Bayesian Learning in Probabilistic Decision Trees

Michael I. Jordan

MIT

Collaborators

Robert Jacobs (Rochester)

Lei Xu (Hong Kong)

Geoffrey Hinton (Toronto)

Steven Nowlan (Synaptics)

Marina Meila (MIT)

Lawrence Saul (MIT)

Outline

- decision trees
- probabilistic decision trees
- EM algorithm and extensions
- model selection, Bayesian computations
- empirical results
  - system identification
  - classification
- theoretical results
  - training set error
  - test set error

Some problems with multi-layered neural networks

- the learning algorithms are slow
- hard to understand the network
- hard to build in prior knowledge
- poor performance on non-stationary data
- not natural for some functions

Supervised learning (aka regression, classification)

We assume that the learner is provided with a *training set*:

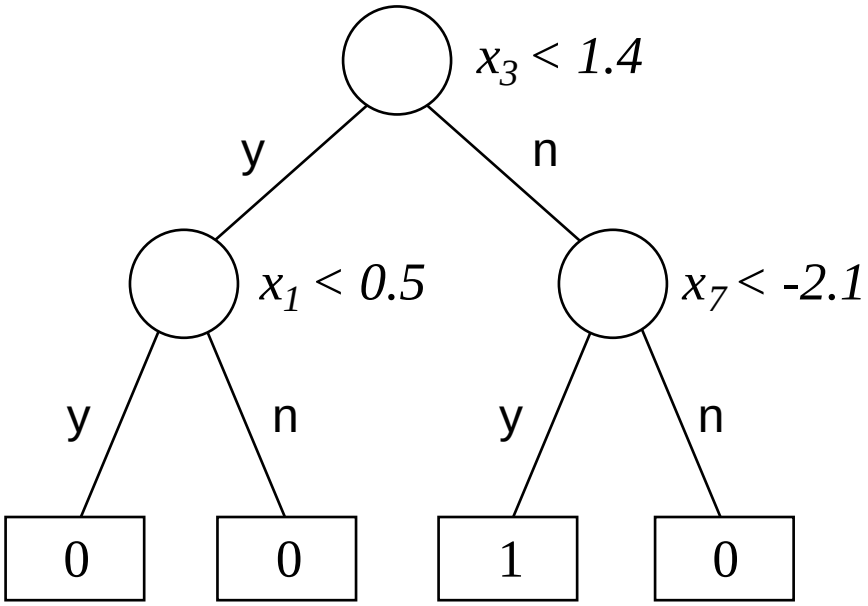
$$\mathcal{X} = \{(\mathbf{x}^{(t)}, \mathbf{y}^{(t)})\}_1^T$$

where $\mathbf{x}$ is an *input vector* and $\mathbf{y}$ is an *output vector*.

We will gauge performance on a *test set*:

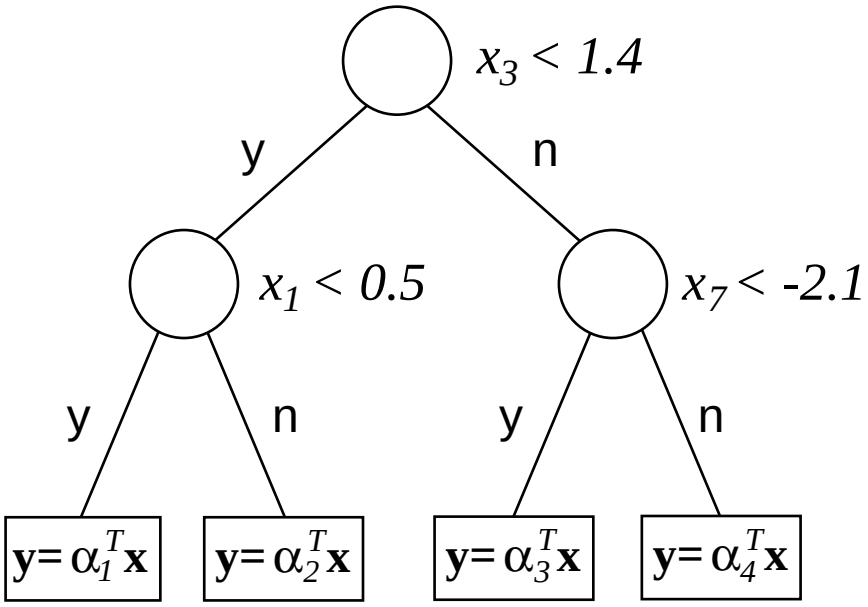
$$\mathcal{X}_s = \{(\mathbf{x}^{(t)}, \mathbf{y}^{(t)})\}_1^{T_s}$$

Decision trees



- drop the data set down the tree
- at each node, try to find a split of the input space (a half-plane) that yields the largest gain in "purity" on left and right
- build a large tree and prune backward to create a nested sequence of trees
- pick the best tree from the sequence using cross-validation

Regression trees



- splitting is based on RSS

Some advantages:

- often much faster than neural networks
- often more interpretable
- allow operating points to be utilized

Some disadvantages:

- non-smooth regression surface
- coordinate dependent
- batch methods

Probabilistic Decision Trees (Hierarchical mixtures of experts—HME)

(Jordan & Jacobs, 1994)

