Clustering – A tutorial overview

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

MLSS Taipei 2006

Outline

- What is clustering?
- Paradigms/Algorithms for clustering
  - Model-based (probabilistic) vs cost-based
  - Parametric (K known) vs Non-parametric
  - Vector data (in $\mathbb{R}^d$) vs graph data
- Fundamental issues
- Cluster validation
- Modern trends

What is clustering? Problem and Notation

- Finding groups in data
  Examples:
- **Problem** Given n data points, separate them into clusters

n	=	number of data points
K	=	number of clusters ($K << n$)
Δ	=	a partition $\Delta = \{C_1, C_2, \ldots, C_K\}$
$k(i)$	=	the label of point i
$\mathcal{L}(\Delta)$	=	cost (loss) of Δ (to be minimized)

- Hard clustering Δ
- Soft clustering $\gamma = \{\gamma_{ki}\}_{k=1:K}^{i=1:n}$
  γ_{ki} = the degree of membership of point i to cluster k

$$\sum_k \gamma_{ki} = 1 \quad \text{for all } i$$

Usually associated with a probabilistic model
Cost $\mathcal{L}(\gamma)$ = $-$likelihood (typically)

Paradigms

Data = vectors $\{x_i\}$ in $\mathbb{R}^d$

Parametric (K known)	Model based [soft]
	Cost based [hard]
Non-parametric (K determined by algorithm)	Dirichlet process [soft]
	Information bottleneck [soft]
	Modes of distribution [hard]
	Gaussian blurring meab shift [hard]

Data = (weighted) graph $[S_{ij}]_{i,j=1:n}$, $S_{ij} = S_{ji} \geq 0$

Graph partitioning

- spectral clustering (hard, parametric, cost based)
- typical cuts (hard non-parametric, cost based)

Classification vs Clustering

	Classification	**Clustering**
Cost $\mathcal{L}$	Expected error (cost)	many! (probabilistic or not)
	Supervised	Unsupervised
Generalization	Performance on new data is what matters	Performance on current data is what matters
K	Known	Unknown
"Goal"	Prediction	Exploration Lots of data to explor
Stage of field	Mature	Young (growing fast now)

Parametric clustering

Cost functions and algorithms

- Cost based
  - Single linkage
  - Min diameter
  - K-means
  - K-medians
- Model based
  - EM algorithm
  - "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. Deleted the largest $K-1$ edges

- **Cost** $\mathcal{L}(\Delta) = -\min_{k,k'} \text{distance}(C_k, C_{k'})$ where $\text{distance}(A, B) = \operatorname{argmin}_{x \in A,\, y \in B} ||x - y||$
- Running time $\mathcal{O}(n^2)$ one of the very costs $\mathcal{L}$ that can be optimized in *polynomial* time
- Sensitive to outliers

Minimum diameter clustering

- **Cost** $\mathcal{L}(\Delta) = \max_k \underbrace{\max_{i,j \in C_k} ||x_i - x_j||}_{\text{diameter}}$
  - Mimimize the diameter of the clusters
  - assumes K given
  - Optimizing this cost is NP-hard
- **Algorithms**
  - **Fastest first traversal** – a factor 2 approximation for the min cost
  - similar to **Anchor** algorithm of Moore
  - can be extended to hierarchical algorithm
  - all algorithms define anchors $a_{1:K} \in \mathcal{D}$

Algorithm Fastest First Traversal

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

1. pick a_1 at random from $\mathcal{D}$
2. for $k = 2 : K$
   $a_k \longleftarrow \operatorname{argmax}_{\mathcal{D}} \text{distance}(x_i, \{a_{1:k-1}\})$
3. for $i = 1 : n$ (assign points to anchors)
   $k(i) = k$ if a_k is the nearest anchor to x_i

K-means clustering

This is originally an algorithm for vector quantization

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 $i = 1 : n$

$$k(i) \;=\; \operatorname*{argmin}_k ||x_i - \mu_k||$$

(assign points to clusters $\Rightarrow$ new clustering)

2. for $k = 1 : K$

$$\mu_k \;=\; \frac{1}{|C_k|} \sum_{i \in C_k} x_i \qquad (1)$$

(recalculate centers)

- Convergence
  - if Δ doesn't change at iteration m it will never change after that
  - provable: convergence in finite number of steps

