Machine Learning 2

DS 4420 - Spring 2020

Clustering I

Byron C. Wallace

Unsupervised learning

- So far we have reviewed some fundamentals, discussed Maximum Likelihood Estimation (MLE) for probabilistic models, and neural networks/backprop SGD

Unsupervised learning

- So far we have reviewed some fundamentals, discussed Maximum Likelihood Estimation (MLE) for probabilistic models, and neural networks/backprop SGD
- We have mostly considered *supervised* settings (implicitly) although the above methods are general; we will shift focus to *unsupervised* learning for a few weeks

Unsupervised learning

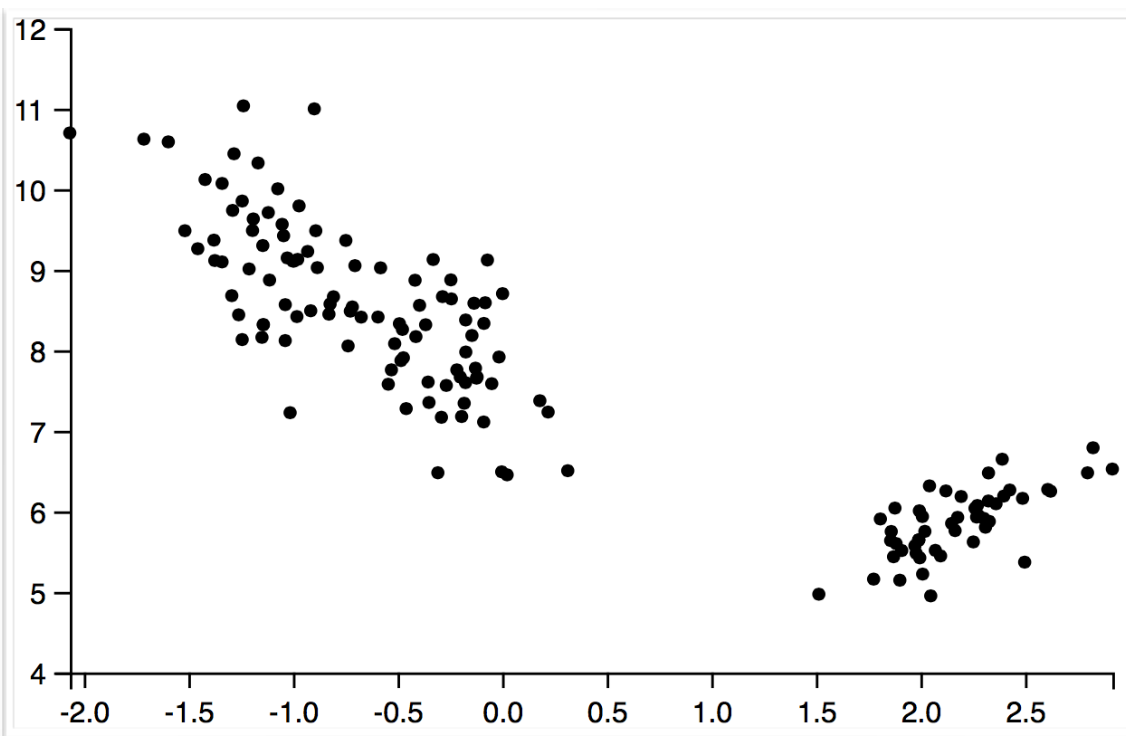
- So far we have reviewed some fundamentals, discussed Maximum Likelihood Estimation (MLE) for probabilistic models, and neural networks/backprop SGD

- We have mostly considered *supervised* settings (implicitly) although the above methods are general; we will shift focus to *unsupervised* learning for a few weeks

- Both the probabilistic and neural perspectives will continue to be relevant here — and we will consider the former explicitly for clustering next week

Clustering

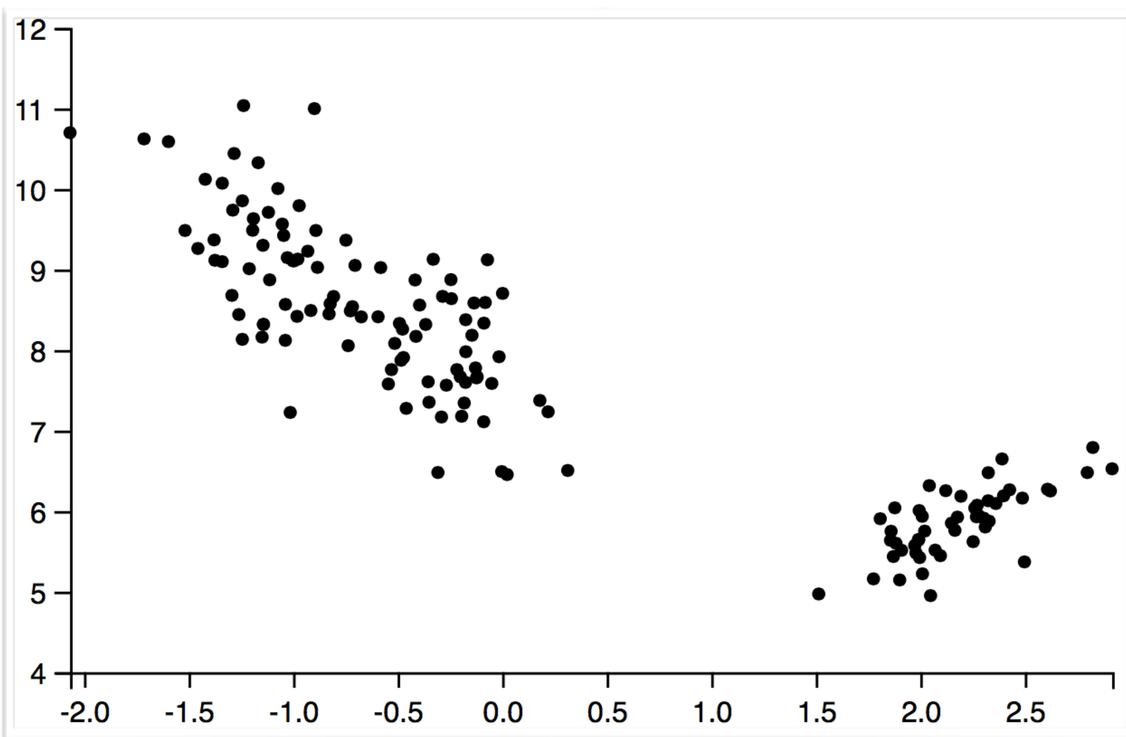
Clustering

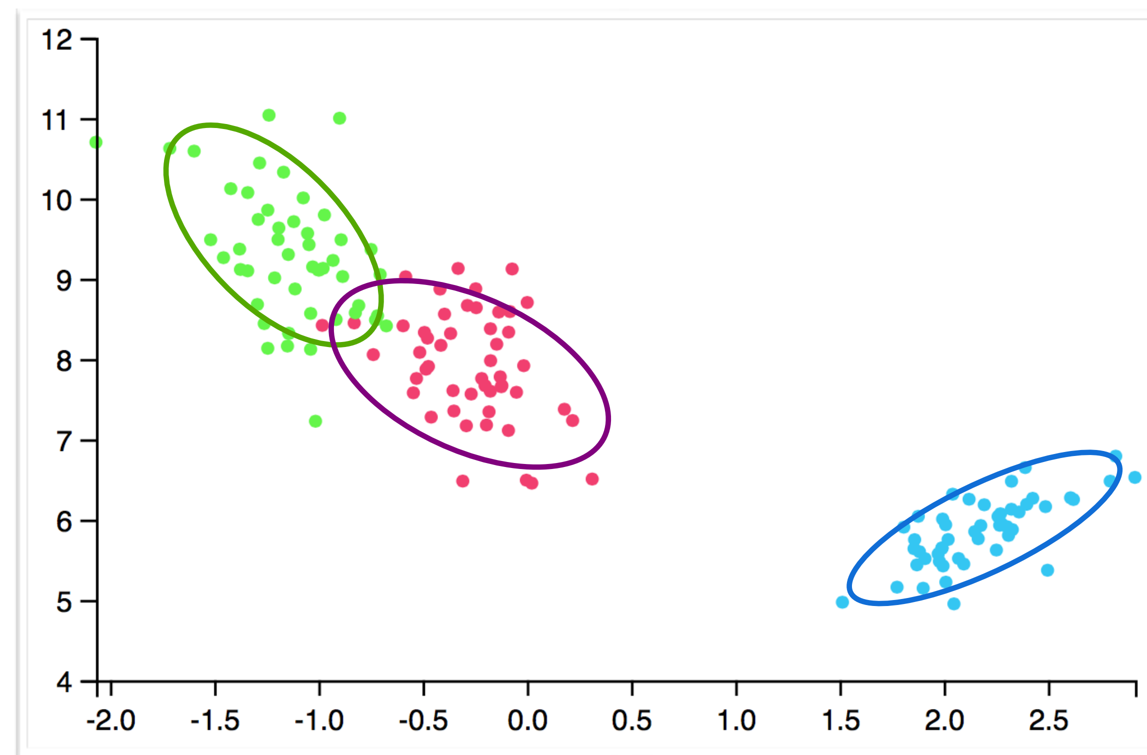


Unsupervised learning (no labels for training)
Group data into similar classes that

- Maximize *inter-cluster* similarity
- Minimize *intra-cluster* similarity

Clustering

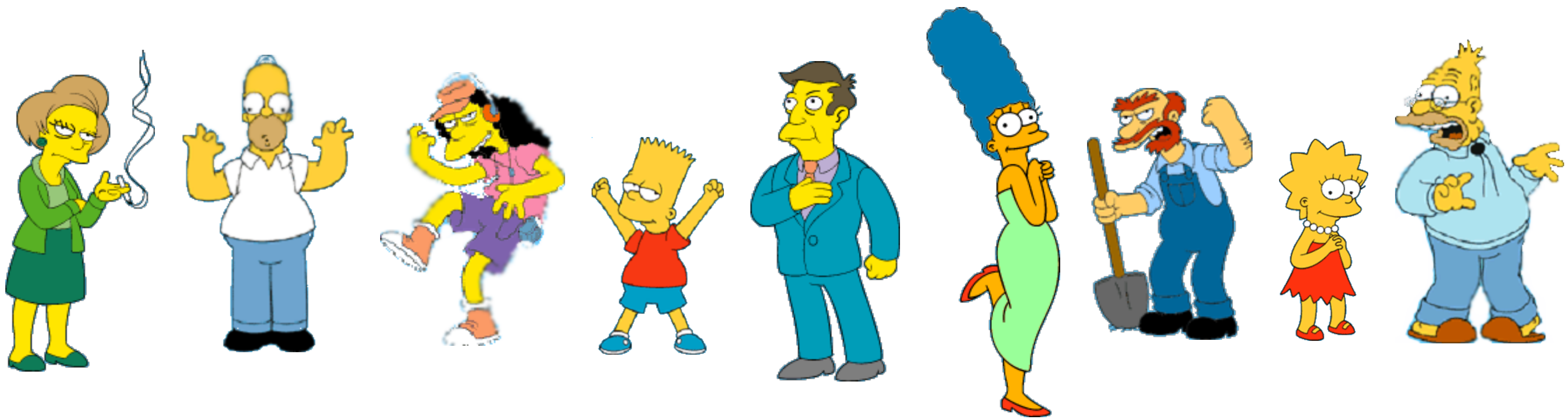




Unsupervised learning (no labels for training)
Group data into similar classes that

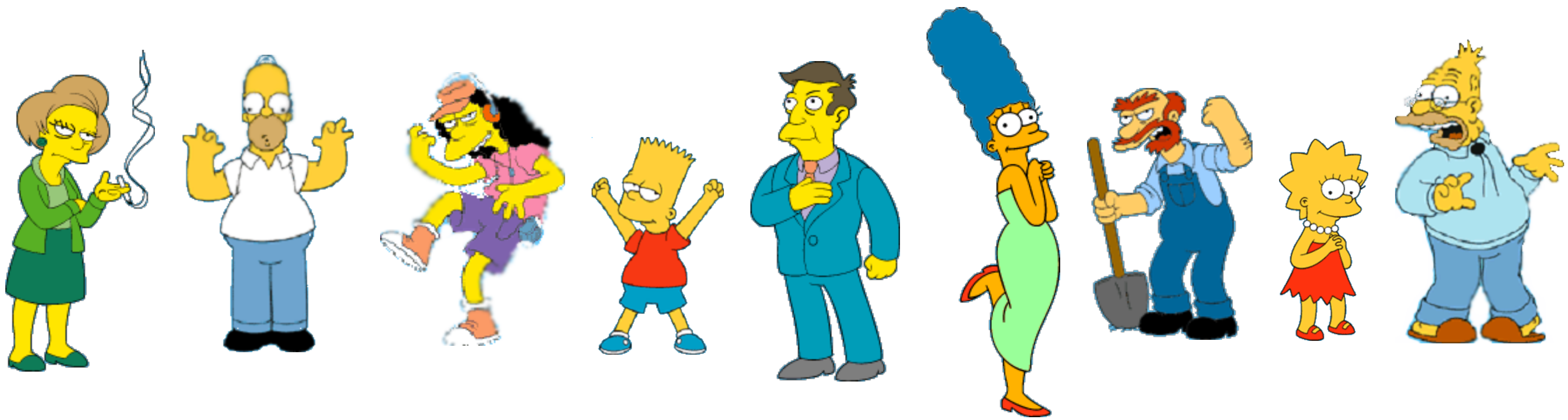
- Maximize *inter-cluster* similarity
- Minimize *intra-cluster* similarity

What is a natural grouping?

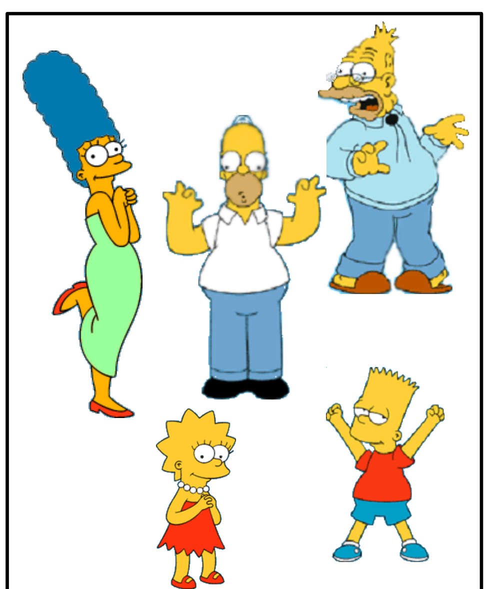


Choice of clustering criterion can be task-dependent

What is a natural grouping?



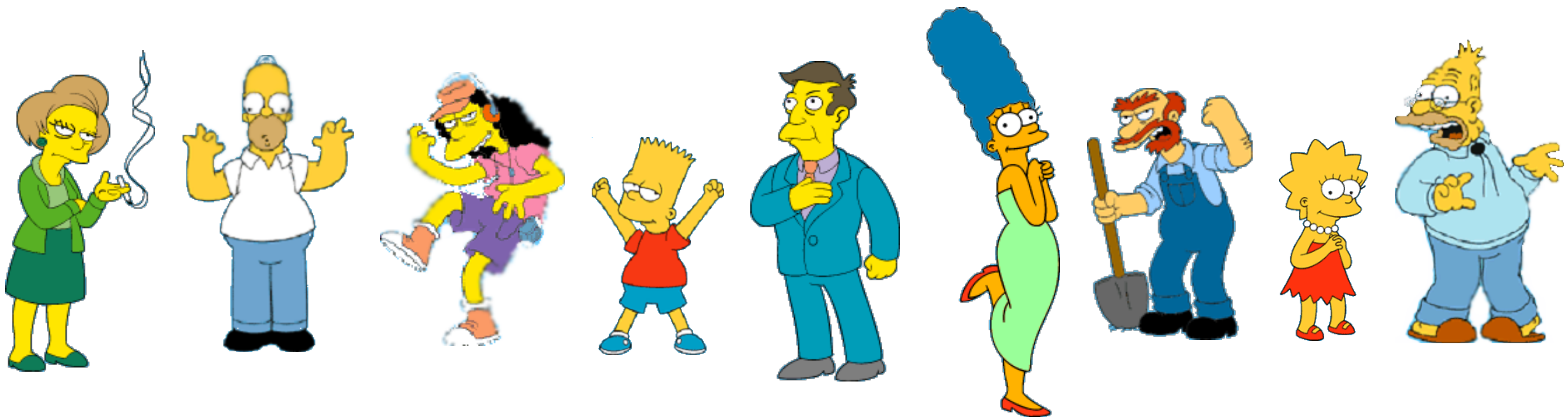
Choice of clustering criterion can be task-dependent



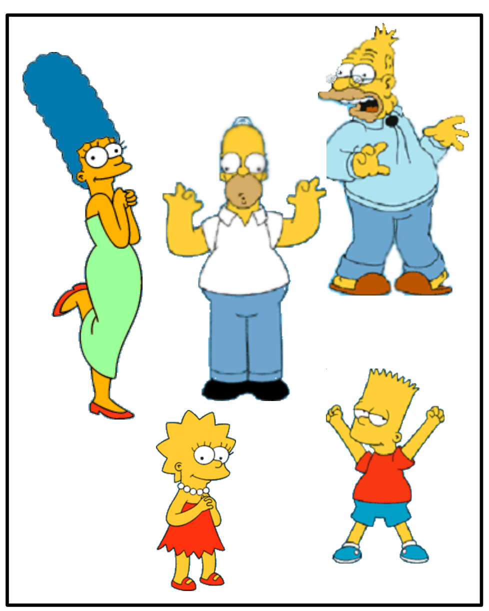
Simpson's Family

School Employees

What is a natural grouping?

