CLUSTERING

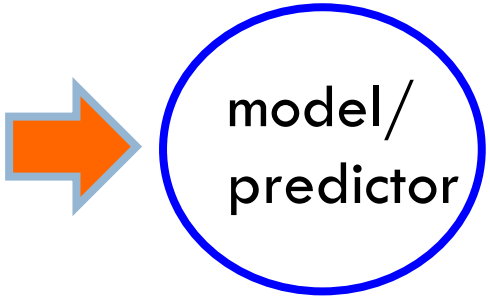
Dr. Kurnianingsih

Supervised learning

label

$label_1$

$label_3$



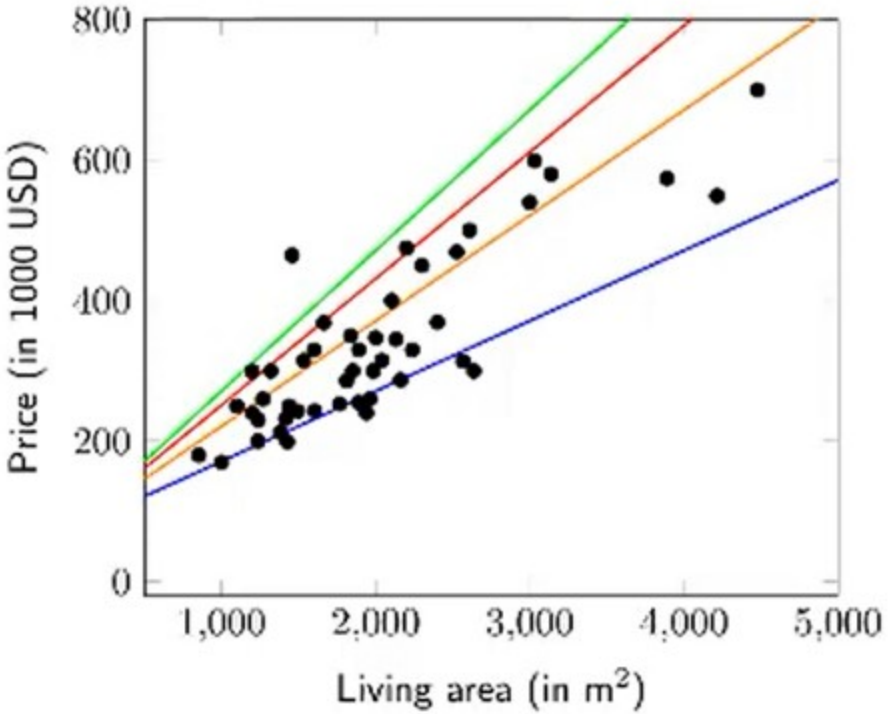
$label_4$

$label_5$

Supervised learning: given labeled examples

Supervised learning

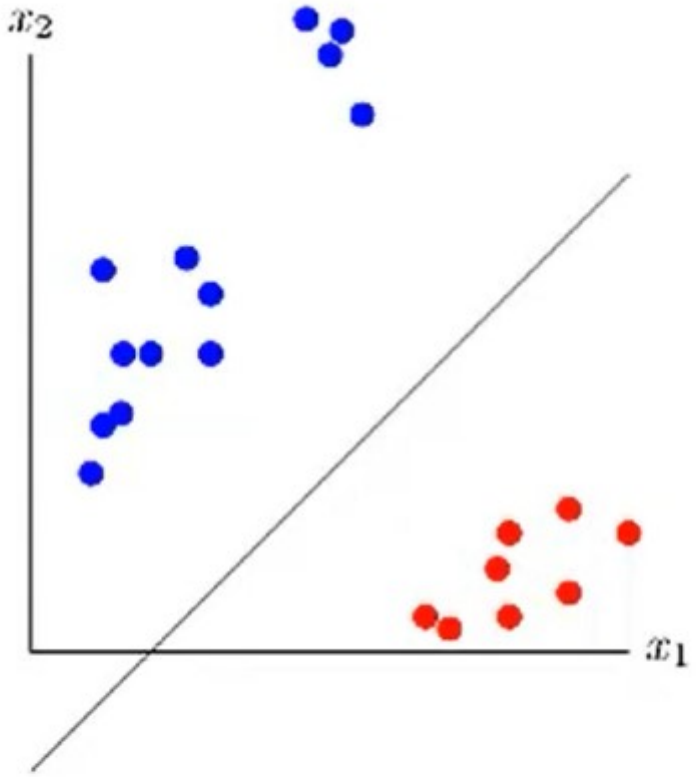
Living area x	Price y
2104	400
1600	330
2400	369
1416	232
3000	540
⋮	⋮



In supervised learning, we are given a training set that consists of $(x^{(i)}, y^{(i)}); i = 1, \ldots, m$. If the target variables y are continuous, we want to find a model h that is a good predictor for y, for example, using linear regression.

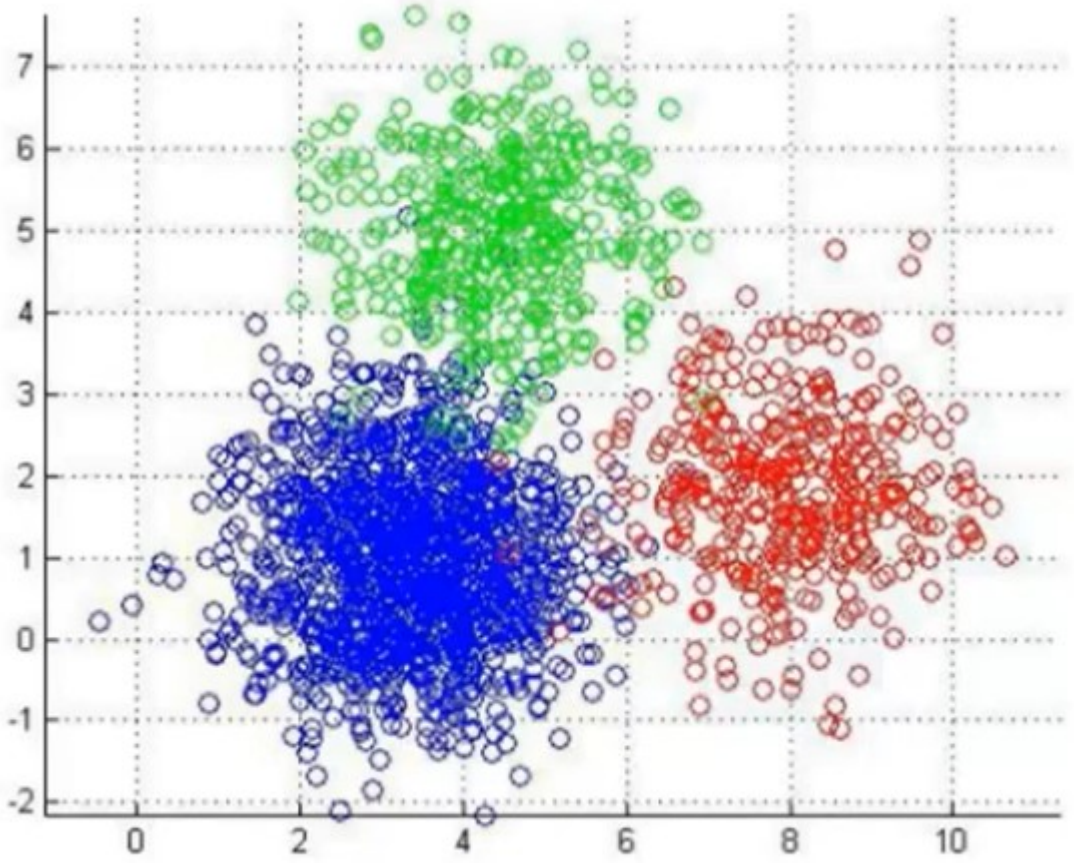
Supervised learning

Cholesterol x_1	Exercise x_2	Status y
100	200	healthy
200	50	unhealthy
90	300	healthy
95	250	healthy
250	30	unhealthy
⋮	⋮	



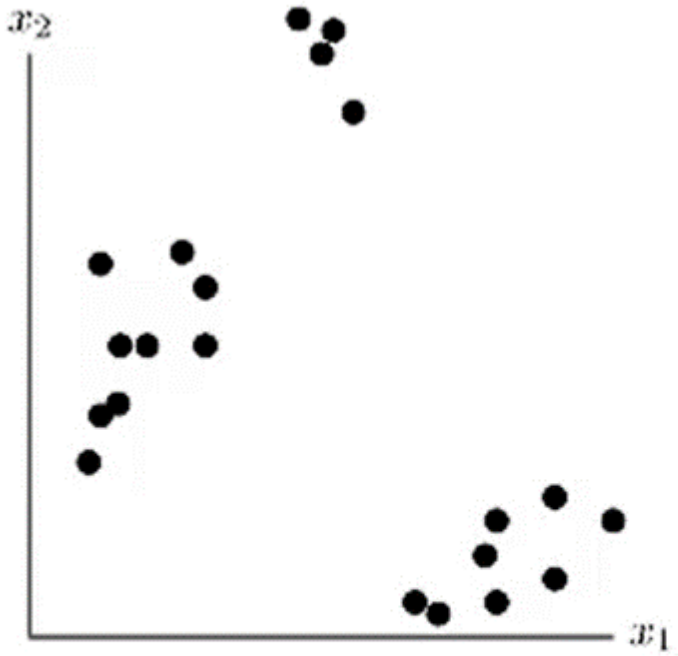
If the target variables y are discrete, for example, classes "healthy" or "unhealthy", we want to classify x into the classes, for example, using SVM.

Unsupervised learning



Unsupervised learning

Cholesterol x_1	Exercise x_2
100	200
200	50
90	300
95	250
250	30
⋮	⋮



In unsupervised learning, our training set has no target variable y, that is, $\{x^{(1)}, \ldots, x^{(m)}\}$, and thus regression and classification is no longer of interest.

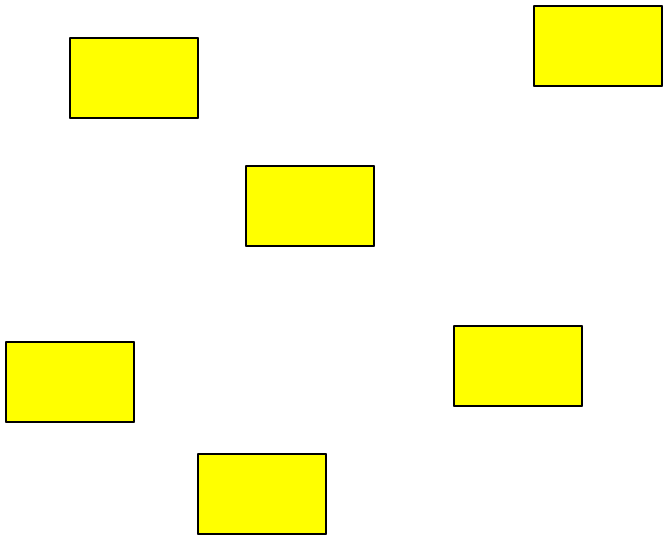
Unsupervised learning





Unupervised learning: given data, i.e. examples, but no labels

Unsupervised learning



Given some example without labels, do something!

Unsupervised learning applications

learn clusters/groups without any label

customer segmentation (i.e. grouping)

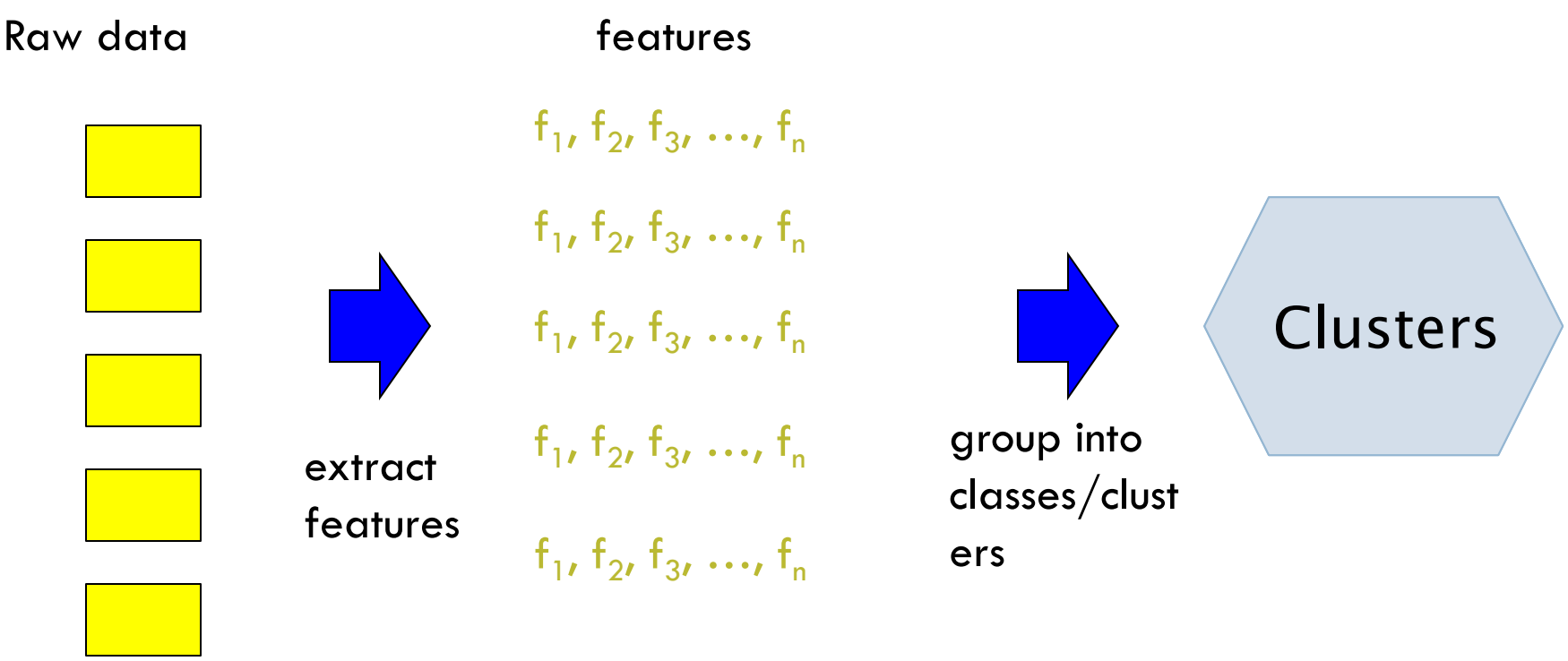
image compression

bioinformatics: learn motifs

find important features

…

Unsupervised learning: clustering



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

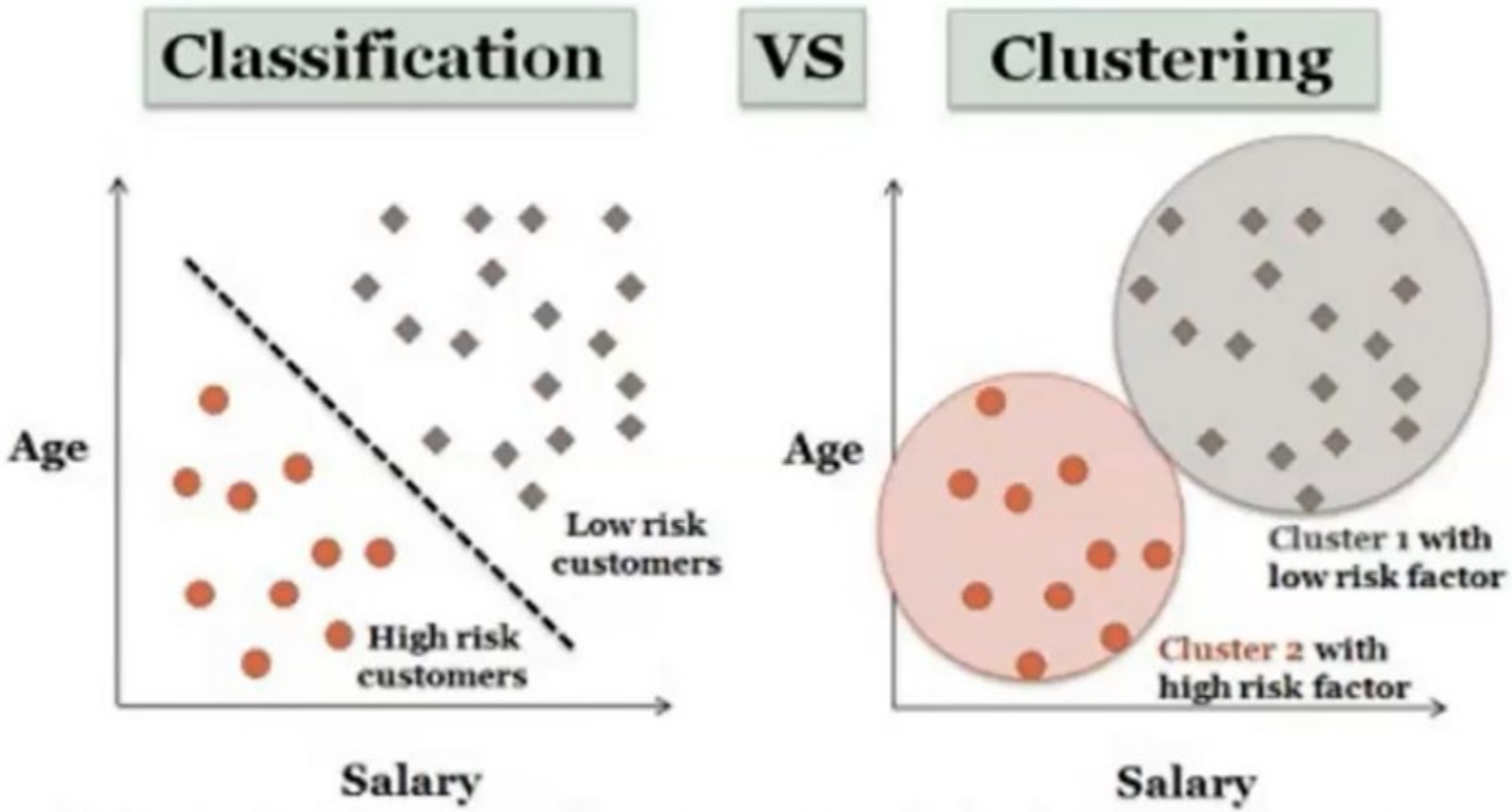
Unsupervised learning: modeling

Most frequently, when people think of unsupervised learning they think clustering

Another category: learning probabilities/parameters for models without supervision

- Learn a translation dictionary
- Learn a grammar for a language
- Learn the social graph

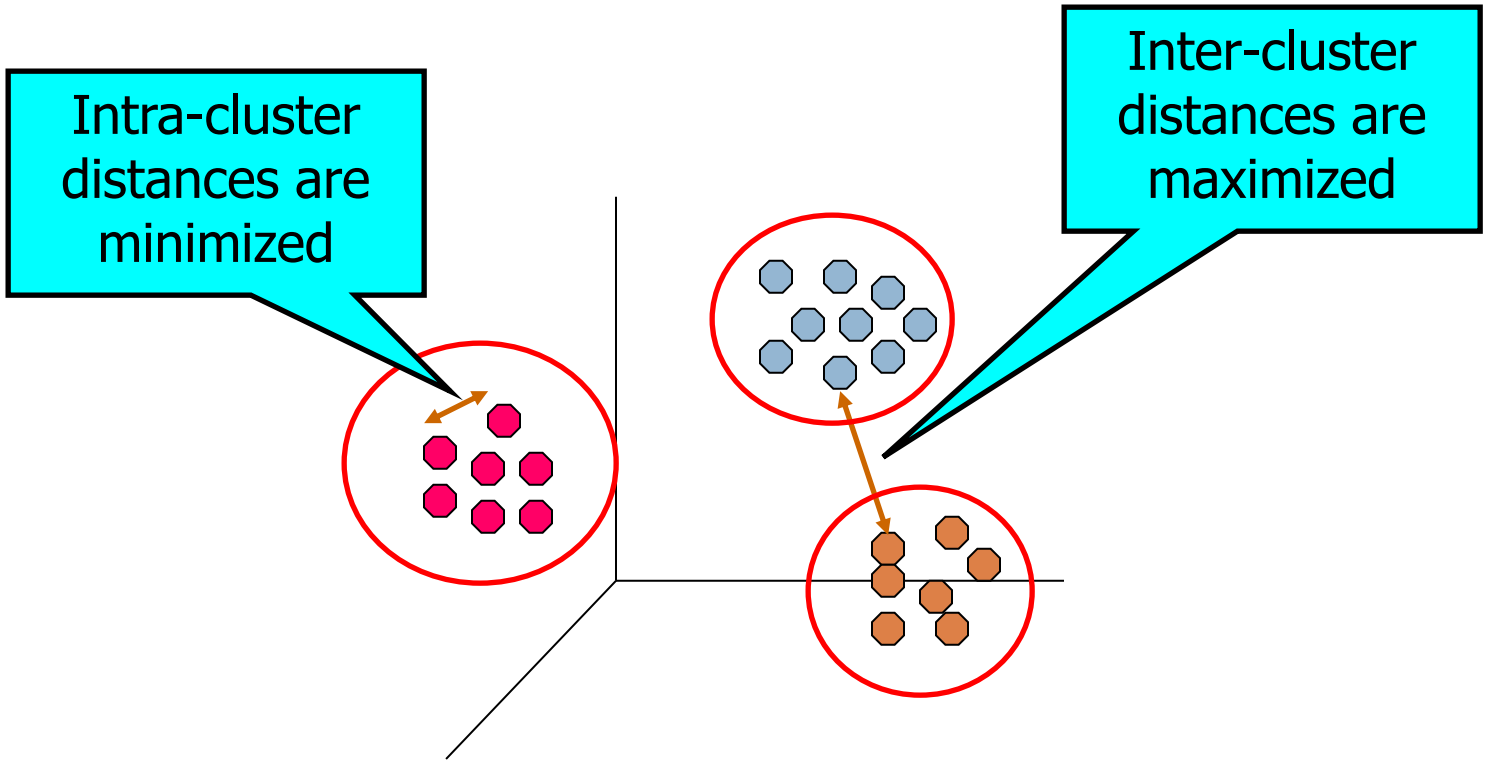
Supervised vs Unsupervised learning



Risk classification for the loan payees on the basis of customer salary

What is a Clustering?

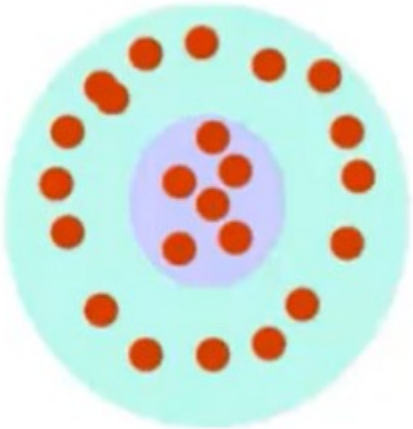
- In general a grouping of objects such that the objects in a group (cluster) are similar (or related) to one another and different from (or unrelated to) the objects in other groups

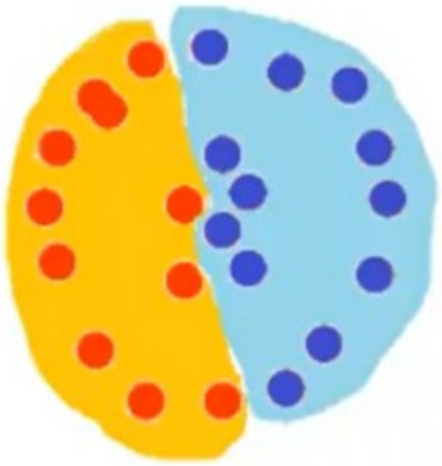


Clustering

The organization of unlabled data into similarity groups called clusters.

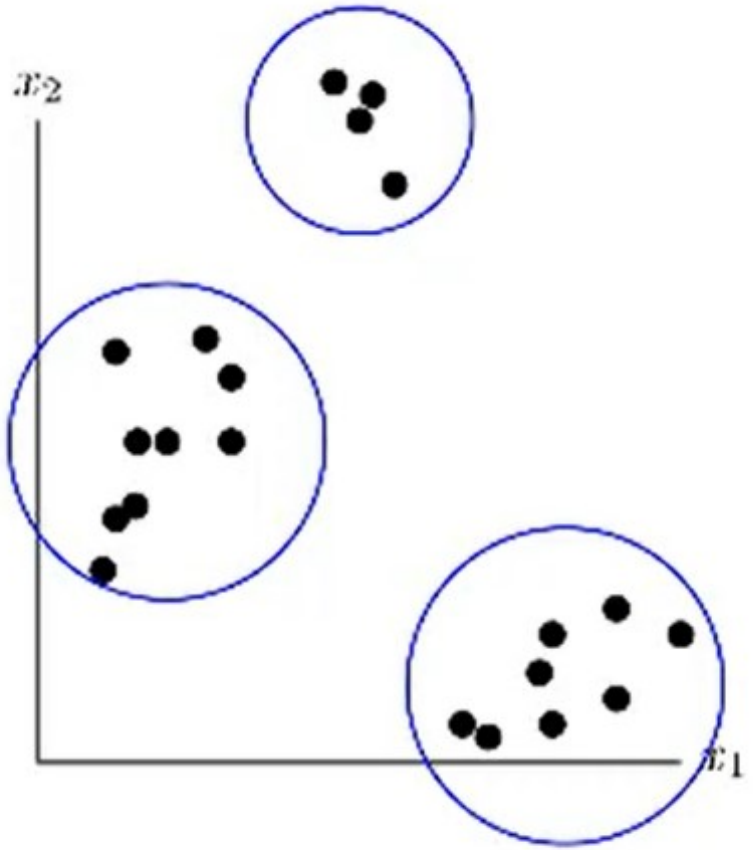
A cluster: a collection of data items which are "similar" between them and "dissimilar" to data items in other clusters.



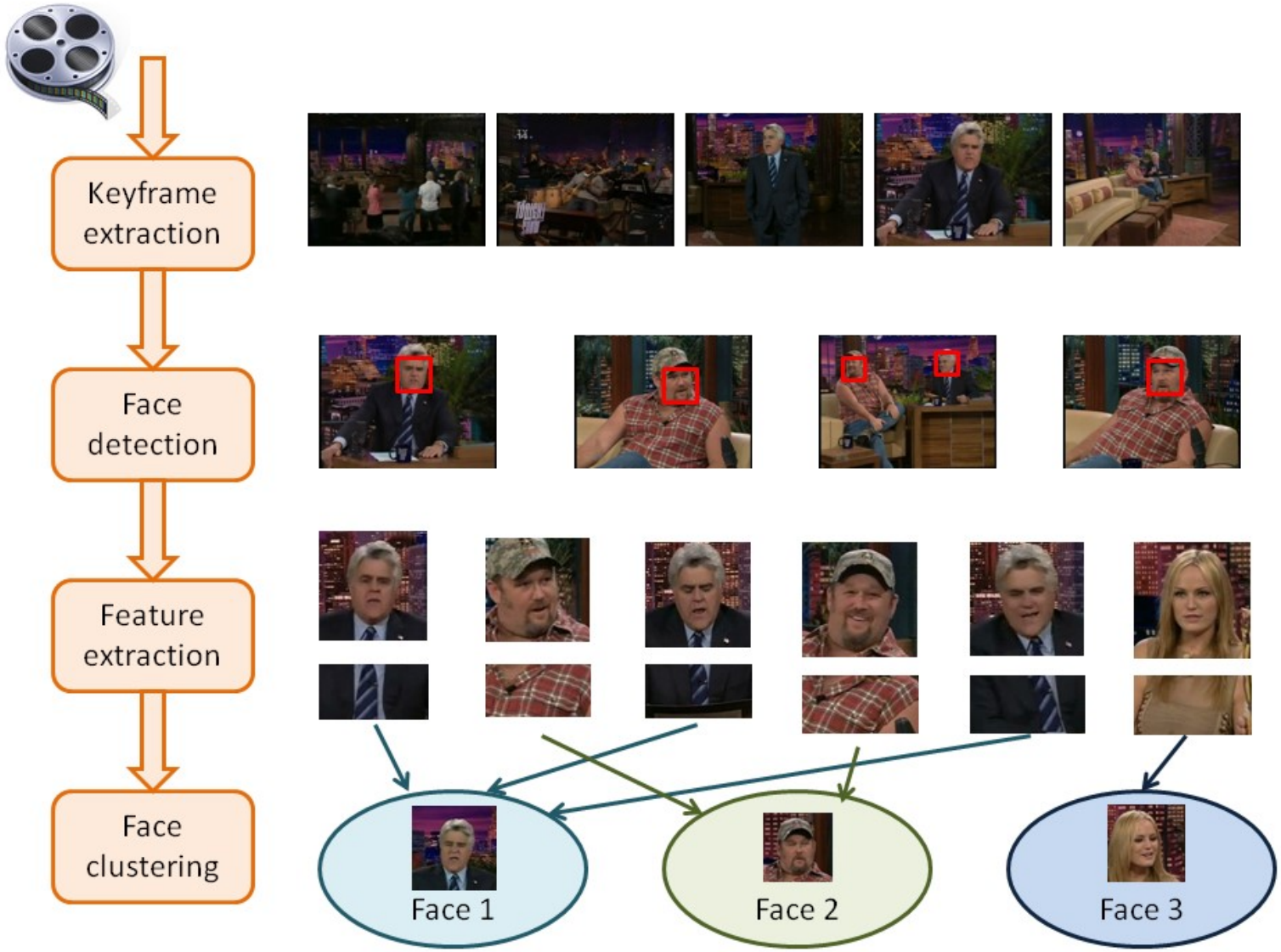


Unsupervised

- For example, **clusters** or smaller groups in a data set
- The idea: partitioning data into distinct groups
  - observations within each group are **similar**
  - observations in different groups are **different**

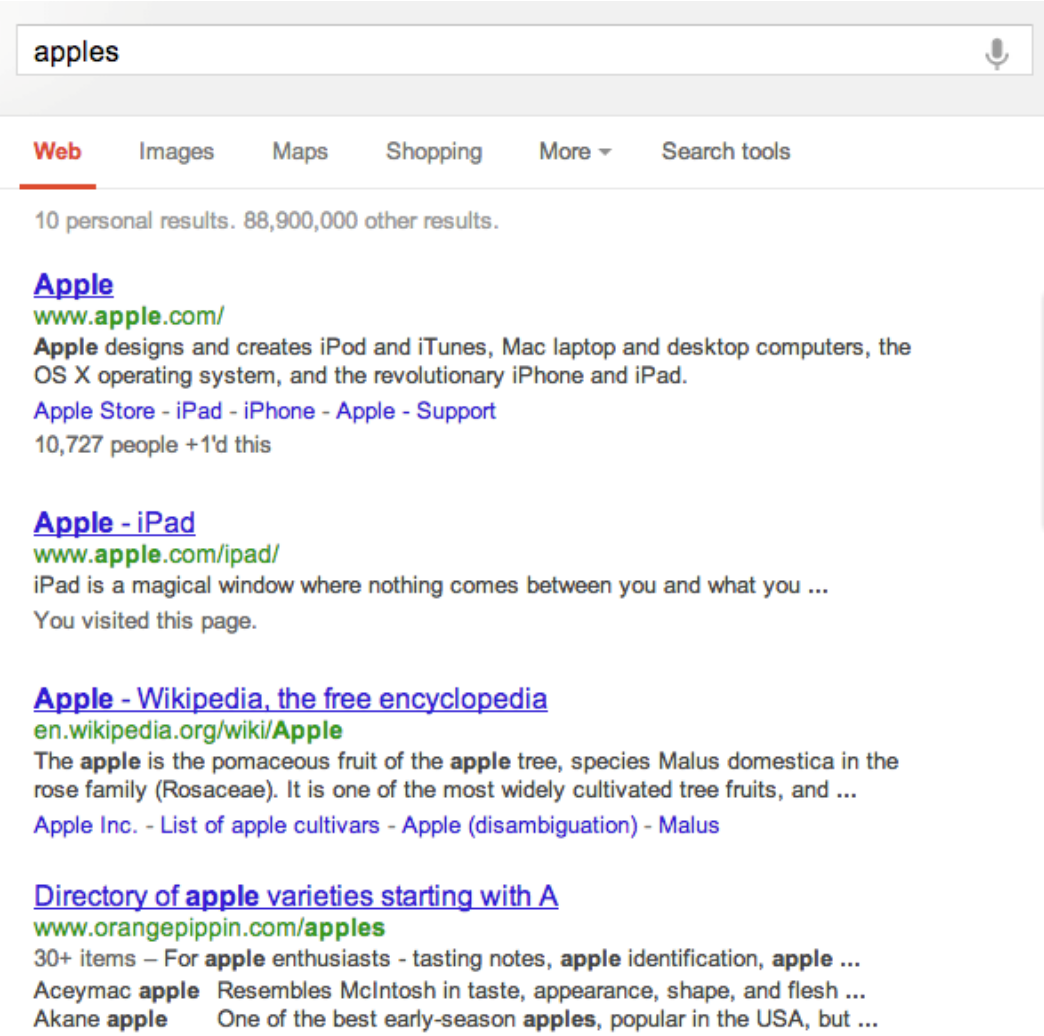


Face Clustering



Face clustering

Search result clustering



Google News

Google

News

Top Stories

- Iran
- Xbox One
- Tarun Tejpal
- Manny Pacquiáo
- Ukraine
- Kabul
- New England Patriots
- Latvia
- Derrick Rose
- Doctor Who

+ Xbox One

E! Online

See realtime coverage

Console Wars 2013: Microsoft's Xbox One vs. Sony's PlayStation 4

E! Online - 1 hour ago

The future is now! Last week, Sony released its next generation console, PlayStation 4. This weekend, Microsoft drops the much touted all-in-one media device, Xbox One. We've been geeking out over the two new systems, and compiling a report on the new ...

Xbox One sales exceed one million in first 24 hours Joystiq - by David Hinkle

Xbox One vs. PS4: A Guide to Making the Toughest Gaming Decision in Years
ABC News - by Joanna Stern

Related
Microsoft »


Wall Street Journal


YouTube


YouTube


Washingto...


RedOrbit

Guardian ...


The Guardian

Xbox One and Microsoft websites marred by problems on launch day

The Guardian
9 hours ago


Written by
Jemima Kiss

Microsoft's Xbox One launch was marred by problems with its online services early on Friday which took down the official website Xbox.

Consumers line up for Xbox One

USA TODAY - Nov 23, 2013

Eager video game players lined up at stores across the country awaiting the arrival of Microsoft's Xbox One, a week to the day after rival Sony introduced its PlayStation 4. The console, available for sale tonight at 12:01 a.m.


Product Re...

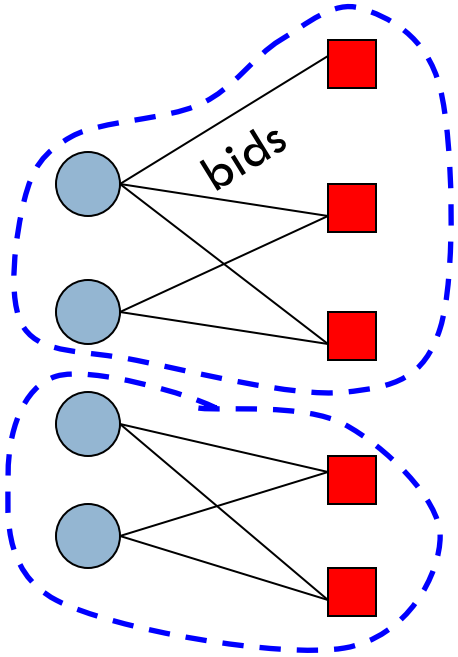
Here are all the Xbox One voice commands

Polygon
9 hours ago


Written by
Megan Farokhmanesh

Microsoft posted a guide to Xbox One voice commands, including how to navigate menus, control volume and multitask, on its Tumblr.

Clustering in search advertising



Find clusters of advertisers and keywords

- Keyword suggestion
- Performance estimation

Types of Clustering

1. **Hierarchical algorithms**: these find successive clusters using previously established clusters.
   1. Agglomerative ("bottom-up"): Agglomerative algorithms begin with each element as a separate cluster and merge them into successively larger clusters.
   2. Divisive ("top-down"): Divisive algorithms begin with the whole set and proceed to divide it into successively smaller clusters.

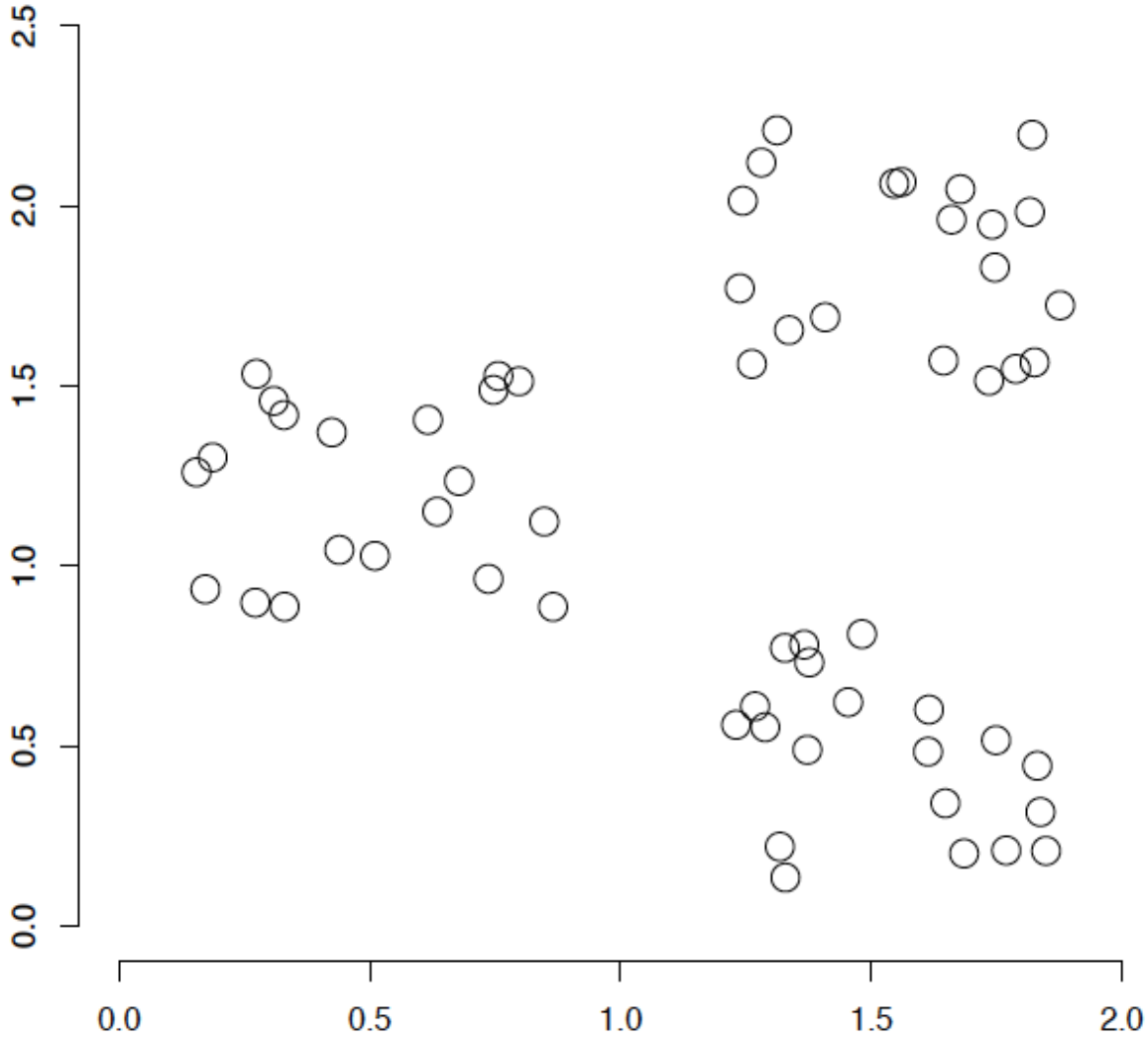
2. **Partitional clustering**: Partitional algorithms determine all clusters at once. They include:

   ***K*-means and derivatives**

   Fuzzy c-means clustering

   QT clustering algorithm

A data set with clear cluster structure



What are some of the issues for clustering?

