Statistical Learning (part II)

July 5, 2005

CS 486/686

University of Waterloo

Outline

- Learning from complete 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 $\theta_{V=true,pa(V)=\mathbf{v}} = Pr(V=true|par(V) = \mathbf{v})$
  - $\theta_{V=true,pa(V)=\mathbf{v}} = \dfrac{\#[V=true,pa(V)=\mathbf{v}]}{\#[V=true,pa(V)=\mathbf{v}] + \#[V=false,pa(V)=\mathbf{v}]}$
  - Assumes all attributes have values…

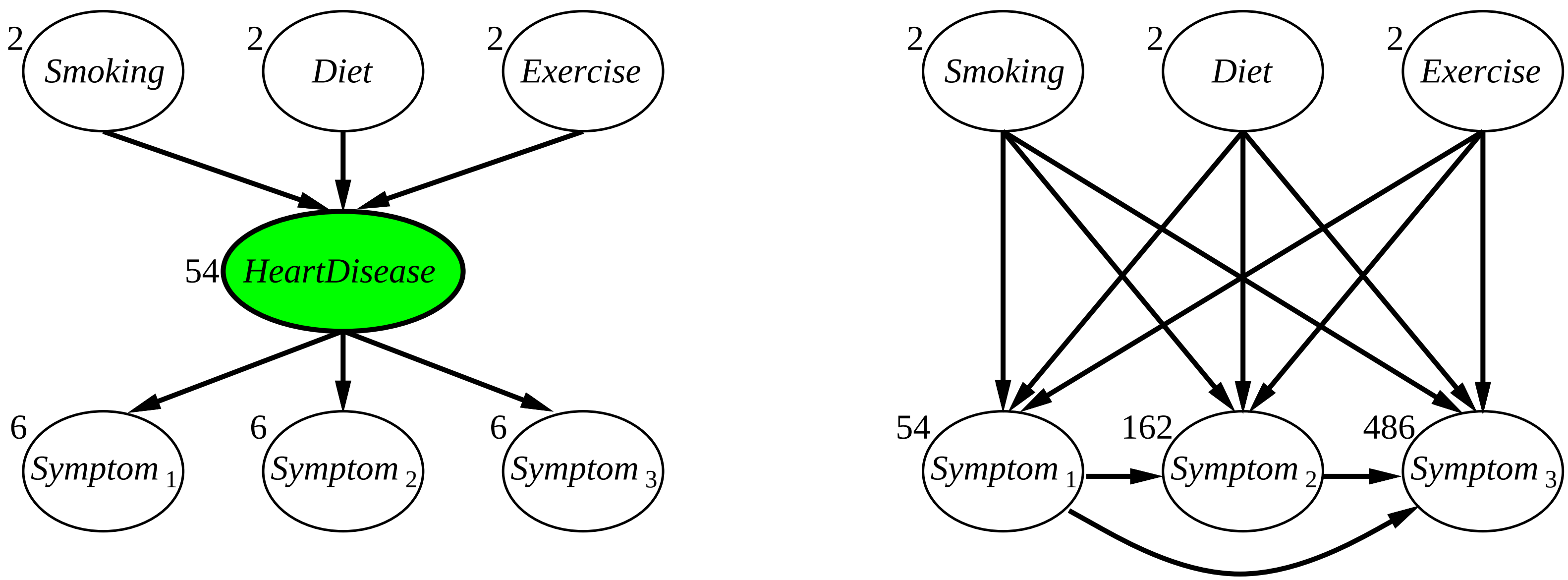
- What if values of some attributes are missing?

