CS281A/Stat241A: Statistical Learning Theory

Sum-Product, Max A Posteriori, Bayesians and Frequentists (9/16/04)

Lecturer: Michael I. Jordan *Scribes: Rosen Diankov and Xiaoyan Liu*

1 Sum-Product Algorithm

1.1 Introduction

The sum-product algorithm only works on trees, but can compute the marginal probabilities of all nodes with the same time complexity as computing the marginal probability of just one node. The reason is that the structure of the tree allows for a node-to-node message passing protocol that reuses messages between computing marginal probabilities. The algorithm works for both undirected and directed graphical models. The directed graphical models can be used only if their *moralized graph* is a tree or a polytree. If directed graphical models are restricted to trees or trees, their family of probability distributions can be represented by undirected potential functions. Given the conditional probabilities $p(x_{root})$ and $p(x_j|x_i)$ of a directed graph, the potentials of the corresponding undirected graph can be parameterized as:

$$\psi(x_{root}) = p(x_{root}) \tag{1}$$

$$\psi(x_i) = 1 \qquad \text{for i} \neq \text{root} \tag{2}$$

$$\psi(x_i, x_j) = p(x_j|x_i) \qquad \text{for i is a parent of j.} \tag{3}$$

We will only work with undirected models and denote the graph as $\mathcal{G} = (\mathcal{V}, \mathcal{E})$ where $\mathcal{V}$ are the vertices and $\mathcal{E}$ are the edges.

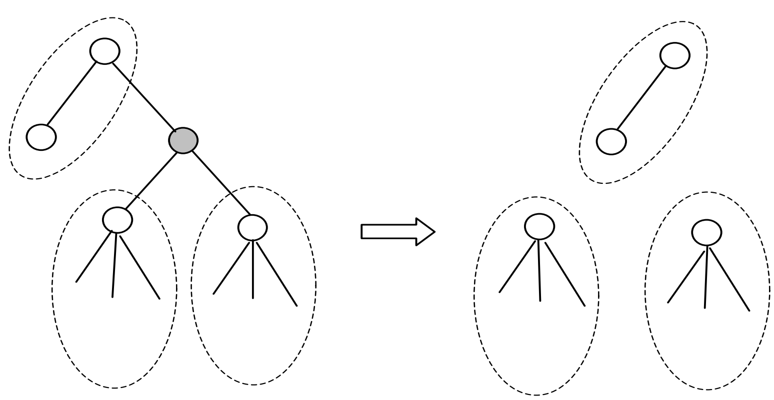

Figure 1: Undirected tree, conditioning on set of nodes X_E creates conditional independencies through graph separation.

1.2 Conditioning

Conditioning on a set E in an undirected tree will subdivide the graph into smaller trees (based on graph separation) whose marginal probabilities can solved independently of each other (Fig. 1). In order to work with only one tree, define new conditional potentials ψ_i^E such that:

$$\psi_i^E(x_i) \equiv \begin{cases} \psi_i(x_i)\delta(x_i, \bar{x}_i), & i \in E \\ \psi_i(x_i), & i \notin E \end{cases} \tag{4}$$

where $\bar{x}_i$ are the current values of E we are conditioning on. The new distribution then becomes:

$$p(x|\bar{x}_E) = \frac{1}{Z^E} \prod_{i \in \mathcal{V}} \psi^E x_i \prod_{(i,j) \in \mathcal{E}} \psi(x_i, x_j), \tag{5}$$

where Z^E is the new normalizing factor.

