CHAPTER ONE

Introduction

IMAGINE YOUR BOSS COMES TO YOU AND SAYS: "HERE ARE 50 GB OF LOGFILES—FIND A WAY TO IMPROVE OUR business!"

What would you do? Where would you start? And what would you do next?

It's this kind of situation that the present book wants to help you with!

Data Analysis

Businesses sit on data, and every second that passes, they generate some more. Surely, there *must* be a way to make use of all this stuff. But how, exactly—that's far from clear.

The task is difficult because it is so vague: there is no specific problem that needs to be solved. There is no specific question that needs to be answered. All you know is the overall *purpose*: improve the business. And all you have is "the data." Where do you start?

You start with the only thing you have: "the data." What is it? We don't know! Although 50 GB sure sounds like a lot, we have no idea what it actually contains. The first thing, therefore, is to *take a look*.

And I mean this literally: the first thing to do is to *look* at the data by plotting it in different ways and looking at graphs. Looking at data, you will notice things—the way data points are distributed, or the manner in which one quantity varies with another, or the large number of outliers, or the total absence of them. . . . I don't know what you will find, but there is no doubt: if you look at data, you will observe things!

These observations should lead to some reflection. "Ten percent of our customers drive ninety percent of our revenue." "Whenever our sales volume doubles, the number of

returns goes up by a factor of four." "Every seven days we have a production run that has twice the usual defect rate, and it's always on a Thursday." How very interesting!

Now you've got something to work with: the amorphous mass of "data" has turned into ideas! To make these ideas concrete and suitable for further work, it is often useful to capture them in a mathematical form: a *model*. A model (the way I use the term) is a mathematical description of the system under study. A model is more than just a description of the data—it also incorporates your understanding of the process or the system that produced the data. A model therefore has *predictive power*: you can predict (with some certainty) that next Thursday the defect rate will be high *again*.

It's at this point that you may want to go back and alert the boss of your findings: "Next Thursday, watch out for defects!"

Sometimes, you may already be finished at this point: you found out enough to help improve the business. At other times, however, you may need to work a little harder. Some data sets do not yield easily to visual inspection—especially if you are dealing with data sets consisting of many different quantities, all of which seem equally important. In such cases, you may need to employ more-sophisticated methods to develop enough intuition before being able to formulate a relevant model. Or you may have been able to set up a model, but it is too complicated to understand its implications, so that you want to implement the model as a computer program and simulate its results. Such computationally intensive methods are occasionally useful, but they always come later in the game. You should only move on to them after having tried all the simple things first. And you will need the insights gained from those earlier investigations as input to the more elaborate approaches.

And finally, we need to come back to the initial agenda. To "improve the business" it is necessary to feed our understanding back into the organization—for instance, in the form of a business plan, or through a "metrics dashboard" or similar program.

What's in This Book

The program just described reflects the outline of this book.

We begin in Part I with a series of chapters on graphical techniques, starting in Chapter 2 with simple data sets consisting of only a single variable (or considering only a single variable at a time), then moving on in Chapter 3 to data sets of two variables. In Chapter 4 we treat the particularly important special case of a quantity changing over time, a so-called time series. Finally, in Chapter 5, we discuss data sets comprising more than two variables and some special techniques suitable for such data sets.

In Part II, we discuss models as a way to not only describe data but also to capture the understanding that we gained from graphical explorations. We begin in Chapter 7 with a discussion of order-of-magnitude estimation and uncertainty considerations. This may

seem odd but is, in fact, crucial: all models are approximate, so we need to develop a sense for the accuracy of the approximations that we use. In Chapters 8 and 9 we introduce basic building blocks that are useful when developing models.

Chapter 10 is a detour. For too many people, "data analysis" is synonymous with "statistics," and "statistics" is usually equated with a class in college that made *no sense* at all. In this chapter, I want to explain what statistics really is, what all the mysterious concepts mean and how they hang together, and what statistics can (and cannot) do for us. It is intended as a travel guide should you ever want to read a statistics book in the future.

Part III discusses several computationally intensive methods, such as simulation and clustering in Chapters 12 and 13. Chapter 14 is, mathematically, the most challenging chapter in the book: it deals with methods that can help select the most relevant variables from a multivariate data set.

In Part IV we consider some ways that data may be used in a business environment. In Chapter 16 we talk about metrics, reporting, and dashboards—what is sometimes referred to as "business intelligence." In Chapter 17 we introduce some of the concepts required to make financial calculations and to prepare business plans. Finally, in chapter 18, we conclude with a survey of some methods from classification and predictive analytics.

