732A96 Advanced Machine Learning

Graphical Models and Hidden Markov Models

Jose M. Peña
IDA, Linköping University, Sweden

Lecture 3: Parameter Learning

Contents

- Parameter Learning for BNs
  - Maximum Likelihood
  - Maximum A Posteriori
  - Expectation Maximization Algorithm
- Parameter Learning for MNs
  - Iterative Proportional Fitting Procedure

Literature

- Main sources
  - Bishop, C. M. *Pattern Recognition and Machine Learning*. Springer, 2006.
  - Koller, D. and Friedman, N. *Probabilistic Graphical Models: Principles and Techniques*. MIT Press, 2009.
- Other sources
  - Koski, T. J. T. and Noble, J. M. A Review of Bayesian Networks and Structure Learning. *Mathematica Applicanda* 40, 51-103, 2012.
  - Murphy, K. P. *Machine Learning: A Probabilistic Perspective*. MIT Press, 2012.
- R resources
  - Højsgaard, S., Edwards, D. and Lauritzen, S. *Graphical Models with R*. Springer, 2012.
  - Nagarajan, R., Scutari, M. and Lébre, S. *Bayesian Networks in R*. Springer, 2013.
  - Scutari, S. Learning Bayesian Networks with the bnlearn R Package. *Journal of Statistical Software* 35, 2010.

Parameter Learning for BNs: Maximum Likelihood

- Given a sample $d_{1:N}$, the log likelihood function is

$$\log p(d_{1:N}|\theta_G, G) = \log[\prod_l p(d_l|\theta_G, G)] = \log[\prod_l \prod_i p(d_l[X_i]|d_l[pa_G(X_i)], \theta_G)]$$

$$= \log\,[\prod_l \prod_i \theta_{X_i=d_l[X_i]|pa_G(X_i)=d_l[pa_G(X_i)]}] = \log\,[\prod_i \prod_j \prod_k \theta^{N_{ijk}}_{X_i=k|pa_G(X_i)=j}]$$

$$= \sum_i \sum_j \sum_k N_{ijk} \log \theta_{X_i=k|pa_G(X_i)=j}$$

- To maximize the log likelihood function subject to the constraint $\sum_k \theta_{X_i=k|pa_G(X_i)=j} = 1$ for all i and j, we maximize

$$\sum_i \sum_j \sum_k N_{ijk} \log \theta_{X_i=k|pa_G(X_i)=j} + \sum_i \sum_j \lambda_{ij}(\sum_k \theta_{X_i=k|pa_G(X_i)=j} - 1)$$

  where λ_{ij} are called Lagrange multipliers.[1]
- Setting to zero the derivative with respect to $\theta_{X_i=k|pa_G(X_i)=j}$ gives

$$\theta_{X_i=k|pa_G(X_i)=j} = -N_{ijk}/\lambda_{ij}$$

- Replacing this into the constraint gives $\lambda_{ij} = -N_{ij}$ and, thus, $\theta^{ML}_{X_i=k|pa_G(X_i)=j} = N_{ijk}/N_{ij}$.

[1] Any stationary point of the Lagrangian function is a stationary point of the original function subject to the constraints. Moreover, the log likelihood function is concave.

Parameter Learning for BNs: Maximum A Posteriori

- Alternatively, we can choose the parameter values θ_G with maximum posterior probability

  $$p(\theta_G|d_{1:N}, G) = p(d_{1:N}|\theta_G, G)p(\theta_G|G)/p(d_{1:N}|G) \propto p(d_{1:N}|\theta_G, G)p(\theta_G|G)$$

  where $p(d_{1:N}|\theta_G, G)$ is the likelihood function, $p(\theta_G|G)$ is a prior probability distribution, and $p(d_{1:N}|G)$ is a normalization constant.
- **Assuming** that $p(\theta_G|G) = \prod_i \prod_j p(\theta_{X_i|pa_G(X_i)=j}|G)$ and $p(\theta_{X_i|pa_G(X_i)=j}|G) \sim Dirichlet(\alpha_{ij1}, \dots, \alpha_{ijk_i})$, we have that

