Learning Under Uncertainty

- We want to learn models from data.

$$P(model|data) = \frac{P(data|model) \times P(model)}{P(data)}.$$

- The likelihood, $P(data|model)$, is the probability that this model would have produced this data.

- The prior, $P(model)$, encodes the learning bias

Bayesian Leaning of Probabilities

- Suppose there are two outcomes A and $\neg A$. We would like to learn the probability of A given some data.

- We can treat the probability of A as a real-valued random variable on the interval [0, 1], called *probA*.

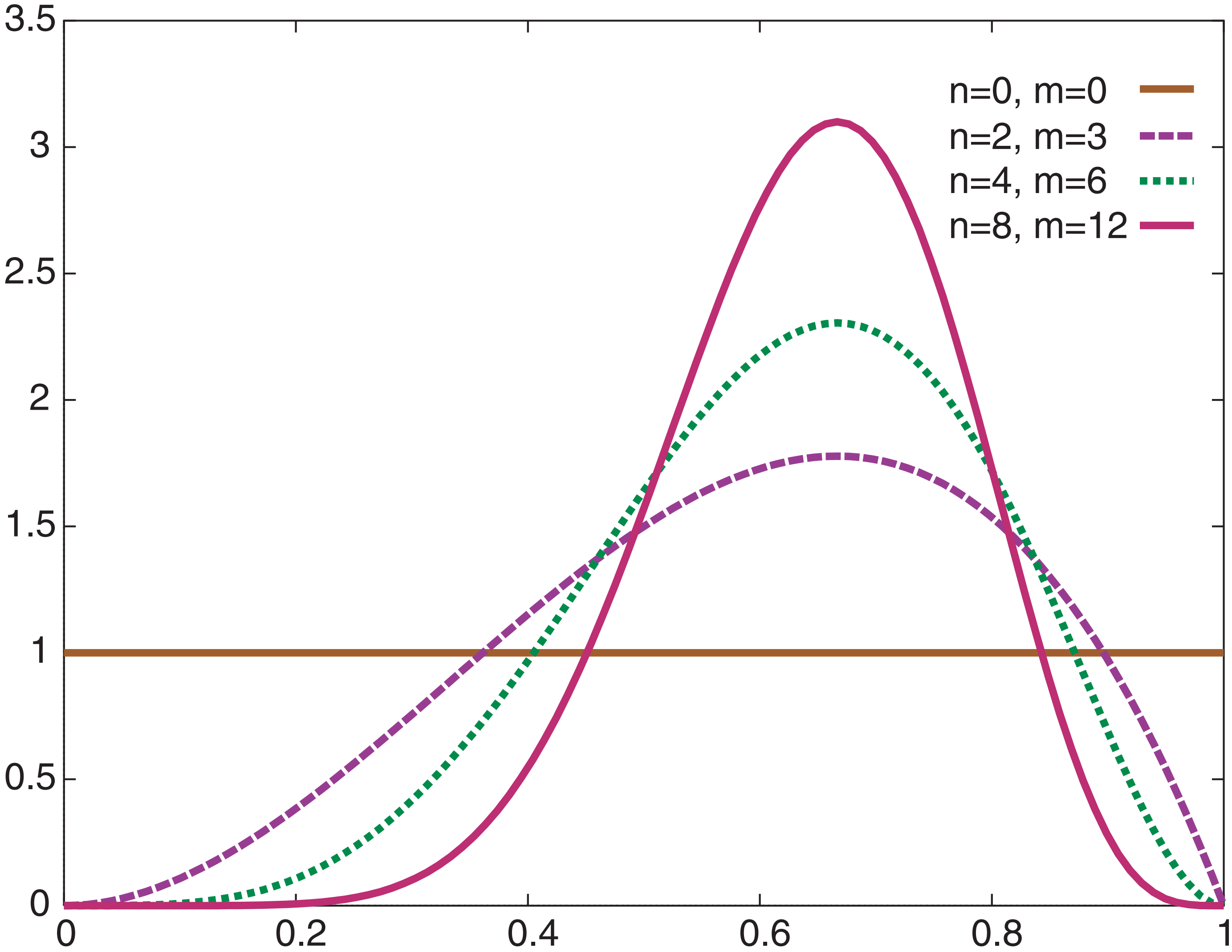
$$P(probA{=}p|data) = \frac{P(data|probA{=}p) \times P(probA{=}p)}{P(data)}$$

- Suppose the data is a sequence of n A's out of independent m trials,

$$P(data|probA{=}p) = p^n \times (1-p)^{m-n}$$

- Uniform prior: $P(probA{=}p) = 1$ for all $p \in [0, 1]$.

