

CSE 6740 Lecture 12

How Can I Learn Fancier (Compositional) Models? (Trees, Networks, and Graphical Models)

Alexander Gray

agray@cc.gatech.edu

Georgia Institute of Technology

Quiz Answers

1. The kernel trick can be applied to any machine learning method. F.
2. Any similarity function is a kernel. F.

Today

1. Tree-structured models: Decision Trees
2. Graph-structured models I: Neural Networks
3. Graph-structured models II: Graphical Models

Tree-structured Models: Decision Trees

A way of making complex models out of simpler parts, within a tree (hierarchical) structure.

Decision Trees

Here's another nonparametric method, which has variations for different tasks (*classification trees, regression trees, density trees*).

Let's consider the regression case first, using squared error $\sum_i (y_i - \hat{f}(x_i))^2$.

Decision Trees

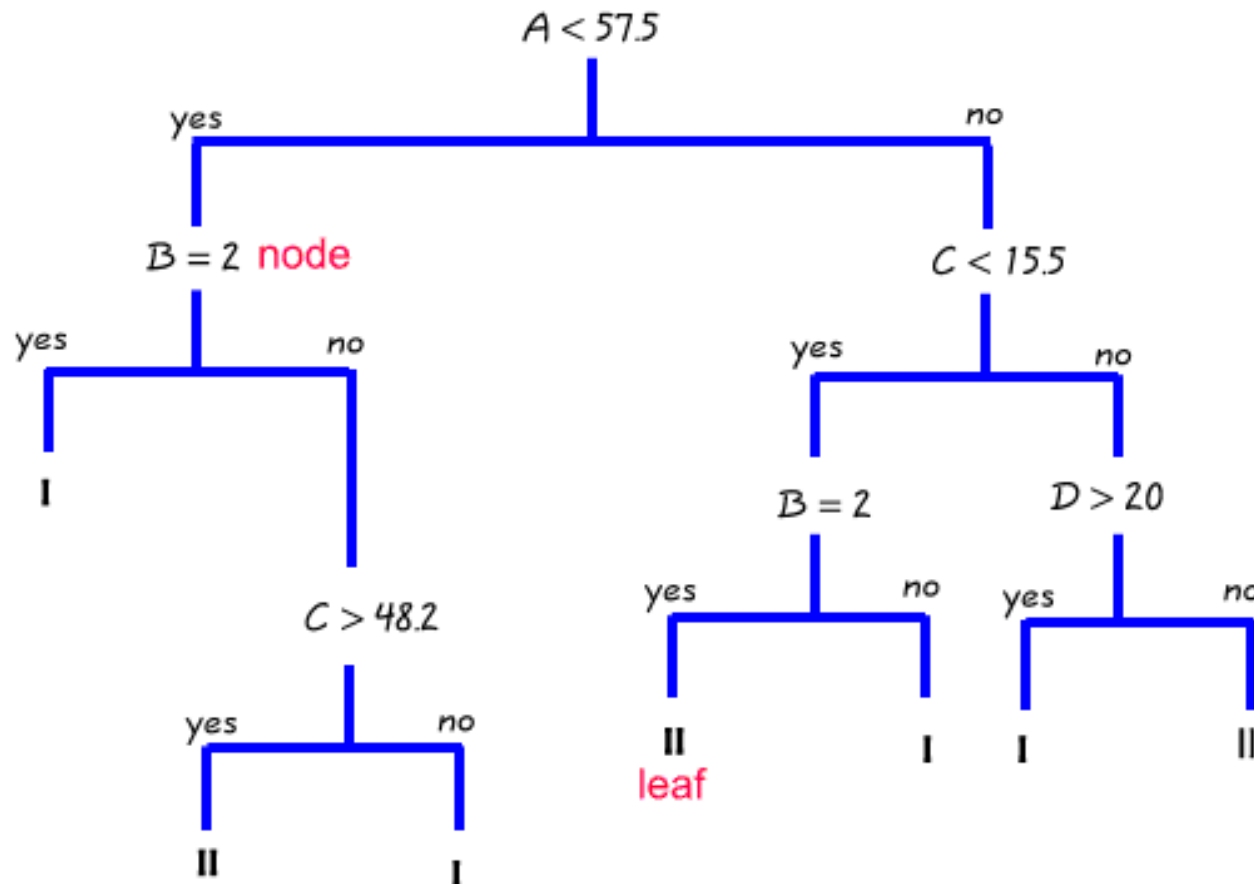
Each node is a hyperrectangular subset of the feature space, so that the overall model is of the form

$$\hat{f}(x) = \sum_m c_m I(x \in R_m). \quad (1)$$

We want hyperrectangles which try to contain points all having similar Y values. Given hyperrectangles, it's easy to see that the best \hat{c}_m is the average in the node, $\hat{c}_m = 1/N_m \sum_i y_i I(x_i \in R_m)$, or $\hat{c}_m = 1/N_m \sum_{x_i \in R_m} y_i$.

