List of Tables

Preface

In writing this book we have three primary objectives. First, we want to provide the reader with an understanding of the types of business problems that advanced analytical tools can address and to provide some insight into the challenges that organizations face in taking profitable advantage of these tools.

Our second objective is to give the reader an intuitive understanding of how different data mining algorithms work. This discussion is largely non-mathematical in nature. However, in places where we think the mathematics is an important aid to intuitive understanding (such as is the case with logistic regression), we provide and explain the underlying mathematics. Given the proper motivation, we think that many readers will find the mathematics to be less intimidating than they might have first thought, and find it useful in making the tools much less of a "black box."

The book's final primary objective is to provide the reader with a readily available "hands-on" experience with data mining tools. When we first started teaching the courses this book is based on (in the late 1990s), there were not many books on business and customer analytics, and the books that were available did not take a hands-on approach. In fairness, given the license costs of user-friendly data mining tools at that time (and commercial software products up to the present day), writing such a book was simply not possible. We both are firm believers in the "learning by doing" principal, and this book reflects this. In addition to hands-on use of software, and the application of that software to data that address the types of problems real organizations face, we have also made an effort to inform the reader of the issues that are likely to creep up in applied data mining projects, and present the CRISP-DM process model as a practical framework for organizing these projects.

This book is intended for two different audiences, but who we think have similar needs. The most obvious is students (and their instructors) in MBA and advanced undergraduate courses in customer and business analytics and applied data mining. Perhaps less apparent are individuals in small- to medium-size organizations (both businesses and not-for-profits) who want to use data mining tools to go beyond database reporting and OLAP tools in order to improve the performance of their organizations. These individuals may have job titles related to marketing, business development, fund raising, or IT, but all see potential benefits in bringing improved analytics capabilities to their

organizations. We have come in contact with many people who helped bring the use of analytics to their organizations. A common theme that emerged from our conversations with these individuals is that the first applications of customer and business analytics by an organization are typically skunkworks projects, with little or no budget, and carried out by an individual or a very small team of people using a learn-as-you-go approach. The high cost of easy-to-use commercial data mining tools (a project that requires multiple thousands of dollars per seat software licenses is no longer a skunkworks project) and a lack of appropriate training materials are often major impediments to these projects. Instead, many of these projects are based on experiments that push Excel beyond its useful limits. This book, and its accompanying R-based software (R Development Core Team, 2011), provides individuals in small and medium-sized organizations with the skills and tools needed to successfully, and less painfully, start to develop an advanced analytics capability within their organizations.

The genesis of this book was an applied MBA-level business data mining course given by Dan Putler at the University of British Columbia that was offered on an experimental basis in the spring term of the 1998–1999 academic year. One of the goals of the experimental course was to determine if the nature of the material would overwhelm MBA students. The course was project based (with the University's Development organization being the first client), and used commercial data mining software from a major vendor, along with the training materials developed by that vendor. The experiment was considered a success, so the following year the course became a regular course at UBC, and partially based on Dan's original materials, Bob Krider developed a similar course at Simon Fraser University for both MBA and undergraduate business students.

We soon decided that the vendor's training materials did not fully meet the needs of the course, and we began to jointly develop a full set of our own tutorials for the vendor's software that better met the course's needs. While our custom tutorials were a major improvement, we soon felt the need to use tools based on R, the widely used open source and free statistical software. There were several reasons for this. First, the process of students moving out of computer labs and onto their own laptops to do computer-oriented coursework was well under way, and the ability of our students to install the commercial software on their own machines suffered from both licensing and practical limitations. Second, our experience was that students often questioned the value of the time spent learning expensive, specialized software tools as part of a class since many of them believed, correctly, that their future employers would not have licenses for the tools, and they themselves would not have the funds to procure the needed software. These concerns are greatly reduced

through the use of mature, open-source tools, since students know the tools will be readily available for free in the future. Third, as we discuss above, we wanted a means by which to meet the needs and financial constraints of individuals in small and medium-size organizations who want to experiment with the use of analytics in their own organizations. Finally, we, like many other academic researchers, were using R to conduct our research (which is robust, powerful, and flexible), and knew it was only a matter of time before R would extensively be used in industry as well, a process that is now well on its way.

While we do our research using R in the "traditional way" (i.e., using the R console's command line interface to issue commands, run script files, and conduct exploratory analyses), a command line interface is a hard sell to most business school students and to individuals in organizations who are interested in learning about and experimenting with data mining tools. Fortunately, at the time we were thinking about moving to R for our courses, John Fox (2005) had recently released the R Commander package, which was intended to be a basic instructional graphical user interface (GUI) for R. This became the basis of the R-based software tools used in this book. Originally we developed a custom version of the R Commander that included functionality needed for data mining, and we contributed a number of functions back to the original R Commander package that were consistent with John's goal of creating a basic instructional GUI for statistics education. Since its introduction, the flexibility of the R Commander package has greatly increased, and it now has an excellent plug-in architecture that allows for very customized tool sets, such as the RcmdrPlugin.BCA package that contains the software tools used for this book.

In addition to John Fox, there are a number of other people we would like to thank. First we would like to thank multiple years of students at the University of British Columbia, Simon Fraser University, and City University of Hong Kong who used draft chapters of the book in courses taught by us and our Simon Fraser University colleague Jason Ho. The students pointed out areas where explanations needed to be clearer, where the tutorials were not exactly right, and a very long list of typographical errors. Their input over the years has been extremely important in shaping this book. Nicu Gandilathe (BCAA) and Matt Johnson (Intrawest) gave us valuable input about how to make the book and the software more useful to customer and business analytics practitioners. We have greatly benefited from conversations and advice given by our colleagues John Claxton (UBC), Maureen Fizzell (SFU), Andrew Gemino (SFU), Ward Hanson (Stanford), Kirthi Kalyanam (Santa Clara University), Geoff Poitras (SFU), Chuck Weinberg (UBC), and Judy Zaichowski (SFU) on both the content of the book and the process of getting

a book published. Our editor at CRC Press, Randi Cohen, has been a pl
pleasure to work with, quickly addressing any questions we have had, a
making every effort to help us when we needed help. We also want to than
Doug MacLachlan (University of Washington) for his review of draft versio
of this manuscript; he has helped to keep us honest. Lastly, and perhaps mos
important, we want to thank both of our families, especially our wives, Lis
Blaney and Clair Krider, for the patience and support they have shown u
while writing this book, including Dan's dad, who kept the pressure on by
frequently asking when the book would be finished.

Daniel S. Putler, Sunnyvale, CA, US

Robert E. Krider, Burnaby, BC, Canad

Part I

Purpose and Process

Chapter 1

Database Marketing and Data Mining

As recently as the early 1970s, most organizations either had little information about their interactions with customers or little ability to access (short of physically examining the contents of paper file folders) and act upon what information they did have for marketing purposes. The intervening 40 years has seen an ongoing revolution in the information systems used by companies. The lowering of computing and data storage costs have been the driving force behind this, making it economically feasible for firms to implement transactional databases, data warehouses, customer relationship management systems, point of sales systems, and the other software and technology tools needed to gather and manage customer information. In addition, a large number of firms have created loyalty and other programs that their customers gladly opt into that, in turn, allows these firms to track the actions of individual customers in a way that would otherwise not be possible.

While falling computing costs and software advances allowed companies to develop increasingly sophisticated databases containing information about their interactions with their own customers, third-party data suppliers have taken advantage of the same information technology advances to collect additional information about those same customers, along with information on potential new customers, using data from credit reporting services, public records, the census, and other sources. As a result, companies now have the potential to prospect for new customers by finding individuals and organizations that are similar in important respects to their existing customers.

Realizing the potential of this newly available customer information has been a challenge to many organizations. While even small organizations now have the ability to develop extensive customer databases, up to now, only a fairly small number of comparatively large organizations have been able to take full advantage of the extensive information assets available to them. To do this, these firms have invested in analytical capabilities, particularly data mining, to develop managerially useful information and insights from the large amounts of raw data available.

The benefits of using these analytical tools are both practical/tactical and strategic in nature. From a practical/tactical perspective, the use of data mining tools can greatly reduce costs by better targeting existing customers,

minimizing losses due to fraud, and more accurately qualifying potential new customers. In addition to lowering marketing costs, these tools can assist in both maintaining and increasing revenues through helping to obtain new customers, and in holding on (and selling more) to existing customers.

From a strategic point of view, organizations are increasingly viewing the development of the analytical capabilities needed to make the most of their data as a long-run competitive advantage. As Thomas Davenport (2006) writes in the *Harvard Business Review*:

> Most companies in most industries have excellent reasons to pursue strategies shaped by analytics. Virtually all the organizations we identified as aggressive analytics competitors are clear leaders in their fields, and they attribute much of their success to the masterful exploitation of data. Rising global competition intensifies the need for this sort of proficiency. Western companies unable to beat their Indian or Chinese competitors on product cost, for example, can seek the upper hand through optimized business processes.

The goal of this book is to provide you, the reader, with both a better understanding of what these analytical tools are and the ability to apply these tools to your own business, particularly as it relates to the marketing function of that business. To start this process, this chapter provides an overview of both database marketing and the data mining tools needed to implement effective database marketing programs.

1.1 Database Marketing

The fundamental requirement for any database marketing program is the development and maintenance of a customer database. In their book *The One to One Future*, Peppers and Rogers (1993) provide the following definition of a customer database:

> A ***Customer Database*** is an organized collection of comprehensive data about individual customers or prospects that is current, accessible, and actionable for such marketing purposes as lead generation, lead qualification, sale of a product or service, or maintenance of customer relationships.

In turn, Peppers and Rogers (1993) define database marketing in the following way:

Database Marketing is the process of building, maintaining, and using customer databases and other databases for the purposes of contacting and transacting.

1.1.1 Common Database Marketing Applications

The above definitions provide a useful starting point, but are a bit abstract. Looking at the most common types of database marketing applications should help make things clearer. Database marketing applications can be placed into three broad categories: (1) selling products and services to new customers; (2) selling additional products and services to existing customers; and (3) monitoring and maintaining existing customer relationships. The two most common types of applications designed to assist in the selling of products and services to new customers are "prospecting" for (i.e., finding) new customers, and qualifying (through activities such as credit scoring) those potential new customers once they have been found.

Database marketing applications designed to sell more to existing customers include cross-selling, up-selling, market basket analysis, and recommendation systems. Cross-selling involves targeting a current customer in order to sell a product or service to that customer that is different from the products or services that customer has previously purchased from the organization. An example of this is a telephone service provider who targets an offer for a DSL subscription package to a customer who currently only purchases residential land line phone service from that provider. In contrast, up-selling involves targeting an offer to an existing customer to upgrade the product or service he or she is currently purchasing from an organization. For instance, a life insurance company that targets one of its current term life insurance policy holders in an effort to move that customer to a whole life policy would be engaged in an up-selling activity.

Market basket analysis involves examining the composition of items in customers' "baskets" on single purchase occasions. Given its nature, market basket analysis is most applicable to retailers, particularly traditional brick and mortar retailers. The goal of the analysis is to find merchandising opportunities that could lead to additional product sales. In particular, a supermarket retailer may find that people who buy fresh fish on a purchase occasion are disproportionately likely to purchase white wine as well. As a result of this finding, the retailer might experiment with placing a display rack of white wine adjacent to the fresh fish counter to determine whether this co-location of products increases sales of white wine, fresh fish, or both.

Common applications designed to monitor and improve customer relationships include customer attrition (or "churn") analysis, customer segmentation, rec-

ommendation systems, and fraud detection. The goal of churn analysis is to find patterns in a current customer's purchase and/or complaint behavior that suggests that the customer is about to become an ex-customer. Knowing whether a profitable customer is at risk of leaving allows the organization to proactively communicate with the customer in order to present a promotional offer or address the customer's concerns in an effort to keep that customers business. Alternatively, a company may avoid taking actions that would encourage an unprofitable customer to remain with the firm. Grouping customers into segments based on their past purchase behavior allows the organization to develop customized promotions and communications for each segment, while recommendation systems, such as the one used by Amazon.com, group products based on which customers have bought them, and then makes recommendations based on the overlap of the buyers of two or more products. Fraud detection allows an organization to uncover customers who are engaged in fraudulent behavior against them. For instance, a consumer package goods company may use data on manufacturer's coupon redemptions on the part of different retail trade accounts in order to develop a model that would flag a particular retail account as being in need of further investigation to determine whether that retailer is fraudulently redeeming bogus coupons that were not actually redeemed by final consumers.

Two Examples

To get a sense of how organizations use database marketing in practice, we examine two different database marketing efforts. The first is an application designed to prospect for new customers, while the second deals with two related projects designed to reduce customer churn. One thing that is common to both these applications is that there are substantial savings in marketing costs (that more than cover the analysis costs) from not conducting blanket promotions.

Keystone Financial

In his article "Digging up Dollars with Data Mining—An Executive's Guide," Tim Graettinger (Graettinger, 1999; Kelly, 2003) describes a database marketing project undertaken by Pennsylvania-based Keystone Financial Bank, a regional bank. Keystone developed a promotional product called LoanCheck with the intention of using it to expand its customer base (a prospecting application). LoanCheck consisted of a $5,000 "check" that could be "cashed" by the recipient at any Keystone Financial Bank branch to initiate a $5,000 loan. To determine which potential new customers Keystone should target with this product, Keystone mailed a LoanCheck offer to its existing customers. Information on which of its existing customers took advantage of the LoanCheck offer was appended to Keystone's customer database. The customer database

was then used to determine the characteristics of customers most likely to respond favorably to the LoanCheck offer using data mining methods, resulting in the creation of a model that predicted the relative likelihood that a customer would respond favorably to the LoanCheck offer. Keystone then applied this model to a database of 400,000 potential new customers it obtained from a credit reporting agency, and then mailed the LoanCheck offer to the set of individuals in that database the model predicted would be most likely to respond favorably to the LoanCheck offer. This database marketing project resulted in Keystone obtaining 12,000 new customers, and earning $1.6M in new revenues.

Verizon Wireless

At the 2003 Teradata Partners User Group Conference and Expo, Ksenija Krunic, head of data mining at Verizon Wireless (a major U.S. mobile phone service provider), described how her company used two related database marketing projects to decrease Verizon Wireless's churn rate for individual customers by one-quarter compared to what it had been (Das, 2003). Specifically, in the first project, Verizon used its customer databases in order to develop a model to predict which of its customers were most likely to defect to another provider at the expiration of their current contract based on the current plan a customer had, a customer's historical calling patterns, and the number and type of service requests made by a customer. The second project involved using the model developed in the first project to create samples of customers likely to leave Verizon at the end of their current contract, and then offer each of these samples a different experimental new plan offer, tracking which customers in each segment accepted the offer (thereby resulting in a contract renewal with Verizon). The data generated from these experimental samples (which consisted of whether a customer took the service and the terms of the offered plan) were combined with the customer calling pattern and service request data to create a second set of models which, together, allow Verizon to determine the best new plan offer to make to a customer who is likely to leave Verizon at the end of his or her contract, before the current contract expires, including not making an offer at all. Using these models, Verizon Wireless was able to decrease its attrition rate from 2 percent per month to 1.5 percent per month (a reduction of 25 percent from the original attrition rate). Given that the cost of acquiring a customer in the mobile phone industry is estimated to be between $320 and $360, the drop in the attrition rate has had a huge impact on Verizon Wireless's bottom line. Verizon has 34.6 million subscribers, so the value of the reduction in churn is roughly $700M per year. In addition, since the promotional mailings are now highly targeted, the company's direct mail budget for "churner mailings" fell 60 percent from what it was prior to the completion of these two related database marketing projects.

1.1.2 Obstacles to Implementing a Database Marketing Program

As the above two examples indicate, the potential rewards from implementing database marketing programs can be enormous. Unfortunately, there are a number of obstacles that can make implementing these programs difficult. First, the data issues can be complex. Specifically, IT systems and tools (such as data warehouses and customer relationship management systems) need to be in place to collect the needed data, clean the data, and integrate data that can come from a large number of different computer systems, databases, and Excel spreadsheets. Second, the data mining tools themselves can be complex since they are based on a combination of advanced statistical and machine learning tools. Finally, the available talent that can be hired who "can do it all" in terms of understanding both the analytical tools and the business problems is scarce. As Davenport (2006) writes: "Analytical talent may be to the early 2000s what programming talent was to the late 1990s. Unfortunately, the U.S. and European labor markets aren't exactly teaming with analytically sophisticated job candidates."

While these three obstacles are not insurmountable, it can take a considerable amount of time and effort to overcome them. The experience of Barclays Bank, as described by Davenport (2006), illustrates this point:

> The UK Consumer Cards and Loans business within Barclays bank, for example, spent five years executing its plan to apply analytics to the marketing of credit cards and other financial products. The company had to make process changes in virtually every aspect of its consumer business: underwriting risk, setting credit limits, servicing accounts, controlling fraud, cross selling, and so on. On the technical side, it had to integrate data on 10 million Barclaycard customers, improve the quality of the data, and build systems to step up data collection and analysis. In addition, the company embarked on a long series of small tests to begin learning how to attract and retain the best customers at the lowest price. And it had to hire new people with top-drawer quantitative skills.

Despite the obstacles, the use of data mining–based database marketing continues to grow. Evidence of this is that the dollar sales of the software tools needed to implement this type of analysis grew 11.5 percent in 2005 over 2004 levels, and industry forecasts made by IDC (Vesset and McDonough, 2006) indicate that this rate of growth will be maintained for the foreseeable future.

1.1.3 Who Stands to Benefit the Most from the Use of Database Marketing?

While most organizations can obtain some benefit from the use of database marketing tools, some will receive substantially greater benefits than others. Three factors are particularly important in driving the returns to database marketing programs: (1) the organization has a large number of customers; (2) customer transaction data can be obtained either as a byproduct of normal operations or through the use of a device, such as a customer loyalty program, by the organization; and (3) the acquisition and/or loss of a customer is expensive to the organization.

Given the nature of these three factors, it is unsurprising that certain industries have emerged as leaders in implementing database marketing programs. These leading industries include (1) telecommunications; (2) banking, insurance, and financial service providers; (3) catalog and online retailers; (4) traditional retailers; (5) airlines, hotel chains, and other travel industry players; and (6) charities, educational institutions, and other not-for-profits.

1.2 Data Mining

As the examples in the previous section indicate, the underlying technology driving database marketing efforts is data mining. In this section we provide an overview of data mining by first providing two definitions of what it is, and then briefly describing commonly used data mining methods. Our descriptions of methods in this chapter are brief (at times almost non-existent) since the balance of the remainder of this book covers these methods in much greater detail.

1.2.1 Two Definitions of Data Mining

Data mining really has two different intellectual roots, statistics and the database and machine leaning fields of computer science. Because of this twin heritage, a large number of different definitions of data mining have been put forward. Probably the most widely used definition of data mining comes from The Gartner Group (Krivda, 1996):

> ***Data mining*** is the process of discovering meaningful new correlations, patterns, and trends by sifting through large amounts of

> data stored in repositories and by using pattern recognition technologies as well as statistical and mathematical techniques.

This definition of data mining flows more from the database and machine learning tradition. In this tradition, data mining is also referred to as "knowledge discovery in databases" or KDD. A common theme in this tradition is that the application of data mining methods to data will reveal new, heretofore unknown patterns that can then be constructively taken advantage of. This world view is in marked contrast to the one of traditional statistics, where patterns are hypothesized to exist a priori, and then statistical methods are used to test whether the hypothesized patterns are supported by the data.

To reveal our bias, we lean toward the statistics world view. The machine learning world view strikes us as being a bit too "auto-magical" for our tastes. Moreover, given our econometrics-oriented training and backgrounds, we are concerned about both spurious correlation and attempting to gain additional insight by understanding the drivers of customer behavior. As a result, we place a lot of emphasis on modeling behavior as a means of predicting it. Given this orientation, the definition of data mining we use is:

> ***Data mining*** is the process of using software tools and *models* to summarize large amounts of data in a way that supports decision-making.

The critical difference in our definition is its focus on models and modeling. We view modeling as the human process of simplifying a complex real world situation by abstracting essential elements. Properly done, modeling improves our understanding, our ability to communicate, and our decision-making.

1.2.2 Classes of Data Mining Methods

Ultimately, data mining uses a set of methods that originated in either statistics or machine learning to summarize the available data. These different methods fall into two broad classes, grouping methods and predictive modeling methods. Within each of these two classes fall literally hundreds of different specific methods (also known as algorithms). In this section we will only mention the most commonly used methods for each of the two classes. We will present these methods in more detail later in the book.

1.2.2.1 Grouping Methods

Grouping methods used in database marketing can be categorized as falling into two distinct types: methods used to group products and services, and

methods used to group customers. The most commonly used method to group products and services is known as *association rules*. Association rules come from machine learning, and examine the co-occurrence of different objects (say the purchase of fresh fish and white wine by customers on the same shopping occasion) and then form a set of "rules" that describe the nature of the most common co-occurrence relationships among objects in a database.

