

Statistical Learning (part II)

November 10, 2009

CS 486/686

University of Waterloo

Outline

- Learning from incomplete Data
 - EM algorithm
- Reading: R&N Ch 20.3

Incomplete data

- So far...
 - Values of all attributes are known
 - Learning is relatively easy
- But many real-world problems have **hidden variables** (a.k.a **latent variables**)
 - Incomplete data
 - Values of some attributes missing

Unsupervised Learning

- Incomplete data → unsupervised learning
- Examples:
 - Categorisation of stars by astronomers
 - Categorisation of species by anthropologists
 - Market segmentation for marketing
 - Pattern identification for fraud detection
 - Research in general!

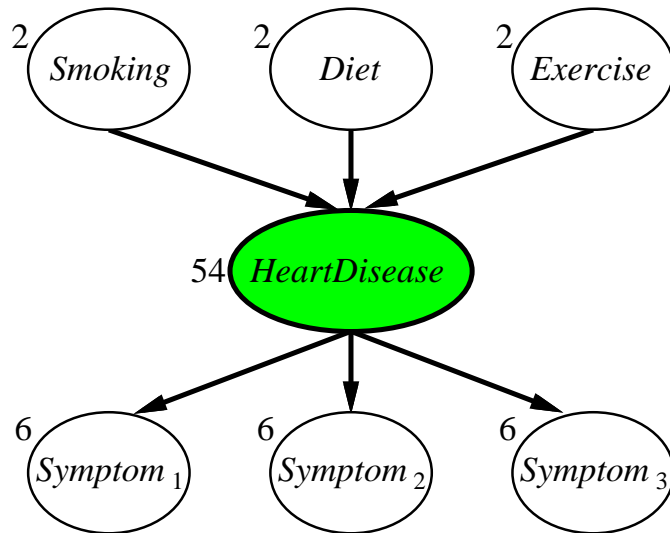
Maximum Likelihood Learning

- ML learning of Bayes net parameters:
 - For $\theta_{V=\text{true}, \text{pa}(V)=\mathbf{v}} = \Pr(V=\text{true} | \text{pa}(V) = \mathbf{v})$
 - $\theta_{V=\text{true}, \text{pa}(V)=\mathbf{v}} = \frac{\#[V=\text{true}, \text{pa}(V)=\mathbf{v}]}{\#[V=\text{true}, \text{pa}(V)=\mathbf{v}] + \#[V=\text{false}, \text{pa}(V)=\mathbf{v}]}$
 - Assumes all attributes have values...
- What if values of some attributes are missing?

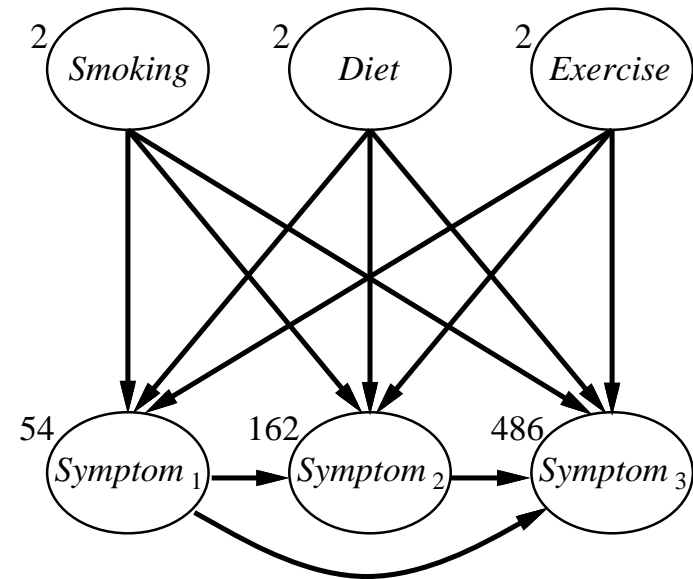
"Naive" solutions for incomplete data

- Solution #1: Ignore records with missing values
 - But what if all records are missing values (i.e., when a variable is hidden, none of the records have any value for that variable)
- Solution #2: Ignore hidden variables
 - Model may become significantly more complex!

