APPLIED MACHINE LEARNING

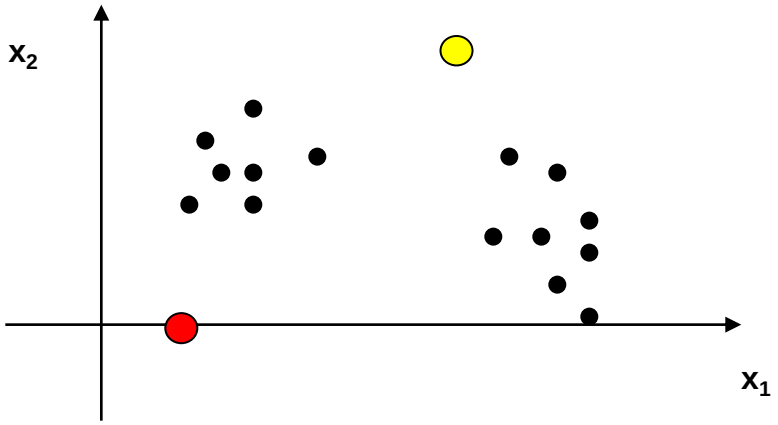
Clustering

Part 2 –Techniques for Clustering

K-Means, Soft K-means, DBSCAN

K-Means

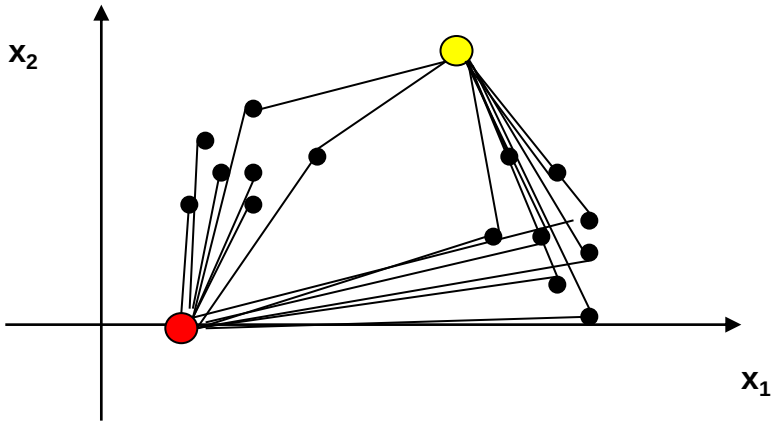
K-means Clustering



Initialization: initialize at random the positions of the centers of the clusters

Here we start with a number of cluster fixed: K=2

K-means Clustering



$$k_i = \arg\min_k \left\{ d\left(x^i, \mu^k \right) \right\}$$

Responsibility of cluster k for point x^i

$$r_i^k = \begin{cases} 1 & \text{if } k_i = k \\ 0 & \text{otherwise} \end{cases}$$

x^i i^{th} data point

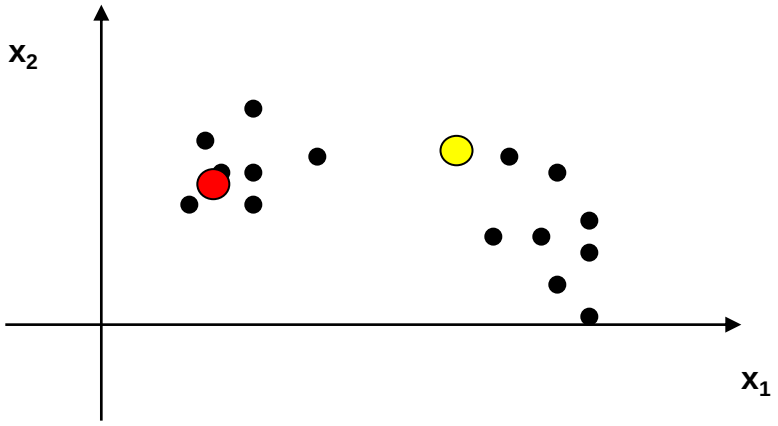
μ^k geometric centroid

Assignment Step:

- Calculate the distance from each data point to each center.
- Assign the datapoint to the “closest” center.

If a tie happens (i.e. two center are equidistant to a data point), one assigns the data point to the cluster with smallest k).

K-means Clustering



$$k_i = \arg\min_k \left\{ d\left(x^i, \mu^k \right) \right\}$$

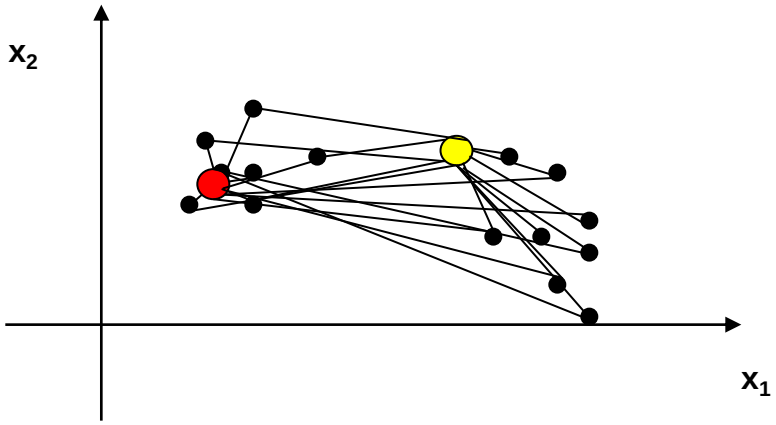
Responsibility of cluster k for point x^i

$$r_i^k = \begin{cases} 1 & \text{if } k_i = k \\ 0 & \text{otherwise} \end{cases}$$

$$\mu^k = \frac{\sum_i r_i^k x^i}{\sum_i r_i^k}$$

Update step (M-Step):
Recompute the position of the center based on the assignment of the points.
The center becomes the centroid of the dataset points assigned to this cluster.

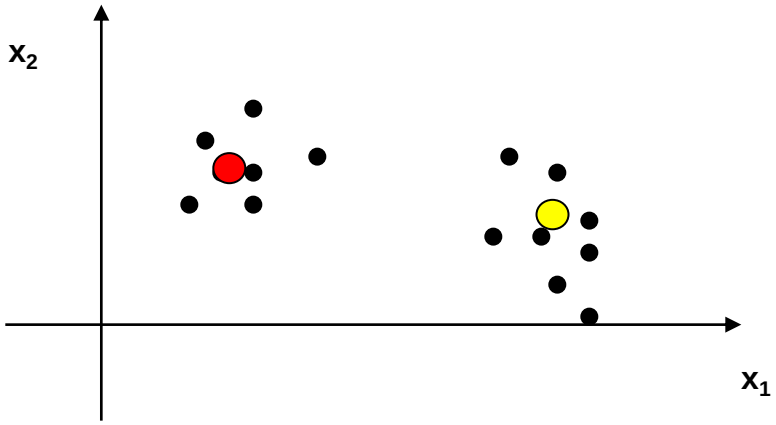
K-means Clustering



Assignment Step:

- Calculate the distance from each data point to each centroid.
- Assign each data point to the “closest” centroid.

