School of Computer Science
Carnegie Mellon

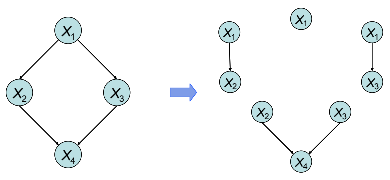
Probabilistic Graphical Models

Parameter Est. in fully observed BNs

Eric Xing

Lecture 7, February 6, 2017

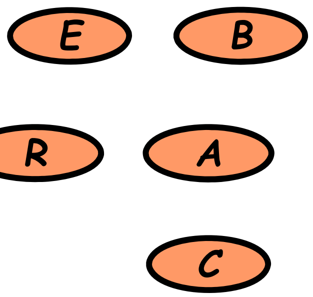
Reading: KF-chap 17



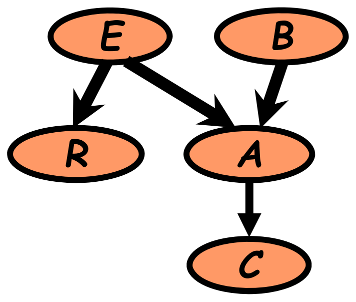
Learning Graphical Models

The goal:

Given set of independent samples (***assignments*** of random variables), find the ***best*** (the most likely?) Bayesian Network (both DAG and CPDs)



(B,E,A,C,R)=(T,F,F,T,F)
(B,E,A,C,R)=(T,F,T,T,F)
....
(B,E,A,C,R)=(F,T,T,T,F)



Structural learning

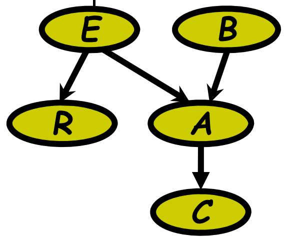
E	B	P(A | E,B)	
e	$\underline{b}$	0.9	0.1
$\underline{e}$	b	0.2	0.8
$\underline{e}$	$\underline{b}$	0.9	0.1
e	b	0.01	0.99

Parameter learning

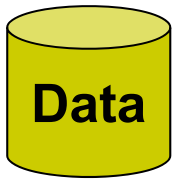
Learning Graphical Models

- Scenarios:
  - completely observed GMs
    - directed
    - undirected
  - partially or unobserved GMs
    - directed
    - undirected (an open research topic)
- Estimation principles:
  - Maximal likelihood estimation (MLE)
  - Bayesian estimation
  - Maximal conditional likelihood
  - Maximal "Margin"
  - Maximum entropy

- We use **learning** as a name for the process of estimating the parameters, and in some cases, the topology of the network, from data.



ML Structural Learning for completely observed GMs



$$\begin{array}{c}(x_1^{(1)},\dots,x_n^{(1)})\\(x_1^{(2)},\dots,x_n^{(2)})\\\dots\\(x_1^{(M)},\dots,x_n^{(M)})\end{array}$$

Two "Optimal" approaches

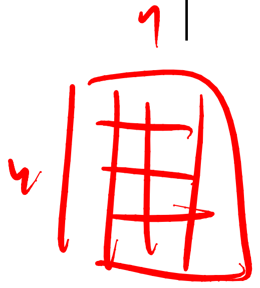
- "Optimal" here means the employed algorithms guarantee to return a structure that maximizes the objectives (e.g., LogLik)
  - Many heuristics used to be popular, but they provide no guarantee on attaining optimality, interpretability, or even do not have an explicit objective
  - E.g.: structured EM, Module network, greedy structural search, etc.

- We will learn two classes of algorithms for guaranteed structure learning, which are likely to be the only known methods enjoying such guarantee, but they only apply to certain families of graphs:
  - Trees: The Chow-Liu algorithm (this lecture)
  - Pairwise MRFs: covariance selection, neighborhood-selection (later)

Structural Search

- How many graphs over *n* nodes? $O(2^{n^2})$

2^{n^2}



- How many trees over *n* nodes? $O(n!)$

$O(n!)$

- But it turns out that we can find exact solution of an optimal tree (under MLE)!
  - Trick: MLE score decomposable to edge-related elements
  - Trick: in a tree each node has only one parent!
  - Chow-liu algorithm

Information Theoretic Interpretation of ML

$$\ell(\theta_G, G; D) = \log p(D \mid \theta_G, G)$$

$$= \log \prod_n \left(\prod_i p(x_{n,i} \mid \mathbf{x}_{n,\pi_i(G)}, \theta_{i|\pi_i(G)}) \right)$$

$$= \sum_i \left(\sum_n \log p(x_{n,i} \mid \mathbf{x}_{n,\pi_i(G)}, \theta_{i|\pi_i(G)}) \right)$$

$$= M \sum_i \left(\sum_{x_i, \mathbf{x}_{\pi_i(G)}} \frac{count(x_i, \mathbf{x}_{\pi_i(G)})}{M} \log p(x_i \mid \mathbf{x}_{\pi_i(G)}, \theta_{i|\pi_i(G)}) \right)$$

$$= M \sum_i \left(\sum_{x_i, \mathbf{x}_{\pi_i(G)}} \hat{p}(x_i, \mathbf{x}_{\pi_i(G)}) \log p(x_i \mid \mathbf{x}_{\pi_i(G)}, \theta_{i|\pi_i(G)}) \right)$$

From sum over data points to sum over count of variable states

Information Theoretic Interpretation of ML (con'd)

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

Decomposable score and a function of the graph structure

Chow-Liu tree learning algorithm

- Objection function:

$$\ell(\theta_G, G; D) = \log \hat{p}(D \mid \theta_G, G)$$
$$= M \sum_i \hat{I}(x_i, \mathbf{x}_{\pi_i(G)}) - M \sum_i \hat{H}(x_i) \quad \Rightarrow \quad \boxed{C(G) = M \sum_i \hat{I}(x_i, \mathbf{x}_{\pi_i(G)})}$$

x_f

[illegible]

- Chow-Liu:
  - For each pair of variable x_i and x_j
    - Compute empirical distribution: $\hat{p}(X_i, X_j) = \frac{count(x_i, x_j)}{M}$
    - Compute mutual information: $\hat{I}(X_i, X_j) = \sum_{x_i, x_j} \hat{p}(x_i, x_j) \log \frac{\hat{p}(x_i, x_j)}{\hat{p}(x_i)\hat{p}(x_j)}$
  - Define a graph with node $x_1, \ldots, x_n$
    - Edge (I,j) gets weight $\hat{I}(X_i, X_j)$

Chow-Liu algorithm (con'd)

- Objection function:

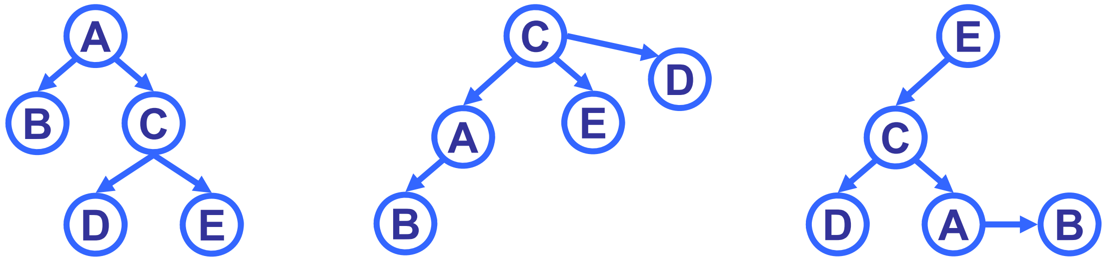
$$\ell(\theta_G, G; D) = \log \hat{p}(D \mid \theta_G, G) = M \sum_i \hat{I}(x_i, \mathbf{x}_{\pi_i(G)}) - M \sum_i \hat{H}(x_i) \Rightarrow \boxed{C(G) = M \sum_i \hat{I}(x_i, \mathbf{x}_{\pi_i(G)})}$$

- Chow-Liu:

  Optimal tree BN

  - Compute maximum weight spanning tree
  - Direction in BN: pick any node as root, do breadth-first-search to define directions
  - I-equivalence:

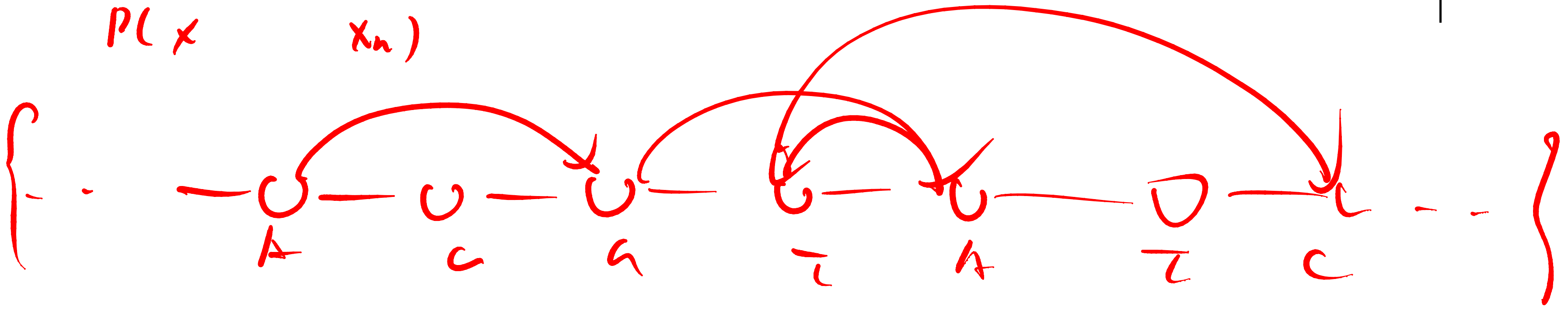
$I_G(x \quad x_y)$



$$C(G) = I(A,B) + I(A,C) + I(C,D) + I(C,E)$$

Tree and mixture-of-tree models for DNA sequence classification

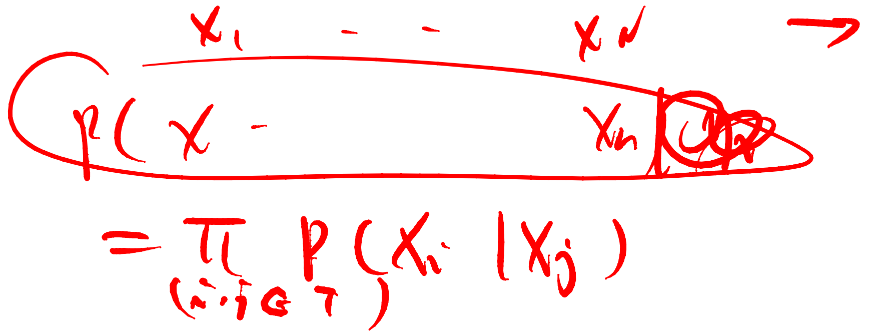
$P(x \quad x_n)$



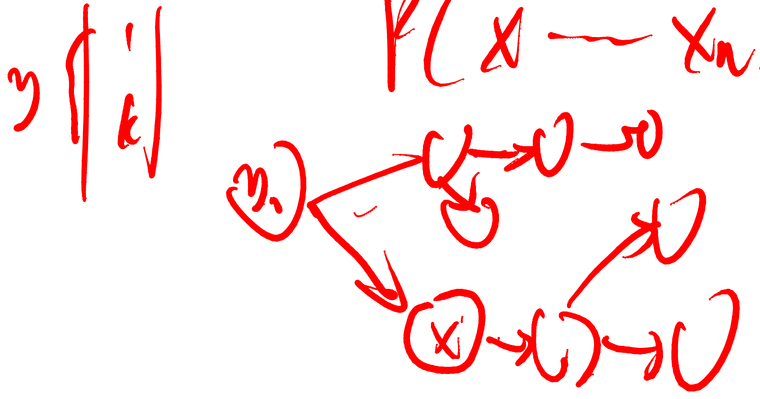
$I(x_i - x_j)$

$\Rightarrow I(x_i - x_j \mid y=k)$

$P(x \;—\; x_n \mid y=k)$

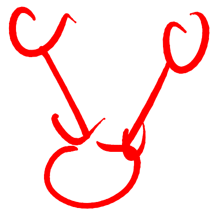


$= \prod_{(i,j \in T)} P(x_i \mid x_j)$

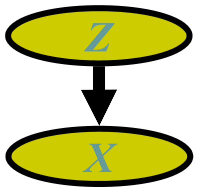


Structure Learning for general graphs

- Theorem:
  - The problem of learning a BN structure with at most *d* parents is NP-hard for any (fixed) *d*≥2