What clustering algorithms have you seen/used?

Issues for clustering

Representation for clustering

- How do we represent an example
  - features, etc.
- Similarity/distance between examples

Flat clustering or hierarchical

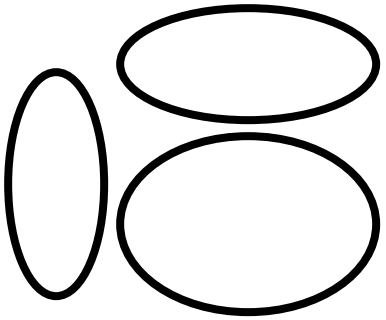
Number of clusters

- Fixed a priori
- Data driven?

Clustering Algorithms

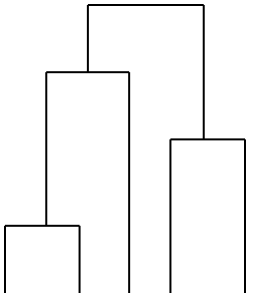
Flat algorithms

- Usually start with a random (partial) partitioning
- Refine it iteratively
  - *K* means clustering
  - Model based clustering
- Spectral clustering



Hierarchical algorithms

- Bottom-up, agglomerative
- Top-down, divisive



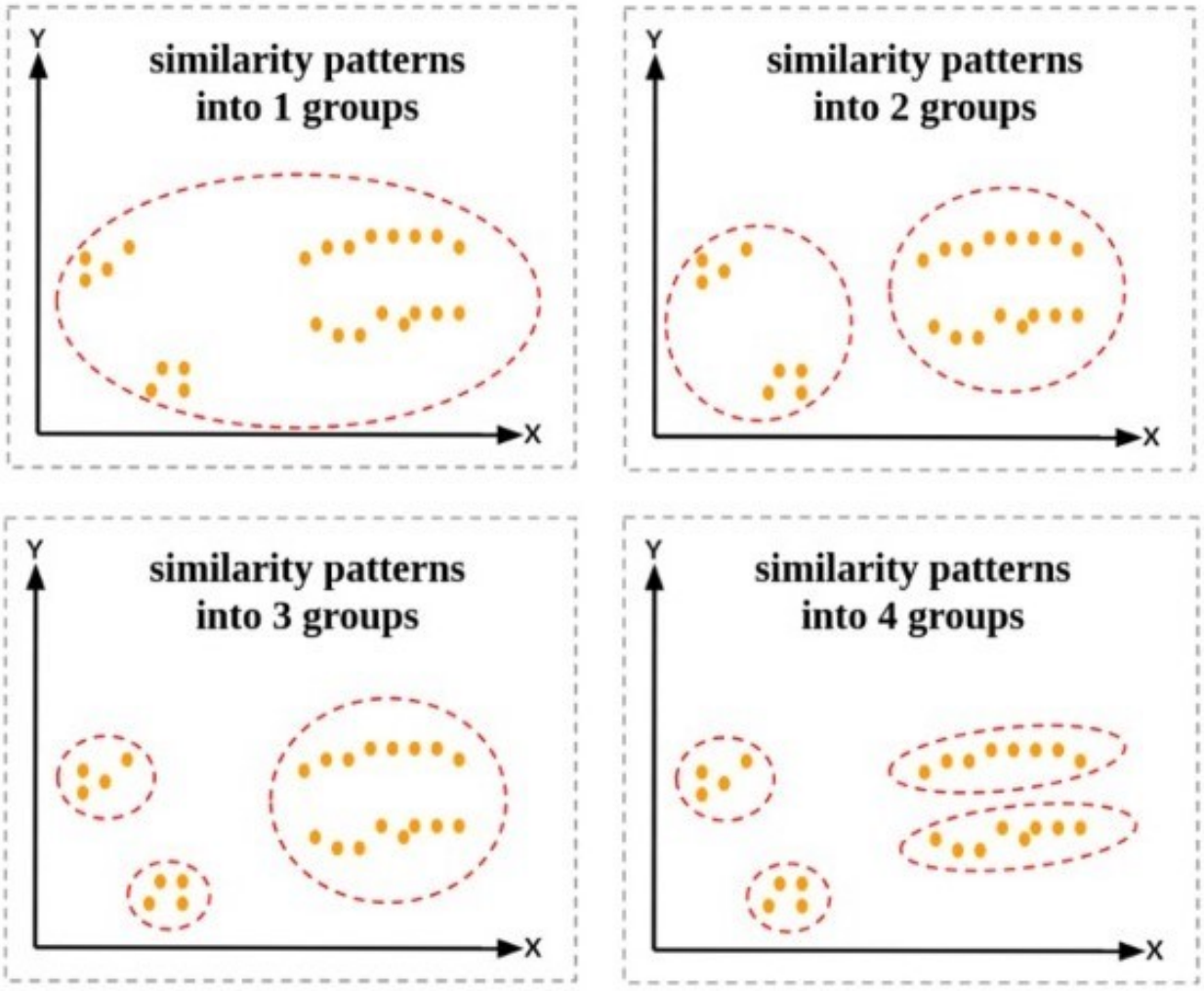
Hard vs. soft clustering

Hard clustering: Each example belongs to exactly one cluster

Soft clustering: An example can belong to more than one cluster (probabilistic)

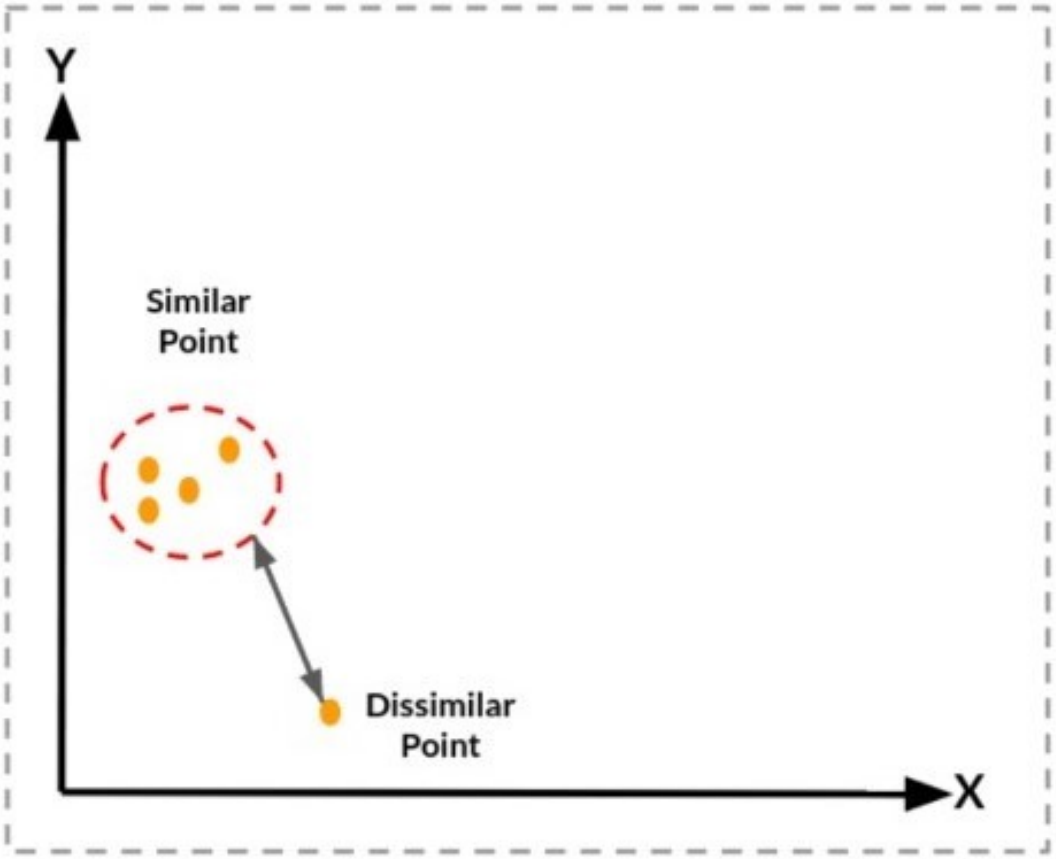
- Makes more sense for applications like creating browsable hierarchies
- You may want to put a pair of sneakers in two clusters: (i) sports apparel and (ii) shoes

Clustering Intuition



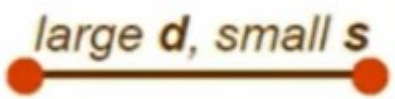
How do we know a point has the same group as another point?

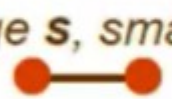
As mentioned above : based on similarity patterns



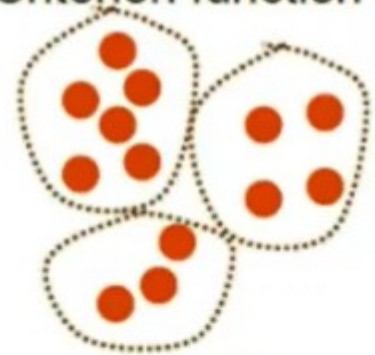
What do we need for Clustering?

1. Proximity measure, *either*
   - similarity measure $s(x_i,x_k)$: large if x_i,x_k are similar
   - dissimilarity(or distance) measure $d(x_i,x_k)$: small if x_i,x_k are similar

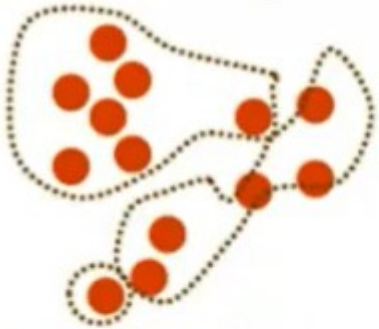




2. Criterion function to evaluate a clustering



good clustering



bad clustering

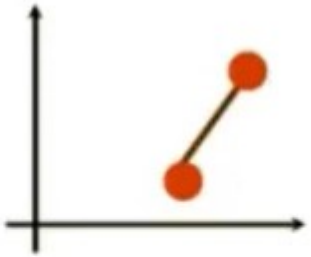
3. Algorithm to compute clustering
   - For example, by optimizing the criterion function

Distance (dissimilarity) measures

- Euclidean distance

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

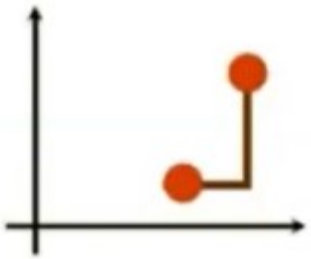
  - translation invariant



- Manhattan (city block) distance

$$d(x_i, x_j) = \sum_{k=1}^{d} \left|x_i^{(k)} - x_j^{(k)}\right|$$

  - approximation to Euclidean distance, cheaper to compute



- They are special cases of **Minkowski distance**:

$$d_p(\mathbf{x}_i, \mathbf{x}_j) = \left(\sum_{k=1}^{m} \left| x_{ik} - x_{jk} \right|^p \right)^{\frac{1}{p}}$$

(p is a positive integer)

Cluster Evaluation

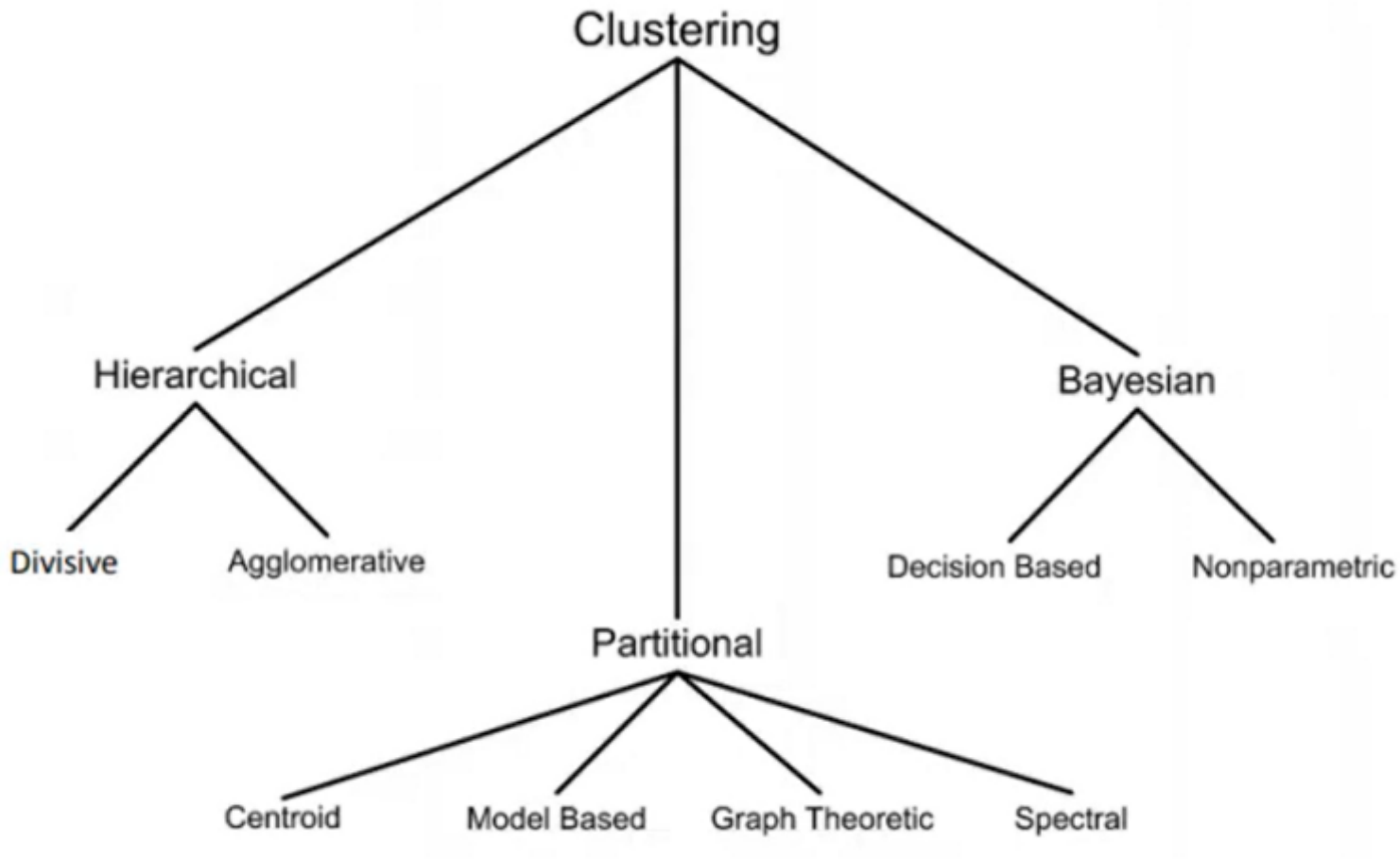
- **Intra-cluster cohesion** (compactness):
  - Cohesion measures how near the data points in a cluster are to the cluster centroid.
  - Sum of squared error (SSE) is a commonly used measure.
- **Inter-cluster separation** (isolation):
  - Separation means that different cluster centroids should be far away from one another.
- In most applications, expert judgments are still the key

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)

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.

Clustering Algorithms

- K-means and its variants
- Hierarchical clustering
- DBSCAN

HIERARCHICAL CLUSTERING

Hierarchical Clustering

Hierarchical versus Flat

Flat methods generate a single partition into k clusters. The number k of clusters has to be determined by the user ahead of time.

Hierarchical methods generate a hierarchy of partitions, i.e.

- a partition P_1 into 1 clusters (the entire collection)
- a partition P_2 into 2 clusters

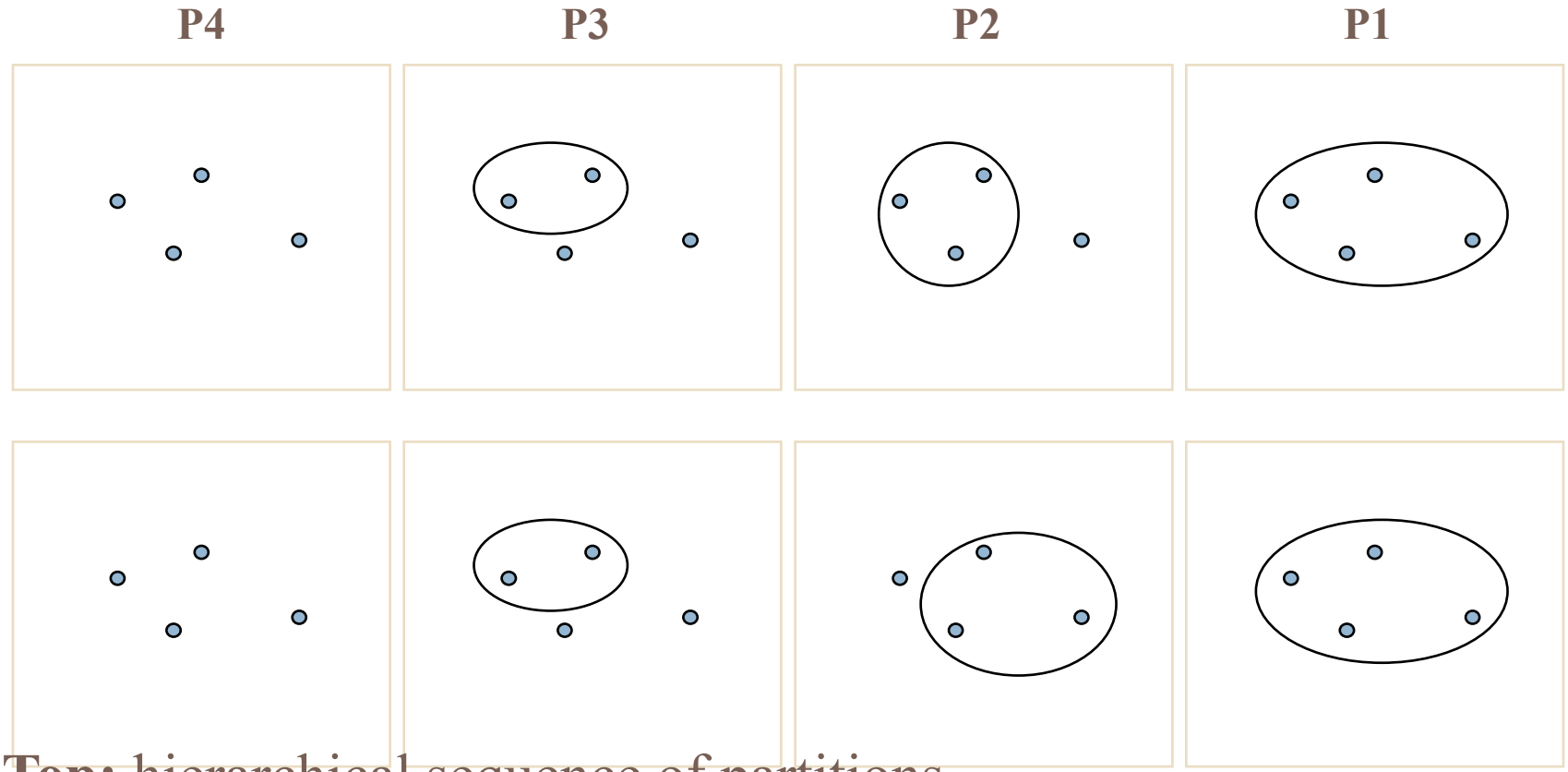
…

- a partition P_n into n clusters (each object forms its own cluster)

It is then up to the user to decide which of the partitions reflects actual sub-populations in the data.

Hierarchical Clustering

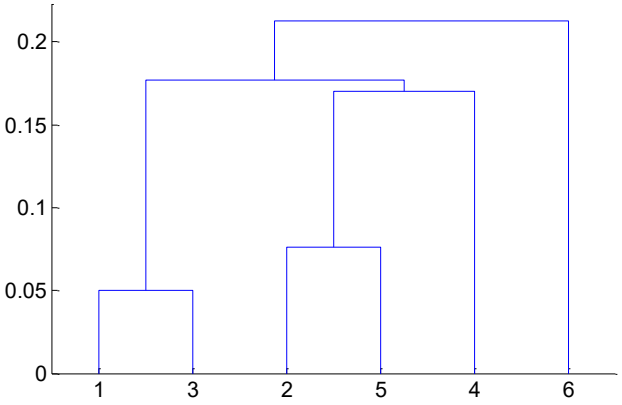
Note: A sequence of partitions is called "hierarchical" if each cluster in a given partition is the union of clusters in the next larger partition.

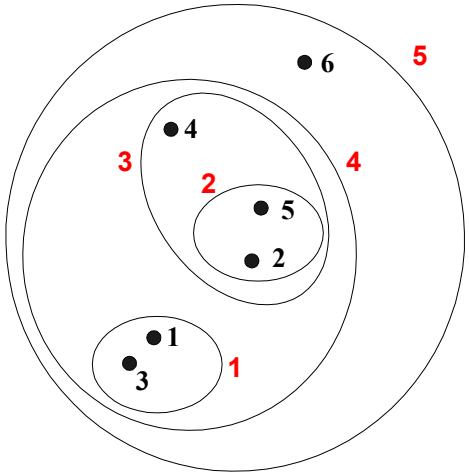


Top: hierarchical sequence of partitions
Bottom: non hierarchical sequence

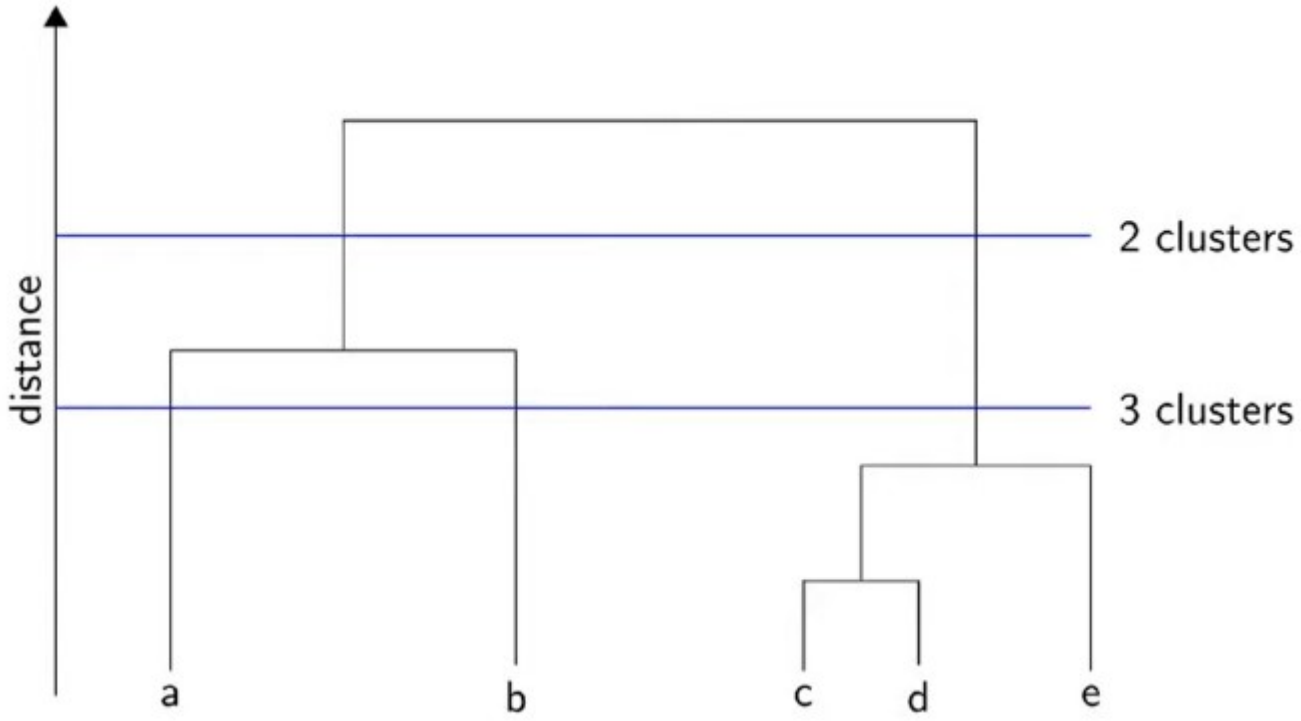
Hierarchical Clustering

- Produces a set of nested clusters organized as a hierarchical tree
- Can be visualized as a dendrogram
  - A tree like diagram that records the sequences of merges or splits





Hierarchical Clustering



- The number of clusters can be determined by cutting the tree horizontally based on the distance of interest
- The tree may represent many useful real-world taxonomies: animal kingdom, etc.

Hierarchical Clustering

There are two approaches:

Agglomerative

- Start with each data points as individual clusters
- Repeatedly merge the closest pair of clusters until only one cluster (or some predetermined number of k clusters) remain

Divisive

- Start with one cluster that includes all data points
- Repeatedly split cluster until each cluster contains a data point (or some predetermined number of k clusters reached)

Strengths of Hierarchical Clustering

- Do not have to assume any particular number of clusters
  - Any desired number of clusters can be obtained by 'cutting' the dendogram at the proper level

- They may correspond to meaningful taxonomies
  - Example in biological sciences (e.g., animal kingdom, phylogeny reconstruction, …)

Agglomerative Clustering Algorithm

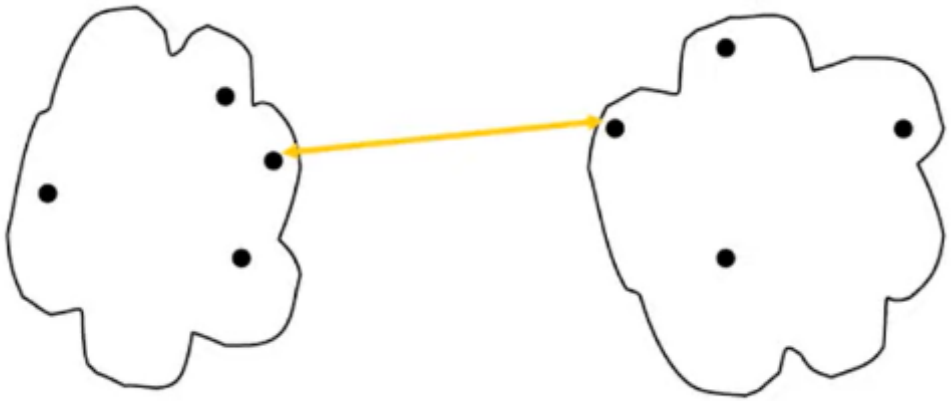
- More popular hierarchical clustering technique
- Basic algorithm is straightforward
  1. Compute the proximity matrix
  2. Let each data point be a cluster
  3. **Repeat**
  4. Merge the two closest clusters
  5. Update the proximity matrix
  6. **Until** only a single cluster remains
- Key operation is the computation of the proximity of two clusters
  - Different approaches to defining the distance between clusters distinguish the different algorithms

Updating Proximity Matrix

There are several methods to update the proximity matrix, such as

- Single linkage
- Complete linkage
- Average linkage

Single Linkage

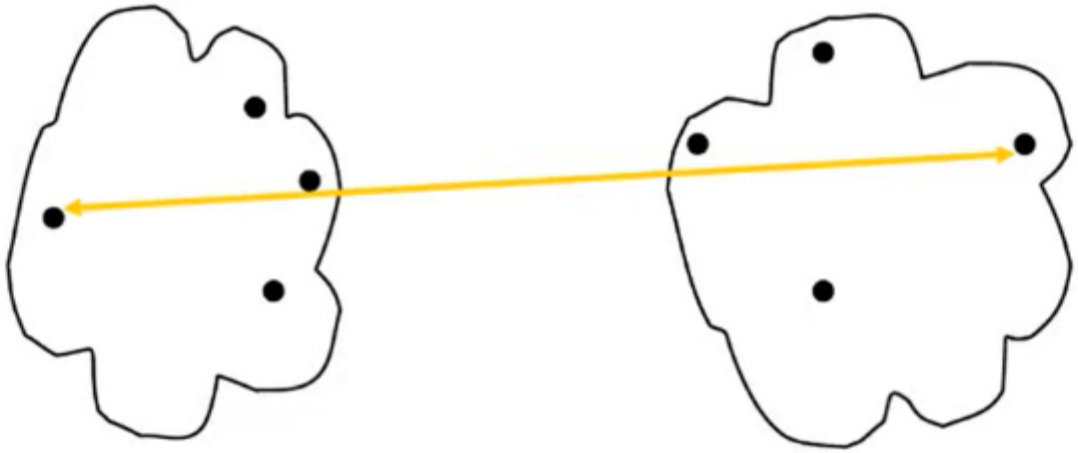


The proximity of two clusters, say C_r and C_s, is equal to the minimum distance between two data points from each cluster, i.e.,

$$d(C_r, C_s) = \min_{x \in C_r, y \in C_s} d(x, y)$$

where $d(x, y)$ is dissimilarity that can be obtained by computing the distance $||x - y||_2$, e.g., Euclidean distance.

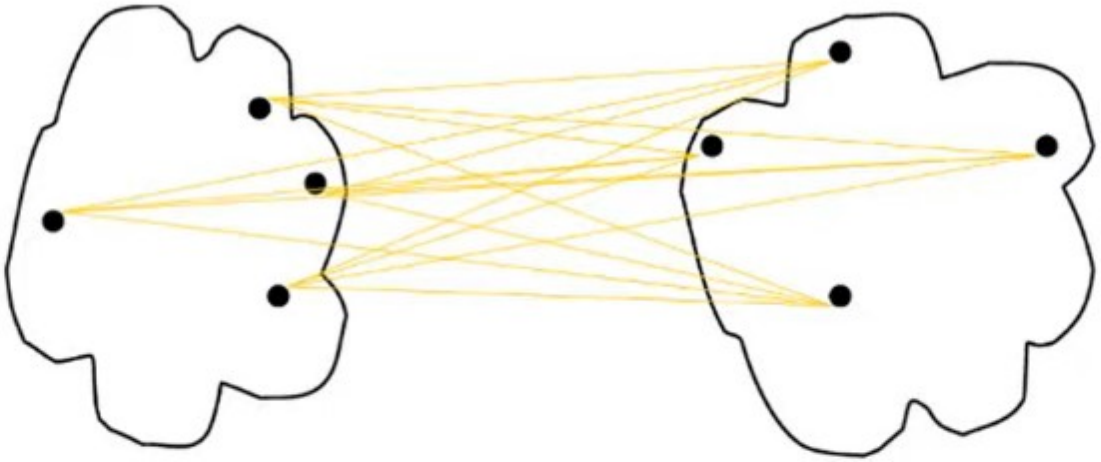
Complete Linkage



The proximity of two clusters, say C_r and C_s, is equal to the maximum distance between two data points from each cluster, i.e.,

$$d(C_r, C_s) = \max_{x \in C_r, y \in C_s} d(x, y).$$

Average Linkage



The proximity of two clusters, say C_r and C_s, is equal to the average distance between the two clusters, i.e.,

$$d(C_r, C_s) = \frac{1}{|C_r||C_s|} \sum_{x \in C_r, y \in C_s} d(x, y).$$

Sample 1: Single Linkage

Given a data set of people ages as follows.

ID	Age
S_1	14
S_2	20
S_3	18
S_4	23
S_5	31
S_6	38

Using single linkage, find clusters from the data above.

Sample 1: Single Linkage

1. Compute the proximity matrix
2. Let each data point be a cluster

	S_1	S_2	S_3	S_4	S_5	S_6
S_1	0	6	4	9	17	24
S_2		0	2	3	11	18
S_3			0	5	13	20
S_4				0	8	15
S_5					0	7
S_6						0

Sample 1: Single Linkage

- While there is more than one cluster
  1. Merge the two closest clusters
  2. Update the proximity matrix

	S_1	S_2	S_3	S_4	S_5	S_6
S_1	0	6	4	9	17	24
S_2		0	2	3	11	18
S_3			0	5	13	20
S_4				0	8	15
S_5					0	7
S_6						0

Sample 1: Single Linkage

- While there is more than one cluster
  1. Merge the two closest clusters
  2. Update the proximity matrix

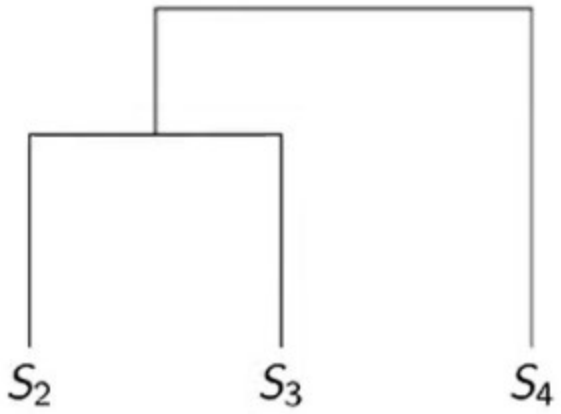
	S_1	$\{S_2, S_3\}$	S_4	S_5	S_6
S_1	0	4	9	17	24
$\{S_2, S_3\}$		0	3	11	18
S_4			0	8	15
S_5				0	7
S_6					0

- $C_1 = S_1, C_2 = \{S_2, S_3\}$
  - $d(C_1, C_2) = min(d(S_1, S_2), d(S_1, S_3)) = min(6, 4) = 4$
- $C_1 = S_4, C_2 = \{S_2, S_3\}$
  - $d(C_1, C_2) = min(d(S_4, S_2), d(S_4, S_3)) = min(3, 5) = 3$
- $C_1 = S_5, C_2 = \{S_2, S_3\}$
  - $d(C_1, C_2) = min(d(S_5, S_2), d(S_5, S_3)) = min(11, 13) = 11$
- $C_1 = S_6, C_2 = \{S_2, S_3\}$
  - $d(C_1, C_2) = min(d(S_6, S_2), d(S_6, S_3)) = min(18, 20) = 18$

Sample 1: Single Linkage

- While there is more than one cluster
  1. Merge the two closest clusters
  2. Update the proximity matrix

	S_1	$\{S_2, S_3\}$	S_4	S_5	S_6
S_1	0	4	9	17	24
$\{S_2, S_3\}$		0	3	11	18
S_4			0	8	15
S_5				0	7
S_6					0



Sample 1: Single Linkage

- While there is more than one cluster
  1. Merge the two closest clusters
  2. Update the proximity matrix

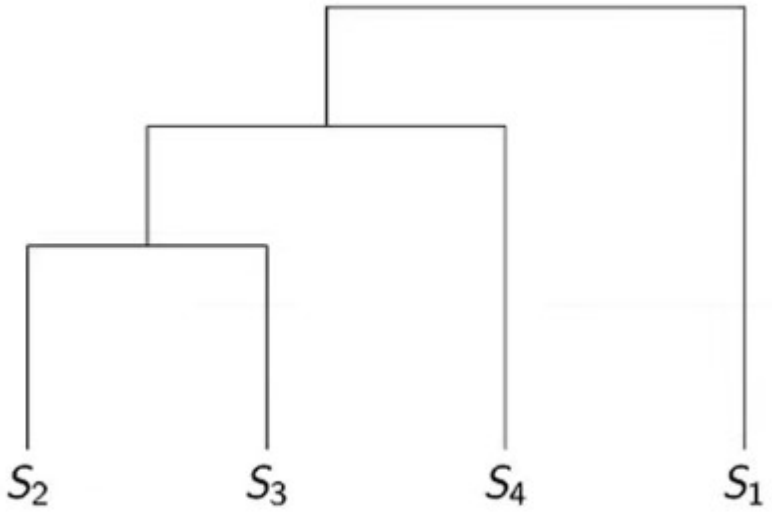
	$\{S_2, S_3, S_4\}$	S_1	S_5	S_6
$\{S_2, S_3, S_4\}$	0	4	8	15
S_1		0	17	24
S_5			0	7
S_6				0

- $C_1 = S_1, C_2 = \{S_2, S_3, S_4\}$
  - $d(C_1, C_2) = min(d(S_1, S_2), d(S_1, S_3), d(S_1, S_4)) = min(6, 4, 9) = 4$
- $C_1 = S_5, C_2 = \{S_2, S_3, S_4\}$
  - $d(C_1, C_2) = min(d(S_5, S_2), d(S_5, S_3), d(S_5, S_4)) = min(11, 13, 8) = 8$
- $C_1 = S_6, C_2 = \{S_2, S_3, S_4\}$
  - $d(C_1, C_2) = min(d(S_6, S_1), d(S_6, S_2), d(S_6, S_3)) = min(18, 20, 15) = 15$

Sample 1: Single Linkage

- While there is more than one cluster
  1. Merge the two closest clusters
  2. Update the proximity matrix

	$\{S_2, S_3, S_4\}$	S_1	S_5	S_6
$\{S_2, S_3, S_4\}$	0	4	8	15
S_1		0	17	24
S_5			0	7
S_6				0



Sample 1: Single Linkage

- While there is more than one cluster
  1. Merge the two closest clusters
  2. Update the proximity matrix

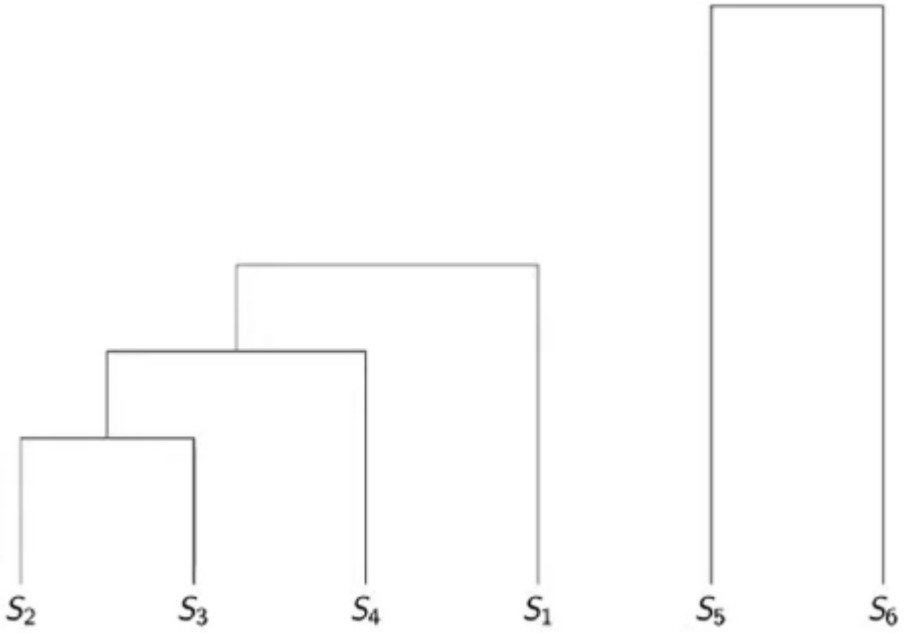
	$\{S_1, S_2, S_3, S_4\}$	S_5	S_6
$\{S_1, S_2, S_3, S_4\}$	0	8	15
S_5		0	7
S_6			0

