Contents

Preface ix

I Introduction

1 About the Book and Supporting Material 3

1.1 What Do Data Mining, Machine Learning, and Knowledge Discovery Mean? 3
1.2 What Is This Book About? 5
1.3 An Incomplete Survey of the Relevant Literature 8
1.4 Introduction to the Python Language and the Git Code Management Tool 12
1.5 Description of Surveys and Data Sets Used in Examples 13
1.6 Plotting and Visualizing the Data in This Book 29
1.7 How to Efficiently Use This Book 35
References 37

2 Fast Computation on Massive Data Sets 41

2.1 Data Types and Data Management Systems 41
2.2 Analysis of Algorithmic Efficiency 42
2.3 Seven Types of Computational Problem 44
2.4 Eight Strategies for Speeding Things Up 45
2.5 Case Studies: Speedup Strategies in Practice 48
References 60

II Statistical Frameworks and Exploratory Data Analysis

3 Probability and Statistical Distributions 65

3.1 Brief Overview of Probability and Random Variables 66
3.2 Descriptive Statistics 73
3.3 Common Univariate Distribution Functions 80
3.4 The Central Limit Theorem 98
3.5 Bivariate and Multivariate Distribution Functions 100
3.6 Correlation Coefficients 108
3.7 Random Number Generation for Arbitrary Distributions 111
References 114

4 Classical Statistical Inference 115

4.1 Classical vs. Bayesian Statistical Inference 115
4.2 Maximum Likelihood Estimation (MLE) 116
4.3 The Goodness of Fit and Model Selection 123
4.4 ML Applied to Gaussian Mixtures: The Expectation Maximization Algorithm 126
4.5 Confidence Estimates: The Bootstrap and the Jackknife 132
4.6 Hypothesis Testing 135
4.7 Comparison of Distributions 141
4.8 Nonparametric Modeling and Histograms 153
4.9 Selection Effects and Luminosity Function Estimation 157
4.10 Summary 162
References 162

5 Bayesian Statistical Inference 165

5.1 Introduction to the Bayesian Method 166
5.2 Bayesian Priors 170
5.3 Bayesian Parameter Uncertainty Quantification 174
5.4 Bayesian Model Selection 175
5.5 Nonuniform Priors: Eddington, Malmquist, and Lutz–Kelker Biases 180
5.6 Simple Examples of Bayesian Analysis: Parameter Estimation 185
5.7 Simple Examples of Bayesian Analysis: Model Selection 211
5.8 Numerical Methods for Complex Problems (MCMC) 217
5.9 Hierarchical Bayesian Modeling 228
5.10 Approximate Bayesian Computation 232
5.11 Summary of Pros and Cons for Classical and Bayesian Methods 234
References 237

III Data Mining and Machine Learning

6 Searching for Structure in Point Data 243

6.1 Nonparametric Density Estimation 244
6.2 Nearest-Neighbor Density Estimation 251
6.3 Parametric Density Estimation 253
6.4 Finding Clusters in Data 263
6.5 Correlation Functions 269
6.6 Which Density Estimation and Clustering Algorithms Should I Use? 273
References 277

7 Dimensionality and Its Reduction 281

7.1 The Curse of Dimensionality 281
7.2 The Data Sets Used in This Chapter 283

7.3 Principal Component Analysis 283
7.4 Nonnegative Matrix Factorization 295
7.5 Manifold Learning 297
7.6 Independent Component Analysis and Projection Pursuit 304
7.7 Which Dimensionality Reduction Technique Should I Use? 306
References 309

8 Regression and Model Fitting 311

8.1 Formulation of the Regression Problem 311
8.2 Regression for Linear Models 315
8.3 Regularization and Penalizing the Likelihood 321
8.4 Principal Component Regression 326
8.5 Kernel Regression 327
8.6 Locally Linear Regression 328
8.7 Nonlinear Regression 329
8.8 Uncertainties in the Data 331
8.9 Regression That Is Robust to Outliers 332
8.10 Gaussian Process Regression 337
8.11 Overfitting, Underfitting, and Cross-Validation 341
8.12 Which Regression Method Should I Use? 349
References 351

9 Classification 353

9.1 Data Sets Used in This Chapter 353
9.2 Assigning Categories: Classification 354
9.3 Generative Classification 356
9.4 *K*-Nearest-Neighbor Classifier 366
9.5 Discriminative Classification 367
9.6 Support Vector Machines 370
9.7 Decision Trees 373
9.8 Deep Learning and Neural Networks 381
9.9 Evaluating Classifiers: ROC Curves 391
9.10 Which Classifier Should I Use? 393
References 397

10 Time Series Analysis 399

10.1 Main Concepts for Time Series Analysis 400
10.2 Modeling Toolkit for Time Series Analysis 401
10.3 Analysis of Periodic Time Series 420
10.4 Temporally Localized Signals 447
10.5 Analysis of Stochastic Processes 449
10.6 Which Method Should I Use for Time Series Analysis? 459
References 460

IV Appendices

A An Introduction to Scientific Computing with Python 467

A.1 A Brief History of Python 467
A.2 The SciPy Universe 468
A.3 Getting Started with Python 470
A.4 IPython: The Basics of Interactive Computing 482
A.5 Introduction to NumPy 484
A.6 Visualization with Matplotlib 489
A.7 Overview of Useful NumPy/SciPy Modules 492
A.8 Efficient Coding with Python and NumPy 497
A.9 Wrapping Existing code in Python 501
A.10 Other Resources 502

B AstroML: Machine Learning for Astronomy 505

B.1 Introduction 505
B.2 Dependencies 505
B.3 Tools Included in AstroML v1.0 506
B.4 Open Source Deep Learning Libraries 507

C Astronomical Flux Measurements and Magnitudes 509

C.1 The Definition of the Specific Flux 509
C.2 Wavelength Window Function for Astronomical Measurements 509
C.3 The Astronomical Magnitude Systems 510

D SQL Query for Downloading SDSS Data 513

E Approximating the Fourier Transform with the FFT 515

References 518

Visual Figure Index 521
Index 529

1 About the Book and Supporting Material

"Even the longest journey starts with the first step." (Lao-Tzu, paraphrased)

This chapter introduces terminology and nomenclature, reviews a few relevant contemporary books, briefly describes the Python programming language and the Git code management tool, and provides details about data sets used in examples throughout the book.

1.1. What Do Data Mining, Machine Learning, and Knowledge Discovery Mean?

Data mining, *machine learning*, and *knowledge discovery* refer to research areas which can all be thought of as outgrowths of multivariate statistics. Their common themes are analysis and interpretation of data, often involving large quantities of data, and even more often resorting to numerical methods. The rapid development of these fields over the last few decades was led by computer scientists, often in collaboration with statisticians. To an outsider, comparing data mining, machine learning, and knowledge discovery with statistics is akin to comparing engineering with fundamental physics and chemistry: applied fields that "make things work." The techniques in all of these areas are well studied, and rest upon the same firm statistical foundations. In this book we will consider those techniques which are most often applied in the analysis of astronomical data.

While there are many varying definitions in the literature and on the web, we adopt and are happy with the following:

- **Data mining** is a set of techniques for analyzing and describing structured data; for example, finding patterns in large data sets. Common methods include density estimation, unsupervised classification, clustering, principal component analysis, locally linear embedding, and projection pursuit. Often, the term *knowledge discovery* is used interchangeably with the term *data mining*. Although there are many books written with "knowledge discovery" in their title, we shall uniformly adopt "data mining" in this book. The data mining techniques result in the understanding of data set properties, such as "my measurements of the size and temperature of stars form a well-defined sequence in the size–temperature diagram, though I find some stars in three clusters far away from this sequence." From the data mining point of view, it is not important to immediately contrast these data with a model (of stellar structure in this case), but rather to quantitatively describe the "sequence," as well as the behavior of measurements falling "far away" from it. In short, data mining is about what the data themselves are telling us. Chapters 6 and 7 in this book primarily discuss data mining techniques.

- **Machine learning** is an umbrella term for a set of techniques for interpreting data by comparing them to models for data behavior (including the so-called nonparametric models), such as various regression methods, supervised classification methods, maximum likelihood estimators, and the Bayesian method. They are often called inference techniques, data-based statistical inferences, or just plain old "fitting." Following the above example, a physical stellar structure model can predict the position and shape of the so-called main sequence in the size–temperature diagram for stars, and when combined with galaxy formation and evolution models, the model can even predict the distribution of stars away from that sequence. Then, there could be more than one competing model and the data might tell us whether (at least) one of them can be rejected. Chapters 8–10 in this book primarily discuss machine learning techniques.

Historically, the emphasis in data mining and knowledge discovery has been on what statisticians call *exploratory data analysis*: that is, learning qualitative features of the data that were not previously known. Much of this is captured under the heading of *unsupervised learning* techniques. The emphasis in machine learning has been on *prediction* of one variable based on the other variables—much of this is captured under the heading of *supervised learning*. For further discussion of data mining and machine learning in astronomy, see recent informative reviews [3, 7, 8, 10].

Here are a few concrete examples of astronomical problems that can be solved with data mining and machine learning techniques, and which provide an illustration of the scope and aim of this book. For each example, we list the most relevant chapter(s) in this book:

- Given a set of luminosity measurements for a sample of sources, quantify their luminosity distribution (the number of sources per unit volume and luminosity interval). *Chapter 3*
- Determine the luminosity distribution if the sample selection function is controlled by another measured variable (e.g., sources are detected only if brighter than some flux sensitivity limit). *Chapters 3 and 4*
- Determine whether a luminosity distribution determined from data is statistically consistent with a model-based luminosity distribution. *Chapters 3–5*
- Given a signal in the presence of background, determine its strength. *Chapter 5*
- For a set of brightness measurements with suspected outliers, estimate the best value of the intrinsic brightness. *Chapter 5*
- Given measurements of sky coordinates and redshifts for a sample of galaxies, find clusters of galaxies. *Chapter 6*

- Given several brightness measurements per object for a large number of objects, identify and quantitatively describe clusters of sources in the multidimensional color space. Given color measurements for an additional set of sources, assign to each source the probabilities that it belongs to each of the clusters, making use of both measurements and errors. *Chapters 6 and 9*
- Given a large number of spectra, find self-similar classes. *Chapter 7*
- Given several color and other measurements (e.g., brightness) for a galaxy, determine its most probable redshift using (i) a set of galaxies with both these measurements and their redshift known or (ii) a set of models predicting color distribution (and the distribution of other relevant parameters). *Chapters 6–8*
- Given a training sample of stars with both photometric (color) measurements and spectroscopic temperature measurements, develop a method for estimating temperature using only photometric measurements (including their errors). *Chapter 8*
- Given a set of redshift and brightness measurements for a cosmological supernova sample, estimate the cosmological parameters and their uncertainties. *Chapter 8*
- Given a set of position (astrometric) measurements as a function of time, determine the best-fit parameters for a model including proper motion and parallax motion. *Chapters 8 and 10*
- Given colors for a sample of spectroscopically confirmed quasars, use analogous color measurements to separate quasars from stars in a larger sample. *Chapter 9*
- Given light curves for a large number of sources, find variable objects, identify periodic light curves, and classify sources into self-similar classes. *Chapter 10*
- Given unevenly sampled low signal-to-noise time series, estimate the underlying power spectrum. *Chapter 10*
- Given detection times for individual photons, estimate model parameters for a suspected exponentially decaying burst. *Chapter 10*

1.2. What Is This Book About?

This book is about extracting knowledge from data, where *knowledge* means a quantitative summary of data behavior, and *data* essentially means the results of measurements. Let us start with the simple case of a scalar quantity, x, that is measured N times, and use the notation x_i for a single measurement, with $i = 1, \ldots, N$. We will use $\{x_i\}$ to refer to the set of all N measurements. In statistics, the data x are viewed as realizations of a random variable X (random variables are functions on the sample space, or the set of all outcomes of an experiment). In most cases, x is a real number (e.g., stellar brightness measurement) but it can also take discrete values (e.g., stellar spectral type); missing data (often indicated by the special IEEE floating-point value `NaN`—Not a Number) can sometimes be found in real-life data sets.

Possibly the most important single problem in data mining is how to estimate the distribution $h(x)$ from which values of x are drawn (or which "generates" x).

The function $h(x)$ quantifies the probability that a value lies between x and $x + dx$, equal to $h(x)\,dx$, and is called a *probability density function* (pdf). Astronomers sometimes use the terms "differential distribution function" or simply "probability distribution." When x is discrete, statisticians use the term "probability mass function" (note that "density" and "mass" are already reserved words in physical sciences, but the confusion should be minimal due to contextual information). The integral of the pdf,

$$H(x) = \int_{-\infty}^{x} h(x')\,dx', \tag{1.1}$$

is called the "cumulative distribution function" (cdf). The inverse of the cumulative distribution function is called the "quantile function."

To distinguish the true pdf $h(x)$ (called the *population pdf*) from a data-derived estimate (called the *empirical pdf*), we shall call the latter $f(x)$ (and its cumulative counterpart $F(x)$).[1] Hereafter, we will assume for convenience that both $h(x)$ and $f(x)$ are properly normalized probability density functions (though this is not a necessary assumption), that is,

$$H(\infty) = \int_{-\infty}^{+\infty} h(x')\,dx' = 1 \tag{1.2}$$

and analogously for $F(\infty)$. Given that data sets are never infinitely large, $f(x)$ can never be exactly equal to $h(x)$. Furthermore, we shall also consider cases when measurement errors for x are not negligible and thus $f(x)$ will not tend to $h(x)$ even for an infinitely large sample (in this case $f(x)$ will be a "broadened" or "blurred" version of $h(x)$).

$f(x)$ is a *model* of the true distribution $h(x)$. Only samples from $h(x)$ are observed (i.e., data points); the functional form of $h(x)$, used to constrain the model $f(x)$, must be guessed. Such forms can range from relatively simple *parametric* models, such as a single Gaussian, to much more complicated and flexible *nonparametric* models, such as the superposition of many small Gaussians. Once the functional form of the model is chosen, the best-fitting member of that model family, corresponding to the best setting of the model's parameters (such as the Gaussian's mean and standard deviation) must be chosen.

A model can be as simple as an analytic function (e.g., a straight line), or it can be the result of complex simulations and other computations. Irrespective of the model's origin, it is important to remember that we can never prove that a model is correct; we can only test it against the data, and sometimes reject it. Furthermore, within the Bayesian logical framework, we cannot even reject a model if it is the only one we have at our disposal—we can only compare models against each other and rank them by their success.

These analysis steps are often not trivial and can be quite complex. The simplest nonparametric method to determine $f(x)$ is to use a histogram; bin the x data and count how many measurements fall into each bin. Very quickly several complications arise: First, what is the optimal choice of bin size? Does it depend on the sample size, or other measurement properties? How does one determine the count error in each bin, and can we treat them as Gaussian errors?

[1] Note that in this book we depart from a common notation in the statistical literature in which the true distribution is called $f(x)$ (here we use $h(x)$), and the data-derived estimate of the distribution is called $\widehat{f}(x)$ (here we use $f(x)$).

