Lecture 4

Graphical Models: Bayesian networks

Financial Machine Learning
Nov. 30, 2020

Oualid Missaoui
Polytechnic School of Tunisia

Agenda

1. **Introduction**

2. **Graph Theory Primer**

3. **Bayesian Networks**
   - Model Specification
   - Inference
   - Parameter Learning
   - Structure learning
   - Bayesian Networks Structures

4. **References**

Probabilistic modeling

- Graphical models are a marriage between graph theory and probability to efficiently represent, learn and infer from full joint distributions.
- If the state of nature can be represented by M variables $X_{i=1,\cdots,M}$, then knowing the joint distribution

  $$P(X_1, X_2, \cdots, X_M)$$

  allows to answer any query against it.
- Classification: $M = D + 1$, $X_{D+1} = Y$ is the class, knowing the joint we can asnwer queries such as:

  $$P(Y = 1|X_1 = 1, X_2 = 1, \cdots, X_D = 1)$$

- For binary random variables, full specification of the joint distribution requires $2^M - 1$ parameters ($1.1259e + 15$ if $M = 50$) .

Explicit representation

The drawbacks of the explicit representation:

- computationally: very expensive to manipulate and generally too large to store in memory
- statistically: if we want to learn the distribution from data, we would need ridiculously large amounts of data to estimate this many paremeters robustly.
- cognitively: it is impossible to acquire so many numbers from a human expert; moreover the numbers are very small and do not correspond to events that people ca reasonably contemplate

These problems were the main barrier to the adoption of probabilistic methods for expert systems until the development of graphical models.

Chain Rule and Factorization

Exploiting conditional independence

- The chain rule of probabilities:

$$\begin{aligned} P(X_1, X_2) &= P(X_1)P(X_2|X_1) \\ P(X_1, X_2, X_3) &= P(X_1)P(X_2|X_1)P(X_3|X_1, X_2) \\ P(X_1, X_2, \cdots, X_n) &= P(X_1)P(X_2|X_1)\cdots P(X_n|X_1, \cdots, X_{n-1}) \\ &= \prod_{i=1}^{n} P(X_i|X_1, \cdots, X_{i-1}) \end{aligned}$$

- No gains yet. The number of parameters required by the factors is:

$$2^{n-1} + 2^{n-2} + \cdots + 1 = 2^n - 1$$

.

Conditional independence

- The joint distribution is factorized as $\prod_{i=1}^{n} P(X_i|X_1, \cdots, X_{i-1})$
- Potential reduction of $P(X_i|X_1, \cdots, X_{i-1})$ parameters if:
    - domain knowledge allows to identify a subset $\text{pa}(X_i) \subset \{X_1, \cdots, X_{i-1}\}$ such that

$$P(X_i|X_1, \cdots, X_{i-1}) = P(X_i|\text{pa}(X_i))$$

- Then

$$P(X_1, X_2, \cdots, X_n) = \prod_{i=1}^{n} P(X_i|\text{pa}(X_i))$$

- the number of parameters might have been substantially reduced.

Example

I'am at work, neighbor John calls to say my alarm is ringing, but neighbor Mary doesn't call. Sometimes it's set off by minor earthquakes. Is there a burglar?
Variables (binary/Boolean): Burglar (B), Earthquake (E), Alarm (A), John calls (J), Mary calls (M)
Question formulation:

$$P(B = 1|J = 1, M = 0) =?$$

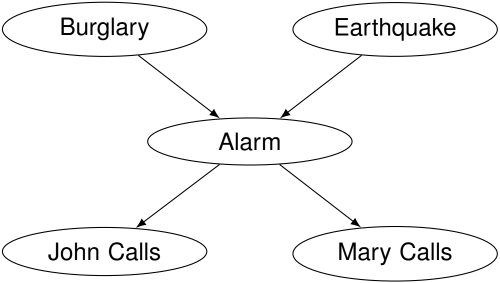
Answer:

$$P(B = 1|J = 1, M = 0) = \frac{\sum_{A,E} P(A, B = 1, E, J = 1, M = 0)}{\sum_{B,E,A} P(A, B, E, J = 1, M = 0)}$$

we need the full joint distribution $P(A, B, E, J, M)$,i.e., $2^5 - 1 = 31$ parameters to find.
Network topology reflects "causal" knowledge:

