Decision Making Under Uncertainty Ch2-Ch3 Stanford AA228 course notes – Philippe Weingertner

Probabilistic models

Probability and BN representation

- Definition of Conditional Probability: $P(A \mid B) = \frac{P(A,B)}{P(B)}$
- Law of Total Probability: $P(A) = \sum_{B \in \mathscr{B}} P(A,B) = \sum_{B \in \mathscr{B}} P(A \mid B)P(B)$
- LTP with CP: $P(A \mid C) = \sum_{B \in \mathscr{B}} P(A,B \mid C) = \sum_{B \in \mathscr{B}} P(A \mid B,C)P(B \mid C)$
- Bayes Rule $P(A \mid B) = \frac{P(B|A)P(A)}{P(B)}$
- $\phi(x) = \frac{1}{\sqrt{2\pi}} \exp\left(-x^2/2\right) \quad \Phi(x) = \int_{-\infty}^{x} \phi(x)dx$
- $p(\omega) = \mathscr{N}(\omega \mid \mu, \sigma^2) = \frac{1}{\sigma\sqrt{2\pi}} e^{-\frac{1}{2}\left(\frac{\omega-\mu}{\sigma}\right)^2} = \frac{1}{\sigma}\phi\left(\frac{\omega-\mu}{\sigma}\right)$
- Truncated Gaussian $\mathscr{N}(\omega \mid \mu, \sigma^2, a, b) = \frac{\mathscr{N}(\omega|\mu,\sigma^2)}{\Phi\left(\frac{b-\mu}{\sigma}\right)-\Phi\left(\frac{a-\mu}{\sigma}\right)}$
- $\mathscr{N}(s \mid \mu, \Sigma) = \frac{1}{(2\pi)^{k/2}|\Sigma|^{1/2}} \exp\left(-\frac{1}{2}(s-\mu)^T \Sigma^{-1}(s-\mu)\right)$
- BN is a compact representation of joint distribution
- $P(E \mid B,S)$ has $(n_E - 1) \times n_B \times n_S$ independant params
- BN chain rule: $P(x_1, \ldots, x_n) = \prod_{i=1}^{n} P(x_i \mid pa_{x_i})$

Conditional Independence

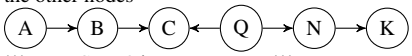
$(A \perp B \mid C) \iff P(A,B \mid C) = P(A \mid C)P(B \mid C)$
$(A \perp B \mid C) \iff P(A \mid C) = P(A \mid B,C)$
chain $X \to Y \to Z$; **fork** $X \leftarrow Y \to Z$
inverted fork or **v-structure** $X \to Y \leftarrow Z$
v-structure $\Longrightarrow$ *check evidence on Y descendants*

on path of influence	chain, fork	v-structure
evidence	**blocking**	**enabling**

$(A \perp B \mid \mathscr{C})$ if all paths between A and B are d-seperated by $\mathscr{C}$
Of course $A \to B$ means no possible d-sep between A, B

Markov Blanket of Q

$\mathscr{C}_B$: the min set of nodes that d-separate a node from all other
If we observe all nodes in $\mathscr{C}_{\mathscr{B}}$ then Q is condit. indep. of the other nodes



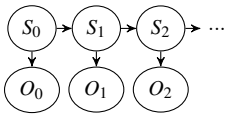
!!! $B \to C \leftarrow Q$ is a v-structure !!!
Evidence in C enables the path of influence
Markov blanket of Q: N, C because they are directly connected to Q
But then, because of the v-structure, we enable a path of influence with evidence in C
So we have to add an evidence in B to block the path of influence
$\Longrightarrow$ Markov Blanket of Q is N, C, B

Hybrid BN and Temporal Models

Hybrid BN example: $P(W)P(M)P(C \mid W,M)P(D,C)$ with W, C continuous

- linear gaussian: $P(c \mid \omega, m) = \begin{cases} \mathscr{N}(c|\theta_1\omega + \theta_2, \theta_3) & m^0 \\ \mathscr{N}(c|\theta_4\omega + \theta_5, \theta_6) & m^1 \end{cases}$
- logit model: $P(d^1 \mid c) = \frac{1}{1+exp\left(-2\frac{c-\theta_7}{\theta_8}\right)}$

Aircraft HMM example: $s_t = (h, \dot{h})$



$P(s_t \mid s_{t-1}) = \mathscr{N}(s_t \mid Ms_{t-1} + b, \Sigma)$
$P(o_t \mid s_t) = \mathscr{N}(o_t \mid [1 \quad 0]s_t, \Sigma)$
S_0 has less parameters than S_i for i$\geq$ 1

Inference

To find a distribution over some unobserved variables given a set of observed variables. It might be used when the structure and parameters of the Bayesian network are known: e.g. classification tasks, where you want to infer a class given a set of observations, Kalman filtering ...

