Clustering Algorithms
CS-477 Computer Vision

Dr. Mohsin Kamal
Associate Professor
dr.mohsinkamal@seecs.edu.pk

School of Electrical Engineering and Computer Science (SEECS)
National University of Sciences and Technology (NUST), Pakistan

Where to go from our basic building block?

Feature Points

Recognition/Classification

Reconstruction

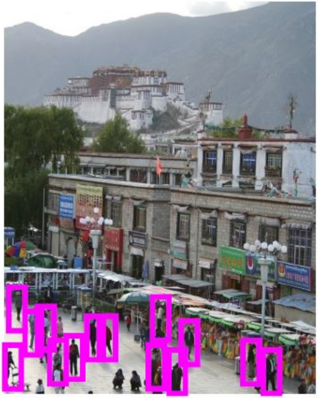
Recognition/Classification

Is this a dog?

Recognition/Classification

Often needs machine learning for compact descriptions of the visual world.

Scene recognition
- Piazza/forest/factory/...



Find pedestrians

Detection: (Recognition + localization)

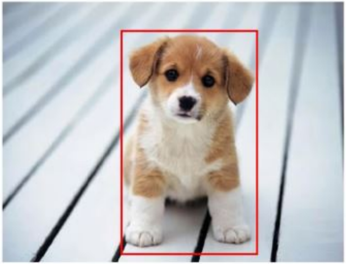
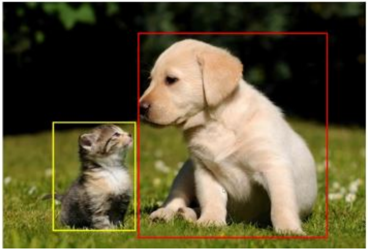
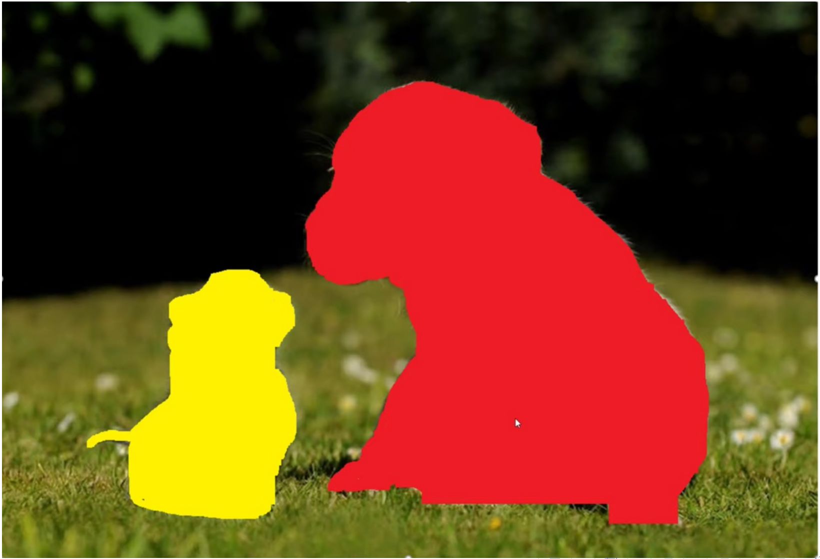


Image classificati
on with
localization



Segmentation: (Semantic Recognition + localization)

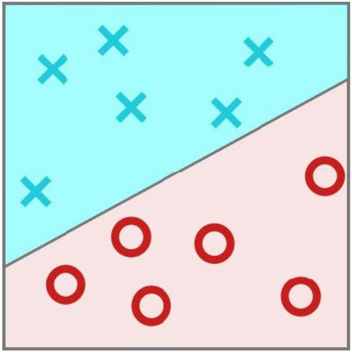


Classification vs. detection vs. segmentation

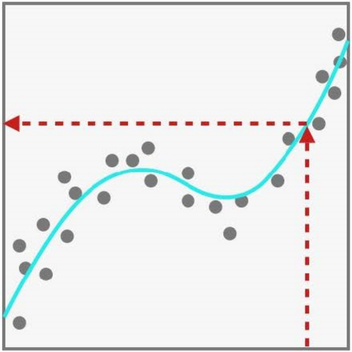
Machine Learning Problems

Supervised Learning	*Unsupervised Learning*
classification or categorization	clustering
regression	dimensionality reduction

Regression vs. Classification



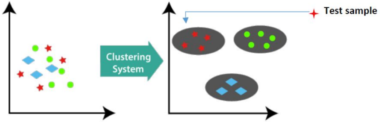
Classification



Regression

What is clustering

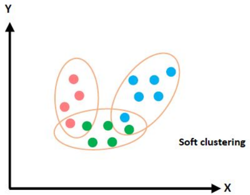
- **Clustering:** the process of combining or grouping similar objects to treat them in the same way
- Objects show similarity to one another in the same cluster
- Dissimilarity to the objects in other clusters

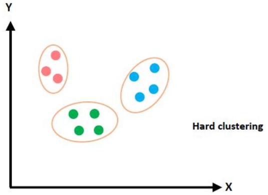


- A form of unsupervised learning - you generally don't have examples demonstrating how the data should be grouped together
- Its unsupervised as it learns from the similar features
- So, it's a method of data exploration - a way of looking for patterns or structure in the data that are of interest

Hard vs. soft clustering

- **Hard clustering:** Each sample/object belongs to exactly one cluster
  - More common and easier to do
- **Soft clustering:** when some samples/objects belong to more than one cluster.
  - As a practical example, we may want to put a pair of sneakers in two clusters: (i) sports apparel and (ii) shoes
  - It can only be done with a soft clustering approach.

