K-Means

Oct. 2015

Youn-Hee Han

http://link.koreatech.ac.kr

Introduction

- K-Means algorithm
  - An unsupervised clustering algorithm
  - "K" stands for number of clusters.
    - It is typically a user input to the algorithm
    - Some criteria can be used to automatically estimate K
  - It is an approximation to an NP-hard combinatorial optimization problem
    - In how many ways can we assign K labels to N observations?
  - K-means algorithm is iterative in nature
  - Works only for numerical data
  - Easy to implement

Introduction

- Partitioning Clustering Approach
  - a typical clustering analysis approach via iteratively partitioning training data set
  - in principle, optimal partition achieved via minimizing the sum of squared distance to its "representative object, m_k" in each cluster

$$E = \sum_{k=1}^{K} \sum_{\mathbf{x} \in C_k} d^2(\mathbf{x}, \mathbf{m}_k)$$

e.g., Euclidean distance $d^2(\mathbf{x}, \mathbf{m}_k) = \sum_{n=1}^{N} (x_n - m_{kn})^2$

Introduction

- Given a K, find a partition of K clusters to optimize the chosen partitioning criterion (cost function)
  - global optimum: exhaustively search all partitions

- The K-means algorithm: a heuristic method
  - K-means algorithm (MacQueen'67): each cluster is represented by the centre of the cluster
  - It converges to stable centriods of clusters.
  - It is the simplest partitioning method for clustering analysis and widely used in data mining applications.

K-means algorithm

- Algorithm
  - Given the cluster number K and N observations, the K-means algorithm is carried out in three steps after initialization:

  - Initialization: set seed points (randomly)
  - 1) Assign each object to the cluster of the nearest seed point measured with a specific distance metric

$$C(i) = \underset{1 \le k \le K}{\operatorname{argmin}} \left\| x_i - m_k \right\|^2, \; i = 1, \ldots, N$$

K-means algorithm

- Algorithm
  - 2) Compute seed points as the centroids of the clusters of the current partition
    - the centroid is the centre, i.e., mean point, of the cluster

$$m_k = \frac{\sum_{i:C(i)=k} x_i}{N_k}, \; k = 1, \ldots, K.$$

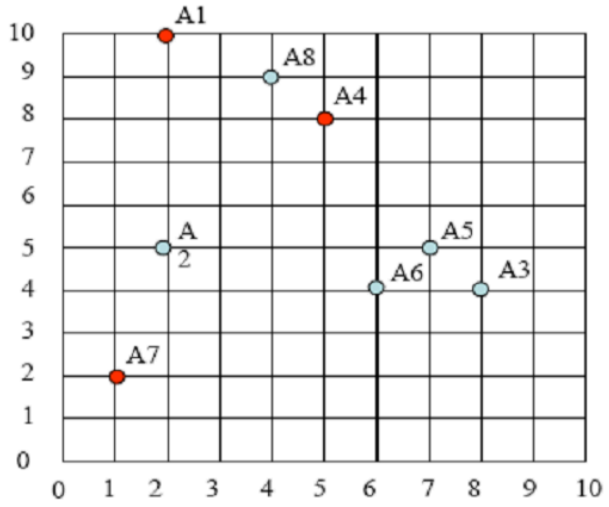
  - 3) Go back to Step 1), stop when no more new assignment (i.e., membership in each cluster no longer changes)

