Classic and Modern Data Clustering

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

IMA, October 4, 2013

Introduction to Statistics and Probability for Topologists
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 $K$
   - $(\alpha, \epsilon)$ Clusterability
   - 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
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 $K$
   - $(\alpha, \epsilon)$ Clusterability
   - 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**
  \[
  D = \{ x_1, x_2, \ldots, x_n \} \text{ a data set}
  \]
  \[
  n = \text{number of data points}
  \]
  \[
  K = \text{number of clusters (} K << n \text{)}
  \]
  \[
  \Delta = \{ C_1, C_2, \ldots, C_K \} \text{ a partition of } D \text{ into disjoint subsets}
  \]
  \[
  k(i) = \text{the label of point } i
  \]
  \[
  \mathcal{L}(\Delta) = \text{cost (loss) of } \Delta \text{ (to be minimized)}
  \]

- **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 \( \gamma = \{ \gamma_{ki} \}_{k=1:K}^{i=1:n} \)
    \[
    \gamma_{ki} = \text{the degree of membership of point } i \text{ to cluster } k
    \]
    \[
    \sum_k \gamma_{ki} = 1 \text{ for all } i
    \]
    (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): Model based [soft]
  
  - **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 $[S_{ij}]_{i,j=1:n}$, $S_{ij} = S_{ji} \geq 0$
  - **Similarity based clustering**
    - Graph partitioning
    - spectral clustering [hard, $K$ fixed, cost based]
    - typical cuts [hard non-parametric, cost based]
    - Affinity propagation [hard/soft non-parametric]
## 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>$K$</td>
<td>Known</td>
<td>Unknown</td>
</tr>
<tr>
<td><strong>“Goal”</strong></td>
<td>Prediction</td>
<td>Exploration [Lots of data to explore!]</td>
</tr>
<tr>
<td><strong>Stage of field</strong></td>
<td>Mature</td>
<td>Still young</td>
</tr>
</tbody>
</table>
Distances between partitions

\( \mathcal{P}(\mathcal{D}) = \) all possible partitions of a finite set \( \mathcal{D} \) (the lattice of partitions)

What is a “good” distance of \( \mathcal{P} \)?

*Depends 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*
Distances between partitions

\[ \mathcal{P}(\mathcal{D}) = \text{all possible partitions of a finite set } \mathcal{D} \ (\text{the lattice of partitions}) \]

What is a “good” distance of \( \mathcal{P} \)?
Depends 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**
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 \( P \)

\[
p_k = \sum_{k'} p_{kk'} \quad p'_{k'} = \sum_k p_{kk'}
\]
The Misclassification Error (ME) distance

(=Max Bipartite Matching)

• Define the Misclassification Error (ME) distance $d_{ME}$

$$d_{ME} = 1 - \max_{\pi} \sum_{k=1}^{K} p_{k,\pi(k)} \quad \pi \in \{\text{all } K-\text{permutations}\}, \ K \leq K' \text{ w.l.o.g}$$

• Interpretation: treat the clusterings as classifications, then minimize the classification error over all possible label matchings

• Or: $nd_{ME}$ 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

$$d_{ME}(\Delta_1, \Delta_2) + d_{ME}(\Delta_1, \Delta_3) \geq d_{ME}(\Delta_2, \Delta_3)$$

• easy to understand (very popular in computer science)

• $d_{ME} \leq 1 - 1/K$

• bad: if clusterings not similar, or $K$ large, $d_{ME}$ is coarse/indiscriminative
The Variation of Information (VI) distance

a.k.a. clusterings as random variables

\[
d_{II}(\Delta, \Delta') = H_\Delta + H_{\Delta'} - 2I_{\Delta',\Delta} = H_{\Delta|\Delta'} + H_{\Delta'|\Delta}
\]

\(d_{II}\) is a metric
The Variation of Information (VI) distance

a.k.a. clusterings as random variables

\[ d_{VI}(\Delta, \Delta') = H_\Delta + H_{\Delta'} - 2I_{\Delta', \Delta} = H_{\Delta|\Delta'} + H_{\Delta'|\Delta} \]

\( d_{VI} \) is a metric

- Imagine points in \( \mathcal{D} \) 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
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
  \[ d_{VI} \leq 2 \ln K_{max} \text{ if } K, K' \leq K_{max} \leq \sqrt{n} \]
  (asymptotically attained)
- \( d_{VI} \leq \ln n \text{ over all partitions (attained) } \)
- 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 ...)
  - from machine learning **Normalized Mutual Information**
    \[ NMI = \frac{I_{\Delta',\Delta}}{H_{\Delta',\Delta}} \]
    - 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
Define $N_{11} = \#\text{ pairs which are together in both clusterings}$, $N_{12} = \#\text{ pairs together in } \Delta$, separated in $\Delta'$, $N_{21}$ (conversely), $N_{22} = \#\text{number pairs separated in both clusterings}$

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

all vary strongly with $K$. Therefore, people use mostly adjusted indices

$$adj(i) = \frac{i - \bar{i}}{\max(i) - \bar{i}}$$
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 $K$
   - $(\alpha, \epsilon)$ Clusterability
   - 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

  \emph{will return to these later if there is time}

- Single linkage $= \min$ Spanning Tree $\setminus K$ edges $= K$-th level in filtration

- Min diameter
  - Fastest first traversal (HS initialization)

- K-medians

- K-means

- Model based (cost is derived from likelihood)

  Model is mixture model (e.g. mixture of Normals)