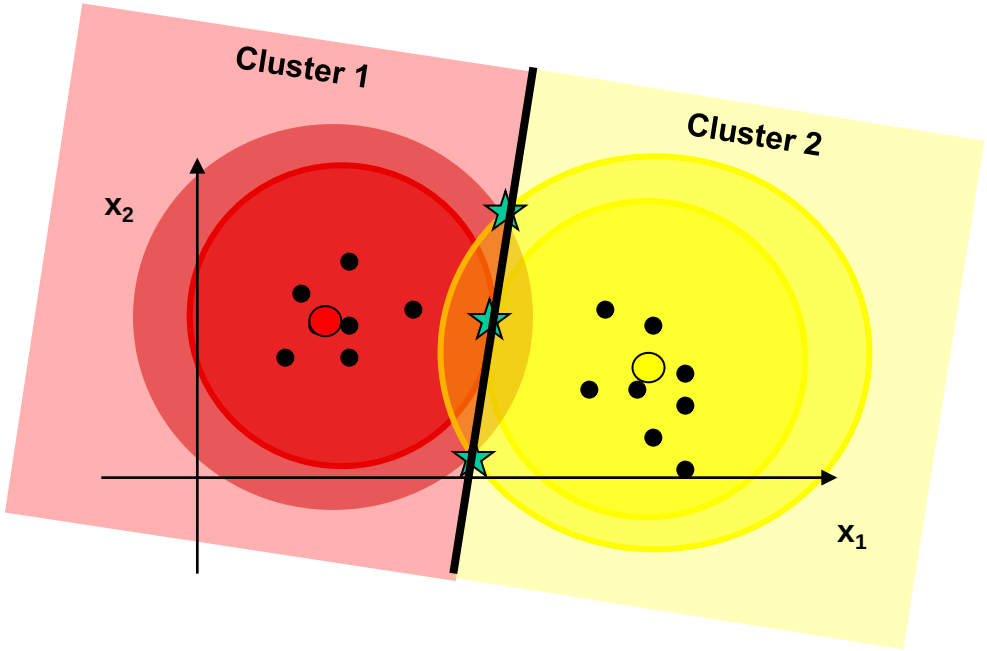
K-means Clustering



Update step (M-Step):
Recompute the position of centroid based on the assignment of the points

Stopping Criterion: Go back to step 2 and repeat the process until the clusters are stable, i.e. the centroids no longer move.

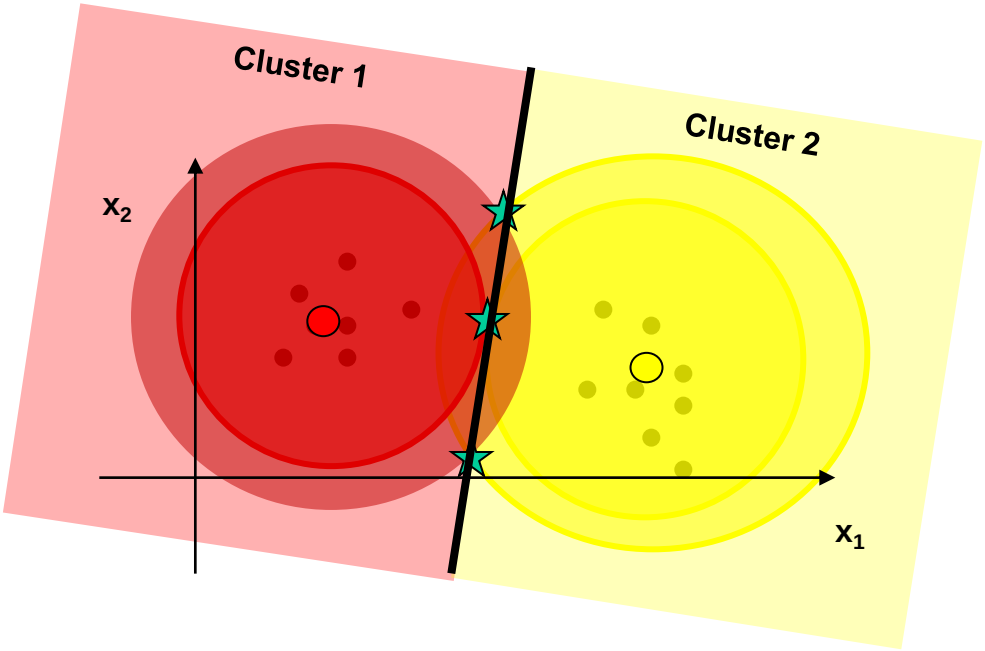
K-means Clustering



Intersection points

K-means creates a hard partitioning of the dataset

K-means Clustering



Intersection points

K-Means clustering minimizes a loss, often a quadratic cost function

$$J\left(\mu^1,...,\mu^K\right)=\sum_{k=1}^{K}\sum_{x^i\in c_k} d\left(x^i,\mu^k\right) \text{ with } d\left(x^i,\mu^k\right)=\sqrt{\sum_{i=1}^{N}\left(x_i^i-\mu_i\right)^2}$$

K-means Clustering

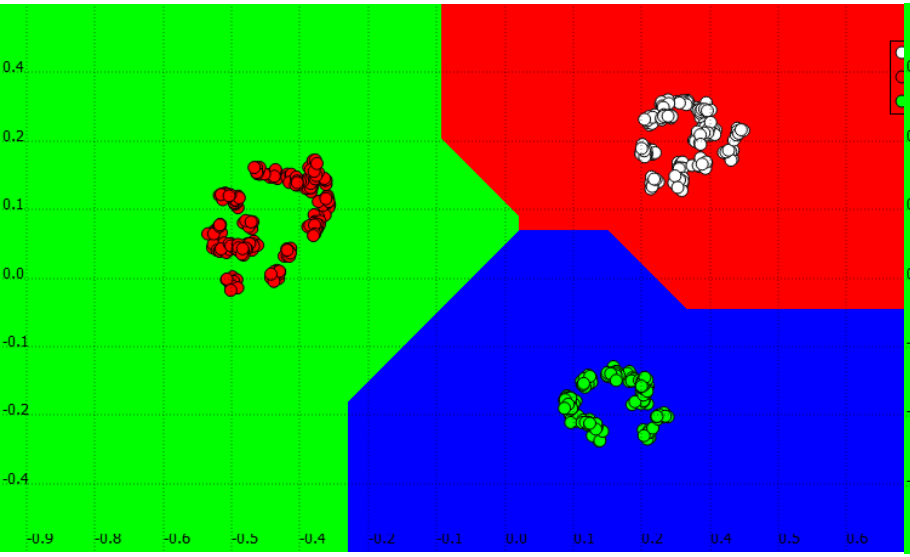
- $p = 1$ Manhattan Distance
- $p = 2$ Euclidean Distance
- $p = \infty$ L-infinity norm $\|x\|_\infty = \max_i |x_i|$

