Lecture 5

Lecture II – Clustering – Part II: Non-parametric clustering

Marina Meilă
mmp@stat.washington.edu

Department of Statistics
University of Washington

CSE 547/STAT 548
Winter 2022

1. Paradigms for clustering ✓
2. Methods based on non-parametric density estimation
   - Mean-Shift ✓
   - Level set methods ↑
   - KDE
3. Model-based: Dirichlet process mixture models ←

DBSCAN

What is clustering? Problem and Notation

- **Informal definition** **Clustering** = Finding groups in data
- **Notation**

$\mathcal{D}$	=	$\{x_1, x_2, \ldots x_n\}$ a **data set**
n	=	number of **data points**
K	=	number of **clusters** ($K << n$)
Δ	=	$\{C_1, C_2, \ldots, C_K\}$ a partition of $\mathcal{D}$ into disjoint subsets
$k(i)$	=	the **label** of point i
$\mathcal{L}(\Delta)$	=	cost (loss) of Δ (to be minimized)

- **Second informal definition** **Clustering** = given n **data points**, separate them into K **clusters**
- Hard vs. soft clusterings
  - **Hard** clustering Δ: an item belongs to only 1 cluster
  - **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)

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

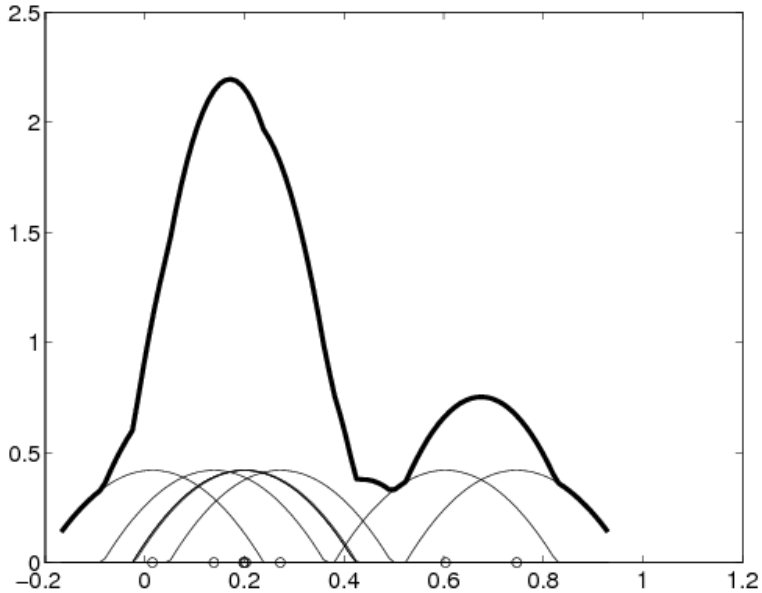
- peak finding algorithms Mean-shift algorithms
- level sets based algorithms
  - Nugent-Stuetzle, Support Vector clustering
- Information Bottleneck ?

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
- f is smoother (less variation) if h larger
- **caveat:** dimension d can't be too large

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

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

Local max of f is solution of implicit equation

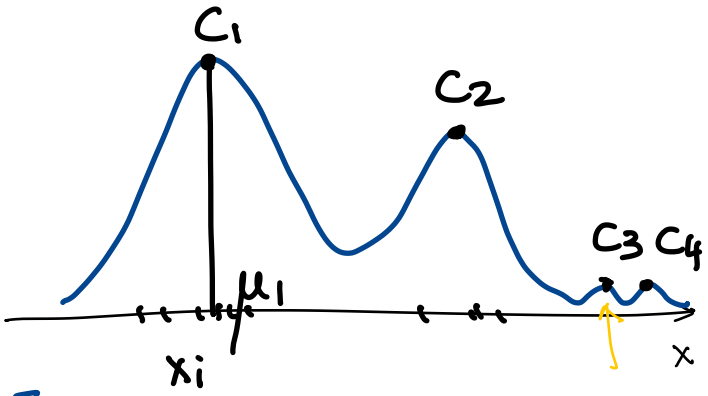
$$x = \underbrace{\frac{\sum_{i=1}^{n} x_i K(\frac{x-x_i}{h})}{\sum_{i=1}^{n} K(\frac{x-x_i}{h})}}_{\text{the mean shift } m(x)}$$

