Two Key Components in Probability

- Probability Distribution Model
  - Real World: possible worlds, atomic events, samples
  - Our Mind: Variables, Value assignments ("entailment")
- Inferences that can be made from the model
  - Sum rule
  - Product rule
  - Conditional
  - Marginalization
  - Normalization

Subjective (Bayesian) Probability: Probability is not in the real world, it is in your head
Objective Probability: Probability is in the real world
Fully Joint Probability Distribution: Know everything about the world, can tell the most likely world
Propositions: Events where the proposition is true

- Rule (1) Sum Rule
  - **P(A|B) + P(~A|B) = 1**
- Rule (2) Product Rule for P(A^B)
  - $P(A \wedge B) = P(A|B)P(B) = P(B|A)P(A)$
  - Or more general: $P(AB|C) = P(A|C)\,P(B|AC) = P(B|C)\,P(A|BC)$

Remember these two rules!

$$P(A|B) = \frac{P(A \wedge B)}{P(B)} \text{ if } P(B) \neq 0$$

Product rule $P(A \wedge B) = P(A|B)P(B) = P(B|A)P(A)$

$$\Rightarrow \text{Bayes' rule } P(A|B) = \frac{P(B|A)P(A)}{P(B)}$$

- Probability is a rigorous formalism for uncertain knowledge
  - Conditional probabilities enable reasoning with uncertain evidence
- A *full joint probability distribution* specifies the probability of every *atomic event*
  - Queries answerable by summing over probabilities of atomic events
- Bayesian theorem/rule provides the basis for the most modern diagnostic reasoning in AI
  - Converts uncertain causal information into diagnostic conclusions
- For nontrivial domains, we must find a way to reduce the size of the joint distribution
  - *(Conditional) independence* provides the tools (next lecture)

- Models with Actions and Sensors (ALFE 4-5) (aka POMDP)
  - Slides 1-20 are essential for you to understand the concepts
  - The rest will follow naturally if you do
- Markov Chains
  - No observation, no explicit actions, transit randomly
- Hidden Markov Model
  - No explicit actions, state transit randomly
- Dynamic Bayesian Networks
  - No explicit actions, States are Bayesian Networks
- Continuous State Model
  - No explicit actions, States are continuous
- POMDP
  - Discrete states with probabilistic actions, sensors, & transitions

Forward Procedure, Backward Procedure, Viterbi Algorithm
Bayesian Networks:

- Temporal models use states, transitions, sensors
  - Transitions may be related to agent's actions, or spontaneous
- Markov assumptions and stationarity assumption, so we need
  - transition model $P(X_t \mid X_{t-1})$ or $P(X_t \mid X_{t-1}, a_{t-1})$
  - sensor model $P(E_t \mid X_t)$
- Tasks: filtering, prediction, smoothing, most likely seq
  - all done recursively with constant cost per time step
- Types of models
  - HMM have a single discrete state variable
  - Dynamic Bayes nets subsume HMMs; exact update intractable
  - Other models may have internal structure driven by actions

Two random variables A B are (absolutely) independent iff

$$P(A|B) = P(A)$$

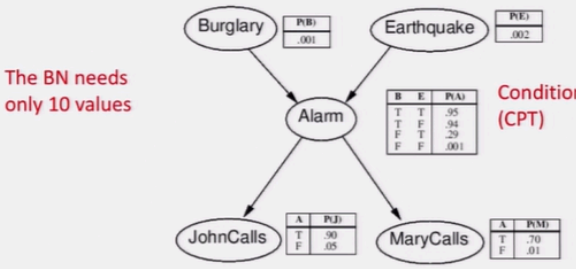
or $$P(A, B) = P(A|B)P(B) = P(A)P(B)$$

Example

I'm 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? The Full Joint Distribution needs 2^5 = 32 values

Variables: *Burglar*, *Earthquake*, *Alarm*, *JohnCalls*, *MaryCalls*

Network topology reflects "causal" knowledge:



The BN needs only 10 values

Conditional Probability Table (CPT)

k=2
d=2
n=5

Note: $\leq k$ parents $\Rightarrow O(d^k n)$ numbers vs. $O(d^n)$

Causal Reasoning:

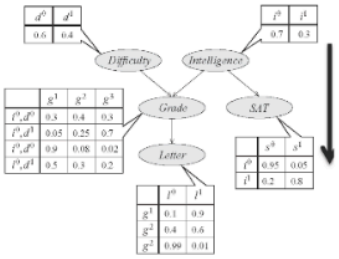
Causal Reasoning

1. How likely is George to get a strong letter (knowing nothing else)?
   - $P(l^1)=0.502$
   - Obtained by summing-out other variables in joint distribution
2. But George is not so intelligent (i^0)
   - $P(l^1|i^0)=0.389$ =P($l^1 i^0$)/P(i^0)
3. Next we find out ECON101 is easy (d^0)
   - $P(l^1|i^0, d^0)=0.513$ =P($l^1 i^0 d^0$)/P($i^0 d^0$)

Observe how probabilities change as evidence is obtained

Try it yourself:

$$P(D,I,G,S,l^1) = \sum_{D,I,G,S} P(D)P(I)P(G|D,I)P(S|I)P(l^1|G)$$

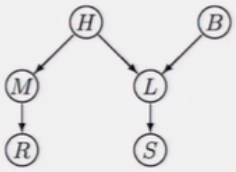


Query is Example of Causal Reasoning:

Predicting downstream effects of factors such as intelligence

Evidential Reasoning:

Evidential Reasoning Example



H = hardware problem
B = bug in lab
M = mailer is running
L = lab is running
R = received email
S = problem solved

Brute force calculation of $P(H \mid E)$ is done by:

1. Apply the conditional probability rule.

$$P(H \mid E) = P(H \wedge E) / P(E)$$

2. Apply the marginal distribution rule to the unknown vertices $\mathbf{U}$.

$$P(H \wedge E) = \sum_{\mathbf{U}=\mathbf{u}} P(H \wedge E \wedge \mathbf{U} = \mathbf{u})$$

3. Apply joint distribution rule for Bayesian networks.

$$\mathbf{P}(X_1, \ldots, X_n) = \prod_{i=1}^{n} \mathbf{P}(X_i \mid Parents(X_i))$$

22

Enumeration Algorithm: Brute Force

Variable Elimination Algorithm

Enumeration is inefficient: repeated computation
e.g., computes $P(J = true|a)P(M = true|a)$ for each value of e

Variable elimination: carry out summations right-to-left,
storing intermediate results (factors) to avoid recomputation

$$\begin{aligned}
&\mathbf{P}(B|J = true, M = true) \\
&= \alpha \underbrace{\mathbf{P}(B)}_{B} \Sigma_e \underbrace{P(e)}_{E} \Sigma_a \underbrace{\mathbf{P}(a|B,e)}_{A} \underbrace{P(J = true|a)}_{J} \underbrace{P(M = true|a)}_{M} \\
&= \alpha \mathbf{P}(B) \Sigma_e P(e) \Sigma_a \mathbf{P}(a|B,e) P(J = true|a) f_M(a) \\
&= \alpha \mathbf{P}(B) \Sigma_e P(e) \Sigma_a \mathbf{P}(a|B,e) f_J(a) f_M(a) \\
&= \alpha \mathbf{P}(B) \Sigma_e P(e) \Sigma_a f_A(a,b,e) f_J(a) f_M(a) \\
&= \alpha \mathbf{P}(B) \Sigma_e P(e) f_{\bar{A}JM}(b,e) \text{ (sum out } A) \\
&= \alpha \mathbf{P}(B) f_{\bar{E}\bar{A}JM}(b) \text{ (sum out } E) \\
&= \alpha f_B(b) \times f_{\bar{E}\bar{A}JM}(b)
\end{aligned}$$

Pointwise Product: Combine 2 functions with overlapping variables by multiplying them

a	b	$f_1(a,b)$
T	T	.3
T	F	.7
F	T	.9
F	F	.1

b	c	$f_2(b,c)$
T	T	.2
T	F	.8
F	T	.6
F	F	.4

a	b	c	$f(a,b,c)$
T	T	T	.3 * .2
T	T	F	.3 * .8
T	F	T	.7 * .6
T	F	F	.7 * .4
F	T	T	.9 * .2
F	T	F	.9 * .8
F	F	T	.1 * .6
F	F	F	.1 * .4

Summing Out: Remove a variable from a function by adding all applicable lines and making a new table

a	b	c	$f(a,b,c)$
T	T	T	.3 * .2
T	T	F	.3 * .8
T	F	T	.7 * .6
T	F	F	.7 * .4
F	T	T	.9 * .2
F	T	F	.9 * .8
F	F	T	.1 * .6
F	F	F	.1 * .4

