KNN Clustering & DBscan

Iran University of Science and Technology

M. S. Tahaei PhD.

Fall 2024

Courtesy: some slides are adopted partly from Dr. Soleymani, Sharif University

Unsupervised learning

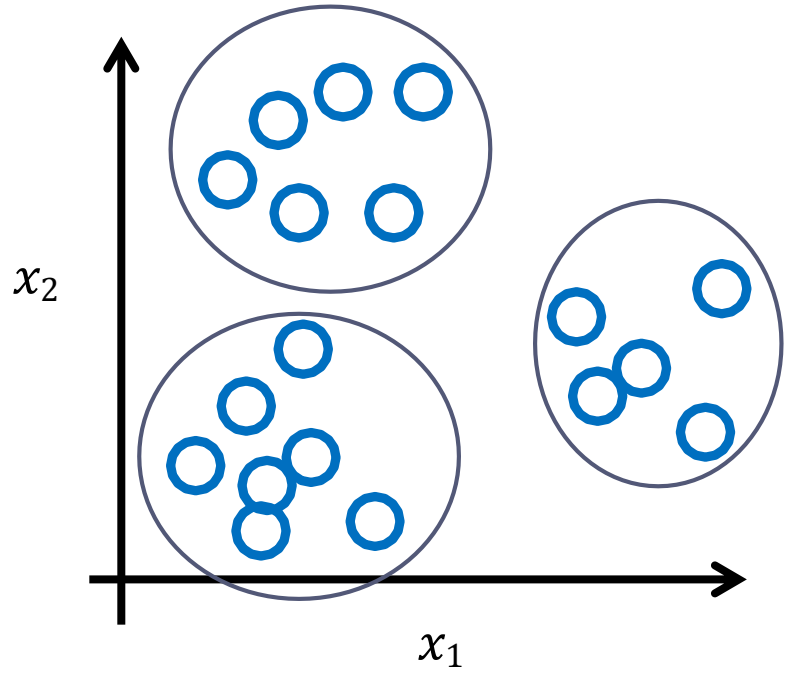
- **Clustering**: partitioning of data into groups of similar data points.

- **Density estimation**
  - Parametric & non-parametric density estimation

- **Dimensionality reduction**: data representation using a smaller number of dimensions while preserving (perhaps approximately) some properties of the data.

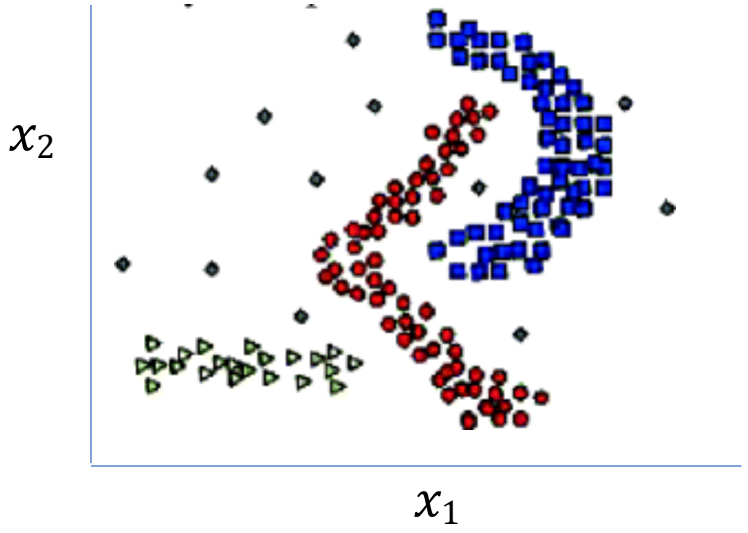
Clustering: Definition

- We have a set of unlabeled data points $\{\boldsymbol{x}^{(i)}\}_{i=1}^{N}$ and we intend to **find groups of similar objects** (based on the observed features)
  - high intra-cluster similarity
  - low inter-cluster similarity



Clustering: Another Definition

- Density-based definition:
  - Clusters are regions of high density that are separated from one another by regions of low density

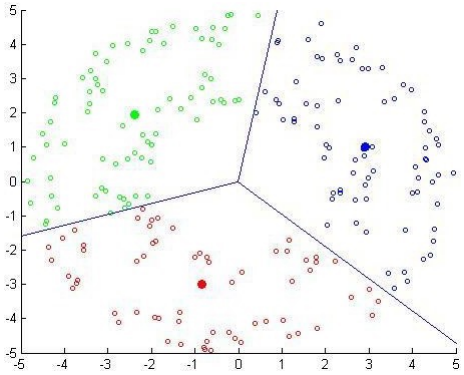


Clustering Purpose

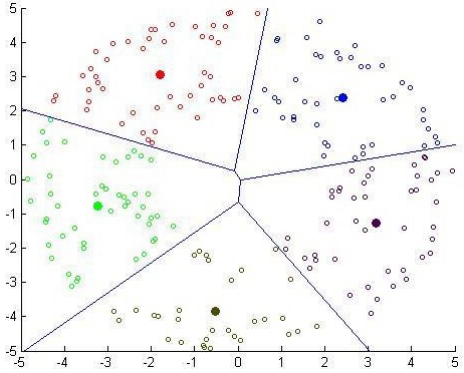
- **Preprocessing stage** to index, compress, or reduce the data
- Representing high-dimensional data in a low-dimensional space (e.g., for visualization purposes).
- As a tool to **understand the hidden structure** in data or to **group** them
  - To gain insight into the structure of the data prior to classifier design
  - To group the data when no label is available

K-means: Vector Quantization

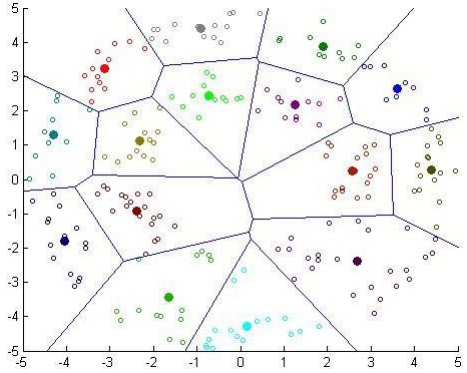
- Data Compression
  - Vector quantization: construct a codebook using k-means
    - cluster means as prototypes representing examples assigned to clusters.



$k = 3$



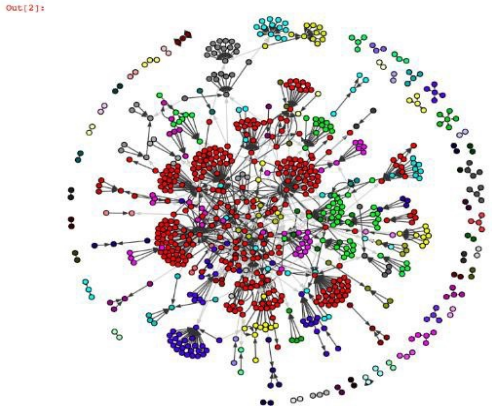
$k = 5$



$k = 15$

Clustering Applications

- Information retrieval (search and browsing)
  - Cluster text docs or images based on their content
  - Cluster groups of users based on their access patterns on webpages
- Cluster users of social networks by interest (community detection).



