Machine Learning and Statistics for Internet Services

Gordon Rios
(gordon007@yahoo.com)

March 15, 2005
April 17, 2005

1 Introduction

Let's assume you're happily and successfully working as an engineer in an internet based organization: it could be a university, an online retailer, a search engine, or a small internet service provider. At some point, probably when you are putting the finishing touches on an important program, your manager will stop by and say something like: *Some of our customers are asking how their subscribers are using the site?* Later he or she might come by again and ask: *Is there any way we can predict what pages or offers each customer's users will like most?* or *Hey, are some of our customers running spam operations?!* Once people gain some visibility into how their systems are performing they often think of more and more ways to use what they've learned. The question is how to think about these kinds of problems. When faced with this kind of request most engineers will dust off their statistics textbook and start buidling some reports. But, can you do more? In short, the answer is yes. This book is filled with clear advice and useful methods that can improve your chances of success when faced with a request for analysis or modeling.

1.1 From Engineer to Scientist

Many people think that within a software or internet services organization there are two distinct camps of technical folk: engineers and scientists. Engineers are practical if conventional while scientist while away their hours deliberating over the finer points without really accomplishing much. Actually, at the best companies you will find the most successful technical people are scientifically minded engineers. The scientific mindset, available even to business folk, is that you need to test your assumptions and ideas by looking at evidence. It's really that simple - and, for machine learning evidence takes the form of data (usually stored in big log files.) Your task is to help decision makers (including yourself) form reasonable assumptions or hypotheses about what's going on, test those assumptions with the data, and then act on the findings.

1.2 To Prove or Not To Prove

It's well known that no theory or hypothesis can be *proved* it just continues to be a theory until it is *disproved*. And in your case it's pretty much the same situation: you'll need to formulate an idea and attempt to disprove it until you are satisified that it's a pretty sturdy idea and then act on it. When acting on a new idea you'll often do so in the form of a *pilot* or limited scope program. Then you use the results of the pilot study to further test and possibly refine your initial idea. Statistics is the science of reducing ideas to hypotheses that can rigorously test using data. You may set out to gather the data explicitly by laying out an *experimental design* or as is more often the case you'll simply gather up a bunch of data laying around somewhere.[1] Since you will do all your data analysis with nifty software and many custom scripts a lot of the activity uses methods from *statistical computing*. You will also be doing a lot of what is referred to as *Data Mining* which usually refers to the idea of gaining *insight* from the data – with the ultimate purpose of making better decisions.

1.3 What About Machine Learning?

If statistics is the science of figuring out important things from data then what has machine learning got to do with anything. Machine learning is a field of computer science that attempts to get computer systems to learn how to do things on their own.[2] In your setting, machine learning means building systems that automatically (or semi-automatically) generate useful information, build models, and perhaps even take actions. Seems like a tall order but if the alternative is for you to build a custom model for each of dozens or hundreds of applications by hand then you'll quickly learn to appreciate any automation you can devise.

2 The Machine Learning Project

It's helpful to think about the machine learning project as a pyramid of effort ranked by how much time and energy you should allocate to them [insert graphic showing the base of the pyramid as data management, middle is feature discovery, and top as analytics] as managing the data, discovering and extracting features of the data, and doing analytics on the features. There's no exact percentages of course but one could playfully call it an *60/30/10 rule* for the relative time you should expect to spend on each activity. A typical machine learning project is comprised of least some of these important steps.

[1] This type of data is called a *convenience sample* but I'll go into more detail on that in the sampling chapter *Collecting Data For Fun and Profit*.

[2] As opposed to the field of Artificial Intelligence (AI) which tries to create computer systems that are or seem to be *intelligent*. Some researchers categorize AI as *top down* design whereas machine learning is *bottom up* design because the systems bootstrap themselves from data. One view is that AI systems are taught how to act intelligently while machine learning systems try to learn for themselves. However, none of these distinctions is important for your success as a scientific engineer.

- Transform your manager, professor, supervisor, friend, coworker, spouse's questions into a statistical framework.
- Decide on the data and problem attributes needed to answer those questions.
- Design a way to reliably capture and process the data (perhaps to generate a report on a regular basis)
- Design and implement the actual tests or models.
- Document and describe what consumers of your work are getting and how they can use it.
- Design and deploy the production system and the maintenance plan.

It may sound like a lot but I'll break down each step into manageable pieces and discuss to what extent each step is required.

2.1 Building the Statistical Framework

In this step you will put on your hat as a consultant and dig deeper into questions people have about the system. Do they want to know about changes over time? Do they want to explain or describe a specific set of events in the past? Do they want to be able to make better decisions in the future? For example, if the questions are about network management they might want to know which users were using a particular system when it experienced a period of unacceptably high latency. In an ISP setting they may want to analyze domain and host activity to figure out if a customer is serving spam pages – further, they may want to automatically generate some kind of report on a regular basis. Essentially, you'll need to write a *story problem* that can be solved by collecting and analyzing data. In the chapter on statistical frameworks I'll outline some common patterns that can be used in a variety of settings.