Algorithm Simple Mean Shift

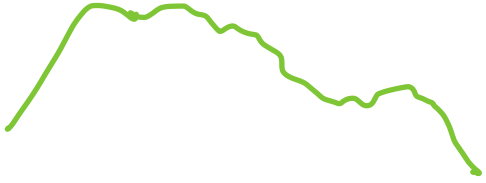
Input Data $\mathcal{D} = \{x_i\}_{i=1:n}$, kernel $K(z)$, h

1. for $i = 1 : n$
   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

- MeanSh$(x_i) \rightarrow \mu_k \Leftrightarrow x_i \in C_k$



KDE h



- Issues
  - $O(n^2)$ operations – subsample! $n' \ll n$
  - $h \rightarrow$ influences K (number clusters)
    - $h \nearrow$ smoother KDE $\Rightarrow$ K smaller
    - $h \searrow$ $\rightarrow K \nearrow$
  - spurious peaks "not significant enough"

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

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

[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
   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 ?

- 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!!

Clustering by Level Sets of KDE_h

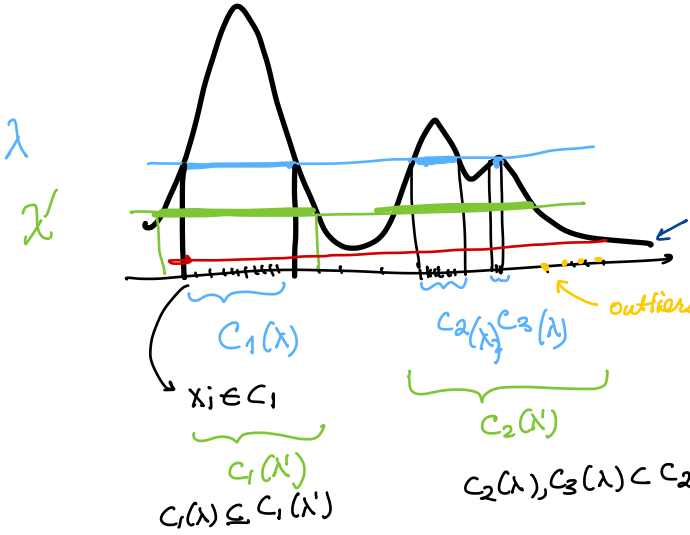
Super

CORRECT

$L_\lambda = \{x \in \mathbb{R}^d \mid f_h \geq \lambda\}$ → hard

Alg

1. find $\{x_i \mid f_h(x_i) \geq \lambda\} = \hat{L}_\lambda$
2. $\Delta =$ "Connected Components" (L_λ)

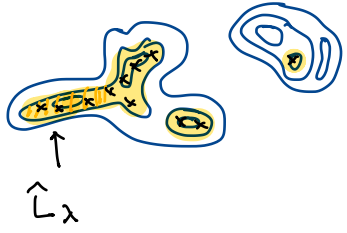


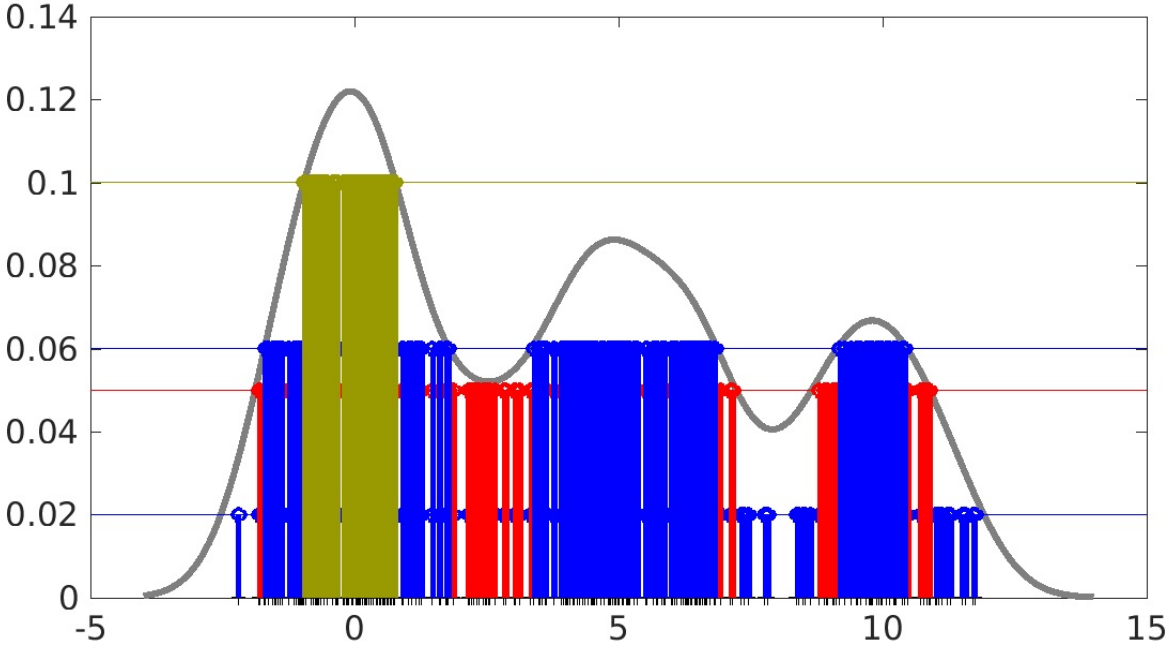
- $C_1(\lambda) \cup C_2(\lambda) \cup C_3(\lambda) = \{x_i \mid f_h(x_i) \geq \lambda\}$