- EM algorithm

- \textit{“Computer science”/”Probably correct” algorithms}
Single Linkage Clustering

**Algorithm Single-Linkage**

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

1. Construct the Minimum Spanning Tree (MST) of $\mathcal{D}$
2. Delete the largest $K - 1$ edges

- **Cost** $\mathcal{L}(\Delta) = - \min_{k,k'} \text{distance}(C_k, C_{k'})$
  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 $\mathcal{L}$ that can be optimized in polynomial time
- Sensitive to outliers!
Minimum diameter clustering

- **Cost** $\mathcal{L}(\Delta) = \max_k \max_{i,j \in C_k} ||x_i - x_j||$
  - 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
    $$\mathcal{L}^{opt} \leq \mathcal{L}(\Delta) \leq 2\mathcal{L}^{opt}$$
  - Rediscovered many times
**Algorithm Fastest First Traversal**

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

defines centers $\mu_1:K \in \mathcal{D}$

(many other clustering algorithms use centers)

1. pick $\mu_1$ at random from $\mathcal{D}$
2. for $k = 2 : K$
   
   $\mu_k \leftarrow \text{argmax} \underset{\mathcal{D}}{\text{distance}}(x_i, \{\mu_1:k-1\})$
3. 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** $\mathcal{L}(\Delta) = \sum_k \sum_{i \in C_k} \|x_i - \mu_k\|$ with $\mu_k \in \mathcal{D}$
  - (usually) assumes centers chosen from the data points (analogy to median)

**Ex:** Show that in 1D $\arg\min_{\mu} \sum_i |x_i - \mu|$ 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)

$$\min_{u,y} \sum_{ij} d_{ij} u_{ij}$$

s.t.
- $\sum_j u_{ij} = 1$ point $i$ is in exactly 1 cluster for all $i$
- $\sum_j y_j \leq k$ there are at most $k$ clusters
- $u_{ij} \leq y_j$ point $i$ can only belong to a center for all $i, j$

Linear Programming Relaxation of K-medians

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

\[(LP) \min_{u,y} \sum_{ij} d_{ij} u_{ij} \]

s.t.
- $\sum_j u_{ij} = 1$
- $\sum_j y_j \leq k$
- $u_{ij} \leq y_j$
**Algorithm K-Medians** (variant of [Bradley and Mangasarian, 2005])

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

1. Solve (LP)
   - obtain fractionary “centers” $y_{1:n}$ and “assignments” $u_{1:n, 1:n}$

2. Sample $K$ centers $\mu_1 \ldots \mu_K$ by
   - $P[\mu_k = \text{point}_{j}] \propto y_j$ (without replacement)

3. Assign points to centers (deterministically)
   
   $$k(i) = \arg\min_k ||x_i - \mu_k||$$

- **Guarantees (Agarwal)**
  - **Given** tolerance $\varepsilon$, confidence $\delta$, $K' = K(1 + \frac{1}{\varepsilon}) \ln \frac{n}{K}$, $\Delta_{K'}$ obtained by **K-medians** with $K'$ centers
  
  $$\mathcal{L}(\Delta_{K'}) \leq (1 + \varepsilon)\mathcal{L}_K^{opt}$$
K-means clustering

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

Algorithm K-Means

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

**Initialize** 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)

   \[ k(i) = \arg\min_k ||x_i - \mu_k|| \]

2. for $k = 1 : K$ (recalculate centers)

   \[ \mu_k = \frac{1}{|C_k|} \sum_{i \in C_k} x_i \] \hspace{1cm} (1)

**Convergence**

- if $\Delta$ doesn’t change at iteration $m$ it will never change after that
- Convergence is proven in finite number of steps
The K-means cost

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

- "least-squares" cost (also called distortion)
- **Proposition** The K-means algorithm decreases \( L(\Delta) \) at every step.

Sketch of proof
- step 1: reassigning the labels can only decrease \( L \)  
  **Ex:** show this
- step 2: reassigning the centers \( \mu_k \) can only decrease \( L \) 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

Therefore, **K-means** converges to a local minimum of the cost \( L \)

Initialization matters (see later)
Equivalent cost functions

- The distortion can also be expressed as
  - sum of (squared) intracluster distances
    \[
    \mathcal{L}(\Delta) = \frac{1}{2} \sum_{k=1}^{K} \sum_{i,j \in C_k} \|x_i - x_j\|^2 + \text{constant} \tag{3}
    \]
  - (negative) sum of (squared) intercluster distances
    \[
    \mathcal{L}(\Delta) = -\frac{1}{2} \sum_{k=1}^{K} \sum_{i \in C_k} \sum_{j \notin C_k} \|x_i - x_j\|^2 + \text{constant} \tag{4}
    \]

Proof of (3)
Replace \(\mu_k\) as expressed in (1) in the expression of \(\mathcal{L}\), then rearrange the terms

Proof of (4)
\[
\sum_{k} \sum_{i,j \in C_k} \|x_i - x_j\|^2 = \sum_{i=1}^{n} \sum_{j=1}^{n} \|x_i - x_j\|^2 - \sum_{k} \sum_{i \in C_k} \sum_{j \notin C_k} \|x_i - x_j\|^2
\]
\[
\text{independent of } \Delta
\]
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 $\mathcal{L}^2(\Delta) = \max_k \max_{i,j \in C_k} \|x_i - x_j\|^2$
Initialization of $\mu_{1:K}$.

