EM ALGORITHM

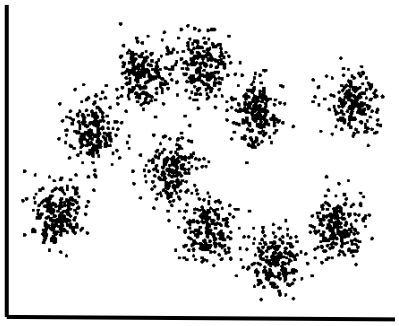
Today

- Reading
  - IR Chapter 16 (AIMA Chapter 20)
- Goals
  - Maximum likelihood estimation
  - Expectation Maximization algorithm
- Midterm quiz 2 is on Thursday
  - Same format as last quiz
  - Look at Tuesday's lecture slides for coverage information

Unsupervised Learning

- Learning without labels
  - Often comes down to clustering
  - Can be used as a surrogate for supervised learning

$$D = \begin{matrix} \mathbf{x}_1 \\ \mathbf{x}_2 \\ \\ \\ \mathbf{x}_N \end{matrix} \begin{bmatrix} x_{11} & x_{12} & x_{13} & \cdots & x_{1M} \\ x_{21} & x_{22} & x_{23} & \cdots & x_{2M} \\ & & & & \\ & & & & \\ x_{N1} & x_{N2} & x_{N3} & \cdots & x_{NM} \end{bmatrix}$$

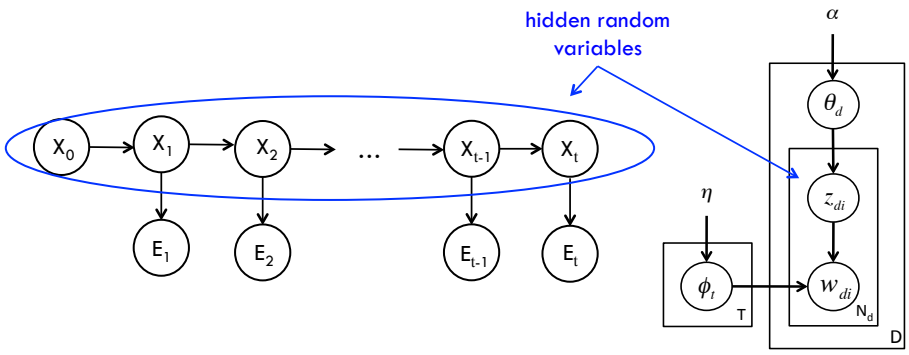


Types of clustering algorithms

- Flat versus Hierarchical
  - Flat algorithms return an unstructured set of clusters
  - Hierarchical algorithms return a hierarchy of clusters
- Sequential (online) versus Batch
  - Sequential algorithms are typically fast
- Hard versus soft
  - Hard algorithms make a hard assignment of elements to clusters
  - Soft algorithms compute a distribution over clusters for each element

Expectation Maximization

- EM is an iterative algorithm for performing maximum likelihood estimation when there are hidden (i.e. unobserved or latent) random variables



Expectation Maximization

- EM is an iterative algorithm for performing maximum likelihood estimation when there are hidden (i.e. unobserved or latent) random variables

- Q1: What is maximum likelihood estimation?
- Q2: What does maximum likelihood estimation look like when there are hidden variables?
- Q3: How does EM help?
- Q4: What does EM have to do with clustering?

Maximum Likelihood Estimation

- Maximum likelihood estimation is a particular type of probabilistic inference

  X = observed data

  θ = model parameters

- The likelihood of X is p(X| θ) viewed as a function of θ
- We want to find the value of θ that maximizes the likelihood of X, i.e. the value of θ that makes the observed data the most likely

$$\theta^{\text{MLE}} = \text{argmax}_\theta \; p(X|\theta)$$

Called the maximum likelihood estimate (MLE)

Maximum Likelihood Estimation

- Example: Suppose we observe a set of test scores. We make the assumption that our data is Normally distributed.

