Bayesian Networks (Structure) Learning

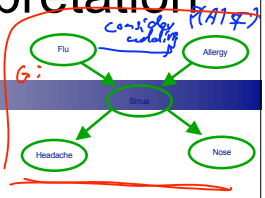
Machine Learning – 10701/15781
Carlos Guestrin
Carnegie Mellon University

November 12th, 2007

Information-theoretic interpretation of maximum likelihood 1

- Given structure, log likelihood of data:

$$\log P(\mathcal{D} \mid \theta_{\mathcal{G}}, \mathcal{G})$$



$$\log \prod_{i=1}^{m} P(x^{(i)} \mid \theta_G, G) = \sum_i \log P(x^{(i)} \mid \theta_G, G)$$

$$= \sum_i \log P(f^{(i)}) P(a^{(i)}) \cdot P(s^{(i)} \mid f^{(i)}, a^{(i)}) \cdot P(h^{(i)} \mid s^{(i)}) P(n^{(i)} \mid s^{(i)})$$

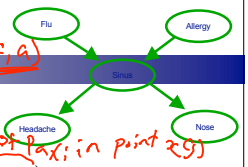
$$= \left[\sum_i \log P(f^{(i)})\right] + \left[\sum_i \log P(a^{(i)})\right] + \left[\sum_i \log P(s^{(i)} \mid f^{(i)}, a^{(i)})\right] + \left[\sum_i \log P(h^{(i)} \mid s^{(i)})\right] + \left[\sum_i \log P(n^{(i)} \mid s^{(i)})\right]$$

$P(F)$ $P(A)$ $P(S|F,A)$ $P(H|S)$ $P(N|S)$

one per CPT $\prod_i P(x_i \mid Pa_{x_i})$

only this term changes to something about $P(A|F)$

Information-theoretic interpretation of maximum likelihood 2



- Given structure, log likelihood of data:

$$\log P(\mathcal{D} \mid \theta_{\mathcal{G}}, \mathcal{G}) = \sum_{j=1}^{m} \sum_{i=1}^{n} \log P\left(X_i = x_i^{(j)} \mid \mathbf{Pa}_{X_i} = \mathbf{x}^{(j)}\left[\mathbf{Pa}_{X_i}\right]\right)$$

data points, nodes, point j, value of Pa_{X_i} in point $x^{(j)}$

$$= \sum_{i=1}^{n} \sum_{j=1}^{m} \log P(x_i = x_i^{(j)} \mid Pa_{x_i} = x^{(j)}[Pa_{x_i}])$$

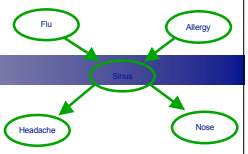
for point j where $F=t, A=f, S=t$: $(\text{count}(F=t, A=f, S=t)]$ $\log P(S=t \mid F=t, A=f)$

$$= \sum_{i=1}^{n} \sum_{x_i} \sum_{u} \frac{\text{count}(x_i = x_i, Pa_{x_i} = u)}{m} \log P(X_i = x_i \mid Pa_{x_i} = u)$$

$\hat{P}(x_i = x_i, Pa_{x_i} = u)$

MLE estimate

Information-theoretic interpretation of maximum likelihood 3



- Given structure, log likelihood of data:

$$\log \hat{P}(\mathcal{D} \mid \theta, \mathcal{G}) = m \sum_{i} \sum_{x_i, \mathbf{Pa}_{x_i,\mathcal{G}}} \hat{P}(x_i, \mathbf{Pa}_{x_i,\mathcal{G}}) \log \hat{P}(x_i \mid \mathbf{Pa}_{x_i,\mathcal{G}})$$

$\mathbf{Pa}_{x_i,\mathcal{G}} = u$

hat because MLE

$$= -m \sum_{i} \hat{H}(X_i \mid Pa_{X_i})$$

MLE tries to pick parents s.t. $H(X_i \mid Pa_{X_i})$ small, i.e. little uncertainty about X_i given Pa_{X_i}

$$= m \sum_{i} \left(I(X_i, Pa_{X_i}) - H(X_i)\right)$$

$H(A \mid B) = -\sum_{ab} P(a,b) \cdot \log P(a \mid b)$

$I(A,B) = \sum_{ab} P(a,b) \log \frac{P(a,b)}{P(a) \cdot P(b)}$

$= H(A) - H(A \mid B)$

Decomposable score

- Log data likelihood

$$\log \hat{P}(\mathcal{D} \mid \theta, \mathcal{G}) = m \sum_i \hat{I}(x_i, \mathbf{Pa}_{x_i,\mathcal{G}}) - M \sum_i \hat{H}(X_i)$$

- Decomposable score:
  - Decomposes over families in BN (node and its parents)
  - Will lead to significant computational efficiency!!!
  - Score($G : D$) = $\sum_i$ FamScore(X_i|$\mathbf{Pa}_{Xi}$: D)

How many trees are there?