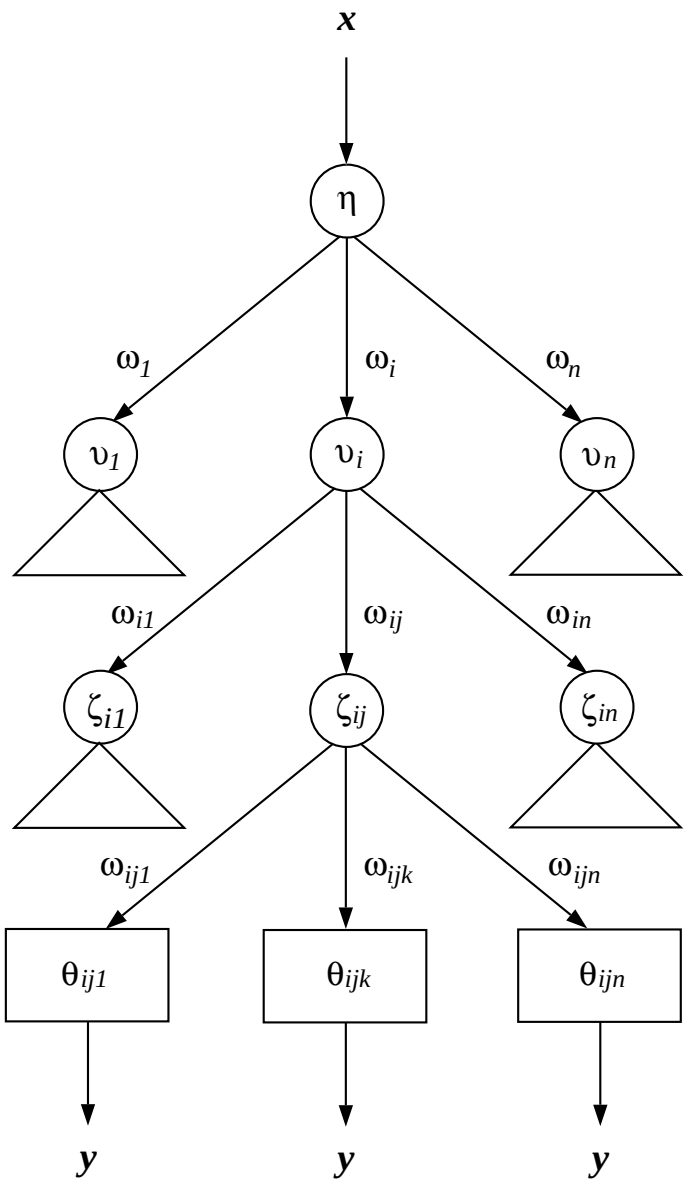
Why probabilities?

- smoother regression surface
- error bars from likelihood/Bayesian theory (e.g., SEM algorithm)
- convergence results from likelihood/Bayesian theory
- can handle categorical variables and missing data in principled ways
- better performance (e.g., leverage issue)

Probabilistic Decision Trees

- drop inputs down the tree and use probabilistic models for decisions
- at leaves of trees use probabilistic models to generate outputs from inputs
- use a Bayes' rule recursion to compute posterior credit for nonterminals in the tree

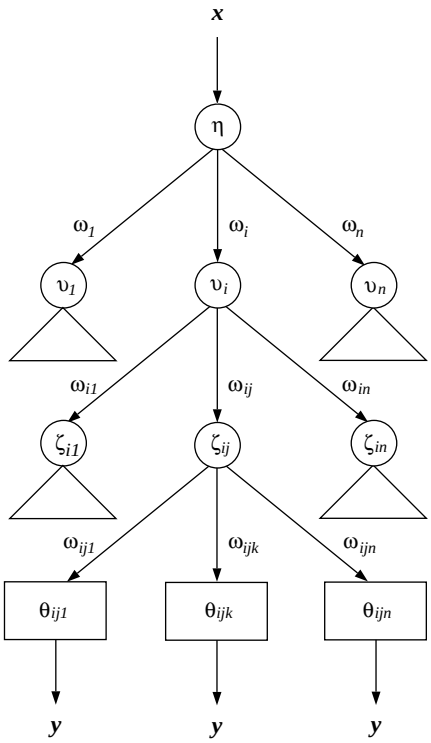
The basic idea is to convert the decision tree into a mixture model



Model the decisions in the decision tree using categorical probability models

- let ω_i, ω_{ij}, ω_{ijk}, ... represent multinomial decision variables at the nonterminals

- these variables will be treated as "missing" data (cf. states of an HMM)
- each path down the tree defines a component of a mixture



Decision models at the nonterminals:

$$P(\omega_i|\mathbf{x}, \eta)$$

$$P(\omega_{ij}|\mathbf{x}, \omega_i, \nu_i)$$

$$P(\omega_{ijk}|\mathbf{x}, \omega_i, \zeta_{ij})$$

Output models at the leaves:

$$P(\mathbf{y}|\mathbf{x}, \omega_i, \omega_{ij}, \omega_{ijk} \ldots, \theta_{ijk\ldots})$$

The total probability of an output $\mathbf{y}$ given an input $\mathbf{x}$ is given by the sum across all paths from the root to the leaves:

$$\begin{aligned} P(\mathbf{y}|\mathbf{x},\Theta) \;=\; & \sum_i P(\omega_i|\mathbf{x},\eta) \sum_j P(\omega_{ij}|\mathbf{x},\omega_i,\nu_i) \\ & \sum_k P(\omega_{ijk}|\mathbf{x},\omega_i,\zeta_{ij}) \cdots \\ & P(\mathbf{y}|\mathbf{x},\omega_i,\omega_{ij},\omega_{ijk}\ldots,\theta_{ijk\cdots}) \end{aligned}$$

This is a (conditional) mixture model.

Moments of this mixture distribution are readily computed by tree traversal processes.

Define

$$\begin{aligned}
\mu &\equiv E(\mathbf{y}|\mathbf{x}) \\
\mu_i &\equiv E(\mathbf{y}|\mathbf{x}, \omega_i) \\
\mu_{ij} &\equiv E(\mathbf{y}|\mathbf{x}, \omega_i, \omega_{ij}) \\
&\cdots \\
\mu_{ijk\cdots} &\equiv E(\mathbf{y}|\mathbf{x}, \omega_i, \omega_{ij}, \omega_{ijk}, \cdots)
\end{aligned}$$

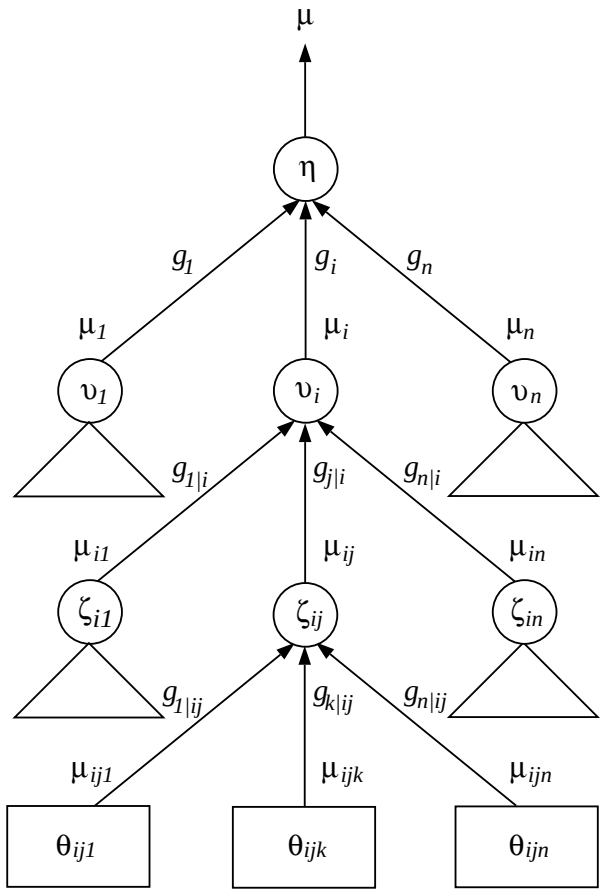
and define

$$\begin{aligned}
g_i &\equiv P(\omega_i|\mathbf{x}, \eta) \\
g_{j|i} &\equiv P(\omega_{ij}|\mathbf{x}, \omega_i, \nu_i) \\
g_{k|ij} &\equiv P(\omega_{ijk}|\mathbf{x}, \omega_i, \zeta_{ij}) \\
&\cdots
\end{aligned}$$