- **Bioinformatics**
  - cluster similar proteins together (similarity wrt chemical structure and/or functionality etc)
  - or cluster similar genes according to microarray data

Partitioning Algorithms: Basic Concept

- Construct a partition of a set of N objects into a set of K clusters
  - The number of clusters K is given in advance
  - Each object belongs to **exactly one** cluster in hard clustering methods

- K-means is the most popular partitioning algorithm

Objective Based Clustering

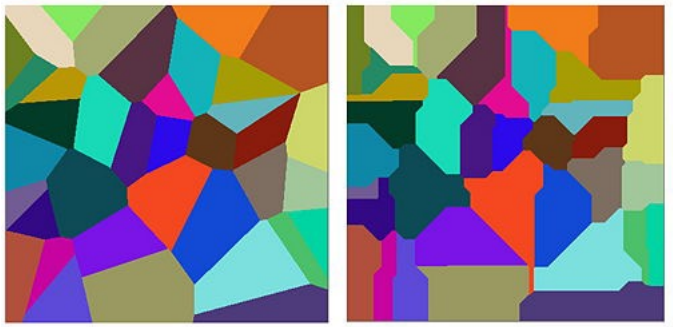
- **Input**: A set of N points, also a distance/dissimilarity measure
- **Output**: a partition of the data.

- **k-median**: find center pts $\mathbf{c}_1, \mathbf{c}_2, \ldots, \mathbf{c}_K$ to minimize

$$\sum_{i=1}^{N} \min_{j \in 1,\ldots,K} d(\boldsymbol{x}^{(i)}, \boldsymbol{c}_j)$$

- **k-means**: find center pts $\mathbf{c}_1, \mathbf{c}_2, \ldots, \mathbf{c}_K$ to minimize

$$\sum_{i=1}^{N} \min_{j \in 1,\ldots,K} d^2(\boldsymbol{x}^{(i)}, \boldsymbol{c}_j)$$



- **k-center**: find partition to minimize the maxim radius

Distance Measure

- Let O_1 and O_2 be two objects from the universe of possible objects. The distance (dissimilarity) between O_1 and O_2 is a real number denoted by $d(O_1, O_2)$

- Specifying the distance $d(\boldsymbol{x}, \boldsymbol{x}')$ between pairs $(\boldsymbol{x}, \boldsymbol{x}')$.
  - E.g.,# keywords in common, edit distance
  - Example: Euclidean distance in the space of features

K-means Clustering

- **Input**: a set $\boldsymbol{x}^{(1)}, \ldots, \boldsymbol{x}^{(N)}$ of data points (in a d-dim feature space) and an integer K

- **Output**: a set of K representatives $\boldsymbol{c}_1, \boldsymbol{c}_2, \ldots, \boldsymbol{c}_K \in \mathbb{R}^d$ as the cluster representatives
  - data points are assigned to the clusters according to their distances to $\boldsymbol{c}_1, \boldsymbol{c}_2, \ldots, \boldsymbol{c}_K$
    - Each data is assigned to the cluster whose representative is nearest to it

- **Objective**: choose $\boldsymbol{c}_1, \boldsymbol{c}_2, \ldots, \boldsymbol{c}_K$ to minimize:

$$\sum_{i=1}^{N} \min_{j \in 1,\ldots,K} d^2(\boldsymbol{x}^{(i)}, \boldsymbol{c}_j)$$

Euclidean k-means Clustering

- **Input**: a set $\boldsymbol{x}^{(1)}, \ldots, \boldsymbol{x}^{(N)}$ of data points (in a d-dim feature space) and an integer K

- **Output**: a set of K representatives $\boldsymbol{c}_1, \boldsymbol{c}_2, \ldots, \boldsymbol{c}_K \in \mathbb{R}^d$ as the cluster representatives
  - data points are assigned to the clusters according to their distances to $\boldsymbol{c}_1, \boldsymbol{c}_2, \ldots, \boldsymbol{c}_K$
    - Each data is assigned to the cluster whose representative is nearest to it

- **Objective**: choose $\boldsymbol{c}_1, \boldsymbol{c}_2, \ldots, \boldsymbol{c}_K$ to minimize:

$$\sum_{i=1}^{N} \min_{j \in 1,\ldots,K} \left\| \boldsymbol{x}^{(i)} - \boldsymbol{c}_j \right\|^2$$

each point assigned to its closest cluster representative

Euclidean k-means Clustering: Computational Complexity

- To find the optimal partition, we need to exhaustively enumerate all partitions
  - In how many ways can we assign k labels to N observations?
- NP hard: even for $k = 2$ or $d = 2$
- For k=1: $\min_{\boldsymbol{c}} \sum_{i=1}^{N} \left\| \boldsymbol{x}^{(i)} - \boldsymbol{c} \right\|^2$
  - $\boldsymbol{c} = \boldsymbol{\mu} = \frac{1}{N} \sum_{i=1}^{N} \boldsymbol{x}^{(i)}$
- For $d = 1$, dynamic programming in time $O(N^2 K)$.

Common Heuristic in Practice: The Lloyd's method

- Input:A set $\mathcal{X}$ of N datapoints $\boldsymbol{x}^{(1)}, \ldots, \boldsymbol{x}^{(N)}$ in $\mathbb{R}^d$

- **Initialize** centers $\boldsymbol{c}_1, \boldsymbol{c}_2, \ldots, \boldsymbol{c}_K \in \mathbb{R}^d$ in any way.
- **Repeat** until there is no further change in the cost.
  - For each j: $\mathcal{C}_j \leftarrow \{\boldsymbol{x} \in \mathcal{X} | \text{where } \boldsymbol{c}_j \text{ is the closest center to } \boldsymbol{x}\}$
  - For each j: $\boldsymbol{c}_1 \leftarrow$mean of members of $\mathcal{C}_j$

Holding centers $\boldsymbol{c}_1, \boldsymbol{c}_2, \ldots, \boldsymbol{c}_K$ fixed
Find optimal assignments $\mathcal{C}_1, \ldots, \mathcal{C}_K$ of data points to clusters

Holding cluster assignments $\mathcal{C}_1, \ldots, \mathcal{C}_K$ fixed
Find optimal centers $\boldsymbol{c}_1, \boldsymbol{c}_2, \ldots, \boldsymbol{c}_K$

K-means Algorithm (The Lloyd's method)

Select k random points $\boldsymbol{c}_1, \boldsymbol{c}_2, \ldots \boldsymbol{c}_k$ as clusters' initial centroids.

