Bayesian Networks

Machine Learning 10-601B

Seyoung Kim

Many of these slides are derived from Tom Mitchell. Thanks!

Learning of Bayes Nets

- Four categories of learning problems
  - Graph structure may be known/unknown
  - Variable values may be fully observed/partially unobserved

- Easy case: learn parameters, when graph structure is *known*, and data is *fully observed*

- Interesting case: graph *known*, data *partially known*

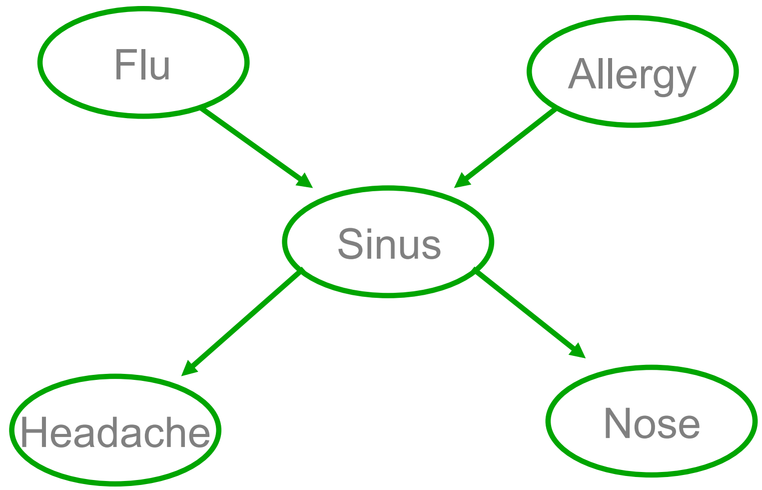
- Gruesome case: graph structure *unknown*, data *partially unobserved*

LEARNING BAYESIAN NETWORK PARAMETERS WITH KNOWN STRUCTURE

Learning CPTs from Fully Observed Data

- Example: Consider learning the parameter

$$\theta_{s|ij} \equiv P(S=1|F=i, A=j)$$



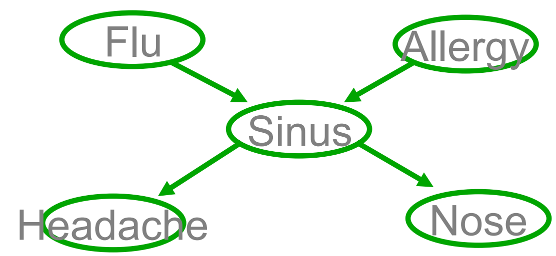
- MLE (Max Likelihood Estimate) is

$$\theta_{s|ij} = \frac{\sum_{k=1}^{K} \delta(f_k = i, a_k = j, s_k = 1)}{\sum_{k=1}^{K} \delta(f_k = i, a_k = j)}$$

k^{th} training example

- Remember why?

MLE estimate of $\theta_{s|ij}$ from fully observed data



- Maximum likelihood estimate

$$\theta \leftarrow \arg\max_{\theta} \log P(data|\theta)$$

- Our case:

$$P(data|\theta) = \prod_{k=1}^{K} P(f_k, a_k, s_k, h_k, n_k)$$

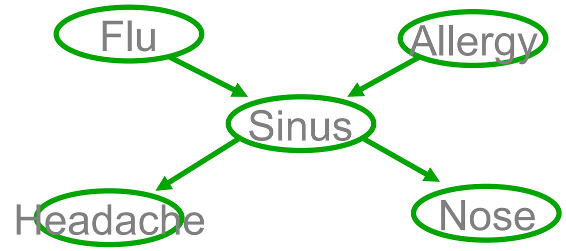
$$P(data|\theta) = \prod_{k=1}^{K} P(f_k)P(a_k)P(s_k|f_k a_k)P(h_k|s_k)P(n_k|s_k)$$

$$\log P(data|\theta) = \sum_{k=1}^{K} \log P(f_k) + \log P(a_k) + \log P(s_k|f_k a_k) + \log P(h_k|s_k) + \log P(n_k|s_k)$$

$$\frac{\partial \log P(data|\theta)}{\partial \theta_{s|ij}} = \sum_{k=1}^{K} \frac{\partial \log P(s_k|f_k a_k)}{\partial \theta_{s|ij}}$$

$$\theta_{s|ij} = \frac{\sum_{k=1}^{K} \delta(f_k = i, a_k = j, s_k = 1)}{\sum_{k=1}^{K} \delta(f_k = i, a_k = j)}$$