- A burglar can set the alarm off
- An earthquake can set the alarm off
- the alarm can cause Mary to call
- the alarm can cause John to call

Example



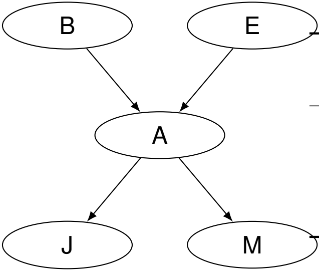
$$\begin{aligned}
P(A, B, E, J, M) &= P(B, E, A, J, M) \\
&= P(B)P(E|B)P(A|E, B)P(J|E, B, A)P(M|E, B, A, J) \\
&= P(B)P(E)P(A|E, B)P(J|A)P(M|A)
\end{aligned}$$

Using the conditional independence relationships, the number of parameters is reduced to 1+1+4+2+2= 10 from 31.

Example

B	P(B)
F	0.01
T	0.99

E	P(E)
F	0.01
T	0.99



		A	
B	E	T	F
F	F	0.4	0.6
F	T	0.01	0.99
T	F	0.01	0.99
T	T	0.01	0.99

	J	
A	T	F
F	0.4	0.6
T	0.01	0.99

	M	
A	T	F
F	0.4	0.6
T	0.01	0.99

Graph Theory Primer

- A graph $G = (\mathcal{V}, \mathcal{E})$ consists of a set of nodes or vertices $\mathcal{V} = \{1, \cdots, D\}$, and a set of edges, $\mathcal{E} = \{(s,t) : s,t \in \mathcal{V}\}$
- The adjacency matrix of graph G is defined as $G(s,t) = 1$ if $(s,t) \in \mathcal{V}$
- If $G(s,t) = 1$ iff $G(t,s) = 1$ we say the graph is undirected, otherwise it is directed.
- Usually, $G(s,s) = 0$: no self loops.

- **Parent** for a directed graph: $\text{pa}(s) = \{t : G(t,s) = 1\}$
- **Child** for a directed graph: $\text{ch}(s) = \{t : G(s,t) = 1\}$
- **Family** for a directed graph: $\text{fam}(s) = \{s\} \cup \text{pa}(s)$
- **Root** for a directed graph is a node with no parents
- **Leaf** for a directed graph is anode with no children

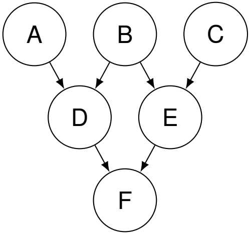
- **Ancestors** for a directed graph $\text{anc}(t) = \{s : s \rightsquigarrow t\}$
- **Descendants** for a directed graph $\text{desc}(s) = \{t : s \rightsquigarrow t\}$
- **Neighbors** for any graph $\text{nbr}(s) = \{t : G(s,t) = 1 \vee G(t,s) = 1\}$
- **Degree** of a node is the number of neighbors.
- **in-degree** for directed graph is the number of parents
- **out-degree** for directed graph is the number of children