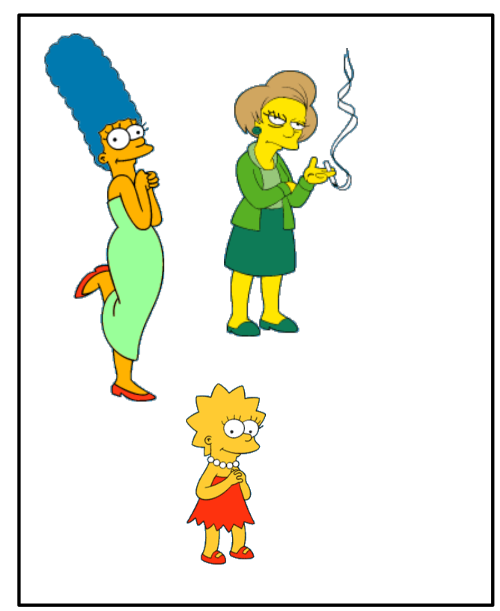


Choice of clustering criterion can be task-dependent

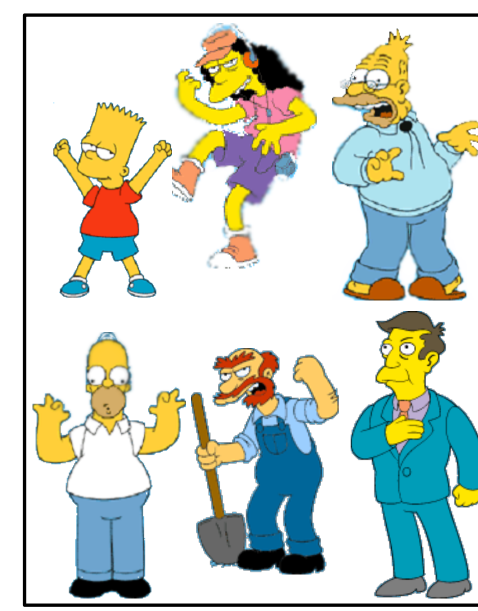


Simpson's Family

School Employees

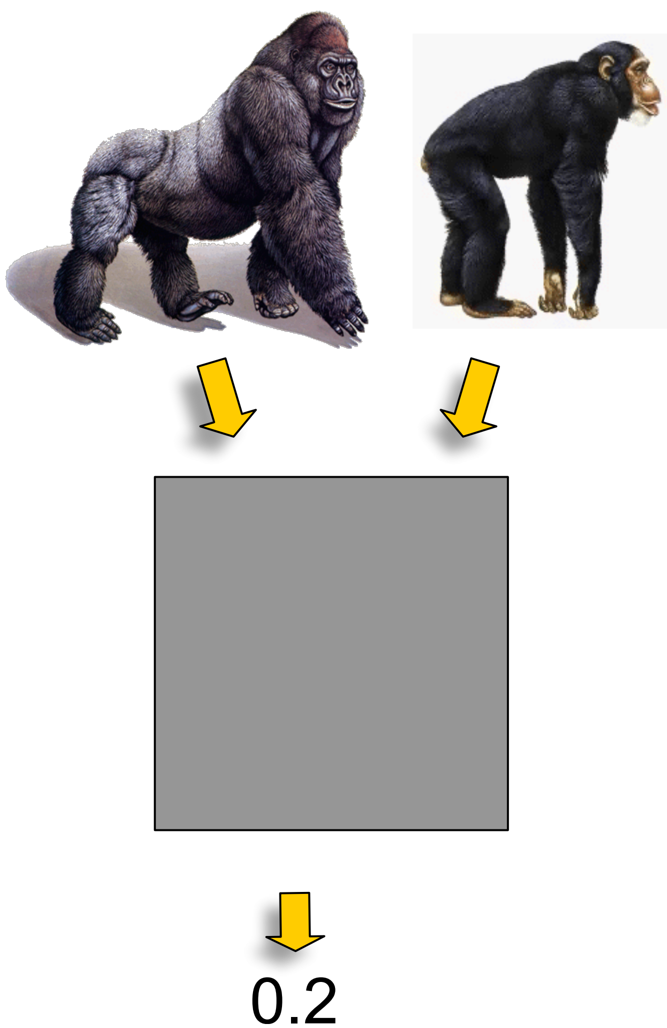


Females



Males

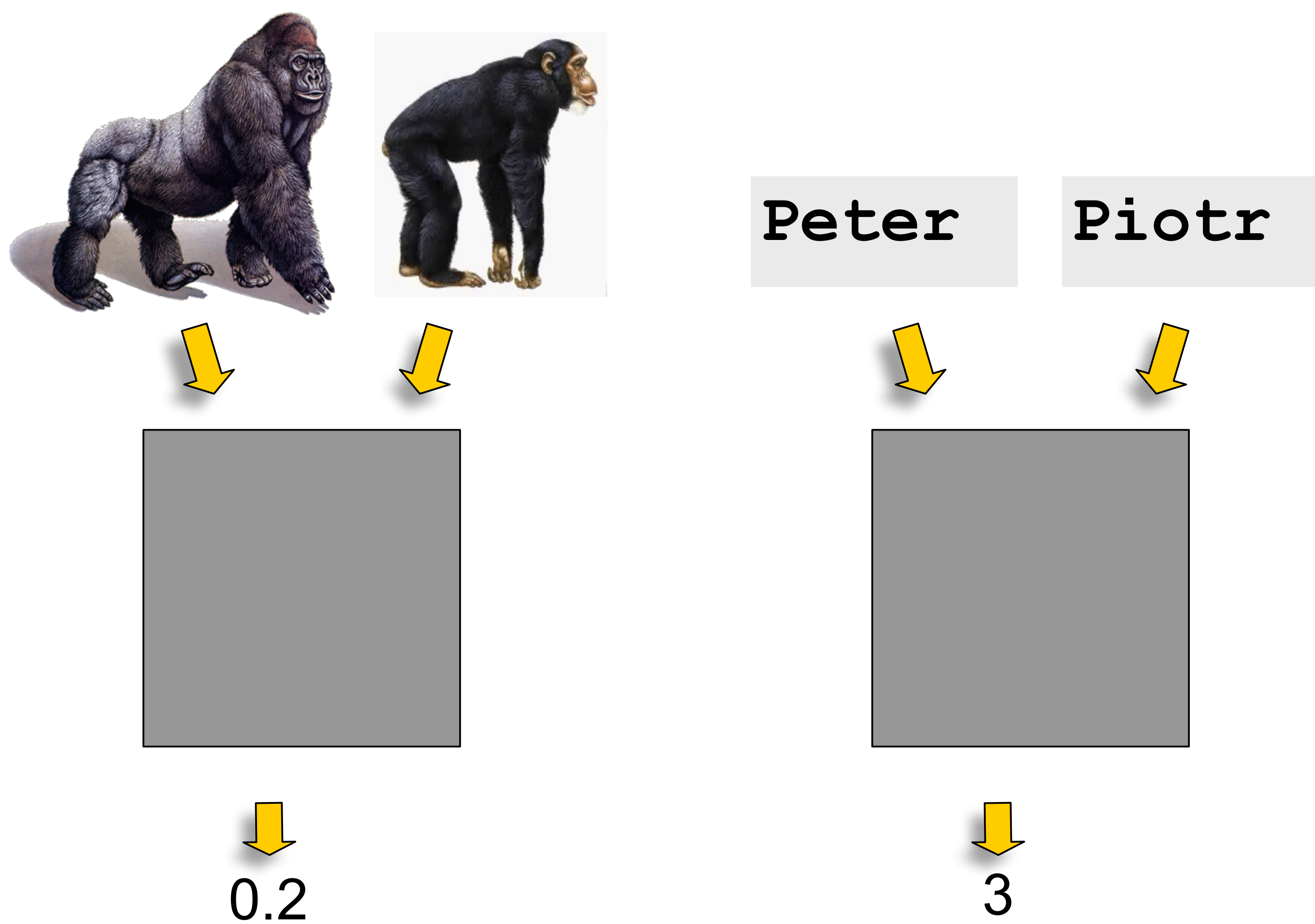
Defining Distance Measures



Dissimilarity/distance: $d(\boldsymbol{x}_1, \boldsymbol{x}_2)$
Similarity: $s(\boldsymbol{x}_1, \boldsymbol{x}_2)$

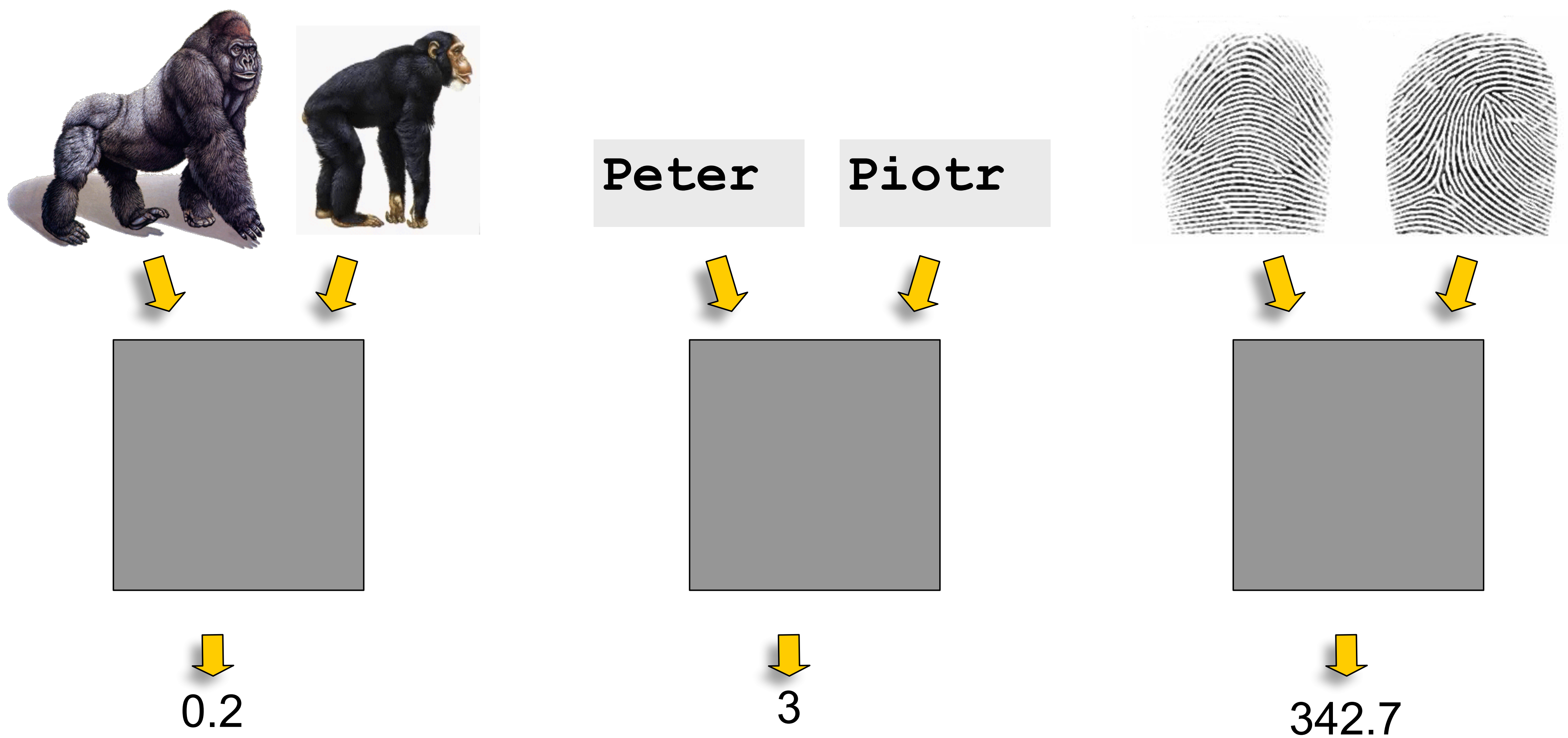
Proximity: $p(\boldsymbol{x}_1, \boldsymbol{x}_2)$

Defining Distance Measures



$$\left.\begin{aligned}\text{Dissimilarity/distance:}\quad & d(\boldsymbol{x}_1, \boldsymbol{x}_2)\\ \text{Similarity:}\quad & s(\boldsymbol{x}_1, \boldsymbol{x}_2)\end{aligned}\right\}\quad \text{Proximity:}\quad p(\boldsymbol{x}_1, \boldsymbol{x}_2)$$

Defining Distance Measures



Dissimilarity/distance: $d(\boldsymbol{x}_1, \boldsymbol{x}_2)$

Similarity: $s(\boldsymbol{x}_1, \boldsymbol{x}_2)$

Proximity: $p(\boldsymbol{x}_1, \boldsymbol{x}_2)$

Distance Measures

- Euclidean Distance $$\sqrt{\left(\sum_{i=1}^{k}(x_i - y_i)^2\right)}$$

Distance Measures

- Euclidean Distance $$\sqrt{(\sum_{i=1}^{k}(x_i - y_i)^2)}$$
- Mahattan Distance $$\sum_{i=1}^{k}|x_i - y_i|$$

Distance Measures

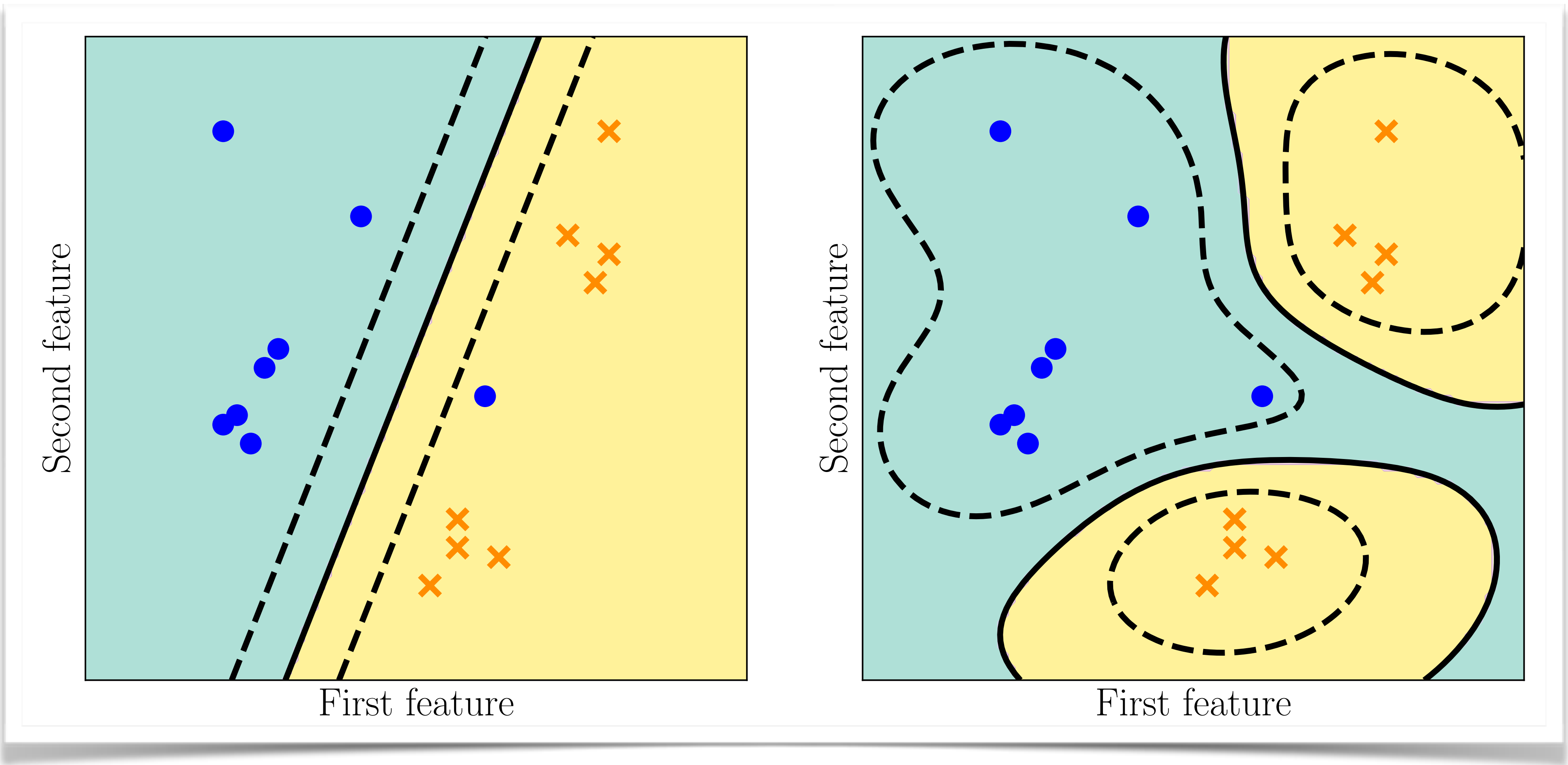
- Euclidean Distance $$\sqrt{(\sum_{i=1}^{k}(x_i - y_i)^2)}$$
- Mahattan Distance $$\sum_{i=1}^{k}|x_i - y_i|$$
- Minkowski Distance $$\left(\sum_{i=1}^{k}(|x_i - y_i|)^q\right)^{\frac{1}{q}}$$

Similarity over *functions of inputs*

- The preceding measures are distances defined on the original input space X
- A better representation may be some function of these features $\boldsymbol{\phi}(\boldsymbol{x})$

Similarity: *Kernels*

Linear (inner-product)	$k(\boldsymbol{x}, \boldsymbol{x}') = (\langle \boldsymbol{x}, \boldsymbol{x}' \rangle + c)$
Polynomial	$k(\boldsymbol{x}, \boldsymbol{x}') = (\langle \boldsymbol{x}, \boldsymbol{x}' \rangle + c)^m$
Radial Basis Function (RBF)	$k(\boldsymbol{x}, \boldsymbol{x}') = \exp^{-\frac{1}{2}\gamma^{-2}\|\|\boldsymbol{x}-\boldsymbol{x}'\|\|^2}$



Linear RBF kernel

Figure from MML book

Why kernels?

“The key insight in kernel-based learning is that you can rewrite many linear models in a way that doesn’t require you to ever explicitly compute $\phi(x)$

- Daume, CIML

Similarities vs Distance Measure

Distance Measure

- D(A, B) = D(B, A) *Symmetry*

Similarities vs Distance Measure

Distance Measure

- $D(A, B) = D(B, A)$ *Symmetry*
- $D(A, A) \geq 0$ *Reflexivity*

Similarities vs Distance Measure

Distance Measure

• $D(A, B) = D(B, A)$	*Symmetry*
• $D(A, A) \geq 0$	*Reflexivity*
• $D(A, B) = 0$ iff $A = B$	*Positivity (Separation)*

Similarities vs Distance Measure

Distance Measure

• $D(A, B) = D(B, A)$	*Symmetry*
• $D(A, A) \geq 0$	*Reflexivity*
• $D(A, B) = 0$ iff $A = B$	*Positivity (Separation)*
• $D(A, B) \leq D(A, C) + D(B, C)$	*Triangular Inequality*

Similarities vs Distance Measure

Distance Measure

• D(A, B) = D(B, A)	*Symmetry*
• D(A, A) ≥ 0	*Reflexivity*
• D(A, B) = 0 iff A = B	*Positivity (Separation)*
• D(A, B) ≤ D(A, C) + D(B, C)	*Triangular Inequality*

Similarity functions

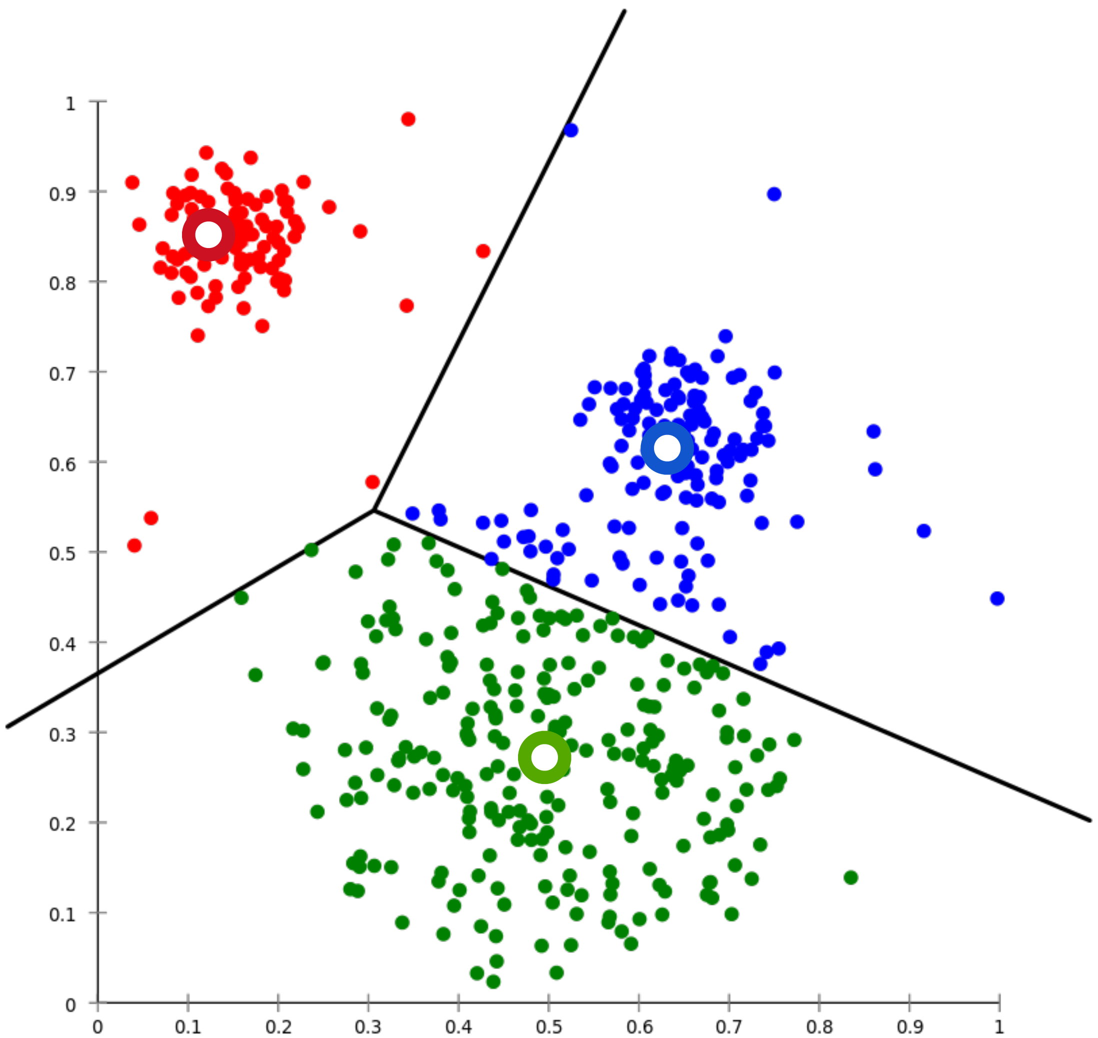
- *Less formal; encodes some notion of similarity but not necessarily well defined*
- *Can be negative*
- *May not satisfy triangular inequality*

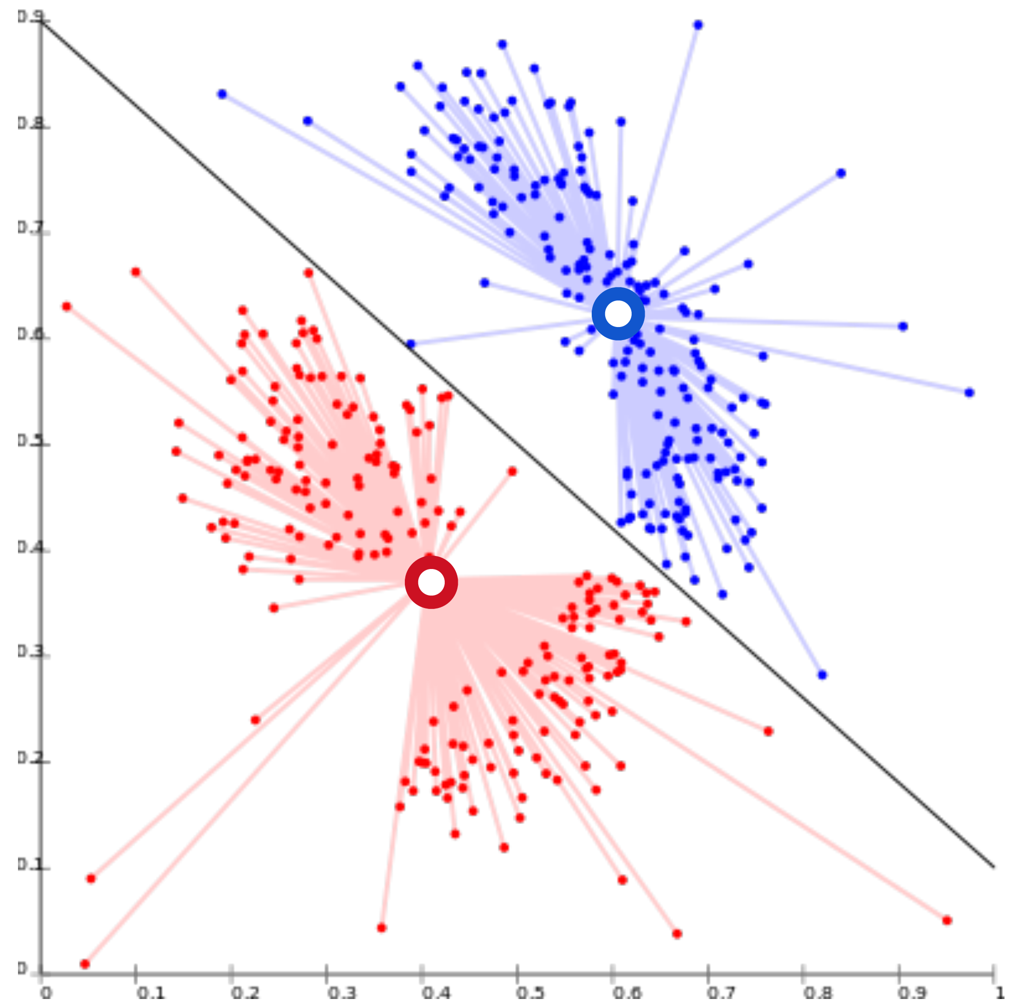
Cosine similarity

$$similarity(A,B) = \frac{A \cdot B}{\|A\| \times \|B\|} = \frac{\sum_{i=1}^{n} A_i \times B_i}{\sqrt{\sum_{i=1}^{n} A_i^2} \times \sqrt{\sum_{i=1}^{n} B_i^2}}$$