At the end of each part of the book you will find an "Intermezzo." These intermezzos are not really part of the course; I use them to go off on some tangents, or to explain topics that often remain a bit hazy. You should see them as an opportunity to relax!

The appendices contain some helpful material that you may want to consult at various times as you go through the text. Appendix A surveys some of the available tools and programming environments for data manipulation and analysis. In Appendix B I have collected some basic mathematical results that I expect you to have at least passing familiarity with. I assume that you have seen this material at least once before, but in this appendix, I put it together in an application-oriented context, which is more suitable for our present purposes. Appendix C discusses some of the mundane tasks that—like it or not—make up a large part of actual data analysis and also introduces some data-related terminology.

What's with the Workshops?

Every full chapter (after this one) includes a section titled "Workshop" that contains some programming examples related to the chapter's material. I use these Workshops for two purposes. On the one hand, I'd like to introduce a number of open source tools and libraries that may be useful for the kind of work discussed in this book. On the other hand, some concepts (such as computational complexity and power-law distributions) must be seen to be believed: the Workshops are a way to demonstrate these issues and allow you to experiment with them yourself.

Among the tools and libraries is quite a bit of Python and R. Python has become somewhat the scripting language of choice for scientific applications, and R is the most popular open source package for statistical applications. *This choice is neither an endorsement nor a recommendation* but primarily a reflection of the current state of available software. (See Appendix A for a more detailed discussion of software for data analysis and related purposes.)

My goal with the tool-oriented Workshops is rather specific: I want to enable you to decide whether a given tool or library is worth spending time on. (I have found that evaluating open source offerings is a necessary but time-consuming task.) I try to demonstrate clearly what purpose each particular tool serves. Toward this end, I usually give one or two short, but not entirely trivial, examples and try to outline enough of the architecture of the tool or library to allow you to take it from there. (The documentation for many open source projects has a hard time making the bridge from the trivial, cut-and-paste "Hello, World" example to the reference documentation.)

What's with the Math?

This book contains a certain amount of mathematics. Depending on your personal predilection you may find this trivial, intimidating, or exciting.

The reality is that if you want to work *analytically*, you will need to develop some familiarity with a few mathematical concepts. There is simply no way around it. (You can work with *data* without any math skills—look at what any data modeler or database administrator does. But if you want to do any sort of *analysis*, then a little math becomes a necessity.)

I have tried to make the text accessible to readers with a minimum of previous knowledge. Some college math classes on calculus and similar topics are helpful, of course, but are by no means required. Some sections of the book treat material that is either more abstract or will likely be unreasonably hard to understand without some previous exposure. These sections are optional (they are not needed in the sequel) and are clearly marked as such.

A somewhat different issue concerns the notation. I use mathematical notation wherever it is appropriate and it helps the presentation. I have made sure to use only a very small set of symbols; check Appendix B if something looks unfamiliar.

Couldn't I have written all the mathematical expressions as computer code, using Python or some sort of pseudo-code? The answer is no, because quite a few *essential* mathematical concepts cannot be expressed in a finite, floating-point oriented machine (anything having to do with a limit process—or real numbers, in fact). But even if I could write all math as code, I don't think I should. Although I wholeheartedly agree that mathematical notation can get out of hand, simple formulas actually provide the easiest, most succinct way to express mathematical concepts.

Just compare. I'd argue that:

$$\sum_{k=0}^{n} \frac{c(k)}{(1+p)^k}$$

is clearer and easier to read than:

```
s = 0
for k in range( len(c) ):
    s += c[k]/(1+p)**k
```

and certainly easier than:

```
s = ( c / (1+p)**numpy.arange(1, len(c)+1) ).sum(axis=0)
```

But that's only part of the story. More importantly, the first version expresses a *concept*, whereas the second and third are merely specific prescriptions for how to perform a certain calculation. They are *recipes*, not ideas.

Consider this: the formula in the first line is a description of a sum—not a specific sum, but any sum of this form: it's the *idea* of this kind of sum. We can now ask how this abstract sum will behave under certain conditions—for instance, if we let the upper limit n go to infinity. What value does the sum have in this case? Is it finite? Can we determine it? You would not even be *able* to ask this question given the code versions. (Remember that I am not talking about an approximation, such as letting n get "very large." I really do mean: what happens if n goes all the way to infinity? What can we say about the sum?)

Some programming environments (like Haskell, for instance) are more at ease dealing with infinite data structures—but if you look closely, you will find that they do so by being (coarse) approximations to mathematical concepts and notations. And, of course, they still won't be able to evaluate such expressions! (All evaluations will only involve a finite number of steps.) But once you train your mind to think in those terms, you can evaluate them *in your mind* at will.

