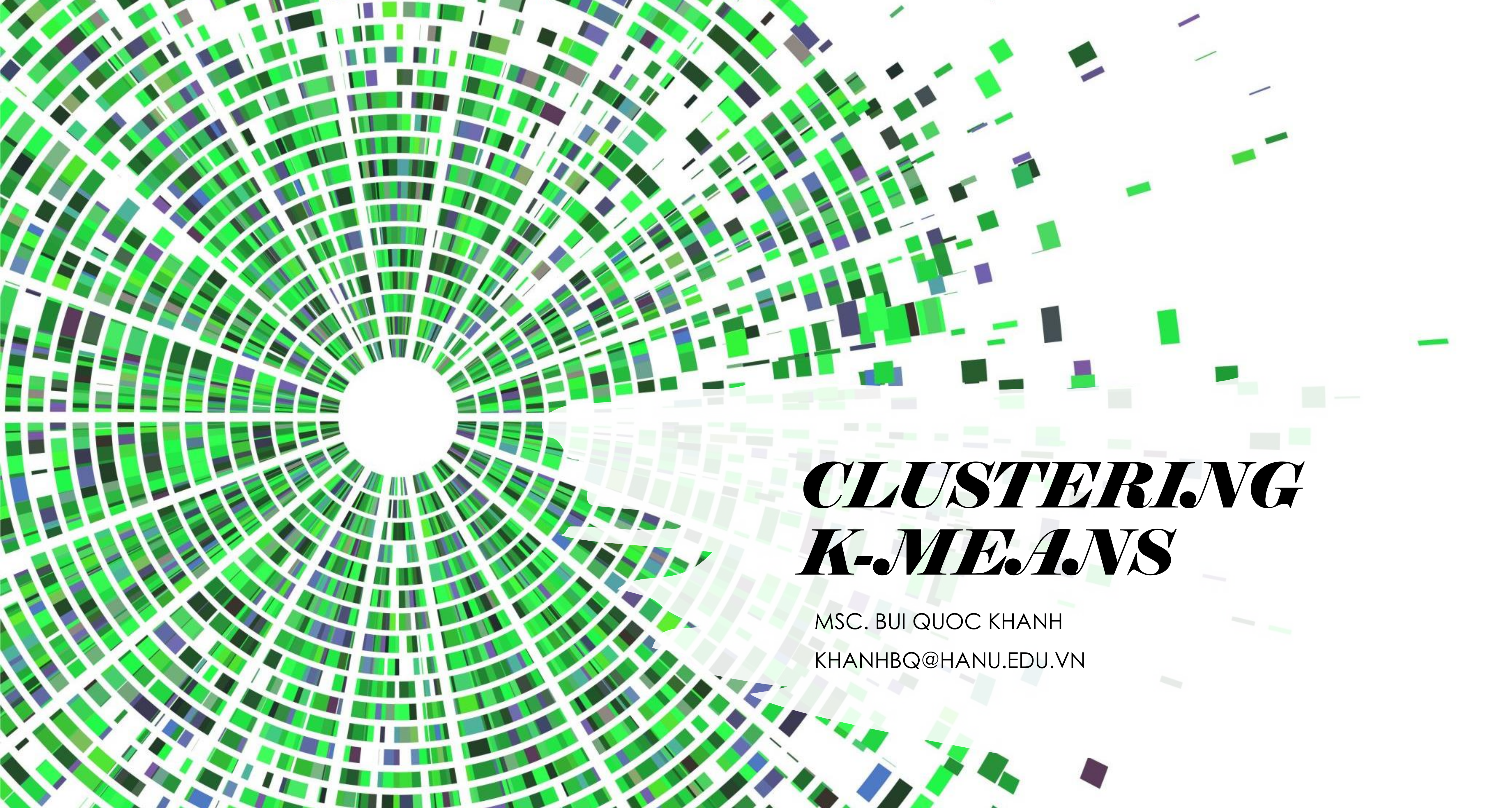
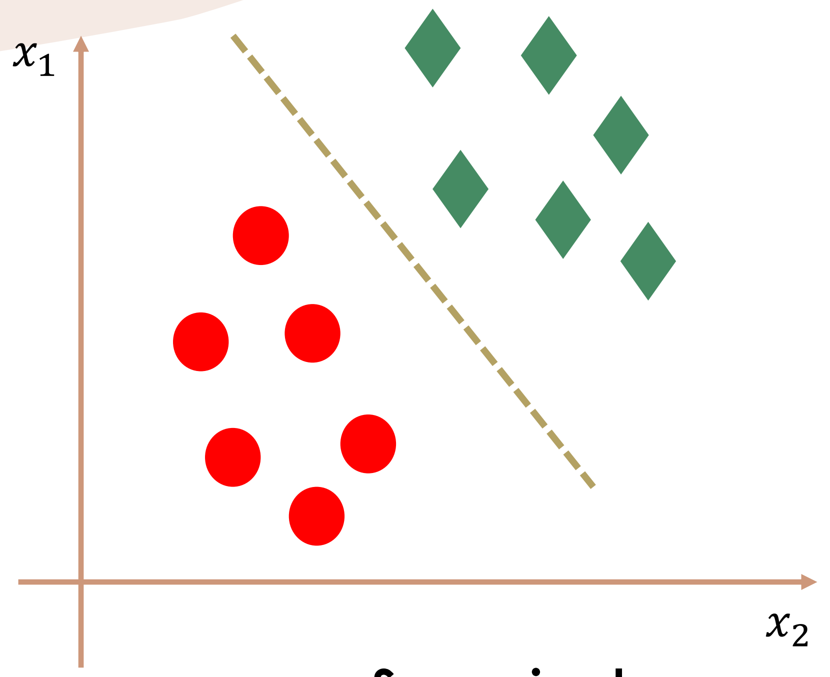


CLUSTERING K-MEANS

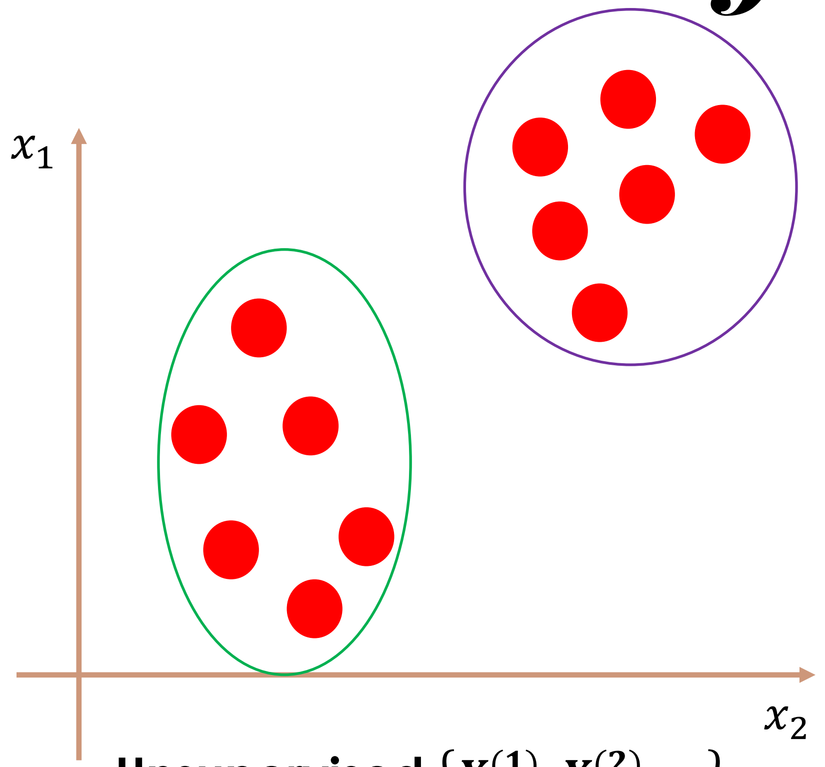
MSC. BUI QUOC KHANH

KHANHBQ@HANU.EDU.VN

Supervised vs Unsupervised Learning



Supervised
$\{(X^{(1)}, y^{(1)}), (X^{(2)}, y^{(2)}), \ldots\}$



Unsupervised $\{X^{(1)}, X^{(2)}, \ldots\}$

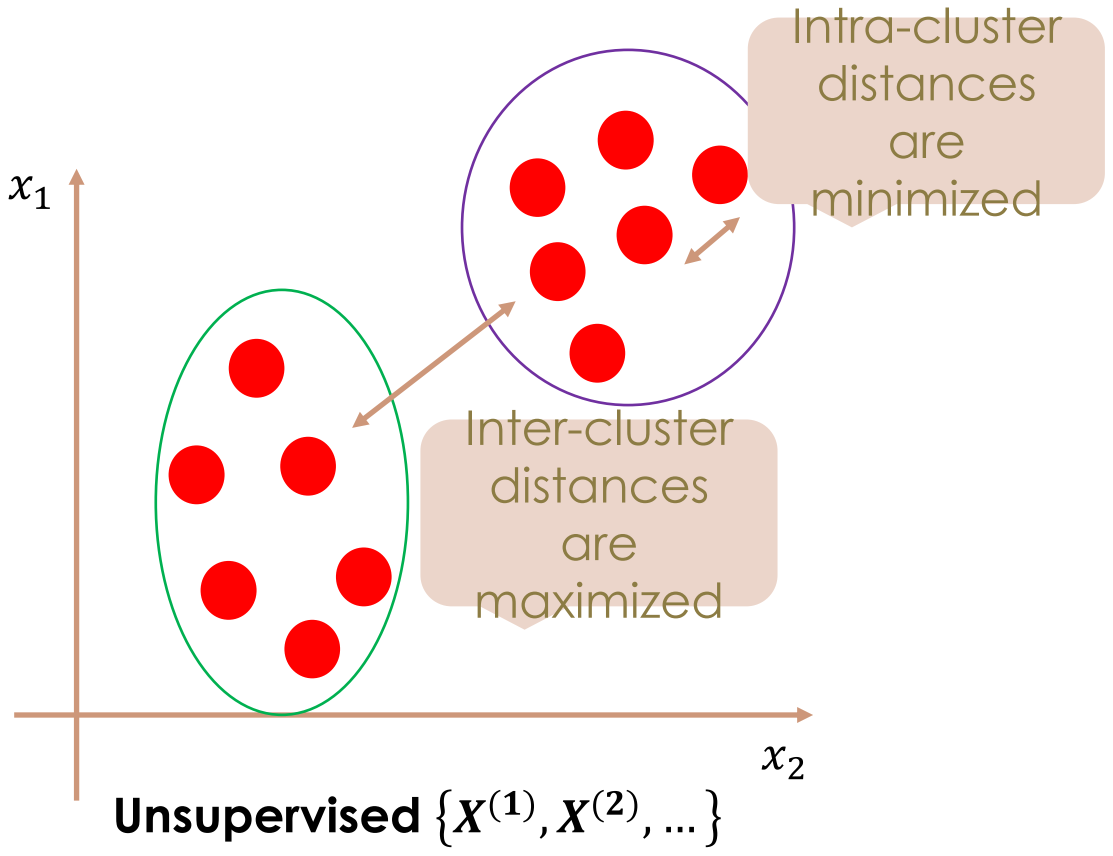
⌂ Goal: Find some structure (unsupervised) in the dataset. A type of structure (or pattern) is to find clusters