$$X = \{63, 77, 85, 81, 92, 93, 86\}$$
$$x_i \sim \text{Normal}(\mu, \sigma^2)$$
$$\theta = \{\mu, \sigma^2\}$$

$$p(X|\theta) = \prod_{i=1}^{N} p(x_i|\theta)$$
$$= \prod_{i=1}^{N} \frac{1}{\sigma\sqrt{2\pi}} \exp\left\{-\frac{(x_i-\mu)^2}{2\sigma^2}\right\}$$

$$\theta^{\text{MLE}} = \text{argmax}_\theta \; p(X|\theta)$$

Maximum Likelihood Estimation

- We can find θ^{MLE} by taking the derivative of the log likelihood with respect to θ and setting equal to 0

$$\frac{\partial\, p(X|\theta)}{\partial \mu} = 0 \implies \hat{\mu} = \frac{1}{N}\sum_{i=1}^{N} x_i \qquad \hat{\mu} = \frac{577}{7} = 82.43$$

$$\frac{\partial\, p(X|\theta)}{\partial \sigma^2} = 0 \implies \hat{\sigma}^2 = \frac{1}{N}\sum_{i=1}^{N} (x_i - \hat{\mu})^2 \qquad \hat{\sigma}^2 = 90.24$$

variance

82.43 ± 9.5

standard deviation

ML Estimation with hidden variables

- Sometimes our probabilistic model includes hidden variables

  X = observed data

  Z = unobserved data

  θ = model parameters

- Computing the likelihood of X now requires summing over Z

$$p(X|\theta) = \sum_{z} p(X, Z = z|\theta)$$

Can be computationally intractable

- The complete-data likelihood is given by

$$p(X, Z|\theta)$$

If we knew Z, life would be simpler

ML Estimation with hidden variables

- Example: Clustering!

$$X = \{x_1, x_2, \ldots, x_N\} \quad \text{where} \quad x_i \in \mathbb{R}^M \qquad \text{// the data points}$$
$$Z = \{z_1, z_2, \ldots, z_N\} \quad \text{where} \quad z_i \in \{1, 2\} \qquad \text{// unknown cluster assignments}$$
$$\theta = \{\alpha, \mu_1, \mu_2, \sigma_1, \sigma_2\} \qquad \text{// the model parameters}$$

- The probabilistic model:

$$z_i \sim \text{Discrete}(\alpha, 1-\alpha)$$
$$x_i | z_i \sim \text{Normal}(\mu_{z_i}, \sigma_{z_i})$$

$$p(X, Z|\theta) = \prod_{i=1}^{N} p(x_i|\mu_{z_i}, \sigma_{z_i}) \cdot p(z_i|\alpha)$$
$$p(X|\theta) = \prod_{i=1}^{N} \left(\sum_k p(x_i|\mu_k, \sigma_k) \cdot p(z_i = k|\alpha) \right)$$

Expectation Maximization

1. Initialize θ
2. Expectation (E-Step): Compute the Q function

$$Q(\theta|\theta^t) = E_{Z|X,\theta^t}\left[\log p(X, Z|\theta)\right]$$
$$= \sum_z p(Z = z|X, \theta^t) \cdot \log p(X, Z = z|\theta)$$

3. Maximization (M-Step): Maximize the Q function

$$\theta^{t+1} = \text{argmax}_\theta \; Q(\theta|\theta^t)$$

4. Repeat E-step and M-step till convergence

E-step

- Compute the expected value of the complete data log likelihood function w.r.t the conditional distribution of Z given X and the current guess of θ^t
  - We don't know Z. If we did know Z, the likelihood would be easy to compute!
  - Let's use the expected value of Z given X and θ^t. This is as good a guess as any (and better than most)!

$$Q(\theta|\theta^t) = E_{Z|X,\theta^t}\left[\log p(X, Z|\theta)\right]$$
$$= \sum_z p(Z = z|X, \theta^t) \cdot \log p(X, Z = z|\theta)$$

M-step

- Optimize this new Q function

$$\theta^{t+1} = \text{argmax}_\theta \; Q(\theta|\theta^t)$$

- We wanted to optimize p(X| θ) but in fact we're optimizing the Q function
- In fact, it can be shown that

$$\text{argmax}_\theta \; Q(\theta|\theta^t) \leq \text{argmax}_\theta \; \log p(X|\theta)$$

EM applied to clustering

- E-Step: Compute the Q function

$$
\begin{aligned}
Q(\theta|\theta^t) &= E[\log p(X, Z|\theta)] \\
&= E[\log \prod_i p(x_i|\mu_{z_i}, \sigma_{z_i})\ p(z_i|\alpha)] \\
&= \sum_i E[\log p(x_i|\mu_{z_i}, \sigma_{z_i}) + \log p(z_i|\alpha)] \\
&= \sum_i \sum_k \underbrace{p(z_i = k|x_i, \theta^t)} \cdot [\log p(x_i|\mu_k, \sigma_k) + \log p(z_i = k|\alpha)]
\end{aligned}
$$

