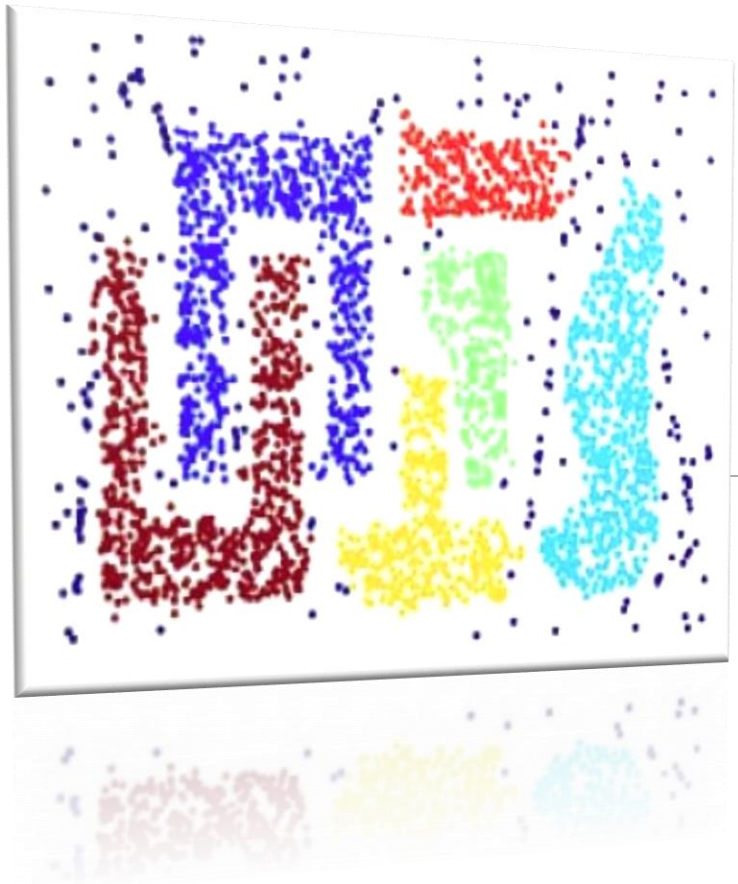
STATISTICAL METHODS IN DATA MINING

DR. ALPER VAHAPLAR



Clustering

Measure of Similarity, Hierarchical Clustering, K-means Clustering

Clustering

- ***Clustering*** is the process of grouping a set of physical or abstract unlabelled objects into classes of similar objects.
- A ***Cluster*** is a collection of data objects that are ***similar*** to one another within the same cluster, and ***dissimilar*** to the objects in other clusters.
- Clustering is an important human activity:
  - Distinguishing animals and plants, male and female, cars and busses etc.
- Goals:
  - Detecting natural groups in data,
  - Creating homogenous classes,
  - Data reduction, Outlier detection.

Clustering

- ✓ Some issues:
- ✓ How to measure similarity?
- ✓ How many clusters?
- ✓ What is the correct cluster?

Clustering

- ✓ Measuring ***Similarity***

  or measuring ***dissimilarity***?

- ✓ A distance measure to calculate the differences between two objects d(x,y) should have the properties:

  1. $d(x, y) \geq 0$ for all x and *y*
  2. $d(x, y) = 0$ only if $x = y$. (Positive definiteness)
  3. $d(x, y) = d(y, x)$ for all x and y. (Symmetry)
  4. $d(x, z) \leq d(x, y) + d(y, z)$ for all points x, y, and z. (Triangle Inequality)

Clustering

- ✓ Distance Function
- ✓ Euclidean Distance

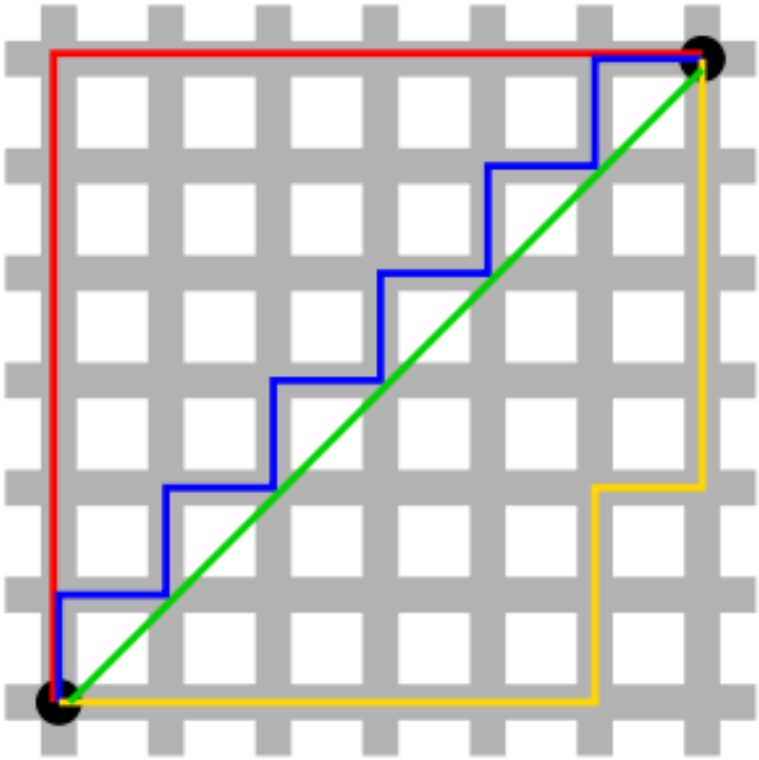
$$d_{\text{Euc}}(x,y) = \sqrt{\sum_i (x_i - y_i)^2}$$

- ✓ Manhattan (City Block) Distance

$$d_{\text{Man}}(x,y) = \sum_i |x_i - y_i|$$

- ✓ Minkowski Distance

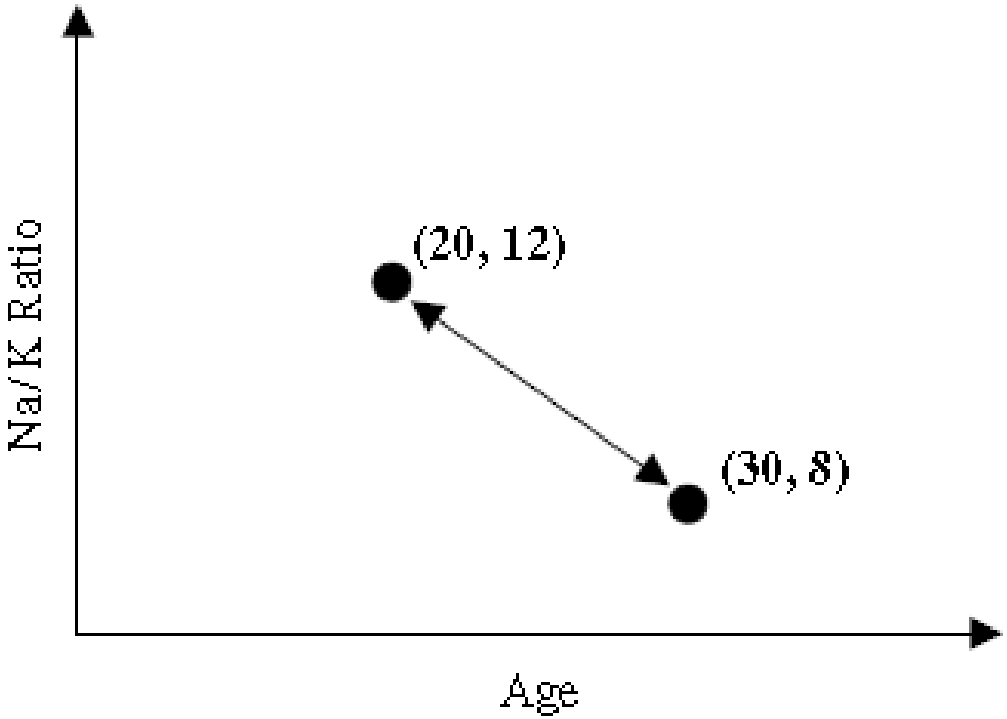
$$d_{\text{Min}}(x,y) = \sqrt[\lambda]{\sum_i |x_i - y_i|^{\lambda}}$$



Euclidean	$dist(x, y) = \sqrt{\sum_i (x_i - y_i)^2}$
Squared Euclidean	$dist(x, y) = \sum_i (x_i - y_i)^2$
City-block (Manhattan)	$dist(x, y) = \sum_i \lvert x_i - y_i \rvert$
Minkowski	$dist(x, y) = \sqrt[\lambda]{\sum_i \lvert x_i - y_i \rvert^{\lambda}}$
Chebychev	$dist(x, y) = \max(\lvert x_i - y_i \rvert)$
Mahalanobis	Pearson Correlation
Power	Spearman Correlation
Percent Disagreement	

Clustering

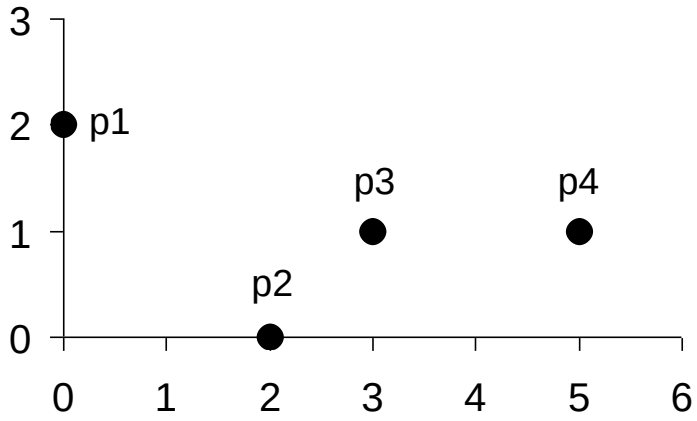
- ✓ Distance Measure
- ✓ Example:



$$d_{\text{Euclidean}}(\mathbf{x}, \mathbf{y}) = \sqrt{\sum_i (x_i - y_i)^2} = \sqrt{(20-30)^2 + (12-8)^2}$$

$$= \sqrt{100 + 16} = 10.77$$

Clustering



point	x	y
p1	0	2
p2	2	0
p3	3	1
p4	5	1

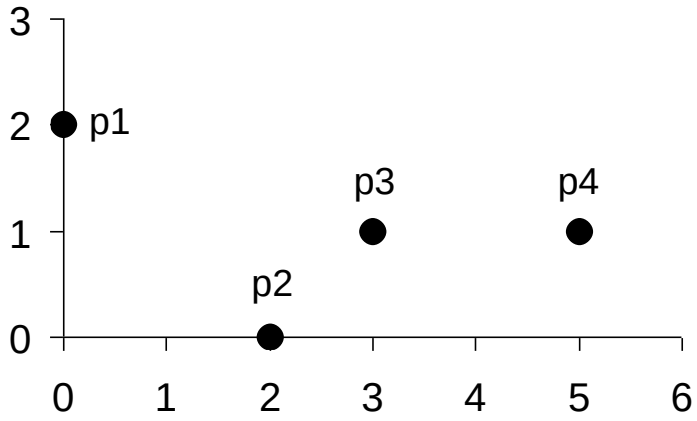
	p1	p2	p3	p4
p1				
p2				
p3				
p4				

Euclidean Distance Matrix

	p1	p2	p3	p4
p1	0	4	4	6
p2	4	0	2	4
p3	4	2	0	2
p4	6	4	2	0

Manhattan Distance Matrix

Clustering



point	x	y
p1	0	2
p2	2	0
p3	3	1
p4	5	1

	p1	p2	p3	p4
p1	0	2.828	3.162	5.099
p2	2.828	0	1.414	3.162
p3	3.162	1.414	0	2
p4	5.099	3.162	2	0

Euclidean Distance Matrix

	p1	p2	p3	p4
p1	0	4	4	6
p2	4	0	2	4
p3	4	2	0	2
p4	6	4	2	0

Manhattan Distance Matrix

Clustering

- ✓ Problems in distance measure
  - Different ranges in data
    - Normalization (min-max, Z-score, etc)
  - Categorical variables

$$\text{different}(x_i, y_i) = \begin{cases} 0 & \text{if } x_i = y_i \\ 1 & \text{otherwise} \end{cases}$$

Clustering

- ✓ Find the distance between Ali and Ayşe, Ali and Veli, Ayşe and Veli

Adı	Yaşı	Kilosu	Gözrengi
Ali	22	65	Siyah
Ayşe	19	52	Ela
Veli	23	60	Siyah

Variable	Yaşı	Kilosu
Min	18	50
Max	30	85

Clustering

- ✓ Find the distance between Ali and Ayşe, Ali and Veli, Ayşe and Veli

Adı	Yaşı	Kilosu	Gözrengi
Ali	22 (0,33)	65 (0,43)	Siyah
Ayşe	19 (0,08)	52 (0,06)	Ela
Veli	23 (0,42)	60 (0,29)	Siyah

Variable	Yaşı	Kilosu
Min	18	50
Max	30	85

d	**Ali**	**Ayşe**	**Veli**
Ali	0	1.096	0.165
Ayşe	1.096	0	1.079
Veli	0.165	1.079	0

Clustering

- ✓ Distance measure for Categorical Variables
- ✓ Binary Data (0/1 - presence/absence – Yes/No)
- ✓ Jackard's Distance

		Object *j*		
		1	0	*sum*
Object *i*	1	*a*	*b*	*a*+*b*
	0	*c*	*d*	*c*+*d*
	sum	*a*+*c*	*b*+*d*	*p*

$$d(i,j) = \frac{b+c}{a+b+c}$$

Example for Clustering Categorical Data

- ✓ Find the Jaccard's distance between Apple and Banana.

Feature of Fruit	Sphere shape	Sweet	Sour	Crunchy
Object i =Apple	Yes	Yes	Yes	Yes
Object j =Banana	No	Yes	No	No

(a = 1, b = 3, c = 0, d= 0)

(3+0) / (1 + 3 + 0) = 3/4 = 0.75

$$d(i,j) = \frac{b+c}{a+b+c}$$

		Object j 1	0	sum
Object i	1	a	b	$a+b$
	0	c	d	$c+d$
	sum	$a+c$	$b+d$	p

Example for Clustering Categorical Data

- ✓ Who are the most likely to have a similar disease?