Decision Trees

Finding the optimal set of hyperrectangles is generally intractable computationally, so we'll proceed greedily, chopping up the input space one dimension at a time:



Decision Trees

Starting with all the data, consider a splitting variable d and split point s , and define the pair of half-planes

$$R_1(d, s) = \{X | X_d \leq s\} \quad \text{and} \quad R_2(d, s) = \{X | X_d > s\}. \quad (2)$$

Then we seek the splitting variable d and split point s corresponding to

$$\min_{d,s} \left(\min_{c_1} \sum_{x_i \in R_1(d,s)} (y_i - c_1)^2 \right. \quad (3)$$

$$\left. + \min_{c_2} \sum_{x_i \in R_2(d,s)} (y_i - c_2)^2 \right). \quad (4)$$

Decision Trees

For any choice of d and s , the inner minimization is solved by

$$\hat{c}_1 = 1/N_1 \sum_i y_i I(x_i \in R_1) \quad (5)$$

$$\hat{c}_2 = 1/N_2 \sum_i y_i I(x_i \in R_2). \quad (6)$$

To minimize over d and s , we can evaluate the inner expression over all possible choices. This can be done relatively efficiently by running over the sorted values of Y .

Decision Trees

We can keep doing this recursively in a top-down fashion. Then we need to determine when to stop splitting nodes.

Equivalently, we can build the tree to completion (the tree T_0 where every data point is a leaf, say), then decide which nodes to collapse to obtain a pruned tree $T \subset T_0$.

Decision Trees

Since the number of leaves of the tree $|T|$ can be thought of as a complexity parameter, we can use to obtain a regularization of the training set error:

$$Q_m(T) = \frac{1}{N_m} \sum_{x_i \in R_m} (y_i - c_m)^2 \quad (7)$$

$$C_\lambda(T) = \sum_{m=1}^{|T|} N_m Q_m(T) + \lambda |T|. \quad (8)$$

We can then cross-validate over λ to minimize the *cost-complexity* criterion $C_\lambda(T)$.

If more than one subtree minimizes this criterion, we use the smallest one.

Decision Tree

For classification, we simply minimize a classification loss instead, such as the cross-entropy (recall that this corresponds to maximum likelihood with a certain error model). First define

$$\hat{p}_{mc} = \frac{1}{N_m} \sum_{x_i \in R_m} I(y_i = c), \quad (9)$$

the proportion of class c observations in node m . We classify the observations in node m as $\hat{c}(m) = \arg \max_c \hat{p}_{mc}$, the majority class in node m . $Q_m(T)$ is now

$$Q_m(T) = - \sum_c \hat{p}_{mc} \log \hat{p}_{mc}. \quad (10)$$

Decision Tree

Note that the actual form of the model is a set of *rules*, which are conjunctions of univariate tests, resulting in a piecewise constant model. The way of obtaining the rules happens to be a computationally efficient (but suboptimal) algorithm.

Task: Regression or classification.

Model class: Nonparametric.

Loss: Squared-error (regression case), likelihood (aka cross-entropy, classification case).

Optimizer: Top-down greedy split selection.

Generalization mechanism: Cross-validation over regularization parameter.

Graph-structured Models I: Neural Networks

A way of making complex models out of simpler parts, within a (typically layered) graph structure.

Neural Network

Neural networks are models of the form

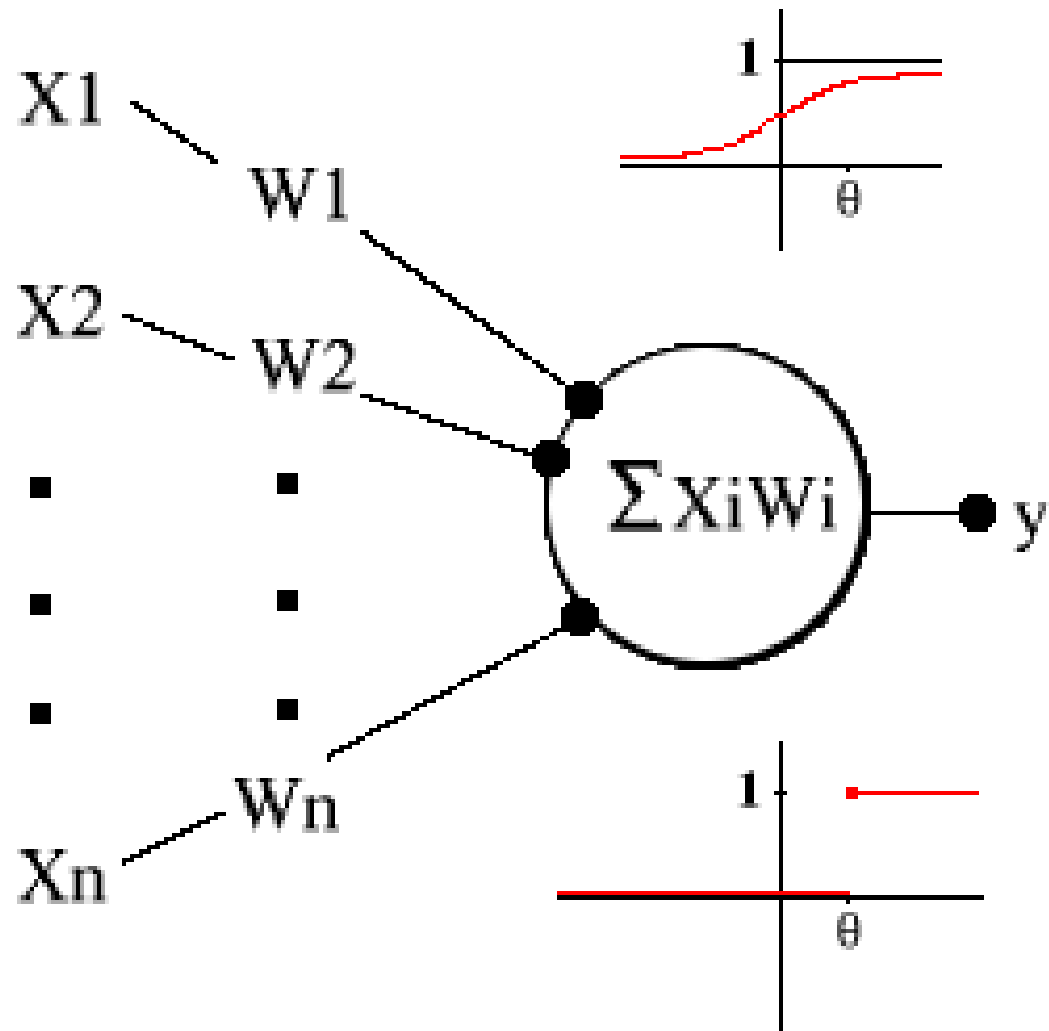
$$Y = \beta_0 + \sum_j \beta_j \sigma(\alpha_0 + \alpha^T X) \quad (11)$$