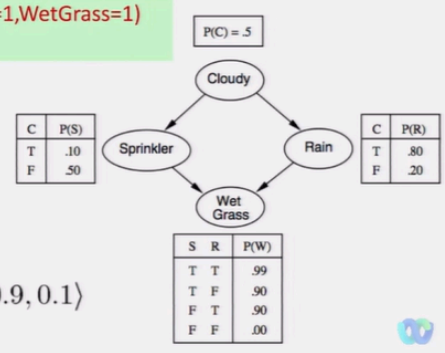
b	c	$f_{\bar{a}}(b,c)$
T	T	.3 * .2 + .9 * .2
T	F	.3 * .8 + .9 * .8
F	T	.7 * .6 + .1 * .6
F	F	.7 * .4 + .1 * .4

Inference by Stochastic Simulation (Approximate Inference)

- Sampling from empty network:

P(Cloudy=1,Sprinkler=0,Rain=1,WetGrass=1)
=0.5*0.9*0.8*0.9=0.324

$\mathbf{P}(Cloudy) = \langle 0.5, 0.5 \rangle$
sample $\rightarrow$ $true$
$\mathbf{P}(Sprinkler|Cloudy) = \langle 0.1, 0.9 \rangle$
sample $\rightarrow$ $false$
$\mathbf{P}(Rain|Cloudy) = \langle 0.8, 0.2 \rangle$
sample $\rightarrow$ $true$
$\mathbf{P}(WetGrass|\neg Sprinkler, Rain) = \langle 0.9, 0.1 \rangle$
sample $\rightarrow$ $true$



- Rejection sampling:

E.g., estimate $\mathbf{P}(Rain|Sprinkler = true)$ using 100 samples
27 samples have $Sprinkler = true$
Of these, 8 have $Rain = true$ and 19 have $Rain = false$.

$\hat{\mathbf{P}}(Rain|Sprinkler = true) = \text{NORMALIZE}(\langle 8, 19 \rangle) = \langle 0.296, 0.704 \rangle$

Similar to a basic real-world empirical estimation procedure

- Likelihood weighting:

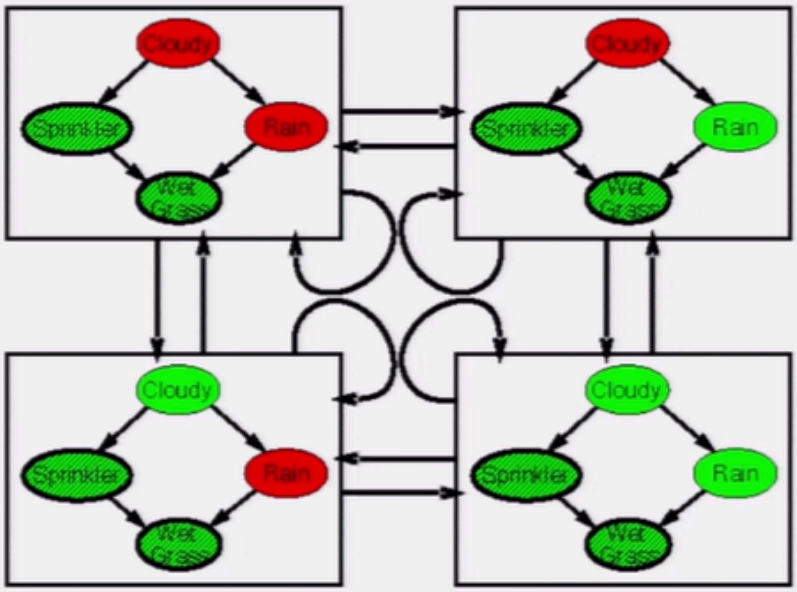
Idea: fix evidence variables, sample only nonevidence variables, and weight each sample by the likelihood it accords the evidence

- Markov Chain Monte Carlo:

"State" of network = current assignment to all variables

Generate next state by sampling one variable given Markov blanket
Sample each variable in turn, keeping evidence fixed

With $Sprinkler = true, WetGrass = true$, there are four states:



Wander about for a while, average what you see

- Bayesian Networks
  - Use Conditional Independence to efficiently represent full probability distribution
  - Can be constructed by ordering variables and check dependencies
  - Inference methods, including
    - Causal or evidential reasoning
    - Exact inferences (NP-hard)
    - Approximation Inferences