- $C_1 = S_5, C_2 = \{S_1, S_2, S_3, S_4\}$
  - $d(C_1, C_2) = min(d(S_5, S_1), d(S_5, S_2), d(S_5, S_3), d(S_5, S_4)) = min(17, 11, 13, 8) = 8$
- $C_1 = S_6, C_2 = \{S_1, S_2, S_3, S_4\}$
  - $d(C_1, C_2) = min(d(S_6, S_1), d(S_6, S_2), d(S_6, S_3), d(S_6, S_4)) = min(24, 18, 20, 15) = 15$

Sample 1: Single Linkage

- While there is more than one cluster
  1. Merge the two closest clusters
  2. Update the proximity matrix

	$\{S_1, S_2, S_3, S_4\}$	S_5	S_6
$\{S_1, S_2, S_3, S_4\}$	0	8	15
S_5		0	7
S_6			0



Sample 1: Single Linkage

- While there is more than one cluster
  1. Merge the two closest clusters
  2. Update the proximity matrix

	$\{S_1, S_2, S_3, S_4\}$	$\{S_5, S_6\}$
$\{S_1, S_2, S_3, S_4\}$	0	8
$\{S_5, S_6\}$		0

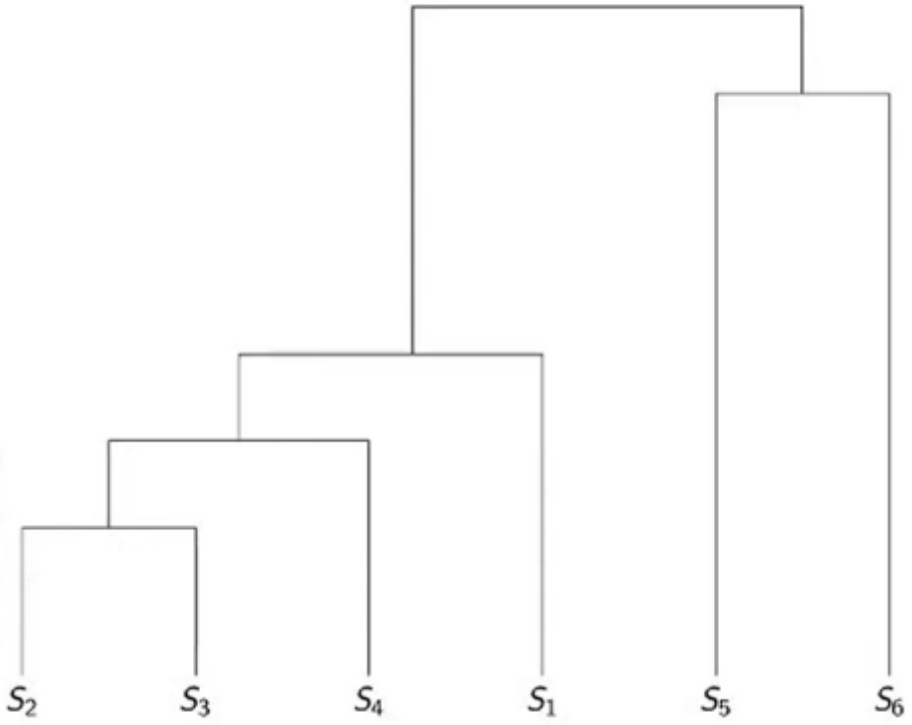
- $C_1 = \{S_5, S_6\}, C_2 = \{S_1, S_2, S_3, S_4\}$

$$\begin{aligned} d(C_1, C_2) =& min(d(S_5, S_1), d(S_5, S_2), d(S_5, S_3), d(S_5, S_4), \\ & d(S_6, S_1), d(S_6, S_2), d(S_6, S_3), d(S_6, S_4)) \\ =& min(24, 18, 20, 15, 17, 11, 13, 8) = 8. \end{aligned}$$

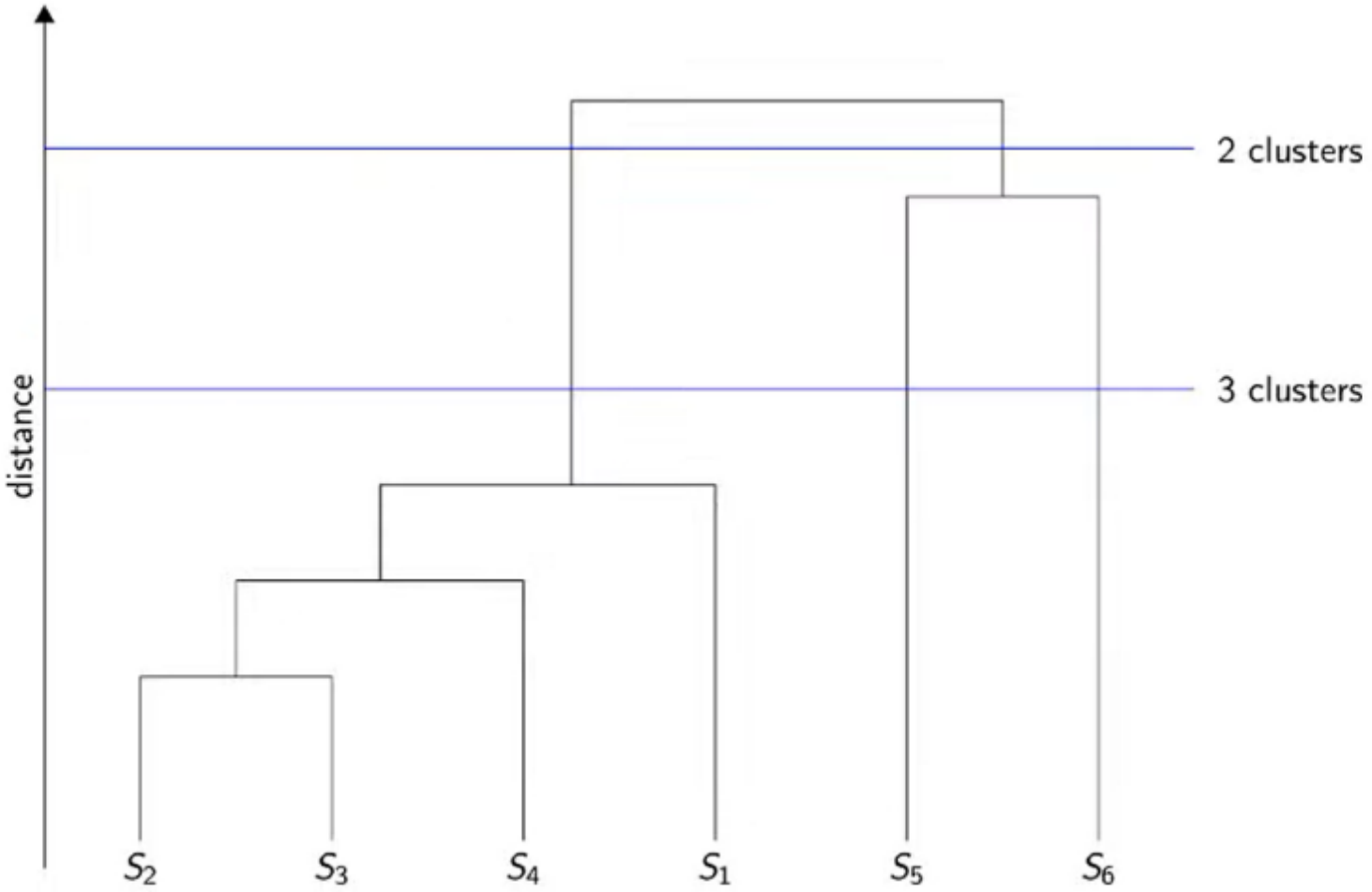
Sample 1: Single Linkage

- While there is more than one cluster
  1. Merge the two closest clusters
  2. Update the proximity matrix

	$\{S_1, S_2, S_3, S_4\}$	$\{S_5, S_6\}$
$\{S_1, S_2, S_3, S_4\}$	0	8
$\{S_5, S_6\}$		0



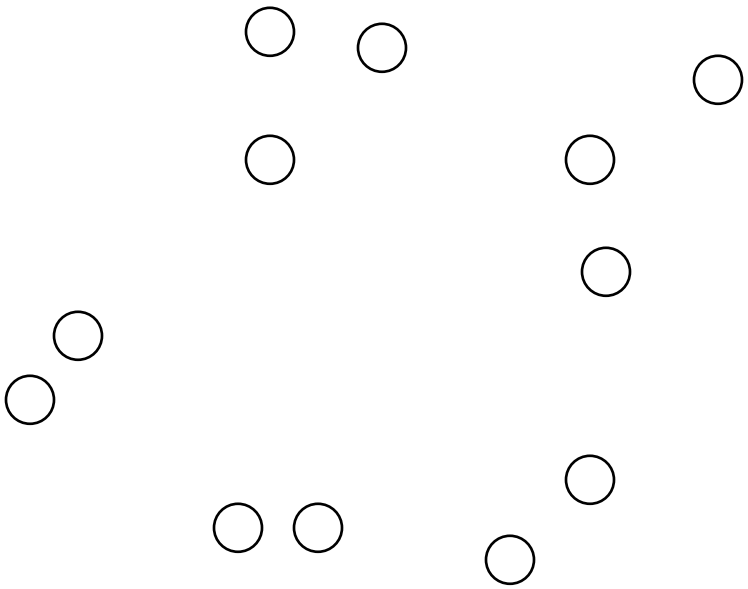
Sample 1: Single Linkage



Cut the dendrogram horizontally to determine the number of clusters.

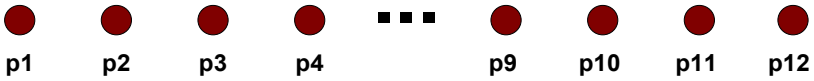
Starting Situation

- Start with clusters of individual points and a proximity matrix



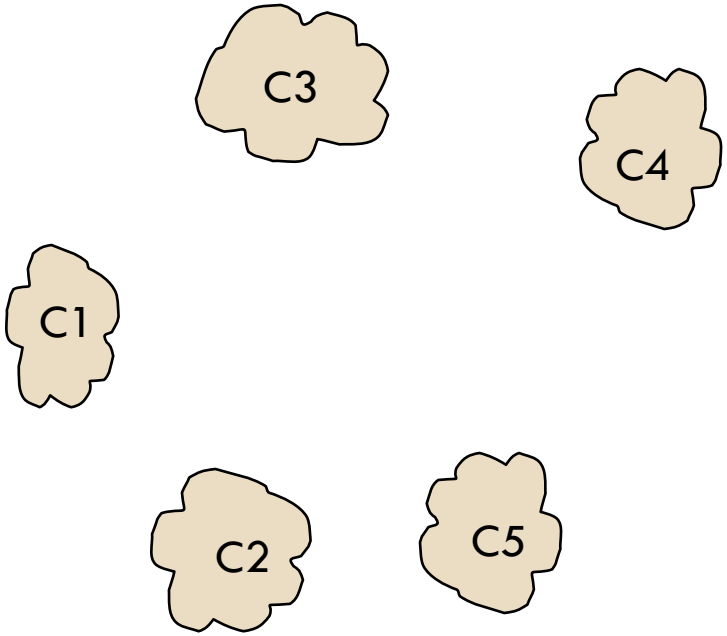
	p1	p2	p3	p4	p5	. . .
p1						
p2						
p3						
p4						
p5						
.						
.						
.						

Proximity Matrix



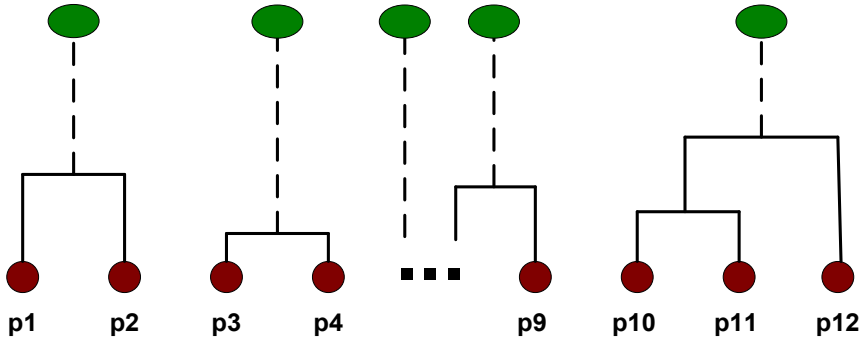
Intermediate Situation

- After some merging steps, we have some clusters



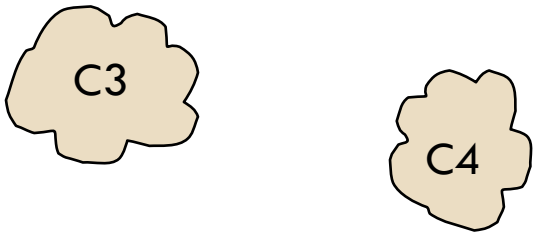
	C1	C2	C3	C4	C5
C1					
C2					
C3					
C4					
C5					

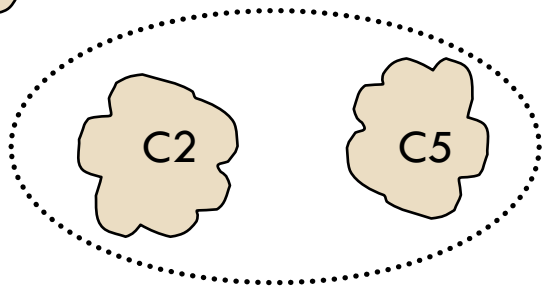
Proximity Matrix



Intermediate Situation

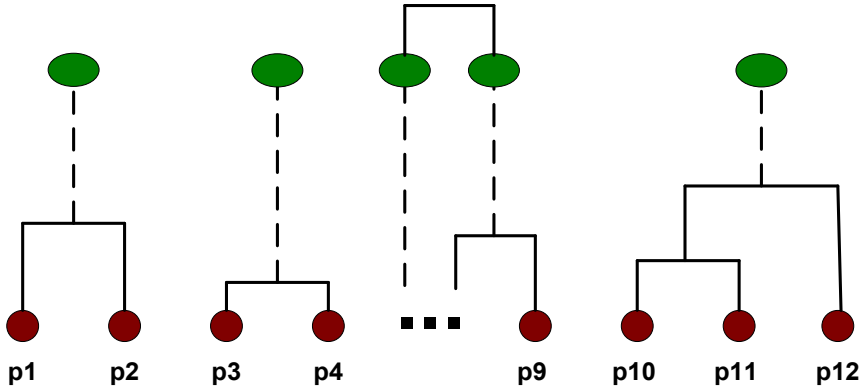
- We want to merge the two closest clusters (C2 and C5) and update the proximity matrix.





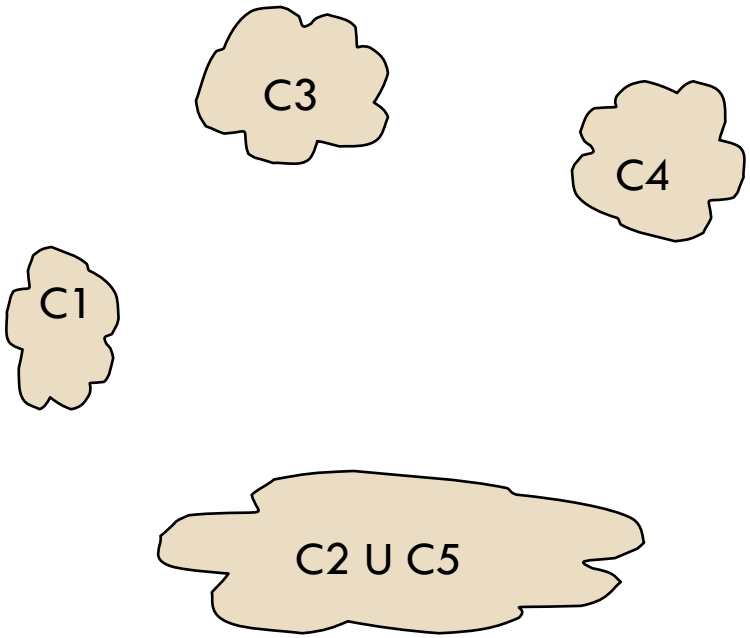
	C1	C2	C3	C4	C5
C1					
C2					
C3					
C4					
C5					

Proximity Matrix



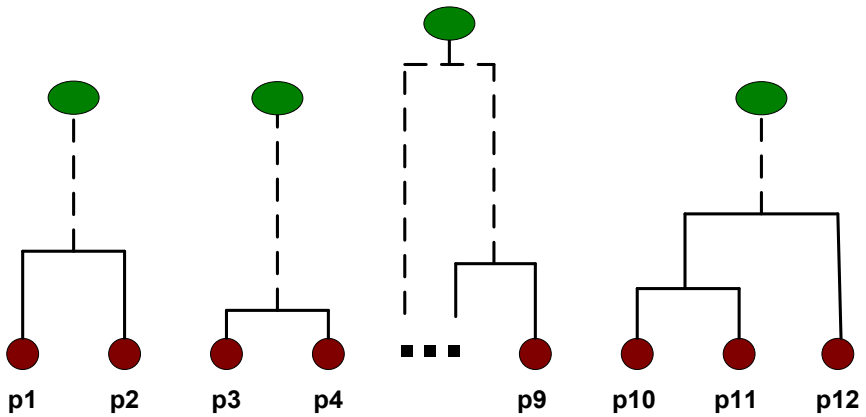
After Merging

- The question is "How do we update the proximity matrix?"

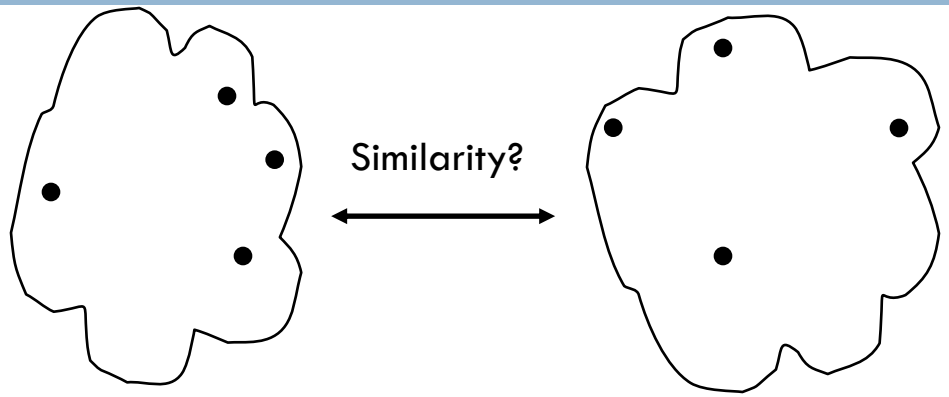


	C1	C2 U C5	C3	C4
C1		?		
C2 U C5	?	?	?	?
C3		?		
C4		?		

Proximity Matrix



How to Define Inter-Cluster Similarity

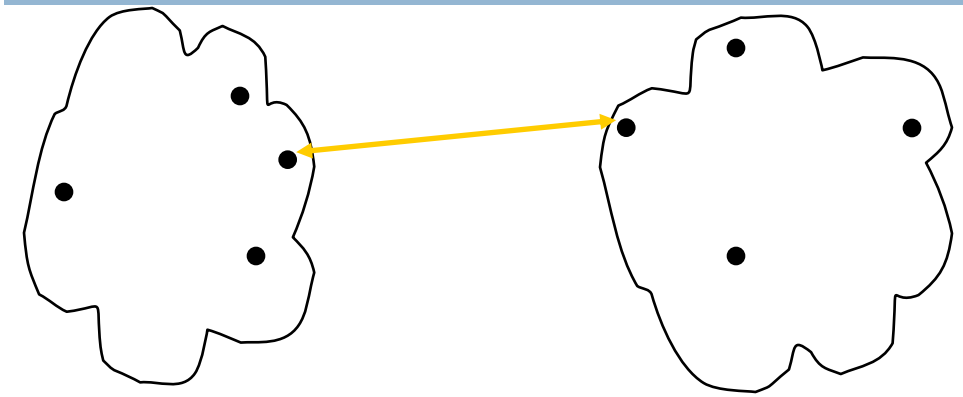


	p1	p2	p3	p4	p5	...
p1						
p2						
p3						
p4						
p5						
.						
.						
.						

Proximity Matrix

- MIN
- MAX
- Group Average
- Distance Between Centroids
- Other methods driven by an objective function
  - Ward's Method uses squared error

How to Define Inter-Cluster Similarity

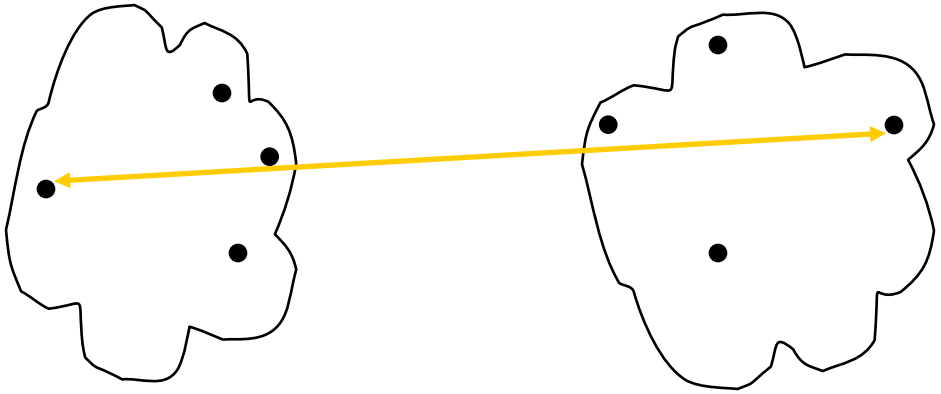


	p1	p2	p3	p4	p5	...
p1						
p2						
p3						
p4						
p5						
.						
.						
.						

Proximity Matrix

- MIN
- MAX
- Group Average
- Distance Between Centroids
- Other methods driven by an objective function
  - Ward's Method uses squared error

How to Define Inter-Cluster Similarity

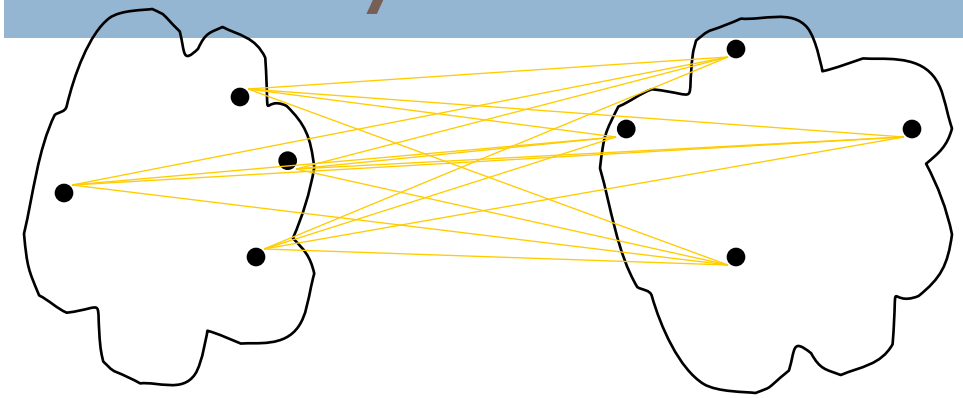


- MIN
- MAX
- Group Average
- Distance Between Centroids
- Other methods driven by an objective function
  - Ward's Method uses squared error

	p1	p2	p3	p4	p5	. . .
p1						
p2						
p3						
p4						
p5						
.						
.						
.						

Proximity Matrix

How to Define Inter-Cluster Similarity

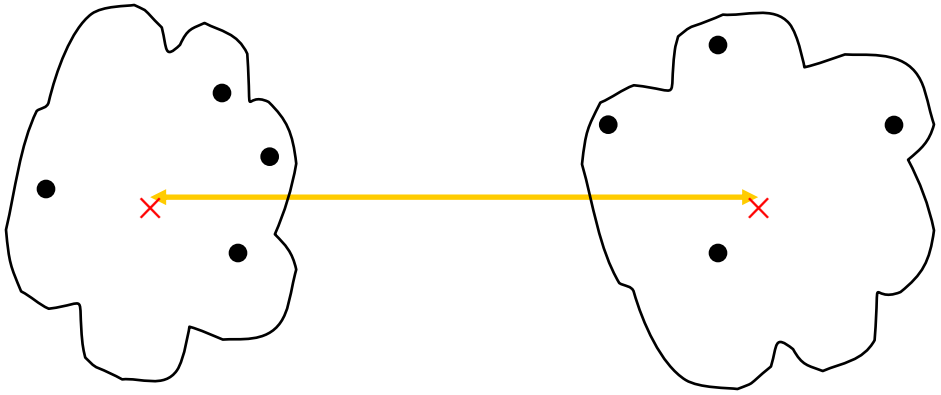


- MIN
- MAX
- Group Average
- Distance Between Centroids
- Other methods driven by an objective function
  - Ward's Method uses squared error

	p1	p2	p3	p4	p5	. . .
p1						
p2						
p3						
p4						
p5						
.						
.						
.						

Proximity Matrix

How to Define Inter-Cluster Similarity



	p1	p2	p3	p4	p5	...
p1						
p2						
p3						
p4						
p5						
.						
.						
.						

Proximity Matrix

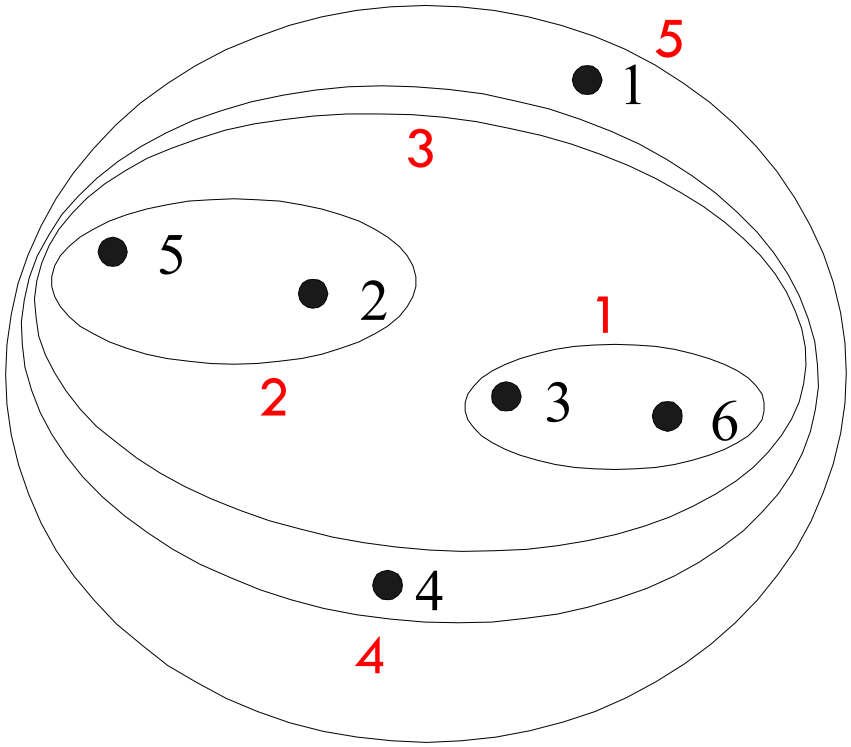
- MIN
- MAX
- Group Average
- Distance Between Centroids
- Other methods driven by an objective function
  - Ward's Method uses squared error

Single Link – Complete Link

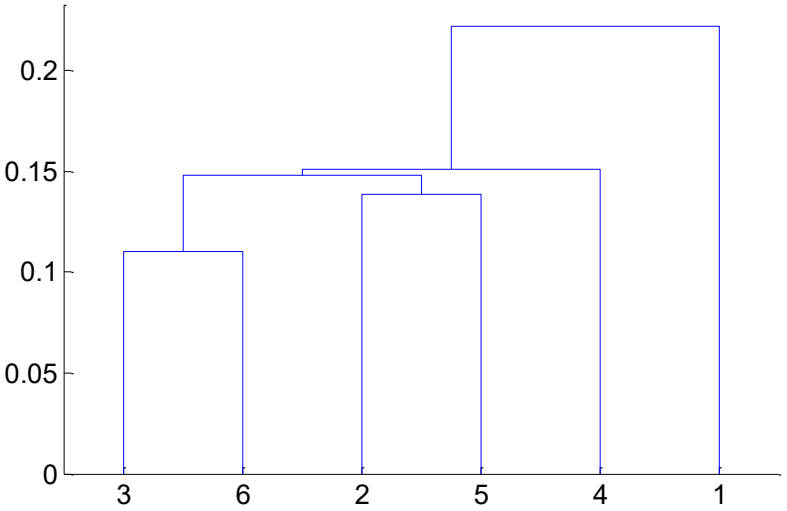
- Another way to view the processing of the hierarchical algorithm is that we create links between their elements in order of increasing distance
  - The MIN – Single Link, will merge two clusters when a single pair of elements is linked
  - The MAX – Complete Linkage will merge two clusters when all pairs of elements have been linked.

Hierarchical Clustering: MIN

	1	2	3	4	5	6
1	0	.24	.22	.37	.34	.23
2	.24	0	.15	.20	.14	.25
3	.22	.15	0	.15	.28	.11
4	.37	.20	.15	0	.29	.22
5	.34	.14	.28	.29	0	.39
6	.23	.25	.11	.22	.39	0

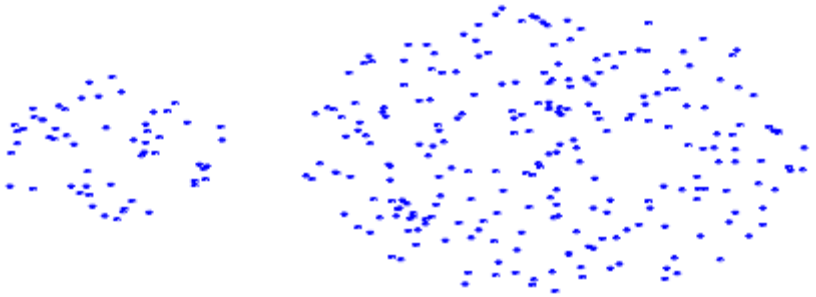


Nested Clusters

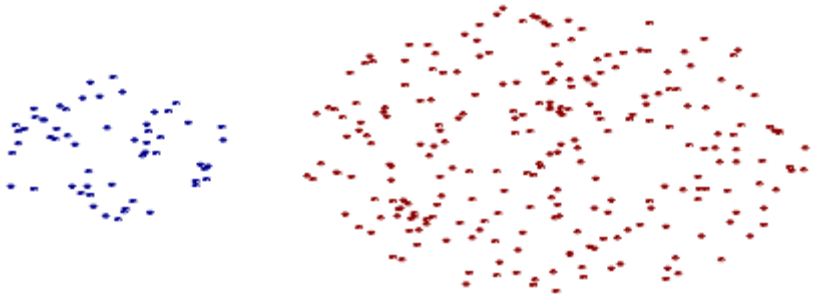


Dendrogram

Strength of MIN



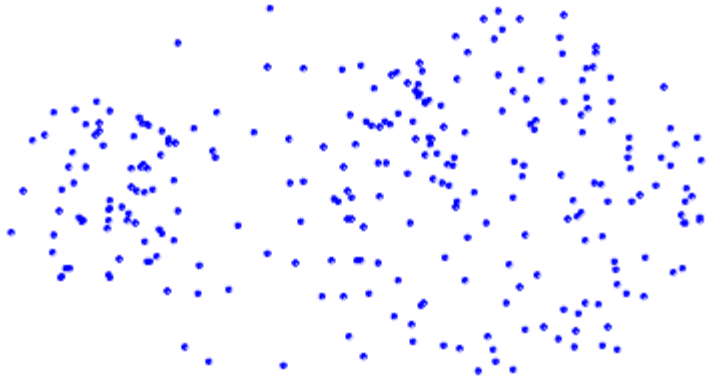
Original Points



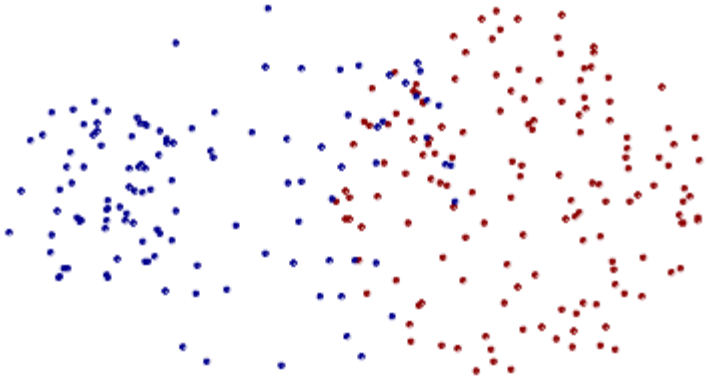
Two Clusters

- Can handle non-elliptical shapes

Limitations of MIN



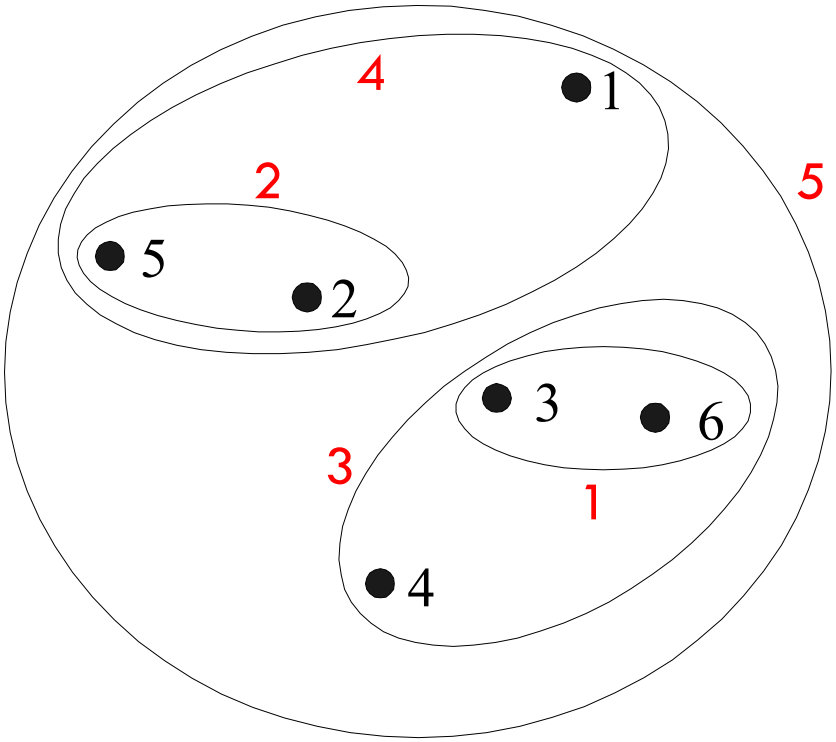
Original Points



Two Clusters

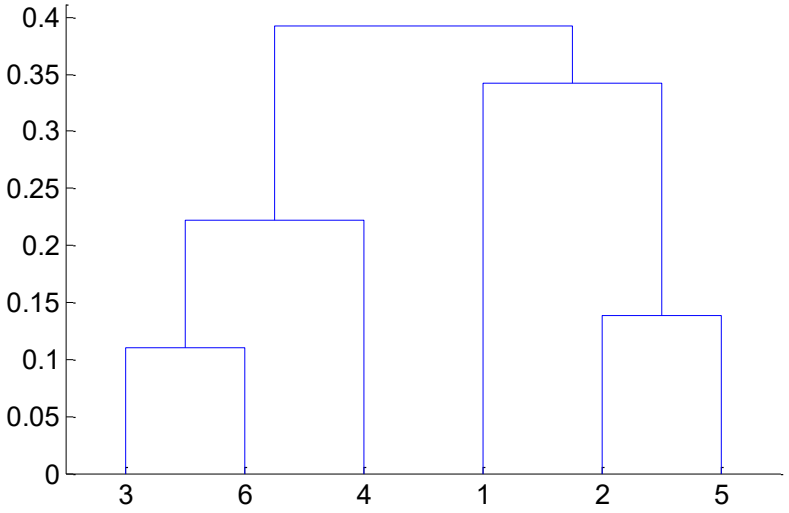
- Sensitive to noise and outliers

Hierarchical Clustering: MAX



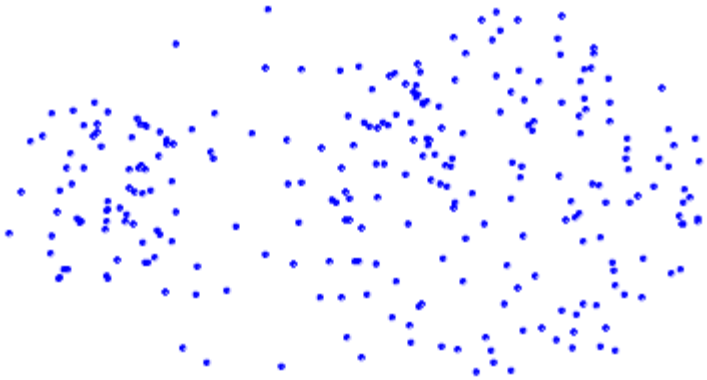
Nested Clusters

	1	2	3	4	5	6
1	0	.24	.22	.37	.34	.23
2	.24	0	.15	.20	.14	.25
3	.22	.15	0	.15	.28	.11
4	.37	.20	.15	0	.29	.22
5	.34	.14	.28	.29	0	.39
6	.23	.25	.11	.22	.39	0

