STATISTICAL METHODS IN AI/ML

Vibhav Gogate
University of Texas, Dallas

LEARNING: Lecture 2

THE UNIVERSITY OF TEXAS AT DALLAS
Erik Jonsson School of Engineering and Computer Science

What we will cover

- Fully observed
- Partially observed

- Parameter Learning
- Structure Learning

- Maximum likelihood estimation approach
- Bayesian approach

- Bayesian network
- Markov network

We will learn 16 classes of learning algorithms

PART 1

Fully Observed Data
Parameter Learning
MLE approach
Bayesian network

Maximum Likelihood Estimation Principle

- **Idea:** Given candidate probabilistic models $\{M_1, \ldots, M_t\}$, select a candidate M_i such that the data is *most probable* w.r.t. M_i.
- Given dataset $\mathcal{D} = \{X^{(1)}, \ldots, X^{(m)}\}$ and a probabilistic model Pr defined using a set of parameters Θ, find a setting of the parameters in Θ such that the likelihood of generating the data, namely $L(\mathcal{D}, \Theta) = \prod_{i=1}^{m} \Pr(X^{(i)}; \Theta)$ is maximized.
  Mathetmatically

$$\Theta^* = \arg\max_{\Theta} \prod_{i=1}^{m} \Pr(X^{(i)}; \Theta)$$

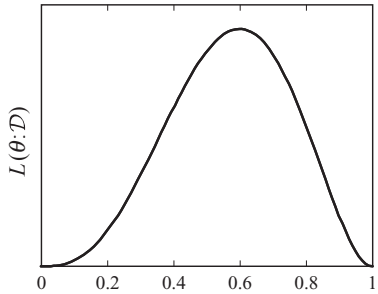
Example: A Biased coin

- Variable x having two outcomes: H=*head* and T=*tail*
- Data set: Tosses of the coin. Example: 100 tosses with 60 heads and 40 tails.
- Let $\Pr(x = h) = \theta$. Candidates are models with different values of θ.
- Value of MLE, θ^* if 60 out of 100 tosses are heads.

MLE scoring for the coin example

Distribution: $\Pr(x = h) = \theta$ and $\Pr(x = t) = 1 - \theta$

- Evaluation metric: How well we can predict the data?
- Example data: H, H, T, H, T
- Likelihood of data $= \prod_i \Pr(x_i) = \theta.\theta.(1-\theta).\theta.(1-\theta)$



MLE scoring for the coin example: Analytical derivation

Distribution: $\Pr(x = h) = \theta$ and $\Pr(x = t) = 1 - \theta$.

- Log-Likelihood function

$$LogL(\theta) = \log(\theta^{\#heads}.(1-\theta)^{\#tails})$$

$$= \#heads.\log(\theta) + \#tails.\log(1-\theta)$$

- MLE Aim: Find θ^* such that $LogL(\theta^*)$ is maximum.
- Differentiate the likelihood function with respect to θ and set the derivative to zero. We get:

$$\theta^* = \frac{\#heads}{\#heads + \#tails}$$

Extending the MLE to Bayesian networks

Demonstrate the approach using an Example

Given: $X_1 \rightarrow X_2$ with parameters $\Theta = \{\theta_1, \theta_{2|0}, \theta_{2|1}\}$ and a Dataset $\mathcal{D}$

To do: Find Θ such that the following likelihood function is maximized:

$$\begin{aligned} L(\mathcal{D}, \Theta) &= \left(\theta_1^{\#_{\mathcal{D}}(X_1=0)}(1-\theta_1)^{\#_{\mathcal{D}}(X_1=1)}\right) \times \\ &\left(\theta_{2|0}^{\#_{\mathcal{D}}(X_2=0,X_1=0)}(1-\theta_{2|0})^{\#_{\mathcal{D}}(X_2=1,X_1=0)}\right) \times \left(\theta_{2|1}^{\#_{\mathcal{D}}(X_2=0,X_1=1)}(1-\theta_{2|1})^{\#_{\mathcal{D}}(X_2=1,X_1=1)}\right) \end{aligned}$$

(Log) Likelihood is decomposable. Each parameter is involved in just two terms.

$$\begin{aligned} LL(\mathcal{D}, \Theta) &= \#_{\mathcal{D}}(X_1 = 0)\log(\theta_1) + \#_{\mathcal{D}}(X_1 = 1)\log(1-\theta_1) + \\ &\#_{\mathcal{D}}(X_2 = 0, X_1 = 0)\log(\theta_{2|0}) + \#_{\mathcal{D}}(X_2 = 1, X_1 = 0)\log(1-\theta_{2|0}) + \\ &\#_{\mathcal{D}}(X_2 = 0, X_1 = 1)\log(\theta_{2|1}) + \#_{\mathcal{D}}(X_2 = 1, X_1 = 1)\log(1-\theta_{2|1}) \end{aligned}$$

