EE 4146 Data Engineering and Learning Systems

Lecture 5: Clustering I

Semester A, 2021-2022

Schedules

Week	Date	Topics
1	Sep. 1	Introduction
2	Sep. 8	Data exploration
3	Sep. 15	Feature reduction and selection (HW1 out)
4	Sep. 22	Mid-Autumn Festival
5	Sep. 29	Clustering I: Kmeans based models (HW1 due in this weekend)
6	Oct. 6	Clustering II: Hierarchical/density based/fuzzing clustering
7	Oct. 13	Midterm (no tutorials this week)
8	Oct. 20	Linear classifiers
9	Oct. 27	Classification based on decision tree (Tutorial on project) (HW2 out)
10	Nov. 3	Bayes based classifier (Tutorial on codes) (HW2 due in this weekend)
11	Nov. 10	KNN and classifier ensemble
12	Nov. 17	Deep learning based models (Quiz)
13	Nov. 24	Summary

End at 6:40 PM with only one break at 5:20 PM

Midterm

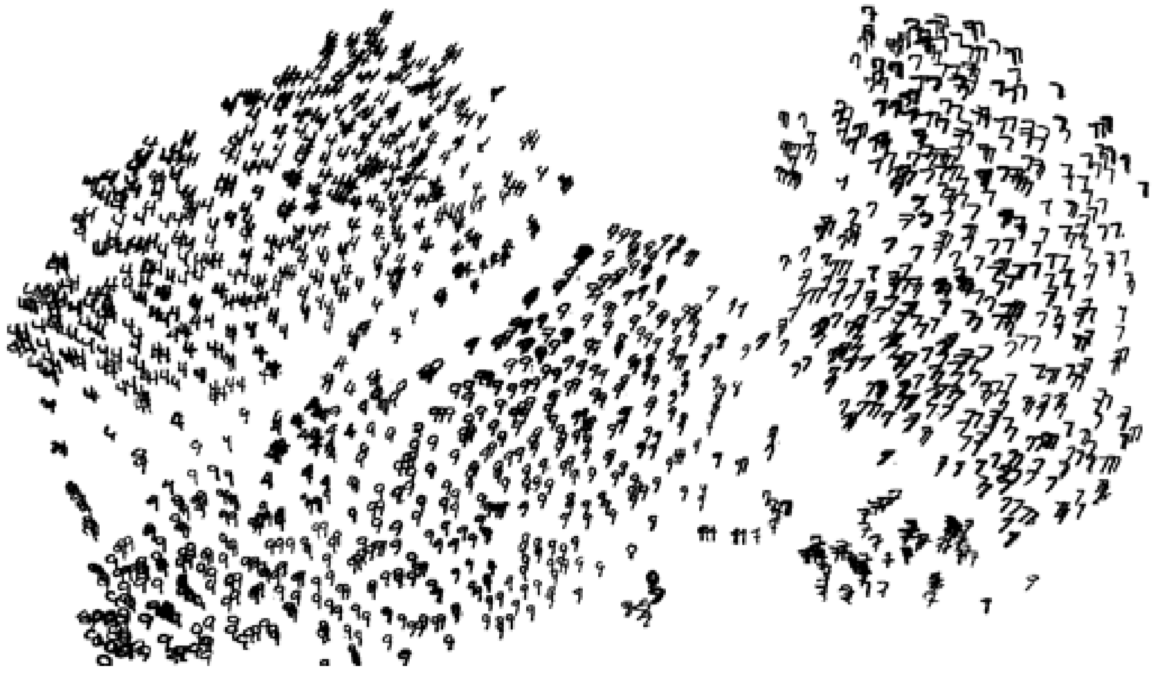
- Face by face
- No electronic devices allowed
- Open-book and open-notes
- Printed midterm paper

Outline

- Introduction of clusters
- Introduction of clustering
- K-means
- K-medoids
  - PAM&CLARA & CLARANS

High Dimensional Data

- Given a cloud of data points we want to understand its structure

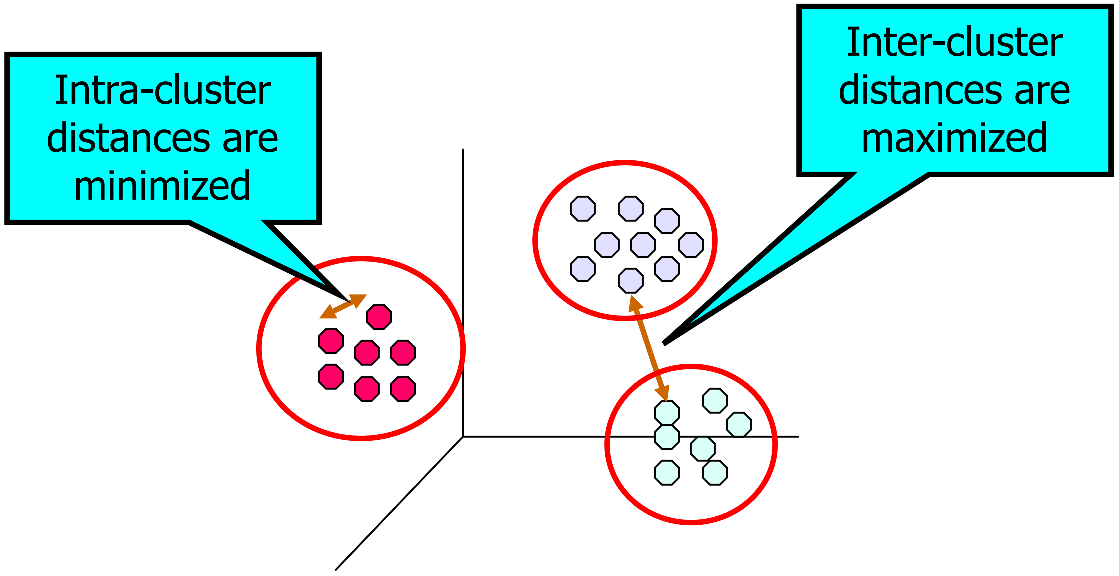


The Problem of Clustering

- Given a set of points, with a notion of distance between points, group the points into some number of clusters, so that
  - Members of a cluster are close/similar to each other
  - Members of different clusters are dissimilar
- Usually:
  - Points are in a high-dimensional space
  - Similarity is defined using a distance measure
  - Euclidean, Cosine, Jaccard, …

What is Cluster Analysis?

- Finding groups of objects such that the objects in a group will be similar (or related) to one another and different from (or unrelated to) the objects in other groups


Intra-cluster distances are minimized
Inter-cluster distances are maximized

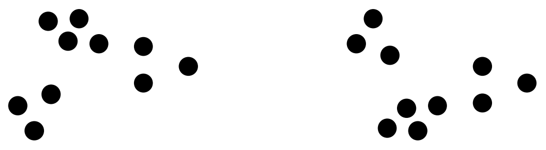
Quality: What Is Good Clustering?

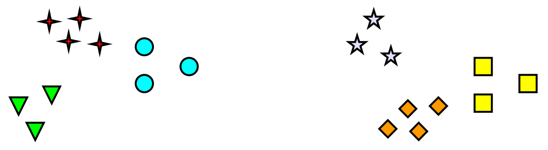
- A good clustering method will produce high quality clusters with
  - high intra-class similarity
  - low inter-class similarity
- The quality of a clustering result depends on both the similarity measure used by the method and its implementation
- The quality of a clustering result also depends on the definition and representation of cluster chosen.

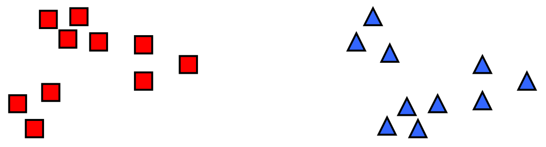
Measure the Quality of Clustering

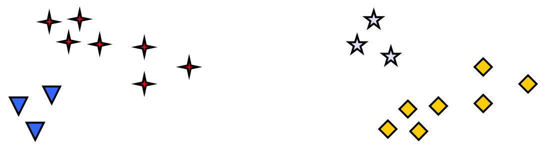
- Dissimilarity/Similarity metric: Similarity is expressed in terms of a distance function, typically metric: *d*(*i, j*)
- There is a separate "quality" function that measures the "goodness" of a cluster.
- The definitions of distance functions are usually very different for interval-scaled, boolean, categorical, ordinal ratio, and vector variables. -> Minkowski Distance, Simple Matching and Jaccard Coefficients, etc.
- Weights should be associated with different variables based on applications and data semantics.
- It is hard to define "similar enough" or "good enough"
  - the answer is typically highly subjective.

Notion of a Cluster can be Ambiguous


How many clusters?


Six Clusters


Two Clusters


Four Clusters

Review

- *Minkowski distance*

$$d(i,j) = \sqrt[q]{(|x_{i_1} - x_{j_1}|^q + |x_{i_2} - x_{j_2}|^q + \ldots + |x_{i_p} - x_{j_p}|^q)}$$

where *i* = (x_{i1}, x_{i2}, …, x_{ip}) and *j* = (x_{j1}, x_{j2}, …, x_{jp}) are two *p*-dimensional data objects, and *q* is a positive integer

- If *q = 1, d* is Manhattan distance

$$d(i,j) = |x_{i_1} - x_{j_1}| + |x_{i_2} - x_{j_2}| + \ldots + |x_{i_p} - x_{j_p}|$$

- *If q = 2, d* is Euclidean distance

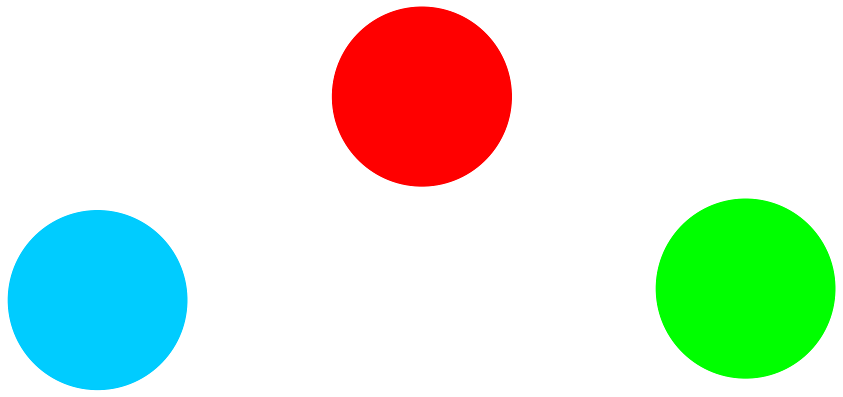
$$d(i,j) = \sqrt{(|x_{i_1} - x_{j_1}|^2 + |x_{i_2} - x_{j_2}|^2 + \ldots + |x_{i_p} - x_{j_p}|^2)}$$

Types of Clusters

- Well-separated clusters
- Center-based clusters
- Contiguous clusters
- Density-based clusters
- Property or Conceptual
- Described by an Objective Function

Types of Clusters: Well-Separated

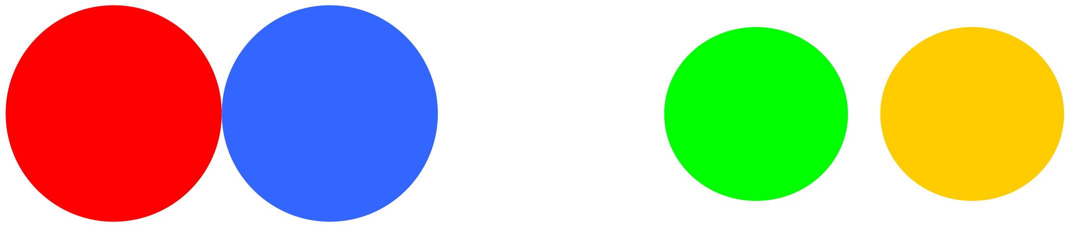
- Well-Separated Clusters:
  - A cluster is a set of points such that any point in a cluster is closer (or more similar) to every other point in the cluster than to any point not in the cluster.



3 well-separated clusters

Types of Clusters: Center-Based

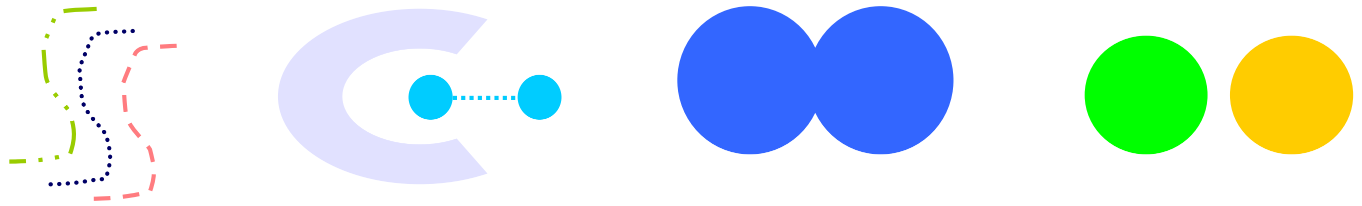
- Center-based
  - A cluster is a set of points such that an point in a cluster is closer (more similar) to the "center" of a cluster, than to the center of any other cluster
  - The center of a cluster is often a centroid, the average of all the points in the cluster, or a medoid, the most "representative" point of a cluster



4 center-based clusters

Types of Clusters: Contiguity-Based

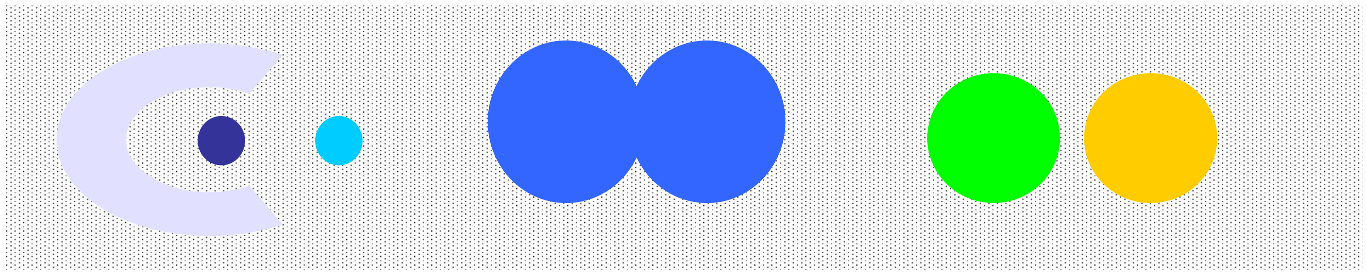
- Contiguous Cluster (Nearest neighbor or Transitive)
  - A cluster is a set of points such that a point in a cluster is closer (or more similar) to one or more other points in the cluster than to any point not in the cluster.



8 contiguous clusters

Types of Clusters: Density-Based

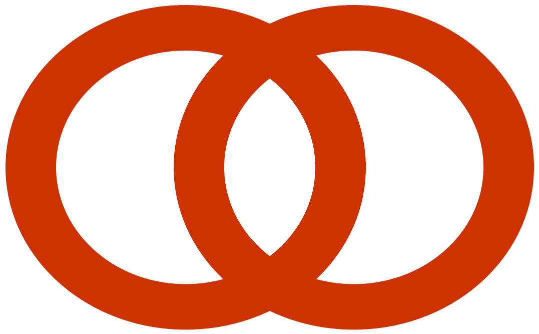
- Density-based
  - A cluster is a dense region of points, which is separated by low-density regions, from other regions of high density.
  - Used when the clusters are irregular or intertwined, and when noise and outliers are present.



6 density-based clusters

Types of Clusters: Conceptual Clusters

- Shared Property or Conceptual Clusters
  - Finds clusters that share some common property or represent a particular concept.



