Probability and Statistics for Computer Science

David Forsyth

Probability and Statistics for Computer Science

Springer

David Forsyth
Computer Science Department
University of Illinois at Urbana Champaign
Urbana, IL, USA

ISBN 978-3-319-64409-7 ISBN 978-3-319-64410-3 (eBook)
https://doi.org/10.1007/978-3-319-64410-3

Library of Congress Control Number: 2017950289

Printed on acid-free paper

This Springer imprint is published by Springer Nature
The registered company is Springer International Publishing AG
The registered company address is: Gewerbestrasse 11, 6330 Cham, Switzerland

To my family

Preface

An understanding of probability and statistics is an essential tool for a modern computer scientist. If your tastes run to theory, then you need to know a lot of probability (e.g., to understand randomized algorithms, to understand the probabilistic method in graph theory, to understand a lot of work on approximation, and so on) and at least enough statistics to bluff successfully on occasion. If your tastes run to the practical, you will find yourself constantly raiding the larder of statistical techniques (particularly classification, clustering, and regression). For example, much of modern artificial intelligence is built on clever pirating of statistical ideas. As another example, thinking about statistical inference for gigantic datasets has had a tremendous influence on how people build modern computer systems.

Computer science undergraduates traditionally are required to take either a course in probability, typically taught by the math department, or a course in statistics, typically taught by the statistics department. A curriculum committee in my department decided that the curricula of these courses could do with some revision. So I taught a trial version of a course, for which I wrote notes; these notes became this book. There is no new fact about probability or statistics here, but the selection of topics is my own; I think it's quite different from what one sees in other books.

The key principle in choosing what to write about was to cover the ideas in probability and statistics that I thought every computer science undergraduate student should have seen, whatever their chosen specialty or career. This means the book is broad and coverage of many areas is shallow. I think that's fine, because my purpose is to ensure that all have seen enough to know that, say, firing up a classification package will make many problems go away. So I've covered enough to get you started and to get you to realize that it's worth knowing more.

The notes I wrote have been useful to graduate students as well. In my experience, many learned some or all of this material without realizing how useful it was and then forgot it. If this happened to you, I hope the book is a stimulus to your memory. You really should have a grasp of all of this material. You might need to know more, but you certainly shouldn't know less.

Reading and Teaching This Book

I wrote this book to be taught, or read, by starting at the beginning and proceeding to the end. Different instructors or readers may have different needs, and so I sketch some pointers to what can be omitted below.

Describing Datasets

This part covers:

- Various descriptive statistics (mean, standard deviation, variance) and visualization methods for 1D datasets
- Scatter plots, correlation, and prediction for 2D datasets

Most people will have seen some, but not all, of this material. In my experience, it takes some time for people to really internalize just how useful it is to make pictures of datasets. I've tried to emphasize this point strongly by investigating a variety of datasets in worked examples. When I teach this material, I move through these chapters slowly and carefully.

Probability

This part covers:

- Discrete probability, developed fairly formally
- Conditional probability, with a particular emphasis on examples, because people find this topic counterintuitive
- Random variables and expectations
- Just a little continuous probability (probability density functions and how to interpret them)
- Markov's inequality, Chebyshev's inequality, and the weak law of large numbers
- A selection of facts about an assortment of useful probability distributions
- The normal approximation to a binomial distribution with large N

I've been quite careful developing discrete probability fairly formally. Most people find conditional probability counterintuitive (or, at least, behave as if they do—you can still start a fight with the Monty Hall problem), and so I've used a number of (sometimes startling) examples to emphasize how useful it is to tread carefully here. In my experience, worked examples help learning, but I found that too many worked examples in any one section could become distracting, so there's an entire section of extra worked examples. You can't omit anything here, except perhaps the extra worked examples.

The chapter on random variables largely contains routine material, but there I've covered Markov's inequality, Chebyshev's inequality, and the weak law of large numbers. In my experience, computer science undergraduates find simulation absolutely natural (why do sums when you can write a program?) and enjoy the weak law as a license to do what they would do anyway. You could omit the inequalities and just describe the weak law, though most students run into the inequalities in later theory courses; the experience is usually happier if they've seen them once before.

The chapter on useful probability distributions again largely contains routine material. When I teach this course, I skim through the chapter fairly fast and rely on students reading the chapter. However, there is a detailed discussion of a normal approximation to a binomial distribution with large N. In my experience, no one enjoys the derivation, but you should know the approximation is available, and roughly how it works. I lecture this topic in some detail, mainly by giving examples.