Example

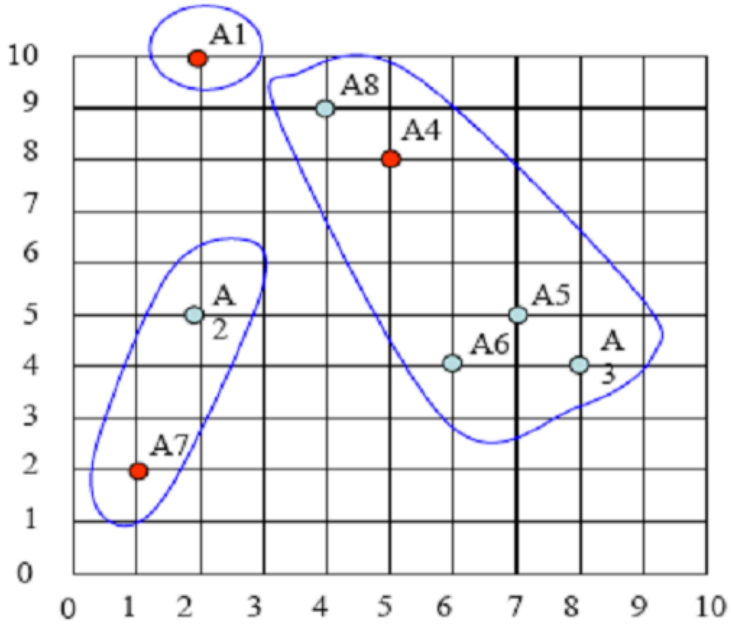
◈Problem

- Cluster the following eight points (with (x, y) representing locations) into three clusters.
  - A1(2, 10), A2(2, 5), A3(8, 4), A4(5, 8), A5(7, 5), A6(6, 4), A7(1, 2), A8(4, 9).
  - Initial cluster centers are: A1(2, 10), A4(5, 8) and A7(1, 2).
  - The distance function between two points a=(x1, y1) and b=(x2, y2) is defined as:

$$\rho(a, b) = |x2 - x1| + |y2 - y1|$$



Example



◈Solution

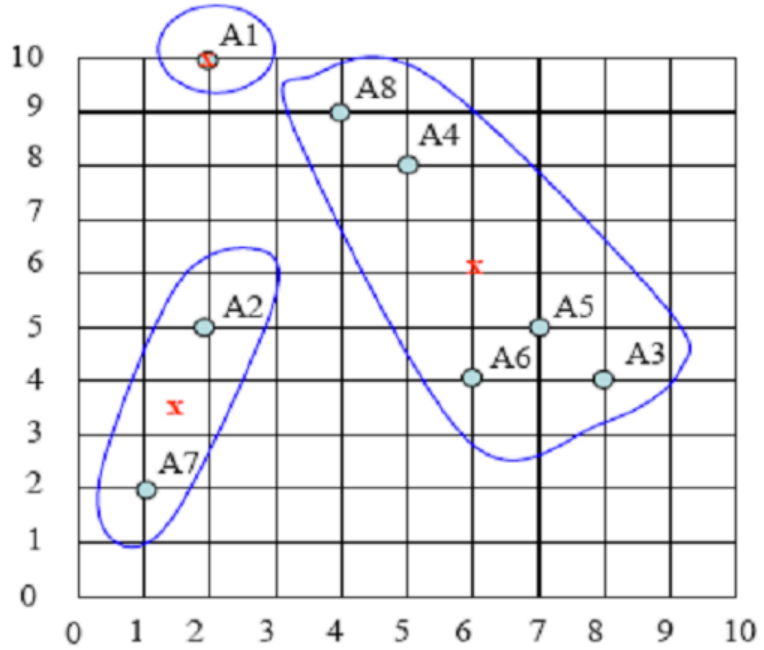
– Iteration 1

		(2, 10)	(5, 8)	(1, 2)	
	Point	**Dist Mean 1**	**Dist Mean 2**	**Dist Mean 3**	**Cluster**
A1	(2, 10)				
A2	(2, 5)				
A3	(8, 4)				
A4	(5, 8)				
A5	(7, 5)				
A6	(6, 4)				
A7	(1, 2)				
A8	(4, 9)				

Example

Solution

- Next, we need to re-compute the new cluster centers (means). We do so, by taking the mean of all points in each cluster.
- For Cluster 1, we only have one point A1(2, 10), which was the old mean, so the cluster center remains the same.
- For Cluster 2, we have ((8+5+7+6+4)/5, (4+8+5+4+9)/5) = (6, 6)
- For Cluster 3, we have ((2+1)/2, (5+2)/2) = (1.5, 3.5)