**The Power Initialization** (see also [Bubeck et al., 2009])

1. 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)
2. run 1 step of K-means
3. remove all centers $\mu_k^0$ that have few points, e.g $|C_k| < \frac{n}{eK'}$
4. from the remaining centers select $K$ centers by **Fastest First Traversal**
   1. pick $\mu_1$ at random from the remaining $\{\mu_{1:K'}^0\}$
   2. for $k = 2 : K$, $\mu_k \leftarrow \arg \max_{\mu_{k'}^0} \min_{j = 1:k-1} ||\mu_{k'}^0 - \mu_j||$, i.e next $\mu_k$ is furthest away from the already chosen centers
5. continue with the standard **K-means** algorithm
This initialization has been shown experimentally and theoretically to work well.

More precisely $K' = \tilde{K}(\ln \tilde{K} + \ln \frac{1}{\delta})$ where $\tilde{K} = n/$(size of smallest cluster) and e.g. $\delta = 0.05$, $1 - \delta =$desired level of confidence

Ex: Find an (approximate) formula for $P[\text{sample all } C_k \mid K']$ 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!
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 $O(n + K^5 \log^9 n + gK^2 \log^5 n)$ where $g = e^{[C/\varepsilon \log(1+1/\varepsilon)]^{d-1}}$ (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 $O(n + K^5 \log^9 n + K^{K+2} \varepsilon^{-(2d+1)} \log^{K+1} n \log^K (\frac{1}{\varepsilon}))$. 
Model based clustering: Mixture models

- The mixture density

\[ f(x) = \sum_{k=1}^{K} \pi_k f_k(x) \quad \text{with} \quad \pi_k \geq 0, \quad \sum_{k=1}^{K} \pi_k = 1 \quad (5) \]

- \( f_k(x) \) = 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

\[ \gamma_{ki} \overset{\text{def}}{=} P[x_i \in C_k] = \frac{\pi_k f_k(x)}{f(x)} \quad \text{for} \quad i = 1 : n, \quad k = 1 : K \quad (6) \]
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)
Criterion for clustering: Max likelihood

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

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

\( \theta^{ML} = \arg\max_{\theta} L(\theta) \)

\( \theta^{ML} \) determines a soft clustering \( \gamma \)

- a soft clustering \( \gamma \) determines a \( \theta \) (see later)
- Therefore we can write

\[
\mathcal{L}(\gamma) = -L(\theta(\gamma))
\]
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 $\mathcal{D} = \{x_i\}_{i=1:n}$, 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 $\Gamma_k = \sum_{i=1}^{m} \gamma_{ki}$, $k = 1 : K$ (note:

$\sum_k \Gamma_k = n$

$$\pi_k = \frac{\Gamma_k}{n}, \; k = 1 : K$$

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

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

1. $\Sigma$ need to be symmetric, positive definite matrices
Define the indicator variables

\[ z_{ik} = \begin{cases} 
1 & \text{if } i \in C_k \\
0 & \text{if } i \not\in C_k 
\end{cases} \]  \hspace{1cm} (8)

denote \( \bar{z} = \{z_{ki}\}_{i=1:n, k=1:K} \)

Define the complete log-likelihood

\[ l_c(\theta, \bar{z}) = \sum_{i=1}^{n} \sum_{k=1}^{K} z_{ki} \ln \pi_k f_k(x_i) \]  \hspace{1cm} (9)

\( E[z_{ki}] = \gamma_{ki} \)

Then

\[ E[l_c(\theta, \bar{z})] = \sum_{i=1}^{n} \sum_{k=1}^{K} E[z_{ki}][\ln \pi_k + \ln f_k(x_i)] \]  \hspace{1cm} (10)

\[ = \sum_{i=1}^{n} \sum_{k=1}^{K} \gamma_{ki} \ln \pi_k + \sum_{i=1}^{n} \sum_{k=1}^{K} \gamma_{ki} \ln f_k(x_i) \]  \hspace{1cm} (11)
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)
Brief analysis of EM

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

- 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 \( Q \) \( \Rightarrow \) \( I(\theta^*) \) local max for \( I(\theta) \)
- 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 \( Q \)

\[
Q(\tilde{P}, \theta) = \mathcal{L}(\theta) - KL(\tilde{P}||P(Z|X, \theta))
\]

where \( P(X, Z|\theta) = \prod_i \prod_k P[z_i = k]P[x_i|\theta_k] \)
\( \tilde{P}(Z) \) 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** \( \max_{\tilde{P}} Q \Leftrightarrow KL(\tilde{P}||P(Z|X, \theta)) \)
- **M step** \( \max_{\theta} Q \Leftrightarrow KL(P(X|Z, \theta^{old})||P(X|\theta)) \)

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 $\frac{\gamma_{ki}}{\Gamma_k}$

**M step**

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

**Ex:** Prove the formulas above

- Note also that **K-means** is **EM** with $\Sigma_k = \sigma^2 I_d$, $\sigma^2 \rightarrow 0$  **Ex:** Prove it
More special cases [Banfield and Raftery, 1993] introduce the following description for a covariance matrix in terms of *volume, shape, alignment with axes* (=determinant, trace, e-vectors). The letters below mean: I=unitary (shape, axes), E=equal (for all $k$), V=unequal