2 Overlapping Circles

Types of Clusters: objective function

- Clusters Defined by an Objective Function
  - Finds clusters that minimize or maximize an objective function.
  - Enumerate all possible ways of dividing the points into clusters and evaluate the `goodness' of each potential set of clusters by using the given objective function.
  - Can have global or local objectives.
    - Hierarchical clustering algorithms typically have local objectives
    - Partitional algorithms typically have global objectives

Types of Clusters: objective function

- Map the clustering problem to a different domain and solve a related problem in that domain
  - Proximity matrix defines a weighted graph, where the nodes are the points being clustered, and the weighted edges represent the proximities between points
  - Clustering is equivalent to breaking the graph into connected components, one for each cluster
  - Want to minimize the edge weight between clusters and maximize the edge weight within clusters

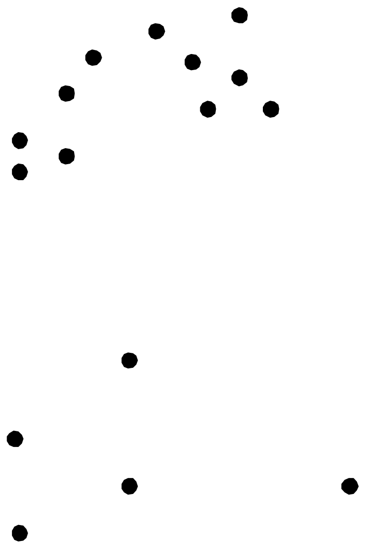
Outline

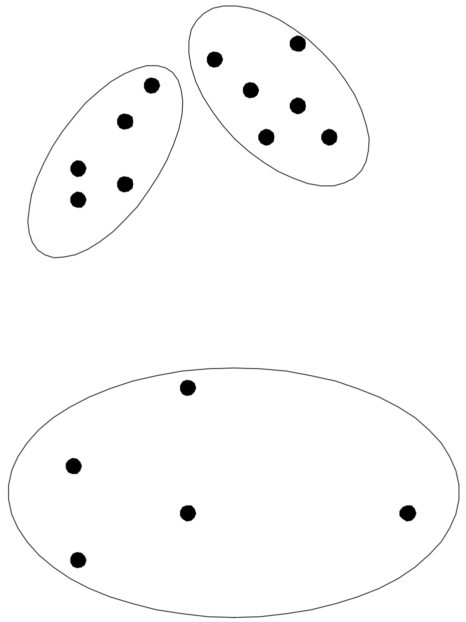
- Introduction of clusters
- Introduction of clustering
- K-means
- K-medoids
  - PAM&CLARA & CLARANS

Types of Clustering

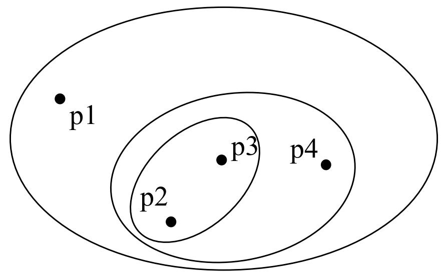
- A clustering is a process to find a set of clusters
- Partitional Clustering
  - A division data objects into non-overlapping subsets (clusters) such that each data object is in exactly one subset
- Hierarchical clustering
  - A set of nested clusters organized as a hierarchical tree
- Density based clustering
  - Discover clusters of arbitrary shape.
  - Cluster dense regions of objects separated by regions of low density

Partitional Clustering

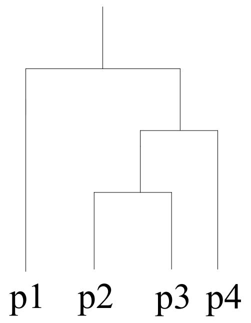

Original Points


A Partitional Clustering

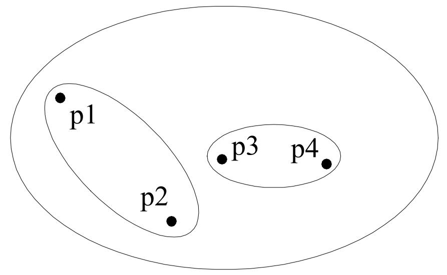
Hierarchical Clustering



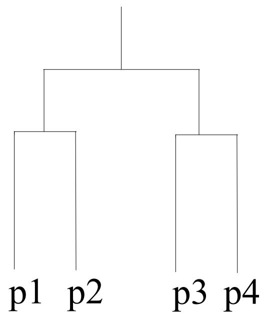
Traditional Hierarchical Clustering



Traditional Dendrogram

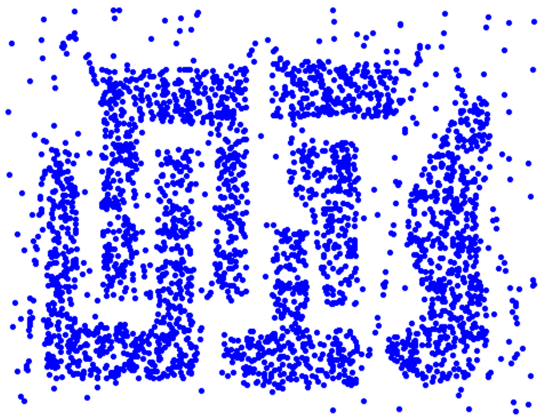


Non-traditional Hierarchical Clustering

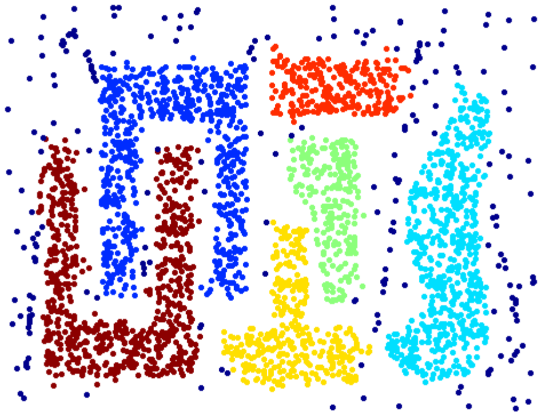


Non-traditional Dendrogram

Density based Clustering



Original Points



Clusters

Clustering Algorithms

- Partitional Clustering
  - Kmeans
- Hierarchical clustering
- Density-based clustering

Outline

- Introduction of clusters
- Introduction of clustering
- K-means
- K-medoids
  - PAM&CLARA & CLARANS

Clustering Algorithms

- K-means
  - Basic introduction-workflow, calculation, codes
  - Initial Centroids Problems & solutions
  - Data issues (distribution, outliers) & solutions

K-means Clustering

- One of the most commonly-used clustering algorithms, because it is easy to implement and quick to run.
- Clustering means grouping similar data together
- From a set of data points, K-means attempts to classify all data points into K clusters
- K clusters may represent K segments of an image
- The algorithm is iterative in nature
  - Data points
  - images

K-means Clustering

- Given
  - A data set of **n objects**
  - **K the number of clusters to form**
- Organize the objects into k cluster **(k<=n)**
- The clusters are formed to optimize an objective partitioning criterion
  - Objects within a cluster are **similar**
  - Objects of different clusters are **dissimilar**

K-means Clustering

- The basic algorithm is very simple
- Number of clusters, K, must be specified
- Each cluster is associated with a centroid (mean or center point)
- Each point is assigned to the cluster with the closest centroid

```
1: Select $K$ points as the initial centroids.
2: **repeat**
3:     Form $K$ clusters by assigning all points to the closest centroid.
4:     Recompute the centroid of each cluster.
5: **until** The centroids don't change
```

K-means: Steps

- Partition the data points into K clusters randomly. Find the centroids of each cluster.
- For each data point:
  - Calculate the distance from the data point to each cluster.
  - Assign the data point to the closest cluster.
- Recompute the centroid of each cluster.
- Repeat steps 2 and 3 until there is no further change in the assignment of data points (or in the centroids).

$$\sum_{i \in \text{clusters}} \left\{ \sum_{j \in \text{elements of i'th cluster}} \left\| x_j - \mu_i \right\|^2 \right\}$$

Kmeans Convergence

Objective

$$\min_{\mu}\min_{C} \sum_{i=1}^{k} \sum_{x \in C_i} |x - \mu_i|^2$$

1. Fix μ, optimize *C:*

Step 1 of kmeans

$$\min_{C} \sum_{i=1}^{k} \sum_{x \in C_i} |x - \mu_i|^2 = \min_{c} \sum_{i}^{n} \left|x_i - \mu_{x_i}\right|^2$$

2. Fix *C*, optimize μ:

$$\min_{\mu} \sum_{i=1}^{k} \sum_{x \in C_i} |x - \mu_i|^2$$

   – Take partial derivative of μ_i and set to zero, we have

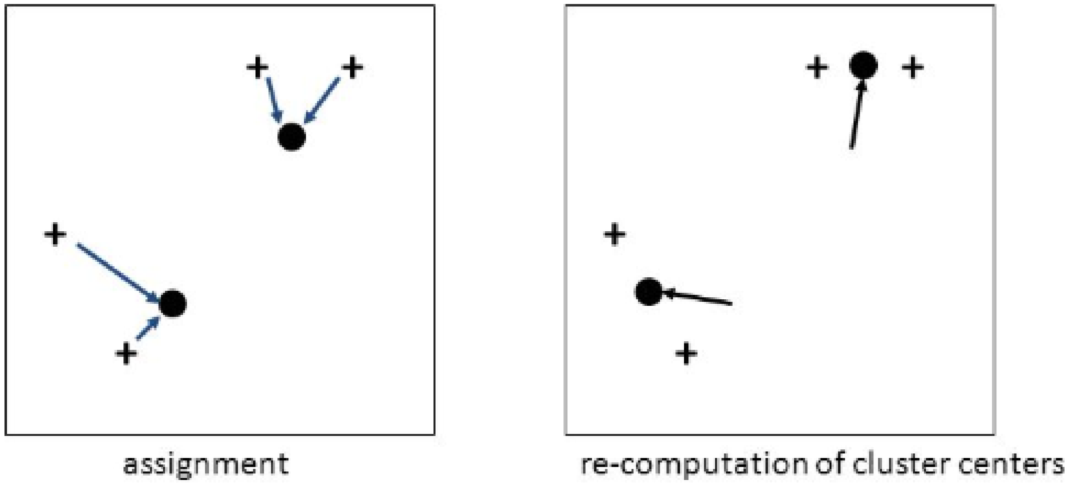
$$\mu_i = \frac{1}{|C_i|} \sum_{x \in C_i} x$$

Step 2 of kmeans

Kmeans takes an alternating optimization approach, each step is guaranteed to decrease the objective – thus guaranteed to converge

Kmeans Convergence

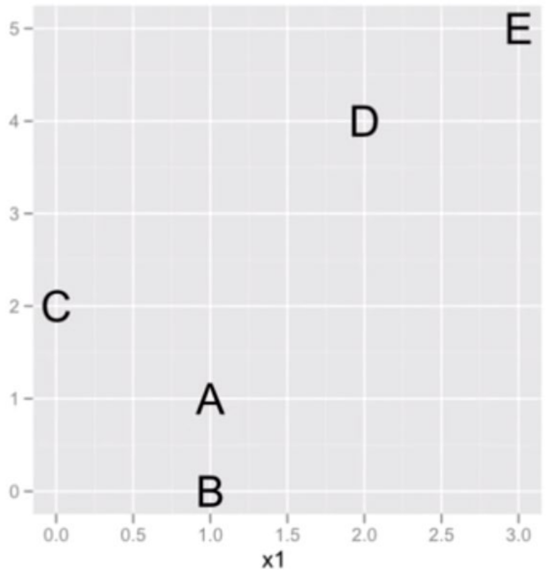
- Each iteration includes two steps
  - Assign points to clusters
  - Recompute the cluster centers

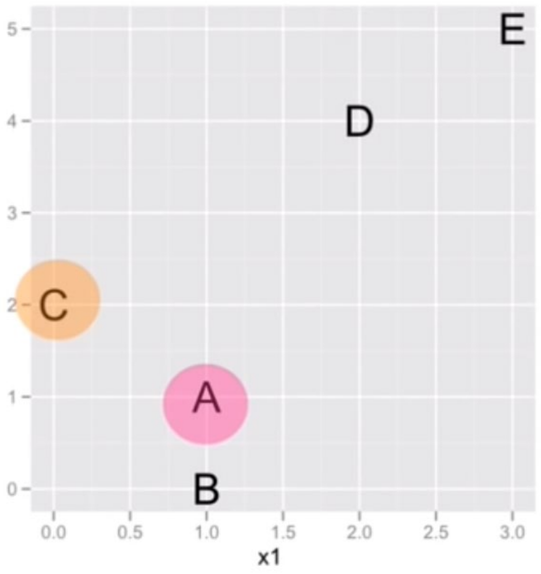


Example (1)

- Use K=2, suppose A and C are randomly selected as the initial means

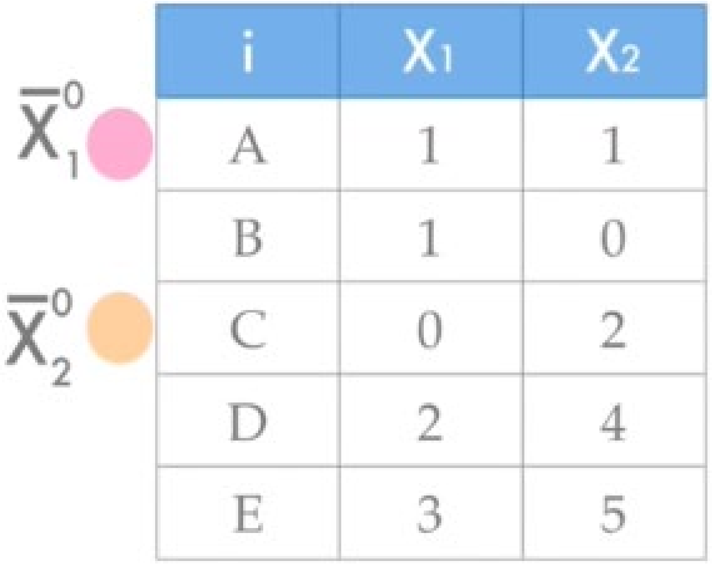
i	X_1	X_2
A	1	1
B	1	0
C	0	2
D	2	4
E	3	5



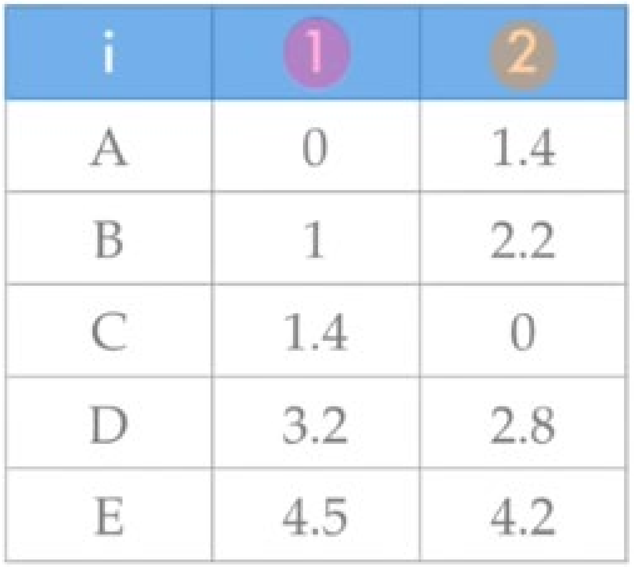


Example (1)

- Compute distances between each of the cluster means and all other points (L_2 Norm)



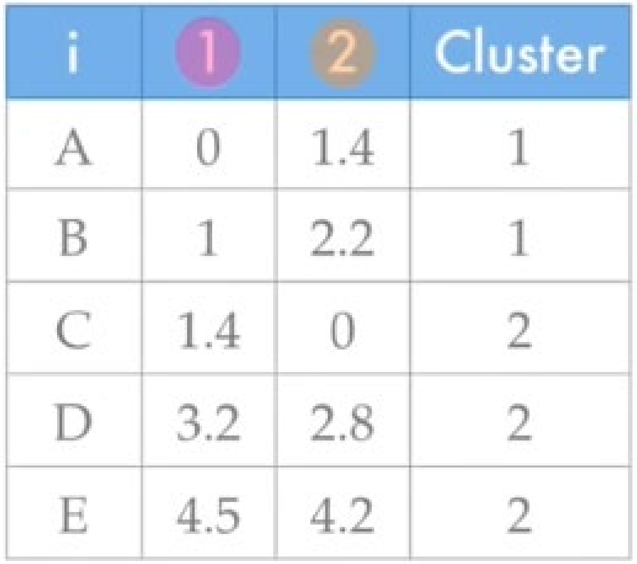
	i	X_1	X_2
$\bar{x}_1^0$ ●	A	1	1
	B	1	0
$\bar{x}_2^0$ ●	C	0	2
	D	2	4
	E	3	5



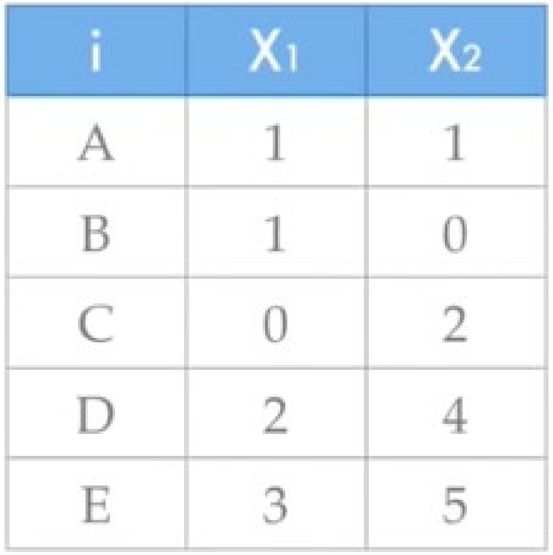
i	1	2
A	0	1.4
B	1	2.2
C	1.4	0
D	3.2	2.8
E	4.5	4.2

Example (1)

- Assign each case to the cluster having the closest mean. Recalculate the cluster means.



i	1	2	Cluster
A	0	1.4	1
B	1	2.2	1
C	1.4	0	2
D	3.2	2.8	2
E	4.5	4.2	2



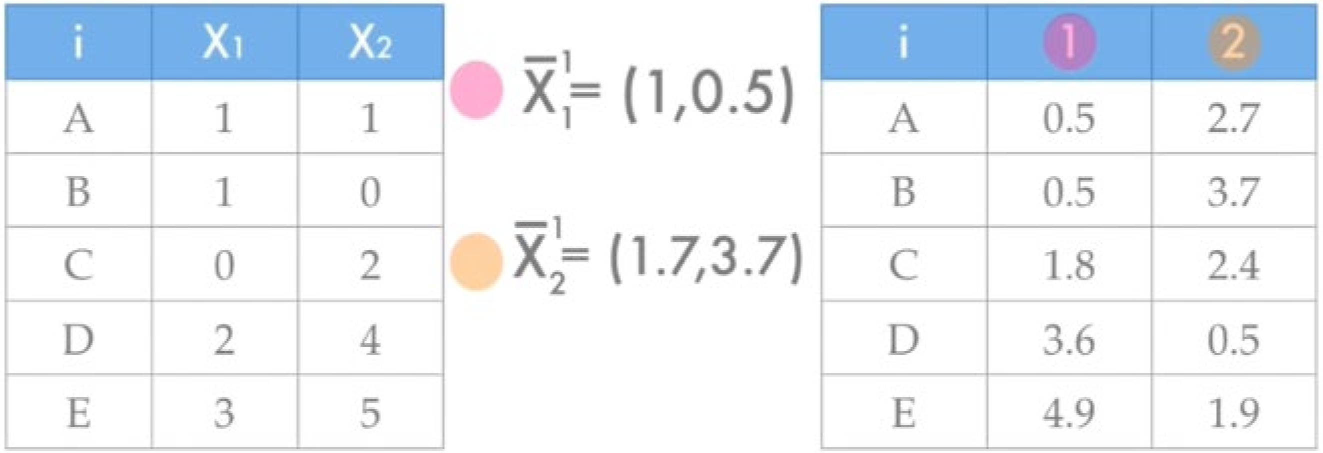
i	X_1	X_2
A	1	1
B	1	0
C	0	2
D	2	4
E	3	5

● $\bar{X}_1^1 = (1,0.5)$

● $\bar{X}_2^1 = (1.7,3.7)$

Example (1)

