Announcements

Quiz today!

- Will be live on moodle after the lecture
- Live until tonight
- 20 mins, 4 questions, based on material from week 1-week 4

Logistic regression

Max Likelihood estimation = $\arg\min_{w} \sum_{i=1}^{n} \log(1 + exp(-yw^{\top}\overrightarrow{x})$

Logistic loss

Another way of thinking about the loss

For class $\in \{0,1\}$, we use the logistic function (or the distribution) :

$$P(y = 1 \,|\, f(w, x)) = \frac{1}{1 + e^{-f(w,x)}} \text{ and}$$

$$P(y = 0 \,|\, f(w, x)) = 1 - \frac{1}{1 + e^{-f(w,x)}} = \frac{1}{1 + e^{f(w,x)}}$$

Logistic loss or the log loss

$$P(y \mid f(w,x)) = \left(\frac{1}{1+e^{-f(w,x)}}\right)^{y} \times \left(1 - \frac{1}{1+e^{-f(w,x)}}\right)^{(1-y)}$$

$$\max \text{ log likehlihood} = \min\left(-1 \times \text{ log likehlihood}\right)$$

$$J(f(w)) = -\log\left(\prod_{i}^{n} P(y_i \mid f(w,x_i))\right)$$

$$= -\sum_{i}^{n} y_i \log(P(y=1 \mid f(w,x))) + (1-y_i)\log(P(y_i=0 \mid f(w,x_i)))$$

Cross entropy loss for multi class case

Extending log loss to multi class case:

For each of example, we have:

$$= -\sum_{c} y_c \log(P(y = c \,|\, f(w, x)))$$

Cross entropy loss for multi class case

Extending log loss to multi class case:

For each of example, we have:

$$= -\sum_{c} y_c \log(P(y = c \,|\, f(w, x)))$$

ML output			Target		
0.1	0.3	0.6	0	0	1
0.2	0.6	0.2	0	1	0
0.3	0.4	0.3	1	0	0

INM431: Machine Learning

Bayesian Networks

Pranava Madhyastha (pranava.madhyastha@city.ac.uk)

Based on materials by Artur S. d'Avila Garcez + Bishop's book + Murphy's book

Review: probability

Random variables: $A, B \in \{0,1\}$

Joint distribution $P(A, B) =$

a	b	$P(a, b)$
0	0	0.3
0	1	0.06
1	0	0.6
1	1	0.04

Conditional distribution

a	$P(A = a \mid B = 1)$
0	0.6
1	0.4

Specific rows

Marginal distribution: $P(A)$

a	$P(A = a)$
0	0.36
1	0.64

Sum over rows

B is marginalised out! (Eliminated)

Probabilistic inference

Given a joint distribution:

$$P(A, B, C, D)$$

Probabilistic inference involves:

a) Conditioning on specific evidence or information (C,D): C =1, D=1

b) Interested in a particular random variable (such as B):

$$P(B \mid C = 1, D = 1)$$

process of answering this (or computing the desired distribution) is called probabilistic inference.

Problems

In general, a joint distribution over n variables has size exponential in n.

How do we even specify an object that large?

Here, we will see that Bayesian networks offer an elegant solution.

Bayesian Networks: a birds eyed view

A technique for describing complex joint distributions (models) using simple, local distributions (conditional probabilities)

- Generally called probabilistic graphical models
- Use local interactions with local distributions
- Local interactions chain together to give global, indirect interactions

Example

Earthquakes and burglaries can cause an alarm to go off. Suppose you hear an alarm. How does learning that there's an earthquake change your beliefs about burglary?

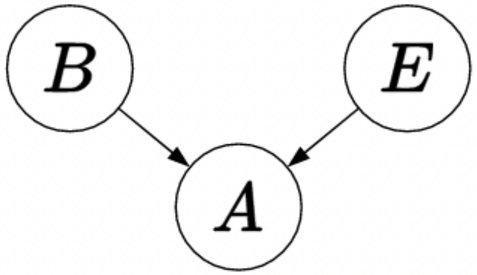
These arise all the time: we have a lot of unknown information which are all dependent on one another.