Variables type: query (or unobserved), evidence (or observed) and hidden

Inference for Classification

Predict a class given some observations
Radar ex.: observe speed + heading fluctuations

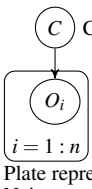
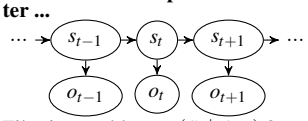


Plate representation of a naive Bayes model
Naive: assume condit. indep. between all observations given our class
Prior $P(C)$, Class-conditional distribution $P(O_i \mid C)$, Posterior $P(C \mid O_{1:n})$
$P(c \mid o_{1:n}) = \frac{P(c,o_{1:n})}{P(o_{1:n})} \propto P(c)\prod_i P(o_i \mid c)$ by BN chain rule
Normalized such that $\sum_c P(c \mid o_{1:n}) = 1$

Inference for Temporal Models: HMM, Kalman Filter ...



Filtering problem: $P(S_t \mid O_{0:t})$?
By Bayes rule
$P(s_t \mid o_{0:t}) = P(s_t \mid o_t, o_{0:t-1}) \propto P(o_t \mid s_t, o_{0:t-1})P(s_t \mid o_{0:t-1})$
By d-sep $(O_t \perp O_{0:t-1} \mid S_t)$
$P(o_t \mid s_t, o_{0:t-1}) = P(o_t \mid s_t)$
$P(s_t \mid o_{0:t}) \propto P(o_t \mid s_t)P(s_t \mid o_{0:t-1})$
By the Law of Total Probability over s_{t-1}
$P(s_t \mid o_{0:t}) \propto P(o_t \mid s_t)\sum_{s_{t-1}} P(s_t, s_{t-1} \mid o_{0:t-1})$
By the definition of conditional probability
$P(s_t \mid o_{0:t}) \propto P(o_t \mid s_t)\sum_{s_{t-1}} P(s_t \mid s_{t-1}, o_{0:t-1})P(s_{t-1} \mid o_{0:t-1})$
By d-sep $(S_t \perp O_{0:t-1} \mid S_{t-1})$
$P(s_t \mid s_{t-1}, o_{0:t-1}) = P(s_t \mid s_{t-1})$
$\mathbf{P(s_t \mid o_{0:t}) \propto P(o_t \mid s_t)\sum_{s_{t-1}} P(s_t \mid s_{t-1})P(s_{t-1} \mid o_{0:t-1})}$
With continuous variables replace $\sum$ with $\int$
Our known model is:

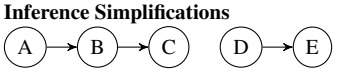
- Observation model: $P(o_t \mid s_t)$
- State transition model: $P(s_t \mid s_{t-1})$

So we get a **recursive formula** about our belief $b_t(s)$ based on all possible previous states belief $b_{t-1}(s')$
Cf algorithm below with stationary state transition distrib and stationary observation distrib (just to reduce typing subscripts...)

Recursive Bayesian Estimation

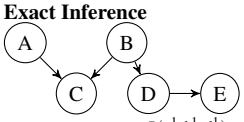
function RECURSIVEBAYESIANESTIMATION
  $b_0(s) \leftarrow P(o_0 \mid s)P(s_0)$ for all s
  Normalize b_0
  for $t \leftarrow 1$ to ∞ **do**
    $b_t(s) \leftarrow P(o_t \mid s)\sum_{s'} P(s \mid s')b_{t-1}(s')$ for all s
    Normalize b_t

