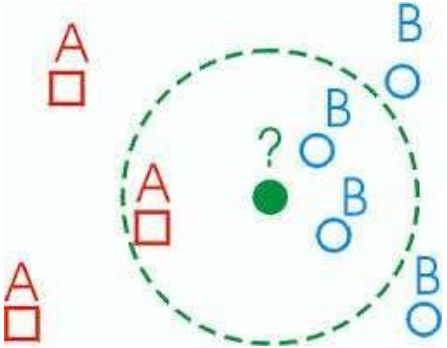
KNN and K-means

- K-means is an unsupervised learning algorithm used for **clustering** problem whereas **KNN** is a supervised learning algorithm used for classification and regression problem.

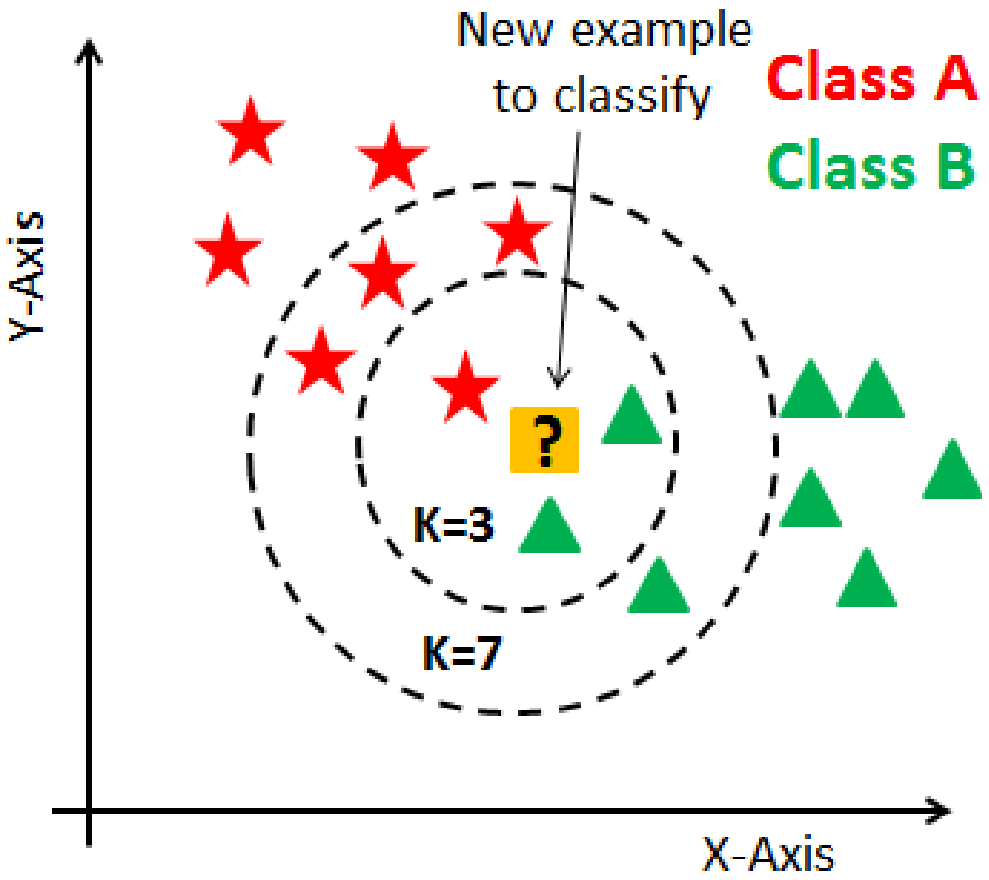
KNN: Classification Approach

- An object (a new instance) is classified by a majority votes for its neighbor classes.
- The object is assigned to the most common class amongst its K nearest neighbors.(*measured by a distant function*)



1. Calculate distance
2. Find closest neighbors
3. Vote for labels

KNN



K-Nearest Neighbor Algorithm

1. All the instances correspond to points in an n-dimensional feature space.
2. Each instance is represented with a set of numerical attributes.
3. Each of the training data consists of a set of vectors and a class label associated with each vector.
4. Classification is done by comparing feature vectors of different K nearest points.
5. Select the K-nearest examples to E in the training set.
6. Assign E to the most common class among its K-nearest neighbors.

KNN: Example

Customer	Age	Income	No. credit cards	Class
George	35	35K	3	No
Rachel	22	50K	2	Yes
Steve	63	200K	1	No
Tom	59	170K	1	No
Anne	25	40K	4	Yes
John	37	50K	2	YES

Distance from John
sqrt $[(35-37)^2+(35-50)^2+(3-2)^2]=15.16$
sqrt $[(22-37)^2+(50-50)^2+(2-2)^2]=15$
sqrt $[(63-37)^2+(200-50)^2+(1-2)^2]=152.23$
sqrt $[(59-37)^2+(170-50)^2+(1-2)^2]=122$
sqrt $[(25-37)^2+(40-50)^2+(4-2)^2]=15.74$

How to choose K?

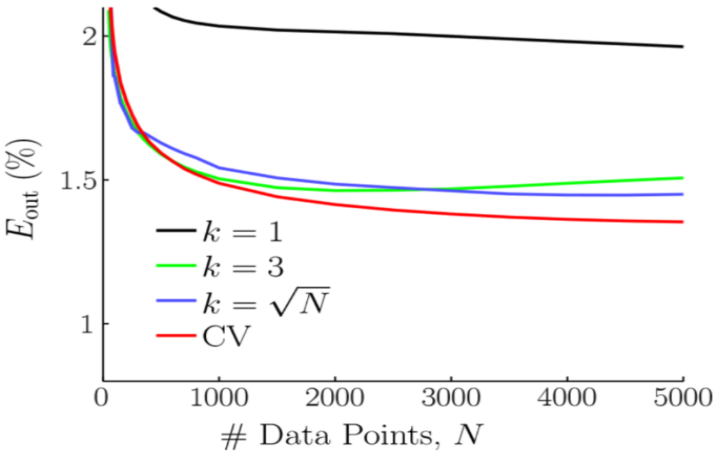
3 ways to choose k

1. $k = 3$

2. $k = \lfloor \sqrt{N} \rfloor$

3. Validation or cross validation

   k-NN rule hypotheses g^- constructed on training set, tested on validation set, and best k is picked



Feature Normalization

- Distance between neighbors could be dominated by some attributes with relatively large numbers.
- Arises when two features are in different scales.
- Important to normalize those features.

$$a_i = \frac{v_i - \min v_i}{\max v_i - \min v_i}$$

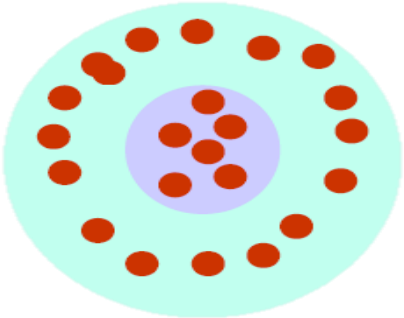
- – Mapping values to numbers between 0 – 1.

