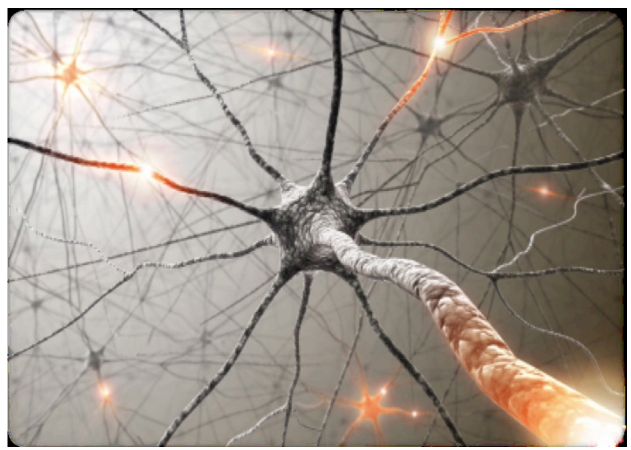
03 Unsupervised Learning
(无监督学习)



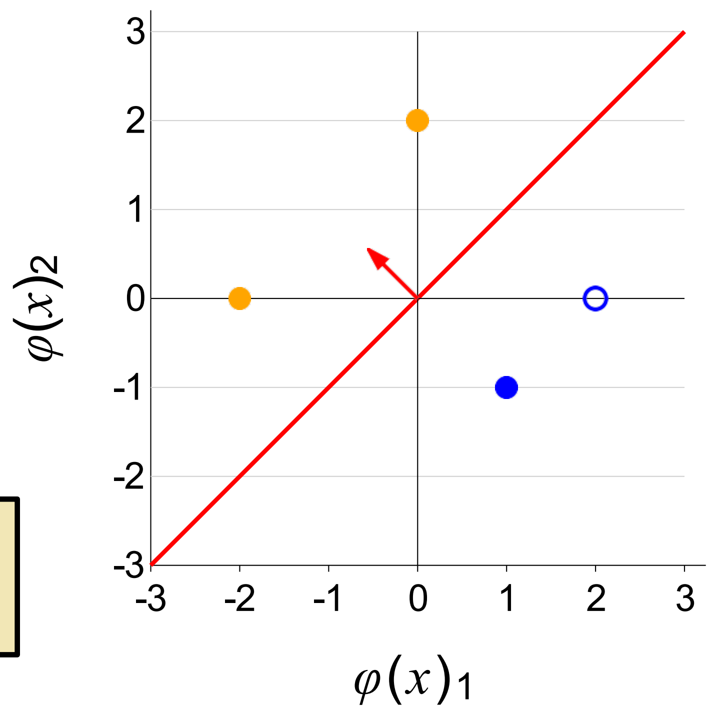
Classification (supervised learning)

training data D_{train}

$\varphi(x)_1$	$\varphi(x)_2$	y
0	2	1
-2	0	1
1	-1	-1

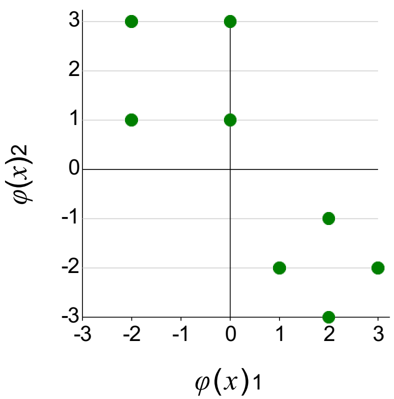
learning algorithm → f **classifier**

$[2, 0] \to f \to -1$



Labeled data is expensive to obtain

Clustering (unsupervised learning)



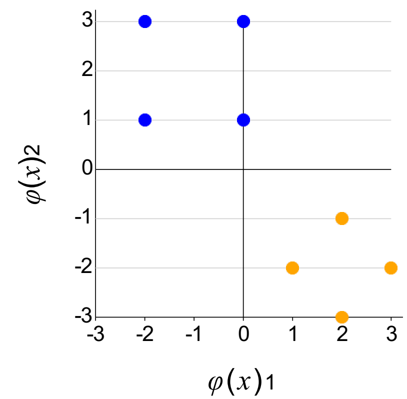
training data D_{train}

$\varphi(x)_1$	$\varphi(x)_2$
-2	1
0	1
-2	3
0	3
2	-1
1	-2
2	-3
3	-2

learning algorithm

assignments

z
1
1
1
1
2
2
2
2



Intuition: Want to assign nearby points to same cluster

Unlabeled data is very cheap to obtain

Supervision?

Supervised learning:

- Prediction: $\mathcal{D}_{\text{train}}$ contains input-output pairs (x, y)
- Fully-labeled data is very expensive to obtain (we can get 10,000 labeled examples)

Unsupervised learning:

- Clustering: $\mathcal{D}_{\text{train}}$ only contains inputs x
- Unlabeled data is much cheaper to obtain (we can get 100 Million unlabeled examples)

Word clustering

Input: raw text (100 million words of news articles)...