1.3 Message Passing

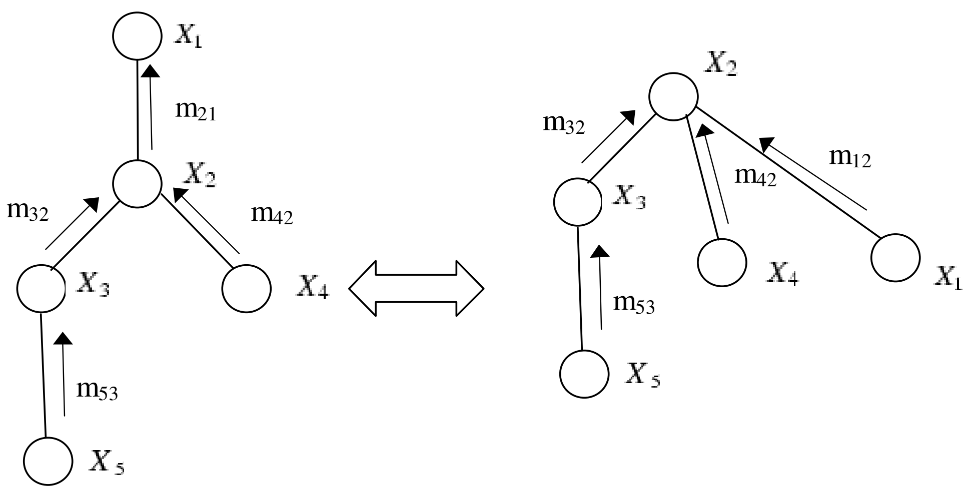

Figure 2: Equivalence of message passing protocol after re-rooting tree.

The SUM-PRODUCT algorithm computes marginal probabilities in a similar way as in the elimination algorithm: first choose an ordering I of the nodes such that children are computed before their parents; then sum up each node in that order by introducing intermediate terms to store the partial results. Take for example the graph in Fig. 2 and sum with an ordering $(5, 4, 3, 2, 1)$, then the intermediate terms will be:

$$m_{53}(x_3) = \sum_{x_5} \psi^E(x_5)\psi(x_3, x_5) \tag{6}$$

$$m_{42}(x_2) = \sum_{x_4} \psi^E(x_4)\psi(x_2, x_4) \tag{7}$$

$$m_{32}(x_2) = \sum_{x_3} \psi^E(x_3)\psi(x_2, x_3)m_{53}(x_3) \tag{8}$$

$$\tag{9}$$

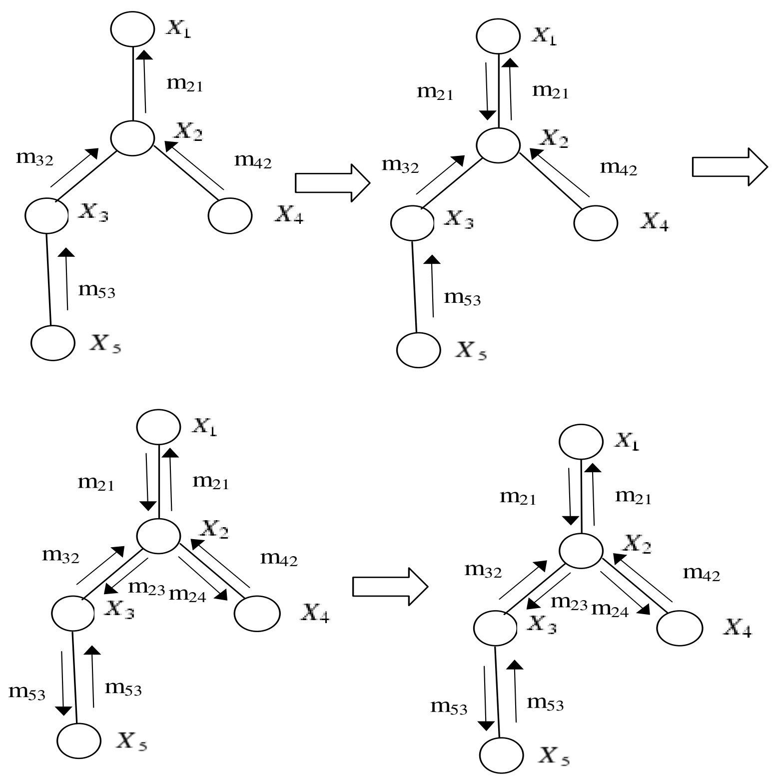

Figure 3: Message passing protocol for a graph.

