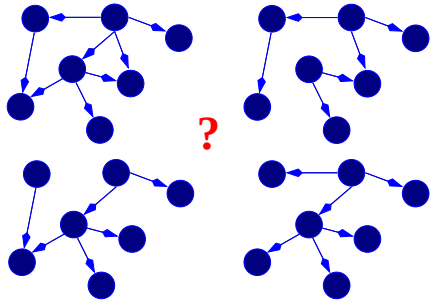
Learning Bayesian Networks

Parameters and Structure

Learning Bayesian networks



- Bayesian networks $\Leftrightarrow$ datasets?
- Generating datasets from Bayesian networks
- Learning:
  - parameter (distribution given structure) learning
  - structure (topology) learning

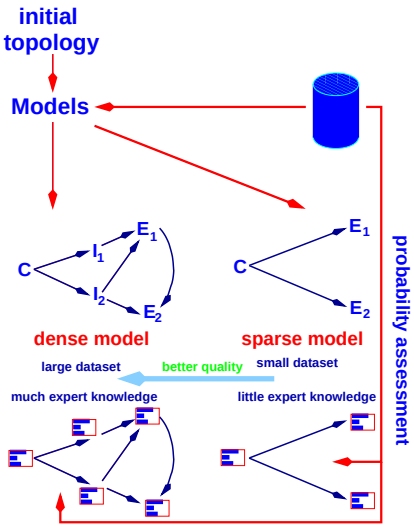
Dataset generation

Let $\mathcal{B} = (G, P)$ be a Bayesian network, with $G = (V(G), A(G))$, and S a sample generator, then a dataset D can be generated as follows:

- visit each vertex $X \in V(G)$ in topological order
- choose a value out of the domain of X, based on:
  - earlier choises of values for $\pi(X)$, denoted by $\hat{\pi}(X)$
  - a random number (generated by a RNG)
  - $P(X \mid \hat{\pi}(X))$

Thus, the generated dataset D reflects the likelihoods in $\mathcal{B}$.

Learning structure and parameters



Learning structure and parameters

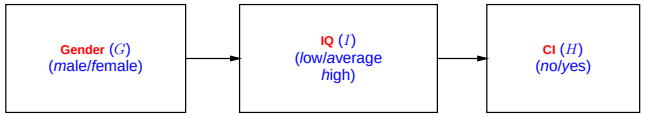
Consider a dataset D consisting of n observations, each of which with N values - one value for each of the variables $\{X_1, \ldots, X_N\}$.

- When S contains no missing values, *complete data*
- When learning BNs from data,
  - Different ADGs can represent the same independence structure and joint probability distribution
  - From learning point of view, those ADGs are equivalent
  - Practical importance: selection of one ADG from a set of equivalent ones

Learning parameters from data

Estimating values of parameters corresponding to an ADG structure and a probability distribution from a database.

Example of the structure of a BN and (tiny) dataset:



Student	Gender	IQ	High mark for CI
1	male	low	no
2	female	average	yes
3	male	high	yes
4	female	high	yes

Learning parameters from data (cont.)

Probability as relative frequency,

$$P(X = x \mid Y = y) = \frac{n_{x \wedge y}}{n_y}$$

where n_x denotes the number of cases in which x holds ($n_\top$: total number of cases).

For the previous example, we obtain:

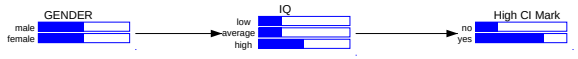
$P(G)$	
m	f
0.5	0.5

	$P(I \mid G)$		
G	l	a	h
m	0.5	0.0	0.5
f	0.0	0.5	0.5

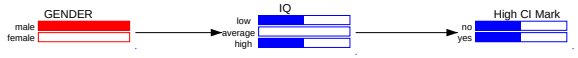
	$P(H \mid I)$	
I	n	y
l	1	0
a	0	1
h	0	1

Resulting Bayesian network

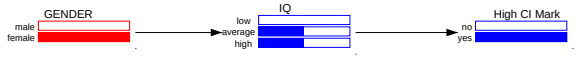
Prior probability:



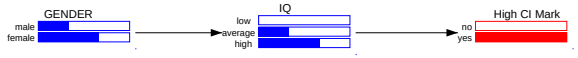
Male student:



Female student:



Good CI student:



Incorporating prior knowledge

Compute the weighted average of

- estimate $\widehat{P}_D(V \mid \pi(V))$ of the conditional probability distribution for variable V based on the dataset D
- Λ reflects prior knowledge (discrete distribution)

These are combined as follows:

$$P(V \mid \pi(V), D) = \frac{n}{n+n_0}\widehat{P}_D(V \mid \pi(V)) + \frac{n_0}{n+n_0}\Lambda$$

where

- n is the size of the dataset D
- n_0 is the estimated size of the (virtual) 'dataset' on which the prior knowledge is based

Example of prior knowledge

Professor S has prior knowledge about the likelihood that a student has achieved a high mark for CI, given a particular IQ, based on having seen 200 students. We have,

- $n_0 = 200$
- Prof. S says that $\lambda = 0.8$ of the students with a high IQ have been awarded a high mark for CI

Available dataset D:

Student	**G**ender	**I**Q	**H**igh mark for CI
1	male	low	no
2	female	average	yes
3	male	high	yes
4	female	high	yes

Example of prior knowledge (cont.)

Based on the dataset D above, it follows:

- $\widehat{P}_D(H = y \mid I = h) = 1$
- $n = 4$

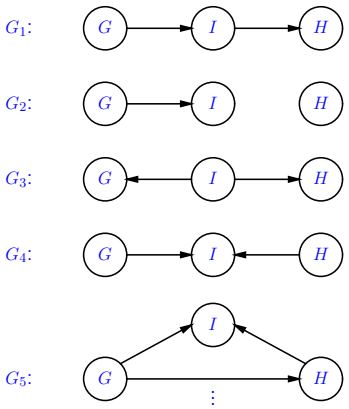
Prof. S and dataset D are combined as follows:

$$\begin{aligned} P(H = y \mid I = h, D) &= \frac{n}{n+n_0}\widehat{P}_D(H = y \mid I = h) \\ &\quad + \frac{n_0}{n+n_0}\theta \\ &= \frac{4}{4+200} \cdot 1 + \frac{200}{4+200} \cdot 0.8 \\ &= 0.020 \cdot 1 + 0.98 \cdot 0.8 = 0.804 \end{aligned}$$

Remark: $P(H = y \mid I = h)$ changed from 1 (data) to 0.804 (data and prior knowledge)

Learning structure from data

Given the above dataset D and the following Bayesian networks:



Which one is best?

Learning structure from data (cont.)

Inducing the structure of a BN from the data:

- Search-and-score algorithms
  - Search algorithm: select subset of (high-quality) BNs
  - Quality measure: decide which one of the (candidate) networks is the best
- Constraint-based structure learning: identifies ADG structure that best encodes a set of conditional dependence and independence relations
- Two ADGs representing the same set of condition dependences and independence relations are *equivalent*

Search+Score methods

- Search algorithm induce Bayesian network models from data
  - Find structures that fit the data
- A score measures how well a Bayesian network describes a set of data
  - Different models can then be compared in order to decide which is best
- As we are going to see next, score functions should take into account model complexity
  - More complex models are penalised

Find a quality measure

- A quality measure is a criterion by which one can order a set of possible BNs
- Desired property: networks leading to the same independence structure should be assigned the same quality value

Let D be a dataset (multi-set) of cases, and $\mathcal{B} = (G, P)$ and $\mathcal{B}' = (G', P')$ be two Bayesian networks, then

$$q = \frac{\Pr(G \mid D)}{\Pr(G' \mid D)}$$

is a (Bayesian) measure, with $\Pr$ a probability distribution defined on BNs and datasets, that can be used to rank Bayesian-network structures.

Find a quality measure (cont.)

Note that:

$$q = \frac{\Pr(G, D)/\Pr(D)}{\Pr(G', D)/\Pr(D)} = \frac{\Pr(G, D)}{\Pr(G', D)}$$

and

$$\Pr(G, D) = \Pr(D \mid G)\Pr(G)$$

Hence:

$$\log \Pr(G, D) = \log \Pr(D \mid G) + \log \Pr(G)$$

must be determined for each Bayesian network $\mathcal{B}$.

Determining $\Pr(D \mid G)$ (cont.)

Let $\mathcal{B} = (G, P)$ be a Bayesian network, with $G = (V(G), A(G))$, and joint probability distribution $P_{\mathcal{B}}$.