Heart disease example



(a)



(b)

- a) simpler (i.e., fewer CPT parameters)
- b) complex (i.e., lots of CPT parameters)

"Direct" maximum likelihood

- Solution 3: maximize likelihood directly
 - Let \mathbf{Z} be hidden and \mathbf{E} observable
 - $h_{ML} = \operatorname{argmax}_h P(\mathbf{e}|h)$
 - $= \operatorname{argmax}_h \sum_{\mathbf{Z}} P(\mathbf{e}, \mathbf{Z}|h)$
 - $= \operatorname{argmax}_h \sum_{\mathbf{Z}} \prod_i CPT(V_i)$
 - $= \operatorname{argmax}_h \log \sum_{\mathbf{Z}} \prod_i CPT(V_i)$
 - Problem: can't push log past sum to linearize product

Expectation-Maximization (EM)

- Solution #4: EM algorithm
 - Intuition: if we knew the missing values, computing h_{ML} would be trivial
- Guess h_{ML}
- Iterate
 - **Expectation:** based on h_{ML} , compute expectation of the missing values
 - **Maximization:** based on expected missing values, compute new estimate of h_{ML}

Expectation-Maximization (EM)

- More formally:

- Approximate maximum likelihood

- Iteratively compute:

$$h_{i+1} = \operatorname{argmax}_h \underbrace{\sum_{\mathbf{Z}} P(\mathbf{Z}|h_i, \mathbf{e}) \log P(\mathbf{e}, \mathbf{Z}|h)}_{\text{Expectation}}$$

Expectation

Maximization

Expectation-Maximization (EM)

- Derivation

$$\begin{aligned} -\log P(\mathbf{e}|\mathbf{h}) &= \log [P(\mathbf{e}, \mathbf{Z}|\mathbf{h}) / P(\mathbf{Z}|\mathbf{e}, \mathbf{h})] \\ &= \log P(\mathbf{e}, \mathbf{Z}|\mathbf{h}) - \log P(\mathbf{Z}|\mathbf{e}, \mathbf{h}) \\ &= \sum_{\mathbf{Z}} P(\mathbf{Z}|\mathbf{e}, \mathbf{h}) \log P(\mathbf{e}, \mathbf{Z}|\mathbf{h}) \\ &\quad - \sum_{\mathbf{Z}} P(\mathbf{Z}|\mathbf{e}, \mathbf{h}) \log P(\mathbf{Z}|\mathbf{e}, \mathbf{h}) \\ &\geq \sum_{\mathbf{Z}} P(\mathbf{Z}|\mathbf{e}, \mathbf{h}) \log P(\mathbf{e}, \mathbf{Z}|\mathbf{h}) \end{aligned}$$

- EM finds a **local maximum** of $\sum_{\mathbf{Z}} P(\mathbf{Z}|\mathbf{e}, \mathbf{h}) \log P(\mathbf{e}, \mathbf{Z}|\mathbf{h})$ which is a **lower bound** of $\log P(\mathbf{e}|\mathbf{h})$

Expectation-Maximization (EM)