(omitting the parameters for simplicity)

Then,

$$\begin{aligned}
\mu &= \sum_i g_i \mu_i \\
\mu_i &= \sum_j g_{j|i} \mu_{ij} \\
\mu_{ij} &= \sum_k g_{k|ij} \mu_{ijk} \\
\mu_{ijk} &= f(\theta_{ijk}^T \mathbf{x})
\end{aligned}$$



Component Models

Decision models

- $P(\omega_i|\mathbf{x}, \eta)$ is a classification model
- any parametric classification model is appropriate—we use a multinomial logit model
- this yields "soft" linear discriminants—soft version of a CART/C4.5 tree

Leaf models

- we use simple generalized linear models
- Regression—linear regression
- Binary classification—logistic regression
- Multiway classification—multinomial logit model
- (can also handle count estimates, failure estimates, etc.)

Multinomial logit model

- *the deterministic component*:

$$g_i = \frac{e^{\xi_i}}{\Sigma_j \, e^{\xi_j}}$$

  where

$$\xi_i = \theta_i^T \mathbf{x}$$

  soft linear discriminants

  - the directions of the θ_i determine the orientations of the discriminant surfaces (i.e., splits)
  - the magnitudes of the θ_i determine the sharpness of the splits

- *the probabilistic component*:

$$P(\mathbf{y}|\mathbf{x}, \theta) = g_1^{y_1} g_2^{y_2} \cdots g_n^{y_n}$$

  where $y_i \in \{0, 1\}$ and $\Sigma_i \, y_i = 1$.

- *the log likelihood*:

$$l(\theta, \mathcal{X}) = \sum_p \sum_i y_i^{(p)} \log g_i^{(p)}$$

  which is the cross-entropy function.

- *the gradient*:

$$\frac{\partial l}{\partial \theta_i} = \sum_p \sum_i (y_i^{(p)} - g_i^{(p)}) \mathbf{x}^{(p)}$$

Computing the Hessian and substituting into the Newton-Raphson formula yields a simple, quadratically-convergent iterative algorithm known as *IRLS* (Iteratively-Reweighted Least Squares).

The Log Likelihood

$$E = \sum_p \log[\sum_i g_i^{(p)} \sum_j g_{j|i}^{(p)} \sum_k g_{k|ij}^{(p)} \cdots P_{ijk\ldots}(\mathbf{y}^{(p)}|\mathbf{x}^{(p)})]$$

- Problem: The log is outside of the sums. How can we optimize such a risk function efficiently?
- Solution: EM

The EM (Expectation-Maximization) Algorithm

(Baum, et al., 1971; Dempster, Laird, & Rubin, 1977)

Special cases:

- mixture likelihood clustering (soft K-means)
- many missing data algorithms
- Baum-Welch algorithm for HMM's

Applications to supervised learning (regression, classification)?

EM—Tutorial

- Suppose that the problem of maximizing a likelihood would be simplified if the values of some additional variables—called "missing variables"—were known
- These values are not known, but given the current values of the parameters, they can be estimated (the **E step**).
- Treat the estimated values as provisionally correct and maximize the likelihood in the usual way (the **M step**).
- We now have better parameter values, so the E step can be repeated. Iterate.

EM—Tutorial (cont.)

"missing" data: $\mathcal{Z}$

"complete" data: $\mathcal{Y} = \{\mathcal{X}, \mathcal{Z}\}$

"complete" likelihood: $l_c(\Theta, \mathcal{Y})$

The complete likelihood is a random variable, so average out the randomness:

E step:

$$Q(\Theta, \Theta^{(t)}) = E[l_c(\Theta, \mathcal{Y})|\mathcal{X}, \Theta^{(t)}],$$

This yields a fixed function Q, which can be optimized:

M step:

$$\Theta^{(t+1)} = \arg\max_{\Theta} Q(\Theta, \Theta^{(t)}).$$

Applying EM to the HME architecture

The missing data are the unknown values of the decisions in the decision tree.

Define indicator variables $z_i,\ z_{j|i},\ z_{k|ij}, \ldots$

Complete likelihood:

$$l_c(\Theta, \mathcal{Y}) = \sum_p \sum_i z_i^{(p)} \sum_j z_{j|i}^{(p)} \cdots \log[g_i^{(p)} g_{j|i}^{(p)} \cdots P_{ijk\ldots}(\mathbf{y}^{(p)}|\mathbf{x}^{(p)})]$$

Incomplete likelihood:

$$l(\Theta, \mathcal{X}) = \sum_p \log[\sum_i g_i^{(p)} \sum_j g_{j|i}^{(p)} \cdots P_{ijk\ldots}(\mathbf{y}^{(p)}|\mathbf{x}^{(p)})]$$

We need to compute the expected values of the missing indicator variables.

Note that, e.g.,

$$E(z_i^{(p)}|\mathbf{x}^{(p)}, \mathbf{y}^{(p)}) = P(\omega_i^{(p)}|\mathbf{x}^{(p)}, \mathbf{y}^{(p)})$$

Example

- one-level tree
- at each leaf, linear regression with Gaussian errors

For the i^{th} leaf and the t^{th} data point:

$$h_i^{(t)} = \frac{g_i^{(t)} e^{-\frac{1}{2}\|\mathbf{y}^{(t)} - \mu_i^{(t)}\|^2}}{\Sigma_j \, g_j^{(t)} e^{-\frac{1}{2}\|\mathbf{y}^{(t)} - \mu_j^{(t)}\|^2}}$$

where $\mu_i^{(t)} = \theta_i^T \, \mathbf{x}^{(t)}$.

This posterior is a normalized distance measure that reflects the relative magnitudes of the residuals $\mathbf{y}^{(t)} - \mu_i^{(t)}$.

