Building Probabilistic Models from Databases

Thesis by

John W. Miller

In Partial Fulfillment of the Requirements

for the Degree of

Doctor of Philosophy

California Institute of Technology

Pasadena, California

1993

(Submitted August 26, 1992)

Acknowledgements

I'd like to thank my advisor Professor Rodney Goodman and the members of his lab for a stimulating, creative, and always fun place to work. I would also like to acknowledge the help of Padhraic Smyth; he listened to my ideas and introduced me to diverse bodies of research which have benefitted me and this thesis.

I am grateful for knowing and working with Professor Joel Franklin, whose love of mathematics and its applications has inspired me.

I am also grateful for my friends who could be persuaded to stay up late discussing matters technical, political or personal, in particular Tim Brown, Kathleen Kramer, Ramésh, and Yasantha Rajakarunanayake.

Thanks are owed to Ian Galton, who championed the cause of elegant presentation, whose advice and friendship have enriched my life at Caltech, and to Kerry Galton, who with Ian opened up their home and their lives to me.

To my parents and family, I am forever grateful for their love and support.

And finally I want to thank Danielle. She continues to be a source of encouragement, perspective, and wisdom. I appreciate the sacrifices she has made in support of this work.

Preface

Why Study Probability Estimation from a Database?

In this thesis the underlying motive is to bring computer resources to bear effectively on applications with less problem specific programming. Certain of the resources available are increasing at an exponential rate—primarily CPU power, memory, long term storage capacity, and on-line information from databases. In contrast, the resources of programming time and the time of the software user are relatively fixed in cost. Given this change in the relative costs of computer applications, there is a great incentive to use information stored in databases as a substitute for decisions made explicitly by a user or programmer.

Although probability estimation is an important problem in its own right, here it is used as a formal tool for describing and manipulating the information that is extracted from databases. The final output of systems designed using these techniques, for instance an image recognition system, may not be probabilistic at all. The focus here is on the probability estimates, rather than on the system which uses the estimates. Examples of types of algorithms that inherently use probabilistic estimates without the estimates being the visible output are data compression algorithms, simulated annealing algorithms, and classification systems trained by sample data.

Two separate results are presented concerning probability estimation from a database. The first result, in Chapters 1–3, is a theoretical development together with a specific algorithm for producing a probabilistic model of a database. The second result, in Chapter 4, defines a necessary and sufficient property of systems that estimate probabilities by error minimization.

Abstract

The problem of creating a probabilistic model using a database is analyzed in this thesis. Two independent results in probabilistic modeling are presented. The first result is a method for creating a model which produces accurate probability estimates. The model is a Gibbs probability distribution representation of the database. This model is created by a new transformation relating the joint probabilities of attributes in the database to Gibbs potentials. The theory of this transformation is presented together with a specific algorithm for efficiently collecting and using the Gibbs potentials. A hash table scheme is used to collect the important potentials without iterative error minimization or repeated searches through a database. The Gibbs modeling scheme allows flexible control of the tradeoffs involving modeling error and sampling error as well as the tradeoffs involved in using the resources of computation time and memory. The performance of the probabilistic modeling algorithm is tested and analyzed. Used as a classifier with a variety of datasets, the Gibbs modeling algorithm was found to equal or surpass the classification results of other models such as neural networks trained with backwards error propagation and the nearest neighbor classification algorithm.

The second independent result is the analysis of systems that use error minimization to estimate probabilities. Minimization to a probability has been a known property of the squared error and cross entropy objective functions. Here the necessary and sufficient conditions for minimization to a probability are developed. It is found that the squared error and cross entropy functions are two of the simplist functions from a family of objective functions which minimize to a probability. If the system is incapable of producing the outputs consistent with the probability estimates, it is shown the minimum error is achieved when the system produces outputs closest to the correct probability estimate outputs. The measure of closeness is described here in terms of the objective function.

Table of Contents

Chapter 1
Introduction to Probabilistic Modeling

1.1 An Example of the Problem

This thesis describes a method of creating a probability distribution from a database. We will begin by considering a simple example. Consider the problem of estimating the probability that a driver with a new car insurance policy will make a large claim on the policy within a year. The insurance company has a database of past policies which includes the history of major claims. The form of the database is shown in Table 1–1.

Table 1–1, Form of the Car Insurance Database

Previously Insured	*Driver's Sex*	*2 or More Moving Violations*	...	*Major Claim*
Yes	Male	No	...	No
No	Male	No	...	Yes
Yes	Female	No	...	No
Yes	Female	Yes	...	No
⋮	⋮	⋮	⋱	⋮
Yes	Male	No	...	Yes

The columns of the database are called *attributes*, the rows are *records*, and the elements comprising a row are called *values*. An *attribute value* is the value together with the column's attribute name. For simplicity this example shows attributes with only two values, but the theory is not limited to this binary case.

In this analysis we will assume that the records from the database are independent, identically distributed (i.i.d.) samples from a probability distribution. This distribution will be called the *underlying* distribution. In contrast, the empirical distribution is based on the frequency of events in the database. It is a sampled estimate of the underlying distribution. Probabilities in the empirical distribution are calculated by counting the number of times an event occurred

in the database and dividing by the number of records in the database. By the law of large numbers, the empirical frequencies must converge to the underlying distribution as the size of the database increases. In Chapters 1-3, we assume that the attributes are discrete. Continuous valued attributes may be used to create a discrete model by the binning method used in Section 3.6. However, other modeling techniques such as kernel based estimation [1] may be more appropriate for this case.

The i.i.d. assumption may be inappropriate if the underlying distribution is changing over time or if the required estimates from the system are from a distribution other than the underlying distribution. For instance the insurance database would not be as useful for making current predictions if the database were years old, because the statistics change slowly over time. Ironically the accuracy of claim predictions would also diminish if the probability estimation system significantly changed the price of insurance offered by the company since this too would alter the underlying distribution of customers who purchase the insurance. By making this i.i.d. assumption we assume that the changes in the distribution are negligible or are modeled separately.

Given a new policy, how can we use the database to estimate the probability of a major claim? Let n be the number of attributes in the database. A *pattern* is a set of attribute values. An i-th order pattern is a pattern made up of i attribute values. A *marginal pattern* is any pattern of order less than n. For example, {Previously Insured – No, Major Claim – Yes} is a second order pattern. As a starting point we will look at three simple techniques for estimating the probability of a major claim. These techniques will be based on first order patterns, second order patterns, and n-th order patterns respectively.

A very simple approach is to ignore all attributes except the major claim attribute and count the number of times a major claim occurred. This probability estimate is based on just one first order pattern. This is often called the *a priori* estimate since it is the estimate we obtain prior to using the information in the record of the new policy.

A much more useful model can be created solely with knowledge of the frequencies of first and second order patterns that include the "major claim" attribute. Let $x_1, x_2, \ldots, x_{n-1}$ represent the $n-1$ attribute values of the new policy. Let x_n indicate the event that there is a major claim on the policy. Let a bar over an attribute value indicate when an event does not occur. For instance $\bar{x}_n$ indicates that there is no major claim on a policy. We are trying to estimate the probability of a major claim given the values of the remaining $n-1$ attributes: $p(x_n|x_1, x_2, \ldots x_{n-1})$. Using Bayes rule

$$p(x_n|x_1, x_2, \ldots x_{n-1}) = \frac{p(x_1, x_2, \ldots x_{n-1}|x_n)p(x_n)}{p(x_1, x_2, \ldots x_{n-1})}.$$

Since $p(x_1, x_2, \ldots x_{n-1}|x_n)p(x_n) + p(x_1, x_2, \ldots x_{n-1}|\bar{x}_n)p(\bar{x}_n) = p(x_1, x_2, \ldots x_{n-1})$, we know:

$$p(x_n|x_1, x_2, \ldots x_{n-1}) = \frac{p(x_1, x_2, \ldots x_{n-1}|x_n)p(x_n)}{p(x_1, x_2, \ldots x_{n-1}|x_n)p(x_n) + p(x_1, x_2, \ldots x_{n-1}|\bar{x}_n)p(\bar{x}_n)}.$$

It can be shown that given no information to the contrary, the best estimate of the probability $p(x_1, x_2, \ldots x_{n-1}|x_n)$ is found by assuming that the $n-1$ conditional probabilities of $p(x_1|x_n)$, $p(x_2|x_n)$, $\ldots p(x_{n-1}|x_n)$ are independent. (This statement follows from the Principle of Maximum Entropy which will be addressed in Section 1.3.) This is called the conditional independence assumption (CI). Thus

$$p(x_n|x_1, x_2, \ldots x_{n-1}) \overset{\text{CI}}{=} \frac{p(x_1|x_n) \cdot p(x_2|x_n) \cdots p(x_{n-1}|x_n)p(x_n)}{p(x_1|x_n) \cdots p(x_{n-1}|x_n)p(x_n) + p(x_1|\bar{x}_n) \cdots p(x_{n-1}|\bar{x}_n)p(\bar{x}_n)}.$$

These second and first order probabilities can be estimated from their empirical distribution in the database. This approximation will have two sources of error, the sampling error and the modeling error. The sampling error is due to the difference between the underlying first and second order probabilities and the frequencies that these patterns occur in the database. The modeling error is the error attributed to the conditional independence assumption. If the attribute values are truly independent when it is known whether or not a major claim was made, then there will be no modeling error.

At another extreme we may look at n-th order patterns in the database. If the database is so large that it contains many records identical (except for the major claim attribute) to the new policy record, then an estimate can be made by counting the number of these records that

include a major claim. This method of estimation has zero modeling error. Since the number of possible unique records grows exponentially in the number of attributes, the probability of any given n-th order pattern in the underlying distribution tends to be very low. This results in very high sampling error because rare events require a larger database to form a reliable estimate of the probability. In contrast, the estimation method based on second order patterns has low sampling error, because a smaller database can be used to accurately estimate the probability of second order patterns.

These three modeling examples highlight the important tradeoff between sampling error and modeling error. If somehow it were possible to create new databases of i.i.d. samples from the underlying distribution, each database would result in a different probability estimate for a given estimation method. The distribution of these estimates has a variance and a mean value. The difference between the mean value and the true underlying probability is the bias. The bias is a quantitative measure of the modeling error. The variance is a measure of the sampling error. Central to the choice of a probability estimating technique is the tradeoff between bias and variance, or modeling error and sampling error.

1.2 A Linear Programming Problem

Let $\mathbf{z}$ be a vector representing the underlying probability distribution. Each element of $\mathbf{z}$ is the probability of a single n-th order pattern. For example, for the binary case using the notation described above:

$$\mathbf{z} = \left.\begin{pmatrix} p(\overline{x}_1, \overline{x}_2, \ldots \overline{x}_n) \\ p(\overline{x}_1, \overline{x}_2, \ldots x_n) \\ \vdots \\ p(x_1, x_2, \ldots \overline{x}_n) \\ p(x_1, x_2, \ldots x_n) \end{pmatrix} \right\} 2^n \text{ terms.}$$

Knowledge of marginal probabilities of the underlying distribution is equivalent to a set of linear constraint on $\mathbf{z}$. For example if $n = 3$ and the marginal probability of the first attribute

is $p(x_1) = .3$ and the marginal probability of the third attribute is $p(x_3) = .9$ then we know two linear constraints:

$$\boldsymbol{\alpha}_i^T \mathbf{z} = \beta_i, \qquad i = 1, 2$$

where

$$\boldsymbol{\alpha_1} = (0\,0\,0\,0\,1\,1\,1\,1)^T, \quad \text{and} \quad \beta_1 = p(x_1) = .3,$$

$$\boldsymbol{\alpha_2} = (0\,1\,0\,1\,0\,1\,0\,1)^T, \quad \text{and} \quad \beta_2 = p(x_3) = .9 \quad .$$

Together a set of marginal distributions describe a linear program:

$$A\mathbf{z} = \mathbf{b} \qquad \mathbf{z} \geq 0 \tag{1–1}$$

where A is a composite matrix formed from the $\boldsymbol{\alpha}$ vectors,

$$A = (\,\boldsymbol{\alpha_0}\boldsymbol{\alpha_1} \ldots \boldsymbol{\alpha_r}\,)^T$$

and

$$\mathbf{b} = (\,\beta_0\beta_1 \ldots \beta_r\,)^T .$$

The first constraint $\boldsymbol{\alpha}_0^T \mathbf{z} = \beta_0$ is the restriction that the sum of all elements in $\mathbf{z}$ must be unity. So $\boldsymbol{\alpha_0} = (1\,1\,1\,\ldots 1)^T$, $\beta_0 = 1$.

Since the β_i values are from the underlying distribution, they cannot be known with certainty. The basic approach used in this thesis is to take the marginal probabilities from the empirical distribution as being the correct β_i values. Marginal probabilities that are known with insufficient precision are not used in forming constraints on $\mathbf{z}$. An exception to this is in Section 2.5 which shows how to incorporate marginal probabilities estimated from very few records.

1.3 Maximum Entropy

There are many different distributions, or $\mathbf{z}$ vectors, with identical marginal probabilities because the matrix A has more columns than rows. Thus problem (1–1) is underconstrained.

The Principle of Maximum Entropy states that the best estimate of the full distribution given the marginals is the distribution that maximizes the entropy [2] [3] [4]. The *entropy* of a distribution is defined (with an abuse of notation we will refer to the distribution by its vector of probabilities):

$$H(\mathbf{z}) = -\sum_{i=1}^{|\mathbf{z}|} z_i \log_2(z_i). \tag{1–2}$$

If random samples are taken from the distribution, the entropy is the lower bound on the number of bits required on average to uniquely represent the sequence of samples. The entropy is a measure of the average information per sample contained in the sequence of samples. The Principle of Maximum Entropy is an analytic statement of the philosophy that the simplest explanation of the evidence is the most likely explanation to be true [4].

It can be shown that each of the three methods of estimation outlined above in the car insurance example are maximum entropy methods. They all give different estimates because each method takes a different set of pattern probabilities to be known.

The linear equation in (1–1) subject to the sign constraint and maximization of (1–2) is a problem in convex optimization. The maximum entropy solution to (1–1) is unique due to the convexity of (1–2) [5].

Using Lagrange's method we can solve for the form of the maximum entropy solution to (1–1) [6] [5].

Given

$$L = H(\mathbf{z}) + \boldsymbol{\lambda}^T(A\mathbf{z} - \mathbf{b}),$$

find $\boldsymbol{\lambda}$, the vector of Lagrange multipliers, such that (1–1) is satisfied as well as the equilibrium conditions

$$\frac{\partial L}{\partial z_i} = 0 \qquad \forall i.$$

Thus

$$-\log z_i - 1 + \boldsymbol{\lambda}^T \mathbf{A}_{\bullet \mathbf{i}} = 0 \qquad \forall i.$$

Here $\mathbf{A_{\bullet i}}$ refers to the i^{th} column from the matrix A. Solve for $\mathbf{z}$

$$z_i = exp\Big(\boldsymbol{\lambda}^T \, \mathbf{A_{\bullet i}} - 1 \Big). \tag{1–3}$$

From the definition of A we know that element $(A)_{i,j}$ is zero or one. It is one if and only if probability z_j is included in the sum for constraint i. Let J_i be the set j s.t. $A_{i,j} = 1$. Then (1–3) can be written:

$$z_i = C \cdot exp\Big(\sum_{j \in J_i} \lambda_j \Big). \tag{1–4}$$

Here all the constants with respect to i have been factored out and labeled C. Distributions governed by equations in the form of (1–4) are called Gibbs distributions.

Since the length of the $\mathbf{z}$ vector is exponential in the number of attributes, equations (1–1) and (1–4) cannot typically be directly solved. Equation (1–4) is presented in order to demonstrate the form of the maximum entropy solution. This Gibbs form has the important property that the probabilities are strictly positive and the number of parameters is equal to the number of known marginal probabilities (the number of linear constraints on the distribution).

1.4 Summary of Chapter 1

- The pattern probabilities create linear constraints on the set of n-th order probabilities.
- The method of maximum entropy produces a single solution to the system of linear constraints.
- The maximum entropy solution is a Gibbs distribution.

1.5 Historical and Bibliographic Notes

The relationship between the Gibbs distribution and maximization of the information theoretic entropy was implied by Shannon and Von Neuman's choice of the word entropy to describe equation (1–2), since the relationship between thermodynamic entropy and the Gibbs distribution was well understood [7] [8]. An analysis of the problem of maximizing the information theoretic entropy under linear constraints, which includes the Lagrange method derivation, is given in reference [5] by Slepian. See [4] and [9] for analyses of the unique properties of the maximum entropy method. Cheeseman presents an algorithm for solving equations (1–1) and (1–4) when the number of attributes is not too large [10]. Many techniques in the literature have been presented for solving the maximum entropy problem under certain restrictions. The technique presented in [11] uses *local event groups*, or LEGs. A LEG is a small set of attributes that are highly dependent. The maximum entropy technique is used to model the probability distribution within a LEG. The computation requirements are kept reasonable by modeling the attributes with sets of small LEGs. When the constraints are defined on non-overlapping sets of attributes the resulting distribution takes on a simple form called a product distribution. The conditional independence method described above is an example of a product distribution solution. Pearl developed methods of organizing the computation of probabilities which take advantage of the product distribution [12].

In the literature there have been successful analyses of neural networks in terms of probabilistic models. The Boltzman Machine, presented in [13], is a neural network learning algorithm that iteratively learns the parameters to a second order Gibbs distribution model. A neural network view of the bias-variance tradeoff was presented in [14].

1.6 References

[1] B.W. Silverman, *Density Estimation for Statistics and Data Analysis*, New York: Chapman and Hall, 1986.

[2] P.M. Lewis, "Approximation Probability Distributions to Reduce Storage Requirements," *Information and Control*, **2**, pp.214–255, 1959.

[3] H.H. Ku and S. Kullback, "Approximating Discrete Probability Distributions," *IEEE Trans. on Information Theory*, **IT-15**, pp.444–447, 1969.

[4] E.T. Jaynes, "On the Rationale of Maximum Entropy Methods," *Proceedings of the IEEE*, **70**, pp.939–952, 1982.

[5] D. Slepian, "On Maxentropic Discrete Stationary Processes," *Bell System Technical Journal*, **51**, pp.629–653, 1972.

[6] E. Swokowski, *Calculus with Analytic Geometry*, Boston: PWS Publishers, 1979.

[7] C. Shannon and W. Weaver, *The Mathematical Theory of Communication*, University of Illinois Press, 1949.

