Lecture 14

Classic and Modern Data Clustering

Marina Meilă
University of Washington
mmp@stat.washington.edu

Outline

1. Paradigms for clustering
2. Parametric clustering algorithms (K given)
   - Cost based / hard clustering
   - Model based / soft clustering
3. Issues in parametric clustering
   - Selecting K
   - Outliers
4. Non-parametric clustering (smoothness given)
   - Based on non-parametric density estimation
   - Dirichlet process mixture models
5. Similarity based / graph clustering
   - Spectral clustering
   - Affinity propagation
6. Cluster validation
7. Special topics

What is clustering? Problem and Notation

- Informal definition Clustering = Finding groups in data
- Notation
  \[ D = \{x_1, x_2, \ldots, x_n\} \text{ a data set} \]
  \[ n = \text{number of data points} \]
  \[ K = \text{number of clusters } (K \ll n) \]
  \[ \Delta = \{C_1, C_2, \ldots, C_K\} \text{ a partition of } D \text{ into disjoint subsets} \]
  \[ k(i) = \text{the label of point } i \]
  \[ L(\Delta) = \text{cost (loss) of } \Delta \text{ (to be minimized)} \]
- Second informal definition Clustering = given n data points, separate them into K clusters
- Hard vs. soft clusterings
  - Hard clustering \( \Delta \): an item belongs to only 1 cluster
  - Soft clustering \( \gamma = \{\gamma_{ki}\}_{i=1}^n \text{ } k=1:K \)
    \[ \gamma_{ki} = \text{the degree of membership of point } i \text{ to cluster } k \]
    \[ \sum_k \gamma_{ki} = 1 \text{ for all } i \]
    (usually associated with a probabilistic model)
Paradigms

Depend on type of data, type of clustering, type of cost (probabilistic or not), and constraints (about \( K \), shape of clusters)

- **Data = vectors** \( \{ x_i \} \) in \( \mathbb{R}^d \)
  - **Parametric** Cost based [hard]
    - \( K \) known
  - **Non-parametric** Dirichlet process mixtures [soft]
    - \( K \) determined by algorithm
      - Information bottleneck [soft]
      - Modes of distribution [hard]
        - Gaussian blurring mean shift [Carreira-Perpinan, 2007] [hard]
- **Data = similarities** between pairs of points \( S_{ij} \) \( i, j = 1:n \)
  - **Similarity based clustering**
    - Graph partitioning spectral clustering [hard, \( K \) fixed, cost based]
    - Affinity propagation typical cuts [hard non-parametric, cost based]

Outline

1. Paradigms for clustering
2. Parametric clustering algorithms (\( K \) given)
   - Cost based / hard clustering
   - Model based / soft clustering
3. Issues in parametric clustering
   - Selecting \( K \)
   - Outliers
4. Non-parametric clustering (smoothness given)
   - Based on non-parametric density estimation
   - Dirichlet process mixture models
5. Similarity based / graph clustering
   - Spectral clustering
   - Affinity propagation
6. Cluster validation
7. Special topics

Classification vs Clustering

<table>
<thead>
<tr>
<th></th>
<th>Classification</th>
<th>Clustering</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Cost (or Loss) ( \mathcal{L} )</strong></td>
<td>Expected error</td>
<td>many! (probabilistic or not)</td>
</tr>
<tr>
<td>Supervised</td>
<td>Unsupervised</td>
<td></td>
</tr>
<tr>
<td><strong>Generalization</strong></td>
<td>Performance on new data is what matters</td>
<td>Performance on current data is what matters</td>
</tr>
<tr>
<td>( K ) Known</td>
<td>Unknown</td>
<td></td>
</tr>
<tr>
<td><strong>&quot;Goal&quot;</strong></td>
<td>Prediction</td>
<td>Exploration</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Loss of data to explore!</td>
</tr>
<tr>
<td><strong>Stage of field</strong></td>
<td>Mature</td>
<td>Still young</td>
</tr>
</tbody>
</table>

Outline:

1. Paradigms for clustering
2. Parametric clustering algorithms (\( K \) given)
   - Cost based / hard clustering
   - Model based / soft clustering
3. Issues in parametric clustering
   - Selecting \( K \)
   - Outliers
4. Non-parametric clustering (smoothness given)
   - Based on non-parametric density estimation
   - Dirichlet process mixture models
5. Similarity based / graph clustering
   - Spectral clustering
   - Affinity propagation
6. Cluster validation
7. Special topics
Parametric clustering algorithms

- Cost based
  - Single linkage (min spanning tree)
  - Min diameter
    - Fastest first traversal (HS initialization)
  - K-medians
  - K-means
- Model based (cost is derived from likelihood)
  - EM algorithm
  - "Computer science"/"Probably correct" algorithms

Single Linkage Clustering

**Algorithm Single-Linkage**

**Input** Data $D = \{x_i\}_{i=1}^n$, number clusters $K$

- Construct the Minimum Spanning Tree (MST) of $D$
- Delete the largest $K - 1$ edges

- **Cost** $L(\Delta) = - \min_{k,k'} \text{distance}(C_k, C_{k'})$
  where $\text{distance}(A, B) = \arg\min_{x \in A, y \in B} ||x - y||$

- Running time $O(n^2)$ one of the very few costs $L$ that can be optimized in polynomial time
- Sensitive to outliers!

Minimum diameter clustering

- **Cost** $L(\Delta) = \max_k \max_{i,j} \max_{\delta \in C_k} \left\| x_i - x_j \right\|$
  - Minimize the diameter of the clusters
  - Optimizing this cost is NP-hard

- **Algorithms**
  - **Fastest First Traversal** [Hochbaum and Shmoys, 1985] – a factor 2 approximation for the min cost
  
  For every $D$, FFT produces a $\Delta$ so that
  
  $$L^{\text{opt}} \leq L(\Delta) \leq 2L^{\text{opt}}$$

  - rediscovered many times
Algorithm Fastest First Traversal

Input Data \( D = \{ x_i \}_{i=1:n} \), number clusters \( K \)

defines centers \( \mu_{1:K} \in D \)

- pick \( \mu_1 \) at random from \( D \)
- for \( k = 2 : K \)
  \( \mu_k \leftarrow \arg \max_D \text{distance}(x_i, \{ \mu_{1:k-1} \}) \)
- for \( i = 1 : n \) (assign points to centers)
  \( k(i) = k \) if \( \mu_k \) is the nearest center to \( x_i \)

K-medians clustering

- Cost \( L(\Delta) = \sum_i \sum_{\mu_k} i \in C_k ||x_i - \mu_k|| \) with \( \mu_k \in D \)
  - (usually) assumes centers chosen from the data points (analogy to median)
  
  Ex: Show that in 1D \( \arg \min_j \sum_j |x_i - \mu| \) is the median of \( \{ x_i \} \)
  - optimizing this cost is NP-hard
  - has attracted a lot of interest in theoretical CS (general from called “Facility location”)

Integer Programming Formulation of K-medians

- Define \( d_{ij} = ||x_i - x_j|| \), \( u_{ij} = 1 \) iff point \( i \) in cluster with center \( x_j \) (0 otherwise), \( y_j = 1 \) iff point \( j \) is cluster center (0 otherwise)

\[
\begin{align*}
\min \quad & \sum_{ij} d_{ij} u_{ij} \\
\text{s.t.} \quad & \sum_i u_{ij} = 1 \quad \text{point } i \text{ is in exactly 1 cluster for all } i \\
& \sum_j y_j \leq k \quad \text{there are at most } k \text{ clusters} \\
& u_{ij} \leq y_j \quad \text{point } i \text{ can only belong to a center for all } i, j
\end{align*}
\]

Linear Programming Relaxation of K-medians

- Define \( d_{ij}, y_j, u_{ij} \) as before, but \( y_j, u_{ij} \in [0, 1] \)

\[
\begin{align*}
\text{(LP)} \quad \min \quad & \sum_{ij} d_{ij} u_{ij} \\
\text{s.t.} \quad & \sum_i u_{ij} = 1 \\
& \sum_j y_j \leq k \\
& u_{ij} \leq y_j
\end{align*}
\]

Algorithm K-Medians (variant of [Bradley and Mangasarian, 2005])

Input Data \( D = \{ x_i \}_{i=1:n} \), number clusters \( K \)

- Solve (LP) obtain fractionary “centers” \( y_{1:n} \) and “assignments” \( u_{1:n,1:n} \)
- Sample \( K \) centers \( \mu_1 \ldots \mu_K \) by
  - \( P[\mu_k = \text{point}] \propto y_j \) (without replacement)
- Assign points to centers (deterministically)
  \( k(i) = \arg \min_k ||x_i - \mu_k|| \)

Guarantees (Agarwal)

- Given tolerance \( \varepsilon, \) confidence \( \delta \), \( K' = K(1 + \frac{1}{\varepsilon}) \ln \frac{n}{\delta} \), \( \Delta_K' \) obtained by K-medians with \( K' \) centers

\[
L(\Delta_K') \leq (1 + \varepsilon)L_K^{\text{opt}}
\]
K-means clustering

This is originally an algorithm for vector quantization [Lloyd, 1982]

Algorithm K-Means

Input  Data $D = \{x_i\}_{i=1:n}$, number clusters $K$

Initial centers $\mu_1, \mu_2, \ldots, \mu_K \in \mathbb{R}^d$ at random

Iterate until convergence

1. for $i = 1 : n$ (assign points to clusters $\Rightarrow$ new clustering)
   
   
   $k(i) = \underset{k}{\arg\min} ||x_i - \mu_k||$

2. for $k = 1 : K$ (recalculate centers)
   
   
   $\mu_k = \frac{1}{|C_k|} \sum_{i \in C_k} x_i$

   (1)

Convergence

- if $\Delta$ doesn’t change at iteration $m$ it will never change after that

The K-means cost

$\mathcal{L}(\Delta) = \sum_{i=1}^{n} ||x_i - \mu_{k(i)}||^2 = \sum_{k=1}^{K} \sum_{i \in C_k} ||x_i - \mu_k||^2$

- "least-squares" cost (also called distortion)

Proposition The K-means algorithm decreases $\mathcal{L}(\Delta)$ at every step.