Four Types of Clustering

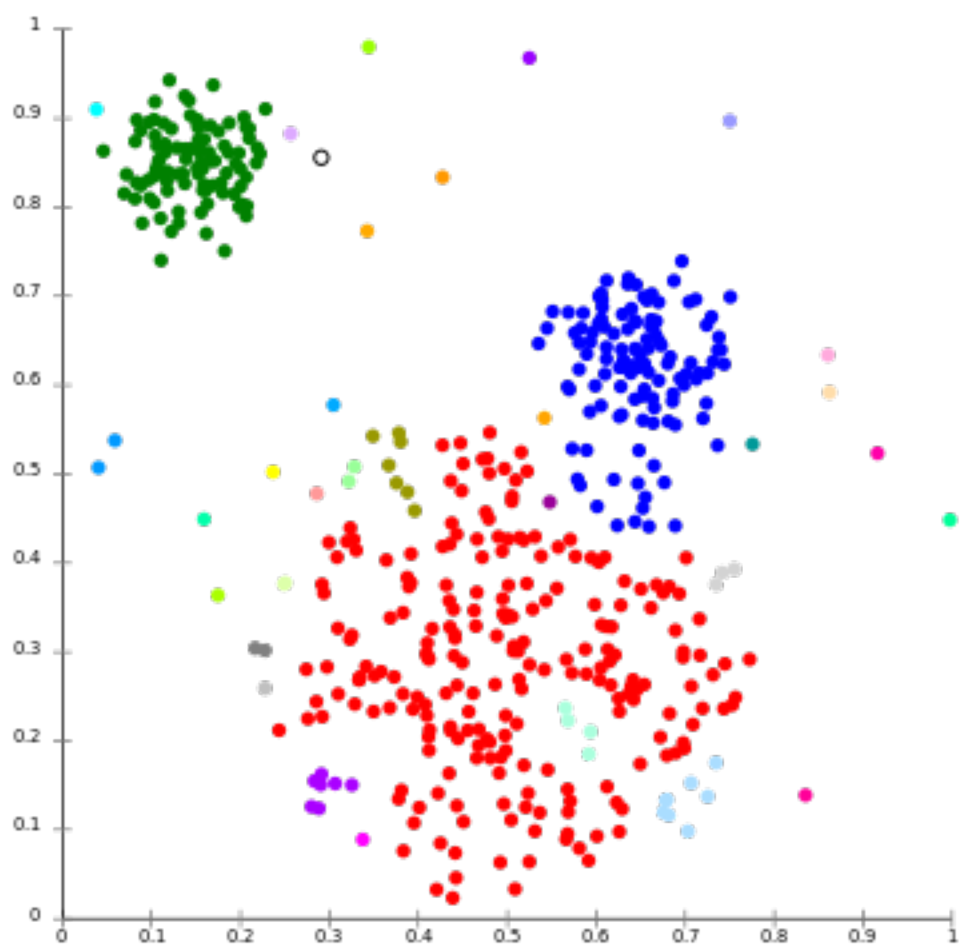
1. *Centroid-based (K-means, K-medoids)*

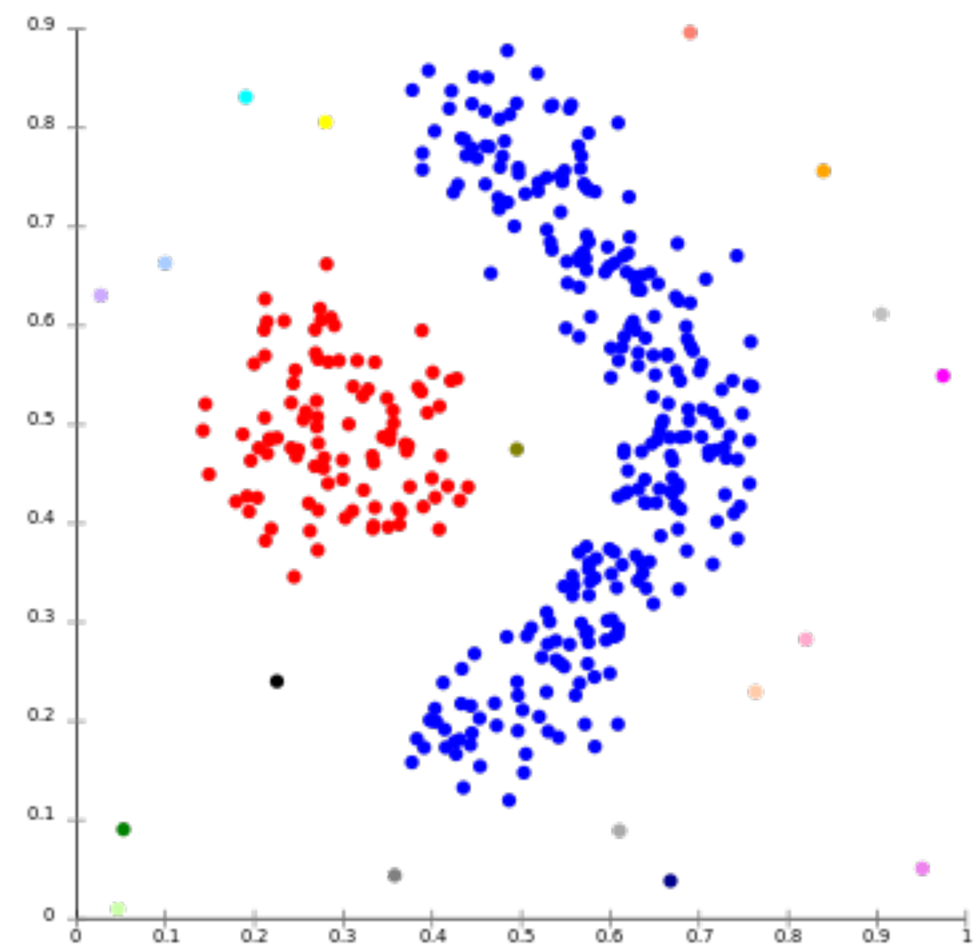




Four Types of Clustering

2. *Connectivity-based (Hierarchical)*

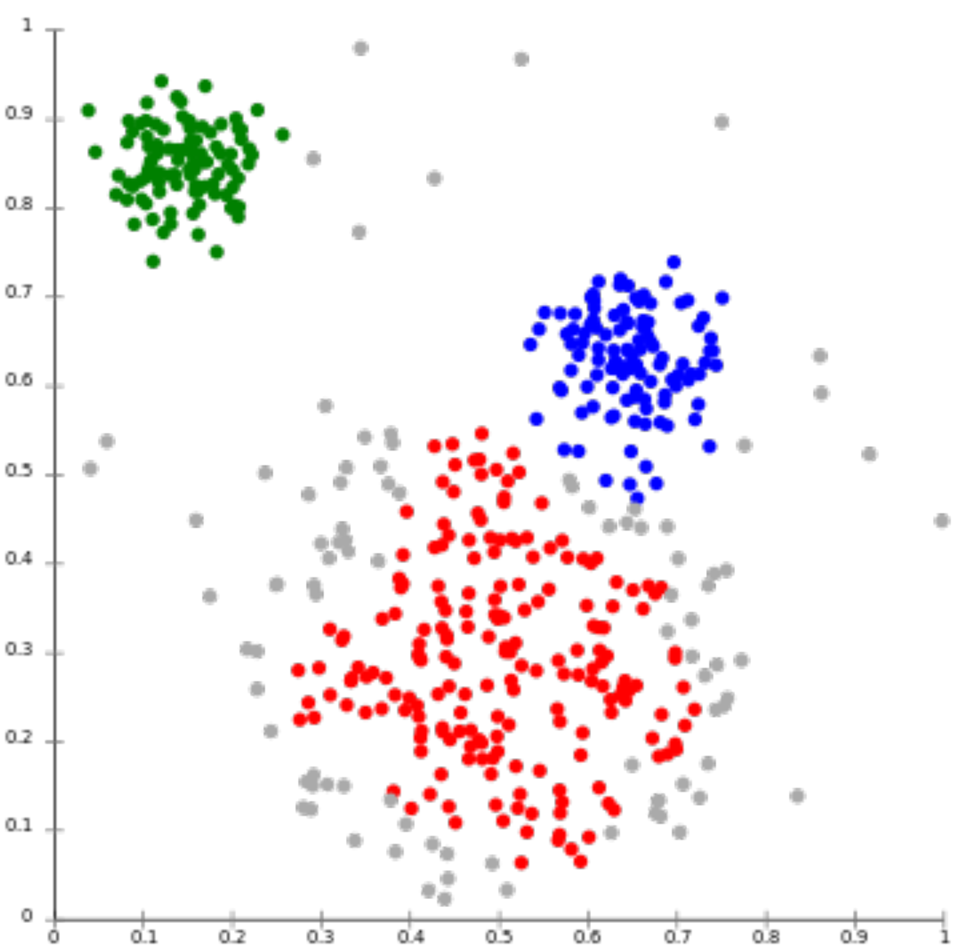


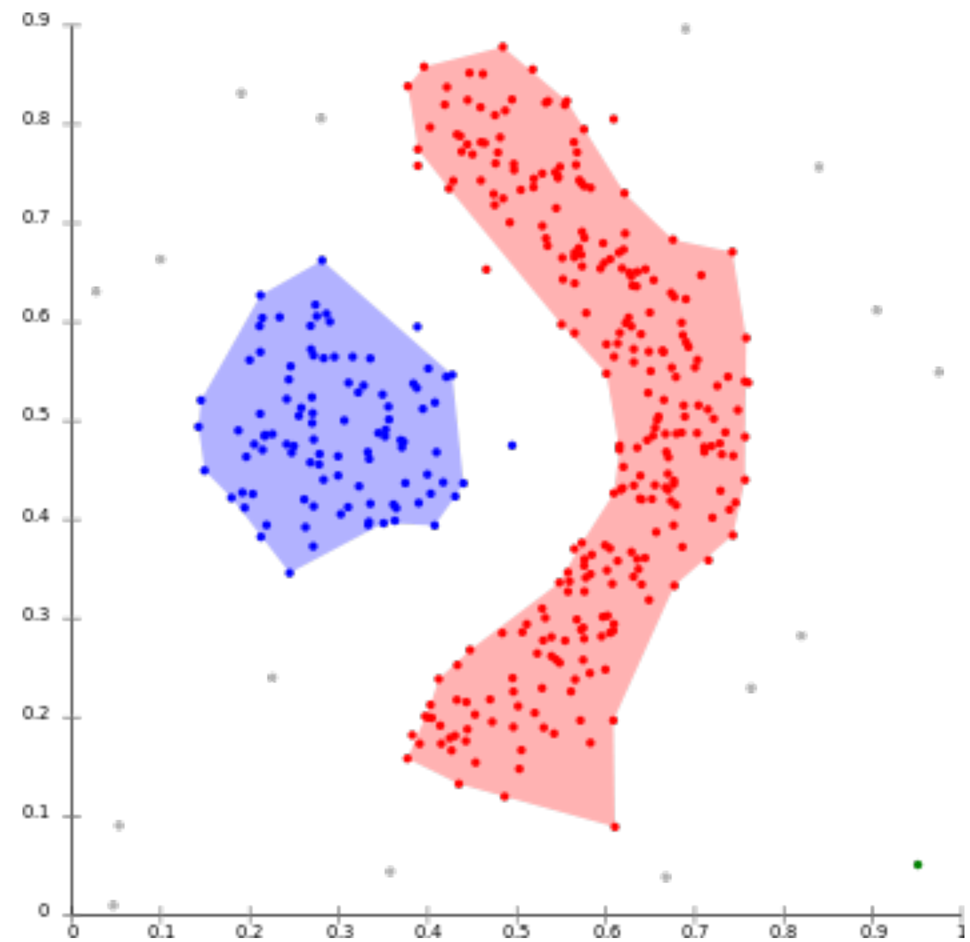


Notion of Clusters: Cut off dendrogram at some depth

Four Types of Clustering

3. *Density-based (DBSCAN, OPTICS)*

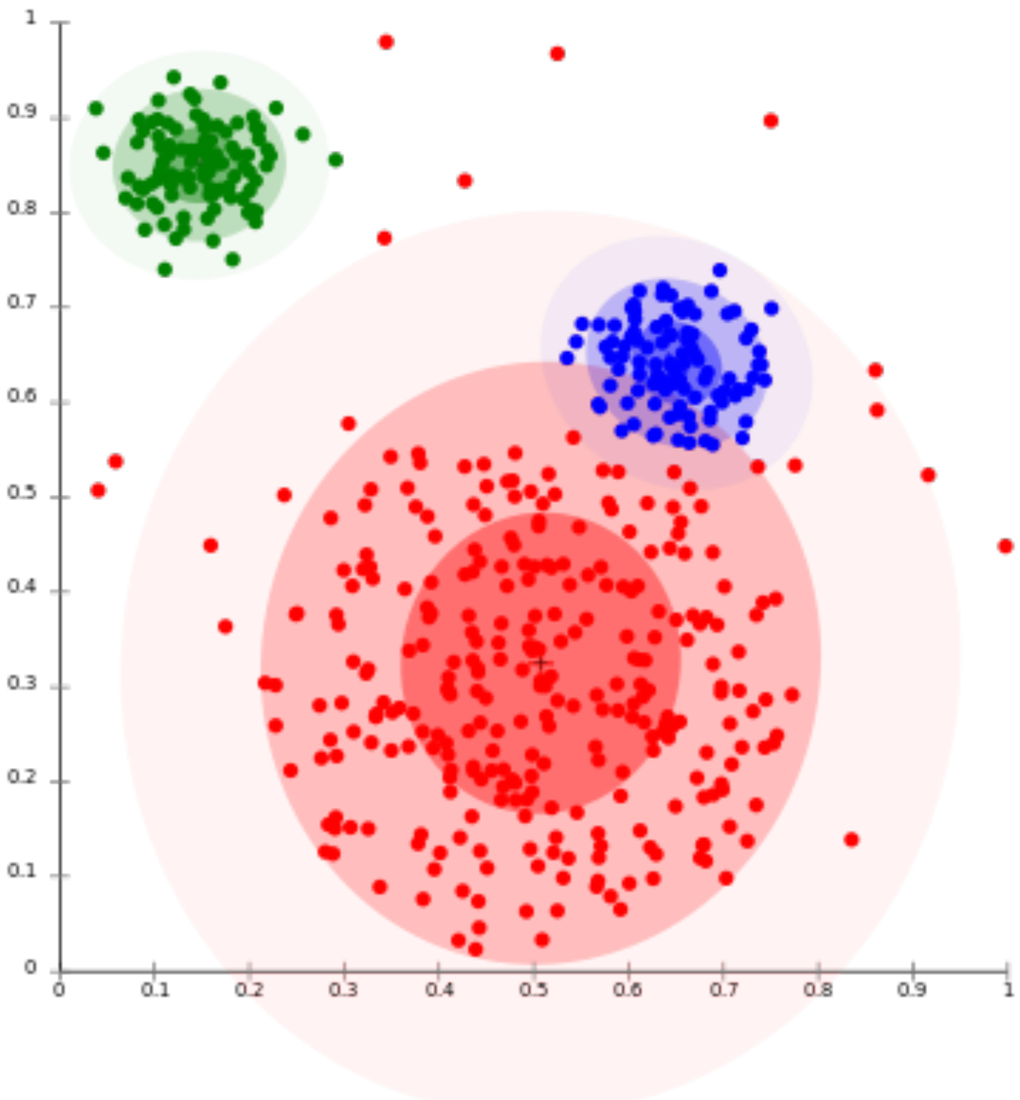


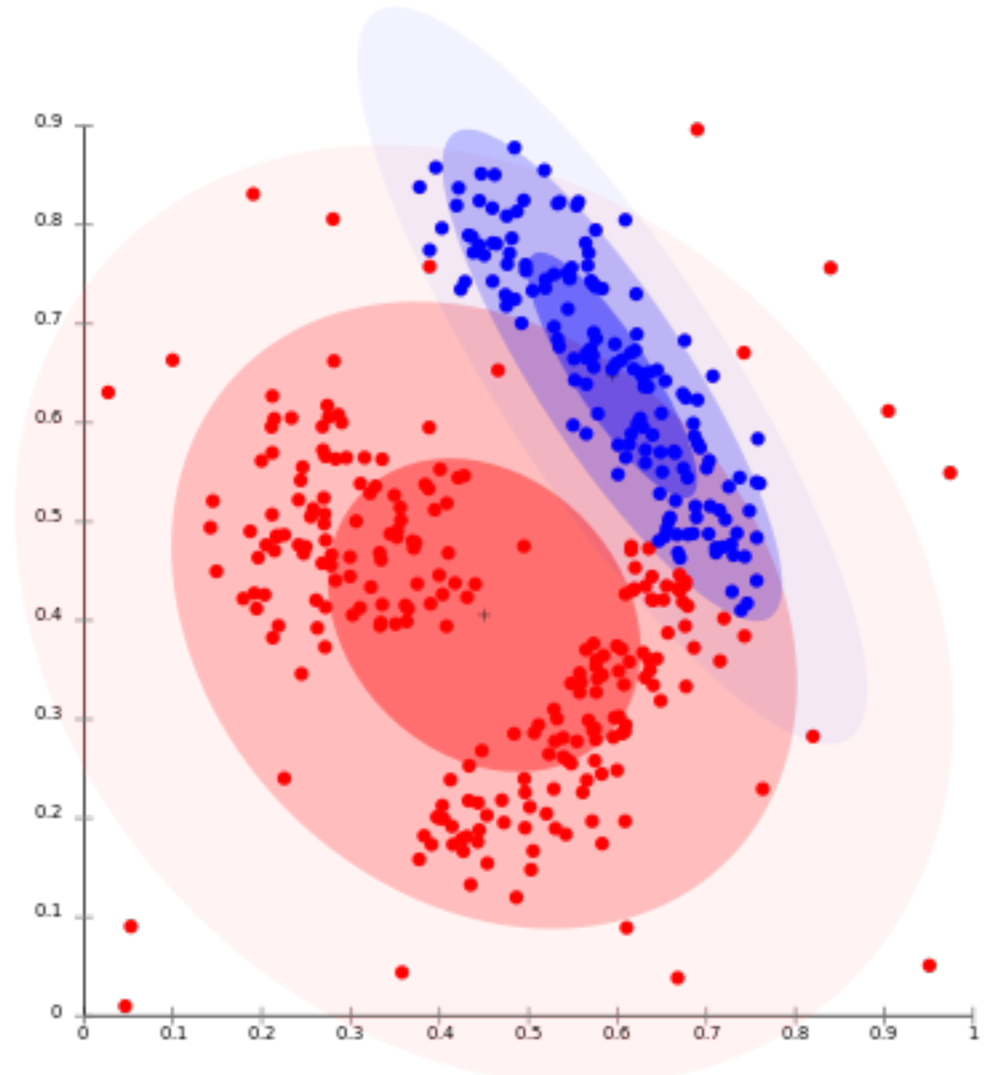


Notion of Clusters: Connected regions of high density

Four Types of Clustering

4. *Distribution-based (Mixture Models)*





Notion of Clusters: Distributions on features

K-Means clustering
(board)

K-means Algorithm

Input:	$X = \{\mathbf{x}_1, \mathbf{x}_2, \ldots, \mathbf{x}_N\}$
	Number of clusters K
Initialize:	K random centroids $\mu_1, \mu_2, \ldots, \mu_K$

K-means Algorithm

Input: $X = \{\mathbf{x}_1, \mathbf{x}_2, \ldots, \mathbf{x}_N\}$
Number of clusters K

Initialize: K random centroids $\mu_1, \mu_2, \ldots, \mu_K$

Repeat Until Convergence

1. For $i = 1, \ldots, K$ do
$C_i = \{\mathbf{x} \in X | i = \arg\min_{1 \leq j \leq K} \| \mathbf{x} - \mu_j \|^2\}$

K-means Algorithm

Input: $X = \{\mathbf{x}_1, \mathbf{x}_2, \ldots, \mathbf{x}_N\}$

Number of clusters K

Initialize: K random centroids $\mu_1, \mu_2, \ldots, \mu_K$

Repeat Until Convergence

1. For $i = 1, \ldots, K$ do

   $$C_i = \{\mathbf{x} \in X | i = \arg\min_{1 \leq j \leq K} \| \mathbf{x} - \mu_j \|^2\}$$

2. For $i = 1, \ldots, K$ do

   $$\mu_i = \arg\min_{\mathbf{z}} \sum_{\mathbf{x} \in C_i} \| \mathbf{z} - \mathbf{x} \|^2$$

K-means Algorithm

Input: $X = \{\mathbf{x}_1, \mathbf{x}_2, \ldots, \mathbf{x}_N\}$
Number of clusters K

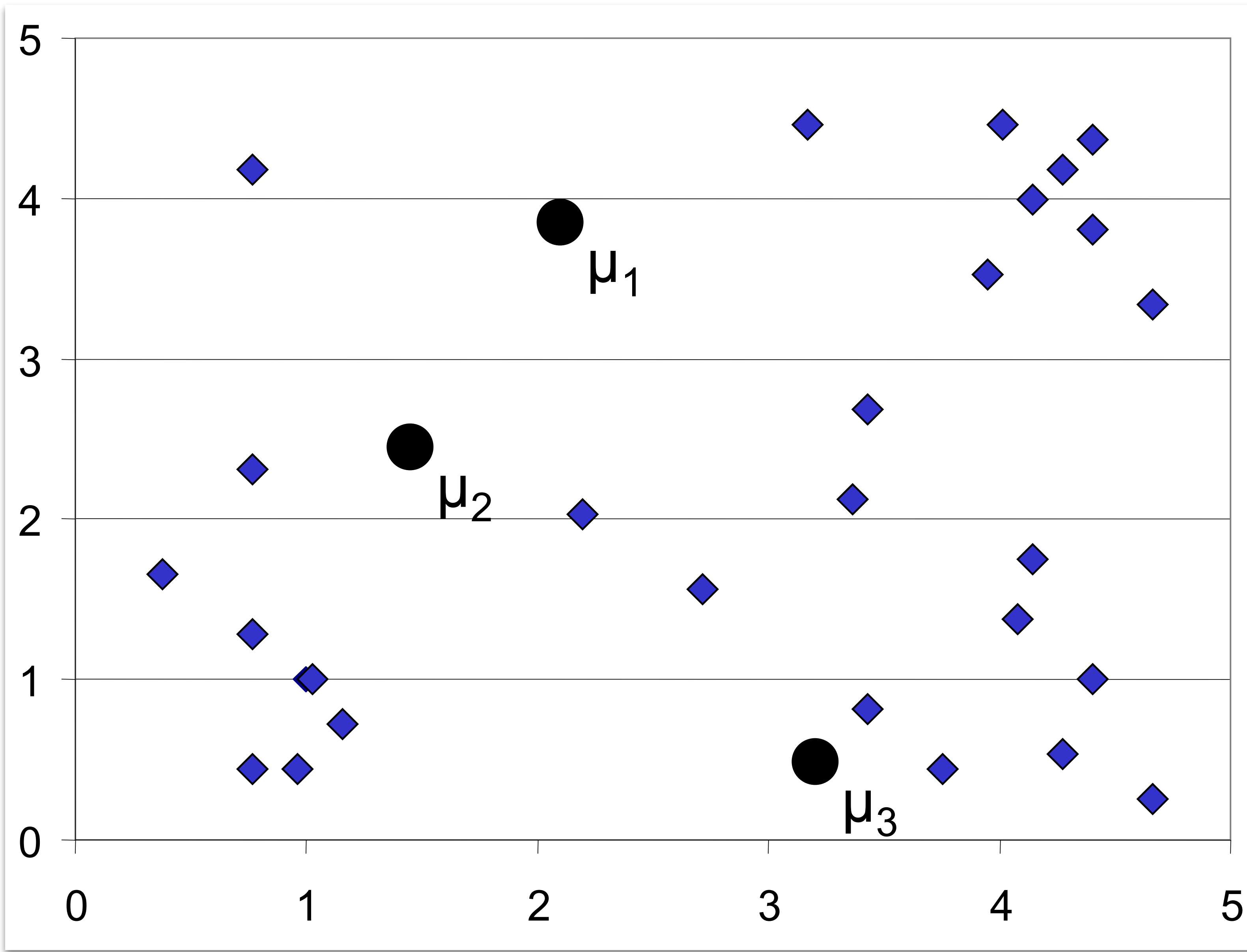
Initialize: K random centroids $\mu_1, \mu_2, \ldots, \mu_K$