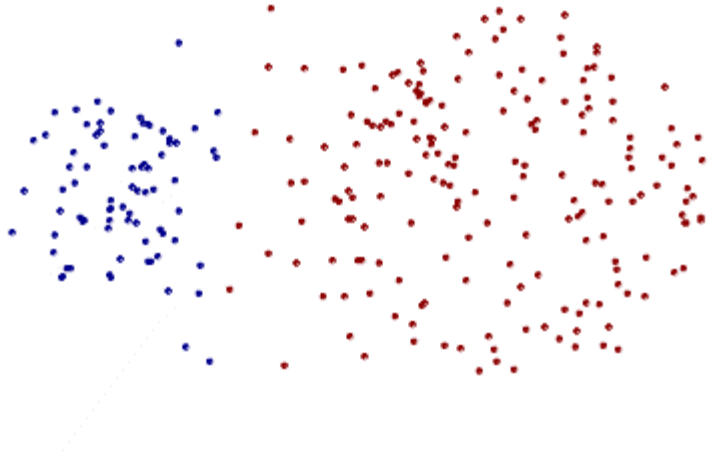


Dendrogram

Strength of MAX



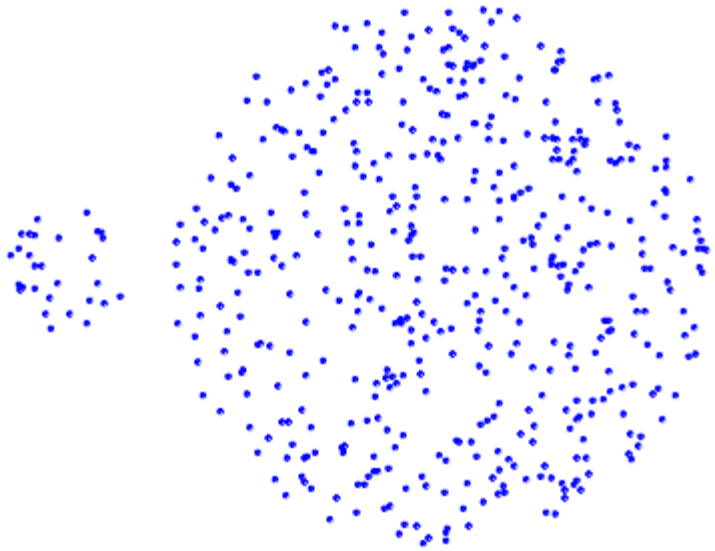
Original Points



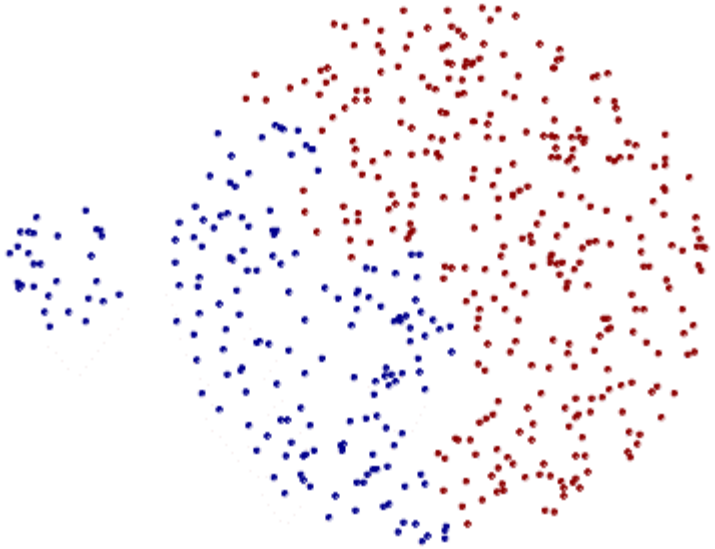
Two Clusters

- Less susceptible to noise and outliers

Limitations of MAX



Original Points



Two Clusters

- Tends to break large clusters
- Biased towards globular clusters

Cluster Similarity: Group Average

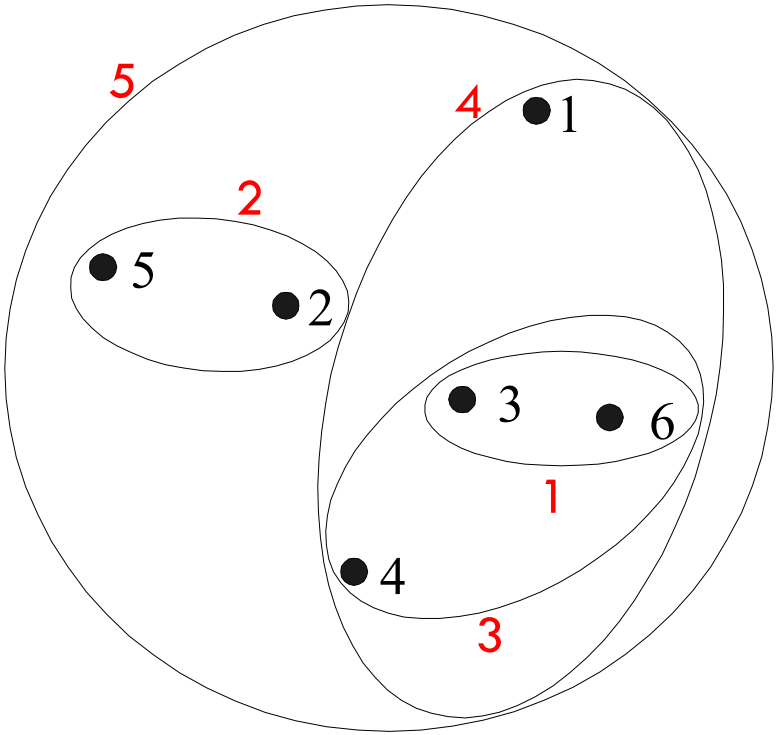
- Proximity of two clusters is the average of pairwise proximity between points in the two clusters.

$$\text{proximity}(\text{Cluster}_i, \text{Cluster}_j) = \frac{\sum_{\substack{p_i \in \text{Cluster}_i \\ p_j \in \text{Cluster}_j}} \text{proximity}(p_i, p_j)}{|\text{Cluster}_i| * |\text{Cluster}_j|}$$

- Need to use average connectivity for scalability since total proximity favors large clusters

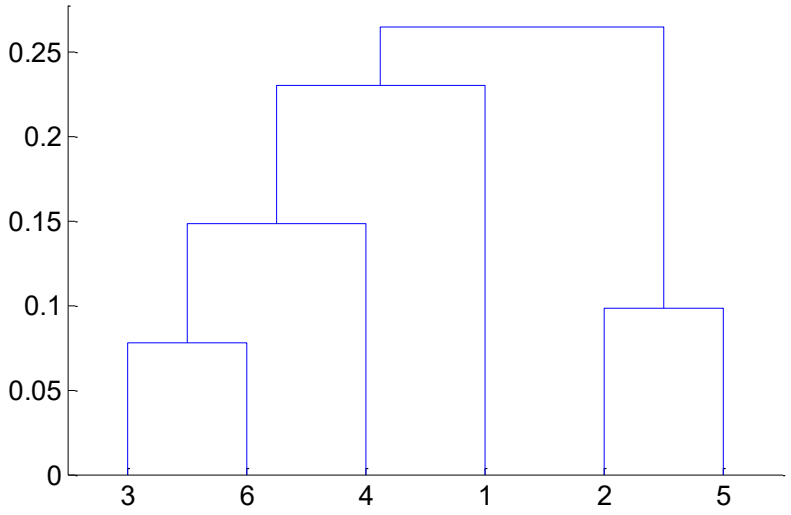
	1	2	3	4	5	6
1	0	.24	.22	.37	.34	.23
2	.24	0	.15	.20	.14	.25
3	.22	.15	0	.15	.28	.11
4	.37	.20	.15	0	.29	.22
5	.34	.14	.28	.29	0	.39
6	.23	.25	.11	.22	.39	0

Hierarchical Clustering: Group Average



Nested Clusters

	1	2	3	4	5	6
1	0	.24	.22	.37	.34	.23
2	.24	0	.15	.20	.14	.25
3	.22	.15	0	.15	.28	.11
4	.37	.20	.15	0	.29	.22
5	.34	.14	.28	.29	0	.39
6	.23	.25	.11	.22	.39	0



Dendrogram

Hierarchical Clustering: Group Average

- Compromise between Single and Complete Link

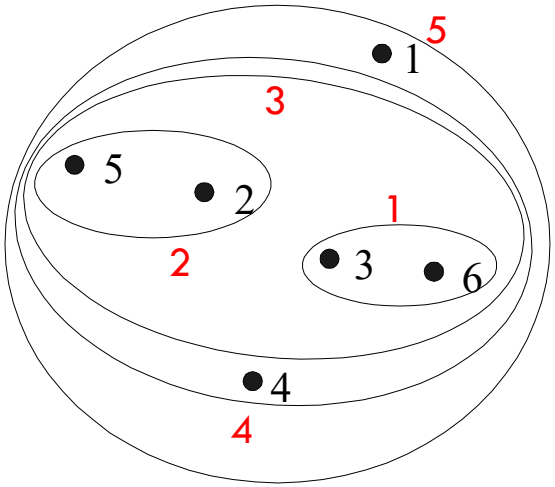
- Strengths
  - Less susceptible to noise and outliers

- Limitations
  - Biased towards globular clusters

Cluster Similarity: Ward's Method

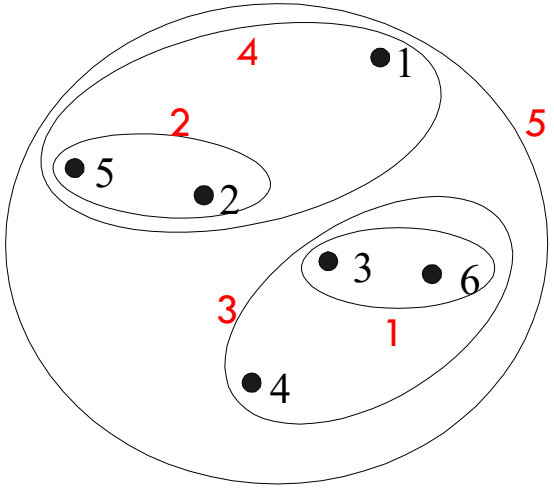
- Similarity of two clusters is based on the increase in squared error (SSE) when two clusters are merged
  - Similar to group average if distance between points is distance squared
- Less susceptible to noise and outliers
- Biased towards globular clusters
- Hierarchical analogue of K-means
  - Can be used to initialize K-means

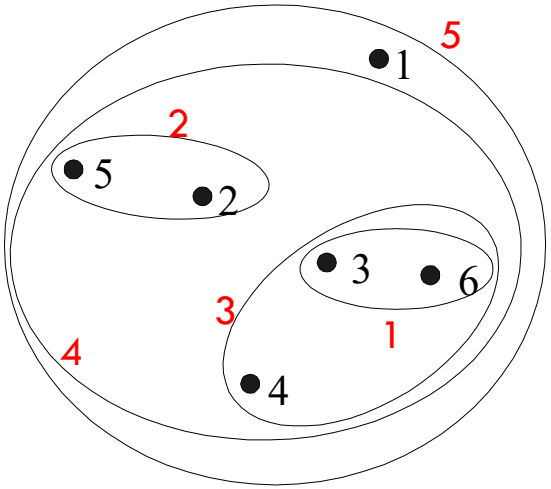
Hierarchical Clustering: Comparison



MIN

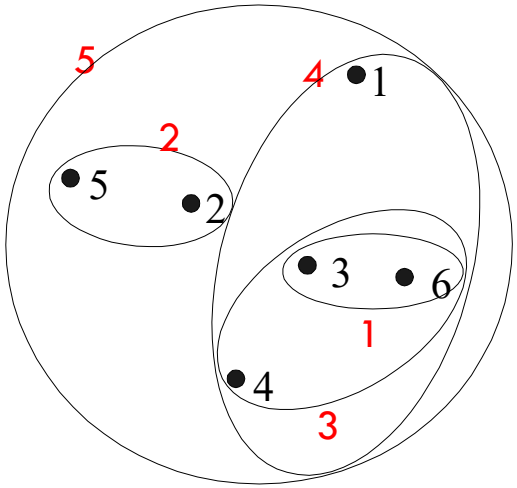
MAX





Group Average

Ward's Method



Hierarchical Clustering: Time and Space requirements

- $O(N^2)$ space since it uses the proximity matrix.
  - N is the number of points.

- $O(N^3)$ time in many cases
  - There are N steps and at each step the size, N^2, proximity matrix must be updated and searched
  - Complexity can be reduced to $O(N^2 \log(N))$ time for some approaches

Hierarchical Clustering: Problems and Limitations

- Computational complexity in time and space
- Once a decision is made to combine two clusters, it cannot be undone
- No objective function is directly minimized
- Different schemes have problems with one or more of the following:
  - Sensitivity to noise and outliers
  - Difficulty handling different sized clusters and convex shapes
  - Breaking large clusters

DBSCAN

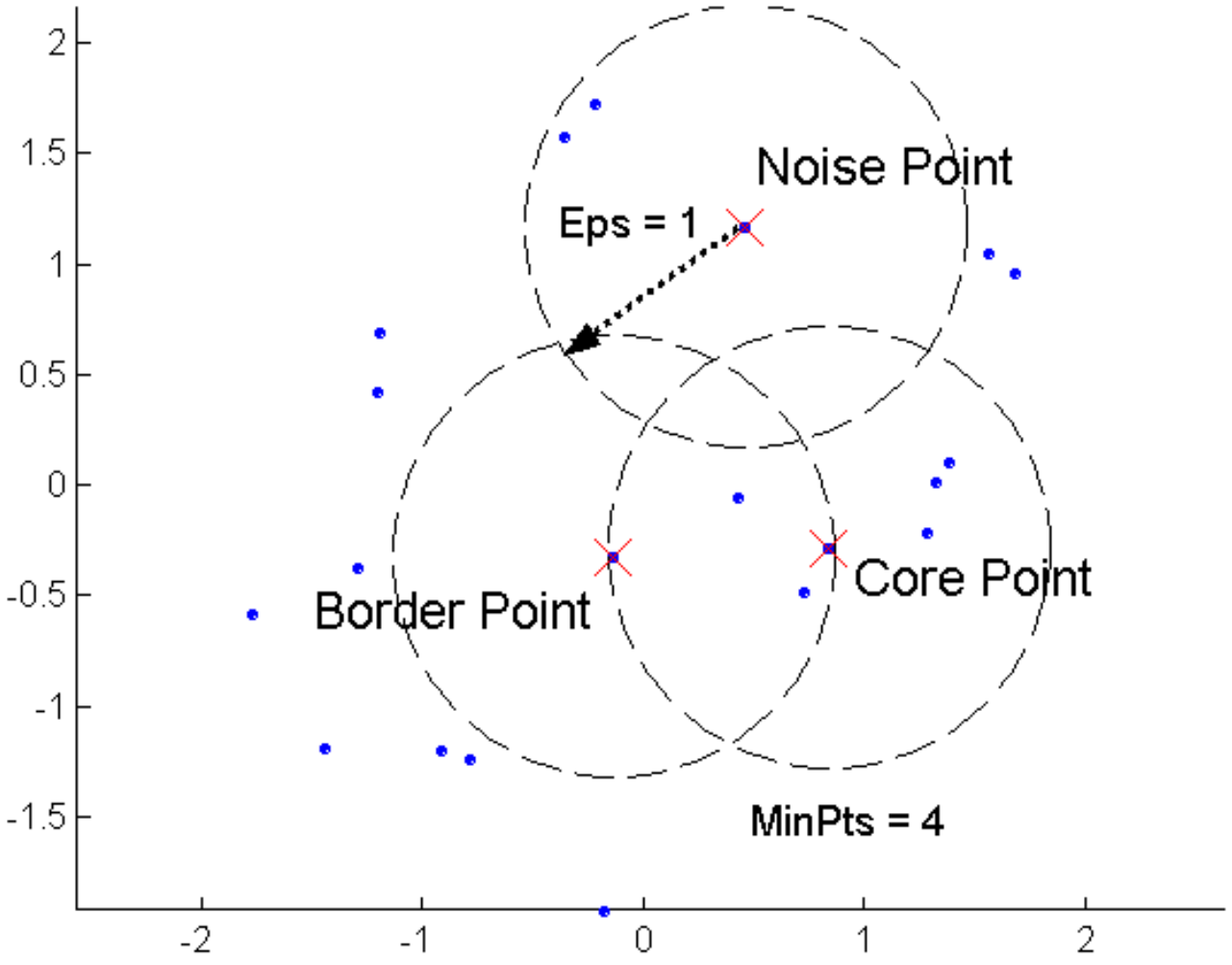
DBSCAN: Density-Based Clustering

- DBSCAN is a Density-Based Clustering algorithm
- Reminder: In density based clustering we partition points into dense regions separated by not-so-dense regions.
- Important Questions:
  - How do we measure density?
  - What is a dense region?
- DBSCAN:
  - Density at point p: number of points within a circle of radius Eps
  - Dense Region: A circle of radius Eps that contains at least MinPts points

DBSCAN

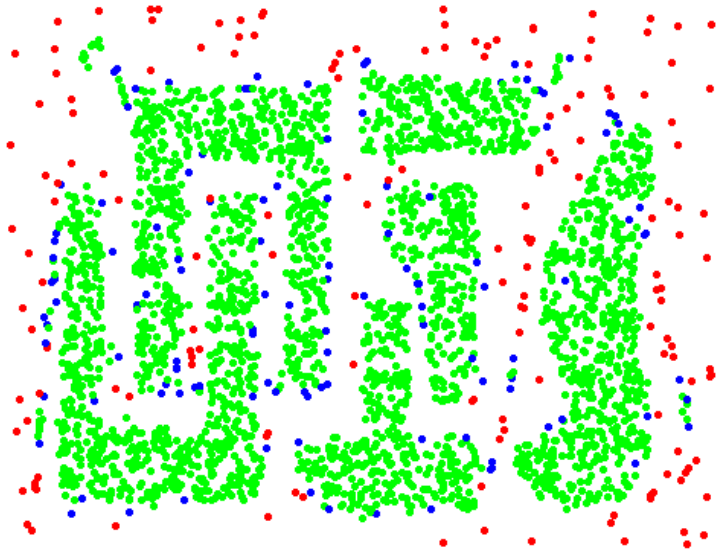
- Characterization of points
  - A point is a core point if it has more than a specified number of points (MinPts) within Eps
    - These points belong in a dense region and are at the interior of a cluster
  - A border point has fewer than MinPts within Eps, but is in the neighborhood of a core point.
  - A noise point is any point that is not a core point or a border point.

DBSCAN: Core, Border, and Noise Points



DBSCAN: Core, Border and Noise Points

Original Points

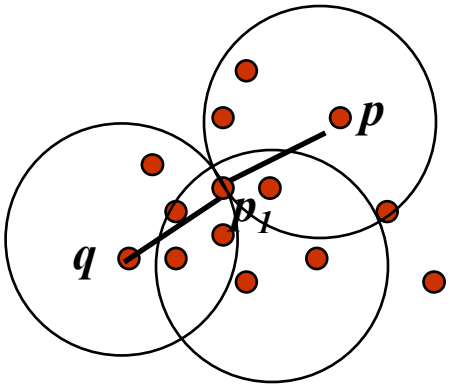


Point types: core, border and noise

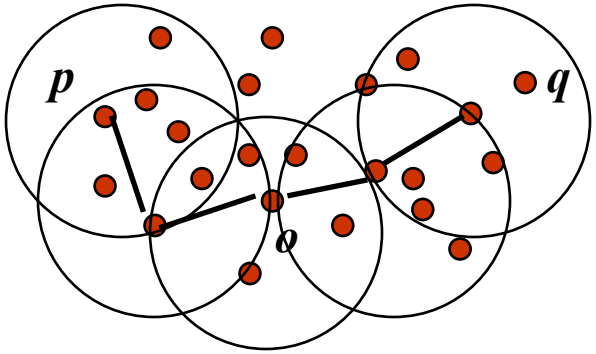
Eps = 10, MinPts = 4

Density-Connected points

- Density edge
  - We place an edge between two core points q and p if they are within distance Eps.



- Density-connected
  - A point p is density-connected to a point q if there is a path of edges from p to q

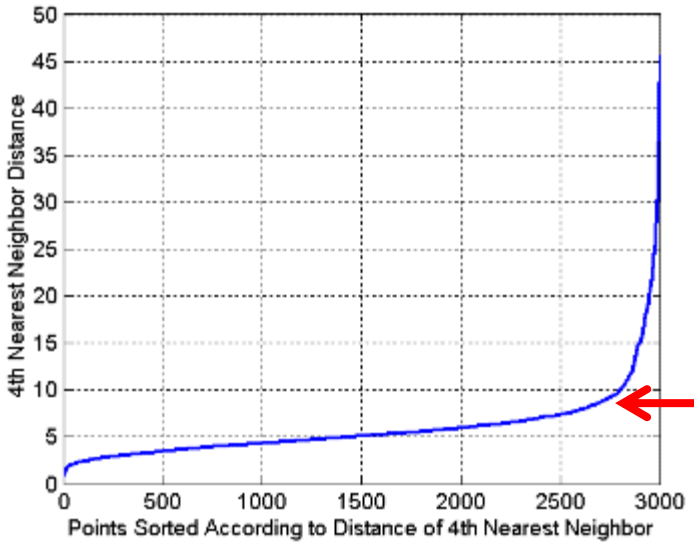


DBSCAN Algorithm

- Label points as core, border and noise
- Eliminate noise points
- For every core point p that has not been assigned to a cluster
  - Create a new cluster with the point p and all the points that are density-connected to p.
- Assign border points to the cluster of the closest core point.

DBSCAN: Determining Eps and MinPts

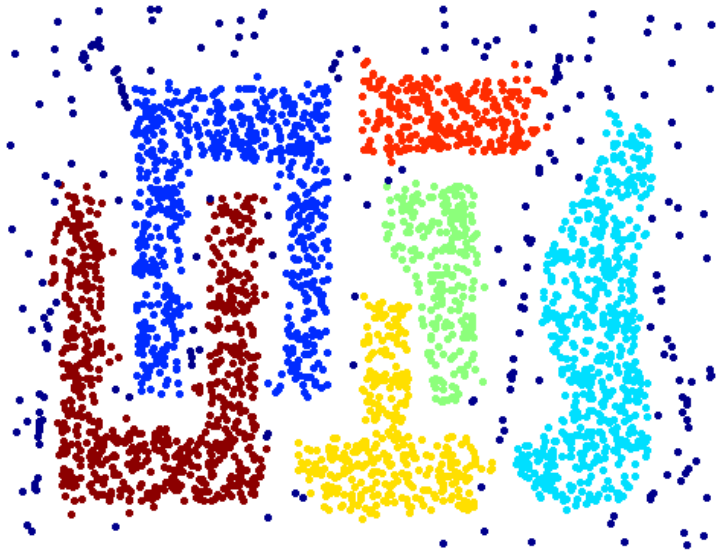
- Idea is that for points in a cluster, their k^{th} nearest neighbors are at roughly the same distance
- Noise points have the k^{th} nearest neighbor at farther distance
- So, plot sorted distance of every point to its k^{th} nearest neighbor
- Find the distance d where there is a "knee" in the curve
  - Eps = d, MinPts = k



Eps ~ 7-10
MinPts = 4

When DBSCAN Works Well

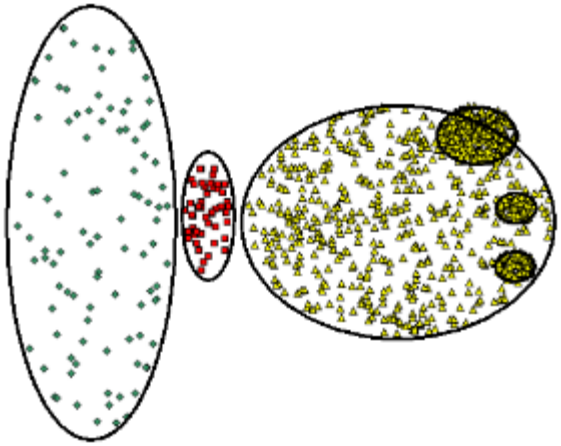
Original Points



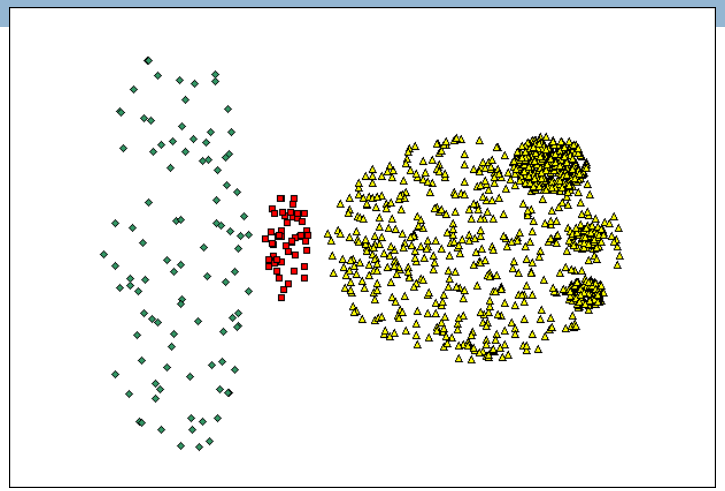
Clusters

- Resistant to Noise
- Can handle clusters of different shapes and sizes

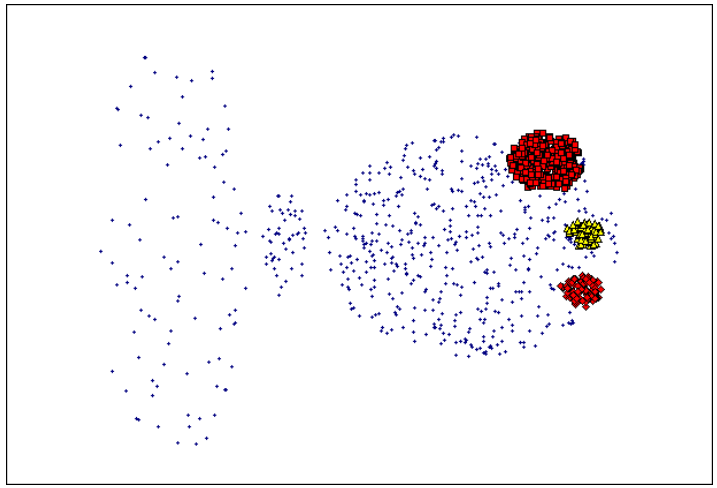
When DBSCAN Does NOT Work Well



Original Points



(MinPts=4, Eps=9.75).



(MinPts=4, Eps=9.92)

- Varying densities
- High-dimensional data

DBSCAN: Sensitive to Parameters

Figure 8. DBScan results for DS1 with MinPts at 4 and Eps at (a) 0.5 and (b) 0.4.

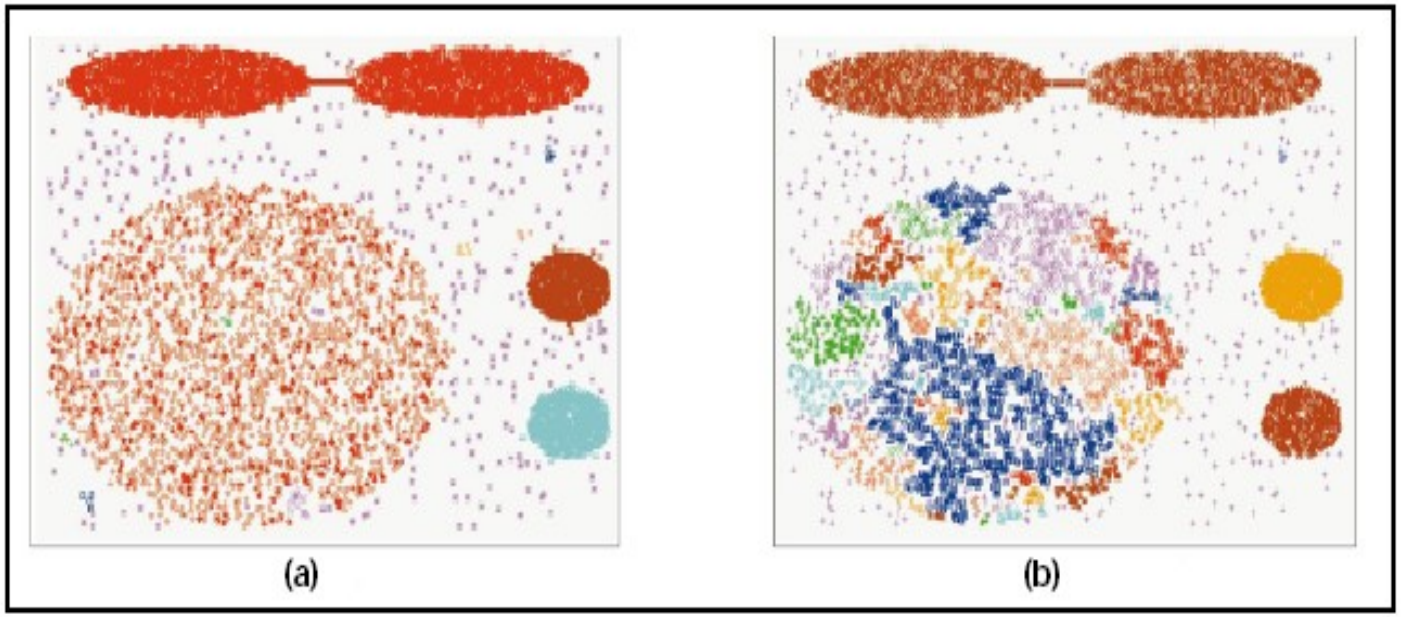
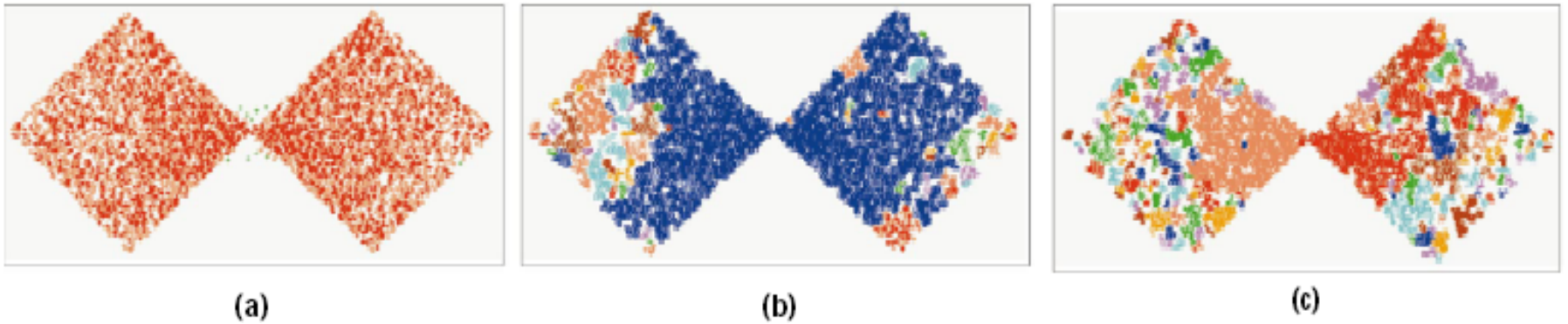


Figure 9. DBScan results for DS2 with MinPts at 4 and Eps at (a) 5.0, (b) 3.5, and (c) 3.0.



Other algorithms

- PAM, CLARANS: Solutions for the k-medoids problem
- BIRCH: Constructs a hierarchical tree that acts a summary of the data, and then clusters the leaves.
- MST: Clustering using the Minimum Spanning Tree.
- ROCK: clustering categorical data by neighbor and link analysis
- LIMBO, COOLCAT: Clustering categorical data using information theoretic tools.
- CURE: Hierarchical algorithm uses different representation of the cluster
- CHAMELEON: Hierarchical algorithm uses closeness and interconnectivity for merging

Mixture Models and the EM Algorithm

Model-based clustering

- In order to understand our data, we will assume that there is a generative process (a model) that creates/describes the data, and we will try to find the model that best fits the data.
  - Models of different complexity can be defined, but we will assume that our model is a distribution from which data points are sampled
  - Example: the data is the height of all people in Greece

- In most cases, a single distribution is not good enough to describe all data points: different parts of the data follow a different distribution
  - Example: the data is the height of all people in Greece and China
  - We need a mixture model
  - Different distributions correspond to different clusters in the data.

Gaussian Distribution

- Example: the data is the height of all people in Greece
  - Experience has shown that this data follows a Gaussian (Normal) distribution
  - Reminder: Normal distribution:

$$P(x) = \frac{1}{\sqrt{2\pi}\sigma} e^{-\frac{(x-\mu)^2}{2\sigma^2}}$$

  - μ = mean, σ = standard deviation

Gaussian Model

- What is a model?
  - A Gaussian distribution is fully defined by the mean μ and the standard deviation σ
  - We define our model as the pair of parameters $\theta = (\mu, \sigma)$

- This is a general principle: a model is defined as a vector of parameters θ

Fitting the model

- We want to find the normal distribution that best fits our data
  - Find the best values for μ and σ
  - But what does best fit mean?

Maximum Likelihood Estimation (MLE)

- Suppose that we have a vector $X = (x_1, \ldots, x_n)$ of values
- And we want to fit a Gaussian $N(\mu, \sigma)$ model to the data
- Probability of observing point x_i:

$$P(x_i) = \frac{1}{\sqrt{2\pi}\sigma} e^{-\frac{(x_i-\mu)^2}{2\sigma^2}}$$

- Probability of observing all points (assume independence)

$$P(X) = \prod_{i=1}^{n} P(x_i) = \prod_{i=1}^{n} \frac{1}{\sqrt{2\pi}\sigma} e^{-\frac{(x_i-\mu)^2}{2\sigma^2}}$$

- We want to find the parameters $\theta = (\mu, \sigma)$ that maximize the probability $P(X|\theta)$

Maximum Likelihood Estimation (MLE)

- The probability $P(X|\theta)$ as a function of θ is called the Likelihood function

$$L(\theta) = \prod_{i=1}^{n} \frac{1}{\sqrt{2\pi}\sigma} e^{-\frac{(x_i-\mu)^2}{2\sigma^2}}$$

- It is usually easier to work with the Log-Likelihood function

$$LL(\theta) = -\sum_{i=1}^{n} \frac{(x_i-\mu)^2}{2\sigma^2} - \frac{1}{2} n \log 2\pi - n \log \sigma$$

- Maximum Likelihood Estimation
  - Find parameters μ, σ that maximize $LL(\theta)$

$$\mu = \frac{1}{n}\sum_{i=1}^{n} x_i = \mu_X$$

Sample Mean

$$\sigma^2 = \frac{1}{n}\sum_{i=1}^{n} (x_i-\mu)^2 = \sigma_X^2$$

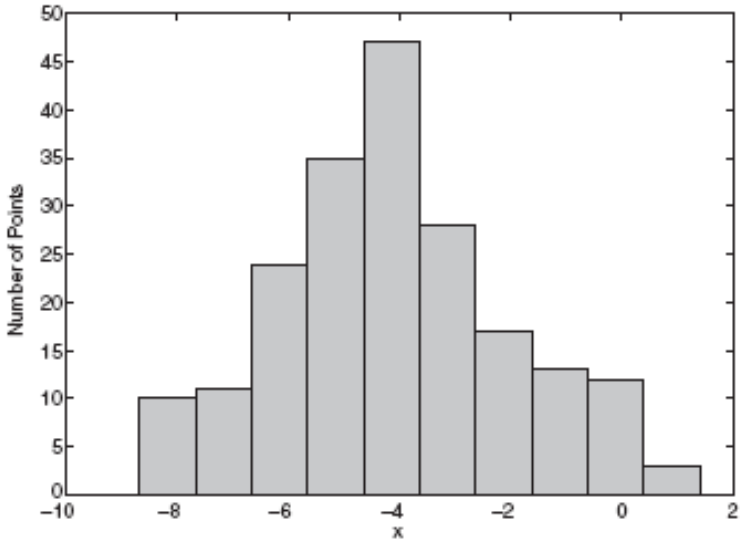
Sample Variance

MLE

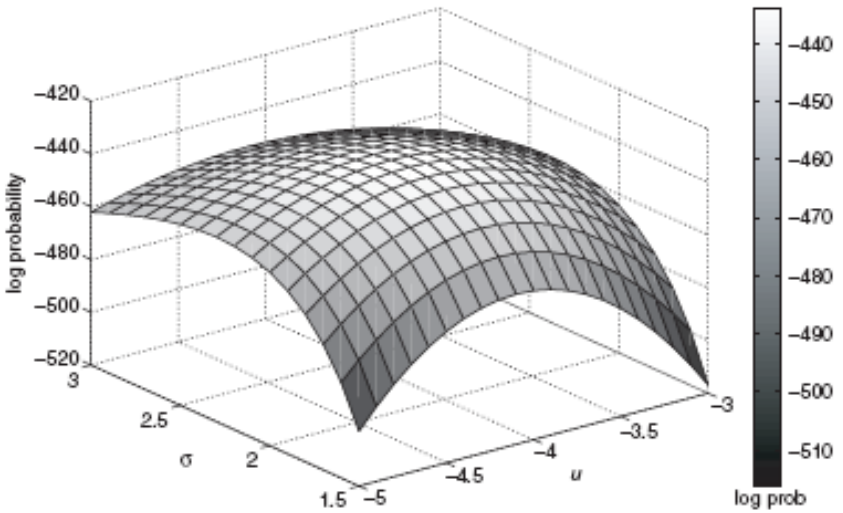
- Note: these are also the most likely parameters given the data

$$P(\theta|X) = \frac{P(X|\theta)P(\theta)}{P(X)}$$

- If we have no prior information about θ, or X, then maximizing $P(X|\theta)$ is the same as maximizing $P(\theta|X)$



(a) Histogram of 200 points from a Gaussian distribution.

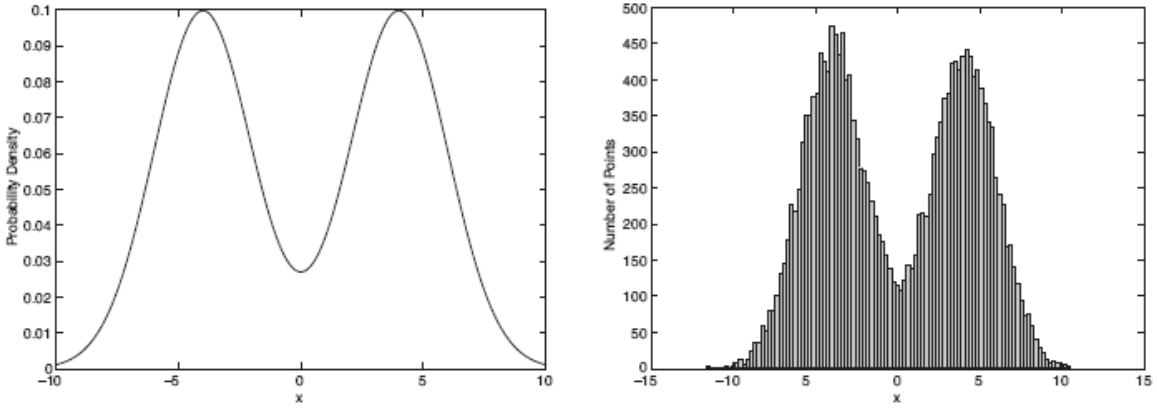


(b) Log likelihood plot of the 200 points for different values of the mean and standard deviation.

Figure 9.3. 200 points from a Gaussian distribution and their log probability for different parameter values.

Mixture of Gaussians

- Suppose that you have the heights of people from Greece and China and the distribution looks like the figure below (dramatization)



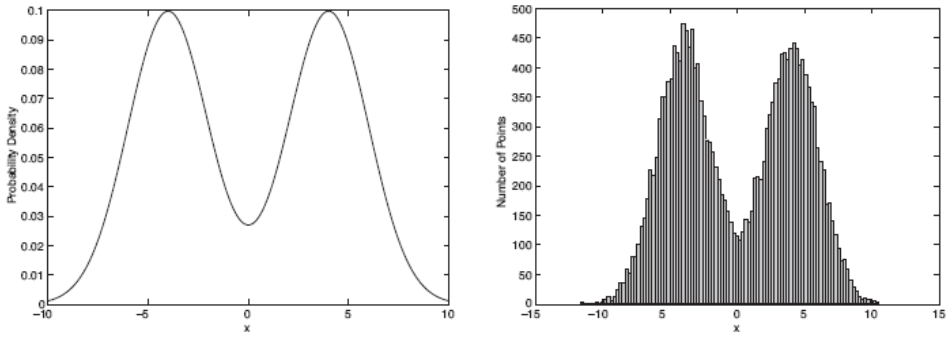
(a) Probability density function for the mixture model.