Repeat until *converges* (or other stopping criterion):

for i=1 to N do:

Assign $\boldsymbol{x}^{(i)}$ to the closet cluster and thus $\mathcal{C}_j$ contains all data that are closer to $\boldsymbol{c}_j$ than to anyother cluster

for j=1 to k do

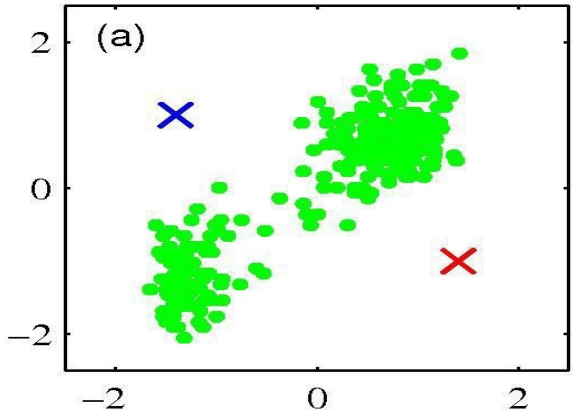
$$\boldsymbol{c}_j = \frac{1}{|\mathcal{C}_j|} \sum_{\boldsymbol{x}^{(i)} \in \mathcal{C}_j} \boldsymbol{x}^{(i)}$$

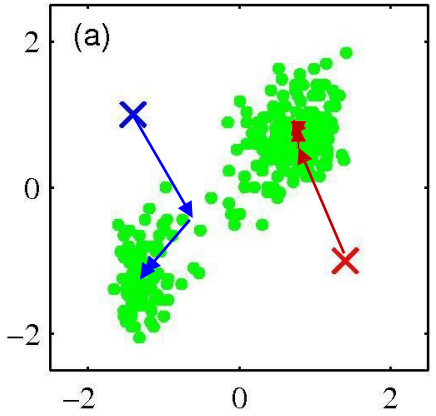
Assign data based on current centers

Re-estimate centers based on current assignment

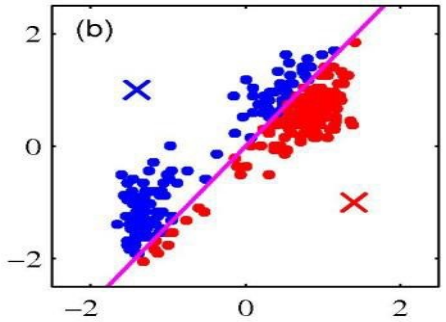
Assigning data to clusters

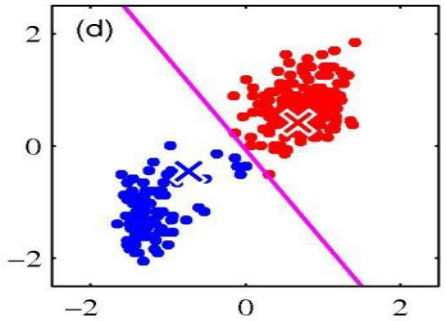
Updating means

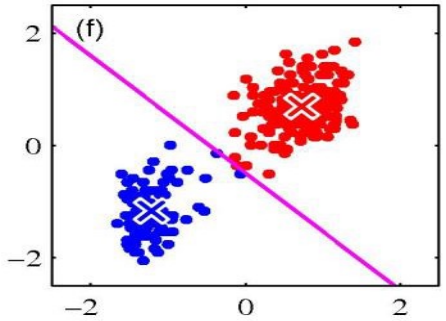


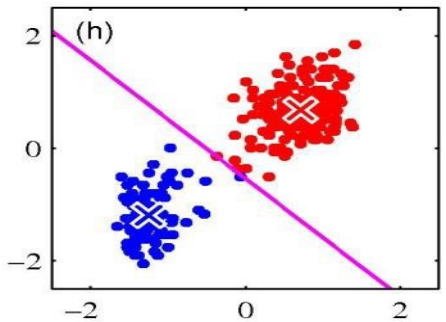


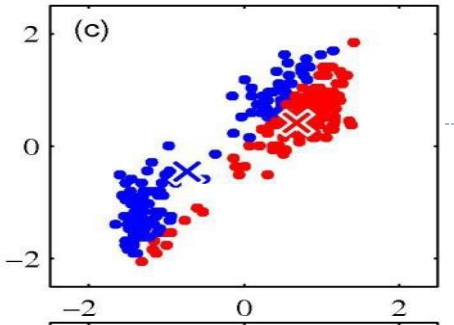
[Bishop]

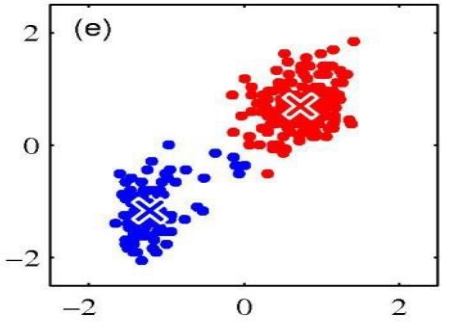


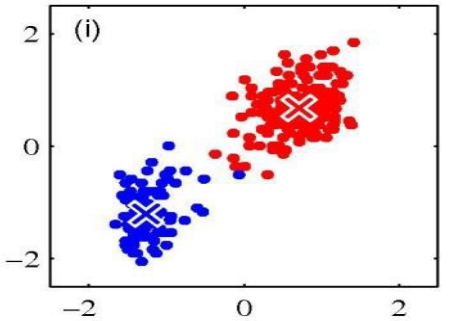


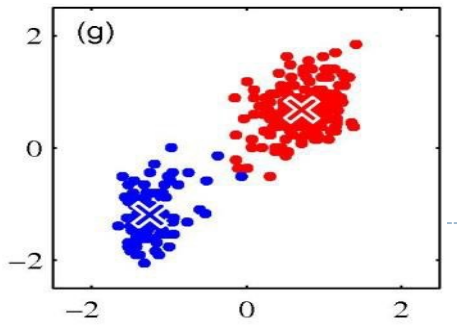












Intra-cluster similarity

- k-means optimizes intra-cluster similarity:

$$J(\mathcal{C}) = \sum_{j=1}^{K} \sum_{\boldsymbol{x}^{(i)} \in \mathcal{C}_j} \left\| \boldsymbol{x}^{(i)} - \boldsymbol{c}_j \right\|^2$$

$$\boldsymbol{c}_j = \frac{1}{|\mathcal{C}_j|} \sum_{\boldsymbol{x}^{(i)} \in \mathcal{C}_j} \boldsymbol{x}^{(i)}$$

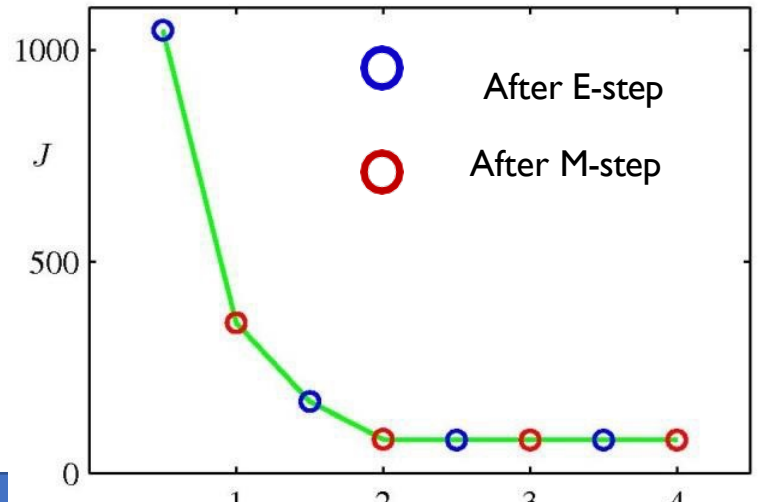
$$\sum_{\boldsymbol{x}^{(i)} \in \mathcal{C}_j} \left\| \boldsymbol{x}^{(i)} - \boldsymbol{c}_j \right\|^2 = \frac{1}{2|\mathcal{C}_j|} \sum_{\boldsymbol{x}^{(i)} \in \mathcal{C}_j} \sum_{\boldsymbol{x}^{(i')} \in \mathcal{C}_j} \left\| \boldsymbol{x}^{(i)} - \boldsymbol{x}^{(i')} \right\|^2$$

the average distance to members of the same cluster

K-means: Convergence

- It always converges.