Repeat Until Convergence

1. For $i = 1, \ldots, K$ do
   $C_i = \{\mathbf{x} \in X | i = \arg\min_{1 \leq j \leq K} \| \mathbf{x} - \mu_j \|^2\}$
2. For $i = 1, \ldots, K$ do
   $\mu_i = \arg\min_{\mathbf{z}} \sum_{\mathbf{x} \in C_i} \| \mathbf{z} - \mathbf{x} \|^2$

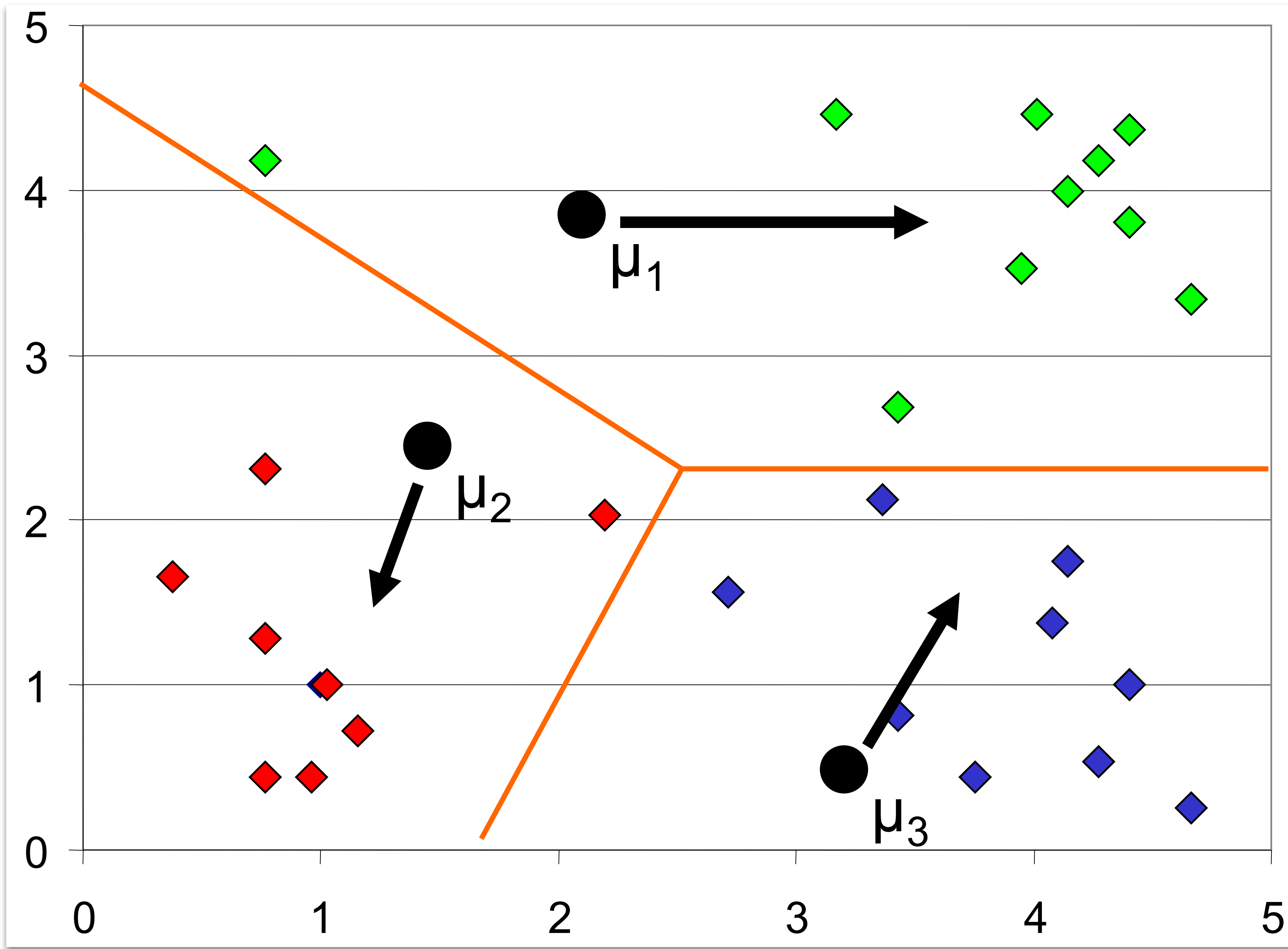
Output: $C_1, C_2, \ldots, C_K$

K-means Clustering



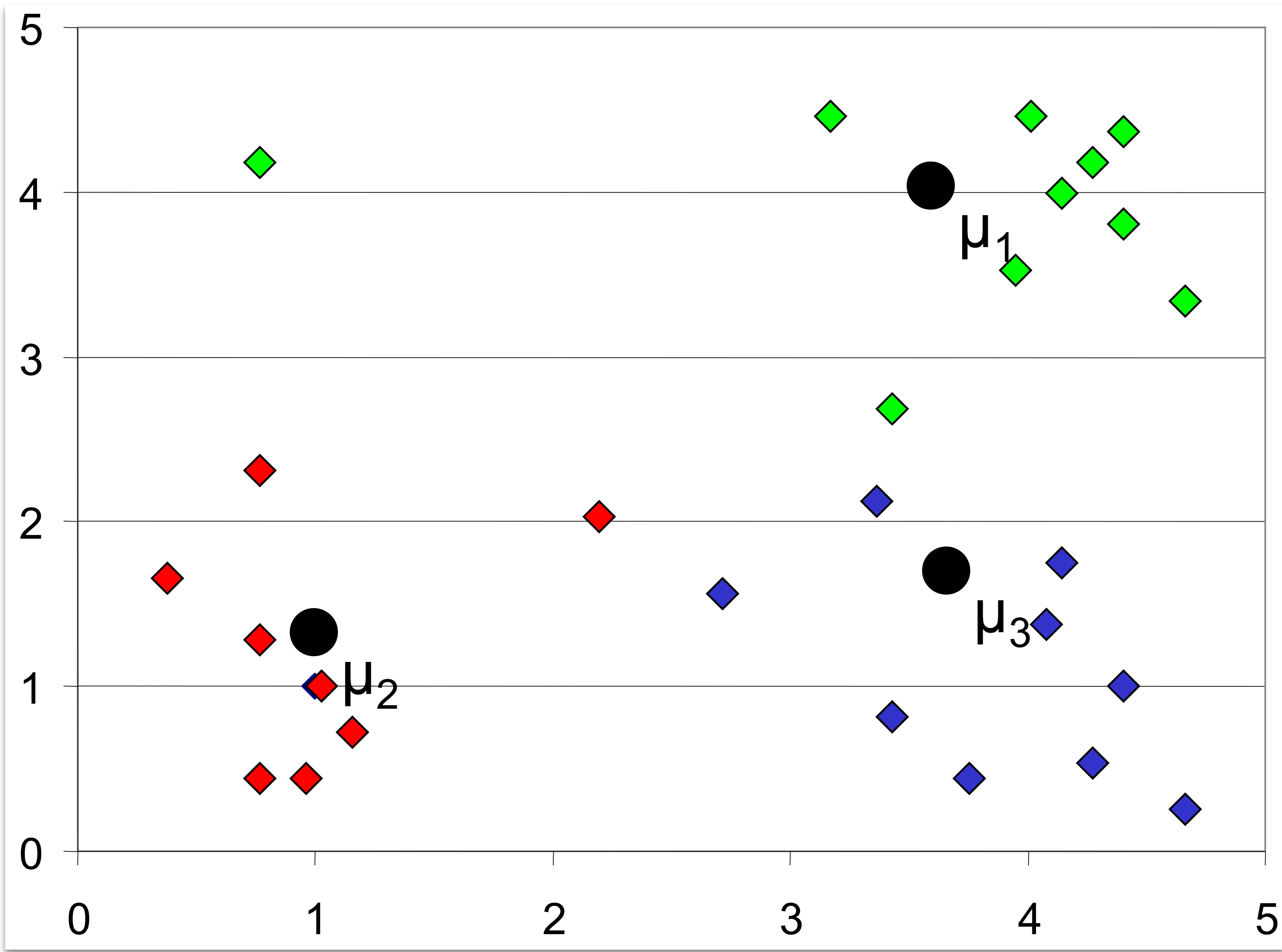
Randomly initialize *K* centroids μ_k

K-means Clustering



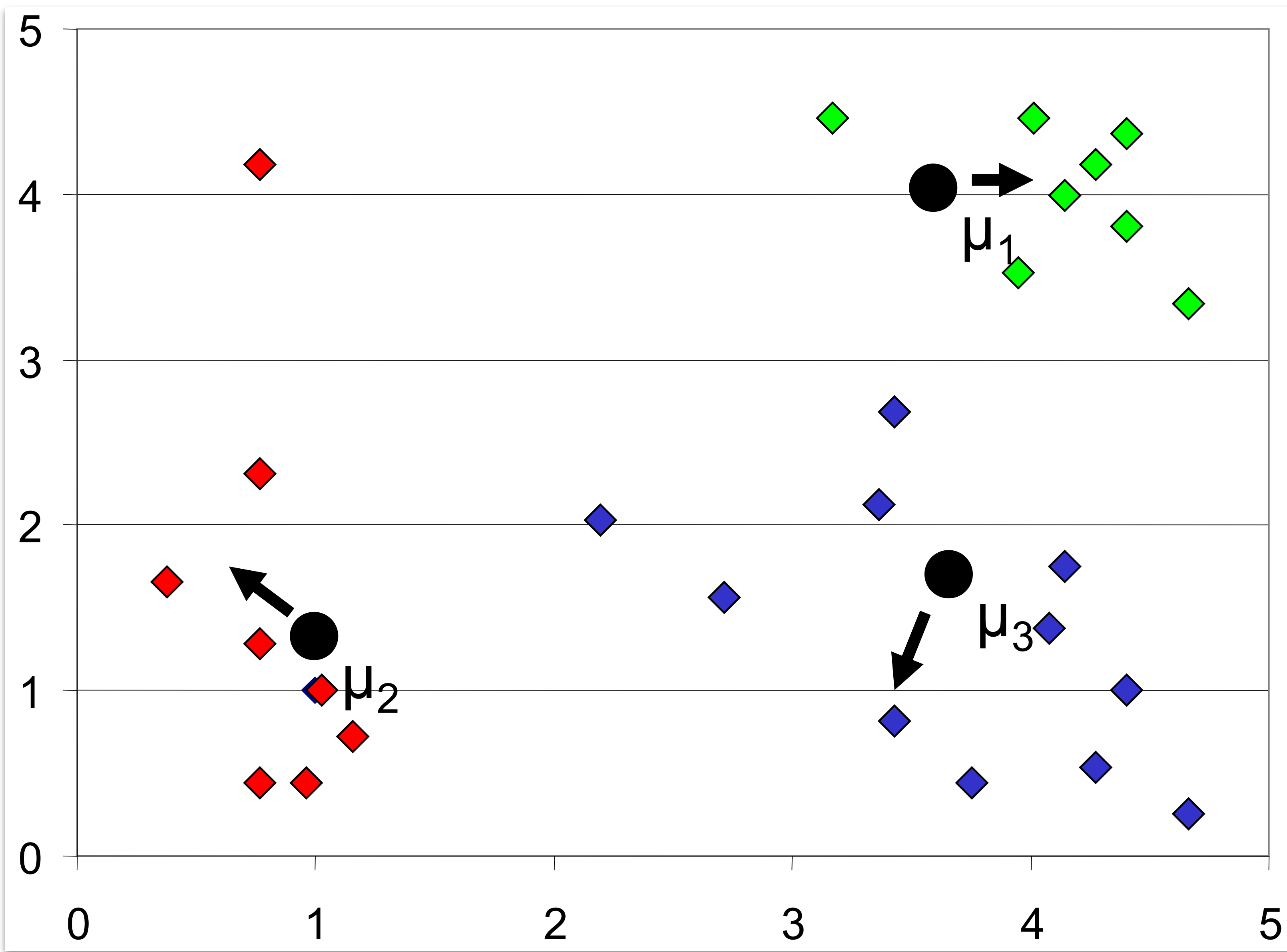
Assign each point to closest centroid, then update centroids to average of points

K-means Clustering



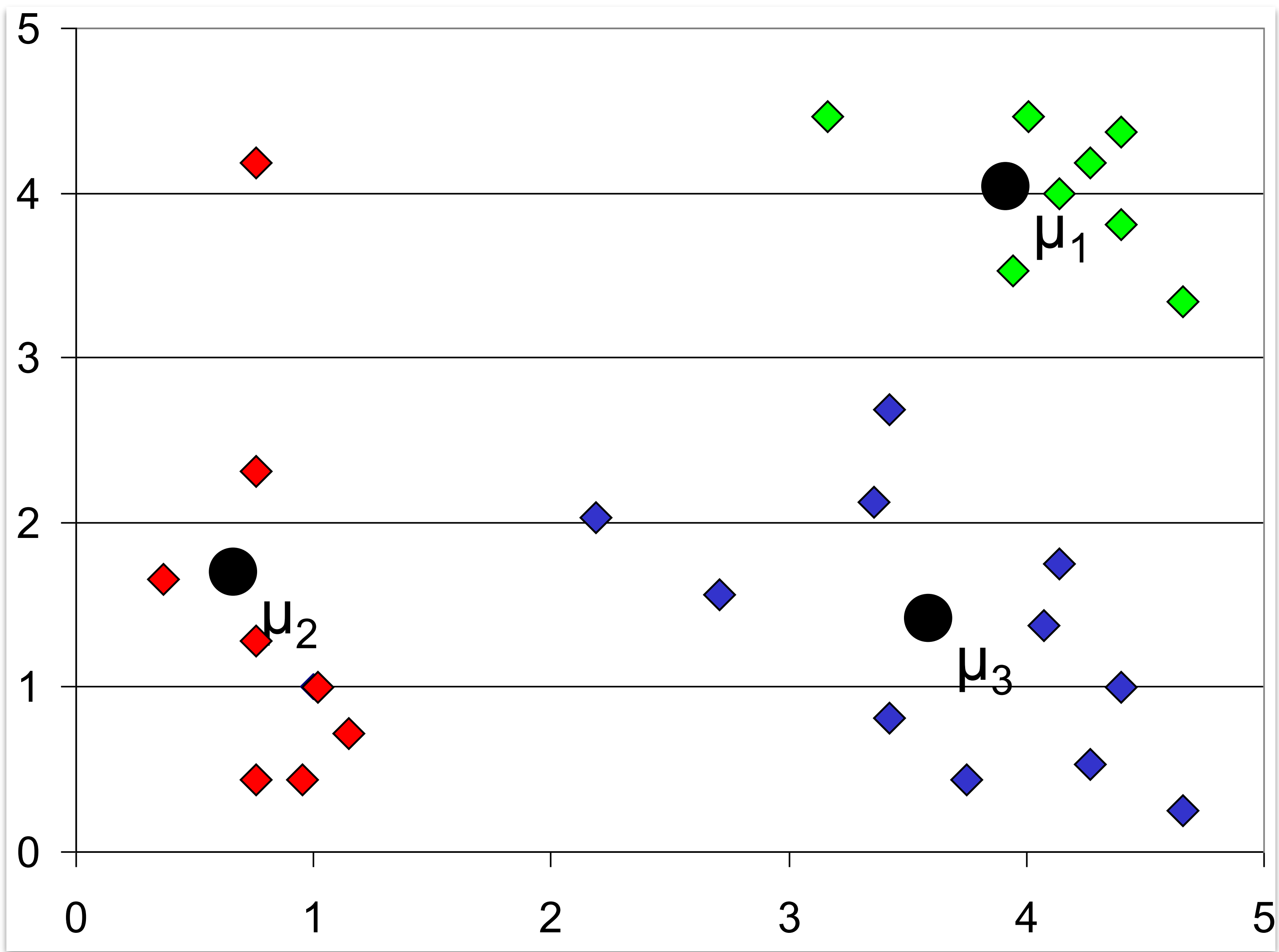
Assign each point to closest centroid,
then update centroids to average of points

K-means Clustering



Repeat until convergence
(no points reassigned, means unchanged)

K-means Clustering



Repeat until convergence
(no points reassigned, means unchanged)

K-means Algorithm

Input: $X = \{\mathbf{x}_1, \mathbf{x}_2, \ldots, \mathbf{x}_N\}$
Number of clusters K

Initialize: K random centroids $\mu_1, \mu_2, \ldots, \mu_K$
Repeat Until Convergence

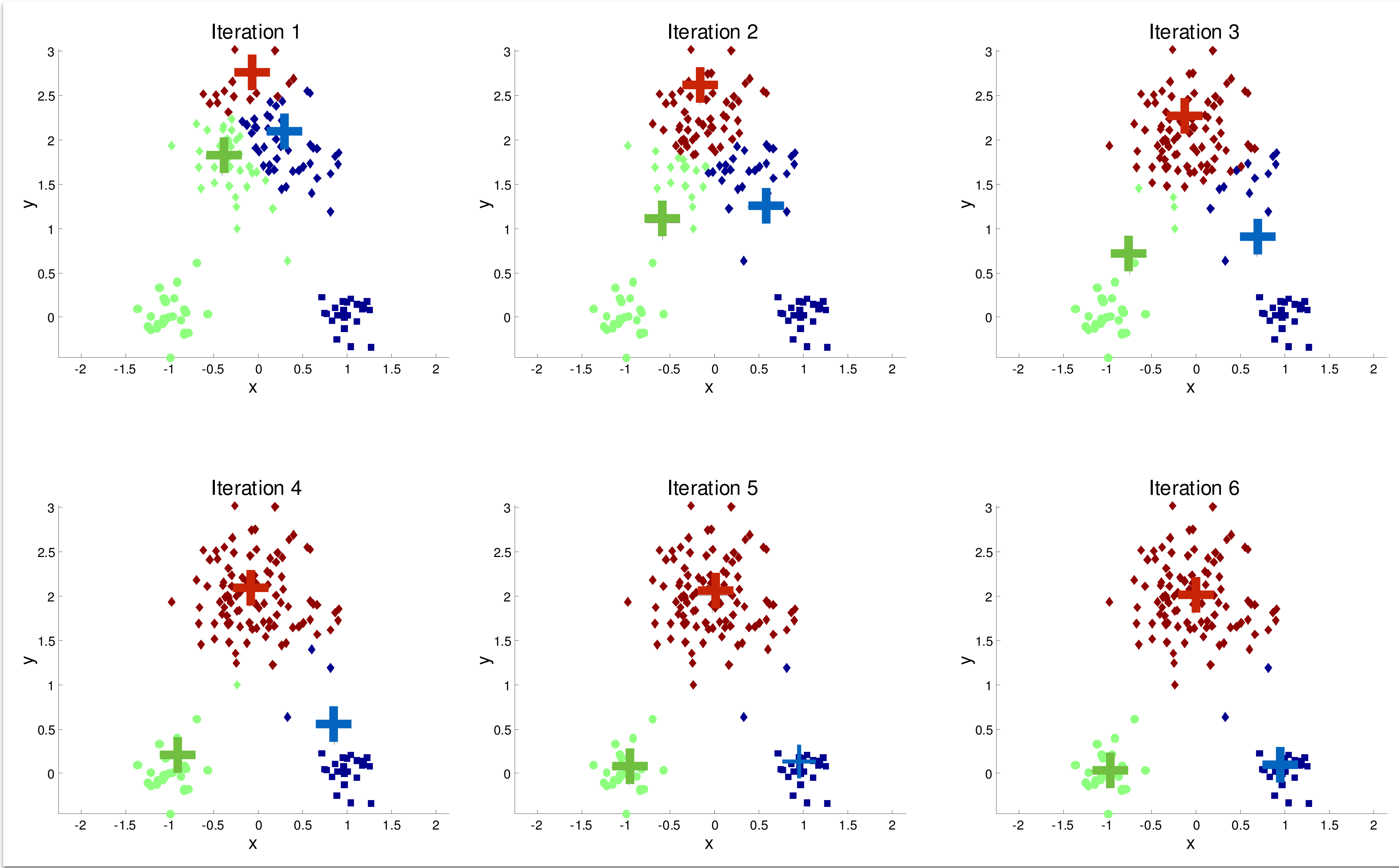
1. For $i = 1, \ldots, K$ do
$C_i = \{\mathbf{x} \in X | i = \arg\min_{1 \leq j \leq K} \| \mathbf{x} - \mu_j \|^2\}$
2. For $i = 1, \ldots, K$ do
$\boxed{\mu_i = \arg\min_{\mathbf{z}} \sum_{\mathbf{x} \in C_i} \| \mathbf{z} - \mathbf{x} \|^2}$

