Bayesian Networks (Structure) Learning

Machine Learning – 10701/15781

Carlos Guestrin

Carnegie Mellon University

November 12th, 2007

Information-theoretic interpretation of maximum likelihood 1

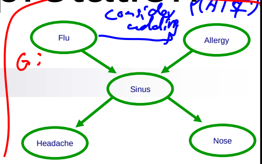
$\log \prod = \sum \log$

$\log ab = \log a + \log b$

- Given structure, log likelihood of data:

params, graph, iid

G:



consider adding $P(A|F)$

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

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

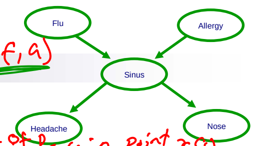
$$= \sum \log P(f^{(i)})\, P(a^{(i)}) \cdot P(s^{(i)} \mid f^{(i)}, a^{(i)}) \cdot P(h^{(i)} \mid s^{(i)})\, P(n^{(i)} \mid s^{(i)})$$

$$= \underbrace{\left[\sum_i \log P(f^{(i)})\right]}_{P(F)} + \underbrace{\left[\sum_i \log P(a^{(i)})\right]}_{P(A)} + \underbrace{\left[\sum_i \log P(s^{(i)} \mid f^{(i)}, a^{(i)})\right]}_{P(S|F,A)} + \underbrace{\left[\sum_i \log P(h^{(i)} \mid s^{(i)})\right]}_{P(H|S)} + \underbrace{\left[\sum_i \log P(n^{(i)} \mid s^{(i)})\right]}_{P(N|S)}$$

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

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

Information-theoretic interpretation of maximum likelihood 2



$P(s \mid f, a)$

- Given structure, log likelihood of data:

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

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

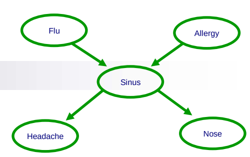
$$= \sum_{i=1}^{n} \sum_{j=1}^{m} \log P(X_i = x_i^{(j)} \mid Pa_{X_i} = x^{(j)}[Pa_{X_i}])$$

for points where $F=t, A=f, S=t$ [Count($F=t, A=f, S=t$)] $\log P(S=t \mid F=t, A=f)$

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

MLE estimate

Information-theoretic interpretation of maximum likelihood 3



- Given structure, log likelihood of data:

MLE

hat because MLE

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

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

MLE tries to pick parents s.t. $H(X_i \mid Pa_{X_i})$ small; little/no uncertainty about $X_i \mid Pa_{X_i}$

num of nodes

$$= m \sum_{i=1}^{n} \left[\hat{I}(X_i, Pa_{X_i}) - \hat{H}(X_i)\right]$$

number of points

$H(A \mid B)$

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

$I(A, B)$

$= \sum_{ab} P(a,b) \log \frac{P(a,b)}{P(a) P(b)}$

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

Decomposable score

- Log data likelihood

constants doesn't depend on G

MLE

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

- 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|$\mathbf{Pa}_{X_i}$: D)

graph

data

$$\text{FamScore}(X_i \mid Pa_{X_i} : D) = m\left[\hat{I}(X_i, Pa_{X_i}) - \hat{H}(X_i)\right]$$

How many trees are there?

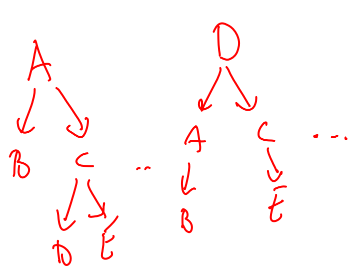
each node at most one parent

e.g. NB, HMMs, ...

Nonetheless – Efficient optimal algorithm finds best tree

$X = \{A, B, C, \ldots Z\}$

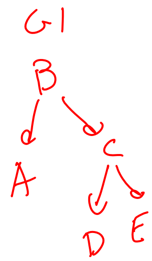
$O(2^{\Theta(n) \log n})$



Scoring a tree 1: equivalent trees

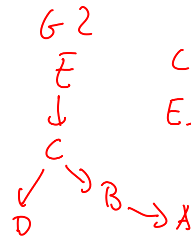
For trees only care about edges not the directions (no v-structures) Symm: $I(A,B) = I(B,A)$

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



$C \perp A \mid B$

$E \perp D \mid C$



$C \perp A \mid B$

$E \perp D \mid C$

$Score(G1) = I(A,B) + I(C,B) + I(C,D) + I(E,C)$

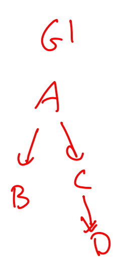
$Score(G2) = Score(G1)$

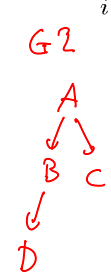
Cool!!