Output:

Cluster 1: Friday Monday Thursday Wednesday Tuesday Saturday Sunday weekends Sundays Saturdays
Cluster 2: June March July April January December October November September August
Cluster 3: water gas coal liquid acid sand carbon steam shale iron
Cluster 4: great big vast sudden mere sheer gigantic lifelong scant colossal
Cluster 5: man woman boy girl lawyer doctor guy farmer teacher citizen
Cluster 6: American Indian European Japanese German African Catholic Israeli Italian Arab
Cluster 7: pressure temperature permeability density porosity stress velocity viscosity gravity tension
Cluster 8: mother wife father son husband brother daughter sister boss uncle
Cluster 9: machine device controller processor CPU printer spindle subsystem compiler plotter
Cluster 10: John George James Bob Robert Paul William Jim David Mike
Cluster 11: anyone someone anybody somebody
Cluster 12: feet miles pounds degrees inches barrels tons acres meters bytes
Cluster 13: director chief professor commissioner commander treasurer founder superintendent dean custodian
Cluster 14: had hadn't hath would've could've should've must've might've
Cluster 15: head body hands eyes voice arm seat eye hair mouth

Impact: used in many state-of-the-art NLP systems

Feature learning using neural Networks

Input: 10 million images (sampled frames from YouTube)

Output:

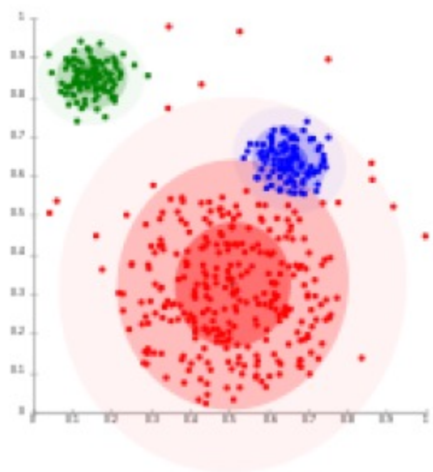
Impact: state-of-the-art results on object recognition (22,000 categories)

Key idea: the real learning objective

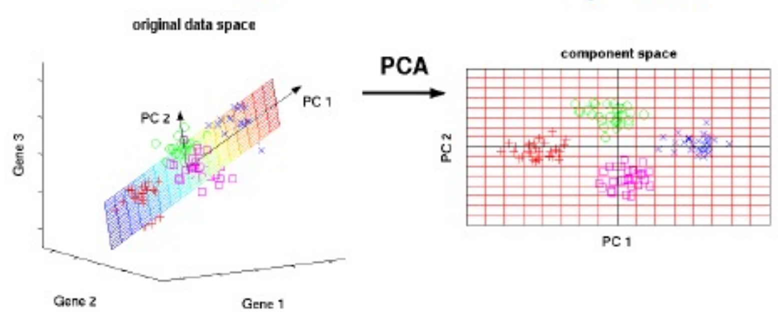
Data has lots of rich latent structures; want methods to discover this structure automatically.

Types of unsupervised learning

Clustering (e.g., K-means):



Dimensionality reduction (e.g., PCA):



Clustering

Definition: clustering

Input: training set of input points

$$\mathcal{D}_{\text{train}} = \{x_1, \dots, x_n\}$$

Output: assignment of each point to a cluster

$$[z_1, \dots, z_n] \text{ where } z_i \in \{1, \dots, K\}$$

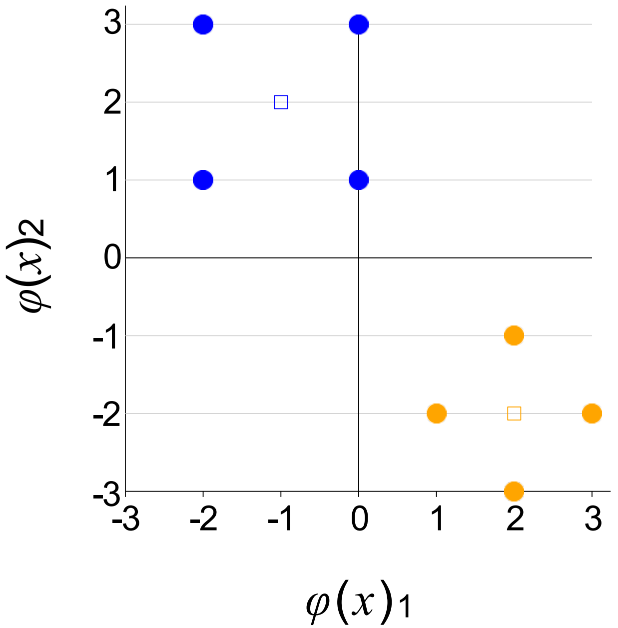
Intuition: want similar points to be in same cluster, dissimilar points to be in different clusters

[blackboard]