[8] M. Schiffer, "Shannon Information is not Entropy," *Physics Letters A*, **154**, pp.361–365, 1991.

[9] Imre Csiszár, "Why Least Squares and Maximum Entropy?" Report of the Mathematical Institute of the Hungarian Academy of Sciences, Budapest, 1989.

[10] P. Cheeseman, "A Method of Computing Generalize Bayesian Probability Values For Expert Systems," in *International Joint Conference on Artificial Intelligence*, Karlsruhe, West Germany 1983.

[11] K. Konolige, "Bayesian Methods for Updating Probabilities," in *A Computer-Based Consultant for Mineral Exploration* (R. Duda, P. Hart, K Konolige and R. Reboh), SRI International report, Menlo Park CA., 1979.

[12] J. Pearl, *Probabilistic Reasoning in Intelligent Systems,* San Mateo CA: Morgan Kaufmann Publishers, 1988.

[13] G.E. Hinton and T.J. Sejnowski, "Learning and Relearning in Boltzmann Machines," in *Parallel Distributed Processing*, Vol. I., pp.282–317, Cambridge MA: MIT Press, 1986.

[14] S. Geman, E. Bienenstock, and R. Doursat, "Neural Networks and the Bias/Variance Dilemma," *Neural Computation*, **4**, pp.1–58, 1992.

Chapter 2
Converting Probabilities to Gibbs Potentials

2.1 Introduction to the Gibbs Model

Our goal is to estimate the probability of a record (a set of attribute values) from a database of records. A subset of the attribute values is called a pattern. Some patterns may be common in the database and thus increase the estimated probability of records that include these patterns. Patterns occurring rarely in the database decrease the estimated probability of records that include them. The method of Gibbs modeling is to assign a parameter to patterns indicating how much the probability should change when the set of attribute values appear together in a record. This parameter is called a potential. Define a set T to be the set of attributes in a database. For a particular record of the database, define the associated set of attribute values to be the configuration ω of the attributes. A configuration is an n-th order pattern, where $n = |T|$. For each attribute define a special attribute value which will be called the ground state. Given any subset $b \subseteq T$ and a configuration ω, the configuration defined by setting all attributes of ω to the ground state if they are not in b will be called a subconfiguration ω_b.

Using this set notation, the Gibbs probability distribution may be defined:

$$p(\omega) = Z^{-1} \cdot \exp\Big(\sum_{b \subseteq T} J_b(\omega) \Big).$$

The function J_b, called the *potential function*, assigns a real value to every subconfiguration of the set b. The term Z is the normalizing constant that makes the sum of probabilities of all configurations equal to unity. The sum is taken over every subset of the attributes, including the set T itself and the empty set $\emptyset$. Without changing the class of distributions that can be represented by this equation [1], define $J_b(\omega) = 0$ if any attribute in b has the ground state value. Thus for all $A \subseteq T$ the distribution can be written

$$p(\omega_A) = Z^{-1} \cdot e^{V_A(\omega)}, \tag{2–1}$$

where

$$V_A(\omega) = \sum_{b \subseteq A} J_b(\omega). \tag{2–2}$$

The function V_A is called the *energy*. (Often energy is defined with the opposite sign. The arbitrary choice of positive sign will be used throughout this work.) Equation (2–1) says that the log of the probability of a configuration is proportional to the energy, and the energy is the sum of all the potentials defined on sets of nodes in the configuration. Equations (2–1) and (2–2) can be used to describe any distribution that has nonzero probabilities for each configuration (by the monotonicity of e^x).

Prior work in producing Gibbs distribution models such as the Boltzmann Machine [2] have primarily used second order potentials (Gibbs distributions with $J_b = 0$ if b is a set of more than two attributes, $|b| > 2$). By adding new attributes, second order potentials can be used to model increasingly complex distributions. In contrast, the work presented in this chapter uses higher order potentials to model complex probability distributions.

The Principle of Inclusion-Exclusion from set theory states that the following two equations are equivalent (see Appendix 2.A for a description of the principle):

$$g(A) = \sum_{b \subseteq A} f(b) \tag{2–3}$$

and

$$f(A) = \sum_{b \subseteq A} (-1)^{|A-b|}\, g(b). \tag{2–4}$$

The sum in each case is taken over every subset of A, including both the empty set $\emptyset$ and the set A itself. The term $|A - b|$ is the number of elements in A that are not in the set b.

This relation can be used to invert formula (2–2) [1]:

$$J_A(\omega) = \sum_{b \subseteq A} (-1)^{|A-b|}\, V_b(\omega). \tag{2–5}$$

Using (2–1), equation (2–5) may be written

$$J_A(\omega) = \sum_{b \subseteq A} (-1)^{|A-b|}\, \ln(\, p(\omega_b)\,). \tag{2–6}$$

(The Z constant factors out of the equation since $\ln(Z)$ is added the same number of times it is subtracted from the sum.)

2.2 A Transformation to Gibbs Potentials

Equation (2–6) provides a technique for modeling distributions by energy functions rather than directly through the observable joint statistics of sets of attributes. In contrast to common transform models such as those based on the Fourier transform, the transformed representation can have more parameters than the untransformed representation. For the case of potentials defined on binary attributes, each attribute in the database can take 2 states in the pattern or the attribute can be excluded from the pattern. Thus for the binary case there are $3^{|T|}$ parameters (subconfiguration potentials), while there are $2^{|T|}$ parameters (configuration probabilities) in the direct representation. For this reason there exist multiple potential function representations for any single probability distribution. Considering only representations that are symmetric in the treatment of different attributes and attribute values (meaning the ordering in the database does not matter), the pair (2–2) and (2–5) can be generalized to show a larger class of potential function representations:

$$V_A(\omega) = \sum_{b \subseteq A} \Phi(|b|, |A|) J_b(\omega) \tag{2–7}$$

and

$$J_A(\omega) = \sum_{b \subseteq A} \Psi(|b|, |A|) V_b(\omega). \tag{2–8}$$

Here the function Φ of Ψ can be chosen to provide different potential function representations. The complete class of representations allows the potential of a pattern to be dependent on the energy of all configurations, rather than on just the energy of subconfiguration defined on subsets of attribute values in the pattern. The Principle of Inclusion-Exclusion can be used to invert a given function Φ to find the Ψ required to make (2–7) and (2–8) equivalent. Equation (2–1) is used as before to relate $V_A(\omega)$ to probabilities. The probabilities of subconfigurations are estimated directly from frequencies of occurrence of the subconfigurations in a database. The special ground state attribute value is assigned to the "unknown state" for each attribute.

Thus $p(\omega_b)$ can be estimated by the marginal probability of a pattern. The probability of the pattern is estimated by counting the frequency within the database that the attributes in b take on the values assigned in the configuration ω.

A typical problem in probability estimation is to calculate the most probable value for one attribute given the values of all other attributes. As an example, take the case where the unknown attribute is binary with values "true" and "false." Let ω be the configuration of all the known values together with the unknown attribute value set to "true." Let $\bar{\omega}$ be the configuration with the unknown attribute set to "false." From equation (2–1) the probability of the unknown attribute being true given the other attribute values is then calculated by

$$\frac{p(\omega)}{p(\omega)+p(\bar{\omega})} = \frac{exp\left(V_A(\omega)\right)}{exp\left(V_A(\omega)\right)+exp\left(V_A(\bar{\omega})\right)}. \tag{2–9}$$

The summation in equation (2–7) includes exactly $\binom{|A|}{|b|}$ potentials of order $|b|$. When equation (2–7) is substituted into (2–9), all of the potentials that do not include the unknown attribute are factored out of the equation. There remains $\binom{|A|-1}{|b|-1}$ potential functions of order $|b|$ in equation (2–9). The particular choice for the function Φ fixes the relative weighting that potentials of different order will have in the determination of probabilities. One useful method of weighting the potentials of different orders is to divide each potential by the number of potentials of the given order. This *normalized* weighting of the potentials is

$$\Phi(m,n) = \binom{n-1}{m-1}^{-1}.$$

Thus

$$V_A(\omega) = \sum_{b \subseteq A} \binom{|A|-1}{|b|-1}^{-1} J_b(\omega). \tag{2–10}$$

The Ψ resulting from this choice of Φ is shown in Appendix 2.C to be

$$\Psi(m,n) = \begin{cases} 1, & \text{if } m = n; \\ -1 \cdot (n-1)^{-1}, & \text{if } m = n-1, \quad m > 0; \\ 0, & \text{otherwise.} \end{cases}$$

Using (2–1) this may be written for $|A| \neq 1$ as

$$J_A(\omega) = Z \cdot \ln(p(\omega_A)) - \sum_{\substack{b \subset A \\ |b|=|A|-1}} (|A|-1)^{-1} \cdot Z \cdot \ln(p(\omega_b)). \tag{2–11}$$

Here the sum is taken over all subsets of A with $|A| - 1$ elements.

The interesting form of equation (2–11) is best demonstrated by showing the formula evaluated for a simple example distribution with three attributes. Consider the case with $T = \{X, Y, Z\}$ and $\omega = \{(X, x), (Y, y), (Z, z)\}$. Lower case letters indicate a specific value for an attribute. Using the abbreviation $p(\omega_{\{X\}}) = p(x), \quad p(\omega_{\{X,Y\}}) = p(x, y)$, etc., the following potentials are defined:

$$
\begin{aligned}
J_{\{X\}}(\omega) &= Z \cdot \ln(p(x)) \\
J_{\{Y\}}(\omega) &= Z \cdot \ln(p(y)) \\
J_{\{Z\}}(\omega) &= Z \cdot \ln(p(z)) \\
J_{\{X,Y\}}(\omega) &= Z \cdot \ln \frac{p(xy)}{p(x)p(y)} \\
J_{\{Y,Z\}}(\omega) &= Z \cdot \ln \frac{p(yz)}{p(y)p(z)} \\
J_{\{X,Z\}}(\omega) &= Z \cdot \ln \frac{p(xz)}{p(x)p(z)} \\
J_{\{X,Y,Z\}}(\omega) &= Z \cdot \ln \frac{p(xyz)}{\sqrt{p(xy)p(yz)p(xz)}}.
\end{aligned}
$$

These potentials tend to be near zero because the numerator within the logarithm is approximated by the denominator:

$$
p(\omega_A) \approx \prod_{\substack{b \subset A \\ |b| = |A|-1}} p(\omega_b)^{\frac{1}{|A|-1}}.
$$

Note that the approximation is exact and the potential is zero if the attributes are independent.

For the example configuration $\omega = \{(X, x), (Y, y), (Z, z)\}$, the equivalence of (2–10) and (2–11) given can be verified. Using (2–1)

$$
V_A(\omega) = Z \cdot \ln\big(p(xyz)\big).
$$

This energy can also be calculated by substitution into (2–10)

$$
\begin{aligned}
V_A(\omega) = &\binom{2}{0}^{-1} Z \Big(\ln(p(x)) + \ln(p(y)) + \ln(p(z))\Big) \\
&+ \binom{2}{1}^{-1} Z \left(\ln \frac{p(xy)}{p(x)p(y)} + \ln \frac{p(yz)}{p(y)p(z)} + \ln \frac{p(xz)}{p(x)p(z)}\right) \\
&+ \binom{2}{2}^{-1} Z \cdot \ln \frac{p(xyz)}{\sqrt{p(xy)p(yz)p(xz)}} \\
= &\, Z \cdot \ln\Big(p(x)p(y)p(z)\Big) \\
&+ .5 \cdot Z \cdot \ln\Big(\frac{p(xy)}{p(x)p(y)} \frac{p(yz)}{p(y)p(z)} \frac{p(xz)}{p(x)p(z)}\Big) \\
&+ Z \cdot \ln\Big(\frac{p(xyz)}{\sqrt{p(xy)p(yz)p(xz)}}\Big) \\
= &\, Z \cdot \ln(p(xyz)).
\end{aligned}
$$

The two methods of calculating the energy, (2–10) and (2–1), are equivalent as we expected from the result shown in Appendix 2.C.

2.3 Using the Transformation to Model a Database

Equation (2–11) is used to define a parameter, the potential, for each pattern that occurs in the database. Sets of potentials create a model of the database. The probability of events in the database can be calculated from the potentials using equation (2–10). Since the number of potentials grows exponentially in the number of attributes, it is not usually possible to store or process every single potential. Instead, only the largest magnitude potentials are used in modeling the database.

The pseudocode shown in Appendix 2.D describes the basic algorithm for collecting the frequency and potentials of the most important patterns in the database (typewriter font will be used to refer to variables or procedures used in the algorithm from Appendix 2.D). The pattern records are kept in a database referenced by the `hash_function`. A hash function is a pseudo-random number generator that uses the database record (the pattern `p`) as the random number seed. Two different patterns passed to the hash function result in two (not necessarily unique) integer numbers approximately uniformly distributed in some interval much larger than $\mathcal{H}$, the length of the hash table. The purpose of the hash table is to quickly find a specific

pattern in the database without doing a search using more than $\mathcal{C}$ pattern comparisons. A hash function *collision* occurs when two patterns have identical hash addresses. When no collision occurs, the hash address is the index into the hash table used to store the pattern, counter and potential. When a collision occurs, the above algorithm attempts to store this information in one of the $\mathcal{C}$ addresses following the hash address. If all of these addresses are already filled, the pattern with the smallest magnitude potential is removed and replaced by the new pattern. Once the function `collect_potentials(database)` is complete, the hash table is filled with patterns with high magnitude potentials. The energy of a record is determined by the weighted sum of all the potentials in the database. The correct weighting is $\binom{n-1}{d-1}^{-1}$, given by formula (2–10), where n is the number of attributes in the record and d is the number of attributes in the given pattern. The log of the probability of a record is approximately proportional to the energy (by equation (2–1)). Thus the database of potentials can be used to estimate ratios of probabilities of records. In contrast to a direct representation of probabilities, the potential representation has the advantage that unknown parameters can be approximated as having value zero. Thus the tradeoff between bias and variance and the limited resources of computer time and memory can be manipulated by choosing which potentials to calculate and store. In the basic algorithm, the tradeoffs are controlled by the choice of the maximum potential order $\mathcal{D}$ and the size $\mathcal{H}$ of the hash table.

There are several modifications to this basic algorithm that improve performance on certain databases and certain applications of the probabilistic model. One variation is to use a scheme to limit the number of patterns under consideration in addition to the method of limiting the pattern order to some constant $\mathcal{D}$. Another variation is to use small sample statistics to estimate pattern probabilities rather than to directly use their frequency. A third variation is to use some value other than the potential in determining which patterns get removed from the database after a collision.

2.4 Limiting the Patterns Under Consideration

In addition to order limiting, some databases have a structure that lends itself to another technique for limiting the number of patterns. This technique is called limiting the diameter of the pattern on a graph. In order to use this technique, a graph (a set of nodes, and a set of edges between nodes) must be defined with each node in the graph representing an attribute. The edges of the graph represent the most closely related (statistically dependent) pairs of attributes. Given this graph, the diameter of a pair of attributes is the minimum number of edges that must be traversed in the graph to move from one attribute's node to the other attribute's node. The diameter of a pattern is the maximum diameter for all the pairs of attributes in the pattern. All patterns with diameter greater than the diameter limit are assumed to have zero potential. A *clique* is a set of nodes such that every pair is connected by an edge. For the special case where the diameter limit is one, the diameter restriction is equivalent to considering only patterns that are cliques of the graph. For this case the unnormalize potential function and inversion given by (2–6) and (2–2) has been shown by Clifford-Hammersley [4] [1] to produce a distribution consistent with every one of the joint probability estimates from attributes on cliques of the graph. Another important case is when the graph forms a tree (a graph with no loops). For this case the maximum entropy distribution has a simple form, called a product distribution, which can be easily calculated from the pattern frequencies without the approximation that the unused potentials (as defined by (2–11)) have zero value [5].

2.5 Small Sample Statistics

In the basic Gibbs modeling algorithm, the sample means (proportional to the number of occurrences of the pattern found in `hash[i]->counter`) from a database are used to estimate the probability of events in an underlying distribution. When there is prior knowledge of a probability to estimate, Bayes rule can be used to incorporate the prior knowledge with the sample mean. This results in an adjustment which shifts the probability estimate toward

the prior. Without this small sample adjustment, potentials can either be treated as "known exactly" or "unknown." The adjustment allows for an in-between case of "known with some uncertainty." The small sample adjustment is effective in reducing the variance of estimates created from small databases.

Let μ be the event probability to be estimated. Let x be a single record of the database. Let $\mathcal{X}$ be the complete set of database records, comprised of n_0 records where the event was false and n_1 records where the event was true. We will consider $\mathcal{X}$ to be a random sample from an underlying distribution rather than a known fixed database. Let $n = n_0 + n_1$. We desire to find the distribution $p(\mu|\mathcal{X})$. Assume first our *a priori* estimate of the distribution of μ is the uniform distribution (the event with mean value μ may still be discrete-valued while the distribution of μ is continuous).

$$p(\mu) = 1 \cdot d\mu \text{ for } 0 \leq \mu \leq 1 \quad .$$

The events within $\mathcal{X}$ are independent identically distributed with probability μ for a true event and $(1 - \mu)$ for a false event, therefore:

$$p(\mathcal{X}|\mu) = (1 - \mu)^{n_0} \, \mu^{n_1}.$$

Bayes rule states

$$p(\mu|\mathcal{X}) = \frac{p(\mathcal{X}|\mu)p(\mu)}{p(\mathcal{X})} = \frac{p(\mathcal{X}|\mu)p(\mu)}{\int p(\mathcal{X}|u)p(u)\,du}.$$

Collecting constants with respect to μ into C_1 gives

$$p(\mu|\mathcal{X}) = C_1 \, (1 - \mu)^{n_0} \, \mu^{n_1}. \tag{2–12}$$

This polynomial has median value $\mu = n_1/n$. This distribution converges (as it must by the law of large numbers) to the normal distribution with mean value equal to the median n_1/n.

Bayes rules shows how to calculate the *a posteriori* distribution $p(\mu|\mathcal{X})$ given arbitrary $p(\mu)$ and $p(\mathcal{X}|\mu)$. A particular simple form for $p(\mu|\mathcal{X})$ results if the distributions $p(\mu)$ and $p(\mathcal{X}|\mu)$ are normal. Let $p(\mu) \sim N(u_0, \sigma_0^2)$ and $p(x|\mu) \sim N(\mu, \sigma)$ and $p(\mu|\mathcal{X}) \sim N(\mu_n, \sigma_n)$.

