Problem Solving with AI Techniques
Bayesian Networks: Learning

Paul Weng

UM-SJTU Joint Institute

VE593, Fall 2018

JOINT INSTITUTE
交大密西根学院

1. How to Build a Bayesian Network?
   - Introduction
   - Parameter Learning
   - Structure Learning

Specifying a Bayes Net

How to specify the structure?

- From prior knowledge of (causal or other) relationships
- From domain experts
- From data (i.e., structure learning)
- By choosing a certain structure

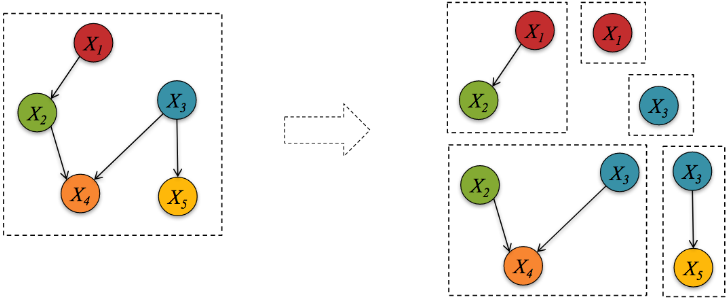
How to specify the conditional probabilities (CPTs or CPDs)?

- From prior knowledge
- From experts
- From data (e.g., parameter learning)

1. How to Build a Bayesian Network?
   - Introduction
   - Parameter Learning
   - Structure Learning

Problem Decomposition: Example

If the structure of the Bayes net is known, the problem of learning the conditional probabilities can be decomposed into independent ones.



from Gormley

$$\mathbb{P}(X_{1:5}) = \mathbb{P}(X_1)\mathbb{P}(X_2 \mid X_1)\mathbb{P}(X_3)\mathbb{P}(X_4 \mid X_2, X_3)\mathbb{P}(X_5 \mid X_5)$$

Maximum Likelihood

- Assumption: Data is generated by a parametric model $\mathbb{P}(\boldsymbol{X} \mid \boldsymbol{\theta})$
- General ML Principle: Given i.i.d. training data $\mathcal{D} = \{\boldsymbol{x}^1, \ldots, \boldsymbol{x}^N\}$
    - compute the likelihood

$$\mathbb{P}(\mathcal{D} \mid \boldsymbol{\theta}) = \prod_{i=1}^{N} \mathbb{P}(\boldsymbol{x}^i \mid \boldsymbol{\theta})$$

    - choose the most likely parameter $\boldsymbol{\theta}^*$

$$\boldsymbol{\theta}^* = \arg\max_{\boldsymbol{\theta}} \mathbb{P}(\mathcal{D} \mid \boldsymbol{\theta})$$

Maximum Likelihood

- Assumption: Data is generated by a parametric model $\mathbb{P}(\boldsymbol{X} \mid \boldsymbol{\theta})$
- General ML Principle: Given i.i.d. training data $\mathcal{D} = \{\boldsymbol{x}^1, \dots, \boldsymbol{x}^N\}$
  - compute the likelihood

$$\mathbb{P}(\mathcal{D} \mid \boldsymbol{\theta}) = \prod_{i=1}^{N} \mathbb{P}(\boldsymbol{x}^i \mid \boldsymbol{\theta})$$

  - choose the most likely parameter $\boldsymbol{\theta}^*$

$$\boldsymbol{\theta}^* = \arg\max_{\boldsymbol{\theta}} \mathbb{P}(\mathcal{D} \mid \boldsymbol{\theta})$$

- In practice, one rather works with the log-likelihood. Why?

Maximum Likelihood

- Assumption: Data is generated by a parametric model $\mathbb{P}(\boldsymbol{X} \mid \boldsymbol{\theta})$
- General ML Principle: Given i.i.d. training data $\mathcal{D} = \{\boldsymbol{x}^1, \dots, \boldsymbol{x}^N\}$
  - compute the likelihood

$$\mathbb{P}(\mathcal{D} \mid \boldsymbol{\theta}) = \prod_{i=1}^{N} \mathbb{P}(\boldsymbol{x}^i \mid \boldsymbol{\theta})$$

  - choose the most likely parameter $\boldsymbol{\theta}^*$

$$\boldsymbol{\theta}^* = \arg\max_{\boldsymbol{\theta}} \mathbb{P}(\mathcal{D} \mid \boldsymbol{\theta})$$