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

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) = $\frac{1+2}{1+1+2} = 0.75$

$$d(i,j) = \frac{b+c}{a+b+c}$$

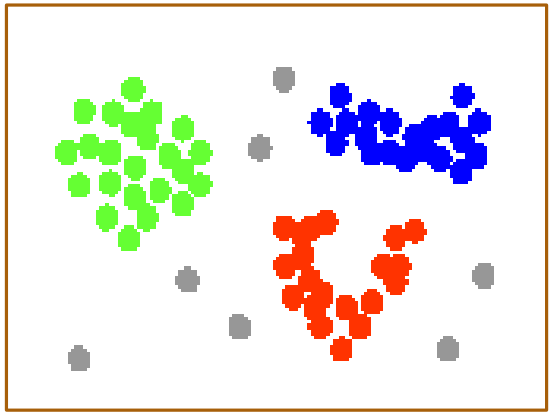
		Object j		
		1	0	*sum*
Object i	1	a	b	$a+b$
	0	c	d	$c+d$
	sum	$a+c$	$b+d$	p

Result: Jim and Mary are unlikely to have a similar disease.

Jack and Mary are the most likely to have a similar disease.

Clustering Methods

- ✓ **Hierarchical Methods**
  - AGNES, DIANA, BIRCH, CURE, CHAMELEON, ...
- ✓ **Partitioning Methods**
  - K-Means, K-Medoids, PAM, CLARA, CLARANS, ...
- ✓ **Density-Based Methods**
  - DBSCAN, OPTICS, DENCLUE, ...
- ✓ **Grid-Based Methods**
  - STING, WaveCluster, CLIQUE ...
- ✓ **Model-Based Methods**
  - COBWEB, CLASSIT, SOM (Self-Organizing Feature Maps) ...



Hierarchical Clustering

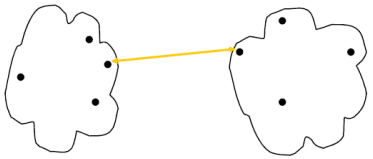
- ✓ A tree like cluster structure (dendrogram)
- ✓ Agglomerative
  - Each item is a tiny cluster of its own at the beginning,
  - Two closest clusters are aggregated,
  - At the end, all items are in one cluster.
- ✓ Divisive methods
  - All items are in one cluster at the beginning,
  - Most dissimilar cluster are seperated,
  - At the end, each record represents its own cluster.

Hierarchical Clustering

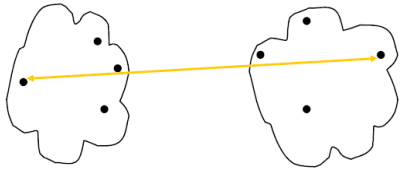
- ✓ Measuring distance between clusters in Hierarchical Clustering
- ✓ **Single linkage**,
  - the nearest-neighbor approach,
  - based on the minimum distance between any record in two clusters
- ✓ **Complete linkage**,
  - the farthest-neighbor approach,
  - based on the maximum distance between any record in two clusters.
- ✓ **Average linkage ,**
  - is designed to reduce the dependence of the cluster-linkage criterion on extreme values, such as the most similar or dissimilar records.
  - the criterion is the average distance of all the records in cluster A from all the records in cluster B.

Hierarchical Clustering

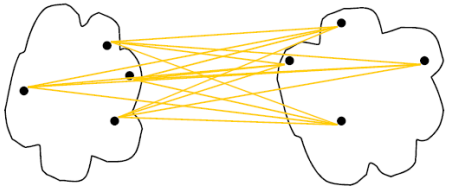
- ✓ Single link: smallest distance between an element in one cluster and an element in the other,



- ✓ Complete link: largest distance between an element in one cluster and an element in the other.

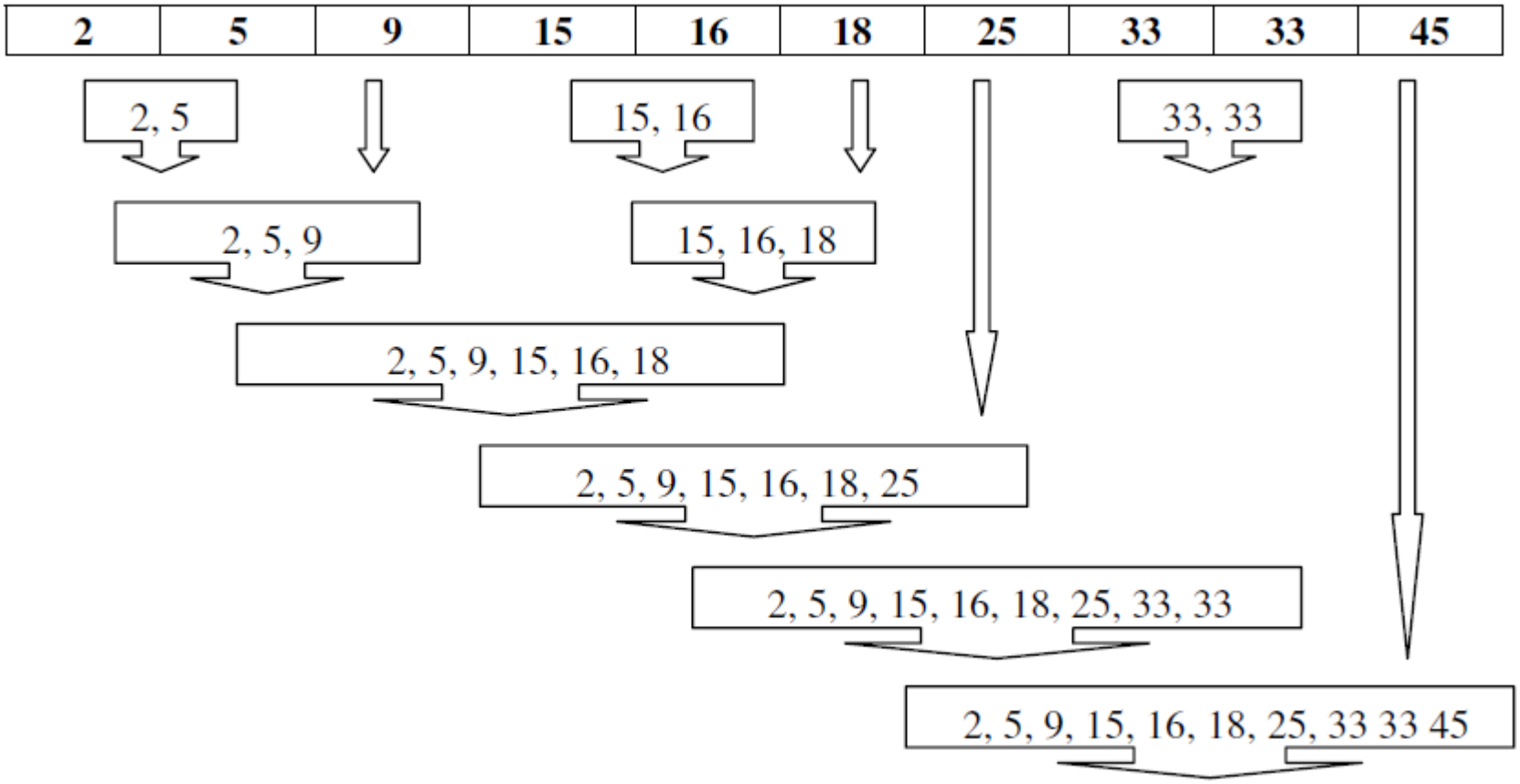


- ✓ Average: avg. distance between an element in one cluster and an element in the other.



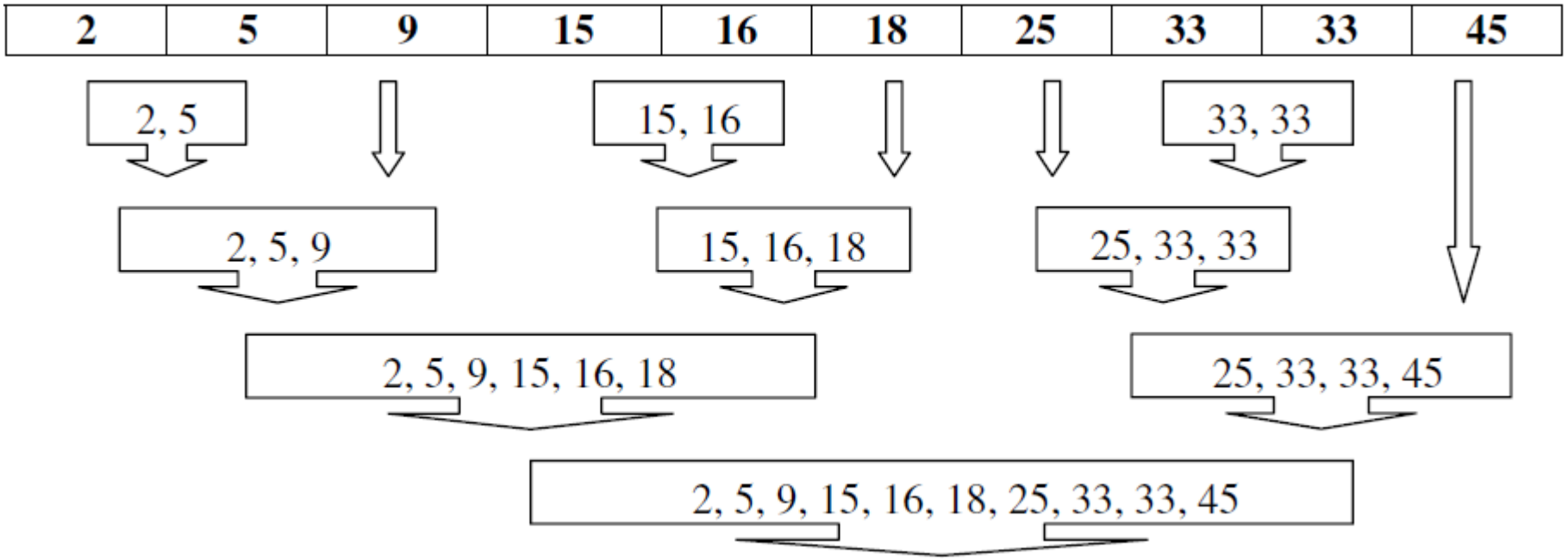
Single-Linkage Clustering - Example

Dataset: 2,5,9,15,16,18,25,33,33,45



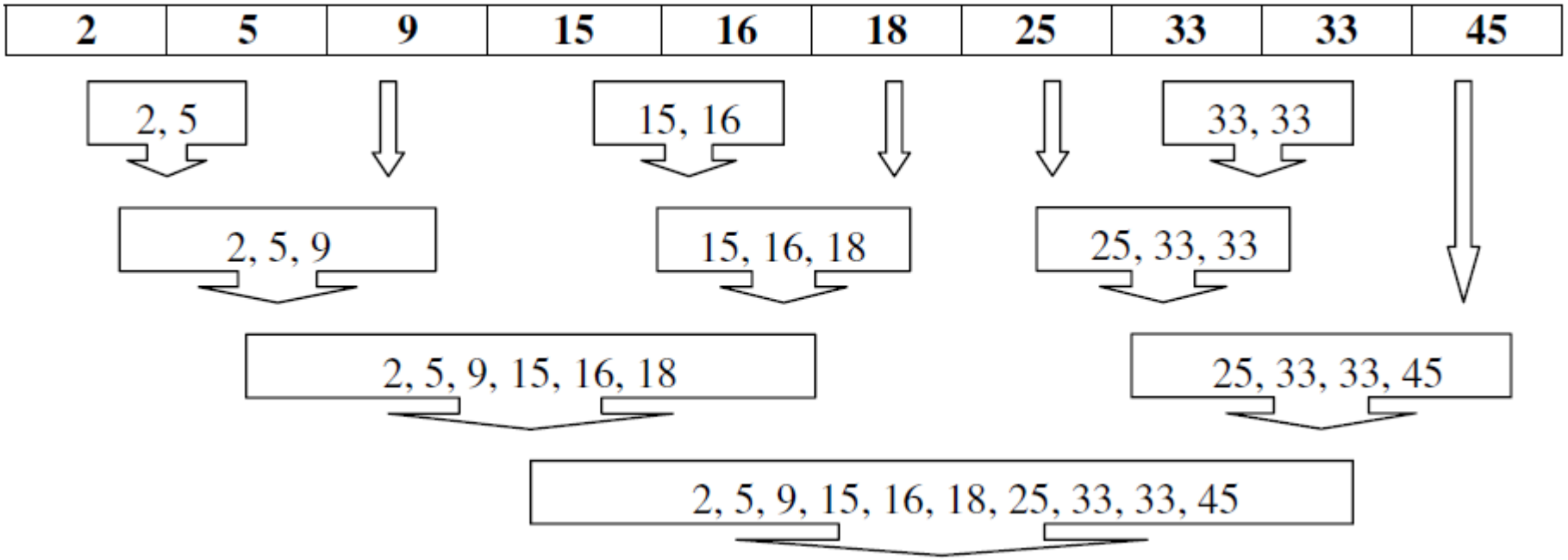
Complete-Linkage Clustering - Example

- ✓ Dataset: 2,5,9,15,16,18,25,33,33,45



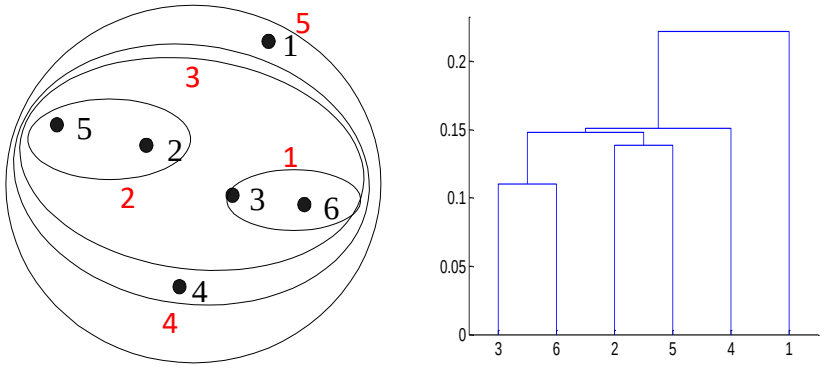
Average-Linkage Clustering - Example

- ✓ Dataset: 2,5,9,15,16,18,25,33,33,45

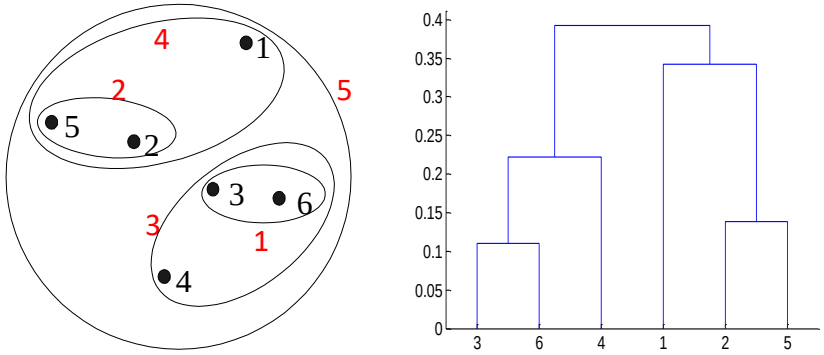


$$The\ average\ dis\tan ce = \frac{\sum_{x \in A} \sum_{y \in B} d(x, y)}{A_{count} * B_{count}}$$

