Bayesian network

Yuzhen Ye
School of Informatics and Computing
Indiana University, Bloomington
Spring 2013
Contents

- Probabilistic graphical models overview
- Bayesian network overview
- Probabilistic inference in Bayesian networks
  - Exact solutions: Enumeration & Variable elimination
  - Approximate approaches: Sampling & MCMC
- Learning Bayesian networks
  - Learning parameters (given model structure)
    - No missing data: MLE
    - Missing data: EM & MCMC
  - Learning graph structure (model selection)
- Bayesian network classifiers
  - Tree augmented NB
Probabilistic graphical models

- Graphical models are a marriage between probability theory and graph theory (Michael Jordan, 1998)

- Graphical models use conditional independence assumptions for efficient representation, inference and learning of joint distributions
  - a compact representation of joint probability distributions;
  - a collection of conditional independence assumptions

- Graphs
  - nodes: random variables (probabilistic distribution over a fixed alphabet)
  - edges (arcs), or lack of edges: conditional independence assumptions
## Classification of probabilistic graphical models

<table>
<thead>
<tr>
<th></th>
<th>Linear</th>
<th>Branching</th>
<th>Application</th>
</tr>
</thead>
<tbody>
<tr>
<td>Directed</td>
<td>Markov Chain (HMM)</td>
<td><strong>Bayesian network</strong> (BN)</td>
<td>Artificial Intelligence (AI)</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>Statistics</td>
</tr>
<tr>
<td>Undirected</td>
<td>Linear chain conditional random field (CRF)</td>
<td>Markov network (MN)</td>
<td>Physics (Ising model)</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>Image/Vision</td>
</tr>
</tbody>
</table>

Both directed and undirected arcs: chain graphs
Bayesian Network Structure

- Directed acyclic graph (DAG) \( G \)
  - Nodes \( x_1, \ldots, x_n \) represent random variables; the parent nodes of \( x_i \), \( (pa_i) \) represents the set of variables that \( x_i \) is dependent on.
  - The variables can be discrete or continuous

- \( G \) encodes local Markov assumptions
  - Conditional independence
    - \( X_i \perp X_j | X_k \)
  - \( x_i \) is independent of its non-descendants given its parents
  - The dependences are modeled by Conditional Probability Distributions (CPDs; for continuous variables) or Conditional Probability Tables (CPTs; for discrete variables)

\[
P(A,B,C,D,E,F,G) = P(A)P(B|A)P(C|A)P(D|B)P(E|B)P(G|E)P(F|C)
\]

- 7 CPDs: each for one edge or one source node

- BN computes the joint distribution of all random variables compactly in a factorized way.
- BN is a compact representation of conditional independence assumptions about a high dimension distribution.
Figure 2. Illustration of Model Parameters for Two-Node Bayesian Network

squares for discrete nodes and circles for continuous nodes

\[
\theta_B = p(B \mid A) \\
\begin{array}{l|c}
A = a_1 & \mu_1, \sigma_1^2 \\
A = a_2 & \mu_2, \sigma_2^2 \\
A = a_3 & \mu_3, \sigma_3^2 \\
\end{array}
\]

http://www.ploscompbiol.org/article/info:doi/10.1371/journal.pcbi.0030129
BN provides compact representation of conditional independence

Joint probabilities: 143 parameters
Conditional probabilities: 24

A network with 100 nodes, each with 3 possible values: \( > 10^{47} \) vs 1,800 parameters!
Conditional independence in BNs: Types of connections

**Serial**
Knowing GP makes RE and CR independent (intermediate cause)

**Diverging**
Knowing SI makes IN and RE independent (common cause)

**Converging**
**NOT** Knowing GP makes RE and SI independent (common effect)
Why Bayesian network?

- Combined with Bayesian method, Bayesian Network can offer solutions to a number of challenges
  - Facilitate the combination of domain knowledge and data
  - Handle incomplete data sets (marginalizing over unknown variables by considering all possible values the unknown variables may take, and averaging over them)
  - Offer an efficient and principled approach for avoiding the over fitting of data
  - Learn about causal relationships
Inference in a Bayesian network

- Inference in probabilistic models in general asks the following questions: given \( P(X_1, X_2, \ldots, X_m) \) and a set of observations \( e = \{X_i = x_i, X_j = x_j, \ldots\} \) (or data), compute
  - Marginals: \( P(X_k | e) \)
  - Probability of evidence: \( P(e) \)
  - Most probable explanation:
    \[
    \arg \max_x P(x | e)
    \]
Approaches to inference