Centroids

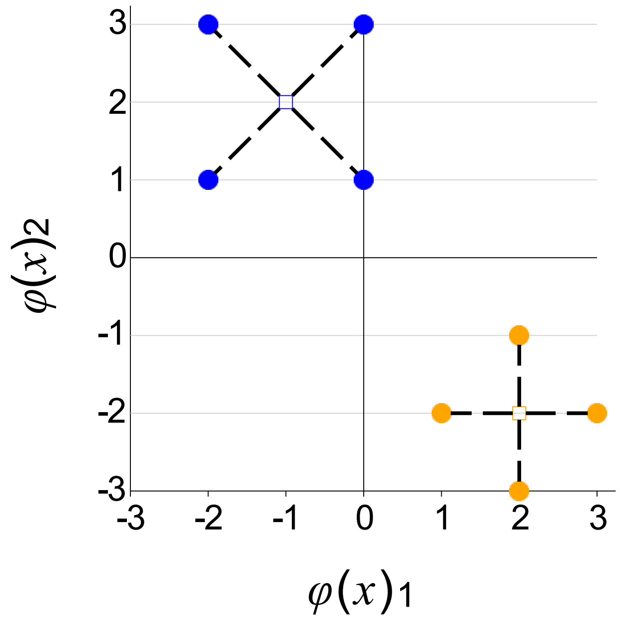
Each cluster $k = 1, \ldots, K$ is represented by a **centroid** $\mu_k \in \mathbb{R}^d$

$$\boldsymbol{\mu} = [\mu_1, \ldots, \mu_K]$$



Intuition: want each point $\varphi(x_i)$ to be close to its assigned centroid μ_{z_i}

K-means objective



$$\text{Loss}_{\text{kmeans}}(\mathbf{z}, \boldsymbol{\mu}) = \sum_{i=1}^{n} \|\phi(x_i) - \mu_{z_i}\|^2$$

$$\min_{\mathbf{z}} \min_{\boldsymbol{\mu}} \text{Loss}_{\text{kmeans}}(\mathbf{z}, \mu)$$

K-means objective

Setup:

- Each cluster k = 1,…,K is represented by a centroid

$$\mu_k \in \mathbb{R}^d$$

- Intuition: want each point close to its assigned centroid μ_{z_i}

Objective function:

$$\text{Loss}_{\text{kmeans}}(z, \mu) = \sum_{i=1}^{n} \|\phi(x_i) - \mu_{z_i}\|^2$$

Need to choose centroids μ and assignments z jointly

K-means: simple example

Example: one-dimensional

Input: $\mathcal{D}_{\text{train}} = \{0, 2, 10, 12\}$

Output: $K = 2$ centroids $\mu_1, \mu_2 \in \mathbb{R}$

If know centroids $\mu_1 = 1$, $\mu_2 = 11$:

$$z_1 = \arg\min\{(0-1)^2, (0-11)^2\} = 1$$
$$z_2 = \arg\min\{(2-1)^2, (2-11)^2\} = 1$$
$$z_3 = \arg\min\{(10-1)^2, (10-11)^2\} = 2$$
$$z_4 = \arg\min\{(12-1)^2, (12-11)^2\} = 2$$

If know assignments $z_1 = z_2 = 1$, $z_3 = z_4 = 2$:

$$\mu_1 = \arg\min_\mu (0-\mu)^2 + (2-\mu)^2 = 1$$
$$\mu_2 = \arg\min_\mu (10-\mu)^2 + (12-\mu)^2 = 11$$

K-means algorithm

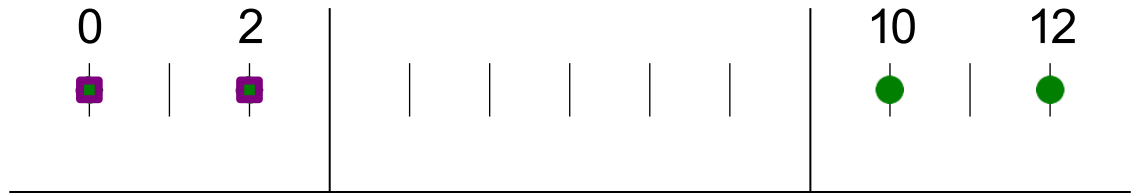
$$\min_{z} \min_{\mu} \mathrm{Loss}_{\mathrm{kmeans}}(z, \mu)$$

Key idea: alternating minimization

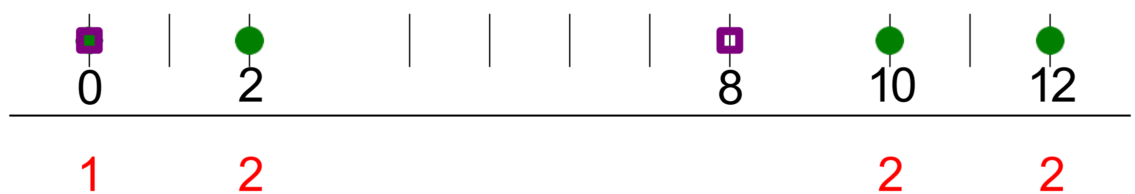
Tackle hard problem by solving two easy problems.

Alternating minimization from random initialization

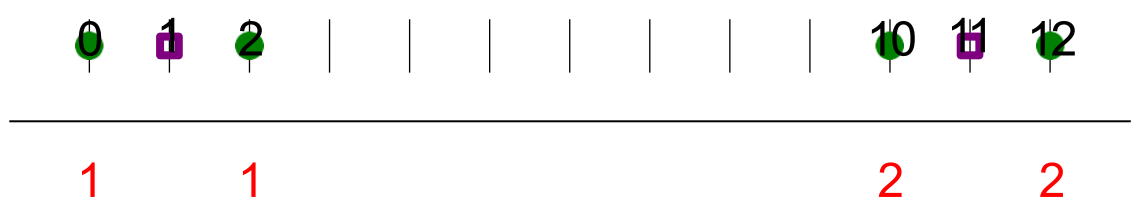
Initialize $\boldsymbol{\mu}$ (0, 2):



Iteration 1:



Iteration 2:



Converged.