If we have access to some of the evidence, how does that affect our belief about the other unknowns?

This is called reasoning under uncertainty.

Example

Earthquakes (E) and burglaries (B) can cause an alarm (A) to go of Suppose you hear an alarm. How does learning that there's an earthquake change your beliefs about burglary?



a) establish that there are three variables E, B, A.

b) connect up the variables to model the dependencies.

c) for each variable, specify a local conditional distribution of that variable given its parent variables

d) define the joint distribution over all the random variables as the product of all the local conditional distributions

$$P(B, E, A) = P(B)P(E)P(A \mid B, E)$$

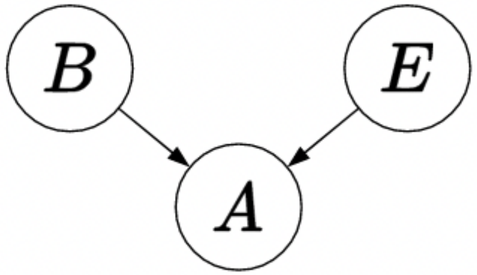
Explaining away

Assume that there are two causes that influence an effect.

Conditioned on the effect:

> Conditioning on one cause reduces the probability of the other cause!

Revisiting the example



Conditioned on A=1, there is some posterior probability of B. Conditioned on the effect A=1 and the other cause E=1, the new posterior probability is reduced.

We then say that the other cause E has explained away B.

Basic agenda

Random variables: capture the state of the world

Edge connections: denote the dependencies between the states

Local distributions = Probability of node given it’s parent

We then define the topology by the joint distribution, which in turn is the product of all of the local conditional distributions together

Probabilistic inference to ask questions about the world

Formal definition

Let $X = (X_1, \cdots, X_n)$ be random variables.

A Bayesian Network is a directed acyclic graph (DAG) that specifies a joint distribution over X as a product of ‘local conditional distributions’, one for each node:

$$P(X_1 = x_1, \cdots, X_n = x_n) = \prod_{i=1}^{n} P(x_i \,|\, x_{\text{parents}(i)})$$

