Machine Learning, Machine Learning (extended)

7 – Unsupervised Learning: Clustering

Kashif Rajpoot

k.m.rajpoot@cs.bham.ac.uk

School of Computer Science

University of Birmingham

Outline

- Supervised vs unsupervised learning
- Clustering
- Similarity measure
- K-means clustering
- Kernelized k-means clustering
- Hierarchical agglomerative clustering

Supervised vs unsupervised learning

- So far, we have looked at supervised learning

- Supervised learning
  - The algorithm learns from $\boldsymbol{x}_1, \boldsymbol{x}_2, \dots, \boldsymbol{x}_N$ and $t_1, t_2, \dots, t_N$ so that it can later classify $\boldsymbol{x}_{new}$
  - The availability of t_n makes learning a supervised task

- What if we only have $\boldsymbol{x}_n$, but not t_n?
  - Unsupervised learning

- Clustering: create a grouping of objects
  - Each group containing "similar" objects

Clustering: examples

- Shopping recommendation system
  - Recommend products likely to be purchased by a customer
- Grouping people from social network
  - Based on user activity
- Brain region network analysis
  - Determine “similar” brain regions that “activate” together or “rest” together

Clustering

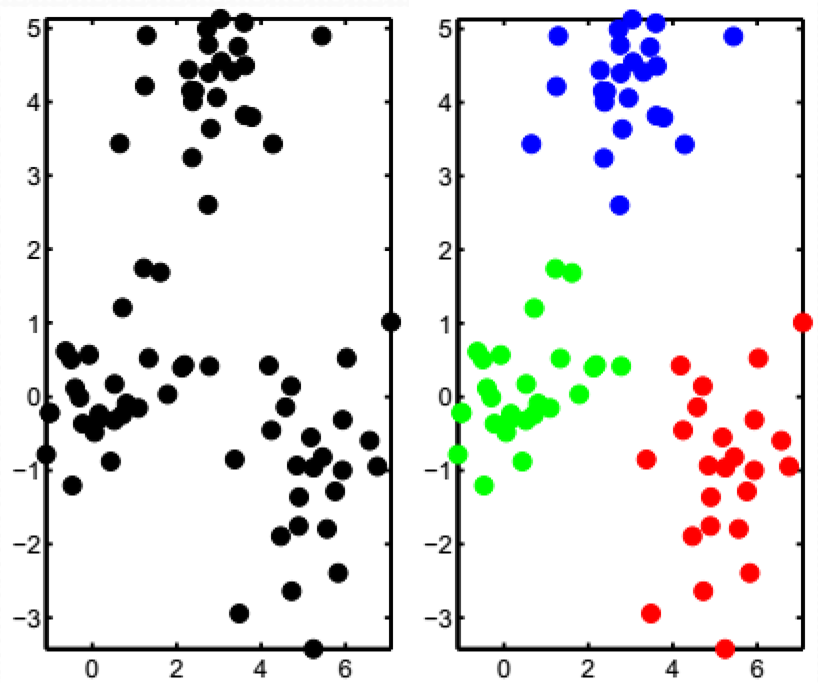
- Clustering: partition data into groups such that each group contains "similar" objects
  - Increase intra-group similarity and inter-group dissimilarity
  - Similar objects should belong to same cluster

- How do we measure "similarity" between objects?
  - "closeness"

$$D_{ij} = (\boldsymbol{x}_i - \boldsymbol{x}_j)^T (\boldsymbol{x}_i - \boldsymbol{x}_j)$$

$$= \left(x_i^{(1)} - x_j^{(1)}\right)^2 + \left(x_i^{(2)} - x_j^{(2)}\right)^2$$

$$= \sum_{d=1}^{M} \left(x_i^{(d)} - x_j^{(d)}\right)^2$$

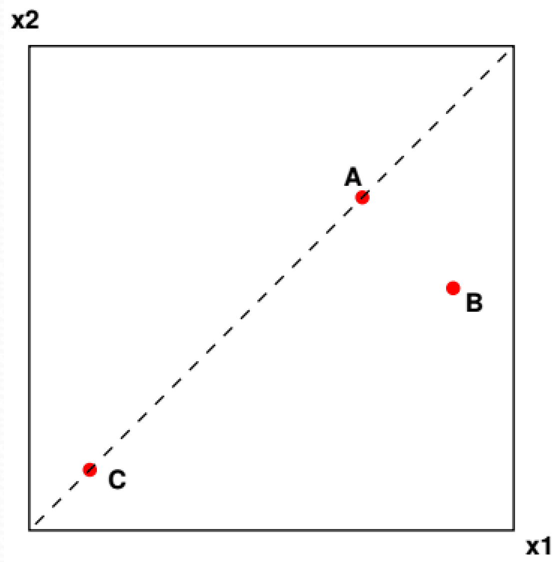


Clustering: Similarity measure

- Clustering: partition data into groups such that each group contains "similar" objects
  - Increase intra-group similarity and inter-group dissimilarity
  - Similar objects should belong to same cluster

- How do we measure "similarity" between objects?
  - "closeness"

- Which points are "closest" or "most similar"?

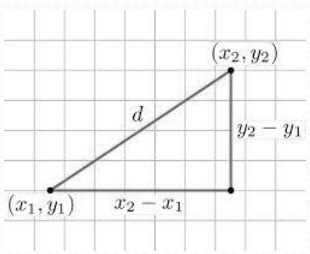


- Choice of similarity measure is dependent on the nature of data and your application

Clustering: Similarity measure

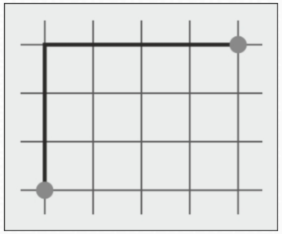
- How to estimate the distance (i.e. similarity) between two objects/points $\boldsymbol{p}$ and $\boldsymbol{q}$?
  - Various measures can be used
- K-means algorithm is flexible to use any distance measure

Euclidean distance



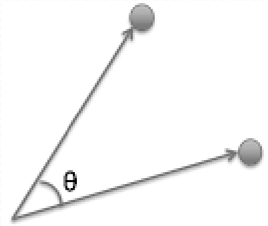
$$D_e(\boldsymbol{p}, \boldsymbol{q}) = \sqrt{\sum_{d=1}^{M} (p^{(d)} - q^{(d)})^2}$$

Manhattan distance



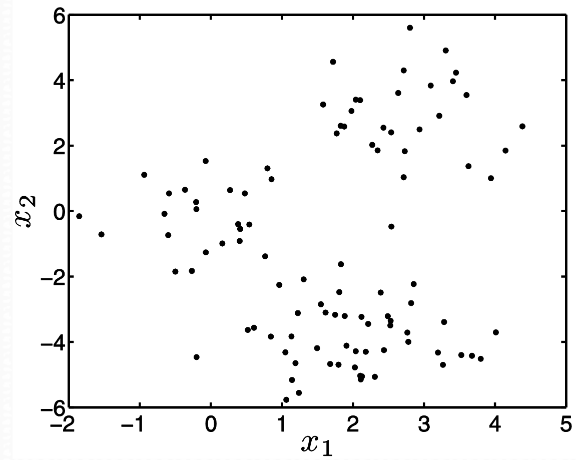
$$D_m(\boldsymbol{p}, \boldsymbol{q}) = \sum_{d=1}^{M} \left| p^{(d)} - q^{(d)} \right|$$

Cosine angle



$$D_c(\boldsymbol{p}, \boldsymbol{q}) = cos^{-1}\left(\frac{\sum_{d=1}^{M} p^{(d)} q^{(d)}}{\sqrt{\sum_{d=1}^{M} {p^{(d)}}^2} \sqrt{\sum_{d=1}^{M} {q^{(d)}}^2}} \right) / \pi$$

K-means clustering



- Consider that there are total K clusters
- In 2D, each cluster k is represented by a cluster centre (i.e. mean)

$$\boldsymbol{\mu}_k = \left[\mu_k^{(1)}, \mu_k^{(2)}\right]^T$$

- Each object $\boldsymbol{x}_n$ is assigned to its closest cluster k on the basis of its distance to cluster mean $\boldsymbol{\mu}_k$
- Distance to cluster k can be estimated in various ways
  - For example: squared Euclidean distance

$$D_{nk} = (\boldsymbol{x}_n - \boldsymbol{\mu}_k)^T(\boldsymbol{x}_n - \boldsymbol{\mu}_k)$$

- No analytical solution for finding the closest cluster mean $\boldsymbol{\mu}_k$ for all objects
  - Iterative solution

K-means clustering

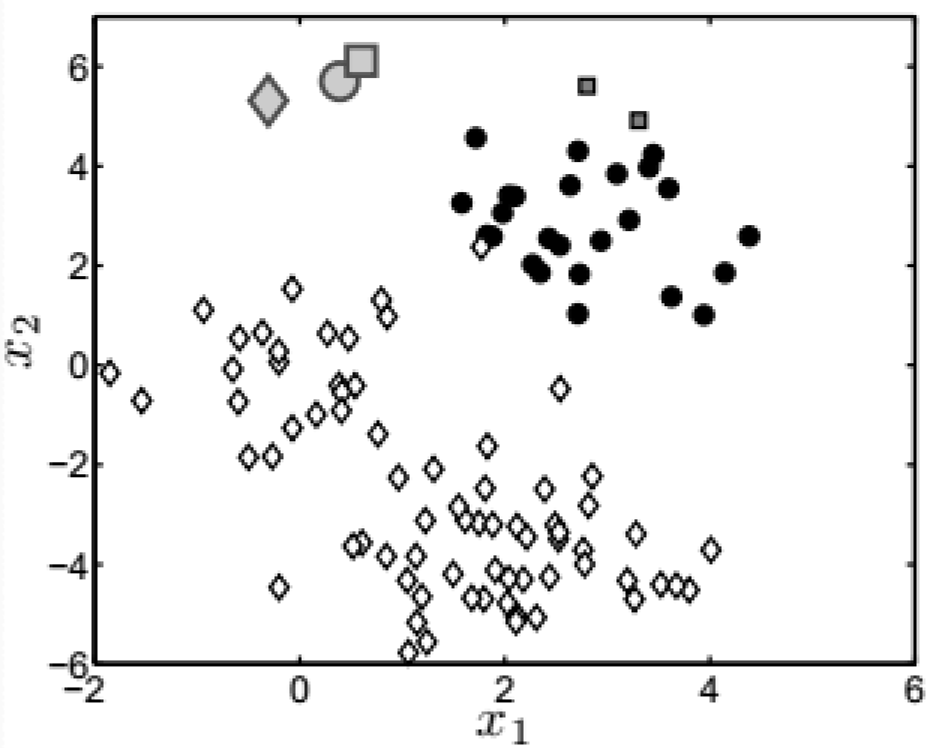
- Iterative algorithm: <u>Method 1</u>
  1. Randomly initialize cluster means $\boldsymbol{\mu}_1, \boldsymbol{\mu}_2, \dots, \boldsymbol{\mu}_K$
  2. Compute $D_{n1}, D_{n2}, \dots, D_{nK}$ for each object $\boldsymbol{x}_n$
  3. Assign the object $\boldsymbol{x}_n$ to cluster k with lowest distance D_{nk}
     - i.e. assign $\boldsymbol{x}_n$ to its closest cluster k with mean $\boldsymbol{\mu}_k$
  4. Update each cluster mean $\boldsymbol{\mu}_k$, to represent the mean of newly updated cluster
     $$\boldsymbol{\mu}_k = \frac{1}{N_k} \sum_{n=1}^{N_k} \boldsymbol{x}_n$$
  5. Stop if assignments don't change, or reached max number of iterations, else jump to step 2