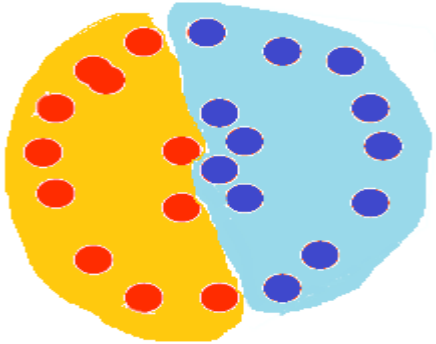
KNN

- **Strengths of KNN**
  - Simple and intuitive.
  - Can be applied to the data from any distribution.
  - Good classification if the number of samples is large enough.
- **Weaknesses of KNN**
  - Takes more time to classify a new example.
  - need to calculate and compare distance from new example to all other examples.
  - Choosing k may be tricky.

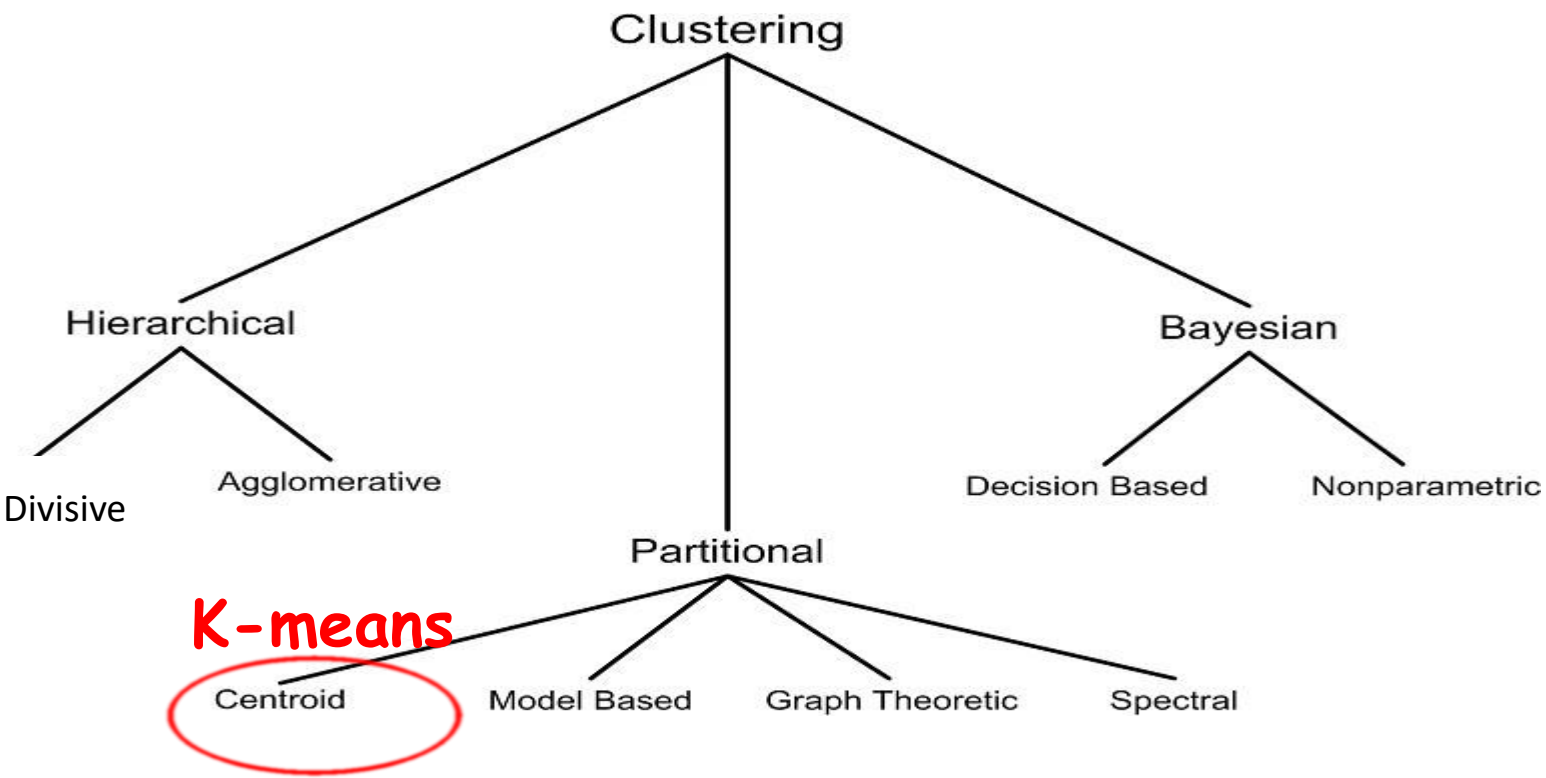
What is clustering?

- The organization of unlabeled data into similarity groups called clusters.
- A cluster is a collection of data items which are “similar” between them, and “dissimilar” to data items in other clusters.





Clustering techniques



Clustering techniques

- **Hierarchical** algorithms find successive clusters using previously established clusters. These algorithms can be either agglomerative ("*bottom-up*") or divisive ("*top-down*"):
  1. Agglomerative algorithms begin with each element as a separate cluster and merge them into successively larger clusters;
  2. Divisive algorithms begin with the whole set and proceed to divide it into successively smaller clusters.
- **Partitional** algorithms typically determine all clusters at once, but can also be used as divisive algorithms in the hierarchical clustering.
- **Bayesian** algorithms try to generate a *posteriori distribution* over the collection of all partitions of the data.