K-means algorithm (Step 1)

- Goal: given centroids $\mu_1, \ldots, \mu_k$, assign each point to the best centroid.

Algorithm: Step 1 of K-means

For each point i = 1, …, n:

Assign i to cluster with closest centroid:

$$z_i \leftarrow \arg \min_{k=1,\ldots,K} \|\phi(x_i) - \mu_k\|^2$$

K-means algorithm (Step 2)

- Goal: given cluster assignments $z_1, \ldots, z_n$, find the best centroid $\mu_1, \ldots, \mu_k$.

Algorithm: Step 2 of K-means

For each cluster k = 1, ..., K:

set μ_k to the average of points assigned to cluster k:

$$\mu_k \to \frac{1}{|\{i : z_i = k\}|} \sum_{i : z_i = k} \phi(x_i)$$

K-means algorithm

Objective:

$$\min_{z} \min_{\mu} \text{Loss}_{\text{kmeans}}(z, \mu)$$

Algorithm: K-means

Initialize $\mu_1, \ldots, \mu_K$ randomly

For t = 1, … , T:

Step 1: set assignments z given μ

Step 2: set centroids μ given z

K-means: simple example

Example: one-dimensional

Input: $\mathcal{D}_{\text{train}} = \{0, 2, 10, 12\}$

Output: $K = 2$ centroids $\mu_1, \mu_2 \in \mathbb{R}$

Initialization (random): $\mu_1 = 0, \mu_2 = 2$

Iteration 1:

- Step 1: $z_1 = 1, z_2 = 2, z_3 = 2, z_4 = 2$
- Step 2: $\mu_1 = 0, \mu_2 = 8$

Iteration 2:

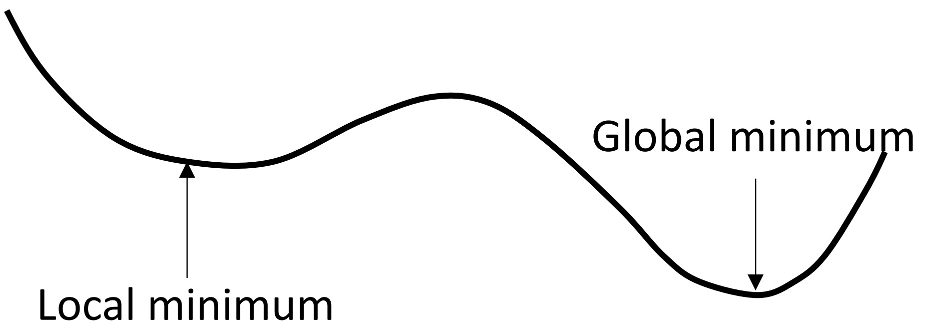
- Step 1: $z_1 = 1, z_2 = 1, z_3 = 2, z_4 = 2$
- Step 2: $\mu_1 = 1, \mu_2 = 11$

K-means: demo

https://www.naftaliharris.com/blog/visualizing-k-means-clustering/

Local minima

K-means is guaranteed to converge to a local minimum, but is not guaranteed to find the global minimum.



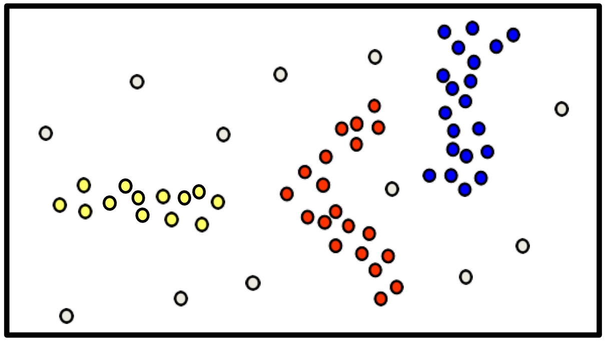
[demo]

Solutions：

- Run multiple times from different random initializations
- Initialize with a heuristic (K-means++)

Density-based Clustering

- **Basic idea**
  - Clusters are dense regions in the data space, separated by regions of lower object density
  - A cluster is defined as a maximal set (最大集合) of density- connected points
  - Discovers clusters of arbitrary shape
- **Method**
  - DBSCAN

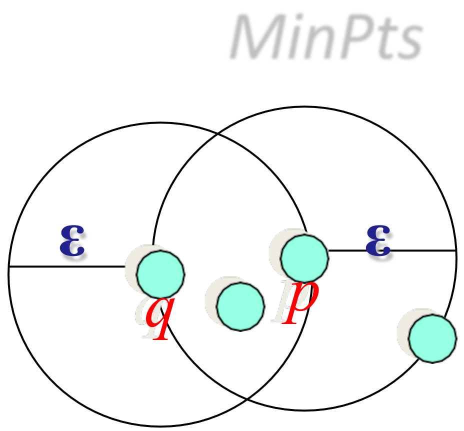


Density Definition

- ε-Neighborhood – Objects within a radius of ε from an object

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

- "High density" - ε-Neighborhood of an object contains at least *MinPts* of objects.