- Compute distances between each of the cluster means and all other points



i	X_1	X_2
A	1	1
B	1	0
C	0	2
D	2	4
E	3	5

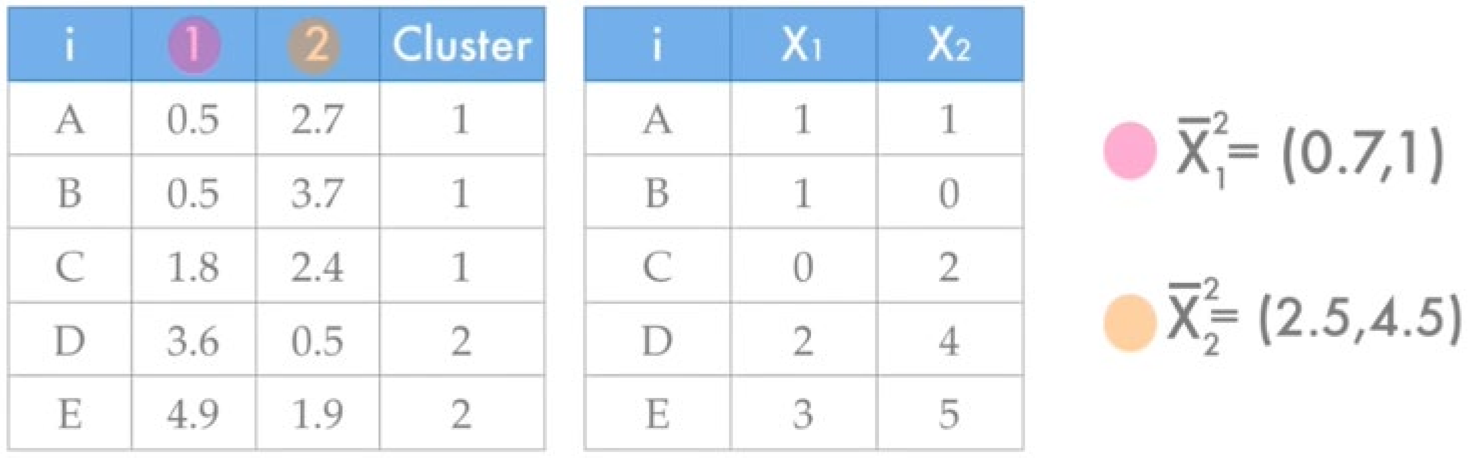
● $\bar{X}_1^1 = (1,0.5)$

● $\bar{X}_2^1 = (1.7,3.7)$

i	1	2
A	0.5	2.7
B	0.5	3.7
C	1.8	2.4
D	3.6	0.5
E	4.9	1.9

Example (1)

- Assign each case to the cluster having the closest mean. Recalculate the cluster means.
- Algorithm has converged-recalculating distances, reassigning cases to cluster results in no change. This is the final solution



i	1	2	Cluster
A	0.5	2.7	1
B	0.5	3.7	1
C	1.8	2.4	1
D	3.6	0.5	2
E	4.9	1.9	2

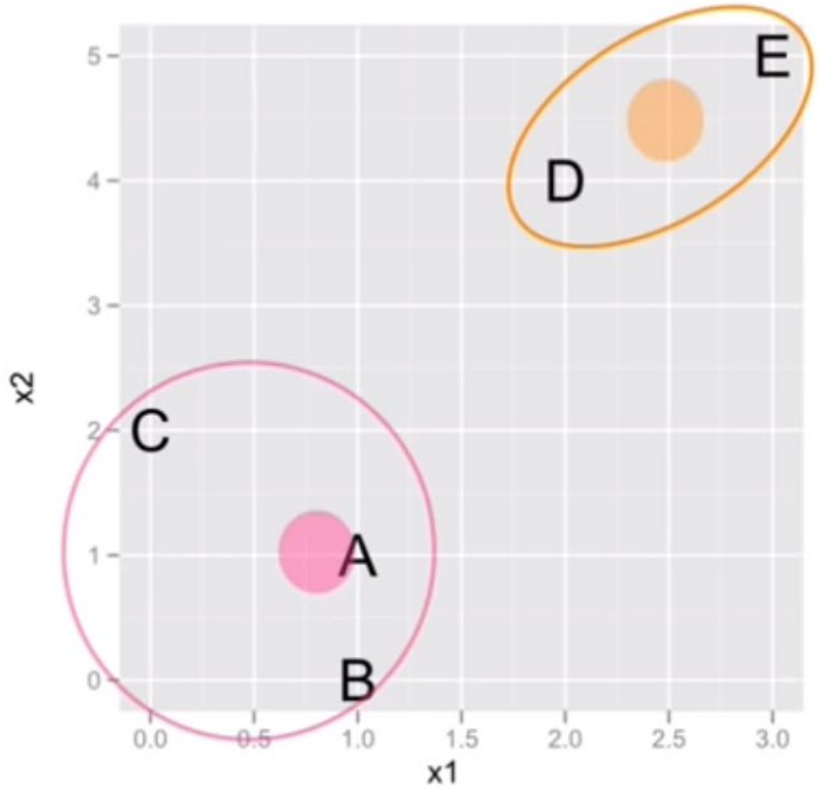
i	X_1	X_2
A	1	1
B	1	0
C	0	2
D	2	4
E	3	5

● $\bar{X}_1^2 = (0.7,1)$

● $\bar{X}_2^2 = (2.5,4.5)$

Example (1)

- Algorithm has converged-recalculating distances, reassigning cases to clusters results in no change, this is the final solution.

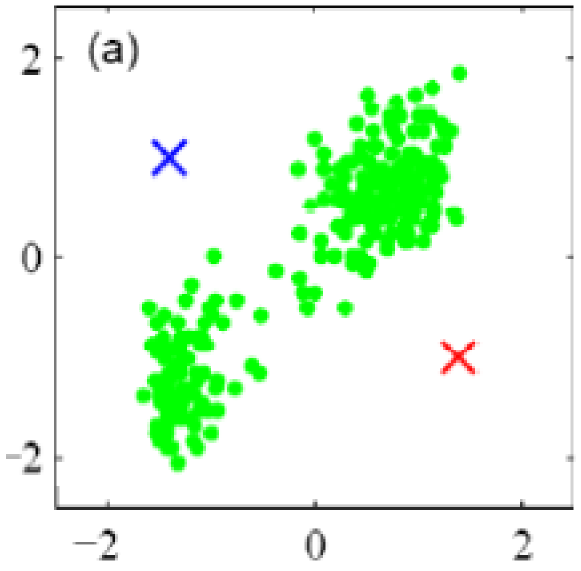


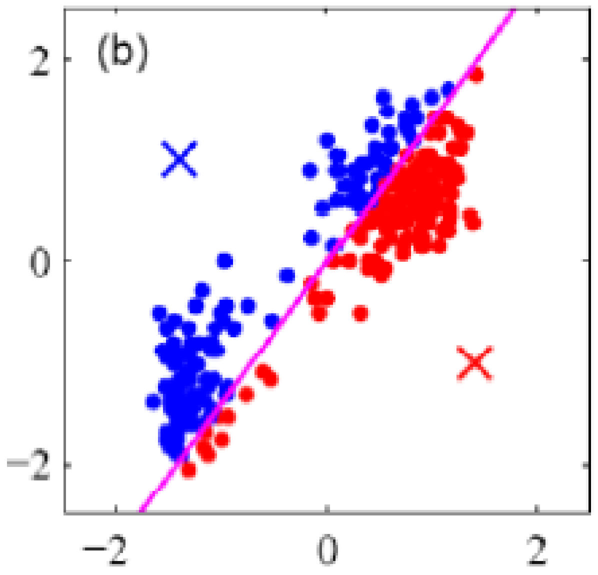
● $\overline{X}_1^2 = (0.7,1)$

● $\overline{X}_2^2 = (2.5,4.5)$

Visual Example (2)

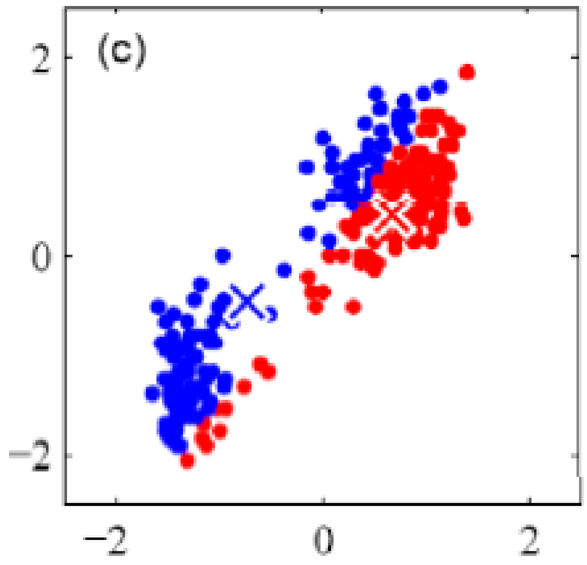
- Pick k random points as cluster centers (means), Here K=2
- Iterative step 1: assign data points to closest cluster center

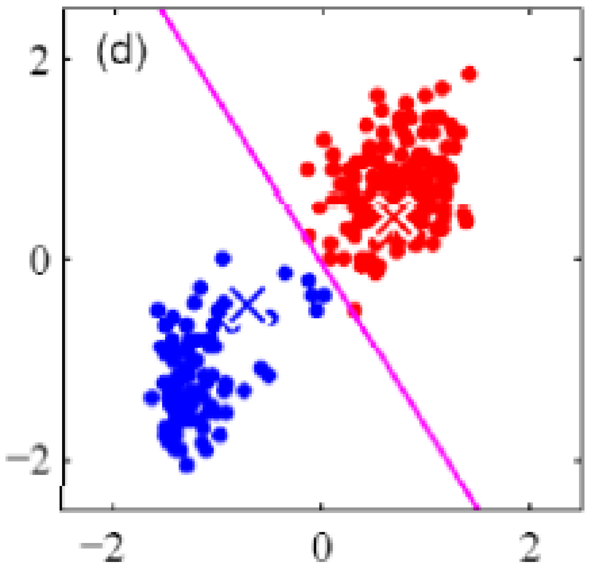




Example (2)

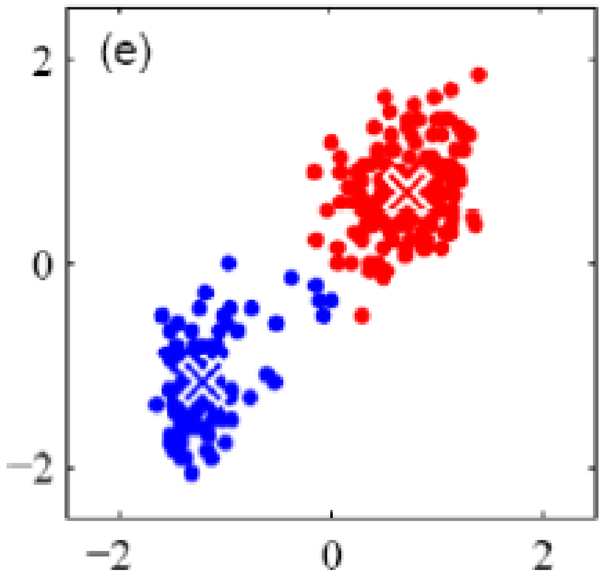
- Iterative step 2: change the cluster center to the average of the assigned points
- Assign data points to closest cluster center

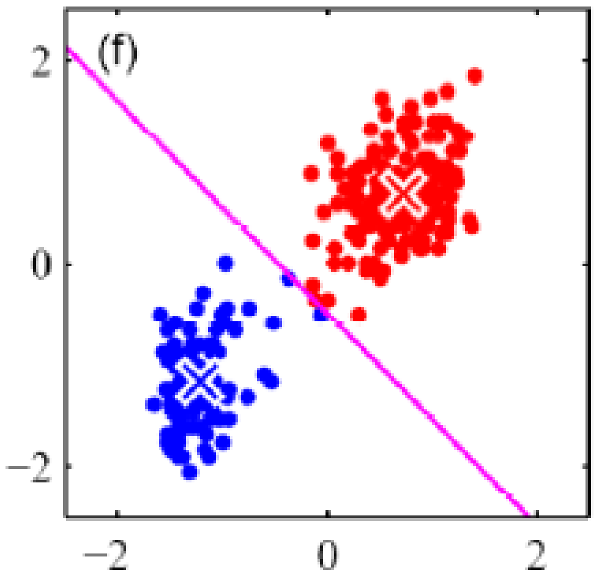




Example (2)

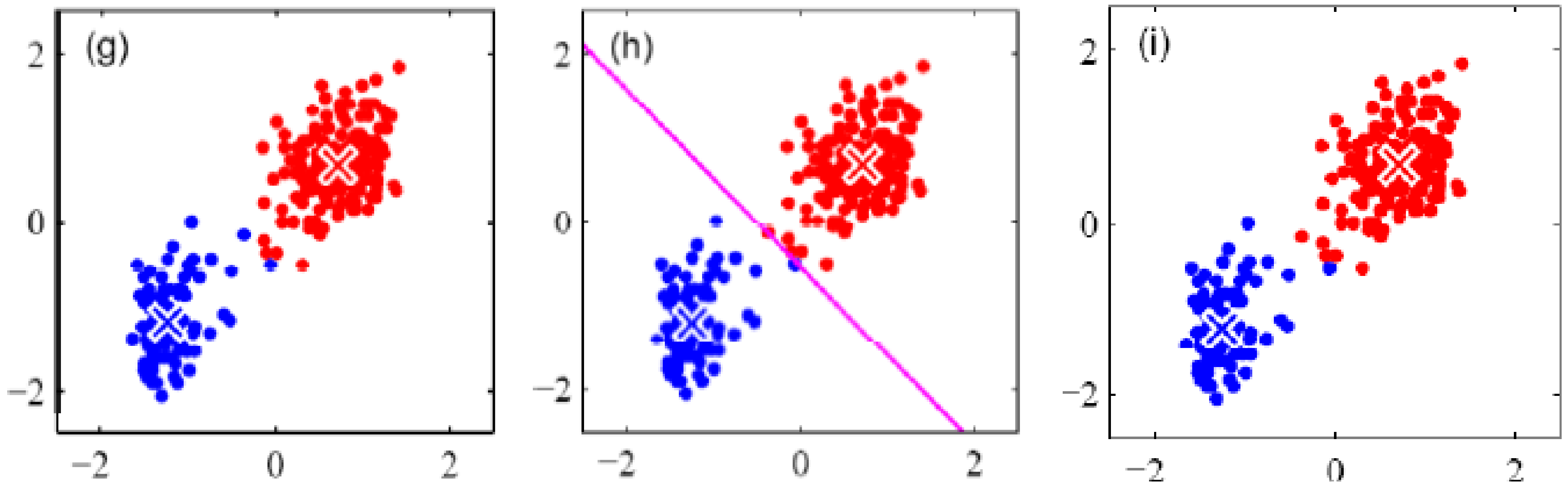
- Iterative step 3: change the cluster center to the average of the assigned points
- Assign data points to closest cluster center