The K-means cost

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

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

  Sketch of proof
  - The solution to

$$\mu_k = \min_{\mu \in \mathbb{R}^d} \sum_{i \in C_k} ||x_i - \mu||^2 \qquad (2)$$

    is given by (1)
  - step 1: reassigning the labels can only decrease $\mathcal{L}$
  - step 2: reassigning the centers μ_k can only decrease $\mathcal{L}$
- Therefore, **K-means** converges to a **local** minimum of the cost $\mathcal{L}$
- Initialization matters (see later)

*(or leaves unchanged)

- The distortion can also be expressed as
  - sum of (squared) intracluster distances

$$\mathcal{L}(\Delta) = \frac{1}{2}\sum_{k=1}^{K}\sum_{i,j\in C_k}||x_i - x_j||^2 + \text{constant} \quad (3)$$

  - (negative) sum of (squared) intercluster distances

$$\mathcal{L}(\Delta) = -\frac{1}{2}\sum_{k=1}^{K}\sum_{i\in C_k}\sum_{j\notin C_k}||x_i - x_j||^2 + \text{constant} \quad (4)$$

Proof of (3)

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

Proof of (4)

$$\sum_k \sum_{i,j \in C_k} ||x_i - x_j||^2 = \underbrace{\sum_{i=1}^{n} \sum_{j=1}^{n} ||x_i - x_j||^2}_{\text{independent of } \Delta} - \sum_k \sum_{i \in C_k} \sum_{j \notin C_k} ||x_i - x_j||^2$$

K-means: Practical issues

- **Initialization** of $\mu_{1:K}$. Heuristics:
  - K points sampled from data
  - K most remote points
  - data mean μ + random perturbations
    $\mu = \frac{1}{n}\sum_{i=1}^{n} x_i$
  - data mean μ + random orthogonal perturbations ($K \leq d$)
- Preprocessing
  - centering $x_i \leftarrow x_i - \mu$
    sets $\mu \leftarrow 0$
    not essential but numerically useful
  - scaling of different coordinates
    affects algorithms' outcome!

The Ward algoritm

- 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, $\mathcal{O}(n^3)$ operations
- Is an agglomerative clustering algorithm

Model based clustering: Mixture models

- The mixture density

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

- $f_k(x)$ = the components of the mixture
  - each is a density
  - if $f_k = Normal(\mu_k, \Sigma_k)$ we call f_k a mixture of Gaussians
  - will assume f_k Gaussian for simplicity
- π_k = the mixing coefficients/mixing proportions (a convex combination)
- A probabilistic model for clustering
- Degree of membership

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

Criterion for clustering: Max likelihood

- denote $\theta = (\pi_{1:K}, \mu_{1:K}, \Sigma_{1:K})$
  (or other parameters if components not Gaussian)
- log likelihood

$$l(\theta) = \ln \prod_{i=1}^{n} f(x_i) = \sum_{i=1}^{n} \ln \sum_{k} \pi_k f_k(x_i) \qquad (7)$$

- denote $\theta^{ML} = \operatorname{argmax}_{\theta} l(\theta)$
- θ^{ML} determines a soft clustering γ
- a soft clustering γ determines a θ (see later)
- Therefore we can write

$$\mathcal{L}(\gamma) = -l(\theta(\gamma))$$

Algorithms for model-based clustering

Maximize the (log-)likelihood w.r.t θ

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

w.h.p = with high probability (over samples)

The EM Algorithm – Idea

- Define the indicator variables

$$z_{ik} = \begin{cases} 1 & \text{if } i \in C_k \\ 0 & \text{if } i \notin C_k \end{cases} \tag{8}$$

  denote $\bar{z} = \{z_{ki}\}_{k=1:K}^{i=1:n}$
- Define the complete log-likelihood

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

- z_{ki} are not known, but for any values of the parameters θ, we can compute their expectations $E[z_{ki}] = \gamma_{ki}$
- Then

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

$$= \sum_{i=1}^{n} \sum_{k=1}^{K} \gamma_{ki} \ln \pi_k + \sum_{i=1}^{n} \sum_{k=1}^{K} \gamma_{ki} \ln f_k(x_i) \tag{11}$$

- If γ_{ki} known, π_k, μ_k, Σ_k can be obtained by separately maximizing the terms of $E[l_c]$ (Maximization)
- If θ known, γ_{ki} can be obtained by (6) (Expectation)

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

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

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

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

- Γ_k represents the "number of points" in cluster k
- $\pi_{1:K}, \mu_{1:K}, \Sigma_{1:K}$ are the maximizers of $l_c(\theta)$ in (11)