- Assumption 1: no missing values in D
- Assumption 2: cases $\mathbf{v} \in D$ have occurred independently
- Assumption 3: discrete network parameters

A common quality measure of a Bayesian model is:

$$\Pr(D \mid G) = \prod_{i=1}^{N} \prod_{j=1}^{q_i} \prod_{k=1}^{r_i} {\theta_{ijk}}^{n_{ijk}}$$

Determining $\Pr(D \mid G)$ (cont.)

In the above formula,

- N is the number of variables in the model
- q_i denotes de number of states over the parents of X_i in the graph ($q_i = 1$, if X_i has no parents)
- r_i denote the number of states for a variable X_i,
- θ is the estimate of the parameters of the model, and
- n_{ijk} denotes the number of cases in the database with X_i in its kth state and parent of X_i in its jth state

This measure estimates the maximum likelihood parameters for the model.

Maximum likelihood score

$$\Pr(D \mid G) = \prod_{i=1}^{N} \prod_{j=1}^{q_i} \prod_{k=1}^{r_i} {\theta_{ijk}}^{n_{ijk}}$$

Usually the log of this value is considered:

$$\log \Pr(D \mid G) = \sum_{i=1}^{N} \sum_{j=1}^{q_i} \sum_{k=1}^{r_i} n_{ijk} \cdot \log \left(\frac{n_{ijk}}{n_{ij}} \right)$$

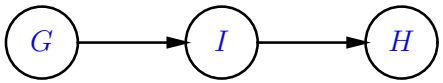
Given a complete database, this becomes a matter of frequency counting, where the parameters are maximized by $\hat{\theta}_{ijk} = \frac{n_{ijk}}{n_{ij}}$ and $n_{ij} = \sum_{k=1}^{r_i} n_{ijk}$

Example

Assume the database,

Student	**G**ender	**I**Q	**H**igh mark for CI
1	male	low	no
2	female	average	yes
3	male	high	yes
4	female	high	yes

and the model described by the graph G_1

G_1: 

Example (cont.)

- There are 3 variables/vertices (N = 3)
- Vertex G does not have any parents

$$\lg \Pr(D \mid G_1) = \sum_G n_G \lg \frac{n_G}{n} + \sum_I \sum_G n_{I \wedge G} \lg \frac{n_{I \wedge G}}{n_G} + \sum_I \sum_H n_{H \wedge I} \lg \frac{n_{H \wedge I}}{n_I}$$

Note: $\lg \equiv \log_2$

Example (cont.)

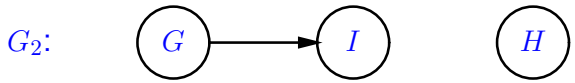
$$\begin{aligned}\lg \Pr(D \mid G_1) &= 2 \lg \frac{2}{4} + 2 \lg \frac{2}{4} + \\ &\quad 1 \lg \frac{1}{2} + 0 \lg \frac{0}{2} + \\ &\quad 0 \lg \frac{0}{2} + 1 \lg \frac{1}{2} + \\ &\quad 1 \lg \frac{1}{2} + 1 \lg \frac{1}{2} + \\ &\quad 1 \lg \frac{1}{1} + 0 \lg \frac{0}{1} + 0 \lg \frac{0}{2} + \\ &\quad 0 \lg \frac{0}{1} + 1 \lg \frac{1}{1} + 2 \lg \frac{2}{2} \\ &= -8\end{aligned}$$

Another example

Assume the same database,

Student	**G**ender	**I**Q	**H**igh mark for CI
1	male	low	no
2	female	average	yes
3	male	high	yes
4	female	high	yes

and another model described by the graph G_2



Example (cont.)

- There are 3 variables/vertices (N = 3)
- Vertex G and H do not have any parents

$$\lg \Pr(D \mid G_2) = \sum_G n_G \lg \frac{n_G}{n} + \sum_I \sum_G n_{I \wedge G} \lg \frac{n_{I \wedge G}}{n_G} + \sum_H n_H \lg \frac{n_H}{n}$$

Another example (cont.)

$$\begin{aligned}\lg \Pr(D \mid G_2) &= 2\lg\frac{2}{4} + 2\lg\frac{2}{4} + \\ &\quad 1\lg\frac{1}{2} + 0\lg\frac{0}{2} + \\ &\quad 0\lg\frac{0}{2} + 1\lg\frac{1}{2} + \\ &\quad 1\lg\frac{1}{2} + 1\lg\frac{1}{2} + \\ &\quad 1\lg\frac{1}{4} + 3\lg\frac{3}{4} \\ &= -11.25\end{aligned}$$

$\Rightarrow \Pr(D \mid G_1) > \Pr(D \mid G_2)$

What about the prior $\Pr(G)$?