In general, any message m_{ji} is computed as:

$$m_{ji}(x_i) = \sum_{x_j} \left(\psi^E(x_j)\psi(x_i, x_j) \prod_{k \in \mathcal{N}(j) \setminus i} m_{kj}(x_j) \right), \tag{10}$$

and the final marginal probability of the root note is:

$$p(x_{root}|\bar{x}_E) \propto \psi^E(x_{root}) \prod_{k \in \mathcal{N}(root)} m_{kroot}(x_{root}). \tag{11}$$

Since the graph is a tree, we could have also set the root to be x_2 (Fig. 2) and computed all the messages from there. Doing so would require a subset of the previous messages plus a new message $m_{12}(x_2)$. The heart of the SUM-PRODUCT algorithm is to reuse the messages and compute the marginal probability of every node. The insight, based on the *message passing protocol*, is that a given node x_i can send a message to one of its neighbors only when the messages from all the other neighbors have been computed. This can be divided into two steps: compute all the messages needed to get the marginal probability of one node; then starting from that node and recursing down the tree, compute all the backward messages. This is shown in Fig. 3.

2 Maximum *a posteriori* Configuration

The problem is to find the maximal probability that can be achieved by some set of random variables given a set of observations. Using Fig. 2 as an example, the equation becomes:

$$\max_x p(x) = \max_{x_1} \max_{x_2} \max_{x_3} \max_{x_4} \max_{x_5} p(x_1)p(x_2|x_1)p(x_3|x_2)p(x_4|x_2)p(x_5|x_3) \tag{12}$$

$$= \max_{x_1} p(x_1) \max_{x_2} p(x_2|x_1) \max_{x_3} p(x_3|x_2) \max_{x_4} p(x_4|x_2) \max_{x_5} p(x_5|x_3). \tag{13}$$

Let the new messages $m_{ji}^{\max}$ be defined similarly as for the marginalization case, except that all sum operators are replaced by max operators:

$$m_{53}^{\max}(x_3) = \max_{x_5} \psi^E(x_5)\psi(x_3, x_5) \tag{14}$$

$$m_{42}^{\max}(x_2) = \max_{x_4} \psi^E(x_4)\psi(x_2, x_4) \tag{15}$$

$$m_{32}^{\max}(x_2) = \max_{x_3} \psi^E(x_3)\psi(x_2, x_2)m_{53}(x_3) \tag{16}$$

$$\tag{17}$$

Then the maximum probability is achieved by:

$$\max_x p(x) = \max_{x_1} \psi^E(x_1)m_{21}(x_1) \tag{18}$$

Doing similar transforms for Equations (11) and (10) yields the MAX-PRODUCT algorithm. Note that there is an underflow problem when working straight with the probabilities. It is often safer to work with the log of the probabilities. Since log is a monotonic function, both $p(x)$ and $\log p(x)$ share the same global maximums in the same configuration x^*. Then equation (10) will become:

$$\log m_{ji}^{\max}(x_i) = \max_{x_j} \left(\log \psi^E(x_j) + \log \psi(x_i, x_j) + \sum_{k \in \mathcal{N}(j) \backslash i} \log m_{kj}^{\max}(x_j) \right), \tag{19}$$

2.1 Maximum Configuration

In order to find a maximum configuration of the unobserved nodes, will have to introduce new terms that remember the variables that ware taken at each step of the *max-product* algorithm. For example, when computing $m_{32}^{\max}(x_2)$ will also want to compute some function $\delta_{32}(x_2)$ such that for a given x_2 it returns the configuration of x_3 that was used to compute the maximum. Similarly, once x_3 is chosen, it will call $\delta_{53}(x_3)$ to compute the maximum configuration of x_5. Given two nodes i and j such that i is the parent of j, $\delta_{ji}(x_i)$ is computed as:

$$\delta_{ji}(x_i) \in arg \max_{x_j} \psi^E(x_j)\psi(x_i, x_j) \prod_{k \in \mathcal{N}(j) \backslash i} m_{kj}^{\max}(x_j), \tag{20}$$

which just becomes a table in the dimension of x_i in the discrete case. The root node then becomes:

$$x_{root}^* \in arg \max_{x_{root}} \psi^E(x_{root}) \prod_{k \in \mathcal{N}(j) \backslash i} m_{k,root}^{\max}(x_j), \tag{21}$$

Once x_{root} is computed, for every neighbor k of the root node, $x_k^* = \delta_{k,root}(x_f^*)$. Starting this propagation at the root node and continuing until all nodes have been hit will yield the maximum configuration.

