Learning structure in Bayes Nets: Scoring functions

- Maximum likelihood scoring
- Minimum description length
- Bayesian scoring

Scoring networks

- Recall from last time: we will do a search over the space of DAGs, then fit parameters on top of the structure
- For the search, we need to assign a *score* (value, goodness) to each network
- The search process requires scoring *many networks*!
- But the application of an operator (add, delete, reverse arc) only changes the local structure of the network
- We need scoring metrics that can be decomposed into scores for each family
- Then we can compute a *change in score* easily

Assumptions

- We are looking for a Bayes net over n random variables
- We have a data set D of i.i.d. samples
- Let $m = |D|$ be the size of the data set
- Each sample has the form: $\mathbf{x_j} = \langle x_{j1}, \dots x_{jn} \rangle$ where x_{ji} is the value of variable X_i in the jth sample
- We assume complete data (all values are known in all samples)

Maximum likelihood scoring

- Recall: the *likelihood function* measures the likelihood of the data given a model. We used this for parameter learning, assuming a *given* structure G:

$$L(\theta, G|D) = p(D|\theta, G) = \prod_{j=1}^{m} p(\mathbf{x_j}|\theta) = \prod_{j=1}^{m} \prod_{i=1}^{n} p(x_{ji}|\text{Parents}(x_{ji}))$$

- We know how to compute the parameters θ (the CPTs) that maximize the likelihood using counts
- The **maximum likelihood score** for a structure G is defined as the likelihood given the **best** parameter setting for that structure:

$$score_L(G) = \log L(\theta, G|D)$$

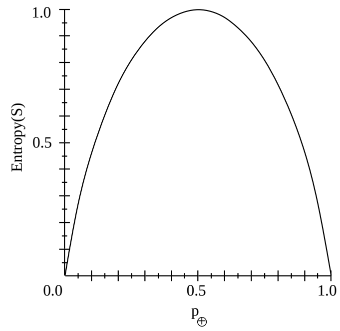
- Does this have an interpretation?

Entropy

- The entropy of a random variable $\mathbf{X}$ drawn from a distribution p is:

$$H_p(\mathbf{X}) = -\sum_{\mathbf{x}} p(\mathbf{x}) \log p(\mathbf{x})$$

- This is trivially extended if $\mathbf{X}$ is a set of random variables and p is their joint distribution.
- Entropy measures the amount of randomness in the distribution. Equivalently, it measures the amount of information.



Entropy and information theory

- Suppose I will get data $\mathbf{x}_j$ and I want to send it over a channel. I know that the probability of item $\mathbf{x}_j$ is p_j.
- Suppose there are 4 possible values, and all are equally likely. Then I can encode them in two bits each, so on every transmission I need 2 bits
- Suppose now $p_0 = 0.5$, $p_1 = 0.25$, $p_2 = p_3 = 0.125$. Can I get a better encoding? What is the expected length of the message that I will have to send over time?

More on information theory

- Suppose I believe the messages are generated according to distribution Q, but really they come from P
- Then my best encoding will take an expected $-\sum_j p_j \log q_j$ bits
- The difference in the number of bits is:

$$-\sum_j p_j \log q_j - \left(-\sum_j p_j \log p_j\right) = \sum_j p_j \log \frac{p_j}{q_j}$$

This is our old friend the *KL distance*! (also called **relative entropy**)

Mutual information

- For two sets of random variables $\mathbf{Y}$ and $\mathbf{Z}$, the mutual information relative to distribution p is:

$$MI_p(\mathbf{Y}, \mathbf{Z}) = \sum_{\mathbf{y},\mathbf{z}} p(\mathbf{y}, \mathbf{z}) \log \frac{p(\mathbf{y}, \mathbf{z})}{p(\mathbf{y})p(\mathbf{z})}$$

- This is the relative entropy between $p(\mathbf{Y}, \mathbf{Z})$ and $p(\mathbf{Y})p(\mathbf{Z})$.
- $MI_p(\mathbf{Y}, \mathbf{Z})$ measures how much information one variable provides about the other