- **EII**: equal volume, round shape (spherical covariance)
- **VII**: varying volume, round shape (spherical covariance)
- **EEI**: equal volume, equal shape, axis parallel orientation (diagonal covariance)
- **VEI**: varying volume, equal shape, axis parallel orientation (diagonal covariance)
- **EVI**: equal volume, varying shape, axis parallel orientation (diagonal covariance)
- **VVI**: varying volume, varying shape, equal orientation (diagonal covariance)
- **EEE**: equal volume, equal shape, equal orientation (ellipsoidal covariance)
- **EEV**: equal volume, equal shape, varying orientation (ellipsoidal covariance)
- **VEV**: varying volume, equal shape, varying orientation (ellipsoidal covariance)
- **VVV**: varying volume, varying shape, varying orientation (ellipsoidal covariance)

(from [Nugent and Meila, 2010])
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 (S)
  - e.g. \( \|\mu_k - \mu_l\| \geq C \max(\sigma_k, \sigma_l) \)
  - with \( \sigma^2_k = \max \) 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 \( \mu_k \))
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 $d' \leq (\varepsilon^2/2 - \varepsilon^3/3) \ln n$. Then for any set $\mathcal{D}$ of $n$ points in $\mathbb{R}^d$, there is a map $f : \mathbb{R}^d \to \mathbb{R}^{d'}$ such that for all $u, v \in V$,

$$
(1 - \varepsilon)||u - v||^2 \leq ||f(u) - f(v)|| \leq (1 + \varepsilon)||u - v||^2
$$

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 $n, \varepsilon$
- [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 $v \sim$ i.i.d. and let $\tilde{v} =$ projection of $v$ on axes $1 : d'$. Then $E[||\tilde{v}||^2] = d'E[v_j^2] = \frac{d'}{d} E[||v||^2]$. The next step is to show that the variance of $||\tilde{v}||^2$ is very small when $d'$ is sufficiently large.
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)**

1. Project points $\{x_i\} \in \mathbb{R}^d$ on $K - 1$-th principal subspace $\Rightarrow \{y_i\} \in \mathbb{R}^K$
2. do distance-based ”harvesting” of clusters in $\{y_i\}$
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, $n >> K(d + \log K)/\pi_{\min}$, 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 $d >> K$)
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 two-step EM algorithm
[Dasgupta and Schulman, 2007]

Assumes $K$ spherical gaussians, separation $\|\mu_k^{true} - \mu_{k'}^{true}\| \geq C \sqrt{d} \sigma_k$

1. Pick $K' = O(K \ln K)$ centers $\mu_0^k$ at random from the data

2. Set $\sigma_0^k = \frac{d}{2} \min_{k \neq k'} \|\mu_0^k - \mu_0^{k'}\|^2$, $\pi_0^k = 1/K'$

3. Run one E step and one M step $\implies \{\pi_1^k, \mu_1^k, \sigma_1^k\}_{k=1:K'}$

4. Compute “distances” $d(\mu_1^k, \mu_1^{k'}) = \frac{\|\mu_1^k - \mu_1^{k'}\|}{\sigma_1^k - \sigma_1^{k'}}$

5. Prune all clusters with $\pi_1^k \leq 1/(4K')$

6. Run Fastest First Traversal with distances $d(\mu_1^k, \mu_1^{k'})$ to select $K$ of the remaining centers. Set $\pi_1^k = 1/K$.

7. Run one E step and one M step $\implies \{\pi_2^k, \mu_2^k, \sigma_2^k\}_{k=1:K}$

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

$$\|\mu_k - \mu_k^{true}\| \leq \|\text{mean}(C_k^{true}) - \mu_k^{true}\| + \varepsilon \sigma_k \sqrt{d}$$
High $d$

True model: centers $\mu_k^*$ at corners of hypercube, $\Sigma_k^* = \sigma I_d$ spherical equal covariances, $\pi_k^* = 1/K$

$n, K$, separation variable

Algorithm: EM with **Power initialization** and projection on $(K - 1)$-th principal subspace
Experimental exploration [Srebro et al., 2006] II

```
 Difference between likelihood of “fair” EM runs and EM from true centers
  • each run (random init)
  • run attaining max likelihood

figures from [Srebro et al., 2006]
```
Experimental exploration [Srebro et al., 2006]
Experimental exploration [Srebro et al., 2006] IV

- Practical limits vs theoretical limits

**Figures from [Srebro et al., 2006]**

<table>
<thead>
<tr>
<th>Source</th>
<th>Condition</th>
<th>Required Sample Size</th>
<th>Methodology</th>
</tr>
</thead>
<tbody>
<tr>
<td>Dasgupta 1999</td>
<td>$s &gt; 0.5d^{1/4}$</td>
<td>$n = \Omega(k^{log^2 1/5})$</td>
<td>Random projection, then mode finding</td>
</tr>
<tr>
<td>Dagupta Schulamn 2000</td>
<td>$s = \Omega(d^{1/4})$ (large $d$)</td>
<td>$n = poly(k)$</td>
<td>2 round EM with $\Theta(k \cdot logk)$ centers</td>
</tr>
<tr>
<td>Arora Kannan 2001</td>
<td>$s = \Omega(d^{1/4} \log d)$</td>
<td></td>
<td>Distance based</td>
</tr>
<tr>
<td>Vempala Wang 2004</td>
<td>$s = \Omega(k^{1/4} \log dk)$</td>
<td>$n = \Omega(d^3 k^2 \log(dk/s5))$</td>
<td>Spectral projection, then distances</td>
</tr>
</tbody>
</table>