Inference

This part covers:

- Samples and populations
- Confidence intervals for sampled estimates of population means
- Statistical significance, including t-tests, F-tests, and χ^2-tests
- Very simple experimental design, including one-way and two-way experiments
- ANOVA for experiments
- Maximum likelihood inference
- Simple Bayesian inference
- A very brief discussion of filtering

The material on samples covers only sampling with replacement; if you need something more complicated, this will get you started. Confidence intervals are not much liked by students, I think because the true definition is quite delicate; but getting a grasp of the general idea is useful. You really shouldn't omit these topics.

You shouldn't omit statistical significance either, though you might feel the impulse. I have never dealt with anyone who found their first encounter with statistical significance pleasurable (such a person might exist, the population being very large). But the idea is so useful and so valuable that you just have to take your medicine. Statistical significance is often seen and sometimes taught as a powerful but fundamentally mysterious apotropaic ritual. I try very hard not to do this.

I have often omitted teaching simple experimental design and ANOVA, but in retrospect this was a mistake. The ideas are straightforward and useful. There's a bit of hypocrisy involved in teaching experimental design using other people's datasets. The (correct) alternative is to force students to plan and execute experiments; there just isn't enough time in a usual course to fit this in.

Finally, you shouldn't omit maximum likelihood inference or Bayesian inference. Many people don't need to know about filtering, though.

Tools

This part covers:

- Principal component analysis
- Simple multidimensional scaling with principal coordinate analysis;
- Basic ideas in classification;
- Nearest neighbors classification;
- Naive Bayes classification;
- Classifying with a linear SVM trained with stochastic gradient descent;
- Classifying with a random forest;
- The curse of dimension;
- Agglomerative and divisive clustering;
- K-means clustering;
- Vector quantization;
- A superficial mention of the multivariate normal distribution;
- Linear regression;
- A variety of tricks to analyze and improve regressions;
- Nearest neighbors regression;
- Simple Markov chains;
- Hidden Markov models.

Most students in my institution take this course at the same time they take a linear algebra course. When I teach the course, I try and time things so they hit PCA shortly after hitting eigenvalues and eigenvectors. You shouldn't omit PCA. I lecture principal coordinate analysis very superficially, just describing what it does and why it's useful.

I've been told, often quite forcefully, you can't teach classification to undergraduates. I think you have to, and in my experience, they like it a lot. Students really respond to being taught something that is extremely useful and really easy to do. Please, please, don't omit any of this stuff.

The clustering material is quite simple and easy to teach. In my experience, the topic is a little baffling without an application. I always set a programming exercise where one must build a classifier using features derived from vector quantization. This is a great way of identifying situations where people think they understand something, but don't really. Most students find the exercise challenging, because they must use several concepts together. But most students overcome the challenges and are pleased to see the pieces intermeshing well. The discussion of the multivariate normal distribution is not much more than a mention. I don't think you could omit anything in this chapter.

The regression material is also quite simple and is also easy to teach. The main obstacle here is that students feel something more complicated must necessarily work better (and they're not the only ones). I also don't think you could omit anything in this chapter.

In my experience, computer science students find simple Markov chains natural (though they might find the notation annoying) and will suggest simulating a chain before the instructor does. The examples of using Markov chains to produce natural language (particularly Garkov and wine reviews) are wonderful fun and you really should show them in lectures. You could omit the discussion of ranking the Web. About half of each class I've dealt with has found hidden Markov models easy and natural, and the other half has been wishing the end of the semester was closer. You could omit this topic if you sense likely resistance, and have those who might find it interesting read it.

Mathematical Bits and Pieces

This is a chapter of collected mathematical facts some readers might find useful, together with some slightly deeper information on decision tree construction. Not necessary to lecture this.

Urbana, IL, USA David Forsyth

Acknowledgments

I acknowledge a wide range of intellectual debts, starting at kindergarten. Important figures in the very long list of my creditors include Gerald Alanthwaite, Mike Brady, Tom Fair, Margaret Fleck, Jitendra Malik, Joe Mundy, Jean Ponce, Mike Rodd, Charlie Rothwell, and Andrew Zisserman.

I have benefited from looking at a variety of sources, though this work really is my own. I particularly enjoyed the following books:

- *Elementary Probability*, D. Stirzaker; Cambridge University Press, 2e, 2003.
- *What is a p-value anyway? 34 Stories to Help You Actually Understand Statistics*, A. J. Vickers; Pearson, 2009.
- *Elementary Probability for Applications*, R. Durrett; Cambridge University Press, 2009.
- *Statistics*, D. Freedman, R. Pisani and R. Purves; W. W. Norton & Company, 4e, 2007.
- *Data Analysis and Graphics Using R: An Example-Based Approach*, J. Maindonald and W. J. Braun; Cambridge University Press, 2e, 2003.
- *The Nature of Statistical Learning Theory*, V. Vapnik; Springer, 1999.