Inference

Marginalise non-ancestors

Condition on the evidence

Probabilistic inference

Given a Bayesian network: $P(X_1 = x_1, \cdots, X_n = x_n)$

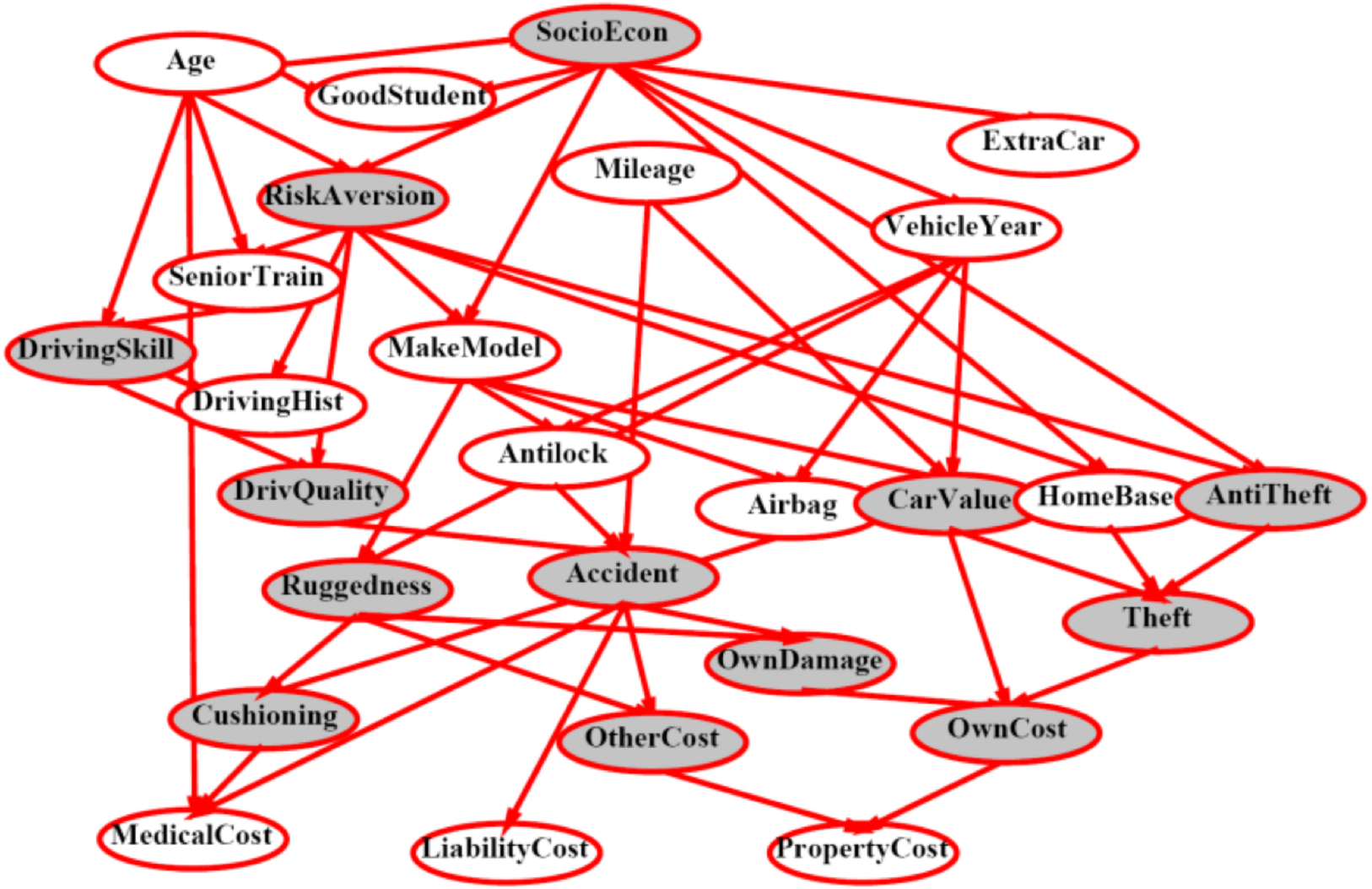
Evidence: $E = e$, where $E \subset X$ is the subset of variables.

When we are asked with a Query: $Q \subset X$ over a subset of variables

We perform probabilistic inference, by:

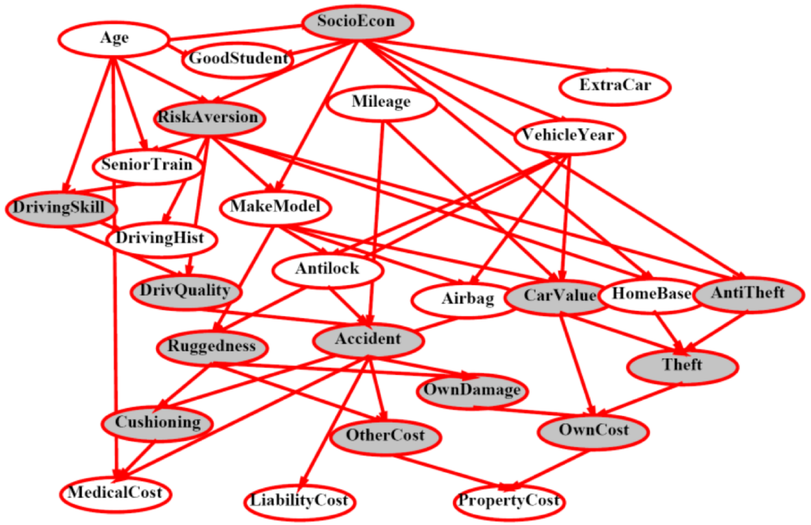
$$P(Q = q \,|\, E = e) \forall q$$

Example: Bayesian network for insurance



Example: Bayesian network for insurance

Given large number of nodes



If we want to specify a probability distribution over this whole graph, assuming there are only two cases

