To Bayes or Not To Bayes: Model Averaging for Bayesian Classifiers

A Thesis
Presented to
The Division of Mathematics and Natural Sciences
Reed College

In Partial Fulfillment
of the Requirements for the Degree
Bachelor of Arts

Brett T. Beutell

May 2014

Approved for the Division
(Mathematics)

Irena Swanson

Acknowledgements

I want to thank a few people.

Table of Contents

Abstract

The preface pretty much says it all.

Dedication

You can have a dedication here if you wish.

Introduction

If it were not for convention, this chapter could perhaps be titled, "An Irresponsibly Quick Introduction to Bayes' Theorem." It is wholly optional for the reader who has taken an undergraduate course in mathematical statistics. This chapter introduces the ideas, terminology, and some historical minutiae about Bayesian reasoning. Naturally, we provide a statement of Bayes' theorem for events and distributions, and we explore its meaning in an inferential framework.

0.1 Much Ado About Bayes

At its inception, Bayes' theorem was not particularly controversial. Pierre Simon LaPlace offered its first formalization in the early 19th century, and he believed that with a sufficient amount of data, the answers it provided were equivalent to those obtained from his frequency based methods [McGrayne, 2011]. (This belief was ultimately proven wrong in the mid-twentieth century.) As time passed, LaPlace preferred his frequentist techniques for the relative ease of their calculations, but he did not outright condemn the use of Bayes in practice. Over a half-century after his death, LaPlace's failure to disown Bayesian reasoning caught the attention of a Scottish mathematician named George Chrystal, who remarked, "The indiscretions of great men should be quietly allowed to be forgotten" [McGrayne, 2011].

Chrystal's comment represents the attitude of early twentieth century statisticians quite well. At this time, frequentism reigned supreme, and the attitude towards Bayes' theorem and its role in statistics was surprisingly sour. For years, statisticians in the academy who employed Bayesian reasoning feared for their reputation if they explicitly made reference to Bayes' theorem in their work [McGrayne, 2011].

Fortunately, with due thanks to the computational advances of the past three decades, Bayes survived its temporary relegation to the catacombs of statistics, and it has emerged as an astoundingly useful tool for the modern statistician. The phrase "Bayesian Inference" fails to evoke the religious fervor it once did. Furthermore, it is now far more commonplace for statisticians to use both Bayesian and frequentist techniques in their work, tailoring their methods to the problem at hand [Liu et al, 2013].

Summarily, the use of Bayesian reasoning is no longer divisive or taboo, and it is a recent phenomenon that we may begin our inquiry without several pages apologizing for our inferential philosophy. With that said, it does not hurt to quickly consider the differences between a Bayesian and a frequentist.

0.2 Bayes'd and Confused

Suppose we seek an estimate of a parameter over some set of random variables. We may call said parameter θ and our random variables (or, *data*) $\vec{X}$. We use $\vec{X}$ to talk about our data in the abstract (that is, before they are observed), and we denote a particular set of observed values as $\vec{x}$.

We represent the conditional probability of θ given a set of observed data as $P(\theta|\vec{X} = \vec{x})$. Vice versa, the conditional probability of the data given a fixed $\theta = \theta_0$ can be written $P(\vec{X}|\theta = \theta_0)$. Note that for the former conditional distribution, we are fixing our data and considering θ a random variable, whereas for the latter, we are doing the converse.

"To Bayes"

Bayesians prefer the former conditional PDF. To a Bayesian, θ is uncertain, so the best means of knowing more about θ is directly through the data. Notably, the choice of words here contains an important qualifier; Bayesians want to know "*more*" about θ. Hence, they specify what is already known about the parameter. Similar to the way a mathematician or logician uses axioms, Bayesians find it important to represent in their methods what it is that they already presume to be true.

We can think of a Bayesian model's presumed truths (often called *prior beliefs*, the *prior distribution* or simply, the *prior*) as a starting point, from which data are used to *update* the subjective belief about the estimate. Bayesians use what are called *non-informative priors* to approximate a lack of prior knowledge about θ.

Besides their inherent subjectivity, the essential characteristic of Bayesian methods is that they treat parameters as random variables. Consequently, Bayesian estimates return a probability distribution, which is called the *posterior distribution*. The posterior describes the relative likelihoods of different values of θ given the data and given the *a priori* beliefs encoded in the prior distribution.

"Not To Bayes"

For the student of frequentism, the Bayesian approach may lend itself to confusion (or, years ago, anger), since frequentists treat θ as fixed ($\theta = \theta_0$), and think of their observed data as probabilistic draws that, in the long run, converge to a particular distribution.

Frequentist techniques have two main advantages. Namely, the math involved in their calculations is generally tidy, and their calculations do not require the subjective prior of Bayes' theorem. However, frequentist methods often suffer from having nuanced and unintuitive interpretations.

For instance, a frequentist p-value in a hypothesis test is a conditional probability that assumes the null hypothesis is true. Hence, it evaluates how probable the observed data would be if $\theta = \theta_0$. On the other hand, a Bayesian p-value would provide the probability of the null hypothesis being true, conditioned on the data.