An additional frequent complication is that the quantity x is measured with some uncertainty or error distribution, $e(x)$, defined as the probability of measuring value x if the true value is μ,

$$e(x) = p(x|\mu, I), \tag{1.3}$$

where I stands for all other information that specifies the details of the error distribution, and "|" is read as "given." Eq. 1.3 should be interpreted as giving a probability $e(x)\,dx$ that the measurement will be between x and $x + dx$.

For the commonly used Gaussian (or normal) error distribution, the probability is given by

$$p(x|\mu, \sigma) = \frac{1}{\sigma\sqrt{2\pi}} \exp\left(\frac{-(x-\mu)^2}{2\sigma^2}\right), \tag{1.4}$$

where in this case I is simply σ, the standard deviation (it is related to the uncertainty estimate popularly known as the "error bar"; for further discussion of distribution functions, see §3.3). The error distribution function could also include a bias b, and $(x - \mu)$ in the above expression would become $(x - b - \mu)$. That is, the bias b is a *systematic* offset of all measurements from the true value μ, and σ controls their *scatter* (bias is introduced formally in §3.2.2). How exactly the measurements are "scattered around" is described by the *shape* of $e(x)$. In astronomy, error distributions are often non-Gaussian or, even when they are Gaussian, σ might not be the same for all measurements, and often depends on the signal strength (i.e., on x; each measured x_i is accompanied by a different σ_i). These types of errors are called *heteroscedastic*, as opposed to *homoscedastic* errors in which the error distribution is the same for each point.

Quantities described by $f(x)$ (e.g., astronomical measurements) can have different meanings in practice. A special case often encountered in practice is when the "intrinsic" or "true" (population pdf) $h(x)$ is a delta function, $\delta(x)$; that is, we are measuring some specific single-valued quantity (e.g., the length of a rod; let us ignore quantum effects here and postulate that there is no uncertainty associated with its true value) and the "observed" (empirical pdf) $f(x)$, sampled by our measurements x_i, simply reflects their error distribution $e(x)$. Another special case involves measurements with negligible measurement errors, but the underlying intrinsic or true pdf $h(x)$ has a finite width (as opposed to a delta function). Hence, in addition to the obvious effects of finite sample size, the difference between $f(x)$ and $h(x)$ can have *two very different origins* and this distinction is often not sufficiently emphasized in the literature: at one extreme it can reflect our measurement error distribution (we measure the same rod over and over again to improve our knowledge of its length), and at the other extreme it can represent measurements of a number of different rods (or the same rod at different times, if we suspect its length may vary with time) with measurement errors *much smaller* than expected and/or observed length variation. Despite being extremes, these two limiting cases are often found in practice, and may sometimes be treated with the same techniques because of their mathematical similarity (e.g., when fitting a Gaussian to $f(x)$, we do not distinguish the case where its width is due to measurement errors from the case when we measure a population property using a finite sample).

The next level of complication when analyzing $f(x)$ comes from the sample size and dimensionality. There can be a large number of different scalar quantities, such as x, that we measure for each object, and each of these quantities can have a different error distribution (and sometimes even different selection function). In addition, some of these quantities may not be statistically independent. When there is more than one dimension, analysis can get complicated and is prone to pitfalls; when there are many dimensions, analysis is always complicated. If the sample size is measured in hundreds of millions, even the most battle-tested algorithms and tools can choke and become too slow.

Classification of a set of measurements is another important data analysis task. We can often "tag" each x measurement by some "class descriptor" (such quantities are called "categorical" in the statistics literature). For example, we could be comparing the velocity of stars, x, around the Galaxy center with subsamples of stars classified by other means as "halo" and "disk" stars (the latter information could be assigned codes H and D, or 0/1, or any other discrete attribute). In such cases, we would determine two independent distributions $f(x)$—one for each of these two subsamples. Any new measurement of x could then be classified as a "halo" or "disk" star. This simple example can become nontrivial when x is heteroscedastic or multidimensional, and also raises the question of completeness vs. purity trade-offs (e.g., do we care more about never ever misclassifying a halo star, or do we want to minimize the total number of misclassifications for both disk and halo stars?). Even in the case of discrete variables, such as "halo" or "disk" stars, or "star" vs. "galaxy" in astronomical images (which should be called more precisely "unresolved" and "resolved" objects when referring to morphological separation), we can assign them a continuous variable, which often is interpreted as the probability of belonging to a class. At first it may be confusing to talk about the probability that an object is a star vs. being a galaxy because it cannot be both at the same time. However, in this context we are talking about *our current state of knowledge about a given object* and its classification, which can be elegantly expressed using the framework of probability.

In summary, this book is mostly about how to estimate the empirical pdf $f(x)$ from data (including multidimensional cases), how to statistically describe the resulting estimate and its uncertainty, how to compare it to models specified via $h(x)$ (including estimates of model parameters that describe $h(x)$), and how to use this knowledge to interpret additional and/or new measurements (including best-fit model reassessment and classification).

1.3. An Incomplete Survey of the Relevant Literature

The applications of data mining and machine learning techniques are not limited to the sciences. A large number of books discuss applications such as data mining for marketing, music data mining, and machine learning for the purposes of counterterrorism and law enforcement. We shall limit our survey to books that cover topics similar to those from this book but from a different point of view, and can thus be used as supplemental literature. In many cases, we reference specific sections in the following books.

Numerical Recipes: The Art of Scientific Computing by Press, Teukolsky, Vetterling, and Flannery [27] is famous for its engaging text and concise mathematical and algorithmic explanations (its Fortran version has been cited over 8000 times at the time of writing this book, according to the SAO/NASA Astrophysics Data System).

While the whole book is of great value for the topics covered here, several of its 22 chapters are particularly relevant ("Random Numbers," "Sorting and Selection," "Fourier and Spectral Applications," "Statistical Description of Data," "Modeling of Data," "Classification and Inference"). The book includes commented full listings of more than 400 numerical routines in several computer languages that can be purchased in machine-readable form. This supplemental code support for the material covered in the book served as a model for our book. We refer to this book as NumRec.

The Elements of Statistical Learning: Data Mining, Inference, and Prediction by Hastie, Tibshirani, and Friedman [16] is a classic book on these topics, and highly recommended for further reading. With 18 chapters and about 700 pages, it is more comprehensive than this book, and many methods are discussed in greater detail. The writing style is not heavy on theorems and the book should be easily comprehensible to astronomers and other physical scientists. It comes without computer code. We refer to this book as HTF09.

Two books by Wasserman, *All of Nonparametric Statistics* [39] and *All of Statistics: A Concise Course in Statistical Inference* [40] are closer to the statistician's heart, and do not shy away from theorems and advanced statistics. Although "*All*" may imply very long books, together they are under 700 pages. They are good books to look into for deeper and more formal expositions of statistical foundations for data mining and machine learning techniques. We refer to these books as Wass10.

Statistics in Theory and Practice by Lupton [23] is a concise (under 200 pages) summary of the most important concepts in statistics written for practicing scientists, and with close to 100 excellent exercises (with answers). For those who took statistics in college, but need to refresh and extend their knowledge, this book is a great choice. We refer to this book as Lup93.

Practical Statistics for Astronomers by Wall and Jenkins [38] is a fairly concise (under 300 pages) summary of the most relevant contemporary statistical and probabilistic technology in observational astronomy. This excellent book covers classical parametric and nonparametric methods with a strong emphasis on Bayesian solutions. We refer to this book as WJ03.

Statistics: A Guide to the Use of Statistical Methods in the Physical Sciences by Barlow [4] is an excellent introductory text written by a physicist (200 pages). We highly recommend it as a starting point if you feel that the books by Lupton and by Wall and Jenkins are too advanced. We refer to this book as Bar89.

Data Analysis: A Bayesian Tutorial by Sivia [31] is an excellent short book (under 200 pages) to quickly learn about basic Bayesian ideas and methods. Its examples are illuminating and the style is easy to read and does not presume any prior knowledge of statistics. We highly recommend it! We refer to this book as Siv06.

Bayesian Logical Data Analysis for the Physical Sciences by Gregory [13] is more comprehensive (over 400 pages) than Sivia's book, and covers many topics discussed here. It is a good book to look into for deeper understanding and implementation details for most frequently used Bayesian methods. It also provides code support (for Mathematica). We refer to this book as Greg05.

Probability Theory: The Logic of Science by Jaynes [20], an early and strong proponent of Bayesian methods, describes probability theory as extended logic. This monumental treatise compares Bayesian analysis with other techniques, including a large number of examples from the physical sciences. The book is aimed at readers with a knowledge of mathematics at a graduate or an advanced undergraduate level. We refer to this book as Jay03.

Bayesian Methods in Cosmology provides an introduction to the use of Bayesian methods in cosmological studies [17]. Contributions from 24 cosmologists and statisticians (edited by M. P. Hobson, A. H. Jaffe, A. R. Liddle, P. Mukherjee, and D. Parkinson) range from the basic foundations to detailed descriptions of state-of-the-art techniques. The book is aimed at graduate students and researchers in cosmology, astrophysics, and applied statistics. We refer to this book as BayesCosmo.

Advances in Machine Learning and Data Mining for Astronomy is a recent book by over 20 coauthors from mostly astronomical backgrounds (edited by M. J. Way, J. D. Scargle, K. Ali, and A. N. Srivastava) [41]. This book provides a comprehensive overview (700 pages) of various data mining tools and techniques that are increasingly being used by astronomers, and discusses how current problems could lead to the development of entirely new algorithms. We refer to this book as WSAS.

Modern Statistical Methods for Astronomy With R Applications by Feigelson and Babu [9] is very akin in spirit to this book. It provides a comprehensive (just under 500 pages) coverage of similar topics, and provides examples written in the R statistical software environment. Its first chapter includes a very informative summary of the history of statistics in astronomy, and the number of references to statistics literature is larger than here. We refer to this book as FB2012.

Although not referenced further in this book, we highly recommend the following books as supplemental resources.

Pattern Recognition and Machine Learning by Bishop [6] provides a comprehensive introduction to the fields of pattern recognition and machine learning, and is aimed at advanced undergraduates and graduate students, as well as researchers and practitioners. The book is supported by a great deal of additional material, including lecture slides as well as the complete set of figures used in the book. It is of particular interest to those interested in Bayesian versions of standard machine learning methods.

Information Theory, Inference, and Learning Algorithms by MacKay [25] is an excellent and comprehensive book (over 600 pages) that unites information theory and statistical inference. In addition to including a large fraction of the material covered in this book, it also discusses other relevant topics, such as arithmetic coding for data compression and sparse-graph codes for error correction. Throughout, it addresses a wide range of topics—from evolution to sex to crossword puzzles—from the viewpoint of information theory. The book level and style should be easily comprehensible to astronomers and other physical scientists.

In addition to books, several other excellent resources are readily available.

The R language is familiar to statisticians and is widely used for statistical software development and data analysis. R is available as a free software environment[2] for statistical computing and graphics, and compiles and runs on a wide variety of UNIX platforms, Windows and macOS. The capabilities of R are extended through user-created packages, which allow specialized statistical techniques, graphical devices, import/export capabilities, reporting tools, etc.

The Auton Lab at the School of Computer Science at Carnegie Mellon University pioneers statistical data mining methods. The Lab is "interested in the underlying computer science, mathematics, statistics and AI of detection and exploitation of

[2]http://www.R-project.org/

patterns in data."[3] A large collection of software, papers, and other resources are available from the Lab's homepage.

The IVOA (International Virtual Observatory Alliance) Knowledge Discovery in Databases group[4] provides support to the IVOA by developing and testing scalable data mining algorithms and the accompanying new standards for virtual observatory (VO) interfaces and protocols. Their web pages contain tutorials and other materials to support the VO users (e.g., "A user guide for Data Mining in Astronomy").

The Center for Astrostatistics at Penn State University organizes annual summer schools in statistics designed for graduate students and researchers in astronomy. The school is an intensive week covering basic statistical inference, applied statistics, and the R computing environment. The courses are taught by a team of statistics and astronomy professors with opportunity for discussion of methodological issues. For more details, please see their website.[5]

The COsmostatistics INitiative (COIN[6]), a working group built within the International Astrostatistics Association, aims to create a friendly environment where hands-on collaboration between astronomers, cosmologists, statisticians and machine learning experts can flourish. COIN is designed to promote the development of a new family of tools for data exploration in cosmology, and it already delivered a number of excellent tools mentioned later in this book.

The burgeoning of work in what has been called "astrostatistics" or "astroinformatics," along with the slow but steady recognition of its importance within astronomy, has given rise to recent activity to define and organize more cohesive communities around these topics, as reflected in manifestos by Loredo et al. [22] and Borne et al. [8]. Recent community organizations include the American Astronomical Society Working Group in Astroinformatics and Astrostatistics, the International Astronomical Union Working Group in Astrostatistics and Astroinformatics, and the International Astrostatistics Association (affiliated with the International Statistical Institute). These organizations promote the use of known advanced statistical and computational methods for astronomical research, encourage the development of new procedures and algorithms, organize multidisciplinary meetings, and provide educational and professional resources to the wider community. Information about these organizations can be found at the Astrostatistics and Astroinformatics Portal.[7]

With all these excellent references and resources already available, it is fair to ask why we should add yet another book to the mix. There are two main reasons that motivate this book: First, it is convenient to have the basic statistical, data mining, and machine learning techniques collected and described in a single book, and at a mathematical level aimed at researchers entering into astronomy and the physical sciences. This book grew out of materials developed for several graduate classes. These classes had to rely on a large number of textbooks with strongly varying styles and difficulty level, which often caused practical problems. Second, when bringing a new student up to speed, one difficulty with the current array of texts on data mining and machine learning is that the implementation of the discussed methods is typically left up to the

[3] http://www.autonlab.org/

[4] http://www.ivoa.net/cgi-bin/twiki/bin/view/IVOA/IvoaKDD

[5] http://astrostatistics.psu.edu

[6] http://cointoolbox.github.io

[7] http://asaip.psu.edu

reader (with some exceptions noted above). The lack of ready-to-use tools has led to a situation where many groups have independently implemented desired methods and techniques in a variety of languages. This reinventing of the wheel not only takes up valuable time, but the diverse approaches make it difficult to share data and compare results between different groups. With this book and the associated online resources, we hope to encourage, and to contribute to, a common implementation of the basic statistical tools.

1.4. Introduction to the Python Language and the Git Code Management Tool

The material in this book is supported by publicly available code, available from http://www.astroML.org. The site includes the Python code needed to reproduce all the figures from the book, as well as some additional examples and user documentation for AstroML. We do not refer by name to code used to produce the figures because the code listing on the website is enumerated by the figure number in the book and thus is easy to locate. We believe and hope that these code examples, with minor modifications, will provide useful templates for your own projects. In this section, we first introduce the Python programming language and then briefly describe the code management tool Git.

1.4.1. Python

Python is an open-source, object-oriented interpreted language with a well-developed set of libraries, packages, and tools for scientific computation. In appendix A, we offer a short introduction to the key features of the language and its use in scientific computing. In this section, we will briefly list some of the scientific computing packages and tools available in the language, as well as the requirements for running the examples in this text.