K-Means clustering

- K-means (MacQueen, 1967) is a partitional clustering algorithm
- Let the set of data points D be $\{\mathbf{x}_1, \mathbf{x}_2, \ldots, \mathbf{x}_n\}$,

  where $\mathbf{x}_i = (x_{i1}, x_{i2}, \ldots, x_{ir})$ is a vector in X □ R^r, and r is the number of dimensions.
- The k-means algorithm partitions the given data into k clusters:
  - Each cluster has a cluster **center**, called **centroid**.
  - k is specified by the user

K-means algorithm

- Given *k*, the *k-means* algorithm works as follows:
  1. Choose *k* (random) data points (seeds) to be the initial centroids, cluster centers
  2. Assign each data point to the closest centroid
  3. Re-compute the centroids using the current cluster memberships
  4. If a convergence criterion is not met, repeat steps 2 and 3

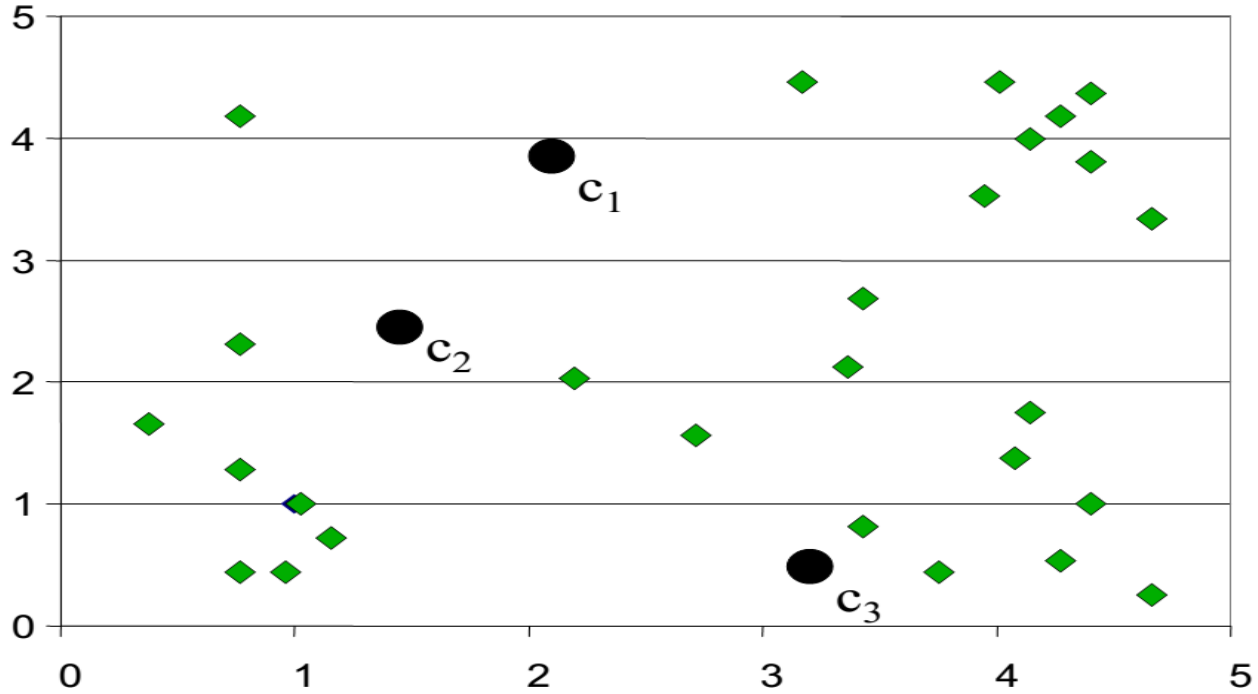
K-means convergence (stopping) criterion

- no (or minimum) re-assignments of data points to different clusters, *or*
- no (or minimum) change of centroids, *or*
- minimum decrease in the **sum of squared error** (SSE),

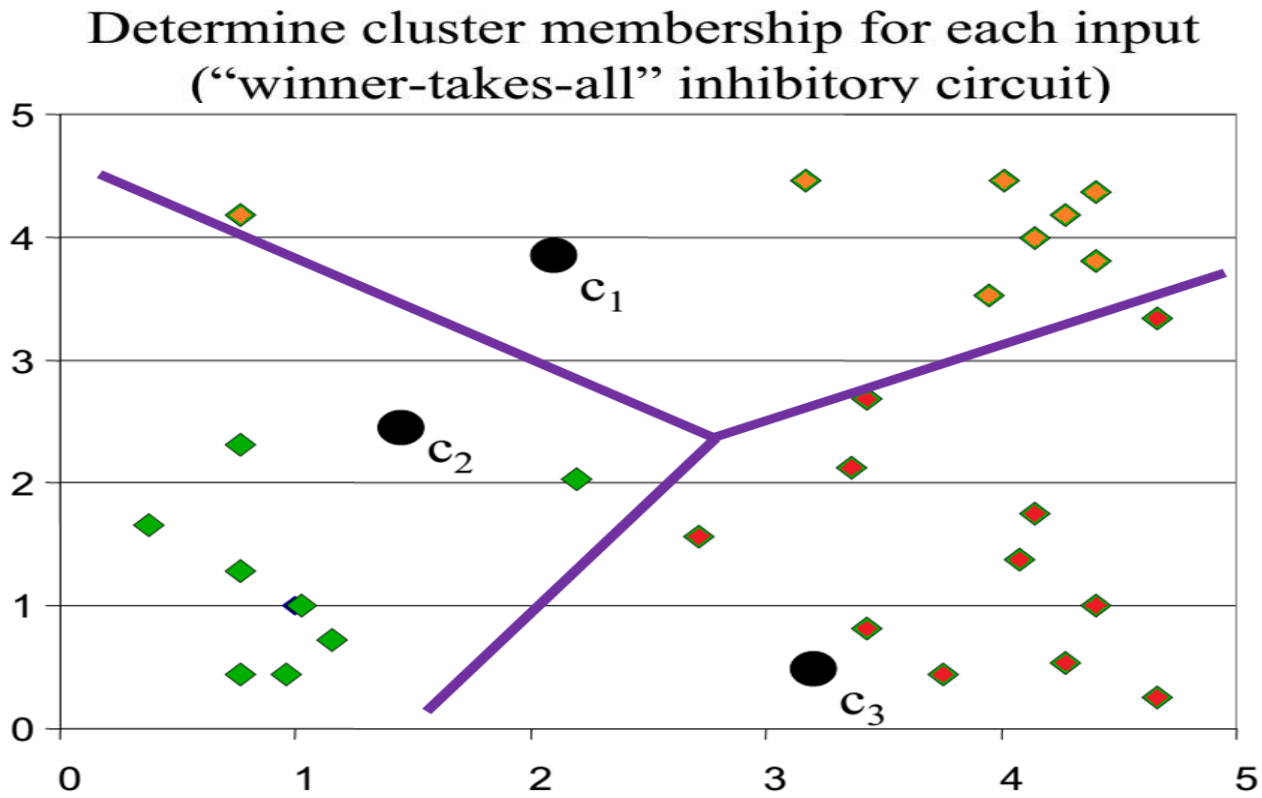
$$SSE = \sum_{j=1}^{k} \sum_{\mathbf{x} \in C_j} d(\mathbf{x}, \mathbf{m}_j)^2$$

