Learning with Partially Observed Data

Lecture 12 – May 4, 2011
CSE 515, Statistical Methods, Spring 2011
Instructor: Su-In Lee
University of Washington, Seattle

Model Selection

- So far, we focused on single model
  - Given \( D = \{ X[1], \ldots, X[M] \} \), find best scoring model \( \tilde{G} = \arg \max G \) \( P(G | D) \)
  - Use it to predict next example \( P(X[M + 1] | D, \tilde{G}) \)
- Implicit assumption
  - Making predictions based on the Bayesian estimation rule:
    \[
    P(X[M + 1] | D) \approx \sum_G P(X[M + 1] | D, G) P(G | D)
    \]
    \( P(X[M + 1] | D) \approx P(X[M + 1] | D, \tilde{G}) \)
  - Valid with many data instances (very large \( M \))
- Pros:
  - We get a single structure
  - Allows for efficient use in our prediction tasks
- Cons:
  - Committing to the independencies of a particular structure
  - Other structures with similar score might be probable given \( D \)
Model Selection

- **Density estimation**
  - Picking one structure may suffice if it distribution $P(X|M+1|D,G)$ is similar for different high-scoring structures.

- **Structure discovery**
  - Several networks with similar scores → one or several of them might be close to the “true” structure, but we cannot distinguish between them given the data D.
  - Drawing a conclusion about the structure from one of the networks can be wrong
  - Thus, instead of picking one of the high-scoring structures, we should focus on estimating the “confidence” of the structural properties we are interested in.

- Define features $f(G)$ (e.g., edge, sub-structure, d-sep property)
- Compute $P(f | D) = \sum_{G} f(G) P(G | D)$
  - Requires summing over exponentially many structures
  - We can reduce the computation assuming a certain ordering

Model Averaging Given an Order

- **Assumptions**
  - Known total order of variables $\alpha$
  - Maximum in-degree for variables $d$

- **Marginal likelihood**
  
  $P(D | \alpha) = \sum_{G \in G_{\alpha}} P(D | G) P(G | \alpha)$
  
  $= \sum_{G \in G_{\alpha}} \prod_{i=1}^{n} \exp\{\text{FamScore}_b(X_i | Pa_{X_i}^{G} : D)\}$

  $\sum_{i=1}^{n} (f_{G}^{(1)} f_{G}^{(2)} \cdots f_{G}^{(m)})$

  $= (f_{1}^{(1)} + \cdots + f_{n}^{(m)}) \cdots (f_{1}^{(1)} + \cdots + f_{n}^{(m)})$

  Cost per family: $O(n^d)$

  Total cost: $O(n^{d+1})$

Using decomposability assumption on prior $P(G | \alpha)$

Since given ordering $\alpha$, parent choices are independent
Model Averaging Given an Order

- Posterior probability of a general feature $f$

$$P(f | \alpha, D) = \frac{P(f, D | \alpha)}{P(D | \alpha)} = \sum_{f(G)} P(f, D | G) P(G | \alpha)$$

$$\prod_{U \subseteq \{X_i \cup X_j \cup \{U \cup U \cup d\}} \exp \left[ \text{FamScore}_a(X_i | \{U \cup U \cup d\}) \right]$$

- $f$: particular choice of parents $U$ for $X_i$

$$P(Pa_{X_i} = U | D, \alpha) = \frac{\exp \left[ \text{FamScore}_a(X_i | \{U \cup U \cup d\}) \right]}{\sum_{U \subseteq \{X_i \cup X_j \cup \{U \cup U \cup d\}} \exp \left[ \text{FamScore}_a(X_i | \{U \cup U \cup d\}) \right]}$$

- $f$: existence of a particular edge between $X_i \rightarrow X_j$

$$P(X_j \in Pa_{X_i} | D, \alpha) = \frac{\sum_{U \subseteq \{X_i \cup X_j \cup \{U \cup U \cup d\}} \exp \left[ \text{FamScore}_a(X_i | \{U \cup U \cup d\}) \right]}{\sum_{U \subseteq \{X_i \cup X_j \cup \{U \cup U \cup d\}} \exp \left[ \text{FamScore}_a(X_i | \{U \cup U \cup d\}) \right]}$$