- **Exact methods**
  - Enumeration
  - **Variable elimination**
  - Belief propagation in polytrees
  - Clustering / join tree algorithms

- **Approximate methods**
  - Stochastic simulation / sampling methods
  - Markov chain Monte Carlo
  - Genetic algorithms
  - Neural networks
  - Simulated annealing
  - Mean field theory
A simple example: credit card fraud

Conditional Probability Tables (CPTs)

P(f=yes)=0.00001
P(f=no)=0.99999

P(a=<30)=0.25
P(a=30-50)=0.40
P(a=>50)=0.35

P(s=male)=0.5
P(s=female)=0.5

P(g=yes|f=yes)=0.2
P(g=no|f=yes)=0.8
P(g=yes|f=no)=0.01
P(g=no|f=no)=0.99

P(j=yes|f=yes,a=*,s=*)=0.05,
P(j=yes|f=no,a=<30,s=male)=0.0001
P(j=yes|f=no,a=30-50,s=male)=0.0004
P(j=yes|f=no,a=>50,s=male)=0.0002
P(j=yes|f=no,a=<30,s=female)=0.0005
P(j=yes|f=no,a=30-50,s=female)=0.0002
P(j=yes|f=no,a=>50,s=female)=0.0001
...
### Inference in the BN

- **BN defines the joint probability for all involved random variables**

\[ p(f,a,s,g,j) = p(f) \cdot p(g \mid f) \cdot p(a) \cdot p(s) \cdot p(j \mid f,a,s) \]

- **One can use BN to compute any probability of interest**
  
  - Computing posterior marginal probability, e.g. The probability of fraud, given the **evidences** \((a,s,g,j)\),

\[
p(f \mid a,s,g,j) = \frac{p(f,a,s,g,j)}{\sum_{f'} p(f',a,s,g,j)} = \frac{p(f) \cdot p(g \mid f) \cdot p(a) \cdot p(s) \cdot p(j \mid f,a,s)}{\sum_{f'} p(f') \cdot p(g \mid f') \cdot p(a) \cdot p(s) \cdot p(j \mid f',a,s)}
\]

When variables are in a set of discrete values, this can be computed!
Inference by enumeration (examples)

- The probability of fraud (age=30-50, sex=female, gas=yes, jewelry=yes)

\[
p(f \mid a, s, g, j) = \frac{p(f) \cdot p(g \mid f) \cdot p(j \mid f, a, s)}{\sum_{f'} p(f') \cdot p(g \mid f') \cdot p(j \mid f', a, s)}
\]

\[
p(f = yes \mid a = 30 - 50, s = female, g = yes, j = yes) =
\]

\[
\frac{0.00001 \times 0.2 \times 0.05}{10^{-7} + 0.99999 \times 0.01 \times 0.002} = 0.005 > > \text{prior (0.00001)}
\]

- The probability that the card holder is female, if the card is not fraud

\[
p(s \mid a, f, g, j) = \frac{p(s) \cdot p(j \mid f, a, s)}{\sum_{s'} p(s') \cdot p(j \mid f, a, s')}
\]

\[
p(s = female \mid a = < 30, f = no, g = yes, j = yes) =
\]

\[
\frac{0.5 \times 0.0005}{2.5 \times 10^{-4} + 0.5 \times 0.0001} = \frac{5}{6}
\]
Inference with missing data

- The probability of fraud, but the **gender of the card holder is unknown**

\[
p(f \mid a, g, j) = \frac{\sum_s (f) \cdot p(g \mid f) \cdot p(j \mid f, a, s')}{\sum_{f', s'} p(f') \cdot p(g \mid f') \cdot p(j \mid f', a, s')}
\]

Don’t see \(p(s')\): \(p(s = male) = p(s = female)\)

\[
p(f = yes \mid a = 30 - 50, g = yes, j = yes) = \frac{0.00001 \times 0.2 \times (0.05 + 0.05)}{2 \times 10^{-7} + 0.999999 \times 0.01 \times (0.002 + 0.0004)} = 0.004
\]

(marginalizing over variable \(s\))
Computational complexity of inference by enumeration

- Computing the joint probability
  - Multiplication of CPDs, $O(n^{\mid V \mid})$
    - $n$: # discrete values; $v$: # variables (nodes)

$$p(f) = \sum_{s',a',g',j'} p(f) \cdot p(a') \cdot p(s') \cdot p(g' \mid f) \cdot p(j' \mid f, a', s')$$

$$\sum_{f',a',s',g',j'} p(f') \cdot p(a') \cdot p(s') \cdot p(g' \mid f') \cdot p(j \mid f', a', s')$$

To compute the denominator, all combinations of $(f, a, s, g, j)$ need to be enumerated, indicating the complexity of $2^5$.