GRAPHS MAKE SAME IND. ASSUMPTIONS $\Leftrightarrow$ Same Score

Scoring a tree 2: similar trees

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





for trees T

$Score(T) = \sum_{(i,j) \in T} I(X_i, X_j)$

Score(G1):

$I(A,B) + I(A,C) + I(C,D)$

Score(G2):

$I(A,B) + I(A,C) + I(B,D)$

$= Score(G1) + I(B,D) - I(C,D)$

Chow-Liu tree learning algorithm 1

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

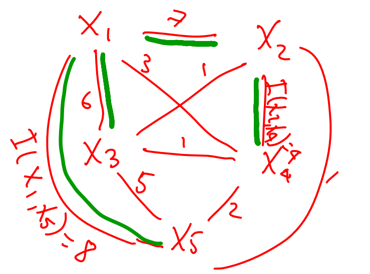
$$\hat{P}(x_i, x_j) = \frac{\text{Count}(x_i, x_j)}{m}$$

  - 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

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



Score (T_opt) = 8+6+7+4

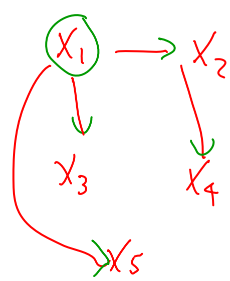
maximum weight Spanning
(Basic, very efficient alg)

Chow-Liu tree learning algorithm 2

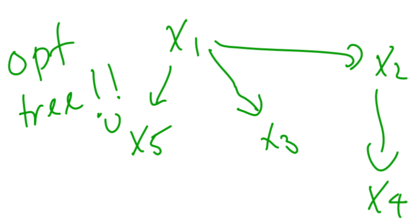
$$\log \hat{P}(\mathcal{D} \mid \theta, \mathcal{G}) = M \sum_i \hat{I}(x_i, \mathbf{Pa}_{x_i,\mathcal{G}}) - M \sum_i \hat{H}(X_i)$$

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

don't get V-structure



$O(n^3)$

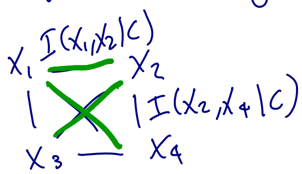


Can we extend Chow-Liu 1

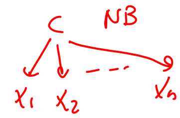
- Tree augmented naïve Bayes (TAN) [Friedman et al. '97]
  - Naïve Bayes model overcounts, because correlation between features not considered
  - Same as Chow-Liu, but score edges with:

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

$Score(X_i \mid X_j, C) = I(X_i, X_j \mid C)$



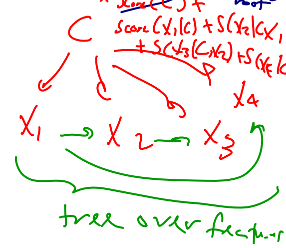
From HW1:



if features correlated doesn't do well

TAN: $Score(TAN_G) = \cancel{Score(C)} + Score(X_1 \mid C) + S(X_2 \mid C, X_1) + S(X_3 \mid C, X_2) + S(X_4 \mid C, X \ldots$

C always root



Can we extend Chow-Liu 2

(trees: treewidth = 1)

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

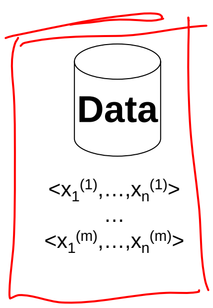
What you need to know about learning BN structures so far

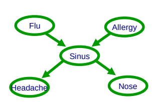
- Decomposable scores
  - Maximum likelihood
  - Information theoretic interpretation
- Best tree (Chow-Liu)
- Best TAN
- Nearly best k-treewidth (in $O(N^{2k+6})$)

Scoring general graphical models – Model selection problem

What's the best structure?