To see this symbolically, consider that $P(\vec{X} = \vec{x}|\theta = \theta_0)$ is, by definition, a statement about the probability of observing $\vec{x}$, given that $\theta = \theta_0$. Frequentists are more apt to talk about the probability of seeing our data, given θ is a specific value.

Table 1: A Broad Comparison of Bayesian and Frequentist Methodologies

	To the Bayesian	**To the frequentist**
Parameters are	Distributions	Fixed-valued
Subjective assessment is	Essential	Disturbing

0.3 The First Rule of Bayes' Club

We introduce a formal, symbolic expression of Bayes' theorem:

$$P(A|B) = \frac{P(B|A)P(A)}{P(B)},$$

where P is a function from the set of all possible events, Ω, to $[0, 1]$; $A, B \in \Omega$ are events; and $P(\Omega) = 1$. We may also call Ω the *sample space*.

The proof of Bayes' theorem for events follows quickly from the axioms of probability and the definition of conditional probability. For our purposes, we note that Bayes' theorem also extends to probability distributions. Let θ be our parameter of interest and x be an observation of a random variable X. Let $\pi(\theta)$ be the prior probability distribution on θ, and let $f(X = x|\theta)$ be a density function for X. We may then express the posterior as follows:

$$\pi(\theta|X = x) = \frac{f(x|\theta)\pi(\theta)}{\int_{\theta\in\Theta} f(x|\theta)\pi(\theta)d\theta} \propto f(x|\theta)\pi(\theta),$$

where $\propto$ denotes proportionality, Θ is the set of all possible parameter values, and clearly, the denominator is equal to $f(x)$. When the domain of θ is discrete, we replace the integral in the denominator with a summation.

The above is conventionally presented alongside the pithy, assonant phrase, "The posterior is proportional to the prior times the likelihood." The numerator of Bayes' theorem is used so frequently that it has its own name, the *marginal* of the posterior. Because the denominator, also called the *normalization constant*, can be a ghastly integral, it is often more practical to reason about the posterior in terms of its marginal. For example, the denominator of Bayes' theorem disappears from consideration when we compare two possible parameter values given the data. (Not to get ahead of ourselves, but this property of proportionality is also central to running Markov Chain Monte Carlo simulations.)

0.4 Classification and Learning

A classification model is a statistical model where, given some data about an observation, we seek to predict its *class*, an unknown categorical variable. We may borrow the language of machine learning and call the predictor variables in our model *features*.

Put tersely, classification requires evaluating probability estimates in a decision theoretic framework. That is to say, we estimate the relative probabilities of some datum belonging to each possible class of the unknown categorical variable, and we have a corresponding decision rule that dictates the class to which said datum should be assigned. This all may seem rather abstract, though. So, let us get concrete.

Example: Spam Email Detection

Spam detection is an oft-cited example of classification. It is suitable for our discussion, since it is an area of classification that has benefited greatly from application of Bayes' rule. In a 1998 study, researchers at Microsoft found that filtering emails for spam based off of hard-coded rules was significantly outperformed by even the most reductive of Bayesian classifiers, the aptly-named *Naïve Bayes* [Sahami et al, 1998]. Here, we use their paper to contextualize what is meant by a classification model. We will continue with this example in the next chapter in order to help characterize Bayesian classifiers.

Suppose we have a set of $1,000$ emails, and for each email, we record four pieces of information according to the table below:

Table 2: Information on a set of 1,000 emails

Variable	**Description**
Hyperlinks	A count of the number of hyperlinks in the body of the email
Domain	The domain of the email's sender (e.g., @reed.edu)
Timestamp	The time at which the email was sent
Spam	Whether or not the email was spam

From these data, we can make predictions about whether or not future, incoming emails are spam. If our data were similar to those used by Sahami et al, we may conclude that, say, a message from @reed.edu, sent at noon and containing zero hyperlinks, is unlikely to be spam. On the other hand, an email from @ALL_UR_InBoXX-R-beLoNg-2-us.com, which was sent at midnight and contains thirty hyperlinks, would seem a little suspicious. Hopefully, our model would classify it as spam.

A Note on Learning

Our method will necessitate data from which we can *train* our classification model. That is, in order to use our model, we must first have a set of pre-classified data from which we can estimate our models' parameters. Thus, in the preceding example, we began with a set of $1,000$ emails that had already been classified as either spam or not-spam. From these initial data, we could make a judgement about new, unclassified data. Such an approach is can be described as *learning* the parameters of the model.

Chapter 1

The Bayesian Network

Bayesian Networks are a useful means of visualizing and reasoning about classification models. Put succinctly, a Bayesian Network is a directed acyclic graph (DAG) where each node represents a random variable in the model, and each edge represents a conditional dependency between two random variables.

1.1 Classification at a Glance

Suppose we have a set of random variables $V = \{C, X_1, \ldots, X_n\}$, and we seek to predict the value of C, a categorical variable, using $X_1, \ldots, X_n$. We call C the *class variable*, and we refer to the X_i as the *feature variables*. For notational convenience, we may denote the set of feature variables by $\vec{X}$, and we may denote a single set of observed values of $\vec{X}$ as the vector $\vec{x} = (X_1 = x_1, \ldots, X_n = x_n)$.