The distance can also be replaced by the L-p norm

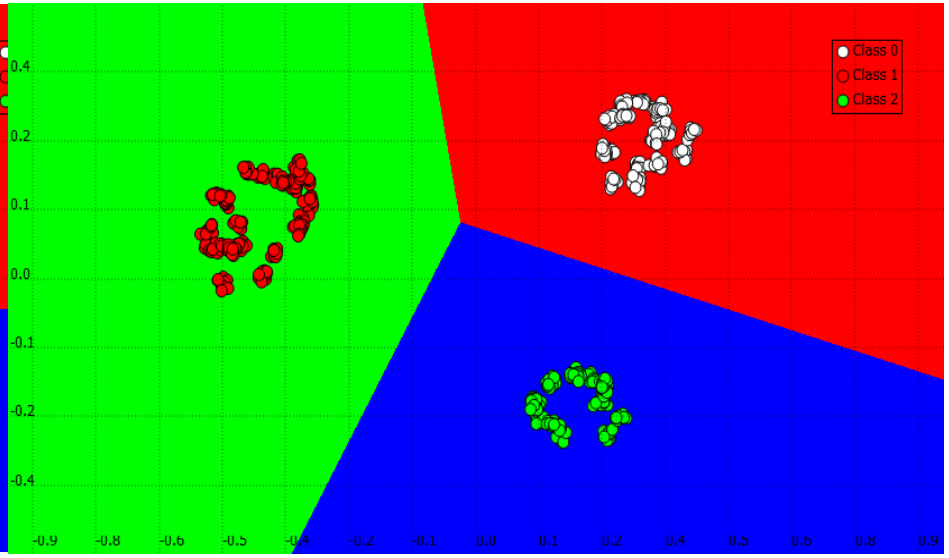
K-Means clustering minimizes a loss, often a quadratic cost function

$$J\left(\mu^1,...,\mu^K\right) = \sum_{k=1}^{K} \sum_{x^i \in c_k} d\left(x^i, \mu^k\right) \text{ with } d\left(x^i, \mu^k\right) = \sqrt[p]{\sum_{i=1}^{N} \left| x_i^i - \mu_i \right|^p}$$

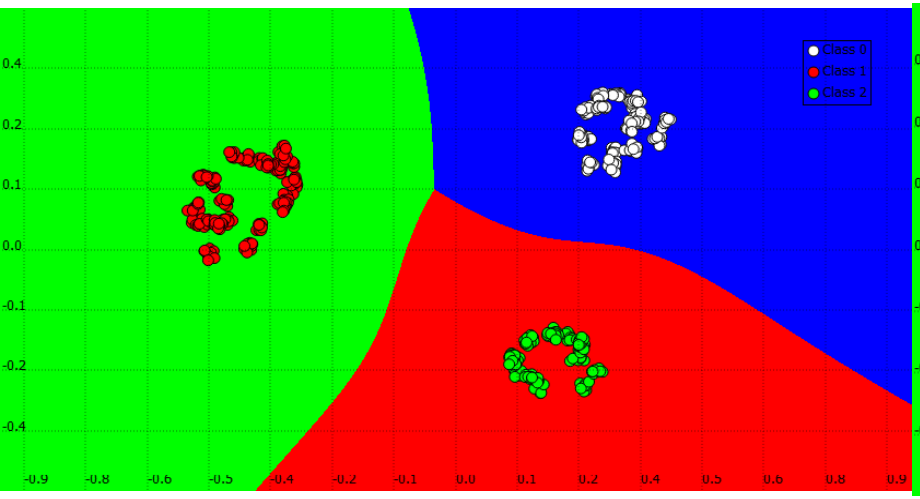
Effect of the distance metric on K-means



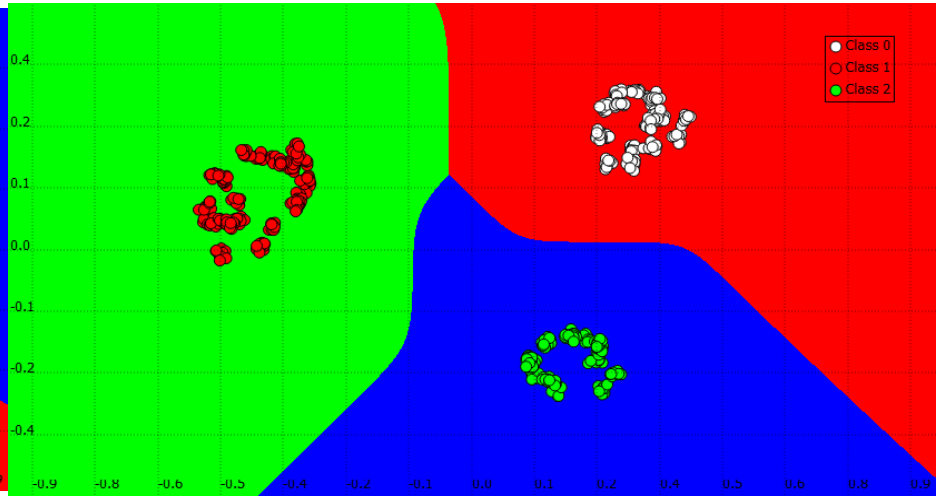
L1-Norm



L2-Norm



L3-Norm



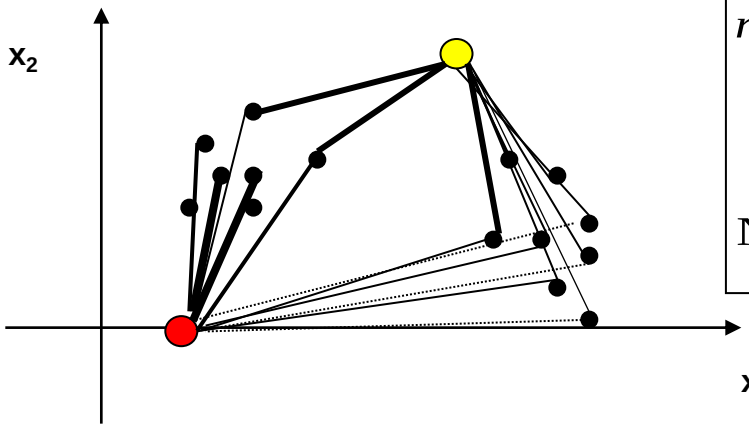
L8-Norm

K-means Clustering: Properties

- ✓ There are always K clusters.
- ✓ The clusters do not overlap:
  Soft K-means relaxes this assumption, see next slides
- ✓ Each member of a cluster is closer to its cluster than to any other cluster.
- ✓ *The algorithm is guaranteed to converge in a finite number of iterations*
- ✓ *But it converges to a local optimum!*
- ✓ *It is hence very sensitive to initialization of the centroids.*

Soft K-means Clustering

Soft K-means Clustering



r_i^k: responsibility of cluster k for point x^i

$$r_i^k = \frac{e^{\left(-\beta \cdot d\left(\mu^k, x^i\right)\right)}}{\sum_{k'} e^{\left(-\beta \cdot d\left(\mu^{k'}, x^i\right)\right)}} \in [0,1], \qquad \beta \in \mathbb{R}_+$$

Normalized over clusters: $\sum_k r_i^k = 1$

Assignment Step (E-step):
Assign each data point to the "closest" centroid.

