Inspire…Educate…Transform.

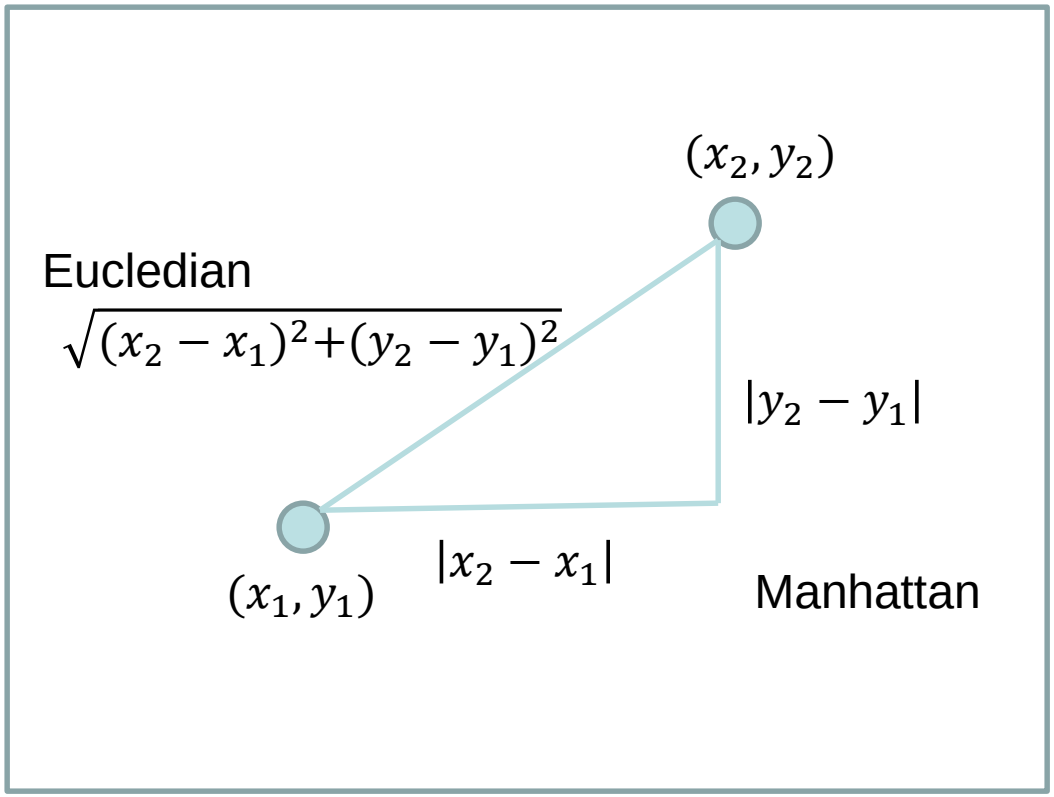
Unsupervised models

Clustering

Lt Suryaprakash Kompalli
Senior Mentor, INSOFE

UNDERSTANDING DISTANCE

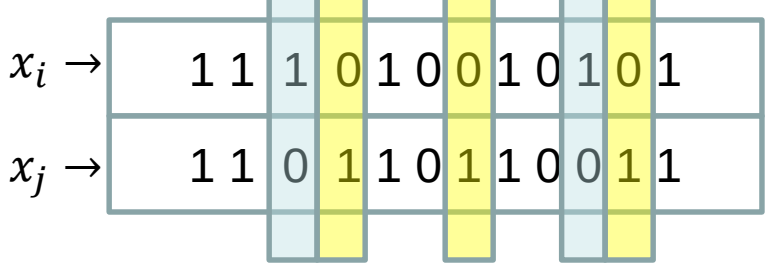
Numeric



Distance between categorical attributes

- For each categorical variable
  - Distance is zero if two records have same value
  - 1 if different
  - Sum the total distance across all attributes

Categorical attributes



For example, in above two feature vectors:

a=5
b=2
c=3
d=2

(10)

Data point i \ Data point j	1	0	
1	a	b	$a+b$
0	c	d	$c+d$
	$a+c$	$b+d$	$a+b+c+d$

a: the number of attributes with the value of 1 for both data points.

b: the number of attributes for which $x_{if} = 1$ and $x_{jf} = 0$, where x_{if} (x_{jf}) is the value of the fth attribute of the data point $\mathbf{x}_i$ ($\mathbf{x}_j$).

c: the number of attributes for which $x_{if} = 0$ and $x_{jf} = 1$.

d: the number of attributes with the value of 0 for both data points.

Symmetric binary attributes

- Hamming distance function: Simple Matching Coefficient, proportion of mismatches of their values

$$dist(\mathbf{x}_i, \mathbf{x}_j) = \frac{b + c}{a + b + c + d}$$

$$Dist = \frac{number\ of\ dissimiar\ attributes\ between\ the\ records}{number\ of\ dissimilar\ attributes + number\ of\ similar\ attributes}$$

Asymmetric binary attributes

- Asymmetric: if one of the states is more important or more valuable than the other.
  - By convention, state 1 represents the more important state, which is typically the rare or infrequent state.
  - Jaccard coefficient is a popular measure

$$Dist = \frac{number\ of\ dissimiar\ attributes\ between\ the\ records}{number\ of\ dissimilar\ attributes + number\ of\ similar\ attributes\ (excluding\ records\ with\ 0,0)}$$

$$dist(\mathbf{x}_i, \mathbf{x}_j) = \frac{b + c}{a + b + c}$$

  - We can have some variations, adding weights

Dissimilarity between Binary Variables

- Example

Name	Gender	Fever	Cough	Test-1	Test-2	Test-3	Test-4
Jack	M	Y	N	P	N	N	N
Mary	F	Y	N	P	N	P	N
Jim	M	Y	P	N	N	N	N

  - gender is a symmetric attribute
  - the remaining attributes are asymmetric binary
  - let the values Y and P be set to 1, and the value N be set to 0

$$d(jack, mary) = \frac{0+1}{2+0+1} = 0.33$$

$$d(jack, jim) = \frac{1+1}{1+1+1} = 0.67$$

$$d(jim, mary) = \frac{1+2}{1+1+2} = 0.75$$

Another distance metric used in supervised learning

Value difference measure (VDM):d_{ij}

$$\sum_{h=1}^{All\ classes} |P(h|val_i) - P(h|val_j)|$$

Value Distance Measure

ID	Age	Income	Family	CCAvg	Personal Loan
1	Young	Low	4	Low	0
2	Old	Low	3	Low	0
3	Middle	Low	1	Low	0
4	Middle	Medium	1	Low	0
5	Middle	Low	4	Low	0
6	Middle	Low	4	Low	0
10	Middle	High	1	High	1
17	Middle	Medium	4	Medium	1
19	Old	High	2	High	1
30	Middle	Medium	1	Medium	1
39	Old	Medium	3	Medium	1
43	Young	Medium	4	Low	1
48	Middle	High	4	Low	1

$$VDM_{family1,family2}$$

$$|P(0|f_1) - P(0|f_2)| + |P(1|f_1) - P(1|f_2)|$$

$$|0.5 - 0| + |0.5 - 1|$$

$$= 1$$

$$VDM_{family1,family3} = |P(0|f_1) - P(0|f_3)| + |P(1|f_1) - P(1|f_3)| = |0.5 - 0.5| + |0.5 - 0.5| = 0$$

Ordinal variables

- Same as numeric
- Look up is better than computation

Look up matrix for ordinal with 3 states

$$\begin{bmatrix} & 1 & 2 & 3 \\ 1 & 0 & 1 & 2 \\ 2 & 1 & 0 & 1 \\ 3 & 2 & 1 & 0 \end{bmatrix} \qquad \begin{bmatrix} & 1 & 2 & 3 \\ 1 & 0 & 1 & 3 \\ 2 & 1 & 0 & 2 \\ 3 & 3 & 2 & 0 \end{bmatrix}$$

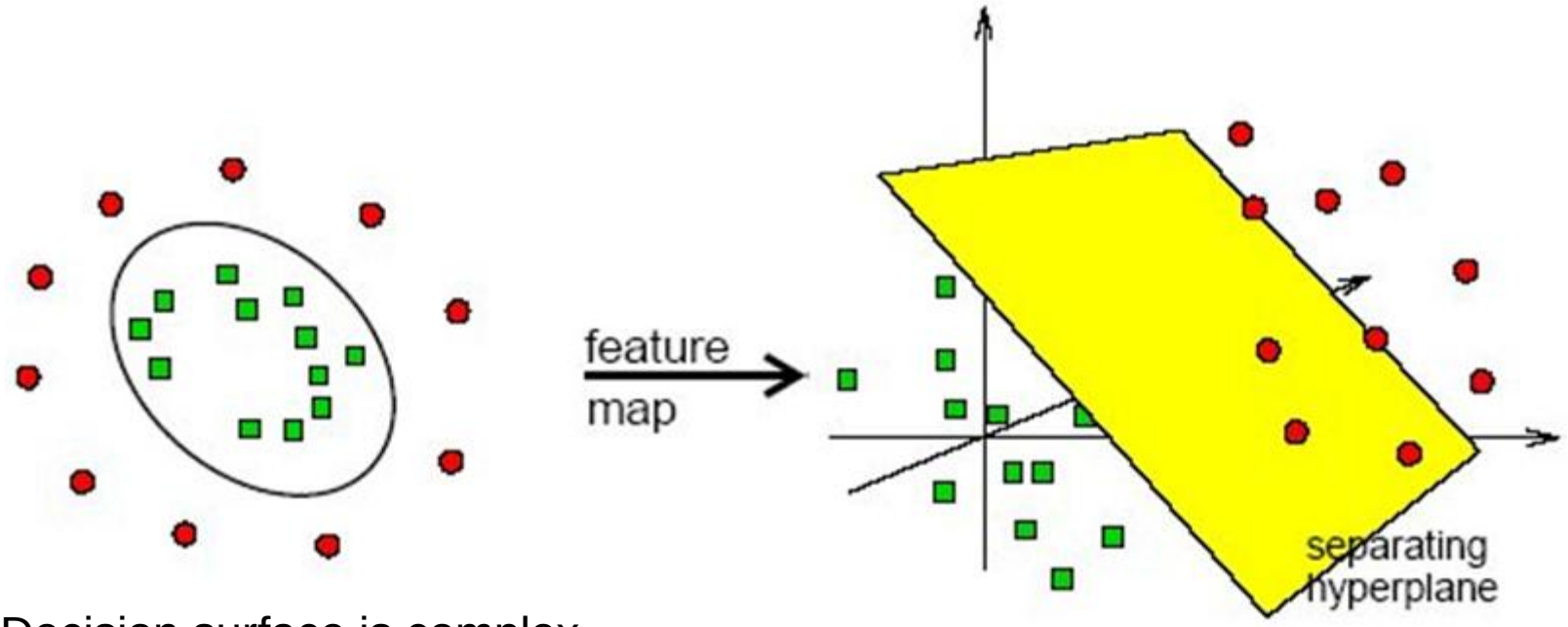
Distance between different ordinals may not be same !!! On the right, distance between state 1 and state 2 is much less than distance between state 1 and state 3.

Depends on what the ordinals represent.