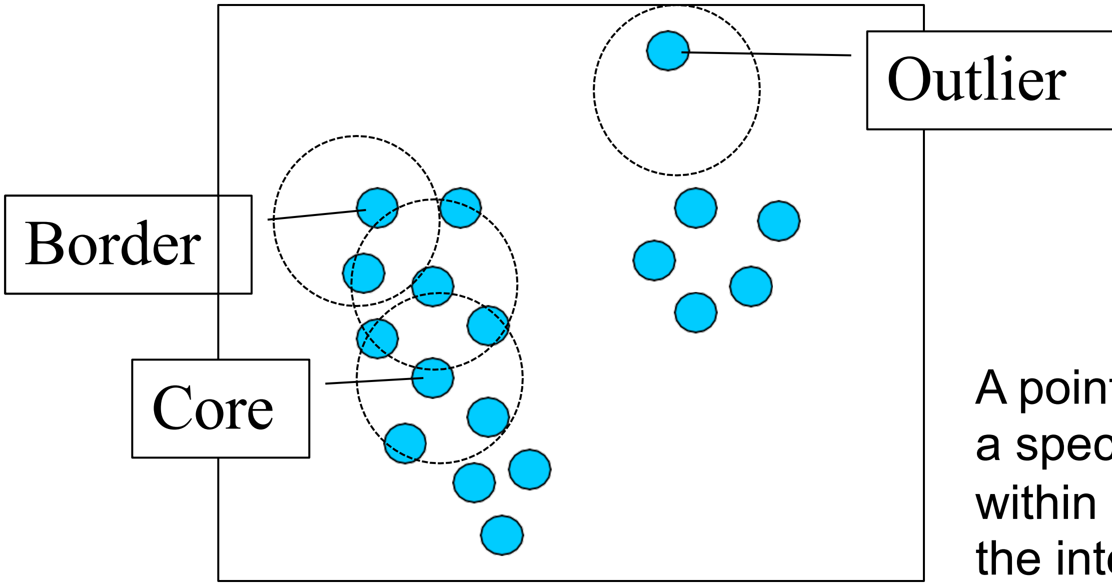
ε-Neighborhood of *p*

ε-Neighborhood of *q*

Density of p is "high" (MinPts = 4)

Density of q is "low" (MinPts = 3)

Core, Border & Outlier



ε= 1unit, MinPts = 5

Given ε and *MinPts*, categorize the objects into three exclusive groups.

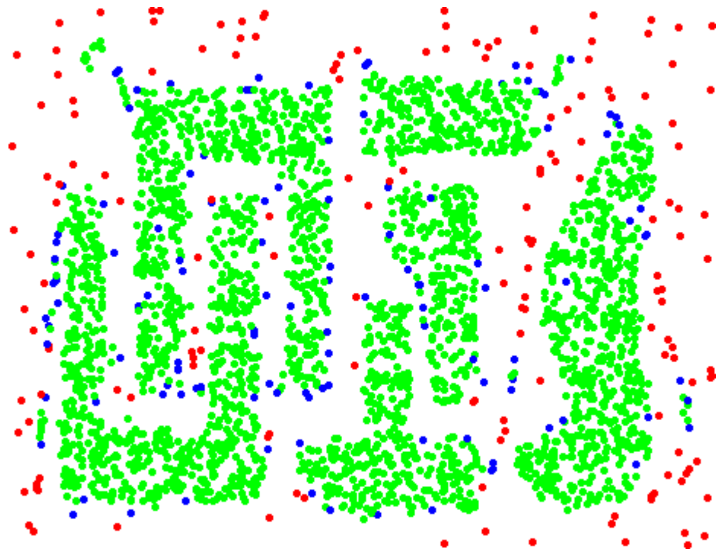
A point is a core point if it has more than a specified number of points (MinPts) within Eps—These are points that are at the interior of a cluster.

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 nor a border point.

Example

Original Points

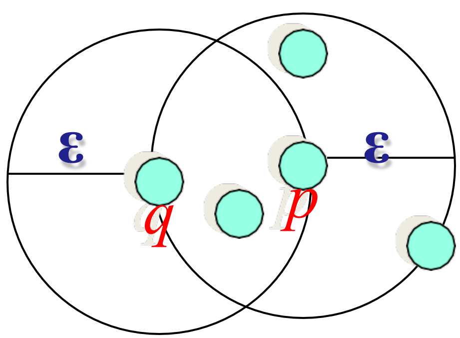


Point types: core, border and outliers

ε = 10, MinPts = 4

Density-reachability

- Directly density-reachable
  - An object *q* is directly density-reachable from object *p* if *p* is a core object and *q* is in p's ε-neighborhood.