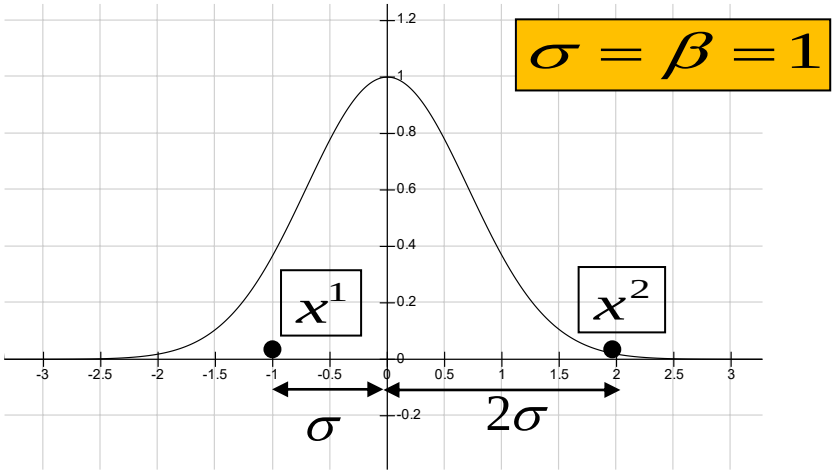
Each data point x^i is given a soft `degree of assignment' to each of the means μ^k.

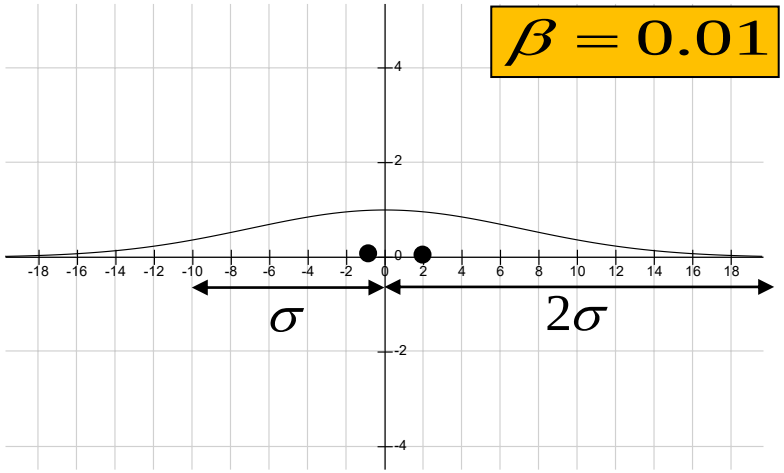
Soft K-means Clustering: The effect of beta

β is the stiffness

$\sigma \equiv \frac{1}{\sqrt{\beta}}$ measures the disparity across cluster

$$r_i^k = \frac{e^{\left(-\beta \cdot d\left(\mu^k, x^i\right)\right)}}{\sum_{k'} e^{\left(-\beta \cdot d\left(\mu^{k'}, x^i\right)\right)}}$$