Incremental Bayesian Learning

SPAM	click for pharmacy
OK	free time today
SPAM	online pharmacy link
OK	no free time
OK	free good pharmacy
SPAM	pharmacy free link
OK	for time today
OK	time is money

Msg = "Pharmacy for pharmacy"

$$P(spam) = \frac{3}{8} \qquad P(\neg spam) = \frac{5}{8}$$

$$P("pharmacy"|spam) = 1/3$$
$$P("pharmacy"|\neg spam) = 1/15$$
$$P("for"|spam) = 1/9$$
$$P("for"|\neg spam) = 1/15$$

$$P(spam|Msg) = \frac{P(spam)P(Msg|spam)}{P(spam)P(Msg|spam) + P(\neg spam)P(Msg|\neg spam)}$$

$$P(Msg|spam) = P(w_1|spam)P(w_2|spam)P(w_3|spam)$$

Clustering:

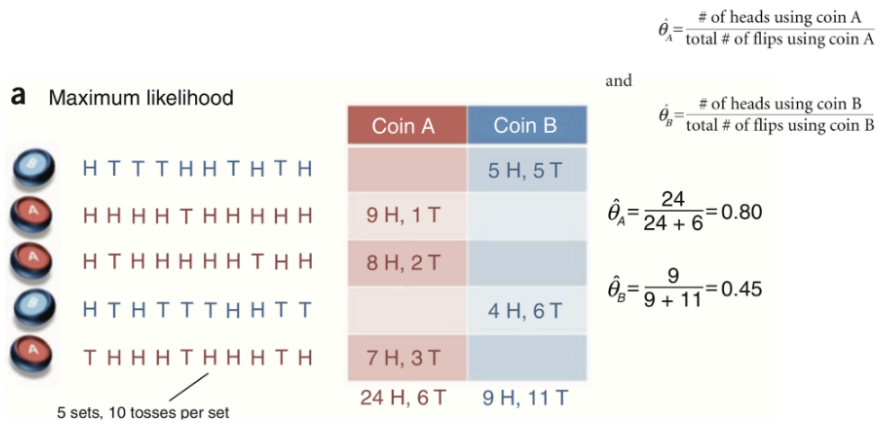
Unsupervised Learning (Clustering)

- Type I: Clustering the data
  - Automatically group the data into clusters
  - Today's lecture
- Type II: Parameter Learning
  - States are known
  - Learn transitions, sensor models, & current state
  - Today's lecture
- Type III: Structural Learning (beyond the scope of CS561)
  - States are not known (only observations and actions are known)
  - Surprise-Based Learning (see www.isi.edu/robots for references)
    - POMDP where states are NOT known and must be learned from experiences
    - States are not just "symbols" but may have internal structures

AUTOCLASS: Check 1 clusters, 2 clusters, 3 clusters, etc.

EM Algorithm:
Estimation Modification

MLE Example (2 coins)



- **E-STEP: Explaining** the Data based on the current model
  - e.g., estimate P(E|M) the likelihood of the experience E given the model M
- **M-STEP: Modifying** the Model using the explained data
  - e.g., Maximizing the parameters of the model M using the knowledge learned from the experience experience
  - E.g., Baum-Welch Learning Procedure

The General EM Algorithm

- E-Step: Estimate P(E|M) the likelihood of the experience E given the model M
  - E.g., computing α, β, γ, ξ using the experience
  - K-means: assigning data to the (closest) clusters
- M-Step: Maximize the parameters of the model M using the knowledge (e.g., explanations) learned from the experience
  - E.g., update P, θ, π using α, β, γ, ξ
  - K-means: move the clusters based on the assignments

Two types of learning in AI

Deductive: Deduce rules/facts from already known rules/facts. (We have already dealt with this)

$$(A \Rightarrow B \Rightarrow C) \Rightarrow (A \Rightarrow C)$$

Inductive: Learn new rules/facts from a data set D.

$$\mathcal{D} = \{\mathbf{x}(n), y(n)\}_{n=1\ldots N} \Rightarrow (A \Rightarrow C)$$

We will be dealing with the latter, *inductive* learning, now

Naive Bayes for Text Classification

Suppose I want to know if a news article is about sports, politics, entertainment