Clustering

- Cluster analysis or clustering is the task of assigning a set of objects into groups (called clusters) so that the objects in the same cluster are more similar (according to some metrics) to each other than to those in other clusters.
- Clustering is a task of descriptive data mining
- Unsupervised technique (no training set)

Clustering

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

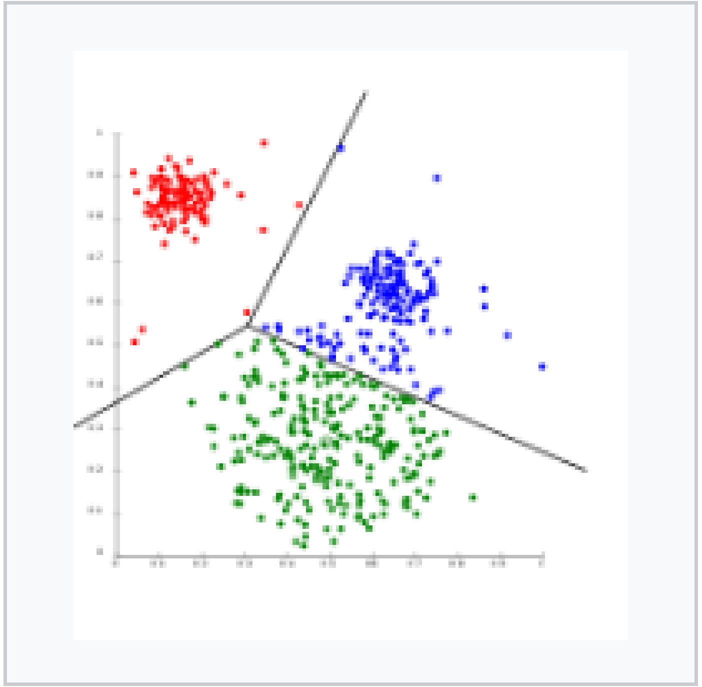

Intra-cluster distances are minimized
Inter-cluster distances are maximized
x_1
x_2
Unsupervised $\{X^{(1)}, X^{(2)}, \dots\}$

Clustering: Real life examples*

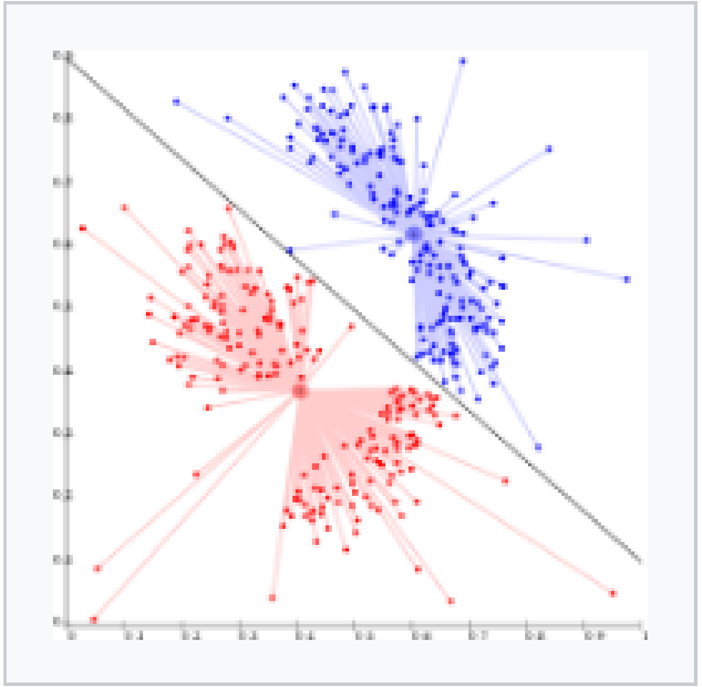
- Market Segmentation
    - Product design based on population attributes (e.g. T-shirt, shoes, etc.)
- Coherent group of friends on Social Media
- Organize computation clusters
- Image segmentation (Clusters: Background, Foreground)
- Species organization (classes / sub-classes)
- Document clustering, news clustering, WWW, etc.
    - We can build, specific knowledge-based systems, for each cluster of documents
- A pre-processing step for other algorithms

Clustering

k-means clustering examples



K-means separates data into Voronoi-cells, which assumes equal-sized clusters (not adequate here)



K-means cannot represent density-based clusters

Source: Wikipedia

K-means problem statement

- Given a set of instances (data set)

$$X = \{x_1, \dots, x_n\}$$

- find a partition of k subsets (clusters) which minimizes the objective function Sum of the Squared Error

$$SSE = \Sigma_{i=1,k} \Sigma_{Xj \in Ci} \text{distance}(x_j, c_i)$$

- where C_i is the i-th cluster, c_i is the centroid of C_i
- The problem is NP-hard

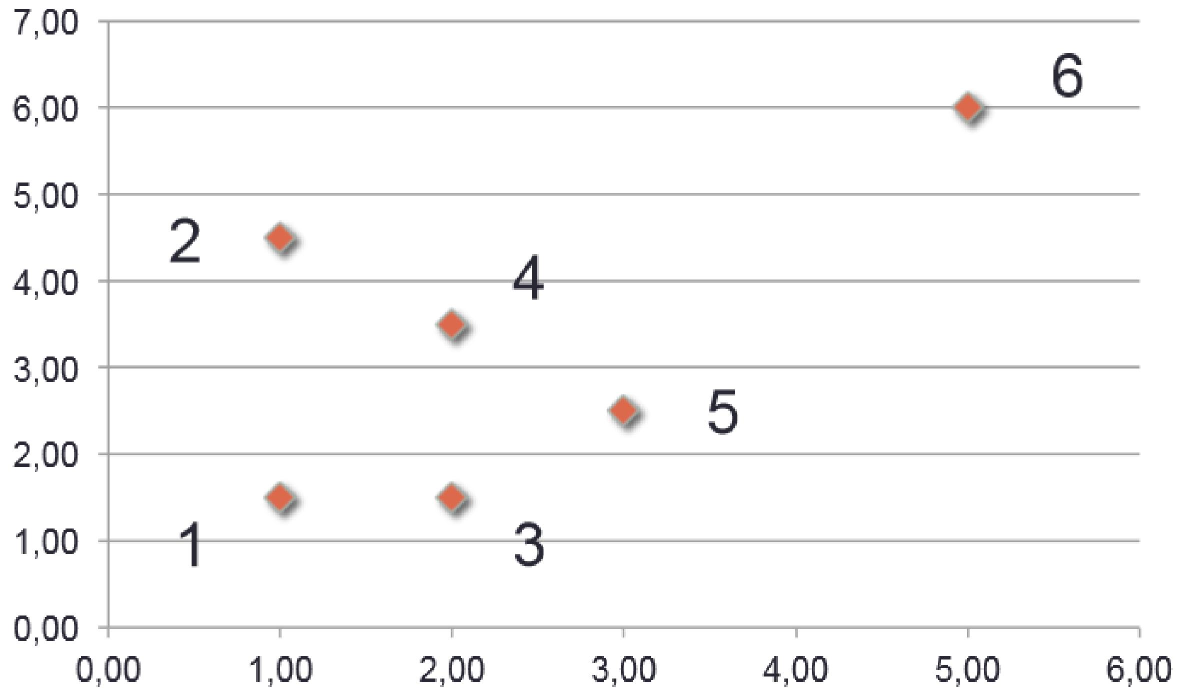
K-means algorithm

- Input: data set, k (number of clusters)
- Select (non-deterministically) k points as initial centroids
- repeat
  - Form k clusters by assigning each point (instance) to the closest centroid
  - Recompute the centroid of each cluster
- until centroids do not change
- end.
- What does “closest” mean?
  - minimum distance, e.g., Euclidean distance

K-means - An Example

Data set

Object	X	Y
1	1.0	1.5
2	1.0	4.5
3	2.0	1.5
4	2.0	3.5
5	3.0	2.5
6	5.0	6.0



Problem: find two clusters that "best" partition the data set

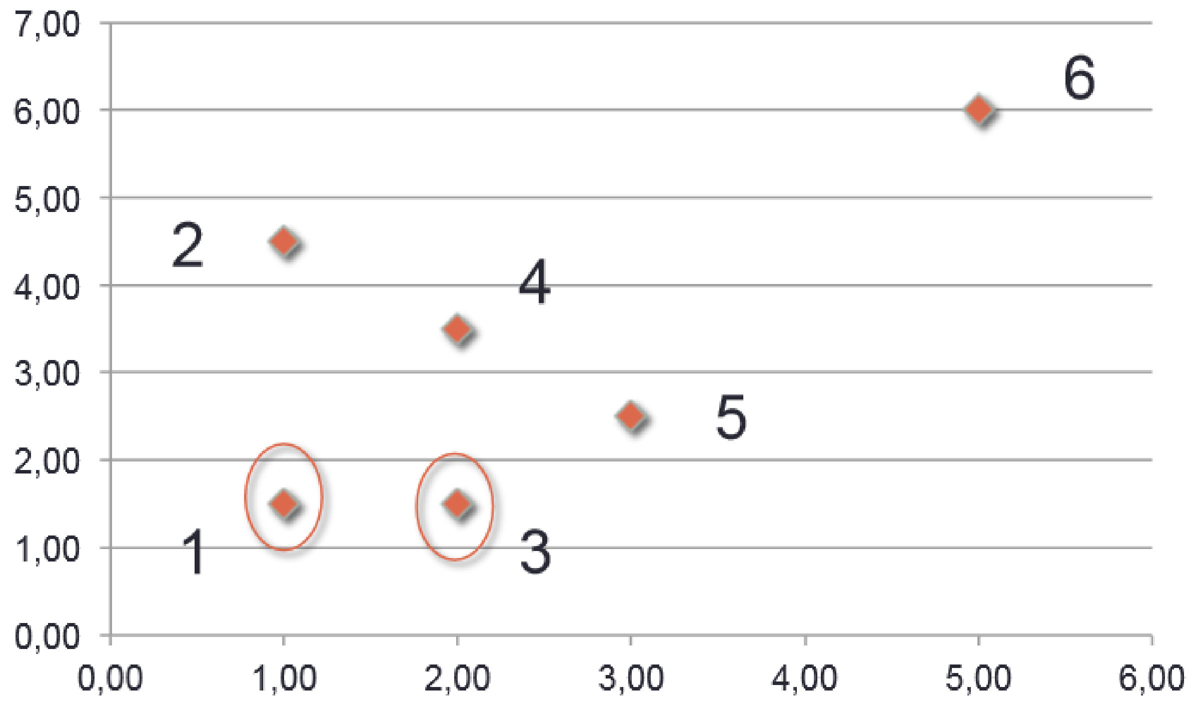
K-means algorithm

- Input: data set, k (number of clusters)
- Select (non-deterministically) k points as initial centroids
- repeat
  - Form k clusters by assigning each point (instance) to the closest centroid
  - Recompute the centroid of each cluster
- until centroids do not change
- end.
- What does “closest” mean?
  - minimum distance, e.g., Euclidean distance
- Centroid: mean value for each variable