Inference Simplifications



Of course A, B, C independant from D, E

Exact Inference



$P(a^1 \mid b^1, d^1) = \frac{P(a^1,b^1,d^1)}{P(b^1,d^1)}$ by definiton of Cond Prob
$P(a^1, b^1, d^1) = \sum_c \sum_e P(a^1, b^1, c, d^1, e)$ by Law of Tot Prob
$P(a^1, b^1, c, d^1, e) = P(a^1)\mathbf{P(b^1)P(d^1 \mid b^1)}P(c \mid a^1, b^1)P(e \mid d)$ by BN
We sum over unobserved variables
4 summations for numerator
8 summations for denominator (but **simplification** here)

Exact Inference with Variables Elimination

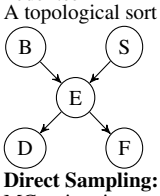
Idea: convert all your conditional proba tables into regular tables
Question: $P(B \mid d^1, e^1)$ on previous BN ? B: query variable
Tables: $T_1(A), T_2(B), T_3(A,B,C), T_4(B,D), T_5(D,E)$
Step1: use evidences and drop all the rows that are inconsistent with the evidences. Example with d^1, e^1
$\Longrightarrow T_1(A), T_2(B), T_3(A,B,C), T_4(B)$ (NB: T_5 is just a number now)
Step2: eliminate hidden variables. Choose an order e.g. A, C.
Choosing an optimal ordering (impacts #computations) is NP-hard
Eliminate A: $T_5(B,C) = \sum_a T_1(a)T_3(a,B,C)$ we get a new table where the per row probability is updated (by product and sum of previous probabilities)
$\Longrightarrow T_2(B), T_4(B), T_5(B,C)$
Eliminate C: $T_6(B) = \sum_c T_5(B,C)$ we get a new table where the per row probability is updated (by sum of previous probabilities)
$\Longrightarrow T_2(B), T_4(B), T_6(B)$
Step3: multiply and normalize the resulting table to get $T(B)$

Complexity of Exact Inference

P: pbs can be solved in polynomial time
complexity $\propto n^c$ with n size of inputs and c a constant
NP: pbs whose solutions can be verified in polynomial time
NP-complete: all pbs X in NP for which it is possible to reduce any other NP pb Y to X in polynomial time
Every NP pb can be reduced (transformed) to 3SAT
NP-hard: a pb X is NP-hard, if there is an NP-complete pb Y such that Y is reducible to X in polynomial time
3SAT can be transformed to BN inference => BN inference is at least as hard as 3SAT
3SAT problem is NP-complete and we can easily show that inference in Bayesian networks is at least this hard.
$\Longrightarrow$ **Inference in Bayesian networks is NP-hard**
$\Longrightarrow$ do not waste time looking for an efficient exact inference algo and focus on approximate inference methods

Approximate Inference

Topological sort: of nodes in a DAG is an ordered list s.t. if there is an edge $A \to B$, then A comes before B in the list: the parents of some node A always appear before A)
Once we have a topological sort, we can begin sampling from the conditional probability distributions: we sample from the parents of a node before sampling from the node itself
A topological sort always exist but may not be unique



Direct Sampling:
MC estimation: sample from the distribution encoded by the joint distrib
Step1: topological sort of the DAG
Step2: iterate through the topological sort and sample from cond probs
Many, many times ... Need lots of samples to get high quality estimate

B	S	E	D	F
1	0	1	1	0
1	1	0	0	0
1	0	1	1	0
0	0	1	1	0

$\Longrightarrow P(b^1) = \frac{3}{4}, P(b^1 \mid d^1) = \frac{2}{3}, \ldots$

Likelihood Weighted Sampling:
How to use efficiently evidences ? e.g. we observed f^1
Fix F and weight the samples appropriately

B	S	E	D	F	ω
1	0	1	1	**1**	$P(f^1 \mid e^1) = \ldots$
0	1	1	1	**1**	$P(f^1 \mid e^1) = \ldots$
0	1	0	1	**1**	$P(f^1 \mid e^0) = \ldots$

$\Longrightarrow P(b^0 \mid f^1) = \frac{P(f^1|e^1)+P(f^1|e^0)}{P(f^1|e^1)+P(f^1|e^1)+P(f^1|e^0)}$
To find out $P(b^0 \mid f^1, d^0)$, fix f^1 and d^0 in the table and replace $P(f^1 \mid e^n)$ with $P(f^1 \mid e^n)P(d^0 \mid e^n)$ in the last column
It has the advantage of not wasting samples: important when dealing with rare events