- Algorithm should converge after a number of iterations

K-means clustering

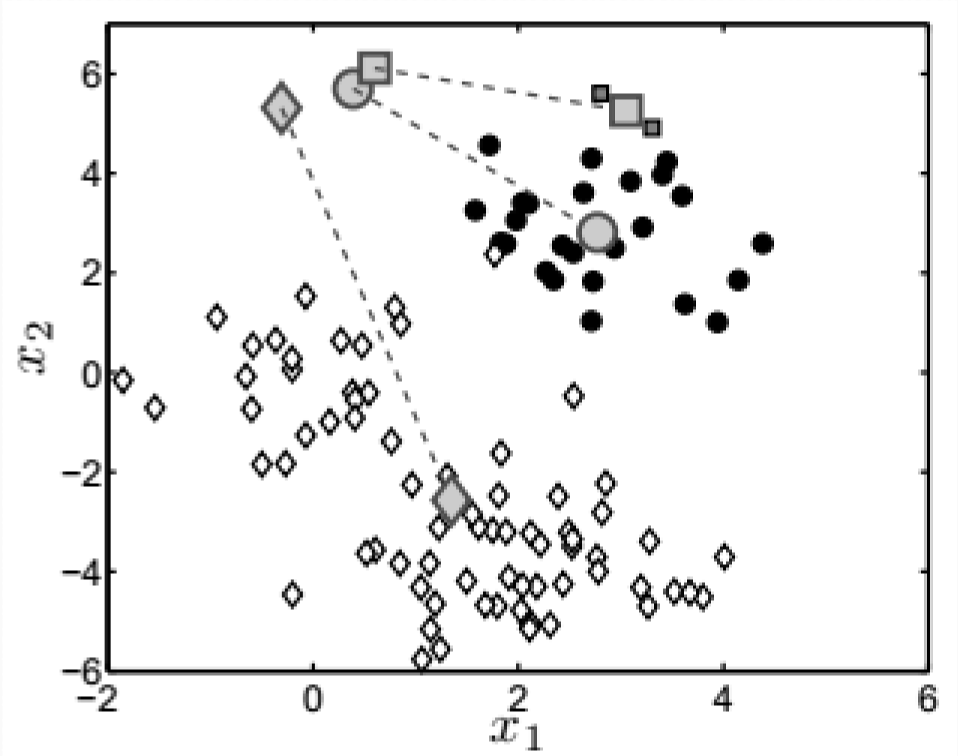
- Randomly initialize cluster means $\boldsymbol{\mu}_1, \boldsymbol{\mu}_2, \dots, \boldsymbol{\mu}_k$

- Assign each $\boldsymbol{x}_n$ to its closest cluster with mean $\boldsymbol{\mu}_k$



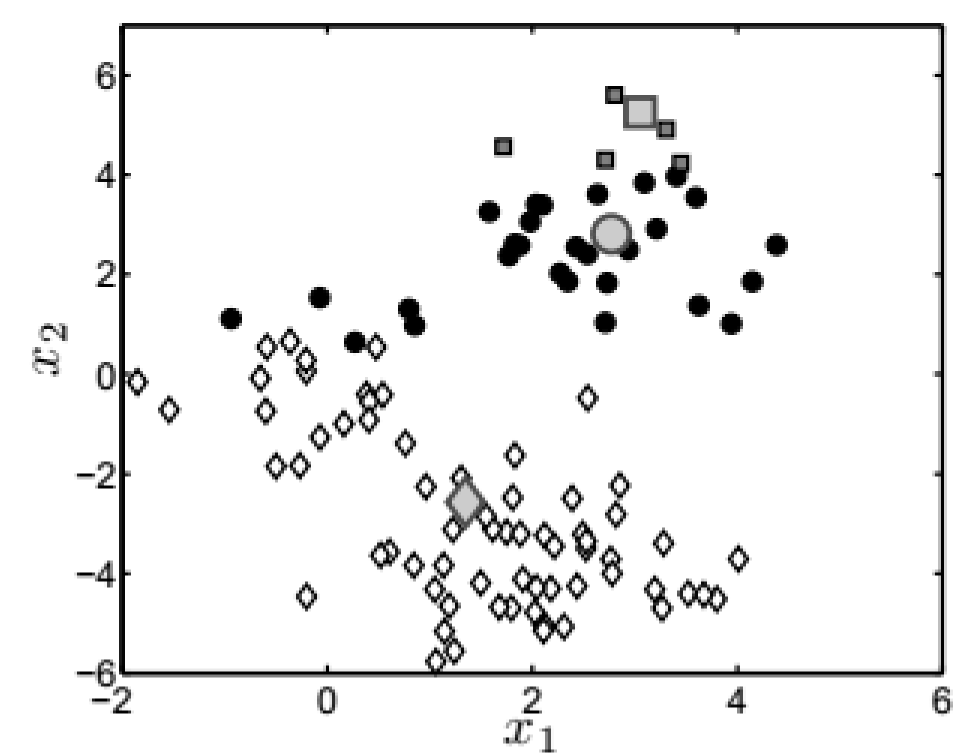
K-means clustering

- Update means $\boldsymbol{\mu}_k = \frac{1}{N_k}\sum_{n=1}^{N_k} \boldsymbol{x}_n$



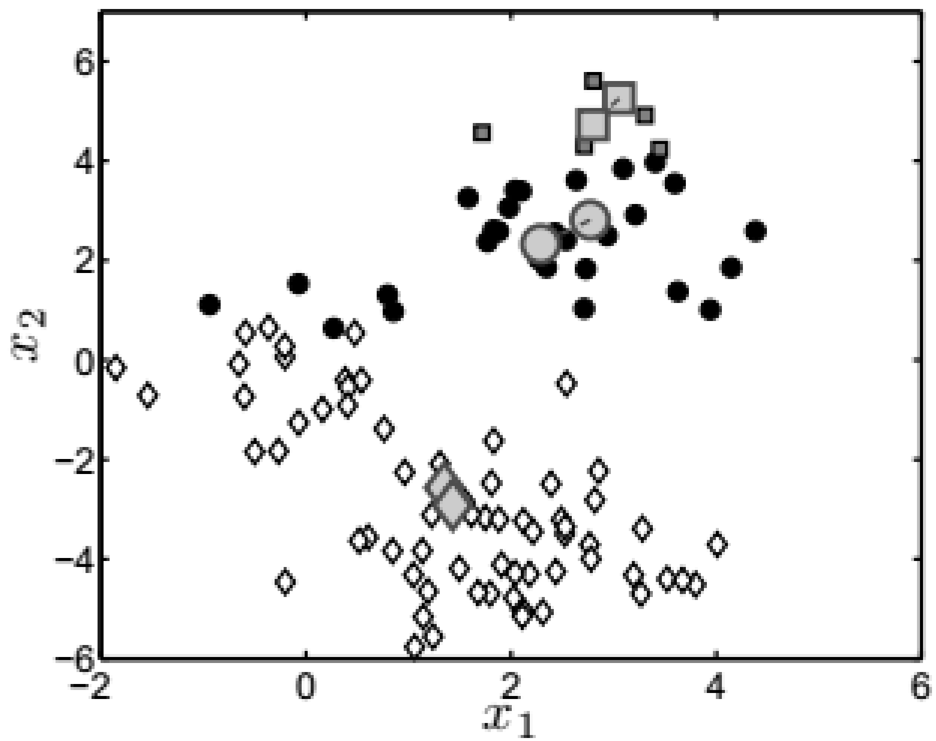
K-means clustering

- Assign each $\boldsymbol{x}_n$ to its closest cluster with mean $\boldsymbol{\mu}_k$



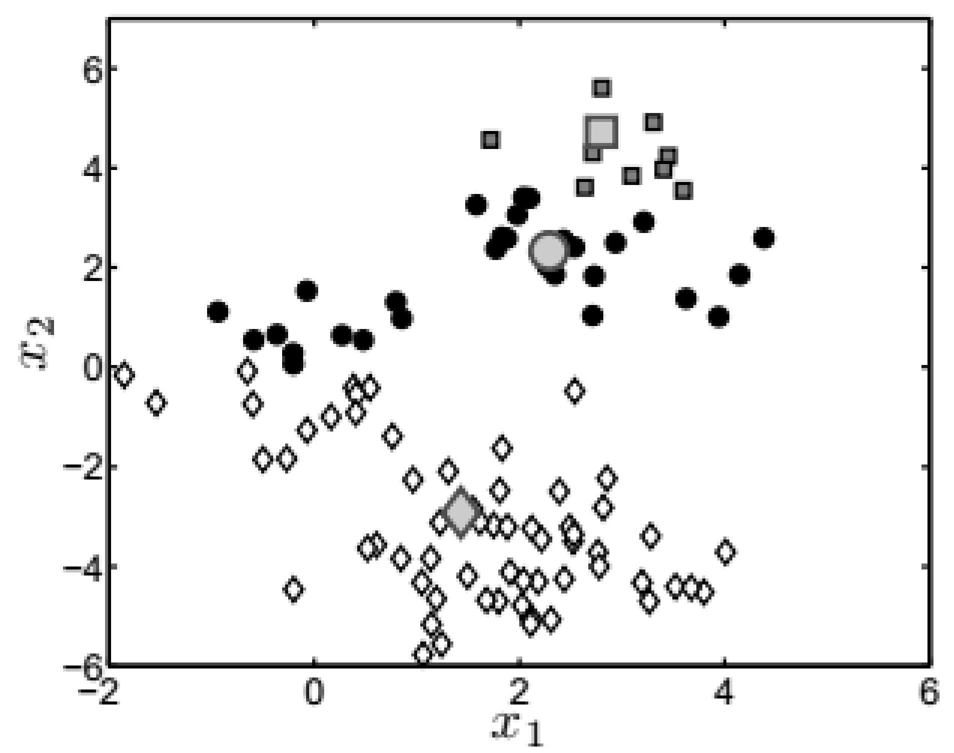
K-means clustering

- Update means $\boldsymbol{\mu}_k = \frac{1}{N_k}\sum_{n=1}^{N_k} \boldsymbol{x}_n$



