(Somewhat) Tractable Representation of Joint Distributions

(Chapter 12

Factored Joint Distributions using Bayesian Networks

- A way of representing and reasoning with the full joint probability distribution
  - Can answer any kind of probabilistic query. For eg. $P(A = yes|B = yes)$ or $P(B = no|A = yes)$
- Capture independence and conditional independence where they exist
  - For eg. $P(A|C) = P(A)$
- Among variables where dependencies exist, encode the relevant portion of the full joint distribution
  - For eg. $P(A = yes|B = yes) = 0.80$
- Use a graphical representation, making it easier to visualise, investigate complexity and study inference algorithms

Bayesian Network: What it Is

- A Bayesian Network is a Directed Acyclic Graph (DAG) in which
  1. Each node denotes some random variable X
  2. Each node has a conditional probability distribution $\mathrm{P}(X | \mathit{Parents}(X))$

- The intuitive meaning of an arc from node X to node Y is that X *directly influences* Y

Aside: Property of DAGs

- An *ancestral ordering* of the n nodes in a directed graph G is an ordering $[v_1, v_2, \ldots, v_n]$ such that the ancestors of a node v_i appear before v_i in the ordering
- Key property: DAGs always have at least one ancestral ordering.

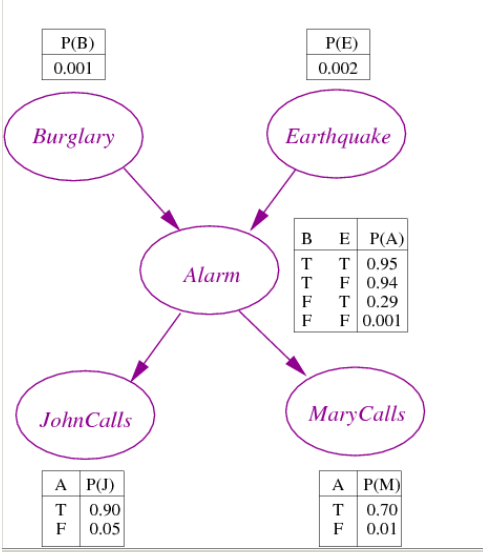
Bayesian Network: Additional Terminology

- If X and its parents are discrete, we can represent the distribution $\mathrm{P}(X|\mathit{Parents}(X))$ by a *conditional probability table* (CPT)

- The CPT specifies the probability of each value of X given each possible combination of values for variables in $\mathit{Parents}(X)$

- A *conditioning case* is a row in the CPT

Another Bayesian Network



P(B)
0.001

P(E)
0.002

B	E	P(A)
T	T	0.95
T	F	0.94
F	T	0.29
F	F	0.001

A	P(J)
T	0.90
F	0.05

A	P(M)
T	0.70
F	0.01

Bayesian Network: What it Means

A Bayesian Network can be understood as:

1. A representation of the full joint distribution over its random variables; or
2. A collection of conditional independence statements.

(1) is helpful in understanding BN construction
(2) is helpful in understanding BN inference

BN Representation of the Full Joint Distribution

- A generic entry in the full joint distribution is

$$\mathrm{P}(X_1 = x_1 \wedge \ldots \wedge X_n = x_n)$$

  or $\mathrm{P}(x_1, \ldots, x_n)$ for short

- By definition, in a BN this is given by

$$\mathrm{P}(x_1, \ldots, x_n) = \prod_{i=1}^{n} \mathrm{P}(x_i | Parents(X_i))$$

Chain Rule

- Generalisation of the product rule

$$\mathrm{P}(x_1 \wedge x_2) = \mathrm{P}(x_2|x_1)\mathrm{P}(x_1)$$

- Chain rule

$$\begin{aligned}\mathrm{P}(x_1,\ldots,x_n) &= \mathrm{P}(x_n|x_{n-1},\ldots,x_1)\mathrm{P}(x_{n-1},\ldots,x_1)\\ &= \mathrm{P}(x_n|x_{n-1},\ldots,x_1)\mathrm{P}(x_{n-1}|x_{n-2},\ldots,x_1)\cdots\mathrm{P}(x1)\\ &= \prod_{i=1}^{n}\mathrm{P}(x_i|x_{i-1},\ldots,x_1)\end{aligned}$$

Chain Rule and BNs

- BN

$$\mathrm{P}(x_1, \ldots, x_n) = \prod_{i=1}^{n} \mathrm{P}(x_i | Parents(x_i))$$

- Chain rule

$$\mathrm{P}(x_1, \ldots, x_n) = \prod_{i=1}^{n} \mathrm{P}(x_i | x_{i-1}, \ldots, x_1)$$

- For a BN to correctly represent the joint distribution:

$$\mathbf{P}(X_i | Parents(X_i)) = \mathbf{P}(X_i | X_{i-1}, \ldots, X_1)$$

Chain Rule and BNs: contd.