Exact inference in an arbitrary BN for discrete variables is NP-hard (Cooper, 1987). When BN contains many undirected cycles (e.g., adding an edge $a \rightarrow g$ forming a cycle $f \rightarrow g \rightarrow a \rightarrow j \rightarrow f$), inference is intractable.
Inference by variable elimination (VE algorithm)

Consider a query that needs to compute the joint probability of \( X=(x_1, x_2, \ldots, x_k) \), where \( x_i \) represents a random variable (i.e., node)

\[
P(X \mid e) = \sum_{x_k} \cdots \sum_{x_2} \sum_{x_1} \prod_{i} P(x_i \mid pa_i)
\]

where \( e \) represents a subset of variables outside \( X \), \( pa_i \) represents the set of parent variables of \( x_i \).

The computation can be accelerated by a Dynamic Programming algorithm, which Iteratively
- move all irrelevant terms outside of innermost sum
- perform innermost sum, getting a new term
- insert the new term into the product
Variable elimination: Example

\[ P(d) = \sum_{a,b,c} P(d \mid b,c)P(b \mid a)P(c \mid a)P(a) \]

\[ = \sum_{b,c} P(d \mid b,c) \left( \sum_{a} P(b \mid a)P(c \mid a)P(a) \right) \]

\[ = \sum_{b,c} P(d \mid b,c) f_1(b,c) \]

\[ f_1(b,c) \text{ can be computed for each combination of (b,c), and used for computing } P(d). \]
A more complex example

\[ \sum_{v,s,x,t,l,a,b} P(v)P(s)P(t \mid v)P(l \mid s)P(b \mid s)P(a \mid t,l)P(x \mid a)P(d \mid a,b) \]
Eliminate: $\nu$

$$P(\nu)P(s)P(t \mid \nu)P(l \mid s)P(b \mid s)P(a \mid t, l)P(x \mid a)P(d \mid a, b)$$

Compute:

$$f_\nu(t) = \sum_\nu P(\nu)P(t \mid \nu)$$

$$\Rightarrow f_\nu(t)P(s)P(l \mid s)P(b \mid s)P(a \mid t, l)P(x \mid a)P(d \mid a, b)$$

Note: let $f_\nu(t) = P(t)$

In general, however, result of elimination is not necessarily a probability term.
Eliminate: $s$

\[ P(v)P(s)P(t \mid v)P(l \mid s)P(b \mid s)P(a \mid t, l)P(x \mid a)P(d \mid a, b) \]
\[ \Rightarrow f_v(t)P(s)P(l \mid s)P(b \mid s)P(a \mid t, l)P(x \mid a)P(d \mid a, b) \]

Compute:

\[ f_s(b, l) = \sum_s P(s)P(b \mid s)P(l \mid s) \]
\[ \Rightarrow f_v(t)f_s(b, l)P(a \mid t, l)P(x \mid a)P(d \mid a, b) \]

Summing on $s$ results in a dimensional matrix $f_s(b, l)$

In general, result of elimination may be a function of several variables.
Eliminate: $x$

\[ P(v)P(s)P(t \mid v)P(l \mid s)P(b \mid s)P(a \mid t, l)P(x \mid a)P(d \mid a, b) \]
\[ \Rightarrow f_v(t)P(s)P(l \mid s)P(b \mid s)P(a \mid t, l)P(x \mid a)P(d \mid a, b) \]
\[ \Rightarrow f_v(t)f_s(b, l)P(a \mid t, l)P(x \mid a)P(d \mid a, b) \]

Compute:

\[ f_x(a) = \sum_x P(x \mid a) \]
\[ \Rightarrow f_v(t)f_s(b, l)f_x(a)P(a \mid t, l)P(d \mid a, b) \]

