CHAPTER 1

Introduction: Data-Analytic Thinking

Dream no small dreams for they have no power to move the hearts of men.

—Johann Wolfgang von Goethe

The past fifteen years have seen extensive investments in business infrastructure, which have improved the ability to collect data throughout the enterprise. Virtually every aspect of business is now open to data collection and often even instrumented for data collection: operations, manufacturing, supply-chain management, customer behavior, marketing campaign performance, workflow procedures, and so on. At the same time, information is now widely available on external events such as market trends, industry news, and competitors' movements. This broad availability of data has led to increasing interest in methods for extracting useful information and knowledge from data—the realm of data science.

The Ubiquity of Data Opportunities

With vast amounts of data now available, companies in almost every industry are focused on exploiting data for competitive advantage. In the past, firms could employ teams of statisticians, modelers, and analysts to explore datasets manually, but the volume and variety of data have far outstripped the capacity of manual analysis. At the same time, computers have become far more powerful, networking has become ubiquitous, and algorithms have been developed that can connect datasets to enable broader and deeper analyses than previously possible. The convergence of these phenomena has given rise to the increasingly widespread business application of data science principles and data-mining techniques.

Probably the widest applications of data-mining techniques are in marketing for tasks such as targeted marketing, online advertising, and recommendations for cross-selling.

Data mining is used for general customer relationship management to analyze customer behavior in order to manage attrition and maximize expected customer value. The finance industry uses data mining for credit scoring and trading, and in operations via fraud detection and workforce management. Major retailers from Walmart to Amazon apply data mining throughout their businesses, from marketing to supply-chain management. Many firms have differentiated themselves strategically with data science, sometimes to the point of evolving into data mining companies.

The primary goals of this book are to help you view business problems from a data perspective and understand principles of extracting useful knowledge from data. There is a fundamental structure to data-analytic thinking, and basic principles that should be understood. There are also particular areas where intuition, creativity, common sense, and domain knowledge must be brought to bear. A data perspective will provide you with structure and principles, and this will give you a framework to systematically analyze such problems. As you get better at data-analytic thinking you will develop intuition as to how and where to apply creativity and domain knowledge.

Throughout the first two chapters of this book, we will discuss in detail various topics and techniques related to data science and data mining. The terms "data science" and "data mining" often are used interchangeably, and the former has taken a life of its own as various individuals and organizations try to capitalize on the current hype surrounding it. At a high level, *data science* is a set of fundamental principles that guide the extraction of knowledge from data. Data mining is the extraction of knowledge from data, via technologies that incorporate these principles. As a term, "data science" often is applied more broadly than the traditional use of "data mining," but data mining techniques provide some of the clearest illustrations of the principles of data science.

It is important to understand data science even if you never intend to apply it yourself. Data-analytic thinking enables you to evaluate proposals for data mining projects. For example, if an employee, a consultant, or a potential investment target proposes to improve a particular business application by extracting knowledge from data, you should be able to assess the proposal systematically and decide whether it is sound or flawed. This does not mean that you will be able to tell whether it will actually succeed—for data mining projects, that often requires trying—but you should be able to spot obvious flaws, unrealistic assumptions, and missing pieces.

Throughout the book we will describe a number of fundamental data science principles, and will illustrate each with at least one data mining technique that embodies the principle. For each principle there are usually many specific techniques that embody it, so in this book we have chosen to emphasize the basic principles in preference to specific techniques. That said, we will not make a big deal about the difference between data

science and data mining, except where it will have a substantial effect on understanding the actual concepts.

Let's examine two brief case studies of analyzing data to extract predictive patterns.

Example: Hurricane Frances

Consider an example from a *New York Times* story from 2004:

> Hurricane Frances was on its way, barreling across the Caribbean, threatening a direct hit on Florida's Atlantic coast. Residents made for higher ground, but far away, in Bentonville, Ark., executives at Wal-Mart Stores decided that the situation offered a great opportunity for one of their newest data-driven weapons … predictive technology.
>
> A week ahead of the storm's landfall, Linda M. Dillman, Wal-Mart's chief information officer, pressed her staff to come up with forecasts based on what had happened when Hurricane Charley struck several weeks earlier. Backed by the trillions of bytes' worth of shopper history that is stored in Wal-Mart's data warehouse, she felt that the company could 'start predicting what's going to happen, instead of waiting for it to happen,' as she put it. (Hays, 2004)

Consider *why* data-driven prediction might be useful in this scenario. It might be useful to predict that people in the path of the hurricane would buy more bottled water. Maybe, but this point seems a bit obvious, and why would we need data science to discover it? It might be useful to project the *amount of increase* in sales due to the hurricane, to ensure that local Wal-Marts are properly stocked. Perhaps mining the data could reveal that a particular DVD sold out in the hurricane's path—but maybe it sold out that week at Wal-Marts across the country, not just where the hurricane landing was imminent. The prediction could be somewhat useful, but is probably more general than Ms. Dillman was intending.

It would be more valuable to discover patterns due to the hurricane that were not obvious. To do this, analysts might examine the huge volume of Wal-Mart data from prior, similar situations (such as Hurricane Charley) to identify *unusual* local demand for products. From such patterns, the company might be able to anticipate unusual demand for products and rush stock to the stores ahead of the hurricane's landfall.

Indeed, that is what happened. *The New York Times* (Hays, 2004) reported that: "… the experts mined the data and found that the stores would indeed need certain products —and not just the usual flashlights. 'We didn't know in the past that strawberry Pop-Tarts increase in sales, like seven times their normal sales rate, ahead of a hurricane,' Ms. Dillman said in a recent interview. 'And the pre-hurricane top-selling item was beer.'"[1]

1. Of course! What goes better with strawberry Pop-Tarts than a nice cold beer?

Example: Predicting Customer Churn

How are such data analyses performed? Consider a second, more typical business scenario and how it might be treated from a data perspective. This problem will serve as a running example that will illuminate many of the issues raised in this book and provide a common frame of reference.

Assume you just landed a great analytical job with MegaTelCo, one of the largest telecommunication firms in the United States. They are having a major problem with customer retention in their wireless business. In the mid-Atlantic region, 20% of cell phone customers leave when their contracts expire, and it is getting increasingly difficult to acquire new customers. Since the cell phone market is now saturated, the huge growth in the wireless market has tapered off. Communications companies are now engaged in battles to attract each other's customers while retaining their own. Customers switching from one company to another is called *churn*, and it is expensive all around: one company must spend on incentives to attract a customer while another company loses revenue when the customer departs.

You have been called in to help understand the problem and to devise a solution. Attracting new customers is much more expensive than retaining existing ones, so a good deal of marketing budget is allocated to prevent churn. Marketing has already designed a special retention offer. Your task is to devise a precise, step-by-step plan for how the data science team should use MegaTelCo's vast data resources to decide which customers should be offered the special retention deal prior to the expiration of their contracts.

Think carefully about what data you might use and how they would be used. Specifically, how should MegaTelCo choose a set of customers to receive their offer in order to best reduce churn for a particular incentive budget? Answering this question is much more complicated than it may seem initially. We will return to this problem repeatedly through the book, adding sophistication to our solution as we develop an understanding of the fundamental data science concepts.

In reality, customer retention has been a major use of data mining technologies—especially in telecommunications and finance businesses. These more generally were some of the earliest and widest adopters of data mining technologies, for reasons discussed later.

Data Science, Engineering, and Data-Driven Decision Making

Data science involves principles, processes, and techniques for understanding phenomena via the (automated) analysis of data. In this book, we will view the ultimate goal

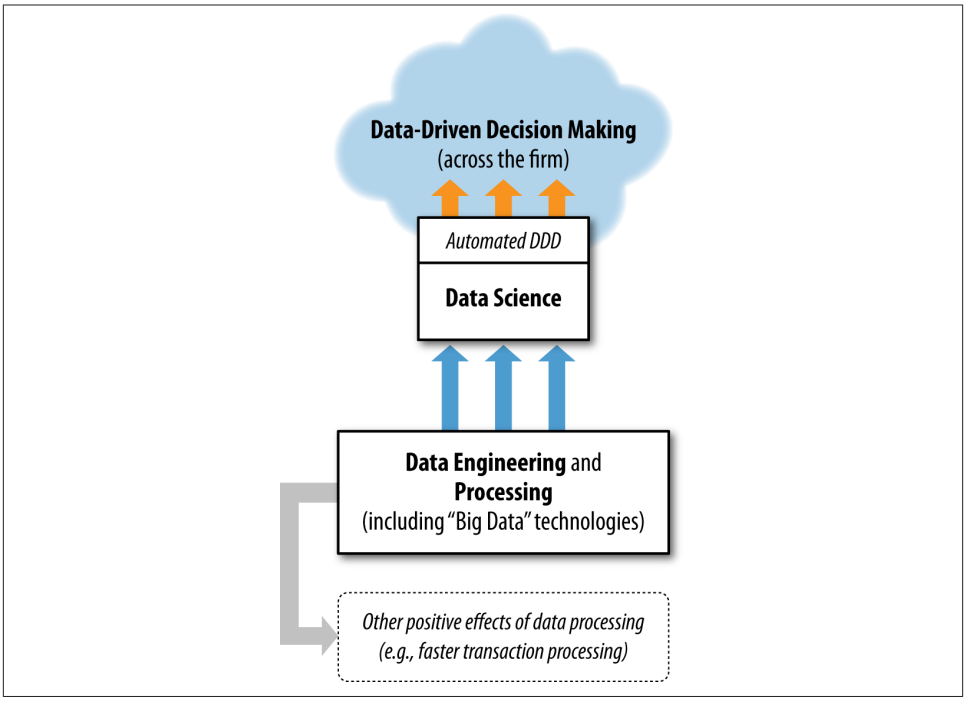


Figure 1-1. Data science in the context of various data-related processes in the organization.

of data science as improving decision making, as this generally is of direct interest to business.

Figure 1-1 places data science in the context of various other closely related and data-related processes in the organization. It distinguishes data science from other aspects of data processing that are gaining increasing attention in business. Let's start at the top.

Data-driven decision-making (DDD) refers to the practice of basing decisions on the analysis of data, rather than purely on intuition. For example, a marketer could select advertisements based purely on her long experience in the field and her eye for what will work. Or, she could base her selection on the analysis of data regarding how consumers react to different ads. She could also use a combination of these approaches. DDD is not an all-or-nothing practice, and different firms engage in DDD to greater or lesser degrees.

The benefits of data-driven decision-making have been demonstrated conclusively. Economist Erik Brynjolfsson and his colleagues from MIT and Penn's Wharton School conducted a study of how DDD affects firm performance (Brynjolfsson, Hitt, & Kim, 2011). They developed a measure of DDD that rates firms as to how strongly they use

data to make decisions across the company. They show that statistically, the more data-driven a firm is, the more productive it is—even controlling for a wide range of possible confounding factors. And the differences are not small. One standard deviation higher on the DDD scale is associated with a 4%–6% increase in productivity. DDD also is correlated with higher return on assets, return on equity, asset utilization, and market value, and the relationship seems to be causal.

The sort of decisions we will be interested in in this book mainly fall into two types: (1) decisions for which "discoveries" need to be made within data, and (2) decisions that repeat, especially at massive scale, and so decision-making can benefit from even small increases in decision-making accuracy based on data analysis. The Walmart example above illustrates a type 1 problem: Linda Dillman would like to discover knowledge that will help Walmart prepare for Hurricane Frances's imminent arrival.

In 2012, Walmart's competitor Target was in the news for a data-driven decision-making case of its own, also a type 1 problem (Duhigg, 2012). Like most retailers, Target cares about consumers' shopping habits, what drives them, and what can influence them. Consumers tend to have inertia in their habits and getting them to change is very difficult. Decision makers at Target knew, however, that the arrival of a new baby in a family is one point where people do change their shopping habits significantly. In the Target analyst's words, "As soon as we get them buying diapers from us, they're going to start buying everything else too." Most retailers know this and so they compete with each other trying to sell baby-related products to new parents. Since most birth records are public, retailers obtain information on births and send out special offers to the new parents.

However, Target wanted to get a jump on their competition. They were interested in whether they could *predict* that people *are expecting* a baby. If they could, they would gain an advantage by making offers before their competitors. Using techniques of data science, Target analyzed historical data on customers who *later* were revealed to have been pregnant, and were able to extract information that could predict which consumers were pregnant. For example, pregnant mothers often change their diets, their wardrobes, their vitamin regimens, and so on. These indicators could be extracted from historical data, assembled into predictive models, and then deployed in marketing campaigns. We will discuss predictive models in much detail as we go through the book. For the time being, it is sufficient to understand that a predictive model abstracts away most of the complexity of the world, focusing in on a particular set of indicators that correlate in some way with a quantity of interest (who will churn, or who will purchase, who is pregnant, etc.). Importantly, in both the Walmart and the Target examples, the

data analysis was not testing a simple hypothesis. Instead, the data were explored with the hope that something useful would be discovered.[2]

Our churn example illustrates a type 2 DDD problem. MegaTelCo has hundreds of millions of customers, each a candidate for defection. Tens of millions of customers have contracts expiring each month, so each one of them has an increased likelihood of defection in the near future. If we can improve our ability to estimate, for a given customer, how profitable it would be for us to focus on her, we can potentially reap large benefits by applying this ability to the millions of customers in the population. This same logic applies to many of the areas where we have seen the most intense application of data science and data mining: direct marketing, online advertising, credit scoring, financial trading, help-desk management, fraud detection, search ranking, product recommendation, and so on.

The diagram in Figure 1-1 shows data science supporting data-driven decision-making, but also overlapping with data-driven decision-making. This highlights the often overlooked fact that, increasingly, business decisions are being made *automatically* by computer systems. Different industries have adopted automatic decision-making at different rates. The finance and telecommunications industries were early adopters, largely because of their precocious development of data networks and implementation of massive-scale computing, which allowed the aggregation and modeling of data at a large scale, as well as the application of the resultant models to decision-making.

In the 1990s, automated decision-making changed the banking and consumer credit industries dramatically. In the 1990s, banks and telecommunications companies also implemented massive-scale systems for managing data-driven fraud control decisions. As retail systems were increasingly computerized, merchandising decisions were automated. Famous examples include Harrah's casinos' reward programs and the automated recommendations of Amazon and Netflix. Currently we are seeing a revolution in advertising, due in large part to a huge increase in the amount of time consumers are spending online, and the ability online to make (literally) split-second advertising decisions.

Data Processing and "Big Data"

It is important to digress here to address another point. There is a lot to data processing that is not data science—despite the impression one might get from the media. Data engineering and processing are critical to support data science, but they are more general. For example, these days many data processing skills, systems, and technologies often are mistakenly cast as data science. To understand data science and data-driven

2. Target was successful enough that this case raised ethical questions on the deployment of such techniques. Concerns of ethics and privacy are interesting and very important, but we leave their discussion for another time and place.

businesses it is important to understand the differences. Data science needs access to data and it often benefits from sophisticated data engineering that data processing technologies may facilitate, but these technologies are not data science technologies per se. They support data science, as shown in Figure 1-1, but they are useful for much more. Data processing technologies are very important for many data-oriented business tasks that do not involve extracting knowledge or data-driven decision-making, such as efficient transaction processing, modern web system processing, and online advertising campaign management.

"Big data" technologies (such as Hadoop, HBase, and MongoDB) have received considerable media attention recently. *Big data* essentially means datasets that are too large for traditional data processing systems, and therefore require new processing technologies. As with the traditional technologies, big data technologies are used for many tasks, including data engineering. Occasionally, big data technologies are actually used for *implementing* data mining techniques. However, much more often the well-known big data technologies are used for data processing *in support of* the data mining techniques and other data science activities, as represented in Figure 1-1.

Previously, we discussed Brynjolfsson's study demonstrating the benefits of data-driven decision-making. A separate study, conducted by economist Prasanna Tambe of NYU's Stern School, examined the extent to which *big data* technologies seem to help firms (Tambe, 2012). He finds that, after controlling for various possible confounding factors, using big data technologies is associated with significant additional productivity growth. Specifically, one standard deviation higher utilization of big data technologies is associated with 1%–3% higher productivity than the average firm; one standard deviation lower in terms of big data utilization is associated with 1%–3% lower productivity. This leads to potentially very large productivity differences between the firms at the extremes.

From Big Data 1.0 to Big Data 2.0

One way to think about the state of big data technologies is to draw an analogy with the business adoption of Internet technologies. In Web 1.0, businesses busied themselves with getting the basic internet technologies in place, so that they could establish a web presence, build electronic commerce capability, and improve the efficiency of their operations. We can think of ourselves as being in the era of Big Data 1.0. Firms are busying themselves with building the capabilities to process large data, largely in support of their current operations—for example, to improve efficiency.

Once firms had incorporated Web 1.0 technologies thoroughly (and in the process had driven down prices of the underlying technology) they started to look further. They began to ask what the Web could do for them, and how it could improve things they'd always done—and we entered the era of Web 2.0, where new systems and companies began taking advantage of the interactive nature of the Web. The changes brought on by this shift in thinking are pervasive; the most obvious are the incorporation of social-

networking components, and the rise of the "voice" of the individual consumer (and citizen).

We should expect a Big Data 2.0 phase to follow Big Data 1.0. Once firms have become capable of processing massive data in a flexible fashion, they should begin asking: "*What can I now do that I couldn't do before, or do better than I could do before?*" This is likely to be the golden era of data science. The principles and techniques we introduce in this book will be applied far more broadly and deeply than they are today.

It is important to note that in the Web 1.0 era some precocious companies began applying Web 2.0 ideas far ahead of the mainstream. Amazon is a prime example, incorporating the consumer's "voice" early on, in the rating of products, in product reviews (and deeper, in the rating of product reviews). Similarly, we see some companies already applying Big Data 2.0. Amazon again is a company at the forefront, providing data-driven recommendations from massive data. There are other examples as well. Online advertisers must process extremely large volumes of data (billions of ad impressions per day is not unusual) and maintain a very high throughput (real-time bidding systems make decisions in tens of milliseconds). We should look to these and similar industries for hints at advances in big data and data science that subsequently will be adopted by other industries.

Data and Data Science Capability as a Strategic Asset

The prior sections suggest one of the fundamental principles of data science: *data, and the capability to extract useful knowledge from data, should be regarded as key strategic assets.* Too many businesses regard data analytics as pertaining mainly to realizing value from some existing data, and often without careful regard to whether the business has the appropriate analytical talent. Viewing these as assets allows us to think explicitly about the extent to which one should invest in them. Often, we don't have exactly the right data to best make decisions and/or the right talent to best support making decisions from the data. Further, thinking of these as assets should lead us to the realization that they are *complementary*. The best data science team can yield little value without the appropriate data; the right data often cannot substantially improve decisions without suitable data science talent. As with all assets, it is often necessary to make investments. Building a top-notch data science team is a nontrivial undertaking, but can make a huge difference for decision-making. We will discuss strategic considerations involving data science in detail in Chapter 13. Our next case study will introduce the idea that thinking explicitly about how to invest in data assets very often pays off handsomely.

The classic story of little Signet Bank from the 1990s provides a case in point. Previously, in the 1980s, data science had transformed the business of consumer credit. Modeling the probability of default had changed the industry from personal assessment of the

likelihood of default to strategies of massive scale and market share, which brought along concomitant economies of scale. It may seem strange now, but at the time, credit cards essentially had uniform pricing, for two reasons: (1) the companies did not have adequate information systems to deal with differential pricing at massive scale, and (2) bank management believed customers would not stand for price discrimination. Around 1990, two strategic visionaries (Richard Fairbanks and Nigel Morris) realized that information technology was powerful enough that they could do more sophisticated predictive modeling—using the sort of techniques that we discuss throughout this book—and offer different terms (nowadays: pricing, credit limits, low-initial-rate balance transfers, cash back, loyalty points, and so on). These two men had no success persuading the big banks to take them on as consultants and let them try. Finally, after running out of big banks, they succeeded in garnering the interest of a small regional Virginia bank: Signet Bank. Signet Bank's management was convinced that modeling profitability, not just default probability, was the right strategy. They knew that a small proportion of customers actually account for *more than* 100% of a bank's profit from credit card operations (because the rest are break-even or money-losing). If they could model profitability, they could make better offers to the best customers and "skim the cream" of the big banks' clientele.

But Signet Bank had one really big problem in implementing this strategy. They did not have the appropriate data to model profitability with the goal of offering different terms to different customers. No one did. Since banks were offering credit with a specific set of terms and a specific default model, they had the data to model profitability (1) for the terms they actually have offered in the past, and (2) for the sort of customer who was actually offered credit (that is, those who were deemed worthy of credit by the existing model).

What could Signet Bank do? They brought into play a fundamental strategy of data science: acquire the necessary data at a cost. Once we view data as a business asset, we should think about whether and how much we are willing to invest. In Signet's case, data could be generated on the profitability of customers given different credit terms by conducting experiments. Different terms were offered at random to different customers. This may seem foolish outside the context of data-analytic thinking: you're likely to lose money! This is true. In this case, losses are the cost of data acquisition. The data-analytic thinker needs to consider whether she expects the data to have sufficient value to justify the investment.

So what happened with Signet Bank? As you might expect, when Signet began randomly offering terms to customers for data acquisition, the number of bad accounts soared. Signet went from an industry-leading "charge-off" rate (2.9% of balances went unpaid) to almost 6% charge-offs. Losses continued for a few years while the data scientists worked to build predictive models from the data, evaluate them, and deploy them to improve profit. Because the firm viewed these losses as investments in data, they persisted despite complaints from stakeholders. Eventually, Signet's credit card operation

turned around and became so profitable that it was spun off to separate it from the bank's other operations, which now were overshadowing the consumer credit success.

Fairbanks and Morris became Chairman and CEO and President and COO, and proceeded to apply data science principles throughout the business—not just customer acquisition but retention as well. When a customer calls looking for a better offer, data-driven models calculate the potential profitability of various possible actions (different offers, including sticking with the status quo), and the customer service representative's computer presents the best offers to make.