“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 **Z** be hidden and **E** observable
  - $h_{ML} = \text{argmax}_h\, P(\mathbf{e}|h)$
    $= \text{argmax}_h\, \Sigma_{\mathbf{Z}}\, P(\mathbf{e},\mathbf{Z}|h)$
    $= \text{argmax}_h\, \Sigma_{\mathbf{Z}}\, \Pi_i\, CPT(V_i)$
    $= \text{argmax}_h\, \log \Sigma_{\mathbf{Z}}\, \Pi_i\, CPT(V_i)$
  - Problem: can’t push log past sum to linearize product

Expectation-Maximization (EM)

- Solution #4: EM algorithm
  - Intuition: if we new the missing values, computing h_{ML} would be trival
- 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:

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

Expectation-Maximization (EM)

- Derivation
  - $\log P(\mathbf{e}|h) = \log [P(\mathbf{e},\mathbf{Z}|h) \ / \ P(\mathbf{Z}|\mathbf{e},h)]$
    $= \log P(\mathbf{e},\mathbf{Z}|h) - \log P(\mathbf{Z}|\mathbf{e},h)$
    $= \Sigma_{\mathbf{Z}} P(\mathbf{Z}|\mathbf{e},h) \log P(\mathbf{e},\mathbf{Z}|h)$
    $\quad - \Sigma_{\mathbf{Z}} P(\mathbf{Z}|\mathbf{e},h) \log P(\mathbf{Z}|\mathbf{e},h)$
    $\geq \Sigma_{\mathbf{Z}} P(\mathbf{Z}|\mathbf{e},h) \log P(\mathbf{e},\mathbf{Z}|h)$

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

Expectation-Maximization (EM)

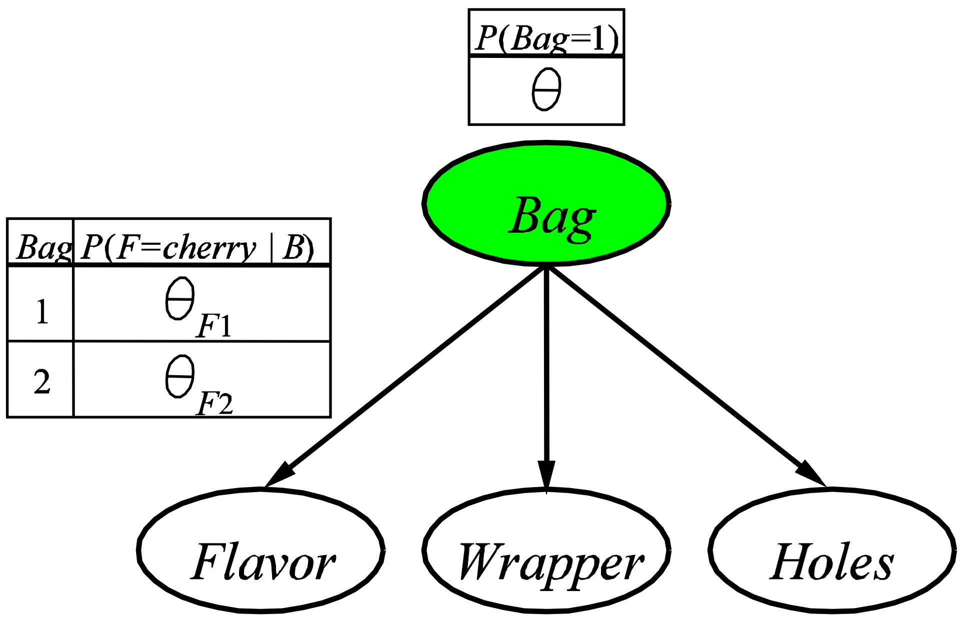
- Log inside sum can linearize product
  - $h_{i+1} = \text{argmax}_h \Sigma_{\mathbf{Z}} P(\mathbf{Z}|h_i,\mathbf{e}) \log P(\mathbf{e},\mathbf{Z}|h)$
    $= \text{argmax}_h \Sigma_{\mathbf{Z}} P(\mathbf{Z}|h_i,\mathbf{e}) \log \Pi_j CPT_j$
    $= \text{argmax}_h \Sigma_{\mathbf{Z}} P(\mathbf{Z}|h_i,\mathbf{e}) \Sigma_j \log CPT_j$
- 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