*Σ_k need to be symmetric, positive definite matrices

Brief analysis of EM

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

- each step of EM increases $Q(\theta, \gamma)$
- Q converges to a local maximum
- at every local max 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 $\theta \longrightarrow \theta^{ML}$

The M step in special cases

	M step
$\Sigma_k = \Sigma$ "same shape & size" clusters	$\Sigma \leftarrow \frac{\sum_{i=1}^{n}\sum_{k=1}^{K}\gamma_{ki}(x_i-\mu_k)(x_i-\mu_k)^T}{n}$
$\Sigma_k = \sigma_k^2 I_d$ "round" clusters	$\sigma_k^2 \leftarrow \frac{\sum_{i=1}^{n}\gamma_{ki}\|\|x_i-\mu_k\|'^2}{d\Gamma_k}$
$\Sigma_k = \sigma^2 I_d$ "round, same size" clusters	$\sigma^2 \leftarrow \frac{\sum_{i=1}^{n}\sum_{k=1}^{K}\gamma_{ki}\|\|x_i-\mu_k\|\|^2}{nd}$

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

EM – Practical issues

- Initialization is important
  - See the heuristic for 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 $\sigma_k^2 =$ max eigenvalue(Σ_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 θ

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

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

The Vempala-Wang algorithm

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 μ_k increases

(Proved by Vempala & 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-th principal subspace $\Rightarrow \{y_i\} \in \mathbb{R}^K$
2. do distance-based "harvesting" of clusters in $\{y_i\}$

Other "CS" algorithms

- (Dasgupta, 2000) round, equal sized Gaussian, random projection
- (Arora & Kannan, 2001) arbitrary shaped Gaussian, distances
- (Achlioptas & 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 heuritics for EM K-means
  - project data on principal subspace (when $d >> K$)

But

- assumes large separation (unrealistic)
- assumes concentration of f_k's (known)
- assumes K known
- tries to find perfect solution (too ambitious)

Clustering – A tutorial overview

Part 2

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

MLSS Taipei 2006

Outline

- What is clustering?
- Paradigms/Algorithms for clustering
  - Model-based (probabilistic) vs cost-based
  - Parametric (K known) vs Non-parametric
  - Vector data (in $\mathbb{R}^d$) vs graph data
- Fundamental issues
  - Selecting K
  - outliers
- Cluster validation
  - internal
  - external - distances between clusterings
- Curent trends in clustering

Selecting K

- In practice, clustering algorithm is run for $K = K_{min} : K_{max}$
  $\Rightarrow$ $\Delta_{K_{min}}, \ldots \Delta_{K_{max}}$ or $\gamma_{K_{min}}, \ldots \gamma_{K_{max}}$
  then choose best Δ_K (or γ_K) from among them
- Typically: increase $K \Rightarrow \mathcal{L}$ decreases
- Need to "penalize" $\mathcal{L}$ with function of number parameters
- mixture models: **BIC**
  - **BIC** = Bayesian Information Criterion
  - let θ_K = parameters for γ_K

$$BIC(\theta_K) = \ln l(\theta_K) - \frac{\#\theta_K}{2} \ln n$$

    $\#\theta_K$=number independent parameters in θ_K
    - e.g $\#\theta_K = K-1+K[d+d(d-1)/2]$ for mixture of Gaussians with full Σ_k's
  - Rule: choose $K = \text{argmax} BIC(\theta_K)$
  - selects true K for $n \to \infty$ and other technical conditions (e.g parameters in compact set)
- hard clusterings
  - based on statistical testing: the **gap** statistic (Tibshirani, Walther, Hastie, 2000)
  - the **Krzanowski-Lai (KL)**, 1985 statistic
  - **X-means** heuristic (Pelleg & Moore, 2000): splits/merge clusters based on statistical tests of Gaussianity
  - 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

$$g(K) = E_{P_0}[\mathcal{L}_{K,n}] - \mathcal{L}(\Delta_K) = \mathcal{L}^0_K - \mathcal{L}(\Delta_K)$$

- choose K^* corresponding to the largest gap
- nice: it can also indicate that data has no clusters

Practicalities

- $\mathcal{L}_K^0 = E_{P_0}[\mathcal{L}_{K,n}]$ can rarely be computed in closed form
  (when P_0 very simple)
- otherwise, estimate $\mathcal{L}_K^0$ be Monte-Carlo sampling
  i.e generate B samples from P_0 and cluster them
- if sampling, variance s_K^2 of estimate $\widehat{\mathcal{L}}_K^0$ 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