Sketch of proof

- step 1: reassigning the labels can only decrease $\mathcal{L}$ Ex: show this
- step 2: reassigning the centers $\mu_k$ can only decrease $\mathcal{L}$
  
  because $\mu_k$ as given by (1) is the solution to

  $\mu_k = \min_{\mu \in \mathbb{R}^d} \sum_{i \in C_k} ||x_i - \mu||^2$ Ex: show this

(2)

Therefore, K-means converges to a local minimum of the cost $\mathcal{L}$

Initialization matters (see later)

Equivalent and similar cost functions

- The distortion can also be expressed using intracluster distances

  $\mathcal{L}(\Delta) = \frac{1}{2} \sum_{k=1}^{K} \frac{1}{n_k} \sum_{i,j \in C_k} ||x_i - x_j||^2$

  (3)

- Correlation clustering is defined as optimizing the related criterion

  $\mathcal{L}(\Delta) = \frac{1}{2} \sum_{k=1}^{K} \sum_{i,j \in C_k} ||x_i - x_j||^2$

  - This cost is equivalent to the (negative) sum of (squared) intercluster distances

  $\mathcal{L}(\Delta) = -\frac{1}{2} \sum_{k=1}^{K} \sum_{i \in C_k} \sum_{j \notin C_k} ||x_i - x_j||^2 + \text{constant}$

  (4)

Proof of (3)

Replace $\mu_k$ as expressed in (1) in the expression of $\mathcal{L}$, then rearrange the terms

Proof of (4)

$\sum_k \sum_{i,j \in C_k} ||x_i - x_j||^2 = \sum_{i,j} ||x_i - x_j||^2 - \sum_k \sum_{i \in C_k} \sum_{j \notin C_k} ||x_i - x_j||^2$

Symmetries between costs

- K-means cost $\mathcal{L}(\Delta) = \min_{\mu_{1:k}} \sum_{k} \sum_{i \in C_k} ||x_i - \mu_k||^2$
- K-medians cost $\mathcal{L}(\Delta) = \min_{\mu_{1:k}} \sum_{k} \sum_{i \in C_k} ||x_i - \mu_k||$

- K-means cost $\mathcal{L}(\Delta) = \sum_k \sum_{i,j \in C_k} ||x_i - x_j||^2$
- min diameter cost $\mathcal{L}^2(\Delta) = \max_k \max_{i,j \in C_k} ||x_i - x_j||^2$
K-means: Practical issues I

- **Initialization** of $\mu_{1:K}$.
  - **The Power Initialization** (see also [Bubeck et al., 2009])
    - pick $\mu_{1:K'}^0$ at random from data set, where $K' = O(K \log K)$
      (this assures that each cluster has at least 1 center w.h.p)
    - run 1 step of K-means
    - remove all centers $\mu_k^0$ that have few points, e.g $|C_k| < \frac{n}{eK'}$
    - from the remaining centers select $K$ centers by **Fastest First Traversal**
      - pick $\mu_1$ at random from the remaining $\{\mu_{1:K'}^0\}$
      - for $k = 2 : K$, $\mu_k \leftarrow \arg \max \min_{j=1:k-1} ||\mu_k^0 - \mu_j||$, i.e next $\mu_k$
        is furthest away from the already chosen centers
    - continue with the standard K-means algorithm

K-means: Practical issues II

This initialization has been shown experimentally and theoretically to work well.
- More precisely $K' = \tilde{K}(\ln \tilde{K} + \ln \frac{1}{\delta})$ where $\tilde{K} = n/\text{(size of smallest cluster)}$ and e.g $\delta = 0.05$, $1 - \delta =$ desired level of confidence

Ex: Find an (approximate) formula for $P[\text{sample all } C_k | K']$ as a function of $K'$ when there are $K$ clusters and $p_k = |C_k|/n$ is the probability of sampling from cluster $k$. Simplify by taking $p_k = 1/K$ for all $K$. Plot the function obtained and show that $K' = K$ is inappropriate.

- **Preprocessing**
  - centering $x_i \leftarrow x_i - \frac{\sum_i x_i}{n}$ (not essential but numerically useful)
  - scaling of different coordinates affects algorithms’ outcome!

Model based clustering: Mixture models

- The mixture density
  $$f(x) = \sum_{k=1}^{K} \pi_k f_k(x) \quad \text{with } \pi_k \geq 0, \sum_{k=1}^{K} \pi_k = 1$$  \hspace{1cm} (5)
- $f_k(x)$ = the components of the mixture
  - each is a density
  - if $f_k = \text{Normal}_{\mu_k, \Sigma_k}$ we call it a mixture of Gaussians
  - will assume $f_k$ Gaussian for simplicity
- $\pi_k$ = the mixing coefficients/mixing proportions (a convex combination)
- A probabilistic model for clustering
- Degree of membership
  $$\gamma_{ki} \overset{\text{def}}{=} P[x_i \in C_k] = \frac{\pi_k f_k(x)}{f(x)} \quad \text{for } i = 1 : n, k = 1 : K$$  \hspace{1cm} (6)

Coresets approach to K-medians and K-means

- A weighted subset of $D$ is a $(K, \varepsilon)$ coreset iff for any $\mu_{1:K}$,
  $$|\mathcal{L}(\mu_{1:K}, A) - \mathcal{L}(\mu_{1:K}; D)| \leq \varepsilon |\mathcal{L}(\mu_{1:K}; D)|$$
- Note that the size of $A$ is not $K$.
- Finding a coreset (fast) lets use find fast algorithms for clustering a large $D$
  - “fast” = linear in $n$, exponential in $\varepsilon^{-d}$, polynomial in $K$
- **Theorem** [Har-Peled and Mazumdar, 2004], Theorem 5.7
  One can compute an $(1 + \varepsilon)$-approximate K-median of a set of $n$ points in time $O(n + K^5 \log^5 n + gK^2 \log^5 n)$ where $g = e^{C/\varepsilon \log(1+1/\varepsilon)/\varepsilon}$ (where $d$ is the dimension of the data)
- **Theorem** [Har-Peled and Mazumdar, 2004], Theorem 6.5
  One can compute an $(1 + \varepsilon)$-approximate K-means of a set of $n$ points in time $O(n + K^5 \log^9 n + K^{K+2} \varepsilon^{-2d-1} \log^{K+1} n \log^{K+\frac{1}{2}})$. 

Degree of membership
  $$\gamma_{ki} \overset{\text{def}}{=} P[x_i \in C_k] = \frac{\pi_k f_k(x)}{f(x)} \quad \text{for } i = 1 : n, k = 1 : K$$  \hspace{1cm} (6)
The Maximum Likelihood Principle

- Given data $D = \{x_1:n\}$ sampled i.i.d from some unknown $P^*$
- Model $P_\theta(x)$ depends on parameter $\theta$
- Problem: How to estimate $\theta$?

Principle: Maximum Likelihood

\[
\text{Likelihood}(\theta|D) = P_\theta(D) = \prod_{i=1}^{n} P_\theta(x_i)
\]

- Often convenient to use log-likelihood $l(\theta)$

\[
l(\theta) = \sum_{i=1}^{n} \ln P_\theta(x_i)
\]

Reason: many $P_\theta$ are expressed with exponential functions (e.g. the Normal distribution)

Criteria for clustering: Max likelihood

- denote $\theta = (\pi_1:K, \mu_1:K, \Sigma_1:K)$ (the parameters of the mixture model)
- Define likelihood $P[D|\theta] = \prod_{i=1}^{n} f(x_i)$
- Typically, we use the log likelihood

\[
l(\theta) = \ln \prod_{i=1}^{n} f(x_i) = \sum_{i=1}^{n} \ln \sum_{k} \pi_k f_k(x_i)
\]

- denote $\theta^{ML} = \arg\max_{\theta} l(\theta)$
- $\theta^{ML}$ determines a soft clustering $\gamma$
- a soft clustering $\gamma$ determines a $\theta$ (see later)
- Therefore we can write

$$L(\gamma) = -l(\theta(\gamma))$$

Algorithms for model-based clustering

Maximize the (log-)likelihood w.r.t $\theta$

- directly - (e.g. by gradient ascent in $\theta$)
- by the EM algorithm (very popular!)
- indirectly, w.h.p. by "computer science" algorithms

w.h.p = with high probability (over data sets)
The Expectation-Maximization (EM) Algorithm

Algorithm Expectation-Maximization (EM)

**Input** Data $D = \{ x_i \}_{i=1:n}$, number clusters $K$

**Initialize** parameters $\pi_{1:K} \in \mathbb{R}$, $\mu_{1:K} \in \mathbb{R}^d$, $\Sigma_{1:K} \in \mathbb{R}^{d \times d}$ at random