- Most structure learning approaches use heuristics
  - Exploit score decomposition
  - Two heuristics that exploit decomposition in different ways
    - Greedy search through space of node-orders
    - Local search of graph structures



ML Parameter Est. for completely observed GMs of given structure

- The data:

  $\{ (z_1,x_1), (z_2,x_2), (z_3,x_3), \dots (z_N,x_N)\}$

Parameter Learning

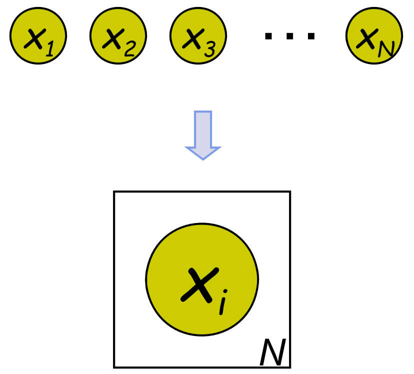
- Assume G is known and fixed,
  - from expert design
  - from an intermediate outcome of iterative structure learning
- Goal: estimate from a dataset of N independent, identically distributed (*iid*) training cases $D = \{x_1, \ldots, x_N\}$.
- In general, each training case $\mathbf{x}_n=(x_{n,1}, \ldots, x_{n,M})$ is a vector of M values, one per node,
  - the model can be completely observable, i.e., every element in x_n is known (no missing values, no hidden variables),
  - or, partially observable, i.e., $\exists i$, s.t. $x_{n,i}$ is not observed.
- **In this lecture we consider learning parameters for a BN with given structure and is completely observable**

$$\ell(\theta; D) = \log p(D \mid \theta) = \log \prod_n \left(\prod_i p(x_{n,i} \mid \mathbf{x}_{n,\pi_i}, \theta_i) \right) = \sum_i \left(\sum_n \log p(x_{n,i} \mid \mathbf{x}_{n,\pi_i}, \theta_i) \right)$$

Review of density estimation

- Can be viewed as single-node graphical models
- Instances of exponential family dist.
- Building blocks of general GM
- MLE and Bayesian estimate

GM:



$P(X = \begin{bmatrix} x_1 \\ \vdots \\ x_6 \end{bmatrix}) = \prod_{i=1}^{6} \theta_i^{x_i}$

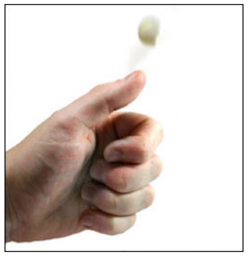
$= \theta_6$

$x = \begin{bmatrix} 0 \\ 0 \\ 0 \\ 0 \\ 0 \\ 1 \end{bmatrix}$

Discrete Distributions

- Bernoulli distribution: Ber(p)

$$P(x) = \begin{cases} 1-p & \text{for } x = 0 \\ p & \text{for } x = 1 \end{cases} \Rightarrow P(x) = p^x (1-p)^{1-x}$$



- Multinomial distribution: Mult($1, \theta$)
  - Multinomial (indicator) variable:

$p($ face $= 6)$

$p(x = 6)$

$$X = \begin{bmatrix} X_1 \\ X_2 \\ X_3 \\ X_4 \\ X_5 \\ X_6 \end{bmatrix}, \quad \text{where} \quad X_j = [0,1], \text{ and } \sum_{j \in [1,\dots,6]} X_j = 1$$

$$X_j = 1 \text{ w.p. } \theta_j, \quad \sum_{j \in [1,\dots,6]} \theta_j = 1 .$$

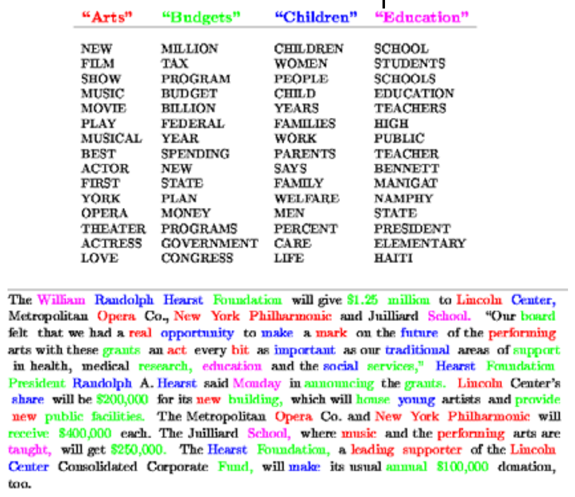
$$p(x(j)) = P(\{X_j = 1, \text{where } j \text{ index the dice-face}\})$$
$$= \theta_j = \theta_A^{x_A} \times \theta_C^{x_C} \times \theta_G^{x_G} \times \theta_T^{x_T} = \prod_k \theta_k^{x_k} = \theta^x$$

Discrete Distributions

- Multinomial distribution: Mult(n, θ)
  - Count variable:

$$n = \begin{bmatrix} n_1 \\ \vdots \\ n_K \end{bmatrix}, \qquad \text{where } \sum_j n_j = N$$

$$p(n) = \frac{N!}{n_1! n_2! \cdots n_K!} \theta_1^{n_1} \theta_2^{n_2} \cdots \theta_K^{n_K} = \frac{N!}{n_1! n_2! \cdots n_K!} \theta^n$$



"Arts"	"Budgets"	"Children"	"Education"
NEW	MILLION	CHILDREN	SCHOOL
FILM	TAX	WOMEN	STUDENTS
SHOW	PROGRAM	PEOPLE	SCHOOLS
MUSIC	BUDGET	CHILD	EDUCATION
MOVIE	BILLION	YEARS	TEACHERS
PLAY	FEDERAL	FAMILIES	HIGH
MUSICAL	YEAR	WORK	PUBLIC
BEST	SPENDING	PARENTS	TEACHER
ACTOR	NEW	SAYS	BENNETT
FIRST	STATE	FAMILY	MANIGAT
YORK	PLAN	WELFARE	NAMPHY
OPERA	MONEY	MEN	STATE
THEATER	PROGRAMS	PERCENT	PRESIDENT
ACTRESS	GOVERNMENT	CARE	ELEMENTARY
LOVE	CONGRESS	LIFE	HAITI

The William Randolph Hearst Foundation will give $1.25 million to Lincoln Center, Metropolitan Opera Co., New York Philharmonic and Juilliard School. "Our board felt that we had a real opportunity to make a mark on the future of the performing arts with these grants an act every bit as important as our traditional areas of support in health, medical research, education and the social services," Hearst Foundation President Randolph A. Hearst said Monday in announcing the grants. Lincoln Center's share will be $200,000 for its new building, which will house young artists and provide new public facilities. The Metropolitan Opera Co. and New York Philharmonic will receive $400,000 each. The Juilliard School, where music and the performing arts are taught, will get $250,000. The Hearst Foundation, a leading supporter of the Lincoln Center Consolidated Corporate Fund, will make its usual annual $100,000 donation, too.

Example: multinomial model

- Data:
  - We observed N ***iid*** die rolls (K-sided): D={5, 1, K, …, 3}
- Representation:

Unit basis vectors: $x_n = \begin{pmatrix} x_{n,1} \\ x_{n,2} \\ \vdots \\ x_{n,K} \end{pmatrix}$, where $x_{n,k} = \{0,1\}$, and $\sum_{k=1}^{K} x_{n,k} = 1$

- Model:

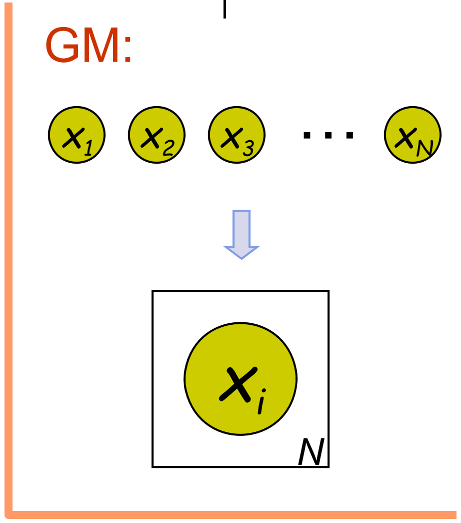
$$X_{n,k} = 1 \text{ w.p. } \theta_k \text{, and } \sum_{k\in\{1,\dots K\}} \theta_k = 1$$

- How to write the likelihood of a single observation x_n?

$$P(x_i) = P(\{x_{n,k} = 1, \text{where } k \text{ index the die-side of the } n\text{th roll}\})$$
$$= \theta_k = \theta_1^{x_{n,1}} \times \theta_2^{x_{n,2}} \times \cdots \times \theta_K^{x_{n,K}} = \prod_{k=1}^{K} \theta_k^{x_{n,k}}$$

- The likelihood of dataset $D=\{x_1, \dots, x_N\}$:

$$P(x_1, x_2, \dots, x_N \mid \theta) = \prod_{n=1}^{N} P(x_n \mid \theta) = \prod_{n=1}^{N}\left(\prod_k \theta_k^{x_{n,k}}\right) = \prod_k \theta_k^{\sum_{n=1}^{N} x_{n,k}} = \prod_k \theta_k^{n_k}$$



MLE: constrained optimization with Lagrange multipliers

- Objective function:

$$\ell(\theta;D) = \underline{\log P(D \mid \theta)} = \log \prod_k \theta_k^{n_k} = \sum_k n_k \log \theta_k$$

- We need to maximize this subject to the constrain $\underline{\sum_{k=1}^{K} \theta_k = 1}$
- Constrained cost function with a Lagrange multiplier

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

- Take derivatives wrt θ_k

$$\frac{\partial \bar{\ell}}{\partial \theta_k} = \frac{n_k}{\theta_k} - \lambda = 0$$

$$n_k = \lambda \theta_k \Rightarrow \sum_k n_k = N = \lambda \sum_k \theta_k = \lambda$$

$$\Longrightarrow \underline{\hat{\theta}_{k,MLE}} = \frac{n_k}{N} \quad \text{or} \quad \hat{\theta}_{k,MLE} = \frac{1}{N} \sum_n x_{n,k}$$

Frequency as sample mean

- Sufficient statistics
  - The counts, $\vec{n} = (n_1, \cdots, n_K)$, $n_k = \sum_n x_{n,k}$, are **sufficient statistics** of data D

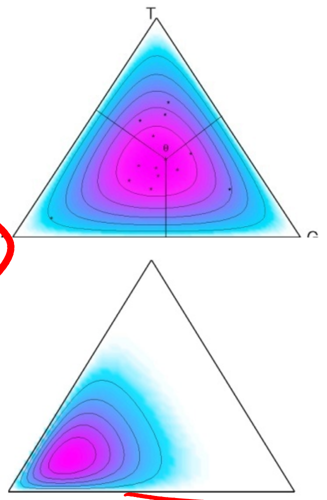
Bayesian estimation:

$\alpha_1 = 1 \quad \alpha_2 = 1$

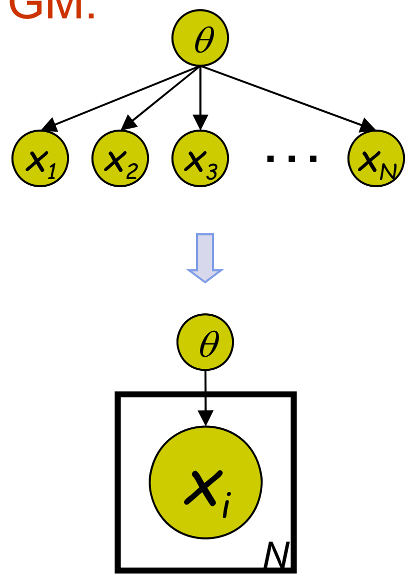
$\alpha_1 = 100 \quad \alpha_2 = 100$

- Dirichlet distribution:

$$P(\theta) = \frac{\Gamma(\sum_k \alpha_k)}{\prod_k \Gamma(\alpha_k)} \prod_k \theta_k^{\alpha_k - 1} = C(\alpha) \prod_k \theta_k^{\alpha_k - 1}$$