We have 2^{27} interactions based on 27 variables

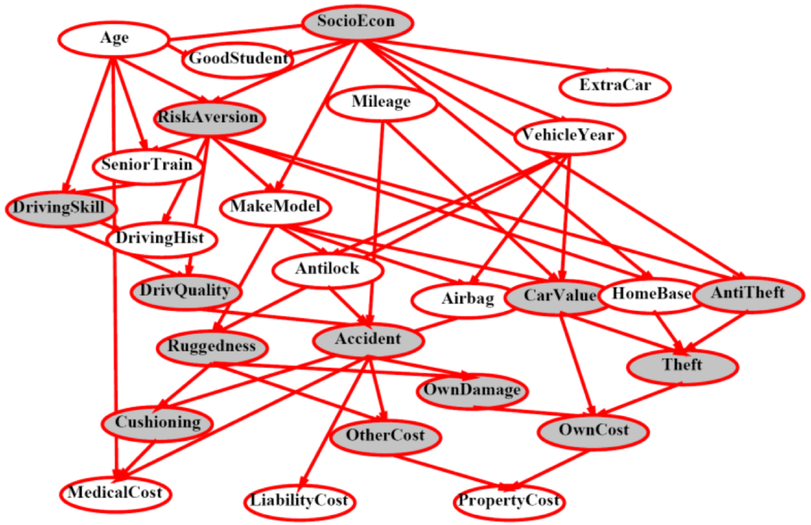
We will instead model with local interactions!

Example: Bayesian network for insurance

There could be many other variables that depend on the variables we've modelled.

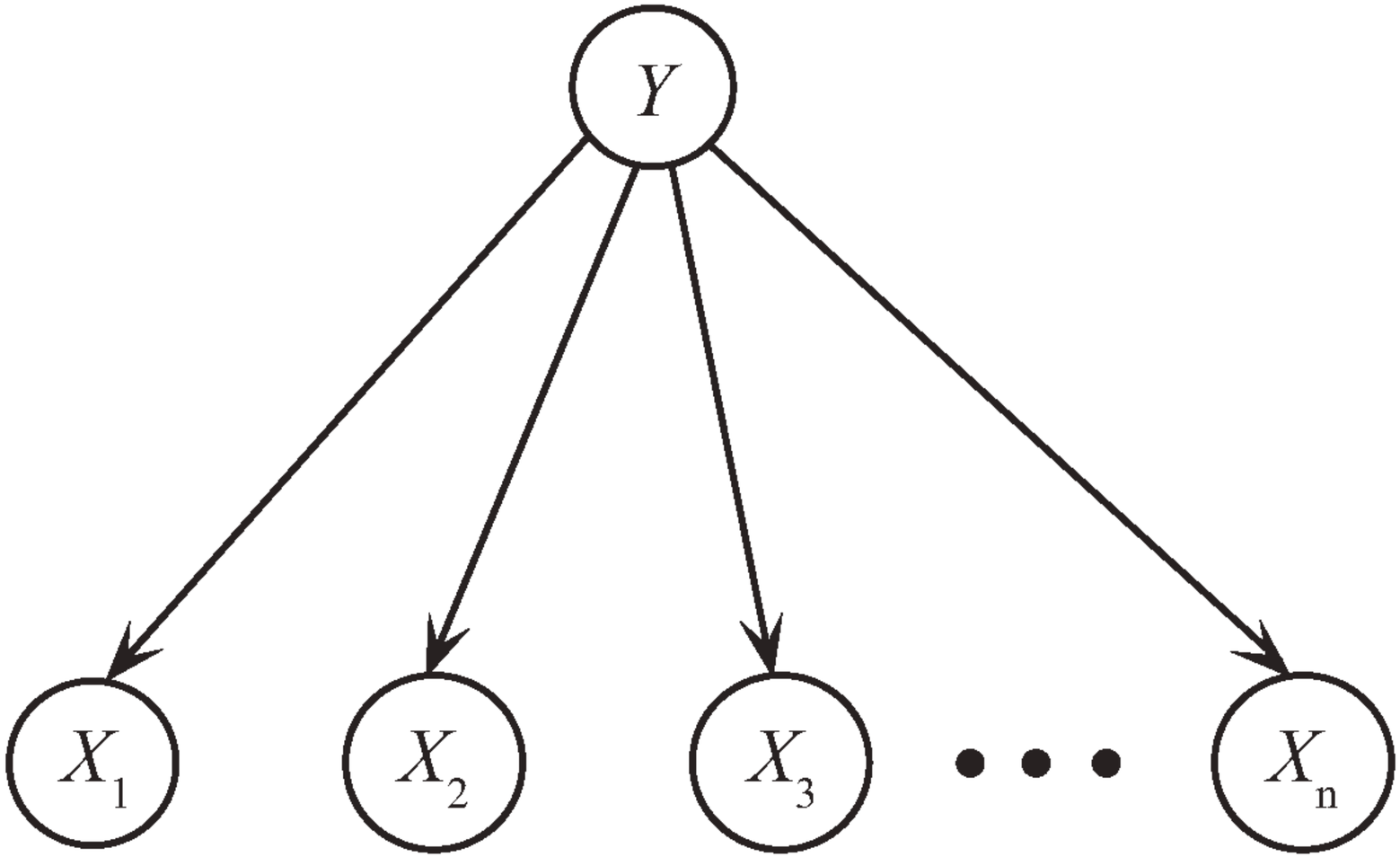
But as long as you don't observe them, they can be ignored mathematically (ignorance is bliss).