where σ is a smooth *sigmoidal* function, often having the form

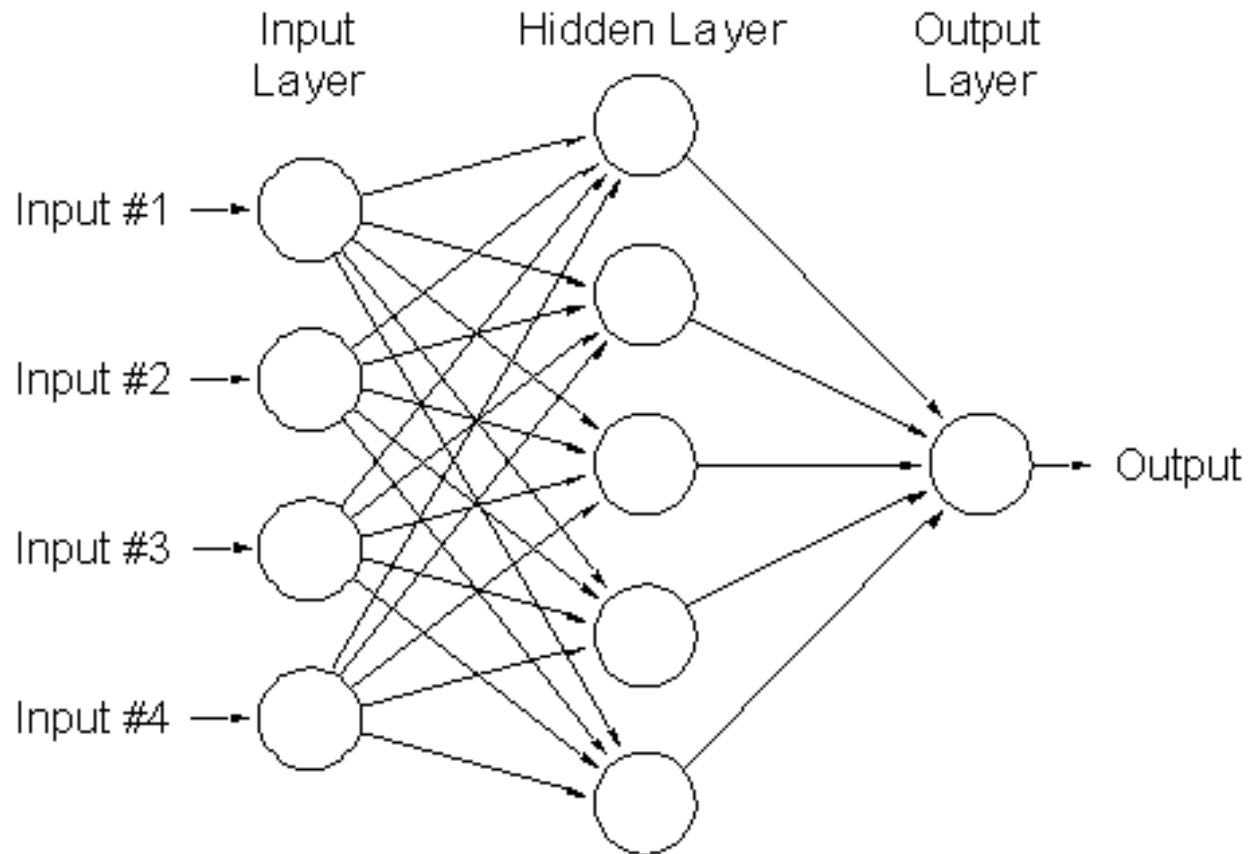
$$\sigma(t) = \frac{e^t}{1 + e^t}. \quad (12)$$

You can represent this as a graph having multiple layers of nodes. You can also seamlessly train for multiple targets at the same time.

Neural Network



Neural Network



Neural Network

This was traditionally optimized using the same method we saw for the perceptron. It was a big deal when various people figured out how to compute the derivative in a model where there are multiple layers, leading to what was called *back-propagation*.

Neural Network

There is also a regression version, which just uses a different loss function.

Task: Regression or classification.

Model class: Nonparametric.

Loss: Squared error (regression case), 0-1 loss (classification case).

Optimizer: Conjugate gradient or other unconstrained optimization, including stochastic gradient descent (called “back-propagation”).

Generalization mechanism: Cross-validation over regularization parameter (called “weight decay”).

Graph-structured Models II: Graphical Models

A way of making complex models out of simpler parts, within a graph structure representing relationships between random variables. The exposition is from Kevin Murphy's online tutorial.