E.g.: If ordinals represent education: class 5, Class 10, Degree. Distance between class 5 and class 10 may be considered less than distance between class 10 and degree

KERNEL TRICK

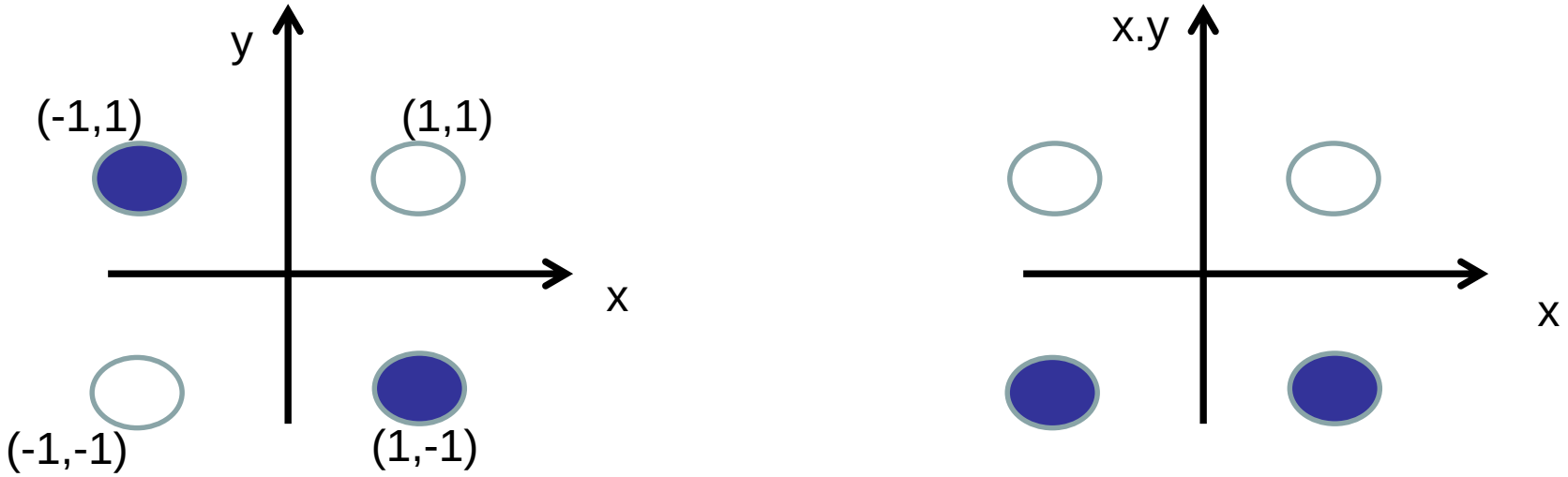
Moving into higher dimensions



Decision surface is complex in lower dimension

Decision surface is simple in higher dimension

Linear Separation of XOR



- XOR is not linearly separable in x,y space
- Linearly separable in x.y space
  - The kernel here is K(x,y) = x.y

Reference: http://www.doc.ic.ac.uk/~dfg/ProbabilisticInference/IDAPILecture18.pdf, Duda and Hart, chapter 3.

Standard Kernels

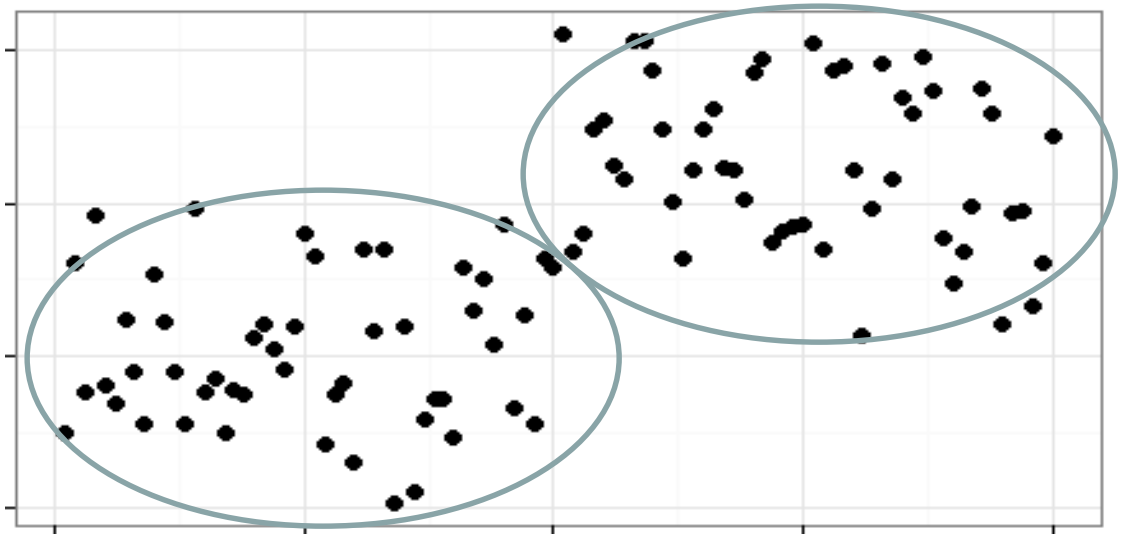
- Polynomial
  - $-(\alpha x^T y + c)^d$
- Radial Basis
  - $-\exp(-\gamma \|x - y\|^2)$

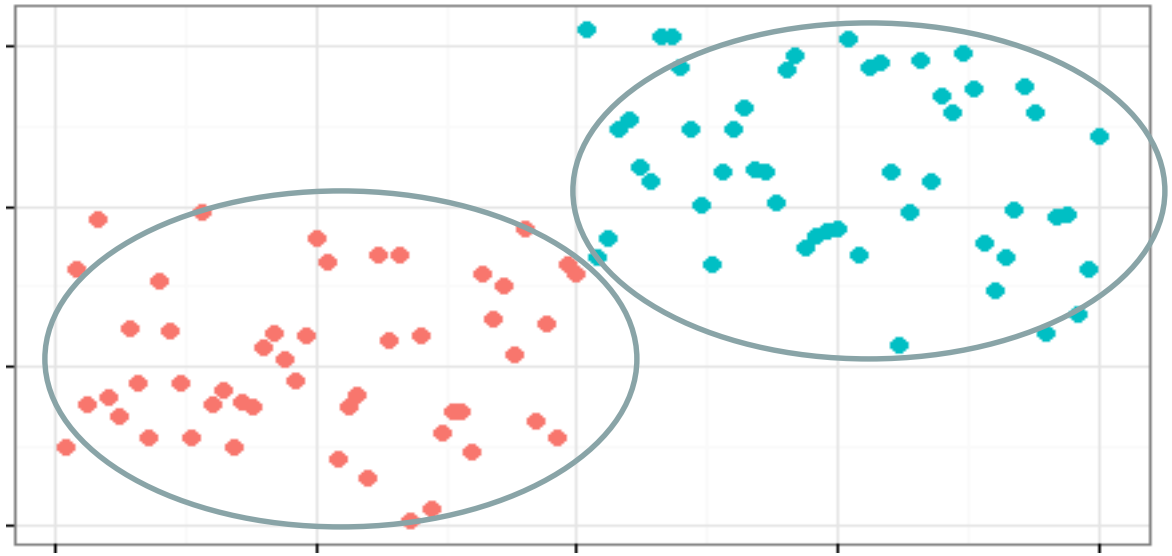
Clustering

Unsupervised learning

- Supervised: Data and target

- Unsupervised: Just data





Clustering

- Finding similarity groups in data, called **clusters**. I.e.,
  - data instances that are similar to (near) each other are in the same cluster
  - data instances that are very different (far away) from each other fall in different clusters.

A few clustering applications

- In marketing, segment customers according to their similarities
  - To do targeted marketing.
  - It is not uncommon to have over 100,000 segments in insurance clustering

Google search

- Given a collection of text documents, organize them according to their content similarities,
  - E.g., Google news
- Blind signal separation (separating two speakers)

Algorithms

- **Hierarchical** approach: Create a hierarchical decomposition of the set of data (or objects) using some criterion (Wald)
- **Partitioning** approach: Construct various partitions and then evaluate them by some criterion, e.g., minimizing the sum of square errors (K-means, Spectral clustering)
- **Model-based** methods: A model is hypothesized for each of the clusters and tries to find the best fit of that model to each other (EM)

HIERARCHICAL (AGGLOMERATIVE) CLUSTERING

Example of agglomerative clustering

	BOS	NY	DC	MIA	CHI	SEA	SF	LA	DEN
BOS	0	206	429	1504	963	2976	3095	2979	1949
NY	206	0	233	1308	802	2815	2934	2786	1771
DC	429	233	0	1075	671	2684	2799	2631	1616
MIA	1504	1308	1075	0	1329	3273	3053	2687	2037
CHI	963	802	671	1329	0	2013	2142	2054	996
SEA	2976	2815	2684	3273	2013	0	808	1131	1307
SF	3095	2934	2799	3053	2142	808	0	379	1235
LA	2979	2786	2631	2687	2054	1131	379	0	1059
DEN	1949	1771	1616	2037	996	1307	1235	1059	0

At each iteration, pick two data points that have least distance between them. Add the points into a cluster.

	BOS/NY	DC	MIA	CHI	SEA	SF	LA	DEN
BOS/NY	0	223	1308	802	2815	2934	2786	1771
DC	223	0	1075	671	2684	2799	2631	1616
MIA	1308	1075	0	1329	3273	3053	2687	2037
CHI	802	671	1329	0	2013	2142	2054	996
SEA	2815	2684	3273	2013	0	808	1131	1307
SF	2934	2799	3053	2142	808	0	379	1235
LA	2786	2631	2687	2054	1131	379	0	1059
DEN	1771	1616	2037	996	1307	1235	1059	0

Note how we update distances between other clusters. The lower distance is picked. Distance between BOS to DC was 429, now set to 233. Distance from BOS to MIA was 1504, now set to 1308.

Averaging may also be used instead of taking distance to the closest point.

	BOS/NY/DC	MIA	CHI	SEA	SF	LA	DEN
BOS/NY/DC	0	1075	671	2684	2799	2631	1616
MIA	1075	0	1329	3273	3053	2687	2037
CHI	671	1329	0	2013	2142	2054	996
SEA	2684	3273	2013	0	808	1131	1307
SF	2799	3053	2142	808	0	379	1235
LA	2631	2687	2054	1131	379	0	1059
DEN	1616	2037	996	1307	1235	1059	0

	BOS/ NY/DC	MIA	CHI	SEA	SF/LA	DEN
BOS/NY/DC	0	1075	671	2684	2631	1616
MIA	1075	0	1329	3273	2687	2037
CHI	671	1329	0	2013	2054	996
SEA	2684	3273	2013	0	808	1307
SF/LA	2631	2687	2054	808	0	1059
DEN	1616	2037	996	1307	1059	0

Note how creation of SF/LA cluster has changed distnces

	BOS/NY/DC	MIA	CHI	SEA	SF	LA	DEN
BOS/NY/DC	0	1075	671	2684	2799	2631	1616
MIA	1075	0	1329	3273	3053	2687	2037
CHI	671	1329	0	2013	2142	2054	996
SEA	2684	3273	2013	0	808	1131	1307
SF	2799	3053	2142	808	0	379	1235
LA	2631	2687	2054	1131	379	0	1059
DEN	1616	2037	996	1307	1235	1059	0