- In practice, one rather works with the log-likelihood. Why?
  - Log-likelihood

$$\log \mathbb{P}(\mathcal{D} \mid \boldsymbol{\theta}) = \sum_{i=1}^{N} \log \mathbb{P}(\boldsymbol{x}^i \mid \boldsymbol{\theta})$$

Maximum Likelihood

- Assumption: Data is generated by a parametric model $\mathbb{P}(\boldsymbol{X} \mid \boldsymbol{\theta})$
- General ML Principle: Given i.i.d. training data $\mathcal{D} = \{\boldsymbol{x}^1, \ldots, \boldsymbol{x}^N\}$
  - compute the likelihood

$$\mathbb{P}(\mathcal{D} \mid \boldsymbol{\theta}) = \prod_{i=1}^{N} \mathbb{P}(\boldsymbol{x}^i \mid \boldsymbol{\theta})$$

  - choose the most likely parameter $\boldsymbol{\theta}^*$

$$\boldsymbol{\theta}^* = \arg\max_{\boldsymbol{\theta}} \mathbb{P}(\mathcal{D} \mid \boldsymbol{\theta})$$

- In practice, one rather works with the log-likelihood. Why?
  - Log-likelihood

$$\log \mathbb{P}(\mathcal{D} \mid \boldsymbol{\theta}) = \sum_{i=1}^{N} \log \mathbb{P}(\boldsymbol{x}^i \mid \boldsymbol{\theta})$$

  - $\boldsymbol{\theta}^* = \arg\max_{\boldsymbol{\theta}} \log \mathbb{P}(\mathcal{D} \mid \boldsymbol{\theta}) = \arg\max_{\boldsymbol{\theta}} \sum_{i=1}^{N} \log \mathbb{P}(\boldsymbol{x}^i \mid \boldsymbol{\theta})$

ML Application to Previous Example

Variables	1	2	3	4	5
X_1	A	A	B	C	C
X_2	T	F	T	F	F
X_3	T	T	F	F	T
X_4	0	1	1	2	0
X_5	F	F	T	T	F

- Application: learn $\mathbb{P}(X_1)$ in previous example
  - What are the parameters for this problem?
  - What is the log-likelihood?

ML Application to Previous Example

Variables	1	2	3	4	5
X_1	A	A	B	C	C
X_2	T	F	T	F	F
X_3	T	T	F	F	T
X_4	0	1	1	2	0
X_5	F	F	T	T	F

- Application: learn $\mathbb{P}(X_1)$ in previous example
  - What are the parameters for this problem?
  - What is the log-likelihood?
- Application: learn $\mathbb{P}(X_3 \mid X_5)$ in previous example
  - What are the parameters for this problem?
  - What is the log-likelihood?

Maximum A Posteriori

- Assumptions: Data is generated by a parametric model $\mathbb{P}(\boldsymbol{X} \mid \boldsymbol{\theta})$ and we have an a priori distribution over parameters $\boldsymbol{\theta}$
- General MAP Principle: Given i.i.d. training data $\mathcal{D} = \{\boldsymbol{x}^1, \ldots, \boldsymbol{x}^N\}$ and assuming that one has a priori distribution $\pi(\boldsymbol{\theta})$
  - compute the posterior
  $$\mathbb{P}(\boldsymbol{\theta} \mid \mathcal{D})$$
  - choose the most probable $\boldsymbol{\theta}$ after observing data
  $$\boldsymbol{\theta}^* = \arg\max_{\boldsymbol{\theta}} \mathbb{P}(\boldsymbol{\theta} \mid \mathcal{D})$$

Maximum A Posteriori

- Assumptions: Data is generated by a parametric model $\mathbb{P}(\boldsymbol{X} \mid \boldsymbol{\theta})$ and we have an a priori distribution over parameters $\boldsymbol{\theta}$
- General MAP Principle: Given i.i.d. training data $\mathcal{D} = \{\boldsymbol{x}^1, \dots, \boldsymbol{x}^N\}$ and assuming that one has a priori distribution $\pi(\boldsymbol{\theta})$
  - compute the posterior
$$\mathbb{P}(\boldsymbol{\theta} \mid \mathcal{D})$$
  - choose the most probable $\boldsymbol{\theta}$ after observing data
$$\boldsymbol{\theta}^* = \arg\max_{\boldsymbol{\theta}} \mathbb{P}(\boldsymbol{\theta} \mid \mathcal{D})$$
- In practice, one rather optimizes $\log \mathbb{P}(\mathcal{D} \mid \boldsymbol{\theta}) + \log \pi(\boldsymbol{\theta})$. Why?