GM:



- Posterior distribution of θ:

$$P(\theta \mid x_1, \ldots, x_N) = \frac{p(x_1, \ldots, x_N \mid \theta) p(\theta)}{p(x_1, \ldots, x_N)} \propto \prod_k \theta_k^{n_k} \prod_k \theta_k^{\alpha_k - 1} = \prod_k \theta_k^{\alpha_k + n_k - 1}$$

  - Notice the isomorphism of the posterior to the prior,
  - such a prior is called a **conjugate prior**

$\theta_k^{ML} = \frac{n_k}{N}$

2 → 3 ~ 102

8   9   108

10   210

Dirichlet parameters can be understood as pseudo-counts

- Posterior mean estimation:

$$\theta_k^{?} = \int \theta_k p(\theta \mid D) d\theta = C \int \theta_k \prod_k \theta_k^{\alpha_k + n_k - 1} d\theta = \frac{n_k + \alpha_k}{N + |\alpha|}$$

More on Dirichlet Prior:

- Where is the normalize constant $C(\alpha)$ come from?

$$\frac{1}{C(\alpha)} = \int \cdots \int \theta_1^{\alpha_1 - 1} \cdots \theta_K^{\alpha_K - 1} d\theta_1 \cdots d\theta_K = \frac{\prod_k \Gamma(\alpha_k)}{\Gamma\left(\sum_k \alpha_k\right)}$$

  - Integration by parts
  - $\Gamma(\alpha)$ is the gamma function: $\Gamma(\alpha) = \int_0^\infty t^{\alpha-1} e^{-t} dt$
  - For inregers, $\Gamma(n+1) = n!$
- Marginal likelihood:

$$p(\{x_1, \ldots, x_N\} \mid \vec{\alpha}) = p(\vec{n} \mid \vec{\alpha}) = \int p(\vec{n} \mid \vec{\theta}) p(\vec{\theta} \mid \vec{\alpha}) d\vec{\theta} = \frac{C(\vec{\alpha})}{C(\vec{n} + \vec{\alpha})}$$

- Posterior in closed-form:

$$P(\vec{\theta} \mid \{x_1, \ldots, x_N\}, \vec{\alpha}) = \frac{p(\vec{n} \mid \theta) p(\theta \mid \vec{\alpha})}{p(\vec{n} \mid \vec{\alpha})} = C(\vec{n} + \vec{\alpha}) \prod_k \theta_k^{\alpha_k + n_k - 1} \quad = \mathrm{Dir}(\vec{n} + \vec{\alpha})$$

- Posterior predictive rate:

$$p(x_{N+1} = i \mid \{x_1, \ldots, x_N\}, \vec{\alpha}) = \int C(\vec{n} + \vec{\alpha}) \prod_k \theta_k^{\alpha_k + n_k - 1} \times \theta_i^{\alpha_k + n_k} d\vec{\theta} = \frac{C(\vec{n} + \vec{\alpha})}{C(\vec{n} + \vec{\alpha} + x_N)} \quad = \frac{n_i + \alpha_i}{|\vec{n}| + |\vec{\alpha}|}$$

Sequential Bayesian updating

- Start with Dirichlet prior $P(\vec{\theta} \mid \vec{\alpha}) = \mathrm{Dir}(\vec{\theta} : \vec{\alpha})$
- Observe N' samples with sufficient statistics $\vec{n}'$. Posterior becomes:

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

- Observe another N'' samples with sufficient statistics $\vec{n}''$. Posterior becomes:

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

- So sequentially absorbing data in any order is equivalent to batch update.

Hierarchical Bayesian Models

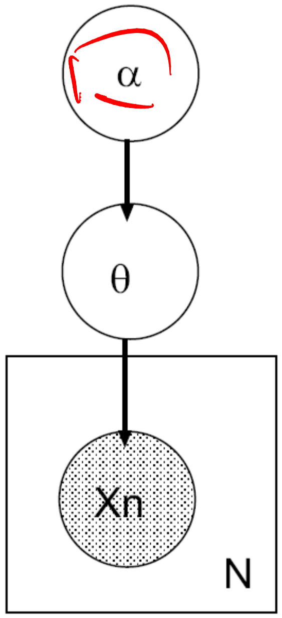
- θ are the parameters for the likelihood $p(x|\theta)$
- α are the parameters for the prior $p(\theta|\alpha)$.
- We can have hyper-hyper-parameters, etc.
- We stop when the choice of hyper-parameters makes no difference to the marginal likelihood; typically make hyper-parameters constants.
- Where do we get the prior?
  - Intelligent guesses
  - Empirical Bayes (Type-II maximum likelihood)

    → computing point estimates of α :

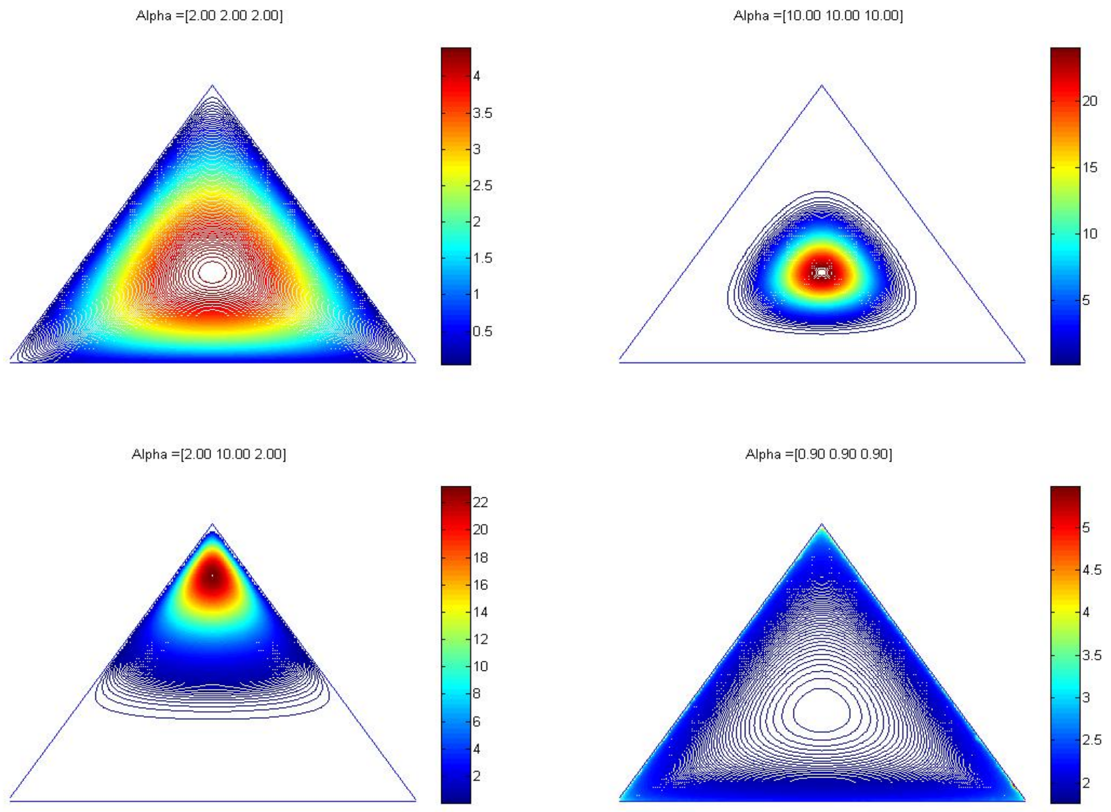
$$\hat{\vec{\alpha}}_{MLE} = \arg\max_{\vec{\alpha}} = p(\vec{n} \mid \vec{\alpha})$$

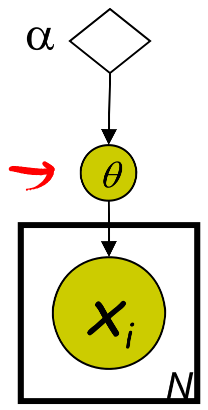
$\int f(x, \theta, \alpha)\, d\theta$

$= f(x, \alpha)$



Limitation of Dirichlet Prior:





The Logistic Normal Prior

$$\theta \sim LN_K(\mu, \Sigma)$$

$$\gamma \sim N_{K-1}(\mu, \Sigma) \qquad \gamma_K = 0$$

$$\theta_i = \exp\left\{ \gamma_i - \log\left(1 + \sum_{i=1}^{K-1} e^{\gamma_i} \right) \right\}$$

$$C(\gamma) = \log\left(1 + \sum_{i=1}^{K-1} e^{\gamma_i} \right)$$

- Log Partition Function
- Normalization Constant

$\mu \quad \Sigma$

$\downarrow$

γ_i

$\downarrow$

$\exp \gamma$

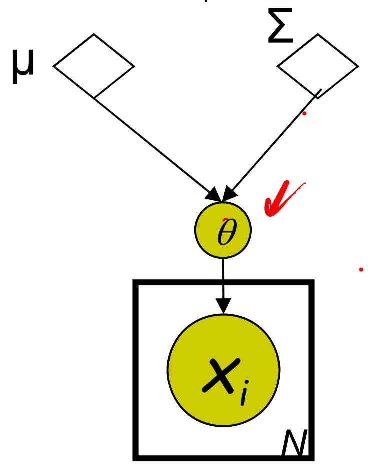
$\sum \exp \gamma_i$

$\downarrow$

θ_i

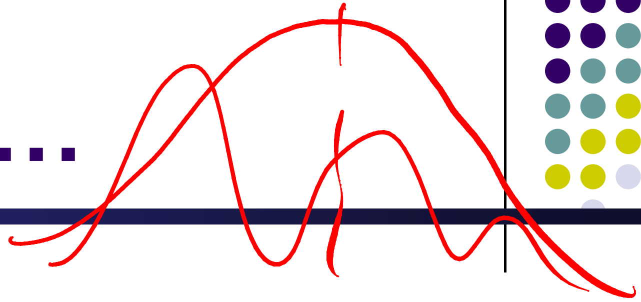
Problem

$P(\theta | x, \mu, \Sigma)$?



- Pro: co-variance structure
- Con: non-conjugate (we will discuss how to solve this later)

Laplace Approximation …



$p(x|\theta)$ multi

$p(\theta|\alpha)$ LN (μ, Σ)