Estimate θ from partially observed data



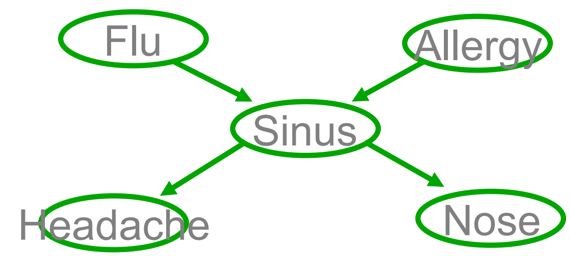
- What if FAHN observed, but not S?
- Can't calculate MLE

$$\theta \leftarrow \arg\max_{\theta} \log \prod_k P(f_k, a_k, s_k, h_k, n_k | \theta)$$

WHAT TO DO?

Estimate θ from partially observed data

- Let X be all observed variable values (over all samples)
- Let Z be all unobserved variable values
- Can't calculate MLE:



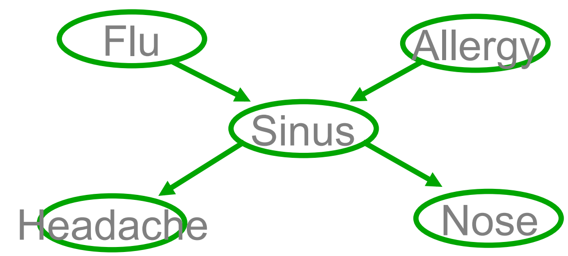
$$\theta \leftarrow \arg\max_{\theta} \log P(X, Z|\theta)$$

- EM seeks to estimate:

$$\theta \leftarrow \arg\max_{\theta} E_{Z|X,\theta}[\log P(X, Z|\theta)]$$

  – EM guaranteed to find local maximum

EM with Partially Observed Data



- EM seeks to estimate:

$$\theta \leftarrow \arg\max_{\theta} E_{Z|X,\theta}[\log P(X,Z|\theta)]$$

- here, observed X={F,A,H,N}, unobserved Z={S}, K samples

$$\log P(X,Z|\theta) = \sum_{k=1}^{K}[\log P(f_k)+\log P(a_k)+\log P(s_k|f_k a_k)+\log P(h_k|s_k)+\log P(n_k|s_k)]$$

$$E_{P(Z|X,\theta)} \log P(X,Z|\theta) = \sum_{k=1}^{K}\sum_{i=0}^{1} P(s_k=i|f_k,a_k,h_k,n_k)$$
$$[log P(f_k)+\log P(a_k)+\log P(s_k|f_k a_k)+\log P(h_k|s_k)+\log P(n_k|s_k)]$$

EM Algorithm

- EM is a general procedure for learning from partially observed data
- Given observed variables X, unobserved Z (X={F,A,H,N}, Z={S}), define

$$Q(\theta'|\theta) = E_{P(Z|X,\theta)}[\log P(X, Z|\theta')]$$

Iterate until convergence:

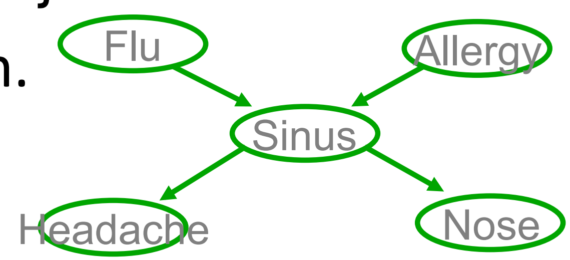
- E Step: Use X and current θ to calculate $P(Z|X,\theta)$
- M Step: Replace current θ by

$$\theta \leftarrow \arg\max_{\theta'} Q(\theta'|\theta)$$

- Guaranteed to find local maximum.
- Each iteration increases $E_{P(Z|X,\theta)}[\log P(X, Z|\theta')]$

E Step: Use X, θ, to Calculate P(Z|X,θ)

- observed X={F,A,H,N}, unobserved Z={S}
- How? Bayesian network inference problem.