For a BN to correctly represent the joint distribution:

$$\mathbf{P}(X_i | Parents(X_i)) = \mathbf{P}(X_i | X_{i-1}, \ldots, X_1)$$

This follows provided:

- There is an ordering such that $Parents(X_i) \subseteq \{X_{i-1}, \ldots, X_1\}$. (true for DAGs).
- X_i is conditionally independent of its non-descendents, given its parents. (This is called the *Markov condition*). It follows that X_i is conditionally independent of its predecessors in the node ordering, given its parents. That is, $Parents(X_i)$ must contain all nodes that directly influence X_i.

Procedure for BN Construction

- Choose relevant variables that describe the domain
- Choose an (ancestral) ordering for the variables
- While there are variables left:
  1. Select next variable X_i in the order and add a node for it.
  2. Set $Parents(X_i)$ to some minimal set of nodes already in the net such that conditional independence property is satisfied.
  3. Define $\mathbf{P}(X_i|Parents(X_i))$.

Principles to Guide Choices

- Goal: build a locally structured (sparse) network. Each node interacts with a bounded number of other nodes (regardless of the total number of nodes).

- Add *root causes* first, and then the variables they influence (construct a causal model, as opposed to a diagnostic model)

Conditional Independence Again

- Recall that a node X is conditionally independent of its predecessors (in an ancestral ordering) given $Parents(X)$

- *Markov Blanket* of X: the set consisting of the parents of X, children of X, and the children's parents.

- It can be shown that X is conditionally independent of all nodes in the network given its Markov blanket.

Note on Representation Size

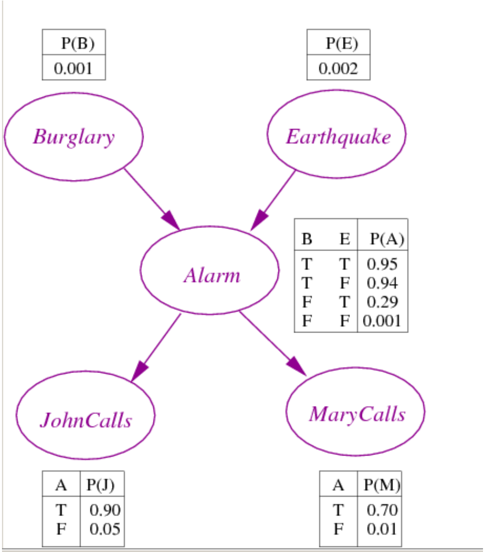
- At first glance, it would appear that the space to represent a Bayes Net is quadratic in the number of variables (possible number of arcs).

- We must also represent the CPT of each node, which in general will have size exponential in the number of parents of the node.

Compact Representations of CPTs

- Nodes may simply have a deterministic logical or numerical relationship to the parents. This function can be stored intentionally (rather than extensionally using a table).
- In other cases, CPTs may fall into one of several common categories or canonical distributions.
  - These canonical forms are based on regularities that permit much more compact representations.
  - Conditional Gaussian distribution (child is numerical, parents are a mixture of discrete and numerical variables)
  - Logit or probit distribution (child is boolean, parents are a mixture of discrete and numerical variables)

Alarm Again



P(B)
0.001

P(E)
0.002

B	E	P(A)
T	T	0.95
T	F	0.94
F	T	0.29
F	F	0.001

A	P(J)
T	0.90
F	0.05

A	P(M)
T	0.70
F	0.01

The Basic Inference Task in a BN

- Given some observed **event** (some assignment of values to a set of **evidence variables**), compute the posterior probability distribution over a set of **query variables**
  - Most common inference task: $\mathbf{P}(X|\mathbf{e})$
- Variables that are neither evidence variables or query variables are **hidden variables**
- A BN is flexible enough that any set of variables can be the query variable, and any other set can be evidence variables