Two methods are commonly used to group customers. The most widely applied is cluster analysis, which is a term used to describe a set of related methods that were developed in statistics (some of the methods date to the 1930s). The most common method of cluster analysis used in data mining is known as *K-Means*. K-Means is one of several "partitioning methods" for cluster analysis that have been developed. K-Means is called a partitioning method since it finds the "best" (using a Euclidean distance-based measure) division of the data into K partitions, where K is the number of partitions specified by the analyst. The other commonly used methods of cluster analysis are known as hierarchical agglomerative methods (Wards method, average linkage, and complete linkage are the most commonly used hierarchical agglomerative methods). Hierarchical agglomerative methods are not typically used in data mining because they do not scale to the number of records often encountered in database marketing applications. However, these methods are well suited to the number of records typically used in sample survey–based marketing research applications.

The second method commonly used to group customers is known as *self-organizing maps* (also called Kohonen maps, after the inventor of the method, Finnish computer scientist Teuvo Kohonen). Euclidean distance is also used as the basis of grouping records in this method. However, how these distances are used is very different across the two methods. K-Means attempts to minimize the sum of the squared Euclidean distances for members within a group, while self-organizing maps use the distances as part of a neural network algorithm. One drawback to both of these methods is that the variables used to group records must be continuous, so categorical variables (such as zip or postal code) cannot be used to group customers. However, there are other clustering methods (such as ROCK clustering; Guha et al. (2000)) that can cluster a set of categorical variables.

1.2.2.2 Predictive Modeling Methods

Three types of methods are commonly used to construct predictive models in data mining: (1) linear and logistic regression; (2) decision trees; and (3) artificial neural networks. Consistent with the class name, the goal of all three methods is to predict a variable of interest. The variable can be either continuous (e.g., total sales of a particular product in the next quarter) or categorical

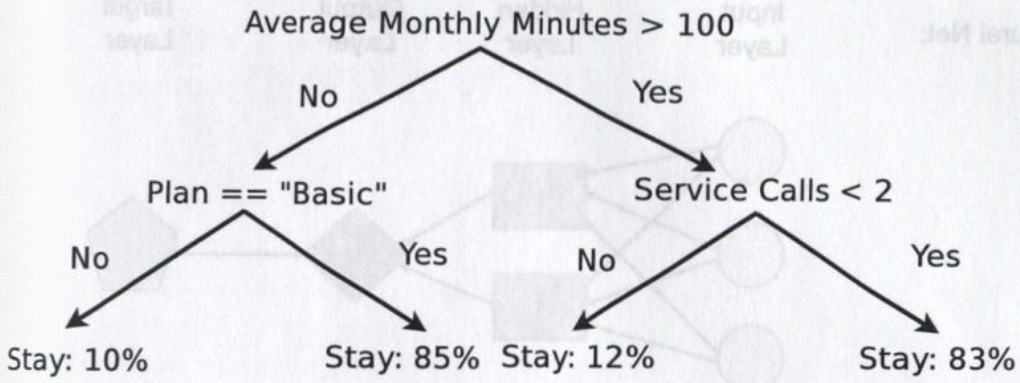
(e.g., whether a customer will respond favorably to a particular direct mail offer) in nature. In the case of a continuous variable, what is predicted is the expected value of that variable (e.g., expected total sales of the product in the next quarter), while in the case of a categorical variable, what is predicted is the probability that the variable will fall into each of the possible categories (e.g., the probability a customer will respond favorably to the direct mail offer).

Linear and logistic regression are two of the most important tools of traditional statistical inference. Both methods use a weighted sum of an analyst-specified set of predictor variables (known as a "linear predictor") to come up with a predicted value. Where the two methods differ is in how this linear predictor is transformed in order to make a prediction. In the case of linear regression, the linear predictor constitutes the prediction, while in logistic regression the linear predictor is transformed in a way such that the predicted probability for each possible category of the categorical variables of interest falls between zero and one, and the sum of the probabilities across the different categories equals one. Both a plus and minus of linear and logistic regression is that the analyst plays a central role in creating a model. The plus to this is that the implied customer behavior underlying a model can be more easily seen, so it is easier for managers to interpret, critique, and learn from that model. The minus is that the quality of a model is closely tied to the skill level of the analyst who created it.

Decision tree methods have origins in both statistics and machine learning. While a number of different algorithms have been proposed (and are commonly used) to create a decision trees, all methods create a set of "if-then" rules leading to a set of final values for the variable being predicted. These final values can be either probabilities for a categorical variable (in which case the tree that is created is known as a "classification tree") or quantities for a continuous variable (where the resulting tree is called a "regression tree"). To give a better sense of what a decision tree looks like, Figure 1.1 shows a hypothetical classification tree of a churn analysis for a mobile telephone service provider.

The example classification tree starts at its "root" with a split on whether the customer had more or less than 100 calling minutes on average each month. If the answer to this question is no, we move to the next "node" where the split is determined based on whether the customer has a subscription to the "Basic" plan. If the answer to this question is yes, then the probability the customer will stay is 85 percent (or a 15 percent probability of leaving), while the probability of a customer staying with the company is only 10 percent if that customer had less than 100 calling minutes per month on average and the customer had a contract for something other than the basic plan. If the

Figure 1.1: An Example Classification Tree

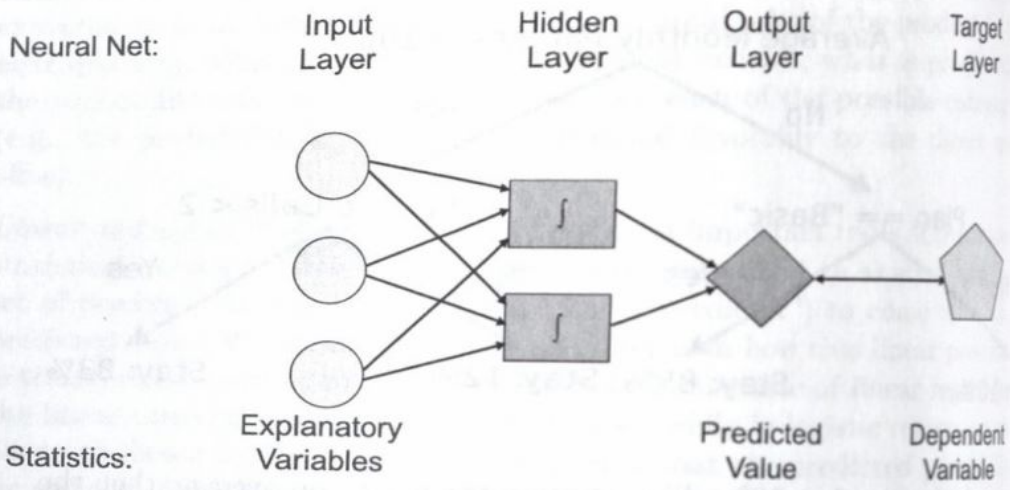


customer had more than 100 calling minutes per month on average, then the second node in the tree is the number of service calls the customer made. Each element at the bottom level of the tree that indicates the probability of staying or leaving the service provider is called a "leaf."

One advantage of decision trees is that most people find their "if-then" structure to be both easy to understand and to act upon. Another advantage is that less skilled analysts will get results similar to those of more skilled analysts since all the variables in the database can be used as predictors in a decision tree (the decision tree algorithm will determine which to include in the tree), and the algorithm automatically "transforms" the relevant variables via the splitting rules. However, decision trees also have a number of disadvantages as well, which we explore later.

An *artificial neural network* is a predictive modeling method developed in machine learning that is based on a simplified version of the brain's neurological structures. Figure 1.2 provides an illustration of a simple neural network. However, explaining even this simple example is fairly involved, so we will refrain from doing so now. The three important things to know at this point are that: (1) neural network models, like decision trees model, are less dependent on the skill of the analyst in developing a good model relative to linear and logistic regression; (2) neural network models are very flexible in terms of the shapes of relationships they can mimic, but this turns out to be something of a mixed blessing; and (3) neural network models are very hard to interpret in a managerially meaningful way, so they amount to "black boxes" that can predict well but provide no insights into underlying customer behavior.

Figure 1.2: An Example Neural Network



1.3 Linking Methods to Marketing Applications

It will probably come as no surprise that there is a strong relationship between the type of database marketing application being undertaken and the class of the data mining method that should be used for that application. Table 1.1 provides a table that relates each common type of database marketing application to the appropriate class of data mining method to use for that application. An examination of the table reveals that the only application type where it could make sense to use both grouping and predictive modeling methods is fraud detection, although, for this application, we expect that predictive modeling methods will be a superior choice to grouping methods. However, it may be possible to group customers in such a way that one of the groups formed has a higher incidence of fraudulent behavior, making grouping methods potentially useful.

Now that you have a better sense of what both database marketing is, and how data mining tools are used in database marketing programs, you are ready to learn more details about the process of implementing a data mining project as part of a database marketing program.

Table 1.1: Linking Marketing Applications with Data Mining Methods

	Data Mining Method Class	
Marketing Application	Grouping	Predictive Modeling
Prospecting		√
Prospect qualifying		√
Cross-selling		√
Up-selling		√
Market basket analysis	√	
Recommendation systems	√	
Attrition and churn		√
Fraud detection	√	√
Customer segmentation	√	

Chapter 2

A Process Model for Data Mining—CRISP-DM

In this chapter we examine a process model for data mining called CRISP-DM. CRISP-DM is an acronym for the CRoss-Industry Standard Process for Data Mining. CRISP-DM has become something of a de facto standard for organizing and conducting data mining projects. The document that describes CRISP-DM in detail is *CRISP-DM 1.0: Step-By-Step Data Mining Guide* (Chapman et al., 2000), which is published by the CRISP-DM consortium, and can be freely downloaded from the CRISP-DM Consortium's web site (www.crisp-dm.org or from this book's web site, www.customeranalyticsbook.com). Our goal is a condensed, somewhat paraphrased, overview of the CRISP-DM 1.0 model. Before doing this, however, we provide some historical context on data mining process models in general and CRISP-DM in particular.

2.1 History and Background

Business-oriented data mining only started to become something of an organized field in the early to mid-1990s. Very quickly, organizations developing internal database marketing and data mining capabilities, data mining consultants, and software vendors selling data mining tools came to realize there was a need to systematically organize the process of data mining. To this end, Barry and Linoff (1997) in their influential book *Data Mining Techniques for Marketing, Sales and Customer Relationship Management* presented a high-level model of the data mining process that they dubbed "the virtuous cycle of data mining." At roughly the same time, software vendors presented process guides for data mining that covered steps from the point at which data were loaded into data mining software tools through to the early stages of the deployment of a data mining–based solution. The best known of these process guides is SAS's SEMMA process (SEMMA is an acronym that stands for sample, explore, modify, model, and assess).

The broadest-based effort to develop a process model for data mining was started in 1996 through a joint effort of Daimler-Benz, ISL (which was later acquired by SPSS, which was, in turn, was recently acquired by IBM), and NCR. Daimler-Benz was an early adopter of data mining methods to address business problems; ISL developed the first data mining software workbench (Clementine, first marketed in 1994); and NCR's interest was based on its desire to add value to its Teradata data warehouse products, and it had entered the data mining consulting field in an effort to help accomplish this. The goal of this initial group was to help new adopters of data mining methods to avoid a long period of trial-and-error learning in implementing these methods to solve business problems. The group believed that the best way to do this was to develop a standard process model that would be non-proprietary, freely available, and data mining software tool neutral.

In 1997, partially supported with funding from the European Commission, the group (along with OHRA Verzekeringen en Bank Groep, a Dutch banking firm) founded a consortium under the CRISP-DM banner. An important part of the consortium's efforts was the creation of a special interest group (or SIG) that allowed for a broader set of individuals and organizations to play a role in the development of the CRISP-DM process model. Drafts of the CRISP-DM model were available as early as 1999, and the consortium released a final version (version 1.0) of the CRISP-DM model in 2000. At the time of this writing (September 2009), efforts are under way to develop version 2 of CRISP-DM.

The major competitor of the CRISP-DM process model is a fusing of Barry and Linoff's (Barry and Linoff, 1997) virtuous cycle of data mining with SAS's SEMMA approach for conducting the actual data analysis. The fusion of the virtuous cycle and SEMMA (which is now much more explicitly done as a result of Barry and Linoff's close ties with SAS) results in a process model that is fairly similar to CRISP-DM. However, CRISP-DM goes into aspects of conducting a data mining project beyond the process model itself. Put another way, the virtuous cycle model fused with SEMMA gives you the guidance you need to conduct a data mining project using best practices, while CRISP-DM does this, plus gives the user a head start in laying out the tasks and milestones in a Gantt chart for a particular project.

2.2 The Basic Structure of CRISP-DM

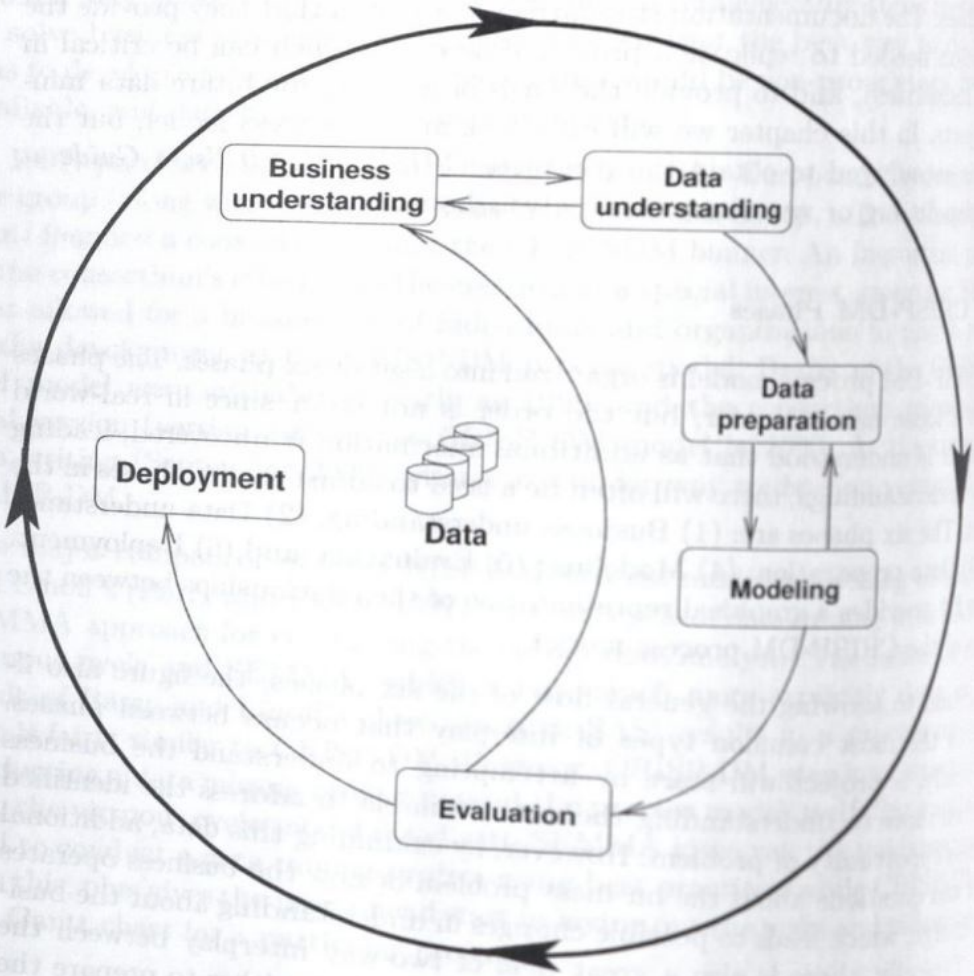
CRISP-DM really provides three different things: (1) a step-by-step "blueprint" for conducting a data mining project (the tasks of a Gantt chart); (2) a specified set of "deliverables" for each phase of a project (the milestones of a Gantt chart); and (3) a set of documentation standards in terms of what information should be included in each report that is in the set of project deliverables. The documentation standards are useful in that they provide the information needed to replicate a project if need be (which can be critical in certain industries), and to provide the basis of learning for future data mining projects. In this chapter we will only look at the process model, but the reader is encouraged to obtain the complete *CRISP-DM 1.0 Users Guide* at www.crisp-dm.org or www.customeranalyticsbook.com.

2.2.1 CRISP-DM Phases

The CRISP-DM process model is organized into a set of six phases. The phases fit into a loose natural order, but the order is not strict since in real-world projects it is understood that as additional information is uncovered, leading to new understandings, there will often be a need to adjust earlier phases in the project. The six phases are: (1) Business understanding; (2) Data understanding; (3) Data preparation; (4) Modeling; (5) Evaluation; and (6) Deployment. Figure 2.1 provides a graphical representation of the relationships between the phases of the CRISP-DM process model.

In addition to showing the general flow of the six phases, the figure also illustrates the most common types of interplay that occurs between phases. Specifically, a project will start by attempting to understand the business and then move to understanding the data available to address the identified business opportunity or problem. However, by examining this data, additional issues and questions about the business problem or how the business operates may emerge, which leads to possible changes in understanding about the business. Typically, there is also a great deal of two-way interplay between the data preparation and the modeling phases of a project, either to prepare the data in a way so that it suitable for use with a different modeling method than was originally planned, or to transform existing variables and/or derive new variables from existing variables that initial modeling suggests may be useful in predicting the behavior under investigation. Finally, it is not uncommon for a data mining project to provide new information and understanding, or new questions, about the business. This new understanding or the new questions

Figure 2.1: Phases of the CRISP-DM Process Model



raised about the business may result in a rethinking about the business either for this project, or, more likely, may suggest other business opportunities or problems that management may want to address.

2.2.2 The Process Model within a Phase

Within each phase, the structure of the CRISP-DM process model is hierarchical in nature. At the top level of the hierarchy is the phase itself (e.g., business understanding). Underneath the phase will be several *generic tasks* (the number of generic tasks differs across phases). As the name suggests, generic tasks are high-level descriptions of activities that need to be carried out for nearly all projects. Each of these generic tasks is then made project specific through the specification of one or more *specialized tasks*, which fall under the generic task at the third level of the hierarchy. The specialized tasks should describe the specific activities that need to be undertaken to accomplish a generic task. For example, one of the generic tasks under the business understanding phase is to determine business objectives. Specialized tasks to address this generic task might include: (1) Review the annual marketing plan for product X for each of the past three years; (2) Schedule and conduct an interview with Amy Ng, Executive Vice President of Marketing; and (3) Schedule and conduct an interview with Gordon Stewart, product manager for product X. At the lowest level of the hierarchy are the *process instances* that specify the deliverable(s) associated with each task. In the example given above, process instances would include items such as a transcript and summary for each of the two interviews, a summary of product X's performance and historical objectives based on the review of product X's past marketing plans, and a listing of the current objectives for product X to address the generic task.

2.2.3 The CRISP-DM Phases in More Detail

In this section we describe in greater detail the six phases of the CRISP-DM process model and give a list of generic tasks for each phase. While additional detail is provided, the details given are far from complete. As stated earlier, we encourage you to obtain a complete copy of the *CRISP-DM 1.0 Users Guide.*

2.2.3.1 Business Understanding

The purpose of the business understanding phase is to develop a set of project objectives and requirements from a business perspective, and then converting those objectives and requirements into a definition of the data mining problem to be addressed, and creating a preliminary plan designed to address that problem.

The generic tasks of this phase are: (1) determine business objectives; (2) assess the situation; (3) determine data mining goals; and (4) produce a project plan. The point of the first generic task is to determine what the organization really wants to accomplish. Often there will be a number of competing objectives and constraints that will need to be uncovered and then assessed in order determine how to make appropriate trade-offs. Not doing this may lead you to provide the right answers to the wrong questions.

The *situation assessment task* requires additional fact-finding to determine the resources, constraints, assumptions, and other factors that potentially influence both the data mining goal and the project plan. Determining the data mining goal involves translating the business objectives into a set of data mining project goals in technical terms. For instance, the business goal of Verizon Wireless in the example given in the last chapter was to increase the renewal rate (reduce churn) of customers whose contracts were expiring, which leads to two related data mining questions. First, which customers were most likely to leave Verizon at the end of their current contracts? Second, how likely was a customer with a particular profile to accept an offer for a particular new plan that Verizon could offer in order to keep that customer's business?

The *project plan*, which is really the end result of this phase, should be comprehensive and detailed. In particular, the plan should specify the anticipated steps, in order, along with potential dependencies. The plan should also specify all inputs and outputs. The creation of this plan is likely to benefit from the use of formal project planning tools such as Gantt charts.

2.2.3.2 Data Understanding

The data understanding phase begins with determining what data are currently available to the organization, whether permission can be granted to use this data for data mining, if there are any restrictions on the use of the data (such as privacy concerns), and what data are relevant to addressing the data mining problem. In addition, there needs to be a determination of whether there are any applicable third-party data available (such as customer credit history data), and the details (such as cost, data structure, data format, etc.) concerning applicable third-party data.

The next part of this phase involves gathering the appropriate data, and then exploring the data to get an understanding of it, identifying any quality problems, and using simple descriptive statistics and visual displays in order to develop an initial understanding that will be of use in the modeling phase.

