10-708: Probabilistic Graphical Models, Spring 2017

6 : Learning Fully Observed Bayesian Networks

Lecturer: Eric P. Xing *Scribes: Lingxue Zhu, Min Lee*

1 Introduction: Learning Graphical Models

In general, given a set of independent samples, the goal for learning graphical models (GMs) involves finding the best (i) graph structures (structural learning) and (ii) conditional probabilities (parameter learning). There are different scenarios for learning GMs, depending on whether data is fully or partially observed, and whether the graph is directed or undirected. The principal of learning GMs (for both structural and parameter learning) can be based on different objectives. Some examples include

1. maximal likelihood estimation (MLE)
2. Bayesian estimation
3. conditional likelihood
4. maximal margin (a non-probabilistic objective)
5. maximum entropy

2 Maximum Likelihood Structural Learning

The goal for structural learning is to find the "optimal" topology of the graph. Although many heuristics of "optimality" have been proposed, most of the algorithms provide no theoretical guarantees. In this class, we will focus on two classes of algorithms for *guaranteed* structural learning: (i) the Chow-Liu algorithm for trees (this lecture); and (ii) the neighborhood selection for pairwise MRFs (later lectures).

Structural learning is a challenging task. For n nodes, there are $O(2^{n^2})$ different possible graphs, which forms a huge search space. Even if we limit our attention to trees, there are still $O(n!)$ different possible trees. Although no optimal algorithm is known for finding a general graph, there do exist an algorithm for finding optimal trees, known as the Chow-Liu algorithm, if the objective function is decomposable to edge-related elements. We will see that the likelihood function is such a decomposable objective.

2.1 Information theoretic interpretation of maximum likelihood

In a Bayesian Network, given M observed data points $D = \{x_{n,i}\}$, where $n = 1, ..., M$ and i denotes the node index. Let $\pi_i(G)$ denote the indices of i's parents under graph structure G, and $\theta_{i|\pi_i(G)}$ denote the parameters of the CPD for node i. Then the log-likelihood function with respect to the graph structure G

and parameters θ_G is:

$$\begin{aligned} l(\theta_G, G; D) &= \log p(D \mid \theta_G, G) \\ &= \log \prod_n \prod_i p(x_{n,i} \mid \mathbf{x}_{n,\pi_i(G)}; \theta_{i|\pi_i(G)}) \\ &= \sum_i \left(\sum_n \log p(x_{n,i} \mid \mathbf{x}_{n,\pi_i(G)}; \theta_{i|\pi_i(G)}) \right) \\ &= M \sum_i \left(\sum_{x_i, \mathbf{x}_{\pi_i(G)}} \frac{\text{count}(x_i, \mathbf{x}_{\pi_i(G)})}{M} \log p(x_i \mid \mathbf{x}_{\pi_i(G)}; \theta_{i|\pi_i(G)}) \right) \\ &= M \sum_i \left(\sum_{x_i, \mathbf{x}_{\pi_i(G)}} \hat{p}(x_i, \mathbf{x}_{\pi_i(G)}) \log p(x_i \mid \mathbf{x}_{\pi_i(G)}; \theta_{i|\pi_i(G)}) \right) \end{aligned}$$

If we plug in the empirical probability, we get the following decomposition:

$$\begin{aligned} l(\theta_G, G; D) &= \log \hat{p}(D \mid \theta_G, G) \\ &= M \sum_i \left(\sum_{x_i, \mathbf{x}_{\pi_i(G)}} \hat{p}(x_i, \mathbf{x}_{\pi_i(G)}) \log \frac{\hat{p}(x_i, \mathbf{x}_{\pi_i(G)}; \theta_{i|\pi_i(G)})}{\hat{p}(\mathbf{x}_{\pi_i(G)})} \frac{\hat{p}(x_i)}{\hat{p}(x_i)} \right) \\ &= M \sum_i \left(\sum_{x_i, \mathbf{x}_{\pi_i(G)}} \hat{p}(x_i, \mathbf{x}_{\pi_i(G)}) \log \frac{\hat{p}(x_i, \mathbf{x}_{\pi_i(G)}; \theta_{i|\pi_i(G)})}{\hat{p}(\mathbf{x}_{\pi_i(G)}) \hat{p}(x_i)} \right) - M \sum_i \left(\sum_{x_i} \hat{p}(x_i) \log \hat{p}(x_i) \right) \\ &= M \sum_i \hat{I}(x_i, \mathbf{x}_{\pi_i(G)}) - M \sum_i \hat{H}(x_i) \end{aligned} \tag{1}$$

