Introduction to Statistical Machine Learning CSC/DSCC 265/465

Lecture 11: Unsupervised Learning – Part II

Cantay Caliskan

Notes and updates

Notes and updates

- Hope all is well!
- **PS5 posted, deadline is Sunday, February 27, 11:59 PM**
  - Quick reminder: You are expected to code ***kmeans++*** for one of the algorithms
    - Not kmeans! ☺
  - Kmeans++ : One (1) convergence (to local minimum) is fine.

Notes on Grading

- Completeness (i), Correctness (ii), Format (iii), Academic Honesty (iv)

Completeness: Your answers will be checked for completeness. Specifically, for a question that requires several steps of thinking / writing / coding, we expect you to complete the full range of steps to answer the question. The range of steps that needs to be completed will be determined by the course material and the specific nature of the question.

Correctness: Your answers will be checked for correctness. For your answer to be correct, you need to have the correct answer, correct implementation, and the correct result. 'Correct' means that you follow the steps suggested by the assignment and your instructor and obtain the expected result without making any theoretical / mathematical / coding mistakes. 'Correctness' is not a binary term, there may be varying degrees of correctness; and, your grade will be evaluated based on how different your answer is from the expected result.

Format: An indispensable part of every assignment is the format. To make sure that your assignment can be read and processed easily, we expect you to follow the guidelines set by the instructor. These guidelines may include specific requirements about text-based answers, code files, and datasets.

Academic Honesty: We assignment that your submission fulfills the academic honesty expectations set by the instructor and put forward in the syllabus. Specifically, when expected, you need to produce work within the limits defined in the syllabus – some of the assignments may require you to work individually, and some others in a team. For more information, please read the syllabus.

Plan for the next lectures

- ***Kmeans***
- ***DBSCAN***
- ***OPTICS***
- ***GMM***
- ***PCA***

Plan for the next lectures

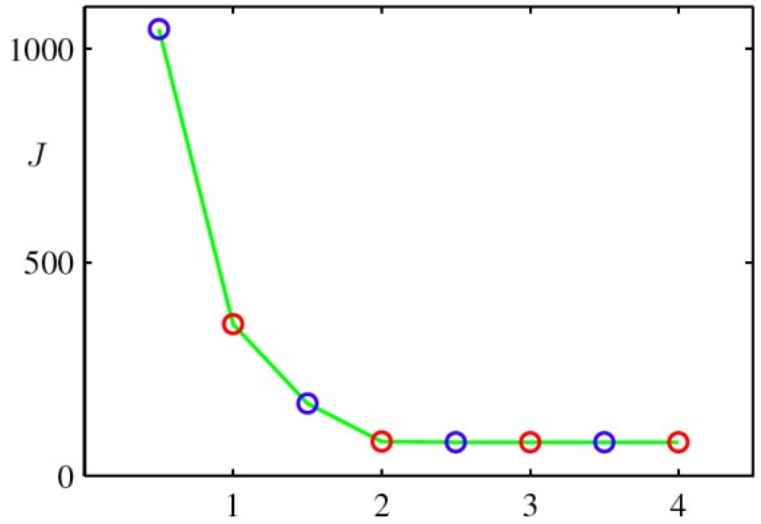
- ***Kmeans***
- *DBSCAN*
- *OPTICS*
- *GMM*
- *PCA*

Theory: Questions about K-Means

- **Why should the center of the points in a cluster be their mean?**
- **What if we used a different distance measure?**
- **How can we choose the best distance?**
- **How to choose K?**
- **How can we choose between alternative clusterings?**
- **Will the algorithm converge**
- **And: Hard cases, unequal spreads, non-circular spreads, in-between points**

Why does K-Means converge?

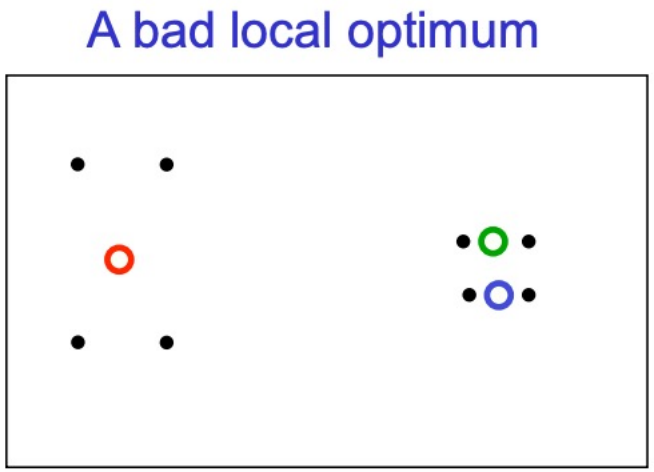
- ★ Whenever the set of points are re-assigned to a new cluster center, ***sum of squared distances (J)*** is reduced
- ★ Whenever a cluster center is moved, ***J*** is reduced
- ★ Test for convergence: If the assignments do not change in the assignment step, convergence has been obtained



- ★ Question: Do we converge to a local minimum or a global minimum?
- ★ Question: What can we do to prevent that?

Local Minima

- Cost function ***J*** is non-convex
  - Coordinate descent is not guaranteed to converge to the global minimum
- There is nothing to prevent ***K-Means*** from getting stuck at local minima
- Solutions:
  - Try many random starting points
  - Non-local splits and merges:
    - Simultaneously **merge** two nearby clusters
    - **Split** a big cluster into two
  - Think about smart ways to initialize!



K-Means++

- ★ Idea: Smart initialization!
- ★ Technically speaking: Consider all the points in the dataset to do the initialization

- ★ ***K-Means++*** algorithm:
  1) Pick the first center(s) randomly
  2) For all points x_i set d_i to be the distance to closest center
  3) Pick the new center to be at x_i with probability proportional to d_i^2
  4) Repeat steps 2+3 until you have *k* centers

Soft K-Means

- ★ Empirical idea: An apple can be **red** or **green**, but also red and green (!)

- ★ Instead of making ***hard*** assignments (=where clusters don't overlap with each other), we can make ***soft*** assignments:
  - ★ Example: One cluster may have a responsibility of 0.7 for a feature vector, another may have a responsibility of 0.3

- ★ Also called ***fuzzy clustering***

- ★ Questions:
  - ★ What happens to convergence guarantee?
  - ★ How do we decide on the soft assignments?

Soft K-Means

- How does ***Soft K-Means*** work?
  - Choose a number of clusters
  - Assign coefficients (weights) randomly to each data point for being in the clusters
  - Repeat until the algorithm has converged (coefficients' change between two iterations is no more than an epsilon-value)
    - Compute the centroid for each cluster
    - For each data point, compute its coefficients of being in the clusters
- Observation: This is very similar to K-Means!

Soft K-Means

★ How does ***Soft K-Means*** work?

- Centroid: $$c_k = \frac{\sum_x w_k(x)^m x}{\sum_x w_k(x)^m}$$

$W_k(x)$ represents the degree of belonging to cluster *k*

- Objective function: $$\arg\min_C \sum_{i=1}^{n}\sum_{j=1}^{c} w_{ij}^m \|\mathbf{x}_i - \mathbf{c}_j\|^2$$

- Coefficients: $$w_{ij} = \frac{1}{\sum_{k=1}^{c}\left(\frac{\|\mathbf{x}_i-\mathbf{c}_j\|}{\|\mathbf{x}_i-\mathbf{c}_k\|}\right)^{\frac{2}{m-1}}}$$

m is the fuzziness parameter

Each data point belongs to each cluster at a degree of $\boldsymbol{w_{ij}}$

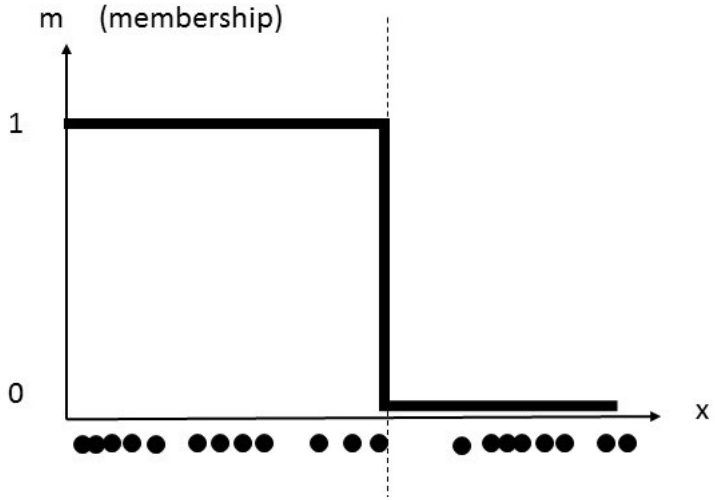
Question: What is the range of ***m***?

Answer: (1, ∞)

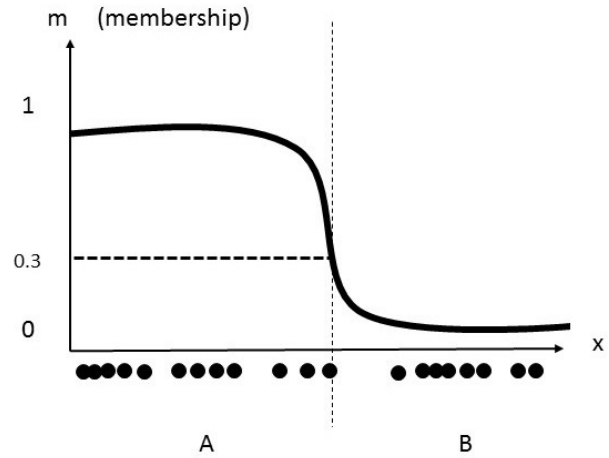
Soft K-Means vs. Hard K-Means

- Let's say we are applying a clustering algorithm on a ***1-dimensional*** dataset where ***k = 2***:

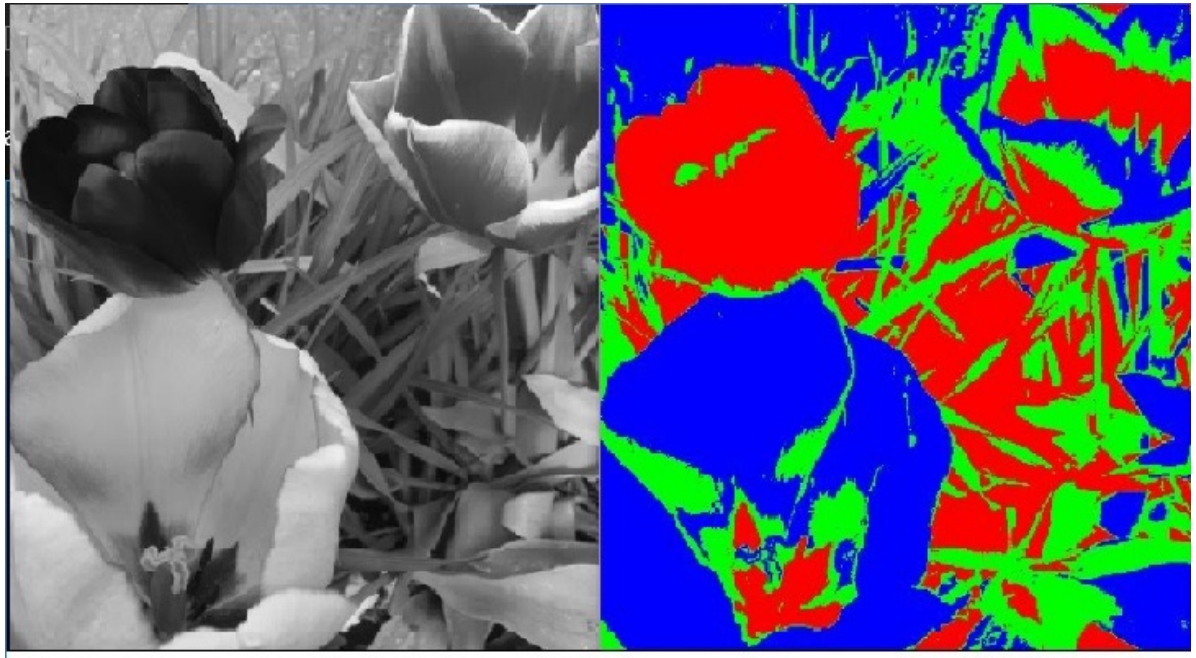
- Membership in ***hard K-Means***:

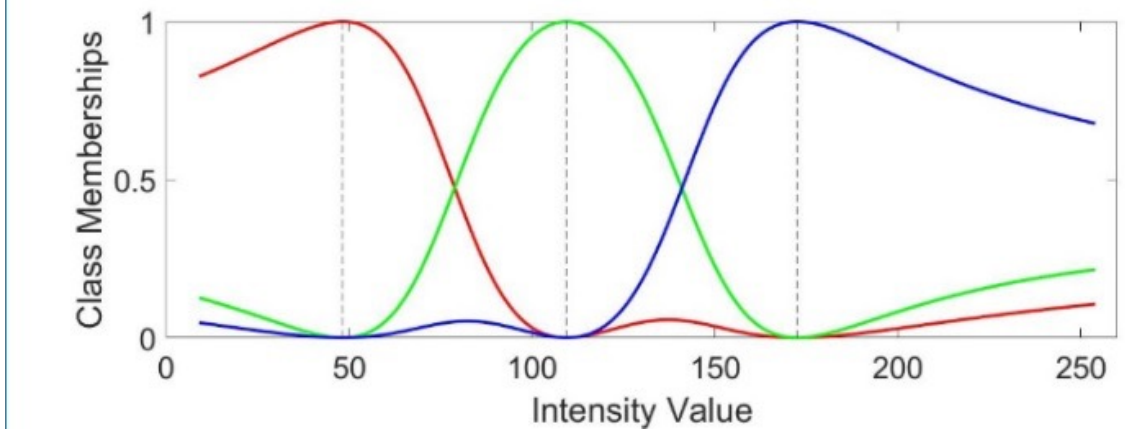


- Membership in ***soft K-Means***:



Soft K-Means Application





Selecting an optimal number of clusters

Quantifying the errors in clustering

- Different options are possible!
- One way to evaluate the choice of K is made using a parameter known as ***WCSS***: ***Within Cluster Sum of Squares***
- = ***sum*** of variances within each cluster

$$WCSS = \sum_{1}^{K} \sum_{1}^{j} \|x_i - c_k\|$$

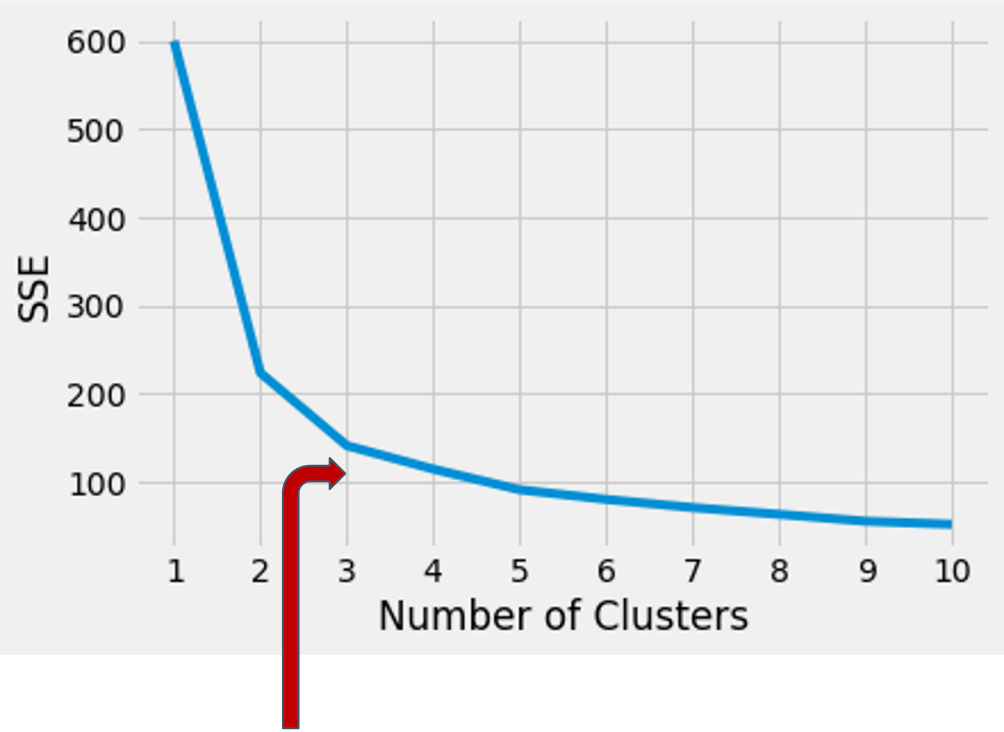
- This is called a ***cost*** function
- Usually denoted by ***J()***

Elbow Method

Less is More!

- **Step 1**: Run the algorithm for different values of K
- **Step 2**: Stop when there is not significant reduction of error anymore

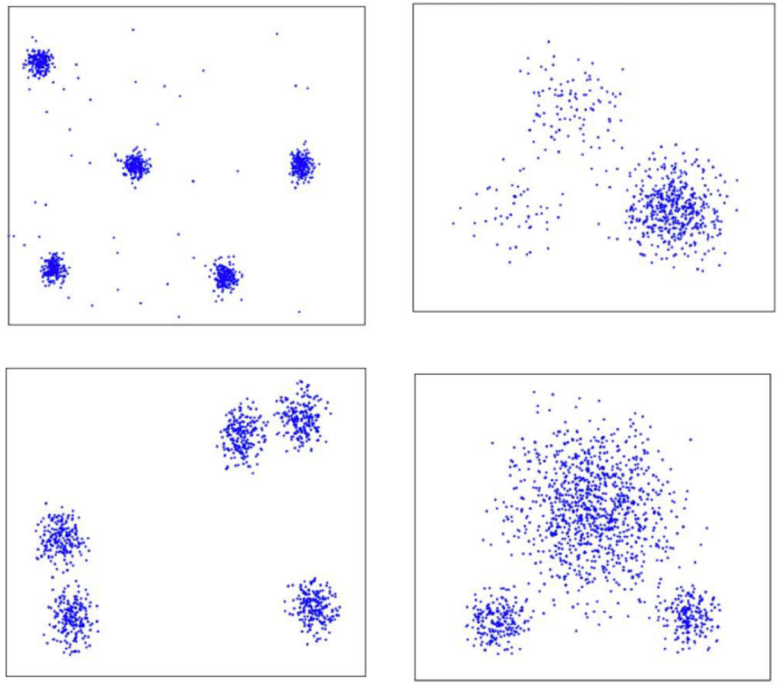
It is easier to interpret the result with fewer of number of clusters!