The KL statistic

- Heuristic
- define $w_K = K^{2/d}\mathcal{L}^V(\Delta_K)$ (lower is better)
- a good K is indicated by "large" jump between $K-1$ and K
- they propose

$$diff(K) = w_{K-1} - w_K$$

choose K that maximixes relative jump $\left|\frac{diff(K)}{diff(K+1)}\right|$

Stability methods for choosing K

- like bootstrap, or crossvalidation
- **Idea**

  for each K

  1. perturb data $\mathcal{D} \rightarrow \mathcal{D}'$
  2. cluster $\mathcal{D}' \rightarrow \Delta'_K$
  3. compare Δ_K, Δ'_K. Are they similar?

  If Δ_K is stable to perturbations then we assume K is correct

- This idea implemented by Ben-Hur et al. (20002) for hard clusterings
- for model-based(soft) clustering 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 Δ_1, Δ'_1
  6. repeat B times and average the results
     - repeat for each K
     - select K where Δ_1, Δ'_1 are closest on average (or most times)

- these methods are supported by experiments
- **Not supported by theory!**
- ...work is in progress on this topic...

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) Σ_{K+1}
  - K-means and EM
    - robust means and variances

      e.g eliminate smallest and largest $pn/2$ samples in mean computation (trimmed mean)
    - K-medians
  - single-linkage: do not count clusters with $< r$ points

Is K meaningful when outliers present?

- alternative: non-parametric clustering

To come next

- Non-parametric clustering
- Cluster validation
  - external
  - internal
- Current trends in clustering

Algorithms 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

Kernel density estimation

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

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

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 $||z||$

The Kernel density estimate (KDE)

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

- f is sum of Gaussians centered on each x_i
- f is smoother (less variation) if h larger
- **caveat:** dimension d can't be too large

The Nugent-Stuetzle algorithm

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 $0 < l_1 < l_2 < \ldots < l_r < \ldots < l_R \geq \sup_x f(x)$
   (a) find level set $L_r = \{x \,|\, f(x) \geq l_r\}$ of f
   (b) 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 (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, $L_r = \{x_i \in \mathcal{D} \,|\, f(x_i) \geq l_r\}$
- 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 possile by resampling

Mean shift algorithms

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

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

$$x = \underbrace{\frac{\sum_{i=1}^{n} x_i K(\frac{x-x_i}{h})}{\sum_{i=1}^{n} K(\frac{x-x_i}{h})}}_{\text{the mean shift}} = m(x)$$

Algorithm Simple Mean Shift

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

1. for $i = 1 : n$
   (a) $x \leftarrow x_i$
   (b) 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

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 $j \in \mathcal{D}_q$
   (a) $x \leftarrow x_j$
   (b) 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 $\mathcal{D} \backslash \mathcal{D}_q$ to the clusters by the nearest-neighbor method

$$k(i) = k(\operatorname*{argmin}_{j \in \mathcal{D}_q} ||x_i - x_j||)$$

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
   (a) for $i = 1 : n$ compute $m(x_i)$
   (b) 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-Perpignan)

- 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 $\sum_{i=1}^n e_i^t/n <$`tol` OR $|H(e^t) - H(e^{t-1})| <$`tol'`

Convergence rate If true f Gaussian, convergence is cubic

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

very fast!!

Support Vector (SV) clustering

Idea

1. build a kernelized density estimator
   - gaussian kernel $K(x, x') = e^{q||x-x'||^2}$
2. threshold the density and find connected components

Algorithm SV

Input data $\mathcal{D}$, 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 $R(tx_i + (1-t)x_j) \leq R$ for t on a grid over [0,1]

Remarks

- 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

Dirichlet process mixtures

- Model-based
- generalization of mixture models to
  - infinite K
  - Bayesian framework
- denote θ_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 on the number of clusters with hyperparameter $\alpha > 0$
- very flexible model