You may not have heard of little Signet Bank, but if you're reading this book you've probably heard of the spin-off: Capital One. Fairbanks and Morris's new company grew to be one of the largest credit card issuers in the industry with one of the lowest charge-off rates. In 2000, the bank was reported to be carrying out 45,000 of these "scientific tests" as they called them.[3]

Studies giving clear quantitative demonstrations of the value of a data asset are hard to find, primarily because firms are hesitant to divulge results of strategic value. One exception is a study by Martens and Provost (2011) assessing whether data on the specific transactions of a bank's consumers can improve models for deciding what product offers to make. The bank built models from data to decide whom to target with offers for different products. The investigation examined a number of different types of data and their effects on predictive performance. Sociodemographic data provide a substantial ability to model the sort of consumers that are more likely to purchase one product or another. However, sociodemographic data only go so far; after a certain volume of data, no additional advantage is conferred. In contrast, detailed data on customers' individual (anonymized) transactions improve performance substantially over just using sociodemographic data. The relationship is clear and striking and—significantly, for the point here—the predictive performance continues to improve as more data are used, increasing throughout the range investigated by Martens and Provost with no sign of abating. This has an important implication: banks with bigger data assets may have an important strategic advantage over their smaller competitors. If these trends generalize, and the banks are able to apply sophisticated analytics, banks with bigger data assets should be better able to identify the best customers for individual products. The net result will be either increased adoption of the bank's products, decreased cost of customer acquisition, or both.

The idea of data as a strategic asset is certainly not limited to Capital One, nor even to the banking industry. Amazon was able to gather data early on online customers, which has created significant switching costs: consumers find value in the rankings and recommendations that Amazon provides. Amazon therefore can retain customers more easily, and can even charge a premium (Brynjolfsson & Smith, 2000). Harrah's casinos

3. You can read more about Capital One's story (Clemons & Thatcher, 1998; McNamee 2001).

famously invested in gathering and mining data on gamblers, and moved itself from a small player in the casino business in the mid-1990s to the acquisition of Caesar's Entertainment in 2005 to become the world's largest gambling company. The huge valuation of Facebook has been credited to its vast and unique data assets (Sengupta, 2012), including both information about individuals and their likes, as well as information about the structure of the social network. Information about network structure has been shown to be important to predicting and has been shown to be remarkably helpful in building models of who will buy certain products (Hill, Provost, & Volinsky, 2006). It is clear that Facebook has a remarkable data asset; whether they have the right data science strategies to take full advantage of it is an open question.

In the book we will discuss in more detail many of the fundamental concepts behind these success stories, in exploring the principles of data mining and data-analytic thinking.

Data-Analytic Thinking

Analyzing case studies such as the churn problem improves our ability to approach problems "data-analytically." Promoting such a perspective is a primary goal of this book. When faced with a business problem, you should be able to assess whether and how data can improve performance. We will discuss a set of fundamental concepts and principles that facilitate careful thinking. We will develop frameworks to structure the analysis so that it can be done systematically.

As mentioned above, it is important to understand data science even if you never intend to do it yourself, because data analysis is now so critical to business strategy. Businesses increasingly are driven by data analytics, so there is great professional advantage in being able to interact competently with and within such businesses. Understanding the fundamental concepts, and having frameworks for organizing data-analytic thinking not only will allow one to interact competently, but will help to envision opportunities for improving data-driven decision-making, or to see data-oriented competitive threats.

Firms in many traditional industries are exploiting new and existing data resources for competitive advantage. They employ data science teams to bring advanced technologies to bear to increase revenue and to decrease costs. In addition, many new companies are being developed with data mining as a key strategic component. Facebook and Twitter, along with many other "Digital 100" companies (*Business Insider*, 2012), have high valuations due primarily to data assets they are committed to capturing or creating.[4] Increasingly, managers need to oversee analytics teams and analysis projects, marketers have to organize and understand data-driven campaigns, venture capitalists must be

4. Of course, this is not a new phenomenon. Amazon and Google are well-established companies that get tremendous value from their data assets.

able to invest wisely in businesses with substantial data assets, and business strategists must be able to devise plans that exploit data.

As a few examples, if a consultant presents a proposal to mine a data asset to improve your business, you should be able to assess whether the proposal makes sense. If a competitor announces a new data partnership, you should recognize when it may put you at a strategic disadvantage. Or, let's say you take a position with a venture firm and your first project is to assess the potential for investing in an advertising company. The founders present a convincing argument that they will realize significant value from a unique body of data they will collect, and on that basis are arguing for a substantially higher valuation. Is this reasonable? With an understanding of the fundamentals of data science you should be able to devise a few probing questions to determine whether their valuation arguments are plausible.

On a scale less grand, but probably more common, data analytics projects reach into all business units. Employees throughout these units must interact with the data science team. If these employees do not have a fundamental grounding in the principles of data-analytic thinking, they will not really understand what is happening in the business. This lack of understanding is much more damaging in data science projects than in other technical projects, because the data science is supporting improved decision-making. As we will describe in the next chapter, this requires a close interaction between the data scientists and the business people responsible for the decision-making. Firms where the business people do not understand what the data scientists are doing are at a substantial disadvantage, because they waste time and effort or, worse, because they ultimately make wrong decisions.

The need for managers with data-analytic skills

The consulting firm McKinsey and Company estimates that "there will be a shortage of talent necessary for organizations to take advantage of big data. By 2018, the United States alone could face a shortage of 140,000 to 190,000 people with deep analytical skills as well as 1.5 million managers and analysts with the know-how to use the analysis of big data to make effective decisions." (Manyika, 2011). Why 10 times as many managers and analysts than those with deep analytical skills? Surely data scientists aren't so difficult to manage that they need 10 managers! The reason is that a business can get leverage from a data science team for making better decisions in multiple areas of the business. However, as McKinsey is pointing out, the managers in those areas need to understand the fundamentals of data science to effectively get that leverage.

This Book

This book concentrates on the fundamentals of data science and data mining. These are a set of principles, concepts, and techniques that structure thinking and analysis. They allow us to understand data science processes and methods surprisingly deeply, without needing to focus in depth on the large number of specific data mining algorithms.

There are many good books covering data mining algorithms and techniques, from practical guides to mathematical and statistical treatments. This book instead focuses on the fundamental concepts and how they help us to think about problems where data mining may be brought to bear. That doesn't mean that we will ignore the data mining techniques; many algorithms are exactly the embodiment of the basic concepts. But with only a few exceptions we will not concentrate on the deep technical details of how the techniques actually work; we will try to provide just enough detail so that you will understand what the techniques do, and how they are based on the fundamental principles.

Data Mining and Data Science, Revisited

This book devotes a good deal of attention to the extraction of useful (nontrivial, hopefully actionable) patterns or models from large bodies of data (Fayyad, Piatetsky-Shapiro, & Smyth, 1996), and to the fundamental data science principles underlying such data mining. In our churn-prediction example, we would like to *take the data* on prior churn and *extract patterns*, for example patterns of behavior, *that are useful*—that can help us to predict those customers who are more likely to leave in the future, or that can help us to design better services.

The fundamental concepts of data science are drawn from many fields that study data analytics. We introduce these concepts throughout the book, but let's briefly discuss a few now to get the basic flavor. We will elaborate on all of these and more in later chapters.

Fundamental concept: *Extracting useful knowledge from data to solve business problems can be treated systematically by following a process with reasonably well-defined stages.* The Cross Industry Standard Process for Data Mining, abbreviated CRISP-DM (CRISP-DM Project, 2000), is one codification of this process. Keeping such a process in mind provides a framework to structure our thinking about data analytics problems. For example, in actual practice one repeatedly sees analytical "solutions" that are not based on careful analysis of the problem or are not carefully evaluated. Structured thinking about analytics emphasizes these often under-appreciated aspects of supporting decision-making with data. Such structured thinking also contrasts critical points where human creativity is necessary versus points where high-powered analytical tools can be brought to bear.

Fundamental concept: *From a large mass of data, information technology can be used to find informative descriptive attributes of entities of interest.* In our churn example, a customer would be an entity of interest, and each customer might be described by a large number of attributes, such as usage, customer service history, and many other factors. Which of these actually gives us information on the customer's likelihood of leaving the company when her contract expires? How much information? Sometimes this process is referred to roughly as finding variables that "correlate" with churn (we will discuss this notion precisely). A business analyst may be able to hypothesize some and test them, and there are tools to help facilitate this experimentation (see "Other Analytics Techniques and Technologies" on page 35). Alternatively, the analyst could apply information technology to automatically discover informative attributes—essentially doing large-scale automated experimentation. Further, as we will see, this concept can be applied recursively to build models to predict churn based on multiple attributes.

Fundamental concept: *If you look too hard at a set of data, you will find something—but it might not generalize beyond the data you're looking at.* This is referred to as *overfitting* a dataset. Data mining techniques can be very powerful, and the need to detect and avoid overfitting is one of the most important concepts to grasp when applying data mining to real problems. The concept of overfitting and its avoidance permeates data science processes, algorithms, and evaluation methods.

Fundamental concept: *Formulating data mining solutions and evaluating the results involves thinking carefully about the context in which they will be used.* If our goal is the extraction of potentially *useful* knowledge, how can we formulate what is useful? It depends critically on the application in question. For our churn-management example, how exactly are we going to use the patterns extracted from historical data? Should the value of the customer be taken into account in addition to the likelihood of leaving? More generally, does the pattern lead to better decisions than some reasonable alternative? How well would one have done by chance? How well would one do with a smart "default" alternative?

These are just four of the fundamental concepts of data science that we will explore. By the end of the book, we will have discussed a dozen such fundamental concepts in detail, and will have illustrated how they help us to structure data-analytic thinking and to understand data mining techniques and algorithms, as well as data science applications, quite generally.

Chemistry Is Not About Test Tubes: Data Science Versus the Work of the Data Scientist

Before proceeding, we should briefly revisit the engineering side of data science. At the time of this writing, discussions of data science commonly mention not just analytical skills and techniques for understanding data but popular tools used. Definitions of data

scientists (and advertisements for positions) specify not just areas of expertise but also specific programming languages and tools. It is common to see job advertisements mentioning data mining techniques (e.g., random forests, support vector machines), specific application areas (recommendation systems, ad placement optimization), alongside popular software tools for processing big data (Hadoop, MongoDB). There is often little distinction between the science and the technology for dealing with large datasets.

We must point out that data science, like computer science, is a young field. The particular concerns of data science are fairly new and general principles are just beginning to emerge. The state of data science may be likened to that of chemistry in the mid-19th century, when theories and general principles were being formulated and the field was largely experimental. Every good chemist had to be a competent lab technician. Similarly, it is hard to imagine a working data scientist who is not proficient with certain sorts of software tools.

Having said this, this book focuses on the science and not on the technology. You will not find instructions here on how best to run massive data mining jobs on Hadoop clusters, or even what Hadoop is or why you might want to learn about it.[5] We focus here on the general principles of data science that have emerged. In 10 years' time the predominant technologies will likely have changed or advanced enough that a discussion here would be obsolete, while the general principles are the same as they were 20 years ago, and likely will change little over the coming decades.

Summary

This book is about the extraction of useful information and knowledge from large volumes of data, in order to improve business decision-making. As the massive collection of data has spread through just about every industry sector and business unit, so have the opportunities for mining the data. Underlying the extensive body of techniques for mining data is a much smaller set of fundamental concepts comprising *data science*. These concepts are general and encapsulate much of the essence of data mining and business analytics.

Success in today's data-oriented business environment requires being able to think about how these fundamental concepts apply to particular business problems—to think data-analytically. For example, in this chapter we discussed the principle that data should be thought of as a business asset, and once we are thinking in this direction we start to ask whether (and how much) we should invest in data. Thus, an understanding of these fundamental concepts is important not only for data scientists themselves, but for any-

5. OK: Hadoop is a widely used open source architecture for doing highly parallelizable computations. It is one of the current "big data" technologies for processing massive datasets that exceed the capacity of relational database systems. Hadoop is based on the MapReduce parallel processing framework introduced by Google.

one working with data scientists, employing data scientists, investing in data-heavy ventures, or directing the application of analytics in an organization.

Thinking data-analytically is aided by conceptual frameworks discussed throughout the book. For example, the automated extraction of patterns from data is a process with well-defined stages, which are the subject of the next chapter. Understanding the process and the stages helps to structure our data-analytic thinking, and to make it more systematic and therefore less prone to errors and omissions.

There is convincing evidence that data-driven decision-making and big data technologies substantially improve business performance. Data science supports data-driven decision-making—and sometimes conducts such decision-making automatically—and depends upon technologies for "big data" storage and engineering, but its principles are separate. The data science principles we discuss in this book also differ from, and are complementary to, other important technologies, such as statistical hypothesis testing and database querying (which have their own books and classes). The next chapter describes some of these differences in more detail.

CHAPTER 2

Business Problems and Data Science Solutions

Fundamental concepts: *A set of canonical data mining tasks; The data mining process; Supervised versus unsupervised data mining.*

An important principle of data science is that data mining is a *process* with fairly well-understood stages. Some involve the application of information technology, such as the automated discovery and evaluation of patterns from data, while others mostly require an analyst's creativity, business knowledge, and common sense. Understanding the whole process helps to structure data mining projects, so they are closer to systematic analyses rather than heroic endeavors driven by chance and individual acumen.

Since the data mining process breaks up the overall task of finding patterns from data into a set of well-defined subtasks, it is also useful for structuring discussions about data science. In this book, we will use the process as an overarching framework for our discussion. This chapter introduces the data mining process, but first we provide additional context by discussing common types of data mining tasks. Introducing these allows us to be more concrete when presenting the overall process, as well as when introducing other concepts in subsequent chapters.

We close the chapter by discussing a set of important business analytics subjects that are not the focus of this book (but for which there are many other helpful books), such as databases, data warehousing, and basic statistics.

From Business Problems to Data Mining Tasks

Each data-driven business decision-making problem is unique, comprising its own combination of goals, desires, constraints, and even personalities. As with much engineering, though, there are sets of common tasks that underlie the business problems. In collaboration with business stakeholders, data scientists decompose a business prob-

lem into subtasks. The solutions to the subtasks can then be composed to solve the overall problem. Some of these subtasks are unique to the particular business problem, but others are common data mining tasks. For example, our telecommunications churn problem is unique to MegaTelCo: there are specifics of the problem that are different from churn problems of any other telecommunications firm. However, a subtask that will likely be part of the solution to any churn problem is to estimate from historical data the probability of a customer terminating her contract shortly after it has expired. Once the idiosyncratic MegaTelCo data have been assembled into a particular format (described in the next chapter), this probability estimation fits the mold of one very common data mining task. We know a lot about solving the common data mining tasks, both scientifically and practically. In later chapters, we also will provide data science frameworks to help with the decomposition of business problems and with the recomposition of the solutions to the subtasks.

A critical skill in data science is the ability to decompose a data-analytics problem into pieces such that each piece matches a known task for which tools are available. Recognizing familiar problems and their solutions avoids wasting time and resources reinventing the wheel. It also allows people to focus attention on more interesting parts of the process that require human involvement—parts that have not been automated, so human creativity and intelligence must come into play.

Despite the large number of specific data mining algorithms developed over the years, there are only a handful of fundamentally different types of tasks these algorithms address. It is worth defining these tasks clearly. The next several chapters will use the first two (classification and regression) to illustrate several fundamental concepts. In what follows, the term "an individual" will refer to an entity about which we have data, such as a customer or a consumer, or it could be an inanimate entity such as a business. We will make this notion more precise in Chapter 3. In many business analytics projects, we want to find "correlations" between a particular variable describing an individual and other variables. For example, in historical data we may know which customers left the company after their contracts expired. We may want to find out which other variables correlate with a customer leaving in the near future. Finding such correlations are the most basic examples of classification and regression tasks.

1. *Classification* and class *probability estimation* attempt to predict, for each individual in a population, which of a (small) set of classes this individual belongs to. Usually the classes are mutually exclusive. An example classification question would be: "Among all the customers of MegaTelCo, which are likely to respond to a given offer?" In this example the two classes could be called `will respond` and `will not respond`.

For a classification task, a data mining procedure produces a model that, given a new individual, determines which class that individual belongs to. A closely related task is *scoring* or class *probability estimation*. A scoring model applied to an individual produces, instead of a class prediction, a score representing the probability (or some other quantification of likelihood) that that individual belongs to each class. In our customer response scenario, a scoring model would be able to evaluate each individual customer and produce a score of how likely each is to respond to the offer. Classification and scoring are very closely related; as we shall see, a model that can do one can usually be modified to do the other.

2. *Regression* ("value estimation") attempts to estimate or predict, for each individual, the numerical value of some variable for that individual. An example regression question would be: "How much will a given customer use the service?" The property (variable) to be predicted here is *service usage*, and a model could be generated by looking at other, similar individuals in the population and their historical usage. A regression procedure produces a model that, given an individual, estimates the value of the particular variable specific to that individual.

   Regression is related to classification, but the two are different. Informally, classification predicts *whether* something will happen, whereas regression predicts *how much* something will happen. The difference will become clearer as the book progresses.

3. *Similarity matching* attempts to *identify* similar individuals based on data known about them. Similarity matching can be used directly to find similar entities. For example, IBM is interested in finding companies similar to their best business customers, in order to focus their sales force on the best opportunities. They use similarity matching based on "firmographic" data describing characteristics of the companies. Similarity matching is the basis for one of the most popular methods for making product recommendations (finding people who are similar to you in terms of the products they have liked or have purchased). Similarity measures underlie certain solutions to other data mining tasks, such as classification, regression, and clustering. We discuss similarity and its uses at length in Chapter 6.

4. *Clustering* attempts to *group* individuals in a population together by their similarity, but not driven by any specific purpose. An example clustering question would be: "Do our customers form natural groups or segments?" Clustering is useful in preliminary domain exploration to see which natural groups exist because these groups in turn may suggest other data mining tasks or approaches. Clustering also is used as input to decision-making processes focusing on questions such as: *What products should we offer or develop*? *How should our customer care teams (or sales teams) be structured*? We discuss clustering in depth in Chapter 6.

5. *Co-occurrence grouping* (also known as frequent itemset mining, association rule discovery, and market-basket analysis) attempts to find *associations* between entities based on transactions involving them. An example co-occurrence question

would be: *What items are commonly purchased together*? While clustering looks at similarity between objects based on the objects' attributes, co-occurrence grouping considers similarity of objects based on their appearing together in transactions. For example, analyzing purchase records from a supermarket may uncover that ground meat is purchased together with hot sauce much more frequently than we might expect. Deciding how to act upon this discovery might require some creativity, but it could suggest a special promotion, product display, or combination offer. Co-occurrence of products in purchases is a common type of grouping known as market-basket analysis. Some *recommendation* systems also perform a type of affinity grouping by finding, for example, pairs of books that are purchased frequently by the same people ("people who bought X also bought Y").

The result of co-occurrence grouping is a description of items that occur together. These descriptions usually include statistics on the frequency of the co-occurrence and an estimate of how surprising it is.

6. *Profiling* (also known as behavior description) attempts to characterize the typical behavior of an individual, group, or population. An example profiling question would be: "What is the typical cell phone usage of this customer segment?" Behavior may not have a simple description; profiling cell phone usage might require a complex description of night and weekend airtime averages, international usage, roaming charges, text minutes, and so on. Behavior can be described generally over an entire population, or down to the level of small groups or even individuals.

   Profiling is often used to establish behavioral norms for anomaly detection applications such as fraud detection and monitoring for intrusions to computer systems (such as someone breaking into your iTunes account). For example, if we know what kind of purchases a person typically makes on a credit card, we can determine whether a new charge on the card fits that profile or not. We can use the degree of mismatch as a suspicion score and issue an alarm if it is too high.

7. *Link prediction* attempts to predict connections between data items, usually by suggesting that a link should exist, and possibly also estimating the strength of the link. Link prediction is common in social networking systems: "Since you and Karen share 10 friends, maybe you'd like to be Karen's friend?" Link prediction can also estimate the strength of a link. For example, for recommending movies to customers one can think of a graph between customers and the movies they've watched or rated. Within the graph, we search for links that do *not* exist between customers and movies, but that we predict should exist and should be strong. These links form the basis for recommendations.

8. *Data reduction* attempts to take a large set of data and replace it with a smaller set of data that contains much of the important information in the larger set. The smaller dataset may be easier to deal with or to process. Moreover, the smaller dataset may better reveal the information. For example, a massive dataset on consumer movie-viewing preferences may be reduced to a much smaller dataset re-

vealing the consumer taste preferences that are latent in the viewing data (for example, viewer genre preferences). Data reduction usually involves loss of information. What is important is the trade-off for improved insight.

9. *Causal modeling* attempts to help us understand what events or actions actually *influence* others. For example, consider that we use predictive modeling to target advertisements to consumers, and we observe that indeed the targeted consumers purchase at a higher rate subsequent to having been targeted. Was this because the advertisements influenced the consumers to purchase? Or did the predictive models simply do a good job of identifying those consumers who would have purchased anyway? Techniques for causal modeling include those involving a substantial investment in data, such as randomized controlled experiments (e.g., so-called "A/B tests"), as well as sophisticated methods for drawing causal conclusions from observational data. Both experimental and observational methods for causal modeling generally can be viewed as "counterfactual" analysis: they attempt to understand what would be the difference between the situations—which cannot both happen—where the "treatment" event (e.g., showing an advertisement to a particular individual) were to happen, and were not to happen.

   In all cases, a careful data scientist should always include with a causal conclusion the exact assumptions that must be made in order for the causal conclusion to hold (there *always* are such assumptions—always ask). When undertaking causal modeling, a business needs to weigh the trade-off of increasing investment to reduce the assumptions made, versus deciding that the conclusions are good enough given the assumptions. Even in the most careful randomized, controlled experimentation, assumptions are made that could render the causal conclusions invalid. The discovery of the "placebo effect" in medicine illustrates a notorious situation where an assumption was overlooked in carefully designed randomized experimentation.

Discussing all of these tasks in detail would fill multiple books. In this book, we present a collection of the most fundamental data science principles—principles that together underlie all of these types of tasks. We will illustrate the principles mainly using classification, regression, similarity matching, and clustering, and will discuss others when they provide important illustrations of the fundamental principles (toward the end of the book).

Consider which of these types of tasks might fit our churn-prediction problem. Often, practitioners formulate churn prediction as a problem of finding *segments* of customers who are more or less likely to leave. This segmentation problem sounds like a classification problem, or possibly clustering, or even regression. To decide the best formulation, we first need to introduce some important distinctions.