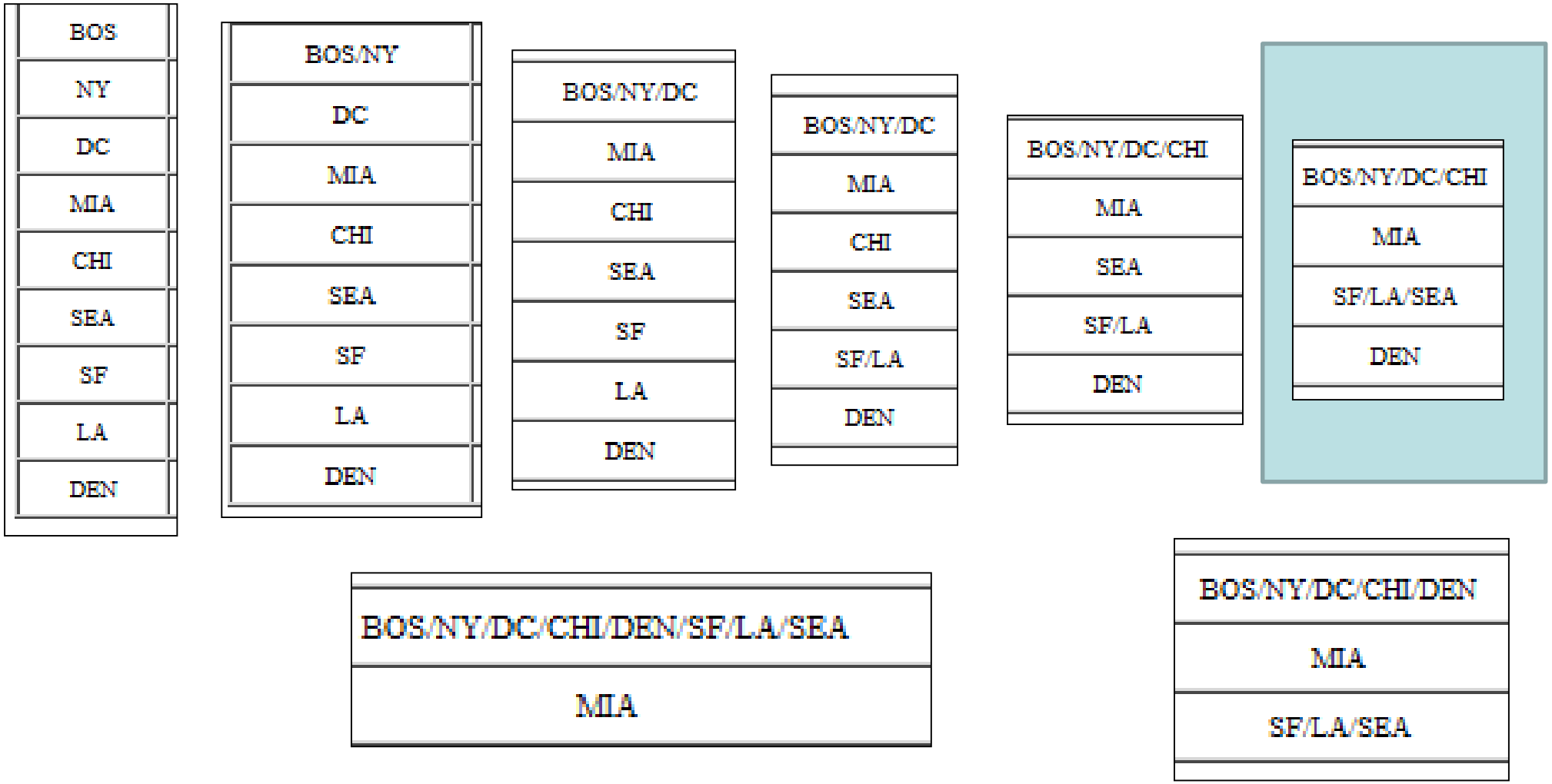
	BOS/NY/DC/ CHI	MIA	SEA	SF/LA	DEN
BOS/NY/DC/CHI	0	1075	2013	2054	996
MIA	1075	0	3273	2687	2037
SEA	2013	3273	0	808	1307
SF/LA	2054	2687	808	0	1059
DEN	996	2037	1307	1059	0

	BOS/NY/DC/CHI	MIA	SF/LA/SEA	DEN
BOS/NY/DC/CHI	0	1075	2013	996
MIA	1075	0	2687	2037
SF/LA/SEA	2054	2687	0	1059
DEN	996	2037	1059	0

	BOS/NY /DC/CHI/DEN	MIA	SF/LA/SEA
BOS/NY/DC/CHI/DEN	0	1075	1059
MIA	1075	0	2687
SF/LA/SEA	1059	2687	0

	BOS/NY /DC/CHI /DEN/SF /LA/SEA	MIA
BOS/NY/DC/CHI/DEN/SF/LA/SEA	0	1075
MIA	1075	0

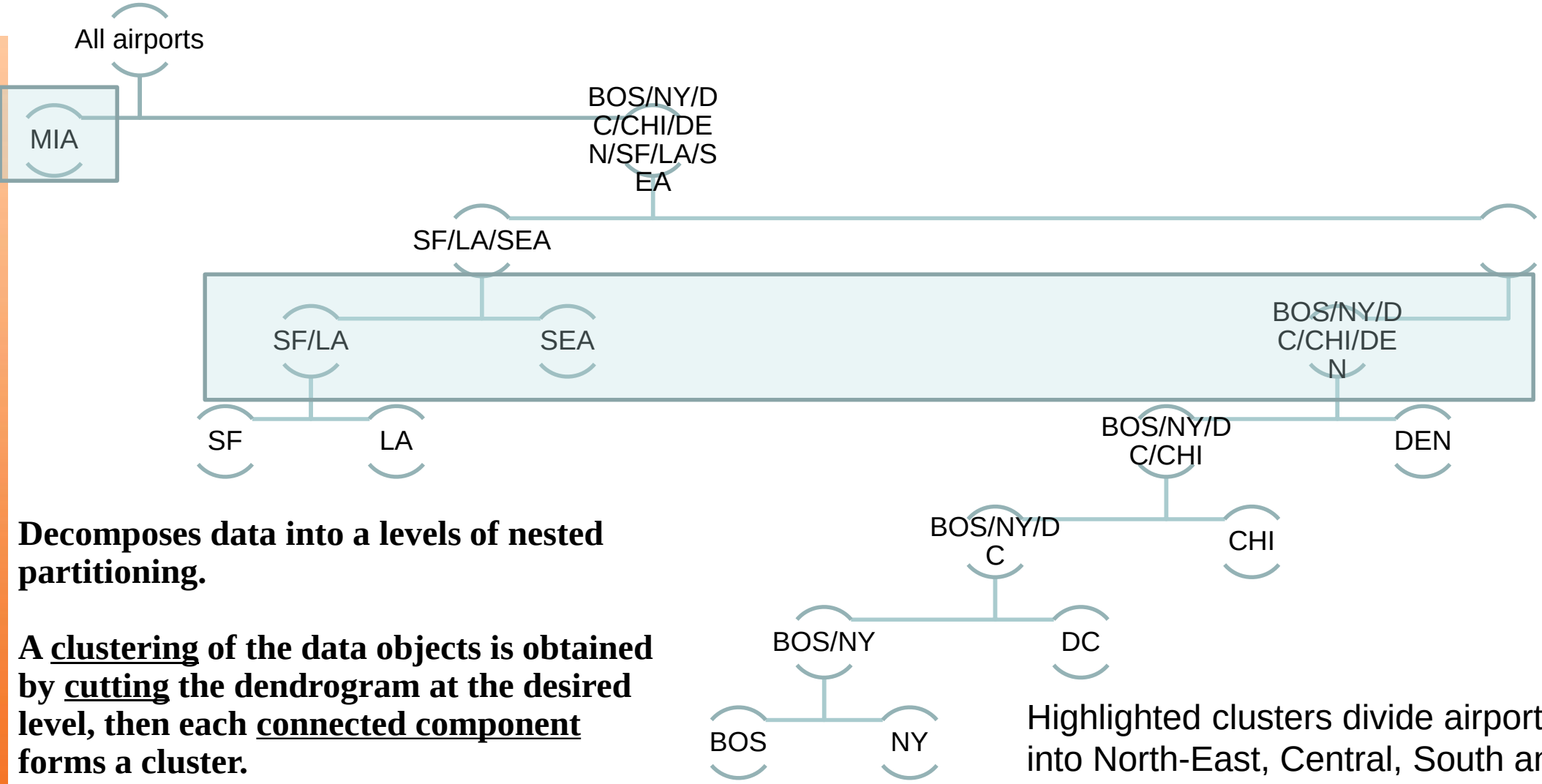
Agglomerative clustering (Hierarchical)



Typically, a particular “level” of the hierarchy is selected to be your clustering result

Highlighted clusters divide airports into North-East, Central, South and Pacific areas

Agglomerative clustering (Hierarchical)



Decomposes data into a levels of nested partitioning.

A clustering of the data objects is obtained by cutting the dendrogram at the desired level, then each connected component forms a cluster.

Highlighted clusters divide airports into North-East, Central, South and Pacific areas

Agglomerative clustering (Hierarchical)

- Assign each item to its own cluster, so that if you have N items, you now have N clusters, each containing just one item.
- Merge most similar clusters into a single cluster, so that now you have one less cluster.
- Compute distances (similarities) between the new cluster and each of the old clusters.
- Repeat steps 2 and 3 until all items are clustered into a single cluster of size N.