- Why should the *K*-means algorithm ever reach a state in which clustering doesn't change.
  - Reassignment stage monotonically decreases J since each vector is assigned to the closest centroid.
  - Centroid update stage also for each cluster minimizes the sum of squared distances of the assigned points to the cluster from its center.



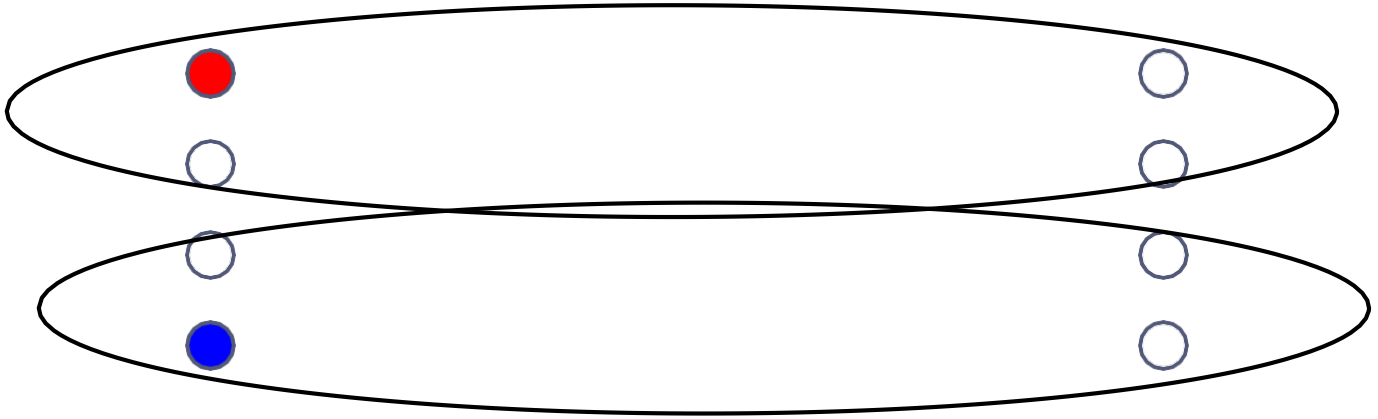
[Bishop]

Local optimum

- It always converges
- but it may converge at a local optimum that is different from the global optimum
  - may be arbitrarily worse in terms of the objective score.

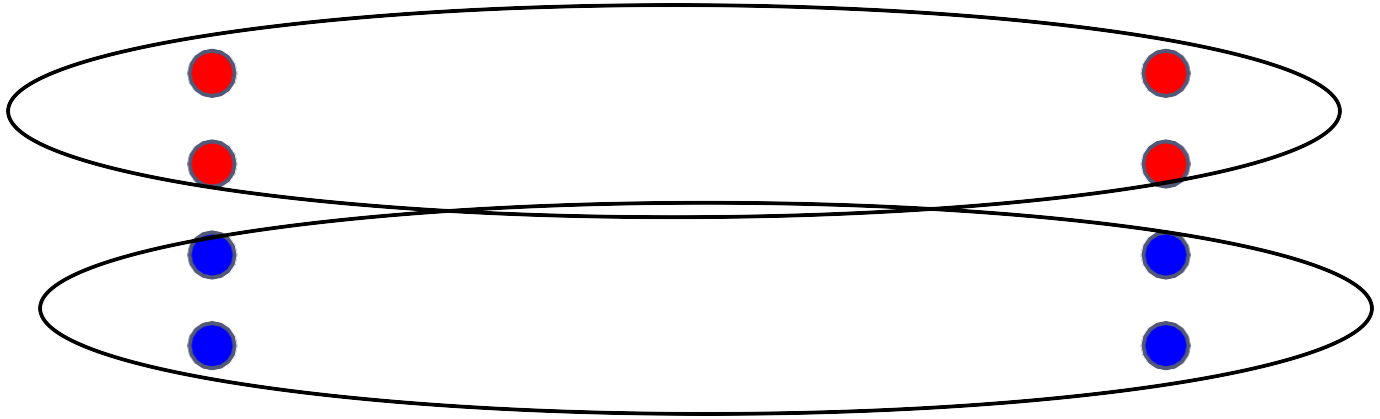
Local optimum

- It always converges
- but it may converge at a local optimum that is different from the global optimum
  - may be arbitrarily worse in terms of the objective score.



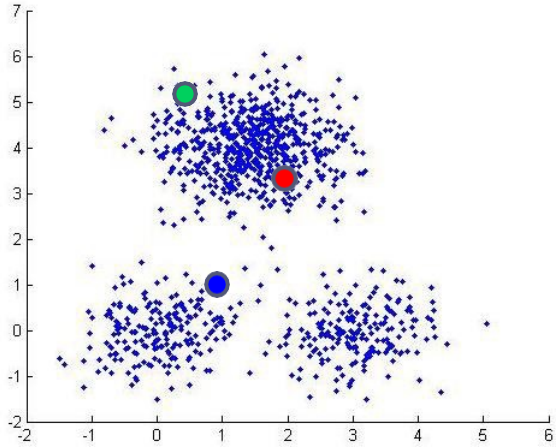
Local optimum

- It always converges
- but it may converge at a local optimum that is different from the global optimum
  - may be arbitrarily worse in terms of the objective score.

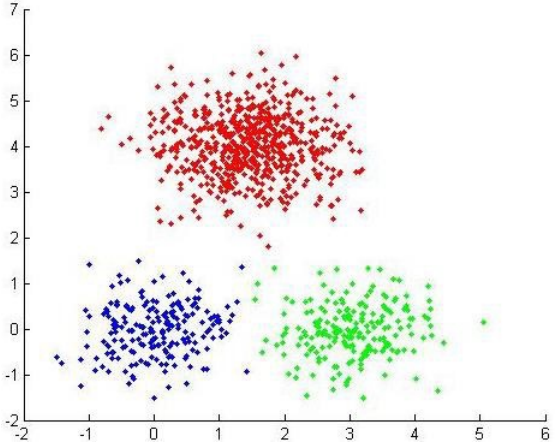


Local optimum: every point is assigned to its nearest center and every center is the mean value of its points.

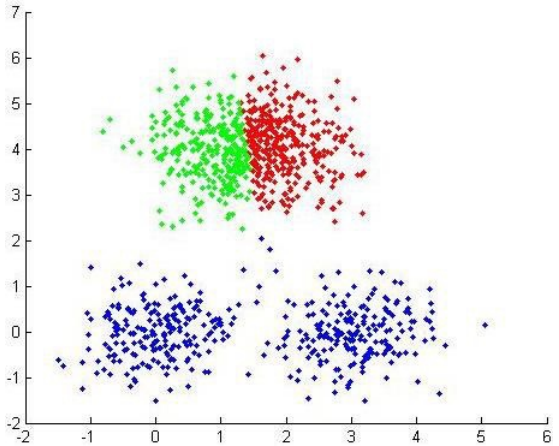
K-means: Local Minimum Problem



Original Data



Optimal Clustering



The obtained Clustering

The Lloyd's method: Initialization

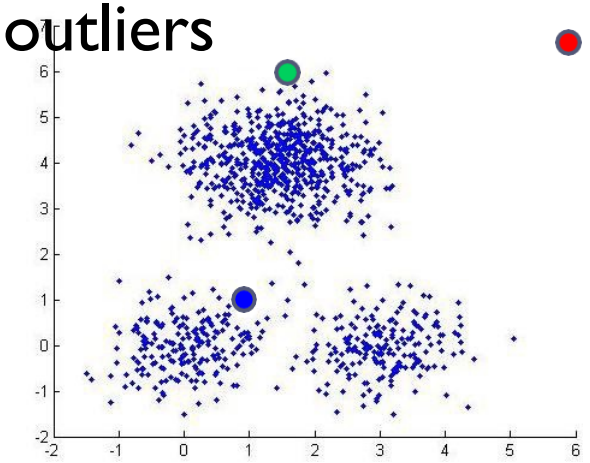
- Initialization is crucial (how fast it converges, quality of clustering)
  - Random centers from the data points
    - Multiple runs and select the best ones
  - Initialize with the results of another method
  - Select good initial centers using a heuristic
    - Furthest traversal
    - K-means ++ (works well and has provable guarantees)