Supervised Versus Unsupervised Methods

Consider two similar questions we might ask about a customer population. The first is: "Do our customers naturally fall into different groups?" Here no specific purpose or *target* has been specified for the grouping. When there is no such target, the data mining problem is referred to as *unsupervised*. Contrast this with a slightly different question: "Can we find groups of customers who have particularly high likelihoods of canceling their service soon after their contracts expire?" Here there is a specific target defined: will a customer leave when her contract expires? In this case, segmentation is being done for a specific reason: to take action based on likelihood of churn. This is called a *supervised* data mining problem.

A note on the terms: Supervised and unsupervised learning

The terms *supervised* and *unsupervised* were inherited from the field of machine learning. Metaphorically, a teacher "supervises" the learner by carefully providing target information along with a set of examples. An unsupervised learning task might involve the same set of examples but would not include the target information. The learner would be given no information about the purpose of the learning, but would be left to form its own conclusions about what the examples have in common.

The difference between these questions is subtle but important. If a specific target can be provided, the problem can be phrased as a supervised one. Supervised tasks require different techniques than unsupervised tasks do, and the results often are much more useful. A supervised technique is given a specific purpose for the grouping—predicting the target. Clustering, an unsupervised task, produces groupings based on similarities, but there is no guarantee that these similarities are meaningful or will be useful for any particular purpose.

Technically, another condition must be met for supervised data mining: there must be *data* on the target. It is not enough that the target information exist in principle; it must also exist in the data. For example, it might be useful to know whether a given customer will stay for at least six months, but if in historical data this retention information is missing or incomplete (if, say, the data are only retained for two months) the target values cannot be provided. Acquiring data on the target often is a key data science investment. The value for the target variable for an individual is often called the individual's *label*, emphasizing that often (not always) one must incur expense to actively label the data.

Classification, regression, and causal modeling generally are solved with supervised methods. Similarity matching, link prediction, and data reduction could be either. Clustering, co-occurrence grouping, and profiling generally are unsupervised. The

fundamental principles of data mining that we will present underlie all these types of technique.

Two main subclasses of *supervised* data mining, classification and regression, are distinguished by the type of target. Regression involves a numeric target while classification involves a categorical (often binary) target. Consider these similar questions we might address with supervised data mining:

"Will this customer purchase service S1 if given incentive I?"
: This is a classification problem because it has a binary target (the customer either purchases or does not).

"Which service package (S1, S2, or none) will a customer likely purchase if given incentive I?"
: This is also a classification problem, with a three-valued target.

"How much will this customer use the service?"
: This is a regression problem because it has a numeric target. The target variable is the amount of usage (actual or predicted) per customer.

There are subtleties among these questions that should be brought out. For business applications we often want a numerical *prediction* over a categorical target. In the churn example, a basic yes/no prediction of whether a customer is likely to continue to subscribe to the service may not be sufficient; we want to model the *probability* that the customer will continue. This is still considered classification modeling rather than regression because the underlying target is categorical. Where necessary for clarity, this is called "class probability estimation."

A vital part in the early stages of the data mining process is (i) to decide whether the line of attack will be supervised or unsupervised, and (ii) if supervised, to produce a precise definition of a target variable. This variable must be a specific quantity that will be the focus of the data mining (and for which we can obtain values for some example data). We will return to this in Chapter 3.

Data Mining and Its Results

There is another important distinction pertaining to mining data: the difference between (1) mining the data to find patterns and build models, and (2) *using* the results of data mining. Students often confuse these two processes when studying data science, and managers sometimes confuse them when discussing business analytics. The use of data mining results should influence and inform the data mining process itself, but the two should be kept distinct.

In our churn example, consider the deployment scenario in which the results will be used. We want to use the model to predict which of our customers will leave. Specifically, assume that data mining has created a class probability estimation model *M*. Given each

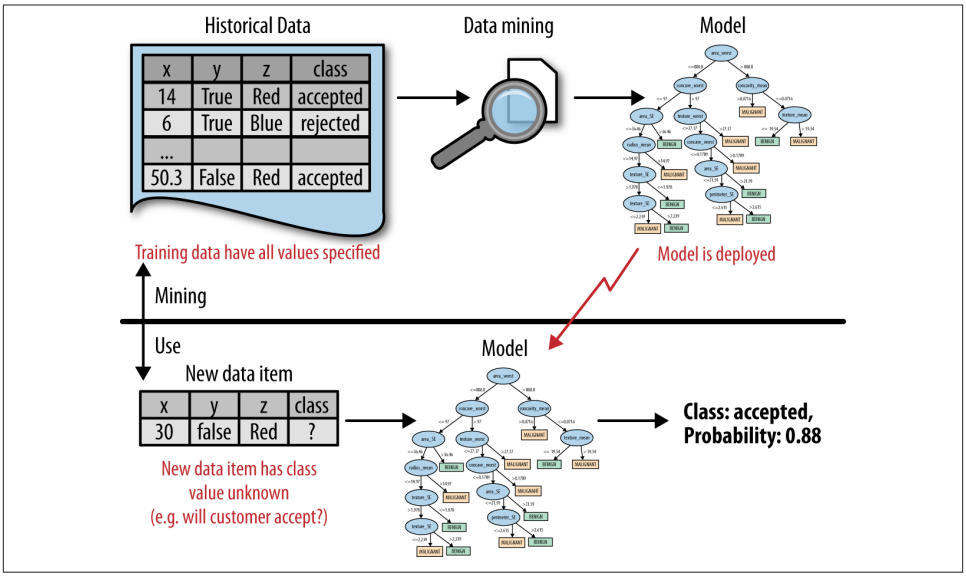


Figure 2-1. Data mining versus the use of data mining results. The upper half of the figure illustrates the mining of historical data to produce a model. Importantly, the historical data have the target ("class") value specified. The bottom half shows the result of the data mining in use, where the model is applied to new data for which we do not know the class value. The model predicts both the class value and the probability that the class variable will take on that value.

existing customer, described using a set of characteristics, *M* takes these characteristics as input and produces a score or probability estimate of attrition. This is the *use* of the results of data mining. The data mining produces the model *M* from some other, often historical, data.

Figure 2-1 illustrates these two phases. Data mining produces the probability estimation model, as shown in the top half of the figure. In the use phase (bottom half), the model is applied to a new, unseen case and it generates a probability estimate for it.

The Data Mining Process

Data mining is a craft. It involves the application of a substantial amount of science and technology, but the proper application still involves art as well. But as with many mature crafts, there is a well-understood process that places a structure on the problem, allowing reasonable consistency, repeatability, and objectiveness. A useful codification of the data

mining process is given by the Cross Industry Standard Process for Data Mining (CRISP-DM; Shearer, 2000), illustrated in Figure 2-2.[1]

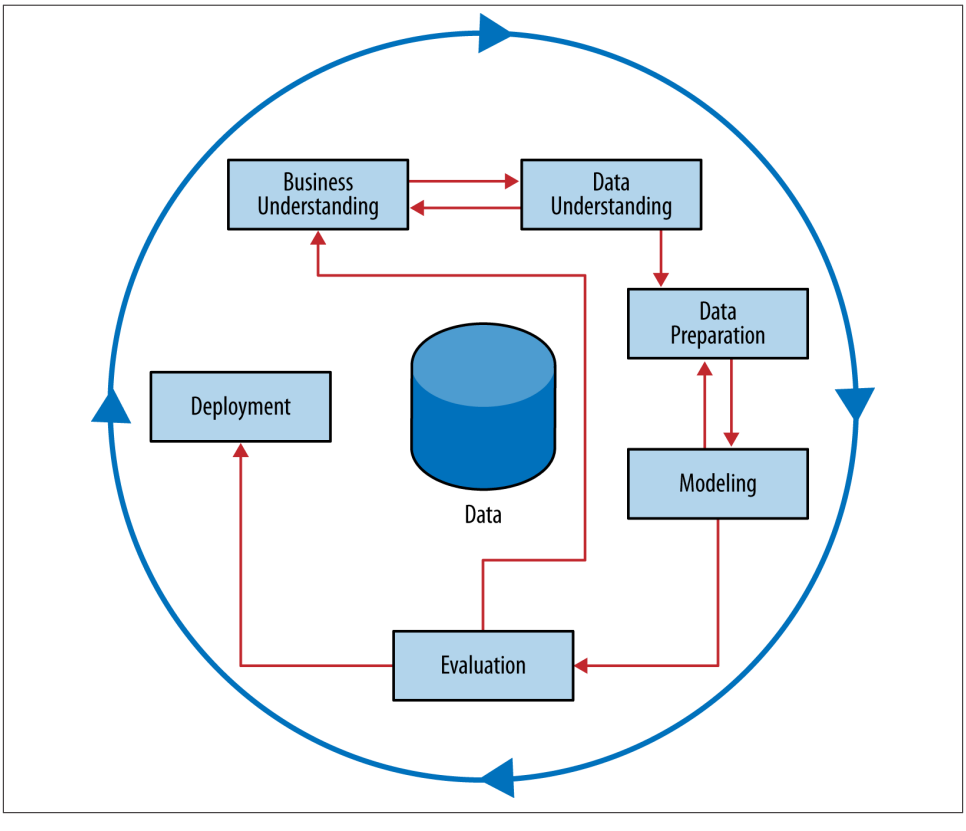


Figure 2-2. The CRISP data mining process.

This process diagram makes explicit the fact that iteration is the rule rather than the exception. Going through the process once without having solved the problem is, generally speaking, not a failure. Often the entire process is an exploration of the data, and after the first iteration the data science team knows much more. The next iteration can be much more well-informed. Let's now discuss the steps in detail.

Business Understanding

Initially, it is vital to understand the problem to be solved. This may seem obvious, but business projects seldom come pre-packaged as clear and unambiguous data mining

1. See also the Wikipedia page on the CRISP-DM process model.

problems. Often recasting the problem and designing a solution is an iterative process of discovery. The diagram shown in Figure 2-2 represents this as cycles within a cycle, rather than as a simple linear process. The initial formulation may not be complete or optimal so multiple iterations may be necessary for an acceptable solution formulation to appear.

The Business Understanding stage represents a part of the craft where the analysts' creativity plays a large role. Data science has some things to say, as we will describe, but often the key to a great success is a creative problem formulation by some analyst regarding how to cast the business problem as one or more data science problems. High-level knowledge of the fundamentals helps creative business analysts see novel formulations.

We have a set of powerful tools to solve particular data mining problems: the basic data mining tasks discussed in "From Business Problems to Data Mining Tasks" on page 19. Typically, the early stages of the endeavor involve designing a solution that takes advantage of these tools. This can mean structuring (engineering) the problem such that one or more subproblems involve building models for classification, regression, probability estimation, and so on.

In this first stage, *the design team should think carefully about the problem to be solved and about the use scenario*. This itself is one of the most important fundamental principles of data science, to which we have devoted two entire chapters (Chapter 7 and Chapter 11). What exactly do we want to do? How exactly would we do it? What parts of this use scenario constitute possible data mining models? In discussing this in more detail, we will begin with a simplified view of the use scenario, but as we go forward we will loop back and realize that often the use scenario must be adjusted to better reflect the actual business need. We will present conceptual tools to help our thinking here, for example framing a business problem in terms of expected value can allow us to systematically decompose it into data mining tasks.

Data Understanding

If solving the business problem is the goal, the data comprise the available raw material from which the solution will be built. It is important to understand the strengths and limitations of the data because rarely is there an exact match with the problem. Historical data often are collected for purposes unrelated to the current business problem, or for no explicit purpose at all. A customer database, a transaction database, and a marketing response database contain different information, may cover different intersecting populations, and may have varying degrees of reliability.

It is also common for the *costs* of data to vary. Some data will be available virtually for free while others will require effort to obtain. Some data may be purchased. Still other data simply won't exist and will require entire ancillary projects to arrange their collection. A critical part of the data understanding phase is estimating the costs and benefits

of each data source and deciding whether further investment is merited. Even after all datasets are acquired, collating them may require additional effort. For example, customer records and product identifiers are notoriously variable and noisy. Cleaning and matching customer records to ensure only one record per customer is itself a complicated analytics problem (Hernández & Stolfo, 1995; Elmagarmid, Ipeirotis, & Verykios, 2007).

As data understanding progresses, solution paths may change direction in response, and team efforts may even fork. Fraud detection provides an illustration of this. Data mining has been used extensively for fraud detection, and many fraud detection problems involve classic supervised data mining tasks. Consider the task of catching credit card fraud. Charges show up on each customer's account, so fraudulent charges are usually caught—if not initially by the company, then later by the customer when account activity is reviewed. We can assume that nearly all fraud is identified and reliably labeled, since the legitimate customer and the person perpetrating the fraud are different people and have opposite goals. Thus credit card transactions have reliable labels (*fraud* and *legitimate*) that may serve as targets for a supervised technique.

Now consider the related problem of catching Medicare fraud. This is a huge problem in the United States costing billions of dollars annually. Though this may seem like a conventional fraud detection problem, as we consider the relationship of the business problem to the data, we realize that the problem is significantly different. The perpetrators of fraud—medical providers who submit false claims, and sometimes their patients—are also legitimate service providers and users of the billing system. Those who commit fraud are a subset of the legitimate users; there is no separate disinterested party who will declare exactly what the "correct" charges should be. Consequently the Medicare billing data have no reliable target variable indicating fraud, and a supervised learning approach that could work for credit card fraud is not applicable. Such a problem usually requires unsupervised approaches such as profiling, clustering, anomaly detection, and co-occurrence grouping.

The fact that both of these are fraud detection problems is a superficial similarity that is actually misleading. In data understanding we need to dig beneath the surface to uncover the structure of the business problem and the data that are available, and then match them to one or more data mining tasks for which we may have substantial science and technology to apply. It is not unusual for a business problem to contain several data mining tasks, often of different types, and combining their solutions will be necessary (see Chapter 11).

Data Preparation

The analytic technologies that we can bring to bear are powerful but they impose certain requirements on the data they use. They often require data to be in a form different from how the data are provided naturally, and some conversion will be necessary.

Therefore a data preparation phase often proceeds along with data understanding, in which the data are manipulated and converted into forms that yield better results.

Typical examples of data preparation are converting data to tabular format, removing or inferring missing values, and converting data to different types. Some data mining techniques are designed for symbolic and categorical data, while others handle only numeric values. In addition, numerical values must often be normalized or scaled so that they are comparable. Standard techniques and rules of thumb are available for doing such conversions. Chapter 3 discusses the most typical format for mining data in some detail.

In general, though, this book will not focus on data preparation techniques, which could be the topic of a book by themselves (Pyle, 1999). We will define basic data formats in following chapters, and will only be concerned with data preparation details when they shed light on some fundamental principle of data science or are necessary to present a concrete example.

More generally, data scientists may spend considerable time early in the process defining the variables used later in the process. This is one of the main points at which human creativity, common sense, and business knowledge come into play. Often the quality of the data mining solution rests on how well the analysts structure the problems and craft the variables (and sometimes it can be surprisingly hard for them to admit it).

One very general and important concern during data preparation is to beware of "leaks" (Kaufman et al. 2012). A leak is a situation where a variable collected in historical data gives information on the target variable—information that appears in historical data but is not actually available when the decision has to be made. As an example, when predicting whether at a particular point in time a website visitor would end her session or continue surfing to another page, the variable "total number of webpages visited in the session" is predictive. However, the total number of webpages visited in the session would not be known until after the session was over (Kohavi et al., 2000)—at which point one would know the value for the target variable! As another illustrative example, consider predicting whether a customer *will be* a "big spender"; knowing the categories of the items purchased (or worse, the amount of tax paid) are very predictive, but are not known at decision-making time (Kohavi & Parekh, 2003). Leakage must be considered carefully during data preparation, because data preparation typically is performed after the fact—from historical data. We present a more detailed example of a real leak that was challenging to find in Chapter 14.

Modeling

Modeling is the subject of the next several chapters and we will not dwell on it here, except to say that the output of modeling is some sort of model or pattern capturing regularities in the data.

The modeling stage is the primary place where data mining techniques are applied to the data. It is important to have some understanding of the fundamental ideas of data mining, including the sorts of techniques and algorithms that exist, because this is the part of the craft where the most science and technology can be brought to bear.

Evaluation

The purpose of the evaluation stage is to assess the data mining results rigorously and to gain confidence that they are valid and reliable before moving on. If we look hard enough at any dataset we will find patterns, but they may not survive careful scrutiny. We would like to have confidence that the models and patterns extracted from the data are true regularities and not just idiosyncrasies or sample anomalies. It is possible to deploy results immediately after data mining but this is inadvisable; it is usually far easier, cheaper, quicker, and safer to test a model first in a controlled laboratory setting.

Equally important, the evaluation stage also serves to help ensure that the model satisfies the original business goals. Recall that the primary goal of data science for business is to support decision making, and that we started the process by focusing on the business problem we would like to solve. Usually a data mining solution is only a piece of the larger solution, and it needs to be evaluated as such. Further, even if a model passes strict evaluation tests in "in the lab," there may be external considerations that make it impractical. For example, a common flaw with detection solutions (such as fraud detection, spam detection, and intrusion monitoring) is that they produce too many false alarms. A model may be extremely accurate (> 99%) by laboratory standards, but evaluation in the actual business context may reveal that it still produces too many false alarms to be economically feasible. (How much would it cost to provide the staff to deal with all those false alarms? What would be the cost in customer dissatisfaction?)

Evaluating the results of data mining includes both quantitative and qualitative assessments. Various stakeholders have interests in the business decision-making that will be accomplished or supported by the resultant models. In many cases, these stakeholders need to "sign off" on the deployment of the models, and in order to do so need to be satisfied by the quality of the model's decisions. What that means varies from application to application, but often stakeholders are looking to see whether the model is going to do more good than harm, and especially that the model is unlikely to make catastrophic

mistakes.[2] To facilitate such qualitative assessment, the data scientist must think about the *comprehensibility* of the model to stakeholders (not just to the data scientists). And if the model itself is not comprehensible (e.g., maybe the model is a very complex mathematical formula), how can the data scientists work to make the behavior of the model be comprehensible.

Finally, a comprehensive evaluation framework is important because getting detailed information on the performance of a deployed model may be difficult or impossible. Often there is only limited access to the deployment environment so making a comprehensive evaluation "in production" is difficult. Deployed systems typically contain many "moving parts," and assessing the contribution of a single part is difficult. Firms with sophisticated data science teams wisely build testbed environments that mirror production data as closely as possible, in order to get the most realistic evaluations before taking the risk of deployment.

Nonetheless, in some cases we may want to extend evaluation into the development environment, for example by instrumenting a live system to be able to conduct randomized experiments. In our churn example, if we have decided from laboratory tests that a data mined model will give us better churn reduction, we may want to move on to an "in vivo" evaluation, in which a live system randomly applies the model to some customers while keeping other customers as a control group (recall our discussion of causal modeling from Chapter 1). Such experiments must be designed carefully, and the technical details are beyond the scope of this book. The interested reader could start with the lessons-learned articles by Ron Kohavi and his coauthors (Kohavi et al., 2007, 2009, 2012). We may also want to instrument deployed systems for evaluations to make sure that the world is not changing to the detriment of the model's decision-making. For example, behavior can change—in some cases, like fraud or spam, in direct response to the deployment of models. Additionally, the output of the model is critically dependent on the input data; input data can change in format and in substance, often without any alerting of the data science team. Raeder et al. (2012) present a detailed discussion of system design to help deal with these and other related evaluation-in-deployment issues.

Deployment

In deployment the results of data mining—and increasingly the data mining techniques themselves—are put into real use in order to realize some return on investment. The clearest cases of deployment involve implementing a predictive model in some information system or business process. In our churn example, a model for predicting the likelihood of churn could be integrated with the business process for churn management

2. For example, in one data mining project a model was created to diagnose problems in local phone networks, and to dispatch technicians to the likely site of the problem. Before deployment, a team of phone company stakeholders requested that the model be tweaked so that exceptions were made for hospitals.

—for example, by sending special offers to customers who are predicted to be particularly at risk. (We will discuss this in increasing detail as the book proceeds.) A new fraud detection model may be built into a workforce management information system, to monitor accounts and create "cases" for fraud analysts to examine.

Increasingly, the data mining techniques themselves are deployed. For example, for targeting online advertisements, systems are deployed that automatically build (and test) models in production when a new advertising campaign is presented. Two main reasons for deploying the data mining system itself rather than the models produced by a data mining system are (i) the world may change faster than the data science team can adapt, as with fraud and intrusion detection, and (ii) a business has too many modeling tasks for their data science team to manually curate each model individually. In these cases, it may be best to deploy the data mining phase into production. In doing so, it is critical to instrument the process to alert the data science team of any seeming anomalies and to provide fail-safe operation (Raeder et al., 2012).

Deployment can also be much less "technical." In a celebrated case, data mining discovered a set of rules that could help to quickly diagnose and fix a common error in industrial printing. The deployment succeeded simply by taping a sheet of paper containing the rules to the side of the printers (Evans & Fisher, 2002). Deployment can also be much more subtle, such as a change to data acquisition procedures, or a change to strategy, marketing, or operations resulting from insight gained from mining the data.

Deploying a model into a production system typically requires that the model be recoded for the production environment, usually for greater speed or compatibility with an existing system. This may incur substantial expense and investment. In many cases, the data science team is responsible for producing a working prototype, along with its evaluation. These are passed to a development team.

Practically speaking, there are risks with "over the wall" transfers from data science to development. It may be helpful to remember the maxim: "Your model is not what the data scientists design, it's what the engineers build." From a management perspective, it is advisable to have members of the development team involved early on in the data science project. They can begin as advisors, providing critical insight to the data science team. Increasingly in practice, these particular developers are "data science engineers"—software engineers who have particular expertise both in the production systems and in data science. These developers gradually assume more responsibility as the project matures. At some point the developers will take the lead and assume ownership of the product. Generally, the data scientists

should still remain involved in the project into final deployment, as advisors or as developers depending on their skills.

Regardless of whether deployment is successful, the process often returns to the Business Understanding phase. The process of mining data produces a great deal of insight into the business problem and the difficulties of its solution. A second iteration can yield an improved solution. Just the experience of thinking about the business, the data, and the performance goals often leads to new ideas for improving business performance, and even new lines of business or new ventures.