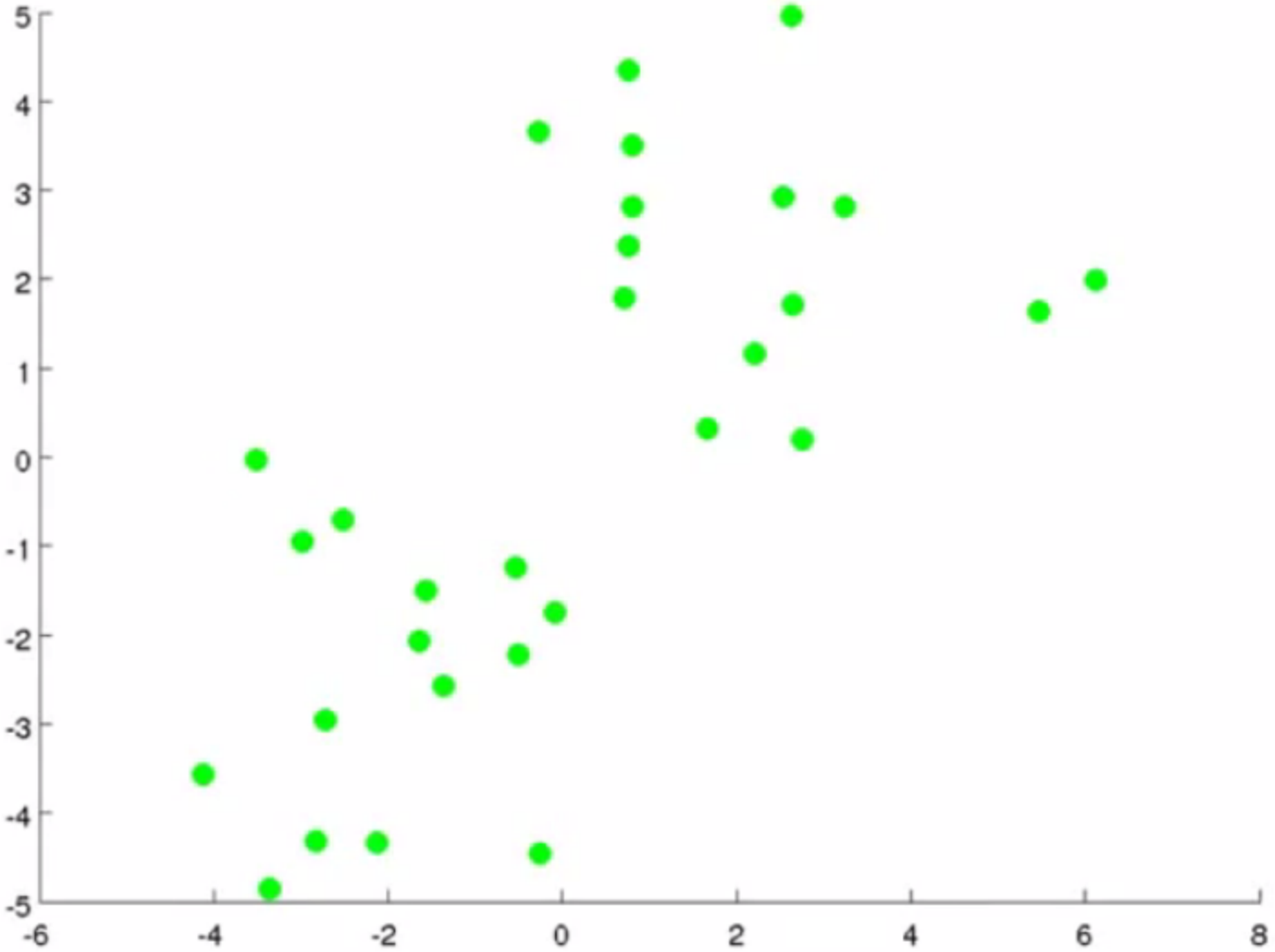
Partitioning algorithms

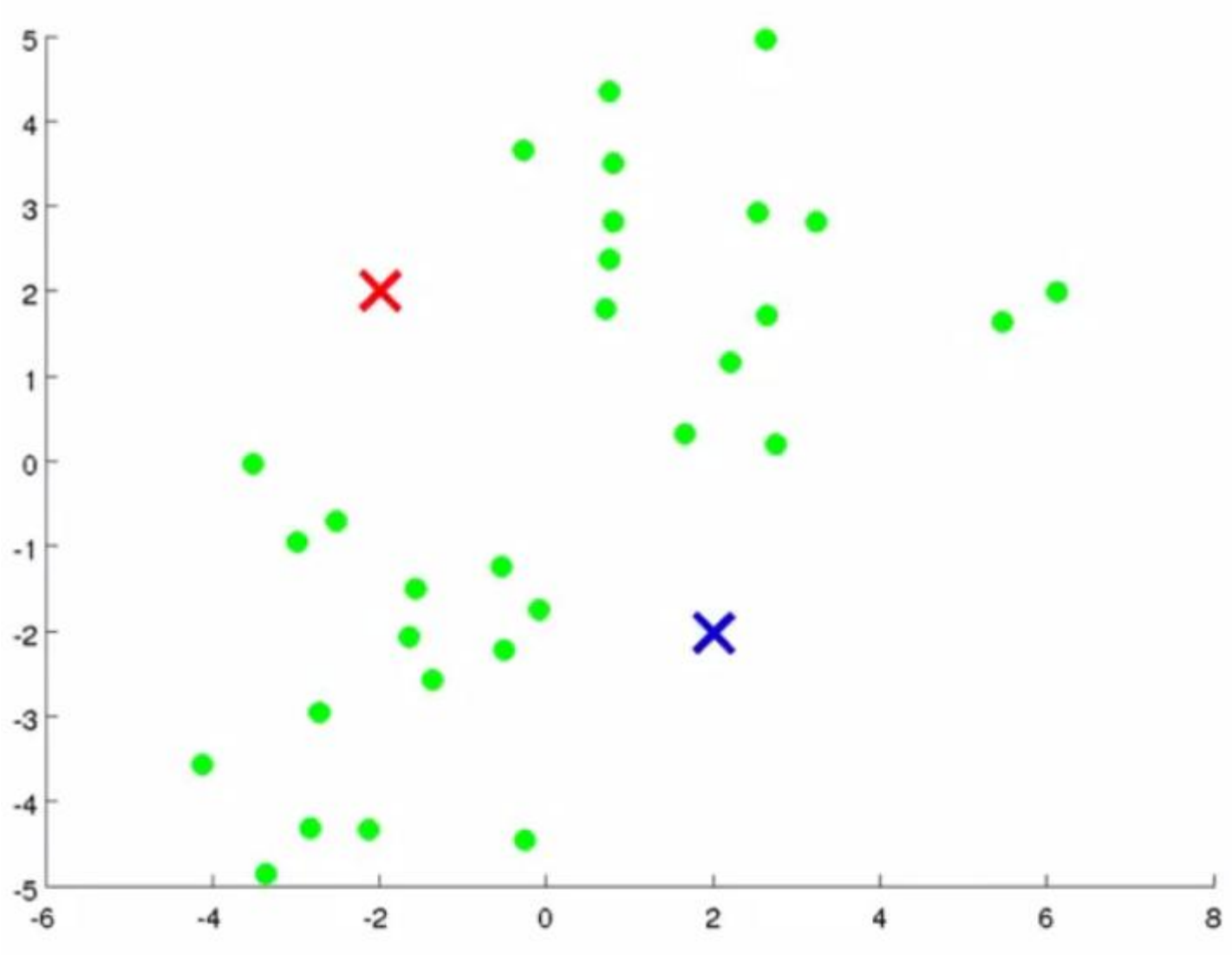
K-MEANS AND K-MEDOIDS

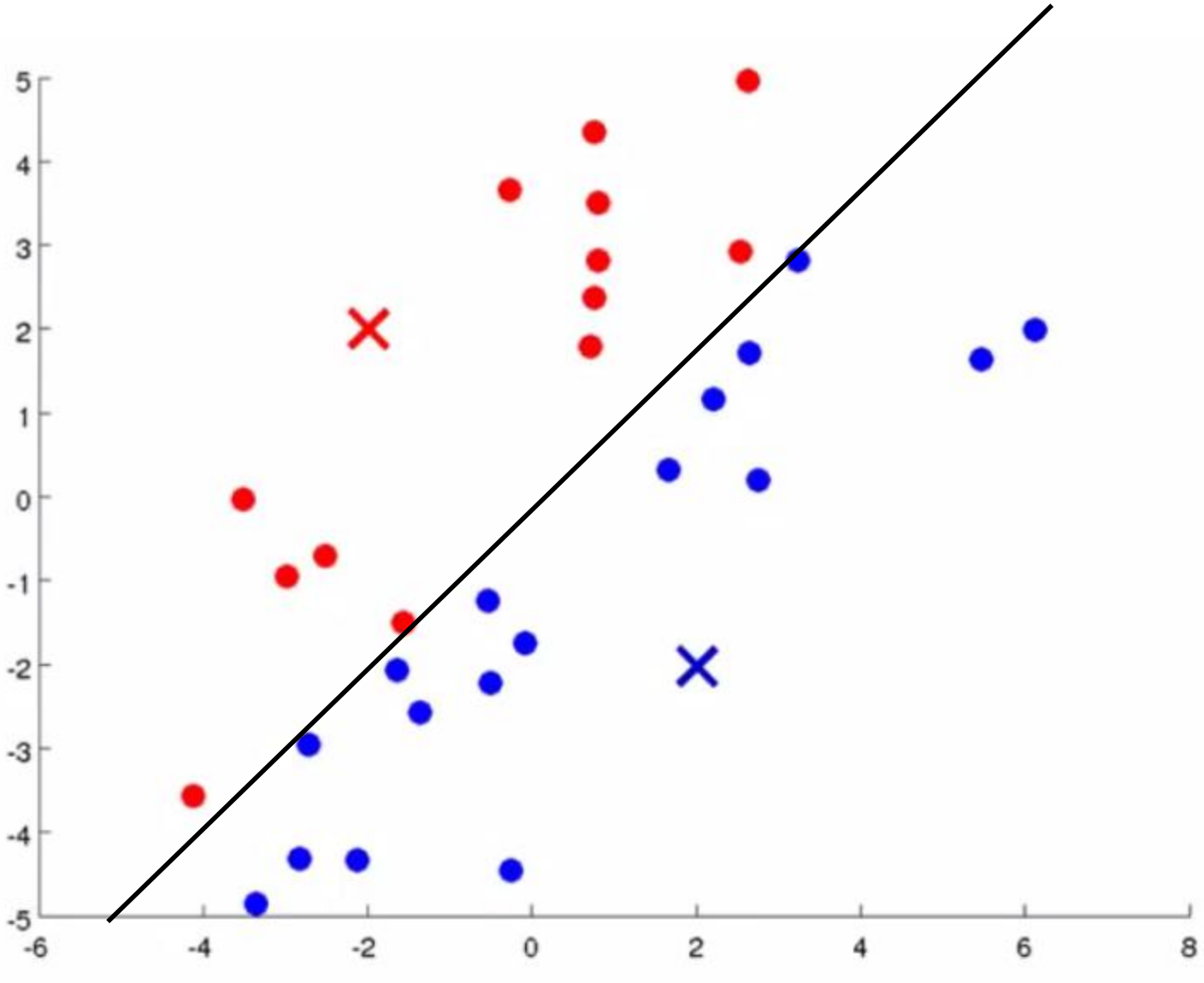
K-means clustering

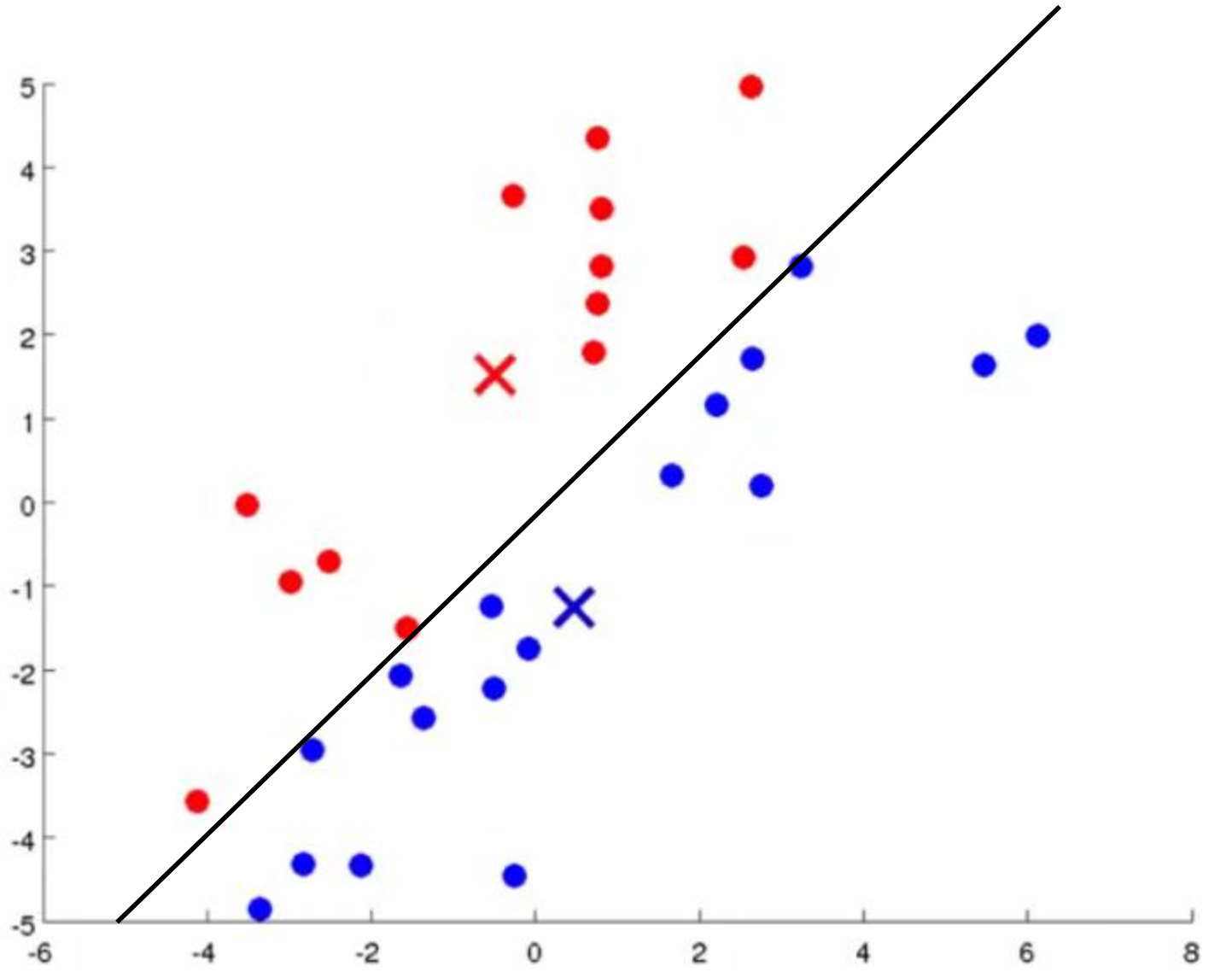
- K-means is a partitional clustering algorithm as it partitions the given data into *k* clusters.
  - Each cluster has a cluster **center**, called **centroid**.
  - *k* is specified by the user