2.2 Describing the Problem

You need to decide what features of the problem are most important and what data is needed to do the analysis and build the systems. Further, you'll divide the data up into two parts: the response values or elements that you care about and the input variables or descriptive attributes that provide context for understanding the response variables.[3] In machine learning the descriptive input or context variables are called *features*. For example, the response variable could be whether or not an incoming email message is spam while the input variables or features could be the text in the message or whether or not in came in from a questionable source. The tricky part is that once you put the system into production only the features can be observed – you need to provide a guess about the response value

[3]In statistics the outputs are called *dependent* variables and the inputs *independent* variables.

(i.e. is the message spam?) The features act like arguments to a magic function whose return value is a guess about the relative likelihood that the message is spam.

2.3 Collecting Data for Fun and Profit

How much data do you need? How should you collect it? Should you sample from the available data or use all of it? If you sample then how should you do it? Do you need to use *stratified sampling* or *importance sampling* or *convenience sampling*.[4] There is a huge literature on the subject of sampling but what you need to know can be covered in a single chapter. The most important concept is intuitive: as best as possible you want to take a *representative* sample of the data. You need a sample that is carefully drawn from the context you want to model (whether the purpose of the model is analysis or prediction.) In general, you'll just take random samples but you'll check to make sure there's enough data to cover the most important dimensions. For example, if your data is regional you'll want to make sure that *small but important* regions are adequately represented in the collected data. Otherwise, you might build a very successful model that fails catastrophically for a *small but important* set of cases.

2.4 Choose Your Weapon

What methods, test, or techniques should you apply to your data? Of course, this depends on the kind of questions you're asking or the kind of system you want to build. Well intentioned analysis often goes astray when it comes time to figure out what to do with the data. One of the fundamental tasks of machine learning is classification. For example, if you want to classify web pages as containing adult material or not. After data is collected you have the response variable of zero or one (for nonadult and adult) and lots of features such as structure of the page, number of images, the specific terms and phrases on the page, etc. Now which technique or method should you apply? I'll cover this question extensively over three chapters.

2.5 Explaining and Justifying What You Did

A very important step that is always required but to varying degree is the necessity of explaining how you got your results. Sometimes you'll need to explain why you chose the approach you did but often it will be more important to demonstrate your results and convince decision makers (including yourself) that these results will persist into the future. I'll cover important tools used for evaluating the performance of machine learning systems.

[4] Actually, I won't go into too much detail about *importance sampling* which is an *important* topic to be sure but probably not necessary for you to be successful in your project.

2.6 Getting Your Work Into Production

Getting the organization to use your work can be a surprisingly challenging task. Good design recommendations and exhaustive documentation of your methods and results are really helpful in this step. There are also some important patterns you can use to reduce the risk of deploying a machine learning system for your organization.

3 Outline of This Book

Each chapter will illustrate key concepts with code examples in python, session commands in *R*, or commands for popular internet packages or libraries. Graphics will be used throughout to summarize results and to show statistical relationships, probability distributions, etc. Every key point in each chapter will be illustrated with examples drawn exclusively from internet services tasks such as email filtering and categorization, web content categorization and relevance ranking, collaborative filtering and user modeling, analysis of network latency and service levels and other website transaction data.

3.1 *Introduction*

- Describe the audience of the book.
- Discuss the purpose of doing machine learning for the book's audience.
- Describe the process of developing machine learning systems.
- The Pyramid of training data, feature discovery, and analysis.

3.2 *How to Think About Your Data*

- Randomness in the data – characterizing the process.
- Random variables or stochastic processes – random variables are dimensions of the data that have characterizable uncertainty; stochastic processes are sequences of random variables. How do you determine which case you have?
- Response variables – the output data you care about.
- Input data or what you can observe before making a decision.
- Distributions of input data.
- Distributions and types of output data.
- The joint distribution of your input and output data.
- Does the distribution of your data vary over time?

- What are the dimensions of the data?
- How linear is your data? What does this mean for your analysis?
- Is your data discrete or continuous? What this means and whether or not you should transform one into the other.

3.3 *Collecting Data for Fun and Profit*

- Should you sample?
- How much data do you need?
- The representative sample and its challenges.
- Uniform sampling: collecting data from logs of unusual size.
- Convenience sampling: using what's laying around – benefits and risks.
- Stratified sampling: balancing your collection to represent important dimensions.
- Importance sampling: over- and undersampling for best effect – when to not be representative.
- Boostrap sampling: making more of what you have.
- Missing values in the data.
- How your analysis objectives affect your data collection strategy.