**General mixture of Gaussians:**
- [Kannan Salmassian Vempala 2005] $s=\Omega(k^{5/2} \log(4kd))$, $n=\Omega(k^2 d \cdot log^5(d))$
- [Achlioptas McSherry 2005] $s>4k+o(k)$, $n=\Omega(k^2 d)$

$n \propto k^{1.5} - k^{1.6}$ for all $d$, separation
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 $K$
   - $(\alpha, \epsilon)$ Clusterability
   - 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
Selecting $K$

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

The **BIC (Bayesian Information) Criterion**

- let $\theta_K = \text{parameters for } \gamma_K$
- let $\#\theta_K = \text{number independent parameters in } \theta_K$

  e.g. for mixture of Gaussians with full $\Sigma_k$’s in $d$ dimensions

  $$\#\theta_K = K - 1 + Kd + \frac{Kd(d - 1)}{2}$$

- define

  $$BIC(\theta_K) = \text{log-likelihood}(\hat{\theta}_K) - \frac{\#\theta_K}{2} \ln n$$

- Select $K$ that maximizes $BIC(\theta_K)$
- selects true $K$ for $n \to \infty$ and other technical conditions (e.g. parameters in compact set)

For $d = 1$, mixture of known distributions: method by Walther
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

(from Nugent and Meila, 2010)
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

(from Nugent and Meila, 2010)
Selecting $K$ for hard clusterings

- based on statistical testing: the gap statistic (Tibshirani, Walther, Hastie, 2000)
- Stability methods
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
  $$g(K) = E_{P_0}[\mathcal{L}_K,n] - \mathcal{L}(\Delta_K) = \mathcal{L}_K^0 - \mathcal{L}(\Delta_K)$$
- choose $K^*$ corresponding to the largest gap
- nice: it can also indicate that data has no clusters
Practicalities

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

- like bootstrap, or crossvalidation
- **Idea** (implemented by [Ben-Hur et al., 2002])
  
  for each $K$
  
  1. perturb data $\mathcal{D} \rightarrow \mathcal{D}'$
  2. cluster $\mathcal{D}' \rightarrow \Delta'_K$
  3. 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]**
  1. divide data into 2 halves $\mathcal{D}_1$, $\mathcal{D}_2$ at random
  2. cluster (by EM) $\mathcal{D}_1 \rightarrow \Delta_1, \theta_1$
  3. cluster (by EM) $\mathcal{D}_2 \rightarrow \Delta_2, \theta_2$
  4. cluster $\mathcal{D}_1$ using $\theta_2 \rightarrow \Delta'_1$
  5. compare $\Delta_1, \Delta'_1$
  6. 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)
Fig. 2.1 Normalized stability scores. Left plots: Data points from a uniform density on $[0,1]^2$. Right plots: Data points from a mixture of four well-separated Gaussians in $\mathbb{R}^2$. The first row always shows the unnormalized instability $\text{Instab}$ for $K = 2, \ldots, 15$. The second row shows the instability $\text{Instab}_{\text{norm}}$ obtained on a reference distribution (uniform distribution). The third row shows the normalized stability $\text{Instab}_{\text{norm}}$.

(from von Luxburg, 2009)
(\alpha, \epsilon) \text{ Clusterability}

- let $\Delta^* = \arg\min \mathcal{L}(\Delta)$, $\mathcal{L}^* = \min \mathcal{L}(\Delta)$

- $\mathcal{D}$ is $(\alpha, \epsilon)$ clusterable w.r.t some cost $\mathcal{L}$ iff
  $\mathcal{L}(\Delta) \leq (1 + \alpha)\mathcal{L}^*$ implies $d(\Delta, \Delta^*) \leq \epsilon$

- if data contains clusters, algorithms work better/faster, guarantees are tighter

- many algorithms, theorems exist based on $(\alpha, \epsilon)$ clusterability

- when is $\mathcal{D}$ $(\alpha, \epsilon)$ clusterable?
  [Meilä, 2006] proves spectral theorem for a class of quadratic cost functions
Clustering with outliers

- What are outliers?
- let $p =$ 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 $K$
   - $(\alpha, \epsilon)$ Clusterability
   - 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
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

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

the mean shift $m(x)$

**Algorithm Simple Mean Shift**

**Input** Data $\mathcal{D} = \{x_i\}_{i=1:n}$, kernel $K(z)$, $h$

1. for $i = 1 : n$
   1. $x \leftarrow x_i$
2. iterate $x \leftarrow m(x)$ until convergence to $m_i$

2. group points with same $m_i$ in a cluster
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 $K(z)$, $h$

1. select $q$ points $\{x_j\}_{j=1:q} = \mathcal{D}_q \subseteq \mathcal{D}$ that cover the data well

2. for $j \in \mathcal{D}_q$
   1. $x \leftarrow x_j$
   2. iterate $x \leftarrow m(x)$ until convergence to $m_j$

3. group points in $\mathcal{D}_q$ with same $m_j$ in a cluster

4. assign points in $\mathcal{D} \setminus \mathcal{D}_q$ to the clusters by the **nearest-neighbor** method

$$k(i) = k(\arg\min_{j \in \mathcal{D}_q} ||x_i - x_j||)$$
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 Blurrring Mean Shift (GBMS)**

**Input** Data $\mathcal{D} = \{x_i\}_{i=1:n}$, Gaussian kernel $K(z)$, $h$

1. Iterate until **STOP**
   1. for $i = 1 : n$ compute $m(x_i)$
   2. for $i = 1 : n$, $x_i \leftarrow m(x_i)$

Remarks

- all $x_i$ converge to a single point
  $\Rightarrow$ need to stop before convergence
Empirical stopping criterion [Carreira-Perpinan, 2007]