Here you can stop at ***K = 3*** :)

External Validation Metrics

- Quality control can be made by using:
  - ***External Validation Metrics***
    - Uses no internal information



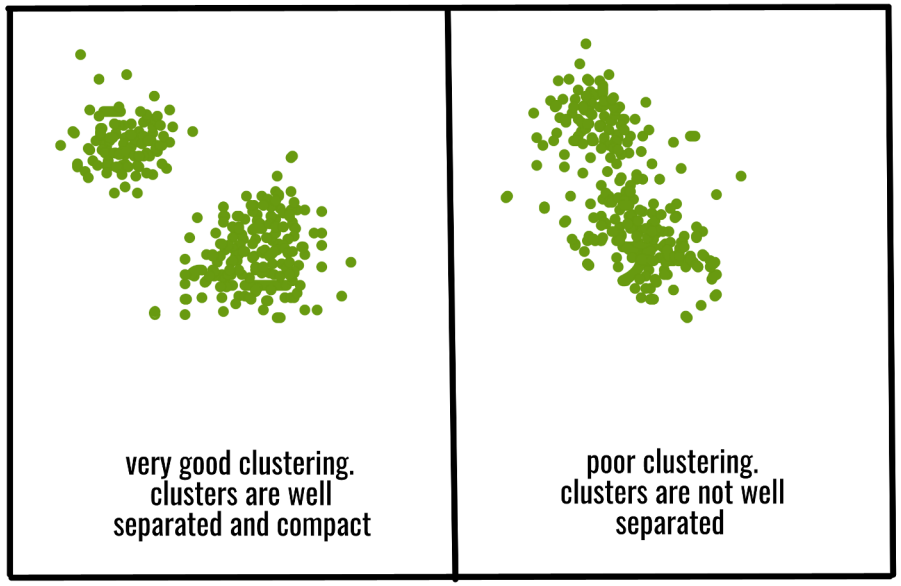
- <u>Reminder</u>: We want to group similar points
- = **Greater** distance between ***dissimilar*** points
- = **Smaller** distance between ***similar*** points

<u>Key Criteria:</u>

- **Compactness**
- **Separation**

Compactness and Separation

- **Compactness**
  - Within cluster distance -> minimized
  - Within cluster variance -> minimized
- **Separation**
  - Between cluster distance -> maximized
  - Between cluster variance -> maximized
- **Compactness + Separation**

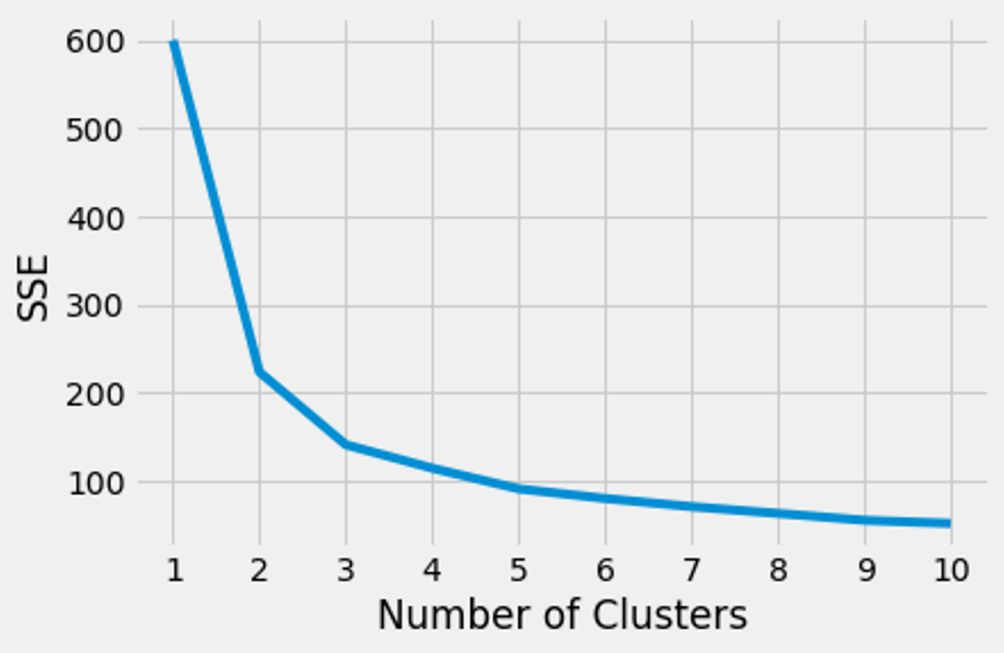


We may want to achieve at least two at the same time.

Calinski-Harabasz Index, Silhouette Score, Davies-Bouldin Index

Choosing the right *K*

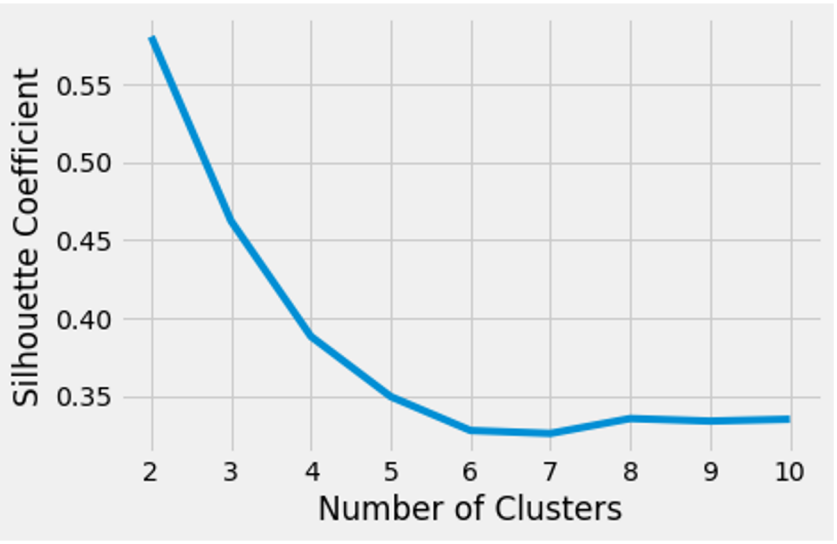
- Dataset on Iris flowers with <u>three (3)</u> different types of species
  - ★ **Theory: 3**



SSE curve says ***3***, as well!

Choosing the right *K*

- Dataset on Iris flowers with <u>three (3)</u> different types of species



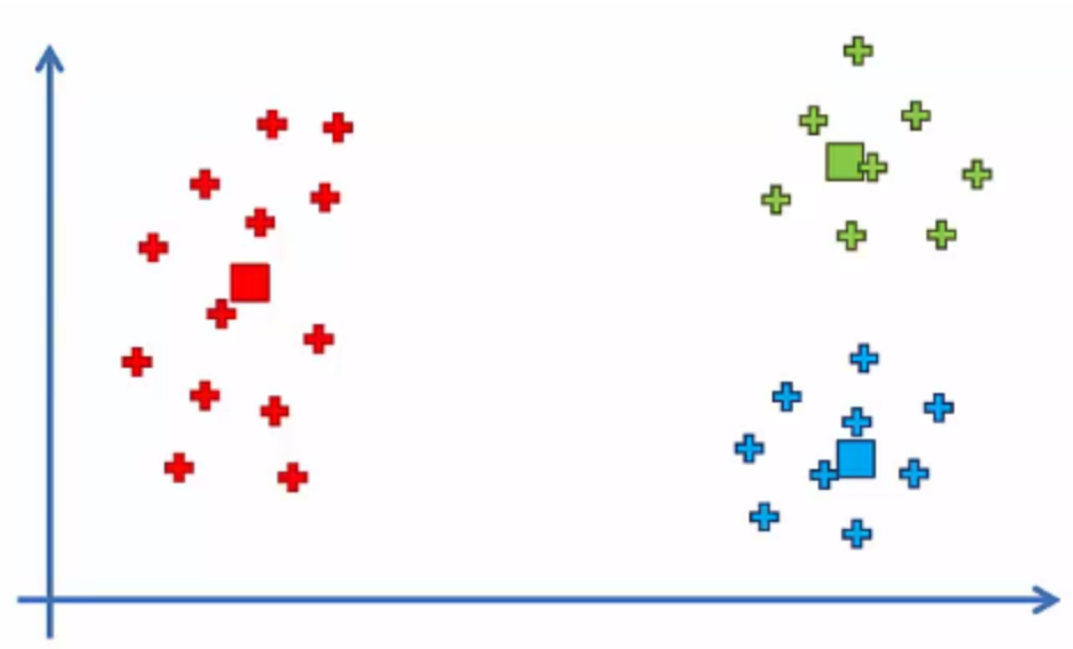
Silhouette -> 2 clusters

Calinski-Harabasz -> 2 clusters

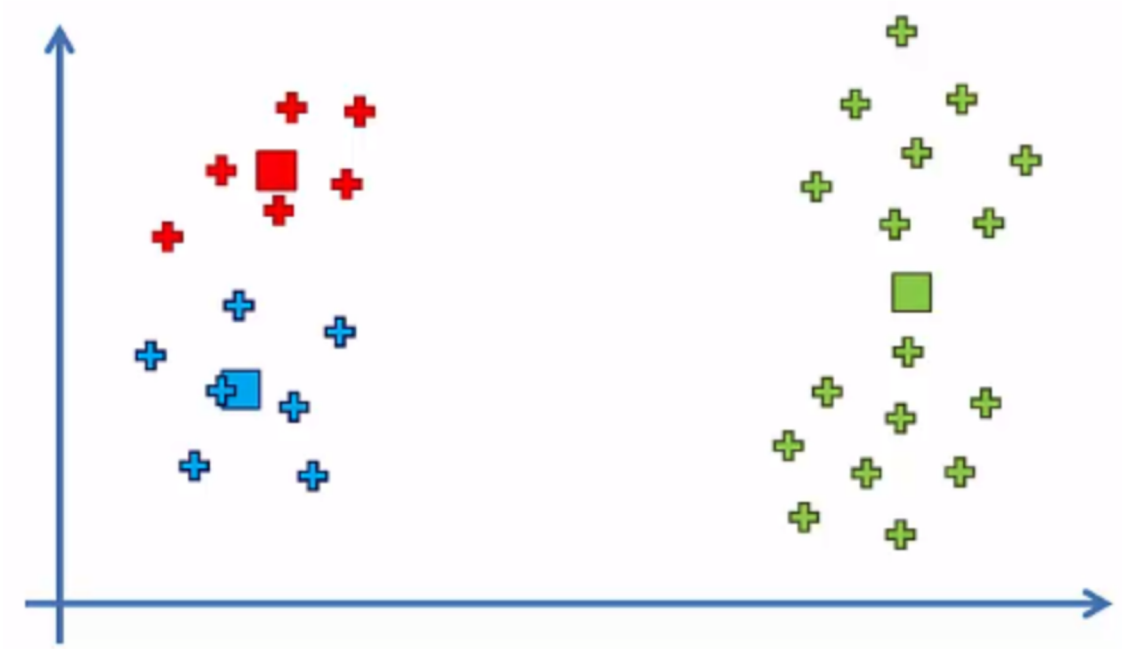
Davies-Bouldin -> 2 clusters

Drawbacks

Effects of Bad Initialization



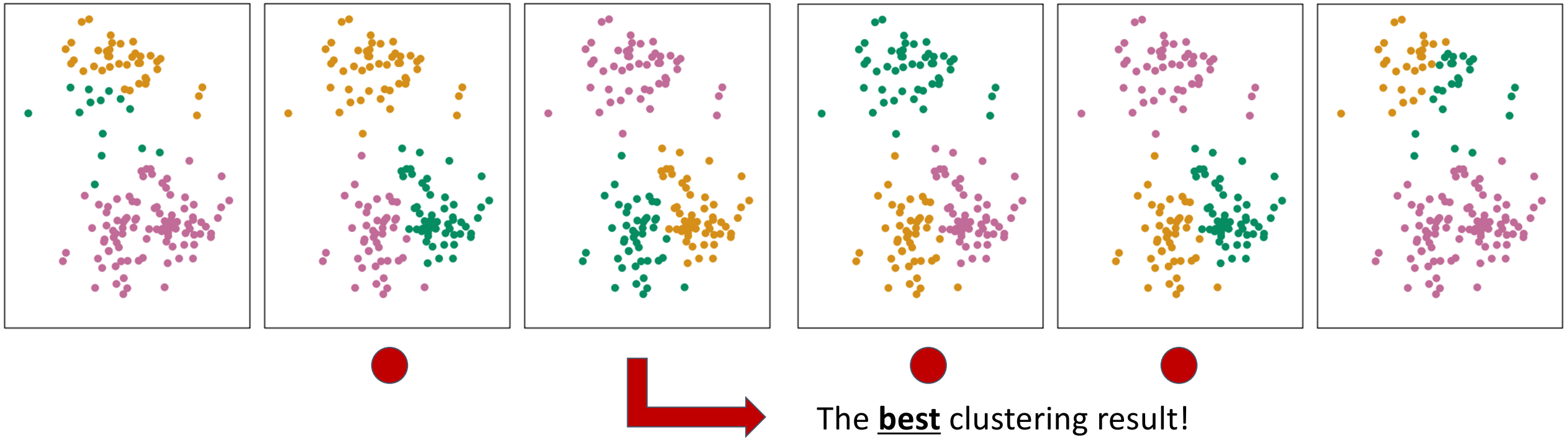
Good Initialization!



Bad Initialization!

Effects of Random Initialization

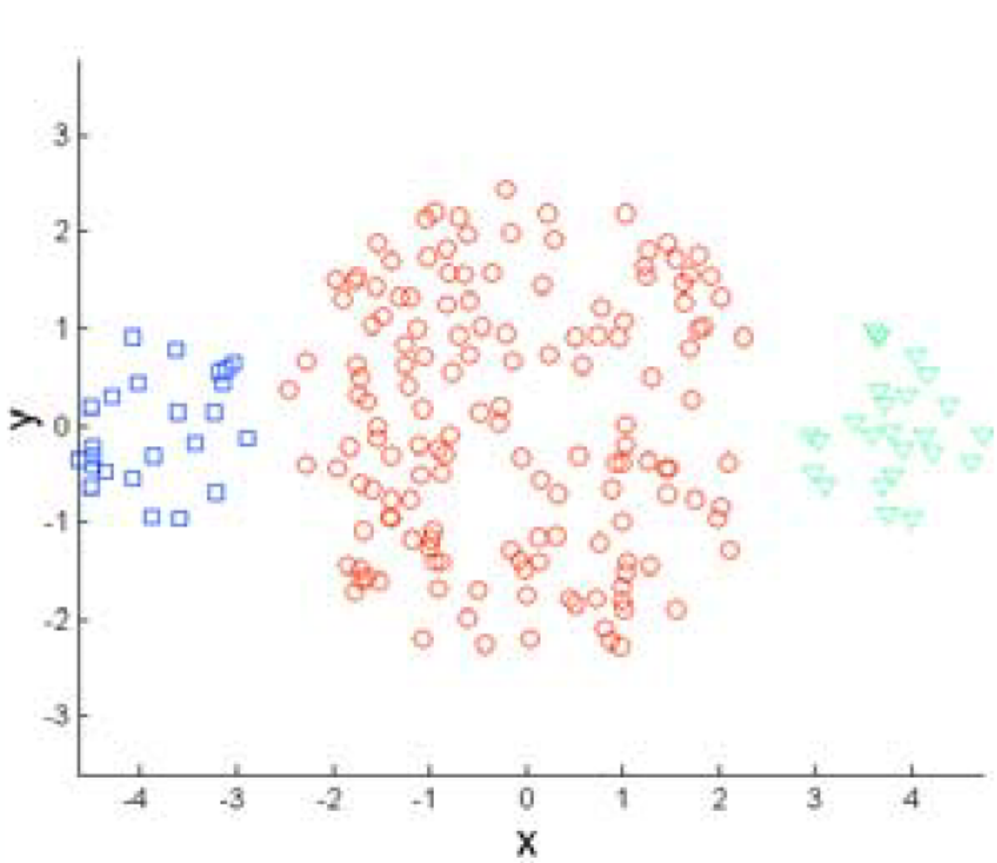
K = 3, same dataset:



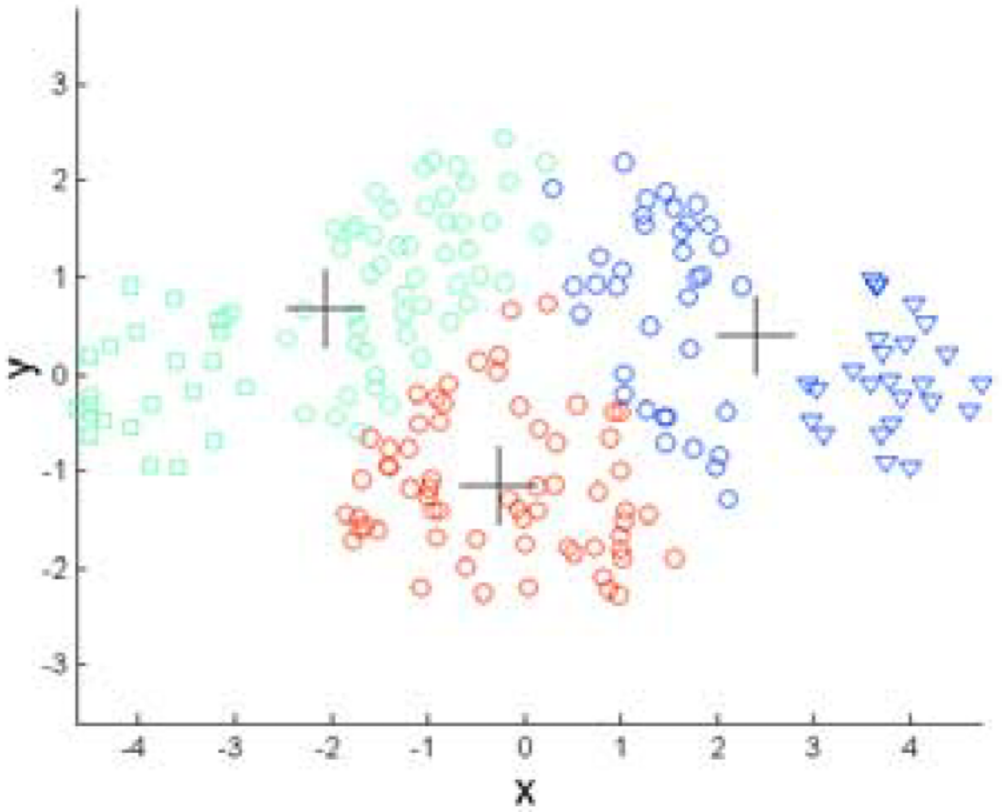
The **best** clustering result!

