COMP90014

Algorithms for Bioinformatics

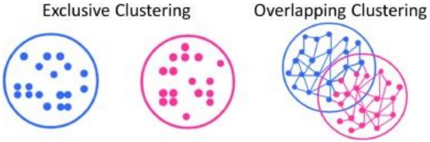
Week 10B: Unsupervised Learning - Clustering II

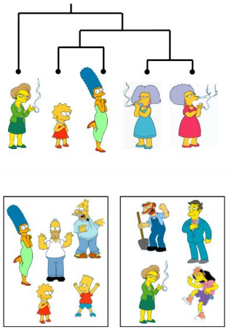
Melbourne Bioinformatics
BIOINFORMATICS + DATA SERVICES + INFRASTRUCTURE, FOR LIFE SCIENCES TODAY

THE UNIVERSITY OF MELBOURNE

Machine learning: clustering

Clustering approaches





Exclusive

- Data points belong to only one cluster

Overlapping

- Data points may belong to many clusters

Hierarchical

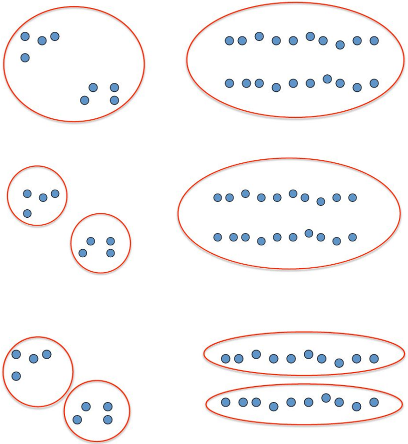
- Assign points to “nested” clusters
- Get all possible clusters for given metric

Partitional

- Split points into “flat” independent clusters
- How many?

Images: Yohoshiva Basaraboyina via medium.com; Eamonn Keogh, Brandeis University

Ingredients for clustering

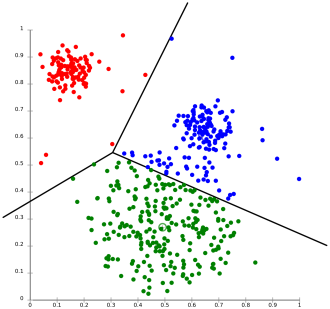


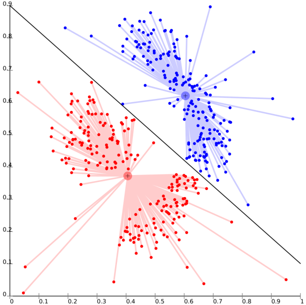
1. similarity metric
   - *e.g.* Euclidean distance
2. a function to evaluate the quality of the clusters
3. clustering algorithm

🤖 clustering is subjective
e.g. how many clusters?
- fixed k clusters
- find the best k to optimize a function

Clustering approaches by cluster definition

1. Centroid-based (*k*-means, *k*-medoids)

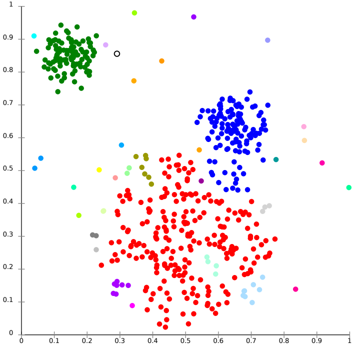


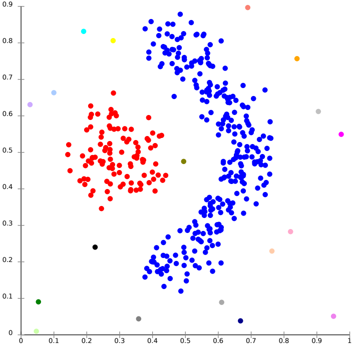


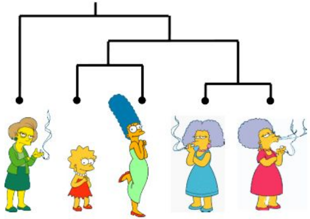
Notion of clusters: Voronoi tessellation/diagram

Chire via Wikimedia Commons (1) (2)