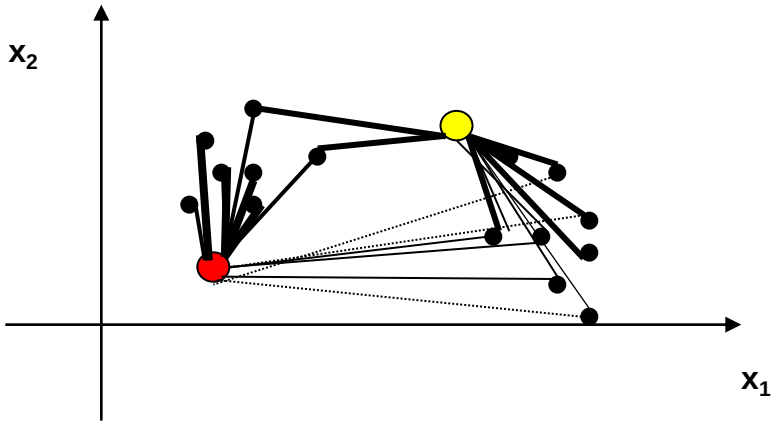




$$x^1 = -1 \quad r_1^1 = 0.99$$
$$x^2 = 2, \quad r_2^1 = 0.01$$

$$x^1 = -1, \quad r_1^1 = 0.51$$
$$x^2 = 2, \quad r_2^1 = 0.49$$

Soft K-means Clustering

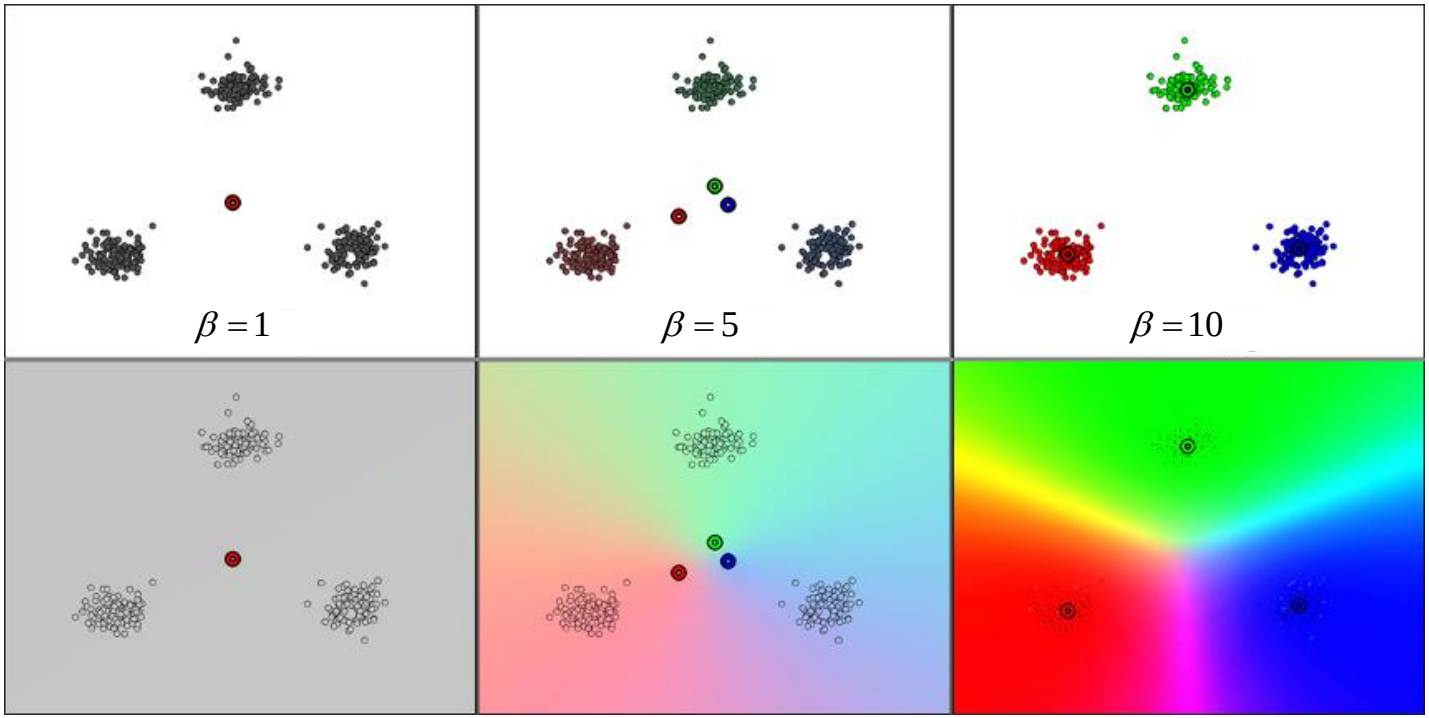


$$\mu^k = \frac{\sum_i r_i^k \cdot x^i}{\sum_i r_i^k}$$

Update step (M-Step):
Recompute the centroids based on the assignment of the points

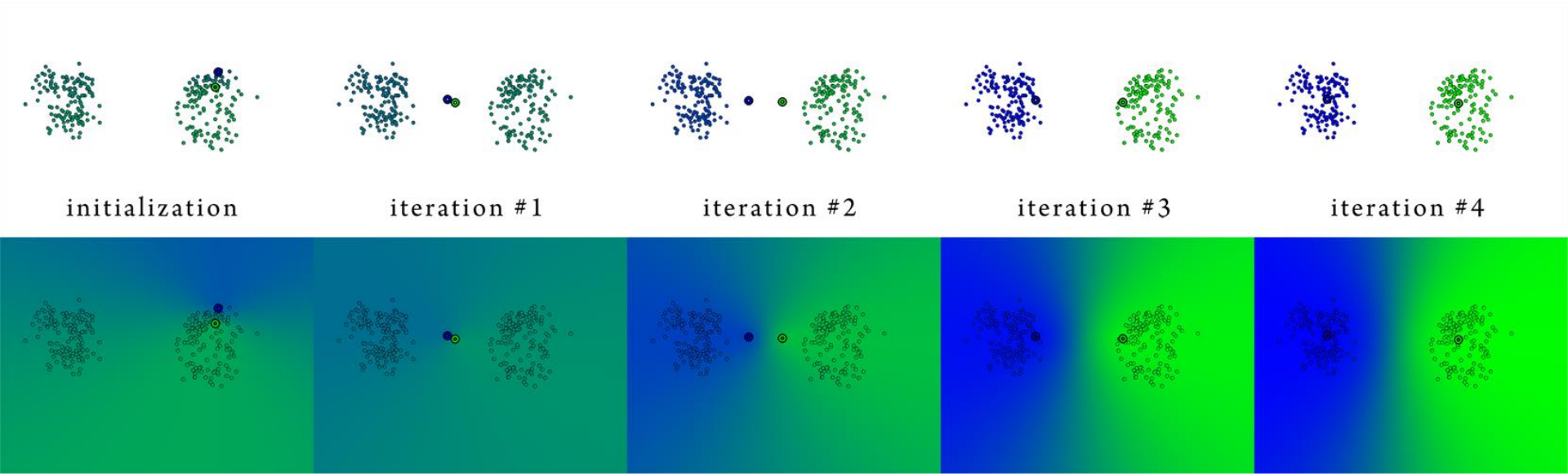
The update algorithm of the soft K-means is identical to that of the hard K-means, aside from the fact that the responsibilities to a particular cluster are now real numbers varying between 0 and 1.

Soft K-means Clustering



Soft K-means algorithm with a small (left), medium (center) and large (right) β

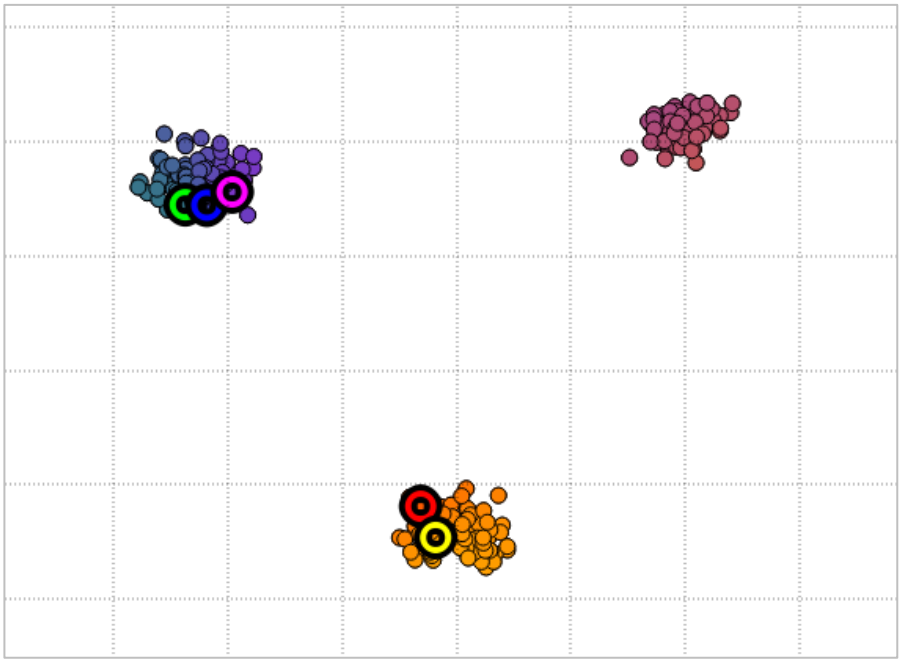
Soft K-means Clustering



Iterations of the Soft K-means algorithm from the random initialization (left) to convergence (right). Computed with $\beta = 10$.