Graphical Models

Graphical models are graphs in which nodes explicitly represent random variables, and the (lack of) arcs represent conditional independence assumptions. Hence they provide a compact representation of joint probability distributions.

There are two major kinds, depending on whether the arcs are directed or not:

1. *Undirected* graphical models, also called *Markov Random Fields* (MRF's) or *Markov networks*
2. *Directed* graphical models also called *Bayesian Networks* or *Belief Networks* (BN's).

There are also mixed graphs, sometimes called *chain graphs*.

Undirected Graphical Models

Undirected graphical models have a simple definition of independence: two (sets of) nodes A and B are conditionally independent given a third set, C , if all paths between the nodes in A and B are separated by a node in C .

Undirected graphical models are more popular with the physics and vision communities. Think of the pixel values of an image as variables, where the arcs simply encode neighborhood relationships. We won't say much more about them.

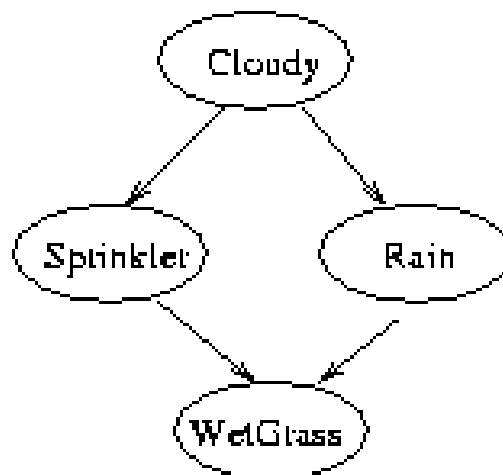
Directed Graphical Models

Directed graphical models contain more information - in particular one can regard an arc from A to B as indicating that A “causes” B . Undirected graphical models are less tractable because of the lack of an ordering between variables.

In addition to the graph structure, it is necessary to specify the parameters of the model. For a directed model, we must specify the *conditional probability distribution* (CPD) at each node. If the variables are discrete, this can be represented as a table (CPT), which lists the probability that the child node takes on each of its different values for each combination of values of its parents.

Sprinkler Example

	$P(C=F)$	$P(C=T)$
	0.5	0.5



C	$P(S=F)$	$P(S=T)$
F	0.5	0.5
T	0.9	0.1

C	$P(R=F)$	$P(R=T)$
F	0.8	0.2
T	0.2	0.8

S	R	$P(W=F)$	$P(W=T)$
F	F	1.0	0.0
T	F	0.1	0.9
F	T	0.1	0.9
T	T	0.01	0.99

Sprinkler Example

We see that the event "grass is wet" ($W=1$) has two possible causes: either the water sprinkler is on ($S=1$) or it is raining ($R=1$). The strength of this relationship is shown in the table. For example, we see that $\mathbb{P}(W = 1|S = 1, R = 1) = 0.9$ (second row), and hence, $\mathbb{P}(W = 0|S = 1, R = 0) = 1 - 0.9 = 0.1$, since each row must sum to one.

Since the C node has no parents, its CPT specifies the prior probability that it is cloudy (in this case, 0.5).

Conditional Independence

Directed graphical models have a more complicated notion of independence, which takes into account the directionality of the arcs.

The simplest conditional independence relationship encoded in a Bayesian network can be stated as follows: a node is independent of its ancestors given its parents, where the ancestor/parent relationship is with respect to some fixed topological ordering of the nodes.

Factorization of the Joint Distribution

By the chain rule of probability, the joint probability of all the nodes in the graph above is

$$\mathbb{P}(C, S, R, W) = \mathbb{P}(C) \cdot \mathbb{P}(S|C) \cdot \mathbb{P}(R|C, S) \cdot \mathbb{P}(W|C, S, R) \quad (13)$$

By using conditional independence relationships, we can rewrite this as

$$\mathbb{P}(C, S, R, W) = \mathbb{P}(C) \cdot P(S|C) \cdot P(R|C) \cdot P(W|S, R) \quad (14)$$

where we were allowed to simplify the third term because R is independent of S given its parent C , and the last term because W is independent of C given its parents S and R .

Factorization of the Joint Distribution

We can see that the conditional independence relationships allow us to represent the joint more compactly.

Here the savings are minimal, but in general, if we had D binary nodes, the full joint would require $O(2^D)$ space to represent, but the factored form would require $O(D2^m)$ space to represent, where m is the maximum fan-in of a node. And fewer parameters makes learning easier.

Bayesian Networks $\not\Rightarrow$ Bayesianism

Bayesian networks do not necessarily imply a commitment to Bayesian statistics. Indeed, it is common to use frequentists methods to estimate the parameters of the CPD's. They are so-called because of Bayes Theorem about conditional probabilities.

It turns out that Bayes nets are a useful representation for hierarchical Bayesian models. In such a model, the parameters are treated like any other random variable, and become nodes in the graph.

Conditional Probability Inference

“Inference” is a general statistical term. In the context of graphical models it means finding out the probability distribution for a variable of interest given that some of the other variables have fixed values.

For example, suppose we observe the fact that the grass is wet. There are two possible causes for this: either it is raining, or the sprinkler is on. Which is more likely?