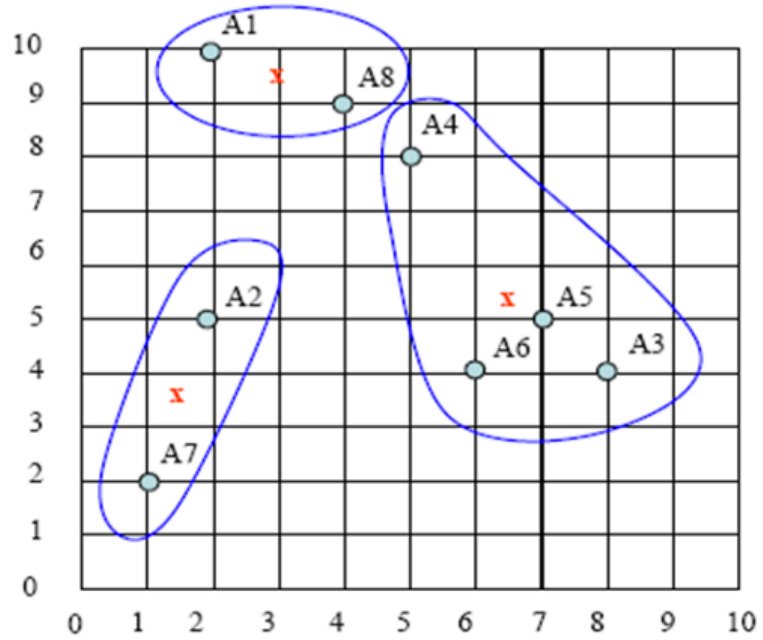
Example

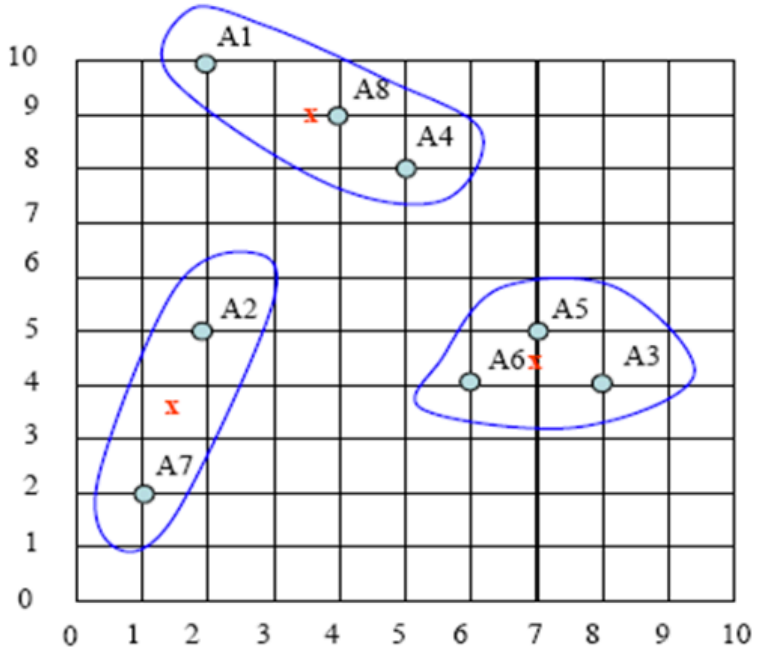
- Solution
  - Next, we go to Iteration2 (epoch2), Iteration3, and so on until the means do not change anymore.
  - In Iteration2, we basically repeat the process from Iteration1 this time using the new means we computed.