Note that this doesn't mean that knowing about the other things isn't useful.



Naïve Bayes algorithm as a Bayesian Network

Naïve Bayes algorithm as Bayes Net



Bayes Net: exercise, draw a bayesian network

Weather

Cavity

Toothache

Gum ache

General Applications

Language modelling: Markov models

Object tracking & Automatic speech recognition: Hidden Markov Models (Lecture 10)

Document classification: ‘?’ (Hint: spam classification)

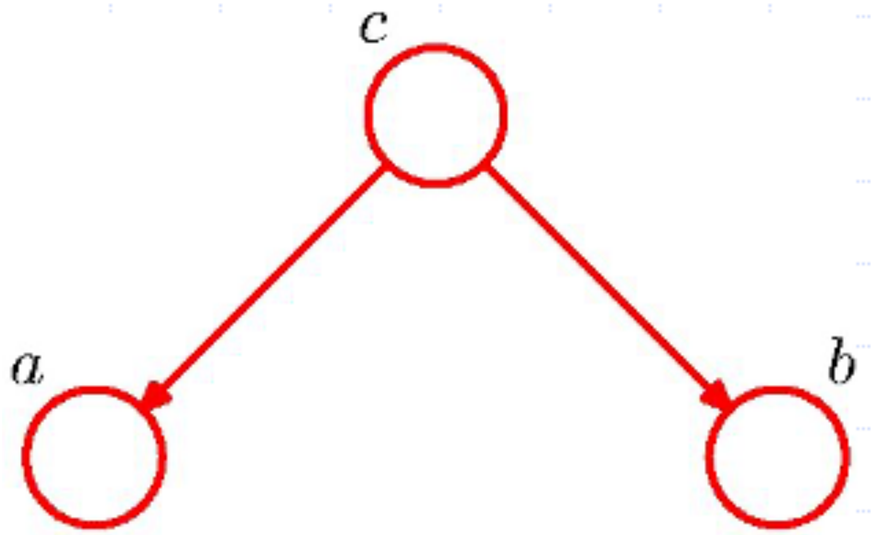
Medical diagnosis

Social network analysis and recommender systems

Some typical local interactions

$$P(a,b,c) = P(a\,|\,c)P(b\,|\,c)P(c)$$

$$P(a,b) = \sum_{c} P(a\,|\,c)P(b\,|\,c)P(c)$$

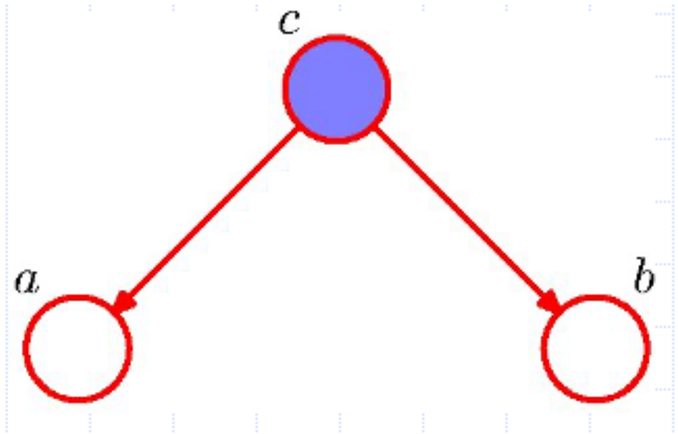


Note: node c is said to be tail-to-tail with respect to the path from a to b

Some typical local interactions

$$P(a,b\,|\,c) = \frac{P(a,b,c)}{P(c)}$$

$$= P(a\,|\,c)P(b\,|\,c)$$

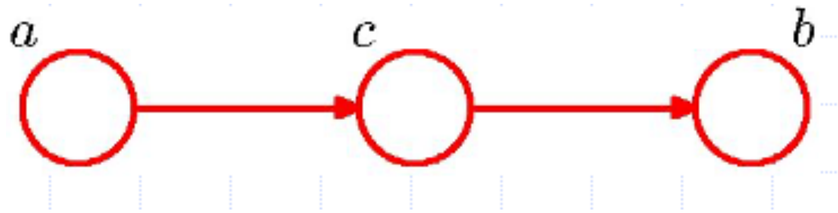


Some typical local interactions

$$P(a,b,c) = P(a)P(c\,|\,a)P(b\,|\,c)$$

$$P(a,b) = P(a)\sum_{c} P(c\,|\,a)P(b\,|\,c)$$

$$= p(a)p(b\,|\,a)$$



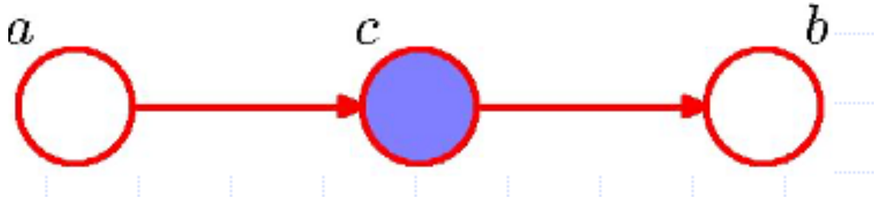
Note: node c is said to be head-to-tail with respect to the path from a to b

