Learning graphical models

Parameter estimation

- We assume the structure of the model is given
- We are given a dataset of examples \( D = \{ x(1), \ldots, x(N) \} \)
- Each example \( x(i) \) is a configuration for all (complete data) or some (incomplete data) variables in the model
- We need to estimate the parameters of the model (conditional probability distributions) from the data
- The simplest approach consists of learning the parameters maximizing the likelihood of the data:

\[
\theta^{\text{max}} = \arg \max_{\theta} p(D|\theta) = \arg \max_{\theta} \mathcal{L}(D, \theta)
\]

Learning Bayesian Networks

Maximum likelihood estimation, complete data

\[
p(D|\theta) = \prod_{i=1}^{N} p(x(i)|\theta) \quad \text{examples independent given } \theta
\]

\[
= \prod_{i=1}^{N} \prod_{j=1}^{m} p(x_j(i)|\text{pa}_j(i), \theta)
\quad \text{factorization for BN}
\]
Maximum likelihood estimation, complete data

$$p(D|\theta) = \prod_{i=1}^{N} \prod_{j=1}^{m} p(x_j(i)|pa_j(i), \theta)$$

factorization for BN

$$= \prod_{i=1}^{N} \prod_{j=1}^{m} p(x_j(i)|pa_j(i), \theta_{X_j|Pa})$$

disjoint CPD parameters

Learning graphical models

Maximum likelihood estimation, complete data

• The parameters of each CPD can be estimated independently:

$$\theta_{X_j|Pa}^{\text{max}} = \text{argmax}_{\theta_{X_j|Pa}} \prod_{i=1}^{N} p(x_j(i)|pa_j(i), \theta_{X_j|Pa})$$

$$\mathcal{L}(\theta_{X_j|Pa}, D)$$

• A discrete CPD $P(X|U)$, can be represented as a table, with:
  - a number of rows equal to the number $\text{Val}(X)$ of configurations for $X$
  - a number of columns equal to the number $\text{Val}(U)$ of configurations for its parents $U$
  - each table entry $\theta_{x|u}$ indicating the probability of a specific configuration of $X = x$ and its parents $U = u$

Learning graphical models

Maximum likelihood estimation, complete data

• Replacing $p(x(i)|pa(i))$ with $\theta_{x(i)|u(i)}$, the local likelihood of a single CPD becomes:
\[ \mathcal{L}(\theta_{X|\text{Pa}}, D) = \prod_{i=1}^{N} p(x(i)|\text{pa}(i), \theta_{X|\text{Pa}}) \]
\[ = \prod_{i=1}^{N} \theta_{x(i)|u(i)} \]
\[ = \prod_{u \in \text{Val}(U)} \left[ \prod_{x \in \text{Val}(X)} \theta_{x|u}^{N_{u,x}} \right] \]

where \( N_{u,x} \) is the number of times the specific configuration \( X = x, U = u \) was found in the data

**Learning graphical models**

**Maximum likelihood estimation, complete data**

- A column in the CPD table contains a multinomial distribution over values of \( X \) for a certain configuration of the parents \( U \)
- Thus each column should sum to one: \( \sum_{x} \theta_{x|u} = 1 \)
- Parameters of different columns can be estimated independently
- For each multinomial distribution, zeroing the gradient of the maximum likelihood and considering the normalization constraint, we obtain:

\[ \theta_{x|u}^{\text{max}} = \frac{N_{u,x}}{\sum_{x} N_{u,x}} \]

- The maximum likelihood parameters are simply the fraction of times in which the specific configuration was observed in the data

**Learning graphical models**

**Adding priors**

- ML estimation tends to overfit the training set
- Configuration not appearing in the training set will receive zero probability
- A common approach consists of combining ML with a prior probability on the parameters, achieving a maximum-a-posteriori estimate:

\[ \theta^{\text{max}} = \arg\max_{\theta} p(D|\theta)p(\theta) \]
Learning graphical models

Dirichlet priors

• The conjugate (read natural) prior for a multinomial distribution is a Dirichlet distribution with parameters $\alpha_{x|u}$ for each possible value of $x$