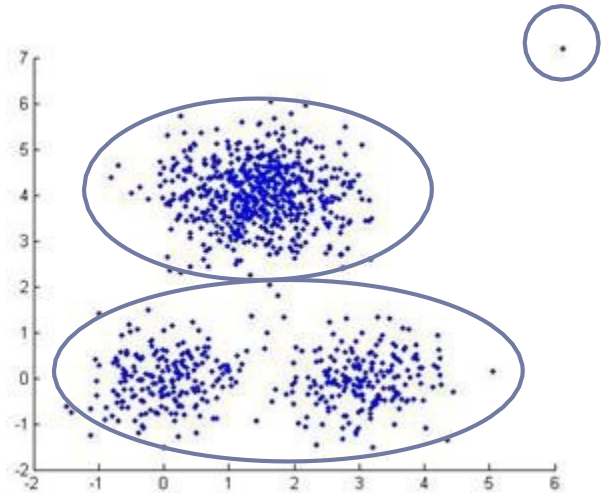
Another Initialization Idea: Furthest Point Heuristic

- Choose $\boldsymbol{c}_1$ arbitrarily (or at random).
- For $j = 2, \ldots, K$
  - Select $\boldsymbol{c}_j$ among datapoints $\boldsymbol{x}^{(1)}, \ldots, \boldsymbol{x}^{(N)}$ that is farthest from previously chosen $\boldsymbol{c}_1, \ldots, \boldsymbol{c}_{j-1}$

Another Initialization Idea: Furthest Point Heuristic

- It is sensitive to outliers





K-means++ Initialization: D2 sampling [AV07]

- Combine random initialization and furthest point initialization ideas
- Let the probability of selection of the point be proportional to the distance between this point and its nearest center.
  - probability of selecting of $\boldsymbol{x}$ is proportional to $D^2(\boldsymbol{x}) = \min_{k<j} \|\boldsymbol{x} - \boldsymbol{c}_k\|^2$.

- Choose $\boldsymbol{c}_1$ arbitrarily (or at random).
- For $j = 2, \ldots, K$
  - Select $\boldsymbol{c}_j$ among data points $\boldsymbol{x}^{(1)}, \ldots, \boldsymbol{x}^{(N)}$ according to the distribution:

$$\Pr(\boldsymbol{c}_j = \boldsymbol{x}^{(i)}) \propto \min_{k<j} \left\|\boldsymbol{x}^{(i)} - \boldsymbol{c}_k\right\|^2$$

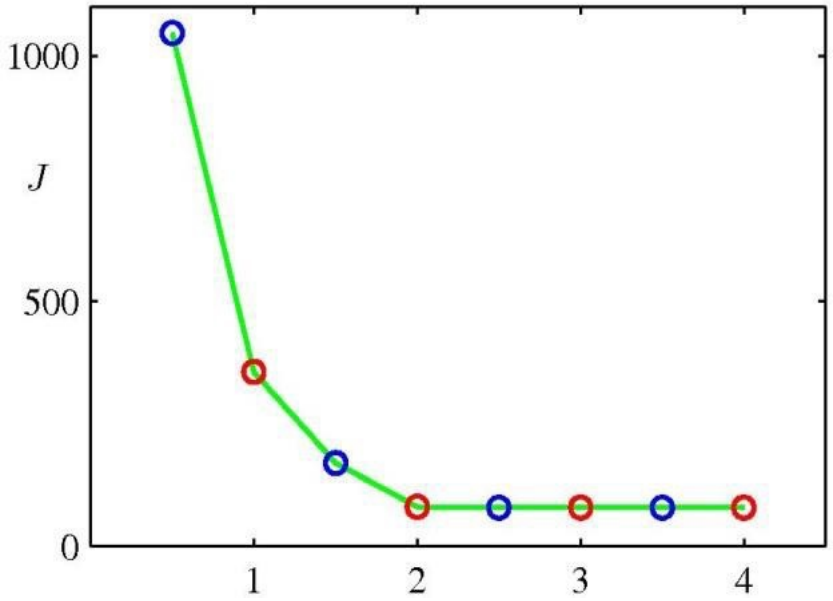
- **Theorem:** K-means++ always attains an $O(\log k)$ approximation to optimal k-means solution in expectation.

How Many Clusters?

- Number of clusters k is given in advance in the k-means algorithm
  - However, finding the "right" number of clusters is a part of the problem
- Tradeoff between having better focus within each cluster and having too many clusters
- Hold-out validation/cross-validation on auxiliary task (e.g., supervised learning task).
- Optimization problem: penalize having lots of clusters
  - some criteria can be used to automatically estimate *k*
    - Penalize the number of bits you need to describe the extra parameter

$$J'(\mathcal{C}) = J(\mathcal{C}) + |\mathcal{C}| \times \log N$$

Elbow finding



- Heuristic: Find large gap between $k-1$-means cost and k- means cost.
  - "knee finding" or "elbow finding".

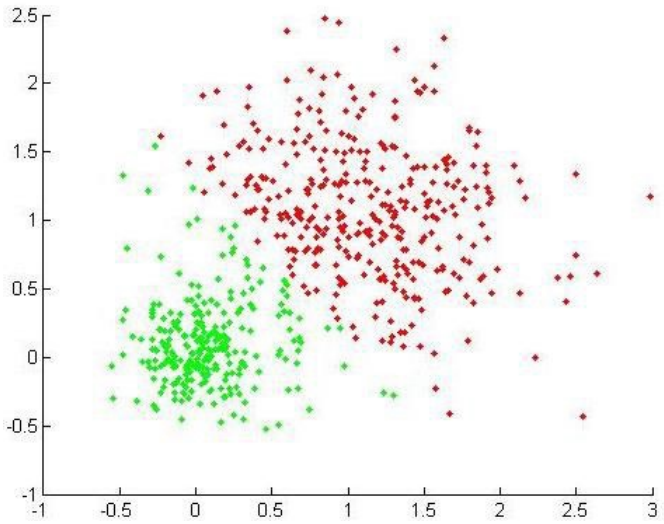
K-means: Advantages and disadvantages

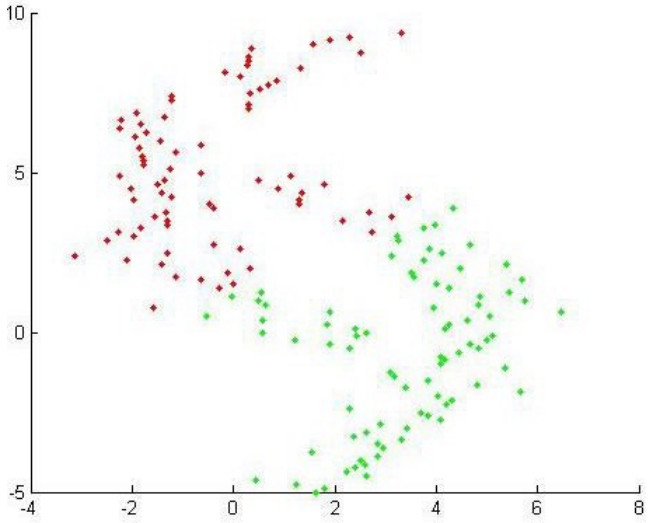
- Strength
  - It is a simple method
  - Relatively efficient: $O(tKNd)$, where t is the number of iterations.
    - Usually $t \ll n$.
    - *K*-means typically converges quickly