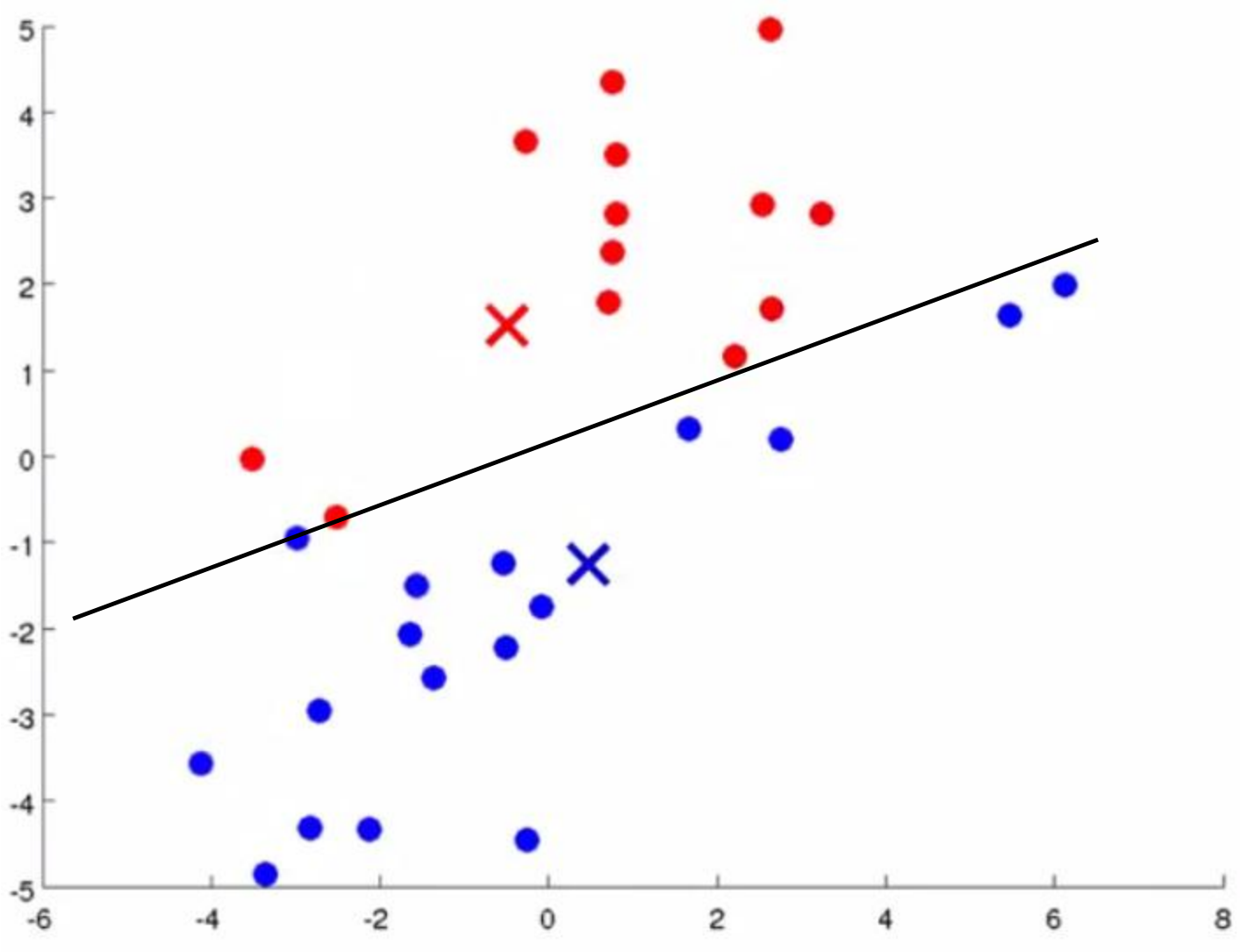
Pictures courtesy Andrew Ngs course on Coursera