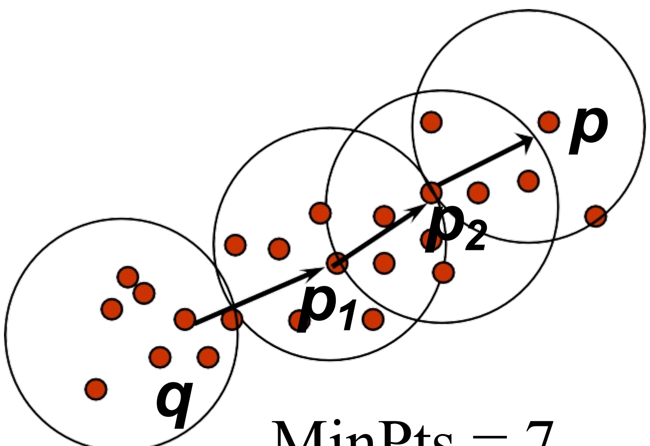


MinPts = 4

- *q* is directly density-reachable from *p*
- *p* is not directly density-reachable from *q*
- Density-reachability is asymmetric

Density-reachability

- Density-Reachable (directly and indirectly):
  - A point p is directly density-reachable from p_2
  - p_2 is directly density-reachable from p_1
  - p_1 is directly density-reachable from q
  - $p \leftarrow p_2 \leftarrow p_1 \leftarrow q$ form a chain

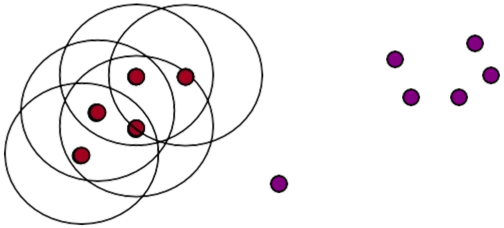


MinPts = 7

- p is (indirectly) density-reachable from q
- q is not density-reachable from p

DBSCAN Algorithm: Example

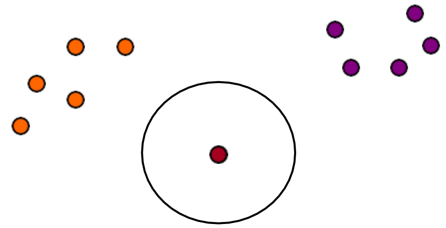
- **Parameter**
  - ε = 2 cm
  - *MinPts* = 3



for each $o \in D$ **do**
  if o is not yet classified **then**
    if o is a core-object **then**
      collect all objects density-reachable from o
      and assign them to a new cluster.
    else
      assign o to NOISE

DBSCAN Algorithm: Example

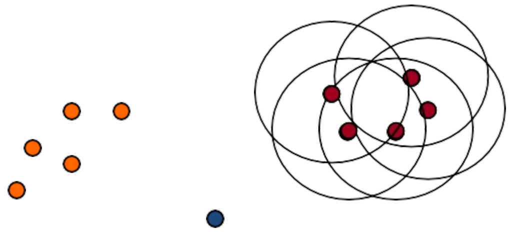
- **Parameter**
  - ε = 2 cm
  - *MinPts* = 3



for each $o \in D$ **do**
    if o is not yet classified **then**
        if o is a core-object **then**
            collect all objects density-reachable from o
            and assign them to a new cluster.
        else
            assign o to NOISE

DBSCAN Algorithm: Example

- **Parameter**
  - ε= 2 cm
  - *MinPts* = 3



for each $o \in D$ **do**
    if o is not yet classified **then**
        if o is a core-object **then**
            collect all objects density-reachable from o
            and assign them to a new cluster.
        else
            assign o to NOISE

DBSCAN: Sensitive to Parameters

Figure 8. DBScan results for DS1 with MinPts at 4 and Eps at (a) 0.5 and (b) 0.4.