Conditional Probability Inference

We can use Bayes' rule to compute the posterior probability of each explanation.

$$\begin{aligned}\mathbb{P}(S = 1|W = 1) &= \frac{\mathbb{P}(S = 1, W = 1)}{\mathbb{P}(W = 1)} && (15) \\ &= \frac{\sum_{c,r} \mathbb{P}(C = c, S = 1, R = r, W = 1)}{\mathbb{P}(W = 1)} = (1.6)\end{aligned}$$

$$\begin{aligned}\mathbb{P}(R = 1|W = 1) &= \frac{\mathbb{P}(R = 1, W = 1)}{\mathbb{P}(W = 1)} && (17) \\ &= \frac{\sum_{c,s} \mathbb{P}(C = c, S = s, R = 1, W = 1)}{\mathbb{P}(W = 1)} = (1.8)\end{aligned}$$

Conditional Probability Inference

where

$$\mathbb{P}(W = 1) = \sum_{c,r,s} \mathbb{P}(C = c, S = s, R = r, W = 1) = 0.65 \quad (19)$$

is the normalizing constant.

So we see that it is more likely that the grass is wet because it is raining.

Explaining Away

In the above example, notice that the two causes "compete" to "explain" the observed data. Hence S and R become conditionally dependent given that their common child, W , is observed, even though they are marginally independent.

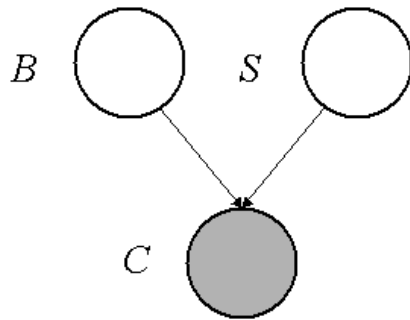
For example, suppose the grass is wet, but that we also know that it is raining. Then the posterior probability that the sprinkler is on goes down:

$$\mathbb{P}(S = 1 | W = 1, R = 1) = 0.19. \quad (20)$$

This is called "explaining away". In statistics, this is known as Berkson's paradox, or "selection bias".

Explaining Away

For a dramatic example of this effect, consider a college which admits students who are either brainy or sporty. Let C denote the event that someone is admitted to college, which is made true if they are either brainy (B) or sporty (S). Suppose in the general population, B and S are independent:



Explaining Away

Now look at a population of college students (those for which C is observed to be true). It will be found that being brainy makes you less likely to be sporty and vice versa, because either property alone is sufficient to explain the evidence on C *i.e.*,

$$\mathbb{P}(S = 1|C = 1, B = 1) \leq \mathbb{P}(S = 1|C = 1). \quad (21)$$

Bottom-up and Top-down Reasoning

In the water sprinkler example, we had evidence of an effect (wet grass), and inferred the most likely cause. This is called *diagnostic*, or *bottom-up*, reasoning, and is what “expert systems” were mainly concerned with.

Bayes nets are generative, so they can also be used for *causal*, or *top-down*, reasoning, e.g. computing the probability that the grass will be wet given that it is cloudy.

Causality

We can rigorously make inferences about *causality*, *i.e.* distinguish causation from mere correlation, by measuring the relationships between at least three variables; the intuition is that one of the variables acts as a "virtual control" for the relationship between the other two, so we don't always need to do experiments to infer causality.

This is its own rich sub-topic. It turns out that computational difficulty is the main obstacle here.

Discrete and Continuous Nodes

Now let's look at some useful models in this framework.

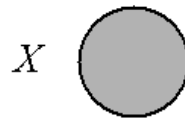
The introductory example used nodes with categorical values and multinomial distributions. It is also possible to create Bayesian networks with continuous valued nodes. The most common distribution for such variables is the Gaussian. For discrete nodes with continuous parents, we can use the logistic/softmax distribution.

Using multinomials, conditional Gaussians, and the softmax distribution, we can have a rich toolbox for making complex models.

Simple Gaussian

Circles denote continuous-valued random variables, squares denote discrete rv's, clear means hidden, and shaded means observed.

$$f(X) = N(\mu, \sigma^2) \quad (22)$$



Mixtures of Gaussians

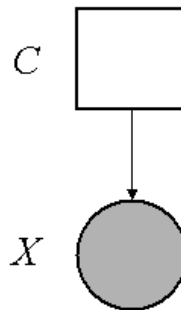
Recall the mixture of Gaussians model:

$$P(C = k) = \pi_k \quad (23)$$

$$f(X|C = k) = N(\mu_k, \sigma_k^2) \quad (24)$$

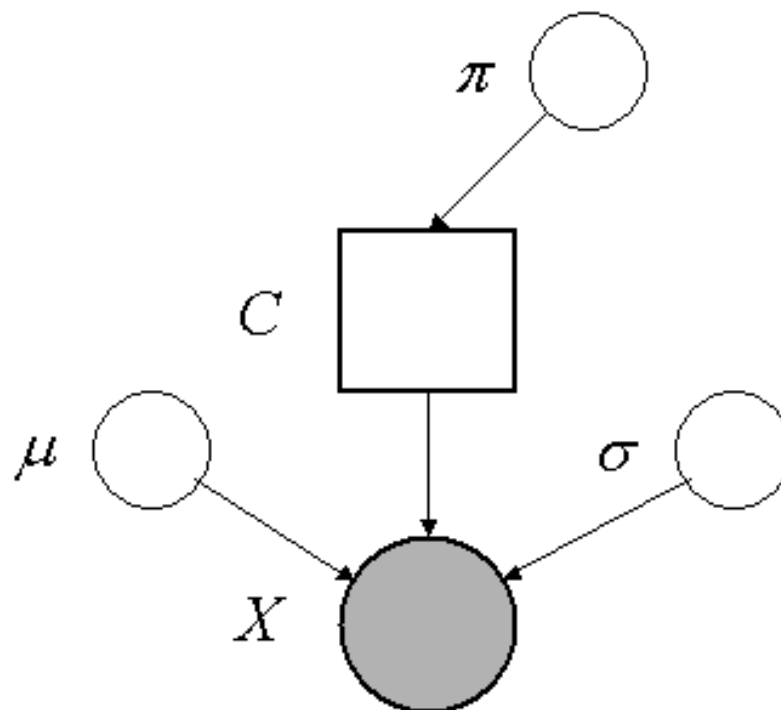
$$f(X) = \sum_{k=1}^K f(X|C = k)P(C = k) = \sum_{k=1}^K \pi_k N(\mu_k, \sigma_k^2) \quad (25)$$

where $\sum_k \pi_k = 1$.