Some typical local interactions

$$P(a,b|c) = \frac{P(a,b,c)}{P(c)}$$

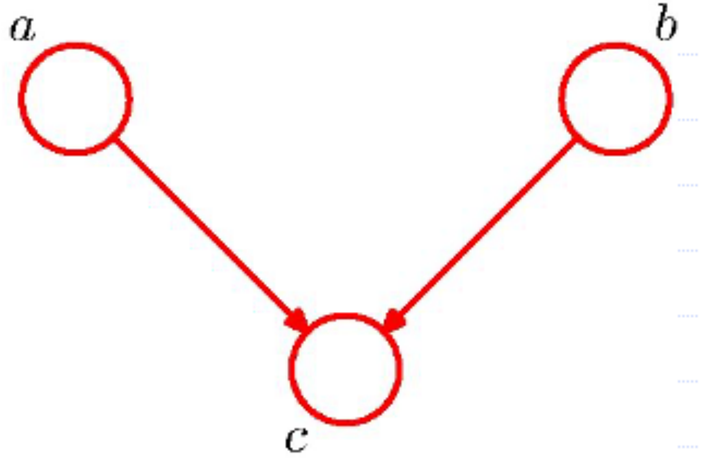
$$= \frac{P(a)P(c|a)P(b|c)}{P(c)}$$

$$= P(a|c)P(b|c)$$



Some typical local interactions

$$P(a, b, c) = P(a)P(b)P(c \mid a, b)$$



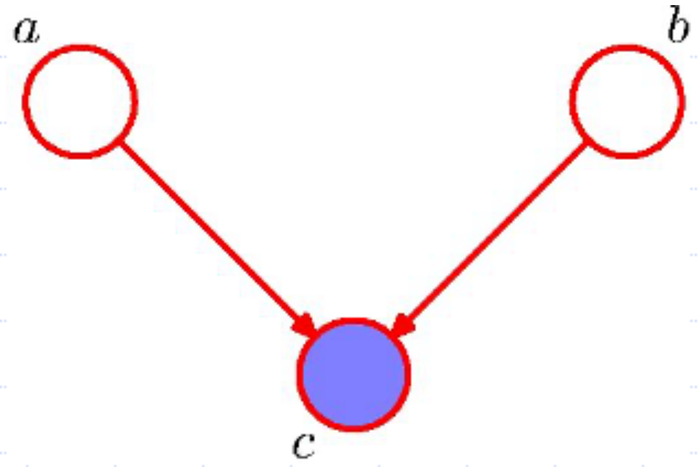
$$P(a, b) = P(a)P(b)$$

Note: node c is said to be head-to-head with respect to the path from a to b

Some typical local interactions

$$P(a,b\,|\,c) = \frac{P(a,b,c)}{P(c)}$$

$$= \frac{P(a)P(b)P(c\,|\,a,b)}{P(c)}$$



Topological constraints

Not all Bayesian nets can represent every joint probability disturbution

The topology of the network enforces specific conditional independencies

On causality

The causal direction may not always be true

C	$P(C)$
0	1/4
1	3/4

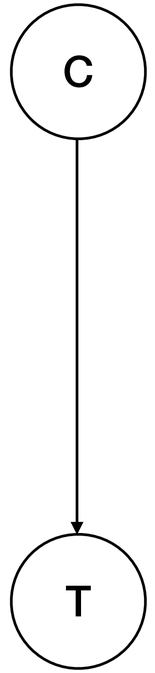
$C = 1$

T	$P(T \mid C)$
0	3/4
1	1/4

$C = 0$

T	$P(T \mid C)$
0	1/2
1	1/2

C	T	$P(C, T)$
0	0	3/16
0	1	1/16
1	0	6/16
1	1	6/16



On causality

Reverse causality

T	$P(T)$
0	9/16
1	7/16

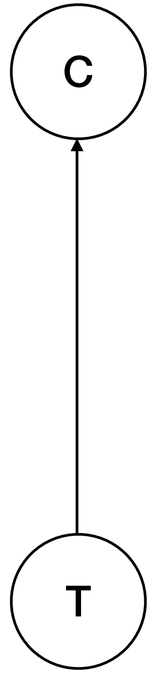
$T = 1$

C	$P(C \mid T)$
0	1/3
1	2/3

$T = 0$

T	$P(T \mid C)$
0	1/7
1	6/7

T	C	$P(C, T)$
0	0	3/16
0	1	1/16
1	0	6/16
1	1	6/16



On causality

The arrows may mean causality sometimes and the topology may indicate causal structure

- Topology always indicates the local dependencies

Bayesian nets sometimes reflect the true causal patterns

- When they do: intuitive, easier to work out