Gibbs Sampling:
MCMC idea: $\mathbb{E}[f(s)]_{\mathscr{P}} \approx \frac{1}{N}\sum_N f(s^{(i)})$
Gibbs idea: generate posterior samples by sweeping through each var to sample from its cond distrib with the remaining vars fixed to their current values
Any ordering can be used, next samples depends probabistically on the current sample BUT CV at the limit
X_1, X_2, X_3 with prio $x_1^{(0)}, x_2^{(0)}, x_3^{(0)}$
$x_1^{(i)} \sim P(X_1 = x_1 \mid X_2 = x_2^{(i-1)}, X_3 = x_3^{(i-1)})$
$x_2^{(i)} \sim P(X_2 = x_2 \mid X_1 = x_1^{(i)}, X_3 = x_3^{(i-1)})$
$x_3^{(i)} \sim P(X_3 = x_3 \mid X_1 = x_1^{(i)}, X_2 = x_2^{(i)})$
Two problems:
- Burn-in period before CV: $\Longrightarrow$discard early samples
- Samples are correlated: $\Longrightarrow$thin the samples by only keeping every k^{th}

Parameter Learning

To find the parameters of the distributions that determine the conditional probabilities between variables. Parameter learning might be used if the structure of the Bayesian network is known, but the conditional probabilities are unknown.

Maximum LLH parameter learning
$\hat{\theta} = \underset{a}{argmax}\, P(D \mid \theta)$
With a binomial distribution (aka pilou-face distrib)
$P(D \mid \theta) = \frac{n!}{m!(n-m)!}\theta^m(1-\theta)^{n-m} \Longrightarrow \ell(\theta) = ln\left(\theta^m(1-\theta)^{n-m}\right)$
Then just solve $\frac{\partial l}{\partial \theta} = 0$ to get $\hat{\theta} = \frac{m}{n}$
PB1: it is a point estimate. No pdf ...
PB2: with few samples can lead to nonsense (rare but dangerous events may be ignored). We want to take into account the number of samples ...
With a Gaussian we would maximize:
$\ell(\mu, \sigma^2) \propto -nln\sigma - \frac{\sum_i (v_i - \mu)^2}{2\sigma^2}$ over the collected points v_i

Bayesian parameter learning
With X a 2-ary R.V.: $P(x^1) = \theta$
$prior = Beta(\theta \mid \alpha, \beta)$ (NB: $Beta(1,1) = \mathscr{U}[0,1]$)
We sample m ones and $n-m$ zeros
$\Longrightarrow posterior = Beta(\alpha + m, \beta + n - m)$
When m, n increase then influence of prior decreases
$Beta(6,6)$ is a 50/50 distrib, $Beta(60,60)$ also but more peaky at the mean
$Beta(2,6)$ (closer to 0) is flipped vs $Beta(6,2)$ (closer to 1)
With X a n-ary R.V.: $P(x^i) = \theta_i$
$prior = Dir(\theta_{1:n} \mid \alpha_{1:n})$ (NB: $Dir(1_{1:n}) = \mathscr{U}$)
We sample m_1 ones, m_2 twos, ..., m_n n-s
$\Longrightarrow posterior = Dir(\theta_{1:n} \mid \alpha_1 + m_1, \ldots, \alpha_n + m_n)$
If $Dir(1,1,1,1,1,1)$ uniform prior $\Rightarrow$ no idea for die
If $Dir(100,100,100,100,100,100)$ $\Rightarrow$ pretty sure fair die
Beta and Dirichlet distributions are conjugate-priors easy to update via Baye's rule
BPL Advantages:
- With the prior, we can leverage expert knowledge (especially useful when data is sparse)
- By computing a posterior distribution, we not only obtain an estimate for the parameters, but also a confidence in our estimate.

BPL Drawbacks:
- We need to come up with a prior distribution. Depending on the prior, the posterior will be affected. This might lead to the use of non-informative priors in the hope of letting the data drive the posterior through the likelihood function, which in turn can lead to results that are similar to using maximum likelihood.
- Computing the posterior can only be done in some cases, but in general, we have to use numerical integration or Monte Carlo methods.