There are four generic tasks associated with the data understanding phase: (1) collect initial data; (2) describe data; (3) explore data; and (4) verify data quality. The *collect initial data task* deals with the issues surrounding

locating, assessing, and obtaining the needed data (from both internal and third-party sources) discussed above. The *describe data task* involves examining the "surface" properties of the acquired data. The surface properties of the data include such things as the format of the data (e.g., a relational database table versus a comma separated value text file); the amount of data (in terms of the number of records and variables); and the names, data types (e.g., categorical or numeric), coding schemes, and definitions of the variables in the data. A key objective of this task is to determine whether the collected data will answer the identified data mining question(s).

The *explore data task* involves the use of frequency distributions, cross-tabulations, means, correlations, and other simple descriptive statistics of the variables of interest, along with a number of other variables initially thought to influence the variables of interest. In addition, graphical tools such as histograms, scatter plots, and other simple plots of the data are frequently useful at this stage. These analyses may help refine the data description, lead to a better understanding of potential data quality issues, and help gain a basic understanding of the nature of relationships between different variables that will be of use in the modeling phase.

The *verify data quality task* addresses the following important questions. Are the data complete (i.e., do they cover all the relevant cases we hope to examine)? Do all the variables seem to be correct (e.g., are there variables that should be all numbers that contain some character entries; is the earliest date of a transaction dated November 17, 1999, when transaction data should go back further than this)? Are there missing values in the data? If yes, how common are they, and why do they occur?

2.2.3.3 Data Preparation

In the data preparation phase the final dataset to be used in model building is constructed from the available raw data. The preparation may include selecting records to use in the analysis; creating "clean" samples of records to use in the modeling process; selecting the variables to use in the analysis; and creating new variables by transforming some of the variables in the raw data or deriving them based on two or more variables in the raw data. As indicated earlier in the chapter, it is not unusual to move back and forth between this phase and the modeling phase as the project progresses.

There are five generic tasks associated with the data preparation phase: (1) select data; (2) clean data; (3) construct data; (4) integrate data; and (5) format data. One of the most important steps in conducting a data mining project is *selecting the data* to use for the actual analysis. Data selection relates to both what variables to have available in the data set to be used for

the actual data mining, as well as the nature of the data records to be used in the analysis. A number of issues come up with respect to the nature of the data records to include in the analysis. Of particular concern for predictive modeling applications is that often a positive response to a promotional offer (the behavior of interest) can represent a very small percentage of the customer base (response rates of around two percent for untargeted promotional offers are not uncommon). In these cases, separating signal from noise within the data can be very difficult. As a result, it is not uncommon to select records for analysis such that the positive responders to the variable of interest are over-sampled relative to the negative responders for the variable of interest. Moreover, some data mining methods (decision trees in particular) perform much better if there are roughly equal proportions of positive and negative responders in the dataset used in the data mining analysis. Thus, it is often the case that the dataset used in the data mining analysis consists of a stratified random sample of the original database in which there are equal numbers of positive and negative responders to the variable of interest, even though the positive responders represent only 2 percent of the original database. However, since we know the relative probabilities of including positive and negative responders, we can project the results based on the analysis dataset back to the overall customer database. Another question that needs to be addressed is how to handle records with missing data for variables that are thought likely to be important for the analysis. One solution is to simply omit records with important missing information. However, this may (or may not) lead to biased results.

The main element of the *data cleaning* task involves how to deal with data with missing values. Specifically, should missing values for a variable be replaced with a default value (say of zero or the mean value of the non-missing values for that variable), and if a default value is used, should another variable be created to indicate what records actually have missing values for that variable? Unfortunately, the answer to this question is "it depends." It turns out that at times non-responses to questions contain information in their own right. For instance, a credit card company in qualifying prospects may find that individuals who did not complete certain questions on their applications have a higher probability of defaulting on a credit card if it is issued compared to individuals who answered those questions.

Other variables with values that are likely to be incorrect must also be dealt with. To illustrate this point, one of the authors was once involved in a project to assist a charitable organization with their fundraising. An issue that became quickly apparent was that a number of variables containing date information had an extremely high number of records where the value of the date was November 17, 1999. Ultimately, it was determined that these values were er-

roneous, and were caused by a botched Y2K conversion of the charity's donor database. The problem then became one of how to deal with the erroneous date data. Fortunately, it turned out that other tables in the database had what was thought to be duplicate information, but in fact had the correct values for the records with corrupted date information, allowing for the erroneous values to be replaced with correct values. If this "duplicate" data was not available, then an important question that would need to have been addressed was whether only data with correct date information should be used in the analysis, thereby creating a "clean" subset of the data, but at the cost of discarding some of the records in the database.

The *construct data task* typically involves creating new variables through transforming a single variable (e.g., taking a natural logarithm of one of the variables in the original database) or creating derived variables from other variables in the database (e.g., dividing the total number of transactions a customer has made with our company by the number of months since that customer's first transaction to create an average transactions per month purchase frequency variable). The other thing this task may involve is creating completely new records. For example, there may be a need to create records for customers who have made no purchases in the past year if we are working with the past year's customer transactions database. By implicitly ignoring customers with no transactions, we may be overlooking important information.

The *integrate data task* involves merging different data tables together (say the transaction history of a customer, that is contained in the transactions database table, with the customer's personal information, contained in the customer information table) in order to create a single dataset that can be used by the data mining tools.

The *format data task* primarily refers to potential minor changes in the structure of variables or the order of variables in a database so that they match what is expected by a particular data mining method. Alternatively, it may involve changing the order of records so that the order is close to random so that certain data mining methods work properly. This issue tends to become most relevant if a stratified sample is created in order to oversample positive responses to the variable of interest, since the stratification may well result in all the records for the positive responders coming first in the new dataset, while the records for all the negative responders follow.

2.2.3.4 Modeling

As its name suggests, in the modeling phase the actual models are constructed and assessed. The generic tasks associated with this task are: (1) select mod-

eling technique(s); (2) generate a test design; (3) build model; and (4) assess model.

As Table 1.1 presented at the end of Chapter 1 indicates, the selection of an appropriate modeling method(s) is dependent on the nature of the database marketing application. However, for most applications, there is more than one appropriate method. In the *select modeling technique(s)* task a decision is made as to which of the possible methods that can be used should be used. The decision could be made to use all applicable tools, and then select the model that is "best" among the set of possible models as part of the assess model task, which, in turn, relies on the testing procedures developed in the generate a test design task.

The *generate a test design task* needs to be done prior to building any models. The main purpose of the testing environment is to assess the quality and validity of different models. This is typically accomplished by taking the dataset created in the data preparation phase and dividing it into two or three different samples. The first of these samples is known as the estimation sample (it is also called the training sample). The purpose of this sample is to actually build the model(s). The second sample is called the validation or test sample, and its purpose is to examine the accuracy and validity of a particular model, and to provide a basis for comparing the accuracy of different models. Based on the validation sample, a "best" model can be selected. However, to get an unbiased estimate of the likely impact (in terms of sales and profits) of the use of this best model, a third sample is needed to make this assessment (which actually occurs in the evaluation phase of the process). This sample is known either as the holdout or validation sample.

The *build model task* is where the previously selected data mining methods are applied to the dataset. You will see exactly how to do this in later chapters.

In the *assess model task*, the focus is on assessing a model on its technical merits as opposed to its business merits. The main concern at this point is the accuracy and generality of the model. An assessment with respect to the business problem being addressed is done in the evaluation phase. The assessment of a model can result in the conclusion that the model can be improved upon, and also suggest ways of making an improvement, resulting in a new build model task.

2.2.3.5 Evaluation

At this point a model (or several) has been created that possesses a reasonable level of accuracy. Before deploying a model, an assessment of its likely impact needs to be made. This can partially be accomplished for predictive models by using the holdout sample to develop an estimate of the returns from using the

model. However, this alone is not sufficient. Another critical factor that needs to be addressed is whether there is some important business issue that has not yet been sufficiently considered. In addition, the potential implications of repeated use of the model must be thought through to determine if there are any potential negative side effects associated with the use of the model.

The generic tasks of this phase are: (1) evaluate results; (2) review process; and (3) determine next steps. The activities associated with *evaluating the results* are discussed in the prior paragraph, while the *review process task* is really a quality assurance assessment, which addresses concerns such as: Was the model correctly built? Were the variables defined in a way that is consistent with the variables available in the actual customer database? Has a variable that is "trivially related" to the target variable (i.e., is a perfect predictor of the target since its value is based on the value of the target variable) been included in the model? Will the variables used in this analysis be available for future analyses? In the *determine next steps task*, the project team needs to decide whether to finish the project and move on to deployment (if appropriate) or whether to revisit certain phases of the project in an attempt to improve upon them.

2.2.3.6 Deployment

The nature of the deployment phase will vary with the nature of the project. Certain customer segmentation studies are done in an effort to gain a better understanding of an organization's customers, so deployment involves effectively communicating the knowledge gained from the project to appropriate people within the organization. In other cases, such as nearly all applications involving predictive modeling, there is a need to incorporate the estimated models into the organization's business decision processes, such as determining which customers should be targeted with a particular offer. In order to successfully deploy a data mining–based solution, four generic tasks may (depending on the type of project) need to be undertaken: (1) plan deployment; (2) plan monitoring and maintenance; (3) produce a final report; and (4) review project.

If relevant, a *deployment plan* needs to determine how best to incorporate any models developed into the relevant business processes of an organization. While this may seem straightforward, we have been involved in several projects where this proved to be a real stumbling block. It can be a real problem when consultants develop the data mining models, and there is insufficient buy-in on the part of personnel (particularly information systems personnel) in the client organization. Thus, an assessment of possible deployment pitfalls must be made. A related issue, but one relevant in almost all data mining

applications, is determining who within the organization needs to be informed of the project's results, and how best to propagate this information.

Prior to deploying a data mining solution, *plans for monitoring and maintaining* that solution must be made. To do this, an assessment of what changes could occur in the future which would trigger an examination of the deployed model (the entrance of a major new competitor, or a sharp rise in interest rates, may trigger an assessment of a model's current predictive accuracy) needs to be undertaken. In addition, a maintenance schedule to periodically test whether a model is still accurate, along with criteria to determine the point at which a model needs to be "refreshed" (i.e., rebuilt using more recent data), needs to be developed.

At the end of the project, the project team needs to *produce a final written report*. Depending on the nature of the project and its deployment, the report may be a summary of the project and the lessons learned from undertaking the project, or it may be a comprehensive presentation of all aspects of the project, with a detailed explanation of the data mining results. In addition to the written report, there may also be a final presentation.

The *project review task* is an assessment of what went both right and wrong with the project. Its main purpose is to provide the basis for learning about what should be done in a similar fashion, and what should be done differently in future projects.

2.2.4 The Typical Allocation of Effort across Project Phases

A natural question to ask is the relative amount of time that is likely to be devoted to different phases of a project. Based on our experience, and those of others, we think reasonable guidelines are:

1. Business understanding: 5 to 15 percent
2. Data understanding: 5 to 10 percent
3. Data preparation: 50 to 60 percent
4. Modeling: 5 to 15 percent
5. Evaluation: 5 to 10 percent
6. Deployment: 10 to 15 percent

Probably the most surprising things about this list is the large amount of time devoted to data preparation and the fairly small amount devoted to modeling.

The length of time spent on the data preparation phase should never be underestimated. While preparing the data seems like it should be straightforward, there is always a real horror show lurking in one of the database tables you will be using, just waiting to be found when you least expect it. The botched Y2K conversion discussed earlier is a classic example of a data preparation horror show. Unfortunately, horror shows never seem to repeat themselves across projects, so each project has it own unique one, and you have no idea what it will be until you stumble upon it. The actual modeling takes remarkably little time because it typically doesn't involve much more then selecting the values of several model parameters, selecting a set of variables, and clicking on an OK button to estimate the model. When things go badly in the modeling phase, it is typically a data problem that is the real culprit, requiring a return to the data preparation phase to correct it.

Having provided some basic background in this chapter and the previous one, we are now ready to begin to get our hands dirty by working more directly with data.

Part II

Predictive Modeling Tools

Chapter 3

Basic Tools for Understanding Data

The primary objective of this chapter is twofold. The first objective is to present a number of tools that are useful for the data understanding phase of the CRISP-DM process model (Chapman et al., 2000). The set of tools we present in this chapter is not the complete set we present in the book. The chapters on linear and logistic regression will present additional visualization tools useful in this phase of a data mining project. We have elected to hold off on the presentation of these visualization tools since they will have greater value in the context of those chapters. The second objective of this chapter is to introduce you to R and the modified version of the R Commander that we will use as the data mining workbench in this book.

Before we can successfully apply tools to better understand our data, we first need to know more about the nature of variable data types. It turns out that how we apply tools to understand a variable depends on what type of a variable it is. "Measurement scales" is the term used to describe the properties of variables that define their type. Consequently, this chapter begins with an introduction to measurement scales and variable types. Following this are four tutorials on basic tools for understanding data. The first tutorial shows you how to load data contained in an Excel file (the file format used to hold a remarkably high percentage of many organizations' data, often inappropriately) as well as data in an R "package" into R. The second tutorial covers obtaining simple descriptive statistics about a data set as a whole, and about individual variables within that data set. The third tutorial covers tools to examine the distribution across records of a single variable (known as a frequency distribution, which is visually displayed using a histogram), while the fourth tutorial looks at a simple multivariate analysis tool known as a contingency table used to look at the relationship(s) between two or more variables. For the last three tutorials, the tutorial will both describe tools and show you how to apply these tools to an example data set using R Commander.

3.1 Measurement Scales

Customers, products, companies, and any other "object" are described by their attributes. Attributes may vary from one object to another or from one time to another. To measure attributes, we assign numbers or symbols to them. The attribute of "eye color" of the object "customer" can be assigned the value blue, brown, or green. The attribute of "age" of the object "customer" can be assigned values of 23 or 76 years, and so on. Age will vary over customers and over time, while eye color (or at least natural eye color) only varies over customers.

A useful and simple way to specify the type of attribute is to identify the properties of numbers that correspond to the underlying properties of the attribute. An attribute like age, for example, has many of the properties of numbers. It makes sense to compare and order customers by age, as well as to talk about differences in age or the ratio of ages of two different customers (e.g., a customer who is 25 is half the age of a customer who is 50).

The following properties of pairs of numbers or values are typically used to describe attributes:

1. Distinctness: $=$ or $\neq$
2. Order: $\leq$, $\geq$
3. Addition: $+$ and $-$
4. Multiplication: $\times$ and $\div$

With these properties, we can define four types of attributes: *nominal*, *ordinal*, *interval*, and *ratio*. Each type possesses all of the properties of the preceding type. That is, if you can divide two numbers, you can also determine if they are equal or not. The reverse is not necessarily true—blue is not equal to brown, but blue divided by brown makes no sense. Thus, *nominal* variables only have the property of distinctness; *ordinal* variables also have the property of distinctness, but they possess the property of order as well; *interval* variables have the first two properties and the property of addition, so that we can meaningfully measure the difference between different values of an interval variable; and *ratio* variables possess all four properties, so we are able to say that one value of a ratio variable is three times as large as another value. Each variable type also has certain statistical operations that are valid for it. For example, it makes sense to talk about the average age of our customers, but not about their average eye color. Nominal and ordinal variables are also typically referred to as *categorical*, while interval and ratio variables are referred to as

Table 3.1: A Summary of Attribute Measurement Scale Types

Attribute Type		Description	Examples	Operations
Categorical	Nominal	Values are just different names. They provide only enough information to distinguish one object from the other ($=$ or $\neq$).	Zip/postal codes, employee ID, eye color, gender	Mode, contingency table, chi-squared
	Ordinal	Values provide enough information to order objects as first, second, third, and so on ($<$,$\leq$,$>$,$\geq$).	Hardness of minerals, grades, street numbers, gold–silver–bronze	Median, percentile, rank correlation
Numeric	Interval	The differences between values are meaningful ($+$, $-$).	Calendar data, temperature in Celsius or Fahrenheit	Mean, standard deviation, Pearson's correlation, T-test
	Ratio	Both differences and ratios are meaningful.	Money, age, height; temperature in Kelvin	Geometric mean, percent variation or elasticity

numeric or *quantitative*. Table 3.1 summarizes attribute measurement scale types.

We also use the terms *discrete* and *continuous* to describe attributes. Continuous attributes can take on any real number values—such as 36.35 years old. Discrete attributes do not take on these sorts of intermediate values. Brown

and blue are discrete, as are first, second, and third. Buy or not buy, which only has two levels, is known as a *binary* discrete variable (or simply a *binary variable*). Usually, interval and ratio variables are continuous, and ordinal and nominal variables are discrete, hence *continuous* and *discrete* are often used interchangeably with *numeric* and *categorical*. However, there are exceptions to this. For example, count data (such as the number of separate purchase occasions a customer has in a particular store in a given year) is both discrete and ratio scaled since the number of trips for any particular customer will always be an integer value.

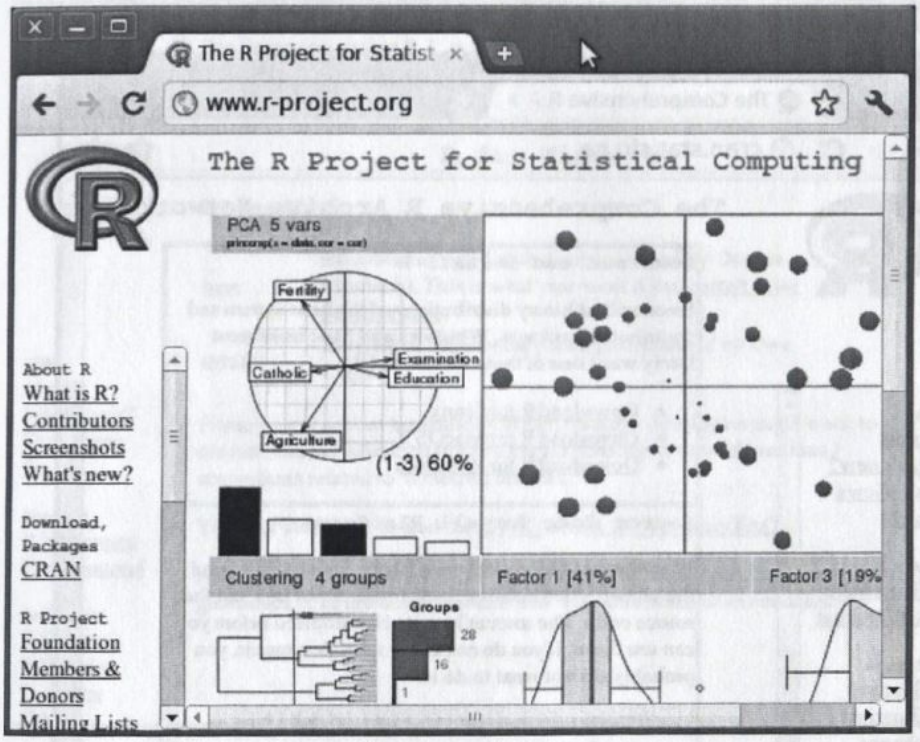
3.2 Software Tools

The software tools used with this book are written in R (R Development Core Team, 2011), a language and system for computational statistics. R is open source software, developed by a large, worldwide team of developers. The software consists of a base system and an extensive library of add-on packages. A recent survey (Rexer et al., 2010) found that R was the software product that had the largest percentage of users (43%) of data mining professionals.

On both Windows and OS X, R has a limited graphic user interface (or GUI). However, these GUIs are operating system specific, and both are fairly difficult to extend. Instead, there are R bindings to several GUI toolkits, including Tcl/Tk, GTK2, Java Swing, and GTK2. Using these GUI tookits, Deducer (Fellows, 2011), pmg (Verzani, 2011), and the R Commander (Rcmdr; Fox, 2005) offer fairly complete basic GUIs for R, and all three are extensible. In addition, the rattle package (Williams, 2009) offers a GUI to R that is oriented toward data mining. One thing that separates the various R GUI packages is the GUI toolkit they are based on. In turn, these different toolkits have different installation requirements, with some requiring more effort to install than others. The GUI toolkits also differ in the level of "quirks" they exhibit across operating systems.