K-means clustering example: step 1

Randomly initialize the cluster centers (synaptic weights)

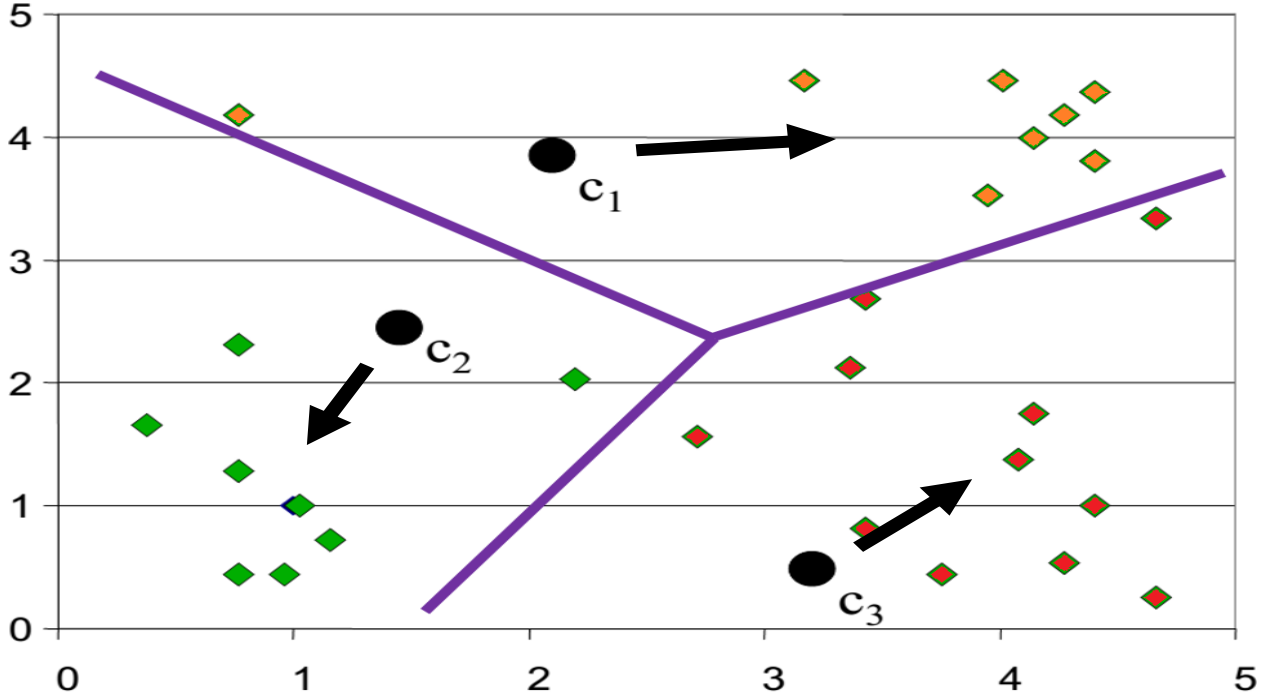


K-means clustering example – step 2