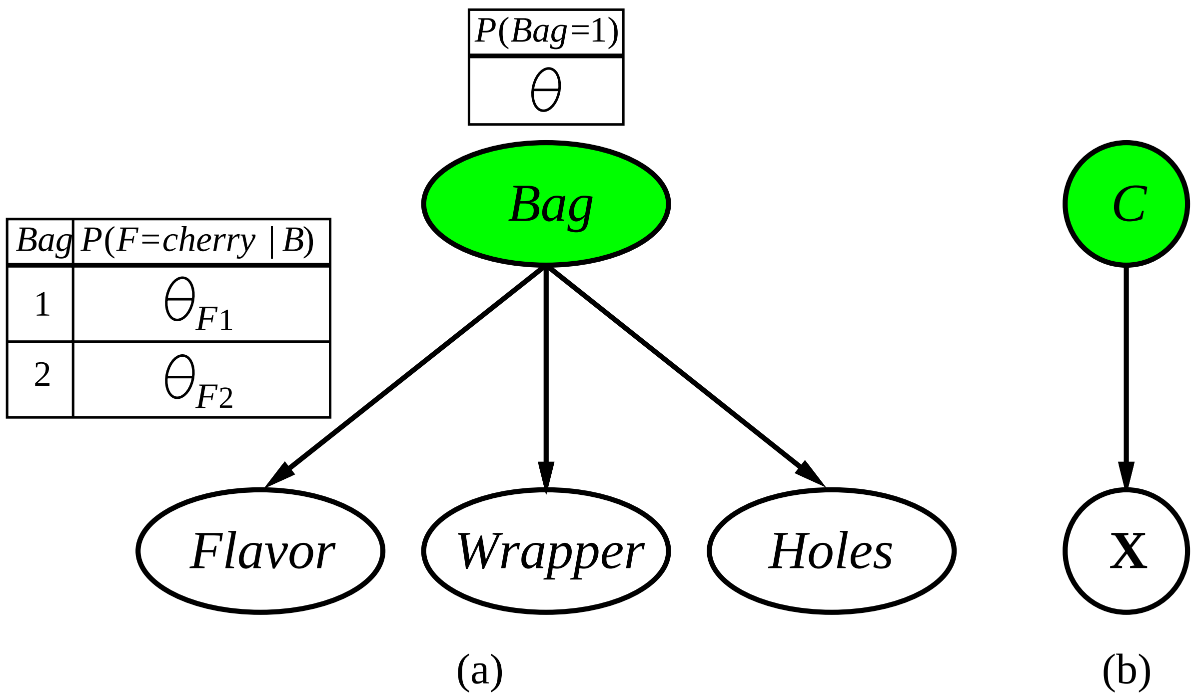
- "Bag" variable is hidden



(a)

Unsupervised Clustering

- “Class” variable is hidden
- Naïve Bayes model



Candy Example

- Unknown Parameters:
  - θ_i = P(Bag=i)
  - θ_{Fi} = P(Flavour=cherry|Bag=i)
  - θ_{Wi} = P(Wrapper=red|Bag=i)
  - θ_{Hi} = P(Hole=yes|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:
  - θ=0.6, θ_{F1}=θ_{W1}=θ_{H1}=0.6, θ_{F2}=θ_{W2}=θ_{H2}=0.4
- Alternate:
  - Expectation: expected # of candies in each bag
  - Maximization: new parameter estimates

Candy Example

- Expectation: expected # of candies in each bag
  - $\#[Bag=i] = \Sigma_j P(B=i|f_j,w_j,h_j)$
  - Compute $P(B=i|f_j,w_j,h_j)$ by variable elimination (or any other inference alg.)