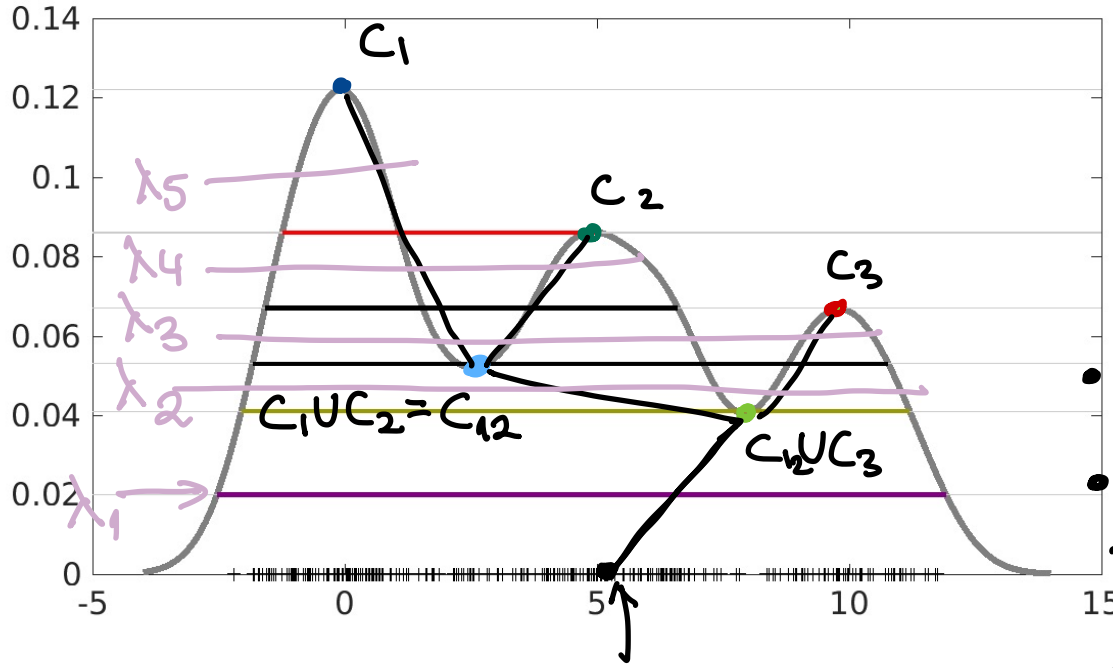
Super level λ set of data points

L_λ

2D :