K-means - An Example

STEP 1: Randomly chose 2 centroids, say, instances 1 and 3

Object	X	Y
1	1.0	1.5
2	1.0	4.5
3	2.0	1.5
4	2.0	3.5
5	3.0	2.5
6	5.0	6.0



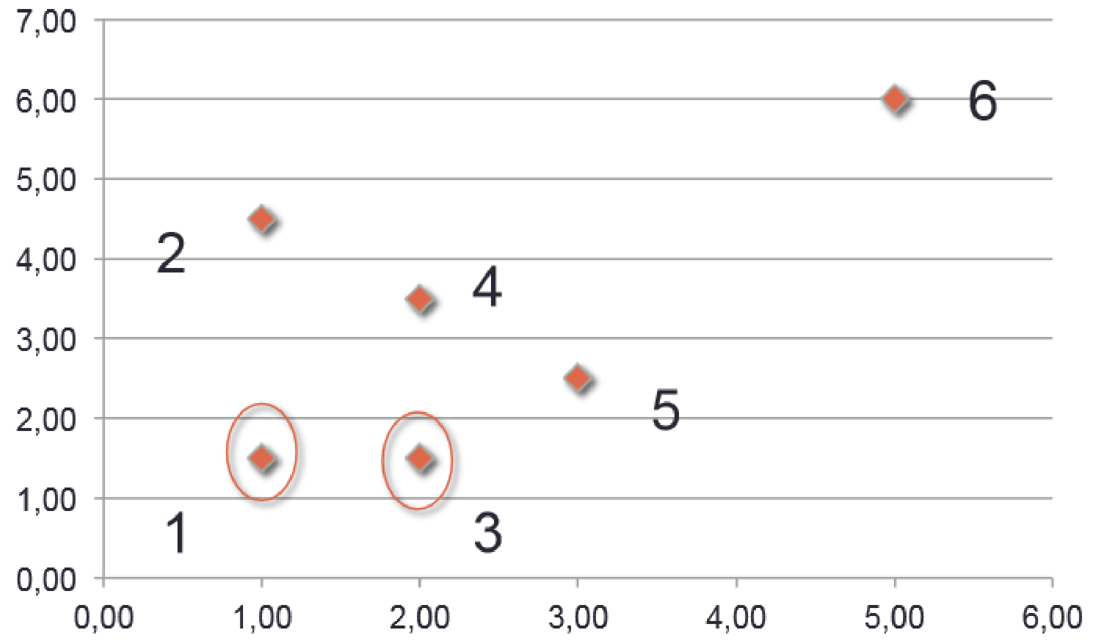
K-means algorithm

- Input: data set, k (number of clusters)
- Select (non-deterministically) k points as initial centroids
- repeat
  - Form k clusters by assigning each point (instance) to the closest centroid
  - Recompute the centroid of each cluster
- until centroids do not change
- end.
- What does “closest” mean?
  - minimum distance, e.g., Euclidean distance
- Centroid: mean value for each variable

K-means - An Example – iteration 1

- ITERATION 1 – centroids
  - c1=(1.0, 1.5)
  - c2=(2.0, 1.5)
- Compute distances

Object	Dist(c_1-point)	Dist(c_2-point)
1	0.00	1.00
2	3.00	3.16
3	1.00	0.00
4	2.24	2.00
5	2.24	1.41
6	6.02	5.41

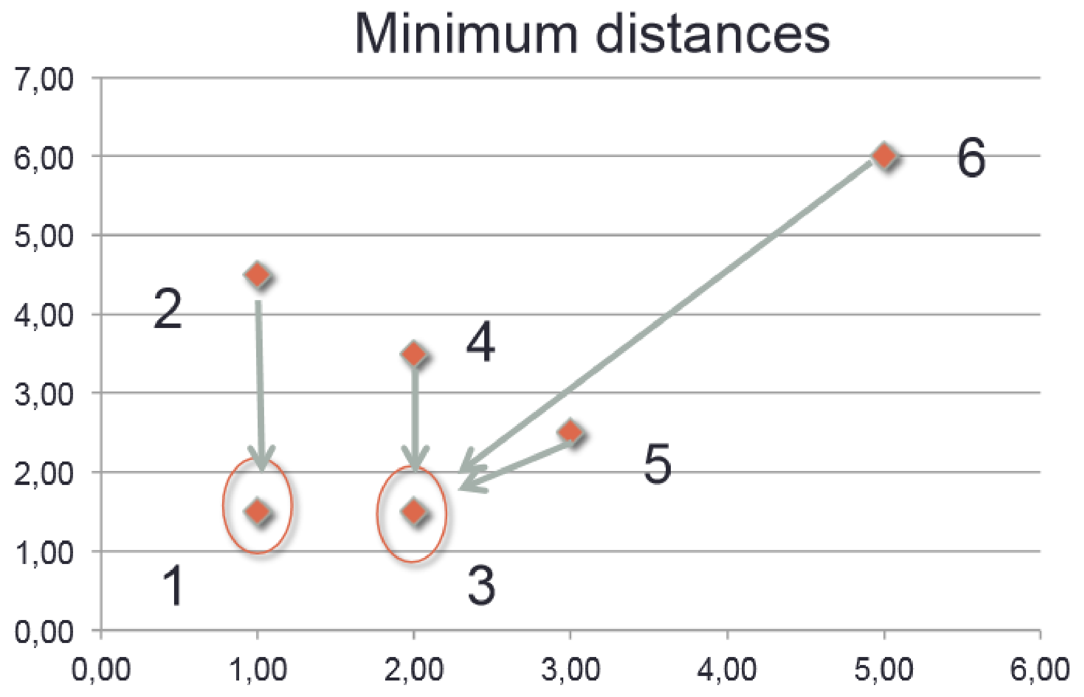


$$\text{distance}(A,B) = \sqrt{(x_1-x_2)^2 + (y_1-y_2)^2}$$

K-means - An Example – iteration 1

- ITERATION 1 – centroids
  - c1=(1.0, 1.5)
  - c2=(2.0, 1.5)
- Compute distances

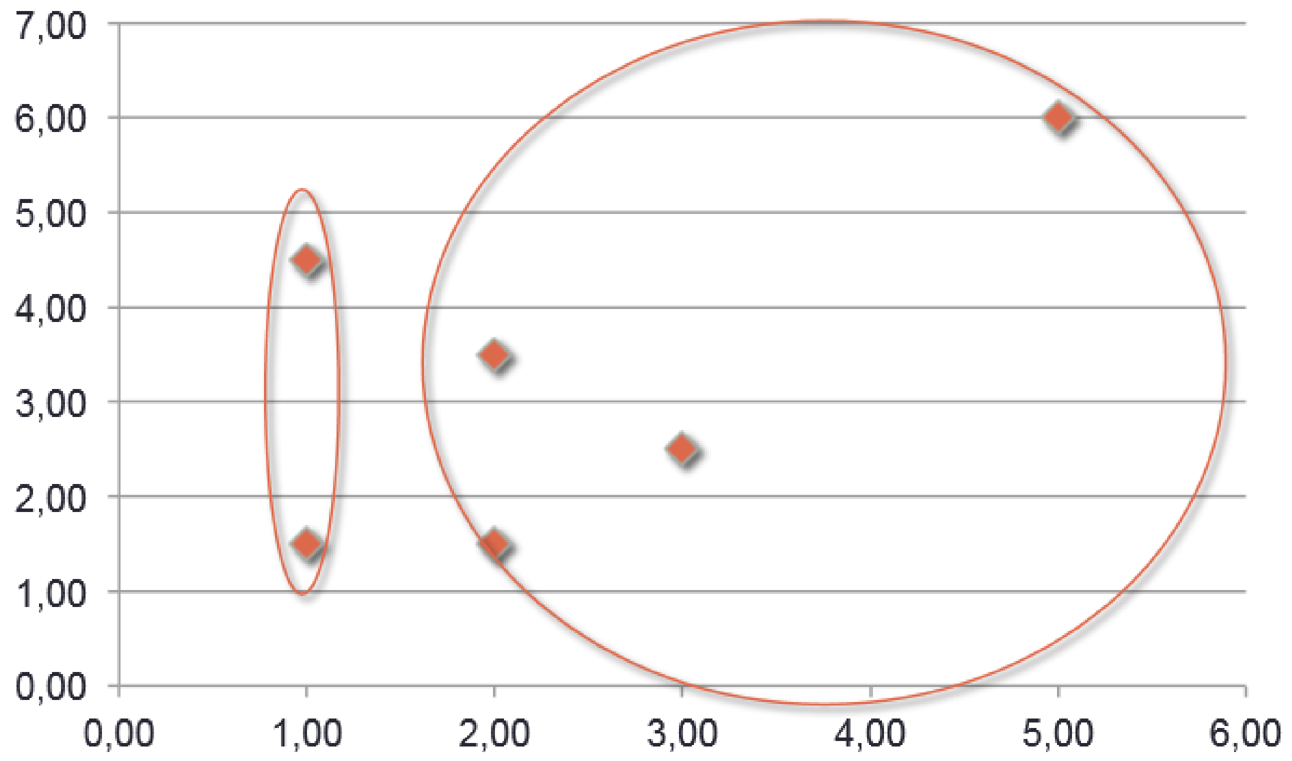
Object	Dist(c_1-point)	Dist(c_2-point)
1	0.00	1.00
2	3.00	3.16
3	1.00	0.00
4	2.24	2.00
5	2.24	1.41
6	6.02	5.41