Posterior probabilities

$$
\begin{aligned}
h_i &\equiv P(\omega_i|\mathbf{x},\mathbf{y}) \\
h_{j|i} &\equiv P(\omega_{ij}|\mathbf{x},\mathbf{y},\omega_i) \\
h_{k|ij} &\equiv P(\omega_{ijk}|\mathbf{x},\mathbf{y},\omega_i,\omega_{ij}) \\
&\ldots
\end{aligned}
$$

(cf. *prior* probabilities)

$$
\begin{aligned}
g_i &\equiv P(\omega_i|\mathbf{x}) \\
g_{j|i} &\equiv P(\omega_{ij}|\mathbf{x},\omega_i) \\
g_{k|ij} &\equiv P(\omega_{ijk}|\mathbf{x},\omega_i,\omega_{ij}) \\
&\ldots
\end{aligned}
$$

Bayes' rule yields:

$$h_i = \frac{g_i \Sigma_j g_{j|i} \Sigma_k g_{k|ij} P_{ijk}(\mathbf{y}|\mathbf{x})}{\Sigma_i g_i \Sigma_j \Sigma_j g_{j|i} \Sigma_k g_{k|ij} P_{ijk}(\mathbf{y}|\mathbf{x})}$$

$$h_{j|i} = \frac{g_{j|i} \Sigma_k g_{k|ij} P_{ijk}(\mathbf{y}|\mathbf{x})}{\Sigma_j g_{j|i} \Sigma_k g_{k|ij} P_{ijk}(\mathbf{y}|\mathbf{x})}$$

$$h_{k|ij} = \frac{g_{k|ij} P_{ijk}(\mathbf{y}|\mathbf{x})}{\Sigma_k g_{k|ij} P_{ijk}(\mathbf{y}|\mathbf{x})}$$

$$\ldots$$

Bayes' rule yields:

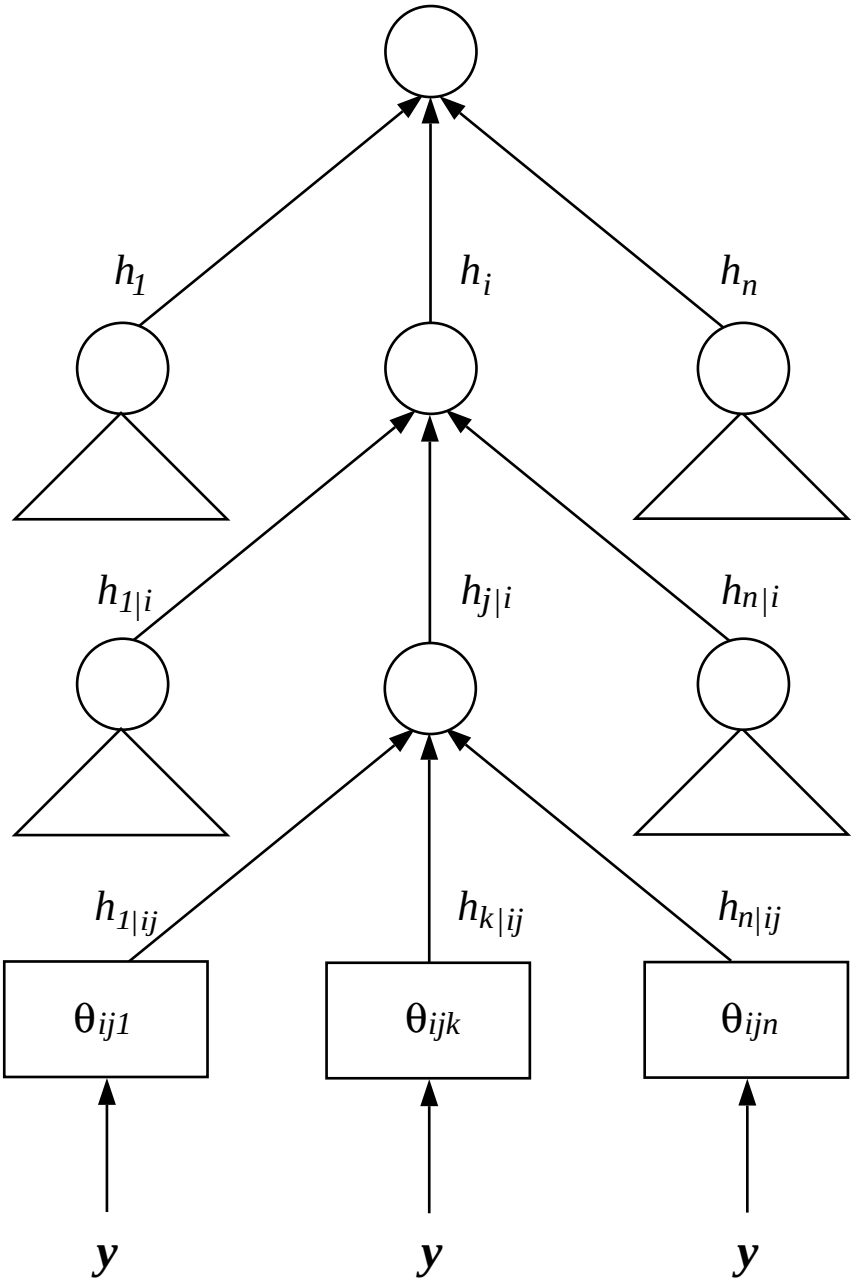
$$h_i = \frac{g_i \Sigma_j g_{j|i} \Sigma_k g_{k|ij} P_{ijk}(\mathbf{y}|\mathbf{x})}{\Sigma_i g_i \Sigma_j \Sigma_j g_{j|i} \Sigma_k g_{k|ij} P_{ijk}(\mathbf{y}|\mathbf{x})}$$

$$h_{j|i} = \frac{g_{j|i} \Sigma_k g_{k|ij} P_{ijk}(\mathbf{y}|\mathbf{x})}{\Sigma_j g_{j|i} \Sigma_k g_{k|ij} P_{ijk}(\mathbf{y}|\mathbf{x})}$$

$$h_{k|ij} = \frac{g_{k|ij} P_{ijk}(\mathbf{y}|\mathbf{x})}{\Sigma_k g_{k|ij} P_{ijk}(\mathbf{y}|\mathbf{x})}$$

$$\ldots$$

Posterior propagation



The E step

- compute the posterior probabilities ("up-down" algorithm)