Example

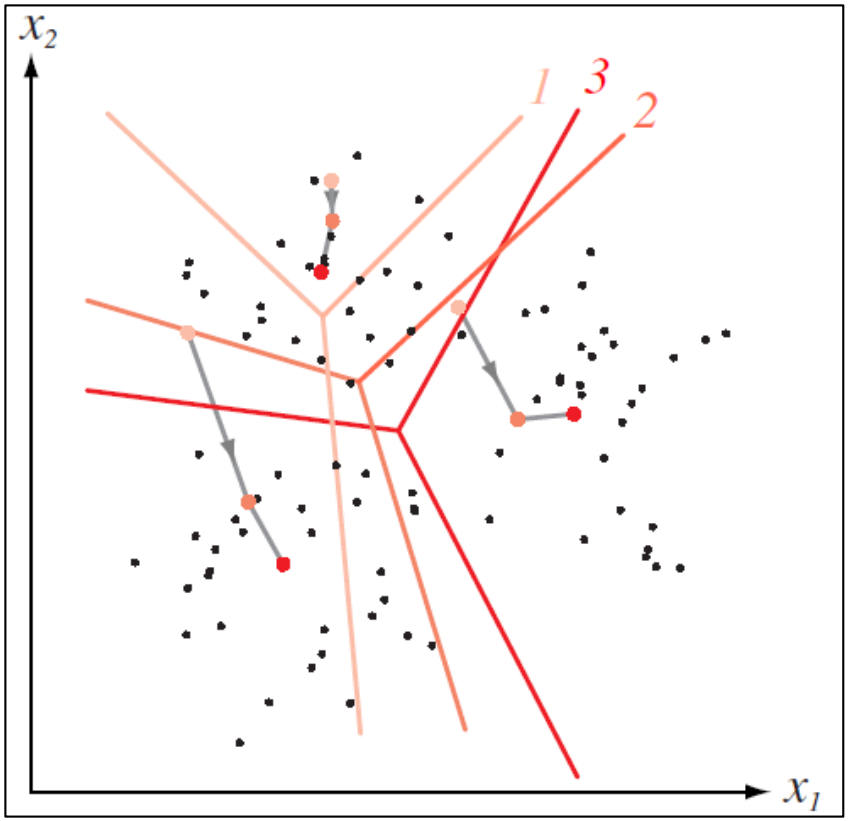
◈Solution

We would need two more epochs. After the 2nd epoch the results would be:
1: {A1, A8}, 2: {A3, A4, A5, A6}, 3: {A2, A7}
with centers C1=(3, 9.5), C2=(6.5, 5.25) and C3=(1.5, 3.5).
After the 3rd epoch, the results would be:
1: {A1, A4, A8}, 2: {A3, A5, A6}, 3: {A2, A7}
with centers C1=(3.66, 9), C2=(7, 4.33) and C3=(1.5, 3.5).



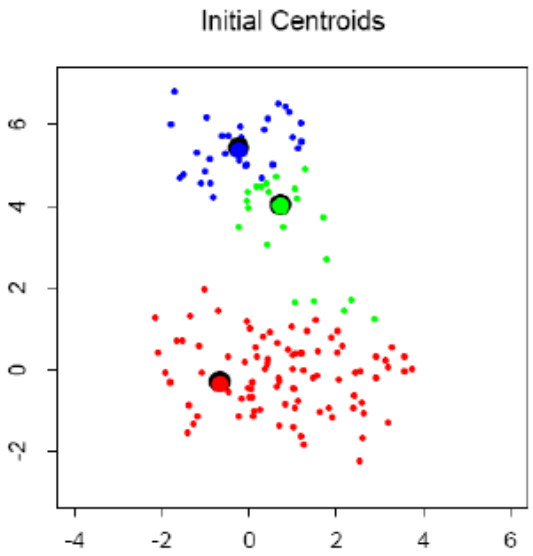


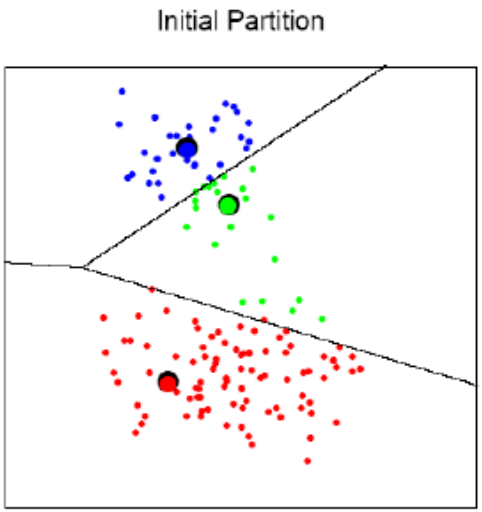
How K-means partitions?

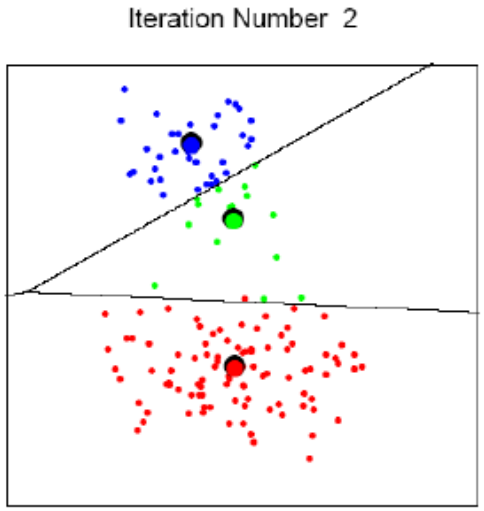


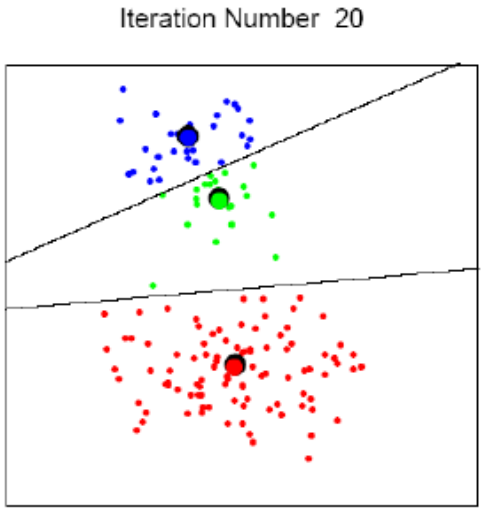
- When K centroids are set/fixed, they partition the whole data space into K mutually exclusive subspaces to form a partition.
- A partition amounts to a Voronoi Diagram
- Changing positions of centroids leads to a new partitioning.