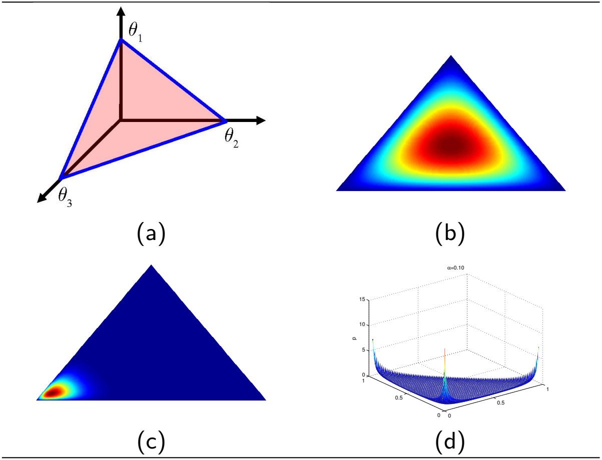
  $$p(\theta_{X_i|pa_G(X_i)=j}|G) \propto \prod_k \theta_{X_i=k|pa_G(X_i)=j}^{\alpha_{ijk}-1}$$

  and thus

  $$p(d_{1:N}|\theta_G, G)p(\theta_G|G)$$
  $$\propto \prod_i \prod_j \prod_k \theta_{X_i=k|pa_G(X_i)=j}^{N_{ijk}} \prod_i \prod_j \prod_k \theta_{X_i=k|pa_G(X_i)=j}^{\alpha_{ijk}-1} = \prod_i \prod_j \prod_k \theta_{X_i=k|pa_G(X_i)=j}^{N_{ijk}+\alpha_{ijk}-1}$$
- The posterior probability distribution is maximized when

  $$\theta_{X_i=k|pa_G(X_i)=j}^{MAP} = (N_{ijk} + \alpha_{ijk} - 1)/(N_{ij} + \alpha_{ij} - k_i)$$

Parameter Learning for BNs: Maximum A Posteriori



(a) The Dirichlet distribution over a 3-valued random variable is defined over the simplex represented by the triangular surface. Points in this surface satisfy $0 \leq \theta_i \leq 1$ and $\sum_i \theta_i = 1$. (b) Dirichlet(2,2,2). (c) Dirichlet(20,2,2). (d) Dirichlet(0.1,0.1,0.1). Source: Murphy (2012).

Parameter Learning for BNs: Expectation Maximization Algorithm

- Let $d_{1:N}$ be an **incomplete sample**, i.e. $d_l[X_i] =?$ for some i and l. Let $o_{1:N}$ denote the observed part of $d_{1:N}$, and $u_{1:N}$ the unobserved part.
- The log likelihood function over $o_{1:N}$ is

$$\log p(o_{1:N}|\theta_G, G) = \log \prod_l \sum_{u_l} p(o_l, u_l|\theta_G, G) = \sum_l \log \sum_{u_l} p(o_l, u_l|\theta_G, G)$$

- To maximize it subject to the constraint $\sum_k \theta_{X_i=k|pa_G(X_i)=j} = 1$ for all i and j, we maximize

$$\sum_l \log \sum_{u_l} p(o_l, u_l|\theta_G, G) + \sum_i \sum_j \lambda_{ij} (\sum_k \theta_{X_i=k|pa_G(X_i)=j} - 1)$$

- Its derivative with respect to $\theta_{X_i=k|pa_G(X_i)=j}$ is

$$\sum_l \frac{\sum_{u_l:c_l[X_i]=k, c_l[pa_G(X_i)]=j} \prod_{i'} \theta_{X_{i'}=c_l[X_{i'}]|pa_G(X_{i'})=c_l[pa_G(X_{i'})]}}{\theta_{X_i=k|pa_G(X_i)=j} \sum_{u_l} p(o_l, u_l|\theta_G, G)} + \lambda_{ij}$$

$$= \sum_l \sum_{u_l:c_l[X_i]=k, c_l[pa_G(X_i)]=j} \frac{p(u_l|o_l, \theta_G, G)}{\theta_{X_i=k|pa_G(X_i)=j}} + \lambda_{ij} = M_{ijk}/\theta_{X_i=k|pa_G(X_i)=j} + \lambda_{ij}$$

  where $c_l = \{o_l, u_l\}$ and $M_{ijk} = \sum_l \sum_{u_l:c_l[X_i]=k, c_l[pa_G(X_i)]=j} p(u_l|o_l, \theta_G, G)$.
- Setting the derivative to zero gives

$$\theta_{X_i=k|pa_G(X_i)=j} = -M_{ijk}/\lambda_{ij}$$

- Replacing this into the constraint gives $\lambda_{ij} = -M_{ij}$ and, thus, $\theta^{ML}_{X_i=k|pa_G(X_i)=j} = M_{ijk}/M_{ij}$. **No closed form solution** but ...