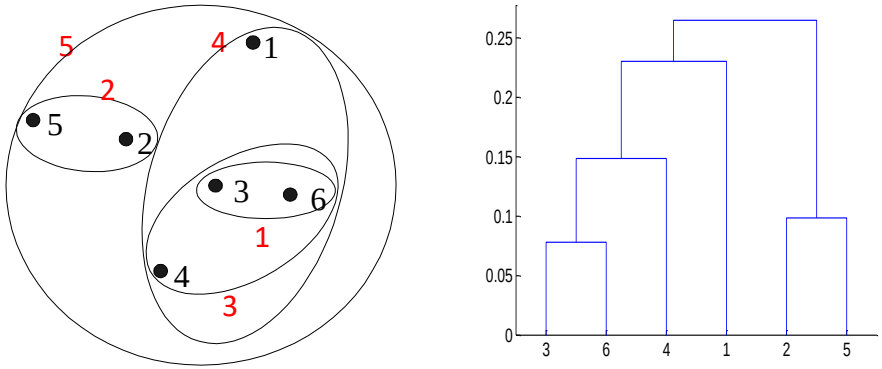
How the Clusters are Merged?



Single Link



Complete Link



Average Link

Hierarchical Clustering

- ✓ Single Linkage
  - Can handle non-elliptical shapes
  - Sensitive to noise and outliers
- ✓ Complete Linkage
  - Less sensitive to noise and outliers
  - Tends to break large clusters and to form more compact, globular clusters
- ✓ Average Linkage
  - Less sensitive to noise and outliers
  - Tends to form more compact, globular clusters (similar to complete linkage)

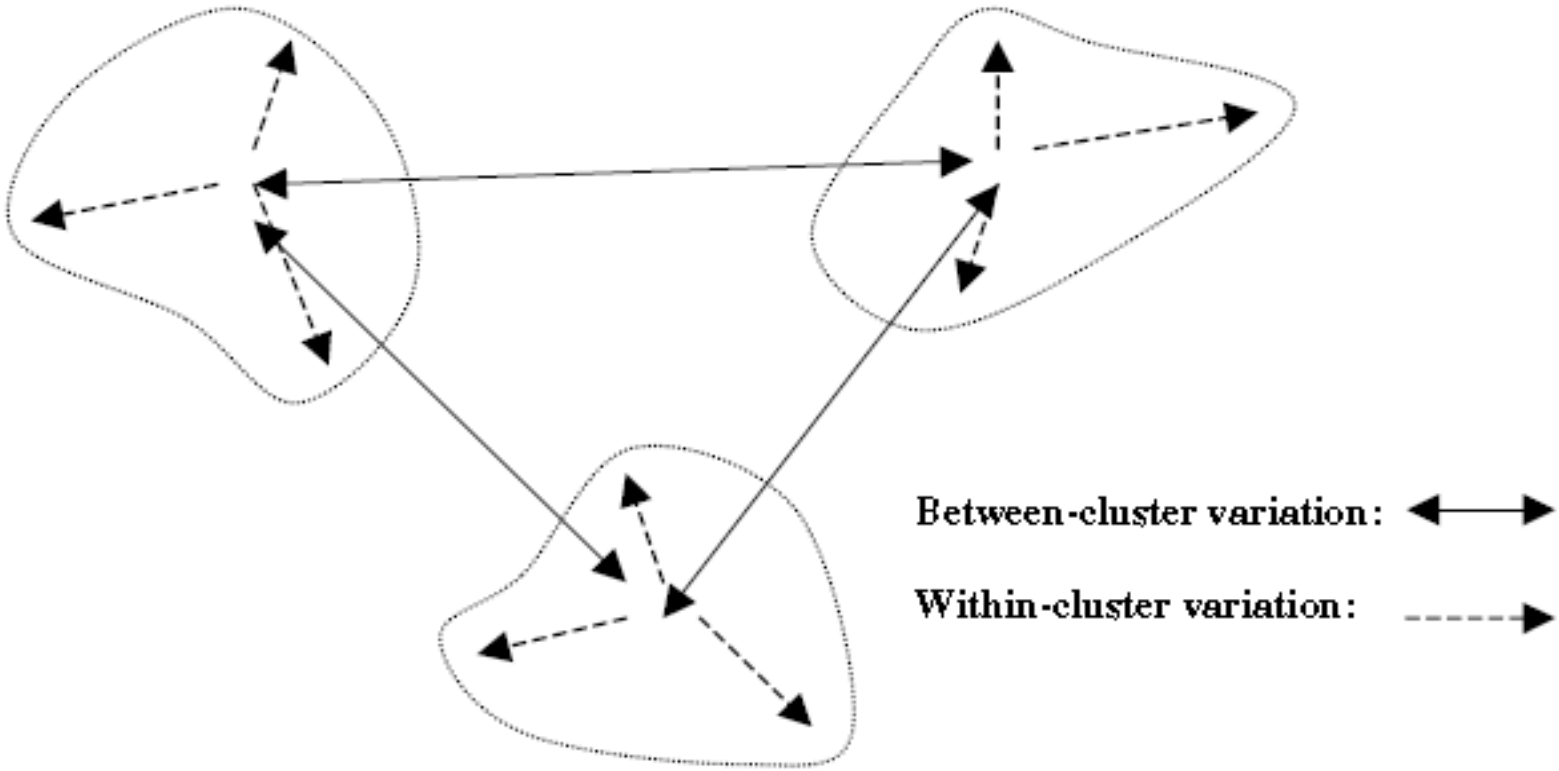
Hierarchical Clustering

- ✓ Advantages
  - Does not require the number of cluster
  - Easy to implement
  - Fast and less complex

- ✓ Disadvantages
  - Need to know where to cut the tree
  - Sensitivity to noise and outliers
  - Difficulty handling different sized clusters and convex shapes
  - Tend to break large clusters

Partition Based Clustering

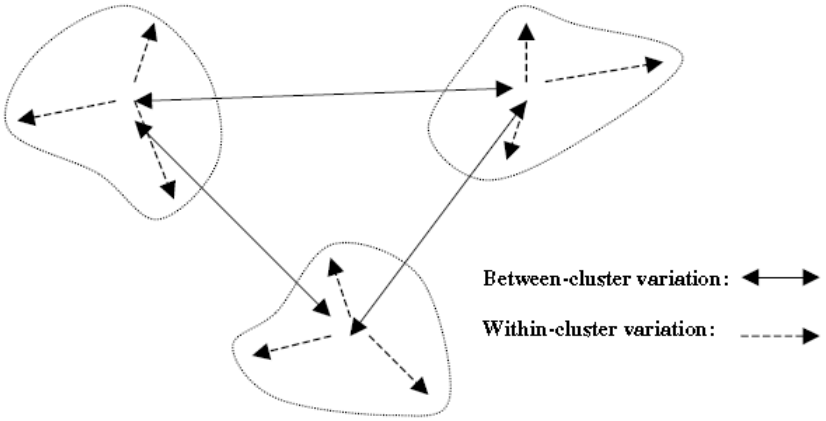
- ✓ Aims to construct a partition of a database **D** of **n** objects into a set of **k** clusters such that the sum of squared distances is minimized.
- ✓ Given a *k*, find a partition of *k clusters* that optimizes the chosen *partitioning criterion* e.g. minimize SSE.



Partition Based Clustering

- ✓ Within Cluster Variation (WCV)

$$SSE = \Sigma_{i=1}^{k} \Sigma_{p \in C_i} d(p - c_i)^2$$



- ✓ Between Cluster Variation (BCV)

$$BCV = d(c_1, c_2)$$

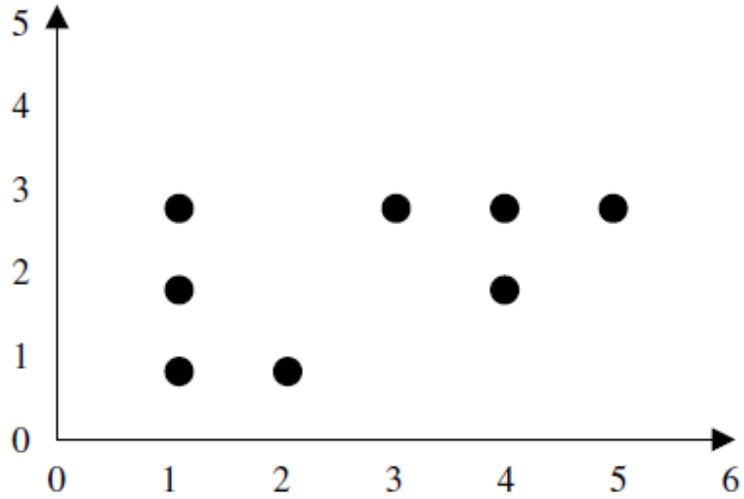
- ✓ Maximize the between-cluster-variation with respect to to within-cluster-variation

$$\frac{BCV}{WCV} = \frac{d(c_1, c_2)}{SSE}$$

Partition Based Clustering

- **k-means Clustering**
- is an algorithm to cluster ***n*** objects based on attributes into ***k*** partitions, $k < n$
- *Step 1: Ask **k**,*
- *Step 2: Randomly assign **k** point as the initial cluster centers,*
- *Step 3: For each data point, find the nearest cluster center and assign it to that cluster,*
- *Step 4: For each k cluster, find the new cluster centers,*
- *Step 5: Repeat Step 3-5 until*
  - *Centers do not move,*
  - *No data point changes cluster,*
  - *Desired SSE is obtained.*

a	*b*	*c*	*d*	*e*	*f*	*g*	*h*
(1,3)	(3,3)	(4,3)	(5,3)	(1,2)	(4,2)	(1,1)	(2,1)



- Step 1: let k be 2
- Step 2: Randomly assign initial cluster centers, let c1=(1,1) and c2=(2,1)

a	*b*	*c*	*d*	*e*	*f*	*g*	*h*
(1,3)	(3,3)	(4,3)	(5,3)	(1,2)	(4,2)	(1,1)	(2,1)

- Step 3: (first pass) for each record find the nearest cluster center. (c1=(1,1) and c2=(2,1))

Point	Distance from c_1	Distance from c_2	Cluster Membership
a	2.00	2.24	C_1
b	2.83	2.24	C_2
c	3.61	2.83	C_2
d	4.47	3.61	C_2
e	1.00	1.41	C_1
f	3.16	2.24	C_2
g	0.00	1.00	C_1
h	1.00	0.00	C_2

$$SSE = \Sigma_{i=1}^{k} \Sigma_{p \in C_i} d(p - c_i)^2$$

$$SSE = \Sigma_{i=1}^{k} \Sigma_{p \in C_i} d(p - c_i)^2 = 2^2 + 2.24^2 + 2.83^2 + 3.61^2 + 1^2 + 2.24^2 + 0^2 + 0^2 = 36.0762$$

a	*b*	*c*	*d*	*e*	*f*	*g*	*h*
(1,3)	(3,3)	(4,3)	(5,3)	(1,2)	(4,2)	(1,1)	(2,1)

$$\frac{BCV}{WCV} = \frac{d(c_1 - c_2)}{SSE} = \frac{1}{36} = 0{,}0278$$

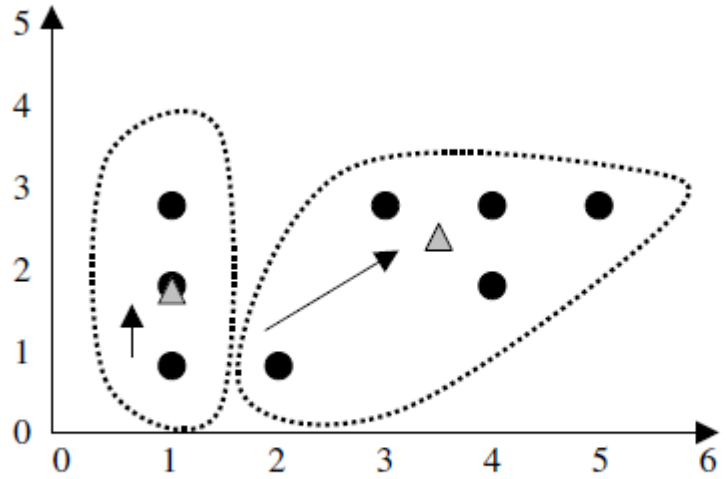
- We expect this ratio to increase with successive passes.
- Step 4: For each of the *k clusters find the cluster centroid and* update the location of each cluster center to the new value of the centroid.

$$newc_1 = \left[\left(\frac{1+1+1}{3}\right) + \left(\frac{3+2+1}{3}\right)\right] = (1,2)$$

$$newc_2 = \left[\left(\frac{3+4+5+4+2}{5}\right) + \left(\frac{3+3+3+2+1}{5}\right)\right] = (3.6,2.4)$$

a	*b*	*c*	*d*	*e*	*f*	*g*	*h*
(1,3)	(3,3)	(4,3)	(5,3)	(1,2)	(4,2)	(1,1)	(2,1)

- Step 5: repeat steps 3 and 4 until convergence.
- Step 3 (second pass) : update cluster centers c1=(1,2) and c2=(3.6,2.4). Calculate the distances between each point and updated cluster centers.



