STATISTICAL METHODS IN AI/ML

Vibhav Gogate
University of Texas, Dallas

LEARNING: Lecture 3
What we will cover today?

Already covered: MLE approach for learning the parameters of a Bayesian network given fully observed data.

- MLE approach for learning the parameters of a Bayesian network
  - Partially observed data
  - Expectation Maximization
  - Gradient Ascent
- MLE approach for learning the structure of a Bayesian network
  - Fully observed data (today and next class)
  - Partially observed data (next class)
Partially Observed Data (POD)

- Missing data, hidden variables
- $H, T, H, ?, T, ?, \ldots$
- Why is the data missing?
  - Randomly missing
  - Deliberately missing
Why is parameter learning in presence of POD challenging?

Likelihood function for POD:

\[ L(\theta, \mathcal{X}) = \prod_{j=1}^{m} \sum_{y \notin x^{(j)}} \Pr(x^{(j)}, y) \]

Compare with Likelihood function for FOD:

\[ L(\theta, \mathcal{X}) = \prod_{j=1}^{m} \Pr(x^{(j)}) \]

Likelihood function for POD:

- is not unimodal.
- cannot be expressed in closed form
- is not decomposable
Why is parameter learning in presence of POD challenging?

**POD case:**
Each point in the sum yields a unimodal distribution. When combined, we get a multi-modal distribution.

- The optimization problem, a.k.a. maximizing our objective, the likelihood of the data is hard. We need an iterative approach.

**FOD case:**
Unimodal distribution
Approach 1: The Expectation Maximization (EM) Algorithm

- Start with random parameters
- Repeat until convergence
  1. Complete the incomplete data using current parameters.
  2. Update the parameters based on the completed data

STEP 1: computes **expected** sufficient statistics (E-step)
STEP 2: **maximizes** the likelihood (M-step)
The Expectation Maximization Algorithm: Example

Data instance: \((a, ?, ?, \bar{d})\)

How to complete this example?

For each possible completion

- **STEP 1**: Compute how likely the completion is.
- **STEP 2**: Data set is now weighted (just as in importance sampling)

\[
\begin{align*}
\theta_a &= .3 \\
\theta_b &= .9 \\
\theta_{c|\bar{a},\bar{b}} &= .83 \\
\theta_{c|\bar{a},b} &= .09 \\
\theta_{c|a,\bar{b}} &= .6 \\
\theta_{c|a,b} &= .2 \\
\theta_{d|\bar{c}} &= .1 \\
\theta_{d|c} &= .8
\end{align*}
\]
The Expectation Maximization Algorithm: E-Step

- Data set is now **bigger and weighted**
- \((a, ?, ?, \bar{d})\) corresponds to four weighted examples
  - \((a, b, c, \bar{d})\), weight = .0492
  - \((a, b, \bar{c}, \bar{d})\), weight = .8852
  - \((a, \bar{b}, c, \bar{d})\), weight = .0164
  - \((a, b, \bar{c}, \bar{d})\), weight = .0492
The Expectation Maximization Algorithm: M-Step

Updating: $\theta_{d|\bar{c}}$

- Unweighted MLE estimate:

$$\theta_{d|\bar{c}} = \frac{\#(d, \bar{c})}{\#(\bar{c})}$$

- Weighted MLE estimate:

$$\theta_{d|\bar{c}} = \frac{\text{sum} - \text{weight}(d, \bar{c})}{\text{sum} - \text{weight}(\bar{c})} = \frac{\sum_{j=1}^{m} \Pr_{\theta}(d, \bar{c}|x^{(j)})}{\sum_{j=1}^{m} \Pr_{\theta}(\bar{c}|x^{(j)})}$$
The EM Algorithm

**Procedure** Compute-ESS ()

\[ G, \quad \text{// Bayesian network structure over } X_1, \ldots, X_n \]
\[ \theta, \quad \text{// Set of parameters for } G \]
\[ D, \quad \text{// Partially observed data set} \]

// Initialize data structures
for each \( i = 1, \ldots, n \)
  for each \( x_i, u_i \in Val(X_i, Pa_X^G) \)
    \[ M[x_i, u_i] \leftarrow 0 \]
  // Collect probabilities from all instances
  for each \( m = 1 \ldots M \)
    Run inference on \( \langle G, \theta \rangle \) using evidence \( o[m] \)
    for each \( i = 1, \ldots, n \)
      for each \( x_i, u_i \in Val(X_i, Pa_X^G) \)
        \[ M[x_i, u_i] \leftarrow M[x_i, u_i] + P(x_i, u_i \mid o[m]) \]
  return \( \{M[x_i, u_i] : \forall i = 1, \ldots, n, \forall x_i, u_i \in Val(X_i, Pa_X^G)\} \)
The EM Algorithm