Nonetheless – Efficient optimal algorithm finds best tree

Scoring a tree 1: equivalent trees

$$\log \hat{P}(\mathcal{D} \mid \theta, \mathcal{G}) = M \sum_i \hat{I}(x_i, \mathbf{Pa}_{x_i,\mathcal{G}}) - M \sum_i \hat{H}(X_i)$$

Scoring a tree 2: similar trees

$$\log \hat{P}(\mathcal{D} \mid \theta, \mathcal{G}) = M \sum_i \hat{I}(x_i, \mathbf{Pa}_{x_i,\mathcal{G}}) - M \sum_i \hat{H}(X_i)$$

Chow-Liu tree learning algorithm 1

- For each pair of variables X_i,X_j
  - Compute empirical distribution:

$$\hat{P}(x_i, x_j) = \frac{\text{Count}(x_i, x_j)}{m}$$

  - Compute mutual information:

$$\hat{I}(X_i, X_j) = \sum_{x_i, x_j} \hat{P}(x_i, x_j) \log \frac{\hat{P}(x_i, x_j)}{\hat{P}(x_i)\hat{P}(x_j)}$$

  - Nodes $X_1,\ldots,X_n$
  - Edge (i,j) gets weight

$$\hat{I}(X_i, X_j)$$

Chow-Liu tree learning algorithm 2

$$\log \hat{P}(\mathcal{D} \mid \theta, \mathcal{G}) = M \sum_i \hat{I}(x_i, \mathbf{Pa}_{x_i,\mathcal{G}}) - M \sum_i \hat{H}(X_i)$$

- Optimal tree BN
  - Compute maximum weight spanning tree
  - Directions in BN: pick any node as root, breadth-first-search defines directions

Can we extend Chow-Liu 1

- Tree augmented naïve Bayes (TAN) [Friedman et al. '97]
  - Naïve Bayes model overcounts, because correlation between features not considered
  - Same as Chow-Liu, but score edges with:

$$\hat{I}(X_i, X_j \mid C) = \sum_{c, x_i, x_j} \hat{P}(c, x_i, x_j) \log \frac{\hat{P}(x_i, x_j \mid c)}{\hat{P}(x_i \mid c)\hat{P}(x_j \mid c)}$$

Can we extend Chow-Liu 2

- (Approximately learning) models with tree-width up to *k*
  - [Chechetka & Guestrin '07]
  - But, $O(n^{2k+6})$…

What you need to know about learning BN structures so far

- Decomposable scores
  - Maximum likelihood
  - Information theoretic interpretation
- Best tree (Chow-Liu)
- Best TAN
- Nearly best k-treewidth (in $O(N^{2k+6})$)

Scoring general graphical models – Model selection problem

What's the best structure?

$\langle x_1^{(1)},\ldots,x_n^{(1)}\rangle$
…
$\langle x_1^{(m)},\ldots,x_n^{(m)}\rangle$

The more edges, the fewer independence assumptions, the higher the likelihood of the data, but will overfit…

Maximum likelihood overfits!

$$\log \hat{P}(\mathcal{D} \mid \theta, \mathcal{G}) = M \sum_i \hat{I}(x_i, \mathbf{Pa}_{x_i,\mathcal{G}}) - M \sum_i \hat{H}(X_i)$$

- Information never hurts:

- Adding a parent always increases score!!!