Ultimately, it comes down to computing this quantity

$$
\begin{aligned}
\omega_k^i &= p(z_i = k|x_i, \theta^t) \\
&\propto p(x_i|\mu_k^t, \sigma_k^t) p(z_i = k|\alpha^t)
\end{aligned}
$$

EM applied to clustering

- M-Step: Maximize Q function w.r.t. θ

$$
\begin{aligned}
\alpha_k^{t+1} &= \frac{1}{N} \sum_i \omega_k^i \\
\mu_k^{t+1} &= \frac{\sum_i \omega_k^i x_i}{\sum_i \omega_k^i} \\
\sigma_k^{t+1} &= \frac{\sum_i (x_i - \mu_k^{t+1})^2}{\sum_i \omega_k^i}
\end{aligned}
$$

- Given updated values, recompute the Q function

EM applied to clustering

- Now let's change our probablistic model. Assume we want to cluster text data

$$X = \{x_1, x_2, \ldots, x_N\} \quad \text{where} \quad x_i \in [0,1]^M \quad \text{// the documents}$$
$$Z = \{z_1, z_2, \ldots, z_N\} \quad \text{where} \quad z_i \in \{1,2\} \quad \text{// unknown cluster assignments}$$
$$\theta = \{\alpha, q_1, q_2\} \quad \text{// the model parameters}$$

- The probabilistic model:

$$z_i \sim \text{Discrete}(\alpha, 1-\alpha)$$
$$x_i | z_i \sim \text{Multivariate Bern}(q_{z_i})$$
$$p(x_i | z_i = k) = \prod_{j=1}^{M} q_{km}^{x_i} \cdot (1 - q_{km})^{(1-x_i)}$$

EM applied to clustering

- E-step: Ultimately it comes down to computing

$$\omega_k^i = p(z_i = k | x_i, \theta^t)$$
$$\propto p(x_i | q_k^t) p(z_i = k | \alpha^t)$$

- M-step: Optimize the Q function

$$\alpha_k^{t+1} = \frac{1}{N} \sum_i \omega_k^i$$
$$q_{km}^{t+1} = \frac{\sum_i \omega_k^i \, \mathrm{I}(x_{im} = 1)}{\sum_i \omega_k^i}$$

Extra Slides

K-means Clustering

- The most well-known (widely-used) clustering algorithm
- Minimizes the sum of the squared distances of each vector from its centroid:

$$||\vec{x} - \mu_i||_2 = \left(\sum_{m=1}^{M} (x_m - \mu_{im})^2 \right)^{1/2}$$ ← Euclidean distance

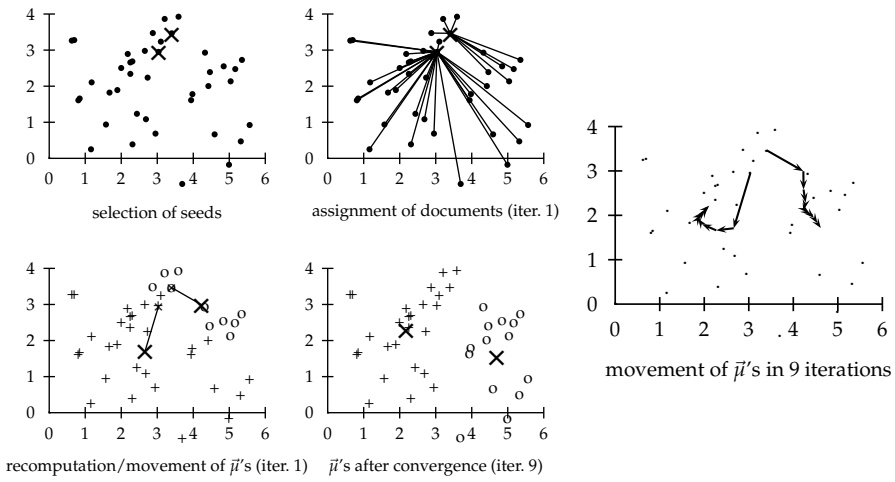
$$\text{RSS} = \sum_{k=1}^{K} \sum_{\vec{x} \in C_k} ||\vec{x} - \mu_k||_2^2$$ ← Residual sum of squares