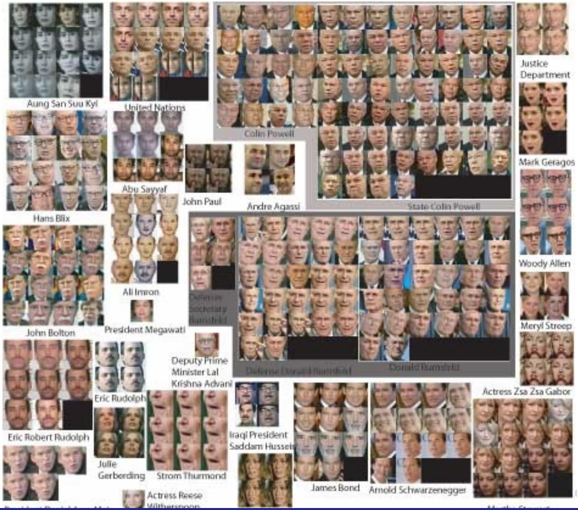




Applications of clustering

Goal: Break up the image into meaningful or perceptually similar regions

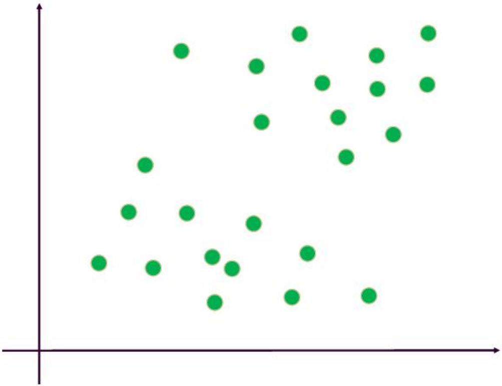
Face clustering

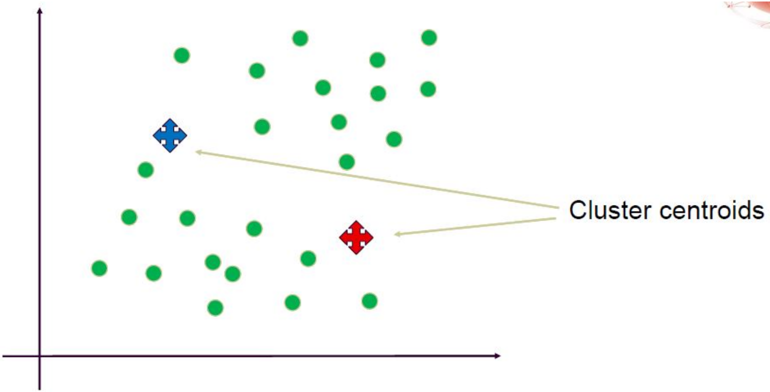


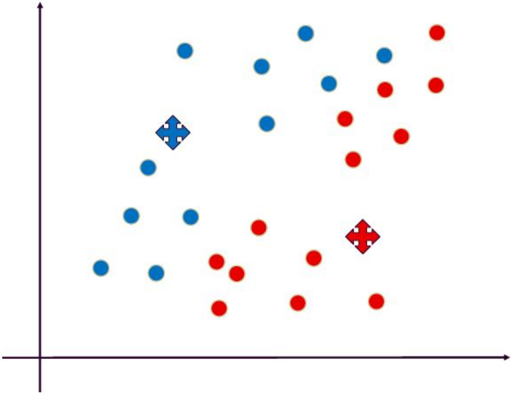
Applications of clustering

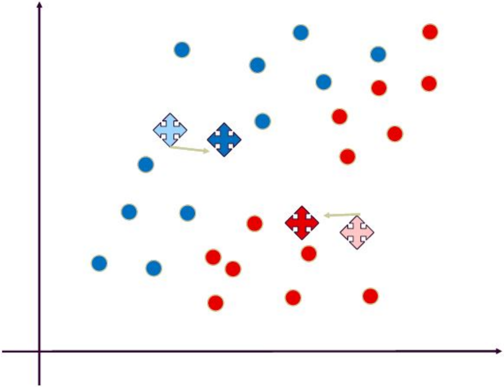
- Intermediate step for other data mining tasks
  - Such as generating a compact summary of data for classification, pattern discovery etc.
  - Outlier detection
- Biology and Bioinformatics.
  - Categorize genes with similar functionality.
- Collaborative filtering, recommendation systems
  - Finding like-minded users or similar products

- Most well-known and popular clustering algorithm:
- Start with some initial cluster centers
- Iterate:
  - Assign/cluster each example to closest center
  - Recalculate centers as the mean of the points in a cluster