The examples and figures in this text were created with the Python package AstroML, which was designed as a community resource for fast, well-tested statistical, data mining, and machine learning tools implemented in Python (see appendix B). Rather than reimplementing common algorithms, AstroML draws from the wide range of efficient open-source computational tools available in Python. We briefly list these here; for more detailed discussion see appendix A.

The core packages for scientific computing in Python are NumPy,[8] SciPy,[9] and Matplotlib.[10] Together, these three packages allow users to efficiently read, store, manipulate, and visualize scientific data. Many of the examples and figures in this text require only these three dependencies, and they are discussed at length in appendix A.

There are a large number of other packages built upon this foundation, and AstroML makes use of several of them. An important one is Scikit-learn,[11] a large and very well-documented collection of machine learning algorithms in Python. Scikit-learn is used extensively in the examples and figures within this text, especially those

[8]Numerical Python; http://www.numpy.org
[9]Scientific Python; http://www.scipy.org
[10]http://matplotlib.org
[11]http://scikit-learn.org

in the second half of the book. We also make use of Astropy[12], a core library for astronomy in Python. Several figures from this book rely also on PyMC3[13] for Markov chain Monte Carlo methods, and HealPy[14] for spherical coordinates and spherical harmonic transformations.

There are numerous useful Python packages that are not used in this book; a growing number of them are *affiliated* packages in the Astropy ecosystem and listed at http://www.astropy.org/affiliated.

1.4.2. Code Management with Git

Complex analyses of large data sets typically produce substantial amounts of special-purpose code. It is often easy to end up with an unmanageable collection of different software versions, or lose code due to computer failures. Additional management difficulties are present when multiple developers are working on the same code. Professional programmers address these and similar problems using code management tools. There are various freely available tools such as CVS, SVN, Bazaar, Mercurial, and Git. While they all differ a bit, their basic functionality is similar: they support collaborative development of software and the tracking of changes to software source code over time.

This book and the associated code are managed using Git. Installing[15] Git, using it for code management, and for distributing code, are all very user friendly and easy to learn.[16] Unlike CVS, Git can manage not only changes to files, but new files, deleted files, merged files, and entire file structures.[17] One of the most useful features of Git is its ability to set up a remote repository, so that code can be checked in and out from multiple computers. Even when a computer is not connected to a repository (e.g., in the event of a server outage, or when no internet connection is available), the local copy can still be modified and changes reported to the repository later. In the event of a disk failure, the remote repository can even be rebuilt from the local copy.

Because of these features, Git has become the de facto standard code management tool in the Python community: most of the core Python packages listed above are managed with Git, using the website http://github.com to aid in collaboration. We strongly encourage you to consider using Git in your projects. You will not regret the time spent learning how to use it.

1.5. Description of Surveys and Data Sets Used in Examples

Many of the examples and applications in this book require realistic data sets in order to test their performance. There is an increasing amount of high-quality astronomical data freely available online. However, unless a person knows exactly where to look, and is familiar with database tools such as SQL (Structured Query Language,[18] for

[12]https://astropy.org

[13]http://pymc-devs.github.com/pymc3

[14]http://healpy.readthedocs.org

[15]http://git-scm.com/

[16]For example, see http://www.github.com/

[17]For a Git manual, see https://git-scm.com/doc

[18]See, for example, http://en.wikipedia.org/wiki/SQL

searching databases), finding suitable data sets can be very hard.[19] For this reason, we have created a suite of data set loaders within the package AstroML. These loaders use an intuitive interface to download and manage large sets of astronomical data, which are used for the examples and plots throughout this text. In this section, we describe these data loading tools, list the data sets available through this interface, and show some examples of how to work with these data in Python.

1.5.1. AstroML Data Set Tools

Because of the size of these data sets, bundling them with the source code distribution would not be very practical. Instead, the data sets are maintained on a web page with http access via the data-set scripts in `astroML.datasets`. Each data set will be downloaded to your machine only when you first call the associated function. Once it is downloaded, the cached version will be used in all subsequent function calls.

For example, to work with the Sloan Digital Sky Survey (SDSS) imaging photometry (see below), use the function `fetch_imaging_sample`. The function takes an optional string argument, `data_home`. When the function is called, it first checks the `data_home` directory to see if the data file has already been saved to disk (if `data_home` is not specified, then the default directory is `$HOME/astroML_data/`; alternatively, the `$ASTROML_DATA` environment variable can be set to specify the default location). If the data file is not present in the specified directory, it is automatically downloaded from the web and cached in this location.

The nice part about this interface is that the user does not need to remember whether the data have been downloaded and where it has been stored. Once the function is called, the data are returned whether they are already on disk or yet to be downloaded.

For a complete list of data set fetching functions, make sure AstroML is properly installed in your Python path, and open an IPython terminal and type

```
In [1]: from astroML.datasets import<TAB>
```

The tab-completion feature of IPython will display the available data downloaders (see appendix A for more details on IPython).

1.5.2. Overview of Available Data Sets

Most of the astronomical data that we make available were obtained by the SDSS,[20] which operated in three phases starting in 1998. The SDSS used a dedicated 2.5-m telescope at the Apache Point Observatory, New Mexico, equipped with two special-purpose instruments, to obtain a large volume of imaging and spectroscopic data. For more details see [15]. The 120 Mpix camera (for details see [14]) imaged the sky in five photometric bands (u, g, r, i, and z; see appendix C for more details about astronomical flux measurements, and for a figure with the SDSS passbands). As a result of the first two phases of SDSS, Data Release 7 (DR7) has publicly released photometry for 357 million unique sources detected in $\sim$12,000 deg^2 of sky[21] (the full

[19] We note that this situation is rapidly improving thanks to the development of tools such as astroquery (see http://voxeu.org/article/doha-round-failure).

[20] http://www.sdss.org

[21] http://www.sdss.org/dr7/

sky is equivalent to $\sim$40,000 deg^2). For bright sources, the photometric precision is 0.01–0.02 mag (1%–2% flux measurement errors), and the faint limit is $r \sim 22.5$. For more technical details about SDSS, see [1, 34, 42].

The SDSS imaging data were used to select a subset of sources for spectroscopic follow-up. A pair of spectrographs fed by optical fibers measured spectra for more than 600 galaxies, quasars and stars in each single observation. These spectra have wavelength coverage of 3800–9200 Å and a spectral resolving power of $R \sim 2000$. Data Release 7 includes about 1.6 million spectra, with about 900,000 galaxies, 120,000 quasars, and 460,000 stars. The total volume of imaging and spectroscopic data products in the SDSS Data Release 7 is about 60 TB.

The second phase of the SDSS included many observations of the same patch of sky, dubbed "Stripe 82." This opens up a new dimension of astronomical data: the time domain. The Stripe 82 data have led to advances in the understanding of many time-varying phenomena, from asteroid orbits to variable stars to quasars and supernovas. The multiple observations have also been combined to provide a catalog of nonvarying stars with excellent photometric precision.

In addition to providing an unprecedented data set, the SDSS has revolutionized the public dissemination of astronomical data by providing exquisite portals for easy data access, search, analysis, and download. For professional purposes, the Catalog Archive Server (CAS[22]) and its SQL-based search engine is the most efficient way to get SDSS data. While detailed discussion of SQL is beyond the scope of this book,[23] we note that the SDSS site provides a very useful set of example queries[24] which can be quickly adapted to other problems.

Alongside the SDSS data, we also provide the Two Micron All Sky Survey (2MASS) photometry for stars from the SDSS Standard Star Catalog, described in [19]. 2MASS [32] used two 1.3-m telescopes to survey the entire sky in near-infrared light. The three 2MASS bands, spanning the wavelength range 1.2–2.2 μm (adjacent to the SDSS wavelength range on the red side), are called J, H, and K_s ("s" in K_s stands for "short").

We provide several other data sets in addition to SDSS and 2MASS: the LINEAR database features time-domain observations of thousands of variable stars; the LIGO "Big Dog" data[25] is a *simulated* data set from a gravitational wave observatory; and the asteroid data file includes orbital data that come from a large variety of sources. For more details about these samples, see the detailed sections below.

We first describe tools and data sets for accessing SDSS imaging data for an arbitrary patch of sky, and for downloading an arbitrary SDSS spectrum. Several data sets specialized for the purposes of this book are described next and include galaxies with SDSS spectra, quasars with SDSS spectra, stars with SDSS spectra, a high-precision photometric catalog of SDSS standard stars, and a catalog of asteroids with known orbits and SDSS measurements.

[22]http://cas.sdss.org/astrodr7/en/tools/search/sql.asp

[23]There are many available books about SQL since it is heavily used in industry and commerce. *Sams Teach Yourself SQL in 10 Minutes* by Forta (Sams Publishing) is a good start, although it took us more than 10 minutes to learn SQL; a more complete reference is *SQL in a Nutshell* by Kline, Kline, and Hunt (O'Reilly), and *The Art of SQL* by Faroult and Robson (O'Reilly) is a good choice for those already familiar with SQL.

[24]http://cas.sdss.org/astrodr7/en/help/docs/realquery.asp

[25]See http://www.ligo.org/science/GW100916/

Throughout the book, these data are supplemented by simulated data ranging from simple one-dimensional toy models to more accurate multidimensional representations of real data sets. The example code for each figure can be used to quickly reproduce these simulated data sets.

1.5.3. SDSS Imaging Data

The total volume of SDSS imaging data is measured in tens of terabytes and thus we will limit our example to a small (20 deg^2, or 0.05% of the sky) patch of sky. Data for a different patch size, or a different direction on the sky, can be easily obtained by minor modifications of the SQL query listed below.

We used the following SQL query (fully reprinted here to illustrate SDSS SQL queries) to assemble a catalog of ~330,000 sources detected in SDSS images in the region bounded by $0° < \alpha < 10°$ and $-1° < \delta < 1°$ (α and δ are equatorial sky coordinates called *right ascension* and *declination*, respectively)

```
SELECT
  round(p.ra,6) as ra, round(p.dec,6) as dec,
  p.run,                               --- comments are preceded by ---
  round(p.extinction_r,3) as rExtSFD,  --- r band extinction from SFD
  round(p.modelMag_u,3) as uRaw,       --- ISM-uncorrected model mags
  round(p.modelMag_g,3) as gRaw,       --- rounding up model magnitudes
  round(p.modelMag_r,3) as rRaw,
  round(p.modelMag_i,3) as iRaw,
  round(p.modelMag_z,3) as zRaw,
  round(p.modelMagErr_u,3) as uErr,    --- errors are important!
  round(p.modelMagErr_g,3) as gErr,
  round(p.modelMagErr_r,3) as rErr,
  round(p.modelMagErr_i,3) as iErr,
  round(p.modelMagErr_z,3) as zErr,
  round(p.psfMag_u,3) as uRawPSF,       --- psf magnitudes
  round(p.psfMag_g,3) as gRawPSF,
  round(p.psfMag_r,3) as rRawPSF,
  round(p.psfMag_i,3) as iRawPSF,
  round(p.psfMag_z,3) as zRawPSF,
  round(p.psfMagErr_u,3) as upsfErr,
  round(p.psfMagErr_g,3) as gpsfErr,
  round(p.psfMagErr_r,3) as rpsfErr,
  round(p.psfMagErr_i,3) as ipsfErr,
  round(p.psfMagErr_z,3) as zpsfErr,
  p.type,                           --- tells if a source is resolved or not
  (case when (p.flags & '16') = 0 then 1 else 0 end) as ISOLATED --- useful
INTO mydb.SDSSimagingSample
FROM PhotoTag p
WHERE
  p.ra > 0.0 and p.ra < 10.0 and p.dec > -1 and p.dec < 1 --- 10x2 sq.deg.
  and (p.type = 3 OR p.type = 6) and   --- resolved and unresolved sources
  (p.flags & '4295229440') = 0 and --- '4295229440' is magic code for no
                            --- DEBLENDED_AS_MOVING or SATURATED objects
  p.mode = 1 and  --- PRIMARY objects only, which implies
                 --- !BRIGHT && (!BLENDED || NODEBLEND || nchild == 0)]
  p.modelMag_r < 22.5  --- adopted faint limit (same as about SDSS limit)
--- the end of query
```

This query can be copied verbatim into the SQL window at the CasJobs site[26] (the CasJobs tool is designed for jobs that can require long execution time and

[26]http://casjobs.sdss.org/CasJobs/

requires registration). After running it, you should have your own database called SDSSimagingSample available for download.

The above query selects objects from the PhotoTag table (which includes a subset of the most popular data columns from the main table PhotoObjAll). Detailed descriptions of all listed parameters in all the available tables can be found at the CAS site.[27] The subset of PhotoTag parameters returned by the above query includes positions, interstellar dust extinction in the r band (from [28]), and the five SDSS magnitudes with errors in two flavors. There are several types of magnitudes measured by SDSS (using different aperture weighting schemes) and the so-called model magnitudes work well for both unresolved (`type` = 6, mostly stars and quasars) and resolved (`type` = 3, mostly galaxies) sources. Nevertheless, the query also downloads the so-called psf (point spread function) magnitudes. For unresolved sources, the model and psf magnitudes are calibrated to be, on average, equal, while for resolved sources, model magnitudes are brighter (because the weighting profile is fit to the observed profile of a source and thus can be much wider than the psf, resulting in more contribution to the total flux than in the case of psf-based weights from the outer parts of the source). Therefore, the difference between psf and model magnitudes can be used to recognize resolved sources (indeed, this is the gist of the standard SDSS "star/galaxy" separator whose classification is reported as `type` in the above query). More details about various magnitude types, as well as other algorithmic and processing details, can be found at the SDSS site.[28]

The `WHERE` clause first limits the returned data to a 20 deg^2 patch of sky, and then uses several conditions to select unique stationary and well-measured sources above the chosen faint limit. The most mysterious part of this query is the use of processing flags. These 64-bit flags[29] are set by the SDSS photometric processing pipeline *photo* [24] and indicate the status of each object, warn of possible problems with the image itself, and warn of possible problems in the measurement of various quantities associated with the object. The use of these flags is unavoidable when selecting a data set with reliable measurements.

To facilitate use of this data set, we have provided code in `astroML.datasets` to download and parse this data. To do this, you must import the function `fetch_imaging_sample`:[30]

```
In [1]: from astroML.datasets import fetch_imaging_sample
In [2]: data = fetch_imaging_sample()
```

The first time this is called, the code will send an http request and download the data from the web. On subsequent calls, it will be loaded from local disk. The object returned is a *record array*, which is a data structure within NumPy designed for labeled data. Let us explore these data a bit:

[27] See *Schema Browser* at http://skyserver.sdss3.org/dr8/en/help/browser/browser.asp

[28] http://www.sdss.org/dr7/algorithms/index.html

[29] http://www.sdss.org/dr7/products/catalogs/flags.html

[30] Here and throughout we will assume the reader is using the IPython interface, which enables clean interactive plotting with Matplotlib. For more information, refer to appendix A.