The Bayesian Approach

Let C have k possible classes. Our goal in a classification setting is to find the probabilities of each class of C, given the observed values of the feature variables. Usually (but not always), we seek the value of the class node that maximizes the equation $P(C|\vec{X})$. We appeal to Bayes' theorem:

$$P(C = c_i|\vec{X}) = \frac{P(\vec{X}|C = c_i)P(C = c_i)}{\displaystyle\sum_{c_j \in \text{dom}\{C\}} P(\vec{X}|C = c_j)P(C = c_j)},$$

where $P(\vec{X}|C = c_i)$ is our likelihood function and $P(C = c_i)$ is our prior probability for the class variable when it is equal to c_i. Notably, our problem does not require we compute the denominator of the posterior, since

$$\arg \max_{c_i \in \text{dom}\{C\}} \{P(C = c_i|\vec{X} = \vec{x})\} = \arg \max_{c_i \in \text{dom}\{C\}} \{P(\vec{X} = \vec{x}|C = c_i)P(C = c_i)\}.$$

To see why, compare the posterior distributions of two possible class values, c_i and c_j, given a vector of inputs $\vec{X}$. Assuming these classes have nonzero posterior probabilities, the normalization constant disappears from the following equation:

$$\frac{P(C=c_i|\vec{X}=\vec{x})}{P(C=c_j|\vec{X}=\vec{x})} = \frac{\frac{P(\vec{x}|c_i)P(c_i)}{P(\vec{x})}}{\frac{P(\vec{x}|c_j)P(c_j)}{P(\vec{x})}} = \frac{P(\vec{x}|c_i)P(c_i)}{P(\vec{x}|c_j)P(c_j)}.$$

Clearly, then, if we obtain a value greater than 1 from the above, we find the class c_i to be more likely than c_j, given the data. Since the normalization constant $P(\vec{x})$ can be unwieldy to compute, this is a very convenient property of Bayesian classifiers. That is, in this setting, we need only compute the marginal of the posterior for each class.

Finding the Marginal

With a few simple results from probability theory, we can refactor the numerator of the posterior. For starters, we invoke the multiplication rule:

$$P(\vec{X}|C)P(C) = P(C, X_1, \ldots, X_n).$$

Then, for convenience, we define $X_0 \equiv C$, and we apply the chain rule of probability [[APPENDIX?]] to obtain

$$P(X_0, \ldots, X_n) = P(\cap_{i=0}^n X_i) = \prod_{j=0}^{n} P(X_j | \cap_{k=0}^{j-1} X_k).$$

Thus, our likelihood function requires calculation of all of the conditional dependencies between feature variables in $\vec{X}$, and in order to accurately classify an input $\vec{x}$, we should construct a model that accounts for these conditional dependencies.

1.2 Directed Graphs

Let V be a set $\{V_0, V_1, \ldots, V_n\}$, and let E be a set of ordered pairs (V_i, V_j) on $V \times V$, (for $n, i, j \in \mathbb{Z}^+$). A *directed graph* G is defined as the tuple (V, E). We call members of V *vertices* or *nodes*, and we call members of E *edges* or *arcs*. Importantly, the order of vertices that compose a particular edge $E_i = (V_j, V_k)$ encodes the *direction* of the edge. I.e., E_i is said to be an edge *from* V_j *to* V_k, (for $i, j, k \in \mathbb{Z}^+$). Figure 1.1 provides an example of how we might illustrate a directed graph. In this context, $V = \{V_0, V_1, V_2\}$ and $E = \{(V_0, V_1), (V_0, V_2), (V_2, V_1)\}$.

Paths and Cycles

It is natural to consider the *paths* through G. A path P of length n is simply an ordered n-tuple of edges. We refer to the first and last nodes in this sequence as the *starting node* V_s and *terminal node* V_t, respectively. With the exception of V_t, if the i^{th} member of P is an edge that points *to* the node V_j, then the $(i+1)^{\text{th}}$ member is an edge pointing *from* node V_j to another node V_k. Thus, a path of length n defines a sequence of $n+1$ nodes, such that each node has an edge from itself to its successor.

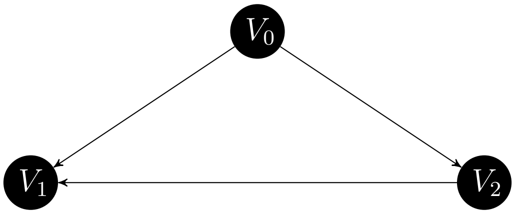

Figure 1.1: A simple directed graph with 3 nodes and 3 edges

By $\mathcal{P}(G)$, let us mean the family of all paths defined on G. For any $P \in \mathcal{P}(G)$, we call P a *cycle* if $V_s = V_t$. We say that a graph G is *cyclic* if there exists at least one cycle in $\mathcal{P}(G)$. Otherwise, we say that G is *acyclic*.