Note that it is not necessary to fail in deployment to start the cycle again. The Evaluation stage may reveal that results are not good enough to deploy, and we need to adjust the problem definition or get different data. This is represented by the "shortcut" link from Evaluation back to Business Understanding in the process diagram. In practice, there should be shortcuts back from each stage to each prior one because the process always retains some exploratory aspects, and a project should be flexible enough to revisit prior steps based on discoveries made.[3]

Implications for Managing the Data Science Team

It is tempting—but usually a mistake—to view the data mining process as a software development cycle. Indeed, data mining projects are often treated and managed as engineering projects, which is understandable when they are initiated by software departments, with data generated by a large software system and analytics results fed back into it. Managers are usually familiar with software technologies and are comfortable managing software projects. Milestones can be agreed upon and success is usually unambiguous. Software managers might look at the CRISP data mining cycle (Figure 2-2) and think it looks comfortably similar to a software development cycle, so they should be right at home managing an analytics project the same way.

This can be a mistake because data mining is an exploratory undertaking closer to research and development than it is to engineering. The CRISP cycle is based around exploration; it iterates on *approaches* and *strategy* rather than on software designs. Outcomes are far less certain, and the results of a given step may change the fundamental understanding of the problem. Engineering a data mining solution directly for deployment can be an expensive premature commitment. Instead, analytics projects should prepare to invest in information to reduce uncertainty in various ways. Small investments can be made via pilot studies and throwaway prototypes. Data scientists should

3. Software professionals may recognize the similarity to the philosophy of "Fail faster to succeed sooner" (Muoio, 1997).

review the literature to see what else has been done and how it has worked. On a larger scale, a team can invest substantially in building experimental testbeds to allow extensive agile experimentation. If you're a software manager, this will look more like research and exploration than you're used to, and maybe more than you're comfortable with.

Software skills versus analytics skills

Although data mining involves software, it also requires skills that may not be common among programmers. In software engineering, the ability to write efficient, high-quality code from requirements may be paramount. Team members may be evaluated using software metrics such as the amount of code written or number of bug tickets closed. In analytics, it's more important for individuals to be able to formulate problems well, to prototype solutions quickly, to make reasonable assumptions in the face of ill-structured problems, to design experiments that represent good investments, and to analyze results. In building a data science team, these qualities, rather than traditional software engineering expertise, are skills that should be sought.

Other Analytics Techniques and Technologies

Business analytics involves the application of various technologies to the analysis of data. Many of these go beyond this book's focus on data-analytic thinking and the principles of extracting useful patterns from data. Nonetheless, it is important to be acquainted with these related techniques, to understand what their goals are, what role they play, and when it may be beneficial to consult experts in them.

To this end, we present six groups of related analytic techniques. Where appropriate we draw comparisons and contrasts with data mining. The main difference is that data mining focuses on the *automated* search for *knowledge*, *patterns*, or *regularities* from data.[4] An important skill for a business analyst is to be able to recognize what sort of analytic technique is appropriate for addressing a particular problem.

Statistics

The term "statistics" has two different uses in business analytics. First, it is used as a catchall term for the computation of particular numeric values of interest from data (e.g., "We need to gather some statistics on our customers' usage to determine what's going wrong here.") These values often include sums, averages, rates, and so on. Let's

4. It is important to keep in mind that it is rare for the discovery to be completely automated. The important factor is that data mining automates at least partially the search and discovery process, rather than providing technical support for manual search and discovery.

call these "summary statistics." Often we want to dig deeper, and calculate summary statistics *conditionally* on one or more subsets of the population (e.g., "Does the churn rate differ between male and female customers?" and "What about high-income customers in the Northeast (denotes a region of the USA)?") Summary statistics are the basic building blocks of much data science theory and practice.

Summary statistics should be chosen with close attention to the business problem to be solved (one of the fundamental principles we will present later), and also with attention to the *distribution* of the data they are summarizing. For example, the average (mean) income in the United States according to the 2004 Census Bureau Economic Survey was over $60,000. If we were to use that as a measure of the average income in order to make policy decisions, we would be misleading ourselves. The distribution of incomes in the U.S. is highly skewed, with many people making relatively little and some people making fantastically much. In such cases, the arithmetic mean tells us relatively little about how much people are making. Instead, we should use a different measure of "average" income, such as the median. The median income—that amount where half the population makes more and half makes less—in the U.S. in the 2004 Census study was only $44,389—considerably less than the mean. This example may seem obvious because we are so accustomed to hearing about the "median income," but the same reasoning applies to any computation of summary statistics: have you thought about the problem you would like to solve or the question you would like to answer? Have you considered the distribution of the data, and whether the chosen statistic is appropriate?

The other use of the term "statistics" is to denote the field of study that goes by that name, for which we might differentiate by using the proper name, Statistics. The field of Statistics provides us with a huge amount of knowledge that underlies analytics, and can be thought of as a component of the larger field of Data Science. For example, Statistics helps us to understand different data distributions and what statistics are appropriate to summarize each. Statistics helps us understand how to use data to test hypotheses and to estimate the uncertainty of conclusions. In relation to data mining, hypothesis testing can help determine whether an observed pattern is likely to be a valid, general regularity as opposed to a chance occurrence in some particular dataset. Most relevant to this book, many of the techniques for extracting models or patterns from data have their roots in Statistics.

For example, a preliminary study may suggest that customers in the Northeast have a churn rate of 22.5%, whereas the nationwide average churn rate is only 15%. This may be just a chance fluctuation since the churn rate is not constant; it varies over regions and over time, so differences are to be expected. But the Northeast rate is one and a half times the U.S. average, which seems unusually high. What is the chance that this is due to random variation? Statistical hypothesis testing is used to answer such questions.

Closely related is the quantification of uncertainty into confidence intervals. The overall churn rate is 15%, but there is some variation; traditional statistical analysis may reveal that 95% of the time the churn rate is expected to fall between 13% and 17%.

This contrasts with the (complementary) process of data mining, which may be seen as hypothesis *generation*. Can we find patterns in data in the first place? Hypothesis generation should then be followed by careful hypothesis testing (generally on different data; see Chapter 5). In addition, data mining procedures may produce numerical estimates, and we often also want to provide confidence intervals on these estimates. We will return to this when we discuss the evaluation of the results of data mining.

In this book we are not going to spend more time discussing these basic statistical concepts. There are plenty of introductory books on statistics and statistics for business, and any treatment we would try to squeeze in would be either very narrow or superficial.

That said, one statistical term that is often heard in the context of business analytics is "correlation." For example, "Are there any indicators that correlate with a customer's later defection?" As with the term statistics, "correlation" has both a general-purpose meaning (variations in one quantity tell us something about variations in the other), and a specific technical meaning (e.g., linear correlation based on a particular mathematical formula). The notion of correlation will be the jumping off point for the rest of our discussion of data science for business, starting in the next chapter.

Database Querying

A *query* is a specific request for a subset of data or for statistics about data, formulated in a technical language and posed to a database system. Many tools are available to answer one-off or repeating queries about data posed by an analyst. These tools are usually frontends to database systems, based on Structured Query Language (SQL) or a tool with a graphical user interface (GUI) to help formulate queries (e.g., query-by-example, or QBE). For example, if the analyst can define "profitable" in operational terms computable from items in the database, then a query tool could answer: "Who are the most profitable customers in the Northeast?" The analyst may then run the query to retrieve a list of the most profitable customers, possibly ranked by profitability. This activity differs fundamentally from data mining in that there is no discovery of patterns or models.

Database queries are appropriate when an analyst already has an idea of what might be an interesting subpopulation of the data, and wants to investigate this population or confirm a hypothesis about it. For example, if an analyst suspects that middle-aged men living in the Northeast have some particularly interesting churning behavior, she could compose a SQL query:

```
SELECT * FROM CUSTOMERS WHERE AGE > 45 and SEX='M' and DOMICILE = 'NE'
```

If those are the people to be targeted with an offer, a query tool can be used to retrieve all of the information about them ("*") from the `CUSTOMERS` table in the database.

In contrast, data mining could be used to come up with this query in the first place—as a pattern or regularity in the data. A data mining procedure might examine prior customers who did and did not defect, and determine that this segment (characterized as "AGE is greater than 45 and SEX is male and DOMICILE is Northeast-USA") is predictive with respect to churn rate. After translating this into a SQL query, a query tool could then be used to find the matching records in the database.

Query tools generally have the ability to execute sophisticated logic, including computing summary statistics over subpopulations, sorting, joining together multiple tables with related data, and more. Data scientists often become quite adept at writing queries to extract the data they need.

On-line Analytical Processing (OLAP) provides an easy-to-use GUI to query large data collections, for the purpose of facilitating data exploration. The idea of "on-line" processing is that it is done in realtime, so analysts and decision makers can find answers to their queries quickly and efficiently. Unlike the "ad hoc" querying enabled by tools like SQL, for OLAP the dimensions of analysis must be pre-programmed into the OLAP system. If we've foreseen that we would want to explore sales volume by region and time, we could have these three dimensions programmed into the system, and drill down into populations, often simply by clicking and dragging and manipulating dynamic charts.

OLAP systems are designed to facilitate manual or visual exploration of the data by analysts. OLAP performs no modeling or automatic pattern finding. As an additional contrast, unlike with OLAP, data mining tools generally can incorporate new dimensions of analysis easily as part of the exploration. OLAP tools can be a useful complement to data mining tools for discovery from business data.

Data Warehousing

Data warehouses collect and coalesce data from across an enterprise, often from multiple transaction-processing systems, each with its own database. Analytical systems can access data warehouses. Data warehousing may be seen as a facilitating technology of data mining. It is not always necessary, as most data mining does not access a data warehouse, but firms that decide to invest in data warehouses often can apply data mining more broadly and more deeply in the organization. For example, if a data warehouse integrates records from sales and billing as well as from human resources, it can be used to find characteristic patterns of effective salespeople.

Regression Analysis

Some of the same methods we discuss in this book are at the core of a different set of analytic methods, which often are collected under the rubric *regression analysis*, and are widely applied in the field of statistics and also in other fields founded on econometric analysis. This book will focus on different issues than usually encountered in a regression analysis book or class. Here we are less interested in explaining a particular dataset as we are in extracting patterns that will generalize to other data, and for the purpose of improving some business process. Typically, this will involve estimating or predicting values for cases that are not in the analyzed data set. So, as an example, in this book we are less interested in digging into the reasons for churn (important as they may be) in a particular historical set of data, and more interested in predicting which customers who have not yet left would be the best to target to reduce future churn. Therefore, we will spend some time talking about testing patterns on new data to evaluate their generality, and about techniques for reducing the tendency to find patterns specific to a particular set of data, but that do not generalize to the population from which the data come.

The topic of explanatory modeling versus predictive modeling can elicit deep-felt debate,[5] which goes well beyond our focus. What is important is to realize that there is considerable overlap in the *techniques* used, but that the lessons learned from explanatory modeling do not all apply to predictive modeling. So a reader with some background in regression analysis may encounter new and even seemingly contradictory lessons.[6]

Machine Learning and Data Mining

The collection of methods for extracting (predictive) models from data, now known as machine learning methods, were developed in several fields contemporaneously, most notably Machine Learning, Applied Statistics, and Pattern Recognition. Machine Learning as a field of study arose as a subfield of Artificial Intelligence, which was concerned with methods for improving the knowledge or performance of an intelligent agent over time, in response to the agent's experience in the world. Such improvement often involves analyzing data from the environment and making predictions about unknown quantities, and over the years this data analysis aspect of machine learning has come to play a very large role in the field. As machine learning methods were deployed broadly, the scientific disciplines of Machine Learning, Applied Statistics, and Pattern Recognition developed close ties, and the separation between the fields has blurred.

5. The interested reader is urged to read the discussion by Shmueli (2010).

6. Those who pursue the study in depth will have the seeming contradictions worked out. Such deep study is not necessary to understand the fundamental principles.

The field of Data Mining (or KDD: Knowledge Discovery and Data Mining) started as an offshoot of Machine Learning, and they remain closely linked. Both fields are concerned with the analysis of data to find useful or informative patterns. Techniques and algorithms are shared between the two; indeed, the areas are so closely related that researchers commonly participate in both communities and transition between them seamlessly. Nevertheless, it is worth pointing out some of the differences to give perspective.

Speaking generally, because Machine Learning is concerned with many types of performance improvement, it includes subfields such as robotics and computer vision that are not part of KDD. It also is concerned with issues of *agency* and *cognition*—how will an intelligent agent use learned knowledge to reason and act in its environment—which are not concerns of Data Mining.

Historically, KDD spun off from Machine Learning as a research field focused on concerns raised by examining real-world applications, and a decade and a half later the KDD community remains more concerned with applications than Machine Learning is. As such, research focused on commercial applications and business issues of data analysis tends to gravitate toward the KDD community rather than to Machine Learning. KDD also tends to be more concerned with the entire process of data analytics: data preparation, model learning, evaluation, and so on.

Answering Business Questions with These Techniques

To illustrate how these techniques apply to business analytics, consider a set of questions that may arise and the technologies that would be appropriate for answering them. These questions are all related but each is subtly different. It is important to understand these differences in order to understand what technologies one needs to employ and what people may be necessary to consult.

1. *Who are the most profitable customers?*

   If "profitable" can be defined clearly based on existing data, this is a straightforward database query. A standard query tool could be used to retrieve a set of customer records from a database. The results could be sorted by cumulative transaction amount, or some other operational indicator of profitability.

2. *Is there really a difference between the profitable customers and the average customer?*

   This is a question about a conjecture or hypothesis (in this case, "There is a difference in value to the company between the profitable customers and the average customer"), and statistical hypothesis testing would be used to confirm or disconfirm it. Statistical analysis could also derive a probability or confidence bound that the difference was real. Typically, the result would be like: "The value of these profitable customers is significantly different from that of the average customer, with probability < 5% that this is due to random chance."

3. *But who really are these customers? Can I characterize them?*

   We often would like to do more than just list out the profitable customers. We would like to describe common characteristics of profitable customers. The characteristics of individual customers can be extracted from a database using techniques such as database querying, which also can be used to generate summary statistics. A deeper analysis should involve determining what characteristics *differentiate* profitable customers from unprofitable ones. This is the realm of data science, using data mining techniques for automated pattern finding—which we discuss in depth in the subsequent chapters.

4. *Will some particular new customer be profitable? How much revenue should I expect this customer to generate?*

   These questions could be addressed by data mining techniques that examine historical customer records and produce predictive models of profitability. Such techniques would generate models from historical data that could then be applied to new customers to generate predictions. Again, this is the subject of the following chapters.

Note that this last pair of questions are subtly different data mining questions. The first, a classification question, may be phrased as a prediction of whether a given new customer will be profitable (yes/no or the probability thereof). The second may be phrased as a prediction of the value (numerical) that the customer will bring to the company. More on that as we proceed.

Summary

Data mining is a craft. As with many crafts, there is a well-defined process that can help to increase the likelihood of a successful result. This process is a crucial conceptual tool for thinking about data science projects. We will refer back to the data mining process repeatedly throughout the book, showing how each fundamental concept fits in. In turn, understanding the fundamentals of data science substantially improves the chances of success as an enterprise invokes the data mining process.

The various fields of study related to data science have developed a set of canonical task types, such as classification, regression, and clustering. Each task type serves a different purpose and has an associated set of solution techniques. A data scientist typically attacks a new project by decomposing it such that one or more of these canonical tasks is revealed, choosing a solution technique for each, then composing the solutions. Doing this expertly may take considerable experience and skill. A successful data mining project involves an intelligent compromise between what the data can do (i.e., what they can predict, and how well) and the project goals. For this reason it is important to keep in mind how data mining results will be used, and use this to inform the data mining process itself.

Data mining differs from, and is complementary to, important supporting technologies such as statistical hypothesis testing and database querying (which have their own books and classes). Though the boundaries between data mining and related techniques are not always sharp, it is important to know about other techniques' capabilities and strengths to know when they should be used.

To a business manager, the data mining process is useful as a framework for analyzing a data mining project or proposal. The process provides a systematic organization, including a set of questions that can be asked about a project or a proposed project to help understand whether the project is well conceived or is fundamentally flawed. We will return to this after we have discussed in detail some more of the fundamental principles themselves—to which we turn now.

CHAPTER 3

Introduction to Predictive Modeling: From Correlation to Supervised Segmentation

Fundamental concepts: *Identifying informative attributes; Segmenting data by progressive attribute selection.*

Exemplary techniques: *Finding correlations; Attribute/variable selection; Tree induction.*

The previous chapters discussed models and modeling at a high level. This chapter delves into one of the main topics of data mining: predictive modeling. Following our example of data mining for churn prediction from the first section, we will begin by thinking of predictive modeling as *supervised* segmentation—how can we segment the population into groups that differ from each other with respect to some quantity of interest. In particular, how can we segment the population with respect to something that we would like to predict or estimate. The target of this prediction can be something we would like to avoid, such as which customers are likely to leave the company when their contracts expire, which accounts have been defrauded, which potential customers are likely not to pay off their account balances (*write-offs*, such as defaulting on one's phone bill or credit card balance), or which web pages contain objectionable content. The target might instead be cast in a positive light, such as which consumers are most likely to respond to an advertisement or special offer, or which web pages are most appropriate for a search query.

In the process of discussing supervised segmentation, we introduce one of the fundamental ideas of data mining: finding or selecting important, informative variables or "attributes" of the entities described by the data. What exactly it means to be "informative" varies among applications, but generally, *information is a quantity that reduces uncertainty about something.* So, if an old pirate gives me information about where his treasure is hidden that does not mean that I know for certain where it is, it only means that my uncertainty about where the treasure is hidden is reduced. The better the information, the more my uncertainty is reduced.

Now, recall the notion of "supervised" data mining from the previous chapter. A key to supervised data mining is that we have some target quantity we would like to predict or to otherwise understand better. Often this quantity is unknown or unknowable at the time we would like to make a business decision, such as whether a customer will churn soon after her contract expires, or which accounts have been defrauded. Having a target variable crystalizes our notion of finding informative attributes: is there one or more other variables that reduces our uncertainty about the value of the target? This also gives a common analytics application of the general notion of correlation discussed above: we would like to find knowable attributes that correlate with the target of interest —that reduce our uncertainty in it. Just finding these correlated variables may provide important insight into the business problem.

Finding informative attributes also is useful to help us deal with increasingly larger databases and data streams. Datasets that are too large pose computational problems for analytic techniques, especially when the analyst does not have access to high-performance computers. One tried-and-true method for analyzing very large datasets is first to select a subset of the data to analyze. Selecting informative attributes provides an "intelligent" method for selecting an informative subset of the data. In addition, attribute selection prior to data-driven modeling can increase the accuracy of the modeling, for reasons we will discuss in Chapter 5.

Finding informative attributes also is the basis for a widely used predictive modeling technique called *tree induction*, which we will introduce toward the end of this chapter as an application of this fundamental concept. Tree induction incorporates the idea of supervised segmentation in an elegant manner, repeatedly selecting informative attributes. By the end of this chapter we will have achieved an understanding of: the basic concepts of predictive modeling; the fundamental notion of finding informative attributes, along with one particular, illustrative technique for doing so; the notion of tree-structured models; and a basic understanding of the process for extracting tree-structured models from a dataset—performing supervised segmentation.

Models, Induction, and Prediction

Generally speaking, a model is a simplified representation of reality created to serve a purpose. It is simplified based on some assumptions about what is and is not important for the specific purpose, or sometimes based on constraints on information or tractability. For example, a map is a model of the physical world. It abstracts away a tremendous amount of information that the mapmaker deemed irrelevant for its purpose. It preserves, and sometimes further simplifies, the relevant information. For example, a road map keeps and highlights the roads, their basic topology, their relationships to places one would want to travel, and other relevant information. Various professions have well-known model types: an architectural blueprint, an engineering prototype, the

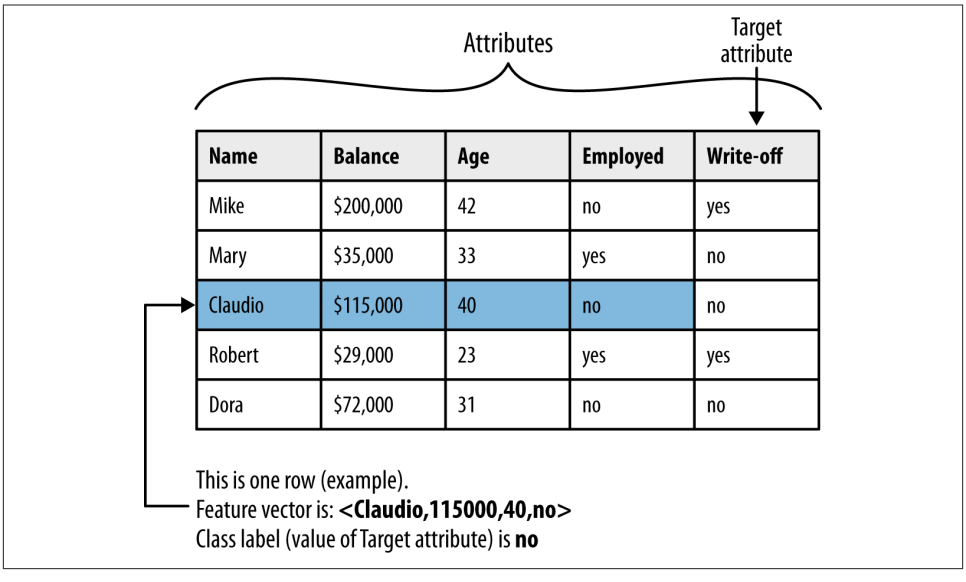


Figure 3-1. Data mining terminology for a supervised classification problem. The problem is supervised because it has a target attribute and some "training" data where we know the value for the target attribute. It is a classification (rather than regression) problem because the target is a category (yes or no) rather than a number.

Black-Scholes model of option pricing, and so on. Each of these abstracts away details that are not relevant to their main purpose and keeps those that are.

In data science, a predictive model is a formula for estimating the unknown value of interest: the target. The formula could be mathematical, or it could be a logical statement such as a rule. Often it is a hybrid of the two. Given our division of supervised data mining into classification and regression, we will consider classification models (and class-probability estimation models) and regression models.

Terminology: Prediction

In common usage, prediction means to forecast a future event. In data science, prediction more generally means *to estimate an unknown value*. This value could be something in the future (in common usage, true prediction), but it could also be something in the present or in the past. Indeed, since data mining usually deals with historical data, models very often are built and tested using events from the past. Predictive models for credit scoring estimate the likelihood that a potential customer will default (become a write-off). Predictive models for spam filtering estimate whether a given piece of email is spam. Predictive models for fraud detection judge wheth-

er an account has been defrauded. The key is that the model is intended to be used to estimate an unknown value.