2. Connectivity-based (hierarchical)



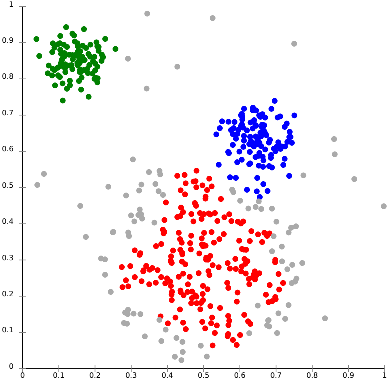


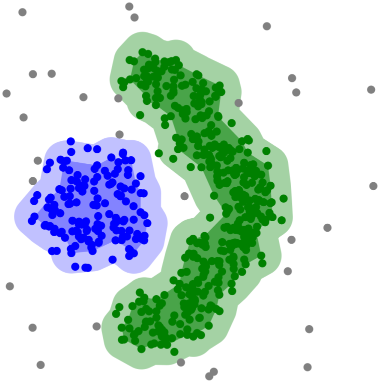


Notion of clusters: cut dendrogram at some depth

Chire via Wikimedia Commons (1) (2)

3. Density-based (DBSCAN, OPTICS)



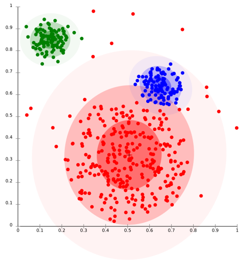


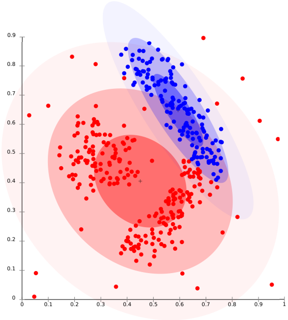
Notion of clusters: connected regions of high density

Chire via Wikimedia Commons (1) (2)

Clustering approaches by cluster definition

4. Distribution-based (Mixture Models)





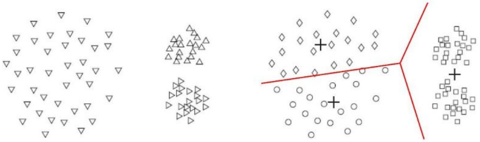
Notion of clusters: distributions on features

5. Network-based

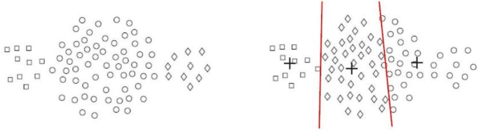
Notion of clusters: graph connectivity

Chire via Wikimedia Commons (1) (2)

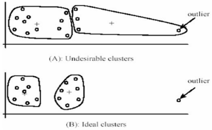
When k-means fails

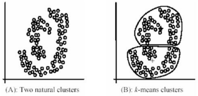


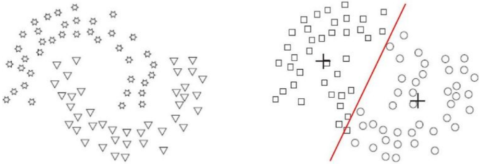
(a) Original points. (b) Three K-means clusters.



(a) Original points. (b) Three K-means clusters.





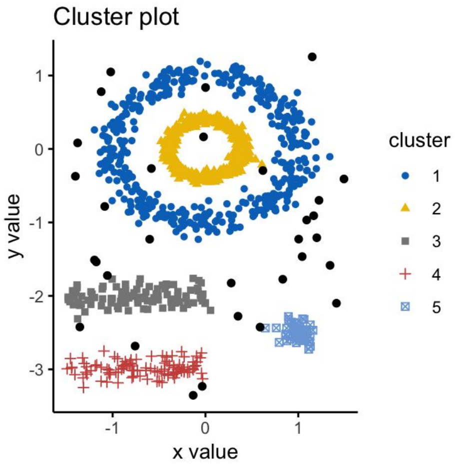


(a) Original points. (b) Two K-means clusters.