- User must set K
- Assumes instances x represented as normalized vectors in a real-valued space

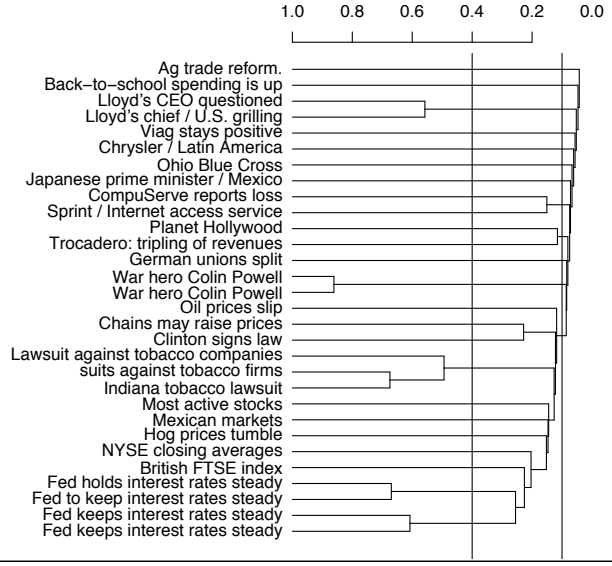
K-means Clustering

K-MEANS($\{\vec{x}_1, \ldots, \vec{x}_N\}, K$)
1 $(\vec{s}_1, \vec{s}_2, \ldots, \vec{s}_K) \leftarrow$ SELECTRANDOMSEEDS($\{\vec{x}_1, \ldots, \vec{x}_N\}, K$)
2 **for** $k \leftarrow 1$ **to** K
3 **do** $\vec{\mu}_k \leftarrow \vec{s}_k$
4 **while** stopping criterion has not been met
5 **do for** $k \leftarrow 1$ **to** K
6 **do** $\omega_k \leftarrow \{\}$
7 **for** $n \leftarrow 1$ **to** N
8 **do** $j \leftarrow \arg\min_{j'} |\vec{\mu}_{j'} - \vec{x}_n|$
9 $\omega_j \leftarrow \omega_j \cup \{\vec{x}_n\}$ *(reassignment of vectors)*
10 **for** $k \leftarrow 1$ **to** K
11 **do** $\vec{\mu}_k \leftarrow \frac{1}{|\omega_k|} \sum_{\vec{x} \in \omega_k} \vec{x}$ *(recomputation of centroids)*
12 **return** $\{\vec{\mu}_1, \ldots, \vec{\mu}_K\}$

K-means Clustering



Agglomerative Clustering



Agglomerative Clustering

SIMPLEHAC$(d_1, \ldots, d_N)$

```
1  for n ← 1 to N
2  do for i ← 1 to N
3     do C[n][i] ← SIM(d_n, d_i)
4     I[n] ← 1  (keeps track of active clusters)
5  A ← []  (assembles clustering as a sequence of merges)
6  for k ← 1 to N − 1
7  do ⟨i, m⟩ ← arg max_{⟨i,m⟩: i≠m ∧ I[i]=1 ∧ I[m]=1} C[i][m]
8     A.APPEND(⟨i, m⟩)  (store merge)
9     for j ← 1 to N
10    do C[i][j] ← SIM(i, m, j)
11       C[j][i] ← SIM(i, m, j)
12    I[m] ← 0  (deactivate cluster)
13 return A
```

Compute similarity matrix

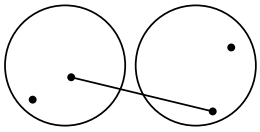
Find two clusters to merge

The i^{th} row and column of C are now the distances for the new cluster

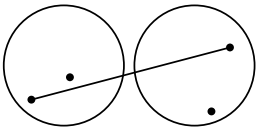
Similarity between cluster c_j and cluster $c_i \cup c_m$

► **Figure 17.2** A simple, but inefficient HAC algorithm.

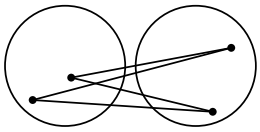
Agglomerative Clustering



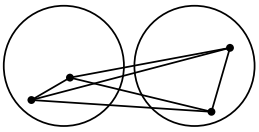
(a) single-link: maximum similarity



(b) complete-link: minimum similarity



(c) centroid: average inter-similarity



(d) group-average: average of all similarities