Inference and Representation: Bayesian Networks

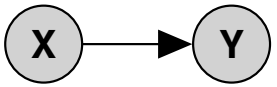
Rahul G. Krishnan

New York University

Lab 3, September 21, 2016

Outline

Parameterizations of Conditional Probability Distributions

- How do we parameterize the conditional probability distributions in a Bayesian network?
- Lets start with the simplest Bayesian network you're all familiar with: 

- Underlies logistic regression, linear regression, image classification and many other methods
- $y \sim P(f(x;\theta))$ is the assumption and θ are the parameters of the Bayesian network

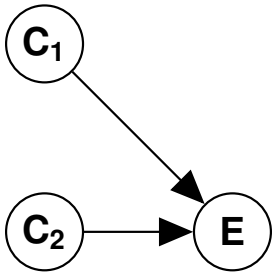
Logistic Regression

- Used when the variable Y is binary i.e $Y \in \{0, 1\}$
- $p(y|x;\theta) = \frac{1}{1+\exp(w^T x+b)}$
- The weights $\theta = \{w, b\}$ are the parameters to be learned

Deep Convolutional Network

- What if y is the set of objects in my dataset? i.e y is a categorical random variable
- Use a convolutional neural network to obtain a feature vector $\vec{z} = g(x; \theta)$
- $p(y_j|x; \theta) = \frac{\exp(z_j)}{\sum_{k=1}^{K} \exp(z_k)}$
- θ corresponds to all the parameters in the neural network

Noisy-Or Parameterization



$$P(E=1|pa(E)) = P(E=1|C_1, C_2)$$
$$= 1 - \underbrace{\prod_{k\in\{1,2\}} \underbrace{(1 - \underbrace{f_k}_{\text{Prob. that } C_k \text{ caused } E})}_{\text{Prob. that } C_k \text{ did } \textit{not} \text{ cause } E}}_{\text{Prob. that neither } C_1 \text{ nor } C_2 \text{ caused } E}$$

The parameters of the model are $\theta = \{f_1, f_2\}$

Learning Bayesian networks

- How do we learn graphical models from data? Maximize the likelihood of data under the model
- We're typically given a finite dataset $\mathcal{D} = \{X_1, \ldots, X_N\}$
- Lets learn a fully observed Bayesian network where we are interested in learning the conditional probability tables.
- The following slides are taken from David's lectures

ML estimation in Bayesian networks

- Suppose that we know the Bayesian network structure G
- $\theta_{x_i|\mathbf{x}_{pa(i)}}$: value of the CPD $p(x_i \mid \mathbf{x}_{pa(i)}; \theta)$
- Maximum likelihood estimation corresponds to solving:

$$\max_{\theta} \sum_{n=1}^{N} \log p(\mathbf{x}^n; \theta) = \max_{\theta} \ell(\theta; \mathcal{D})$$

- This is equal to:

$$\begin{aligned} \max_{\theta} \sum_{n=1}^{N} \log p(\mathbf{x}^n; \theta) &= \max_{\theta} \sum_{n=1}^{N} \sum_{i=1}^{|V|} \log p(x_i^n \mid \mathbf{x}_{pa(i)}^n; \theta) \\ &= \max_{\theta} \sum_{i=1}^{|V|} \sum_{n=1}^{N} \log p(x_i^n \mid \mathbf{x}_{pa(i)}^n; \theta) \end{aligned}$$

- An independent optimization problem for each CPD!

ML estimation in Bayesian networks

- Suppose that we know the Bayesian network structure G
- $\theta_{x_i|\mathbf{x}_{pa(i)}}$: value of the CPD $p(x_i \mid \mathbf{x}_{pa(i)}; \theta)$
- Maximum likelihood estimation corresponds to solving:

$$\max_{\theta} \sum_{n=1}^{N} \log p(\mathbf{x}^n; \theta) = \max_{\theta} \ell(\theta; \mathcal{D})$$

- This is equal to:

$$\begin{aligned} \max_{\theta} \sum_{n=1}^{N} \log p(\mathbf{x}^n; \theta) &= \max_{\theta} \sum_{n=1}^{N} \sum_{i=1}^{|V|} \log p(x_i^n \mid \mathbf{x}_{pa(i)}^n; \theta) \\ &= \max_{\theta} \sum_{i=1}^{|V|} \sum_{n=1}^{N} \log p(x_i^n \mid \mathbf{x}_{pa(i)}^n; \theta) \end{aligned}$$

- An independent optimization problem for each CPD!