Parameter Learning for BNs: Expectation Maximization Algorithm

EM algorithm	
Set θ_G to some initial values	
Repeat until θ_G does not change	
Compute $p(U_l \mid o_l, \theta_G, G)$ for all l	/* E step */
Compute M_{ijk}	
Set $\theta_G = M_{ijk}/M_{ij}$	/* M step */

- Note that computing $p(U_l \mid o_l, \theta_G, G)$ requires inference.

Parameter Learning for BNs: Expectation Maximization Algorithm

- As shown before, maximizing the log likelihood function over O is inefficient as no closed form solution exists.
- Moreover, it is ineffective due to **multimodality**, i.e. each completion of the data defines a unimodal function but their sum may be multimodal.
- Consider instead maximizing its expectation

$$\mathrm{E}_{U_{1:N}}[\log p(o_{1:N}, U_{1:N}|\theta_G, G)] = \sum_l \sum_{u_l} p(u_l|o_l, \theta_G, G) \log p(o_l, u_l|\theta_G, G)$$

$$= \sum_l \sum_{u_l} p(u_l|o_l, \theta_G, G) \sum_i \log \theta_{X_i=c_l[X_i]|pa_G(X_i)=c_l[pa_G(X_i)]}$$

where $c_l = \{o_l, u_l\}$. Then

$$\mathrm{E}_{U_{1:N}}[\log p(o_{1:N}, U_{1:N}|\theta_G, G)] = \sum_i \sum_j \sum_k M_{ijk} \log \theta_{X_i=k|pa_G(X_i)=j}$$

where $M_{ijk} = \sum_l \sum_{u_l: c_l[X_i]=k, c_l[pa_G(X_i)]=j} p(u_l|o_l, \theta_G, G)$.

- Then, $\theta^{ML}_{X_i=k|pa_G(X_i)=j} = M_{ijk}/M_{ij}$. No closed form solution but it suggests the EM algorithm too.

Parameter Learning for BNs: Expectation Maximization Algorithm

- Another way to motivate the EM algorithm is as follows:

$$\log p(o_{1:N}|\theta_G, G) = L(q, \theta_G) + KL(q||p)$$

  where
  - $L(q, \theta_G) = \sum_{u_{1:N}} q(u_{1:N}) \log \left[p(o_{1:N}, u_{1:N}|\theta_G, G)/q(u_{1:N}) \right]$
  - $KL(q||p) = -\sum_{u_{1:N}} q(u_{1:N}) \log \left[p(u_{1:N}|o_{1:N}, \theta_G, G)/q(u_{1:N}) \right]$
  - $q(U_{1:N})$ is a probability distribution.
- To see it, note that $L(q, \theta_G)$

$$= \sum_{u_{1:N}} q(u_{1:N}) [\log p(u_{1:N}|o_{1:N}, \theta_G, G) + \log p(o_{1:N}|\theta_G, G)] - \sum_{u_{1:N}} q(u_{1:N}) \log q(u_{1:N})$$

$$= \log p(o_{1:N}|\theta_G, G) + \sum_{u_{1:N}} q(u_{1:N}) \log p(u_{1:N}|o_{1:N}, \theta_G, G) - \sum_{u_{1:N}} q(u_{1:N}) \log q(u_{1:N})$$

$$= \log p(o_{1:N}|\theta_G, G) - KL(q||p)$$

- Note that $KL(q||p) \geq 0$ and, thus, $\log p(o_{1:N}|\theta_G, G) \geq L(q, \theta_G)$ for any $q(U_{1:N})$.
- E step: Maximize the lower bound $L(q, \theta_G)$ by setting $q(U_{1:N}) = p(U_{1:N}|o_{1:N}, \theta_G, G)$, since then $KL(q||p) = 0$.
- Note that now $L(q, \theta_G) = \mathrm{E}_{U_{1:N}}[\log p(o_{1:N}, U_{1:N}|\theta_G, G)] + \mathit{constant}$.
- M step: Maximize the lower bound $L(q, \theta_G)$ with respect to θ_G.
- The last step may introduce a non-zero $KL(q||p)$, resulting in an iterative process: The EM algorithm.