Anil Yadav via [slideshare]

Machine learning: clustering

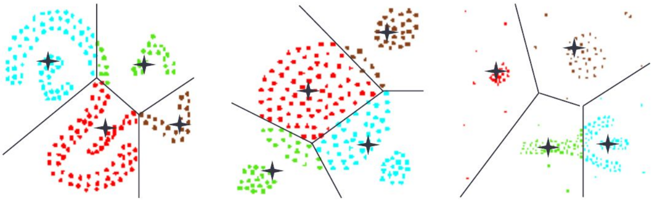
Concepts



- clusters are ***dense regions in the data space***, separated by regions of lower object density
- for any point in a cluster, the local point density around that point has to exceed some threshold (ε)
- the set of points from one cluster is spatially connected

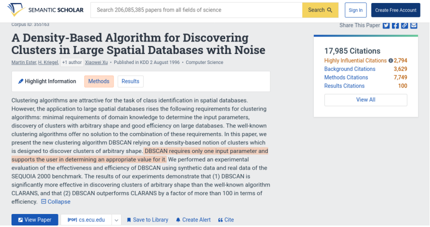
Sagi Shaier via towardsdatascience.com

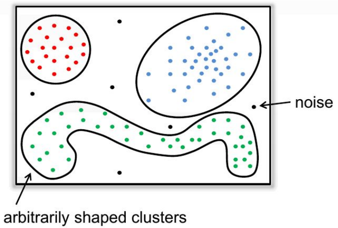
Density-based clustering



- When k-means fails ($K = 4$)
- DBSCAN:
  - discover clusters of arbitrary shape
  - handle noise and outliers

DBSCAN





- first density-based clustering algorithm
- one the most widely used/cited clustering algorithms

Intuition:

- ***a cluster is a region of high density***
- noise points lie in regions of low density

We need to:

- define neighbourhood of a data point
- define high density

Ester *et al.*, 1996. KDD-96 Proceedings

Definitions

ε-neighbourhood: objects within a radius ε of an object.

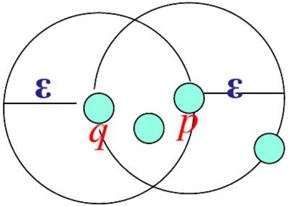
$$N_\varepsilon(p) : \{q \mid d(p,q) \leq \varepsilon\}$$

ε: **input parameter**.

High-density: ε-neighbourhood of an object contains at least minpts of objects.

minpts: **input parameter**.

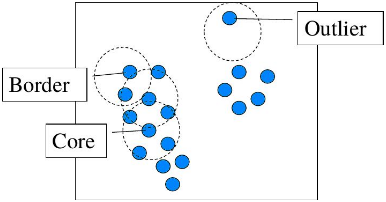
ε-neighbourhood of p and q: Density of p is "high" (minpts $= 4$); Density of q is "low" (minpts $= 4$)



Shah, 2012 10.1109/NUICONE.2012.6493211

Definitions

Given ε and minpts, categorize the objects into three exclusive categories.

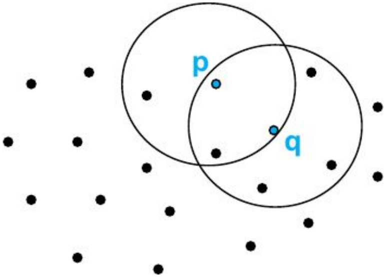


Core point: It has more than minpts objects within ε.

Border point: It has fewer than minpts within ε, but is in the neighbourhood of a core point.

Noise/outlier point: Any point that is not a core point nor a border point.

Direct density reachability



minpts $= 5$

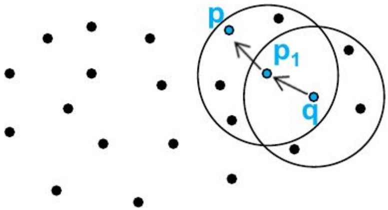
An object p is ***directly density-reachable*** from object q if:

- q is a core object and
- p is in its ε-neighborhood

- is p directly density-reachable from q?
- is q directly density-reachable from p?
- Density-reachability is an asymmetric relationship

Khan *et al.*, 2018. 10.1109/CEEICT.2018.8628138

Density reachability



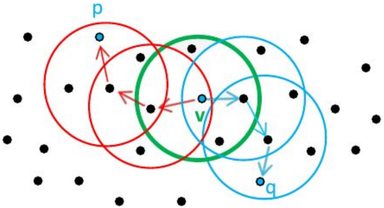
minpts $= 5$

A point p is *density-reachable* from a point q if:

- There is a chain of points $p_1, p_2, \dots, p_k$, with $p_1 = q$ and $p_k = p$, such that p_{i+1} is directly density-reachable for all $1 < i < k - 1$
- Asymmetric

Khan *et al.*, 2018. [10.1109/CEEICT.2018.8628138](https://doi.org/10.1109/CEEICT.2018.8628138)