where $\hat{I}(x_i, x_{\pi_i(G)})$ is the mutual information between x_i and parents of x_i, and $\hat{H}(x_i)$ is the entropy of node x_i, under the empirical probability $\hat{p}$ on the data points. Note that the second term $\hat{H}(x_i)$ doesn't depend on the graph structure.

2.2 The Chow-Liu tree learning algorithm

When a graph G has tree structure, every node has at most one parent, and the decomposition in eq.(1) becomes:

(i) the first term becomes $M \sum_{j \to i} \hat{I}(x_i, x_j)$, which only involves edge-related elements;

(ii) the second term becomes $M \sum_i \hat{H}(x_i)$, which only involves the marginal probability on each node.

Based on this observation, the Chow-Liu algorithm finds the optimal tree structure that maximizes the log-likelihood as follows:

1. For each pair of variables x_i, x_j, compute the empirical mutual information

$$\hat{I}(x_i, x_j) = \sum_{x_i, x_j} \hat{p}(x_i, x_j) \log \frac{\hat{p}(x_i, x_j)}{\hat{p}(x_i) \hat{p}(x_j)}$$

where $\hat{p}(x_i, x_j) = \text{count}(x_i, x_j)/M$, $\hat{p}(x_i) = \text{count}(x_i)/M$ are the empirical distributions.

2. Define a graph over nodes $x_1, ..., x_n$, and assign weight $\hat{I}(x_i, x_j)$ over each edge $i \sim j$.

3. Find the optimal tree with the maximum sum of edge-weights (maximum weight spanning tree). Finally, we determine the direction in the tree: first pick any node as root, then perform a breadth-first-search to define directions.

Note that if there are several I-equivalent optimal trees, the Chow-Liu algorithm will find one of them.

2.3 Structural learning for general graphs

In general, learning the optimal structure for general graphs is NP-hard:

Theorem 1. *The problem of learning a BN structure with at most d parents is NP-hard for any fixed $d \geq 2$.*

Most structure learning approaches use heuristics that exploit decomposition in different ways. However, no theoretical guarantees are provided.

3 Parameter Learning in Fully Observed BNs

Suppose the graph structural is known and fixed, the goal in this section is to learn the graph parameters based on N independent, identically distributed data points $D = \{x_1, ..., x_N\}$. In general, in a graph with M nodes, each training data $\mathbf{x}_n$ is a M-dimensional vector $(x_{n,1}, ..., x_{n,M})$. In this section, we consider the scenario where each $\mathbf{x}_n$ is completely observable.

We start from density estimation, which can be viewed as learning parameters in a single-node graphical model and is the building block for general graphical models. We know that there are two different estimation approaches: (i) maximum likelihood estimation (MLE), and (ii) Bayesian estimates.

3.1 Parameter estimation in Multinomial distribution

First, we illustrate the parameter learning procedure for a discrete distribution, the Multinomial distribution. For the ease of derivation, we use the one-hot representation of Multinomial distribution in the following analysis. Specifically, if we observed N *i.i.d.* die rolls on a K-sided die, then each observation $\vec{x}_n = (x_{n,1}, ..., x_{n,K})'$ is a K-dimensional binary vector, where $x_{n,k} \in \{0, 1\}$, $\sum_{k=1}^{K} x_{n,k} = 1$, and

$$\mathbb{P}(X_{n,k} = 1) = \theta_k, \quad \sum_{k=1}^{K} \theta_k = 1$$

Then the likelihood function of dataset $D = \{\vec{x}_1, ..., \vec{x}_N\}$ is

$$p(D; \theta) = \prod_{n=1}^{N} \left(\prod_{k=1}^{K} \theta_k^{x_{n,k}} \right) = \prod_{k=1}^{K} \theta_k^{\sum_n x_{n,k}} = \prod_{k=1}^{K} \theta_k^{n_k}$$

where n_k is the number of times that $x_{n,k} = 1$ in the dataset, and apparently $\sum_{k=1}^{K} n_k = N$.