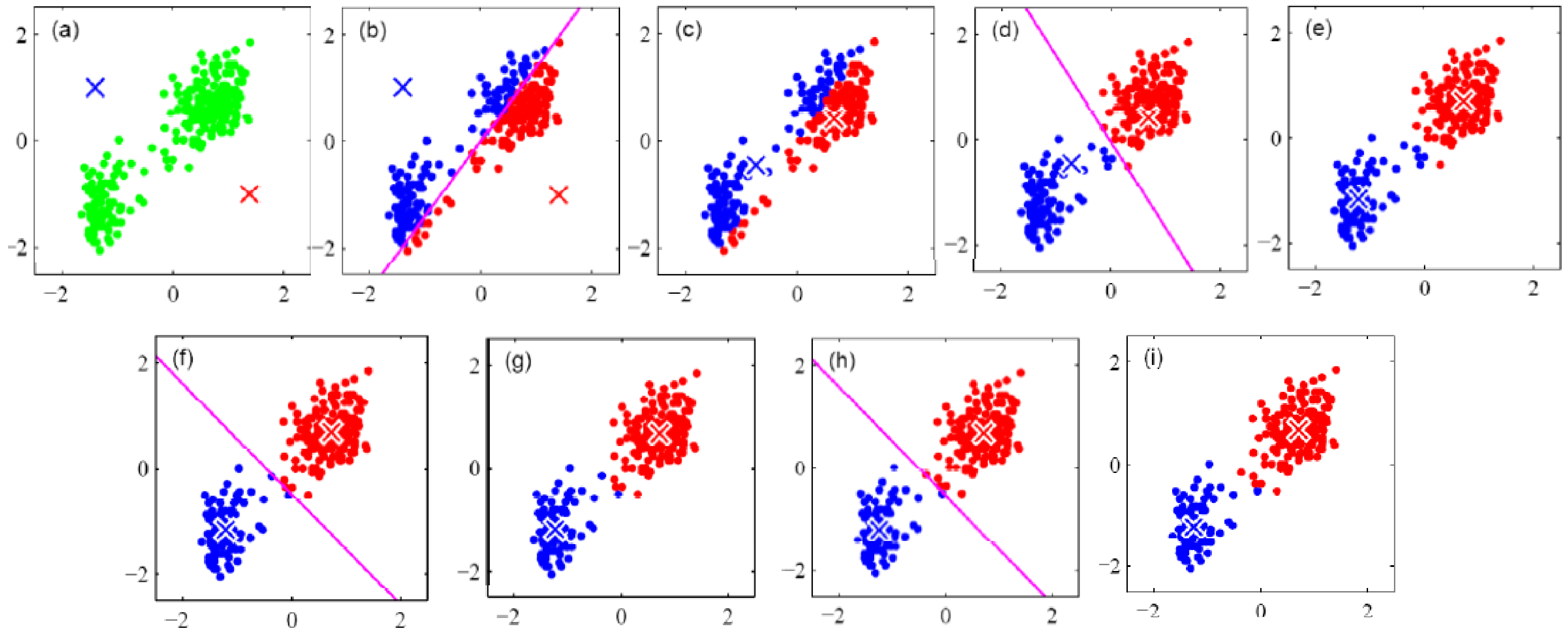


Example (2)

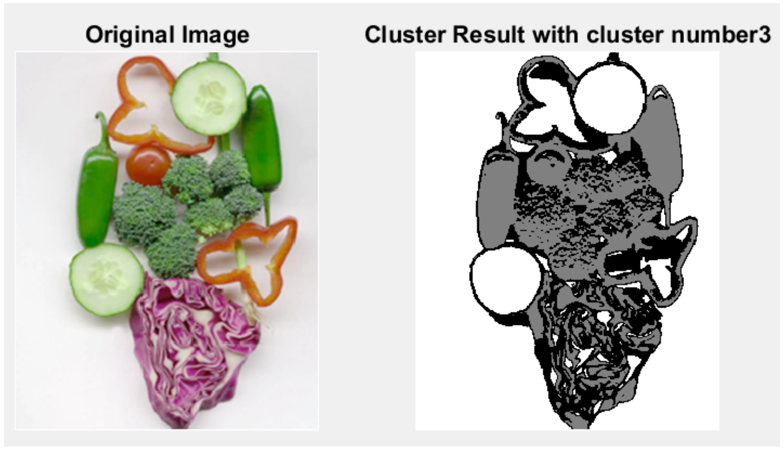
- Repeat until convergence

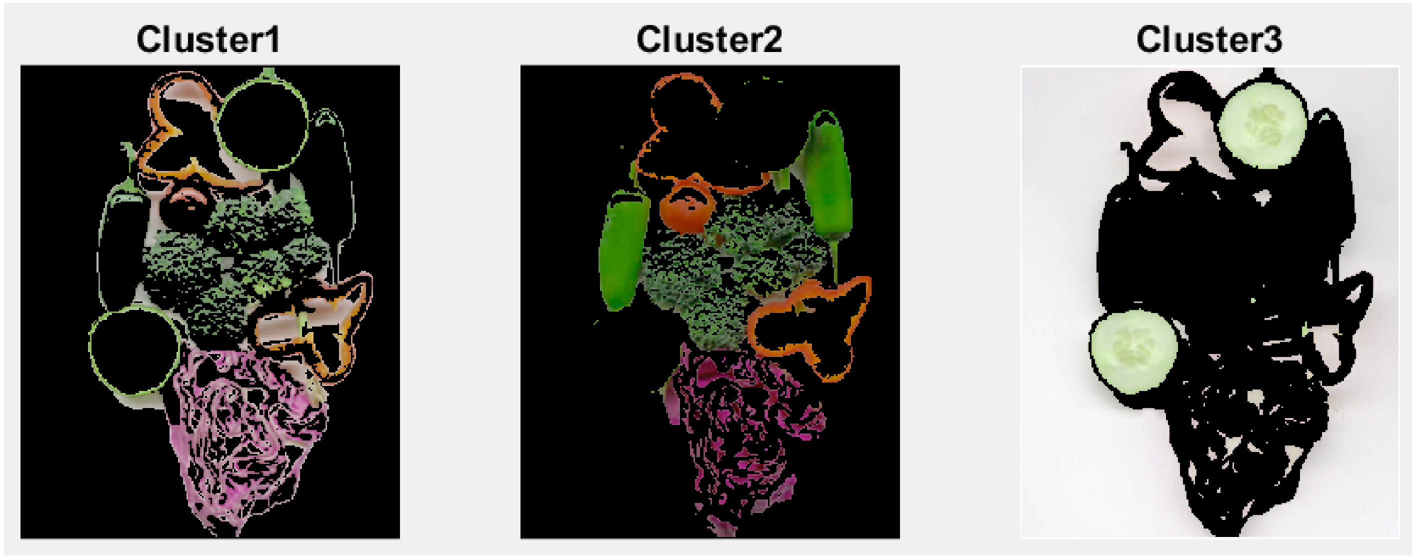


Example (2)

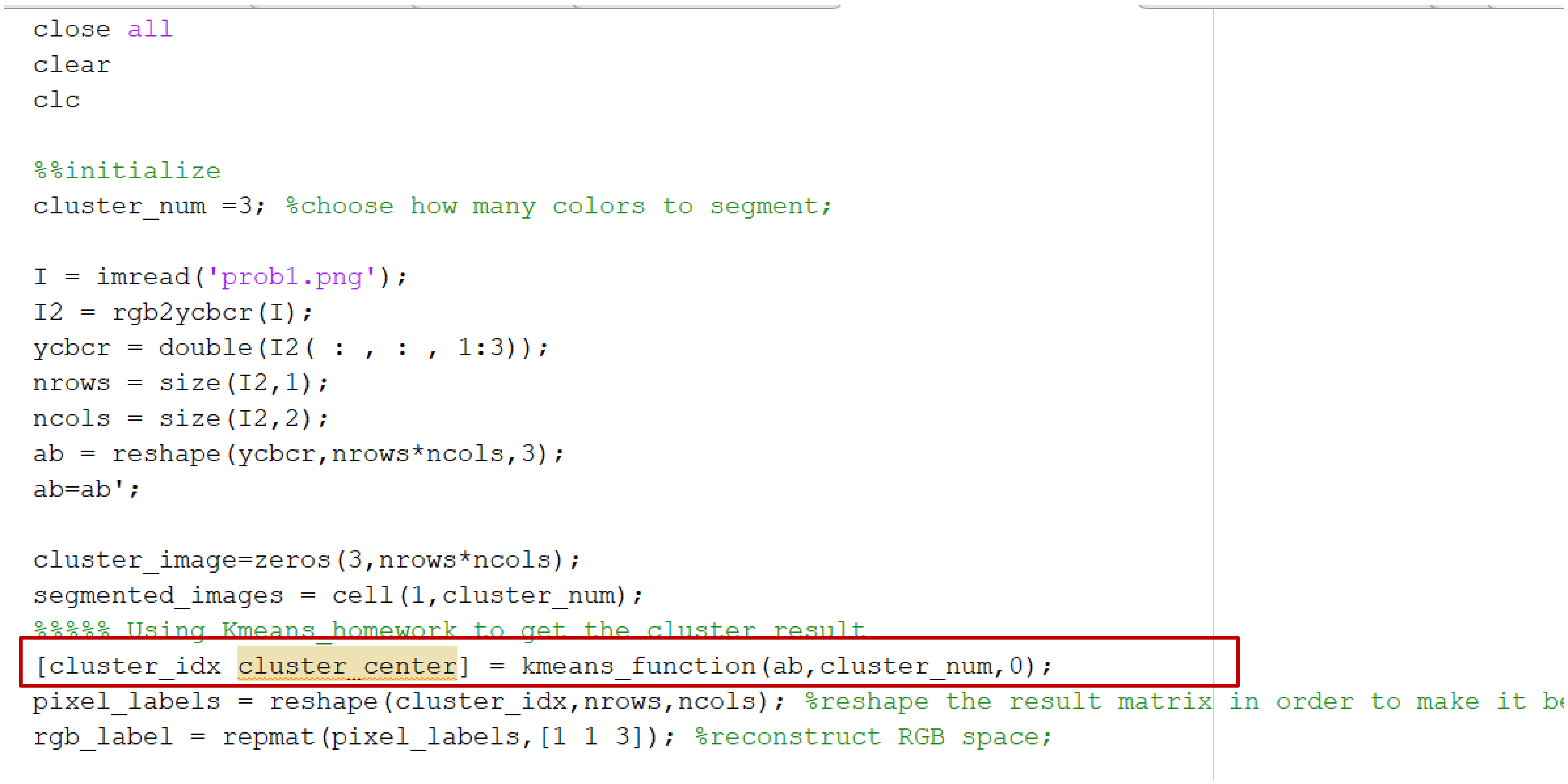


Example (3)


Original Image
Cluster Result with cluster number3

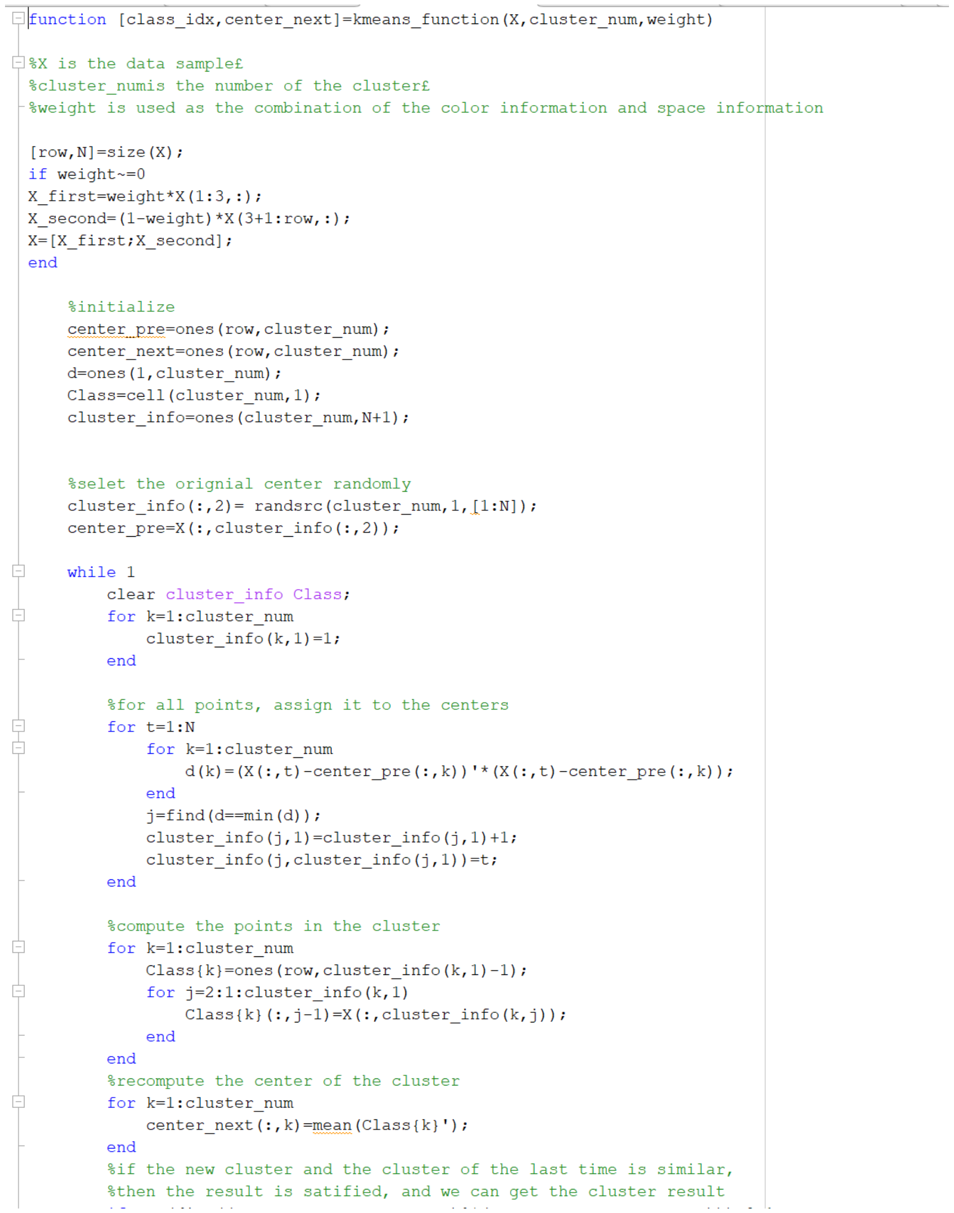

Cluster1
Cluster2
Cluster3

Example (3)



```
close all
clear
clc

%%initialize
cluster_num =3; %choose how many colors to segment;

I = imread('prob1.png');
I2 = rgb2ycbcr(I);
ycbcr = double(I2( : , : , 1:3));
nrows = size(I2,1);
ncols = size(I2,2);
ab = reshape(ycbcr,nrows*ncols,3);
ab=ab';

cluster_image=zeros(3,nrows*ncols);
segmented_images = cell(1,cluster_num);
%%%%% Using Kmeans_homework to get the cluster result
[cluster_idx cluster_center] = kmeans_function(ab,cluster_num,0);
pixel_labels = reshape(cluster_idx,nrows,ncols); %reshape the result matrix in order to make it be
rgb_label = repmat(pixel_labels,[1 1 3]); %reconstruct RGB space;
```

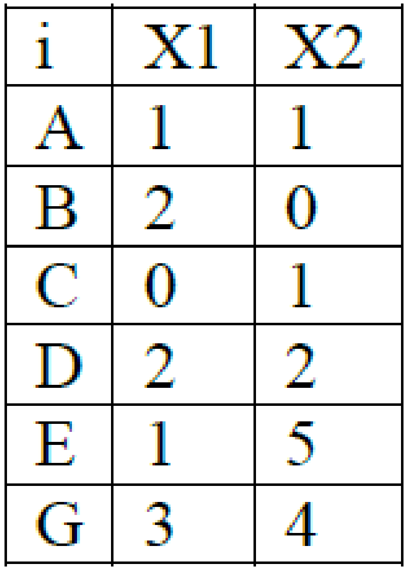
Example (3)



```
function [class_idx,center_next]=kmeans_function(X,cluster_num,weight)

%X is the data sample£
%cluster_numis the number of the cluster£
%weight is used as the combination of the color information and space information

[row,N]=size(X);
if weight~=0
X_first=weight*X(1:3,:);
X_second=(1-weight)*X(3+1:row,:);
X=[X_first;X_second];
end

    %initialize
    center_pre=ones(row,cluster_num);
    center_next=ones(row,cluster_num);
    d=ones(1,cluster_num);
    Class=cell(cluster_num,1);
    cluster_info=ones(cluster_num,N+1);


    %selet the orignial center randomly
    cluster_info(:,2)= randsrc(cluster_num,1,[1:N]);
    center_pre=X(:,cluster_info(:,2));

    while 1
        clear cluster_info Class;
        for k=1:cluster_num
            cluster_info(k,1)=1;
        end

        %for all points, assign it to the centers
        for t=1:N
            for k=1:cluster_num
                d(k)=(X(:,t)-center_pre(:,k))'*(X(:,t)-center_pre(:,k));
            end
            j=find(d==min(d));
            cluster_info(j,1)=cluster_info(j,1)+1;
            cluster_info(j,cluster_info(j,1))=t;
        end

        %compute the points in the cluster
        for k=1:cluster_num
            Class{k}=ones(row,cluster_info(k,1)-1);
            for j=2:1:cluster_info(k,1)
                Class{k}(:,j-1)=X(:,cluster_info(k,j));
            end
        end
        %recompute the center of the cluster
        for k=1:cluster_num
            center_next(:,k)=mean(Class{k}');
        end
        %if the new cluster and the cluster of the last time is similar,
        %then the result is satified, and we can get the cluster result
```

Example (4)

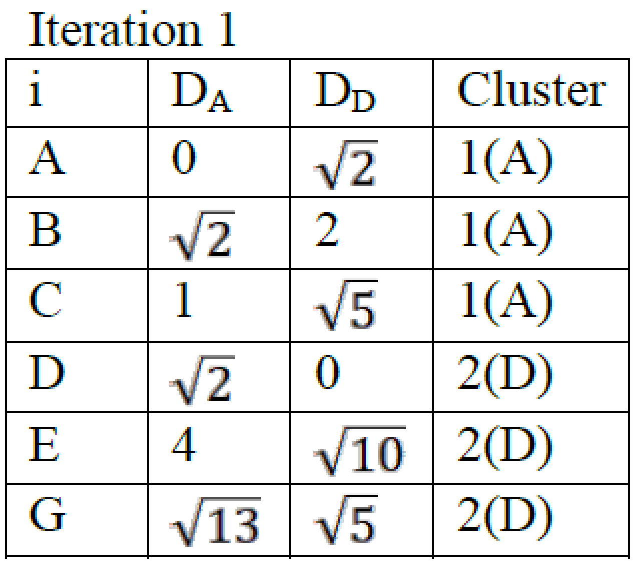
- Suppose A, D are randomly selected as initial means. Illustrate first iterations of Kmeans method by the following data input with K=2.



i	X1	X2
A	1	1
B	2	0
C	0	1
D	2	2
E	1	5
G	3	4

Example (4)

- After first iteration, New data center
  - 1 (1, 2/3)
  - 2 (2, 11/3)



Iteration 1

i	D_A	D_D	Cluster
A	0	$\sqrt{2}$	1(A)
B	$\sqrt{2}$	2	1(A)
C	1	$\sqrt{5}$	1(A)
D	$\sqrt{2}$	0	2(D)
E	4	$\sqrt{10}$	2(D)
G	$\sqrt{13}$	$\sqrt{5}$	2(D)

Evaluating K-means Clusters

- Most common measure is Sum of Squared Error (SSE)
  - For each point, the error is the distance to the nearest cluster
  - To get SSE, we square these errors and sum them.

$$SSE = \sum_{i=1}^{K} \sum_{x \in C_i} dist^2(m_i, x)$$

  - x is a data point in cluster C_i and m_i is the representative point for cluster C_i
  - Given two clusters, we can choose the one with the smallest error

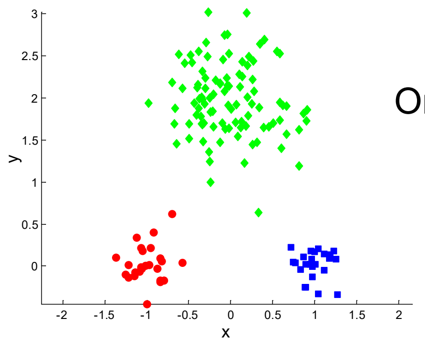
- One easy way to reduce SSE is to increase K, i.e. the number of clusters
  - A good clustering with smaller K can have a lower SSE than a poor clustering with higher K

K-means

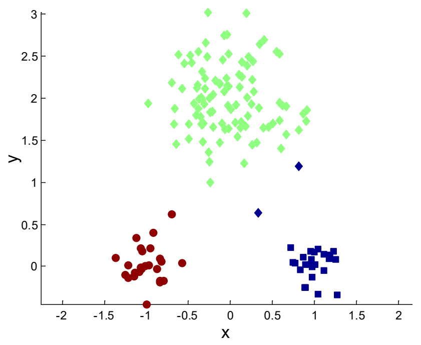
- Advantages
  - Easy for implementation and high-speed performance
  - Measurable and efficient in large data collection
- Disadvantages
  - Selection of optimal number of clusters is difficult
  - Selection of the initial centroids is random, and different initializations lead to different results.