Extending the MLE to Bayesian networks

(Log) Likelihood is decomposable. Each parameter is involved in just two terms.

$$\begin{aligned} LL(\mathcal{D},\Theta) &= \#_{\mathcal{D}}(X_1=0)\log(\theta_1)+\#_{\mathcal{D}}(X_1=1)\log(1-\theta_1)+ \\ &\#_{\mathcal{D}}(X_2=0,X_1=0)\log(\theta_{2|0})+\#_{\mathcal{D}}(X_2=1,X_1=0)\log(1-\theta_{2|0})+ \\ &\#_{\mathcal{D}}(X_2=0,X_1=1)\log(\theta_{2|1})+\#_{\mathcal{D}}(X_2=1,X_1=1)\log(1-\theta_{2|1}) \end{aligned}$$

Taking derivatives and equating them to zero, we get:

$$\theta_1 = \frac{\#_{\mathcal{D}}(X_1=0)}{\#_{\mathcal{D}}(X_1=0)+\#_{\mathcal{D}}(X_1=1)}$$

$$\theta_{2|0} = \frac{\#_{\mathcal{D}}(X_2=0,X_1=0)}{\#_{\mathcal{D}}(X_2=0,X_1=0)+\#_{\mathcal{D}}(X_2=1,X_1=0)} = \frac{\#_{\mathcal{D}}(X_2=0,X_1=0)}{\#_{\mathcal{D}}(X_1=0)}$$

$$\theta_{2|1} = \frac{\#_{\mathcal{D}}(X_2=0,X_1=1)}{\#_{\mathcal{D}}(X_2=0,X_1=1)+\#_{\mathcal{D}}(X_2=1,X_1=1)} = \frac{\#_{\mathcal{D}}(X_2=0,X_1=1)}{\#_{\mathcal{D}}(X_1=1)}$$

Extending the MLE principle to a Bayesian network

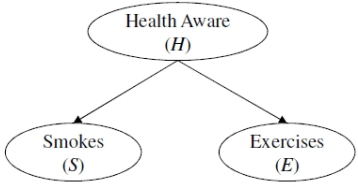
Given a Bayesian network:

- Given (fully observed) data $\mathcal{D}$, MLE solution is:

$$\theta^*_{x_i|pa(x_i)} = \frac{\#_{\mathcal{D}}(x_i, pa(x_i))}{\#_{\mathcal{D}}(pa(x_i))}$$

  where $\#_{\mathcal{D}}(x_i, pa(x_i))$ is the number of times the tuple $(x_i, pa(x_i))$ appears in $\mathcal{D}$. $\#_{\mathcal{D}}(pa(x_i))$ is the number of times the tuple $pa(x_i)$ appears in $\mathcal{D}$.
- $\#_{\mathcal{D}}(x_i, pa(x_i))$ is called the sufficient statistic.
- Any function of the data is called a statistic. A sufficient statistic is a statistic that contains all of the information in the data set that is needed for a particular estimation task.

MLE Learning example: Bayesian network



(a) network structure

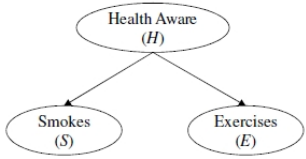
Case	H	S	E
1	T	F	T
2	T	F	T
3	F	T	F
4	F	F	T
5	T	F	F
6	T	F	T
7	F	F	F
8	T	F	T
9	T	F	T
10	F	F	T
11	T	F	T
12	T	T	T
13	T	F	T
14	T	T	T
15	T	F	T
16	T	F	T

(b) complete data

H	S	E	$\Pr_{\mathcal{D}}(.)$
T	T	T	$2/16$
T	T	F	$0/16$
T	F	T	$9/16$
T	F	F	$1/16$
F	T	T	$0/16$
F	T	F	$1/16$
F	F	T	$2/16$
F	F	F	$1/16$

(c) empirical distribution

MLE Learning example: Bayesian network



We have the following parameter estimates:

H	θ^{ml}_H
h	3/4
$\bar{h}$	1/4

H	S	$\theta^{ml}_{S\|H}$
h	s	1/6
h	$\bar{s}$	5/6
$\bar{h}$	s	1/4
$\bar{h}$	$\bar{s}$	3/4

H	E	$\theta^{ml}_{E\|H}$
h	e	11/12
h	$\bar{e}$	1/12
$\bar{h}$	e	1/2
$\bar{h}$	$\bar{e}$	1/2

MLE Learning: Bayesian network (fully observable case)

Impact of data set size

- ML estimate will have different values depending upon the size of the data set
- The variance of the estimate will decrease as the data set increases in size.
- Estimating probabilities that are quite small (or quite large) is hard because a large number of samples are required to reach a reliable estimate.