Soft K-means Clustering

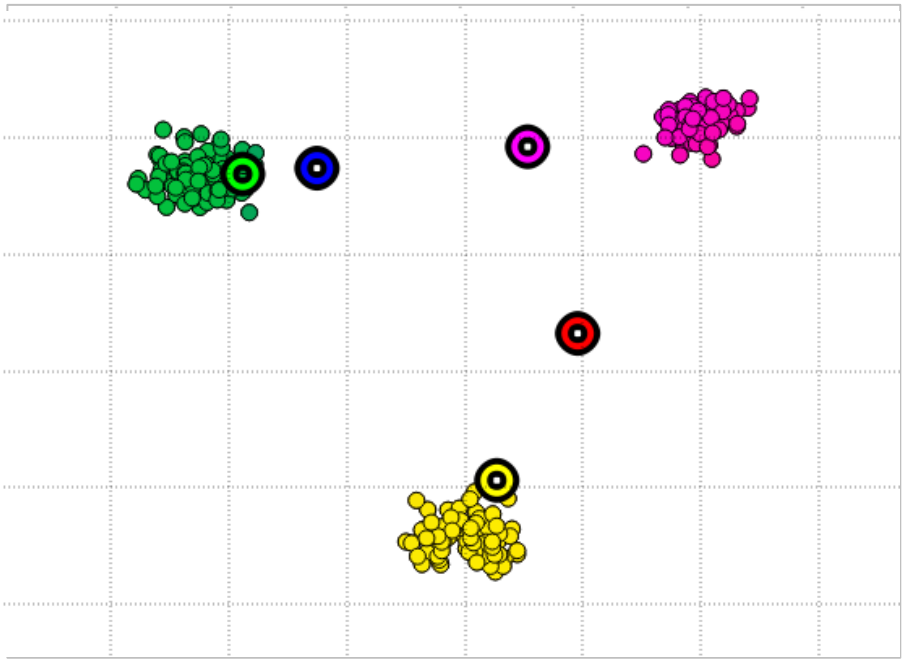
Soft K-means requires to set K, the number of clusters like K-means. But, sometimes, it can determine the true number of clusters.



Initialization: K=5
The 3 centroids are located in the cluster top left and 2 on the bottom cluster.

Soft K-means Clustering

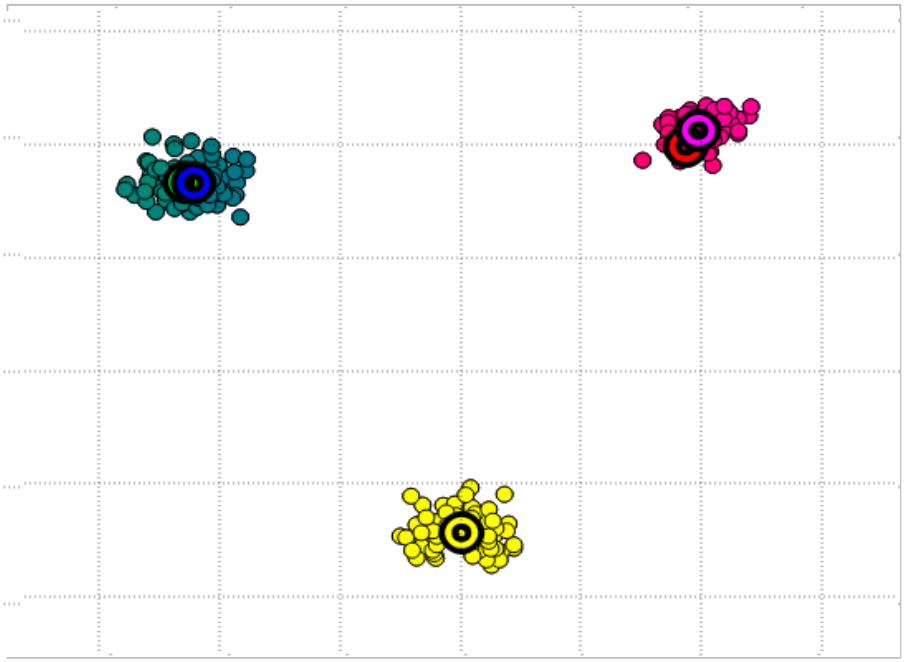
Soft K-means requires to set K, the number of clusters like K-means. But, sometimes, it can determine the true number of clusters.



After 2 iterations

Soft K-means Clustering

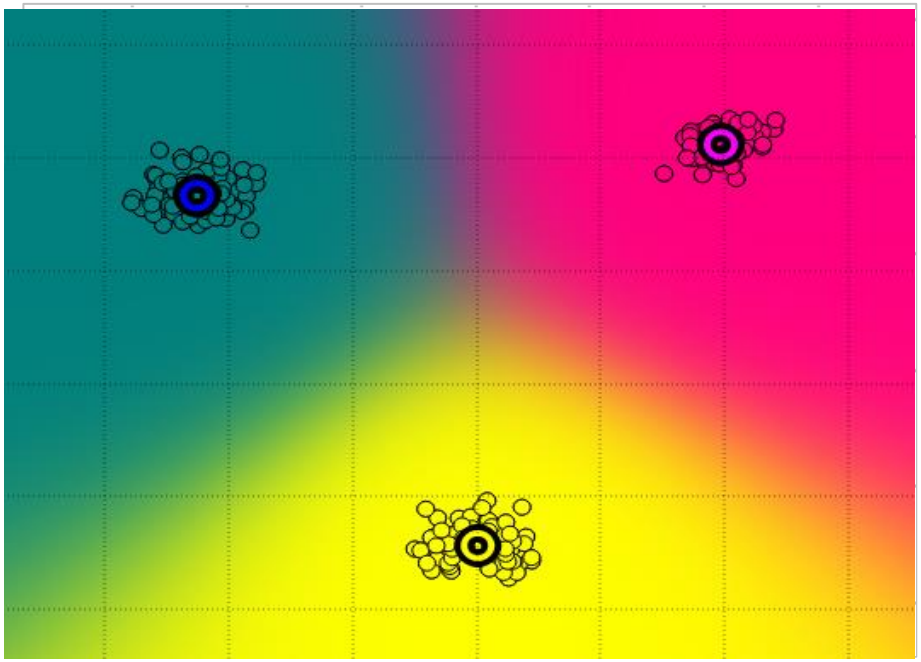
Soft K-means requires to set K, the number of clusters like K-means. But, sometimes, it can determine the true number of clusters.



After 4 iterations

Soft K-means Clustering

Soft K-means requires to set K, the number of clusters like K-means. But, sometimes, it can determine the true number of clusters.



At convergence

K-means / soft K-means Clustering: Advantages

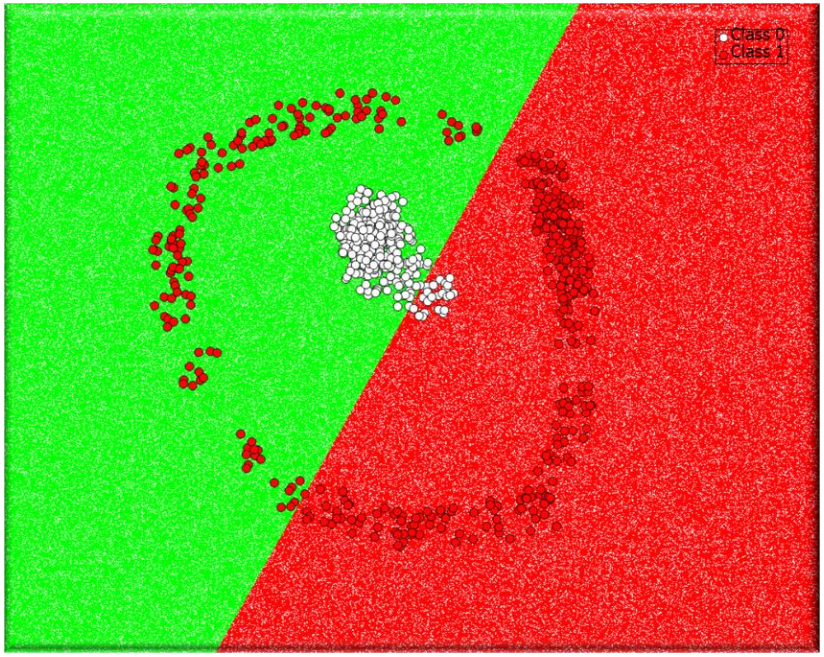
- The algorithm is guaranteed to converge in a finite number of iterations (but it converges to a local optimum!)
- It is computationally cheap and faster than other clustering techniques - update step is ~O(KM).