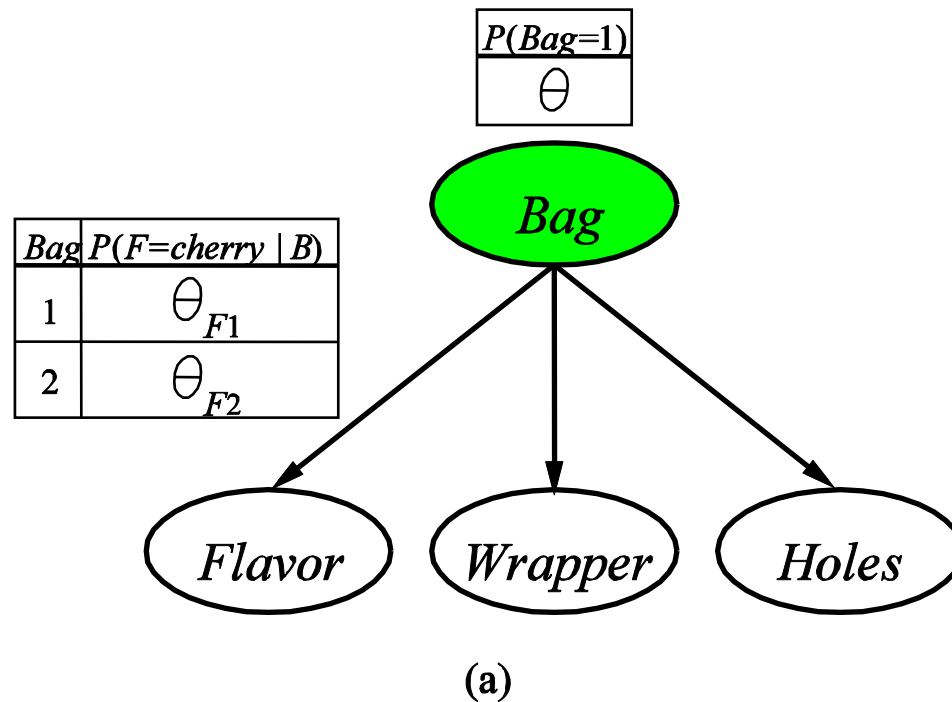
- Log inside sum can linearize product
 - $h_{i+1} = \operatorname{argmax}_h \sum_Z P(Z|h_i, \mathbf{e}) \log P(\mathbf{e}, Z|h)$
 - = $\operatorname{argmax}_h \sum_Z P(Z|h_i, \mathbf{e}) \log \prod_j CPT_j$
 - = $\operatorname{argmax}_h \sum_Z P(Z|h_i, \mathbf{e}) \sum_j \log CPT_j$
- Monotonic improvement of likelihood
 - $P(\mathbf{e}|h_{i+1}) \geq P(\mathbf{e}|h_i)$

Candy Example

- Suppose you buy two bags of candies of unknown type (e.g. flavour ratios)
- You plan to eat sufficiently many candies of each bag to learn their type
- Ignoring your plan, your roommate mixes both bags...
- How can you learn the type of each bag despite being mixed?