Dir

$p(\theta|x, \alpha) \propto p(x|\theta)\, p(\theta|\alpha)$

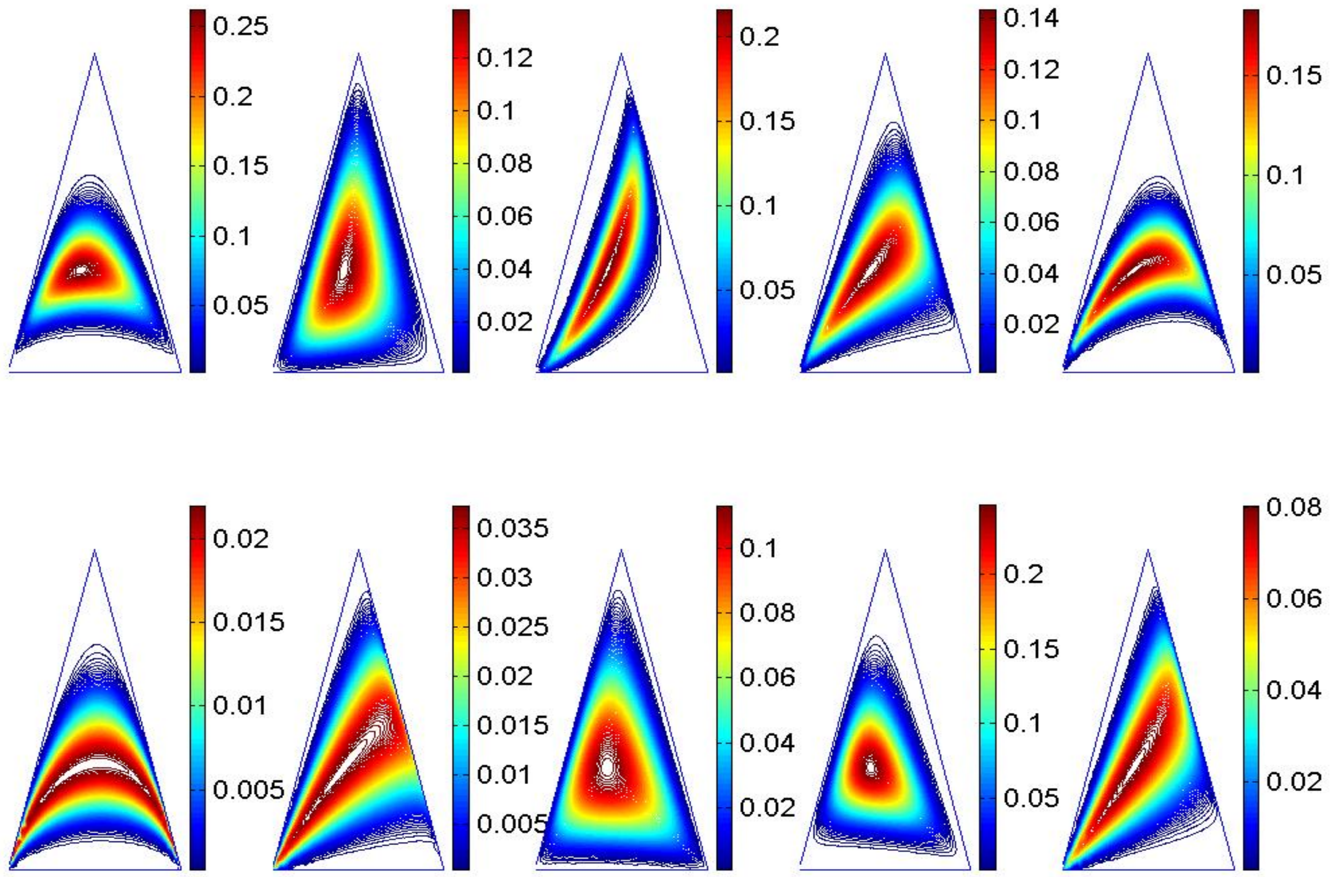
$p(\theta|\alpha, x)$ Dir

p Multi(x, θ) Norm$(\log\theta, -)$

$f(\theta_0) + f'(\theta-\theta_0) + f''(\theta-\theta_0)^2$

$= N(\nu_i, \mu', \Sigma'')$

Logistic Normal Densities

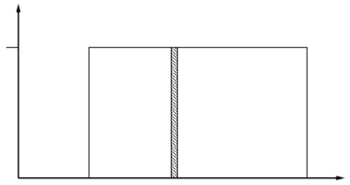


Logistic Normal

Continuous Distributions

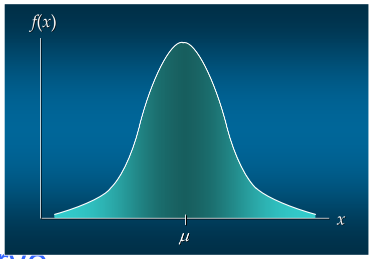
- Uniform Probability Density Function

$$p(x) = 1/(b-a) \quad \text{for } a \le x \le b$$
$$= 0 \quad \text{elsewhere}$$



- Normal (Gaussian) Probability Density Function

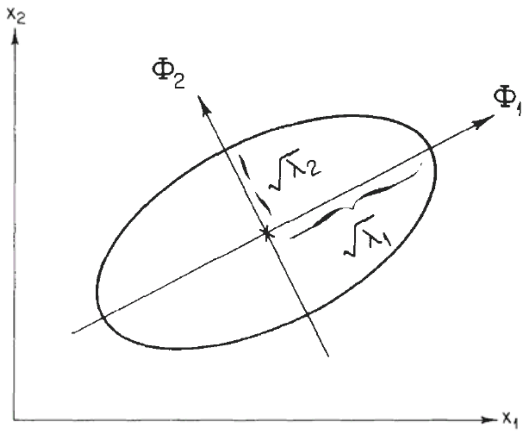
$$p(x) = \frac{1}{\sqrt{2\pi}\sigma} e^{-(x-\mu)^2/2\sigma^2}$$



  - The distribution is symmetric, and is often illustrated as a bell-shaped curve.
  - Two parameters, μ (mean) and σ (standard deviation), determine the location and shape of the distribution.
  - The highest point on the normal curve is at the mean, which is also
  - The mean can be any numerical value: negative, zero, or positive.

- Multivariate Gaussian

$$p(X;\vec{\mu},\Sigma) = \frac{1}{\left(\sqrt{2\pi}\right)^{n/2}|\Sigma|^{1/2}} \exp\left\{-\frac{1}{2}(X-\vec{\mu})^T\Sigma^{-1}(X-\vec{\mu})\right\}$$



MLE for a multivariate-Gaussian

- It can be shown that the MLE for μ *and* Σ is

$$\mu_{MLE} = \frac{1}{N}\sum_n (x_n)$$

$$\Sigma_{MLE} = \frac{1}{N}\sum_n (x_n - \mu_{ML})(x_n - \mu_{ML})^T = \frac{1}{N} S$$

$$x_n = \begin{pmatrix} x_{n,1} \\ x_{n,2} \\ \vdots \\ x_{n,K} \end{pmatrix}$$

$$X = \begin{pmatrix} --- & x_1^T & --- \\ --- & x_2^T & --- \\ & \vdots & \\ --- & x_N^T & --- \end{pmatrix}$$

where the scatter matrix is

$$S = \sum_n (x_n - \mu_{ML})(x_n - \mu_{ML})^T = \left(\sum_n x_n x_n^T\right) - N\mu_{ML}\mu_{ML}^T$$

- The sufficient statistics are $\Sigma_n x_n$ and $\Sigma_n x_n x_n^T$.
- Note that $X^T X = \Sigma_n x_n x_n^T$ may not be full rank (eg. if $N < D$), in which case Σ_{ML} is not invertible

Bayesian parameter estimation for a Gaussian

- There are various reasons to pursue a Bayesian approach
  - We would like to update our estimates sequentially over time.
  - We may have prior knowledge about the expected magnitude of the parameters.
  - The MLE for Σ may not be full rank if we don't have enough data.
- We will restrict our attention to conjugate priors.
- We will consider various cases, in order of increasing complexity:
  - Known σ, unknown μ
  - Known μ, unknown σ
  - Unknown μ and σ

Bayesian estimation: unknown μ, known σ

- Normal Prior:

$$P(\mu) = \left(2\pi\tau^2\right)^{-1/2} \exp\left\{-(\mu-\mu_0)^2/2\tau^2\right\}$$

- Joint probability:

$$P(x,\mu) = \left(2\pi\sigma^2\right)^{-N/2} \exp\left\{-\frac{1}{2\sigma^2}\sum_{n=1}^{N}(x_n-\mu)^2\right\} \times \left(2\pi\tau^2\right)^{-1/2} \exp\left\{-(\mu-\mu_0)^2/2\tau^2\right\}$$

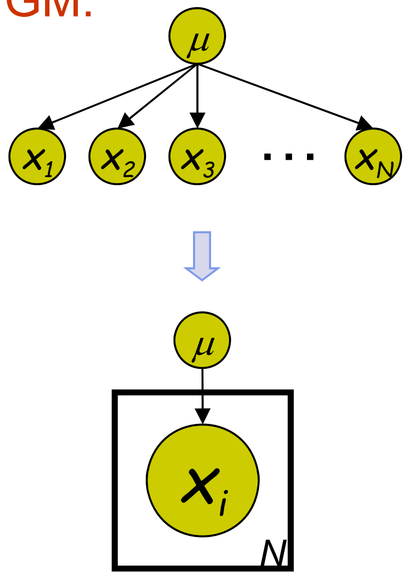
- Posterior:

$$P(\mu \mid x) = \left(2\pi\tilde{\sigma}^2\right)^{-1/2} \exp\left\{-(\mu-\tilde{\mu})^2/2\tilde{\sigma}^2\right\}$$

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 $\tilde{\sigma}^2 = \left(\frac{N}{\sigma^2}+\frac{1}{\tau^2}\right)^{-1}$

Sample mean

GM:



Bayesian estimation: unknown μ, known σ

$$\mu_N = \frac{N/\sigma^2}{N/\sigma^2 + 1/\sigma_0^2}\bar{x} + \frac{1/\sigma_0^2}{N/\sigma^2 + 1/\sigma_0^2}\mu_0 , \qquad \tilde{\sigma}^2 = \left(\frac{N}{\sigma^2} + \frac{1}{\sigma_0^2}\right)^{-1}$$

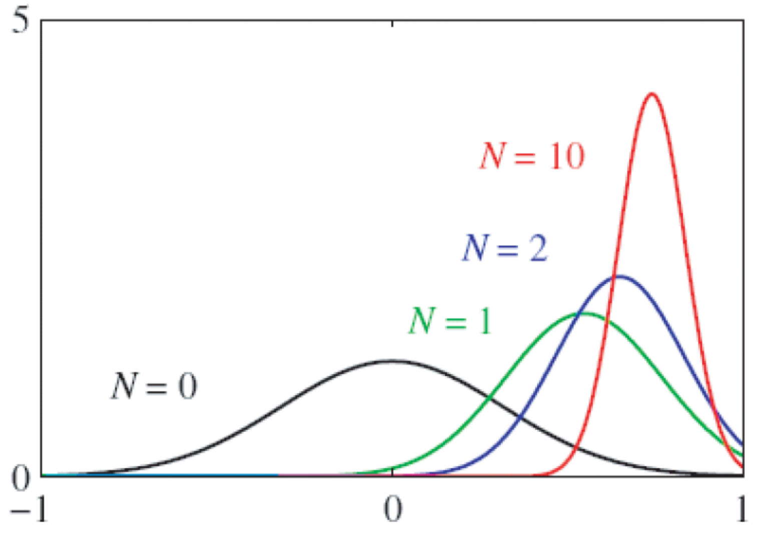
- The posterior mean is a convex combination of the prior and the MLE, with weights proportional to the relative noise levels.
- The precision of the posterior $1/\sigma^2_N$ is the precision of the prior $1/\sigma^2_0$ plus one contribution of data precision $1/\sigma^2$ for each observed data point.

