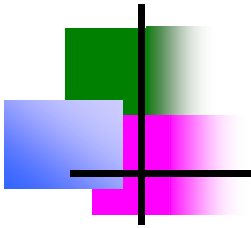
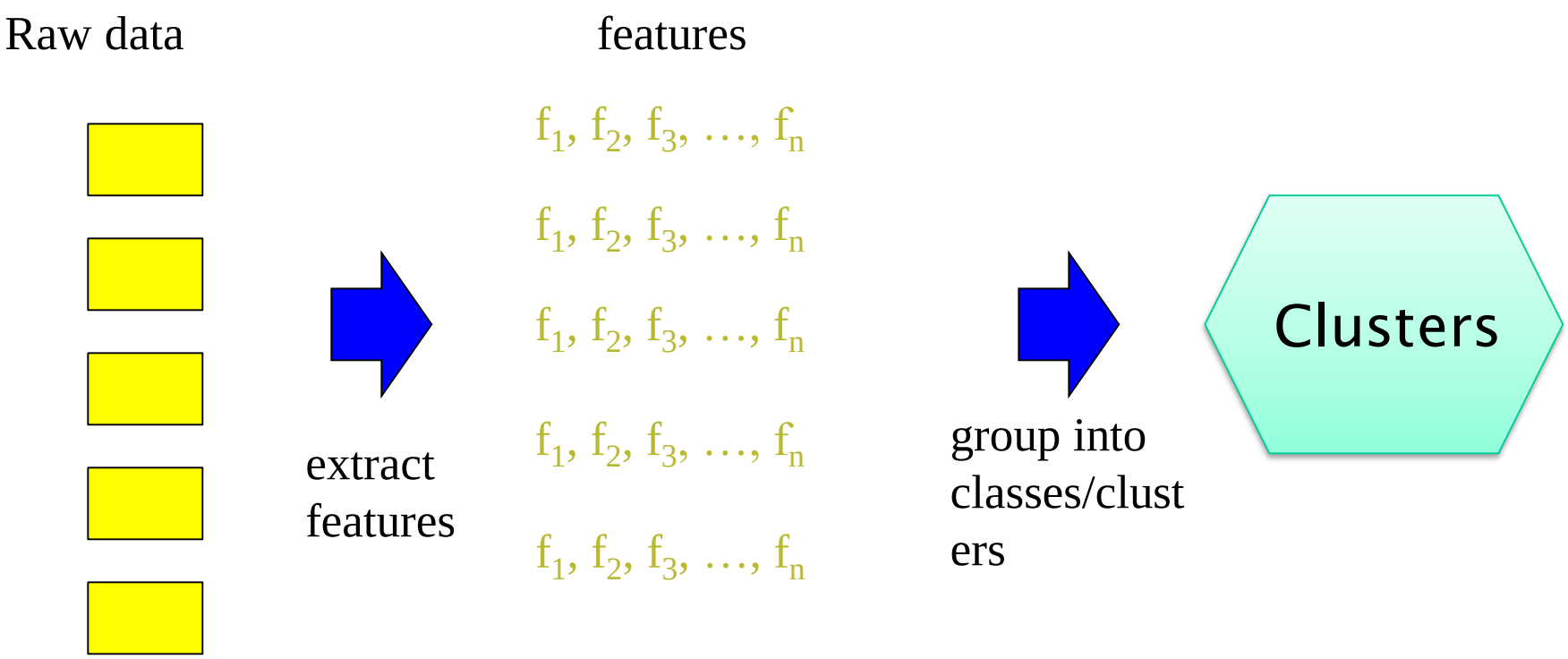


Machine Learning and Computational Intelligence
Lecture 13

Sanjeeb Prasad Panday, PhD
Associate Professor
Dept. of Electronics and Computer Engineering
Director (ICTC)
IOE, TU

Unsupervised learning: clustering

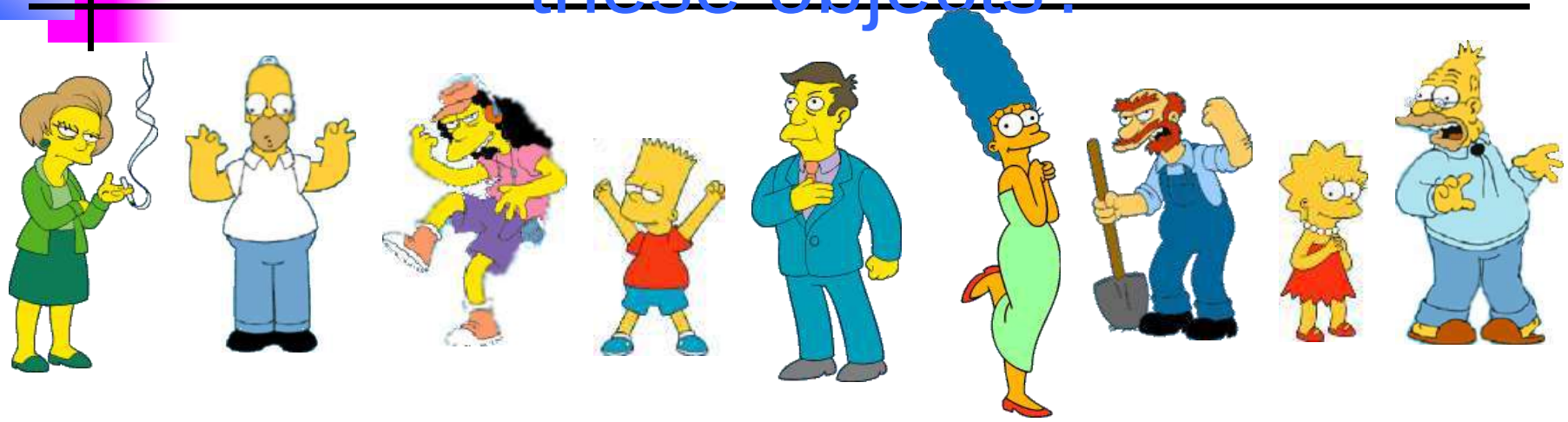


No "supervision", we're only given data and want to find natural groupings

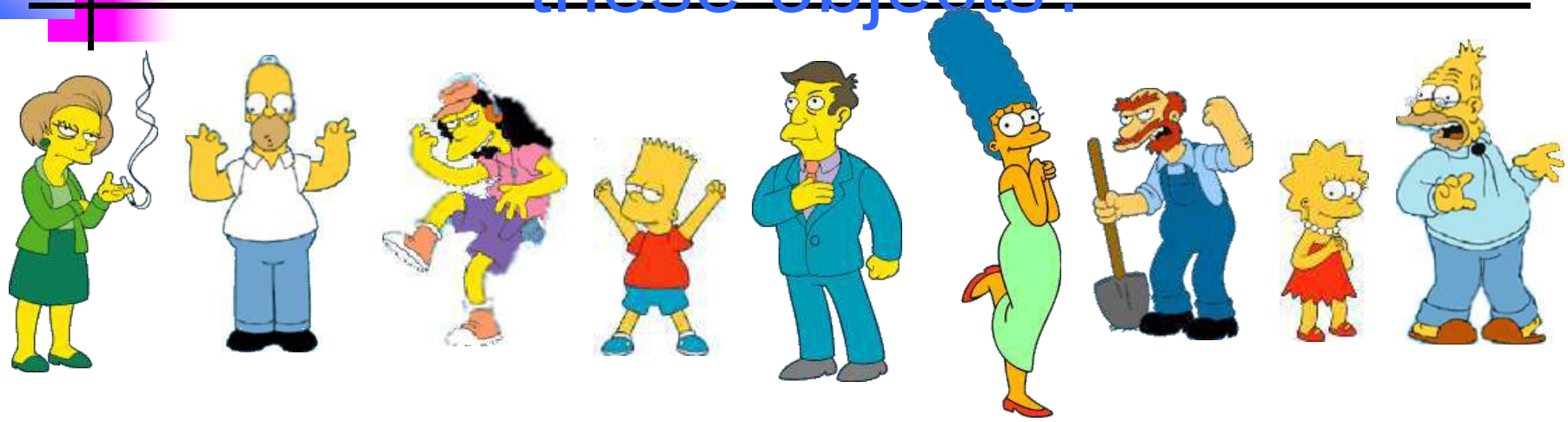
What is Clustering?

- **Clustering** is the classification of objects into different groups, or more precisely, the partitioning of a data set into subsets (clusters), so that the data in each subset (ideally) share some common trait - often according to some defined distance measure.
- Clustering is a technique for finding similarity groups in data, called **clusters**. I.e.,
  - it groups data instances that are similar to (near) each other in one cluster and data instances that are very different (far away) from each other into different clusters.

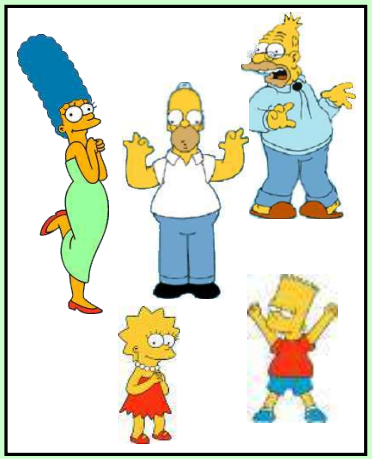
What is a natural grouping among these objects?



What is a natural grouping among these objects?

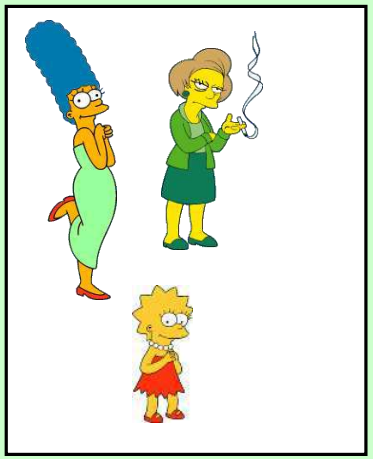


Clustering is subjective



Simpson's Family

School Employees



Females

Males

What is Similarity?

The quality or state of being similar; likeness; resemblance; as, a similarity of features.

Webster's Dictionary

Similarity is hard to define, but…
"*We know it when we see it*"

The real meaning of similarity is a philosophical question. We will take a more pragmatic approach.

What Is a Good Clustering?

- A distance (similarity, or dissimilarity) function
- Clustering quality
  - Inter-clusters distance $\Rightarrow$ maximized
  - Intra-clusters distance $\Rightarrow$ minimized
- The quality of a clustering result depends on the algorithm, the distance function, and the application.
- Precise definition of clustering quality is difficult
  - Application-dependent
  - Ultimately subjective

Defining Distance Measures

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

Peter Piotr

0.23

3

342.7

Common Distance measures:

- *Distance measure* will determine how the *similarity* of two elements is calculated and it will influence the shape of the clusters.

  They include:

1. The Euclidean distance (also called 2-norm distance) is given by:

$$d(\mathbf{x}, \mathbf{y}) = \sqrt{\sum_{k=1}^{n} (x_k - y_k)^2}$$

2. The Manhattan distance (also called taxicab norm or 1-norm) is given by:

$$d(x, y) = \sqrt[2]{\sum_{i=1}^{p} |x_i - y_i|^2}$$

Euclidean Distance

- Euclidean Distance

$$d(\mathbf{x}, \mathbf{y}) = \sqrt{\sum_{k=1}^{n} (x_k - y_k)^2}$$

where n is the number of dimensions (attributes) and x_k and y_k are, respectively, the k^{th} attributes (components) or data objects $\mathbf{x}$ and $\mathbf{y}$.

- Standardization is necessary, if scales differ.

Distance based on Numeric Data

- **Minkowski distance: A popular distance measure**

$$d(i, j) = \sqrt[h]{|x_{i1} - x_{j1}|^h + |x_{i2} - x_{j2}|^h + \cdots + |x_{ip} - x_{jp}|^h}$$

  - where $i = (x_{i1}, x_{i2}, \ldots, x_{ip})$ and $j = (x_{j1}, x_{j2}, \ldots, x_{jp})$ are two p-dimensional data objects, and h is the order (the distance so defined is also called L-*h* norm)

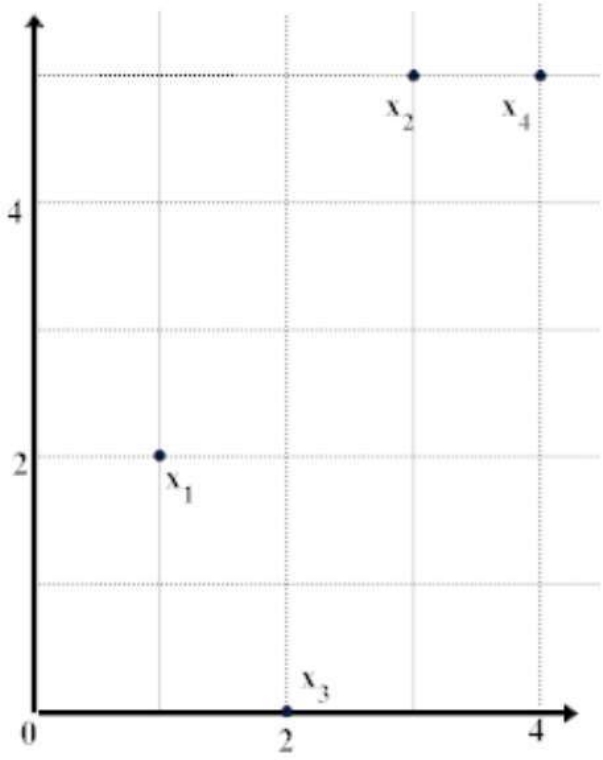
- **Note that Euclidean and Manhattan distances are special cases**
  - $h = 1$: (L_1 norm) Manhattan distance

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

  - $h = 2$: (L_2 norm) Euclidean distance

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

Distance based on Numeric Data



Data Matrix

point	attribute1	attribute2
x1	1	2
x2	3	5
x3	2	0
x4	4	5

Distance Matrix (Manhattan)

	x1	*x2*	*x3*	*x4*
x1	0			
x2	5	0		
x3	3	6	0	
x4	6	1	7	0

Distance Matrix (Euclidean)

	x1	*x2*	*x3*	*x4*
x1	0			
x2	3.61	0		
x3	2.24	5.1	0	
x4	4.24	1	5.39	0

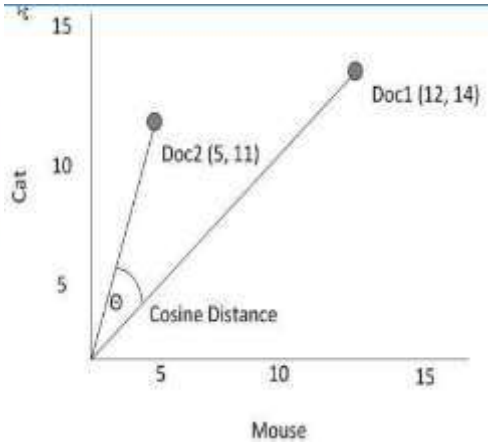
Manhattan distance

Taxicab Distance = $|X_A - X_B| + |Y_A - Y_B|$

Vector Based Similarity measures

- **In some situations, distance measures provide a skewed view of data**
  - E.g., when the data is very sparse and 0's in the vectors are not significant
  - In such cases, typically vector-based similarity measures are used
  - Most common measure: Cosine similarity

$$X = \langle x_1, x_2, \cdots, x_n \rangle \qquad Y = \langle y_1, y_2, \cdots, y_n \rangle$$



  - Dot product of two vectors:

$$sim(X,Y) = X \bullet Y = \sum_i x_i \times y_i$$

  - Cosine Similarity = normalized dot product
  - the norm of a vector X is:

$$\|X\| = \sqrt{\sum_i x_i^2}$$

  - the cosine similarity is:

$$sim(X,Y) = \frac{X \bullet Y}{\|X\| \times \|y\|} = \frac{\sum_i (x_i \times y_i)}{\sqrt{\sum_i x_i^2} \times \sqrt{\sum_i y_i^2}}$$

Application Information Retrieval

- **Documents are represented as "bags of words"**
- **Represented as vectors when used computationally**
  - A vector is an array of floating point (or binary in case of bit maps)
  - Has direction and magnitude
  - Each vector has a place for every term in collection (most are sparse)

Document Ids

	nova	galaxy	heat	actor	film	role
A	1.0	0.5	0.3			
B	0.5	1.0				
C		1.0	0.8	0.7		
D		0.9	1.0	0.5		
E				1.0		1.0
F					0.7	
G	0.5		0.7			0.9
H		0.6		1.0	0.3	0.2
I			0.7	0.5		0.3

a document vector

$$D_i = w_{d_{i1}}, w_{d_{i2}}, \ldots, w_{d_{it}}$$

$$Q = w_{q1}, w_{q2}, \ldots, w_{qt}$$

$w = 0$ if a term is absent

Common Distance measures:

3.The maximum norm is given by:

$$d(x, y) = \max_{1 \le i \le p} |x_i - y_i|$$

4. The Mahalanobis distance corrects data for different scales and correlations in the variables.

5. Inner product space: The angle between two vectors can be used as a distance measure when clustering high dimensional data

6. Hamming distance (sometimes edit distance) measures the minimum number of substitutions required to change one member into another.

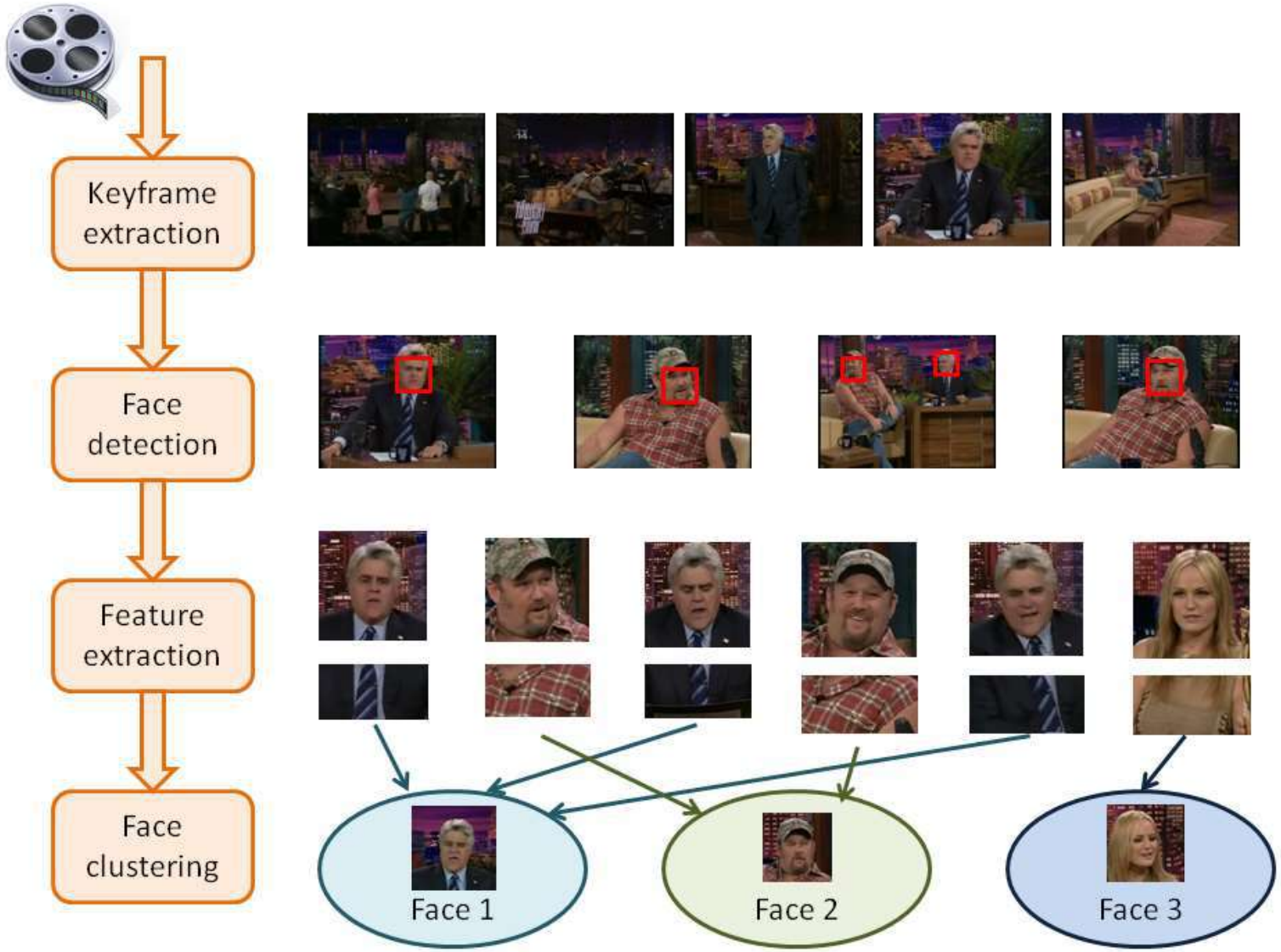
What is clustering for?

- Let us see some real-life examples
- Example 1: groups people of similar sizes together to make “small”, “medium” and “large” T-Shirts.
  - Tailor-made for each person: too expensive
  - One-size-fits-all: does not fit all.
- Example 2: In marketing, segment customers according to their similarities
  - To do targeted marketing.

What is clustering for? (cont…)

- Example 3: Given a collection of text documents, we want to organize them according to their content similarities,
  - To produce a topic hierarchy
- In fact, clustering is one of the most utilized data mining techniques.
  - It has a long history, and used in almost every field, e.g., medicine, psychology, botany, sociology, biology, archeology, marketing, insurance, libraries, etc.
  - In recent years, due to the rapid increase of online documents, text clustering becomes important.

Face Clustering



Face clustering

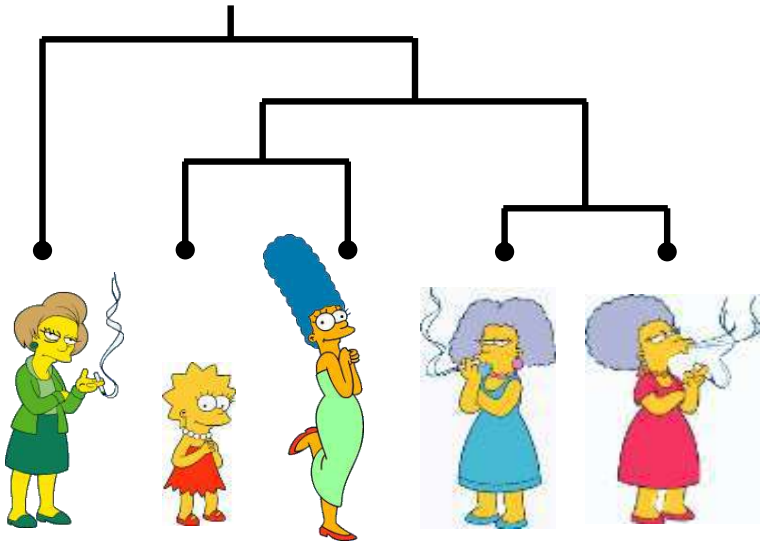
Examples of Clustering Applications

- Marketing: Help marketers discover distinct groups in their customer bases, and then use this knowledge to develop targeted marketing programs
- Land use: Identification of areas of similar land use in an earth observation database
- Insurance: Identifying groups of motor insurance policy holders with a high average claim cost
- Urban planning: Identifying groups of houses according to their house type, value, and geographical location
- Seismology: Observed earth quake epicenters should be clustered along continent faults

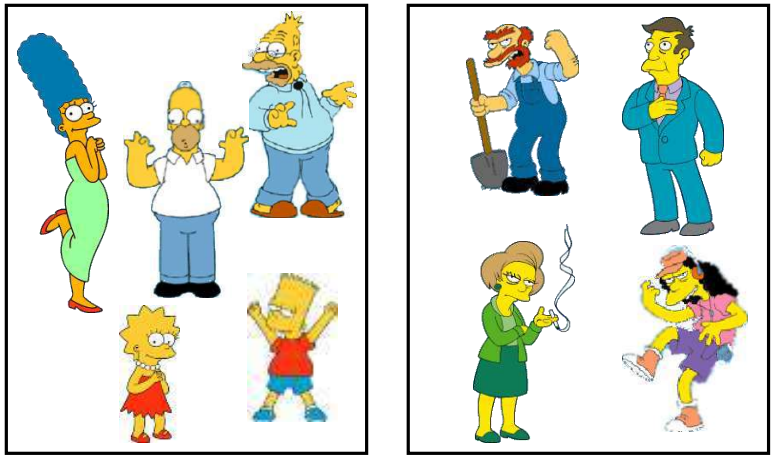
Two Types of Clustering

- **Partitional algorithms:** Construct various partitions and then evaluate them by some criterion
- **Hierarchical algorithms:** Create a hierarchical decomposition of the set of objects using some criterion

Hierarchical



Partitional

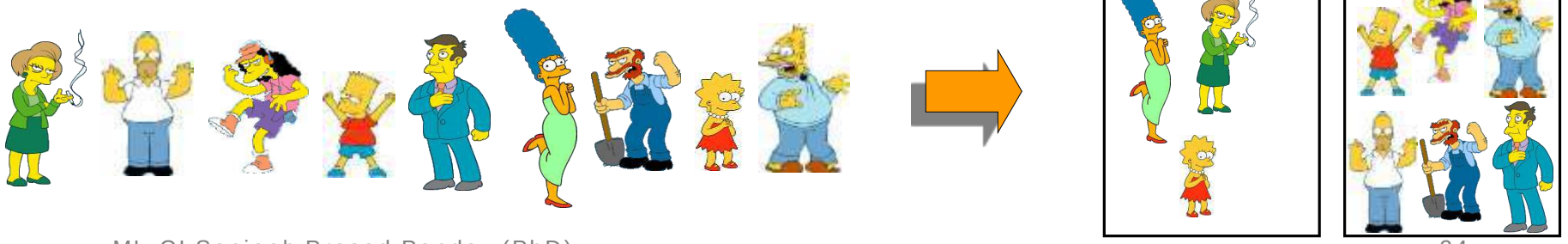


Hard vs. soft clustering

- Hard clustering: Each document belongs to exactly one cluster
  - More common and easier to do
- Soft clustering: A document can belong to more than one cluster.
  - Makes more sense for applications like creating browsable hierarchies
  - You may want to put a pair of sneakers in two clusters: (i) sports apparel and (ii) shoes
  - You can only do that with a soft clustering approach.

Partitional Clustering

- Nonhierarchical, each instance is placed in exactly one of K nonoverlapping clusters.
- Since only one set of clusters is output, the user normally has to input the desired number of clusters K.



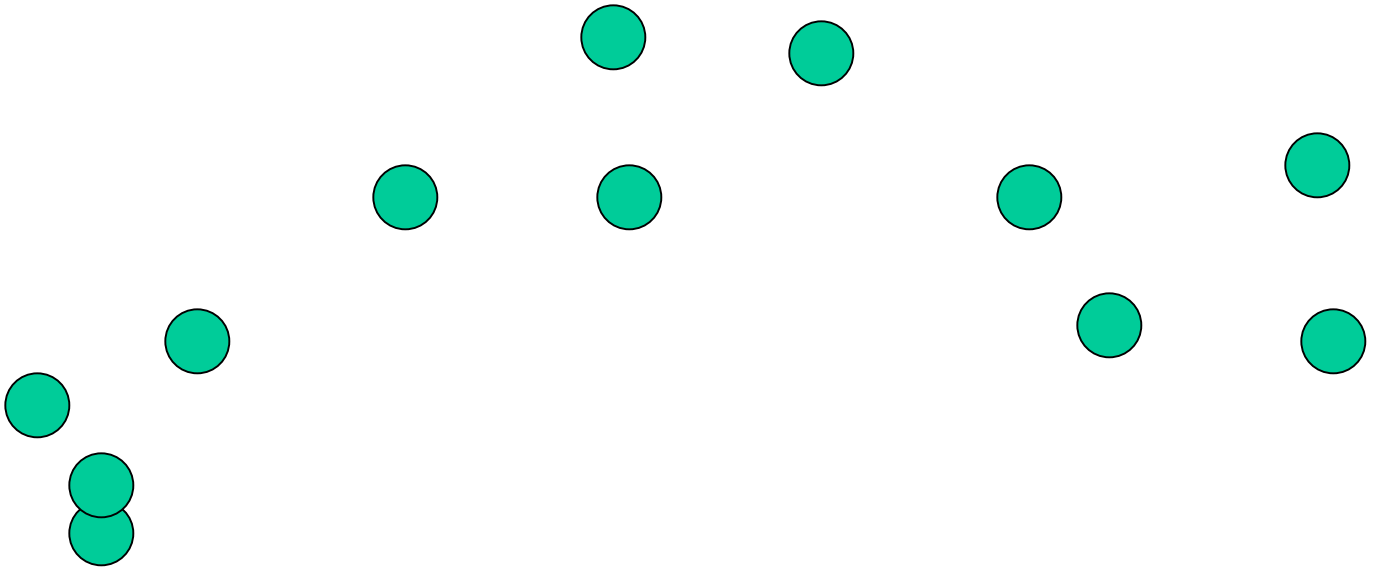
K-means

- Most well-known and popular clustering algorithm:
- The **k-means algorithm** is an algorithm to cluster *n* objects based on attributes into *k* partitions, where $k < n$.
- Start with some initial cluster centers

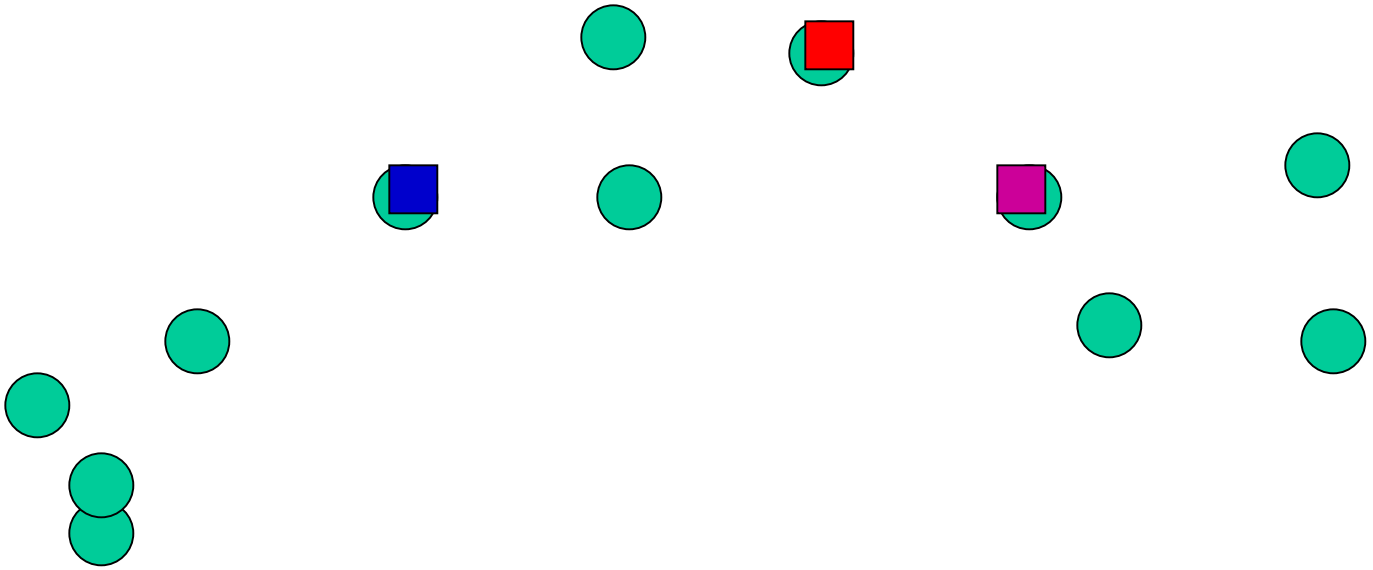
Iterate:

- Assign/cluster each example to closest center
- Recalculate centers as the mean of the points in a cluster