Bayesian nets may not reflect anything to do with causality

- Sometimes we may never have a causal process due to missing information

Learning Bayesian Networks

Consider this dataset:

case	x_1	x_2	x_3
1	1	0	0
2	1	1	1
3	0	0	1
4	1	1	1
5	0	0	0
6	0	1	1
7	1	1	1
8	0	0	0
9	1	1	1
10	0	0	0

Find the Bayesian network that best describes the observations

Learning Bayesian Networks

Probabilistic Inference

Parameter Learning with MAP and MLE (something similar to Naïve Bayes): mostly uses counting and normalising

Expectation maximisation (Lecture 9 and 10)

Structure Learning

General approach

In general, many different models are plausible

Key point: we want to come up with stories of how the data was generated through quantities of interest - the **Generative Story**

The number of possible Bayesian nets

Number of Directed Acyclic Graphs (DAGs) is super-exponential on the number of variables

Number of Variables	Possible DAG
1	1
2	3
3	25
4	543
5	29,281
6	3,781,503
7	1,138,779,265
8	78,370,2329,343
9	1,213,442,454,842,881

Score function: evaluates how well a given DAG matches the data, e.g. apply maximum likelihood and select the DAG that predicts the data with the highest probability

Structure Learning: score and search

Start from an initial structure (generated randomly or from domain knowledge) and move to the neighbour with the best score in the structure space until a local maximum of the score function is reached.