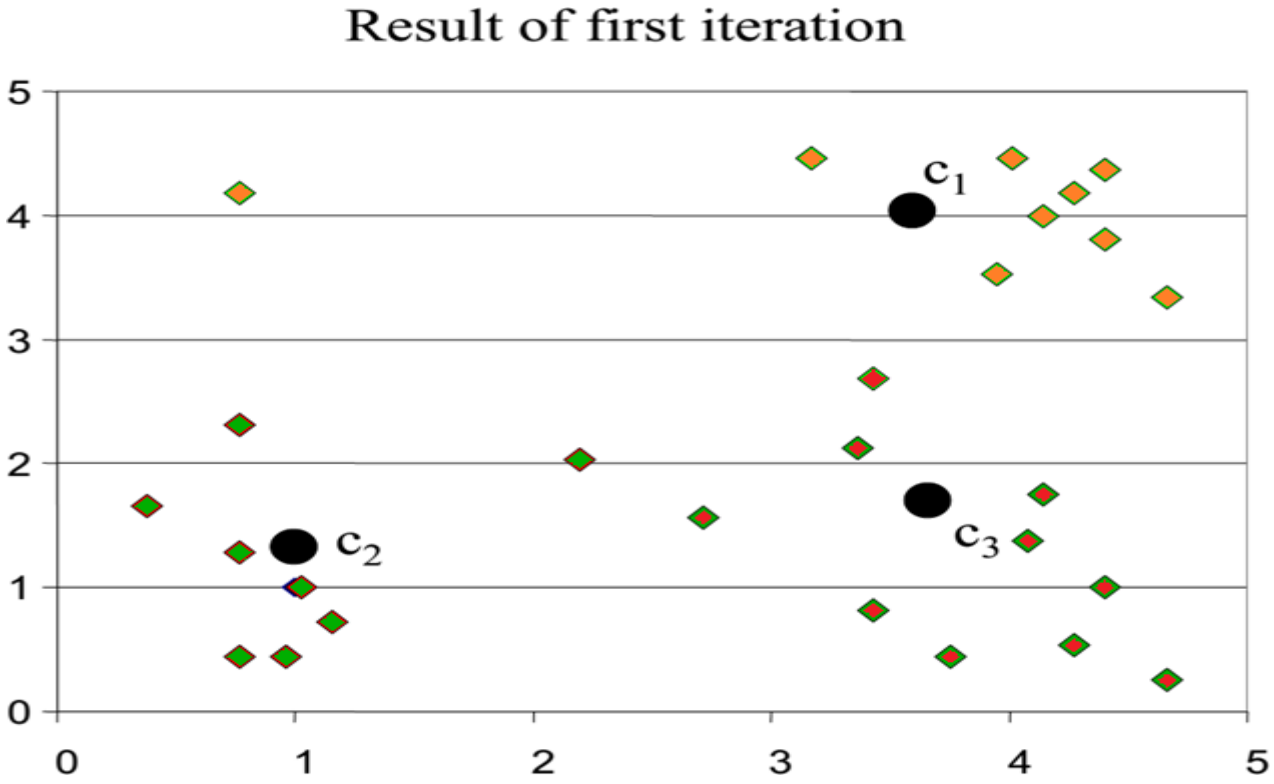


K-means clustering example – step 3

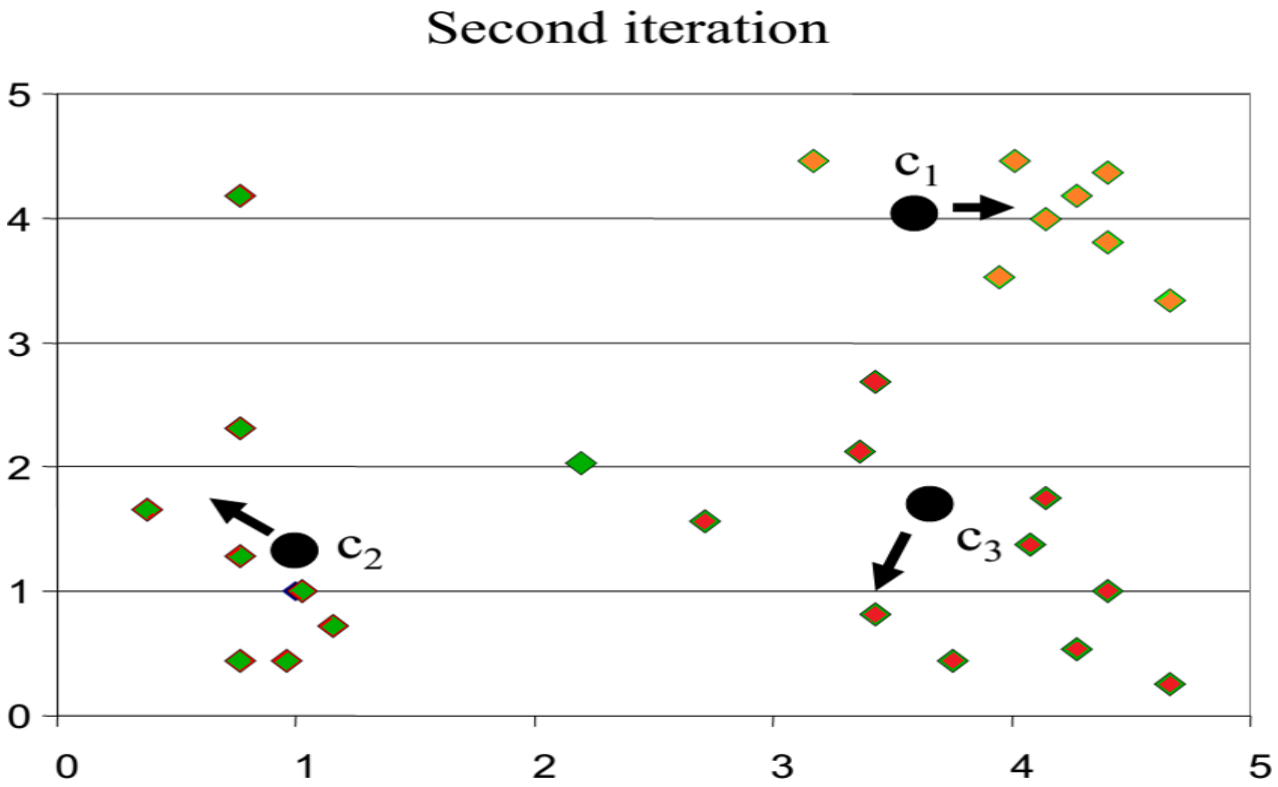
Re-estimate cluster centers (adapt synaptic weights)