• The resulting maximum-a-posteriori estimate is:

$$
\theta_{x|u}^{\max} = \frac{N_{u,x} + \alpha_{x|u}}{\sum_x (N_{u,x} + \alpha_{x|u})}
$$

• The prior is like having observed $\alpha_{x|u}$ imaginary samples with configuration $X = x, U = u$

Learning graphical models

Incomplete data

• With incomplete data, some of the examples miss evidence on some of the variables

• Counts of occurrences of different configurations cannot be computed if not all data are observed

• The full Bayesian approach of integrating over missing variables is often intractable in practice

• We need approximate methods to deal with the problem

Learning with missing data: Expectation-Maximization

E-M for Bayesian nets in a nutshell

• Sufficient statistics (counts) cannot be computed (missing data)

• Fill-in missing data inferring them using current parameters (solve inference problem to get expected counts)

• Compute parameters maximizing likelihood (or posterior) of such expected counts

• Iterate the procedure to improve quality of parameters

Learning with missing data: Expectation-Maximization

Expectation-Maximization algorithm

**e-step** Compute the expected sufficient statistics for the complete dataset, with expectation taken wrt the joint distribution for $X$ conditioned of the current value of $\theta$ and the known data $D$:

$$
E_{p(x|D,\theta)}[N_{ijk}] = \sum_{l=1}^{n} p(X_i(l) = x_k, Pa_i(l) = pa_j|X_i, \theta)
$$

• If $X_i(l)$ and $Pa_i(l)$ are observed for $X_i$, it is either zero or one

• Otherwise, run Bayesian inference to compute probabilities from observed variables
Learning with missing data: Expectation-Maximization

Expectation-Maximization algorithm

m-step compute parameters maximizing likelihood of the complete dataset $D_c$ (using expected counts):

$$\theta^* = \arg\max_{\theta} p(D_c|\theta)$$

which for each multinomial parameter evaluates to:

$$\theta^*_{ijk} = \frac{E_p(\mathbf{x}|\mathbf{D}, \theta)[N_{ijk}]}{\sum_{k=1}^{r_i} E_p(\mathbf{x}|\mathbf{D}, \theta)[N_{ijk}]}$$

Note

ML estimation can be replaced by maximum a-posteriori (MAP) estimation giving:

$$\theta^*_{ijk} = \frac{\alpha_{ijk} + E_{p(\mathbf{x}|\mathbf{D}, S, \theta)}[N_{ijk}]}{\sum_{k=1}^{r_i} (\alpha_{ijk} + E_{p(\mathbf{x}|\mathbf{D}, S, \theta)}[N_{ijk}])}$$

Learning structure of graphical models

Approaches

constraint-based test conditional independencies on the data and construct a model satisfying them

score-based assign a score to each possible structure, define a search procedure looking for the structure maximizing the score

model-averaging assign a prior probability to each structure, and average prediction over all possible structures weighted by their probabilities (full Bayesian, intractable)

Appendix: Learning the structure

Bayesian approach

• Let $S$ be the space of possible structures (DAGS) for the domain $X$.
• Let $D$ be a dataset of observations
• Predictions for a new instance are computed marginalizing over both structures and parameters:

$$p(X_{N+1}|D) = \sum_{S \in S} \int_{\theta} P(X_{N+1}, S, \theta|D) d\theta$$

$$= \sum_{S \in S} \int_{\theta} P(X_{N+1}|S, \theta, D) P(S, \theta|D) d\theta$$

$$= \sum_{S \in S} \int_{\theta} P(X_{N+1}|S, \theta) P(\theta|S, D) P(S|D) d\theta$$

$$= \sum_{S \in S} P(S|D) \int_{\theta} P(X_{N+1}|S, \theta) P(\theta|S, D) d\theta$$
Learning the structure

Problem
Averaging over all possible structures is too expensive

Model selection

• Choose a best structure \( S^* \) and assume \( P(S^*|D) = 1 \)
• Approaches:
  – Score-based:
    * Assign a score to each structure
    * Choose \( S^* \) to maximize the score
  – Constraint-based:
    * Test conditional independencies on data
    * Choose \( S^* \) that satisfies these independencies

Score-based model selection

Structure scores

• Maximum-likelihood score:
  \[
  S^* = \arg\max_{S \in S} p(D|S)
  \]
• Maximum-a-posteriori score:
  \[
  S^* = \arg\max_{S \in S} p(D|S)p(S)
  \]