Other Properties

- Likelihood Function is unimodal. Namely, the ML estimate is unique.
- ML estimate can be computed in closed form. Computational Complexity is linear in the number of variables, parameters and the number of examples.

PART 2

Partially Observed Data
Parameter Learning
MLE approach
Bayesian network

- Examples: missing data, hidden variables, some variables are just not observable
- Gradient Ascent
- Expectation maximization (The EM algorithm)

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:

$$L(\theta, \mathcal{X}) = \prod_{j=1}^{m} \sum_{\mathbf{y} \notin \mathbf{x}^{(j)}} \Pr_{\theta}(\mathbf{x}^{(j)}, \mathbf{y})$$

Compare with Likelihood function for FOD:

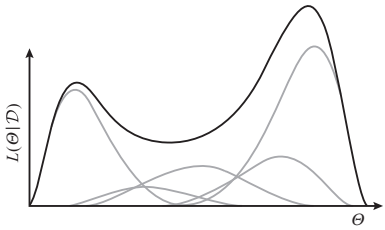
$$L(\theta, \mathcal{X}) = \prod_{j=1}^{m} \Pr_{\theta}(\mathbf{x}^{(j)})$$

Likelihood function for POD:

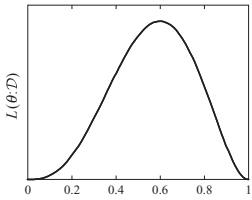
- is not unimodal.
- is not decomposable because of the sum over $\mathbf{Y}$.

As a result, there is no closed form solution.

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.



FOD case:
Unimodal distribution

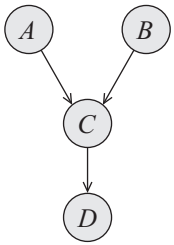
- The optimization problem, a.k.a. maximizing our objective, the likelihood of the data is hard. We need an iterative approach.

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 (E-step)



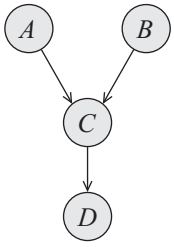
$\theta_a = .3$
$\theta_b = .9$
$\theta_{c|\bar{a},\bar{b}} = .83$
$\theta_{c|\bar{a},b} = .09$
$\theta_{c|a,\bar{b}} = .6$
$\theta_{c|a,b} = .2$
$\theta_{d|\bar{c}} = .1$
$\theta_{d|c} = .8$

E-step details:

- **STEP 1:** Complete each partially observed example in all possible ways
- **STEP 2:** Compute how likely each *completed example* is according to the current parameters. 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 $= .0164$
  - $(a, \bar{b}, \bar{c}, \bar{d})$, weight $= .0492$

The Expectation Maximization Algorithm: Example (E-step)



$\theta_a = .3$
$\theta_b = .9$
$\theta_{c|\bar{a},\bar{b}} = .83$
$\theta_{c|\bar{a},b} = .09$
$\theta_{c|a,\bar{b}} = .6$
$\theta_{c|a,b} = .2$
$\theta_{d|\bar{c}} = .1$
$\theta_{d|c} = .8$

Let us say we have just two examples in our dataset: $(a, ?, ?, \bar{d})$ and $(?, b, c, d)$.

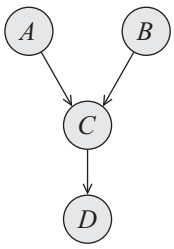
$(a, ?, ?, \bar{d})$ corresponds to four weighted examples

- $(a, b, c, \bar{d})$, weight $= .0492$
- $(a, b, \bar{c}, \bar{d})$, weight $= .8852$
- $(a, \bar{b}, c, \bar{d})$, weight $= .0164$
- $(a, \bar{b}, \bar{c}, \bar{d})$, weight $= .0492$

$(?, b, c, d)$ corresponds to two weighted examples

- (a, b, c, d), weight $= 0.4878$
- $(\bar{a}, b, c, d)$, weight $= 0.5122$

The Expectation Maximization Algorithm: Example (M-step)



$\theta_a = .3$
$\theta_b = .9$
$\theta_{c|\bar{a},\bar{b}} = .83$
$\theta_{c|\bar{a},b} = .09$
$\theta_{c|a,\bar{b}} = .6$
$\theta_{c|a,b} = .2$
$\theta_{d|\bar{c}} = .1$
$\theta_{d|c} = .8$

M-step: Update the parameters based on the bigger and weighted dataset.