3.4 *Analyzing Data*

- Independent and Dependent Data: why you care.
- The magic of *conditional* independence.
- Measuring dependence: If you know X has value A, what does that tell you about the likely value of Y?
- Efficiently Calculating important measures: correlation (simple) and mutual information (more advanced).
- Reducing uncertainty – the principle of information gain.
- The internet user base provides large amounts of independent trials – ideal environment for statistical testing.
- The significance of significance testing: should you trust your statistics?

1. Testing differences in variable statistics – statistical tests and the magical p-value.
2. Testing differences in distributions – Kullback-Leibler distance.
3. Testing differences for related paired cases – often overlooked but has important implications for assessing signficance.

- Estimating the density of the data: a way to really characterize what's going on.
- Is the data changing over time? What that means for your anlysis and models. Measuring the importance and required frequency of updating your analyisis and modeling.

3.5 *Making the Best Use of Your Data*

- Feature Discovery: transforming your data into *features* that can be used for your analysis.
- Feature Engineering: designing the features of your data for maximum effectiveness (e.g. focused effects, thresholding, etc.)
- Description versus Prediction: is there a response variable? Supervised versus Unsupervised machine learning.
- Description: What can be done without a response variable?
  1. Clustering the input data into meaningful groups.
  2. Assigning input cases to groups.
  3. Analyzing relationships between variables or features and determining cause and effect.
- Prediction of Response Variables: Typed (*Categorical*), Ranked (*Ordinal*), or Continuous Values (*Cardinal*)[5]
  1. Typed variables: classification methods and Bayes Rule.
  2. Continuous valued variables: regression methods.
  3. Ranked variables: either classification or regression – why it's harder and what can be done about it.
- Detailed description of modeling different types of response data and why this is a frequent source of mistakes in many projects.

[5]Where we believe that the numerical difference between two values actually measures something we care about.

- What to do about missing values? From simple (ignore cases) to sophisticated (transduction)
- Catalogue of prototypical machine learning projects for developing and analyzing internet services data.

3.6 *Building Models for Prediction: General Issues*

- Classification of typed or categorical variables using probability models or minimizing risk of making an error.
- Binary versus Multiclass Classification
- Estimating the probability of a type given the observed inputs or features.
- Scoring versus estimating probabilities.
- Bayes Risk: what it means for you.
- Balancing data for classification and analysis.
- Reweighting cases for statistics and modeling.
- Sparse data: computational advantages, practical concerns, and when to create it.
- Over- and underfitting: key insights from statistical learning theory about the balance between model capacity and generalization error.

3.7 *Building Models for Prediction: Specific Methods*

These could easily be separate chapters for each major technique but it's conceptually helpful to introduce them together.

- Each modeling method will have example applications to classification and regression.
- Nearest Neighbor: Easy to build, assumes you can specify distance between cases, computational cost in production is high, the past is the model.
- Classification Trees: easy to build, easy to see inspect the model, good for smaller numbers of nonsparse features.
- Logistic regression: take advantage of sparse data – good for large scale modeling.
- Methods that automatically balance overfitting versus underfitting: SVM's and boosting, etc.

3.8 *Testing Models and Conclusions*

- Comparing the accuracy or performance for classification models:

    1. The Confusion Matrix
    2. The ROC Curve
    3. Precision and

3.10 *Systems that Build Models and Make Decisions Automatically*

- What about the case where data collection depends on decision making? Often my data collection depends on what I decide in any given case.
- Systems that learn to make good decisions *online* versus models trained on data gathered *offline*.
- Exploration versus Exploitation: balancing the search for better solutions with taking advantage of what you already have.
- Dynamic programming and reinforcment learning.
- Connections with automated experimental design.
- Risks and concerns with online decision making
- Barriers to adoption in large organizations.
- Internet services ideal proving ground. Payoff for organizations that use these methods (e.g. Google and Adsense)

3.11 *Going into Production*

- Presenting results of how models impact production.
- Designing pilots and experiments to measure model efficacy.
- Pattern for low risk deployment: Champion vs. Challenger pool.
- Decoupling experimentation from production without losing information.
- The production pilot: internet services and randomized user testing.
- Test and control: more significance testing using conservative assumptions.

3.12 *References*

- Libraries for Python and Perl.
- *Weka Machine Learning Project*
- *The R System* and commerical version: *S-Plus*
- *Octave* and commerical version: MATLAB
- *SVM-Light*

- Internet sources for more information (e.g. citeseer, etc.)
- Books and Journals.
- *MathWorld*.

3.13 *Appendix: The Gory Details*

- Basic math details and justifications for core results.
- A brief guide to common probability distributions.
- A look at the optimization algorithms used to build models described in the book.
- A look at the major statistical tests and details about the mathematics of using them.
- Nonparametric statistics: if computers were available when modern statistical theory was devised.
- Discussion of key statistics results useful for machine learning: central limit theorem; Markov's, Chernoff's, and Chebyshev's inequalities;