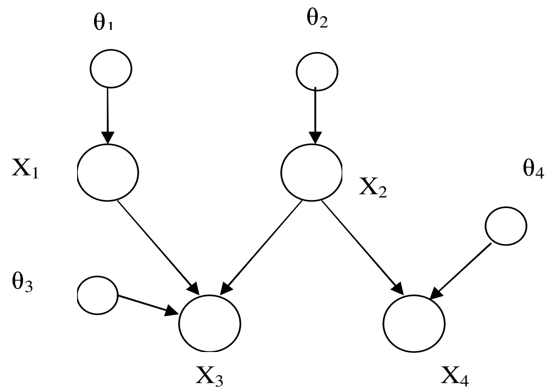

Figure 4: Example of Bayesian vision of parameters as random variables.

3 Bayesians vs Frequentists

Bayesian statistics is at some level an attempt to deny any fundamental distinction between probability theory and statistics. Probability theory provides the capability for inverting relationship between uncertain quantities, which is the essence of Bayes rule, and Bayesian statistics represents an attempt to treat all statistical inference as probabilistic inference. For example, frequentists view $p(x|\theta)$ as a conditional probability distribution, i.e. the assignment of the probability mass to the unknown random variable X, given a fixed parameter θ. Bayesian statisticians, however, view X as known values, i.e. they have observed the realization x, and view θ as unknown. They would like to be able to invert the relationship between x and θ. The Bayesian method will implement the notion of inversion using Bayes rule:

$$p(\theta|x) = \frac{p(x|\theta)p(\theta)}{p(x)}$$

where $p(\theta)$ is the prior probability, $p(\theta|x)$ is the posterior probability and $p(x|\theta)$ is the likelihood function. E.g. $p(x|\theta) = \prod_{i=1}^{n} p(x_i|x_{\pi_i}, \theta_i)$. The numerator $p(x, \theta)$ is just the joint probability of X and θ.

However, the frequentist approach wishes to avoid the use of prior probability in statistics, and thus avoids the use of Bayes rule for the purpose of assigning probabilities to parameters. θ is considered fixed by frequentists while in contrast θ is considered a random variable by Bayesian statisticians. From the point view of frequentist statistics, there is no single preferred methodology for inverting the relationship between parameters and data. They want to consider various estimators of θ. One establishes various general criteria for evaluating the quality of various estimators, and choose the estimator that is best according to these criteria. One particular estimator that is widely used in frequentist statistics is the maximum likelihood estimator. Bayesians believe in considering all possible estimators of parameters to estimate the best one while frequentists believe in finding the best estimator of the parameter based on a loss function.

For example, given that the parameters in Fig 4 are independent, the joint probability distribution becomes:

$$p(x, \theta) = p(\theta_1)p(\theta_2)p(\theta_3)p(\theta_4)p(x_1|\theta_1)p(x_2|\theta_2)p(x_3|x_1, x_2, \theta_3)p(x_4|x_2, \theta_4)$$

Hence, computing the posterior probability $p(\theta|x)$ for Bayesians is just an inference problem in a computational sense, i.e. all x_i's are observed and all θ's are unobserved in the graph corresponding to the probabilistic inference problem (Fig 5).

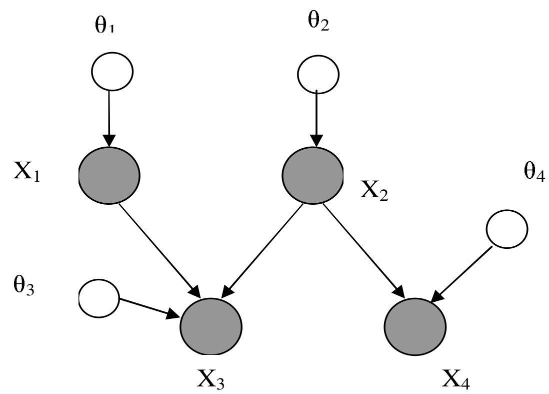

Figure 5: Parameter estimation as an inference problem: the Bayesian perspective.