**Iterate** until convergence

_E step_ (Optimize clustering) for $i = 1 : n$, $k = 1 : K$

\[
\gamma_{ki} = \frac{\pi_k f_k(x)}{f(x)}
\]

_M step_ (Optimize parameters)

let $\Gamma_k = \sum_{i=1}^{n} \gamma_{ki}$, $k = 1 : K$ (note: $\sum_k \Gamma_k = n$)

\[
\pi_k = \frac{\Gamma_k}{n}, \quad k = 1 : K
\]

\[
\mu_k = \sum_{i=1}^{n} \frac{\gamma_{ki}}{\Gamma_k} x_i
\]

\[
\Sigma_k = \frac{\sum_{i=1}^{n} \gamma_{ki} (x_i - \mu_k)(x_i - \mu_k)^T}{\Gamma_k}
\]

The Expectation-Maximization (EM) Algorithm

If $\theta$ known, $\gamma_{ki}$ can be obtained by (6) (Expectation)

If $\gamma_{ki}$ known, $\pi_k, \mu_k, \Sigma_k$ can be obtained by separately maximizing the terms of $E[l_c]$ (Maximization)

The EM Algorithm – Motivation

- Define the indicator variables
  \[
  z_{ik} = \begin{cases} 
  1 & \text{if } i \in C_k \\
  0 & \text{if } i \notin C_k
  \end{cases}
  \]

- Denote $\bar{z} = \{ z_{ik} \}_{k=1:K}$

- Define the complete log-likelihood
  \[
  l_c(\theta, \bar{z}) = \sum_{i=1}^{n} \sum_{k=1}^{K} z_{ik} \ln \pi_k f_k(x_i)
  \]

- $E[z_{ki}] = \gamma_{ki}$

- Then
  \[
  E[l_c(\theta, \bar{z})] = \sum_{i=1}^{n} \sum_{k=1}^{K} E[z_{ki}] [\ln \pi_k + \ln f_k(x_i)]
  \]

- $Q(\theta, \gamma) = \sum_{i=1}^{n} \sum_{k=1}^{K} \gamma_{ki} \ln \pi_k f_k(x_i)$

- Brief analysis of EM

  - each step of EM increases $Q(\theta, \gamma)$
  - $Q$ converges to a local maximum
  - at every local maxi of $Q$, $\theta \leftrightarrow \gamma$ are fixed point
  - $Q(\theta^*, \gamma^*)$ local max for $Q \Rightarrow l(\theta^*)$ local max for $l(\theta)$
  - under certain regularity conditions $\theta \rightarrow \theta^{ML}$
    [McLachlan and Krishnan, 1997]
  - the E and M steps can be seen as projections [Neal and Hinton, 1998]
Probabilistic alternate projection view of EM [Neal and Hinton, 1998]

- Let $z_i$ = which gaussian generated $i$? (random variable), $X = (x_{1:n})$, $Z = (z_{1:n})$
- Redefine $Q$

$$Q(\hat{P}, \theta) = L(\theta) - KL(\hat{P}||P(Z|X, \theta))$$

where $P(X, Z|\theta) = \prod_i \prod_k P[z_i = k] P[x_i|\theta_k]$
$\hat{P}(Z)$ is any distribution over $Z$,
$KL(P(w)||Q(w)) = \sum_w P(w) \ln \frac{P(w)}{Q(w)}$ the Kullbach-Leibler divergence

Then,
- **E step** $\max \hat{P} Q \iff KL(\hat{P}||P(Z|X, \theta))$
- **M step** $\max \theta Q \iff KL(P(X|Z, \theta_{old})||P(X|\theta))$
- Interpretation: KL is “distance”, “shortest distance” = projection

The M step in special cases

- Note that the expressions for $\mu_k, \Sigma_k = \text{expressions for } \mu, \Sigma$ in the normal distribution, with data points $x_i$ weighted by $\gamma_{ki}$

**M step**

<table>
<thead>
<tr>
<th>General case</th>
<th>$\Sigma_k = \sum_{i=1}^n \gamma_{ki} (x_i - \mu_k)(x_i - \mu_k)^T$</th>
</tr>
</thead>
<tbody>
<tr>
<td>Same shape &amp; size clusters</td>
<td>$\Sigma = \sum_{k=1}^K \gamma_{ki} \Sigma_k$</td>
</tr>
<tr>
<td>Round clusters</td>
<td>$\Sigma_k = \sigma_k^2 I_d$</td>
</tr>
<tr>
<td>Round, same size clusters</td>
<td>$\Sigma_k = \sigma_k^2 \frac{1}{d} \sum_{i=1}^n</td>
</tr>
</tbody>
</table>

Ex: Prove the formulas above

- Note also that **K-means** is **EM** with $\Sigma_k = \sigma_k^2 I_d$, $\sigma_k^2 \to 0$

EM – Practical issues

- Initialization is important
  - Use **Power initialization** (with EM replacing K-means)
- Exact maximization in **M step** is not essential. Sufficient to increase $Q$. This is called **Generalized EM**
"Computer science" algorithms for mixture models

- Assume clusters well-separated
  - e.g. \( ||\mu_k - \mu_l|| \geq C \max(\sigma_k, \sigma_l) \)
  - with \( \sigma_k^2 = \max \text{eigenvalue}(\Sigma_k) \)
- true distribution is mixture
  - of Gaussians
  - of log-concave \( f_k \)'s (i.e. \( \ln f_k \) is concave function)
- then, w.h.p. \((n, K, d, C)\)
  - we can label all data points correctly
  - \( \Rightarrow \) we can find good estimate for \( \theta \)

Even with (S) this is not an easy task in high dimensions

Because \( f_k(\mu_k) \to 0 \) in high dimensions (i.e. there are few points from Gaussian \( k \) near \( \mu_k \))

Other "CS" algorithms I

- [Dasgupta, 2000] round, equal sized Gaussian, random projection
- [Arora and Kannan, 2001] arbitrary shaped Gaussian, distances
- [Achlioptas and McSherry, 2005] log-concave, principal subspace projection

Example Theorem (Achlioptas & McSherry, 2005) If data come from \( K \) Gaussians, \( n \gg K(d + \log K)/\pi_{\text{min}} \), and

\[
||\mu_k - \mu_l|| \geq 4\sigma_k \sqrt{1/\pi_k + 1/\pi_l} + 4\sigma_k \sqrt{K \log nK + K^2}
\]

then, w.h.p. \( 1 - \delta(d, K, n) \), their algorithm finds true labels

Good

- theoretical guarantees
- no local optima
- suggest heuristics for EM K-means
  - project data on principal subspace (when \( d \gg K \))

The Vempala-Wang algorithm [Vempala and Wang, 2004]

Idea

Let \( \mathcal{H} = \text{span}(\mu_1, K) \)

Projecting data on \( \mathcal{H} \)

\( \approx \) preserves \( ||x_i - x_j|| \) if \( k(i) \neq k(j) \)

\( \approx \) reduces \( ||x_i - x_j|| \) if \( k(i) = k(j) \)

density at \( \mu_k \) increases

(Proved by Vempala & Wang, 2004[Vempala and Wang, 2004]) \( \mathcal{H} \approx K\)-th principal subspace of data

Algorithm Vempala-Wang (sketch)

- Project points \( \{x_i\} \in \mathbb{R}^d \) on \( K-1 \)-th principal subspace \( \Rightarrow \{y_i\} \in \mathbb{R}^K \)
- do distance-based "harvesting" of clusters in \( \{y_i\} \)

Other "CS" algorithms II

But

- strong assumptions: large separation (unrealistic), concentration of \( f_k \)'s (or \( f_k \) known), \( K \) known
- try to find perfect solution (too ambitious)
A fundamental result

The Johnson-Lindenstrauss Lemma For any \( \varepsilon \in (0, 1) \) and any integer \( n \), let \( d' \) be a positive integer such that \( d' \geq 4(\varepsilon^2/2 - \varepsilon^3/3)^{-1} \ln n \). Then for any set \( D \) of \( n \) points in \( \mathbb{R}^d \), there is a map \( f : \mathbb{R}^d \to \mathbb{R}^{d'} \) such that for all \( u, v \in V \),

\[
(1 - \varepsilon)||u - v||^2 \leq ||f(u) - f(v)||^2 \leq (1 + \varepsilon)||u - v||^2 \tag{12}
\]

Furthermore, this map can be found in randomized polynomial time.

- note that the embedding dimension \( d' \) does not depend on the original dimension \( d \), but depends on \( n, \varepsilon \)
- \[\text{Dasgupta and Gupta, 2002}\] show that: the mapping \( f \) is linear and that w.p. \( 1 - \frac{1}{n} \) a random projection (rescaled) has this property
- their proof is elementary

Projecting a fixed vector \( v \) on a a random subspace is the same as projecting a random vector \( v \) on a fixed subspace. Assume \( v = [v_1, \ldots, v_d] \) with \( v \sim \text{i.i.d.} \) and let \( \tilde{v} = \text{projection of } v \text{ on axes } 1 : d' \). Then \( E[||v||^2] = d' E[|v|^2] \). The next step is to show that the variance of \( ||\tilde{v}||^2 \) is very small when \( d' \) is sufficiently large.