The M step

- The Q function decouples into a set of separate maximum likelihood problems
- At the nonterminals, fit multinomial logit models, with the posteriors $h_i^{(t)}$, $h_{j|i}^{(t)}$, etc., serving as the targets
- At the leaves, obtain weighted likelihoods where the weights are the product of the posteriors from root to leaf

The M step (in more detail)

The maximization of $Q(\Theta, \Theta^{(t)})$ decouples into a set of weighted MLE problems:

$$\eta_i^{(t+1)} = \arg\max_{\eta_i} \sum_p \sum_i h_i^{(p)} \log g_i^{(p)},$$

(a cross-entropy cost)

$$\eta_{ij}^{(t+1)} = \arg\max_{\eta_{ij}} \sum_p \sum_i h_i^{(p)} \sum_j h_{j|i}^{(p)} \log g_{j|i}^{(p)},$$

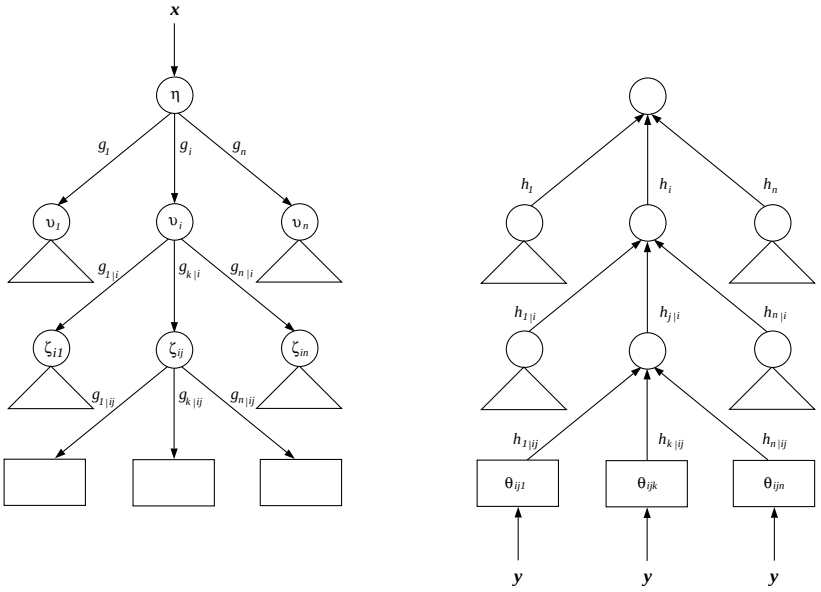
(a weighted cross-entropy cost)

$$\theta_{ij}^{(t+1)} = \arg\max_{\theta_{ij}} \sum_p \sum_i h_i^{(p)} \sum_j h_{j|i}^{(p)} \log P_{ijk\ldots}(\mathbf{y}^{(p)}|\mathbf{x}^{(p)})$$

(a general weighted log likelihood)

Each of these are weighted ML problems for generalized linear models (GLIM's). They can be solved efficiently using iteratively-reweighted least squares (IRLS).

HME Parameter Estimation



- drop the data set down the tree
- for each data point, compute the posterior probabilities for every branch of the tree
- at each nonterminal, use the posterior probabilities as (soft) classification targets
- at each leaf, fit a local model, where each data point is weighted by the product of the posterior probabilities from the root to that leaf

Model selection

How do we choose the structure of the tree?

- initialize with CART or C4.5 (cf. K-means)
  - can preserve local variable selection
- ridge regression
- cross-validation stopping within a fixed deep hierarchy (EM iterations "grow" the effective degrees of freedom)

Bayesian issues

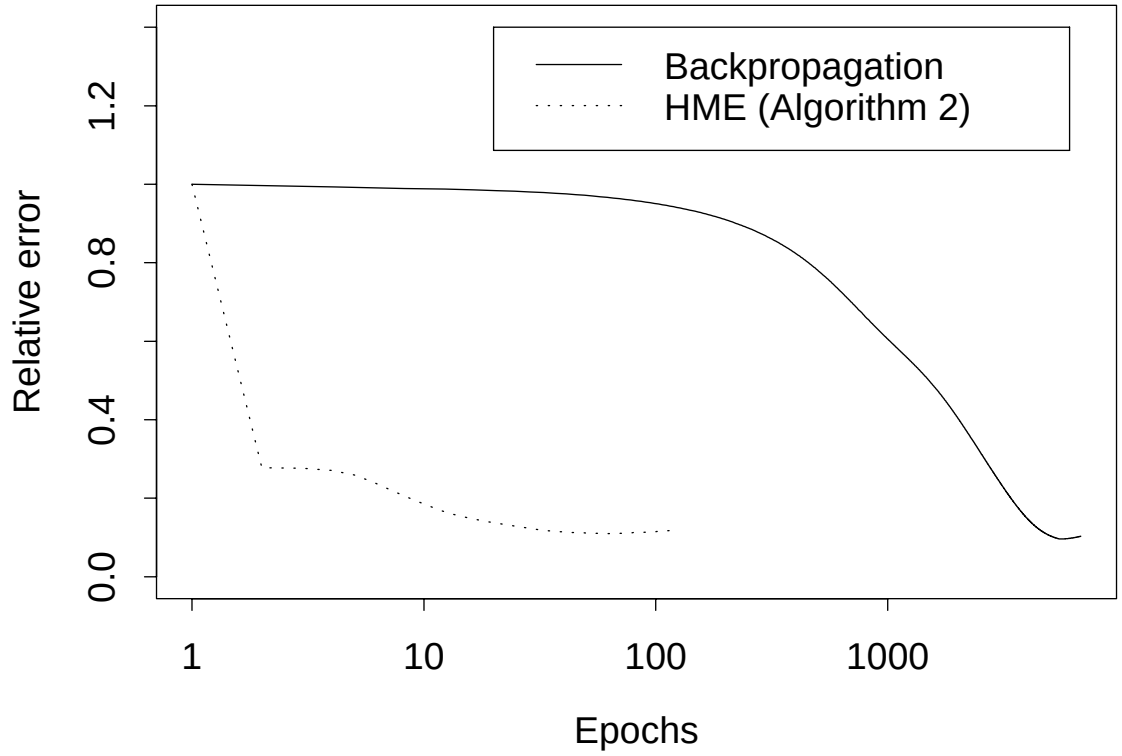
- Dirichlet priors
- Gibbs' sampling is straightforward
- Gaussian approximation of posterior via SEM calculation of Hessian
- Mean-field approximation of posterior

Regression: A System Identification Problem

- Forward dynamics of a four-joint, three-dimensional arm
- Twelve input variables, four output variables
- 15,000 points in the training set
- 5,000 points in the test set