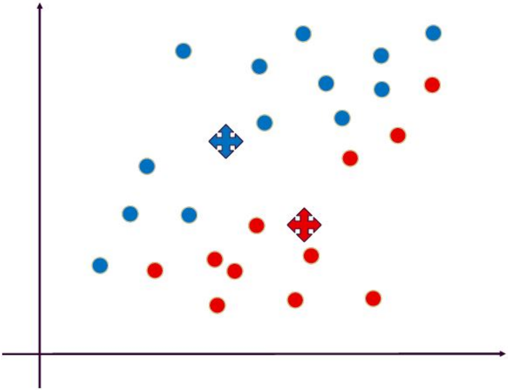


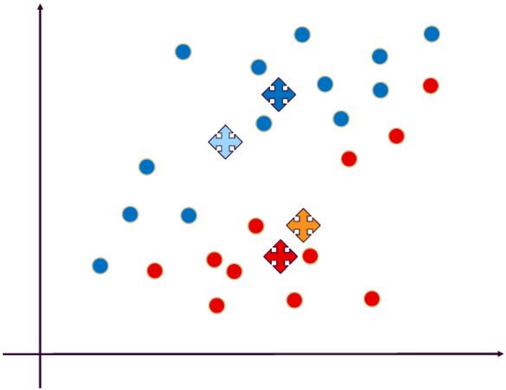

Cluster centroids

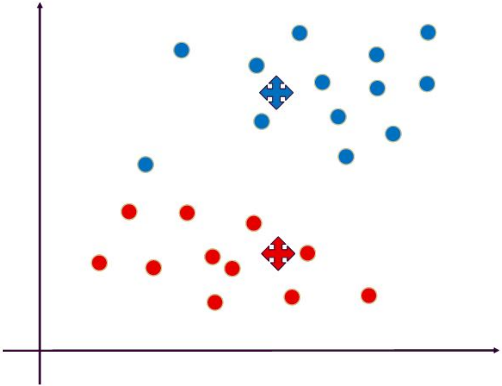


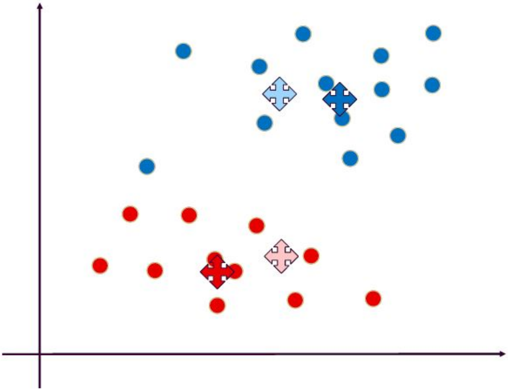


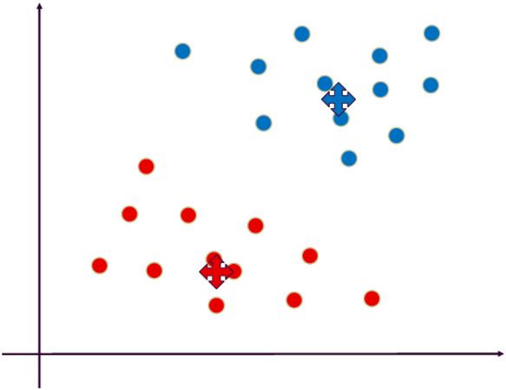
New cluster centroids

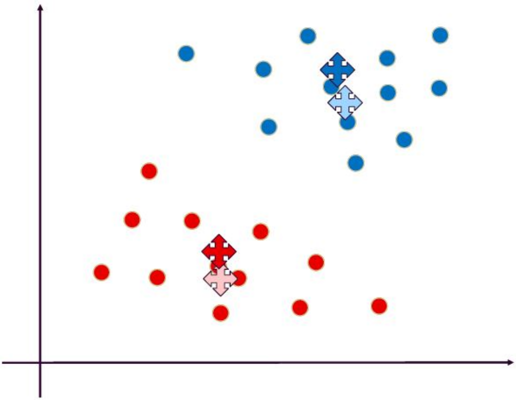


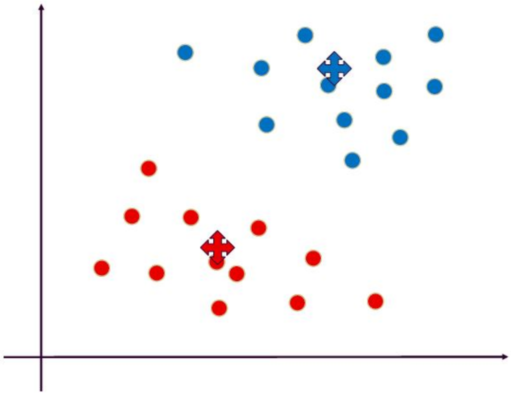












INPUT:

- K – number of clusters ← hyperparameter
- Initial locations of cluster centroids $\{\mu_1^0, \mu_2^0, \mu_3^0, \ldots \mu_K^0\}$ $\mu_i^0 \in \mathbb{R}^n$
- Training set: $\{x_1, x_2, x_3, \ldots x_N\}$ $x_i \in \mathbb{R}^n$

OUTPUT:

- Final locations of cluster centroids $\{\mu_1^*, \mu_2^*, \mu_3^*, \ldots \mu_K^*\}$ $\mu_i^* \in \mathbb{R}^n$

Randomly initialize K cluster centroids $\mu_1^0, \mu_2^0, \mu_3^0, \ldots \mu_K^0 \in \mathbb{R}^n$

Repeat{

for i = 1 to N

$c^{(i)} \in \{1, 2, \ldots K\} :=$ index of cluster centroid closest to x_i

for k = 1 to K

$\mu_k :=$ average (mean) of points assigned to cluster k

}

Optimization objective

$c^{(i)} \in \{1, 2, \ldots K\} :=$ index of cluster to which x_i is currently assigned