- Sequentially updating the mean
  - $\mu* = 0.8$ (unknown), $(\sigma^2)* = 0.1$ (known)
  - Effect of single data point

$$\mu_1 = \mu_0 + (x - \mu_0)\frac{\sigma_0^2}{\sigma^2 + \sigma_0^2} = x - (x - \mu_0)\frac{\sigma_0^2}{\sigma^2 + \sigma_0^2}$$

  - Uninformative (vague/ flat) prior, $\sigma^2_0 \rightarrow \infty$

$$\mu_N \rightarrow \mu_0$$



Other scenarios

- Known μ, unknown $\lambda = 1/\sigma_2$
  - The conjugate prior for λ is a Gamma with shape a_0 and rate (inverse scale) b_0

$$p(\lambda|a,b) = \frac{1}{\Gamma(a)} b^a \lambda^{a-1} \exp(-b\lambda)$$

  - The conjugate prior for σ^2 is Inverse-Gamma

$$IG(\sigma^2|a,b) = \frac{1}{\Gamma(a)} b^a (\sigma^2)^{-(a+1)} \exp(-b/(\sigma^2))$$

- Unknown μ and unknown σ_2
  - The conjugate prior is

    Normal-Inverse-Gamma

$$\begin{aligned} P(\mu,\sigma^2) &= P(\mu|\sigma^2)P(\sigma^2) \\ &= \mathcal{N}(\mu|m,\sigma^2 V)\ \ IG(\sigma^2|a,b) \end{aligned}$$

  - Semi conjugate prior

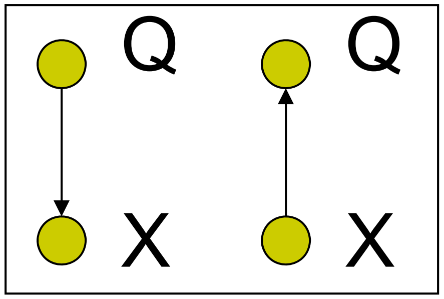
- Multivariate case:
  - The conjugate prior is

    Normal-Inverse-Wishart

$$\begin{aligned} P(\mu,\Sigma) &= P(\mu|\Sigma)P(\Sigma) \\ &= \mathcal{N}(\mu|\mu_0, \frac{1}{\kappa_0}\Sigma)\ \ \mathcal{IW}(\Sigma|\Lambda_0^{-1}, \nu_0) \end{aligned}$$

Estimation of conditional density

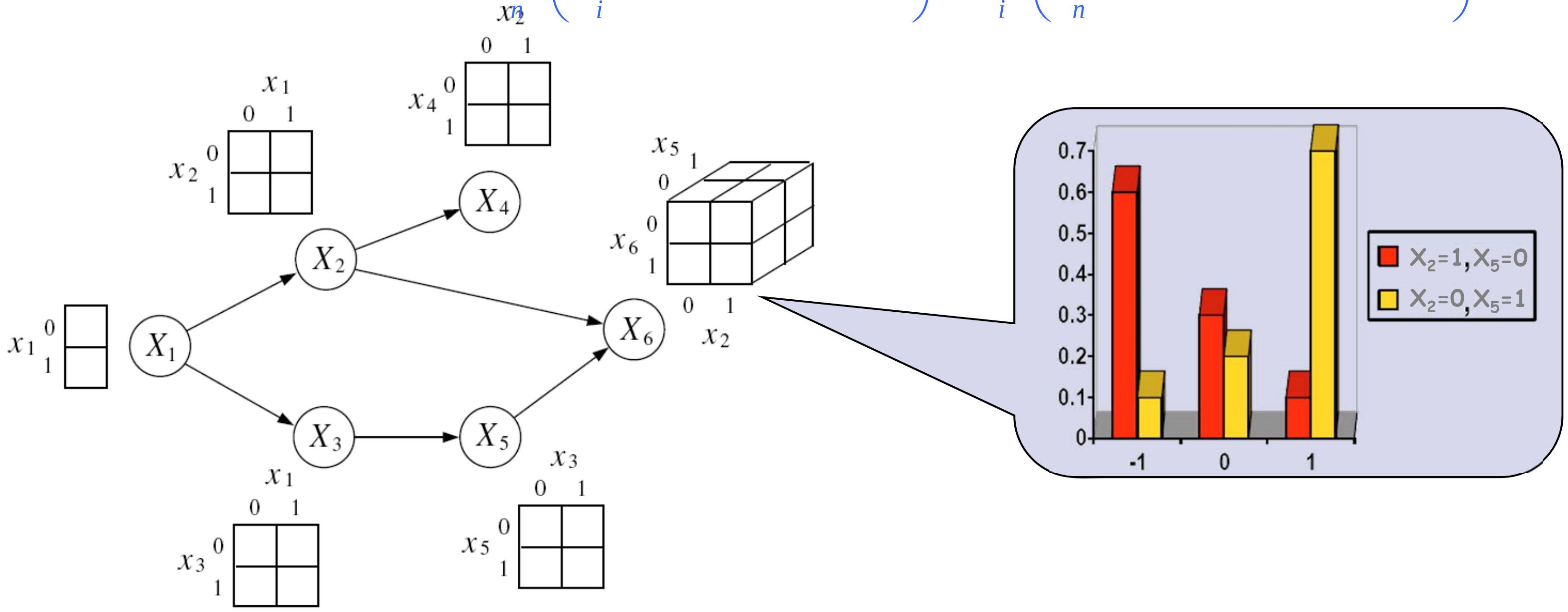
- Can be viewed as two-node graphical models
- Instances of GLIM
- Building blocks of general GM
- MLE and Bayesian estimate
- See supplementary slides



MLE for general BNs

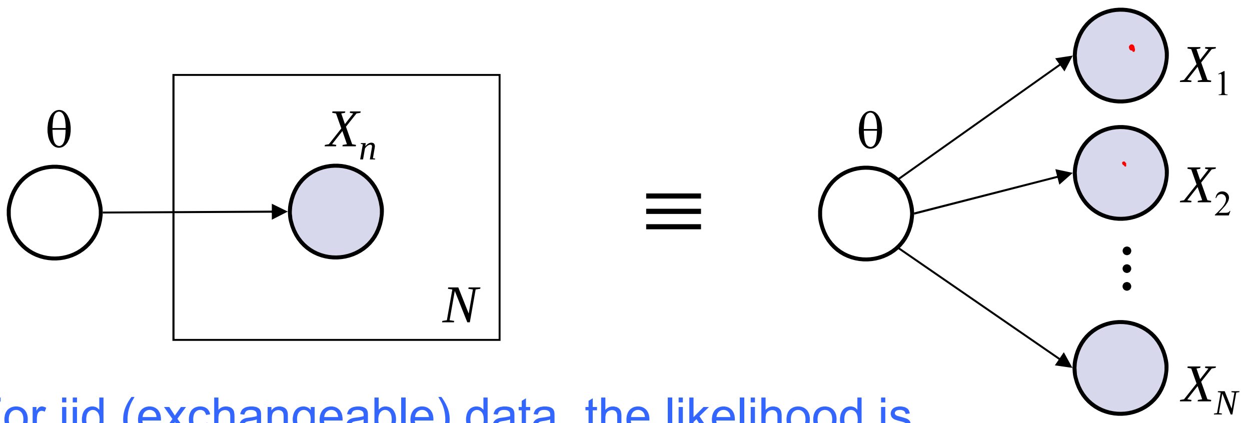
- If we assume the parameters for each CPD are globally independent, and all nodes are fully observed, then the log-likelihood function decomposes into a sum of local terms, one per node:

$$\ell(\theta;D) = \log p(D \mid \theta) = \log \prod_n \left(\prod_i p(x_{n,i} \mid \mathbf{x}_{n,\pi_i}, \theta_i) \right) = \sum_i \left(\sum_n \log p(x_{n,i} \mid \mathbf{x}_{n,\pi_i}, \theta_i) \right)$$



Plates

- A plate is a "macro" that allows subgraphs to be replicated



- For iid (exchangeable) data, the likelihood is

$$p(D \mid \theta) = \prod_n p(x_n \mid \theta)$$

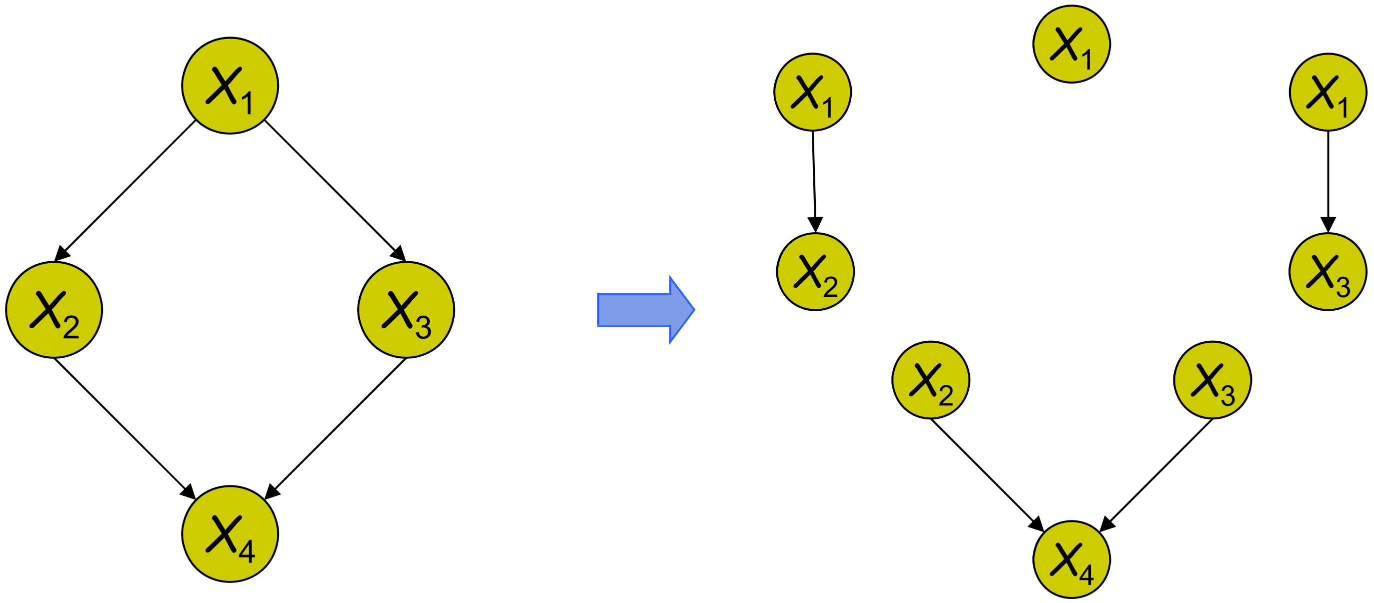
- We can represent this as a Bayes net with *N* nodes.
  - The rules of plates are simple: repeat every structure in a box a number of times given by the integer in the corner of the box (e.g. N), updating the plate index variable (e.g. n) as you go.
  - Duplicate every arrow going into the plate and every arrow leaving the plate by connecting the arrows to each copy of the structure.

Decomposable likelihood of a BN

- Consider the distribution defined by the directed acyclic GM:

$$p(x|\theta) = p(x_1|\theta_1)p(x_2|x_1,\theta_2)p(x_3|x_1,\theta_3)p(x_4|x_2,x_3,\theta_4)$$

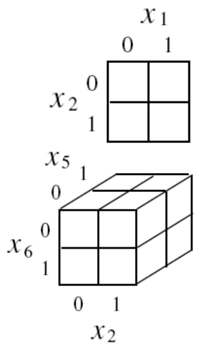
- This is exactly like learning four separate small BNs, each of which consists of a node and its parents.



MLE for BNs with tabular CPDs

- Assume each CPD is represented as a table (multinomial) where

$$\theta_{ijk} \overset{\text{def}}{=} p(X_i = j \mid X_{\pi_i} = k)$$



  - Note that in case of multiple parents, $\mathbf{X}_{\pi_i}$ will have a composite state, and the CPD will be a high-dimensional table
  - The sufficient statistics are counts of family configurations

$$n_{ijk} \overset{\text{def}}{=} \sum_n x_{n,i}^j x_{n,\pi_i}^k$$

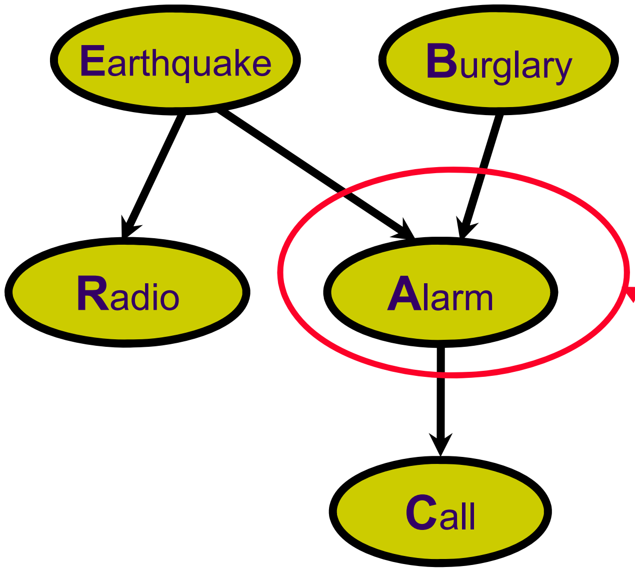
- The log-likelihood is

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

- Using a Lagrange multiplier to enforce $\sum_j \theta_{ijk} = 1$, we get:

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

How to define parameter prior?