(b) 20,000 points generated from the mixture model.

Figure 9.2. Mixture model consisting of two normal distributions with means of -4 and 4, respectively. Both distributions have a standard deviation of 2.

Mixture of Gaussians

- In this case the data is the result of the mixture of two Gaussians
  - One for Greek people, and one for Chinese people
  - Identifying for each value which Gaussian is most likely to have generated it will give us a clustering.



(a) Probability density function for the mixture model.

(b) 20,000 points generated from the mixture model.

Figure 9.2. Mixture model consisting of two normal distributions with means of -4 and 4, respectively. Both distributions have a standard deviation of 2.

Mixture model

- A value x_i is generated according to the following process:
  - First select the nationality
    - With probability π_G select Greek, with probability π_C select China ($\pi_G + \pi_C =$

We can also thing of this as a Hidden Variable Z

  - Given the nationality, generate the point from the corresponding Gaussian
    - $P(x_i|\theta_G) \sim N(\mu_G, \sigma_G)$ if Greece
    - $P(x_i|\theta_G) \sim N(\mu_G, \sigma_G)$ if China

Mixture Model

- Our model has the following parameters

$$\Theta = (\pi_G, \pi_C, \mu_G, \mu_C, \sigma_G, \sigma_C)$$

Mixture probabilities | Distribution Parameters

- For value x_i, we have:

$$P(x_i|\Theta) = \pi_G P(x_i|\theta_G) + \pi_C P(x_i|\theta_C)$$

- For all values $X = (x_1, \ldots, x_n)$

$$P(X|\Theta) = \prod_{i=1}^{n} P(x_i|\Theta)$$

- We want to estimate the parameters that maximize the Likelihood of the data

Mixture Models

- Once we have the parameters $\Theta = (\pi_G, \pi_C, \mu_G, \mu_C, \sigma_G, \sigma_C)$ we can estimate the membership probabilities $P(G|x_i)$ and $P(C|x_i)$ for each point x_i:
  - This is the probability that point x_i belongs to the Greek or the Chinese population (cluster)

$$P(G|x_i) = \frac{P(x_i|G)P(G)}{P(x_i|G)P(G) + P(x_i|C)P(C)}$$
$$= \frac{P(x_i|G)\pi_G}{P(x_i|G)\pi_G + P(x_i|C)\pi_C}$$

EM (Expectation Maximization) Algorithm

- Initialize the values of the parameters in Θ to some random values
- Repeat until convergence
  - E-Step: Given the parameters Θ estimate the membership probabilities $P(G|x_i)$ and $P(C|x_i)$
  - M-Step: Compute the parameter values that (in expectation) maximize the data likelihood

$$\pi_G = \frac{1}{n}\sum_{i=1}^{n} P(G|x_i) \qquad \pi_C = \frac{1}{n}\sum_{i=1}^{n} P(C|x_i)$$

Fraction of population in G,C

$$\mu_C = \sum_{i=1}^{n} \frac{P(C|x_i)}{n * \pi_C} x_i \qquad \mu_G = \sum_{i=1}^{n} \frac{P(G|x_i)}{n * \pi_G} x_i$$

MLE Estimates if π's were fixed

$$\sigma_C^2 = \sum_{i=1}^{n} \frac{P(C|x_i)}{n * \pi_C} (x_i - \mu_C)^2 \qquad \sigma_G^2 = \sum_{i=1}^{n} \frac{P(G|x_i)}{n * \pi_G} (x_i - \mu_G)^2$$

K-Means CLUSTERING

K-means

Most well-known and popular clustering algorithm:

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

Most well-known and popular clustering algorithm that cluster *n* objects based on attributes into *k* partitions, where $k < n$.

- It assumes that the object attributes form a vector space.
- An algorithm for partitioning (or clustering) N data points into K disjoint subsets S_i containing data points so as to minimize the sum-of-squares criterion

$$J = \sum_{j=1}^{K} \sum_{n \in S_j} |x_n - \mu_j|^2,$$

where x_n is a vector representing the the n^{th} data point and u_i is the geometric centroid of the data points in S_i.

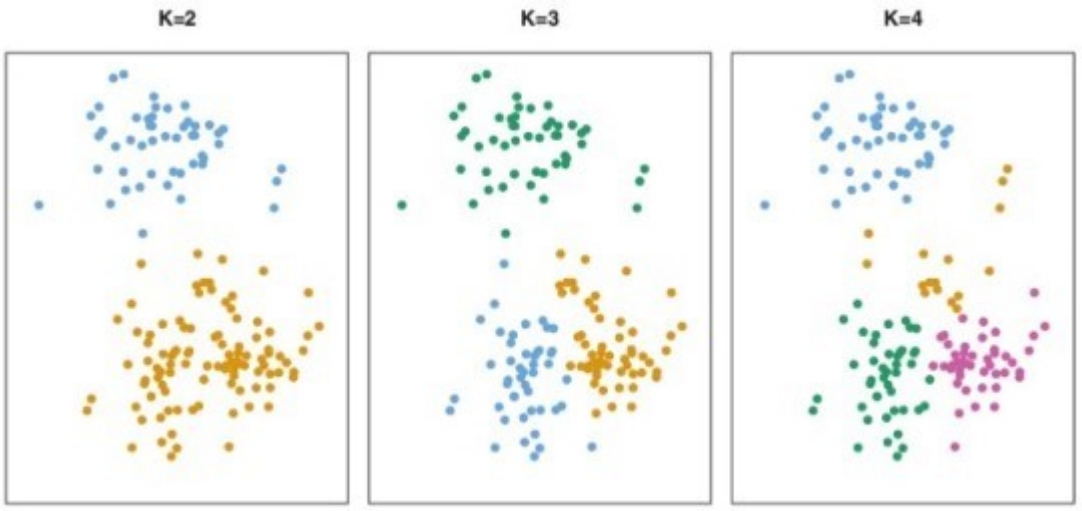
Start with some initial cluster centers. Iterate:

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

K-means

- Simply speaking k-means clustering is an algorithm to group the objects based on attributes/features into K number of group.
- K is positive integer number.
- The grouping is done by minimizing the sum of squares of distances between data and the corresponding cluster centroid.

How to know the pattern similarity with K-means clustering method?



source : Intoduction to Statistical Learning with Applications in R page 387

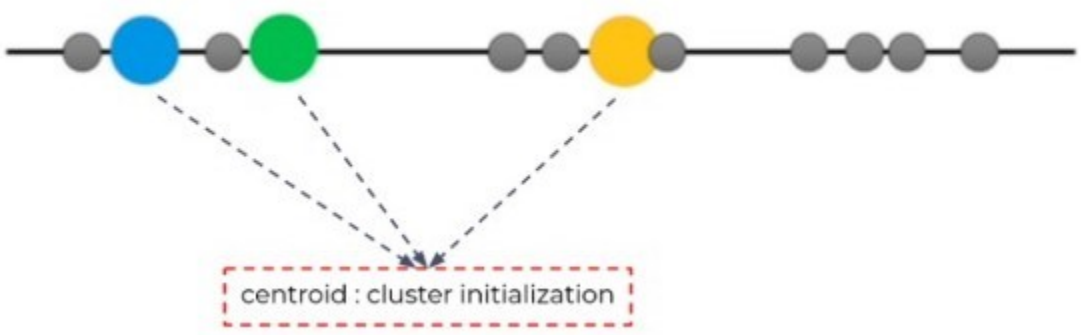
- K-means clustering is a simple and elegant approach for partitioning a data set into K distinct, non-overlapping clusters.
- To perform K-means clustering, we must first specify the desired number of clusters K; then the K-means algorithm will assign each observation to exactly one of the K clusters

How K-means works

Suppose, data is provided as above.

Step 1.
Determine the value "K", the value "K" represents the number of clusters.
in this case, we'll select K=3. That is to say, we want to identify 3 clusters.

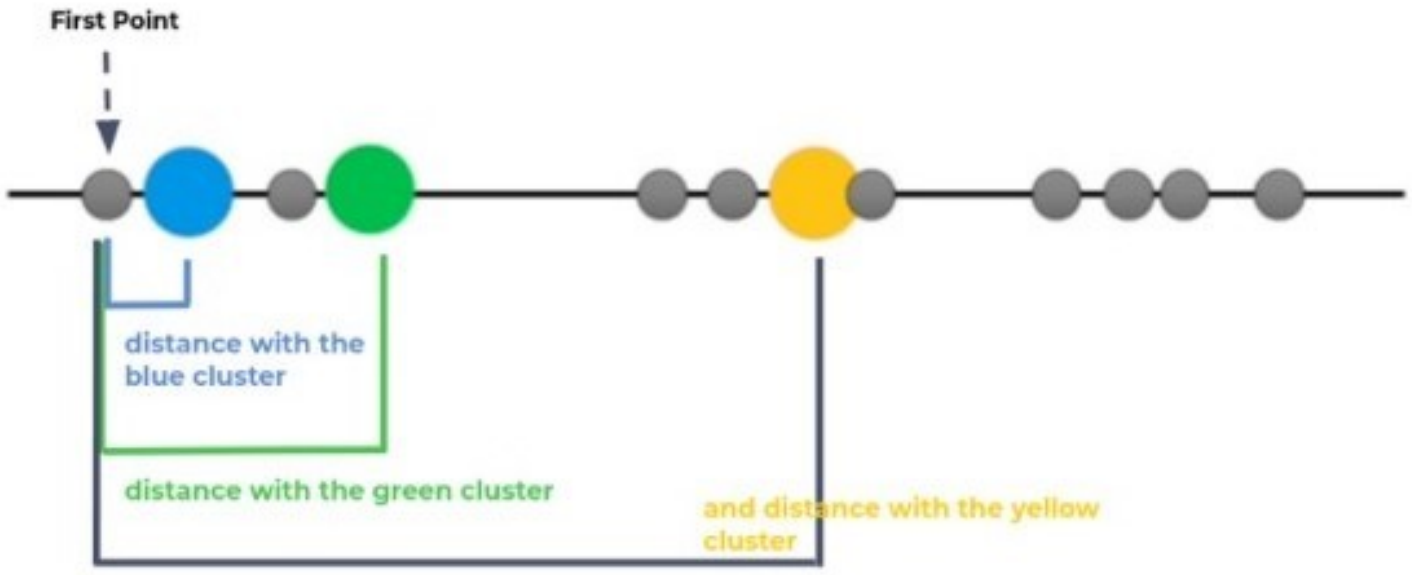
Step 2.
Randomly select 3 distinct centroid (new data points as cluster initialization)
for example — attempts 1. "K" is equal 3 so there are 3 centroid, in which case it will be the cluster initialization



How K-means works

Step 3. Measure the distance (euclidean distance) between each point and the centroid

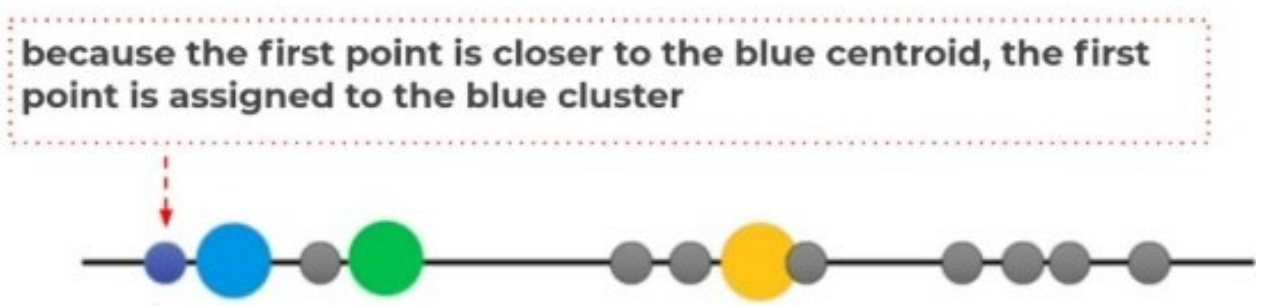
for example, measure the distance between first point and the centroid.



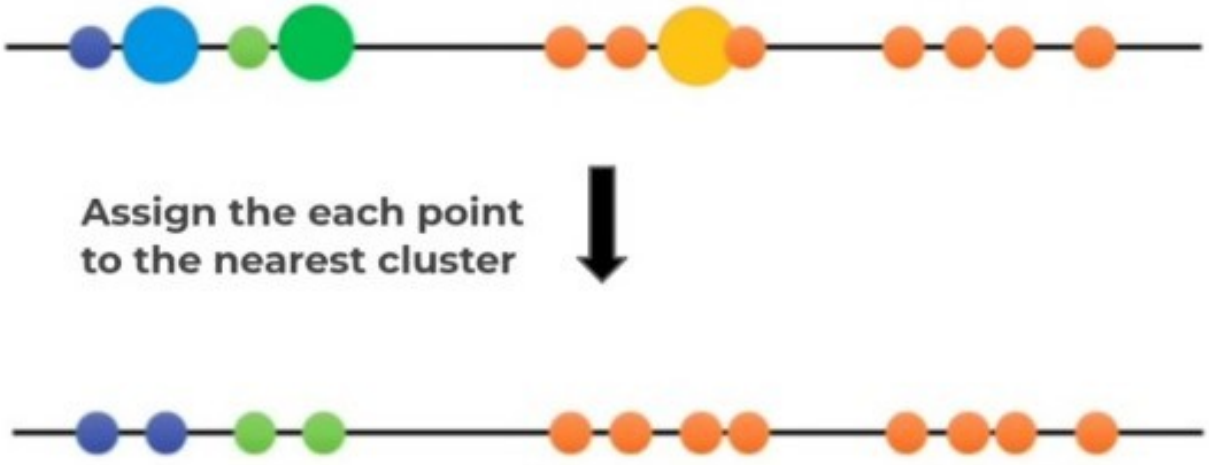
How K-means works

Step 4. Assign each point to the nearest cluster.

for example, measure the distance between first point and the centroid.

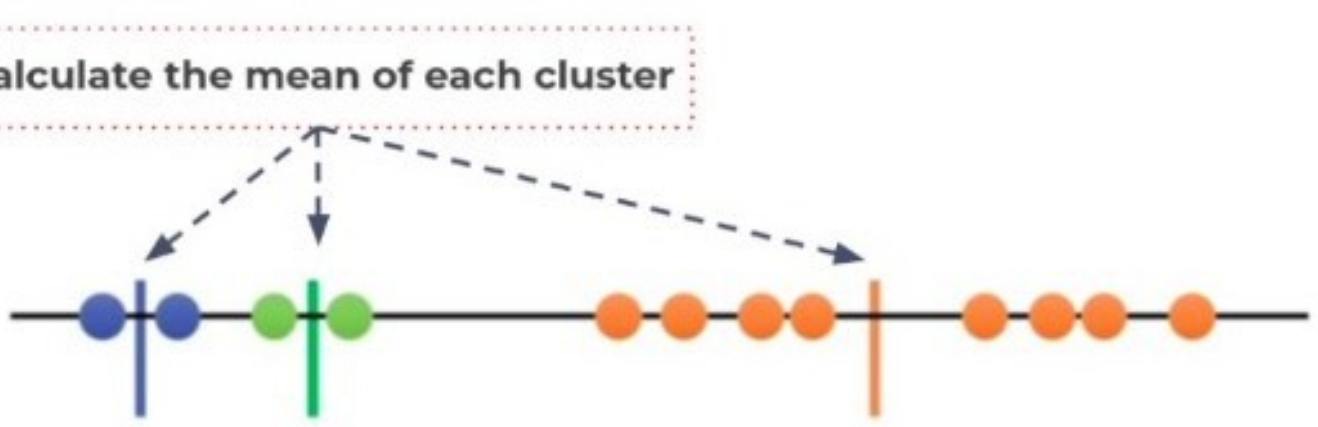


Do the same treatment for the other unlabeled point, until we get this



How K-means works

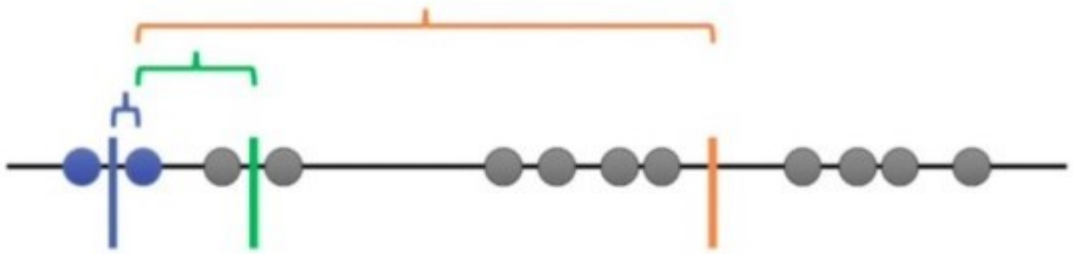
Step 5. Calculate the mean of each cluster as new centroid



Update the centroid with mean of each cluster

How K-means works

Step 6. Repeat step 3–5 with the new center of cluster



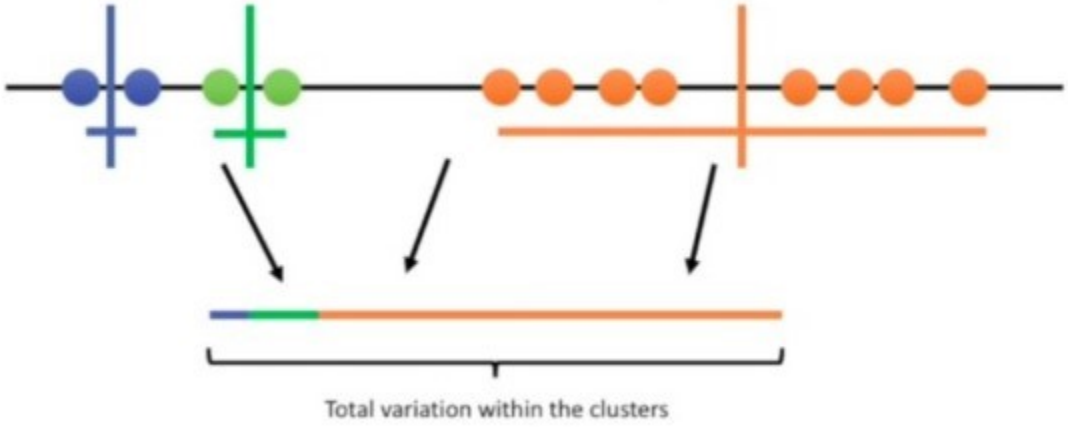
Repeat until stop:

- Convergence. (No further changes)
- Maximum number of iterations.

Since the clustering did not change at all during the last iteration, we're done.

How K-means works

Step 7. Calculate the variance of each cluster



Since K-means clustering can't "see" the best clustering, it is only option is to keep track of these clusters, and their total variance, and do the whole thing over again with different starting points.

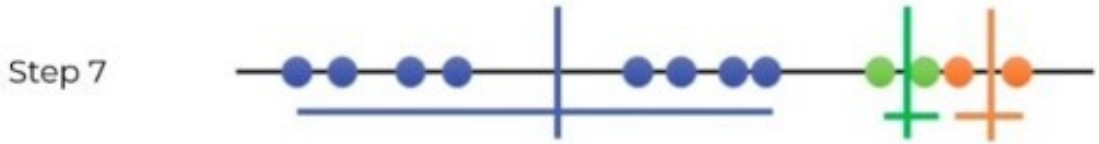
How K-means works

Step 8. Repeat step 2–7 until get the lowest sum of variance

For example — attempts 2 with different random centroid

How K-means works

For example — attempts 3 with different random centroid

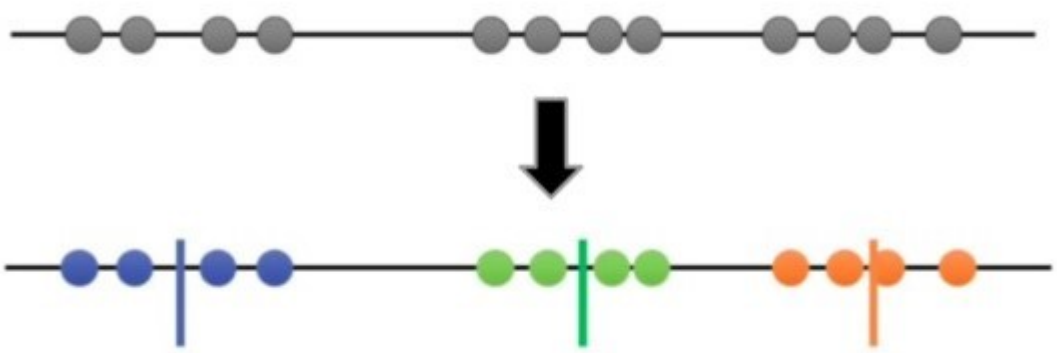


Repeat until stop:

- **Until we get the lowest sum of variance and pick those cluster as our result**

How K-means works

Final result of clustering is



Algorithm K-Means

1. Initialize cluster centroids

$$\mu_1, \mu_2, \ldots, \mu_k \in \mathbb{R}^n$$

2. Repeat until convergence (no change)
   1. Assign each ith observation to the closest cluster centroid

$$c^{(i)} := \arg\min_j ||x^{(i)} - \mu_j||^2$$

   2. For each cluster, move the centroid to the mean of observations belong to the cluster

$$\mu_j = \frac{\sum_{i=1}^{m} 1\{c^{(i)} = j\} x^{(i)}}{\sum_{i=1}^{m} 1\{c^{(i)} = j\}}$$

Algorithm K-Means

1. Initialize cluster centroids

$$\mu_1, \mu_2, \ldots, \mu_k \in \mathbb{R}^n$$

2. Repeat until convergence (no change)
   1. Assign each ith observation to the closest cluster centroid

$$c^{(i)} := \arg\min_j ||x^{(i)} - \mu_j||^2$$

   2. For each cluster, move the centroid to the mean of observations belong to the cluster

$$\mu_j = \frac{\sum_{i=1}^{m} 1\{c^{(i)} = j\} x^{(i)}}{\sum_{i=1}^{m} 1\{c^{(i)} = j\}}$$

Algorithm K-Means

1. Initialize cluster centroids

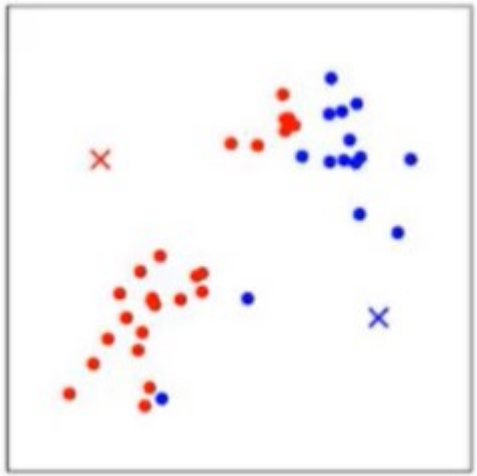
$$\mu_1, \mu_2, \ldots, \mu_k \in \mathbb{R}^n$$

2. Repeat until convergence (no change)
   1. Assign each ith observation to the closest cluster centroid

$$c^{(i)} := \arg\min_j ||x^{(i)} - \mu_j||^2$$

   2. For each cluster, move the centroid to the mean of observations belong to the cluster

$$\mu_j = \frac{\sum_{i=1}^{m} 1\{c^{(i)} = j\} x^{(i)}}{\sum_{i=1}^{m} 1\{c^{(i)} = j\}}$$



Algorithm K-Means

1. Initialize cluster centroids

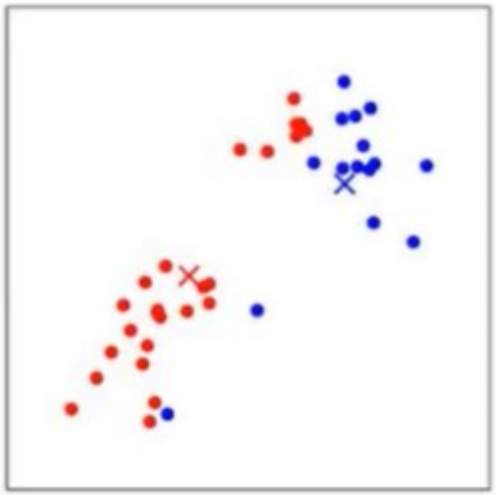
$$\mu_1, \mu_2, \ldots, \mu_k \in \mathbb{R}^n$$

2. Repeat until convergence (no change)
   1. Assign each ith observation to the closest cluster centroid

$$c^{(i)} := \arg\min_j ||x^{(i)} - \mu_j||^2$$

   2. For each cluster, move the centroid to the mean of observations belong to the cluster

$$\mu_j = \frac{\sum_{i=1}^{m} 1\{c^{(i)} = j\} x^{(i)}}{\sum_{i=1}^{m} 1\{c^{(i)} = j\}}$$



Algorithm K-Means

1. Initialize cluster centroids

$$\mu_1, \mu_2, \ldots, \mu_k \in \mathbb{R}^n$$

2. Repeat until convergence (no change)
   1. Assign each ith observation to the closest cluster centroid

$$c^{(i)} := \arg\min_j ||x^{(i)} - \mu_j||^2$$

   2. For each cluster, move the centroid to the mean of observations belong to the cluster

$$\mu_j = \frac{\sum_{i=1}^{m} 1\{c^{(i)} = j\} x^{(i)}}{\sum_{i=1}^{m} 1\{c^{(i)} = j\}}$$

Algorithm K-Means

1. Initialize cluster centroids

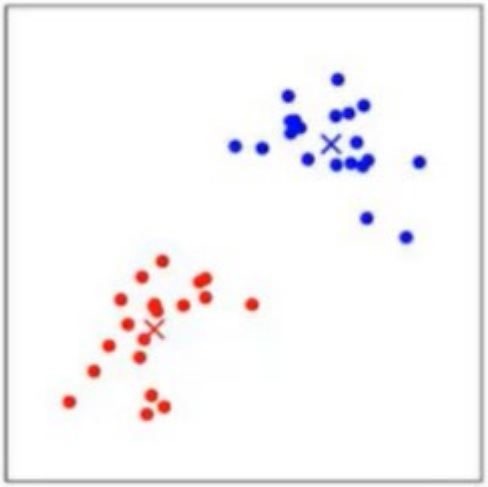
$$\mu_1, \mu_2, \ldots, \mu_k \in \mathbb{R}^n$$

2. Repeat until convergence (no change)
   1. Assign each ith observation to the closest cluster centroid

$$c^{(i)} := \arg\min_j ||x^{(i)} - \mu_j||^2$$

   2. For each cluster, move the centroid to the mean of observations belong to the cluster

$$\mu_j = \frac{\sum_{i=1}^{m} 1\{c^{(i)} = j\} x^{(i)}}{\sum_{i=1}^{m} 1\{c^{(i)} = j\}}$$



Summary: Algorithm K-Means

1. Initialize cluster centroids

$$\mu_1, \mu_2, \ldots, \mu_k \in \mathbb{R}^n$$

2. Repeat until convergence (no change)
   1. Assign each ith observation to the closest cluster centroid

$$c^{(i)} := \arg\min_j ||x^{(i)} - \mu_j||^2$$

   2. For each cluster, move the centroid to the mean of observations belong to the cluster

$$\mu_j = \frac{\sum_{i=1}^{m} 1\{c^{(i)} = j\} x^{(i)}}{\sum_{i=1}^{m} 1\{c^{(i)} = j\}}$$

Algorithm K-Means

K-Means algorithm is guaranteed to converge. In particular we look at the **distortion function**:

$$J(c, \mu) = \sum_{i=1}^{m} ||x^{(i)} - \mu_{c^{(i)}}||^2.$$

- J measures the sum of squared distances between each observation $x^{(i)}$ to its corresponding cluster centroid $\mu_{c^{(i)}}$.
- K-Means repeatedly minimizes J with respect to c while holding μ fixed;
- and then minimizes J with respect to μ while holding c fixed.
- Thus, J is monotonically decrease and the value of J must converge.

Sample: Algorithm K-Means

Given a data set as follows:

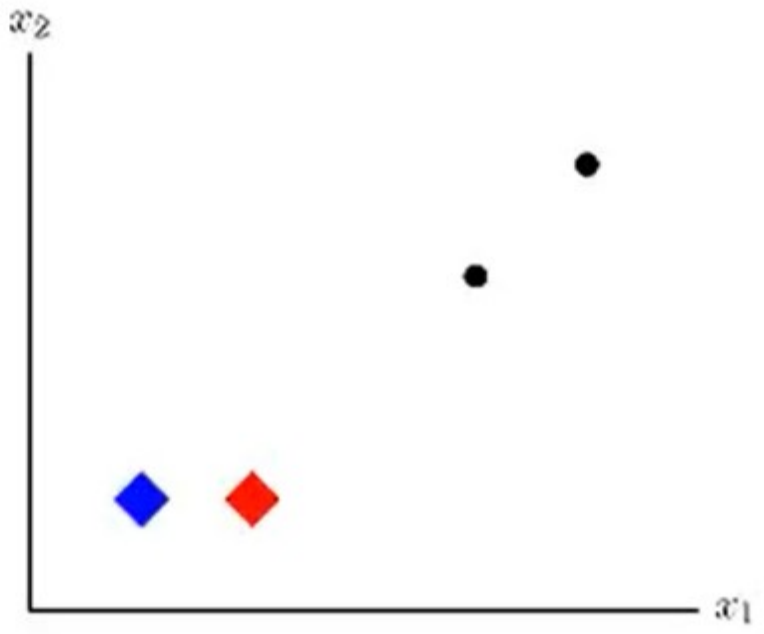
$x^{(i)}$	x_1	x_2
$x^{(1)}$	1	1
$x^{(2)}$	2	1
$x^{(3)}$	4	3
$x^{(4)}$	5	4

Find two clusters using K-Means!

Sample: Algorithm K-Means

1. Initialize two cluster centroids

$$\mu_1 = x^{(1)} = (1, 1)$$
$$\mu_2 = x^{(2)} = (2, 1)$$

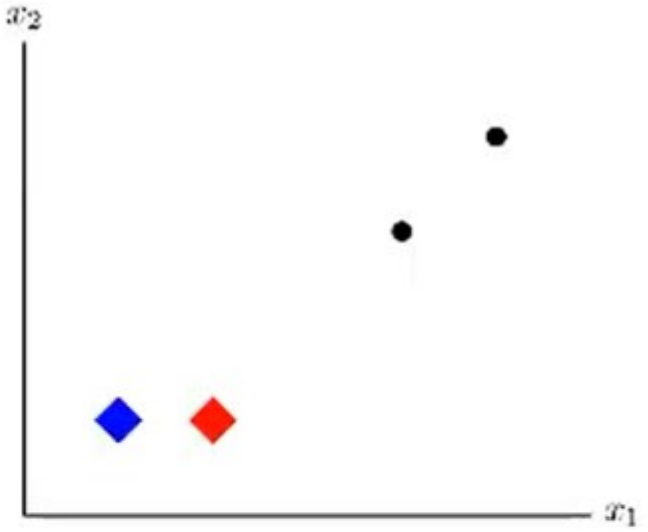


Sample: Algorithm K-Means

Repeat until convergence,

- Assign each ith observation to the closest cluster centroid

$$c^{(i)} := \arg\min_j ||x^{(i)} - \mu_j||^2$$



For $x^{(1)}$,

$$||x^{(1)} - \mu_1||^2 = \sqrt{(1-1)^2 + (1-1)^2} = 0$$
$$||x^{(1)} - \mu_2||^2 = \sqrt{(1-2)^2 + (1-1)^2} = 1$$

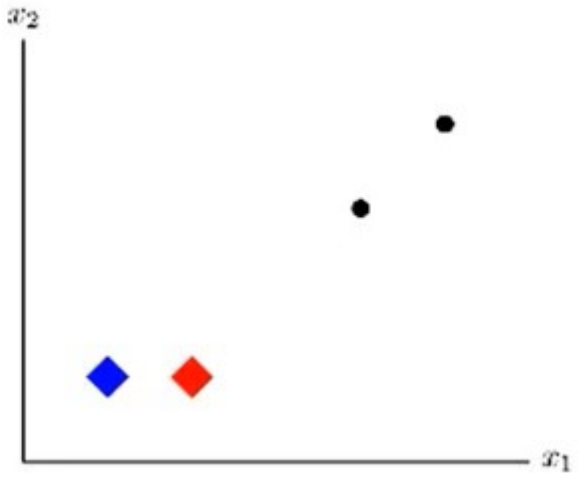
therefore, $c^{(1)} = 1$.

Sample: Algorithm K-Means

Repeat until convergence,

- Assign each ith observation to the closest cluster centroid

$$c^{(i)} := \arg\min_j ||x^{(i)} - \mu_j||^2$$



For $x^{(2)}$,

$$||x^{(2)} - \mu_1||^2 = \sqrt{(2-1)^2 + (1-1)^2} = 1$$
$$||x^{(2)} - \mu_2||^2 = \sqrt{(2-2)^2 + (1-1)^2} = 0$$

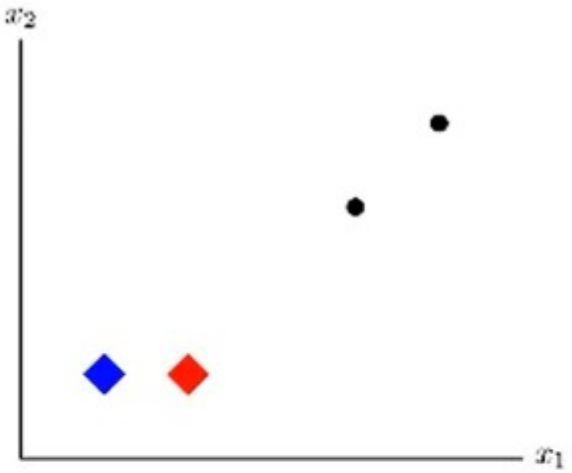
therefore, $c^{(2)} = 2$.

Sample: Algorithm K-Means

Repeat until convergence,

- Assign each ith observation to the closest cluster centroid

$$c^{(i)} := \arg\min_j ||x^{(i)} - \mu_j||^2$$



For $x^{(3)}$,

$$||x^{(3)} - \mu_1||^2 = \sqrt{(4-1)^2 + (3-1)^2} = 3.61$$
$$||x^{(3)} - \mu_2||^2 = \sqrt{(4-2)^2 + (3-1)^2} = 2.83$$

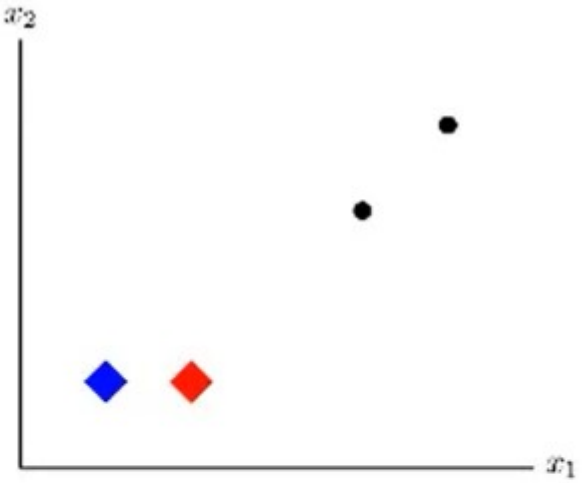
therefore, $c^{(3)} = 2$.

Sample: Algorithm K-Means

Repeat until convergence,

- Assign each ith observation to the closest cluster centroid

$$c^{(i)} := \arg\min_j ||x^{(i)} - \mu_j||^2$$



For $x^{(4)}$,

$$||x^{(4)} - \mu_1||^2 = \sqrt{(5-1)^2 + (4-1)^2} = 5$$
$$||x^{(4)} - \mu_2||^2 = \sqrt{(5-2)^2 + (4-1)^2} = 4.24$$

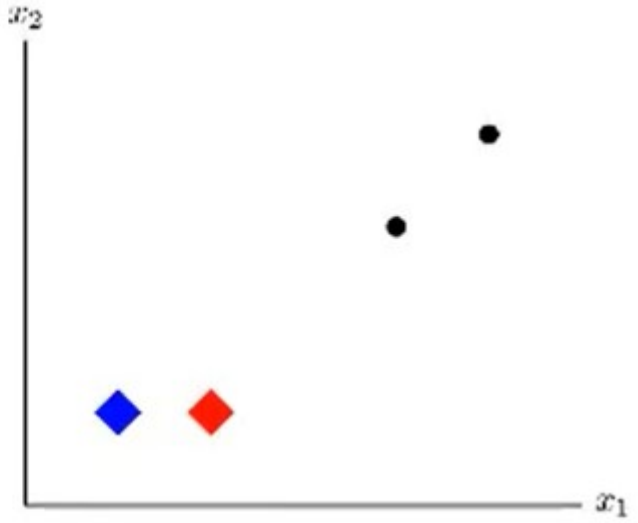
therefore, $c^{(4)} = 2$.

Sample: Algorithm K-Means

Repeat until convergence,

- Assign each ith observation to the closest cluster centroid

$$c^{(i)} := \arg\min_j ||x^{(i)} - \mu_j||^2$$



Put together, we have $c^{(1)} = 1$ and $c^{(2)} = c^{(3)} = c^{(4)} = 2$.
Evaluating the distortion function,

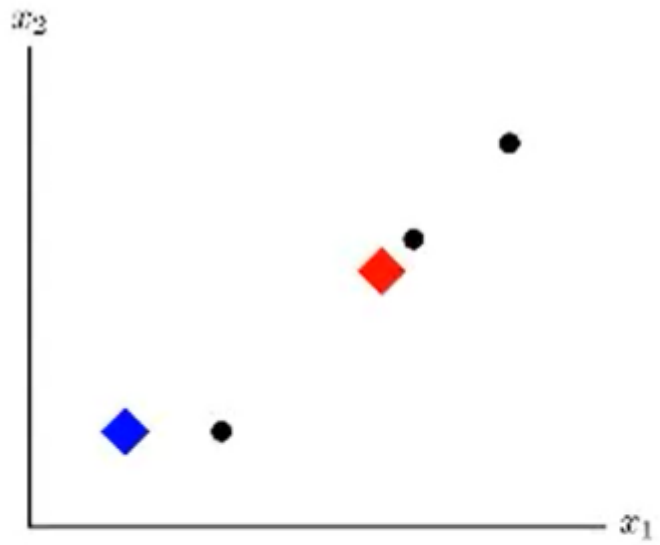
$$J(c, \mu) = \sum_{i=1}^{m} ||x^{(i)} - \mu_{c^{(i)}}||^2 = 0 + 0 + 2.83 + 4.24 = 7.07.$$

Sample: Algorithm K-Means

Repeat until convergence,

- For each cluster, move the centroid to the mean of observations belong to the cluster

$$\mu_j = \frac{\sum_{i=1}^{m} 1\{c^{(i)} = j\} x^{(i)}}{\sum_{i=1}^{m} 1\{c^{(i)} = j\}}$$



Therefore,

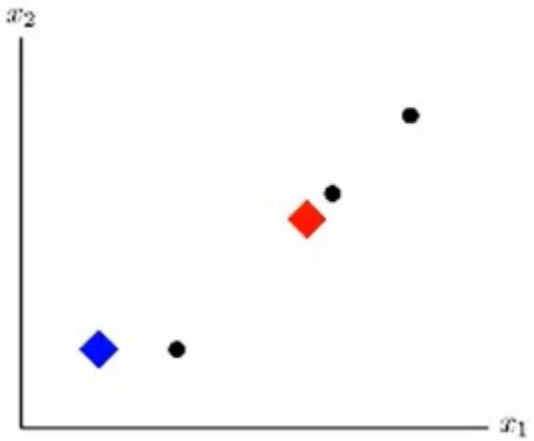
$$\mu_1 = x^{(1)} = (1, 1)$$

$$\mu_2 = \frac{x^{(2)} + x^{(3)} + x^{(4)}}{3} = \frac{1}{3}(2 + 4 + 5, 1 + 3 + 4) = \left(\frac{11}{3}, \frac{8}{3}\right) = (3.67, 2.67)$$

Sample: Algorithm K-Means

- Assign each ith observation to the closest cluster centroid

$$c^{(i)} := \arg\min_j ||x^{(i)} - \mu_j||^2$$



For $x^{(1)}$,

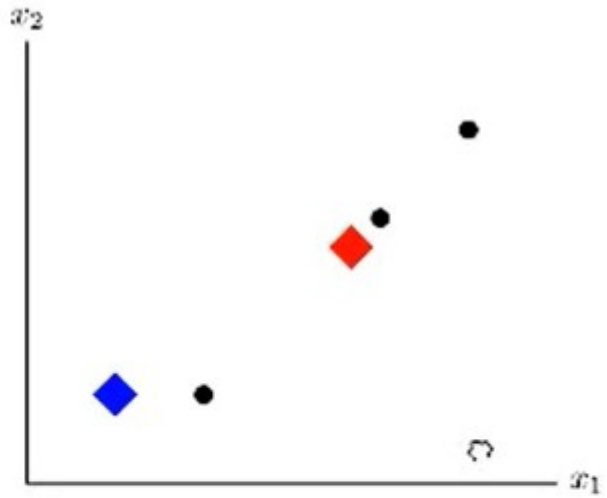
$$||x^{(1)} - \mu_1||^2 = \sqrt{(1-1)^2 + (1-1)^2} = 0$$
$$||x^{(1)} - \mu_2||^2 = \sqrt{(1-3.67)^2 + (1-2.67)^2} = 3.15$$

therefore, $c^{(1)} = 1$.