- define $e_i^t = ||x_i^t - x_i^{t-1}||$ the change in $x_i$ at $t$
- define $H(e^t)$ the entropy of the histogram of $\{e_i^t\}$
- STOP when $\sum_{i=1}^n e_i^t / n < \text{tol}$ OR $|H(e^t) - H(e^{t-1})| < \text{tol'}$

**Convergence rate** If true for Gaussian, convergence is cubic

$$||x_i^t - x^*|| \le C ||x_i^{t-1} - x^*||^3$$

very fast!!
Algorithms based on level sets

Build a cluster tree = a filtration ≈ hierarchical clustering
Algorithms based on level sets

Build a cluster tree \( = \text{a filtration} \approx \text{hierarchical clustering} \)

**Algorithm Nugent-Stuetzle**

**Input**
Data \( \mathcal{D} = \{ x_i \}_{i=1:n} \), kernel \( K(z) \)

1. Compute KDE \( f(x) \) for chosen \( h \)
2. for levels \( 0 < l_1 < l_2 < \ldots < l_r < \ldots < l_R \geq \sup_x f(x) \)
   1. find level set \( L_r = \{ x | f(x) \geq l_r \} \) of \( f \)
   2. 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 \( C_{r,k} \subseteq \text{some cluster } C_{r-1,k'} \)
- therefore output is hierarchical clustering
- some levels can be pruned (if no change, 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 \( d \), \( L_r = \{ x_i \in \mathcal{D} \mid f(x_i) \geq l_r \} \)
- 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])
Chaudhuri-Dasgupta Algorithm

- Uses $k$-nearest neighbor graphs (filtration)
- Parameters $k$ (nearest neighbors) and $\alpha \in [1, 2]$
- for $r \geq 0$, $G_r = (V_r, E_r)$ with
  - $x_i \in V_r$ iff distance to $k$-nn of $x_i \leq r$
  - $(x_i, x_j) \in E_r$ iff $||x_i - x_j|| \leq \alpha r$

**Consistency Theorem** For any $\epsilon$ (separation parameter) and $\delta$ (confidence), $\alpha \in [\sqrt{2}, 2]$ (graph density), if $k = C \log^2 (1/\delta) \frac{d \log n}{\epsilon^2}$
for any two clusters $C, C'$ in cluster tree, there exists a level $r$ so that $C \cap D, C' \cap D$ are clusters at level $r$
Consistency Theorem

For any $\epsilon$ (separation parameter) and $\delta$ (confidence), $\alpha \in [\sqrt{2}, 2]$ (graph density), if
$$k = C \log^2(1/\delta) \frac{d \log n}{\epsilon^2}$$
for any two clusters $C, C'$ in cluster tree, there exists a level $r$ so that $C \cap D, C' \cap D$ are clusters at level $r$

- $r$ depends on $\lambda =$"bridge" between $C, C'$ (and $\sigma > 0$ "tube" width)

$$r^d \omega_d \lambda = \frac{k}{n} + \ldots \text{confidence term}$$

- it follows that the needed sample size $n$ at level $\lambda$

$$n = O \left( \frac{d}{\lambda \epsilon^2 (\sigma/2)^{d \omega_d}} \log \frac{d}{\lambda \epsilon^2 (\sigma/2)^{d \omega_d}} \right)$$

- this sample complexity $n$ is almost tight
- for $\alpha < \sqrt{2}$ sample complexity is exponential in $d$

- New results [Balakrishnan & Rinaldo 2013]

- Remark: algorithm(s) can be applied in any metric space
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 $q$, $C = 1/np$
   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)

2. 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 $[0,1]$
Remarks

- 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 $\frac{\text{margin errors}}{n} \to \frac{1}{nC} = p$ 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 $P_z$

$$P(z^1, \ldots, N) = \prod_{j=1}^{r} \theta_j^{N_j}$$

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 $\Gamma(p) = \int_0^\infty t^{p-1} e^{-t} dt$. 

\[ \text{where } \Gamma(p) = \int_0^\infty t^{p-1} e^{-t} dt. \]
Dirichlet process mixtures

- Model-based
- generalization of mixture models to
  - infinite $K$
- Bayesian framework
- denote $\theta_k =$ 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 \( \text{Normal}_{0,\sigma_0^2} \)
- 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 \( i - 1 \)
  
  \[ k(i) = \begin{cases} 
  k & \text{w.p. } \frac{n_k}{i-1+\alpha}, \ k = 1 : K \\
  K + 1 & \text{w.p. } \frac{\alpha}{i-1+\alpha} \end{cases} \]  

  1 if \( k(i) = K + 1 \) sample \( \mu_i \equiv \mu_{K+1} \) from \( \text{Normal}(0,\sigma_0^2) \)
  2 sample \( x_i \) from \( \text{Normal}(\mu_{k(i)}, \sigma^2) \)
  3 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

The clustering problem

- $\alpha, g_0, \beta, \{f\}$ given
- $D$ given
- wanted $\theta_1:n$ (not all distinct!)