- Four-level tree, with binary branches
- Compare to backpropagation in an MLP, with 60 hidden units
- Compare to CART, MARS

Batch algorithms



Summary—batch algorithms

Architecture	Relative Error	# Epochs
linear	.31	NA
backprop	.09	5,500
HME (Algorithm 1)	.10	35
HME (Algorithm 2)	.12	39
CART	.17	NA
CART (linear)	.13	NA
MARS	.16	NA

An On-Line Variant of HME

Use techniques from recursive estimation theory (Ljung & Söderström, 1986) to obtain the following on-line algorithm:

Expert networks:

$$U_{ij}^{(t+1)} = U_{ij}^{(t)} + h_i^{(t)} h_{j|i}^{(t)} (\mathbf{y}^{(t)} - \mu_{ij}^{(t)}) \mathbf{x}^{(t)T} R_{ij}^{(t)},$$

where R_{ij} is updated as follows:

$$R_{ij}^{(t)} = R_{ij}^{(t-1)} - \lambda^{-1} \frac{R_{ij}^{(t-1)} \mathbf{x}^{(t)} \mathbf{x}^{(t)T} R_{ij}^{(t-1)}}{\lambda [h_{ij}^{(t)}]^{-1} + \mathbf{x}^{(t)T} R_{ij}^{(t-1)} \mathbf{x}^{(t)}},$$

and λ is a decay parameter.

Top-level gating networks:

$$\mathbf{v}_i^{(t+1)} = \mathbf{v}_i^{(t)} + S_i^{(t)}(\ln h_i^{(t)} - \xi_i^{(t)})\mathbf{x}^{(t)},$$

$$S_i^{(t)} = S_i^{(t-1)} - \lambda^{-1}\frac{S_i^{(t-1)}\mathbf{x}^{(t)}\mathbf{x}^{(t)T}S_i^{(t-1)}}{\lambda + \mathbf{x}^{(t)T}S_i^{(t-1)}\mathbf{x}^{(t)}}.$$

Lower-level gating networks:

$$\mathbf{v}_{ij}^{(t+1)} = \mathbf{v}_{ij}^{(t)} + S_{ij}^{(t)}h_i^{(t)}(\ln h_{j|i}^{(t)} - \xi_{ij}^{(t)})\mathbf{x}^{(t)},$$

$$S_{ij}^{(t)} = S_{ij}^{(t-1)} - \lambda^{-1}\frac{S_{ij}^{(t-1)}\mathbf{x}^{(t)}\mathbf{x}^{(t)T}S_{ij}^{(t-1)}}{\lambda[h_i^{(t)}]^{-1} + \mathbf{x}^{(t)T}S_{ij}^{(t-1)}\mathbf{x}^{(t)}}.$$

Classification

Task	Baseline	CART	HME	Bayes
Heart	.44	.22	.18	.18
Pima	.35	.26	.22	.21
Orbitals	.48	.29	.23	.21

(*Error rates are computed using 10-fold cross-validation*)

Convergence results

(Jordan & Xu, 1994)

Theorem 1 *Assume that the training set $\mathcal{X}$ is generated by the mixture model ("realizable" case)*

Let us denote

$$P = \mathrm{diag}[P_g^{(k)}, P_1, \cdots, P_K, P_{\Sigma_1}, \cdots, P_{\Sigma_K}]$$

$$H(\Theta) = \frac{\partial^2 l(\Theta)}{\partial \Theta \partial \Theta^T}$$

where P_i are covariance matrices of the component models.

Then with probability one,

(1) Letting $-M, -m$ *(here* $M > m > 0$*) be the minimum and maximum eigenvalues of the negative definite matrix* $(P^{\frac{1}{2}})^T H(\Theta)(P^{\frac{1}{2}})$*, we have*

$$l(\Theta^*) - l(\Theta^{(k)}) \leq r^k [l(\Theta^*) - l(\Theta_0)], \qquad (1)$$

$$\|P^{-\frac{1}{2}}(\Theta^{(k)} - \Theta^*)\| \leq |r|^{k/2} \sqrt{\frac{2}{m}[l(\Theta^*) - l(\Theta_0)]}, \qquad (2)$$

where $r = 1 - (1 - \frac{M}{2})\frac{m^2}{M} < 1$*. We also have* $0 < |r| < 1$ *when* $M < 2$*.*

(2) For any initial point $\Theta_0 \in D_\Theta$*,* $\lim_{k\to\infty} \Theta^{(k)} = \Theta^*$ *when* $M < 2$*.*

Test Set Error

(Saul & Jordan, 1995)

Hard split model

$$y(\mathbf{x}) = \frac{1}{\sqrt{N}}(\mathbf{w}_1 \cdot \mathbf{x})\Phi(\mathbf{v} \cdot \mathbf{x}) + \frac{1}{\sqrt{N}}(\mathbf{w}_2 \cdot \mathbf{x})\Phi(-\mathbf{v} \cdot \mathbf{x})$$

Consider a structurally identically teacher with weight vectors $\mathbf{w}_1^*, \mathbf{w}_2^*, \mathbf{v}^*$.

Order parameters

$$\mathcal{R} = \begin{pmatrix} R_v & X_1 & X_2 \\ Y_1 & R_1 & C_1 \\ Y_2 & C_2 & R_2 \end{pmatrix} = \frac{1}{N} \begin{pmatrix} \mathbf{v}^* \cdot \mathbf{v} & \mathbf{w}_1^* \cdot \mathbf{v} & \mathbf{w}_2^* \cdot \mathbf{v} \\ \mathbf{v}^* \cdot \mathbf{w}_1 & \mathbf{w}_1^* \cdot \mathbf{w}_1 & \mathbf{w}_2^* \cdot \mathbf{w}_1 \\ \mathbf{v}^* \cdot \mathbf{w}_2 & \mathbf{w}_1^* \cdot \mathbf{w}_2 & \mathbf{w}_2^* \cdot \mathbf{w}_2 \end{pmatrix}.$$

Loss

$$\epsilon(\mathbf{v}, \mathbf{w}_1, \mathbf{w}_2; \mathbf{x}) = \frac{1}{2}\Big\{\left[y^*(\mathbf{x}) - \mathbf{w}_1 \cdot \mathbf{x}\right]^2 \Phi(\mathbf{v} \cdot \mathbf{x}) + \left[y^*(\mathbf{x}) - \mathbf{w}_2 \cdot \mathbf{x}\right]^2 \Phi(-\mathbf{v} \cdot \mathbf{x})\Big\}$$

