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.

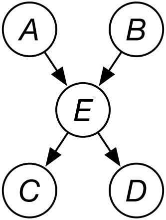
Learning a Belief Network

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

$\longrightarrow$ supervised learning

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

$P(A)$
$P(B)$
$P(E \mid A, B)$
$P(C \mid E)$
$P(D \mid E)$

Learning conditional probabilities

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

$$P(E = t \mid A = t \wedge B = f) = \frac{(\#\text{examples: } E = t \wedge A = t \wedge B = f) + c_1}{(\#\text{examples: } A = t \wedge B = f) + c}$$

  where c_1 and c reflect prior (expert) knowledge ($c_1 \leq c$).
- When there are many parents to a node, there can little or no data for each conditional probability:

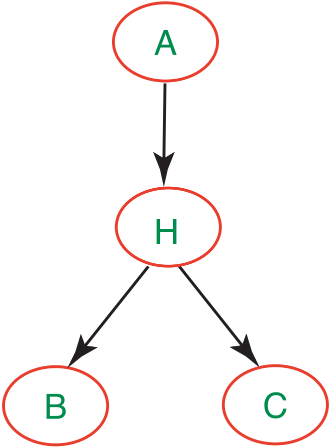
Learning conditional probabilities

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

$$P(E = t \mid A = t \wedge B = f)$$
$$= \frac{(\#\text{examples: } E = t \wedge A = t \wedge B = f) + c_1}{(\#\text{examples: } A = t \wedge B = f) + c}$$

  where c_1 and c reflect prior (expert) knowledge ($c_1 \leq c$).
- When there are many parents to a node, there can little or no data for each conditional probability: use supervised learning to learn a decision tree, linear classifier, a neural network or other representation of the conditional probability.

Unobserved Variables

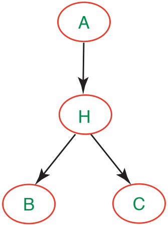


- What if you had only observed values for A, B, C?

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

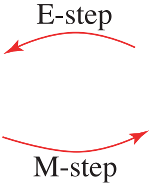
EM Algorithm

Model



Augmented Data

A	B	C	H	$Count$
t	f	t	t	0.7
t	f	t	f	0.3
f	t	t	f	0.9
f	t	t	t	0.1
	$\dots$			$\dots$



Probabilities

$P(A)$
$P(H \mid A)$
$P(B \mid H)$
$P(C \mid H)$

EM Algorithm

- Repeat the following two steps:
  - E-step give the expected number of data points for the unobserved variables based on the given probability distribution. Requires probabilistic inference.

EM Algorithm

- Repeat the following two steps:
  - E-step give the expected number of data points for the unobserved variables based on the given probability distribution. Requires probabilistic inference.
  - M-step infer the (maximum likelihood) probabilities from the data. This is the same as the fully-observable case.

EM Algorithm

- Repeat the following two steps:
  - E-step give the expected number of data points for the unobserved variables based on the given probability distribution. Requires probabilistic inference.
  - M-step infer the (maximum likelihood) probabilities from the data. This is the same as the fully-observable case.
- Start either with made-up data or made-up probabilities.

EM Algorithm

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

Belief network structure learning (I)

Given examples $\mathbf{e}$, and model m:

$$P(m \mid \mathbf{e}) = \frac{P(\mathbf{e} \mid m) * P(m)}{P(\mathbf{e})}.$$

- A model here is a belief network.

Belief network structure learning (I)

Given examples $\mathbf{e}$, and model m:

$$P(m \mid \mathbf{e}) = \frac{P(\mathbf{e} \mid m) * P(m)}{P(\mathbf{e})}.$$

- A model here is a belief network.
- A bigger network can always fit the data better.