$\mu_k :=$ cluster centroid k

$\mu_{c^{(i)}} :=$ centroid of cluster to which example x_i has been assigned

OPTIMIZATION OBJECTIVE:

$$J(c^{(1)}, c^{(2)}, \ldots, c^{(N)}, \mu_1, \mu_2, \ldots, \mu_K) = \frac{1}{N} \sum_{i=1}^{N} \left\| x_i - \mu_{c^{(i)}} \right\|^2$$

$$\min_{\substack{c^{(1)}, \ldots, c^{(N)} \\ \mu_1, \ldots, \mu_K}} J(c^{(1)}, c^{(2)}, \ldots, c^{(N)}, \mu_1, \mu_2, \ldots, \mu_K)$$

distortion

Randomly initialize K cluster centroids $\mu_1^0, \mu_2^0, \mu_3^0, \ldots \mu_K^0 \in \mathbb{R}^n$

Repeat{

for i = 1 to N

$c^{(i)} \in \{1, 2, \ldots K\} :=$ index of cluster centroid closest to x_i

for k = 1 to K

$\mu_k :=$ average (mean) of points assigned to cluster k

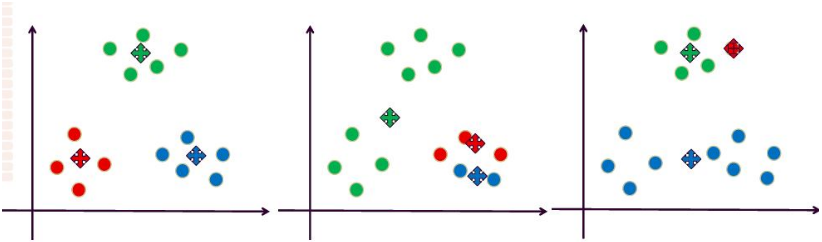
}

$$J(c^{(1)}, c^{(2)}, \ldots, c^{(N)}, \mu_1, \mu_2, \ldots, \mu_K)$$

Algorithm initialization

- Randomly pick K training points and set $\mu_1^0, \mu_2^0, \mu_3^0, \cdots, \mu_K^0$ equal to these points
- The initialization of the cluster centroids sometimes affects the final result (two runs of the K-means Algorithm may result in two different models)
- Some variants of the K-means Algorithm compute the initial positions of the centroids based on some properties of the dataset

Local minima



$$J(c^{(1)}, c^{(2)}, \ldots, c^{(N)}, \mu_1, \mu_2, \ldots, \mu_K) = \frac{1}{N} \sum_{i=1}^{N} \left\| x_i - \mu_{c^{(i)}} \right\|^2$$

Random initialization

For j = 1 to 100 {

- Randomly initialize K-means
- Run K-means. Obtain: $c^{(1)}, \ldots, c^{(N)}, \mu_1, \ldots, \mu_K$
- Computer cost function (distortion)

$$J(c^{(1)}, c^{(2)}, \ldots, c^{(N)}, \mu_1, \mu_2, \ldots, \mu_K)$$

}

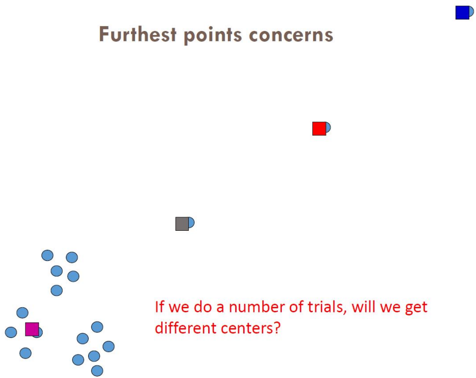
Pick clustering solution that gave the lowest distortion

$$J(c^{(1)}, c^{(2)}, \ldots, c^{(N)}, \mu_1, \mu_2, \ldots, \mu_K)$$

choosing the value of K

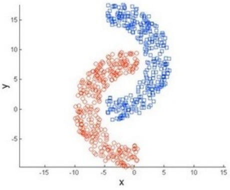
- Choosing the right number of clusters K is a difficult task. There are several methods, but none is optimal.
- One approach is to break up the dataset into training and test data and to use the concept of prediction strength to determine a suitable value of K.
- The cost function (distortion) can be used to find the relationship between the number of clusters and the distortion. However, as the number of clusters increases, the distortion may decrease, but there are trade-offs.

Limitations with K-Means

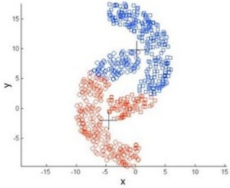


If we do a number of trials, will we get different centers?

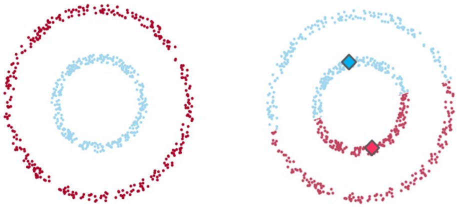
Limitations with K-Means



Original Points

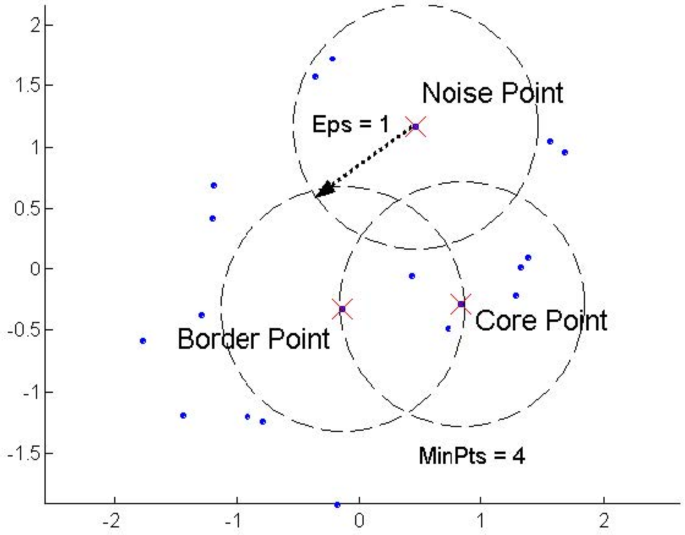


K-means (2 Clusters)