The easiest GUI toolkit to deal with from a user installation perspective, and the one that is most mature, is Tcl/Tk. The Tcl/Tk toolkit is installed as part of an R installation on Windows (so if you have R, you have the Tcl/Tk toolkit on a Windows system), it requires only one additional software package (beyond R itself) to be installed on OS X, and is likely to already be installed on most Linux systems. The only fairly complete basic R GUI that uses the Tcl/Tk toolkit is the R Commander. The combination of this fact, along with the R Commander's greater level of maturity, and the ease with which custom

Figure 3.1: The R Project's Comprehensive R Archive Network (CRAN)



plug-in packages can be developed led us to adopt it for the software tools to accompany this book.[1]

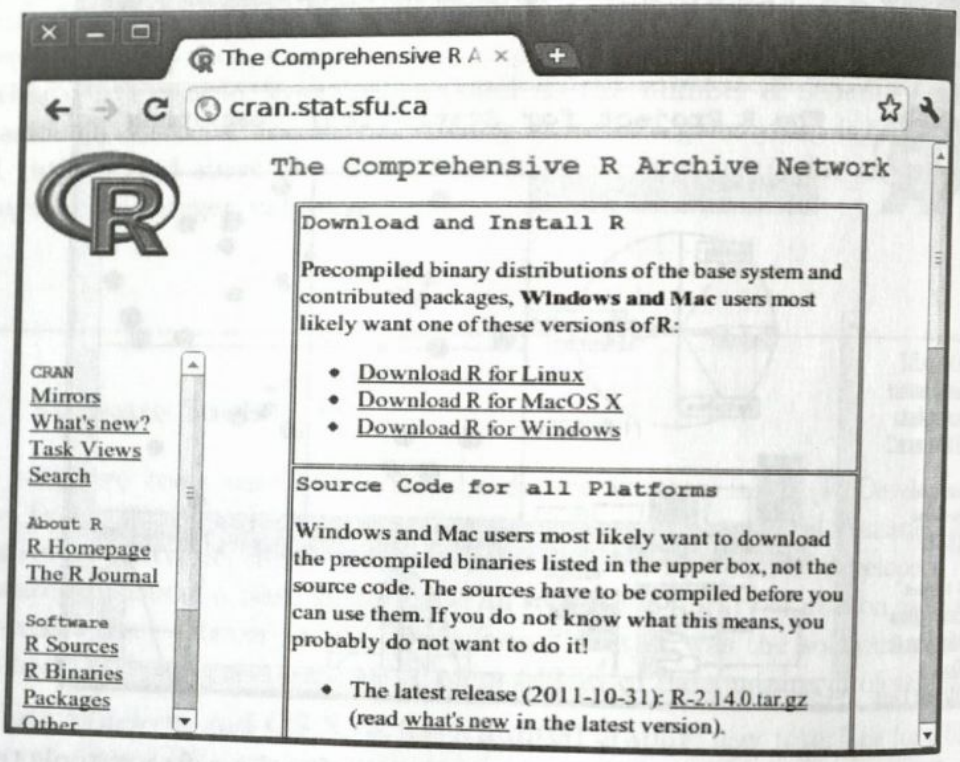
The R system software as well as most R packages are available from the Comprehensive R Archive Network (which is typically referred to as CRAN). CRAN has mirror repositories located around the world. Selecting the mirror repository nearest to you should reduce the time it takes to download the software. Like R itself, the software developed for this book is open source, and is publicly available from CRAN. In what follows, we provide step-by-step instructions on how to obtain and install the needed software on both Windows and OS X systems.

3.2.1 Getting R

The preferred way to obtain R is to open a web browser and go to the R project's main site at http://www.r-project.org, where you will see a page that looks similar to Figure 3.1. In the left-hand navigation bar you should

[1]The data mining–oriented rattle package uses the GTK2 toolkit, which makes it potentially challenging to install on Windows and (particularly) OS X systems. In addition, our approach to applied data mining is a bit different from the one implicit in the rattle GUI.

Figure 3.2: The Entry Page into the Comprehensive R Archive Network (CRAN)



see a link to CRAN (which has been placed in a rectangle in Figure 3.1). Click on this link, which will bring you to a page to select the CRAN mirror site (organized by country) from which to download the software. After selecting a mirror site you will see the page shown in Figure 3.2. At the top of this page are links to operating specific precompiled binaries. Select the link to the operating system running on your computer.

After clicking on the appropriate link, Windows users will see the page shown in Figure 3.3. Click on the "base" subdirectory (the top-left link on the main part of the page) to get to the actual download page, which is shown in Figure 3.4. Click on the top link of the main part of this page to download the installer for the current version of R. You may also want to browse through the frequently asked questions to see how to troubleshoot issues that may come up during the installation (such as administrator privilege issues under Windows 7 and Windows Vista).

Mac OS X users will first see the page shown in Figure 3.5. Click on the link in the top-left of the main page to download the installer for the base R system. In addition, you will need to click on the link to "the tools directory" to get

Figure 3.3: The R for Windows Page

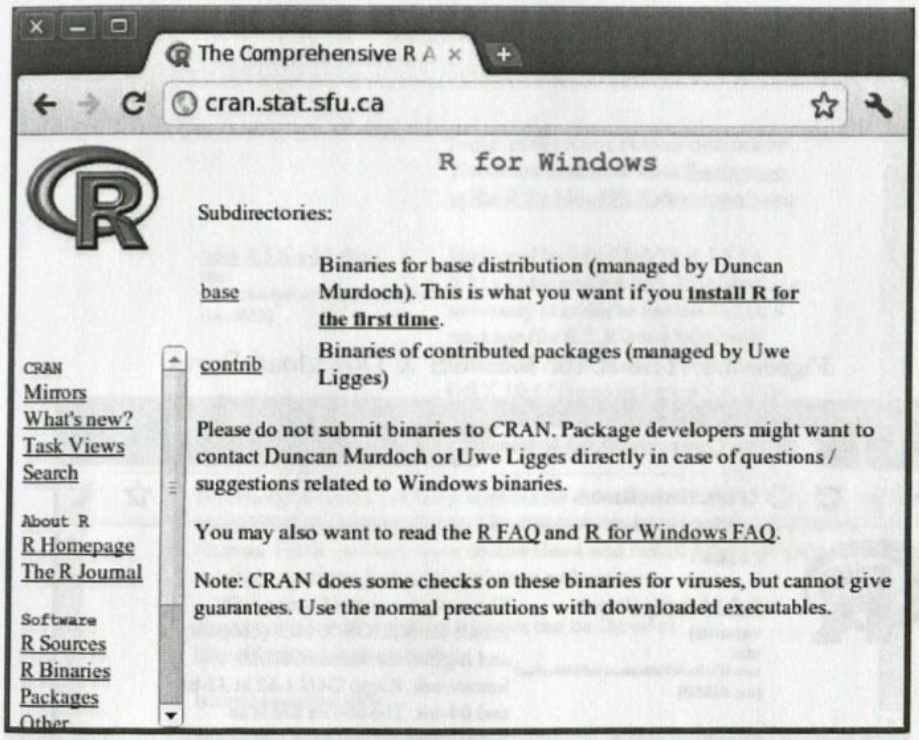


Figure 3.4: R for Windows Download Page

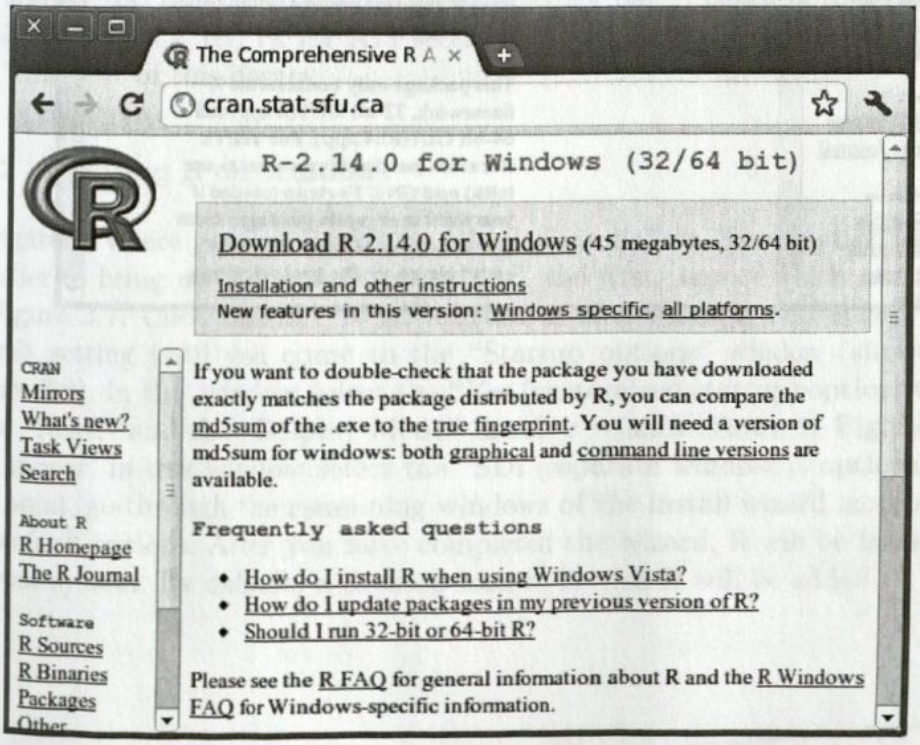


Figure 3.5: The R for Mac OS X Download Page

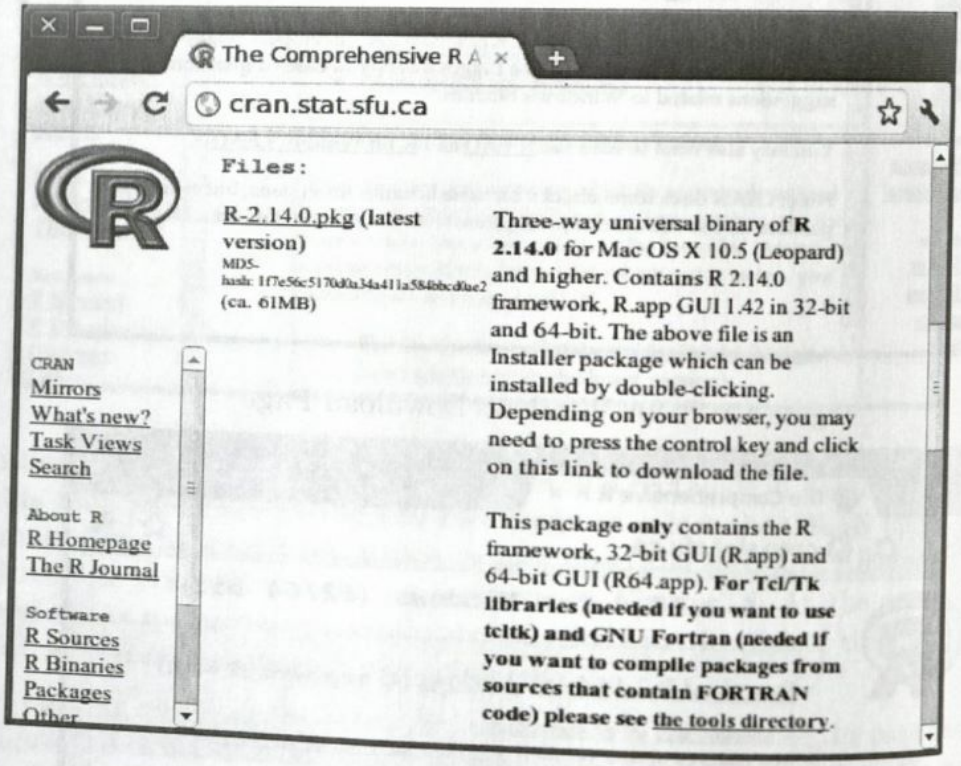
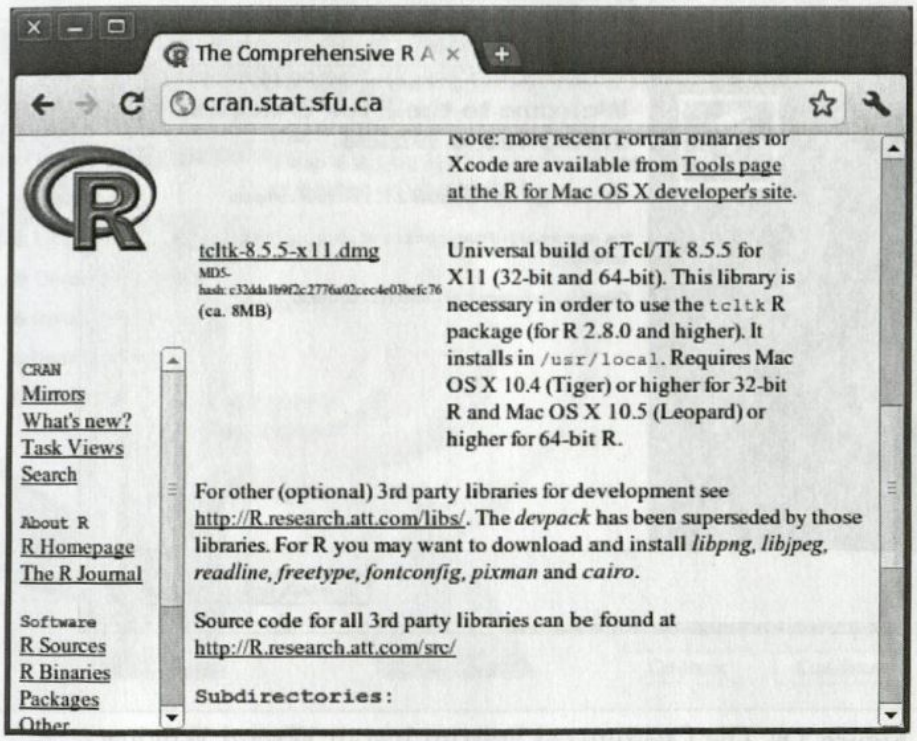


Figure 3.6: Mac OS X X11 Tcl/Tk Download Page



the needed Tcl/Tk library. On the tools directory page (shown in Figure 3.6), scroll down to the Tcl/Tk for X11 section, and click on the link to download the *.dmg file of this library.

3.2.2 Installing R on Windows

Navigate to where you save the R for Windows installers. Double-click on the installer to bring up the installation wizard, the first page of which is shown in Figure 3.7. Click on the "Next" button in this window, and accept the default setting until you come to the "Startup options" window (shown in Figure 3.8). In this window select the "Yes (customized startup) option, then press "Next," and the "Display Mode" installer window shown in Figure 3.9 will appear. In this window select the "SDI (separate windows)" options. At this point, go through the remaining windows of the install wizard, accepting the default options. After you have completed the wizard, R will be installed on your system. By default, a desktop icon to launch R will be added.

Figure 3.7: The R for Windows Installation Wizard

Figure 3.8: The Customized Startup Install Wizard Window

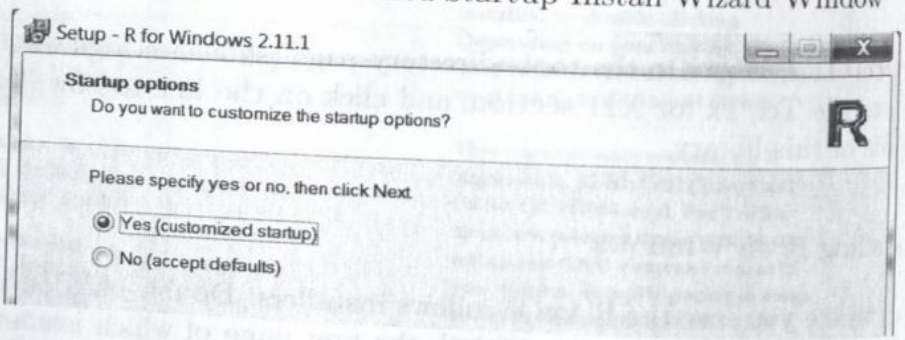


Figure 3.9: The Installation Wizard Display Interface Selection Window

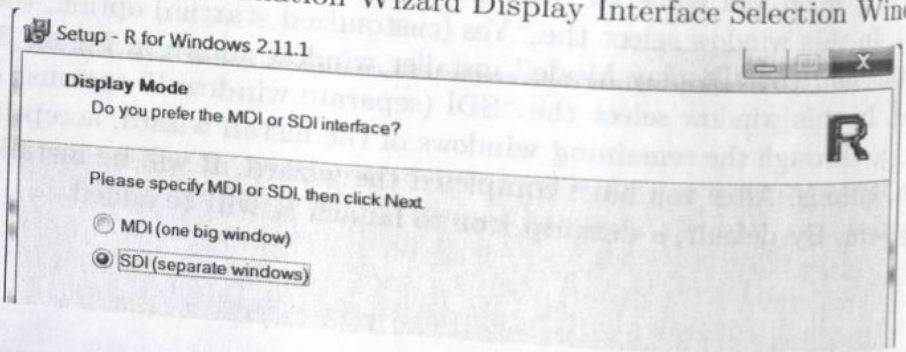
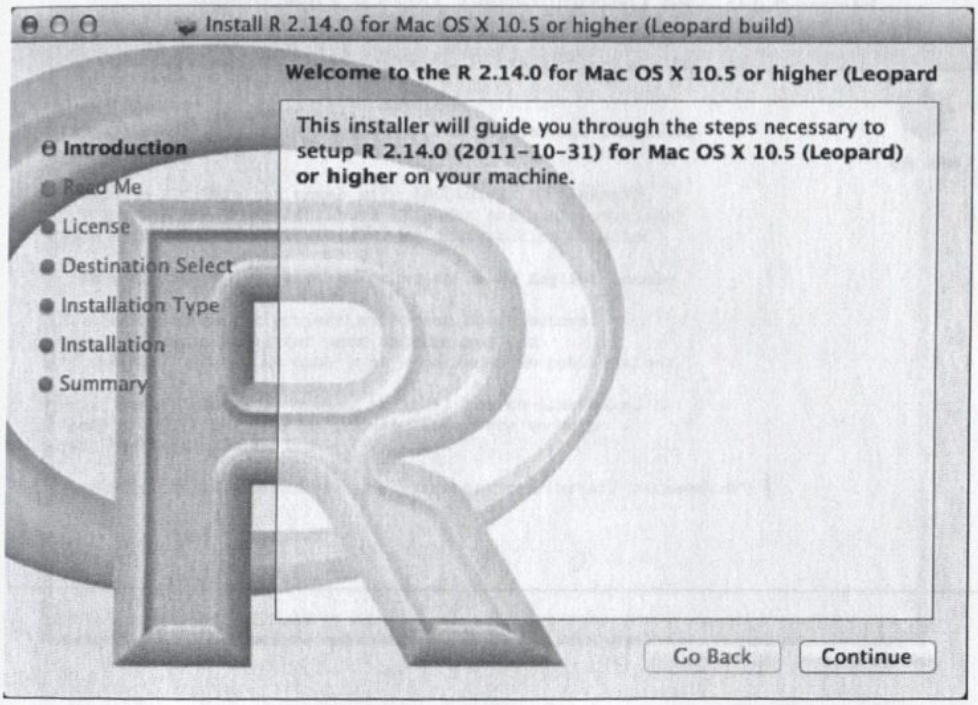


Figure 3.10: The R for Mac OS X Installer Wizard Splash Screen



3.2.3 Installing R on OS X

For reasons of convenience, you may want to copy the two files you downloaded onto the desktop. To start the installation, double-click on the R-2.x.x.pkg (the x's will actually be replaced by numbers), which will launch the R installer wizard, the first window of which is shown in Figure 3.10. In the case of OS X, accepting all of the default options works well. As a result, just follow the instructions in the installation wizard. Along the way you will be asked to accept the license terms (which you should) and asked to authenticate (i.e., enter your password) just before the software is actually installed on your system. The next thing to do is double-click on the tcltk-8.5.x.dmg file (again, the x will be replaced by doing this). The tcltk-8.5.x.dmg is a disk image file, so double-clicking on it will cause the disk image to uncompress. This will reveal a single file (see Figure 3.11), which will have the name tcltk.pkg. Double-click on this file to start the installer, and you should see a window similar to the one in Figure 3.12. You should accept all the default settings for this package as you go through the wizard, and you will again be asked to accept the license and enter your password to authenticate just prior to the actual installation of the software.