Clusters and centroids Δ after first pass through *k*-means algorithm.

a	*b*	*c*	*d*	*e*	*f*	*g*	*h*
(1,3)	(3,3)	(4,3)	(5,3)	(1,2)	(4,2)	(1,1)	(2,1)

- Step 3 (second pass) : update cluster centers c1=(1,2) and c2=(3.6,2.4). Calculate the distances between each point and updated cluster centers.

Point	Distance from c_1	Distance from c_2	Cluster Membership
a	1.00	2.67	C_1
b	2.24	0.85	C_2
c	3.16	0.72	C_2
d	4.12	1.52	C_2
e	0.00	2.63	C_1
f	3.00	0.57	C_2
g	1.00	2.95	C_1
h	1.41	2.13	C_1

$$SSE = \Sigma_{i=1}^{k} \Sigma_{p \in C_i} d(p - c_i)^2 = 1^2 + 0.85^2 + 0.72^2 + 1.52^2 + 0^2 + 0.57^2 + 1^2 + 1.41^2 = 7{,}88$$

a	*b*	*c*	*d*	*e*	*f*	*g*	*h*
(1,3)	(3,3)	(4,3)	(5,3)	(1,2)	(4,2)	(1,1)	(2,1)

$$\frac{BCV}{WCV} = \frac{d(c_1 - c_2)}{SSE} = \frac{2.63}{7.88} = 0,3338$$

- Step 4 (second pass) : For each of the *k clusters find the cluster centroid and* update the location of each cluster center to the new value of the centroid.

$$newc_1 = \left[\left(\frac{1+1+1+2}{4}\right) + \left(\frac{3+2+1+1}{4}\right)\right] = (1.25,1.75)$$

$$newc_2 = \left[\left(\frac{3+4+5+4}{4}\right) + \left(\frac{3+3+3+2}{4}\right)\right] = (4,2.75)$$

- Step 5: repeat steps 3 and 4 until convergence.

a	b	c	d	e	f	g	h
(1,3)	(3,3)	(4,3)	(5,3)	(1,2)	(4,2)	(1,1)	(2,1)

- Step 3 (third pass) : update cluster centers c1=(1.25,1.75) and c2=(4,2.75). Calculate the distances between each point and updated cluster centers.

Point	Distance from c_1	Distance from c_2	Cluster Membership
a	1.27	3.01	C_1
b	2.15	1.03	C_2
c	3.02	0.25	C_2
d	3.95	1.03	C_2
e	0.35	3.09	C_1
f	2.76	0.75	C_2
g	0.79	3.47	C_1
h	1.06	2.66	C_1

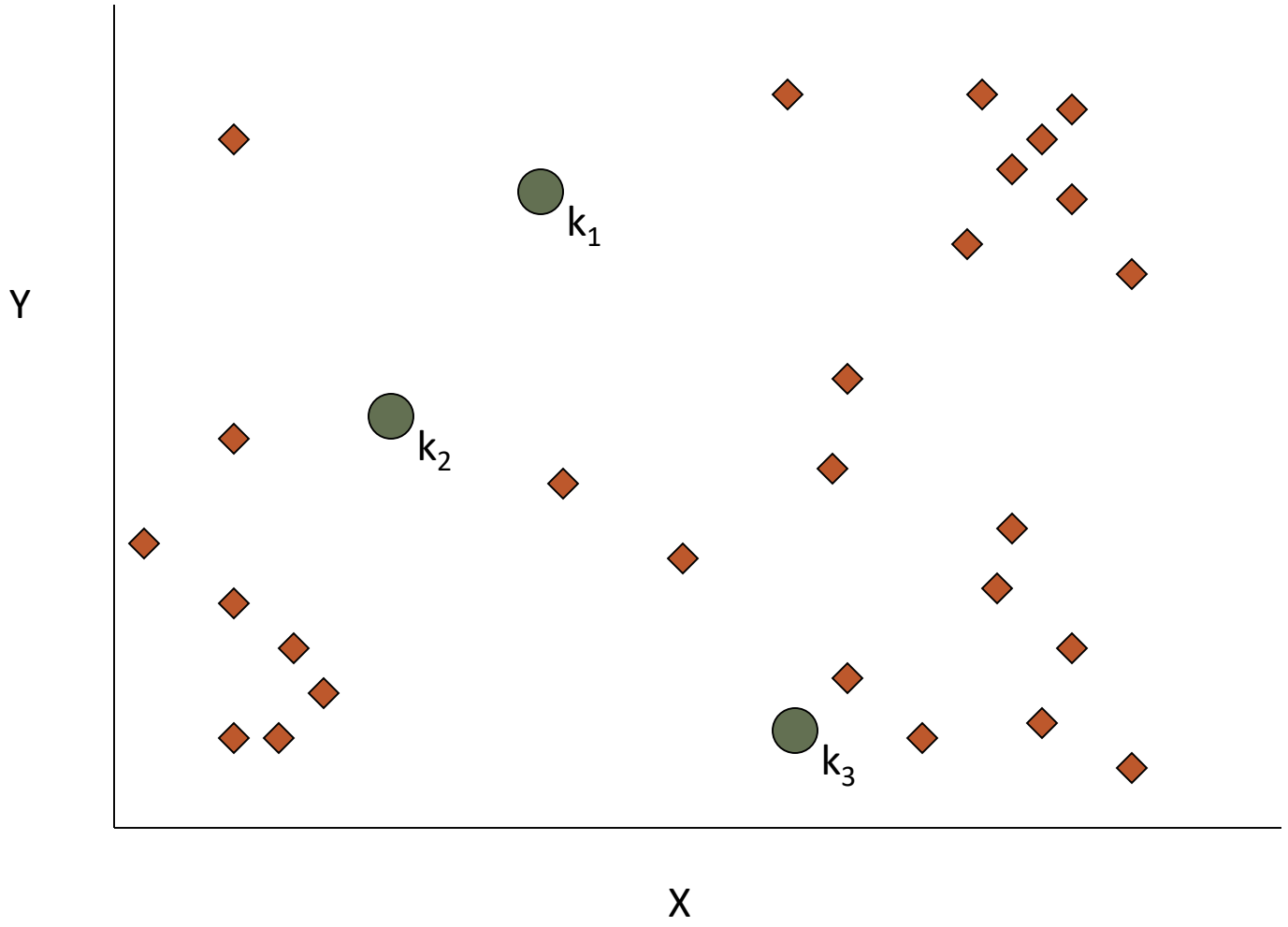
$$SSE = \Sigma_{i=1}^{k} \Sigma_{p \in C_i} d(p - c_i)^2 = 6{,}25$$

$$\frac{BCV}{WCV} = \frac{d(c_1 - c_2)}{SSE} = \frac{2.93}{6{,}25} = 0{,}4688$$

- Step 4 (third pass) : For each of the *k clusters find the cluster centroid and* update the location of each cluster center to the new value of the centroid. Since no records have shifted cluster membership, the cluster centroids therefore also remain unchanged.
- *Step 5: Repeat steps 3 and 4 until convergence or termination. Since the centroids* remain unchanged, the algorithm terminates.

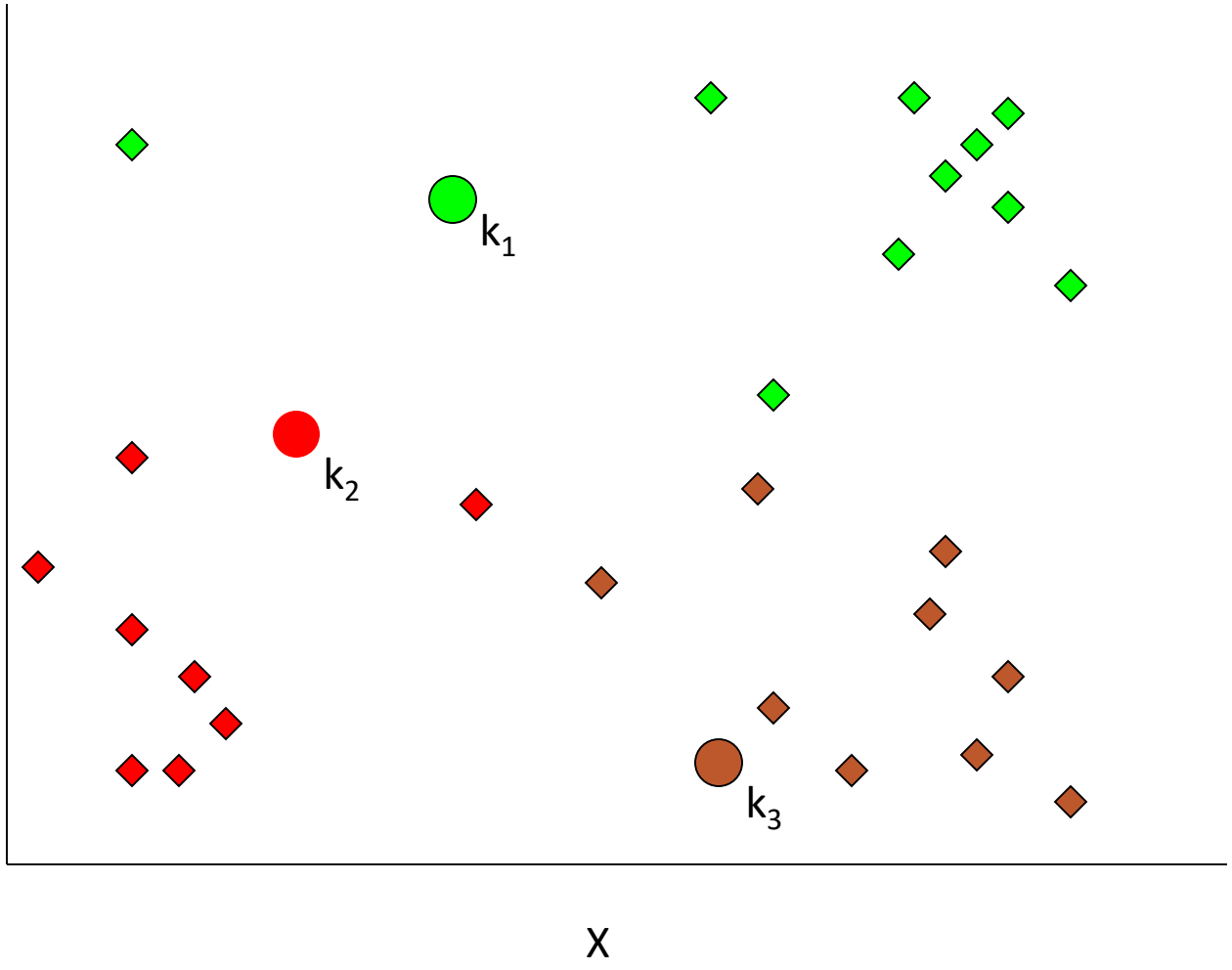
K-means example, step 1

Pick 3 initial cluster centers (randomly)