Sample: Algorithm K-Means

- Assign each ith observation to the closest cluster centroid

$$c^{(i)} := \arg\min_j ||x^{(i)} - \mu_j||^2$$



For $x^{(2)}$,

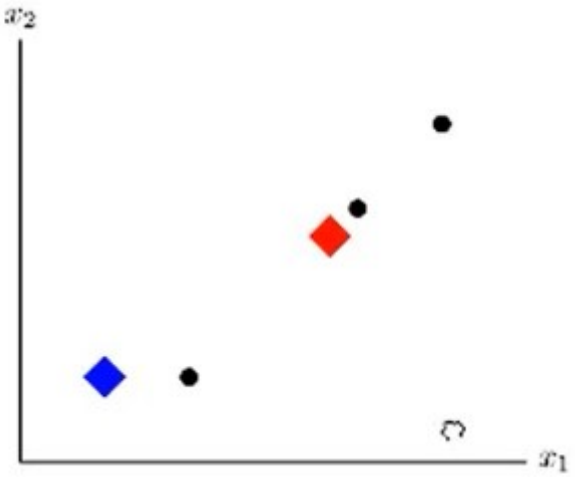
$$||x^{(2)} - \mu_1||^2 = \sqrt{(2-1)^2 + (1-1)^2} = 1$$
$$||x^{(2)} - \mu_2||^2 = \sqrt{(2-3.67)^2 + (1-2.67)^2} = 2.36$$

therefore, $c^{(2)} = 1$.

Sample: Algorithm K-Means

- Assign each ith observation to the closest cluster centroid

$$c^{(i)} := \arg\min_j ||x^{(i)} - \mu_j||^2$$



For $x^{(3)}$,

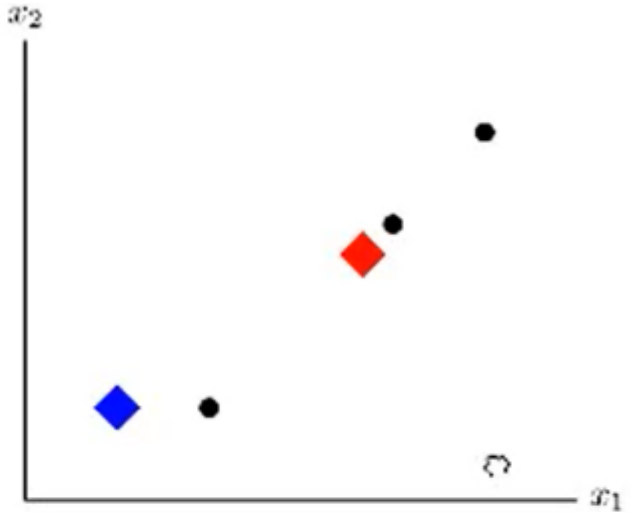
$$||x^{(3)} - \mu_1||^2 = \sqrt{(4-1)^2 + (3-1)^2} = 3.61$$
$$||x^{(3)} - \mu_2||^2 = \sqrt{(4-3.67)^2 + (3-2.67)^2} = 0.47$$

therefore, $c^{(3)} = 2$.

Sample: Algorithm K-Means

- Assign each ith observation to the closest cluster centroid

$$c^{(i)} := \arg\min_j ||x^{(i)} - \mu_j||^2$$



For $x^{(4)}$,

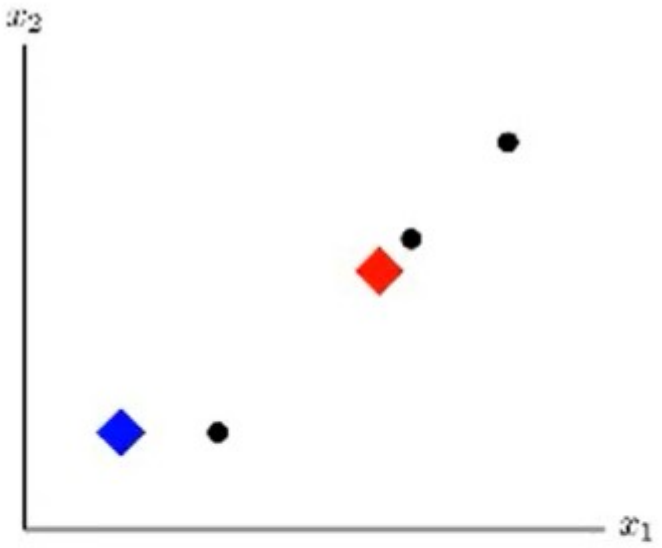
$$||x^{(4)} - \mu_1||^2 = \sqrt{(5-1)^2 + (4-1)^2} = 5$$
$$||x^{(4)} - \mu_2||^2 = \sqrt{(5-3.67)^2 + (4-2.67)^2} = 1.88$$

therefore, $c^{(4)} = 2$.

Sample: Algorithm K-Means

- Assign each ith observation to the closest cluster centroid

$$c^{(i)} := \arg\min_j ||x^{(i)} - \mu_j||^2$$



Put together, we have $c^{(1)} = c^{(2)} = 1$ and $c^{(3)} = c^{(4)} = 2$.
Evaluating the distortion function,

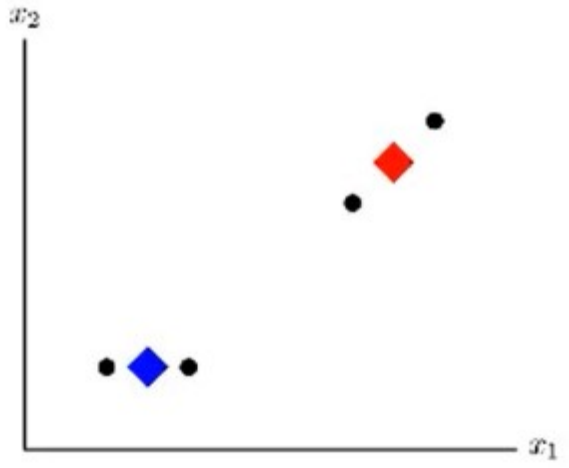
$$J(c, \mu) = \sum_{i=1}^{m} ||x^{(i)} - \mu_{c^{(i)}}||^2 = 0 + 1 + 0.47 + 1.88 = 3.35.$$

Sample: Algorithm K-Means

Repeat until convergence,

- For each cluster, move the centroid to the mean of observations belong to the cluster

$$\mu_j = \frac{\sum_{i=1}^{m} 1\{c^{(i)} = j\} x^{(i)}}{\sum_{i=1}^{m} 1\{c^{(i)} = j\}}$$



Therefore,

$$\mu_1 = \frac{x^{(1)} + x^{(2)}}{2} = \frac{1}{2}(1+2, 1+1) = \left(\frac{3}{2}, \frac{2}{2}\right) = (1.5, 1)$$

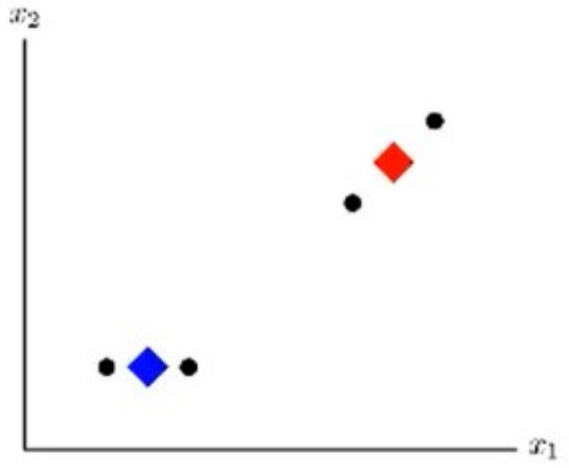
$$\mu_2 = \frac{x^{(3)} + x^{(4)}}{2} = \frac{1}{2}(4+5, 3+4) = \left(\frac{9}{2}, \frac{7}{2}\right) = (4.5, 3.5)$$

Sample: Algorithm K-Means

Repeat until convergence,

- For each cluster, move the centroid to the mean of observations belong to the cluster

$$\mu_j = \frac{\sum_{i=1}^{m} 1\{c^{(i)} = j\} x^{(i)}}{\sum_{i=1}^{m} 1\{c^{(i)} = j\}}$$



Therefore,

$$\mu_1 = \frac{x^{(1)} + x^{(2)}}{2} = \frac{1}{2}(1+2, 1+1) = \left(\frac{3}{2}, \frac{2}{2}\right) = (1.5, 1)$$

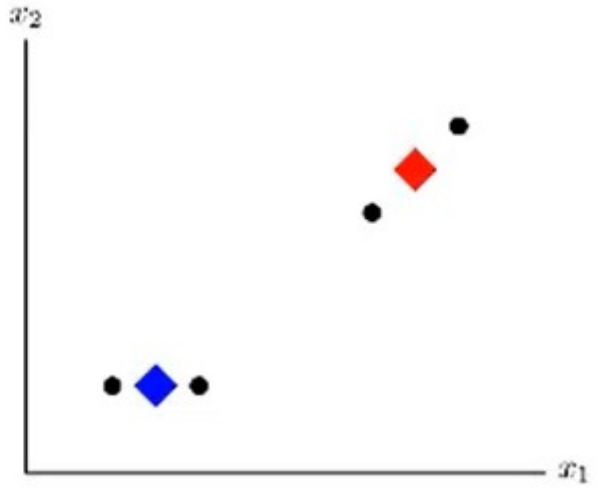
$$\mu_2 = \frac{x^{(3)} + x^{(4)}}{2} = \frac{1}{2}(4+5, 3+4) = \left(\frac{9}{2}, \frac{7}{2}\right) = (4.5, 3.5)$$

Sample: Algorithm K-Means

Repeat until convergence,

- Assign each ith observation to the closest cluster centroid

$$c^{(i)} := \arg\min_j ||x^{(i)} - \mu_j||^2$$



For $x^{(1)}$,

$$||x^{(1)} - \mu_1||^2 = \sqrt{(1-1.5)^2 + (1-1)^2} = 0.5$$
$$||x^{(1)} - \mu_2||^2 = \sqrt{(1-4.5)^2 + (1-3.5)^2} = 4.3$$

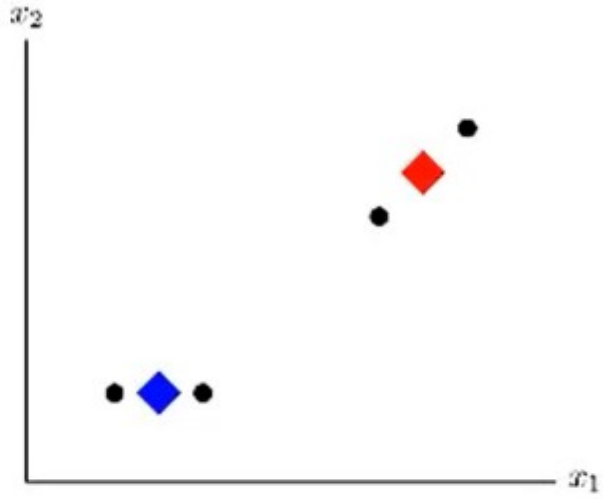
therefore, $c^{(1)} = 1$.

Sample: Algorithm K-Means

Repeat until convergence,

- Assign each ith observation to the closest cluster centroid

$$c^{(i)} := \arg\min_j ||x^{(i)} - \mu_j||^2$$



For $x^{(2)}$,

$$||x^{(2)} - \mu_1||^2 = \sqrt{(2-1.5)^2 + (1-1)^2} = 0.5$$
$$||x^{(2)} - \mu_2||^2 = \sqrt{(2-4.5)^2 + (1-3.5)^2} = 3.53$$

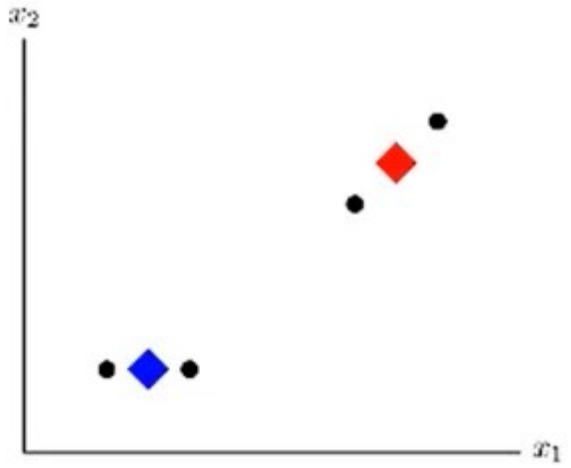
therefore, $c^{(2)} = 1$.

Sample: Algorithm K-Means

Repeat until convergence,

- Assign each ith observation to the closest cluster centroid

$$c^{(i)} := \arg\min_j ||x^{(i)} - \mu_j||^2$$



For $x^{(3)}$,

$$||x^{(3)} - \mu_1||^2 = \sqrt{(4-1.5)^2 + (3-1)^2} = 3.2$$
$$||x^{(3)} - \mu_2||^2 = \sqrt{(4-4.5)^2 + (3-3.5)^2} = 0.71$$

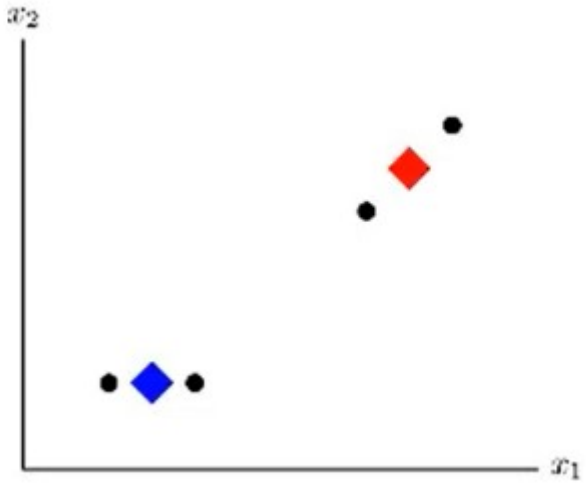
therefore, $c^{(3)} = 2$.

Sample: Algorithm K-Means

Repeat until convergence,

- Assign each ith observation to the closest cluster centroid

$$c^{(i)} := \arg\min_j ||x^{(i)} - \mu_j||^2$$



For $x^{(4)}$,

$$||x^{(4)} - \mu_1||^2 = \sqrt{(5-1.5)^2 + (4-1)^2} = 4.61$$
$$||x^{(4)} - \mu_2||^2 = \sqrt{(5-4.5)^2 + (4-3.5)^2} = 0.71$$

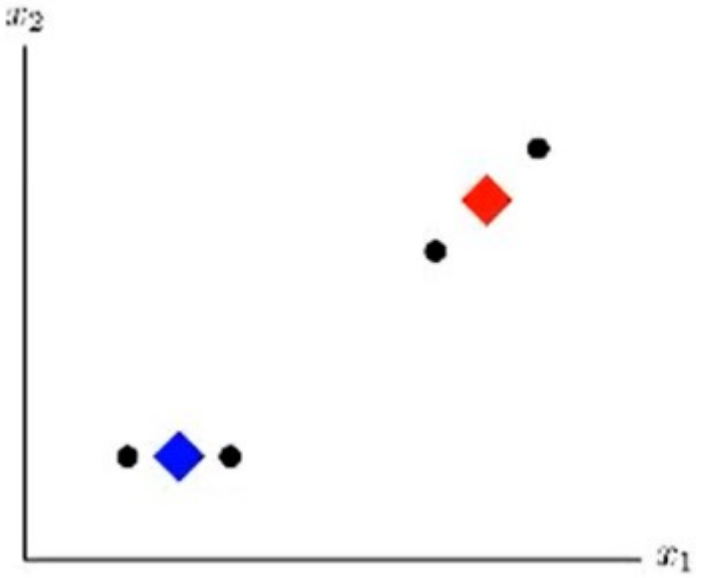
therefore, $c^{(4)} = 2$.

Sample: Algorithm K-Means

Repeat until convergence,

- Assign each ith observation to the closest cluster centroid

$$c^{(i)} := \arg\min_j ||x^{(i)} - \mu_j||^2$$



Put together, we have $x^{(1)} = x^{(2)} = 1$ and $x^{(3)} = x^{(4)} = 2$.
Evaluating the distortion function,

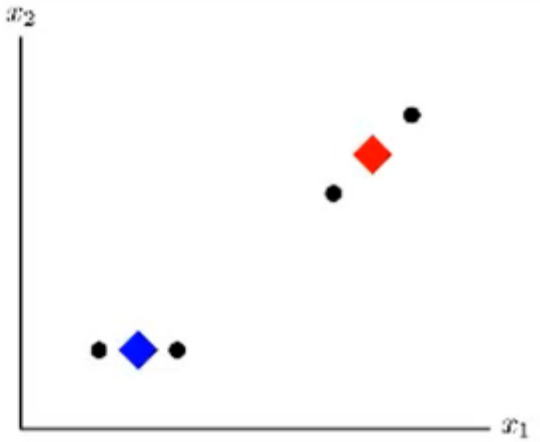
$$J(c, \mu) = \sum_{i=1}^{m} ||x^{(i)} - \mu_{c^{(i)}}||^2 = 0.5 + 0.5 + 0.71 + 0.71 = 2.42.$$

Sample: Algorithm K-Means

Repeat until convergence,

- Assign each ith observation to the closest cluster centroid

$$c^{(i)} := \arg\min_j ||x^{(i)} - \mu_j||^2$$

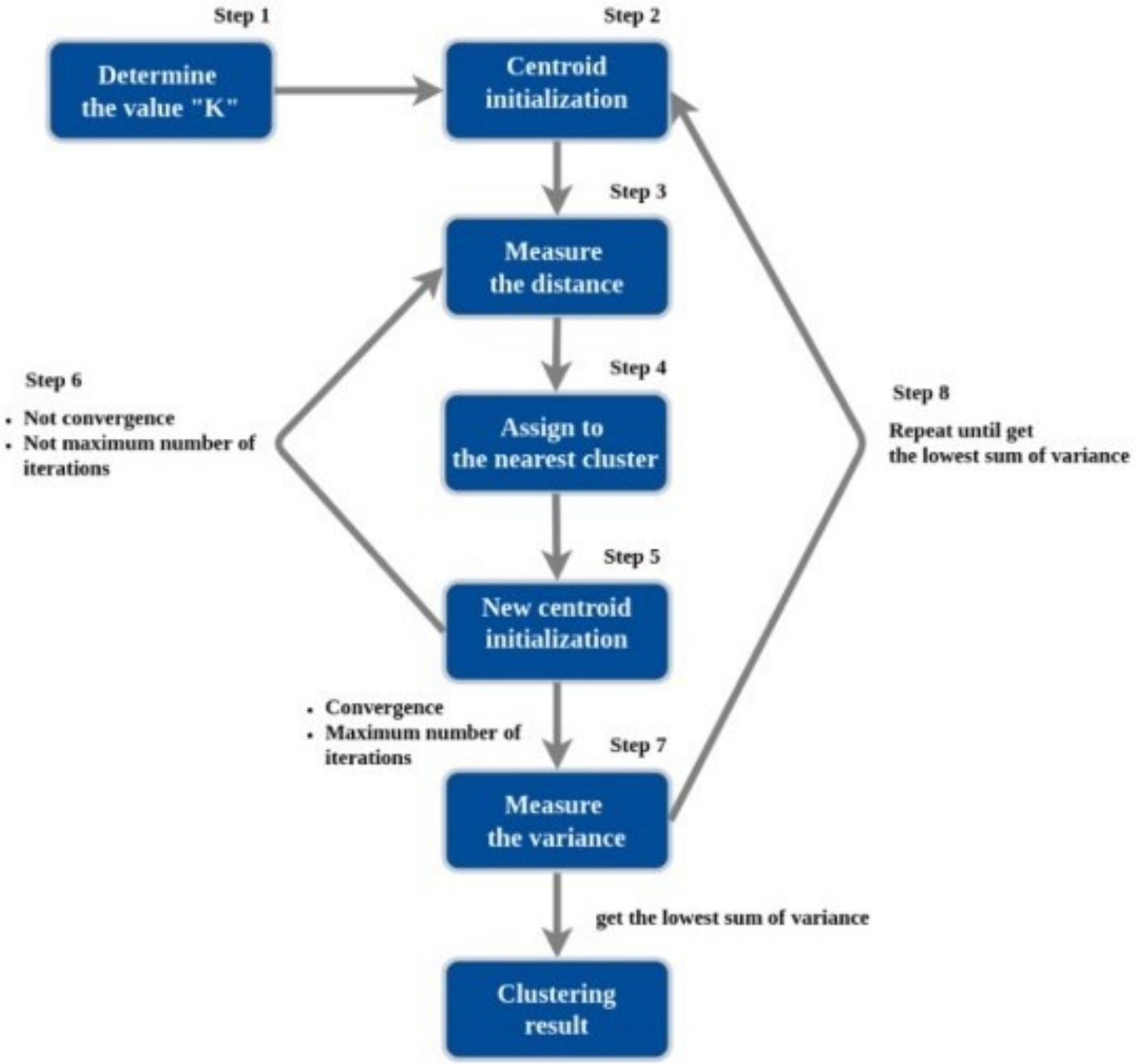


Put together, we have $x^{(1)} = x^{(2)} = 1$ and $x^{(3)} = x^{(4)} = 2$.
Evaluating the distortion function,

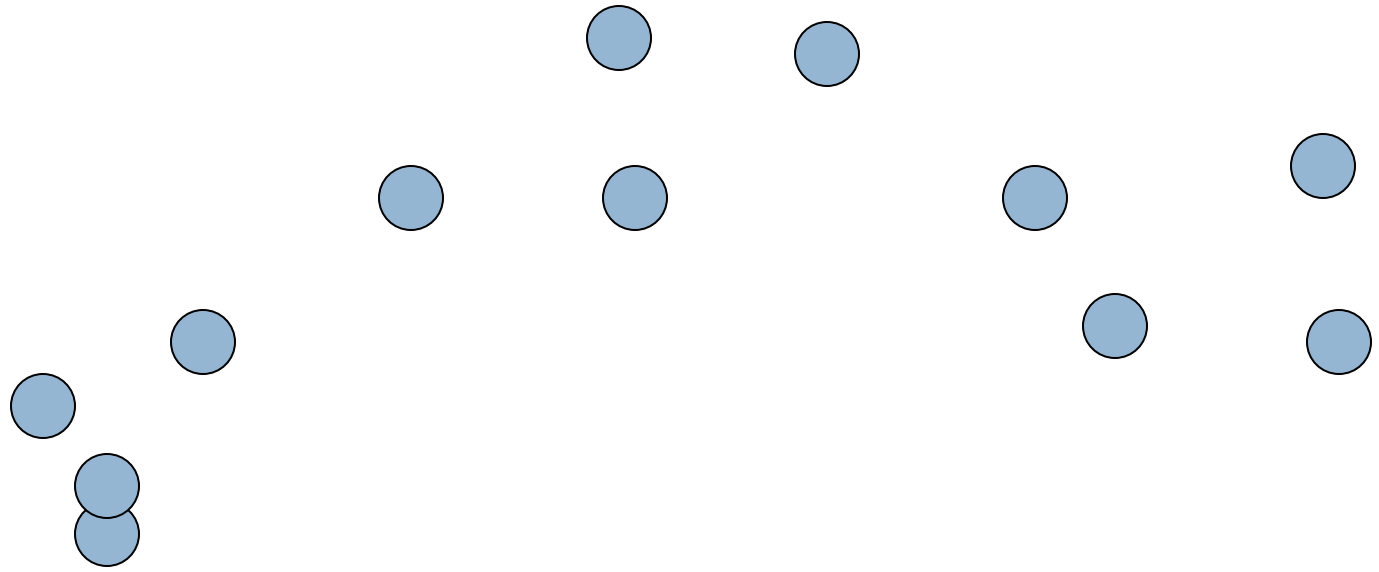
$$J(c, \mu) = \sum_{i=1}^{m} ||x^{(i)} - \mu_{c^{(i)}}||^2 = 0.5 + 0.5 + 0.71 + 0.71 = 2.42.$$

We see that the cluster membership $c^{(i)}$ doesn't change, which implies that the next centroids to remain the same. Thus, K-Means stops here, and yields the above clustering.

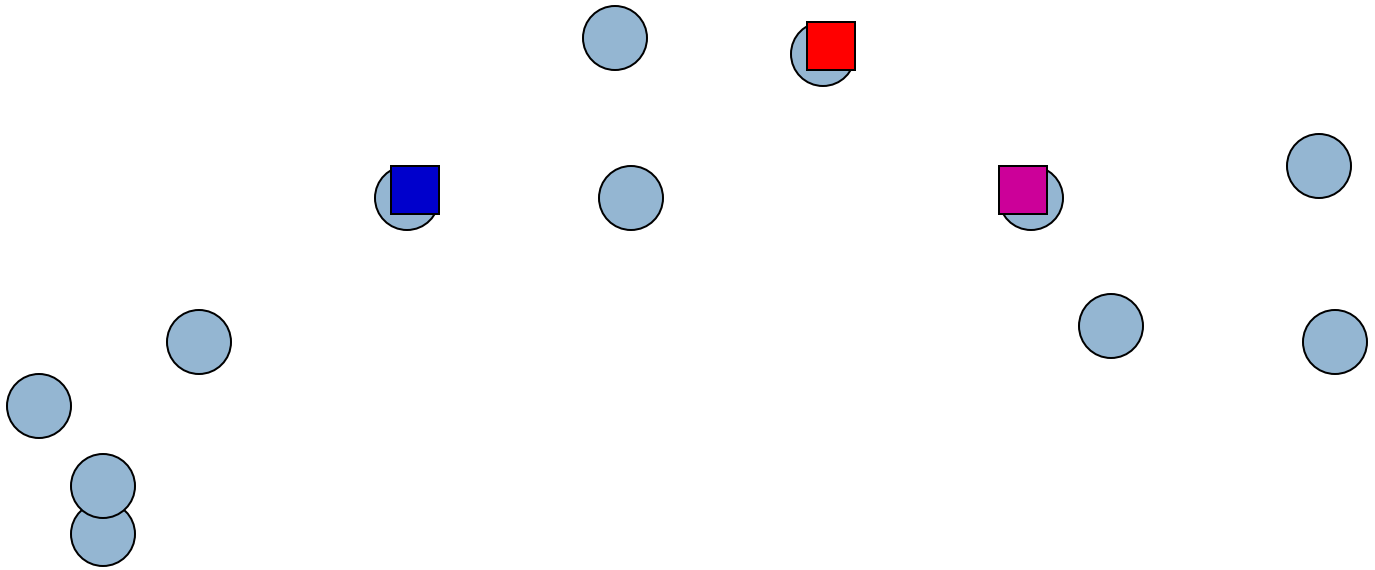
How K-means works



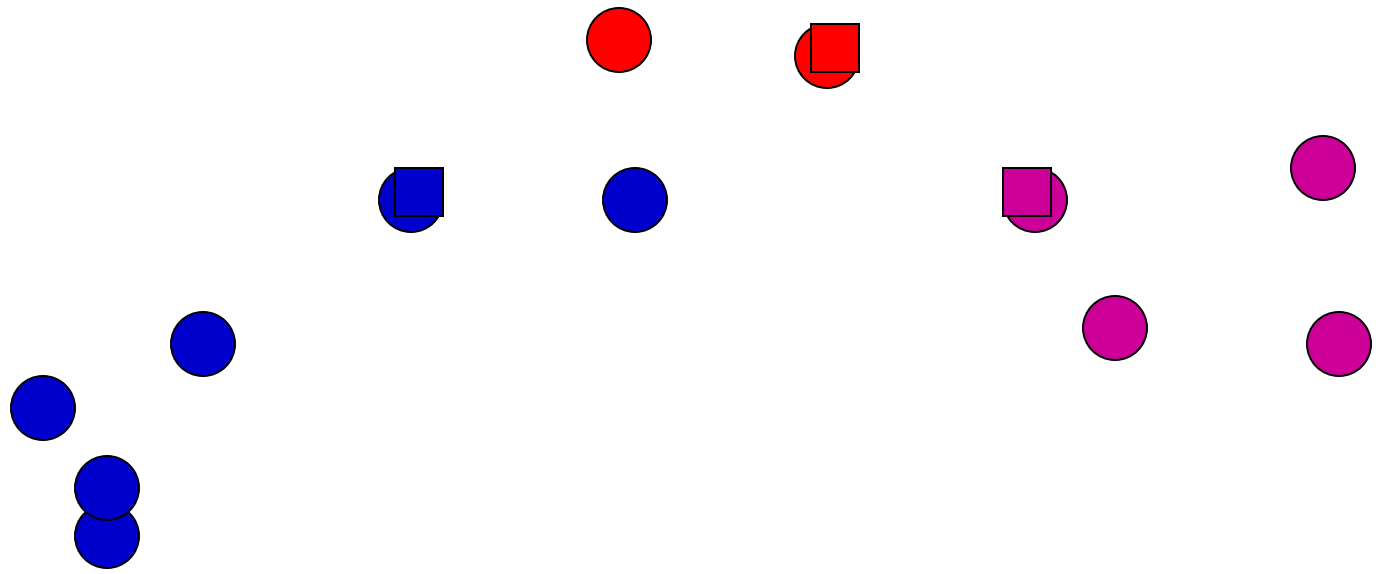
K-means: an example



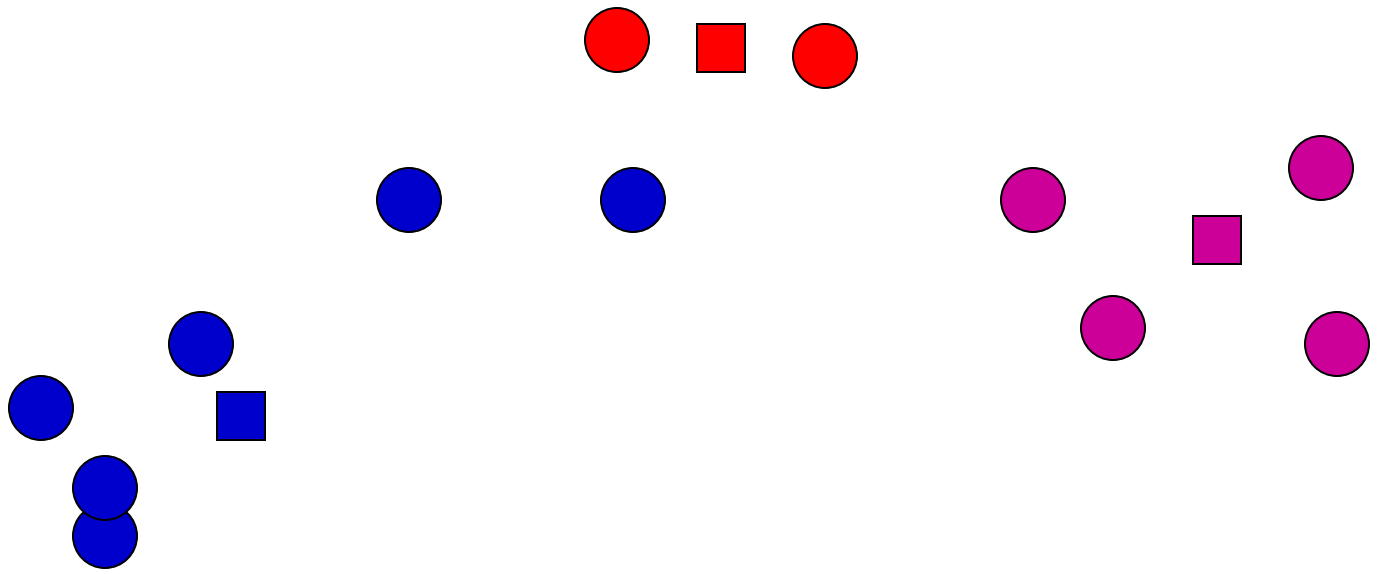
K-means: Initialize centers randomly



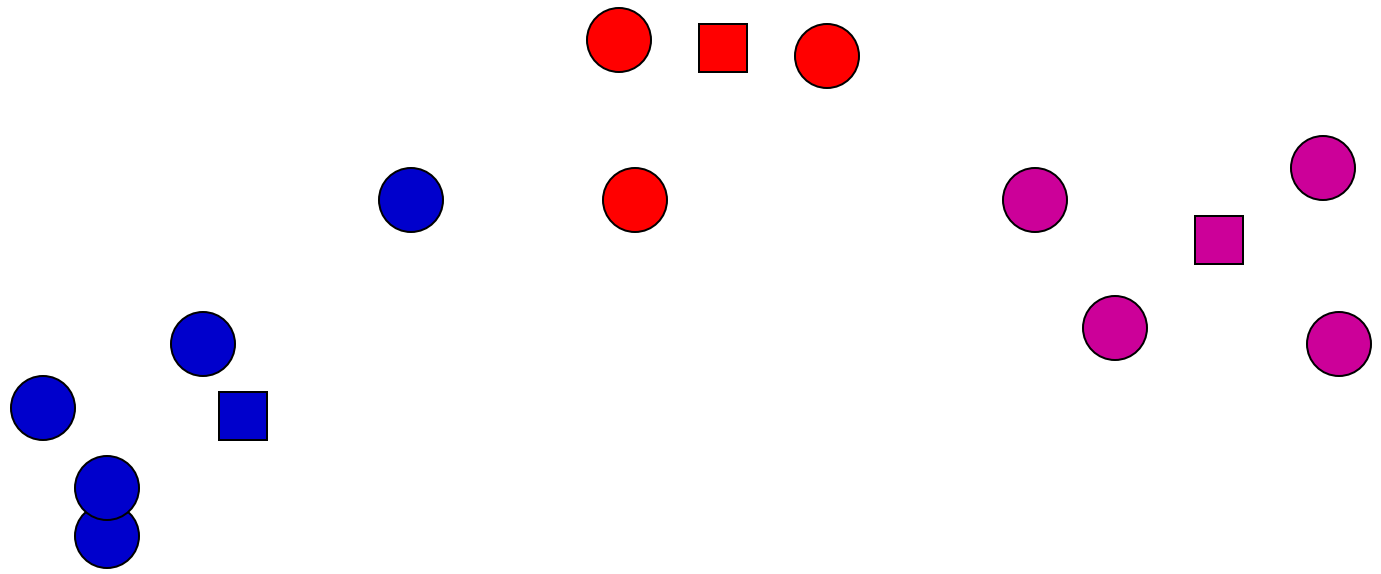
K-means: assign points to nearest center



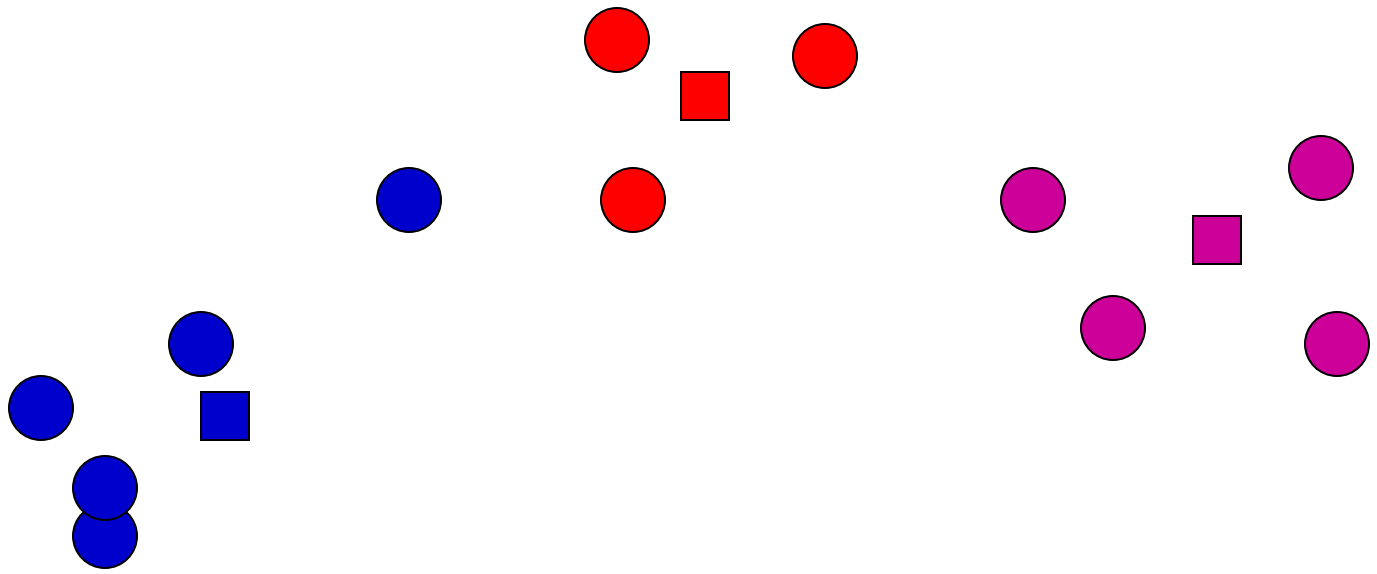
K-means: readjust centers



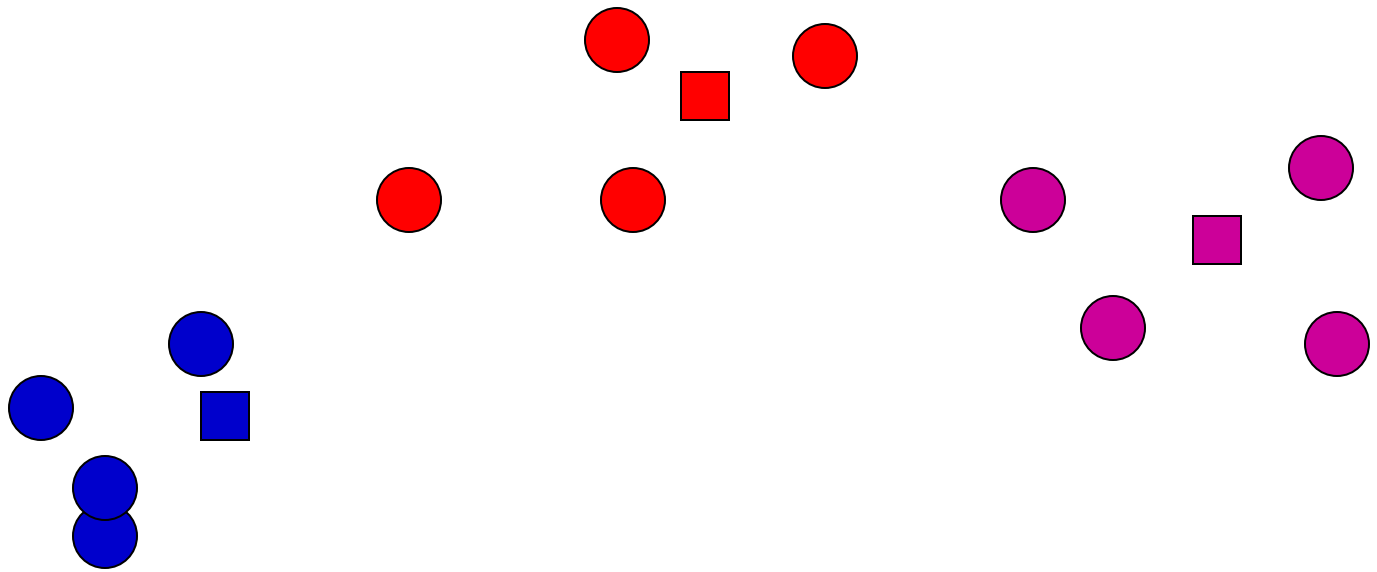
K-means: assign points to nearest center