Factorization: $p(\mathbf{X}=\mathbf{x}) = \prod_{i=1}^{M} p(x_i \mid \mathbf{x}_{\pi_i})$

Local Distributions defined by, e.g., multinomial parameters:

$$p(x_i^k \mid \mathbf{x}_{\pi_i}^j) = \theta_{x_i^k \mid \mathbf{x}_{\pi_i}^j}$$

$$p(\theta \mid G)\,?$$

Assumptions **(Geiger & Heckerman 97,99):**

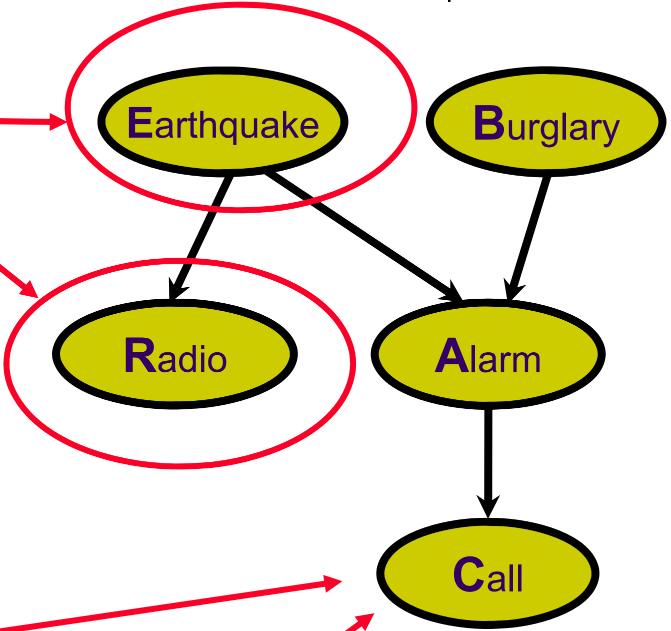
- Complete Model Equivalence
- Global Parameter Independence
- Local Parameter Independence
- Likelihood and Prior Modularity

Global & Local Parameter Independence

- Global Parameter Independence

  For every DAG model:

$$p(\theta_m \mid G) = \prod_{i=1}^{M} p(\theta_i \mid G)$$



- Local Parameter Independence

  For every node:

$$p(\theta_i \mid G) = \prod_{j=1}^{q_i} p(\theta_{x_i^k \mid \mathbf{x}_{\pi_i}^j} \mid G)$$

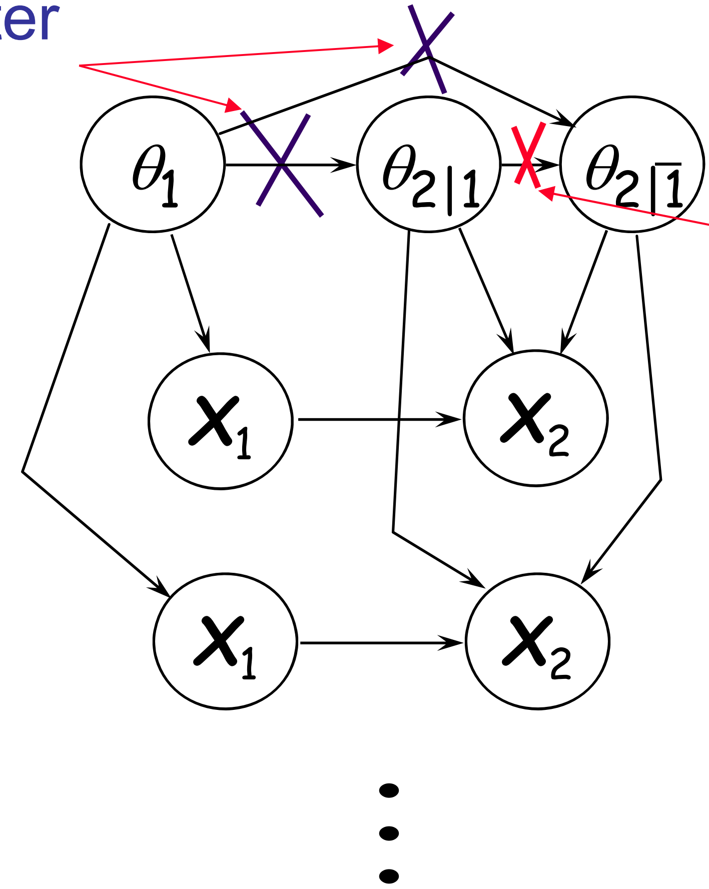
$$P(\theta_{Call|Alarm=YES})$$

independent of

$$P(\theta_{Call|Alarm=NO})$$

Parameter Independence, Graphical View

Global Parameter Independence



Local Parameter Independence

sample 1

sample 2

Provided all variables are observed in all cases, we can perform Bayesian update each parameter independently !!!

Which PDFs Satisfy Our Assumptions? (Geiger & Heckerman 97,99)

- **Discrete DAG Models:** $x_i \mid \pi_{x_i}^j \sim \mathrm{Multi}(\theta)$

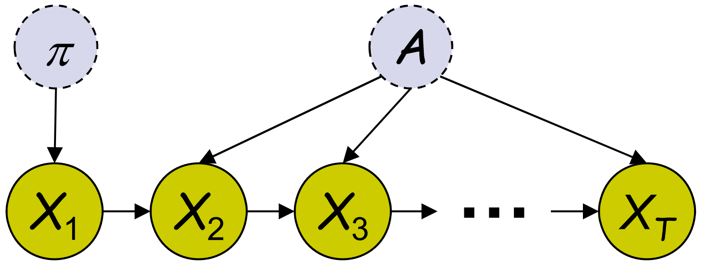
  Dirichlet prior:
  $$P(\theta) = \frac{\Gamma(\sum_k \alpha_k)}{\prod_k \Gamma(\alpha_k)} \prod_k \theta_k^{\alpha_k - 1} = C(\alpha) \prod_k \theta_k^{\alpha_k - 1}$$

- **Gaussian DAG Models:** $x_i \mid \pi_{x_i}^j \sim \mathrm{Normal}(\mu, \Sigma)$

  Normal prior:
  $$p(\mu \mid \nu, \Psi) = \frac{1}{(2\pi)^{n/2} \mid \Psi \mid^{1/2}} \exp\left\{ -\frac{1}{2} (\mu - \nu)' \Psi^{-1} (\mu - \nu) \right\}$$

  Normal-Wishart prior:
  $$p(\mu \mid \nu, \alpha_\mu, \mathbf{W}) = \mathrm{Normal}\left(\nu, (\alpha_\mu \mathbf{W})^{-1}\right),$$
  $$p(\mathrm{W} \mid \alpha_w, \mathrm{T}) = c(n, \alpha_w) \left|\mathrm{T}\right|^{\alpha_w / 2} \left|\mathrm{W}\right|^{(\alpha_w - n - 1)/2} \exp\left\{ \frac{1}{2} \mathrm{tr}\{\mathrm{TW}\} \right\},$$
  where $\mathbf{W} = \Sigma^{-1}$.

Parameter sharing



- Consider a time-invariant (stationary) 1st-order Markov model
  - Initial state probability vector: $\pi_k \overset{\text{def}}{=} p(X_1^k = 1)$
  - State transition probability matrix: $A_{ij} \overset{\text{def}}{=} p(X_t^j = 1 \mid X_{t-1}^i = 1)$
- The joint: $p(X_{1:T} \mid \theta) = p(x_1 \mid \pi) \prod_{t=2}^{T} \prod_{t=2} p(X_t \mid X_{t-1})$
- The log-likelihood: $\ell(\theta; D) = \sum_n \log p(x_{n,1} \mid \pi) + \sum_n \sum_{t=2}^{T} \log p(x_{n,t} \mid x_{n,t-1}, A)$
- Again, we optimize each parameter separately
  - π is a multinomial frequency vector, and we've seen it before
  - What about A?

Learning a Markov chain transition matrix

- A is a stochastic matrix: $\sum_j A_{ij} = 1$
- Each row of A is multinomial distribution.
- So **MLE** of A_{ij} is the fraction of transitions from *i* to *j*

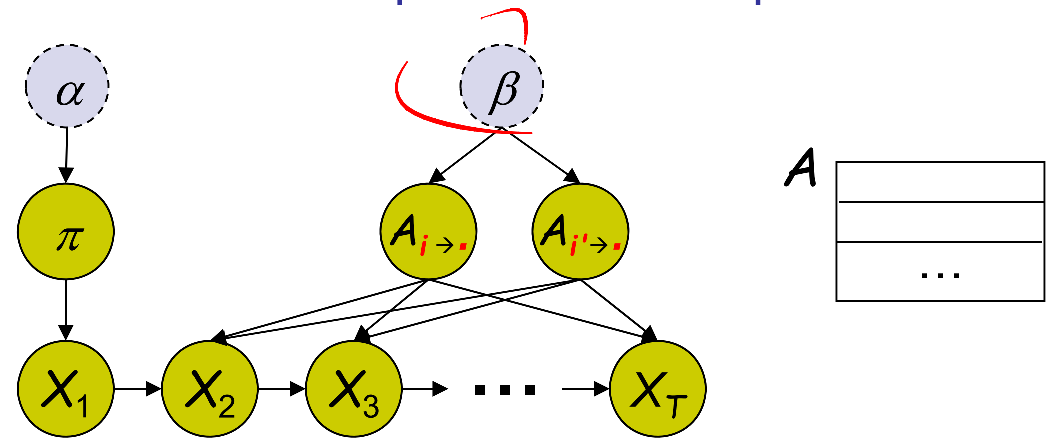
$$\underline{A_{ij}^{ML} = \frac{\#(i \to j)}{\#(i \to \bullet)}} = \frac{\sum_n \sum_{t=2}^{T} x_{n,t-1}^{i} x_{n,t}^{j}}{\sum_n \sum_{t=2}^{T} x_{n,t-1}^{i}}$$

- Application:
  - if the states X_t represent words, this is called a *bigram language model*
- Sparse data problem:
  - If $i \to j$ did not occur in data, we will have A_{ij} =0, then any future sequence with word pair $i \to j$ will have zero probability.
  - A standard hack: *backoff smoothing* or *deleted interpolation*

$$\tilde{A}_{i\to\bullet} = \overbrace{\lambda\eta_t} + (1-\lambda)A_{i\to\bullet}^{ML}$$

Bayesian language model

- Global and local parameter independence



- The posterior of $A_{i\rightarrow\cdot}$ and $A_{i'\rightarrow\cdot}$ is factorized despite v-structure on X_t, because X_{t-1} acts like a multiplexer
- Assign a Dirichlet prior β_i to each row of the transition matrix:

$$A_{ij}^{Bayes} \overset{\text{def}}{=} p(j \mid i, D, \beta_i) = \frac{\#(i \rightarrow j) + \beta_{i,k}}{\#(i \rightarrow \bullet) + |\beta_i|} = \lambda_i \beta'_{i,k} + (1-\lambda_i) A_{ij}^{ML}, \text{ where } \lambda_i = \frac{|\beta_i|}{|\beta_i| + \#(i \rightarrow \bullet)}$$

- We could consider more realistic priors, e.g., mixtures of Dirichlets to account for types of words (adjectives, verbs, etc.)