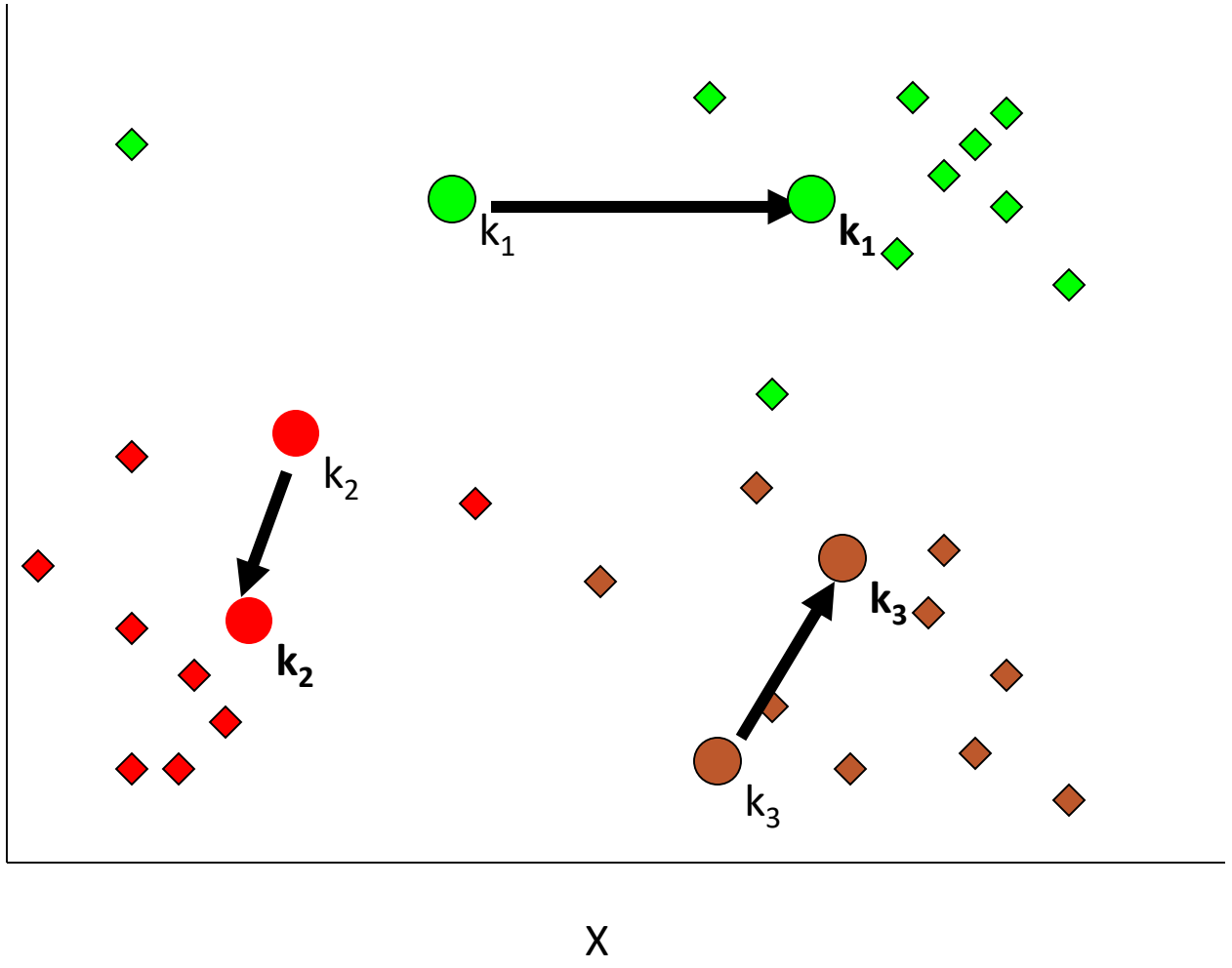
K-means example, step 2

Assign each point to the closest cluster center



K-means example, step 3

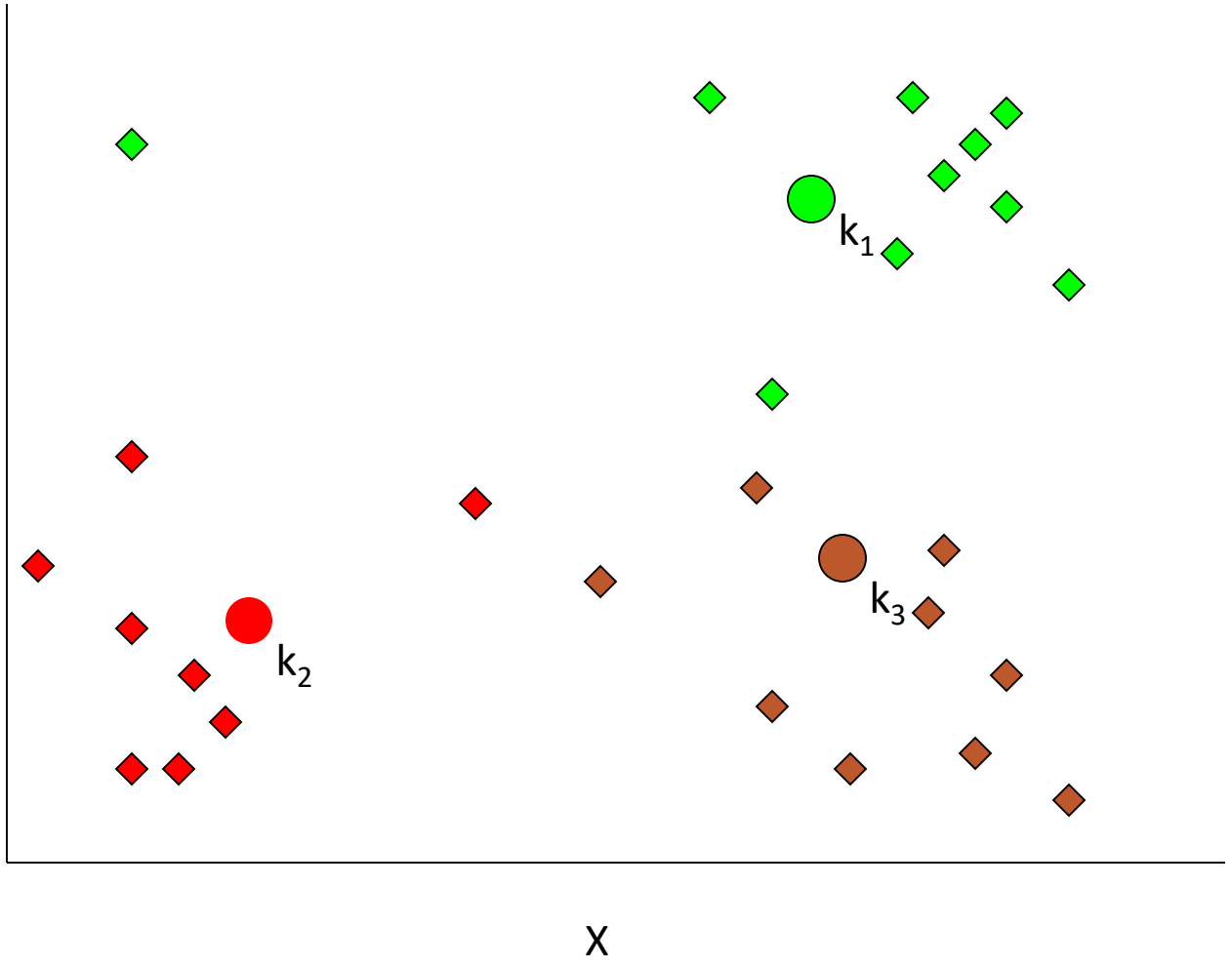
Move
each cluster center
to the mean
of each cluster



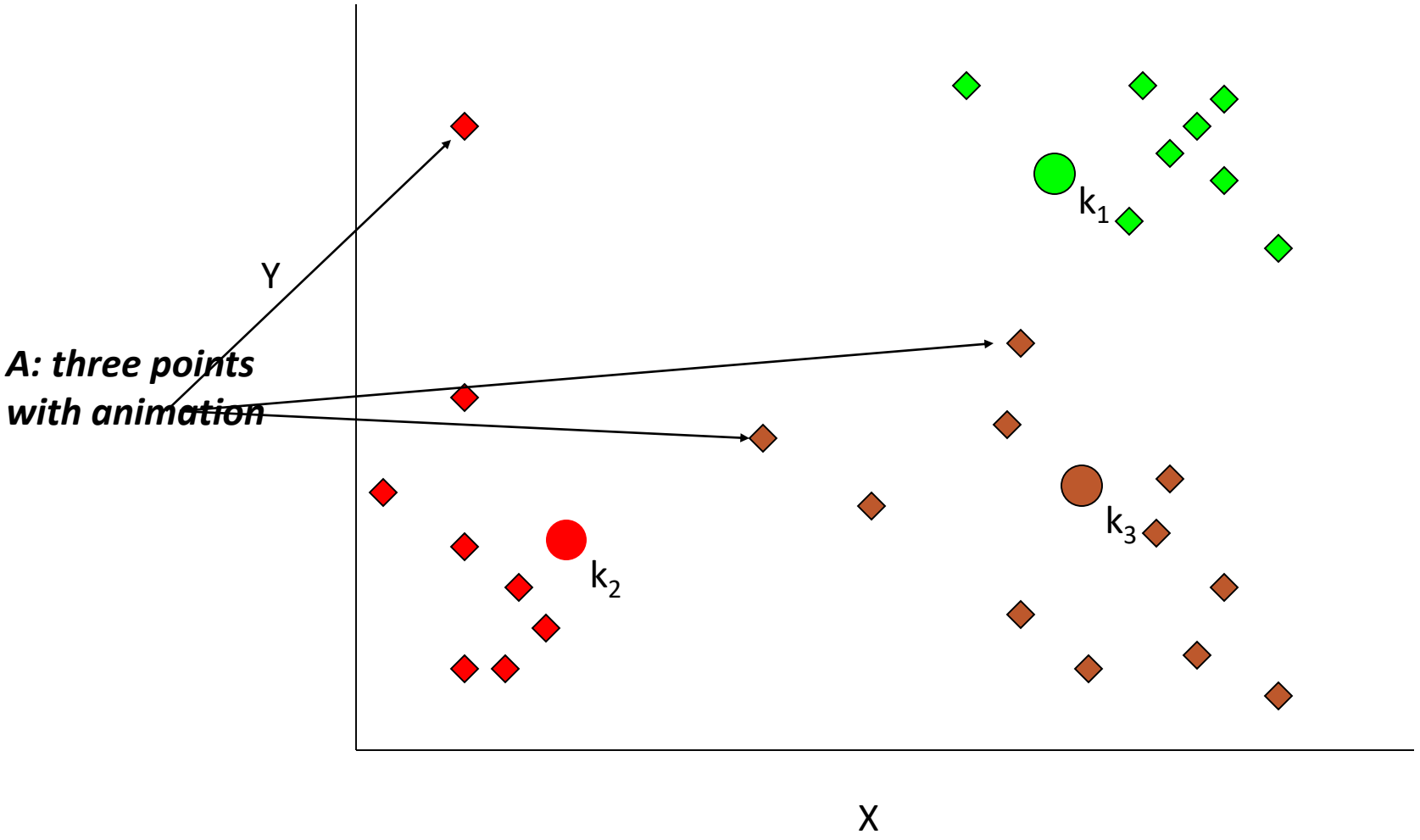
K-means example, step 4

Reassign
points
closest to a different
new cluster center

Q: Which points are reassigned?