$$P(S_k = 1 | f_k a_k h_k n_k, \theta) =$$

$$\frac{P(S_k = 1, f_k, a_k, h_k, n_k)}{P(f_k, a_k, h_k, n_k)} = \frac{P(S_k = 1, f_k, a_k, h_k, n_k)}{P(S_k = 1, f_k, a_k, h_k, n_k) + P(S_k = 0, f_k, a_k, h_k, n_k)}$$

$$P(S_k = 1 | f_k a_k h_k n_k, \theta) = \frac{P(S_k = 1, f_k a_k h_k n_k | \theta)}{P(S_k = 1, f_k a_k h_k n_k | \theta) + P(S_k = 0, f_k a_k h_k n_k | \theta)}$$

EM and estimating $\theta_{s|ij}$

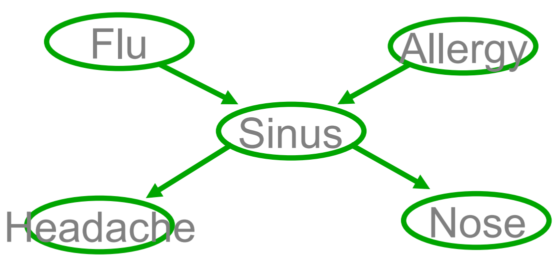
- observed X = {F,A,H,N}, unobserved Z={S}

E step: Calculate $P(Z_k|X_k;\theta)$ for each training example, k

$$P(S_k = 1|f_k a_k h_k n_k, \theta) = \underbrace{E[s_k]}_{P(Z|X;\theta)} = \frac{P(S_k = 1, f_k a_k h_k n_k|\theta)}{P(S_k = 1, f_k a_k h_k n_k|\theta) + P(S_k = 0, f_k a_k h_k n_k|\theta)}$$

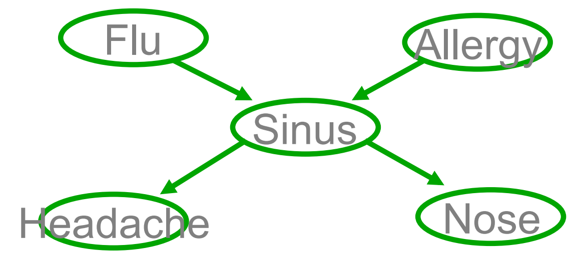
M step: update all relevant parameters. For example:

$$\theta_{s|ij} \leftarrow \frac{\sum_{k=1}^{K} \delta(f_k = i, a_k = j) \;\; E[s_k]}{\sum_{k=1}^{K} \delta(f_k = i, a_k = j)}$$



Recall MLE was: $\theta_{s|ij} = \dfrac{\sum_{k=1}^{K} \delta(f_k = i, a_k = j, s_k = 1)}{\sum_{k=1}^{K} \delta(f_k = i, a_k = j)}$

EM and estimating θ



- More generally, given observed set X, unobserved set Z of boolean values

E step: Calculate for each training example

the expected value of each unobserved variable
inference algorithm!

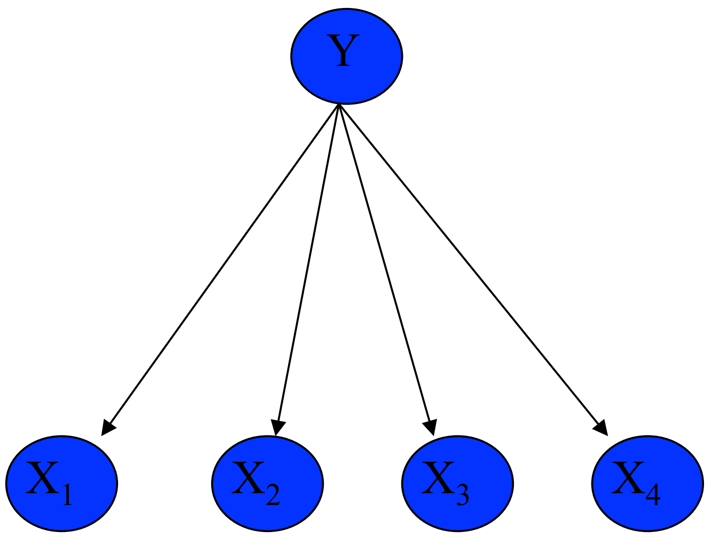
M step:

Calculate estimates similar to MLE, but replacing each count by its <u>expected count</u>

$$\delta(Y=1) \rightarrow E_{Z|X,\theta}[Y] \qquad \delta(Y=0) \rightarrow (1 - E_{Z|X,\theta}[Y])$$

Using Unlabeled Data to Help Train Naïve Bayes Classifier

Learn P(Y|X)



Y	X1	X2	X3	X4
1	0	0	1	1
0	0	1	0	0
0	0	0	1	0
?	0	1	1	0
?	0	1	0	1

LEARNING BAYESIAN NETWORK STRUCTURE

Learning Bayesian Network Structure

- Learning a Bayesian network structure: open problem in general!
  - can require lots of data (else high risk of overfitting)
  - Can constrain the search space to improve computational efficiency

Learning Bayesian Network Structure

- Learning a Bayesian network structure
  - Tree structure
    - Restrictive model structure
    - Efficient learning and inference algorithms
  - A general directed acyclic graph structure
    - Very large search space of candidate BN structures
    - Inexact method: heuristic search, efficient
    - Exact method: dynamic programming, exponential time complexity

Learning a Tree-structured Bayesian Network

- A key result: Chow-Liu algorithm finds “best” tree-structured network
  - suppose P(X) is true distribution, T(X) is our tree-structured network, where X = <X1, … Xn>
  - Chow-Liu minimizes Kullback-Leibler divergence:

$$KL(P(\mathbf{X}) \,||\, T(\mathbf{X})) \equiv \sum_k P(\mathbf{X}=k) \log \frac{P(\mathbf{X}=k)}{T(\mathbf{X}=k)}$$

Chow-Liu Algorithm

- Key result: To minimize KL(P || T), it suffices to find the tree network T that maximizes the sum of mutual information over its edges

- Mutual information for an edge between variable A and B:

$$I(A,B) = \sum_a \sum_b P(a,b) \log \frac{P(a,b)}{P(a)P(b)}$$

- This works because for tree networks with nodes $\mathbf{X} \equiv \langle X_1 \ldots X_n \rangle$

$$\begin{aligned} KL(P(\mathbf{X}) \,||\, T(\mathbf{X})) &\equiv \sum_k P(\mathbf{X}=k) \log \frac{P(\mathbf{X}=k)}{T(\mathbf{X}=k)} \\ &= -\sum_i I(X_i, Pa(X_i)) + \sum_i H(X_i) - H(X_1 \ldots X_n) \end{aligned}$$

Chow-Liu Algorithm

Step 1: For each pair of variables A,B, use data to estimate P(A,B), P(A), P(B)

Step 2: For each pair of variables A,B, calculate mutual information

$$I(A,B) = \sum_a \sum_b P(a,b) \log \frac{P(a,b)}{P(a)P(b)}$$

Step 3: Calculate the maximum spanning tree over the set of variables, using edge weights I(A,B) (given N variables, this costs only $O(N^2)$ time)

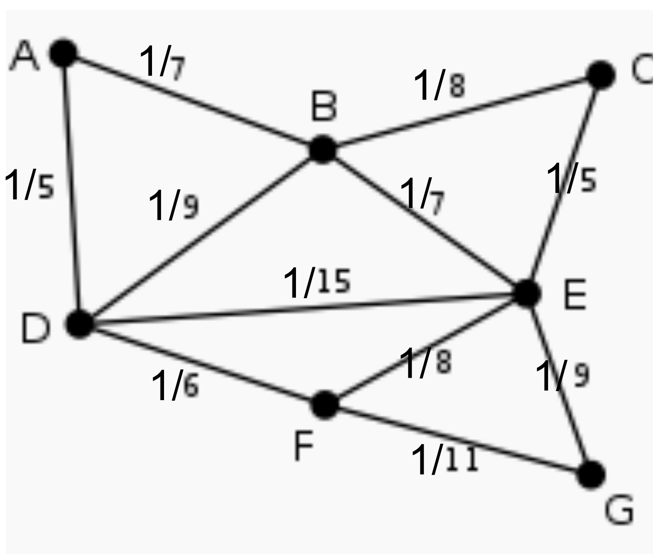
Step 4: Add arrows to edges to form a directed-acyclic graph by picking an arbitrary node as root and directing edges outward from the root