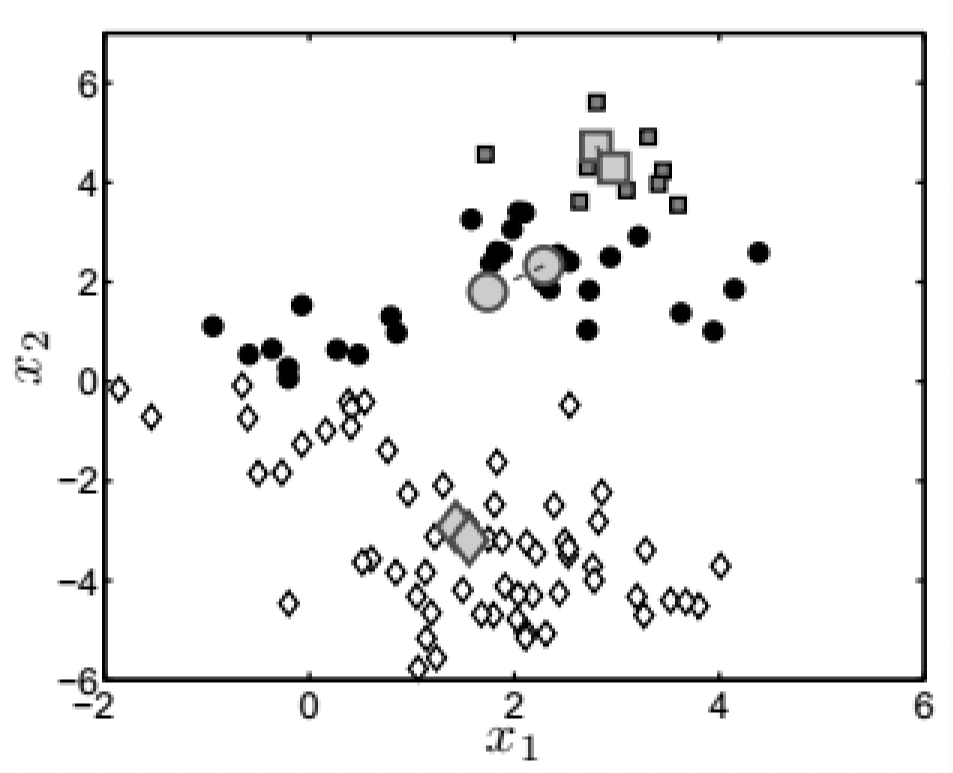
K-means clustering

- Assign each $\boldsymbol{x}_n$ to its closest cluster with mean $\boldsymbol{\mu}_k$



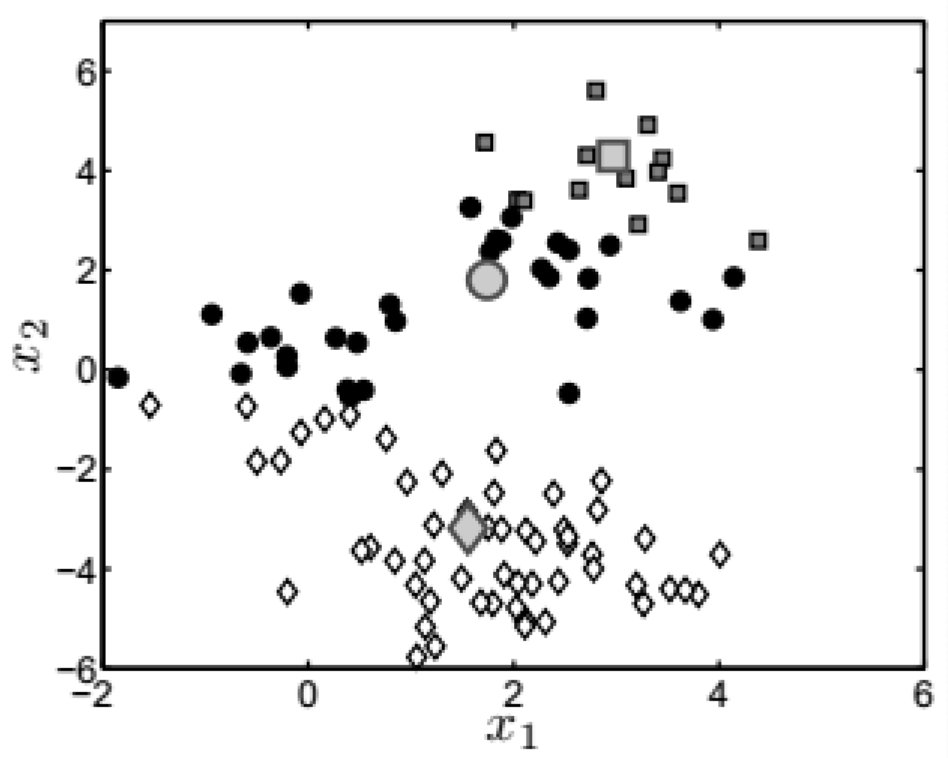
K-means clustering

- Update means $\boldsymbol{\mu}_k = \frac{1}{N_k}\sum_{n=1}^{N_k} \boldsymbol{x}_n$



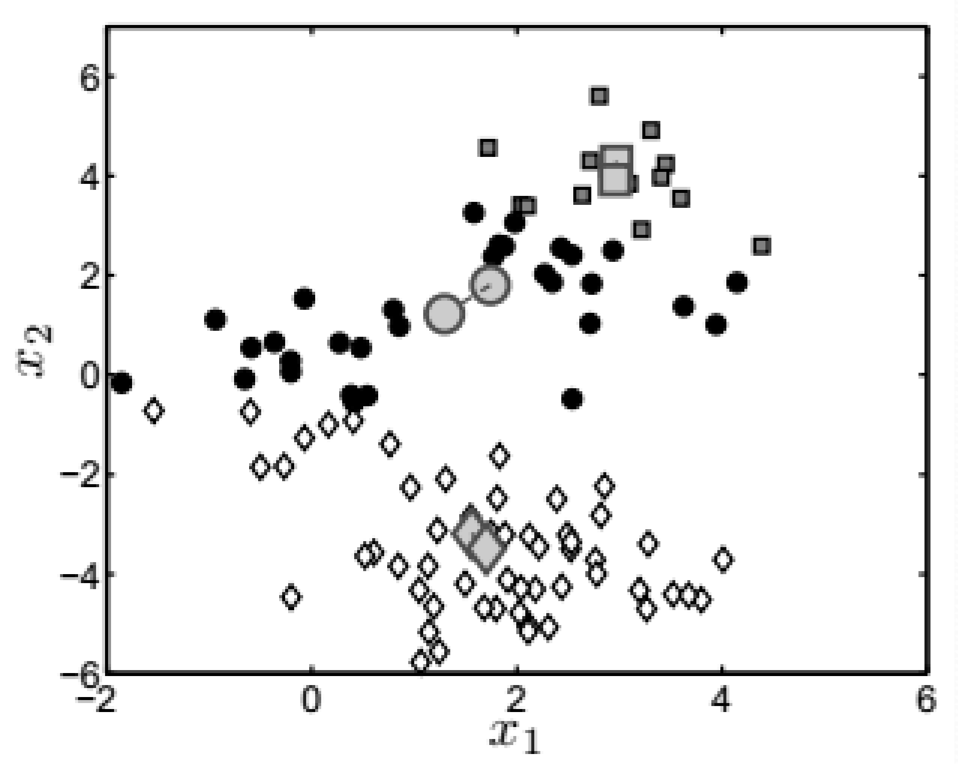
K-means clustering

- Assign each $\boldsymbol{x}_n$ to its closest cluster with mean $\boldsymbol{\mu}_k$



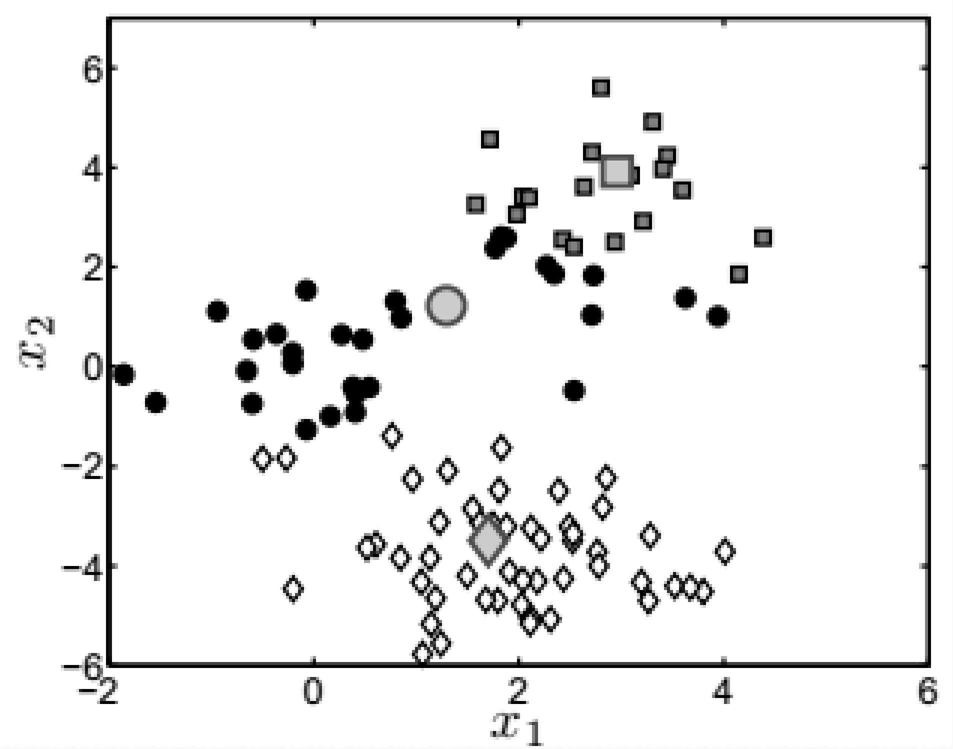
K-means clustering

- Update means $\boldsymbol{\mu}_k = \frac{1}{N_k}\sum_{n=1}^{N_k}\boldsymbol{x}_n$



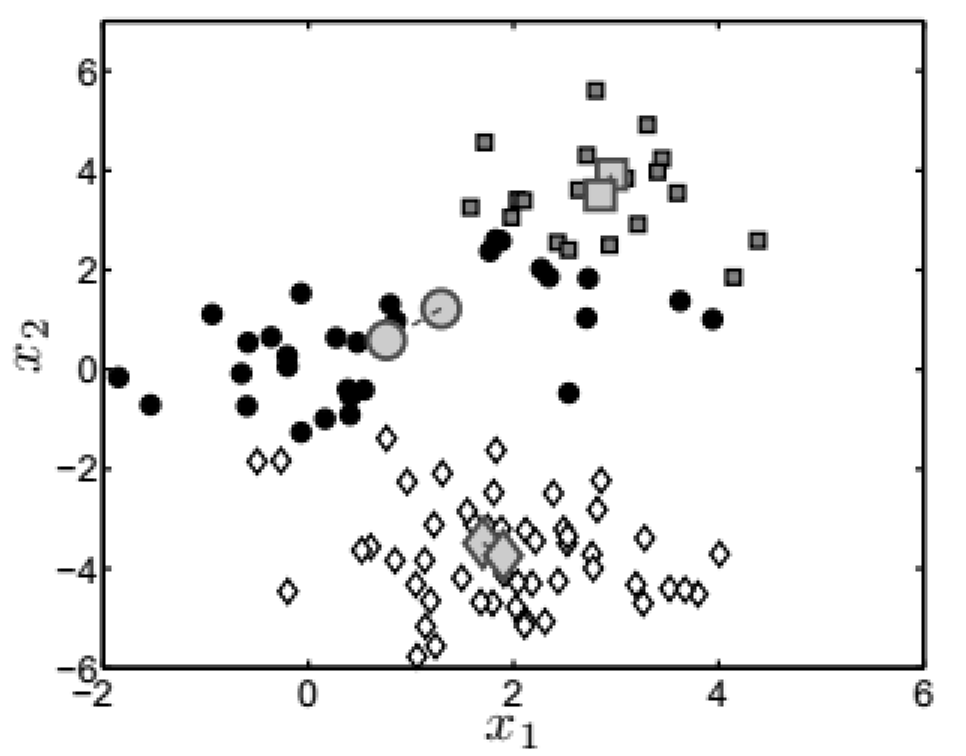
K-means clustering

- Assign each $\boldsymbol{x}_n$ to its closest cluster with mean $\boldsymbol{\mu}_k$