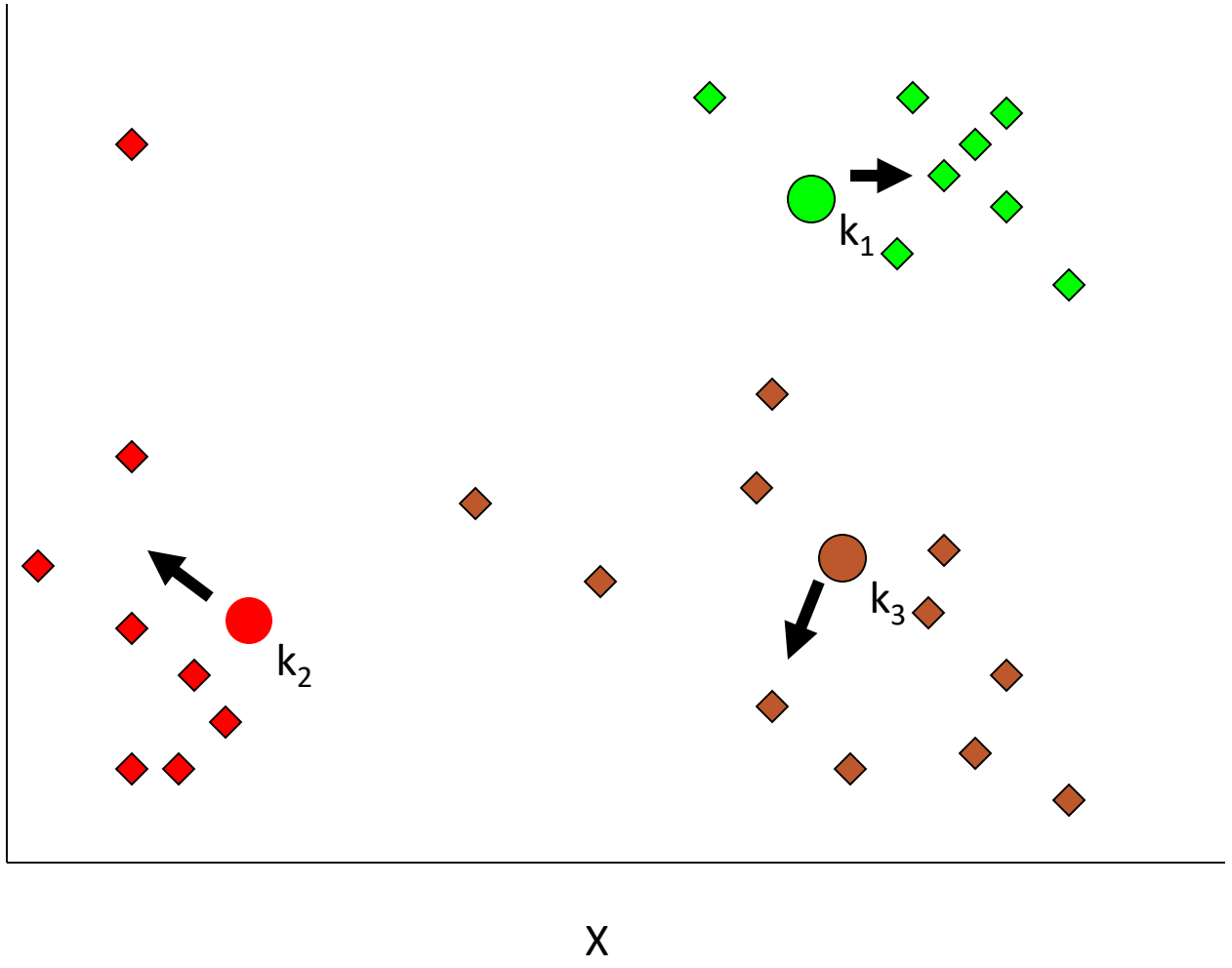


K-means example, step 4 …



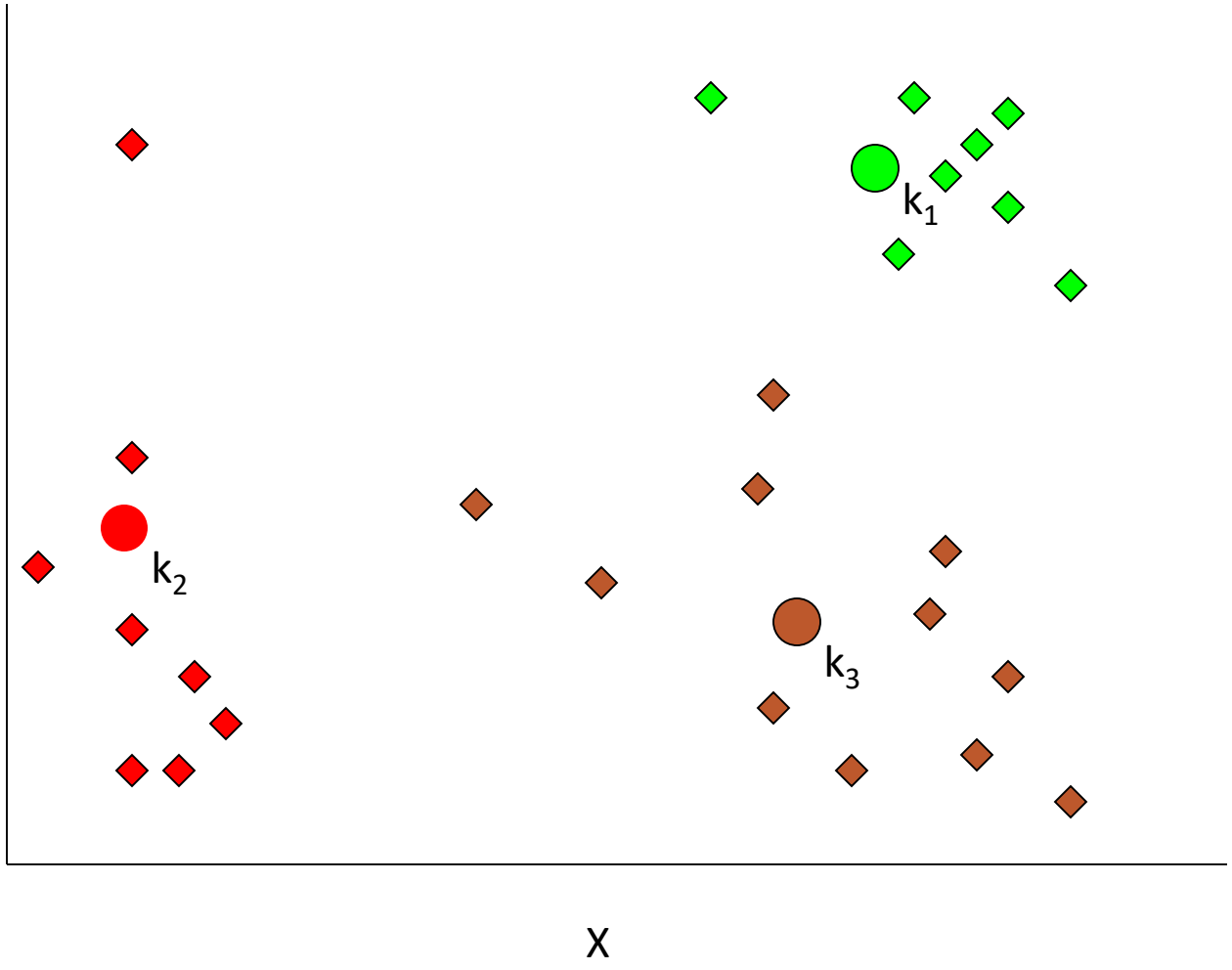
K-means example, step 4b

re-compute
cluster means



K-means example, step 5

move cluster centers to cluster means

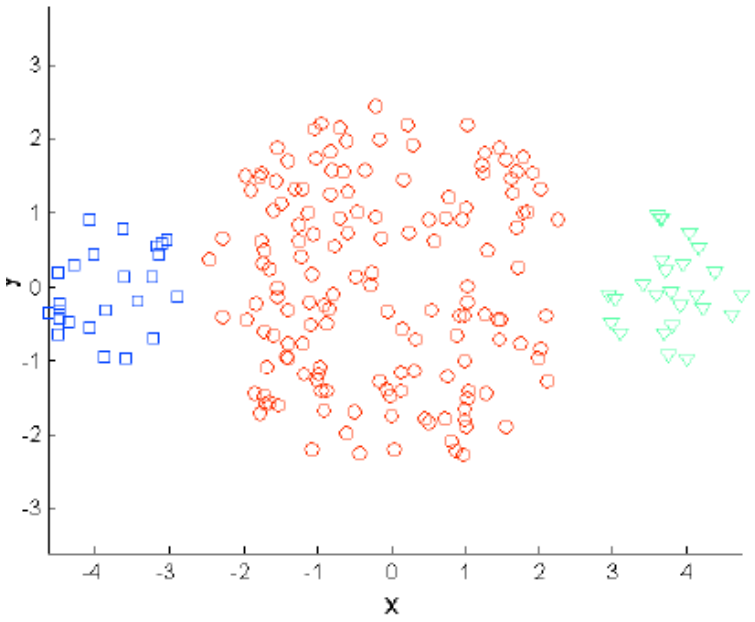


k-means Clustering

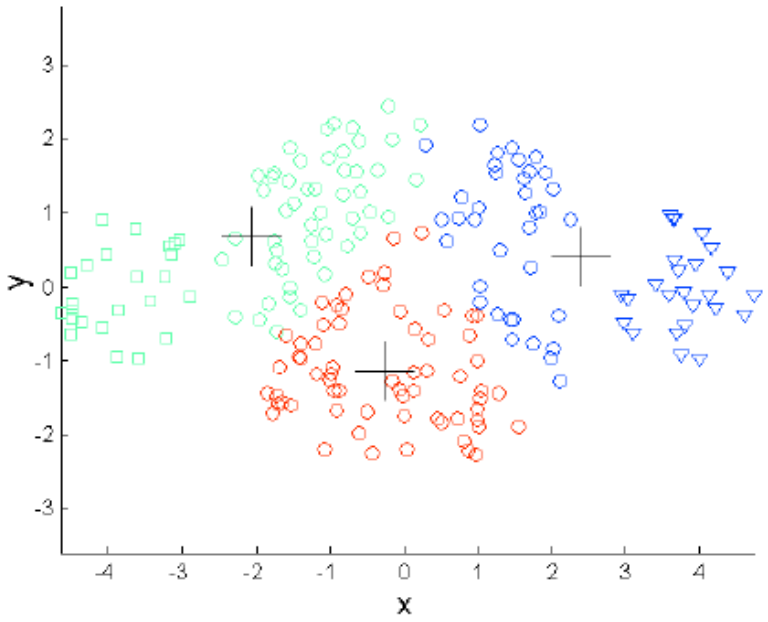
- ✓ Strength:
  - Relatively efficient and fast: *O(tkn)*
  - Easy to understand
  - Often terminates at a local optimum

- ✓ Weakness
  - Applicable only when *mean* is defined, then what about categorical data?
  - Need to specify *k,* the *number* of clusters, in advance
  - Unable to handle noisy data and *outliers*
  - Not suitable to discover clusters with *non-convex* shapes
  - Result can vary significantly depending on initial choice of centroids
  - Total steps can vary depending on initial choice of centroids

k-means Clustering

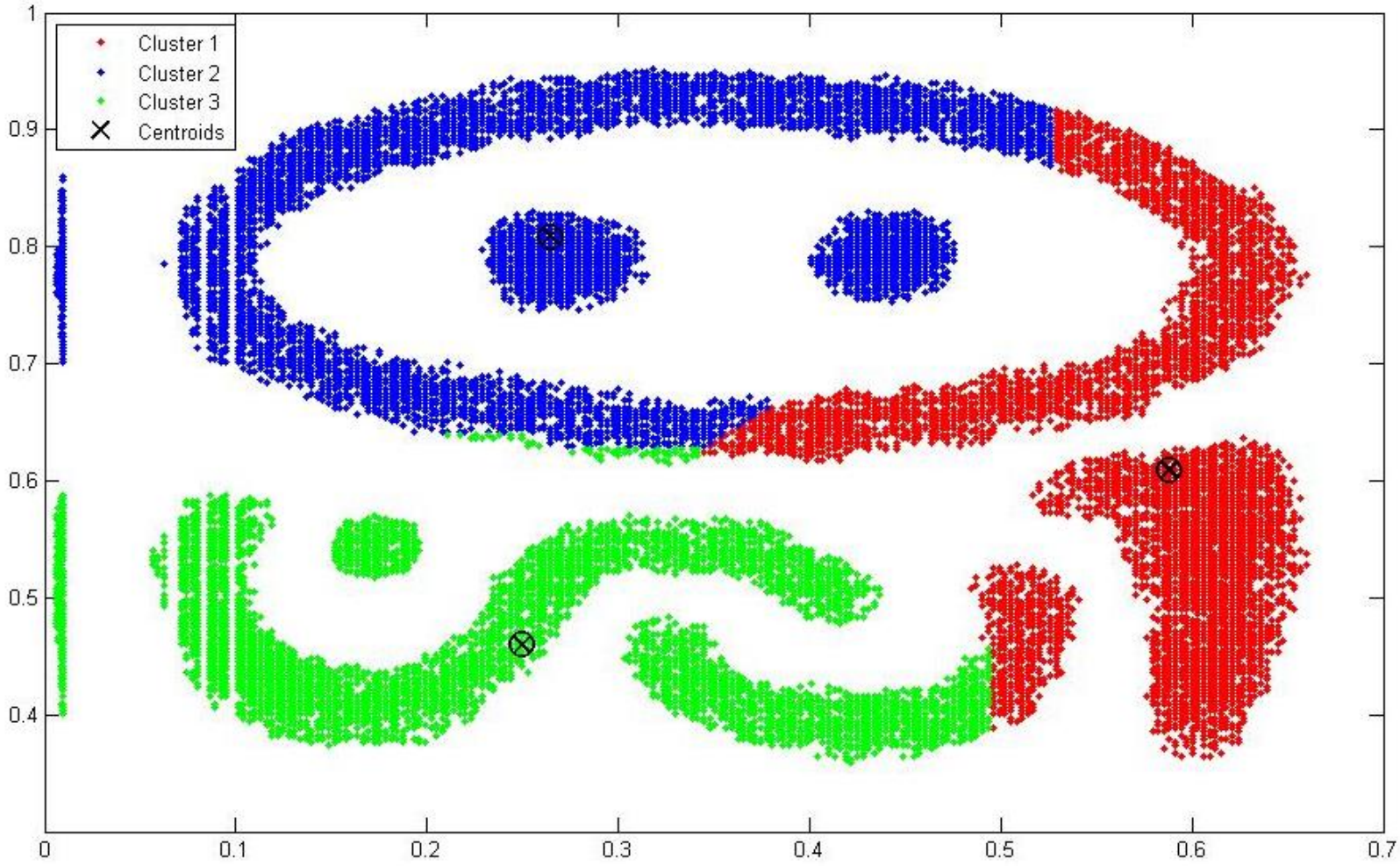


Original Points



K-means (3 Clusters)

k-means Clustering



k-means Clustering

- ✓ Alternatives
- ✓ **K-medians** – instead of mean, use medians of each cluster
  - Mean of 1, 3, 5, 7, 1009 is **205**
  - Median of 1, 3, 5, 7, 1009 is **5**
- ✓ **K-modes** – to cluster categorical data by using modes instead of means for clusters.
- ✓ **K-medoids**
  - A medoid can be defined as the object of a cluster, whose average dissimilarity to all the objects in the cluster is minimal.
  - PAM (Partitioning Around Medoids) Algorithm
- ✓ **Fuzzy c-means**
  - a method of clustering which allows one piece of data to belong to two or more clusters.

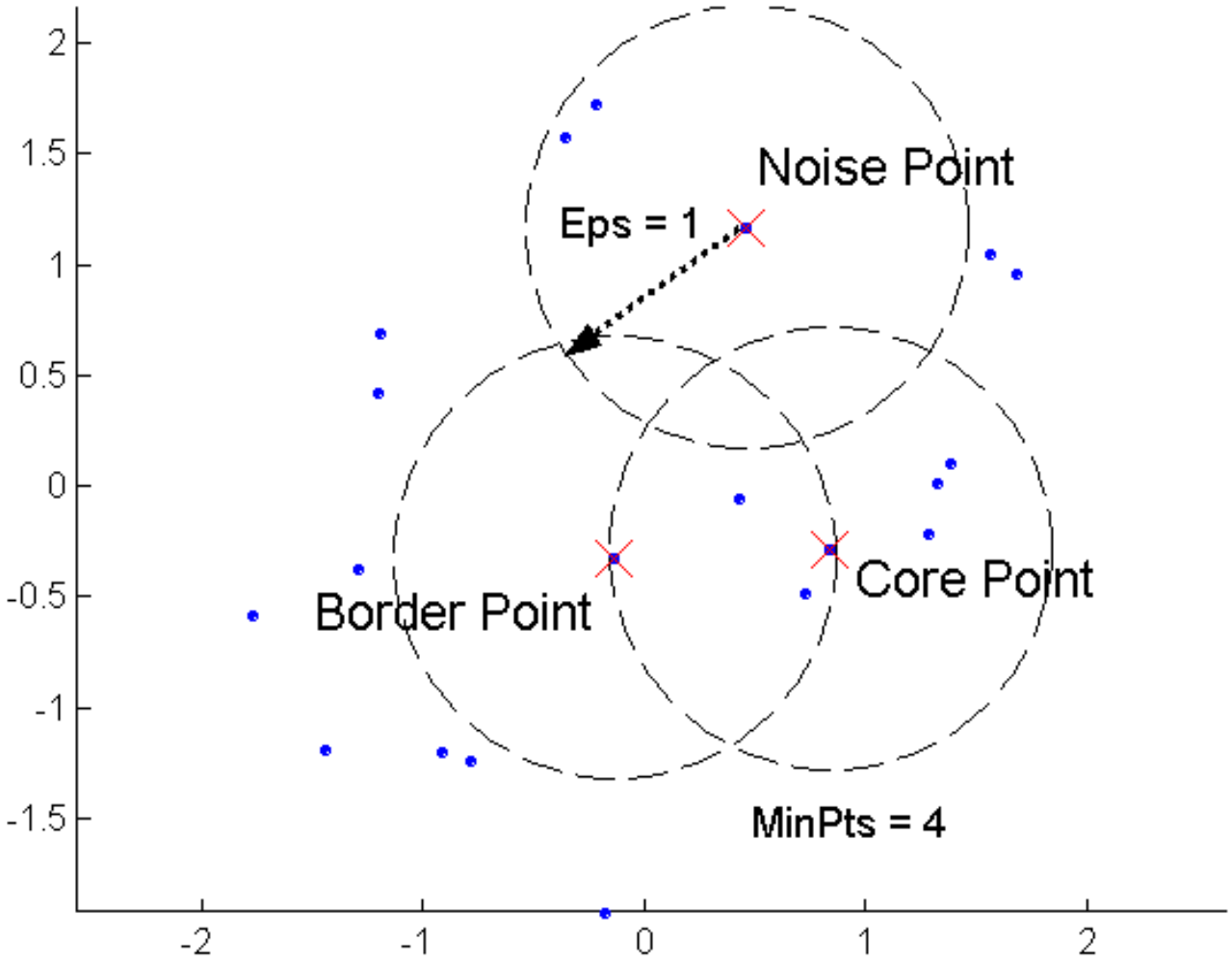
Density Based Clustering

- ✓ Clustering based on density (local cluster criterion), such as density-connected points.
- ✓ Major features:
  - Discover clusters of arbitrary shape
  - Handle noise
  - One scan
  - Need density parameters as termination condition
- ✓ Several interesting studies:
  - DBSCAN: Ester, et al. (KDD'96)
  - OPTICS: Ankerst, et al (SIGMOD'99).
  - DENCLUE: Hinneburg & D. Keim (KDD'98)

DBSCAN

- ✓ Density-Based Spatial Clustering of Applications with Noise.
  - Density = number of points within a specified radius (***Eps***)
  - A point is a ***core point*** if it has more than a specified number of points (MinPts) within Eps
    - These are points that 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