Initial Centroids Problems

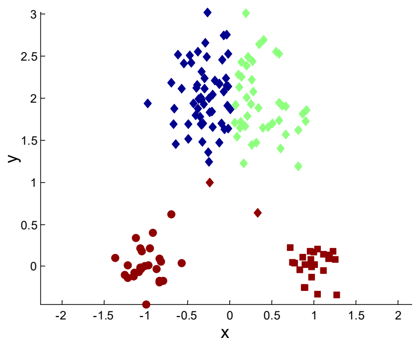
- With different initial centroids, the final clustering results are different.



Original Points

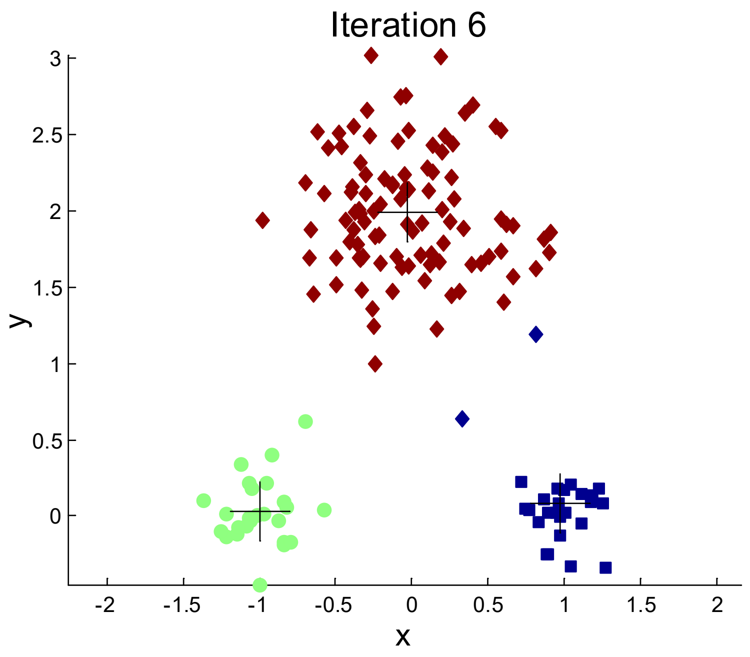


Optimal Clustering

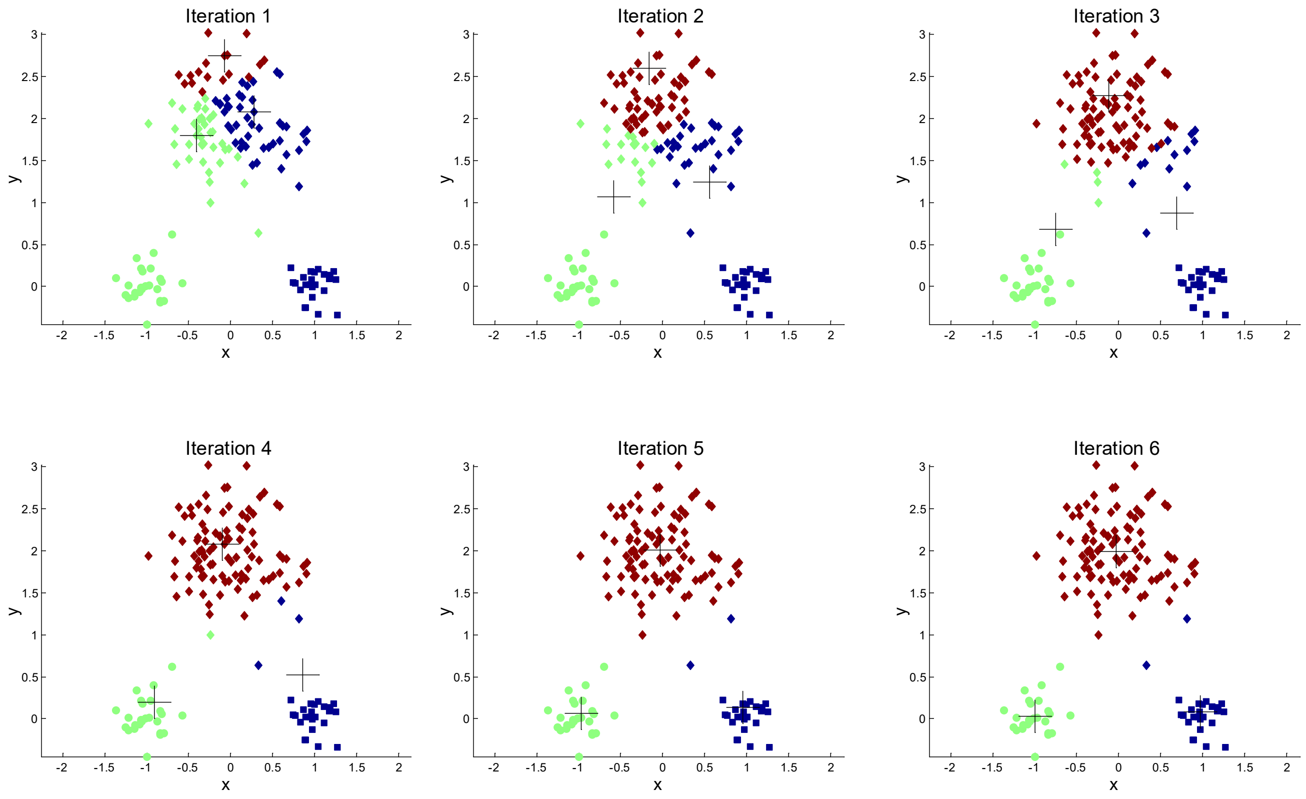


Sub-optimal Clustering

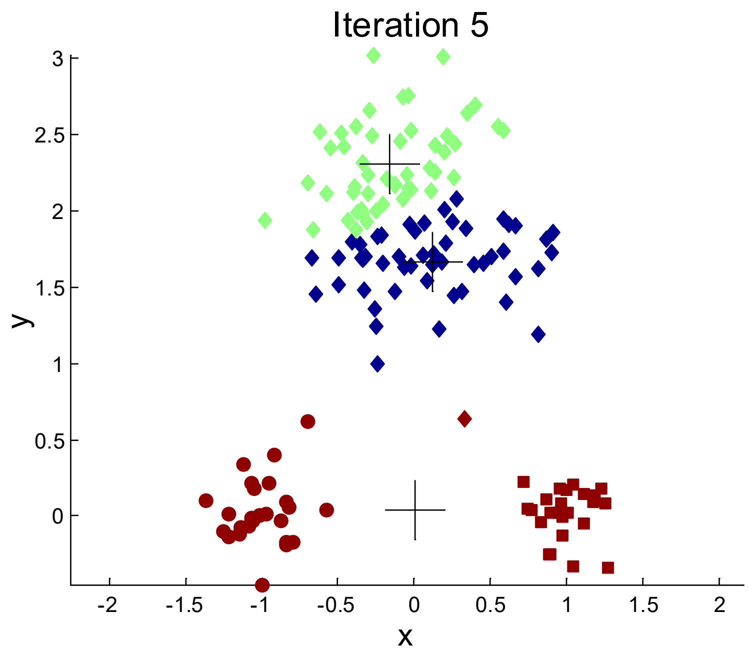
Initial Centroids Problems (Case i)



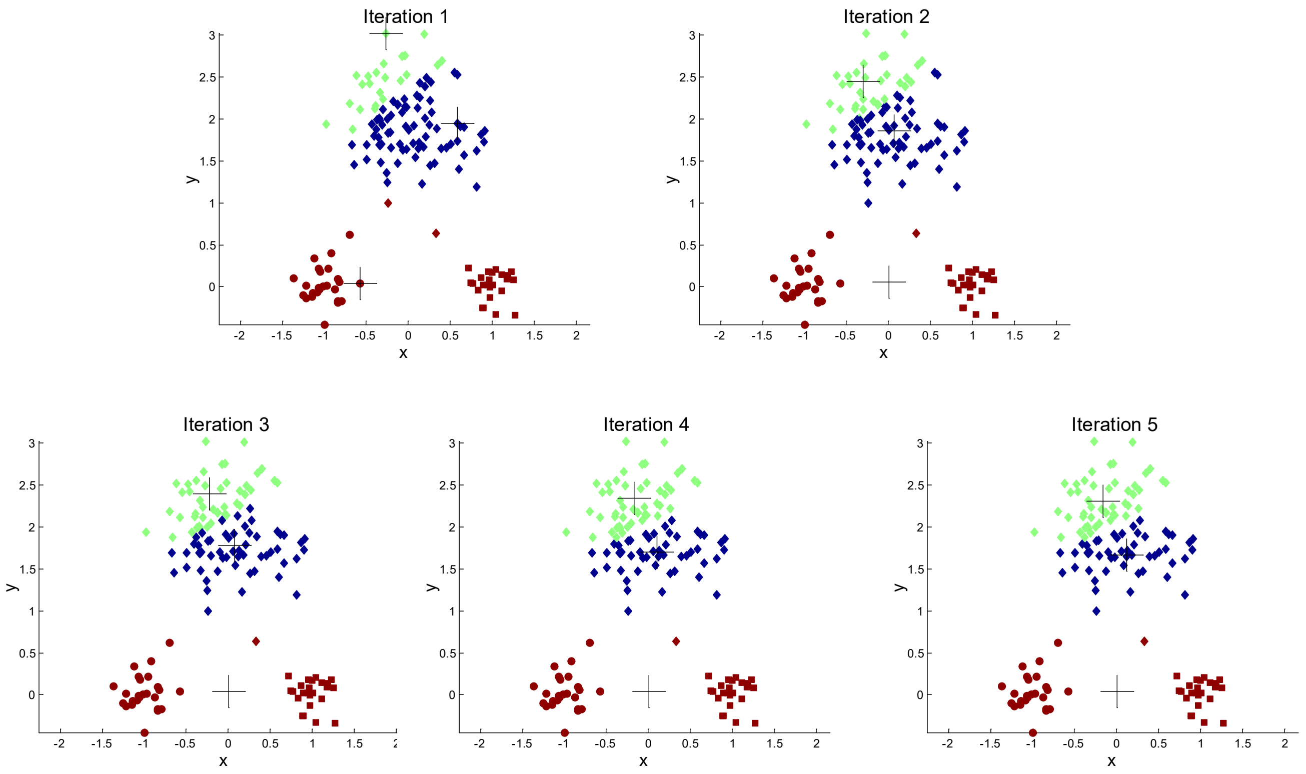
Initial Centroids Problems (Case i)



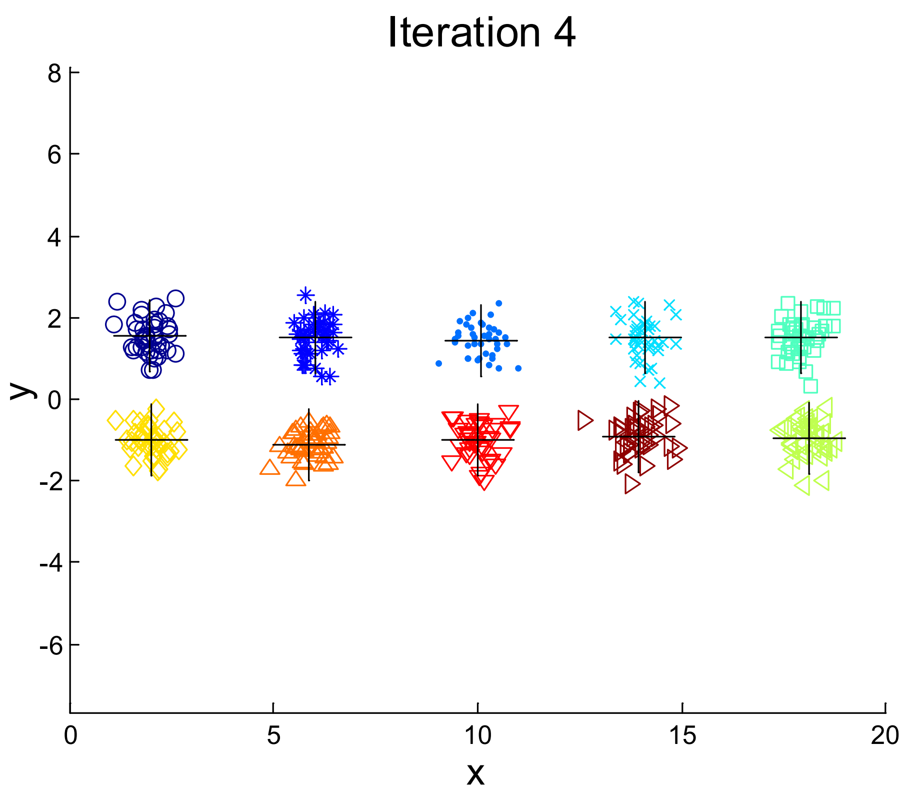
Initial Centroids Problems (Case ii)



Initial Centroids Problems (Case ii)