A sampling model for the data

- **Example: Gaussian mixtures**, $d = 1$, $\sigma_k = \sigma$ fixed
- $\theta = \mu$
- prior for μ $Normal(0, \sigma_0^2)$
- 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 $i - 1$
    1.
    $$k(i) = \begin{cases} k & \text{w.p } \frac{n_k}{i-1+\alpha}, \; k = 1 : K \\ K+1 & \text{w.p } \frac{\alpha}{i-1+\alpha} \end{cases} \tag{12}$$
    2. if $k(i) = K+1$ sample $\mu_i \equiv \mu_{K+1}$ from $Normal(0\sigma_0^2)$
    3. sample x_i from $Normal(\mu_{k(i)}, \sigma^2)$
- can be shown that the distribution of $x_{1:n}$ is interchangeable (does not depend on data permutation)
- hyperparameters
  - σ_0 controls spread of centers
  - α controls number of cluster centers
  - α large $\Rightarrow$ many clusters
  - cluster sizes non-uniform (larger clusters attract more new points)
  - many single point clusters possible

general Dirichlet mixture

- cluster densities $\{f(x, \theta)\}$
- cluster membership $k(i)$ sampled as in (12)
- parameters θ sampled from prior $g_0(\theta, \beta)$
- x_i sampled from $f(x, \theta_{k(i)})$
- Model Hyperparameters α, β

Clustering with Dirichlet mixtures

The clustering problem

- $\alpha, g_0, \beta, \{f\}$ given
- $\mathcal{D}$ given
- wanted $\theta_{1:n}$ (not all distinct!)
- sometimes also hard clustering Δ
- note: $\theta_{1:n}$ and prior g_0 determine a soft clustering

This problem cannot be solved in closed form

Usually solved by MCMC (Markov Chain Monte Carlo) sampling

MCMC estimation for Dirichlet mixture

Input $\alpha, g_0, \beta, \{f\}$, $\mathcal{D}$

State cluster assignments $k(1:n)$,
parameters θ_k, $k \in \{k(1:n)\}$

Iterate

1. (reassign data to clusters)
   for $i = 1:n$
   (a) if $n_{k(i)} = 1$ delete this cluster and its $\theta_{k(i)}$
   (b) resample $k(i)$ as in (12)
   (pretend i is n-th point)
   (c) if $k(i)$ is new label, sample a new $\theta_{k(i)}$ from g_0
2. (resample cluster parameters)
   for $k \in \{k(1:n)\}$
   (a) sample θ_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 maximizes posterior of $k(1:n), \{\theta_k\}$ given data
- do not depend critically on initialization
- K and outliers selected automatically, naturally
- require hyperparameters

Note that **Dirichlet mixture** is inbetween 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 (a few large clusters and many small ones)

Cluster validation

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

External cluster validation

Scenarios

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

A distance $d(\Delta, \Delta')$ 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 $P = M/n$

$$p_{kk'} = \frac{m_{kk'}}{n}$$

- define also $p_k = n_k/n,\ p'_{k'} = n'_{k'}/n$

The Misclassification Error (ME) distance

- assume $K \leq K'$
- let Π_K = the set of all K-permutations and $\pi \in \Pi_K$
- Define the Misclassification Error (ME) distance d_{ME}

$$d_{ME} = 1 - \max_{\pi \in \Pi_K} \sum_{k=1}^{K} p_{k,\pi(k)}$$

- 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 (Hungarian)** algorithm
- Is a metric: symmetric, ≥ 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
- recommended: for small K

The Variation of Information (VI) distance

Clusterings as random variables

- 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

- 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 Δ' given Δ

$$H_{\Delta'|\Delta} = -\sum_k p_k \sum_{k'} \frac{p_{kk'}}{p_k} \ln \frac{p_{kk'}}{p_k}$$

- 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)

$$\begin{aligned} I_{\Delta,\Delta} &= H_{\Delta} + H_{\Delta'} - H_{\Delta',\Delta} \\ &= H_{\Delta'} - H_{\Delta'|\Delta} \end{aligned}$$

- 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

$$\begin{aligned} d_{VI}(\Delta, \Delta') &= H_\Delta + H_{\Delta'} - 2I_{\Delta',\Delta} \\ &= H_{\Delta|\Delta'} + H_{\Delta'|\Delta} \end{aligned}$$

- Interpretation: d_{VI} is the sum of information gained and information lost when labels are switched from $k()$ to $k'()$
- d_{VI} symmetric, ≥ 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 properties and comparison criteria

- n-invariance
- locality
- convex additivity
- indices i
  - from statistics (Rand, adjusted Rand, Jaccard, ...)
  - in [0,1], with $i(\Delta, \Delta') = 1$ for $\Delta = \Delta'$
  - not so good properties