Output: $C_1, C_2, \ldots, C_K$

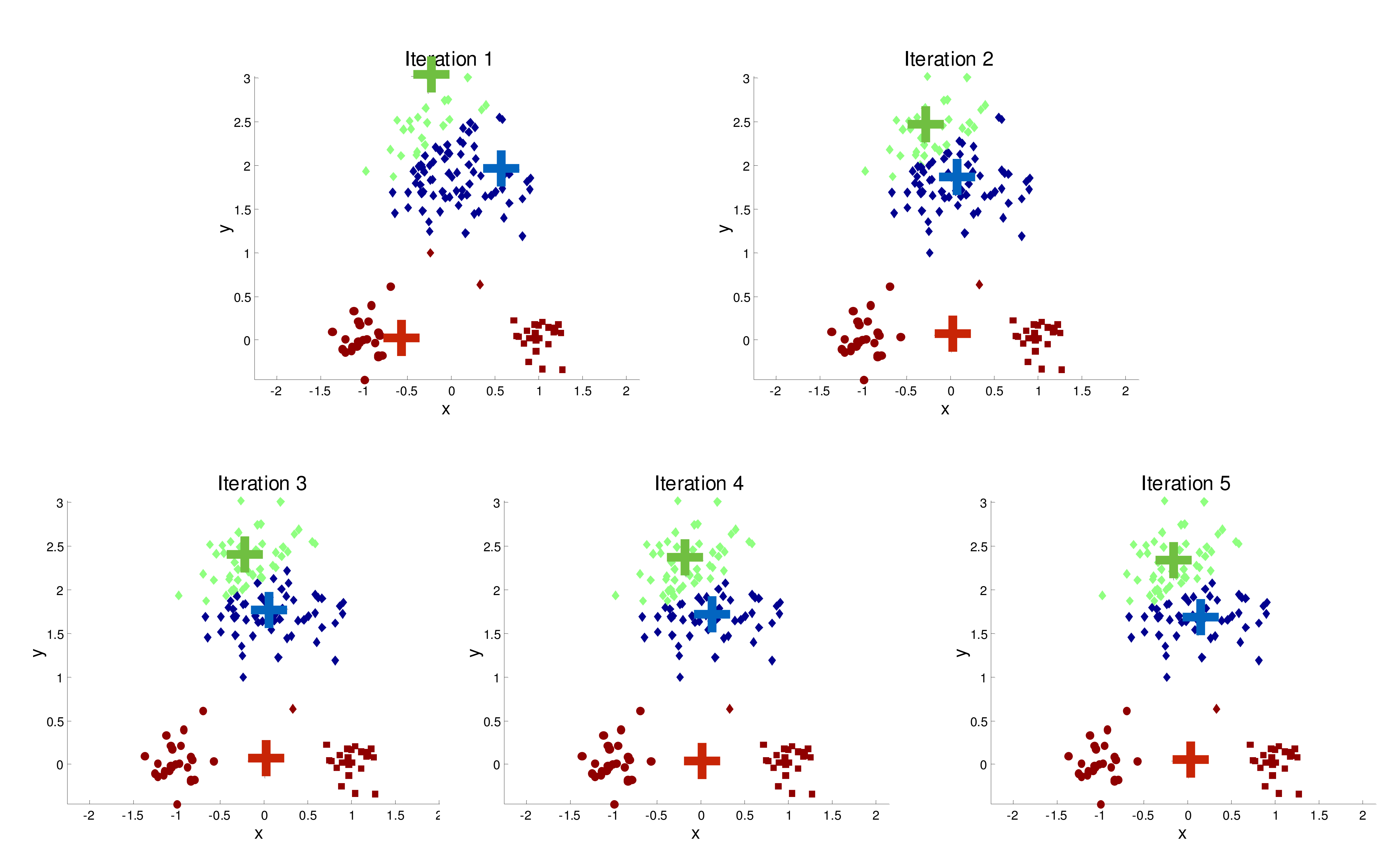
- K-means: Set μ to mean of points in C
- K-medoids: Set $\mu = x$ for point in C with minimum SSE

Let's see some examples in Python

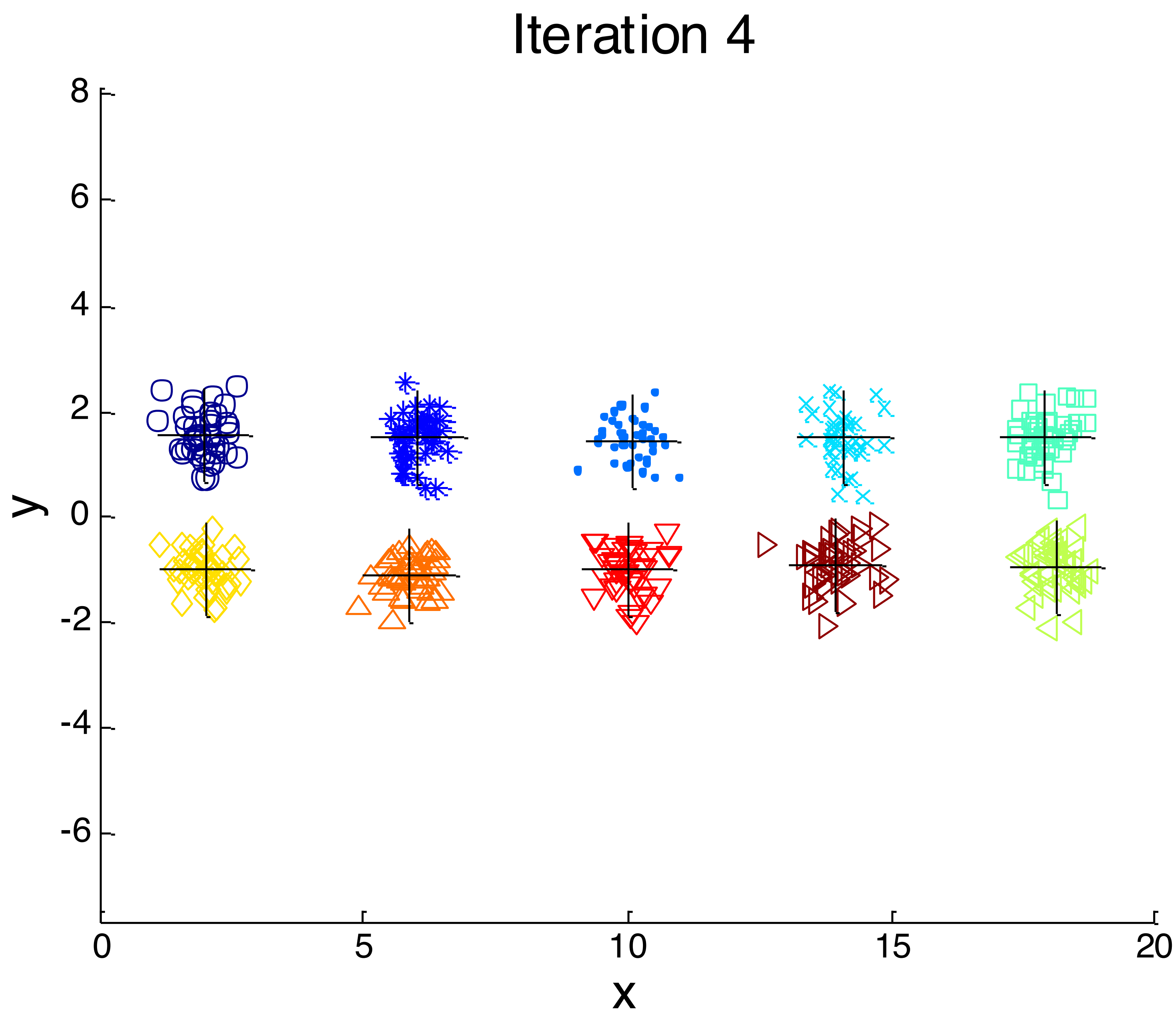
“Good” Initialization of Centroids



“Bad” Initialization of Centroids

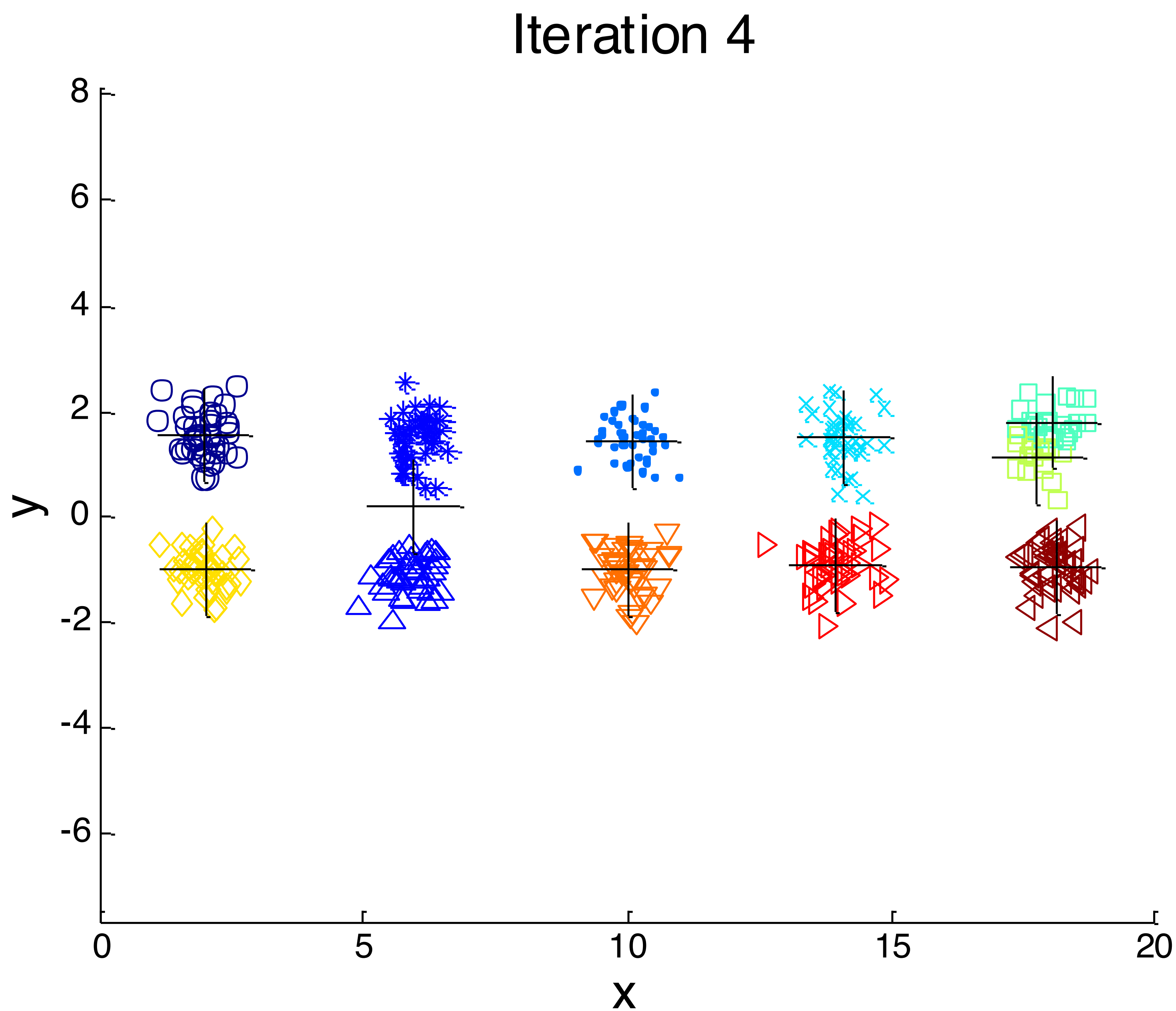


Example: 10 Clusters



5 pairs of clusters, two initial points in each pair

Example: 10 Clusters



5 pairs of clusters, two initial points in each pair

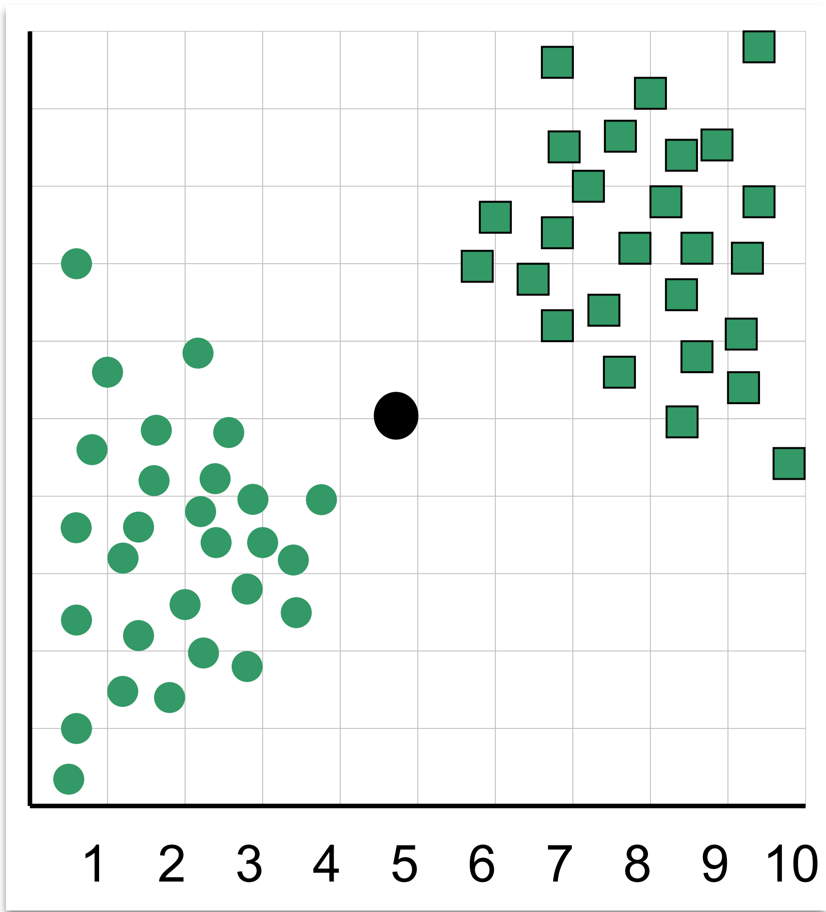
Importance of Initial Centroids

Initialization tricks

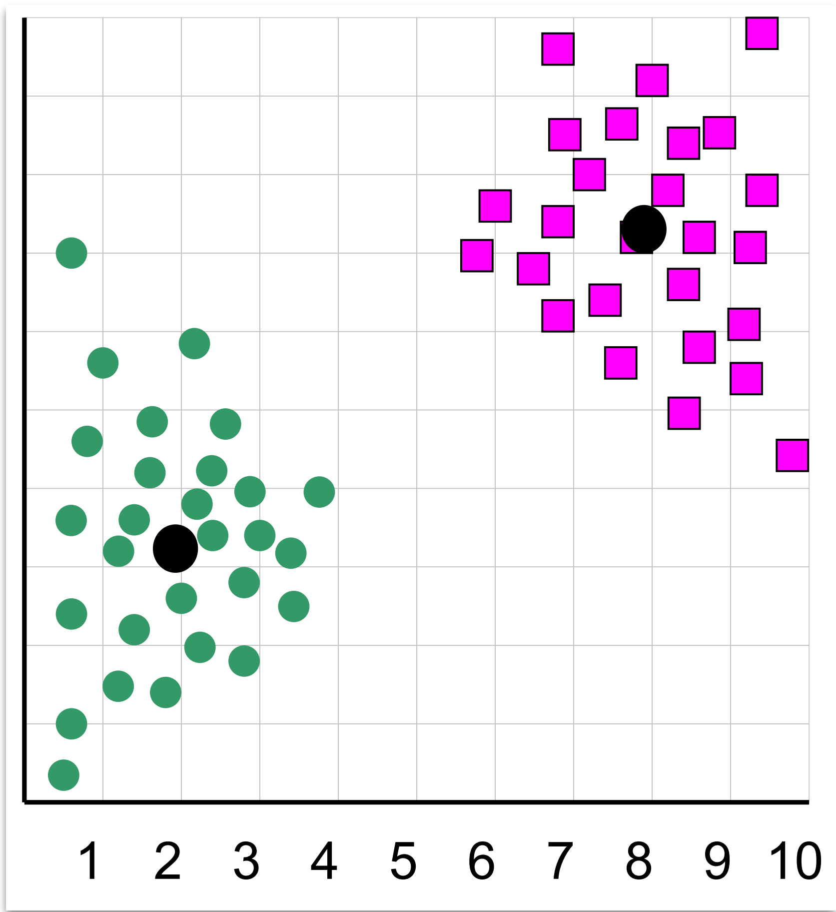
- Use multiple restarts
- Initialize with hierarchical clustering
- Select more than K points,
  keep most widely separated points