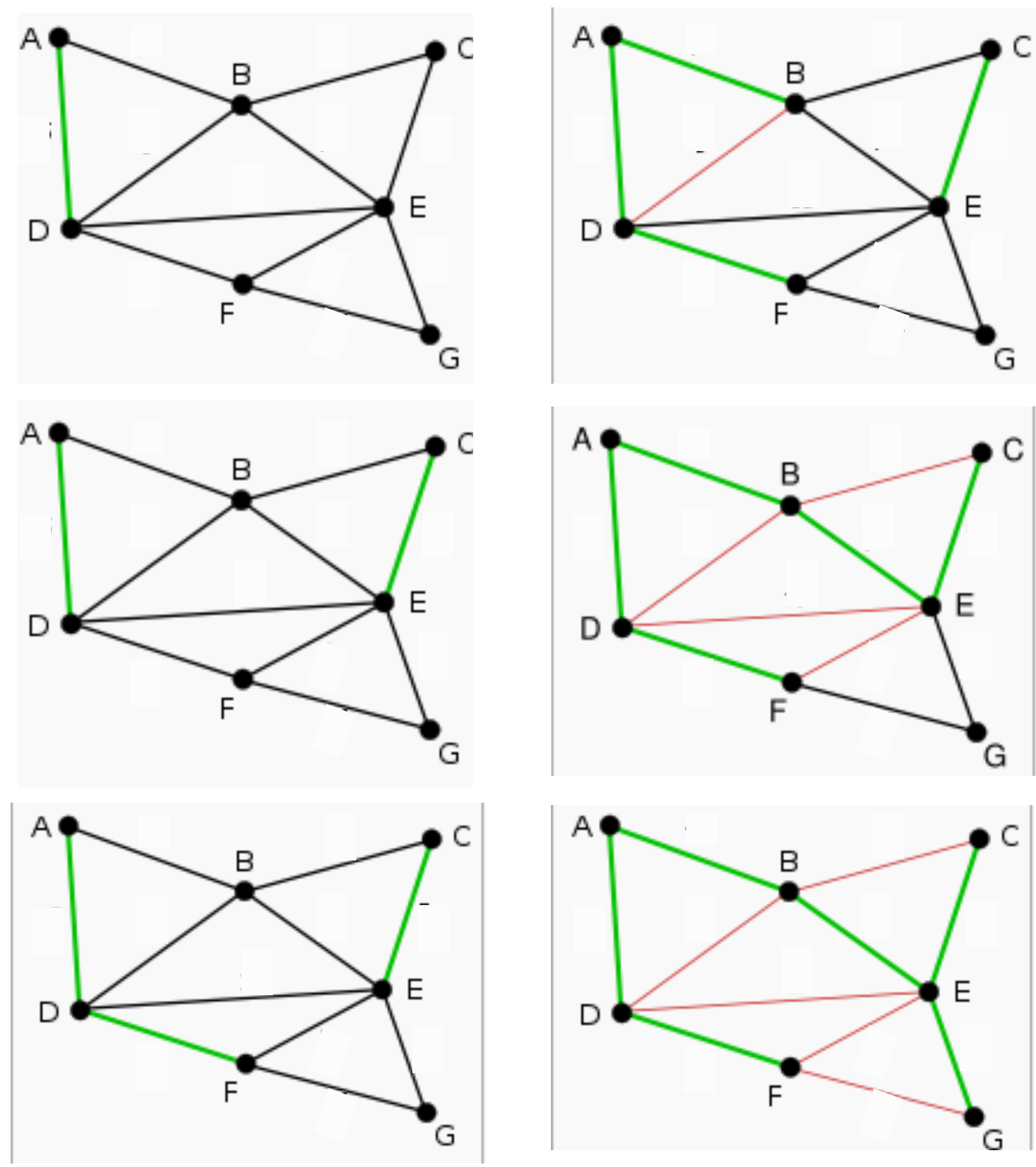
Step 5: Learn the CPD's for this graph

Maximum Spanning Tree Algorithm

- Kruskal's Algorithm
  - Start with the empty graph and add edges one by one
  - As the next edge to add, choose one that
    - Is not in graph yet
    - Does not introduce a cycle. Has the maximum weight

Chow-Liu algorithm example
Greedy Algorithm to find Max-Spanning Tree





[courtesy A. Singh, C. Guestrin]