Belief network structure learning (I)

Given examples **e**, and model m:

$$P(m \mid \mathbf{e}) = \frac{P(\mathbf{e} \mid m) * P(m)}{P(\mathbf{e})}.$$

- A model here is a belief network.
- A bigger network can always fit the data better.
- $P(m)$ lets us encode a preference for simpler models (e.g, smaller networks)

Belief network structure learning (I)

Given examples $\mathbf{e}$, and model m:

$$P(m \mid \mathbf{e}) = \frac{P(\mathbf{e} \mid m) * P(m)}{P(\mathbf{e})}.$$

- A model here is a belief network.
- A bigger network can always fit the data better.
- $P(m)$ lets us encode a preference for simpler models (e.g, smaller networks)

$\longrightarrow$ search over network structure looking for the most likely model.

Belief network structure learning (I)

Given examples $\mathbf{e}$, and model m:

$$P(m \mid \mathbf{e}) = \frac{P(\mathbf{e} \mid m) * P(m)}{P(\mathbf{e})}.$$

- A model here is a belief network.
- A bigger network can always fit the data better.
- $P(m)$ lets us encode a preference for simpler models (e.g, smaller networks)

$\longrightarrow$ search over network structure looking for the most likely model.

- Taking logarithms (and negating):

$$\arg\max_m P(m \mid \mathbf{e}) = \arg\min_m(-\log P(\mathbf{e} \mid m) - \log P(m))$$

Description Length

Bayes Rule:

$$P(h|d) \propto P(d|h)P(h)$$

$$\begin{aligned}\arg\max_h P(h|d) &= \arg\max_h P(d|h)P(h) \\ &= \arg\max_h (\log P(d|h) + \log P(h))\end{aligned}$$

Description Length

Bayes Rule:

$$P(h|d) \propto P(d|h)P(h)$$

$$\begin{aligned}\arg\max_h P(h|d) &= \arg\max_h P(d|h)P(h) \\ &= \arg\max_h (\log P(d|h) + \log P(h))\end{aligned}$$

- $\log P(d|h)$ measures fit to data
- $\log P(h)$ measures model complexity

Information theory overview

- A bit is a binary digit.
- 1 bit can distinguish

Information theory overview

- A bit is a binary digit.
- 1 bit can distinguish 2 items
- k bits can distinguish

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

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

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

Consider 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	0	b	10	c	110	d	111

This code uses bits.

Information and Probability

Consider 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	0	b	10	c	110	d	111

This code uses 1 to 3 bits. On average, it uses

Consider 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	0	b	10	c	110	d	111

This code uses 1 to 3 bits. On average, it uses

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

The string *aacabbda* has code

Information and Probability

Consider 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	0	b	10	c	110	d	111

This code uses 1 to 3 bits. On average, it uses

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

The string *aacabbda* has code 00110010101110.
The code 0111110010100 represents string

Information and Probability

Consider 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	0	b	10	c	110	d	111

This code uses 1 to 3 bits. On average, it uses

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

The string *aacabbda* has code 00110010101110.
The code 0111110010100 represents string *adcabba*

Information Content

- To identify x, you need $-\log_2 P(x)$ bits.

Information Content

- To identify x, you need $-\log_2 P(x)$ bits.
- Given a distribution over a set of values, to identify a member, the expected number of bits is

Information Content

- To identify x, you need $-\log_2 P(x)$ bits.
- Given a distribution over a set of values, to identify a member, the expected number of bits is

$$\sum_x -P(x) * \log_2 P(x)$$

is the information content or entropy of the distribution.

Information Content

- To identify x, you need $-\log_2 P(x)$ bits.
- Given a distribution over a set of values, to identify a member, the expected number of bits is

$$\sum_x -P(x) * \log_2 P(x)$$

is the information content or entropy of the distribution.