Density **connectivity**



minpts $= 5$

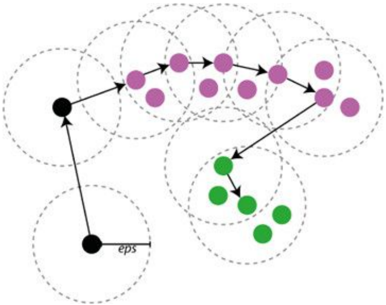
A point *p* is *density-connected* to a point *q* if:

- there is a point *v*, such that both *p* and *q* are *density-reachable* from *v*

- Symmetric

Khan *et al.*, 2018. 10.1109/CEEICT.2018.8628138

Cluster definition



Given a data set D of points, parameter ε and minpts:

- a cluster C is a subset of D satisfying two criteria.

Maximality:

- $\forall p, q$ if $p \in C$
  and if q is density-reachable from p,
  then also $q \in C$

Connectivity:

- $\forall p, q \in C$,
  p and q are density-connected

- clusters will contain both core and border points
- noise/outliers:
  - points in D not belonging to any cluster

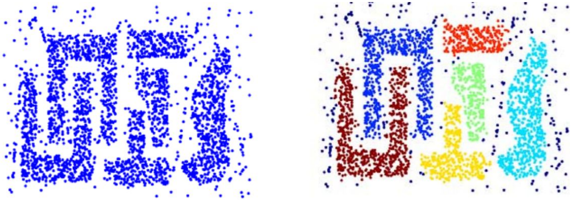
Zielasko *et al.*, 2016. 10.1111/cgf.12881

DBSCAN algorithm

```
Procedure Dbscan(X, ε, minpts):
    foreach unvisited point x ∈ X do
        mark x as visited
        N ← GetNeighbours(x, ε)
        if |N| < minpts then
            mark x as noise
        else
            C ← {x}
            foreach point x′ ∈ N do
                N ← N \ x′
                if x′ is not visited then
                    mark x′ as visited
                    N′ ← GetNeighbours(x′, ε)
                    if |N′| ≥ minpts then
                        N ← N ∪ N′
                if x′ is not yet member of any
                  cluster then
                    C ← C ∪ {x′}
```

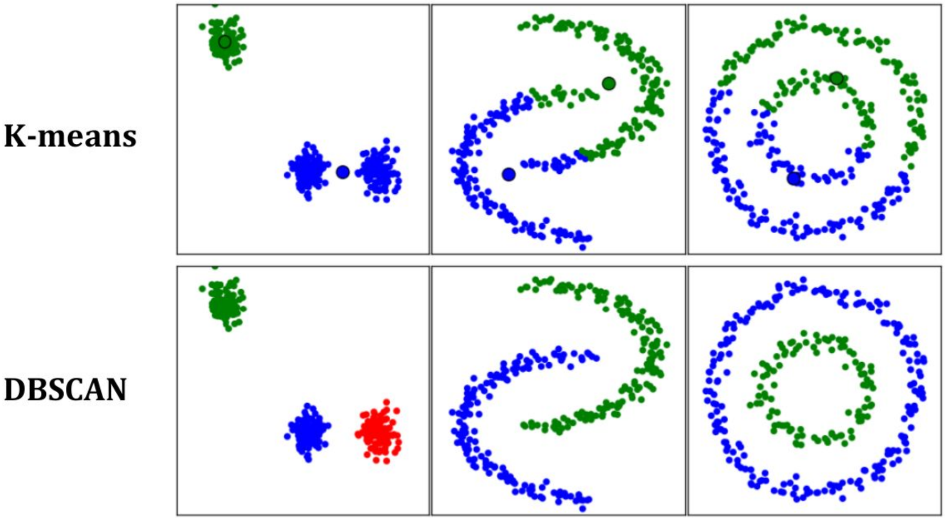
- Input parameters:
  - X points, ε and minpts
- the algorithm proceeds by arbitrarily picking (scanning) up points in the dataset until all points have been visited
- if p is a core point (at least minpts points within a radius of ε) collect all density-reachable points from p and assign to a new cluster
- assign p to noise otherwise
- don't change p's cluster assignment