Relatives

The vocabulary used to describe the relationships amongst nodes in a graph is markedly familial. Given the nodes $V_i, V_j, V_k \in V$, we say that V_i is an *ancestor* of V_j if there exists a $P \in \mathcal{P}(G)$ such that V_i precedes V_j in the sequence of nodes defined by P. Conversely, we call V_j a *descendant* of V_i. We also give special names to a node's immediate ancestors and descendants. Formally, for V_j, we may define the set of *parents* of V_j as $\{V_i : (V_i, V_j) \in E\}$. Similarly, we define the set of *children* of V_j as $\{V_k : (V_j, V_k) \in E\}$. A node whose set of parents is empty is called the *root node* of our graph.

The Adjacency Matrix

We may represent $G = (V, E)$, where $V = \{V_0, \dots, V_{n-1}\}$ has order n, with an $n \times n$ matrix $A = (a_{ij})$, such that for each entry a_{ij},

$$a_{ij} = \begin{cases} 1, & \text{if} \quad (V_{i-1}, V_{j-1}) \in E; \\ 0, & \text{otherwise.} \end{cases}$$

We refer to A as a *binary node-node adjacency matrix* or just an *adjacency matrix*. For example, the adjacency matrix corresponding to the first figure of this section would be

$$\begin{pmatrix} 0 & 1 & 1 \\ 0 & 0 & 0 \\ 0 & 1 & 0 \end{pmatrix}$$

since there is an arc from V_0 to V_1, an arc from V_0 to V_2, and an arc from V_2 to V_1. We may also consider the sum of powers of A,

$$Y = A + A^2 + \cdots + A^n = \sum_{i=1}^{n} A^i.$$

The resulting matrix is (y_{ij}), where y_{ij} represents the number of unique paths from $V_s \equiv V_{i-1}$ to $V_t \equiv V_{j-1}$ in $\mathcal{P}(G)$. Thus, to formalize the notion of an acyclic graph, we say that $G = (V, E)$ is acyclic if $tr(Y) = 0$. Otherwise, G is cyclic. To continue our example from above, we would calculate Y to be

$$\begin{pmatrix} 0 & 2 & 1 \\ 0 & 0 & 0 \\ 0 & 1 & 0 \end{pmatrix}$$

which is consistent with our visualization of the graph. (For example, we identify two distinct paths from V_0 to V_1, and $y_{1,2} = 2$.)

1.3 Bayesian Networks

Let $\mathcal{B} \equiv \langle G, \Theta \rangle$, where $G = (V, E)$ is a directed acyclic graph, $V = \{V_0, V_1, \ldots, V_n\}$ is a set of $n + 1$ random variables, and Θ is a set of parameter estimates corresponding to the structure of G. We say that $\mathcal{B}$ is a Bayesian Network.

Restricted Bayesian Networks

A *restricted* Bayesian network caps the number of possible conditional dependencies for a given feature variable at some nonnegative integer k. Hence, the graph that represents the network should have at most k arcs from each of its feature nodes. The Tree Augmented Network (TAN), displayed below, is an example of a restricted Bayesian Network with $k = 1$.

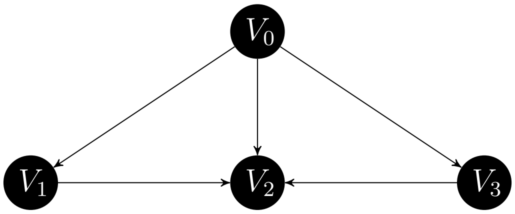

Figure 1.2: TAN with three features

Of course, a Bayesian network without restrictions on its nodes' conditional dependencies is called an *unrestricted* Bayesian network.

1.4 Bayesian Classifiers

A Bayesian classifier is simply a Bayesian network applied to a classification problem. We consider the necessary structural and decision theoretic modifications to $\mathcal{B}$ that are necessary for this application.

Networks Constraints

In a classification setting, we place a few extra constraints on the graph of $\mathcal{B}$. Namely, we may call V_0 the class variable, and we may require that the binary node-node adjacency matrix A of G satisfies the following properties:

$$\sum_{i=1}^{n} a_{i1} = 0, \quad \text{and} \quad \sum_{j=1}^{n} a_{1j} \neq 0.$$

In words, we mean there are no directed edges pointing to the class node, and the class node has an edge to at least one feature node. Obviously, the class node is the root of G.

Making a Decision

To predict the class for an observation, we use a *decision rule.* Our decision rule provides a means of selecting one proposed classification over another, given a set of pre-classified data and a new input that has yet to be classified. Recall that in surveying the general problem of classification, we chose a decision rule that returned the class maximizing our posterior probability. Albeit common, this is not the only possible decision rule. For instance, there may be an unequal cost associated with certain misclassifications of $\vec{x}$. In such cases, we can modify our decision rule's *loss function* accordingly; however, this facet of decision theory is not central to our discussion.

The Naive Bayes Classifier

The Naïve Bayes classifier (NBC) is highly reductive; yet, it is a surprisingly effective means of classifying data. In essence, we make a simplifying assumption about our data in order to yield a more wieldy computation of the posterior. Recall that the posterior distribution of the class variable $C \equiv X_0$ has the following property:

$$P(X_0|X_1, \ldots, X_n) \propto P(X_0, \ldots, X_n) = P(\cap_{i=0}^{n} X_i) = \prod_{j=0}^{n} P(X_j| \cap_{k=0}^{j-1} X_k).$$

If we assume independence amongst the feature variables $X_1, \ldots, X_n$, then the above simplifies to

$$P(X_0) \prod_{j=1}^{n} P(X_j| \cap_{k=0}^{j-1} X_k) = P(C) \prod_{j=1}^{n} P(X_j|C).$$

Note that our independence assumption makes the NBC's graph a restricted Bayesian network with $k = 0$.