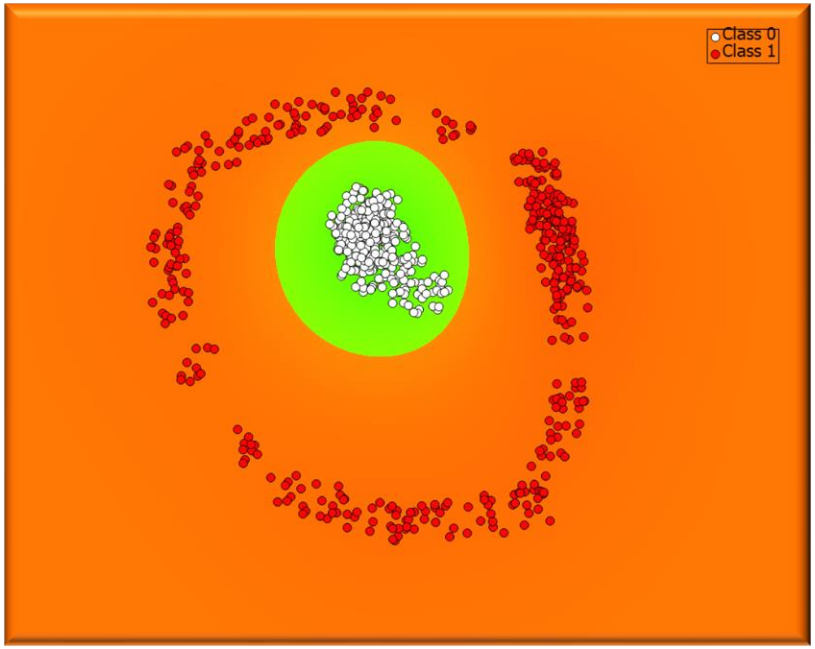
K-means / soft K-means Clustering: Drawbacks

- Does not work well with *non-globular* clusters.
- Sensitive to initialization
  Different initial partitions can result in different final clusters.
- Assumes a *fixed* number *K* of clusters
  In soft-K-means, clusters may merge and reduce to true number

→ It is, therefore, good practice to run the algorithm several times using different K values, to determine the optimal number of clusters.

Linear and nonlinear K-means





K-means is a linear technique and can separate clusters only linearly, or quasi-linearly (for norm-p, p>2).

Kernel K-means can separate clusters through non-linear boundaries, as shown above.

***Density*-based spatial clustering of applications with *noise* (*DBSCAN*)**

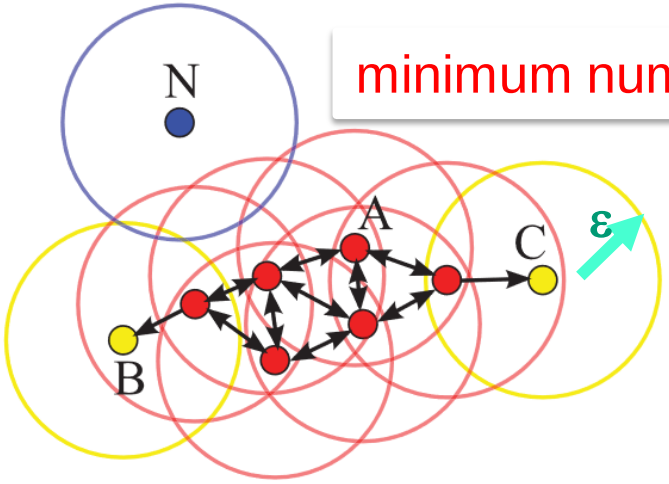
DBSCAN Principle

Minimum density level estimation, based on threshold for

1. the number of neighbors **minimum number of points**
2. located within **some radius ε**

Blue point not reachable = noise

minimum number of points >= 2



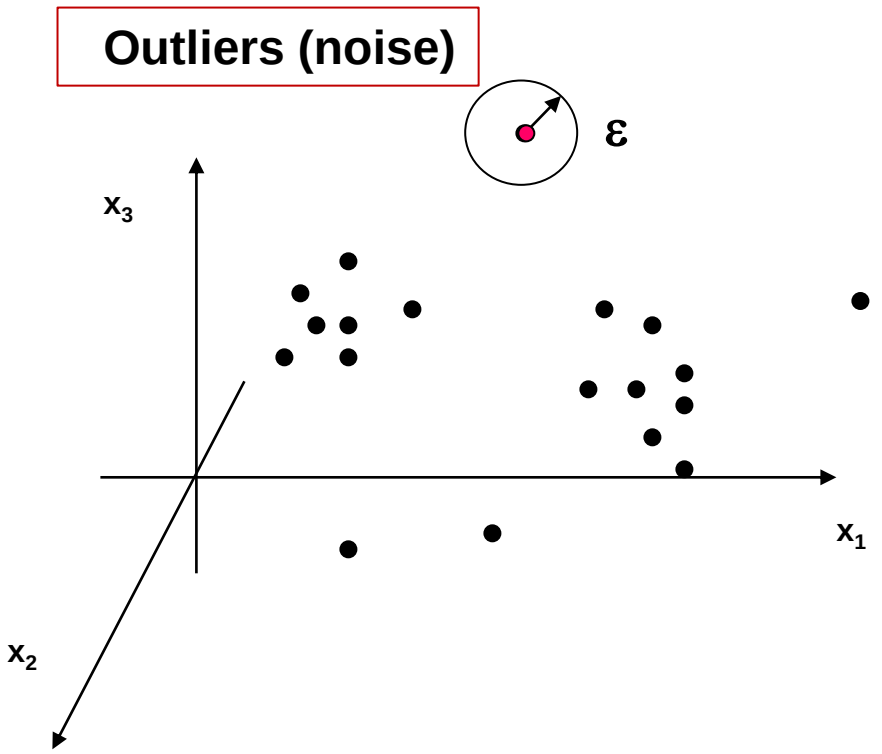
Red points are **core points** (direct density reachable)

Yellow points are **border points** (density reachable)

GOAL: Find areas, which satisfy the minimum density, and which are separated by areas of lower density. These area form a cluster.