This is in contrast to *descriptive* modeling, where the primary purpose of the model is not to estimate a value but instead to gain insight into the underlying phenomenon or process. A descriptive model of churn behavior would tell us what customers who churn typically look like.[1] A descriptive model must be judged in part on its intelligibility, and a less accurate model may be preferred if it is easier to understand. A predictive model may be judged solely on its predictive performance, although we will discuss why intelligibility is nonetheless important. The difference between these model types is not as strict as this may imply; some of the same techniques can be used for both, and usually one model can serve both purposes (though sometimes poorly). Sometimes much of the value of a predictive model is in the understanding gained from looking at it rather than in the predictions it makes.

Before we discuss predictive modeling further, we must introduce some terminology. Supervised learning is model creation where the model describes a relationship between a set of selected variables (*attributes* or *features*) and a predefined variable called the *target* variable. The model estimates the value of the target variable as a function (possibly a probabilistic function) of the features. So, for our churn-prediction problem we would like to build a model of the propensity to churn as a function of customer account attributes, such as age, income, length with the company, number of calls to customer service, overage charges, customer demographics, data usage, and others.

Figure 3-1 illustrates some of the terminology we introduce here, in an oversimplified example problem of credit write-off prediction. An *instance* or *example* represents a fact or a data point—in this case a historical customer who had been given credit. This is also called a *row* in database or spreadsheet terminology. An instance is described by a set of *attributes* (fields, columns, variables, or features). An instance is also sometimes called a *feature vector*, because it can be represented as a fixed-length ordered collection (vector) of feature values. Unless stated otherwise, we will assume that the values of all the attributes (but not the target) are present in the data.

1. Descriptive modeling often is used to work toward a causal understanding of the data generating process (*why do people churn?*).

Many Names for the Same Things

The principles and techniques of data science historically have been studied in several different fields, including machine learning, pattern recognition, statistics, databases, and others. As a result there often are several different names for the same things. We typically will refer to a *dataset*, whose form usually is the same as a *table* of a database or a *worksheet* of a spreadsheet. A dataset contains a set of *examples* or *instances*. An instance also is referred to as a *row* of a database table or sometimes a *case* in statistics.

The features (table columns) have many different names as well. Statisticians speak of *independent variables* or *predictors* as the attributes supplied as input. In operations research you may also hear *explanatory variable*. The target variable, whose values are to be predicted, is commonly called the *dependent variable* in statistics. This terminology may be somewhat confusing; the independent variables may not be independent of each other (or anything else), and the dependent variable doesn't always depend on all the independent variables. For this reason we have avoided the dependent/independent terminology in this book. Some experts consider the target variable to be included in the set of features, some do not. The important thing is rather obvious: the target variable is not used to predict itself. However, it may be that prior values for the target variable are quite helpful to predict future values—so such prior values may be included as features.

The creation of models from data is known as model induction. Induction is a term from philosophy that refers to generalizing from specific cases to general rules (or laws, or truths). Our models are general rules in a statistical sense (they usually do not hold 100% of the time; often not nearly), and the procedure that creates the model from the data is called the induction algorithm or learner. Most inductive procedures have variants that induce models both for classification and for regression. We will discuss mainly classification models because they tend to receive less attention in other treatments of statistics, and because they are relevant to many business problems (and thus much work in data science focuses on classification).

Terminology: Induction and deduction

Induction can be contrasted with *deduction*. Deduction starts with general rules and specific facts, and creates other specific facts from them. The *use* of our models can be considered a procedure of (probabilistic) deduction. We will get to this shortly.

The input data for the induction algorithm, used for inducing the model, are called the *training* data. As mentioned in Chapter 2, they are called *labeled* data because the value for the target variable (the label) is known.

Let's return to our example churn problem. Based on what we learned in Chapter 1 and Chapter 2, we might decide that in the modeling stage we should build a "supervised segmentation" model, which divides the sample into segments having (on average) higher or lower tendency to leave the company after contract expiration. To think about how this might be done, let's now turn to one of our fundamental concepts: How can we select one or more attributes/features/variables that will best divide the sample *with respect to our target variable of interest*?

Supervised Segmentation

Recall that a predictive model focuses on estimating the value of some particular target variable of interest. An intuitive way of thinking about extracting patterns from data in a supervised manner is to try to segment the population into subgroups that have different values for the target variable (and within the subgroup the instances have similar values for the target variable). If the segmentation is done using values of variables that will be known when the target is not, then these segments can be used to predict the value of the target variable. Moreover, the segmentation may at the same time provide a human-understandable set of segmentation patterns. One such segment expressed in English might be: "Middle-aged professionals who reside in New York City on average have a churn rate of 5%." Specifically, the term "middle-aged professionals who reside in New York City" is the definition of the segment (which references some particular attributes) and "a churn rate of 5%" describes the predicted value of the target variable for the segment.[2]

Often we are interested in applying data mining when we have many attributes, and are not sure exactly what the segments should be. In our churn-prediction problem, who is to say what are the best segments for predicting the propensity to churn? If there exist in the data segments with significantly different (average) values for the target variable, we would like to be able to extract them automatically.

This brings us to our fundamental concept: how can we judge whether a variable contains important information about the target variable? How much? We would like automatically to get a selection of the more informative variables with respect to the particular task at hand (namely, predicting the value of the target variable). Even better, we might like to rank the variables by how good they are at predicting the value of the target.

Consider just the selection of the single most informative attribute. Solving this problem will introduce our first concrete data mining technique—simple, but easily extendable to be very useful. In our example, what variable gives us the most information about

2. The predicted value can be estimated from the data in different ways, which we will get to. At this point we can think of it roughly as an average of some sort from the training data that fall into the segment.

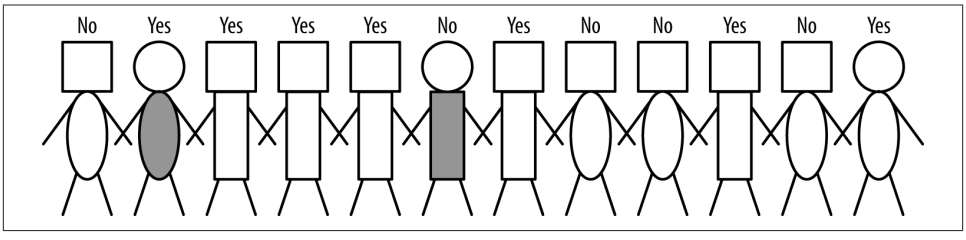


Figure 3-2. A set of people to be classified. The label over each head represents the value of the target variable (write-off or not). Colors and shapes represent different predictor attributes.

the future churn rate of the population? Being a professional? Age? Place of residence? Income? Number of complaints to customer service? Amount of overage charges?

We now will look carefully into one useful way to select informative variables, and then later will show how this technique can be used repeatedly to build a supervised segmentation. While very useful and illustrative, please keep in mind that direct, multivariate supervised segmentation is just one application of this fundamental idea of selecting informative variables. This notion should become one of your conceptual tools when thinking about data science problems more generally. For example, as we go forward we will delve into other modeling approaches, ones that do not incorporate variable selection directly. When the world presents you with very large sets of attributes, it may be (extremely) useful to harken back to this early idea and to select a subset of informative attributes. Doing so can substantially reduce the size of an unwieldy dataset, and as we will see, often will improve the accuracy of the resultant model.

Selecting Informative Attributes

Given a large set of examples, how do we select an attribute to partition them in an informative way? Let's consider a binary (two class) classification problem, and think about what we would like to get out of it. To be concrete, Figure 3-2 shows a simple segmentation problem: twelve people represented as stick figures. There are two types of heads: square and circular; and two types of bodies: rectangular and oval; and two of the people have gray bodies while the rest are white.

These are the attributes we will use to describe the people. Above each person is the binary target label, *Yes* or *No*, indicating (for example) whether the person becomes a loan write-off. We could describe the data on these people as:

- Attributes:
  - — head-shape: square, circular
  - — body-shape: rectangular, oval
  - — body-color: gray, white
- Target variable:
  - — write-off: Yes, No

So let's ask ourselves: which of the attributes would be best to segment these people into groups, in a way that will distinguish write-offs from non-write-offs? Technically, we would like the resulting groups to be as *pure* as possible. By pure we mean *homogeneous with respect to the target variable*. If every member of a group has the same value for the target, then the group is pure. If there is at least one member of the group that has a different value for the target variable than the rest of the group, then the group is impure.

Unfortunately, in real data we seldom expect to find a variable that will make the segments pure. However, if we can reduce the impurity substantially, then we can both learn something about the data (and the corresponding population), and importantly for this chapter, we can use the attribute in a predictive model—in our example, predicting that members of one segment will have higher or lower write-off rates than those in another segment. If we can do that, then we can for example offer credit to those with the lower predicted write-off rates, or can offer different credit terms based on the different predicted write-off rates.

Technically, there are several complications:

1. Attributes rarely split a group perfectly. Even if one subgroup happens to be pure, the other may not. For example, in Figure 3-2, consider if the second person were not there. Then *body-color=gray* would create a pure segment (*write-off=no*). However, the other associated segment, *body-color=white*, still is not pure.
2. In the prior example, the condition *body-color=gray* only splits off one single data point into the pure subset. Is this better than another split that does not produce any pure subset, but reduces the impurity more broadly?
3. Not all attributes are binary; many attributes have three or more distinct values. We must take into account that one attribute can split into two groups while another might split into three groups, or seven. How do we compare these?
4. Some attributes take on numeric values (continuous or integer). Does it make sense to make a segment for every numeric value? (No.) How should we think about creating supervised segmentations using numeric attributes?

Fortunately, for classification problems we can address all the issues by creating a formula that evaluates how well each attribute splits a set of examples into segments, with respect to a chosen target variable. Such a formula is based on a *purity measure*.

The most common splitting criterion is called *information gain*, and it is based on a purity measure called *entropy*. Both concepts were invented by one of the pioneers of information theory, Claude Shannon, in his seminal work in the field (Shannon, 1948).

Entropy is a measure of disorder that can be applied to a set, such as one of our individual segments. Consider that we have a set of *properties* of members of the set, and each member has one and only one of the properties. In supervised segmentation, the member properties will correspond to the values of the target variable. Disorder corresponds to how mixed (impure) the segment is with respect to these properties of interest. So, for example, a mixed up segment with lots of write-offs and lots of non-write-offs would have high entropy.

More technically, entropy is defined as:

Equation 3-1. Entropy

$$entropy = - p_1 \log(p_1) - p_2 \log(p_2) - \cdots$$

Each p_i is the probability (the relative percentage) of property i within the set, ranging from $p_i = 1$ when all members of the set have property i, and $p_i = 0$ when no members of the set have property i. The … simply indicates that there may be more than just two properties (and for the technically minded, the logarithm is generally taken as base 2).

Since the entropy equation might not lend itself to intuitive understanding, Figure 3-3 shows a plot of the entropy of a set containing 10 instances of two classes, + and –. We can see then that entropy measures the general disorder of the set, ranging from zero at minimum disorder (the set has members all with the same, single property) to one at maximal disorder (the properties are equally mixed). Since there are only two classes, $p_+ = 1 - p_-$. Starting with all negative instances at the lower left, $p_+ = 0$, the set has minimal disorder (it is pure) and the entropy is zero. If we start to switch class labels of elements of the set from – to +, the entropy increases. Entropy is maximized at 1 when the instance classes are balanced (five of each), and $p_+ = p_- = 0.5$. As more class labels are switched, the + class starts to predominate and the entropy lowers again. When all instances are positive, $p_+ = 1$ and entropy is minimal again at zero.

As a concrete example, consider a set S of 10 people with seven of the *non-write-off* class and three of the *write-off* class. So:

$$p(\text{non-write-off}) = 7 / 10 = 0.7$$
$$p(\text{write-off}) = 3 / 10 = 0.3$$

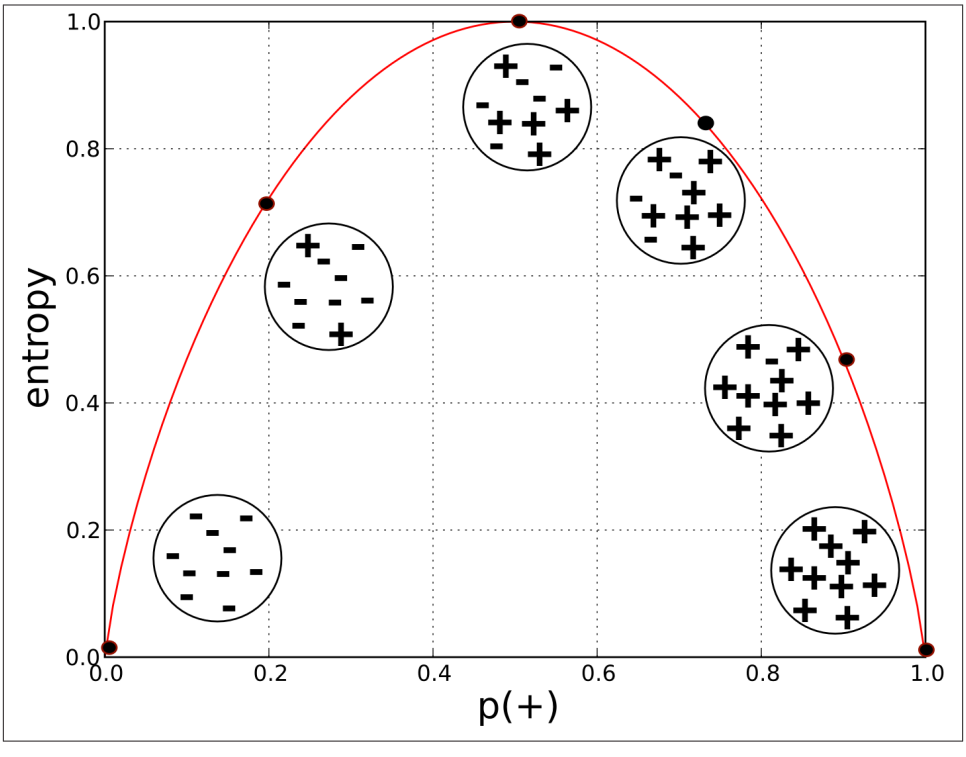


Figure 3-3. Entropy of a two-class set as a function of p(+).

$$\begin{aligned} entropy(S) &= -[0.7 \times \log_2(0.7) + 0.3 \times \log_2(0.3)] \\ &\approx -[0.7 \times -0.51 + 0.3 \times -1.74] \\ &\approx 0.88 \end{aligned}$$

Entropy is only part of the story. We would like to measure how *informative* an attribute is with respect to our target: how much gain in information it gives us about the value of the target variable. An attribute segments a set of instances into several subsets. Entropy only tells us how impure one individual subset is. Fortunately, with entropy to measure how disordered any set is, we can define *information gain* (IG) to measure how much an attribute improves (decreases) entropy over the whole segmentation it creates. Strictly speaking, information gain measures the *change* in entropy due to any amount of new information being added; here, in the context of supervised segmentation, we consider the information gained by splitting the set on all values of a single attribute. Let's say the attribute we split on has k different values. Let's call the original set of examples the *parent* set, and the result of splitting on the attribute values the k *children* sets. Thus, information gain is a function of both a parent set and of the children

resulting from some partitioning of the parent set—how much information has this attribute provided? That depends on how much purer the children are than the parent. Stated in the context of predictive modeling, if we were to know the value of this attribute, how much would it increase our knowledge of the value of the target variable?

Specifically, the definition of information gain (IG) is:

Equation 3-2. Information gain

$$IG(parent, children) = entropy(parent) - [p(c_1) \times entropy(c_1) + p(c_2) \times entropy(c_2) + \cdots]$$

Notably, the entropy for each child (c_i) is weighted by the proportion of instances belonging to that child, $p(c_i)$. This addresses directly our concern from above that splitting off a single example, and noticing that that set is pure, may not be as good as splitting the parent set into two nice large, relatively pure subsets, even if neither is pure.

As an example, consider the split in Figure 3-4. This is a two-class problem (• and ★). Examining the figure, the children sets certainly seem "purer" than the parent set. The parent set has 30 instances consisting of 16 dots and 14 stars, so:

$$\begin{aligned} entropy(parent) &= -[p(\bullet) \times \log_2 p(\bullet) + p(\star) \times \log_2 p(\star)] \\ &\approx -[0.53 \times -0.9 + 0.47 \times -1.1] \\ &\approx 0.99 \quad \text{(very impure)} \end{aligned}$$

The entropy of the *left* child is:

$$\begin{aligned} entropy(Balance < 50K) &= -[p(\bullet) \times \log_2 p(\bullet) + p(\star) \times \log_2 p(\star)] \\ &\approx -[0.92 \times (-0.12) + 0.08 \times (-3.7)] \\ &\approx 0.39 \end{aligned}$$

The entropy of the *right* child is:

$$\begin{aligned} entropy(Balance \geq 50K) &= -[p(\bullet) \times \log_2 p(\bullet) + p(\star) \times \log_2 p(\star)] \\ &\approx -[0.24 \times (-2.1) + 0.76 \times (-0.39)] \\ &\approx 0.79 \end{aligned}$$

Using Equation 3-2, the information gain of this split is:

$$
\begin{aligned}
IG &= entropy(parent) - [p(\text{Balance} < 50\text{K}) \times entropy(\text{Balance} < 50\text{K}) \\
&\quad + p(\text{Balance} \geq 50\text{K}) \times entropy(\text{Balance} \geq 50\text{K})] \\
&\approx 0.99 - [0.43 \times 0.39 + 0.57 \times 0.79] \\
&\approx 0.37
\end{aligned}
$$

So this split reduces entropy substantially. In predictive modeling terms, the attribute provides a lot of information on the value of the target.

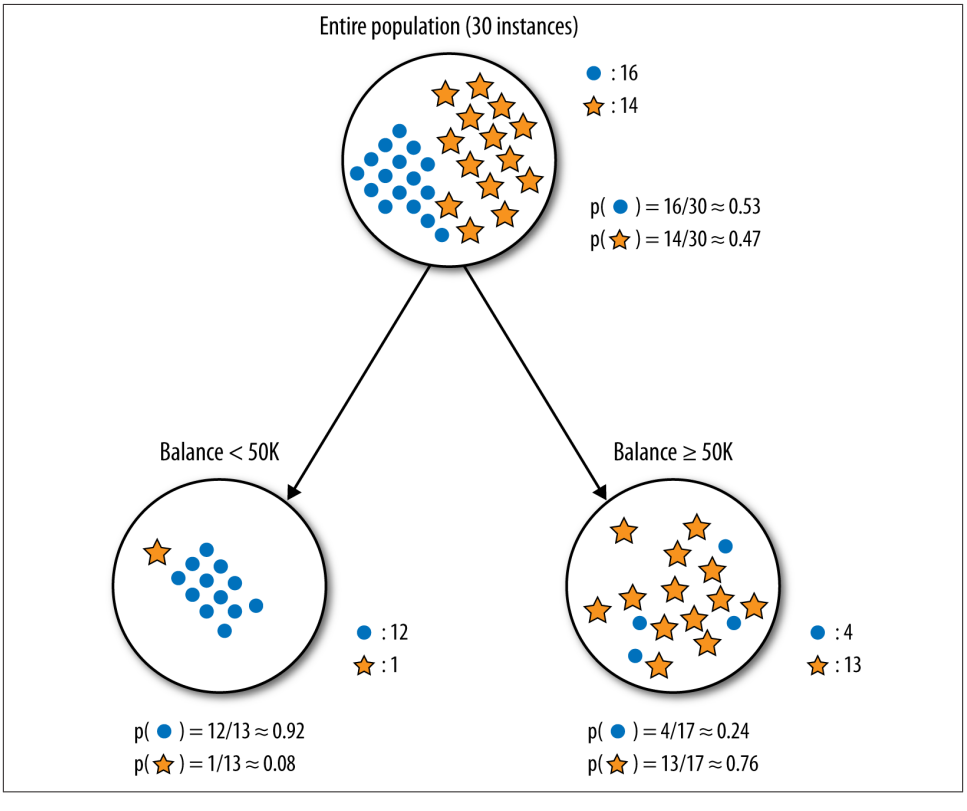


Figure 3-4. Splitting the "write-off" sample into two segments, based on splitting the Balance attribute (account balance) at 50K.

As a second example, consider another candidate split shown in Figure 3-5. This is the same parent set as in Figure 3-4, but instead we consider splitting on the attribute *Residence* with three values: OWN, RENT, and OTHER. Without showing the detailed calculations:

$$entropy(parent) \approx 0.99$$
$$entropy(\text{Residence=OWN}) \approx 0.54$$
$$entropy(\text{Residence=RENT}) \approx 0.97$$
$$entropy(\text{Residence=OTHER}) \approx 0.98$$
$$IG \approx 0.13$$

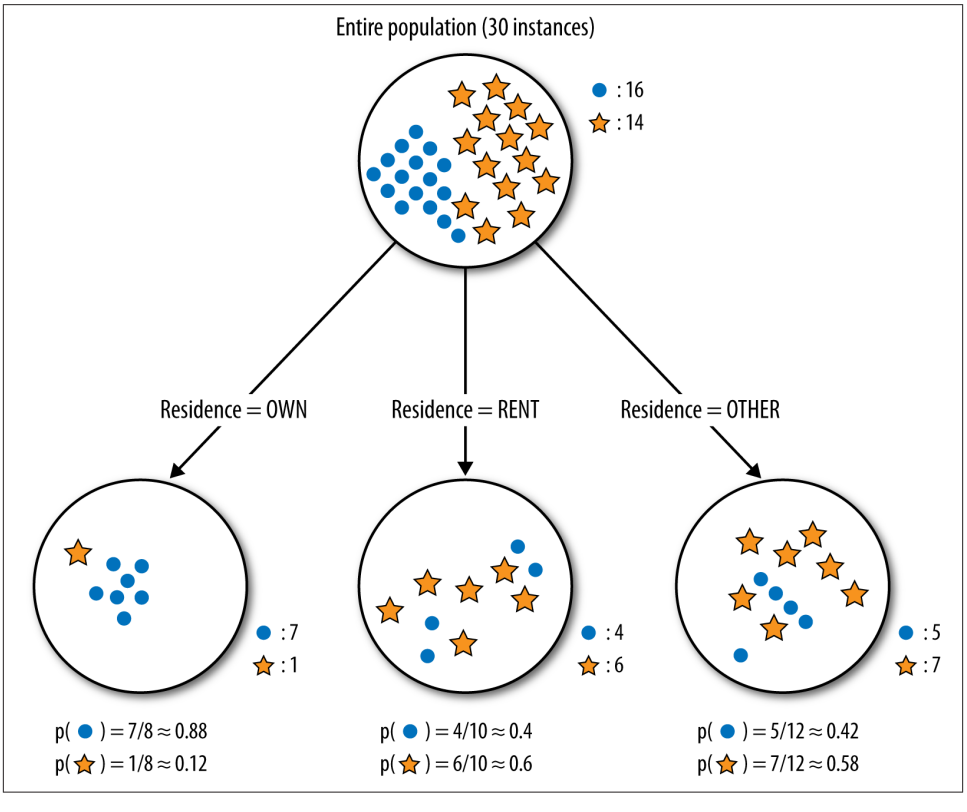


Figure 3-5. A classification tree split on the three-valued Residence attribute.