How K-means partitions?

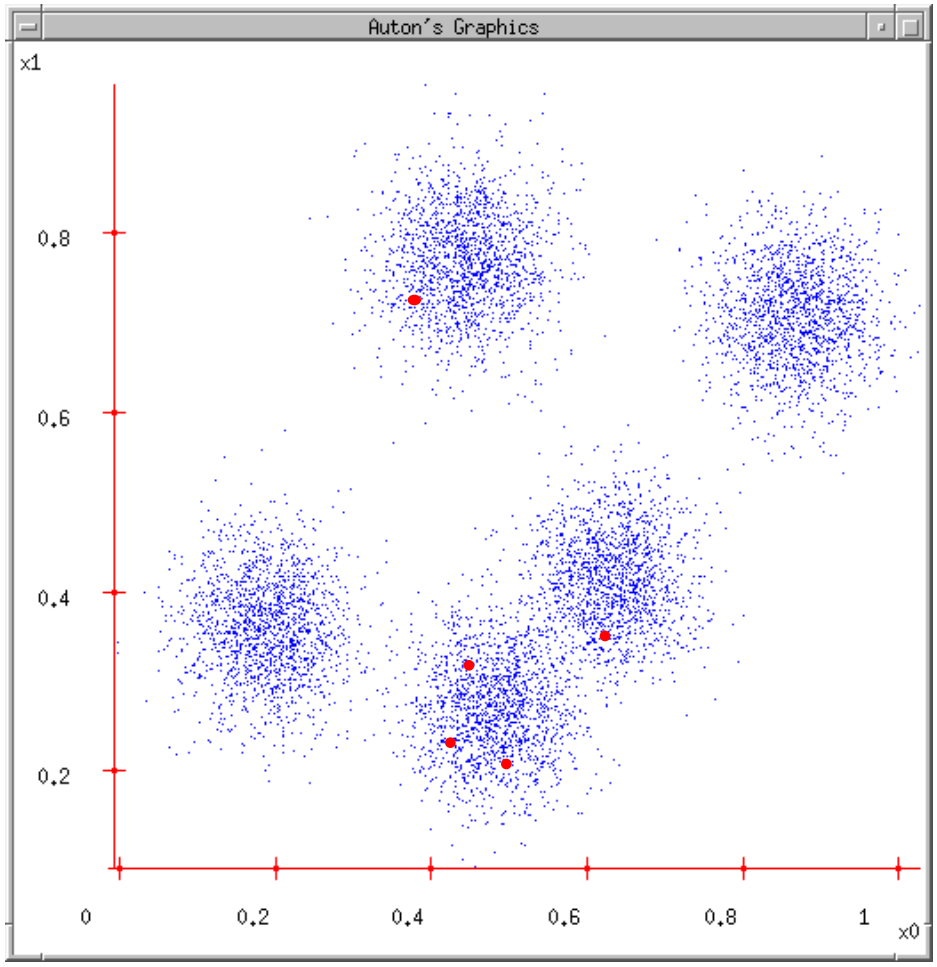






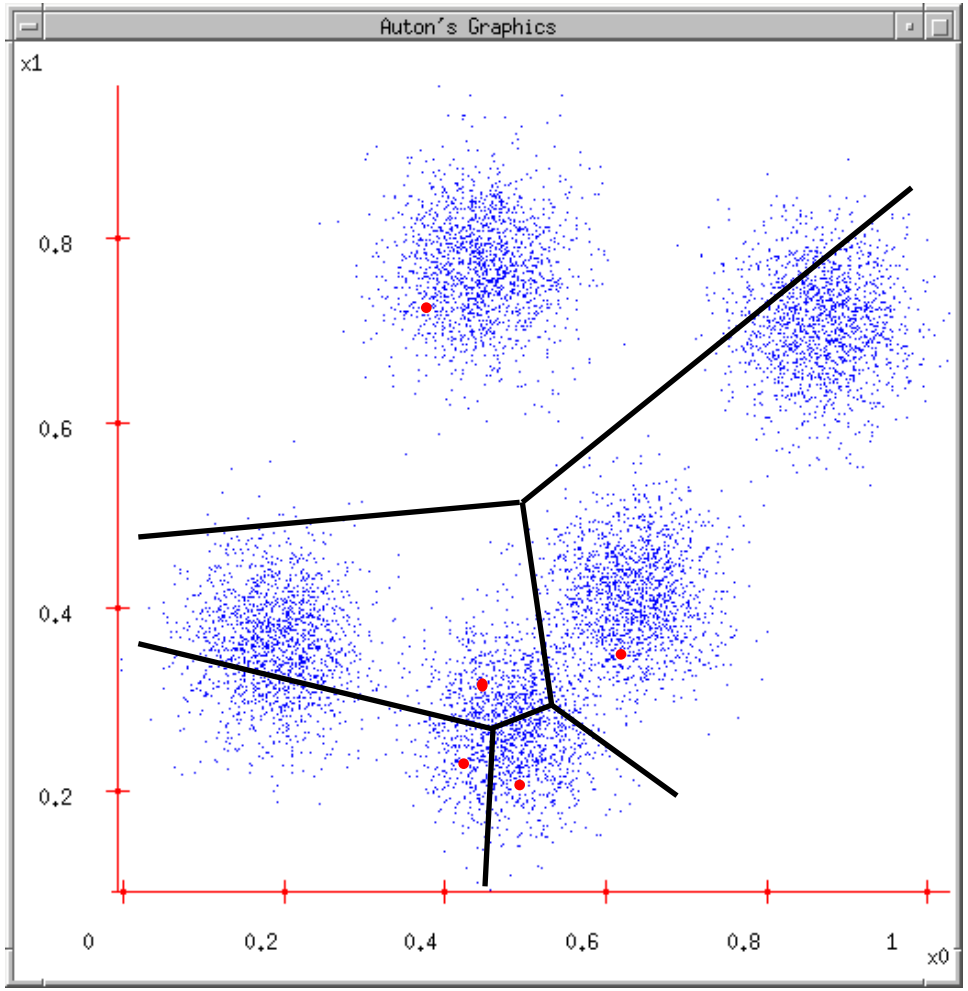


K-means Demo