Bayesian score avoids overfitting

- Given a structure, distribution over parameters

$$\log P(D \mid \mathcal{G}) = \log \int_{\theta_\mathcal{G}} P(D \mid \mathcal{G}, \theta_\mathcal{G}) P(\theta_\mathcal{G} | \mathcal{G}) d\theta_\mathcal{G}$$

- Difficult integral: use Bayes information criterion (BIC) approximation (equivalent as M! 1)

$$\log P(D \mid \mathcal{G}) \approx \log P(D \mid \mathcal{G}, \theta_\mathcal{G}) - \frac{\text{NumberParams}(\mathcal{G})}{2} \log M + \mathcal{O}(1)$$

- Note: regularize with MDL score
- Best BN under BIC still NP-hard

Structure learning for general graphs

- In a tree, a node only has one parent
- **Theorem**:
  - 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
  - Exploit score decomposition
  - (Quickly) Describe two heuristics that exploit decomposition in different ways

Learn BN structure using local search

Starting from Chow-Liu tree

Local search, possible moves:
- Add edge
- Delete edge
- Invert edge

Score using BIC

What you need to know about learning BNs

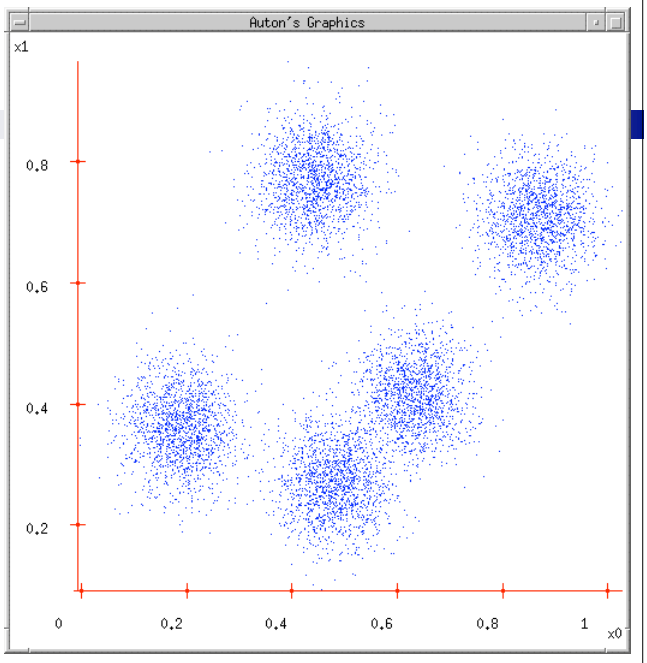
- Learning BNs
  - Maximum likelihood or MAP learns parameters
  - Decomposable score
  - Best tree (Chow-Liu)
  - Best TAN
  - Other BNs, usually local search with BIC score