Score





→ 17

→ 19 > 17

fewer independencies
less bias, ⇒ fit data
fully connected!! (high variance) better (high MLE)

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: $I(A;\{B,C\}) \geq I(A;B)$

$\Rightarrow I(X_i ; Pa_{X_i})$ increases as $|Pa_{X_i}|$ increases

$\Rightarrow$ fully connected graph ML

- Adding a parent always increases score!!!

Bayesian score avoids overfitting

(encodes "regularization")

$Score(G) = m \sum_i Score(X_i \mid Pa_{X_i})$

- Given a structure, distribution over parameters

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

- Difficult integral: use Bayes information criterion (BIC) approximation (equivalent as ~~M→1~~) $M \to \infty$ constant

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

Bayesian Score (regularized)	data Likelihood	num of params (# of parents)
as a function of m	m	log m
Simple Graph	lower	lower
complete Graph	higher	higher

adding parents to X_i NumParams: increase exponentially

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

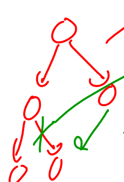
Structure learning for general graphs

- 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~~ heuristic~~s~~ that exploit decomposition ~~in different ways~~

Learn BN structure using local search

FamScore ~~is~~ includes regularization

Starting from Chow-Liu tree



Can start from other points

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

G, consider Adding edge $X_i \to X_j$ to generate G'

$$Score(G') - Score(G)$$

$$\underbrace{= FamScore(X_i \mid Pa_{X_i,G} \cup X_j) - FamScore(X_i \mid Pa_{X_i,G})}_{\text{local easy to compute}}$$

Score using BIC

17

19

20

⋮

Be Careful! don't create a cycle!

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

Supervised

X, y   x, +

x, -

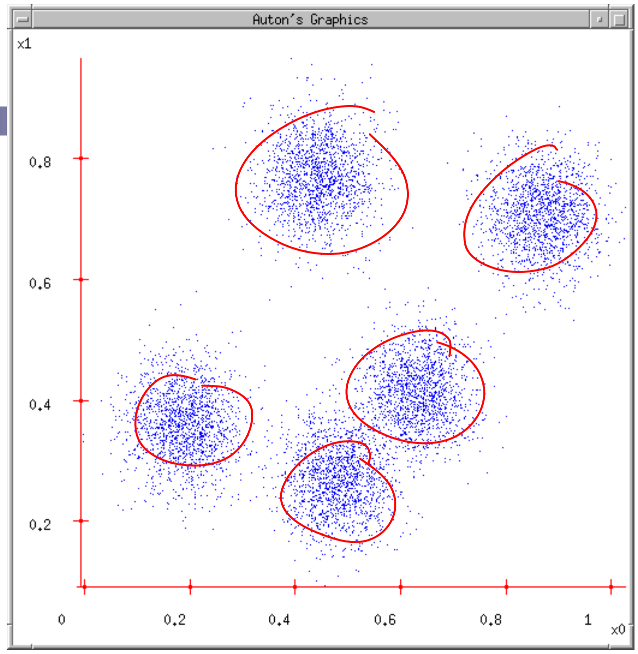
⋮

unsupervised

X ← "learn something"