A wonderful feature of modern scientific life is the willingness of people to share data on the Internet. I have roamed the Internet widely looking for datasets, and have tried to credit the makers and sharers of data accurately and fully when I use the dataset. If, by some oversight, I have left you out, please tell me and I will try and fix this. I have been particularly enthusiastic about using data from the following repositories:

- *The UC Irvine Machine Learning Repository*, at http://archive.ics.uci.edu/ml/.
- *Dr. John Rasp's Statistics Website*, at http://www2.stetson.edu/~jrasp/.
- *OzDASL: The Australasian Data and Story Library*, at http://www.statsci.org/data/.
- *The Center for Genome Dynamics, at the Jackson Laboratory*, at http://cgd.jax.org/ (which contains staggering amounts of information about mice).

I looked at Wikipedia regularly when preparing this manuscript, and I've pointed readers to neat stories there when they're relevant. I don't think one could learn the material in this book by reading Wikipedia, but it's been tremendously helpful in restoring ideas that I have mislaid, mangled, or simply forgotten.

Typos spotted by Han Chen (numerous!), Henry Lin (numerous!), Eric Huber, Brian Lunt, Yusuf Sobh, and Scott Walters. Some names might be missing due to poor record-keeping on my part; I apologize. Jian Peng and Paris Smaragdis taught courses from versions of these notes and improved them by detailed comments, suggestions, and typo lists. TAs for this course have helped improve the notes. Thanks to Minje Kim, Henry Lin, Zicheng Liao, Karthik Ramaswamy, Saurabh Singh, Michael Sittig, Nikita Spirin, and Daphne Tsatsoulis. TAs for related classes have also helped improve the notes. Thanks to Tanmay Gangwani, Sili Hui, Ayush Jain, Maghav Kumar, Jiajun Lu, Jason Rock, Daeyun Shin, Mariya Vasileva, and Anirud Yadav.

I have benefited hugely from reviews organized by the publisher. Reviewers made many extremely helpful suggestions, which I have tried to adopt; among many other things, the current material on inference is the product of a complete overhaul recommended by a reviewer. Reviewers were anonymous to me at time of review, but their names were later revealed so I can thank them by name. Thanks to:

University of Texas, Arlington	Dr. Ashis Biswas
University of California, Davis	Dr. Dipak Ghosal
St. Louis University	James Mixco
University of Tulsa	Sabrina Ripp
University of Rhode Island	Catherine Robinson
Morgan State University	Dr. Eric Sakk
University of Texas, Dallas	Dr. William Semper

Remaining typos, errors, howlers, infelicities, cliché, slang, jargon, cant, platitude, attitude, inaccuracy, fatuousness, etc., are all my fault: Sorry.

Contents

About the Author

David Forsyth grew up in Cape Town. He received a B.Sc. (Elec. Eng.) from the University of the Witwatersrand, Johannesburg, in 1984, an M.Sc. (Elec. Eng.) from that university in 1986, and a D.Phil. from Balliol College, Oxford, in 1989. He spent 3 years on the faculty at the University of Iowa and 10 years on the faculty at the University of California at Berkeley and then moved to the University of Illinois. He served as program cochair for IEEE Computer Vision and Pattern Recognition in 2000, 2011, and 2018; general cochair for CVPR 2006 and ICCV 2019; and program cochair for the European Conference on Computer Vision 2008 and is a regular member of the program committee of all major international conferences on computer vision. He has served six terms on the SIGGRAPH program committee. In 2006, he received an IEEE technical achievement award, in 2009 he was named an IEEE Fellow, and in 2014 he was named an ACM Fellow. He served as editor in chief of IEEE TPAMI from 2014 to 2017. He is lead coauthor of *Computer Vision: A Modern Approach*, a textbook of computer vision that ran to two editions and four languages. Among a variety of odd hobbies, he is a compulsive diver, certified up to normoxic trimix level.

Notation and Conventions

A dataset is a collection of d-tuples (a d-tuple is an ordered list of d elements). Tuples differ from vectors, because we can always add and subtract vectors, but we cannot necessarily add or subtract tuples. There are always N items in any dataset. There are always d elements in each tuple in a dataset. The number of elements will be the same for every tuple in any given tuple. Sometimes we may not know the value of some elements in some tuples.