(1) Try to incorporate background knowledge about $\mathcal{B}$, or

(2) Assume that all Bayesian networks are equally likely, i.e. $\Pr(G)$ is a uniform probability distribution

For (2) it holds that:

$$\log \Pr(G, D) = \log \Pr(D \mid G) + c$$

with $c \in \mathbb{R}$, a constant

Hence,

$$\log q = \log \Pr(D \mid G) - \log \Pr(D \mid G')$$

This is called the logarithmic Bayes factor.

Limitations of $\Pr(G, D)$

We have seen that the

$$\log \Pr(G, D) = \log \Pr(D \mid G) + \log \Pr(G)$$

can be used as a quality measure.

However, $\Pr(G, D)$ is usually higher for more complex (i.e. with more arcs) networks.

So, a measure that takes into account the complexity of a candidate network is needed.

Penalising factor

Solution: add factor r that penalises complexity

$$r = -\frac{1}{2}k \cdot \log n$$

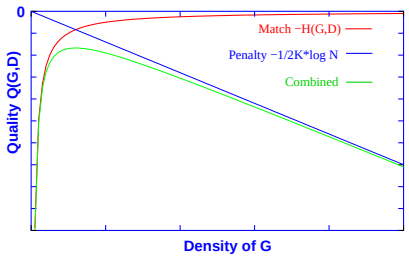
where k number of parameters required to completely specify the joint probability distribution.

For BN with binary variables, $k = \sum_{X \in V(G)} 2^{|\pi(X)|}$)

Result:

$$Q(G, D) = \log \Pr(G) + \log \Pr(D \mid G) - \frac{1}{2}k \cdot \log n$$

Quality measure Q



$$Q(G, D) = \log \Pr(G) - n \cdot H(G, D) - \frac{1}{2} k \cdot \log n$$

where:

- $\Pr(G)$: prior probability of G
- $-H(G, D)$: negative value of match
- $-\frac{1}{2} k \cdot \log n$: penalty term

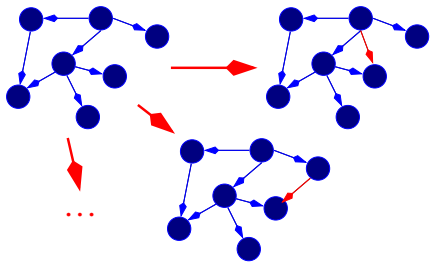
How many ADGs have to be considered?

Network structures for N vertices (Robinson formula):

$$f(N) = \sum_{i=1}^{N} (-1)^{i+1} \binom{N}{i} 2^{i(N-i)} f(N-i)$$

n	Number of ADGs
1	1
2	3
3	25
⋮	⋮
8	783,702,329,343
9	1,213,442,454,842,881
10	4,175,098,976,430,598,143

Searching an optimal graph



The space of possible directed acyclic graphs for n variables is very large.

Therefore, heuristic methods are necessary in order to optimize the score in such space.

K2 algorithm

- An ordering of the nodes is assumed
- A maximum number of parents for each node is also given
- It holds that initial node does not have parents
- Then, for each node,
  - it starts with the empt set of parents
  - adds as parent the node preceding it in the given order, if this produces an increase in the score
- It continues adding parents while the score increases and the number of parents does not exceed the maximum

K2 algorithm (cont)

The key property of the score which makes such an algorithm feasible is decomposability.

Such score guarantees that local changes imply only local computations.

The log likelihood is a sum of functions depending of nodes and their parents. If only one node changes its parents, the only part of the score corresponding to this node has to be recomputed.

Heuristic search

Algorithm for a graph G:

(1) add (or delete) one arc

(2) compute the gain in quality

(3) repeat (1) and (2) for every possible arc

(4) choose the arc with maximal gain, and add (delete) it

The above algorithm is called for the null graph (adding), or the complete graph (deleting).