Choosing K

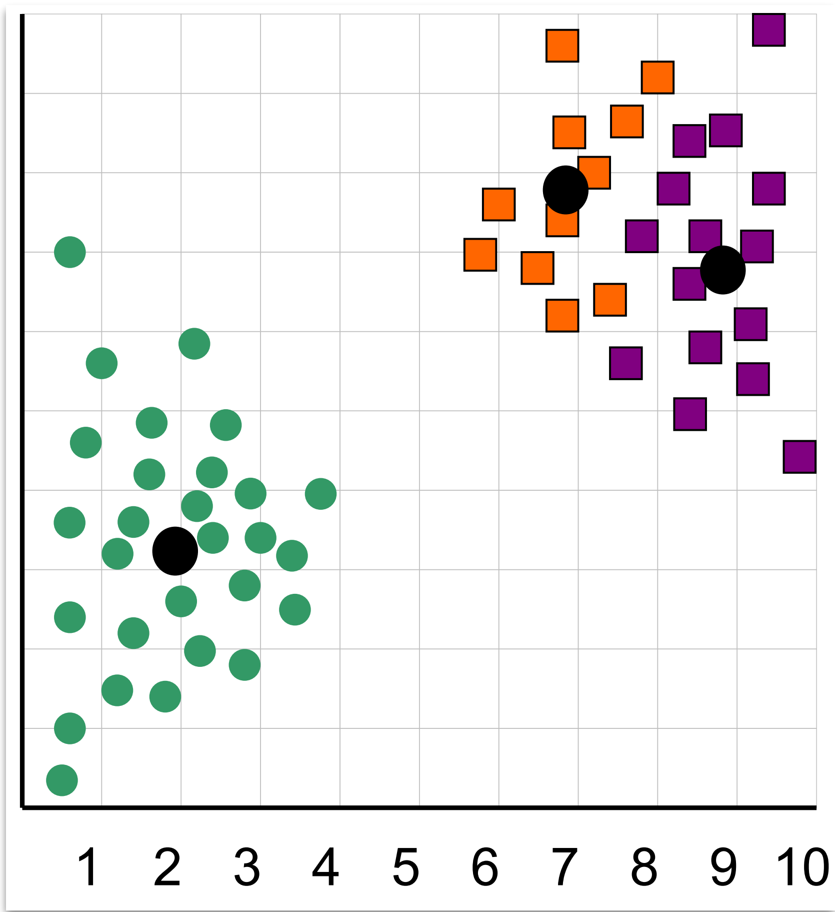
K=1, SSE=873



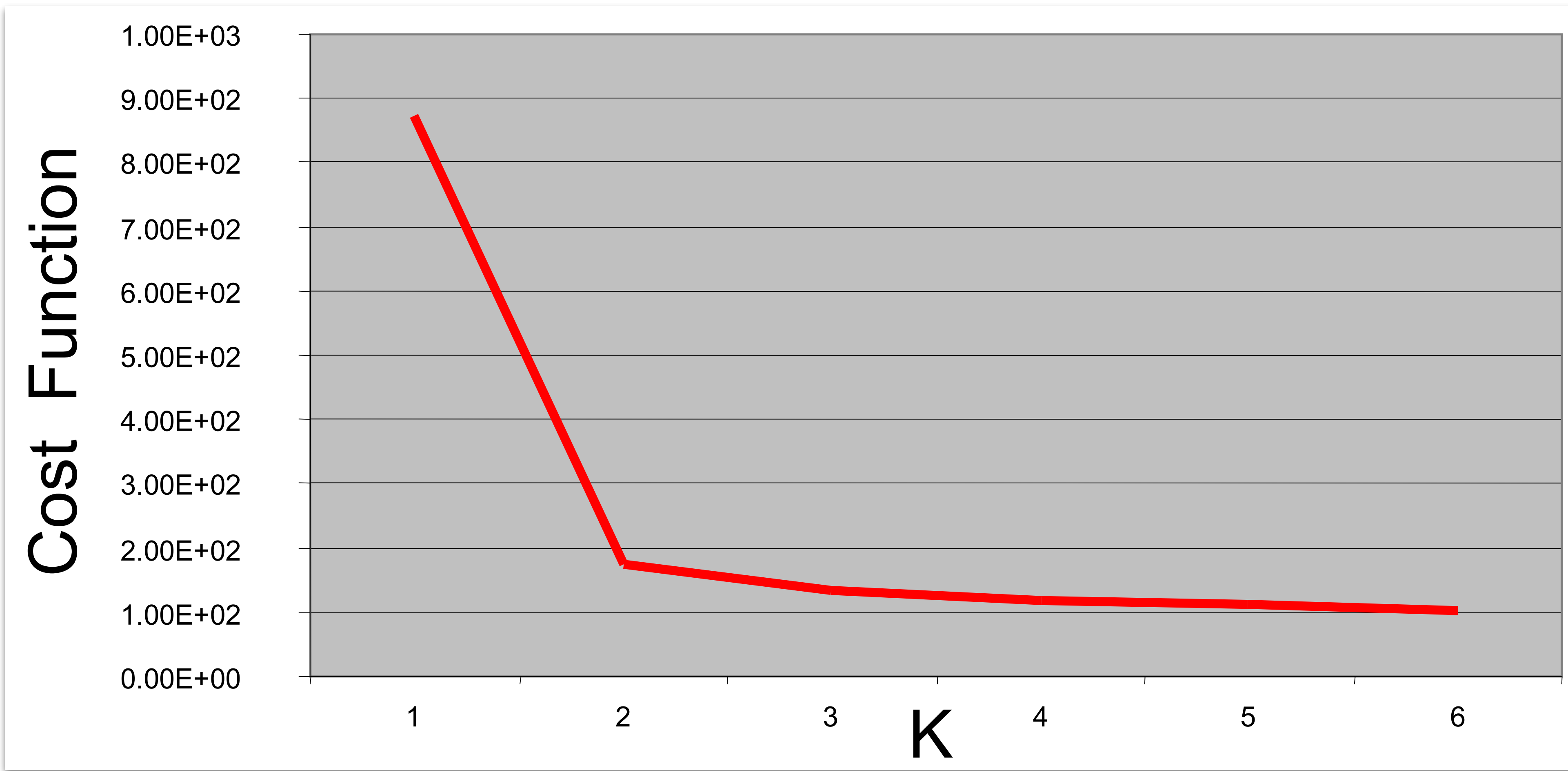
K=2, SSE=173



K=3, SSE=134



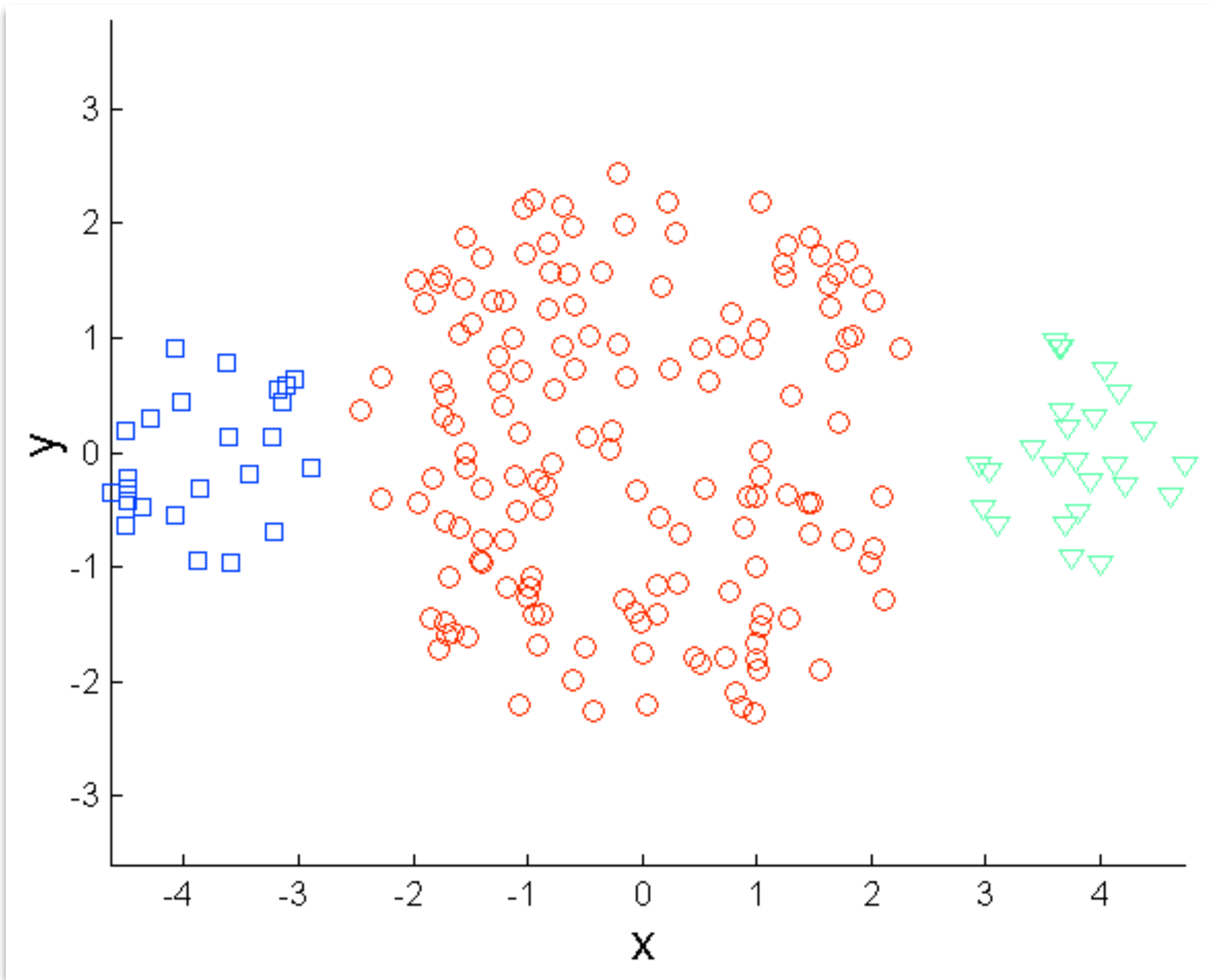
Choosing K



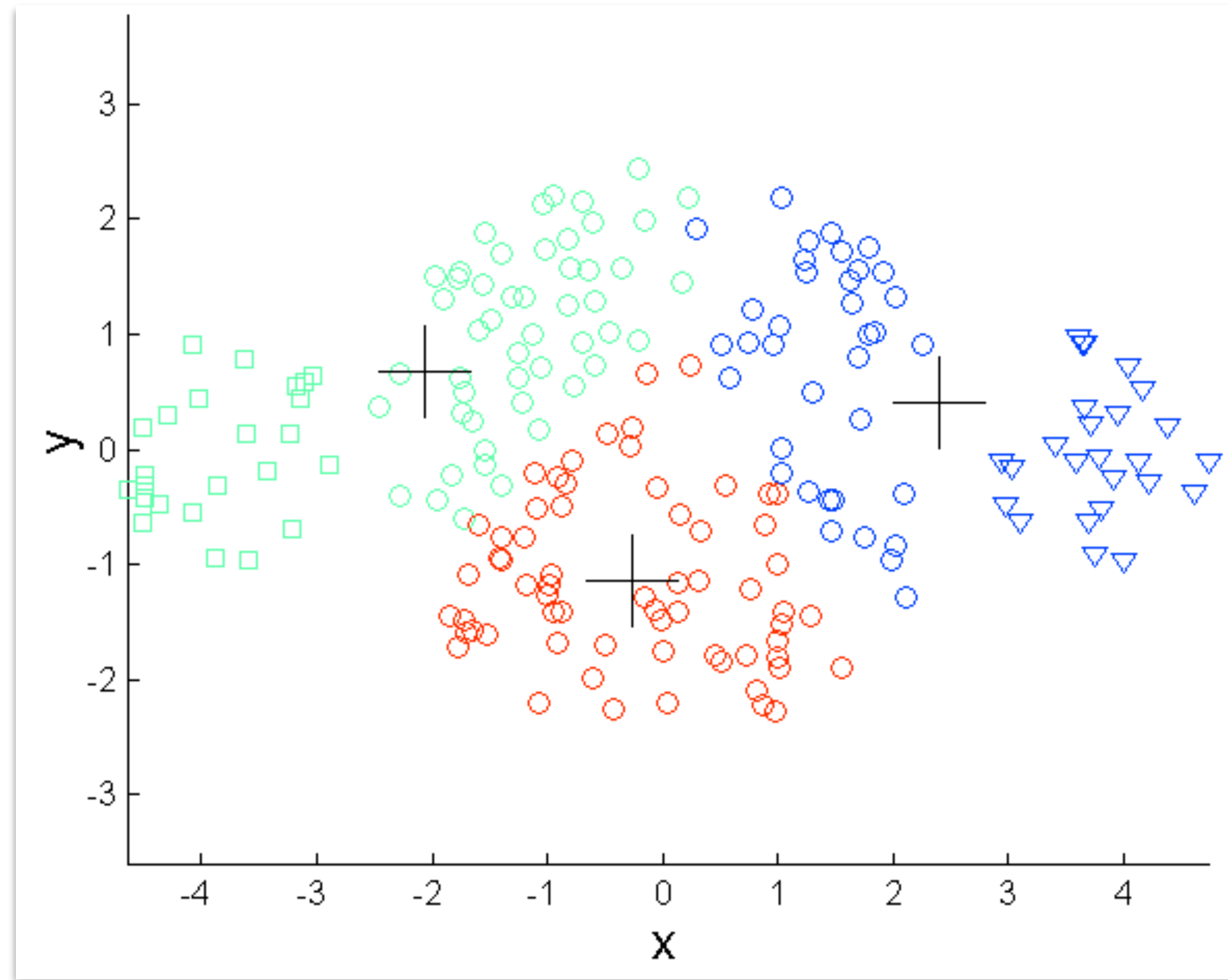
"Elbow finding" (a.k.a. "knee finding")

Set K to value just above "abrupt" increase

K-means Limitations: Differing Sizes

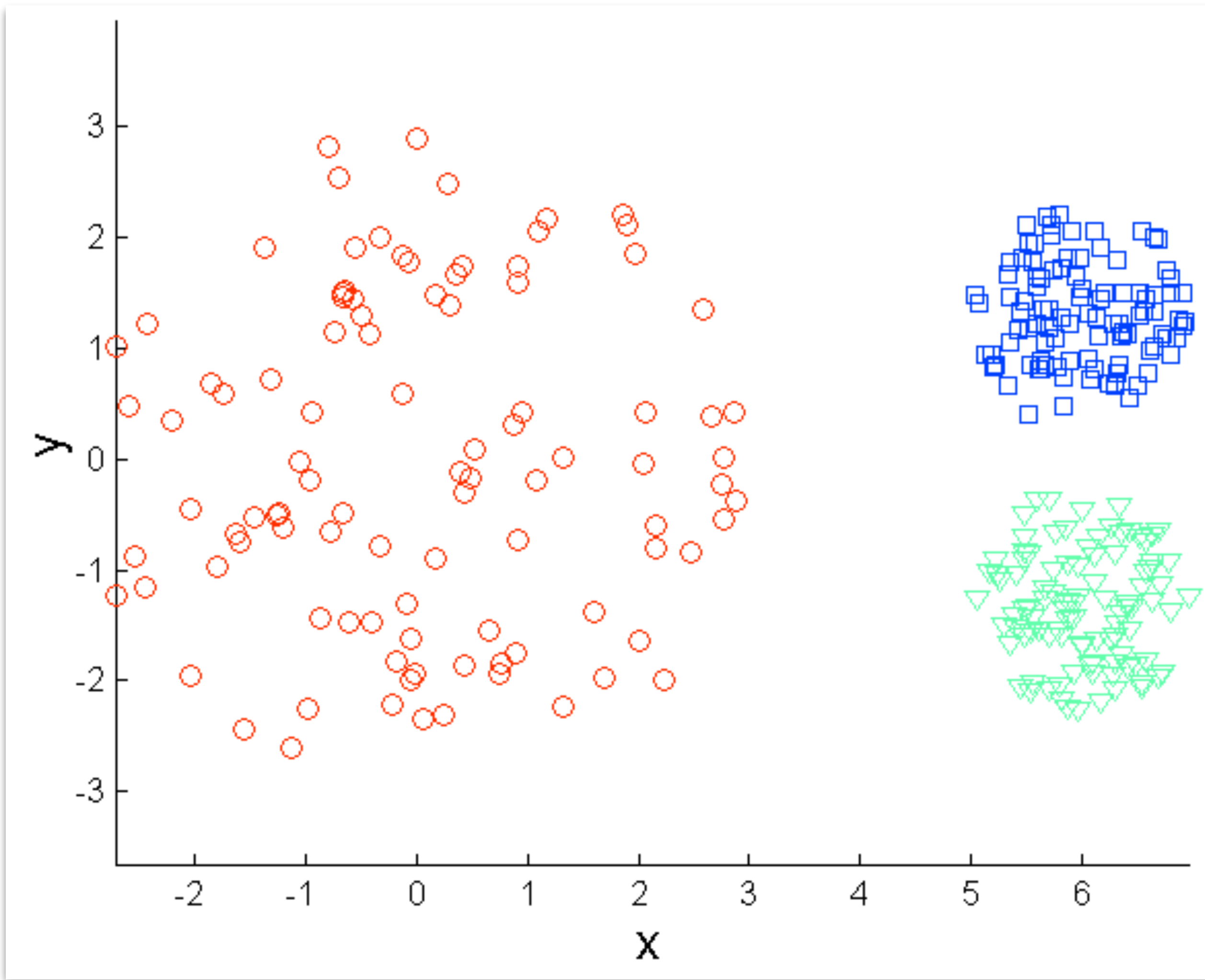


Original Points

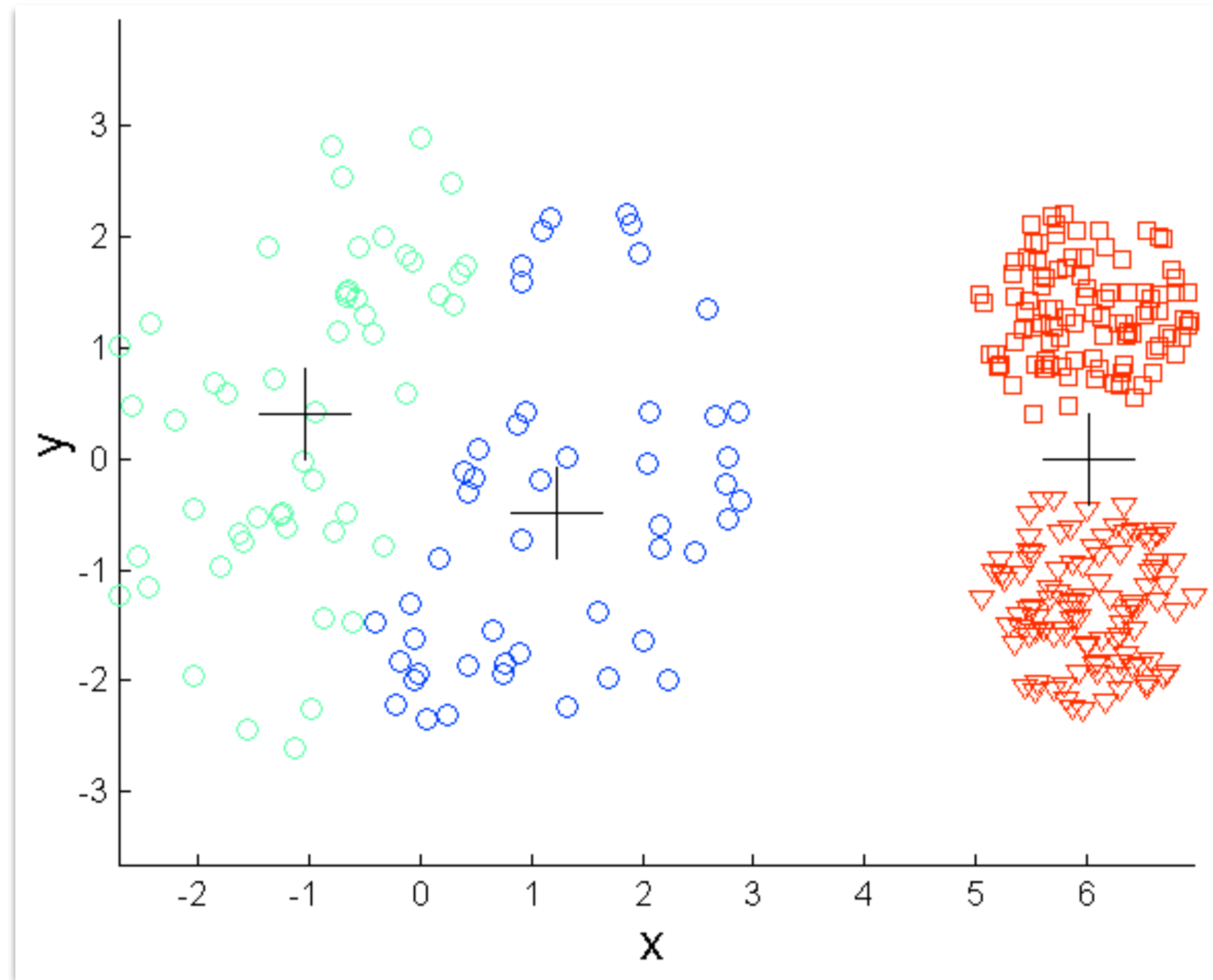


K-means (3 clusters)

K-means Limitations: Different Densities

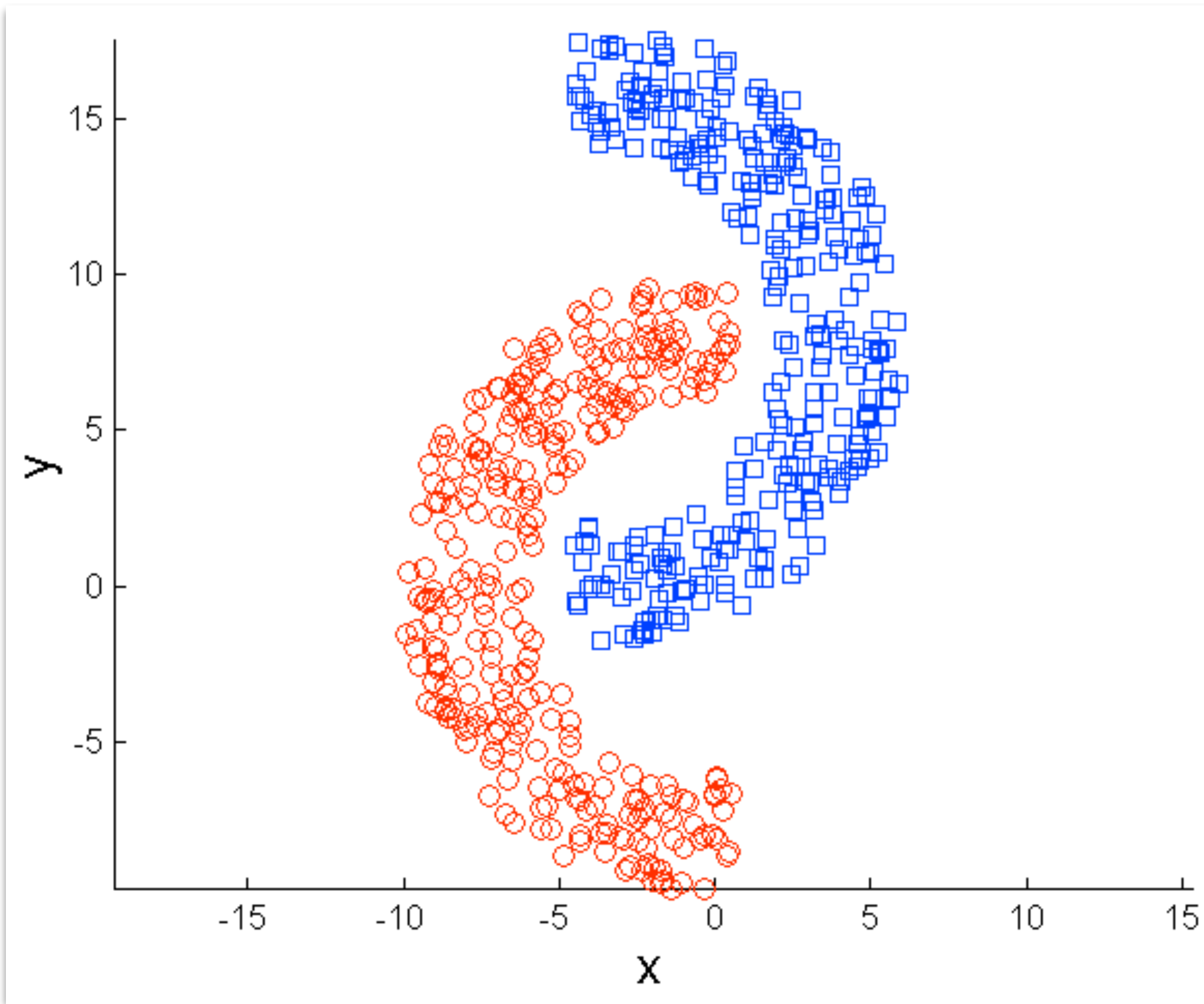


Original Points

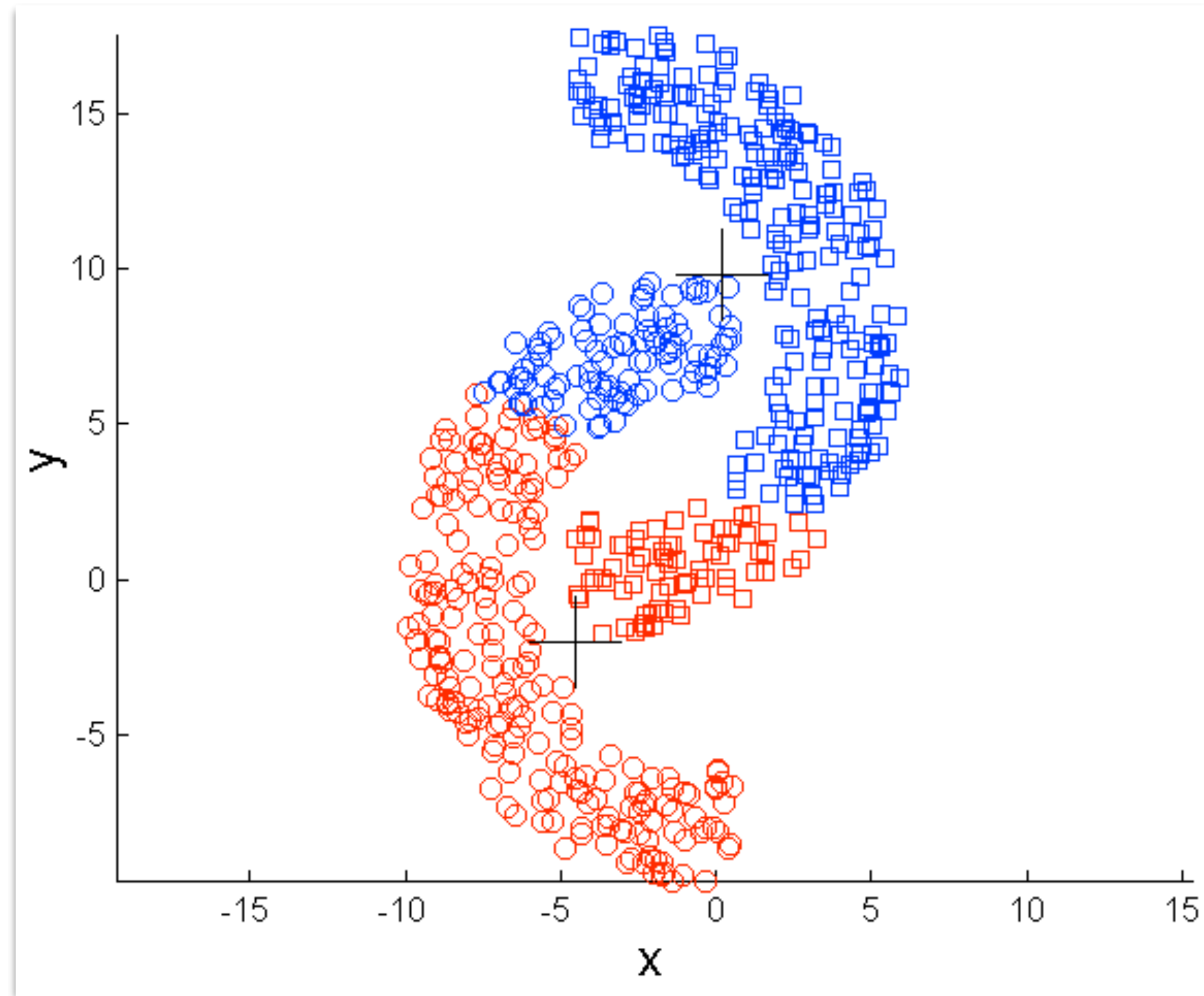


K-means (3 clusters)