Source: The Elements of Statistical Learning (Hastie, Tibshirani, Friedman, 2009)

Limitations of K-Means: Different Sizes

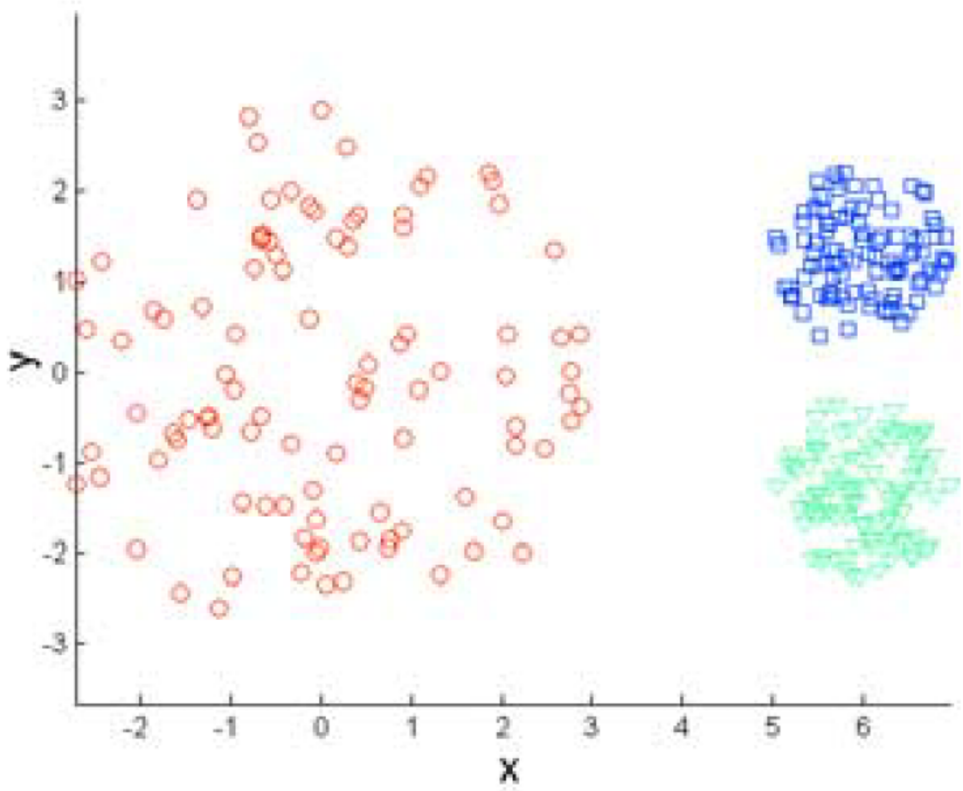


Original Points

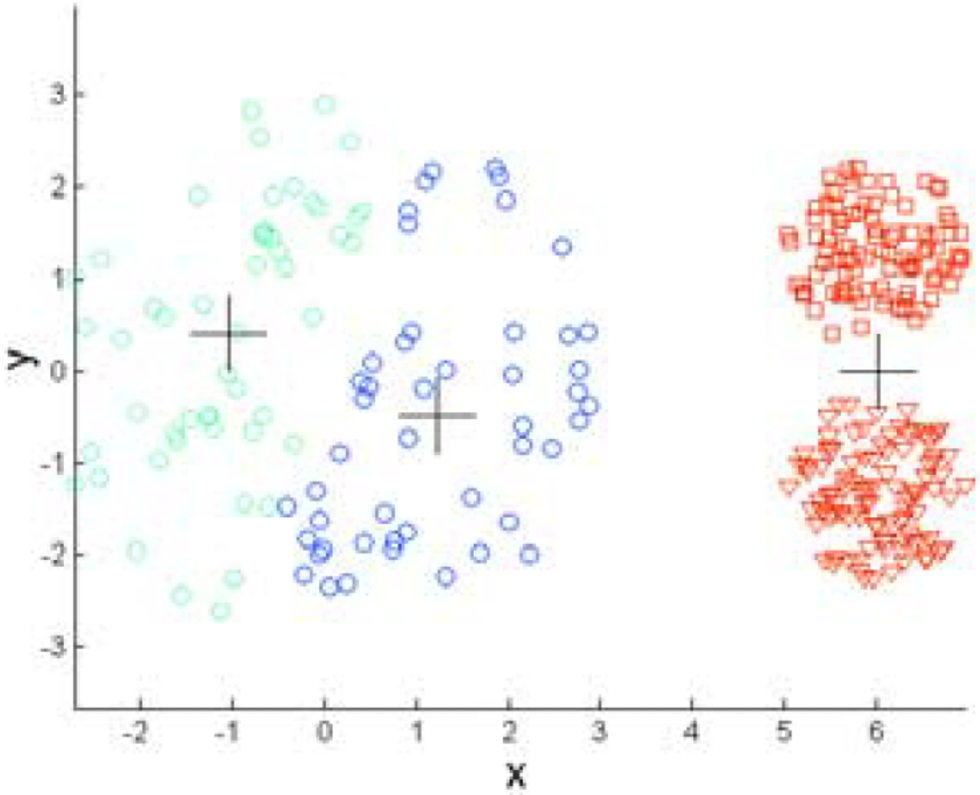


K-means (3 Clusters)

Limitations of K-Means: Different Density

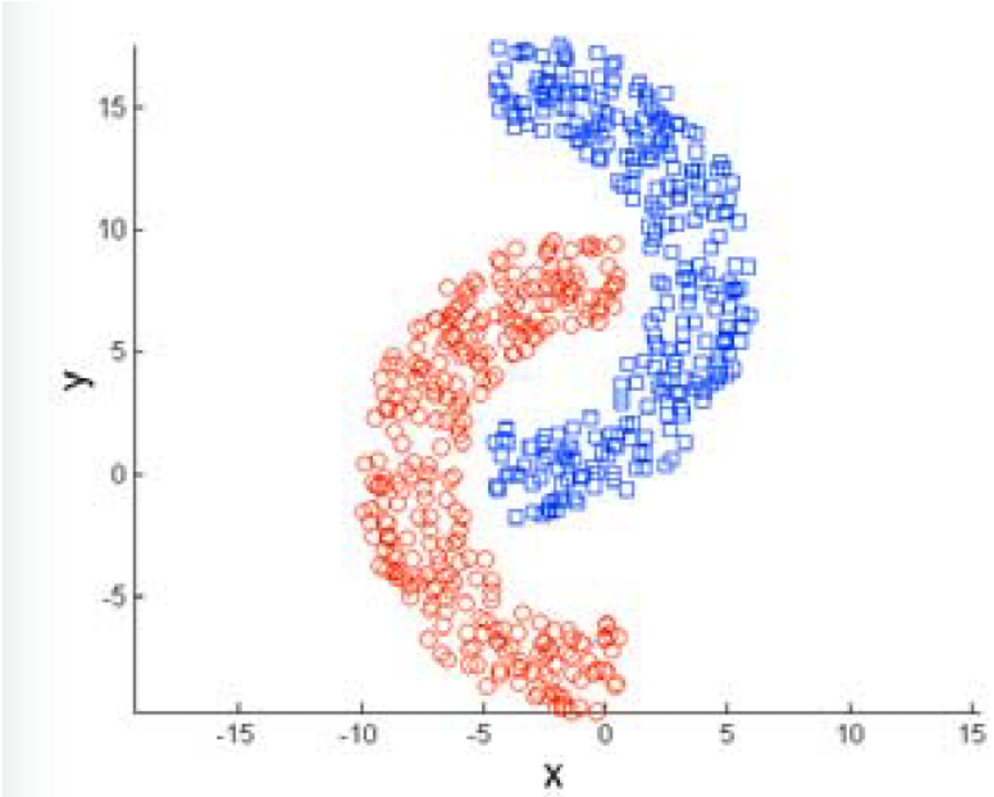


Original Points

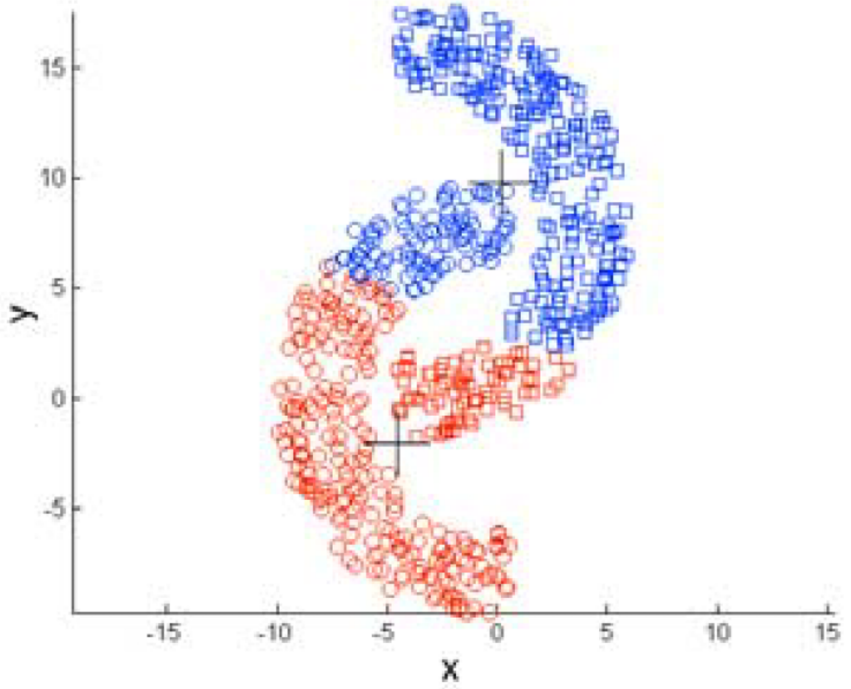


K-means (3 Clusters)

Limitations of K-Means: Non-Spherical Shapes



Original Points



K-means (2 Clusters)

Example: Effect of Different Initializations

https://www.youtube.com/watch?v=9nKfViAfajY

Discussion on the K-Means algorithm

- Finds a ***local*** optimum
- Often ***converges*** quickly
  - But not always!
- The choice of the ***initial points*** can have large influence on the result
- Tends to find ***spherical clusters***
- ***Outliers*** can cause a problem
- Different ***densities*** may cause a problem

Fixes: Pre-Processing

- **Removing the outliers**
  - Highly recommend the `pyod` library of `Python`
- **Scaling the data**
  - Scaling between 0 and 1
  - Standard scaling
- **Stratified sampling**
  - Reduce the density of overly represented points
- **Doing some research**
  - What do the others say?

Plan for the next lectures

- *Kmeans*
- ***DBSCAN***
- *OPTICS*
- *GMM*
- *PCA*

Density-based clustering

Density-Based Clustering Methods

- Most well-known one:
  - **DBSCAN**
  - ***"Density-based spatial clustering of applications with noise"***

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

- Major features:
  - Discover clusters of arbitrary shape
  - Handle noise

DBSCAN: Three types of points

- **You need two parameters: minPts and Eps**

- **Core points**: Interior points of a density-based cluster. A point p is a core point if for distance ***Eps***:

- **Border points**: Not a core point, but within the neighborhood of a core point (it can be in the neighborhoods of many core points)

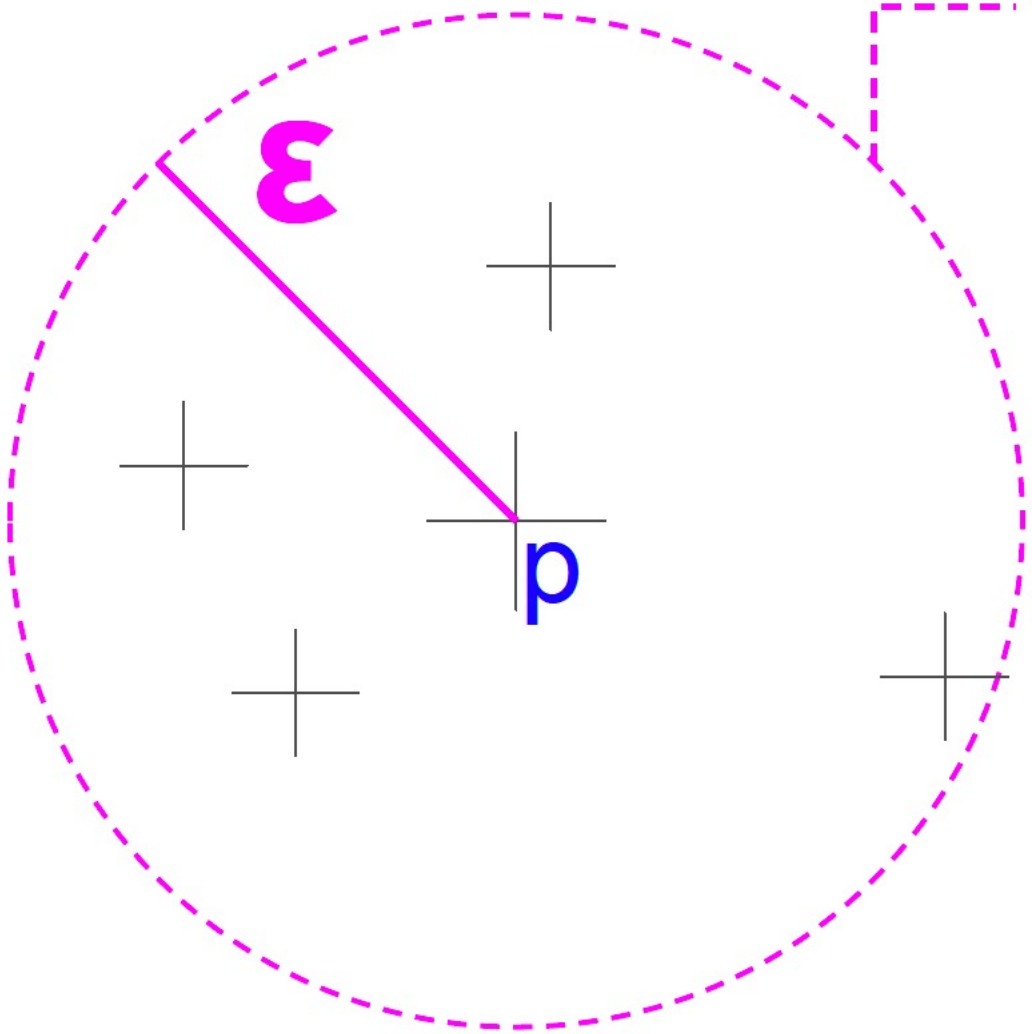
- **Noise points**: Not a core or a border point

DBSCAN: The Algorithm

- Label all points as core, border, or noise points
- Eliminate noise points
- Assign a relationship between all core points that are within **Eps** of each other
- Make each group of connected core points into a separate cluster
- Assign each border point to one of the cluster of its associated core points