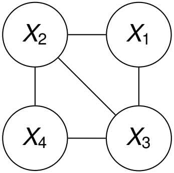
Graph Theory Primer

- Cycle or loop for any graph is a series of nodes such that we can get back to where we started. For directed graphs, we may speak of a directed cycle
- DAG is a directed graph with no directed cycles
- Topological ordering (total ordering) is a numbering of the nodes such that parents have lower numbers than their children
- Path $s \rightsquigarrow t$ is a series of directed edges leading from s to t
- A trail $s \rightsquigarrow t$ is a series of edges ($\rightleftharpoons$, can be in any direction) leading from s to t
- An undirected tree is an undirected graph with no cycles
- A directed tree is a DAG with an orientation (selected root node) in which each node at most one parent
- A polytree is a DAG with an orientation (selected root node). Nodes are allowed to have multiple parents.
- A forest is a set of trees.
- A subgraph G_A is the graph created by using the nodes in $A \subset \mathcal{V}$ and their corresponding edges in $\mathcal{E}$
- A clique for an undirected graph is a set of nodes that are all neighbors of each other.
- A maximal clique is a clique which cannot be made any larger without losing the clique property.

Graphical models representation of the joint distribution

- A graphical model (GM) is a way to represent a joint distribution by making conditional assumptions. In particular, the nodes in the graph represent random variables, and the absence (lack) of edges represent conditional independence assumptions.
- There are several kinds of graphical models depending on whether the graph is directed, undirected, or some combination of directed and undirected.





Graphical models representation of the joint distribution

Bayesian network

A distribution P factorizes according to a Bayesian network G if P can be expressed as:

$$P(X_1, \cdots, X_D) = \prod_{t=1}^{D} P(X_t | \text{pa}(X_t))$$

Markov network

A distribution P_Φ is a Gibbs distribution parametrized by a set of factors $\Phi = \{\phi_1(D_1), \cdots, \phi_K(D_K)\}$ if it is defined as follows

$$P_\Phi(X_1, \cdots, X_N) = \frac{1}{Z} \prod_{i=1}^{N} \phi_i(\boldsymbol{D}_i)$$

where

$$Z = \sum_{X_1, \cdots, X_N} \prod_{i=1}^{N} \phi_i(\boldsymbol{D}_i)$$

We say that a distirbution P_Φ with $\Phi = \{\phi_1(D_1), \cdots, \phi_K(D_K)\}$ factorizes over a Markov network $\mathcal{M}$ if each $\boldsymbol{D}_k$ is a complete subgraph of $\mathcal{M}$.

Bayesian Network Structure

Definition

A bayesian network structure $\mathcal{G}$ is a directed graph whose nodes represent random variables $X_1, \cdots, X_N$. Let $\text{Pa}^{\mathcal{G}}_{X_i}$ denote the parents of X_i in $\mathcal{G}$, and ND_{X_i} denote the variables in the graph that are not descendants of X_i. Then $\mathcal{G}$ encodes the following set of conditional independence assumptions, called the local independencies, and denoted by $\mathcal{I}_l(\mathcal{G})$:

$$\forall X_i : (X_i \perp\!\!\!\perp \text{ND}_{X_i} \mid \text{Pa}^{\mathcal{G}}_{X_i})$$

Definition

Let P be a distribution over $\mathcal{X}$. We define $\mathcal{I}(P)$ to be the set of independence assertions of the form $(\boldsymbol{X} \perp\!\!\!\perp \boldsymbol{Y} \mid \boldsymbol{Z})$ that hold in P.
We say that $\mathcal{G}$ is an I-map (independency map) for P if $\mathcal{I}_l(\mathcal{G}) \subseteq \mathcal{I}(P)$.

Bayesian Network Structure

Definition

Let $\mathcal{G}$ be a Bayesian network structure graph over the variables $X_1, \cdots, X_n$. We say that a distribution P over the same space factorizes ccording to $\mathcal{G}$ if P can be expressed as a product:

$$P(X_{,} \cdots , X_n) = \prod_{i=1}^{n} P(X_i \mid \mathrm{Pa}^{\mathcal{G}}_{X_i})$$

This equation is called the chain rule for the Bayesian networks. The individual factors $P(X_i \mid \mathrm{Pa}^{\mathcal{G}}_{X_i})$ are called conditional probability distributions (CPDs) or local probabilistic models.