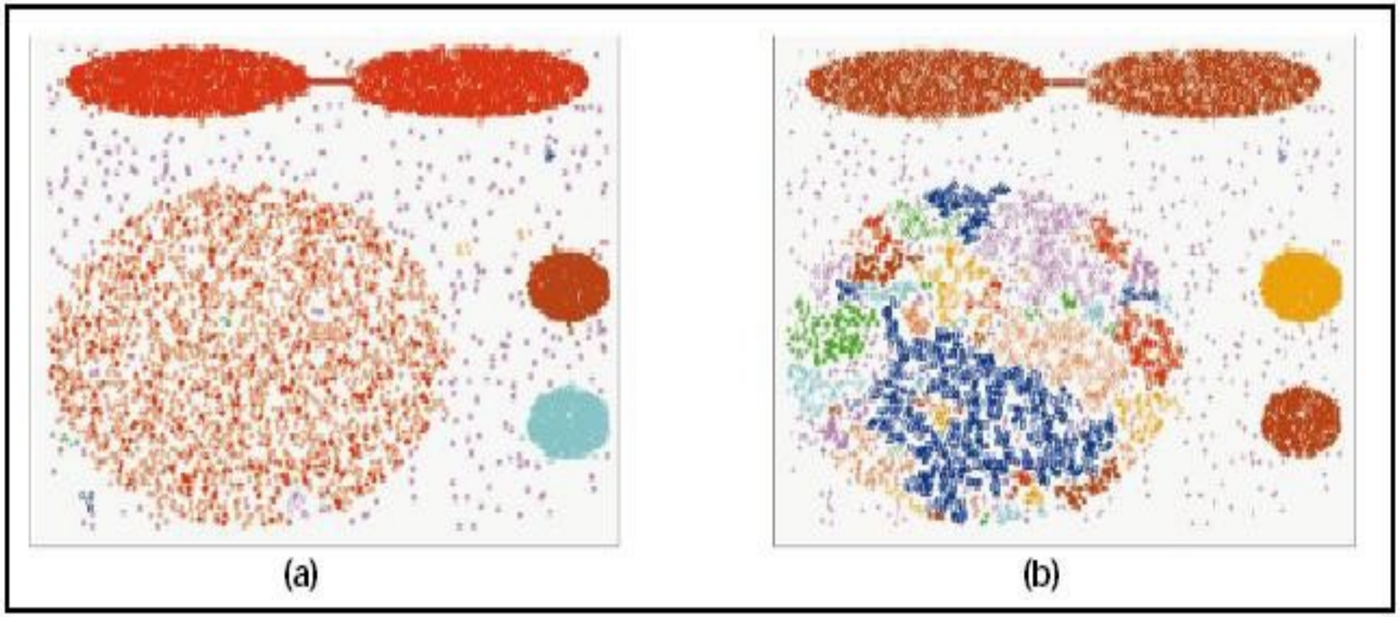
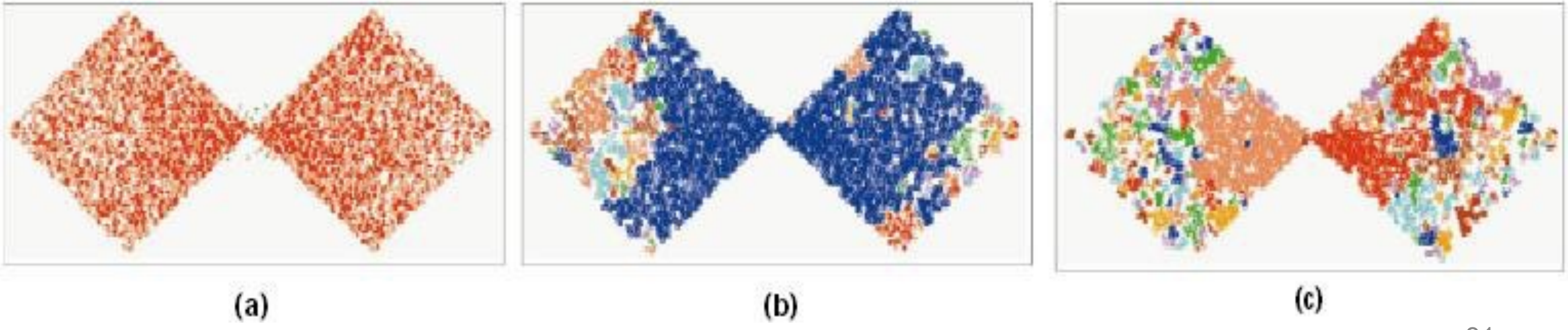


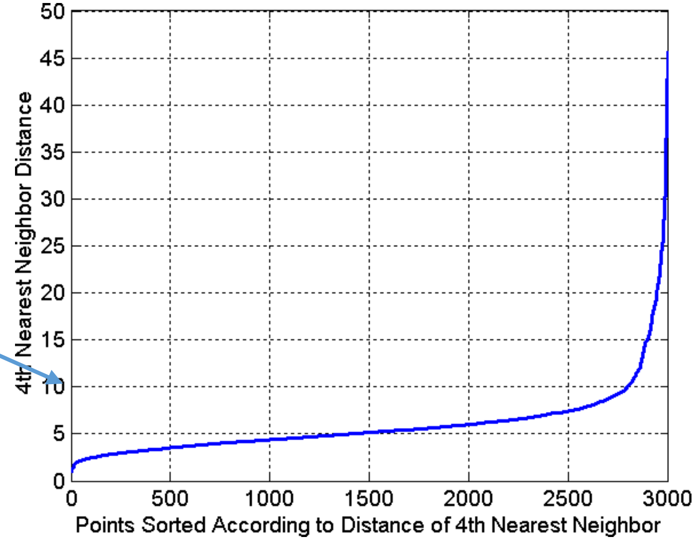
Figure 9. DBScan results for DS2 with MinPts at 4 and Eps at (a) 5.0, (b) 3.5, and (c) 3.0.



DBSCAN: Determining EPS and MinPts

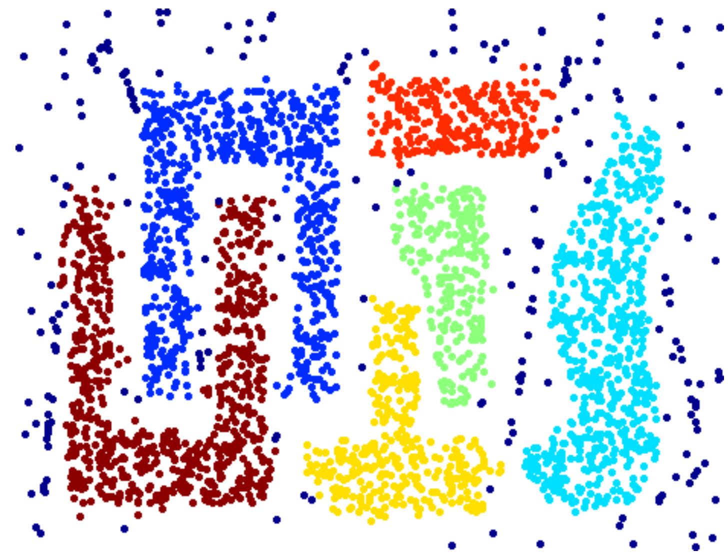
- Idea is that for points in a cluster, their k^{th} nearest neighbors are at roughly the same distance
- Noise points have the k^{th} nearest neighbor at farther distance
- So, plot sorted distance of every point to its k^{th} nearest neighbor (for example, k=4 as blew)