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

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

Information Gain

Given 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:

Information Gain

Given 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:

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

Information Gain

Given 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:

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

- $I(true)$ is the expected number of bits needed before the test

Information Gain

Given 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:

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

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

Comparing Distributions

- Suppose a code is designed to be optimal for probability distribution Q, so that x is described using $-log_2 Q(x)$ bits.
- Suppose P is another probability distribution. The expected number of bits to describe P using the code for Q is

$$\sum_x -P(x) * \log_2 Q(x)$$

Comparing Distributions

- Suppose a code is designed to be optimal for probability distribution Q, so that x is described using $-log_2 Q(x)$ bits.
- Suppose P is another probability distribution. The expected number of bits to describe P using the code for Q is

$$\sum_x -P(x) * \log_2 Q(x)$$

- The difference between this and the entropy of P — describing P using its optimal code – is the Kullback–Leibler (KL) divergence (also called relative entropy):

$$D_{KL}(P \,||\, Q) = (\sum_x -P(x) * \log Q(x)) - \sum_x -P(x) * \log P(x)$$
$$= \sum_x -P(x) * \log(Q(x)/P(x))$$

Comparing Distributions

- Suppose a code is designed to be optimal for probability distribution Q, so that x is described using $-log_2 Q(x)$ bits.
- Suppose P is another probability distribution. The expected number of bits to describe P using the code for Q is

$$\sum_x -P(x) * \log_2 Q(x)$$

- The difference between this and the entropy of P — describing P using its optimal code – is the Kullback–Leibler (KL) divergence (also called relative entropy):

$$D_{KL}(P \,||\, Q) = (\sum_x -P(x) * \log Q(x)) - \sum_x -P(x) * \log P(x)$$
$$= \sum_x -P(x) * \log(Q(x)/P(x))$$

- When is this large?

Comparing Distributions

- Suppose a code is designed to be optimal for probability distribution Q, so that x is described using $-log_2 Q(x)$ bits.
- Suppose P is another probability distribution. The expected number of bits to describe P using the code for Q is

$$\sum_x -P(x) * \log_2 Q(x)$$

- The difference between this and the entropy of P — describing P using its optimal code – is the Kullback–Leibler (KL) divergence (also called relative entropy):

$$D_{KL}(P \,||\, Q) = (\sum_x -P(x) * \log Q(x)) - \sum_x -P(x) * \log P(x)$$
$$= \sum_x -P(x) * \log(Q(x)/P(x))$$

- When is this large? When $P(x) \gg Q(x)$ for some x.

A belief network structure learning algorithm

- Search over total orderings of variables.

A belief network structure learning algorithm

- Search over total orderings of variables.
- For each total ordering $X_1, \ldots, X_n$ use supervised learning to learn $P(X_i \mid X_1 \ldots X_{i-1})$.

A belief network structure learning algorithm

- Search over total orderings of variables.
- For each total ordering $X_1, \ldots, X_n$ use supervised learning to learn $P(X_i \mid X_1 \ldots X_{i-1})$.
- Return the network model found with minimum:
  $-\log P(\mathbf{e} \mid m) - \log P(m)$

A belief network structure learning algorithm

- Search over total orderings of variables.
- For each total ordering $X_1, \ldots, X_n$ use supervised learning to learn $P(X_i \mid X_1 \ldots X_{i-1})$.
- Return the network model found with minimum:
  $-\log P(\mathbf{e} \mid m) - \log P(m)$
  - $P(\mathbf{e} \mid m)$ can be obtained by inference.

A belief network structure learning algorithm

- Search over total orderings of variables.
- For each total ordering $X_1, \ldots, X_n$ use supervised learning to learn $P(X_i \mid X_1 \ldots X_{i-1})$.
- Return the network model found with minimum:
  $-\log P(\mathbf{e} \mid m) - \log P(m)$
  - $P(\mathbf{e} \mid m)$ can be obtained by inference.
  - How to determine $-\log P(m)$?

Bayesian Information Criterion (BIC) Score

$$P(m \mid \mathbf{e}) = \frac{P(\mathbf{e} \mid m) * P(m)}{P(\mathbf{e})}$$

$$-\log P(m \mid \mathbf{e}) \propto -\log P(\mathbf{e} \mid m) - \log P(m)$$

Bayesian Information Criterion (BIC) Score

$$P(m \mid \mathbf{e}) = \frac{P(\mathbf{e} \mid m) * P(m)}{P(\mathbf{e})}$$

$$-\log P(m \mid \mathbf{e}) \propto -\log P(\mathbf{e} \mid m) - \log P(m)$$

- $-\log P(\mathbf{e} \mid m)$ is the negative log likelihood of model m:

Bayesian Information Criterion (BIC) Score

$$P(m \mid \mathbf{e}) = \frac{P(\mathbf{e} \mid m) * P(m)}{P(\mathbf{e})}$$

$$-\log P(m \mid \mathbf{e}) \propto -\log P(\mathbf{e} \mid m) - \log P(m)$$

- $-\log P(\mathbf{e} \mid m)$ is the negative log likelihood of model m: number of bits to describe the data in terms of the model.