$$\text{distance}(A,B) = \sqrt{(x_1-x_2)^2 + (y_1-y_2)^2}$$

K-means - An Example – iteration 1

- ITERATION 1
  - C1 = {1, 2}
  - C2 = {3, 4, 5, 6}



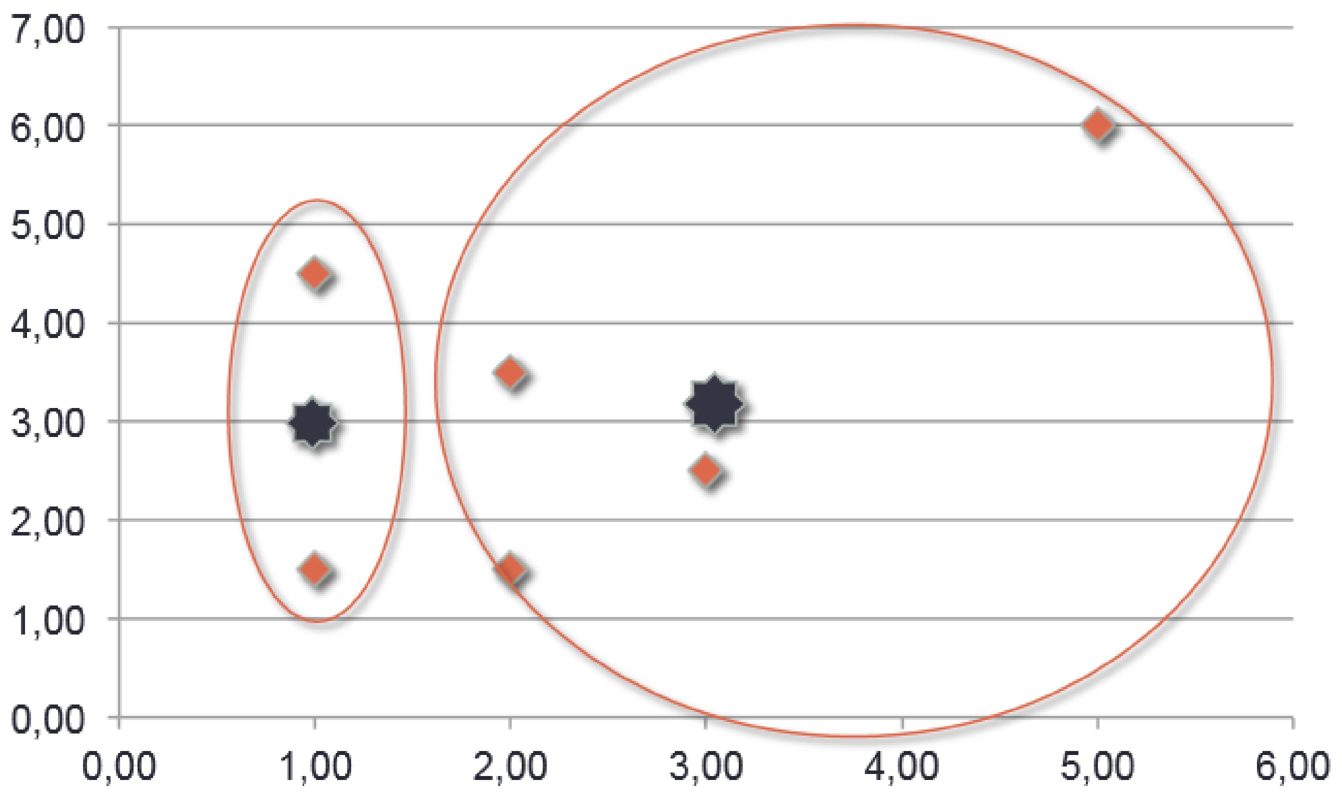
Clusters after Iteration 1

K-means algorithm

- Input: data set, k (number of clusters)
- Select (non-deterministically) k points as initial centroids
- repeat
  - Form k clusters by assigning each point (instance) to the closest centroid
  - Recompute the centroid of each cluster
- until centroids do not change
- end.
- What does “closest” mean?
  - minimum distance, e.g., Euclidean distance
- Centroid: mean value for each variable

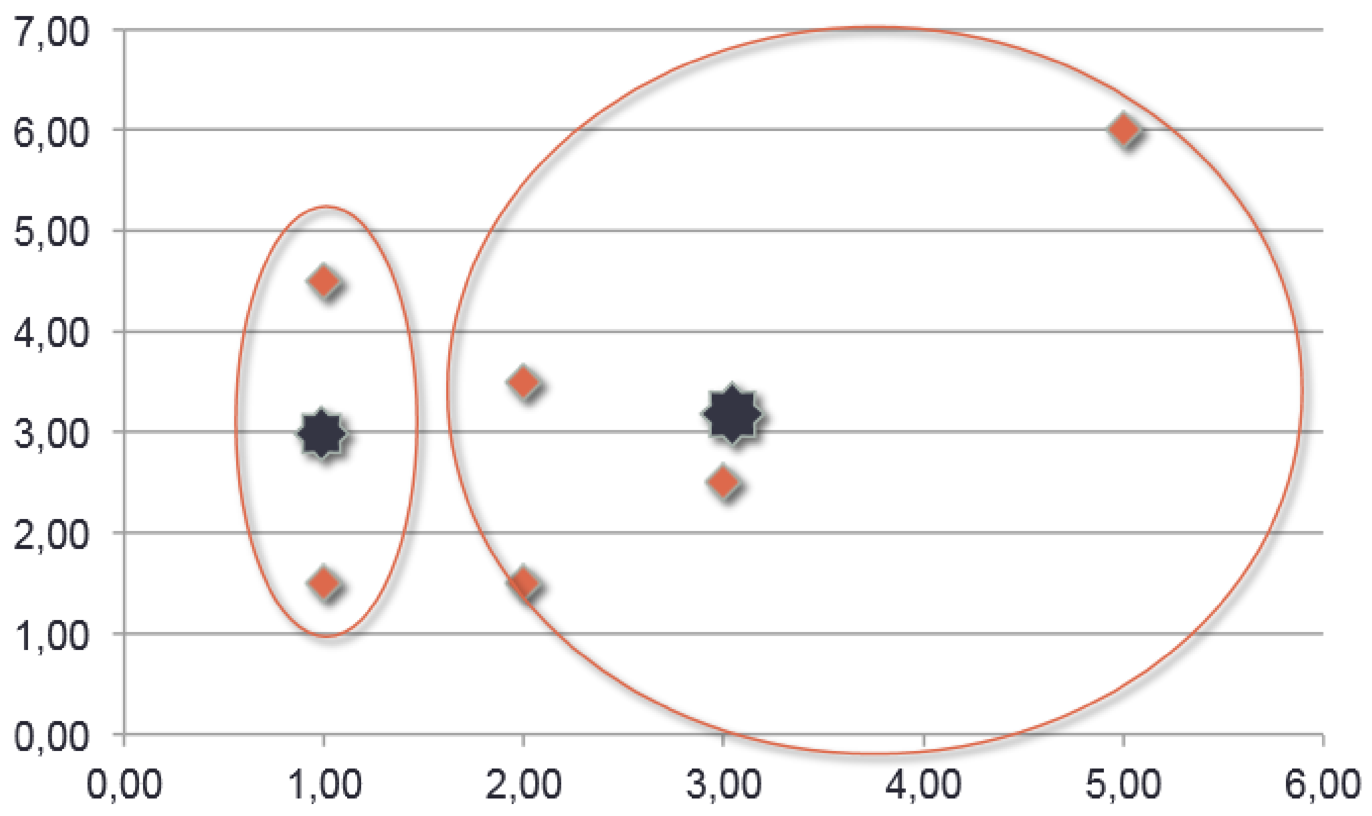
K-means - An Example – iteration 1

- ITERATION 1
- centroid for C_1
  - $x=(1+1)/2 = 1$
  - $y= (1.5+4.5)/2 = 3$
  - $c_1= (1, 3)$
- centroid for C_2
  - $x=(2.0+2.0+3.0+5.0)/4 = 3.0$
  - $y=(1.5+3.5+2.5+6.0)/4=3.34$
  - $c_2=(3, 3.34)$
- Exit condition not satisfied



K-means - An Example – iteration 2

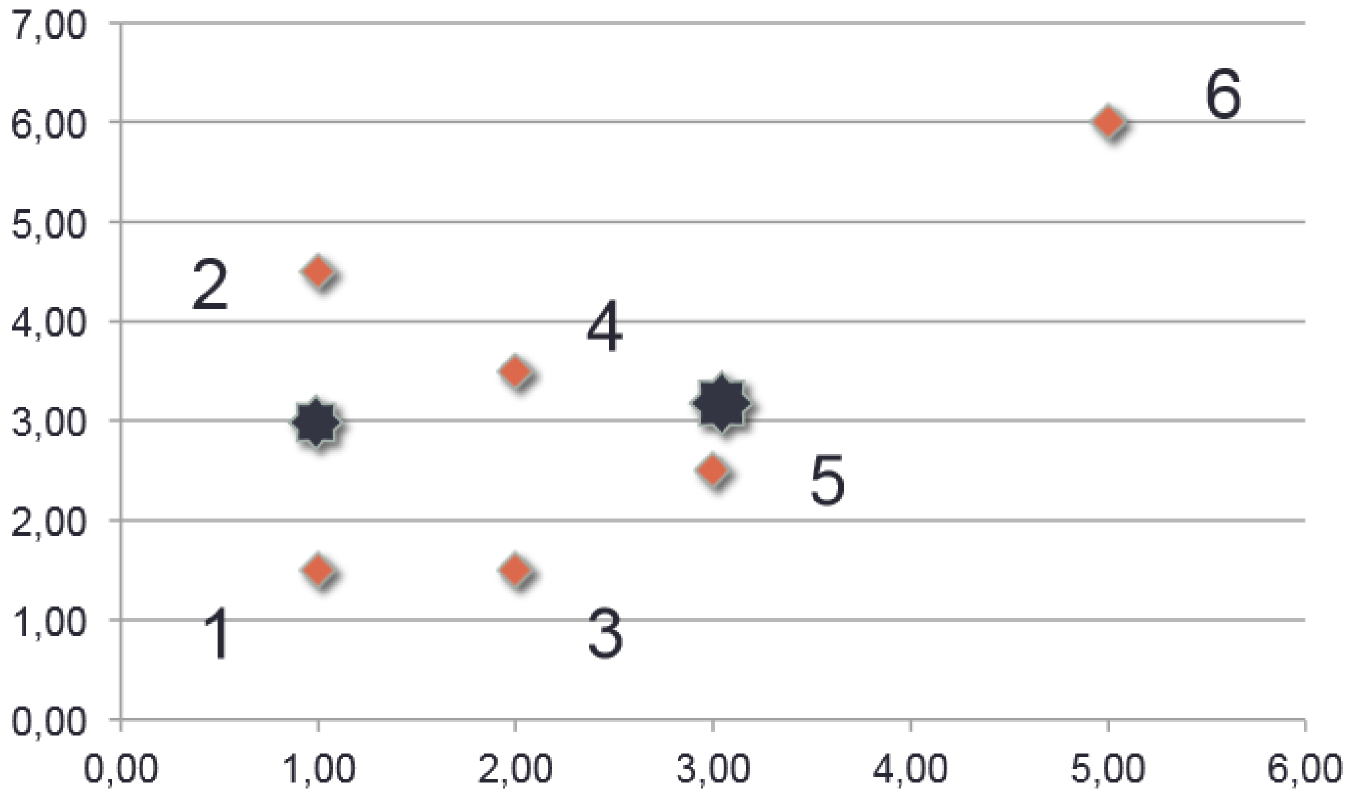
- ITERATION 2 – initial clusters
  - C1 = {1, 2}
  - C2 = {3, 4, 5, 6}
- with centroids
  - c1= (1, 3)
  - c2=(3, 3.34)