Nonparametric learning

The number of parameters scale with the amount of data
Given observations $o_{1:n}$, kernel density estimation represents the density as follows: $p(x) = \frac{1}{n}\sum_i K(x - o_i)$
Where K is a kernel function which integrates to 1
It assigns greater density to values near the observed data points
Common kernel: zero-mean Gaussian distribution
Bandwidth: means standard deviation

Structure Learning

To find the (unknown) structure of the Bayesian network. This could arise if you had a large dataset and wanted to find the relationships between variables.

To generate a model we can use for simulation

Bayesian structure scoring and graph search
Maximum likelihood approach: finding G that maximizes $P(G \mid D)$, where D represents the available data.
The Bayesian score metric, a log likelihood, is used to evaluate how good a Bayesian Network Graph fits the observed data.
$ln\, P(G \mid D) = ln\, P(G) + \sum_{i=1}^{n}\sum_{j=1}^{q_i} ln\left(\frac{\Gamma(\alpha_{ij0})}{\Gamma(\alpha_{ij0}+m_{ij0})}\right) + \sum_{k=1}^{r_i} ln\left(\frac{\Gamma(\alpha_{ijk}+m_{ijk})}{\Gamma(\alpha_{ijk})}\right)$ (NB: $\Gamma(n) = (n-1)!$)
The Bayesian score is not convex
Graph search: K2 search + local search (hill climbing)

Markov Equivalence Classes
2 Graphs are Markov equivalent iff they have:
- same edges without regard to direction
- same immoral v-structures i.e. $X \to Y \leftarrow Z$ with X and Z not directly connected

Markov equiv class: set of all DAG that are Markov equiv. Bscores within a class are very close if not equal.
Instead of searching the space of DAG, search the space of Markov equiv classes (smaller)
Achtung with **PDAG** (with undirected edges): we search equivalents DAGs by directing the undirected edges without changing already directed edges !!! (cf exercise 19) But v-structures count is based on original PDAG !!!

Decision Problems

Rationale DM is reasoning about uncertainty and objectives (preferences)

Utility Theory

Constraints on Rational Preferences
Completeness: we can compare them $A \succ B, A \prec B, A \sim B$
Transitivity: $A \succeq B, B \succeq C \Rightarrow A \succeq C$
Continuity: $A \succeq C \succeq B \Rightarrow \exists p$ s.t. $[A:p; B:1-p] \sim C$
translated to $\boxed{pU(A) + (1-p)U(B) = U(C)}$
Independence: $A \succ B \Rightarrow \forall (C,p), [A:p; C:1-p] \succ [B:p; C:1-p]$

We define U a utility related to preferences (or objectives or costs) so that it follows these 4 axioms
$\Longrightarrow$ utility of a lottery is
$U([S_1:p_1; \ldots; S_n:p_n]) = \sum_{i=1}^{n} p_i U(S_i)$

Maximum Expected Utility Principle
$EU(a \mid o) = \sum_{s'} P(s' \mid a, o)U(s')$
A rational agent chooses $a^* = \underset{a}{argmax}\, EU(a \mid o)$

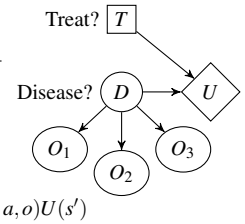
Utility elicitation: how to infer U from humans

Multiple Variables: under some assumptions
$U(x_{1:n}) = \sum_i U(x_i)$

BN + Utility $\Longrightarrow$ Decision Network

Evaluating Decision Networks

T	D	U(T,D)
0	0	0
0	1	-10
1	0	-1
1	1	-1



$EU(a \mid o) = \sum_{s'} P(s' \mid a, o)U(s')$
where s' represents an instanciation of the nodes in the decision network
we begin by instantiating the action nodes and observed chance nodes and we sum over the unobserved variables
$EU(t^1 \mid o_1^1) = \sum_{o_3}\sum_{o_2}\sum_{d} P(d, o_2, o_3 \mid t^1, o_1^1)U(t^1, d, o_1^1, o_2, o_3)$
$EU(t^1 \mid o_1^1) = \sum_d \sum_{o_2}\sum_{o_3} P(d, o_2, o_3 \mid t^1, o_1^1)U(t^1, d)$ per BN
$EU(t^1 \mid o_1^1) = \sum_d P(d \mid t^1, o_1^1)U(t^1, d)$ by removing marginalization
From there, any of the exact or approx inference method can be used to evaluate $P(d \mid t^1, o_1^1)$
$\Longrightarrow$**Inference pb: what is the distribution of the variables that are parents to the utility node ?**
To decide whether to apply a treatment compare $EU(t^1 \mid o_1^1)$ vs $EU(t^0 \mid o_1^1)$
Variant: remove action and chance nodes from decision networks if they have no children (as per conditional, informational or functional edges)