Example: Spam Detection

Let us continue the classification example from section 0.4. Recall that we are given a set of 1,000 emails, pre-classified as either spam or not-spam. Further, for each email, we have information on the sender's domain, the number of hyperlinks in the body of the email, and the time at which the email was sent.

It is generally easier to learn the parameters of a Bayesian network when the feature variables are categorical. Thus, we might use the domain information to create a variable that indicates whether or not the sender's domain ended in ".edu." For the count of hyperlinks, we might define categories of Low (0 to 4 hyperlinks), Medium (5 to 10 hyperlinks), and High (more than 10 hyperlinks). Using the timestamp of the emails, we might create an indicator for whether or not an email was sent between midnight and noon. In the end, we could assign our variables to nodes in a graph as follows:

Table 1.1: Variables from the training set of 1,000 emails

Node	Variable	Description
V_0	Spam	1 if Spam; 0 otherwise
V_1	edu	1 if from .edu; 0 otherwise
V_2	Hyperlink Count	Low, Medium, or High
V_3	Time	1 if sent between midnight and noon; 0 otherwise.

Our Naïve Bayes classifier would have a graph with the structure depicted in Figure 1.3. Of course, to complete our classifier, we would learn our model parameters from the training set.

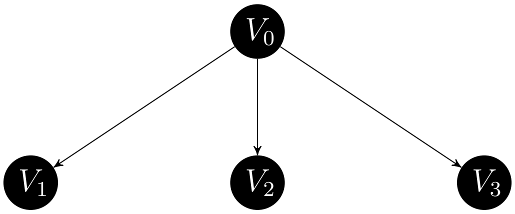

Figure 1.3: A Naive Bayes Classifier

1.5 Model Structure

Unless we have overwhelming prior knowledge that tells us to use a specific graph, we need a means of finding the structure for our Bayesian classifier's network. Sadly, though, the size of the space of all possible network structures is *super-exponential* on the number of nodes in a network's graph. Specifically, if n is the number of nodes in a graph, the structure space increases at $2^{O(n^2 \log n)}$ [Koeller, 2001]. Thus, when we have a large number of variables in our model, restricting the number of conditional dependencies in our model is sometimes desirable, as it reduces the size of the structure space. Most importantly, though, we may choose between two modes of reasoning about model structure.

Model Selection

The first, called *model selection*, involves finding the most probable model structure given our data, which we can then use to estimate our parameters. Generally, this would involve cleverly searching through the space of possible models (or a subset thereof) and ranking structures according to a score function. The structure with the highest score would be our most likely model, and we then would use it to estimate our parameters.

Model Averaging

The second technique, called *model averaging* is a little more involved. In lieu of selecting one highly likely model, model averaging would have us make estimates across a space of possible network structures, which are each weighted by their likelihood. Hence, model averaging is a useful tool when we have several model structures that are roughly equiprobable. There are many ways to go about model averaging, and we shall get to one of them in due time. For now, though, it suffices to know that model averaging is computationally very difficult, so we require a means to approximate it by simulation.

Chapter 2

Monte Carlo with Markov Chains

The reader may be familiar with Good Old Fashioned Monte Carlo (GOFMC) methods involving independent and identically distributed (IID) data. However, Monte Carlo simulations can also be done with a surprisingly simple stochastic process called Markov Chains. In fact, these Markov Chain Monte Carlo (MCMC) techniques are quite useful for approximating draws from high-dimensional posterior distributions for which we cannot easily find the normalization constant.

This chapter provides a cursory overview of GOFMC, and then defines Markov chains with the intent of introducing a class of algorithms for Markov Chain Monte Carlo simulation.

2.1 GOFMC

An Intuitive Example

Imagine we seek to find the area of a peculiar two-dimensional shape called Minnesota. We are given no general formula for its area, and all of our attempts to analytically

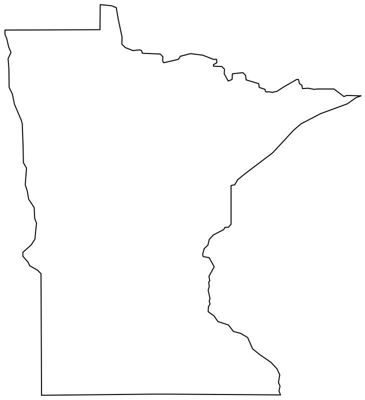

Figure 2.1: A Two-Dimensional Shape Called Minnesota

represent the curve that traces its perimeter have proven themselves fruitless. With credit to a talk on Monte Carlo Tree Search given by Peter Drake at the University of Portland, the technique suggested by GOFMC would be the following:

1. Place Minnesota inside a square with edges of known length s.

2. Randomly throw n darts such that they land inside the square.

3. Record x, the number of darts that landed inside Minnesota.

4. Multiply the proportion of darts that hit Minnesota ($\frac{x}{n}$) by the area of the square.

Symbolically, our GOFMC estimator for the area would be

$$s^2 * \sum_{j=1}^{n} \frac{\iota(x_j)}{n},$$

where the x_j represent dart-throws, and

$$\iota(x_j) = \begin{cases} 1, & \text{if the dart hit Minnesota;} \\ 0, & \text{otherwise.} \end{cases}$$

The following illustrates our dart-throwing technique from above:

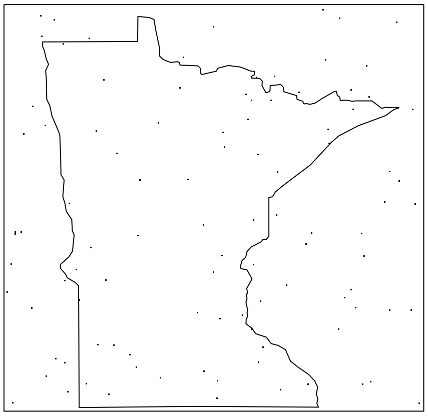

Figure 2.2: Estimating the Area of an Oddly Shaped State

Obviously, our estimate is not exact, but as n approaches infinity, the Law of Large Numbers tells us that we get closer and closer to the true area of Minnesota. Thus is the motivation for GOFMC.

The General Case

Suppose we are given a distribution π and we seek to find the expectation of a function f over π. The central idea of GOFMC is that we may generate IID random variables $X_1, \ldots, X_n$ from π in order to estimate $E[f(X_i)]$. A logical estimator, then, by simple application of the method of moments, is

$$\frac{1}{n}\sum_{i=1}^{n} f(X_i).$$

Of course, with an estimate comes the corresponding notion of its error. In this case, we define the GOFMC error as

$$\epsilon = E[f(X_i)] - \frac{1}{n}\sum_{i=1}^{n} f(X_i).$$

By application of the Central Limit Theorem, we know that as $n \to \infty$, the above converges to a normal distribution with a mean $\mu = 0$ and variance inversely proportional to n. Hence, with large enough n, we can produce fairly accurate estimates from our randomly generated data, and we can have an idea of how erroneous our estimates are.

2.2 Markov Chains

Andrey Markov used his eponymous chain only once in practice, to analyze the occurrence of vowels in a Poeshkin poem (3). According to legend, Enrico Fermi ran Markov chains in his head to combat insomnia, but we normal humans tend to reserve such calculations for computers. Thankfully, though, the intuition behind Markov chains is fairly easy to grasp.

Definition

Let s_i be a state from a set of possible states (the *state space*) $\mathcal{S}$, and let I be an index set. What we call a *Markov chain* $\theta^{(i)}$ is a collection of states from $\mathcal{S}$ indexed by $i \in I$. Markov chains can be thought of as a sequence of probabilistic transitions from state to state, where the probability of transitioning to the next state in the chain depends solely on the current state.

Thus, Markov chains have associated *transition probabilities*. When $\mathcal{S}$ is discrete and has finite order n, we may specify these probabilities in an n by n *transition matrix*, $T = (t_{jk})$, where $t_{jk} = P(\theta^{i+1} = s_k|\theta^i = s_j)$. Each row of T admits a probability distribution for a corresponding state in $\mathcal{S}$.

Using T, we may define a *transition function* $p : \mathcal{S} \times \mathcal{S} \to [0, 1]$ on our Markov chain, where $p(s_j, s_k) = t_{jk}$. Consistent with our definition above, $p(s_j, s_k)$ depends solely on s_j, and not on any past or future states of the chain. For the sake of concision, we may also refer to transitions as *steps*, and when the chain assumes a state s_j, we also say that it *hits* state s_j.

Example: Andrey the Chameleon

To illustrate our definition of Markov chains, we introduce a charismatic chameleon named Andrey. Suppose that Andrey can only assume four distinct colors: blue, green, yellow, or red. Hence, Andrey's state space $\mathcal{S}$ may be represented as $\mathcal{S} = \{B, G, Y, R\}$.

We assume that Andrey has no control over the color to which he changes. Instead, the color to which Andrey changes is a probabilistic process, and it only depends on the color that he currently assumes. Andrey's transition matrix T is given by

$$\begin{array}{c|cccc} & B & G & Y & R \\ \hline B & .25 & .25 & .25 & .25 \\ G & .8 & .1 & .1 & 0 \\ Y & .5 & .3 & .02 & .18 \\ R & 1 & 0 & 0 & 0 \end{array}$$

For example, t_{2j} is the probability distribution $P(\theta^{i+1}|\theta^i = G)$. That is, if Andrey is currently green, he has an 80% chance of next turning blue, a 10% chance of remaining green, a 10% chance of turning yellow, and a 0% chance of turning red.

A possible run of Andrey's associated Markov chain for $1 \leq i \leq 3$ might look like the following:

$$(\theta^1 = \text{B}, \quad \theta^2 = \text{G}, \quad \theta^3 = \text{B}).$$

Notice, however, that it would be impossible to observe the following collection of states:

$$(\theta^1 = \text{B}, \quad \theta^2 = \text{R}, \quad \theta^3 = \text{Y}),$$

since $p(\text{R}, \text{Y}) = 0$.