Unsupervised Learning Clustering K-means

Machine Learning – 10701/15781
Carlos Guestrin
Carnegie Mellon University

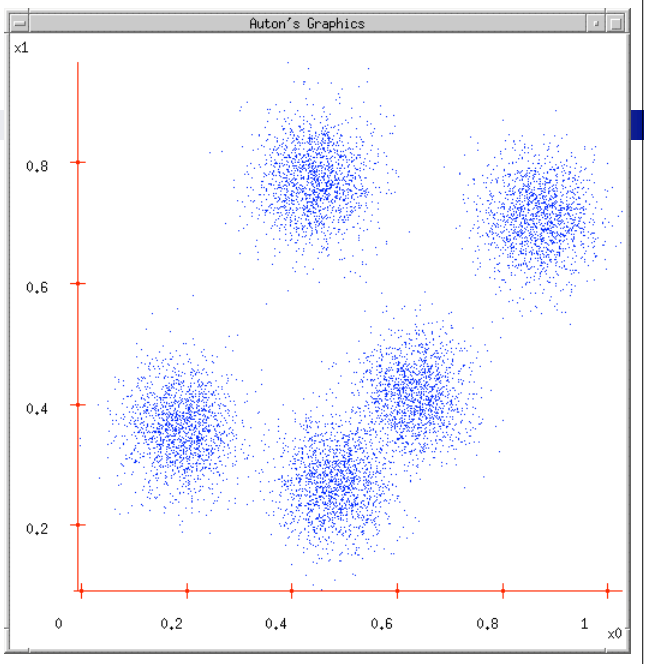
November 12th, 2007

Some Data



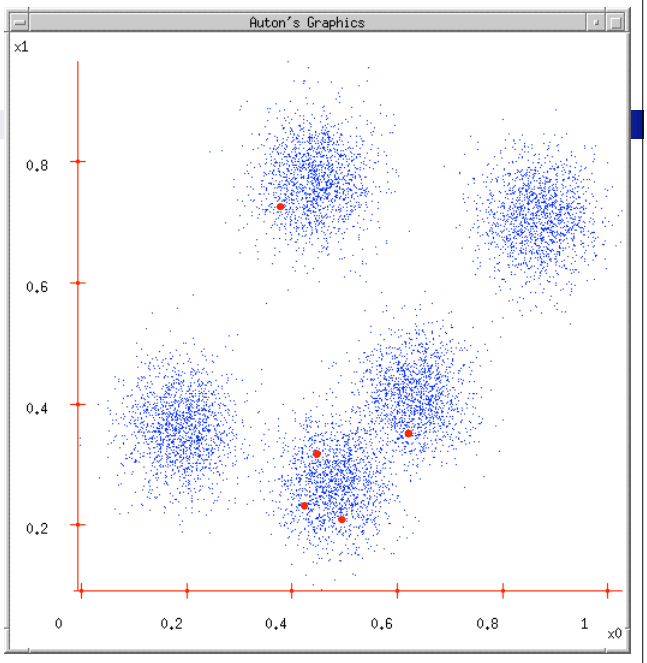
K-means

1. Ask user how many clusters they'd like. *(e.g. k=5)*



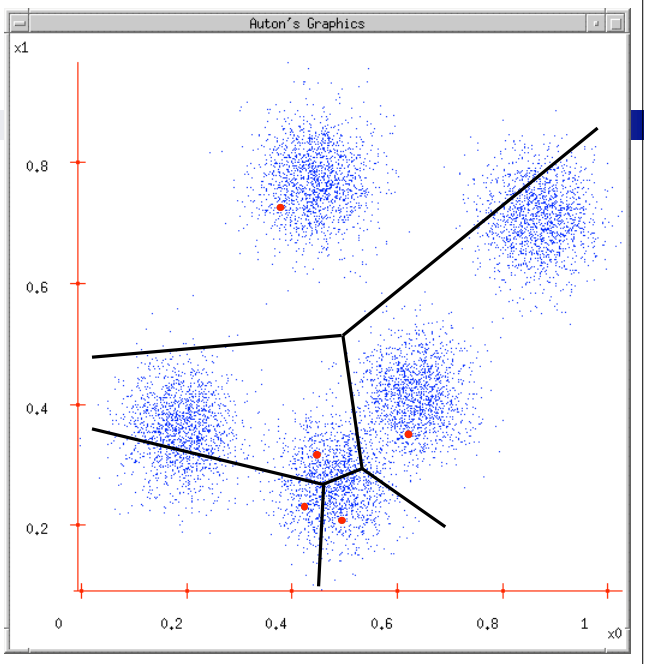
K-means

1. Ask user how many clusters they'd like. *(e.g. k=5)*
2. Randomly guess k cluster Center locations



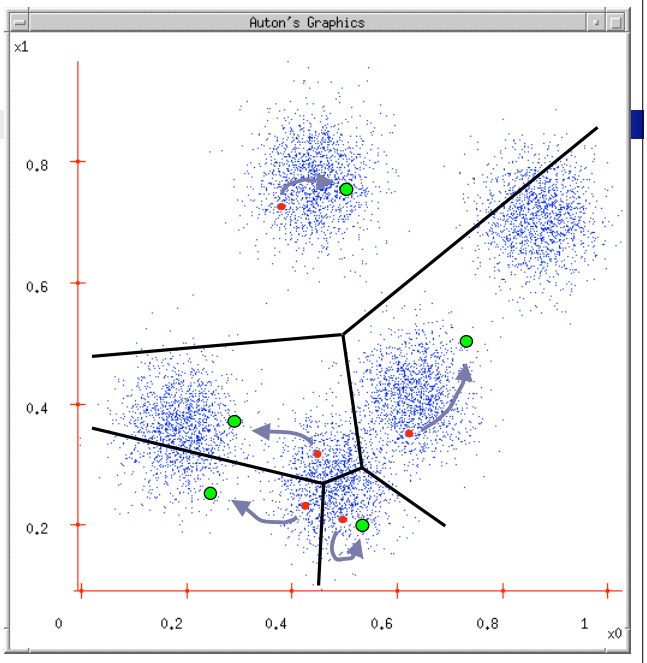
K-means

1. Ask user how many clusters they'd like. *(e.g. k=5)*
2. Randomly guess k cluster Center locations
3. Each datapoint finds out which Center it's closest to. (Thus each Center "owns" a set of datapoints)