Note: $f_x(a) = 1$ for all values of $a$!!
Eliminate: $t$

\[ P(v)P(s)P(t \mid v)P(l \mid s)P(b \mid s)P(a \mid t,l)P(x \mid a)P(d \mid a,b) \]
\[ \Rightarrow f_v(t)P(s)P(l \mid s)P(b \mid s)P(a \mid t,l)P(x \mid a)P(d \mid a,b) \]
\[ \Rightarrow f_v(t)f_s(b,l)P(a \mid t,l)P(x \mid a)P(d \mid a,b) \]
\[ \Rightarrow f_v(t)f_s(b,l)f_x(a)P(a \mid t,l)P(d \mid a,b) \]

Compute: 
\[ f_t(a,l) = \sum_t f_v(t)P(a \mid t,l) \]
\[ \Rightarrow f_s(b,l)f_x(a)f_t(a,l)P(d \mid a,b) \]
Eliminate: $I$

\[
P(v)P(s)P(t \mid v)P(l \mid s)P(b \mid s)P(a \mid t,l)P(x \mid a)P(d \mid a,b)
\Rightarrow f_v(t)P(s)P(l \mid s)P(b \mid s)P(a \mid t,l)P(x \mid a)P(d \mid a,b)
\Rightarrow f_v(t)f_s(b,l)P(a \mid t,l)P(x \mid a)P(d \mid a,b)
\Rightarrow f_v(t)f_s(b,l)f_x(a)P(a \mid t,l)P(d \mid a,b)
\Rightarrow f_s(b,l)f_x(a)f_t(a,l)P(d \mid a,b)
\]

Compute: $f_l(a,b) = \sum_t f_s(b,l)f_t(a,l)$

\[
\Rightarrow f_l(a,b)f_x(a)P(d \mid a,b)
\]
Eliminate: $a,b$

$$P(v)P(s)P(t \mid v)P(l \mid s)P(b \mid s)P(a \mid t,l)P(x \mid a)P(d \mid a,b)$$

$$\Rightarrow f_v(t)P(s)P(l \mid s)P(b \mid s)P(a \mid t,l)P(x \mid a)P(d \mid a,b)$$

$$\Rightarrow f_v(t)f_s(b,l)P(a \mid t,l)P(x \mid a)P(d \mid a,b)$$

$$\Rightarrow f_v(t)f_s(b,l)f_x(a)P(a \mid t,l)P(d \mid a,b)$$

$$\Rightarrow f_s(b,l)f_x(a)f_t(a,l)P(d \mid a,b)$$

$$\Rightarrow f_l(a,b)f_x(a)P(d \mid a,b) \Rightarrow f_a(b,d) \Rightarrow f_b(d)$$

Compute:

$$f_a(b,d) = \sum_a f_l(a,b)f_x(a)p(d \mid a,b) \quad f_b(d) = \sum_b f_a(b,d)$$
Complexity of VE algorithm

\[ P(v)P(s)P(t \mid v)P(l \mid s)P(b \mid s)P(a \mid t, l)P(x \mid a)P(d \mid a, b) \]
\[ \Rightarrow f_v(t)P(s)P(l \mid s)P(b \mid s)P(a \mid t, l)P(x \mid a)P(d \mid a, b) \]
\[ \Rightarrow f_v(t)f_s(b, l)P(a \mid t, l)P(x \mid a)P(d \mid a, b) \]
\[ \Rightarrow f_v(t)f_s(b, l)f_x(a)P(a \mid t, l)P(d \mid a, b) \]
\[ \Rightarrow f_s(b, l)f_x(a)f_t(a, l)P(d \mid a, b) \]
\[ \Rightarrow f_l(a, b)f_x(a)P(d \mid a, b) \Rightarrow f_a(b, d) \Rightarrow f_b(d') \]

Complexity: \( O(|V| \cdot n^{k+1}) \) instead of \( O(n^{|V|}) \), where \( k \) is the maximum in-degree of a node in the graph (here \( k=2 \)).
Variable elimination

- Can be exponential for arbitrary graph;
- **Hard to determine the order of variables to be eliminated**
  - Find the optimal order is NP-hard
- In practice, it may be quite efficient on sparse graph
- However, hard for inference problems in bioinformatics.
Approximate inference: sampling

- Suppose you are given values for some subset of the variables (evidences), E, and want to infer values for unknown variables, Z