Value Of Information
$VOI(O' \mid o) = (\sum_{o'} P(o' \mid o)EU^*(o, o')) - EU^*(o) \geq 0$
It is the increase of Expected Value gained by using an Observation variable to make a better decision

Game Theory
Strategy can be pure (deterministic) or mixte (stochastic)
Strategy profile: $S_{1:n}$assignment of strategies to all n agents
Utility of agent i: $U_i(s_{1:n}) = U_i(s_i, s_{-i})$
Dominant strategy: single best response s_i^* whatever the strategy of others
Dominant strategy equilibrium: when all agents have a dominant strategy, it is their combination
Prisonner's dilemna:

	T	R
T	-5,-5	0,-10
R	-10,0	-1,-1

Nash Equilibrium: a strategy profile (s_i, s_{-i}) where no agent can do better by **unilaterally** changing strategy
Every game has at least 1 Nash equilibrium.
Collision avoidance game:

	Climb	Desc
Climb	-5,-5	-1,0
Desc	0,-1	-4,-4

PB: it is not reflective of how humans behave

Behavioral Game Theory
PBs with Nash Equilibrium:
- Often multiple NEs
- Difficult to compute for humans
- Opponents might not compute NE; even if you do

Logit level-k model: is more representative of humans behavior
- Humans are more likely to make errors when they are costly
- Humans reasoning is limited in strategic look-ahead

2 parameters:
- Precision: $\lambda \geq 0$ sensitivity to utility differences (0 is insensitive)
- Depth: $k > 0$ of rationality

Level-0: selects actions randomly
Level-1: assume opponents are level-0 and selects actions according to softmax (or logit) distribution $P(a_i) \propto e^{\lambda U_i(a_i, s_{-i})}$ where here s_{-i} is just uniform level-0
Level-k: assume opponents are level-k-1 ...
The parameters k, λ can be learned from data by maximising likelihood $P(Model \mid Data)$
Logit level-k model: collect a bunch of data and fits the model
Nash Equilibrium: no data needed, you have to work out the solution (but no known polynomial algo to find it)

Traveller's dilemna
2 suitcases. Cost has to be estimated $\in [2\$, 100\$]$
$U_i(a_i, a_{-i}) = \begin{cases} a_i & \text{if } a_i = a_{-i} \\ a_i + 2 & \text{if } a_i < a_{-i} \\ a_i - 2 & \text{if } a_i > a_{-i} \end{cases}$
$[100\$, 100\$]$ is that optimal choice ? If opponent puts 100$ I can earn more money by putting 99$ and so on ... down to 2$
NE is at $[2\$, 2\$]$ just check the Utility matrix. If I know you are going to play $2 I will stick to my choice. No better action can be deduced by knowing opponent action in this state. In all other cases of the matrix, I would change my choice.
$\lambda = 0.3, k = 0$: uniform between $[2\$, 100\$]$
$\lambda = 0.3, k = 1$: mostly between $[70\$, 100\$]$
$\lambda = 0.3, k = 2$: mostly between $[80\$, 100\$]$
$\lambda = 0.3, k = 3$: mostly between $[80\$, 100\$]$ but more concentrated at 90$. Close to human behaviors.
$\lambda = 0.5$ more precision $\Longrightarrow$ more spiky. λ is a precision parameter that controls the variance on the behavior.

Useful Functions

Gamma function
Real-valued generalization of the factorial.
$\Gamma(z) = \int_0^\infty x^{z-1}e^{-x}dx \Longrightarrow \Gamma(z+1) = z\Gamma(z)$
If $n \in \mathbb{N}$ then $\Gamma(n) = (n-1)!$
Beta distribution
$f(x; \alpha, \beta) = \frac{\Gamma(\alpha+\beta)}{\Gamma(\alpha)\Gamma(\beta)}x^{\alpha-1}(1-x)^{\beta-1}$