Figure 3.11: The Uncompressed Tcl/Tk Disk Image

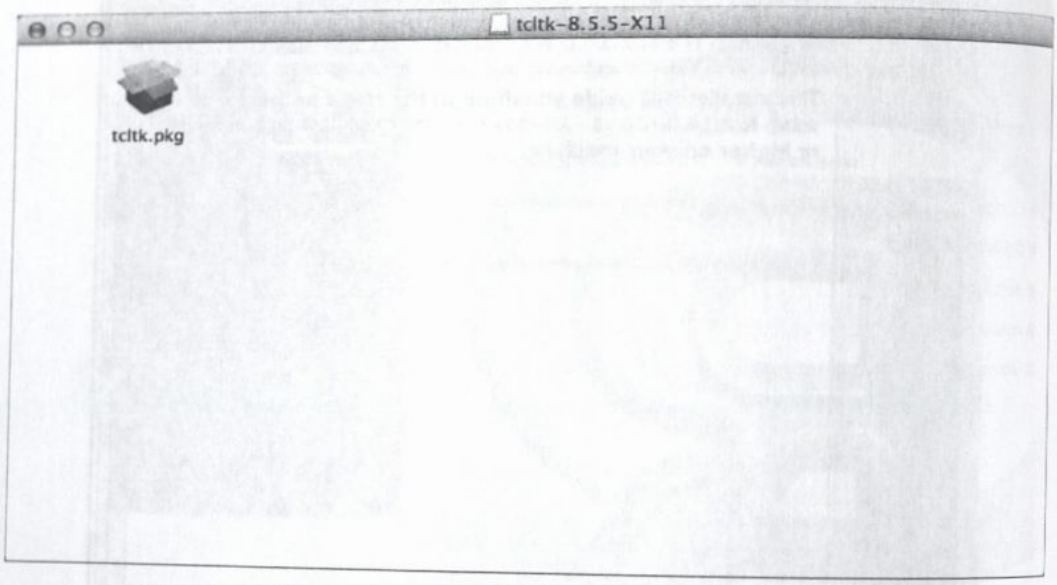


Figure 3.12: The Tcl/Tk Installation Wizard

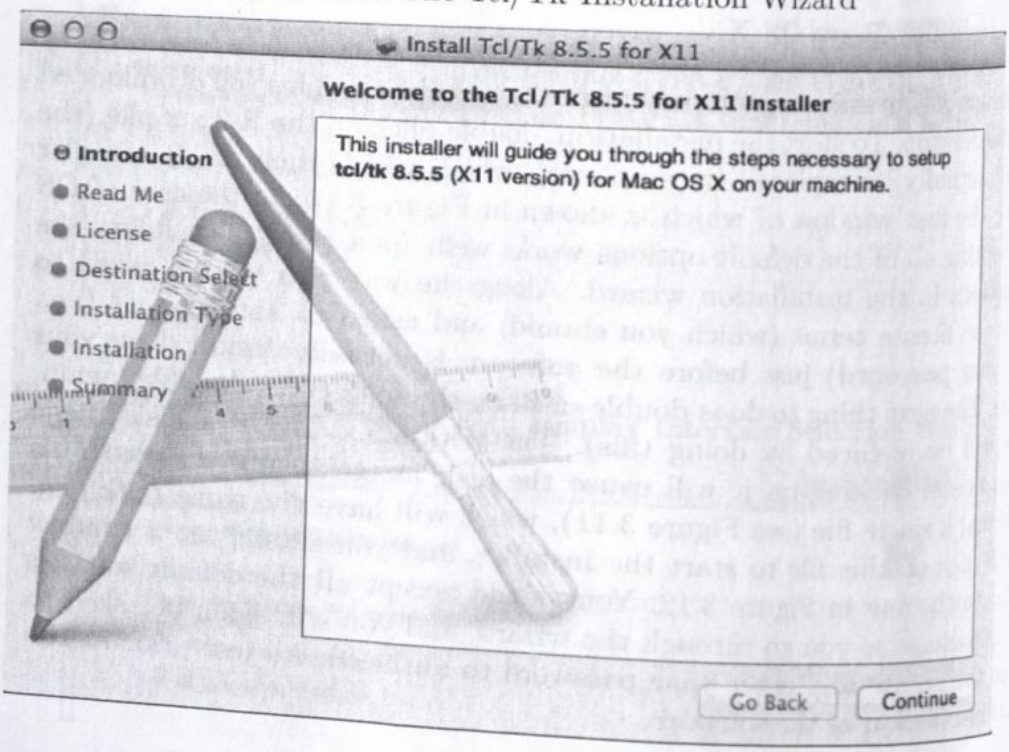
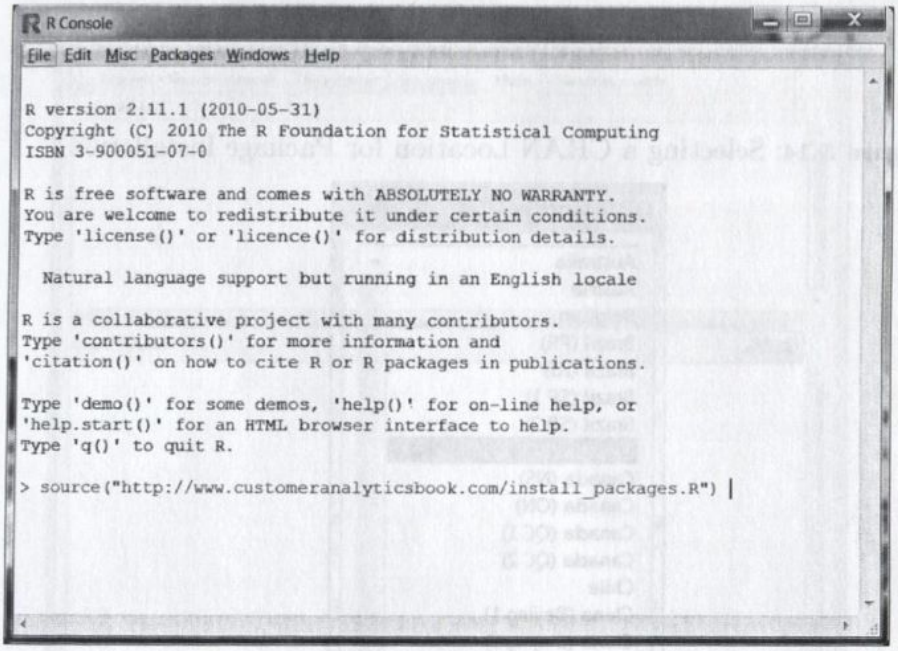


Figure 3.13: The Source Command to Install the RcmdrPlugin.BCA Package

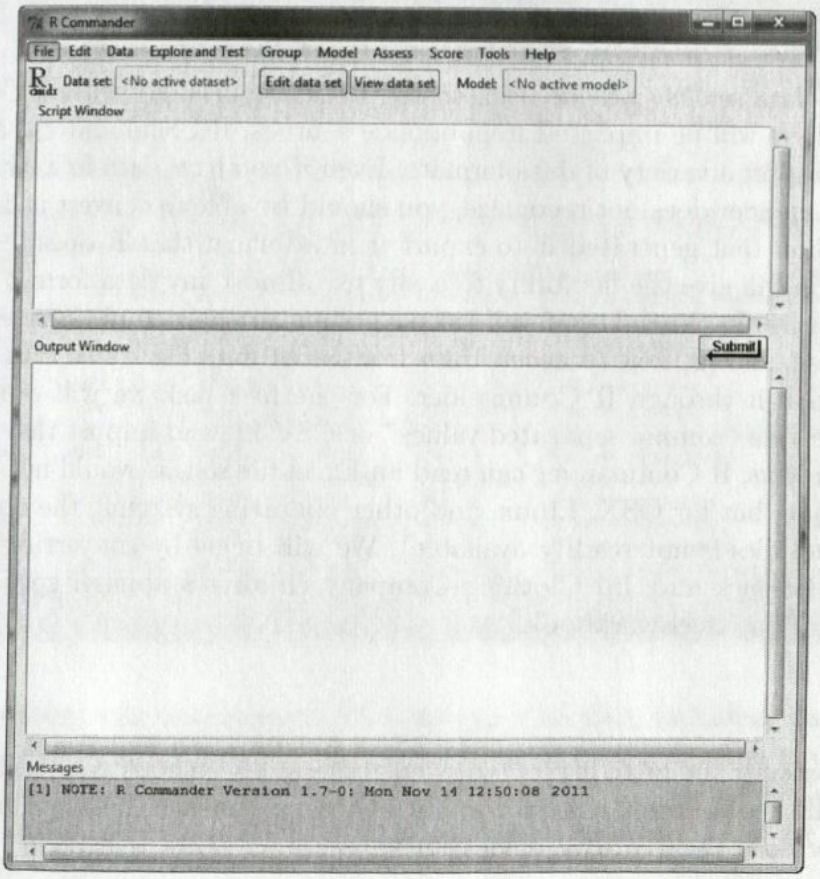


3.2.4 Installing the RcmdrPlugin.BCA Package and Its Dependencies

To install the software that goes along with this book you need to launch R. Mac OS X users will want to launch X11 prior to launching R. X11 should live in the Utilities folder within the Applications folder of the Finder. Windows Vista and Windows 7 users are likely to want to launch R by right-clicking on the R desktop icon and selecting "Run as Administrator." Once R is launched, you should see the R Console window (the Windows version of the R Console is shown in Figure 3.13). The R console provides a command line interface to R, along with a small number of GUI tools to work with the R system. To install the RcmdrPlugin.BCA package, along with the packages that it uses, we need to issue one command using R's command line interface. The R prompt is a greater than sign (>), which will be at the bottom of the R Console. At the prompt enter the command: source("http://www.customeranalyticsbook.com/install_packages.R"), which is illustrated in Figure 3.13, and then press enter. At this point a pop-up window will appear (shown in Figure 3.14) asking you to select a CRAN mirror repository from which to download the needed packages. Select the mirror site nearest to you, and then press OK. At this point you will see a lot of activity in the R Console window as R downloads and installs the needed packages. When R is done installing the software, a new prompt (a ">") will appear at the bottom of the R Console. When it does, you can launch into

Figure 3.14: Selecting a CRAN Location for Package Installation

Figure 3.15: The R Commander Main Window



the data mining tools that accompany this book by entering the command: library(RcmdrPlugin.BCA). What will happen next is the R Commander GUI will appear, momentarily disappear, and then reappear. When it reappears, you should see a window very similar to the one shown in Figure 3.15. In future sessions, you will need to enter the "library(RcmdrPlugin.BCA)" command at the beginning of the session to load the needed tools.[2]

[2]For Mac OS X users, if you enter this command and receive an error message that ends with the line "Error: package 'tcltk' could not be loaded," it means that you did not start X11 before launching R. Exit from R using the File → Quit drop-down menu, start X11, start R, and then enter the library command to load the RcmdrPlugin.BCA package.

3.3 Reading Data into R Tutorial

Most of the data sets we will be using in the book are included with R Commander. Others will be imported from outside sources. R Commander is able to directly import a variety of data formats. Even if you have data in a format that R Commander does not recognize, you should be able to convert it using the application that generated it to export it in a format that R does recognize, and this will give the flexibility to easily use almost any data format you happen to have. In this lab you will learn (1) how to convert an outside file and import it, and (2) how to access data from an R data library so that you can work with it through R Commander. For the first task we will convert an Excel file to a "comma separated values" or CSV file and import that file. (Under Windows, R Commander can read an Excel file so this would not normally be done, but for OSX, Linux, and other operating systems, the ability to read Excel files is not readily available). We will begin by converting and importing the Jack and Jill Clothing Company children's apparel spending data set from an Excel workbook.

1.

Use a web browser and go to http://www.customeranalyticsbook.com/jackjill.xls and you will then be asked whether you want to open or save this file. Choose Save to download the file to your local drive.

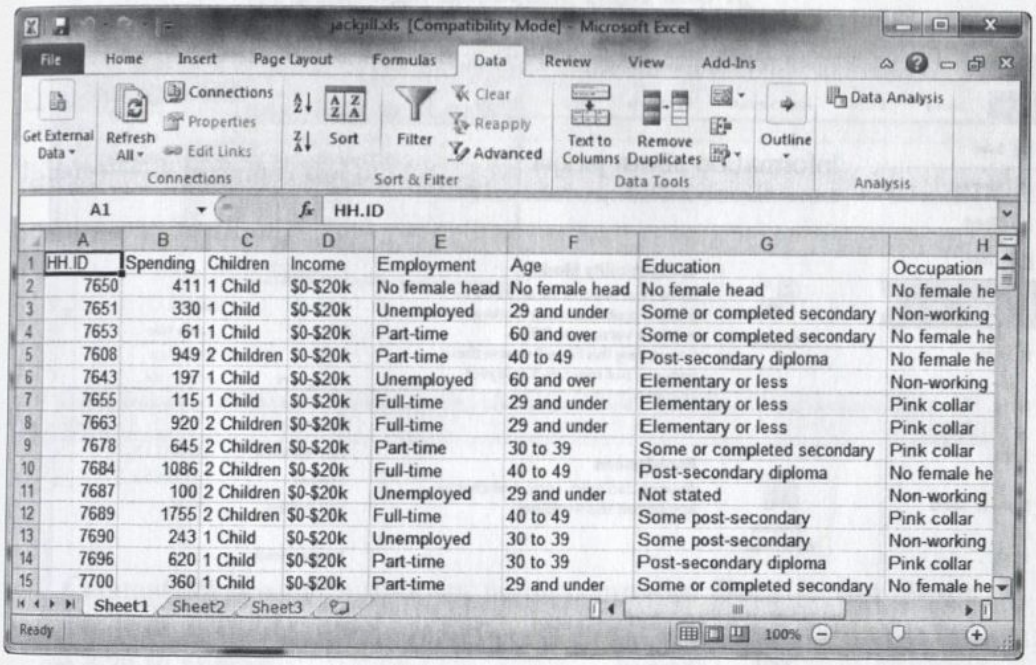
2.

The **Save As** dialog box will appear asking you where to save the file "jackjill.xls" (the suffix may not appear, depending on your operating system's settings). Navigate to the folder in which you would like to keep your files related to the tutorials for this book, and then press the **Save** button.

3.

After the file "jackjill.xls" has been downloaded to your local drive, locate it and **open it in Excel.** Once you have done this, you should see the file shown in Figure 3.16. This file contains children's apparel spending and household socioeconomic information for 557 households for the year 1992. Look through the file. You will notice that most of the columns contain text as opposed to numbers. Most of the variables in this data set are categorical (some nominal and some ordinal), and the text values describe the category.

Figure 3.16: jackjill.xls



jackjill.xls [Compatibility Mode] - Microsoft Excel

File | Home | Insert | Page Layout | Formulas | Data | Review | View | Add-Ins

Get External Data | Refresh All | Connections | Properties | Edit Links | Sort | Filter | Clear | Reapply | Advanced | Text to Columns | Remove Duplicates | Outline | Data Analysis

Connections | Sort & Filter | Data Tools | Analysis

A1 | HH.ID

	A	B	C	D	E	F	G	H
1	HH.ID	Spending	Children	Income	Employment	Age	Education	Occupation
2	7650	411	1 Child	$0-$20k	No female head	No female head	No female head	No female he
3	7651	330	1 Child	$0-$20k	Unemployed	29 and under	Some or completed secondary	Non-working
4	7653	61	1 Child	$0-$20k	Part-time	60 and over	Some or completed secondary	No female he
5	7608	949	2 Children	$0-$20k	Part-time	40 to 49	Post-secondary diploma	No female he
6	7643	197	1 Child	$0-$20k	Unemployed	60 and over	Elementary or less	Non-working
7	7655	115	1 Child	$0-$20k	Full-time	29 and under	Elementary or less	Pink collar
8	7663	920	2 Children	$0-$20k	Full-time	29 and under	Elementary or less	Pink collar
9	7678	645	2 Children	$0-$20k	Part-time	30 to 39	Some or completed secondary	Pink collar
10	7684	1086	2 Children	$0-$20k	Full-time	40 to 49	Post-secondary diploma	No female he
11	7687	100	2 Children	$0-$20k	Unemployed	29 and under	Not stated	Non-working
12	7689	1755	2 Children	$0-$20k	Full-time	40 to 49	Some post-secondary	Pink collar
13	7690	243	1 Child	$0-$20k	Unemployed	30 to 39	Some post-secondary	Non-working
14	7696	620	1 Child	$0-$20k	Part-time	30 to 39	Post-secondary diploma	Pink collar
15	7700	360	1 Child	$0-$20k	Part-time	29 and under	Some or completed secondary	No female he

Sheet1 | Sheet2 | Sheet3

Ready | 100%

R Commander can read several different types of files, including Excel files like this one. However, if a file cannot be read, often changing to a different file type will solve the problem. To illustrate this first trick in your tool kit, we will convert the *.xls file to a comma separated value file, with the file extension *.csv.

4.

As shown in Figure 3.17, use the pull-down menu command **File→ Save As**, which will bring up the **Save As** dialog box shown in Figure 3.18. Navigate to where you want to save the CSV file you are about to create, and, as shown in Figure 3.18, use the drop-down menu to **select "CSV (Comma delimited)(*.csv)" and save** the file. By default, the file will have the name jackjill.csv.

5.

Launch R by double-clicking its icon, which, after a few moments, will bring up the R console. **Enter the command library(RcmdrPlugin.BCA)** to bring up the R Commander GUI, which is shown in Figure 3.19. It contains drop-down menus, buttons, a script window, an output window, and a messages window.

Figure 3.17: Saving a File to Another Format in Excel

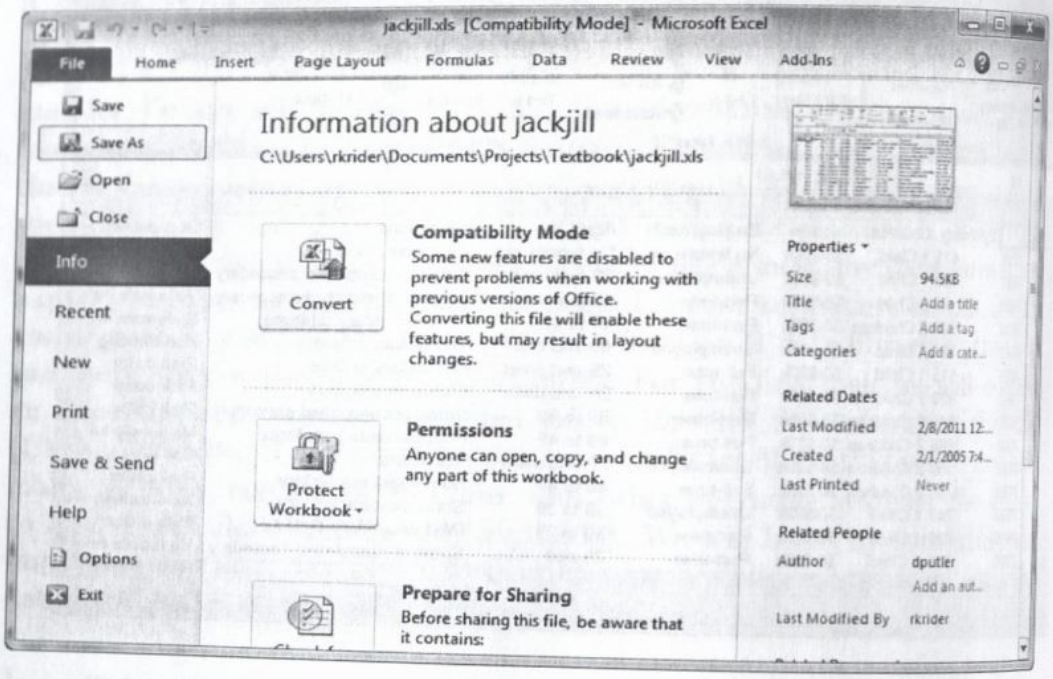


Figure 3.18: Saving a CSV File in Excel

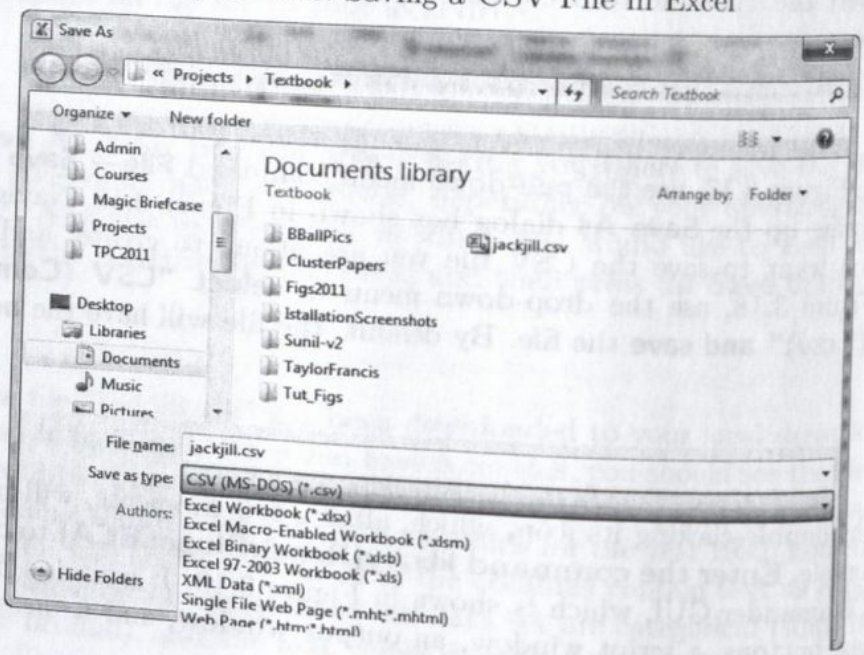
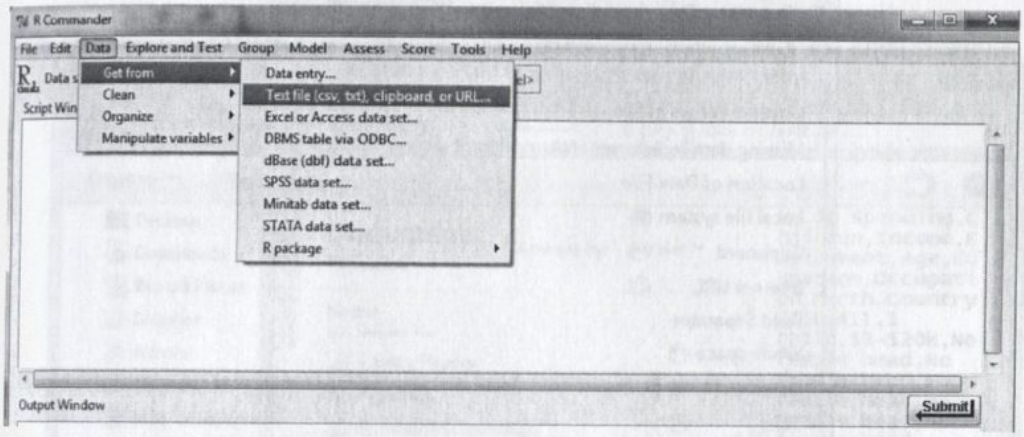


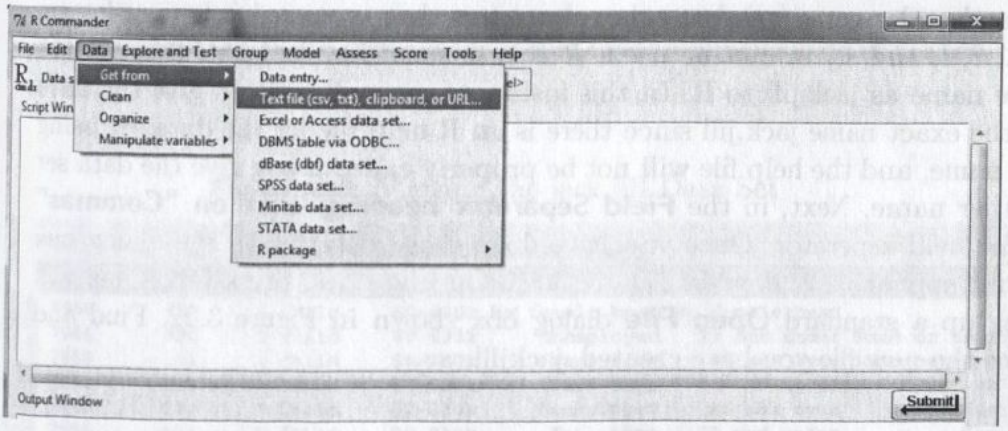
Figure 3.19: Importing Data into R



6.