Posterior Probabilities for Different Data



MAP model

- The maximum a posteriori probability (MAP) model is the model that maximizes $P(model|data)$. That is, it maximizes:

$$P(data|model) \times P(model)$$

- Thus it minimizes:

$$(-\log P(data|model)) + (-\log P(model))$$

which is the number of bits to send the data given the model plus the number of bits to send the model.

Information theory overview

- A bit is a binary digit.
- 1 bit can distinguish 2 items
- k bits can distinguish 2^k items
- n items can be distinguished using $\log_2 n$ bits
- Can you do better?

Information and Probability

Let's design a code to distinguish elements of $\{a, b, c, d\}$ with

$$P(a) = \frac{1}{2}, P(b) = \frac{1}{4}, P(c) = \frac{1}{8}, P(d) = \frac{1}{8}$$

Consider the code:

$$a \quad 0 \qquad b \quad 10 \qquad c \quad 110 \qquad d \quad 111$$

This code sometimes uses 1 bit and sometimes uses 3 bits.

On average, it uses

$$\begin{aligned} & P(a) \times 1 + P(b) \times 2 + P(c) \times 3 + P(d) \times 3 \\ = \quad & \frac{1}{2} + \frac{2}{4} + \frac{3}{8} + \frac{3}{8} = 1\frac{3}{4} \text{ bits.} \end{aligned}$$

The string *aacabbda* has code 00110010101110.

Information Content

- To identify x, you need $-\log_2 P(x)$ bits.
- If you have a distribution over a set and want to a identify a member, you need the expected number of bits:

$$\sum_x -P(x) \times \log_2 P(x).$$

This is the information content or entropy of the distribution.

- The expected number of bits it takes to describe a distribution given evidence e:

$$I(e) = \sum_x -P(x|e) \times \log_2 P(x|e).$$

Information Gain

If you have a test that can distinguish the cases where α is true from the cases where α is false, the information gain from this test is:

$$I(true) - (P(\alpha) \times I(\alpha) + P(\neg\alpha) \times I(\neg\alpha)).$$

- $I(true)$ is the expected number of bits needed before the test
- $P(\alpha) \times I(\alpha) + P(\neg\alpha) \times I(\neg\alpha)$ is the expected number of bits after the test.

Averaging Over Models

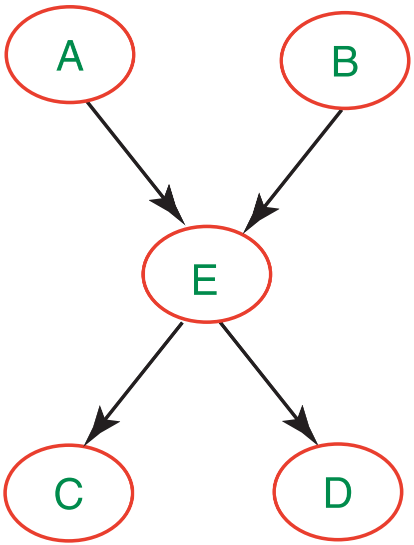
- **Idea:** Rather than choosing the most likely model, average over all models, weighted by their posterior probabilities given the data.
- If you have observed n A's out of m trials
  - the most likely value (MAP) is $\frac{n}{m}$
  - the expected value is $\frac{n+1}{m+2}$

Learning a Belief Network

- If you
  - know the structure
  - have observed all of the variables
  - have no missing data
- you can learn each conditional probability separately.