Note:
- $\theta_1:n$ determines a hard clustering $\Delta$

Estimating $\theta_1:n$ cannot be solved in closed form
Usually solved by MCMC (Markov Chain Monte Carlo) sampling
Clustering with Dirichlet mixtures via MCMC

MCMC estimation for Dirichlet mixture

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

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

Iterate for \( i = 1 : n \) (reassign data to clusters)

1. resample \( k(i) \) by

\[
k(i) = \begin{cases} 
\text{existing } k & \text{w.p. } \frac{n_k-1}{n-1+\alpha} f(x_i, \theta_k) \\
\text{new cluster} & \text{w.p. } \frac{\alpha}{n-1+\alpha} \int f(x_i, \theta) g_0(\theta) d\theta
\end{cases}
\]  

(14)

2. if \( k(i) \) is new label, sample a new \( \theta_{k(i)} \) \( \propto g_0 f(x_i, \theta) \)

2. for \( k \in \{k(1 : n)\} \) (resample cluster parameters)

1. sample \( \theta_k \) from posterior \( g_k(\theta) \propto g_0(\theta, \beta) \prod_{i \in C_k} f(x_i, \theta) \)

\( g_k \) 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 in between parametric and non-parametric
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
1. Paradigms for clustering

2. Parametric clustering algorithms (K given)
   - Cost based / hard clustering
   - Model based / soft clustering

3. Issues in parametric clustering
   - Selecting $K$
   - $(\alpha, \epsilon)$ Clusterability
   - 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
**Similarity based clustering**

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

<table>
<thead>
<tr>
<th>Points</th>
<th>Features</th>
</tr>
</thead>
<tbody>
<tr>
<td>Image segmentation</td>
<td>pixels</td>
</tr>
<tr>
<td>Social network</td>
<td>people</td>
</tr>
<tr>
<td>Text analysis</td>
<td>words</td>
</tr>
</tbody>
</table>

The features are summarized by a single similarity measure \( S_{ij} \)

- e.g. \( S_{ij} = e^{\sum_k \alpha_k \text{feature}_k(i,j)} \) 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 $V = \{1, 2, \ldots, n\} \rightarrow \{x_1, x_2, \ldots, x_n\} \in \mathbb{R}^d$ 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 $\Delta = \{C_1:K\}$ 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

$S_{ij} \propto 1 / \text{dist}(i,j)$
A random walks view

Define

\[ P_{ij} = \frac{S_{ij}}{D_i} \quad \text{for all } i, j \in V \]

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

\[ P_{i,\text{red}} = Pr[i \rightarrow \text{red} | i] = \sum_{j \in \text{red}} P_{ij} \]
... 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 $\mathbb{R}^d$

Another look at $P_{i,c}$

<table>
<thead>
<tr>
<th>$i$</th>
<th>$P_{i,\text{red}}$</th>
<th>$P_{i,\text{yellow}}$</th>
</tr>
</thead>
<tbody>
<tr>
<td>1/5</td>
<td>4/5</td>
<td>4/5</td>
</tr>
<tr>
<td>1/5</td>
<td>4/5</td>
<td>4/5</td>
</tr>
<tr>
<td>1/5</td>
<td>4/5</td>
<td>4/5</td>
</tr>
<tr>
<td>1/5</td>
<td>4/5</td>
<td>4/5</td>
</tr>
<tr>
<td>1/5</td>
<td>4/5</td>
<td>4/5</td>
</tr>
<tr>
<td>1/5</td>
<td>4/5</td>
<td>4/5</td>
</tr>
<tr>
<td>1/5</td>
<td>4/5</td>
<td>4/5</td>
</tr>
<tr>
<td>1/5</td>
<td>4/5</td>
<td>4/5</td>
</tr>
</tbody>
</table>

**a piecewise constant function**

$f_{\text{red}} \equiv f_{\text{red}}$

$f_{\text{yel}} \equiv f_{\text{yel}}$
Some questions

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

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

Theorem [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

$$\sum_{j \in C'} P_{ij} = R_{CC'} \text{ whenever } i \in C, \text{ for all } C, C' \in \Delta$$
The spectral mapping

The spectral mapping: Data as elements of $v^2$, $v^3$

These eigenvectors are called piecewise constant (PC)
Spectral clustering in a nutshell

- Weighted graph
- Similarity matrix $S$
- Transition matrix $P$
- First $K$ eigenvectors of $P$
- $K$ clusters

$n$ vertices to cluster; observations are pairwise similarities

$n \times n$, symmetric $S_{ij} \geq 0$
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 $P = [P_{ij}]_{i,j=1}^{n}$ with
   
   $$P_{ij} \leftarrow S_{ij}/D_i, \text{ for } i,j = 1 : n$$

2. Compute the largest $K$ eigenvalues $\lambda_1 = 1 \geq \lambda_2 \geq \ldots \geq \lambda_K$ 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
   
   1. take $\mu_1$ randomly from $x_1, \ldots x_n$
   2. 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) $\mathcal{O}(nk)/\text{iteration}$.
- 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
  [Meilă and Shi, 2001a],[Bach and Jordan, 2006],[Meilă 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 $\lambda^t_k v^k$
- 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 \( s_{ii} \) ⇒ more likely \( i \) will become an exemplar ⇒ more clusters.
**Affinity Propagation**

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

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

Iterate the following steps until convergence:

1. $a_{ik} \leftarrow 0$ for $i, k = 1 : n$
2. for all $i$
   1. 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)
   2. for all $k$ update responsibilities
      
      $r_{ik} \leftarrow \begin{cases} 
      s_{ik} - s^*, & \text{if } k \not\in A_i^* \\
      s_{ik} - \max_{k'} \not\in A_i^* (s_{ik} + a_{ik}) & \text{otherwise}
      \end{cases}$

3. for all $k$ update availabilities
   1. $a_{kk} \leftarrow \sum_{i \neq k} [r_{ik}]_+$ where $[r_{ik}]_+ = r_{ik}$ if $r_{ik} > 0$ and 0 otherwise.
   2. 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 $K$
   - $(\alpha, \epsilon)$ Clusterability
   - 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 \( \mathcal{D} \), truth \( \Delta^* \); algorithm \( A \) produces \( \Delta \) is \( \Delta \) close to \( \Delta^* \)?