Model Averaging

- We cannot assume that order is known $\alpha$

- Solution: Sample from posterior distribution of $P(G | D)$

  - If we manage to sample graphs $G_1, \ldots, G_K$ from $P(G | D)$

  - Estimate feature probability by

    $$P(f | D) \approx \frac{1}{K} \sum_{i=1}^{K} f(G_i)$$

  - Sampling can be done by MCMC (Markov chain Monte Carlo)

    - Next week
Notes on Learning Local Structures

- Beyond table CPDs

- Define score with local structures
  - Example: in tree CPDs, score decomposes by leaves (not by $X_i$ and a particular value on Par $X_i$)

- Prior may need to be extended
  - Example: in tree CPDs, penalty for tree structure per CPD (depth of the tree)

- Extend search operators to local structure
  - Example: in tree CPDs, we need to search for tree structure
  - Can be done by local encapsulated search or by defining new global operations

Structure Search: Summary

- Discrete optimization problem

- In general, NP-Hard
  - Need to resort to heuristic search
  - In practice, search is relatively fast (~100 vars in ~10 min)
    - Decomposability
    - Sufficient statistics

- In some cases, we can reduce the search problem to an easy optimization problem
  - Example: learning trees, a fixed ordering $\alpha$
Let’s turn to the main topic for today...

LEARNING WITH PARTIALLY OBSERVED DATA

Training Data $D$

- Until now, we assumed that the training data is **fully observed**
  - Each instance assigns values to all the variables in our domain
Incomplete Data

- In reality, this assumption might not be true.

\[
D \quad \text{Training instance}
\]

\[
\begin{align*}
X_1 & : 3, 1, 2, 1, 1, 1, 0, 2, 9, 8, \ldots \\
X_2 & : 1, ?, 1, 0, 7, 2, 3, 6, 5, \ldots \\
X_3 & : 0, 0, 1, 2, 1, 0, 8, 2, 2, 3, \ldots \\
X_4 & : 1, 2, 5, 2, 9, 0, 1, 3, 4, 5, \ldots
\end{align*}
\]

- Missing values, Hidden variables

- Challenges
  - Foundational – is the learning task well defined?
  - Computational – how can we learn with missing data?

Treating Missing Data

- How should we treat missing data?
  - Based on data missing mechanism
  - Case I: A coin is tossed on a table, occasionally it drops and measurements are not taken (random missing)
    - Sample sequence: H, T, ?, ?, T, ?, ?, H
    - Treat missing data by ignoring it
  - Case II: A coin is tossed, but only heads are reported (deliberate missing values)
    - Sample sequence: H, ?, ?, ?, H, ?, H
    - Treat missing data by filling it with Tails

We need to consider the data missing mechanism
Modeling Data Missing Mechanism

- Let’s try to model the data missing mechanism
- $X = \{X_1, \ldots, X_n\}$ are random variables
- $O_X = \{O_{X_1}, \ldots, O_{X_n}\}$ are observability variables
  - Always observed
- $Y = \{Y_1, \ldots, Y_n\}$ new random variables
  - Val$(Y_i) = \text{Val}(X_i) \cup \{?\}$
  - $Y_i$ is a deterministic function of $X_i$ and $O_{X_1}$:
    $$Y_i = \begin{cases} X_i & O_X = o_i^X \\ ? & O_X = o_i^? \end{cases}$$

Modeling Missing Data Mechanism

![Diagram of missing data mechanism]

**Case I**
(random missing values)

**Case II**
(deliberate missing values)

$$P(Y = H) = \theta \psi$$
$$P(Y = T) = (1 - \theta) \psi$$
$$P(Y = ?) = (1 - \psi)$$

MLE

$$\hat{\theta} = \frac{M_H}{M_H + M_T + M}$$
$$\hat{\psi} = \frac{M_H}{M_H + M_T + M}$$
Modeling Missing Data Mechanism

Case I
(random missing values)

Case II
(deliberate missing values)

$\psi \theta X OX$

MLE ?