- Randomly sample a very large number of instances from BN
  - Generate instances for all variables – start at root variables and move “forward” in a “topological order” of the nodes
    - topological ordering of a directed graph is a linear ordering of its vertices such that for every directed edge $uv$ from vertex $u$ to vertex $v$, $u$ comes before $v$ in the ordering
    - There always exists a topological order in a DAG.
  - This is much easier to compute than the joint probability

- Reject the instances inconsistent with E
- Use the frequency of values for Z in the retained instances to get estimated probabilities
- Accuracy of the results depends on the size of the sample (asymptotically approaches the exact results)
Likelihood weighting

- Idea: do not sample instances that need to be rejected
  - Sample only from the unknown variables $Z$
  - weight each sample according to the likelihood that it would occur, given the evidence $E$
  - Markov Chain Monte Carlo (MCMC) algorithm
MCMC algorithm

- A random walk through variable space, counting instances during sampling
  - Initialize with a random instance, consistent with evidence variables E
  - At each step, for a non-evidence variable, randomly sample its value, based on the other current assigned variables
- When samples approach infinite, MCMC reaches an accurate estimate of the actual joint distribution
Predicting diseases from BN

- Target: Gene1, Gene2
- Evidence: Protein1
- BMI: body mass index
Learning Bayesian networks: four cases

- Known graph—learn parameters
  - Complete data (ML, MAP)
  - Incomplete data (EM)

- Unknown graph—learn graph and parameters
  - Complete data; optimization problem (search in space of graphs)
  - Incomplete data; structural EM
Learning parameters: complete data

The **maximum likelihood estimate (MLE)** of $\theta_{ij}$ can be computed by a frequency model,

$$\theta_{ijk} = \frac{N_{ijk}}{\sum_k N_{ijk}}$$

where $\theta_{ijk}$ is the number of cases in D in which $X_i = x_i^k$ and $P_{a_i} = p_{a_i}^j$. If we assume the prior distribution of $\theta_{ij}$ follow a Dirichlet distribution with parameters $\alpha_{ij} = (\alpha_{ij1}, \ldots, \alpha_{ijr_i})$, i.e., the pseudo-counts, we have the MAP estimates,

$$\theta_{ijk} = \frac{\alpha_{ijk} + N_{ijk}}{\sum_k (\alpha_{ijk} + N_{ijk})}$$

Assumption: the **parameters are independent** (i.e., $\theta_{ij}$ are mutually independent)
An example

Adding pseudo-count 1 for each case

P(j=yes|f=yes,a=*,s=*)=0.6,
P(j=yes|f=no,a=<30,s=male)=0.5
P(j=yes|f=no,a=30-50,s=male)=0.25
P(j=yes|f=no,a=>50,s=male)=0.33
P(j=yes|f=no,a=<30,s=female)=0.50
P(j=yes|f=no,a=30-50,s=female)=0.67
P(j=yes|f=no,a=>50,s=female)=0.33

Sample size is too small!
Learning parameters with missing data

- Important property of the missing data
  - the absence of the data is dependent on the actual state of the variable
    - e.g., a missing datum in a drug study may indicate that a patient became too sick, perhaps due to the side effects of the drug, to continue in the study.
  - the absence of the data and the state of the variable are independent

- BN can handle both situations; the 2nd one is simpler and will be discussed here.
Learning parameters: missing data

- Gibbs sampling (MCMC) algorithm
  - Randomly choose an initial state for each of the variables without observations, forming the initial configuration
  - Pick a random variable $x_i$, compute its probability distribution *given the states of the other n-1 variables*
  - Sample a state of variable $x_i$, forming a new configuration
  - Iterate the two previous steps, and record all visited configurations
  - Compute the MLE parameters involving the variables with missing data
Learning parameters: missing data

- EM algorithm: finding a local ML
  - Randomly assign parameters to the distribution involving the variables without observations
  - E-step: using BN inference algorithm to obtain the probability distribution of these variables, given the entire network
  - M-step: update model parameters by using MLE based on the frequencies derived from E-step
  - Iterate between E and M steps until the model converges
Learning graph structure

- Constrained-based structure learning algorithms (dependence analysis and search)
  - Independence test: \( P(X,Y) = P(X) \cdot P(Y) \)

- Structure scoring methods (optimization of a scoring function) (scoring and search)

Find \( \hat{G} = \arg \max_G \text{Score}(G) \)

- Hybrid methods
  - Constraint-based methods can be more efficient for large samples; the detection of conditional independencies may be sensitive; and may not assign a direction to every edge
  - Score-based approach is generally preferred, esp when dealing with small sample size and noisy data.
Constraint-based methods

- Constraint-based methods focus on identifying conditional independence relationships (i.e., Markov conditions) between variables using observed data; conditional independencies are used to constrain the underlying network structure.