A two-step EM algorithm

[Dasgupta and Schulman, 2007]

Assumes \( K \) spherical gaussians, separation \( ||\mu_k^\text{true} - \mu_k^\text{true}|| \geq C \sqrt{d} \sigma_k \)
- Pick \( K' = \mathcal{O}(K \ln K) \) centers \( \mu_k^0 \) at random from the data
- Set \( \sigma_k^0 = \frac{d}{d} \min_{k \neq k'} ||\mu_k^0 - \mu_k^0||^2 \), \( \pi_k^0 = 1/K' \)
- Run one E step and one M step \( \Rightarrow \{\pi_k^1, \mu_k^1, \sigma_k^1\}_{k=1:K'} \)
- Compute “distances” \( d(\mu_k^1, \mu_k^1) = \frac{||\mu_k^1 - \mu_k^1||}{\sigma_k^1 - \sigma_k^1} \)
- Prune all clusters with \( \pi_k^1 \leq 1/4K' \)
- Run Fastest First Traversal with distances \( d(\mu_k^1, \mu_k^1) \) to select \( K \) of the remaining centers. Set \( \pi_k^1 = 1/K \).
- Run one E step and one M step \( \Rightarrow \{\pi_k^2, \mu_k^2, \sigma_k^2\}_{k=1:K} \)

Theorem For any \( \delta, \varepsilon > 0 \) if \( d \) large, \( n \) large enough, separation \( C \geq d^{1/4} \) the Two step EM algorithm obtains centers \( \mu_k \) so that

\[
||\mu_k - \mu_k^\text{true}|| \leq ||\text{mean}(\mu_k^\text{true}) - \mu_k^\text{true}|| + \varepsilon \sigma_k \sqrt{d}
\]
Experimental exploration [Srebro et al., 2006] III

- Practical limits vs theoretical limits
  - Figures from [Srebro et al., 2006]

Outline

1. Paradigms for clustering
   - Parametric clustering algorithms (K given)
     - Cost based / hard clustering
     - Model based / soft clustering

2. Issues in parametric clustering
   - Selecting K
   - Outliers

3. Non-parametric clustering (smoothness given)
   - Based on non-parametric density estimation
   - Dirichlet process mixture models

4. Similarity based / graph clustering
   - Spectral clustering
   - Affinity propagation

5. Cluster validation

6. Special topics

Selecting K

- Run clustering algorithm for \( K = K_{\text{min}} : K_{\text{max}} \)
  - obtain \( \Delta K_{\text{min}} : \cdots : \Delta K_{\text{max}} \) or \( \gamma_{K_{\text{min}}} : \cdots : \gamma_{K_{\text{max}}} \)
  - choose best \( \Delta K \) (or \( \gamma K \)) from among them
- Typically increasing \( K \Rightarrow \text{cost} \ L \) decreases
  - (\( L \) cannot be used to select \( K \))
- Need to "penalize" \( L \) with function of number parameters
Selecting $K$ for mixture models

The BIC (Bayesian Information) Criterion
- Let $\theta_K$ = parameters for $\gamma_K$
- Let $\#\theta_K$ = number independent parameters in $\theta_K$
  - E.g. for mixture of Gaussians with full $\Sigma_k$'s in $d$ dimensions
    \[ \#\theta_K = K - 1 + Kd + K(d - 1)/2 \]
- Define
  \[ BIC(\theta_K) = l(\theta_K) - \frac{\#\theta_K}{2} \ln n \]
- Select $K$ that maximizes $BIC(\theta_K)$
- Selects true $K$ for $n \to \infty$ and other technical conditions (e.g. parameters in compact set)

Number of Clusters vs. BIC EII (A), VII (B), EEI (C), VEI (D), EVI (E), VVI (F), EEE (G), EEV (H), VEV (I), VVV (J)

EEV, 8 Cluster Solution

Selecting $K$ for hard clusterings
- Based on statistical testing: the gap statistic (Tibshirani, Walther, Hastie, 2000)
- Stability methods

(From [Nugent and Meila, 2010])
The gap statistic

Idea

- for some cost $\mathcal{L}$ compare $\mathcal{L}(\Delta_K)$ with its expected value under a null distribution
  - choose null distribution to have no clusters
    - Gaussian (fit to data)
    - uniform with convex support
    - uniform over $K_0$ principal components of data
  - null value = $E_{P_0}[\mathcal{L}_{K,n}]$ the expected value of the cost of clustering $n$ points from $P_0$ into $K$ clusters
- the gap
  $$g(K) = E_{P_0}[\mathcal{L}_{K,n}] - \mathcal{L}(\Delta_K) = \mathcal{L}_0^0 - \mathcal{L}(\Delta_K)$$
- choose $K^*$ corresponding to the largest gap
- nice: it can also indicate that data has no clusters

Practicalities

- $\mathcal{L}_0^0 = E_{P_0}[\mathcal{L}_{K,n}]$ can rarely be computed in closed form (when $P_0$ very simple)
- otherwise, estimate $\mathcal{L}_0^0$ be Monte-Carlo sampling
  - i.e generate $B$ samples from $P_0$ and cluster them
- if sampling, variance $s_k^2$ of estimate $\mathcal{L}_k^0$ must be considered
- $s_k^2$ is also estimated from the samples
- selection rule: $K^*$ = smallest $K$ such that $g(K) \geq g(K + 1) - s_{K+1}$
- favored $\mathcal{L}^V(\Delta) = \sum_k \frac{1}{|C_k|} \sum_{i \in C_k} ||x_i - \mu_k||^2 \approx$ sum of cluster variances

Stability methods for choosing $K$

- like bootstrap, or crossvalidation
- Idea (implemented by [Ben-Hur et al., 2002]) for each $K$
  - perturb data $D \rightarrow D'$
  - cluster $D' \rightarrow \Delta'_k$
  - compare $\Delta_K, \Delta'_K$. Are they similar?
    - If yes, we say $\Delta_K$ is stable to perturbations

Fundamental assumption If $\Delta_K$ is stable to perturbations then $K$ is the correct number of clusters

- these methods are supported by experiments (not extensive)
- not YET supported by theory ... see [von Luxburg, 2009] for a summary of the area

A stability based method for model-based clustering

- The algorithm of [Lange et al., 2004]
  - divide data into 2 halves $D_1, D_2$ at random
  - cluster (by EM) $D_1 \rightarrow \Delta_1, \theta_1$
  - cluster (by EM) $D_2 \rightarrow \Delta_2, \theta_2$
  - cluster $D_1$ using $\theta_2 \rightarrow \Delta'_1$
  - compare $\Delta_1, \Delta'_1$
  - repeat $B$ times and average the results
    - repeat for each $K$
  - select $K$ where $\Delta_1, \Delta'_1$ are closest on average (or most times)
Clustering with outliers

- What are outliers?
- let $p = \text{proportion of outliers (e.g. 5\%-10\%)}$

Remedies
- mixture model: introduce a $K + 1$-th cluster with large (fixed) $\Sigma_{K+1}$, bound $\Sigma_k$ away from 0
- K-means and EM
  - robust means and variances
    - e.g. eliminate smallest and largest $pn_k/2$ samples in mean computation (trimmed mean)
  - K-medians [Charikar and Guha, 1999]
    - replace Gaussian with a heavier-tailed distribution (e.g. Laplace)
- single-linkage: do not count clusters with $< r$ points

Is $K$ meaningful when outliers present?
- alternative: non-parametric clustering

Outline

1. Paradigms for clustering
2. Parametric clustering algorithms (K given)
   - Cost based / hard clustering
   - Model based / soft clustering
3. Issues in parametric clustering
   - Selecting $K$
   - Outliers
4. Non-parametric clustering (smoothness given)
   - Based on non-parametric density estimation
   - Dirichlet process mixture models
5. Similarity based / graph clustering
   - Spectral clustering
   - Affinity propagation
6. Cluster validation
7. Special topics

Methods based on non-parametric density estimation

Idea
- The clusters are the isolated peaks in the (empirical) data density
  - group points by the peak they are under
  - some outliers possible
  - $K = 1$ possible (no clusters)
  - shape and number of clusters $K$ determined by algorithm
  - structural parameters
    - smoothness of the density estimate
    - what is a peak

Algorithms
- peak finding algorithms Mean-shift algorithms
- level sets based algorithms
  - Nugent-Stuetzle, Support Vector clustering
- Information Bottleneck [Tishby and Slonim, 2000]

Kernel density estimation

Input
- data $\mathcal{D} \subseteq \mathbb{R}^d$
  - Kernel function $K(z)$
  - parameter kernel width $h$ (is a smoothness parameter)

Output
- $f(x)$ a probability density over $\mathbb{R}^d$

$$f(x) = \frac{1}{Nh^d} \sum_{i=1}^{n} K\left(\frac{x - x_i}{h}\right)$$

- $f$ is sum of Gaussians centered on each $x_i$
The kernel function

- Example $K(z) = \frac{1}{(2\pi)^{d/2}}e^{-||z||^2/2}$, $z \in \mathbb{R}^d$ is the Gaussian kernel
- In general
  - $K()$ should represent a density on $\mathbb{R}^d$, i.e. $K(z) \geq 0$ for all $z$ and $\int K(z)dz = 1$
  - $K()$ symmetric around 0, decreasing with $||z||$
- In our case, $K$ must be differentiable

Remarks

- mean shift iteration guaranteed to converge to a max of $f$
- computationally expensive
- a faster variant...