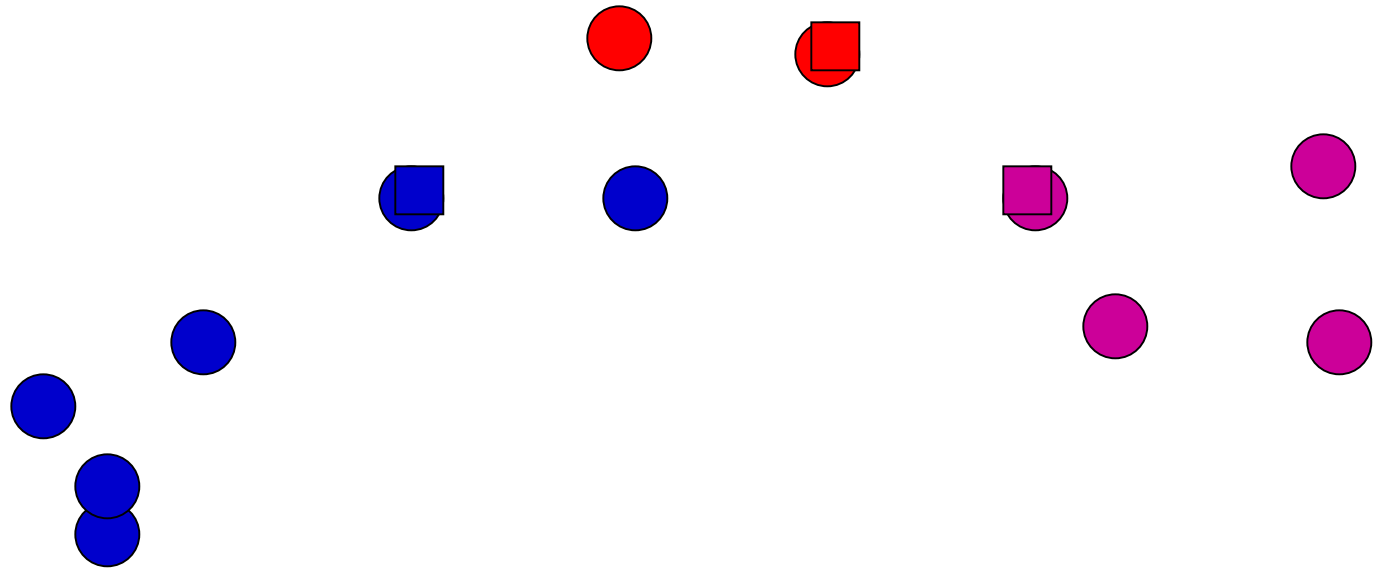
K-means: an example



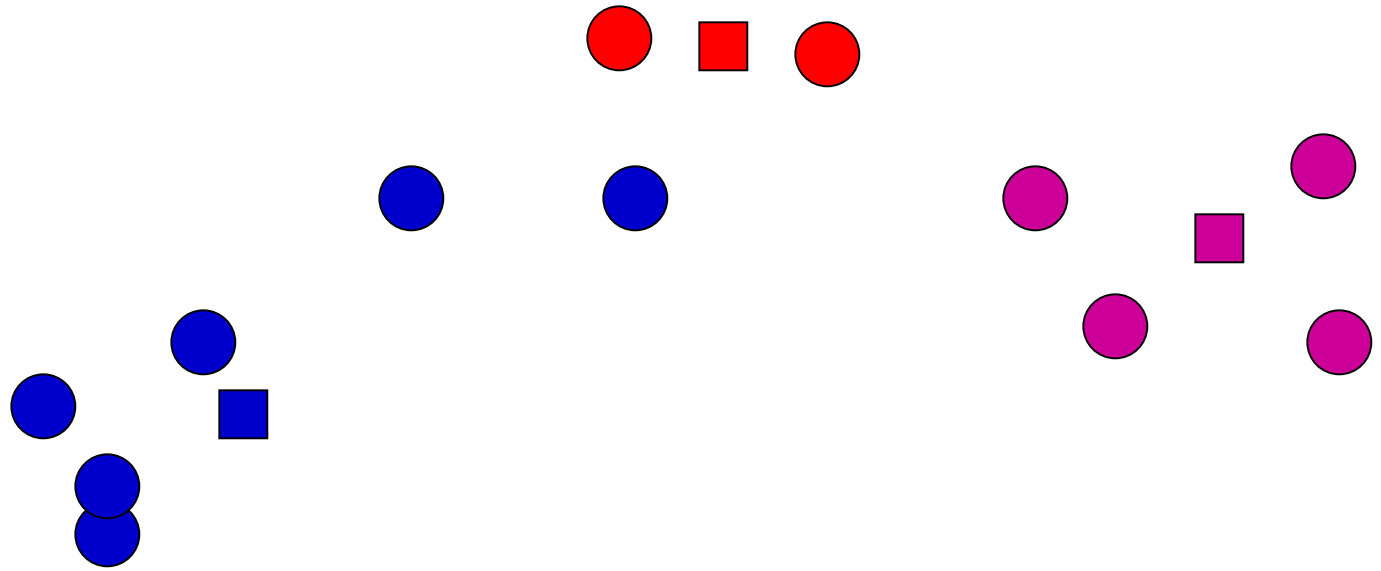
K-means: Initialize centers randomly



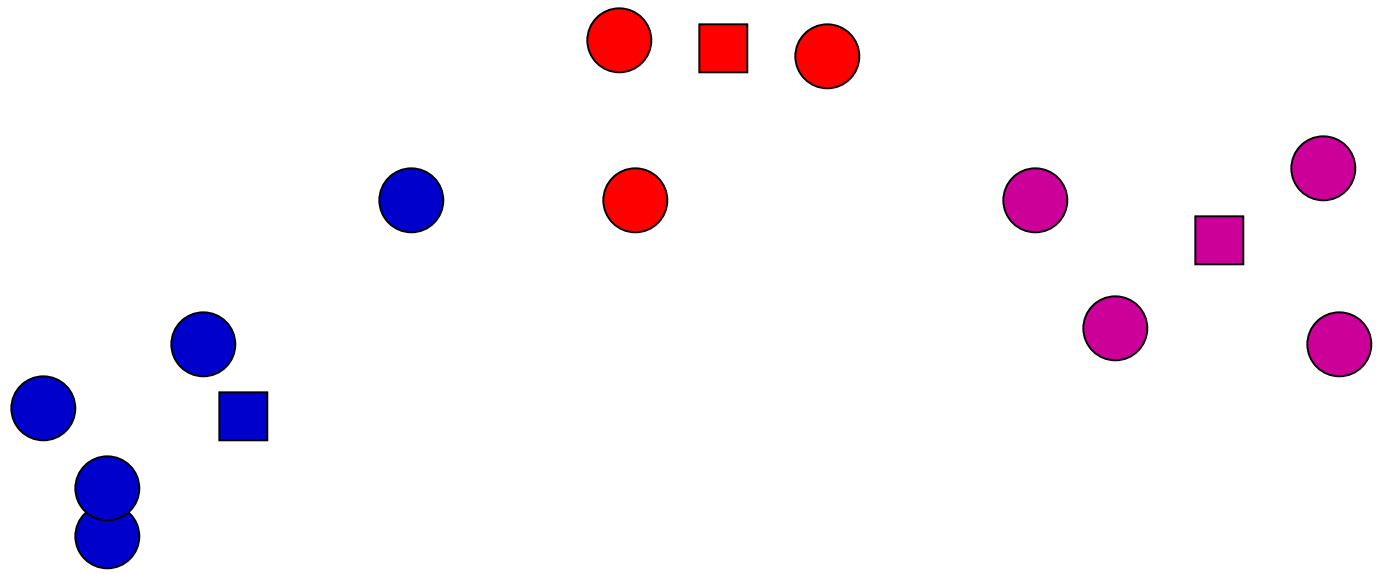
K-means: assign points to nearest center



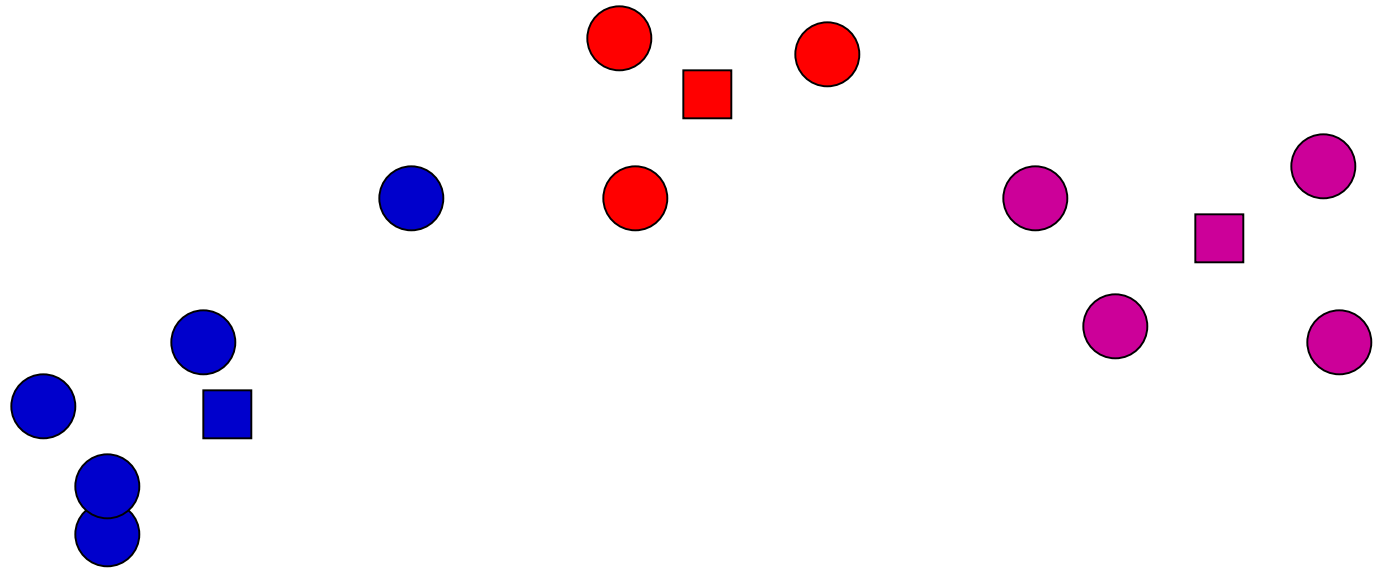
K-means: readjust centers



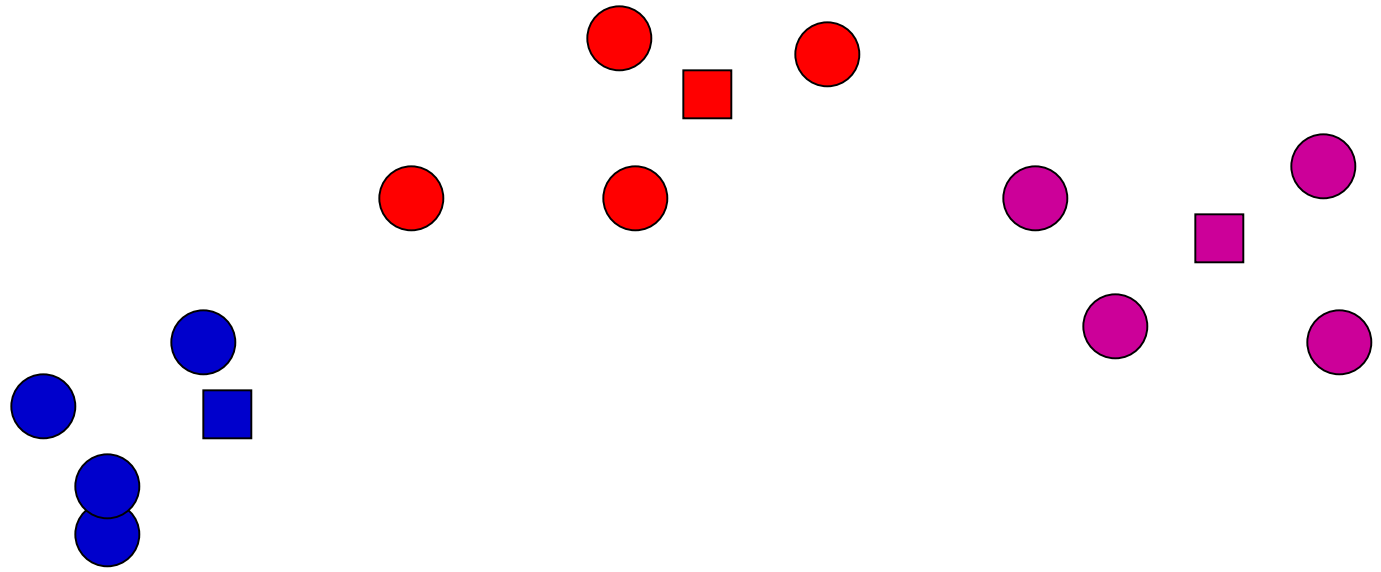
K-means: assign points to nearest center



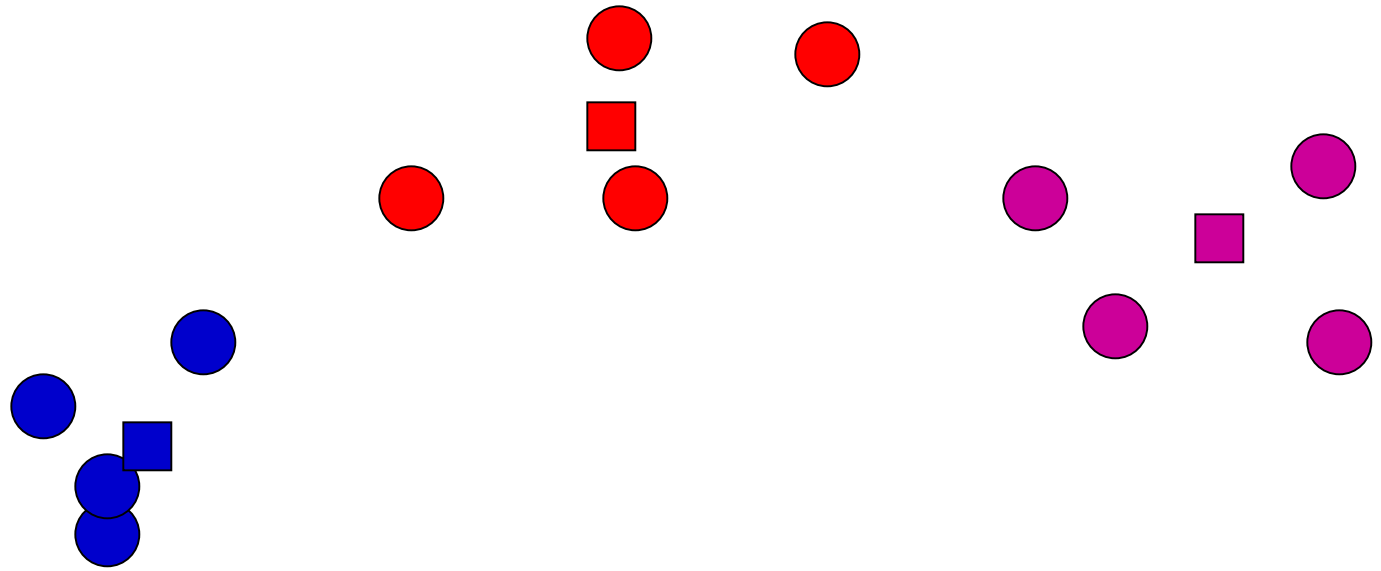
K-means: readjust centers



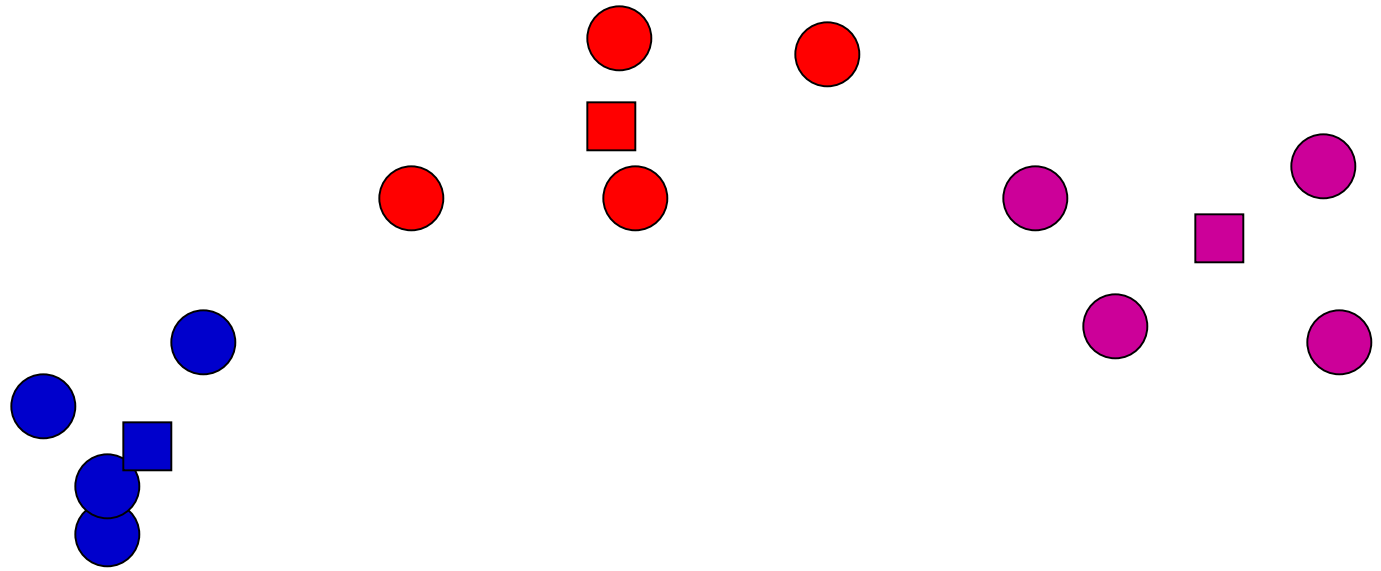
K-means: assign points to nearest center



K-means: readjust centers



K-means: assign points to nearest center



No changes: Done

Algorithm *k-means*

- 1. Decide on a value for *k*.
- 2. Initialize the *k* cluster centers (randomly, if necessary).
- 3. Decide the class memberships of the *N* objects by assigning them to the nearest cluster center.
- 4. Re-estimate the *k* cluster centers i.e. Recalculate centers as the mean of the points in a cluster
- 5. If none of the *N* objects changed membership in the last iteration, exit. Otherwise goto 3.

Evaluating Cluster assignment

K-means tries to minimize what is called the “k-means” loss function:

that is, the sum of the squared distances from each point to the associated cluster center

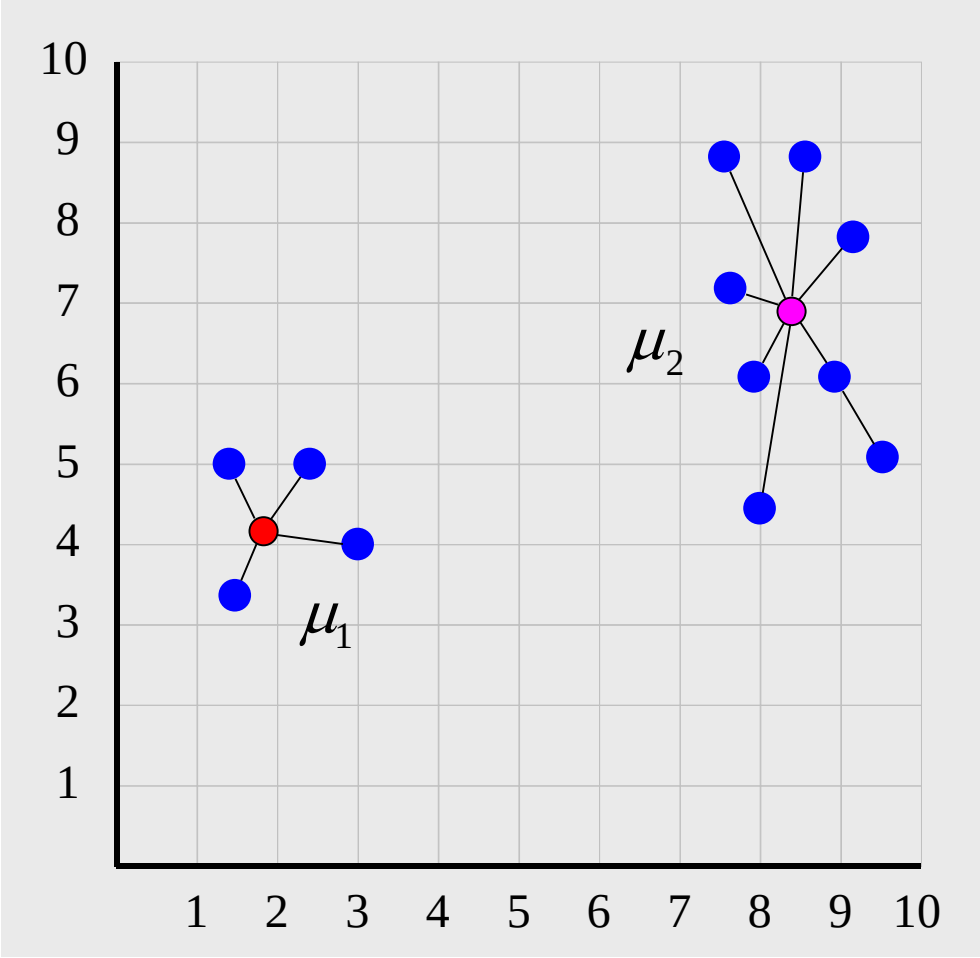
$$loss = \sum_{i=1}^{n} d(x_i, \mu_k)^2$$

where μ_k is cluster center for x_i

Sum of squared error

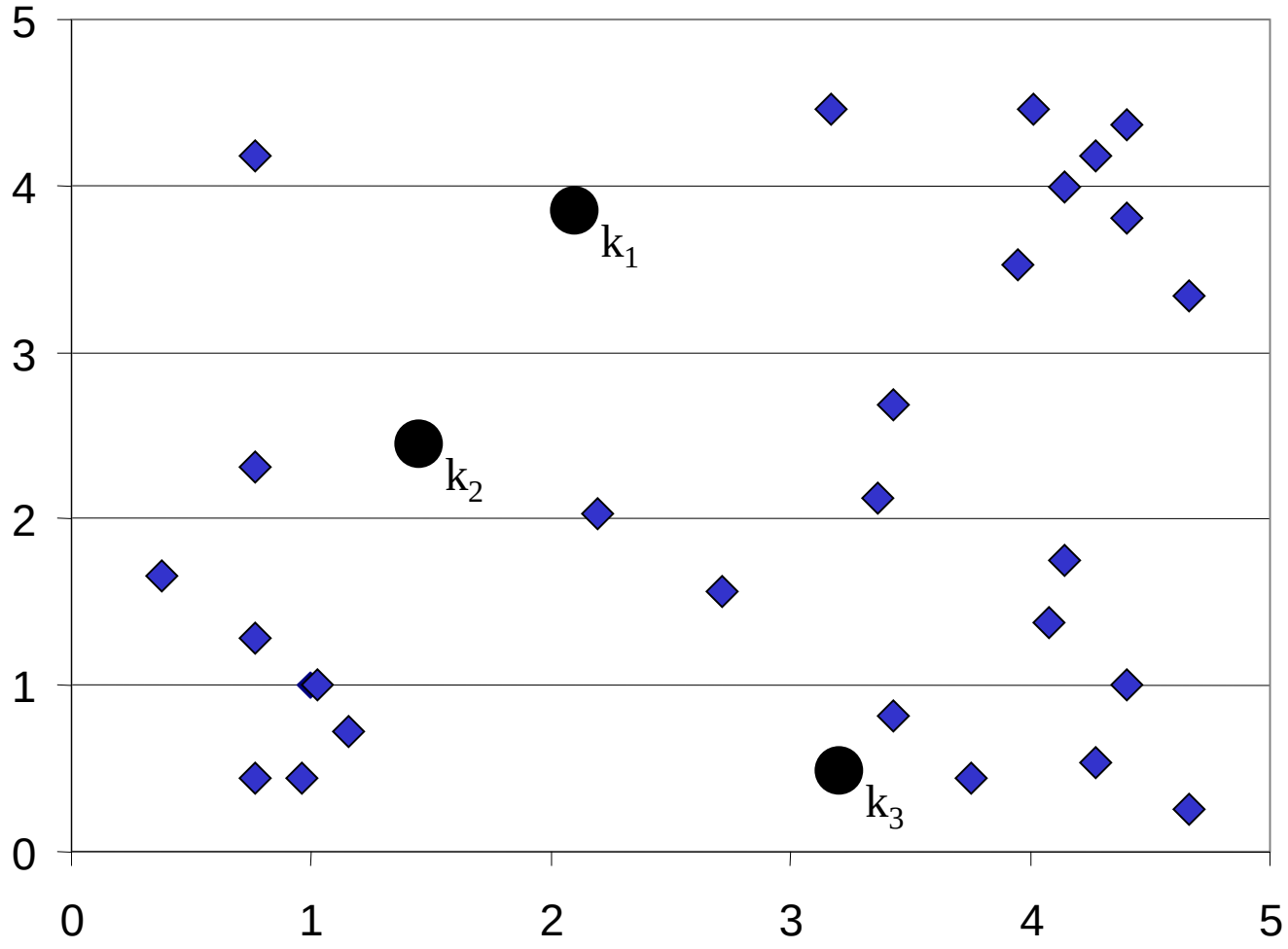
$$se = \sum_{i=1}^{K} \sum_{i=1}^{n} (x_{Ki} - \mu_k)^2$$

$$d(x,y) = \sqrt{\sum_{i=1}^{n} (x_i - y_i)^2}$$

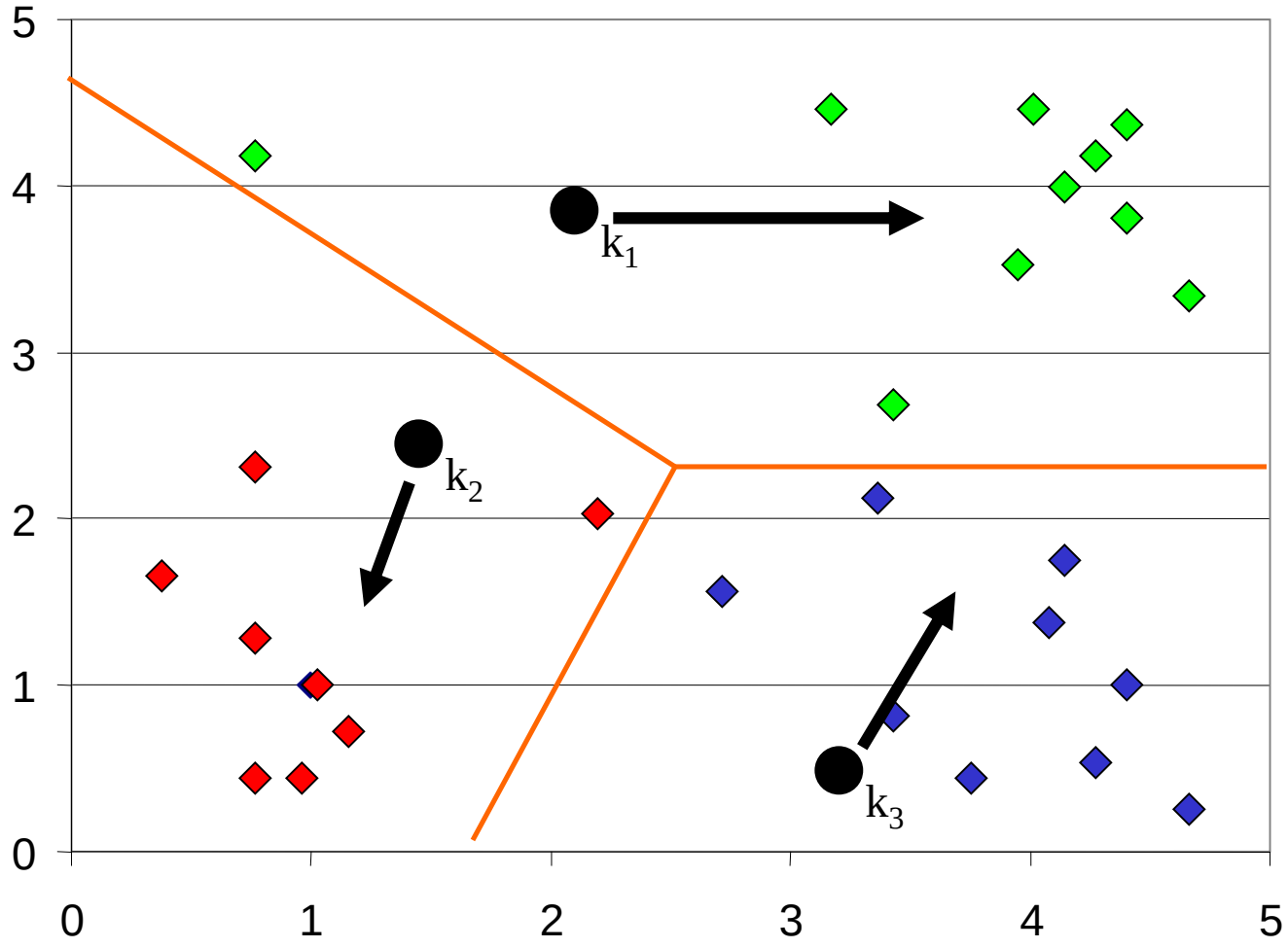


K-means Clustering: Step 1

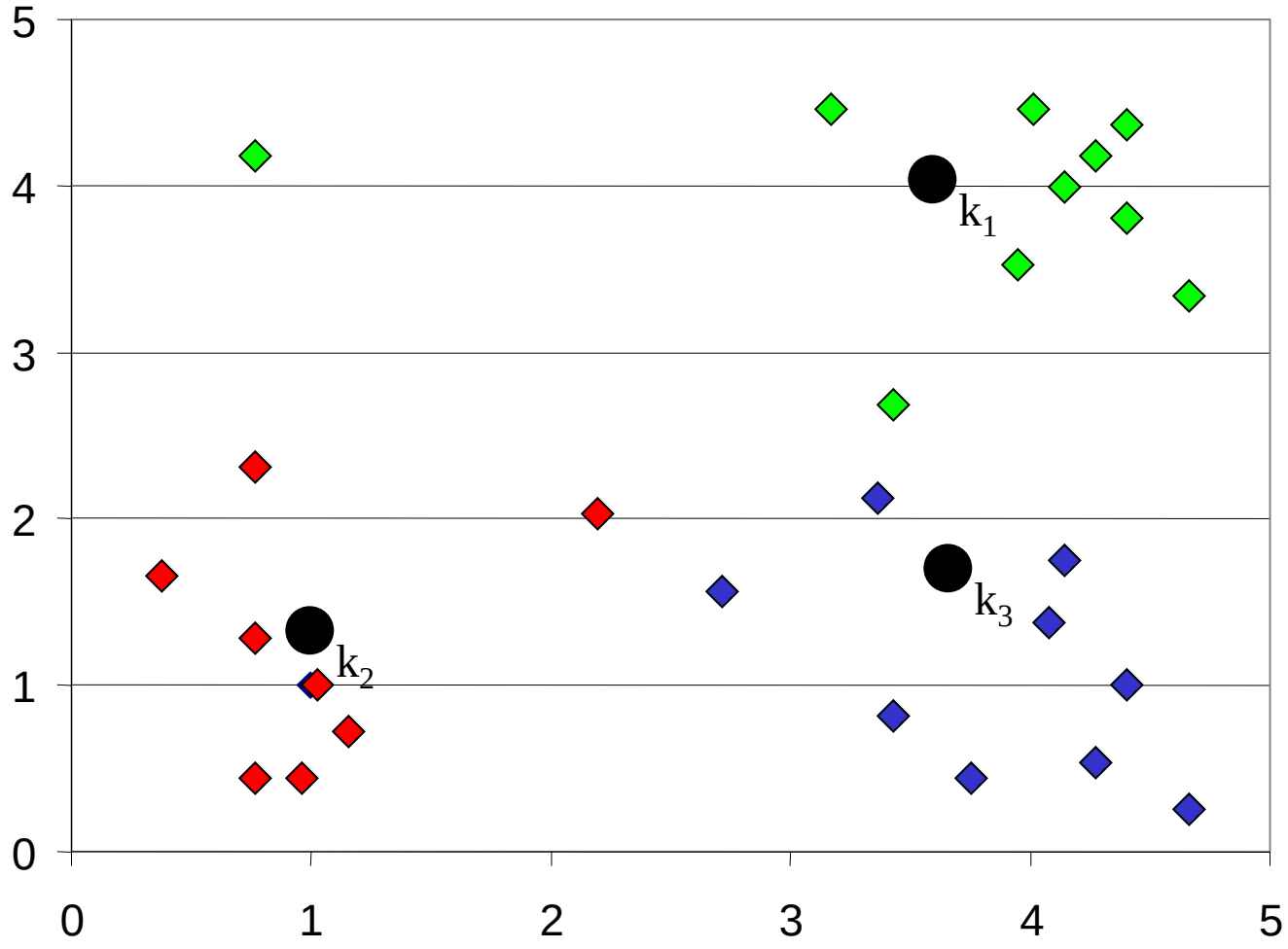
Algorithm: k-means, Distance Metric: Euclidean Distance