Basic Properties

We say a state $s_j \in \mathcal{S}$ is *irreducible* if it is possible to get to any other $s_k \in \mathcal{S}$ from s_j. If all $s_j \in \mathcal{S}$ are irreducible, the Markov chain $\theta^{(i)}$ over $\mathcal{S}$ is also said to be irreducible.

By Z_i we denote the number of steps before a chain hits state s_i once. We refer to this quantity as the *recurrence* or *hitting time* for state s_i. We denote the number of steps before a chain hits s_i an arbitrary number of times as Z_i^q, where q is a positive integer. Clearly, Z_i^q is always an integer, and Z_i is simply shorthand for Z_i^1. We may also define $Z_i^0 = 0$.

Since Markov chains are stochastic processes, Z_i^q is a random variable. Hence, we may consider its expectation $E[Z_i^q]$. If for all $s_i \in \mathcal{S}$, the expected hitting time is finite ($E[Z_i^1] < \infty$), then we say our Markov chain is *positive recurrent.*

A Markov chain is *periodic* if, for at least one state s_i, it can only return to s_i after a number of steps equal to a multiple of some positive integer $k > 1$. Markov chains that contain no periodic states are called *aperiodic.* We say a Markov chain is *ergodic* if it is both aperiodic and positive recurrent.

Lastly, the *stationary distribution* π of a Markov chain is a PDF such that the transition matrix T defined by $\theta^{(i)}$ maintains π. Symbolically, this looks like the following:

$$\sum_{s_j \in \mathcal{S}} \pi(s_j) * p(s_j, s_k) = \sum_{s_j \in \mathcal{S}} \pi(s_j) * P(s_k|s_j) = \pi(s_k).$$

2.3 The Ergodic Theorem

The Ergodic Theorem for Markov chains is an analog of the Law of Large Numbers for IID data. It ties together the concepts of aperiodicity, positive recurrence, and the stationary distribution. Importantly, it is through a basic corollary to the Ergodic theorem that we are able to justify use of MCMC in order to estimate draws from intractable distributions.

Statement

Let $\theta^{(i)}$ be an ergodic Markov Chain. Let $V_j(n)$ be defined as follows:

$$V_j(n) = \sum_{i=0}^{n-1} \iota_j(\theta^i),$$

where

$$\iota_j(\theta^i) = \begin{cases} 1, & \text{if} \quad \theta^i = s_j; \\ 0, & \text{otherwise.} \end{cases}$$

We interpret $V_j(n)$ as the number of visits to state s_j before θ^n in our Markov chain. Let $E[Z_j] = m_j$ be the expected hitting time of state s_j. Then,

$$P\left(\frac{V_j(n)}{n} \longrightarrow \frac{1}{m_j}, \text{ as } n \to \infty\right) = 1.$$

That is, the proportion of times we hit s_j converges in probability to the inverse of the expected recurrence time.

Proof

We begin our proof by invoking a result not included herein. Namely, the proportion of times we hit s_j in the chain is the same regardless of the initial state θ^0 of the chain as n goes to ∞. Hence, we may fix s_j, and without loss of generality, we assume it is the initial state of the chain ($\theta^0 = s_j$).

We define $U_j^r = Z_j^r - Z_j^{r-1}$. Clearly, then,

$$\sum_{k=1}^{r} U_j^k = (Z_j^1 - Z_j^0) + \cdots + (Z_j^{r-1} - Z_j^{r-2}) + (Z_j^r - Z_j^{r-1}) = Z_j^r.$$

The quantities $U_j^1, \ldots, U_j^r$ are IID random variables, meaning we can find their expectation. Since we may intuitively think of U_j^r as the number of steps between the $(r-1)^{\text{th}}$ and r^{th} hits of s_j in our Markov chain, we see that for all r,

$$E[U_j^r] = E[Z_j^r - Z_j^{r-1}] = r * E[Z_j] - (r-1) * E[Z_j] = E[Z_j] = m_j,$$

which follows from the linearity of expectation. Hence, by the Strong Law of Large Numbers [[APPENDIX?]], we get:

$$P\left(\frac{\sum_{k=1}^{n} U_j^k}{n} \to m_j\right) = 1$$

Now, we write

$$\sum_{k=1}^{V_j(n)} U_j^k \leq n - 1,$$

where our sum gives us the number of steps until the last hit of s_i before θ^n in the chain. Obviously, then, the sum could not exceed $n-1$, since we are only considering the number of steps to a state that must occur before the n^{th} step. We may also consider

$$\sum_{k=1}^{V_j(n)+1} U_j^k \geq n,$$

where our summation may also be written $Z_j^{V_j(n)+1}$, which is the number of steps until the first hit of s_j after step θ^{n-1}. Thus, we may squeeze n and divide the resulting inequality by $V_j(n)$.

$$\frac{\sum_{k=1}^{V_j(n)} U_j^k}{V_j(n)} \leq \frac{n}{V_j(n)} \leq \frac{\sum_{k=1}^{V_j(n)+1} U_j^k}{V_j(n)}$$

$$.$$

Using the convergence in probability of the U_j^k to m_j, our sums become

$$\frac{V_j(n) * m_j}{V_j(n)} \leq \frac{n}{V_j(n)} \leq \frac{(V_j(n)+1) * m_j}{V_j(n)}$$

$$,$$

and as $n \to \infty$, we see $\frac{V_j(n)+1}{V_j(n)} \to 1$, giving us

$$m_j \leq \frac{n}{V_j(n)} \leq m_j.$$

Hence, as $n \to \infty$,

$$P\left(\frac{n}{V_j(n)} \longrightarrow m_j\right) = 1$$

and we have our result.