Empirical risk (training energy)

$$E = \sum_{p=1}^{P} \epsilon(\mathbf{v}, \mathbf{w}_1, \mathbf{w}_2; \mathbf{x}^{(p)})$$

Test set error (under a Gibbs' distribution)

$$\epsilon_g(\mathcal{R}) = 1 - \left[1 - \frac{\cos^{-1}(R_v)}{\pi}\right]\left(\frac{R_1 + R_2}{2}\right) - \left[\frac{\cos^{-1}(R_v)}{\pi}\right]\left(\frac{C_1 + C_2}{2}\right) + \frac{(X_1 - X_2)(Y_1 - Y_2)}{2\pi\sqrt{1 - R_v^2}}$$

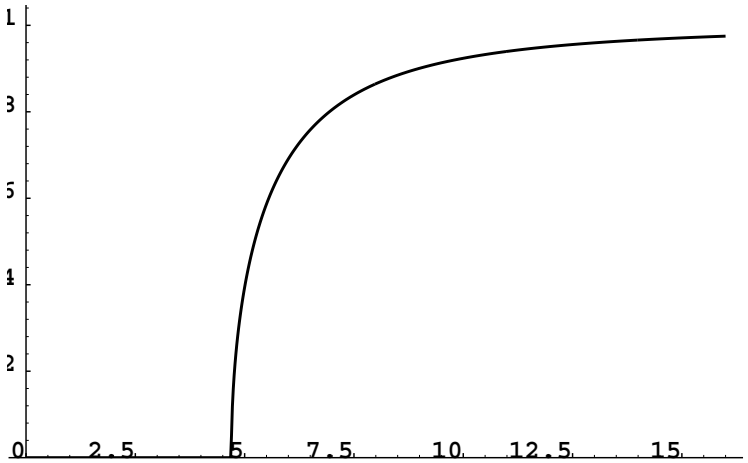
High temperature limit

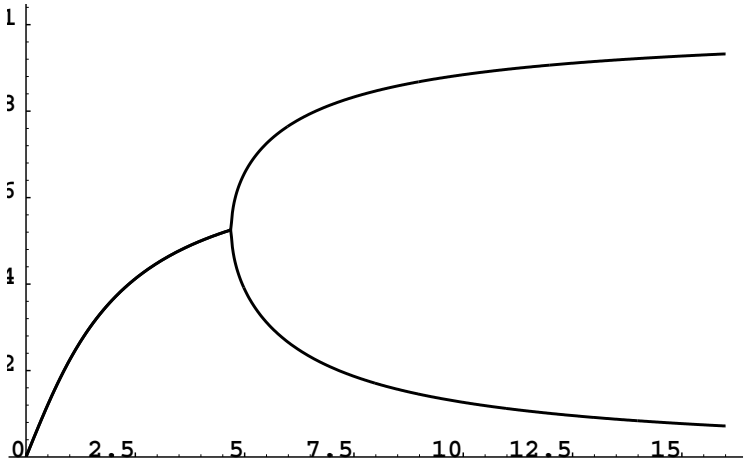
- $\beta \to 0$ (where $\beta = 1/T$ in the Gibbs' distribution)
- $\alpha \to \infty$ ($\alpha = P/N$)
- $\tilde{\alpha}$ remains finite ($\tilde{\alpha} = \alpha\beta$; a signal-to-noise ratio)

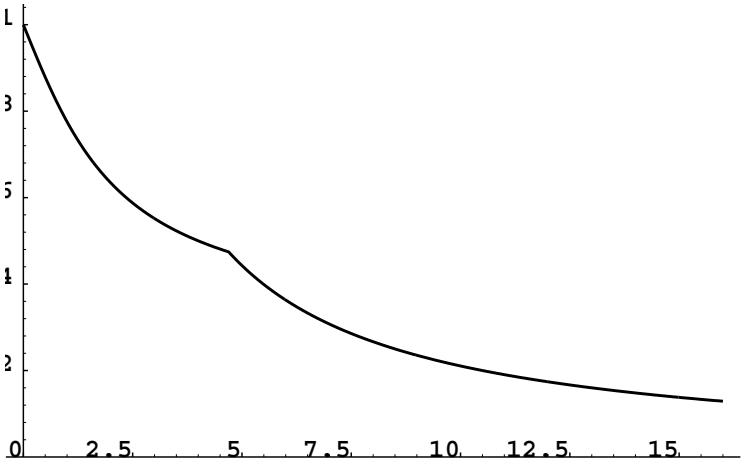
Results

- $\epsilon_g \sim \frac{2}{\tilde{\alpha}}$ (cf. perceptron)
- A continuous phase transition at

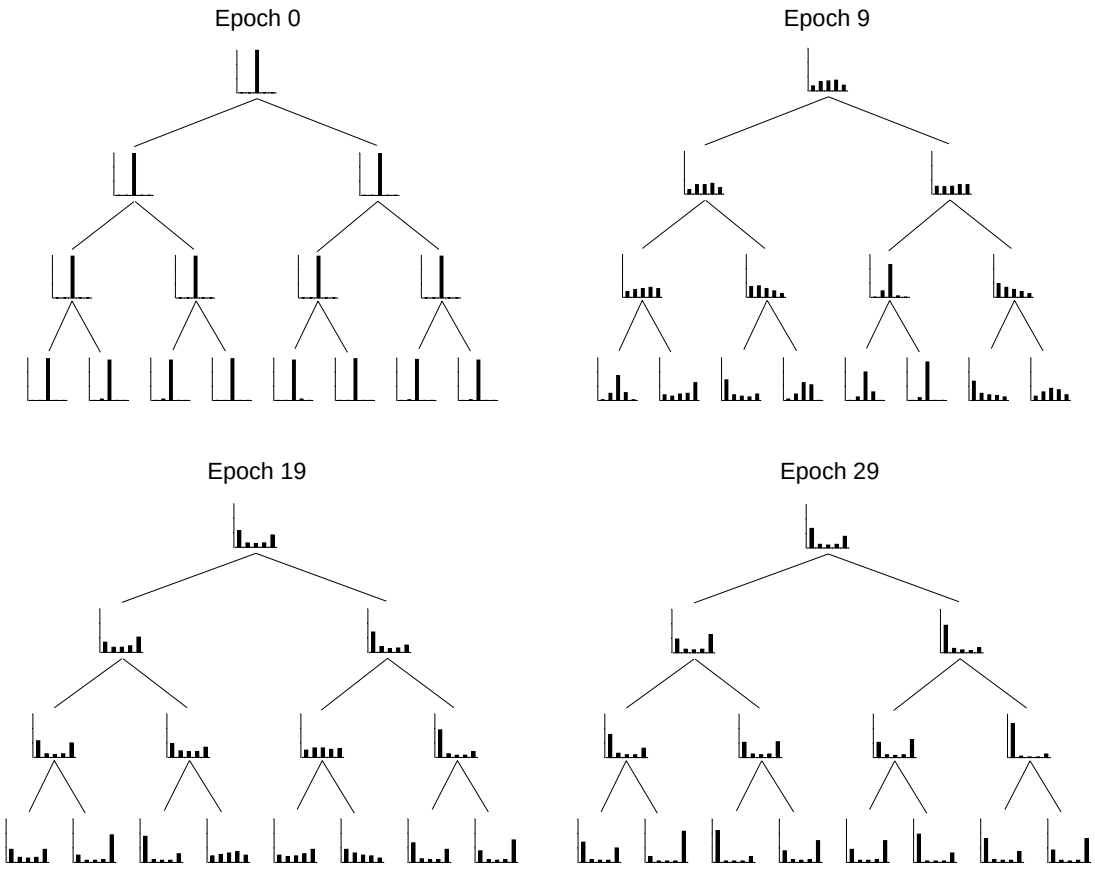
$$\tilde{\alpha}_c = \pi\sqrt{1 + \pi^2/8} \approx 4.695$$