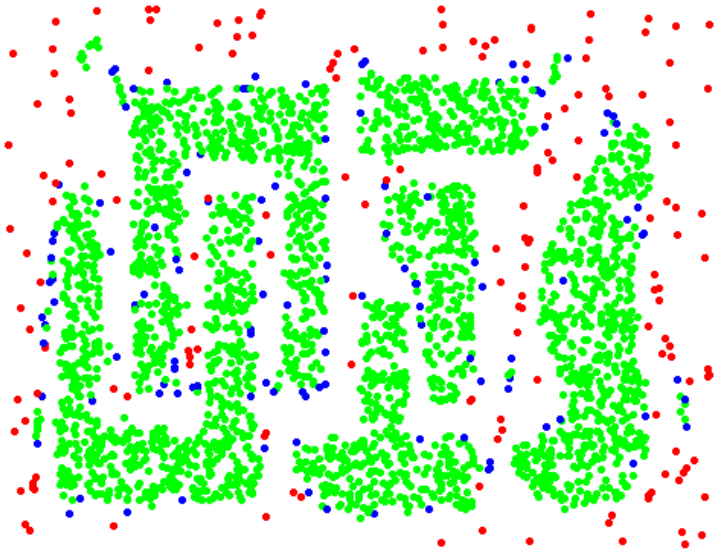
DBSCAN Algorithm

- ✓ Eliminate noise points
- ✓ Perform clustering on the remaining points

current_cluster_label ← 1
for all core points **do**
  if the core point has no cluster label **then**
    current_cluster_label ← *current_cluster_label* + 1
    Label the current core point with cluster label *current_cluster_label*
  end if
  for all points in the *Eps*-neighborhood, except i^{th} the point itself **do**
    if the point does not have a cluster label **then**
      Label the point with cluster label *current_cluster_label*
    end if
  end for
end for

DBSCAN: Core, Border and Noise Points

Original Points

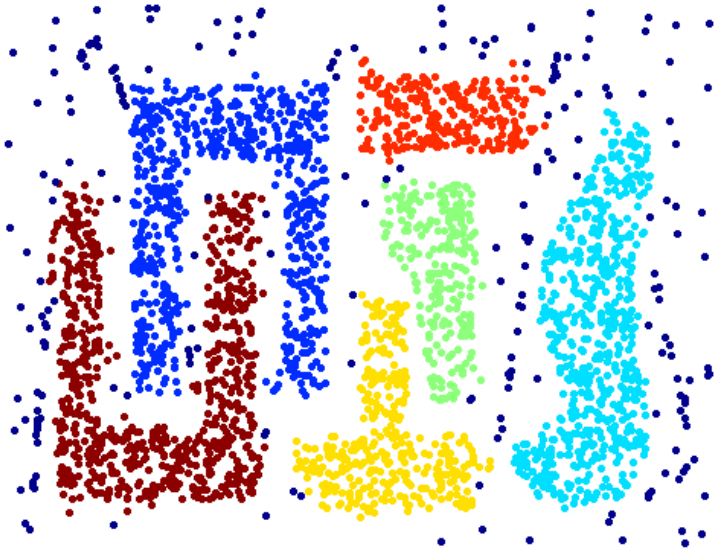


Eps = 10, MinPts = 4

Point types: core, border and noise

When DBSCAN Works Well

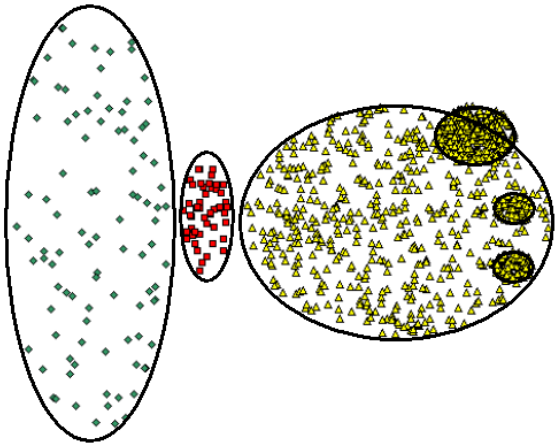
Original Points



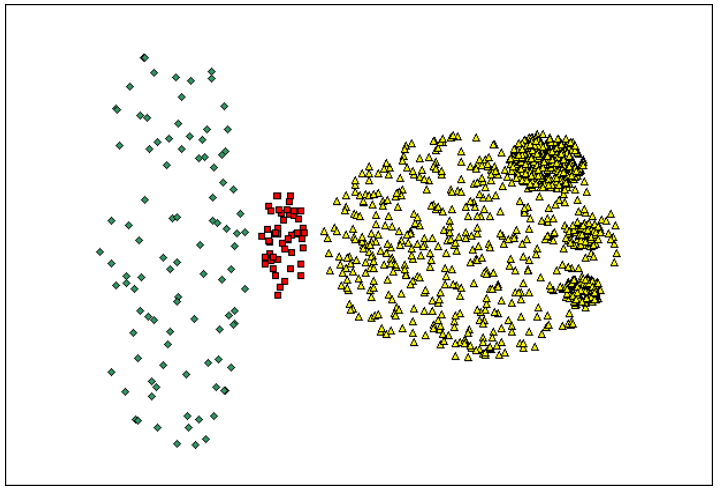
Clusters

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

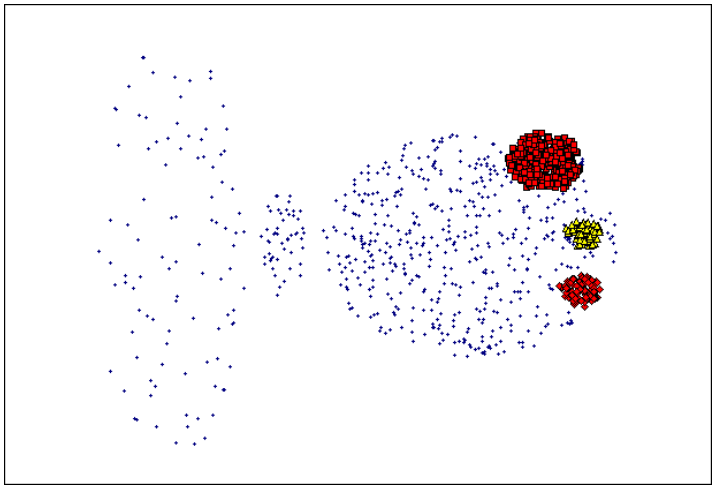
When DBSCAN Does NOT Work Well



Original Points



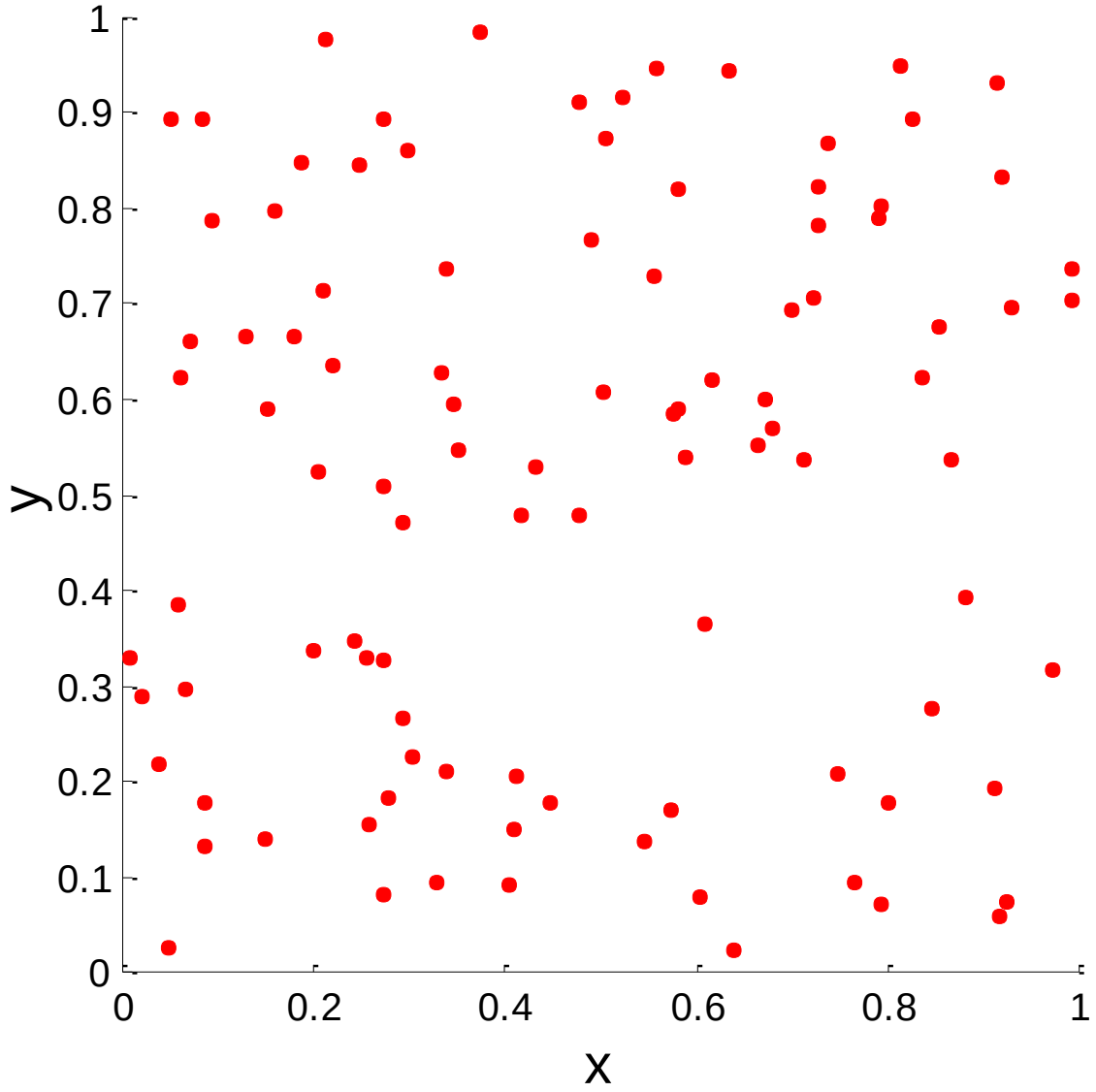
(MinPts=4, Eps=9.75).



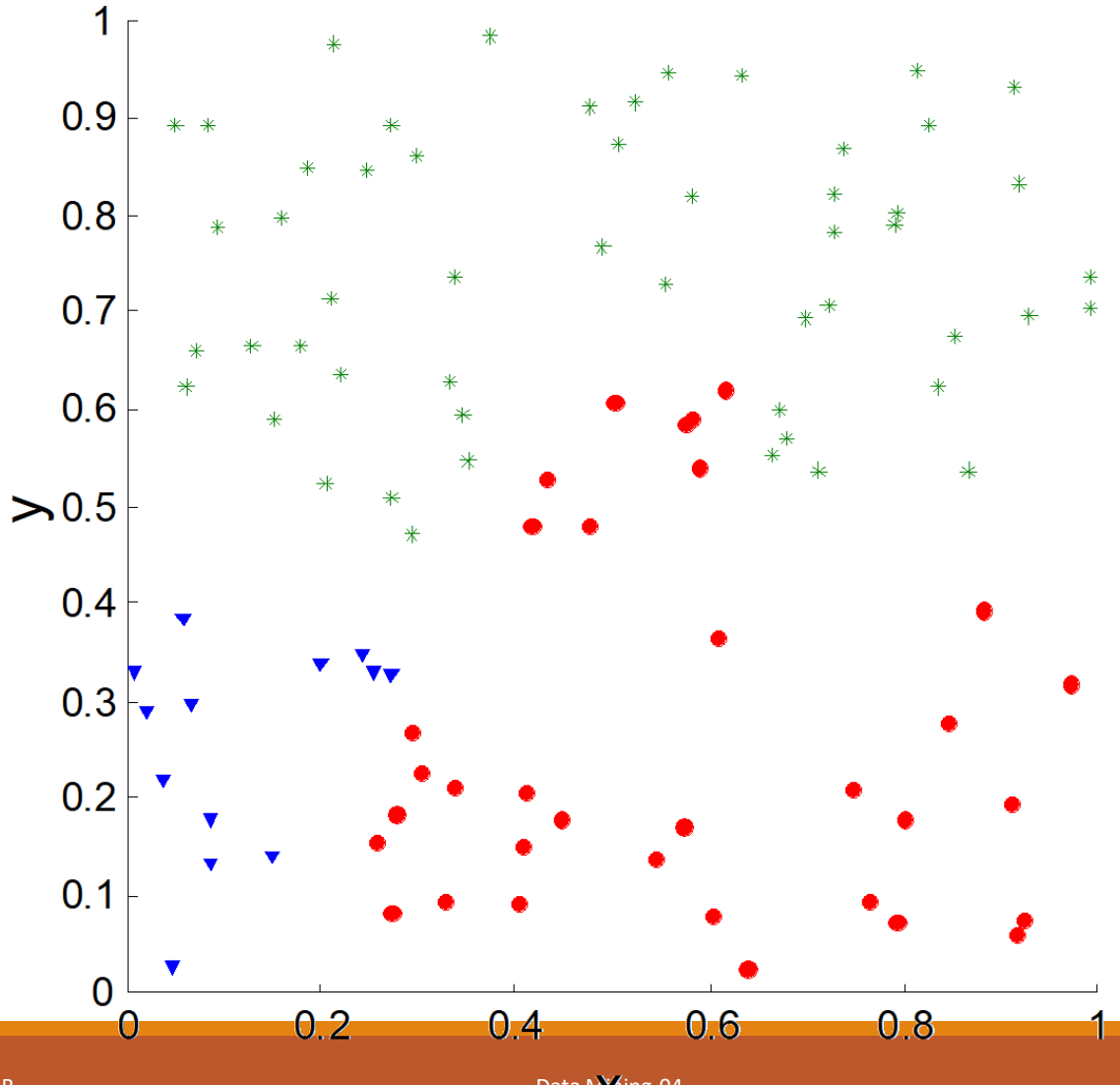
(MinPts=4, Eps=9.92)