General Bayesian Network Structure Learning

- A naïve approach: exhaustive search
  - Compute the score of every structure and pick the one with the highest score
  - Exponentially large search space
  - Maintaining DAG constraint is challenging
- Heuristic search

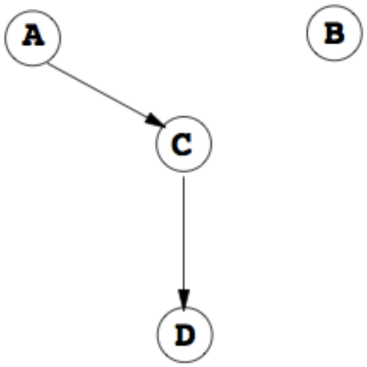
Hill Climbing Algorithm

- Start with an initial structure
- Repeat until termination:
  - Generate a set of structures by modifying the current structure.
  - Compute their scores.
  - Pick the one with the highest score and use it as the current model in the next step.
  - Terminate when model score cannot be improved.
- Return the best network.

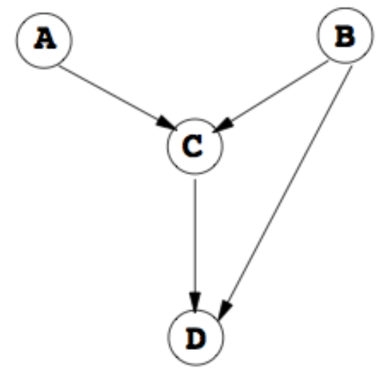
Search Operators

- Search operators for modifying a structure:
  - Add an arc
  - Delete an arc
  - Reverse an arc
- Note:
  - The add-arc and reverse-arc not permitted if results in directed cycles

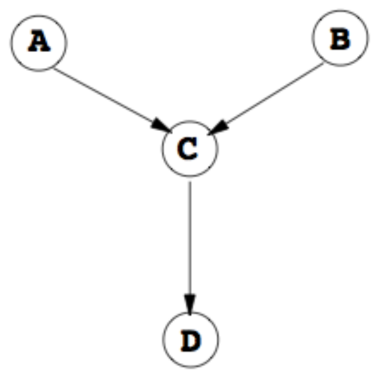
Search Operators

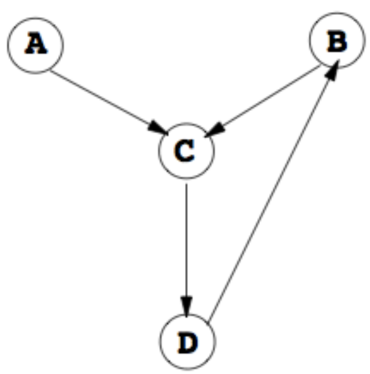


Delete B->C

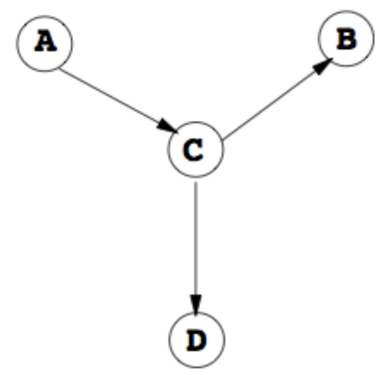


Add B->D





Add D->B, illegal



Reverse B->C

Evaluating Candidate Models

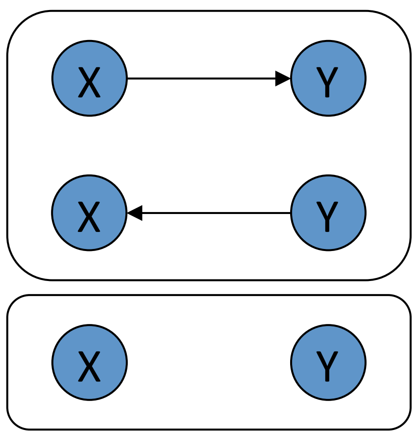
- Suppose there are n variables
- The number of candidate models at each iteration: $O(n^2)$
- We need to compute the score of each of the candidate models
  - This is the most time-consuming step
  - Structures of scoring functions can be exploited to simplify the computation