- DBSCAN (density-based spatial clustering of applications with noise).
- It is density based, as opposed to centroid-based (K-means).
- The advantage of DBSCAN is that it build clusters of arbitrary shape, while centroid-based algorithms create clusters that have the shape of a hypersphere.
- However, it has two hyperparameters to be selected.
  1. Radius (Eps)
  2. Minimum number of points (MinPts)
- **Dense Region:** A circle of radius Eps that contains at least MinPts points

- Characterization of points
  - A point is a core point if it has more than a specified number of points (MinPts) within Eps
    - These points belong in a dense region and are at the interior of a cluster
  - A border point has fewer than MinPts within Eps, but is in the neighborhood of a core point.
  - A noise point is any point that is not a core point or a border point.


Noise Point
Eps = 1
Border Point
Core Point
MinPts = 4
2
1.5
1
0.5
0
-0.5
-1
-1.5
-2
-1
0
1
2

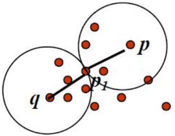
Density-Connected points

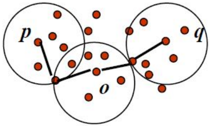
Density edge:

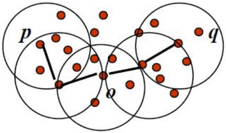
- We place an edge between two core points q and p if they are within distance Eps.

Density-connected:

- A point p is density-connected to a point q if there is a path of edges from p to q





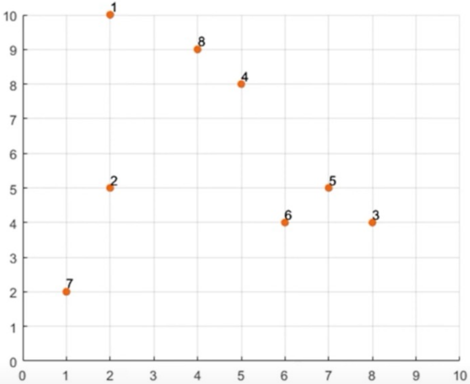


DBSCAN Algorithm

- Label points as core, border and noise
- Eliminate noise points
- For every core point p that has not been assigned to a cluster
  - Create a new cluster with the point p and all the points that are density-connected to p.
- Assign border points to the cluster of the closest core point.

DBSCAN Algorithm

Point	X	Y
P1	2	10
P2	2	5
P3	8	4
P4	5	8
P5	7	5
P6	6	4
P7	1	2
P8	4	9



Eps=2 MinPts=3

DBSCAN Algorithm

P1	0.00							
P2	5.00	0.00						
P3	8.49	6.08	0.00					
P4	3.61	4.24	5.00	0.00				
P5	7.07	5.00	1.41	3.61	0.00			
P6	7.21	4.12	2.00	4.12	1.41	0.00		
P7	8.06	3.16	7.28	7.21	6.71	5.39	0.00	
P8	2.24	4.47	6.40	1.41	5.00	5.39	7.62	0.00
	P1	**P2**	**P3**	**P4**	**P5**	**P6**	**P7**	**P8**

Eps=2

MinPts=3

P1:	P2:	P3:P5,P6	P4: P8
P5:P3,P6	P6:P3, P5	P7:	P8:P4

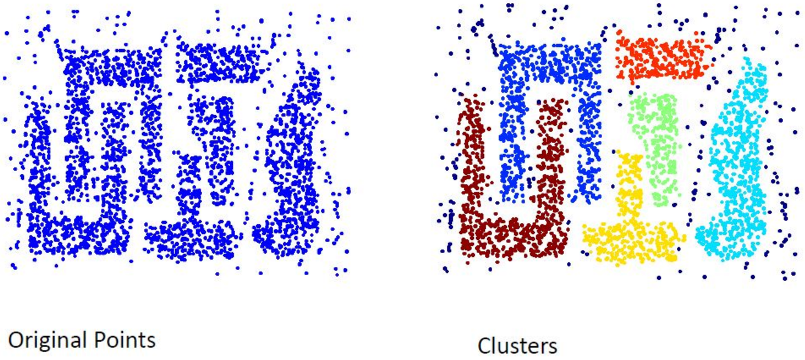
DBSCAN Algorithm

Point	Status	
P1	Noise	
P2	Noise	
P3	Core	
P4	Noise	
P5	Core	
P6	Core	
P7	Noise	
P8	Noise	

Good read: https://towardsdatascience.com/dbscan-make-density-based-clusters-by-hand-2689dc335120

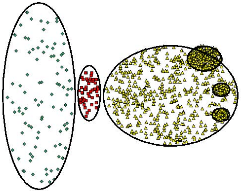
DBSCAN Algorithm

When DBSCAN works well



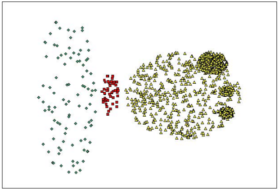
- Resistant to Noise
- Can handle clusters of different shapes and sizes