K-means - An Example – iteration 2

- ITERATION 2 – centroids
- c1= (1, 3)
- c2=(3, 3.34)
- Compute distances

Object	Dist(c_1-point)	Distc_2-point)
1	1.50	2.74
2	1.50	2.29
3	1.80	2.12
4	1.12	1.01
5	2.06	0.87
6	5.00	3.30

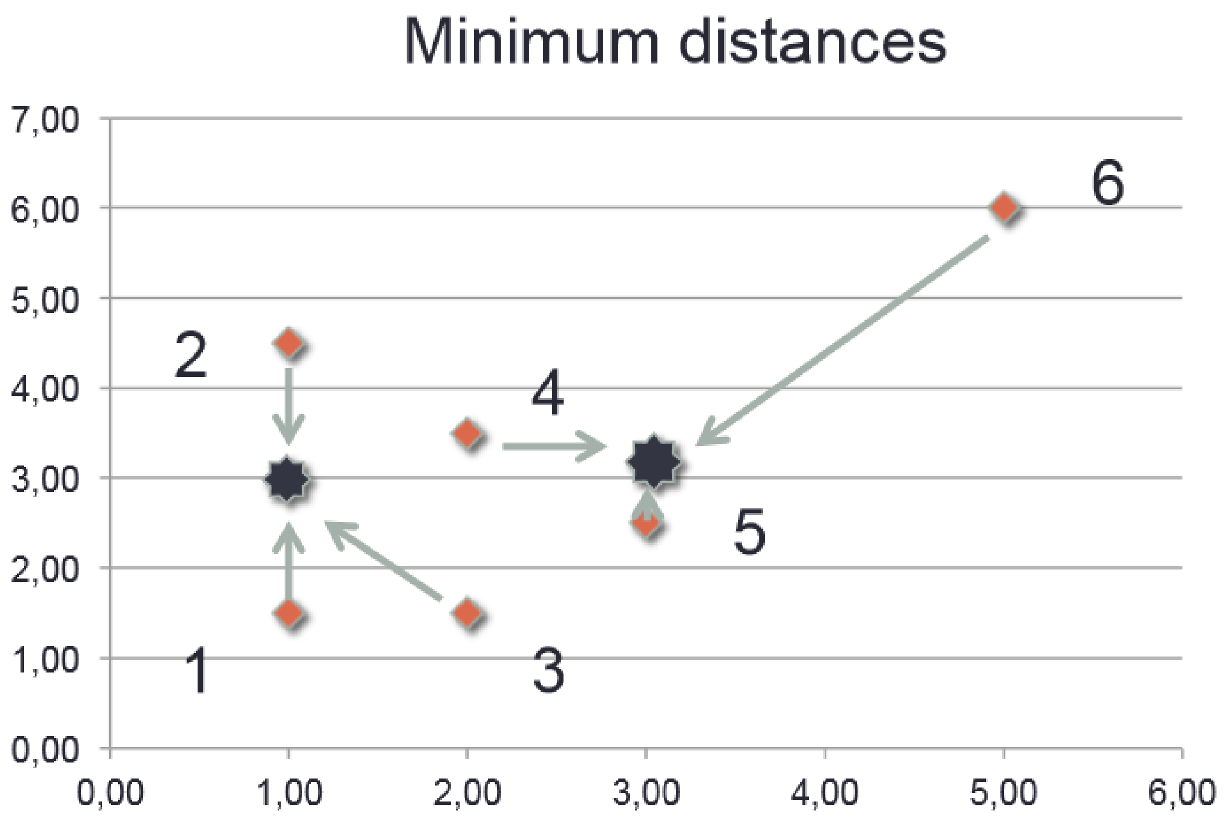


$$\text{distance}(A,B) = \sqrt{(x_1-x_2)^2 + (y_1-y_2)^2}$$

K-means - An Example – iteration 2

- ITERATION 2 – centroids
- c1= (1, 3)
- c2=(3, 3.34)
- Compute distances

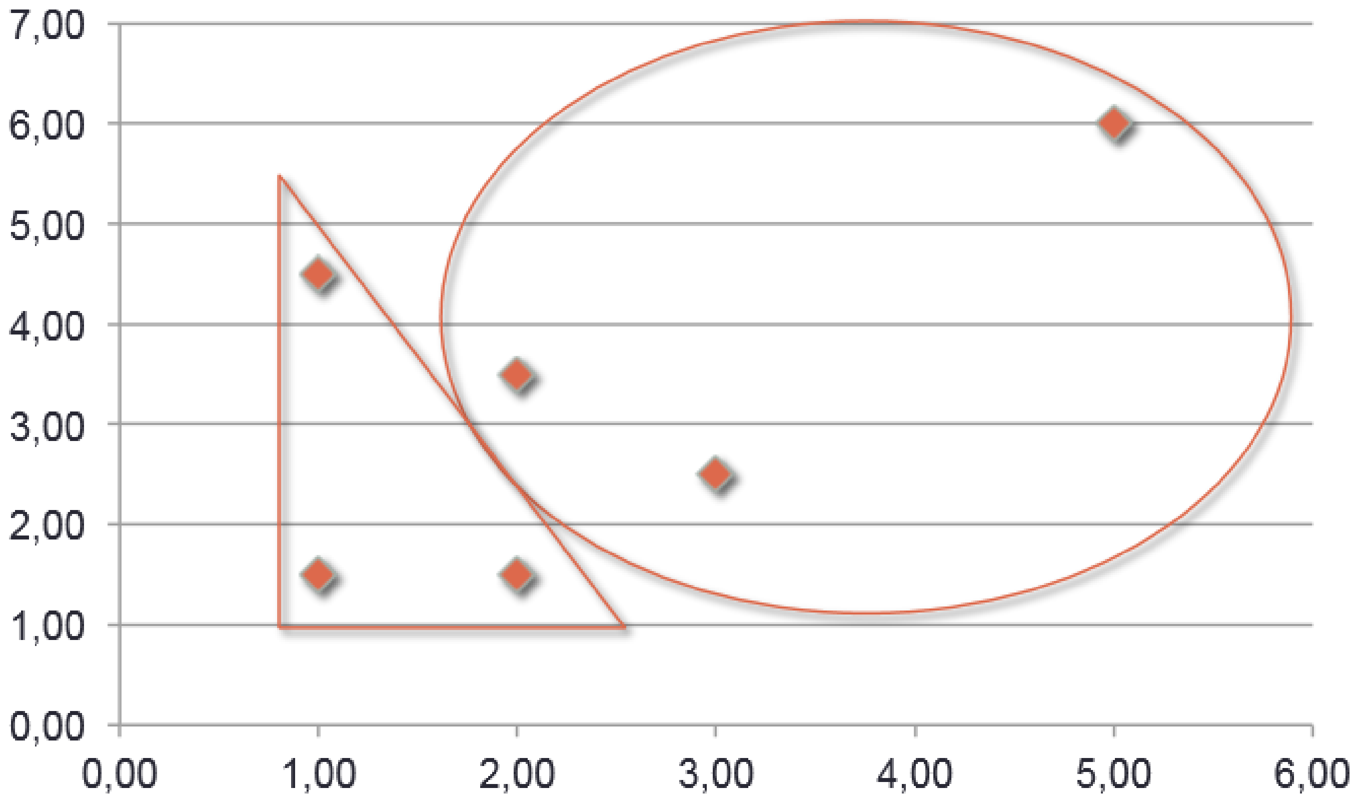
Object	Dist(c_1-point)	Distc_2-point)
1	1.50	2.74
2	1.50	2.29
3	1.80	2.12
4	1.12	1.01
5	2.06	0.87
6	5.00	3.30



$$\text{distance}(A,B) = \sqrt{(x_1-x_2)^2 + (y_1-y_2)^2}$$

K-means - An Example – iteration 2

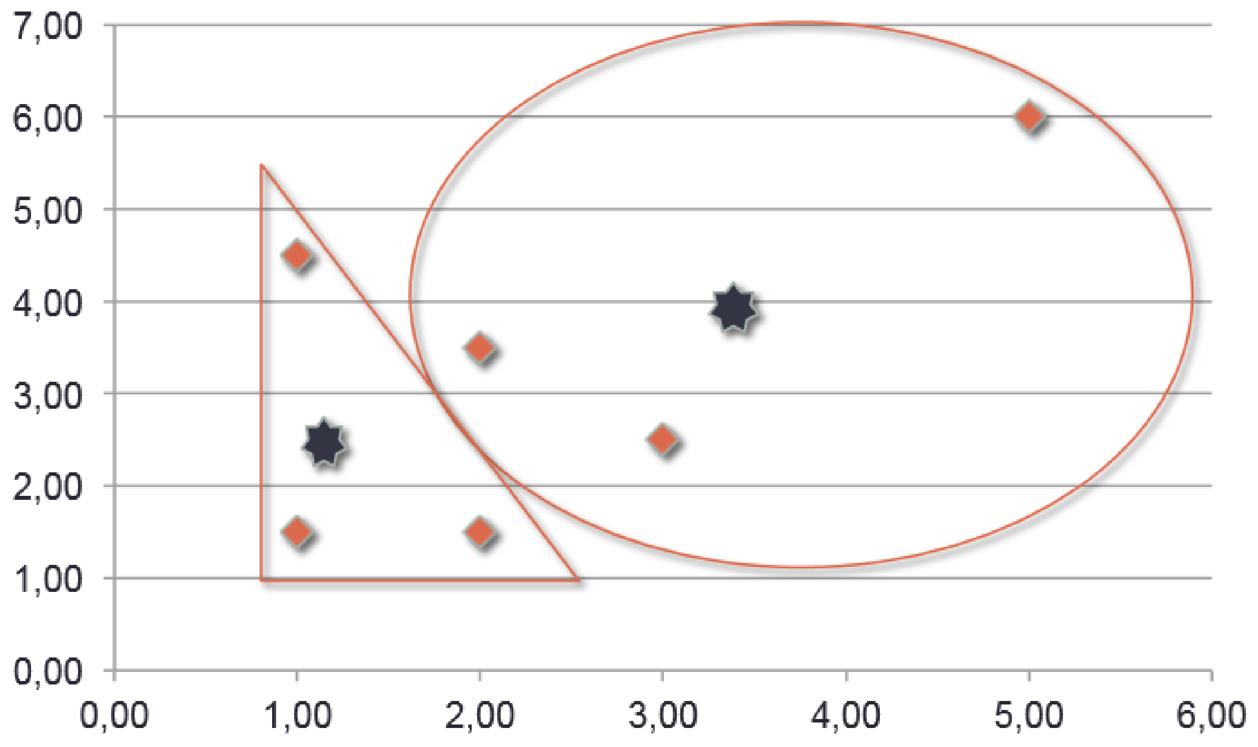
- ITERATION 2
  - C1 = {1, 2, 3}
  - C2= {4, 5, 6}