K-means Clustering: Step 2

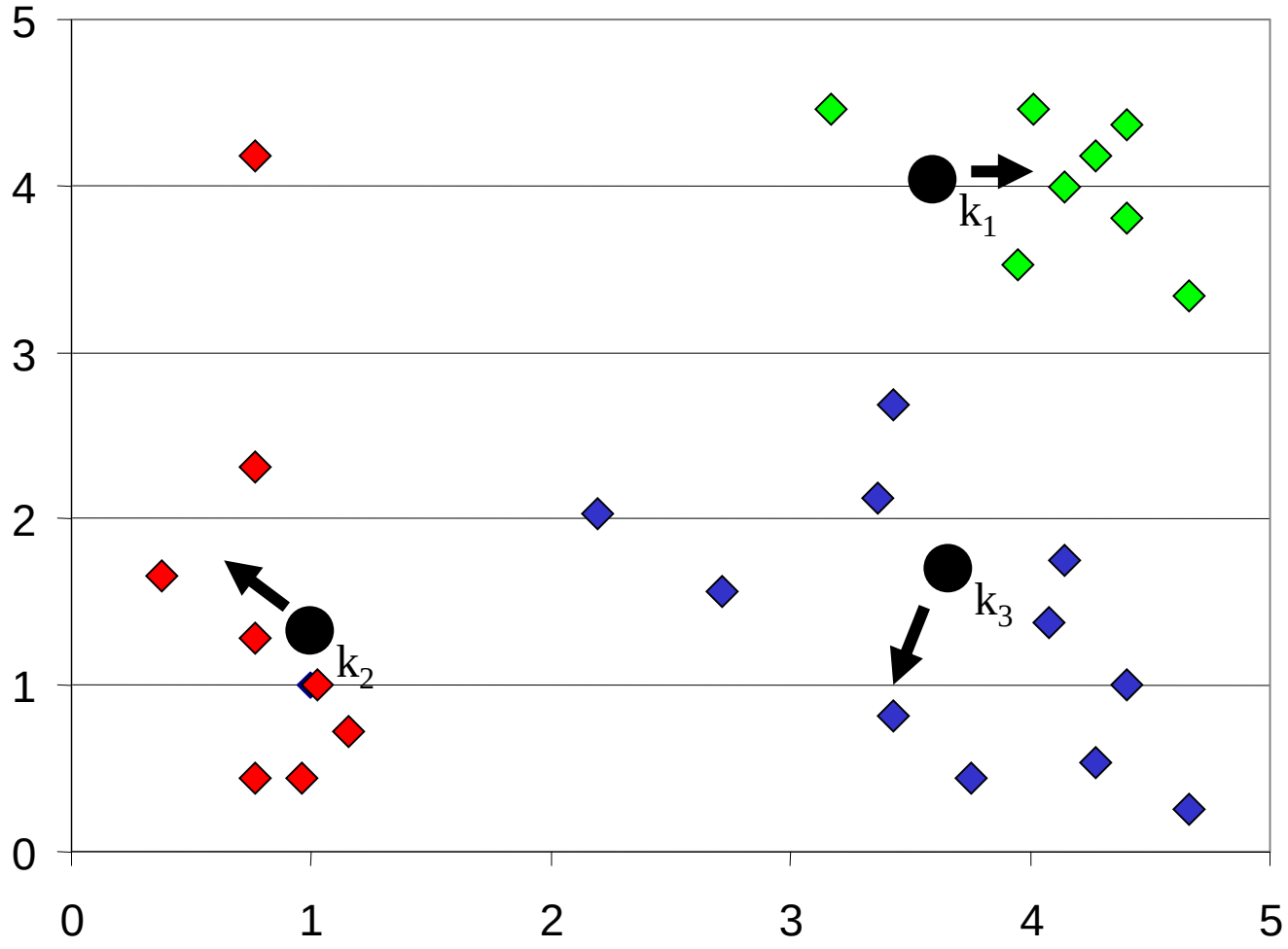


K-means Clustering: Step 3

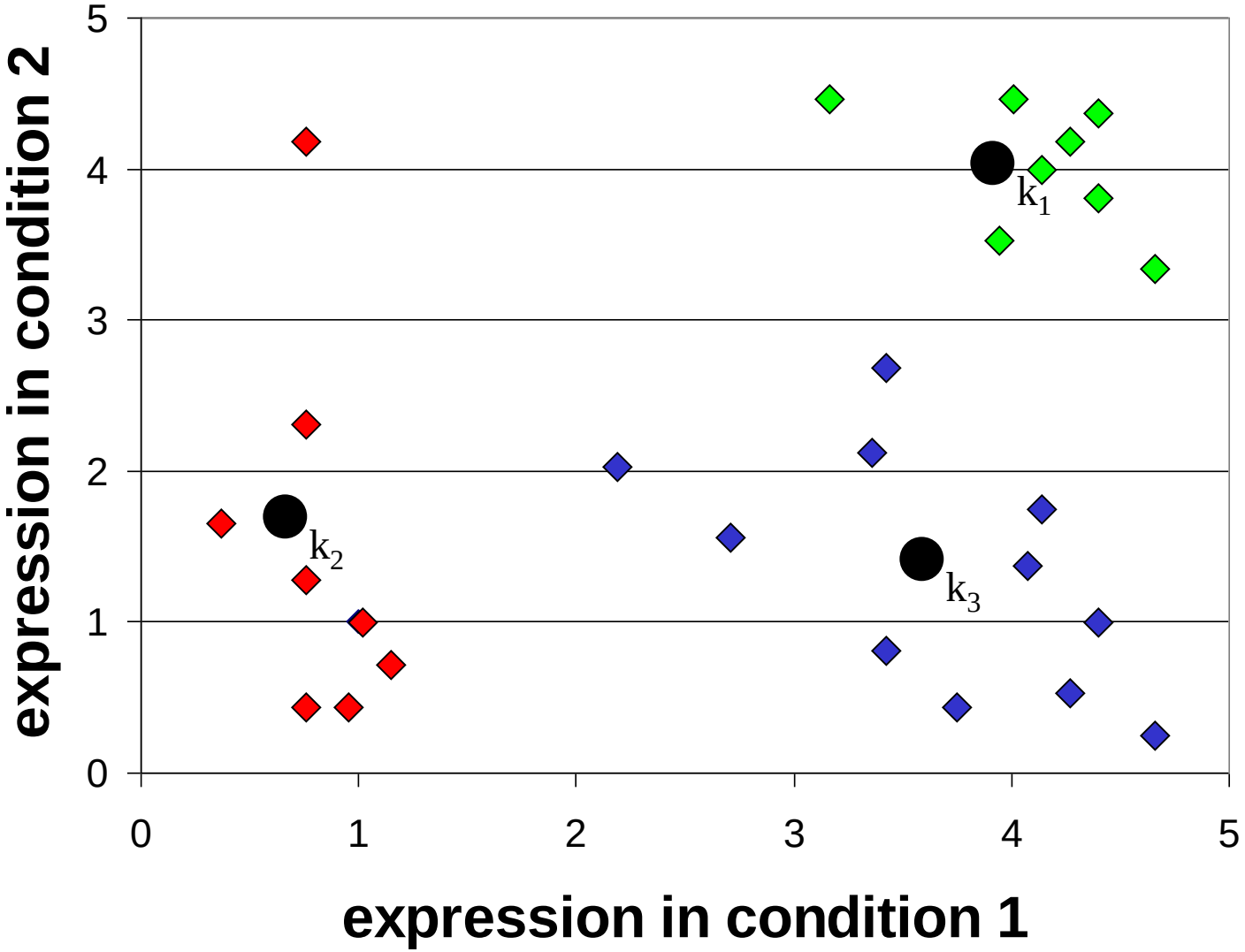


K-means Clustering: Step 4

Algorithm: k-means, Distance Metric: Euclidean Distance



K-means Clustering: Step 5

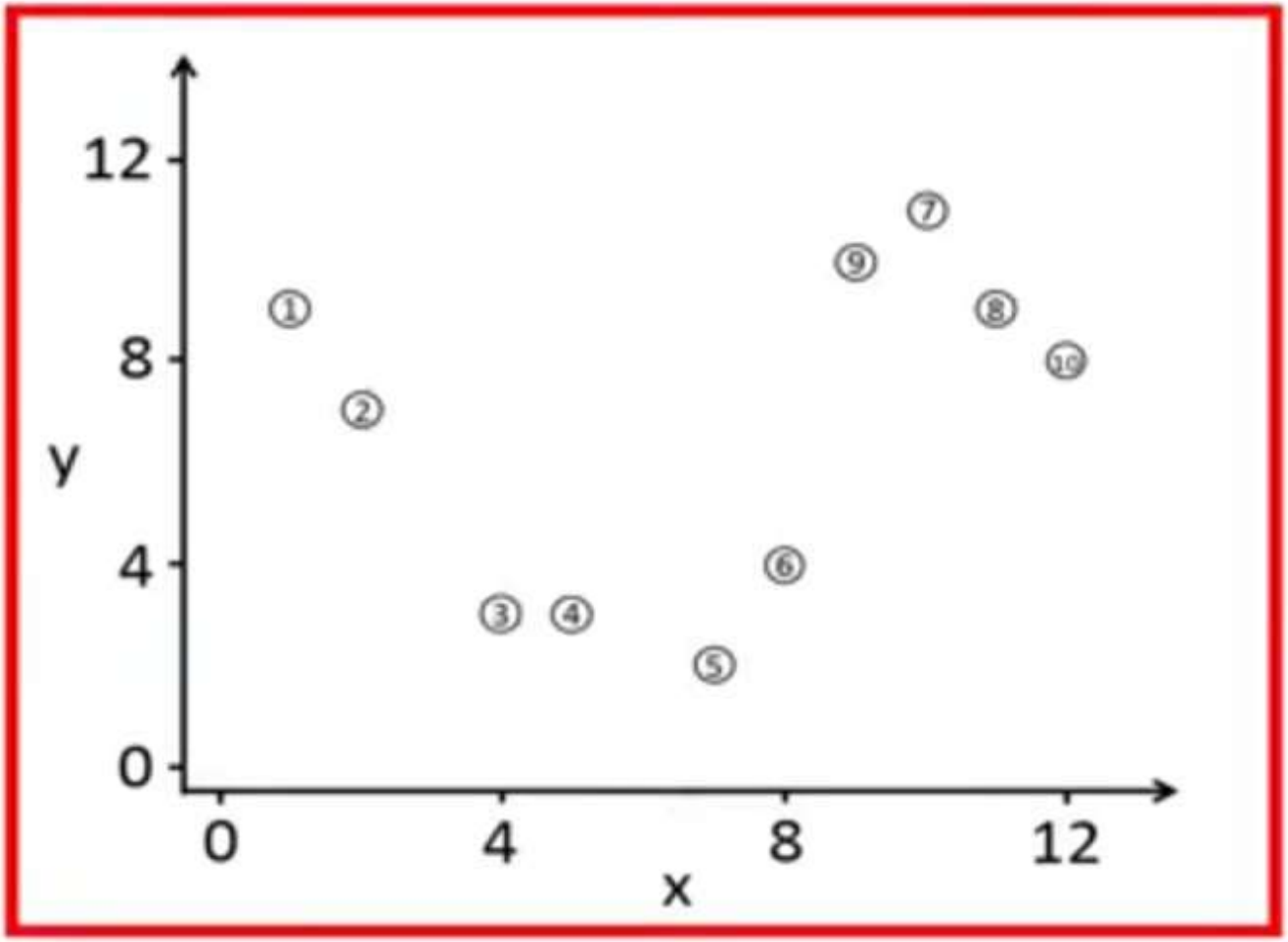


Clustering

K-means clustering is a method to divide the data into *k* clusters or groups. In comparison to hierarchical clustering, k-means clustering allows the user to determine the number of clusters that should be generated.

k-means clustering

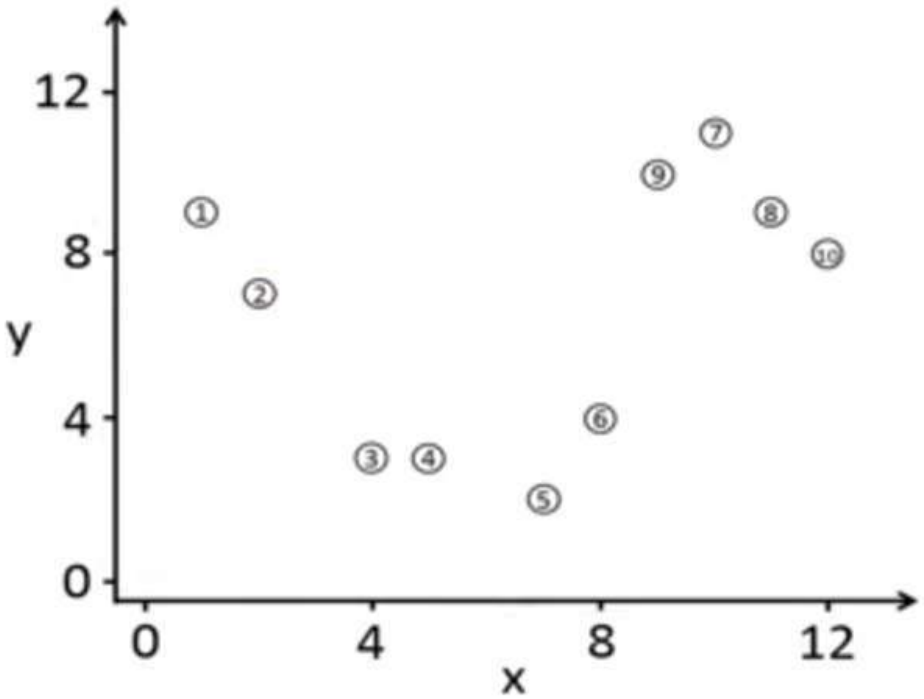
	x	y
1	1	9
2	2	7
3	4	3
4	5	3
5	7	2
6	8	4
7	10	11
8	11	9
9	9	10
10	12	8



Let's make a scatter plot of the data,

$k=3$

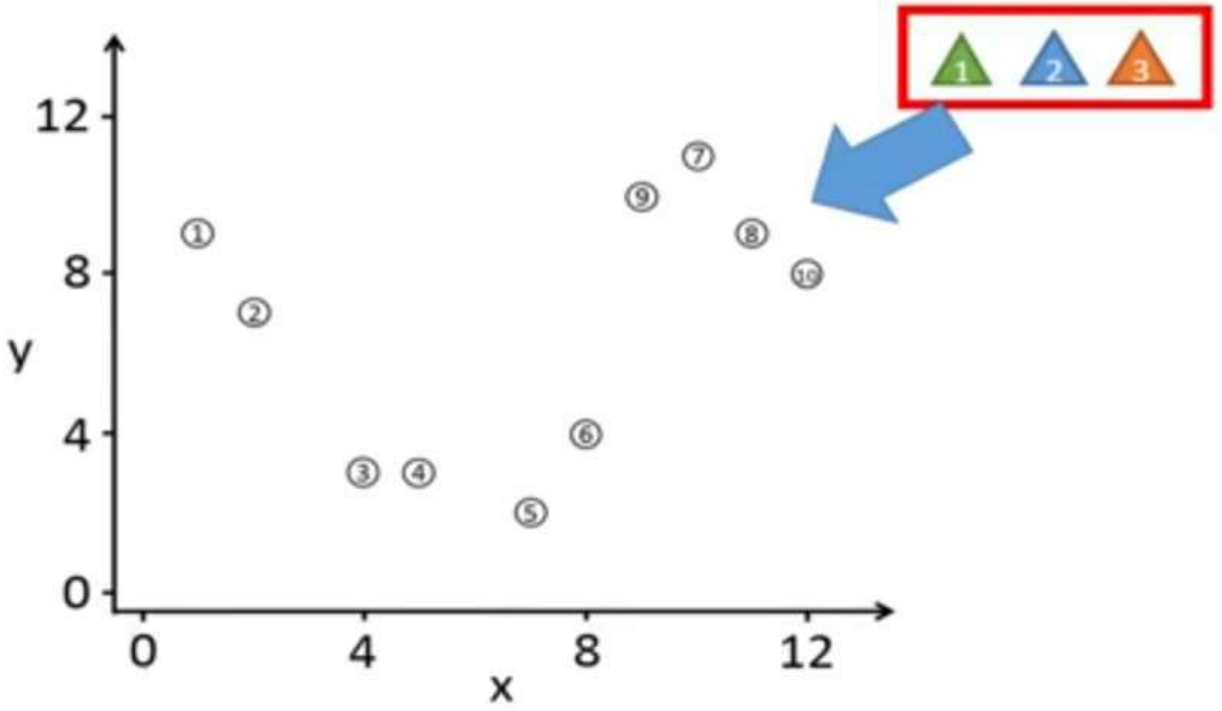
	x	y
1	1	9
2	2	7
3	4	3
4	5	3
5	7	2
6	8	4
7	10	11
8	11	9
9	9	10
10	12	8



The first thing that we should do is to determine the value of *k*, which is the number of clusters the algorithm should identify. Later, we will try different values of *k*, but we here begin to set *k* to three.

k=3

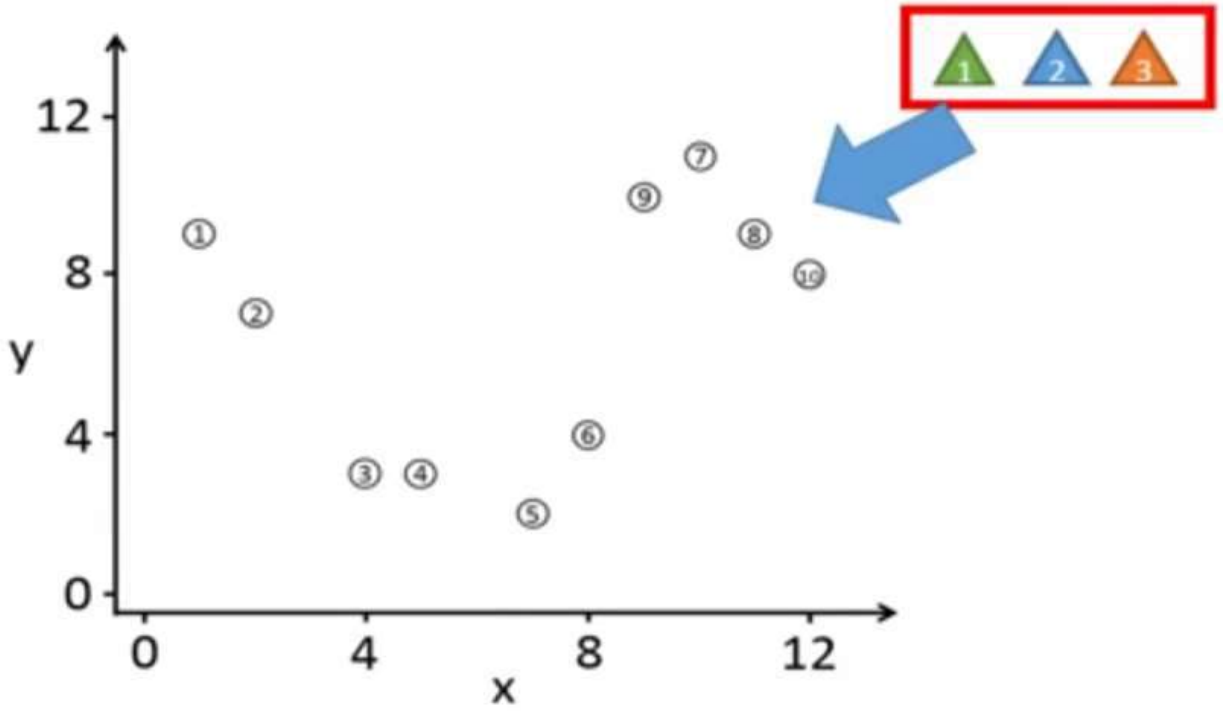
	x	y
1	1	9
2	2	7
3	4	3
4	5	3
5	7	2
6	8	4
7	10	11
8	11	9
9	9	10
10	12	8



Next, we should place the so-called centroids at some initial positions. Since k is equal to three, we need three centroids. Several different methods can be used to randomly select the initial positions of these centroids.

k=3

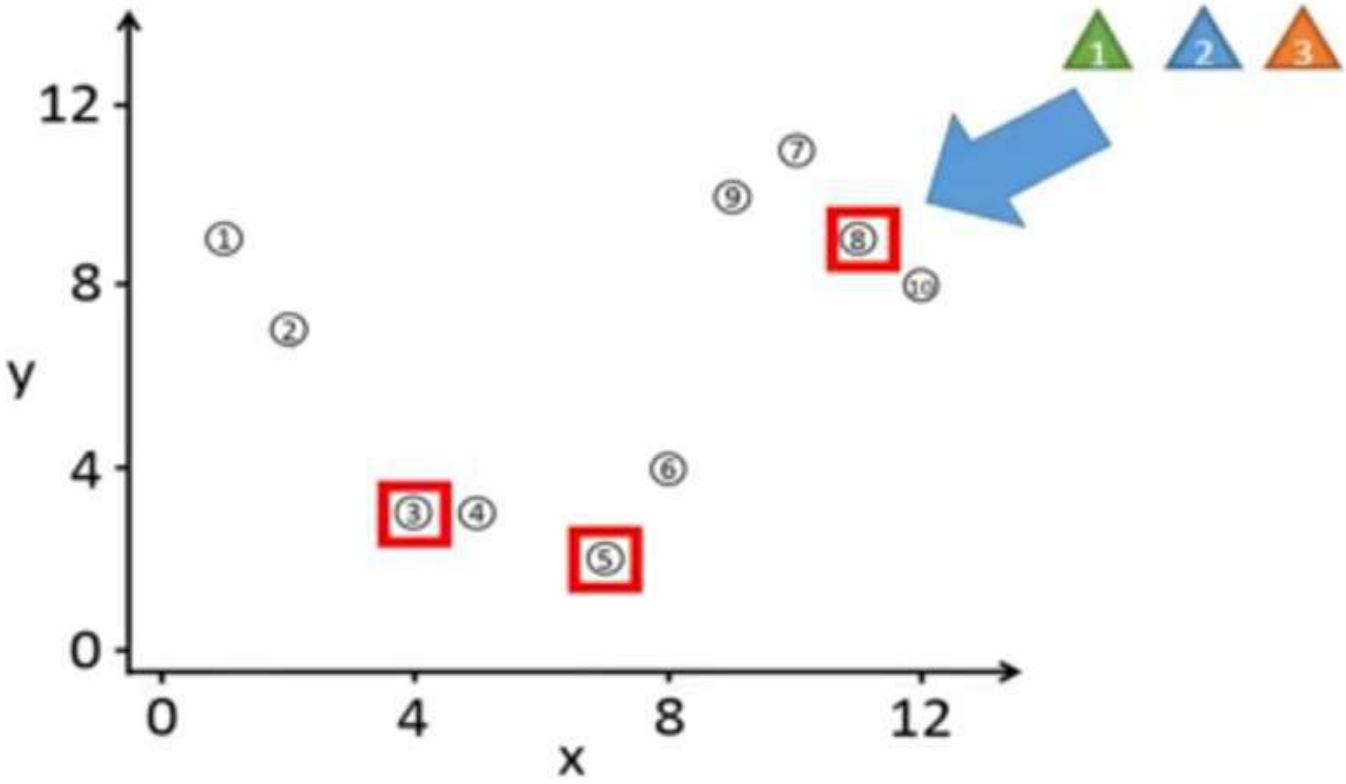
	x	y
1	1	9
2	2	7
3	4	3
4	5	3
5	7	2
6	8	4
7	10	11
8	11	9
9	9	10
10	12	8



One way is to use the Forgy method, which chooses k random data points and use the coordinates of these as initial positions for the centroids.

k=3

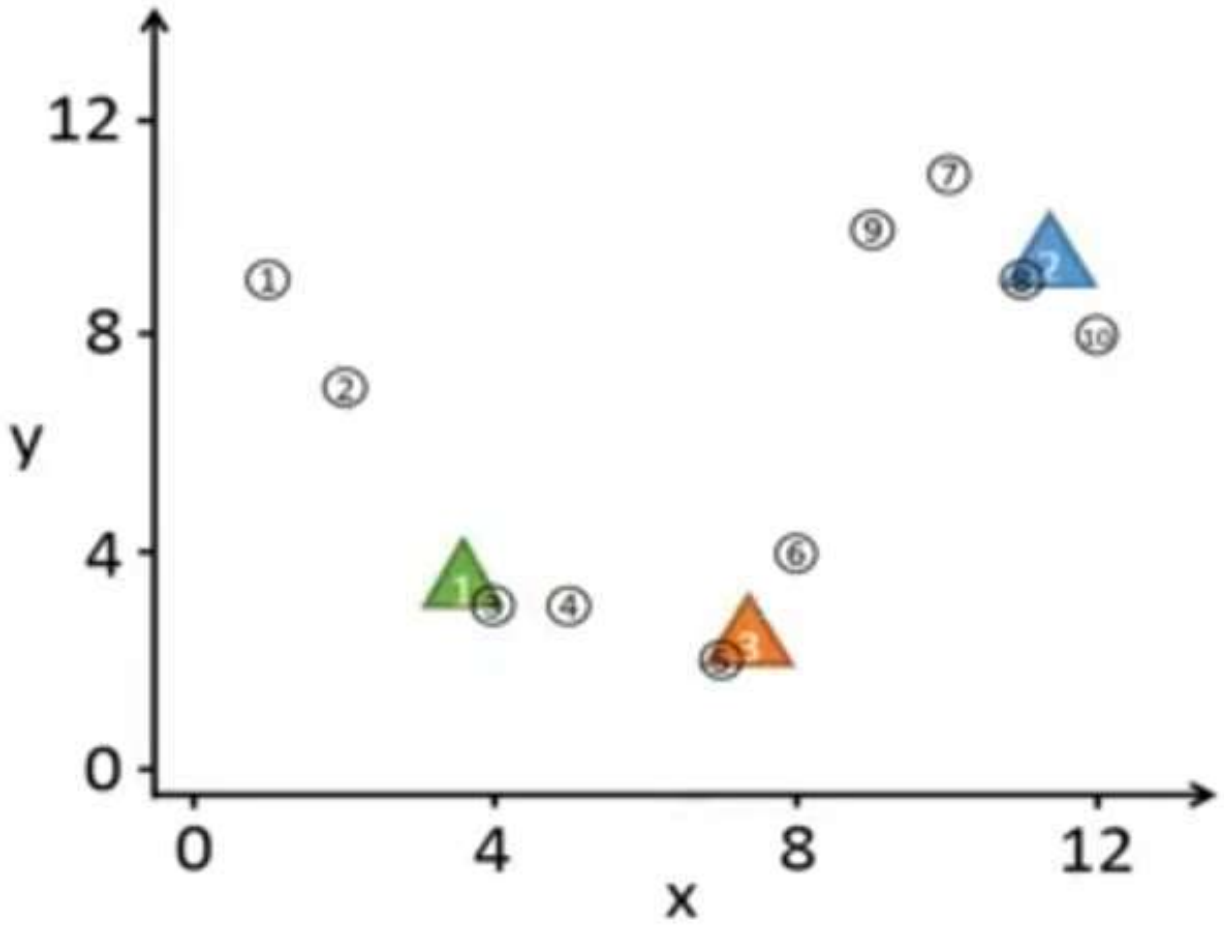
	x	y
1	1	9
2	2	7
3	4	3
4	5	3
5	7	2
6	8	4
7	10	11
8	11	9
9	9	10
10	12	8



For example, let's say that the following three data points were randomly selected.

$k=3$

	x	y
1	1	9
2	2	7
3	4	3
4	5	3
5	7	2
6	8	4
7	10	11
8	11	9
9	9	10
10	12	8

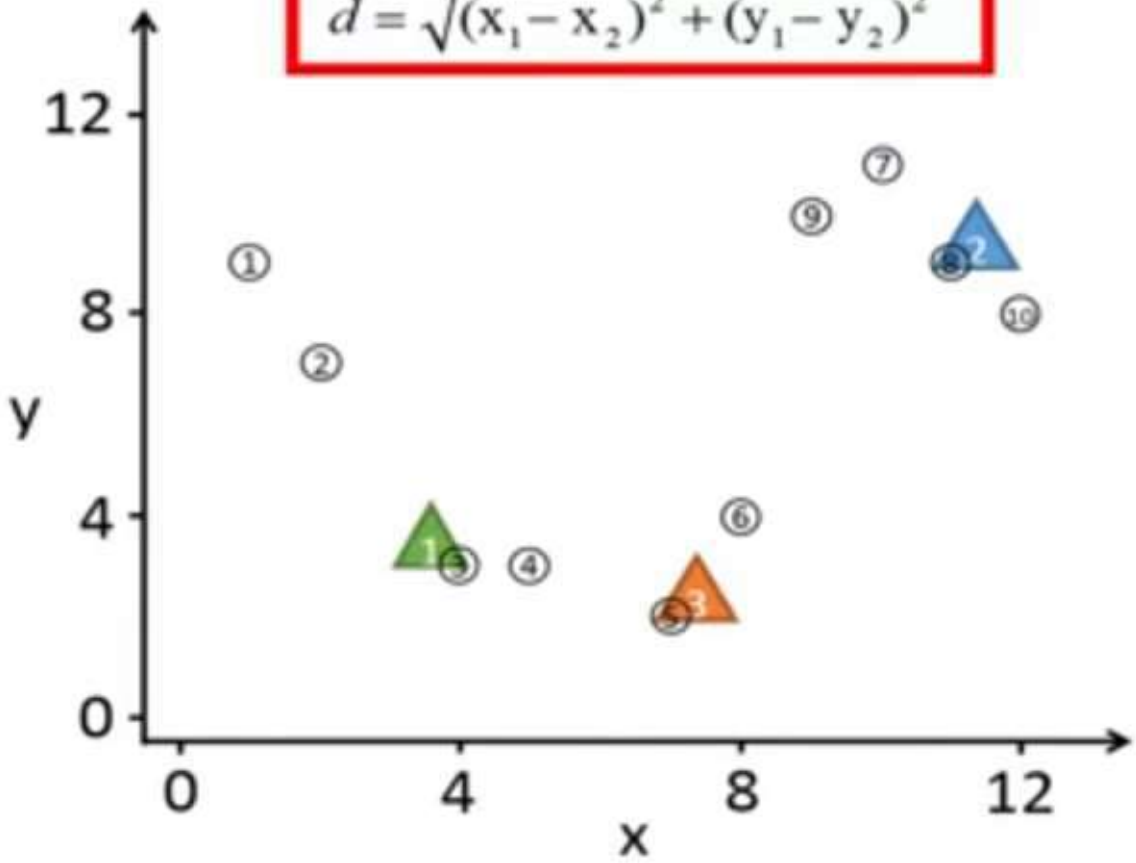


We therefore move the centroids to the coordinates of these three data points.

k=3

	x	y
1	1	9
2	2	7
3	4	3
4	5	3
5	7	2
6	8	4
7	10	11
8	11	9
9	9	10
10	12	8

$$d = \sqrt{(x_1 - x_2)^2 + (y_1 - y_2)^2}$$

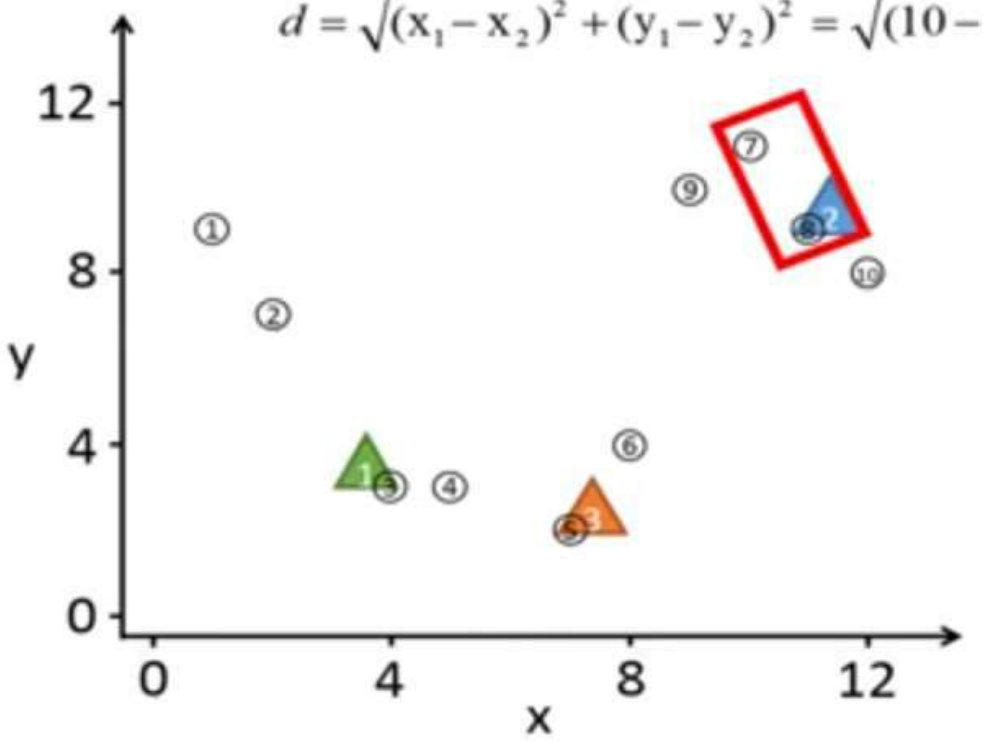


Next, we assign each data point to the closest centroid based on, for example, the Euclidean distance.

k=3

	x	y
1	1	9
2	2	7
3	4	3
4	5	3
5	7	2
6	8	4
7	10	11
8	11	9
9	9	10
10	12	8

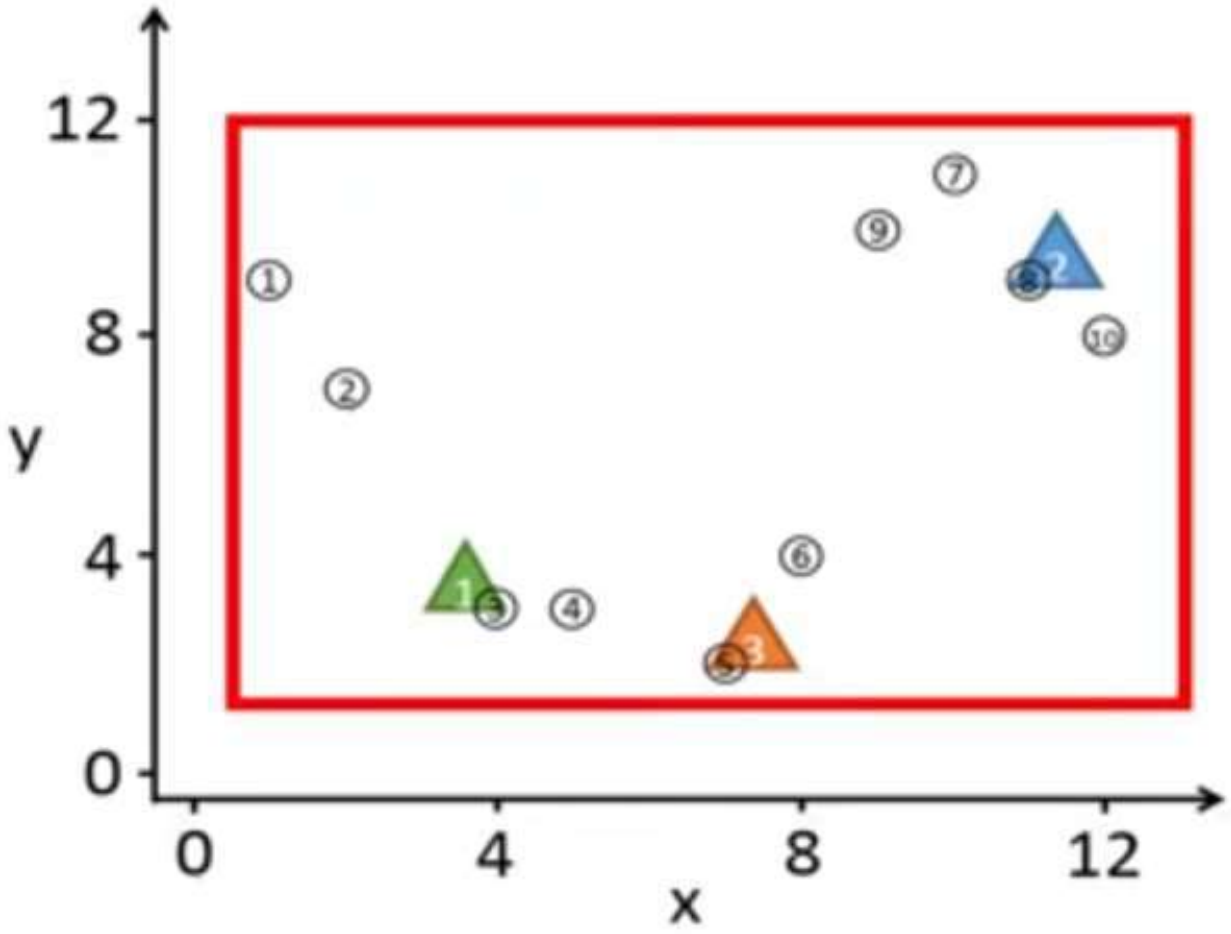
$$d = \sqrt{(x_1 - x_2)^2 + (y_1 - y_2)^2} = \sqrt{(10-11)^2 + (11-9)^2} = \sqrt{5}$$



To calculate the Euclidean distance between data point number 7 and the second centroid,

$k=3$

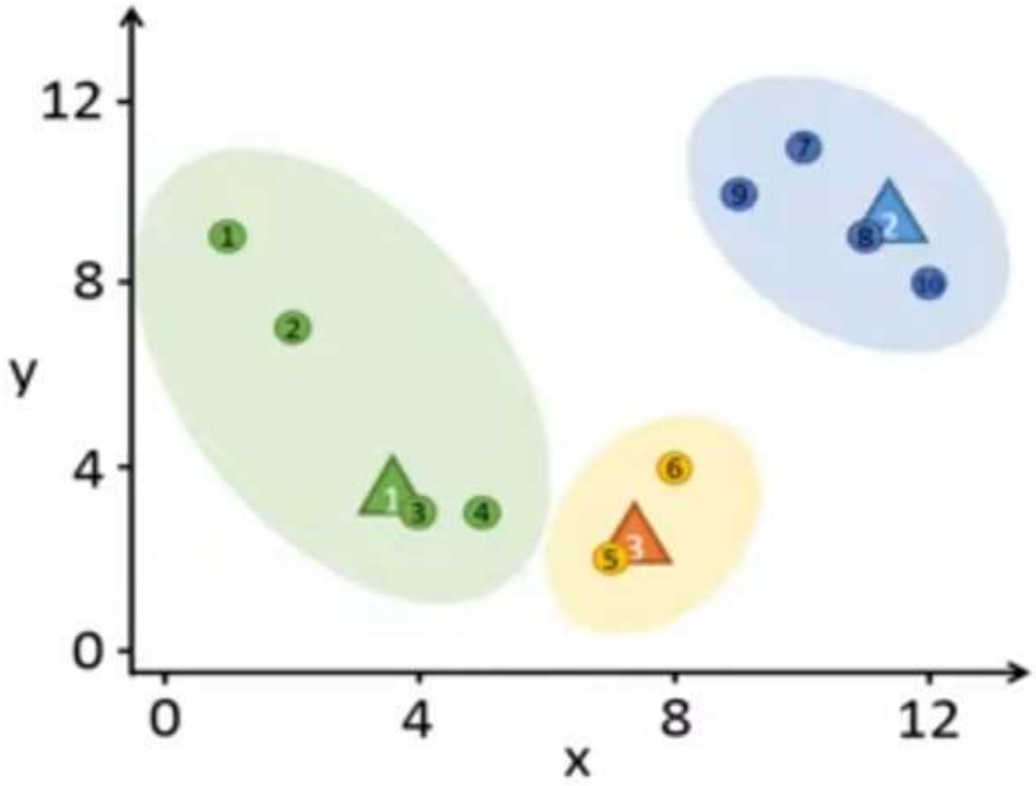
	x	y
1	1	9
2	2	7
3	4	3
4	5	3
5	7	2
6	8	4
7	10	11
8	11	9
9	9	10
10	12	8



We calculate the Euclidean distance between each data point and all the centroids.

k=3

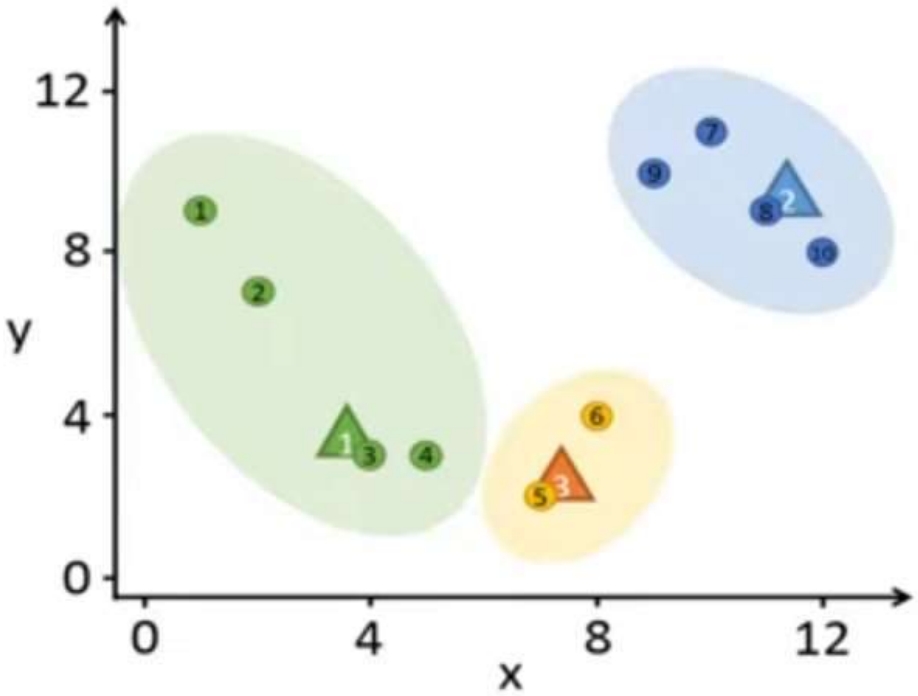
	x	y
1	1	9
2	2	7
3	4	3
4	5	3
5	7	2
6	8	4
7	10	11
8	11	9
9	9	10
10	12	8



and that these data points are closest to centroid number two. We have therefore created three initial clusters.