- $(a, b, c, \bar{d})$, weight $= .0492$
- $(a, b, \bar{c}, \bar{d})$, weight $= .8852$
- $(a, \bar{b}, c, \bar{d})$, weight $= .0164$
- $(a, \bar{b}, \bar{c}, \bar{d})$, weight $= .0492$
- (a, b, c, d), weight $= 0.4878$
- $(\bar{a}, b, c, d)$, weight $= 0.5122$

Updated Parameters:

- $\theta_a = \frac{0.0492+0.8852+0.0164+0.0492+0.4878}{0.0492+0.8852+0.0164+0.0492+0.4878+0.5122} = \frac{1.4878}{2} = 0.7439$

The EM Algorithm: Improving Time and Space Complexity

- Time complexity of the algorithm as presented is impractical when large number of variables are missing. Given N incomplete examples, each having p missing values, the **Time complexity** is $\Omega(N \times 2^p)$.
- Instead, we could use the following expected sufficient statistics to compute the parameter $P(x|\mathbf{u}) = \theta_{x|\mathbf{u}}$ given a dataset $\{\mathbf{o}^{(1)}, \ldots, \mathbf{o}^{(N)}\}$:

$$\theta_{x|\mathbf{u}} = \frac{\text{sum-weight}(x, \mathbf{u})}{\text{sum-weight}(\mathbf{u})} = \frac{\sum_{j=1}^{N} \Pr(x, \mathbf{u}|\mathbf{o}^{(j)})}{\sum_{j=1}^{m} \Pr(\mathbf{u}|\mathbf{o}^{(j)})}$$

- Recall that in the unweighted case, the counts over data points were the sufficient statistics. Here, the sum-weights are the sufficient statistics.
- In other words, for each example indexed by j, we only need to compute $\Pr(x, \mathbf{u}|\mathbf{o}^{(j)})$ for each parameter $\theta_{x|\mathbf{u}}$ using an inference algorithm. Complexity of the new scheme $O(N \times \text{Inf})$ where Inf is the complexity of the inference scheme.

The EM Algorithm

Procedure Compute-ESS (
  $\mathcal{G}$, // Bayesian network structure over $X_1, \ldots, X_n$
  $\boldsymbol{\theta}$, // 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 $x_i, \boldsymbol{u}_i \in Val(X_i, \mathrm{Pa}^{\mathcal{G}}_{X_i})$
4 $\bar{M}[x_i, \boldsymbol{u}_i] \leftarrow 0$
5 // Collect probabilities from all instances
6 **for** each $m = 1 \ldots M$
7 Run inference on $\langle \mathcal{G}, \boldsymbol{\theta} \rangle$ using evidence $\boldsymbol{o}[m]$
8 **for** each $i = 1, \ldots, n$
9 **for** each $x_i, \boldsymbol{u}_i \in Val(X_i, \mathrm{Pa}^{\mathcal{G}}_{X_i})$
10 $\bar{M}[x_i, \boldsymbol{u}_i] \leftarrow \bar{M}[x_i, \boldsymbol{u}_i] + P(x_i, \boldsymbol{u}_i \mid \boldsymbol{o}[m])$
11 **return** $\{\bar{M}[x_i, \boldsymbol{u}_i] : \forall i = 1, \ldots, n, \forall x_i, \boldsymbol{u}_i \in Val(X_i, \mathrm{Pa}^{\mathcal{G}}_{X_i})\}$

The EM Algorithm

Procedure Expectation-Maximization (
  $\mathcal{G}$, // Bayesian network structure over $X_1, \ldots, X_n$
  $\boldsymbol{\theta}^0$, // Initial set of parameters for $\mathcal{G}$
  $\mathcal{D}$ // Partially observed data set
)

1 **for** each $t = 0, 1 \ldots,$ until convergence
2 // E-step
3 $\{\bar{M}_t[x_i, \boldsymbol{u}_i]\} \leftarrow$ Compute-ESS$(\mathcal{G}, \boldsymbol{\theta}^t, \mathcal{D})$
4 // M-step
5 **for** each $i = 1, \ldots, n$
6 **for** each $x_i, \boldsymbol{u}_i \in Val(X_i, \mathrm{Pa}_{X_i}^{\mathcal{G}})$
7 $\theta^{t+1}_{x_i|\boldsymbol{u}_i} \leftarrow \frac{\bar{M}_t[x_i, \boldsymbol{u}_i]}{\bar{M}_t[\boldsymbol{u}_i]}$
8 **return** $\boldsymbol{\theta}^t$