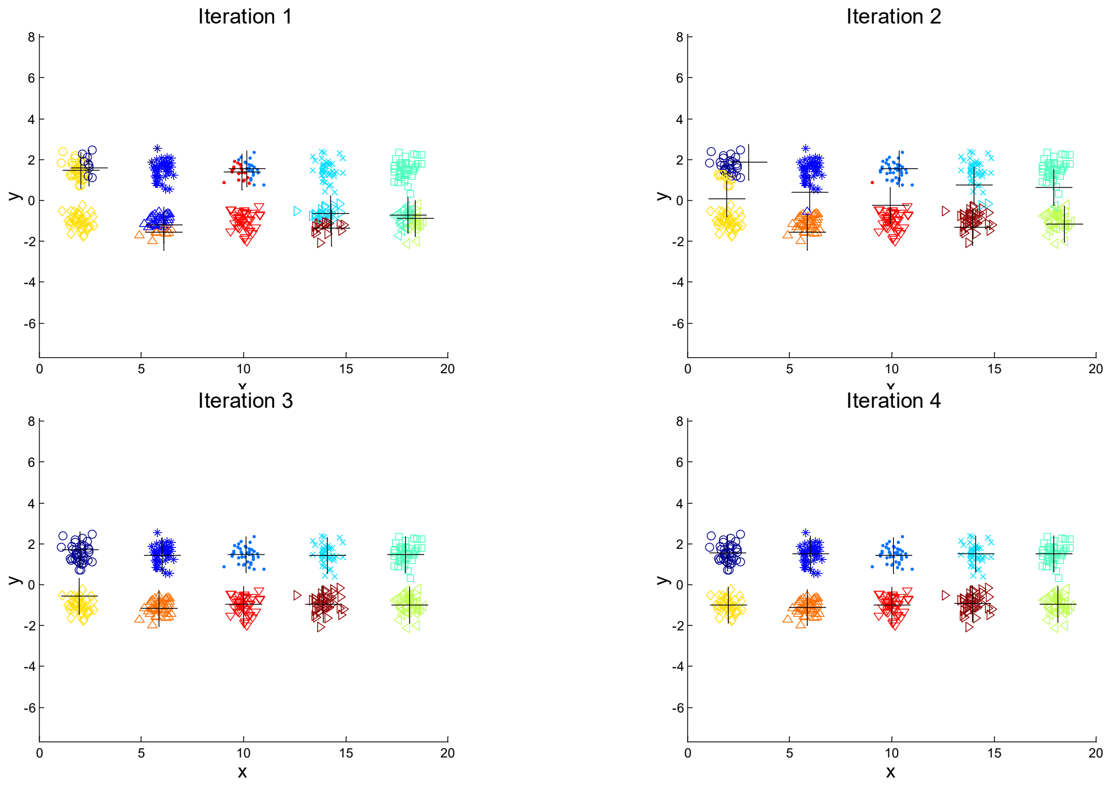


10 Clusters Example (Good Clusters)



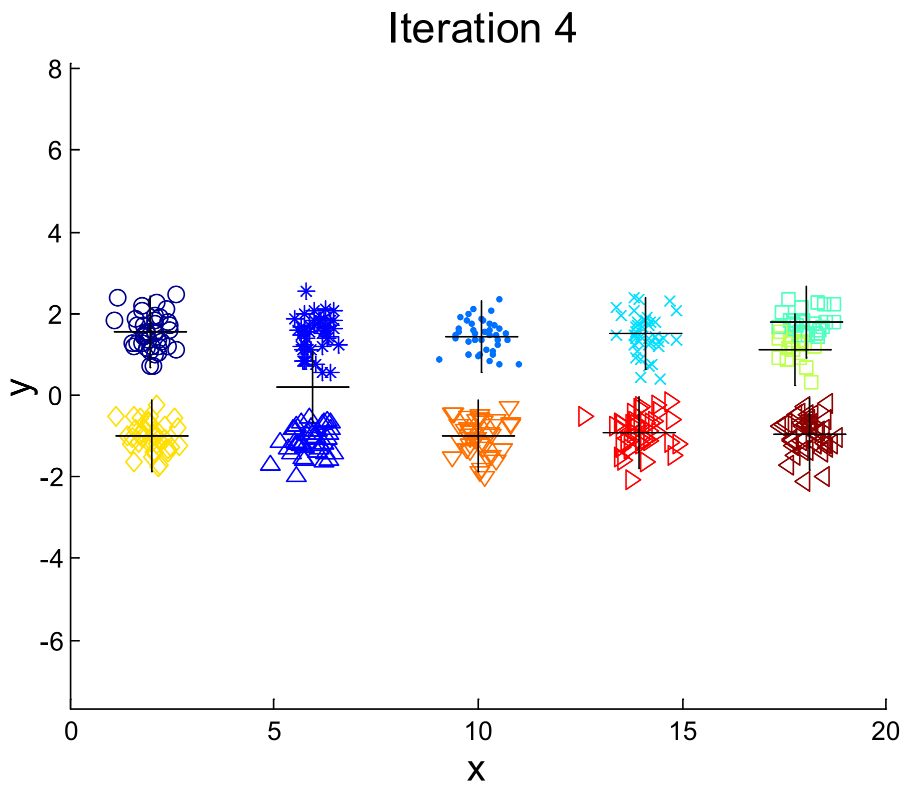
Starting with two initial centroids in one cluster of each pair of clusters

10 Clusters Example (Good Clusters)



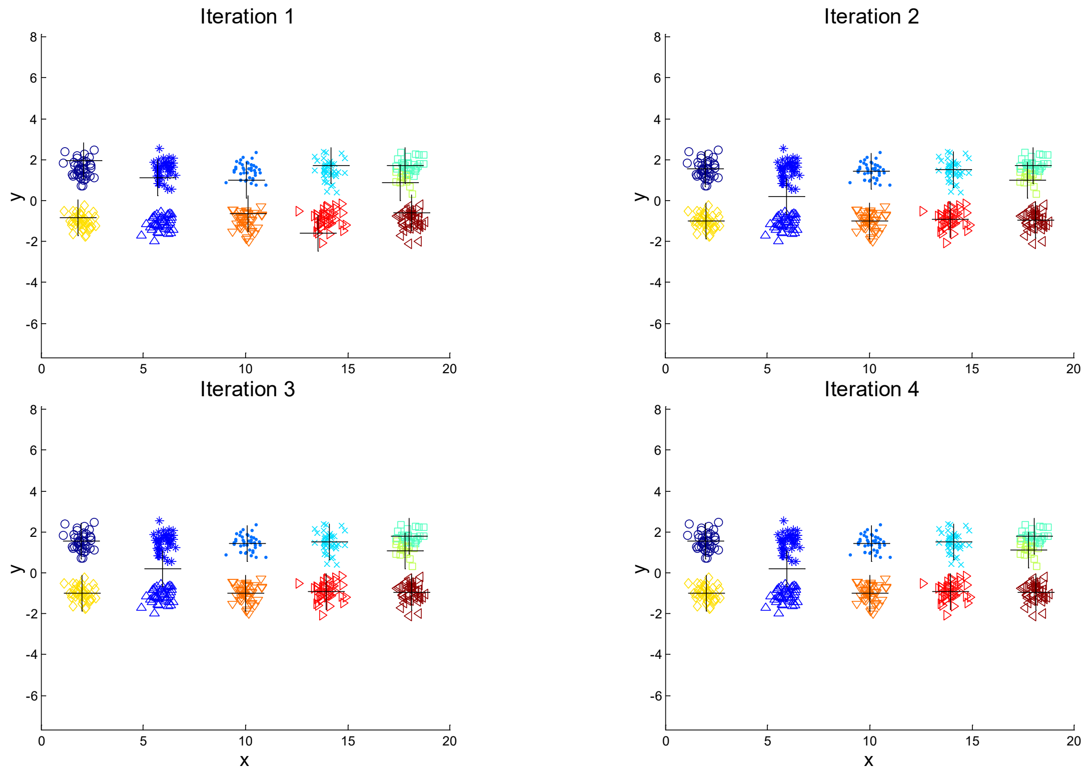
Starting with two initial centroids in one cluster of each pair of clusters

10 Clusters Example (Bad Clusters)



Starting with some pairs of clusters having three initial centroids, while other have only one.

10 Clusters Example (Bad Clusters)



Starting with some pairs of clusters having three initial centroids, while other have only one.

Solutions to Initial Centroids Problem

- Multiple runs
  - Helps, but probability is not on your side
- Sample and use hierarchical clustering to determine initial centroids
- Select more than k initial centroids and then select among these initial centroids
  - Select most widely separated
- Post-processing
- Bisecting K-means
  - Not as susceptible to initialization issues

Pre-processing and Post-processing

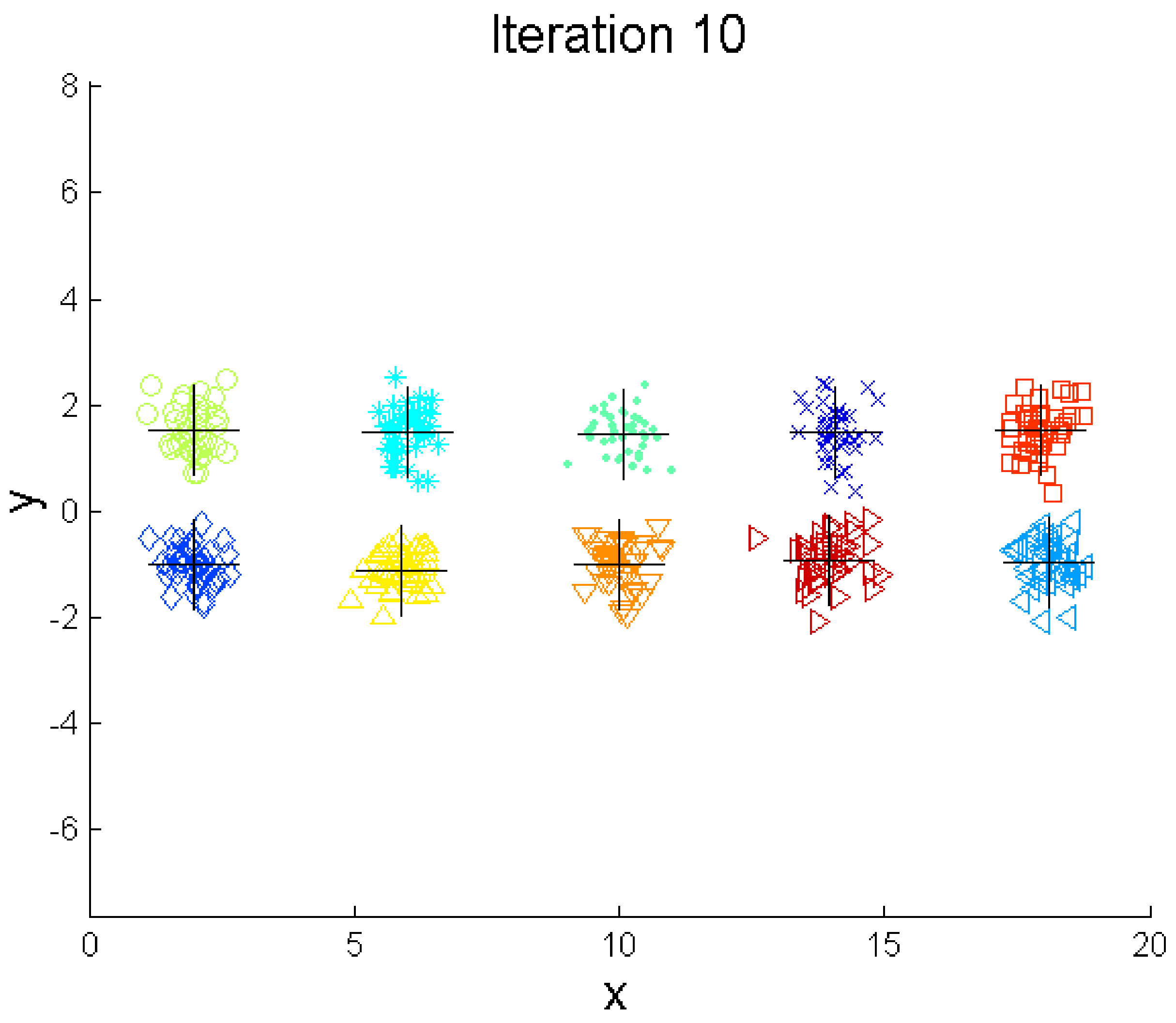
- Pre-processing
  - Normalize the data
  - Eliminate outliers
- Post-processing
  - Eliminate small clusters that may represent outliers
  - Split ‘loose’ clusters, i.e., clusters with relatively high SSE
  - Merge clusters that are ‘close’ and that have relatively low SSE
  - Can use these steps during the clustering process

Bisecting K-means

- Bisecting K-means algorithm
  - Variant of K-means that can produce a partitional or a hierarchical clustering

```
1: Initialize the list of clusters to contain the cluster containing all points.
2: repeat
3:     Select a cluster from the list of clusters
4:     for $i = 1$ to $number\_of\_iterations$ do
5:         Bisect the selected cluster using basic K-means
6:     end for
7:     Add the two clusters from the bisection with the lowest SSE to the list of clusters.
8: until Until the list of clusters contains $K$ clusters
```

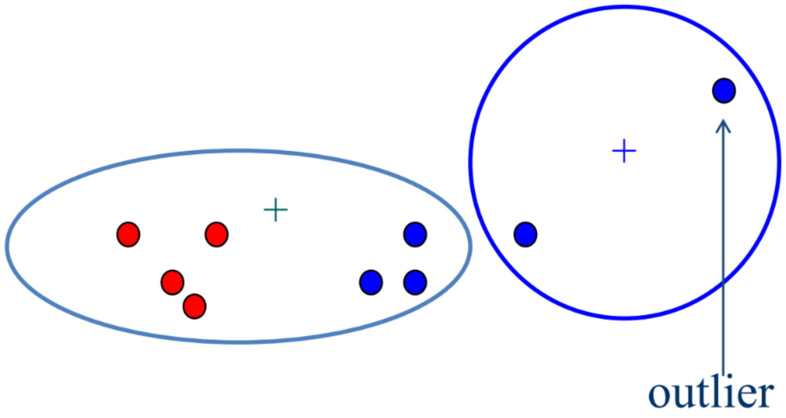
Bisecting K-means Example



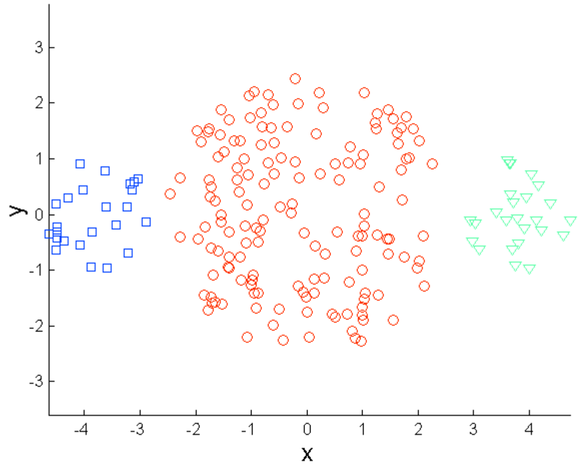
Limitations of K-means

- K-means has problems when clusters are of differing
    - Sizes
    - Densities
    - Non-globular shapes

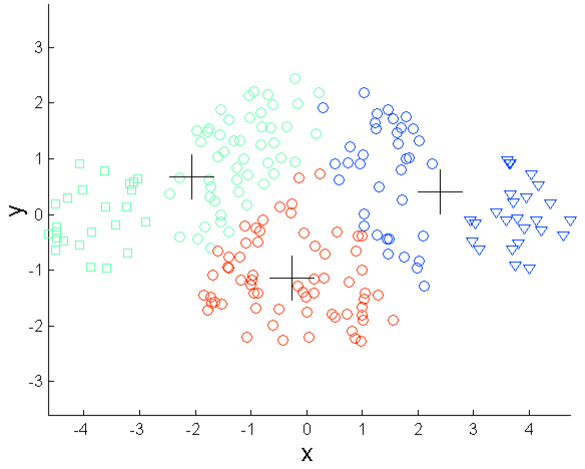
- K-means has problems when the data contains outliers
    - May substantially distort the distribution of the data



Limitations of K-means: Differing Sizes

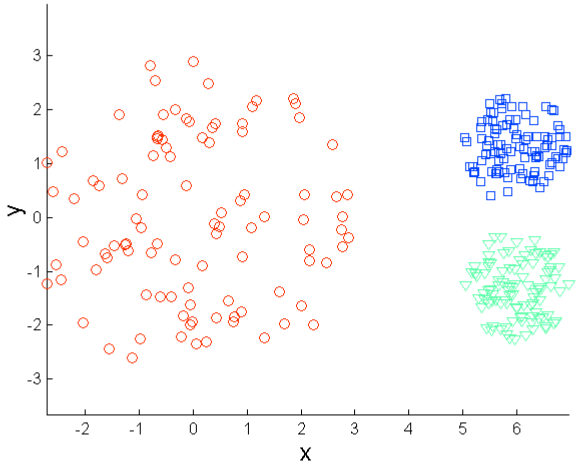


Original Points

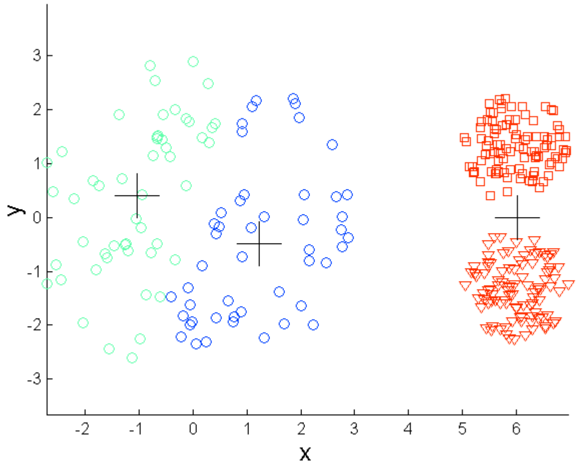


K-means (3 Clusters)

Limitations of K-means: Differing Density

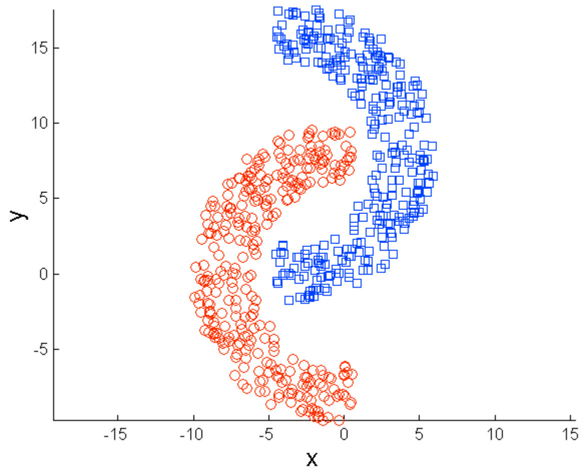


Original Points

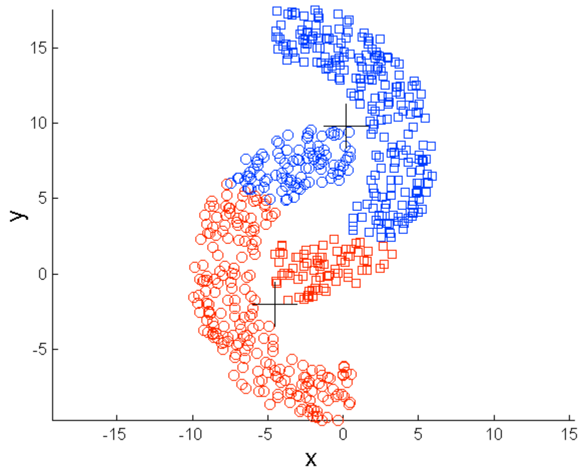


K-means (3 Clusters)

Limitations of K-means: Non-globular Shapes



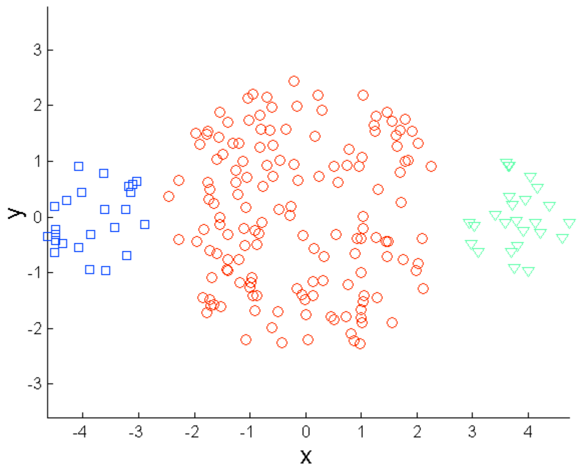
Original Points



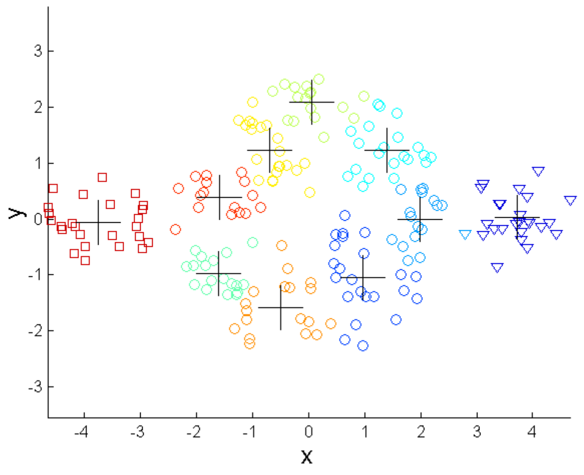
K-means (2 Clusters)

Overcoming K-means Limitations

- One solution is to use many clusters.
  - Find parts of clusters, but need to put together.