Problems with Hill Climbing

- Local maxima:
  All one-edge changes reduced the score, but not optimal yet
- Plateaus:
  Neighbors have the same score
- Solutions:
  - Random restart
  - TABU-search:
    - Keep a list of K most recently visited structures and avoid them
    - Avoid plateau
  - Simulated annealing

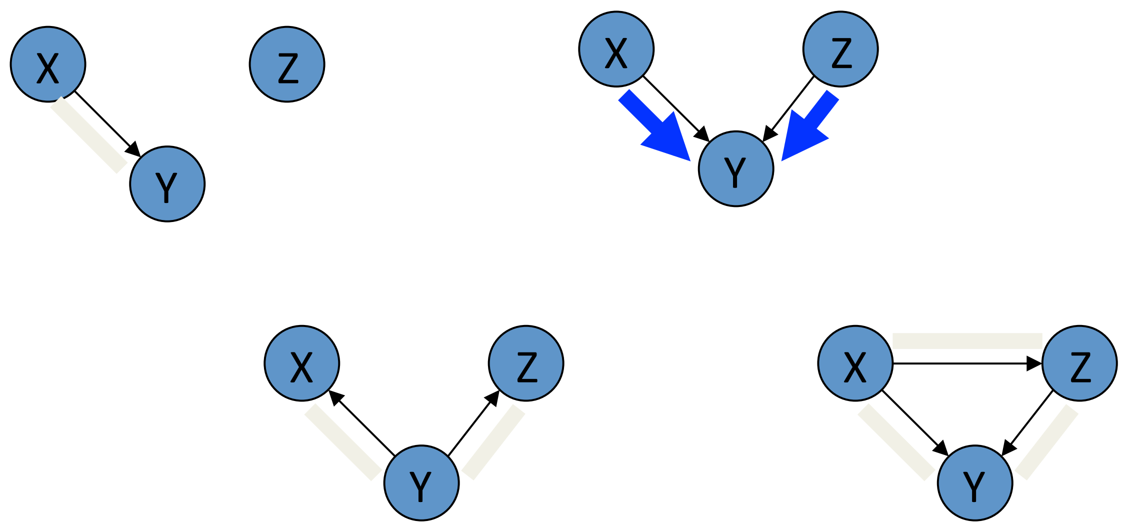
Equivalence Classes

- Bayesian networks with network structures in the same equivalence class are not distinguishable
- Two network structures are equivalent if
  - They have the same skeleton, ignoring edge directions
  - They have the same set of collider nodes



Theorem 1 (Verma & Pearl 1990)

- Two DAGs are equivalent if and only if they have the same skeletons and the same v-structures



Overfitting and Structure Learning

- As the network has more edges, the complexity of model increases and the model is more likely to overfit training data

- How to fight overfitting?
  - Assume a simpler network structure e.g., tree
  - Cross validation
  - Minimum description length

Cross Validation

- Holdout validation:
  - Split data into training set and validation set
  - Parameter estimation based on training set
  - Model score: likelihood based on validation set

- Cross validation:
  - Split data into k subsets
  - Use each subset as validation set and the rest as training set, and obtains a score
  - Total model score: average of the scores for all the cases

Minimum Description Length

- Machine learning is about finding regularities in data
- Regularities should allow us to describe the data concisely
- Find model to minimize

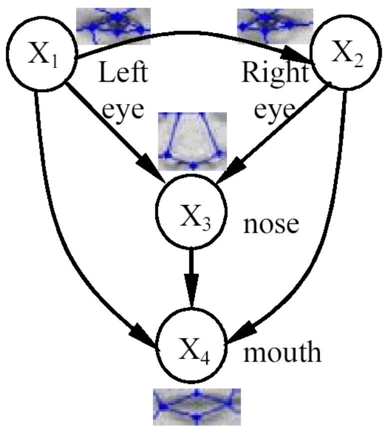
Description length of model + Description length of data