3.1.1 Maximum likelihood estimation

Finding the MLE for the Multinomial distribution is a optimization problem over $\vec{\theta} = (\theta_1, ..., \theta_K)$, subject to the constrain $\sum_{k=1}^{K} \theta_k = 1$. Using the log-likelihood with Lagrange multiplier, we get the constrained cost function

$$\bar{l} = \sum_{k=1}^{K} n_k \log \theta_k + \lambda \left(1 - \sum_{k=1}^{K} \theta_k\right)$$

Taking derivatives with respect to θ_k, it is easy to see that

$$\frac{\partial \bar{l}}{\partial \theta_k} = \frac{n_k}{\theta_k} - \lambda = 0 \quad \Rightarrow \quad N = \sum_k n_k = \sum_k \lambda \theta_k = \lambda$$

and we obtain the MLE

$$\hat{\theta}_{k,MLE} = \frac{n_k}{N}$$

which is the fraction of each realization in the dataset. Note that the estimation only depends on $\vec{n} = (n_1, ..., n_K)$, which are the *sufficient statistics* of data D.

3.1.2 Bayesian estimation with Dirichlet prior

In Bayesian estimation, θ is treated as a random variable following a prior distribution. Here, we introduce the Dirichlet distribution $\mathrm{Dir}(\vec{\alpha})$ for $\vec{\theta}$ over a simplex:

$$p(\vec{\theta}) = \frac{\Gamma(\sum_k \alpha_k)}{\prod_k \Gamma(\alpha_k)} \prod_k \theta_k^{\alpha_k - 1}$$

If $\vec{\theta} \sim \mathrm{Dir}(\vec{\alpha})$, then we can compute the posterior distribution of $\vec{\theta}$:

$$p(\vec{\theta} \,|\, x_1, ..., x_N) = \frac{p(x_1, ..., x_N \,|\, \vec{\theta})\, p(\vec{\theta})}{p(x_1, ..., x_N)} \propto \prod_k \theta_k^{n_k} \prod_k \theta_k^{\alpha_k - 1} = \prod_k \theta_k^{\alpha_k + n_k - 1}$$

and we see that the posterior of $\vec{\theta}$ follows $\mathrm{Dir}(\vec{\alpha} + \vec{n})$, where $\vec{n} = (n_1, ..., n_K)$ and $\vec{\alpha} + \vec{n}$ is a K-dimensional vector obtained by element-wise addition. Therefore, the posterior distribution belongs to the same family as the prior. Such a prior is called a *conjugate prior*. Furthermore, the posterior mean of θ_k is

$$\mathbb{E}_{\theta|x}[\theta_k] = \int_\theta \theta_k p(\theta|x) = \frac{n_k + \alpha_k}{N + \sum_{k'} \alpha_{k'}}$$

Interestingly, the Dirichlet prior can be understood as adding pseudo counts to the data, and the magnitude of $\vec{\alpha}$ corresponds to the strength of the prior.