0.14
0.12
0.1
0.08
0.06
0.04
0.02
0
-5
0
5
10
15



- Cluster tree
- Only finite set of λ levels needed

- $\{ \Delta(\lambda), \lambda \text{ in Levels} \}$ = a hierarchical clustering

$\Delta = \{C_1, \dots, C_K\}$ a clustering

Practically:
- fine grid on λ not needed
- only interested in stable levels

The Nugent-Stuetzle algorithm

Algorithm Nugent-Stuetzle

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

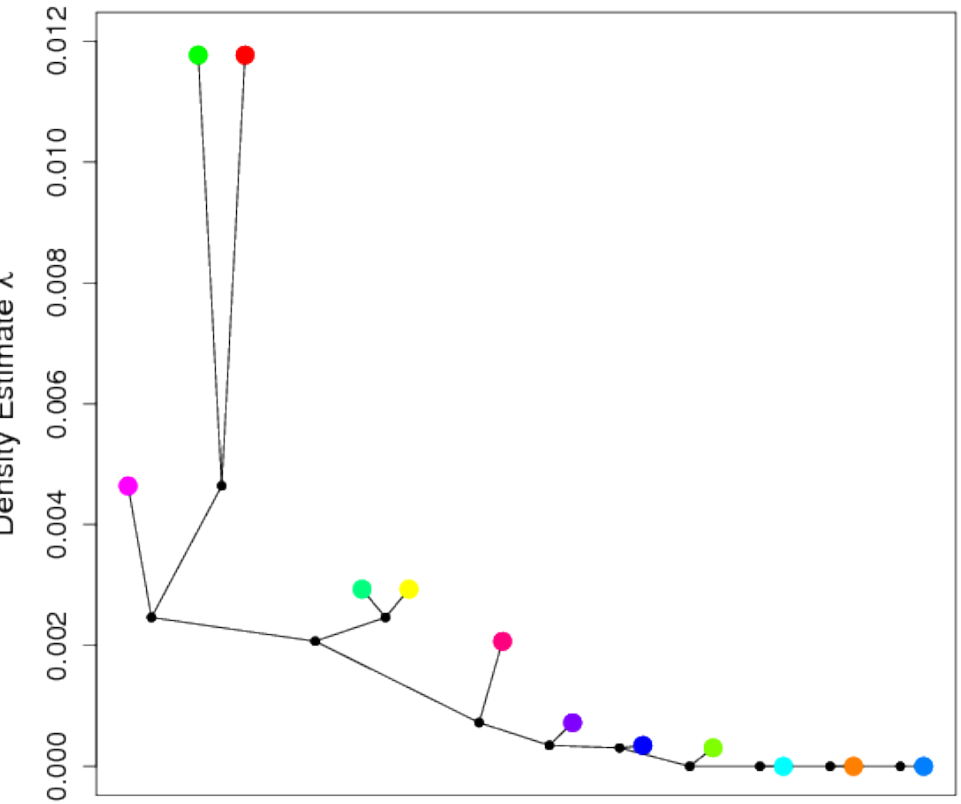
1. Compute KDE $f(x)$ for chosen h
2. for levels $0 < l_1 < l_2 < \ldots < l_r < \ldots < l_R \geq \sup_x f(x)$
   1. find **level set** $L_r = \{x \mid f(x) \geq l_r\}$ of f
   2. if L_r **disconnected** then each connected component is a cluster $\rightarrow (C_{r,1}, C_{r,2}, \ldots C_{r,K_r})$

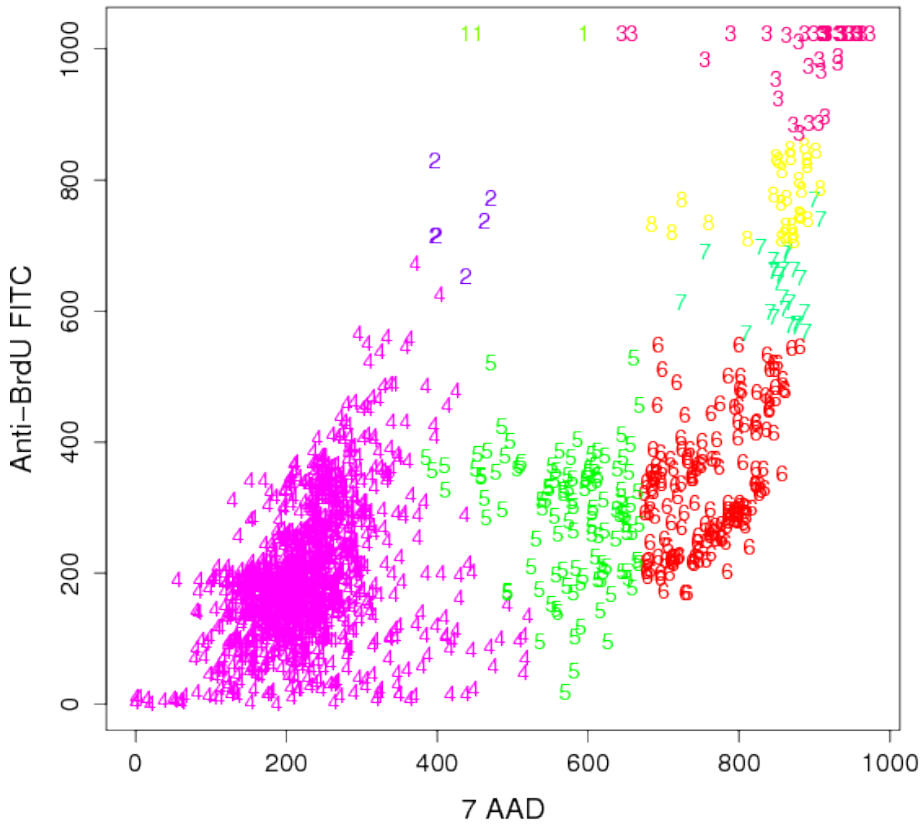
Output clusters $\{(C_{r,1}, C_{r,2}, \ldots C_{r,K_r})\}_{r=1:R}$