Complexity and strength of DBSCAN



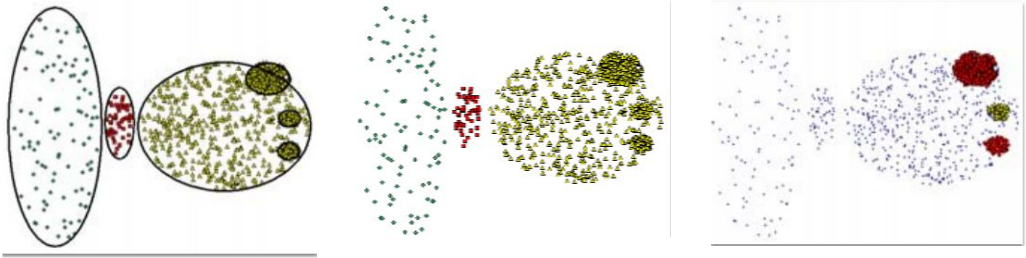
- time complexity:
  - $O(n^2)$ if done naïvely
  - $O(n \times \log n)$ with a spatial index
    - only works in relatively low dimensions
- space complexity: $O(n)$
- can handle arbitrary shapes
- can handle clusters of different sizes
- resistant to noise

DBSCAN *vs.* *k*-means



scikit-learn [documentation]

Weaknesses of DBSCAN



Goal:

- varying densities
- high dimensional data
- overlapping clusters

Different ε configurations:

- setting ε and minpts can be tricky

Determining ε

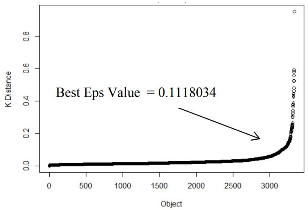
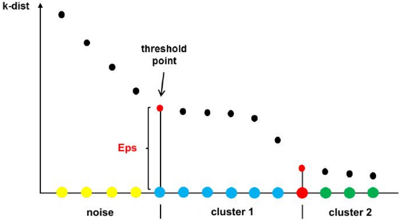


Figure 2 Points sorted by distance to the 3rd nearest neighbor



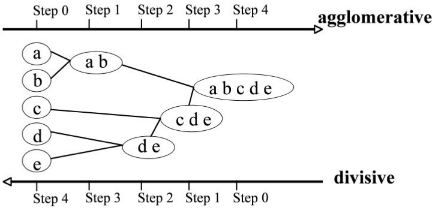
k-distance: calculate distance of k^{th} nearest neighbor for each point
$k = \text{minpts} - 1$

- plot in ascending / descending order
- set ε to the maximum distance before the threshold
- noise points have their k^{th} nearest neighbour at higher distances

Machine learning: clustering

1. Clustering approaches
2. Density-based clustering
3. Divisive clustering

Hierarchical clustering



What are the next clusters to merge?

$$\begin{aligned}\text{Single linkage:}\quad & D_{k,g} = \min(D_{k,i}, D_{k,j})\\ \text{Complete linkage:}\quad & D_{k,g} = \max(D_{k,i}, D_{k,j})\\ \text{Average linkage:}\quad & D_{k,g} = \frac{D_{k,i} + D_{k,j}}{2}\end{aligned}$$

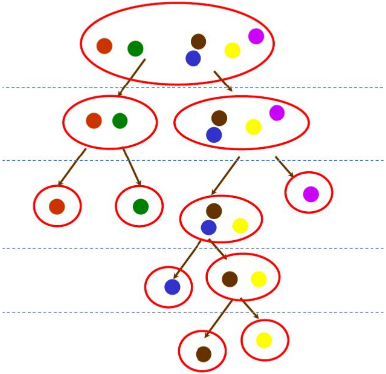
Agglomerative clustering (bottom-up)

- each data point starts as a single cluster
- join clusters into bigger clusters till we reach one single cluster with all points

Divisive clustering (top-down)

- start with one big cluster
- at each step, split into smaller clusters
- stop at desired number of clusters
- *e.g.* when points are in single clusters

Divisive hierarchical clustering



- Any partitional algorithm that generates a fixed number of clusters can be used to implement divisive hierarchical clustering
  - *e.g.* k-means, with $k = 2$
  - keep partitioning clusters iteratively

Challenge: use k-means to implement divisive hierarchical clustering on a set of points X.

You can use assume the function `kmeans()` is available (you don't need to implement it yourself).

Hint: start by dividing X into two clusters, then recursively run `kmeans()` on the output until each cluster has only 1 item.

Thank you!

Today: Unsupervised Learning - Clustering II

Next time: Supervised Learning