We use the same notation for a tuple and for a vector. Most of our data will be vectors. We write a vector in bold, so $\mathbf{x}$ could represent a vector or a tuple (the context will make it obvious which is intended).

The entire dataset is $\{\mathbf{x}\}$. When we need to refer to the ith data item, we write $\mathbf{x}_i$. Assume we have N data items, and we wish to make a new dataset out of them; we write the dataset made out of these items as $\{\mathbf{x}_i\}$ (the i is to suggest you are taking a set of items and making a dataset out of them). If we need to refer to the jth component of a vector $\mathbf{x}_i$, we will write $x_i^{(j)}$ (notice this isn't in bold, because it is a component, not a vector, and the j is in parentheses because it isn't a power). Vectors are always column vectors.

When I write $\{kx\}$, I mean the dataset created by taking each element of the dataset $\{x\}$ and multiplying by k; and when I write $\{x + c\}$, I mean the dataset created by taking each element of the dataset $\{x\}$ and adding c.

Terms

- $\mathsf{mean}\,(\{x\})$ is the mean of the dataset $\{x\}$ (Definition 1.1, page 7).
- $\mathsf{std}\,(\{x\})$ is the standard deviation of the dataset $\{x\}$ (Definition 1.2, page 10).
- $\mathsf{var}\,(\{x\})$ is the standard deviation of the dataset $\{x\}$ (Definition 1.3, page 13).
- $\mathsf{median}\,(\{x\})$ is the standard deviation of the dataset $\{x\}$ (Definition 1.4, page 13).
- $\mathsf{percentile}(\{x\}, k)$ is the $k\%$ percentile of the dataset $\{x\}$ (Definition 1.5, page 14).
- $\mathsf{iqr}\{x\}$ is the interquartile range of the dataset $\{x\}$ (Definition 1.7, page 15).
- $\{\hat{x}\}$ is the dataset $\{x\}$, transformed to standard coordinates (Definition 1.8, page 18).
- Standard normal data is defined in Definition 18 (page 19).
- Normal data is defined in Definition 1.10 (page 19).
- $\mathsf{corr}\,(\{(x, y)\})$ is the correlation between two components x and y of a dataset (Definition 2.1, page 39).
- $\emptyset$ is the empty set.
- Ω is the set of all possible outcomes of an experiment.
- Sets are written as $\mathcal{A}$.
- $\mathcal{A}^c$ is the complement of the set $\mathcal{A}$ (i.e., $\Omega - \mathcal{A}$).
- $\mathcal{E}$ is an event (page 341).
- $P(\{\mathcal{E}\})$ is the probability of event $\mathcal{E}$ (page 341).
- $P(\{\mathcal{E}\}|\{\mathcal{F}\})$ is the probability of event $\mathcal{E}$, conditioned on event $\mathcal{F}$ (page 341).
- $p(x)$ is the probability that random variable X will take the value x, also written as $P(\{X = x\})$ (page 341).
- $p(x, y)$ is the probability that random variable X will take the value x and random variable Y will take the value y, also written as $P(\{X = x\} \cap \{Y = y\})$ (page 341).
- $\underset{x}{\text{argmax}} f(x)$ means the value of x that maximizes $f(x)$.
- $\underset{x}{\text{argmin}} f(x)$ means the value of x that minimizes $f(x)$.
- $\max_i(f(x_i))$ means the largest value that f takes on different elements of the dataset $\{x_i\}$.
- $\hat{\theta}$ is an estimated value of a parameter θ.

Background Information

Cards: A standard deck of playing cards contains 52 cards. These cards are divided into four suits. The suits are spades and clubs (which are black) and hearts and diamonds (which are red). Each suit contains 13 cards: ace, 2, 3, 4, 5, 6, 7, 8, 9, 10, jack (sometimes called knave), queen, and king. It is common to call jack, queen, and king *court cards*.

Dice: If you look hard enough, you can obtain dice with many different numbers of sides (though I've never seen a three-sided die). We adopt the convention that the sides of an N-sided die are labeled with numbers $1 \ldots N$ and that no number is used twice. Most dice are like this.

Fairness: Each face of a fair coin or die has the same probability of landing upmost in a flip or roll.

Roulette: A roulette wheel has a collection of slots. There are 36 slots numbered with digits 1 . . . 36, and then one, two, or even three slots numbered with zero. There are no other slots. Odd-numbered slots are colored red, and even-numbered slots are colored black. Zeros are green. A ball is thrown at the wheel when it is spinning, and it bounces around and eventually falls into a slot. If the wheel is properly balanced, the ball has the same probability of falling into each slot. The number of the slot the ball falls into is said to "come up."