Algorithm Mean Shift (Comaniciu-Meer)

Input Data $\mathcal{D} = \{x_i\}_{i=1:n}$, kernel $K(z)$, $h$
- select $q$ points $\{x_j\}_{j=1:q} = \mathcal{D}_q \subseteq \mathcal{D}$ that cover the data well
- for $j \in \mathcal{D}_q$
  - $x \leftarrow x_j$
  - iterate $x \leftarrow m(x)$ until convergence to $m_j$
- group points in $\mathcal{D}_q$ with same $m_j$ in a cluster
- assign points in $\mathcal{D} \setminus \mathcal{D}_q$ to the clusters by the nearest-neighbor method
  $k(i) = k(\text{argmin}_{j \in \mathcal{D}_q} ||x_i - x_j||)$

Mean shift algorithms

Idea find points with $\nabla f(x) = 0$
Assume $K(z) = e^{-||z||^2/2\sqrt{2\pi}}$ Gaussian kernel

$$\nabla f(x) = -\frac{1}{Nh^d} \sum_{i=1}^n K\left(\frac{x-x_i}{h}\right)(x-x_i)/h$$

Local max of $f$ is solution of implicit equation

$$x = \frac{\sum_{i=1}^n x_iK\left(\frac{x-x_i}{h}\right)}{\sum_{i=1}^n K\left(\frac{x-x_i}{h}\right)}$$

Algo Simple Mean Shift

Input Data $\mathcal{D} = \{x_i\}_{i=1:n}$, kernel $K(z)$, $h$
- for $i = 1:n$
  - $x \leftarrow x_i$
  - iterate $x \leftarrow m(x)$ until convergence to $m_i$
- group points with same $m_i$ in a cluster

Remarks

- all $x_i$ converge to a single point
  ⇒ need to stop before convergence

Gaussian blurring mean shift

Idea

- like Simple Mean Shift but points are shifted to new locations
- the density estimate $f$ changes
- becomes concentrated around peaks very fast

Algorithm Gaussian Blurring Mean Shift (GBMS)

Input Data $\mathcal{D} = \{x_i\}_{i=1:n}$, Gaussian kernel $K(z)$, $h$
- Iterate until STOP
  - for $i = 1:n$ compute $m(x_i)$
  - for $i = 1:n$, $x_i \leftarrow m(x_i)$
Empirical stopping criterion [Carreira-Perpinan, 2007]
- define $e_t^i = ||x_t^i - x_{t-1}^i||$ the change in $x_i$ at $t$
- define $H(e^t)$ the entropy of the histogram of $\{e_t^i\}$
- STOP when $\sum_{i=1}^n e_t^i/n < \text{tol}$ OR $|H(e^t) - H(e^{t-1})| < \text{tol}'$

**Convergence rate** If true $f$ Gaussian, convergence is **cubic**

$$||x^i_t - x^*|| \leq C||x^i_{t-1} - x^*||^3$$

very fast!!

**Algorithm Nugent-Stuetzle**

**Input** Data $D = \{x_i\}_{i=1:n}$, kernel $K(z)$
- Compute KDE $f(x)$ for chosen $h$
- for levels $0 < l_1 < l_2 < \ldots < l_r < \ldots < l_R \geq \sup_x f(x)$
  - find level set $L_r = \{x \mid f(x) \geq l_r\}$ of $f$
  - if $L_r$ disconnected then each connected component is a cluster
    $\rightarrow (C_{r,1}, C_{r,2}, \ldots C_{r,K_r})$
**Output** clusters $\{(C_{r,1}, C_{r,2}, \ldots C_{r,K_r})\}_{r=1:R}$

**Remarks**

- every cluster $C_{r,k} \subseteq$ some cluster $C_{r-1,k'}$
- therefore output is hierarchical clustering
- some levels can be pruned (i.e. $K_r = K_{r-1}$)
- algorithm can be made recursive, i.e. efficient
- finding level sets of $f$ tractable only for $d = 1, 2$
- for larger $d$, $L_r = \{x_i \in D \mid f(x_i) \geq l_i\}$
- to find connected components
  - for $i \neq j \in L_r$
    - if $f(tx_i + (1-t)x_j) \geq l_r$ for $t \in [0,1]$ then $k(i) = k(j)$
- confidence intervals possible by resampling

Cluster tree with 13 leaves (8 clusters, 5 artifacts)

(from [Nugent and Meila, 2010])
Support Vector (SV) clustering

Idea same as for Nugent-Stuetzle, but use kernelized density estimator instead of KDE

Algorithm SV

Input data \( D \), parameters \( q \) kernel width, \( p \in (0,1) \) proportion of outliers

1. construct a 1-class SVM with parameters \( q \), \( C = 1/np \)
2. this is equivalent to enclosing the data in a sphere in feature space
   for any \( x \) its distance from center of sphere is
   \[
   R^2(x) = K(x, x) - 2 \sum_j \alpha_j K(x, x_j) + \sum_{i,j} K(x_i, x_j)
   \]
   for \( x_i \) support vector, \( R(x_i) = R \) (same for all)
3. for all pairs \( i, j = 1 : n \)
   - \( i, j \) in same cluster if segment \([i, j]\) is within sphere with radius \( R \) in feature space
   - practically, test if \( R(tx_i + (1 - t)x_j) < R \) for \( t \) on a grid over \([0,1]\)

Remark

- the kernel used by SV is \( K(x, x') = e^{-q||x - x'||^2} \)
- \( q \) controls boundary smoothness
- SV’s lie on cluster boundaries, “margin error” points lie outside clusters (are outliers)
- SV theory \( \frac{\text{margin errors}}{n} \to \frac{1}{\pi C} = p \) for large \( n \)
- hence \( p \) controls the proportion of outliers
- \( p, q \) together control \( K \)
  - \( p \) larger, \( q \) smaller \( \Rightarrow K \) smaller

The Dirichlet distribution

- \( Z \in \{1 : r\} \) a discrete random variable, let \( \theta_j = P_z(j), j = 1, \ldots r \).
- Multinomial distribution Probability of i.i.d. sample of size \( N \) from \( P_z \)
  \[
  P(z^{1:N}) = \prod_{j=1}^{r} \theta_j^{N_j}
  \]
  where \( N_j = \# \text{the value } j \text{ is observed}, j = 1, \ldots r \)
- \( N_1, r \) are the sufficient statistics of the data.
- The Dirichlet distribution is defined over domain of \( \theta_1, \ldots, r \), with real parameters \( N_1', \ldots, r > 0 \) by
  \[
  D(\theta_1, \ldots, r; N_1', \ldots, r) = \frac{\Gamma(\sum_{j} N_j')}{\prod_j \Gamma(N'_j)} \prod_j \theta_j^{N_j'-1}
  \]
  where \( \Gamma(p) = \int_0^\infty t^{p-1}e^{-t}dt \).

Dirichlet process mixtures

- Model-based
- generalization of mixture models to
  - infinite \( K \)
- Bayesian framework
- denote \( \theta_k = \text{parameters for component } f_k \)
- assume \( f_k(x) \equiv f(x, \theta_k) \in \{f(x, \theta)\} \)
- assume prior distributions for parameters \( g_0(\theta) \)
- prior with hyperparameter \( \alpha > 0 \) on the number of clusters
- very flexible model
A sampling model for the data

- **Example:** Gaussian mixtures, \( d = 1 \), \( \sigma_k = \sigma \) fixed
- \( \theta = \mu \)
- Prior for \( \mu \) is \( \text{Normal}_{0,\sigma^2_0} \)
- **Sampling process**
  - for \( i = 1 : n \) sample \( x_i, k(i) \) as follows
    - denote \( \{1 : K\} \) the clusters after step \( i - 1 \)
    - define \( n_k \) the size of cluster \( k \) after step \( i - 1 \)
    - \( k(i) = \begin{cases} 
      k & \text{w.p. } \frac{n_k}{n-1+\alpha}, \ k = 1 : K \\
      K + 1 & \text{w.p. } \frac{\alpha}{n-1+\alpha} \end{cases} \) \( (13) \)
  - if \( k(i) = K + 1 \) sample \( \mu_i \equiv \mu_{K+1} \) from \( \text{Normal}(0,\sigma^2_0) \)
  - sample \( x_i \) from \( \text{Normal}(\mu_k(i), \sigma^2) \)
- can be shown that the distribution of \( x_{1:n} \) is interchangeable (does not depend on data permutation)

The hyperparameters

- \( \sigma_0 \) controls spread of centers
  - should be large
- \( \alpha \) controls number of cluster centers
  - \( \alpha \) large \( \Rightarrow \) many clusters
- cluster sizes non-uniform (larger clusters attract more new points)
- many single point clusters possible

**General Dirichlet mixture model**

- cluster densities \( \{f(x, \theta)\} \)
- parameters \( \theta \) sampled from prior \( g_0(\theta, \beta) \)
- cluster membership \( k(i) \) sampled as in \( (13) \)
- \( x_i \) sampled from \( f(x, \theta_{k(i)}) \)
- **Model Hyperparameters** \( \alpha, \beta \)

Clustering with Dirichlet mixtures via MCMC

**MCMC estimation for Dirichlet mixture**

**Input** \( \alpha, g_0, \beta, \{f\}, \mathcal{D} \)

**State**

- cluster assignments \( k(i), i = 1 : n \), parameters \( \theta_k \) for all distinct \( k \)

**Iterate**

- for \( i = 1 : n \) (reassign data to clusters)
  - resample \( k(i) \) by
    \[
    k(i) = \begin{cases} 
      \text{existing } k & \text{w.p. } \frac{n_k-1}{n-1+\alpha} f(x_i, \theta_k) \\
      \text{new cluster} & \text{w.p. } \frac{\alpha}{n-1+\alpha} \int f(x, \theta) g_0(\theta) d\theta \end{cases} \] \( (14) \)
  - if \( k(i) \) is new label, sample a new \( \theta_{k(i)} \propto g_0 f(x_i, \theta) \)
  - for \( k \in \{k(1 : n)\} \) (resample cluster parameters)
    - sample \( \theta_k \) from posterior \( g_k(\theta) \propto g_0(\theta, \beta) \Pi_{i \in C_k} f(x_i, \theta) \)
    - \( g_k \) can be computed in closed form if \( g_0 \) is conjugate prior