$\hat{\theta} = \psi \theta = ?$

$L(D : \theta, \psi) = \theta_{\theta M} \cdot (1 - \theta)_{\theta M} \cdot \psi_{O_{\psi} \theta_{\theta M}} \cdot \psi_{O_{\psi} \theta_{\theta M}} \cdot (\theta - \psi_{O_{\psi} \theta_{\theta M}} + (1 - \theta)(1 - \psi_{O_{\psi} \theta_{\theta M}}))^{M_x}$

Decoupling of Observation Mechanism

- When can we ignore the missing data mechanism and focus only on the likelihood?
  - Missing Completely at Random (MCAR)
    - For every $X_i$, $\text{Ind}(X_i; O_{\psi})$, a very strong assumption
    - Sufficient but not necessary for the decomposition of the likelihood
  - Missing at Random (MAR) is sufficient
    - The probability that the value of $X_i$ is missing is independent of its actual value, given other observed values $O_{\psi}$

- In both cases, the likelihood decomposes
  - When there are missing values in $D$, try to model such that MAR holds.
Incomplete Data

- In reality, this assumption might not be true.

\[ D \]

\[
\begin{align*}
X_1 & = 3 \text{?}\ 1\ 1\ 1\ ?\ 0\ 2\ 9\ 8 \ldots \\
X_2 & = 1\ ?\ 1\ 1\ 0\ 0\ 7\ 2\ 3\ 6\ 5 \ldots \\
X_3 & = 0\ 0\ 1\ ?\ 1\ 0\ 8\ 2\ ?\ 2\ 3 \ldots \\
X_4 & = 1\ 2\ 5\ -2\ ?\ 0\ 1\ 3\ 4\ 5 \ldots 
\end{align*}
\]

- Missing values, **Hidden variables**

- Challenges
  - Foundational — is the learning task well defined?
  - Computational — how can we learn with missing data?

Hidden (Latent) Variables

- Attempt to learn a model with hidden variables
  - In this case, MCAR always holds (variable is always missing)

\[ \Theta_{\text{MCAR}} \]

\[ \Theta_{\text{MCAR}} = F \]

- Why should we care about unobserved variables?
Hidden (Latent) Variables

- Hidden variables also appear in clustering

- Naïve Bayes model:
  - Class variable is hidden
  - Observed attributes are independent given the class

![Diagram showing Hidden (Latent) Variables and Naïve Bayes model]

### Data Table

<table>
<thead>
<tr>
<th>X_1</th>
<th>X_2</th>
<th>...</th>
<th>X_N-1</th>
<th>X_N</th>
</tr>
</thead>
<tbody>
<tr>
<td>3</td>
<td>1</td>
<td></td>
<td>1</td>
<td>2</td>
</tr>
<tr>
<td>0</td>
<td>1</td>
<td>...</td>
<td>38</td>
<td>28</td>
</tr>
<tr>
<td>0</td>
<td>0</td>
<td></td>
<td>-1</td>
<td>0</td>
</tr>
<tr>
<td>1</td>
<td>2</td>
<td></td>
<td>3</td>
<td>4</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>H</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
</tr>
<tr>
<td>2</td>
</tr>
<tr>
<td>1</td>
</tr>
<tr>
<td>3</td>
</tr>
<tr>
<td>1</td>
</tr>
<tr>
<td>2</td>
</tr>
<tr>
<td>2</td>
</tr>
<tr>
<td>1</td>
</tr>
</tbody>
</table>

### Question

How do missing data affect the likelihood function?
Likelihood for Complete Data

Input Data:

<table>
<thead>
<tr>
<th>x</th>
<th>y</th>
</tr>
</thead>
<tbody>
<tr>
<td>x0</td>
<td>y0</td>
</tr>
<tr>
<td>x1</td>
<td>y1</td>
</tr>
</tbody>
</table>