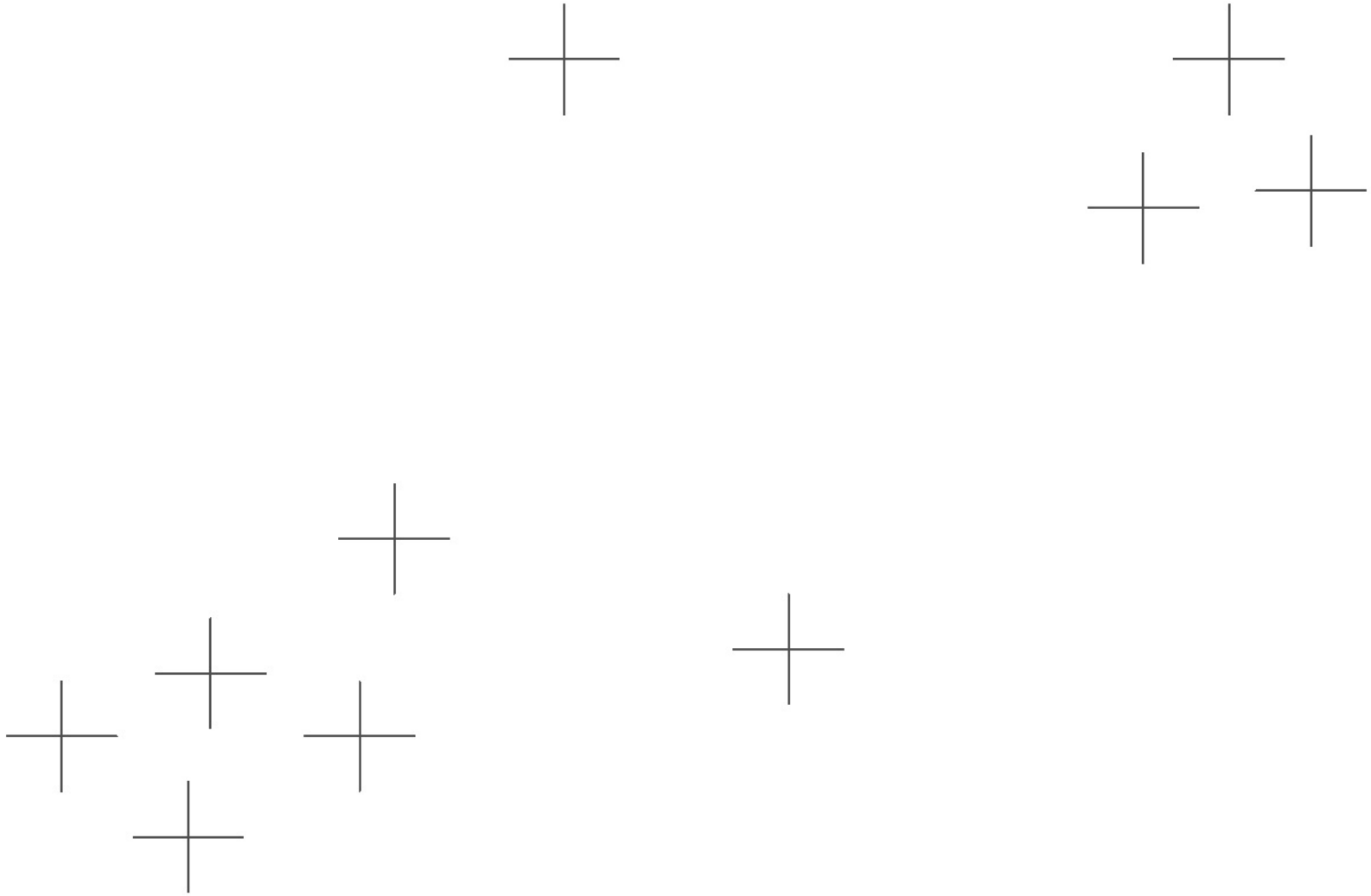
ε**p**silon **m**inPts

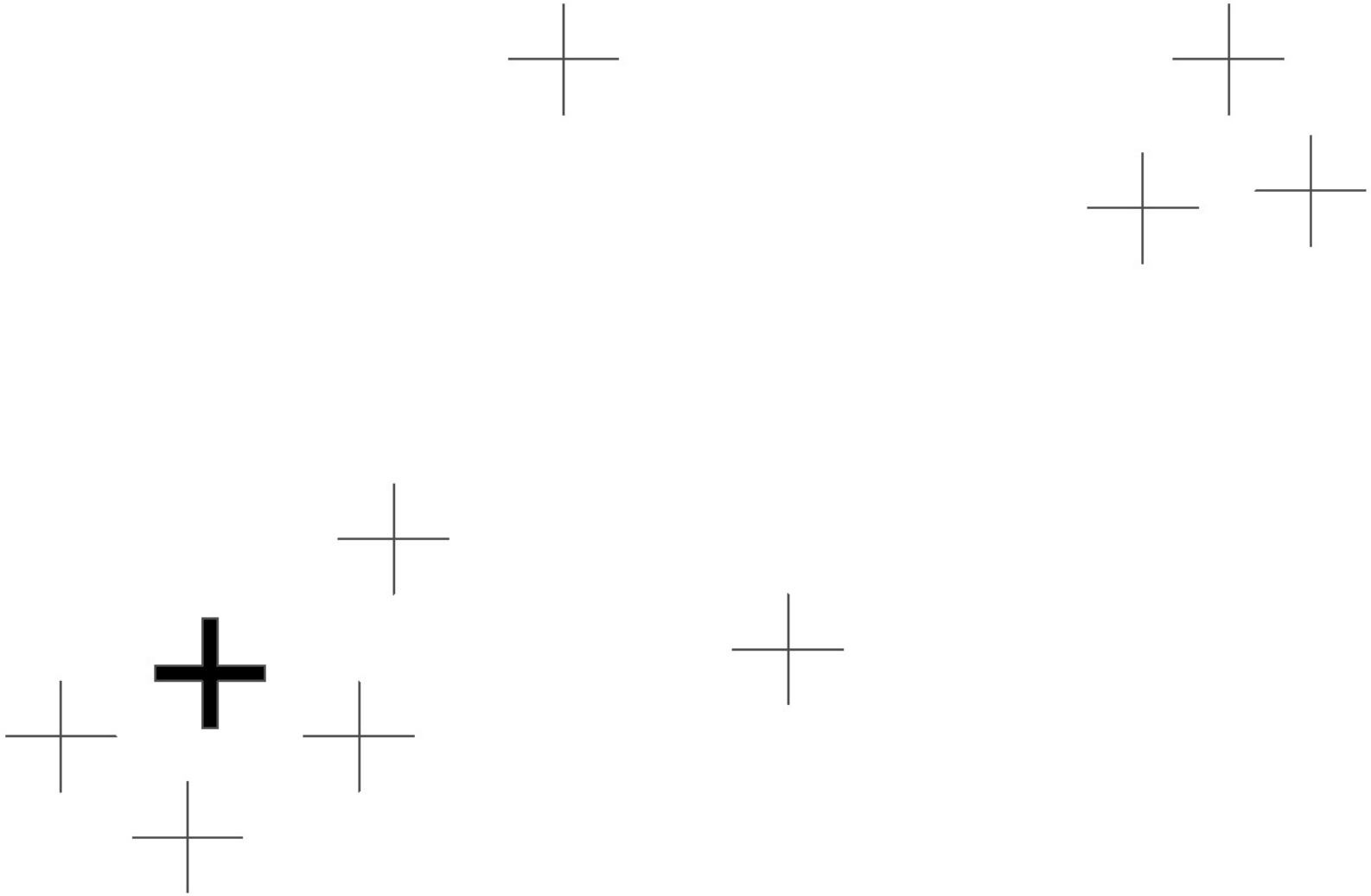
no need to specify # of clusters

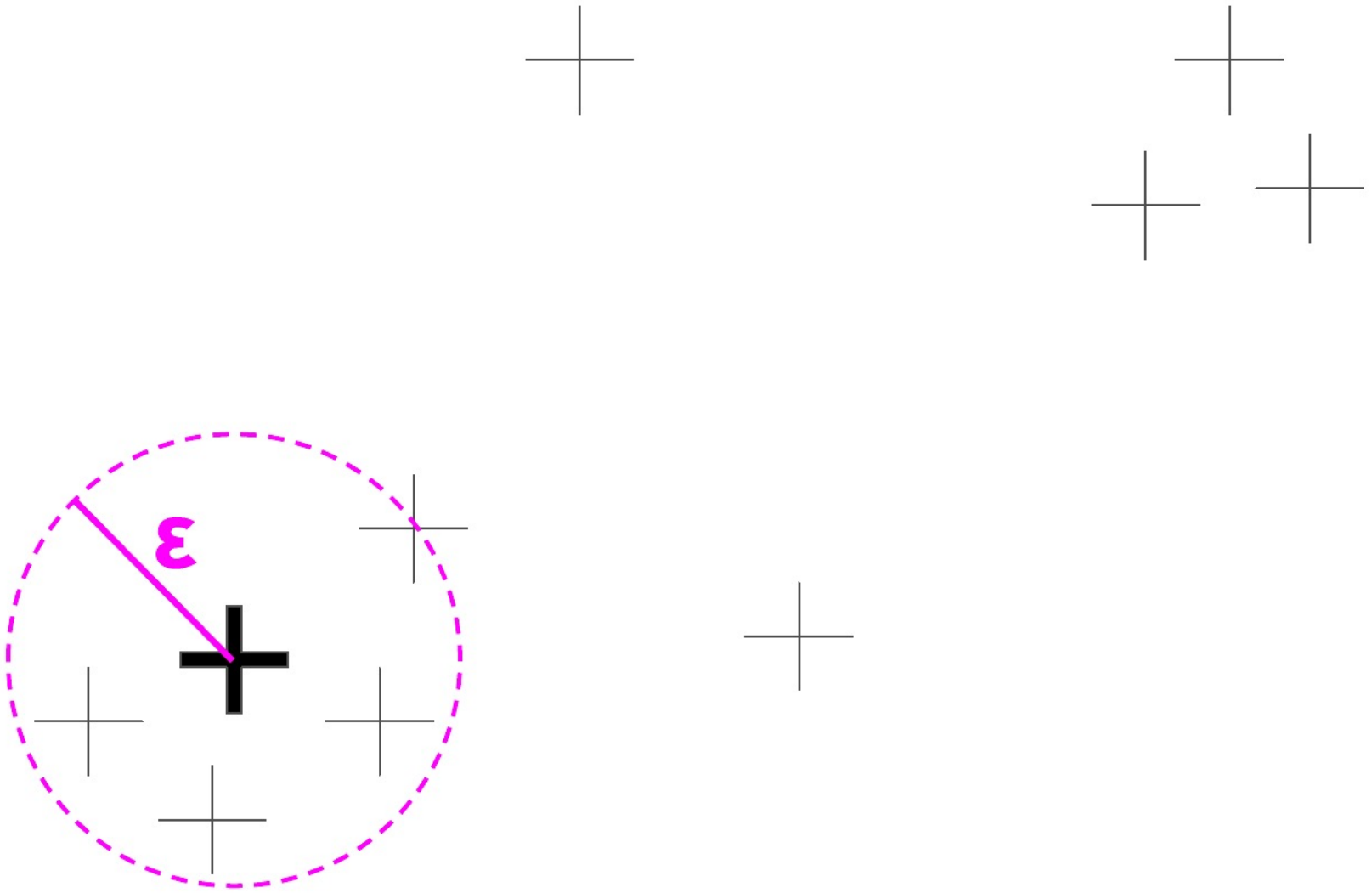


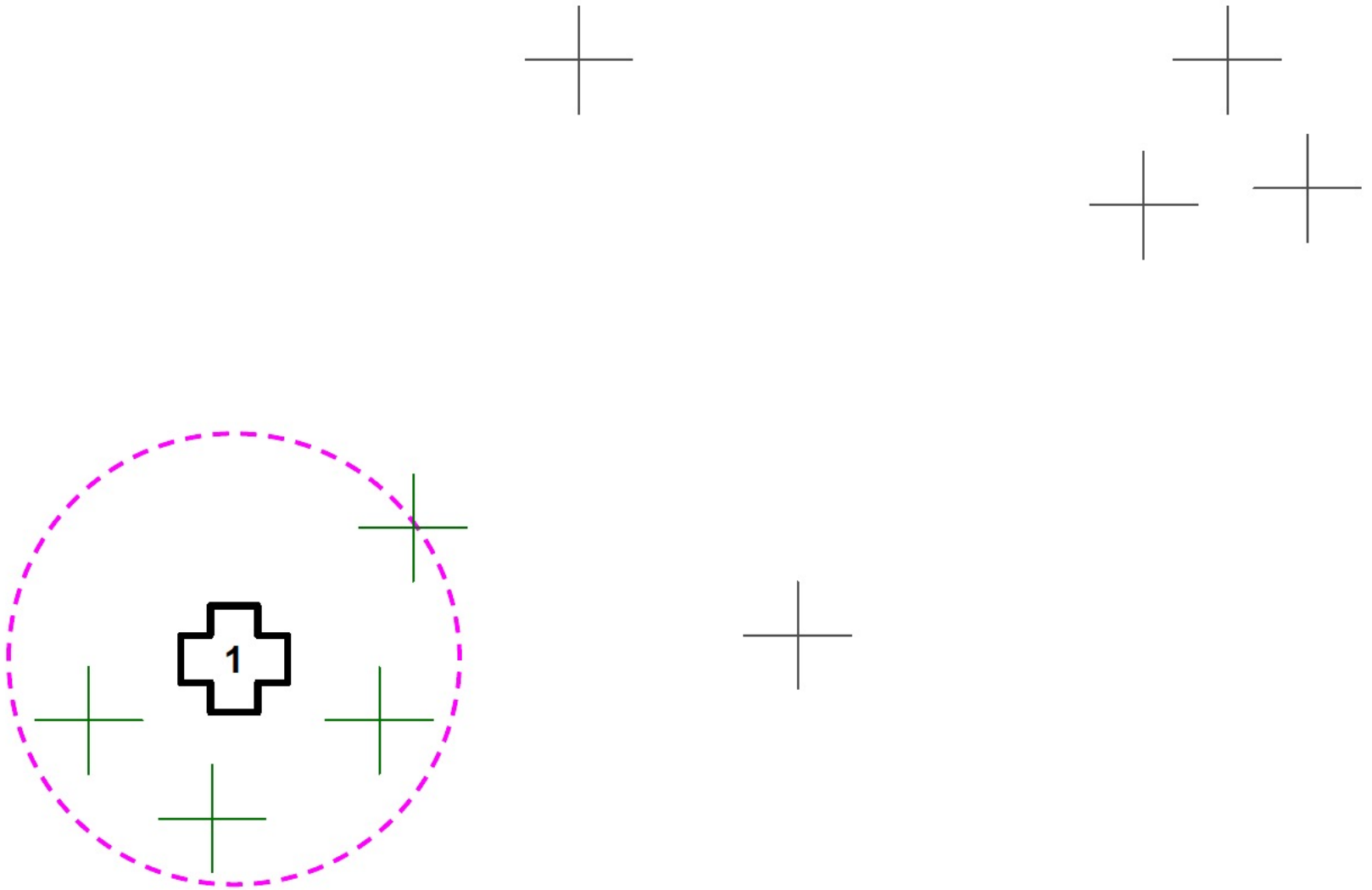
≥minPts

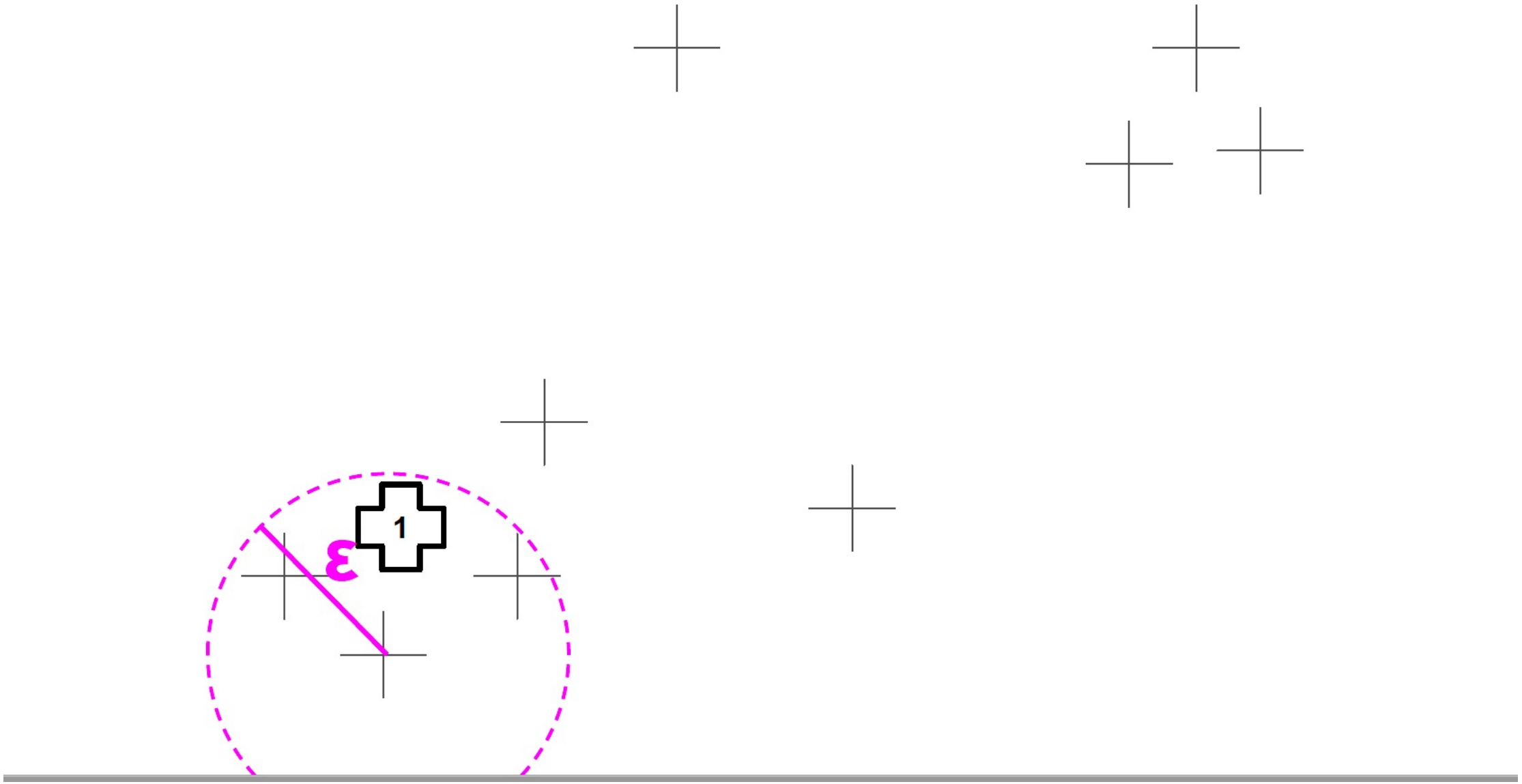
p is **core** point



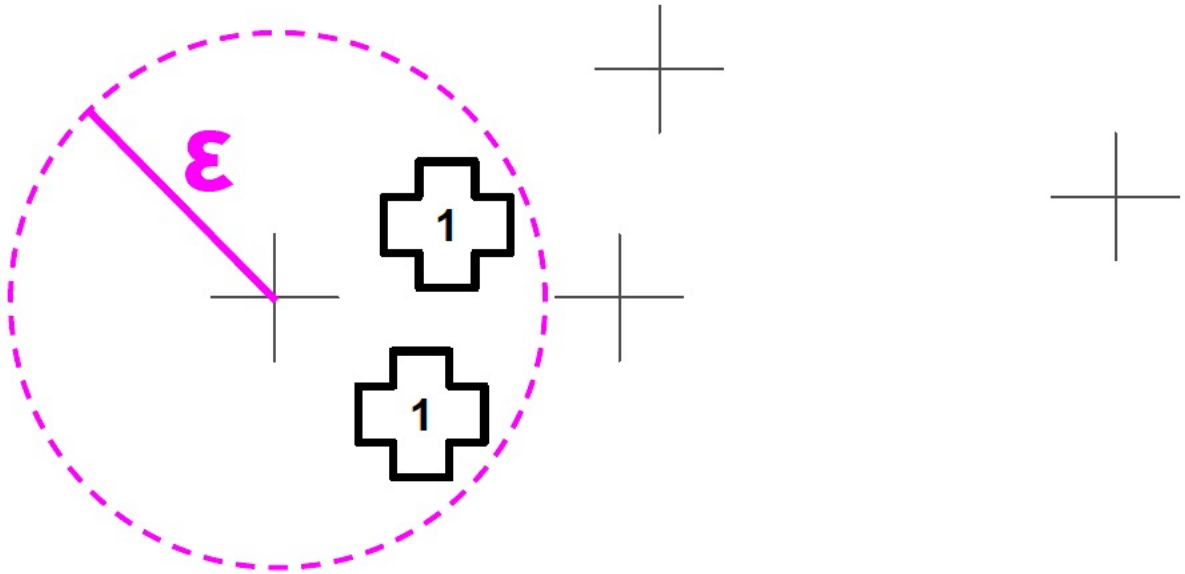



ε

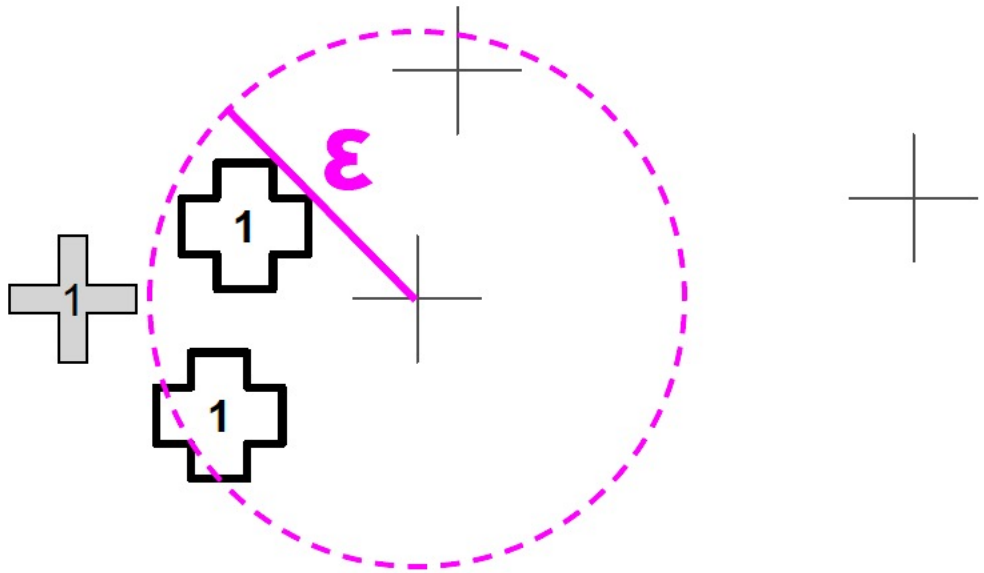

1

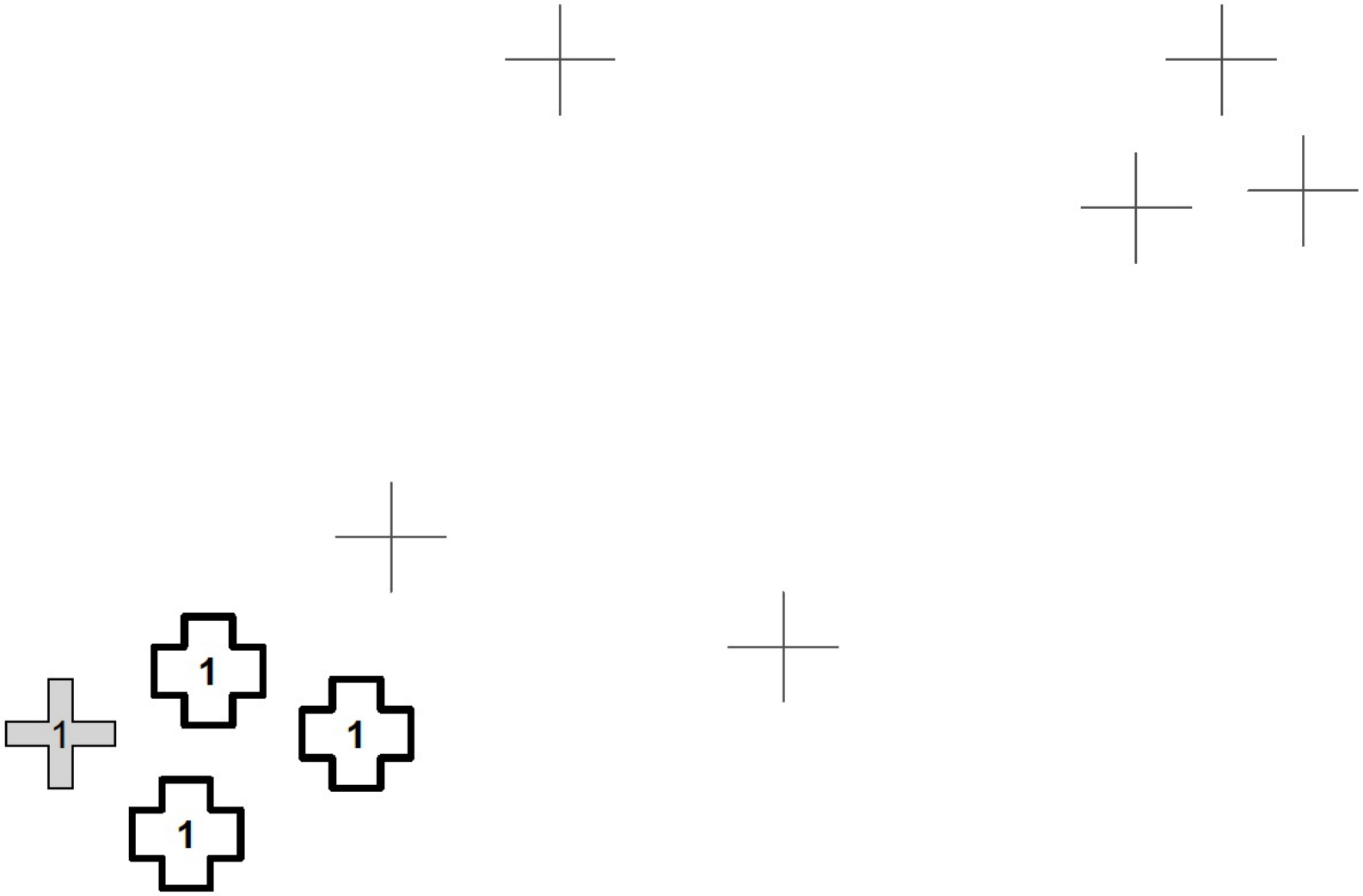

1
ε

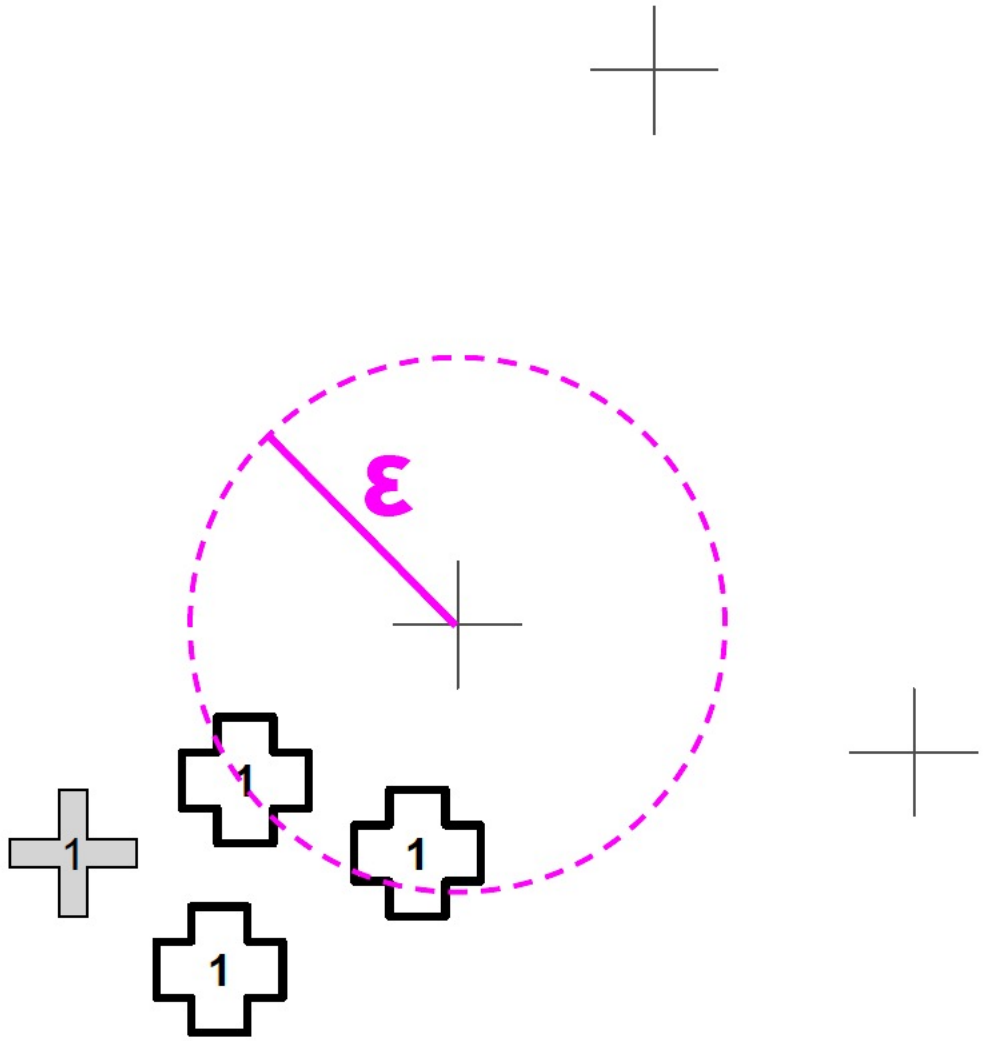

1
1

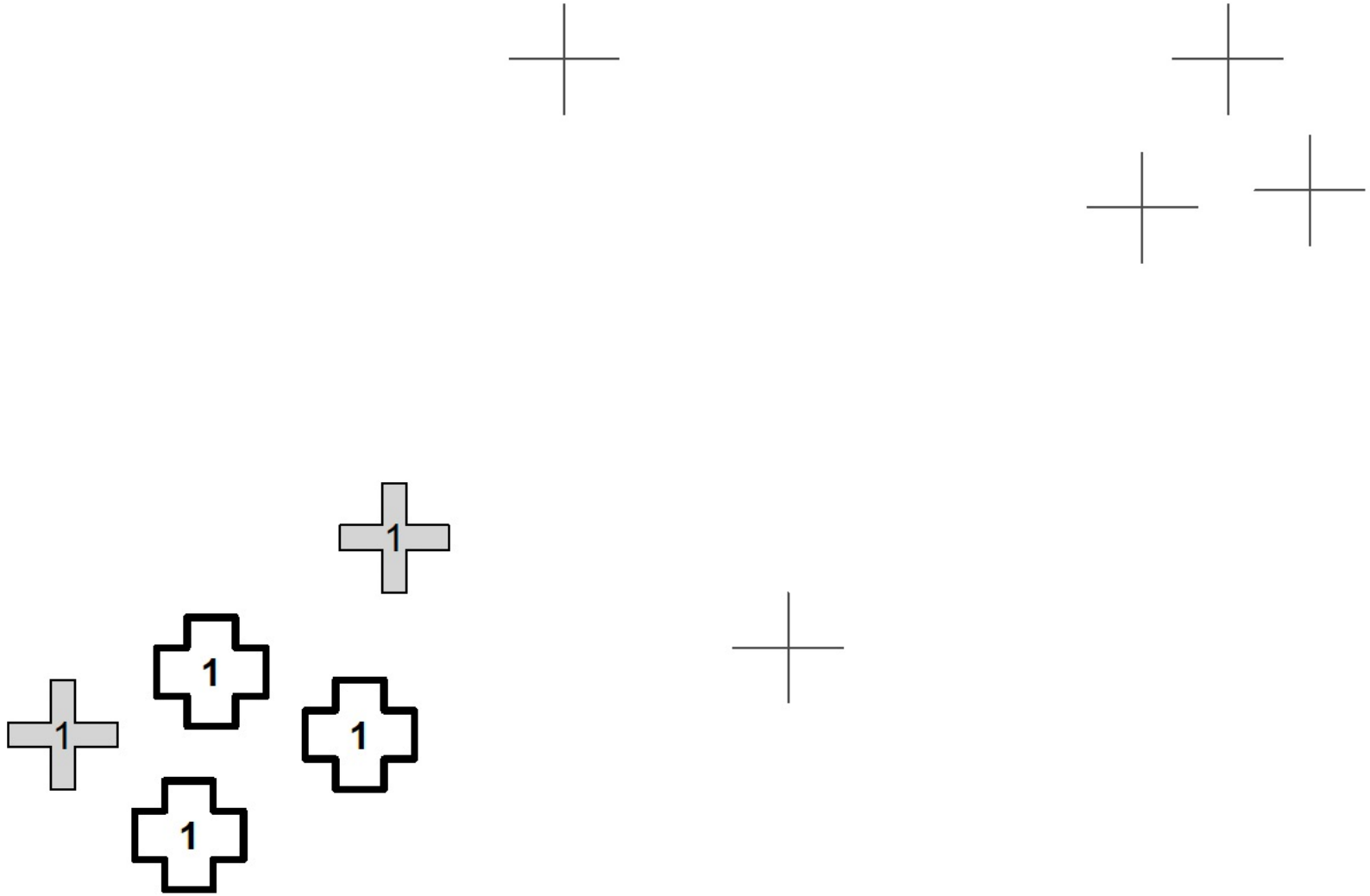

ε
1
1

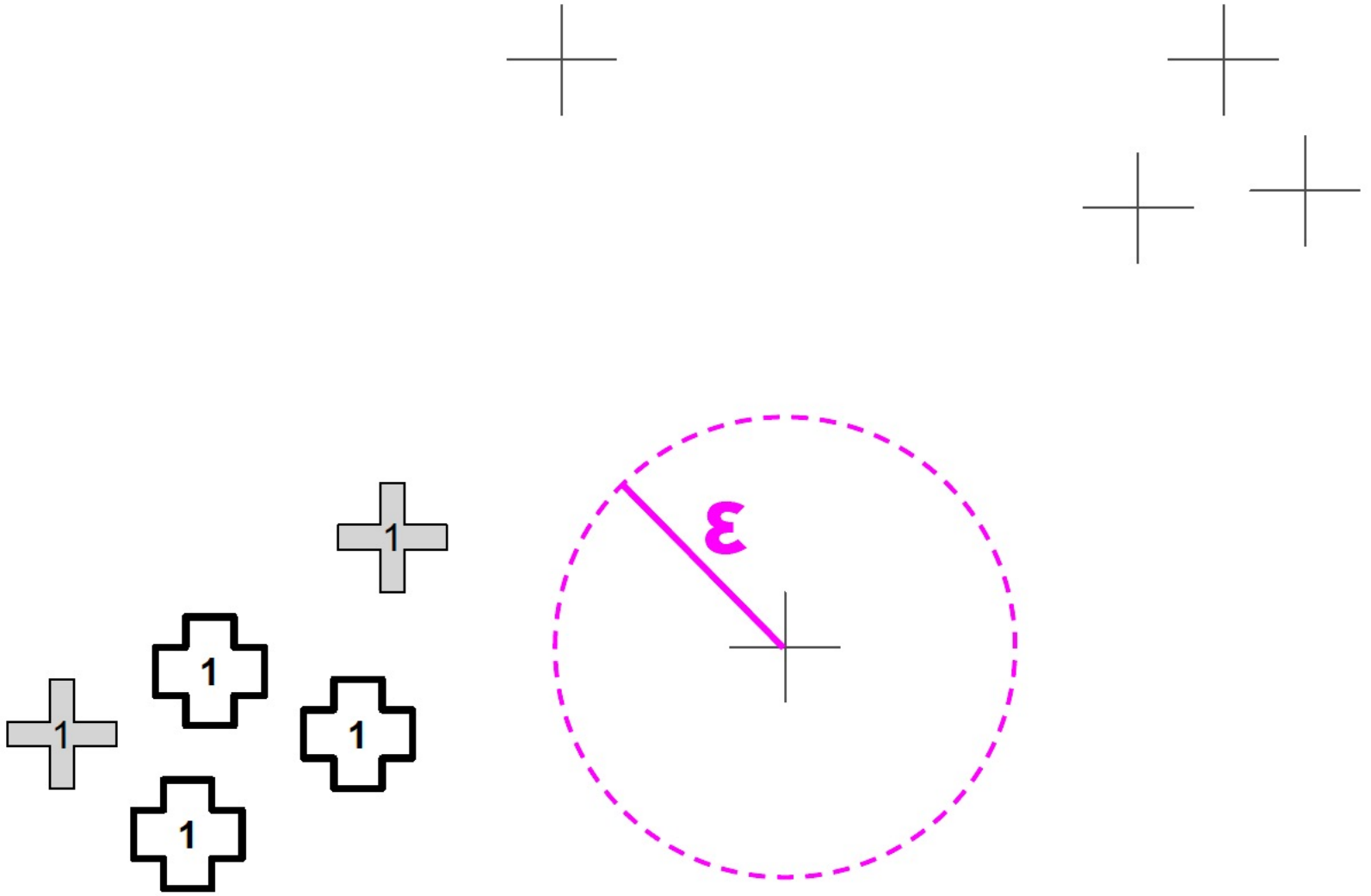

1
1
1

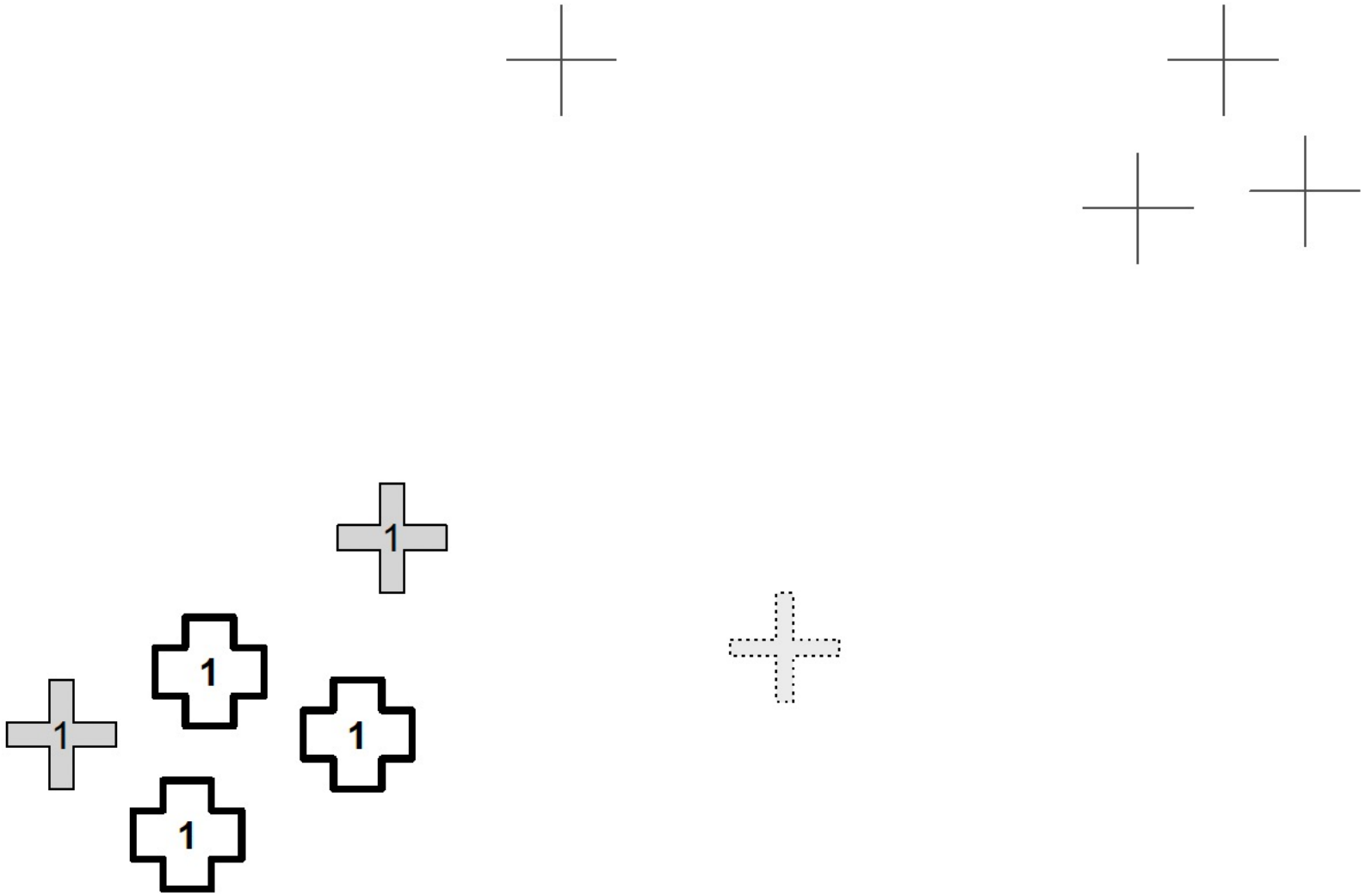

ε
1
1
1


1
1
1
1

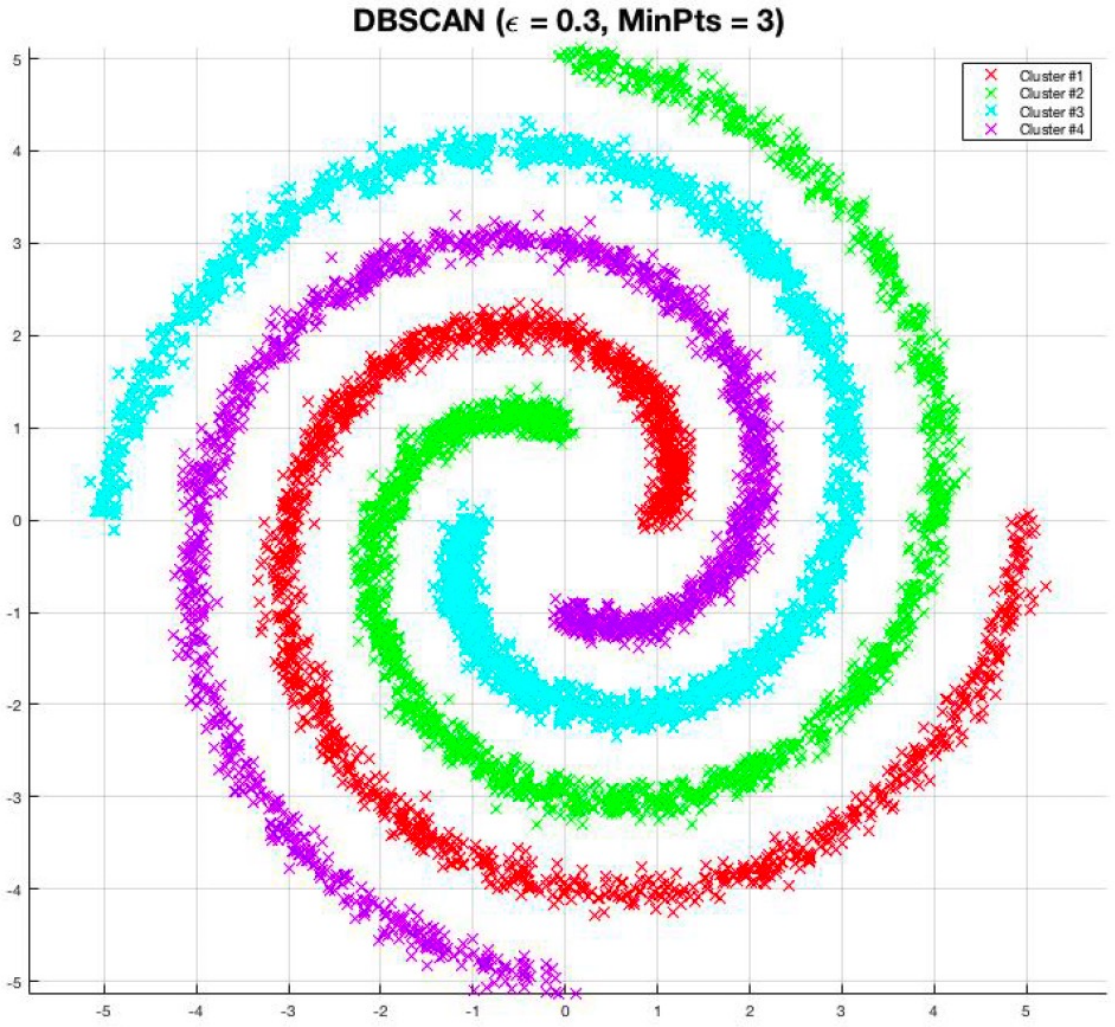

ε
1
1
1
1


1
1
1
1
1

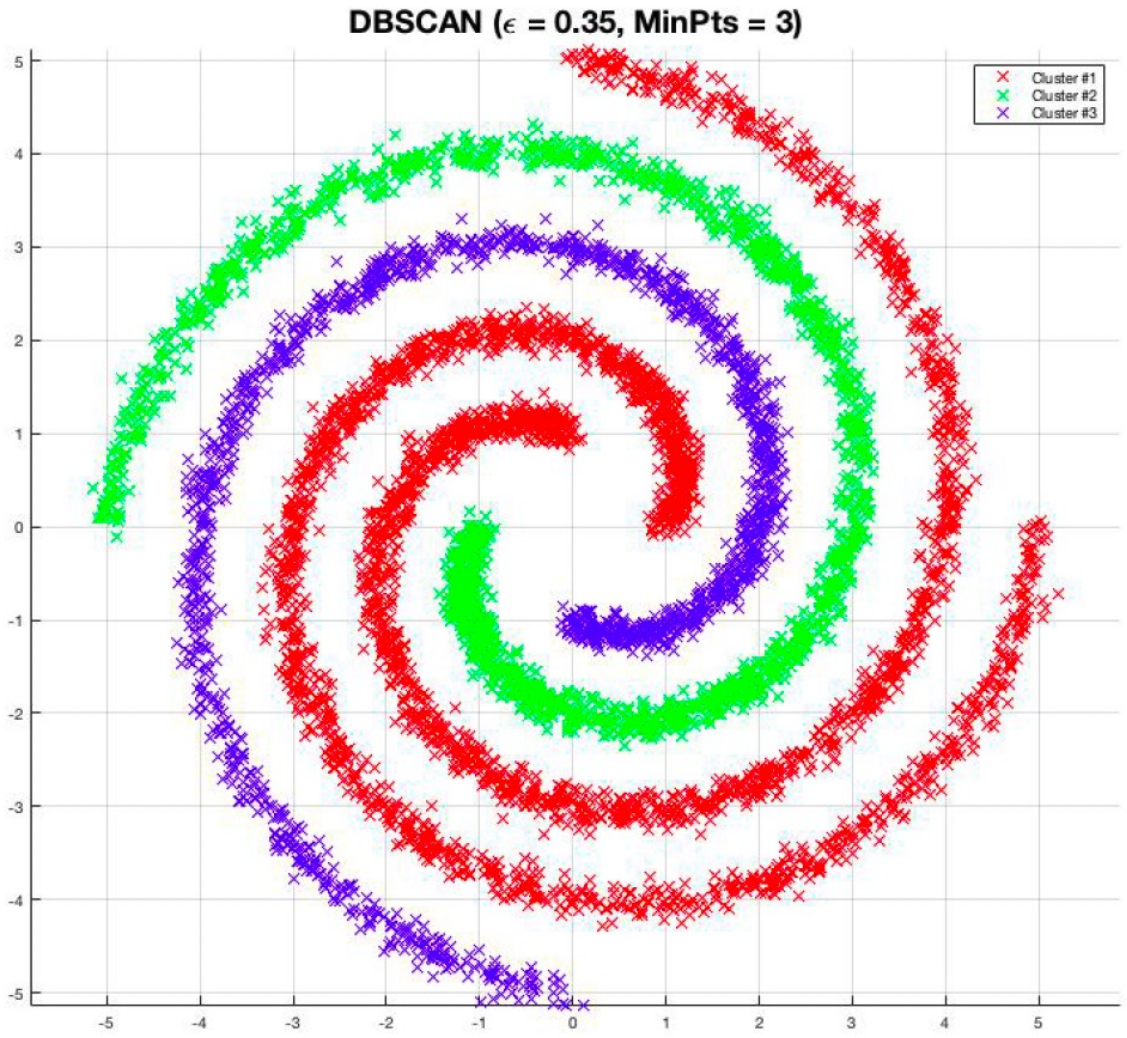

1
1
1
1
1
ε


1
1
1
1
1

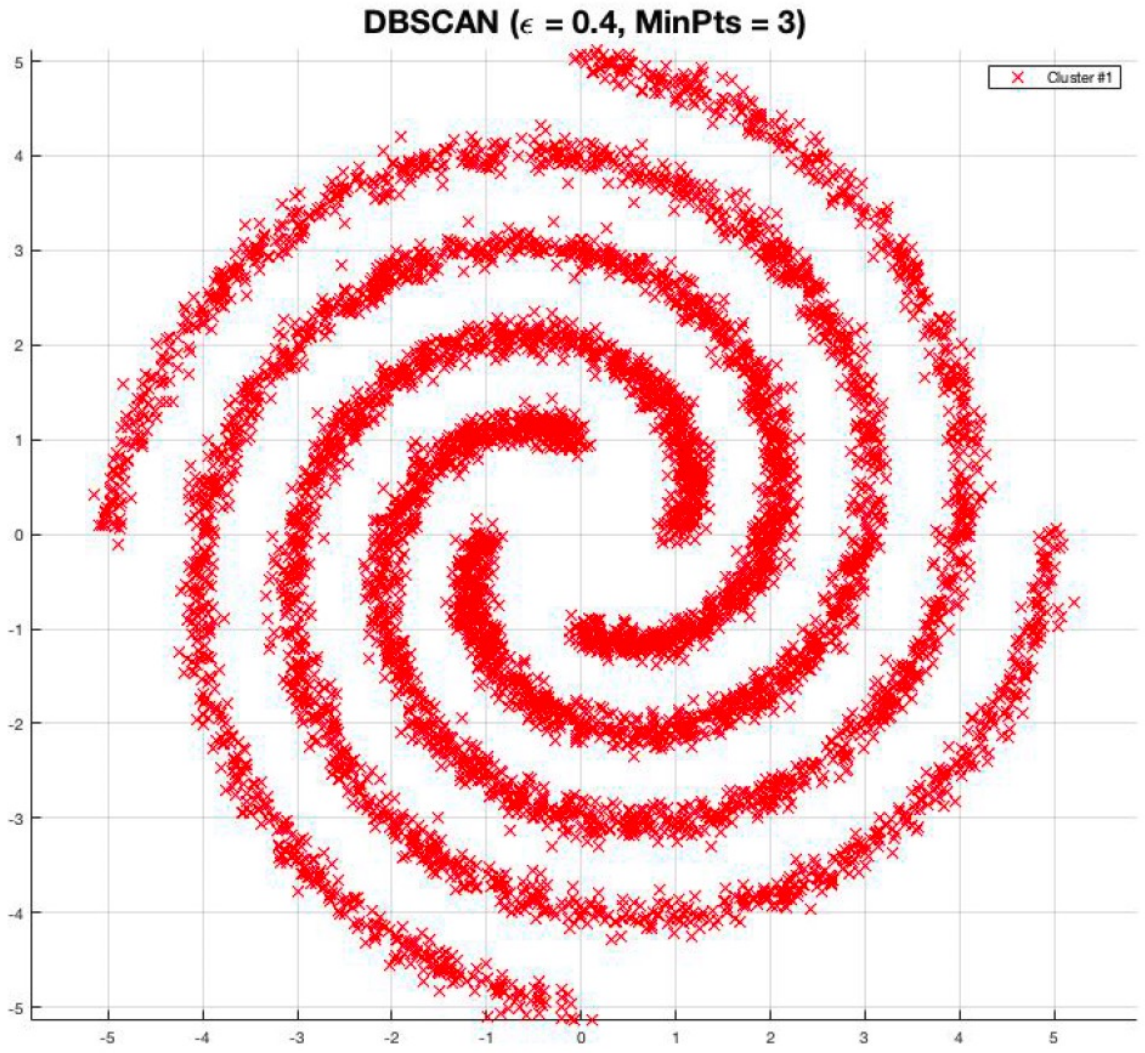