Properties of mutual information

- $MI_p(\mathbf{Y}, \mathbf{Z}) \geq 0$
- $MI_p(\mathbf{Y}, \mathbf{Z}) = 0 \equiv \mathbf{Y}$ and $\mathbf{Z}$ are independent
- $MI_p(\mathbf{Y}, \mathbf{Z}) = H_p(\mathbf{Y}) \equiv \mathbf{Y}$ is totally predictable given $\mathbf{Z}$

Likelihood score in terms of entropy

- We can show that:

$$L(G|D) = m \sum_{\imath=1}^{n} (MI_{\hat{p}}(X_i, \text{Parents}(X_i)) - H_{\hat{p}}(X_i))$$

  where m is the number of instances, and $\hat{p}$ is the probability distribution generated by the maximum likelihood fit to the parameters of G
- Nice intuitive explanation: the larger the dependency of each variable on its parents, the larger the score.
- **Bad news:** see homework!

Overfitting

General problem for all learning algorithms!

Possible solutions:

- Restricting the hypothesis space

  E.g. restrict the number of parents allowed for any node, or the number of parameters in any CPT
- Minimum description length: prefer compact models over large ones
- Bayesian approach: use prior knowledge to set priors over structures

Minimum description length (MDL) principle

- Suppose we want to transmit data D over a communication channel
- To save space, we want a compact model of D - note that a Bayes net can be viewed as such a model
- We also need enough information to get the exact instances back
- If we know the probability distribution of the data, p, then we can encode the instances based on *universal coding:* most likely instances get the fewest bits (as seen before)

MDL for Bayes nets

We need to encode the graph structure G, the CPTs at each node, and then the instances themselves. We want to minimize the total description length:

- Suppose the graph is encoded in $DL(G)$ bits
- We have $\sum_i$ ParentValues$(X_i)(|X_i| - 1)$ parameters in the CPTs. Each has to be encoded in some number of bits B. The typical choice is $B = \frac{\log m}{2}$
- So transmitting the parameters takes a number of bits:

$$\frac{\log m}{2} \sum_i \text{ParentValues}(X_i)(|X_i| - 1)$$

- For the data, the optimal encoding length is: $-\log p(x_1, \ldots x_m | G, \theta) = -score_L(G, \theta | D)$

MDL score

$$\begin{aligned} score_{MDL} &= score_L(G, \theta | D) - DL(G) - \frac{\log m}{2} \sum_i \text{ParentValues}(X_i)(|X_i| - 1) \\ &= m \sum_i MI_{\hat{p}}(X_i, \text{Parents}(X_i)) - m \sum_i H_{\hat{p}}(X_i) \\ &- \frac{\log M}{2} \sum_i \text{ParentValues}(X_i)(|X_i| - 1) - DL(G) \end{aligned}$$

- The entropy term is the same for any graph, so we can ignore it
- The description length of the graph, $DL(G)$, does not depend on the size of the data set m. So for large m, this can be ignored

More observations

$$score_{MDL} \approx m \sum_i MI_{\hat{p}}(X_i, \text{Parents}(X_i)) - \frac{\log M}{2} \sum_i \text{ParentValues}(X_i)(|X_i| - 1)$$

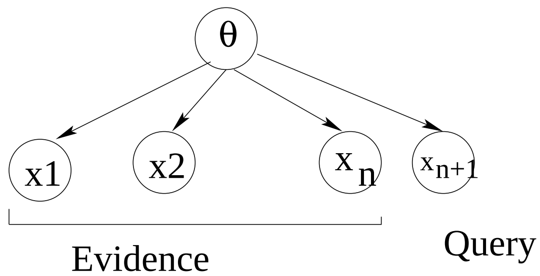
- There is a trade-off between the size of the graph and how well we fit the data:
  - If the graph is large, the score decreases
  - If a variable is highly dependent on its parents, the score increases
- As m grows very large, the emphasis will be on the fit to the data, so asymptotically (as $m \to \infty$), MDL will find the same network as max. likelihood