When DBSCAN does NOT work well

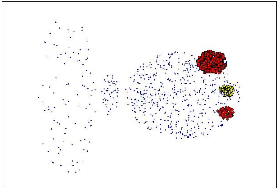


Original Points

- Varying densities
- High-dimensional data



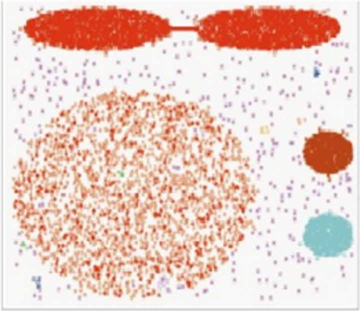
(MinPts = 4, Eps = 0.75)



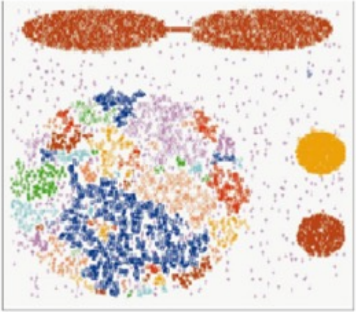
(MinPts = 4, Eps = 9.92)

Sensitive to parameters

MinPts = 4, Eps = 0.5



MinPts = 4, Eps = 0.4



- Produces a set of nested clusters organized as a hierarchical tree
- Can be visualized as a dendrogram
  - A tree-like diagram that records the sequences of merges or splits
- No assumptions on the number of clusters
  - Any desired number of clusters can be obtained by 'cutting' the dendrogram at the proper level

Basic algorithm

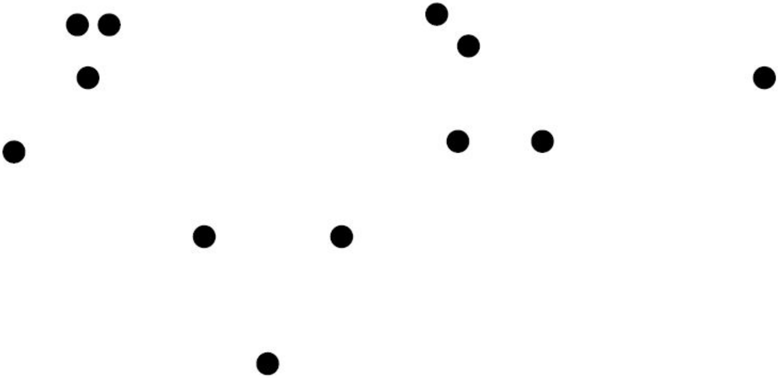
- Initially consider each data point be a cluster
- Compute the distance matrix between the input data points

Repeat

- Merge the two closest clusters
- Update the distance matrix
- **Until** required criterion is satisfied

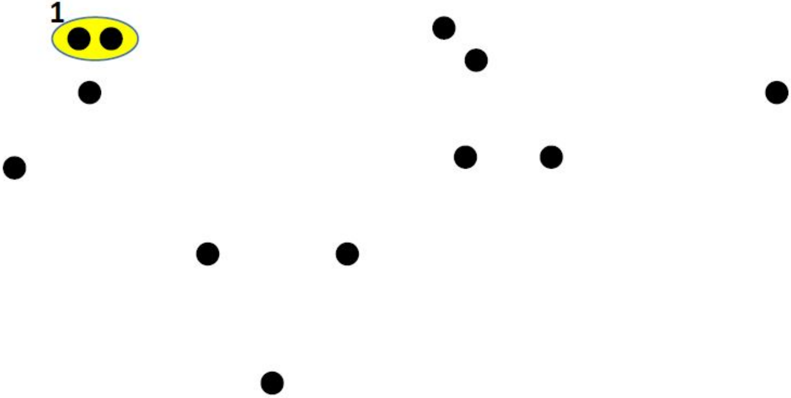
Approach

Agglomerative (Bottom up)