Likelihood:

\[ L(D; \theta) = \prod P(x[1], y[1]) \cdot P(x[2], y[2]) \cdot P(x[3], y[3]) \]

\[ = P(x^0, y^0) \cdot P(x^0, y^0) \cdot P(x^1, y^1) \]

\[ = \prod \left( \theta_{y0|x0} \cdot \theta_{y1|x0} \cdot \theta_{y0|x1} \cdot \theta_{y1|x1} \right) \]

- Likelihood decomposes by variables
- Likelihood decomposes within CPDs
- Likelihood function is log-concave \( \rightarrow \) unique global maximum that has a simple analytic closed form.

Likelihood for Incomplete Data

Input Data:

<table>
<thead>
<tr>
<th>x</th>
<th>y</th>
</tr>
</thead>
<tbody>
<tr>
<td>?</td>
<td>y0</td>
</tr>
<tr>
<td>x</td>
<td>?</td>
</tr>
</tbody>
</table>

Likelihood:

\[ L(D; \theta) = \prod P(y^0) \cdot \left\{ \sum P(x, y^0) \right\} \]

\[ = \prod \left( \theta_{x0 \cdot x0 \cdot y0} \cdot \theta_{x1 \cdot x0 \cdot y0} \cdot \theta_{x0 \cdot x1 \cdot y0} \right) \cdot \left( \theta_{x0 \cdot x0 \cdot y1} + \theta_{x1 \cdot x0 \cdot y1} + \theta_{x0 \cdot x1 \cdot y1} \right) \]

\[ = \prod \left( \theta_{x0 \cdot x0} \cdot \theta_{x0 \cdot x1} + \theta_{x1 \cdot x0} \cdot \theta_{x1 \cdot x1} \right) \cdot \theta_{y0|x0} \cdot \theta_{y1|x0} \cdot \theta_{y0|x1} \cdot \theta_{y1|x1} \]

- Likelihood does not decompose by variables
- Likelihood does not decompose within CPDs
- Computing likelihood per instance requires inference!
Likelihood with Missing Data

- Multimodal likelihood function with incomplete data
  - Likelihood function is not log-concave → local maxima cannot be obtained by a simple analytic closed form

Gradient Ascent:
- Follow gradient of likelihood w.r.t. to parameters
- Add line search and conjugate gradient methods to get fast convergence

MLE from Incomplete Data

- Take steps proportional to the positive of the gradient.
MLE from Incomplete Data

- Nonlinear optimization problem

Expectation Maximization (EM):
- Use “current point” to construct alternative function (which is “nice”)
- Guaranty: maximum of new function has better score than current point

Gradient Ascent and EM
- Find local maxima
- Require multiple restarts to find approx. to the global maximum
- Require computations in each iteration
Gradient Ascent

- Theorem:

\[
\frac{\partial \log P(D) }{ \partial \theta_{x_i,p_{a_i}}} = \frac{1}{\theta_{x_i,p_{a_i}}} \sum P(x_i, p_{a_i}, o[m], \Theta) \]

- Proof:

\[
\frac{\partial \log P(D|\Theta)}{ \partial \theta_{x_i,p_{a_i}}} = \sum_m \frac{\partial \log P(o[m]|\Theta)}{ \partial \theta_{x_i,p_{a_i}}} = \sum_m \frac{1}{P(o[m]|\Theta)} \frac{\partial P(o[m]|\Theta)}{ \partial \theta_{x_i,p_{a_i}}} = \sum_m \frac{\partial \log P(o[m]|\Theta)}{ \partial \theta_{x_i,p_{a_i}}}
\]

How do we compute?
Gradient Ascent

\[ \frac{\partial \log P(D | \Theta)}{\partial \theta_{x_i, pa_i}} = \sum_m \frac{1}{P(o[m] | \Theta)} \frac{\partial P(o[m] | \Theta)}{\partial \theta_{x_i, pa_i}} \]

\[ = \sum_m \frac{1}{P(o[m] | \Theta)} P(x_i, pa_i, o[m] | \Theta) \frac{\partial \log \theta_{x_i, pa_i}}{\partial \theta_{x_i, pa_i}} \]

\[ = \sum_m P(x_i, pa_i | o[m], \Theta) \frac{\partial \log \theta_{x_i, pa_i}}{\partial \theta_{x_i, pa_i}} \]

