Module 4: Clustering

What is Cluster Analysis?

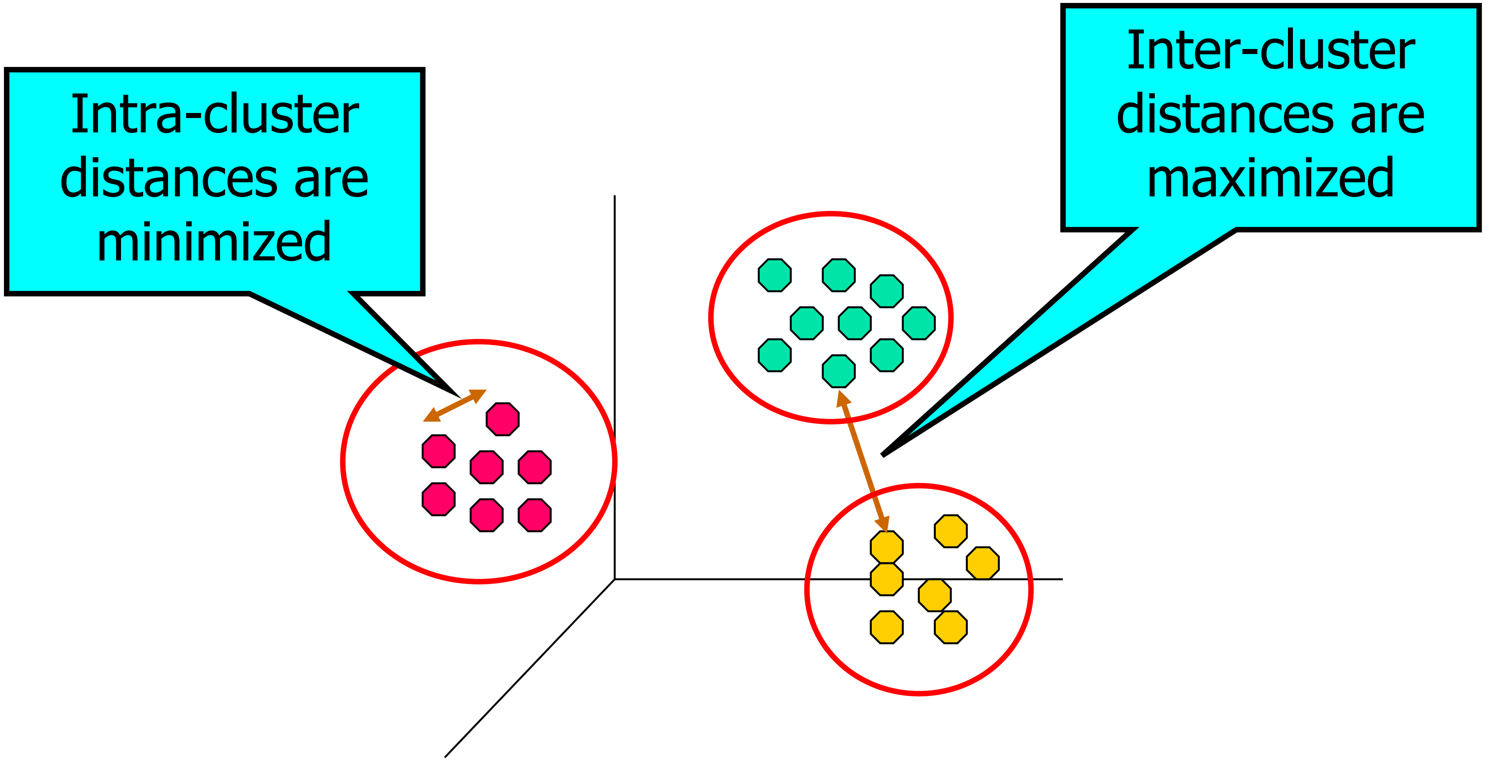
- Cluster: a collection of data objects
  - Similar to one another within the same cluster
  - Dissimilar to the objects in other clusters
- Cluster analysis
  - Finding similarities between data according to the characteristics found in the data and grouping similar data objects into clusters
- Unsupervised learning: no predefined classes
- Typical applications
  - As a stand-alone tool to get insight into data distribution
  - As a preprocessing step for other algorithms

Examples of Clustering Applications

- Marketing: Help marketers discover distinct groups in their customer bases, and then use this knowledge to develop targeted marketing programs
- Land use: Identification of areas of similar land use in an earth observation database
- Insurance: Identifying groups of motor insurance policy holders with a high average claim cost

What is Cluster Analysis?

- Finding groups of objects such that the objects in a group will be similar (or related) to one another and different from (or unrelated to) the objects in other groups



Quality: What Is Good Clustering?

- A good clustering method will produce high quality clusters with
  - high intra-class similarity
  - low inter-class similarity
- The quality of a clustering result depends on both the similarity measure used by the method and its implementation

Measure the Quality of Clustering

- Dissimilarity/Similarity metric: Similarity is expressed in terms of a distance function, typically metric: $d(i, j)$
- There is a separate "quality" function that measures the "goodness" of a cluster.
- The definitions of distance functions are usually very different for interval-scaled, boolean, categorical, ordinal ratio, and vector variables.

Requirements of Clustering in Data Mining

- Scalability
- Ability to deal with different types of attributes
- Discovery of clusters with arbitrary shape
- Minimal requirements for domain knowledge to determine input parameters
- Able to deal with noise and outliers
- Incremental clustering & Insensitive to order of input records
- Interpretability and usability

Types of Data in Cluster Analysis

- Data matrix
  - (n objects * p variables)

$$\begin{bmatrix} x_{11} & \cdots & x_{1f} & \cdots & x_{1p} \\ \cdots & \cdots & \cdots & \cdots & \cdots \\ x_{i1} & \cdots & x_{if} & \cdots & x_{ip} \\ \cdots & \cdots & \cdots & \cdots & \cdots \\ x_{n1} & \cdots & x_{nf} & \cdots & x_{np} \end{bmatrix}$$

- Dissimilarity matrix
  - (object by object)
  - d(i,j)- dissimilarity between object i and j

$$\begin{bmatrix} 0 & & & & \\ d(2,1) & 0 & & & \\ d(3,1) & d(3,2) & 0 & & \\ : & : & : & & \\ d(n,1) & d(n,2) & \cdots & \cdots & 0 \end{bmatrix}$$

- where $d(i, j)$ is the measured difference or dissimilarity between objects i and j. In general, $d(i, j)$ is a nonnegative number that is close to 0 when objects i and j are highly similar or "near" each other, and becomes larger the more they differ.

Type of data in clustering analysis

- Interval-scaled variables
- Binary variables
- Nominal, ordinal, and ratio variables
- Variables of mixed types

Interval-valued variables

- Continuous measurements of roughly linear scale
- e.g. height, weight, weather temperature
- **Standardization of Numerical data**
  - Calculate the mean absolute deviation:

$$s_f = \frac{1}{n}(|x_{1f} - m_f| + |x_{2f} - m_f| + ... + |x_{nf} - m_f|)$$

where $m_f = \frac{1}{n}(x_{1f} + x_{2f} + ... + x_{nf})$.

  - Calculate the standardized measurement (*z-score*)

$$z_{if} = \frac{x_{if} - m_f}{s_f}$$

- Using mean absolute deviation is more robust to outliers than standard deviation

Similarity and Dissimilarity Between Objects

- Distances are normally used to measure the similarity or dissimilarity between two data objects
- the dissimilarity(or similarity) between the objects described by interval-scaled variables is typically computed based on the distance between each pair of objects. The most popular distance measure **is Euclidean distance**, which is defined as

$$d(i,j)=\sqrt{(|x_{i_1}-x_{j_1}|^2+|x_{i_2}-x_{j_2}|^2+...+|x_{i_p}-x_{j_p}|^2)}$$

Similarity and Dissimilarity Between Objects (Cont.)

- **Manhattan distance,** defined as

$$d(i,j) = |x_{i_1} - x_{j_1}| + |x_{i_2} - x_{j_2}| + \ldots + |x_{i_p} - x_{j_p}|$$

- Properties
  - $d(i,j) \geq 0$
  - $d(i,i) = 0$
  - $d(i,j) = d(j,i)$

Similarity and Dissimilarity Between Objects

- **Minkowski** distance is a generalization of both **Euclidean** distance and **Manhattan** distance. It is defined as

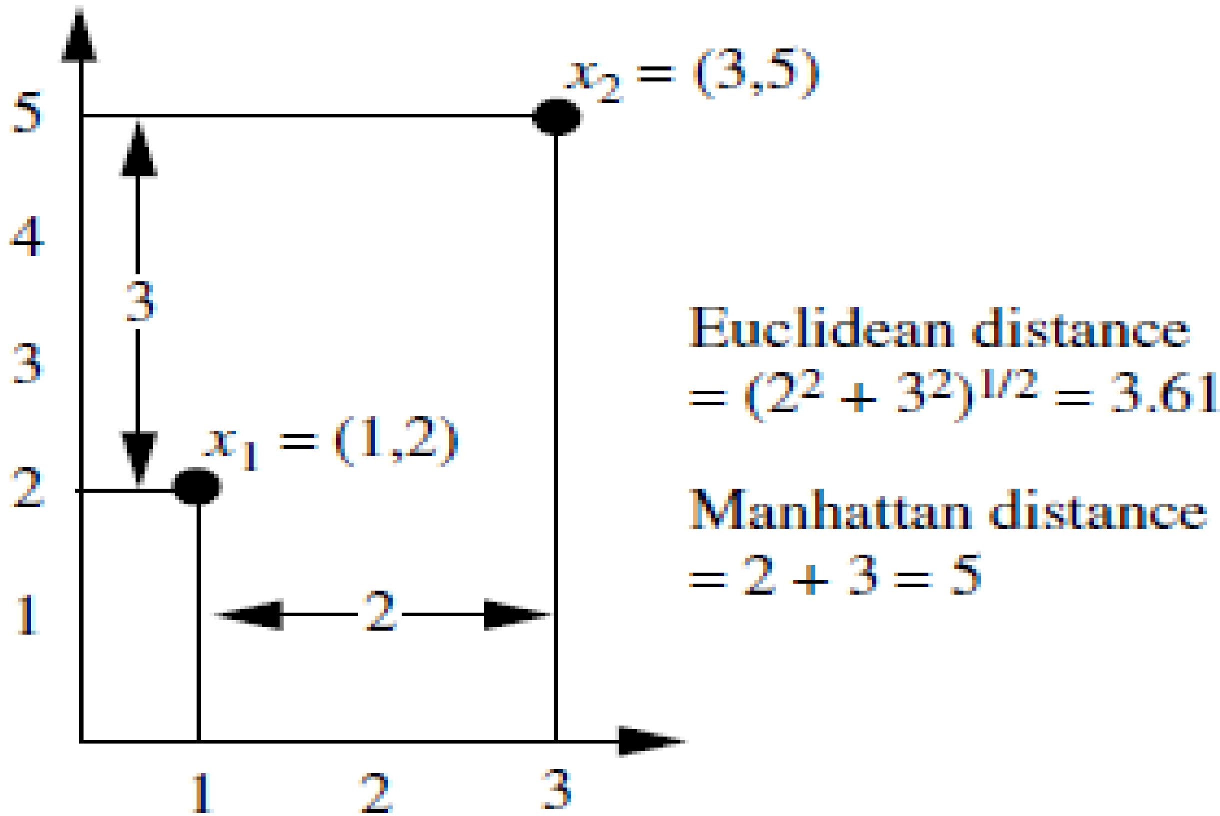
$$d(i,j) = \sqrt[q]{(|x_{i_1} - x_{j_1}|^q + |x_{i_2} - x_{j_2}|^q + ... + |x_{i_p} - x_{j_p}|^q)}$$

where $i = (x_{i1}, x_{i2}, \ldots, x_{ip})$ and $j = (x_{j1}, x_{j2}, \ldots, x_{jp})$ are two p-dimensional data objects, and q is a positive integer

- It represents the Manhattan distance when $q = 1$ and Euclidean distance when $q = 2$

Example

- Let ***x*1** = (1, 2) and ***x*2** = (3, 5) represent two objects.



Binary Variables

- A binary variable has only two states: 0 or 1, where 0 means that the variable is absent, and 1 means that it is present.
-

Binary Variables

- If all binary variable are thought of as having the same weight, then 2-by-2 contingency table of
  - where **a** is the number of variables that equal 1 for both objects i and j,
  - **b** is the number of variables that equal 1 for object i but that are 0 for object j,
  - **c** is the number of variables that equal 0 for object i but equal 1 for object j, and
  - **d** is the number of variables that equal 0 for both objects i and j. The total number of variables is p,
  - where p = a+b+c+d.

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

Binary Variables

- ***symmetric and asymmetric binary variables***
- A binary variable is symmetric if both of its states are equally valuable and carry the same weight; that is, there is no preference on which outcome should be coded as 0 or 1.
- One such example could be the attribute *gender* having the states *male* and *female*.
- Dissimilarity that is based on symmetric binary variables is called symmetric binary dissimilarity. Its dissimilarity (or distance) measure, defined as

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

Binary Variables

- ***symmetric and asymmetric binary variables***
- A binary variable is asymmetric if the outcomes of the states are not equally important, such as the *positive* and *negative* outcomes of a disease *test*.
- Dissimilarity based on such variables is called **asymmetric binary dissimilarity**, where the number of negative matches, *d*, is considered unimportant and thus is ignored in the computation.

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

- Jaccard coefficient : we can measure the distance between two binary variables based on the notion of *similarity* instead of *dissimilarity*. The asymmetric binary similarity between the objects *i* and *j*, or *sim*(*i*, *j*), can be computed as,

$$sim_{Jaccard}(i, j) = \frac{a}{a+b+c} = 1 - \mathrm{d(i, j)}.$$

Binary Variables

- A contingency table for binary data

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

- Distance measure for symmetric binary variables:

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

- Distance measure for asymmetric binary variables:

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

- Jaccard coefficient (*similarity* measure for *asymmetric* binary variables):

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

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

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

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

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

ID	Disciplinary failure	Social drinker	Social smoker
1	0	0	0
2	0	1	0
3	1	1	0
4	0	1	1
5	0	0	0
6	0	1	0

Nominal Variables(categorical)

- A generalization of the binary variable in that it can take more than 2 states, e.g., red, yellow, blue, green
- Method 1: Simple matching
  - *m*: # of matches, *p*: total # of variables

$$d(i, j) = \frac{p - m}{p}$$

Example

object identifier	test-1 (categorical)	test-2 (ordinal)	test-3 (ratio-scaled)
1	code-A	excellent	445
2	code-B	fair	22
3	code-C	good	164
4	code-A	excellent	1,210

$$\begin{bmatrix} 0 & & & \\ d(2,1) & 0 & & \\ d(3,1) & d(3,2) & 0 & \\ d(4,1) & d(4,2) & d(4,3) & 0 \end{bmatrix}$$

- one categorical variable *test-1*, we set $p = 1$ in so that $d(i, j)$ evaluates to 0 if objects i and j match,
- and 1 if the objects differ.

$$\begin{bmatrix} 0 & & & \\ 1 & 0 & & \\ 1 & 1 & 0 & \\ 0 & 1 & 1 & 0 \end{bmatrix}$$

Ordinal Variables

- A discrete ordinal variable resembles a categorical variable, except that the M states of the ordinal value are ordered in a meaningful sequence.
- An ordinal variable can be discrete or continuous
- Order is important, e.g., rank
- Can be treated like interval-scaled
  - replace x_{if} by their rank $r_{if} \in \{1, \ldots, M_f\}$
  - Since each ordinal variable can have a different number of states, it is often necessary to map the range of each variable onto [0.0,1.0] so that each variable has equal weight. This can be achieved by replacing the rank r_{if} of the i^{th} object in the f^{th} variable by

$$z_{if} = \frac{r_{if} - 1}{M_f - 1}$$

  - compute the dissimilarity using methods for interval-scaled variables

Example

object identifier	test-1 (categorical)	test-2 (ordinal)	test-3 (ratio-scaled)
1	code-A	excellent	445
2	code-B	fair	22
3	code-C	good	164
4	code-A	excellent	1,210

$$\begin{bmatrix} 0 & & & \\ 1 & 0 & & \\ 0.5 & 0.5 & 0 & \\ 0 & 1.0 & 0.5 & 0 \end{bmatrix}$$

- There are three states for *test*-2, namely *fair*, *good*, and *excellent*, *Mf* =3.
- For step 1, if we replace each value for *test*-2 by its rank, the four objects are assigned the ranks 3, 1, 2, and 3, respectively.
- Step 2 normalizes the ranking by mapping rank 1 to 0.0, rank 2 to 0.5, and rank 3 to 1.0.
- For step 3, we can use the Euclidean distance

Ratio-Scaled Variables

- Ratio-scaled variable: a positive measurement on a nonlinear scale, approximately at exponential scale, such as Ae^{Bt} or Ae^{-Bt}
- Methods:
  - treat them like interval-scaled variables—*not a good choice!* (why?—the scale can be distorted)
  - apply logarithmic transformation

$$y_{if} = log(x_{if})$$

  - treat them as continuous ordinal data treat their rank as interval-scaled

Example

object identifier	*test-1 (categorical)*	*test-2 (ordinal)*	*test-3 (ratio-scaled)*
1	code-A	excellent	445
2	code-B	fair	22
3	code-C	good	164
4	code-A	excellent	1,210

$$\begin{bmatrix} 0 & & & \\ 1.31 & 0 & & \\ 0.44 & 0.87 & 0 & \\ 0.43 & 1.74 & 0.87 & 0 \end{bmatrix}$$

- Taking the *log* of *test-3* results in the values 2.65, 1.34, 2.21, and 3.08 for the objects 1 to 4, respectively.
- Using the Euclidean distance on the transformed values, we obtain the dissimilarity matrix.

1. Given the following measurements for the variable *age*:

18, 22, 25, 42, 28, 43, 33, 35, 56, 28,

standardize the variable by the following:

(a) Compute the mean absolute deviation of *age*.

(b) Compute the z-score for the first four measurements.

7.3

2. Given two objects represented by the tuples (22, 1, 42, 10) and (20, 0, 36, 8):

(a) Compute the *Euclidean distance* between the two objects.

(b) Compute the *Manhattan distance* between the two objects.

(c) Compute the *Minkowski distance* between the two objects, using $q = 3$.

Vector Objects

- Vector objects: keywords in documents, gene features in micro-arrays, etc.
- Broad applications: information retrieval, biologic taxonomy, etc.
- Cosine measure

```
Cos(x, y) = x . y / ||x|| * ||y||
```

where,
•**x . y** = product (dot) of the vectors ‘x’ and ‘y’.
•||**x**|| and ||**y**|| = length of the two vectors ‘x’ and ‘y’.
•||**x**|| * ||**y**|| = cross product of the two vectors ‘x’ and ‘y’.

- Let $\boldsymbol{x}$ = (1, 1, 0, 0) and $\boldsymbol{y}$ = (0, 1, 1, 0). Similarity between $\boldsymbol{x}$ and $\boldsymbol{y}$ is

$$s(x, y) = \frac{(0+1+0+0)}{\sqrt{2}\sqrt{2}} = 0.5$$

- The ‘x’ vector has values, **x = { 3, 2, 0, 5 }**
  The ‘y’ vector has values, **y = { 1, 0, 0, 0 }**

x . y = 3*1 + 2*0 + 0*0 + 5*0 = 3

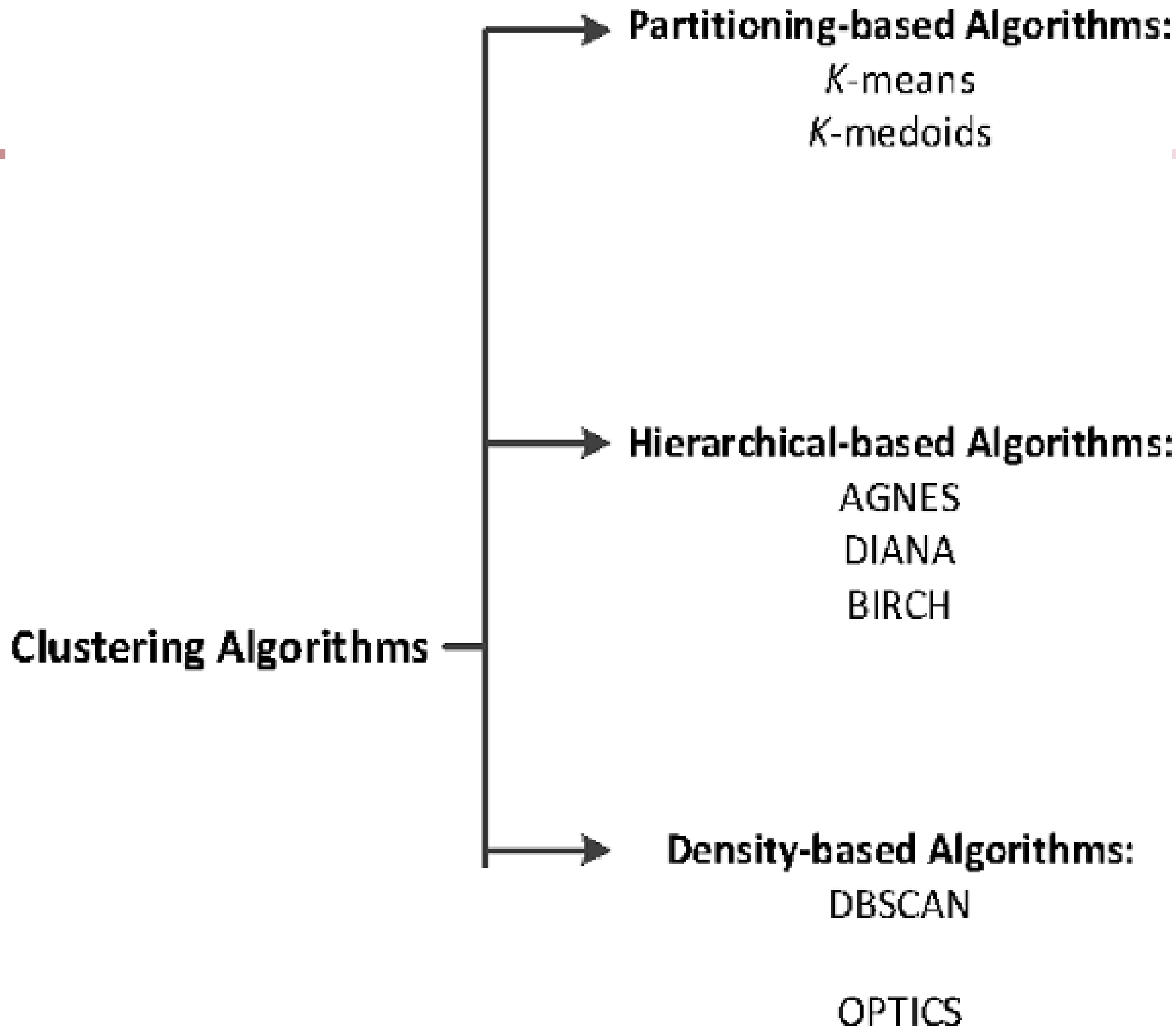
$||x|| = \sqrt{(3)^2 + (2)^2 + (0)^2 + (5)^2} = 6.16$

$||y|| = \sqrt{(1)^2 + (0)^2 + (0)^2 + (0)^2} = 1$

∴ Cos(x, y) = 3 / (6.16 * 1) = 0.49

The dissimilarity between the two vectors ‘x’ and ‘y’ is given by –

∴ Dis(x, y) = 1 - Cos(x, y) = 1 - 0.49 = 0.51


Clustering Algorithms
Partitioning-based Algorithms:
K-means
K-medoids
Hierarchical-based Algorithms:
AGNES
DIANA
BIRCH
Density-based Algorithms:
DBSCAN
OPTICS

Partitioning Algorithms: Basic Concept

- Given *D*, a data set of *n* objects, and *k*, the number of clusters to form, a partitioning algorithm organizes the objects into *k* partitions ($k <= n$), where each partition represents a cluster.
- The clusters are formed to optimize an objective partitioning criterion, such as a dissimilarity function based on distance, so that the objects within a cluster are "similar," whereas the objects of different clusters are "dissimilar" in terms of the data set attributes.
  - Heuristic methods: *k-means* and *k-medoids* algorithms
  - *k-means* : Each cluster is represented by the center of the cluster
  - *k-medoids* or PAM (Partition around medoids) : Each cluster is represented by one of the objects in the cluster

The *K-Means* Clustering Method

- Given *k*, the *k-means* algorithm is implemented in four steps:
  - Partition objects into *k* nonempty subsets
  - Compute seed points as the centroids of the clusters of the current partition (the centroid is the center, i.e., *mean point*, of the cluster)
  - Assign each object to the cluster with the nearest seed point
  - Go back to Step 2, stop when no more new assignment

K-means Clustering

Algorithm: *k*-means. The *k*-means algorithm for partitioning, where each cluster's center is represented by the mean value of the objects in the cluster.

Input:

- *k*: the number of clusters,
- *D*: a data set containing *n* objects.

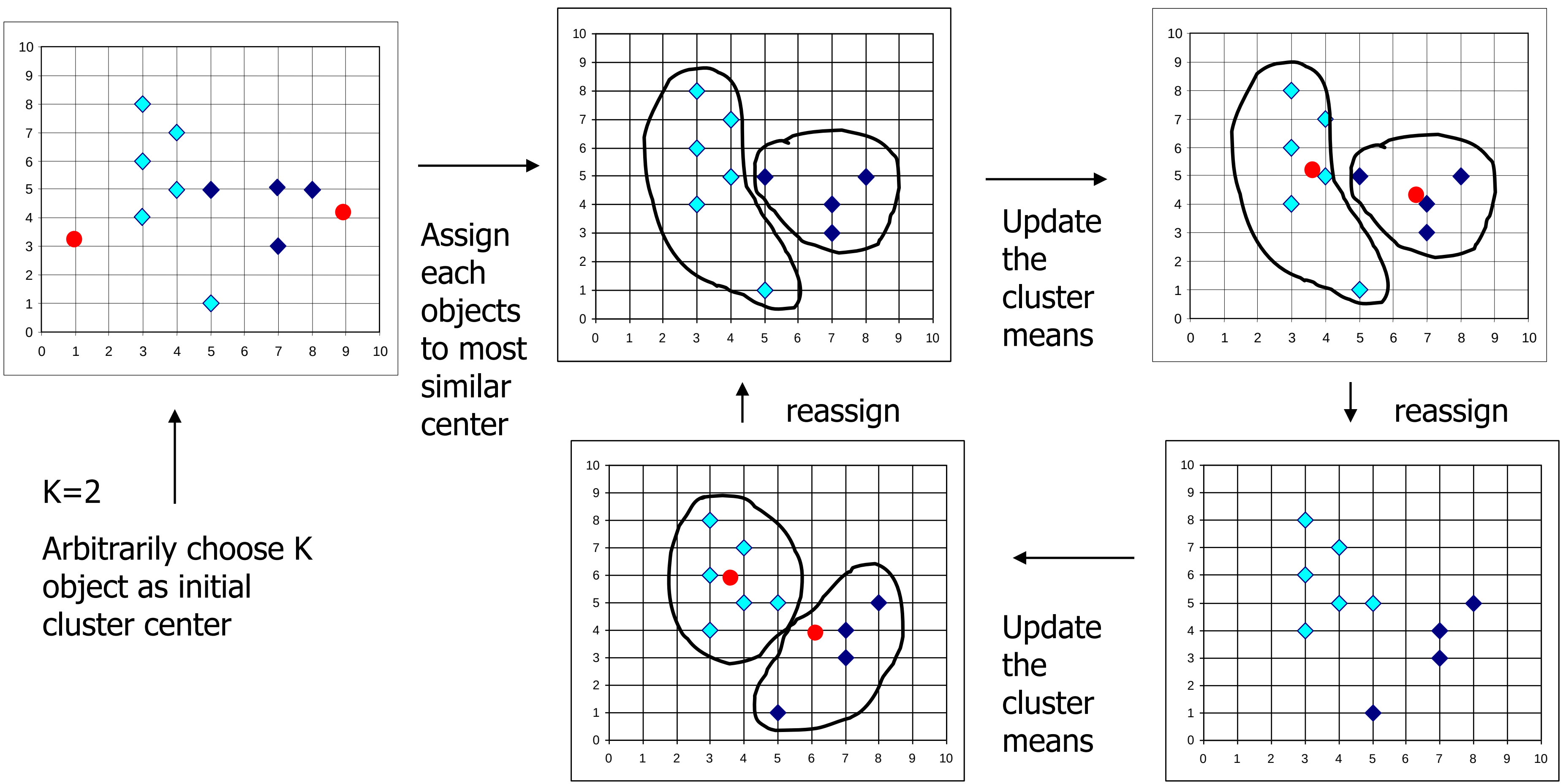
Output: A set of *k* clusters.