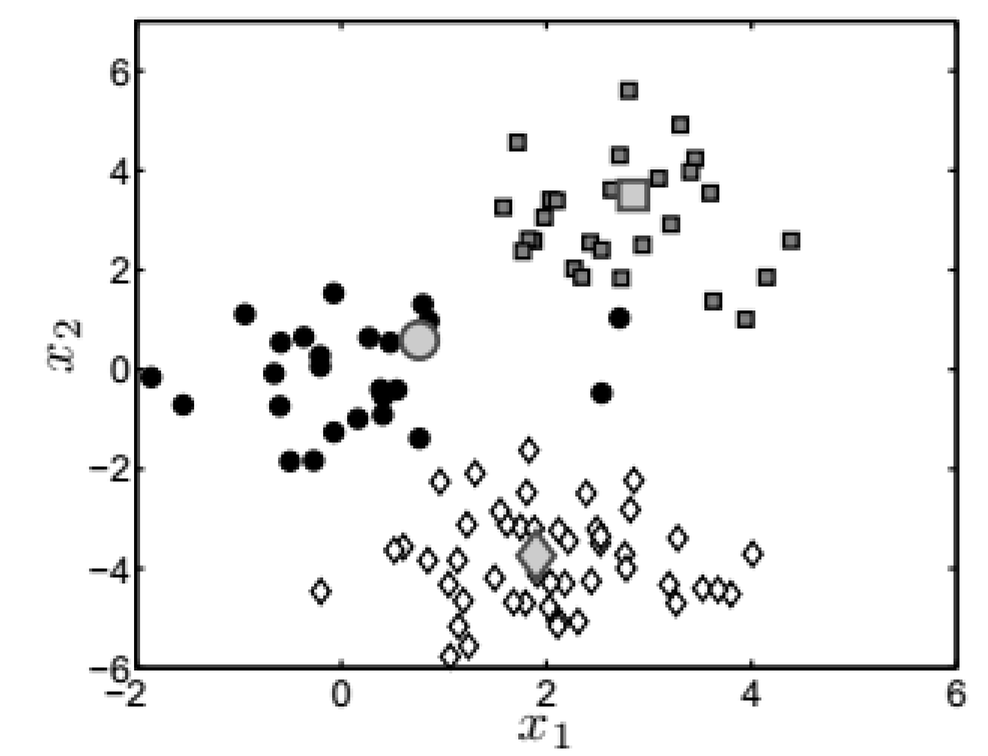
K-means clustering

- Update means $\boldsymbol{\mu}_k = \frac{1}{N_k}\sum_{n=1}^{N_k} \boldsymbol{x}_n$



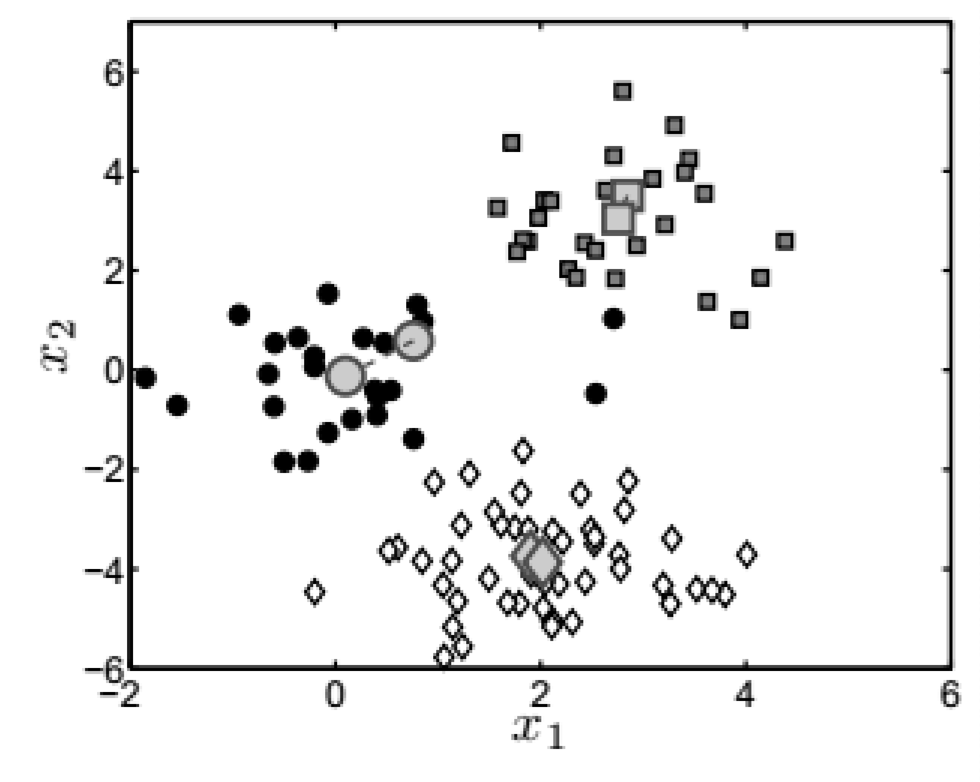
K-means clustering

- Assign each $\boldsymbol{x}_n$ to its closest cluster with mean $\boldsymbol{\mu}_k$



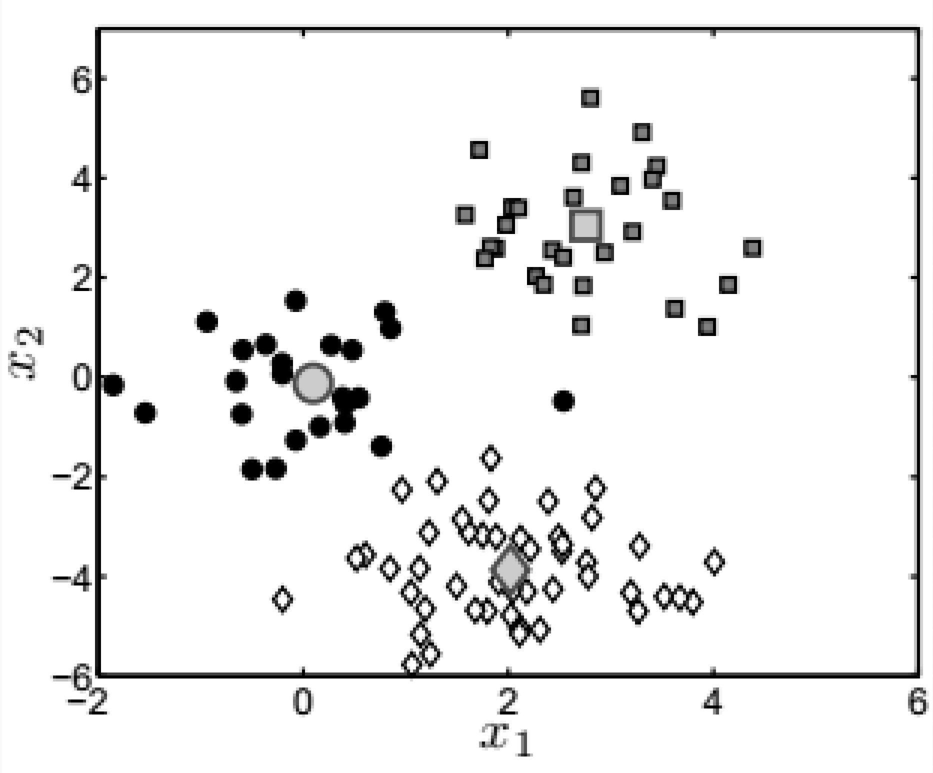
K-means clustering

- Update means $\boldsymbol{\mu}_k = \frac{1}{N_k}\sum_{n=1}^{N_k} \boldsymbol{x}_n$



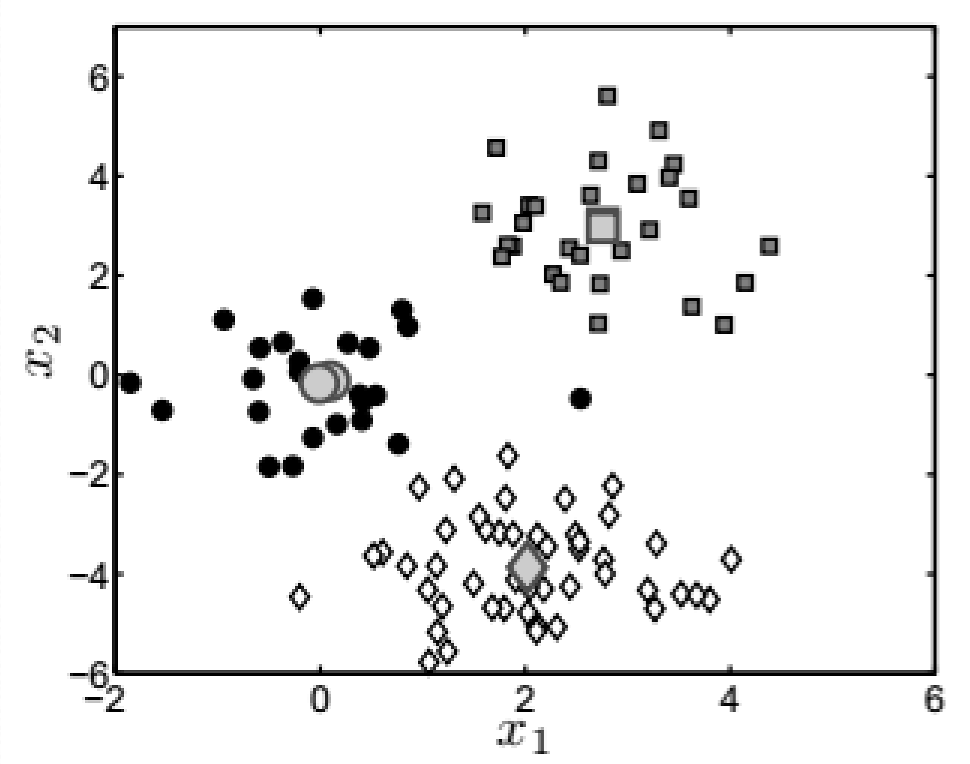
K-means clustering

- Assign each $\boldsymbol{x}_n$ to its closest cluster with mean $\boldsymbol{\mu}_k$