Parameters affect the results!



Parameters affect the results!

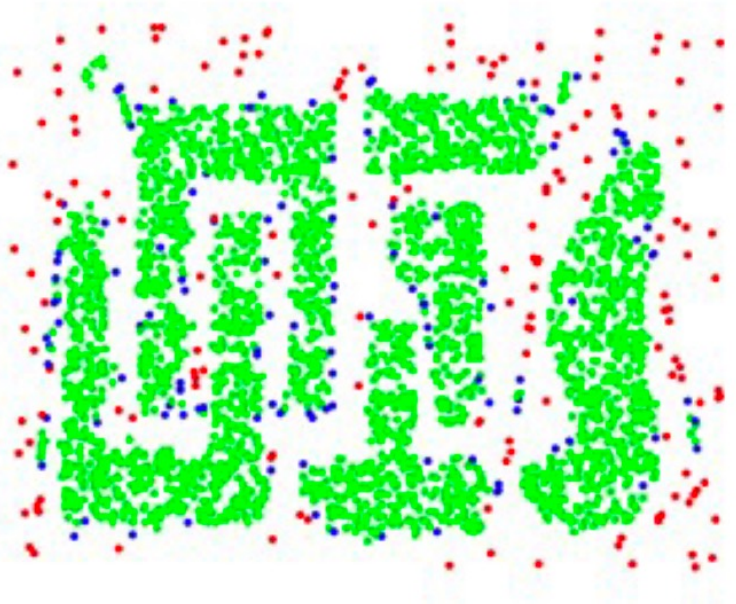


Parameters affect the results!



Another example

Original Points



Point types: core
border and noise

MinPts = 4

Plan for the next lectures

- *Kmeans*
- *DBSCAN*
- ***OPTICS***
- *GMM*
- *PCA*

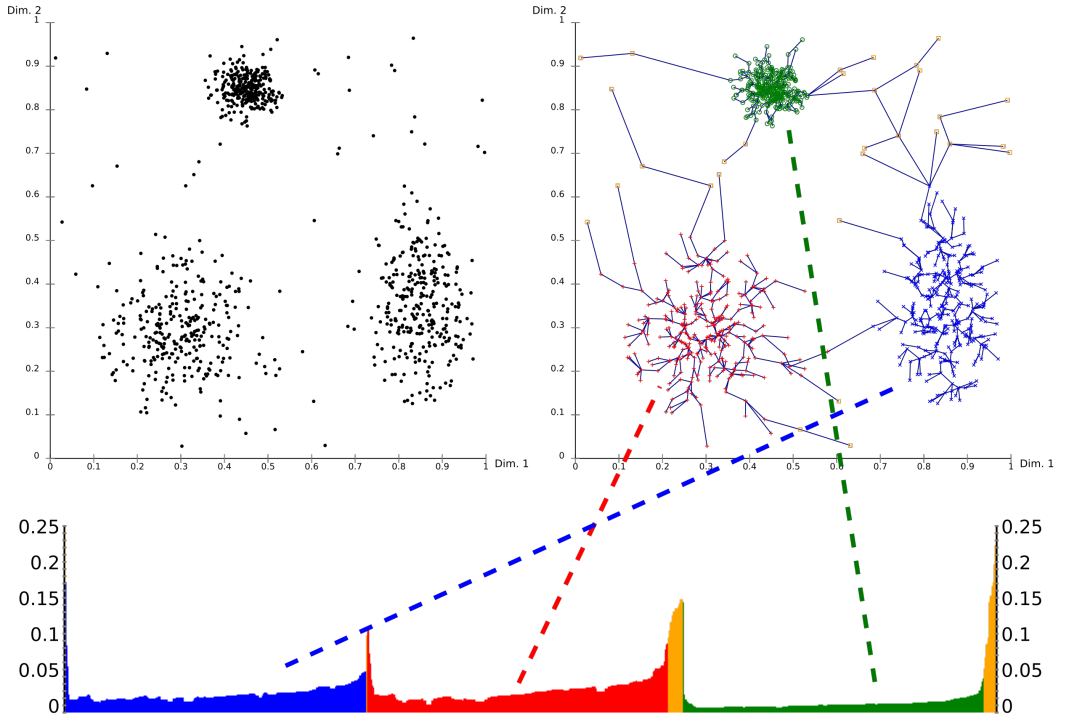
Another example of density-based clustering: OPTICS

OPTICS

- Name: **Ordering Points To Identify the Clustering Structure** (Ankerst et al. 1999)

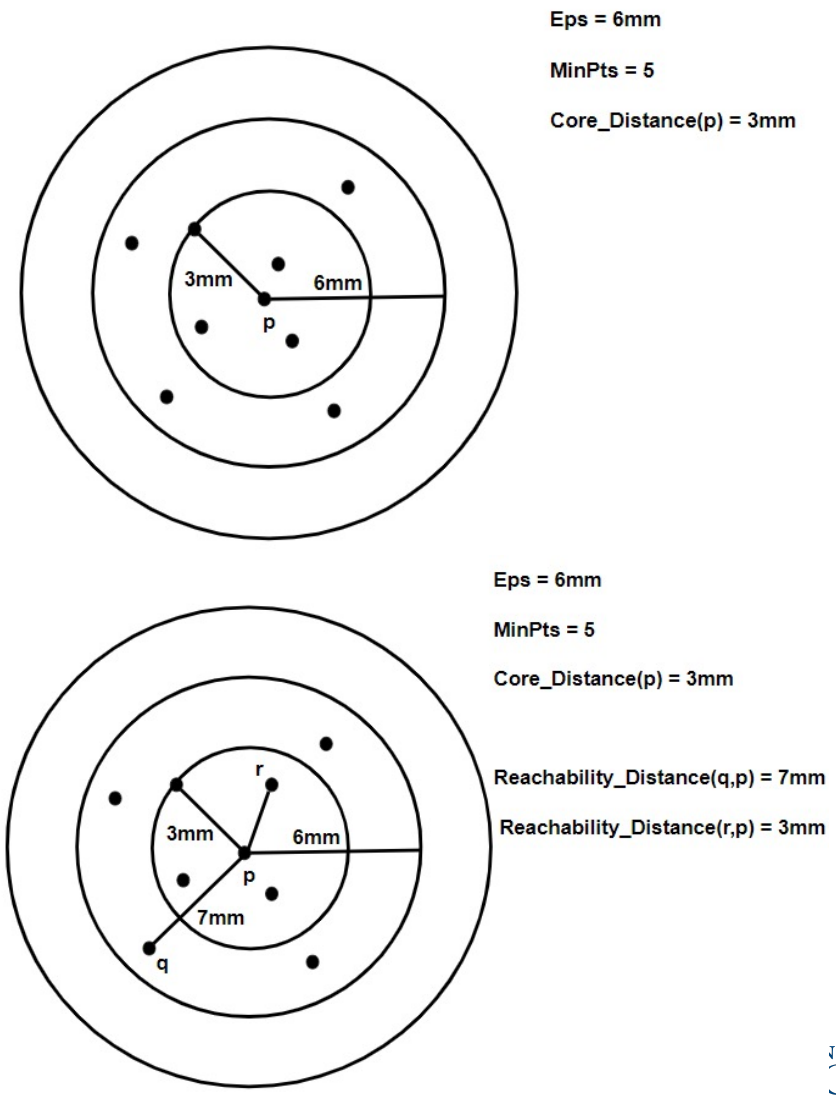
- Idea: Similar to DBSCAN…Also requires an epsilon (ε) and *MinPts* value.

  - Addresses one of DBSCAN's major weaknesses: the problem of detecting meaningful clusters in data of varying density