Parameter Learning for MNs: Iterative Proportional Fitting Procedure

- Given a complete sample $d_{1:N}$, the log likelihood function is

$$p(d_{1:N}|\theta_G, G) = \sum_{K \in Cl(G)} \sum_{k} N_k \log \varphi(k) - N \log Z$$

  where N_K is the number of instances in $d_{1:N}$ where K takes value k. Then

$$p(d_{1:N}|\theta_G, G)/N = \sum_{K \in Cl(G)} \sum_{k} p_e(k) \log \varphi(k) - \log Z$$

  where $p_e(X)$ is the empirical probability distribution obtained from $d_{1:N}$.
- Let $Q \in Cl(G)$. The derivative with respect to $\varphi(q)$ is

$$\frac{\partial p(d_{1:N}|\theta_G, G)/N}{\partial \varphi(q)} = \frac{p_e(q)}{\varphi(q)} - \frac{1}{Z} \frac{\partial Z}{\partial \varphi(q)}$$

- Let $Y = X \setminus Q$. Then

$$\frac{\partial Z}{\partial \varphi(q)} = \sum_{y} \prod_{K \in Cl(G) \setminus Q} \varphi(k, \overline{k}) = \frac{Z}{\varphi(q)} \sum_{y} \prod_{K \in Cl(G) \setminus Q} \varphi(k, \overline{k}) \frac{\varphi(q)}{Z} = \frac{Z}{\varphi(q)} p(q|\theta_G, G)$$

  where $\overline{k}$ denotes the elements of q corresponding to the elements of $K \cap Q$.
- Putting together the results above, we have that

$$\frac{\partial p(d_{1:N}|\theta_G, G)/N}{\partial \varphi(q)} = \frac{p_e(q)}{\varphi(q)} - \frac{p(q|\theta_G, G)}{\varphi(q)}$$

Parameter Learning for MNs: Iterative Proportional Fitting Procedure

- Setting the derivative to zero gives [2]

$$\varphi^{ML}(k) = \varphi(k) p_e(k) / p(k|\theta_G, G)$$

for all $K \in Cl(G)$. **No closed form solution** but ...

IPFP

Set $p(X)$ to some initial probability distribution
Compute $\phi(K)$ for all $K \in Cs(G)$ as shown before
Set $\psi(K) = \exp \phi(K)$ for all $K \in Cs(G)$
Compute $\varphi(K)$ for all $K \in Cl(G)$ as shown before
Repeat until convergence
    Set $\varphi(K) = \varphi(K) p_e(K) / p(K|\theta_G, G)$ for all $K \in Cl(G)$

- Iterative coordinate ascend method.
- Note that computing $p(K|\theta_G, G)$ in the last line requires inference. Moreover, the multiplication and division are elementwise.
- Initializing the factors $\varphi(K)$ from a probability distribution instead of directly ensures that $Z = 1$ and, thus, it does not need to be computed, which is NP-hard.

[2] The log likelihood function is concave.

Parameter Learning for MNs: Iterative Proportional Fitting Procedure

- We now show that the IPFP preserves Z across iterations.
- Let the factor updated in the current iteration have the superscript $t+1$, whereas the rest of the factors have the superscript t. Let $Q \in Cl(G)$ and $Y = X \setminus Q$. Then

$$p^{t+1}(Q|\theta_G, G) = \sum_w p^{t+1}(w, Q|\theta_G, G) = \sum_y \varphi^{t+1}(Q) \frac{1}{Z^{t+1}} \prod_{K \in Cl(G) \setminus Q} \varphi^t(k)$$

$$= \sum_y \varphi^t(Q) \frac{p_e(Q)}{p^t(Q|\theta_G, G)} \frac{1}{Z^{t+1}} \prod_{K \in Cl(G) \setminus Q} \varphi^t(k) = \frac{p_e(Q)}{p^t(Q|\theta_G, G)} \frac{Z^t}{Z^{t+1}} \sum_w p^t(w, Q|\theta_G, G)$$

$$= \frac{p_e(Q)}{p^t(Q|\theta_G, G)} \frac{Z^t}{Z^{t+1}} p^t(Q|\theta_G, G) = p_e(Q) \frac{Z^t}{Z^{t+1}}$$

- Then

$$1 = \sum_q p^{t+1}(q|\theta_G, G) = \frac{Z^t}{Z^{t+1}} \sum_q p_e(q) = \frac{Z^t}{Z^{t+1}}$$

- **Exercise**. Sketch how to perform parameter learning for MNs from an incomplete sample.