Method:

(1) arbitrarily choose *k* objects from *D* as the initial cluster centers;
(2) repeat
(3) (re)assign each object to the cluster to which the object is the most similar, based on the mean value of the objects in the cluster;
(4) update the cluster means, i.e., calculate the mean value of the objects for each cluster;
(5) until no change;

The *K-Means* Clustering Method

- Example



Example

Example 1

- Using K-mean solve the following with k=2
  {2,25,10,15,5,20,4,40}
- Let c1=2 and c2=10, then d(i,j)

K=2	2	25	10	15	5	20	4	40
2	0	23	8	13	3	18	2	38
10	8	15	0	5	5	10	6	30

- New c1= (2+5+4)/3=3.67, c2=(25+10+15+20+40)/5=22

K=2	2	25	10	15	5	20	4	40
3.67	1.67	21.33	6.33	11.33	1.33	16.33	0.33	36.33
22	20	3	12	7	17	2	18	18

Example 1 (conti..)

- New c1= (2+10+5+4)/4=5.25, c2=(25+15+20+40)/4=25

K=2	2	25	10	15	5	20	4	40
5.25	3.25	19.75	4.75	9.75	0.25	14.75	1.25	34.75
25	23	0	15	10	20	5	21	15

- New c1= (2+10+5+4+15)/5=7.2, c2=(25+20+40)/3=28.33

K=2	2	25	10	15	5	20	4	40
7.2	5.2	17.8	2.8	7.8	2.2	12.8	3.2	32.8
28.33	26.33	3.33	18.33	13.33	23.33	8.33	24.33	11.67

Use k-mean Algorithm to create three clusters for given set of values :–
{ 2, 3, 7, 8, 9, 15, 17, 19, 25 }.

Example 2

- Using K-mean solve the following with k=2
  {2,25,10,15,5,20,4,40}
- Let c1=4 and c2=40, then d(i,j)

K=2	2	25	10	15	5	20	4	40
4	2	21	6	11	1	16	0	36
40	38	15	30	25	35	20	36	0

- New c1= (2+10+15+5+20+4)/6=9.33, c2=(25+40)/2=32.5

K=2	2	25	10	15	5	20	4	40
9.33	7.33	15.67	0.67	5.67	4.33	10.67	5.33	30.67
32.5	29.5	7.5	22.5	17.5	27.5	12.5	28.5	7.5

- so the final clusters are:

  c1= 2,4,5,10,15,20 with mean=9.33,

  c2= 25,40 with mean=32.5

K-means

- The algorithm attempts to determine k partitions that minimize the square-error function.

$$E = \sum_{i=1} \sum_{x=c} [d(x, \bar{x}_i)]^2$$

  - **E**: the sum of the squared error for all objects in the data set
  - **x**: the data point in the space representing an object
  - **xi**: is the mean of cluster Ci

The algorithm that finds clusters in such a way that the above sum is minimized would be good clustering

Example 1

- Apply K-means algorithm on {2,25,10,15,5,20,4,40}to create two clusters

Randomly select two data objects

Let c1=2 and c2=10

Iteration1:

K=2	2	25	10	15	5	20	4	40
2	0	23	8	13	3	18	2	38
10	8	15	0	5	5	10	6	30

- After Iteration 1
- C1– (2,5,4) and C2 (25,10,15,20,40)

  c1= (2+5+4)/3=3.67,

  c2=(25+10+15+20+40)/5=22

Example 1

{2,25,10,15,5,20,4,40}

Iteration 2:

New c1= (2+5+4)/3=3.67, c2=(25+10+15+20+40)/5=22

K=2	2	25	10	15	5	20	4	40
3.67	1.67	21.33	6.33	11.33	1.33	16.33	0.33	36.33
22	20	3	12	7	17	2	18	18

- After Iteration 2
- C1– (2,10,5,4) and C2 (25,15,20,40)
- c1= (2+10+5+4)/4=5.25
- c2=(25+15+20+40)/4=25

Example 1

{2,25,10,15,5,20,4,40}

Iteration 3:

- New c1= (2+10+5+4)/4=5.25, c2=(25+15+20+40)/4=25

K=2	2	25	10	15	5	20	4	40
5.25	3.25	19.75	4.75	9.75	0.25	14.75	1.25	34.75
25	23	0	15	10	20	5	21	15

- After Iteration 3
- C1– (2,10,15,5,4) and C2 (25,15,20,40)
- c1= (2+10+15+5+4)/5=7.2
- c2=(25+20+40)/3=28.33

Example 1 (conti..)

Iteration 4:

- New c1= (2+10+5+4+15)/5=7.2, c2=(25+20+40)/3=28.33

K=2	2	25	10	15	5	20	4	40
7.2	5.2	17.8	2.8	7.8	2.2	12.8	3.2	32.8
28.33	26.33	3.33	18.33	13.33	23.33	8.33	24.33	11.67

- After Iteration 4
- C1– (2,10,15,5,4) and C2 (25,20,40)
- c1= (2+10+5+4+15)/5=7.2,
- c2=(25+20+40)/3=28.33
- As there is no more relocations occur after iteration 4. so we can stop.
- Thus C1 – (2,10,15,5,4) and C2(25, 20,40) with Centroid of Cluster1 as 7.2 and Cluster2 28.33

Example 2

- Using K-mean solve the following with k=2
  {2,25,10,15,5,20,4,40}
- Let c1=2 and c2=25, then d(i,j)

K=2	2	25	10	15	5	20	4	40
4	2	21	6	11	1	16	0	36
40	38	15	30	25	35	20	36	0

- New c1= (2+10+15+5+20+4)/6=9.33, c2=(25+40)/2=32.5

K=2	2	25	10	15	5	20	4	40
9.33	7.33	15.67	0.67	5.67	4.33	10.67	5.33	30.67
32.5	29.5	7.5	22.5	17.5	27.5	12.5	28.5	7.5

- so the final clusters are:

  c1= 2,4,5,10,15,20 with mean=9.33,

  c2= 25,40 with mean=32.5

K-means

- The algorithm attempts to determine k partitions that minimize the square-error function.

$$E = \sum_{i=1} \sum_{x=c} d(x, \bar{x}_i)^2$$

  - **E**: the sum of the squared error for all objects in the data set
  - **x**: the data point in the space representing an object
  - **xi**: is the mean of cluster Ci

The algorithm that finds clusters in such a way that the above sum is minimized would be good clustering

number of clusters

number of cases

centroid for cluster *j*

case *i*

objective function

$$J = \sum_{j=1}^{k} \sum_{i=1}^{n} \underbrace{\left\| x_i^{(j)} - c_j \right\|^2}_{\text{Distance function}}$$

Example 1 (conti..)

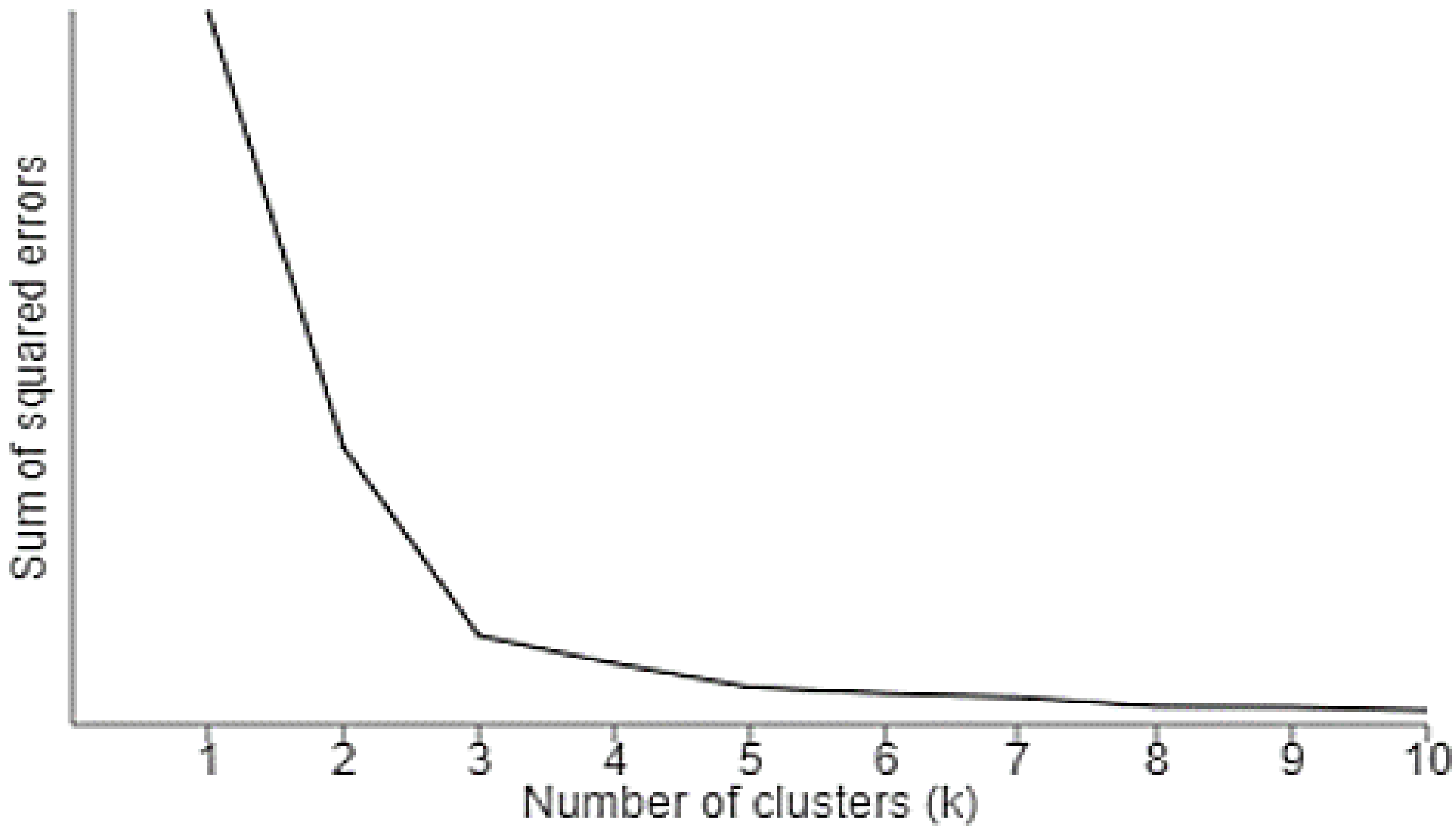
K=2	2	25	10	15	5	20	4	40
7.2	5.2	17.8	2.8	7.8	2.2	12.8	3.2	32.8
28.33	26.33	3.33	18.33	13.33	23.33	8.33	24.33	11.67

- C1 – (2,10,15,5,4) and C2(25, 20,40) with Centroid of Cluster1 as 7.2 and Cluster2 28.33

- $E = \{(5.2)^2+(2.8)^2+(7.8)^2+(2.2)^2+(3.2)^2\}+\{(3.33)^2+(8.33)^2+(11.67))^2$

  $=\{104.56\}+\{216.66\}$

  $= 321.22$


Sum of squared errors
1
2
3
4
5
6
7
8
9
10
Number of clusters (k)

K=2	2	25	10	15	5	20	4	40
9.33	7.33	15.67	0.67	5.67	4.33	10.67	5.33	30.67
32.5	29.5	7.5	22.5	17.5	27.5	12.5	28.5	7.5

7.33	7.5	0.67	5.67	4.33	10.67	5.33	7.5	
53.72	56.25	0.44	32.14	18.74	113.84	28.40	56.25	359.83

K=2	2	25	10	15	5	20	4	40
7.2	5.2	17.8	2.8	7.8	2.2	12.8	3.2	32.8
28.33	26.33	3.33	18.33	13.33	23.33	8.33	24.33	11.67

5.2	3.33	2.8	7.8	2.2	8.33	3.2	11.7	
27.04	11.08	7.84	60.84	4.84	69.38	10.24	136.18	327.46

- Find dissimilarity matrix with **Euclidean** distance

ID	Name	Height	Weight
x1	Ram	64	60
x2	Shyam	60	61
x3	Gita	59	70
x4	Mohan	68	71

- **Euclidean** distance

$$d(i,j)=\sqrt{(|x_{i_1}-x_{j_1}|^2+|x_{i_2}-x_{j_2}|^2+\ldots+|x_{i_p}-x_{j_p}|^2)}$$

	x1	x2	x3	x4
x1	0			
x2	d(x2,x1)	0		
x3	d(x3,x1)	d(x3,x2)	0	
x4	d(x4,x1)	d(x4,x2)	d(x4,x3)	0

d(x3,x1)	Sqrt[$(59-64)^2+(70-60)^2$]	11.18
d(x3,x2)	Sqrt[$(59-60)^2+(70-61)^2$]	9.06
d(x4,x1)	Sqrt[$(68-64)^2+(71-60)^2$]	11.7
d(x4,x2)	Sqrt[$(68-60)^2+(71-61)^2$]	12.81

	x1	x2	x3	x4
x1	0			
x2		0		
x3	11.18	9.06	0	
x4	11.7	12.81		0

Example 2

- Using K-mean solve the following with k=2 use **Euclidean** distance to find similarity.

ID	Name	Height	Weight
x1	Ram	64	60
x2	Shyam	60	61
x3	Gita	59	70
x4	Mohan	68	71

- Let c1=x1 and c2=x2, then d(i,j) by **Euclidean** distance

	x1	x2	x3	x4
x1	0			
x2	d(x2,x1)	0		
x3	d(x3,x1)	d(x3,x2)	0	
x4	d(x4,x1)	d(x4,x2)	d(x4,x3)	0

Example 2

- Using K-mean solve the following with k=2

ID	Name	Height	Weight
x1	Ram	64	60
x2	Shyam	60	61
x3	Gita	59	70
x4	Mohan	68	71

	x1	x2	x3	x4
x1	0			
x2		0		
x3	11.18	9.06	0	
x4	11.7	12.81		0

- Let c1=x1 and c2=x2, then d(i,j) by **Euclidean** distance

d(x3,x1)	$Sqrt[(59-64)^2+(70-60)^2]$	11.18
d(x3,x2)	$Sqrt[(59-60)^2+(70-61)^2]$	**9.06**
d(x4,x1)	$Sqrt[(68-64)^2+(71-60)^2]$	**11.7**
d(x4,x2)	$Sqrt[(68-60)^2+(71-61)^2]$	12.81

K=2	X3	X4
x1	11.18	11.7
x2	9.06	12.81

- New c1=(64+68/2, 60+71/2)=(66,65.5),

  c2= (60+59/2, 61+70/2)=(59.5,65.5)

Example 2

- New c1=(64+68/2, 60+71/2)=(66,65.5), c2= (60+59/2, 61+70/2)=(59.5,65.5)

d(x1,(66,65.5))	**5.85**
d(x1,(59.5,65.5))	7.1
d(x2,(66,65.5))	7.5
d(x2,(59.5,65.5))	**4.53**
d(x3,(66,65.5))	8.32
d(x3,(59.5,65.5))	**4.53**
d(x4,(66,65.5))	**5.85**
d(x4,(59.5,65.5))	10.12

ID	Name	Height	Weight
x1	Ram	64	60
x2	Shyam	60	61
x3	Gita	59	70
x4	Mohan	68	71

K=2	X1	X2	X3	X4
66,65.5	5.85	7.5	8.32	5.85
59.5,65.5	7.1	4.53	4.53	10.12

- New c1=(64+68/2, 60+71/2)=(66,65.5),

  c2= (60+59/2, 61+70/2)=(59.5,65.5)

Assignment2

1. Given two objects represented by the tuples (22, 1, 42, 10) and (20, 0, 36, 8):

   (a) Compute the *Euclidean distance* between the two objects.
   (b) Compute the *Manhattan distance* between the two objects.
   (c) Compute the *Minkowski distance* between the two objects, using $q = 3$.

2. Suppose that the data mining task is to cluster the following eight points (with (x, y) representing location) into three clusters:

$$A_1(2, 10), A_2(2, 5), A_3(8, 4), B_1(5, 8), B_2(7, 5), B_3(6, 4), C_1(1, 2), C_2(4, 9).$$

   The distance function is Euclidean distance. Suppose initially we assign A_1, B_1, and C_1 as the center of each cluster, respectively. Use the *k-means* algorithm to show *only*

   (a) The three cluster centers after the first round execution
   (b) The final three clusters

3. Explain K-means clustering and solve the following with k=3

   { 2, 3, 6, 8, 9, 12, 15, 18, 22 }

Ans

1. (a) Compute the *Euclidean distance* between the two objects.

$$d(i,j) = \sqrt{(x_{i1}-x_{j1})^2 + (x_{i2}-x_{j2})^2 + \cdots + (x_{in}-x_{jn})^2}$$
$$= \sqrt{|22-20|^2 + |1-0|^2 + |42-36|^2 + |10-8|^2} = 6.71$$

(b) Compute the *Manhattan distance* between the two objects.

$$d(i,j) = |x_{i1}-x_{j1}| + |x_{i2}-x_{j2}| + \cdots + |x_{in}-x_{jn}|$$
$$= |22-20| + |1-0| + |42-36| + |10-8| = 11$$

(c) Compute the *Minkowski distance* between the two objects, using $p = 3$.

$$d(i,j) = (|x_{i1}-x_{j1}|^q + |x_{i2}-x_{j2}|^q + \cdots + |x_{in}-x_{jn}|^p)^{1/p}$$
$$= (|22-20|^3 + |1-0|^3 + |42-36|^3 + |10-8|^3)^{1/3} = 6.15$$

2. Answer:

(a) After the first round, the three new clusters are: (1) $\{A_1\}$, (2) $\{B_1, A_3, B_2, B_3, C_2\}$, (3) $\{C_1, A_2\}$, and their centers are (1) (2, 10), (2) (6, 6), (3) (1.5, 3.5).

(b) The final three clusters are: (1) $\{A_1, C_2, B_1\}$, (2) $\{A_3, B_2, B_3\}$, (3) $\{C_1, A_2\}$.

Comments on the *K-Means* Method

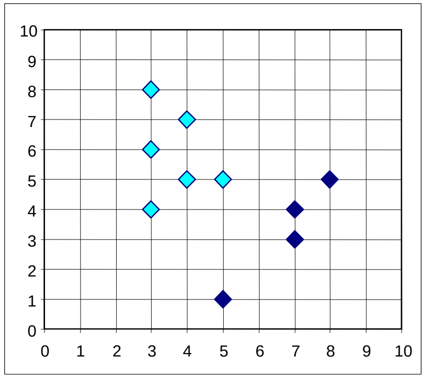
- Strength: *Relatively efficient*: $O(tkn)$, where n is # objects, k is # clusters, and t is # iterations. Normally, k, $t << n$.
- It works well when the clusters are compact clouds ,that are rather well separated from one another
- Weakness
  - Applicable only when *mean* is defined, then what about categorical data?
  - Need to specify *k,* the *number* of clusters, in advance
  - Unable to handle noisy data and *outliers*

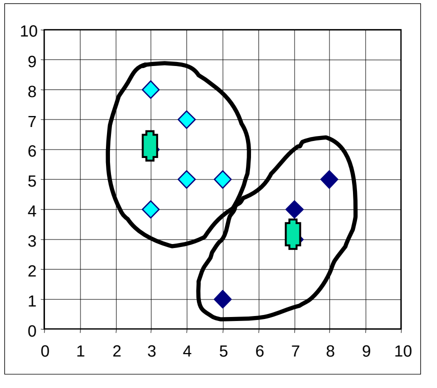
Variations of the *K-Means* Method

- A few variants of the *k-means* which differ in
  - Selection of the initial *k* means
  - Dissimilarity calculations

What Is the Problem of the K-Means Method?

- The k-means algorithm is sensitive to outliers !
  - Since an object with an extremely large value may substantially distort the distribution of the data.
- 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.





The *K-Medoids* Clustering Method

- Minimize the sensitivity of k-means to outliers
- Pick actual objects to represent clusters instead of mean values
- Each remaining object is clustered with the representative object (**Medoid**) to which is the most similar
- The algorithm minimizes the sum of the dissimilarities between each object and its corresponding reference point

$$E = \sum_{i=1} \sum_{x=c} d^{(x, \bar{x}_i)}$$

  - **E**: the sum of absolute error for all objects in the data set
  - **x**: the data point in the space representing an object
  - **xi**: is the representative object of cluster Ci

The *K-Medoids* Clustering Method

- Initial representatives are chosen randomly
- The iterative process of replacing representative objects by no representative objects continues as long as the quality of the clustering is improved
- For each representative Object O
- For each non-representative object R, swap O and R
- Choose the configuration with the lowest cost
- Cost function is the difference in absolute error-value if a current representative object is replaced by a non-representative object

The *K-Medoids* Clustering Method

- Algorithm: *k*-medoids. PAM, a *k*-medoids algorithm for partitioning based on medoid or central objects.
- Input:
  - *k*: the number of clusters,
  - *D*: a data set containing *n* objects.
- Output: A set of *k* clusters.

The *K-Medoids* Clustering Method

Method:

(1) arbitrarily choose *k* objects in *D* as the initial representative objects or seeds;

(2) repeat

(3) assign each remaining object to the cluster with the nearest representative object;

(4) randomly select a nonrepresentative object, ***o***random;

(5) compute the total cost, *S*, of swapping representative object, ***oj***, with ***o***random;

(6) if *S* < 0 then swap ***oj*** with ***o***random to form the new set of *k* representative objects;

(7) until no change;

Example

- Using K-medoid solve the following with k=2

ID	Name	Height	Weight
x1	Ram	64	60
x2	Shyam	60	61
x3	Gita	59	70
x4	Mohan	68	71

- Let c1=x1 and c2=x2, then d(i,j) by **Euclidean** distance

	x1	x2	x3	x4
x1	0			
x2	d(x2,x1)	0		
x3	d(x3,x1)	d(x3,x2)	0	
x4	d(x4,x1)	d(x4,x2)	d(x4,x3)	0

Example

d(i,j) by **Euclidean** distance

ID	Name	Height	Weight
x1	Ram	64	60
x2	Shyam	60	61
x3	Gita	59	70
x4	Mohan	68	71

	x1	x2	x3	x4
x1	0			
x2	4.123	0		
x3	11.18	9.06	0	
x4	11.7	12.81	9.05	0

d(x3,x1)	$Sqrt[(59-64)^2+(70-60)^2]$	11.18
d(x3,x2)	$Sqrt[(59-60)^2+(70-61)^2]$	9.06
d(x4,x1)	$Sqrt[(68-64)^2+(71-60)^2]$	11.7
d(x4,x2)	$Sqrt[(68-60)^2+(71-61)^2]$	12.81
d(x2,x1)	$Sqrt[(60-64)^2+(61-60)^2]$	4.123
d(x3,x4)	$Sqrt[(59-68)^2+(70-71)^2]$	9.05

- Let c1=x1 and c2=x2

Example

- **Computing Clustering Quality** : In order to calculate the quality of above clustering. The absolute error mean is

$$E=d(x3,x2)+d(x4,x1)$$
$$=20.76$$

Iteration 2

- Either x3 or x4 is selected in place of x1 or x2.
- For each of these four cases , a new clustering and its corresponding quality need to be computed.
- The new clustering that has the best quality should be selected.
  - Consider the case where x4 is selected in place of x1
  - C1=x4 and c2=x2
  - Clusters are Cluster1=(x3,x4) and cluster2= (x1,x2)
  - **Computing Clustering Quality** : The absolute error mean becomes

$$E = d(x1,x2) + d(x3,x4)$$
$$= 13.17$$

Iteration 2

- The above procedure is repeated for each of the other three cases.
- The case in which there is least error is selected.
- If the absolute error is less than the error during iteration 1, then corresponding change in medoids is enforced and the next iteration is started.

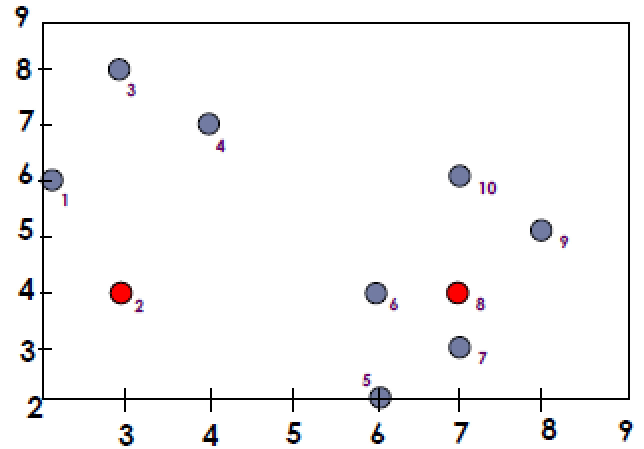
K-Medoids -Example

Using K-medoid solve the following with k=2

o1	2	6
o2	3	4
o3	3	8
o4	4	7
o5	6	2
o6	6	4
o7	7	3
o8	7	4
o9	8	5
o10	7	6

K-Medoids -Example

o1	2	6
o2	3	4
o3	3	8
o4	4	7
o5	6	2
o6	6	4
o7	7	3
o8	7	4
o9	8	5
o10	7	6



Goal: create two clusters

Choose randmly two medoids

O2 = (3,4)

O8 = (7,4)

Example

- **Goal: create two clusters**
- **Cluster1** O2 =(3,4)
- **Cluster2** O8= (7,4)
- Using **Manhattan distance** find dissimilarity matrix

	o1	o2	o3	o4	o5	o6	o7	o8	o9	o10
o1	0									
o2	d(o2,o1)	0								
o3	d(o3,o1)	d(o3,o2)	0							
o4	d(o4,o1)	d(o4,o2)	d(o4,o3)	0						
o5	d(o5,o1)	d(o5,o2)	d(o5,o3)	d(o5,o4)	0					
o6	d(o6,o1)	d(o6,o2)	d(o6,o3)	d(o6,o4)	d(o6,o5)	0				
o7	d(o7,o1)	d(o7,o2)	d(o7,o3)	d(o7,o4)	d(o7,o5)	d(o7,o6)	0			
o8	d(o8,o1)	d(o8,o2)	d(o8,o3)	d(o8,o4)	d(o8,o5)	d(o8,o6)	d(o8,o7)	0		
o9	d(o9,o1)	d(o9,o2)	d(o9,o3)	d(o9,o4)	d(o9,o5)	d(o9,o6)	d(o9,o7)	d(o9,o8)	0	
o10	d(o10,o1)	d(o10,o2)	d(o10,o3)	d(o10,o4)	d(o10,o5)	d(o10,o6)	d(o10,o7)	d(o10,o8)	d(o10,o9)	0

o1	2	6
o2	3	4
o3	3	8
o4	4	7
o5	6	2
o6	6	4
o7	7	3
o8	7	4
o9	8	5
o10	7	6

	o1	o2	o3	o4	o5	o6	o7	o8	o9	o10
o1	0									
o2	3	0								
o3	3	4	0							
o4	3	4	2	0						
o5	8	5	9	7	0					
o6	6	3	7	5	2	0				
o7	8	5	9	7	2	2	0			
o8	7	4	8	6	3	1	1	0		
o9	7	6	8	6	5	3	3	2	0	
o10	5	6	6	4	5	3	3	1	2	0

K-Medoids -Example

- **Cluster1** = {O1, O2, O3, O4}
- **Cluster2** = {O5, O6, O7, O8, O9, O10}
- Compute the absolute error criterion **[for the set of Medoids (O2,O8)]**

- $E = (3+0+4+4)+(3+1+1+0+2+2) = 20$

Example

Goal: create two clusters C1= O2 = (3,4) **C2 =** O8 = (7,4)
Using **Manhattan distance** find dissimilarity matrix

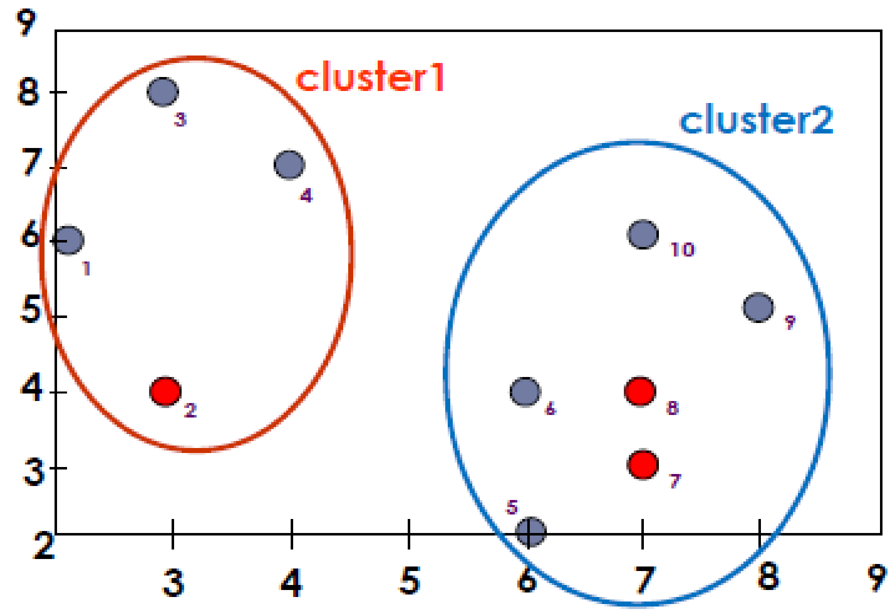
o1	2	6
o2	3	4
o3	3	8
o4	4	7
o5	6	2
o6	6	4
o7	7	3
o8	7	4
o9	8	5
o10	7	6

	o1	o2	o3	o4	o5	o6	o7	o8	o9	o10
o1	0									
o2	3	0								
o3	3	4	0							
o4	3	4	2	0						
o5	8	5	9	7	0					
o6	6	3	7	5	2	0				
o7	8	5	9	7	2	2	0			
o8	7	4	8	6	3	1	1	0		
o9	7	6	8	6	5	3	3	2	0	
o10	5	6	6	4	5	3	3	2	2	0

K-Medoids -Example

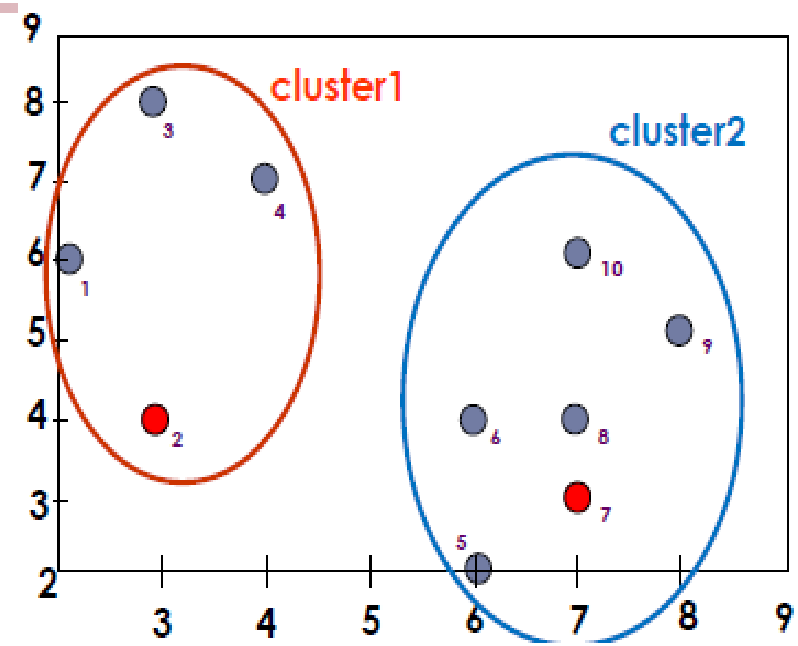
- **Cluster1** = {O1, O2, O3, O4}
- **Cluster2** = {O5, O6, O7, O8, O9, O10}
- Compute the absolute error criterion **[for the set of Medoids (O2,O8)]**
- $E = (3+ 4 + 4)+(3+1+1+0+ 2+ 2)= 20$

K-Medoids -Example



- **Iteration 2**
- Choose a random object **O7**
- Swap **O8** and **O7**
- Compute the absolute error criterion **[for the set of Medoids(O2,O7)]**
- $E = (3+ 4 + 4) + (2 + 2 +1+ 3+ 3) = 22$

K-Medoids -Example



- **Iteration 2**
- Compute the cost function
- Absolute error [for O2,O7] – Absolute error [O2,O8]
- S> 0   it is a bad idea to replace O8 by O7

- In this example, changing the medoid of cluster 2 did not change the assignments of objects to clusters.