Introduction
Graph Theory Primer
Bayesian Networks
Model Specification
Inference
Parameter Learning
Structure learning
Bayesian Networks Structures
References

Bayesian Networks

Definition

A bayesian network is a pair $\mathcal{B} = (\mathcal{G}, P)$ where P factorizes over $\mathcal{G}$, and where P is specified a set of CPDs associated with $\mathcal{G}$ nodes.

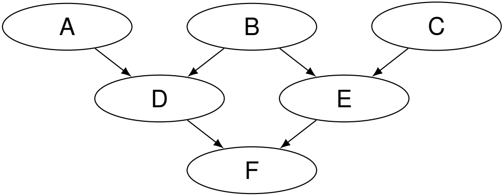
Theorem

Let $\mathcal{G}$ be a bayesian network structure over a set of random variables $\mathcal{X}$, and let P be a joint distribution over the same space. $\mathcal{G}$ is an I-map for P, if and only if P factorizes according to $\mathcal{G}$.

Proof.

See Daphne Koller and Nir Friedman *Probabilistic Graphical Models*[**?**], pages 62 and 63. □

Example



This BN factorizes $P(A, B, C, D, E, F)$ into the following list of factors:

$$P(A), P(B), P(C), P(D|A, B), P(E|B, C), P(F|D, E)$$

Key properties

- Nodes can be ordered such that parents come before children (topological ordering)
- Ordered Markov property:
  - a node only depends on its immediate parents, not on all predecessors in the ordering
- Markov Property: there are no direct dependencies in the system being modeled which are not already explicitly shown via arcs
  - Independence-maps: every independence suggested by the lack of an arc is real in the system
  - Whereas the independencies suggested by a lack of arcs are generally required to exist in the system being modeled, it is not generally required that the arcs in a BN correspond to real dependencies in the system.
  - it is necessary that the graph does not mislead us regarding independencies in the distribution.

Determining conditional independences from a DAG

How to read off conditional independencies from a graph.

- Given a Bayesian network and two variables X and Y
- Question:
  - Are X and Y independent ?
  - What are the (graph-theoretic) conditions under which X and Y are independent?
  - intuitive meaning of independence:
    - X and Y are dependent under some condition Z iff knowledge about one influences belief about the other under Z.
    - X and Y are independent under some condition Z iff knowledge about one does not influence belief about the other under Z

Determining conditional independences from a DAG

- Direct connection

$$X \longrightarrow Y$$

- Indirect connection
  - Causal trail

    $$X \longrightarrow Z \longrightarrow Y$$

    active if and only if Z is not observed. So, observing Z, X and Y are independent.
  - Evidential trail

    $$X \longleftarrow Z \longleftarrow Y$$

    active if and only if Z is not observed. So, observing Z, X and Y are independent.
  - Common cause

    $$X \longleftarrow Z \longrightarrow Y$$

    active if and only if Z is not observed. So, observing Z, X and Y are independent.
  - Common effect

    $$X \longrightarrow Z \longleftarrow Y$$

    active if and only if either Z or one of Z's descendants is observed..So, observing Z, X and Y are dependent.

d-Separation

Definition

Let $\mathcal{G}$ be a BN structure, and $X_1 \leftrightharpoons \cdots \leftrightharpoons X_n$ a trail in $\mathcal{G}$. Let $\boldsymbol{Z}$ be a subset of observed variables. The trail $X_1 \leftrightharpoons \cdots \leftrightharpoons X_n$ is **active** given $\boldsymbol{Z}$ if

- whenever we have a *v*-structure $X_{i-1} \longrightarrow X_i \longleftarrow X_{i+1}$, then X_i or one of its descendants are in $\boldsymbol{Z}$;
- no other node along the trail is in $\boldsymbol{Z}$