K-means Limitations: Non-globular Shapes

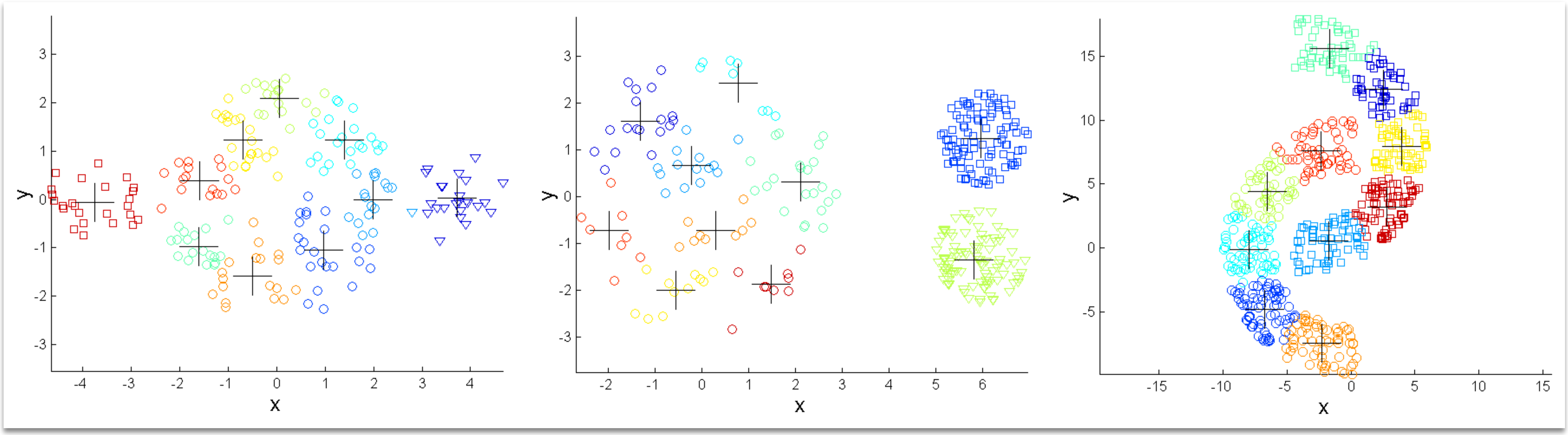


Original Points



K-means (2 clusters)

Overcoming K-means Limitations



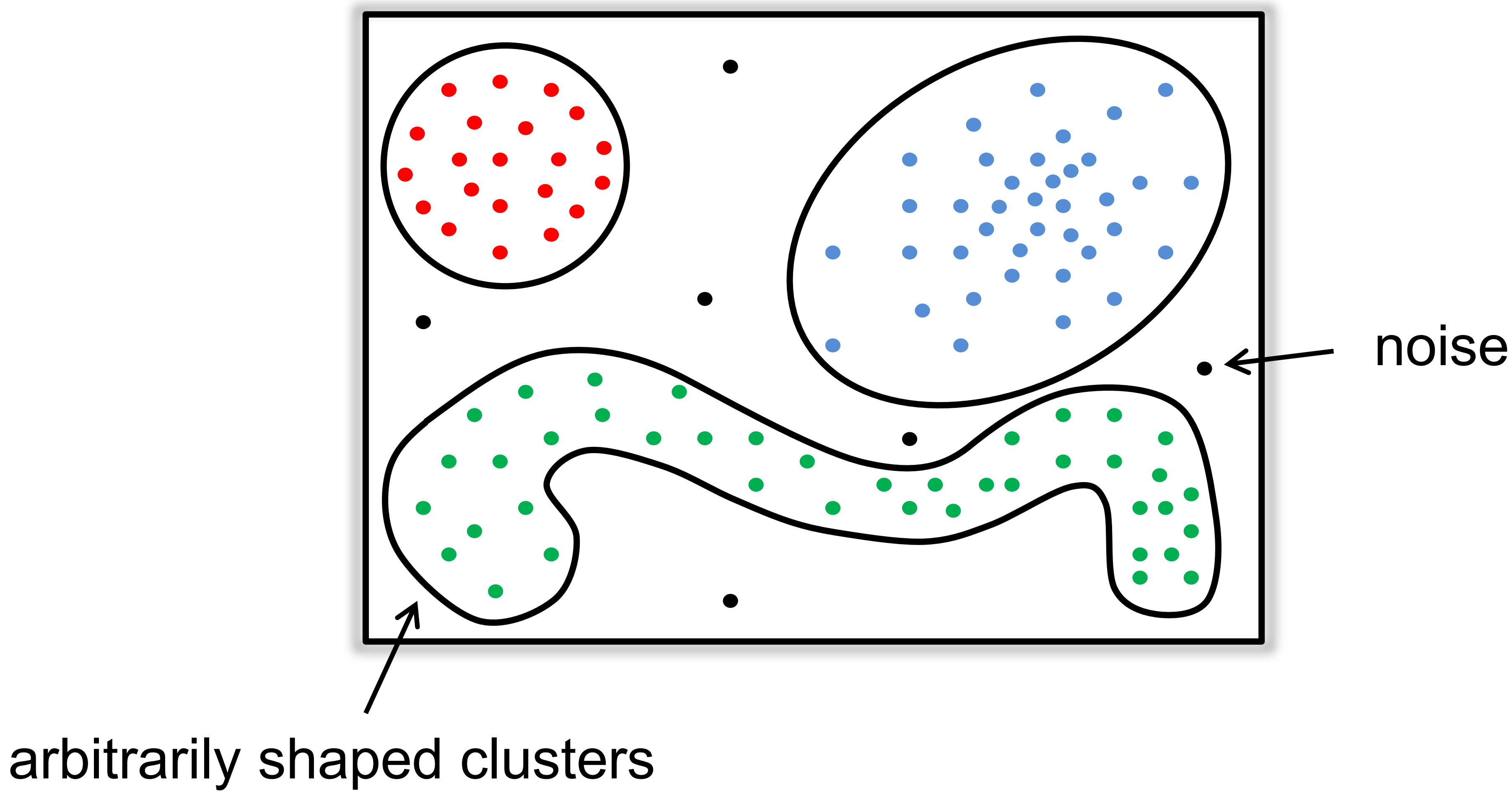
Intuition: "Combine" smaller clusters into larger clusters

- *One Solution:* Hierarchical Clustering
- *Another Solution:* Density-based Clustering

K-means in action: Download the notebook starter for today from blackboard (and CSV file)

Density-based Clustering

DBSCAN



[PDF] A density-based algorithm for discovering clusters in large spatial databases with noise.

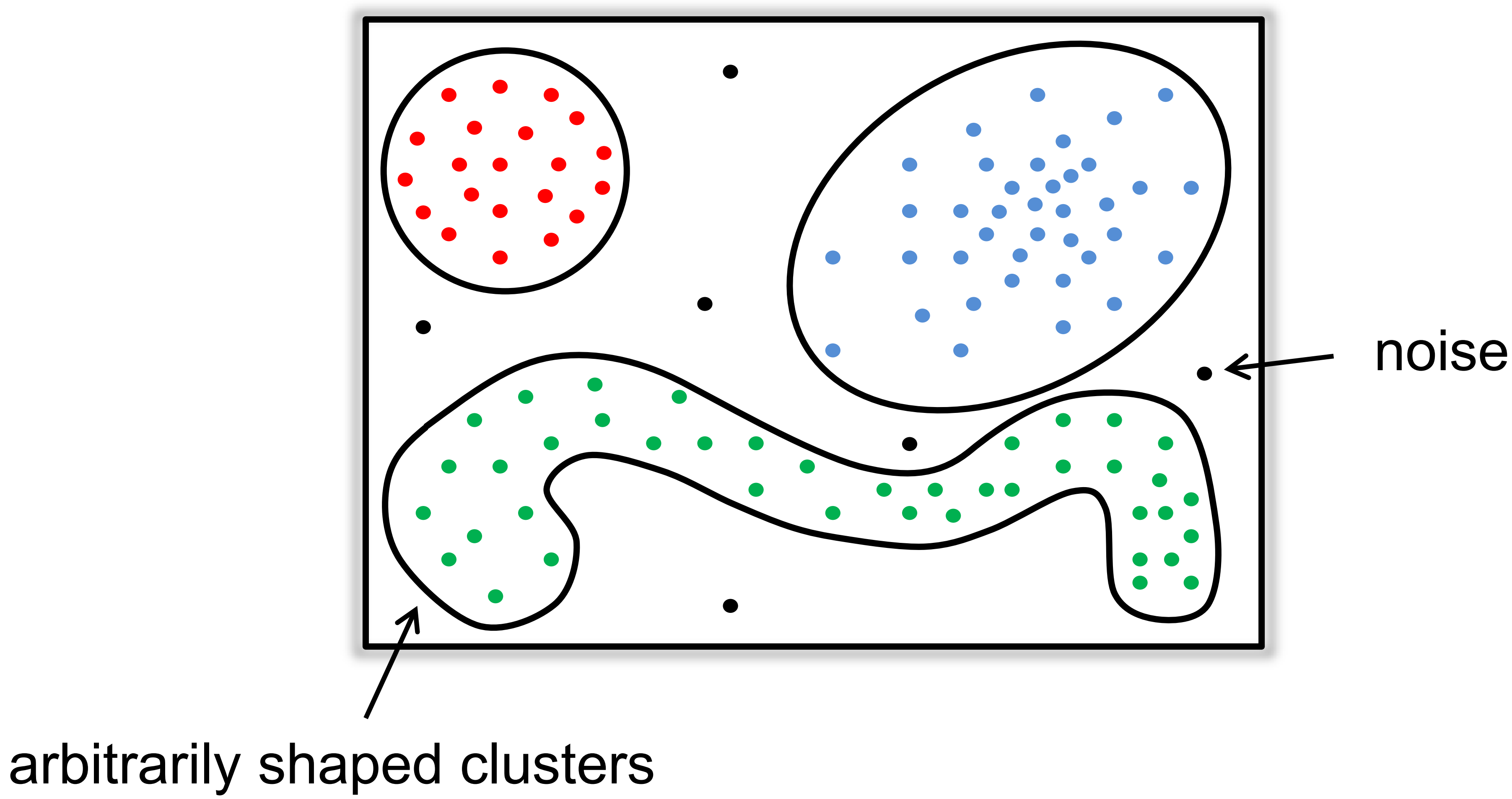
M Ester, HP Kriegel, J Sander, X Xu - Kdd, 1996 - aaai.org

Abstract Clustering algorithms are attractive for the task of class identification in spatial databases. However, the application to large spatial databases rises the following requirements for clustering algorithms: minimal requirements of domain knowledge to ...

Cited by 8901 Related articles All 70 versions Cite Save More

(one of the most-cited clustering methods)

DBSCAN



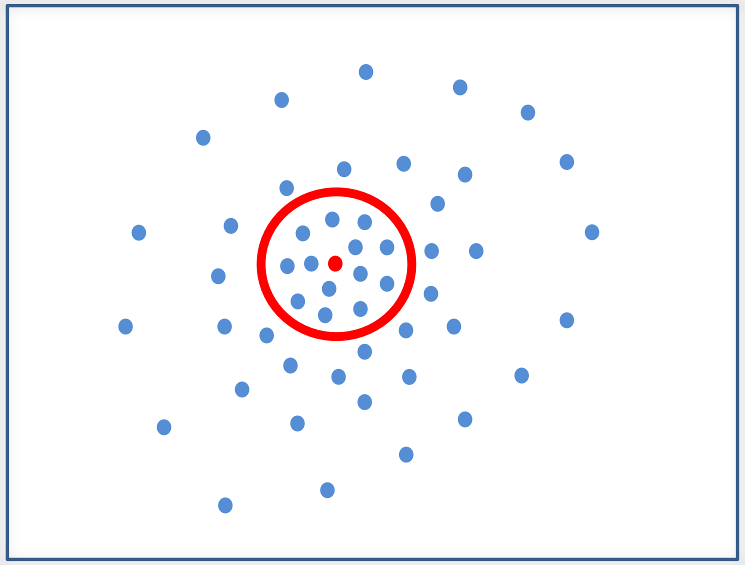
Intuition

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

Defining “High Density”

Naïve approach

For each point in a cluster there are at least a minimum number (MinPts) of points in an Eps-neighborhood of that point.

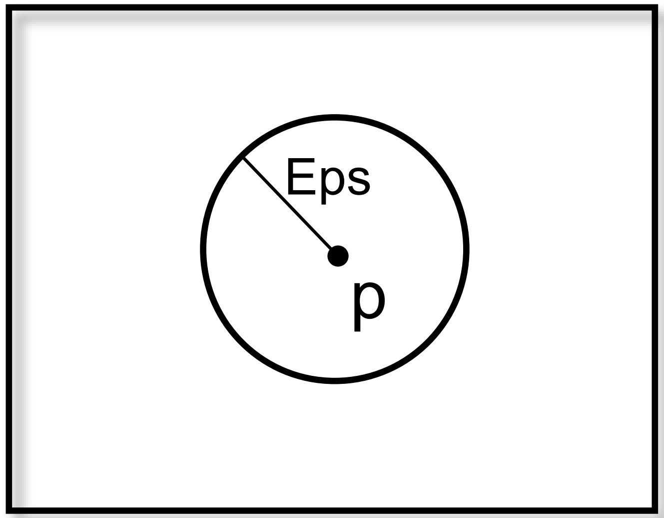


cluster

Defining "High Density"

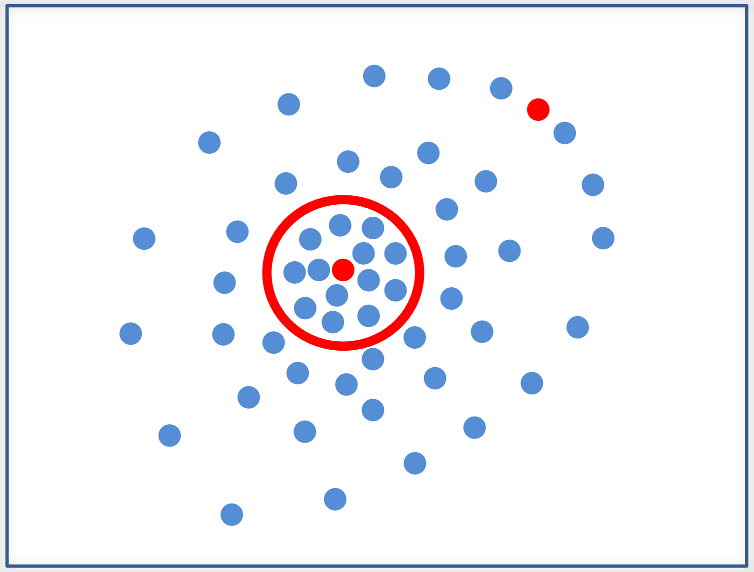
Eps-neighborhood of a point p

$N_{Eps}(p) = \{ q \in D \mid dist(p, q) \leq Eps \}$



Defining “High Density”

- In each cluster there are two kinds of points:
  - points inside the cluster (core points)
  - points on the border (border points)



cluster

An Eps-neighborhood of a border point contains significantly less points than an Eps-neighborhood of a core point.

Defining "High Density"

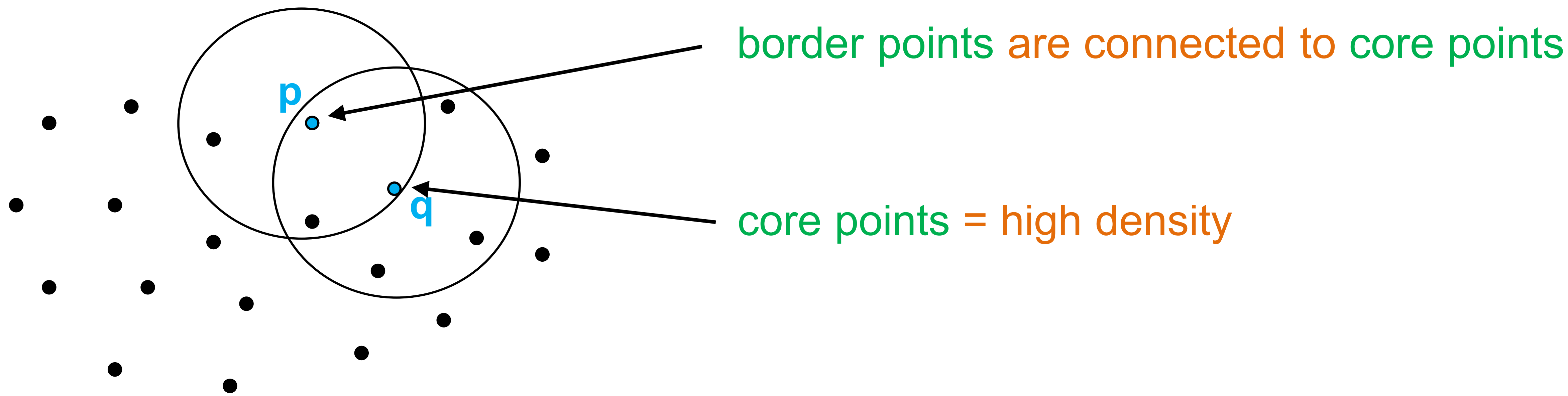
Better notion of cluster

For every point p in a cluster C there is a point $q \in C$, so that

(1) p is inside of the Eps-neighborhood of q

and

(2) $N_{Eps}(q)$ contains at least MinPts points.



Density Reachability

Definition

A point p is directly density-reachable from a point q with regard to the parameters Eps and MinPts, if

1) $p \in N_{Eps}(q)$ (reachability)

2) $| N_{Eps}(q) | \geq MinPts$ (core point condition)

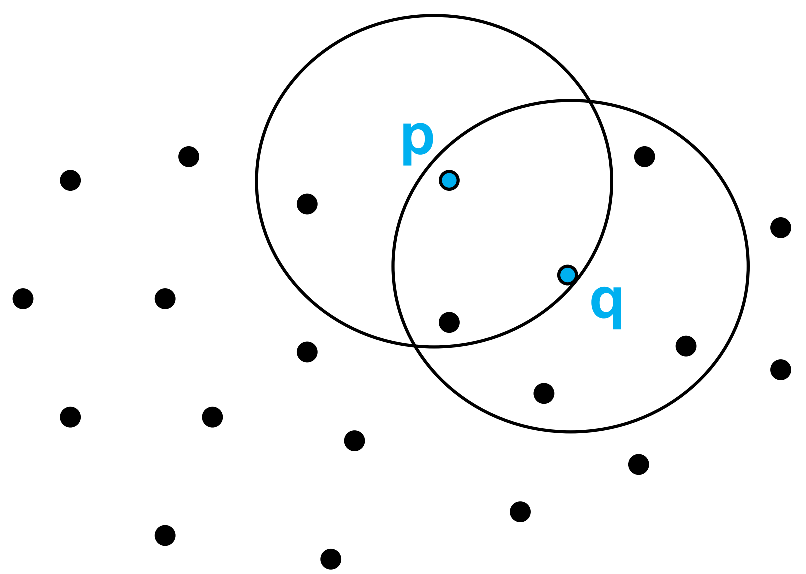
Density Reachability

Definition

A point p is directly density-reachable from a point q with regard to the parameters Eps and MinPts, if

1) $p \in N_{Eps}(q)$ (reachability)

2) $| N_{Eps}(q) | \geq MinPts$ (core point condition)



Parameter: MinPts = 5

p directly density reachable from q

$p \in N_{Eps}(q)$

$| N_{Eps}(q) | = 6 \geq 5 = MinPts$ (core point condition)

q **not** directly density reachable from p

$| N_{Eps}(p) | = 4 < 5 = MinPts$ (core point condition)

Note: This is an asymmetric relationship

Density Reachability

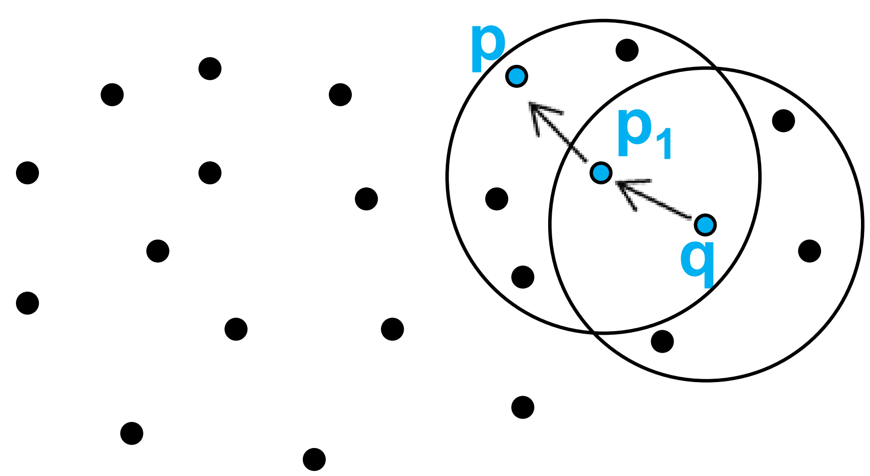
Definition

A point p is density-reachable from a point q

with regard to the parameters Eps and MinPts

if there is a chain of points $p_1, p_2, \ldots, p_s$ with $p_1 = q$ and $p_s = p$

such that p_{i+1} is directly density-reachable from p_i for all $1 < i < s-1$.