In R Commander use the pull-down menu command **Data** → **Get From** → **Text file (csv, txt) clipboard, or URL...**, which will cause the dialog box in Figure 3.20 to appear.

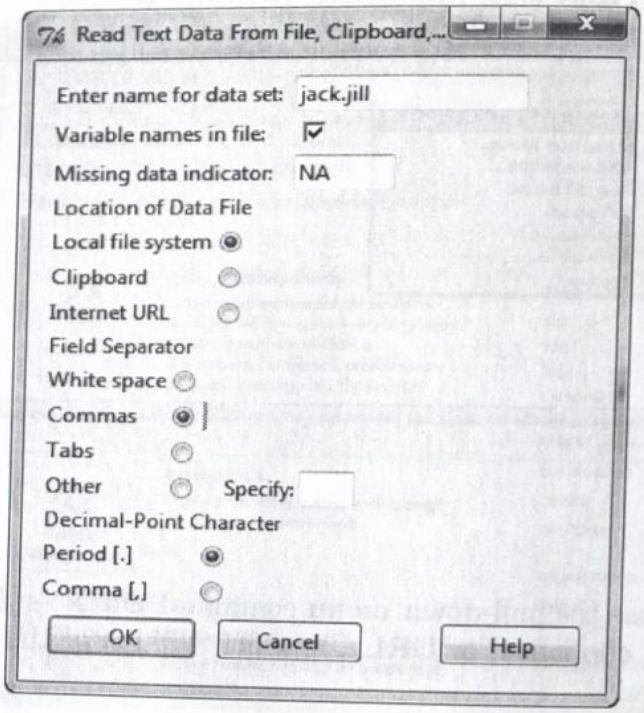
Figure 3.20: The Import Text File Dialog Box



7.

In this dialog box, **enter `jack.jill`** in the "Enter name for data set:". Whatever name you enter here is what R will use to refer to this data set after it is read into R and stored as an R readable file. It does not have to be the same as the name of the input file, which will be the case here. Generally, you have a wide degree of latitude in what you can name a data set. Although

Figure 3.21: The Completed Import Text Dialog Box



the name needs to start with a letter, it *can* contain letters, numbers, periods (.), and underscores (_), but other characters that are not numbers or letters (e.g., ?, !, and <) *cannot* be used. *R is case sensitive*, so Jack.Jill is not the same name as jack.jill to R. In this instance, you will want to give the data set the exact name jack.jill since there is an R help file for the data set using this name, and the help file will not be properly called if you give the data set another name. Next, in the **Field Separator heading, click on "Commas"** as the field seperator. Once you have done these two things, the dialog box should appear as it does in Figure 3.21. Once it does, press **OK**. This will bring up a standard **Open File** dialog box shown in Figure 3.22. Find and select the new file you have created, jackjill.csv.

8.

The bottom window in R Commander provides informative messages, such as errors, and should always be monitored. If the file has imported correctly, this window will now show that the dataset jack.jill has 557 rows and 9 columns, indicating successful loading of the database. If you press the **View data set** button on the R Commander toolbar you will be able to view the jack.jill data set, as is shown in Figure 3.23.

Figure 3.22: The Standard Open File Dialog Box

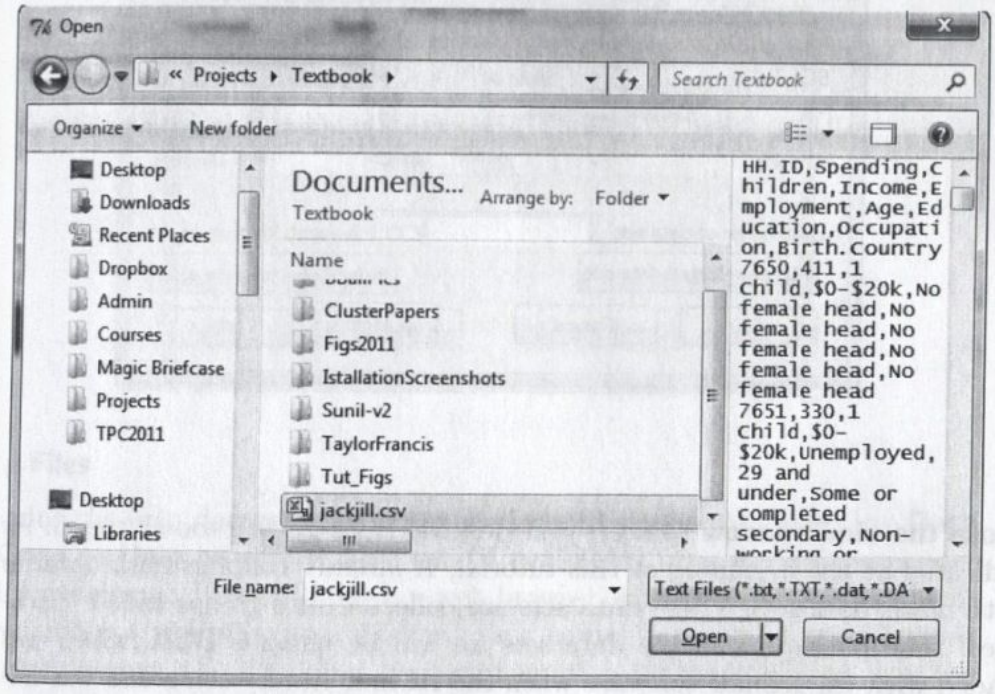
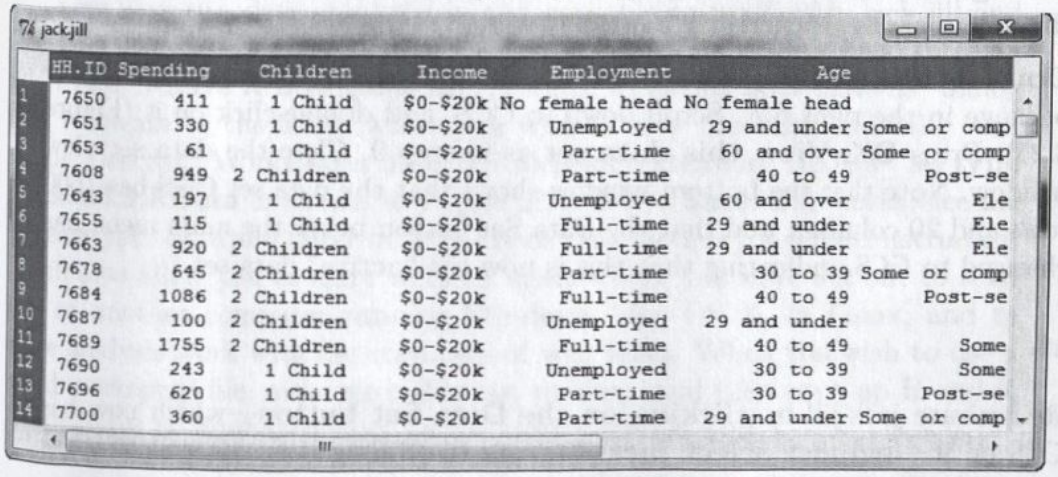
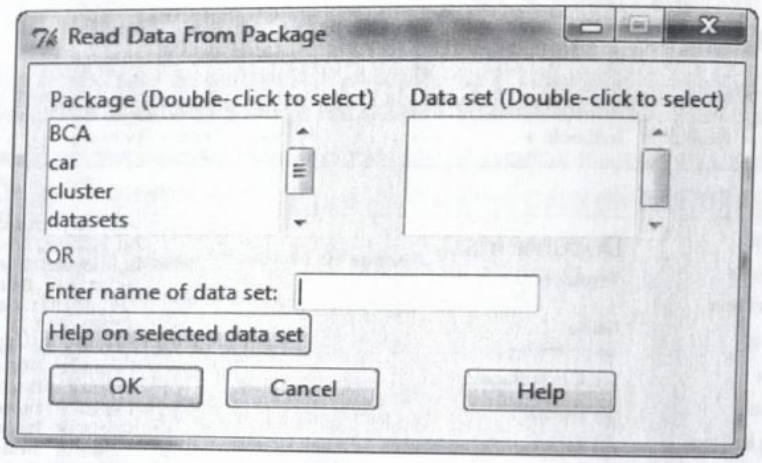


Figure 3.23: Viewing the jack.jill Data Set



	HH.ID	Spending	Children	Income	Employment	Age	
1	7650	411	1 Child	$0-$20k	No female head	No female head	
2	7651	330	1 Child	$0-$20k	Unemployed	29 and under	Some or comp
3	7653	61	1 Child	$0-$20k	Part-time	60 and over	Some or comp
4	7608	949	2 Children	$0-$20k	Part-time	40 to 49	Post-se
5	7643	197	1 Child	$0-$20k	Unemployed	60 and over	Ele
6	7655	115	1 Child	$0-$20k	Full-time	29 and under	Ele
7	7663	920	2 Children	$0-$20k	Full-time	29 and under	Ele
8	7678	645	2 Children	$0-$20k	Part-time	30 to 39	Some or comp
9	7684	1086	2 Children	$0-$20k	Full-time	40 to 49	Post-se
10	7687	100	2 Children	$0-$20k	Unemployed	29 and under	
11	7689	1755	2 Children	$0-$20k	Full-time	40 to 49	Some
12	7690	243	1 Child	$0-$20k	Unemployed	30 to 39	Some
13	7696	620	1 Child	$0-$20k	Part-time	30 to 39	Post-se
14	7700	360	1 Child	$0-$20k	Part-time	29 and under	Some or comp

Figure 3.24: Reading a Data Set in a Package



9.

Close the view window to keep your desktop from getting too cluttered. As indicated at the beginning of this tutorial, R already contains data libraries with many data sets. These data sets are collected into groups called "packages." The package with the data sets we will be using is **BCA**, which was loaded with the RCmdr software when the RCmdr library command was initially entered.

10.

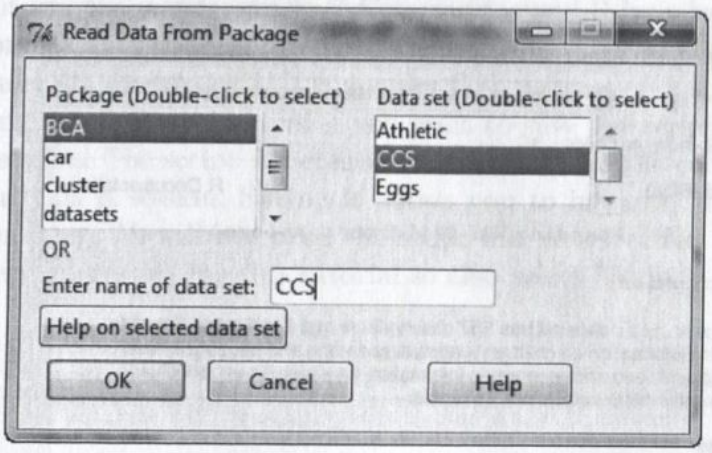
Select **Data** → **Get From** → **R** → **package Read data set from an attached package...**, which will bring up the dialog box shown in Figure 3.24, and shows the packages currently loaded.

Double-click on the package BCA to list the five data sets available in the package in the right box. Scroll down to CCS, and double-click on it (Figure 3.25). Press **OK**. **View this data set** as in step 9. Close the data set view window. Note that the bottom window shows that the data set CCS has 1600 rows and 20 columns, and that the Data Set button below the main menu has changed to CCS, indicating that this is now the "active" data set.

11.

Re-activate jack.jill by **clicking on the Data Set button**—which now has CCS on it—and then **select jack.jill** from the dialog box. After closing the box, jack.jill will be the active data set.

Figure 3.25: Selecting the CCS Data Set2



Help Files

Examine the help documentation for the jack.jill data set by selecting **Data → Clean → Help on active data set (if available)** from the R Commander pull-down menus. This will launch a web browser with the help file displayed. If you did not give the data set the exact name "jack.jill," you will get a warning message indicating that no documentation for the name you gave the data set exists. If this is the case, repeat steps 7 and 8 taking care to give the data set the name that corresponds to the help file. After reading the help file for the jack.jill data set, close the help system window.

12.

You now have two data sets read in and available to work with, jack.jill and CCS. To save them so that they will be available to you for the next tutorial, select **File → Save R workspace as...**, which will bring up a "Save as" dialog box. Navigate to the folder where you wish to save the workspace file, such as the Desktop. Workspaces have the filename extension ".RData" so **type Tutorial3.3.RData** as shown in Figure 3.27. Select **Save**. The workspace file will save your data and (later on in the book) any models you have constructed. It will also allow you to start working again where you have left off; to send files to another computer running Windows, Mac OS X, or Linux; and to share analysis work with the members of your team. When you wish to use a saved workspace file, make sure R is not running, and then start up R and R Commander by double-clicking on the workspace file icon.

Figure 3.26: Data Set Help

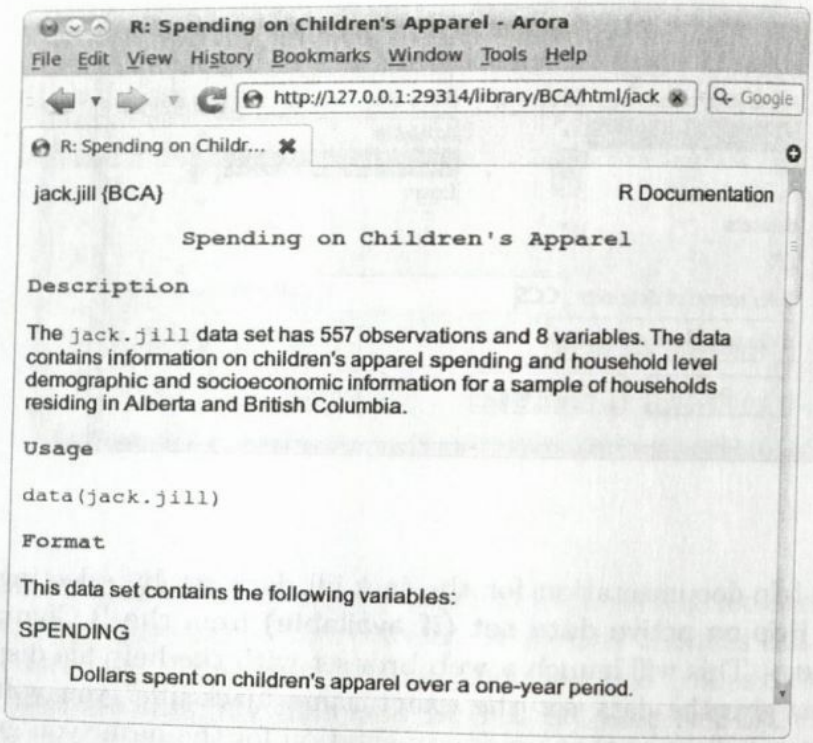
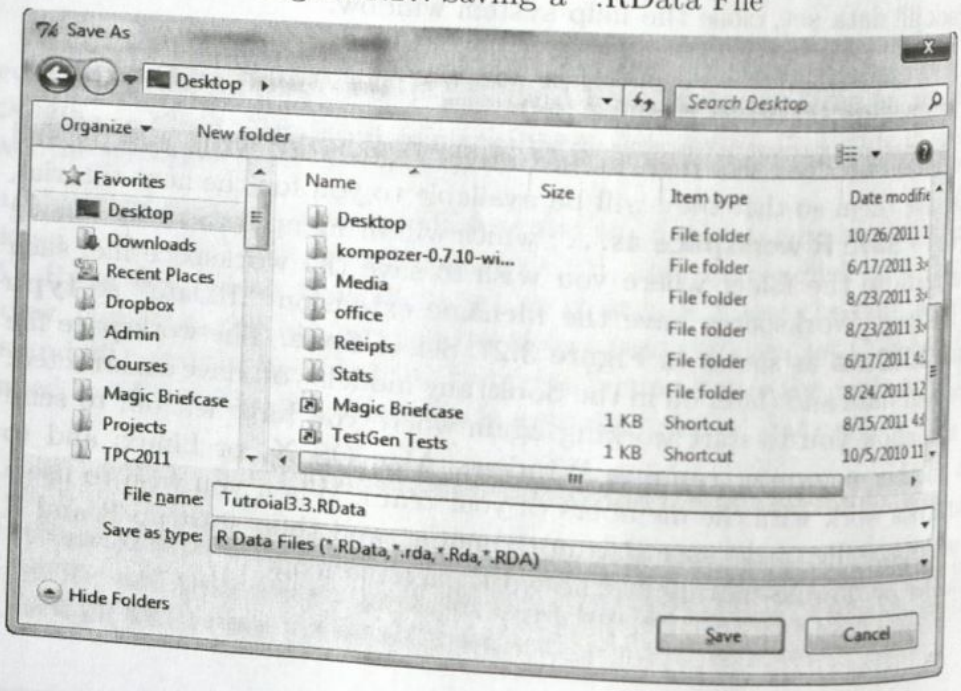


Figure 3.27: Saving a *.RData File



13.

At this point you can quit from the R Commander and R by selecting **File** → **Exit** → **From Commander and R**. R Commander will present a dialog box asking if you really want to exit. If you select **OK**, a series of dialog boxes will appear asking for confirmation and if you want to save the script and output files of the session. The script file contains a record of all the commands you issued during this R session. Saving it allows you to instantly duplicate the entire session later. We will not need the script file, so select **No**. The output file is not very interesting for this tutorial so also select **No** for it.

3.4 Creating Simple Summary Statistics Tutorial

In this tutorial you will see how to use several different tools that will allow you to quickly get an initial understanding of the data you are examining. If you are continuing from the last tutorial, you should be ready to go. Otherwise, **double-click on the Tutorial3.3.Rdata** file you created as part of the last tutorial. (Do not start R from its own icon when you want to start from a saved workspace.) **Enter library(RcmdrPlugin.BCA)** in the R Console to Start R Commander. Make jack.jill the active data set by using the **Data Set** button.

1.

There is one "data cleaning" chore that should be done for most data sets. The jack.jill data set contains the variable `HH.ID`, and is an example of a case identifier (or observation identifier). For the jack.jill data set, it is a household identification number. Since these "variables" are merely identifiers, they will not actually be used in any analyses that we will do. We can ensure that we will never accidentally use it as an analysis variable by setting `HH.ID` as a record-name data type now. This step should ALWAYS be taken with a new data set that has identification variables — and most will. Click on **Data** → **Clean** → **Set record names...**, which bring up the dialog box shown in Figure 3.28.

2.