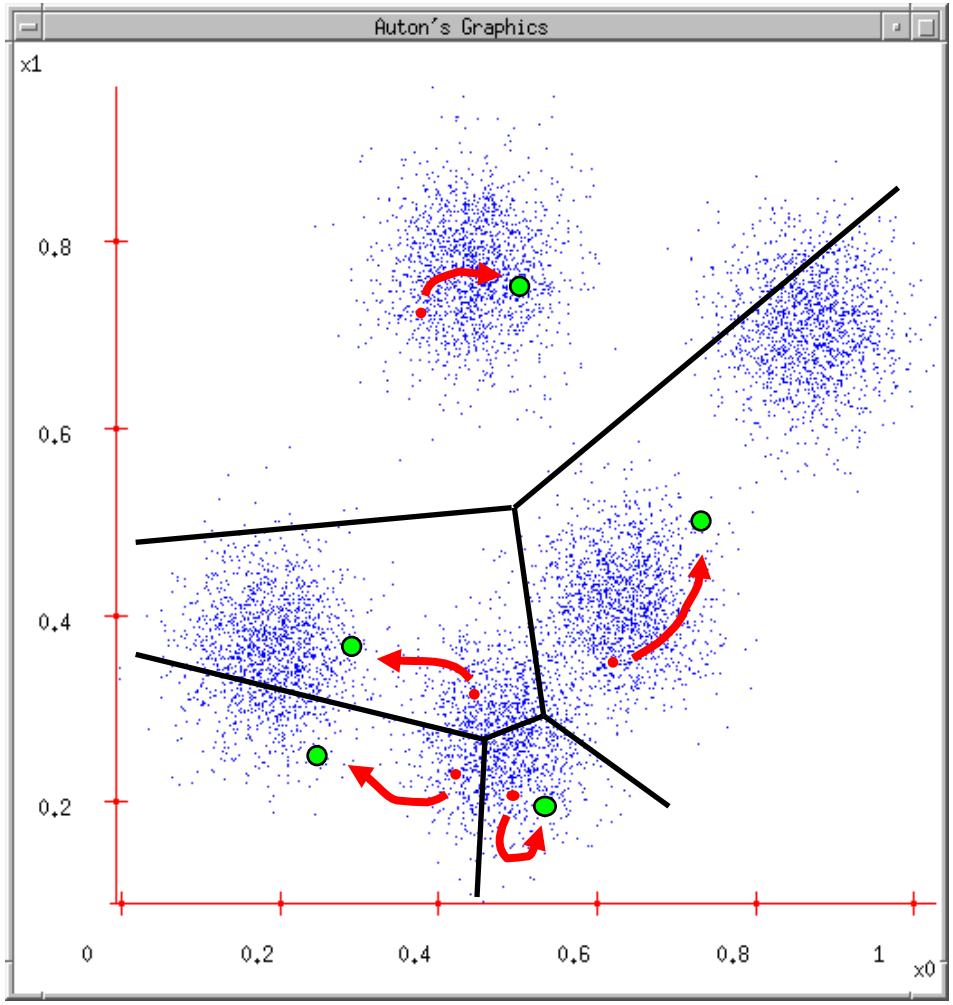
1. User set up the number of clu sters they'd like. *(e.g. K=5)*
2. Randomly guess K cluster Ce nter locations