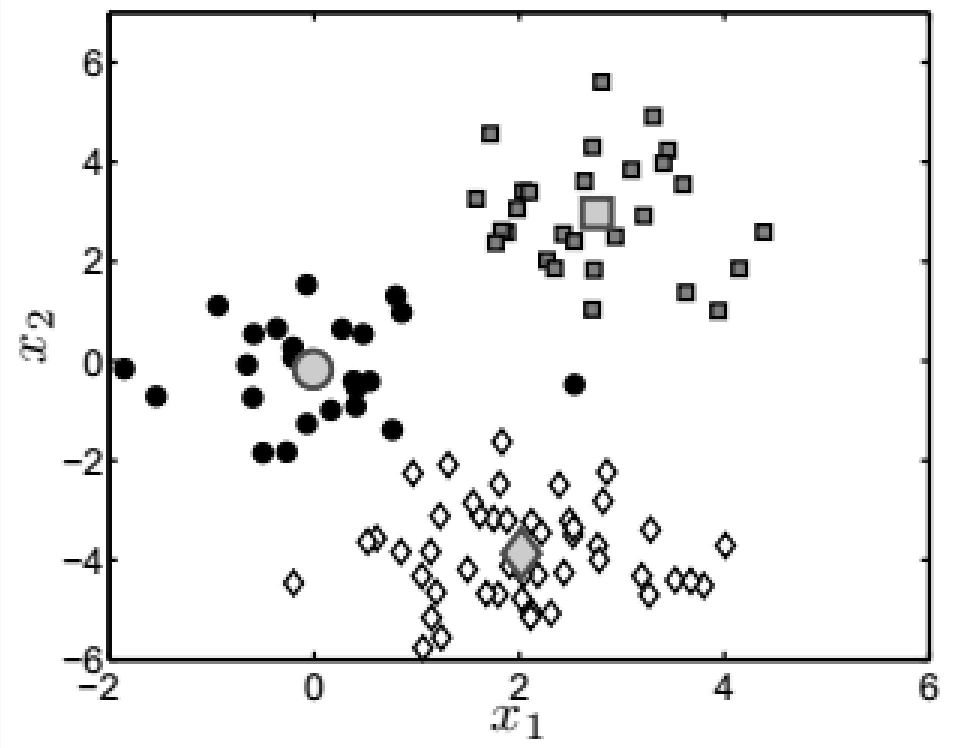
K-means clustering

- Update means $\boldsymbol{\mu}_k = \frac{1}{N_k}\sum_{n=1}^{N_k} \boldsymbol{x}_n$



K-means clustering

- Solution at convergence



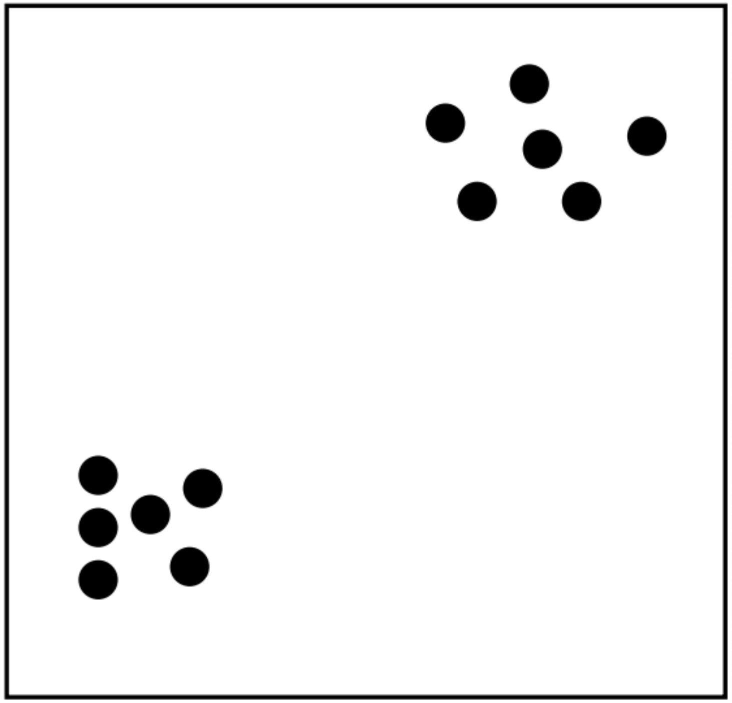
K-means clustering

- Iterative algorithm: Method 2
  1. Randomly assign each object $\boldsymbol{x}_n$ to one of K clusters
  2. Compute cluster means $\boldsymbol{\mu}_1, \boldsymbol{\mu}_2, \dots, \boldsymbol{\mu}_K$ for each cluster k, to represent the mean of newly updated cluster
  $$\boldsymbol{\mu}_k = \frac{1}{N_k} \sum_{n=1}^{N_k} \boldsymbol{x}_n$$
  3. Compute $D_{n1}, D_{n2}, \dots, D_{nK}$ for each object $\boldsymbol{x}_n$
  4. Assign the object $\boldsymbol{x}_n$ to cluster k with lowest distance D_{nk}
     - i.e. assign $\boldsymbol{x}_n$ to its closest cluster k with mean $\boldsymbol{\mu}_k$
  5. Stop iterations if assignments don't change, or maximum number of iterations reached, else jump to step 2

- Algorithm will converge after a number of iterations

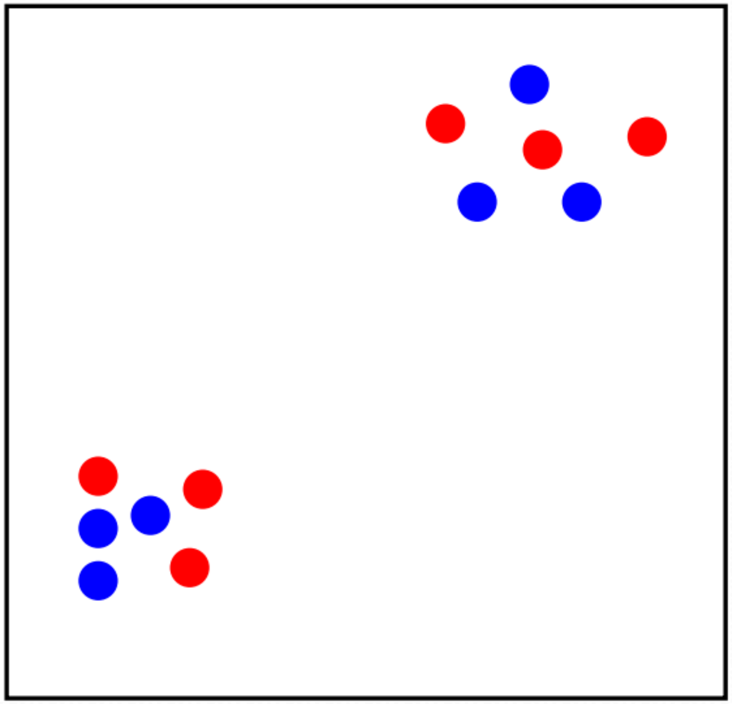
K-means clustering

- Cluster data points with Method 2



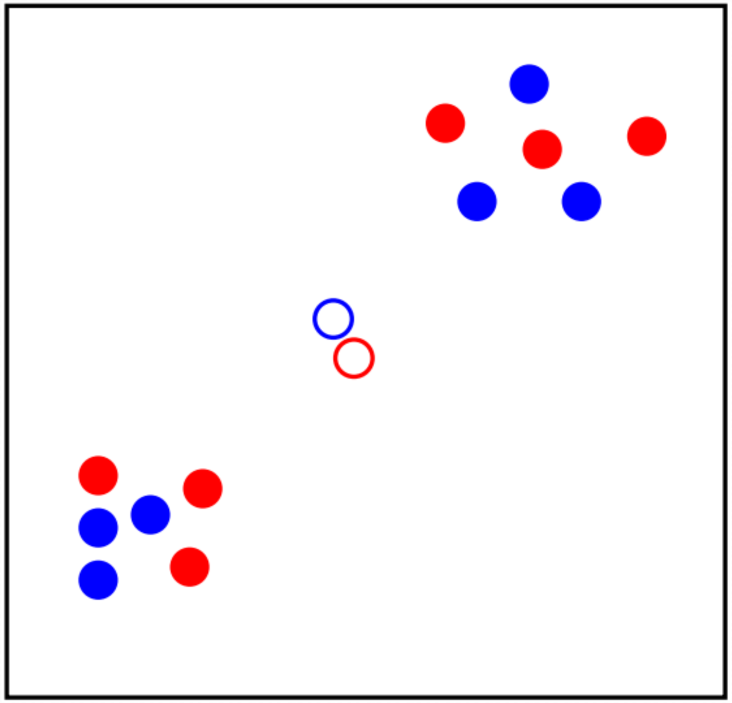
K-means clustering

- Randomly assign each object $\boldsymbol{x}_n$ to one of K clusters



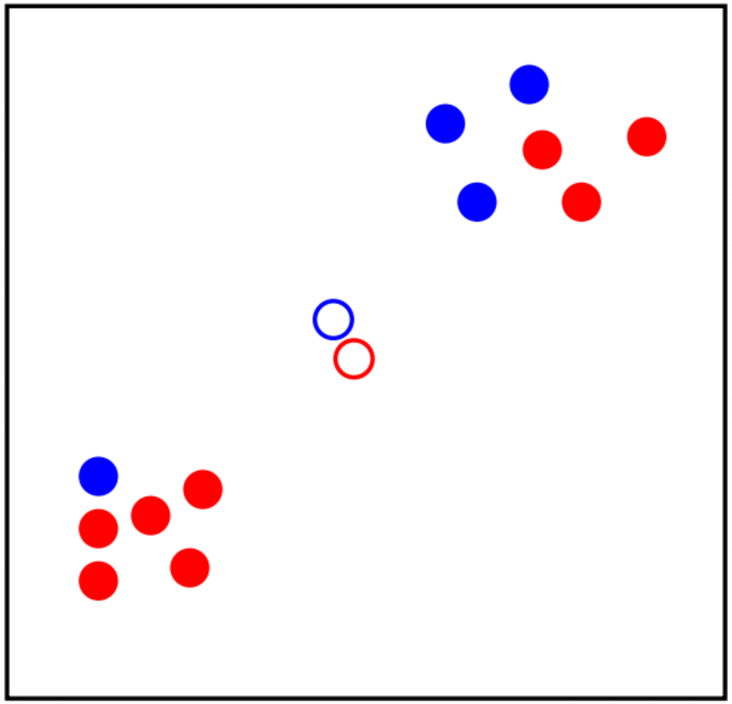
K-means clustering

- Update means $\boldsymbol{\mu}_k = \frac{1}{N_k}\sum_{n=1}^{N_k} \boldsymbol{x}_n$



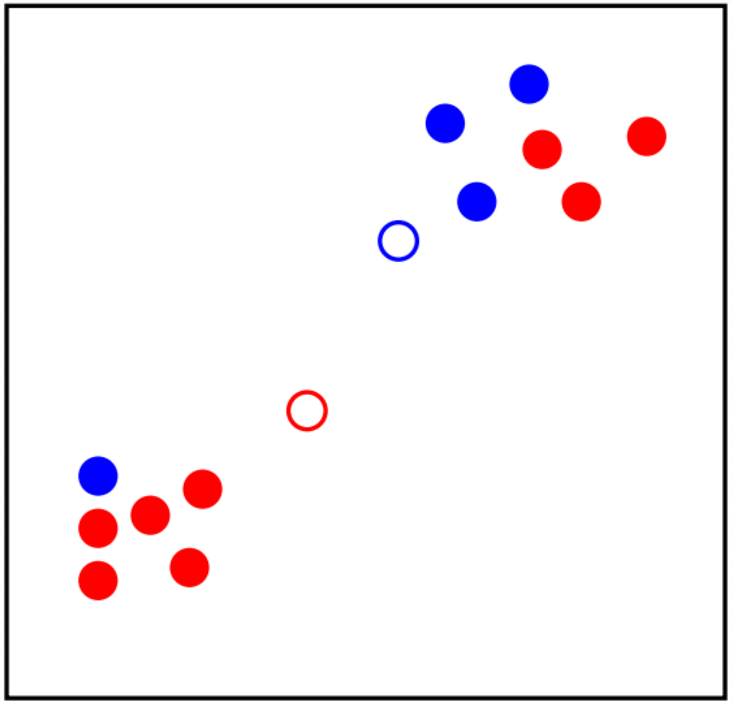
K-means clustering

- Assign each $\boldsymbol{x}_n$ to its closest cluster with mean $\boldsymbol{\mu}_k$