Clusters after Iteration 2

K-means - An Example – iteration 2

- ITERATION 2
- centroid for C1
  - x=(1+1 +2)/3 = 1.33
  - y= (1.5+4.5+1.5)/3 = 2.50
  - c1= (1.33, 2.50)
- centroid for C2
  - x=(2.0+3.0+5.0)/3 = 3.33
  - y=(3.5+2.5+6.0)/3= 4.00
  - c2=(3.33, 4.00)



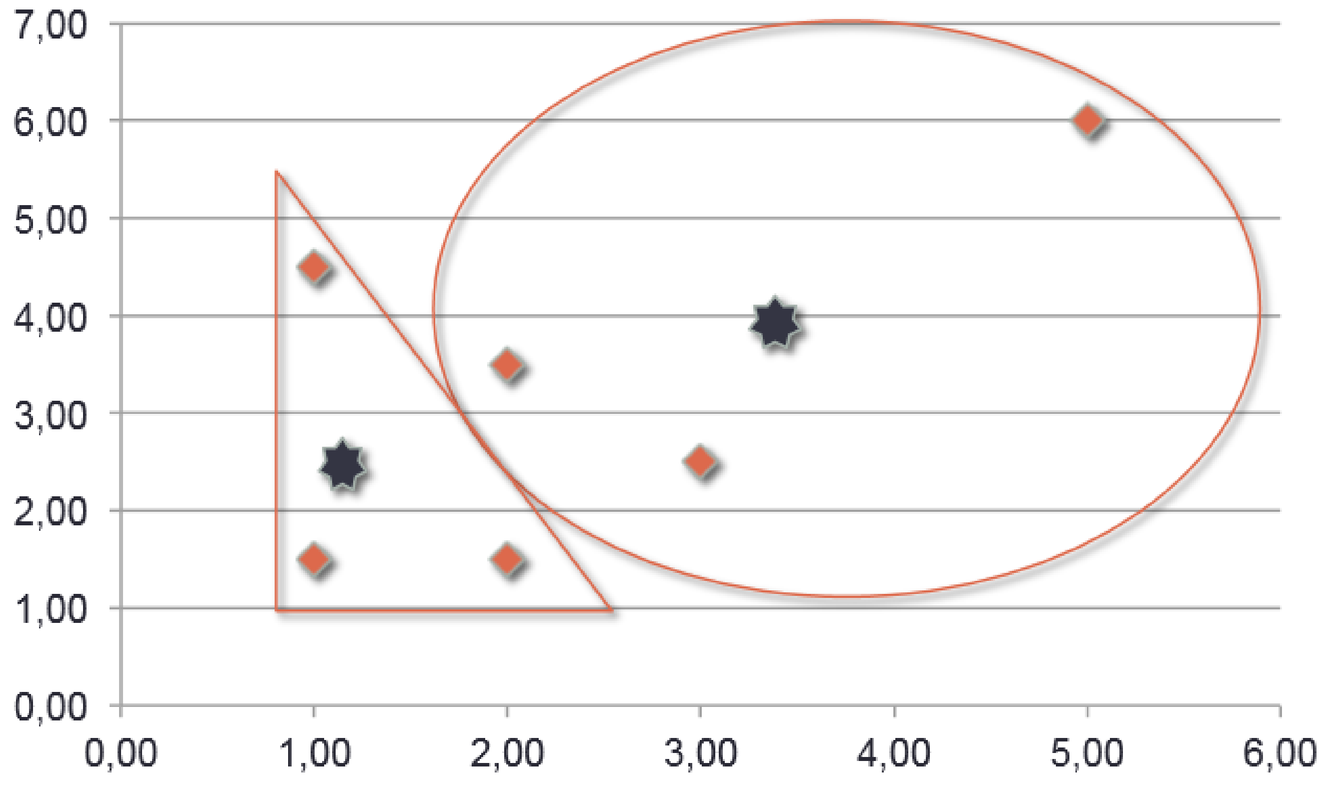
- Exit condition not satisfied

Clusters after Iteration 2

K-means - An Example – iteration 3

- ITERATION 3
  - C1 = {1, 2, 3}
  - C2 = {4, 5, 6}

  - c1= (1.33, 2.50)
  - c2=(3.33, 4.00)

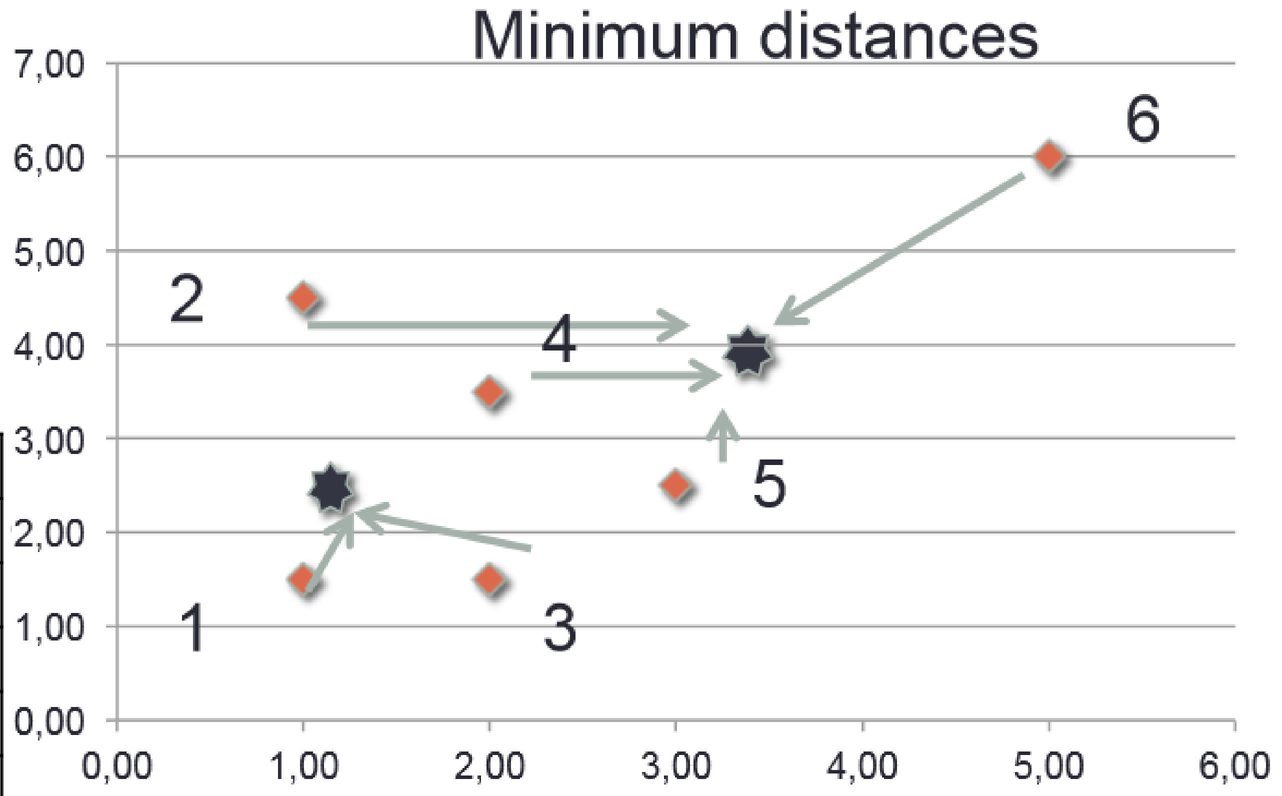


Clusters after the Iteration 2

K-means - An Example – iteration 3

- ITERATION 3 – centroids
  - c1= (1.33, 2.50)
  - c2=(2.75, 4.12)
- Compute distances

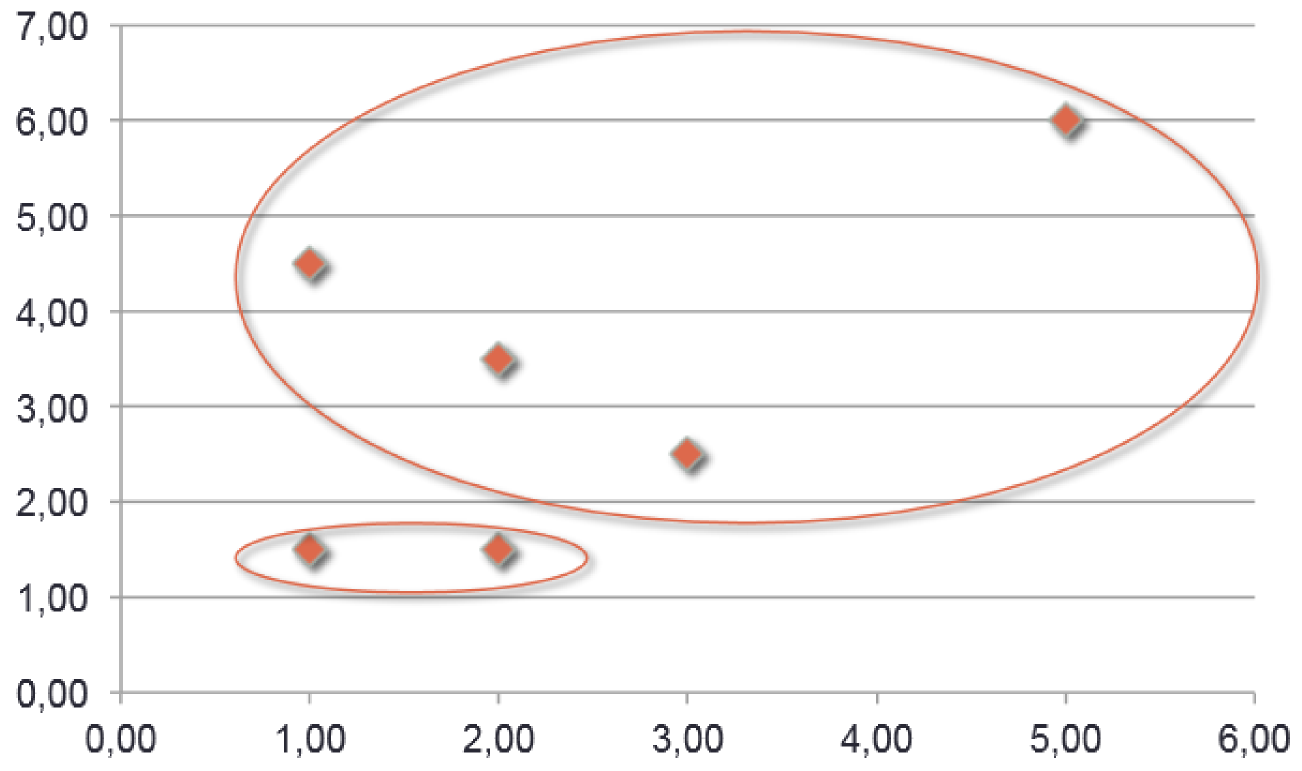
Object	Dist(C1-point)	Dist(C2-point)
1	1,05	2,83
2	2,03	1,42
3	1,20	2,52
4	1,20	0,60
5	1,67	1,64
6	5,07	3,34