$k=3$

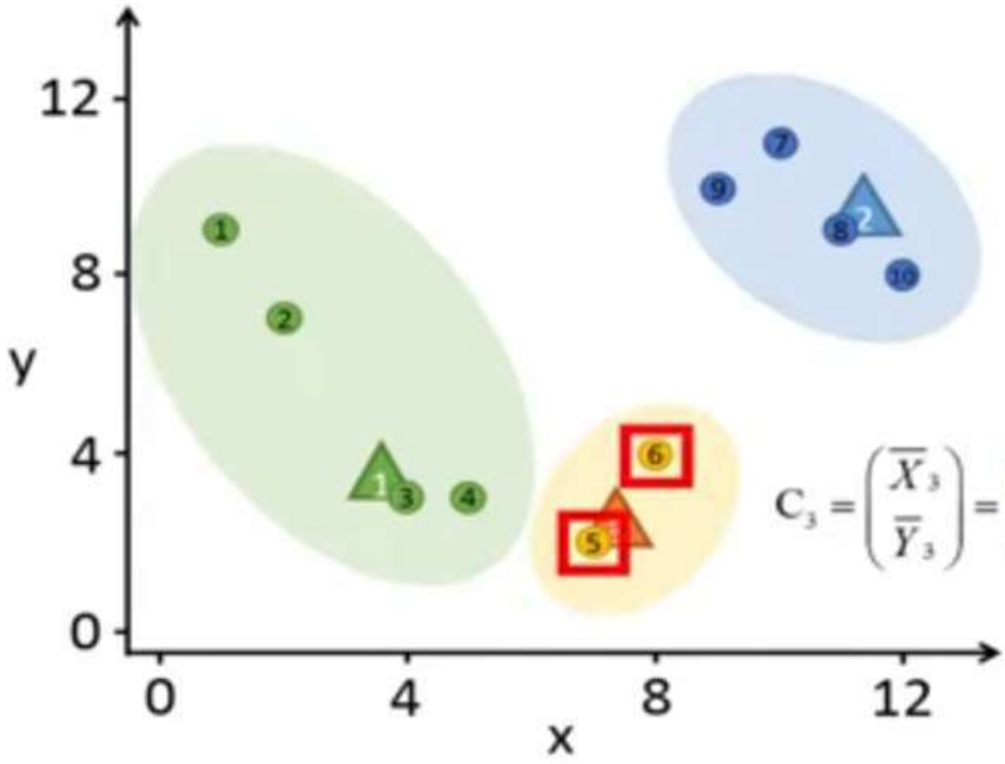
	x	y
1	1	9
2	2	7
3	4	3
4	5	3
5	7	2
6	8	4
7	10	11
8	11	9
9	9	10
10	12	8



Next, we update the positions of the centroids so that they correspond to the mean x and y positions of the data points in the corresponding cluster.

k=3

	x	y
1	1	9
2	2	7
3	4	3
4	5	3
5	7	2
6	8	4
7	10	11
8	11	9
9	9	10
10	12	8

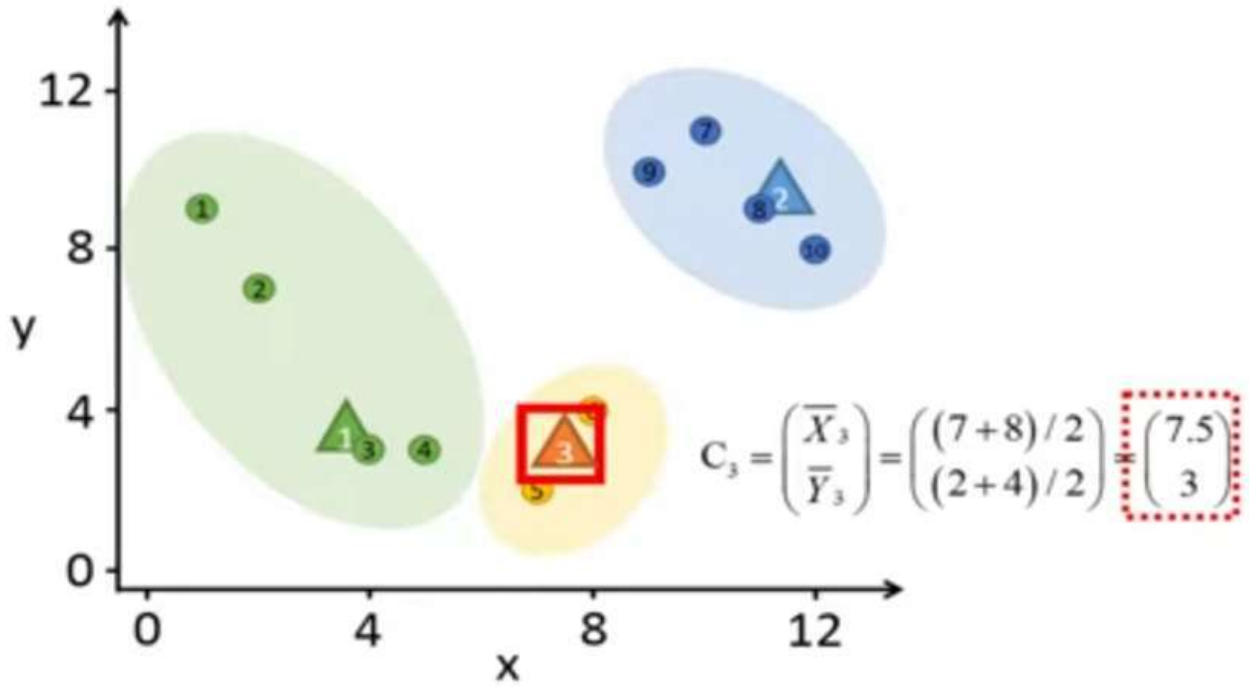


$$C_3 = \begin{pmatrix} \overline{X}_3 \\ \overline{Y}_3 \end{pmatrix} = \begin{pmatrix} (7+8)/2 \\ (2+4)/2 \end{pmatrix} = \begin{pmatrix} 7.5 \\ 3 \end{pmatrix}$$

For example, the mean of the x-coordinates of the two data points in cluster three is 7.5,

k=3

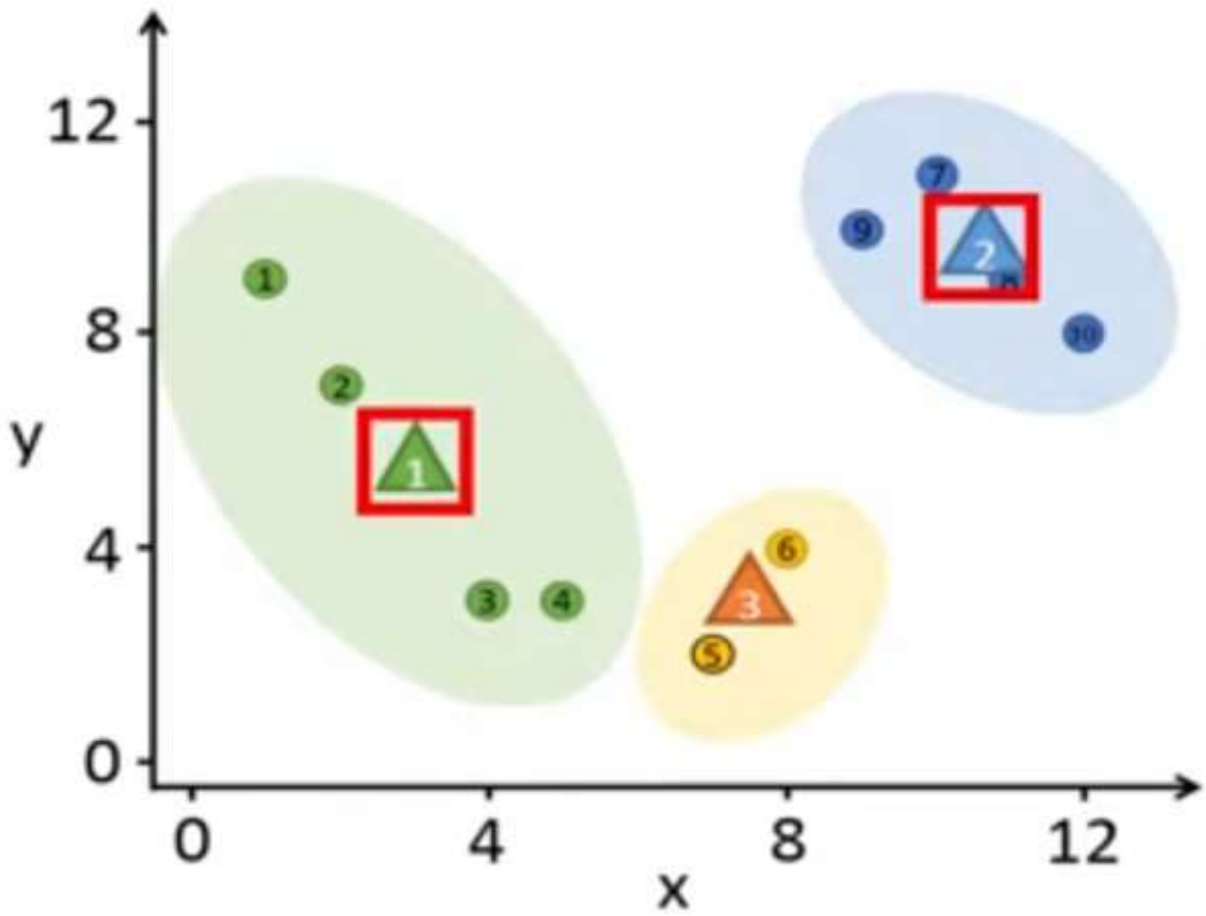
	x	y
1	1	9
2	2	7
3	4	3
4	5	3
5	7	2
6	8	4
7	10	11
8	11	9
9	9	10
10	12	8



We therefore move this centroid to the coordinates that represent the mean position of the two data points in the third cluster.

$k=3$

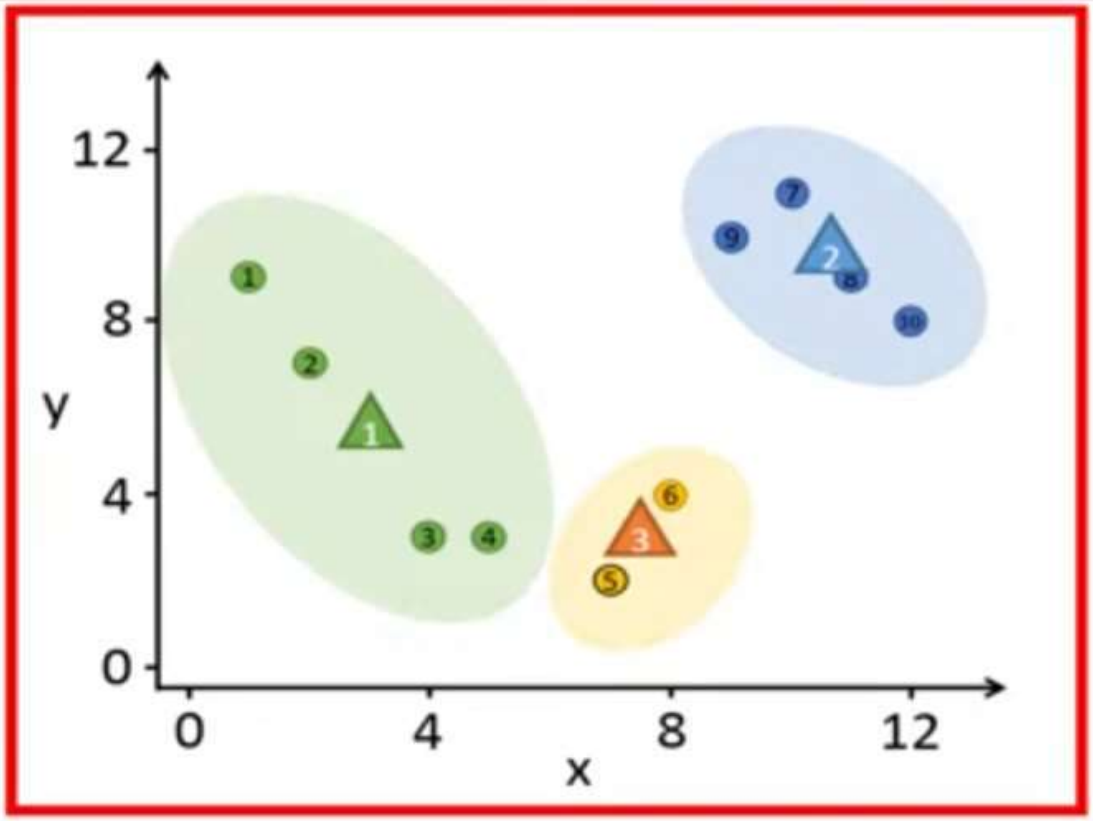
	x	y
1	1	9
2	2	7
3	4	3
4	5	3
5	7	2
6	8	4
7	10	11
8	11	9
9	9	10
10	12	8



Similarly, we update the positions of the centroids of the first and second clusters.

k=3

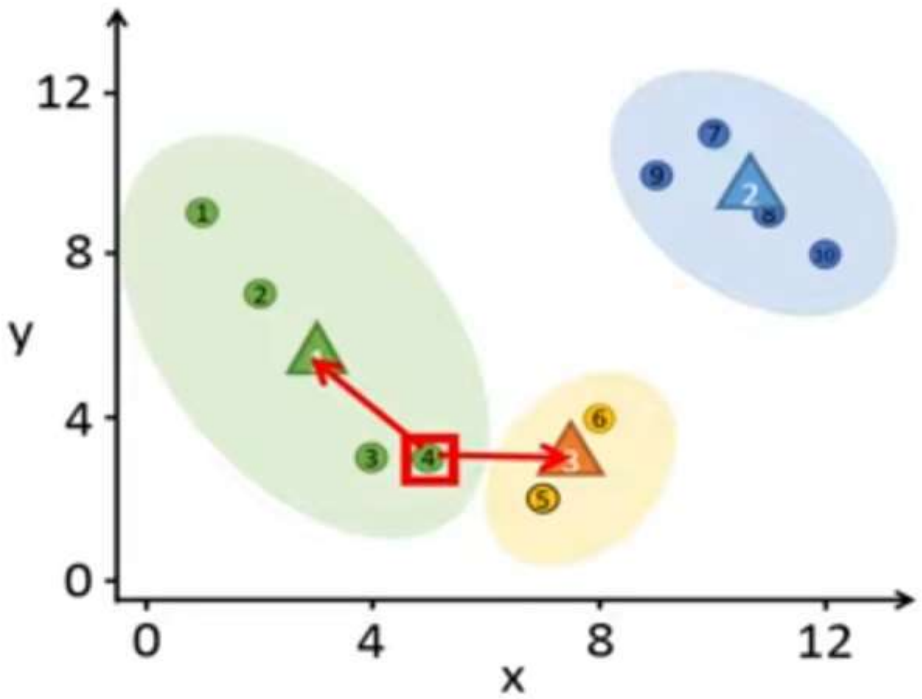
	x	y
1	1	9
2	2	7
3	4	3
4	5	3
5	7	2
6	8	4
7	10	11
8	11	9
9	9	10
10	12	8



We now repeat the previous steps, where we first find the shortest distance between all data points and the centroids.

$k=3$

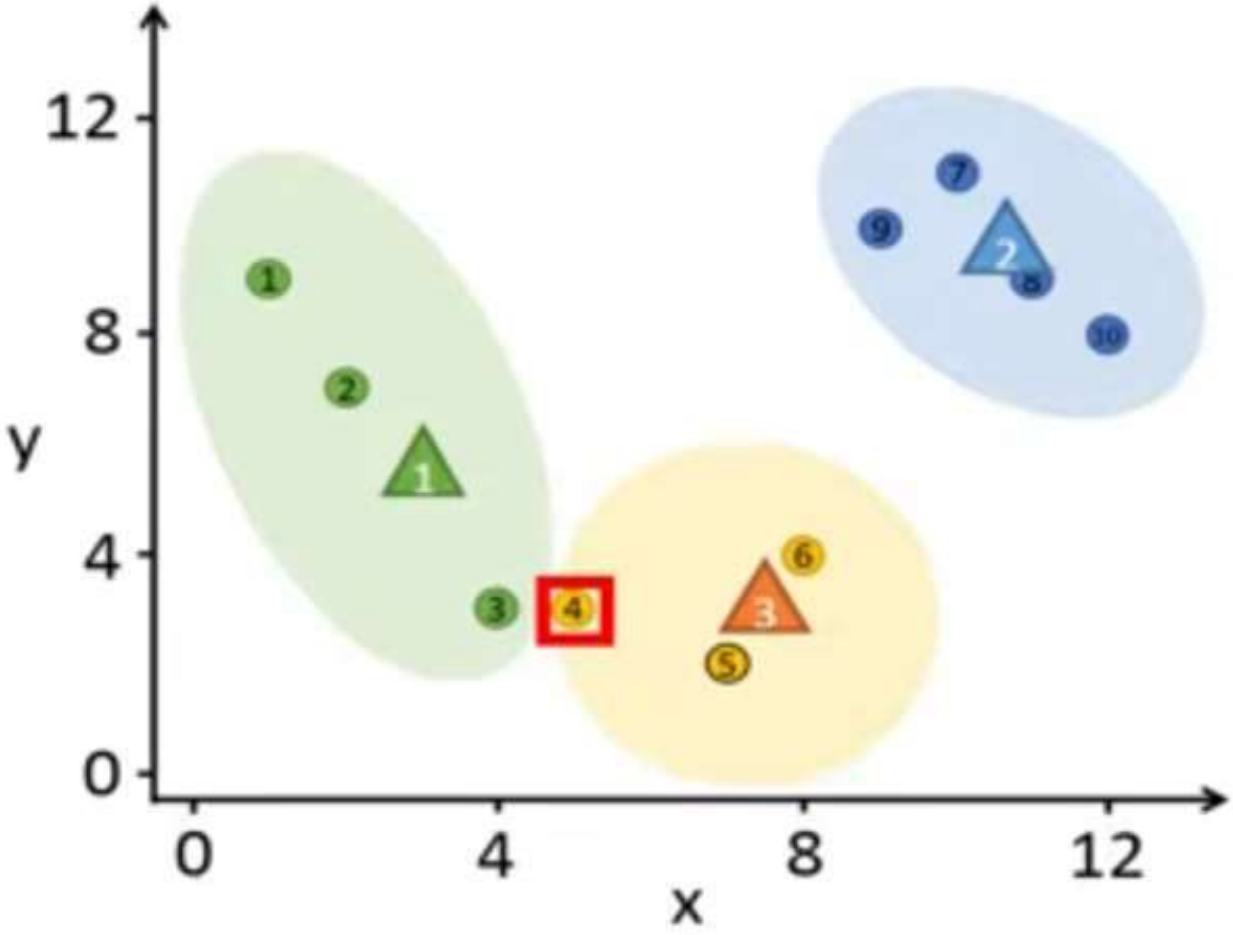
	x	y
1	1	9
2	2	7
3	4	3
4	5	3
5	7	2
6	8	4
7	10	11
8	11	9
9	9	10
10	12	8



Since data point number 4 now has a shorter distance to the centroid in the third cluster, compared to the centroid in the first cluster,

$k=3$

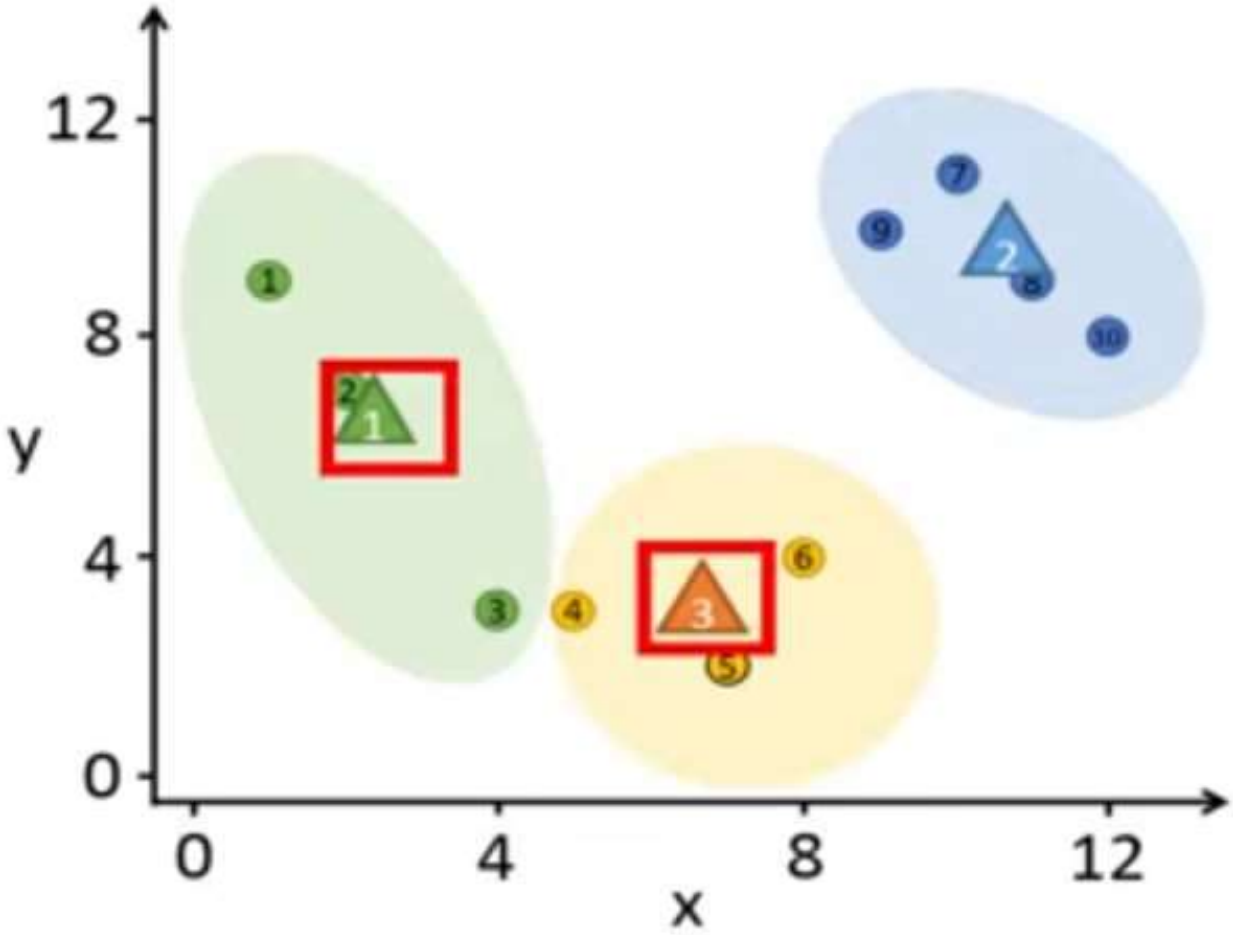
	x	y
1	1	9
2	2	7
3	4	3
4	5	3
5	7	2
6	8	4
7	10	11
8	11	9
9	9	10
10	12	8



it will therefore switch cluster from cluster one to cluster three.

$k=3$

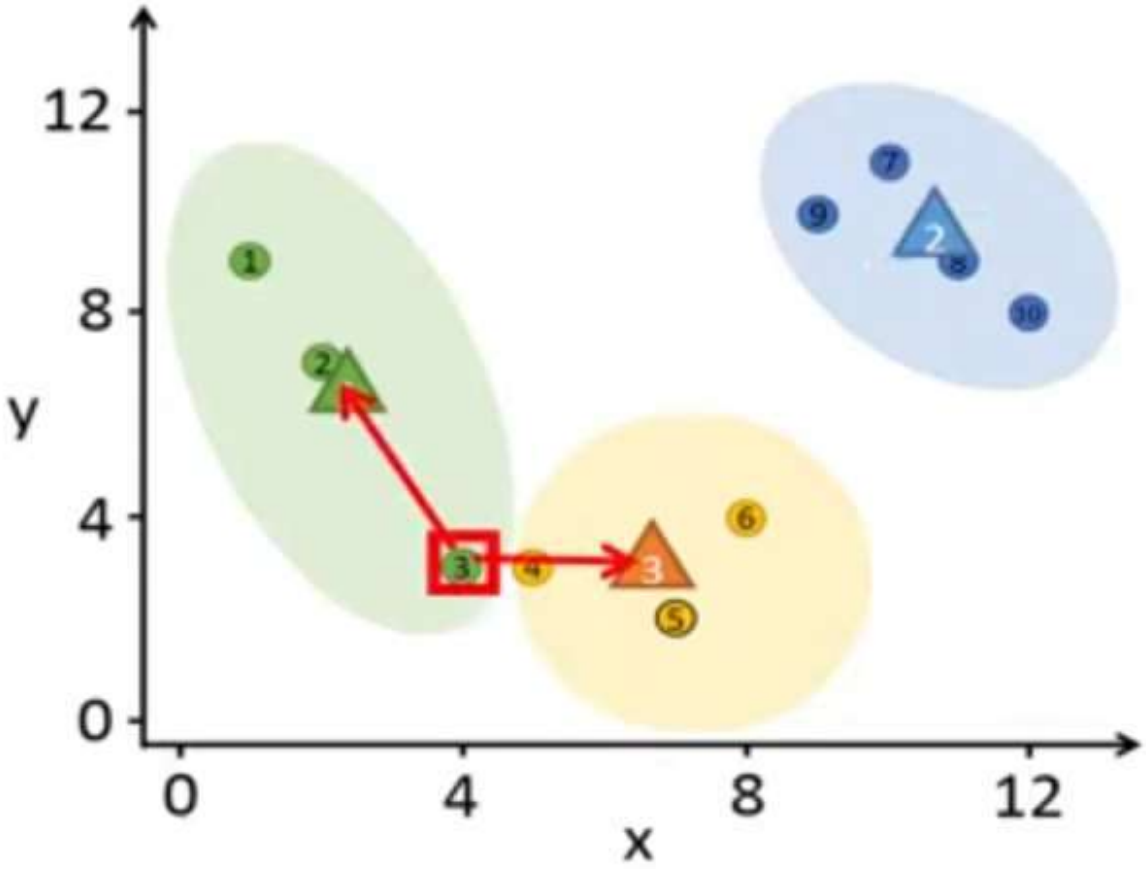
	x	y
1	1	9
2	2	7
3	4	3
4	5	3
5	7	2
6	8	4
7	10	11
8	11	9
9	9	10
10	12	8



Next, we update the positions of the centroids based on their associated data points.

k=3

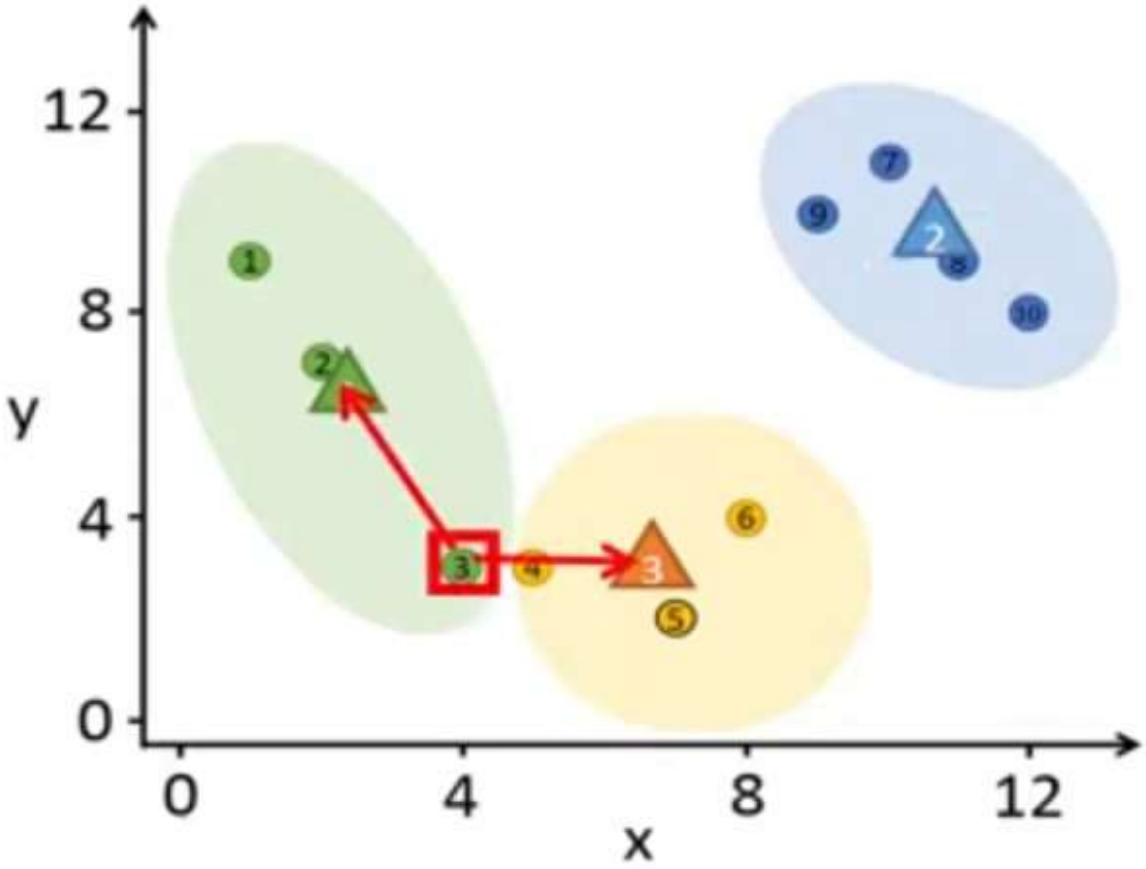
	x	y
1	1	9
2	2	7
3	4	3
4	5	3
5	7	2
6	8	4
7	10	11
8	11	9
9	9	10
10	12	8



Since data point number 3 is now closer to the centroid of the third cluster it will also switch cluster.

k=3

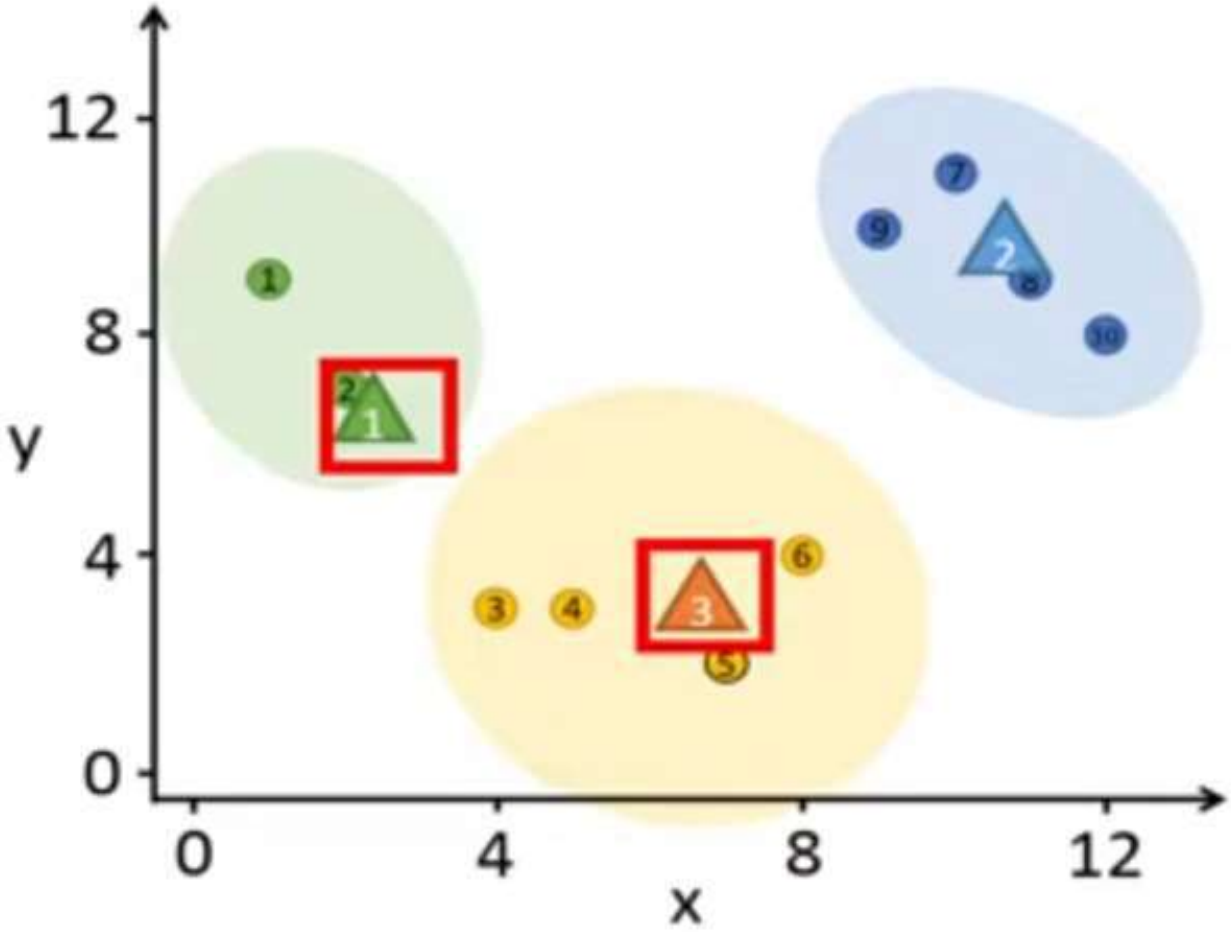
	x	y
1	1	9
2	2	7
3	4	3
4	5	3
5	7	2
6	8	4
7	10	11
8	11	9
9	9	10
10	12	8



Since data point number 3 is now closer to the centroid of the third cluster it will also switch cluster.