  - Main difference: Allows search radius (ε-neighborhood) around each point expand dynamically



OPTICS: New Metrics

- ***Core distance:***
  - It is the minimum ε to classify a given point as a **core point**. If the given point is not a **core point**, then its ***core distance*** is undefined.
- ***Reachability distance:***
  - The ***reachability distance*** between two core points ***p*** and ***q*** is the maximum of the ***core distance*** of ***p*** and the ***Euclidean Distance***(or some other distance metric) between ***p*** and ***q***.
  - Note: Reachability distance is not defined if ***p*** is not a core point.



OPTICS: How To Run

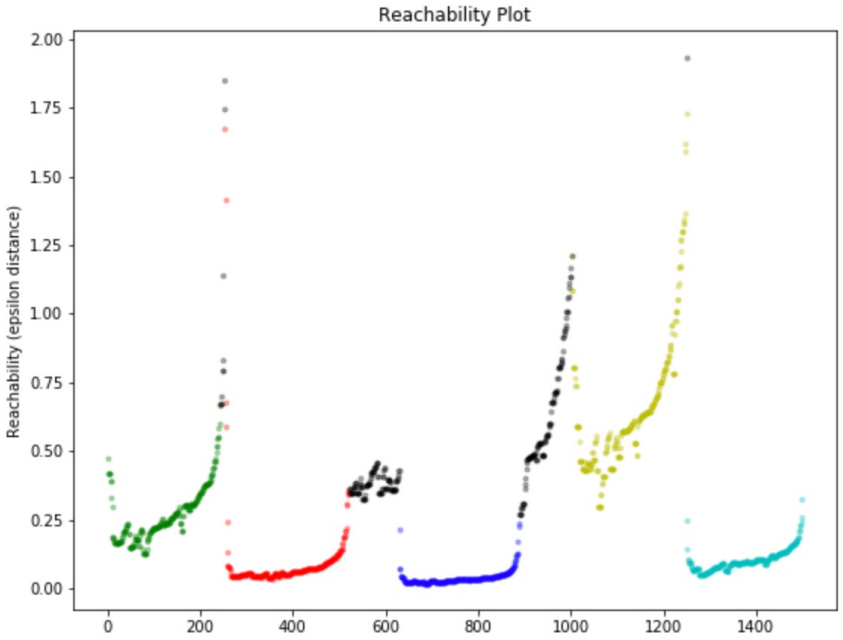
- **Step 1**: Mark every point in the dataset as '*unclustered*'. Also mark both core distances and reachability distance as '*undefined*'.