It may come as a surprise, but mathematics is *not* a method for calculating things. Mathematics is a theory of *ideas*, and ideas—not calculational prescriptions—are what I would like to convey in this text. (See the discussion at the end of Appendix B for more on this topic and for some suggested reading.)

If you feel uncomfortable or even repelled by the math in this book, I'd like to ask for just one thing: try! Give it a shot. Don't immediately give up. Any frustration you may experience at first is more likely due to lack of familiarity rather than to the difficulty of the material. I promise that none of the content is out of your reach.

But you have to let go of the conditioned knee-jerk reflex that "math is, like, *yuck*!"

What You'll Need

This book is written with programmers in mind. Although previous programming experience is by no means required, I assume that you are able to take an idea and

implement it in the programming language of your choice—in fact, I assume that this is your prime motivation for reading this book.

I don't expect you to have any particular mathematical background, although some previous familiarity with calculus is certainly helpful. You will need to be able to count, though!

But the most important prerequisite is not programming experience, not math skills, and certainly not knowledge of anything having to do with "statistics." The most important prerequisite is *curiosity*. If you aren't curious, then this book is not for you. If you get a new data set and you are not *itching* to see what's in it, I won't be able to help you.

What's Missing

This is a book about data analysis and modeling with an emphasis on applications in a business settings. It was written at a beginning-to-intermediate level and for a general technical audience.

Although I have tried to be reasonably comprehensive, I had to choose which subjects to include and which to leave out. I have tried to select topics that are useful and relevant in practice and that can safely be applied by a nonspecialist. A few topics were omitted because they did not fit within the book's overall structure, or because I did not feel sufficiently competent to present them.

Scientific data. This is not a book about scientific data analysis. When you are doing scientific research (however you wish to define "scientific"), you really need to have a solid background (and that probably means formal training) in the field that you are working in. A book such as this one on general data analysis cannot replace this.

Formal statistical analysis. A different form of data analysis exists in some particularly well-established fields. In these situations, the environment from which the data arises is fully understood (or at least believed to be understood), and the methods and models to be used are likewise accepted and well known. Typical examples include clinical trials as well as credit scoring. The purpose of an "analysis" in these cases is not to find out anything new, but rather to determine the model parameters with the highest degree of accuracy and precision for each newly generated set of data points. Since this is the kind of work where details matter, it should be left to specialists.

Network analysis. This is a topic of current interest about which I know nothing. (Sorry!) However, it does seem to me that its nature is quite different from most problems that are usually considered "data analysis": less statistical, more algorithmic in nature. But I don't know for sure.

Natural language processing and text mining. Natural language processing is a big topic all by itself, which has little overlap (neither in terms of techniques nor applications) with

the rest of the material presented here. It deserves its own treatment—and several books on this subject are available.

Big data. Arguably the most painful omission concerns everything having to do with *Big Data*. Big Data is a pretty new concept—I tend to think of it as relating to data sets that not merely don't fit into main memory, but that no longer fit comfortably on a single *disk*, requiring compute clusters and the respective software and algorithms (in practice, map/reduce running on Hadoop).

The rise of Big Data is a remarkable phenomenon. When this book was conceived (early 2009), Big Data was certainly on the horizon but was not necessarily considered mainstream yet. As this book goes to print (late 2010), it seems that for many people in the tech field, "data" has become nearly synonymous with "Big Data." That kind of development usually indicates a fad. The reality is that, in practice, many data sets are "small," and in particular many *relevant* data sets are small. (Some of the most important data sets in a commercial setting are those maintained by the finance department—and since they are kept in Excel, they *must* be small.)

Big Data is not necessarily "better." Applied carelessly, it can be a huge step backward. The amazing insight of classical statistics is that you don't need to examine every single member of a population to make a definitive statement about the whole: instead you can sample! It is also true that a carefully selected sample may lead to better results than a large, messy data set. Big Data makes it easy to forget the basics.

It is a little early to say anything definitive about Big Data, but the current trend strikes me as being something quite *different*: it is not just classical data analysis on a larger scale. The approach of classical data analysis and statistics is *inductive*. Given a part, make statements about the whole: from a sample, estimate parameters of the population; given an observation, develop a theory for the underlying system. In contrast, Big Data (at least as it is currently being used) seems primarily concerned with individual data points. Given that *this specific* user liked *this specific* movie, what other *specific* movie might he like? This is a very different question than asking which movies are most liked by what people in general!

Big Data will not replace general, inductive data analysis. It is not yet clear just where Big Data will deliver the greatest bang for the buck—but once the dust settles, somebody should definitely write a book about it!