$k=3$

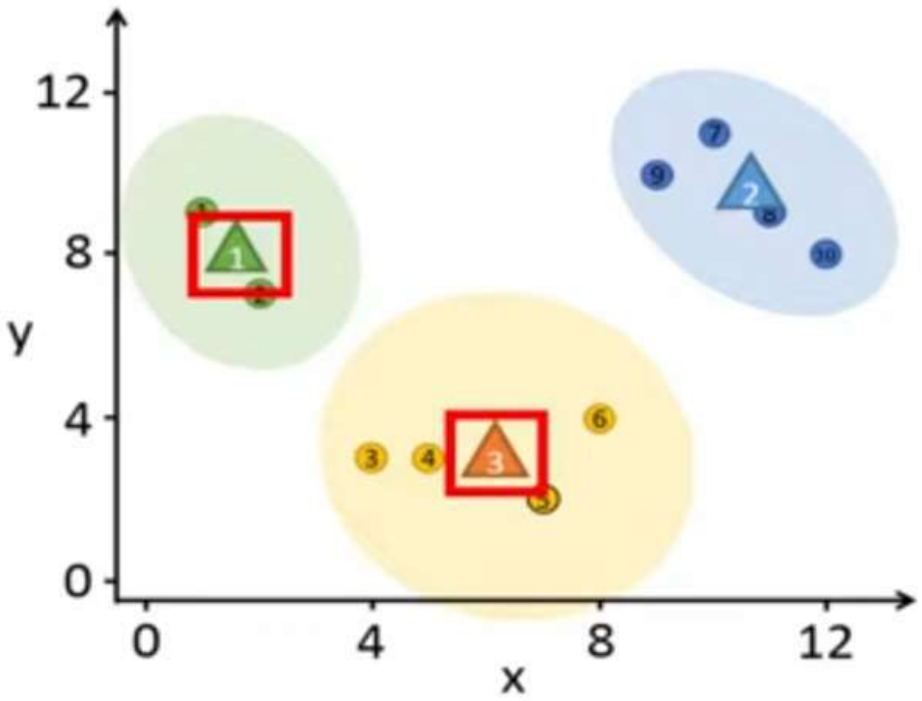
	x	y
1	1	9
2	2	7
3	4	3
4	5	3
5	7	2
6	8	4
7	10	11
8	11	9
9	9	10
10	12	8



Next, we update the positions of the centroids again,

$k=3$

	x	y
1	1	9
2	2	7
3	4	3
4	5	3
5	7	2
6	8	4
7	10	11
8	11	9
9	9	10
10	12	8



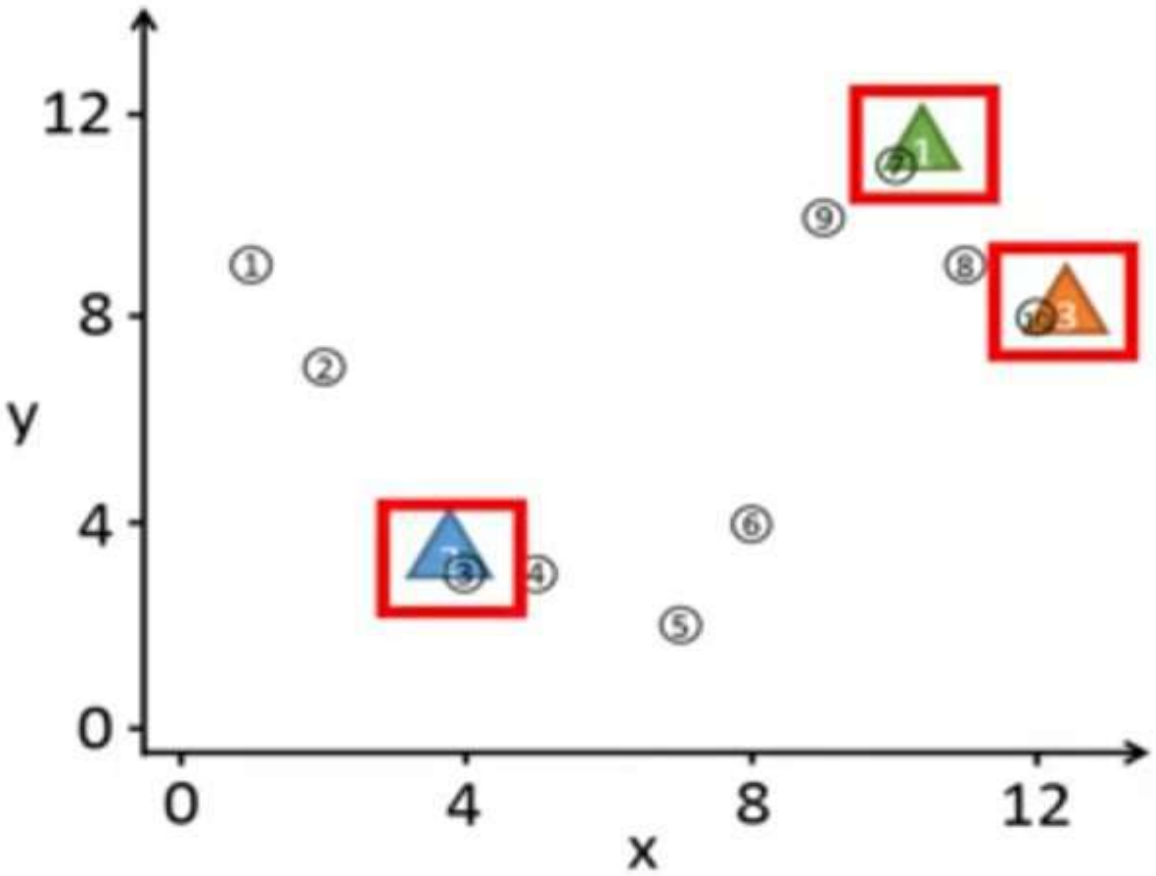
so that they have the following positions. Since all data points are now closest to the centroid of the cluster they belong to, no data point will switch cluster. When this happens, the algorithm will stop.

Initial positions of the centroids

So, do the initial positions of the centroids affect the clustering?

Initial positions of the centroids

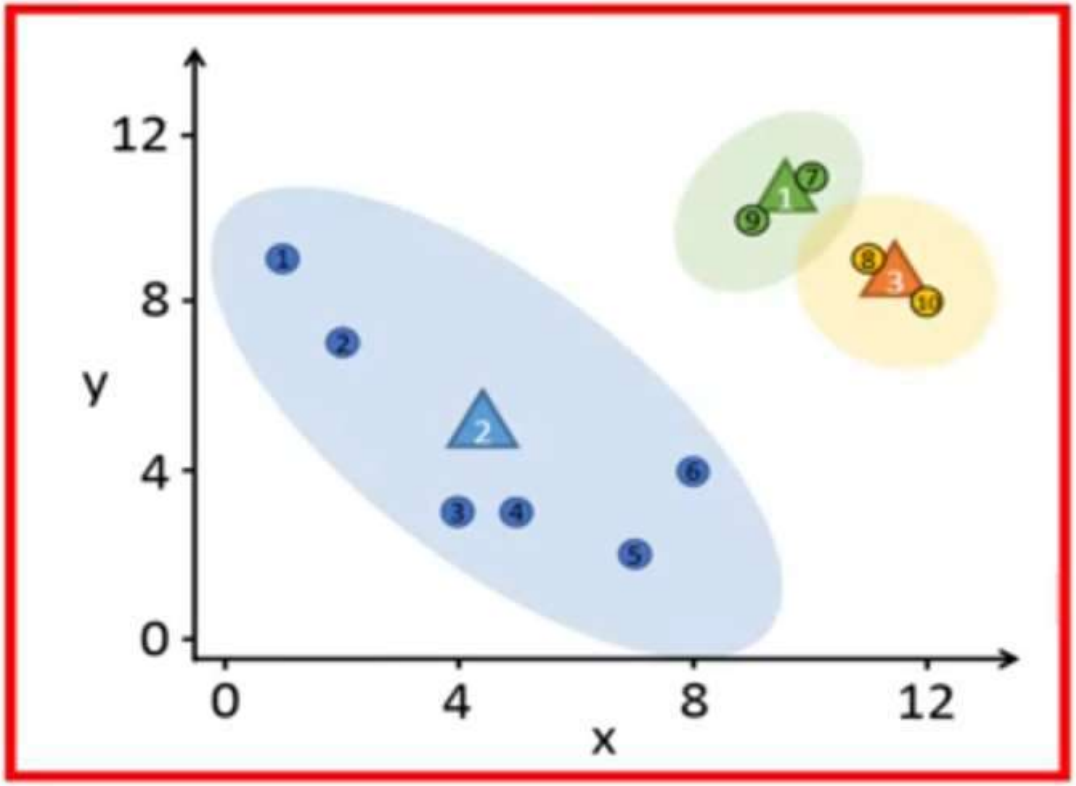
	x	y
1	1	9
2	2	7
3	4	3
4	5	3
5	7	2
6	8	4
7	10	11
8	11	9
9	9	10
10	12	8



For example, what would happen if the centroids are initially placed at the following positions?

Initial positions of the centroids

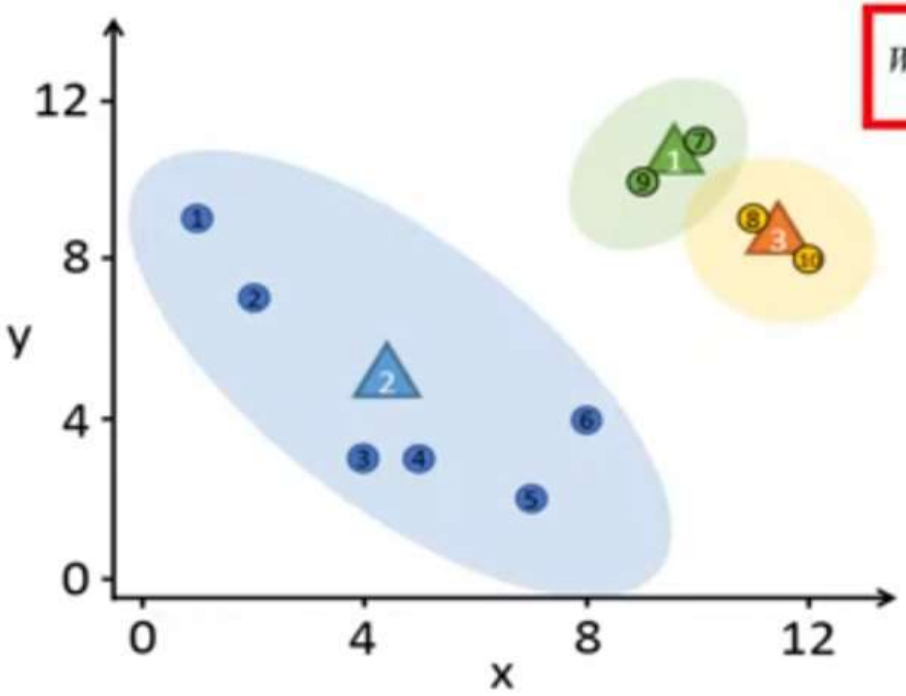
	x	y
1	1	9
2	2	7
3	4	3
4	5	3
5	7	2
6	8	4
7	10	11
8	11	9
9	9	10
10	12	8



The algorithm will then end up with the following three clusters. So, are these clusters better than the ones we generated earlier? To compare different types of outputs, we need some sort of measure for how well the clustering has performed.

Initial positions of the centroids

	x	y
1	1	9
2	2	7
3	4	3
4	5	3
5	7	2
6	8	4
7	10	11
8	11	9
9	9	10
10	12	8

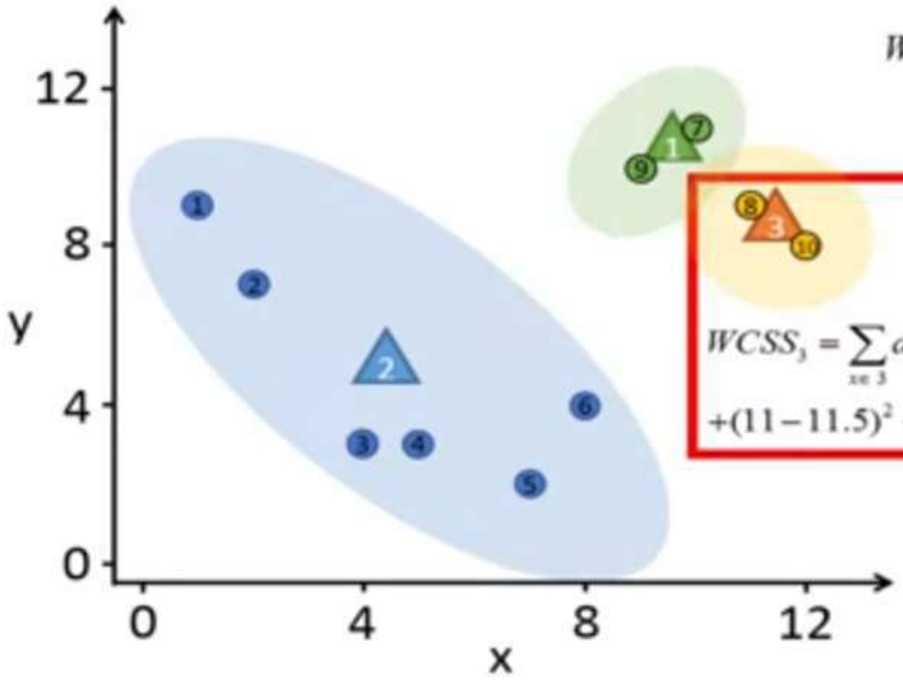


$$WCSS = \sum_{i=1}^{n_c} \sum_{x \in i} d(x, \bar{x}_i)^2$$

One measure is the so-called within cluster sum of squares (WCSS), which is the sum of the squared Euclidean distance between each data point and the centroid of the corresponding cluster. This measure tells how close the data points are to the centroids.

Initial positions of the centroids

	x	y
1	1	9
2	2	7
3	4	3
4	5	3
5	7	2
6	8	4
7	10	11
8	11	9
9	9	10
10	12	8

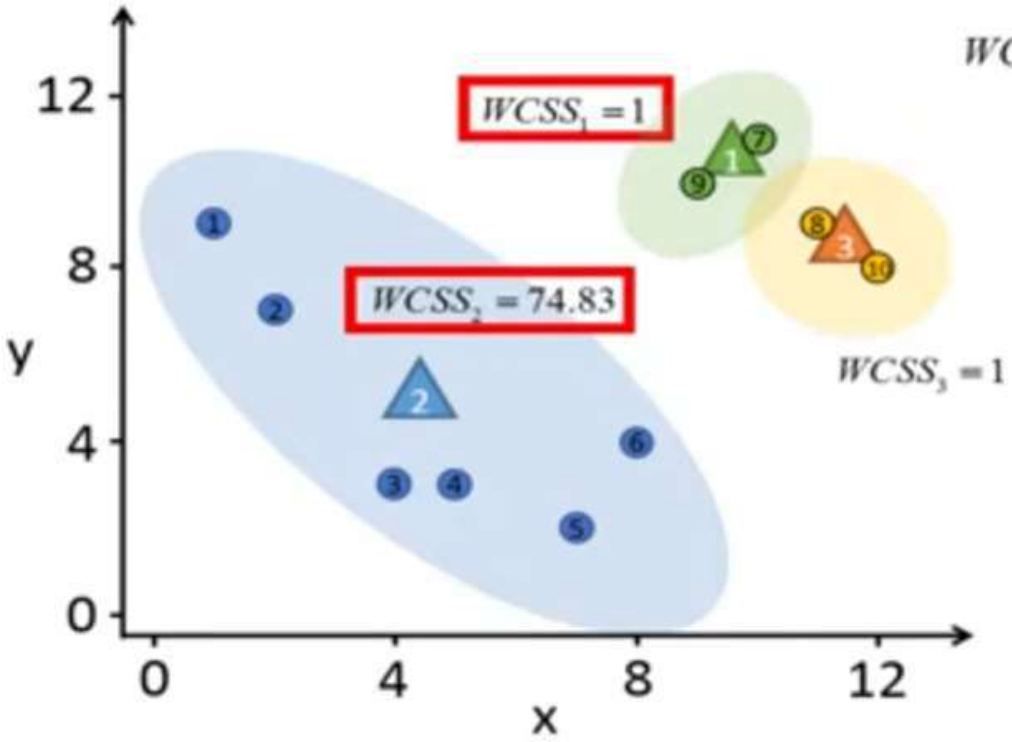


$$WCSS = \sum_{i=1}^{n_c} \sum_{x \in i} d(x, \bar{x}_i)^2$$

$$WCSS_3 = \sum_{x \in 3} d(x, \bar{x}_3)^2 = (12-11.5)^2 + (8-8.5)^2 + (11-11.5)^2 + (9-8.5)^2 = 1$$

For example, the within cluster sum of squares of the third cluster is calculated like this,

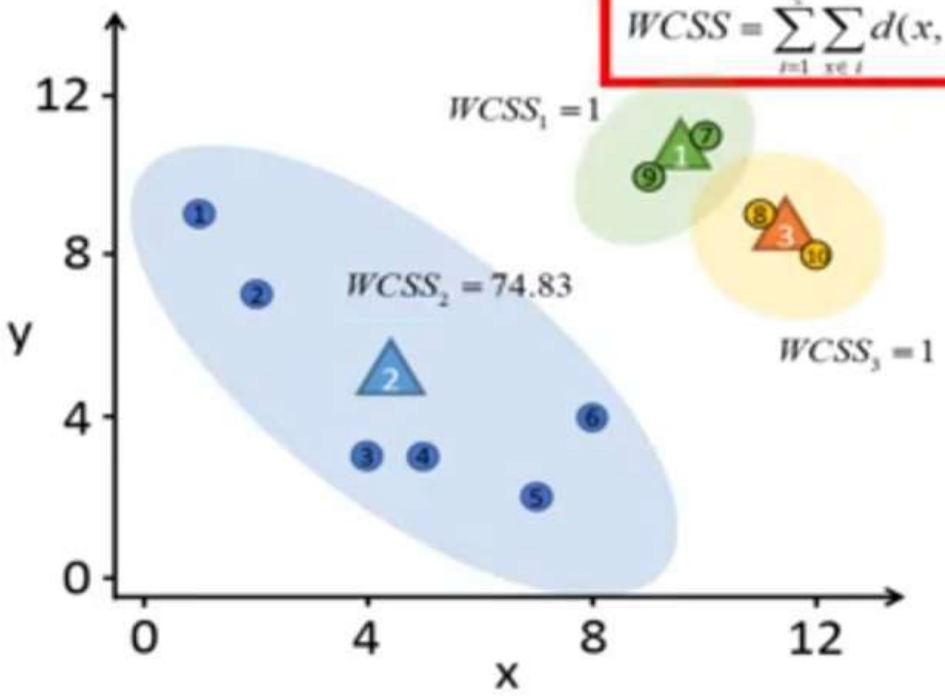
	x	y
1	1	9
2	2	7
3	4	3
4	5	3
5	7	2
6	8	4
7	10	11
8	11	9
9	9	10
10	12	8



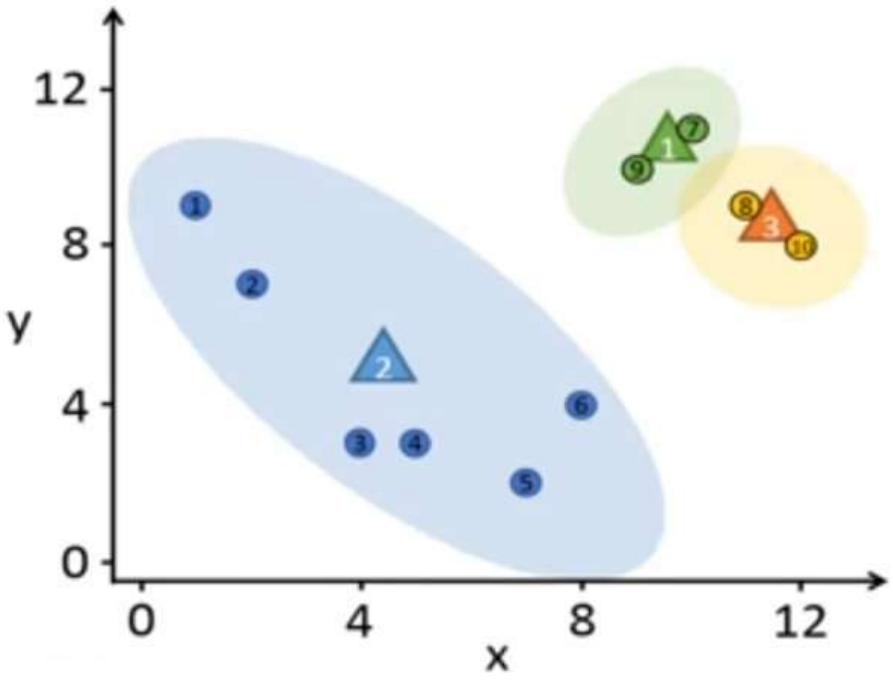
We calculate the within cluster sum of squares for the other two clusters in a similar way.

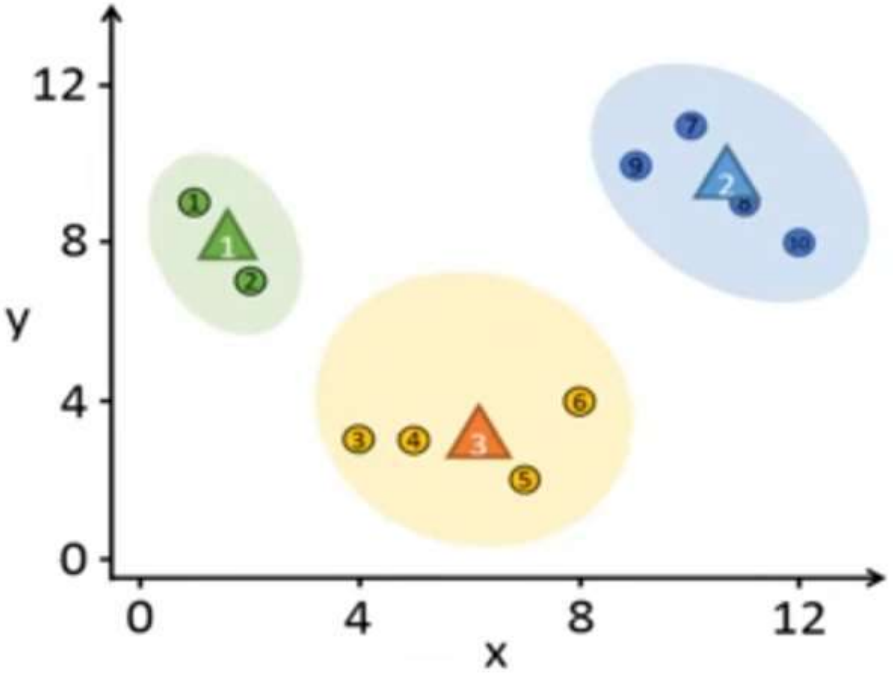
	x	y
1	1	9
2	2	7
3	4	3
4	5	3
5	7	2
6	8	4
7	10	11
8	11	9
9	9	10
10	12	8

$$WCSS = \sum_{i=1}^{n_c} \sum_{x \in i} d(x, \bar{x}_i)^2 = 1 + 74.83 + 1 = 76.83$$

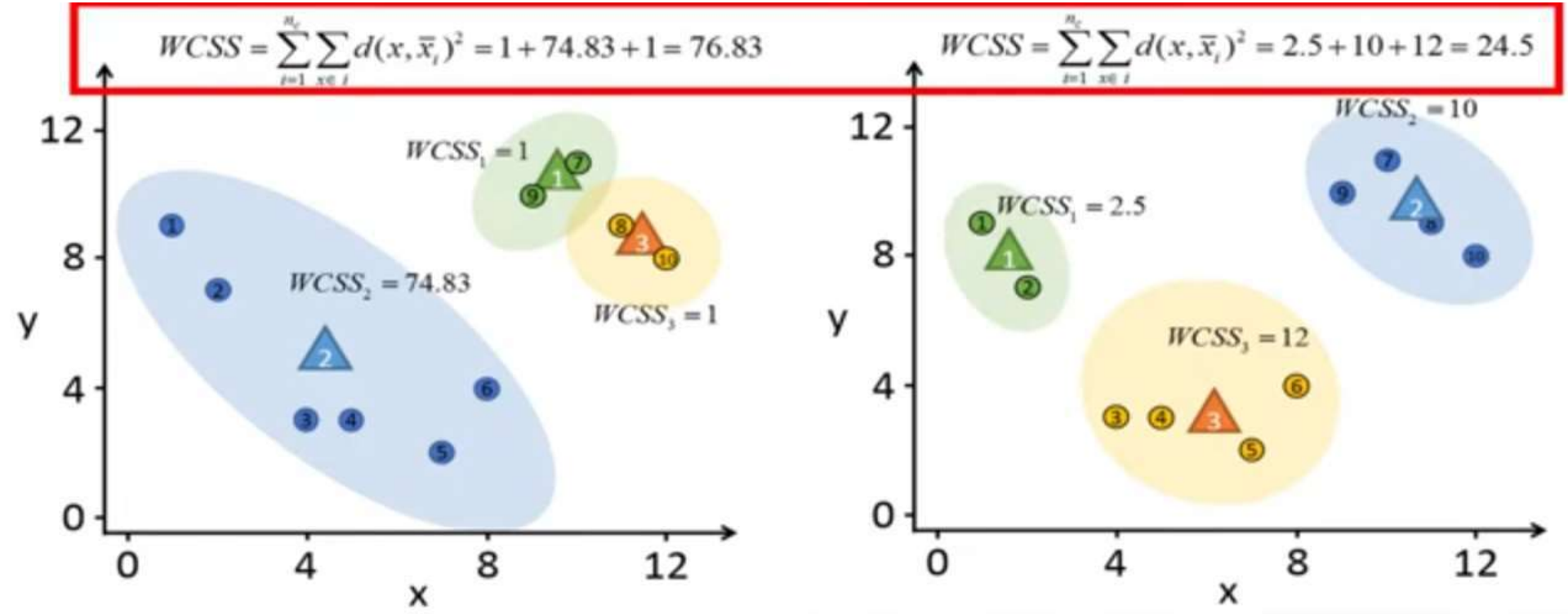


Then we sum these to get the total within cluster sum of squares, which is here equal to 76.83.



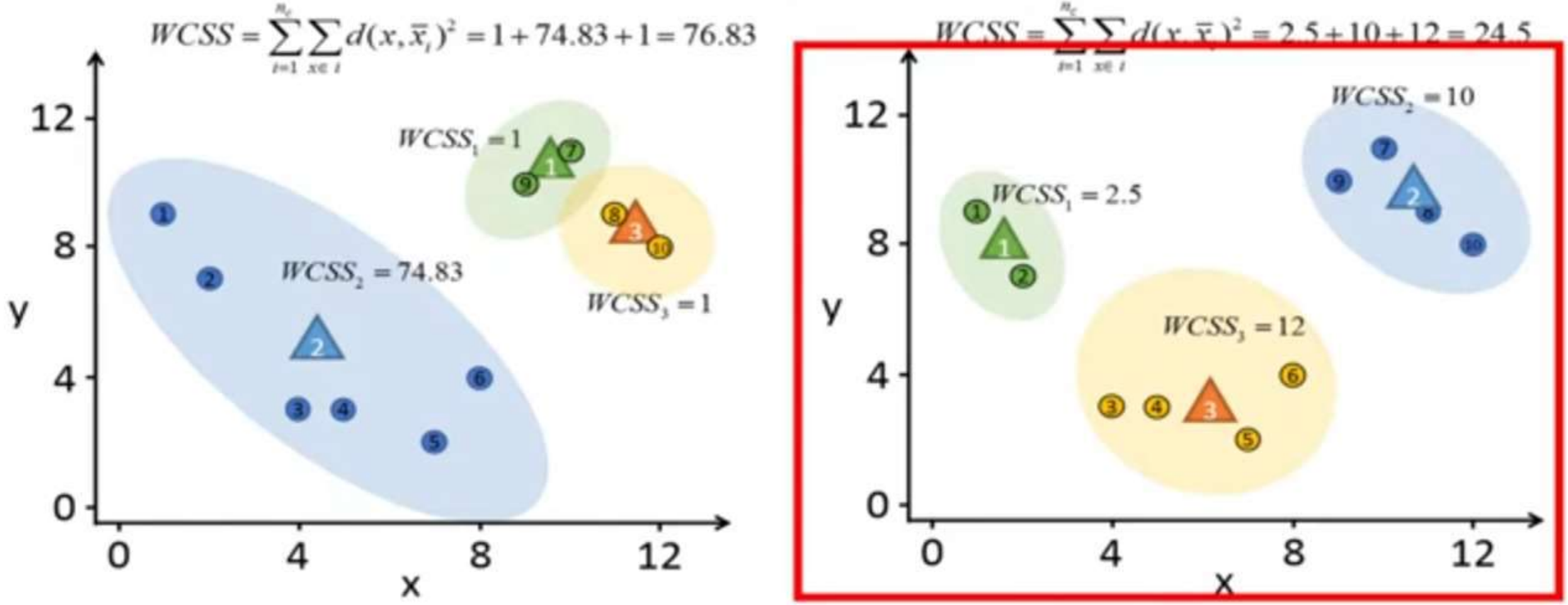


Remember that k-means clustering may result in different outputs depending on the initial location of the centroids. We will here use the total within cluster sum of squares to determine which of these two outputs that generate the "best" clusters.



We therefore calculate the total within cluster sum of squares of the two different outputs.

Initial positions of the centroids



we would select this clustering, which seems reasonable because the data points are generally closer to the centroids, and the clusters are well separated.

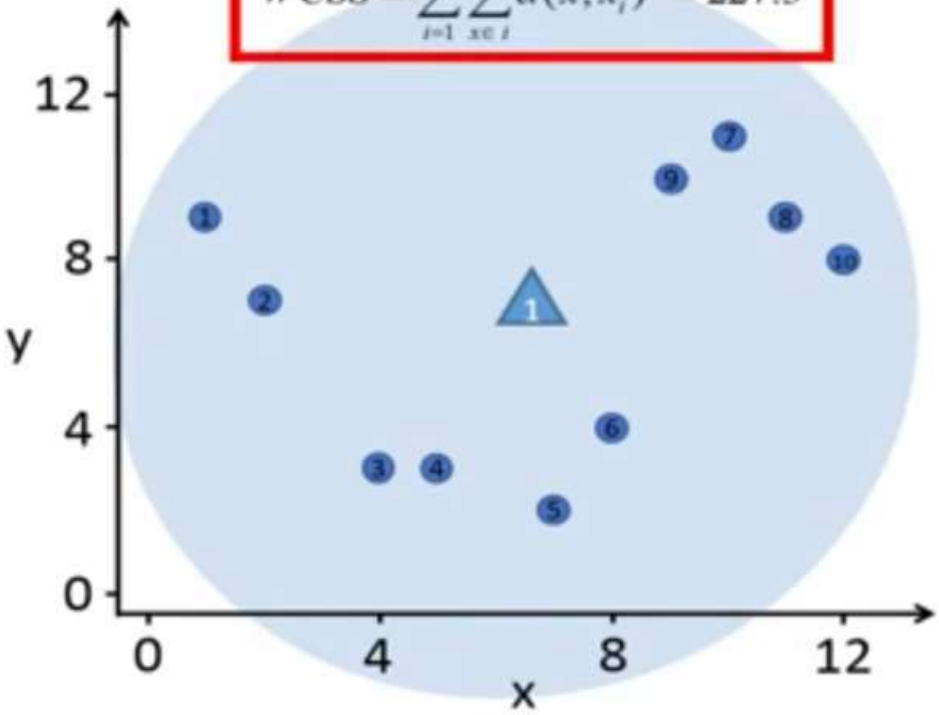
Try different values of *k*

It is therefore important that we try many different initial positions of the centroids and that we then select the clustering that results in the smallest within cluster sum of squares.

The same method can also be used when we evaluate different values of *k*.

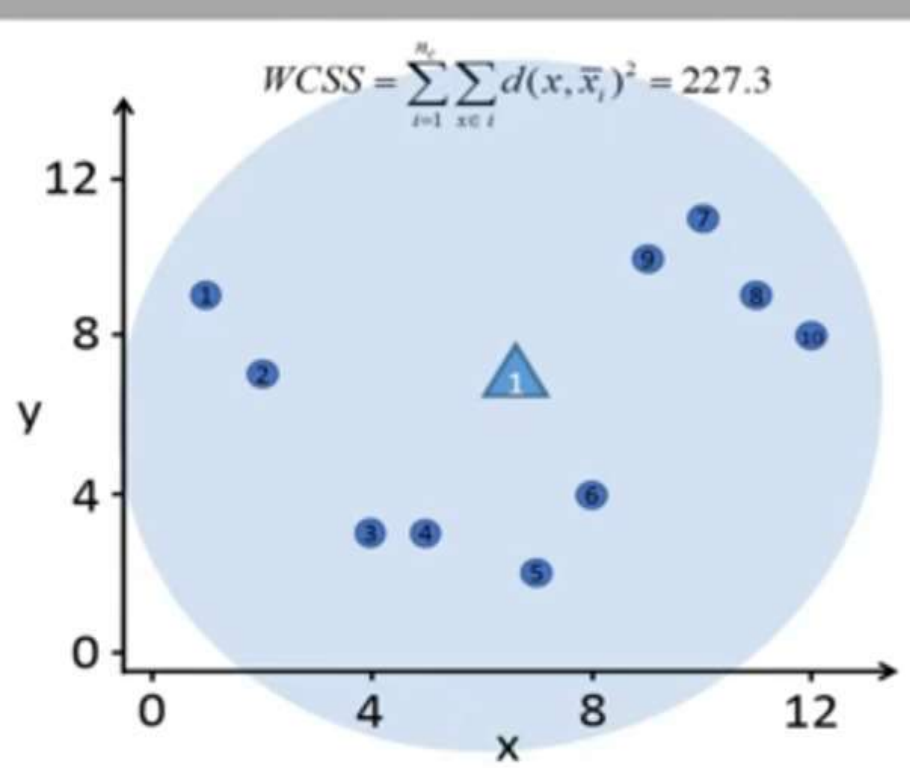
k=1

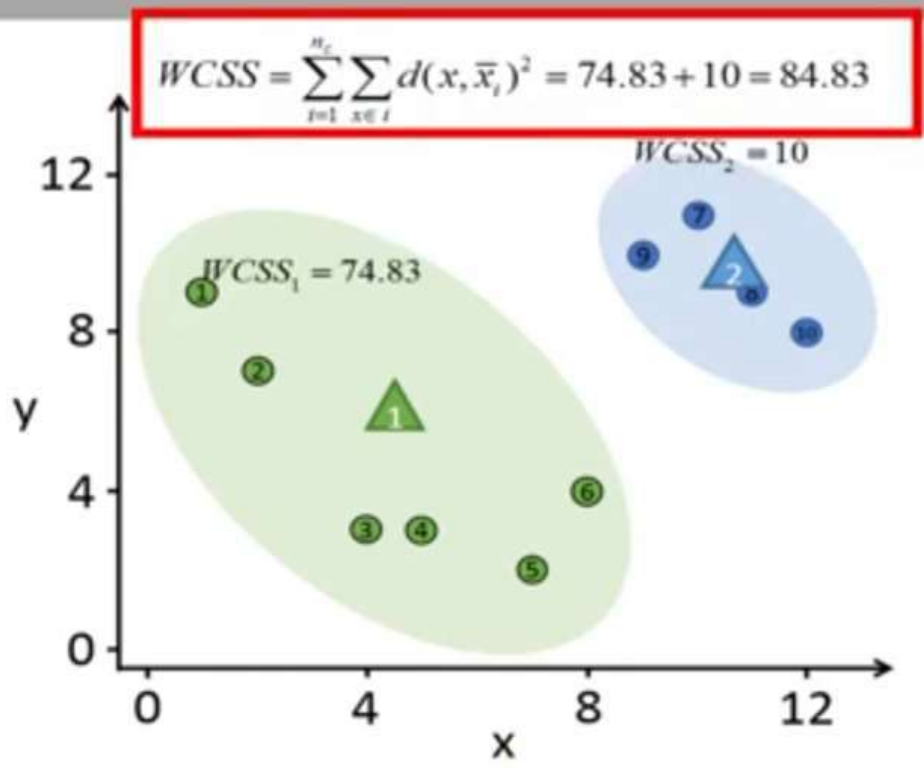
$$WCSS = \sum_{i=1}^{n_i} \sum_{x \in i} d(x, \bar{x}_i)^2 = 227.3$$



the within cluster sum of squares is equal to 227.3. Since we only have one cluster in this case, the between cluster sum of squares is equal to zero.