**Output** a state with high posterior
Summary: Parametric vs. non-parametric

**Parametric clustering**
- Optimizes a cost $\mathcal{L}$
- Most costs are NP-hard to optimize
- Assumes more detailed knowledge of cluster shapes
- Assumes $K$ known (But there are wrapper methods to select $K$)
- Gets harder with larger $K$
- Older, more used and studied

**Non-parametric clustering**
- density-based methods have no cost function
  - Dirichlet clustering samplers posterior of $k\{1:n\}, \{\theta_k\}$ given $\mathcal{D}$
  - do not depend critically on initialization
  - $K$ and outliers selected automatically, naturally
  - require hyperparameters (= smoothness parameters)

Note that Dirichlet mixture is inbetween parametric and non-parametric

Outline

1. Paradigms for clustering
2. Parametric clustering algorithms (K given)
   - Cost based / hard clustering
   - Model based / soft clustering
3. Issues in parametric clustering
   - Selecting $K$
   - Outliers
4. Non-parametric clustering (smoothness given)
   - Based on non-parametric density estimation
   - Dirichlet process mixture models
5. Similarity based / graph clustering
   - Spectral clustering
   - Affinity propagation
6. Cluster validation
7. Special topics

When to use

- Parametric
  - shape of clusters known
  - $K$ not too large or known
  - clusters of comparable sizes
- Non-parametric (density based)
  - shape of clusters arbitrary
  - $K$ large or many outliers
  - clusters sizes in large range (a few large clusters and many small ones)
  - dimension $d$ small (except for SV)
  - lots of data
- Dirichlet mixtures
  - shape of clusters known
  - clusters sizes in large range

Similarity based clustering

- **Paradigm:** the features we observe are measures of similarity/dissimilarity between pairs of data points, e.g.

<table>
<thead>
<tr>
<th>feature</th>
<th>points</th>
<th>features</th>
</tr>
</thead>
<tbody>
<tr>
<td>Image segmentation</td>
<td>pixels</td>
<td>distance in color space or location, separated by a contour, belong to same texture</td>
</tr>
<tr>
<td>Social network</td>
<td>people</td>
<td>friends, coworkers, phone calls, emails</td>
</tr>
<tr>
<td>Text analysis</td>
<td>words</td>
<td>appear in same context</td>
</tr>
</tbody>
</table>

The features are summarized by a single **similarity measure** $S_{ij}$
- e.g $S_{ij} = \exp\left(\sum_k \alpha_k \text{feature}_k(i,j)\right)$ for all points $i,j$
- symmetric $S_{ij} = S_{ji}$
- non-negative $S_{ij} \geq 0$

We want to put points that are similar to each other in the same cluster, dissimilar points in different clusters

- Problem is often cast as a **graph cut** problem
  - points = graph nodes, similarity $S_{ij} = \text{weight of edge } ij$
Paradigms for grouping

- **Graph cuts**
  - remove some edges $\Rightarrow$ disconnected graph
  - the groups are the connected components
- **By similar behavior**
  - nodes $i, j$ in the same group iff $i, j$ have the same pattern of connections w.r.t other nodes
- **By Embedding**
  - map nodes $V = \{1, 2, \ldots, n\} \longrightarrow \{x_1, x_2, \ldots, x_n\} \in \mathbb{R}^d$ then use standard classification and clustering methods

Definitions

- $V = \{1, 2, \ldots, n\}$
- node **degree** or **volume**
  $$D_i = \sum_{j \in V} S_{ij}$$
- **volume** of cluster $C \subseteq V$
  $$D_C = \sum_{i \in C} D_i$$
- **cut** between subsets $C, C' \subseteq V$
  $$\sum_{i \in C} \sum_{j \in C'} S_{ij}$$
- **Multiway Normalized Cut** of a partition $\Delta = \{C_1, \ldots, C_K\}$ of $V$
  $$\text{MNCut}(\Delta) = \sum_{k=1}^{K} \sum_{k' \neq k} \frac{\text{Cut}(C_k, C_{k'})}{D_{C_k}}$$
  in particular, for $K = 2$,
  $$\text{MNCut}(C, C') = \text{Cut}(C, C') \left( \frac{1}{D_C} + \frac{1}{D_{C'}} \right)$$

Motivation for MNCut

- **A random walks view**

  - Define
    $$P_{ij} = \frac{S_{ij}}{D_j} \text{ for all } i, j \in V$$
  - in matrix notation $P = D^{-1}S$ where $P = [P_{ij}], D = \text{diag}(D_1, \ldots, D_n)$
  - $P$ defines a **random walk** over the graph nodes $V$
Grouping from the random walks point of view

- **Idea**: group nodes together if they transition in the same way to other clusters

\[ P_{i,red} = \Pr[i \rightarrow red | i] = \sum_{j \in \text{red}} P_{ij} \]

... is the same as grouping by embedding

- **embedding** of \( V \) = mapping from \( V \) into \( \mathbb{R}^d \)
- **Wanted**: similar points embedded near each other
  - ideally, points in the same cluster mapped to the same point in \( \mathbb{R}^d \)

Lumpability

- A vector \( v \) is piecewise constant w.r.t a clustering \( \Delta \) iff \( v_i = v_j \) whenever \( i, j \) in same \( C \in \Delta \)

\[ P_{i,red} = f_{red} \]
\[ P_{i,yellow} = f_{yellow} \]

\[ \sum_{j \in C'} P_{ij} = R_{CC'} \] whenever \( i \in C \), for all \( C, C' \in \Delta \)

Some questions

- Not all graphs embed perfectly
- How many dimensions do we need?
- Nice, but we need to know the clusters in advance...

Lumpability [Lumpability][Meila&Shi 2001] Let \( S \) be a similarity matrix and \( \Delta \) a clustering with \( K \) clusters. Then \( P \) has \( K \) piecewise constant eigenvectors w.r.t \( \Delta \) iff

\[ \sum_{j \in C'} P_{ij} = R_{CC'} \] whenever \( i \in C \), for all \( C, C' \in \Delta \)
The spectral clustering

Spectral clustering in a nutshell

Spectral Clustering Algorithm

Properties of spectral clustering

The spectral mapping

Spectral clustering

An algorithm based on [Meilă and Shi, 2001b] and [Ng et al., 2002].

Spectral Clustering Algorithm

Input

- Similarity matrix $S$, number of clusters $K$

1. **Transform** $S$: Set $D_i = \sum_{j=1}^{n} S_{ij}$, $j = 1: n$ the node degrees. Form the transition matrix $P = [P_{ij}]_{ij=1}^n$ with $P_{ij} \leftarrow S_{ij}/D_i$, for $i,j = 1: n$

2. Compute the largest $K$ eigenvalues $\lambda_1 \geq \lambda_2 \geq \ldots \geq \lambda_K$ and eigenvectors $v_1, \ldots, v_K$ of $P$.

3. **Embed the data in principal subspace** Let $V = [v_2 \ v_3 \ \ldots \ v_K] \in \mathbb{R}^{n \times K}$, $x_i \leftarrow i$-th row of $V$.

4. **(orthogonal initialization)** Find $K$ initial centers by
   - Take $\mu_1$ randomly from $x_1, \ldots, x_n$
   - For $k = 2, \ldots, K$ set $\mu_k = \arg\min_{x_i} \max_{k' < k} \mu_{k'}^T x_i$.

5. Run the $K$-means algorithm on the “data” $x_1:n$ starting from the centers $\mu_1:K$.

Properties of spectral clustering

- Arbitrary cluster shapes (main advantage)
- Elegant mathematically
- Practical up to medium sized problems
- Running time (by Lanczos algorithm) $O(nk)/$iteration.
- Works well when $K$ known, not too large estimating $K$ [Azran and Ghahramani, 2006]
- Depend heavily on the similarity function (main problem)
- Learning the similarities
  - [Meiñă and Shi, 2001a], [Bach and Jordan, 2006], [Meiñă et al., 2005], [Shortreed
  - Outliers become separate clusters (user must adjust $K$ accordingly!)
- Very popular, many variants which aim to improve on the above
  - Diffusion maps [Nadler et al., 2006]: normalize the eigenvectors $\lambda_k^Tv_k$
- Practical fix, when $K$ large: only compute a fixed number of eigenvectors $d < K$. This avoids the effects of noise in lower ranked eigenvectors.
Affinity propagation

- **Idea** Each item \(i \in D\) finds an **exemplar** item \(k \in D\) to "represent" it
- Affinity Propagation is to spectral clustering what Mean Shift is to K-means
- number of exemplars not fixed in advance
- quantities of interest
  - similarities \(s_{ij}, i \neq j\) (given)
  - **availability** \(a_{ik}\) of \(k\) for \(i = \) how much support there is from other items for \(k\) to be an exemplar
  - **responsibility** \(r_{ik}\) that measures how fit is \(k\) to represent \(i\), as compared to other possible candidates \(k'\).
- diagonal elements \(s_{ii}\) represent **self-similarities**
  - larger \(s_{ii}\) \(\Rightarrow\) more likely \(i\) will become an exemplar \(\Rightarrow\) more clusters