Bayesian Information Criterion (BIC) Score

$$P(m \mid \mathbf{e}) = \frac{P(\mathbf{e} \mid m) * P(m)}{P(\mathbf{e})}$$

$$-\log P(m \mid \mathbf{e}) \propto -\log P(\mathbf{e} \mid m) - \log P(m)$$

- $-\log P(\mathbf{e} \mid m)$ is the negative log likelihood of model m: number of bits to describe the data in terms of the model.
- $|\mathbf{e}|$ is the number of examples. Each proposition can be true for between 0 and $|\mathbf{e}|$ examples, so there are   different probabilities to distinguish.

Bayesian Information Criterion (BIC) Score

$$P(m \mid \mathbf{e}) = \frac{P(\mathbf{e} \mid m) * P(m)}{P(\mathbf{e})}$$

$$-\log P(m \mid \mathbf{e}) \propto -\log P(\mathbf{e} \mid m) - \log P(m)$$

- $-\log P(\mathbf{e} \mid m)$ is the negative log likelihood of model m: number of bits to describe the data in terms of the model.
- $|\mathbf{e}|$ is the number of examples. Each proposition can be true for between 0 and $|\mathbf{e}|$ examples, so there are $|\mathbf{e}| + 1$ different probabilities to distinguish. Each one can be described in bits.

Bayesian Information Criterion (BIC) Score

$$P(m \mid \mathbf{e}) = \frac{P(\mathbf{e} \mid m) * P(m)}{P(\mathbf{e})}$$

$$-\log P(m \mid \mathbf{e}) \propto -\log P(\mathbf{e} \mid m) - \log P(m)$$

- $-\log P(\mathbf{e} \mid m)$ is the negative log likelihood of model m: number of bits to describe the data in terms of the model.
- $|\mathbf{e}|$ is the number of examples. Each proposition can be true for between 0 and $|\mathbf{e}|$ examples, so
  there are $|\mathbf{e}| + 1$ different probabilities to distinguish.
  Each one can be described in $\log(|\mathbf{e}| + 1) \approx \log(|\mathbf{e}|)$ bits.

Bayesian Information Criterion (BIC) Score

$$P(m \mid \mathbf{e}) = \frac{P(\mathbf{e} \mid m) * P(m)}{P(\mathbf{e})}$$

$$-\log P(m \mid \mathbf{e}) \propto -\log P(\mathbf{e} \mid m) - \log P(m)$$

- $-\log P(\mathbf{e} \mid m)$ is the negative log likelihood of model m: number of bits to describe the data in terms of the model.
- $|\mathbf{e}|$ is the number of examples. Each proposition can be true for between 0 and $|\mathbf{e}|$ examples, so there are $|\mathbf{e}| + 1$ different probabilities to distinguish. Each one can be described in $\log(|\mathbf{e}| + 1) \approx \log(|\mathbf{e}|)$ bits.
- If there are $||m||$ independent parameters ($||m||$ is the dimensionality of the model):

$$-\log P(m \mid \mathbf{e}) \propto$$

Bayesian Information Criterion (BIC) Score

$$P(m \mid \mathbf{e}) = \frac{P(\mathbf{e} \mid m) * P(m)}{P(\mathbf{e})}$$

$$-\log P(m \mid \mathbf{e}) \propto -\log P(\mathbf{e} \mid m) - \log P(m)$$

- $-\log P(\mathbf{e} \mid m)$ is the negative log likelihood of model m: number of bits to describe the data in terms of the model.
- $|\mathbf{e}|$ is the number of examples. Each proposition can be true for between 0 and $|\mathbf{e}|$ examples, so there are $|\mathbf{e}| + 1$ different probabilities to distinguish. Each one can be described in $\log(|\mathbf{e}| + 1) \approx \log(|\mathbf{e}|)$ bits.
- If there are $||m||$ independent parameters ($||m||$ is the dimensionality of the model):