- Typically, hypothesis testing procedures, such as the chi-square test and mutual information test, are first used to remove edges from a fully connected undirected graph based on findings of unconditional independence.

- Then directions are added to edges between nodes according to the d-separation (directed separation) criteria.
Grow-shrink method

- Based on the concept of Markov blanket
  - The Markov blanket of a node in a BN consists of its parents, children, and its children’s other parents.

- The GS algorithm
  - Starts with a variable X and an empty set S. The growing phase adds variables to S if they are dependent on X, conditional on the variables currently in S. In the shrinking phase, variables that are rendered independent of X, based on the current members of S, are then removed from S.
  - Represent S (together with X) as a fully connected, undirected network.
  - Examining triples of variables using the d-separation criteria (e.g., remove spousal links between two nodes Y and Z by looking for a d-separating set around Y and Z, and give directions to edges if conditioning on a middle node creates a dependency.)
D-separation criteria

- If X, Y and Z are three disjoint sets of nodes in a BN, then Y is said to d-separate X from Z if and only if Y blocks every path from a node in X to a node in Z.

- **Cancer (C), environmental exposure (E), a biomarker (B) and three SNPs (S1, S2, and S3)**
  - E d-separates both B and C from S3
  - S1 d-separates C from S2 (all serial paths)
  - E also d-separates nodes B and C from each other (as part of a divergent path)
  - C does NOT d-separate E from S1 because convergent paths are not blocked

- **Types of paths**
  - **Serial**
  - **Converging**
  - **Diverging**
Score-based methods

Find \( \hat{G} = \arg \max_{G} \text{Score}(G) \)

Score(G) – measures how well a model fits the data
Finding the best model is NP-hard optimization; Use heuristic search algorithms instead

- Scoring function
- Search space
- Search strategy
Scoring functions

- Likelihood scores
  \[
  \max_{(G, \theta_G)} L(\langle G, \theta_G \rangle \mid D) = \max_S \left( \max_{\theta_S} L(\langle S, \theta_S \rangle \mid D) \right) = \max_S L(\langle S, \hat{\theta}_S \rangle \mid D)
  \]

- Penalized log-likelihood scores
  - BIC Bayesian information criteria
  - MDL Minimum description length

- Bayesian scores
  \[
  \max_G P(G \mid D) \propto \max_S P(D \mid S) P(S) = \int P(D \mid \theta_S, S) P(\theta_S \mid S) d\theta_S
  \]
  - Prior probability used for \( P(S) \) and \( P(\theta_S \mid S) \)
Search methods

- Most search methods make successive changes of edge linkages to the network, and employ the local criterion to assess the merit of each change.

- One simple heuristic search algorithm is greedy search
  - may be stuck at local minima; can start from multiple initial points
  - global optimization approaches can apply: simulated annealing, best-first search, etc
Bayesian networks as classifiers

- Two types of nodes: a class node (C) and attribute nodes
- A BN can be used as a classifier that gives the posterior probability distribution of the class node, given attributes X.

\[
P(C|x, G) = \frac{P(C, x|G)}{P(x|G)} \propto P(C, x|G)
\]

\[c^* = \arg \max_j P(c_j, x|G)\]

- A NB (Naïve Bayes) classifier can be viewed as a BN classifier with a simple structure
Model selection trade-offs

Naïve Bayes – too simple
(less parameters, but bad model)

Unrestricted BN – too complex
(possible overfitting + complexity)

Various approximations between the two extremes

TAN:
tree-augmented Naïve Bayes
[Friedman et al. 1997]

Based on Chow-Liu Tree Method
(CL) for learning trees
[Chow-Liu, 1968]

Tree-approximation

True distribution $P(X)$

```
A
  / \
 /   \
C-----B
  |   |
  |   |
D-----E
```

Tree-approximation $P'(X)$

```
A
  / \
 /   \
C-----B
  |   |
  |   |
D-----E
```
Extensions to NB classifier