**Procedure** Expectation-Maximization (\(G, \theta^0, D\))

1. for each \(t = 0, 1, \ldots\), until convergence
2. // E-step
3. \(\{\tilde{M}_t[x_i, u_i]\} \leftarrow \text{Compute-ESS}(G, \theta^t, D)\)
4. // M-step
5. for each \(i = 1, \ldots, n\)
6. for each \(x_i, u_i \in \text{Val}(X_i, Pa_{X_i}^G)\)
7. \(\theta^{t+1}_{x_i | u_i} \leftarrow \frac{\tilde{M}_t[x_i, u_i]}{\tilde{M}_t[u_i]}\)
8. return \(\theta^t\)
EM: Properties

- EM may converge to different parameters, with different likelihoods, depending on the initial estimates $\theta^{(0)}$ that it starts with.
- Each iteration of the EM algorithm will have to perform inference on a Bayesian network.
- In each iteration, the algorithm computes the probability of each instantiation $(x, u)$ given each example as evidence.
- All of these computations correspond to posterior marginals over network families.
EM: Properties

- EM parameter estimates are the only estimates that maximize the expected log-likelihood function.
- EM is indeed searching for estimates that maximize the expected log-likelihood function, which also explains its name.
- Parameters that maximize the expected log-likelihood function cannot decrease the log-likelihood function.
  - Each iteration of EM can only increase the likelihood and never decrease it.
  - It will always converge to a local maxima.
Gradient Ascent

- A generic optimization algorithm
- Operates by moving the parameters in the direction of the gradient.

How to compute the gradient?
For a data instance:

\[
\frac{\partial \Pr(o)}{\partial \Pr(x|u)} = \frac{1}{\Pr(x|u)} \Pr(x, u, o)
\]

For a data-set:

\[
\frac{\partial \log - L(\theta, X)}{\partial \Pr(x|u)} = \frac{1}{\Pr(x|u)} \sum_{i=1}^{m} \Pr(x, u|x^{(i)})
\]
Gradient Ascent: Example

\[ \theta_a = 0.3 \]
\[ \theta_b = 0.9 \]
\[ \theta_{c|\bar{a},\bar{b}} = 0.83 \]
\[ \theta_{c|\bar{a},b} = 0.09 \]
\[ \theta_{c|a,\bar{b}} = 0.6 \]
\[ \theta_{c|a,b} = 0.2 \]
\[ \theta_d|\bar{c} = 0.1 \]
\[ \theta_d|c = 0.8 \]

Data instance: \((a, ?, ?, \bar{d})\)

Gradient w.r.t. \(\theta_{d|\bar{c}}\) = ?
Gradient w.r.t. \(\theta_{\bar{d}|\bar{c}}\) = ?
Gradient w.r.t. \(\theta_{d|c}\) = ?
Gradient w.r.t. \(\theta_{\bar{d}|c}\) = ?
Gradient Ascent: Algorithm for computing the gradient

Algorithm 19.1 Computing the gradient in a network with table-CPDs

```
Procedure Compute-Gradient :
    $\mathcal{G}$, // Bayesian network structure over $X_1, \ldots, X_n$
    $\theta$, // Set of parameters for $\mathcal{G}$
    $\mathcal{D}$ // Partially observed data set

1 )
    // Initialize data structures
2     for each $i = 1, \ldots, n$
3         for each $x_i, u_i \in \text{Val}(X_i, \text{Pa}_{X_i})$
4             $\overline{M}[x_i, u_i] \leftarrow 0$
5     // Collect probabilities from all instances
6     for each $m = 1 \ldots M$
7         Run clique tree calibration on $(\mathcal{G}, \theta)$ using evidence $o[m]$
8            for each $i = 1, \ldots, n$
9                for each $x_i, u_i \in \text{Val}(X_i, \text{Pa}_{X_i})$
10                   $\overline{M}[x_i, u_i] \leftarrow \overline{M}[x_i, u_i] + P(x_i, u_i | o[m])$
11             // Compute components of the gradient vector
12            for each $i = 1, \ldots, n$
13                for each $x_i, u_i \in \text{Val}(X_i, \text{Pa}_{X_i})$
14                    $\delta_{x_i u_i} \leftarrow \frac{1}{\delta_{x_i u_i}} \overline{M}[x_i, u_i]$
15     return $\{\delta_{x_i u_i} : \forall i = 1, \ldots, n, \forall (x_i, u_i) \in \text{Val}(X_i, \text{Pa}_{X_i})\}$
```
Gradient Ascent: Algorithm

Algorithm A.10 Simple gradient ascent algorithm

Procedure Gradient-Ascent ( 
\[ \theta^1 \], \hspace{0.5cm} // Initial starting point 
\[ f_{\text{obj}} \], \hspace{0.5cm} // Function to be optimized 
\[ \delta \] \hspace{0.5cm} // Convergence threshold 
)

1. \( t \leftarrow 1 \)
2. do
3. \( \theta^{t+1} \leftarrow \theta^t + \eta \nabla f_{\text{obj}}(\theta^t) \)
4. \( t \leftarrow t + 1 \)
5. while \( \| \theta^t - \theta^{t-1} \| > \delta \)
6. return (\( \theta^t \))