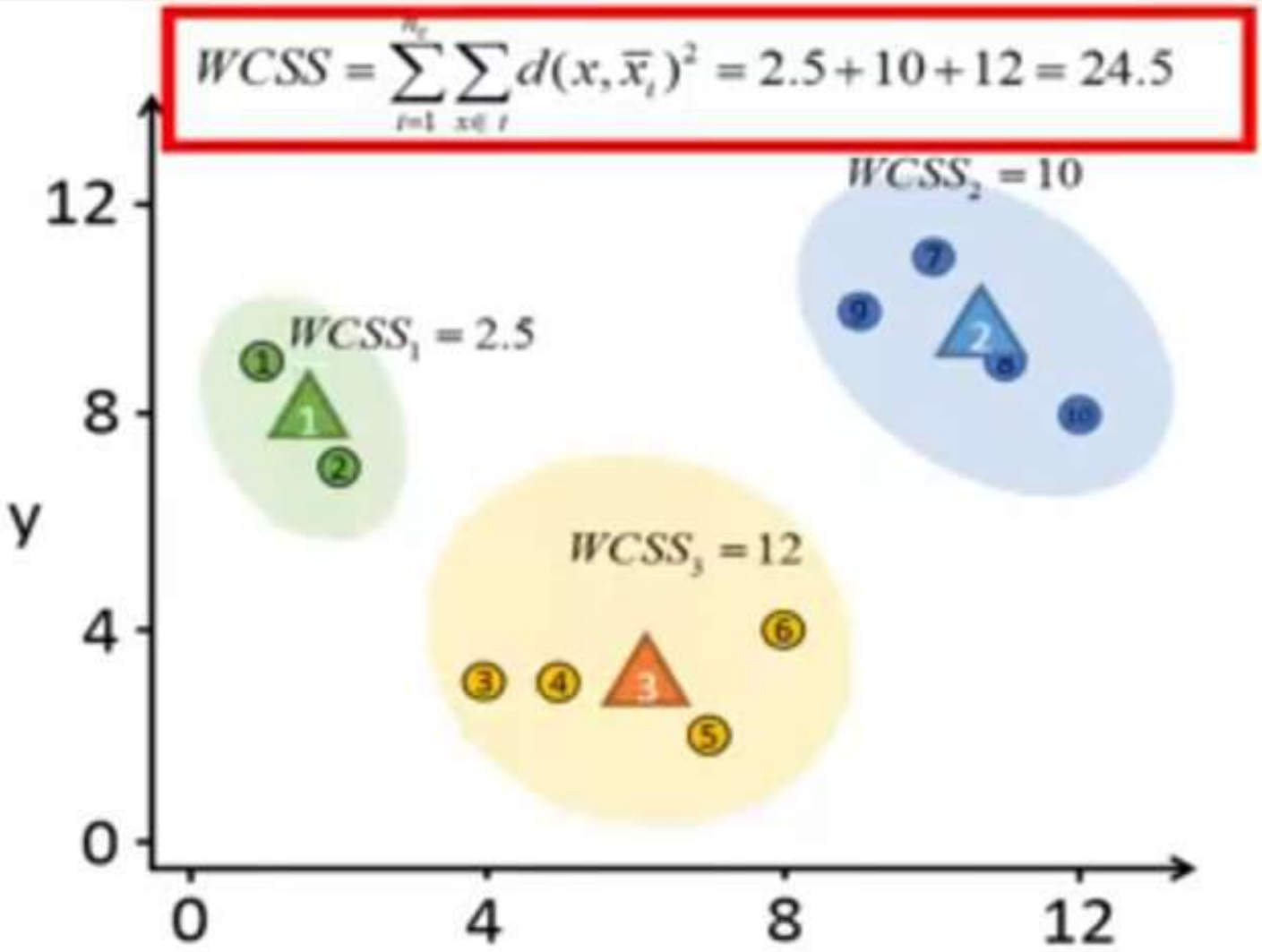
k=2





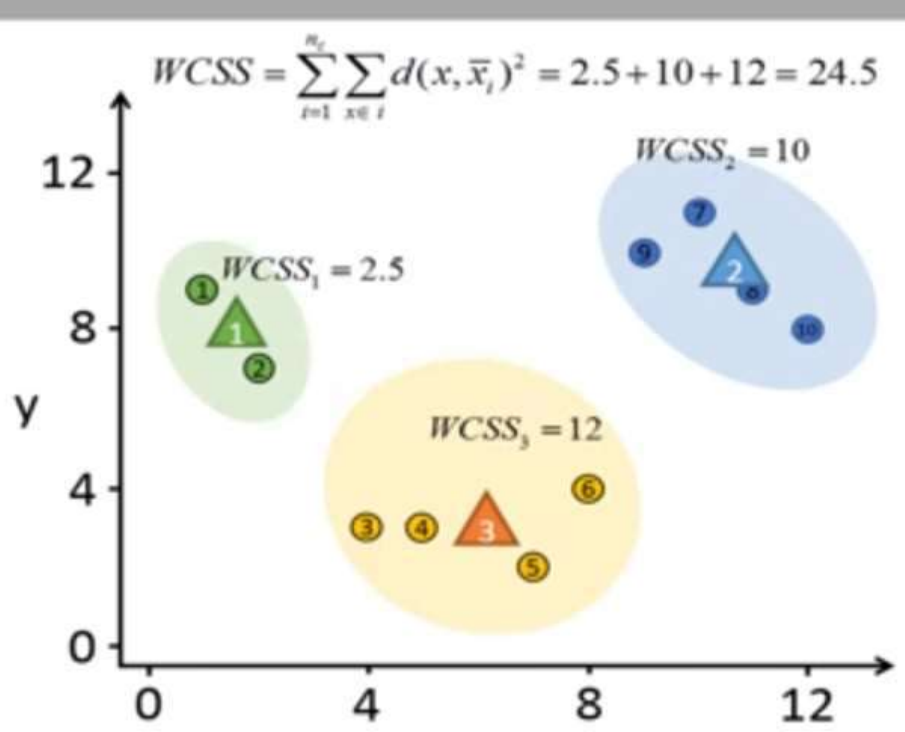
with a within cluster sum of squares that has decreased to 84.83 because the data points are now much closer to the centroids.

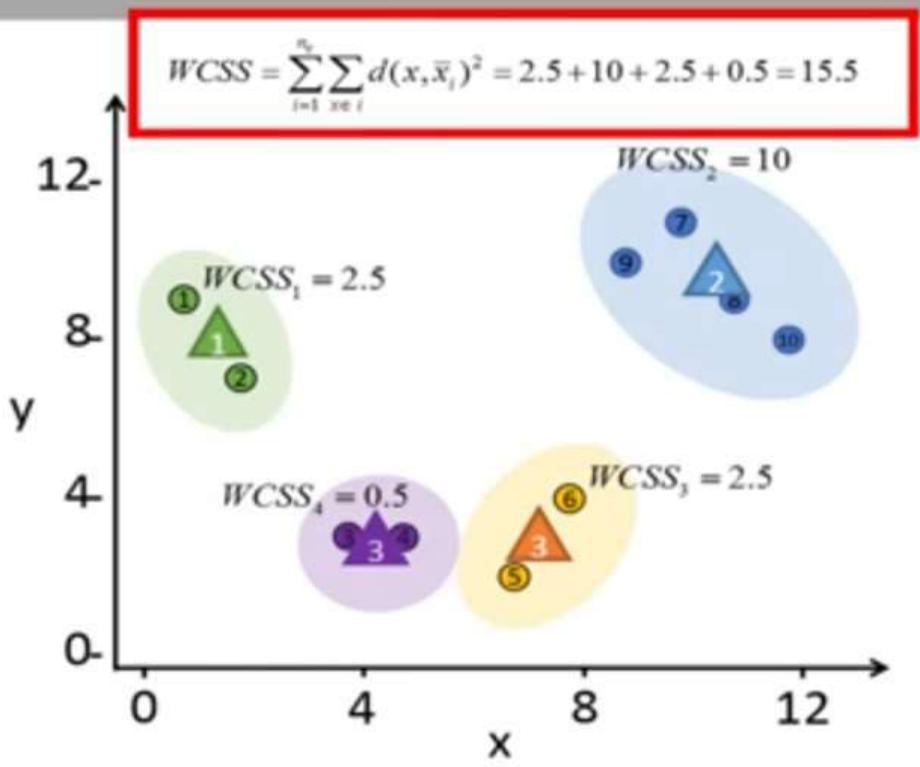
k=3



If we set *k* to three, the within cluster sum of squares has been reduced to 24.5,

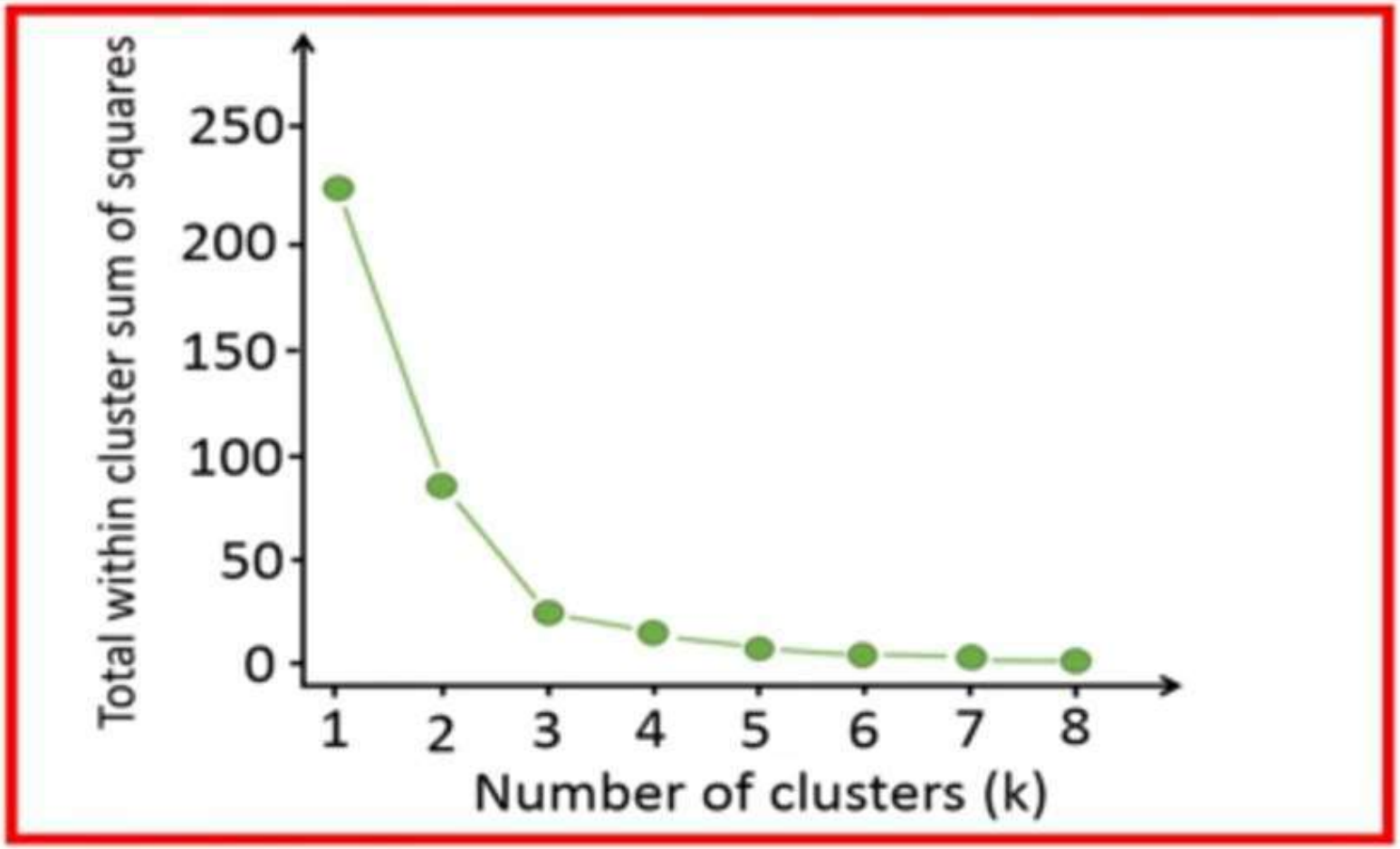
k=4





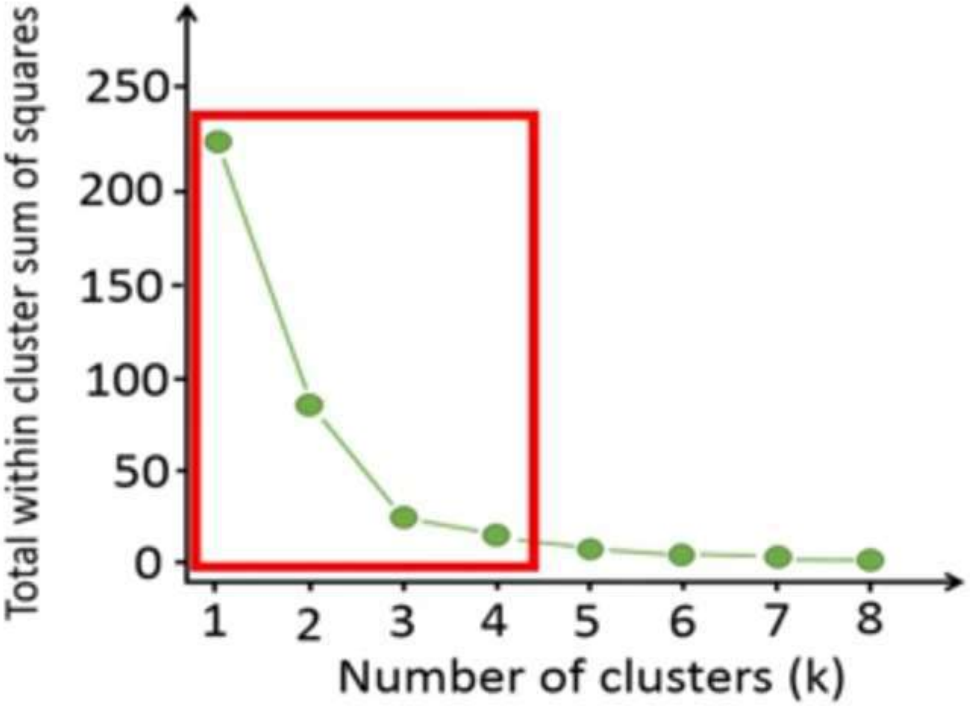
and if we set *k* to four, the within cluster sum of squares has been reduced to 15.5, and so forth.

The elbow method



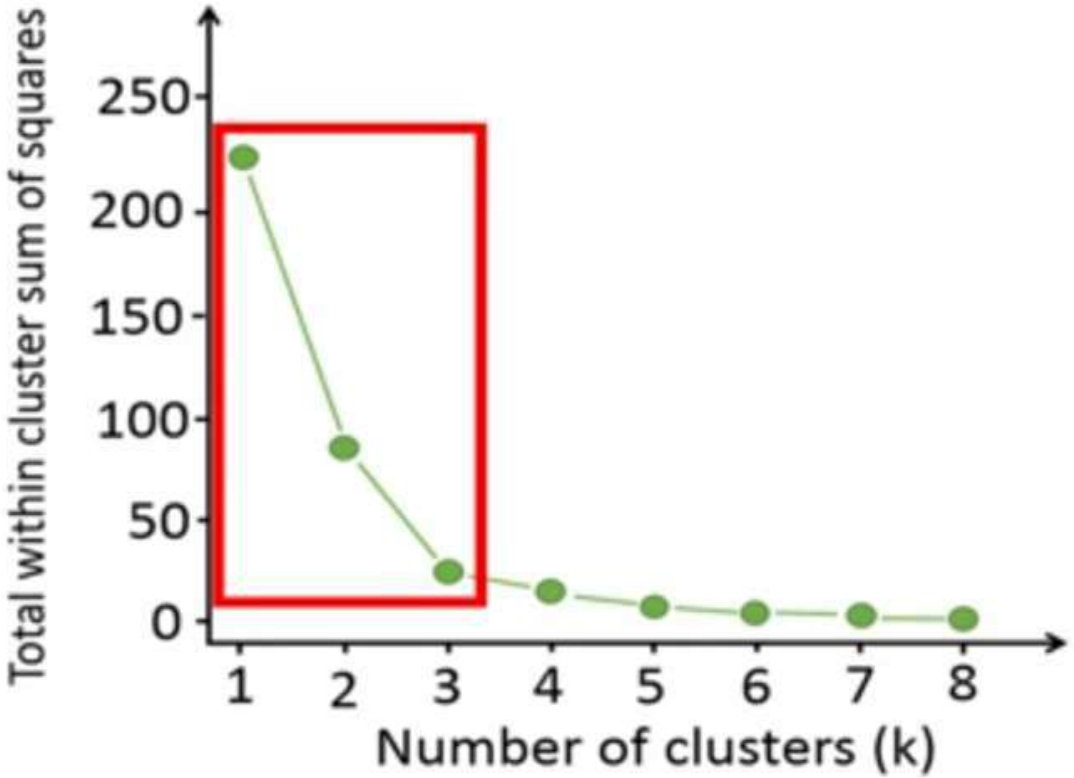
If we plot the total within cluster sum of squares for different values of *k*, we will get the following plot.

The elbow method



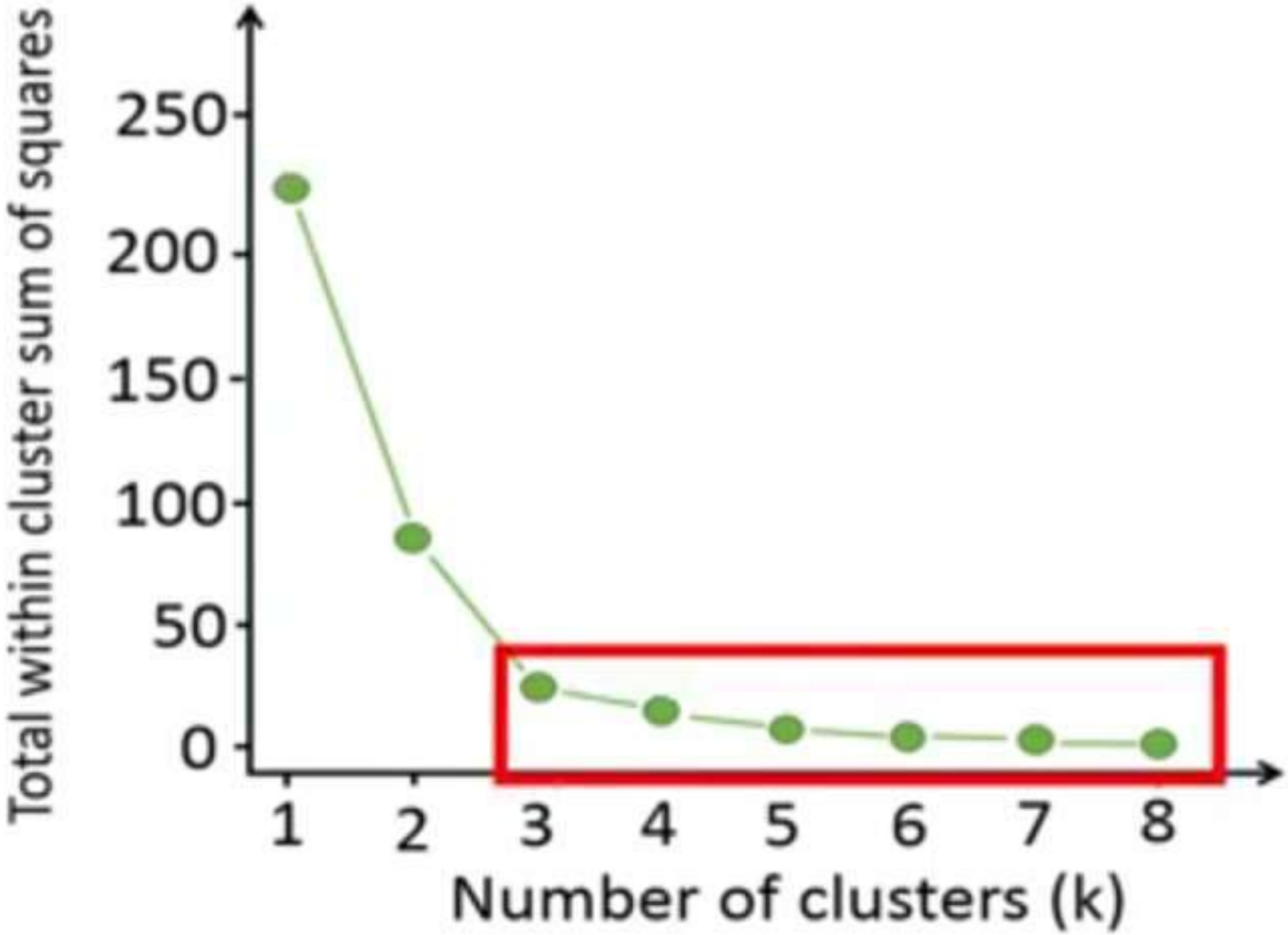
These points represent the total within cluster sum of squares from our previous examples when we tried four different values of *k*.

The elbow method



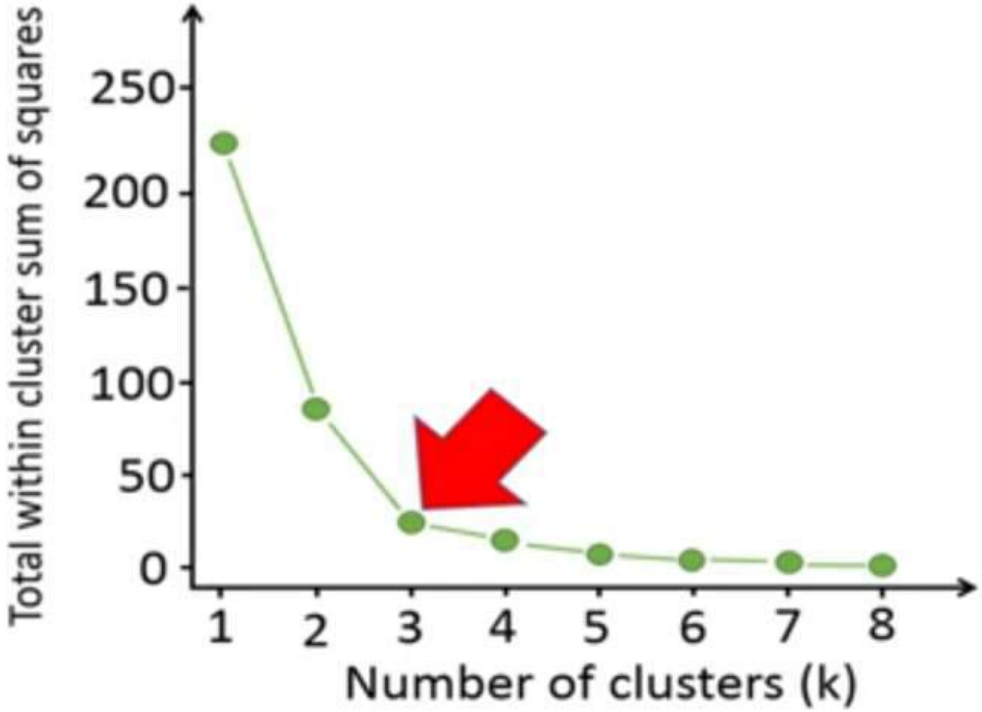
Note that there is a steep decrease in the total within cluster sum of squares when *k* is increased from one to three,

The elbow method



but much less when *k* is increased from three to eight.

The elbow method



The optimal value of *k* is the value at the "elbow" of the curve. The optimal number of clusters for our example data is therefore three. Finding the optimal value of *k* this way is called the elbow method.

k-means clustering

	x	y	u	v
1	1	9	5	3
2	2	7	4	4
3	4	3	2	5
4	5	3	5	6
5	7	2	2	7
6	8	4	2	12
7	10	11	4	4
8	11	9	6	3
9	9	10	4	2
10	12	8	9	4

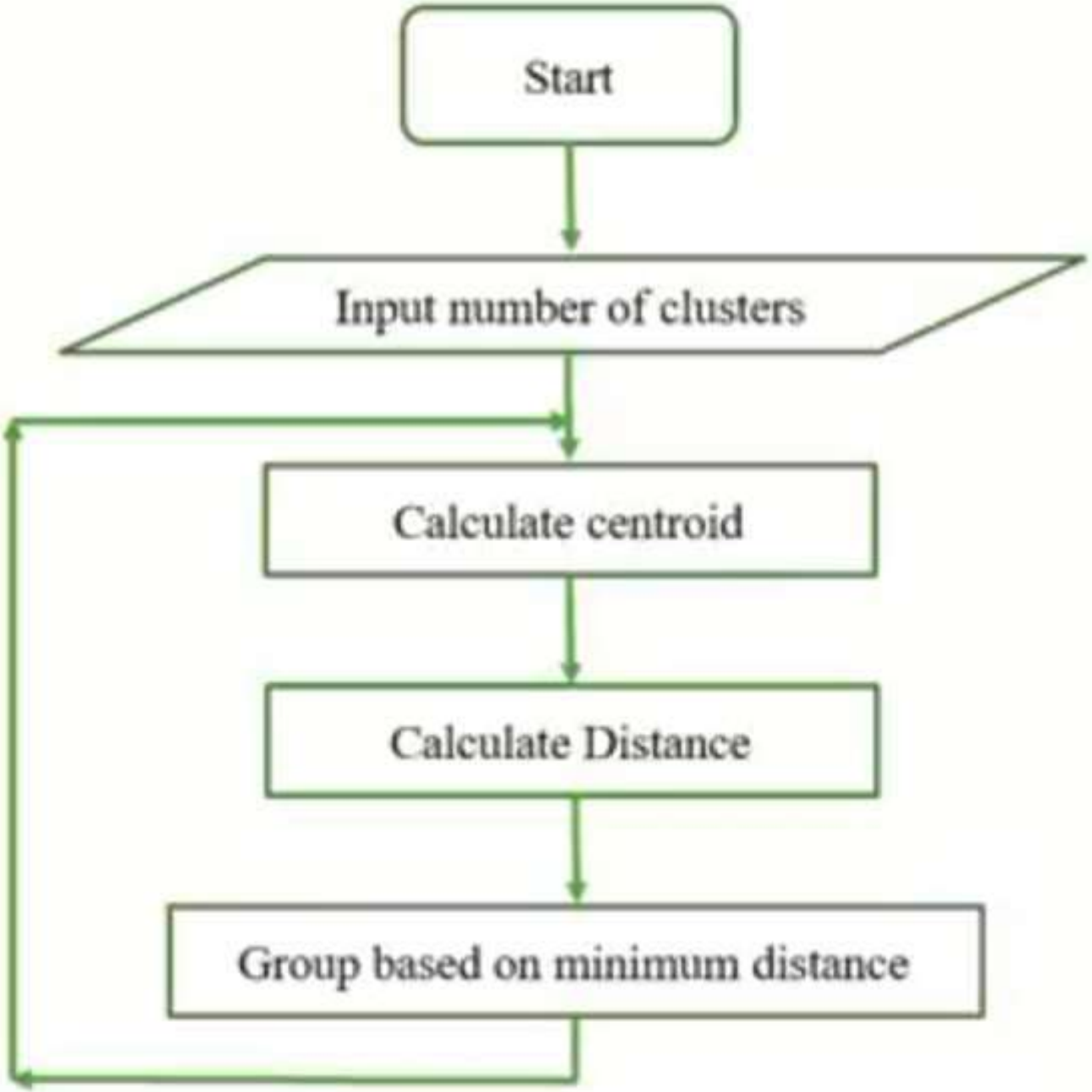
Imagine that we now instead would have four variables and would like to find three clusters based on all these four variables. It is now impossible to plot this in a simple way so that we can identify three clusters.

k-means clustering

	x	y	u	v
1	1	9	5	3
2	2	7	4	4
3	4	3	2	5
4	5	3	5	6
5	7	2	2	7
6	8	4	2	12
7	10	11	4	4
8	11	9	6	3
9	9	10	4	2
10	12	8	9	4

$$d = \sqrt{(x_1 - x_2)^2 + (y_1 - y_2)^2 + (u_1 - u_2)^2 + (v_1 - v_2)^2}$$

However, a computer can easily calculate the Euclidean distance in the four-dimensional space and determine which data points in this space that belong to each of the three clusters.


Start
Input number of clusters
Calculate centroid
Calculate Distance
Group based on minimum distance

- Apply K-Mean Clustering for the following data sets for two clusters. Tabulate all the assignments.

Sample No	X	Y
1	185	72
2	170	56
3	168	60
4	179	68
5	182	72
6	188	77

- Given k = 2

Initial Centroid

Cluster	X	Y
k1	185	72
k2	170	56

- Calculate Euclidean distance using the given equation.

$$\text{Distance } [(x,y), (a,b)] = \sqrt{(x-a)^2+(x-b)^2}$$

Initial Centroid

Cluster	X	Y
k1	185	72
k2	170	56

- Calculate Euclidean distance using the given equation.

$$\text{Distance } [(x,y), (a,b)] = \sqrt{(x-a)^2+(x-b)^2}$$

$$\text{Cluster 1 } (185,72) = \sqrt{(185-185)^2+(72-72)^2} = 0$$

$$\text{Distance from Cluster 2} = \sqrt{(170-185)^2+(56-72)^2}$$

$$(170,56) = \sqrt{(-15)^2+(-16)^2}$$

$$= \sqrt{255+256}$$

$$= \sqrt{481}$$

$$= 21.93$$

Calculate Euclidean distance using the given equation.

$$\text{Distance } [(x,y), (a,b)] = \sqrt{(x-a)^2+(x-b)^2}$$

$$\text{Distance from Cluster 1} = \sqrt{(185-170)^2+(72-56)^2}$$

(185,72)

$$= \sqrt{(15)^2+(16)^2}$$

$$= \sqrt{255+256}$$

$$= \sqrt{481}$$

$$= 21.93$$

$$\text{Cluster 2 } (170,56) = \sqrt{(170-170)^2+(56-56)^2} = 0$$

Cluster	Centroid		
	X	Y	ASSIGNMENT
k1	0	21.93	1
k2	21.93	0	2

- Calculate Euclidean distance for the next dataset (168,60)

$$\text{Distance } [(x,y), (a,b)] = \sqrt{(x-a)^2+(x-b)^2}$$

$$\text{Distance from Cluster 1} = \sqrt{(168-185)^2+(60-72)^2}$$

(185,72)

$$= \sqrt{(-17)^2+(-12)^2}$$

$$= \sqrt{283+144}$$

$$= \sqrt{433}$$

$$= 20.808$$

- Calculate Euclidean distance for the next dataset (168,60)

$$\text{Distance } [(x,y), (a,b)] = \sqrt{(x-a)^2+(x-b)^2}$$

$$\text{Distance from Cluster 2} = \sqrt{(168-170)^2+(60-56)^2}$$

$$(170,56) \qquad = \sqrt{(-2)^2+(-4)^2}$$

$$= \sqrt{4+16}$$

$$= \sqrt{20}$$

$$= 4.472$$

Dataset	Euclidean Distance		
	Cluster 1	Cluster 2	ASSIGNMENT
(168,60)	20.808	4.472	2

- Update the cluster centroid.

Cluster	X	Y
k1	185	72
k2	= (170 + 168)/ 2 = 169	= (60+56)/ 2 = 58

- Calculate Euclidean distance for the next dataset (179,68)

$$\text{Distance } [(x,y), (a,b)] = \sqrt{(x-a)^2+(x-b)^2}$$

$$\text{Distance from Cluster 1} = \sqrt{(179-185)^2+(68-72)^2}$$

$$(185,72) \qquad = \sqrt{(-6)^2+(-4)^2}$$

$$= \sqrt{36+16}$$

$$= \sqrt{52}$$

$$= 7.211103$$

- Calculate Euclidean distance for the next dataset (179,68)

$$\text{Distance } [(x,y), (a,b)] = \sqrt{(x-a)^2+(x-b)^2}$$

$$\text{Distance from Cluster 2} = \sqrt{(179-169)^2+(68-58)^2}$$

(169,58)

$$= \sqrt{(10)^2+(10)^2}$$

$$= \sqrt{100+100}$$

$$= \sqrt{200}$$

$$= 14.14214$$

Dataset	Euclidean Distance		
	Cluster 1	Cluster 2	ASSIGNMENT
(179,68)	7.211103	14.14214	1

- Update the cluster centroid.

Cluster	X	Y
k1	= 185+179/2 =182	= 72+68/2 =70
k2	169	58

- Calculate Euclidean distance for the next dataset (182,72)

$$\text{Distance } [(x,y), (a,b)] = \sqrt{(x-a)^2+(x-b)^2}$$

$$\text{Distance from Cluster 1} = \sqrt{(182-182)^2+(72-70)^2}$$

$$(182,70) \qquad = \sqrt{(0)^2+(2)^2}$$

$$= \sqrt{0+4}$$

$$= \sqrt{4}$$

$$= 2$$

- Calculate Euclidean distance for the next dataset (182,72)

$$\text{Distance } [(x,y), (a,b)] = \sqrt{(x-a)^2+(x-b)^2}$$

$$\text{Distance from Cluster 2} = \sqrt{(182-169)^2+(72-58)^2}$$

(169,58)

$$= \sqrt{(13)^2+(14)^2}$$

$$= \sqrt{169+196}$$

$$= \sqrt{365}$$

$$= 19.10$$

Dataset	Euclidean Distance		
	Cluster 1	Cluster 2	ASSIGNMENT
(182,72)	2	19.10	1

- Update the cluster centroid.