Thus, Eps =10



When DBSCAN Works Well

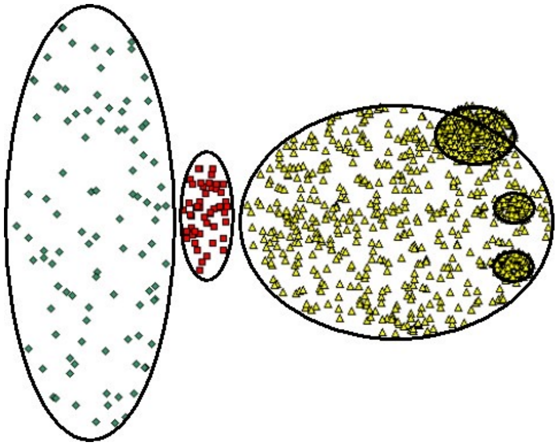
Original Points



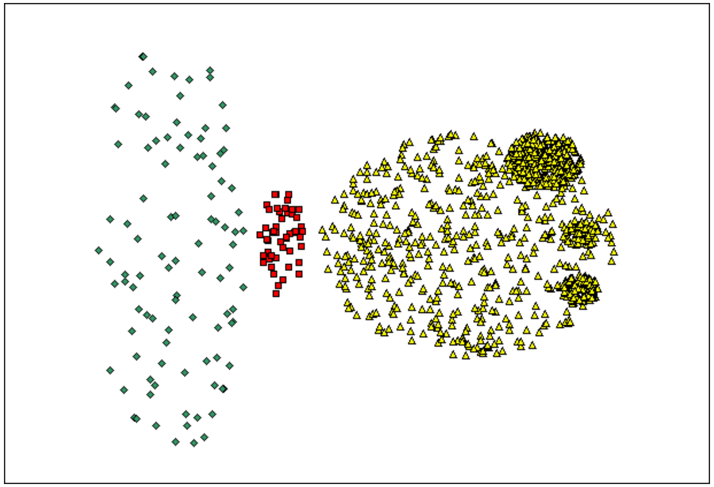
Clusters

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

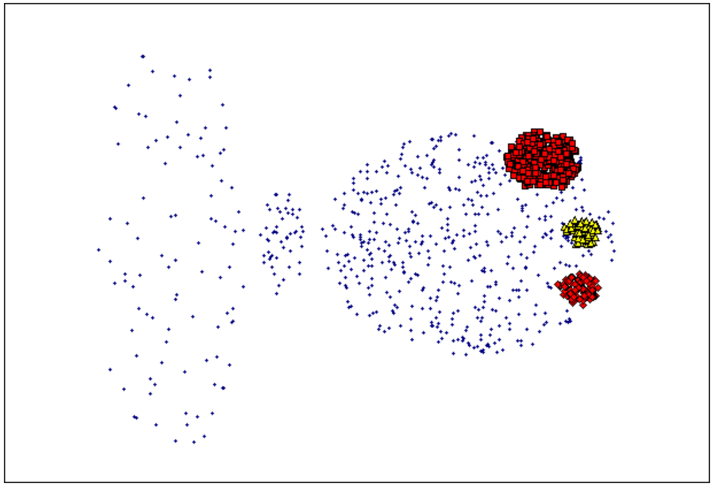
When DBSCAN Does NOT Work Well



Original Points



(MinPts=4, Eps=9.92).



(MinPts=4, Eps=9.75)

- **Cannot handle varying densities**
- **sensitive to parameters—hard to determine the correct set of parameters**

Summary of DBSCAN

- The basic idea of density-based clustering
- The two important parameters and the definitions of neighborhood and density in DBSCAN
- Core, border and outlier points
- DBSCAN algorithm
- DBSCAN's pros and cons

Unsupervised learning summary

- Leverage tons of unlabeled data

- Difficult optimization:

latent variables

z  parameters

μ

Unsupervised learning use cases:

- Data exploration and discovery
- Providing representations to downstream supervised learning

Summary so far

- Feature extraction (think hypothesis classes) [modeling]
- Prediction (linear, neural network, k-means) [modeling]
- Loss functions (compute gradients) [modeling]
- Optimization (stochastic gradient, alternating minimization) [algorithms]
- Generalization (think development cycle) [modeling]

Challenges

Capabilities:

- More complex prediction problems (translation, generation)
- Unsupervised learning: automatically discover structure

Responsibilities:

- Feedback loops: predictions affect user behavior, which generates data
- Fairness: build classifiers that don't discriminate?
- Privacy: can we pool data together
- Interpretability: can we understand what algorithms are doing?

Acknowledgement

Reference and thanks to:

- **Stanford University CS221 Course:**

Artificial Intelligence: Principles and Techniques

https://stanford-cs221.github.io/autumn2022/

- **CMU 10-701/15-781 Course:**

Machine Learning

https://www.cs.cmu.edu/~aarti/Class/10701/

- **SUNY Buffalo CSE601 Course:**

Data Mining and Bioinformatics

https://cse.buffalo.edu/~jing/cse601