Structure constraints

- Background knowledge in terms of constraints on the structure of the ADG can be specified (e. g., $X \perp\!\!\!\perp Y$)
- That is, makes use of particular properties of graphical models, namely conditional dependences and independences
- It is essential to induce such relations from the probability distribution implied by the data
- Can recover the correct DAG (i.e., perfect map)
- Does not get stuck in local optima, unlike search strategies based which aim to optimise a scoring function

PC Algorithm

The PC algorithm (implemented in Tetrad and Hugin tools) have the following steps:

- Test the conditional independence between each pair of variables in order to derive the conditional dependences and independences
- Identify the graph skeleton induced by those relations
- Identify convergent connections ($X \rightarrow Z \leftarrow Y$ structures)
- Identify derived directions

Example

From data, determine the validity of conditional statements such as $X \perp\!\!\!\perp Y \mid S_{XY}$

For instance,

$$M_{\perp\!\!\!\perp} = \{B \perp\!\!\!\perp E, B \perp\!\!\!\perp R, B \perp\!\!\!\perp W \mid \{A\}, A \perp\!\!\!\perp R \mid \{E\}, E \perp\!\!\!\perp W \mid \{A\}, R \perp\!\!\!\perp W \mid \{A\}\}$$

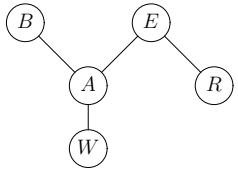
$$M_{\not\perp\!\!\!\perp} = \{B \not\perp\!\!\!\perp A, B \not\perp\!\!\!\perp A \mid \{E\}, B \not\perp\!\!\!\perp A \mid \{R\}, B \not\perp\!\!\!\perp A \mid \{W\}, B \not\perp\!\!\!\perp A \mid \{E, R\}, B \not\perp\!\!\!\perp A \mid \{E, W\}, B \not\perp\!\!\!\perp A \mid \{R, W\}, B \not\perp\!\!\!\perp A \mid \{E, R, W\}, A \not\perp\!\!\!\perp E, \ldots, A \not\perp\!\!\!\perp W, \ldots, E \not\perp\!\!\!\perp R, \ldots\}$$

Example (cont)

The skeleton of the graph is constructed from the conditional dependence and independence statements.

For each pair of variables X and Y where no independence statement $X \perp\!\!\!\perp Y \mid S_{XY}$ exists, the undirected edge (X, Y) is created in the skeleton.

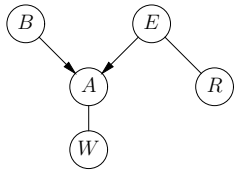
Given the previous statements,



Example (cont)

Once the skeleton has been identified, convergent connections are then identified.

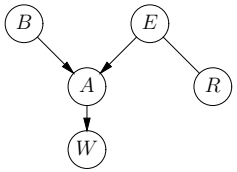
Based on the skeleton, search for subsets of variables $\{X, Y, Z\}$ s.t. X and Y are neighbours, and Z and Y are neighbours while X and Z are not neighbours. For each subset, a collider $X \rightarrow Z \leftarrow Y$ is created.



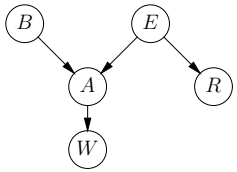
Example (cont)

Directions of arcs are then derived, following four rules. Rules are repeatedly applied until no further edge can be given an orientation. In our example, this results in (a).

Since based on data alone the direction between E and R cannot be determined, direction can be either chosen at random or provided by expert knowledge (if any).



(a)



(b)

References

- Spirtes, P., Glymour, C. and Scheines, R. *Causation, Prediction and Search Adaptive Computation and Machine Learning*, 2 ed, MIT Press, 2000.
- Lauritzen, S. L. The EM algorithm for graphical association models with missing data. *Computational Statistics and Analysis*, **19**: 191-201, 1995.
- Bouckaert, R. R. *Bayesian Belief Networks: From Construction to Inference*. Ph.D. Thesis, Dept. of Computer Science, Utrecht University, 1995.
- Spirtes, P. and Meek, C. Learning Bayesian Networks with Discrete Variables from Data. In *Proc. of the 1st Intl. Conf. on Knowledge Discovering and Data Mining*, pp. 294-300. AAAI Press, 1995.
- R.W. Robinson. *Counting unlabeled acyclic graphs*. LNM 622, Springer, NY, 1977.