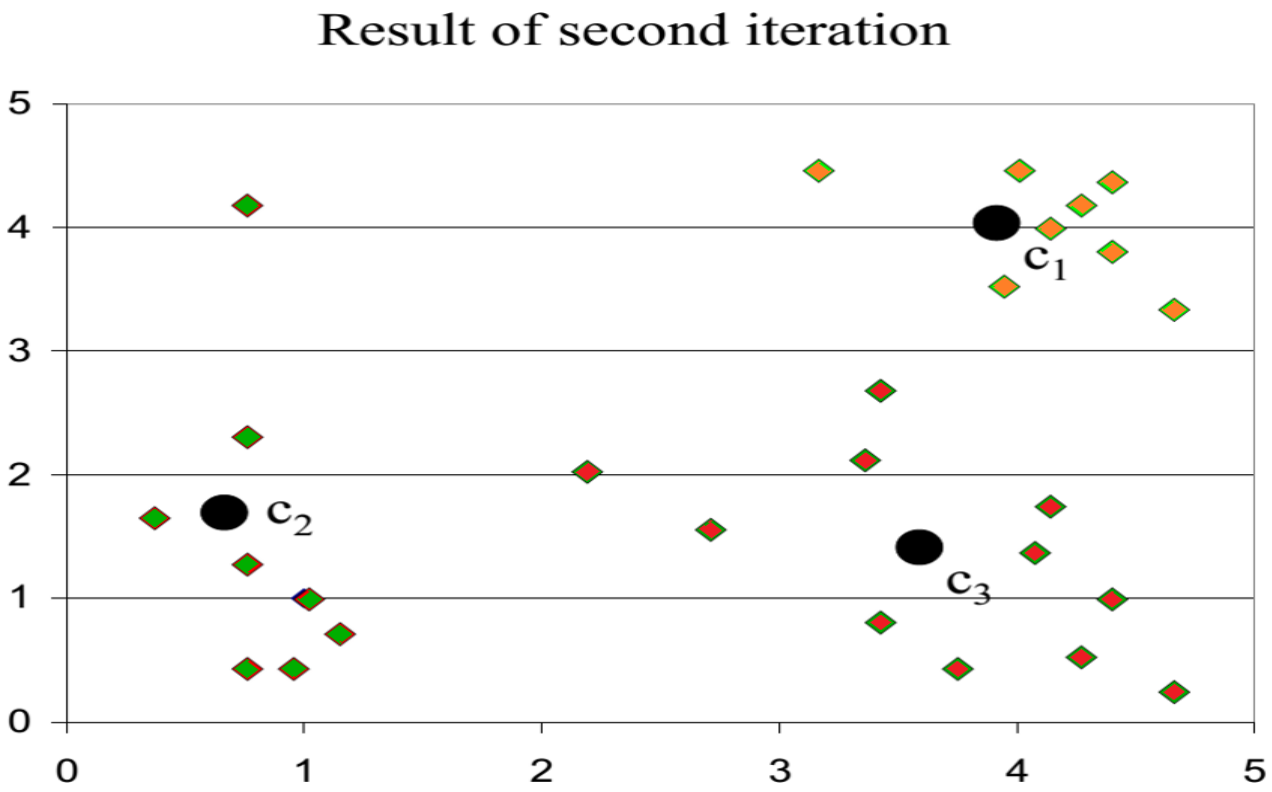
K-means clustering example



K-means clustering example



K-means clustering example



How many clusters?

3 clusters or 2 clusters?

- Possible approaches
  1. fix the number of clusters to ***k***
  2. find the best clustering according to the criterion function (number of clusters may vary)

How to choose K?

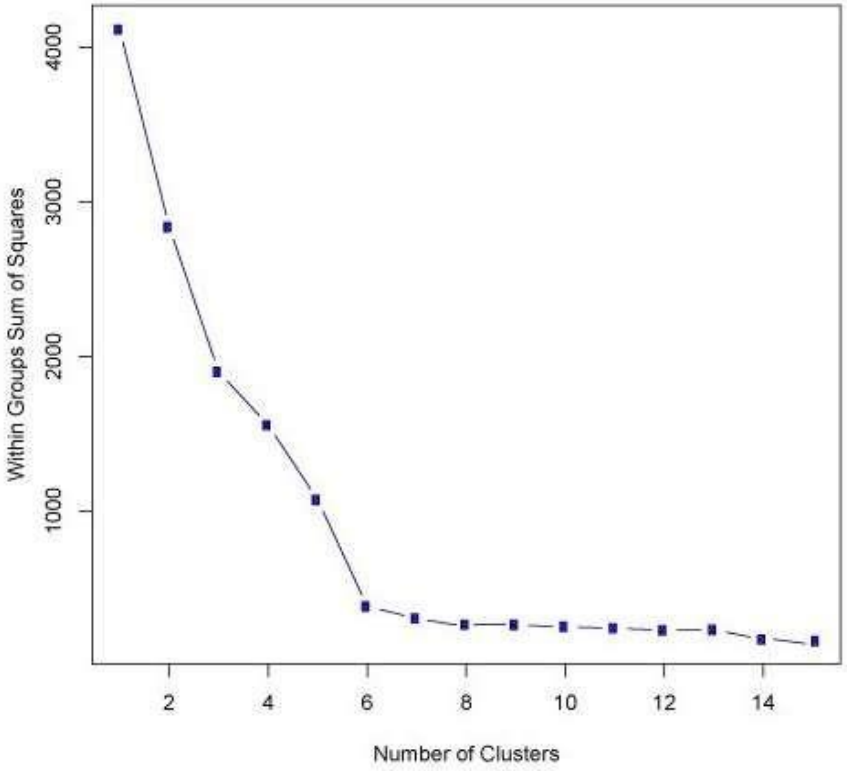
Elbow Method:

Compute the sum of squared error (SSE) for some values of k (for example 2, 4, 6, 8, etc.). The SSE is defined as the sum of the squared distance between each member of the cluster and its centroid.

Mathematically:

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

If you plot k against the SSE, you will see that *the error decreases as k gets larger*; this is because when the number of clusters increases, they should be smaller, so distortion is also smaller. The idea of the elbow method is to **choose the k at which the SSE decreases abruptly.**



Why use K-means?

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

    where n is the number of data points,

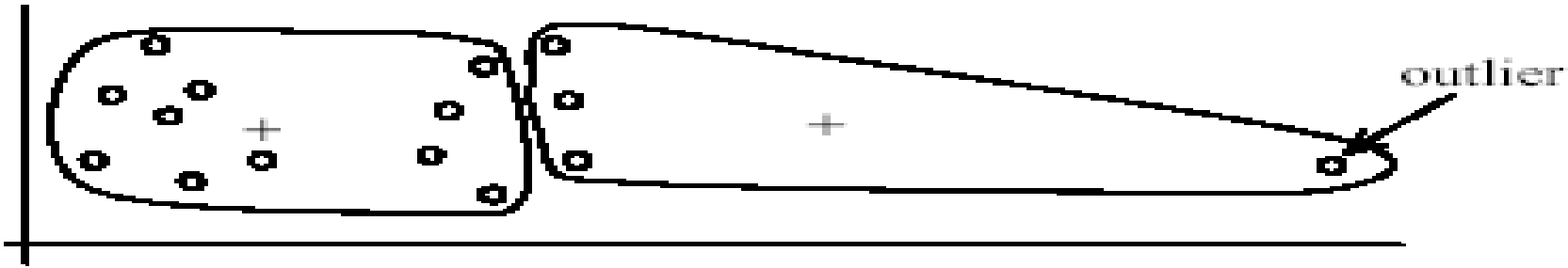
    k is the number of clusters, and

    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.