Cluster	X	Y
k1	= 182+182/2 =182	= 70+72/2 = 71
k2	169	58

- Calculate Euclidean distance for the next dataset (188,77)

$$\text{Distance } [(x,y), (a,b)] = \sqrt{(x-a)^2+(x-b)^2}$$

$$\text{Distance from Cluster 1} = \sqrt{(188-182)^2+(77-71)^2}$$

$$(182,71) \qquad = \sqrt{(6)^2+(6)^2}$$

$$= \sqrt{36+36}$$

$$= \sqrt{72}$$

$$= 8.4852$$

- Calculate Euclidean distance for the next dataset (188,77)

$$\text{Distance } [(x,y), (a,b)] = \sqrt{(x-a)^2+(x-b)^2}$$

$$\text{Distance from Cluster 2} = \sqrt{(188-169)^2+(77-58)^2}$$

$$(169,58) \qquad = \sqrt{(19)^2+(19)^2}$$

$$= \sqrt{361+361}$$

$$= \sqrt{722}$$

$$= 26.87$$

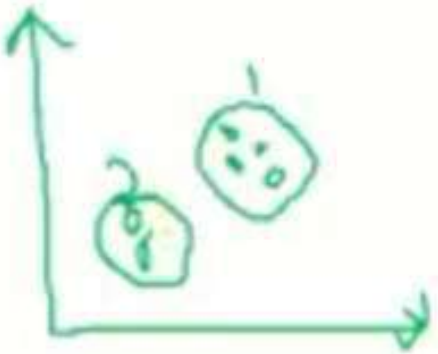
Dataset	Euclidean Distance		
	Cluster 1	Cluster 2	ASSIGNMENT
(188,77)	8.4852	26.87	1

- Updated cluster centroid.

Cluster	X	Y
k1	= 182+188/2 = 185	= 71+77/2 = 74
k2	169	58

Final Assignment

Dataset No	X	Y	Assignment
1	185	72	1
2	170	56	2
3	168	60	2
4	179	68	1
5	182	72	1
6	188	77	1



Final Assignment

Dataset No	X	Y	Assignment
1	185	72	1
2	170	56	2
3	168	60	2
4	179	68	1
5	182	72	1
6	188	77	1

Strengths of k-means

- Strengths:
  - Simple: easy to understand and to implement
  - Efficient: Time complexity: $O(tkn)$,

    where n is the number of data points,

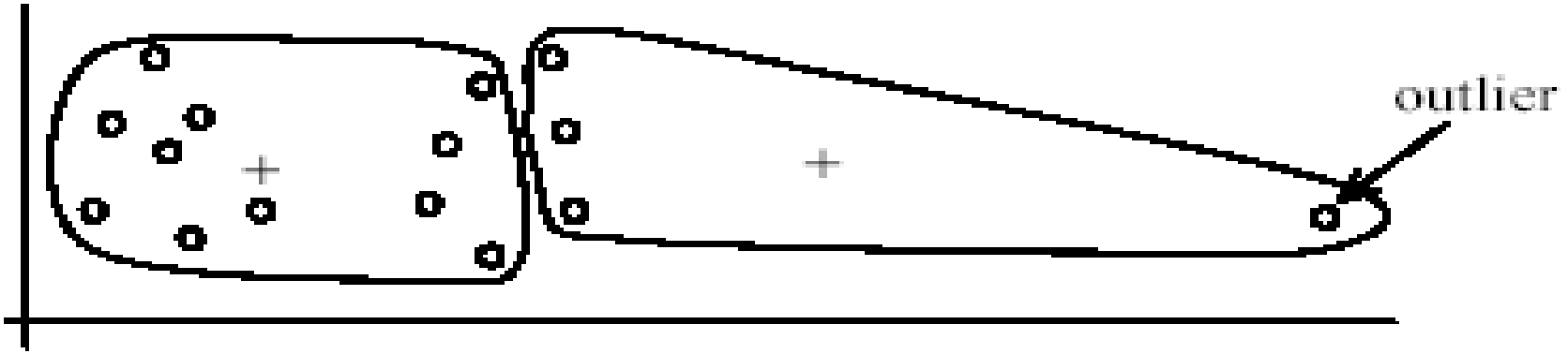
    k is the number of clusters, and

    t is the number of iterations.
  - Since both k and t are small. k-means is considered a linear algorithm.
- K-means is the most popular clustering algorithm.
- Note that: it terminates at a local optimum if SSE is used. The global optimum is hard to find due to complexity.

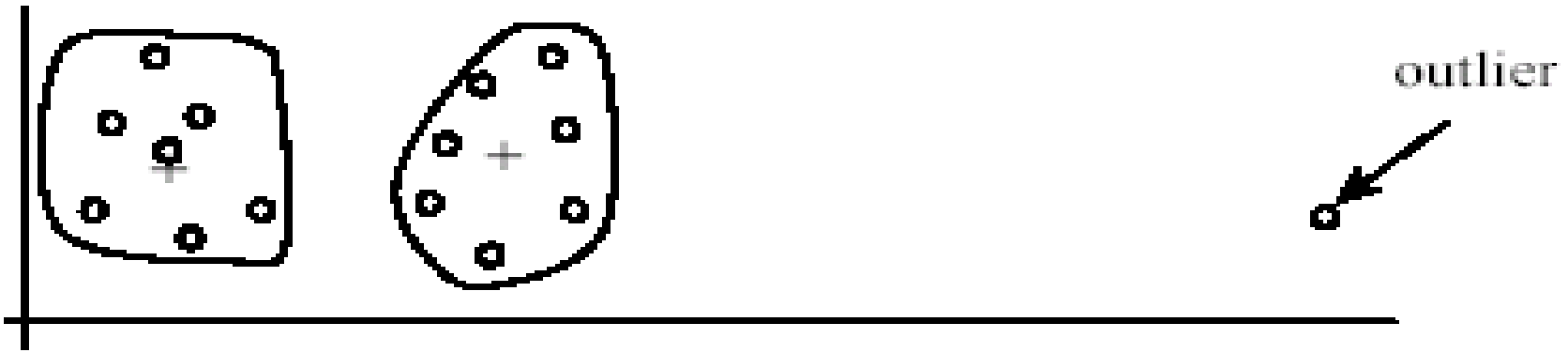
Weaknesses of k-means

- The algorithm is only applicable if the mean is defined.
  - For categorical data, *k*-mode - the centroid is represented by most frequent values.
- The user needs to specify *k*.
- The algorithm is sensitive to **outliers**
  - Outliers are data points that are very far away from other data points.
  - Outliers could be errors in the data recording or some special data points with very different values.

Weaknesses of k-means:Problems with outliers



(A): Undesirable clusters



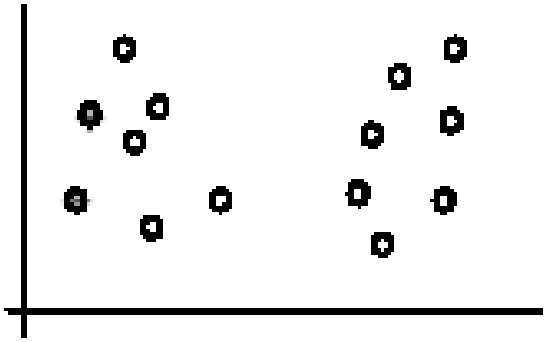
(B): Ideal clusters

Weaknesses of k-means: To deal with outliers

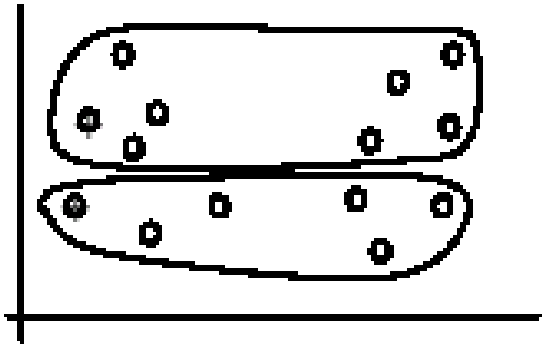
- One method is to remove some data points in the clustering process that are much further away from the centroids than other data points.
  - To be safe, we may want to monitor these possible outliers over a few iterations and then decide to remove them.
- Another method is to perform random sampling. Since in sampling we only choose a small subset of the data points, the chance of selecting an outlier is very small.
  - Assign the rest of the data points to the clusters by distance or similarity comparison, or classification

Weaknesses of k-means (cont …)

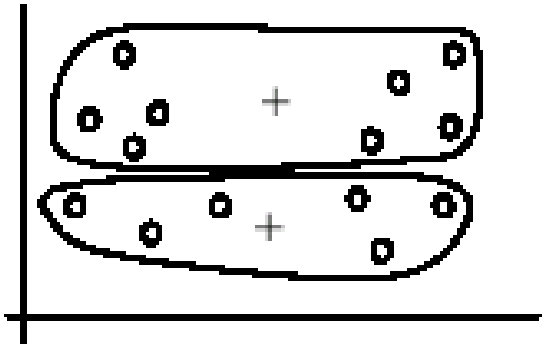
- The algorithm is sensitive to initial seeds.



(A). Random selection of seeds (centroids)



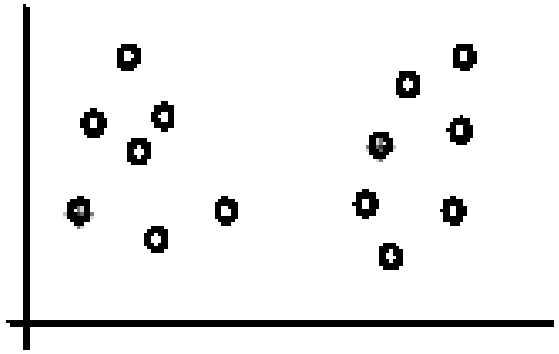
(B). Iteration 1



(C). Iteration 2

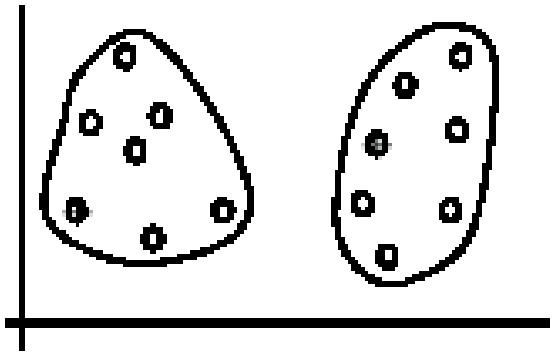
Weaknesses of k-means (cont …)

- If we use different seeds: good results

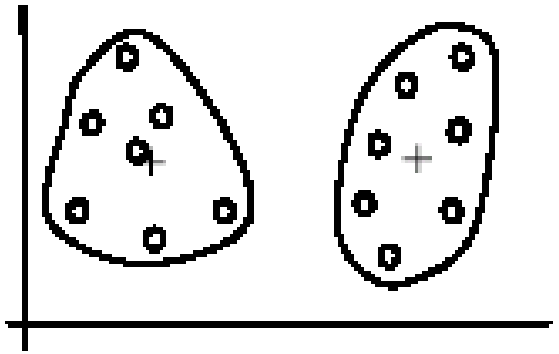


There are some methods to help choose good seeds

(A). Random selection of k seeds (centroids)



(B). Iteration 1



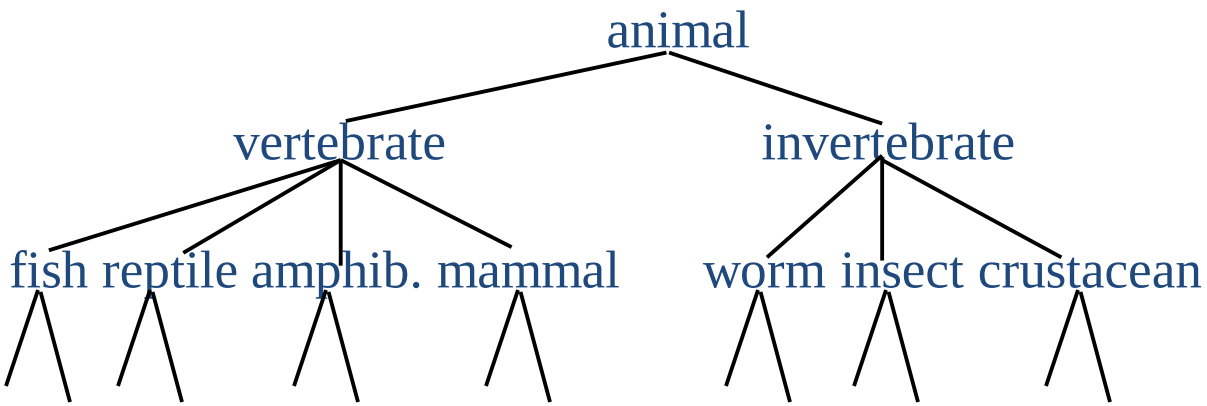
(C). Iteration 2

K-means summary

- Despite weaknesses, *k*-means is still the most popular algorithm due to its simplicity, efficiency and
  - other clustering algorithms have their own lists of weaknesses.
- No clear evidence that any other clustering algorithm performs better in general
  - although they may be more suitable for some specific types of data or applications.
- Comparing different clustering algorithms is a difficult task. No one knows the correct clusters!

Hierarchical Clustering

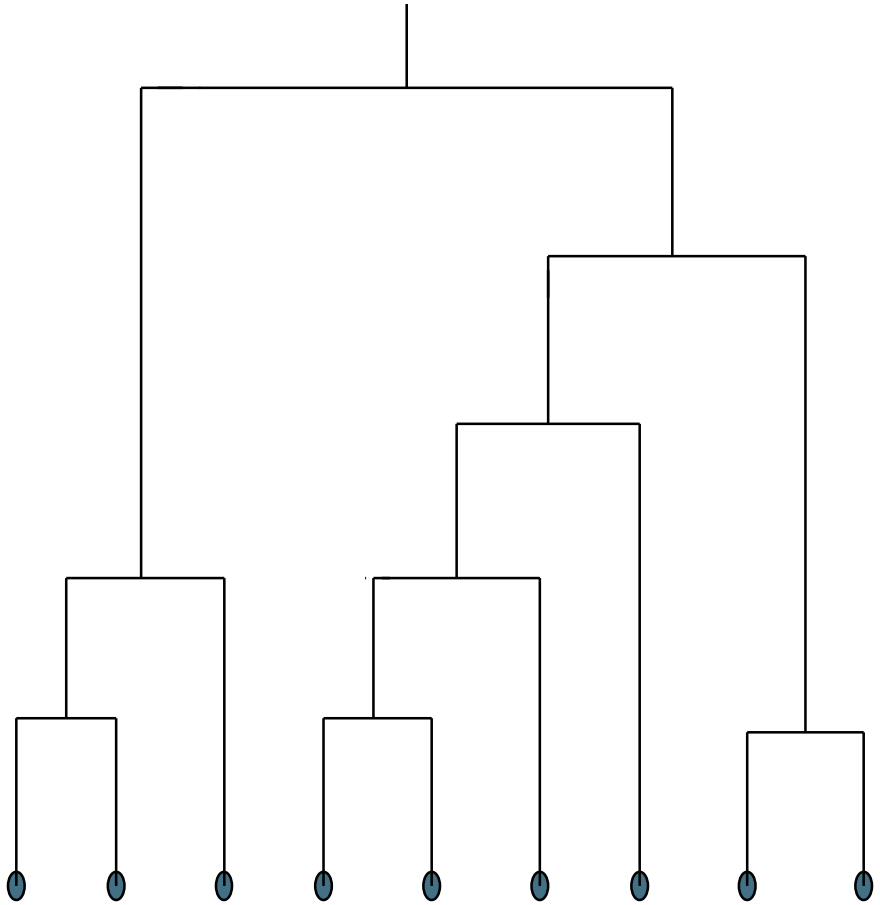
- Build a tree-based hierarchical taxonomy (*dendrogram*) from a set of documents.



- One approach: recursive application of a partitional clustering algorithm.

Dendrogram: Hierarchical Clustering

- Clustering obtained by cutting the dendrogram at a desired level: each connected component forms a cluster.



Hierarchical Agglomerative Clustering (HAC)

- Starts with each doc in a separate cluster
  - then repeatedly joins the *closest pair* of clusters, until there is only one cluster.
- The history of merging forms a binary tree or hierarchy.

Note: the resulting clusters are still "hard" and induce a partition

Closest pair of clusters

- Many variants to defining closest pair of clusters
- **Single-link**
  - Similarity of the *most* cosine-similar (single-link)
- **Complete-link**
  - Similarity of the "furthest" points, the *least* cosine-similar
- **Centroid**
  - Clusters whose centroids (centers of gravity) are the most cosine-similar
- **Average-link**
  - Average cosine between pairs of elements

Single Link Agglomerative Clustering

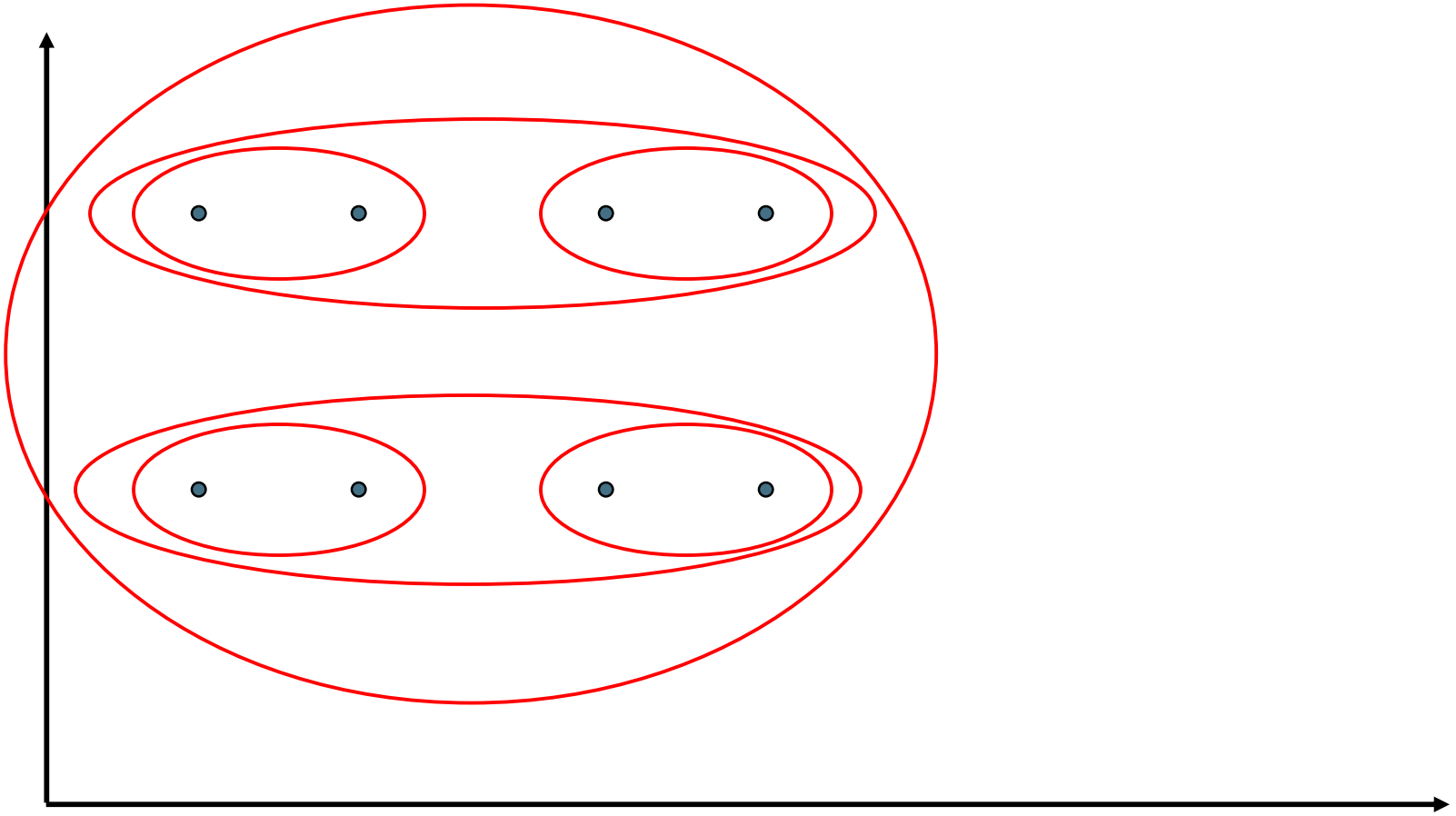
- Use maximum similarity of pairs:

$$sim(c_i, c_j) = \max_{x \in c_i, y \in c_j} sim(x, y)$$

- Can result in "straggly" (long and thin) clusters due to chaining effect.
- After merging c_i and c_j, the similarity of the resulting cluster to another cluster, c_k, is:

$$sim((c_i \cup c_j), c_k) = \max(sim(c_i, c_k), sim(c_j, c_k))$$

Single Link Example



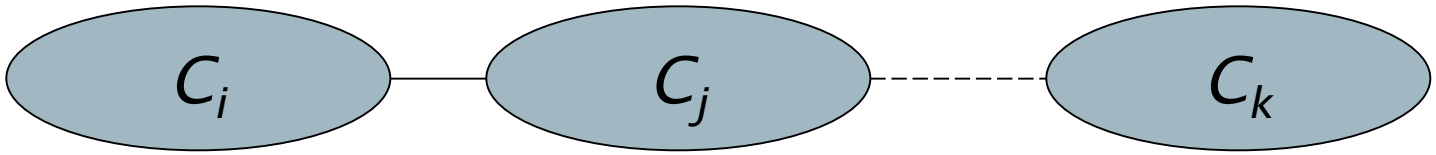
Complete Link

- Use minimum similarity of pairs:

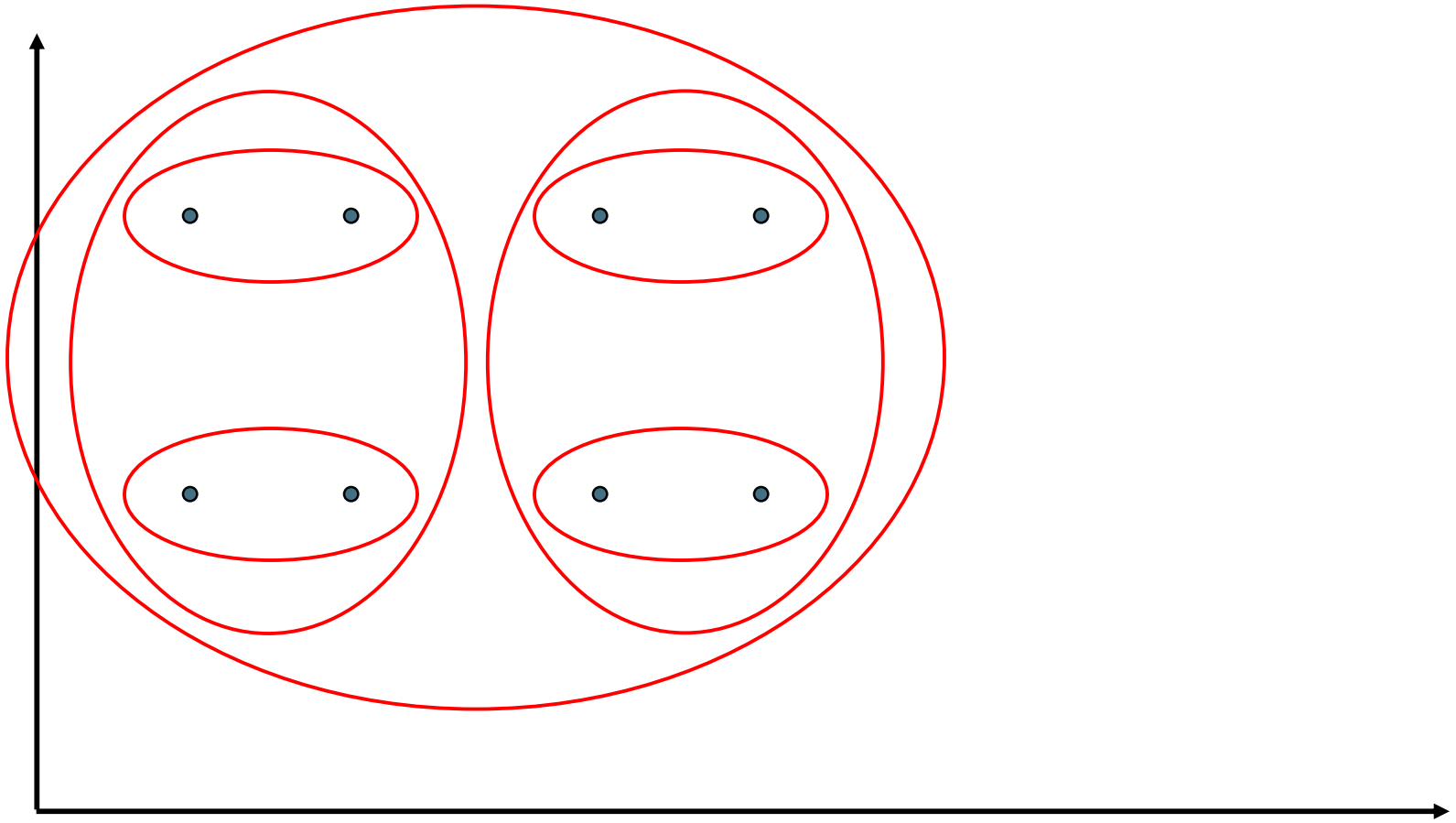
$$sim(c_i, c_j) = \min_{x \in c_i, y \in c_j} sim(x, y)$$

- Makes "tighter," spherical clusters that are typically preferable.
- After merging c_i and c_j, the similarity of the resulting cluster to another cluster, c_k, is:

$$sim((c_i \cup c_j), c_k) = \min(sim(c_i, c_k), sim(c_j, c_k))$$



Complete Link Example



Computational Complexity

- In the first iteration, all HAC methods need to compute similarity of all pairs of N initial instances, which is $O(N^2)$.
- In each of the subsequent $N-2$ merging iterations, compute the distance between the most recently created cluster and all other existing clusters.
- In order to maintain an overall $O(N^2)$ performance, computing similarity to each other cluster must be done in constant time.
  - Often $O(N^3)$ if done naively or $O(N^2 \log N)$ if done more cleverly

Hierarchical Clustering

- Starts with one cluster , individual item in its own cluster and iteratively merge clusters until all the items belong to one cluster.
- Bottom up approach is followed to merge the clusters together.
- Dendrograms are pictorially used to represent the HAC.

Hierarchical Clustering

- Single-nearest distance or single linkage.
- Complete-farthest distance or complete linkage.
- Average-average distance or average linkage.

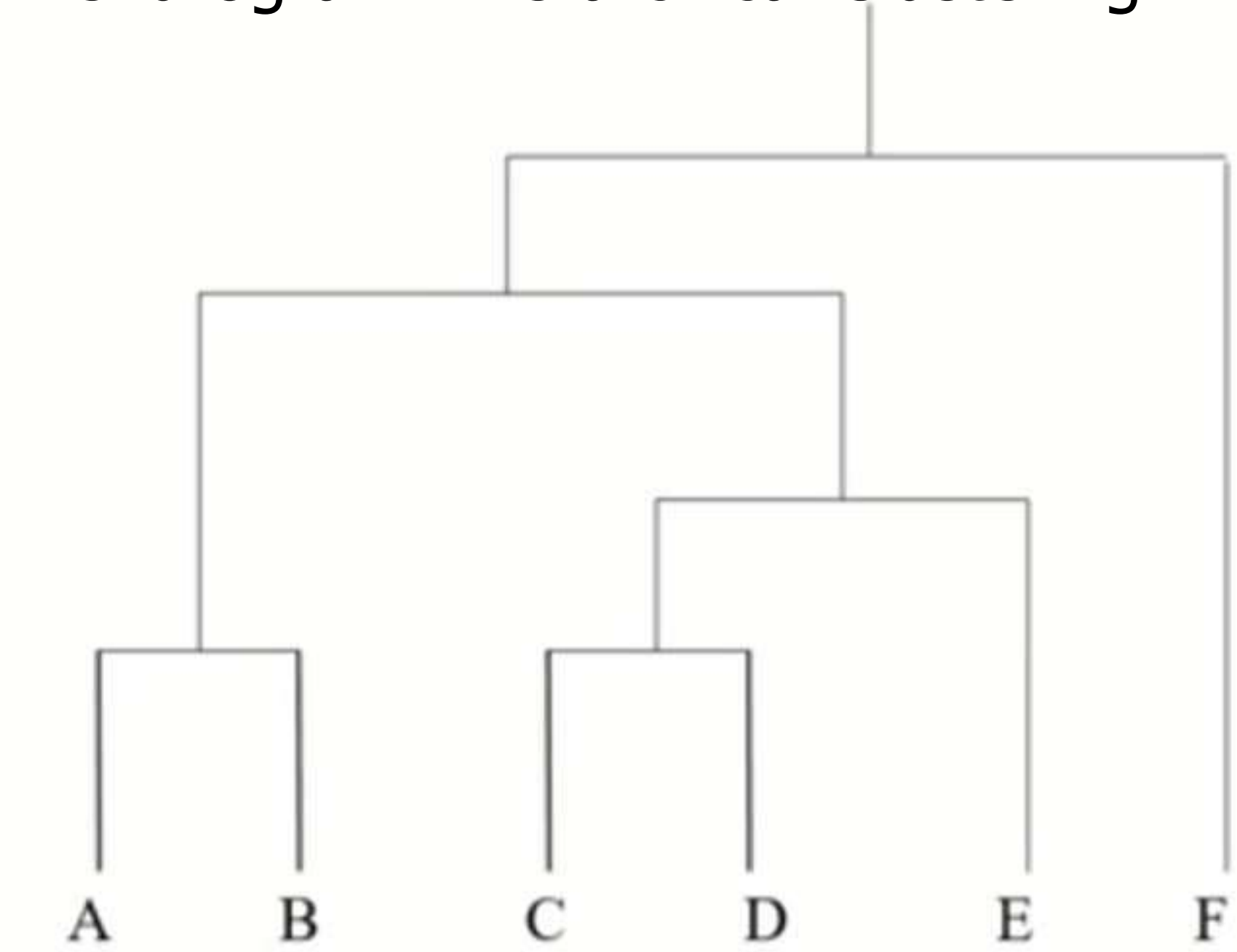
Hierarchical Clustering

Single Linkage	This is the distance between the closest members of the two clusters.
Complete Linkage	This is the distance between the members that are farthest apart.
Average Linkage	This method involves looking at the distances between all pairs and averages all of these distances. This is also called Unweighted Pair Group Mean Averaging.

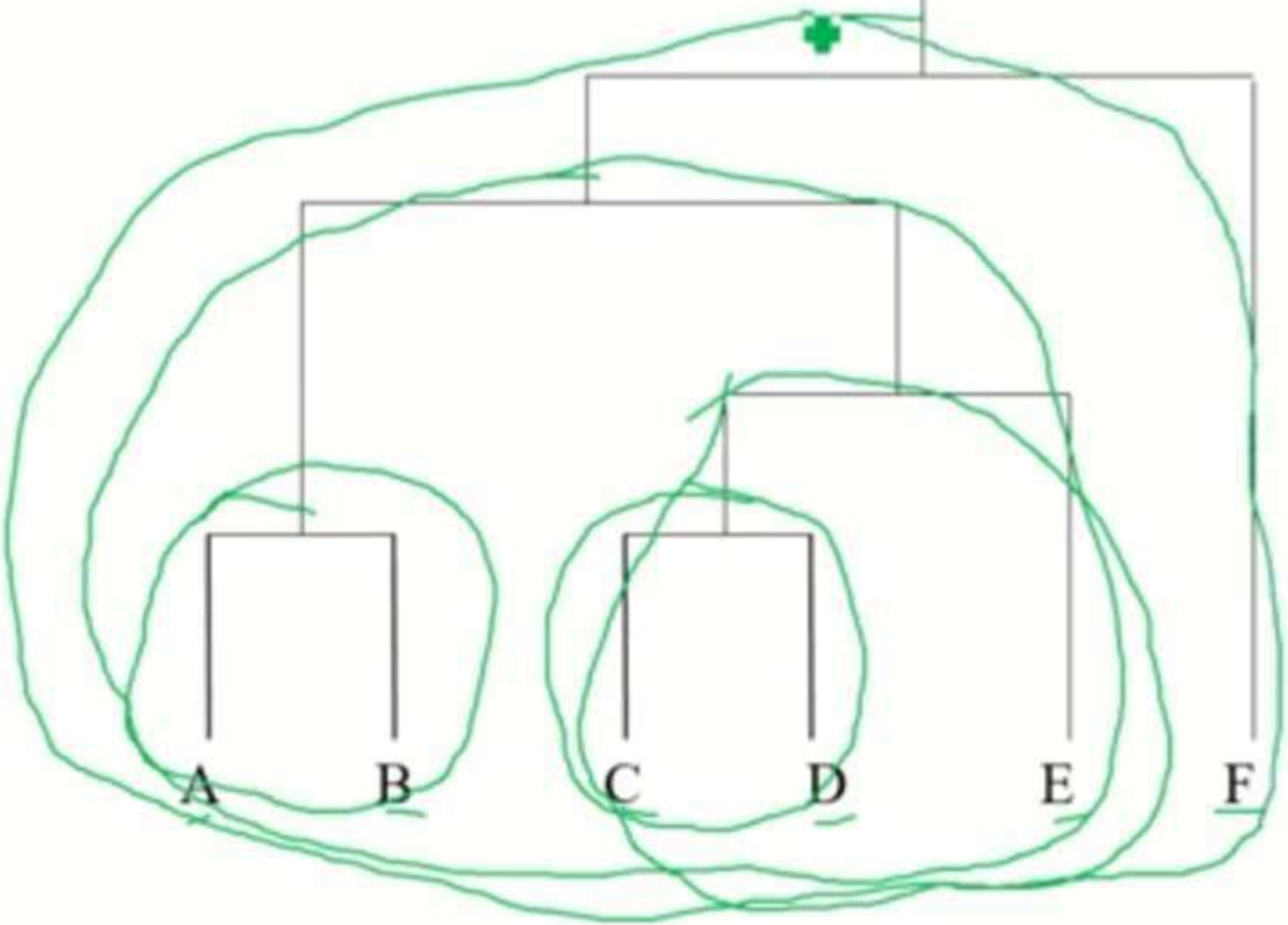
Dendrogram: Hierarchical Clustering

- A tree like structure which represents hierarchical technique.
  - ✓ Leaf- Individual.
  - ✓ Root – One cluster.
- A cluster at level 1, is the merger of its child cluster at level i + 1.