The Residence variable does have a positive information gain, but it is lower than that of Balance. Intuitively, this is because, while the one child Residence=OWN has considerably reduced entropy, the other values RENT and OTHER produce children that are no more pure than the parent. Thus, based on these data, the Residence variable is less informative than Balance.

Looking back at our concerns from above about creating supervised segmentation for classification problems, information gain addresses them all. It does not require absolute

purity. It can be applied to any number of child subsets. It takes into account the relative sizes of the children, giving more weight to larger subsets.[3]

Numeric variables

We have not discussed what exactly to do if the attribute is numeric. Numeric variables can be "discretized" by choosing a split point (or many split points) and then treating the result as a categorical attribute. For example, Income could be divided into two or more ranges. Information gain can be applied to evaluate the segmentation created by this discretization of the numeric attribute. We still are left with the question of how to choose the split point(s) for the numeric attribute. Conceptually, we can try all reasonable split points, and choose the one that gives the highest information gain.

Finally, what about supervised segmentations for regression problems—problems with a numeric target variable? Looking at reducing the impurity of the child subsets still makes intuitive sense, but information gain is not the right measure, because entropy-based information gain is based on the distribution of the *properties* in the segmentation. Instead, we would want a measure of the purity of the numeric (target) values in the subsets.

We will not go through a derivation here, but the fundamental idea is important: a natural measure of impurity for numeric values is *variance*. If the set has all the same values for the numeric target variable, then the set is pure and the variance is zero. If the numeric target values in the set are very different, then the set will have high variance. We can create a similar notion to information gain by looking at reductions in variance between parent and children. The process proceeds in direct analogy to the derivation for information gain above. To create the best segmentation given a numeric target, we might choose the one that produces the best weighted average variance reduction. In essence, we again would be finding variables that have the best correlation with the target, or alternatively, are most predictive of the target.

Example: Attribute Selection with Information Gain

Now we are ready to apply our first concrete data mining technique. For a dataset with instances described by attributes and a target variable, we can determine which attribute is the most informative with respect to estimating the value of the target variable. (We will delve into this more deeply below.) We also can rank a set of attributes by their informativeness, in particular by their information gain. This can be used simply to understand the data better. It can be used to help predict the target. Or it can be used

3. Technically, there remains a concern with attributes with very many values, as splitting on them may result in large information gain, but not be predictive. This problem ("overfitting") is the subject of Chapter 5.

to reduce the size of the data to be analyzed, by selecting a subset of attributes in cases where we can not or do not want to process the entire dataset.

To illustrate the use of information gain, we introduce a simple but realistic dataset taken from the machine learning dataset repository at the University of California at Irvine.[4] It is a dataset describing edible and poisonous mushrooms taken from The Audubon Society Field Guide to North American Mushrooms. From the description:

> This dataset includes descriptions of hypothetical samples corresponding to 23 species of gilled mushrooms in the Agaricus and Lepiota Family (pp. 500–525). Each species is identified as definitely edible, definitely poisonous, or of unknown edibility and not recommended. This latter class was combined with the poisonous one. The Guide clearly states that there is no simple rule for determining the edibility of a mushroom; no rule like "leaflets three, let it be" for Poisonous Oak and Ivy.

Each data example (instance) is one mushroom sample, described in terms of its observable attributes (the features). The twenty-odd attributes and the values for each are listed in Table 3-1. For a given example, each attribute takes on a single discrete value (e.g., *gill-color=black*). We use 5,644 examples from the dataset, comprising 2,156 poisonous and 3,488 edible mushrooms.

This is a classification problem because we have a target variable, called *edible?*, with two values *yes* (edible) and *no* (poisonous), specifying our two classes. Each of the rows in the training set has a value for this target variable. We will use information gain to answer the question: "Which single attribute is the most useful for distinguishing edible (edible?=Yes) mushrooms from poisonous (edible?=No) ones?" This is a basic attribute selection problem. In much larger problems we could imagine selecting the best ten or fifty attributes out of several hundred or thousand, and often you want do this if you suspect there are far too many attributes for your mining problem, or that many are not useful. Here, for simplicity, we will find the single best attribute instead of the top ten.

Table 3-1. The attributes of the Mushroom dataset

Attribute name	Possible values
CAP-SHAPE	`bell, conical, convex, flat, knobbed, sunken`
CAP-SURFACE	`fibrous, grooves, scaly, smooth`
CAP-COLOR	`brown, buff, cinnamon, gray, green, pink, purple, red, white, yellow`
BRUISES?	`yes, no`
ODOR	`almond, anise, creosote, fishy, foul, musty, none, pungent, spicy`
GILL-ATTACHMENT	`attached, descending, free, notched`
GILL-SPACING	`close, crowded, distant`

4. See this UC Irvine Machine Learning Repository page.

Attribute name	Possible values
GILL-SIZE	broad, narrow
GILL-COLOR	black, brown, buff, chocolate, gray, green, orange, pink, purple, red, white, yellow
STALK-SHAPE	enlarging, tapering
STALK-ROOT	bulbous, club, cup, equal, rhizomorphs, rooted, missing
STALK-SURFACE-ABOVE-RING	fibrous, scaly, silky, smooth
STALK-SURFACE-BELOW-RING	fibrous, scaly, silky, smooth
STALK-COLOR-ABOVE-RING	brown, buff, cinnamon, gray, orange, pink, red, white, yellow
STALK-COLOR-BELOW-RING	brown, buff, cinnamon, gray, orange, pink, red, white, yellow
VEIL-TYPE	partial, universal
VEIL-COLOR	brown, orange, white, yellow
RING-NUMBER	none, one, two
RING-TYPE	cobwebby, evanescent, flaring, large, none, pendant, sheathing, zone
SPORE-PRINT-COLOR	black, brown, buff, chocolate, green, orange, purple, white, yellow
POPULATION	abundant, clustered, numerous, scattered, several, solitary
HABITAT	grasses, leaves, meadows, paths, urban, waste, woods
EDIBLE? *(Target variable)*	yes, no

Since we now have a way to measure information gain this is straightforward: we are asking for the single attribute that gives the highest information gain.

To do this, we calculate the information gain achieved by splitting on each attribute. The information gain from Equation 3-2 is defined on a parent and a set of children. The parent in each case is the whole dataset. First we need *entropy*(*parent*), the entropy of the whole dataset. If the two classes were perfectly balanced in the dataset it would have an entropy of 1. This dataset is slightly unbalanced (more edible than poisonous mushrooms are represented) and its entropy is 0.96.

To illustrate entropy reduction graphically, we'll show a number of *entropy graphs* for the mushroom domain (Figure 3-6 through Figure 3-8). Each graph is a two-dimensional description of the entire dataset's entropy as it is divided in various ways by different attributes. On the *x* axis is the proportion of the dataset (0 to 1), and on the *y* axis is the entropy (also 0 to 1) of a given piece of the data. The amount of shaded area in each graph represents the amount of entropy in the dataset when it is divided by some chosen attribute (or not divided, in the case of Figure 3-6). Our goal of having the lowest entropy corresponds to having as *little* shaded area as possible.

The first chart, Figure 3-6, shows the entropy of the entire dataset. In such a chart, the highest possible entropy corresponds to the entire area being shaded; the lowest possible entropy corresponds to the entire area being white. Such a chart is useful for visualizing information gain from different partitions of a dataset, because any partition can be shown simply as slices of the graph (with widths corresponding to the proportion of the dataset), each with its own entropy. The weighted sum of entropies in the information gain calculation will be depicted simply by the total amount of shaded area.

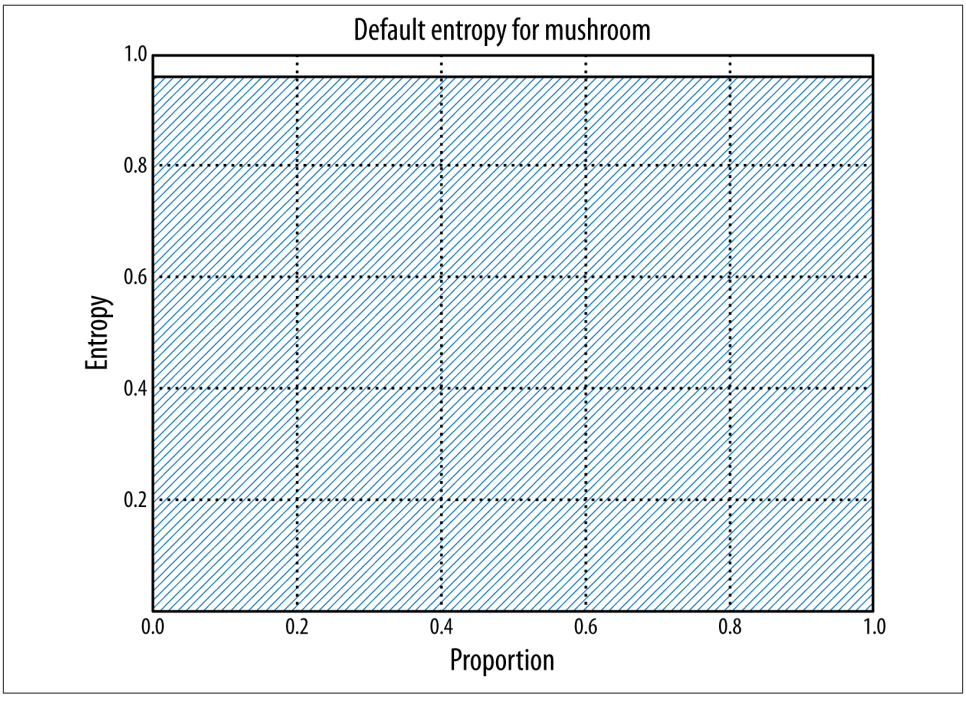


Figure 3-6. Entropy chart for the entire Mushroom dataset. The entropy for the entire dataset is 0.96, so 96% of the area is shaded.

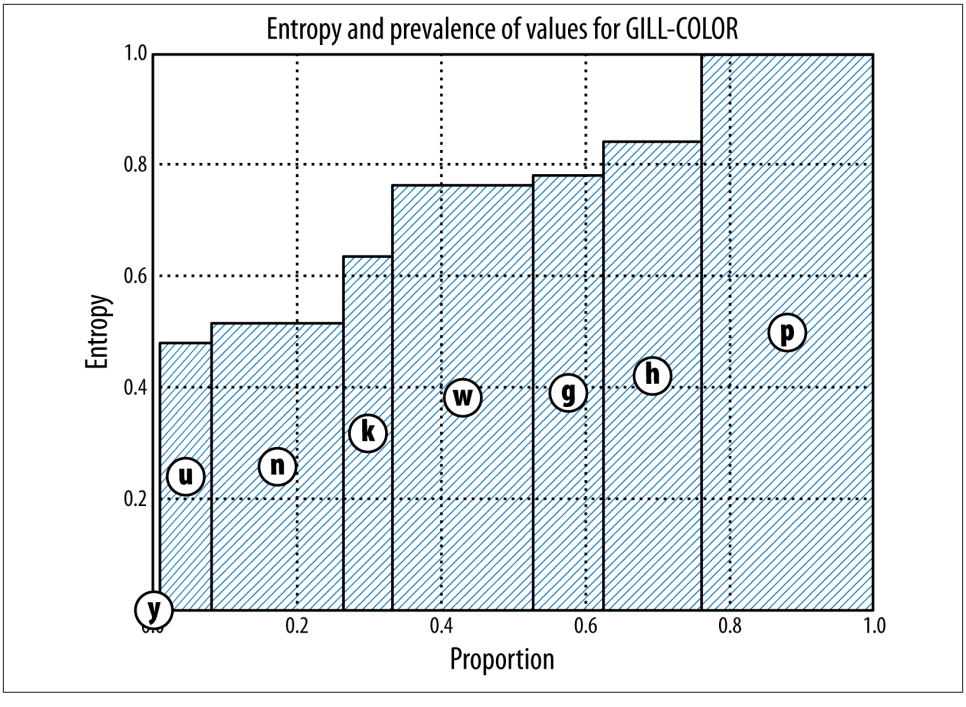


Figure 3-7. Entropy chart for the Mushroom dataset as split by GILL-COLOR. The amount of shading corresponds to the total (weighted sum) entropy, with each bar corresponding to the entropy of one of the attribute's values, and the width of the bar corresponding to the prevalence of that value in the data.

For our entire dataset, the global entropy is 0.96, so Figure 3-6 shows a large shaded area below the line $y = 0.96$. We can think of this as our starting entropy—any informative attribute should produce a new graph with less shaded area. Now we show the entropy charts of three sample attributes. Each value of an attribute occurs in the dataset with a different frequency, so each attribute splits the set in a different way.

Figure 3-7 shows the dataset split apart by the attribute GILL-COLOR, whose values are coded as y (`yellow`), u (`purple`), n (`brown`), and so on. The width of each attribute represents what proportion of the dataset has that value, and the height is its entropy. We can see that GILL-COLOR reduces the entropy somewhat; the shaded area in Figure 3-7 is considerably less than the area in Figure 3-6.

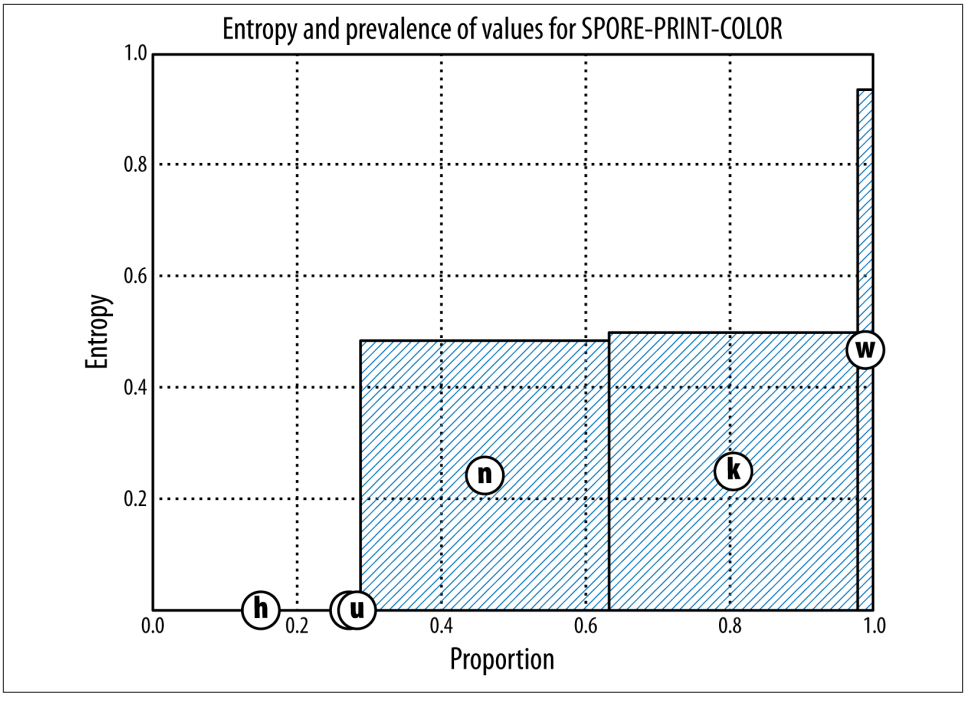


Figure 3-8. Entropy chart for the Mushroom dataset as split by SPORE-PRINT-COLOR. The amount of shading corresponds to the total (weighted sum) entropy, with each bar corresponding to the entropy of one of the attribute's values, and the width of the bar corresponding to the prevalence of that value in the data.

Similarly, Figure 3-8 shows how SPORE-PRINT-COLOR decreases uncertainty (entropy). A few of the values, such as h (`chocolate`), specify the target value perfectly and thus produce zero-entropy bars. But notice that they don't account for very much of the population, only about 30%.

Figure 3-9 shows the graph produced by ODOR. Many of the values, such as a (`almond`), c (`creosote`), and m (`musty`) produce zero-entropy partitions; only n (`no odor`) has a considerable entropy (about 20%). In fact, ODOR has the highest information gain of any attribute in the Mushroom dataset. It can reduce the dataset's total entropy to about 0.1, which gives it an information gain of 0.96 – 0.1 = 0.86. What is this saying? Many odors are completely characteristic of poisonous or edible mushrooms, so odor is a very informative attribute to check when considering mushroom edibility.[5] If you're

5. This assumes odor can be measured accurately, of course. If your sense of smell is poor you may not want to bet your life on it. Frankly, you probably wouldn't want to bet your life on the results of mining data from a field guide. Nevertheless, it makes a nice example.

going to build a model to determine the mushroom edibility using only a *single* feature, you should choose its odor. If you were going to build a more complex model you might start with the attribute ODOR before considering adding others. In fact, this is exactly the topic of the next section.

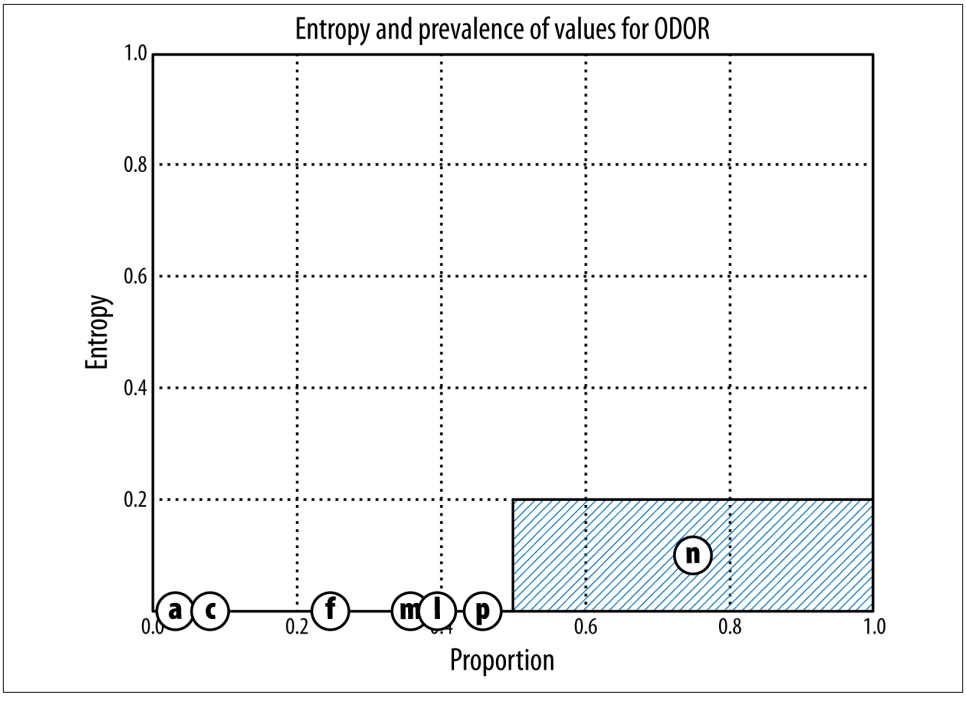


Figure 3-9. Entropy chart for the Mushroom dataset as split by ODOR. The amount of shading corresponds to the total (weighted sum) entropy, with each bar corresponding to the entropy of one of the attribute's values, and the width of the bar corresponding to the prevalence of that value in the data.

Supervised Segmentation with Tree-Structured Models

We have now introduced one of the fundamental ideas of data mining: finding informative attributes from the data. Let's continue on the topic of creating a supervised segmentation, because as important as it is, attribute selection alone does not seem to be sufficient. If we select the single variable that gives the most information gain, we create a very simple segmentation. If we select multiple attributes each giving some information gain, it's not clear how to put them together. Recall from earlier that we would like to create segments that use multiple attributes, such as "Middle-aged professionals who reside in New York City on average have a churn rate of 5%." We now

introduce an elegant application of the ideas we've developed for selecting important attributes, to produce a multivariate (multiple attribute) supervised segmentation.

Consider a segmentation of the data to take the form of a "tree," such as that shown in Figure 3-10. In the figure, the tree is upside down with the root at the top. The tree is made up of *nodes*, interior nodes and terminal nodes, and branches emanating from the interior nodes. Each interior node in the tree contains a test of an attribute, with each branch from the node representing a distinct value, or range of values, of the attribute. Following the branches from the root node down (in the direction of the arrows), each path eventually terminates at a terminal node, or *leaf*. The tree creates a segmentation of the data: every data point will correspond to one and only one path in the tree, and thereby to one and only one leaf. In other words, each leaf corresponds to a segment, and the attributes and values along the path give the characteristics of the segment. So the rightmost path in the tree in Figure 3-10 corresponds to the segment "Older, unemployed people with high balances." The tree is a *supervised* segmentation, because each leaf contains a value for the target variable. Since we are talking about classification, here each leaf contains a classification for its segment. Such a tree is called a *classification tree* or more loosely a *decision tree*.

Classification trees often are used as predictive models—"tree structured models." In use, when presented with an example for which we do not know its classification, we can predict its classification by finding the corresponding segment and using the class value at the leaf. Mechanically, one would start at the root node and descend through the interior nodes, choosing branches based on the specific attribute values in the example. The nonleaf nodes are often referred to as "decision nodes," because when descending through the tree, at each node one uses the values of the attribute to make a decision about which branch to follow. Following these branches ultimately leads to a final decision about what class to predict: eventually a terminal node is reached, which gives a class prediction. In a tree, no two parents share descendants and there are no cycles; the branches always "point downwards" so that every example always ends up at a leaf node with some specific class determination.