- Weakness
  - Need to specify *K*, the *number* of clusters, in advance
  - Often terminates at a *local optimum*.
  - Not suitable to discover clusters with arbitrary shapes
  - Works for numerical data. What about categorical data?
  - Noise and outliers can be considerable trouble to *K*-means

k-means Algorithm: Limitation

- In general, k-means is unable to find clusters of arbitrary shapes, sizes, and densities
  - Except to very distant clusters





K-means

- K-means was proposed near 60 years ago
  - thousands of clustering algorithms have been published since then
  - However, K-means is still widely used.

- This speaks to the difficulty in designing a general purpose clustering algorithm and the ill-posed problem of clustering.

Density-Based Clustering

Clustering based on density (local cluster criterion), such as density-connected points

Major features:

Discover clusters of arbitrary shape

Handle noise

One scan

Need density parameters as termination condition

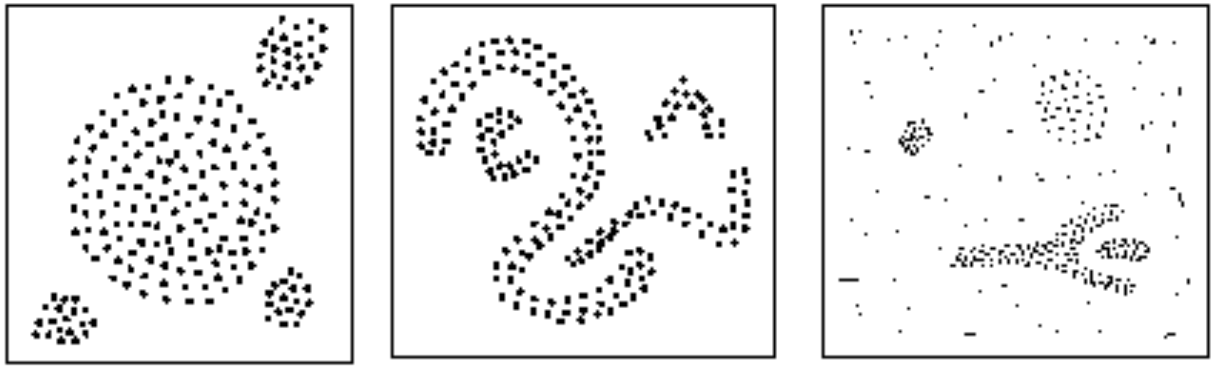
Several interesting studies:

DBSCAN: Ester, et al. (KDD'96)

OPTICS: Ankerst, et al (SIGMOD'99).

CLIQUE: Agrawal, et al. (SIGMOD'98)

Density-Based Clustering



Clustering based on density (local cluster criterion), such as density-connected points

Each cluster has a considerable higher density of points than outside of the cluster

DBSCAN

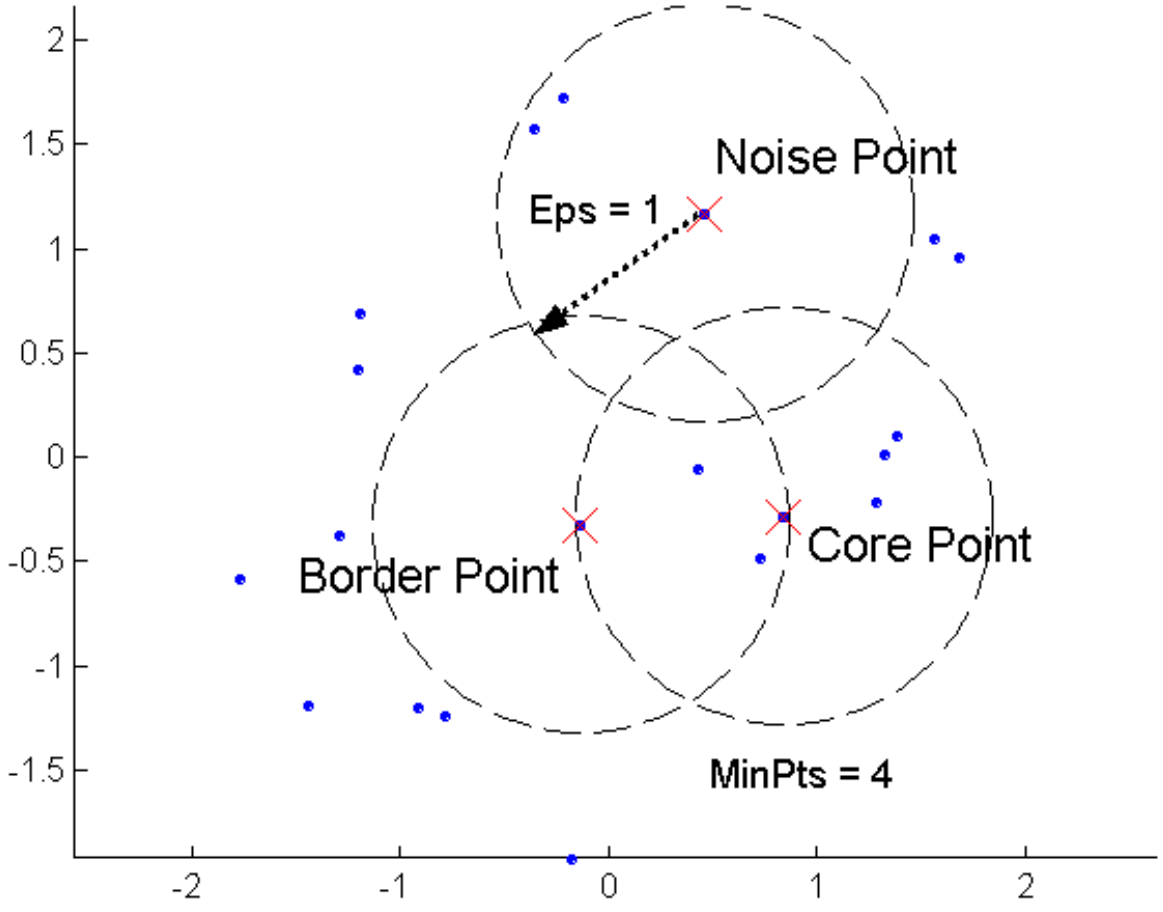
DBSCAN is a density-based algorithm.

- Density = number of points within a specified radius r (Eps)
- A point is a core point if it has more than a specified number of points (MinPts) within Eps

These are points that 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.