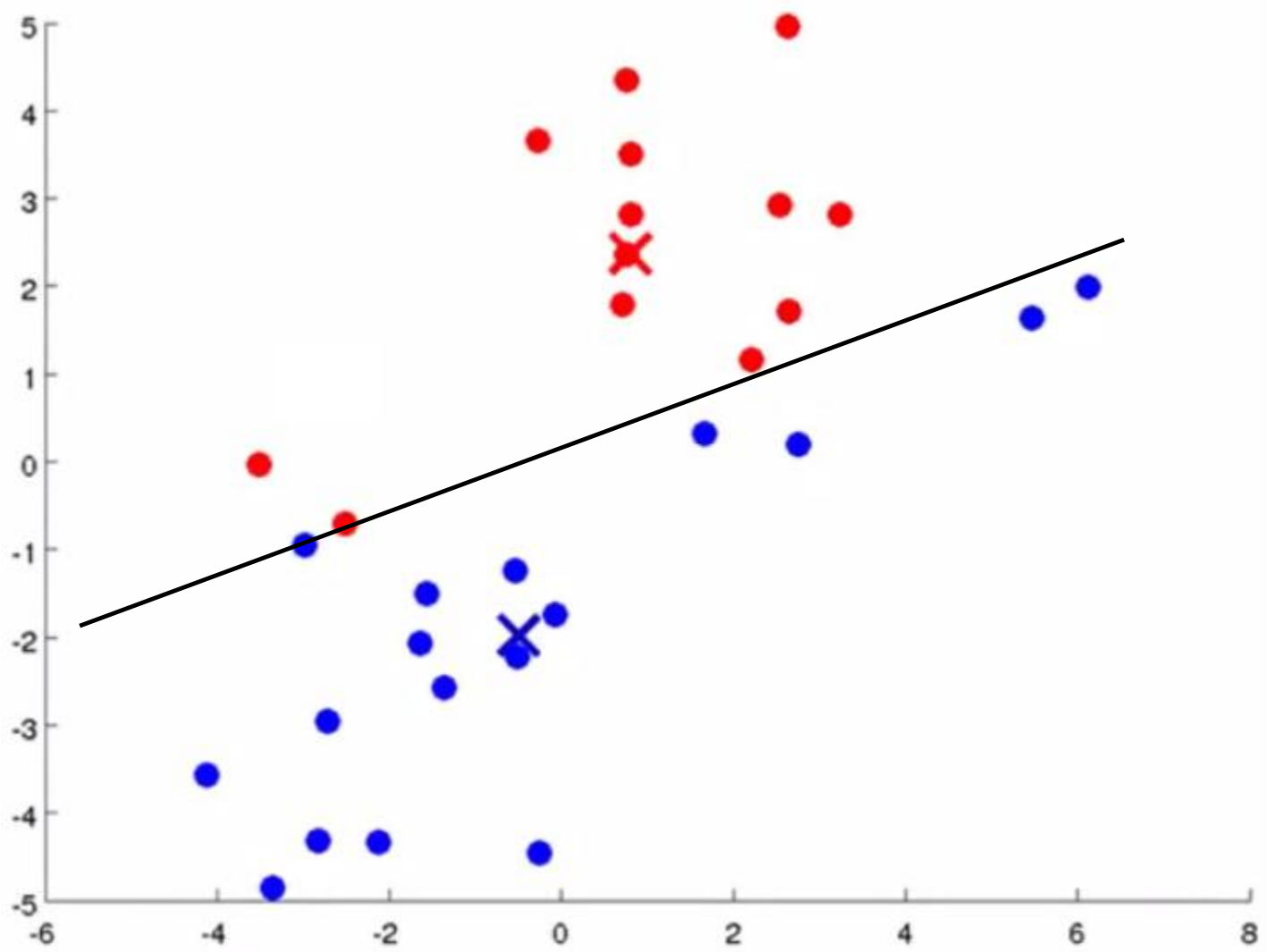


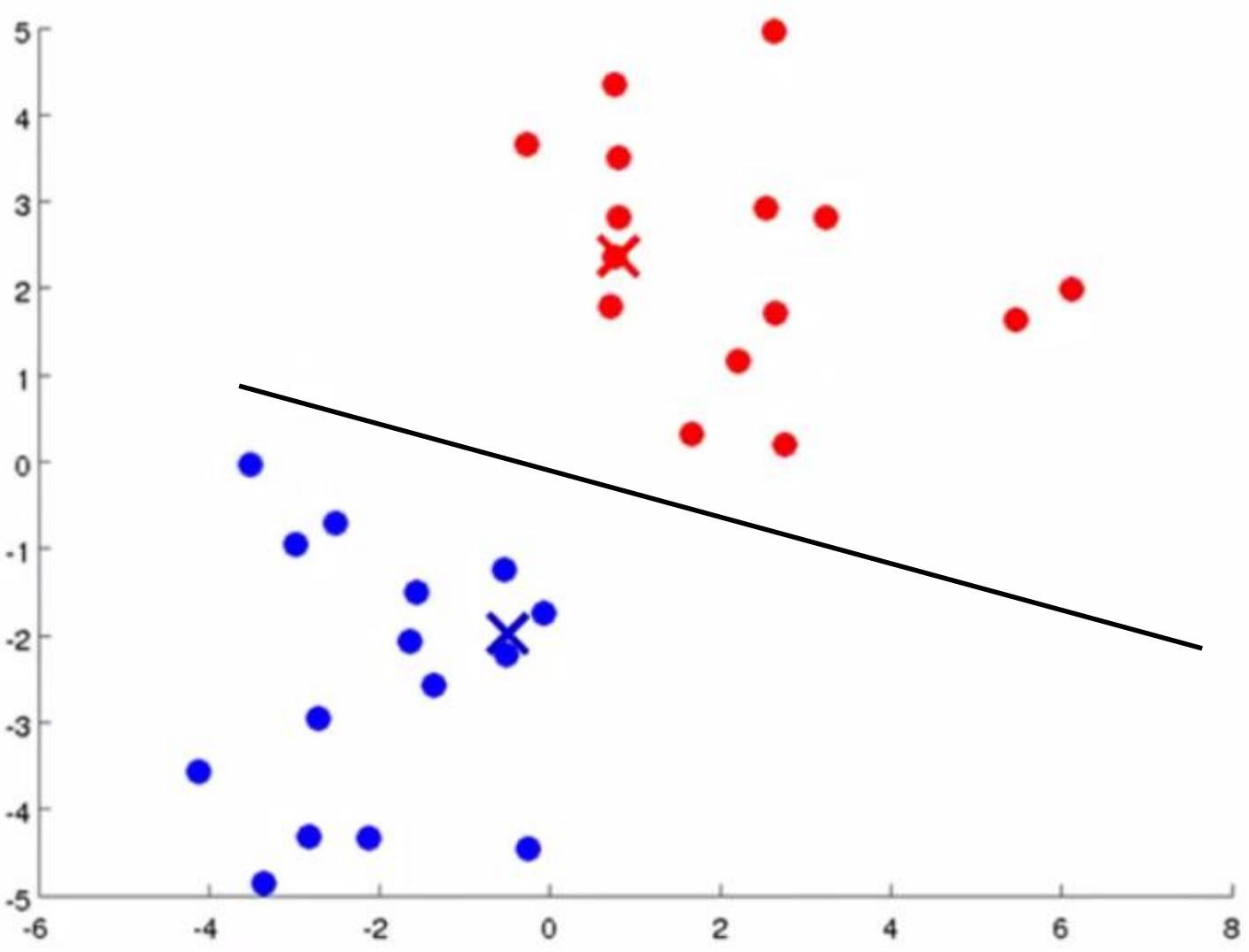

5
4
3
2
1
0
-1
-2
-3
-4
-5
-6
-4
-2
0
2
4
6
8

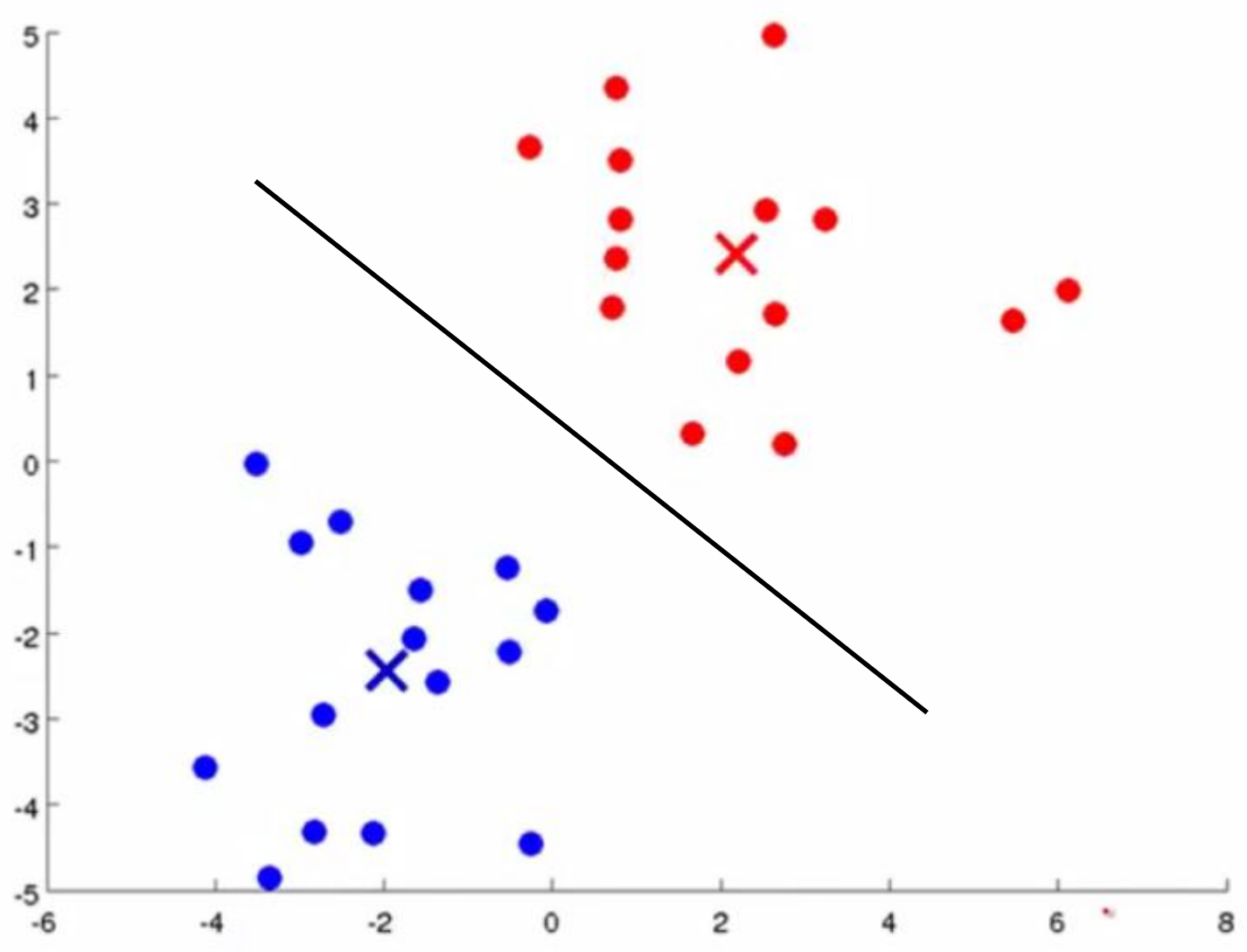

5
4
3
2
1
0
-1
-2
-3
-4
-5
-6
-4
-2
0
2
4
6
8


5
4
3
2
1
0
-1
-2
-3
-4
-5
-6
-4
-2
0
2
4
6
8


5
4
3
2
1
0
-1
-2
-3
-4
-5
-6
-4
-2
0
2
4
6
8


5
4
3
2
1
0
-1
-2
-3
-4
-5
-6
-4
-2
0
2
4
6
8


5
4
3
2
1
0
-1
-2
-3
-4
-5
-6
-4
-2
0
2
4
6
8


5
4
3
2
1
0
-1
-2
-3
-4
-5
-6
-4
-2
0
2
4
6
8

K-means algorithm

- Given *k*, the *k-means* algorithm works as follows:

1. Randomly choose *k* data points (seeds) to be the initial centroids, cluster centers
2. Assign each data point to the closest centroid
3. Re-compute the centroids using the current cluster memberships.
4. If a convergence criterion is not met, or **if some clusters don't get any points** go to 2.

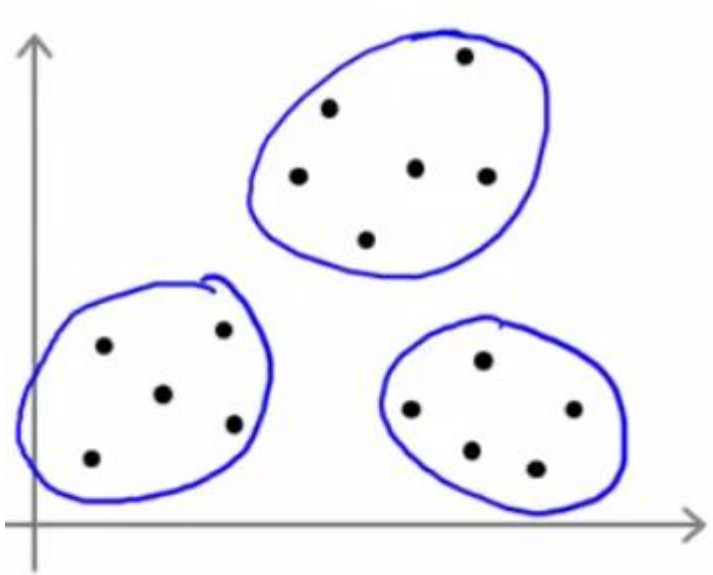
Stopping/convergence criterion

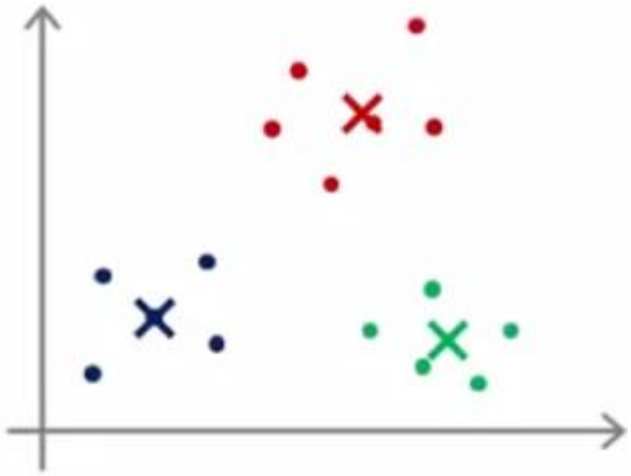
1. no (or minimum) re-assignments of data points to different clusters,
2. no (or minimum) change of centroids, or
3. minimum decrease in the **sum of squared error** (SSE),

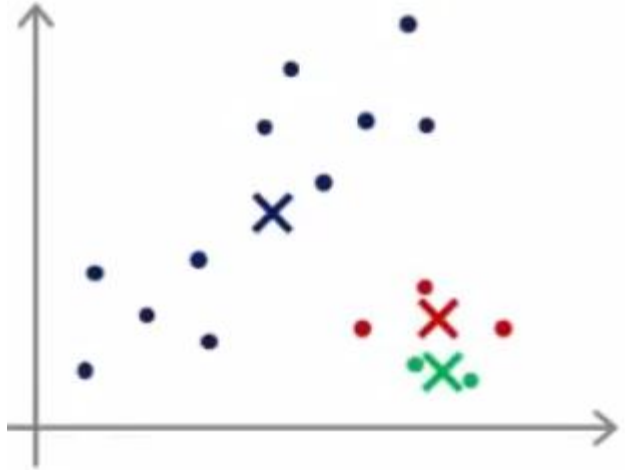
$$SSE = \sum_{j=1}^{k} \sum_{\mathbf{x} \in C_j} EuclidianDist(\mathbf{x}, \mathbf{m}_j)^2 \quad (1)$$

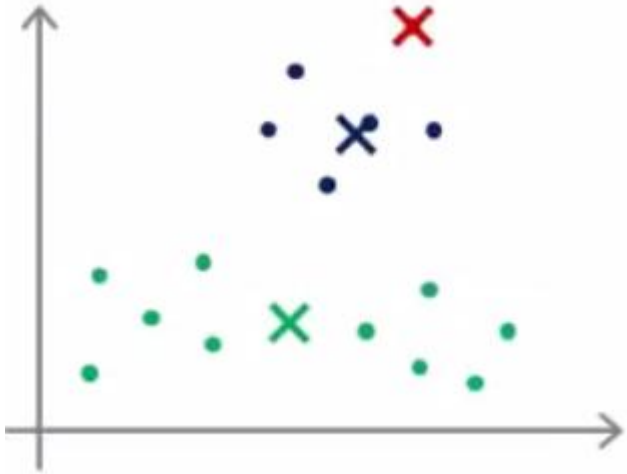
– C_j is the jth cluster, $\mathbf{m}_j$ is the centroid (mean) of cluster C_j

Local optima









What Is the Problem with K-Means?