$$\text{distance}(A,B) = \sqrt{(x_1-x_2)^2 + (y_1-y_2)^2}$$

K-means - An Example – iteration 3

- ITERATION 3
- $C_1 = \{1, 3\}$
- $C_2 = \{2, 4, 5, 6\}$

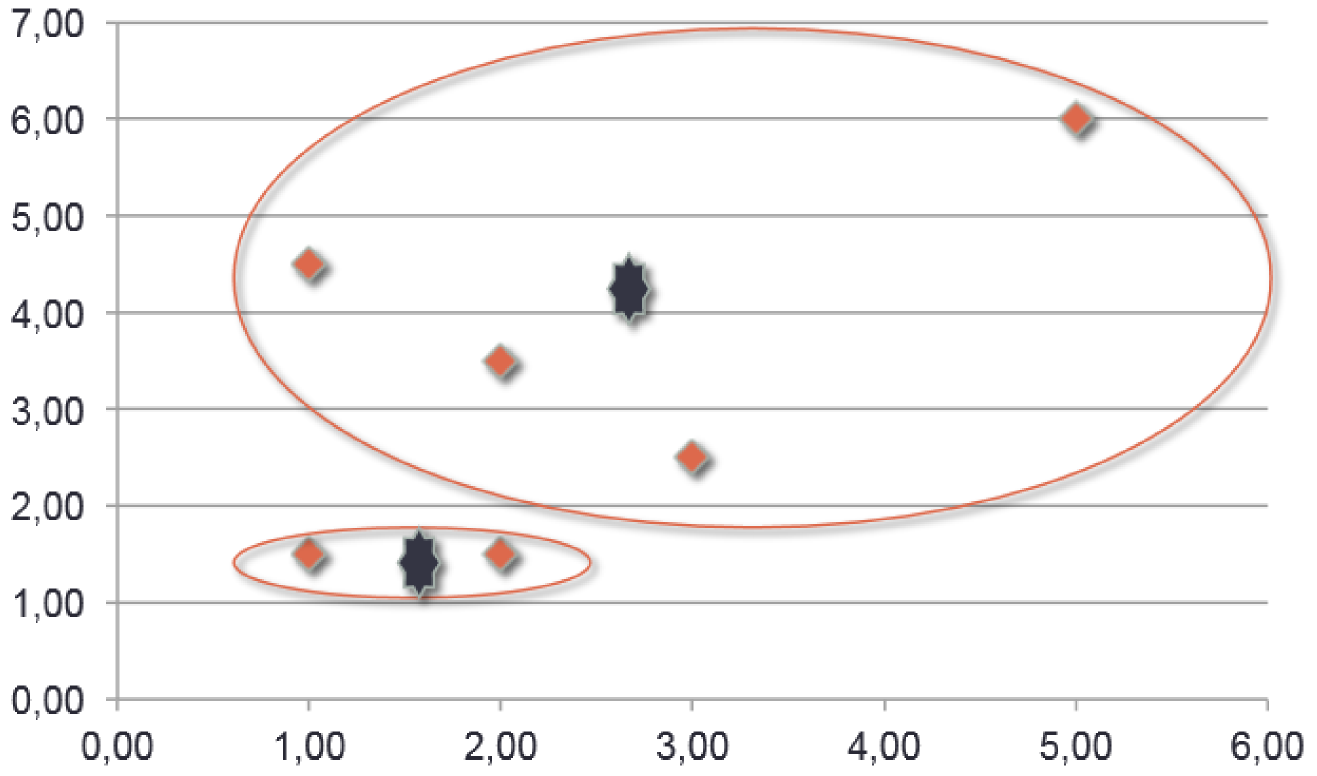


Clusters after Iteration 3

K-means - An Example – iteration 3

- ITERATION 3
- centroid for C1
  - x=(1+2)/2 = 1.5
  - y= (1.5+1.5)/2 = 1.5
  - c1= (1.5, 1.5)
- centroid for C2
  - x=(1+2+3+5)/4 = 2.75
  - y=(4.5+3.5+2.5+6.0)/4= 4.12
  - c2=(2.75, 4.12)

- Exit condition not satisfied

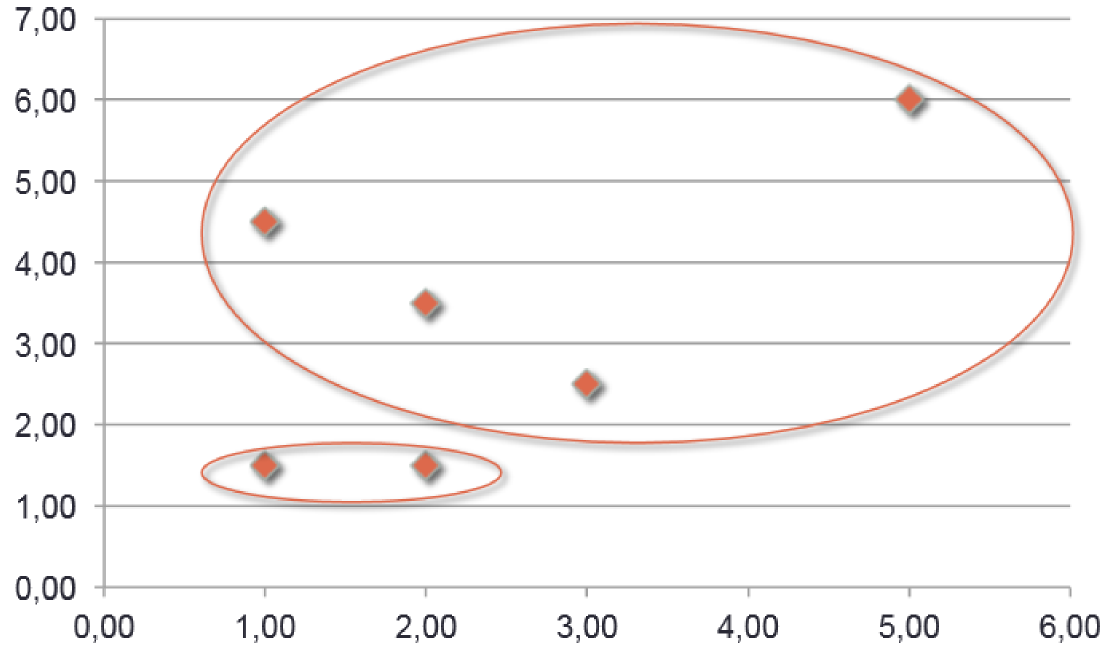


Clusters after Iteration 3

K-means - An Example – iteration 4

- ITERATION 4
  - C1 = {1, 3}
  - C2 = {2 ,4, 5, 6}

  - c1= (1.5, 1.5)
  - c2=(2.75, 4.12)
- ……..
- After iteration 4 we get the same centroids, so the algorithm stops



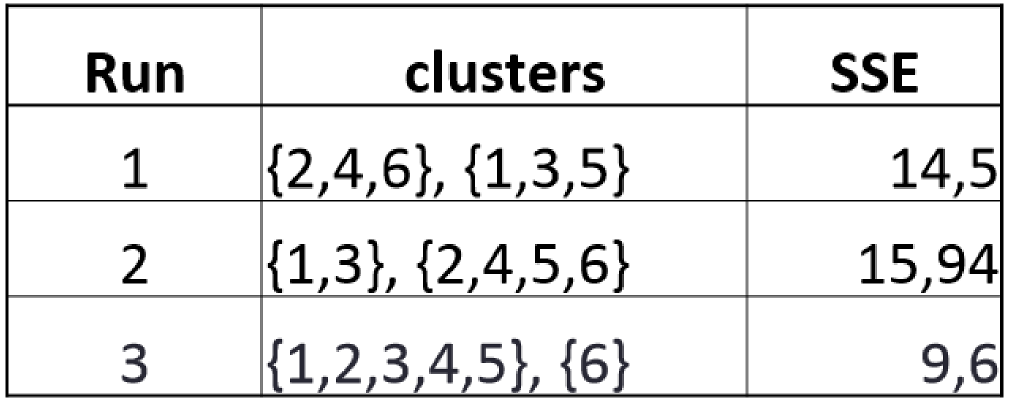
Clusters after Iteration 4

Choosing the best solution

- The value of the objective function for the above solution is

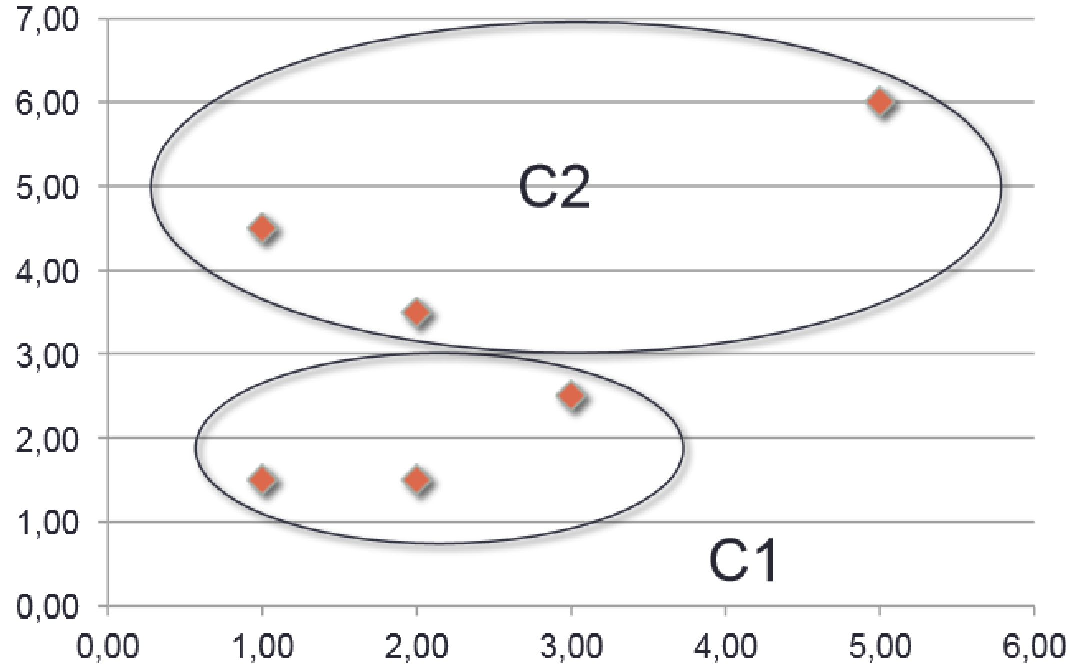
$$SSE = \Sigma_{i=1,2}\ \Sigma_{Xj \in Pi}\ \text{distance}(X_j,C_i) = 15.94$$