Consider how we would use the classification tree in Figure 3-10 to classify an example of the person named Claudio from Figure 3-1. The values of Claudio's attributes are *Balance=115K*, *Employed=No*, and *Age=40*. We begin at the root node that tests *Employed*. Since the value is *No* we take the right branch. The next test is *Balance*. The value of *Balance* is 115K, which is greater than 50K so we take a right branch again to a node that tests *Age*. The value is 40 so we take the left branch. This brings us to a leaf node specifying *class=Not Write-off*, representing a prediction that Claudio will not default. Another way of saying this is that we have classified Claudio into a segment defined by *(Employed=No, Balance=115K, Age<45)* whose classification is *Not Write -off*.

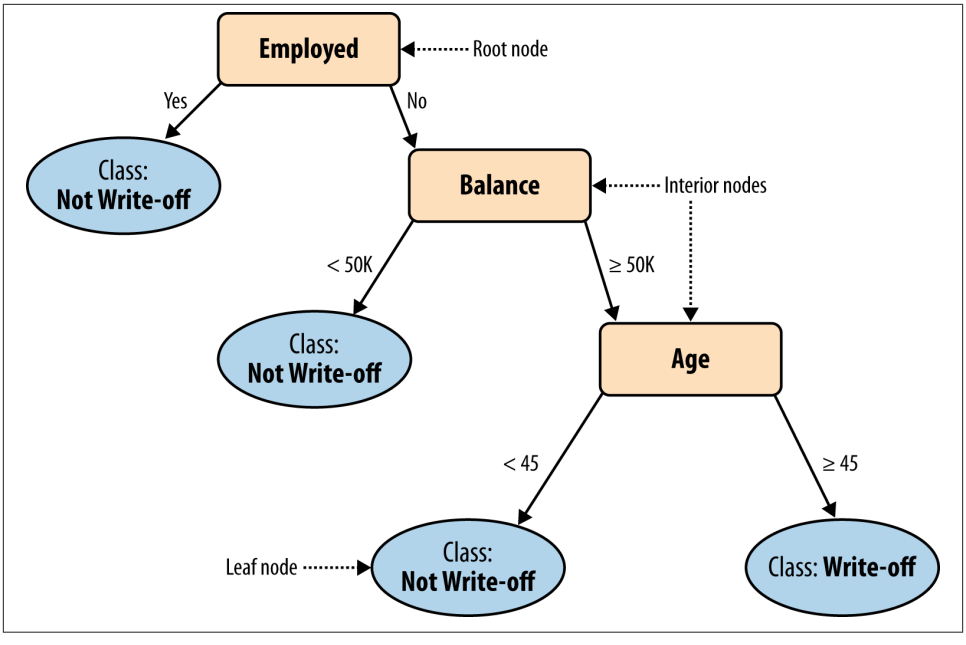


Figure 3-10. A simple classification tree.

Classification trees are one sort of tree-structured model. As we will see later, in business applications often we want to predict the probability of membership in the class (e.g., the probability of churn or the probability of write-off), rather than the class itself. In this case, the leaves of the *probability estimation tree* would contain these probabilities rather than a simple value. If the target variable is numeric, the leaves of the *regression tree* contain numeric values. However, the basic idea is the same for all.

Trees provide a model that can represent exactly the sort of supervised segmentation we often want, and we know how to use such a model to predict values for new cases (in "use"). However, we still have not addressed how to create such a model from the data. We turn to that now.

There are many techniques to induce a supervised segmentation from a dataset. One of the most popular is to create a tree-structured model (*tree induction*). These techniques are popular because tree models are easy to understand, and because the induction procedures are elegant (simple to describe) and easy to use. They are robust to many common data problems and are relatively efficient. Most data mining packages include some type of tree induction technique.

How do we create a classification tree from data? Combining the ideas introduced above, the goal of the tree is to provide a supervised segmentation—more specifically, to partition the instances, based on their attributes, into subgroups that have similar values

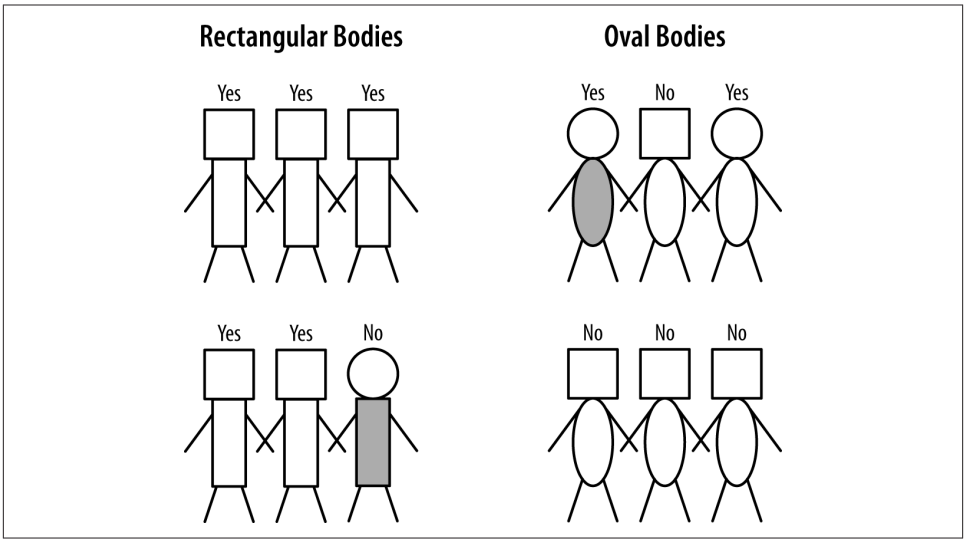


Figure 3-11. First partitioning: splitting on body shape (rectangular versus oval).

for their target variables. We would like for each "leaf" segment to contain instances that tend to belong to the same class.

To illustrate the process of classification tree induction, consider the very simple example set shown previously in Figure 3-2.

Tree induction takes a divide-and-conquer approach, starting with the whole dataset and applying variable selection to try to create the "purest" subgroups possible using the attributes. In the example, one way is to separate people based on their body type: rectangular versus oval. This creates the two groups shown in Figure 3-11. How good is this partitioning? The rectangular-body people on the left are mostly *Yes*, with a single *No* person, so it is mostly pure. The oval-body group on the right has mostly *No* people, but two *Yes* people. This step is simply a direct application of the attribute selection ideas presented above. Let's consider this "split" to be the one that yields the largest information gain.

Looking at Figure 3-11, we can now see the elegance of tree induction, and why it resonates well with so many people. The left and right subgroups are simply smaller versions of the problem with which we initially were faced! We can simply take each data subset and *recursively* apply attribute selection to find the best attribute to partition it. So in our example, we recursively consider the oval-body group (Figure 3-12). To split this group again we now consider another attribute: head shape. This splits the group in two on the right side of the figure. How good is this partitioning? Each new group has a single target label: four (square heads) of *No*, and two (round heads) of

Yes. These groups are "maximally pure" with respect to class labels and there is no need to split them further.

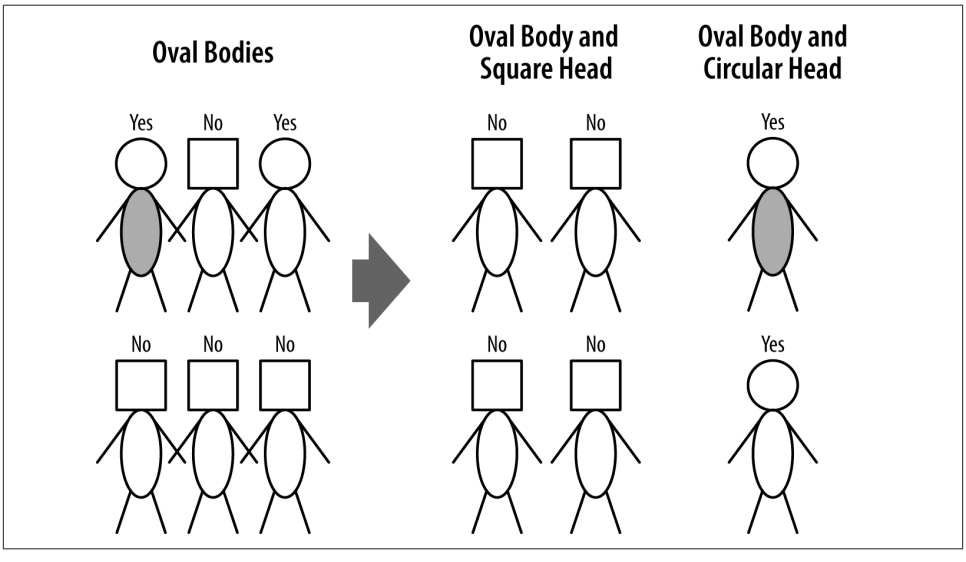


Figure 3-12. Second partitioning: the oval body people sub-grouped by head type.

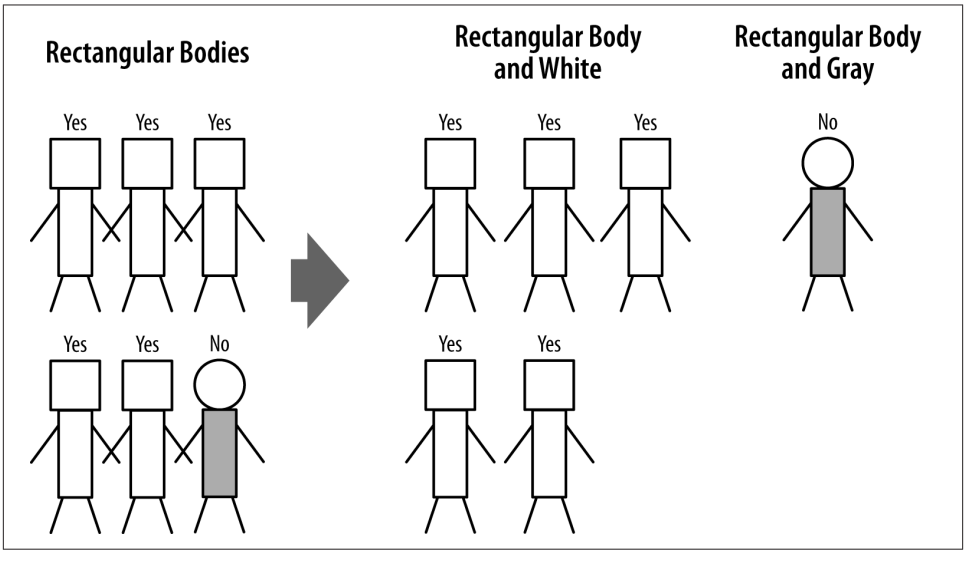


Figure 3-13. Third partitioning: the rectangular body people subgrouped by body color.

We still have not done anything with the rectangular body group on the left side of Figure 3-11, so let's consider how to split them. There are five *Yes* people and one *No* person. There are two attributes we could split upon: head shape (square or round), and body color (white or gray). Either of these would work, so we arbitrarily choose body color. This produces the groupings in Figure 3-13. These are pure groups (all of one type) so we are finished. The classification tree corresponding to these groupings is shown in Figure 3-14.

In summary, the procedure of classification tree induction is a recursive process of divide and conquer, where the goal at each step is to select an attribute to partition the current group into subgroups that are as pure as possible with respect to the target variable. We perform this partitioning recursively, splitting further and further until we are done. We choose the attributes to split upon by testing all of them and selecting whichever yields the purest subgroups. When are we done? (In other words, when do we stop recursing?) It should be clear that we would stop when the nodes are pure, or when we run out of variables to split on. But we may want to stop earlier; we will return to this question in Chapter 5.

Visualizing Segmentations

Continuing with the metaphor of predictive model building as supervised segmentation, it is instructive to visualize exactly how a classification tree partitions the instance space. The instance space is simply the space described by the data features. A common form of instance space visualization is a scatterplot on some pair of features, used to compare one variable against another to detect correlations and relationships.

Though data may contain dozens or hundreds of variables, it is only really possible to visualize segmentations in two or three dimensions at once. Still, visualizing models in instance space in a few dimensions is useful for understanding the different *types* of models because it provides insights that apply to higher dimensional spaces as well. It may be difficult to compare very different families of models just by examining their form (e.g., a mathematical formula versus a set of rules) or the algorithms that generate them. Often it is easier to compare them based on how they partition the instance space.

For example, Figure 3-15 shows a simple classification tree next to a two-dimensional graph of the instance space: Balance on the *x* axis and Age on the *y* axis. The root node of the classification tree tests Balance against a threshold of 50K. In the graph, this corresponds to a vertical line at 50K on the *x* axis splitting the plane into Balance<50K and Balance≥50K. At the left of this line lie the instances whose Balance values are less than 50K; there are 13 examples of class Write-off (black dot) and 2 examples of class non-Write-off (plus sign) in this region.

On the right branch out of the root node are instances with Balance≥50K. The next node in the classification tree tests the Age attribute against the threshold 45. In the

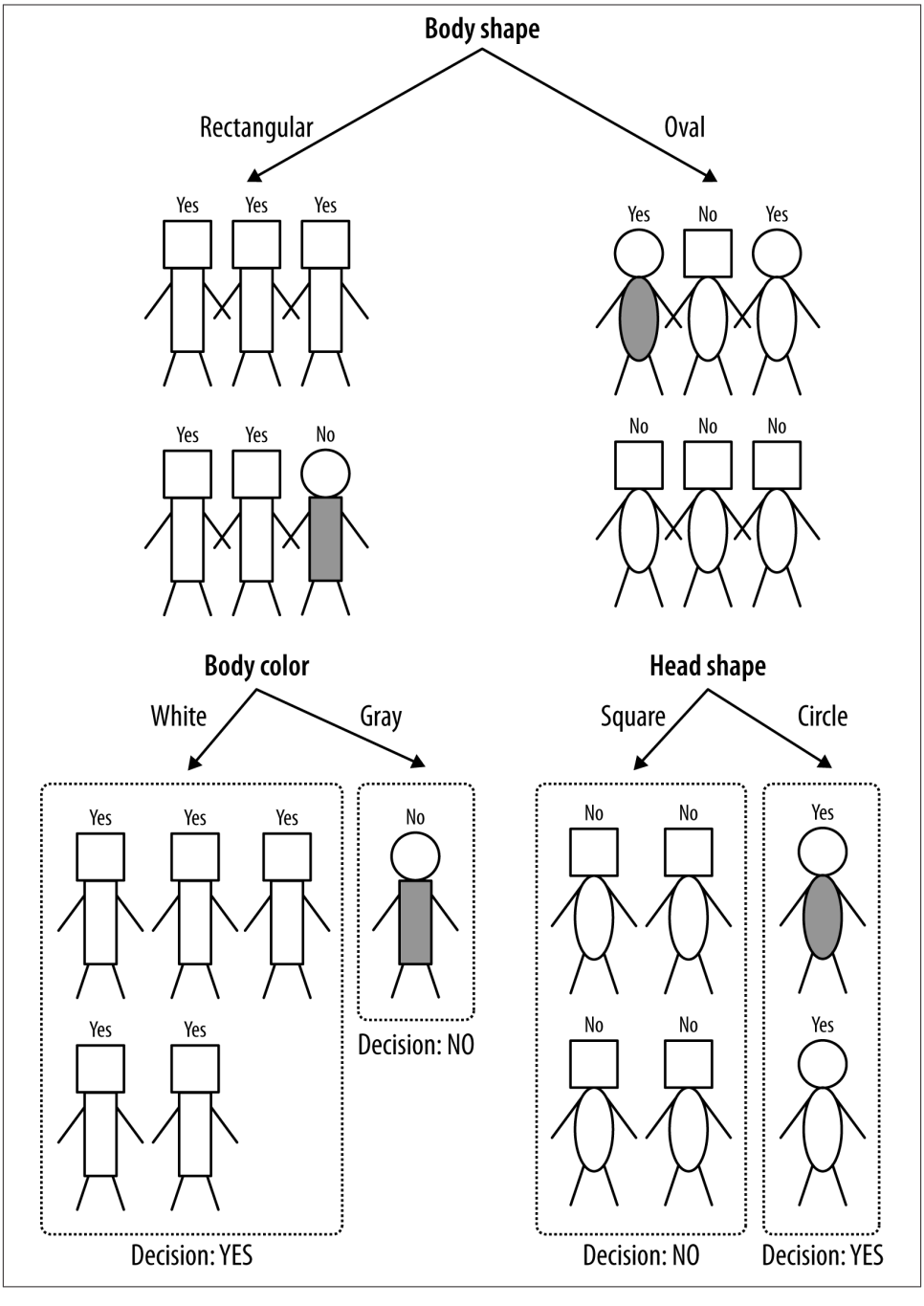


Figure 3-14. The classification tree resulting from the splits done in Figure 3-11 to Figure 3-13.

graph this corresponds to the horizontal dotted line at Age=45. It appears only on the right side of the graph because this partition only applies to examples with Balance≥50. The Age decision node assigns to its left branch instances with Age<45, corresponding to the lower right segment of the graph, representing: (Balance≥50K AND Age<45).

Notice that each internal (decision) node corresponds to a split of the instance space. Each leaf node corresponds to an unsplit region of the space (a segment of the population). Whenever we follow a path in the tree out of a decision node we are restricting attention to one of the two (or more) subregions defined by the split. As we descend through a classification tree we consider progressively more focused subregions of the instance space.

Decision lines and hyperplanes

The lines separating the regions are known as *decision lines* (in two dimensions) or more generally *decision surfaces* or *decision boundaries*. Each node of a classification tree tests a single variable against a fixed value so the decision boundary corresponding to it will always be perpendicular to the axis representing this variable. In two dimensions, the line will be either horizontal or vertical. If the data had three variables the instance space would be three-dimensional and each boundary surface imposed by a classification tree would be a two-dimensional plane. In higher dimensions, since each node of a classification tree tests one variable it may be thought of as "fixing" that one dimension of a decision boundary; therefore, for a problem of n variables, each node of a classification tree imposes an $(n–1)$-dimensional "hyperplane" decision boundary on the instance space.

You will often see the term *hyperplane* used in data mining literature to refer to the general separating surface, whatever it may be. Don't be intimidated by this terminology. You can always just think of it as a generalization of a line or a plane.

Other decision surfaces are possible, as we shall see later.

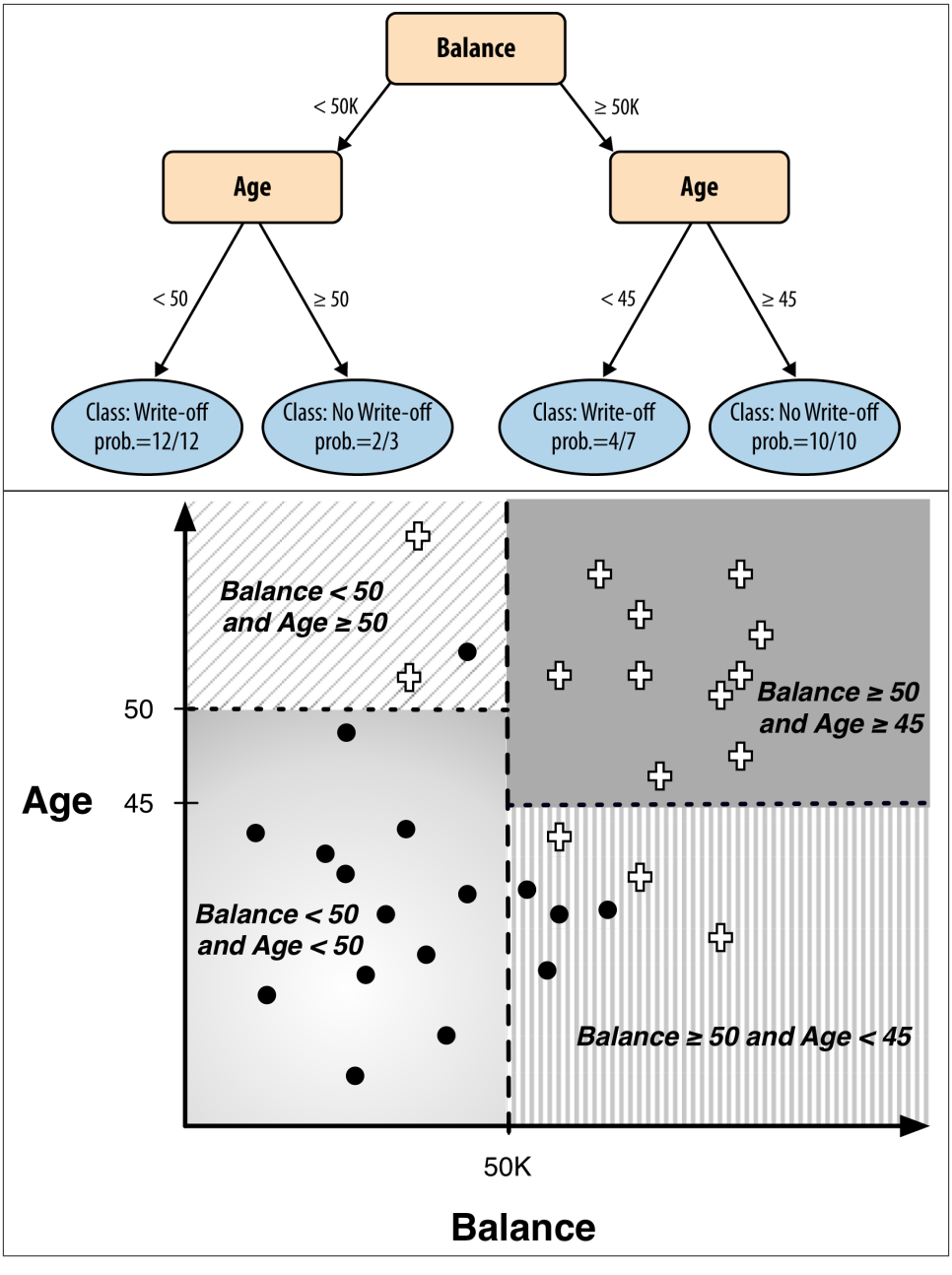


Figure 3-15. A classification tree and the partitions it imposes in instance space. The black dots correspond to instances of the class Write-off, the plus signs correspond to instances of class non-Write-off. The shading shows how the tree leaves correspond to segments of the population in instance space.