Classes: sports, politics, entertainment

Probability that a document *d* belongs to class *c*: $P(d|c)$

Probability of class *c* given document *d*: $P(c \mid d)$

// d_t is data
// c_j is the concept or hypothesis

Compute for every class, and choose the max to predict which c^*

$$P(c|d) = \frac{P(c)P(d|c)}{P(d)}$$

$$\begin{aligned} c^* &= \arg\max_c P(c|d) \\ &= \arg\max_c \frac{P(c)P(d|c)}{P(d)} \\ &= \arg\max_c P(c)P(d|c) \end{aligned}$$

Support Vector Machines: Finds the maximum margin separator when classifying, can handle data that is not linearly separable

Decision Tree:

- ID3 Algorithm: Constructs Decision trees greedily using max gain
  - Max-Gain: Choose the attribute that has the largest expected information gain–i.e., attribute that results in smallest expected size of subtrees rooted at its children
- Entropy of Probability Distribution of Message P

$$I(P) = -\left[p_1 * \log(p_1) + p_2 * \log(p_2) + \ldots + p_n * \log(p_n) \right]$$

probability of msg 2

information in msg 2

Examples:

- If P is (0.5, 0.5), then I(P) = .5*1 + 0.5*1 = 1
- If P is (0.67, 0.33) then I(P) = -(2/3*log(2/3) + 1/3*log(1/3)) = 0.92
- If P is (1, 0), then I(P) = 1*log(1) + 0*log(0) = 0

The more uniform the probability distribution, the greater its information: More information is conveyed by a message telling you which event actually occurred

- A chosen attribute A divides the training set E into subsets $E_1, \ldots, E_v$ according to their values for A, where A has v distinct values.

$$remainder(A) = \sum_{i=1}^{v} \frac{p_i + n_i}{p + n} I\left(\frac{p_i}{p_i + n_i}, \frac{n_i}{p_i + n_i}\right)$$

- Information Gain (IG) or *reduction in entropy* from the attribute test:

$$IG(A) = I\left(\frac{p}{p+n}, \frac{n}{p+n}\right) - remainder(A)$$

- Choose the attribute with the largest IG

K-Fold Cross Validation

Summary

- Learning needed for unknown environments, lazy designers
- Learning agent = performance element + learning element
- For supervised learning, the aim is to find a simple hypothesis approximately consistent with training examples
- Decision tree learning using information gain
- Learning performance = prediction accuracy measured on test set

Neural Network

- Mcculloh Pitts neuron
  - Inputs multiply by a weight, sum them all up, compare to number in the neuron, if greater than threshold return 1, else return 0
-
  - single layer perceptrons can only represent linear decision surfaces
  - multi-layer perceptrons can represent non-linear decision surfaces.

Older Stuff

Partially Observable Markov Decision Process (POMDP) and Markov Decision Process (MDP)

- Partially Observable: The agent doesn't see the states, only percepts
- MDP: State and Actions, Initial State and Probability Distributions, Transition Model, Reward functions
- POMDP: State and Actions, Transition Model, Reward function, Sensor Model, Belief of current state

Hidden Markov Model

- Actions, Percepts, States, Appearance (State -> Observations), Transitions, Current State

The HMM for Little Prince's Planet

(with uncertain actions and sensors)

$A \equiv$ {forward, backward, turn-around} {*f, b, t*}

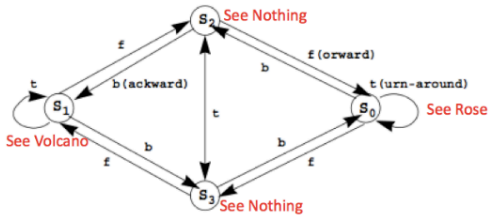
$Z \equiv$ {rose, volcano , nothing}

$S \equiv \{s_1, s_2, s_3, s_4\}$

$\phi \equiv$ {$P(s_3|s_0,f)=.51$, $P(s_2|s_1,b)=.32$, $P(s_4|s_3,t)=.89$, …}

$\theta \equiv$ {$P(rose|s_0)=.76$, $P(volcano|s_1)=.83$, $P(nothing|s_3)=.42$, …}

$\pi_1(0)=0.25, \pi_2(0)=0.25, \pi_3(0)=0.25, \pi_4(0)=0.25$



- Lookup tables for Transition Probabilities, Appearance Probabilities, and Initial State Probabilities
- Markov Assumption: History doesn't matter