- By re-running the algorithm with different choices of the initial centroids, we get different solutions
- The result of K-means is strongly affected by the choice of the initial centroids

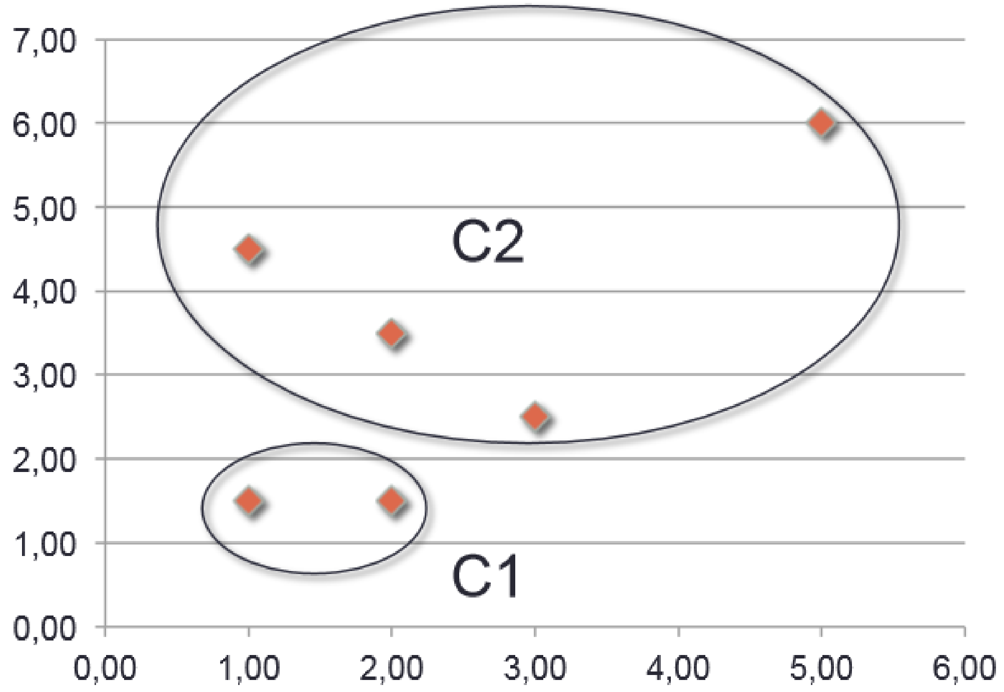


Run	clusters	SSE
1	{2,4,6}, {1,3,5}	14,5
2	{1,3}, {2,4,5,6}	15,94
3	{1,2,3,4,5}, {6}	9,6

Choosing the best solution

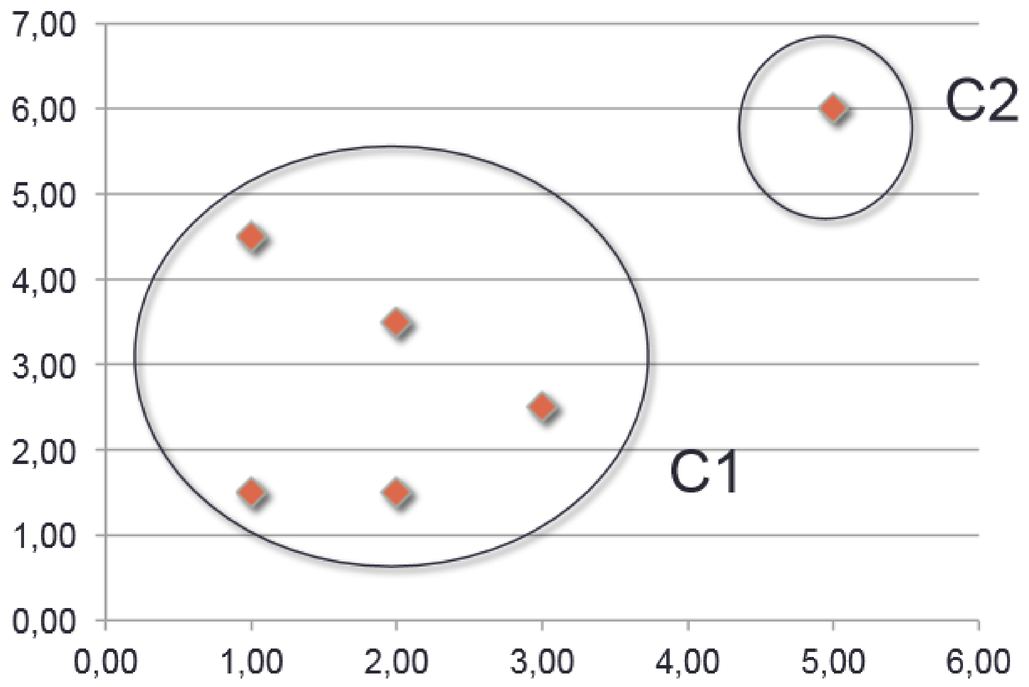


Run 1: SSE=14.5



Run 2: SSE=15.94

Choosing the best solution



Run 3: SSE=9.6

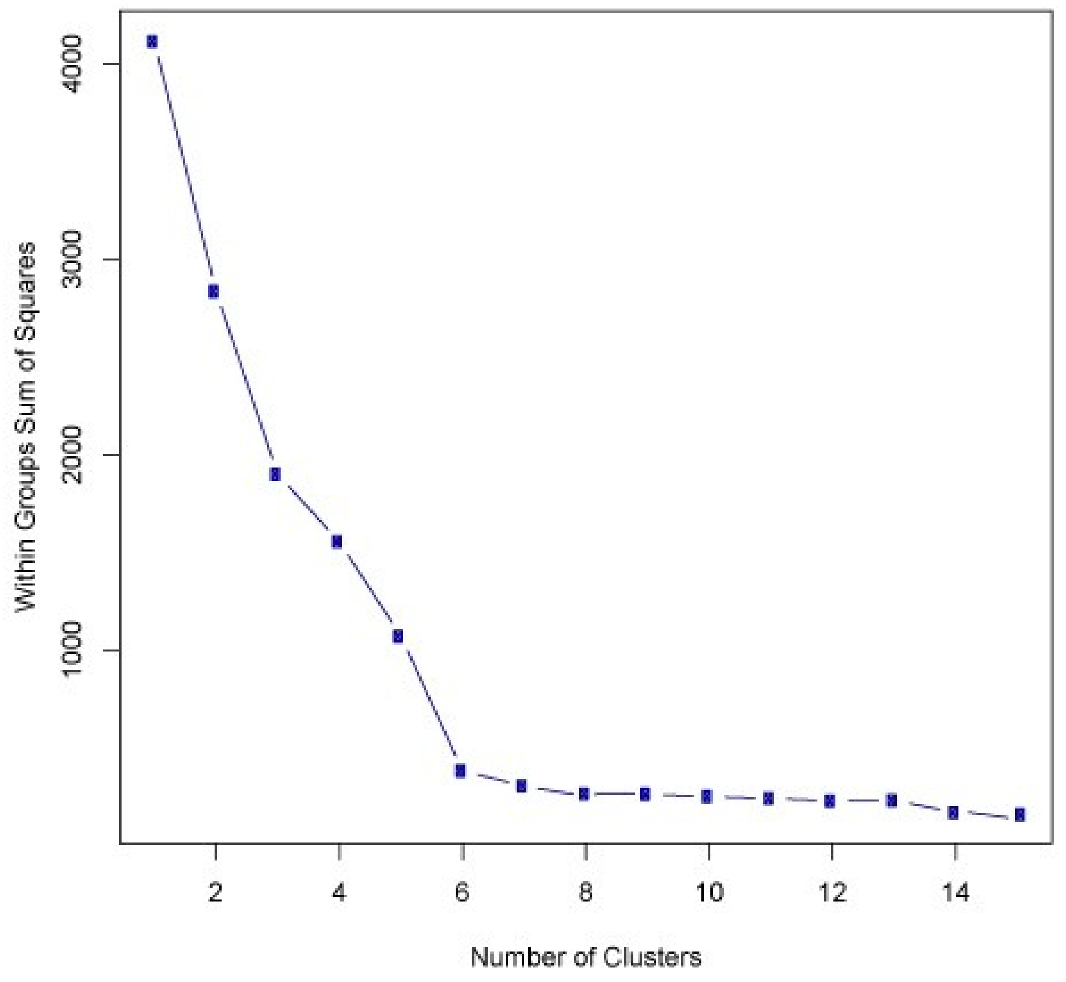
Choosing the best solution

- K-means is very fast
- It is usually run different times, by varying the initial centroids
- The solution with the minimum SSE is chosen

Run	clusters	SSE
1	{2,4,6}, {1,3,5}	14,5
2	{1,3}, {2,4,5,6}	15,94
3	**{1,2,3,4,5}, {6}**	**9,6**

K-means: Choosing the number of clusters

- Common approach is to choose by hand (we don't have labels)
- Elbow method
  - Plot error for different values of K
  - Look for Elbow shape
  - Why is error decreasing as K increases?
- Look at the down stream purpose
  - Image compression (#clusters: Tradeoff between size of the image and quality)
  - T-shirt (3 sizes vs 5 sizes: Tradeoff between reduction of cost vs providing better fit)



Source: https://www.quora.com/How-can-we-choose-a-good-K-for-K-means-clustering

K-means and document data

- Documents are represented as a document-term matrix
- Given two vectors of attributes, A and B, the cosine similarity, cos(θ), is represented using a dot product and magnitude as

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

- Since term frequencies are positive, cos(θ) ranges from 0 to 1, with 1 meaning exactly the same

K-means and document data

	w_1	w_2	w_3	w_4	w_5	**Class**
d1	0	1	1	1	0	Sport, politics
d2	0	0	1	1	1	gossip
d3	1	0	0	1	0	Sport, gossip
d4	1	0	0	1	0	politics

$$\text{CosSim}(d_1,d_2) = \frac{\Sigma\, d_1(i) \times d_2(i)}{\sqrt{\ \Sigma\, d_1(i)^2} \times \sqrt{\ \Sigma\, d_2(i)^2}}$$

$$\text{CosSim}(d_1,d_2) = \frac{2}{\sqrt{3} \times \sqrt{3}} = 0.66$$

$$\text{CosSim}(d_3,d_4) = \frac{2}{\sqrt{2} \times \sqrt{2}} = 1 \text{ (identical)}$$

Pros and Cons

Strength	• K-means is simple and efficient, so as multiple runs can be performed
Weaknesses	• K-means is restricted to data for which there is a notion of centroid • K is to be specified in advance • Outliers can lower the quality of resulting clusters