ML estimation in Bayesian networks

- Suppose that we know the Bayesian network structure G
- $\theta_{x_i|\mathbf{x}_{pa(i)}}$: value of the CPD $p(x_i \mid \mathbf{x}_{pa(i)}; \theta)$
- Maximum likelihood estimation corresponds to solving:

$$\max_{\theta} \sum_{n=1}^{N} \log p(\mathbf{x}^n; \theta) = \max_{\theta} \ell(\theta; \mathcal{D})$$

- This is equal to:

$$\begin{aligned} \max_{\theta} \sum_{n=1}^{N} \log p(\mathbf{x}^n; \theta) &= \max_{\theta} \sum_{n=1}^{N} \sum_{i=1}^{|V|} \log p(x_i^n \mid \mathbf{x}_{pa(i)}^n; \theta) \\ &= \max_{\theta} \sum_{i=1}^{|V|} \sum_{n=1}^{N} \log p(x_i^n \mid \mathbf{x}_{pa(i)}^n; \theta) \end{aligned}$$

- An independent optimization problem for each CPD!

ML estimation in Bayesian networks

$$\begin{aligned}
\ell(\theta; \mathcal{D}) = \log p(\mathcal{D}; \theta) &= \sum_{i=1}^{|V|} \sum_{n=1}^{N} \log p(x_i^n \mid \mathbf{x}_{pa(i)}^n; \theta) \\
&= \sum_{i=1}^{|V|} \sum_{\mathbf{x}_{pa(i)}} \sum_{\mathbf{x}_i} \sum_{\substack{\hat{\mathbf{x}} \in \mathcal{D}: \\ \hat{\mathbf{x}}_i, \hat{\mathbf{x}}_{pa(i)} = \mathbf{x}_i, \mathbf{x}_{pa(i)}}} \log p(x_i \mid \mathbf{x}_{pa(i)}; \theta) \\
&= \sum_{i=1}^{|V|} \sum_{\mathbf{x}_{pa(i)}} \sum_{\mathbf{x}_i} N_{x_i, \mathbf{x}_{pa(i)}} \log \theta_{x_i \mid \mathbf{x}_{pa(i)}},
\end{aligned}$$

where $N_{x_i, \mathbf{x}_{pa(i)}}$ is the number of times that the (partial) assignment $x_i, \mathbf{x}_{pa(i)}$ is observed in the training data

ML estimation in Bayesian networks

$$
\begin{aligned}
\ell(\theta; \mathcal{D}) = \log p(\mathcal{D}; \theta) &= \sum_{i=1}^{|V|} \sum_{n=1}^{N} \log p(x_i^n \mid \mathbf{x}_{pa(i)}^n; \theta) \\
&= \sum_{i=1}^{|V|} \sum_{\mathbf{x}_{pa(i)}} \sum_{\mathbf{x}_i} \sum_{\substack{\hat{\mathbf{x}} \in \mathcal{D}: \\ \hat{\mathbf{x}}_i, \hat{\mathbf{x}}_{pa(i)} = \mathbf{x}_i, \mathbf{x}_{pa(i)}}} \log p(x_i \mid \mathbf{x}_{pa(i)}; \theta) \\
&= \sum_{i=1}^{|V|} \sum_{\mathbf{x}_{pa(i)}} \sum_{\mathbf{x}_i} N_{x_i, \mathbf{x}_{pa(i)}} \log \theta_{x_i \mid \mathbf{x}_{pa(i)}},
\end{aligned}
$$

where $N_{x_i, \mathbf{x}_{pa(i)}}$ is the number of times that the (partial) assignment $x_i, \mathbf{x}_{pa(i)}$ is observed in the training data