Schubert, E. et al. "DBSCAN revisited, revisited: why and how you should (still) use DBSCAN." *ACM Trans. on Database Systems 2017*

DBSCAN Algorithm

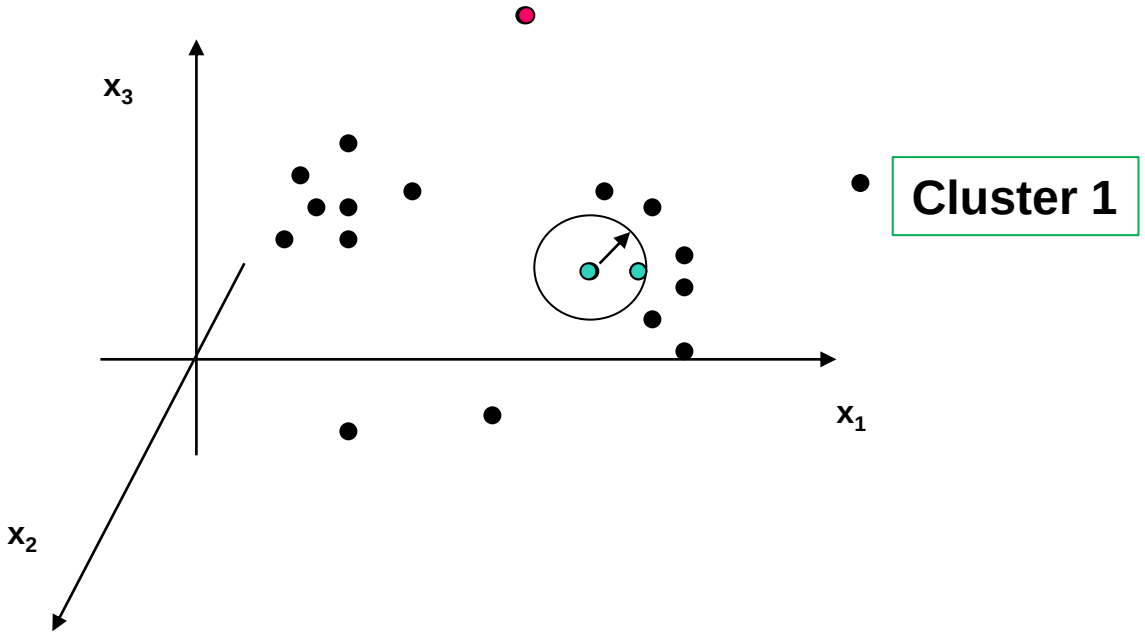


Set the hyperparameters:

- ε: size of neighborhood
- **mdata**: minimum number of datapoints

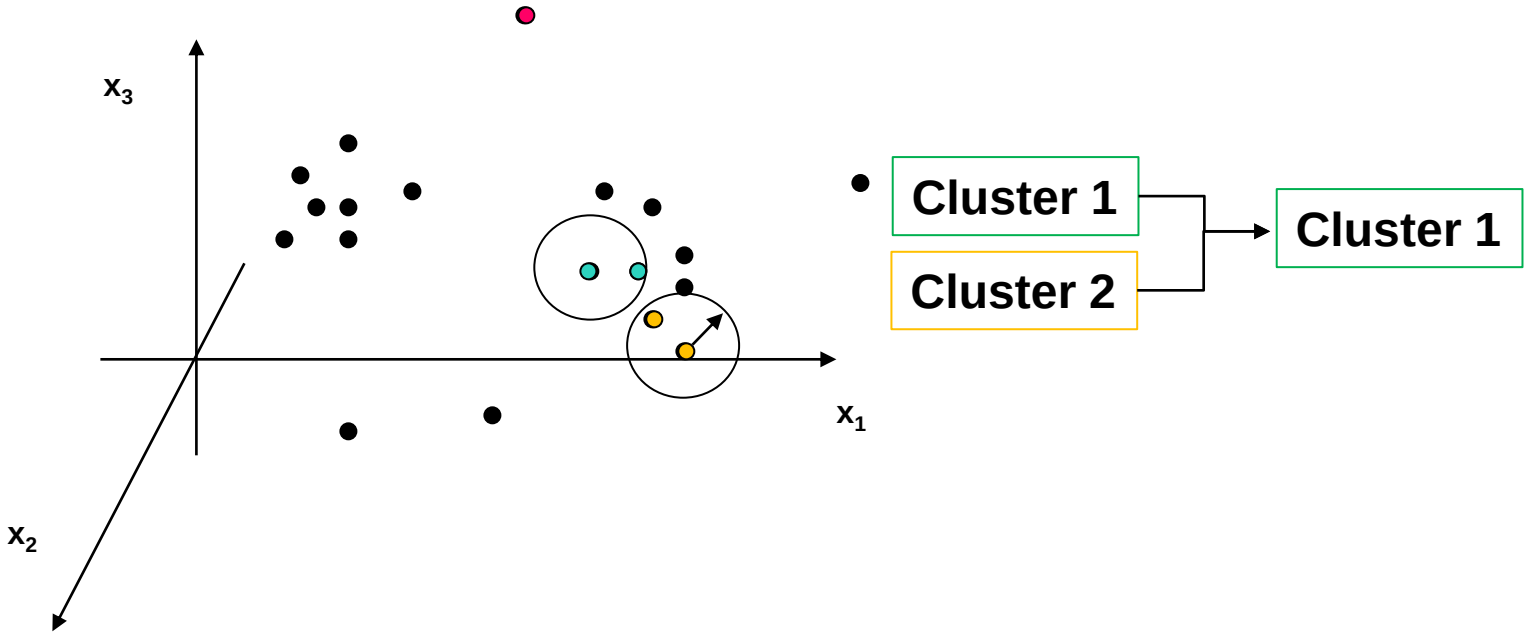
1. Pick first point in the database, or a point at random
2. If no label, compute number of datapoints within ε of this point
3. If this is < m_{data}, set this datapoint as an outlier
4. Go back to 1

DBSCAN Algorithm



1. Pick next point in the database, or a point at random
2. If no label, compute number of datapoints within ε of this point
3. If this is >= m_{data}, create a cluster
4. Go back to 1

DBSCAN Algorithm



1. Continue with the next points in the list
2. Merge two clusters if distance between clusters < ε

Comparison: K-means / DBSCAN

	K-means	DBSCAN
Hyperparameters	K: Nb of clusters	ε: size, Mdata: min. nb of datapoints
Computational cost	O(K*M) 	O(M*log(M)), M: nb datapoints 
Type of cluster	Globular	Non-globular (arbitrary shapes, non-linear boundaries) 
Robustness to noise	Not robust 	Robust to outliers within ε 

Both K-means and BDSCAN depend on choosing well the hyperparameters
→ To determine the hyperparameters, use evaluation methods for clustering (next)