Computing \( P(D|S) \)

Maximum likelihood approximation

• The easiest solution is to approximate \( P(D|S) \) with the maximum-likelihood score over the parameters:
  \[
  P(D|S) \approx \max_{\theta} P(D|S, \theta)
  \]
• Unfortunately, this boils down to adding a connection between two variables if their empirical mutual information over the training set is non-zero (proof omitted)
• Because of noise, empirical mutual information between any two variables is almost never exactly zero ⇒ fully connected network

Computing \( P(D|S) \equiv P_S(D) \): Bayesian-Dirichlet scoring

Simple case: setting

• \( X \) is a single variable with \( r \) possible realizations (\( r \)-faced die)
• \( S \) is a single node
• Probability distribution is a multinomial with Dirichlet priors \( \alpha_1, \ldots, \alpha_r \).
• \( D \) is a sequence of \( N \) realizations (die tosses)
Computing \( P_S(D) \): Bayesian-Dirichlet scoring

Simple case: approach

- Sort \( D \) according to outcome:
  \[
  D = \{x^1_1, x^1_1, \ldots, x^1_1, x^2_1, \ldots, x^r_1, x^2_2, \ldots, x^r_r\}
  \]
- Its probability can be decomposed as:
  \[
  P_S(D) = \prod_{t=1}^{N} P_S(X(t) | X(t-1), \ldots, X(1))_{D(t-1)}
  \]
- The prediction for a new event given the past is:
  \[
  P_S(X(t+1) = x_k | D(t)) = \mathbb{E}_{p_S(\theta | D(t))}[\theta_k] = \frac{\alpha_k + N_k(t)}{\alpha + t}
  \]
  where \( N_k(t) \) is the number of times we have \( X = x_k \) in the first \( t \) examples in \( D \)

Computing \( P_S(D) \): Bayesian-Dirichlet scoring

Simple case: approach

\[
  P_S(D) = \frac{\alpha_1 \alpha_1 + 1}{\alpha \alpha + 1} \cdots \frac{\alpha_1 + N_1 - 1}{\alpha + N_1 - 1} \\
  \cdot \frac{\alpha_2 \alpha_2 + 1}{\alpha + N_2} \frac{\alpha_2 + N_2 - 1}{\alpha + N_2 - 1} \\
  \cdot \frac{\alpha_r \alpha_r + 1}{\alpha + N_r} \cdots \frac{\alpha_r + N_r - 1}{\alpha + N - 1}
  = \frac{\Gamma(\alpha)}{\Gamma(\alpha + N)} \prod_{k=1}^{\alpha} \frac{\Gamma(\alpha_k + N_k)}{\alpha_k}
\]
  where we used the Gamma function \( \Gamma(x + 1) = x\Gamma(x) \):
  \[
  \alpha(1 + \alpha) \ldots (N - 1 + \alpha) = \frac{\Gamma(N + \alpha)}{\Gamma(\alpha)}
  \]

Computing \( P_S(D) \): Bayesian-Dirichlet scoring

General case

\[
  P_S(D) = \prod_{i} \prod_{j} \frac{\Gamma(\alpha_{ij})}{\Gamma(\alpha_{ij} + N_{ij})} \prod_{k=1}^{r} \frac{\Gamma(\alpha_{ijk} + N_{ijk})}{\alpha_{ijk}}
\]

where
- \( i \in \{1, \ldots, n\} \) ranges over nodes in the network
- \( j \in \{1, q_i\} \) ranges over configurations of \( X_i \)'s parents
- \( k \in \{1, r_i\} \) ranges over states of \( X_i \)

Note
- The score is decomposable: it is the product of independent scores associated with the distribution of each node in the net
Search strategy

Approach

• Discrete search problem: NP-hard for nets whose nodes have at most $k > 1$ parents.

• Heuristic search strategies employed:
  
  – Search space: set of DAGs
  – Operators: add, remove, reverse one arc
  – Initial structure: e.g. random, fully disconnected, ...
  – Strategies: hill climbing, best first, simulated annealing

Note

Decomposable scores allow to recompute local scores only for a single move