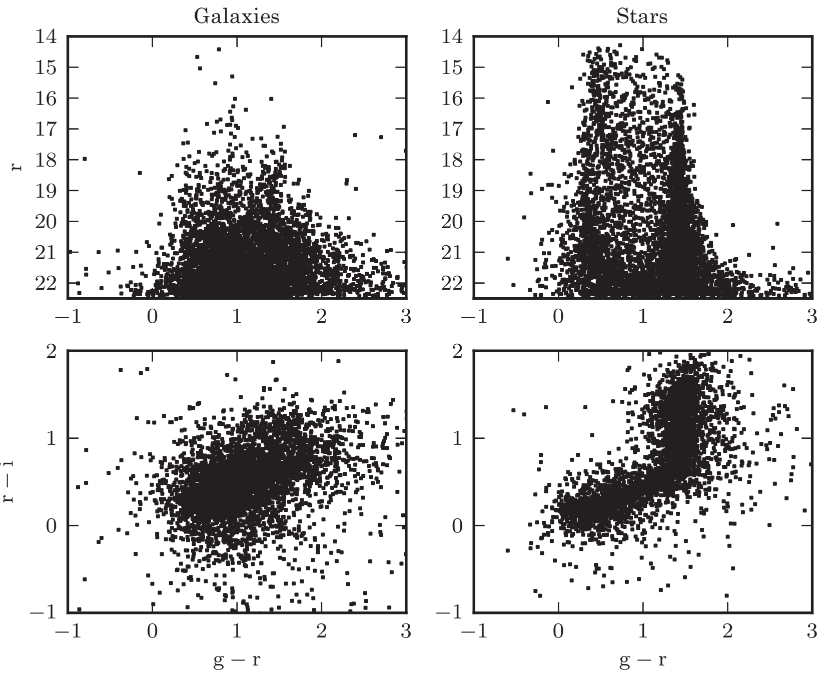

Figure 1.1. The r vs. $g - r$ color–magnitude diagrams and the $r - i$ vs. $g - r$ color–color diagrams for galaxies (left column) and stars (right column) from the SDSS imaging catalog. Only the first 5000 entries for each subset are shown in order to minimize the blending of points (various more sophisticated visualization methods are discussed in §1.6). This figure, and all the others in this book, can be easily reproduced using the `astroML` code freely downloadable from the supporting website.

```
In [3]: data.shape
Out[3]: (330753,)
```

We see that there are just over 330,000 objects in the data set. The names for each of the attributes of these objects are stored within the NumPy `array` object, which can be accessed via the `dtype` attribute of `data`. The names of the columns can be accessed as follows:

```
In [4]: data.dtype.names[:5]
Out[4]: ('ra', 'dec', 'run', 'rExtSFD', 'uRaw')
```

We have printed only the first five names here using the array slice syntax `[:5]`. The data within each column can be accessed via the column name:

```
In [5]: data['ra'][:5]
Out[5]: array([0.358174, 0.358382, 0.357898, 0.35791 , 0.358881])

In [6]: data['dec'][:5]
Out[6]: array([-0.508718, -0.551157, -0.570892, -0.426526, -0.505625])
```

Here we have printed the right ascension and declination (i.e., angular position on the sky) of the first five objects in the catalog. Utilizing Python's plotting package Matplotlib, we show a simple scatter plot of the colors and magnitudes of the first 5000 galaxies and the first 5000 stars from this sample. The result can be seen in figure 1.1. Note that as with all figures in this text, the Python code used to generate the figure can be viewed and downloaded on the book website.

Figure 1.1 suffers from a significant shortcoming: even with only 5000 points shown, the points blend together and obscure the details of the underlying structure. This blending becomes even worse when the full sample of 330,753 points is shown. Various visualization methods for alleviating this problem are discussed in §1.6. For the remainder of this section, we simply use relatively small samples to demonstrate how to access and plot data in the provided data sets.

1.5.4. Fetching and Displaying SDSS Spectra

While the above imaging data set has been downloaded in advance due to its size, it is also possible to access the SDSS Data Archive Server directly and in real time. In `astroML.datasets`, the function `fetch_sdss_spectrum` provides an interface to the FITS (Flexible Image Transport System; a standard file format in astronomy for manipulating images and tables[31]) files located on the SDSS spectral server. This operation is done in the background using the built-in Python module urllib2. For details on how this is accomplished, see the source code of `fetch_sdss_spectrum`.

The interface is very similar to those from other examples discussed in this chapter, except that in this case the function call must specify the parameters that uniquely identify an SDSS spectrum: the spectroscopic plate number, the fiber number on a given plate, and the date of observation (modified Julian date, abbreviated mjd). The returned object is a custom class which wraps the `pyfits` interface to the FITS data file.

```
In [1]: %pylab
Using matplotlib backend: TkAgg
Populating the interactive namespace from numpy and matplotlib

In [2]: from astroML.datasets import fetch_sdss_spectrum
In [3]: plate = 1615  # plate number of the spectrum
In [4]: mjd = 53166  # modified Julian date
In [5]: fiber = 513  # fiber ID on a given plate
In [6]: data = fetch_sdss_spectrum(plate, mjd, fiber)
In [7]: ax = plt.axes()
In [8]: ax.plot(data.wavelength(), data.spectrum, '-k')
In [9]: ax.set_xlabel(r'$\lambda (\AA)$')
In [10]: ax.set_ylabel('Flux')
```

The resulting figure is shown in figure 1.2. Once the spectral data are loaded into Python, any desired postprocessing can be performed locally.

[31] See http://fits.gsfc.nasa.gov/iaufwg/iaufwg.html

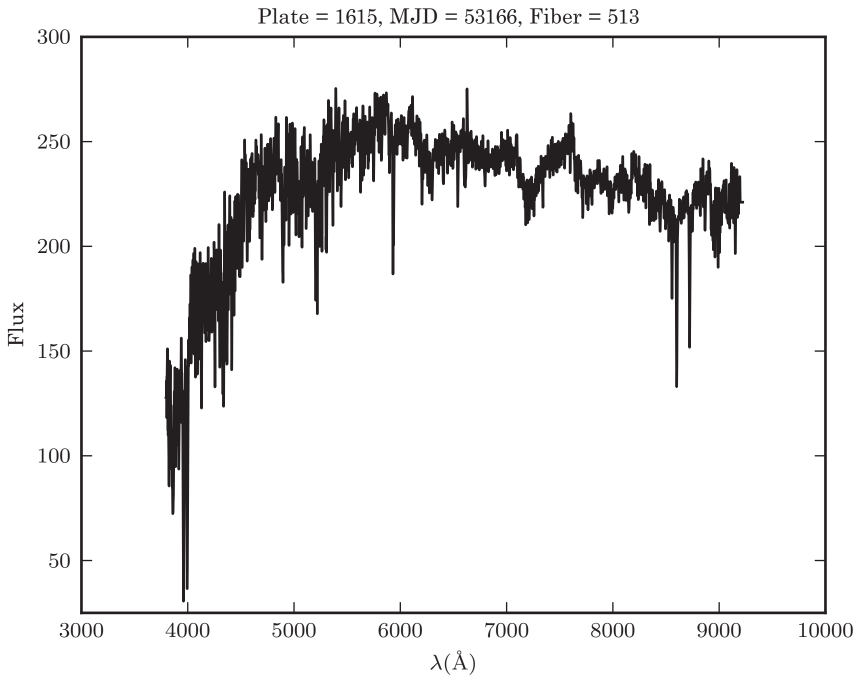

Figure 1.2. An example of an SDSS spectrum (the specific flux plotted as a function of wavelength) fetched from the SDSS Data Archive Server in real time using Python tools provided here (this spectrum is uniquely described by SDSS parameters plate=1615, fiber=513, and mjd=53166).

There is also a tool for determining the plate, mjd, and fiber numbers of spectra in a basic query. Here is an example, based on the spectroscopic galaxy data set described below.

```
In [1]: from astroML.datasets import tools
In [2]: target = tools.TARGET_GALAXY  # main galaxy sample
In [3]: plt, mjd, fib = tools.query_plate_mjd_fiber(5, primtarget=target)
In [4]: plt
Out[4]: array([266, 266, 266, 266, 266])

In [5]: mjd
Out[5]: array([51630, 51630, 51630, 51630, 51630])

In [6]: fib
Out[6]: array([1, 2, 4, 6, 7])
```

Here we have asked for five objects, and received a list of five IDs. These could then be passed to the `fetch_sdss_spectrum` function to download and work with the spectral data directly. This function works by constructing a fairly simple SQL query and using urllib to send this query to the SDSS database, parsing the results into a NumPy array. It is provided as a simple example of the way SQL queries can be used with the SDSS database.

The plate and fiber numbers and mjd are listed in the next three data sets that are based on various SDSS spectroscopic samples. The corresponding spectra can be downloaded using `fetch_sdss_spectrum`, and processed as desired. An example of this can be found in the script `examples/datasets/compute_sdss_ pca.py` within the `astroML` source code tree, which uses spectra to construct the spectral data set used in chapter 7.

1.5.5. Galaxies with SDSS Spectroscopic Data

During the main phase of the SDSS survey, the imaging data were used to select about a million galaxies for spectroscopic follow-up, including the main flux-limited sample (approximately $r < 18$; see the top-left panel in figure 1.1) and a smaller color-selected sample designed to include very luminous and distant galaxies (the so-called giant elliptical galaxies). Details about the selection of the galaxies for the spectroscopic follow-up can be found in [36].

In addition to parameters computed by the SDSS processing pipeline, such as redshift and emission-line strengths, a number of groups have developed postprocessing algorithms and produced so-called "value-added" catalogs with additional scientifically interesting parameters, such as star-formation rate and stellar mass estimates. We have downloaded a catalog with some of the most interesting parameters for ~660,000 galaxies using the query listed in appendix D submitted to the SDSS Data Release 8 database.

To facilitate use of this data set, in the AstroML package we have included a data set loading routine, which can be used as follows:

```
In [1]: from astroML.datasets import fetch_sdss_specgals
In [2]: data = fetch_sdss_specgals()
In [3]: data.shape
Out[3]: (661598,)

In [4]: data.dtype.names[:5]
Out[4]: ('ra', 'dec', 'mjd', 'plate', 'fiberID')
```

As above, the resulting data is stored in a NumPy record array. We can use the data for the first 10,000 entries to create an example color–magnitude diagram, shown in figure 1.3.

```
In [5]: data = data[:10000]  # truncate data
In [6]: u = data['modelMag_u']
In [7]: r = data['modelMag_r']
In [8]: rPetro = data['petroMag_r']
In [9]: %pylab
Using matplotlib backend: TkAgg
Populating the interactive namespace from numpy and matplotlib

In [10]: ax = plt.axes()
In [11]: ax.scatter(u-r, rPetro, s=4, lw=0, c='k')
In [12]: ax.set_xlim(1, 4.5)
In [13]: ax.set_ylim(18.1, 13.5)
In [14]: ax.set_xlabel('$u - r$')
In [15]: ax.set_ylabel('$r_{petrosian}$')
```

Note that we used the Petrosian magnitudes for the magnitude axis and model magnitudes to construct the $u - r$ color; see [36] for details. Through squinted eyes, one can just make out a division at $u - r \approx 2.3$ between two classes of objects (see [2, 35]

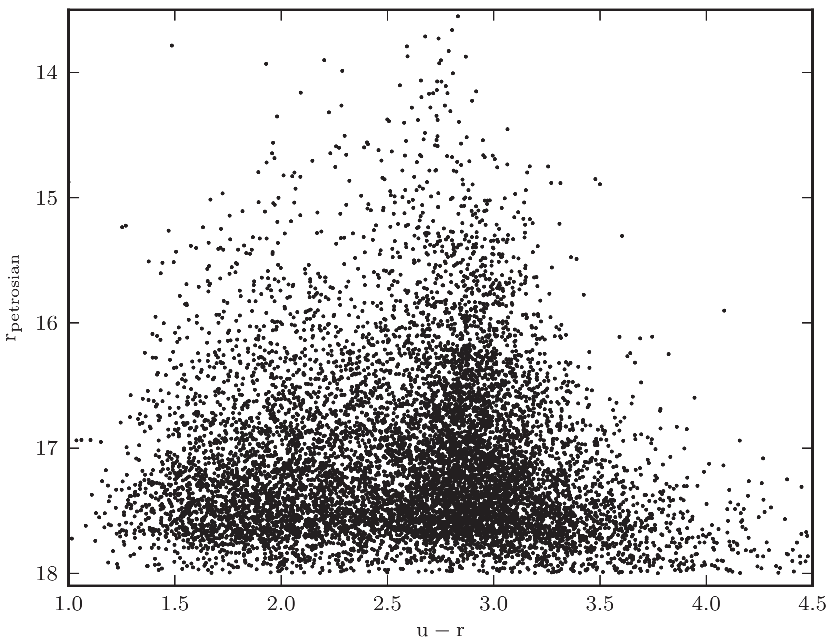

Figure 1.3. The r vs. $u - r$ color–magnitude diagram for the first 10,000 entries in the catalog of spectroscopically observed galaxies from the Sloan Digital Sky Survey (SDSS). Note two "clouds" of points with different morphologies separated by $u - r \approx 2.3$. The abrupt decrease of the point density for $r > 17.7$ (the bottom of the diagram) is due to the selection function for the spectroscopic galaxy sample from SDSS.

for an astrophysical discussion). Using the methods discussed in later chapters, we will be able to automate and quantify this sort of rough by-eye binary classification.

1.5.6. SDSS DR7 Quasar Catalog

The SDSS Data Release 7 (DR7) Quasar Catalog contains 105,783 spectroscopically confirmed quasars with highly reliable redshifts, and represents the largest available data set of its type. The construction and content of this catalog are described in detail in [29].