- Example:
  - $\#[Bag=1] = 612$
  - $\#[Bag=2] = 388$

Candy Example

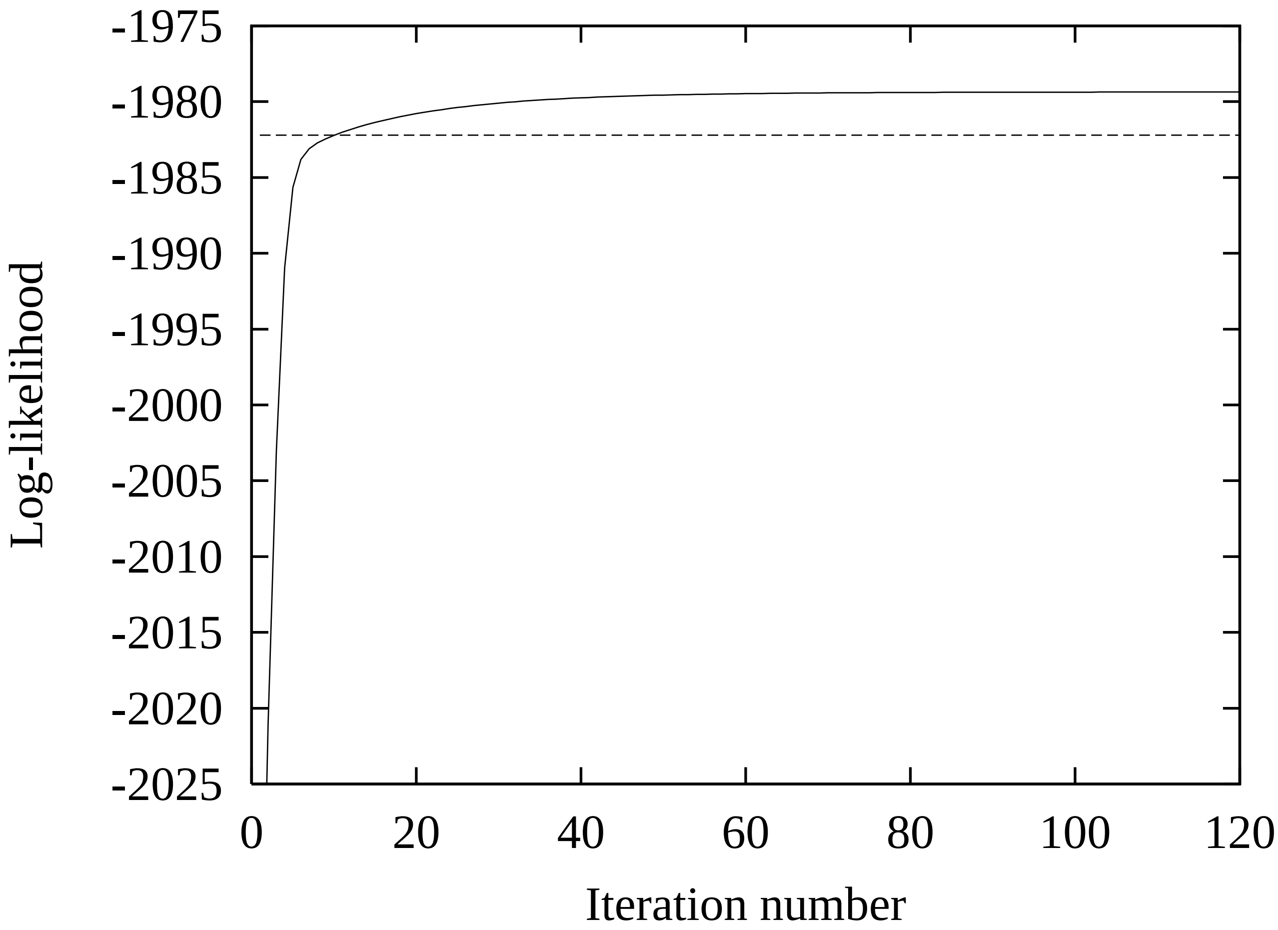
- Maximization: relative frequency of each bag
  - $\theta_1 = 612/1000 = 0.612$
  - $\theta_2 = 388/1000 = 0.388$

Candy Example

- Expectation: expected # of cherry candies in each bag
  - $\#[B=i,F=cherry] = \Sigma_j P(B=i|f_j=cherry,w_j,h_j)$
  - Compute $P(B=i|f_j=cherry,w_j,h_j)$ by variable elimination (or any other inference alg.)

- Maximization:
  - $\theta_{F1} = \#[B=1,F=cherry] / \#[B=1] = 0.668$
  - $\theta_{F2} = \#[B=2,F=cherry] / \#[B=2] = 0.389$

Candy Example



Bayesian networks

- EM algorithm for general Bayes nets
- Expectation:
  - $\#[V_i=v_{ij},Pa(V_i)=pa_{ik}]$ = 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:
  - Neural networks
  - Russell and Norvig Sect. 20.5