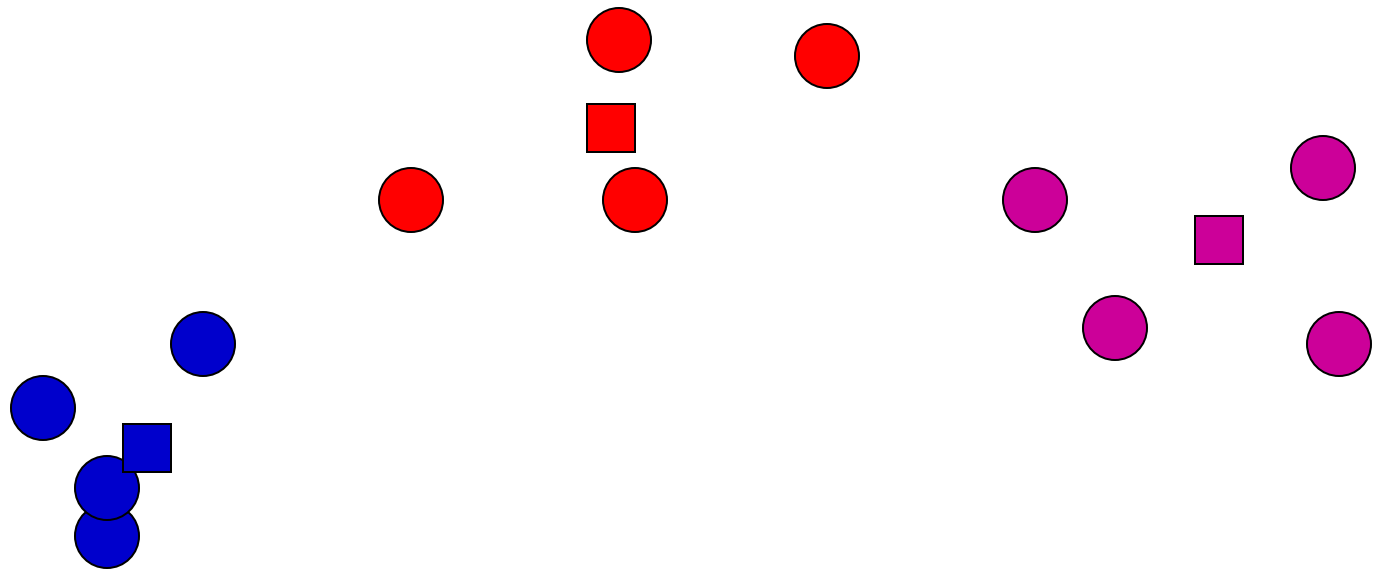
K-means: readjust centers



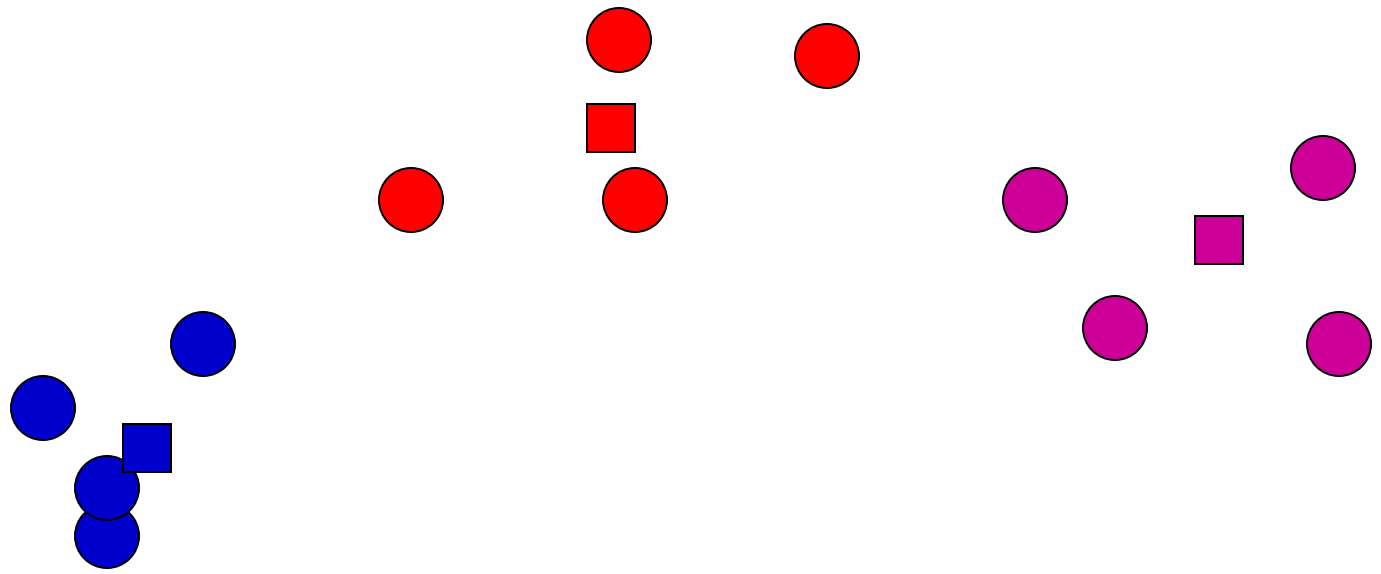
K-means: assign points to nearest center



K-means: readjust centers



K-means: assign points to nearest center

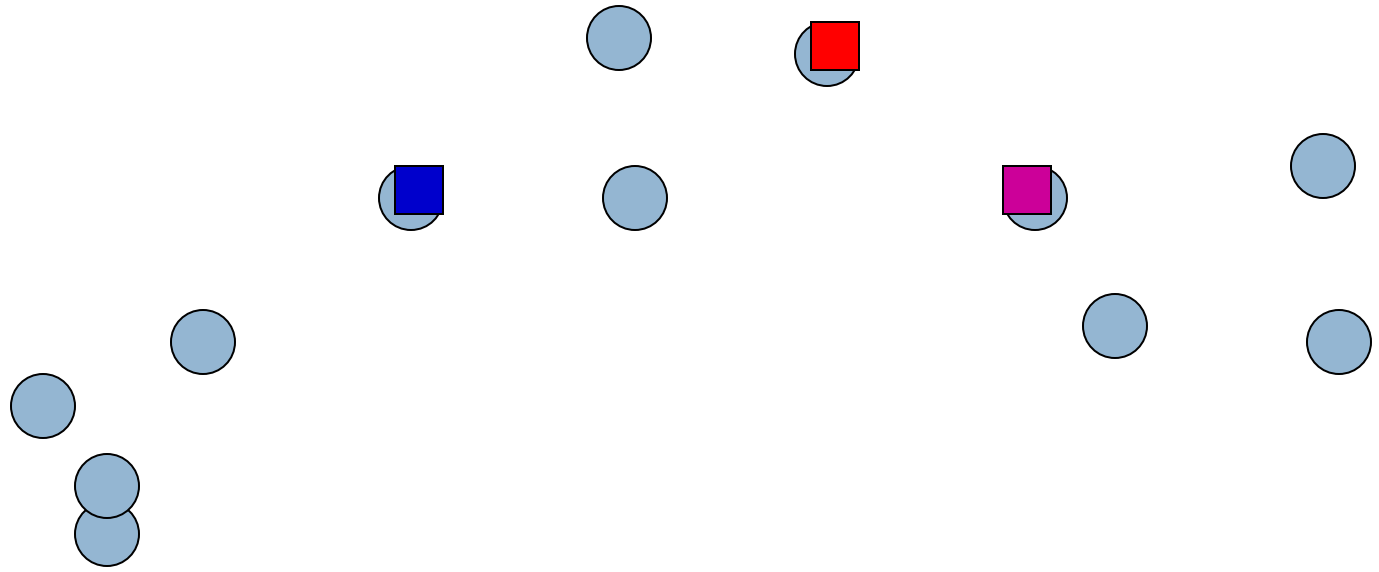


No changes: Done

K-means

Iterate:

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



How do we do this?

K-means

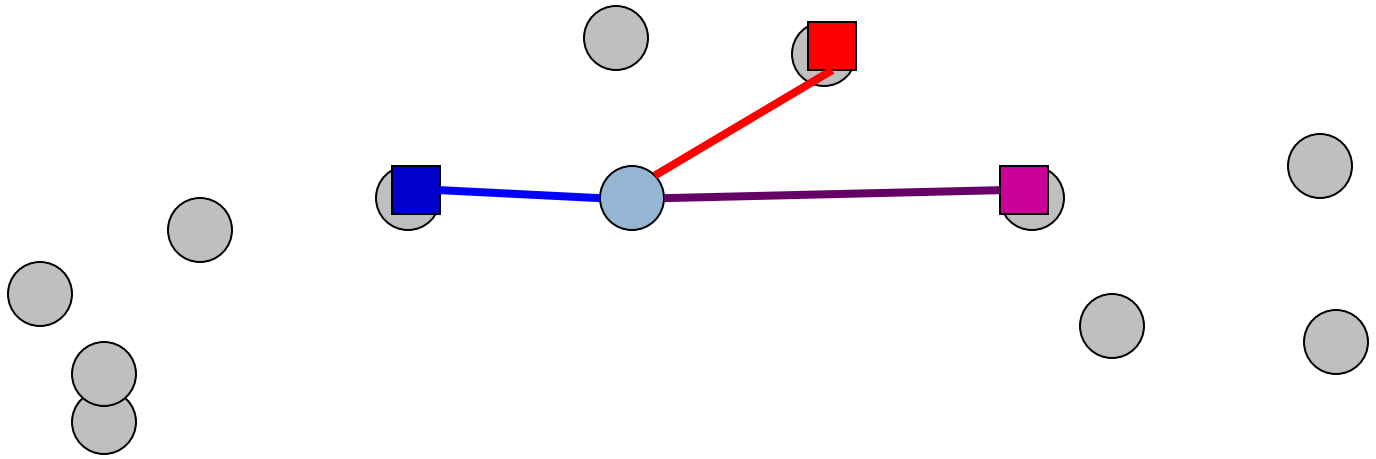
Iterate:

- **Assign/cluster each example to closest center**

  iterate over each point:

  - get distance to each cluster center
  - assign to closest center (hard cluster)

- Recalculate centers as the mean of the points in a cluster



K-means

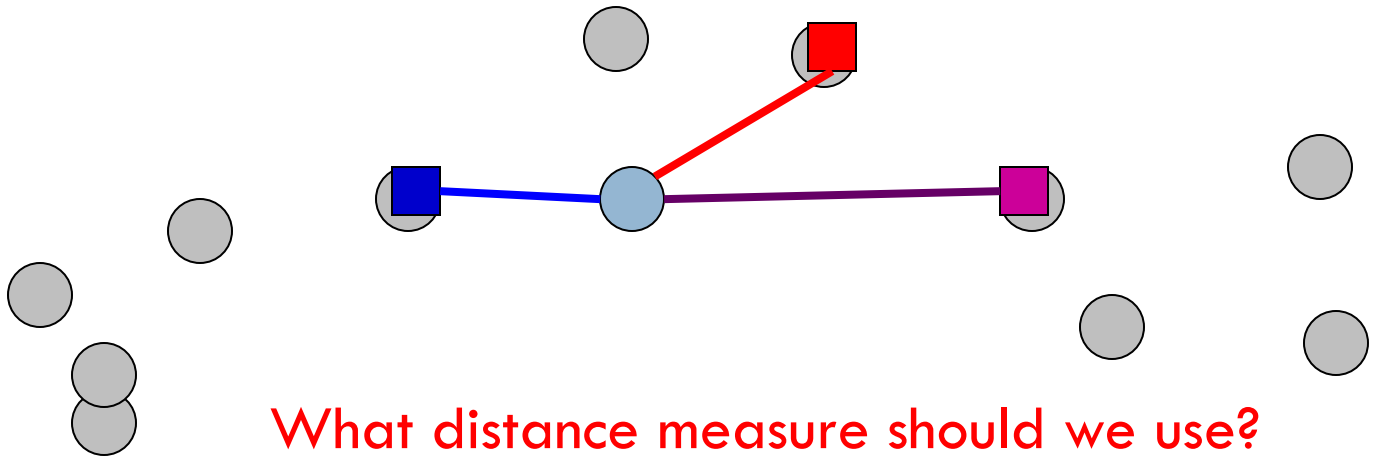
Iterate:

- **Assign/cluster each example to closest center**

  iterate over each point:

  - get **distance** to each cluster center
  - assign to closest center (hard cluster)

- Recalculate centers as the mean of the points in a cluster



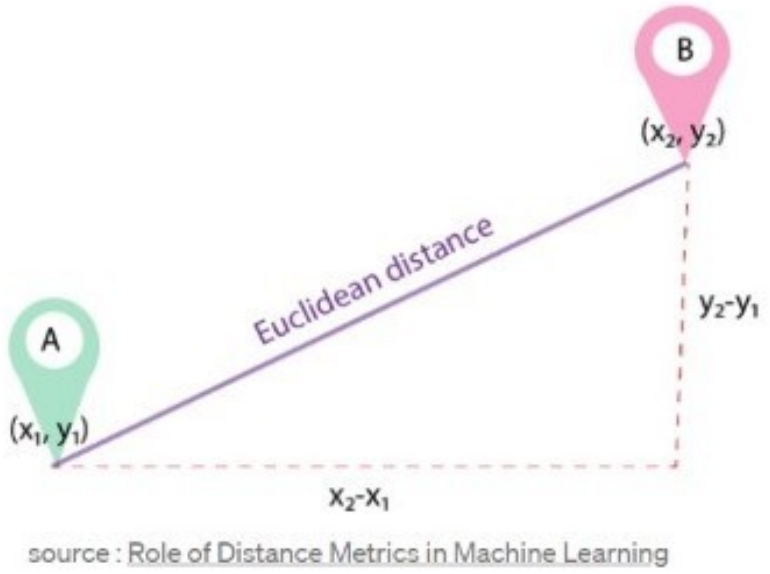
What distance measure should we use?

Distance measures: Euclidean

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

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

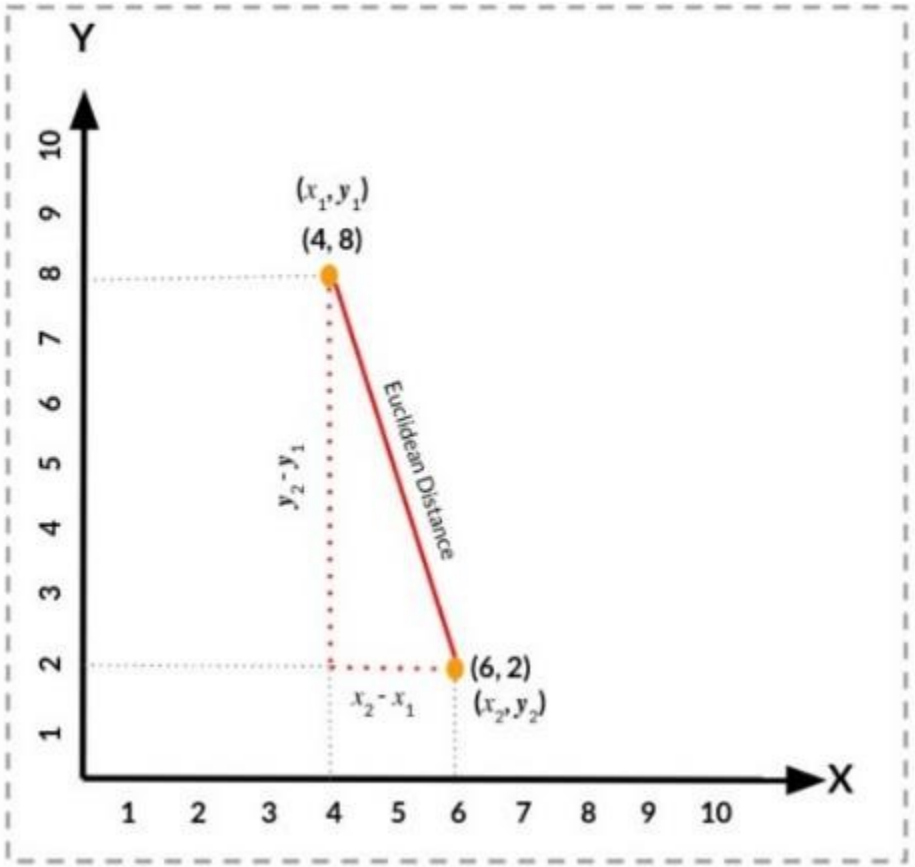
good for spatial data

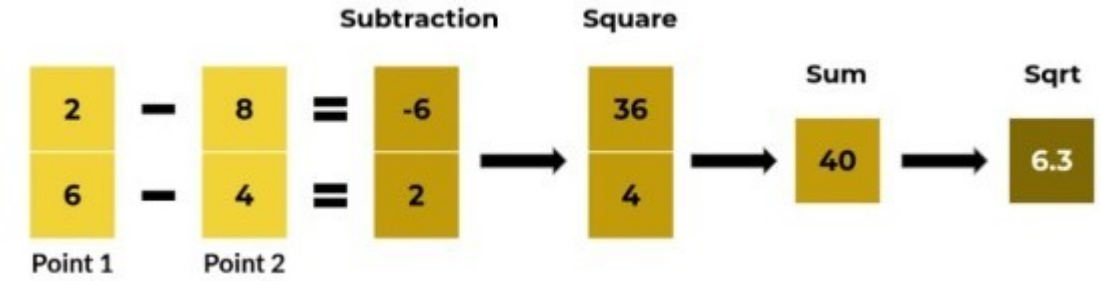


source : Role of Distance Metrics in Machine Learning

Distance measures: Euclidean

Example case:

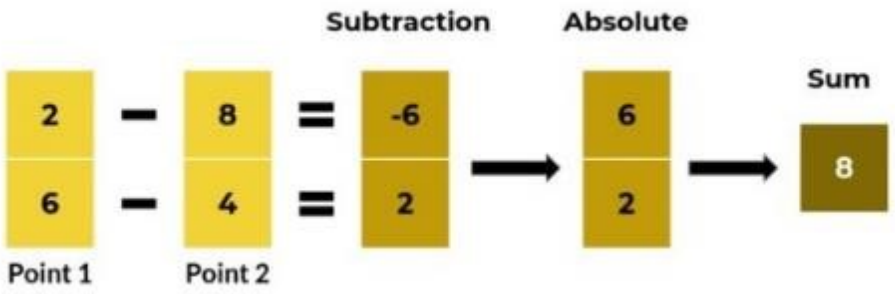


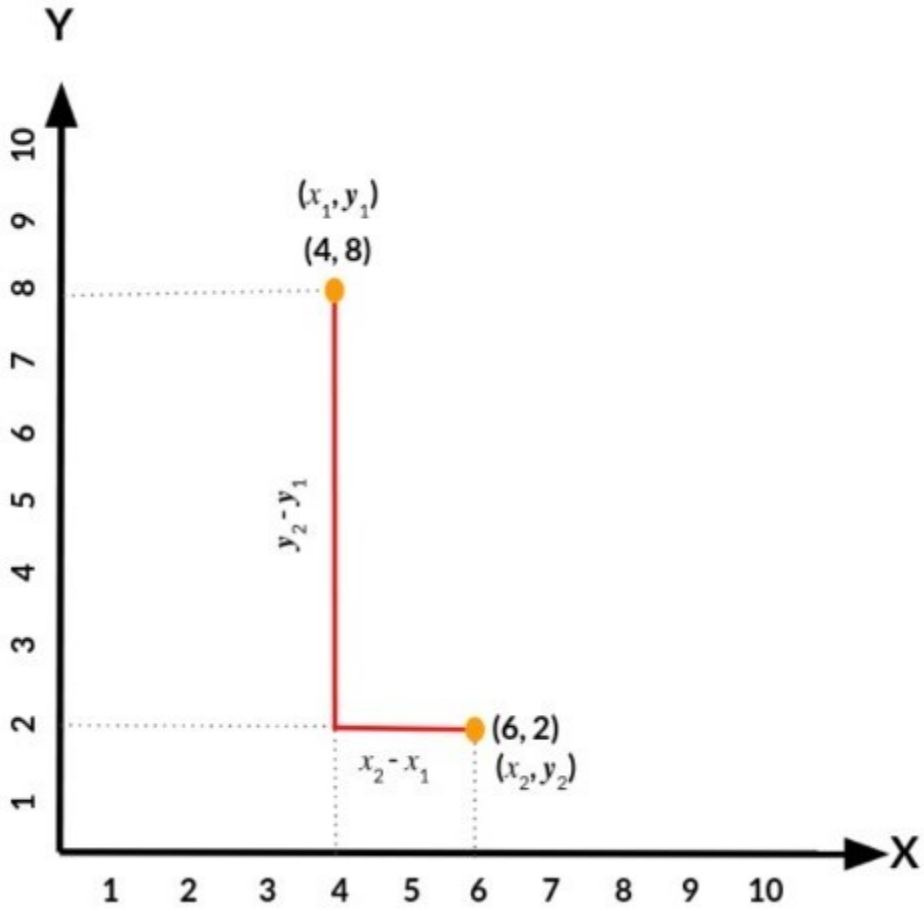


Distance measures: Manhattan

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

$$d(x, y) = \sum_{i=1}^{k} |x_i - y_i|$$





Distance measures: Minkowski

Minkowski Distance is the generalized form of Euclidean and Manhattan Distance

Mathematically, we can write this formula as

$$d(x,y) = \left(\sum_{i=1}^{k} (|x_i - y_i|)^q \right)^{1/q}$$

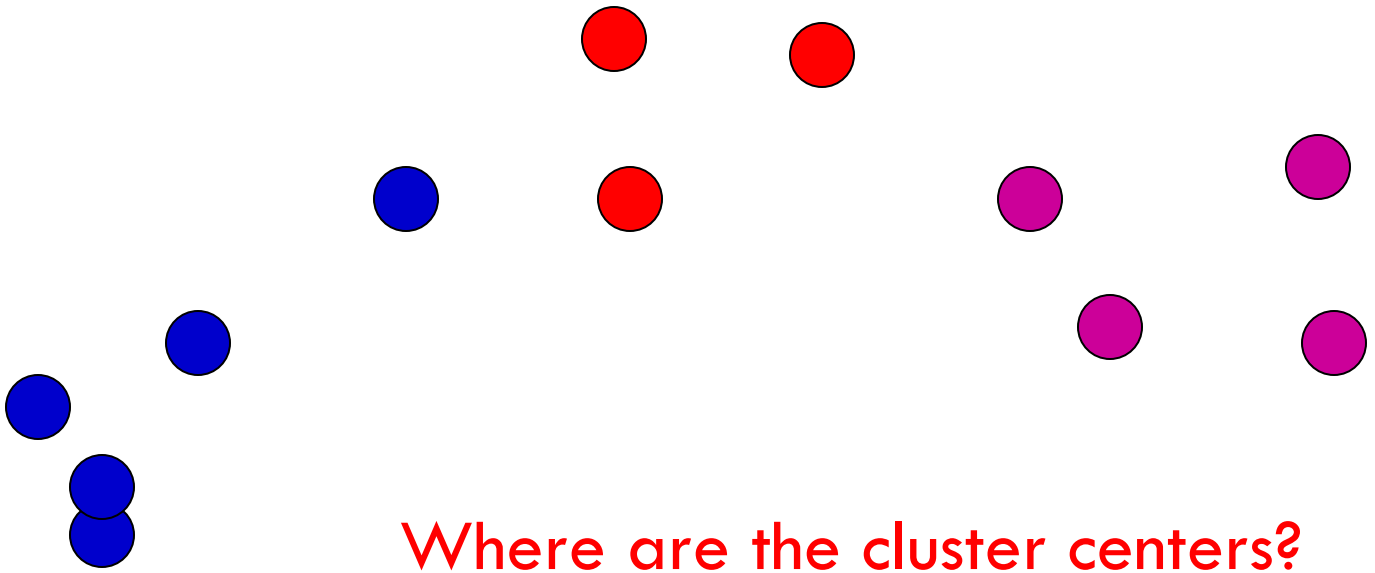
Minkowski distance can work like Manhattan or Euclidean distance.

The selected P value will determine how the Minkowski distance works
- q = 1: Manhattan distance
- q = 2: Euclidean distance

K-means

Iterate:

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

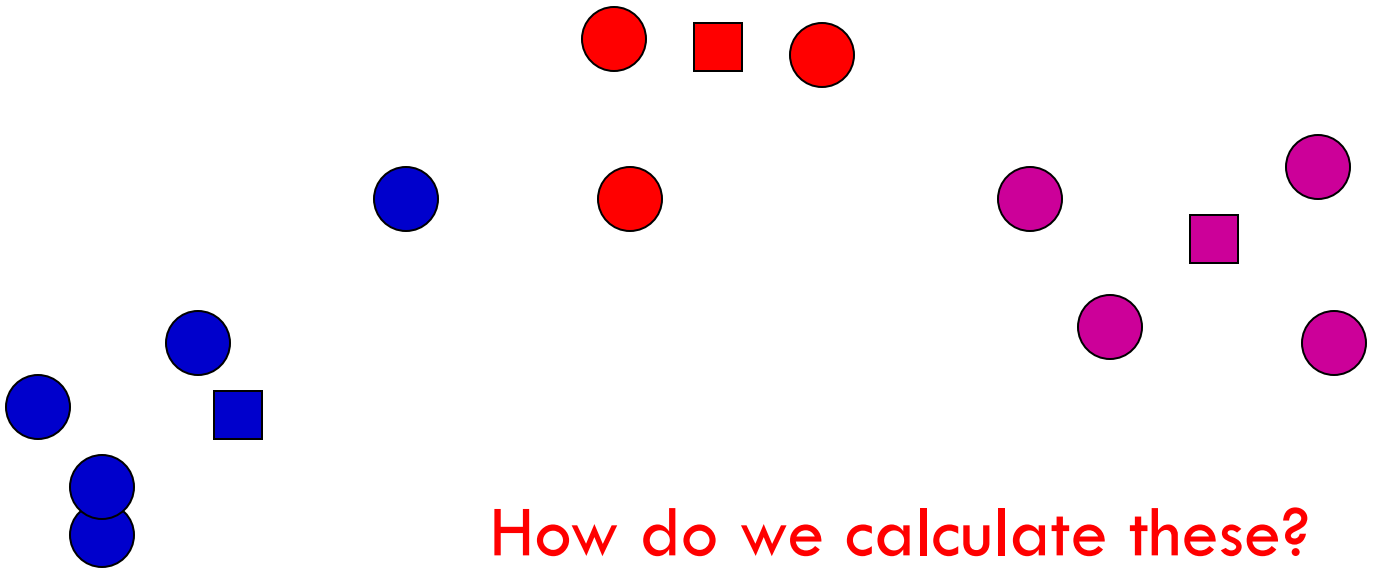


Where are the cluster centers?

K-means

Iterate:

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



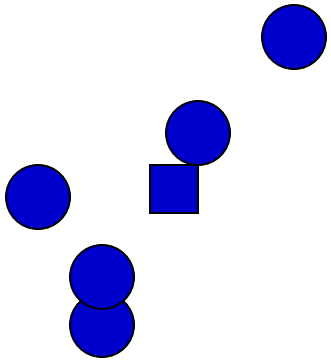
How do we calculate these?

K-means

Iterate:

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

Mean of the points in the cluster:



$$\mu(C) = \frac{1}{|C|} \sum_{x \in C} x$$

where:

$$x + y = \sum_{i=1}^{n} x_i + y_i \qquad \frac{x}{|C|} = \sum_{i=1}^{n} \frac{x_i}{|C|}$$

K-means loss function

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

$$loss = \sum_{i=1}^{n} d(x_i, \mu_k)^2 \quad \text{where } \mu_k \text{ is cluster center for } x_i$$

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

Minimizing k-means loss

Iterate:

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

---

$$loss = \sum_{i=1}^{n} d(x_i, \mu_k)^2 \quad \text{where } \mu_k \text{ is cluster center for } x_i$$

Does each step of k-means move towards reducing this loss function (or at least not increasing)?

Minimizing k-means loss

Iterate:

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

---

$$loss = \sum_{i=1}^{n} d(x_i, \mu_k)^2 \quad \text{where } \mu_k \text{ is cluster center for } x_i$$

This isn't quite a complete proof/argument, but:

1. Any other assignment would end up in a larger loss

1. The mean of a set of values minimizes the squared error

Minimizing k-means loss

Iterate:

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

---

$$loss = \sum_{i=1}^{n} d(x_i, \mu_k)^2 \quad \text{where } \mu_k \text{ is cluster center for } x_i$$

Does this mean that k-means will always find the minimum loss/clustering?

Minimizing k-means loss

Iterate:

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

---

$$loss = \sum_{i=1}^{n} d(x_i, \mu_k)^2 \quad \text{where } \mu_k \text{ is cluster center for } x_i$$

NO! It will find a *minimum*.

Unfortunately, the k-means loss function is generally not convex and for most problems has many, many minima

We're only guaranteed to find one of them

K-means variations/parameters

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

What are some other variations/parameters we haven't specified?

K-means variations/parameters

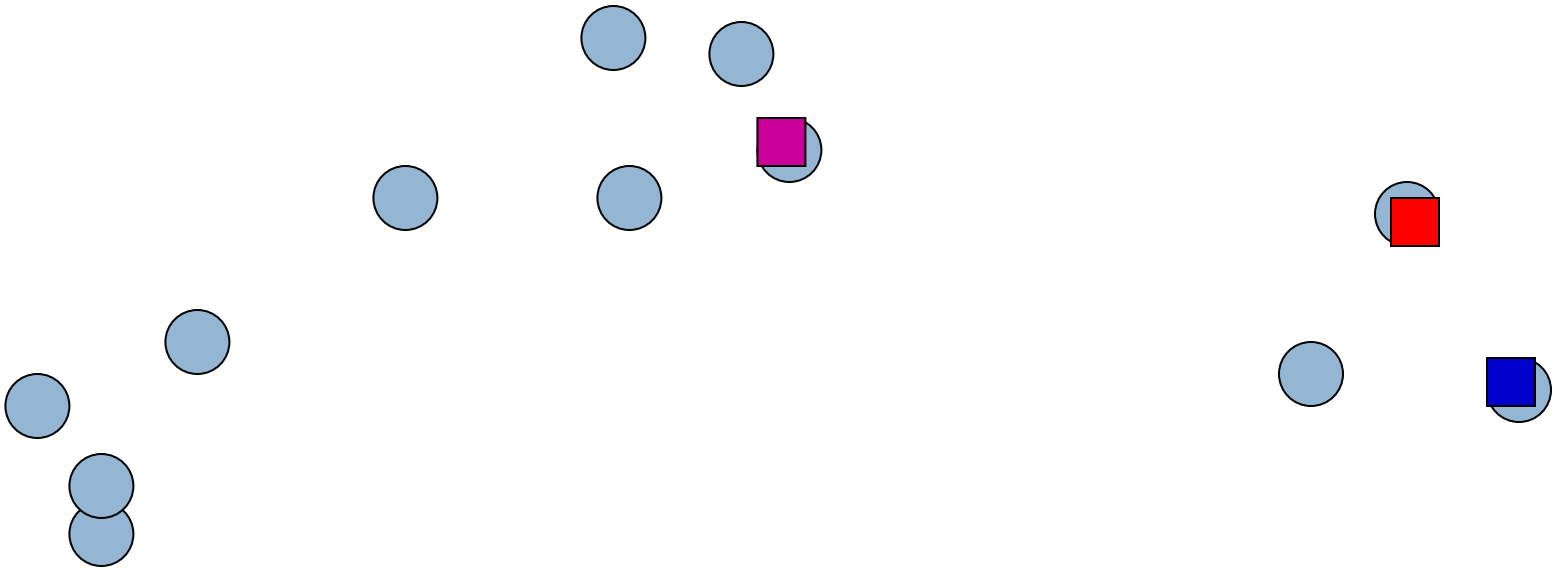
Initial (seed) cluster centers

Convergence

- A fixed number of iterations
- partitions unchanged
- Cluster centers don't change

K!

K-means: Initialize centers randomly



What would happen here?

Seed selection ideas?

Seed choice

Results can vary drastically based on random seed selection

Some seeds can result in poor convergence rate, or convergence to sub-optimal clusterings

Common heuristics

- Random centers in the space
- Randomly pick examples
- Points least similar to any existing center (furthest centers heuristic)
- **Try out multiple starting points**
- Initialize with the results of another clustering method

Furthest centers heuristic

μ_1 = pick random point

for i = 2 to K:

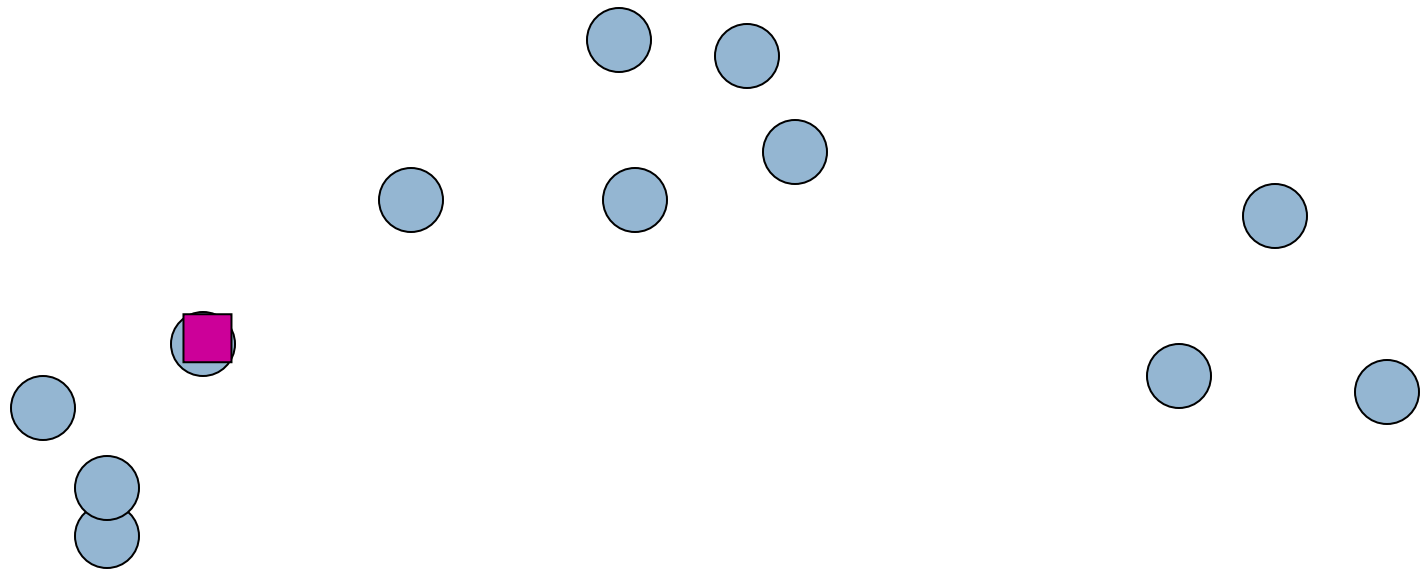
μ_i = point that is furthest from **any** previous centers

$$\mu_i = \underset{x}{\arg\max} \; \min_{\mu_j : 1 < j < i} d(x, \mu_j)$$

point with the largest distance to any previous center

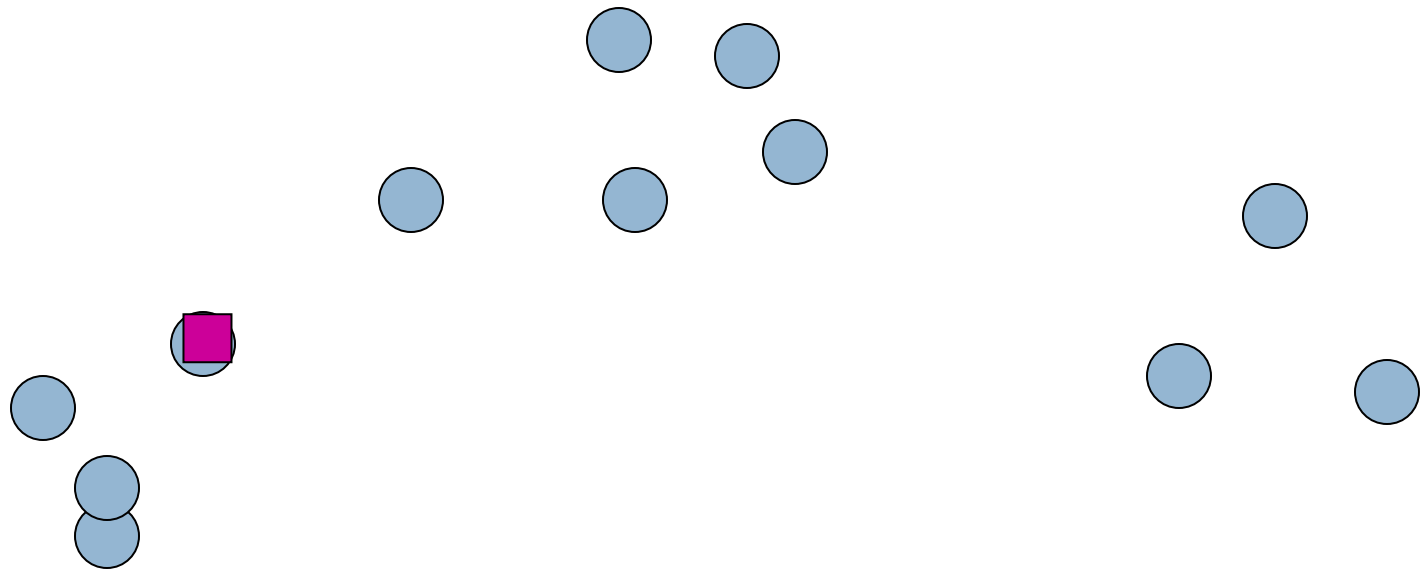
smallest distance from x to any previous center

K-means: Initialize furthest from centers



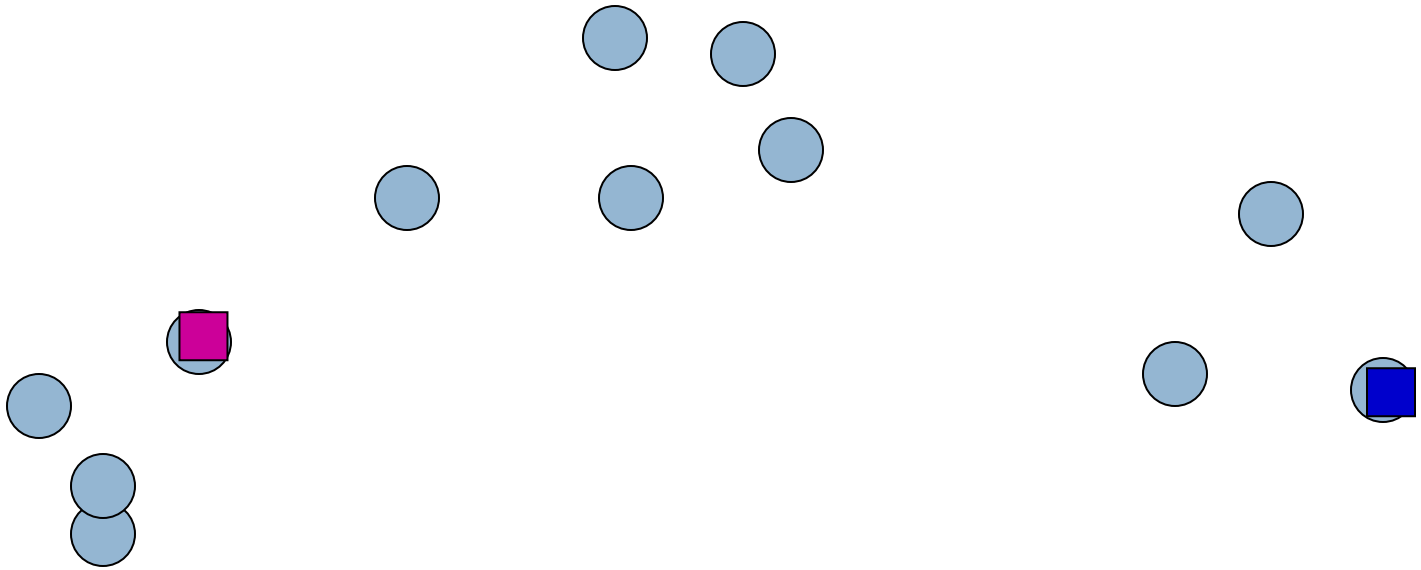
Pick a random point for the first center

K-means: Initialize furthest from centers



What point will be chosen next?

