Bayes net inference

Announcements

- Assignment 1
  - See Canvas

Back to AI…

- In AI we often want to predict an unknown answer given known answers to past problems
  - E.g., Given current weather observations, will it rain later?
- Whether it will rain (R) may depend on hundreds or thousands of observations, $V_1, V_2, \ldots V_{1000}$
  - Temperatures across U.S., moisture in atmosphere, etc…
- Given enough days of data, we could directly estimate a probability function $P(R \mid V_1, V_2, \ldots, V_{1000})$
  - Then problem would be solved!
  - How many days of data would you need?

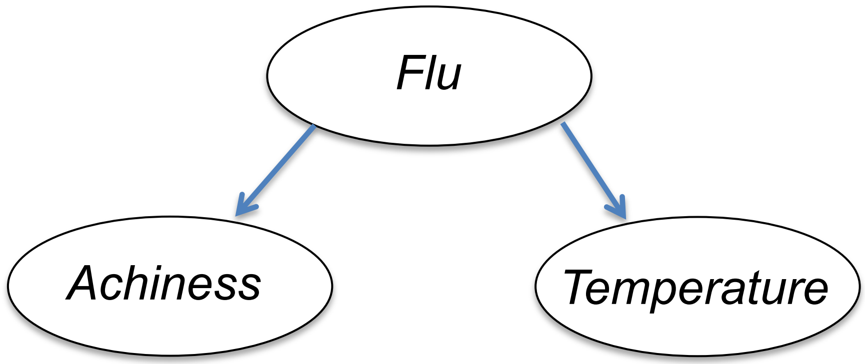
A huge problem

- Say all variables of $(R, V_1, V_2, \ldots, V_{1000})$ are binary
  - Need at least 2^{1000} days of data just to observe all possible combinations of the variables
  - Need to observe multiple days for each combination of variables to estimate conditional probability robustly
  - Simply impossible from a computational, representational, or intuitive point of view
- This seemed fatal for the first ~30 years of AI research
  - Graphical models are a framework for avoiding this problem by making assumptions about the structure of a model

Bayesian Networks

Another example

- Say we want to decide whether someone has the flu (F) based on their temperature (T) and achiness (A)
- A, T, and F are clearly **not** independent
- But a weaker assumption of conditional independence may be appropriate, $A \perp T | F$
  - Says that A and T are independent *for a given value of F*
  - We can represent this assumption with a *Bayesian network*:

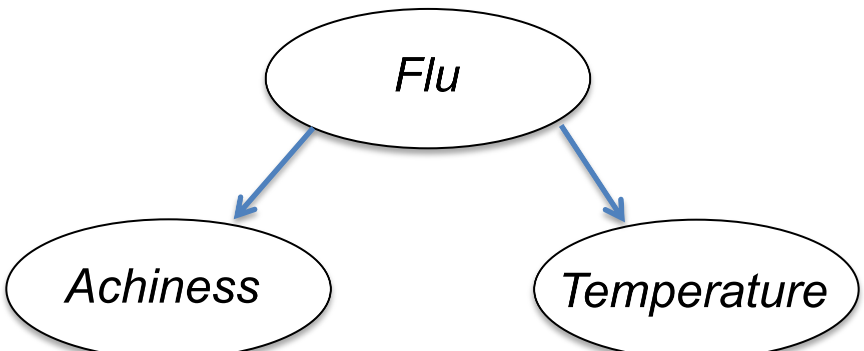


Another example

- Now we can factor P(A,T,F) as:

$$P(A,T,F) = P(A|F)P(T|F)P(F)$$

- To decide whether someone has the flu given observed symptoms, we'll want to compute P(F | A, T)
  - How to compute this?



Back to the weather…

- We want to compute probability of rain (R) given observed variables V_1, V_2, … V_{1000}. Using Bayes' law,

$$P(R|V_1, V_2, ..., V_{1000}) = \frac{P(V_1, V_2, ..., V_{1000}|R)P(R)}{P(V_1, V_2, ..., V_{1000})}$$