The *K-Medoids* Clustering Method

- *PAM* works effectively for small data sets, but does not scale well for large data sets

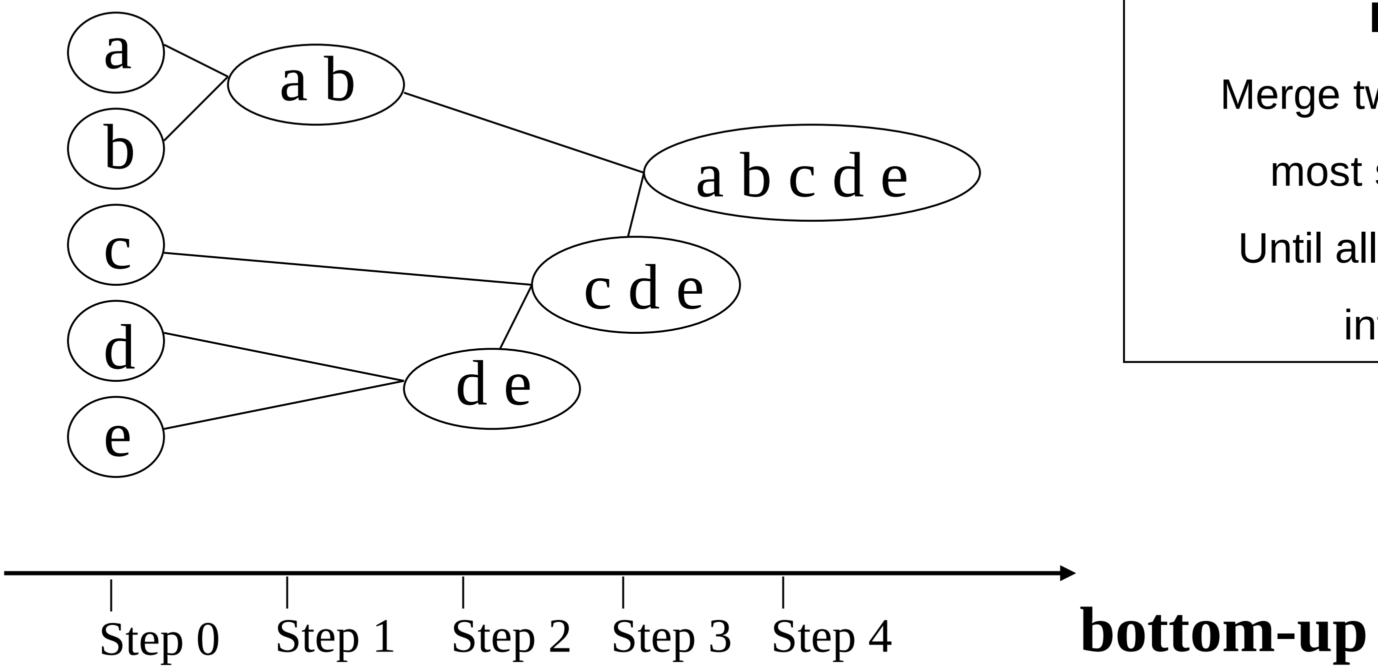
Hierarchical Clustering

Hierarchical Clustering

- A Hierarchical Clustering method works by grouping data objects into a tree of clusters
- Two types
  - Agglomerative (bottom-up/ merging)
  - Divisive(top- down/ splitting)
- It suffers from inability to perform adjustment once a merge or split decision has been executed.

Hierarchical Clustering

- Agglomerative approach



Initialization:

Each object is a cluster

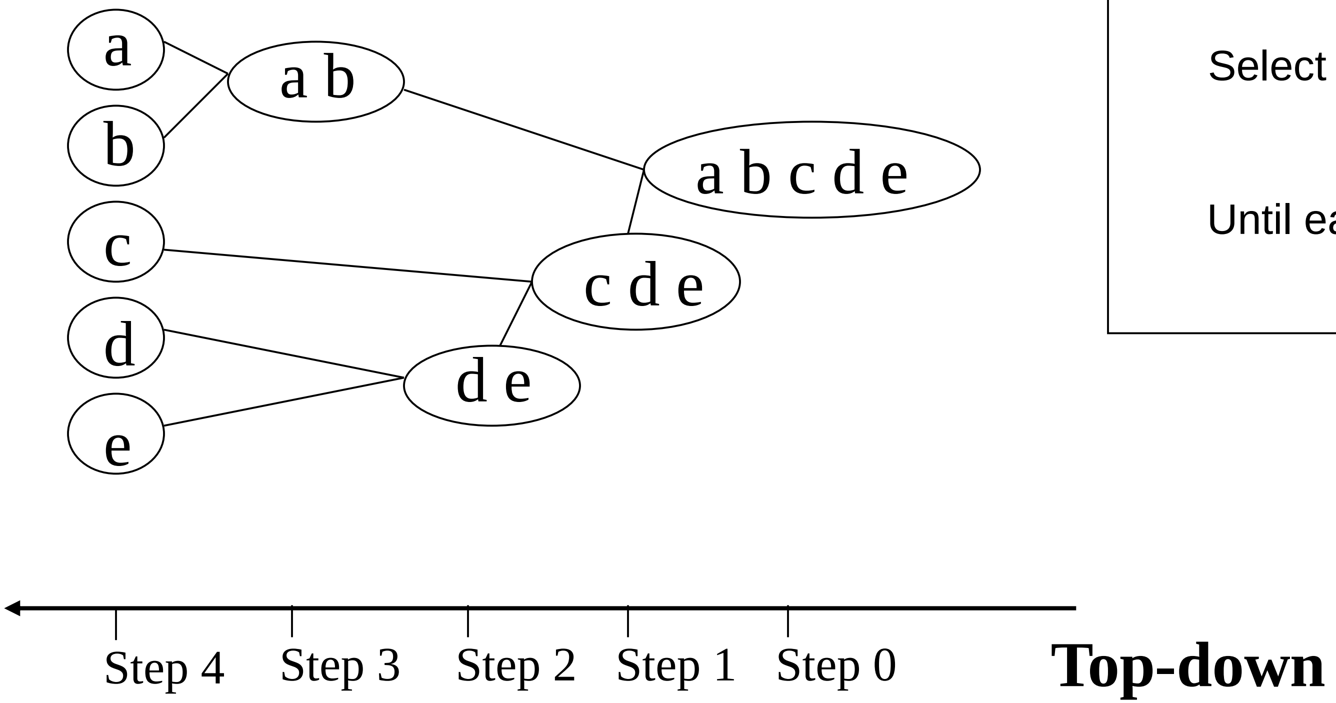
Iteration:

Merge two clusters which are most similar to each other;

Until all objects are merged into a single cluster

Hierarchical Clustering

- Divisive Approaches



Initialization:

All objects stay in one cluster

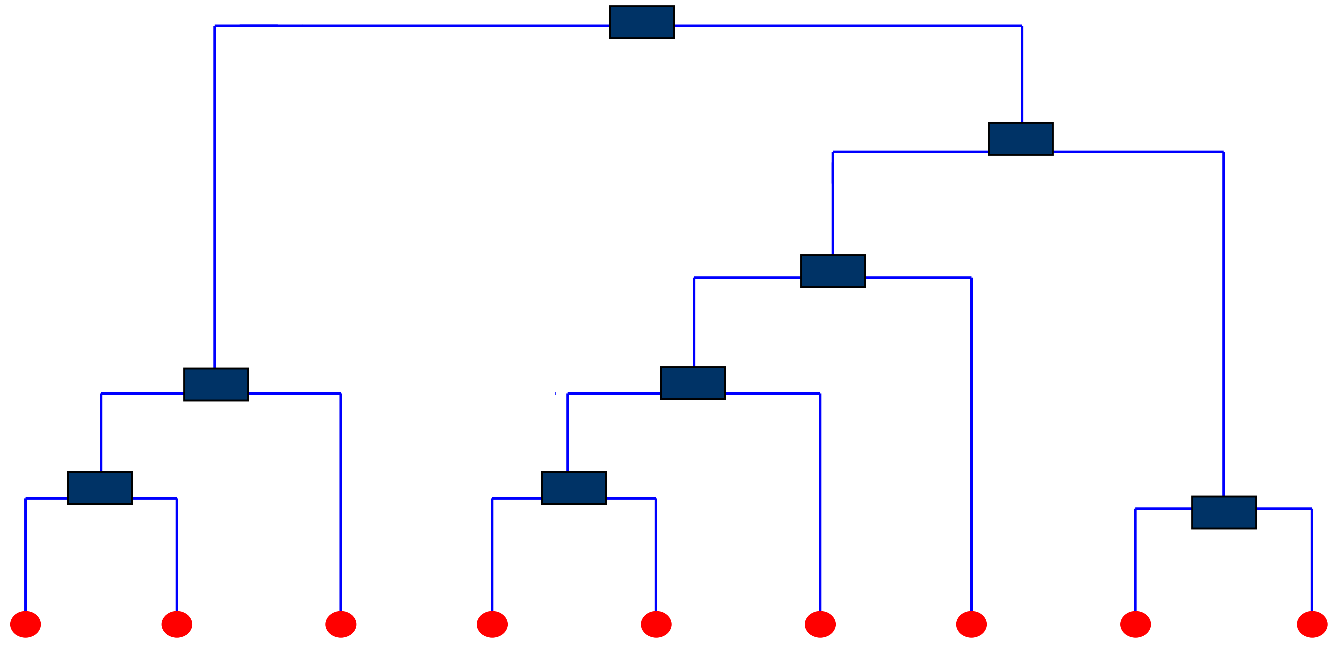
Iteration:

Select a cluster and split it into two sub clusters

Until each leaf cluster contains only one object

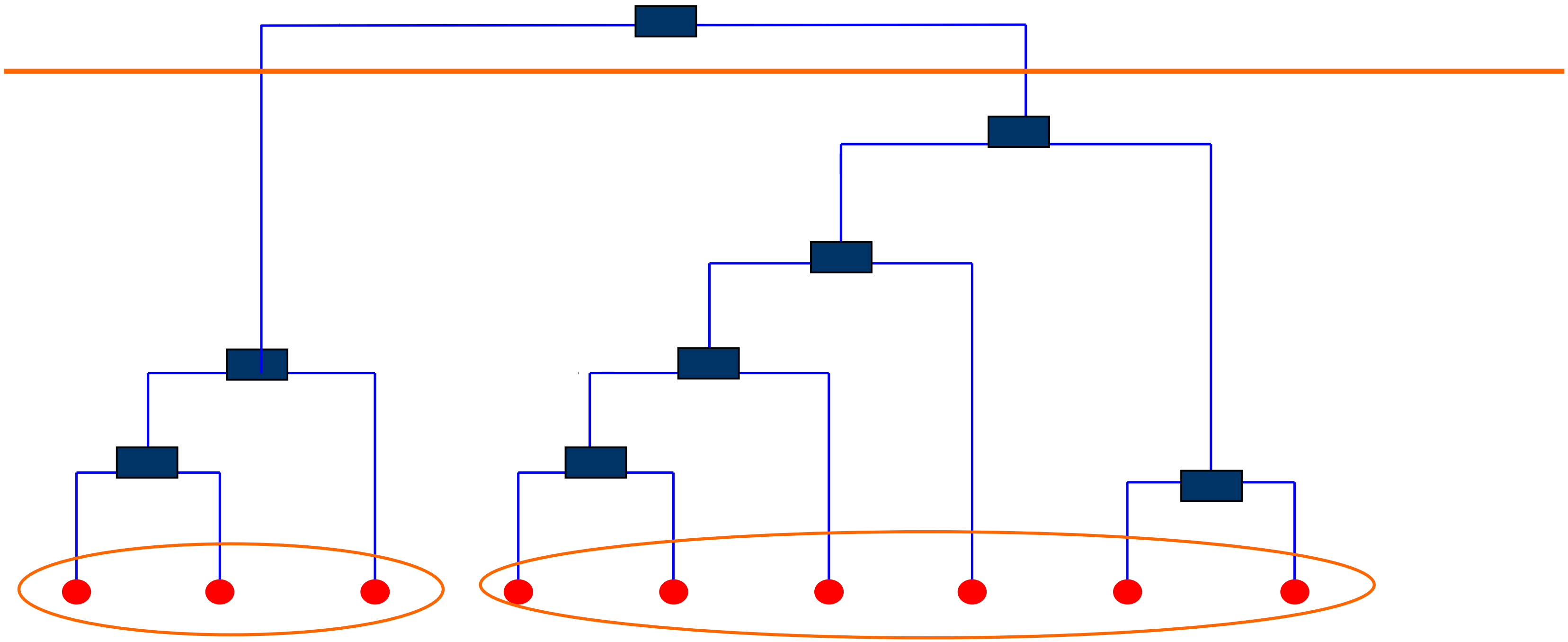
Dendrogram

- A binary tree that shows how clusters are merged/split hierarchically
- Each node on the tree is a cluster; each leaf node is a singleton cluster



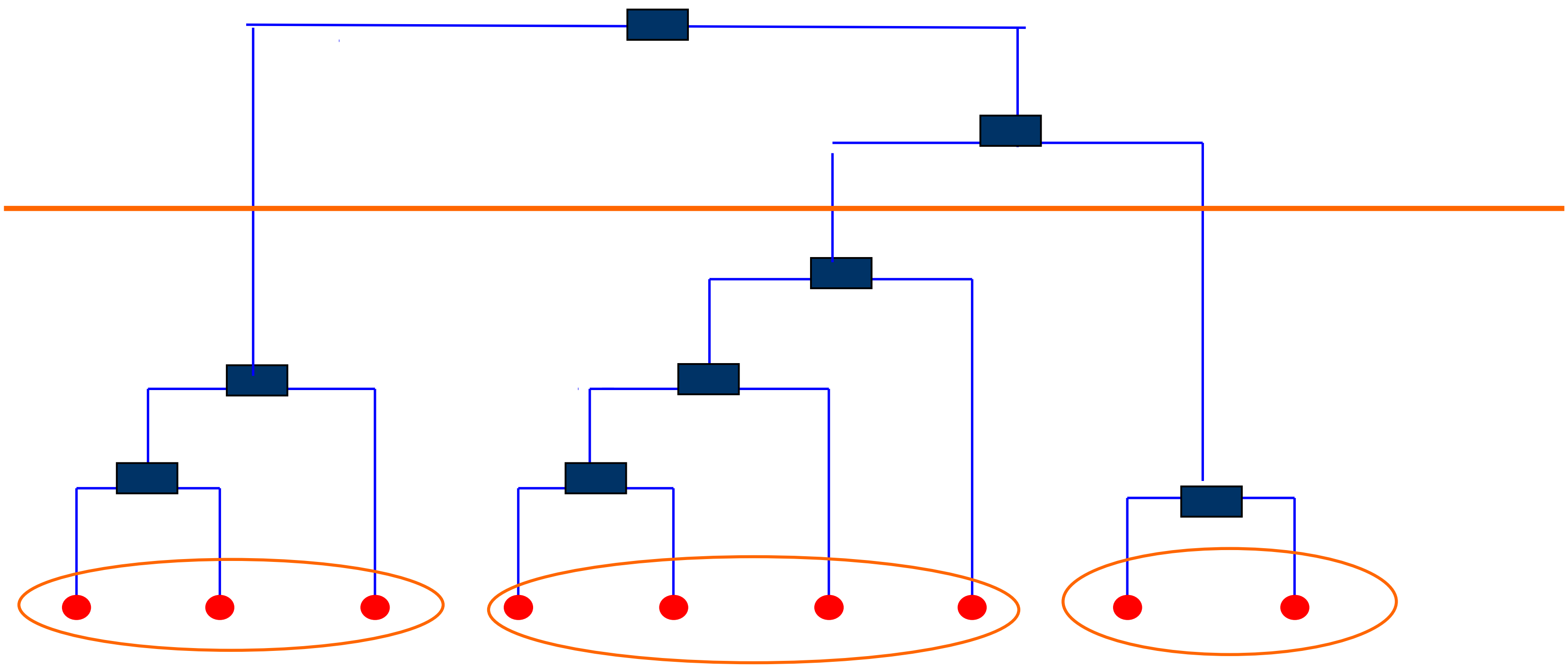
Dendrogram

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



Dendrogram

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



How to Merge Clusters?

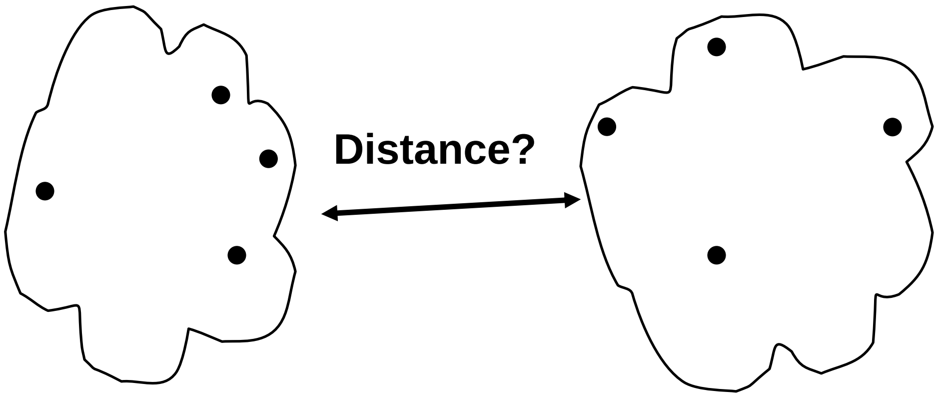
- How to measure the distance between clusters?

◆ **Single-link**

◆ **Complete-link**

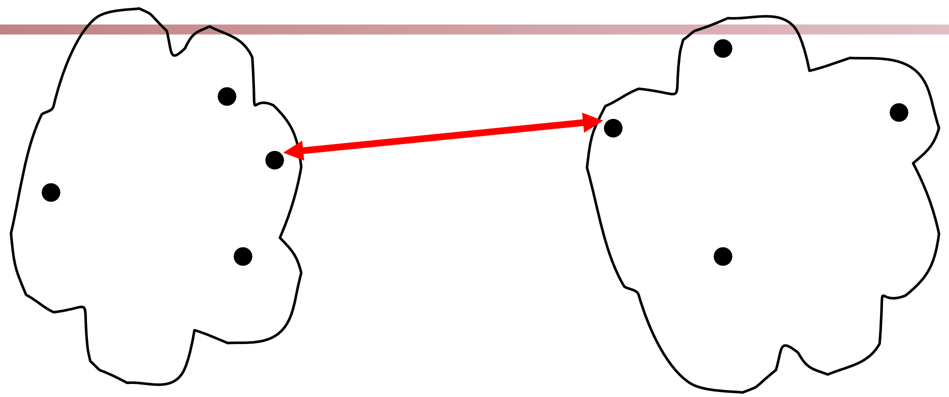
◆ **Average-link**

◆ **Centroid distance**



Hint: ***<u>Distance between clusters</u>*** is usually defined on the basis of ***<u>distance between objects.</u>***

How to Define Inter-Cluster Distance

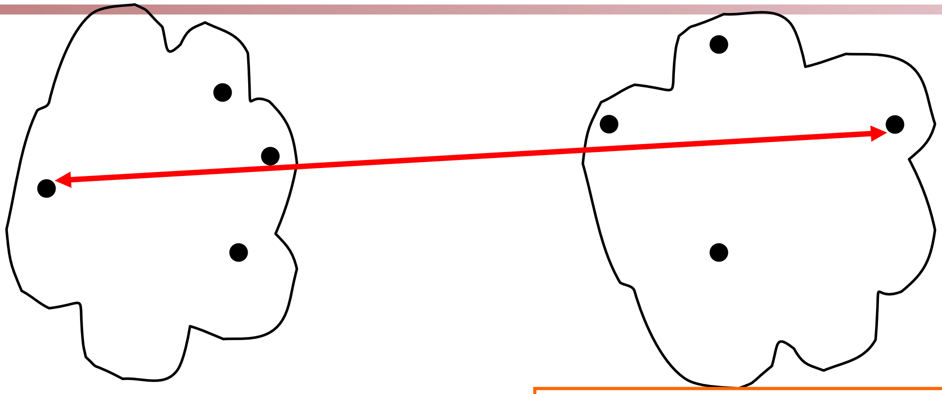


- **Single-link**
- **Complete-link**
- **Average-link**
- **Centroid distance**

$$d_{\min}(C_i, C_j) = \min_{p \in C_i, q \in C_j} d(p, q)$$

The distance between two clusters is represented by the distance of the *closest pair of data objects* belonging to different clusters.

How to Define Inter-Cluster Distance

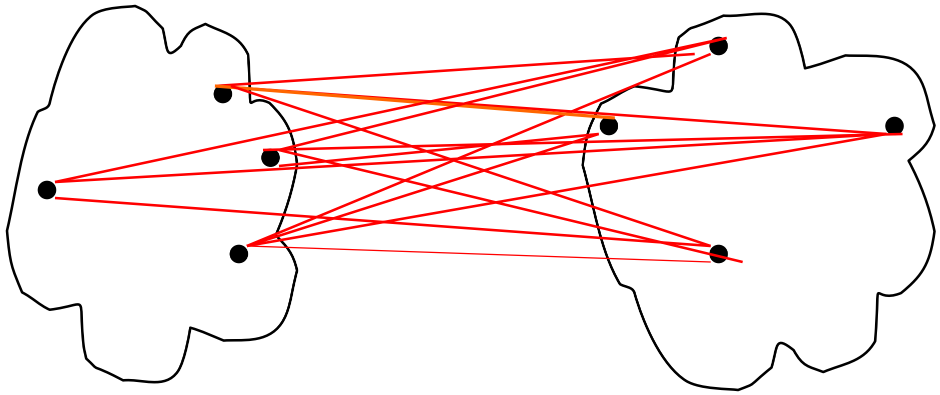


- Single-link
- Complete-link
- Average-link
- Centroid distance

$$d_{\min}(C_i, C_j) = \max_{p \in C_i, q \in C_j} d(p, q)$$

The distance between two clusters is represented by the distance of the ***farthest pair of data objects*** belonging to different clusters.

How to Define Inter-Cluster Distance

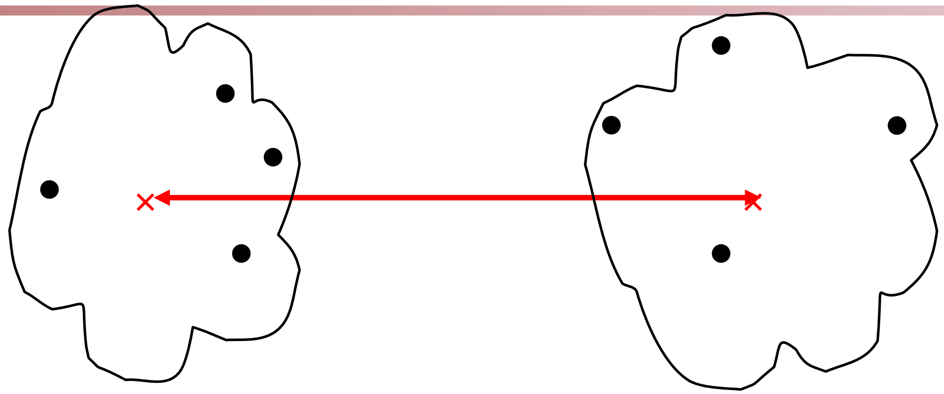


- **Single-link**
- **Complete-link**
- **Average-link**
- **Centroid distance**

$$d_{\min}(C_i, C_j) = \underset{p \in C_i, q \in C_j}{avg} d(p, q)$$

The distance between two clusters is represented by the *average* distance of *all pairs of data objects* belonging to different clusters.