K-means Demo



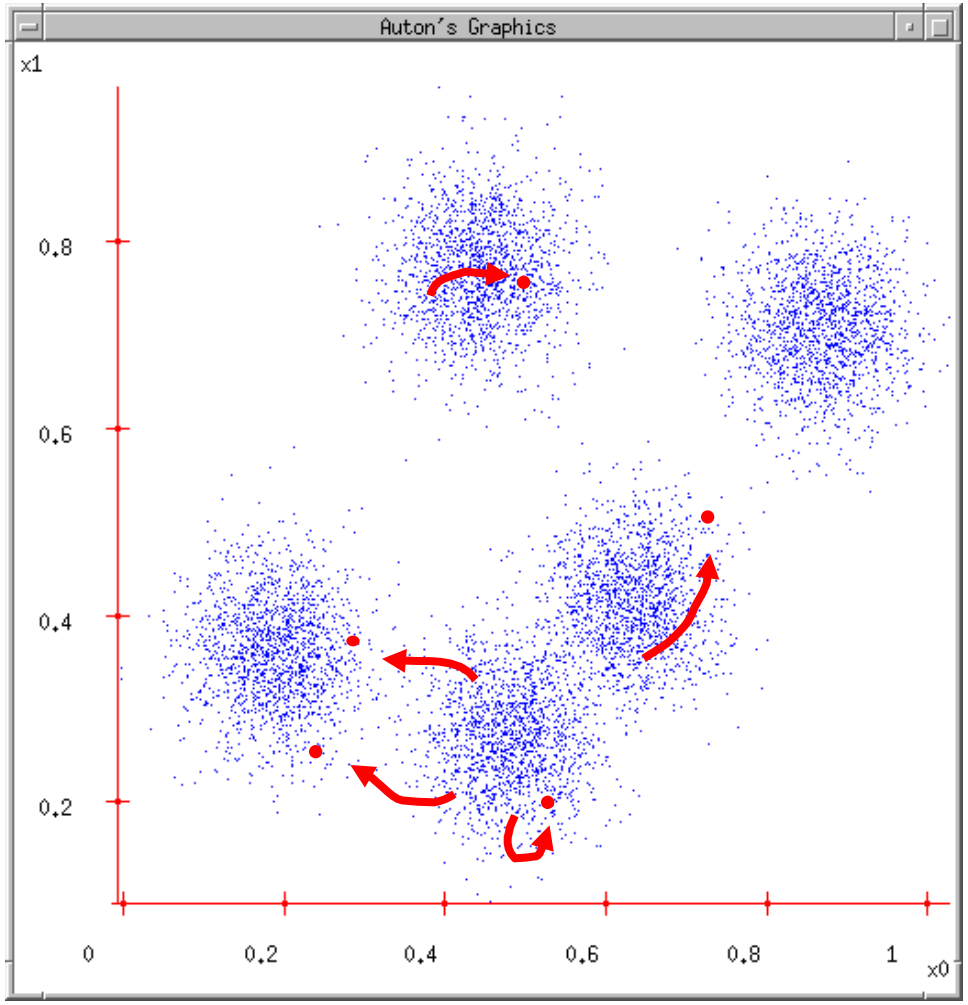
1. User set up the number of clu sters they'd like. *(e.g. K=5)*
2. Randomly guess *K* cluster Ce nter locations
3. Each data point finds out whi ch Center it's closest to. (Thu s each Center "owns" a set of data points)