DBSCAN: Core, Border, and Noise points



DBSCAN

Two parameters (eps and MinPts)***:***

- ε: Maximum radius of the neighbourhood
- ***MinPts***: Minimum number of points in an Eps-neighbourhood of that point
- $N_{\varepsilon}(p)$: ***{q belongs to D | dist(p,q) <= ε}***

Directly density-reachable**:** A point ***p*** is directly density-reachable from a point ***q*** wrt. ε, ***MinPts*** if

1) p belongs to $N_{\varepsilon}(q)$

2) core point condition:

$|N_{\varepsilon}(q)| >= MinPts$

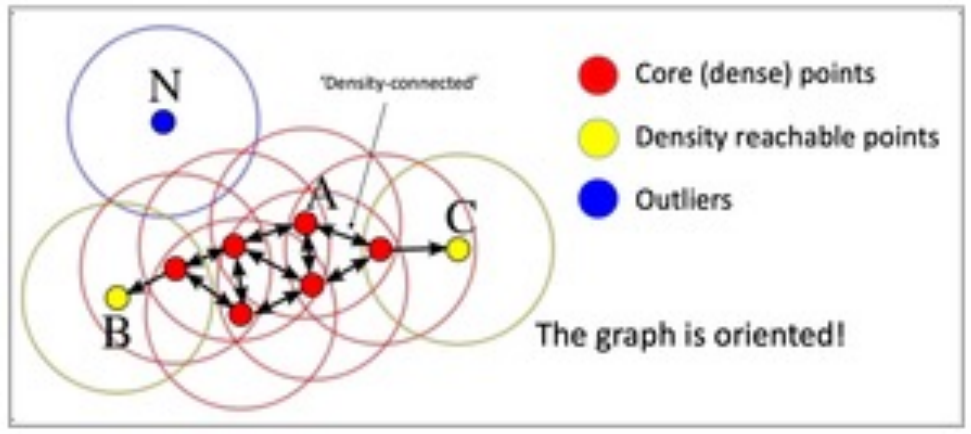
Density-Reachable and Density-Connected

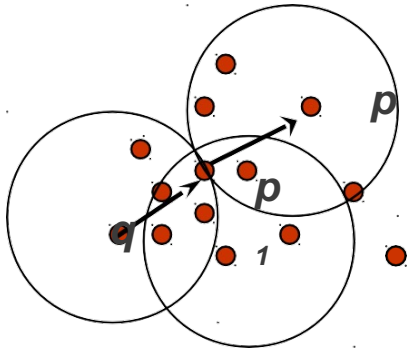
(w.r.t. *Eps*, *MinPts)*

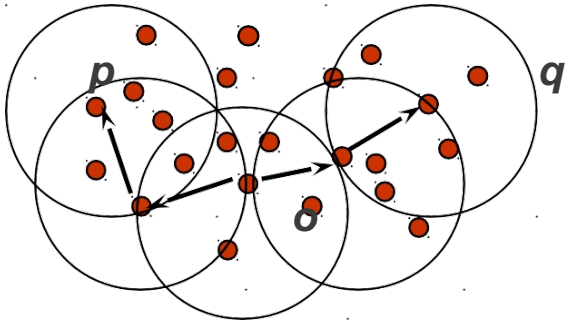
Let p be a core point, then every point in its *Eps* neighborhood is said to be directly density-reachable from p.

A point *p* is density-reachable from a point core point *q* if there is a chain of points $p_1, \ldots, p_n, p_1 = q, p_n = p$

A point *p* is density-connected to a point *q* if there is a point *o* such that both, *p* and *q* are density-reachable from *o*

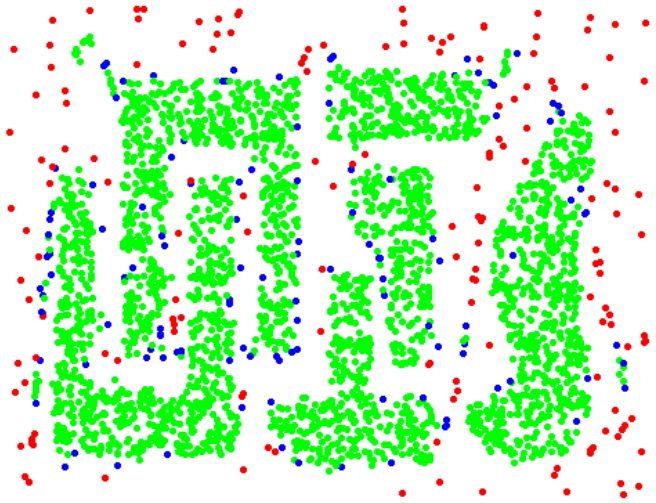






DBSCAN: Large Eps

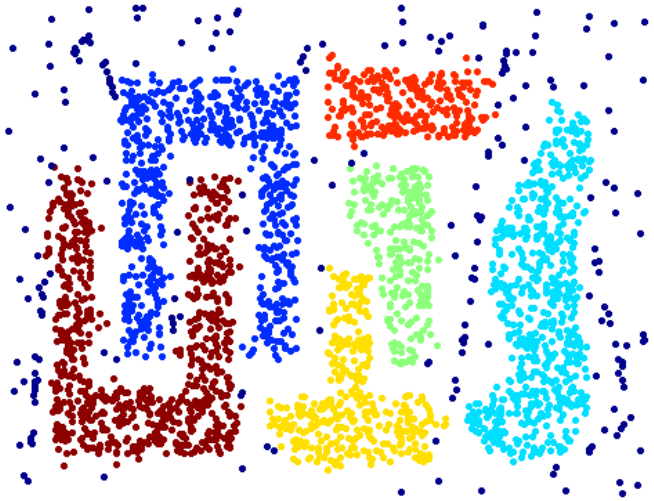
Original Points



Point types: core, border and noise

DBSCAN: Optimal Eps

Original Points

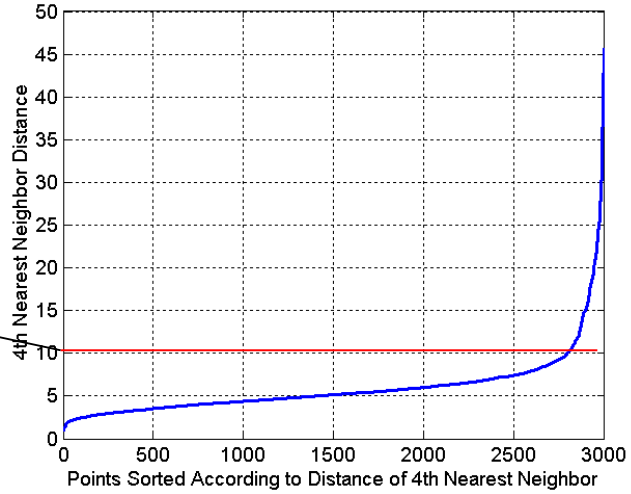


Clusters

Determining Eps and MinPts

- Idea is that for points in a cluster, their k^{th} nearest neighbors are at roughly the same distance
- Noise points have the k^{th} nearest neighbor at farther distance
- So, plot sorted distance of every point to its k^{th} nearest neighbor (e.g., k=4)

Thus, eps = 10



DBSCAN: Algorithm

Let ClusterCount=0. For every point *p*:

1. If *p* it is not a core point, assign a null label to it [e.g., zero]
2. If *p* is a core point, a new cluster is formed

   [with label ClusterCount:= ClusterCount+1]

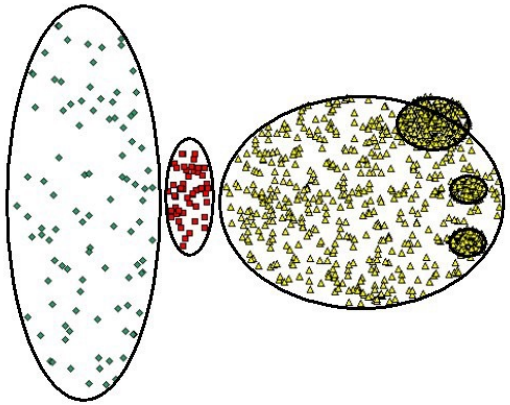
   Then find all points density-reachable from p and classify them in the cluster.