Dendrogram: Hierarchical Clustering



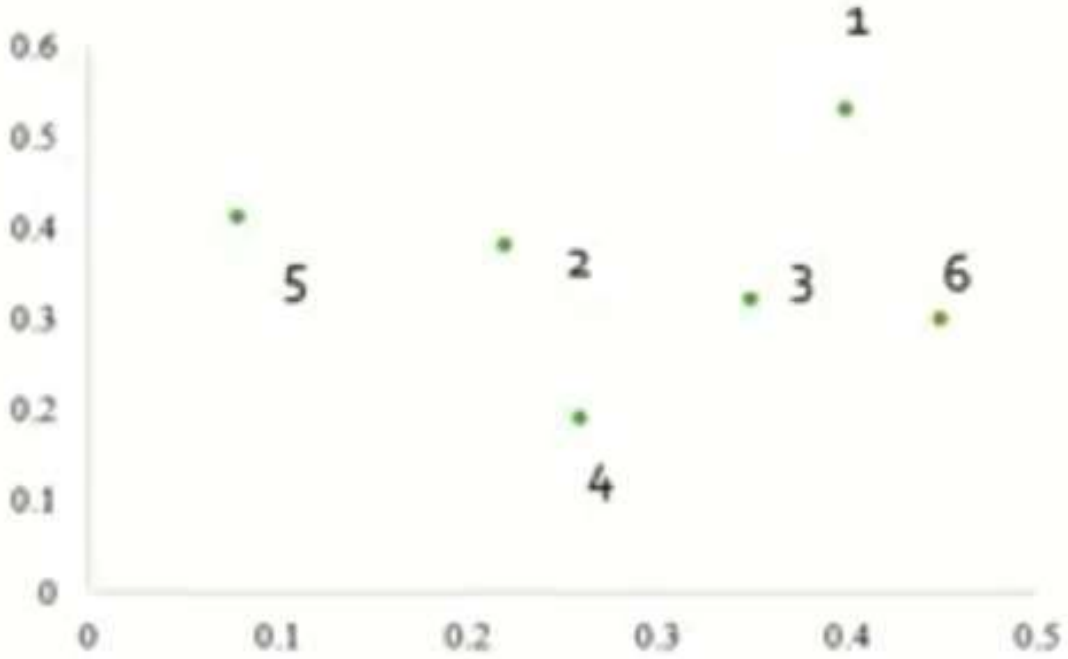
Dendrogram: Hierarchical Clustering



- Find the clusters using complete link technique. Use Euclidean distance, and draw the dendrogram.

	X	Y
P1	0.40	0.53
P2	0.22	0.38
P3	0.35	0.32
P4	0.26	0.19
P5	0.08	0.41
P6	0.45	0.30

	X	Y
P1	0.40	0.53
P2	0.22	0.38
P3	0.35	0.32
P4	0.26	0.19
P5	0.08	0.41
P6	0.45	0.30



- Calculate Euclidean distance, create the distance matrix.

$$\text{Distance } [(x,y), (a,b)] = \sqrt{(x-a)^2+(x-b)^2}$$

$$\text{Distance (P1,P2)} = \sqrt{(0.40-0.22)^2+(0.53-0.38)^2}$$

$$\text{(0.40,0.53), (0.22,0.38)} = \sqrt{(0.18)^2+(0.15)^2}$$

$$= \sqrt{0.0324 + 0.0225}$$

$$= \sqrt{0.0549}$$

$$= 0.23$$

- The distance matrix is

	P1	P2	P3	P4	P5	P6
P1	0					
P2	0.23	0				
P3	0.22	0.15	0			
P4	0.37	0.20	0.15	0		
P5	0.34	0.14	0.28	0.29	0	
P6	0.23	0.25	0.11	0.22	0.39	0

- The distance matrix is

	P1	P2	P3	P4	P5	P6
P1	0					
P2	0.23	0				
P3	0.22	0.15	0			
P4	0.37	0.20	0.15	0		
P5	0.34	0.14	0.28	0.29	0	
P6	0.23	0.25	0.11	0.22	0.39	0

- The distance matrix is

	P1	P2	P3	P4	P5	P6
P1	0					
P2	0.23	0				
P3	0.22	0.15	0			
P4	0.37	0.20	0.15	0		
P5	0.34	0.14	0.28	0.29	0	
P6	0.23	0.25	0.11	0.22	0.39	0

- To update the distance matrix MAX[dist(P3,P6),P1)]
- MAX(dist(P3,P1), (P6,P1))

  = MAX[(0.22,0.23)]

  = 0.23
- To update the distance matrix MAX[dist(P3,P6),P2)]
- MAX(dist(P3,P2), (P6,P2))

  = MAX[(0.15,0.25)]

  = 0.25

	P1	P2	P3	P4	P5	P6
P1	0					
P2	0.23	0				
P3	0.22	0.15	0			
P4	0.37	0.20	0.15	0		
P5	0.34	0.14	0.28	0.29	0	
P6	0.23	0.25	0.11	0.22	0.39	0

- To update the distance matrix MAX[dist(P3,P6),P1)]
- MAX(dist(P3,P1), (P6,P1))

  = MAX[(0.22,0.23)]

  = 0.23
- To update the distance matrix MAX[dist(P3,P6),P2)]
- MAX(dist(P3,P2), (P6,P2))

  = MAX[(0.15,0.25)]

  = 0.25

	P1	P2	P3	P4	P5	P6
P1	0					
P2	0.23	0				
P3	0.22	0.15	0			
P4	0.37	0.20	0.15	0		
P5	0.34	0.14	0.28	0.29	0	
P6	0.23	0.25	0.11	0.22	0.39	0

- To update the distance matrix MAX[dist(P3,P6),P1)]
- MAX(dist(P3,P1), (P6,P1))

  = MAX[(0.22,0.23)]

  = 0.23
- To update the distance matrix MAX[dist(P3,P6),P2)]
- MAX(dist(P3,P2), (P6,P2))

  = MAX[(0.15,0.25)]

  = 0.25

	P1	P2	P3	P4	P5	P6
P1	0					
P2	0.23	0				
P3	0.22	0.15	0			
P4	0.37	0.20	0.15	0		
P5	0.34	0.14	0.28	0.29	0	
P6	0.23	0.25	0.11	0.22	0.39	0

- To update the distance matrix MAX[dist(P3,P6),P4)]
- MAX(dist(P3,P4), (P6,P4))

  = MAX[(0.15,0.22)]

  = 0.22
- To update the distance matrix MAX[dist(P3,P6),P5)]
- MAX(dist(P3,P5), (P6,P5))

  = MAX[(0.28,0.39)]

  = 0.39

	P1	P2	P3	P4	P5	P6
P1	0					
P2	0.23	0				
P3	0.22	0.15	0			
P4	0.37	0.20	0.15	0		
P5	0.34	0.14	0.28	0.29	0	
P6	0.23	0.25	0.11	0.22	0.39	0

- To update the distance matrix MAX[dist(P3,P6),P4)]
- MAX(dist(P3,P4), (P6,P4))

  = MAX[(0.15,0.22)]

  = 0.22
- To update the distance matrix MAX[dist(P3,P6),P5)]
- MAX(dist(P3,P5), (P6,P5))

  = MAX[(0.28,0.39)]

  = 0.39

	P1	P2	P3	P4	P5	P6
P1	0					
P2	0.23	0				
P3	0.22	0.15	0			
P4	0.37	0.20	0.15	0		
P5	0.34	0.14	0.28	0.29	0	
P6	0.23	0.25	0.11	0.22	0.39	0

- To update the distance matrix MAX[dist(P3,P6),P4)]
- MAX(dist(P3,P4), (P6,P4))

  = MAX[(0.15,0.22)]

  = 0.22
- To update the distance matrix MAX[dist(P3,P6),P5)]
- MAX(dist(P3,P5), (P6,P5))

  = MAX[(0.28,0.39)]

  = 0.39

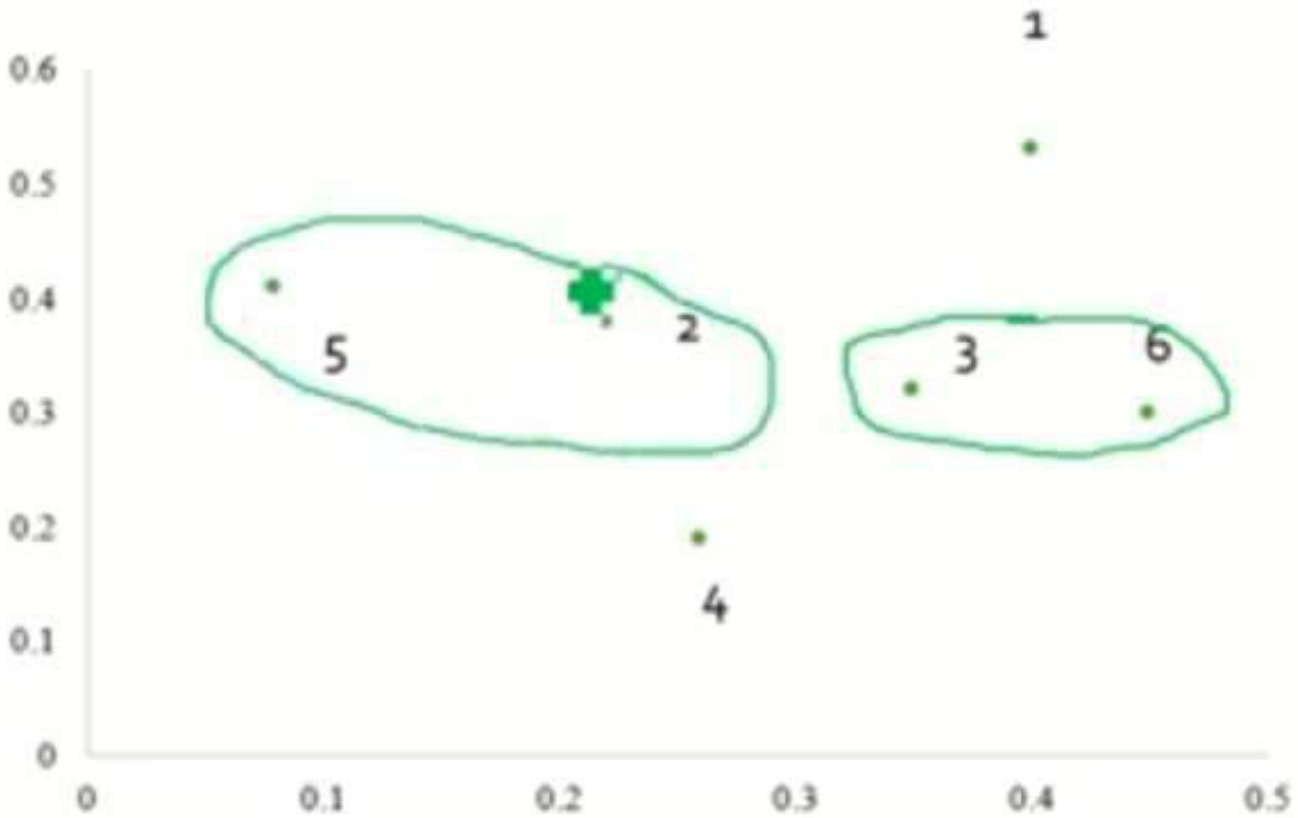
	P1	P2	P3	P4	P5	P6
P1	0					
P2	0.23	0				
P3	0.22	0.15	0			
P4	0.37	0.20	0.15	0		
P5	0.34	0.14	0.28	0.29	0	
P6	0.23	0.25	0.11	0.22	0.39	0

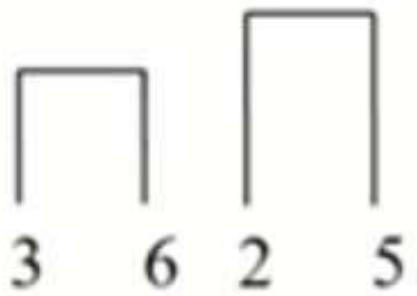
- The updated distance matrix for cluster P3,P6

	P1	P2	P3,P6	P4	P5
P1	0				
P2	0.23	0			
P3,P6	0.23	0.25	0		
P4	0.37	0.20	0.22	0	
P5	0.34	0.14	0.39	0.29	0

- The distance matrix

	P1	P2	P3,P6	P4	P5
P1	0				
P2	0.23	0			
P3,P6	0.23	0.25	0		
P4	0.37	0.20	0.22	0	
P5	0.34	0.14	0.39	0.29	0


1
0.6
0.5
0.4
0.3
0.2
0.1
0
5
2
3
6
4
0
0.1
0.2
0.3
0.4
0.5


3
6
2
5

- The distance matrix , cluster (P2,P5)

	P1	P2	P3,P6	P4	P5
P1	0				
P2	0.23	0			
P3,P6	0.23	0.25	0		
P4	0.37	0.20	0.22	0	
P5	0.34	0.14	0.39	0.29	0

- To update the distance matrix MAX[dist(P2,P5),P1)]
- MAX[dist(P2,P1), (P5,P1)]

  = MAX[(0.23,0.34)]

  = 0.34
- To update the distance matrix MAX[dist(P2,P5),(P3,P6)]
- MAX[dist(P2,(P3,P6)), (P5,(P3,P6))]

  = MAX[(0.25,0.39)]

  = 0.39

	P1	P2	P3,P6	P4	P5
P1	0				
P2	0.23	0			
P3,P6	0.25	0.25	0		
P4	0.37	0.20	0.22	0	
P5	0.34	0.14	0.39	0.29	0

- To update the distance matrix MAX[dist(P2,P5),P1)]
- MAX[dist(P2,P1), (P5,P1)]

  = MAX[(0.23,0.34)]

  = 0.34
- To update the distance matrix MAX[dist(P2,P5),(P3,P6)]
- MAX[dist(P2,(P3,P6)), (P5,(P3,P6))]

  = MAX[(0.25,0.39)]

  = 0.39

	P1	P2	P3,P6	P4	P5
P1	0				
P2	0.23	0			
P3,P6	0.25	0.25	0		
P4	0.37	0.20	0.22	0	
P5	0.34	0.14	0.39	0.28	0

- To update the distance matrix MAX[dist(P2,P5),P1)]
- MAX[dist(P2,P1), (P5,P1)]

  = MAX[(0.23,0.34)]

  = 0.34
- To update the distance matrix MAX[dist(P2,P5),(P3,P6)]
- MAX[dist(P2,(P3,P6)), (P5,(P3,P6))]

  = MAX[(0.25,0.39)]

  = 0.39

	P1	P2	P3,P6	P4	P5
P1	0				
P2	0.23	0			
P3,P6	0.23	0.25	0		
P4	0.37	0.20	0.22	0	
P5	0.34	0.14	0.39	0.29	0

- To update the distance matrix MAX[dist(P2,P5),P4)]
- MAX[dist(P2,P4), (P5,P4)]

  = MAX[(0.20,0.29)]

  = 0.29

	P1	P2	P3,P6	P4	P5
P1	0				
P2	0.23	0			
P3,P6	0.23	0.23	0		
P4	0.37	0.20	0.22	0	
P5	0.34	0.14	0.39	0.29	0

- To update the distance matrix MAX[dist(P2,P5),P4)]
- MAX[dist(P2,P4), (P5,P4)]

  = MAX[(0.20,0.29)]

  = 0.29

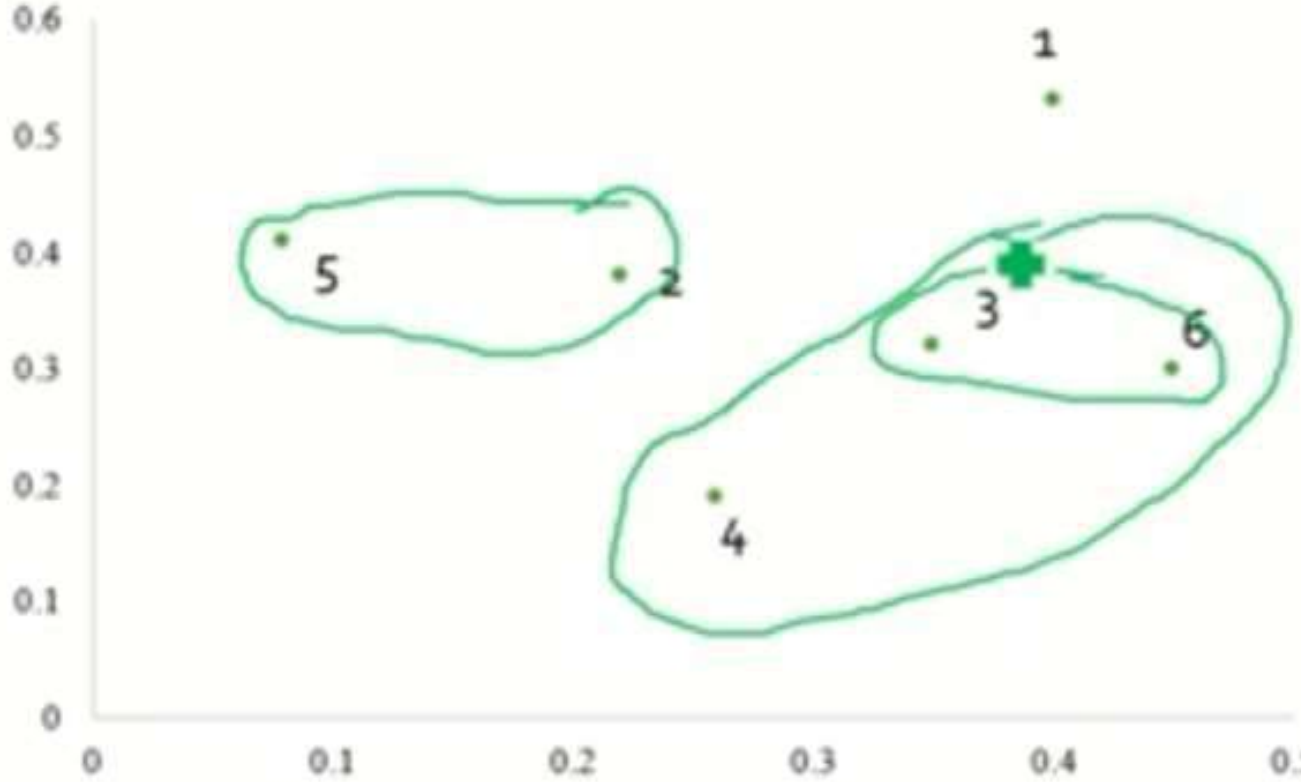
	P1	P2	P3,P6	P4	P5
P1	0				
P2	0.23	0			
P3,P6	0.23	0.23	0		
P4	0.37	0.20	0.22	0	
P5	0.34	0.14	0.39	0.29	0

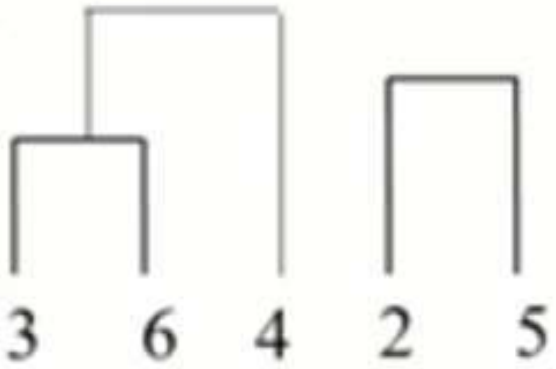
- The updated distance matrix for cluster (P2,P5)

	P1	P2,P5	P3,P6	P4
P1	0			
P2,P5	0.34	0		
P3,P6	0.23	0.39	0	
P4	0.37	0.29	0.22	0

- The distance matrix is

	P1	P2,P5	P3,P6	P4
P1	0			
P2,P5	0.34	0		
P3,P6	0.23	0.39	0	
P4	0.37	0.29	0.22	0


0.6
0.5
0.4
0.3
0.2
0.1
0
0
0.1
0.2
0.3
0.4
0.5
1
2
3
4
5
6


3
6
4
2
5

- To update the distance matrix[((P3,P6),P4) , P1]
- MAX[dist(P3,P6),P1), (P4,P1)]

  = MAX[(0.23,0.37)]

  = 0.37
- To update the distance matrix MAX[dist((P3,P6),P4),(P2,P5)]
- MAX[dist((P3,P6),(P2,P5)), (P4,(P2,P5))]

  = MAX[(0.39,0.29)]

  = 0.39

	P1	P2,P5	P3,P6	P4
P1	0			
P2,P5	0.34	0		
P3,P6	0.23	0.39	0	
P4	0.37	0.29	0.22	0

- To update the distance matrix[((P3,P6),P4) , P1]
- MAX[dist(P3,P6),P1), (P4,P1)]

  = MAX[(0.23,0.37)]

  = 0.37
- To update the distance matrix MAX[dist((P3,P6),P4),(P2,P5)]
- MAX[dist((P3,P6),(P2,P5)), (P4,(P2,P5))]

  = MAX[(0.39,0.29)]

  = 0.39

	P1	P2,P5	P3,P6	P4
P1	0			
P2,P5	0.34	0		
P3,P6	0.23	0.39	0	
P4	0.37	0.29	0.22	0

- To update the distance matrix[((P3,P6),P4) , P1]
- MAX[dist(P3,P6),P1), (P4,P1)]

  = MAX[(0.23,0.37)]

  = 0.37
- To update the distance matrix MAX[dist((P3,P6),P4),(P2,P5)]
- MAX[dist((P3,P6),(P2,P5)), (P4,(P2,P5))]

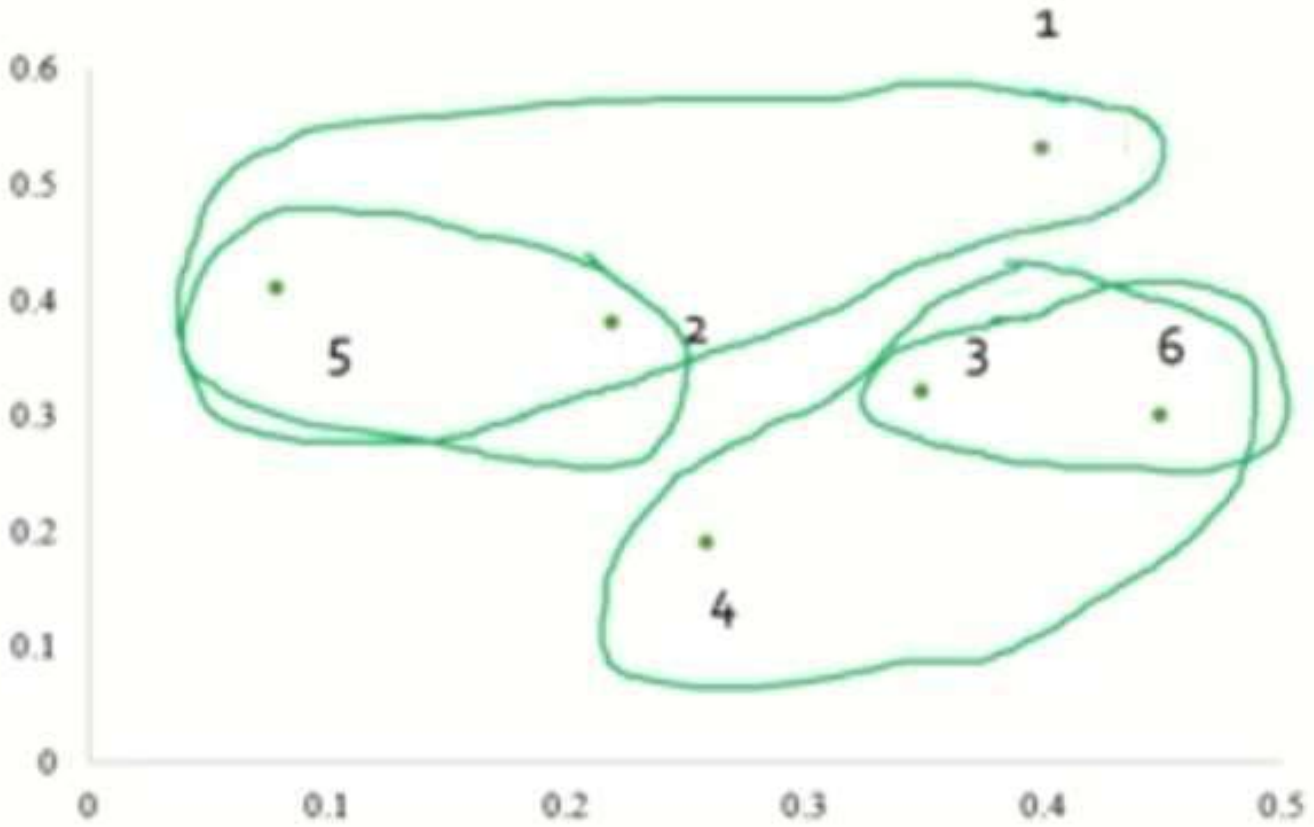
  = MAX[(0.39,0.29)]

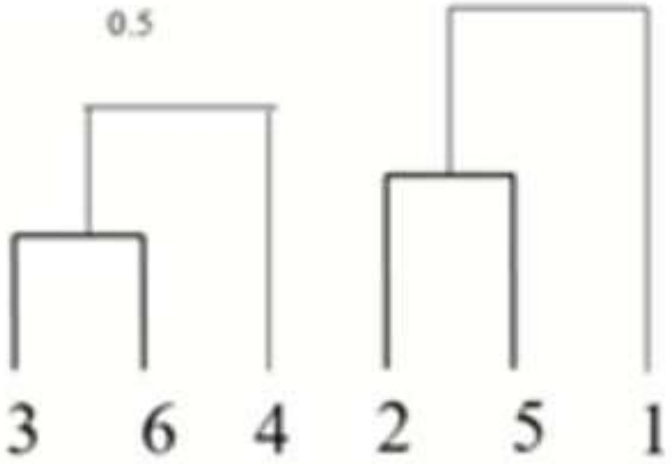
  = 0.39

	P1	P2,P5	P3,P6	P4
P1	0			
P2,P5	0.34	0		
P3,P6	0.23	0.39	0	
P4	0.37	0.29	0.22	0

- The updated distance matrix for cluster P3,P6,P4

	P1	P2,P5	P3,P6,P4
P1	0		
P2,P5	0.34	0	
P3,P6,P4	0.37	0.39	0


1
2
3
4
5
6
0.6
0.5
0.4
0.3
0.2
0.1
0
0
0.1
0.2
0.3
0.4
0.5


3
6
4
2
5
1

- **Cluster** [(P2,P5),P1]
- MAX[dist(P2,P5),(P3,P6,P4), (P1,(P3,P6,P4))]

  = MAX[(0.39,0.37)]

  = 0.39

	P1	P2,P5	P3,P6,P4
P1	0		
P2,P5	0.34	0	
P3,P6,P4	0.37	0.39	0

- **Cluster** [(P2,P5),P1]
- MAX[dist(P2,P5),(P3,P6,P4), (P1,(P3,P6,P4))]

  = MAX[(0.39,0.37)]

  = 0.39

	P1	P2,P5	P3,P6,P4
P1	0		
P2,P5	0.34	0	
P3,P6,P4	0.37	0.39	0

- **Cluster** [(P2,P5),P1]
- MAX[dist(P2,P5),(P3,P6,P4), (P1,(P3,P6,P4))]

  = MAX[(0.39,0.37)]

  = 0.39

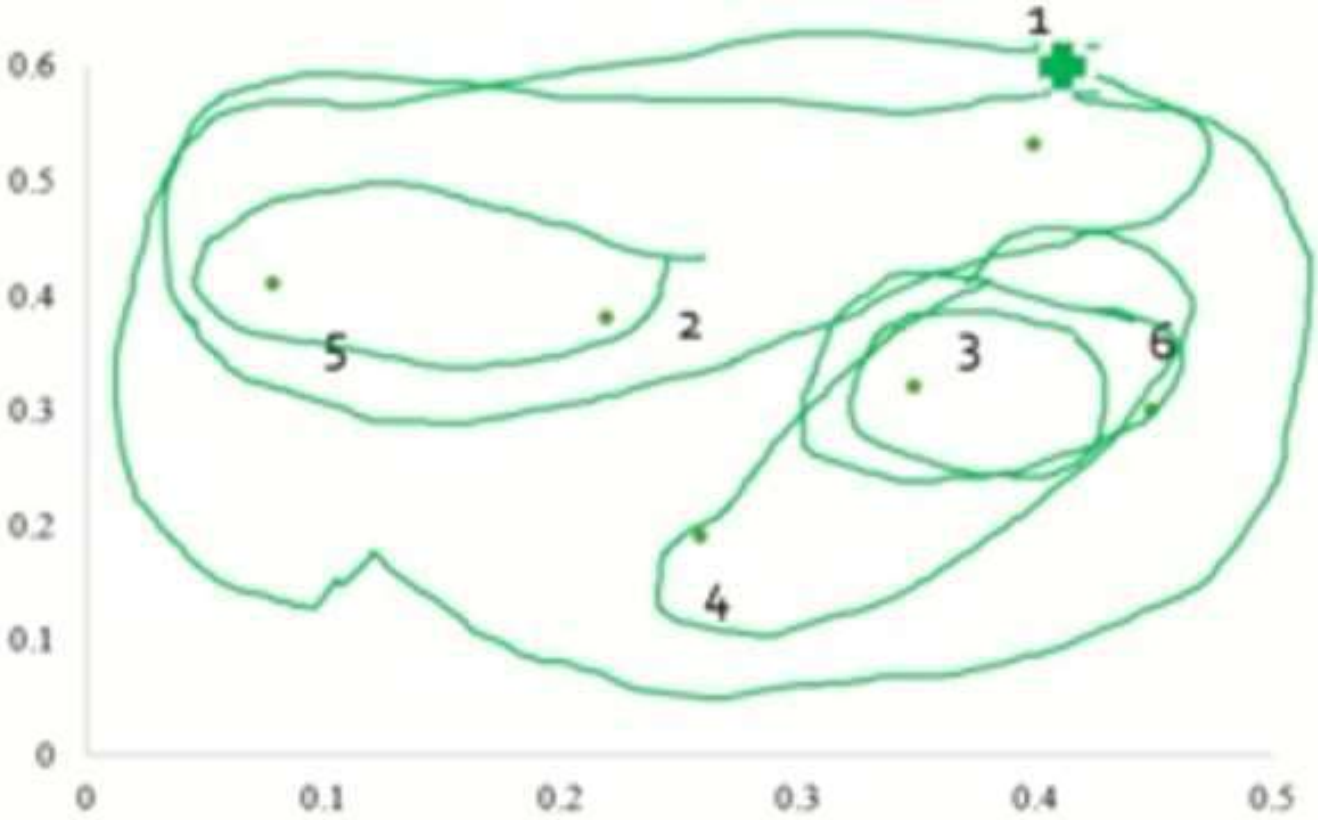
	P1	P2,P5	P3,P6,P4
P1	0		
P2,P5	0.34	0	
P3,P6,P4	0.37	0.39	0

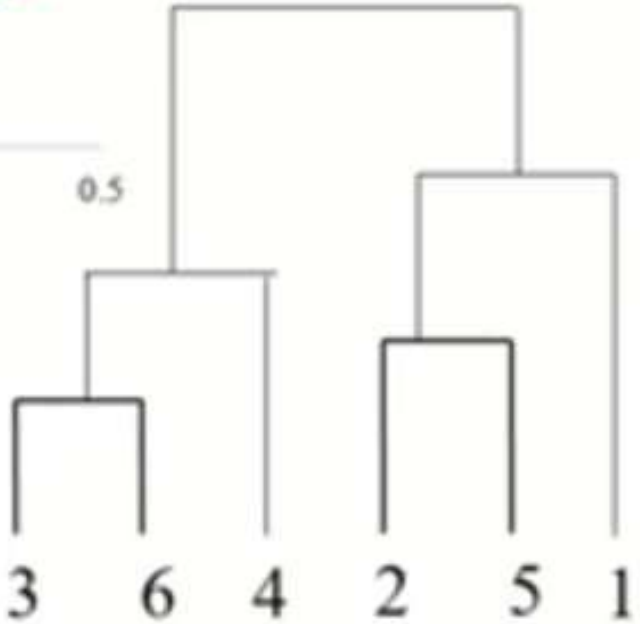
- The updated distance matrix for cluster P2,P5,P1

	P2,P5,P1	P3,P6,P4
P2,P5,P1	0	
P3,P6,P4	0.39	0

- The distance matrix

	P2,P5,P1	P3,P6,P4
P2,P5,P1	0	
P3,P6,P4	0.39	0


1
2
3
4
5
6
0.6
0.5
0.4
0.3
0.2
0.1
0
0
0.1
0.2
0.3
0.4
0.5


3
6
4
2
5
1

What Is A Good Clustering?

- Internal criterion: A good clustering will produce high quality clusters in which:
  - the intra-class (that is, intra-cluster) similarity is high
  - the inter-class similarity is low
  - The measured quality of a clustering depends on both the document representation and the similarity measure used

External criteria for clustering quality

- Quality measured by its ability to discover some or all of the hidden patterns or latent classes in gold standard data
- Assesses a clustering with respect to ground truth … requires *labeled data*
- Assume documents with C gold standard classes, while our clustering algorithms produce K clusters, $\omega_1, \omega_2, \ldots, \omega_K$ with n_i members.

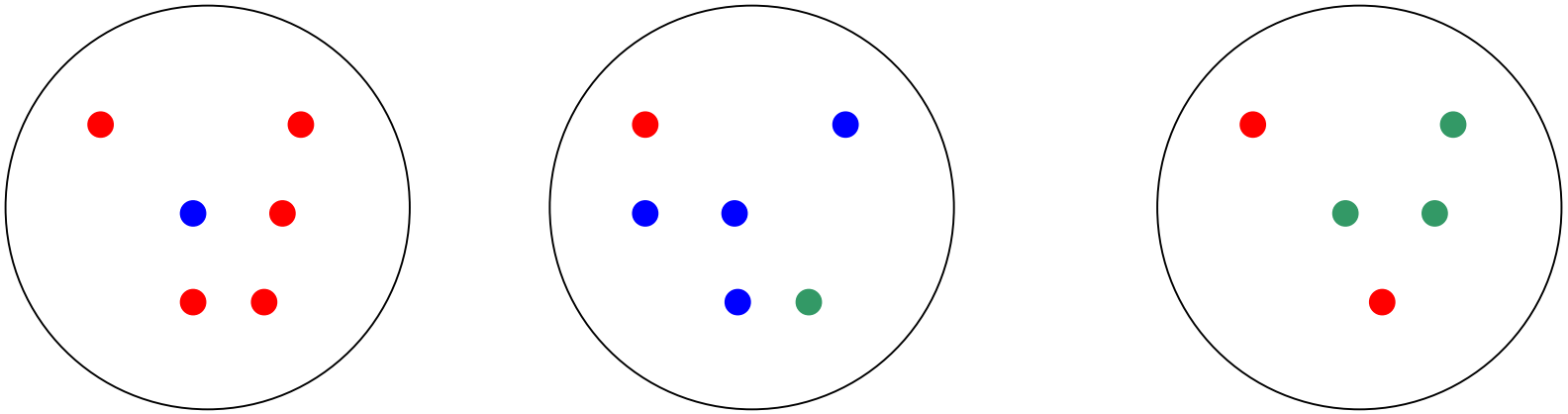
External Evaluation of Cluster Quality

- Simple measure: purity, the ratio between the dominant class in the cluster π_i and the size of cluster ω_i

$$Purity(\omega_i) = \frac{1}{n_i} \max_j (n_{ij}) \quad j \in C$$

- Biased because having *n* clusters maximizes purity
- Others are entropy of classes in clusters (or mutual information between classes and clusters)

Purity example



Cluster I Cluster II Cluster III

Cluster I: Purity = 1/6 (max(5, 1, 0)) = 5/6

Cluster II: Purity = 1/6 (max(1, 4, 1)) = 4/6

Cluster III: Purity = 1/5 (max(2, 0, 3)) = 3/5

Rand Index measures between pair decisions. Here RI = 0.68

Number of points	Same Cluster in clustering	Different Clusters in clustering
Same class in ground truth	20	24
Different classes in ground truth	20	72

Rand index and Cluster F-measure

$$RI = \frac{A + D}{A + B + C + D}$$

Compare with standard Precision and Recall:

$$P = \frac{A}{A + B} \qquad R = \frac{A}{A + C}$$

People also define and use a cluster F-measure, which is probably a better measure.

Thank you for your attention