TAN model: A network with an edge between the class node and each of the attributes (to ensure that all attributes are part of the class variable Markov blanket)

- Dashed lines, edges required by NB classifier
- Solid lines, correlation edges between attributes (relax the independence assumption between the attributes)

Ref: Friedman et al, 1997
CL algorithm for constructing a tree BN from data

1. Compute $I_{\hat{p}_D}(X_i; X_j)$ between each pair of variables, $i \neq j$, where

$$I_P(X; Y) = \sum_{x,y} P(x,y) \log \frac{P(x,y)}{P(x)P(y)}$$

is the \textit{mutual information} function. Roughly speaking, this function measures how much information $Y$ provides about $X$. See Appendix A for a more detailed description of this function.

2. Build a complete undirected graph in which the vertices are the variables in $X$. Annotate the weight of an edge connecting $X_i$ to $X_j$ by $I_{\hat{p}_D}(X_i; X_j)$.

3. Build a maximum weighted spanning tree.

4. Transform the resulting undirected tree to a directed one by choosing a root variable and setting the direction of all edges to be outward from it.

CL prove that this procedure finds the tree that maximizes the likelihood given the data $D$.

Ref: Chou and Liu, 1968
Learning of causal relationships

- **Causal Markov condition**: 
  - Variable $a$ is a *direct* cause of variable $b$ if and only if there is a direct edge from $a$ to $b$; then the BN is called a *causal graph*.
  - Variable $a$ is a cause of variable $b$ (or $b$ is dependent on $a$) if there exists a *d-connecting* path from $a$ to $b$ given *evidence* $E$ (a set of variables)
    - A path from $a$ to $b$ is *d-connecting* if each interior node $n$ in the path is either
      - Linear or diverging and not a member of $E$; or
      - Converging, and either $n$ or one of its descendants is in $E$. 
**d-separation path**

**Serial**
- a is the cause of b; but 
a is not the cause of b, given c as evidence

**Converging**
- a is not the cause of b; but 
a is the cause of b, given c as evidence

**Diverging**
- b is the cause of a; 
b may not be the cause of a, if c is given as evidence
Dynamic Bayesian network (DBN)

(Vehicle localization task) A moving car tried to track its current location using the data obtained from a, possibly faulty sensor. The system state can be encoded (very simply) using the: Location – the car’s current location, Velocity – the car’s current velocity; Weather – the current weather; Failure – the failure status of the sensor; and Obs – the current observation. We have one such set of variables for every time point t. A joint probability distribution over all of these sets defines a probability distribution over trajectories of the car. Using this distribution, we want to ask a variety of queries, such as 1) given a sequence of observations about the car, where is it now? 2) where is it likely to be in 10 minutes? 3) did it stop at the red light?
DBN for monitoring a car
a 2-time-slice DBN (2-DBN)

Assumptions: 1) the sensor observation is generated at each time point independently given other variables; 2) all variables are interface variables except for obs.
Reasoning if the model is given

- Given a sequence of observations about the car, where is it now?
  - $P(\text{obs}_t|\text{obs}_0, \ldots, \text{obs}_{t-2}, \text{obs}_{t-1})$

- Where is it likely to be in 10 minutes?
  - $P(\text{obs}_{t+10}|\text{obs}_0, \ldots, \text{obs}_{t-2}, \text{obs}_{t-1})$

- Did it stop at the red light?
  - $P(V_t|\text{obs}_0, \ldots, \text{obs}_{t-2}, \text{obs}_{t-1})$
HMM as a 2-DBN

DBN is more general than HMM: 1) the CPD of hidden states can be modeled by a BN, rather than a simple Markov chain; 2) more than one observation variable can be modeled simultaneously (like multivariate HMM).
R packages

- mlbench
  - machine learning benchmark problems
  - E.g., pima data: PimaIndiansDiabetes2

- bnlearn - an R package for Bayesian network learning and inference (http://www.bnlearn.com/)
Applications of BN

Sparse candidate algorithm:
Only consider a small number of parent candidates for each gene to constrain the search space of the network

Local probability models:
Multinomial model
Linear Gaussian model

General architecture of the magic Bayesian Network.

The system (MAGIC) formally incorporates expert knowledge about relative accuracies of data sources to combine them within a normative framework.

Conditional probability tables for each connection were assessed formally from yeast genetics expert.

Troyanskaya O G et al. PNAS 2003;100:8348-8353