Remarks

- every cluster $C_{r,k} \subseteq$ 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 possile by resampling

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





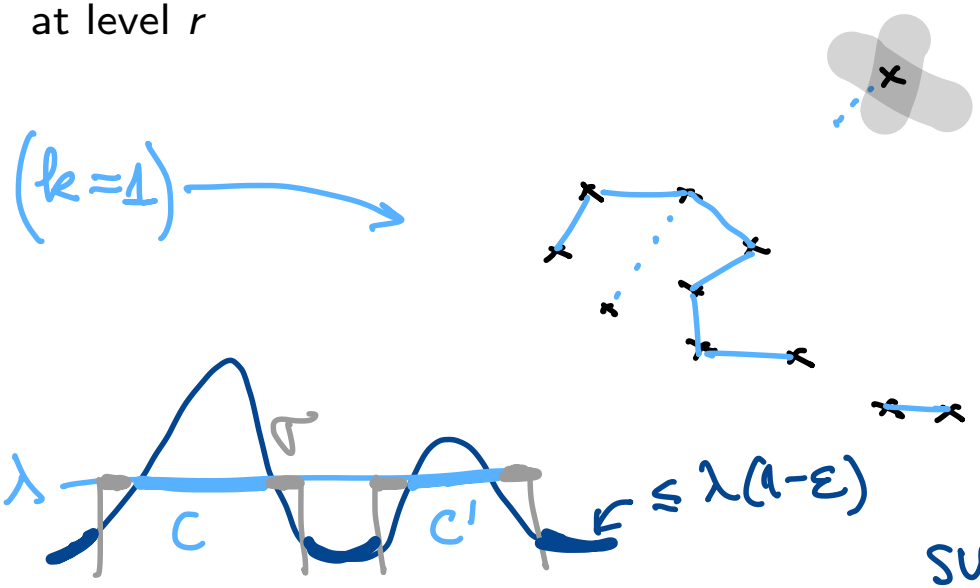
(from **?**)

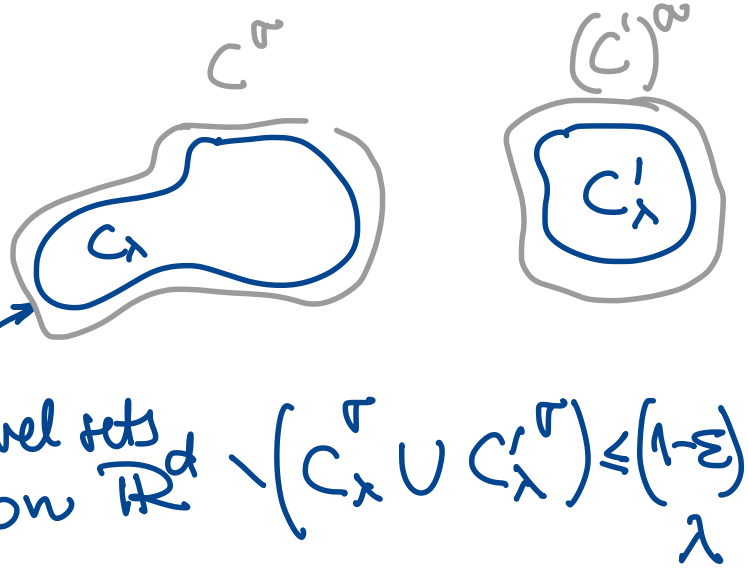
Chaudhuri-Dasgupta Algorithm

- Uses *k*-**nearest neighbor** graphs (filtration)
- Parameters k (nearest neigbhors) 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 $||x_i - x_j|| \leq \alpha r$

d = dimension
$x_{1:n} \in \mathbb{R}^d$
δ = confidence
ε = separation

Consistency Theorem For any ϵ (separation parameter) and δ (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 λ so that $C \cap \mathcal{D}, C' \cap \mathcal{D}$ are clusters at level r