How to Define Inter-Cluster Distance



m_i,m_j are the means of C_i, C_j,

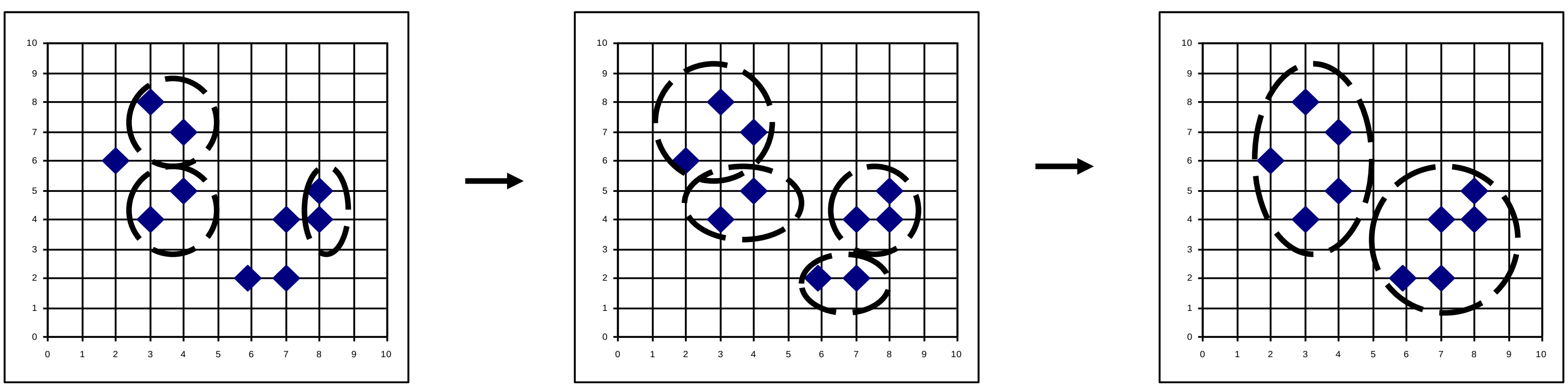
- **Single-link**
- **Complete-link**
- **Average-link**
- **Centroid distance**

$$d_{mean}(C_i, C_j) = d(m_i, m_j)$$

The distance between two clusters is represented by the distance between *the means of the clusters.*

AGNES (Agglomerative Nesting)

- Implemented in statistical analysis packages, e.g., Splus
- Use the Single-Link method and the dissimilarity matrix.
- Merge nodes that have the least dissimilarity
- Go on in a non-descending fashion
- Eventually all nodes belong to the same cluster



Agglomerative clustering algorithm

- Basic algorithm
  1. Compute the distance matrix between the input data points
  2. Let each data point be a cluster
  3. **Repeat**
  4. Merge the two closest clusters
  5. Update the distance matrix
  6. **Until** only a single cluster remains

Agglomerative clustering algorithm-Example

- Problem: Assume that the database D is given by the table below. Follow single link technique to find clusters in D. Use Euclidean distance measure.

	x	y
p1	0.40	0.53
p2	0.22	0.38
p3	0.35	0.32
p4	0.26	0.19
p5	0.08	0.41
p6	0.45	0.30

Example

- Calculate the distance from each object (point) to all other points, using Euclidean distance measure, and place the numbers in a distance matrix.

	p1	p2	p3	p4	p5	p6
p1	0					
p2	0.24	0				
p3	0.22	0.15	0			
p4	0.37	0.20	0.15	0		
p5	0.34	0.14	0.28	0.29	0	
p6	0.23	0.25	0.11	0.22	0.39	0

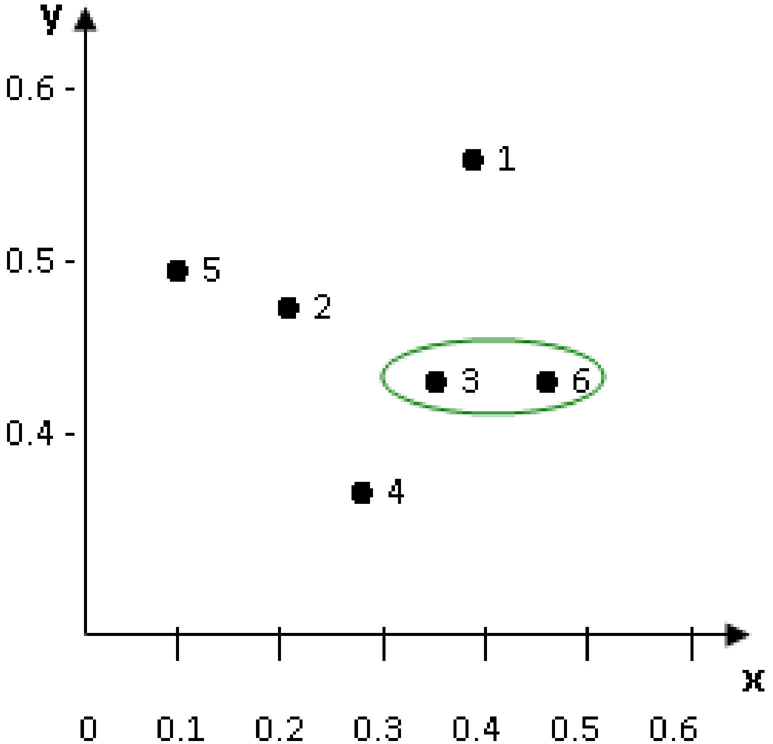
Example

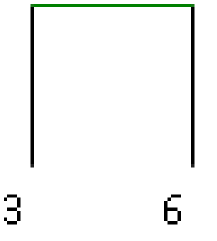
- Identify the two clusters with the shortest distance in the matrix, and merge them together. Re-compute the distance matrix, as those two clusters are now in a single cluster,

	p1	p2	p3	p4	p5	p6
p1	0					
p2	0.24	0				
p3	0.22	0.15	0			
p4	0.37	0.20	0.15	0		
p5	0.34	0.14	0.28	0.29	0	
p6	0.23	0.25	0.11	0.22	0.39	0

- p3 and p6 have the smallest distance from all - 0.11
- merge those two in a single cluster, and re-compute the distance matrix.

Example





Example

- After merging ,(p3, p6) together in a cluster, now there is one entry for (p3, p6) in the table, and no longer have p3 or p6 separately.
- Therefore, re-compute the distance from each point to our new cluster - (p3, p6).
- With the single link method the proximity of two clusters is defined as the minimum of the distance between any two points in the two clusters. Therefore, the distance between let's say (p3, p6) and p1 would be calculated as follows:
- *dist*((p3, p6), p1) = MIN (*dist*(p3, p1) , *dist*(p6, p1))

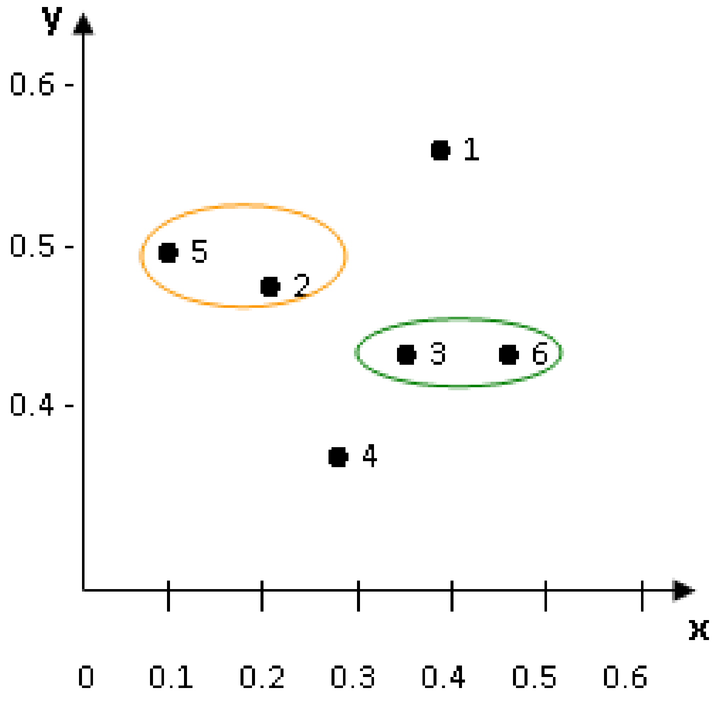
  = MIN (0.22 , 0.23)

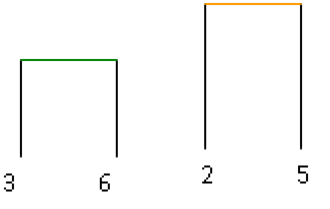
  = 0.22
- Repeat above Step until all clusters are merged.

Example

	p1	p2	(p3, p6)	p4	p5
p1	0				
p2	0.24	0			
(p3, p6)	0.22	0.15	0		
p4	0.37	0.20	0.15	0	
p5	0.34	0.14	0.28	0.29	0

- looking at the distance matrix above, p2 and p5 have the smallest distance from all - 0.14
- So, we merge those two in a single cluster, and re-compute the distance matrix.


y
0.6
0.5
0.4
1
5
2
3
6
4
x
0
0.1
0.2
0.3
0.4
0.5
0.6


3
6
2
5

Example

- Since, we have merged (p2, p5) together in a cluster, we now have one entry for (p2, p5) in the table, and no longer have p2 or p5 separately.
- Therefore re-compute the distance from all other points / clusters to our new cluster - (p2, p5). The distance between (p3, p6) and (p2, p5) would be calculated as follows:
- dist((p3, p6), (p2, p5))

$$= \text{MIN}(\text{dist}(p3, p2), \text{dist}(p6, p2), \text{dist}(p3, p5), \text{dist}(p6, p5))$$

$$= \text{MIN}(0.15, 0.25, 0.28, 0.39)$$

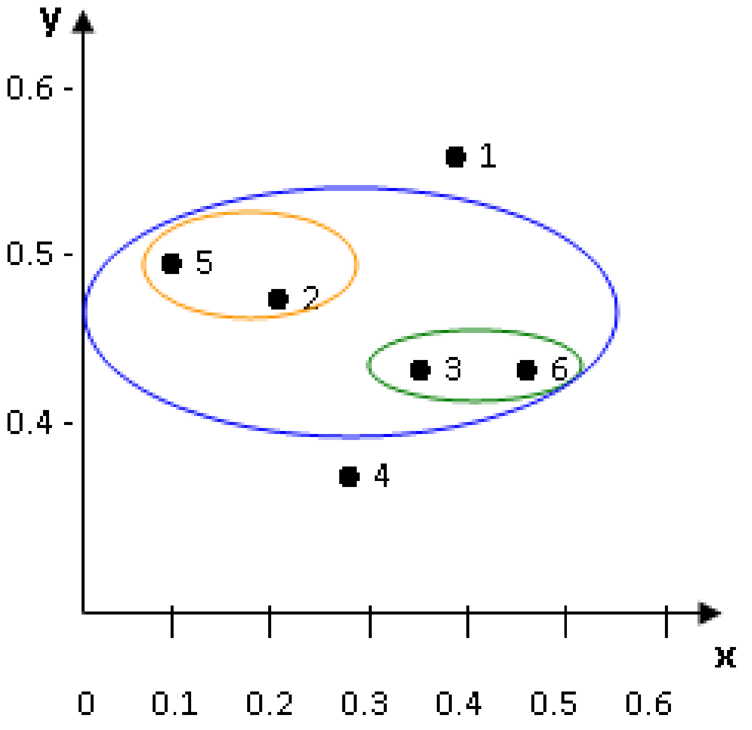
$$= 0.15$$

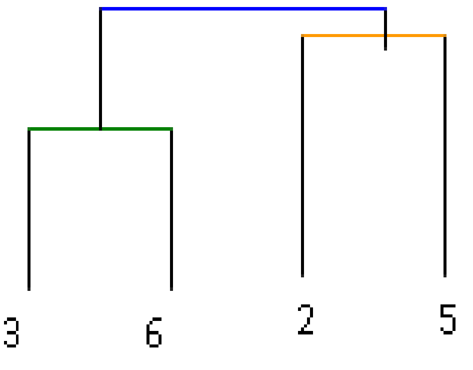
Example

	p1	(p2, p5)	(p3, p6)	p4
p1	0			
(p2, p5)	0.24	0		
(p3, p6)	0.22	0.15	0	
p4	0.37	0.20	0.15	0

- So, looking at the last distance matrix above (p2, p5) and (p3, p6) have the smallest distance from all - 0.15 .
- p4 and (p3, p6) have the same distance - 0.15 .
- In that case, we can pick either one.
- choose (p2, p5) and (p3, p6). So, we merge those two in a single cluster, and re-compute the distance matrix.

Example



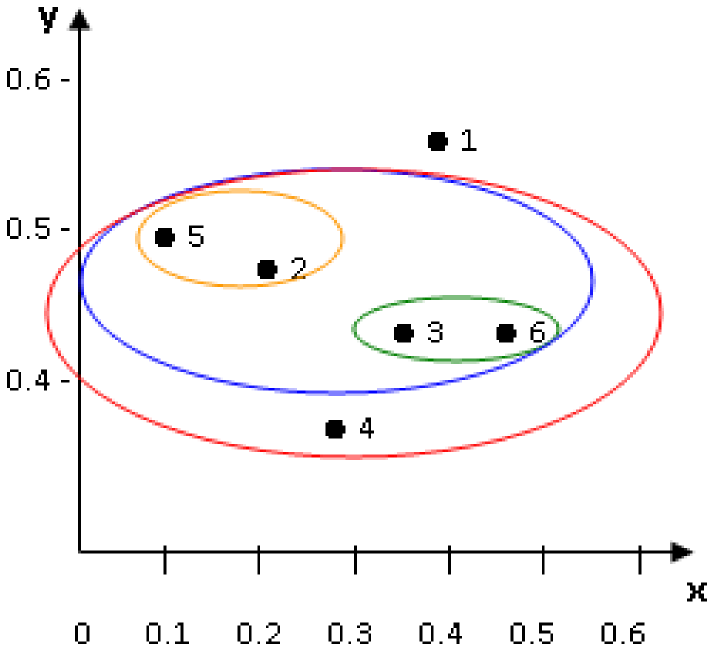


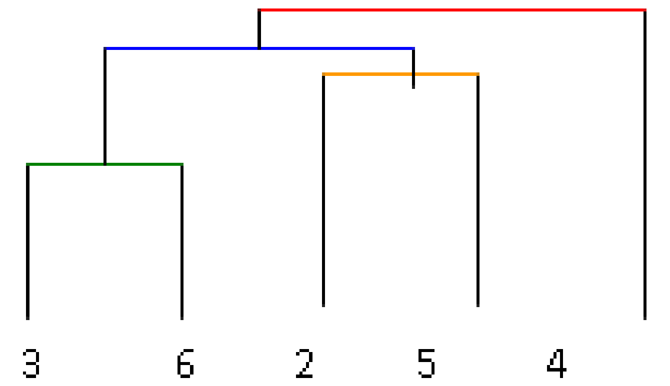
Example

	p1	(p2, p5, p3, p6)	p4
p1	0		
(p2, p5, p3, p6)	0.22	0	
p4	0.37	0.15	0

- So, looking at the last distance matrix above, (p2, p5, p3, p6) and p4 have the smallest distance from all - 0.15.
- So, merge those two in a single cluster, and re-compute the distance matrix.

Example



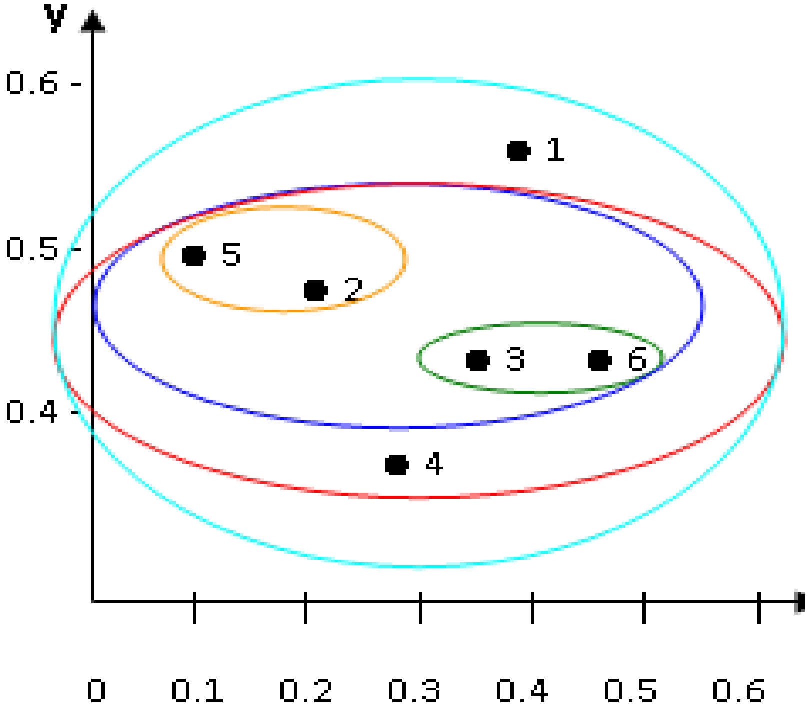


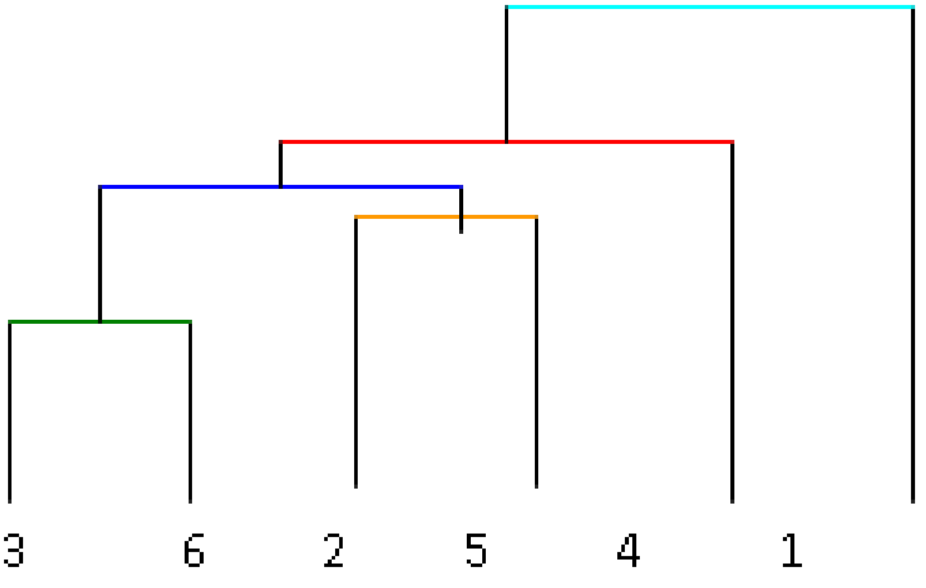
Example

	p1	(p2, p5, p3, p6, p4)
p1	0	
(p2, p5, p3, p6, p4)	0.22	0

- So, looking at the last distance matrix above, we see that (p2, p5, p3, p6, p4) and p1 have the smallest distance - 0.22 (the only one left).
- So, merge those two in a single cluster.
- There is no need to re-compute the distance matrix, as there are no more clusters to merge.

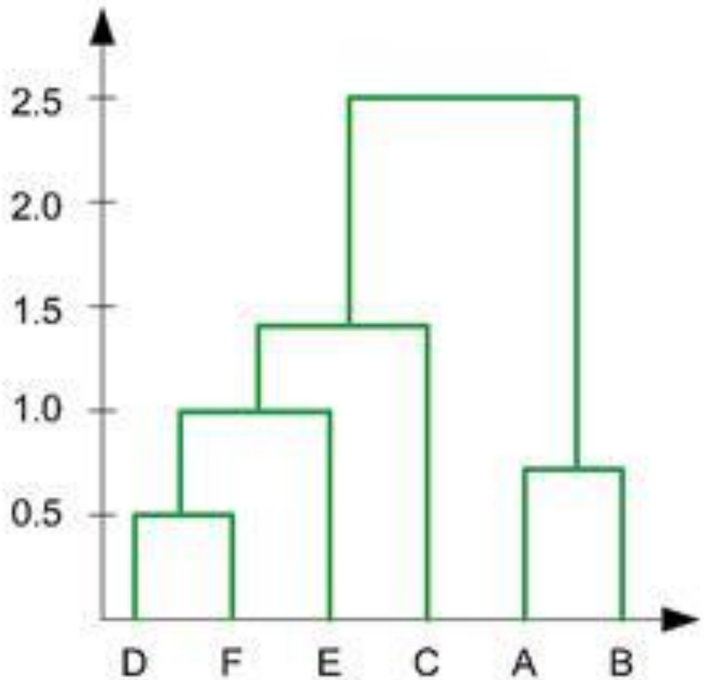
Example

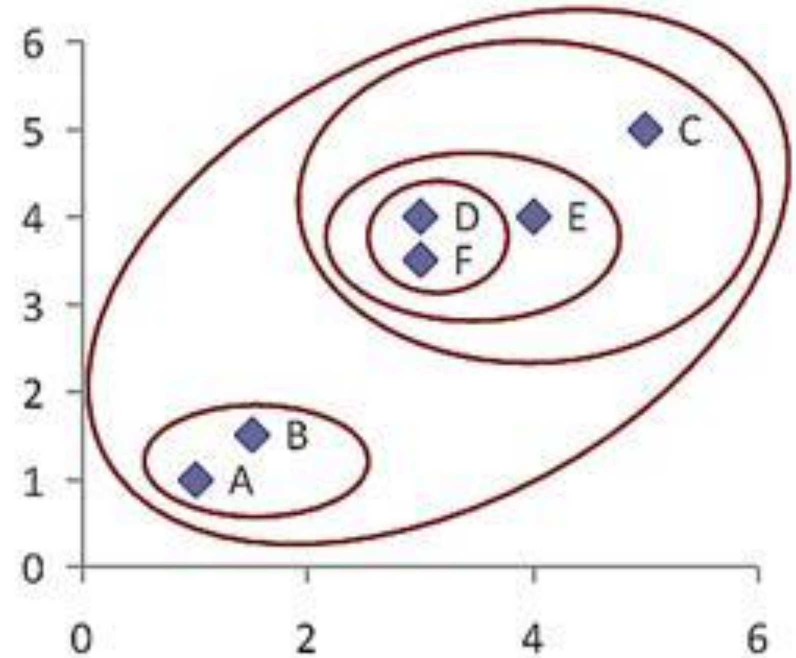




Apply Agnes on following distance matrix

Dist	A	B	C	D	E	F
A	0.00	0.71	5.66	3.61	4.24	3.20
B	0.71	0.00	4.95	2.92	3.54	2.50
C	5.66	4.95	0.00	2.24	1.41	2.50
D	3.61	2.92	2.24	0.00	1.00	0.50
E	4.24	3.54	1.41	1.00	0.00	1.12
F	3.20	2.50	2.50	0.50	1.12	0.00


2.5
2.0
1.5
1.0
0.5
D
F
E
C
A
B


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

Apply Agnes on following data objects

A-1,1

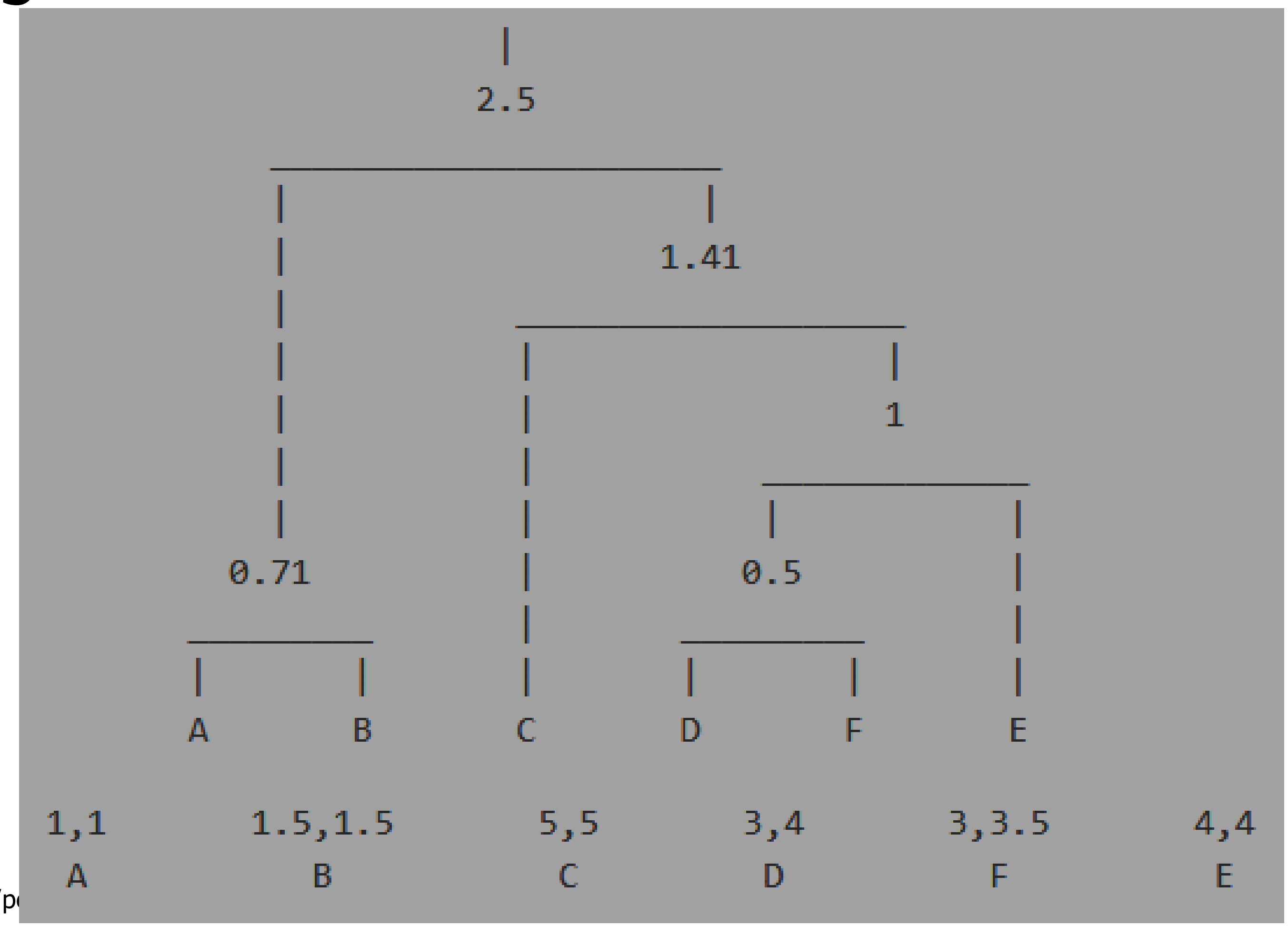
B-1.5,1.5

C-5,5

D-3,4

E-4,4

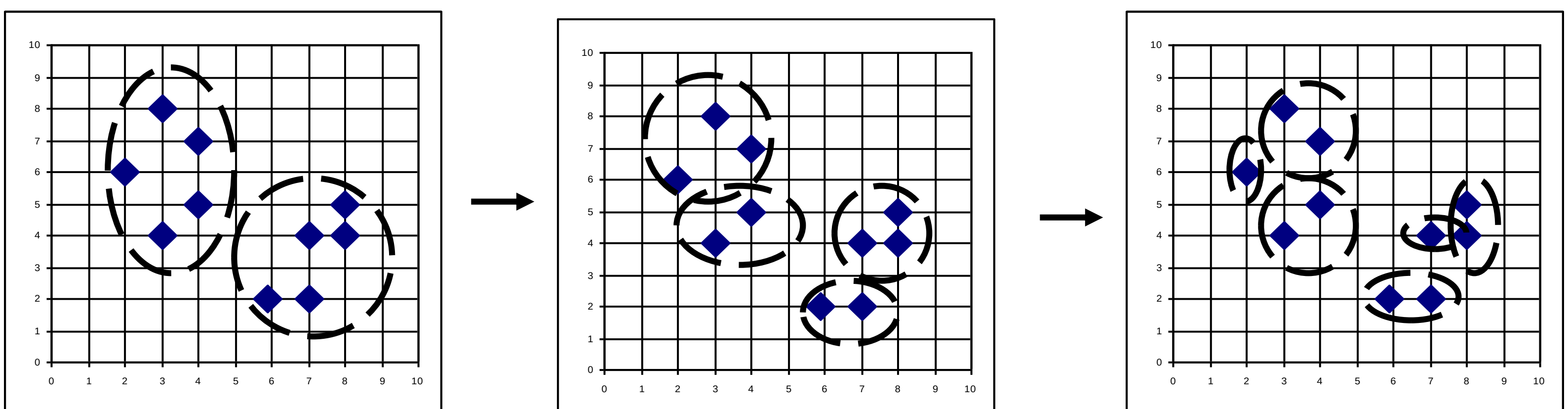
F-3,3.5



https://p

DIANA (Divisive Analysis)

- Implemented in statistical analysis packages, e.g., Splus
- Inverse order of AGNES
- Splits based on maximum euclidean distance between closest neighboring objects in the cluster.
- Splitting process repeats until, Eventually each node forms a cluster on its own



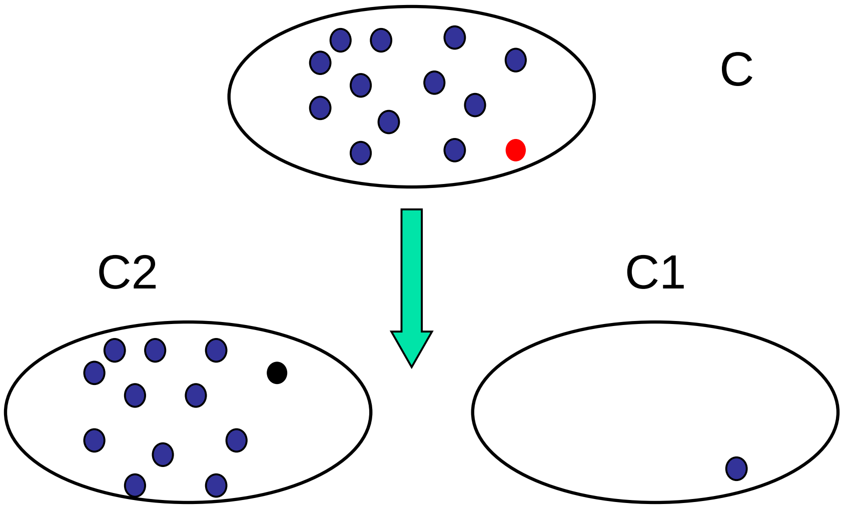
DIANA

- First, all of the objects form one cluster.
- The cluster is split according to some principle, such as the minimum Euclidean distance between the closest neighboring objects in the cluster.
- The cluster splitting process repeats until, eventually, each new cluster contains a single object or a termination condition is met.

