- Learning in BNs
- Maximum likelihood (ML) estimation

  Estimate CPTs that maximize \( \text{"likelihood"} \)\n  \[
  \text{probability of observed data}
  \]

- Complete data

  \[
  \{(x_1(t), x_2(t), \ldots, x_n(t))\}_{t=1}^T \quad T \text{ complete instantiations of nodes } x_1, \ldots, x_n
  \]

- Notation

  Indicator function
  \[
  I(x, x') = \begin{cases} 
  1 & \text{if } x = x' \\
  0 & \text{otherwise}
  \end{cases}
  \]

- ML estimates

  \[
  P_{ML}(X_i = x \mid \text{pa}_i = \pi) = \begin{cases} 
  \frac{\text{count}(X_i = x, \text{pa}_i = \pi)}{\text{count}(\text{pa}_i = \pi)} & \text{for nodes w/ parents} \\
  \frac{\text{count}(X_i = x)}{T} & \text{for root nodes}
  \end{cases}
  \]

  \[
  = \frac{\sum_{t=1}^T I(x_i(t), x) I(\text{pa}_i(t), \pi)}{\sum_{t=1}^T I(\text{pa}_i(t), \pi)}
  \]

  \[
  = \frac{\sum_{t=1}^T I(x_i(t), x)}{T}
  \]
Ex: Naive Bayes model for document classification

* Variables
  \[ Y \in \{1, 2, \ldots, m\} \text{ possible document topics (e.g. } Y \in \{\text{spam, not spam}\}\)  
  \[ X_i \in \{0, 1\} \text{ = does } i\text{th word in vocabulary (dictionary) appear in document?} \]

* BN = DAG + CPTs

\[
\begin{align*}
  & Y \\
  & \downarrow \\
  & X_1 \quad X_2 \quad \ldots \quad X_n \\
  & \downarrow \\
  & \mathbb{P}(Y=y) \\
  & \mathbb{P}(X_i=1 \mid Y=y)
\end{align*}
\]

* ML estimation of CPTs

Collect and label a large corpus of \( N \) documents

\[
\begin{align*}
  \mathbb{P}_{\text{ML}}(Y=y) &= \text{fraction of documents with topic } y \\
  &= \frac{\text{count}(Y=y)}{N} \\
  &= \text{fraction of documents w/ topic } y
\end{align*}
\]

\[
\begin{align*}
  \mathbb{P}_{\text{ML}}(X_i=1 \mid Y=y) &= \frac{\text{count}(X_i=1, Y=y)}{\text{count}(Y=y)} \\
  &= \text{fraction of documents of topic } y \\
  & \text{that contain } i\text{th word in vocabulary}
\end{align*}
\]
\[ P(Y = y | X = x) = \frac{P(X = x | Y = y)P(Y = y)}{P(x)} \]

Bayes rule

\[ \leq \prod_{i=1}^{n} P(X_i = x_i | Y = y)P(Y = y) \]

conditional independence

\[ \leq P(X = x, Y = y') \]

marginalization

\[ \leq \prod_{y' \neq y} P(X_i = x_i | Y = y')P(Y = y') \]

product rule + cond. ind.

**Strengths of model**

1. Easy to estimate from a large corpus of documents
2. Simplest baseline

**Weaknesses of model**

1. Assumption that words appear independently given the topic (naive!)
2. "Bag of words" representation ignores order
3. Documents have only one topic
Ex: Markov models of language

Why do we need language models?

\[\text{MM} \rightarrow \text{speech recognizer} \rightarrow \text{eyelid in joyful} \leftrightarrow \text{Alvin lap hula} \rightarrow \text{I live in La Jolla}\]

- Let \( w_i \) denote word at \( i \)-th position in sentence
- How to model \( P(w_1, w_2, \ldots, w_L) \)?
  - Probability of sentence with \( L \) words \( w_1, w_2, \ldots, w_L \).