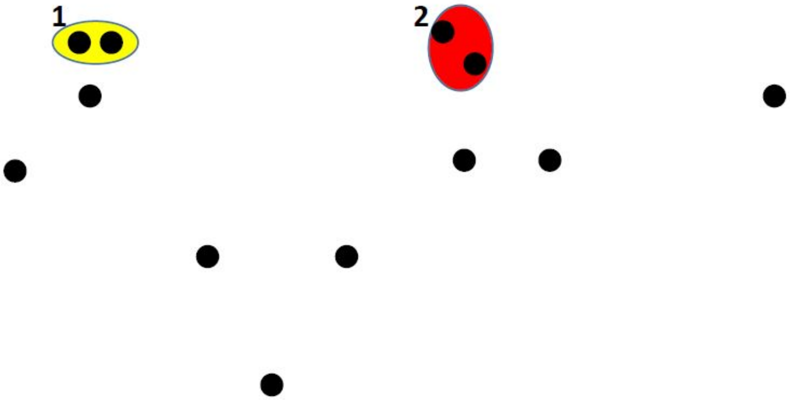
Approach

Agglomerative (Bottom up)
1st iteration



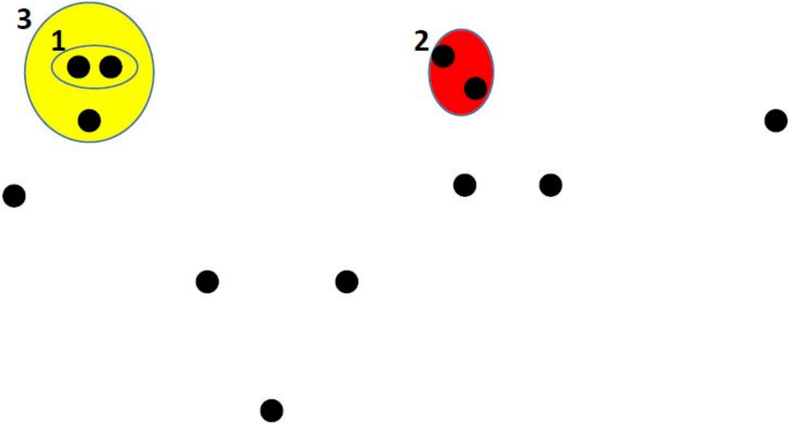
Approach

Agglomerative (Bottom up)
2nd iteration



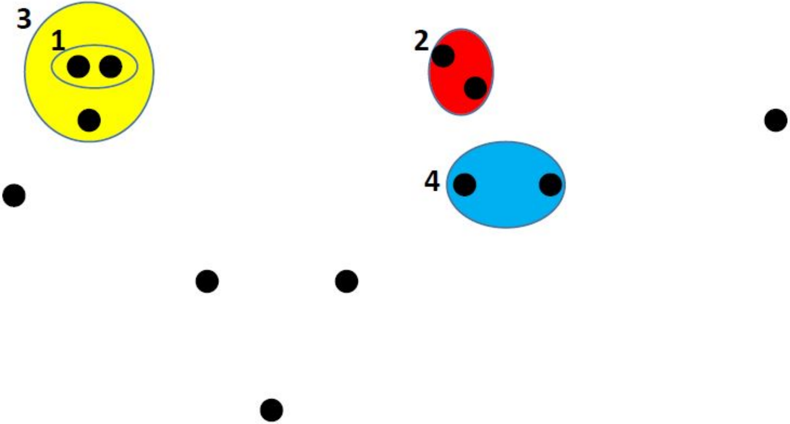
Approach

Agglomerative (Bottom up)
3rd iteration



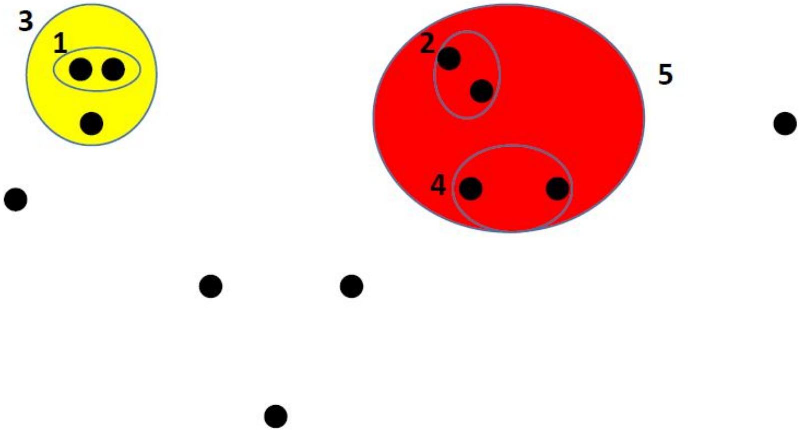
Approach

Agglomerative (Bottom up)
4th iteration



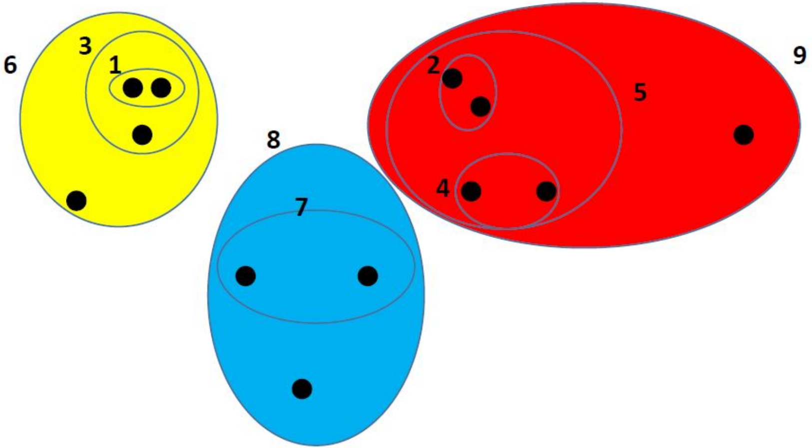
Approach

Agglomerative (Bottom up)
5th iteration

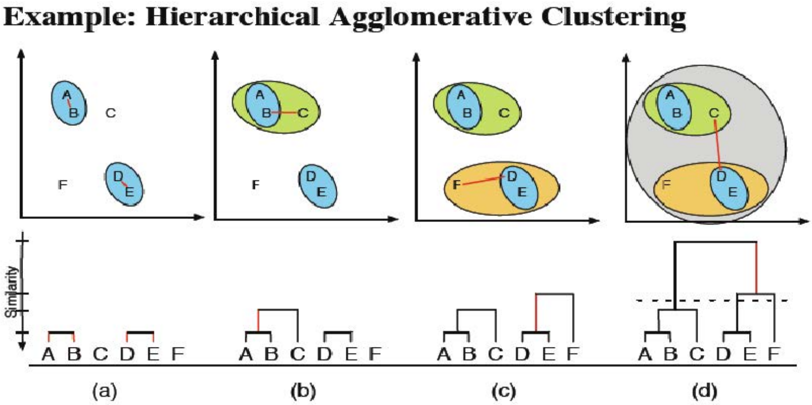


Approach

Agglomerative (Bottom up)
Finally, k clusters left



Approach


Example: Hierarchical Agglomerative Clustering
A
B
C
D
E
F
Similarity
A B C D E F
(a)
(b)
(c)
(d)

