). *Neural Networks in Finance and Investing*, New York: McGraw Hill.

Index