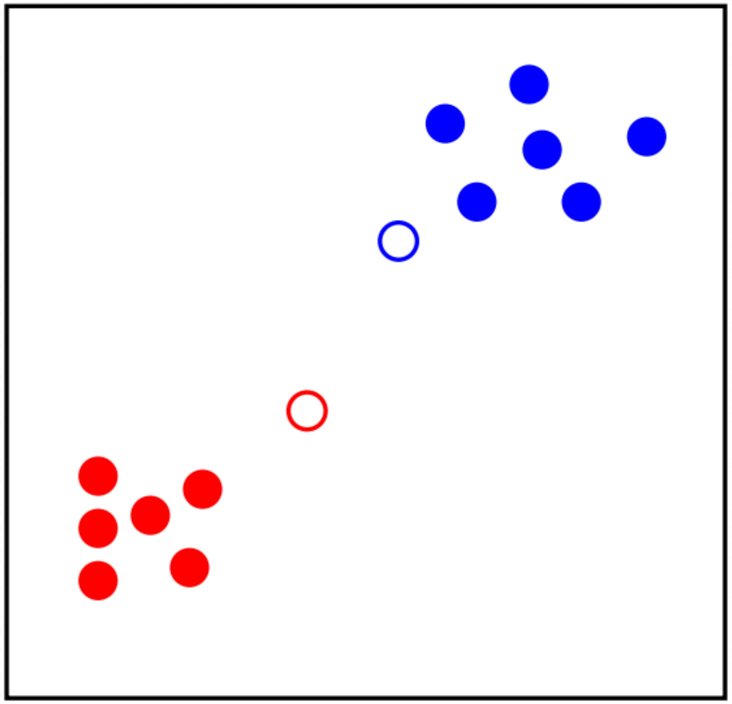
K-means clustering

- Update means $\boldsymbol{\mu}_k = \frac{1}{N_k} \sum_{n=1}^{N_k} \boldsymbol{x}_n$



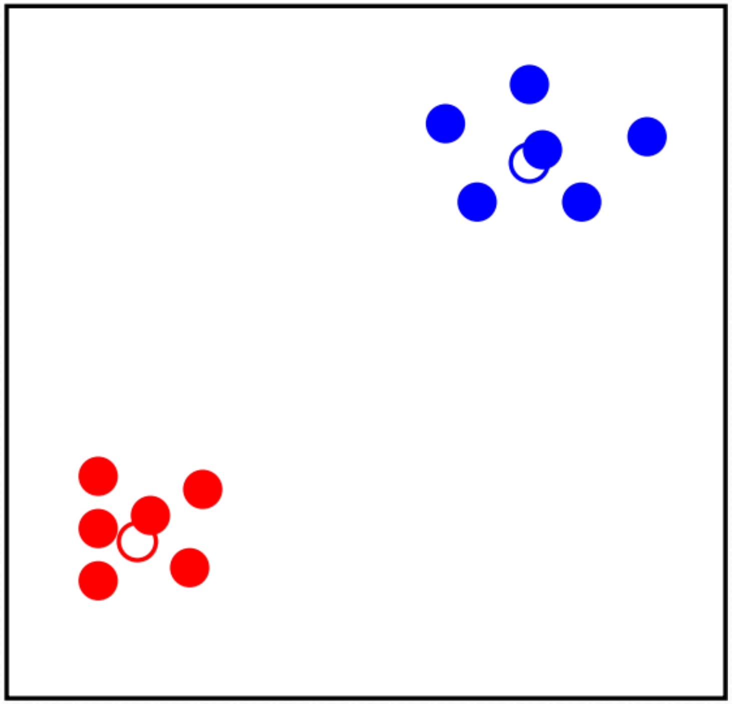
K-means clustering

- Assign each $\boldsymbol{x}_n$ to its closest cluster with mean $\boldsymbol{\mu}_k$



K-means clustering

- Update means $\boldsymbol{\mu}_k = \frac{1}{N_k}\sum_{n=1}^{N_k} \boldsymbol{x}_n$



K-means clustering

- The k-means clustering algorithm converges to a local minimum of the following *intra-group distance estimate*:

$$\mathcal{D} = \sum_{k=1}^{K} \sum_{n=1}^{N_k} (\boldsymbol{x}_n - \boldsymbol{\mu}_k)^T (\boldsymbol{x}_n - \boldsymbol{\mu}_k)$$

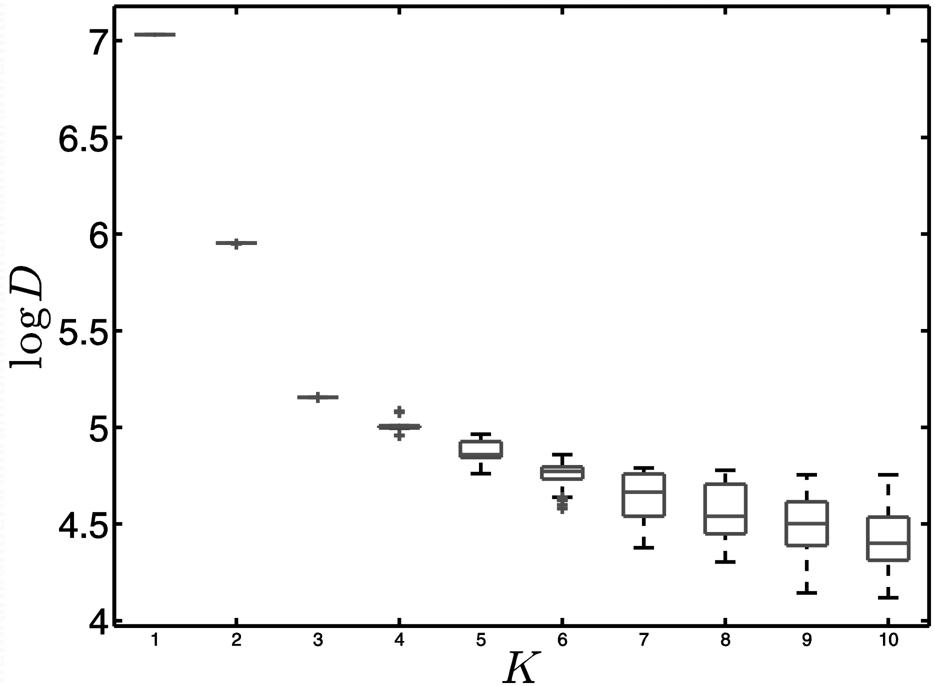
  - i.e. total distance between all the objects and their respective cluster means

- Global optimum vs local optimum
  - Depends on random initialization

- Repeat runs of k-means clustering, pick one with lowest $\mathcal{D}$

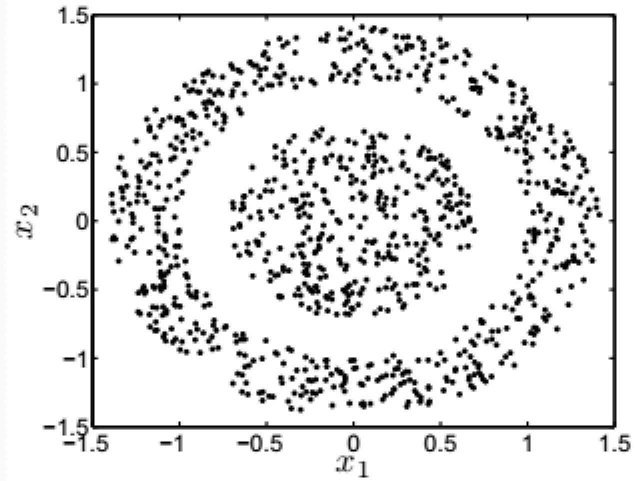
How to choose K?

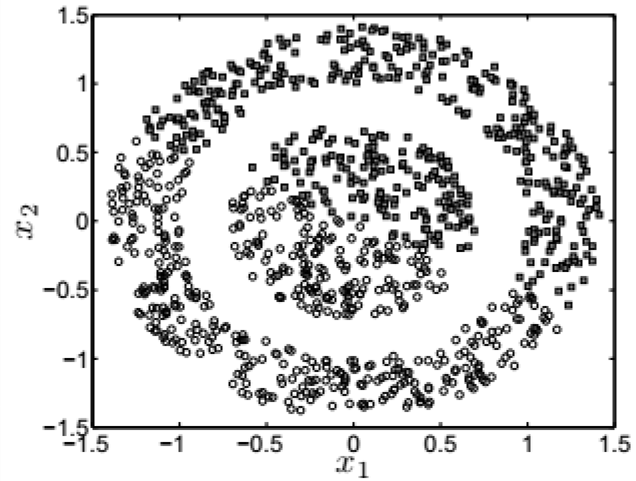
- Model selection
- 50 repeat runs
- $\mathcal{D}$ decreases with increasing K
  - Why?
- $N = K$?
- K often depends on application/problem/data



K-means limitations

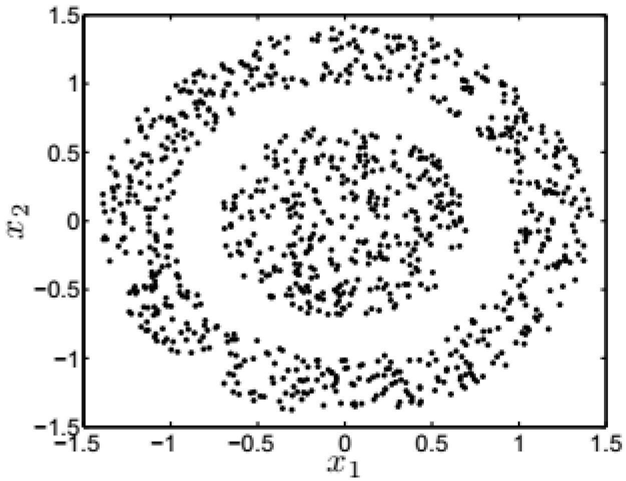
- K-means assumes clear cluster structure
  - What if it's lacking?
- Cluster means?
- Do objects conform to the similarity concept that K-means clustering heavily relies on?
- Outer cluster can't be represented by a single cluster mean





Data transformation

- How about transforming data in to a new space where it can be separable?
  - $\boldsymbol{x} \to \phi(\boldsymbol{x})$



- For this example, consider $\phi(\boldsymbol{x}) = {x_n^{(1)}}^2 + {x_n^{(2)}}^2$?
- In practice, there is a very neat trick which doesn't even require the explicit data transformation
  - Kernel trick
- Kernel function (a kind of similarity measure)
  - A function that is equivalent to the dot product of vectors in the transformed space
  - $k(\boldsymbol{x}_m, \boldsymbol{x}_n) = \phi(\boldsymbol{x}_m)^T \phi(\boldsymbol{x}_n)$

Kernel function

- There is a number of off-the-shelf kernels that have been shown to work well

- Linear kernel
  - $k(\boldsymbol{x}_m, \boldsymbol{x}_n) = \boldsymbol{x}_m{}^T \boldsymbol{x}_n$

- Gaussian kernel
  - $k(\boldsymbol{x}_m, \boldsymbol{x}_n) = exp\{-\gamma(\boldsymbol{x}_m - \boldsymbol{x}_n)^T(\boldsymbol{x}_m - \boldsymbol{x}_n)\}$
  - $k(\boldsymbol{x}_m, \boldsymbol{x}_n) = exp\{-\gamma\|\boldsymbol{x}_m - \boldsymbol{x}_n\|^2\}$