$$-\log P(m \mid \mathbf{e}) \propto -\log P(\mathbf{e} \mid m) + ||m|| \log(|\mathbf{e}|)$$

This is the Bayesian Information Criterion (BIC) score.

Belief network structure learning (II)

- Given a total ordering, to determine $parents(X_i)$ do independence tests to determine which features should be the parents

Belief network structure learning (II)

- Given a total ordering, to determine *parents*(X_i) do independence tests to determine which features should be the parents
- XOR problem: just because features do not give information individually, does not mean they will not give information in combination

Belief network structure learning (II)

- Given a total ordering, to determine *parents*(X_i) do independence tests to determine which features should be the parents
- XOR problem: just because features do not give information individually, does not mean they will not give information in combination
- Search over total orderings of variables

Missing Data

- You cannot just ignore missing data unless you know it is missing at random.
- Is the reason data is missing correlated with something of interest?
- For example: data in a clinical trial to test a drug may be missing because:

Missing Data

- You cannot just ignore missing data unless you know it is missing at random.
- Is the reason data is missing correlated with something of interest?
- For example: data in a clinical trial to test a drug may be missing because:
  - the patient dies

Missing Data

- You cannot just ignore missing data unless you know it is missing at random.
- Is the reason data is missing correlated with something of interest?
- For example: data in a clinical trial to test a drug may be missing because:
  - the patient dies
  - the patient had severe side effects

Missing Data

- You cannot just ignore missing data unless you know it is missing at random.
- Is the reason data is missing correlated with something of interest?
- For example: data in a clinical trial to test a drug may be missing because:
  - the patient dies
  - the patient had severe side effects
  - the patient was cured

Missing Data

- You cannot just ignore missing data unless you know it is missing at random.
- Is the reason data is missing correlated with something of interest?
- For example: data in a clinical trial to test a drug may be missing because:
  - the patient dies
  - the patient had severe side effects
  - the patient was cured
  - the patient had to visit a sick relative.

Missing Data

- You cannot just ignore missing data unless you know it is missing at random.
- Is the reason data is missing correlated with something of interest?
- For example: data in a clinical trial to test a drug may be missing because:
  - the patient dies
  - the patient had severe side effects
  - the patient was cured
  - the patient had to visit a sick relative.

  — ignoring some of these may make the drug look better or worse than it is.

Missing Data

- You cannot just ignore missing data unless you know it is missing at random.
- Is the reason data is missing correlated with something of interest?
- For example: data in a clinical trial to test a drug may be missing because:
  - the patient dies
  - the patient had severe side effects
  - the patient was cured
  - the patient had to visit a sick relative.

  — ignoring some of these may make the drug look better or worse than it is.
- In general you need to model why data is missing (see Chapter 11)

General Learning of Belief Networks

- We have a mixture of observational data and data from randomized studies.

General Learning of Belief Networks

- We have a mixture of observational data and data from randomized studies.
- We are not given the structure.

General Learning of Belief Networks

- We have a mixture of observational data and data from randomized studies.
- We are not given the structure.
- We don't know whether there are hidden variables or not. We don't know the domain size of hidden variables.

General Learning of Belief Networks

- We have a mixture of observational data and data from randomized studies.
- We are not given the structure.
- We don't know whether there are hidden variables or not. We don't know the domain size of hidden variables.
- There is missing data.

General Learning of Belief Networks

- We have a mixture of observational data and data from randomized studies.
- We are not given the structure.
- We don't know whether there are hidden variables or not. We don't know the domain size of hidden variables.
- There is missing data.

... this is too difficult for current techniques!