Within this dialog box **scroll to and click on `HH.ID`** with your mouse to highlight it, and then press **OK**. `HH.ID` has now become an observation identifier, and no longer appears as a variable in jack.jill. You can check this by

Figure 3.28: The Set Record Names Dialog

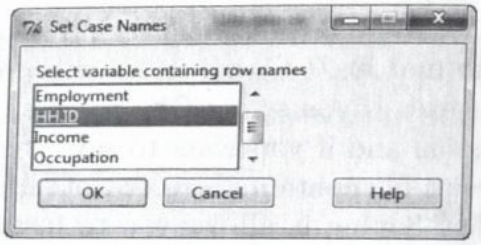


Figure 3.29: Variable Summary for the jack.jill Data Set

```
> variable.summary(jack.jill)
                Class %.NA Levels Min.Level.Size     Mean       SD
Spending      integer    0     NA              NA 784.377 761.6432
Children       factor    0      4              15       NA       NA
Income         factor    0      8              42       NA       NA
Employment     factor    0      4               9       NA       NA
Age            factor    0      6               5       NA       NA
Education      factor    0      7               2       NA       NA
Occupation     factor    0      6               8       NA       NA
Birth.Country  factor    0      6               7       NA       NA
```

clicking the **View data set** button. Close the data set window after viewing to reduce desktop clutter.

3.

To get information on all of the variables in this data set, select **Data** → **Clean** → **Summarize Variables**, which will cause the table in Figure 3.29 to appear in your R Commander Output window.

Variable Attributes

This table lists a number of attributes for each variable in the data set. Inspecting this table is a critical part of the data understanding phase, and helps with identifying data problems and hence the data preparation steps necessary to fix them.

The variable names appear in the first column. The second column (Class) gives the class of each variable, which incorporates both the measurement scale and the internal computer coding used for the variables. The combination of scales and coding can initially be confusing, but there are only four classes, and getting comfortable with them is essential for data analysis. The possible classes of variables are *numeric*, *integer*, *factor*, and *character*. *Numeric* and *integer* variables are ratio-scaled numbers (integer variables do not have fractional values, but for our purposes can be treated as numeric variables). A *factor* is a categorical variable. The summary table does not distinguish between ordered (ordinal-scaled) and unordered (nominal-scaled) factors, but

we will usually treat them as nominal. From this, you can conclude that we will be working primarily with ratio and nominal variables. A *character* variable only contains text as labels and are not variables that can be used in an analysis. This is fine if the variables are merely record identifiers, but if they are variables that need to be used in an analysis, we will need to convert them. We will see how to convert them later, usually to factors.

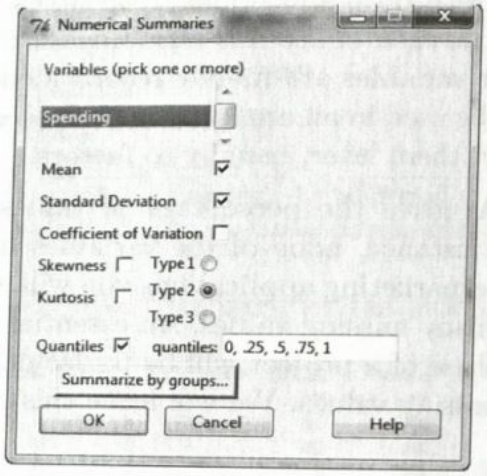
The third column (%.NA) gives the percentage of values that are missing for that variable. In this instance, none of the variables has missing values. However, in most database marketing applications you will encounter data sets that have variables with many missing values. An essential data cleaning step in the data preparation phase of a project will be to decide what, if anything, needs to be done about missing values. We will leave this important question for later.

The third and fourth columns (Levels and Min.Level.Size) provide information for factor variables (thus `SPENDING`, a numeric variable, has no information, `NA`, in this column). The Levels column indicates the number of categories or "levels" for a factor variable. For example, a factor "gender" would have two levels, male and female. Min.Level.Size indicates the number of records for the category or "level" that has the fewest records. A data set consisting of 14 records, with 11 males and 3 females, would have the a minimum level size of 3 for gender. These columns are provided because factors with many levels and levels with too few records can cause problems during the model building phase. Moreover, any observed effects associated with levels with few records are statistically unreliable. What qualifies as too few levels depends on the analysis details, but as a start we should at least note that `EMPLOYMENT`, `AGE`, `EDUCATION`, `OCCUPATION`, and `BIRTHCNTRY` are all likely cases. One common thing to do in the data preparation phase is to combine two or more factor levels into one level, which thus has more records and increases the minimum level size, and which also decreases the number of factor levels. In the next tutorial you will see exactly how to do this using R Commander.

Exercise: How many categories for age are defined in these data? The age category with the fewest households has how many households?

The fifth and sixth columns (Mean and SD) contain the mean and standard deviation for each numeric variable in the data set (the values are missing for factor and character variables). The mean is useful as an indication of the magnitude of a variable (in the tens, hundreds, thousands, etc.). The standard deviation provides a measure of the spread or range of the data. One thing to be on the lookout for is a variable with a standard deviation equal to zero. A variable that has a standard deviation of zero is not to be a variable at all, it is a constant (i.e., it only has one single value in all records), and the inclusion of this variable as input to a number of data mining methods can

Figure 3.30: The Numerical Summary Dialog



result in problems. Generally, we either delete constants from the data set, or determine why the variable is a constant since it suggests that there may have been a problem in prior processing of the data.

4.

In the next tutorial we will see a number of tools to get descriptive statistics for factor variables. For the remainder of this tutorial we will look at simple summary statistics for numeric variables. We can gain some additional information about the SPENDING variable by selecting **Explore and Test** → **Summarize** → **Numerical summaries. . .**, which will generate the dialog box shown in Figure 3.30.

5.

The only numeric variable in jack.jill is SPENDING, so it is the only variable shown in the **Variables (pick one or more)** selection box. To make sure it is properly selected, click on the variable name. We will leave the other options as they are. Right now we will only do a summary of SPENDING across all households, so press **OK** in order to create the output table shown in Figure 3.31.

You will notice that the mean and standard deviation ("sd" in the new output) is the same as it is in Figure 3.29 (761.6432). The additional information is the values of the variables at each of its quartiles. The 0% quartile is the minimum value of SPENDING in the data set ($13), the 50% quartile is the median value of SPENDING ($585), and the 100% quartile is the maximum value of SPENDING ($5940) in the data set. The large gap between the 75%

Figure 3.31: A Numerical Summary of SPENDING

```
> numSummary(jack.jill[,"Spending"], statistics=c("mean", "sd", "quantiles"),
+   quantiles=c(0,.25,.5,.75,1))
    mean       sd 0% 25% 50% 75% 100%   n
784.377 761.6432 13 309 585 965 5940 557
```

and 100% quartiles (relative to the other inter-quartile gaps) indicates that the distribution of SPENDING is likely to be highly skewed, with a very small percentage of households having extremely high spending levels. The fact that the mean is much higher than the median is also a result of such skewing. The last value ("n") in the summary is the number of non-missing values for the variable. Since the data set contains 557 records, this again confirms that SPENDING has no missing values.

Exercise: Return to the numerical summaries dialog box, but this time select the **Summarize by groups... button**. This will bring up a dialog box that lists the categorical variables. In this box, **select CHILDREN,** then OK, and then OK again in the summary box. Compare the output with the previous output, and explain what you see.

6.

Because there is only one numeric variable in the jack.jill data set, it is not useful for the next set of tools we want to illustrate. Consequently, at this point load the CCS data set. To do this, press on the **Data Set** button and select CCS as the active data set. In a later tutorial we will describe the CCS data set in much greater detail. For now it is important to know that this data set relates to fundraising activities undertaken by a Canadian Charitable Society (or CCS). Some of the variables in the data set capture past behavior on the part of individuals with respect to their giving to the CCS, while other variables are derived from Census data and are averages of demographic attributes in the individual's neighborhood.

7.

At this point we want to gain an understanding of how the variables that capture past giving behavior relate to one another. In particular, we want to look at the correlations between these variables. Select **Explore and Text** → **Summarize** → **Correlation matrix...** to generate the dialog box shown in Figure 3.32.

Figure 3.32: The Correlation Matrix Dialog

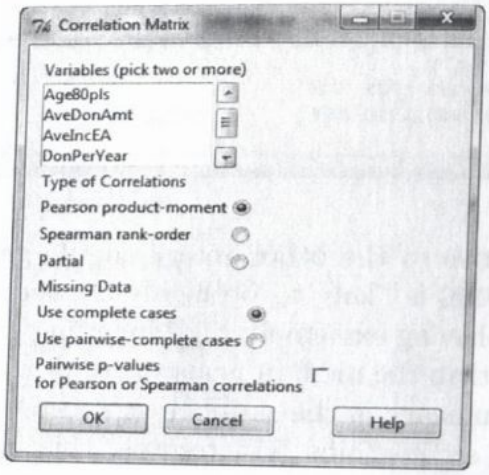


Figure 3.33: Correlation Matrix Results

```
> cor(CCS[,c("AveDonAmt","DonPerYear","LastDonAmt","YearsGive")],
+   use="complete")
             AveDonAmt DonPerYear LastDonAmt   YearsGive
AveDonAmt   1.00000000  0.1262545 0.86586622  0.03632848
DonPerYear  0.12625451  1.0000000 0.09886880 -0.45153640
LastDonAmt  0.86586622  0.0988688 1.00000000  0.05135673
YearsGive   0.03632848 -0.4515364 0.05135673  1.00000000
```

8.

The four numerical variables that capture past giving behavior are `AveDonAmt` (the average amount given per donation to the CCS by a giver), `DonPerYear` (the number of donations per year made by a giver), `LastDonAmt` (the amount of a giver's last donation), and `YearsGive` (the number of years since the giver's first donation to the CCS). Within the **Variables (pick two or more)** selection box, **select these four variables**. You can select multiple variables by pressing the control key (Ctrl on most keyboards) when clicking on a variable with a mouse. Once you have selected these four variables, press the **OK** button to produce the correlation matrix shown in Figure 3.33.

Correlations

The correlation matrix indicates that `AveDonAmt` and `LastDonAmt` have a correlation of 0.866, which is high (correlation coefficients are bounded between -1 and 1). It is not surprising that the higher the *average* donation amount, the higher the *last* donation amount. In subsequent tutorials on predictive modeling, we will see that using two highly correlated (either postively or negatively) variables such as these as predictors can cause problems, and that

we need to be aware of and manage those problems. The other two variables that are somewhat (negatively) correlated are `DonPerYear` and `YearsGive`, with correlation −0.452: the more donations per year, the less time the individual has been a donor, perhaps somewhat unexpected. This correlation is probably not strong enough to cause problems for predictive models using both of these variables as predictors.

Exercise: Mr. White donates to the society three times each year. Mrs. Brown donates twice each year. Who is more likely to have donated first?

At this point you can either quit from R and R Commander, or go directly to the next tutorial. If you quit, you may save the R workspace as before. Note that in this tutorial we have not altered either the jack.jill or CCS data sets. Therefore, as long as you saved the R workspace after the "Reading Data into R Tutorial," there will be no change and there is no need to save the R workspace again.

3.5 Frequency Distributions and Histograms Tutorial

In this tutorial you will learn to use R to explore one variable at a time, called *univariate* (one variable) analysis. The two tools are frequency distributions and histograms. We will also learn a handy tool for converting between variable types. Frequency distributions and histograms are intended to be used with categorical variables (i.e., variables that have either a nominal or ordinal scale), which R calls "factor variables." The example used in this tutorial is the amount of money spent on children's apparel by 557 households for a one-year period, which is a ratio-scaled continuous variable that we will learn how to convert into a categorical variable. This will be useful in the future as there is often a need to break up the range of a continuous variable into a number of discrete categories. This is often referred to as "binning" or "bucketing" a continuous variable, and R Commander provides a very nice tool for this.

1.

If you did not quit R and Rcmdr after completing the last tutorial, continue with the next step. If you did quit, restart R by **double-clicking on the workspace file Tutorial3.3.RData**. If you saved it on your desktop, it will appear with a blue R icon.

Figure 3.34: Select Data Set Dialog

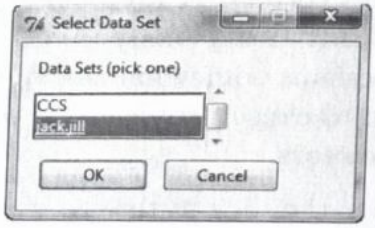
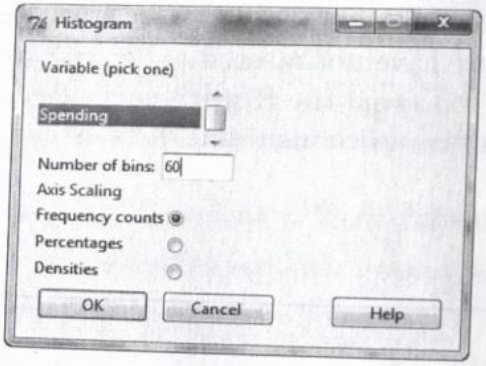


Figure 3.35: Histogram Dialog



2.

Once the R Commander window is up, click on the data set button (the button will have either the label **No active data set** or **CCS** on the R Commander toolbar). The button brings up the **Select Data Set** dialog box shown in Figure 3.34. The two data sets read into R from the first tutorial are available. Select jack.jill by **clicking on it, and then on OK**.

3.

We are ready to make a histogram of spending on children's apparel across households. To access R Commander's **Histogram** dialog box, use the pull-down menu **Explore and Test** → **Visualize** → **Histogram...**, which brings up the **Histogram** dialog box shown in Figure 3.35.

4.

As you have probably guessed, Figure 3.35 is the "filled-in" dialog box, with the number of bins set to 60, while the one on your screen is set to auto. The only variable that appears in the variable list is `SPENDING`. The reason for this is that `SPENDING` is the only continuous (coded as "numeric') variable in the jack.jill data set. The histogram tool is only for continuous variables. For

factor variables, a similar visualization tool is a bar graph. We will use the bar graph tool later in this tutorial.

5.

The variable Spending is automatically selected. The **Number of bins** field allows you to enter the number of bins (or "buckets") to use in breaking up the range of the SPENDING variable. The values for this variable range from \$13 to nearly \$6000, thus entering a value of 60 in this field will break the range into roughly \$100 increments, and since we have 557 records, will give an average of about 9 records in each bin. This will give us a pretty good visual display of the distribution of children's apparel spending across the households in our sample. **Enter the number 60** in the "Number of bins:" field, and then press the **OK** button. R will create a graphics window on your desktop containing the histogram shown in Figure 3.36.

Exercise: By inspecting the histogram, can you predict whether the average or the median of spending will be higher? Explain, and describe each in terms of its application to ordinal and interval data.

Figure 3.36: Children's Apparel Spending Histogram

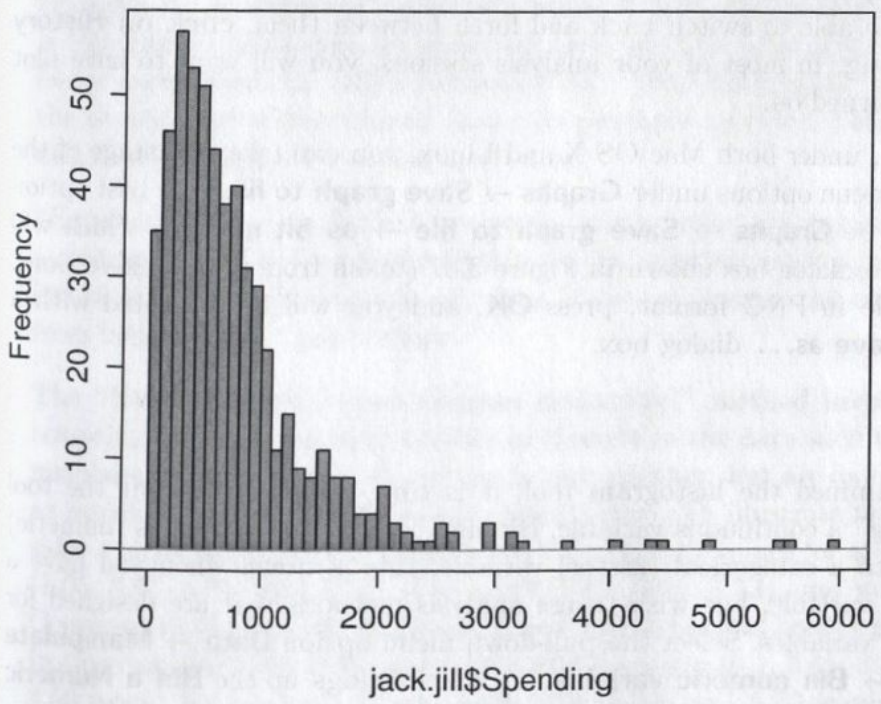
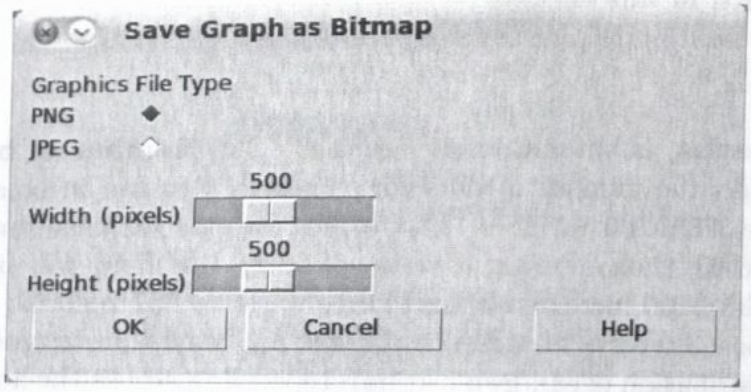


Figure 3.37: Save Plot Dialog



6.

Copying, Pasting, and Recording Graphics

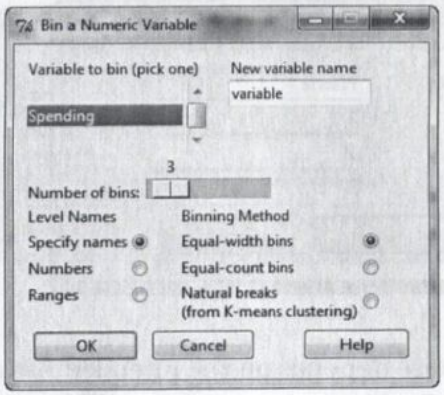
Under Windows, a quick way to copy a graph to the clipboard for pasting into another document is to **right-click on the graph** and from the context menu select **Copy as a Metafile...** (or as a bitmap). Try this out now since it will be useful throughout the rest of the book. Open a Word document, and then paste the graph into the document. When you create you next plot, the current plot will be written over and lost. If you want to save all of your plots and be able to switch back and forth between them, **click on History → Recording**. In most of your analysis sessions, you will want to have plot recording turned on.

As an aside, under both Mac OS X and Linux, you can take advantage of the pull-down menu options under **Graphs → Save graph to file**. The best option is likely to be **Graphs → Save graph to file → as bit map...**, which will bring up the dialog box shown in Figure 3.37 (taken from the Linux version). Save the file in PNG format, press **OK**, and you will be presented with a standard **Save as...** dialog box.

7.

Having examined the histogram tool, it is time to take a look at the tool for "binning" a continuous variable. Binning converts a continuous (numeric) variable into a categorical (factor) variable; this is useful when you have a continuous variable, but wish to use analysis methods that are designed for categorical variables. Select the pull-down menu option **Data → Manipulate variables → Bin numeric variable...**, which brings up the **Bin a Numeric Variable** dialog box shown in Figure 3.38.

Figure 3.38: Bin Numeric Variable Dialog



8.

Binning Methods

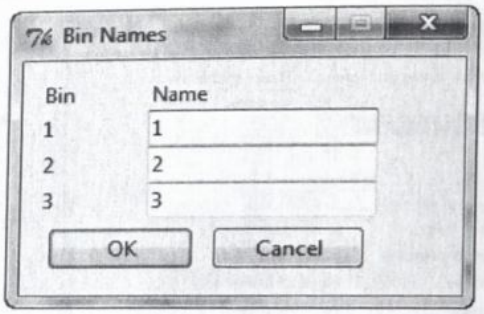
R Commander's binning tool offers three different methods of dividing a continuous variable into a fixed number of categories.

- The "Equal-width bins" method uses the same binning method as the histogram. It divides variable values into a set of equal size ranges. For example, if the minimum value of a variable is 0, the maximum value is 100, and 10 categories are specified, then the first level of the binned factor corresponds to values between 0 and 10 of the original variable, the second level of the binned factor corresponds to values between 10 and 20 of the original variable, and so on.

- "Equal-count bins" finds the appropriate break points along the range of a continuous variable so that each bin has an approximately equal number of cases, or individuals in our data. Ties can prevent the categories from being exactly equal in size.

- The "Natural breaks (from K-mean clustering)" method involves attempting to find a specified number of clusters in the data such that the members of each cluster are similar to one another, but are as different as possible from the members of other clusters. To illustrate this, consider a situation in which we have the following 10 values for a variable (1.05, 5.63, 5.71, 3.45, 3.47, 3.50, 1.10, 1.11, 5.75, 5.73), and we ask R Commander to create three groups based on the natural breaks method. In this instance 1.05, 1.10, and 1.11 would be in group 1, 3.45, 3.47, and 3.50 would be in group 2, and 5.63, 5.71, 5.73, and 5.75 would be in group 3.