- Polynomial kernel
  - $k(\boldsymbol{x}_m, \boldsymbol{x}_n) = (\boldsymbol{x}_m{}^T \boldsymbol{x}_n + c)^\beta$

Kernelized k-means

- Can we kernelize k-means?
- *Kernel function*
  - A function that corresponds to inner product of vectors in some other transformed space
- As long as an algorithm has data appearing only in inner products in model learning, kernels can be used

$$k(\boldsymbol{x}_m, \boldsymbol{x}_n) = \phi(\boldsymbol{x}_m)^T \phi(\boldsymbol{x}_n)$$

Kernelized k-means

- Distance estimate (squared Euclidean)

$$D_{nk} = (\boldsymbol{x}_n - \boldsymbol{\mu}_k)^T(\boldsymbol{x}_n - \boldsymbol{\mu}_k)$$

- Cluster mean calculation

$$\boldsymbol{\mu}_k = \frac{1}{N_k}\sum_{n=1}^{N_k}\boldsymbol{x}_n$$

- Distance can be estimated as:

$$D_{nk} = \left(\boldsymbol{x}_n - \frac{1}{N_k}\sum_{m=1}^{N_k}\boldsymbol{x}_m\right)^T\left(\boldsymbol{x}_n - \frac{1}{N_k}\sum_{m=1}^{N_k}\boldsymbol{x}_m\right)$$

$$D_{nk} = \boldsymbol{x}_n^T\boldsymbol{x}_n - \frac{2}{N_k}\sum_{m=1}^{N_k}\boldsymbol{x}_m^T\boldsymbol{x}_n + \left(\frac{1}{N_k}\right)^2\sum_{m=1}^{N_k}\sum_{l=1}^{N_k}\boldsymbol{x}_m^T\boldsymbol{x}_l$$

Kernelized k-means

- Distance can be estimated as:

$$D_{nk} = \boldsymbol{x}_n^T \boldsymbol{x}_n - \frac{2}{N_k} \sum_{m=1}^{N_k} \boldsymbol{x}_m^T \boldsymbol{x}_n + \left(\frac{1}{N_k}\right)^2 \sum_{m=1}^{N_k} \sum_{l=1}^{N_k} \boldsymbol{x}_m^T \boldsymbol{x}_l$$

- With kernel trick, dot products could be replaced:

$$D_{nk} = k(\boldsymbol{x}_n, \boldsymbol{x}_n) - \frac{2}{N_k} \sum_{m=1}^{N_k} k(\boldsymbol{x}_m, \boldsymbol{x}_n) + \left(\frac{1}{N_k}\right)^2 \sum_{m=1}^{N_k} \sum_{l=1}^{N_k} k(\boldsymbol{x}_m, \boldsymbol{x}_l)$$

- Let's recall $\boldsymbol{\mu}_k$ will be

$$\boldsymbol{\mu}_k = \frac{1}{N_k} \sum_{n=1}^{N_k} \phi(\boldsymbol{x}_n)$$

but what's $\phi(\boldsymbol{x}_n)$?

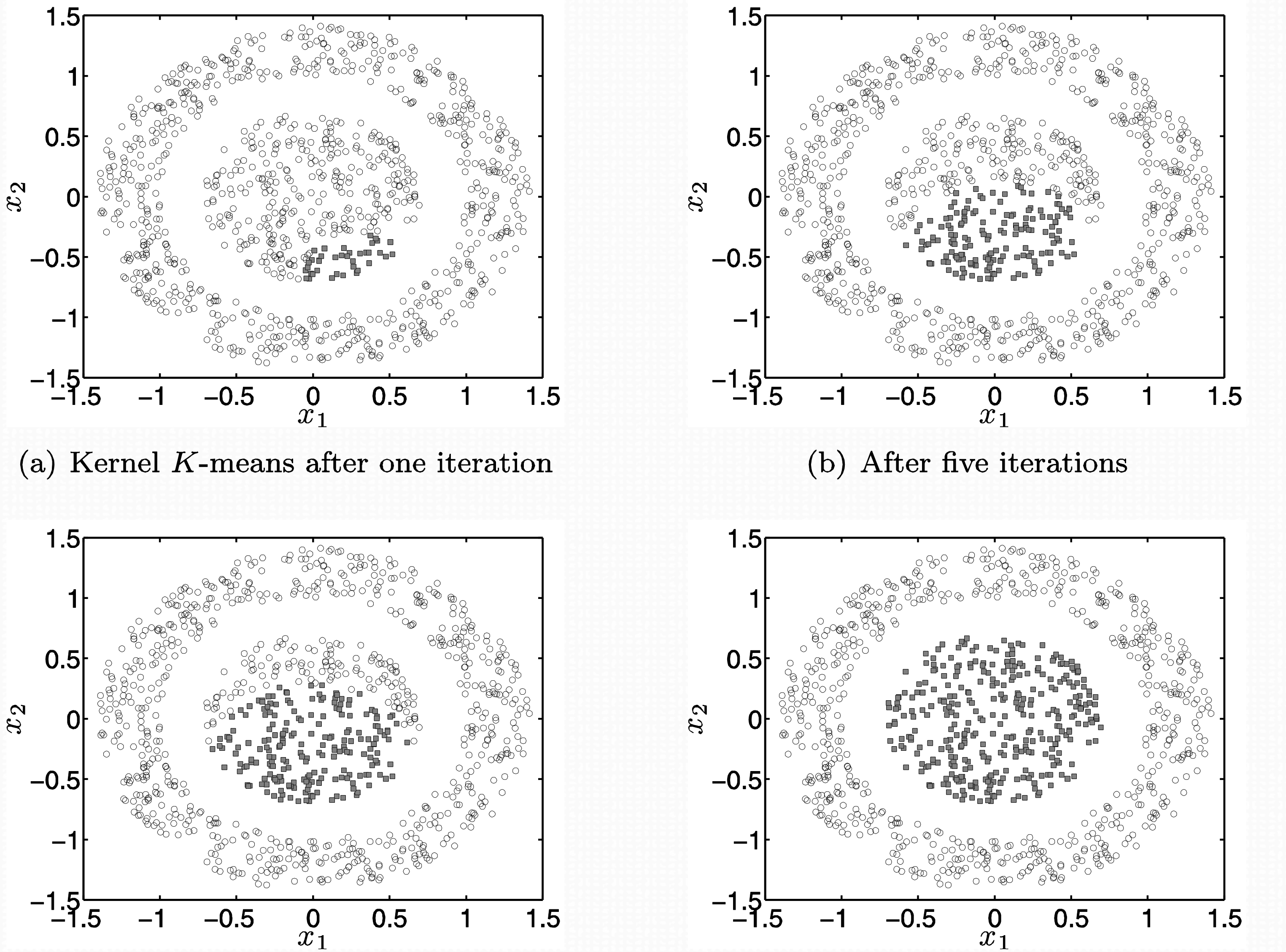
Kernelized k-means

- Algorithm:
  1. Randomly assign each object $\boldsymbol{x}_n$ to one of K clusters
  2. Compute $D_{n1}, D_{n2}, \ldots, D_{nK}$ for each object $\boldsymbol{x}_n$, using the kernel trick

$$D_{nk} = k(\boldsymbol{x}_n, \boldsymbol{x}_n) - \frac{2}{N_k}\sum_{m=1}^{N_k} k(\boldsymbol{x}_m, \boldsymbol{x}_n) + \left(\frac{1}{N_k}\right)^2 \sum_{m=1}^{N_k}\sum_{l=1}^{N_k} k(\boldsymbol{x}_m, \boldsymbol{x}_l)$$

  3. Assign the object $\boldsymbol{x}_n$ to cluster k with lowest distance D_{nk}
     - i.e. assign $\boldsymbol{x}_n$ to its closest cluster
  4. Stop iterations if assignments don't change, or maximum number of iterations reached, else jump to step 2

Kernelized k-means



(a) Kernel K-means after one iteration

(b) After five iterations

(c) After 10 iterations

(d) At convergence (30 iterations)

Kernelized k-means

- Brings flexibility to simple k-means algorithm
- Need to set additional kernel parameters
- Sensitive to initialization
  - Often, it's good to run standard k-means algorithm before kernelized k-means for decent initialization

K-means: limitations

- Sensitive to initialization
- Computationally expensive
- Often needs repeated runs
- Makes hard assignments (i.e. each object can belong only to one cluster)
  - Soft assignment (each object has a probability of belonging to each cluster)

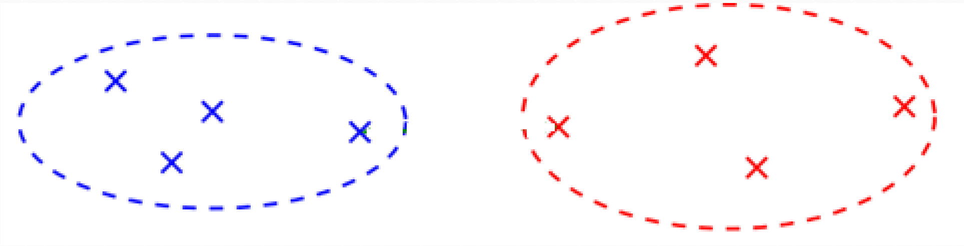
Hierarchical clustering

- Bottom-up (agglomerative) approach
  - Each object starts as a singleton cluster
  - Two most similar clusters are merged iteratively
  - Stop when all objects are in same cluster

- Top-down (divisive) approach
  - All objects belong to same cluster
  - "outsider" objects from least cohesive cluster are removed iteratively
  - Stop when each object is a singleton cluster

Clustering: similarity measure

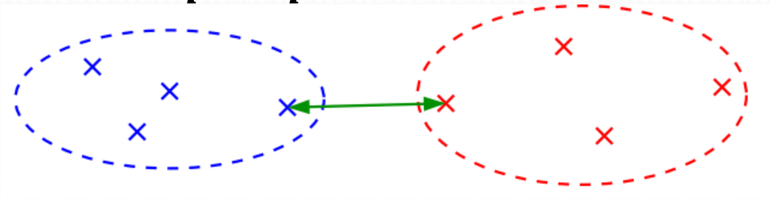
- We know how to estimate similarity (or distance between pair of objects
- How do we measure similarity between a pair of clusters (R and S)?