Inference By Enumeration

$$\mathbf{P}(Burglary|johnCalls, maryCalls) = \langle 0.284, 0.716 \rangle$$

- How can we compute such answers?
- One approach: compute the full joint distribution represented by the network. We can then answer any query – but this would defeat the purpose of using a Bayesian Network.
- Instead, we will use the fact that a conditional probability can be obtained by summing terms from the full joint. Recall:

$$\mathbf{P}(X|\mathbf{e}) = \alpha \mathbf{P}(X, \mathbf{e}) = \alpha \sum_{\mathbf{y}} \mathbf{P}(X, \mathbf{e}, \mathbf{y})$$

Getting a Full Joint Value from a BN

- Recall that in a BN

$$\mathrm{P}(x_1, \ldots, x_n) = \prod_{i=1}^{n} \mathrm{P}(x_i | \mathit{Parents}(X_i))$$

- So, the joint probability required by the r.h.s. of the equation on the previous slide can be calculated using the BN.

Example I

$$\begin{aligned}
\mathbf{P}(B|j,m) &= \alpha \sum_{\mathbf{e}} \sum_{\mathbf{a}} \mathbf{P}(B,j,m,\mathbf{e},\mathbf{a}) \\
\mathbf{P}(B|j,m) &= \alpha \sum_{\mathbf{e}} \sum_{\mathbf{a}} \mathbf{P}(B)\mathrm{P}(j|\mathbf{a})\mathrm{P}(m|\mathbf{a})\mathrm{P}(\mathbf{e})\mathbf{P}(\mathbf{a}|B,\mathbf{e})
\end{aligned}$$

Solve separately for $B = \mathit{true}$ and $B = \mathit{false}$:

$$\begin{aligned}
\mathrm{P}(b|j,m) &= \alpha \sum_{\mathbf{e}} \sum_{\mathbf{a}} \mathrm{P}(b)\mathrm{P}(j|\mathbf{a})\mathrm{P}(m|\mathbf{a})\mathrm{P}(\mathbf{e})\mathrm{P}(\mathbf{a}|b,\mathbf{e}) \\
&= \alpha \mathrm{P}(b) \sum_{\mathbf{e}} \mathrm{P}(\mathbf{e}) \sum_{\mathbf{a}} \mathrm{P}(j|\mathbf{a})\mathrm{P}(m|\mathbf{a})\mathrm{P}(\mathbf{a}|b,\mathbf{e})
\end{aligned}$$

(Similarly for $B = \mathit{false}$)

Example II

From the CPTs in the BN:

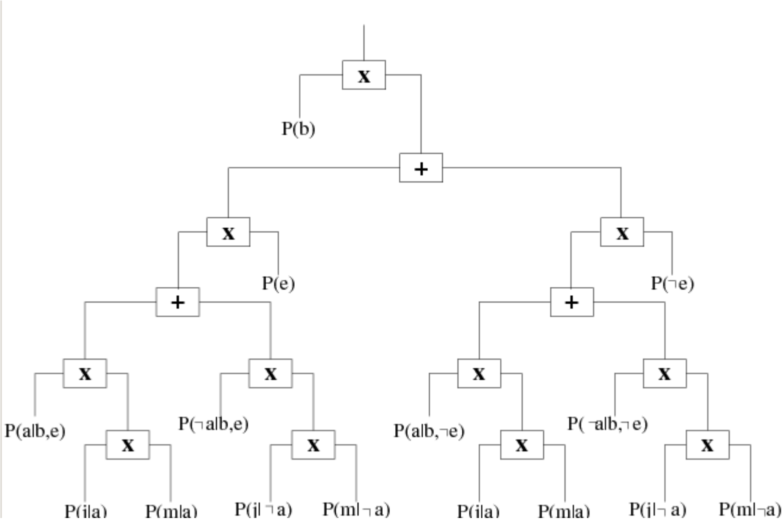
$$\begin{aligned} \mathrm{P}(b|j,m) &= \alpha 0.000592 \\ \mathrm{P}(\neg b|j,m) &= \alpha 0.001494 \end{aligned}$$