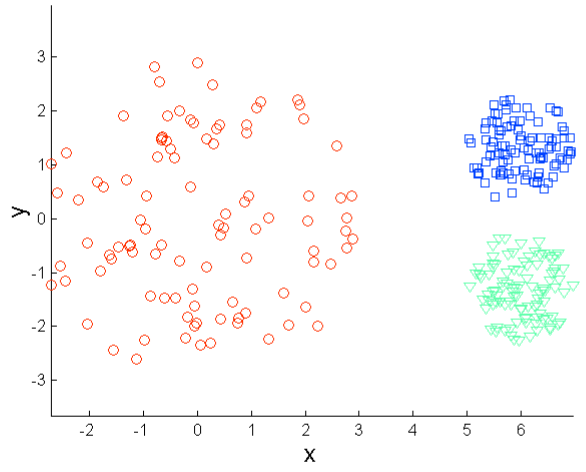
Original Points



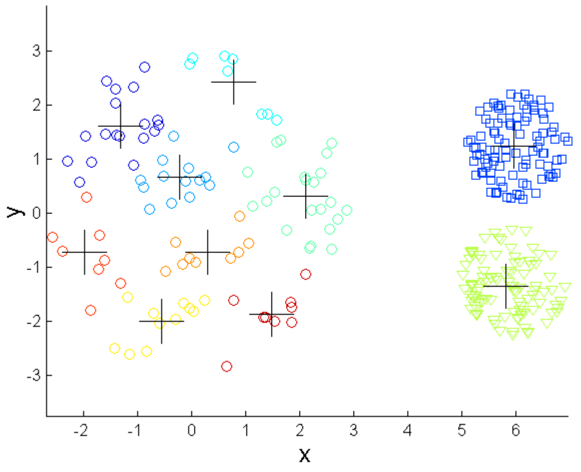
K-means Clusters

Overcoming K-means Limitations

- One solution is to use many clusters.
  - Find parts of clusters, but need to put together.



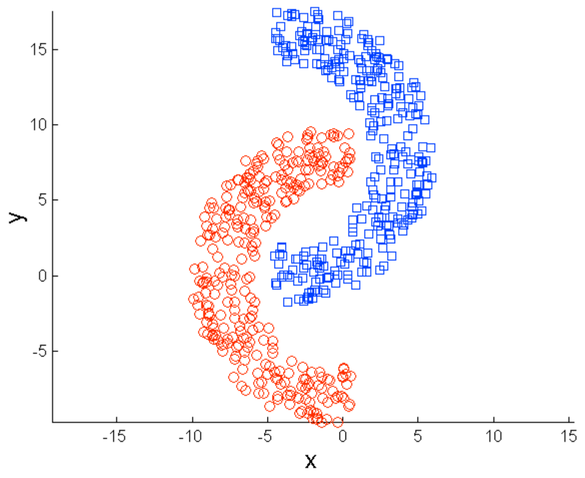
Original Points



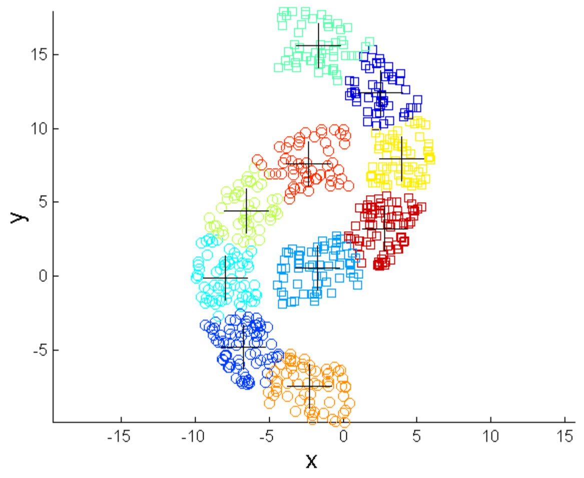
K-means Clusters

Overcoming K-means Limitations

- One solution is to use many clusters.
  - Find parts of clusters, but need to put together.



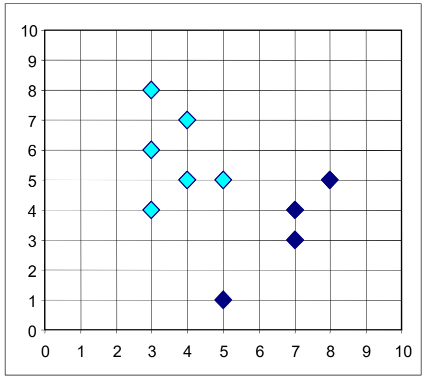
Original Points

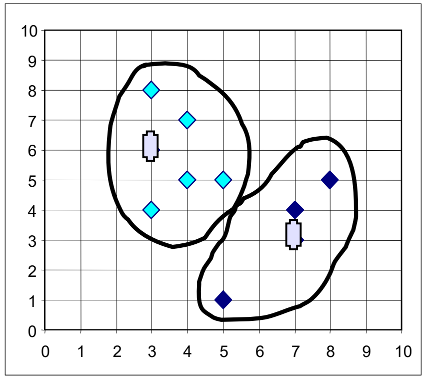


K-means Clusters

Outlier Problem to K-means

- The k-means algorithm is sensitive to outliers !
  - Since an object with an extremely large value may substantially distort the distribution of the data.
- Solution: Instead of taking the mean value of the object in a cluster as a reference point, medoids can be used, which is the most centrally located object in a cluster.





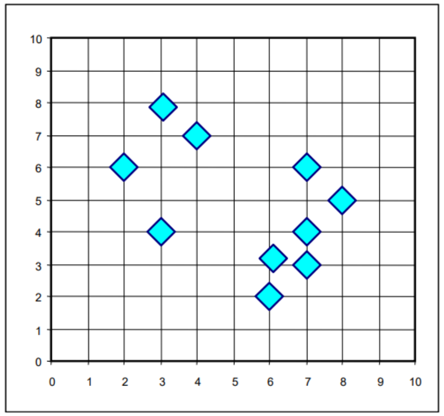
Outline

- Introduction of clusters
- Introduction of clustering
- K-means
- K-medoids
    - PAM&CLARA & CLARANS

The K-Medoids Clustering Method

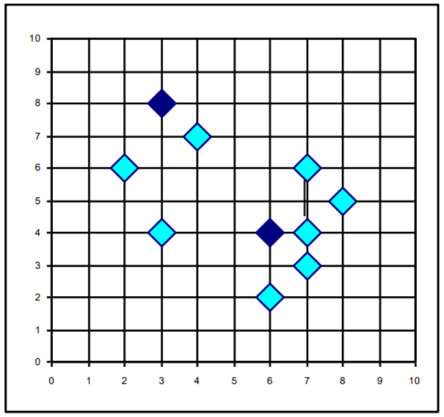
- Find *representative* objects, called medoids, in clusters
- *PAM* (Partitioning Around Medoids, 1987)
  - starts from an initial set of medoids and iteratively replaces one of the medoids by one of the non-medoids if it improves the total distance of the resulting clustering.
  - *PAM* works effectively for small data sets, but does not scale well for large data sets.
- *CLARA* (Kaufmann & Rousseeuw, 1990)
- *CLARANS* (Ng & Han, 1994): Randomized sampling

PAM: A Typical K-Medoids Algorithm



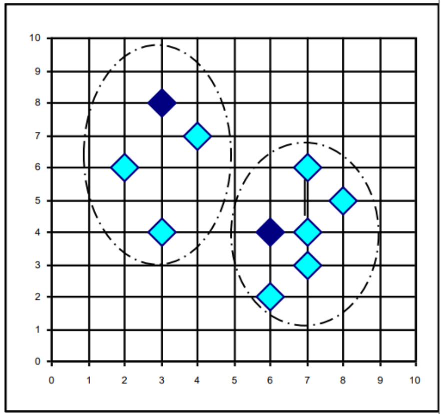
K=2

Arbitrary choose k object as initial medoids

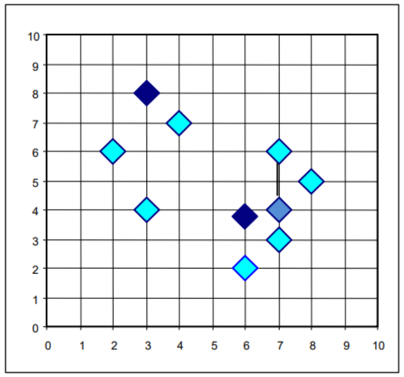


Assign each remaining object to nearest medoids

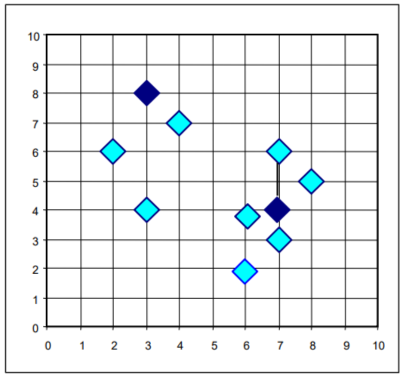
Total Cost = 20



Randomly select a nonmedoid object,O_{ramdom}



Compute total cost of swapping



Swapping O and O_{ramdom}

If quality is improved.

Do loop

Until no change

PAM (Partitioning Around Medoids)

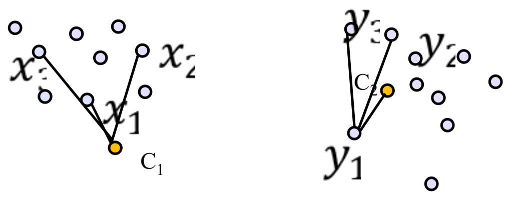
- Use real objects to represent the clusters (called medoids)
  1. Select ***k*** representative objects arbitrarily
  2. For each pair of selected object (***i***) and non-selected object (***h***), calculate the Total swapping Cost (TC_{ih})
  3. For each pair of ***i*** and ***h***,
     1. If $TC_{ih} < 0$, ***i*** is replaced by ***h***
     2. Then assign each non-selected object to the most similar representative object
  4. repeat steps 2-3 until there is no change in the medoids or in TC_{ih}.

Total swapping Cost (TC_{ih})

- Total swapping cost $TC_{ih}=\sum_j C_{jih}$
- Where C_{jih} is the cost of swapping *i* with *h* for all non medoid objects *j*
- C_{jih} will vary depending upon different cases

Sum of Squared Error (SSE)

$$\text{SSE }(E) = \sum_{i=1}^{k} \sum_{j \in C_i} d^2(j, i)$$

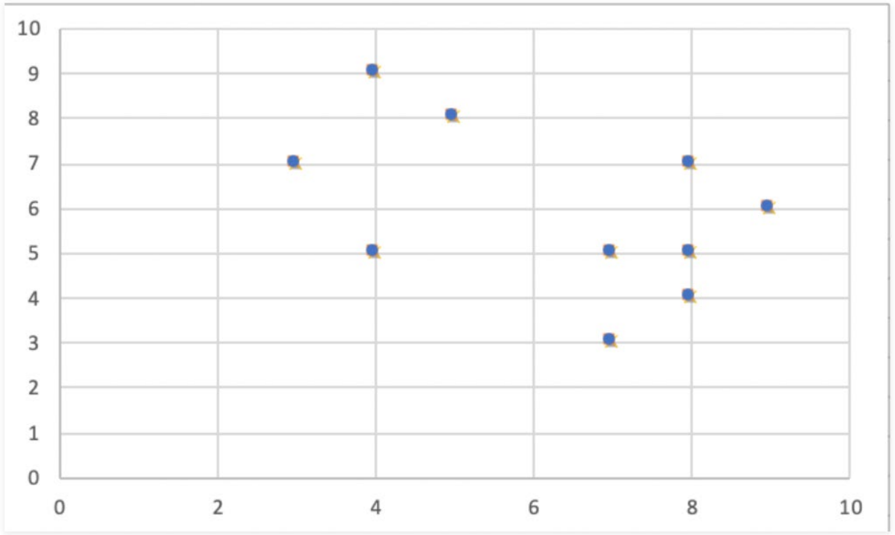


$$\text{SSE }(E) = d^2(x_1,\mathbf{c_1}) + d^2(x_2,\mathbf{c_1}) + \cdots + d^2\left(x_{n_1},\mathbf{c_1}\right) +$$
$$d^2(y_1,\mathbf{c_2}) + d^2(y_2,\mathbf{c_2}) + \cdots + d^2(y_{n_2},\mathbf{c_2})$$

PAM or K-Medoids: Example (1)

- Data objects

	X	Y
0	8	7
1	3	7
2	4	9
3	9	6
4	8	5
5	5	8
6	7	3
7	8	4
8	7	5
9	4	5



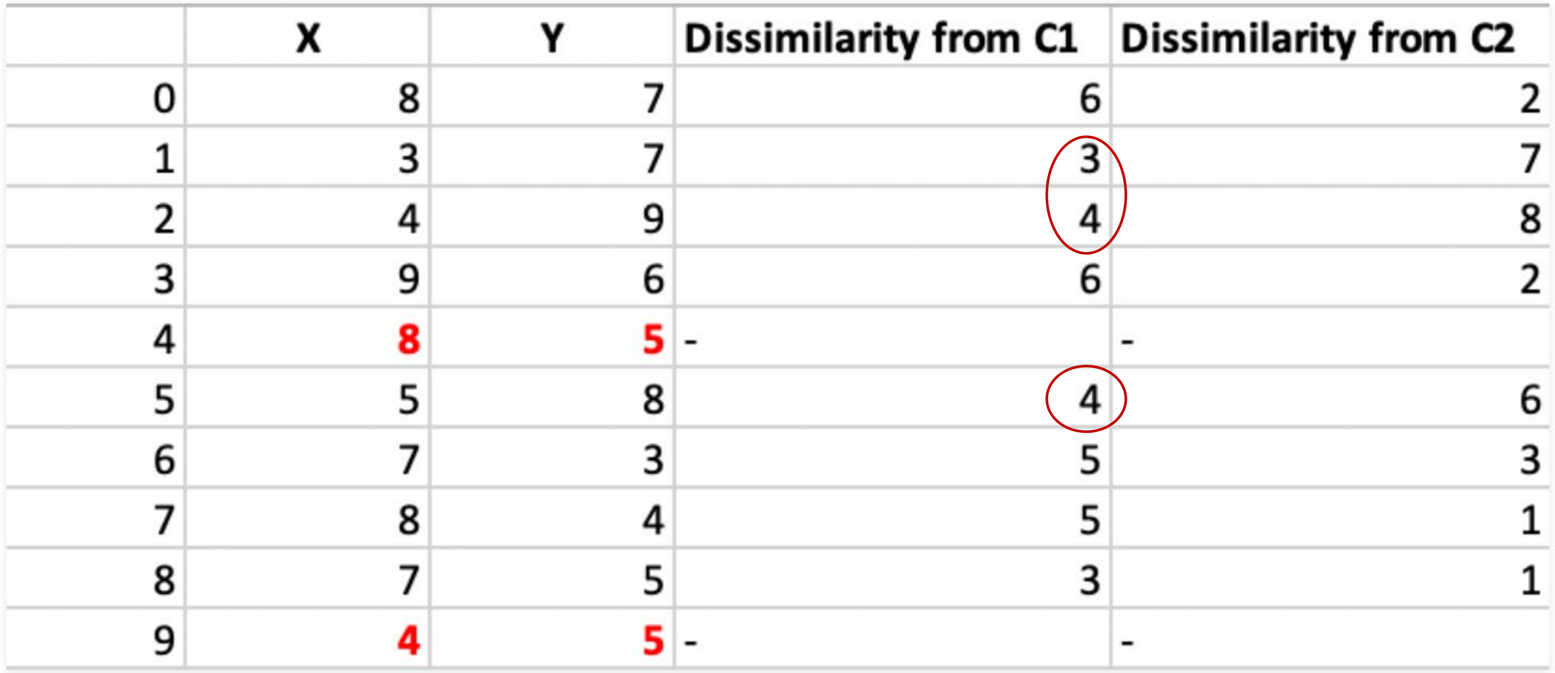
Goal: create two clusters

Choose randomly two medoids

C_1 = (4,5) and C_2 = (8,5)

PAM or K-Medoids: Example (1)

- Calculating cost through Manhattan distance.
  - The dissimilarity of each non-medoid point with the medoids is calculated Each point is assigned to the cluster of that medoid whose dissimilarity is less.
  - The points 1, 2, 5 go to cluster C1 and 0, 3, 6, 7, 8 go to cluster C2.
  - The Cost = (3 + 4 + 4) + (3 + 1 + 1 + 2 + 2) = 20



		X	Y	Dissimilarity from C1	Dissimilarity from C2
	0	8	7	6	2
	1	3	7	3	7
	2	4	9	4	8
	3	9	6	6	2
C_2	4	**8**	**5**	-	-
	5	5	8	4	6
	6	7	3	5	3
	7	8	4	5	1
	8	7	5	3	1
C_1	9	**4**	**5**	-	-

PAM or K-Medoids: Example (1)

- Randomly select one non-medoid point and recalculate the cost.
  - Let the randomly selected point be (8, 4).
  - The dissimilarity– C1 (4, 5) and C2 (8, 4) is calculated and tabulated.
  - The points 1, 2, 5 go to cluster C1 and 0, 3, 4, 6, 8 go to cluster C2.
  - The New cost = (3 + 4 + 4) + (2 + 2 + 1 + 3 + 3) = 22
  - Swap Cost = New Cost – Previous Cost = 22 – 20 and 2 >0
  - As the swap cost is not less than zero, we undo the swap.