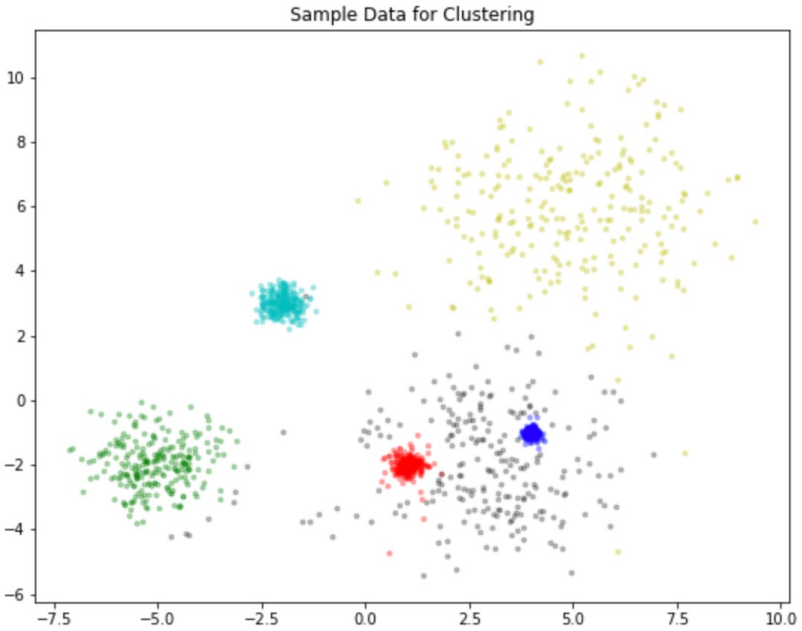
- **Step 2**: Pick any point in the dataset and check the ε-neighborhood of each point (like DBSCAN).
  - Assign a **cluster label** to **core points** once **core distance** is computed.

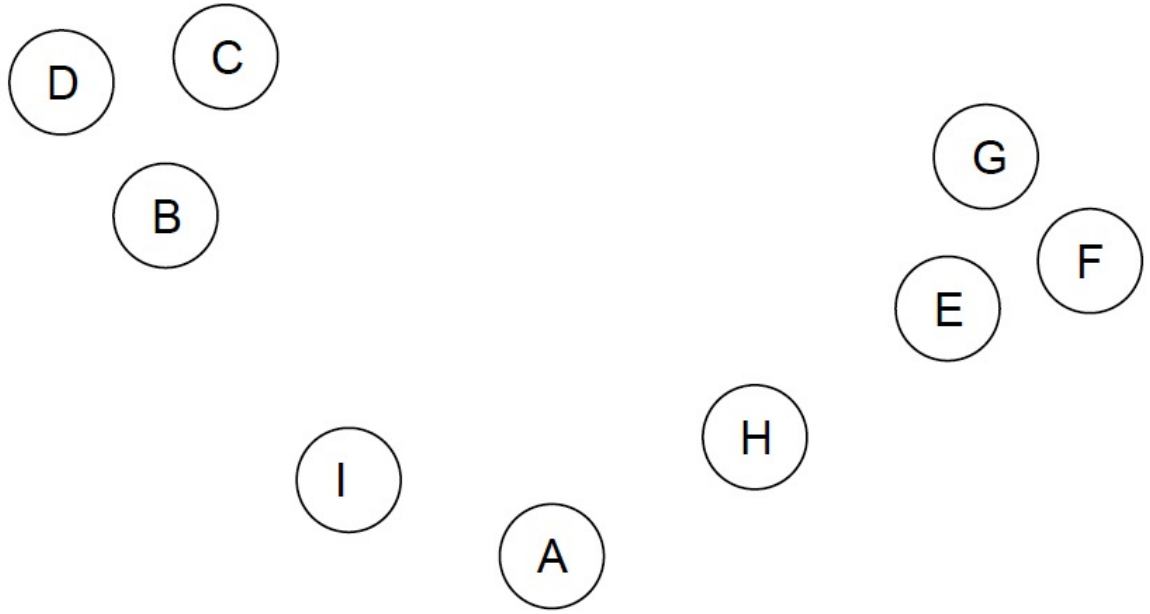
- **Step 3**: Compute reachability distance of every point in the ε-neighborhood.
  - If ***reachability distance*** is *undefined*, replace it with the newly computed ***reachability distance***.
  - If ***reachability distance*** exists for that point, replace it with newly computed ***reachability distance***, if:
    - **new reachability distance < old reachability distance**

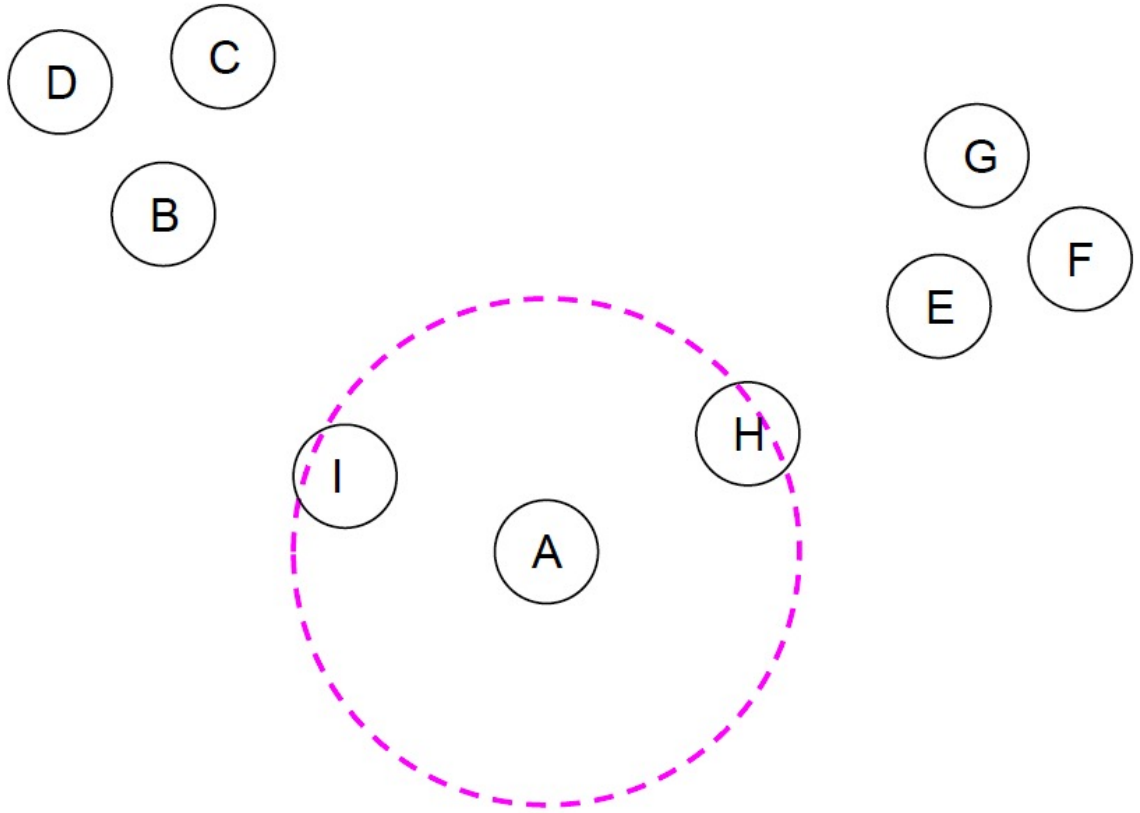
OPTICS: How To Run

- **Step 4**: Repeat Step 2 for every point in the ε-neighborhood starting with the point that has the closest ***reachability distance*** (and rank the distances).
- **Step 5**: Extract the ***cluster labels*** by looking at the distribution of reachability distances from ***core points***; and identify the 'valleys' (local minimums and maximums) in the plot.



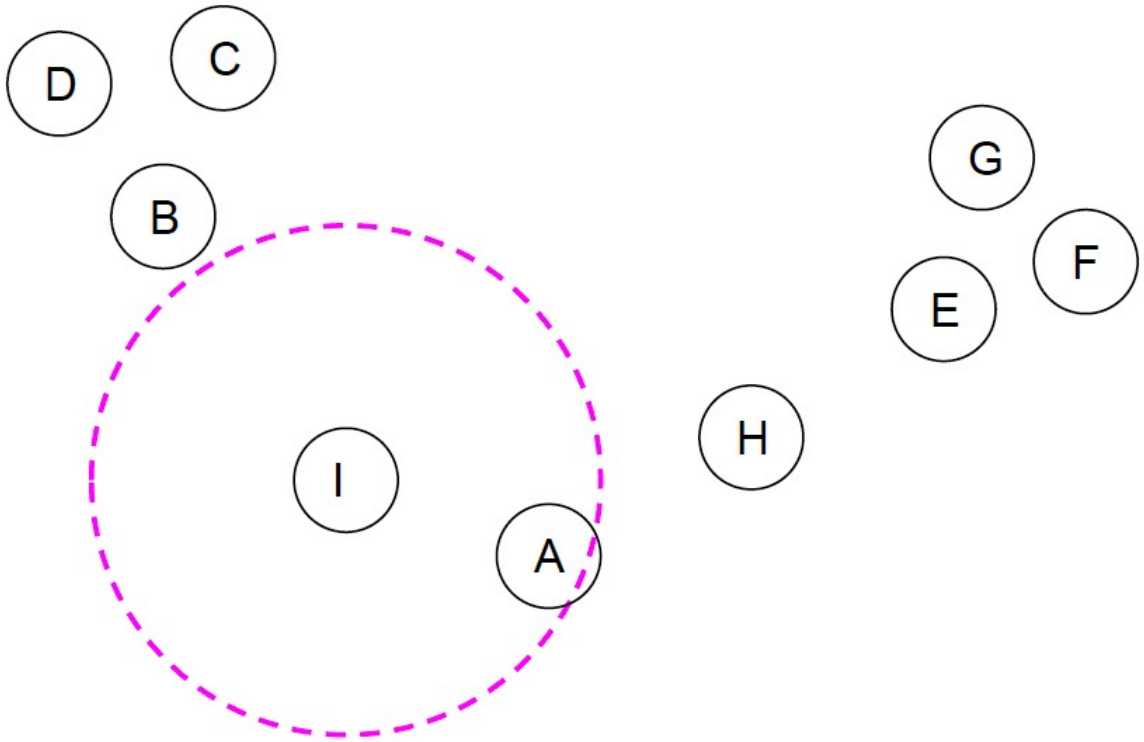



D
C
B
G
F
E
H
I
A



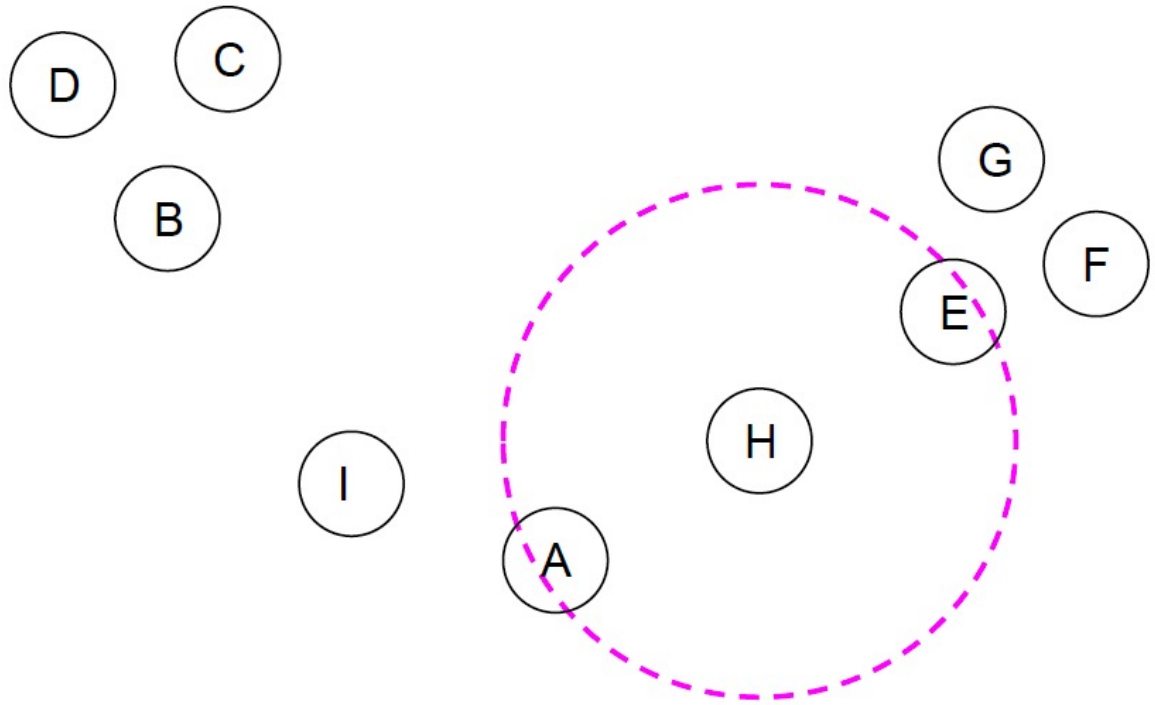
(I ; 30) (H ; 30)

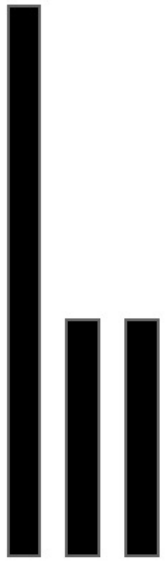
A



(H ; 30)

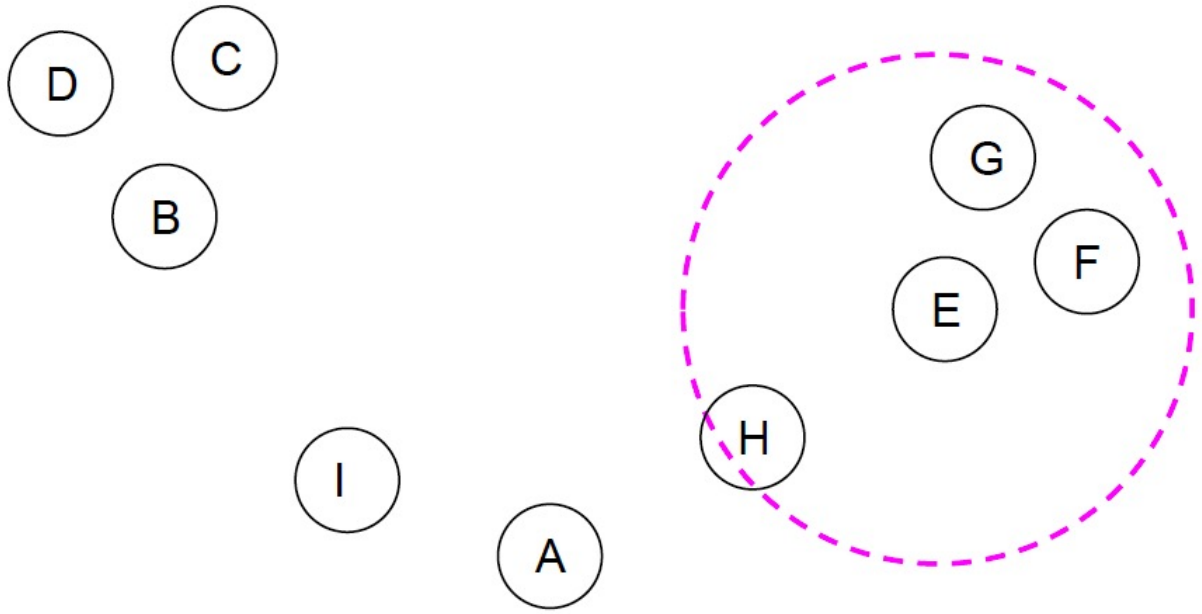
A I





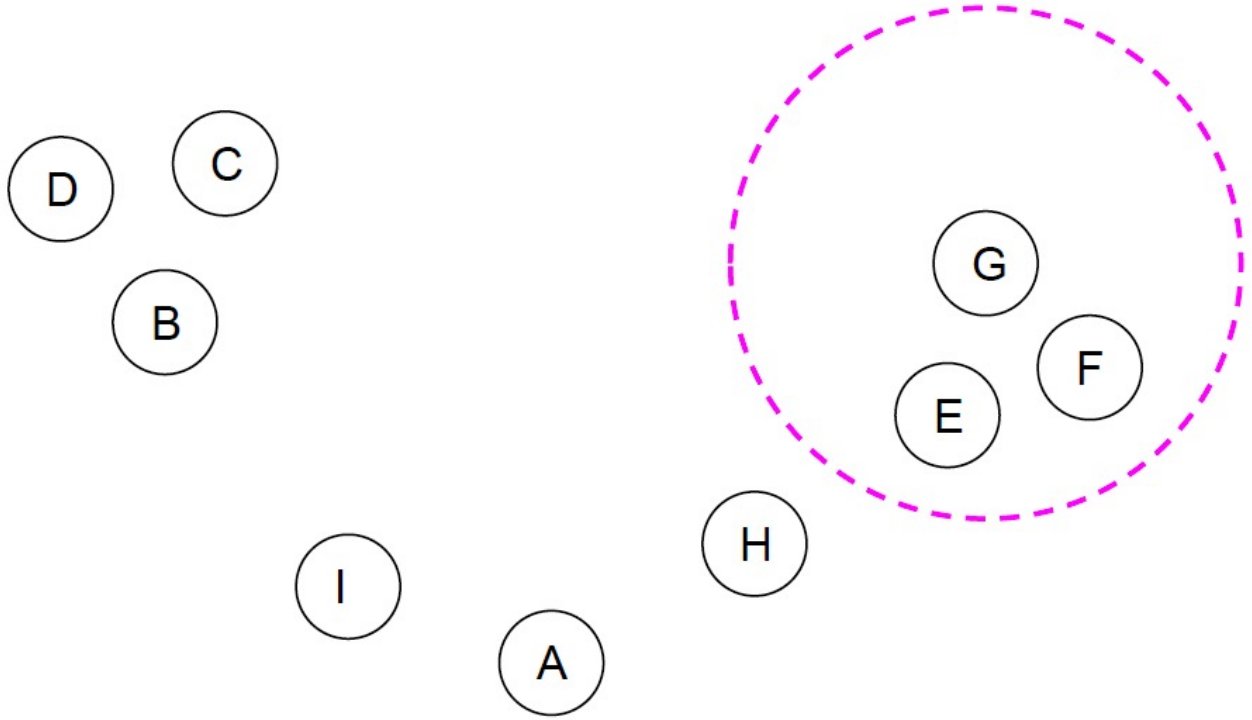
A I H

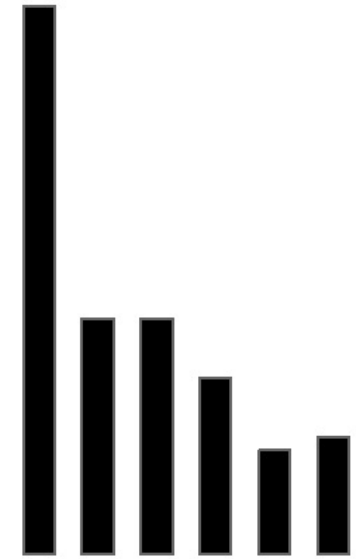
(E ; 28)