Normalising:

$$\begin{aligned} 0.000592\alpha + 0.001494\alpha &= 1.0 \\ \alpha &\approx 479 \end{aligned}$$

$$\mathbf{P}(B|j,m) = \langle 0.284, 0.716 \rangle$$

Expression Tree for Computation



Some Observations

- Number of terms in the sum is exponential in the number of *hidden variables*.
- Many sub-expressions are repeated on multiple branches. Each could be computed once and saved . . . leads to the idea of *variable elimination*.
- Variable elimination avoids repeating subcomputations (recall simplification of expression trees by compilers)

Special Bayesian Networks I

- If we are concerned with the problem of conditional class probability estimation ($\mathbf{P}(Y|\mathbf{X})$), then some special Bayesian networks result by making specific assumptions
- Suppose we have observed n data points, each of which labelled by a random variable Y which takes values from a discrete set (say: $\{+,-\}$, for simplicity) Each data point is a d-dimensional random vector $\mathbf{X} = [X_1, X_2, \ldots, X_d]^T$ where the $X_i \in \Re$.
- Given a particular vector $\mathbf{x} = [x_1, x_2, \ldots, x_d]^T$ we wish to obtain an estimate of the conditional probabilities of $Y = +$ (the corresponding probability for $Y = -$ follows automatically). From Bayes rule, the relevant posterior is given as:

$$P(Y = +|\mathbf{x}) = \frac{P(\mathbf{x}|Y = +)P(Y = +)}{P(\mathbf{x})}$$

Special Bayesian Networks II

Two well-known simple cases follow from specific assumptions about the data:

1. The assumption that the class-conditional densities $P(\mathbf{x}|\cdot)$ are from the exponential class (of which the Gaussian is a member) results in:

   $$P(Y = +|\mathbf{x}) = \frac{1}{1 + e^{-\xi}}$$

   where ξ is a linear equation of the X_i.
   The *logistic regression* procedure uses the sample data and the maximum likelihood principle (correctly, conditional likelilood) to estimate probabilities under this assumption.

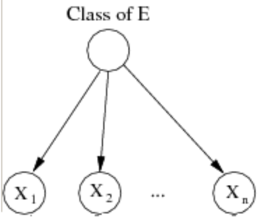
Special Bayesian Networks III

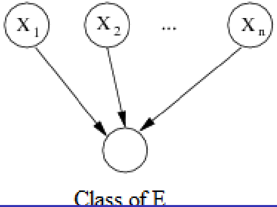
2. The assumption that the X_i are conditionally independent of each other given the value of Y, results in:

$$P(Y = +|\mathbf{x}) \propto \prod_{1}^{d} P(x_i|Y = +)P(Y = +)$$

The *naive Bayes* procedure uses sample data to estimate probabilities under this assumption.

- Both naive Bayes and logistic regression procedures simply encode specific Bayesian network topologies that reflect the underlying assumptions described





Special Bayesian Networks IV

- For both logistic regression and naive Bayes, estimation of parameters (logistic regression) or class-conditional probabilities (naive Bayes) can be done very efficiently
- These represent two of the simplest kinds of Bayesian networks for which tractable computation procedures exist. They have been shown empirically to be able to model a very wide range of observed data quite well

Summary

- A BN by definition contains a unique DAG and a unique joint distribution on the variables in the DAG. The joint distribution can be retrieved from the conditional distribution of variables given their parents.
  - Significant savings in computation result from exploiting conditional dependencies
  - The full joint distribution is expressed as the product of smaller conditional probabilites
  - This makes BNs a special kind of *factor graph* in which the factors are conditional (or prior) probabilities over random variables
- The general inference problem in Bayesian Networks is computationally hard, but for special cases, it can be quite efficient

ToDo

- LLMs can act as a bridge between natural language and structured probabilistic models like BNs, enabling both the definition of network structures and the computation of inference results
- Using the Alarm example:

  Specify. Give a text description of the nodes and relationships and use an LLM to visually show you the Directed Acyclic Graph (DAG) and the corresponding Conditional Probability Tables (CPTs).

  Code. Use and LLM to generatea Python scripts using libraries such as `pgmpy` to define the model, add evidence, and query the network for posterior probabilities.

  Infer. Use the code to obtain the probability for $P(B \mid j, m)$ and compare against the exact calculation in the slides