Definition

Let $\boldsymbol{X}$, $\boldsymbol{Y}$, $\boldsymbol{Z}$ be three sets of nodes in $\mathcal{G}$. We say that $\boldsymbol{X}$ and $\boldsymbol{Y}$ are d-separated given $\boldsymbol{Z}$, denoted d-sep$_{\mathcal{G}}(\boldsymbol{X}, \boldsymbol{Y} \mid \boldsymbol{Z})$, if there is no active trail between any node $X \in \boldsymbol{X}$ and $Y \in \boldsymbol{X}$ given $\boldsymbol{Z}$.
We use $\mathcal{I}(\mathcal{G})$ to denote the set of independencies that correspond to d-separation:

$$\mathcal{I}(\mathcal{G}) = \{(\boldsymbol{X} \perp\!\!\!\perp \boldsymbol{Y} \mid \boldsymbol{Z}) : \text{d-sep}_{\mathcal{G}}(\boldsymbol{X}, \boldsymbol{Y} \mid \boldsymbol{Z})\}$$

This set is also called the set of global Markov independencies

d-Separation

Theorem

For almost all distributions P that factorize over $\mathcal{G}$, that is, for all distributions except for a set of measure zero in the space of CPD parametrizations, we have that

$$\mathcal{I}(\mathcal{P}) = \mathcal{I}(\mathcal{G})$$

Proof.

See Daphne Koller and Nir Friedman *Probabilistic Graphical Models*[**?**], pages 73 and 74. □

Introduction
Graph Theory Primer
Bayesian Networks
Model Specification
Inference
Parameter Learning
Structure learning
Bayesian Networks Structures
References

Inference

- General form of query: $P(Q|E = e)$?
    - Q is a list of query variables, usually one.
    - E is a list of evidence variables, and e is the corresponding list observed values
    - Inference refers to the process of computing the answer to a query
- Given a Bayesian network and a random variable X, deciding whether $P(X = x) > 0$ is NP-hard.
- This implies that there is no general inference procedure that will work efficiently for all network configurations
- But for particular families of networks, inference can be done efficiently
- In other cases, instead of exact inference (computing the probabilities exactly) we will use approximate inference (computing the probabilities with reasonable precision)

Brute force inference algorithm

- Brute force algorithm for computing $P(Q|E=e)$ in a Bayesian network:
  - Get the joint probability distribution $P(X)$ over the set X of all variables by multiplying conditional probabilities.
  - Marginalize

$$P(Q,E) = \sum_{X-Q\cup E} P(X), P(E) = sum_Q P(Q,E)$$

  - Condition

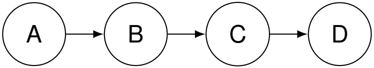
$$P(Q|E=e) = \frac{P(Q,E=e)}{P(E=e)}$$

- Not making use of the factorization , exponential complexity
- key question: how to leverage the factorization to avoid exponential complexity

Introduction
Graph Theory Primer
Bayesian Networks
Model Specification
Inference
Parameter Learning
Structure learning
Bayesian Networks Structures
References