Learning belief network example

Model



Data

A	B	C	D	E
t	f	t	t	f
f	t	t	t	t
t	t	f	t	f
		$\dots$		

$\rightarrow$ Probabilities

$P(A)$

$P(B)$

$P(E|A,B)$

$P(C|E)$

$P(D|E)$

Learning conditional probabilities

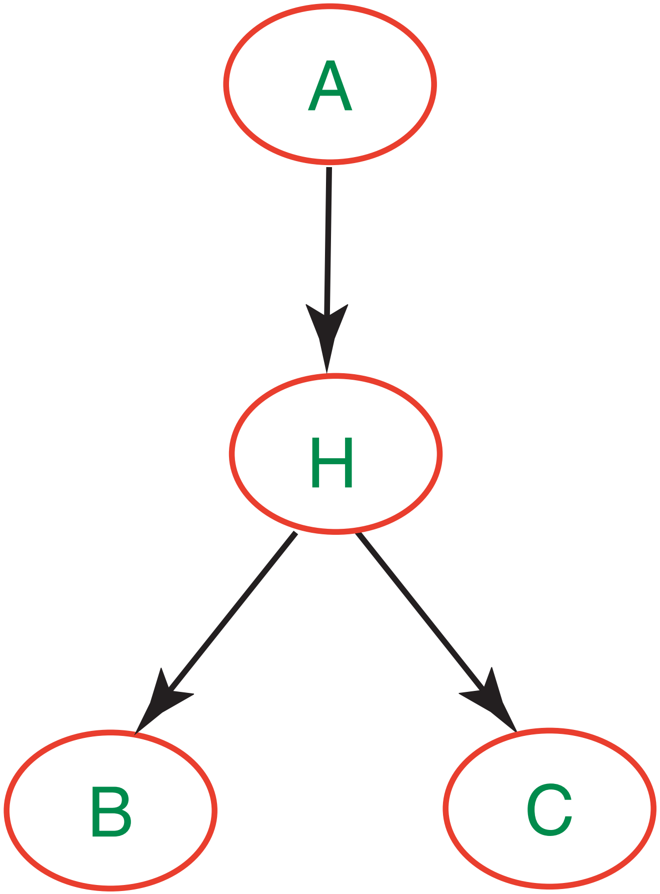
- Each conditional probability distribution can be learned separately:
- For example:

$$P(E = t | A = t \wedge B = f) = \frac{(\#\text{examples: } E = t \wedge A = t \wedge B = f) + m_1}{(\#\text{examples: } A = t \wedge B = f) + m}$$

where m_1 and m reflect our prior knowledge.

- There is a problem when there are many parents to a node as then there is little data for each probability estimate.

Unobserved Variables



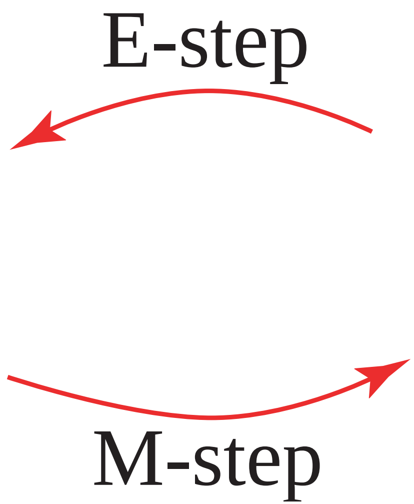
- What if we had only observed values for *A*, *B*, *C*?

A	*B*	*C*
t	*f*	*t*
f	*t*	*t*
t	*t*	*f*
	...	