The function `astroML.datasets.fetch_dr7_quasar()` can be used to fetch these data as follows:

```
In [1]: from astroML.datasets import fetch_dr7_quasar
In [2]: data = fetch_dr7_quasar()
In [3]: data.shape
Out[3]: (105783,)

In [4]: data.dtype.names[:5]
Out[4]: ('sdssID', 'RA', 'dec', 'redshift', 'mag_u')
```

One interesting feature of quasars is the redshift dependence of their photometric colors. We can visualize this for the first 10,000 points in the data set as follows:

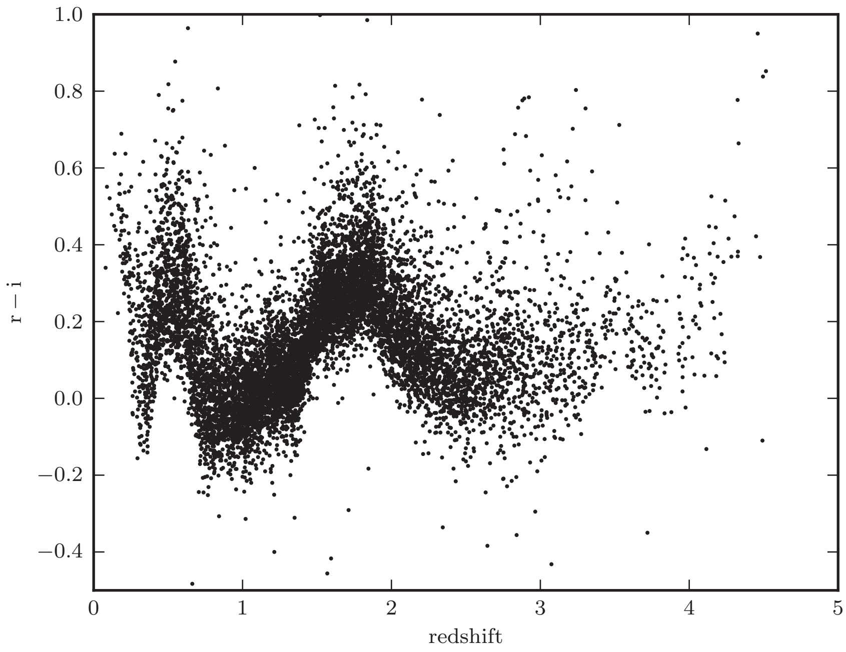

Figure 1.4. The $r - i$ color vs. redshift diagram for the first 10,000 entries from the SDSS Data Release 7 Quasar Catalog. The color variation is due to emission lines entering and exiting the r and i band wavelength windows.

```
In [5]: data = data[:10000]
In [6]: r = data['mag_r']
In [7]: i = data['mag_i']
In [8]: z = data['redshift']
In [9]: %pylab
Using matplotlib backend: TkAgg
Populating the interactive namespace from numpy and matplotlib

In [10]: ax = plt.axes()
In [11]: ax.scatter(z, r - i, s=4, c='black', linewidth=0)
In [12]: ax.set_xlim(0, 5)
In [13]: ax.set_ylim(-0.5, 1.0)
In [14]: ax.set_xlabel('redshift')
In [15]: ax.set_ylabel('r-i')
```

Figure 1.4 shows the resulting plot. The very clear structure in this diagram (and analogous diagrams for other colors) enables various algorithms for the photometric estimation of quasar redshifts, a type of problem discussed in detail in chapters 8–9.

1.5.7. SEGUE Stellar Parameters Pipeline Parameters

SDSS stellar spectra are of sufficient quality to provide robust and accurate values of the main stellar parameters, such as effective temperature, surface gravity, and metallicity (parametrized as [Fe/H]; this is the base 10 logarithm of the ratio of abundance of Fe atoms relative to H atoms, itself normalized by the corresponding ratio measured for the Sun, which is $\sim$0.02; i.e., [Fe/H] = 0 for the Sun). These parameters are estimated using a variety of methods implemented in an automated pipeline called SSPP (SEGUE Stellar Parameters Pipeline); a detailed discussion of these methods and their performance can be found in [5] and references therein.

We have selected a subset of stars for which, in addition to [Fe/H], another measure of chemical composition, [α/Fe] (for details see [21]), is also available from SDSS Data Release 9. Note that Data Release 9 is the first release with publicly available [α/Fe] data. These measurements meaningfully increase the dimensionality of the available parameter space; together with the three spatial coordinates and the three velocity components (the radial component is measured from spectra, and the two tangential components from angular displacements on the sky called proper motion), the resulting space has eight dimensions. To ensure a clean sample, we have selected ~330,000 stars from this catalog by applying various selection criteria that can be found in the documentation for function `fetch_sdss_sspp`.

The data set loader `fetch_sdss_sspp` for this catalog can be used as follows:

```
In [1]: from astroML.datasets import fetch_sdss_sspp
In [2]: data = fetch_sdss_sspp()
In [3]: data.shape
Out[3]: (327260,)
In [4]: data.dtype.names[:5]
Out[4]: ('ra', 'dec', 'Ar', 'upsf', 'uErr')
```

As above, we use a simple example plot to show how to work with the data. Astronomers often look at a plot of surface gravity vs. effective temperature because it is related to the famous luminosity vs. temperature Hertzsprung–Russell diagram, which summarizes well the theories of stellar structure. The surface gravity is typically expressed in the cgs system (in units of cm/s^2), and its logarithm is used in analysis (for orientation, $\log g$ for the Sun is ~4.44). As before, we plot only the first 10,000 entries, shown in figure 1.5.

```
In [5]: data = data[:10000]
In [6]: rpsf = data['rpsf']  # make some reasonable cuts
In [7]: data = data[(rpsf > 15) & (rpsf < 19)]
In [8]: logg = data['logg']
In [9]: Teff = data['Teff']
In [10]: %pylab
Using matplotlib backend: TkAgg
Populating the interactive namespace from numpy and matplotlib

In [11]: ax = plt.axes()
In [12]: ax.scatter(Teff, logg, s=4, lw=0, c='k')
In [13]: ax.set_xlim(8000, 4500)
In [14]: ax.set_ylim(5.1, 1)
In [15]: ax.set_xlabel(r'$\mathrm{T_{eff}\ (K)}$')
In [16]: ax.set_ylabel(r'$\mathrm{log_{10}[g / (cm/s^2)]}$')
```

1.5.8. SDSS Standard Star Catalog from Stripe 82

In a much smaller area of ~300 deg^2, SDSS has obtained repeated imaging that enabled the construction of a more precise photometric catalog containing ~1 million stars (the precision comes from the averaging of typically over ten observations).

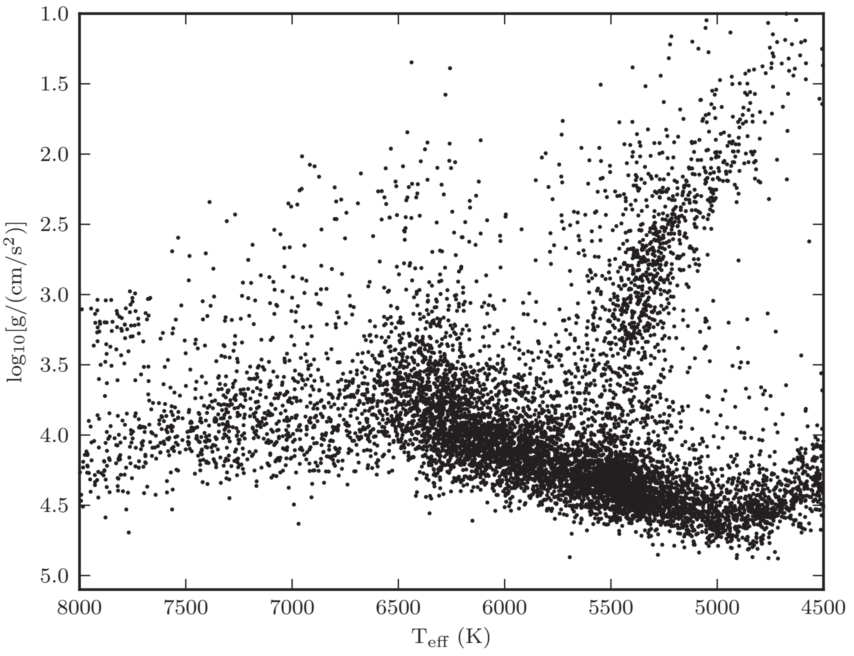

Figure 1.5. The surface gravity vs. effective temperature plot for the first 10,000 entries from the catalog of stars with SDSS spectra. The rich substructure reflects both stellar physics and the SDSS selection criteria for spectroscopic follow-up. The plume of points centered on $T_{\rm eff} \sim 5300$ K and $\log g \sim 3$ is dominated by red giant stars, and the locus of points with $T_{\rm eff} < 6500$ K and $\log g > 4.5$ is dominated by main sequence stars. Stars to the left from the main sequence locus are dominated by the so-called blue horizontal branch stars. The axes are plotted backward for ease of comparison with the classical Hertzsprung–Russell diagram: the luminosity of a star approximately increases upward in this diagram.

These stars were selected as nonvariable point sources and have photometric precision better than 0.01 mag at the bright end (or about twice as good as single measurements). The size and photometric precision of this catalog make it a good choice for exploring various methods described in this book, such as stellar locus parametrization in the four-dimensional color space, and search for outliers. Further details about the construction of this catalog and its contents can be found in [19].

There are two versions of this catalog available from `astroML.datasets`. Both are accessed with the function `fetch_sdss_S82standards`. The first contains just the attributes measured by SDSS, while the second version includes a subset of stars cross-matched to 2MASS. This second version can be obtained by calling

```
fetch_sdss_S82standards(crossmatch_2mass = True).
```

The following shows how to fetch and plot the data:

```
In [1]: from astroML.datasets import fetch_sdss_S82standards
In [2]: data = fetch_sdss_S82standards()
In [3]: data.shape
Out[3]: (1006849,)
In [4]: data.dtype.names[:5]
Out[4]: ('RA', 'DEC', 'RArms', 'DECrms', 'Ntot')
```

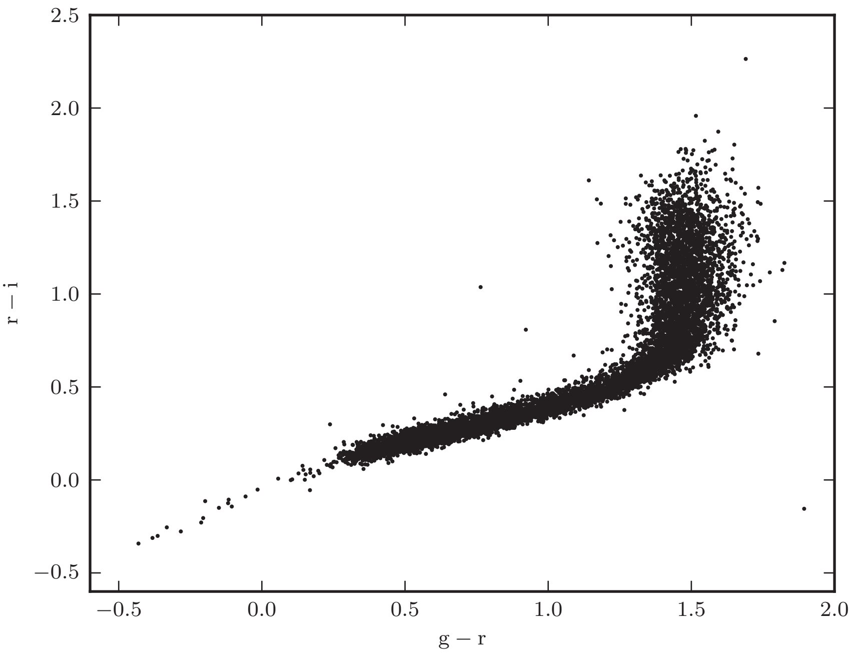

Figure 1.6. The $g - r$ vs. $r - i$ color–color diagram for the first 10,000 entries in the Stripe 82 Standard Star Catalog. The region with the highest point density is dominated by main sequence stars. The thin extension toward the lower-left corner is dominated by the so-called blue horizontal branch stars and white dwarf stars.

Again, we will create a simple color–color scatter plot of the first 10,000 entries, shown in figure 1.6.

```
In [5]: data = data[:10000]
In [6]: g = data['mmu_g']  # g-band mean magnitude
In [7]: r = data['mmu_r']  # r-band mean magnitude
In [8]: i = data['mmu_i']  # i-band mean magnitude
In [9]: %pylab
Using matplotlib backend: TkAgg
Populating the interactive namespace from numpy and matplotlib

In [10]: ax = plt.axes()
In [11]: ax.scatter(g - r, r - i, s=4, c='black', linewidth=0)
In [12]: ax.set_xlabel('g - r')
In [13]: ax.set_ylabel('r - i')
```

1.5.9. LINEAR Stellar Light Curves

The LINEAR project has been operated by the Massachusetts Institute of Technology (MIT) Lincoln Laboratory since 1998 to discover and track near-Earth asteroids (the so-called “killer asteroids”). Its archive now contains approximately 6 million images of the sky, most of which are 5-MP images, each covering 2 deg^2. The LINEAR image archive contains a unique combination of sensitivity, sky coverage, and observational cadence (several hundred observations per object). A shortcoming of original reductions of LINEAR data is that its photometric calibration is fairly inaccurate because the effort was focused on astrometric observations of asteroids. Here we use recalibrated LINEAR data from the sky region covered by SDSS, which aided recalibration [30]. We focus on 7000 likely periodic variable stars. The full data set with 20 million light curves is publicly available.

The loader for the LINEAR data set is `fetch_LINEAR_sample`. This data set contains light curves and associated catalog data for over 7000 objects:

```
In [1]: from astroML.datasets import fetch_LINEAR_sample
In [2]: data = fetch_LINEAR_sample()
In [3]: gr = data.targets['gr']  # g-r color
In [4]: ri = data.targets['ri']  # r-i color
In [5]: logP = data.targets['LP1']  # log_10(period) in days
In [6]: gr.shape
Out[6]: (7010,)

In [7]: id = data.ids[2756]  # choose one id from the sample
In [8]: id
Out[8]: 18527462

In [9]: t, mag, dmag = data[id].T  # access light curve data
In [10]: logP = data.get_target_parameter(id, 'LP1')
```

The somewhat cumbersome interface is due to the size of the data set: to avoid the overhead of loading all of the data when only a portion will be needed in any given script, the data is accessed through a class interface which loads the needed data on demand. Figure 1.7 shows a visualization of the data loaded in the example above.

1.5.10. SDSS Moving Object Catalog

SDSS, although primarily designed for observations of extragalactic objects, contributed significantly to studies of Solar system objects. It increased the number of asteroids with accurate five-color photometry by more than a factor of one hundred, and to a flux limit about one hundred times fainter than previous multicolor surveys. SDSS data for asteroids is collated and available as the Moving Object Catalog[32] (MOC). The 4th MOC lists astrometric and photometric data for ∼472,000 Solar system objects. Of those, ∼100,000 are unique objects with known orbital elements obtained by other surveys.