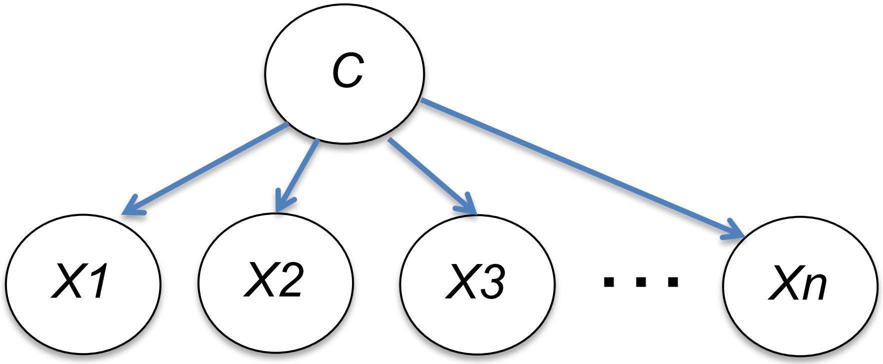
  - Now, assuming that V_1 … V_{1000} are conditionally independent given R:

$$P(V_1, V_2, ..., V_{1000}|R) = \prod_{1}^{1000} P(V_i|R)$$

  - How many parameters do we need to estimate in this factored model?

Naïve Bayes model

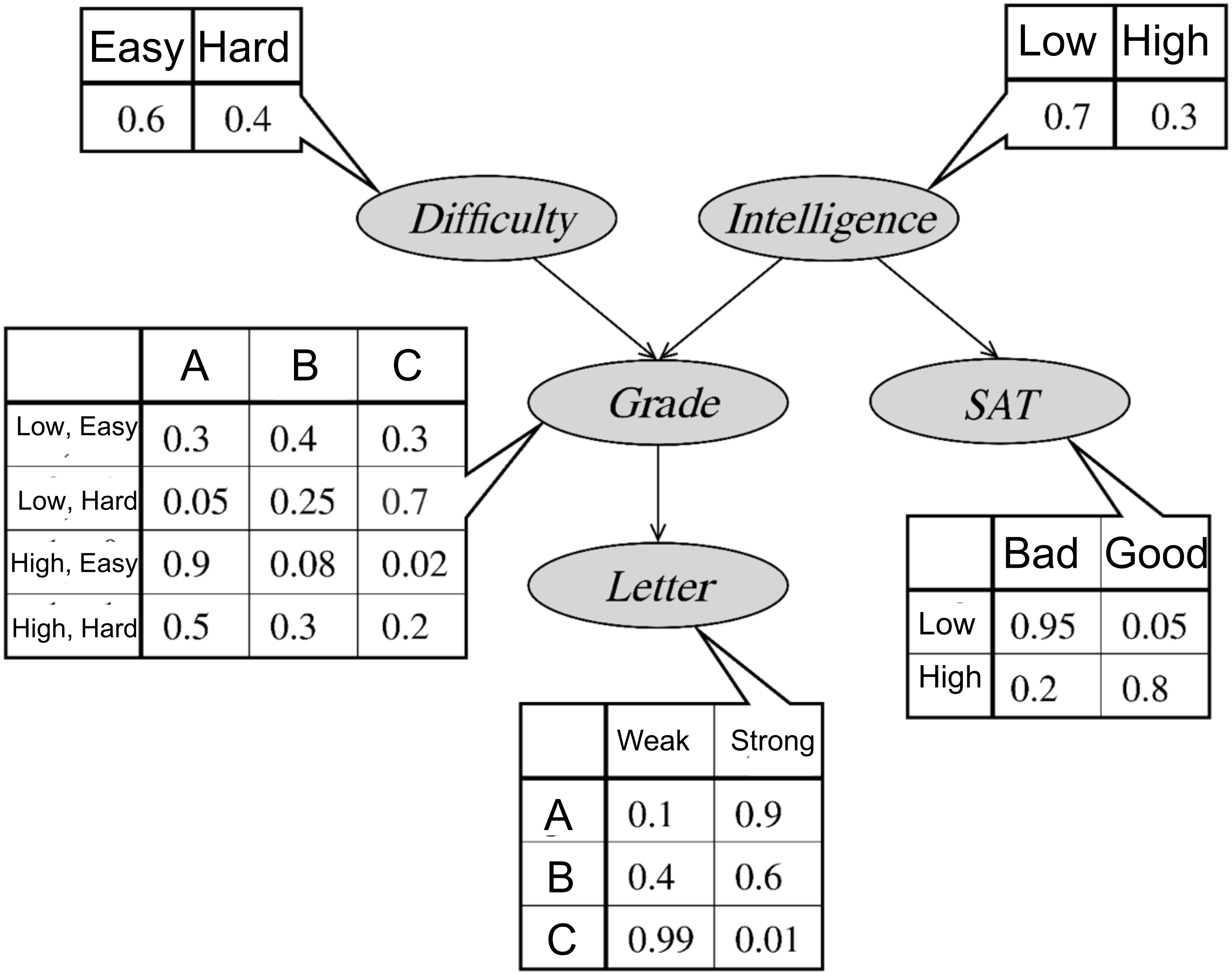
- Assuming conditional independence among observed variables is called *naïve Bayes*
  - Class label *C* we want to infer
  - Set of observable variables X1, X2, … Xn
  - Assume that observable variables are independent conditioned on the class label *C*
  - Estimate prior distribution P(C) and conditional distributions P(X1|C), …, P(Xn | C) from training data
  - Use Bayes' Law to calculate P(C | X1 … Xn)