- Requires computation: \( P(x_i, pa_i | o[m], \Theta) \) for all \( i, m \)

- Can be done with clique-tree algorithm, since \( X_i, Pa_i \) are in the same clique

Gradient Ascent Summary

Pros

- Flexible, can be extended to non table CPDs

Cons

- Need to project gradient onto space of legal parameters
- For reasonable convergence, need to combine with advanced methods (conjugate gradient, line search)
Expectation Maximization (EM)

- Tailored algorithm for optimizing likelihood functions.

**Intuition**
- Parameter estimation is easy given complete data.
- Computing probability of missing data is “easy” (=inference) given parameters.

**Strategy**
- Pick a starting point for parameters.
- “Complete” the data using current parameters.
- Estimate parameters relative to data completion.
- Iterate.
- Procedure guaranteed to improve at each iteration.

**Procedure**

1. Initialize parameters to $\theta^0$.
2. Iterate E-step and M-step.
3. In the $t$-th iteration, we do:
   - **Expectation (E-step):**
     - Let $o[m]$ be the observed data in the $m$-th training instance.
     - For each $m$ and each family $X_i, Pa_i$, compute $P(X_i, Pa_i | o[m], \theta^{(0)})$.
     - Compute the expected sufficient statistics for each values $x$, $u$ on $X_i, Pa_i$, respectively.
     \[
     \sum_{m=1}^{M} P(X_i = x, Pa_i = u | o[m], \theta^{(0)})
     \]
   - **Maximization (M-step):**
     - Treat the expected sufficient statistics as observed and set the parameters to the MLE with respect to the ESS.
     \[
     \theta^{(t+1)}_{X_i=x, Pa_i=u} = \frac{\sum_{m=1}^{M} P(X_i = x, Pa_i = u | o[m], \theta^{(0)})}{\sum_{m=1}^{M} P(Pa_i = u | o[m], \theta^{(0)})}
     \]
Expectation Maximization (EM)

**Initial network**

\[ X \rightarrow Y \]

**Training data**

<table>
<thead>
<tr>
<th>X</th>
<th>Y</th>
</tr>
</thead>
<tbody>
<tr>
<td>?</td>
<td>y_0</td>
</tr>
<tr>
<td>x_0</td>
<td>y_1</td>
</tr>
<tr>
<td>?</td>
<td>y_n</td>
</tr>
</tbody>
</table>

**E-Step** (inference)

- Expected counts
  - \( N(X) \)
  - \( N(X,Y) \)

**M-Step** (reparameterize)

**Iterate**

**Updated network**

\[ X \rightarrow Y \]

---

**Expectation Maximization (EM)**

- **Formal Guarantees:**
  - \( \mathbb{E}(\mathcal{D};\Theta(t+1)) \geq \mathbb{E}(\mathcal{D};\Theta(t)) \)
    - Each iteration improves the likelihood
  - If \( \Theta(t+1) = \Theta(t) \), then \( \Theta(t) \) is a stationary point of \( \mathbb{E}(\mathcal{D};\Theta) \)
    - Usually, this means a local maximum

- **Main cost:**
  - Computations of expected counts in E-Step
  - Requires inference for each instance in training set
    - Usually the same as in gradient ascent!

- **Reading material on EM**
  - Please read Andrew Ng’s lecture note
EM – Practical Considerations

- **Initial parameters**
  - Highly sensitive to starting parameters
  - Choose randomly
  - Choose by guessing from another source

- **Stopping criteria**
  - Small change in data likelihood
  - Small change in parameters

- **Avoiding bad local maxima**
  - Multiple restarts
  - Early pruning of unpromising starting points

Acknowledgement

- These lecture notes were generated based on the slides from Prof Eran Segal.