Inference example

- Example Bayesian network of binary (Boolean) random variables



The joint distribution factorizes as
$P(A, B, C, D) = P(A)P(B|A)P(C|B)P(D|C)$

- Query: $P(D = 1)$?
- Computation

$$P(D=1) = \sum_{A,B,C} P(A, B, C, D = 1) \tag{1}$$

$$= \sum_{A,B,C} P(A)P(B|A)P(C|B)P(D = 1|C) \tag{2}$$

$$= \sum_{B,C} P(C|B)P(D = 1|C) \sum_{A} P(A)P(B|A) \tag{3}$$

$$= \sum_{C} P(D = 1|C) \sum_{C} P(C|B) \sum_{A} P(A)P(B|A) \tag{4}$$

Inference example

$$\begin{aligned}
P(D=1) \quad = \quad & P(A=1)P(B=1|A=1)P(C=1|B=1)P(D=1|C=1) \\
+ \quad & P(A=0)P(B=1|A=0)P(C=1|B=1)P(D=1|C=1) \\
+ \quad & P(A=1)P(B=0|A=1)P(C=1|B=0)P(D=1|C=1) \\
+ \quad & P(A=0)P(B=0|A=0)P(C=1|B=0)P(D=1|C=1) \\
+ \quad & P(A=1)P(B=1|A=1)P(C=0|B=1)P(D=1|C=0) \\
+ \quad & P(A=0)P(B=1|A=0)P(C=0|B=1)P(D=1|C=0) \\
+ \quad & P(A=1)P(B=0|A=1)P(C=0|B=0)P(D=1|C=0) \\
+ \quad & P(A=0)P(B=0|A=0)P(C=0|B=0)P(D=1|C=0)
\end{aligned}$$

$$\begin{aligned}
& P(A=1)P(B=0|A=1)P(C=1|B=0)P(D=1|C=1) \quad + \\
& P(A=0)P(B=0|A=0)P(C=1|B=0)P(D=1|C=1) \\
& = [P(A=1)P(B=0|A=1) + P(A=0)P(B=0|A=0)] \\
& \qquad \times P(C=1|B=0)P(D=1|C=1)
\end{aligned}$$

Eliminating a variable

- Suppose: $P(Z_1, Z_2, \cdots, Z_m) = f_1 \times f_2 \times \cdots \times f_n$
- Obtaining a factorization of $P(Z_2, \cdots, Z_m)$ could be done with much less computation:
  Procedure `eliminate`$(\mathcal{F}, Z)$:
  - Inputs: $\mathcal{F}$
  - Output: Another list of functions

  1. Remove from the $\mathcal{F}$ all the functions, say $f_1, \cdots, f_k$, that involve Z
  2. Compute new function $g = \prod_{i=1}^{k} f_i$
  3. Compute new function $h = \sum_Z g$
  4. Add the new function h to $\mathcal{F}$
  5. Return $\mathcal{F}$
- $\sum_Z \prod_{i=1}^{k} f_i$ can be much cheaper than $\sum_Z P(Z_1, Z_2, \cdots, Z_m)$

The Variable Elimination Algorithm

```
Procedure VE($\mathcal{F}, Q, E, e, \rho$)
  • Inputs:
      • $\mathcal{F}$ the list of CPTs in a BN
      • $Q$ a list of query variables
      • $E$ a list of observed variables and $e$ the observed values
      • $\rho$ ordering of variables not in $Q \cup E$ (Elimination ordering)
  • Output: $P(Q|E = e)$
1 While $\rho$ is not empty,
    1 Remove the first variable $Z$ from $\rho$
    2 Call eliminate($\mathcal{F}, Z$).
2 EndWhile
3 Set $h$= product of all the factors in $\mathcal{F}$
4 Instantiate observed variables in $h$ to their observed values
5 Return $\frac{h(Q)}{\sum_Q h(Q)}$
```

Approximate Inference using Sampling

Sampling from a Bayesian Network can be performed using

- Prior Sampling
- Rejection Sampling
- Likelihood weighting
- Gibbs Sampling

Prior Sampling

Given a topological ordering $X_1, X_2, \cdots, X_N$, prior sampling sample x_i from $P(X_i|\text{pa}(X_i))$, and return $x_1, x_2, \cdots, x_N$

Gibbs Sampling

Let $P(X_1, X_2, \cdots, X_n | e_1, \cdots, e_m)$ denote the joint distribution of a set of random variables $(X_1, X_2, \cdots, X_n)$ conditioned on a set of evidence variables $(e_1, \cdots, e_m)$.

Gibbs sampling is an algorithm to generate a sequence of samples from such a joint probability distribution. A Gibbs sampler runs a Markov chain on $(X_1, X_2, \cdots, X_n)$

For convenience of notation, we denote the set $(X_1, X_2, \cdots, X_{i-1}, X_{i+1}, \cdots, X_n)$ as X_{-i}, and $\boldsymbol{e} = (e_1, \cdots, e_m)$.