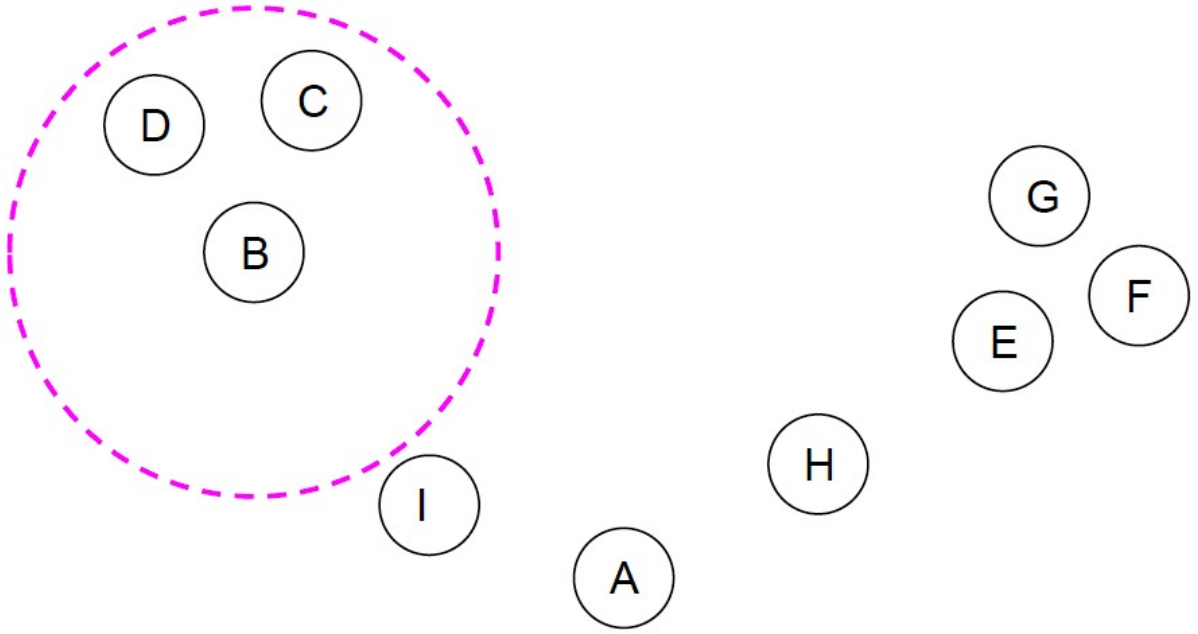
A I H E

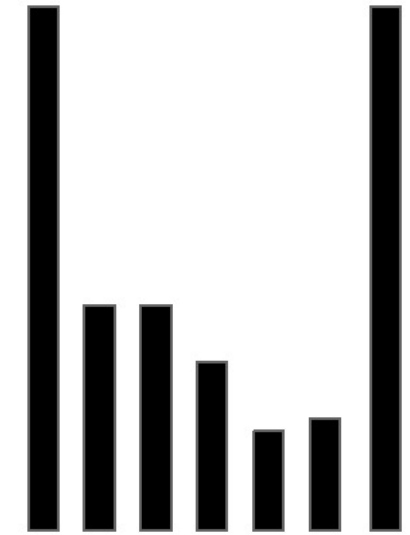
(G ; 10) (F ; 12)





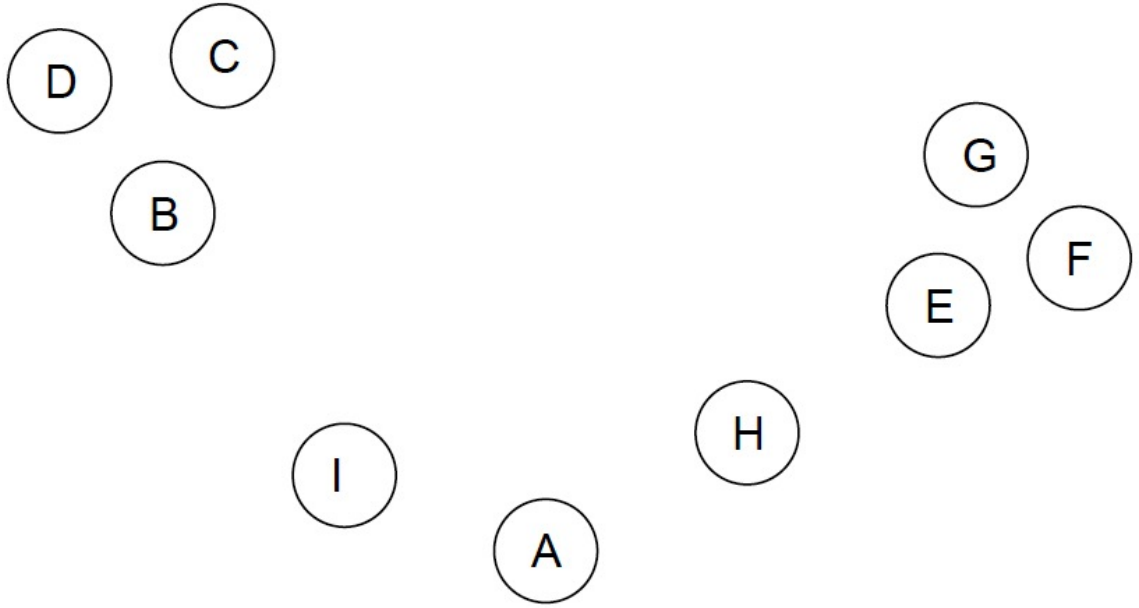
A I H E G F

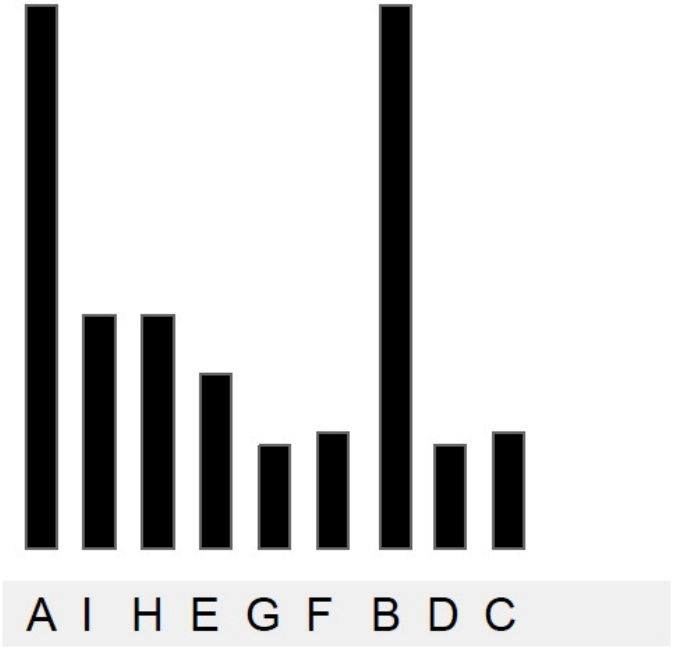


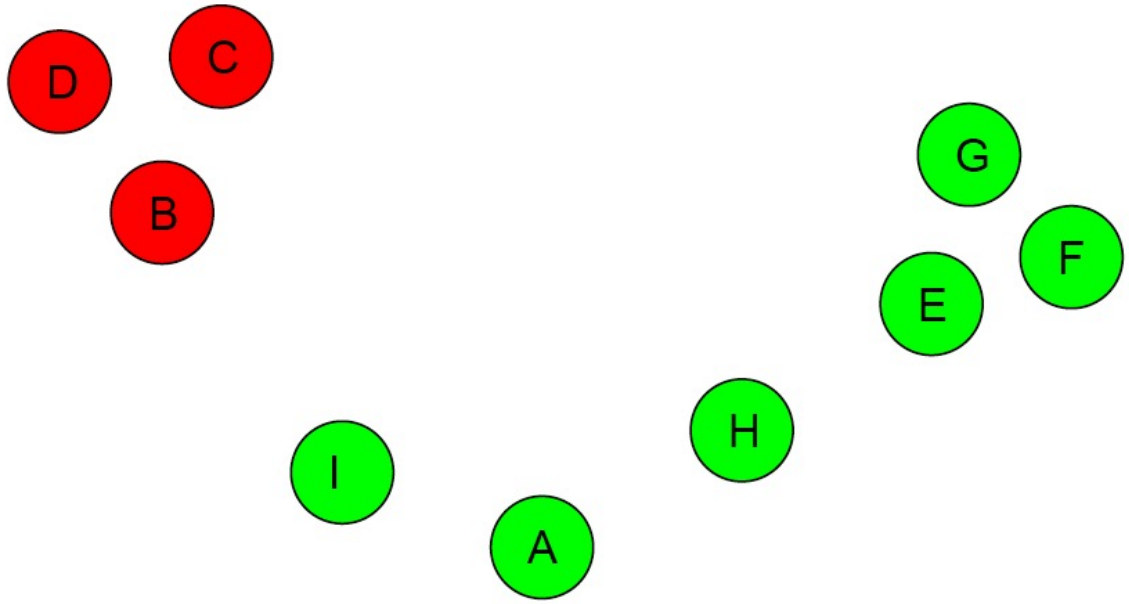


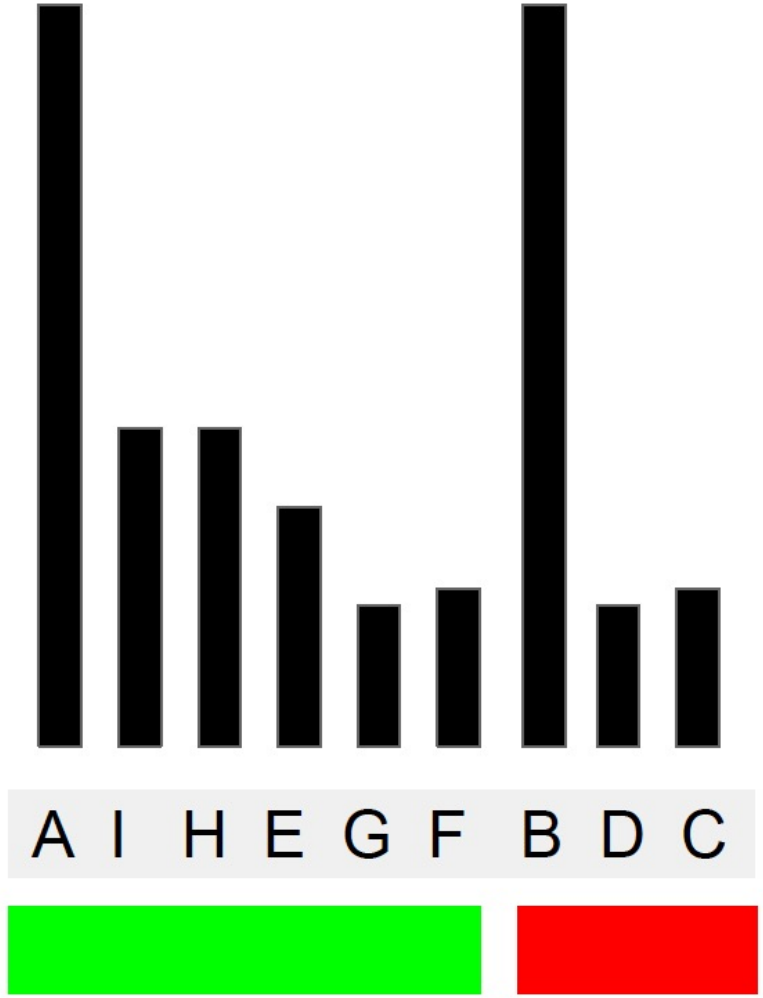
A I H E G F B

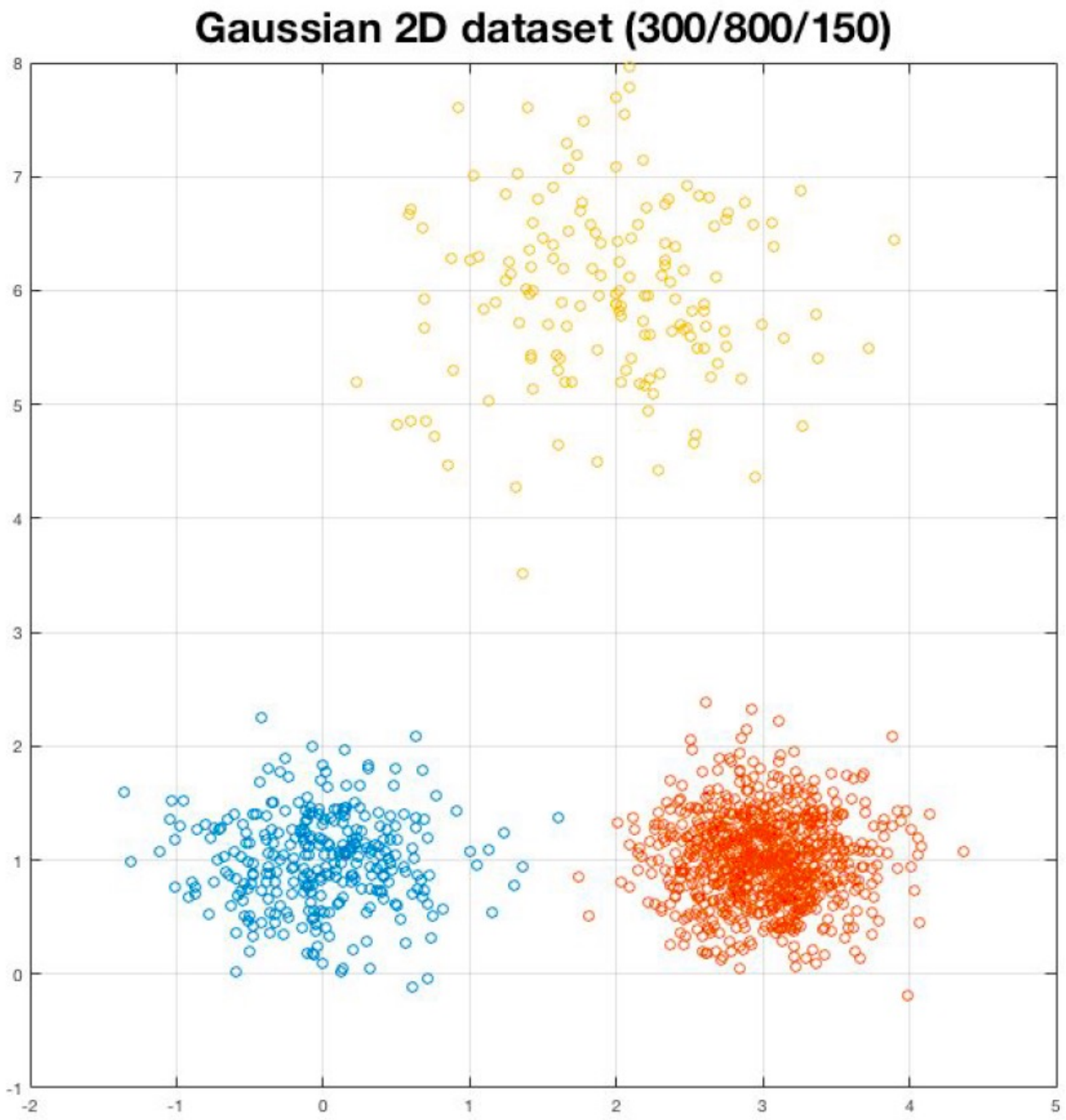
(D ; 11) (C ; 13)

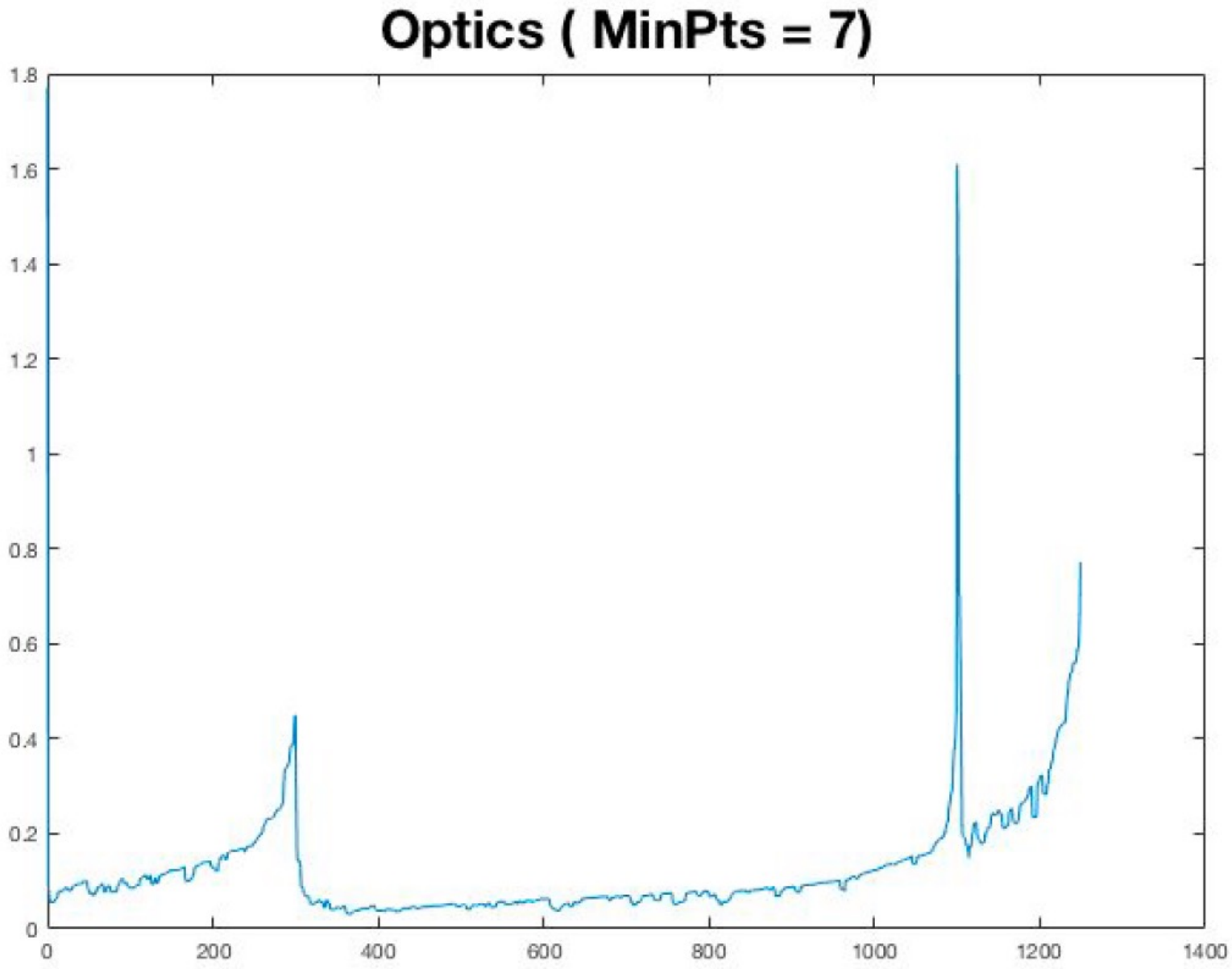

D
C
B
G
F
E
H
I
A


A I H E G F B D C


D
C
B
G
F
E
H
I
A


A I H E G F B D C


Gaussian 2D dataset (300/800/150)
8
7
6
5
4
3
2
1
0
-1
-2
-1
0
1
2
3
4
5


Optics (MinPts = 7)
1.8
1.6
1.4
1.2
1
0.8
0.6
0.4
0.2
0
0
200
400
600
800
1000
1200
1400