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



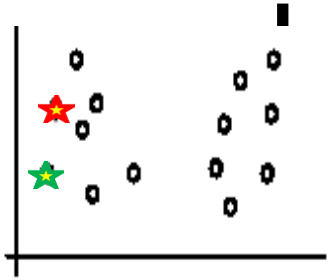
(A): Undesirable clusters

(B): Ideal clusters

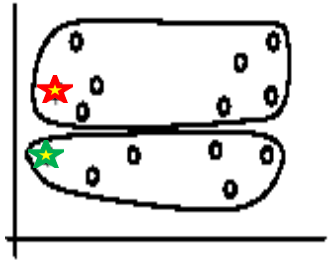
Dealing with outliers

- Remove some data points 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.
- Perform random sampling: by choosing a small subset of the data points, the chance of selecting an outlier is much smaller
  - Assign the rest of the data points to the clusters by distance or similarity comparison, or classification

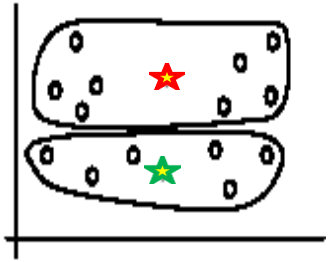
Sensitivity to initial



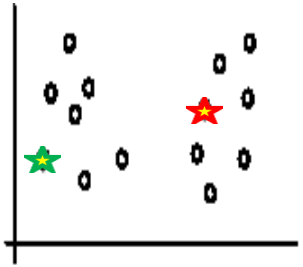
Random selection of seeds (centroids)



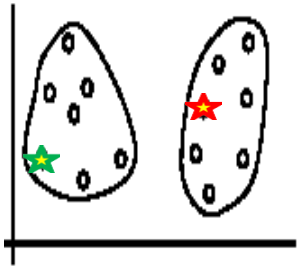
Iteration 1



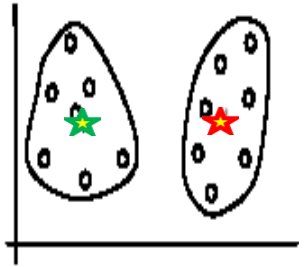
Iteration 2



Random selection of seeds (centroids)



Iteration 1

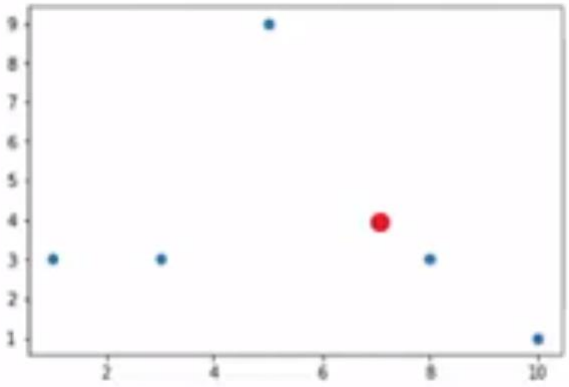


Iteration 2

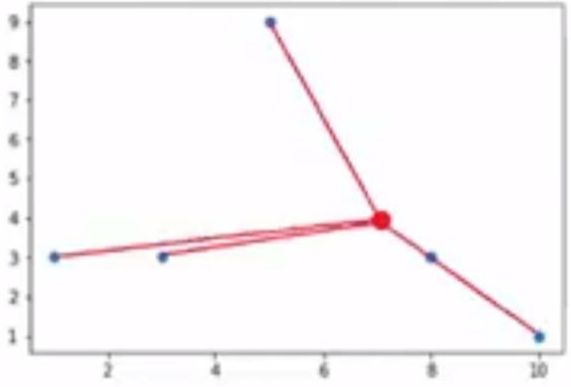
k-Means Initialization Improvements

- **kmeans++** : http://ilpubs.stanford.edu:8090/778/1/2006-13.pdf
- We begin by choosing a random point from the data.
- Then, we choose ***the next point such that is more probable to lie at a large distance from the first point***. We do so by sampling a point from a probability distribution that is proportional to the squared distance of a point from the first center.
- The remaining points are generated by a probability distribution that is proportional to the squared distance of each point from its closest center. So, a point having a large distance from its closest center is more likely to be sampled.

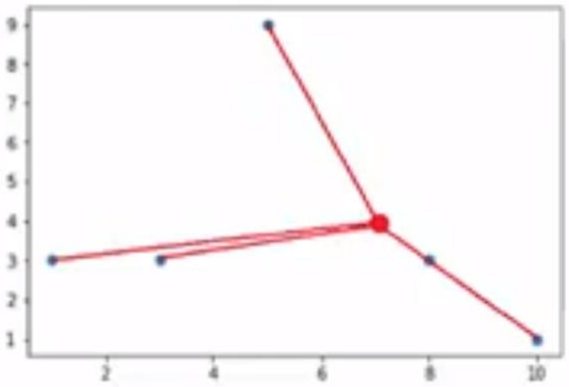
x	$\min(d(x, z_i)^2)$
(7,4)	-
(8,3)	
(5,9)	
(3,3)	
(1,3)	
(10,1)	



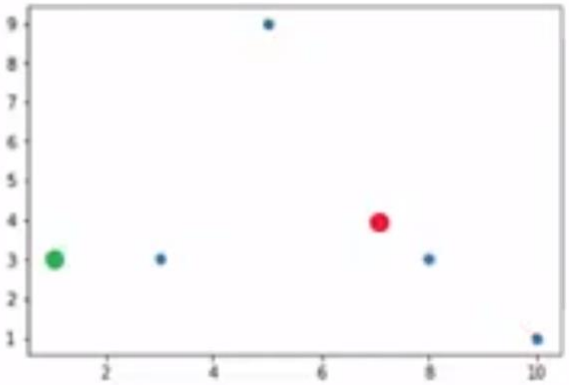
x	$\min(d(x, z_i)^2)$
(7,4)	-
(8,3)	2
(5,9)	29
(3,3)	17
(1,3)	37
(10,1)	18



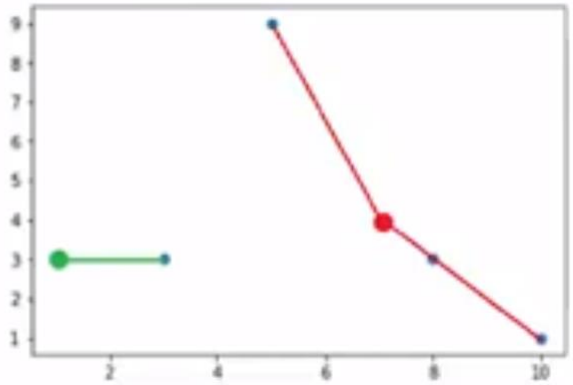
x	prob
(7,4)	-
(8,3)	2/103
(5,9)	29/103
(3,3)	17/103
(1,3)	37/103
(10,1)	18/103