A variation of Gibbs sampler

1. Initialize:
   -
   -
2. For $t = 1, 2, \cdots$
   - Pick an index i, $1 \leq i \leq n$ uniformly at random
   - Sample x_i from $P(X_i | x_{-i}^{(t-1)}, \boldsymbol{e})$
   - Let

Learning

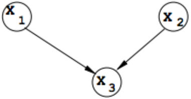
- D variables: $X_1, \cdots, X_D$
- Number of states of X_d: $1, 2, \cdots, r_d = |\Omega_{X_d}|$
- Number of configurations of parents of X_d: $1, 2, \cdots, q_d = |\Omega_{pa(X_d)}|$
- Parameters to be estimated:

$$\theta_{djk} = P(X_d = j | pa(X_d) = k),$$

$$d = 1, \cdots, D; j = 1, \cdots, r_d; k = 1, \cdots, q_d$$

Learning

- Example: Consider the Bayesian network shown below. Assume all variables are binary, taking values 1 and 2.



$$
\begin{aligned}
\theta_{111} &= P(X_1{=}1), \theta_{121} = P(X_1{=}2) \\
\theta_{211} &= P(X_2{=}1), \theta_{221} = P(X_2{=}2) \\
pa(X_3) = 1 : \theta_{311} &= P(X_3{=}1|X_1 = 1, X_2 = 1), \theta_{321} = P(X_3{=}2|X_1 = 1, X_2 = 1) \\
pa(X_3) = 2 : \theta_{312} &= P(X_3{=}1|X_1 = 1, X_2 = 2), \theta_{322} = P(X_3{=}2|X_1 = 1, X_2 = 2) \\
pa(X_3) = 3 : \theta_{313} &= P(X_3{=}1|X_1 = 2, X_2 = 1), \theta_{323} = P(X_3{=}2|X_1 = 2, X_2 = 1) \\
pa(X_3) = 4 : \theta_{314} &= P(X_3{=}1|X_1 = 2, X_2 = 2), \theta_{324} = P(X_3{=}2|X_1 = 2, X_2 = 2)
\end{aligned}
$$

Learning

- Log likelihood:

$$
\begin{aligned}
l(\theta|\mathcal{D}) &= \log \mathcal{L}(\theta|\mathcal{D}) \\
&= \log \prod_i P(\boldsymbol{x}_i|\theta) \\
&= \sum_i \log P(\boldsymbol{x}_i|\theta)
\end{aligned}
$$

- Let

$$
\chi(d, j, k : \boldsymbol{x}_i) = \begin{cases} 1, & \text{if } X_d = j, pa(X_d) = k \text{ in } \boldsymbol{x}_i \\ 0 \text{ otherwise} \end{cases}
$$

- In general

$$
\log P(\boldsymbol{x}_i|\theta) = \sum_{djk} \chi(d, j, k : \boldsymbol{x}_i) \log \theta_{djk}
$$

- define

$$
m_{djk} = \sum_i \chi(d, j, k : \boldsymbol{x})
$$

It is the number of data cases where $X_d = j$ and $pa(X_d) = k$.

Learning

- Then

$$
\begin{aligned}
l(\theta, \mathcal{D}) &= \sum_i \log P(\boldsymbol{x}_i|\theta) \\
&= \sum_i \sum_{d,j,k} \chi(d,j,k : \boldsymbol{x}_i) \log \theta_{djk} \\
&= \sum_{d,j,k} \sum_i \chi(d,j,k : \boldsymbol{x}_i) \log \theta_{djk} \\
&= \sum_{d,j,k} m_{djk} \log \theta_{djk}
\end{aligned}
$$

- the MLE estimate for $\theta_{djk} = P(X_d = j|pa(X_D) = k)$ is :

$$
\theta^*_{djk} = \frac{\text{number of cases where} X_d = j \text{ and } pa(X_d) = k}{\text{number of cases where} pa(X_d) = k}
$$