Consistency of a scoring function

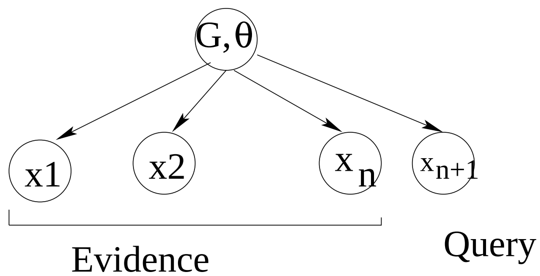
- Suppose that there exists a true model, G^*, which generated the data
- A scoring function is called **consistent** if the following two properties hold with increasing probability, as $m \to \infty$:
  - G^* maximizes the score
  - All structures G that are not equivalent to G^* (in the I-map sense) will have strictly lower score
- Both max. likelihood and MDL are consistent scoring functions

Bayesian approach

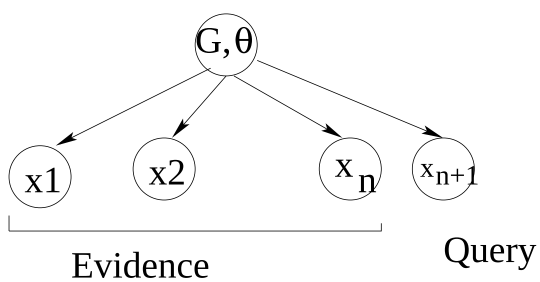
- Main idea: put a distribution over any unknowns!!
- Last time we used this idea to learn parameters of a network



- Now we use it for learning structure:



Bayesian inference for scoring structures



- We compute $p(x_{m+1}|D)$ as an expectation over the unknown structure G (assuming we consider all possible structures):

$$p(x_{m+1}|D) = \sum_G p(x_{m+1}|G, D)p(G|D)$$

- Computing $p(x_{m+1}|G, D)$ is easy - the same as prediction with known structure
- We need to compute $p(G|D)$ (the score of the network).

Bayesian scoring of network structures

- By Bayes rule, we have:

$$p(G|D) = \frac{p(D|G)p(G)}{p(D)}$$

  $p(D)$ is the normalizing factor, same for all structures, so it can be dropped

- So the **Bayesian score** is:

$$score_B(G|D) = \log p(D|G) + \log p(G)$$

  - $p(G)$ is the prior over network structures. It allows control of the complexity of the network (e.g. we can penalize dense nets).
  - $p(D|G)$ is called the **marginal likelihood** of the data given the structure (we marginalize out the parameters)

Marginal likelihood

We compute $p(D|G)$ by marginalizing the network parameters θ:

$$p(D|G) = \int p(D|G,\theta)p(\theta|G)d\theta$$

- $p(D|G,\theta)$ is the likelihood function, $L(G,\theta|D)$.
- $p(\theta|G)$ is the prior over the parameters
- Problem: we need a prior for *all parameters* in the network!

Priors

- Quite often all nets are considered equally likely
- To get the parameter priors, we assume a prior *over the joint* (e.g. the joint is uniform) and an *equivalent sample size*
- Given a network structure, our joint prior factorizes over the network. So we can compute local priors for all parameters!

Bayesian vs. likelihood scoring

- To compute the ML score of a network, we used the "best" parameter setting:

$$\theta^* = \arg\max_{\theta} L(G, \theta | D)$$

- The Bayesian score considers *all possible parameter settings* and computes an expected value of the likelihood over all these settings:

$$p(D|G) = \int p(D|\theta, G) p(\theta|G) d\theta$$

- Intuitively, the integral measures the sensitivity to the choice of parameters

Asymptotic behavior

- If $p(\theta|G)$ is "well-behaved", and we have a reasonable prior, then:

$$\log p(D|G) = score_{MDL} + O(1)$$

So they asymptotically give the same answer.

- Bayesian score is usually less sensitive to noise in the data.