ML estimation in Bayesian networks

$$
\begin{aligned}
\ell(\theta; \mathcal{D}) = \log p(\mathcal{D}; \theta) &= \sum_{i=1}^{|V|} \sum_{n=1}^{N} \log p(x_i^n \mid \mathbf{x}_{pa(i)}^n; \theta) \\
&= \sum_{i=1}^{|V|} \sum_{\mathbf{x}_{pa(i)}} \sum_{\mathbf{x}_i} \sum_{\substack{\hat{\mathbf{x}} \in \mathcal{D}: \\ \hat{\mathbf{x}}_i, \hat{\mathbf{x}}_{pa(i)} = \mathbf{x}_i, \mathbf{x}_{pa(i)}}} \log p(x_i \mid \mathbf{x}_{pa(i)}; \theta) \\
&= \sum_{i=1}^{|V|} \sum_{\mathbf{x}_{pa(i)}} \sum_{\mathbf{x}_i} N_{x_i, \mathbf{x}_{pa(i)}} \log \theta_{x_i \mid \mathbf{x}_{pa(i)}},
\end{aligned}
$$

where $N_{x_i, \mathbf{x}_{pa(i)}}$ is the number of times that the (partial) assignment $x_i, \mathbf{x}_{pa(i)}$ is observed in the training data

ML estimation in Bayesian networks

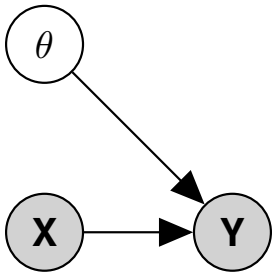
- We have the closed form ML solution:

$$\theta^{ML}_{x_i|\mathbf{x}_{pa(i)}} = \frac{N_{x_i,\mathbf{x}_{pa(i)}}}{\sum_{\hat{x}_i} N_{\hat{x}_i,\mathbf{x}_{pa(i)}}}$$

- We were able to estimate each CPD independently because the objective **decomposes** by variable and parent assignment

Regularization as Bayesian Inference

- So far, we've created a distinction between random variables and parameters that are part of the CPDs
- However, our parameters may also be treated as random variables
- Learning θ turns into a an inference problem, specifically Maximum A-Posteriori inference
- θ is an unobserved latent variable that we will perform MAP inference to estimate



Normal Priors as L2 Regularization

$$\begin{aligned}\underset{\theta}{\operatorname{argmax}}\ \log p(\theta|x,y) &= \underset{\theta}{\operatorname{argmax}}\ \log \frac{p(x,y|\theta)p(\theta)}{p(x,y)} \\ &= \underset{\theta}{\operatorname{argmax}}\ \log p(x,y|\theta) + \log p(\theta) + \text{constant} \\ &= \underset{\theta}{\operatorname{argmax}}\ \log p(y|x,\theta) + \log p(\theta) + \text{constant}\end{aligned}$$

- Assuming $p(\theta) = \text{Normal}(0, C\mathbb{I})$ (where Λ diagonal) then, $\log p(\theta) = C^{-1}\theta^2$ (ignoring constants)
- $\underset{\theta}{\operatorname{argmax}}\ \log p(y|x,\theta) + C^{-1}\theta^2$
- Learning with L2 regularization

Structure Learning in Bayesian networks

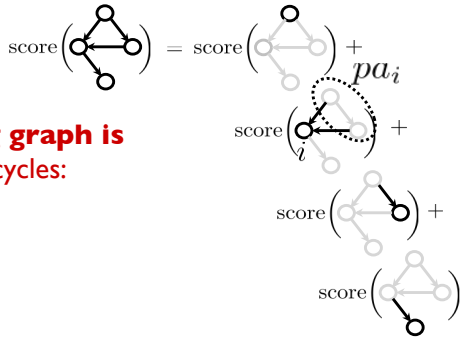
To learn the structure of a Bayesian Network given data, we will have to contend with the following:

- Search space: What set of skeletons, directed edges should we consider?
- Moving between structures: How should we move from one structure to another?
- Scoring functions: How do we evaluate how good our current structure is?