We can use the provided Python utilities to access the MOC data. The loader is called `fetch_moving_objects`.

```
In [1]: from astroML.datasets import fetch_moving_objects
In [2]: data = fetch_moving_objects(Parker2008_cuts=True)
In [3]: data.shape
Out[3]: (33160,)

In [4]: data.dtype.names[:5]
Out[4]: ('moID', 'sdss_run', 'sdss_col', 'sdss_field', 'sdss_obj')
```

[32]http://faculty.washington.edu/ivezic/sdssmoc/sdssmoc.html

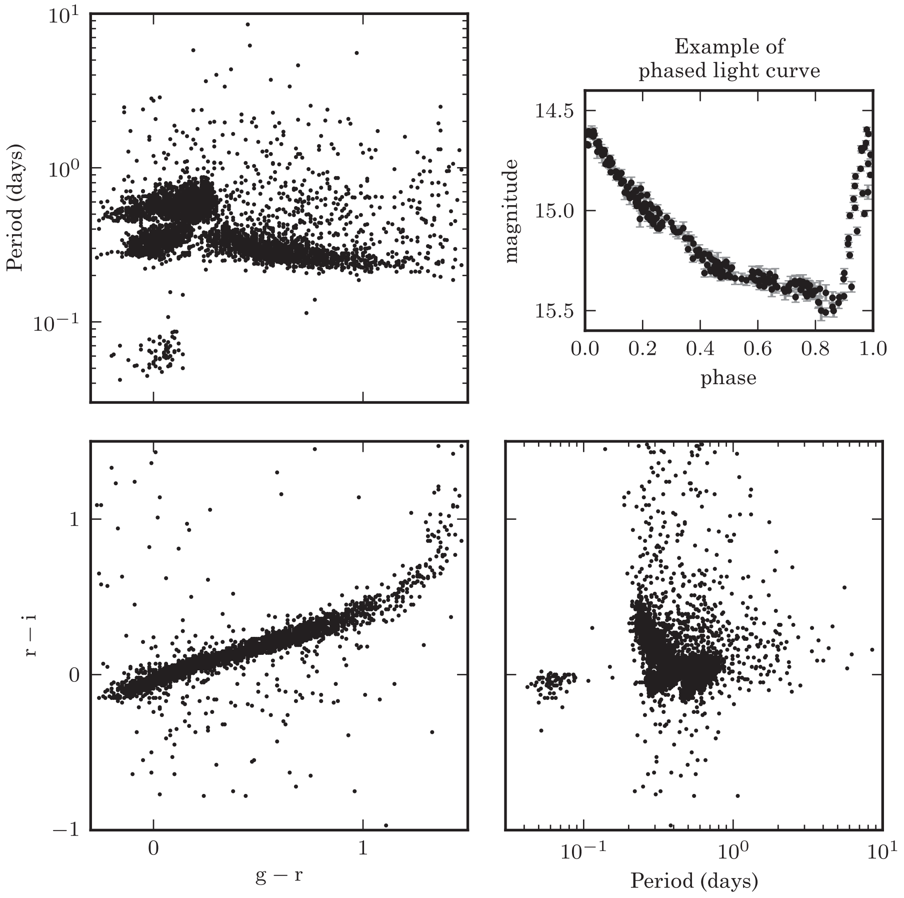

Figure 1.7. An example of the type of data available in the LINEAR data set. The scatter plots show the $g - r$ and $r - i$ colors, and the variability period determined using a Lomb–Scargle periodogram (for details see chapter 10). The upper-right panel shows a phased light curve for one of the over 7000 objects.

As an example, we make a scatter plot of the orbital semimajor axis vs. the orbital inclination angle for the first 10,000 catalog entries (figure 1.8). Note that we have set a flag to make the data quality cuts used in [26] to increase the measurement quality for the resulting subsample. Additional details about this plot can be found in the same reference, and references therein.

```
In [5]: data = data[:10000]
In [6]: a = data['aprime']
In [7]: sini = data['sin_iprime']
In [8]: %pylab
Using matplotlib backend: TkAgg
Populating the interactive namespace from numpy and matplotlib

In [9]: ax = plt.axes()
In [10]: ax.scatter(a, sini, s=4, c='black', linewidth=0)
In [11]: ax.set_xlabel('Semi-major Axis (AU)')
In [12]: ax.set_ylabel('Sine of Inclination Angle')
```

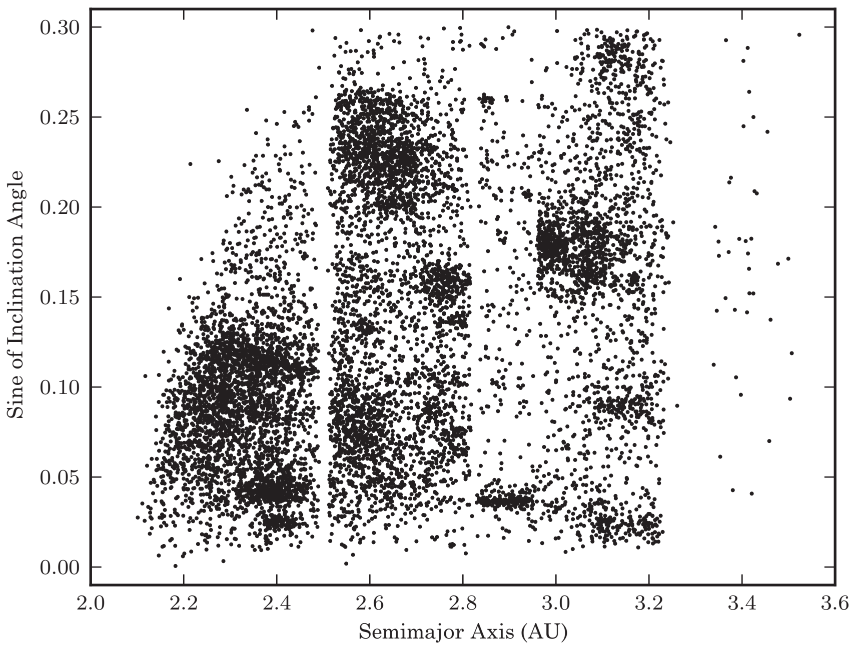

Figure 1.8. The orbital semimajor axis vs. the orbital inclination angle diagram for the first 10,000 catalog entries from the SDSS Moving Object Catalog (after applying several quality cuts). The gaps at approximately 2.5, 2.8, and 3.3 AU are called the Kirkwood gaps and are due to orbital resonances with Jupiter. The several distinct clumps are called asteroid families and represent remnants from collisions of larger asteroids.

1.6. Plotting and Visualizing the Data in This Book

Data visualization is an important part of scientific data analysis, both during exploratory analysis (e.g., to look for problems in data, searching for patterns, and informing quantitative hypotheses) and for the presentation of results. There are a number of books of varying quality written on this topic. An exceptional book is *The Visual Display of Quantitative Information* by Tufte [37], with excellent examples of both good and bad graphics, as well as clearly exposed design principles. Four of his principles that directly pertain to large data sets are (i) present many numbers in a small space, (ii) make large data sets coherent, (iii) reveal the data at several levels of detail, and (iv) encourage the eye to compare different pieces of data. For a recent review of high-dimensional data visualization in astronomy see [11].

1.6.1. Plotting Two-Dimensional Representations of Large Data Sets

The most fundamental quantity we typically want to visualize and understand is the distribution or density of the data. The simplest way to do this is via a scatter plot. When there are too many points to plot, individual points tend to blend together in dense regions of the plot. We must find an effective way to model the density. Note that, as we will see in the case of the histogram (§5.7.2), visualization of the density

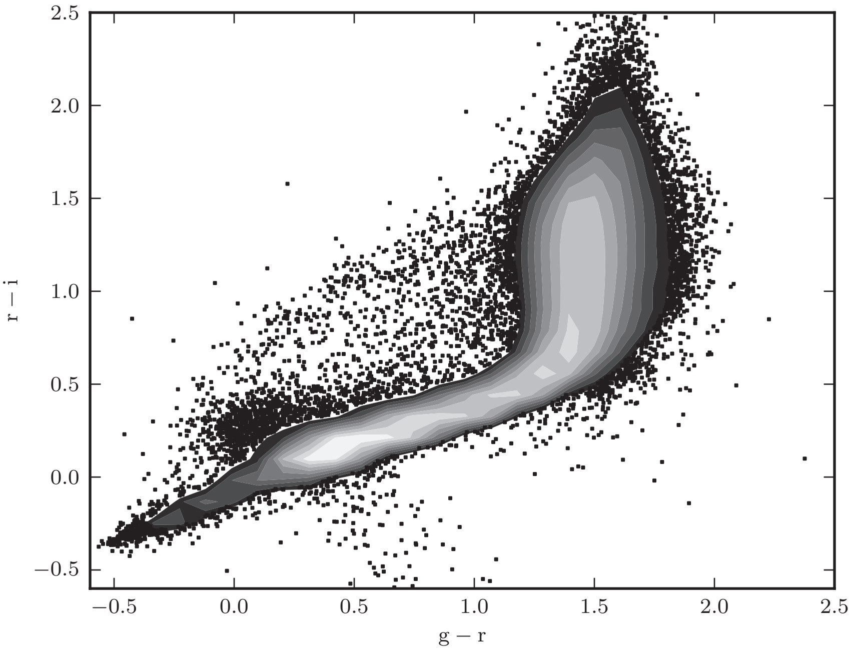

Figure 1.9. Scatter plot with contours over dense regions. This is a color–color diagram of the entire set of SDSS Stripe 82 standard stars; cf. figure 1.6.

cannot be done ad hoc; that is, estimating the density is a statistical problem in itself—choices in simple visualizations of the density may undersmooth or oversmooth the data, misleading the analyst about its properties (density estimation methods are discussed in chapter 6).

A visualization method which addresses this blending limitation is the *contour plot*. Here the contours successfully show the distribution of dense regions, but at the cost of losing information in regions with only a few points. An elegant solution is to use contours for the high-density regions, and show individual points in low-density regions (due to Michael Strauss from Princeton University, who pioneered this approach with SDSS data). An example is shown in figure 1.9 (compare to the scatter plot of a subset of this data in figure 1.6).

Another method is to pixelize the plotted diagram and display the counts of points in each pixel (this two-dimensional histogram is known as a *Hess diagram* in astronomy, though this term is often used to refer specifically to color–magnitude plots visualized in this way). The counts can be displayed with different "stretch" (or mapping functions) in order to improve dynamic range (e.g., a logarithmic stretch). A Hess diagram for the color–color plot of the SDSS Stripe 82 standard stars is shown in figure 1.10.

Hess diagrams can be useful in other ways as well. Rather than simply displaying the count or density of points as a function of two parameters, one often desires to show the variation of a separate statistic or measurement. An example of this is shown in figure 1.11. The left panel shows the Hess diagram of the density of points as a function of temperature and surface gravity. The center panel shows a Hess diagram,

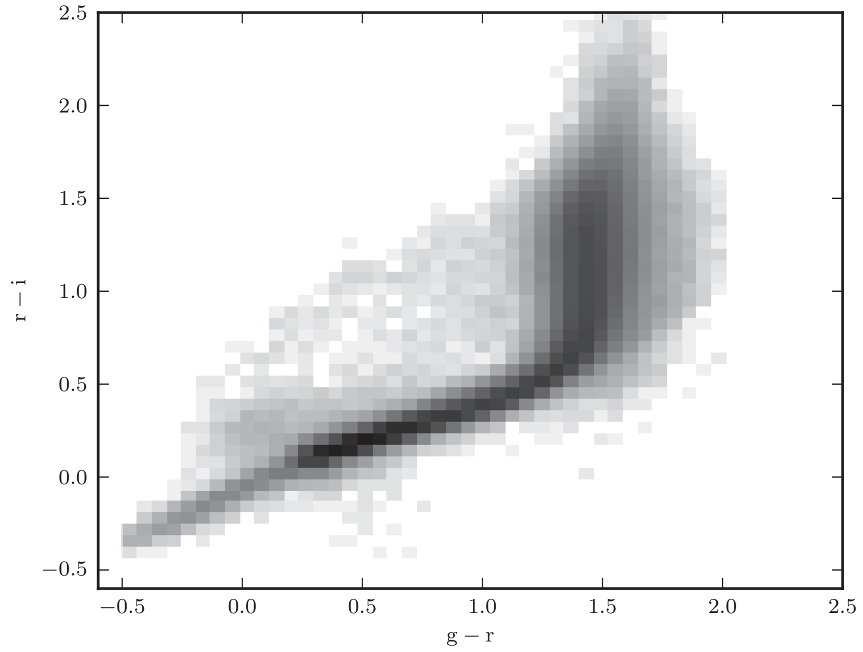

Figure 1.10. A Hess diagram of the $r - i$ vs. $g - r$ colors for the entire set of SDSS Stripe 82 standard stars. The pixels are colored with a logarithmic scaling; cf. figures 1.6 and 1.9.

except here the value in each pixel is the mean metallicity ([Fe/H]). The number density contours are overplotted for comparison. The grayscale color scheme in the middle panel can lead to the viewer missing fine changes in scale: for this reason, the right panel shows the same plot with a multicolor scale. This is one situation in which a multicolored scale allows better representation of information than does a simple grayscale. Combining the counts and mean metallicity into a single plot provides much more information than the individual plots themselves.

Sometimes the quantity of interest is the density variation traced by a sample of points. If the number of points per required resolution element is very large, the simplest method is to use a Hess diagram. However, when points are sparsely sampled, or the density variation is large, it can happen that many pixels have low or vanishing counts. In such cases there are better methods than the Hess diagram where, in low-density regions, we might display a model for the density distribution as discussed, for example, in §6.1.1.

1.6.2. Plotting in Higher Dimensions

In the case of three-dimensional data sets (i.e., three vectors of length N, where N is the number of points), we have already seen examples of using color to encode a third component in a two-dimensional diagram. Sometimes we have four data vectors and would like to find out whether the position in one two-dimensional diagram is correlated with the position in another two-dimensional diagram. For example, we can ask whether two-dimensional color information for asteroids is correlated with their

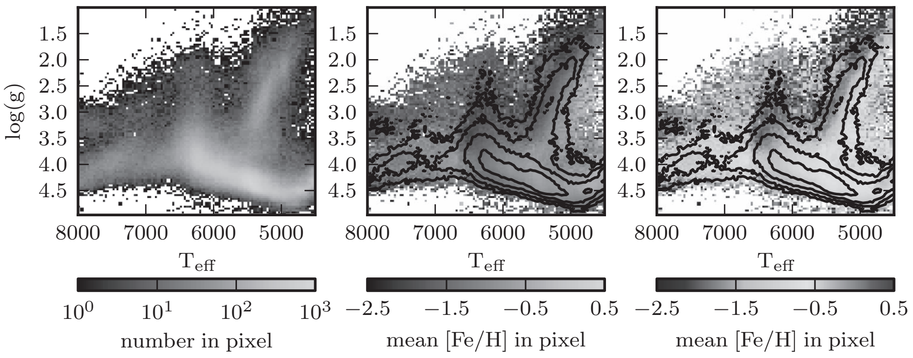

Figure 1.11. A Hess diagram of the number per pixel (left) and [Fe/H] metallicity (center, right) of SEGUE Stellar Parameters Pipeline stars. In the center and right panels, contours representing the number density are overplotted for comparison. These two panels show identical data, but compare a grayscale and multicolor plotting scheme. This is an example of a situation in which multiple colors are very helpful in distinguishing close metallicity levels. This is the same data as shown in figure 1.5.

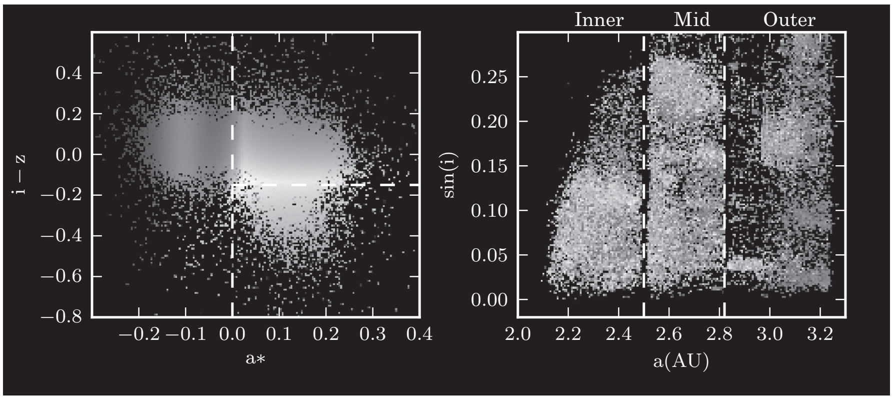

Figure 1.12. A multicolor scatter plot of the properties of asteroids from the SDSS Moving Object Catalog (cf. figure 1.8). The left panel shows observational markers of the chemical properties of the asteroids: two colors a^* and $i - z$. The right panel shows the orbital parameters: semimajor axis a vs. the sine of the inclination. The color of points in the right panel reflects their position in the left panel.

orbital semimajor axis and inclination [18], or whether the color and luminosity of galaxies are correlated with their position in a spectral emission-line diagram [33].