Maximum A Posteriori

- Assumptions: Data is generated by a parametric model $\mathbb{P}(\boldsymbol{X} \mid \boldsymbol{\theta})$ and we have an a priori distribution over parameters $\boldsymbol{\theta}$
- General MAP Principle: Given i.i.d. training data $\mathcal{D} = \{\boldsymbol{x}^1, \ldots, \boldsymbol{x}^N\}$ and assuming that one has a priori distribution $\pi(\boldsymbol{\theta})$
  - compute the posterior

$$\mathbb{P}(\boldsymbol{\theta} \mid \mathcal{D})$$

  - choose the most probable $\boldsymbol{\theta}$ after observing data

$$\boldsymbol{\theta}^* = \arg\max_{\boldsymbol{\theta}} \mathbb{P}(\boldsymbol{\theta} \mid \mathcal{D})$$

- In practice, one rather optimizes $\log \mathbb{P}(\mathcal{D} \mid \boldsymbol{\theta}) + \log \pi(\boldsymbol{\theta})$. Why?

$$\begin{aligned}
\boldsymbol{\theta}^* &= \arg\max_{\boldsymbol{\theta}} \mathbb{P}(\boldsymbol{\theta} \mid \mathcal{D}) \\
&= \arg\max_{\boldsymbol{\theta}} \frac{\mathbb{P}(\mathcal{D} \mid \boldsymbol{\theta})\pi(\boldsymbol{\theta})}{\mathbb{P}(\mathcal{D})} \\
&= \arg\max_{\boldsymbol{\theta}} \log \mathbb{P}(\mathcal{D} \mid \boldsymbol{\theta}) + \log \pi(\boldsymbol{\theta})
\end{aligned}$$

MAP Application to Previous Example

	1	2	3	4	5
X_1	A	A	B	C	C
X_2	T	F	T	F	F
X_3	T	T	F	F	T
X_4	0	1	1	2	0
X_5	F	F	T	T	F

- Application: learn $\mathbb{P}(X_1)$ in previous example
  - Assume the prior is Dirichlet$(1, 1, 1)$. What does it mean?
  - What is the function to be maximized?

MAP Application to Previous Example

	1	2	3	4	5
X_1	A	A	B	C	C
X_2	T	F	T	F	F
X_3	T	T	F	F	T
X_4	0	1	1	2	0
X_5	F	F	T	T	F

- Application: learn $\mathbb{P}(X_1)$ in previous example
  - Assume the prior is Dirichlet$(1, 1, 1)$. What does it mean?
  - What is the function to be maximized?
- Application: learn $\mathbb{P}(X_3 \mid X_5)$ in previous example
  - Assume the prior is Beta$(5, 1)$ for $X_5 = T$ and it is Beta$(1, 4)$ for $X_5 = F$. What does it mean?
  - What is the function to be maximized?

1. How to Build a Bayesian Network?
- Introduction
- Parameter Learning
- Structure Learning

Structure Learning

- Problem formulation: $\max_{G \in \mathcal{G}} f(G, \mathcal{D})$ where $\mathcal{G}$= all DAGs with n nodes, $\mathcal{D} = \{\boldsymbol{x}^1, \ldots, \boldsymbol{x}^N\}$= i.i.d. training data

Structure Learning

- Problem formulation: $\max_{G \in \mathcal{G}} f(G, \mathcal{D})$ where $\mathcal{G}$= all DAGs with n nodes, $\mathcal{D} = \{\boldsymbol{x}^1, \ldots, \boldsymbol{x}^N\}$= i.i.d. training data
- Learning the ***structure*** of Bayes net is NP-hard
  - Size of space of DAG is super-exponential in number of variables
  - Comparatively, learning the CPTs is easy

Structure Learning

- Problem formulation: $\max_{G \in \mathcal{G}} f(G, \mathcal{D})$ where $\mathcal{G}=$ all DAGs with n nodes, $\mathcal{D} = \{\boldsymbol{x}^1, \ldots, \boldsymbol{x}^N\}=$ i.i.d. training data
- Learning the ***structure*** of Bayes net is NP-hard
  - Size of space of DAG is super-exponential in number of variables
  - Comparatively, learning the CPTs is easy
- Two families of methods for structure learning
  - Methods based on satisfying conditional independences
    - Idea: Statistical test (conditional) independences
    - deterministic given data
    - relatively fast (when limiting the number of tests)
    - clear stopping criteria (e.g. threshold for statistical independence tests)
    - initial errors snowball
  - Bayesian methods based on maximizing a score (BIC, MDL, BD...)
    - Idea: From prior on graph structures, update belief with data
    - handle missing data
    - may get stuck in local optimum
    - solution depends on initial conditions