MinPts = 5

$|N_{Eps}(q)| = 5 = MinPts$ (core point condition)

$|N_{Eps}(p_1)| = 6 \geq 5 = MinPts$ (core point condition)

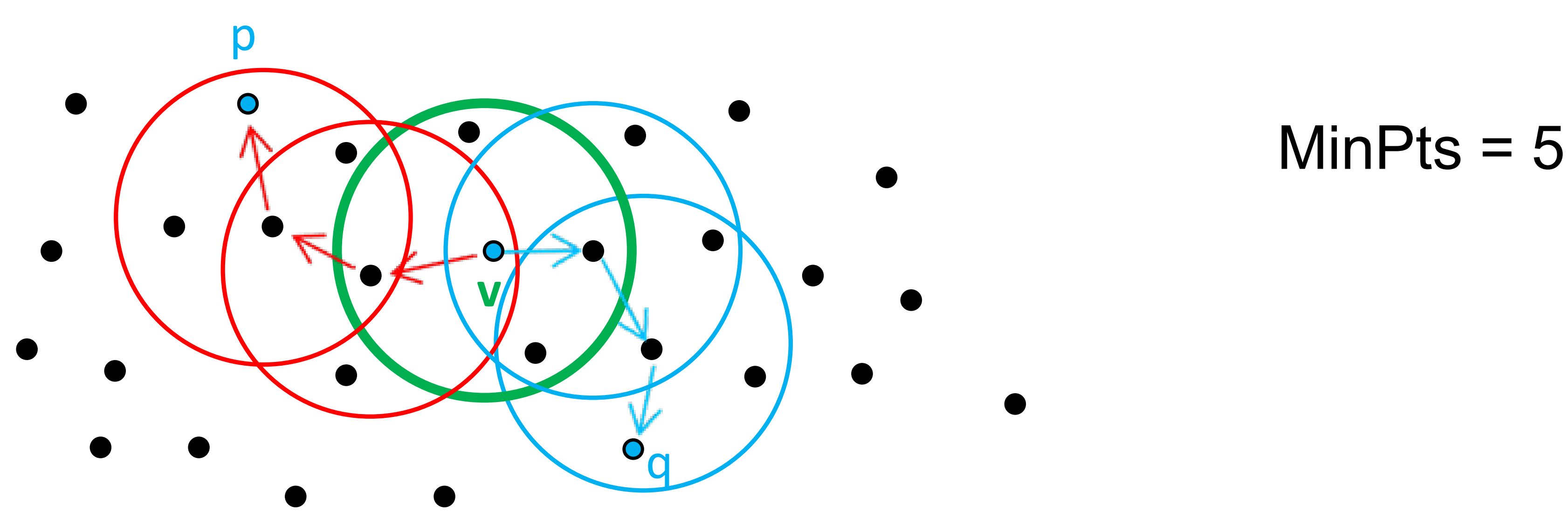
Density Connectivity

Definition (density-connected)

A point p is density-connected to a point q

with regard to the parameters Eps and MinPts

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



Note: This is a symmetric relationship

Definition of a Cluster

A cluster with regard to the parameters Eps and MinPts
is a non-empty subset C of the database D with

1) For all p, q ∈ D: (Maximality)
   If p ∈ C and q is density-reachable from p
   with regard to the parameters Eps and MinPts,
   then q ∈ C.

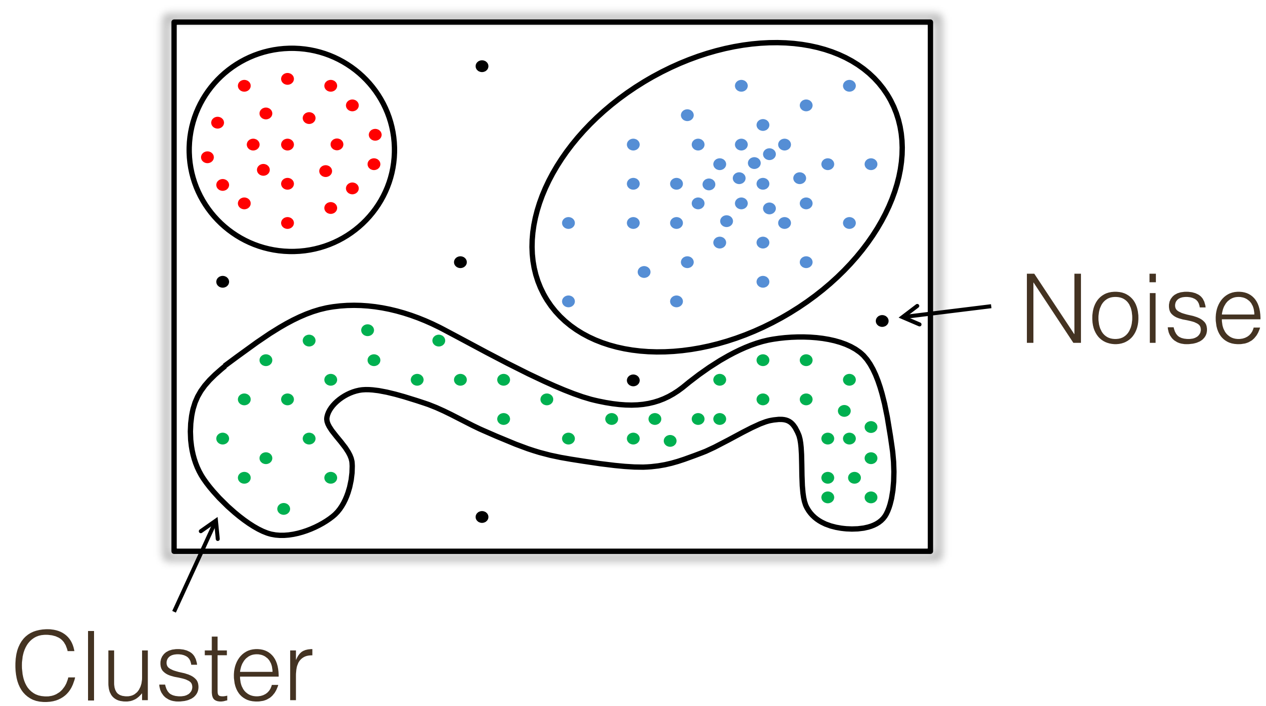
2) For all p, q ∈ C: (Connectivity)
   The point p is density-connected to q
   with regard to the parameters Eps and MinPts.

Definition of Noise

Let $C_1,\ldots,C_k$ be the clusters of the database D
with regard to the parameters Eps_i and $MinPts_l$ (i=1,...,k).

The set of points in the database D not belonging to any cluster $C_1,\ldots,C_k$
is called noise:

$$Noise = \{ p \in D \mid p \notin C_i \text{ for all } i = 1,\ldots,k\}$$



DBSCAN Algorithm

(1) Start with an arbitrary point p from the database and
retrieve all points density-reachable from p
with regard to Eps and MinPts.

DBSCAN Algorithm

(1) Start with an arbitrary point p from the database and
retrieve all points density-reachable from p
with regard to Eps and MinPts.

(2) If p is a core point, the procedure yields a cluster
with regard to Eps and MinPts
and all points in the cluster are classified.

DBSCAN Algorithm

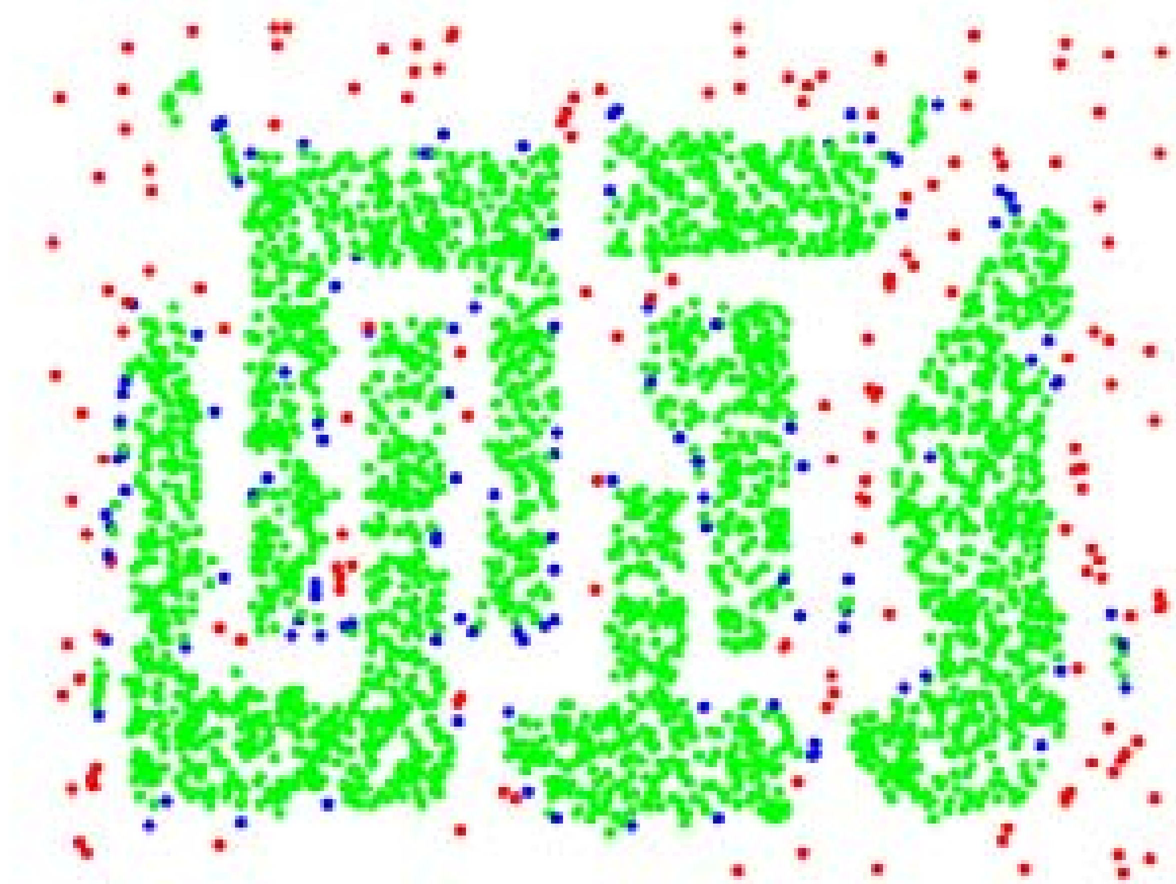
(1) Start with an arbitrary point p from the database and
retrieve all points density-reachable from p
with regard to Eps and MinPts.

(2) If p is a core point, the procedure yields a cluster
with regard to Eps and MinPts
and all points in the cluster are classified.

(3) If p is a border point, no points are density-reachable from p
and DBSCAN visits the next unclassified point in the database.

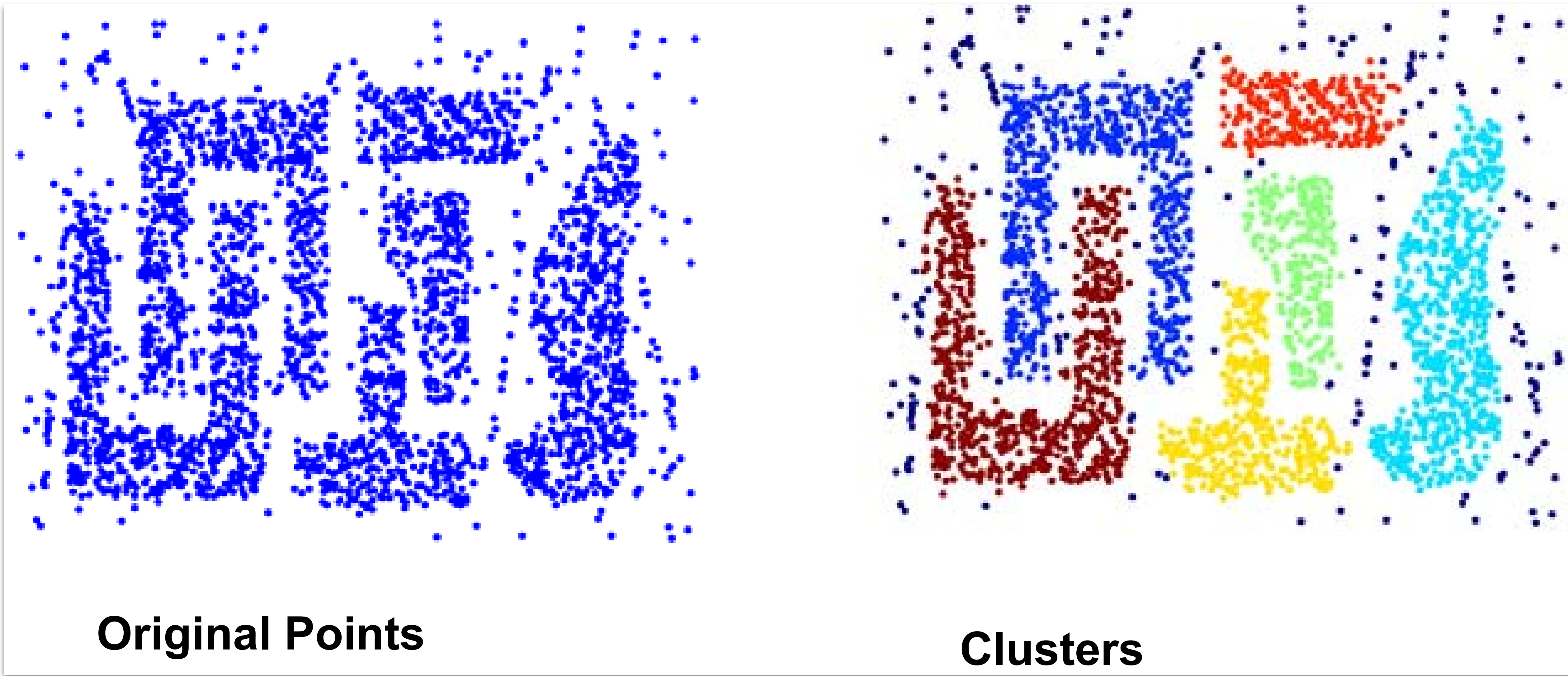
DBSCAN Algorithm

Original Points



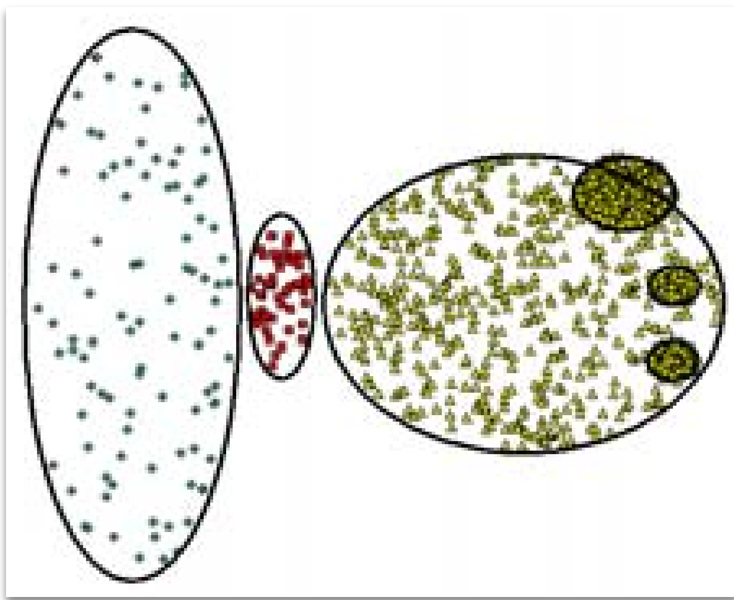
Point types: core, border and noise

DBSCAN strengths

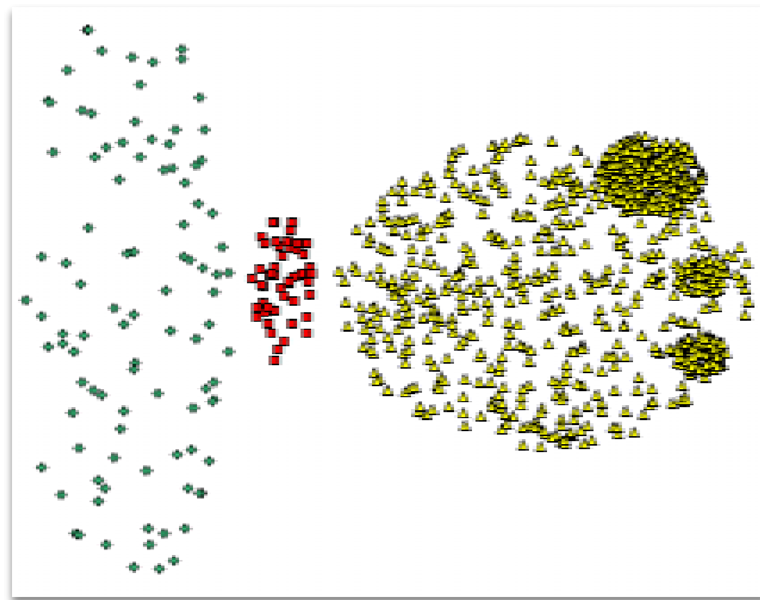


- Resistant to noise
- Can handle arbitrary shapes

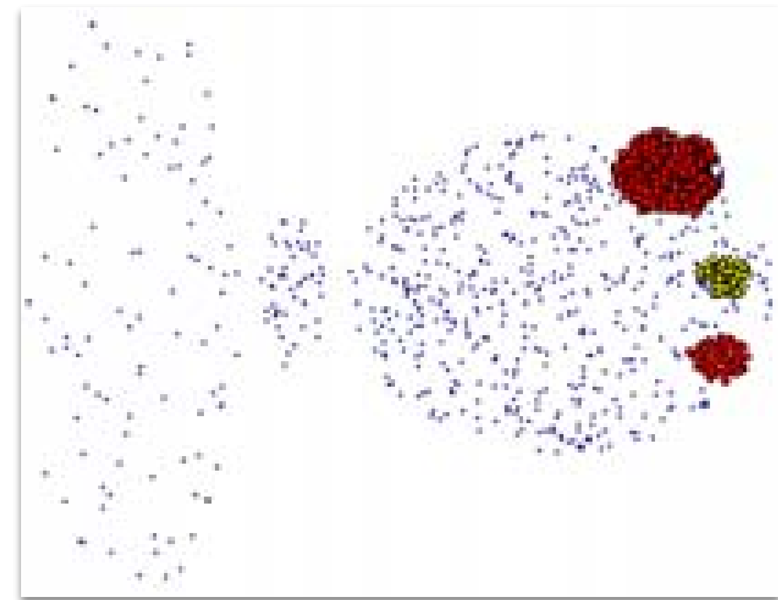
DBSCAN Weaknesses



Ground Truth



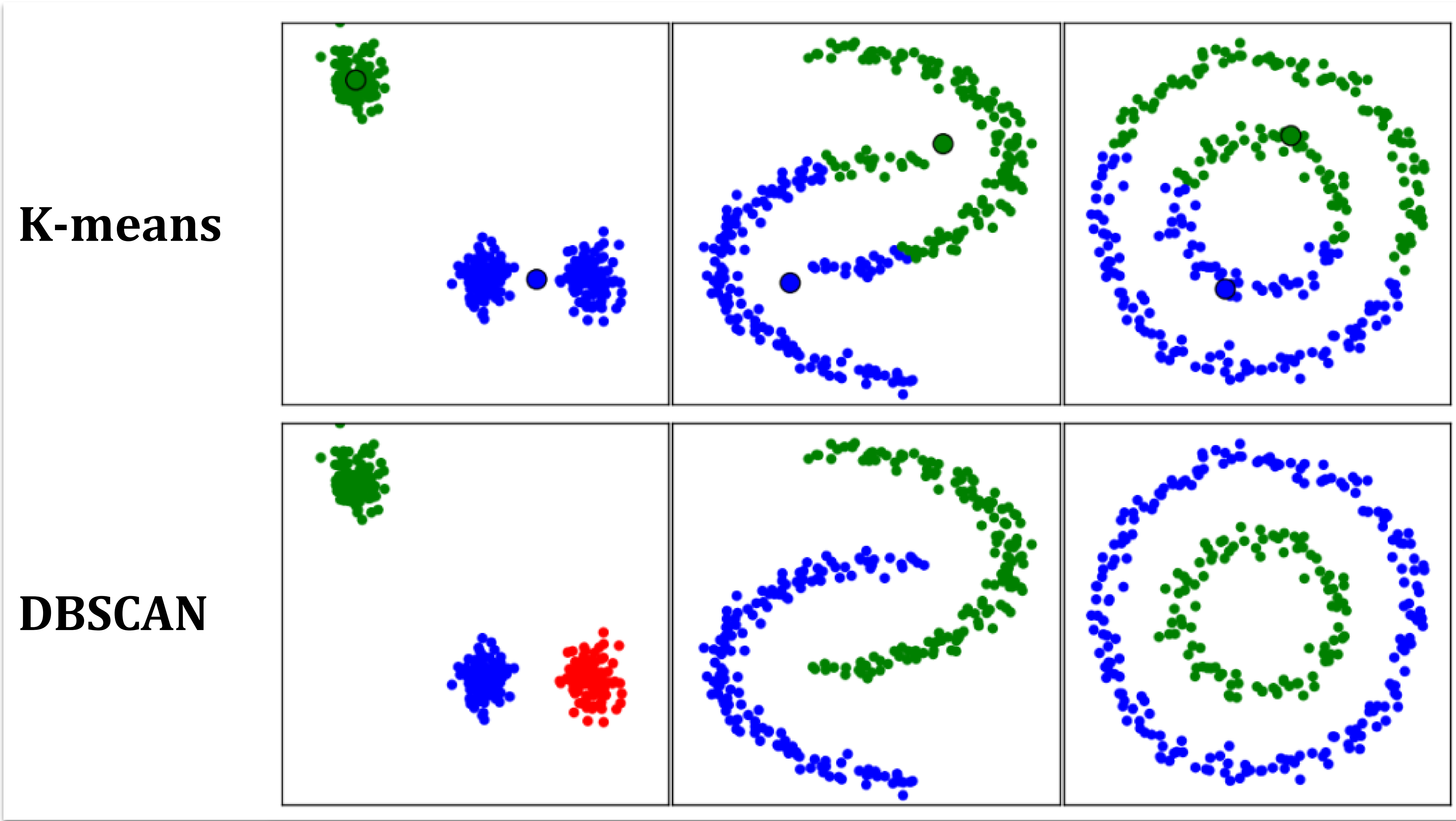
MinPts **= 4, Eps=9.92**



MinPts **= 4, Eps=9.75**

Sensitive to hyperparameters

K-means vs DBSCAN



Let’s see what it does with Trump’s tweets…