K-means Demo



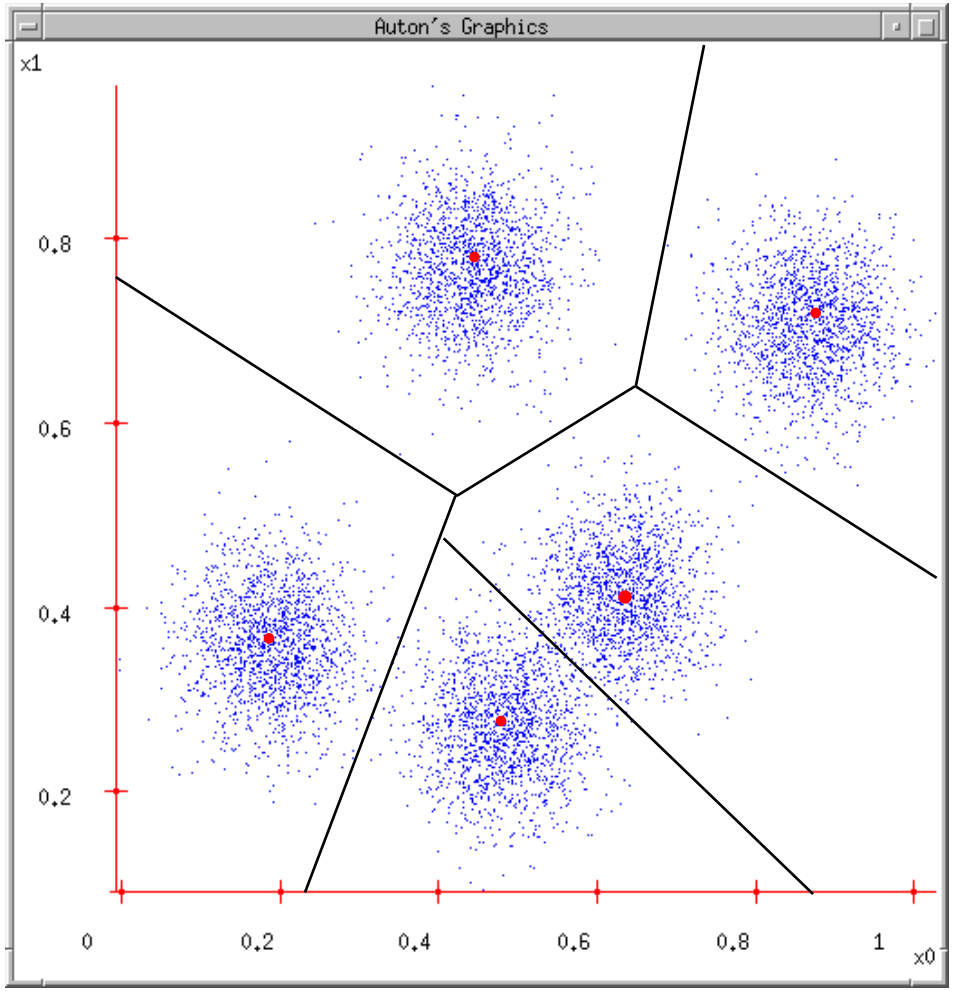
1. User set up the number of clu sters they'd like. *(e.g. K=5)*
2. Randomly guess *K* cluster cen tre locations
3. Each data point finds out whi ch centre it's closest to. (Thus each Center "owns" a set of d ata points)
4. Each centre finds the centroid of the points it owns

K-means Demo



1. User set up the number of clu sters they'd like. (e.g. *K=5*)
2. Randomly guess *K* cluster cen tre locations
3. Each data point finds out whi ch centre it's closest to. (Thus each centre "owns" a set of d ata points)
4. Each centre finds the centroid of the points it owns
5. …and jumps there

K-means Demo



1. User set up the number of clu sters they'd like. (e.g. *K=5*)
2. Randomly guess *K* cluster cen tre locations
3. Each data point finds out whi ch centre it's closest to. (Thus each centre "owns" a set of d ata points)
4. Each centre finds the centroid of the points it owns
5. …and jumps there
6. …Repeat until terminated!

Relevant Issues

- Efficient in computation
  - O(tKn), where n is number of objects, K is number of clusters, and t is number of iterations. Normally, K, t << n.

- Local optimum
  - sensitive to initial seed points
  - converge to a local optimum: maybe an unwanted solution

Relevant Issues

- Other problems
  - Need to specify K, the number of clusters, in advance
  - Unable to handle noisy data and outliers
    - K-Medoids algorithm
  - Not suitable for discovering clusters with non-convex shapes
  - Applicable only when mean is defined, then what about categorical data?
    - K-mode algorithm

Summary

- K-means algorithm is a simple yet popular method for clustering analysis
- Its performance is determined by initialization and appropriate distance measure
- There are several variants of K-means to overcome its weaknesses
  - K-Medoids: resistance to noise and/or outliers
  - K-Modes: extension to categorical data clustering analysis
  - CLARA: extension to deal with large data sets
  - Mixture models (EM algorithm): handling uncertainty of clusters