Figure 3.39: Specifying Level Names Dialog



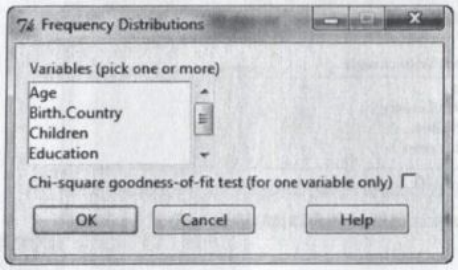
The choice of a method to use depends on the intended use of the new categorical variable. For example, in the next tutorial we will learn to use *contingency tables,* a quick, common, and useful analysis method, but only for categorical variables. A limitation of contingency tables is that we should avoid having cells in the table with very few observations, otherwise our statistical tests may not be meaningful. In converting our continuous variables to categorical, the "Equal-count bins" method is least likely to result in contingency table cells with small counts. We also want to keep the number of levels of the categorical variable small, since more levels means fewer observations in each level. We will start by creating a new categorical variable from the continuous variable by binning the spending data into roughly three equal groups (e.g., a low, medium, and high spending group).

Select `SPENDING` in the "Variable to bin (pick one)" field. In the "New variable name" field **enter** `Spend.Cat` (so you will know this new variable is the categorical version of the spending variable). Use the slider to **select 3** as the number of bins. **Select the "Specify names"** radio button among the "Level Names" choices, and **select the "Equal-count bins"** radio button among the "Binning Method" choices. Once you have done all of this press **OK**. A second dialog box (shown in Figure 3.39) will appear so that you can assign level names that are more meaningful, thus making your output easier to interpret.

9.

The data values of the original variable (`SPENDING`) are sorted from smallest to largest. As a result, **in the first field replace "1" with "Low," in the second field replace "2" with "Medium," and in the third field replace "3" with "High." Press OK** and the `Spend.Cat` variable will be created. To confirm the creation of the new variable, **click on the View Data Set** button, and scroll to the right to the end of the variable list where `Spend.Cat` will have been added. **Close the View Data window.**

Figure 3.40: Frequency Distribution Dialog



10.

As indicated above, the three groups of values for `Spend.Cat` will be of *roughly* equal sizes. We can create a frequency distribution of `Spend.Cat` to see just how "rough" equal sizes are. To do this, use the pull-down menu option **Explore and Test** → **Summarize** → **Frequency distributions...**, which will bring up the dialog box shown in Figure 3.40.

11.

In this dialog scroll down and **select `Spend.Cat` as Variable (pick one) and press OK**. The results of the frequency distribution will appear in R Commander's output window, and should look identical to Figure 3.41. The first table indicates the number of households in each bin, and the second, the percent of the total households in each bin. The frequency distribution indicates that the groups are not of exactly equal sizes, but they are close enough for our purposes.

Figure 3.41: Frequency Distribution of Binned Spending

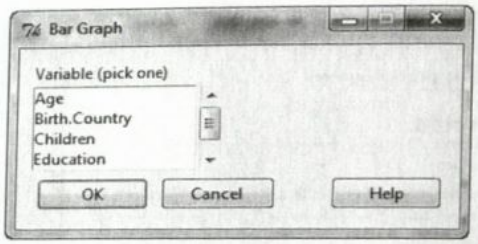
```
   Low Medium   High
   188    183    186

> round(100*.Table/sum(.Table), 2)  # percentages for Spend.Cat

   Low Medium   High
 33.75  32.85  33.39

> remove(.Table)
```

Figure 3.42: Bar Graph Dialog



12.

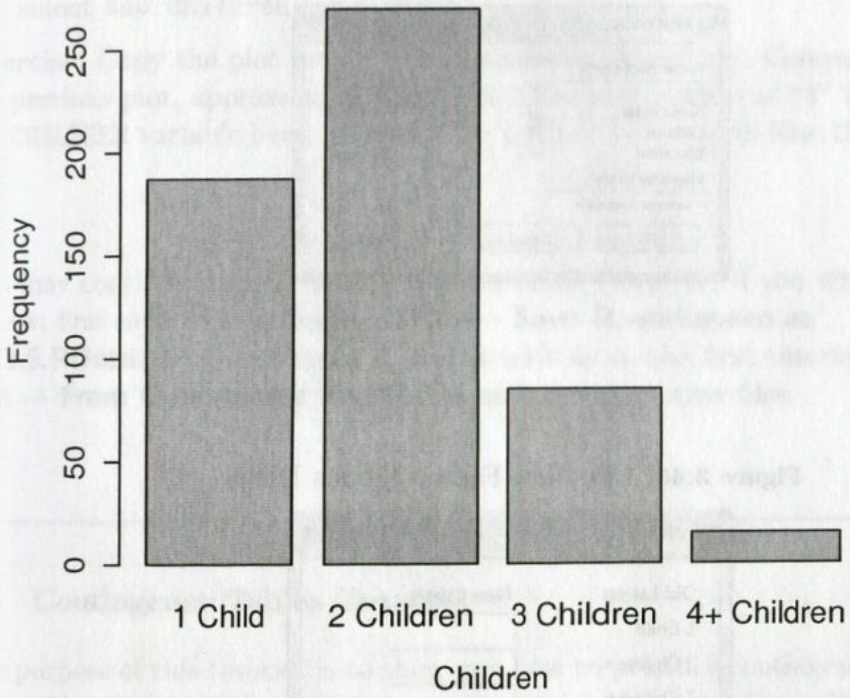
The final tool to be introduced in this lab allows you to take an existing categorical variable and "re-label" factor levels, which allows us not only to change the names, but to combine levels, thereby reducing the total number of levels. There are a number of reasons why we might want to do this, one of which we have already discussed. Specifically, fewer levels means more observations in each level, which may be needed for contingency table analysis of relations between variables. One variable that seems likely to be related to household spending on children's clothes is the number of children in a household. The current `CHILDREN` variable has four levels: "1 child," "2 children," "3 children," and "4+ children." Creating a bar graph of `CHILDREN` illustrates the potential problem with this variable. To create this bar graph select the pull-down menu option **Explore and Test** → **Visualize** → **Bar graph**..., which brings up the dialog box shown in Figure 3.42. In this dialog box **select** `CHILDREN` and **press OK** to create the bar graph shown in Figure 3.43. If you don't see it, the graphics window may be behind other windows. You can bring it to the front from the Windows taskbar, which is normally located at the bottom of your screen.

Exercise: Copy the plot into a word processing document.

13.

The bar graph reveals that there are very few households in our sample with four or more children, causing any inferences about this group's behavior in the entire population to be very imprecise and of limited usefulness for decision making. Therefore, it makes sense to combine the last two levels so that `CHILDREN` has only three levels (i.e., "1 child," "2 children," and "3+ children"). We have two tools that can do this. One of these tools, Recode variable, is a very powerful, general purpose recoding tool, which we will use in a later tutorial. The second tool (which can be accessed with the pull-down menu option **Data** → **Manipulate Variables** → **Relabel factor levels**...) is a more limited tool specifically designed for categorical variables, for relabeling

Figure 3.43: Bar Graph of the Number of Children Present



and combining factor levels. It will bring up the dialog box shown in Figure 3.44.

14.

In this dialog box select `CHILDREN` for the "Factor (pick one)," enter `New.Children` in the "Name for factor" field, and press **OK**. A second dialog box (shown in Figure 3.45) will then appear.

15.

In the **New Labels** dialog box **enter "1 Child" in the first field, "2 Children" in the second field, and "3+ Children" in both the third and fourth fields**. When you are done, your **New Labels** dialog box should appear as the one shown in Figure 3.46. When it does, press **OK** to create the new factor `New.Children`. View the data set and confirm the changes from the `CHILDREN` column to the `New.Children` column.

Figure 3.44: Relabel a Factor Dialog

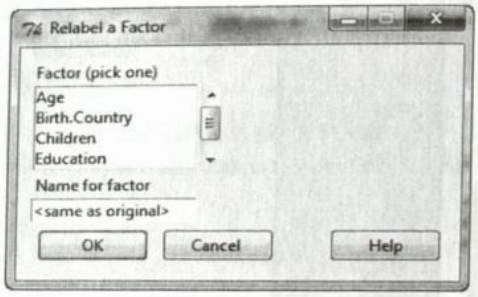


Figure 3.45: The New Factor Names Dialog

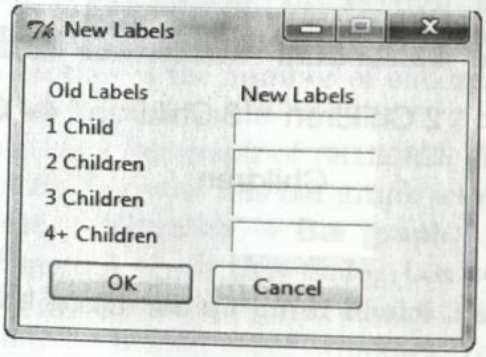
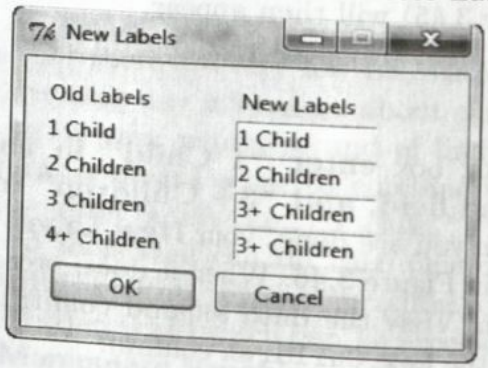


Figure 3.46: The Completed New Factor Labels Dialog



16.

Plot the graph again, select **Explore and Test → Visualize → Bar graph**..... and **select** `New.Children` for plotting.

Exercise: Copy the plot into a word processing document. Comparing with the previous plot, approximately how much has the number of "3" families in the `CHILDREN` variable been increased by in the "3+" case in `New.Children`?

17.

You may continue directly to the next tutorial. However, if you wish to quit R now, first save your workspace **(File → Save R workspace as**... as **Tutorial3.5.RData**. Then exit from R and Rcmdr as in the first tutorial: **File → Exit → From Commander and R**. Do not save any other files.

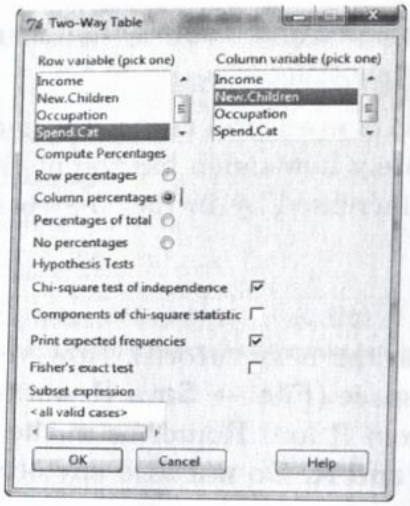
3.6 Contingency Tables Tutorial

The purpose of this tutorial is to show you how to produce contingency tables. To avoid any potential confusion, contingency tables are also called "cross-tabulations" and "cross-tabs." They are the simplest type of *multivariate* analysis (i.e., methods for studying the relationships among multiple variables) available and are common in market research. This tutorial will only consider two-way (two variable) contingency tables. Interpreting three-way tables is often difficult, and four-way or higher-order tables are essentially impossible to interpret. R Commander will also calculate a chi-square test statistic to help you evaluate the statistical independence of the two variables under study. You will also learn how to change the level order in a factor variable, to create more easily interpreted and presentation-friendly contingency tables.

1.

If you are continuing directly from the last tutorial without exiting R, go to the next step of this tutorial. If you exited R and saved the workspace file in the last tutorial, start RCmdr by **double-clicking on the workspace file Tutorial3.5.RData**. In R Commander, **activate the jack.jill data set using the Data Set button.**

Figure 3.47: The Contingency Table Dialog



2.

Use the pull-down menu option **Explore and Test → Test → Contingency tables → Two-way table...** to bring up the dialog box shown in Figure 3.47.

3.

The first thing we will examine is the relationship between spending on children's apparel and the number of children present in the household. We should, of course, have some rough idea (i.e., "theory") about how these variables should be related! In the dialog box **select `Spend.Cat` as the "Row variable (pick one)" and `New.Children` (the condensed version of `CHILDREN`) as the "Column variable (pick one)" variable. Next select the "Column percentages" radio button from the "Compute Percentages"** choices. This will cause two different tables to be produced. The first table contains the raw counts (number of households) with a given level of the two factors (e.g., the number of "low" spending households with only one child present), while the second table will provide the percentage of high, medium, and low children's apparel spenders for a given number of children present, as given by each level of `New.Children`. Percentage figures help interpret a contingency table since it compensates for differences in the total number of households of different types (in this instance with differing numbers of children). By default the "Chi-square test of independence" box is checked to give us that useful statistic. **Next, check the "Print expected frequencies" option** which prints out the number of cell counts that would be *expected* in each cell *if* there was no relation between the two variables, that is, assum-

Figure 3.48: Children's Apparel Spending vs. Number of Children

```
Output Window                                                    Submit
> .Table
          New.Children
Spend.Cat 1 Child 2 Children 3+ Children
   Low          99          78          11
   Medium       62          89          32
   High         26         103          57

> colPercents(.Table) # Column Percentages
          New.Children
Spend.Cat 1 Child 2 Children 3+ Children
   Low        52.9        28.9          11
   Medium     33.2        33.0          32
   High       13.9        38.1          57
   Total     100.0       100.0         100
   Count     187.0       270.0         100

> .Test <- chisq.test(.Table, correct=FALSE)

> .Test

        Pearson's Chi-squared test

data:  .Table
X-squared = 77.4454, df = 4, p-value = 6.053e-16


> .Test$expected # Expected Counts
          New.Children
Spend.Cat  1 Child 2 Children 3+ Children
   Low     63.11670   91.13106    33.75224
   Medium  61.43806   88.70736    32.85458
   High    62.44524   90.16158    33.39318
```

ing the variables are independent. We will not be using "Fisher's exact test," which provides an alternative hypothesis test of independence for a two-by-two contingency table (a table with two variables where each variable has only two levels). Among the "Hypothesis Tests" options, select only the chi-square test. Once you have done all this press **OK**, and the results of the table (shown in the R Commander output window) should look like Figure 3.48.

4.

The second table indicates that 52.9% of households with only one child present are in the lowest spending category, while only 13.9% are in the highest spending category. In contrast, only 11% of households with three or more children are in the lowest spending category, while 57% are in the highest spending category. For comparison, the final table shows the expected counts *if there was no relationship* between the variables. Households would then be expected to be distributed across the spending categories in exactly the same proportions as the overall distribution, which in this case is equal thirds. The actual distribution in the above tables is very different from the lower table. The extremely small p-value (6.053 times 10^{-16}) of the chi-squared test is a measure of this difference, indicating that it is essentially certain that in the

Figure 3.49: Children's Apparel Spending vs. Income

```
          Income
Spend.Cat $0-$20k $100k+ $20k-$30k $30k-$40k $40k-$50k $50k-$60k $60k-$75k $75k-$100k
   Low          31      5        23        34        33        26        25         11
   Medium       21     14        15        22        32        24        26         29
   High         17     23        10        22        21        28        23         42

> colPercents(.Table) # Column Percentages
          Income
Spend.Cat $0-$20k $100k+ $20k-$30k $30k-$40k $40k-$50k $50k-$60k $60k-$75k $75k-$100k
   Low       44.9   11.9      47.9      43.6      38.4      33.3      33.8       13.4
   Medium    30.4   33.3      31.2      28.2      37.2      30.8      35.1       35.4
   High      24.6   54.8      20.8      28.2      24.4      35.9      31.1       51.2
   Total     99.9  100.0      99.9     100.0     100.0     100.0     100.0      100.0
   Count     69.0   42.0      48.0      78.0      86.0      78.0      74.0       82.0

> .Test <- chisq.test(.Table, correct=FALSE)

> .Test

        Pearson's Chi-squared test

data:  .Table
X-squared = 46.0957, df = 14, p-value = 2.705e-05
```

whole population there is a relationship between household spending on children's apparel and the number of children present in the household (provided we have carefully taken a random sample, of course!). We have seen the nature of this relationship in the column percentages table. In sum, it is safe to conclude that households with a greater number of children present tend to spend more on children's apparel compared with those with fewer households. Of course, from a common sense view, if we did not find that this was the case, we would begin to question the validity of our sample!

Exercise: Highlight and copy the table into a word processing document.

5.

Create a second contingency table by repeating step 2, but this time use `INCOME` as the column variable. You can widen the RCmdr window before creating the contingency table to avoid text wrap around. Your results should look like those in Figure 3.49.

6.

One thing to notice in Figure 3.49 is that the column for the highest income group ($100k+) is out of place, falling second, rather than last, which can make interpreting and presenting the table more complicated than need be. The problem is due to the fact that, by default, R orders and prints the levels of a factor alphabetically, and "$100k+" comes before "$20k–$30k." However, it is easy to solve this problem by using the pull-down menu option **Data** → **Manipulate Variables** → **Reorder factor levels...**, which will bring up the dialog box shown in Figure 3.50.

Figure 3.50: Reorder Factor Levels Dialog

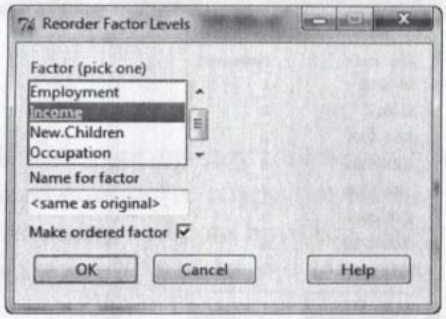
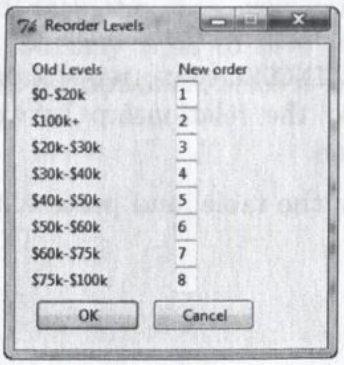


Figure 3.51: The Second Reorder Levels Dialog



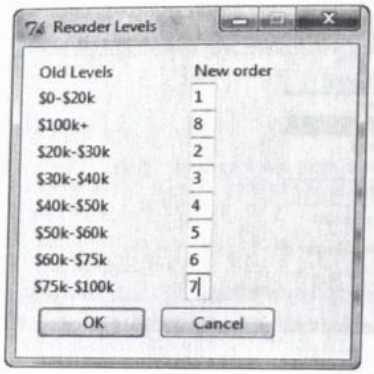
7.

Select `INCOME` as **"Factor (pick one),"** and keep the factor's original name. An ordered factor is an ordinal scaled variable, while an unordered factor is a nominally scaled variable. Income is really an ordinal scaled variable, so the **"Make ordered factor" option should be checked. Finally, press OK** and **select Yes** when asked to Overwrite Variable. The dialog box shown in Figure 3.51 will appear.

8.

The **Reorder Levels** dialog box allows us to explicitly reorder the factor levels. The "$0- $20k" level is correctly placed in the first position, but the remaining levels are out of place. For instance, the "$100k+" level should be in the eighth (not second) position, and the "$20k–$30k" level should be in the second (not third) position. Reorder the factor the way it should be. When you are done, this dialog box should look like the one shown in Figure 3.52. When it does, press **OK**.

Figure 3.52: The Completed Reorder Factor Level Dialog



9.

Re-create the `Spend.Cat` and `INCOME` contingency table using the reordered `INCOME` variable and examine the relationship between children's apparel spending and household income.

Exercise: Highlight and copy the table, and paste it into a word processing document.

10.

Next we explore three additional relations with Spending: (1) `Spend.Cat` and `EDUCATION`; (2) `Spend.Cat` and `AGE`; and (3) `Spend.Cat` and `BIRTHCNTRY`. First explore these three variables individually **using frequency distributions.** Note that there are levels with very small counts, which will not be great for our tables. Also (if you inspect the help file for this data) you will see that these variables are specifically for the *female head of the household.* In the real world there are households with children and no mother. Therefore there are no data for these variables in these households. The frequency distribution shows that this occurs 9–10 times. That is few enough that we can quickly remove those households individually. Click on the **View data set** button to see which rows the offending variable occurs in. For example the very first household has no female head. As long as we have set the variable HH.ID as the case name, you will see that this record is named "7650," which R interprets as a character string rather than a number because of the quotes surrounding it. Select **DATA→ Clean→ Remove Selected Records** to bring up a dialog box. You can delete the record by its row position, in this case 1, or by its name, in this case "7650," with quotes. Enter "7650" in quotes. Leave the data set name the same and Click **OK.** The first row of the data will now disappear. **Repeat the exercise for the remaining households that have no female head.**

Now with your cleaner data, create your contingency tables.

Exercise: Copy and paste the three tables into the word processing document.

11.

Exercise: Based on the five contingency tables you have done, the chi-square statistic, and the expected counts (to assess the validity of the statistic) which demographic and socioeconomic factors have the largest impact on household spending for children's apparel? Which appear to have no impact?

12.

Since we are now finished with the data, you can exit without saving the workspace file. However, you may wish to save your output file for future reference using **File → Save output as...** since it will contain the contingency tables you have generated.