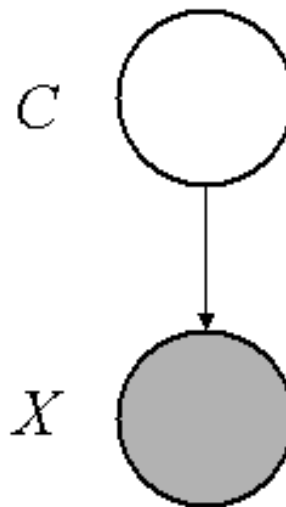
Bayesian Mixtures of Gaussians

Now each parameter is also a random variable. We'll denote them with small nodes.



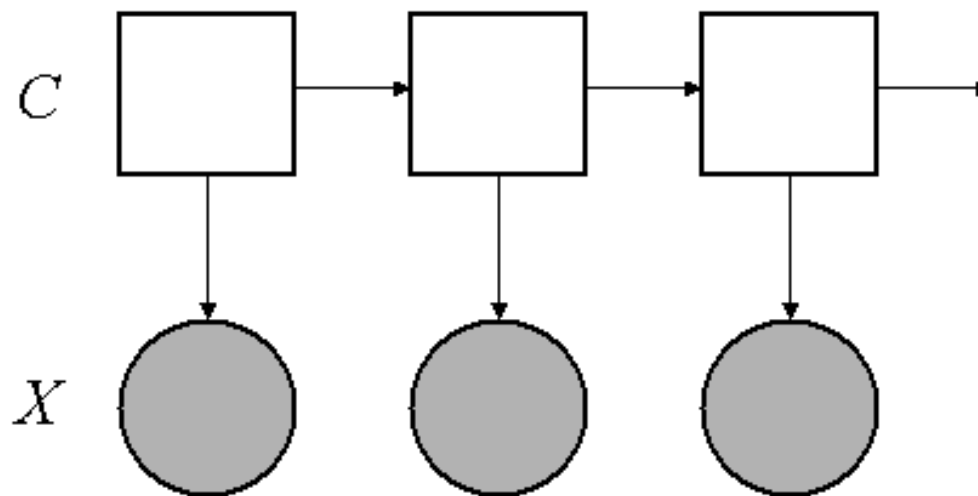
Principal Component Analysis

A model for dimensionality reduction. We'll come to this later.



Hidden Markov Model

A model for dependent (time series) data. Like a temporal mixture of Gaussians.



Kalman Filter

A model for dependent (time series) data, where the hidden variable is continuous (a Gaussian).

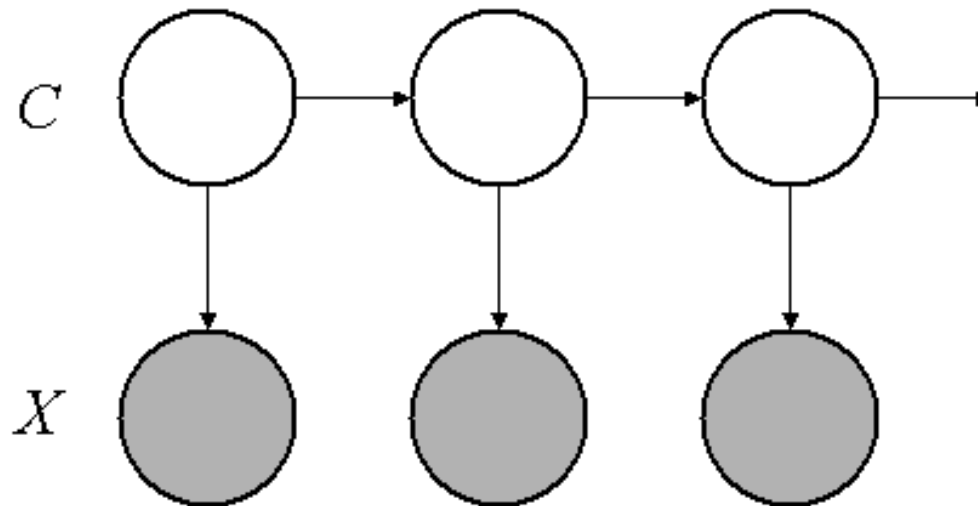
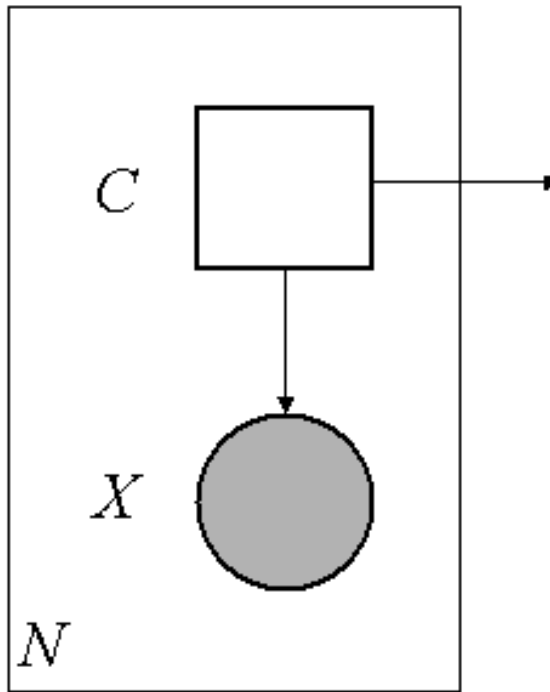