### Affinity Propagation

**Affinity Propagation Algorithm** [Frey and Dueck, 2007]

**Input** Similarity matrix \(S = [s_{ik}]_{i=1}^n\), parameter \(\lambda = 0.5\)

Iterate the following steps until convergence

1. set \(a_{ik} \leftarrow 0\) for \(i, k = 1 : n\)
2. for all \(k\)
   - Find the best exemplar for \(i\):
     \(s^* \leftarrow \max_k (s_{ik} + a_{ik}),\)
     \(A_i^* \leftarrow \arg \max_k (s_{ik} + a_{ik})\) (can be a set of items)
   - **for all** \(k\) update responsibilities
     \(r_{ik} \leftarrow \begin{cases} s_{ik} - s^* , & \text{if } k \notin A_i^* \\ s_{ik} - \max_{k' \notin A_i^*} (s_{ik} + a_{ik}) & \text{otherwise} \end{cases}\)
3. for all \(k\) update availabilities
   - \(a_{kk} \leftarrow \sum_{i \neq k} [r_{ik}]_+\) where \([r_{ik}]_+ = r_{ik}\) if \(r_{ik} > 0\) and 0 otherwise.
   - for all \(i, a_{ik} \leftarrow \min \{0, r_{kk} + \sum_{i' \neq i, k} [r_{i'k}]_+\}\)
4. Assign an exemplar to \(i\) by \(k(i) \leftarrow \arg \max_{k'} (r_{ik'} + a_{ik'})\)

Outline

1. Paradigms for clustering
2. Parametric clustering algorithms (K given)
   - Cost based / hard clustering
   - Model based / soft clustering
3. Issues in parametric clustering
   - Selecting \(K\)
   - Outliers
4. Non-parametric clustering (smoothness given)
   - Based on non-parametric density estimation
   - Dirichlet process mixture models
5. Similarity based / graph clustering
   - Spectral clustering
   - Affinity propagation
6. Cluster validation
7. Special topics

Cluster validation

- **External**
  - when the true clustering \(\Delta^*\) is known
  - compares result(s) \(\Delta\) obtained by algorithm \(A\) with \(\Delta^*\)
  - validates algorithms/methods
- **Internal** - no external reference
External cluster validation

Scenarios
- given data $D$, truth $\Delta^*$; algorithm $A$ produces $\Delta$ is $\Delta$ close to $\Delta^*$?
- given data $D$, truth $\Delta^*$; algorithm $A$ produces $\Delta$, algorithm $A'$ produces $\Delta'$ which of $\Delta$, $\Delta'$ is closer to $\Delta^*$?
- multiple datasets, multiple algorithms which algorithm is better?

A distance between clusterings $d(\Delta, \Delta')$ needed

Requirements for a distance
- Depend on the application
  - Applies to any two partitions of the same data set
  - Makes no assumptions about how the clusterings are obtained
  - Values of the distance between two pairs of clusterings comparable under the weakest possible assumptions
  - Metric (triangle inequality) desirable
  - understandable, interpretable

The confusion matrix
- Let $\Delta = \{C_{1:K}\}$, $\Delta' = \{C'_{1:K'}\}$
- Define $n_k = |C_k|$, $n'_k = |C'_{k'}|$ $m_{kk'} = |C_k \cap C'_{k'}|$, $k = 1:K$, $k' = 1:K'$
- note: $\sum_k m_{kk'} = n'_k$, $\sum_k' m_{kk'} = n_k$, $\sum_{k,k'} m_{kk'} = n$
- The confusion matrix $M \in \mathbb{R}^{K \times K'}$ is
  $$M = [m_{kk'}]_{k'=1:K'}^{k=1:K}$$
- all distances and comparison criteria are based on $M$
- the normalized confusion matrix $P = M/n$
  $$p_{kk'} = \frac{m_{kk'}}{n}$$
- The normalized cluster sizes $p_k = n_k/n$, $p_{k'} = n'_k/n$ are the marginals of $P$
  $$p_k = \sum_{k'} p_{kk'}$$

The Misclassification Error (ME) distance
- Define the Misclassification Error (ME) distance $d_{\text{ME}}$
  $$d_{\text{ME}} = 1 - \max_\pi \sum_{k=1}^K p_k.\pi(k) \quad \pi \in \{\text{all } K\text{-permutations}, K \leq K'\text{.log}\}$$
- Interpretation: treat the clusterings as classifications, then minimize the classification error over all possible label matchings
- Or: $nd_{\text{ME}}$ is the Hamming distance between the vectors of labels, minimized over all possible label matchings
- can be computed in polynomial time by Max bipartite matching algorithm (also known as Hungarian algorithm)
- Is a metric: symmetric, $\geq 0$, triangle inequality
  $$d_{\text{ME}}(\Delta_1, \Delta_2) + d_{\text{ME}}(\Delta_1, \Delta_3) \geq d_{\text{ME}}(\Delta_2, \Delta_3)$$
- easy to understand (very popular in computer science)
- $d_{\text{ME}} \leq 1 - 1/K$
- bad: if clusterings not similar, or $K$ large, $d_{\text{ME}}$ is coarse/indiscriminative
- recommended: for small $K$
The Variation of Information (VI) distance
Clusterings as random variables

- Imagine points in $D$ are picked randomly, with equal probabilities
- Then $k(i), k'(j)$ are random variables
  with $Pr[k] = p_k$, $Pr[k, k'] = p_{kk'}$

Incursion in information theory I
Entropy of a random variable/clustering $H_\Delta = -\sum_k p_k \ln p_k$

- $0 \leq H_\Delta \leq \ln K$
- Measures uncertainty in a distribution (amount of randomness)
- Joint entropy of two clusterings
  $H_{\Delta, \Delta'} = -\sum_{k,k'} p_{kk'} \ln p_{kk'}$
- $H_{\Delta', \Delta} \leq H_\Delta + H_{\Delta'}$ with equality when the two random variables are independent
- Conditional entropy of $\Delta'$ given $\Delta$
  $H_{\Delta' | \Delta} = -\sum_k p_k \sum_{k'} \frac{p_{kk'}}{p_k} \ln \frac{p_{kk'}}{p_k}$
- Measures the expected uncertainty about $k'$ when $k$ is known
- $H_{\Delta' | \Delta} \leq H_{\Delta'}$ with equality when the two random variables are independent

Incursion in information theory II

- Mutual information between two clusterings (or random variables)
  $I_{\Delta, \Delta'} = H_{\Delta} + H_{\Delta'} - H_{\Delta, \Delta'}$
  $= H_{\Delta'} - H_{\Delta' | \Delta}$
- Measures the amount of information of one r.v. about the other
- $I_{\Delta, \Delta'} \geq 0$, symmetric. Equality iff r.v.’s independent

The VI distance

- Define the Variation of Information (VI) distance
  $d_{VI}(\Delta, \Delta') = H_\Delta + H_{\Delta'} - 2I_{\Delta, \Delta'}$
  $= H_{\Delta | \Delta'} + H_{\Delta' | \Delta}$
- Interpretation: $d_{VI}$ is the sum of information gained and information lost when labels are switched from $k()$ to $k'(())$
- $d_{VI}$ symmetric, $\geq 0$
- $d_{VI}$ obeys triangle inequality (is a metric)

Other properties

- Upper bound $d_{VI} \leq 2 \ln K_{\max}$ if $K, K' \leq K_{\max} \leq \sqrt{n}$
  (asymptotically attained)
- $d_{VI} \leq \ln n$ over all partitions (attained)
- Unbounded! and grows fast for small $K$
Other criteria and desirable properties

- Comparing clustering by indices of similarity $i(\Delta, \Delta')$
  - from statistics (Rand, adjusted Rand, Jaccard, Fowlkes-Mallows ...)
  - range $=[0,1]$, with $i(\Delta, \Delta') = 1$ for $\Delta = \Delta'$
  - the properties of these indices not so good
  - any index can be transformed into a "distance" by $d(\Delta, \Delta') = 1 - i(\Delta, \Delta')$
- Other desirable properties of indices and distances between clusterings
  - $n$-invariance
  - locality
  - convex additivity

Other desirable properties of indices and distances between clusterings

- $n$-invariance
- locality
- convex additivity

Define $N_{11} = \#$ pairs which are together in both clusterings, $N_{12} = \#$ pairs together in $\Delta$, separated in $\Delta'$, $N_{21}$ (conversely), $N_{22} = \#$ number pairs separated in both clusterings

- Rand index $= \frac{N_{11} + N_{22}}{\# \text{pairs}}$
- Jaccard index $= \frac{N_{11}}{\# \text{pairs}}$
- Fowlkes-Mallows $= \text{Precision} \times \text{Recall}$

- all vary strongly with $K$. Therefore, Adjusted indices used mostly
  
  $$adj(i) = \frac{i - \bar{i}}{\max(i) - \bar{i}}$$

Internal cluster(ing) validation

Why?