0
2.5
5
7.5
10
12.5
15

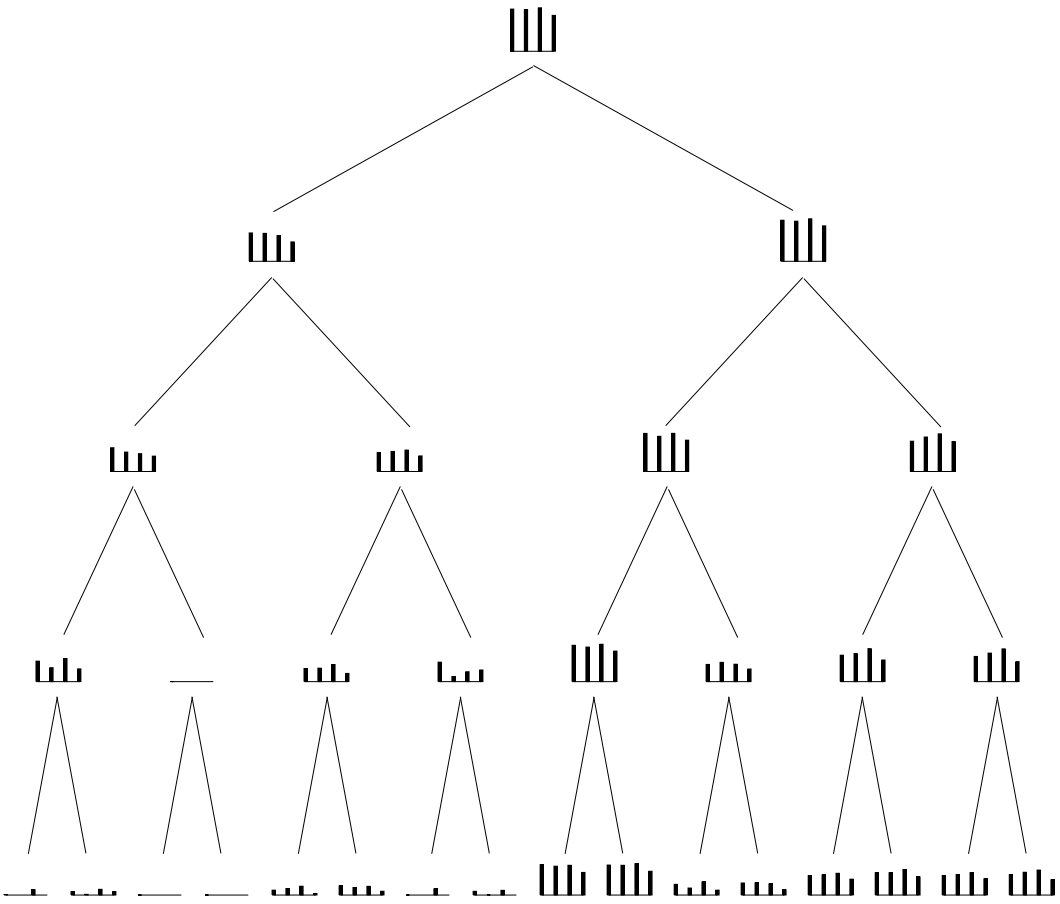

0
2.5
5
7.5
10
12.5
15


0
2.5
5
7.5
10
12.5
15

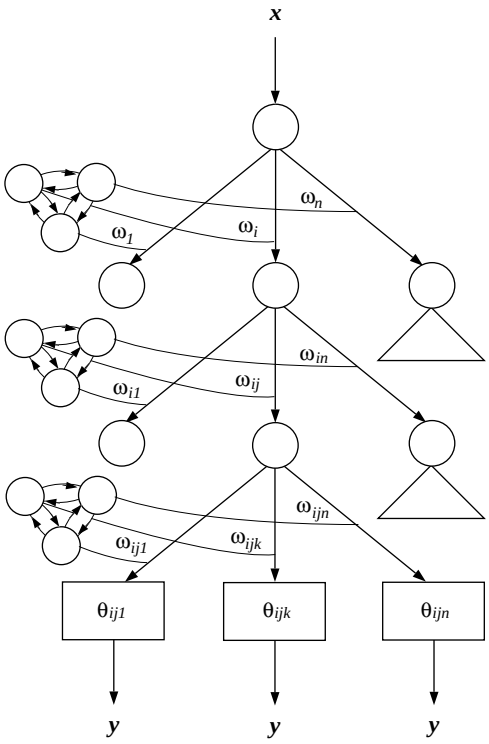
A Histogram Tree



A Deviance Tree



Hidden Markov Decision Trees



- Each decision at a node is dependent on the decision at the previous moment at that node
  - This yields a Markov model at each node
- An EM algorithm can be derived, treating the Markov states as hidden variables
  - It combines a forward-backward pass with an up-down pass

Conclusions

- A probabilistic approach to decision tree modeling
  - ridge function splits
  - smooth regression functions
  - any GLIM can be used as a leaf model
- EM algorithm (and SEM)
- Bayesian methods
  - Gibbs' sampling
  - mean-field methods