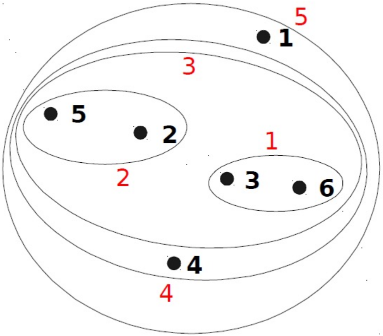
Single-link clustering

- Single-link distance between clusters C_i and C_j is the minimum distance between any object in C_i and any object in C_j
- The distance is defined by the two most similar objects

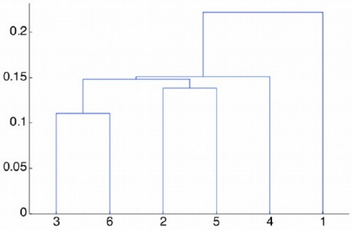
$$D_{single} = \min_{x,y}\{d(x,y) | x \in C_i, y \in C_j\}$$

Single-link clustering

Determined by one pair of points, i.e., by one link in the proximity graph.



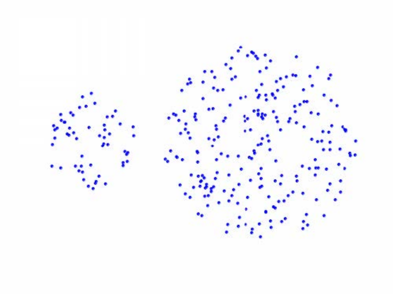
Nested Clusters



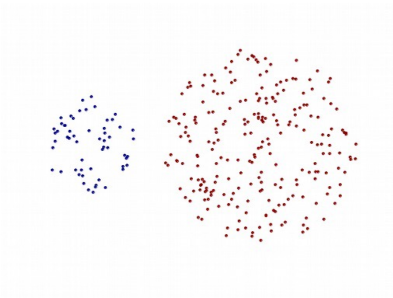
Dendrogram

Strengths of single-link clustering

- It can handle non-elliptical shapes



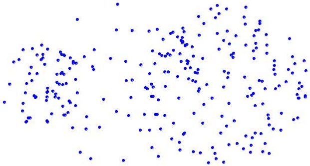
Original points



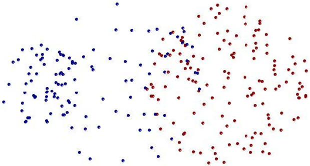
Two clusters

Limitations of single-link clustering

- Sensitive to noise and outliers
- It produces long, elongated clusters



Original points



Two clusters

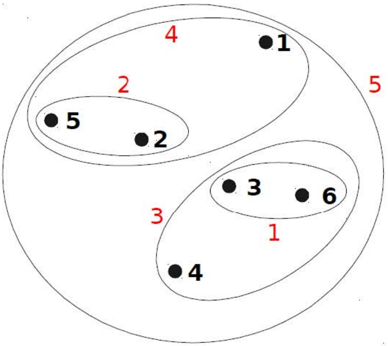
Complete-link clustering

- Complete-link distance between clusters C_i and C_j is the maximum distance between any object in C_i and any object in C_j
- The distance is defined by the two most dissimilar objects

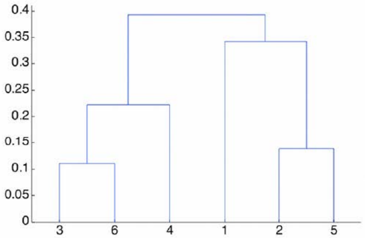
$$D_{complete} = \max_{x,y}\{d(x, y)|x \in C_i, y \in C_j\}$$

Complete-link clustering

Distance between clusters is determined by the two most distant points in the different clusters



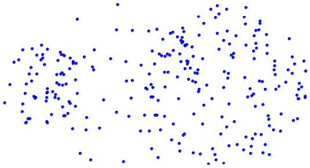
Nested Clusters



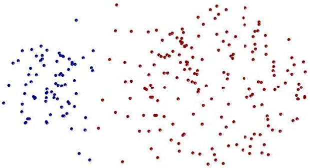
Dendrogram

Strengths of Complete-link clustering

- More balanced clusters (with equal diameter)
- Less susceptible to noise



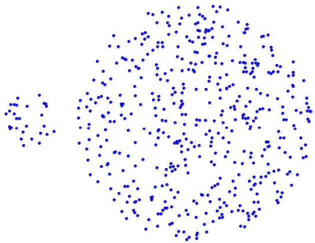
Original points



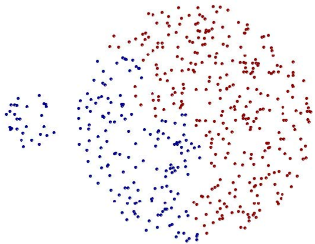
Two clusters

Limitations of Complete-link clustering

- Tends to break large clusters
- All clusters tend to have the same diameter – small clusters are merged with larger ones



Original points



Two clusters