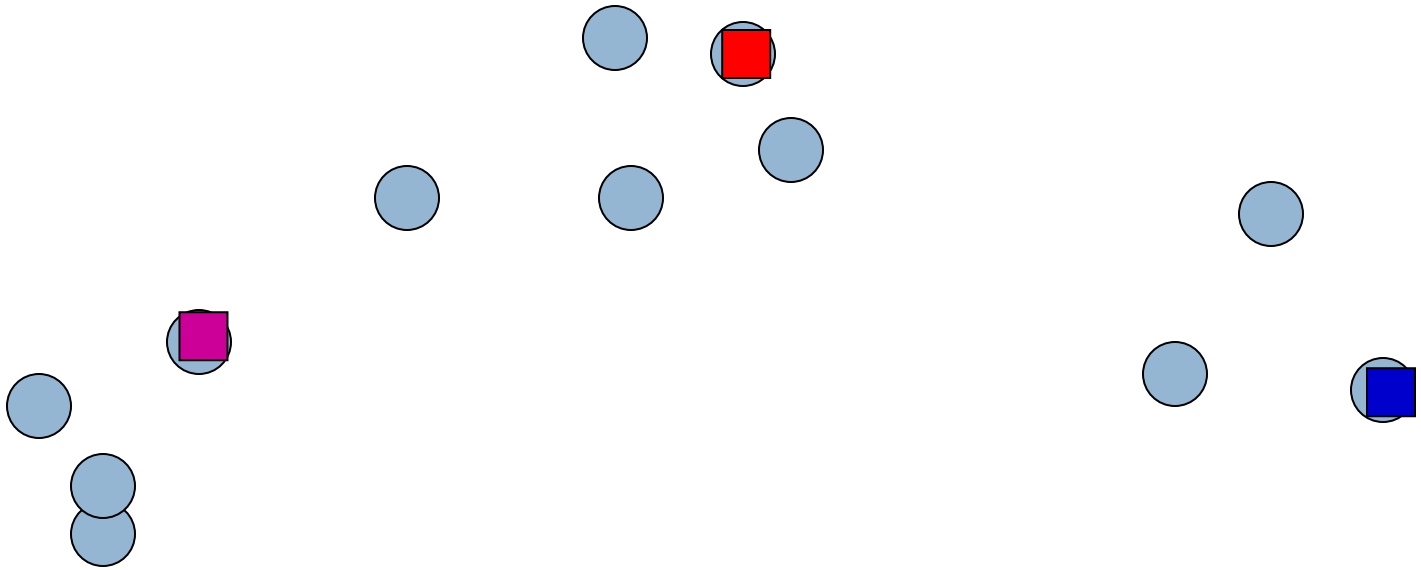
K-means: Initialize furthest from centers



Furthest point from center

What point will be chosen next?

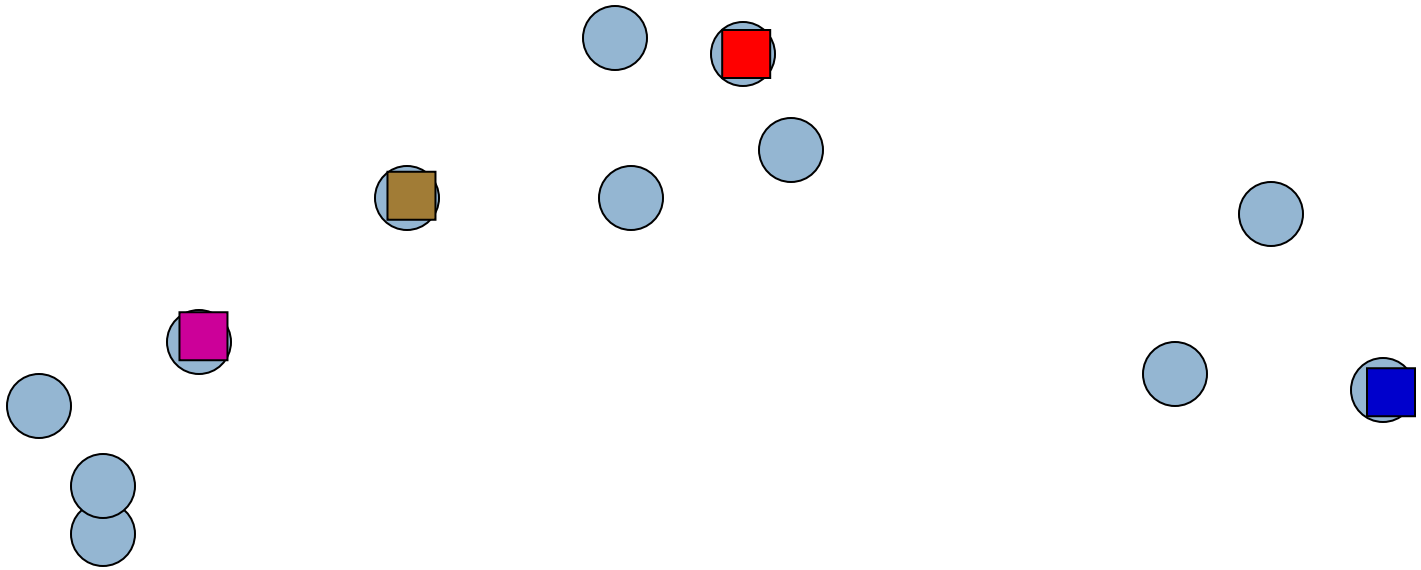
K-means: Initialize furthest from centers



Furthest point from center

What point will be chosen next?

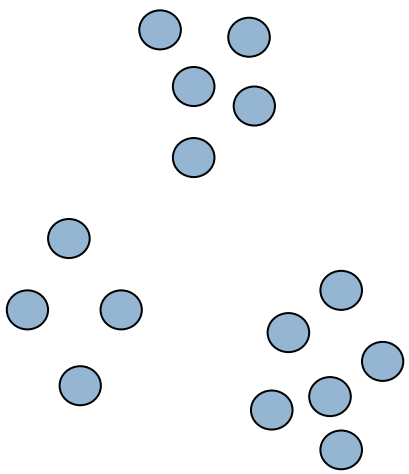
K-means: Initialize furthest from centers



Furthest point from center

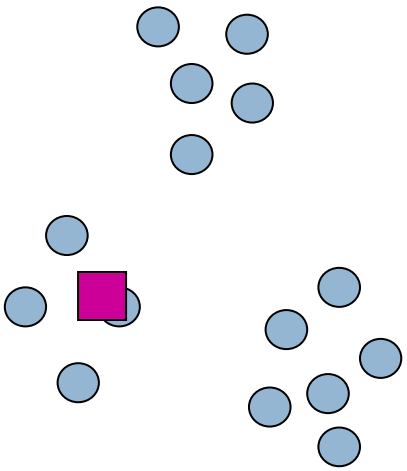
Any issues/concerns with this approach?

Furthest points concerns



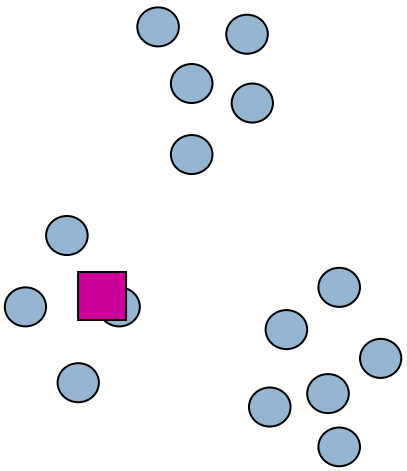
If k = 4, which points will get chosen?

Furthest points concerns



If we do a number of trials, will we get different centers?

Furthest points concerns



Doesn't deal well with outliers

Introduction

- Partitioning Clustering Approach
  - a typical clustering analysis approach via iteratively partitioning training data set to learn a partition of the given data space
  - learning a partition on a data set to produce several non-empty clusters (usually, the number of clusters given in advance)
  - in principle, optimal partition achieved via minimising the sum of squared distance to its "representative object" in each cluster

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

e.g., Euclidean distance $d^2(\mathbf{x}, \mathbf{m}_k) = \sum_{n=1}^{N} (x_{n}^{190} - m_{kn})^2$

Introduction

- Given a *K*, find a partition of *K clusters* to optimise 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 and the algorithm converges to stable centriods of clusters.
  - K-means algorithm is the simplest partitioning method for clustering analysis and widely used in data mining applications.

K-means Algorithm

- Given the cluster number *K*, the *K-means* algorithm is carried out in three steps after initialisation:

Initialisation: set seed points (randomly)

1) Assign each object to the cluster of the nearest seed point measured with a specific distance metric
2) Compute new seed points as the centroids of the clusters of the current partition (the centroid is the centre, i.e., *mean point*, of the cluster)
3) Go back to Step 1), stop when no more new assignment (i.e., membership in each cluster no longer changes)

Example (k=2)

Individual	Variable 1	Variable 2
1	1.0	1.0
2	1.5	2.0
3	3.0	4.0
4	5.0	7.0
5	3.5	5.0
6	4.5	5.0
7	3.5	4.5

Step 1:

Individual	Variable 1	Variable 2
1	1.0	1.0
2	1.5	2.0
3	3.0	4.0
4	5.0	7.0
5	3.5	5.0
6	4.5	5.0
7	3.5	4.5

	Individual	Mean Vector
Group 1	1	(1.0, 1.0)
Group 2	4	(5.0, 7.0)

Initialization: Randomly we choose following two centroids (k=2) for two clusters.

In this case the 2 centroid are:
m1=(1.0,1.0) and
m2=(5.0,7.0).

Step 2:

Thus, we obtain two clusters containing:

{1,2,3} and {4,5,6,7}.

Their new centroids are:

$$m_1 = \left(\frac{1}{3}(1.0+1.5+3.0), \frac{1}{3}(1.0+2.0+4.0)\right) = (1.83, 2.33)$$

$$m_2 = \left(\frac{1}{4}(5.0+3.5+4.5+3.5), \frac{1}{4}(7.0+5.0+5.0+4.5)\right)$$

$$= (4.12, 5.38)$$

Individual	Centroid 1	Centroid 2
1	0	7.21
2 (1.5, 2.0)	1.12	6.10
3	3.61	3.61
4	7.21	0
5	4.72	2.5
6	5.31	2.06
7	4.30	2.92

$$d(m_1, 2) = \sqrt{|1.0-1.5|^2 + |1.0-2.0|^2} = 1.12$$

$$d(m_2, 2) = \sqrt{|5.0-1.5|^2 + |7.0-2.0|^2} = 6.10$$

Step 3:

- Now using these centroids we compute the Euclidean distance of each object, as shown in table.

Individual	Centroid 1	Centroid 2
1	1.57	5.38
2	0.47	4.28
3	2.04	1.78
4	5.64	1.84
5	3.15	0.73
6	3.78	0.54
7	2.74	1.08

- Therefore, the new clusters are:

  {1,2} and {**3**,4,5,6,7}

- Next centroids are: m1=(1.25,1.5) and m2 = (3.9,5.1)

Step 4:

The clusters obtained are:

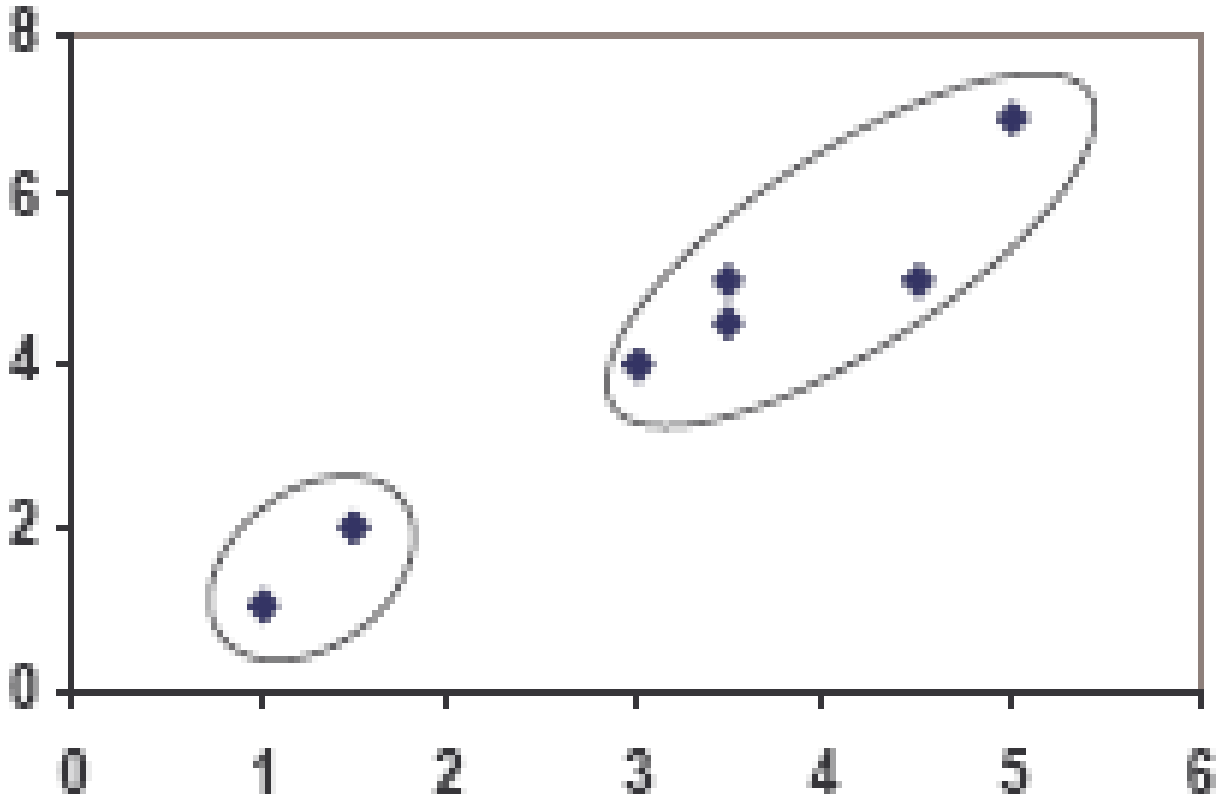
{1,2} and {3,4,5,6,7}

Therefore, there is no change in the cluster.

Thus, the algorithm comes to a halt here and final result consist of 2 clusters {1,2} and {3,4,5,6,7}.

Individual	Centroid 1	Centroid 2
1	0.56	5.02
2	0. 56	3.92
3	3.05	1.42
4	6.86	2.20
5	4.16	0.41
6	4.78	0.61
7	3.75	0.72

PLOT



With K = 3

Individual	$m_1 = 1$	$m_2 = 2$	$m_3 = 3$	cluster
1	0	1.11	3.61	1
2	1.12	0	2.5	2
3	3.61	2.5	0	3
4	7.21	6.10	3.61	3
5	4.72	3.61	1.12	3
6	5.31	4.24	1.80	3
7	4.30	3.20	0.71	3

C_3

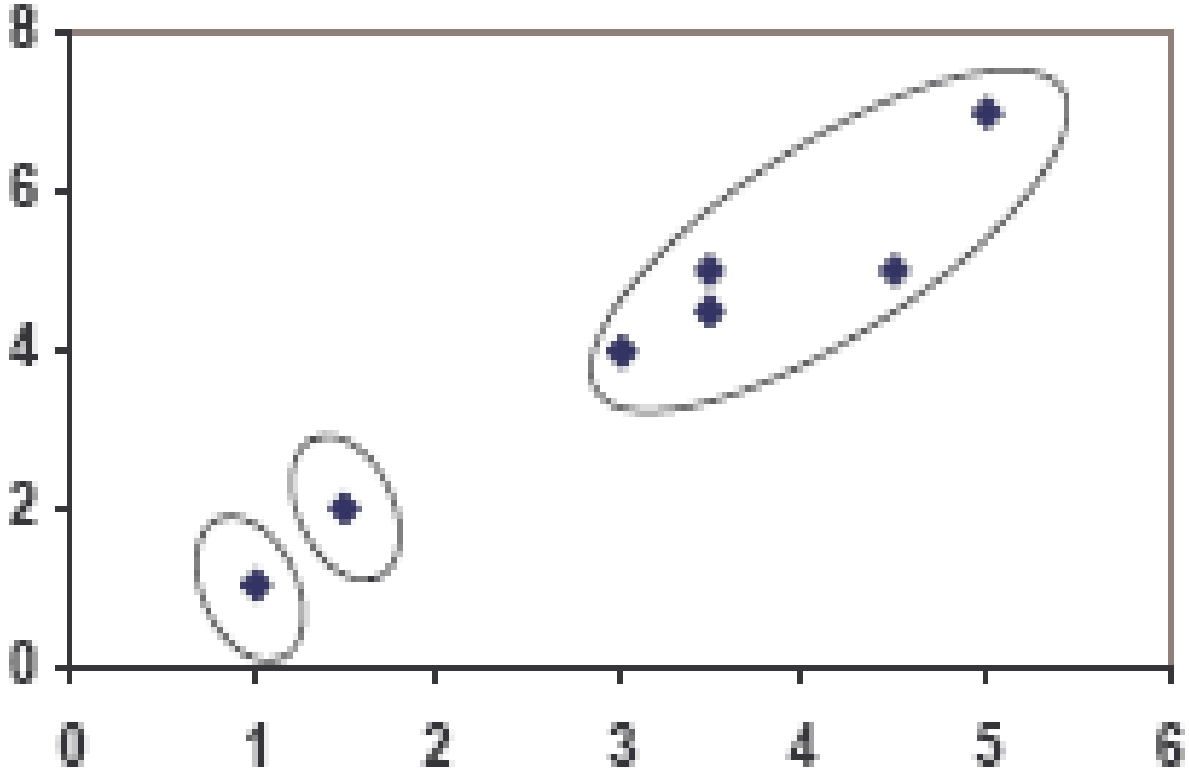
clustering with initial centroids (1, 2, 3)

Step 1

Individual	m_1 (1.0, 1.0)	m_2 (1.5, 2.0)	m_3 (3.9,5.1)	cluster
1	0	1.11	5.02	1
2	1.12	0	3.92	2
3	3.61	2.5	1.42	3
4	7.21	6.10	2.20	3
5	4.72	3.61	0.41	3
6	5.31	4.24	0.61	3
7	4.30	3.20	0.72	3

Step 2

PLOT

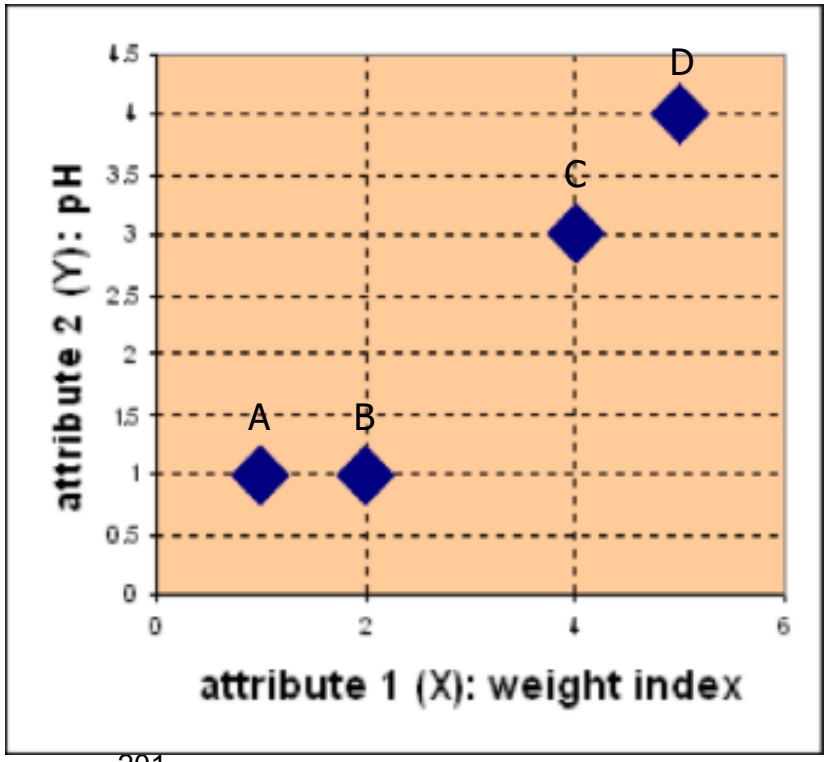


Real-Life Numerical Example of K-Means Clustering

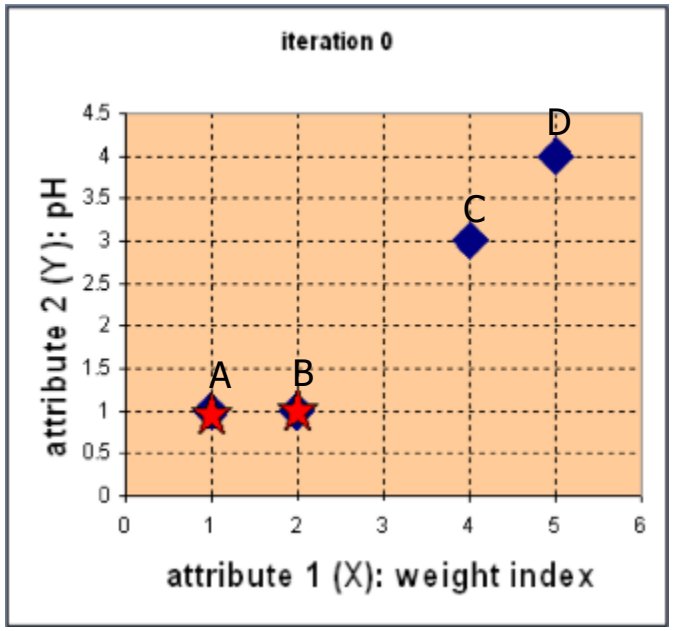
- Problem

Suppose we have 4 types of medicines and each has two attributes (pH and weight index). Our goal is to group these objects into *K=2* group of medicine.

Object	Attribute1 (X): weight index	Attribute 2 (Y): pH
Medicine A	1	1
Medicine B	2	1
Medicine C	4	3
Medicine D	5	4



STEP 1:



$$c_1 = A, c_2 = B$$

$$D^0 = \begin{bmatrix} 0 & 1 & 3.61 & 5 \\ 1 & 0 & 2.83 & 4.24 \end{bmatrix} \quad \begin{matrix} c_1 = (1,1) & group-1 \\ c_2 = (2,1) & group-2 \end{matrix}$$

$$\begin{matrix} A & B & C & D \end{matrix}$$

$$\begin{bmatrix} 1 & 2 & 4 & 5 \\ 1 & 1 & 3 & 4 \end{bmatrix} \quad \begin{matrix} X \\ Y \end{matrix}$$

Euclidean distance

$$d(D,c_1) = \sqrt{(5-1)^2 + (4-1)^2} = 5$$

$$d(D,c_2) = \sqrt{(5-2)^2 + (4-1)^2} = 4.24$$

Assign each object to the cluster with the nearest seed point

Initial value of centroids : Suppose we use medicine A and medicine B as the first centroids.

Let and c_1 and c_2 denote the coordinate of the centroids, then c_1=(1,1) and c_2=(2,1)

- **Objects-Centroids distance** : we calculate the distance between cluster centroid to each object. Let us use Euclidean distance, then we have distance matrix at iteration 0 is

$$\mathbf{D}^0 = \begin{bmatrix} 0 & 1 & 3.61 & 5 \\ 1 & 0 & 2.83 & 4.24 \end{bmatrix} \quad \begin{matrix} \mathbf{c}_1 = (1,1) & group-1 \\ \mathbf{c}_2 = (2,1) & group-2 \end{matrix}$$

$$\begin{matrix} A & B & C & D \end{matrix}$$

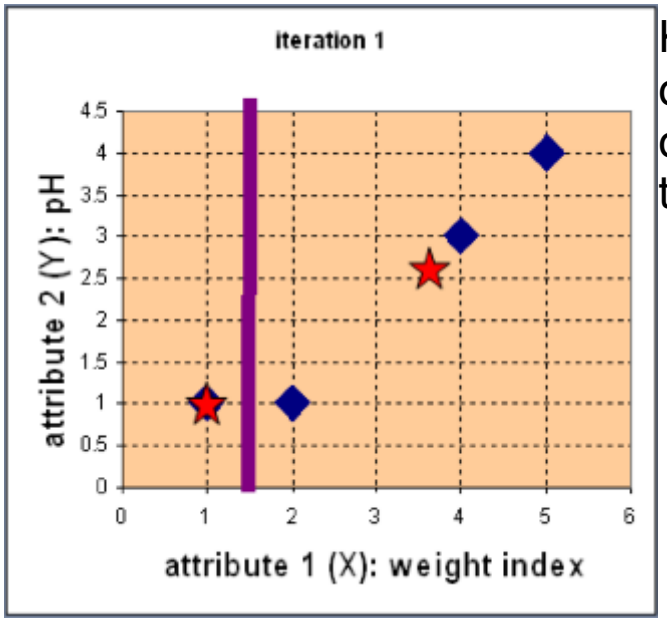
$$\begin{bmatrix} 1 & 2 & 4 & 5 \\ 1 & 1 & 3 & 4 \end{bmatrix} \quad \begin{matrix} X \\ Y \end{matrix}$$

- Each column in the distance matrix symbolizes the object.
- The first row of the distance matrix corresponds to the distance of each object to the first centroid and the second row is the distance of each object to the second centroid.

- For example, distance from medicine C = (4, 3) to the first centroid $\mathbf{c}_1 = (1,1)$ is , $\sqrt{(4-1)^2+(3-1)^2} = 3.61$ and its distance to the second centroid is , $\mathbf{c}_2 = (2,1)$ is $\sqrt{(4-2)^2+(3-1)^2} = 2.83$ etc.

STEP 2:
Compute new centroids of the current partition

- **Objects clustering** : We assign each object based on the minimum distance.
- Medicine A is assigned to group 1, medicine B to group 2, medicine C to group 2 and medicine D to group 2.
- The elements of Group matrix below is 1 if and only if the object is assigned to that group.



Knowing the members of each cluster, now we compute the new centroid of each group based on these new memberships.

$$c_1 = (1,\ 1)$$

$$c_2 = \left(\frac{2+4+5}{3},\ \frac{1+3+4}{3}\right)$$

$$= \left(\frac{11}{3},\ \frac{8}{3}\right)$$

- **<u>Iteration-1, Objects-Centroids distances</u>** : The next step is to compute the distance of all objects to the new centroids.
- Similar to step 2, we have distance matrix at iteration 1 is

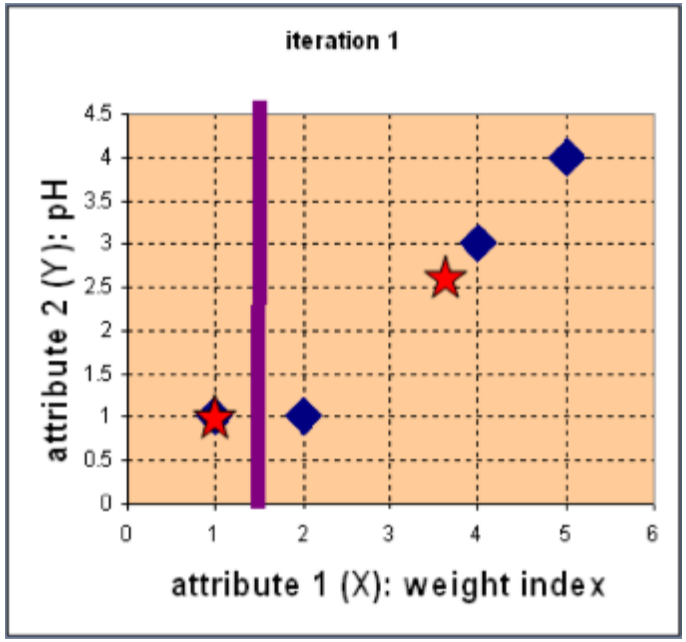
$$\mathbf{D}^1 = \begin{bmatrix} 0 & 1 & 3.61 & 5 \\ 3.14 & 2.36 & 0.47 & 1.89 \end{bmatrix} \quad \begin{matrix} \mathbf{c}_1 = (1,1) & group-1 \\ \mathbf{c}_2 = (\frac{11}{3}, \frac{8}{3}) & group-2 \end{matrix}$$

$$\begin{matrix} A & B & C & D \end{matrix}$$

$$\begin{bmatrix} 1 & 2 & 4 & 5 \\ 1 & 1 & 3 & 4 \end{bmatrix} \begin{matrix} X \\ Y \end{matrix}$$

STEP 2:

Renew membership based on new centroids



Compute the distance of all objects to the new centroids

$$D^1 = \begin{bmatrix} 0 & 1 & 3.61 & 5 \\ 3.14 & 2.36 & 0.47 & 1.89 \end{bmatrix} \quad \begin{matrix} c_1 = (1,1) & group-1 \\ c_2 = (\frac{11}{3}, \frac{8}{3}) & group-2 \end{matrix}$$

$$\begin{matrix} A & B & C & D \end{matrix}$$

$$\begin{bmatrix} 1 & 2 & 4 & 5 \\ 1 & 1 & 3 & 4 \end{bmatrix} \quad \begin{matrix} X \\ Y \end{matrix}$$

Assign the membership to objects

STEP 3:

Repeat the first two steps until its convergence

- **Iteration-1, Objects clustering:** Based on the new distance matrix, we move the medicine B to Group 1 while all the other objects remain. The Group matrix is shown below

$$G^1 = \begin{bmatrix} 1 & 1 & 0 & 0 \\ 0 & 0 & 1 & 1 \end{bmatrix} \quad \begin{matrix} group-1 \\ group-2 \end{matrix}$$

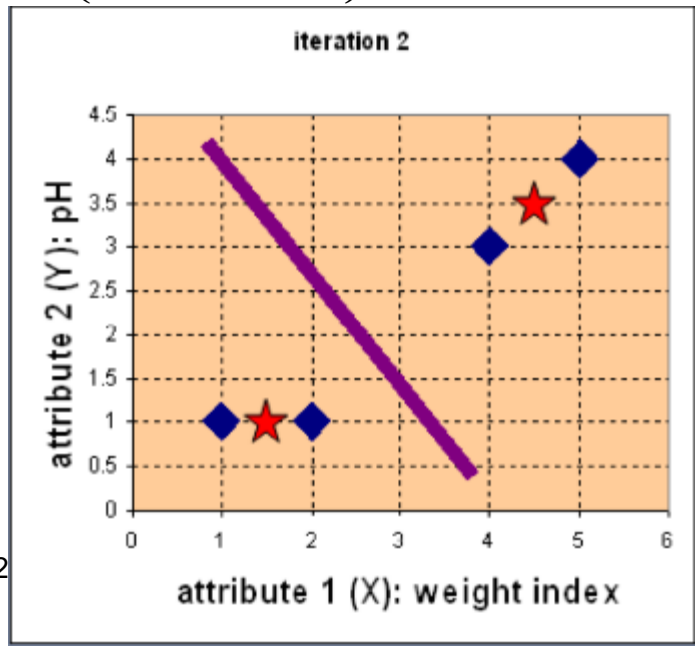
$$A \quad B \quad C \quad D$$

- **Iteration 2, determine centroids:** Now we repeat step 4 to calculate the new centroids coordinate based on the clustering of previous iteration. Group1 and group 2 both has two members, thus the new centroids are $c_1 = (\frac{1+2}{2}, \frac{1+1}{2}) = (1\frac{1}{2}, 1)$ and $c_2 = (\frac{4+5}{2}, \frac{3+4}{2}) = (4\frac{1}{2}, 3\frac{1}{2})$

Knowing the members of each cluster, now we compute the new centroid of each group based on these new memberships.

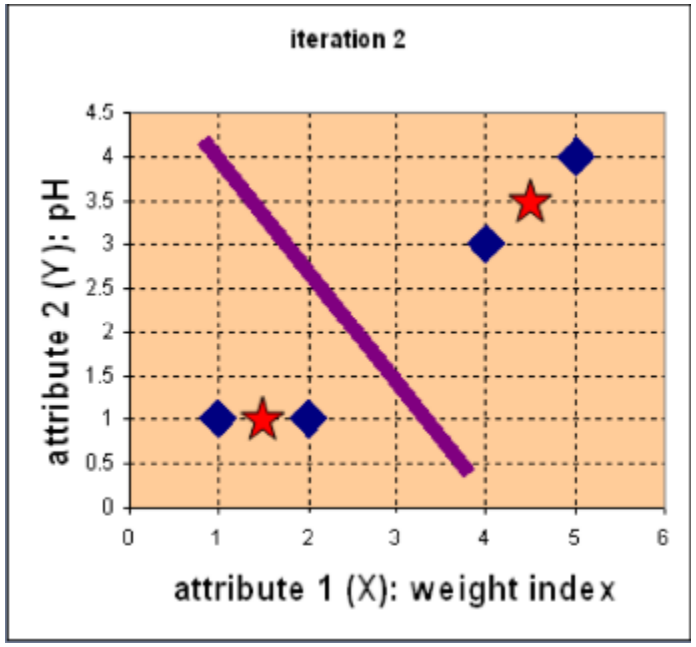
$$c_1 = \left(\frac{1+2}{2}, \frac{1+1}{2}\right) = (1\frac{1}{2}, 1)$$

$$c_2 = \left(\frac{4+5}{2}, \frac{3+4}{2}\right) = (4\frac{1}{2}, 3\frac{1}{2})$$



STEP 3:

Repeat the first two steps until its convergence



Compute the distance of all objects to the new centroids

$$D^2 = \begin{bmatrix} 0.5 & 0.5 & 3.20 & 4.61 \\ 4.30 & 3.54 & 0.71 & 0.71 \end{bmatrix} \quad \begin{matrix} c_1 = (1\frac{1}{2}, 1) & group-1 \\ c_2 = (4\frac{1}{2}, 3\frac{1}{2}) & group-2 \end{matrix}$$

$$\begin{matrix} A & B & C & D \end{matrix}$$

$$\begin{bmatrix} 1 & 2 & 4 & 5 \\ 1 & 1 & 3 & 4 \end{bmatrix} \begin{matrix} X \\ Y \end{matrix}$$

Stop due to no new assignment
Membership in each cluster no longer change

- **Iteration-2, Objects clustering:** Again, we assign each object based on the minimum distance.

$$\mathbf{G}^2 = \begin{bmatrix} 1 & 1 & 0 & 0 \\ 0 & 0 & 1 & 1 \end{bmatrix} \quad \begin{matrix} group-1 \\ group-2 \end{matrix}$$

$$A \quad B \quad C \quad D$$

- We obtain result that $\mathbf{G}^2 = \mathbf{G}^1$. Comparing the grouping of last iteration and this iteration reveals that the objects does not move group anymore.
- Thus, the computation of the k-mean clustering has reached its stability and no more iteration is needed..

FINAL Result

Object	Feature1(X): weight index	Feature2 (Y): pH	Group (result)
Medicine A	1	1	1
Medicine B	2	1	1
Medicine C	4	3	2
Medicine D	5	4	2

Weaknesses of K-Mean Clustering

1. When the numbers of data are not so many, initial grouping will determine the cluster significantly.
2. The number of cluster, K, must be determined before hand. Its disadvantage is that it does not yield the same result with each run, since the resulting clusters depend on the initial random assignments.
3. We never know the real cluster, using the same data, because if it is inputted in a different order it may produce different cluster if the number of data is few.
4. It is sensitive to initial condition. Different initial condition may produce different result of cluster. The algorithm may be trapped in the *local optimum*.

Relevant Issues

- Computational complexity
  - 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
- 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)
  - how to evaluate the *K*-mean performance?

Latihan:
Single/Complete Linkage dan K-Mean

X1 = Daya tahan asam (detik)	X2 = Kekuatan (Kg/m^2)
8	4
4	5
4	6
7	7
5	6
6	5