Splitting Process of DIANA

Initialization:

1. Choose the object O_h which is most dissimilar to other objects in C.
2. Let C1={O_h}, C2=C-C1.



Splitting Process of DIANA (Cont'd)

Iteration:

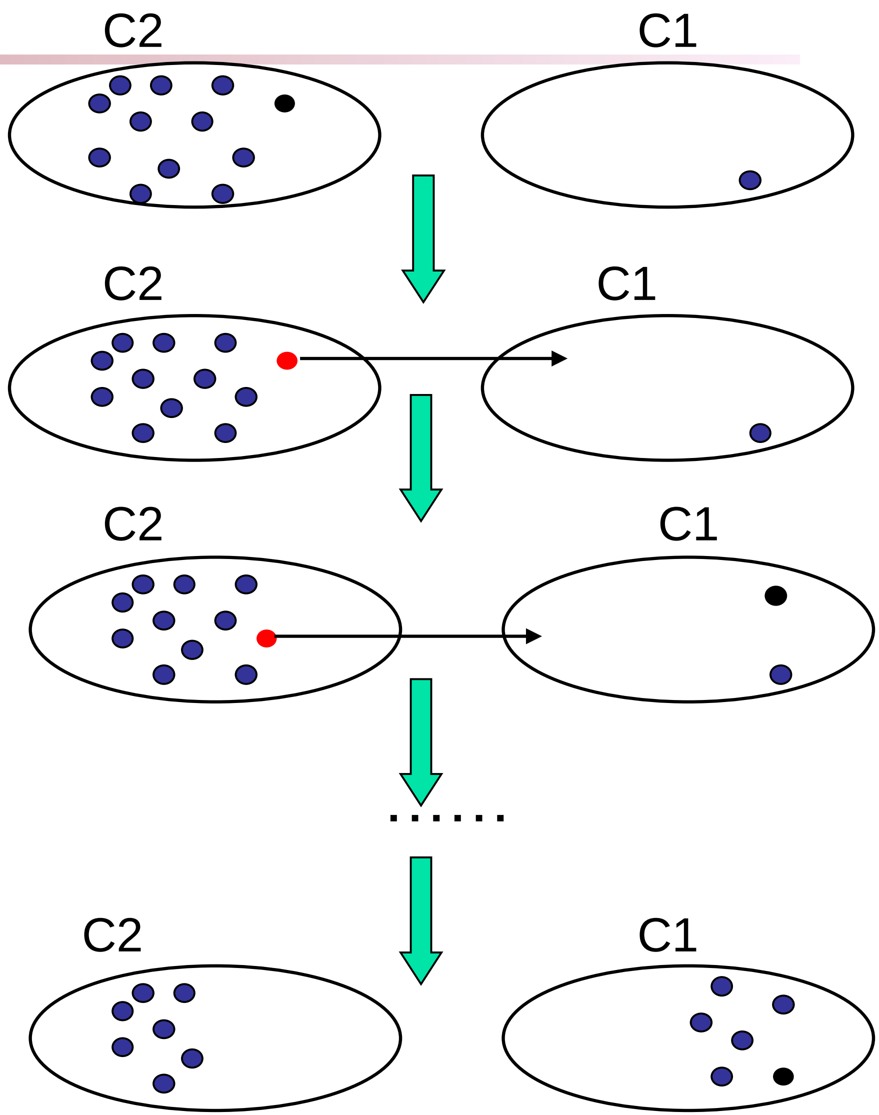
3. For each object Oi in C2, tell whether it is more close to C1 or to other objects in C2

$$D_i = \operatorname*{avg}_{j \in C_2} d(O_i, O_j) - \operatorname*{avg}_{j \in C_1} d(O_i, O_j)$$

4. Choose the object O_k with greatest D score.

5. If $D_k>0$, move Ok from C2 to C1, and repeat 3-5.

6. Otherwise, stop splitting process.



Divisive Hierarchical Clustering (DIANA) Example

- To divide the selected cluster, the algorithm first looks for its most disparate observation (i.e., which has the largest average dissimilarity to the other observations of the selected cluster).
- This observation initiates the splinter group

Sites	Sites 1	2	3	4	5	6	mean
1	0	1.4	9.7	15.9	15.1	13.7	11.16
2	1.4	0	9.3	15.2	14.4	12.7	10.6
3	9.7	9.3	0	10.9	10	13.8	10.74
4	15.9	15.2	10.9	0	2.2	8.2	10.48
5	15.1	14.4	10	2.2	0	8.3	10
6	13.7	12.7	13.8	8.2	8.3	0	$\boxed{11.34}$

Split off site #6

Divisive Hierarchical Clustering (DIANA) Example

Sites	Clusters 1-5	6
1	10.5	13.7
2	10.1	12.7
3	10.0	13.8
4	11.1	8.2
5	10.4	8.3

Clusters: (1,2,3) and (4,5,6)

Sites	Sites 1	2	3	*mean*
1		1.4	9.7	5.55
2	1.4		9.3	5.35
3	9.7	9.3		$\boxed{9.5}$

Split off site #3

Sites	Clusters 1-2	3
1	1.4	$\boxed{9.7}$
2	1.4	9.3

Clusters: (1,2) and (3)

Divisive Hierarchical Clustering (DIANA)
Example

Sites	4	5	6	*mean*
4		2.2	8.2	*5.2*
5	2.2		8.3	*5.25*
6	8.2	8.3		*8.25*

Sites	4-5	6
4	2.2	8.2
5	2.2	8.3

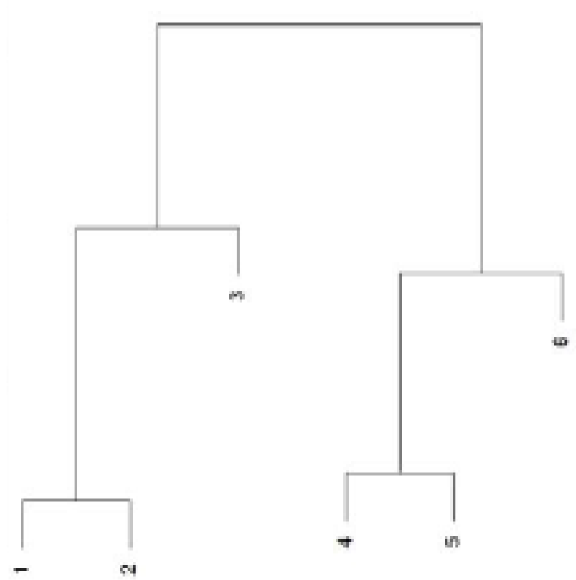
- Split off site #6 **Clusters: (4,5) and (6)**

Sites	1	2
1		1.4
2	1.4	

Split 1 and 2

Sites	4	5
4		2.2
5	2.2	

Split 4 and 5



Dendrogran

Agglomerative and Divisive

Agglomerative Hierarchical Clustering

- Bottom-up strategy
- Each cluster starts with only one object
- Clusters are merged into larger and larger clusters until:
  - All the objects are in a single cluster
  - Certain termination conditions are satisfied

Divisive Hierarchical Clustering

- Top-down strategy
- Start with all objects in one cluster
- Clusters are subdivided into smaller and smaller clusters until:
  - Each object forms a cluster on its own
  - Certain termination conditions are satisfied

Apply DIANA on following distance matrix

Dist	A	B	C	D	E	F
A	0.00	0.71	5.66	3.61	4.24	3.20
B	0.71	0.00	4.95	2.92	3.54	2.50
C	5.66	4.95	0.00	2.24	1.41	2.50
D	3.61	2.92	2.24	0.00	1.00	0.50
E	4.24	3.54	1.41	1.00	0.00	1.12
F	3.20	2.50	2.50	0.50	1.12	0.00

Hierarchical Clustering

- Strengths
  - Do not need to input k, the number of clusters
- Weakness
  - Do not scale well; time complexity of at least $O(n^2)$, where n is total number of objects
  - Can never undo what was done previously

Recent Hierarchical Clustering Methods

- Integration of hierarchical with distance-based clustering
  - BIRCH (1996): uses CF-tree and incrementally adjusts the quality of sub-clusters

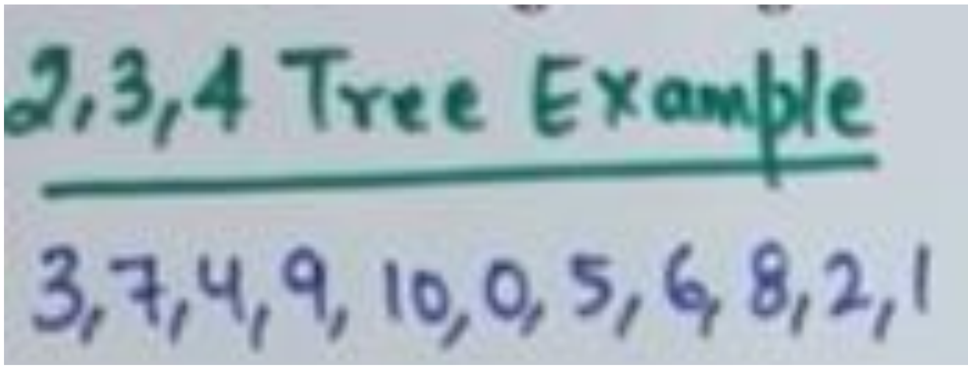
BIRCH (1996)

- Birch: Balanced Iterative Reducing and Clustering using Hierarchies
- BIRCH is designed for clustering a large amount of numerical data by integration of hierarchical clustering and other clustering methods such as iterative partitioning
- BIRCH introduced two concepts clustering feature and clustering feature tree which are used to summarize cluster representation.
- These structures help the clustering method achieve good speed and scalability in large databases and also make it effective for incremental and dynamic clustering of incoming objects.

BIRCH (1996)

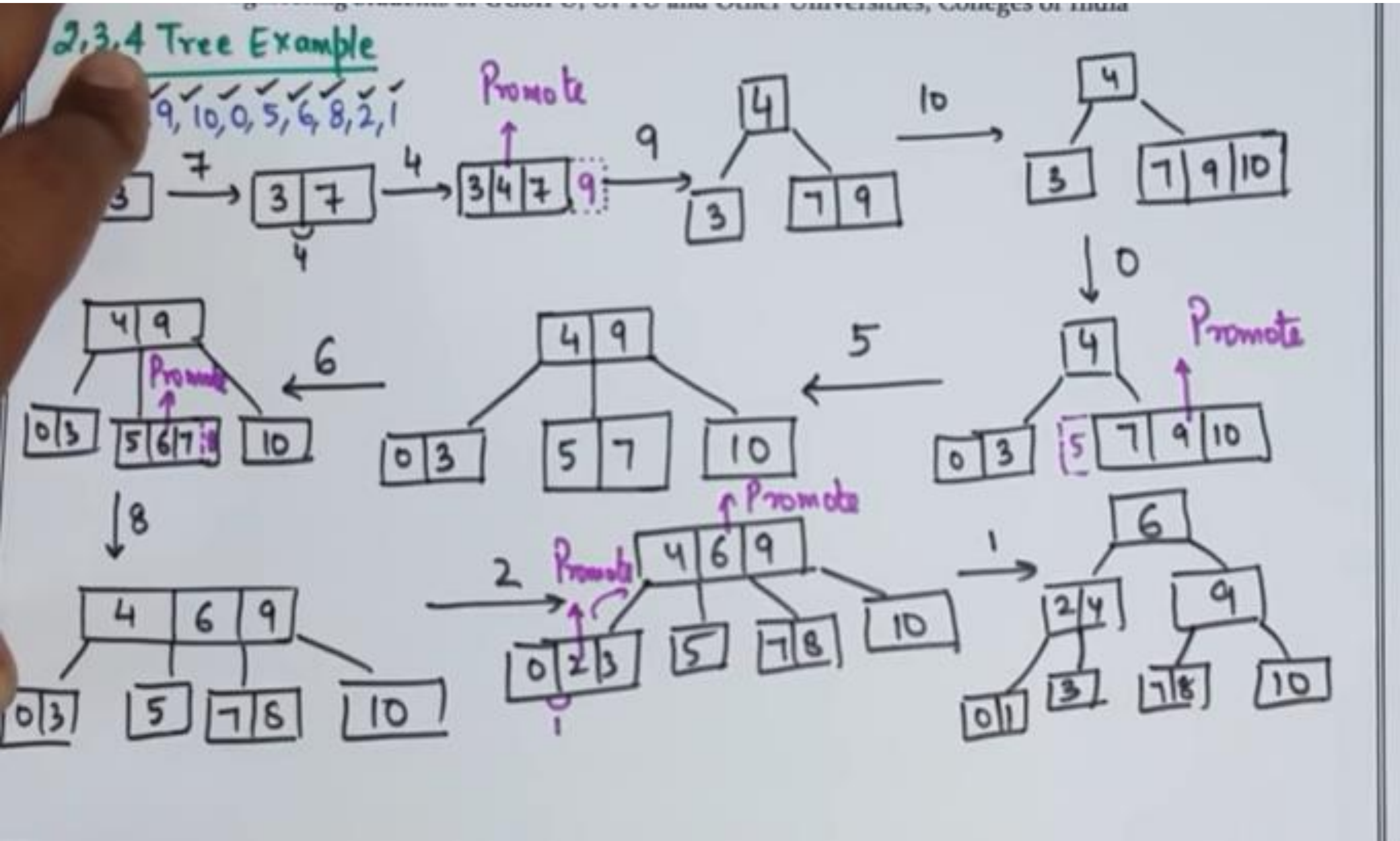
- Incrementally construct a CF (Clustering Feature) tree, a hierarchical data structure for multiphase clustering
  - Phase 1: scan DB to build an initial in-memory CF tree
  - Phase 2: use an arbitrary clustering algorithm to cluster the leaf nodes of the CF-tree
- *Scales linearly*: finds a good clustering with a single scan and improves the quality with a few additional scans
- *Weakness:* handles only numeric data, and sensitive to the order of the data record.

- https://www.youtube.com/watch?v=9tioidySJmM



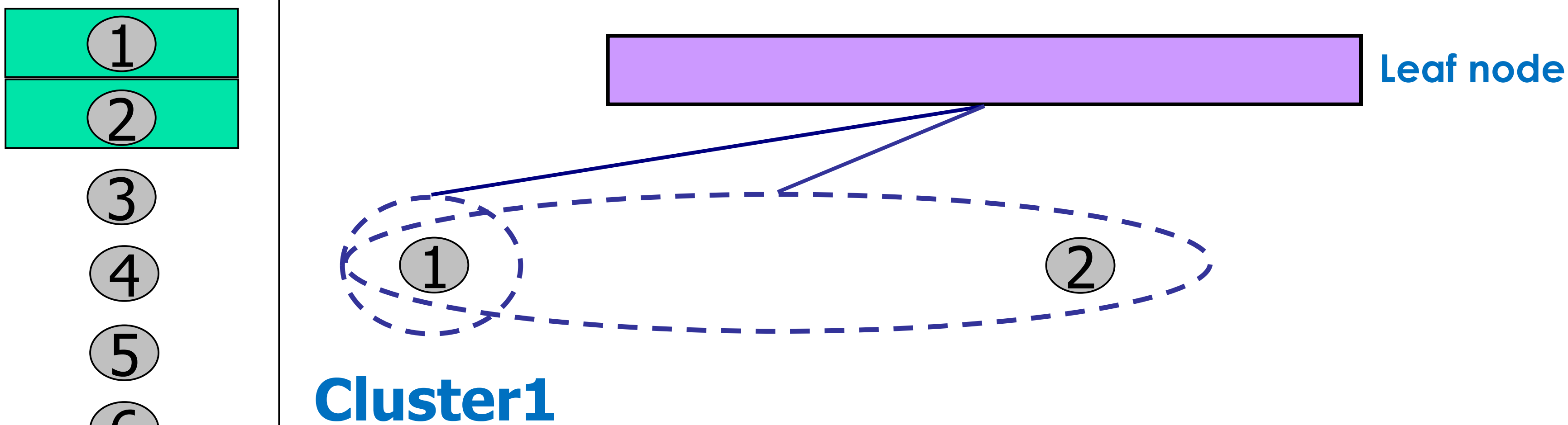
2,3,4 Tree Example

3, 7, 4, 9, 10, 0, 5, 6, 8, 2, 1



BIRCH: The Idea by example

Data Objects | **Clustering Process (build a tree)**

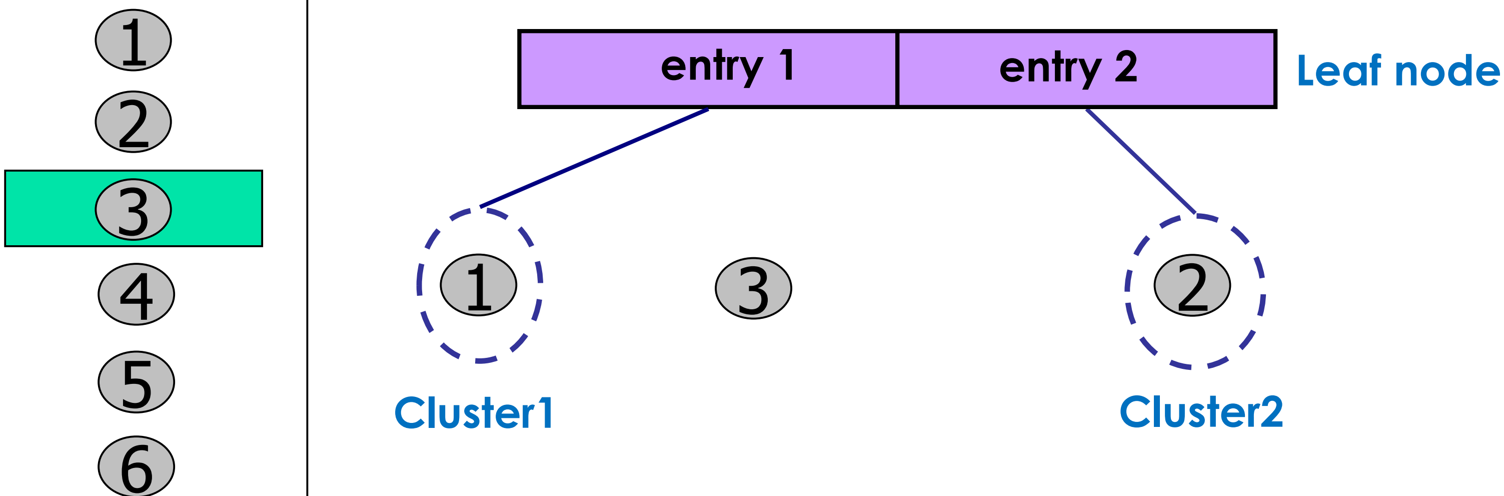


If cluster 1 becomes too large (not compact) by adding object 2, then split the cluster

BIRCH: The Idea by example

Data Objects

Clustering Process (build a tree)

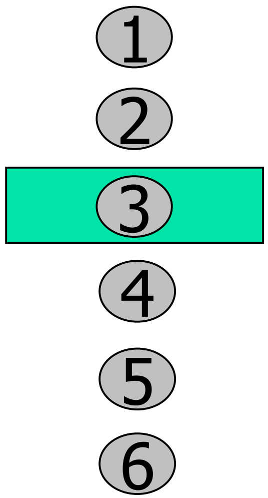


entry1 is the closest to object 3

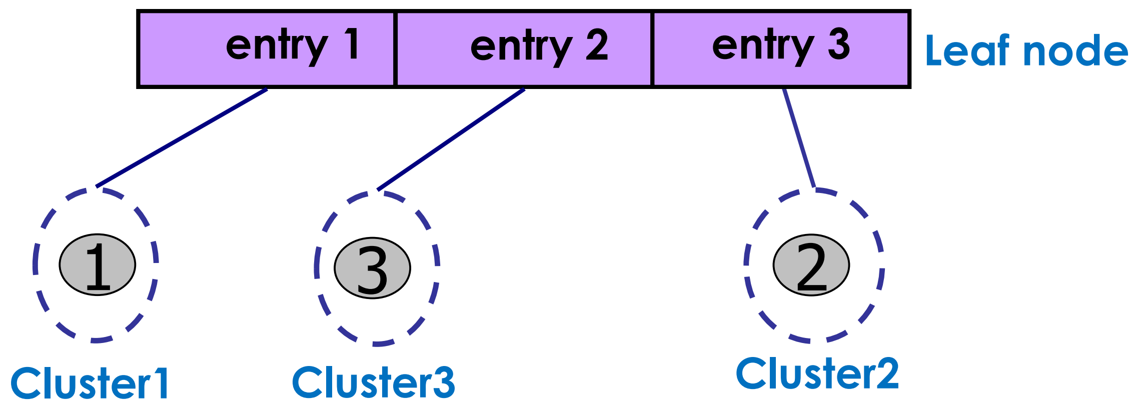
If cluster 1 becomes too large by adding object 3, then split the cluster

BIRCH: The Idea by example

Data Objects



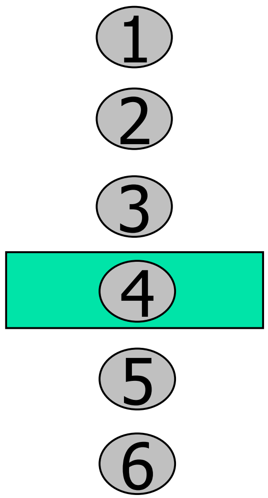
Clustering Process (build a tree)



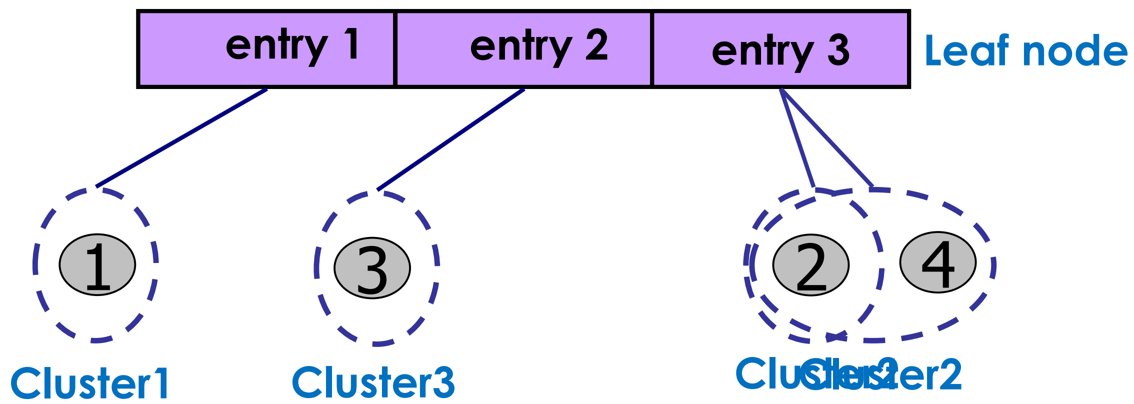
Leaf node with three entries

BIRCH: The Idea by example

Data Objects



Clustering Process (build a tree)



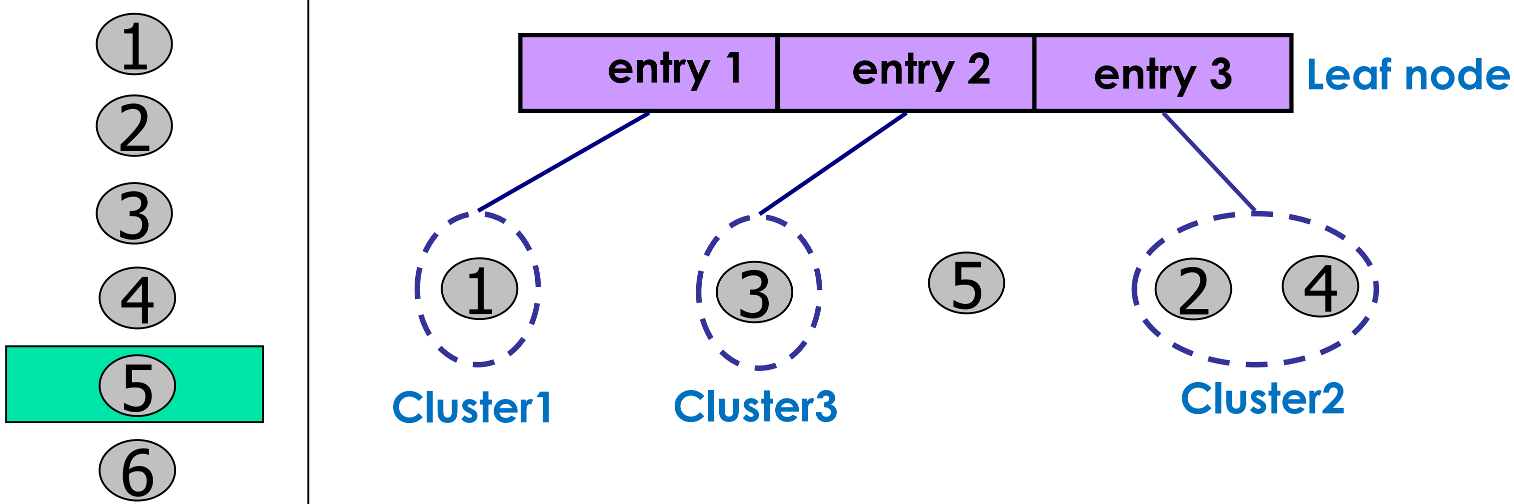
entry3 is the closest to object 4

Cluster 2 remains compact when adding object 4 then add object 4 to cluster 2

BIRCH: The Idea by example

Data Objects

Clustering Process (build a tree)

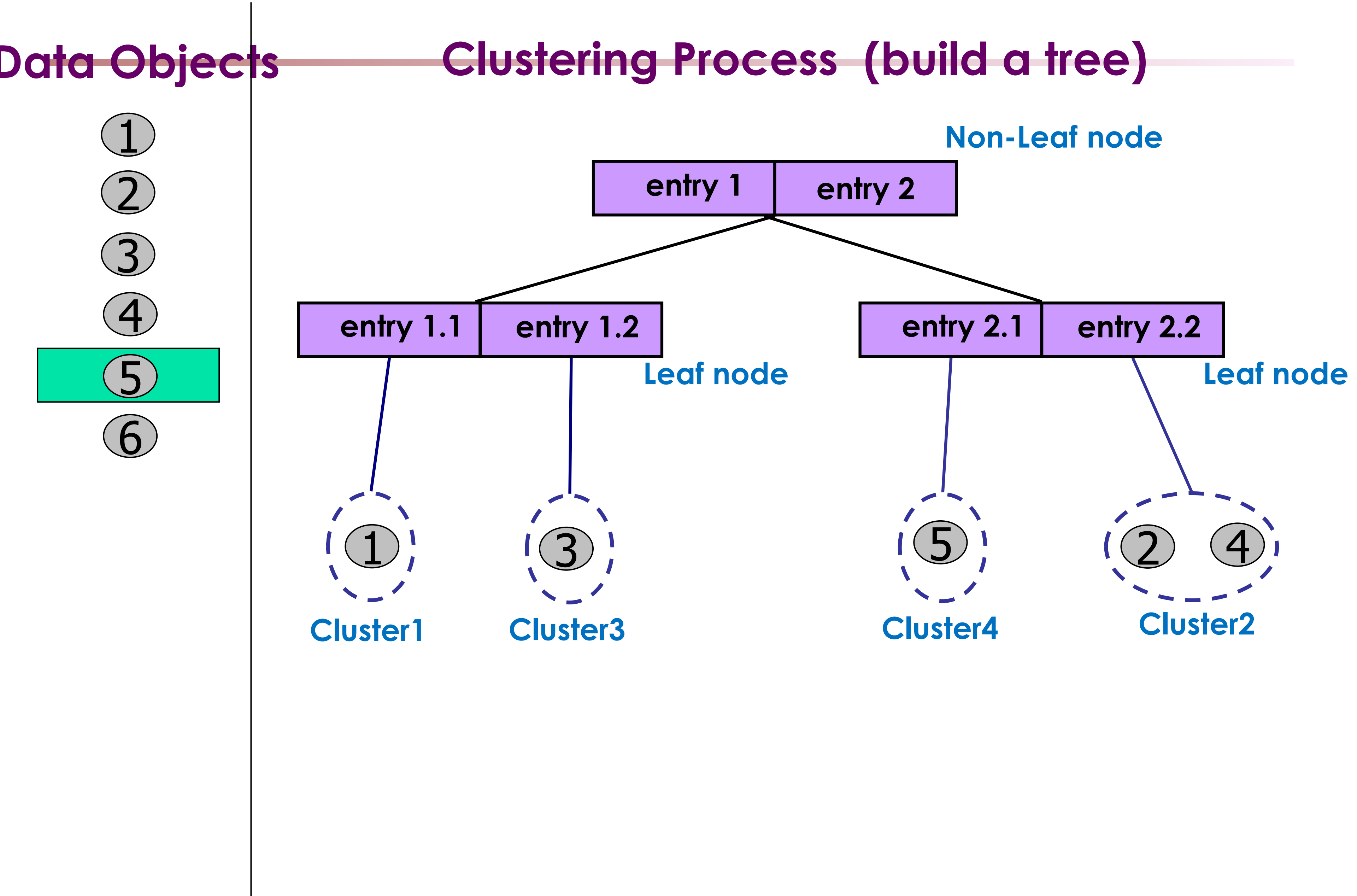


entry2 is the closest to object 5

Cluster 3 becomes too large by adding object 5
then split cluster 3?