		X	Y	Dissimilarity from C1	Dissimilarity from C2
	0	8	7	6	3
	1	3	7	3	8
	2	4	9	4	9
	3	9	6	6	3
	4	8	5	4	1
	5	5	8	4	7
	6	7	3	5	2
C_2	7	8	4	-	-
	8	7	5	3	2
C_1	9	4	5	-	-

PAM or K-Medoids: Example (1)

- Randomly select one non-medoid point and recalculate the cost.
  - Let the randomly selected point be (4, 9).
  - The dissimilarity– C1 (4, 9) and C2 (8, 5) is calculated and tabulated.

		x	Y	Dissimilarity from C1	Dissimilarity from C2
	0	8	7	6	2
	1	3	7	3	7
C_1	2	4	9	-	-
	3	9	6	8	2
C_2	4	8	5	-	-
	5	5	8	2	6
	6	7	3	9	3
	7	8	4	9	1
	8	7	5	7	1
	9	4	5	4	4

PAM or K-Medoids: Example (1)

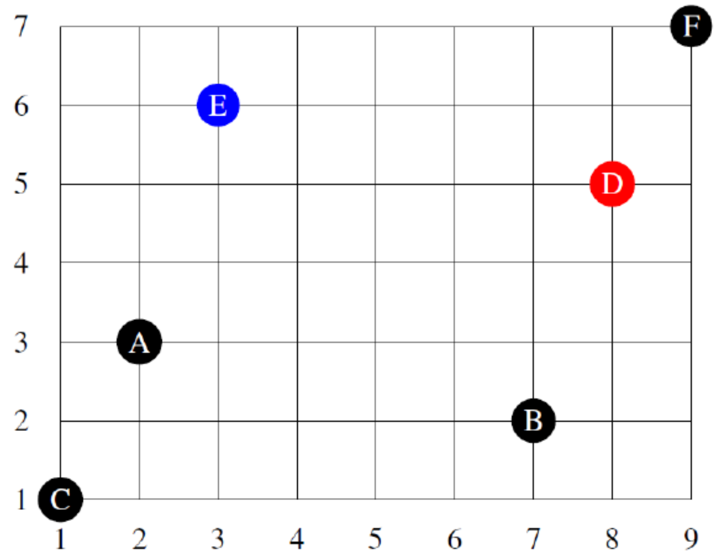
- Randomly select one non-medoid point and recalculate the cost.
  - Let the randomly selected point be (4, 9).
  - The dissimilarity– C1 (4, 9) and C2 (8, 5) is calculated and tabulated.
  - The points 1,5,9 go to cluster C1 and 0,3,6,7,8 go to cluster C2.
  - The New cost = (3 + 2 + 4) + (2 + 2 + 3 + 1 + 1) = 18
  - Swap Cost = New Cost – Previous Cost = 18 – 20 and -2 <0
  - As the swap cost is less than zero, we do the swap. Hence C1(4, 9) and C2(8, 5) are the medoids. We repeat until no change in the medoids.

		x	Y	Dissimilarity from C1	Dissimilarity from C2
	0	8	7	6	2
	1	3	7	3	7
C_1	2	4	9	-	-
	3	9	6	8	2
C_2	4	8	5	-	-
	5	5	8	2	6
	6	7	3	9	3
	7	8	4	9	1
	8	7	5	7	1
	9	4	5	4	4

PAM or K-Medoids: Example (2)

Consider the following 2-dimensional data set:

	A	B	C	D	E	F
x_1	2	7	1	8	3	9
x_2	3	2	1	5	6	7



Perform the first loop of the PAM algorithm (k = 2) using the Manhattan distance. Select D and E (highlighted in the plot) as initial medoids and compute the resulting medoids and clusters.

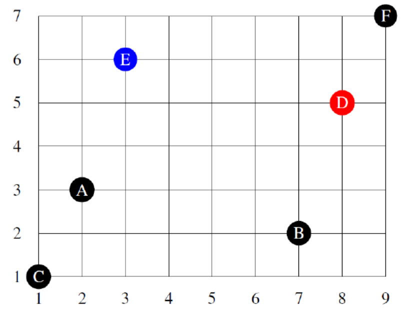
Hint: When $C(m)$ denotes the cluster of medoid m, and M denotes the set of medoids, then the total distance TD may be computed as

$$TD = \sum_{m \in M} \sum_{o \in C(m)} d(m, o)$$

PAM or K-Medoids: Example (2)

Consider the following 2-dimensional data set:

	A	B	C	D	E	F
x_1	2	7	1	8	3	9
x_2	3	2	1	5	6	7



Perform the first loop of the PAM algorithm (k = 2) using the Manhattan distance. Select D and E (highlighted in the plot) as initial medoids and compute the resulting medoids and clusters.

Hint: When $C(m)$ denotes the cluster of medoid m, and M denotes the set of medoids, then the total distance TD may be computed as

$$TD = \sum_{m \in M} \sum_{o \in C(m)} d(m, o)$$

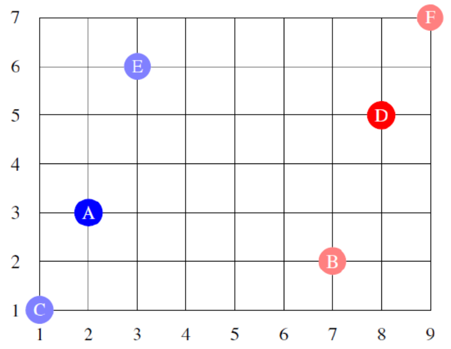
We have the following distance values (values which are clear by symmetry and reflexivity are left out):

	B	C	D	E	F
A	6	3	8	4	11
B		7	4	8	7
C			11	7	14
D				6	3
E					7

The following table shows assignments and $TD_{m \leftrightarrow n}$ value for each pair $(m, n) \in M \times N$ with $M = \{D, E\}$ and $N = \{A, B, C, F\}$.

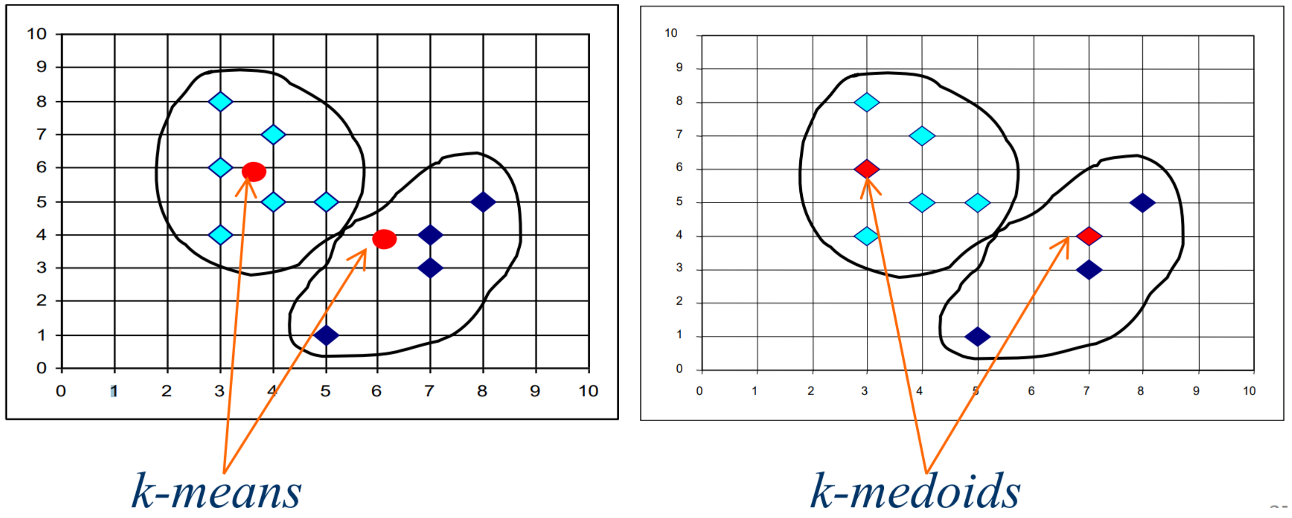
Medoids		Assignment						TD
m_1	m_2	A	B	C	D	E	F	
D	E	1	0	1	0	1	0	**18**
D	A	1	0	1	0	1	0	**14**
D	B	1	1	1	0	0	0	22
D	C	1	0	1	0	0	0	16
D	F	0	0	0	0	0	1	29
E	A	1	1	1	0	0	0	22
E	B	0	1	0	1	0	0	22
E	C	1	1	1	0	0	0	23
E	F	0	1	0	1	0	1	21

The table shows that swapping E and A yields the largest improvement in terms of TD. The updated clustering after the first iteration is shown in the following figure.



K-Medoids Clustering Method

- Difference between K-means and K-medoids
  - K-means: Computer cluster centers (may not be the original data point)
  - K-medoids: Each cluster’s centroid is represented by a point in the cluster
  - K-medoids is more robust than K-means in the presence of outliers because a medoid is less influenced by outliers or other extreme values



PAM

- Starts from an initial set of medoids and iteratively replaces one of the medoids by one of the non-medoids if it improves the total distance of the resulting clustering
- PAM is more robust than k-means in the presence of noise and outliers because a medoid is less influenced by outliers or other extreme values than a mean.
- PAM works efficiently for small data sets but does not scale well for large data sets.

→ Sampling based method, CLARA (Clustering LARge Applications)

Outline

- Introduction of clusters
- Introduction of clustering
- K-means
- K-medoids
    - PAM&CLARA

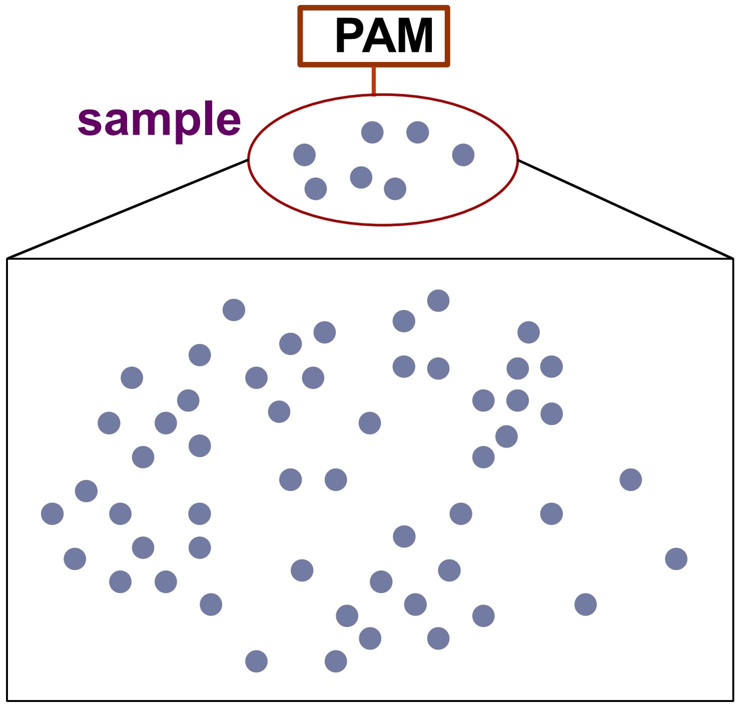
CLARA (Clustering Large Applications)

- CLARA (Kaufmann and Rousseeuw in 1990)
- It draws multiple samples of the data set, applies PAM on each sample, and gives the best clustering as the output.
- Strength: deals with larger data sets than PAM.
- Weakness:
  - Efficiency depends on the sample size.
  - A good clustering based on samples will not necessarily represent a good clustering of the whole data set if the sample is biased.

CLARA (Clustering Large Applications)

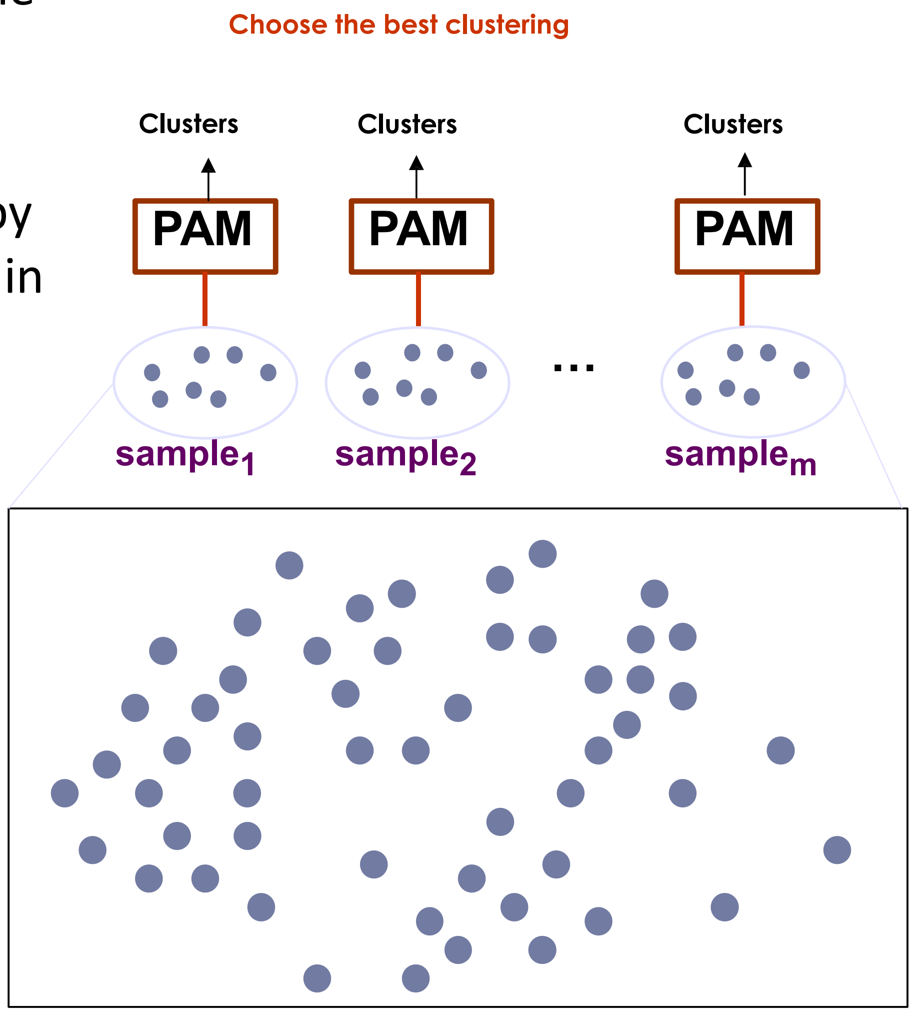
- *CLARA draws a sample of* the dataset and applies PAM on the sample in order to find the medoids.
- If the sample is best representative of the entire dataset then the metroids of the sample should approximate the medoids of the entire dataset.
- Medoids are chosen from the sample.
    - Note that the algorithm cannot find the best solution if one of the best k-medoids is not among the selected sample.

CLARA (Clustering Large Applications)



CLARA (Clustering Large Applications)

- To improve the approximation, multiple samples are drawn and the best clustering is returned as the output.
- The clustering accuracy is measured by the average dissimilarity of all objects in the entire dataset.



CLARA (Clustering Large Applications)

- For i= 1 to m, repeat the following steps
  - Draw a sample of N objects randomly from the entire data set, and call A lgorithm PAM to find k medoids of the sample.
  - For each object in the entire data set, determine which of the k medoids is the most similar to it.
  - Calculate the average dissimilarity ON THE ENTIRE DATASET of the cluster ing obtained in the previous step. If this value is less than the current min imum, use this value as the current minimum, and retain the k medoids f ound in Step (1) as the best set of medoids obtained so far.