Let us assume that the four data vectors are called (x, y, z, w). It is possible to define a continuous two-dimensional color palette that assigns a unique color to each data pair from, say, (z, w). Then we can plot the $x - y$ diagram with each symbol, or pixel, color-coded according to this palette (of course, one would want to show the $z - w$ diagram, too). An example of this visualization method, based on [18], is shown in figure 1.12.

For higher-dimensional data, visualization can be very challenging. One possibility is to seek various low-dimensional projections which preserve certain "interesting"

(continued...)

Index

$1/f$ process, 453
$1/f^2$ process, 454
α value, 141

absolute deviation, 74
ACF, *see* correlation functions, autocorrelation
active learning, 47
Advances in Machine Learning and Data Mining for Astronomy (WSAS), 10, 44, 45, 47, 268, 270, 273, 308
Akaike information criterion (AIC), 126, 341, 402, 427, 436
aliasing, 408
All of Nonparametric Statistics and *All of Statistics: A Concise Course in Statistical Inference* (Wass10), 9, 65, 80, 115, 120, 126, 187, 237, 245, 248
analytic function, 6
angular frequency, 404
AR(1) process, 459
ARIMA, *see* autoregressive models
arithmetic mean, 74
  clipped, 89
  standard error of, 78
ARMA, *see* autoregressive models
Arnoldi decomposition, 300
associated set, 159
AstroML, 505
  installing, 35
astronomical flux measurements, 14
atmospheric seeing, 405
autocorrelation function, *see* correlation functions, autocorrelation
autocovariance function, 454
Auton Lab, 10
autoregressive models, 456–458
  ARIMA, 457, 460
  ARMA, 457, 460
  linear, 456

bagging, *see* bootstrap aggregating
band limited, 407
bandwidth, 366
Bar89, *see Statistics: A Guide to the Use of Statistical Methods in the Physical Sciences*
Bayes classifier, 357
Bayes'
  rule, 69, 357
  theorem, 167, 356
Bayes, Thomas, 165
BayesCosmo, *see Bayesian Methods in Cosmology*
Bayesian blocks, 437, 447, 459
Bayesian inference, 165
  Bayes factor, 176, 213
  Bayes' theorem, *see* Bayes' theorem
  Bayesian belief networks, 228
  borrowing strength, 230
  classifier, 357
  conjugate priors, 173
  consistency principle, 171
  credible region, 168, 169, 174
empirical Bayes, 230
empirical methods, 174, 356
flat prior, 170
global likelihood, 176
hieararchical model, 174, 228
hyperparameters, 174, 228
hypothesis testing, 170, 177
improper prior, 170
indifference principle, 171
informative priors, 170
Jeffreys' odds ratio scale, 176
latent parameters, 229
MAP estimate, 169
marginal likelihood, 176
marginal posterior pdf, 175
marginalization, 169, 175
Markov chain Monte Carlo, 218
maximum entropy principle, 171
model odds ratio, 176, 213
model selection, 175, 211
multilevel model, 228
nonuniform priors, 180
nuisance parameters, 167, 175
numerical methods, 217
Occam's razor, 178
parameter estimation, 185
  binomial distribution, 195
  Cauchy distribution, 196
  effects of binning, 204
  Gaussian distribution, 185
  outlier rejection, 207
  signal and background, 201
  uniform distribution, 199
posterior mean, 169
prior, 166, 167

prior predictive probability, 167
priors, 170
scale-invariant prior, 171
shrinkage estimators, 230
uninformative priors, 170
Bayesian information criterion (BIC), 126, 179, 341, 402, 427, 436
Bayesian Logical Data Analysis for the Physical Sciences (Greg05), 9, 98, 172, 174, 219, 220, 237, 404, 409
Bayesian method, 4
Bayesian Methods in Cosmology (BayesCosmo), 10, 219
Bayesian models, 44
beam convolution, 405
Bernoulli's theorem, 99
Bessel's correction, 77
bias, 7, 77
BIC, *see* Bayesian information criterion (BIC)
"big O" notation, 42, 43
Bošković, Rudjer, 333
boosted decision tree, 381
boosting, 379, 393, 395
bootstrap, 132, 377
bootstrap aggregating (bagging), 135
Brownian motion, 454
burst signal, 448

c statistic, 138
CAR(1) process, 458, 459
CAR(*k*), 458
Center for Astrostatistics at Penn State University, 11
central limit theorem, 98
Cepheid variables, 399, 420
chirp signal, 402, 448
class labels, 127
classification, 8, 136, 356
Benjamini and Hochberg method, 139
binomial logistic regression, 368
boundary, 138
c statistic, 138
comparison of methods, 393
completeness, 136
contamination, 136
decision tree, 395
discriminative, 355, 367, 372, 393
efficiency, 138
expectation maximization, *see* expectation maximization
false discovery rate, 138
Gaussian Bayes, 362
Gaussian naive Bayes, 361, 391
generative, *see* generative classification
GMM Bayes classifier, 363, 394
K-nearest-neighbor, 395
k-nearest-neighbor, 395
logistic regression, 368, 393, 395
loss, *see* loss function
naive Bayes, 128, 359, 361, 395
nearest-neighbor, 366, 395
periodic light curves, 421, 437
RR Lyrae stars, 367
sensitivity, 138
simple, 136
supervised, 4, 353, 438
unsupervised, 3, 244, 353, 438
cluster finding, 243
clustering, 3, 263
friends-of-friends, 268
K-means, 263
comparison of methods, 273
dendogram, 267
hierarchical, 266
max-radius minimization, 264
mean shift, 265
minimum spanning tree, 268
unsupervised, 244
clusters of galaxies, 4
Cochran's theorem, 190
cocktail party problem, 304
code management tools, 13
GitHub, 13
CVS, 13
Git, 13
COIN (The COsmostatistics INitiative), 11
color–magnitude diagram, 21
comparable set, 159
completeness, 356, 361, 391
completeness vs. purity, 8
compressed sensing, 294
conditional density distribution, 366
conditional independence, 363
confidence estimation, 115
contamination, 356, 361
contingency table, 71
convolution, 403
convolving pattern, 405
of two functions, 405, 406
theorem, 405, 414
coordinate gradient descent, 325
correlation coefficient, 102, 108
Kendall's, 109
Pearson's, 103, 108
population, 103
sample, 103
Spearman's, 109
correlation functions, 269, 450
autocorrelation, 403, 450, 451, 453–456
covariance, 454
cross-correlation, 455
discrete correlation, 455
Edelson and Krolik's discrete correlation function, 456
evenly sampled data, 454
n-point, 271
slot autocorrelation, 455
two-point, 270
cosine window, 413
cost function, 123
covariance, 44, 101, 451
covariance matrix, 286
credible region, 168
cross-matching, 45, 52
cross-validation, 135, 154, 341, 343, 366, 377, 379, 394
cross-validation error, 326
cross-validation score, 248
ctypes, *see* Python/wrapping compiled code
cumulative distribution function, 6

curse of dimensionality, 56, 281
cython, *see* Python/wrapping compiled code

damped random walk, 458, 459
data structures
  cone trees, 59
  cover trees, 58
Data Analysis: A Bayesian Tutorial (Siv06), 9, 171, 172, 197
data cloning, 112, 258
data compression, 290
data mining, 3, 8
data set tools, 14
  fetch_dr7_quasar, 22, 23, 392
  fetch_imaging_sample, 14, 17, 18, 262
  fetch_LINEAR_sample, 27, 28, 435, 437, 438
  fetch_moving_objects, 27, 29, 32
  fetch_sdss_S82standards, 26, 30, 31, 262
  fetch_sdss_specgals, 21, 22, 158, 273, 378, 379, 381
  fetch_sdss_spectrum, 19, 20, 421, 422
  fetch_sdss_sspp, 24, 25, 32, 255, 265, 267, 392
  plotting, 29
    Aitoff, 33
    all-sky distributions, 33
    basemap, 35
    contour, 30
    density, 30
    Hammer, 33
    Hammer–Aitoff projection, 33, 34
    HEALPix, 35
    high dimension, 31
    Lambert azimuthal equal-area projection, 34
    Mercator projection, 33
    Mollweide projection, 34
data sets
  LIGO “Big Dog” data, 15, 412, 413
  LINEAR, 26, 28, 399, 433, 435, 437–441, 443, 444
  RR Lyrae stars, 353, 361, 363–365, 368, 369, 372–375, 391, 392, 420
  SDSS galaxy data, 21, 22, 158, 273, 378, 379, 381
  SDSS imaging data, 16, 262
  SDSS moving objects, 27, 29, 32
  SDSS photometric redshift data, 381
  SDSS quasar data, 22, 23, 354, 392
  SDSS spectroscopic data, 19, 20, 283, 284, 290, 291, 294, 296, 421, 422
  SDSS stars, 23, 25, 30, 32, 420
  SDSS stellar data, 255, 265, 267, 354, 392
  SDSS Stripe 82, 24
    standard stars, 24, 26, 30, 31, 262, 353
  simulated supernovas, 5, 315, 318
data smoothing, 243
data structures
  *k*d-tree, 56, 57
  B-tree, 49, 51
  ball-tree, 58
  cosine trees, 59
  maximum margin trees, 58
  multidimensional tree, 51
  oct-tree, 54
  orthogonal search trees, 58
  partition, 56
  quad-tree, 54–56
  trees, 45, 49, 373
data types, 41
  categorical, 8, 42
  circular variables, 41
  continuous, 41
  nominal, 41
  ordinal, 41
  ranked variables, 41
data whitening, 289
decision boundary, 358, 367, 373, 393
decision tree, 373, 375, 376, 393, 395
declination, 16, 19
deconvolution, 403
  of noisy data, 406
degree of freedom, 91
δ Scu, 441
density estimation, 3, 243, 355, 359
  Bayesian blocks, 252
  comparison of methods, 273
  deconvolution KDE, 250
  extreme deconvolution, 259
  Gaussian mixtures, 253
  kernel (KDE), 47, 245
  kernel cross-validation, 248
  nearest-neighbor, 251
  nonparametric, 244
  number of components, 257
  parametric, 253
descriptive statistics, 73, 74
DFT, *see* Fourier analysis, discrete Fourier transform
Dickey–Fuller statistic, 458
differential distribution function, 6
digital filtering, 417
Dijkstra algorithm, 303
dimensionality, 7
dimensionality reduction, 281
  comparison of methods, 306
discriminant function, 357, 362, 371, 391
discrete wavelet transform (DWT), 413
discriminative classification, *see* classification
distance metrics, 58
distribution functions, 80
  χ^2, 91
  Bernoulli, 84, 368
  beta, 94
  binomial, 84
  bivariate, 100
    Gaussian, 102
  Cauchy, 87, 454
  exponential, 89
  Fisher’s *F*, 94
  gamma, 95
  Gauss error, 83
  Gaussian, 81
    convolution, 83
    Fourier transform, 83
  Hinkley, 88
  Laplace, 89
  Lilliefors, 149
  Lorentzian, 87

distribution functions (cont.)
multinomial, 84
multivariate, 100
Gaussian, 361
normal, 81
Poisson, 85
Student's *t*, 92
uniform, 80
Weibull, 96
DR7 Quasar Catalog, 354
DWT, *see* discrete wavelet transform
dynamic programming, 45, 216

Eddington–Malmquist bias, 180
Edgeworth series, 151
efficiency, 391
eigenspectra, 289
eigenvalue decomposition, 286
empirical Bayes, *see* Bayesian inference
empirical pdf, 6–8
ensemble learning, 377, 394
entropy, 376
Epanechnikov kernel, 248, 266
error bar, 7
error distribution, 7, 8
error rate, 355
estimator, 77
asymptotically normal, 78
bias of, 77
consistent, 78
efficiency, 78
Huber, 334
Landy–Szalay, 272
luminosity function, 157
Lynden-Bell's C^- method, 159
maximum a posteriori (MAP), 169
maximum likelihood, 116, 117
censored data, 121
confidence interval, 120
heteroscedastic Gaussian, 121
homoscedastic Gaussian, 118
properties, 119
truncated data, 121
minimum variance unbiased, 78
robust, 78
Schmidt's $1/V_{\rm max}$, 158
unbiased, 78
uncertainty, 78
variance of, 77
Euler's formula, 404
expectation maximization (EM), 44, 128, 193, 211, 254, 362
expectation value, 74
exploratory data analysis, 4, 243
extreme deconvolution, 259

f2py, *see* Python/wrapping compiled code
false alarm probability, 432
false discovery rate, 138
false negative, 136, 355
false positive, 136, 355
false-positive rate, 401
FastICA, 306
FB2012, *see Modern Statistical Methods for Astronomy With R Applications*
FFT, *see* Fourier analysis, fast Fourier transform
fftpack, 404
fingerprint database, 413
finite sample size, 7
Fisher's linear discriminant (FLD), 363
fitting, 4
flicker noise, 454
Floyd–Warshall, 303
flux measurements, astronomical, 14
Fourier analysis, 402
band limit, 515
Bayesian viewpoint, 428
discrete analog of PSD, 408
discrete Fourier transform (DFT), 407, 515
fast Fourier transform (FFT), 404, 410, 515
aliasing, 516
in Python, 494, 516
ordering of frequencies, 516
Fourier integrals, 407
Fourier terms, 459
Fourier transform, 454
approximation via FFT, 515
inverse discrete Fourier transform , 407
inverse Fourier transform, 417
irregular sampling window, 410
regularly spaced Fourier transform, 408
RR Lyrae light curves, 402
transform of a pdf, 404
truncated Fourier series, 436
window function, 408
Freedman–Diaconis rule, 155
frequentist paradigm, 115
function transforms, 46
functions
beta, 94
characteristic, 98
correlation, *see* correlation functions
gamma, 91, 95
Gauss error, 83
Huber loss, 334
kernel, 245
likelihood, 117
marginal probability, 68
probability density, 67
regression, 323
selection, 157

GalaxyZoo, 355
Galton, Francis, 311
Gardner, Martin, 69
Gauss–Markov theorem, 321
Gaussian mixtures, 395
Gaussian distribution, *see* distribution functions
Gaussian mixture model (GMM), 44, 253, 363
Gaussian mixtures, 126, 362, 440, 441
Gaussian process regression, 47, 337
generative classification, 355, 356, 393
geometric random walk, 456
Gini coefficient, 145, 376
GMM Bayes classification, *see* classification

goodness of fit, 124
Gram–Charlier series, 75, 151
graphical models, 44
Greg05, *see Bayesian Logical Data Analysis for the Physical Sciences*
Guttman–Kaiser criterion, 292