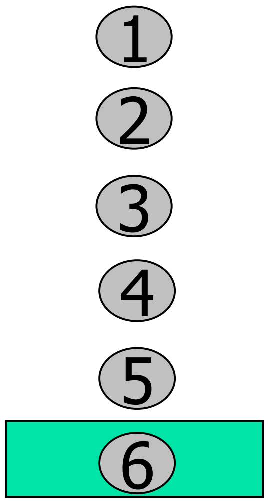
BUT there is a limit to the number of entries a node can have
Thus, split the node

BIRCH: The Idea by example

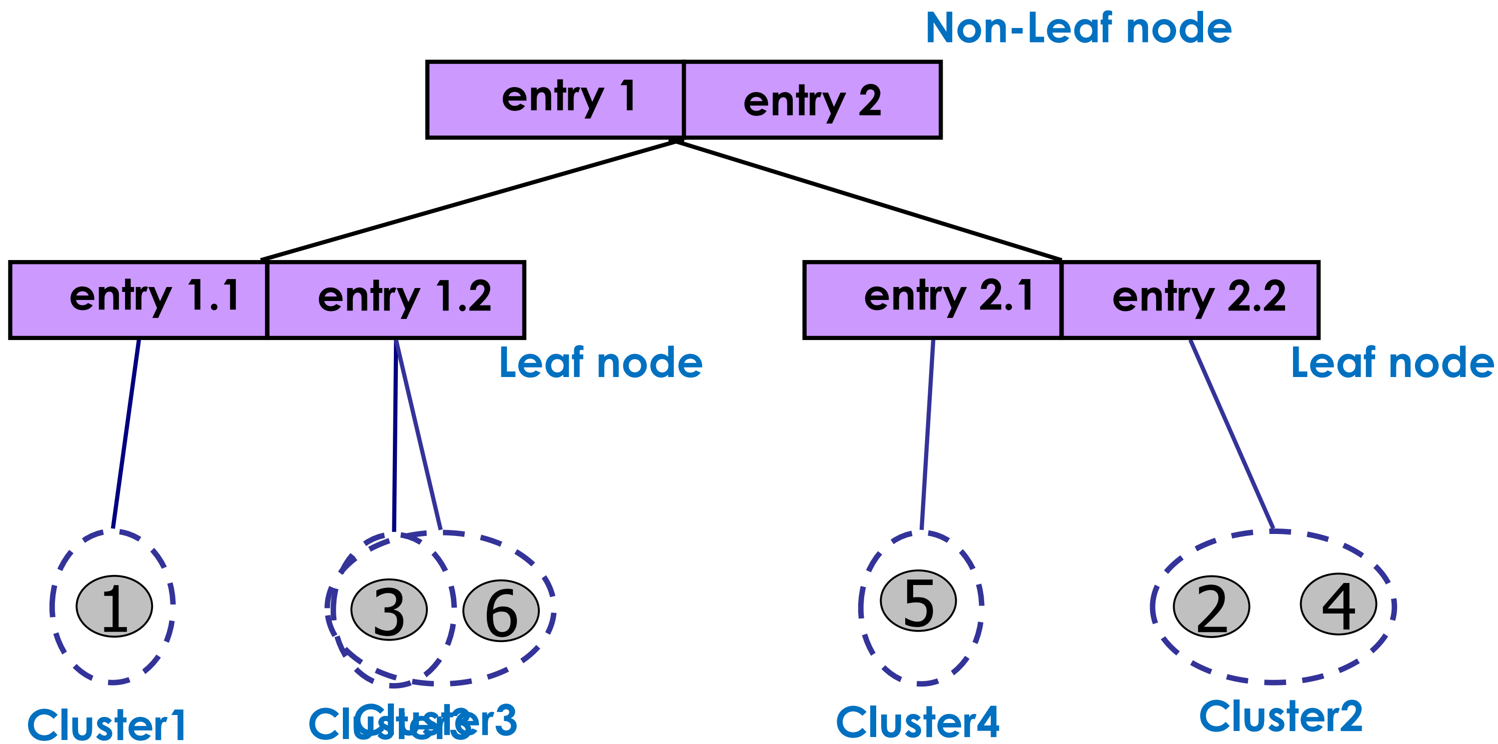


BIRCH: The Idea by example

Data Objects



Clustering Process (build a tree)



entry1.2 is the closest to object 6

Cluster 3 remains compact when adding object 6 then add object 6 to cluster 3

Clustering Feature Vector in BIRCH

Clustering Feature: $CF = (N, \overrightarrow{LS}, SS)$

N: **Number of data points**

LS: $\sum^{N}_{i=1} = \overrightarrow{X_i}$

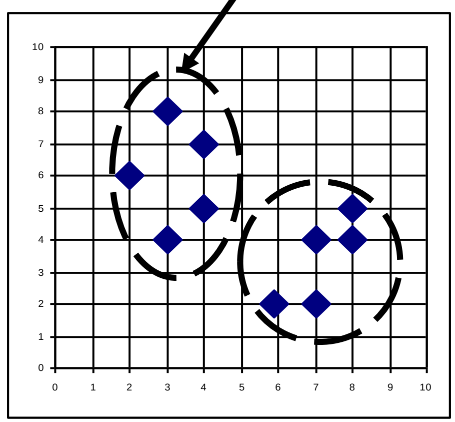
SS: $\sum^{N}_{i=1} = \overrightarrow{X_i}^2$

$$R = \sqrt{\frac{\sum (x_i - \bar{x})^2}{n}}$$

OR

$$R = \sqrt{\frac{SS - (LS)^2/n}{n}}$$

CF = (5, (16,30),(54,190))



(3,4)
(2,6)
(4,5)
(4,7)
(3,8)

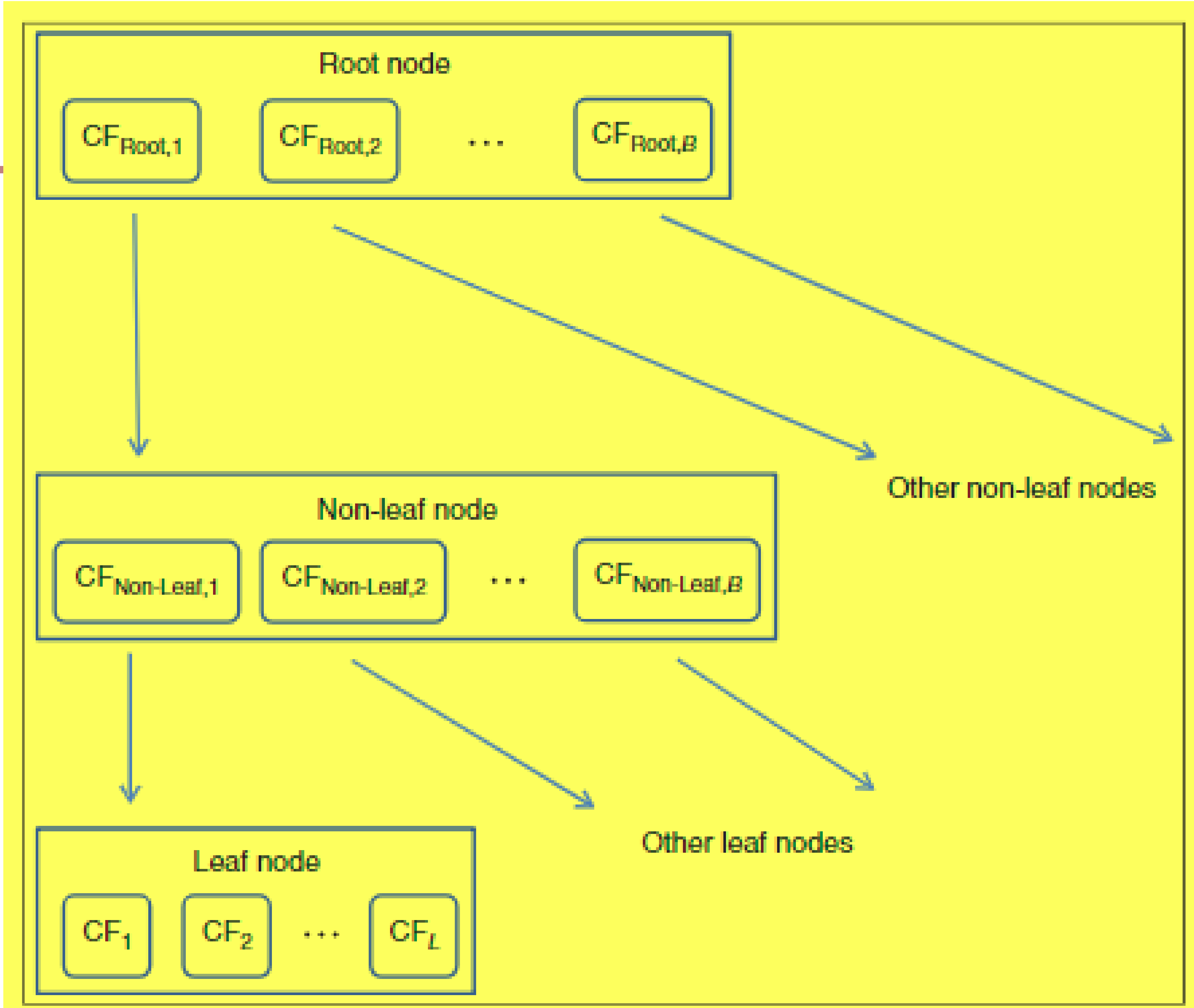
1) For each given record, BIRCH compares the location of that record with the location of each CF in the root node, using either the linear sum or the mean of the CF. BIRCH passes the incoming record to the root node CF closest to the incoming record.

2) The record then descends down to the non-leaf child nodes of the root node CF selected in step 1. BIRCH compares the location of the record with the location of each non-leaf CF. BIRCH passes the incoming record to the non-leaf node CF closest to the incoming record.

3) The record then descends down to the leaf child nodes of the non-leaf node CF selected in step 2. BIRCH compares the location of the record with the location of each leaf. BIRCH tentatively passes the incoming record to the leaf closest to the incoming record.

4) Perform one of (a) or (b):

   a) If the radius (defined below) of the chosen leaf including the new record doesnot exceed the Threshold T , then the incoming record is assigned to that leaf. The leaf and all of its parent CFs are updated to account for the new data point.
   b) If the radius of the chosen leaf including the new record does exceed the Threshold T , then a new leaf is formed, consisting of the incoming record only. The parent CFs are updated to account for the new data point


Root node
CF Root,1
CF Root,2
CF Root,B
Other non-leaf nodes
Non-leaf node
CF Non-Leaf,1
CF Non-Leaf,2
CF Non-Leaf,B
Other leaf nodes
Leaf node
CF 1
CF 2
CF L

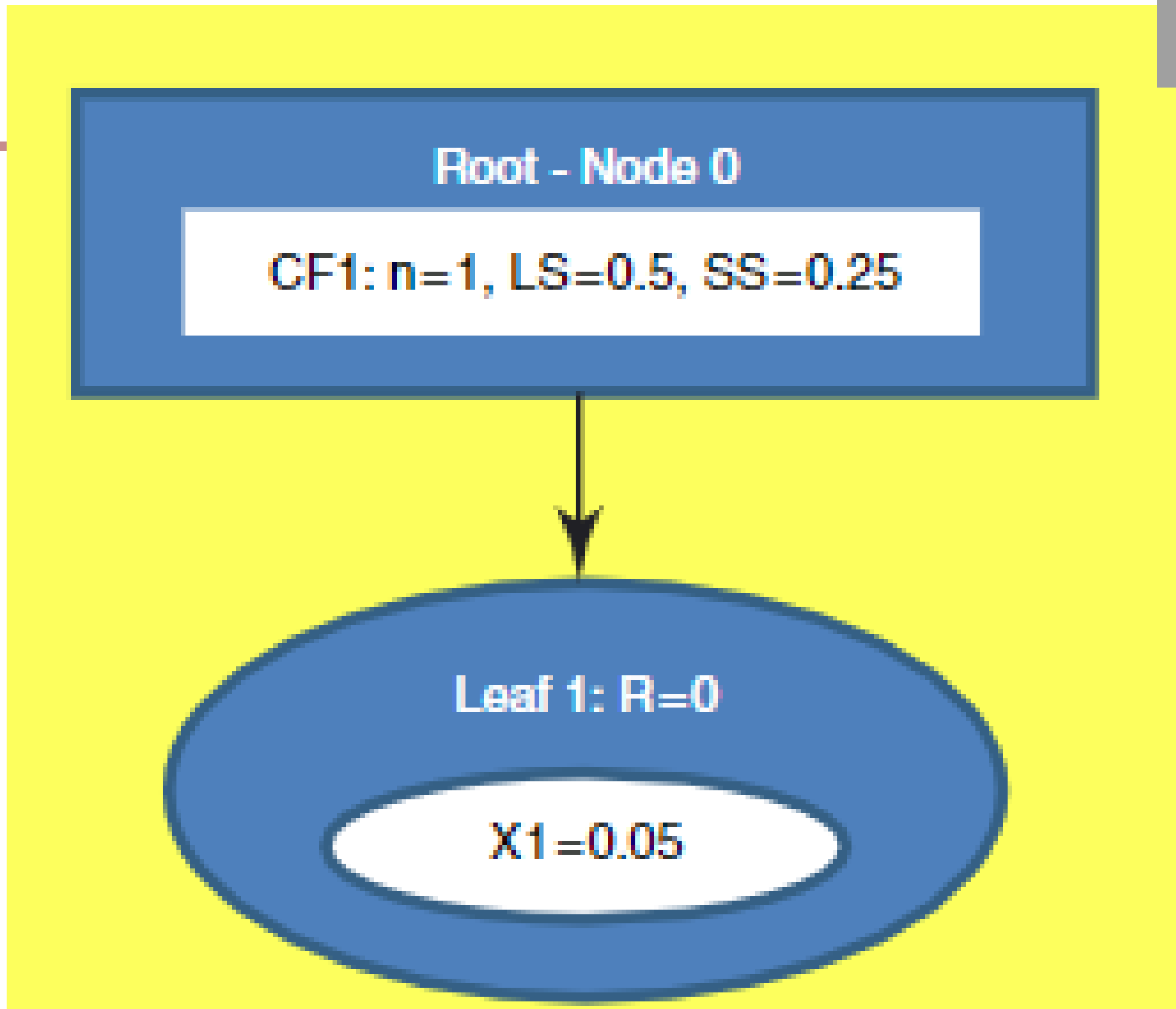
x1 = 0.5 x2 = 0.25 x 3= 0 x4 = 0.65 x5 = 1 x6 = 1.4 x7 = 1.1

Let us define our CF tree parameters as follows:

- Threshold T=0.15;
  no leaf may exceed 0.15 in radius.
- Number of entries in a leaf node L=2.
- Branching factor B=2;
  maximum number of child nodes for each non-leaf node.

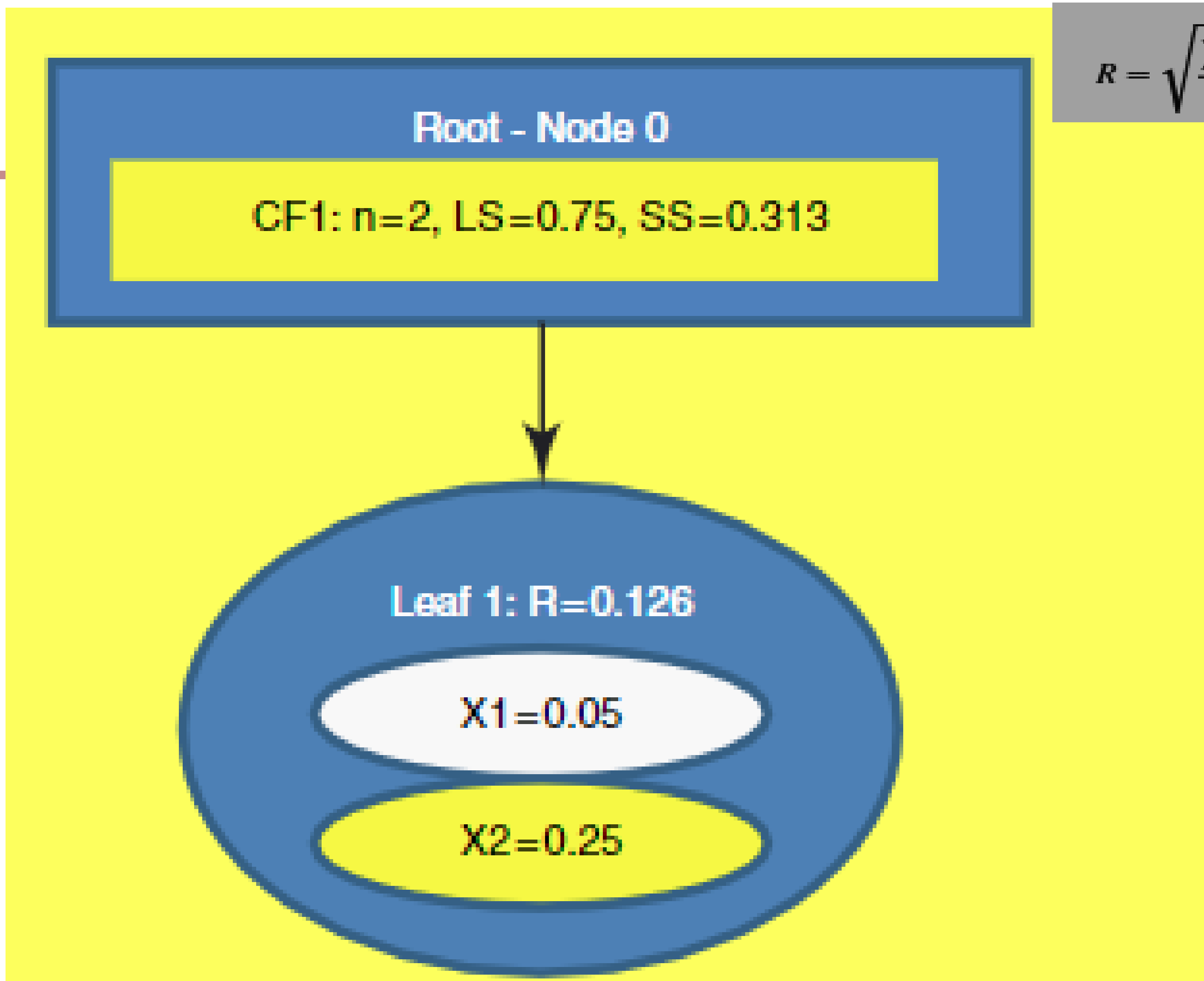
The first data value x = 0.5 is entered.
The root node is initialized with the 1 CF values of the first data value.
A new leaf Leaf1 is created, and BIRCH assigns the first record x to Leaf1.
Because it contains only one record,
the radius of 1 Leaf1 is zero, and thus less than T=0.15.

$$R = \sqrt{\frac{\sum (x_i - \overline{x})^2}{n}}$$



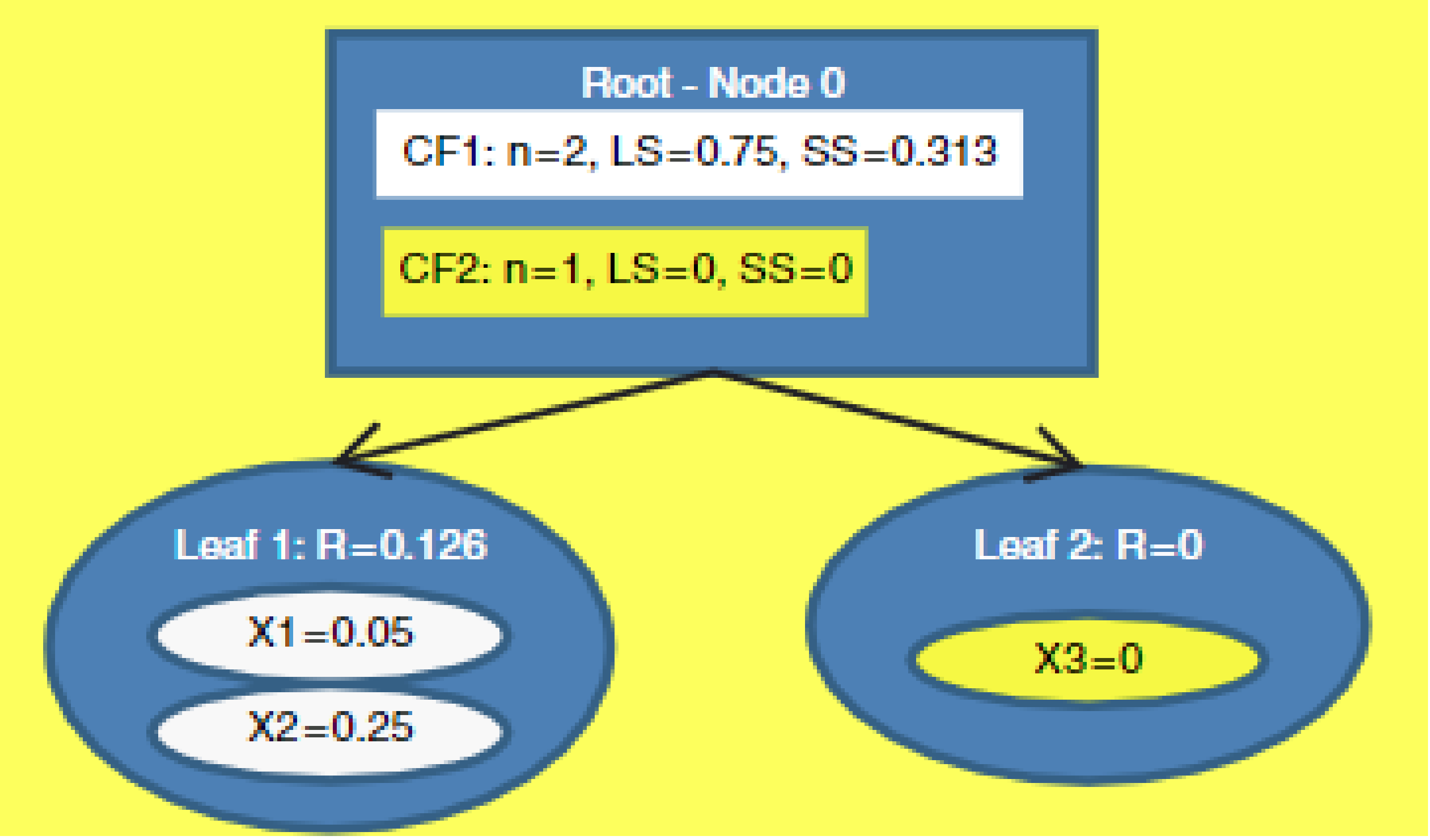
CF Tree after the first data value is entered

$$R = \sqrt{\frac{\sum (x_i - \overline{x})^2}{n}}$$



Second data value entered:

T=0.15 is exceeded, so x3 is not assigned to Leaf1. Instead, a new leaf is initialized, called Leaf2



Third data value entered: A new leaf is initialized

The fourth data value x4=0.65 is entered

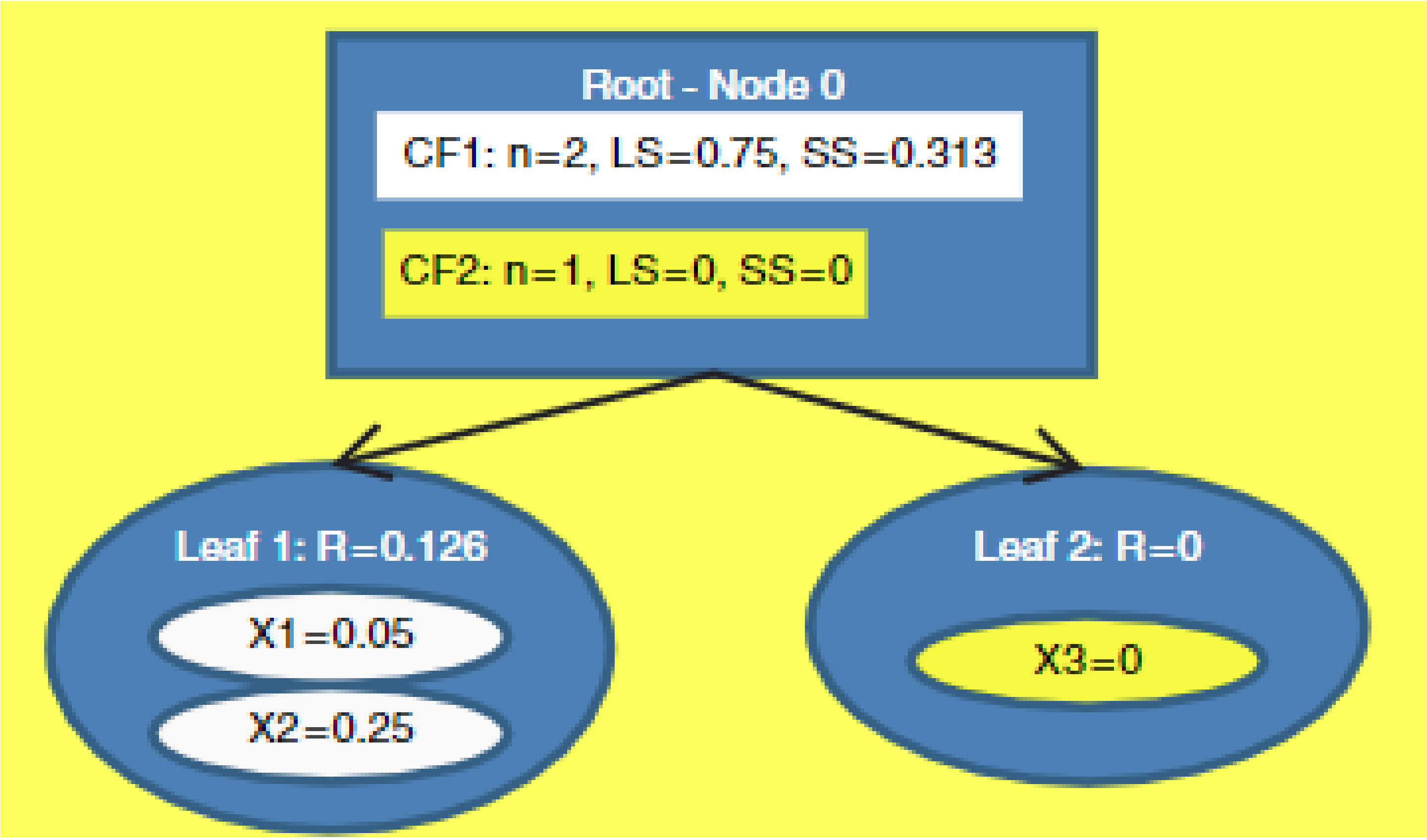
BIRCH compares x4 to the locations of CF1 and CF2.