Application of Bayes rule to these distributions shows the following relations [6]:

$$\mu_n = \left(\frac{n\sigma_0^2}{n\sigma_0^2 + \sigma_0^2}\right)\frac{n_1}{n_0 + n_1} + \left(\frac{\sigma^2}{n\sigma_0^2 + \sigma_0^2}\right)\mu_0, \tag{2–13}$$

$$\sigma_n^2 = \frac{\sigma_0^2\sigma^2}{n\sigma_0^2 + \sigma_0^2}.$$

Thus the distribution $p(\mu|\mathcal{X})$ is a weighted combination of the sample mean $\frac{n_1}{n_0+n_1}$ and the prior estimate μ_0. The relative weighting is determined by the ratio

$$SSF = \frac{\sigma^2}{\sigma_0^2}.$$

The value SSF is the small sample factor which indicates the relative uncertainty in the prior estimate.

For even very small $n \geq 1$ non normal distributions $p(x|\mu)$ and $p(\mu)$ have an effect closely approximated by normal distributions (for example, see how quickly equation (2–12) converges to the normal distribution even though it resulted from a prior distribution with no central peak). The approximation error goes to zero as n increases. The *small sample statistics modification* for the Gibbs modeling algorithm is to use μ_n given by (2–13) rather than the sample mean $n_1/(n_0 + n_1)$. In order to use the small sample statistics modification, the prior estimate μ_0 must be known and the small sample factor SSF must be chosen. The Gibbs energy model provides a technique for estimating the probability of any pattern. For every pattern of order $\mathcal{D}$ the Gibbs model probability estimate using patterns of order $\mathcal{D}-1$ is taken to be the prior estimate μ_0. The choice of SSF depends on the distribution to be modeled. Larger values for SSF result in models that are based less on higher order statistics and more on the lower order statistics. Setting $SSF = 0$ results in removing the small sample modification entirely.

2.6 Pattern Rejection when there is a Hash Collision

Another portion of the Gibbs modeling algorithm that can be modified for certain applications is the rejection criteria for patterns in the hash table. The basic algorithm uses the magnitude of the pattern potential to determine which pattern is least important in the probabilistic model. If the distribution is tested by calculating the probability of random samples from the same distribution that created the database, better performance can be achieved by keeping potentials for patterns occurring often in the database. For a given pattern the potential is a measure of how much the occurrence of the pattern changes the estimated probability. The frequency of the pattern is an estimate of the probability that this potential will actually be used in estimating the probability. Thus a useful rejection criteria is to use the product of the frequency times the potential, rather than the potential itself. Another useful modification of the pattern rejection criterion is to use the pattern frequency rather than the pattern potential. This produces results similar to the above method of using the frequency times the potential, except that it speeds up the algorithm by making it unnecessary to do the `find_potential(p)` step repeatedly while processing the records in the database.

2.7 Summary of Chapter 2

- The method of Gibbs modeling is to assign a parameter to each subset of the complete set of attribute values in the database. This parameter, called a potential, defines how much the probability should change when the subset of attribute values (called a pattern) appears together in a record.

- There is not a unique Gibbs potential representation for a probability distribution. In choosing a particular Gibbs representation it is desirable to find a representation with most of the potentials set to zero.

- A new transformation to define Gibbs potentials from a probability distribution is presented. Each potential has the form $\log(p(\omega_A)/\hat{p}(\omega_A))$. Here $p(\omega_A)$ is the marginal probability of a pattern from the database and $\hat{p}(\omega_A)$ is an estimate of this same probability based on the probability of smaller subsets of attribute values.

- An algorithm is presented for collecting the patterns with the largest magnitude potentials in the database. The remaining potentials are all assumed to have zero potential. The basic algorithm is extended to take advantage of known zero potential patterns, to use small sample statistics, and to customize the collected potentials to applications of the probability model.

2.8 Historical and Bibliographic Notes

Clifford and Hammersley used the unnormalized inversion formula of equation (2–6) to show the equivalence of a finite Markov Random Field (MRF) to a Gibbs distribution. An MRF is a modeling technique where the attributes are associated with nodes of a graph. Two attributes are called *neighbors* if their corresponding nodes in the graph are connected by an edge. In an MRF the probability of a set of attribute values given all of the other attribute values in the record is equal to the probability given only those values of the attributes that

are neighbors. This is a generalization of a Markov Process. A Markov process is an MRF for a linear graph (a graph with no loops, and with one or two edges connected to every node). The Clifford-Hammersley theorem shows that for a finite graph, a Markov Random Field with nonzero probabilities is equivalent to a Gibbs distribution where all patterns that include two non-neighbor attributes are zero [7]. This result was preceded by Averintsev for the special case of a lattice graph [8]. The excellent book by Kindermann and Snell describes applications of MRF-Gibbs models in economics, sociology, and neuron firing [1]. The influential paper by Geman and Geman applied the MRF approach to the problem of correcting a distorted image [9]. When the graph associated with the attribute dependencies takes on a tree structure, the methods based on the product distribution may be used [5] [10] [11]. The results presented in this chapter grew from attempts to improve the models created from a database using the direct Clifford-Hammersley Gibbs potential representation, without assuming an *a priori* structure for the graph of dependencies.

The mathematical technique of the Principle of Inclusion-Exclusion is used to invert a function defined as the sum of functions on all subsets of a set. If the ordering operator $\subseteq$, that describes which sets are subsets of others, is replaced by any other ordering operator, or if not all subsets are summed, then the function can still be inverted. This more general technique is called Möbius Inversion. Möbius Inversion can be used to generalize the Clifford-Hammersley theorem to other classes of restricted Gibbs distributions. Stanley shows methods of finding the Möbius Inversion and lists many of the important applications [3].

2.9 References

[1] R. Kindermann, J.L. Snell, *Markov Random Fields and their Applications*, Providence, RI: American Mathematical Society, 1980.

[2] G.E. Hinton and T.J. Sejnowski, "Learning and Relearning in Boltzmann Machines," in *Parallel Distributed Processing*, Vol. I., pp.282–317, Cambridge MA: MIT Press, 1986.

[3] R. P. Stanley, *Enumerative Combinatorics Volume I*, Monterey CA: Wadsworth & Brooks/Cole, 1986.

[4] C. Preston, *Gibbs States on Countable Sets*, Cambridge University Press, 1974.

[5] J. Pearl, *Probabilistic Reasoning in Intelligent Systems,* San Mateo CA: Morgan Kaufmann Publishers, 1988.

[6] R. Duda and P. Hart, *Pattern Classification and Scene Analysis*, New York: Wiley, 1973.

[7] J. Besag, "Spatial Interaction and the Statistical Analysis of Lattice Systems," *J. Royal Statist. Soc.* Series B, **36**, pp.192–225, 1974.

[8] M.B. Averintsev, "On a Method of Describing Discrete Parameter Random Fields," *Problemy Peredachi Informatsii*, **6**, pp.100–109, 1970.

[9] S. Geman and D. Geman, "Stochastic Relaxation, Gibbs Distributions, and the Bayesian Restoration of Images," *IEEE Pattern Analysis and Machine Intelligence*, **6**, pp.721–741, 1984.

[10] P.M. Lewis , "Approximation Probability Distributions to Reduce Storage Requirements," Information and Control, **2**, pp.214–255, 1959.

[11] K. Konolige, "Bayesian Methods for Updating Probabilities," in *A Computer-Based Consultant for Mineral Exploration* (R. Duda, P. Hart, K Konolige and R. Reboh), SRI International report, Menlo Park CA., 1979.

Appendix 2.A
The Principle of Inclusion-Exclusion

Given

$$g(A) = \sum_{b \subseteq A} f(b) \tag{2–14}$$

where A is a set of n "properties":

$$A = \{t_1, t_2, \ldots t_n\}.$$

Evaluate (2–14) for the cases $n = 0, 1, 2$ to show for all $j \neq i$ the equalities:

$$g(\emptyset) = f(\emptyset), \tag{2–15}$$

$$g(\{t_i\}) = f(\{t_i\}) + f(\emptyset), \tag{2–16}$$

$$g(\{t_i, t_j\}) = f(\{t_i, t_j\}) + f(\{t_i\}) + f(\{t_j\}) + f(\emptyset),$$

Now we can solve for $f(\{t_i, t_j\})$

$$f(\{t_i, t_j\}) = g(\{t_i, t_j\}) - f(\{t_i\}) - f(\{t_j\}) - f(\emptyset),$$

Use (2–15) and (2–16) to show

$$f(\{t_i, t_j\}) = g(\{t_i, t_j\}) - g(\{t_i\}) - g(\{t_j\}) + g(\emptyset).$$

Similarly for $n = 3$, (2–14) can be inverted to show

$$\begin{aligned} f(\{t_i, t_j, t_k\}) = {} & g(\{t_i, t_j, t_k\}) \\ & - g(\{t_i, t_j\}) - g(\{t_i, t_k\}) - g(\{t_j, t_k\}) \\ & + g(\{t_i\}) + g(\{t_j\}) + g(\{t_k\}) \\ & - g(\emptyset). \end{aligned}$$

Let $|A - b|$ be the cardinality of the set $A - b$. The general inversion formula is simply [3]:

$$f(A) = \sum_{b \subseteq A} (-1)^{|A-b|}\, g(b). \tag{2–17}$$

Appendix 2.B
Approximating the Energy by Setting Potentials to Zero

Since the number of potentials of a given order increases exponentially with the order, it is useful to approximate the energy of a configuration by defining a maximum order $\mathcal{D}$ such that all potentials of greater order are assumed to be zero

$$J_b(\omega) = 0 \quad \forall \ b \text{ such that } |b| > \mathcal{D}.$$

Let $\hat{V}_A(\omega)$ be the resulting approximation to the energy $V_A(\omega)$. Let $|A| = n$.

Theorem:

Given

$$J_A(\omega) = V_A(\omega) \ - \sum_{\substack{b \subseteq A \\ |b|=n-1}} (n-1)^{-1} \cdot V_b(\omega) \tag{2–18}$$

and the order $\mathcal{D}$ approximation to equation (2–10):

$$\hat{V}_A(\omega) = \sum_{i=1}^{\mathcal{D}} \binom{n-1}{i-1}^{-1} \sum_{\substack{b \subseteq A \\ |b|=i}} J_b(\omega), \tag{2–19}$$

then

$$\hat{V}_A(\omega) = \sum_{\substack{b \subseteq A \\ |b|=\mathcal{D}}} \binom{n-1}{\mathcal{D}-1}^{-1} V_b(\omega).$$

Corollary:

For the case $\mathcal{D} = n$, the approximation is exact

$$\hat{V}_A(\omega) = V_A(\omega).$$

The theorem's results has a simple interpretation. The energy of a configuration is approximated by a scaled average of the energies of the configurations of order $\mathcal{D}$. Using equation (2–1) to relate energies to probabilities, shows that the estimated probability is a scaled geometric mean of the order $\mathcal{D}$ marginal probabilities.

Proof:

We start with the given equation for $\hat{V}_A(\omega)$

$$\hat{V}_A(\omega) = \sum_{i=1}^{\mathcal{D}} \binom{n-1}{i-1}^{-1} \sum_{\substack{b \subseteq A \\ |b|=i}} J_b(\omega).$$

Use equation (2–18) to substitute $J_b(\omega)$ out of the equation:

$$\hat{V}_A(\omega) = \sum_{i=1}^{\mathcal{D}} \binom{n-1}{i-1}^{-1} \sum_{\substack{b \subseteq A \\ |b|=i}} \left(V_b(\omega) - \sum_{\substack{c \subset b, |b| \neq 1 \\ |c|=|b|-1}} (i-1)^{-1} \cdot V_c(\omega) \right).$$

Separate the term in the first sum where $i = \mathcal{D}$

$$\hat{V}_A(\omega) = \sum_{\substack{b \subseteq A \\ |b|=\mathcal{D}}} \binom{n-1}{\mathcal{D}-1}^{-1} V_b(\omega) +$$

$$\left(\sum_{i=1}^{n-1} \binom{n-1}{i-1}^{-1} \sum_{\substack{b \subset A \\ |b|=i}} V_b(\omega) \right) - \left(\sum_{i=1}^{n} \binom{n-1}{i-1}^{-1} \sum_{\substack{b \subseteq A \\ |b|=i}} \sum_{\substack{c \subset b, |b| \neq 1 \\ |c|=|b|-1}} (i-1)^{-1} \cdot V_c(\omega) \right).$$

By subtracting $\hat{V}_A(\omega)$ from both sides using equation (2–19) we see that it is sufficient to show

$$\sum_{i=1}^{\mathcal{D}-1} \binom{n-1}{i-1}^{-1} \sum_{\substack{b \subset A \\ |b|=i}} V_b(\omega) = \sum_{i=1}^{\mathcal{D}} \binom{n-1}{i-1}^{-1} \sum_{\substack{b \subseteq A \\ |b|=i}} \sum_{\substack{c \subset b, |b| \neq 1 \\ |c|=|b|-1}} (i-1)^{-1} \cdot V_c(\omega).$$

Since the right hand sum has no terms when $i = 1$, the starting index may be changed

$$\sum_{i=1}^{\mathcal{D}-1} \binom{n-1}{i-1}^{-1} \sum_{\substack{b \subset A \\ |b|=i}} V_b(\omega) = \sum_{i=2}^{\mathcal{D}} \binom{n-1}{i-1}^{-1} \sum_{\substack{b \subseteq A \\ |b|=i}} \sum_{\substack{c \subset b \\ |c|=|b|-1}} (i-1)^{-1} \cdot V_c(\omega).$$

The right hand side inner double summation counts a given $V_c(\omega)$ once for every b such that $c \subset b \subseteq A$ with $i = |b| = |c| + 1$. This occurs exactly $|A| - |c| = n - i + 1$ times. Thus

$$\sum_{i=1}^{\mathcal{D}-1} \binom{n-1}{i-1}^{-1} \sum_{\substack{b \subset A \\ |b|=i}} V_b(\omega) = \sum_{i=2}^{\mathcal{D}} \binom{n-1}{i-1}^{-1} \sum_{\substack{c \subset A \\ |c|=i-1}} \frac{n-i+1}{i-1} \cdot V_c(\omega).$$

Now perform a change of variables. Let $j = i - 1$ on the right hand side

$$\sum_{i=1}^{\mathcal{D}-1} \binom{n-1}{i-1}^{-1} \sum_{\substack{b \subset A \\ |b|=i}} V_b(\omega) = \sum_{j=1}^{\mathcal{D}-1} \binom{n-1}{j}^{-1} \sum_{\substack{c \subset A \\ |c|=j}} \frac{n-j}{j} \cdot V_c(\omega).$$

Clearly both sides are identical since

$$\binom{n-1}{i-1}^{-1} = \binom{n-1}{i}^{-1} \frac{n-i}{i}.$$

Q.E.D.

Appendix 2.C
Inversion of the Normalized Transformation

Theorem:

Let T be a finite set. Each element of T will be called an attribute. Each attribute can take on one of a finite set of states called attribute values. A collection of attribute values for every element of T is called a configuration ω. For all $A \subseteq T$ (including both the empty set $A = \emptyset$ and the full set $A = T$), let $V_A(\omega)$ and $J_A(\omega)$ be functions mapping the states of the elements of A to the real numbers. Define $\binom{m}{n} = m!/((m-n)! \cdot n!)$ to be "m choose n."

Let $V_\emptyset(\omega) = 0$, $J_\emptyset(\omega) = 0$, and let $V_A(\omega) = J_A(\omega)$ if $|A| = 1$.

Then for $|A| > 1$:

$$V_A(\omega) = \sum_{b \subseteq A} \binom{|A|-1}{|b|-1}^{-1} J_b(\omega), \tag{2–20}$$

and

$$J_A(\omega) = V_A(\omega) \; - \sum_{\substack{b \subset A \\ |b|=|A|-1}} (|A|-1)^{-1} \cdot V_b(\omega) \tag{2–21}$$

are equivalent in that any assignment of V_A and J_A values for all $A \subseteq T$ will satisfy (2–20) if and only if they also satisfy (2–21).

Proof:

Let $\mathcal{J}$ be any assignment of the values $J_A(\omega)$ for all $A \subseteq T$. Let $\mathcal{V}$ be any assignment of all the values $V_A(\omega)$ for all $A \subseteq T$. Then clearly (2–20) maps any assignment $\mathcal{J}$ to a unique $\mathcal{V}$. We will represent this mapping by the function f, so (2–20) is abbreviated $\mathcal{V} = f(\mathcal{J})$. Similarly (2–21) maps any assignment $\mathcal{V}$ to a unique $\mathcal{J}$. Equation (2–21) will be abbreviated $\mathcal{J} = g(\mathcal{V})$. The result of Appendix 2.B, applied with the value $\mathcal{D}$ set to n, shows that $f(g(\mathcal{V})) = \mathcal{V}$. In lemma C1 below, it is shown $g(f(\mathcal{J})) = \mathcal{J}$. Therefore the equations (2–20) and (2–21) are inverse one-to-one mappings and the association of assignments between $\mathcal{J}$ and $\mathcal{V}$ are identical for the two equations. *Q.E.D.*

Lemma C1: $g(f(\mathcal{J})) = \mathcal{J}$

Let $|A| = n$. It is sufficient to show that substituting V_b out of (2–21) using (2–20) yields an identity:

$$\begin{aligned} J_A(\omega) &= V_A(\omega) - \sum_{\substack{b\subset A, n\neq 1 \\ |b|=n-1}} (n-1)^{-1} \cdot V_b(\omega) \\ &= \sum_{b\subseteq A} \binom{n-1}{|b|-1}^{-1} J_b(\omega) - \sum_{\substack{b\subset A, n\neq 1 \\ |b|=n-1}} (n-1)^{-1} \sum_{c\subseteq b} \binom{|b|-1}{|c|-1}^{-1} J_c(\omega). \end{aligned}$$

Separate the term in the first sum for which $b = A$

$$\begin{aligned} J_A(\omega) = J_A(\omega) \; + \\ \sum_{b\subset A, b\neq A} \binom{n-1}{|b|-1}^{-1} J_b(\omega) - \sum_{\substack{b\subset A, n\neq 1 \\ |b|=n-1}} (n-1)^{-1} \sum_{c\subseteq b} \binom{|b|-1}{|c|-1}^{-1} J_c(\omega). \end{aligned}$$

By subtracting $J_A(\omega)$ we see that it is sufficient to show

$$\sum_{b\subset A, b\neq A} \binom{n-1}{|b|-1}^{-1} J_b(\omega) = \sum_{\substack{b\subset A, n\neq 1 \\ |b|=n-1}} \sum_{c\subseteq b} (n-1)^{-1} \binom{|b|-1}{|c|-1}^{-1} J_c(\omega).$$

Now the right hand side double sum counts a given $J_c(\omega)$ once for every b such that $c \subseteq b \subset A$ with $|b| = |A| - 1 = n - 1$. This occurs $|A| - |c| = n - |c|$ times. So the equation can be written

$$\sum_{b\subset A, b\neq A} \binom{n-1}{|b|-1}^{-1} J_b(\omega) = \sum_{c\subset A, c\neq A} \frac{n-|c|}{n-1} \binom{n-2}{|c|-1}^{-1} J_c(\omega).$$

Both sides are identical since:

$$\binom{n-1}{i-1}^{-1} = \frac{n-i}{n-1} \binom{n-2}{i-1}^{-1}.$$

Q.E.D.