- Varying densities
- High-dimensional data

Clusters found in Random Data

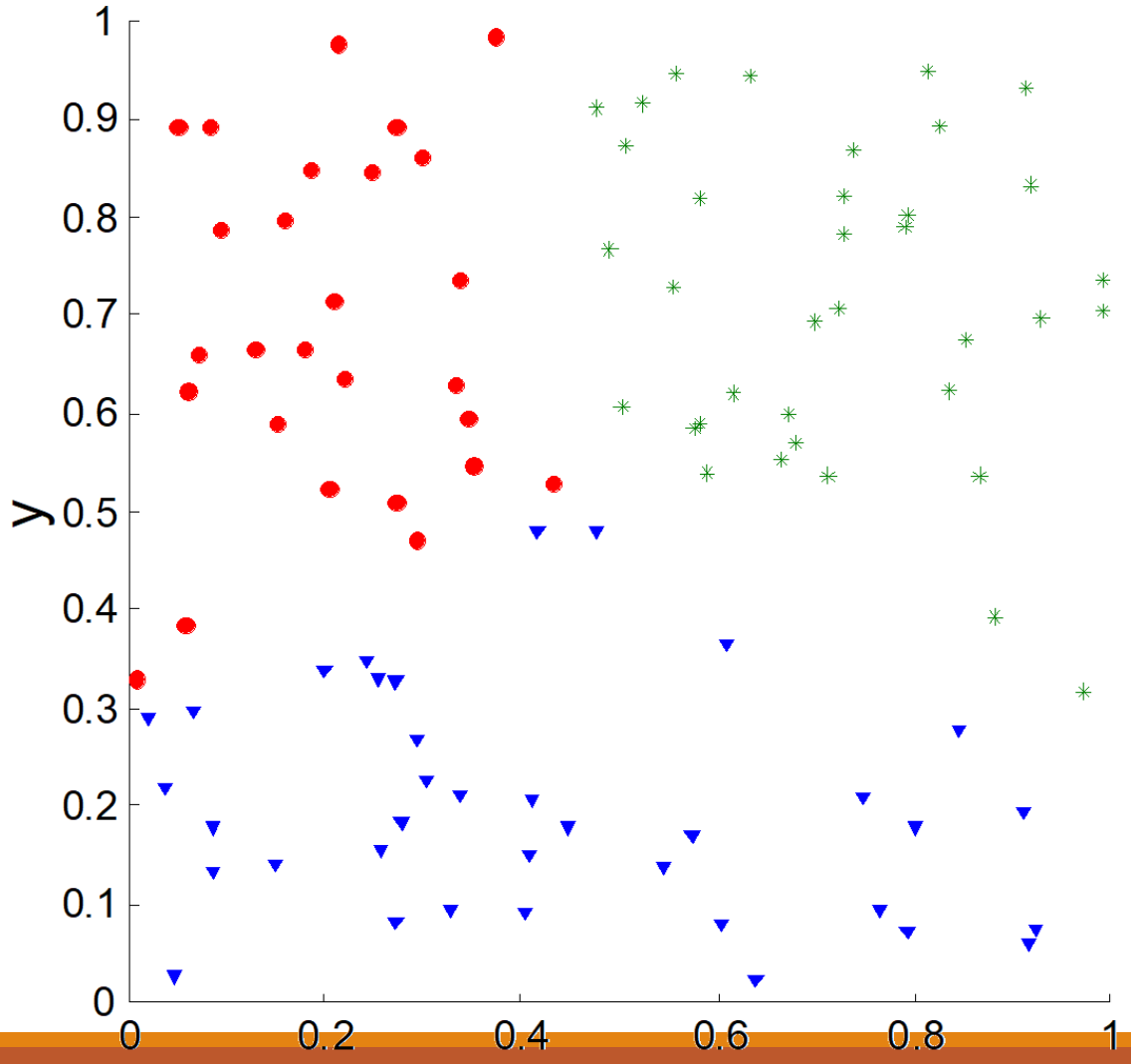


Clusters found in Random Data



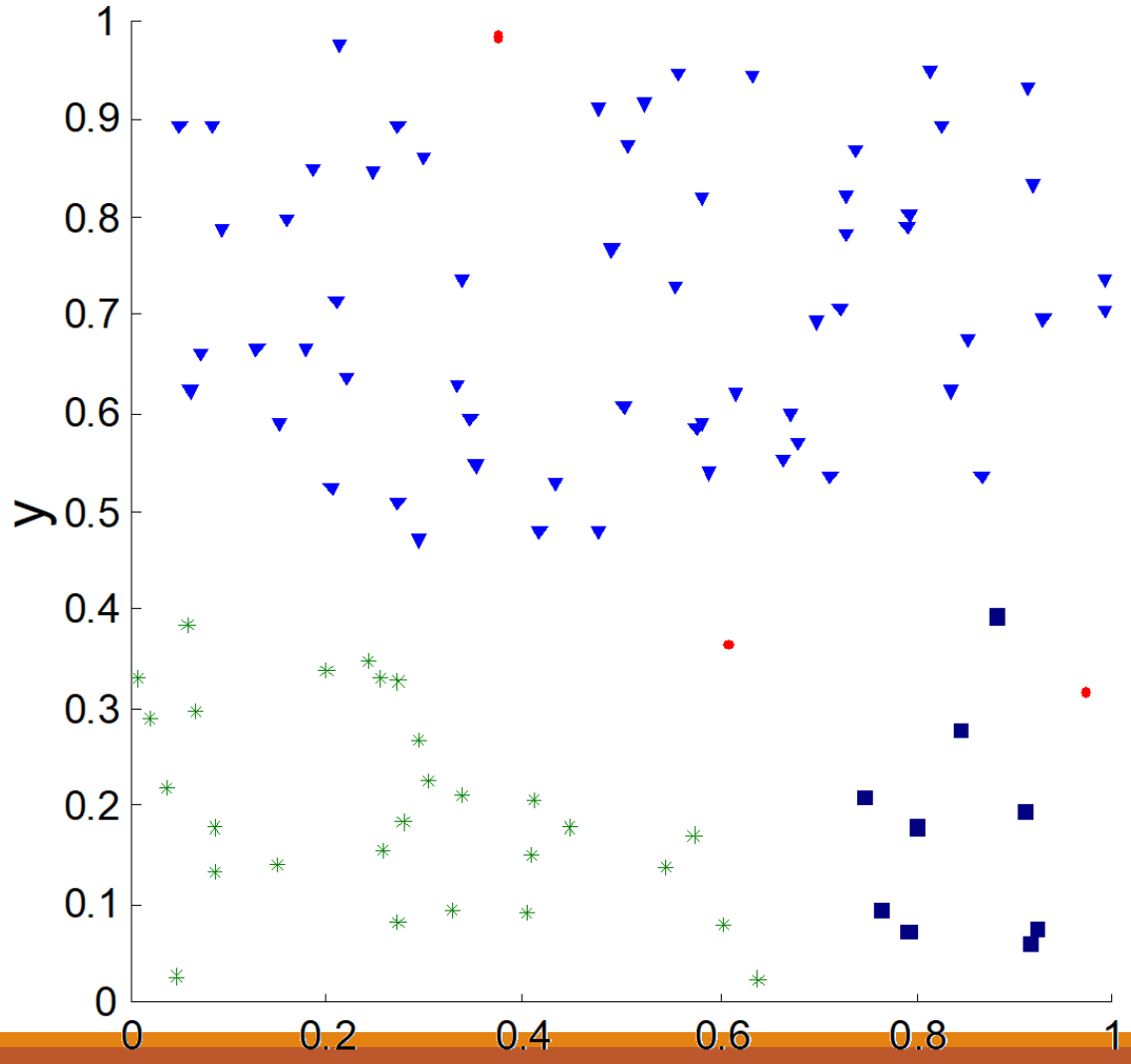
Complete Link

Clusters found in Random Data



K-means

Clusters found in Random Data



Grid Based Clustering Methods

- Simplest approach is to divide region into a number of rectangular cells of equal volume and define density as # of points the cell contains

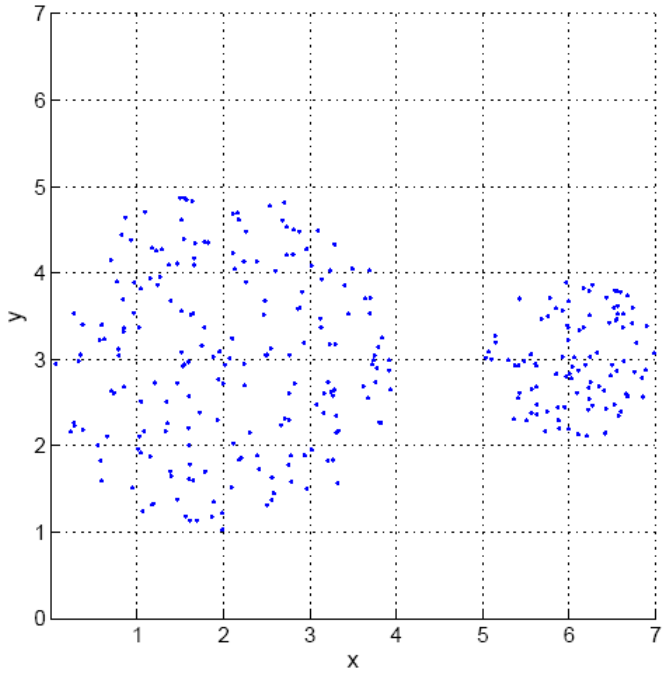


Figure 7.13. Cell-based density.

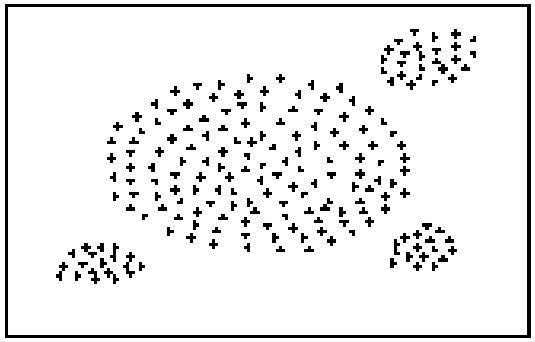
0	0	0	0	0	0	0
0	0	0	0	0	0	0
4	17	18	6	0	0	0
14	14	13	13	0	18	27
11	18	10	21	0	24	31
3	20	14	4	0	0	0
0	0	0	0	0	0	0

Table 7.6. Point counts for each grid cell.

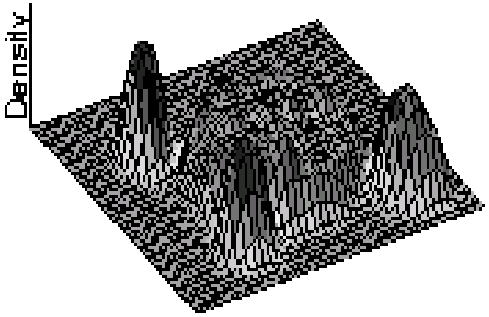
Model Based Methods

Attempt to optimize the fit between the given data and some mathematical model

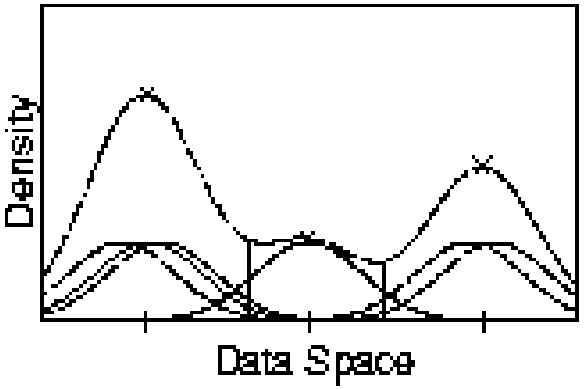
It uses statistical functions



(a) Data Set



(c) Gaussian

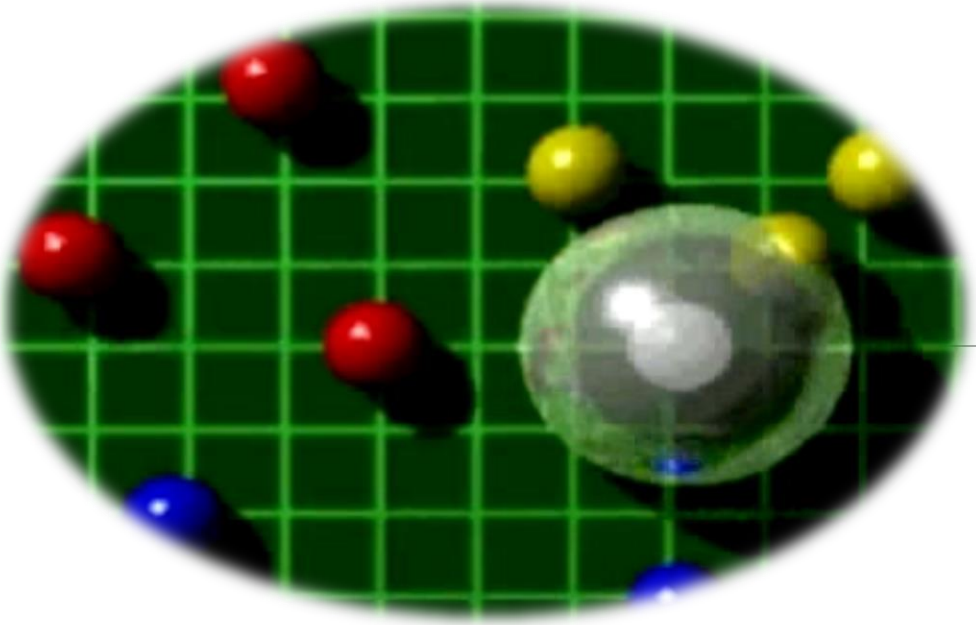


(a) $\xi = 2$

(b) $\xi = 2$

(c) $\xi = 1$

(d) $\xi = 1$



Next…

- ✓Classification
- ✓K-nearest neighborhood
- ✓Decision Trees

Exercise on SPSS Modeler

- ✓ The Iris Plant data set.
  - Iris.xls
- ✓ **Attribute Information:**
- ✓ 1. sepal length in cm
  2. sepal width in cm
  3. petal length in cm
  4. petal width in cm
  5. class:
  -- Iris Setosa
  -- Iris Versicolour
  -- Iris Virginica

Iris Setosa

Iris Virginica

Iris Versicolor

Exercise on SPSS Modeler

- ✓ Summarize the data set by the operations
  - Show in table,
  - Data Audit,
  - Summary Statistics,
  - Histograms,
  - Plot Diagram,
- ✓ Add a new field
  - Sepal-width/sepal-length,
  - Bin this new field
- ✓ Apply k-means to the data set,