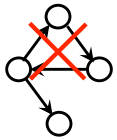
Approach 1: Score-based approaches

- Q: What is the maximum likelihood graph?
  - A: The complete graph! Because $H(X \mid Y) \leq H(X)$ **always**.
  - Must *regularize* to recover a sparse graph and have any hope of recoverying true structure
  - BIC and BDe (Bayesian Dirichlet score) are decomposable (they factorize over the CPDs in the graph)
- Obtain a combinatorial optimization problem over acyclic graphs:

$$\text{score}(G; D) = \sum_{i=1}^{n} \text{score}(i|pa_i, D)$$



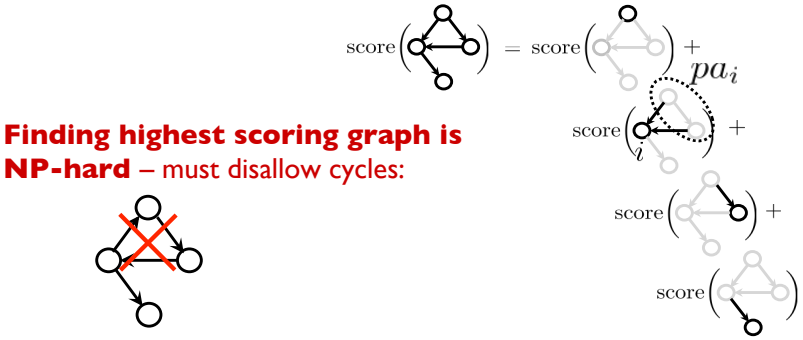
Finding highest scoring graph is NP-hard – must disallow cycles:



Approach 1: Score-based approaches

- Q: What is the maximum likelihood graph?
  - A: The complete graph! Because $H(X \mid Y) \leq H(X)$ **always**.
  - Must *regularize* to recover a sparse graph and have any hope of recoverying true structure
  - BIC and BDe (Bayesian Dirichlet score) are decomposable (they factorize over the CPDs in the graph)
- Obtain a combinatorial optimization problem over acyclic graphs:

$$\text{score}(G; D) = \sum_{i=1}^{n} \text{score}(i|pa_i, D)$$



Bayesian Information Criterion

- Bayesian Information Criterion is one such score function
- Given dataset $\mathcal{D}$ of size N, on a model m with parameters θ^m where d^m is the number of parameters:

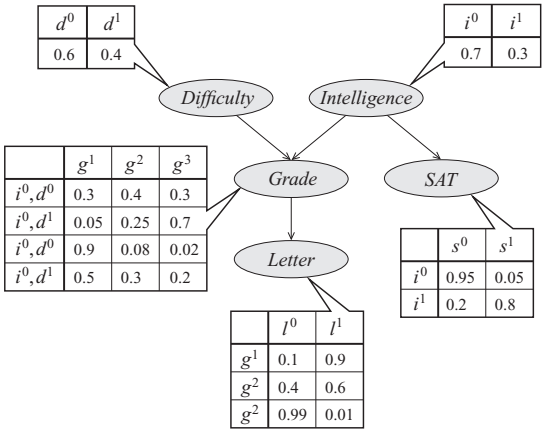
$$\log p(\mathcal{D}) \propto \underbrace{\log p(\mathcal{D}; \theta^m_{\text{ML}})}_{\text{How well does the current structure explain the data}} - \underbrace{\frac{d^m \log N}{2}}_{\text{Regularize so that we do not find a fully connected graph}}$$

- We start with a model m and make local changes such as adding and subtracting edges till we find a local-optima of the above score function

Demo: Structure Learning with Causal Explorer

- Causal Explorer `http://www.dsl-lab.org/supplements/mmhc_paper/mmhc_index.html` available at `http://www.dsl-lab.org/causal_explorer/`
- MATLAB package that implements structure learning algorithms
- Min-Max Hill Climbing (MMHC) Algorithm : Starts from a skeleton of the Bayesian network and uses a greedy search procedure to orient the edges of the graph
- Real-World Alarm Dataset `https://www.cs.cmu.edu/afs/cs/project/jair/pub/volume19/leisink03a-html/node11.html`. Monitoring patients in intensive care units