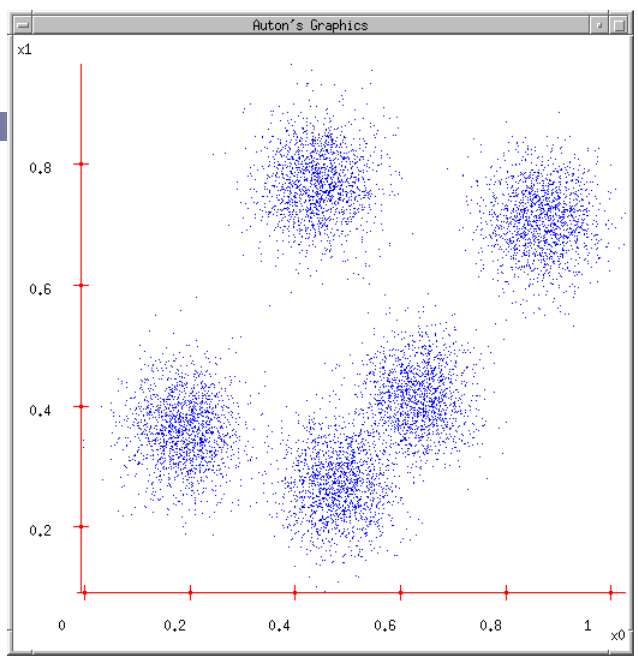
Hard Cluster

X ← split into 5 sets



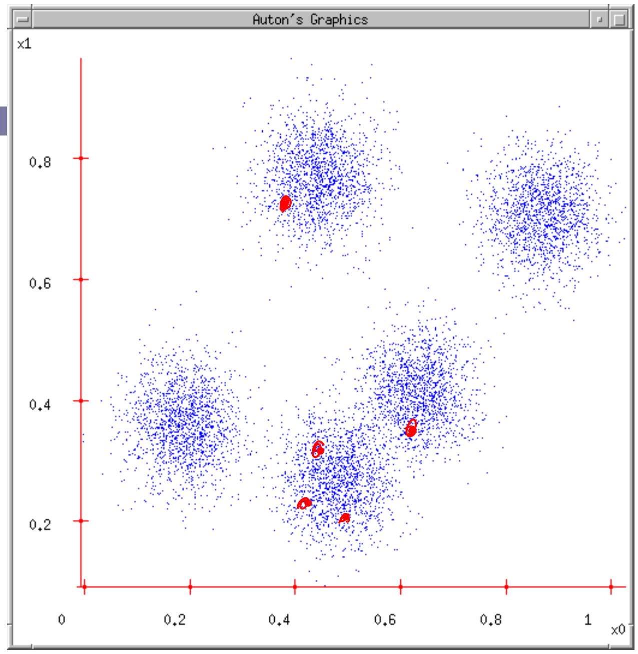
K-means

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



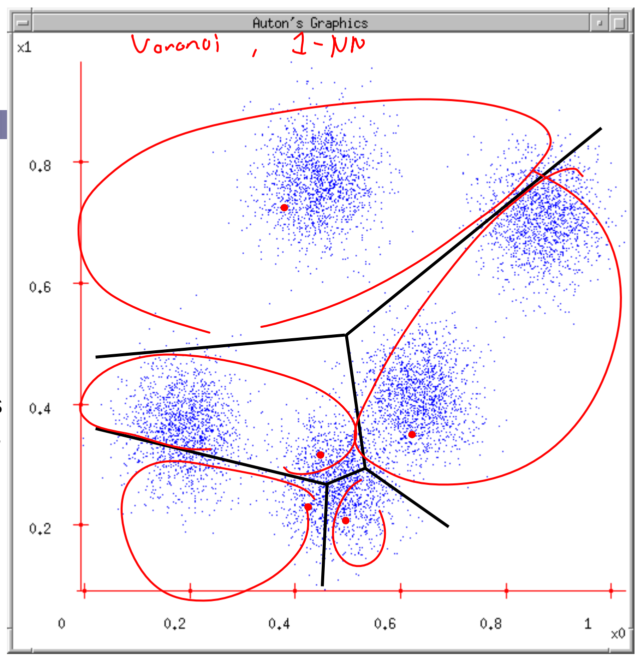
K-means

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



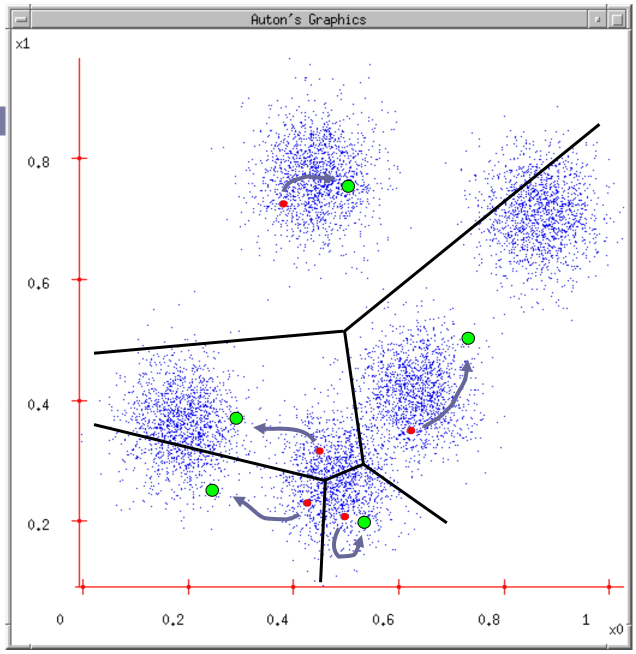
K-means

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)



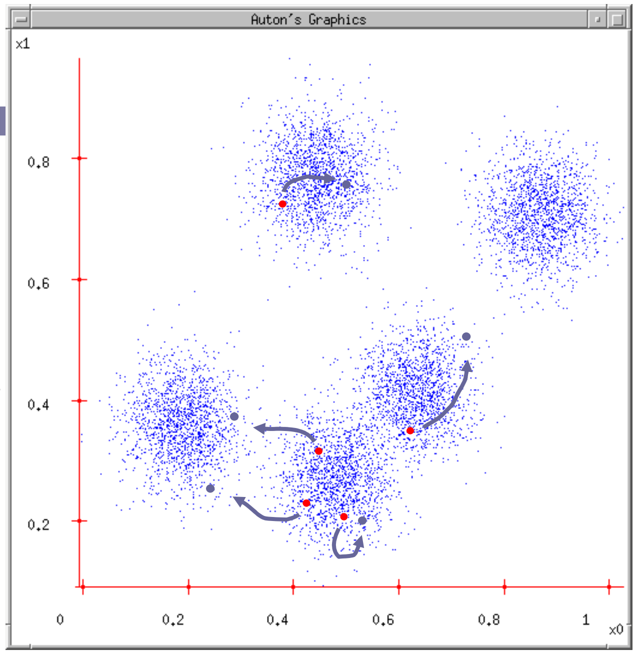
K-means

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 $j \in \{1, \ldots m\}$ to nearest center: center of point j is closest to j
  - $C^{(t)}(j) \leftarrow \arg\min_i ||\mu_i - x_j||^2$
- **Recenter**: μ_i becomes centroid of its point:
  - $\mu_i^{(t+1)} \leftarrow \arg\min_\mu \sum_{j:C(j)=i} ||\mu - x_j||^2$ opt $\mu_i = \dfrac{\sum_{j:C(j)=i} x_j}{\sum_{j:C(j)=i} 1}$ is the mean!!
  - Equivalent to $\mu_i \leftarrow$ average of its points!

What is K-means optimizing?

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

Does K-means converge??? Part 1