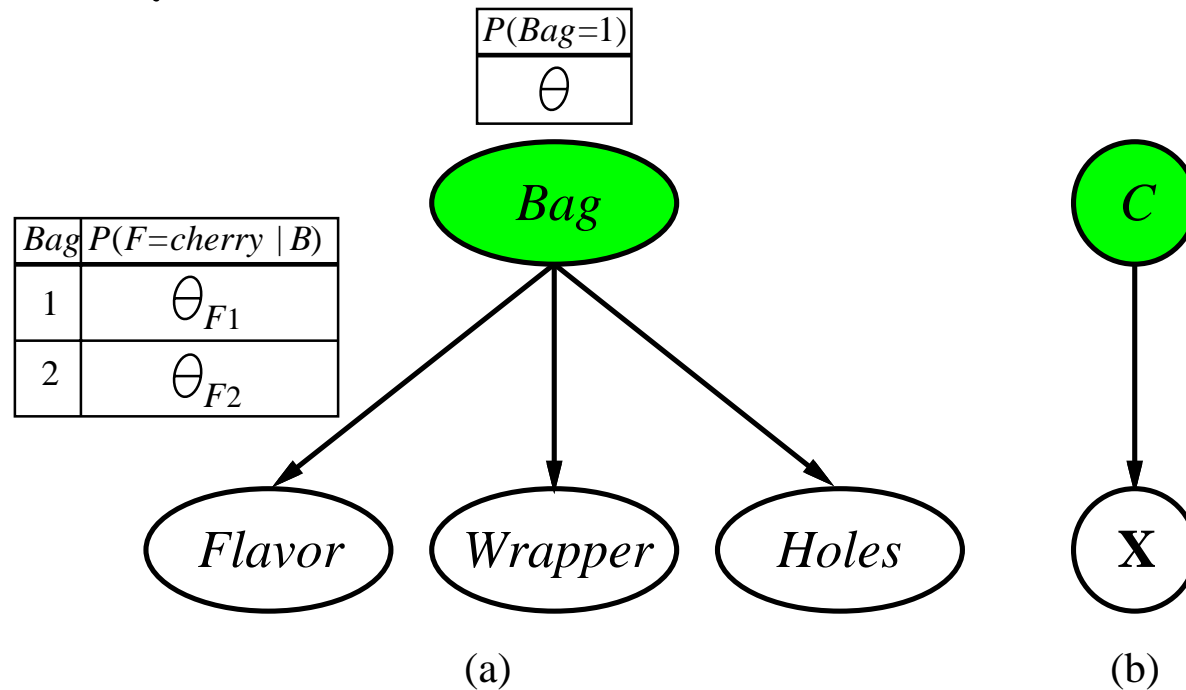
Candy Example

- "Bag" variable is hidden



Unsupervised Clustering

- "Class" variable is hidden
- Naive Bayes model



Candy Example

- Unknown Parameters:
 - $\theta_i = P(\text{Bag}=i)$
 - $\theta_{Fi} = P(\text{Flavour}=\text{cherry} | \text{Bag}=i)$
 - $\theta_{Wi} = P(\text{Wrapper}=\text{red} | \text{Bag}=i)$
 - $\theta_{Hi} = P(\text{Hole}=\text{yes} | \text{Bag}=i)$
- When eating a candy:
 - F , W and H are observable
 - B is hidden

Candy Example

- Let true parameters be:
 - $\theta = 0.5$, $\theta_{F1} = \theta_{W1} = \theta_{H1} = 0.8$, $\theta_{F2} = \theta_{W2} = \theta_{H2} = 0.3$
- After eating 1000 candies:

	W=red		W=green	
	H=1	H=0	H=1	H=0
F=cherry	273	93	104	90
F=lime	79	100	94	167

Candy Example

- EM algorithm
- Guess h_0 :
 - $\theta=0.6, \theta_{F1}=\theta_{W1}=\theta_{H1}=0.6, \theta_{F2}=\theta_{W2}=\theta_{H2}=0.4$
- Alternate:
 - Expectation: expected # of candies in each bag
 - Maximization: new parameter estimates

Candy Example

- Expectation: expected # of candies in each bag
 - $\#[\text{Bag}=i] = \sum_j P(B=i|f_j, w_j, h_j)$
 - Compute $P(B=i|f_j, w_j, h_j)$ by variable elimination (or any other inference alg.)
- Example:
 - $\#[\text{Bag}=1] = 612$
 - $\#[\text{Bag}=2] = 388$