Example: HMM: two scenarios

- **Supervised learning**: estimation when the “right answer” is known
  - **Examples:**

    GIVEN: a genomic region $x = x_1 \ldots x_{1,000,000}$ where we have good (experimental) annotations of the CpG islands

    GIVEN: the casino player allows us to observe him one evening, as he changes dice and produces 10,000 rolls

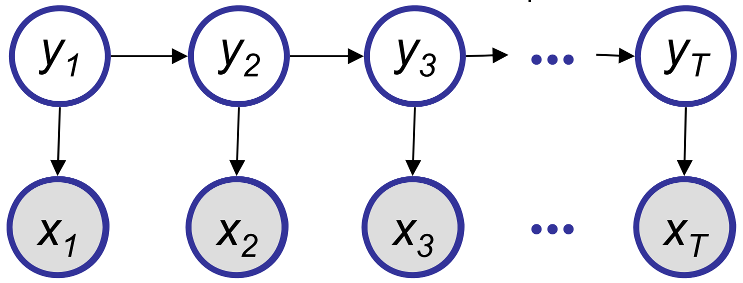
- **Unsupervised learning**: estimation when the “right answer” is unknown
  - **Examples:**

    GIVEN: the porcupine genome; we don’t know how frequent are the CpG islands there, neither do we know their composition

    GIVEN: 10,000 rolls of the casino player, but we don’t see when he changes dice

- **QUESTION:** Update the parameters θ of the model to maximize $P(x|\theta)$ --- Maximal likelihood (ML) estimation

Recall definition of HMM



- Transition probabilities between any two states

$$p(y_t^j = 1 \mid y_{t-1}^i = 1) = a_{i,j},$$

or $$p(y_t \mid y_{t-1}^i = 1) \sim \text{Multinomial}\left(a_{i,1}, a_{i,2}, \ldots, a_{i,M}\right), \forall i \in \mathbb{I}.$$

- Start probabilities

$$p(y_1) \sim \text{Multinomial}\left(\pi_1, \pi_2, \ldots, \pi_M\right).$$

- Emission probabilities associated with each state

$$p(x_t \mid y_t^i = 1) \sim \text{Multinomial}\left(b_{i,1}, b_{i,2}, \ldots, b_{i,K}\right), \forall i \in \mathbb{I}.$$

or in general: $$p(x_t \mid y_t^i = 1) \sim \mathrm{f}\left(\cdot \mid \theta_i\right), \forall i \in \mathbb{I}.$$

Supervised ML estimation

- Given $x = x_1 \ldots x_N$ for which the true state path $y = y_1 \ldots y_N$ is known,
  - **Define:**

A_{ij}	= # times state transition $i \rightarrow j$ occurs in $\mathbf{y}$
B_{ik}	= # times state i in $\mathbf{y}$ emits k in $\mathbf{x}$

  - **We can show that the maximum likelihood parameters θ are:**

$$a_{ij}^{ML} = \frac{\#(i \rightarrow j)}{\#(i \rightarrow \bullet)} = \frac{\sum_n \sum_{t=2}^{T} y_{n,t-1}^i y_{n,t}^j}{\sum_n \sum_{t=2}^{T} y_{n,t-1}^i} = \frac{A_{ij}}{\sum_{j'} A_{ij'}}$$

$$b_{ik}^{ML} = \frac{\#(i \rightarrow k)}{\#(i \rightarrow \bullet)} = \frac{\sum_n \sum_{t=1}^{T} y_{n,t}^i x_{n,t}^k}{\sum_n \sum_{t=1}^{T} y_{n,t}^i} = \frac{B_{ik}}{\sum_{k'} B_{ik'}}$$

  - **What if x is continuous? We can treat $\{(x_{n,t}, y_{n,t}) : t = 1:T, n = 1:N\}$ as $N \times T$ observations of, e.g., a Gaussian, and apply learning rules for Gaussian …**

Supervised ML estimation, ctd.

- **Intuition:**
  - When we know the underlying states, the best estimate of θ is the average frequency of transitions & emissions that occur in the training data

- **Drawback:**
  - Given little data, there may be **overfitting**:
    - $P(x|\theta)$ is maximized, but θ is unreasonable

      0 probabilities – VERY BAD

- **Example:**
  - Given 10 casino rolls, we observe

    x = 2, 1, 5, 6, 1, 2, 3, 6, 2, 3
    y = F, F, F, F, F, F, F, F, F, F

  - Then: $a_{FF} = 1$; $a_{FL} = 0$

    $b_{F1} = b_{F3} = .2$;

    $b_{F2} = .3$; $b_{F4} = 0$; $b_{F5} = b_{F6} = .1$

Pseudocounts

- Solution for small training sets:
  - Add pseudocounts

    A_{ij} = # times state transition $i \rightarrow j$ occurs in $\mathbf{y}$ + R_{ij}

    B_{ik} = # times state i in $\mathbf{y}$ emits k in $\mathbf{x}$ + S_{ik}

  - R_{ij}, S_{ij} are pseudocounts representing our prior belief
  - Total pseudocounts: $R_i = \Sigma_j R_{ij}$, $S_i = \Sigma_k S_{ik}$,
    - --- "strength" of prior belief,
    - --- total number of imaginary instances in the prior

- Larger total pseudocounts $\Rightarrow$ strong prior belief

- Small total pseudocounts: just to avoid 0 probabilities --- smoothing

- This is equivalent to Bayesian est. under a uniform prior with "parameter strength" equals to the pseudocounts

Summary: Learning GM

- For fully observed BN, the log-likelihood function decomposes into a sum of local terms, one per node; thus learning is also factored
  - Structural learning
    - Chow liu
    - Neighborhood selection
  - Learning single-node GM – density estimation: exponential family dist.
    - Typical discrete distribution
    - Typical continuous distribution
    - Conjugate priors
  - Learning two-node BN: GLIM
    - Conditional Density Est.
    - Classification
  - Learning BN with more nodes
    - Local operations

Supplemental review:

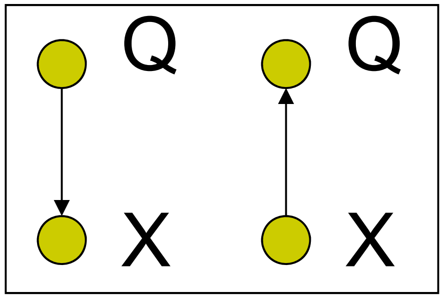
Two node fully observed BNs

Conditional mixtures

Linear/Logistic Regression

Classification

- Generative and discriminative approaches

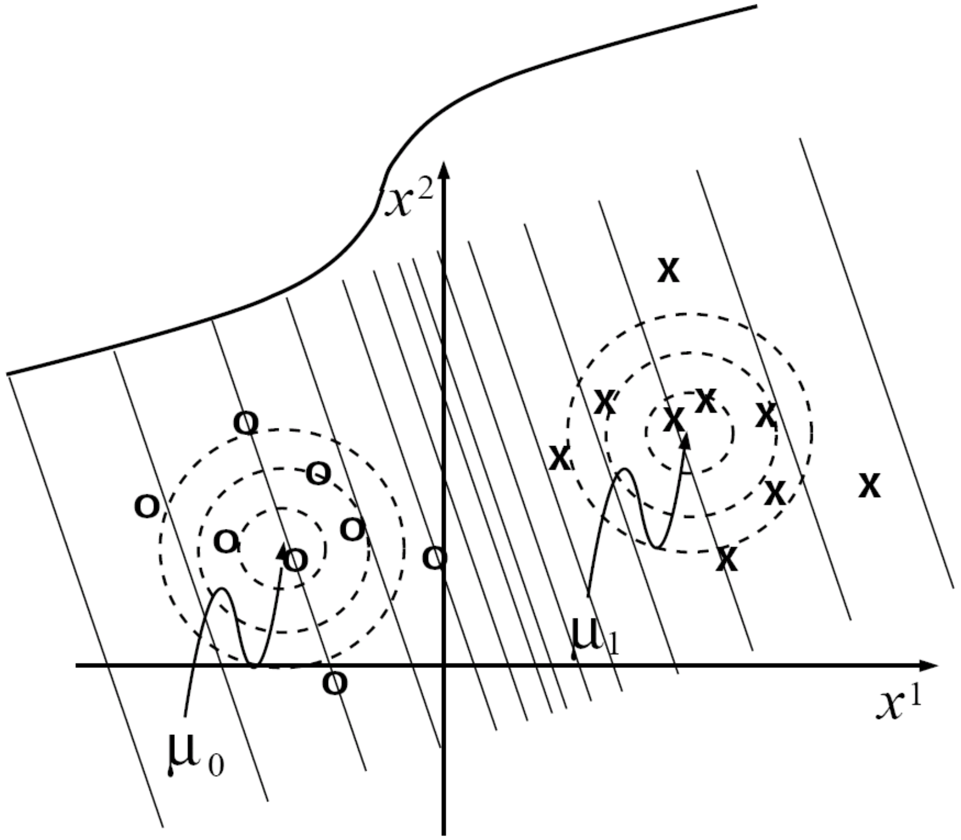


Classification:

- Goal: Wish to learn f: X $\rightarrow$ Y

- Generative:
  - Modeling the joint distribution of all data

- Discriminative:
  - Modeling only points at the boundary



Conditional Gaussian

- The data:

$$\{(x_1,y_1),(x_2,y_2),(x_3,y_3),\cdots,(x_N,y_N)\}$$

- Both nodes are observed:
  - *Y* is a class indicator vector

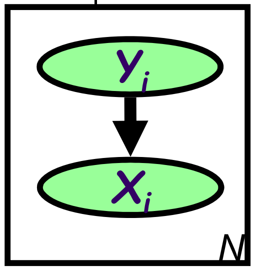
$$p(y_n) = \text{multi}(y_n : \pi) = \prod_k \pi_k^{y_{n,k}}$$

  - *X* is a conditional Gaussian variable with a class-specific mean

$$p(x_n \mid y_{n,k}=1,\mu,\sigma) = \frac{1}{(2\pi\sigma^2)^{1/2}} \exp\left\{-\tfrac{1}{2\sigma^2}(x_n-\mu_k)^2\right\}$$

$$p(x \mid y,\mu,\sigma) = \prod_n \left(\prod_k N(x_n : \mu_k,\sigma)^{y_{n,k}} \right)$$

GM:

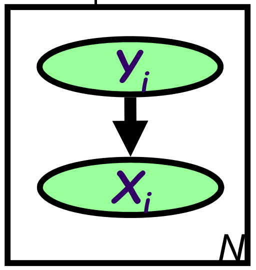


MLE of conditional Gaussian

- Data log-likelihood

$$\ell(\theta; D) = \log \prod_n p(x_n, y_n) = \log \prod_n p(y_n \mid \pi) p(x_n \mid y_n, \mu, \sigma)$$

GM:



- MLE

)

$$\hat{\pi}_{k,MLE} = \arg\max_{\pi} \ell(\theta; D), \qquad \hat{\pi}_{k,MLE} = \sum_n y_{n,k} \Big/ N = n_k \Big/ N$$

the fraction of samples of class *m*

$$\hat{\mu}_{k,MLE} = \arg\max \ell(\theta; D), \qquad \hat{\mu}_{k,MLE} = \frac{\sum_n y_{n,k} x_n}{\sum_n y_{n,k}} = \frac{\sum_n y_{n,k} x_n}{n_k}$$

the average of samples of class *m*

Bsyesian estimation of conditional Gaussian

- Prior:

$$P(\vec{\pi} \mid \vec{\alpha}) = \mathrm{Dir}(\vec{\pi} : \vec{\alpha})$$

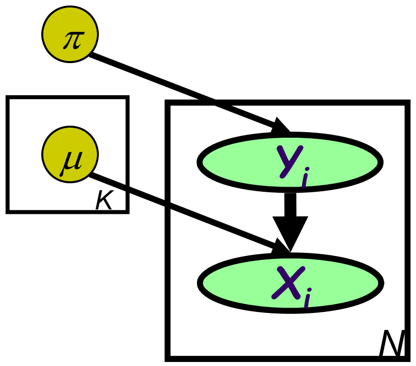
$$P(\mu_k \mid \nu) = \mathrm{Normal}(\mu_k : \nu, \tau)$$

- Posterior mean (Bayesian est.)

$$\pi_{k,Bayes} = \frac{N}{N+|\alpha|}\widehat{\pi}_{k,ML} + \frac{|\alpha|}{N+|\alpha|}\frac{\alpha_k}{|\alpha|} = \frac{n_k + \alpha_k}{N+|\alpha|}$$

$$\mu_{k,Bayes} = \frac{n_k/\sigma^2}{n_k/\sigma^2 + 1/\tau^2}\widehat{\mu}_{k,ML} + \frac{1/\tau^2}{n_k/\sigma^2 + 1/\tau^2}\nu\,, \quad \text{and } \sigma^2_{Bayes} = \left(\frac{N}{\sigma^2} + \frac{1}{\tau^2}\right)^{-1}$$

GM:



Classification

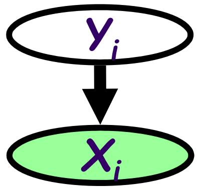
- Gaussian Discriminative Analysis:
  - The joint probability of a datum and it label is:

$$p(x_n, y_{n,k}=1 \mid \mu,\sigma) = p(y_{n,k}=1) \times p(x_n \mid y_{n,k}=1, \mu, \sigma)$$
$$= \pi_k \frac{1}{(2\pi\sigma^2)^{1/2}} \exp\left\{ -\tfrac{1}{2\sigma^2}(x_n - \mu_k)^2 \right\}$$