- given data \( \mathcal{D} \), 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
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 $\mathcal{L}$
  - $\Delta$ is valid if $\mathcal{L}(\Delta)$ is "small"
- but how small is "small"?
- Note: rescaling data may change $\mathcal{L}(\Delta)$
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

- \( \mathcal{L}(\Delta) = \text{const} - \text{trace} \ X^T(\Delta)AX(\Delta) \)

- 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
    $$\tilde{X}_{ik} = \begin{cases} 1 & i \in C_k \\ 0 & i \not\in C_k \end{cases} \text{ for } i = 1 : n, k = 1 : K$$
  - normalized (redundant) representation
    $$X_{ik} = \begin{cases} 1/\sqrt{|C_k|} & i \in C_k \\ 0 & i \not\in C_k \end{cases} \text{ for } i = 1 : n, k = 1 : K$$

Therefore $X_k^T X_{k'} = \delta(k, k')$, $X$ orthogonal matrix

- 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
Example: the K-means cost

Remember

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

In matrix form

\[ \mathcal{L}(\Delta) = -\frac{1}{2} X^T A X + \text{constant} \]

Where

\[ A_{ij} = x_i^T x_j \]

Is the Gram matrix of the data

If data centered, ie \( \sum_i x_i = 0 \) and \( Y \) rotated appropriately (see Meila, 2006)[Meilă, 2006]

\[ \mathcal{L}(\Delta) = -\frac{1}{2} Y^T A Y + \text{constant} \]

Assume k-means cost from now on
minimizing $\mathcal{L}(\Delta)$ is equivalent to

$$\max Y^T AY$$

over all $Y \in \mathbb{R}^{n \times (K-1)}$ that represent a clustering

a relaxation

$$\max Y^T AY$$

over all $Y \in \mathbb{R}^{n \times (K-1)}$ orthogonal

solution to relaxed problem is

$$Y^* = \text{eigenvectors}_{1:K-1} \text{ of } A$$

$$\mathcal{L}^* = \sum_{k=1}^{K-1} \lambda_k(A)$$

$\mathcal{L}^* = \text{constant}$ — $L^* = \text{trace } A - L^*$ is lower bound for $\mathcal{L}$

$$\mathcal{L}^* \leq \mathcal{L}(\Delta) \text{ for all } \Delta$$
A theorem (Meila, 2006)

Theorem

- define

\[ \delta = \frac{Y^T A Y - \sum_{k=1}^{K-1} \lambda_k}{\lambda_{K-1} - \lambda_K} \]

\[ \varepsilon(\delta) = 2\delta[1 - \delta/(K - 1)] \]

- define \( p_{\min}, p_{\max} = \frac{\min, \max |C_k|}{n} \)

- then, whenever \( \varepsilon(\delta) \leq p_{\min} \), we have that

\[ d_{ME}(\Delta, \Delta^{opt}) \leq \varepsilon(\delta)p_{\max} \]

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 \( d_{ME}(\Delta, \Delta^{opt}) \leq p_{\min} \)
- 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
data $d = 35$, $\sigma = 0.4$

$n = 100$

$n = 200$

$n = 1000$
1 Paradigms for clustering

2 Parametric clustering algorithms (K given)
   - Cost based / hard clustering
   - Model based / soft clustering

3 Issues in parametric clustering
   - Selecting $K$
   - $(\alpha, \epsilon)$ Clusterability
   - 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

- Non-parametric mixtures
- Non-parametric Bayesian (i.e. Dirichlet Process mixtures)
- Information bottleneck
- (spectral clustering and manifold learning)

- Hierarchical clustering – single linkage
- 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 identifyability appear
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 $\mathcal{L}$ the least, until $K$ clusters left
- Greedy, recursive algorithm, $O(n^3)$ operations
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.
  - [?] 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
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
  
  1. $p \leftarrow Pp$
  2. for $j = 1 : n$, if $p_j < \varepsilon$ set $p_j = 0$

  Output $C(i) = \{ j \mid p_j > 0 \}$

  - $C(i)$ is the set of items attainable from $i$ by a “likely” path
  - Original algorithm has sparsest cut guarantees
  
  Used as subroutine by other algorithms.
Links

- Yee Whye Teh’s tutorial on DP Mixtures
  http://mlg.eng.cam.ac.uk/tutorials/07/ywt.pdf
- Lecture on exponential family models http:
On spectral learning of mixtures of distributions.

Learning mixtures of arbitrary gaussians.

Spectral methods for automatic multiscale data clustering.

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.

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.
*Journal of the Royal Statistical Society B*.

Multivariate information bottleneck.

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

A best possible heuristic for the k-center problem.


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.

Humana Press, Springer.

X-means: Extending K-means with efficient estimation of the number of clusters.
Unsupervised spectral learning.
In Jaakkola, T. and Bachhus, F., editors, Proceedings of the 21st Conference

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.
In Proceedings of the 23rd International Conference on Machine Learning
(ICML).

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.