- Most algorithms output a clustering even if no clusters in data (parametric algorithms)
  - How to decide whether to accept it or not?
  - related to selection of $K$
  - Some algorithms are run multiple times (e.g EM)
    - How to select the clustering(s) to keep?
  - Validate by the cost $L$
    - $\Delta$ is valid if $L(\Delta)$ is "small"
    - but how small is "small"?
    - Note: rescaling data may change $L(\Delta)$

Heuristics

- Gap heuristic
  - single linkage:
    - define $l_r$ length of $r$-th edge added to MST
    - $l_1 \leq l_2 \leq \cdots \leq l_{n-K} \leq l_{n-K+1} \leq \cdots$
    - intracluster deleted
    - $l_{n-K} / l_{n-K+1} \leq 1$ should be small
  - min diameter:
    - $\frac{L(\Delta)}{\max_{i,j \in D} ||x_i - x_j||}$
    - $\frac{L(\Delta)}{\min_{k,k'} \text{distance}(C_k, C_{k'})}$
    - etc
**Quadratic cost**

- \( \mathcal{L}(\Delta) = \text{const} - \text{trace} \ X^T(\Delta) A X(\Delta) \)
- with \( X = \) matrix representation for \( \Delta \)
- then, if cost value \( \mathcal{L}(\Delta) \) small, we can prove that clustering \( \Delta \) is almost optimal

This holds for K-means (weighted, kernelized) and several graph partitioning costs (normalized cut, average association, correlation clustering, etc)

**Matrix Representations**

- matrix representations for \( \Delta \)
  - unnormalized (redundant) representation
    \[ \tilde{X}_{ik} = \begin{cases} 1 & i \in C_k \\ 0 & i \notin C_k \end{cases} \quad \text{for } i = 1 : n, k = 1 : K \]
  - normalized (redundant) representation
    \[ X_{ik} = \begin{cases} 1/\sqrt{|C_k|} & i \in C_k \\ 0 & i \notin C_k \end{cases} \quad \text{for } i = 1 : n, k = 1 : K \]
  - therefore \( X_k^T X_{k'} = \delta(k, k') \), \( X \) orthogonal matrix
  - \( X_k = \) column \( k \) of \( X \)
- normalized non-redundant representation
  - \( X_K \) is determined by \( X_{1:K-1} \)
  - hence we can use \( Y \in \mathbb{R}^{n \times (K-1)} \) orthogonal representation
  - intuition: \( Y \) represents a subspace (is an orthogonal basis)
  - \( K \) centers in \( \mathbb{R}^d, d \geq K \) determine a \( K-1 \) dimensional subspace plus a translation

**A spectral lower bound**

- minimizing \( \mathcal{L}(\Delta) \) is equivalent to
  \[ \max Y^T A Y \]
  over all \( Y \in \mathbb{R}^{n \times (K-1)} \) that represent a clustering
- a relaxation
  \[ \max Y^T A Y \]
  over all \( Y \in \mathbb{R}^{n \times (K-1)} \) orthogonal
- solution to relaxed problem is
  \[ Y^* = \text{eigenvectors } 1:K-1 \text{ of } A \]
  \[ \mathcal{L}^* = \sum_{k=1}^{K-1} \lambda_k(A) \]
- \( \mathcal{L}^* = \text{constant} - \text{trace } A - \mathcal{L}^* \) is lower bound for \( \mathcal{L} \)
  \[ \mathcal{L}^* \leq \mathcal{L}(\Delta) \quad \text{for all } \Delta \]
A theorem (Meila, 2006)

Theorem

- define
  \[ \delta = Y^TAY - \sum_{k=1}^{K-1} \lambda_k \]
  \[ \varepsilon(\delta) = 2\delta[1 - \delta/(K - 1)] \]

- define \( p_{\text{min}}, p_{\text{max}} = \min_{\lambda} \max_n |C_k| \)

- then, whenever \( \varepsilon(\delta) \leq p_{\text{min}} \), we have that
  \[ d_{ME}(\Delta, \Delta^{opt}) \leq \varepsilon(\delta)p_{\text{max}} \]

where \( d_{ME} \) is misclassification error distance

Remarks

- it is a worst-case result
- makes no (implicit) distributional assumptions
- when theorem applies, bound is good \( d_{ME}(\Delta, \Delta^{opt}) \leq p_{\text{min}} \)
- applies only if a good clustering is found (not all data, clusterings)
- intuition: if data well clustered, \( K - 1 \) principal subspace is aligned with cluster centers

Outline

1. Paradigms for clustering
2. Parametric clustering algorithms (K given)
   - Cost based / hard clustering
   - Model based / soft clustering
3. Issues in parametric clustering
   - Selecting K
   - Outliers
4. Non-parametric clustering (smoothness given)
   - Based on non-parametric density estimation
   - Dirichlet process mixture models
5. Similarity based / graph clustering
   - Spectral clustering
   - Affinity propagation
6. Cluster validation
7. Special topics

What I didn’t talk about

- Hierarchical clustering
- Subspace clustering (or clustering on subsets of attributes)
- Bi-clustering (and multi-way-clustering)
- Partial clustering
- Ensembles of clusterings, consensus clustering, and clustering clusterings
Hierarchical clustering

- **Divisive** (top down)
  - starts with all data in one cluster, divides recursively into 2 (or more) clusters
  - Example: spectral clustering, min diameter
- **Agglomerative** (bottom up)
  - starts $n$ cluster containing 1 item, merges 2 clusters recursively
  - Example: Ward algorithm, single linkage
- **Hierarchical Dirichlet processes**

**Remarks**
- Any cost based clustering paradigm can produce a hierarchical clustering
- Any non-parametric level-sets paradigm can produce a hierarchical clustering
- Mixture models (finite or not) can also be defined hierarchically. Issues of identifiability appear

Subspace clustering

- Problem: each cluster is defined by a subset of relevant attributes (features)
  - Examples: user modeling (clusters of users vs clusters of products/services), gene expression data
- Known as Clustering on Subsets of Attributes (COSA) Biclustering (and Multiway Clustering). Subspace clustering
- Amounts to clustering both the data exemplars and the data features

**Approaches**
- **COSA** [Friedman and Meulman, 2004] cost based, + additional entropy term. Alternate minimization algorithm.
- **[Hoff, 2005]** Dirichlet process mixtures approach. Each $f(.; \theta_k)$ samples a set of relevant features. Estimated by MCMC
- **Multivariate Information Bottleneck** [Friedman et al., 2001] Information theory based. Estimation by alternate (KL-divergence) projections.
- many others...see IEEE TKDE

The Ward agglomerative algorithm [Ward, 1963]

- **Cost** = same as K-means
- **Algorithm idea:**
  - Start with $n$ single point clusters
  - Merge the two clusters that increase $L$ the least, until $K$ clusters left
- **Greedy**, recursive algorithm, $O(n^3)$ operations

Partial clustering

- **Problem**: Given a node, find its cluster
- **Premise**: the data set is extremely large, there are many small clusters, possibly $O(n)$
- **Nibble** algorithm of [Spielman and Teng, 2008]
  - Given: a graph, by its Markov transition matrix $P$
  - Start with node $i$, tolerance $\varepsilon$, number steps $t$
  - Initialize $p \in \mathbb{R}^n$ with $p_i = 1$, $p_j = 0$ for $j \neq i$
  - **Iterate for $t$ steps**
    - $p \leftarrow Pp$
    - for $j = 1 : n$, if $p_j < \varepsilon$ set $p_j = 0$
  - **Output** $C(i) = \{ j | p_j > 0 \}$
  - $C(i)$ is the set of items attainable from $i$ by a “likely” path
  - Original algorithm has sparsest cut guarantees
    - Used as subroutine by other algorithms.
On spectral learning of mixtures of distributions.

Learning mixtures of arbitrary gaussians.

Spectral methods for automatic multiscale data clustering.
In Computer Vision and Pattern Recognition, pages 190–197. IEEE Computer Society.

Learning spectral clustering with applications to speech separation.

Model-based gaussian and non-gaussian clustering.

A stability based method for discovering structure in clustered data.
In Pacific Symposium on Biocomputing, pages 6–17.

Clustering via concave minimization.
In Advances in Neural Information Processing systems (NIPS), Cambridge, MA. MIT Press.

How the initialization affects the stability of the k-means algorithm.

Gaussian mean shift is an EM algorithm.

Improved combinatorial algorithms for the facility location and k-median problems.

Experiments with random projection.

An elementary proof of a theorem of johnson and lindenstrauss.

A probabilistic analysis of em for mixtures of separated, spherical gaussians.

Clustering by passing messages between data points.

Clustering on subsets of attributes.

Multivariate information bottleneck.

Coresets for k-means and k-median clustering and their applications.

A best possible heuristic for the k-center problem.

Subset clustering of binary sequences, with an application to genomic abnormality data.

Stability-based validation of clustering solutions.

Least squares quantization in PCM.

The EM algorithm and extensions.
Wiley, New York, NY.

The uniqueness of a good optimum for K-means.

Learning segmentation by random walks.

A random walks view of spectral segmentation.
In Jaakkola, T. and Richardson, T., editors, Artificial Intelligence and Statistics AISTATS.

Regularized spectral learning.

Diffusion maps, spectral clustering and eigenfunctions of fokker-planck operators.

A view of the em algorithm that justifies incremental, sparse, and other variants.

On spectral clustering: Analysis and an algorithm.

Statistical Methods in Molecular Biology, chapter An Overview of Clustering Applied to Molecular Biology.
Humana Press, Springer.
X-means: Extending K-means with efficient estimation of the number of clusters.

Unsupervised spectral learning.

A local clustering algorithm for massive graphs and its application to nearly-linear time graph partitioning.

An investigation of computational and informational limits in gaussian mixture clustering.

Data clustering by markovian relaxation and the information bottleneck method.
In NIPS, pages 640–646.

A spectral algorithm for learning mixtures of distributions.

A comparison of spectral clustering algorithms.
TR 03-05-01, University of Washington.

Clustering stability: An overview.

Hierarchical grouping to optimize an objective function.