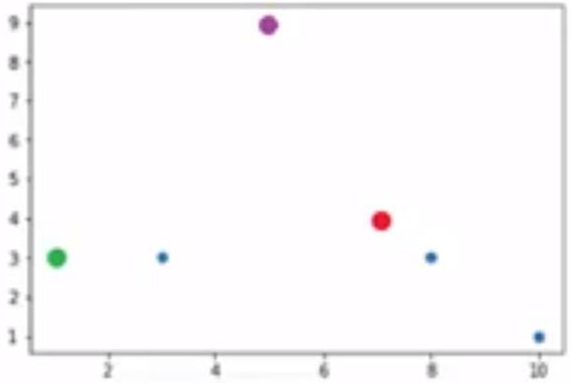
x	$\min(d(x, z_i)^2)$
(7,4)	-
(8,3)	
(5,9)	
(3,3)	
(1,3)	-
(10,1)	



x	prob
(7,4)	-
(8,3)	2/55
(5,9)	29/55
(3,3)	4/55
(1,3)	-
(10,1)	18/55

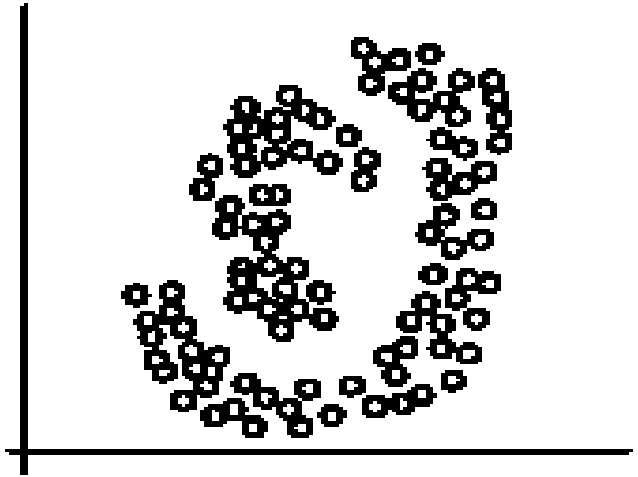


x	prob
(7,4)	-
(8,3)	
(5,9)	-
(3,3)	
(1,3)	-
(10,1)	

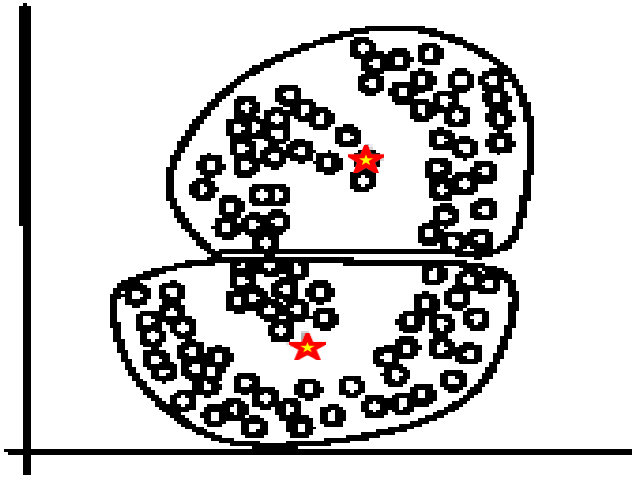


Special data structures

- The *k*-means algorithm is not suitable for discovering clusters that are not hyper-ellipsoids (or hyper-spheres).



(A): Two natural clusters

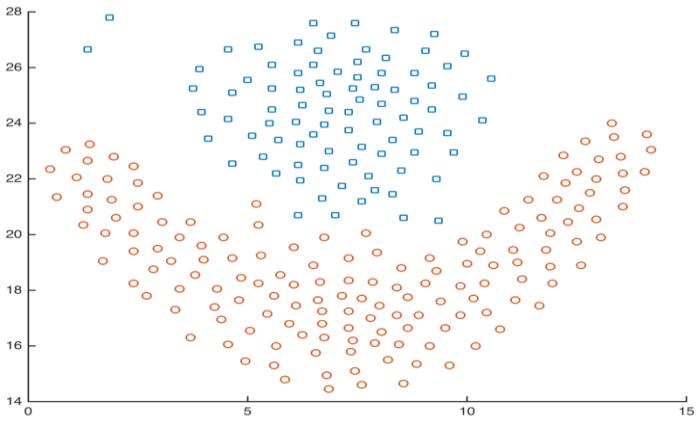


(B): *k*-means clusters

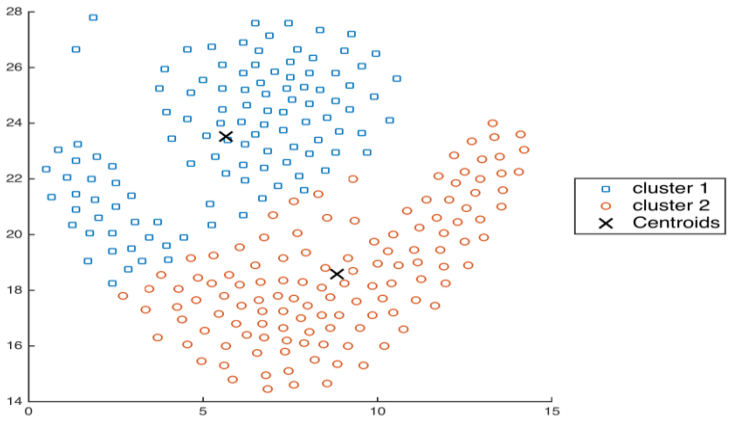
Hyperspherical Clusters

k-Means Limitations

Original Data



Results of *k*-means Clustering



Data available at: http://cs.joensuu.fi/sipu/datasets/
Original data source: Fu, L. and E. Medico. *BMC bioinformatics*, 2007. 8(1): p. 3.

K-means summary

- Despite weaknesses, *k*-means is still the most popular algorithm due to its simplicity and efficiency
- No clear evidence that any other clustering algorithm performs better in general
- Comparing different clustering algorithms is a difficult task. No one knows the correct clusters!