- Simplifying assumptions
  1. Finite context/memory
     \[
P(w_L | w_1, w_2, \ldots, w_{L-1}) = P(w_L | \underbrace{w_{L-(n-1)}, w_{L-(n-2)}, \ldots, w_{L-1}}_{\text{n-1 previous words}})\]

     special case: "bigram" model
     \[
P(w_L | w_1, \ldots, w_{L-1}) = P(w_L | w_{L-1})\]

  2. Position invariance
     \[
P(w_{L+1} = w' | w_L = w) = P(w_L = w' | w_{L-1} = w)\]

- BN for bigram model of language

\[
\begin{array}{ccccccc}
  w_1 & \rightarrow & w_2 & \rightarrow & \cdots & \rightarrow & w_{L-1} & \rightarrow & w_L \\
\end{array}
\]

same CPTs at all non-root nodes
- Learning bigram model
  - collect large corpus of text, \( \sim 10^8 \) words
  - vocabulary size \( V \sim 10^5 \) dictionary entries
  - count \( C_i = \# \text{ times word } i \text{ appears} \)
    \( C_{ij} = \# \text{ times word } j \text{ follows word } i \)
  - estimate \( P_{ML}(w_e = j \mid w_{e-1} = i) = \frac{C_{ij}}{C_i} \)

- Note: no generalization to unseen word combinations
  (will have 0 probability)

- "n-gram" model: condition on \( n-1 \) previous words
  \( n=1 \) unigram
  \( n=2 \) bigram
  \( n=3 \) trigram

  \[ P(w_e \mid w_{1}, \ldots, w_{e-1}) = P(w_e \mid w_{e-(n-1)}, \ldots, w_{e-1}) \]

  n-gram model counts get more sparse as \( n \) increases
ML estimation from incomplete data

- Given fixed graph (DAG) over discrete nodes \( \{X_1, X_2, \ldots, X_n\} \)
  Also data set of \( T \) partial instantiations of \( \{X_1, X_2, \ldots, X_n\} \)

\[
\begin{array}{cccc}
\text{Ex:} & X_1 & X_2 & X_3 & X_4 \\
1 & 0 & ? & 1 & 1 \\
2 & 1 & ? & ? & 1 \\
3 & 0 & ? & 1 & 1 \\
4 & ? & ? & ? & 0
\end{array}
\]

- Goal: estimate CPTs \( P(X_i = x | \text{pa}_i = \pi) \) that maximize the marginal (not joint) probability of partially observed data, (not complete)

- Variables in BN
  \( X = \) all nodes
  \( H = \) hidden nodes
  \( V = \) visible nodes

- Log-likelihood: assume \( T \) examples are i.i.d. from joint distribution \( P(X_1, X_2, \ldots, X_n) \)

\[
L = \log P(\text{data}) = \sum \log P(V(t) = v(t)) = \sum \log P(V(t) = v(t))
\]
\[
\sum_{t=1}^{T} \log \sum_{h} P(V(t) = v(t), H(t) = h) \]
marginalizing over joint for hidden nodes \(X = HUV\)

\[
\sum_{t=1}^{T} \log \sum_{h} \prod_{i=1}^{T} P(X_i = x_i | pa_i = \pi_i) \bigg| V(t) = v(t), H(t) = h
\]

- more complicated to optimize \(L\) from incomplete data
- no "closed-form" solution
- iterative solution

**Expectation-Maximization (EM) algorithm**

Iterative procedure to maximize \(L(data)\) for incomplete data in terms of CPTs.

By analogy, ML estimates for complete data

\[
P_{ML}(X_i = x | pa_i = \pi_i) = \frac{\text{count}(X_i = x, pa_i = \pi_i)}{\text{count}(pa_i = \pi_i)} = \frac{\sum_{t=1}^{T} I(X_i^{(t)}, x) I(pa_i^{(t)}, \pi_i)}{\sum_{t=1}^{T} I(pa_i^{(t)}, \pi_i)}
\]

For incomplete data, we must "fill in" hidden values:

\[
P_{ML}(X_i = x | pa_i = \pi_i) \leftarrow \frac{\sum_{t=1}^{T} P(X_i = x, pa_i = \pi_i | V = v(t))}{\sum_{t=1}^{T} P(pa_i = \pi_i | V = v(t))}
\]

Intuition: expected statistics ("counts") under \(P(H|V)\)

Substitute for observed counts in complete data case