Appendix 2.D
An Algorithm for Collecting Potentials from a Database

Four procedures, `process_pattern(p,hash,`$\mathcal{H}$`,`$\mathcal{C}$`)`, `abs(x)`, `hash_function(p)`, and `find_potential(p)` are used in the basic algorithm `collect_potentials(database)`.

```
collect_potentials(database)
  Define a maximum potential order 𝒟
  Define a hash table with ℋ entries.
    Each entry includes a pattern, an integer counter, and a real number potential.
    These are referred to by hash[i]-> pattern, hash[i]-> counter, and
     hash[i]-> potential respectively, for i between 0 and ℋ - 1
  Define a collision range 𝒞
  Initialize all counters in the hash table to zero.
  for each record r in the database (each record is n pairs of attribute values)
    for each order d between 1 and 𝒟
      for each pattern p of order d in the record r (there are (n d) patterns)
        process_pattern(p,hash,ℋ,𝒞)
  for each order d between 1 and 𝒟
    for each hash table position i from 0 to ℋ − 1
      if hash[i]->pattern has order d
      then
        hash[i]->potential := find_potential(hash[i]->pattern)
```

```
process_pattern(p,hash,H,C)
  address := hash_function(p) modulo (H - C)
  found := false
  min_p_address := address
  i := address
  while i < (address + C) and  found == false
    if hash[i]->pattern == p
    then
      hash[i]->counter   := hash[i]->counter + 1
      hash[i]->potential := find_potential(p)
      found := true;
    if abs(hash[i]->potential) < abs(hash[min_p_address]->potential)
    then
      min_p_address := i;
    i := i + 1
  if found == false
  then
    hash[min_p_address]->pattern := p
    hash[min_p_address]->count := 1
    hash[min_p_address]->potential := find_potential(p)
```

`abs(x)`

returns the absolute value of the number x

`hash_function(p)`

returns a pseudo-random positive integer based on the seed p,

where p is a database pattern.

The difference of two hash values, hash_function(p1) - hash_function(p2),

is zero if p1 == p2, otherwise the difference is approximately uniformly distributed

over the range from $-1 * 2^{31}$ to $1 * 2^{31}$.

`find_potential(p)`

returns potential of a pattern p, using equation (2–11) with $Z = 1$ and

the frequencies of patterns as defined in the hash table.

Chapter 3
Analysis of the Gibbs Modeling Algorithm

3.1 Introduction

In formalizing the problem of estimating a probability distribution from a database, Chapter 1 demonstrated that the Principle of Maximum Entropy forces a Gibbs distribution solution. Chapter 2 presented an algorithm that models a database by a Gibbs distribution. This Gibbs modeling algorithm works with limited memory and processor time by discarding the small potential patterns in the database and assuming they have zero potential. If the discarded patterns truly have zero potential, the model is an exact representation of the empirical distribution of the database. In practice, however, the rejected potentials have nonzero values. Approximating these potentials with value zero introduces errors in the model. Although the approximation retains the form of a Gibbs distribution, it is not the maximum entropy solution. The purpose of this chapter is to analyze the error in the approximation inherent in the Gibbs modeling algorithm. Four different approaches will be taken. First the limiting case of large database size $\mathcal{R}$ and high maximum order $\mathcal{D}$ is discussed. Next a test distribution is introduced. Analytic equations for the expected results of the algorithm for the test distribution are presented and compared with program output. Next the use of the algorithm in a classification system is studied. The classification system applied to the test distribution is shown to be optimal when sampling error is neglected. Finally the algorithm is tested as a classifier on a variety of databases. The classifier performance is compared to well known classification methods such as the nearest neighbor algorithm and feed forward neural network classifiers.

3.2 Error in the Probability Estimation

In this section we examine the effects of database size and model size on probability estimation using the Gibbs model. Let $\mathcal{R}$ be the number of records in the database used to produce the probability model. Let $\mathcal{D}$ be the maximum order of patterns in the energy model; all patterns with more than $\mathcal{D}$ attributes are assigned zero potential. This analysis neglects the effect of hash collisions in the Gibbs modeling algorithm. When a hash collision occurs there is not enough memory to store new pattern potentials. As described in Section 2.3, when a collision occurs a low magnitude potential is removed from the model. Here we analyze the case where there is sufficient memory to store all the patterns of order less than $\mathcal{D}$. We will assume that all potentials of order $\mathcal{D}$ and less are defined (as in Section 2.2):

$$J_A(\omega) = Z \cdot \ln(f(\omega_A)) - (|A| - 1)^{-1} \cdot Z \sum_{\substack{b \subset A \\ |b|=|A|-1}} \ln(f(\omega_b)). \tag{3–1}$$

Here $f(\omega_b)$ is the empirical frequency with which the attributes in set b take on the values assigned in ω. Given the potentials of patterns, the energy of a particular pattern is written

$$V_A(\omega) = \sum_{b \subseteq A} \binom{|A| - 1}{|b| - 1}^{-1} J_b(\omega).$$

Choosing $Z = 1$, the energy for a pattern is the natural log of the probability estimate:

$$\ln(\hat{p}_{\mathcal{D}}(\omega_A)) = \sum_{\substack{b \subseteq A \\ |b| \leq \mathcal{D}}} \binom{|A| - 1}{|b| - 1}^{-1} J_b(\omega). \tag{3–2}$$

The subscript $\mathcal{D}$ in the estimate, $\hat{p}_{\mathcal{D}}$, is written to indicate that patterns of order greater than $\mathcal{D}$ were assumed to have zero potential. The records in the database are assumed to be identically distributed samples from some underlying distribution. By the law of large numbers, as $\mathcal{R}$ increases $f(\omega_b)$ must converge to the probability $p(\omega_b)$ of the pattern ω_b in the underlying distribution. Equation (3–1) gives a transformed representation of the frequency of events in the database. The results of Section 2.2 show that this energy representation is equivalent to the frequency representation for the case where $\mathcal{D} = \mathcal{N}$. For large $\mathcal{R}$ this frequency becomes equivalent to the underlying probability distribution. Thus a starting

point in our understanding of the modeling effect of $\mathcal{R}$ and $\mathcal{D}$ is that the error in the probability estimation goes to zero for $\mathcal{D} = \mathcal{N}$ and large $\mathcal{R}$.

Although the modeling error is zero if $\mathcal{D} = \mathcal{N}$, for smaller $\mathcal{D}$ the modeling error cannot be bound usefully without making further assumptions about the distribution to be modeled. This fact follows from the work of Minsky and Papert [1]. They demonstrated that there exist distributions on $\mathcal{N}$ attributes which cannot be modeled by a direct weighting of features of order less than $\mathcal{N}$. An example of this is the "parity distribution" on $\mathcal{N}$ binary attributes. In this distribution all records that have an even number of attributes in the 1 state are equally probable, with $p = 2^{1-\mathcal{N}}$. All other records have zero probability. It is easy to verify that for this distribution every potential of order less than $\mathcal{N}$ as defined by (3–1) is zero. Thus, for the parity distribution, the Gibbs model with $\mathcal{D} < \mathcal{N}$ will estimate that every length $\mathcal{N}$ pattern is equally probable. As might be expected, modeling techniques that estimate high order probabilities from lower order probabilities will break down when there is no information in the low order patterns. This is not a drawback of this particular modeling scheme, but rather a general limitation on any system that collects statistics estimating marginal probabilities. Any distribution which we can hope to model must have useful statistics that can be observed with a database smaller than the large number (exponential in $\mathcal{N}$) of possible patterns in the database. A common characteristic of these approachable databases is the distribution does not qualitatively change when a random error is added to each record with some small probability. This noise is often present naturally in real-world databases.

3.3 A Simple Distribution Example

Consider as a simple test distribution, the distribution created by the following algorithm for creating a single record:

1. Pick a number between 1 and $\mathcal{M}$ equiprobably.
2. Set each of $\mathcal{N}$ attribute values to this number.
3. For each attribute, with probability ϵ change the attribute value to one of the other $\mathcal{M}-1$ numbers, choosing each number with equal probability.

Due to the symmetry of the distribution, the order of the attributes in a pattern does not affect the probability. Thus the probability can be written as a function of $n_1, n_2, \ldots n_{\mathcal{M}}$ where n_i is the number of attributes with value i. $\mathcal{N}$ is the total number of attributes, therefore $\mathcal{N} = n_1 + n_2 + \ldots n_{\mathcal{M}}$. The probability of a pattern is

$$p = \frac{1}{\mathcal{M}} \sum_{i=1}^{\mathcal{M}} (1-\epsilon)^{n_i} \left(\frac{\epsilon}{\mathcal{M}-1}\right)^{\mathcal{N}-n_i} .$$

This distribution cannot be perfectly represented by low order patterns. By examining the probability estimates from this distribution we can see how the Gibbs energy transformation effectively models noisy high order features using low order patterns.

As a specific test problem let $\mathcal{N} = 5$, $\mathcal{M} = 10$, and $\epsilon = .6$. We will calculate the probability that the last attribute has value 1 given that all other attributes have value 1. With "–" indicating that a particular attribute can take on any value, the conditional probability can be written as

$$p(1,1,1,1,1|1,1,1,1,-) = \frac{p(1,1,1,1,1)}{p(1,1,1,1,1)+p(1,1,1,1,2)+\cdots+p(1,1,1,1,10)} \approx .3977 .$$

Figure 3–1 shows the experimental results for this test problem. Figure 3–2 shows the results using the small sample modification described in Section 2.5.

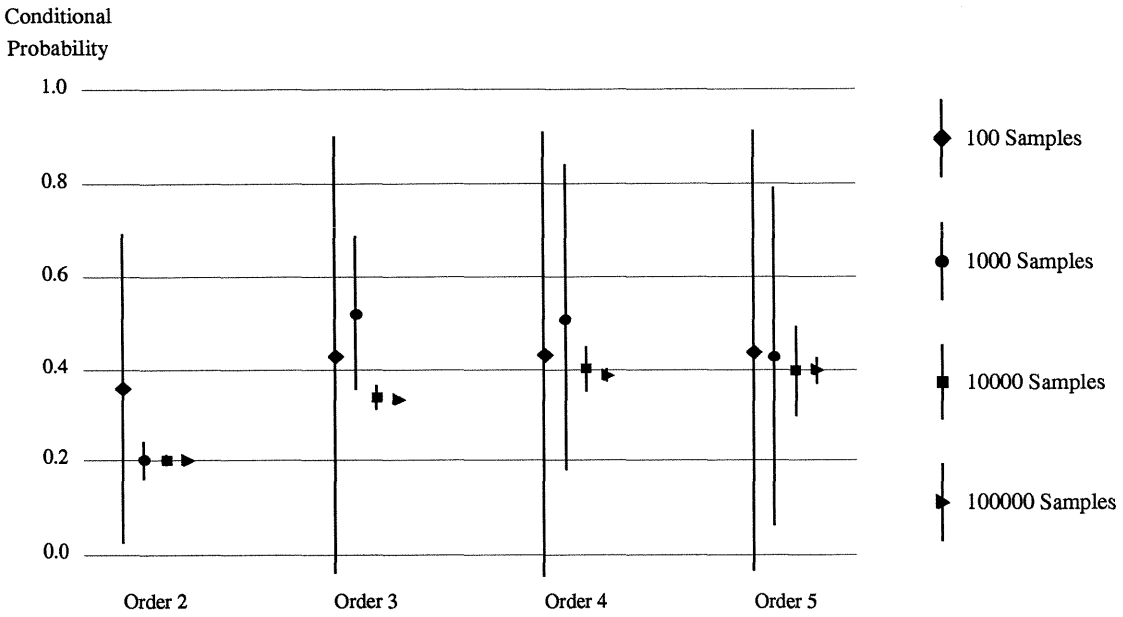

Figure 3–1

Probability Estimates by Sample Size and Model Order

Mean Value plus and minus One Standard Deviation

Unconditioned Probability is .001025

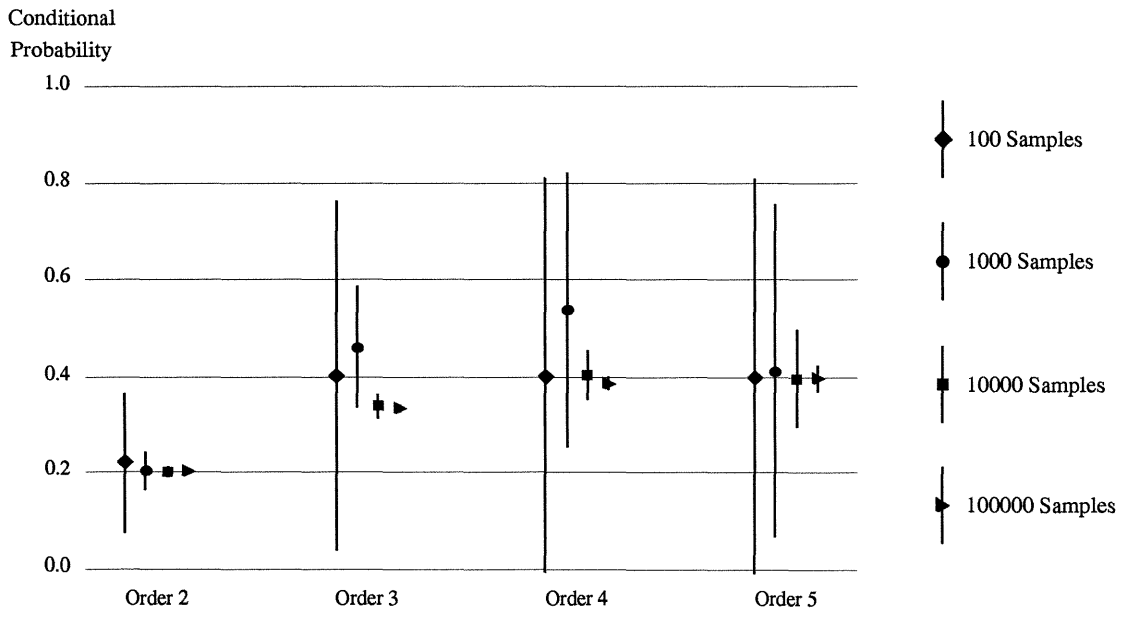

Figure 3–2

Probability Estimates Using Small Sample Statistics

SSF = 1

The experiment was to fix $\mathcal{R}$ and $\mathcal{D}$ while calculating the conditional probability (using the Gibbs model) from a large number of databases independently created using the algorithm described above. Each database resulted in a single probability estimate. The mean value and standard deviation for these estimates from the different databases were calculated. The graph shows a bar from the mean value plus one standard deviation to the mean value minus one standard deviation. The mean value is identified with a marker. The graph shows the mean value and deviation for 5 different sample sizes and 4 different values of $\mathcal{D}$. Let $p_{\mathcal{D}}$ be the abbreviation for the conditional probability estimate when patterns of order greater than $\mathcal{D}$ are assigned zero potential. For the case of zero sampling error (for large $\mathcal{R}$), the conditional probability estimates for the simple test distribution can be calculated directly from equations (3–3), (3–1), and (3–2). Due to the symmetry in the distribution (3–3), the formula for the estimated odds ($\frac{p}{1-p}$) takes on a simple form (use the result of Appendix 2.B in simplifying the expression):

$$\frac{p_1}{1-p_1} = \left(\frac{1}{9}\right) \approx .111$$

$$\frac{p_2}{1-p_2} = \left(\frac{p_1}{1-p_1}\right) \cdot \left(\frac{p(1,1,-,-,-)}{p(1,2,-,-,-)}\right) \approx .111 \cdot 2.25$$

$$\frac{p_3}{1-p_3} = \left(\frac{p_2}{1-p_2}\right) \cdot \left(\frac{p(1,1,1,-,-)p(1,2,-,-,-)}{p(1,1,2,-,-)p(1,1,-,-,-)}\right) \approx .111 \cdot 2.25 \cdot 2.0$$

$$\frac{p_4}{1-p_4} = \left(\frac{p_3}{1-p_3}\right) \cdot \left(\frac{p(1,1,1,1,-)p(1,1,2,-,-)}{p(1,1,1,2,-)p(1,1,1,-,-)}\right) \approx .111 \cdot 2.25 \cdot 2.0 \cdot 1.26$$

$$\frac{p_5}{1-p_5} = \left(\frac{p_4}{1-p_4}\right) \cdot \left(\frac{p(1,1,1,1,1)p(1,1,1,2,-)}{p(1,1,1,1,2)p(1,1,1,1,-)}\right) \approx .111 \cdot 2.25 \cdot 2.0 \cdot 1.26 \cdot 1.05 \quad .$$

Since p_5 has zero modeling error, the modeling error $e_{\mathcal{D}}$ in the order d estimate of the log odds can be calculated ($e_{\mathcal{D}} = \ln \frac{1-p_5}{p_5} - \ln \frac{1-p_{\mathcal{D}}}{p_{\mathcal{D}}}$):

$$e_1 = \ln(1.05) + \ln(1.26) + \ln(2.0) + \ln(2.25)$$

$$e_2 = \ln(1.05) + \ln(1.26) + \ln(2.0)$$

$$e_3 = \ln(1.05) + \ln(1.26)$$

$$e_4 = \ln(1.05)$$

$$e_5 = 0 \quad .$$

Written in this way, it is clear the incremental change in modeling error with each step increase

of $\mathcal{D}$ is strictly decreasing in magnitude. This is true for the entire class of distributions formed by varying the number of attributes $\mathcal{N}$ and the noise parameter of ϵ used in this example.