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

Quadratic cost

- $\mathcal{L}(\Delta) = const - \text{trace}\, X^T(\Delta) A X(\Delta)$
- with X = matrix reprentation for Δ
- then, if cost value $\mathcal{L}(\Delta)$ small, we can prove that clustering Δ is almost optimal
- This holds for K-means (weighted, kernelized) and several graph partioning costs (normalized cut,average association, etc)

Matrix Representations

- matrix reprentations for Δ
  - 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 reprentation
    - 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 $K-1$ dimesional 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)

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

- Assume k-means cost from now on

A spectral lower bound

- to minimize $\mathcal{L}(\Delta)$ 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 provlem is

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

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

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

$$\mathcal{L}^* \leq \mathcal{L}(\Delta) \text{ for all } \Delta$$

A theorem (Meila, 2006)

Theorem

- define
$$\delta = \frac{Y^T A Y - \sum_{k=1}^{K-1} \lambda_k}{\lambda_{K-1} - \lambda_K}$$
- define
$$\epsilon(\delta) = 2\delta[1 - \delta/(K-1)]$$
- define $p_{min}, p_{max} = \frac{\min,\max |C_k|}{n}$
- then, whenever $\epsilon(\delta) \leq p_{min}$, we have that
$$d_{ME}(\Delta, \Delta^{opt}) \leq \epsilon(\delta) p_{max}$$
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 $d_{ME}(\Delta, \Delta^{opt}) \leq p_{pmin}$
- does not apply for all data sets
- only if a good clustering is found
- intuiton: if data well clustered, $K-1$ principal subspace is aligned with cluster centers

Heurisitcs

- **Gap** heuristic
- K-means: use δ as above
- single linkage:
  - define l_r length of r-th edge added to MST

$$\underbrace{l_1 \leq l_2 \leq \ldots l_{n-K}}_{\text{intracluster}} \leq \underbrace{l_{n-K+1} \leq \cdots}_{\text{deleted}}$$

  - $l_{n-K}/l_{n-K+1} \leq 1$ should be small
- min diameter:

$$\frac{\mathcal{L}(\Delta)}{\max_{i,j \in \mathcal{D}} ||x_i - x_j||}$$

$$\frac{\mathcal{L}(\Delta)}{\min_{k,k'} \text{distance}(C_k, C_{k'})}$$

- etc

What I didn't talk about

- Graph partitioning
- K-medians algorithms (Charikar and Guha, 1999, Bradley & Mangasarian)
- Hierarchical clustering
- Ensembles of clusterings, consensus clustering, and clustering clusterings
- Subspace clustering (or clustering on subsets of attributes), bi-clustering, partial clustering

Dimitris Achlioptas and Frank McSherry. On spectral learning of mixtures of distributions. In Peter Auer and Ron Meir, editors, *18th Annual Conference on Learning Theory, COLT 2005*, pages 458–471, Berlin/Heidelberg, 2005. Springer.

Asa Ben-Hur, Andre Elisseeff, and Isabelle Guyon. A stability based method for discovering structure in clustered data. In *Pacific Symposium on Biocomputing*, pages 6–17, 2002.

M. Charikar and S. Guha. Improved combinatorial algorithms for the facility location and k-median problems. In *40th Annual Symposium on Foundations of Computer Science*, pages 378–388, 1999.

Sanjoy Dasgupta. Learning polytrees. In Kathryn Blackmond Laskey and Henri Prade, editors, *Proceedings of the 15th Conference on Uncertainty in AI*, San Francisco, CA, 1999. Morgan Kaufmann.

Sanjoy Dasgupta. Experiments with random projection. In *UAI '00: Proceedings of the 16th Conference on Uncertainty in Artificial Intelligence*, pages 143–151, San Francisco, CA, USA, 2000. Morgan Kaufmann Publishers Inc.

Tilman Lange, Volker Roth, Mikio L. Braun, and Joachim M. Buhmann. Stability-based validation of clustering solutions. *Neural Comput.*, 16(6):1299–1323, 2004.

Dan Pelleg and Andrew Moore. X-means: Extending K-means with efficient estimation of the number of clusters. In Ivan Bratko and Saso Džeroski, editors, *Proceedings of the 17th International Conference on Machine Learning*, pages 727–734, San Francisco, CA, 2000. Morgan Kaufmann.

Santosh Vempala and Grant Wang. A spectral algorithm for learning mixtures of distributions. *Journal of Computer Systems Science*, 68(4):841–860, 2004.

J.H. Ward. Hierarchical grouping to optimize an objective function. *J. Amer. Statist. Assoc.*, 58:236 – 244, 1963.