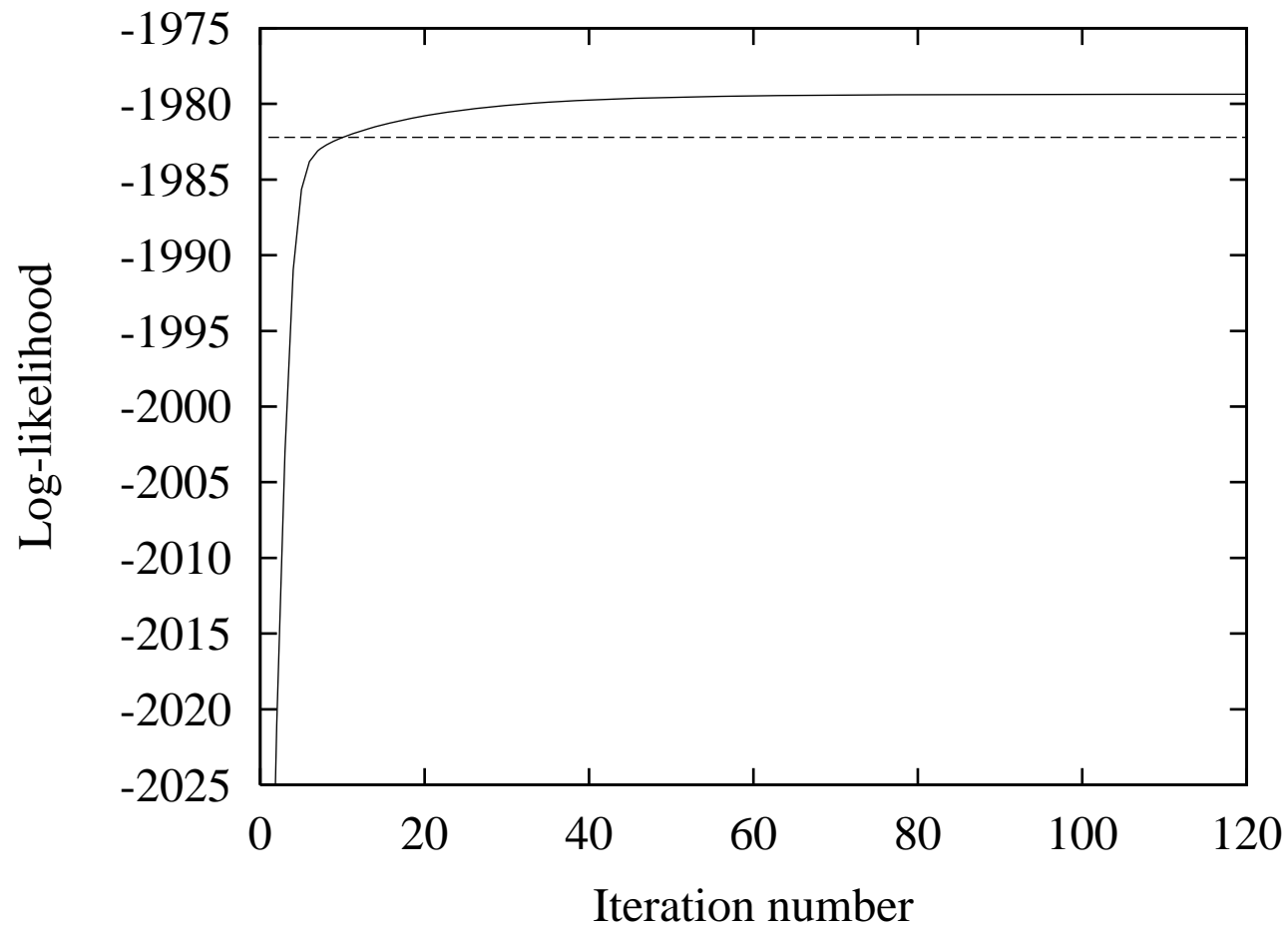
Candy Example

- Maximization: relative frequency of each bag
 - $\theta_1 = 612/1000 = 0.612$
 - $\theta_2 = 388/1000 = 0.388$

Candy Example

- Expectation: expected # of cherry candies in each bag
 - $\#[B=i, F=\text{cherry}] = \sum_j P(B=i | f_j=\text{cherry}, w_j, h_j)$
 - Compute $P(B=i | f_j=\text{cherry}, w_j, h_j)$ by variable elimination (or any other inference alg.)
- Maximization:
 - $\theta_{F_1} = \#[B=1, F=\text{cherry}] / \#[B=1] = 0.668$
 - $\theta_{F_2} = \#[B=2, F=\text{cherry}] / \#[B=2] = 0.389$

Candy Example



Bayesian networks

- EM algorithm for general Bayes nets
- Expectation:
 - $\#[V_i=v_{ij}, Pa(V_i)=pa_{ik}] = \text{expected frequency}$
- Maximization:
 - $\theta_{v_{ij}, pa_{ik}} = \#[V_i=v_{ij}, Pa(V_i)=pa_{ik}] / \#[Pa(V_i)=pa_{ik}]$

Next Class

- Next Class:
 - Markov networks