Structure learning

Structure learning for discrete, fully observed networks:

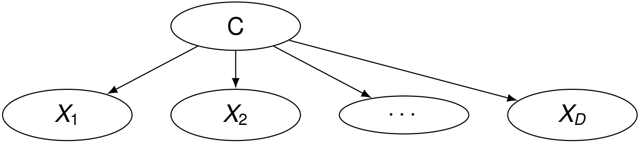
- Score-based structure estimation (BIC/BDeu/K2 score; exhaustive search, hill climb/tabu search), are applications of general optimisation techniques; each candidate DAG is assigned a network score maximise as the objective function.
- Constraint-based structure estimation (PC) s identify conditional independence constraints with statistical tests, and link nodes that are not found to be independent.
- Hybrid structure estimation (MMHC): have a restrict phase implementing a constraint-based strategy to reduce the space of candidate DAGs; and a maximise phase implementing a score-based strategy to find the optimal DAG in the restricted space.

Constraint-based structure learning

Constraint based learning algorithms typically have two phases:

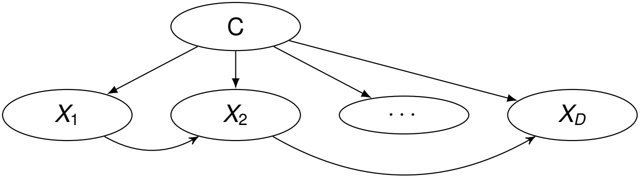
- a (conditional) independence test phase and
- an edge orientation phase

Naive Bayes



$$P(C|X_1, \cdots, X_D) = P(C)\frac{\prod_{d=1}^{D} P(X_d|C)}{P(X_1, \cdots, X_D)} \tag{5}$$

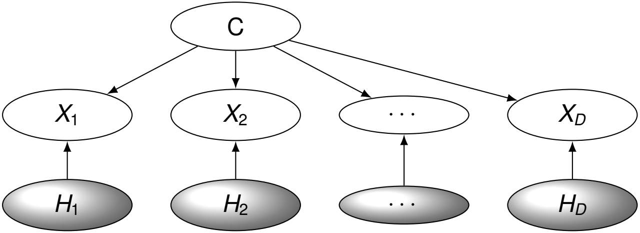
Tree Augmented Naive Bayes



$$P(X_1, \cdots, X_D) = \prod_{t=1}^{D} P(X_t|\text{pa}(X_t)) \tag{6}$$

- pa(X_t) is a single parent
- The tree is learned using a minimum spanning tree (MST) algorithm (Prim or Kruskal)
- The MST is computed from the fully connected graph where the nodes are $X_1, \cdots, X_D$ and the weights are the mutual information between the nodes.

Hidden Naive Bayes



$$P(X_1, X_2, \cdots, X_D, C) = P(C) \prod_{d=1}^{D} P(X_d | H_d, C) \tag{7}$$

where

$$P(X_d | H_d, C) = \sum_{j=1, j \neq d}^{D} w_{dj} \times P(X_d | X_j, C)$$

and

$$w_{dj} = \frac{I_P(X_d, X_j | C)}{\sum_{k=1, k \neq d}^{D} I_P(X_d, X_k | C)}$$

and

$$I_P(X, Y | Z) = \sum_{x,y,z} P(x, y, z) \log \frac{P(x, y | z)}{P(x | z) P(y | z)}$$

References

1. Christopher M. Bishop, *Pattern Recognition and Machine Learning*, 4Myeloma Press, 2010.
2. Kevin P. Murphy, *Machine Learning, A Probabilistic Perspective*, Oxford University Press, 2007.
3. Daphne Koller and Nir Friedman, *Probabilistic Graphical Models. Principles and Techniques*, MIT Press, 2009 **??**.
4. David Barber, *Bayesian Reasoning and Machine Learning*, Cambridge University Press, 2012.