The log odds calculated above correspond to the following probabilities: $p_1 = \frac{1}{10}$, $p_2 = \frac{1}{5}$, $p_3 = \frac{1}{3}$, $p_4 = \frac{29}{75}$, $p_5 = \frac{173}{435} \approx .3977$. Figure 3–1 clearly shows a convergence to these values as $\mathcal{R}$ increases. For fixed $\mathcal{R}$ and small standard deviation, the experimental deviation is approximately proportional to $\mathcal{D}$. For fixed $\mathcal{D}$ and small standard deviation the experimental deviation is proportional to $\mathcal{R}^{-.5}$.

The correct (zero sampling error and zero modeling error) probability is approximately .3977. This is the ratio of two small probabilities. A large number of samples are required to estimate the conditional probability because the unconditioned probability $p(1,1,1,1,1) = .001025$ is such a low probability event. The graph of Figure 3–1 shows how the choice of $\mathcal{D}$ (when $\mathcal{R}$ is limited) involves a tradeoff between bias and variance. For small $\mathcal{D}$ the model uses the probability estimates of a small number of relatively high probability events. Small databases drawn from the test distribution will consistently give similar probability estimates (low variance). For large $\mathcal{D}$, the model uses parameter estimates taken from the frequencies of relatively low probability events. The probability of these events falls exponentially as $\mathcal{D}$ increases. Many more samples from the database are required to estimate these low probabilities. Thus for fixed $\mathcal{R}$ the variance, or sampling error, tends to increase with increased D. This is shown clearly in Figure 3–1. For fixed $\mathcal{R}$, increasing $\mathcal{D}$ causes an increase in the spread of the probability estimates. Figure 3–2 shows the same experimental test as in Figure 3–1 except the small sample modification described in Section 2.5 was used with small sample factor $SSF = 1$. As expected, the small sample modification reduces the variance of the estimates. This improvement in the consistency of the estimates comes at the cost of a slight bias of the estimates toward the lower order estimates. For example, the order 3 estimates with $SSF = 1$ are all less than the order 3 estimates calculated with no small sample modification ($SSF = 0$). This is because the small sample modification shifts the order 3 estimate towards the order 2 estimate.

3.4 Comparison with Conditional Independence Estimation

Another technique for estimating the marginal probability of an attribute given an assigned value for other attributes, is to assume that the other attributes are independent when the value of the unknown attribute is fixed. This is the conditional independence (CI) assumption. It is a model based on second order statistics, so it can be directly compared to the Gibbs model with $\mathcal{D} = 2$. The CI estimate of the conditional probability can be calculated using Bayes rule. Let x_i indicate a particular value for attribute i. Take, as in the example distribution, a distribution with $\mathcal{N} = 5$:

$$p(x_5|x_1, x_2, x_3, x_4) = \frac{p(x_1, x_2, x_3, x_4|x_5)p(x_5)}{p(x_1, x_2, x_3, x_4)}.$$

Using conditional independence (as in Section 1.1),

$$p(x_5|x_1, x_2, x_3, x_4) \stackrel{\text{CI}}{=} \frac{p(x_1|x_5)p(x_2|x_5)p(x_3|x_5)p(x_4|x_5)p(x_5)}{p(x_1, x_2, x_3, x_4)}.$$

The conditional probability distribution for x_5 can be calculated without direct knowledge of the fourth order pattern $p(x_1, x_2, x_3, x_4)$ since

$$\sum_i p(x_5 = i|x_1, x_2, x_3, x_4) = 1 \quad .$$

For the test problem described in Section 3.3, this conditional independence estimate of the odds of the conditional probability is

$$\frac{p}{1-p} = \left(\frac{1}{9}\right) \cdot \left(\frac{p(1, 1, -, -, -)}{p(1, 2, -, -, -)}\right)^4 \approx .111 \cdot 25.63 \quad .$$

This corresponds to the inaccurate probability estimate of $p \approx .74$ (the correct probability is $\approx .3977$). Certainly, the poor estimate is due to the assumption of conditional independence being incorrect for this example. The conditional independence assumption can lead to poor estimation when the database consists of noisy high order patterns.

3.5 Analysis of Classification from the Example Distribution

If the Gibbs model probability distributions are used simply to choose the most probable value for a single attribute, the system is being used as a classifier. This classifier performs

well on noisy discrete valued databases. In this section we will show that the Gibbs model classifier (neglecting sampling error) is optimal for the test distribution described above. The classifier is optimal because it always chooses the class that results in the greatest estimated probability pattern. For the test distribution this pattern will be shown to also have the greatest actual probability (assuming as in Section 3.3 that all patterns of order $\mathcal{D}$ and less are stored).

Let $\mathcal{M}$ be the number of values for each attribute ($\mathcal{M} = 10$ for the results shown in Figure 3–1). Let ϵ be the probability of an error being introduced to a single attribute ($\epsilon = .6$ in the example shown in Figure 3–1). The the probability can be written as a function of $n_1, n_2, \ldots n_{\mathcal{M}}$ where n_i is the number of attributes with value i. Let $\mathcal{N}$ be the total number of attributes so that $\mathcal{N} = n_1 + n_2 + \ldots n_{\mathcal{M}}$. The probability of a pattern with $n_1, n_2, \ldots n_{\mathcal{M}}$ is

$$p = \frac{1}{\mathcal{M}} \sum_{i=1}^{\mathcal{M}} (1-\epsilon)^{n_i} \left(\frac{\epsilon}{\mathcal{M}-1}\right)^{\mathcal{N}-n_i}.$$

Without loss of generality, assume n_1 is larger than all other n_i values. Let p' be the probability of the pattern that results if we reduce n_1 by one and increase n_2 by one:

$$\begin{aligned} p' = & \frac{1}{\mathcal{M}} (1-\epsilon)^{n_1-1} \left(\frac{\epsilon}{\mathcal{M}-1}\right)^{\mathcal{N}-n_1+1} \\ & + \frac{1}{\mathcal{M}} (1-\epsilon)^{n_2+1} \left(\frac{\epsilon}{\mathcal{M}-1}\right)^{\mathcal{N}-n_2-1} \\ & + \frac{1}{\mathcal{M}} \sum_{i=3}^{\mathcal{M}} (1-\epsilon)^{n_i} \left(\frac{\epsilon}{\mathcal{M}-1}\right)^{\mathcal{N}-n_i}. \end{aligned}$$

The ratio can be written:

$$\frac{p}{p'} = \frac{(1-\epsilon)^{n_1} \left(\frac{\epsilon}{\mathcal{M}-1}\right)^{\mathcal{N}-n_1} + (1-\epsilon)^{n_2} \left(\frac{\epsilon}{\mathcal{M}-1}\right)^{\mathcal{N}-n_2} + C}{(1-\epsilon)^{n_1-1} \left(\frac{\epsilon}{\mathcal{M}-1}\right)^{\mathcal{N}-n_1+1} + (1-\epsilon)^{n_2+1} \left(\frac{\epsilon}{\mathcal{M}-1}\right)^{\mathcal{N}-n_2-1} + C},$$

where C is a constant with respect to n_1 and n_2: $C = \sum_{i=3}^{\mathcal{M}} (1-\epsilon)^{n_i} \left(\frac{\epsilon}{\mathcal{M}-1}\right)^{\mathcal{N}-n_i}$.

Let $h(n_i) = (1-\epsilon)^{n_i} \left(\frac{\epsilon}{\mathcal{M}-1}\right)^{\mathcal{N}-n_i}$. Then

$$\frac{p}{p'} = \frac{h(n_1) + h(n_2) + C}{h(n_1 - 1) + h(n_2 + 1) + C}.$$

If $\epsilon < \frac{\mathcal{M}-1}{\mathcal{M}}$ then $h(n_1) \geq h(n_2+1)$. For some nonnegative δ we can therefore write $h(n_1) = (1+\delta)h(n_2+1)$. Since $h(n_1)\cdot h(n_2) = h(n_2+1)\cdot h(n_1-1)$, we can also write $h(n_1-1) = (1+\delta)h(n_2)$. Then

$$\frac{p}{p'} = \frac{(1+\delta)h(n_2+1)+h(n_2)+C}{(1+\delta)h(n_2)+h(n_2+1)+C} \geq 1.0 \quad .$$

Thus $p \geq p'$ if $\epsilon < \frac{\mathcal{M}-1}{\mathcal{M}}$ and $n_1 > n_2$. Therefore for the classification problem with an unknown attribute value, the optimal class value assignment is to set the attribute value to the number appearing most often in the known attributes. No other assignment can have a higher probability of being correct if $\epsilon < \frac{\mathcal{M}-1}{\mathcal{M}}$. This is simply the assignment with the fewest number of attributes different from one of the $\mathcal{M}$ points in the distribution formed when $\epsilon = 0$; this is the sequence closest to one of the sequences $\{1,1,1,\ldots 1\},\{2,2,2,\ldots 2\},\ldots\{\mathcal{M},\mathcal{M},\mathcal{M},\ldots\mathcal{M}\}$.

The order $\mathcal{D}$ Gibbs model classifier is optimal if it always chooses the class that forms the sequence closest to one of the patterns in the $\epsilon = 0$ distribution. This will be shown by comparing two estimated probabilities. One estimate is the probability of the sequence formed by assigning the class to the value occurring most frequently in the rest of the sequence. The other estimate is the probability with the class value assigned to another attribute value. In a manner similar to the aboving discussion showing the optimal decision rule based on the true probabilities, we will show that reducing n_1 and increasing another (arbitrary) value count, say n_2, will certainly reduce the **estimated** probability.

Using the result of Appendix 2.B, the estimated probability of the Gibbs model when every pattern of order $\mathcal{D}$ and less is used can be simplified to the following form:

$$\hat{p}(\omega) = \binom{\mathcal{N}-1}{\mathcal{D}-1}^{-1} \prod_{\substack{b\subseteq T \\ |b|=\mathcal{D}}} p(\omega_b). \tag{3-4}$$

For a particular ω_b, let d_i be the number of times attribute value i occurs in the pattern ω_b such that $d_1 + d_2 + \ldots d_{\mathcal{M}} = \mathcal{D} = |b|$. The number $p(\omega_b)$ can then be written

$$p(\omega_b) = f(d_1, d_2, \ldots d_{\mathcal{D}}) \equiv \frac{1}{\mathcal{M}} \sum_{i=1}^{\mathcal{M}} (1-\epsilon)^{d_i} \left(\frac{\epsilon}{\mathcal{M}-1}\right)^{\mathcal{D}-d_i} .$$

Assume $\epsilon < \frac{\mathcal{M}-1}{\mathcal{M}}$. As shown above, of any two partial configurations ω_b and ω'_b, the one closest to one of $\{1,1,1,\ldots 1\},\{2,2,2,\ldots 2\},\ldots\{\mathcal{M},\mathcal{M},\mathcal{M},\ldots\mathcal{M}\}$ will have the greater probability. As before, let n_i be the number of times attribute value i occurs in the pattern ω, so that $n_1 + n_2 + \ldots n_{\mathcal{M}} = \mathcal{N}$. Thus a given set of d_i values occurs in the probability estimation formula exactly G times where

$$G = \prod_{j=1}^{\mathcal{M}} \binom{n_j}{d_j}.$$

Therefore the formula for the estimated probability can be written as a function of the order $\mathcal{D}$ true probabilities from the distribution;

$$\hat{p}(\omega) = \sum_{\substack{d_1+\ldots+d_{\mathcal{M}}=\mathcal{D} \\ 0\le d_i \le n_i}} \binom{\mathcal{N}-1}{\mathcal{D}-1}^{-1} \prod_{j=1}^{\mathcal{M}} \binom{n_j}{d_j} \; f(d_1, d_2, \ldots d_{\mathcal{D}}). \tag{3-5}$$

The first product is taken over every set of integer d_i values such that $d_1 + \ldots + d_{\mathcal{M}} = \mathcal{D}$ and $0 \le d_i \le n_i$. Assume, without loss of generality, that the maximum n_i value is n_1. If n_1 is reduced by 1 and another n_i value (without loss of generality say n_2) is increased by 1, then some factors in the outer product of (3–5) will increase and some will decrease. The factors of increasing probability will be those with $d_2 > d_1$. The factors with decreasing probability will be those with $d_2 < d_1$. This follows from the result above which shows that the more probable of a pair of sequences is the one closest to one of $\{1,1,1,\ldots 1\},\{2,2,2,\ldots 2\},\ldots\{\mathcal{M},\mathcal{M},\mathcal{M},\ldots\mathcal{M}\}$. For every factor with $l = d_1 + d_2$ and $d_2 > d_1$, there is an identical factor with the same l but with $d_2 < d_1$. This is by the assumption that $n_1 > n_2$. The partial product π of these paired factors is

$$\pi = C \cdot \binom{n_1}{d_1}\binom{n_2}{d_2} f(d_1, d_2, \ldots d_{\mathcal{D}}) \binom{n_1}{d_2}\binom{n_2}{d_1} f(d_2, d_1, \ldots d_{\mathcal{D}}),$$

where C is a constant with respect to n_1 and n_2. Now if n_1 is changed to $n_1 - 1$ and n_2 changed to $n_2 + 1$, the partial product changes to

$$\pi' = C \cdot \binom{n_1-1}{d_1}\binom{n_2+1}{d_2} f(d_1, d_2, \ldots d_{\mathcal{D}}) \binom{n_1-1}{d_2}\binom{n_2+1}{d_1} f(d_2, d_1, \ldots d_{\mathcal{D}}).$$

It is easy to show that the ratio of the two partial products is greater than or equal to 1 if $n_1 > n_2$:

$$\frac{\pi}{\pi'} = \left(\frac{n_1}{n_2+1}\right)^2 \left(\frac{n_1-d_2}{n_2+1-d_2}\right)\left(\frac{n_1-d_1}{n_2+1-d_1}\right) \ge 1.0 \quad .$$

The partial product included every factor from (3–5) that increases when n_2 is incremented and n_1 decremented. Since $\pi \geq \pi'$, the net result is that the probability estimate cannot increase when n_2 is incremented and n_1 decremented. In fact, the estimate can remain unchanged only if $n_1 = n_2 + 1$. Thus the maximum probability estimate will occur by setting the class value equal to the value that occurs most often in the remainder of the sequence. If two sequences occur equally often, then either class value results in the same probability estimate. This is identical to the class choice that results from choosing the greatest actual probability input. Thus the Gibbs model estimate is optimal for the test distribution.

3.6 Empirical Results

This section describes the results of classification tests using the Gibbs model. The probability model is used as a classifier by calculating the probabilities of each unknown class value together with the known attribute values. The most probable combination is then chosen as the predicted class. Used as a classifier, the Gibbs model tied or outperformed published results on a variety of databases. Table 3–1 outlines these results on seven datasets available over the Internet through anonymous ftp [2] [3]. The tests were all performed with program parameters $\mathcal{D} = 4$, $\mathcal{H} = 100000$, and $\mathcal{C} = 5$, except the binary House Voting problem used $\mathcal{D} = 5$. Pattern potentials not found in the database were handled differently at classification time than in the basic Gibbs modeling algorithm used for probability estimation. Patterns of order greater than $\mathcal{D}$ were assigned zero value potential. Patterns of order less than or equal to $\mathcal{D}$ that were not found in the database of potentials were assigned a large negative value of $\mathcal{P} = -1000$. (Any negative number much larger in magnitude than the largest magnitude potentials works well.) This *penalty modification* improves classification performance by providing an extra large penalty for a classification pattern that includes low order patterns absent from the training dataset. The hash collision rejection criterion (as described in Section 2.6) was chosen to be the number of times the pattern occurred in the database. This is quickly calculated and has been found to work well with the penalty modification. As an implementation detail, patterns that do not include the classification attribute are never used in the classifier, so they are excluded in the generation of the pattern potentials.

Table 3–1, Summary of Classification Results

Database	A	C	R	Train	Test	Trials	Gibbs Rate	Compare
House Voting	16	2	435	335	100	50	95.32%	95% [4]
Letter Recog.	26	26	20000	16000	4000	3	88.93%	80% [5]
Iris	4	3	150	120	30	100	96.27%	n.a.
Iris	4	3	150	149	1	1000	97.1%	See Appendix 3.A
Breast Cancer	9	2	699	599	100	100	97.3%	n.a.
Breast Cancer	9	2	369	200	169	100	95.7%	93.7% [6]
Monk's 1	6	2	432	124	432	1	94.0%	See Appendix 3.B
Monk's 2	6	2	432	169	432	1	69.4%	"
Monk's 3	6	2	432	122	432	1	97.5%	"

A = Attribute count in the database, excluding the class attribute
C = Class count
R = Record count
Train = Number of records used to create the energy model for a single trial
Test = Number of records tested in a single trial
Trials = Number of independent train–test trials used to calculate the rate
Gibbs Rate = Gibbs energy model classification rate
Compare = Baseline classification result of other classification methods, with reference

The first dataset in Table 3–1 is the voting record of members of the U.S. House of Representatives on 16 bills from the 98th Congress (1984). The classification problem tested was to predict the party, Democratic or Republican, of the representative from their votes. The task was made difficult by Congress members who voted outside of party lines and by members of Congress who did not vote on many bills. The database of 435 records was randomly divided into 335 training records and 100 test records. The 335 voting records were used to create the model. The model was then used to classify the remaining 100 records as Republican or Democrat. This was repeated 50 times, each time using a different random division of the

database into training and test record sets. The result of this experiment using the Gibbs model was an average of 95.32% correct classification. This is comparable to the best results previously reported in the machine learning literature, for example the 95% classification reported in [4].

The second dataset is the Letter Recognition dataset first used in [5]. The objective of this test was to identify a letter from features (such as the total number of pixels and the width of the character) taken from the image of the letter. The 26 attributes were created (by the author of the database) by taking the feature measurements and scaling them to create 16 attribute values for each feature. This representation was used directly in creating the Gibbs model. Notice that with this representation the Gibbs algorithm did not use the natural ordering of the attribute values $(0, 1, \ldots 15)$ in creating the model. Table 3–1 shows the experimental setup which resulted in a 88.93% correct classification rate from the Gibbs modeling algorithm. This compares favorably with the 80% classification rate reported by the author of the database. The Gibbs model results were collected from the very first experiment using the algorithm with the dataset. No difficult parameter adjustment is necessary to get the algorithm to classify at these rates.