Corollary

Given a bounded function $f : \mathcal{S} \to B$,

$$P\left(\frac{1}{n}\sum_{j=0}^{n-1} f(\theta^j) = \sum_{s_k \in \mathcal{S}} \pi(s_k) f(s_k)\right) = 1, \text{ as } n \longrightarrow \infty.$$

This corollary justifies the use of Monte Carlo methods that estimate expectation of a function over simulated draws from the stationary distribution of our Markov chain.

2.4 MCMC

Recall that in the case of GOFMC, we have a distribution π from which we can easily simulate random draws, and we then estimate the expectation of a function over π through random draws. To perform GOFMC, we must know the PDF for π. Sometimes, however, we are not so lucky as to be able to fully compute the PDF of our distribution of interest. This occurs frequently in high-dimensional Bayesian analyses, where the denominator of our posterior calculation tends to towards intractability. Thus, we seek an alternative way to simulate draws from π when we only know π up to a constant. For this, we can use a Markov chain.

Metropolis Hastings

To construct a Markov chain with a stationary distribution equal to our posterior, we must first consider how to transition from state to state. First we need to ensure that the stationary distribution of our chain exists. To that end, we introduce the *detailed balance* condition. That is, for all $s_i, s_j \in \mathcal{S}$,

$$\pi(s_j)p(s_j, s_k) = \pi(s_k)p(s_k, s_j)$$

This condition is not necessary for the convergence of our chain to the posterior, but it suffices (2).

Next, we must consider the transition function p, which must guarantee that π is the unique stationary distribution of our Markov chain. To that end, we split p into a *proposal function* ($q : \mathcal{S} \to \mathcal{S}$), which randomly proposes a new state in $\mathcal{S}$, and an *acceptance rule*, which returns the probability that we accept the proposed transition.

$$p(s_j, s_k) = q(s_j, s_k)\alpha(s_j, s_k)$$

We may assume the proposal function is a random walk with symmetric error. Again, this is not wholly necessary, but it suffices for our purposes (2).

Construction of the acceptance rule is slightly more involved. Essentially, we define a rule that returns a probability of transitioning to the proposed state. Suppose we are in state s_j, and q proposes a move to state s_k. We accept the transition to state s_j with probability equal to

$$\alpha(s_j, s_k) = \begin{cases} \min\{1, \frac{\pi(s_k)q(s_j|s_k)}{\pi(s_j)q(s_k|s_j)}\}, & \textbf{if} \quad s_j \neq s_k; \\ 1 - \int q(s_j, s_k)\alpha(s_j, s_k)ds_k, & \textbf{otherwise}. \end{cases}$$

The result is a Markov chain that converges to π, our posterior distribution.

Sources of Error

We face two issues with running Markov chains to approximate the posterior. First, if we sample from the chain directly, our sample is not independent. Hence, taking all of the values from our simulation would not be an IID sample from the posterior. Secondly, the Markov chain converges to the posterior, which means in its early iterations, samples may not approximate the posterior.

To deal with these issues, we can run a *burn-in* period for the chain, and then sample every n^{th} state of the chain. These are clever heuristics to make our estimates more accurate, but there is little way to quantify how well they perform. Although we know the chain converges to the posterior, we have no theorem which will tell us how many steps in the chain we must take to get an accurate simulation.

Chapter 3

Bayesian Model Averaging for Bayesian Networks

3.1 Bayes' Theorem for Models

3.2 MCMC for BMA

Appendix A

Appendix A

A.1 The Chain Rule for Probability

References

[1] M. Sahami, S. Dumais, D. Heckerman, and E. Horvitz. A Bayesian approach to filtering junk email., AAAI Workshop on Learning for Text Categorization, July 1998, Madison, Wisconsin. AAAI Technical Report WS-98-05

[2] D. Gamerman and H. F. Lopes. *Markov Chain Monte Carlo. Boca Raton, FL: Chapman & Hall/CRC, 2006.*

[3] S.B. McGrayne. The theory that would not die: How Bayes' rule cracked the enigma code, hunted down Russian submarines, and emerged triumphant from two centuries of controversy. New Haven, CT: Yale University Press, 2011.

[4] S. Liu, M. Zhu, and Y. Yang. A Bayesian Classifier Learning Algorithm Based on Optimization Model. *Mathematical Problems in Engineering, 2013. doi: 10.1155/2013/975935.*

[5] N. Friedman and D. Koller. Being Bayesian About Network Structure. Netherlands: Kluwer Academic Publishers, 2000. URL http://robotics.stanford.edu/ koller/Papers/Friedman+Koller:MLJ03.pdf