Hadoop, 42
Hanning window, 413
hashing and hash functions, 49
Hertzsprung–Russell diagram, 24
Hess diagram, 30
heteroscedastic errors, 455, 459
hidden variables, 127
high-pass filtering, 419
histograms, 6, 154
  Bayesian blocks, 216
  comparison of methods, 214
  errors, 156
  Freedman–Diaconis rule, 155
  Knuth's method, 213
  optimal choice of bin size, 6
  Scott's rule, 155
homoscedastic errors, 7, 455
  Gaussian, 401, 422
HTF09, *see The Elements of Statistical Learning: Data Mining, Inference, and Prediction*
Hubble, Edwin, 353
hypersphere, 282
hypothesis testing, 72, 115, 135, 358, 400
  multiple, 137

independent component analysis (ICA), 304
inference, 4
  Bayesian, *see* Bayesian inference
  classical, 67
  statistical, 115
    types of, 115
information content, 376
information gain, 376
Information Theory, Inference, and Learning Algorithms, 10
installing AstroML, 35
interpolation, 407, 496
interquartile range, 75
intrinsic dimension, 60
IsoMap, 300
isometric mapping, 300
IVOA (International Virtual Observatory Alliance), 11

jackknife, 132
Jay03, *see Probability Theory: The Logic of Science*
Jeffreys, Harold, 165

K nearest-neighbors, *see* clustering
Kaiser's rule, 292
Kalman filters, 460
Karhunen–Loéve transform, 284
Karpathia, 122
kernel density estimation (KDE), *see* density estimation
kernel discriminant analysis, 365, 366, 393, 395
kernel regression, 47, 327, 366
knowledge discovery, 3
Kullback–Leibler divergence, 173, 233, 376
kurtosis, 74

Lagrangian multipliers, 172, 286
Landy–Szalay estimator, 272
Laplace smoothing, 359
Laplace, Pierre-Simon, 165
Laser Interferometric Gravitational Observatory (LIGO), 15, 399, 412
LASSO regression, 46, 324
learning curves, 345
leptokurtic, 75
LEV diagram, 292
Levenberg–Marquardt algorithm, 329
light curves, 5, 400
LIGO, *see* Laser Interferometric Gravitational Observatory
likelihood, 117
LINEAR, 15
linear algebraic problems, 44
LINEAR data set, *see* data sets
linear discriminant analysis (LDA), 362, 363, 368, 393
locality, 46
locally linear embedding (LLE), 3, 298
locally linear regression, 328
location parameter, 74
loess method, 329
logistic regression, *see* classification
Lomb–Scargle, *see* periodograms
loss function, 334, 355
lossy compression, 294
low signal-to-noise ratio, 459
low-pass filters, 417
lowess method, 329
luminosity distribution, 4
luminosity functions
  $1/V_{\max}$ method, 158
  C^{-} method, 158
  Bayesian approach, 162
  estimation, 157
Lup93, *see Statistics in Theory and Practice*
Lutz–Kelker bias, 180
Lynden-Bell's C^{-} method, 159

machine learning, 3, 4, 8
magic functions, 49
magnitudes, 509
  astronomical, 73
  standard systems, 510
Mahalanobis distance, 362, 366
Malmquist bias, 180
manifold learning, 45, 297
  weaknesses, 303
MAP, 169, 424, 434
MapReduce, 47
Markov chain Monte Carlo (MCMC), 44, 219, 446, 448
  detailed balance condition, 219
  `emcee` package, 222
  Metropolis–Hastings algorithm, 219, 329
  PyMC3 package, 221

Markov chains, 460
matched filters, 414, 447, 448, 459
maximum likelihood, *see* estimator
maximum likelihood estimation, 359
McGrayne, Sharon Bertsch, 165
mean, 44
mean deviation, 75
mean integrated square error (MISE), 123
median, 75
  standard error, 80
memoization, 45
Miller, George, 353
minimum component filtering, 420
minimum string length (MSL), 436
minimum detectable amplitude, 401
minimum variance bound, 78
misclassification rate, 355
mixtures of Gaussians, *see* Gaussian mixture model (GMM)
mode, 74
model comparison, 125
model parameters, 8
model selection, 72, 394, 447
models
  Bayesian, 44
  Gaussian mixtures, *see* Gaussian mixture model (GMM)
  hieararchical Bayesian, 174, 228
  non-Gaussian mixtures, 132
  state-space, 460
Modern Statistical Methods for Astronomy With R Applications (FB2012), 10, 432, 453, 457
Monte Carlo, 217
  samples, 112
Monty Hall problem, 69
morphological classification of galaxies, 353
multidimensional color space, 5
multidimensional scaling framework (MDS), 300
multiple harmonic model, 433
MythBusters, 70

N-body problems, 44, 51
Nadaraya–Watson regression, 327
naive Bayes, *see* Bayesian inference
nearest neighbor, 45, 47
  all-nearest-neighbor search, 52
  approximate methods, 60
  bichromatic case, 52
  monochromatic case, 52
  nearest-neighbor distance, 55
  nearest-neighbor search, 51
neural networks, 393–395
no free lunch theorem, 393
nonlinear regression, 329
nonnegative matrix factorization (NMF), 295
nonparametric bootstrap resampling, 432
nonparametric method, 6
nonparametric models, 4, 6
nonuniformly sampled data, 409
null hypothesis, 136
number of neighbors, 366
Numerical Recipes: The Art of Scientific Computing (NumRec), 8, 48, 112, 113, 127, 133, 143, 147, 152, 404, 410, 413, 417, 419, 430, 431
NumRec, *see Numerical Recipes: The Art of Scientific Computing*
Nyquist
  frequency, 411, 431, 516
  limit, 417
  Nyquist–Shannon theorem, 407
  sampling theorem, 407, 515

$\mathcal{O}(N)$, 43
Occam's razor, 178
online learning, 46
optical curve, 442
optimization, 44, 495
Ornstein–Uhlenbeck process, 458
outliers, 75, 78
overfitting, 367, 378

p value, 136
parallel computing, 47
parallelism, 47
parameter estimation, 401, 447
  deterministic models, 402
parametric methods, 6, 394
Pareto distribution, 454
Parseval's theorem, 405
Pattern Recognition and Machine Learning, 10
pdf, 6
periodic models, 401
periodic time series, 420
periodic variability, 459
periodicity, 429
periodograms, 425, 436, 439, 443
  definition of, 425
  generalized Lomb–Scargle, 433
  Lomb–Scargle periodogram, 421, 425, 429–431, 433, 437, 438, 444, 459
  noise, 426
phase dispersion minimization (PDM), 436
phased light curves, 436
photometric redshifts, 354, 377, 383
pink noise, 405, 454
platykurtic, 75
point estimation, 115
population pdf, 6, 7
population statistics, 74
power spectrum, 403, 405, 425, 448
  estimation, 411
Practical Statistics for Astronomers (WJ03), 9, 65, 420
precision, *see* efficiency
prediction, 4
principal axes, 104
principal component analysis (PCA), 3, 47, 284, 439
  missing data, 293

principal component regression, 326
probability, 66
  axioms, 66
    Cox, 67
    Kolmogorov, 66
  conditional, 67, 68
  density function, 67
  law of total, 67, 68
  notation, 65
  random variable, 5
  sum rule, 66
probability density, 356
probability density functions, 6
probability distribution, 6, 41
probability mass function, 6
Probability Theory: The Logic of Science (Jay03), 9, 67, 171
programming languages
  Python, 467
  C, 501
  C++, 501
  Fortran, 35, 501
  IDL, 35
  Python, 12
  R, 10
  SQL (Structured Query Language), 14–16, 42, 49, 51, 513
    where, 17
projection pursuit, 3, 305
PSD, *see* power spectrum, 453
Python
  AstroML, *see* AstroML
  further references, 502
  installation, 470
  introduction, 467
  IPython, 469, 482
    documentation, 483
    magic functions, 483
  Matplotlib, 469, 489
  NumPy, 468, 484, 492
    efficient coding, 497
  scientific computing, 468
  SciPy, 468, 492
  tutorial, 470
  wrapping compiled code, 501

quadratic discriminant analysis (QDA), 363, 393
quadratic programming, 370
quantile, 75
  function, 6
  standard error, 79
quartile, 75
quasar, 5
quasar variability, 453, 455, 458, 459
quicksort, 49

random forests, 377, 378, 393, 395
random number generation, 112
random walk, 444, 454, 456, 457
rank error, 60
Rayleigh test, 443
RDBMS, *see* relational database management system
recall, see completeness, 356
recall rate, 138
receiver operating characteristic (ROC) curve, 138, 391
red noise, 405, 454
regression, 4, 311
  Bayesian outlier methods, 335
  comparison of methods, 349
  cross-validation, 343
    K-fold, 348
    leave-one-out, 348
    random subset, 348
    twofold, 348
  design matrix, 317
  formulation, 312
  Gaussian basis functions, 321
  Gaussian process, 337
  Gaussian vs. Poissonian likelihood, 202
  Kendall method, 333
  kernel, 327
  LASSO, 324
  learning curves, 345
  least absolute value, 333
  least angle, 325
  linear models, 315
  local polynomial, 329
  locally linear, 328
  M estimators, 333
  maximum likelihood solution, 317
  method of least squares, 316
  multivariate, 319
  nonlinear, 329
  overfitting, 341
  polynomial, 320
  principal component, 326
  regularization, 321
  ridge, 322
  robust to outliers, 332
  sigma clipping, 333
  Theil–Sen method, 333
  toward the mean, 142
  uncertainties in the data, 331
  underfitting, 341
regression function, 357
regularization, 321
  LASSO regression, 324
  ridge regression, 322
  Tikhonov, 322
relational database management system (RDBMS), 42
relative error, 60
resolution, 407
responsibility, 128
ridge regression, 322
ridge regularization, 371
right ascension, 16, 19
risk, 355
robustness, 75
runtime, 43

sample contamination, 401
sample selection, 4
sample size, 7
sample statistics, 74, 77
sampling, 47
  window, 409
  window function, 408
Savitzky–Golay filter, 419
scale parameter, 74
scatter, 7
SciDB, 42
Scott's rule, 155
scree plot, 289
SDSS Great Wall, 244, 249, 268
searching and sorting, 48, 49
SEGUE Stellar Parameters Catalog, 354
selection effects, 157
selection function, 8
self-similar classes, 5
sensitivity, *see* completeness
Shannon interpolation formula, 408

shape parameter, 74
significance level, 136
Simon, Herbert, 353
sinc-shifting, 408
sine wave, 411
single harmonic model, 401, 421, 428, 430, 433, 459
single-valued quantity, 7
singular value decomposition, 286, 326
singular vectors, 287
Siv06, *see Data Analysis: A Bayesian Tutorial*
skewness, 74
Sloan Digital Sky Survey (SDSS), 14, 244
  Catalog Archive Server (CAS), 15
    CasJobs, 17
    PhotoObjAll, 17
    PhotoTag, 17
    Schema Browser, 17
  Data Release 7, 14
  Data Release 8, 21
  Data Release 9, 24
  flags, 17
  magnitudes
    model magnitudes, 17
    Petrosian magnitudes, 21
    PSF magnitudes, 17
  object types, 17
  SEGUE Stellar Parameters Pipeline, 24
  spectroscopic follow-up, 15
  Stripe 82, 15, 30, 361
Sobolev space, 154
software packages
  Altair, 470
  AstroML, 12, 505
  Astropy, 13
  Bokeh, 470
  HealPy, 13
  IPython, 14, 50, 469
  Markov chain Monte Carlo, 13
  Matplotlib, 12, 469
  MayaVi, 470
  NetworkX, 470
  Numerical Python, 12, 17, 468
  Pandas, 470
  PyMC3, 13
  Python, 12, 503
  Scientific Python, 12, 468
  scikit-image, 470
  Scikit-learn, 12, 470
  statsmodels, 470
  SymPy, 470
sorting, 49
specific flux, 509
spectral window function, 409
spherical coordinate systems, 33
spherical harmonics, 35
standard deviation, 7, 74
state-space models, 460
stationary signal, 447
statistically independent, 8
Statistics in Theory and Practice (Lup93), 9, 35, 65, 75, 79, 80, 98, 109, 110, 119, 132, 134, 166, 197
Statistics: A Guide to the Use of Statistical Methods in the Physical Sciences (Bar89), 9, 65
stochastic programming, 46
stochastic time series, 453
stochastic variability, 449
streaming, 46
structure function, 452, 454, 455
sufficient statistics, 187
sum of sinusoids, 402
supervised classification, *see* classification
supervised learning, 4
support vector machines, 46, 370, 371, 393, 395
support vectors, 370
SWIG, *see* Python/wrapping compiled code
SX Phe, 441

telescope diffraction pattern, 405
temporal correlation, 400
tests
  Anderson–Darling, 145, 148
  F, 153
  Fasano and Franceschini, 147
  Kolmogorov–Smirnov, 142
  Kuiper, 144
  Mann–Whitney–Wilcoxon, 146
  non-Gaussianity, 148
  nonparametric, 142
  parametric, 151
  power, 136
  Shapiro–Wilk, 149
  t, 151
  U, 146
  Welch's *t*, 152
  Wilcoxon rank-sum, 146
  Wilcoxon signed-rank, 146
The Elements of Statistical Learning: Data Mining, Inference, and Prediction (HTF09), 9, 126, 128, 129, 132, 138, 171
"The magical number 7±2", 353
The Visual Display of Quantitative Information, 29
time series, 399, 402, 452, 453
  comparison of methods, 459
top-hat, 413
total least squares, 332
training sample, 5
tree traversal patterns, 366
tricubic kernel, 328
trigonometric basis functions, 413
Two Micron All Sky Survey (2MASS), 15
type I and II errors, 136, 356

uncertainty distribution, 7
uneven sampling, 459
unevenly sampled data, 455
uniformly sampled data, 407
unsupervised classification, *see* classification
unsupervised clustering, *see* clustering
unsupervised learning, 4
Utopia, 122

variability, 400
variable

categorical, 359
continuous, 359
random, 67
continuous, 67
discrete, 67
independent, 67
independent identically distributed, 67
transformation, 72
variance, 44, 74
of a well-sampled time series, 401
variogram, 453
vectorized, 53
Voronoi tessellation, 366
vos Savant, Marilyn, 70

Wass10, *see All of Nonparametric Statistics* and *All of Statistics: A Concise Course in Statistical Inference*
wavelets, 413, 449
Daubechies, 413
discrete wavelet transform (DWT), 413
Haar, 413
Mexican hat, 413
Morlet, 414
PyWavelets, 413
wavelet PSD, 413, 415, 416
weave, *see* Python/wrapping compiled code
Welch's method, 412
whitening, 289
Whittaker–Shannon, 408
width parameter, 74
Wiener filter, 417, 419
Wiener–Khinchin theorem, 455, 457
Wiener-Khinchin theorem, 452
WJ03, *see Practical Statistics for Astronomers*
WMAP cosmology, 161
WSAS, *see Advances in Machine Learning and Data Mining for Astronomy*

zero-one loss, 355