This greedy learning process can re-start several times with different initial structures to improve the result.

K2 Algorithm

The K2 search begins by assuming that a node (representing a discrete variable) has no parents and then adds incrementally that parent from a given ordering whose addition increases the score of the resulting structure the most.

We stop adding parents to the node when the score stops to increase.

Since we are using a fixed ordering, we do not need to check for cycles, and can choose the parents for each node independently.

K2 algorithm: basic skeleton

1. Applies to discrete variables; $x \in \{0,1,2,\cdots\}$

2. Assumes a maximum number n of parents for each node

3. Starts from an initial Bayesian network and moves to add parents incrementally to each node (deterministically or stochastically) until a local maximum is reached (i.e. score and search)

K2 algorithm: basic skeleton

4. Assumes a total order on the set of variables such that, e.g. if max no. of parents $n = 2$ and order is x_1, x_2, x_3, x_4 then:

x_1 can't have parents,

x_2 may have x_1 as parent,

x_3 may have x_1 and x_2 as parents,

x_4 may have two of x_1 ,x_2 and x_3 as parents.

The greedy search

Starts at a specific point (an initial tree, network, etc.) in the hypothesis space

Considers all nearest neighbours of the current point, and moves to the neighbour that has the highest score (with a probability in the case of stochastic search)

If no neighbours have higher score than the current point (i.e., we have reached a local maximum), the algorithm stops.

K2 score function

$$g(i, \pi_i) = \prod_{j=1}^{q_i} \frac{(r_i - 1)!}{(N_{ij} + r_i - 1)!} \prod_{k=1}^{r_i} N_{ijk}!$$

To compare the networks where node x_i has sets of parents π_i ; the highest score wins.

r_i is the size of the list of all possible values of x_i

q_i is the size of list with the Cartesian product of all possible values for the parents of x_i

N_{ijk} is the number of times in D that x_i takes its k^{th} value and the parents of x_i in π_i take the j^{th} instance of the Cartesian product

$N_{ij} = \sum_k N_{ijk}$: number of times the parents take the j^{th} instance

To summarise

A scoring function evaluates how well a given Bayesian network G matches the data D

Given a scoring function, the best Bayesian network is the one that maximises this scoring function

An ad-hoc scoring function is used based on maximum likelihood.

Other scoring functions

Criteria for model selection among a finite set of models! Seek to maximise likelihood by adding parameters (i.e. increasing the number of edges in the graph).

Doing so may result in overfitting. Therefore, add a penalty for larger DAGs, e.g.:

$$\max(\log(P(D \mid G)) - \frac{k}{2} \log N$$

where, D = data, G = graph, k = number of parameters in the model (e.g. number of coefficients of a regression model, number of entries in the probability tables of a Bayesian net), N = number of examples in D

Other scoring functions

In practice, given two or more graphs: For each graph, estimate maximum likelihood

$$L = \max(\log P(D \mid G))$$

from your dataset

Calculate the Bayesian Information Criterion

$$\text{BIC} = k \log N - 2L$$

The graph with the lowest BIC is preferred

Akaike information criterion (AIC): Similar to BIC but uses information loss:

$$\text{AIC} = 2k - 2 \log L$$

L is the maximum value of the likelihood function for a model

Other scoring functions

Some scoring functions are based on the concept of Mutual Information $I(X, Y)$: it measures how much information X and Y share, i.e. how much knowing one reduces uncertainty about the other.

Remember decision trees and mutual information?

Reading week!

Gentle suggestion: start thinking about the final projects

May be get started?

Next lecture: regression models