- The k-means algorithm is sensitive to outliers !
- K-Medoids: Instead of taking the **mean** value of the object in a cluster as a reference point, **medoids** can be used, which is the **most centrally located** object in a cluster.

What Is the Problem with Medoids?

- More robust than k-means in the presence of noise and outliers because a medoid is less influenced by outliers or other extreme values than a mean
- Works efficiently for small data sets but does not **scale well** for large data sets.
  - In regular K-Means, the new mean is merely average of values O(nkt)
  - In K-Medoid, if a particular cluster has n_c data points, you will need $(n_c)^2$ computations to determine the medoid for that cluster
  - $O(k(n-k)^2)$ for each iteration

where n is # of data, k is # of clusters, t: # of iterations

K-means versus Hierarchical

- K-means produces a single partitioning
- K-means needs the number of clusters to be specified
- K-means is usually more efficient run-time wise

- Hierarchical Clustering can give different partitions depending on the level-of-resolution we are looking at
- Hierarchical clustering doesn't need the number of clusters to be specified
- Hierarchical clustering can be slow (has to make several merge/Split decisions)

HOW DO WE EMPLOY DISTANCE IN A CLUSTER

What do we mean by distance *between* Clusters

- Single link: smallest distance between an element in one cluster and an element in the other, i.e., $dis(K_i, K_j) = min(t_{ip}, t_{jq})$
- Complete link: largest distance between an element in one cluster and an element in the other, i.e., $dis(K_i, K_j) = max(t_{ip}, t_{jq})$
- Average: average distance between an element in one cluster and an element in the other, i.e., $dis(K_i, K_j) = avg(t_{ip}, t_{jq})$
- **Centroid: distance between the centroids of two clusters, i.e., $dis(K_i, K_j) = dis(C_i, C_j)$**
- Medoid: distance between the medoids of two clusters, i.e., $dis(K_i, K_j) = dis(M_i, M_j)$
  - Medoid: one chosen, centrally located object in the cluster

Centroid, Radius & Diameter of a Cluster (for numerical data sets)

- Centroid: the "middle" of a cluster

$$C_m = \frac{\sum_{i=1}^{N} (t_{ip})}{N}$$

- Radius: square root of average distance from any point of the cluster to its centroid

$$R_m = \sqrt{\frac{\sum_{i=1}^{N} (t_{ip} - c_m)^2}{N}}$$

- Diameter: square root of average mean squared distance between all pairs of points in the cluster

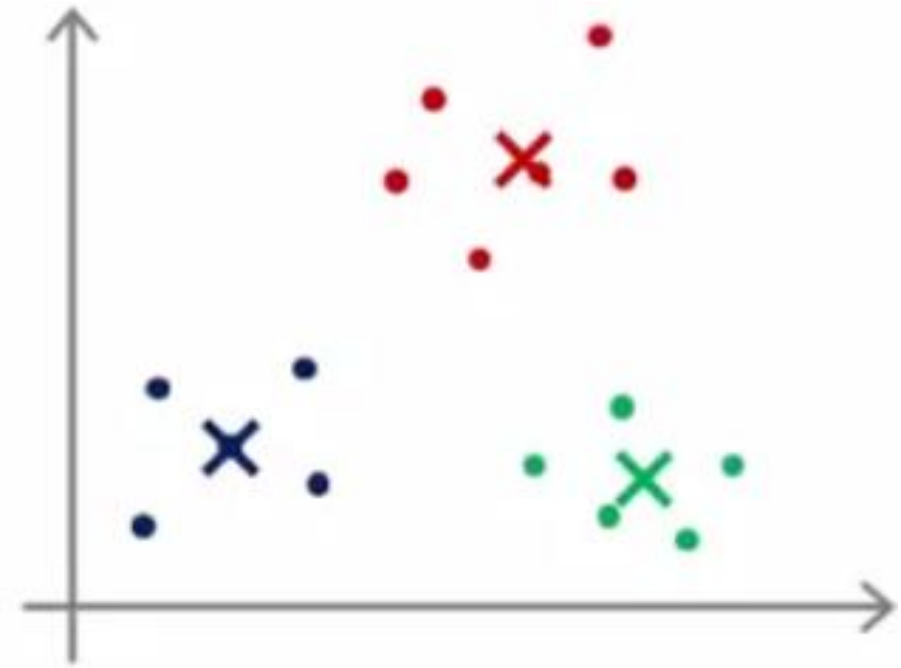
$$D_m = \sqrt{\frac{\sum_{i=1}^{N} \sum_{i=1}^{N} (t_{ip} - t_{iq})^2}{N(N-1)}}$$

ENGINEERING

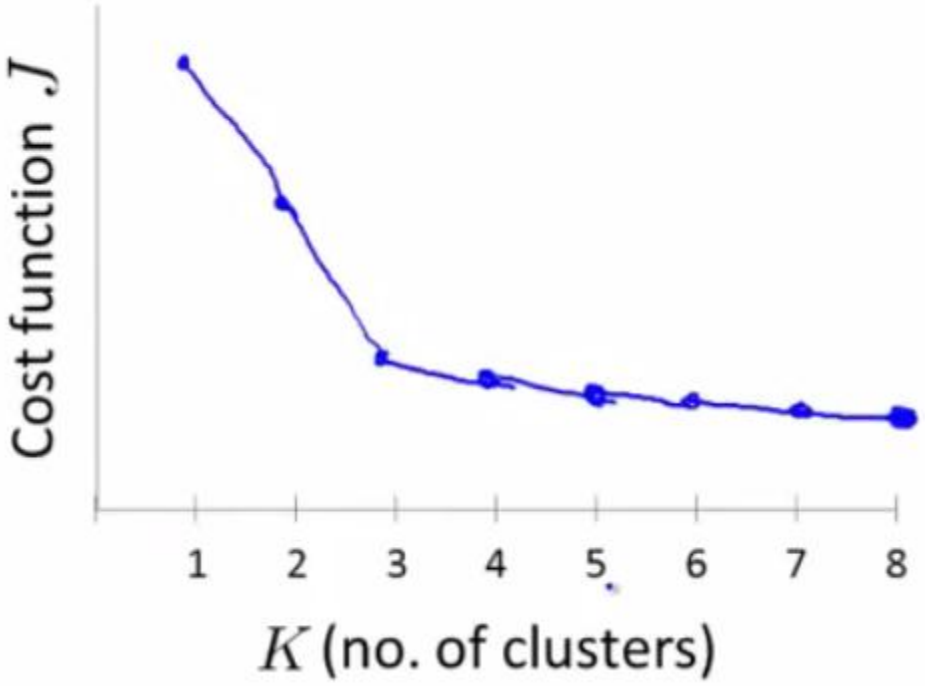
Stability Check of the Clusters

- To check the stability of the clusters take a random sample of 95% of records. Compute the clusters. If the clusters formed are very similar to the original, then the clusters are fine.