- Optimize potential function:

$$\min_{\mu}\min_{C} F(\mu, C) = \min_{\mu}\min_{C} \underbrace{\sum_{i=1}^{k}}_{\text{clusters}} \underbrace{\sum_{j:C(j)=i}}_{\text{points in cluster}} \underbrace{||\mu_i - x_j||^2}_{\text{distance}}$$

- Fix μ, optimize C

$\mu = \bar{\mu}$

$$\min_C \sum_{i=1}^{k} \sum_{j:C(j)=i} ||\bar{\mu}_i - x_j||^2$$

$$= \min_C \sum_{j=1}^{m} ||\bar{\mu}_{C(j)} - x_j||^2$$

optimize each center independently

$$= \sum_{j=1}^{m} \min_C ||\bar{\mu}_{C(j)} - x_j||^2 \Rightarrow \underbrace{C(j) = \arg\min_i ||\bar{\mu}_i - x_j||^2}_{\text{Classify Step}}$$

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

$C = \bar{C}$

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

$$= \sum_{i=1}^{k} \underbrace{\min_{\mu_i} \sum_{j:\bar{C}(j)=i} ||\mu_i - x_j||^2}_{\mu_i \text{ is mean of points in cluster } i}$$

$\Rightarrow$ re center in K-means

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!)

$F(a,b)$

$\forall_{a,b} F(a,b) \geq$ ___

- K-means is a coordinate descent algorithm!