The third dataset is the often used Iris flower dataset [7]. In this dataset the objective was to predict the type of flower from three length measurements taken from a single plant. The two interesting points to note about this dataset are that the database is very small (only 150 records) and the attributes take on real values rather than a discrete set of attribute values. Since the Gibbs modeling algorithm counts the occurrence of discrete attribute values, somehow the attribute values must be "binned" into ranges with the range used as the discrete value. Given the fixed number of records there is a bias–variance tradeoff in setting the number of bins. If very few bins are used the attribute value does not give very precise information about the length measurement, resulting in a consistent error, or bias, in the probability estimates created from the model. The use of only a few bins does, however, result in more consistent estimates (less variance) because the number of samples in each bin is larger. Once the number of bins is set, there is a problem in setting the ranges for

each bin. Another potential problem in modeling this database is that the attribute values have an ordering which is important. The basic Gibbs modeling algorithm has no way of incorporating the order of the attribute values into the model.

The binning method used for the Iris database experiment was to simply convert each real-valued attribute into a percentile ranking. Each percentile was then converted into three new attribute values. The first attribute value was the decile ranking. The decile ranking is an attribute with 10 values which indicates, within 10%, what percentage of the samples had a larger value. The second attribute value was a 3-tile ranking. This attribute indicated if the value was in the upper, middle, or lower third of values for all of the samples. The third attribute value was a 2-tile ranking, which indicated if the value is above or below the median value. This encoding of the 4 real valued attributes produced 12 discrete attributes. The overlap of the different n-tile rankings, in part, had the effect of making the bin ordering information available to the model. The 12 attributes included a great deal of redundancy because, for instance, the 2-tile ranking was completely determined by the decile ranking. This encoding technique effectively allowed the dataset to determine the bias-variance tradeoff in bin size. Probability estimation techniques that rely on assuming the attributes are independent would not work well with this redundant encoding scheme. However, the Gibbs modeling scheme effectively adds this dependency into the model.

Two separate experimental results are reported in Table 3–1. The two experiments used different sized training sets to simplify comparison with previously published results. The experiment using 149 training records can be directly compared with the results for other training algorithms tabulated in Appendix 3.A. As with the Letter Recognition classification result, the Gibbs model classification rates of 96.27% and 97.1% are were calculated from the first tests of the algorithm using the Iris dataset. There was no optimization of the parameters to improve the classification results. The excellent Gibbs model classification results are somewhat surprising given that the probability estimates generated from combinations of pairs or triplets of events must be very inaccurate from a database this small. Fortunately, as discussed in Section 3.5, the inaccuracies do not tend to change the probability ranking

of the most probable classes. A graphical comparison of the Gibbs model classifier and a nearest neighbor classifier is given by Figures 3–3 and 3–4 in Appendix 3.A.

The fourth dataset is the Wisconsin Breast Cancer Database [8]. The experiment was to predict whether or not the cells were malignant or benign from 9 attributes (Clump Thickness, Uniformity of Cell Size, Uniformity of Cell Shape, Marginal Adhesion, Single Epithelial Cell Size, Bare Nuclei, Bland Chromatin, Normal Nucleoli, Mitoses). Each attribute in the database has a value between 1 and 10. These attribute values were used directly in the Gibbs modeling algorithm (thus the natural ordering of the attribute values 1,2,...10 was not used in the model). Two experiments were performed. One experiment used the full database dividing it up repeatedly into training sets of 599 instances and test sets of 100 instances. This produced a classification rate of 97.3%. The second test used a modified smaller dataset which has been used previously in the literature. This test setup of 200 training instances and 169 test instances was chosen to duplicate the experiment performed in [6] which reported the best classification rate of 93.7% from 4 different algorithms. The Gibbs model gave a classification rate averaging 95.7% on the first test of the algorithm with this data.

The final datasets tested are Monk's-1, Monk's-2, and Monk's-3. The Monk's problems are three problems in Boolean logic used to compare learning algorithms. A full description of the Monk's problems can be found in reference [9]. The Monk's problems are unlike the other datasets in that the training data and test data are provided separately by the creators of the database. Thus the experiment was only 1 trial. The attractive property of this dataset is that dozens of classification results have been published comparing a variety of algorithms [9]. Many of the tests were performed by the original creators of the algorithms. The three Monk's problems are samples from a discrete space of 6 attributes. Each sample is labeled TRUE or FALSE. The labels TRUE and FALSE are determined by three different Boolean functions for the three Monk's problems. The third Monk's problem has an additional complication in that noise was added to the training data so that 6 of the inputs are actually inconsistent with the Boolean function. This noise makes Monk's-3 the ideal type of problem for the probabilistic approach of the Gibbs modeling algorithm. Figure 3–5 in Appendix 3.B shows both the

input data and the output results for the Monk's-3 dataset. The test result of 97.5% can be compared directly with the classification results shown in Appendix 3.B, Table 3–3 (reported in [9]). The test results for the noiseless Monk's-1 and Monk's-2 dataset were 94.0% and 69.4% respectively. For the noiseless case most of the non-probabilistic classification models outperformed the results from the Gibbs model. For the case with added noise (Monk's-3) the Gibbs modeling algorithm outperformed most of the well known algorithms listed in Table 3–3. Recent versions of the AQ family of algorithms that specifically attempt to model the data as compact Boolean functions were the only algorithms to outperform the basic Gibbs modeling algorithm on the Monk's problem with added noise.

3.7 Summary of Chapter 3

- The bias error in estimated probabilities using the Gibbs distribution goes to zero when all possible potentials are used in the probability model.
- The variance of the probability estimates using the Gibbs model increases when the maximum pattern order $\mathcal{D}$ is increased.
- For the symmetric test distribution, the Gibbs model used as a classifier produces the optimal decision rule even if the maximum order $\mathcal{D}$ is less than the number of attributes $\mathcal{N}$.
- Test results show the Gibbs model performs well as a classifier without difficult adjustment of parameters or iterative training.

3.8 Historical and Bibliographic Notes

The convergence of sample means to the underlying expected value for the discrete case is well covered in Feller Volume I [10]. Since the Gibbs distribution cannot represent zero probability events, an extremely low probability is assigned by the use of the penalty term $\mathcal{P}$ described in Section 3.6. This is similar to methods used in optimization where disallowed solutions are incorporated into the model by assigning them a very high cost. For example Hopfield and Tank's neural network implementation of the traveling salesman problem used a large energy term to disallow illegal paths of the salesman [11]. The general problem of binning real numbers for use in probability estimation was studied by Hughes who examined the tradeoffs in choosing the correct number of bins [12] [13]. The Iris dataset measurements were made by E. Anderson and used by a variety of authors including Fisher in his study of discriminant analysis [13]. Duda and Hart (in discussing clustering techniques) used the two-dimensional cross section of the Iris data show in Figures 3–3 and 3–4 [13].

3.9 References

[1] M. Minsky and S. Papert, *Perceptrons*, Cambridge MA: MIT Press, 1969.

[2] P. Murphy, and D. Aha, *UCI Repository of Machine Learning Databases* [Machine-readable data repository at ics.uci.edu in directory /pub/machine-learning-databases]. Irvine, CA: University of California, Department of Information and Computer Science, 1992.

[3] J. Pollack, *Ohio State Repository of Neural Network Papers and Information* [Machine-readable data repository at archive.cis.ohio-state.edu, the Monk's datasets are in file pub/neuroprose/thrun.comparison.dat.Z]. Columbus, OH: Ohio State Department of Computer and Information Science, 1992.

[4] Schlimmer, J. C., "Concept Acquisition Through Representational Adjustment," UCI Ph.D. Thesis 1987.

[5] P. W. Frey and D. J. Slate, "Letter Recognition Using Holland-style Adaptive Classifiers," *Machine Learning*, **6**, pp.161–182, 1991.

[6] J. Zhang, "Selecting Typical Instances in Instance-Based Learning," in *Proceedings of the Ninth International Machine Learning Conference* Aberdeen, Scotland, pp.470–479, San Mateo CA: Morgan Kaufmann, 1992.

[7] S. Weiss, and I. Kapouleas, "An Empirical Comparison of Pattern Recognition, Neural Nets, and Machine Learning Classification Methods," in *Proceedings of the 11th International Joint Conference on Artificial Intelligence* Vol. 1, Los Gatos, CA: Morgan Kaufmann, pp.781–787, 1992.

[8] W.H. Wolberg and O.L. Mangasarian, "Multisurface Method of Pattern Separation for Medical Diagnosis Applied to Breast Cytology," *Proceedings of the National Academy of Sciences, U.S.A.*, **87**, pp.9193–9196, 1990.

[9] S. Thrun et al., "The MONK'S Problems, A Performance Comparison of Different Learning Algorithms," Carnegie Mellon University Technical Report CMU–CS–91–197, December 1991.

[10] W. Feller, *An Introduction to Probability Theory and Its Applications*, New York: Wiley, 1957.

[11] J. Hopfield and D. Tank, "Neural Computation of Decisions in Optimization Problems," *Biological Cybernetics*, **52**, pp.141–152, 1986.

[12] G.F.Hughes , "On the Mean Accuracy of Statistical Pattern Recognizers," *IEEE Trans. Info. Theory*, **IT–14**, pp.55–63, 1968.

[13] R. Duda and P. Hart, *Pattern Classification and Scene Analysis*, New York: Wiley, 1973.

Appendix 3.A
Performance Results of Classifiers on the Iris Dataset

Weiss and Kapouleas tested a variety of classification algorithms on the Iris dataset. See reference [7] for a full explanation of their test procedures. The test method used was the "leave one out method," meaning they trained on all but one of the 150 records and then tested on the one remaining record. The rates were averaged over 5 trials. For purposes of comparison these results are shown below in Table 3–2. As shown in Table 3–1, the Gibbs model produces a classification accuracy of 97.1%.

Figure 3–3 shows the results of a test of the Iris database using only the first two attributes for classification. The Gibbs model classifier was trained on the entire 150 training points from the Iris dataset. An X-Y location in the graph represents a particular input to the Gibbs model. By testing the model with a variety of two-dimensional inputs, it is possible to map out regions, called decision regions, where all inputs are assigned to the same class. The graph is shaded to show the decision regions. The nearest neighbor classification algorithm is the simple procedure of assigning an unknown class to be the same class as the nearest sample in the training database. Figure 3–4 graphs the decision regions for the nearest neighbor algorithm trained with the Iris dataset.

Table 3–2, Performance Comparison on the Iris Dataset

From "An Empirical Comparison of Pattern Recognition, Neural Nets, and Machine Learning Classification Methods," by S. Weiss and I. Kapouleas [7], used with permission.

Classification Method	*Test Rate*
Linear	98.0%
Quadratic	97.3%
Nearest Neighbor	96.0%
Bayes Independence	93.3%
Bayes 2nd Order	84.0%
Neural Net (BP)	96.7%
PVM rule	96.0%
Optimal Rule Size 2	98.0%
CART Tree	95.3%

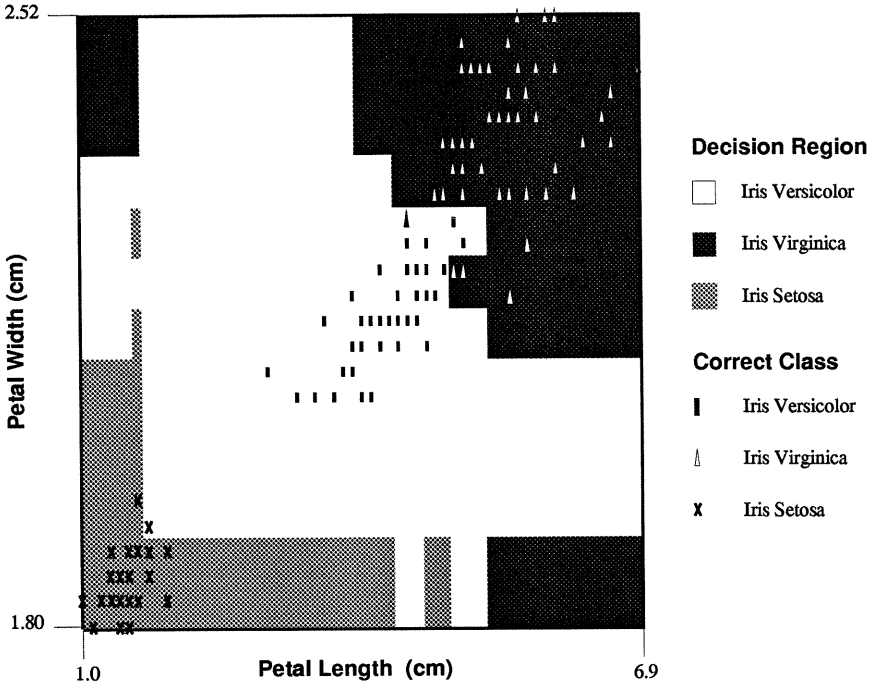

Figure 3–3, Gibb's Model Decision Regions

Iris Dataset using Two Attributes

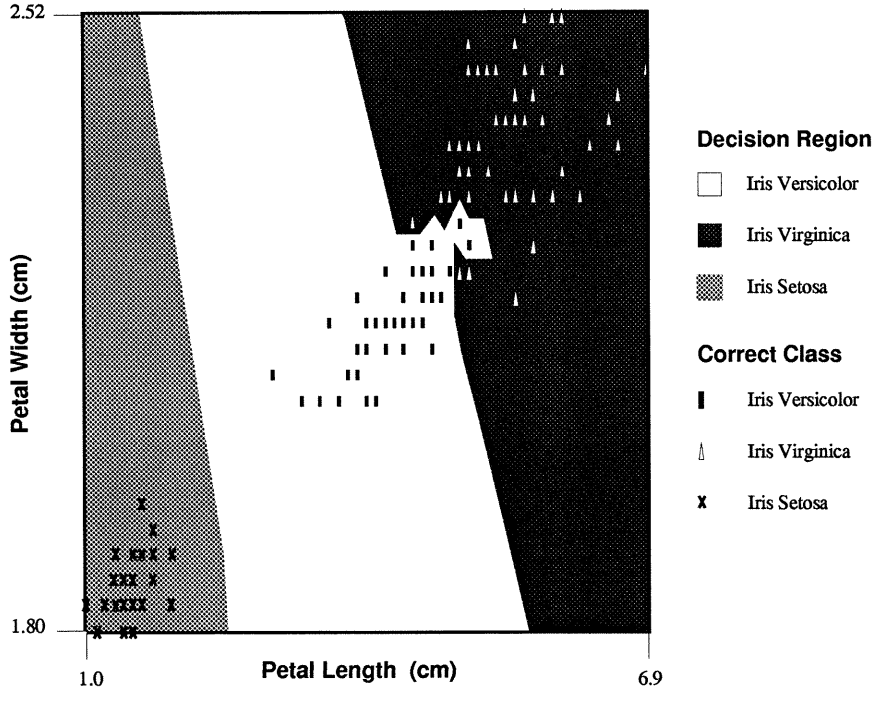

Figure 3–4, Nearest Neighbor Decision Regions

Iris Dataset using Two Attributes

Appendix 3.B

Performance Results of Classifiers on the Monk's Dataset

Figure 3–5 shows the input and output dataset for the Monk's-3 problem described in Section 3.6. Eleven of the test point results are incorrect, giving a 97.5% classification rate. This can be compared with the classification rates for a variety of other methods shown in Table 3–3.

Table 3–3, Monks Comparison of Classification Results

From "The MONK'S Problems, A Performance Comparison of Different Learning Algorithms," by S. Thrun et al., Carnegie Mellon University CMU-CS-91-197, December 1991 [9], used with permission.

Algorithm	Reported by	Monks 1	Monks 2	Monks 3
AQ17-DCI	J.Bala et al.	100%	100%	94.2%
AQ17-HCI	"	100%	93.1%	100%
AQ17-FCLS	"		92.6%	97.2%
AQ14-NT	"			100%
AQ15-GA	"	100%	86.8%	100%
Assistant Professional	B.Cestnik et al.	100%	81.3%	100%
mFOIL	S.Džeroski	100%	69.2%	100%
ID5R	W.Van de Velde	81.7%	61.8%	
IDL	"	97.2%	66.2%	
ID5R-hat	"	90.3%	65.7%	
TDIDT	"	75.7%	66.7%	
ID3	J.Kreuziger et al.	98.6%	67.9%	94.4%
ID3, no windowing	"	83.2%	69.1%	95.6%
ID5R	"	79.7%	69.2%	95.2%
AQR	"	95.9%	79.7%	87.0%
CN2	"	100%	69.0%	89.1%
CLASSWEB .10	"	71.8%	64.8%	80.8%
CLASSWEB .15	"	65.7%	61.6%	85.4%
CLASSWEB .20	"	63.0%	57.2%	75.2%
PRISM	S.Keller	86.3%	72.7%	90.3%
ECOWEB leaf prediction	Y.Reich et al.	71.8%	67.4%	68.2%
ECOWEB l.p. & info.util.	Y.Reich et al.	82.7%	71.3%	68.0%
Backpropagation	S.Thrun	100%	100%	93.1%
Backprop. with Wt.Decay	"	100%	100%	97.2%
Cascade Correlation	S.Fahlman	100%	100%	97.2%