The K-nn density estimator

$\hat{p}(x)$ = k-nn density est.

k = parameter smoothing

The K-nn density estimator

- Let $B_r(x)$ be the (closed) ball of radius r centered at x
- If $|B_r(x^i) \cap \mathcal{D}| = k$ then $\hat{p}(x^i) = \frac{1}{r^n \omega_n} \frac{k}{n}$ is an estimate of the density at x^i
  - $\omega_n = \pi^{n/2}/\Gamma(n/2+1)$ is the volume of the unit ball in $\mathbb{R}^n$
  - intuitively, the ball of radius r contains k/n probability mass
  - Note that the density $\hat{p}$ is not required to integrate to 1

$x_{(k)}$ = k'th nn of x

$\|x - x_{(k)}\| = r_x$

$|B_x(r_x) \cap \mathcal{D}| = k$

$$\hat{p}(x) = \frac{P(B_x(r_x))}{\text{Vol}(B_x(r_x))} = \frac{k/n}{r_x^d \omega_d}$$

"Vol($B_0(1)$)

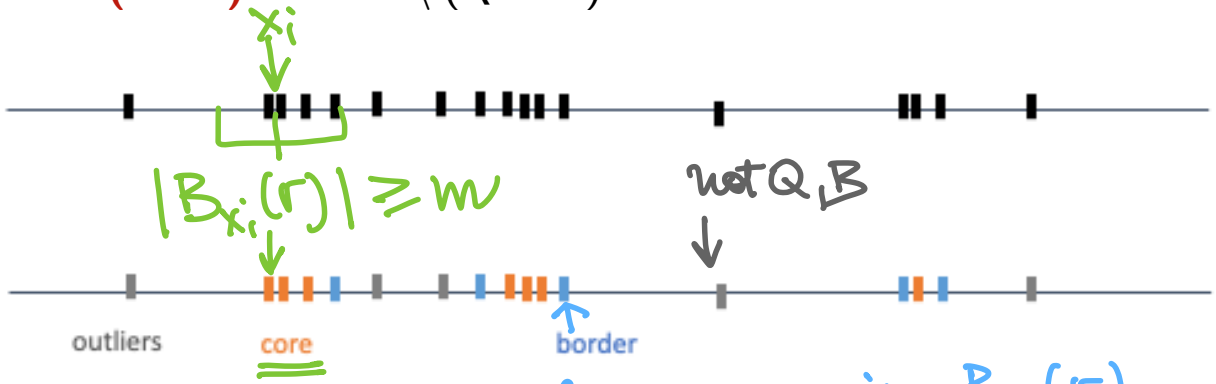
DBScan

Simple → good for large n

- Introduced with no proof, but widely used. Implicitly based on the K-nn estimator
- Parameters r radius, m minimum number points
- Definitions **core** $Q = \{x^i \in \mathcal{D}, \text{ with } B_r(x^i) \cap \mathcal{D}| \geq m\}$
- **border** $B = \{x^i \in \mathcal{D} \setminus Q, \text{ so that } x^i \in B_r(x^j), x^j \in Q\}$
- **outliers (noise)** $O = \mathcal{D} \setminus (Q \cup B)$

m = threshold
smoothing r = radius

$x_i \in Q \Rightarrow p(x_i)$ high



has core $x_{i'}$ in $B_{x_i}(r)$

$Q \rightarrow$ Neighborhood graph

$x_i, x_j \in Q$
$\|x_i - x_j\| < r$

- Algorithm idea
- Construct directed graph $\mathcal{G}$ with edges (i, j) where $x^i \in Q, j \in B_r(x^i)$
- The graph edges between core points are undirected/symmetric, the other are from core to border
1. Clusters are determined by the connected components of the graph restricted to Q.
2. The border points are assigned to a cluster containing x^j so that $x^i \in B_r(x^j)$, $x^j \in Q$ Note that this assignment is not unique!

- Heuristic algorithm estimates r, m

[Chaudhuri-Dasgupta Algorithm]

Consistency Theorem For any ϵ (separation parameter) and δ (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 $C \cap \mathcal{D}$, $C' \cap \mathcal{D}$ are clusters at level r

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

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

- it follows that the needed sample size n at level λ

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

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

- New results **[Kent, B. P., Rinaldo, A. and Verstynen, T. 2013]**

- Remark: algorithm(s) can be applied in any metric space