One of the advantages of using conjugate prior is that we can perform sequential Bayesian updating. Starting with the Dirichlet prior, we have

$$p(\vec{\theta}|\vec{\alpha}) = \mathrm{Dir}(\vec{\theta} : \vec{\alpha})$$

After observing N' samples with sufficient statistics $\vec{n}' = (n'_1, ..., n'_K)$, the posterior becomes

$$p(\vec{\theta}|\vec{\alpha}, \vec{n}') = \mathrm{Dir}(\vec{\theta} : \vec{\alpha} + \vec{n}')$$

If we further observe another N'' samples with sufficient statistics $\vec{n}'' = (n''_1, ..., n''_K)$, the posterior becomes

$$p(\vec{\theta}|\vec{\alpha}, \vec{n}', \vec{n}'') = \mathrm{Dir}(\vec{\theta} : \vec{\alpha} + \vec{n}' + \vec{n}'')$$

So we can update the posterior with sequentially observed data.

3.1.3 The Logistic Normal prior

The limitation of the Dirichlet prior is that it cannot capture the correlation among different outcomes. To overcome this issue, we introduce the logistic normal prior that allows more flexible covariance structures. The Logistic normal distribution $LN_K(\mu, \Sigma)$ for $\vec{\theta} \in \mathbb{R}^K$ is defined as follows:

$$\begin{aligned}(\gamma_1, ..., \gamma_{K-1}) &\sim \text{Normal}_{K-1}(\mu, \Sigma), \quad \gamma_K = 0 \\ \theta_k &= \exp\left\{\gamma_k - \log\left(1 + \sum_{k=1}^{K-1} e^{\gamma_k}\right)\right\}\end{aligned} \tag{2}$$

However, the drawback of Logistic Normal is that it is non-conjugate, so it is computationally more demanding for parameter learning.

3.2 Parameter Estimation for Multivariate Gaussian

This section describes the parameter estimation of multivariate Gaussian using MLE and Bayesian estimation.

3.2.1 MLE for Multivariate Gaussian

Given a multivariate Gaussian distribution, $p(X; \mu, \Sigma) = \frac{1}{2\pi^{N/2}|\Sigma|^{\frac{1}{2}}} exp\{-\frac{1}{2}(X-\mu)^T\Sigma(X-\mu)\}$,
let first find the loglikelihood.

$$l(X; \mu, \Sigma) = -\frac{N}{2} ln|\Sigma| - \frac{1}{2}\sum_{i=1}^{N}(x_i - \mu)^T\Sigma(x_i - \mu))$$

To find MLE of μ, we solve following equation by taking its derivative w.r.t μ and setting it to zero

$$\frac{\partial}{\partial \mu}(-\frac{1}{2}\sum_{i=1}^{N}(-2\mu^T\Sigma^{-1}x_i + \mu^T\Sigma^{-1}\mu)) = 0$$

To find the MLE of Σ, we can rewrite loglikelihood as follows

$$l(X; \mu, \Sigma) = -\frac{N}{2} ln|\Sigma| - \frac{1}{2} Trace(\Sigma^{-1}\sum_{i=1}^{N}(x_i - \mu)(x_i - \mu)^T))$$

Then, we can find the MLE of μ and Σ as follows:

$$\begin{aligned}\mu_{MLE} &= \frac{1}{N}\sum_i x_i \\ \Sigma_{MLE} &= \frac{1}{N}\sum_i (x_i - \mu_{MLE})(x_i - \mu_{MLE})^T\end{aligned}$$

3.2.2 Bayesian Estimation for Multivariate Gaussian

We want to pursue this Bayesian approach for following reasons:
1) Update estimates sequentially over time

2) Have prior knowledge about the expected magnitude of the parameters
3) MLE for Σ may not be full rank

Let's consider the condition when μ is unknown and Σ is known.
Given the conjugate prior, $p(\mu) = (2\pi\tau^2)^{-N/2} exp\{-(\mu-\mu_0)^2/2\tau^2\}$ and
the likelihood of the Gaussian, $p(x|\mu) = (2\pi\sigma^2) exp\{-\sum_{i=1}^{N}(x_i-\mu)^2/2\sigma^2\}$, we can find the joint distribution as multiplication of two Gaussian distributions.

$$p(\mu|x) = (2\pi\sigma^2) exp\{-\sum_{i=1}^{N}(x_i-\mu)^2/2\sigma^2\} * (2\pi\tau^2)^{-N/2} exp\{-(\mu-\mu_0)^2/2\tau^2\}$$