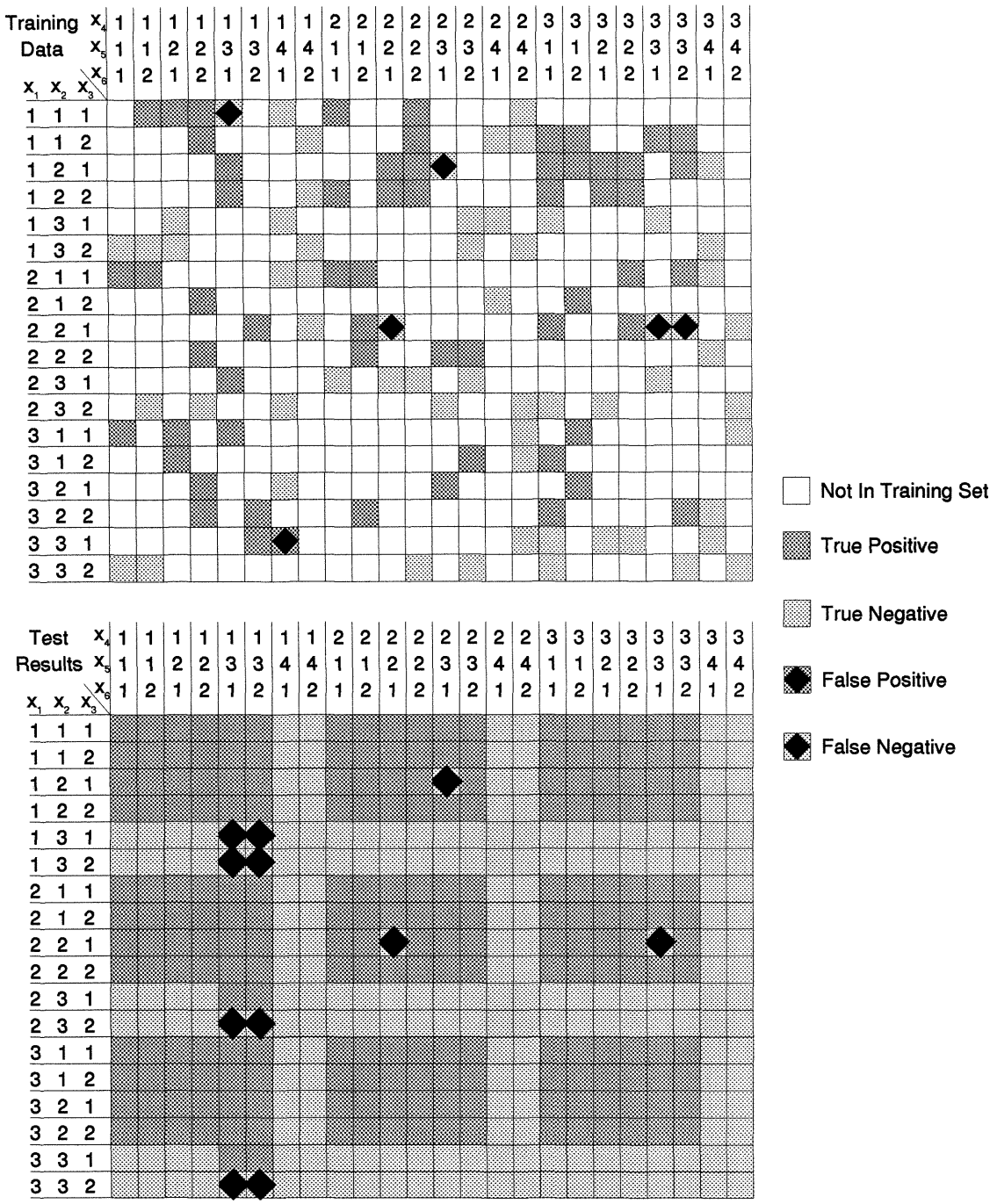

Figure 3–5, Monk's-3 – Input Data and Test Results

The correct Boolean function is $(x_5 = 3 \text{ AND } x_4 = 1) \text{ OR } (x_5 \neq 4 \text{ AND } x_2 \neq 3)$. The order 3 Gibbs model was trained on 122 examples — 6 of them misclassified. Test results were 97.5% correct of the 432 combinations of values for 6 attributes.

Chapter 4
On Loss Functions that Minimize to Expected Values

4.1 Background

A *loss function*, or *objective function*, is a function used to compare parameters when fitting a mathematical model to data. For example, in linear regression, the problem is to find the line $f(x)$ that best "fits" a collection of data points x_i, y_i $(i = 1, \ldots N)$. The line is a model M, which gives an estimate $\hat{y} = f(x)$ of the value y for each value x. The parameters to the model, $\underline{\theta}$, are the two values required to describe a line. Normally in linear regression the squared error (SE) loss function $L(y, \hat{y}) = (y - \hat{y})^2$ is used, meaning that the "best fit" is considered to be the $\underline{\theta}$ that minimizes the average loss: $\frac{1}{N}\sum_{i=1}^{N} L(y, \hat{y})$. In the context of neural networks, the values $\underline{\theta}$ are the network's weights and thresholds. The estimate $\hat{y}$ is the network's output. We will call this method of finding the minimum average loss the "training algorithm," and the $\{\underline{x}_i, y_i\}$ values the "training data." Without loss of generality, y is assumed to be scalar. Table 4–1 summarizes our notation. Figure 4–1 diagrams the relationship between the introduced symbols.

Table 4–1, Summary of Notation

Symbol	*Explanation*	*Neural Network Example*
θ	Parameters to a model.	Weights and thresholds.
x_i, y_i	Training data: x_i is input, y_i is output for sample i. The subscript is often omitted.	Training Data
$M(\theta)$	Model, a set of mappings $x \rightarrow \hat{y}$ parameterized by θ.	The set of all possible functions that a network architecture can implement.
$f(x)$	The mapping produced by the model with a given parameter set θ.	A network with fixed weights and fixed thresholds.
$\hat{y}_i$	For some given x_i the value $f(x_i)$. This value is assumed to be scalar.	The network output.
$L(y_i, \hat{y}_i)$	The error between the desired and actual output.	Squared Error (for example)

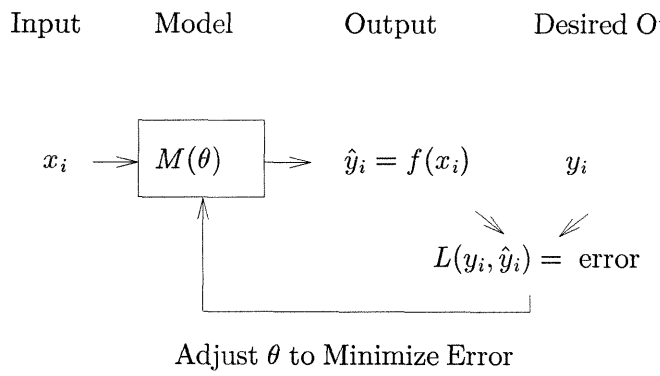

Figure 4–1, Training Diagram

4.2 Probability Estimation

Consider estimating the conditional mean, $E[y|\underline{x}]$, the mean value of y taken over all training samples with a given value for $\underline{x}$. For the case where y is a scalar binary value, $y \in \{0, 1\}$:

$$E[y|\underline{x}] = p(y = 1|\underline{x}).$$

In classification problems $p(y = 1|\underline{x})$ is sometimes called the *a posteriori* probability of y, the probability of $y = 1$ after the evidence $\underline{x}$ is known. In this chapter we deal with systems that estimate $E[y|\underline{x}]$, and include systems for estimating *a posteriori* probabilities as a special case. The training data is assumed to consist of independent samples from some underlying probability distribution over $\underline{x}, y$. There are two separate reasons why our estimate of $E[y|\underline{x}]$ will be different from the "true" probabilities which exist in the underlying distribution. The first reason is due to the limitation of the dataset. If the dataset is infinite, by the law of large numbers we can get an arbitrarily accurate estimate of the conditional probabilities $E[y|\underline{x}]$ by counting the frequency of occurrence in the training dataset. Since the dataset is finite, we only have a sampled estimate of this true underlying probability. This sampling error in

the estimation is not considered here. Instead the probabilities referred to here will be the relative frequencies found in the training set. The second reason for error in estimation is due to limitations of the model. A model may not have a parameter set $\underline{\theta}$ such that the function $f(\underline{x}) = E[y|\underline{x}]$ can be perfectly represented. A *sufficiently powerful* model $M(\underline{\theta})$ is one that, for some $\underline{\theta}$, is capable of producing $\hat{y}(\underline{\theta}, \underline{x}) = E[y|\underline{x}]$. For a sufficiently powerful model, the average loss must be minimized at $\hat{y} = E[y|\underline{x}]$:

$$\min_{\underline{\theta}} E[L(y, \hat{y})|\underline{x}] \text{ is achieved when } \hat{y}(\underline{\theta}, \underline{x}) = E[y|\underline{x}].$$

We assume the values of the target y in the training sets are bounded such that $0 \leq y \leq 1$. This choice of upper and lower bounds for y is consistent with model outputs that represent probabilities. The results can be extended to problems with other upper and lower bounds by appropriately scaling and shifting the y values. The unbounded case is not studied here, but it can be shown that if no restriction is placed on the distribution of y, then no loss function can be guaranteed to minimize to the expected value of y.

We will find it useful to rewrite the "minimization at $\hat{y} = E[y|x]$" requirement in a way that explicitly shows the probability distribution used to calculate the expected value. Call this distribution $p(y)$. Until Section 4.7, all distributions and expected values will be "given $\underline{x}$." To simplify the notation, the explicit reference to the input $\underline{x}$ will be dropped in Sections 4.2 through 4.6. We can now replace the expected value with a definite integral, giving the formal definition:

$$\begin{gathered} \forall\, p(y) \text{ s.t. } \int_0^1 p(y)\, dy = 1 \\ \min_{0<\hat{y}<1} \int_0^1 p(y) \cdot L(y, \hat{y})\, dy \;=\; \int_0^1 p(y) \cdot L(y, \overline{y})\, dy, \end{gathered} \tag{4–1}$$

where $\overline{y} \equiv \int_0^1 p(y) \cdot y\, dy$. If $L(y, \hat{y})$ satisfies this property (4–1), we call it "P-admissible," to indicate that the loss function is admissible for use in probability estimation or expected value estimation. A sufficiently powerful model trained using a P-admissible loss function will provide an estimate $\hat{y} = E[y|\underline{x}]$. In Sections III through VI, we study the set of loss functions that are P-admissible. In Section VII, the results are extended to models that may not be sufficiently powerful to approximate $E[y|\underline{x}]$. This is the more realistic case.

4.3 A Simple Example

Consider a dataset describing a set of patients. Let:

$$x = \text{symptom of a disease,}$$

$$y = \text{presence of disease in patient (1 or 0).}$$

Our training data is a set of medical records for 100 patients, all with symptom x. 90 of these patients have disease y, 10 do not, so from this sample:

$$E[y|x] = p(y = 1|x) = \frac{90}{100} = .9 \quad .$$

If we train a model on this sequence of zeros and ones, does the model produce as an output $E[y|x] = .9$? The answer is yes, if $L(y, \hat{y})$ is P-admissible.

4.4 Two P-admissible Loss Functions

The squared error function is known to be P-admissible:

$$L_{se}(y, \hat{y}) = (\hat{y} - y)^2.$$

Least squared minimization has the property that the derivative with respect to $\hat{y}$ is a linear function. This can be used to advantage in designing computationally efficient gradient based optimization schemes (training algorithms).

A loss function commonly used in neural network training algorithms is the cross entropy function [1][2][3][4]:

$$L_{ce}(y, \hat{y}) = y \cdot \log\left(\frac{y}{\hat{y}}\right) + (1 - y)\log\left(\frac{1 - y}{1 - \hat{y}}\right).$$

The cross entropy loss function is used for maximum likelihood estimation of model parameters when the training data consists of classification labels [1]. L_{ce} is also P-admissible [2].

This is shown by finding the value of $\hat{y}$ such that $E[L_{ce}(y,\hat{y})|\underline{x}]$ is minimized. Equation (4–1) requires that the minimization solution be $\hat{y} = E[y]$. We can easily derive this by solving the minima problem:

$$\begin{aligned}\frac{\partial}{\partial \hat{y}} E[L_{ce}(y,\hat{y})] &= 0 \\ &= \frac{\partial}{\partial \hat{y}} \int_0^1 p(y) \cdot L_{ce}(y,\hat{y})\, dy \\ &= \int_0^1 p(y) \cdot \frac{\partial}{\partial \hat{y}} L_{ce}(y,\hat{y})\, dy \\ &= \int_0^1 p(y) \cdot \frac{\hat{y}-y}{\hat{y}(1-\hat{y})}\, dy \\ &= \frac{1}{\hat{y}(1-\hat{y})}\ (\hat{y} - E[y]).\end{aligned}$$

This partial derivative equals zero when $\hat{y} = E[y]$. Since the second partial derivative is positive at $\hat{y} = E[y]$, this extremum is the minimum. Thus (4–1) is satisfied and the cross entropy function is P-admissible.

4.5 Necessary and Sufficient Conditions for P-admissibility

Define $h(\hat{y}) = L(0,\hat{y})$ to be the value of the loss function when the target output y is zero. In Appendix 4.A it is shown that P-admissibility is equivalent to the restrictions (4–2) and (4–3):

$$L(y,\hat{y}) = \int h'(\hat{y}) \cdot \frac{\hat{y}-y}{\hat{y}}\, d\hat{y} + C(y), \tag{4–2}$$

$$h'(\hat{y}) > 0 \quad \text{for } 0 < \hat{y} < 1\ . \tag{4–3}$$

The prime indicates the derivative, $h'(\hat{y}) = \frac{d}{d\hat{y}} h(\hat{y})$. The value $C(y)$ is a constant with respect to $\hat{y}$. It has no effect on minimization and may be set to zero when defining a loss function. The restrictions (4–2) and (4–3) may be used to generate loss functions that minimize to the conditional expectation. Note that a P-admissible loss function may be described entirely by $h(\hat{y}) = L(0,\hat{y})$, the value of the loss function when the target $y = 0$. For example the squared error loss function has error $h(\hat{y}) = \hat{y}^2$ when the desired output is zero. Substitute this into (4–2) with $C(y) = y^2$ to find $L(y,\hat{y}) = (y-\hat{y})^2$ as expected.

4.6 A Symmetry Restriction

So far we have used P-admissibility to restrict the class of loss functions under study. A second restriction on a loss function is the condition of logical symmetry:

$$L(0, \hat{y}) = L(1, 1 - \hat{y}). \tag{4–4}$$

This condition is natural when the labels $y = 1$ and $y = 0$ are arbitrary. In the simple medical records example, we used $y = 1$ to indicate the presence of a disease. If instead we had used $y = 0$ to indicate the disease, would the results have been the same? The answer is true generally only if the loss function obeys the symmetry condition (4–4).

Symmetry can be used to greatly simplify equation (4–2). Define

$$k(\hat{y}) = \int \frac{h'(\hat{y})}{\hat{y}}\, d\hat{y}.$$

Then (4–2) may be written

$$L(y, \hat{y}) = h(\hat{y}) - y \cdot k(\hat{y}) + C(y). \tag{4–5}$$

Thus $L(1, 1 - \hat{y}) = h(1 - \hat{y}) - k(1 - \hat{y}) + C(1)$, and by definition $L(0, \hat{y}) = h(\hat{y})$. From (4–4) find $k(\hat{y}) = h(\hat{y}) - h(1 - \hat{y}) - C(1)$. Finally, substitute back into (4–5) to get the simple form:

$$L(y, \hat{y}) = h(\hat{y}) + y[h(1 - \hat{y}) - h(\hat{y})] + C_1(y).$$

We show in Appendix 4.B that symmetry further restricts the form of a P-admissible loss function. Specifically $h(\hat{y})$ must satisfy (4–6):

$$\frac{1 - \hat{y}}{\hat{y}} = \frac{h'(1 - \hat{y})}{h'(\hat{y})}. \tag{4–6}$$

Hampshire and Pearlmutter [5] independently arrived at equation (4–6) for the case where the targets are binary $\{0, 1\}$. In this chapter we show that this result applies to objective function analysis for more general distributions $p(y)$.

It follows from equation (4–6) that at least one of the following cases must be true:

$$h'(\hat{y}) \text{ has a zero at } \hat{y} = 0 \quad \text{or} \quad h'(\hat{y}) \text{ has a pole at } \hat{y} = 1.$$

The simplest functions satisfying the above restriction are:

$$h_1'(\hat{y}) = \hat{y},$$
$$h_2'(\hat{y}) = \frac{1}{1-\hat{y}}.$$

By substitution into (4–2), it is seen that h_1 defines the objective function:

$$L_1(y, \hat{y}) = .5\hat{y}^2 - \hat{y}y.$$

An objective function can be multiplied by a constant with respect to $\hat{y}$ or added to a constant w.r.t. $\hat{y}$ without changing the minimization, thus $L_1(y, \hat{y})$ is equivalent to $L_{se} = (\hat{y} - y)^2$. Similarly h_2 may be substituted into (4–2) to generate the cross entropy objective function. Hence, by applying the result of Appendix 4.B, we see that the well known loss functions L_{se} and L_{ce} are two of the most simple functions of a class which satisfy P-admissibility and the symmetry condition.

4.7 Insufficiently Powerful Models

In Section 4.2 we developed P-admissibility by considering that a sufficiently powerful model should produce conditional expected values. In practice the model may not be able to produce these ideal outputs for every different input $\underline{x}$. What if the model is not sufficiently powerful?

Let $g(\underline{x}) = E[y|\underline{x}]$. We reintroduce the explicit reference to $\underline{x}$ in the notation. From Section 4.2 we know $g(\underline{x})$ is the output of a sufficiently powerful model after training with a P-admissible loss function. Let $\hat{y}$ be the output of an "imperfect" model, one that may not be sufficiently powerful. In Appendix 4.C it is shown that if $L(y, \hat{y})$ is P-admissible then,

$$\min_{\underline{\theta}} E[L(y, \hat{y})] \text{ is equivalent to } \min_{\underline{\theta}} E[L(g(\underline{x}), \hat{y})]. \tag{4–7}$$

Here the expected value is taken over the joint distribution of $\underline{x}, y$ in the training data. In words, (4–7) says the model that is found by minimizing the expected loss, produces outputs which come as close as possible to the output of an "ideal" model. Furthermore, this "closeness" is defined by the loss function. For instance if the loss function is squared error, then a real model trained to minimize the expected squared error between y and $\hat{y}$ will produce an output for which the expected square error between $\hat{y} = f(\underline{x})$ and $E[y|\underline{x}]$ is minimized. This was shown for the squared error loss function by White [6]. Result (4–7) generalizes this property to all possible loss functions that produce estimates of the conditional expectation.