  - Given a datum x_n, we predict its label using the conditional probability of the label given the datum:

$$p(y_{n,k}=1 \mid x_n, \mu, \sigma) = \frac{\pi_k \frac{1}{(2\pi\sigma^2)^{1/2}} \exp\left\{ -\tfrac{1}{2\sigma^2}(x_n - \mu_k)^2 \right\}}{\sum_{k'} \pi_{k'} \frac{1}{(2\pi\sigma^2)^{1/2}} \exp\left\{ -\tfrac{1}{2\sigma^2}(x_n - \mu_{k'})^2 \right\}}$$

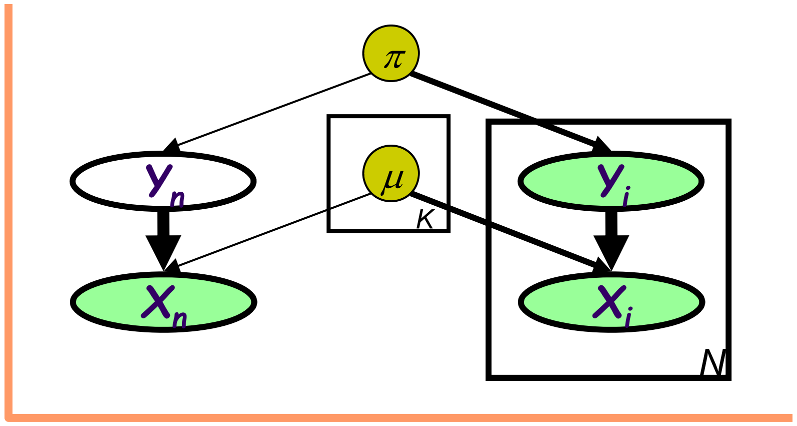
- This is basic inference
  - introduce evidence, and then normalize



Transductive classification

- Given X_n, what is its corresponding Y_n when we know the answer for a set of training data?

GM:



- Frequentist prediction:
  - we fit π, μ and σ from data first, and then …

$$p(y_{n,k}=1 \mid x_n, \mu, \sigma, \pi) = \frac{p(y_{n,k}=1, x_n \mid \mu, \sigma, \pi)}{p(x_n \mid \mu, \sigma, \pi)} = \frac{\pi_k N(x_n, \mid \mu_k, \sigma)}{\sum_{k'} \pi_{k'} N(x_n, \mid \mu_{k'}, \sigma)}$$

- Bayesian:
  - we compute the posterior dist. of the parameters first …

Linear Regression

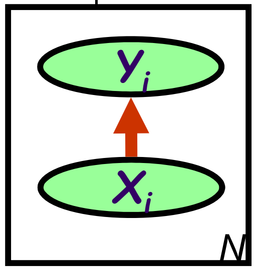
- The data:

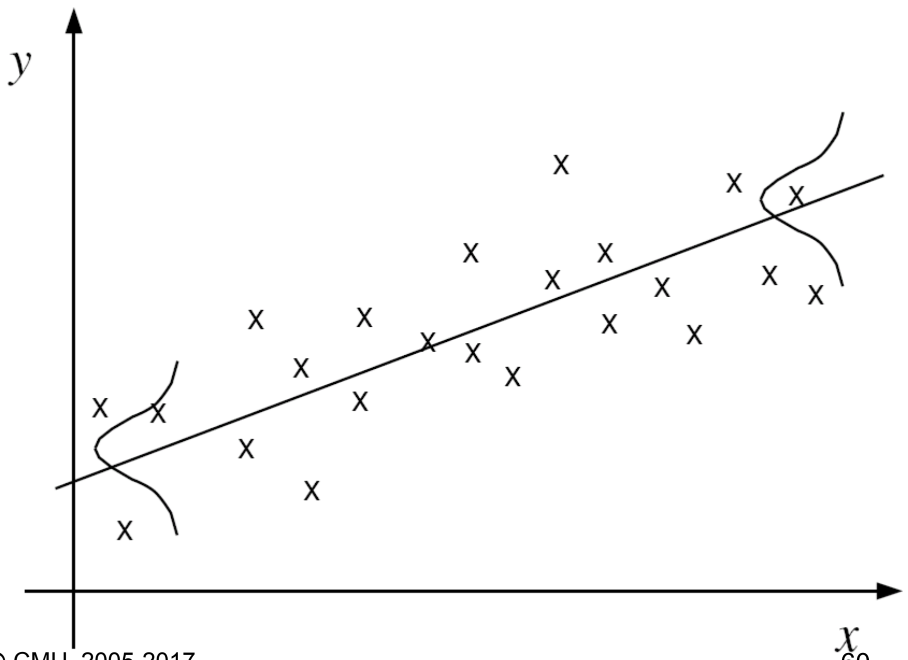
$$\{(x_1, y_1), (x_2, y_2), (x_3, y_3), \cdots, (x_N, y_N)\}$$

- Both nodes are observed:
  - X is an input vector
  - Y is a response vector

    (we first consider y as a generic continuous response vector, then we consider the special case of classification where y is a discrete indicator)

- A regression scheme can be used to model $p(y|x)$ directly, rather than $p(x,y)$





A discriminative probabilistic model

- Let us assume that the target variable and the inputs are related by the equation:

$$y_i = \theta^T \mathbf{x}_i + \varepsilon_i$$

where ε is an error term of unmodeled effects or random noise

- Now assume that ε follows a Gaussian $N(0,\sigma)$, then we have:

$$p(y_i \mid x_i;\theta) = \frac{1}{\sqrt{2\pi}\sigma}\exp\left(-\frac{(y_i - \theta^T \mathbf{x}_i)^2}{2\sigma^2}\right)$$

- By independence assumption:

$$L(\theta) = \prod_{i=1}^{n} p(y_i \mid x_i;\theta) = \left(\frac{1}{\sqrt{2\pi}\sigma}\right)^n \exp\left(-\frac{\sum_{i=1}^{n}(y_i - \theta^T \mathbf{x}_i)^2}{2\sigma^2}\right)$$

Linear regression

- Hence the log-likelihood is:

$$l(\theta) = n \log \frac{1}{\sqrt{2\pi}\sigma} - \frac{1}{\sigma^2}\frac{1}{2}\sum_{i=1}^{n}(y_i - \theta^T \mathbf{x}_i)^2$$

- Do you recognize the last term?

Yes it is: $$J(\theta) = \frac{1}{2}\sum_{i=1}^{n}(\mathbf{x}_i^T \theta - y_i)^2$$

- It is same as the MSE!

A recap:

- LMS update rule

$$\theta^{t+1} = \theta^t + \alpha(y_n - \mathbf{x}_n{}^T \theta^t)\mathbf{x}_n$$

  - Pros: on-line, low per-step cost
  - Cons: coordinate, maybe slow-converging

- Steepest descent

$$\theta^{t+1} = \theta^t + \alpha \sum_{i=1}^{n} (y_n - \mathbf{x}_n{}^T \theta^t)\mathbf{x}_n$$

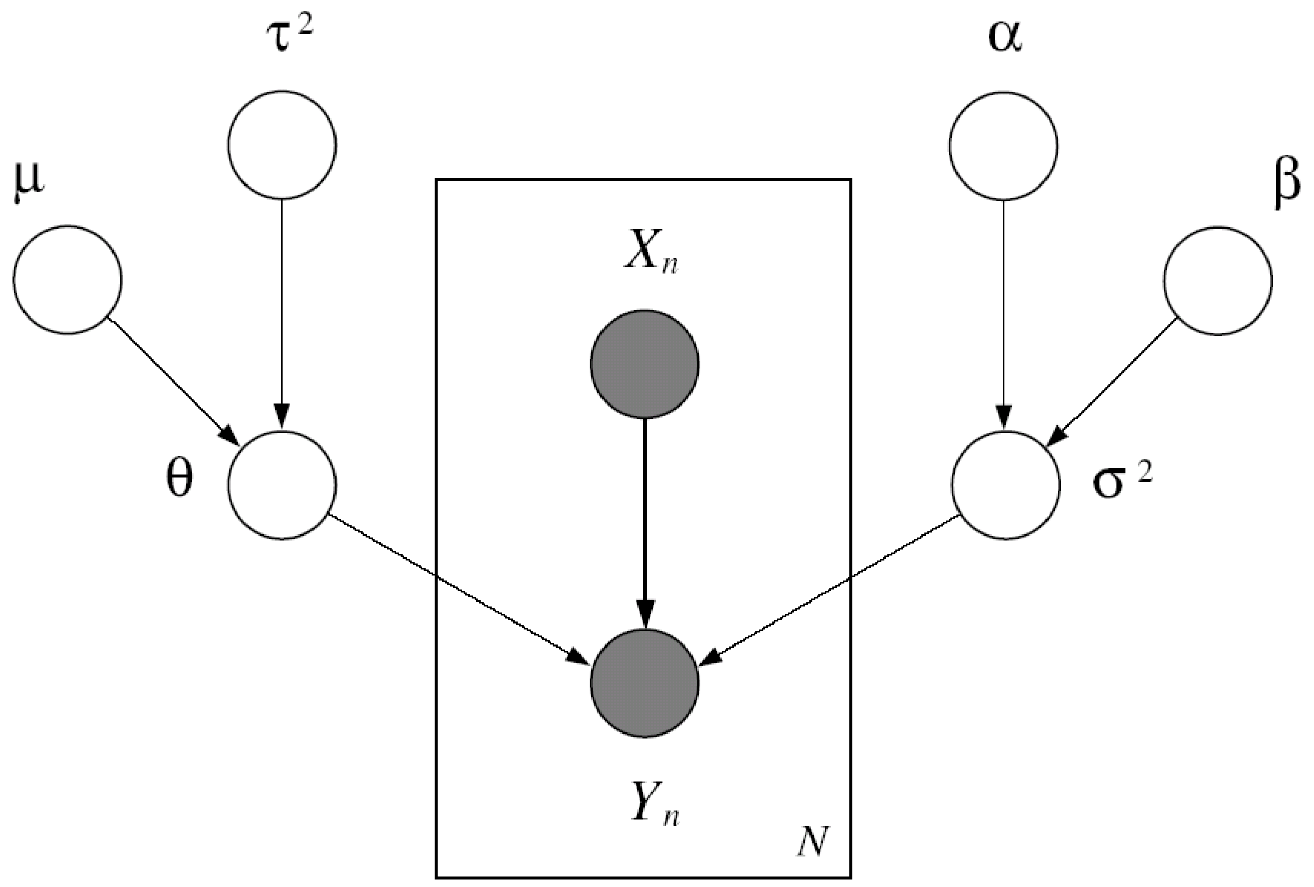
  - Pros: fast-converging, easy to implement
  - Cons: a batch,

- Normal equations

$$\theta^* = \left(X^T X\right)^{-1} X^T \vec{y}$$

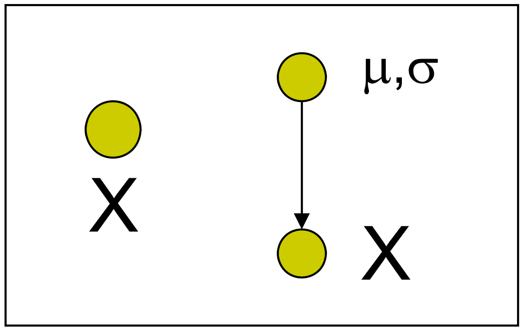
  - Pros: a single-shot algorithm! Easiest to implement.
  - Cons: need to compute pseudo-inverse $(X^T X)^{-1}$, expensive, numerical issues (e.g., matrix is singular ..)

Bayesian linear regression

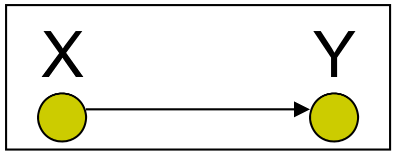


Simple GMs are the building blocks of complex BNs

- Density estimation
  - Parametric and nonparametric methods



- Regression
  - Linear, conditional mixture, nonparametric



- Classification
  - Generative and discriminative approach