Plate Notation

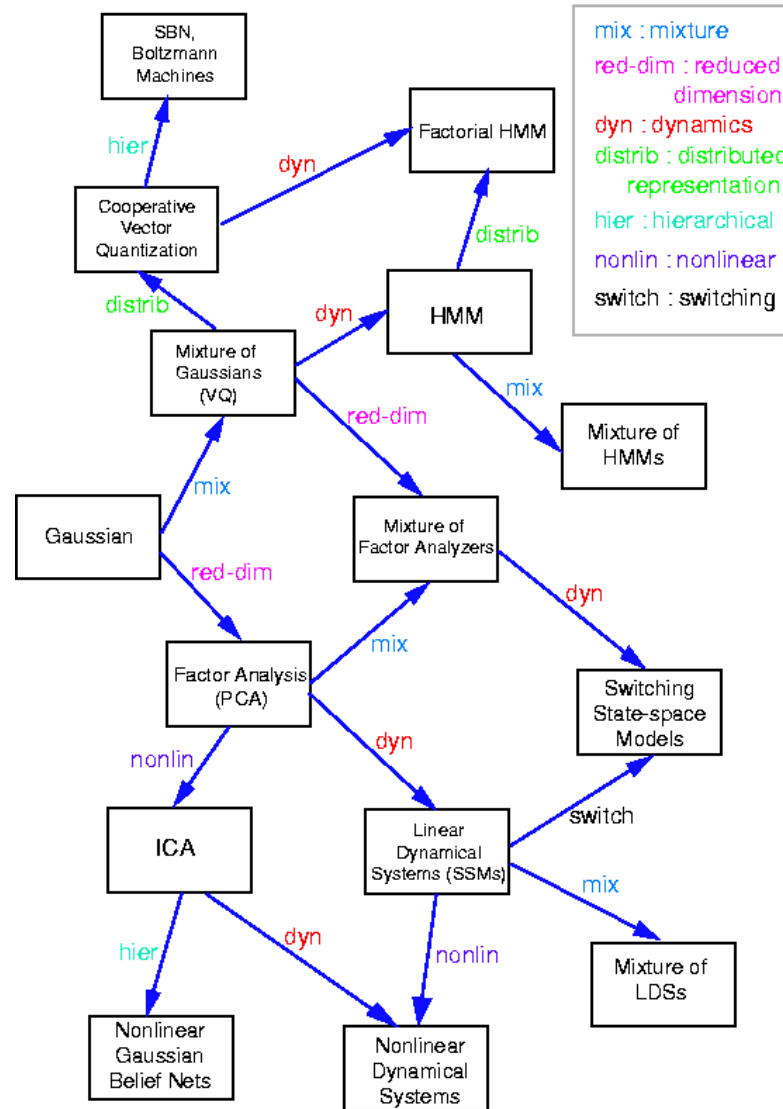
A plate, denoted by a surrounding square, indicates repetition of the graphical model it contains.



Model Lego Blocks

- Mixture version
- Hierarchical version
- Nonlinear version
- Dynamic version
- Switching version
- Dimension reduction version
- Distributed version

Model Mania



Graphical Model Computations

Now let's look at how to compute basic quantities in graphical models. Let (E, Q) be a partitioning of the variables (“evidence” and “query” variables). Two main quantities we desire are:

- *Marginal probabilities:*

$$\mathbb{P}(Q = q) = \sum_e \mathbb{P}(Q = q, E = e) \quad (26)$$

or $\mathbb{P}(E = e) = \sum_q \mathbb{P}(Q = q, E = e)$.

- *Maximum a posteriori (MAP) probabilities:*

$$\mathbb{P}^*(Q = q) = \max_e \mathbb{P}(Q = q, E = e). \quad (27)$$

Graphical Model Computations

From these basic quantities we can obtain other quantities such as *conditional probabilities*:

$$\mathbb{P}(Q = q | E = e) = \frac{\mathbb{P}(Q = q, E = e)}{\mathbb{P}(E = e)} = \frac{\mathbb{P}(Q = q, E = e)}{\sum_q \mathbb{P}(Q = q, E = e)}. \quad (28)$$

In general multiple marginalizations (summations) must be performed. For example if (E, Q, H) (“evidence”, “query”, and “hidden”) is a partitioning of the variables,

$$\mathbb{P}(Q = q | E = e) = \frac{\mathbb{P}(Q = q, E = e)}{\sum_q \mathbb{P}(Q = q, E = e)} \quad (29)$$

$$= \frac{\sum_h \mathbb{P}(Q = q, E = e, H = h)}{\sum_q \sum_h \mathbb{P}(Q = q, E = e, H = h)}. \quad (30)$$