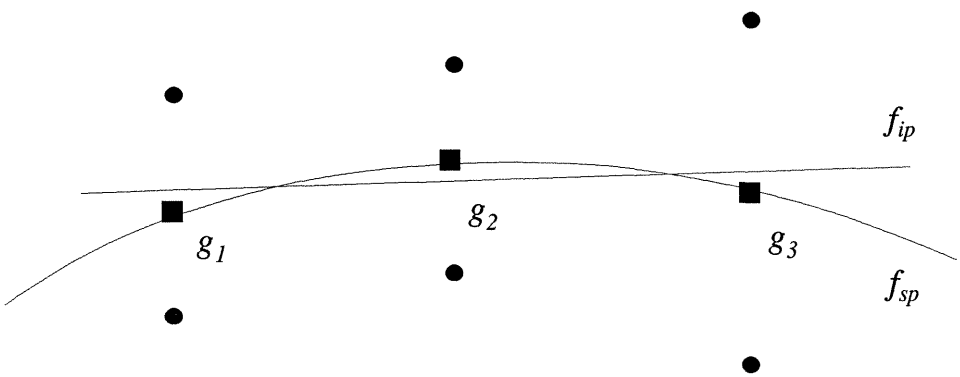

Figure 4–2, Comparison of Two Models Fitted to Six Points

Figure 4–2 shows an example graphically. Here we have 6 points of training data with three unique $\underline{x}$. The conditional expected values of y for these three inputs are shown by g_1, g_2, g_3. The figure shows a curve f_{sp} from a sufficiently powerful model. A sufficiently powerful model in this case is any set of curves (parameterized by $\underline{\theta}$) that includes a curve passing though g_1, g_2, g_3. For example, second order polynomials are sufficiently powerful for this case since there are three points. If we set the $\underline{\theta}$ by minimizing the expected value of a P-admissible loss function, then the function $f_{sp}(x)$ will pass through these points. This is the result of Section 4.2. Now suppose we fit these points with a line. A line is "insufficiently powerful" for this problem, because it does not have the 3 degrees of freedom necessary to pass through g_1, g_2, g_3. Result (4–7) says that the line will come as close as possible to the three points.

The measure of "closeness" is the same loss function that we used with the 6 training points. In other words, we would have found the same line if the training points had been g_1, g_2, g_3, rather than the 6 points. This would be false if we had selected $\underline{\theta}$ using a non-P-admissible loss function. For instance if the loss function were $L(y, \hat{y}) \equiv (y - \hat{y})^4$ then f_{sp} would not pass through the points g_1, g_2, g_3, and the line would not be a quartic error approximation of the outputs of the f_{sp} function.

4.8 Summary of Chapter 4

- We have generalized and extended previously known results on the topic of obtaining conditional estimates from a trained model. In particular, we derived necessary and sufficient conditions for an objective function to minimize to the expected value of the desired output y given an input $\underline{x}$.

- The objective function $L(y, \hat{y})$ was found to be uniquely specified by the function $L(0, \hat{y})$.

- The function $L(0, \hat{y})$ was found to satisfy further restrictions when a condition of logical symmetry is required. These restrictions and the relation between $L(y, \hat{y})$ and $L(0, \hat{y})$ define the class of all objective functions that minimize to the conditional expectation. This includes objective functions that minimize to a probability.

- The two simplest functions in this class were found to be the well-known squared error and cross entropy objective functions.

- When the model is incapable of mapping all inputs $\underline{x}$ to the ideal output $g(\underline{x}) = E[y|\underline{x}]$, it was found that after training to minimize $E[L(y, \hat{y})]$, the model minimizes the expected error in its approximation of $g(\underline{x})$ as measured by $L(g(\underline{x}), \hat{y})$.

4.9 References

[1] E. Baum and F. Wilczek, "Supervised Learning of Probability Distributions by Neural Networks," in *Neural Information Processing Systems*, D. Anderson ed., pp.52–61, American Institute of Physics, 1988.

[2] A. El-Jaroudi and J. Makhoul, "A New Error Criterion For Posterior Probability Estimation With Neural Nets," in *Proc. 1990 Int. Joint Conf. Neural Networks*, San Diego, pp.III-185–192, IEEE, June 1990.

[3] S. Solla, E. Levin, and M. Fleisher, "Accelerated Learning in Layered Neural Networks," *Complex Systems*, **2(6)**, pp.625–640, 1988.

[4] H. Gish, "A Probabilistic Approach to the Understanding and Training of Neural Network Classifiers," in *Proceeding of the 1990 IEEE Conference on Acoustics, Speech and Signal Processing*, Albuquerque, pp.1361–1364, IEEE, 1990.

[5] J. Hampshire, B. Pearlmutter, "Equivalence Proofs for Multi-layer Perceptron Classifiers and the Bayesian Discriminant Function," in *Proceedings of the 1990 Connectionist Models Summer School,* eds. D. Touretzky et al., pp.159–172, San Mateo CA: Morgan Kaufmann, 1990.

[6] H. White, "Learning in Artificial Neural Networks: a Statistical Perspective," *Neural Computation*, **1(4)**, pp.425–464, 1990.

Appendix 4.A
Conditions for Minimization to E(y|x)

Proof: (4–8) $\Leftrightarrow$ $\big((4\text{–}9)\cdot(4\text{–}10)\big)$

where (4–8) (4–9) and (4–10) are defined as:

$$\forall\, p(y) \text{ s.t. } \int_0^1 p(y)\,dy = 1,\ \ \overline{y} = \int_0^1 p(y)\cdot y\,dy,\ \ \overline{y}\in(0,1)$$
$$\min_{0<\hat{y}<1} \int_0^1 p(y)\cdot L(y,\hat{y})\,dy \;=\; \int_0^1 p(y)\cdot L(y,\overline{y})\,dy \tag{4–8}$$

$$L(y,\hat{y}) = \int h'(\hat{y})\cdot\frac{\hat{y}-y}{\hat{y}}\,d\hat{y} + C(y) \tag{4–9}$$

$$h'(\hat{y}) > 0 \quad \text{for } 0<\hat{y}<1 \tag{4–10}$$

and where we assume $h'(\hat{y}) = \frac{d}{d\hat{y}}h(\hat{y})$ exists for $\hat{y}\in(0,1)$.

First it will be shown that $\big((4\text{–}9)\cdot(4\text{–}10)\big) \Rightarrow (4\text{–}8)$.

Consistent with the notation used in (4–8), we will use an overbar to indicate the expected value of a function taken with respect to $p(y)$:

$$\overline{L}(\hat{y}) = \int_0^1 p(y)L(y,\hat{y})\,dy.$$

Substitute in (4–9),

$$\overline{L}(\hat{y}) = \int_0^1 p(y)\Big[\int h'(\hat{y})\frac{\hat{y}-y}{\hat{y}}\,d\hat{y} + C(y)\Big]\,dy.$$

Take the derivative with respect to $\hat{y}$:

$$\frac{\partial}{\partial\hat{y}}\overline{L}(\hat{y}) = \int_0^1 p(y)h'(\hat{y})\Big(\frac{\hat{y}-y}{\hat{y}}\Big)\,dy. \tag{4–11}$$

Here we took the derivative under the integral. This equality holds if $h'(\hat{y})$ exists, as we have assumed for $\hat{y}\in(0,1)$.

A local minimum can occur for $x\in(0,1)$ if and only if:

$$\frac{\partial}{\partial\hat{y}}\overline{L}(\hat{y}) = 0 \quad\text{and}\quad \frac{\partial^2}{\partial\hat{y}^2}\overline{L}(\hat{y}) > 0\ .$$

From (4–11):

$$\begin{aligned}\frac{\partial}{\partial\hat{y}}\overline{L}(\hat{y}) &= h'(\hat{y})\int_0^1 p(y)\,dy - \frac{h'(\hat{y})}{\hat{y}}\int_0^1 y\,p(y)\,dy,\\ &= h'(\hat{y})\Big(1-\frac{\overline{y}}{\hat{y}}\Big).\end{aligned} \tag{4–12}$$

Given (4–10) it is clear that $\frac{\partial}{\partial\hat{y}}\overline{L}(\hat{y}) = 0$ for $\hat{y}\in(0,1)$ if and only if $\hat{y}=\overline{y}$. Taking derivatives again shows that this extremum represents a minimum:

$$\begin{aligned}\frac{\partial^2}{\partial\hat{y}^2}\overline{L}(\hat{y}) &= h''(\hat{y}) - h''(\hat{y})\,\frac{\overline{y}}{\hat{y}} + \frac{\overline{y}}{\hat{y}^2} + h'(\hat{y})\,\frac{\overline{y}}{\hat{y}^2},\\ \frac{\partial^2}{\partial\hat{y}^2}\overline{L}(\overline{y}) &= \frac{h'(\overline{y})}{\overline{y}} > 0 \quad .\end{aligned}$$

Thus it has been shown $\big((4\text{–}9)\cdot(4\text{–}10)\big)\Rightarrow(4\text{–}8)$.

It remains to be proven that $(4\text{–}8)\Rightarrow\big((4\text{–}9)\cdot(4\text{–}10)\big)$. Condition (4–8) certainly requires:

$$\int_0^1 p(y)\frac{\partial}{\partial\hat{y}}L(y,\hat{y})\,dy = 0 \;\text{ at }\; \hat{y}=\overline{y}. \tag{4–13}$$

Without loss of generality, as a change of notation, let

$$\frac{\partial}{\partial\hat{y}}L(y,\hat{y}) = G(y,\hat{y})\cdot(\hat{y}-y). \tag{4–14}$$

Substituting this into (4–13):

$$\int_0^1 p(y)G(y,\hat{y})(\hat{y}-y)\,dy = 0 \;\text{ at }\; \hat{y}=\overline{y}. \tag{4–15}$$

Now consider the distribution

$$p(y) = \rho\cdot\delta(y-y_1) + (1-\rho)\delta(y-y_2) \qquad (0\le y_1 < y_2 \le 1). \tag{4–16}$$

The δ function is the unit impulse function. Condition (4–8) requires that the minima be at $\overline{y}$ for all $p(y)$, therefore it must be true for the binary distribution given in (4–16). Notice $\overline{y} = \rho y_1 + (1-\rho)y_2$. Evaluating (4–15) with this distribution gives:

$$\rho G(y_1,\hat{y})(\overline{y}-y_1) + (1-\rho)G(y_2,\hat{y})(\overline{y}-y_2) = 0 \;\text{ at }\; \hat{y}=\overline{y},$$

$$\rho G(y_1,\overline{y})[\rho y_1 + (1-\rho)y_2 - y_1] + (1-\rho)G(y_2,\overline{y})[\rho y_1 + (1-\rho)y_2 - y_2] = 0,$$

$$\rho(1-\rho)(y_2-y_1)(G(y_1,\overline{y}) - G(y_2,\overline{y})) = 0 \quad .$$

Therefore $G(y_1, \overline{y}) = G(y_2, \overline{y})$, where ρ can be chosen to set $\overline{y}$ arbitrarily in (y_1, y_2). Thus $G(y_1, \hat{y}) = G(y_2, \hat{y}))$ for any $y_1 < \hat{y} < y_2$. Therefore $G(y, \hat{y})$ is independent of y, so we may write:

$$\frac{\partial}{\partial \hat{y}} L(y, \hat{y}) = G(\hat{y}) \cdot (\hat{y} - y).$$

Evaluating at $y = 0$ and using the definition of $h(\hat{y})$ shows:

$$G(\hat{y}) = \frac{h'(\hat{y})}{\hat{y}}.$$

Substituting this $G(\hat{y})$ back into (4–14) and then integrating both sides of the equation gives the desired result (4–9). The result (4–8) $\Rightarrow$ (4–10) follows easily from the requirement that the unique extremum found by (4–9), be a minimum rather than a maximum. Since (4–8) $\Rightarrow$ $\big((4\text{–}9) \cdot (4\text{–}10)\big)$ and $\big((4\text{–}9) \cdot (4\text{–}10)\big) \Rightarrow$ (4–8), the equivalence has been shown. *Q.E.D.*

Appendix 4.B
Restrictions on $h(\hat{y})$

Proof: $\big((4\text{–}8) \cdot (4\text{–}17)\big) \Rightarrow (4\text{–}18)$

Where (4–8) is given in Appendix 4.A, and (4–17) and (4–18) are:

$$L(0, \hat{y}) = L(1, 1 - \hat{y}). \tag{4–17}$$

$$\frac{1 - \hat{y}}{\hat{y}} = \frac{h'(1 - \hat{y})}{h'(\hat{y})}. \tag{4–18}$$

Given (4–8), as in Appendix 4.A, the equilibrium equation (4–13) must hold for distribution (4–16) with $y_1 = 0$ and $y_2 = 1$,

$$\rho \frac{\partial}{\partial \hat{y}} L(1, \hat{y}) + (1 - \rho) \frac{\partial}{\partial \hat{y}} L(0, \hat{y}) = 0 \text{ at } \hat{y} = \rho.$$

Using (4–17):

$$-\rho \frac{\partial}{\partial \hat{y}} L(0, 1 - \hat{y}) + (1 - \rho) \frac{\partial}{\partial \hat{y}} L(0, \hat{y}) = 0 \text{ at } \hat{y} = \rho,$$

$$\frac{1 - \rho}{\rho} = \frac{h'(1 - \rho)}{h'(\rho)}.$$

Since ρ is arbitrary $\in (0, 1)$, equation (4–18) is proven. *Q.E.D.*

Appendix 4.C
Minimization to a Probability with Insufficiently Powerful Models

Proof: (4–8) ⇒ (4–19)

The condition for p-admissibility, (4–8), is defined above and (4–19) is defined:

$$\min_{\underline{\theta}} E[L(y,\hat{y})] \text{ is equivalent to } \min_{\underline{\theta}} E[L(g(\underline{x})),\hat{y}], \tag{4–19}$$

where $g(\underline{x}) = \int p(y|\underline{x}) \cdot y\, dy$ and $\hat{y} = f(\underline{x},\underline{\theta})$. Here the expected value symbol E refers to the expected value taken over a probability distribution on $\underline{x}, y$. So for any function $F(\underline{x}, y)$ the expected value $E[F(\underline{x}, y)]$ is shorthand for the equivalent integral forms:

$$E[F(\underline{x},y)] = \int_{\underline{x}}\int_{y} p(\underline{x},y)F(\underline{x},y)\, dy\, d\underline{x}, \tag{4–20}$$

$$= \int_{\underline{x}} p(\underline{x})[\int_{y} p(y|\underline{x})F(\underline{x},y)\, dy]\, d\underline{x}. \tag{4–21}$$

We assume these integrals actually exist and that the distribution functions $p(\underline{x}, y)$, $p(\underline{x})$, and $p(y|\underline{x})$ all exist where they are evaluated within the integral. This assumption is certainly true if $p(\underline{x}, y)$ is the probability of choosing a training pair $(\underline{x}, y)$ from a random sample from a finite or a countably infinite training set. As shown in Appendix 4.A, (4–8) ⇒ (4–9).

Let

$$k(\hat{y}) = \int \frac{h'(\hat{y})}{\hat{y}}\, d\hat{y}.$$

Then (4–9) may be written

$$L(y,\hat{y}) = h(\hat{y}) + C(y) - y \cdot k(\hat{y}). \tag{4–22}$$

Thus,

$$\begin{aligned} E[L(y,\hat{y})] &= E[h(\hat{y}) + C(y) - y \cdot k(\hat{y})], \\ &= E[h(\hat{y})] + E[C(y)] - E[y \cdot k(\hat{y})]. \end{aligned}$$

Using equation (4–21) with $F = y \cdot k(\hat{y})$:

$$E[L(y,\hat{y})] = E[h(\hat{y})] + E[C(y)] - \int_{\underline{x}} p(\underline{x})[\int_{y} p(y|\underline{x})\; y \cdot k(\hat{y})\, dy]\, d\underline{x}.$$

Note that given an input $\underline{x}$, the output $\hat{y}$ is constant.

Thus, $k(\hat{y})$ may be factored out of the integral over y:

$$E[L(y,\hat{y})] = E[h(\hat{y})] + E[C(y)] - \int_{\underline{x}} p(\underline{x}) \cdot k(\hat{y}) [\int_{y} p(y|\underline{x})\ y\, dy]\, d\underline{x}.$$

Using the definition $g(\underline{x}) = \int p(y|\underline{x}) \cdot y\, dy$,

$$E[L(y,\hat{y})] = E[h(\hat{y})] + E[C(y)] - \int_{\underline{x}} p(\underline{x}) \cdot k(\hat{y}) \cdot g(\underline{x})\, d\underline{x}.$$

Substitute the definition $p(\underline{x}) = \int_y p(\underline{x}, y)\, dy$,

$$E[L(y,\hat{y})] = E[h(\hat{y})] + E[C(y)] - \int_{\underline{x}} [\int_{y} p(\underline{x}, y)\, dy] \cdot k(\hat{y}) \cdot g(\underline{x})\, d\underline{x}.$$

Now use (4–20)

$$\begin{aligned} E[L(y,\hat{y})] &= E[h(\hat{y})] + E[C(y)] - E[g(\underline{x}) \cdot k(\hat{y})], \\ &= E[h(\hat{y}) + g(\underline{x}) - g(\underline{x}) \cdot k(\hat{y})] + E[C(y) - g(\underline{x})]. \end{aligned}$$

Use (4–22) once more

$$E[L(y,\hat{y})] = E[L(g(\underline{x}),\hat{y})] + E[C(y) - g(\underline{x})]. \qquad (4\text{–}23)$$

Since $E[C(y) - g(\underline{x})]$ is a constant with respect to $\underline{\theta}$, it follows from (4–23):

$$\min_{\underline{\theta}} E[L(y,\hat{y})] \text{ is equivalent to } \min_{\underline{\theta}} E[L(g(\underline{x}),\hat{y})].$$

Q.E.D.

Chapter 5
Conclusion

Chapter 1 introduces the problem of probabilistic estimation from a discrete-valued database. The tradeoff between bias and variance is discussed and the Principle of Maximum Entropy is used to introduce the Gibbs distribution.

The work presented in Chapters 2–3 concerns probability estimation without iterative error minimization. The requirement for closed-form equations for the parameters to the probabilistic model guides the development of a novel Gibbs modeling algorithm. This Gibbs model is a representation based on a large number of parameters easily calculated from probabilities of events found in a database. The Gibbs representation is analyzed and tested. One way to use the Gibbs modeling system is to build a classifier based upon the probability estimates. Test results show excellent performance for this classifier on a variety of problems.

The classifier presented is described in terms of the Gibbs modeling algorithm. This emphasizes the probabilistic basis. However, the model can be expressed and implemented as a feed forward neural network or as a rule based model. The results of Chapters 1–3 support the implementation of these techniques using the Gibbs modeling algorithm. More generally, the results further support the use of the Gibbs distribution and probabilistic methods in database modeling systems.

The problem addressed in Chapter 4 concerns iterative minimization of error as a method of producing a probabilistic model. The simple and general form of the basic iterative scheme for creating a probabilistic model is shown to force a particular form on the objective function used in minimization. This form is defined and analyzed. The work presented in Chapter 4 supports the use of two well known objective functions while rejecting many other objective functions as unsuitable for probability estimation systems.