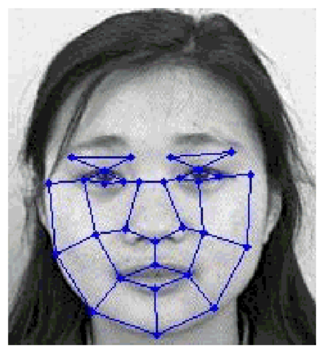
$$\frac{d}{2}\log N$$

Negative data log likelihood

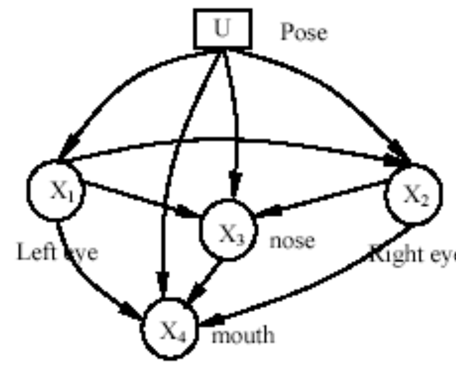
Face Modeling/Recognition Using Bayesian Networks

Face feature finder (separate)

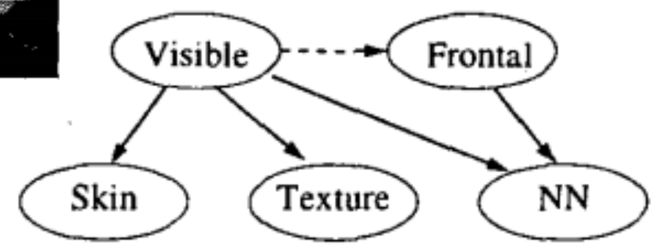




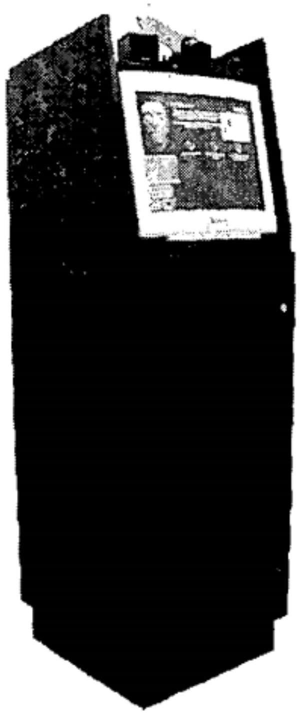
Add Pose switching variable

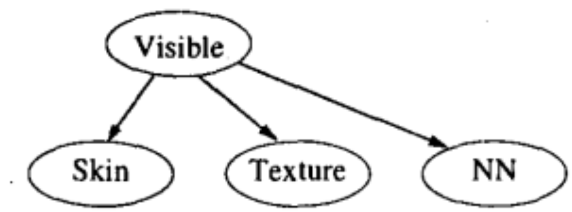


Speaker Detection with Bayesian Networks



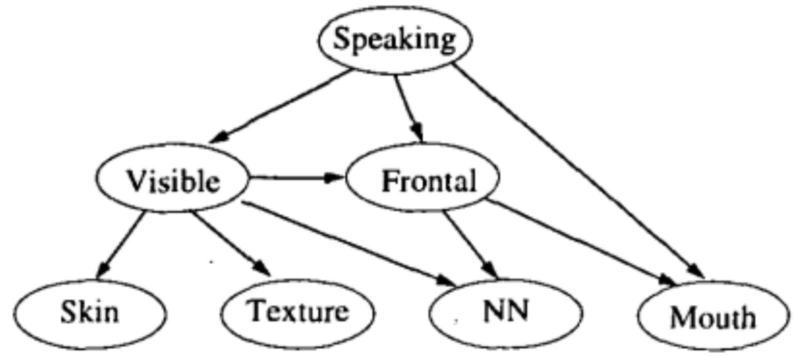
Modeling pose with polytree structure





Naïve Bayes

- Visible: Is there a speaker?
- Frontal: 0 for non-frontal, 1 for frontal faces
- Speaking: Is the person speaking?
- Skin, Texture, NN: face features



Full speaker-detection system

Rehg, et al. 1999

Bayes Nets – What You Should Know

- Representation
  - Bayes nets represent joint distribution as a DAG + Conditional Distributions
  - D-separation lets us decode conditional independence assumptions
- Inference
  - NP-hard in general
  - For some graphs, closed form inference is feasible
  - Variable elimination, stochastic methods
- Learning
  - Easy for known graph, fully observed data (MLE's, MAP est.)
  - EM for partly observed data, known graph
  - Learning graph structure: Chow-Liu for tree-structured networks
  - Hardest when graph unknown, data incompletely observed