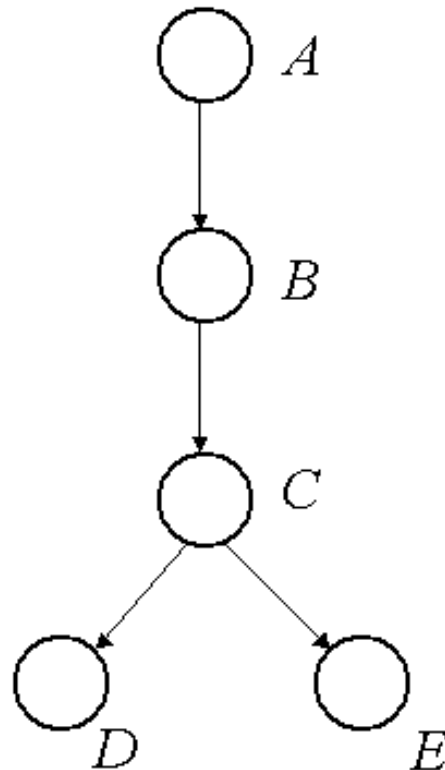
Graphical Model Computations

It would be nice if we could treat both types of graphs in the same way. Note that the directed graph equation is a special case of the undirected graph equation, except that the parent sets are not necessarily cliques. If we modified the original graph a bit, we could make it a special case on the modified graph.

The *moral graph* is an undirected graph obtained by connecting all the parents of each node in the original graph and removing the directions.

Elimination

The core algorithmic idea is that of *elimination*. Let's do an example:



Elimination

We'll develop an algorithm for computing a single marginal probability $\mathbb{P}(E = e)$, for which we'll use the shorthand $P(e)$. Starting with the joint probability we'll have to, in some fashion, sum over every variable other than E :

$$P(e) = \sum_a \sum_b \sum_c \sum_d P(a, b, c, d, e). \quad (31)$$

We must pick an order in which to sum - we'll pick the order (a, b, d, c) . We'll also write the joint in the compact form afforded by knowledge of the conditional independencies specified by the graph:

$$P(e) = \sum_c \sum_d \sum_b \sum_a P(a)P(b|a)P(c|b)p(d|c)p(e|c). \quad (32)$$

Elimination

Now let's rearrange this summation so that we push in sums as far rightward as possible, based on the variables they depend on:

$$P(e) = \sum_c p(e|c) \sum_d p(d|c) \sum_b p(c|b) \sum_a p(a)p(b|a). \quad (33)$$

Each time we finish a summation over a variable, we end up with an expression, which we can think of as a “message” which we pass to the next summation:

$$P(e) = \sum_c p(e|c) \sum_d p(d|c) \sum_b p(c|b)m_{ab}(b). \quad (34)$$

Elimination

Continuing on in this fashion,

$$p(e) = \sum_c p(e|c) \sum_d p(d|c) \sum_b p(c|b) m_{ab}(b) \quad (35)$$

$$= \sum_c p(e|c) \sum_d p(d|c) m_{bc}(c) \quad (36)$$

$$= \sum_c p(e|c) m_{bc}(c) \sum_d p(d|c) \quad (37)$$

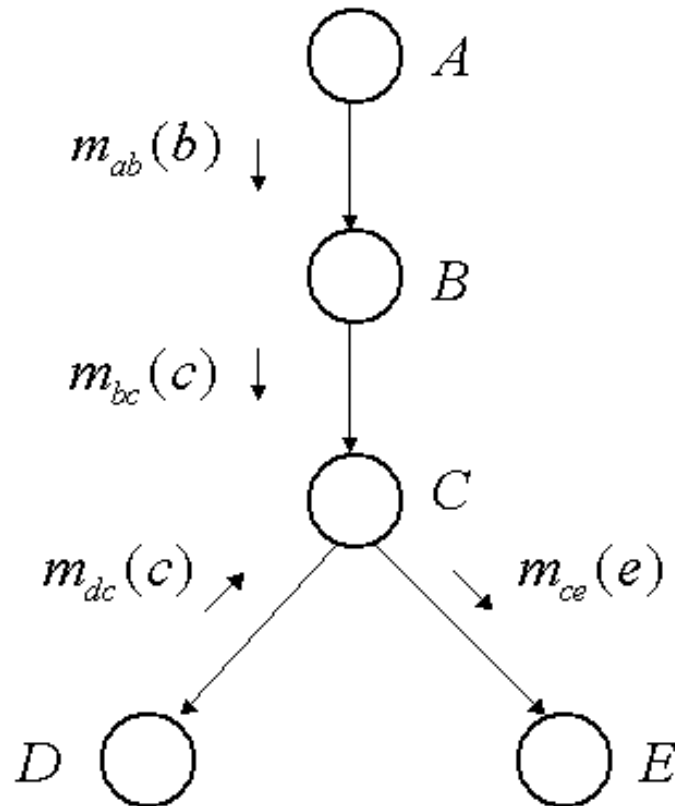
$$= \sum_c p(e|c) m_{bc}(c) m_{dc}(c) \quad (38)$$

$$= m_{ce}(e). \quad (39)$$

The final expression is a function of e only and is the desired marginal probability.

Elimination

Here are the messages created during this run of the elimination algorithm:



Main Things You Should Know

- What a decision tree is
- What a neural network is
- What graphical models are and why they are useful
- The computations needed in graphical models

Quiz

1. (T/F) A decision tree is regularized.
2. (T/F) A neural network is nonparametric.
3. (T/F) A graphical model encodes dependencies between random variables.