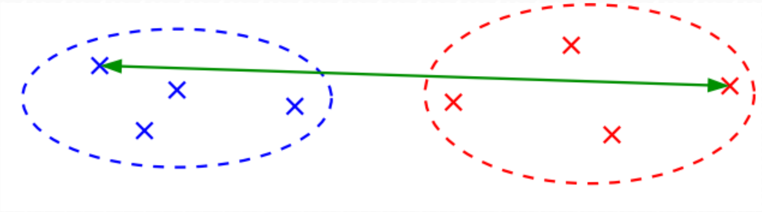
- Linkage
  - Min/single link
  - Max/complete link
  - Average link

Clustering: similarity measure

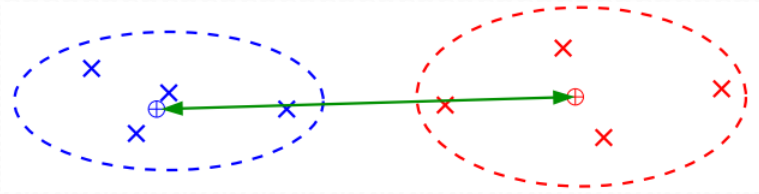
- Min/single link: $D(R,S) = \min_{\boldsymbol{p}\in R, \boldsymbol{q}\in S} D(\boldsymbol{p},\boldsymbol{q})$
  - Clusters can get very large



- Max/complete link: $D(R,S) = \max_{\boldsymbol{p}\in R, \boldsymbol{q}\in S} D(\boldsymbol{p},\boldsymbol{q})$
  - Results in small, round clusters



- Average link: $D(R,S) = \frac{1}{|R||S|} \sum_{\boldsymbol{p}\in R, \boldsymbol{q}\in S} D(\boldsymbol{p},\boldsymbol{q})$
  - Compromise between min and max links



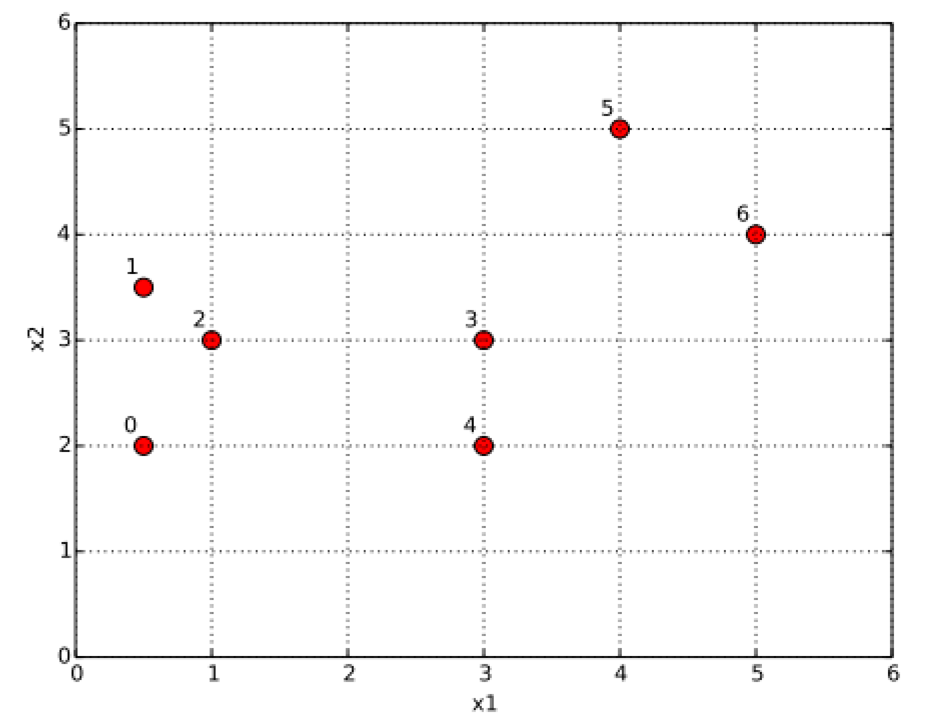
Hierarchical clustering

- Hierarchical agglomerative clustering (HAC) is a bottom-up clustering strategy
  - Each object is a singleton cluster
  - Successively merge closest pair of clusters together, till all clusters merge

1. Consider each object as a singleton cluster
2. Compute distance between all pairs of clusters
3. Remove the closest pair of clusters from the data, and replace them as a single cluster
4. If not all objects belong to same single cluster, jump to step 2

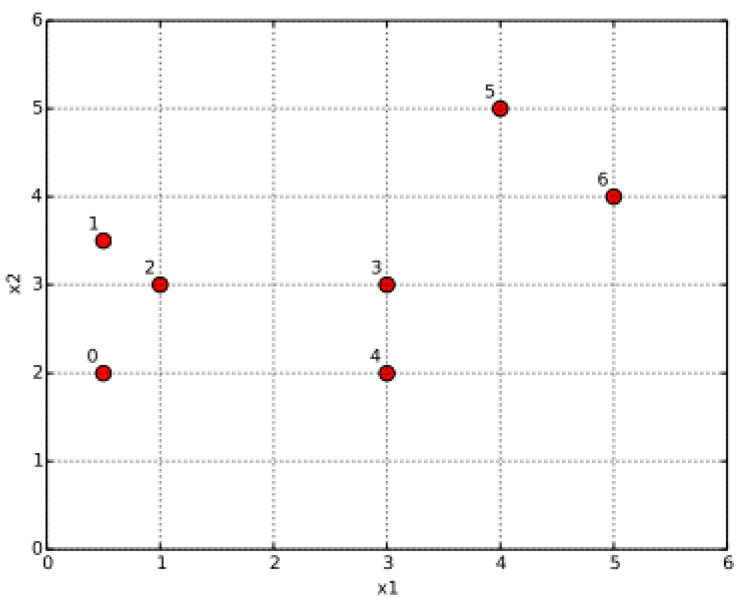
Hierarchical clustering

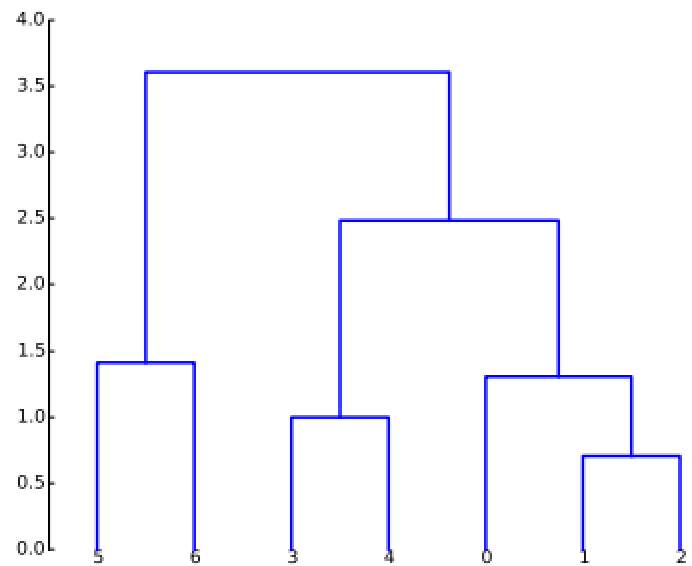
- Hierarchically cluster



Hierarchical clustering

- The results of hierarchical clustering can be visualized as a _dendrogram_ where the height of each “junction” in the dendrogram represents the distance between the pair of clusters
- Hierarchical clustering generates all number of clusters
  - Cut horizontally across the dendrogram





Flat clustering vs hierarchical clustering

- Flat clustering (K-means) results in a single partitioning of objects
- Hierarchical clustering can produce different partitioning results depending on the level-of-resolution we are looking at

- Flat clustering needs to know K in advance
- Hierarchical clustering doesn't need to know K

Flat clustering vs hierarchical clustering

- Flat clustering is usually more efficient (run-time) – if we know K
  - NxK distance computations at each iteration
- Hierarchical clustering can be slow (has to make several merge/split decisions)
  - Choosing 2 closest objects from N objects requires $\frac{N!}{2!(N-2)!}$ => $N * (N-1)/2$ calculations just for first pairing
- No clear consensus on which of the two produces better clustering
  - Results depend upon nature of data

Summary

- Two common clustering methods
  - By associating an object with its nearest cluster mean
  - By successively merging smaller clusters to hierarchically form larger clusters
- Free choices
  - How many clusters?
  - What distance metric?
  - What linkage method?
- Optimal combination is often found by trial-and-error
- Kernelized k-means

Exercise (ungraded)

- Use K-means algorithm and Euclidean distance metric to cluster the following 2-dimensional objects in to 3 clusters
  - O1 (2,10), O2 (2,5), O3 (8,4), O4 (5,8), O5 (7,5), O6 (6,4), O7 (1,2), O8 (4,9)
- Suppose that initial cluster centres are O1, O4, and O7. Run the K-means algorithm steps for 1 iteration and show:
  - The clusters (i.e. objects belonging to each cluster)
  - The centres of new clusters
  - Draw a 10x10 grid with all the 8 objects and show the clusters and centres after the first iteration

Exercise (ungraded)

- Use min/single link to perform agglomerative clustering by showing the dendrogram
  - The data is described by the distance matrix

	A	B	C	D
A	0	1	4	5
B		0	2	6
C			0	3
D				0

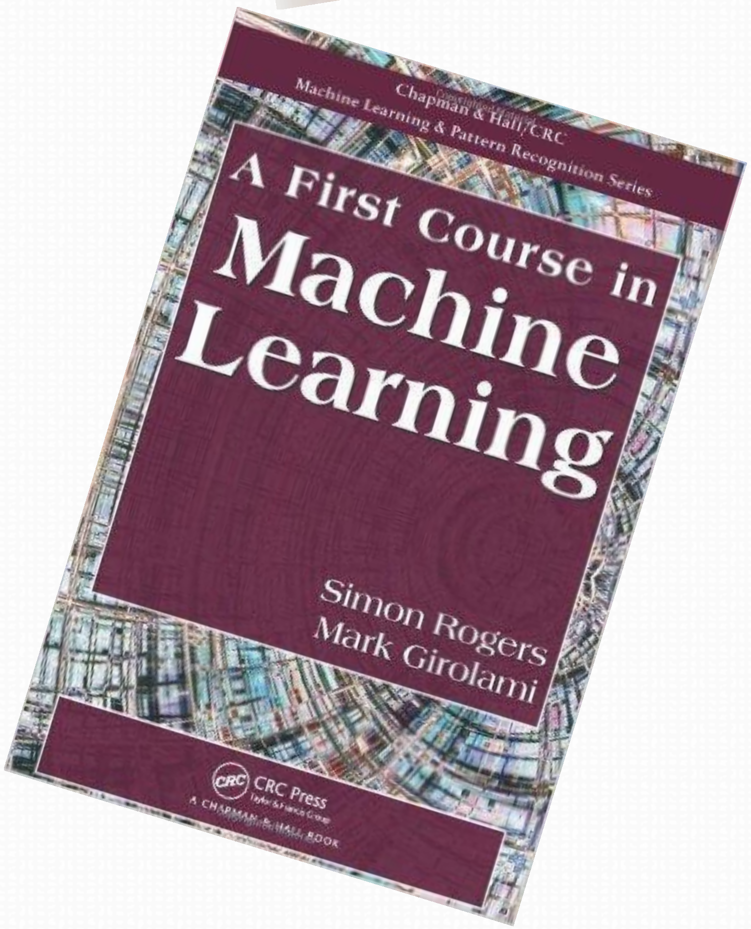
- Use max/complete link to perform agglomerative clustering by showing the dendrogram
  - The data is described by the distance matrix
- Note: the height of each "junction" in the dendrogram represents the distance between the pair of clusters

Exercise (ungraded)

- Use single and complete link agglomerative clustering to cluster the following 8 objects by showing the dendrograms
  - O1 (2,10), O2 (2,5), O3 (8,4), O4 (5,8), O5 (7,5), O6 (6,4), O7 (1,2), O8 (4,9)

Exercise (ungraded)

- Try MATLAB code - kmeansexample.m (from FCML book website)
- Try MATLAB code - kmeansK.m (from FCML book website)
- Try MATLAB code - kernelkmeans.m (from FCML book website)
- Try MATLAB code – kmeans_cluster.m (from Canvas)
- Try MATLAB code – run_kmeans.m (from Canvas)



Author’s material
(Simon Rogers)

Iain Styles

That's All

Thank You