Then, we can find the posterior distribution as follows

$$p(\mu|x) \propto (2\pi\tilde{\sigma}^{-1/2}) exp\{-(\mu-\tilde{\mu})^2/2\tilde{\sigma}^2\}$$

where $\tilde{\mu} = \frac{N/\sigma^2}{N/\sigma^2+1/\tau^2}\bar{x} + \frac{1/\tau^2}{N/\sigma^2+1/\tau^2}\mu_0$ and $\tilde{\sigma}^2 = (\frac{N}{\sigma^2} + \frac{1}{\tau^2})^{-1}$

The posterior mean is a convex combination of the prior and the MLE with weights proportional to the relative noise levels.

3.3 MLE for General Bayesian Networks

3.3.1 Decomposing the Likelihood of a Bayesian Network

Let assume that the parameters for each CPD are globally independent and all nodes are fully observed. Then, we can decompose loglikelihood into a sum of local terms, one per node:

$$l(\theta; D) = logp(D|\theta) = \sum_i (\sum_n logp(x_{n,i}|x_{n,\pi_i}, \theta_i))$$

For discrete models, assume each CPD is represented as a high dimensional table, where $\theta_{ijk} = p(X_i = j|X_{pi_i} = k)$. The sufficient statistics are counts of family configuration, $n_{ijk} = \sum_n x^j_{n,i} x^k_{n,pi_i}$.
Then, the loglikelihood is

$$l(\theta; D) = log \prod_{i,j,k} \theta_{ijk}^{n_{ijk} = \sum_{i,j,k} n_{ijk} log\theta_{ijk}}$$

Adding a Largrange term to enforce $\sum_j \theta ijk = 1$, we obtain

$$\theta^{ML}_{ijk} = \frac{n_{ijk}}{\sum_{j'} n_{ij'k}}$$

We find that MLE becomes following ratio: the counts of the joint configuration of a child and its parents to the counts of the marginal counts of its parents in that configuration.

3.3.2 Learning a Markov chain transition matrix

Consider a stationary 1st order Markov Model with the initial state probability vector, $\pi_k = p(X^k_1 = 1)$ and state transition probability matrix $A_{ij} = p(X^j_t = 1|X^i_{t-1} = 1)$ where $\sum_j A_{ij} = 1$

MLE of A_{ij} can be expressed as

$$A_{ij}^{ML} = \frac{\#(i \to j)}{\#(i \to *)} = \frac{\sum_n \sum_{t=2}^{T} x_{n,t-1^i x_{n,t}^j}}{\sum_n \sum_{t=2}^{T} x_{n,t-1}^i}$$

When we have a sparse data and $i \to j$ is not occurred, then we will have $A_{ij} = 0$. To handle this sparse data issue, we can use a standard hack, such as backoff smoothing or deleted interpolation

$$\hat{A}_{i\to} = \lambda\eta + (1-\lambda)A_{i\to}^{ML}$$

3.3.3 Learning a Hidden Markov Model (HMM)

Consider a HMM with the observed sequence $x_1, x_2, ..., x_T$ and the hidden state sequence, $y_1, y_2, ..., y_T$.
Let define A_{ij} = # of state transition $i \to j$ occurs in y and B_{ik} = # of state i in y emits k in x. Then, MLE are computed as

$$a_{ij}^{ML} = \frac{\#(i \to j)}{\#(i \to *)} = \frac{A_{ij}}{\sum_{j'} A_{ij'}}$$

$$b_{ik}^{ML} = \frac{\#(i \to k)}{\#(i \to *)} = \frac{B_{ij}}{\sum_{k'} B_{ik'}}$$

If we have small traing sets, we can introduce pseudo-counts that represent our prior belief:R_{ij}, S_{ij}.
Then, define A_{ij} = # of state transition $i \to j$ occurs in y + R_{ij} and B_{ik} = # of state i in y emits k in x + S_{ik}.