EM: Properties and Summary

- Each iteration of EM (the E plus M step) can only increase the likelihood and never decrease it. Therefore, EM will always converge to a local maxima.
- 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. The sufficient statistics are the posterior probabilities of all the parameters.
- Since Inference will be exponential in general, one often has to use approximate inference algorithms such as IJGP or sampling algorithms in practice. Convergence guarantees do not exist when approximate algorithms are used.

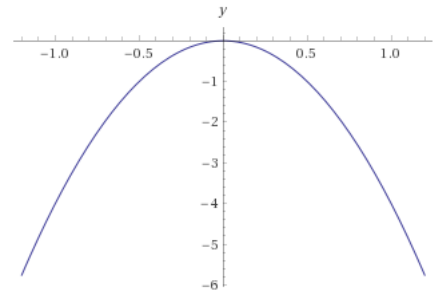
Gradient Ascent: Algorithm

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

Algorithm A.10 Simple gradient ascent algorithm

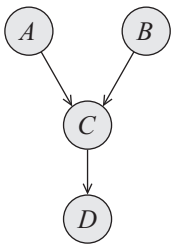
Procedure Gradient-Ascent (
  $\boldsymbol{\theta}^1$, // Initial starting point
  f_{obj}, // Function to be optimized
  δ // Convergence threshold
)

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



- Remember: Derivative is the slope of the line that is tangent to the function
- Question: What if the learning rate is small? (Slow convergence) or large? (Fail to converge; even diverge)

Gradient Ascent: Example



$\theta_a = .3$
$\theta_b = .9$
$\theta_{c|\bar{a},\bar{b}} = .83$
$\theta_{c|\bar{a},b} = .09$
$\theta_{c|a,\bar{b}} = .6$
$\theta_{c|a,b} = .2$
$\theta_{d|\bar{c}} = .1$
$\theta_{d|c} = .8$

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: Gradient of Log Likelihood

- How to compute the gradient?
  For a data instance:
  $$\frac{\partial \Pr(\mathbf{o})}{\partial \Pr(x|\mathbf{u})} = \frac{1}{\Pr(x|\mathbf{u})} \Pr(x, \mathbf{u}, \mathbf{o})$$
  For a data-set:
  $$\frac{\partial LL(\theta, \mathcal{X})}{\partial \Pr(x|\mathbf{u})} = \frac{1}{\Pr(x|\mathbf{u})} \sum_{j=1}^{m} \Pr_{\theta}(x, \mathbf{u}|\mathbf{x}^{(j)})$$

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 $X_1, \ldots, X_n$
  $\boldsymbol{\theta}$, // 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** $x_i, \boldsymbol{u}_i \in Val(X_i, \mathrm{Pa}_{X_i}^{\mathcal{G}})$
4 $\bar{M}[x_i, \boldsymbol{u}_i] \leftarrow 0$
5 // Collect probabilities from all instances
6 **for each** $m = 1 \ldots M$
7 Run clique tree calibration on $\langle \mathcal{G}, \boldsymbol{\theta} \rangle$ using evidence $\boldsymbol{o}[m]$
8 **for each** $i = 1, \ldots, n$
9 **for each** $x_i, \boldsymbol{u}_i \in Val(X_i, \mathrm{Pa}_{X_i}^{\mathcal{G}})$
10 $\bar{M}[x_i, \boldsymbol{u}_i] \leftarrow \bar{M}[x_i, \boldsymbol{u}_i] + P(x_i, \boldsymbol{u}_i \mid \boldsymbol{o}[m])$
11 // Compute components of the gradient vector
12 **for each** $i = 1, \ldots, n$
13 **for each** $x_i, \boldsymbol{u}_i \in Val(X_i, \mathrm{Pa}_{X_i}^{\mathcal{G}})$
14 $\delta_{x_i|\boldsymbol{u}_i} \leftarrow \frac{1}{\theta_{x_i|\boldsymbol{u}_i}} \bar{M}[x_i, \boldsymbol{u}_i]$
15 **return** $\{\delta_{x_i|\boldsymbol{u}_i} : \forall i = 1, \ldots, n, \forall (x_i, \boldsymbol{u}_i) \in Val(X_i, \mathrm{Pa}_{X_i}^{\mathcal{G}})\}$