   [Reassign the zero labels but not the others]

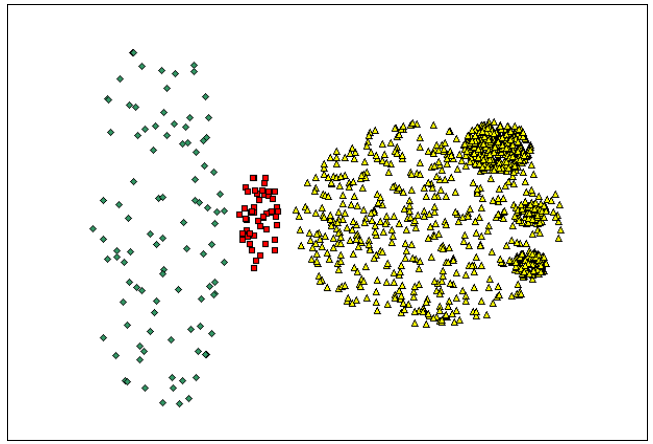
Repeat this process until all of the points have been visited.

Since all the zero labels of border points have been reassigned in 2, the remaining points with zero label are noise.

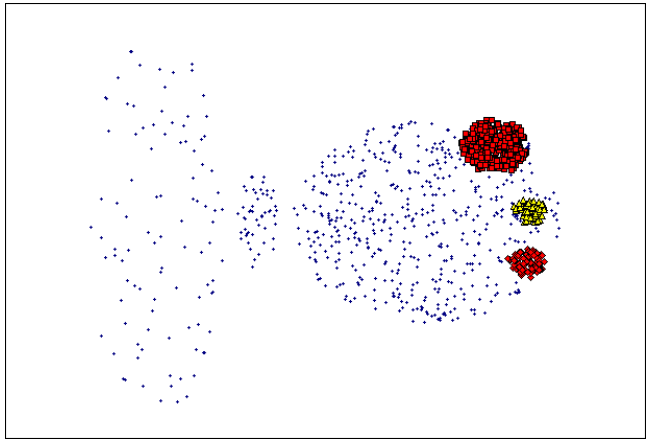
DBSCAN: Flaws



Original Points



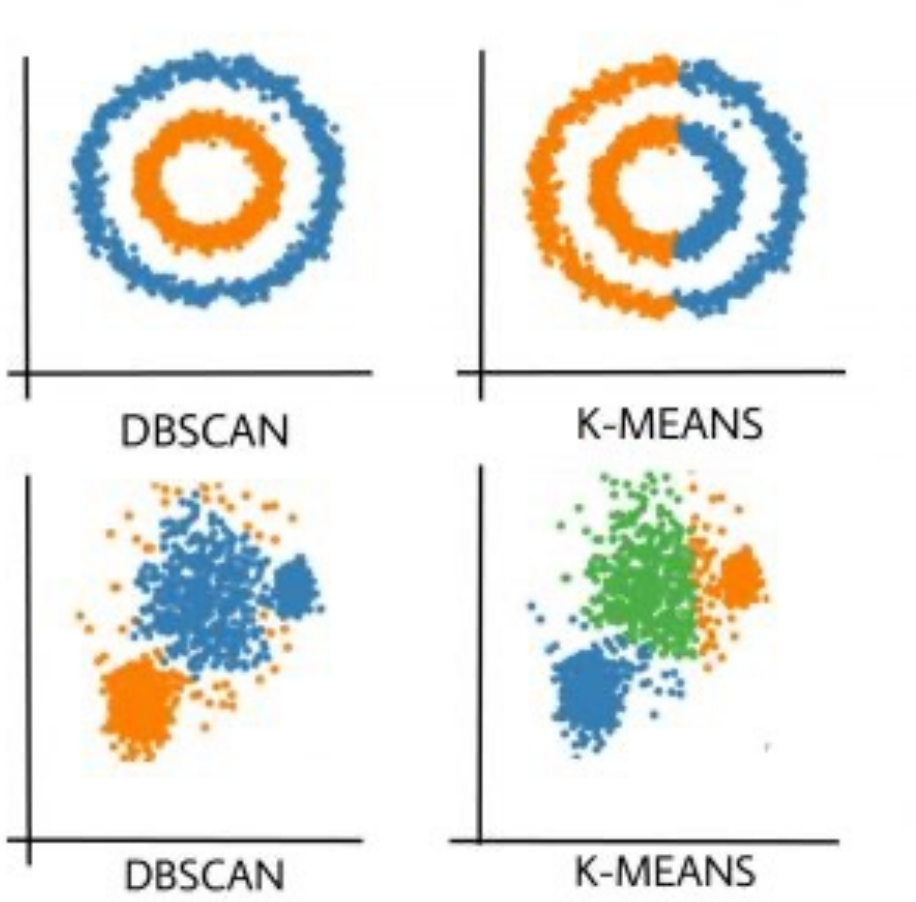
(MinPts=4, Eps=large value).



(MinPts=4, Eps=small value; min density increases)

- **Varying densities**
- **High-dimensional data**

K-Means and DBSCAN



K-Means and DBSCAN Clustering Algorithms

Aspect	K-Means	DBSCAN
Clustering Approach	Partition-based, assumes spherical clusters.	Density-based, identifies high-density regions.
Input Parameters	Requires number of clusters (k).	Requires ε (neighborhood radius) and MinPts.
Shape of Clusters	Works best for spherical or convex clusters.	Finds clusters of arbitrary shape.
Handling Noise	Sensitive to noise and outliers.	Handles noise effectively by marking as outliers.
Scalability	Efficient for large datasets; O(nkdi).	Less scalable for high-dimensional data; $O(n^2)$.
Sensitivity to Parameters	Requires specifying k; results depend on initialization.	Sensitive to ε and MinPts.
Clusters Overlap	Struggles with overlapping clusters.	Handles overlapping clusters better.
Assumption of Data	Assumes clusters are spherical and linearly separable.	No assumptions about shape or size of clusters.
Use Cases	Large datasets with compact, well-separated clusters.	Datasets with noise or arbitrary-shaped clusters.

Resources

- Pang-Ning Tan, Michael Steinbach, Vipin Kumar. Introduction to Data Mining. Michigan State University. University of Minnesota.

- Yaser S. Abu-Mostafa, Malik Maghdon-Ismail, and Hsuan Tien Lin,“Learning from Data”, 2012.

- C. Bishop, “Pattern Recognition and Machine Learning”.