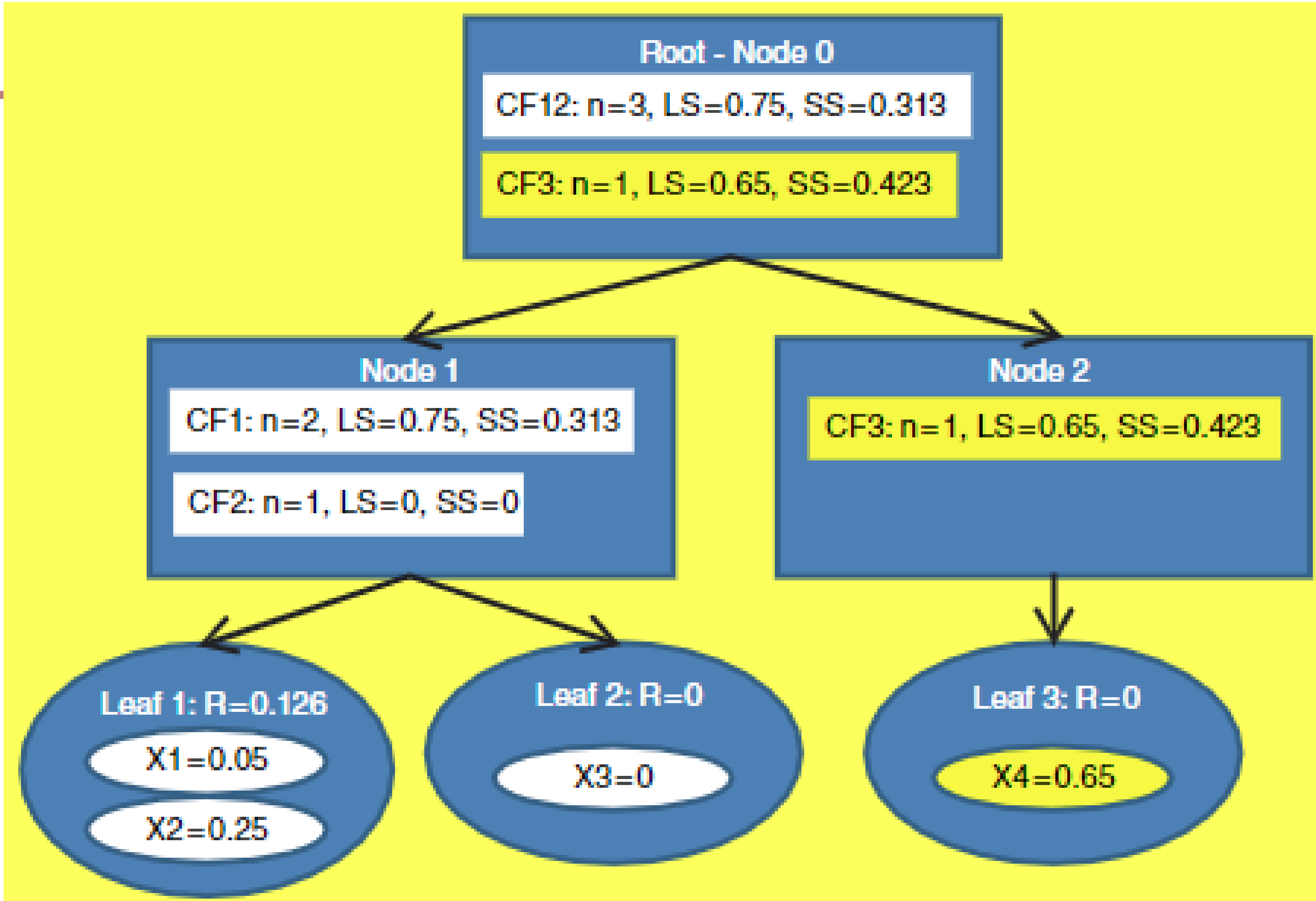
The location is measured by $x = LS / n$.

We have $x_{CF1} = 0.75/2 = 0.375$ and

$x_{CF2} = 0/1 = 0$.

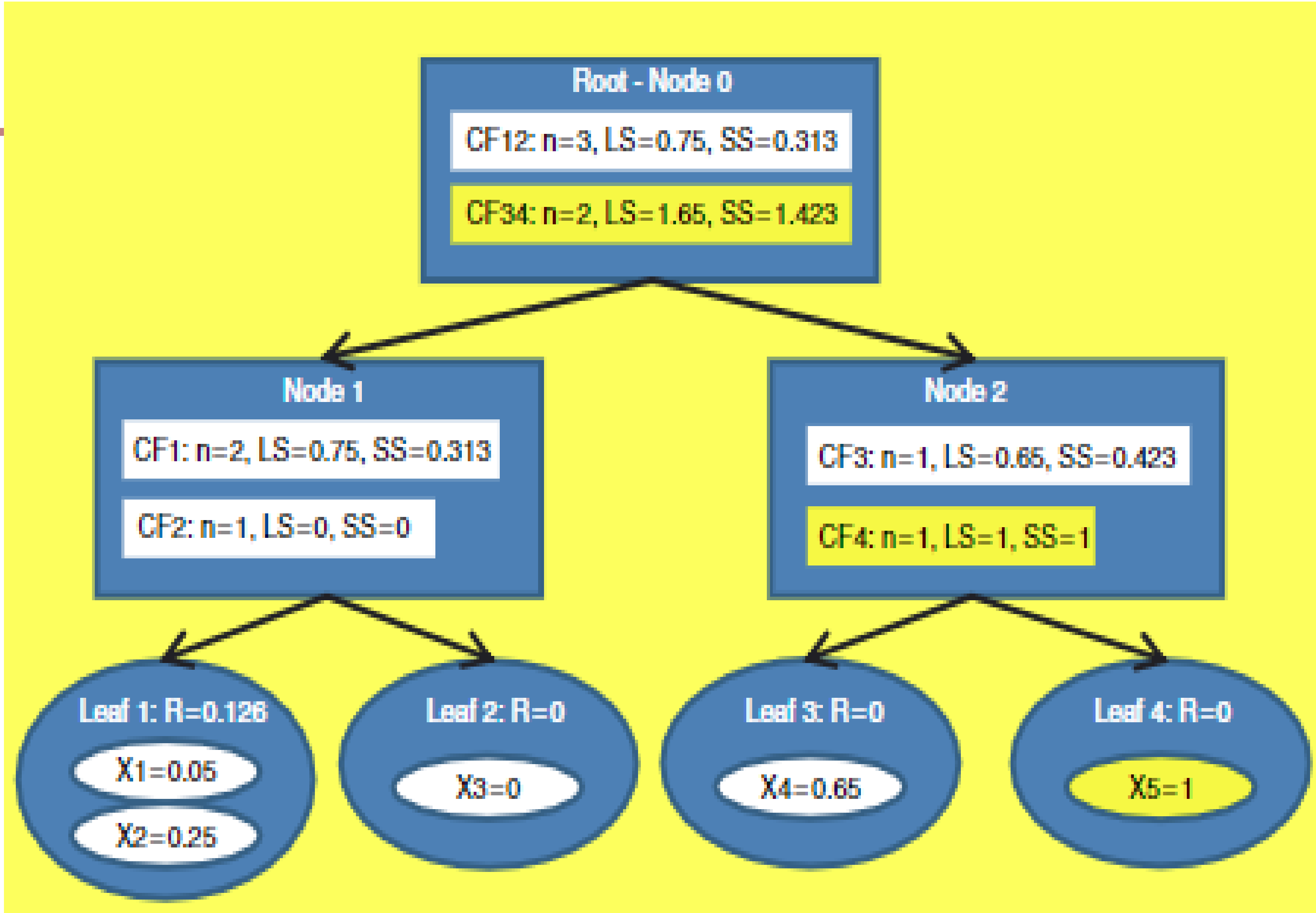
The data point x4 = 0.65 is thus closer to CF1 than to CF2




Root - Node 0
CF12: n=3, LS=0.75, SS=0.313
CF3: n=1, LS=0.65, SS=0.423
Node 1
CF1: n=2, LS=0.75, SS=0.313
CF2: n=1, LS=0, SS=0
Node 2
CF3: n=1, LS=0.65, SS=0.423
Leaf 1: R=0.126
X1=0.05
X2=0.25
Leaf 2: R=0
X3=0
Leaf 3: R=0
X4=0.65

The fifth data value $x_5 = 1$ is entered. BIRCH compares $x_5 = 1$ with the location of CF_{12} and CF_3. We have $\bar{x}_{CF_{12}} = 0.75/3 = 0.25$ and $\bar{x}_{CF_4} = 0.65/1 = 0.65$. The data point $x_5 = 1$ is thus closer to CF_3 than to CF_{12}. BIRCH passes x_5 to

CF_3. The radius of CF_3 now increases to $R = 0.175 > T = 0.15$, so x_5 cannot be assigned to CF_3. Instead, a new leaf in leaf node *Leaf*4 is initialized, with CF, CF_4, containing x_5 only.

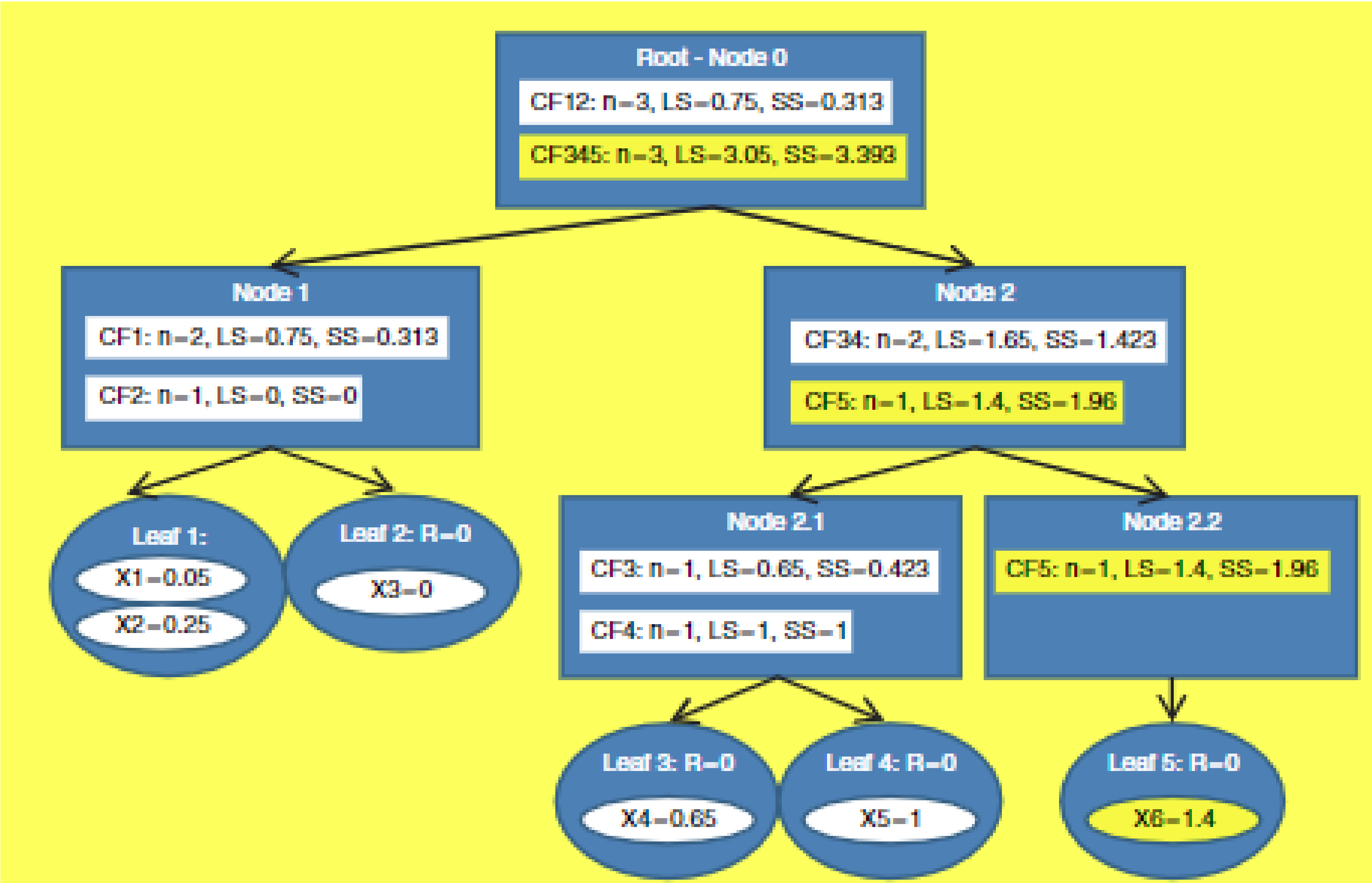


Fifth data value entered: Another new leaf is initialized

The sixth data value $x_6 = 1.4$ is entered. At the root node, BIRCH compares $x_6 = 1.4$ with the location of CF_{12} and CF_{34}. We have $\bar{x}_{CF_{12}} = 0.75/3 = 0.25$ and $\bar{x}_{CF_{34}} = 1.65/2 = 0.825$. The data point $x_6 = 1.4$ is thus closer to CF_{34},

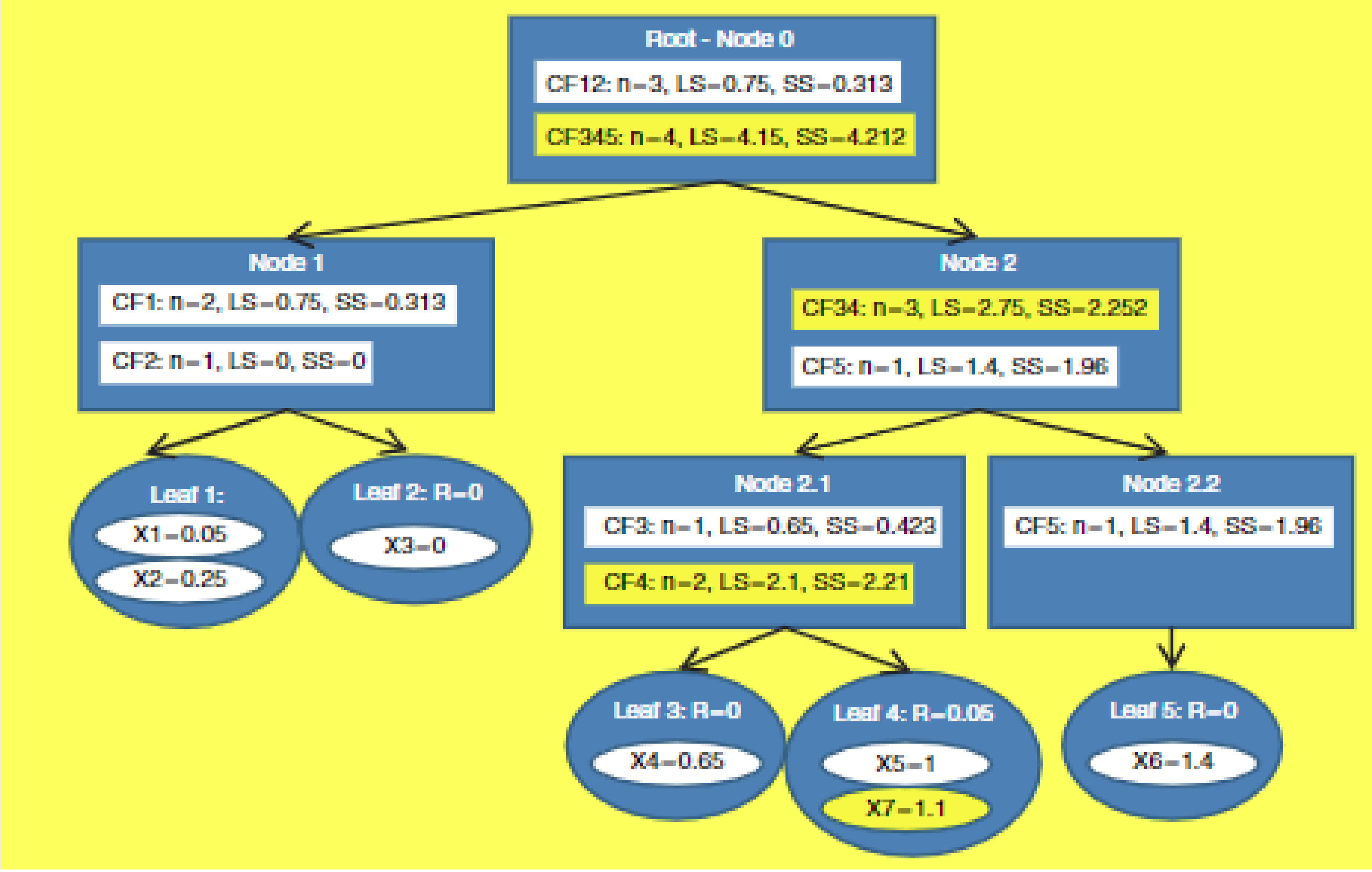
passes x_6 to CF_{34}. The record descends to *Node* 2, and BIRCH compares $x_6 = 1.4$ with the location of CF_3 and CF_4. We have $\bar{x}_{CF_3} = 0.65$ and $\bar{x}_{CF_4} = 1$. The data point $x_6 = 1.4$ is thus closer to CF_4 than to CF_3. BIRCH tentatively passes x_6 to CF_4.

radius of CF_4 now increases to $R = 0.2 > T = 0.15$. The Threshold value $T = 0.15$ is exceeded, so x_6 is not assigned to CF_4. But the branching factor $B = 2$ means that we may have at most two leaf nodes branching off of any non-leaf node. Therefore, we will need a new set of non-leaf nodes, *Node*2.1 and *Node*2.2, branching off from *Node*2. *Node*2.1 contains CF_3 and CF_4, while *Node*2.2 contains the desired new CF_5 and the new leaf node *Leaf 5* as its only child, containing only the information from x_6.



Sixth data value entered: A new leaf node is needed, as are a new non-leaf node and a root node.

the last data value $x_7 = 1.1$ is entered. In the root node, BIRCH compares $x_7 = 1.1$ with the location of CF_{12} and CF_{345}. We have $\bar{x}_{CF_{12}} = 0.25$ and $\bar{x}_{CF_{345}} = 1.02$, so that $x_7 = 1.1$ is closer to CF_{345}, and BIRCH passes x_7 to CF_{345}. The record then descends down to *Node* 2. The comparison at this node has $x_7 = 1.1$ closer to CF_{34} than to CF_5. The record then descends down to *Node* 2.1. Here, $x_7 = 1.1$ closer to CF_4 than to CF_3. BIRCH tentatively passes x_7 to CF_4, and to *Leaf* 4. The radius of *Leaf* 4 becomes $R = 0.05$, which does not exceed the radius threshold value of $T = 0.15$. Therefore, BIRCH assigns x_7 to *Leaf* 4.

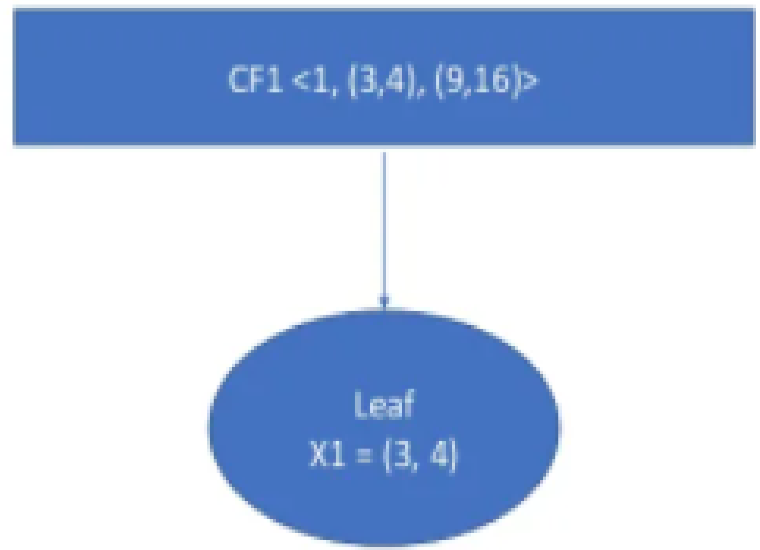


Seventh (and last) data value entered: Final form of CF tree.

Let Have Following Data
X1=(3,4), x2= (2,6), x3=(4,5), x4=(4,7), x5=(3,8), x6=(6,2), x7=(7,2), x8=(7,4), x9=(8,4), x10=(7,9)
Cluster the Above Data Using BIRCH Algorithm, , considering T<1.5, and Max Branch = 2

For each Data Point we need to evaluate Radius and Cluster Feature:

->Consider Data Pint (3,4):



As it is alone in the Feature map, Hence
1. Radius = 0
2. Cluster Feature CF1 <N, LS, SS>
   N = 1 as there is now one data point under consideration.
   LS = Sum of Data Point under consideration = (3,4)
   SS = Square Sum of Data Point Under Consideration
   $= (3^2, 4^2)=(9,16)$
3. Now construct the Leaf with Data Point X1 and Branch as CF1.

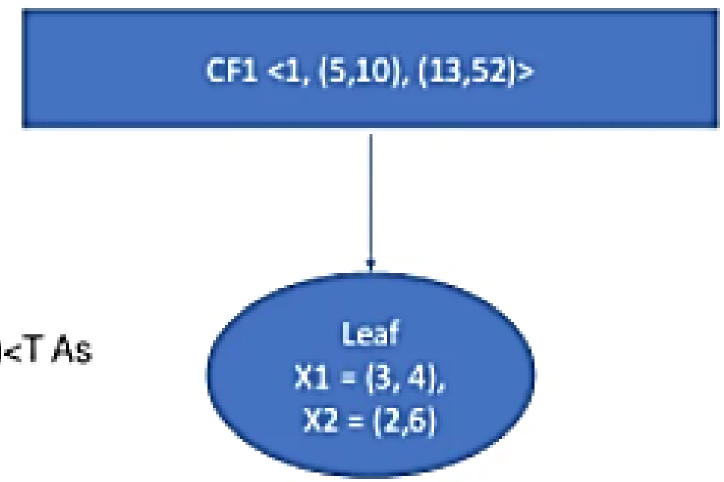
Let Have Following Data

X1=(3,4), x2= (2,6), x3=(4,5), x4=(4,7), x5=(3,8), x6=(6,2), x7=(7,2), x8=(7,4), x9=(8,4), x10=(7,9)

Cluster the Above Data Using BIRCH Algorithm, considering T<1.5, and Max Branch = 2

For each Data Point we need to evaluate Radius and Cluster Feature:

->Consider Data Pint x2 = (2,6):



1. Linear Sum LS = (3,4) + (2,6) = (5,10)
2. Square Sum SS = $(3^2+2^2 , 4^2+6^2)$ =(13, 52)

Now Evaluate Radius considering N=2

$$R = \sqrt{\frac{SS-LS^2/N}{N}} = \sqrt{\frac{(13,52)-(5,10)^2/2}{2}} = \sqrt{\frac{(13,52)-(25,100)/2}{2}} =$$

$$\sqrt{\frac{(13,52)-(12.5,50)}{2}} = \sqrt{(6.5,26)-(6.25,25)} = \sqrt{(0.25,1)}$$ =(0.5, 1)<T As (0.25,1) < (T, T), hence X2 will cluster with Leaf X1.

2. Cluster Feature CF1 <N, LS, SS> = <2,(5,10),(13,52)>

    N = 2 as there is now two data point under CF1.

    LS = (3,4) + (2,6) = (5,10)

    SS = $(3^2+2^2 , 4^2+6^2)$ =(13, 52)

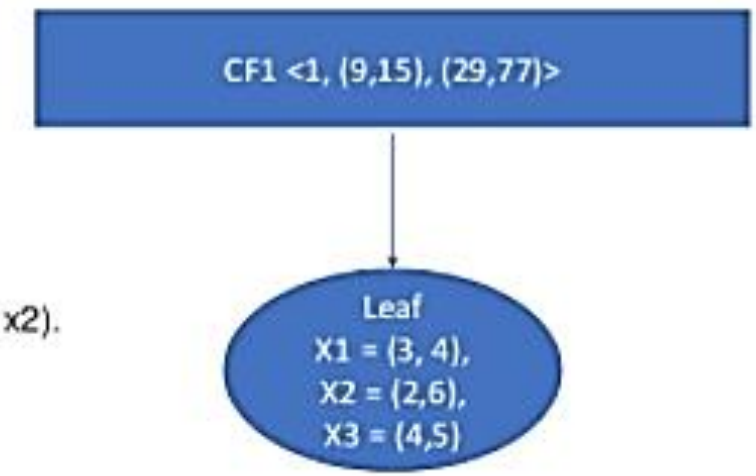
Let Have Following Data

X1=(3,4), x2= (2,6), x3=(4,5), x4=(4,7), x5=(3,8), x6=(6,2), x7=(7,2), x8=(7,4), x9=(8,4), x10=(7,9)

Cluster the Above Data Using BIRCH Algorithm, considering T<1.5, and Max Branch = 2

For each Data Point we need to evaluate Radius and Cluster Feature:

->Consider Data Pint x3 = (4,5) on CF1:



1. Linear Sum LS = (4,5) + (5,10) = (9,15)
2. Square Sum SS = (4^2+13 , 5^2 + 52) =(29, 77)

Now Evaluate Radius considering N=3

$R = \sqrt{\frac{SS - LS^2/N}{N}} = \sqrt{\frac{(29,77)-(9,15)^2/3}{3}}$ =(0.47, 0.4714)<T

As (0.47, 0.471) < (T, T), hence X3 will cluster with Leaf (X1, x2).

2. Cluster Feature CF1 <N, LS, SS> = <3,(9,15),(29,77)>
   N = 3 as there is now Three data point under CF1.
   LS = (4,5) + (5,10) = (9,15)
   SS = (4^2+13 , 5^2 + 52) =(29, 77)

Example

Let Have Following Data
X1=(3,4), x2= (2,6), x3=(4,5), x4=(4,7), x5=(3,8), x6=(6,2), x7=(7,2), x8=(7,4), x9=(8,4), x10=(7,9)
Cluster the Above Data Using BIRCH Algorithm, considering T<1.5, and Max Branch = 2

For each Data Point we need to evaluate Radius and Cluster Feature:

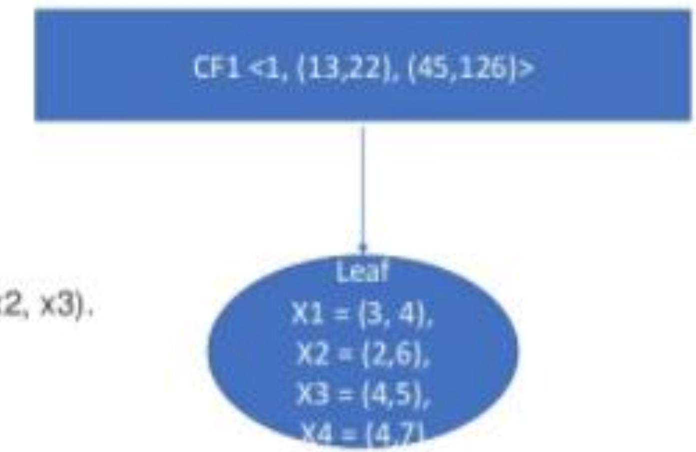
->Consider Data Pint x4 = (4,7) on CF1:

1. Linear Sum LS = (4,7) + (9,15) = (13,22)
2. Square Sum SS = (4^2+29 , 7^2 + 77) =(45, 126)

Now Evaluate Radius considering N=4

$R = \sqrt{\frac{SS-LS^2/N}{N}} = \sqrt{\frac{(45,126)-(13,22)^2/4}{4}}$ =(0.41, 0.55)

As (0.41, 0.55) < (T, T), hence X4 will cluster with Leaf (X1, x2, x3).
2. Cluster Feature CF1 <N, LS, SS> = <4,(13,22),(45,126)>
N = 4 as there is now four data point under CF1.
LS = (4,7) + (9,15) = (13,22)
SS = (4^2+29 , 7^2 + 77) =(45, 126)



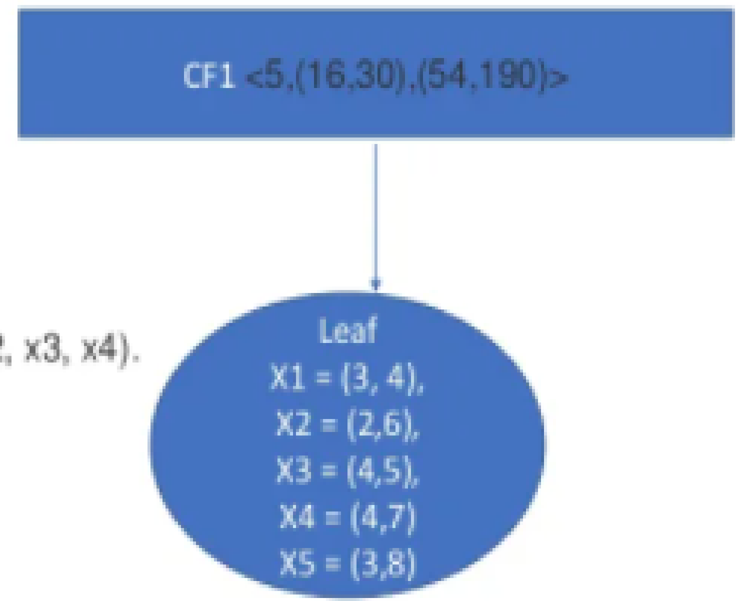
Let Have Following Data

X1=(3,4), x2= (2,6), x3=(4,5), x4=(4,7), x5=(3,8), x6=(6,2), x7=(7,2), x8=(7,4), x9=(8,4), x10=(7,9)

Cluster the Above Data Using BIRCH Algorithm, considering T<1.5, and Max Branch = 2

For each Data Point we need to evaluate Radius and Cluster Feature:

->Consider Data Pint x5 = (3,8) on CF1:



1. Linear Sum LS = (3,8) + (13,22) = (16,30)
2. Square Sum SS = (3^2+45 , 8^2 + 126) =(54, 190)

Now Evaluate Radius considering N=5

$R = \sqrt{\frac{SS-LS^2/N}{N}} = \sqrt{\frac{(54,190)-(16,30)^2/5}{5}}$ =(0.33, 0.63)

As (0.33, 0.63) < (T, T), hence X5 will cluster with Leaf (X1, x2, x3, x4).