Approach 2: Independence Tests



d^0	d^1
0.6	0.4

i^0	i^1
0.7	0.3

	g^1	g^2	g^3
i^0, d^0	0.3	0.4	0.3
i^0, d^1	0.05	0.25	0.7
i^0, d^0	0.9	0.08	0.02
i^0, d^1	0.5	0.3	0.2

	s^0	s^1
i^0	0.95	0.05
i^1	0.2	0.8

	l^0	l^1
g^1	0.1	0.9
g^2	0.4	0.6
g^2	0.99	0.01

The network structure implies several conditional independence statements:

$$D \perp I$$

$$G \perp S \mid I$$

$$D \perp L \mid G$$

$$L \perp S \mid G$$

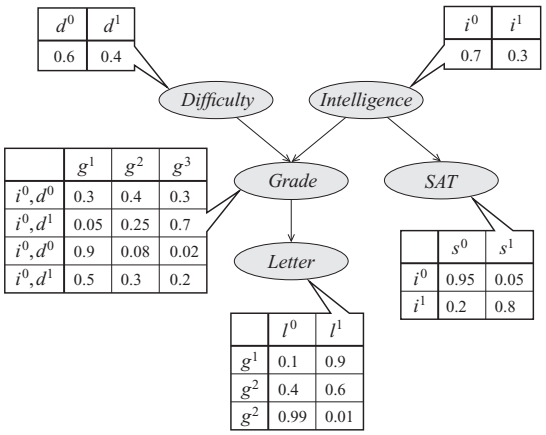
$$L \perp S \mid I$$

$$D \perp S$$

If two variables are (conditionally) independent, structure has no edge between them

- Must make assumption that data is drawn from an I-map of the graph
- Very brittle: if we say that $X_i \perp X_j | X_{\mathbf{v}}$ and they in fact are not, the resulting structure can be very off

Approach 2: Independence Tests



d^0	d^1
0.6	0.4

i^0	i^1
0.7	0.3

	g^1	g^2	g^3
i^0, d^0	0.3	0.4	0.3
i^0, d^1	0.05	0.25	0.7
i^0, d^0	0.9	0.08	0.02
i^0, d^1	0.5	0.3	0.2

	s^0	s^1
i^0	0.95	0.05
i^1	0.2	0.8

	l^0	l^1
g^1	0.1	0.9
g^2	0.4	0.6
g^2	0.99	0.01

The network structure implies several conditional independence statements:

$$D \perp I$$

$$G \perp S \mid I$$

$$D \perp L \mid G$$

$$L \perp S \mid G$$

$$L \perp S \mid I$$

$$D \perp S$$

If two variables are (conditionally) independent, structure has no edge between them

- Must make assumption that data is drawn from an I-map of the graph
- Very brittle: if we say that $X_i \perp X_j | X_{\mathbf{v}}$ and they in fact are not, the resulting structure can be very off

Recap

- Represent a distribution over random variables $X_1, \ldots, X_N$ with an undirected graphical model
- $X_1, \ldots, X_N$ may be discrete or continuous, we will focus on the discrete case here
- Conditional independence as graph separation

Markov Random Fields (undirected graphical models)

- Rather than CPDs, we specify (non-negative) **potential functions** over sets of variables associated with cliques C of the graph,

$$p(x_1, \ldots, x_n) = \frac{1}{Z} \prod_{c \in C} \phi_c(\mathbf{x}_c)$$

- Z is the **partition function** and normalizes the distribution:

$$Z = \sum_{\hat{x}_1, \ldots, \hat{x}_n} \prod_{c \in C} \phi_c(\hat{\mathbf{x}}_c)$$

- Like CPD's, $\phi_c(\mathbf{x}_c)$ can be represented as a table, but it is *not normalized*
- Called *undirected graphical models, Markov random fields* (MRFs), or *Markov networks*

Comparing BNs to MRFs

- There are some $I(p)$'s that can be represented by MRFs but not BNs, and vice versa. (Examples are v-structure, and four friends' hair color from lecture).
- Advantage of MRFs: conditional independance as graph separation
- Disadvantage: hard to compute the partition function (sum over all possible states), often resort to approximations
- Disadvantage: no longer a natural way to *sample* data