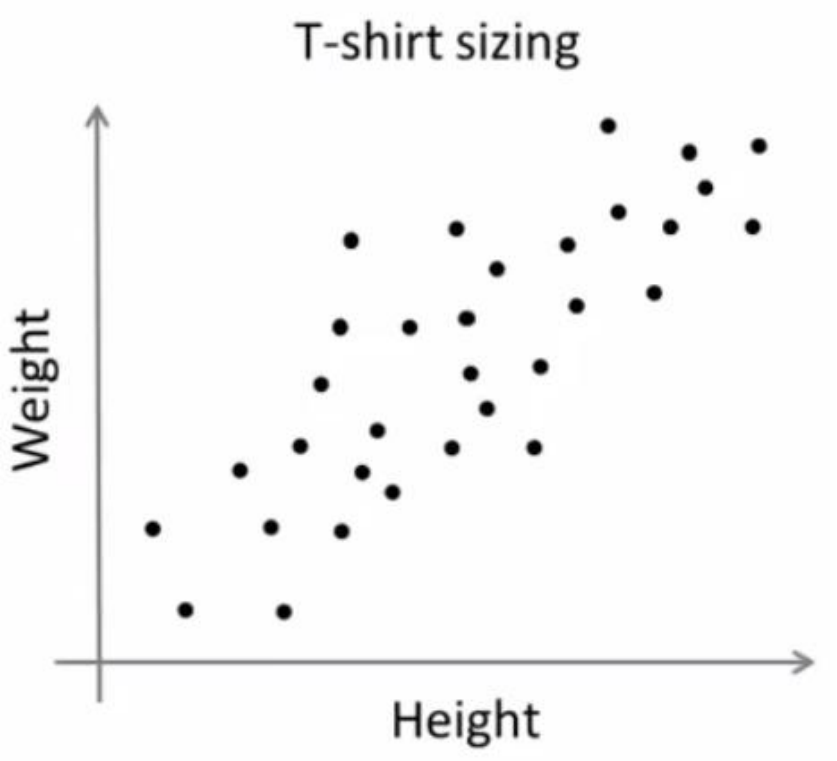
Linearly clustered data

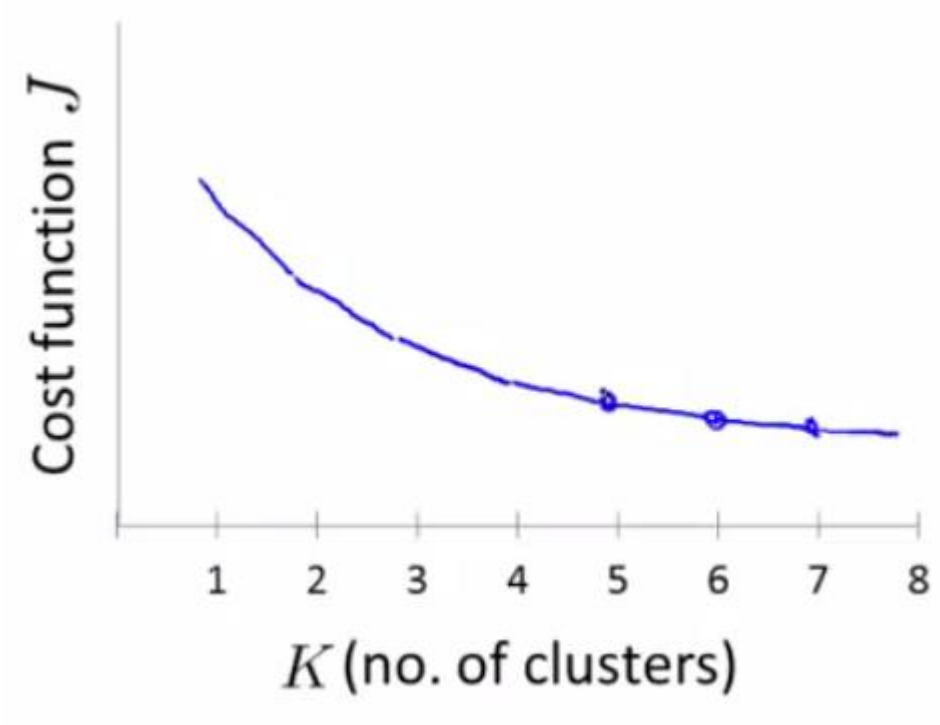


Nice



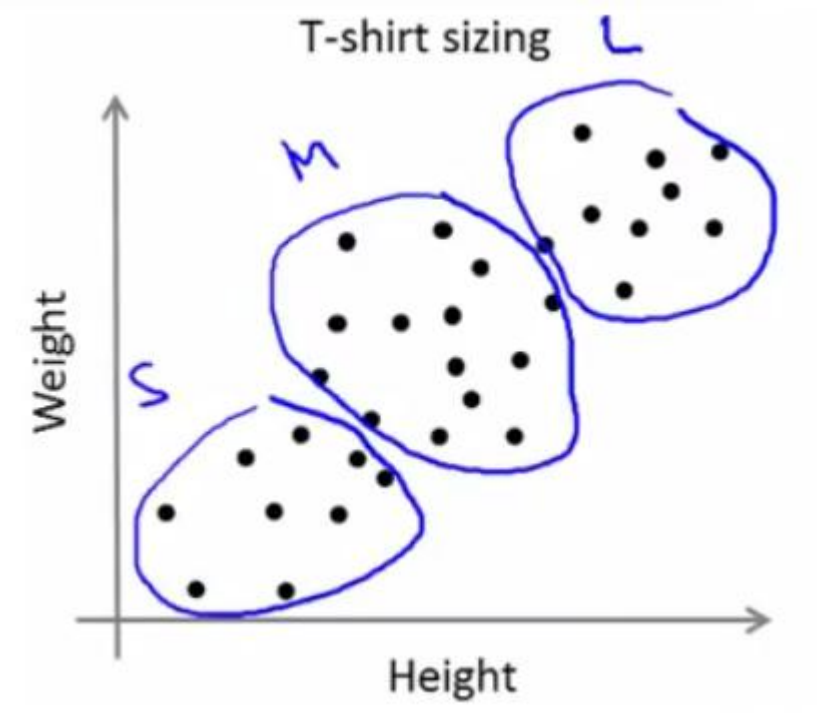
Linearly separable but merged

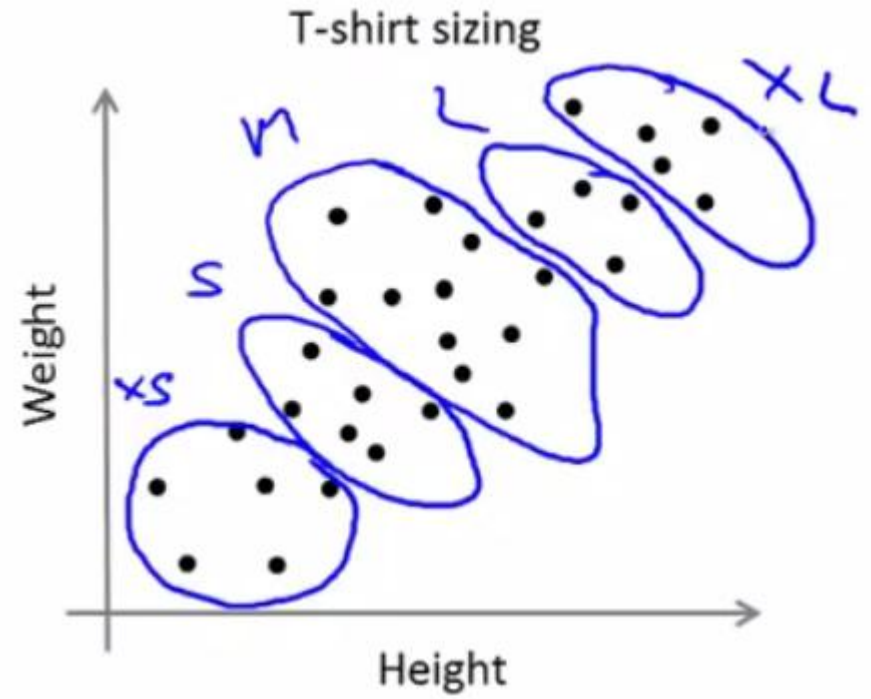




Data points of individuals of a particular height and weight

Linearly separable but merged





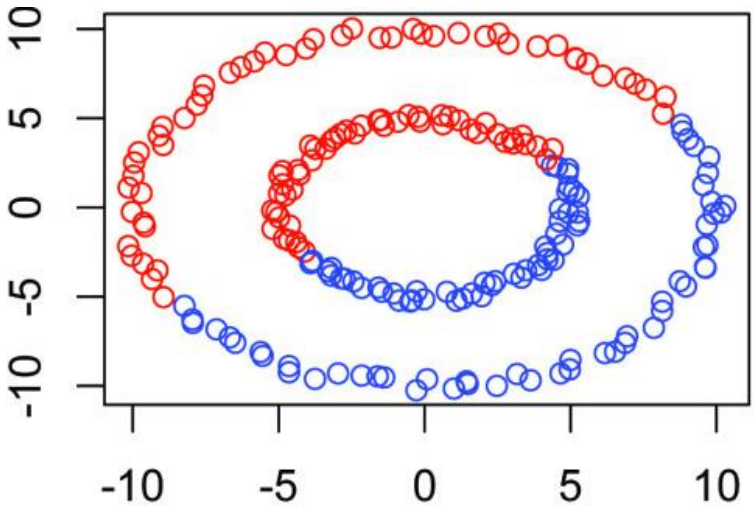
Clustering the data points into T-shirt sizes. Top: Three sizes, Right: Five sizes

Linearly separable

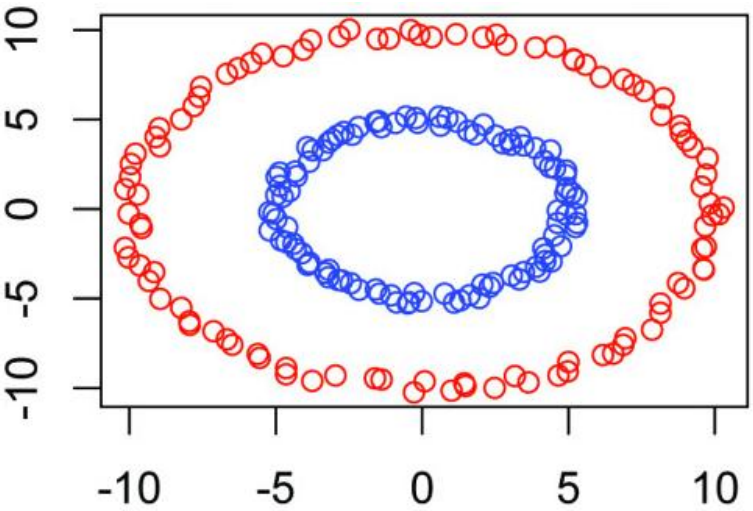
- Run 50-500 simulations for small k (2-10). For large k (100 or so), we can do 1-5 simulations
- Pick the one that gives the best SSE

k-means Vs. Kernel k-means

k-means



Kernel k-means



Performance of Kernel K-means

Clustering accuracy (%) achieved by each clustering algorithm for 10 data sets

Data set	Conventional				Kernel			
	k-means	FCM	Average	Mountain	*k*-means	FCM	Average	Mountain
BENSAID	79.59	73.47	100.0	85.71	83.67	93.88	100.0	100.0
DUNN	70.00	70.00	100.0	83.33	71.11	95.56	100.0	100.0
IRIS	89.33	89.33	90.67	52.67	96.00	93.33	89.33	93.33
ECOLI	42.86	49.11	76.49	51.19	68.75	61.01	77.38	69.05
CIRCLE	50.76	52.79	62.44	55.84	100.0	93.40	82.74	62.94
BLE-3	65.67	65.67	56.00	70.33	76.33	74.67	100.0	71.67
BLE-2	88.50	87.75	100.0	85.25	100.0	94.00	100.0	100.0
UE-4	77.25	66.00	71.45	73.50	100.0	98.50	100.0	84.75
UE-3	95.83	95.00	100.0	51.17	98.83	96.67	100.0	95.67
ULE-4	76.25	94.75	76.25	96.25	98.00	96.25	100.0	96.25
Avg. (%)	73.60	74.39	83.33	70.52	89.27	89.73	94.95	87.37

Evaluation of the performance of clustering algorithms in kernel-induced feature space, Pattern Recognition, 2005

Kernel Trick

- The original way is to transform each data point into a high dimensional space and then do computation
  - This can be computationally complex
- Note that we need to compute distances in Euclidian space, or sometimes scalar product

Kernel Trick

- Take two points: $X_1 = (x_{11}, x_{12})$
  $X_2 = (x_{21}, x_{21})$
  - $Euclidian\ distance: d(X_1, X_2) = \sqrt{(x_{11} - x_{21})^2 + (x_{12} - x_{22})^2}$
  - $Dot\ product: X_1.X_2 = (x_{11}.x_{21}) + (x_{12}.x_{22})$
- Take to a higher dimensional space
  $\check{X}_1 = \left(x_{11}, x_{12}, {x_{11}}^2, {x_{12}}^2, \sqrt{2}x_{11}x_{12}\right)$
  $\check{X}_2 = \left(x_{21}, x_{22}, {x_{21}}^2, {x_{22}}^2, \sqrt{2}x_{21}x_{22}\right)$

Kernel Trick

- $Euclidian\ distance: d(X_1, X_2) = \sqrt{(x_{11} - x_{21})^2 + (x_{12} - x_{22})^2} = \sqrt{((x_{11})^2 - 2x_{11}x_{21} + (x_{21})^2) + ((x_{12})^2 - 2x_{12}x_{22} + (x_{22})^2)}$
- $Dot\ product: X_1.X_2 = (x_{11}.x_{21}) + (x_{12}.x_{22})$

- Take to a higher dimensional space

$$\check{X}_1 = \left(x_{11}, x_{12}, {x_{11}}^2, {x_{12}}^2, \sqrt{2}x_{11}x_{12}\right)$$

$$\check{X}_2 = \left(x_{21}, x_{22}, {x_{21}}^2, {x_{22}}^2, \sqrt{2}x_{21}x_{22}\right)$$

While computing $d(\check{X}_1, \check{X}_2)$ re-use the six values computed for $d(X_1, X_2)$

Computing $\check{X}_1.\check{X}_2$ also can be done by reusing $X_1.X_2$

Kernel trick continued

- This is where Kernel trick comes into picture - we can choose a kernel where the transformed space is of high dimension and yet it is easy to compute the similarity score in the original space.

Other Topics

- Clustering large datasets
  - Select a small % of data, run K-means or K-medoids
  - CLARA and CLARANS (Ng and Han 1994, 2002)
- Parallel and Efficient implementations of K-means / K-medoids

http://www.math.unipd.it/~dulli/corso04/ng94efficient.pdf
https://anuradhasrinivas.files.wordpress.com/2013/04/lesson8-clustering.pdf
http://www.vlfeat.org/overview/kmeans.html
http://repository.cmu.edu/cgi/viewcontent.cgi?article=2397&context=compsci
http://www.cs.ucsb.edu/~veronika/MAE/Global_Kernel_K-Means.pdf

International School of Engineering

Plot 63/A, 1st Floor, Road # 13, Film Nagar, Jubilee Hills, Hyderabad - 500 033

For Individuals: +91-9502334561/63 or 040-65743991

For Corporates: +91-9618483483

Web: http://www.insofe.edu.in

Facebook: https://www.facebook.com/insofe

Twitter: https://twitter.com/Insofeedu

YouTube: http://www.youtube.com/InsofeVideos

SlideShare: http://www.slideshare.net/INSOFE

LinkedIn: http://www.linkedin.com/company/international-school-of-engineering