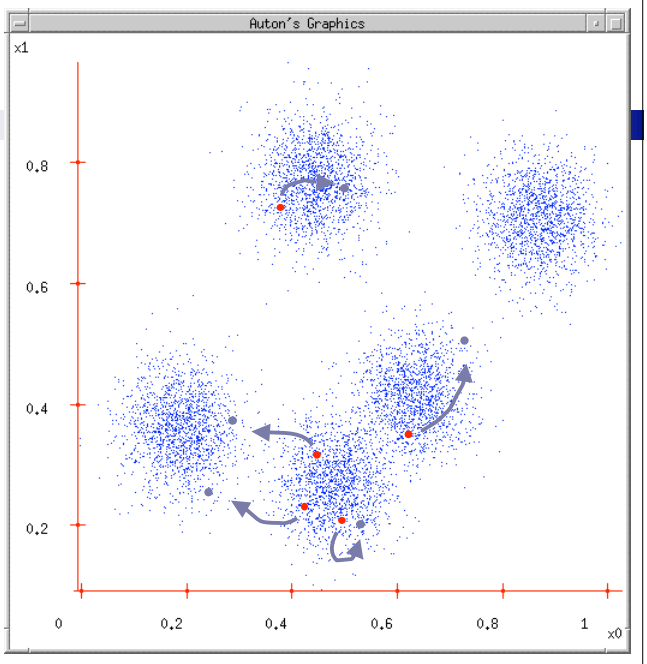
K-means

1. Ask user how many clusters they'd like. *(e.g. k=5)*
2. Randomly guess k cluster Center locations
3. Each datapoint finds out which Center it's closest to.
4. Each Center finds the centroid of the points it owns



K-means

1. Ask user how many clusters they'd like. *(e.g. k=5)*
2. Randomly guess k cluster Center locations
3. Each datapoint finds out which Center it's closest to.
4. Each Center finds the centroid of the points it owns…
5. …and jumps there
6. …Repeat until terminated!



K-means

- Randomly initialize *k* centers
  - $\mu^{(0)} = \mu_1^{(0)}, \ldots, \mu_k^{(0)}$
- **Classify**: Assign each point j∈{1,…m} to nearest center:
  - $C^{(t)}(j) \leftarrow \arg\min_i ||\mu_i - x_j||^2$
- **Recenter**: μ_i becomes centroid of its point:
  - $\mu_i^{(t+1)} \leftarrow \arg\min_\mu \sum_{j:C(j)=i} ||\mu - x_j||^2$
  - Equivalent to μ_i ← average of its points!

What is K-means optimizing?

- Potential function $F(\mu,C)$ of centers μ and point allocations C:
  - $F(\mu, C) = \sum_{j=1}^{m} ||\mu_{C(j)} - x_j||^2$
- Optimal K-means:
  - $\min_\mu \min_C F(\mu,C)$

Does K-means converge??? Part 1

- Optimize potential function:

$$\min_{\mu} \min_{C} F(\mu, C) = \min_{\mu} \min_{C} \sum_{i=1}^{k} \sum_{j:C(j)=i} ||\mu_i - x_j||^2$$

- Fix μ, optimize C

Does K-means converge??? Part 2

- Optimize potential function:

$$\min_{\mu} \min_{C} F(\mu, C) = \min_{\mu} \min_{C} \sum_{i=1}^{k} \sum_{j:C(j)=i} ||\mu_i - x_j||^2$$

- Fix C, optimize μ

Coordinate descent algorithms

$$\min_{\mu} \min_{C} F(\mu, C) = \min_{\mu} \min_{C} \sum_{i=1}^{k} \sum_{j:C(j)=i} ||\mu_i - x_j||^2$$

- Want: $\min_a \min_b F(a,b)$
- Coordinate descent:
  - fix a, minimize b
  - fix b, minimize a
  - repeat
- Converges!!!
  - if F is bounded
  - to a (often good) local optimum
    - as we saw in applet (play with it!)

- K-means is a coordinate descent algorithm!