EM Algorithm

Augmented Data

A	B	C	H
t	f	t	t
f	t	t	f
t	t	f	t
	$\cdots$		



Probabilities

$P(A)$

$P(H|A)$

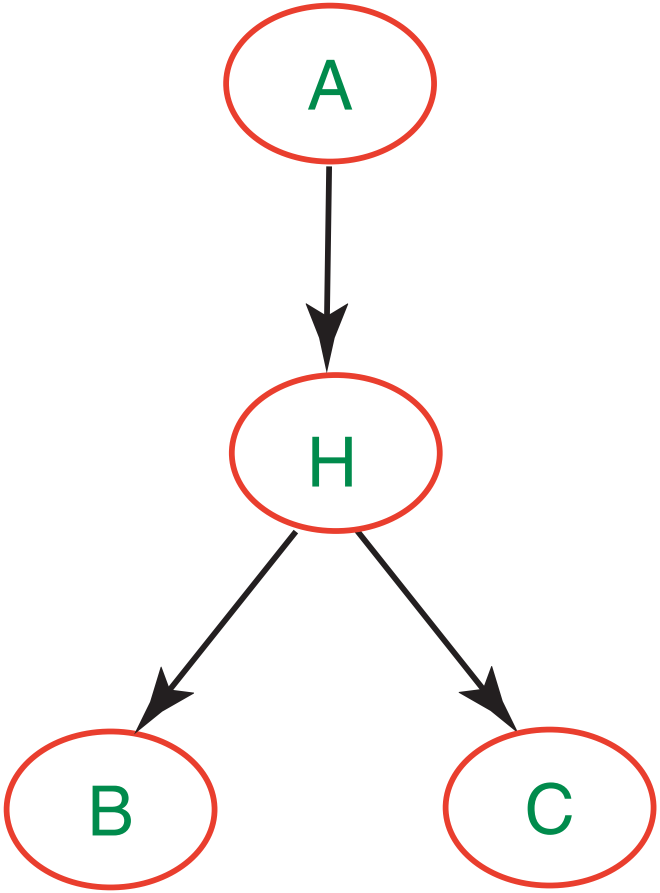
$P(B|H)$

$P(C|H)$

EM Algorithm

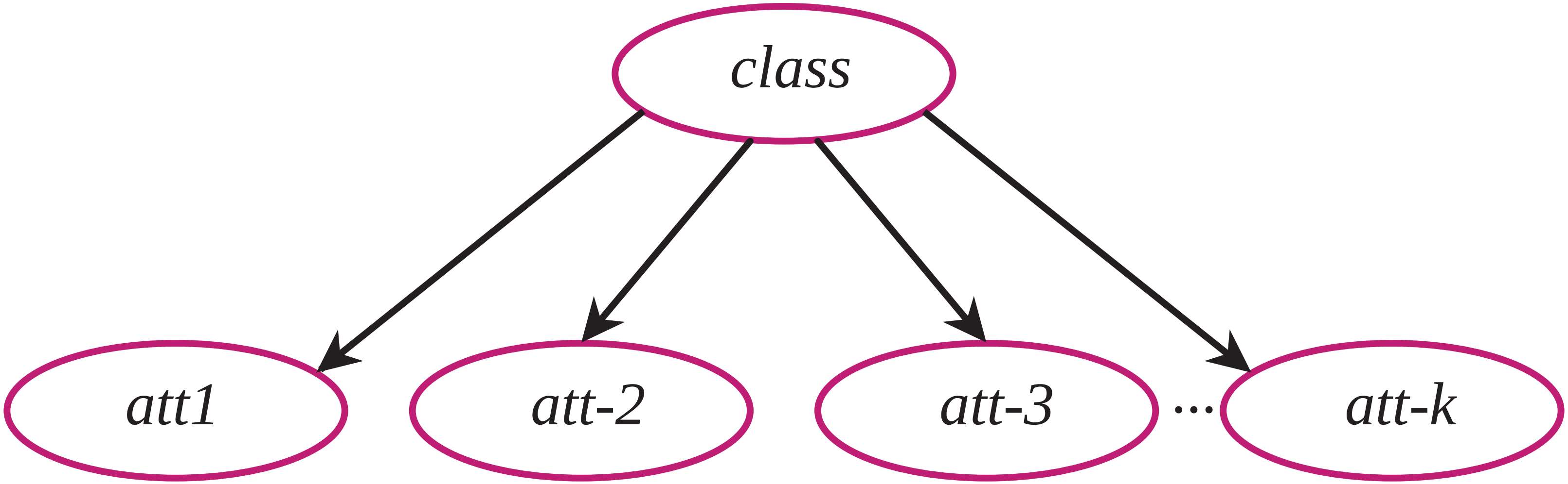
- Repeat the following two steps:
  - **E-step** give the expected number of data points for the unobserved variables based on the given probabilty distribution.
  - **M-step** infer the (maximun likelihood) probabilities from the data. This is the same as the full observable case.
- Start either with made-up data or made-up probabilities.
- EM will converge to a local maxima.

Example Data



A	B	C	Count
t	t	t	143
t	t	f	329
t	f	t	57
t	f	f	271
f	t	t	87
f	t	f	66
f	f	t	23
f	f	f	24

Naive Bayesian Classifier



Unsupervised Learning

- Given a collection of data, find natural classifications.
- This can be seen as the naive Bayesian classifier with the classification unobserved.
- EM can be used to learn classification.

Bayesian learning of decision trees

$$P(model|data) = \frac{P(data|model) \times P(model)}{P(data)}.$$

- A model here is a decision tree
- We allow for decision trees with probabilities at the leaves
- A bigger decision tree can always fit the data better
- $P(model)$ lets us encode a preference for smaller decision trees.

Data for decision tree learning

att_1	att_2	*class*	count
t	t	c1	10
t	t	c2	3
t	f	c1	5
t	f	c2	12
f	t	c1	7
f	t	c2	14
f	f	c1	8
f	f	c2	1

Probabilities From Experts

- Bayes rule lets us combine expert knowledge with data

$$P(model|data) = \frac{P(data|model) \times P(model)}{P(data)}.$$

- The experts prior knowledge of the model (i.e., $P(model)$) can be expressed as a pair $\langle n, m \rangle$ that can be interpreted as though they had observed n A's out of m trials.

- This estimate can be combined with data.

- Estimates from multiple experts can be combined together.