Another example

- Suppose we want to model students in B551, using several random variables:
  - Intelligence (I)
  - GPA (G)
  - SAT score (S)
  - Difficulty of courses taken (D)
  - Strength of letter of recommendation (L)
- Intuitively, arrows in the BN represent direct dependencies between variables
  - Assuming these dependences, how does the joint distribution P(I,G,S,D,L) factor?

Conditional probability distributions



Easy	Hard
0.6	0.4

Low	High
0.7	0.3

	A	B	C
Low, Easy	0.3	0.4	0.3
Low, Hard	0.05	0.25	0.7
High, Easy	0.9	0.08	0.02
High, Hard	0.5	0.3	0.2

	Bad	Good
Low	0.95	0.05
High	0.2	0.8

	Weak	Strong
A	0.1	0.9
B	0.4	0.6
C	0.99	0.01

Bayesian networks

- A Bayesian network is defined by a pair (G,P), where:
  - G is a dag (directed acyclic graph), with nodes corresponding to variables {X1, X2, … Xn} and edges to direct dependencies
  - P is a probability distribution that satisfies independence assumptions induced by G
- The dag G encodes the conditional independence assumptions:

$$X_i \perp \mathrm{Nd}(X_i) \mid \mathrm{Pa}(X_i)$$

  - where Nd(Xi) is the set of non-descendants of Xi, and Pa(Xi) is the set of parents of Xi

Factorization of Bayes nets

- Given a Bayes net (G,P) over variables {X1, X2 … Xn}, the joint probability distribution factors as,

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

Independencies in Bayes nets

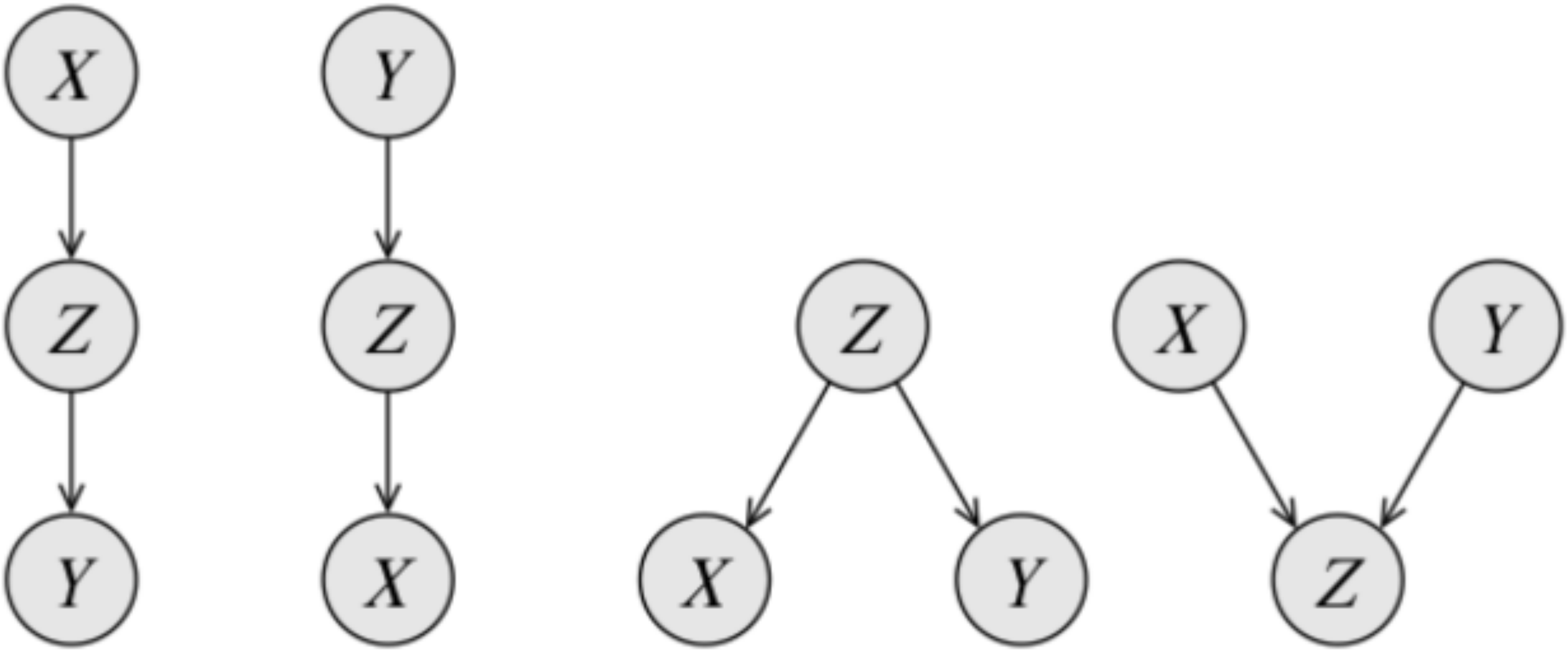
- We already have a set of some conditional independence relationships, given by:

$$X_i \perp \mathrm{Nd}(X_i) \mid \mathrm{Pa}(X_i)$$

  - These are just the relationships directly defined by G; there are often others

For three nodes, Four cases

- Is $X \perp Y \mid Z$ in each case? **Is $X \perp Y$ in each case?**



(a) (b) (c) (d)

Solving problems with Bayes Nets

Solving problems with Bayes nets

- We'd like to use Bayes nets to estimate (distributions of) variables, given observed values for some variables
  - aka *Conditional Probability Queries:* Given a set of variables **E** and corresponding values **e**, estimate distributions over unobservable values **Y**, i.e. P(**Y** | **E**=**e**)

Marginal inference example

- Alice flips a fair coin, and tells the result to Bob.
- Bob tells Charlie the true result 80% of the time, but lies 20% of the time.
- If Charlie hears heads, he tells Donna heads with probability 90% and tails with probability 10%. If he hears tails, he tells Donna heads 40% of the time and tails 60% of the time.
- What is the probability Donna hears heads?

Marginal inference example

- We want to compute P(D)

$$P(D) = \sum_C \sum_B \sum_A P(A, B, C, D)$$

$$P(D) = \sum_C \sum_B \sum_A P(A)P(B|A)P(C|B)P(D|C)$$

$$P(D) = \sum_C \sum_B P(C|B)P(D|C) \left(\sum_A P(A)P(B|A) \right)$$

$$P(D) = \sum_C \sum_B P(C|B)P(D|C)\tau_1(B)$$

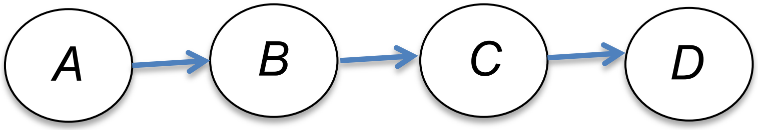
$$P(D) = \sum_C P(D|C) \left(\sum_B P(C|B)\tau_1(B) \right)$$

$$P(D) = \sum_C P(D|C)\tau_2(C)$$

Dynamic programming

- This idea of caching intermediate results (in the form of tables τ_1 and τ_2) is called *dynamic programming*
  - General algorithmic concept
  - Other examples: Dijkstra's algorithm, string algorithms (e.g. for bioinformatics), Tower of Hanoi puzzle, …

How did we avoid exponential time?



- Two important ingredients:
  - 1. The independence assumptions of the Bayes net allowed us to factor the joint distribution into simpler terms, each of which involved only a few variables.
  - 2. Dynamic programming let us “cache” intermediate results, avoiding re-computing them repeatedly.

More generally…

- More generally, notice that for any **sets** of random variables **U**, **V**, **W**, and **X**, and random variable $Z \notin \mathbf{U} \cup \mathbf{V}$,

$$\sum_Z P(\mathbf{U}|\mathbf{V}) P(\mathbf{W}|\mathbf{X}) = P(\mathbf{U}|\mathbf{V}) \sum_Z P(\mathbf{W}|\mathbf{X})$$

  - So, in the chain example above, this lets us do:

$$\begin{aligned} P(D) &= \sum_C \sum_B \sum_A P(A)P(B|A)P(C|B)P(D|C) \\ &= \sum_C \sum_B P(C|B)P(D|C) \left(\sum_A P(A)P(B|A) \right) \\ &= \sum_C P(D|C) \left(\sum_B P(C|B) \left(\sum_A P(A)P(B|A) \right) \right) \\ &= \sum_C P(D|C) \sum_B P(C|B) \sum_A P(A)P(B|A) \end{aligned}$$

Next class

- Hidden Markov Models