Trees as Sets of Rules

Before moving on from the interpretation of classification trees, we should mention their interpretation as logical statements. Consider again the tree shown at the top of Figure 3-15. You classify a new unseen instance by starting at the root node and following the attribute tests downward until you reach a leaf node, which specifies the instance's predicted class. If we trace down a single path from the root node to a leaf, collecting the conditions as we go, we generate a rule. Each rule consists of the attribute tests along the path connected with AND. Starting at the root node and choosing the left branches of the tree, we get the rule:

```
IF (Balance < 50K) AND (Age < 50) THEN Class=Write-off
```

We can do this for every possible path to a leaf node. From this tree we get three more rules:

```
IF (Balance < 50K) AND (Age ≥ 50) THEN Class=No Write-off
IF (Balance ≥ 50K) AND (Age < 45) THEN Class=Write-off
IF (Balance ≥ 50K) AND (Age ≥ 45) THEN Class=No Write-off
```

The classification tree is equivalent to this rule set. If these rules look repetitive, that's because they are: the tree gathers common rule prefixes together toward the top of the tree. Every classification tree can be expressed as a set of rules this way. Whether the tree or the rule set is more intelligible is a matter of opinion; in this simple example, both are fairly easy to understand. As the model becomes larger, some people will prefer the tree or the rule set.

Probability Estimation

In many decision-making problems, we would like a more informative prediction than just a classification. For example, in our churn-prediction problem, rather than simply predicting whether a person will leave the company within 90 days of contract expiration, we would much rather have an estimate of the probability that he will leave the company within that time. Such estimates can be used for many purposes. We will discuss some of these in detail in later chapters, but briefly: you might then rank prospects by their probability of leaving, and then allocate a limited incentive budget to the highest probability instances. Alternatively, you may want to allocate your incentive budget to the instances with the highest expected loss, for which you'll need (an estimate of) the probability of churn. Once you have such probability estimates you can use them in a more sophisticated decision-making process than these simple examples, as we'll describe in later chapters.

There is another, even more insidious problem with models that give simple classifications, rather than estimates of class membership probability. Consider the problem of estimating credit default. Under normal circumstances, for just about any segment of the population to which we would be considering giving credit, the probability of write-off will be very small—far less than 0.5. In this case, when we build a model to estimate the classification (write-off or not), we'd have to say that for each segment, the members are likely not to default—and they will all get the same classification (not write-off). For example, in a naively built tree model every leaf will be labeled "not write-off." This turns out to be a frustrating experience for new data miners: after all that work, the model really just says that no one is likely to default? This does *not* mean that the model is useless. It may be that the different segments indeed have very different probabilities of write-off, they just all are less than 0.5. If instead we use these probabilities for assigning credit, we may be able reduce our risk substantially.

So, in the context of supervised segmentation, we would like each segment (leaf of a tree model) to be assigned an estimate of the probability of membership in the different classes. Figure 3-15 more generally shows a "probability estimation tree" model for our simple write-off prediction example, giving not only a prediction of the class but also the estimate of the probability of membership in the class.[6]

Fortunately, the tree induction ideas we have discussed so far can easily produce probability estimation trees instead of simple classification trees.[7] Recall that the tree induction procedure subdivides the instance space into regions of class purity (low entropy). If we are satisfied to assign the same class probability to every member of the segment corresponding to a tree leaf, we can use instance counts at each leaf to compute a class probability estimate. For example, if a leaf contains n positive instances and m negative instances, the probability of any new instance being positive may be estimated as $n/(n+m)$. This is called a *frequency-based* estimate of class membership probability.

At this point you may spot a problem with estimating class membership probabilities this way: we may be overly optimistic about the probability of class membership for segments with very small numbers of instances. At the extreme, if a leaf happens to have only a single instance, should we be willing to say that there is a 100% probability that members of that segment will have the class that this one instance happens to have?

6. We often deal with binary classification problems, such as write-off or not, or churn or not. In these cases it is typical just to report the probability of membership in one chosen class $p(c)$, because the other is just $1 - p(c)$.

7. Often these are still called classification trees, even if the decision maker intends to use the probability estimates rather than the simple classifications.

This phenomenon is one example of a fundamental issue in data science ("overfitting"), to which we devote a chapter later in the book. For completeness, let's quickly discuss one easy way to address this problem of small samples for tree-based class probability estimation. Instead of simply computing the frequency, we would often use a "smoothed" version of the frequency-based estimate, known as the Laplace correction, the purpose of which is to moderate the influence of leaves with only a few instances. The equation for binary class probability estimation becomes:

$$p(c) = \frac{n + 1}{n + m + 2}$$

where n is the number of examples in the leaf belonging to class c, and m is the number of examples not belonging to class c.

Let's walk through an example with and without the Laplace correction. A leaf node with two positive instances and no negative instances would produce the same frequency-based estimate ($p = 1$) as a leaf node with 20 positive instances and no negatives. However, the first leaf node has much less evidence and may be extreme only due to there being so few instances. Its estimate should be tempered by this consideration. The Laplace equation smooths its estimate down to $p = 0.75$ to reflect this uncertainty; the Laplace correction has much less effect on the leaf with 20 instances ($p \approx 0.95$). As the number of instances increases, the Laplace equation converges to the frequency-based estimate. Figure 3-16 shows the effect of Laplace correction on several class ratios as the number of instances increases (2/3, 4/5, and 1/1). For each ratio the solid horizontal line shows the uncorrected (constant) estimate, while the corresponding dashed line shows the estimate with the Laplace correction applied. The uncorrected line is the asymptote of the Laplace correction as the number of instances goes to infinity.

Example: Addressing the Churn Problem with Tree Induction

Now that we have a basic data mining technique for predictive modeling, let's consider the churn problem again. How could we use tree induction to help solve it?

For this example, we have a historical data set of 20,000 customers. At the point of collecting the data, each customer either had stayed with the company or had left (churned). Each customer is described by the variables listed in Table 3-2.

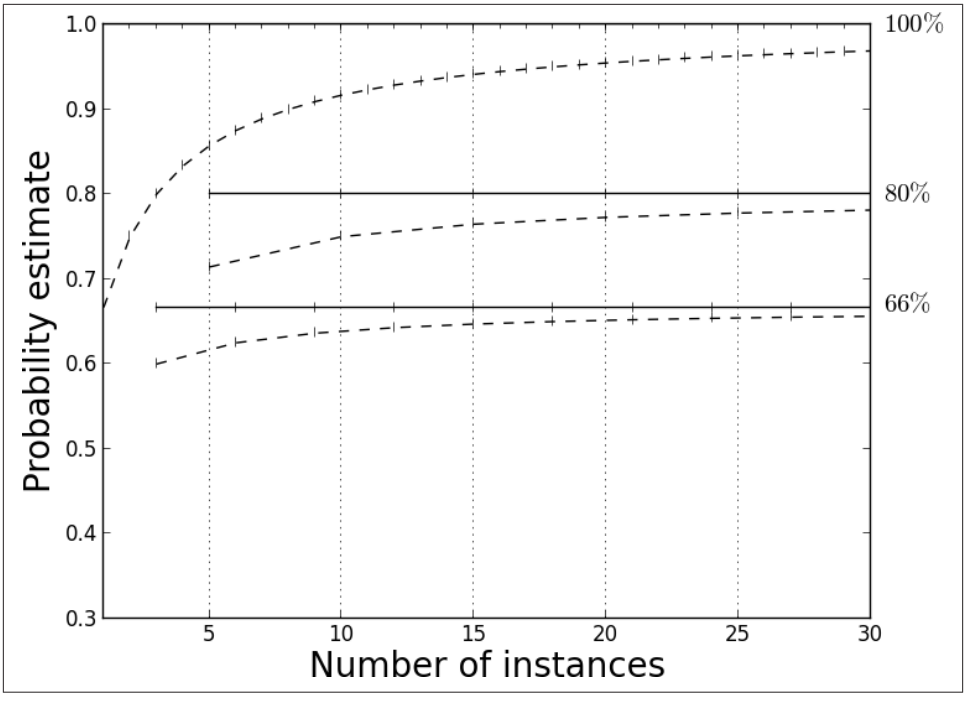


Figure 3-16. The effect of Laplace smoothing on probability estimation for several instance ratios.

Table 3-2. Attributes for the cellular phone churn-prediction problem

Variable	Explanation
COLLEGE	Is the customer college educated?
INCOME	Annual income
OVERAGE	Average overcharges per month
LEFTOVER	Average number of leftover minutes per month
HOUSE	Estimated value of dwelling (from census tract)
HANDSET_PRICE	Cost of phone
LONG_CALLS_PER_MONTH	Average number of long calls (15 mins or over) per month
AVERAGE_CALL_DURATION	Average duration of a call
REPORTED_SATISFACTION	Reported level of satisfaction
REPORTED_USAGE_LEVEL	Self-reported usage level
LEAVE *(Target variable)*	Did the customer stay or leave (churn)?

These variables comprise basic demographic and usage information available from the customer's application and account. We want to use these data with our tree induction technique to predict which new customers are going to churn.

Before starting to build a classification tree with these variables, it is worth asking, *How good are each of these variables individually?* For this we measure the information gain of each attribute, as discussed earlier. Specifically, we apply Equation 3-2 to each variable independently over the entire set of instances, to see what each gains us.

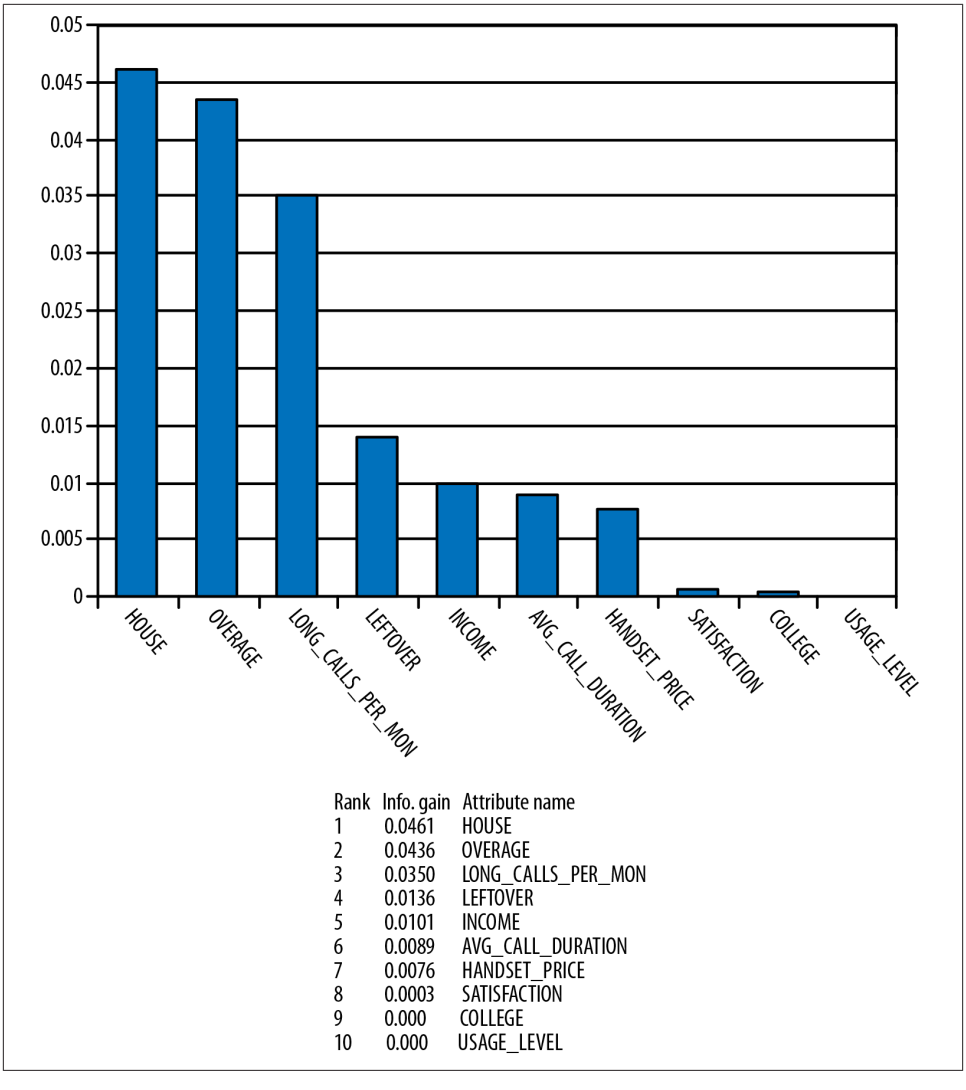
The results are in Figure 3-17, with a table listing the exact values. As you can see, the first three variables—the house value, the number of leftover minutes, and the number of long calls per month—have a higher information gain than the rest.[8] Perhaps surprisingly, neither the amount the phone is used nor the reported degree of satisfaction seems, in and of itself, to be very predictive of churning.

Applying a classification tree algorithm to the data, we get the tree shown in Figure 3-18. The highest information gain feature (HOUSE) according to Figure 3-17 is at the root of the tree. This is to be expected since it will always be chosen first. The second best feature, OVERAGE, also appears high in the tree. However, the order in which features are chosen for the tree doesn't exactly correspond to their ranking in Figure 3-17. Why is this?

The answer is that the table ranks each feature by how good it is *independently*, evaluated separately on the entire population of instances. Nodes in a classification tree depend on the instances above them in the tree. Therefore, except for the root node, features in a classification tree are not evaluated on the entire set of instances. The information gain of a feature depends on the set of instances against which it is evaluated, so the ranking of features for some internal node may not be the same as the global ranking.

We have not yet discussed how we decide to stop building the tree. The dataset has 20,000 examples yet the tree clearly doesn't have 20,000 leaf nodes. Can't we just keep selecting more attributes to split upon, building the tree downwards until we've exhausted the data? The answer is yes, we can, but we should stop long before the model becomes that complex. This issue ties in closely with model generality and overfitting, whose discussion we defer to Chapter 5.

8. Note that the information gains for the attributes in this churn data set are much smaller than those shown previously for the mushroom data set.



Rank	Info. gain	Attribute name
1	0.0461	HOUSE
2	0.0436	OVERAGE
3	0.0350	LONG_CALLS_PER_MON
4	0.0136	LEFTOVER
5	0.0101	INCOME
6	0.0089	AVG_CALL_DURATION
7	0.0076	HANDSET_PRICE
8	0.0003	SATISFACTION
9	0.000	COLLEGE
10	0.000	USAGE_LEVEL

Figure 3-17. Churn attributes from Table 3-2 ranked by information gain.

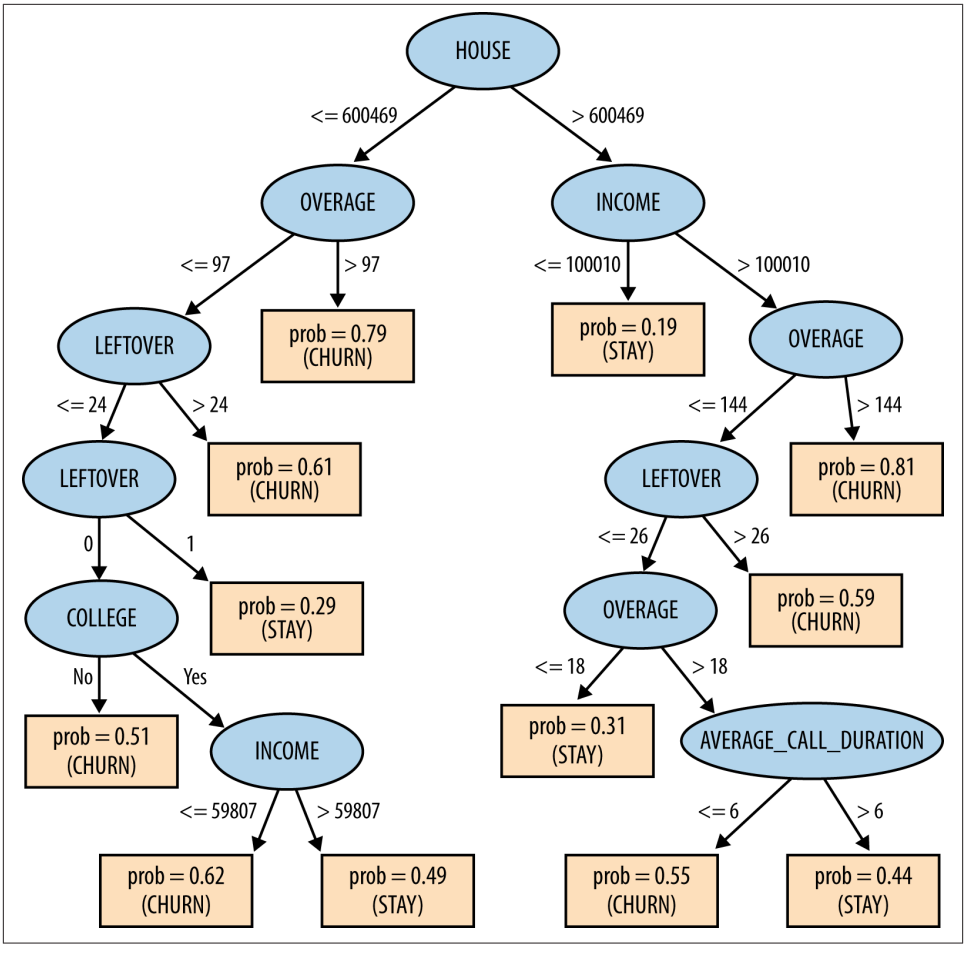


Figure 3-18. Classification tree learned from the cellular phone churn data. Rectangular leaves correspond to segments of the population, defined by the path from the root at the top. Probabilities at the leaves are the estimated probabilities of churning for the corresponding segment; in parentheses are shown the classifications resulting from applying a decision threshold of 0.5 to the probabilities (i.e., are the individuals in the segment more likely to CHURN or to STAY?).

Consider a final issue with this dataset. After building a tree model from the data, we measured its accuracy against the data to see how good of a model it is. Specifically, we used a training set consisting half of people who churned and the other half who did not; after learning a classification tree from this, we applied the tree to the dataset to see how many of the examples it could classify correctly. The tree achieved 73% accuracy on its decisions. This raises two questions:

1. First, do you trust this number? If we applied the tree to another sample of 20,000 people from the same dataset, do you think we'd still get about 73% accuracy?
2. If you *do* trust the number, does it mean this model is good? In other words, is a model with 73% accuracy worth using?

We will revisit these questions in Chapter 7 and Chapter 8, which delve into issues of model evaluation.

Summary

In this chapter, we introduced basic concepts of predictive modeling, one of the main tasks of data science, in which a model is built that can estimate the value of a target variable for a new unseen example. In the process, we introduced one of data science's fundamental notions: finding and selecting informative attributes. Selecting informative attributes can be a useful data mining procedure in and of itself. Given a large collection of data, we now can find those variables that correlate with or give us information about another variable of interest. For example, if we gather historical data on which customers have or have not left the company (churned) shortly after their contracts expire, attribute selection can find demographic or account-oriented variables that provide information about the likelihood of customers churning. One basic measure of attribute information is called *information gain*, which is based on a purity measure called *entropy*; another is variance reduction.

Selecting informative attributes forms the basis of a common modeling technique called tree induction. Tree induction recursively finds informative attributes for subsets of the data. In so doing it segments the space of instances into similar regions. The partitioning is "supervised" in that it tries to find segments that give increasingly precise information about the quantity to be predicted, the target. The resulting tree-structured model partitions the space of all possible instances into a set of segments with different predicted values for the target. For example, when the target is a binary "class" variable such as churn versus not churn, or write-off versus not write-off, each leaf of the tree corresponds to a population segment with a different estimated probability of class membership.

As an exercise, think about what would be different in building a tree-structured model for regression rather than for classification. What would need to be changed from what you've learned about classification tree induction?

Historically, tree induction has been a very popular data mining procedure because it is easy to understand, easy to implement, and computationally inexpensive. Research on tree induction goes back at least to the 1950s and 1960s. Some of the earliest popular

tree induction systems include CHAID (Chi-squared Automatic Interaction Detection) (Kass, 1980) and CART (Classification and Regression Trees) (Breiman, Friedman, Olshen, & Stone, 1984), which are still widely used. C4.5 and C5.0 are also very popular tree induction algorithms, which have a notable lineage (Quinlan, 1986, 1993). J48 is a reimplementation of C4.5 in the Weka package (Witten & Frank, 2000; Hall et al., 2001).

In practice, tree-structured models work remarkably well, though they may not be the most accurate model one can produce from a particular data set. In many cases, especially early in the application of data mining, it is important that models be understood and explained easily. This can be useful not just for the data science team but for communicating results to stakeholders not knowledgeable about data mining.

CHAPTER 4
Fitting a Model to Data

Fundamental concepts: *Finding "optimal" model parameters based on data; Choosing the goal for data mining; Objective functions; Loss functions.*

Exemplary techniques: *Linear regression; Logistic regression; Support-vector machines.*

As we have seen, predictive modeling involves finding a model of the target variable in terms of other descriptive attributes. In Chapter 3, we constructed a supervised segmentation model by recursively finding informative attributes on ever-more-precise subsets of the set of all instances, or from the geometric perspective, ever-more-precise subregions of the instance space. From the data we produced both the structure of the model (the particular tree model that resulted from the tree induction) and the numeric "parameters" of the model (the probability estimates at the leaf nodes).

An alternative method for learning a predictive model from a dataset is to start by specifying the structure of the model with certain numeric parameters left unspecified. Then the data mining calculates the best parameter values given a particular set of training data. A very common case is where the structure of the model is a parameterized mathematical function or equation of a set of numeric attributes. The attributes used in the model could be chosen based on domain knowledge regarding which attributes ought to be informative in predicting the target variable, or they could be chosen based on other data mining techniques, such as the attribute selection procedures introduced in Chapter 3. The data miner specifies the form of the model and the attributes; the goal of the data mining is to tune the parameters so that the model fits the data as well as possible. This general approach is called *parameter learning* or *parametric modeling*.

In certain fields of statistics and econometrics, the bare model with unspecified parameters is called "the model." We will clarify that this is the structure of the model, which still needs to have its parameters specified to be useful.