Heuristic Method: Maximum Weight Spanning Tree

- Maximum Weight Spanning Tree: Given a connected valued graph, find a spanning tree (i.e., connected acyclic subgraph) with maximum weight
  - Polynomial problem
  - Kruskal algorithm or Prim algorithm

- Principle of heuristic: Define a score (e.g., mutual information) for each pair of variables, compute MWST, choose root and orient edges

$$I(X, Y) = D(\hat{\mathbb{P}}(X, Y) || \hat{\mathbb{P}}(X)\hat{\mathbb{P}}(Y))$$

Heuristic Method: K2 Algorithm

- Problem simplification: $\max_{G \in \mathcal{G}'} f(G, \mathcal{D})$ where $\mathcal{G}'$=DAGs satisfying a fixed topological order (wlog $X_1, X_2, \ldots, X_n$)
- Principle of heuristic: For $i = 2, \ldots, n$, choose parents of X_i as $\arg\max_{I \subseteq \{1,\ldots,i-1\}, |I|=K}$ score$(X_i, X_I, \mathcal{D})$ in a greedy way

---

K2$\big((X_1, \ldots, X_n),\ K,\ \mathcal{D}\big)$
for $i = 1$ *to* n **do**
    parents[i] $\leftarrow \{\ \}$; $s \leftarrow$ score(X_i,parents[i], $\mathcal{D}$); continue $\leftarrow$ true
    while *continue and* $|$*parents*[i]$| < K$ **do**
        $j^* \leftarrow \arg\max_j$ score(X_i,parents[i] $\cup \{X_j\}, \mathcal{D}$)
        sNew $\leftarrow$ score(X_i,parents[i] $\cup \{X_{j^*}\}, \mathcal{D}$)
        if *sNew* $> s$ **then**
            $s \leftarrow$sNew; parents[i] $\leftarrow$ parents[i] $\cup\{X_{j^*}\}$
        else continue $\leftarrow$ false ;
return parents

Score Function of Original K2 Algorithm

- Original K2 algorithm considers $\mathbb{P}(B \mid \mathcal{D})$ where B is a Bayes net
- Under four assumptions, $\mathbb{P}(B \mid \mathcal{D})$ can be decomposed into a tractable score function
  - All variables are discrete
  - The data points of $\mathcal{D}$ are independently generated given a BN
  - No missing data
  - The distributions over CPTs are uniform given a BN
- $\text{score}(X, X_I, \mathcal{D}) = \prod_{j=1}^{q_I} \frac{(r-1)!}{(N_j + r - 1)!} \prod_{k=1}^{r} N_{jk}!$
  where r is the number of possible values of X, q_I the number of instantiations of X_I, N_{jk} the number of occurrences of k-th value of X_i and j-th value of X_I in $\mathcal{D}$, and $N_j = \sum_{k=1}^{r} N_{jk}$

Example

	1	2	3	4	5	6	7	8	9	10
X_1	1	1	0	1	0	0	1	0	1	0
X_2	0	1	0	1	0	1	1	0	1	0
X_3	0	1	1	1	0	1	1	0	1	0

- Assume the order is X_1, X_2, X_3
- Take $K = 2$
- Run the K2 algorithm.
- Which structure does it return?

Other Score Function: Bayesian Information Criterion

- Issue: Using the likelihood favors Bayes nets with too many parents
- Idea: Use a score function that penalizes such solutions:

$$BIC(B, \mathcal{D}) = 2\log(\mathbb{P}(\mathcal{D} \mid B, \hat{\theta}_B)) - n_B \log(N)$$

  where $\hat{\theta}_B$ is the ML estimator of the parameters of B, n_B is the dimension of B, which is its number of free parameters.
- Remark: Bayesian Information Criterion is usually defined with the opposite sign
- The computation of this score can be decomposed over nodes:

$$bic(X, parents(X), \mathcal{D}) = 2\log(\mathbb{P}(\mathcal{D}_{X,parents(X)} \mid B, \hat{\theta}_{X,B})) - n_{X,B} \log(N)$$

  where $\mathcal{D}_{X,parents(X)}$ is the data restricted to X and its parents, $\hat{\theta}_{X,B}$ is the ML estimator of the parameters for node X, and $n_{X,B}$ is the number of free parameters for node X.