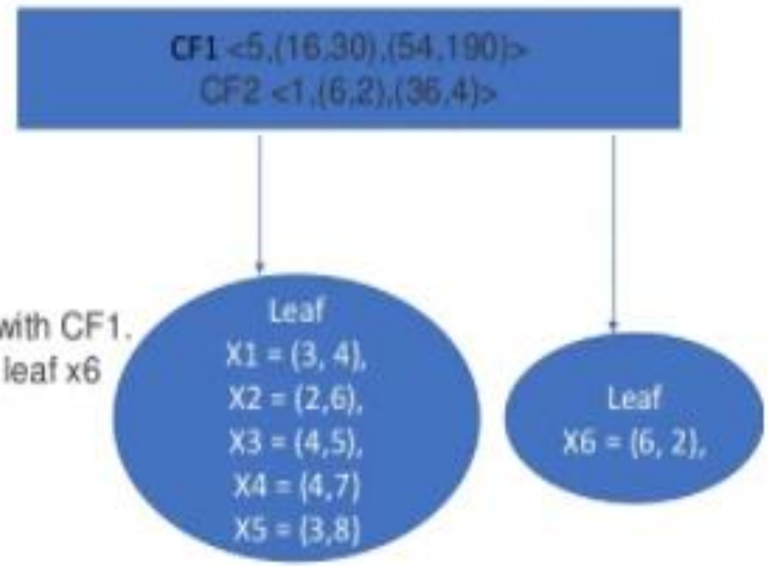
2. Cluster Feature CF1 <N, LS, SS> = <5,(16,30),(54,190)>

N = 5 as there is now four data point under CF1.

Let Have Following Data
X1=(3,4), x2= (2,6), x3=(4,5), x4=(4,7), x5=(3,8), x6=(6,2), x7=(7,2), x8=(7,4), x9=(8,4), x10=(7,9)
Cluster the Above Data Using BIRCH Algorithm, considering T<1.5, and Max Branch = 2

For each Data Point we need to evaluate Radius and Cluster Feature:

->Consider Data Pint x6 = (6,2) on CF1:



1. Linear Sum LS = (6,2) + (16,30) = (22,32)
2. Square Sum SS = (6^2+54 , 2^2 + 190) =(90, 194)

Now Evaluate Radius considering N=6

$R = \sqrt{\frac{SS-LS^2/N}{N}} = \sqrt{\frac{(90,194)-(22,32)^2/6}{6}}$ =(1.24, 1.97)

As (1.24, 1.97) < (T, T), False. hence X6 will Not form cluster with CF1. CF1 will remain as it was in previous step. And New CF2 with leaf x6 will be created.

2. Cluster Feature CF2 <N, LS, SS> = <1,(6,2),(36,4)>
   N = 1 as there is now one data point under CF2.
   LS = (6,2)
   SS = (6^2, 2^2)= (36,4)

Let Have Following Data

X1=(3,4), x2= (2,6), x3=(4,5), x4=(4,7), x5=(3,8), x6=(6,2), x7=(7,2), x8=(7,4), x9=(8,4), x10=(7,9)

Cluster the Above Data Using BIRCH Algorithm, considering T<1.5, and Max Branch = 2

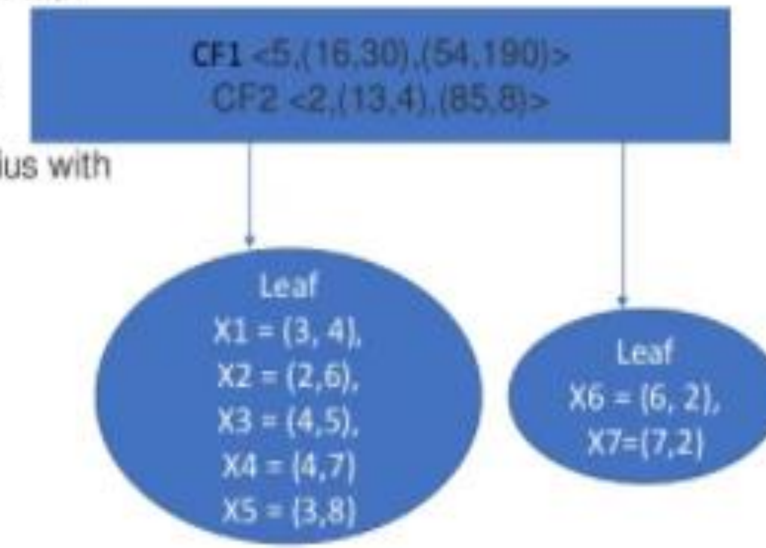
For each Data Point we need to evaluate Radius and Cluster Feature:

->Consider Data Pint x7 = (7,2). As There are Two Branch CF1 and CF2 hence we need to find with which branch X7 is nearer, then with that leaf radius will be evaluated.

With CF1 = LS/N= (16,30)/5=(8,6) As there are N=5 Data Point

With CF2 = LS/N= (6,2)/1=(6,2) As there is N=1 Data Point

Now x7 is closer to (6,2) then (8,6). Hence X7 will calculate radius with CF2.



1. Linear Sum LS = (7,2) + (6,2) = (13,4)
2. Square Sum SS = (7^2+36 , 2^2 + 4) =(85, 8)

Now Evaluate Radius considering N=2

$$R = \sqrt{\frac{SS - LS^2/N}{N}} = \sqrt{\frac{(85,8)-(13,4)^2/2}{2}} = (0.5,\ 0)$$

As (0.5, 0) < (T, T), True. hence X7 will form cluster with CF2

2. Cluster Feature CF2 <N, LS, SS> = <2,(13,4),(85,8)>

   N = 2 as there is now two data point under CF2.

   LS = (7,2) + (6,2) = (13,4)

   SS = (7^2+36 , 2^2 + 4) =(85, 8)

Let Have Following Data

X1=(3,4), x2= (2,6), x3=(4,5), x4=(4,7), x5=(3,8), x6=(6,2), x7=(7,2), x8=(7,4), x9=(8,4), x10=(7,9)

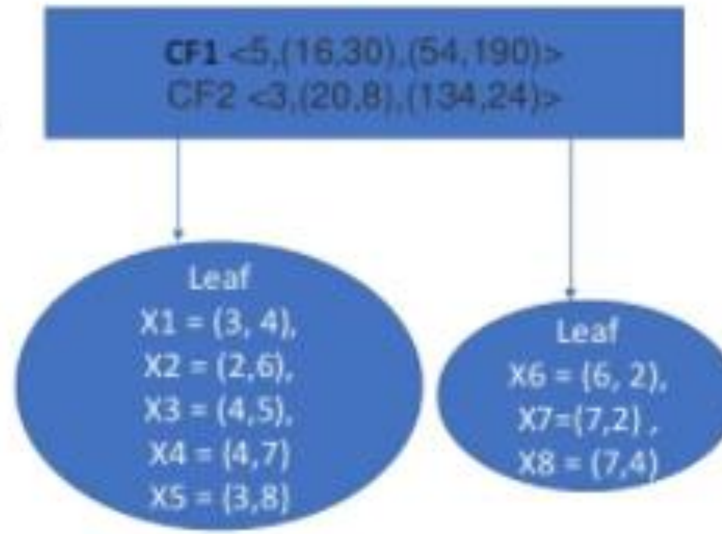
Cluster the Above Data Using BIRCH Algorithm, considering T<1.5, and Max Branch = 2

->Consider Data Pint x8 = (7,4). As There are Two Branch CF1 and CF2 hence we need to find with which branch X8 is nearer, then with that leaf, radius will be evaluated.

With CF1 = LS/N= (16,30)/5=(8,6) As there are N=5 Data Point

With CF2 = LS/N= (13,4)/2=(6.5,2) As there is N=2 Data Point

Now x8 is closer to (6.5,2) then (8,6). Hence X8 will calculate radius with CF2.



1. Linear Sum LS = (7,4) + (13,4) = (20,8)
2. Square Sum SS = $(7^2+85\ ,\ 4^2+8)$ =(134, 24)

Now Evaluate Radius considering N=3

$$R = \sqrt{\frac{SS - LS^2/N}{N}} = \sqrt{\frac{(134,24)-(20,8)^2/3}{3}} = (0.47,\ 0.94)$$

As (0.47, 94) < (T, T), True. hence X8 will form cluster with CF2

2. Cluster Feature CF2 <N, LS, SS> = <3,(20,8),(134,24)>

   N = 3 as there is now two data point under CF2.

   LS (7,4) + (13,4) = (20,8)

   SS = (134,24)

Let Have Following Data

X1=(3,4), x2= (2,6), x3=(4,5), x4=(4,7), x5=(3,8), x6=(6,2), x7=(7,2), x8=(7,4), x9=(8,4) , x10=(7,9)

Cluster the Above Data Using BIRCH Algorithm, considering T<1.5, and Max Branch = 2

->Consider Data Pint x9 = (8,4). As There are Two Branch CF1 and CF2 hence we need to find with which branch X9 is nearer, then with that leaf, radius will be evaluated.

With CF1 = LS/N= (16,30)/5=(8,6) As there are N=5 Data Point

With CF2 = LS/N= (20,8)/3=(6.6,2.6) As there is N=3 Data Point

Now x9 is closer to (6.6,2.6) then (8,6). Hence X8 will calculate radius with CF2.

1. Linear Sum LS = (8,4) + (20,8) = (28,12)
2. Square Sum SS = (8^2+134 , 4^2 + 24) =(198, 40)

Now Evaluate Radius considering N=4

$$R = \sqrt{\frac{SS-LS^2/N}{N}} = \sqrt{\frac{(198,40)-(28,12)^2/4}{4}} =(0.70, 1)$$

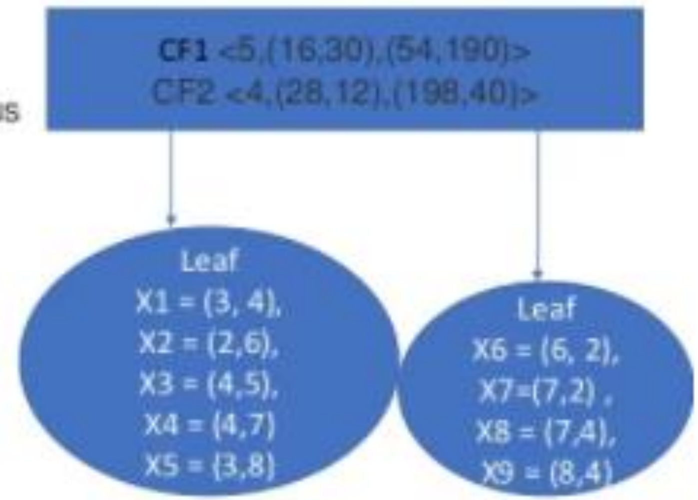
As (0.7, 1) < (T, T), True. hence X9 will form cluster with CF2

2. Cluster Feature CF2 <N, LS, SS> = <4,(28,12),(198,40)>

   N = 4 as there is now four data point under CF2.

   LS = (28,12)

   SS = (198,40)



Example

Let Have Following Data

X1=(3,4), x2= (2,6), x3=(4,5), x4=(4,7), x5=(3,8), x6=(6,2), x7=(7,2), x8=(7,4), x9=(8,4) , x10=(7,9)

Cluster the Above Data Using BIRCH Algorithm, considering T<1.5, and Max Branch = 2

->Consider Data Pint x10 = (7,9). As There are Two Branch CF1 and CF2 hence we need to find with which branch X9 is nearer, then with that leaf, radius will be evaluated.

With CF1 = LS/N= (16,30)/5=(8,6) As there are N=5 Data Point

With CF2 = LS/N= (28,12)/4=(7,3) As there is N=4 Data Point

Now x10 is closer to (8,6) then (7,3). Hence X10 will calculate radius with CF1.

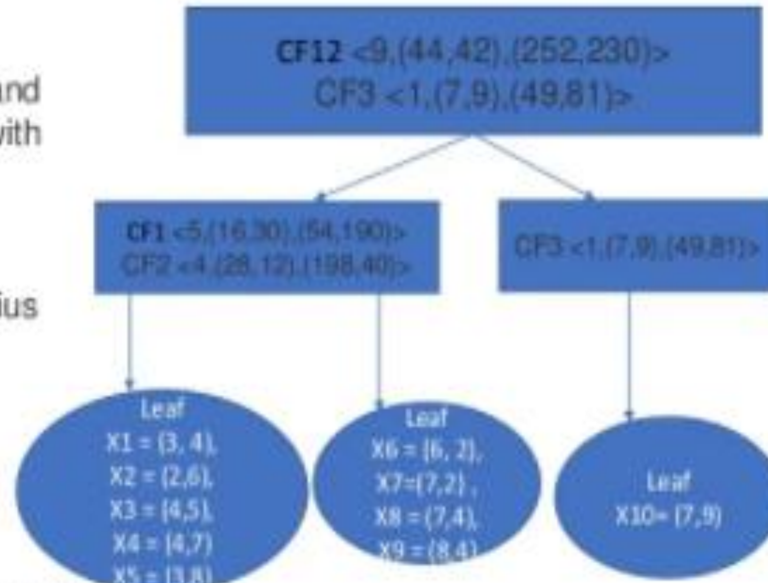
1. Linear Sum LS = (7,9) + (16,30) = (23,39)
2. Square Sum SS = (7^2+54 , 9^2 + 190) =(103, 271)

Now Evaluate Radius considering N=6

$R = \sqrt{\frac{SS-LS^2/N}{N}} = \sqrt{\frac{(103,271)-(23,39)^2/6}{6}}$ =(1.57, 1.70)

As (1.57, 1.70) < (T, T), False. hence X10 will become new leaf and Create new cluster feature CF3. But in a Branch only two CF is allowed hence Branch will Split.

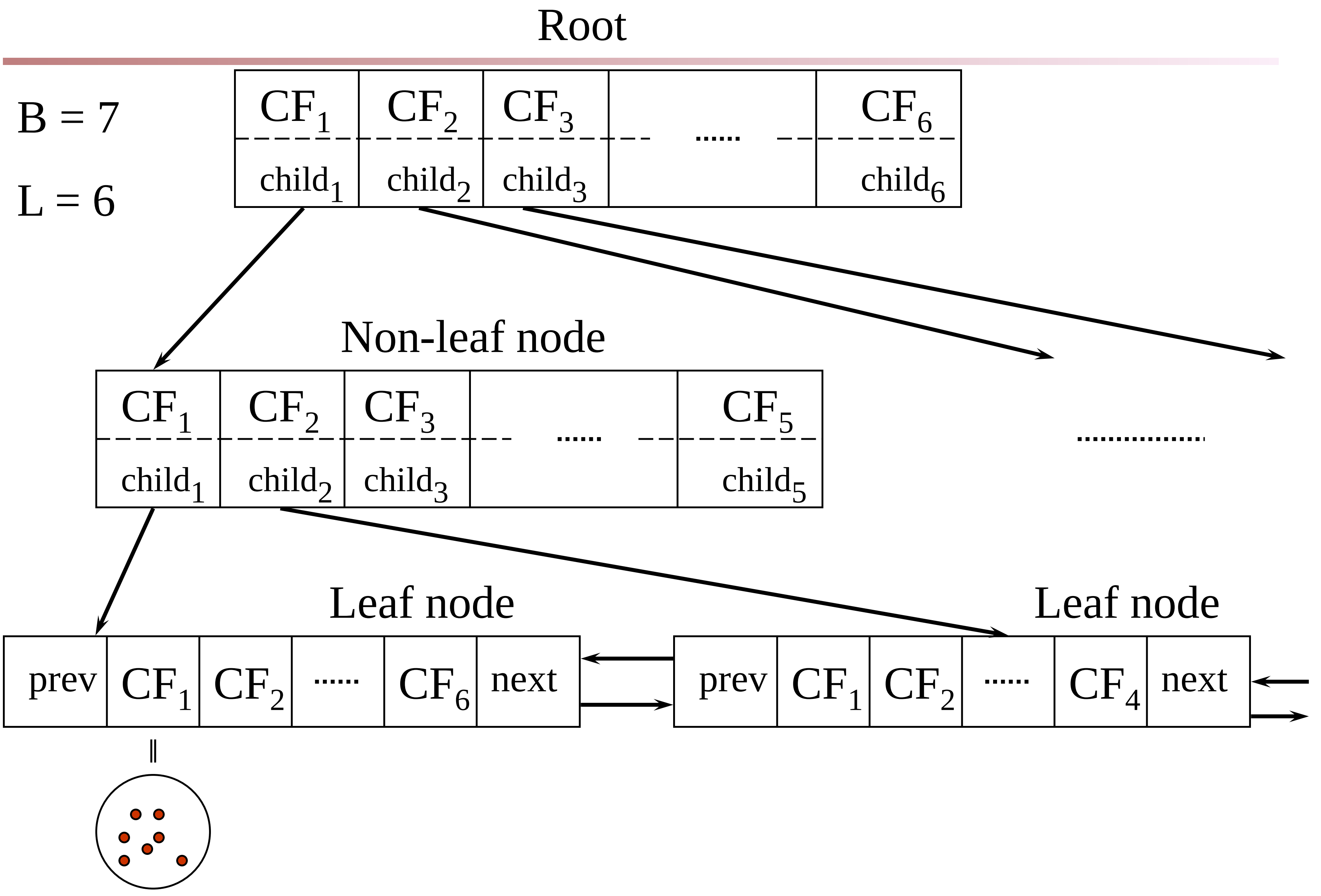
2. Cluster Feature CF3 <N, LS, SS> = <1,(7,9),(49,81)>



CF-Tree in BIRCH

- Clustering feature:
  - summary of the statistics for a given subcluster
- A CF tree is a height-balanced tree that stores the clustering features for a hierarchical clustering
  - A nonleaf node in a tree has descendants or “children”
  - The nonleaf nodes store sums of the CFs of their children
- A CF tree has two parameters
  - Branching factor: specify the maximum number of children.
  - threshold: max diameter of sub-clusters stored at the leaf nodes

The CF Tree Structure



DBSCAN – Density-Based Spatial Clustering of Applications with Noise

DBSCAN

Density-based Clustering locates regions of high density that are separated from one another by regions of low density.

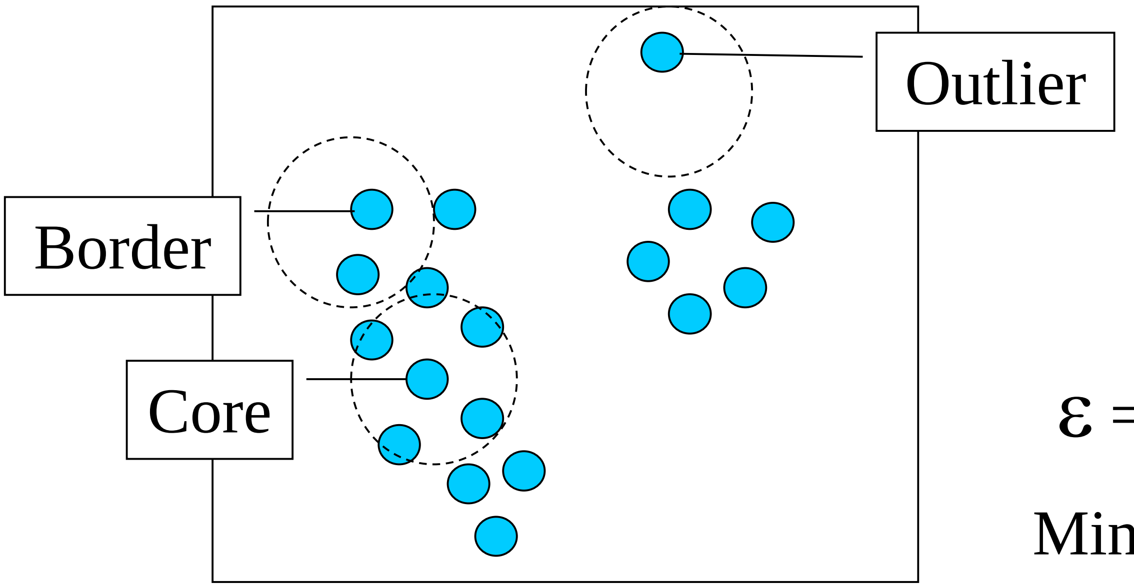
- Density = number of points within a specified radius (Eps)

- DBSCAN is a density-based algorithm.
  - 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

DBSCAN

- A noise point is any point that is not a core point or a border point.
- Any two core points are close enough– within a distance *Eps* of one another – are put in the same cluster
- Any border point that is close enough to a core point is put in the same cluster as the core point
- Noise points are discarded

Border & Core

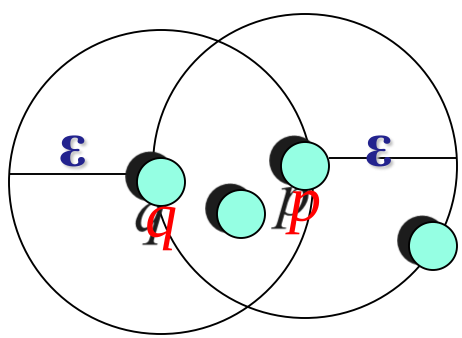


ε = 1unit

MinPts = 5

Concepts: ε-Neighborhood

- **ε-Neighborhood** - Objects within a radius of ε from an object. (epsilon-neighborhood)
- **Core objects** - ε-Neighborhood of an object contains at least **MinPts** of objects



ε-Neighborhood of *p*

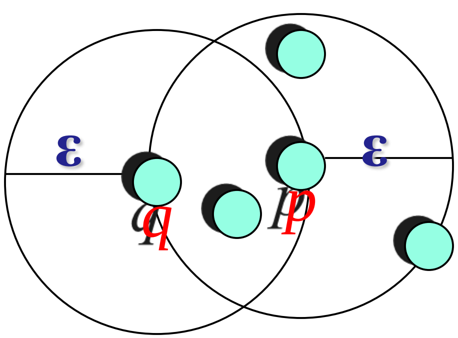
ε-Neighborhood of *q*

p is a core object (MinPts = 4)

q is not a core object

Concepts: Reachability

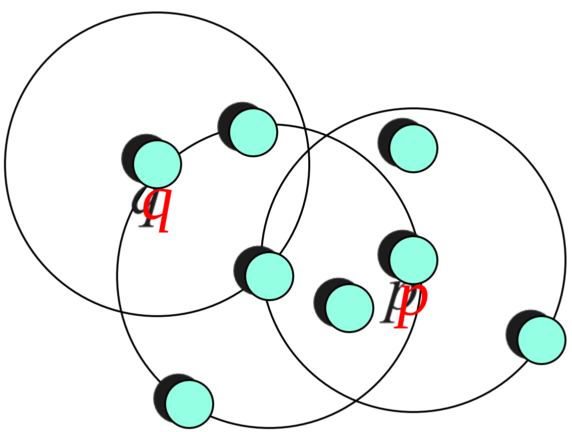
- **Directly density-reachable**
  - An object q is directly density-reachable from object p if q is within the ε-Neighborhood of p and p is a core object.



- q is directly density-reachable from p
- p is not directly density-reachable from q?

Concepts: Reachability

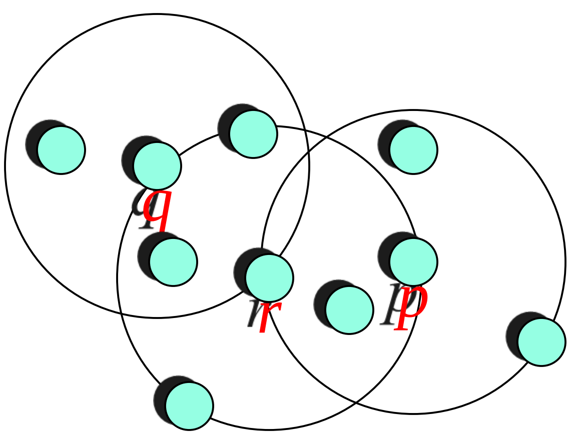
- **Density-reachable:**
  - An object *p* is density-reachable from *q* w.r.t ε and *MinPts* if there is a chain of objects $p_1,\ldots,p_n$, with $p_1=q$, $p_n=p$ such that p_{i+1} is directly density-reachable from p_i w.r.t ε and *MinPts* for all $1 <= i <= n$



- *q* is density-reachable from *p*
- *p* is not density- reachable from *q*?

Concepts: Connectivity

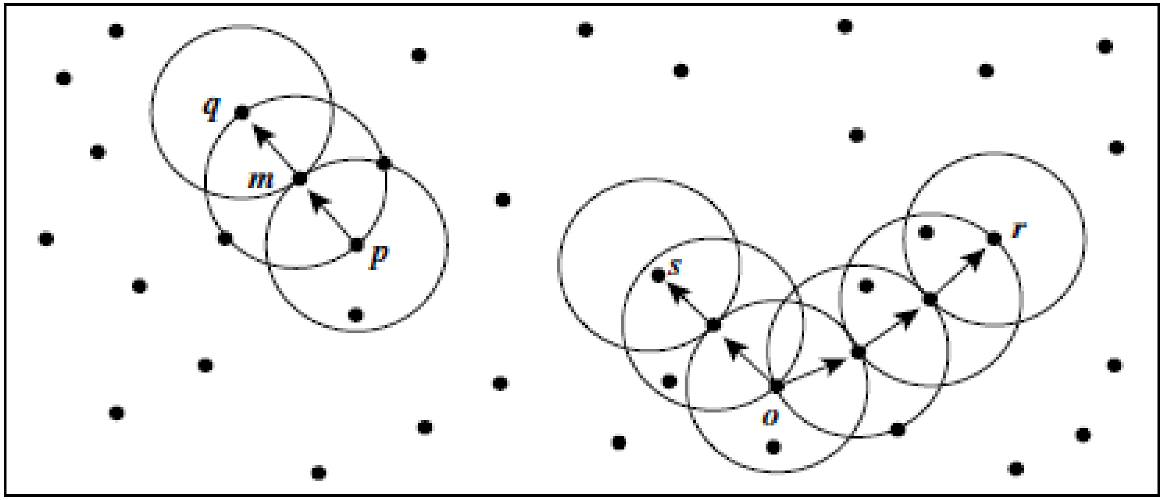
- **Density-connectivity**
  - Object *p* is density-connected to object *q* w.r.t ε and *MinPts* if there is an object *o* such that both *p* and *q* are density-reachable from *o* w.r.t ε and *MinPts*



- *P* and *q* are density-connected to each other by *r*
- Density-connectivity is symmetric

Density-Reachable and Density-Connected

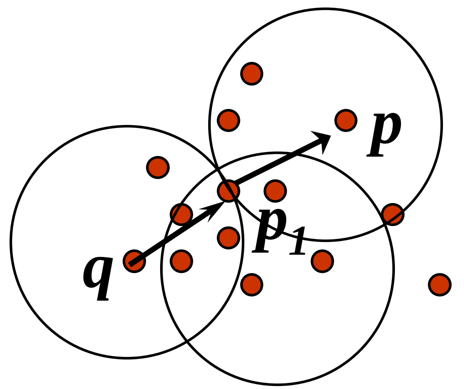
- Of the labeled points, m, p, o, and r are core objects because each is in an ε-neighborhood containing at least three points.
- q is directly density-reachable from m. m is directly density-reachable from p and vice versa.
- q is (indirectly) density-reachable from p because q is directly density-reachable from m and m is directly density-reachable from p. However, p is not density-reachable from q because q is not a core object. Similarly, r and s are density-reachable from o, and o is density-reachable from r.
- o, r, and s are all density-connected. ■



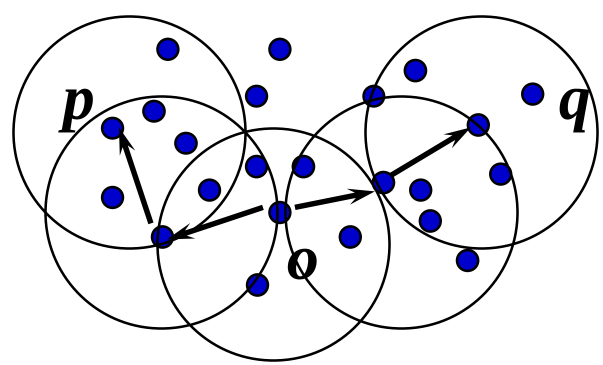
Concepts: cluster & noise

- **Cluster**: a cluster ***C*** in a set of objects ***D*** w.r.t ε and *MinPts* is a non empty subset of ***D*** satisfying
  - Maximality: For all *p*, *q* if $p \in$ ***C*** and if *q* is density-reachable from *p* w.r.t ε and *MinPts*, then also $q \in$ ***C.***
  - Connectivity: for all $p, q \in$ ***C***, *p* is density-connected to *q* w.r.t ε and *MinPts* in ***D.***
  - ***Note:*** cluster contains *core objects* as well as *border objects*
- **Noise:** objects which are not directly density-reachable from at least one core object.

(Indirectly) Density-reachable:



Density-connected

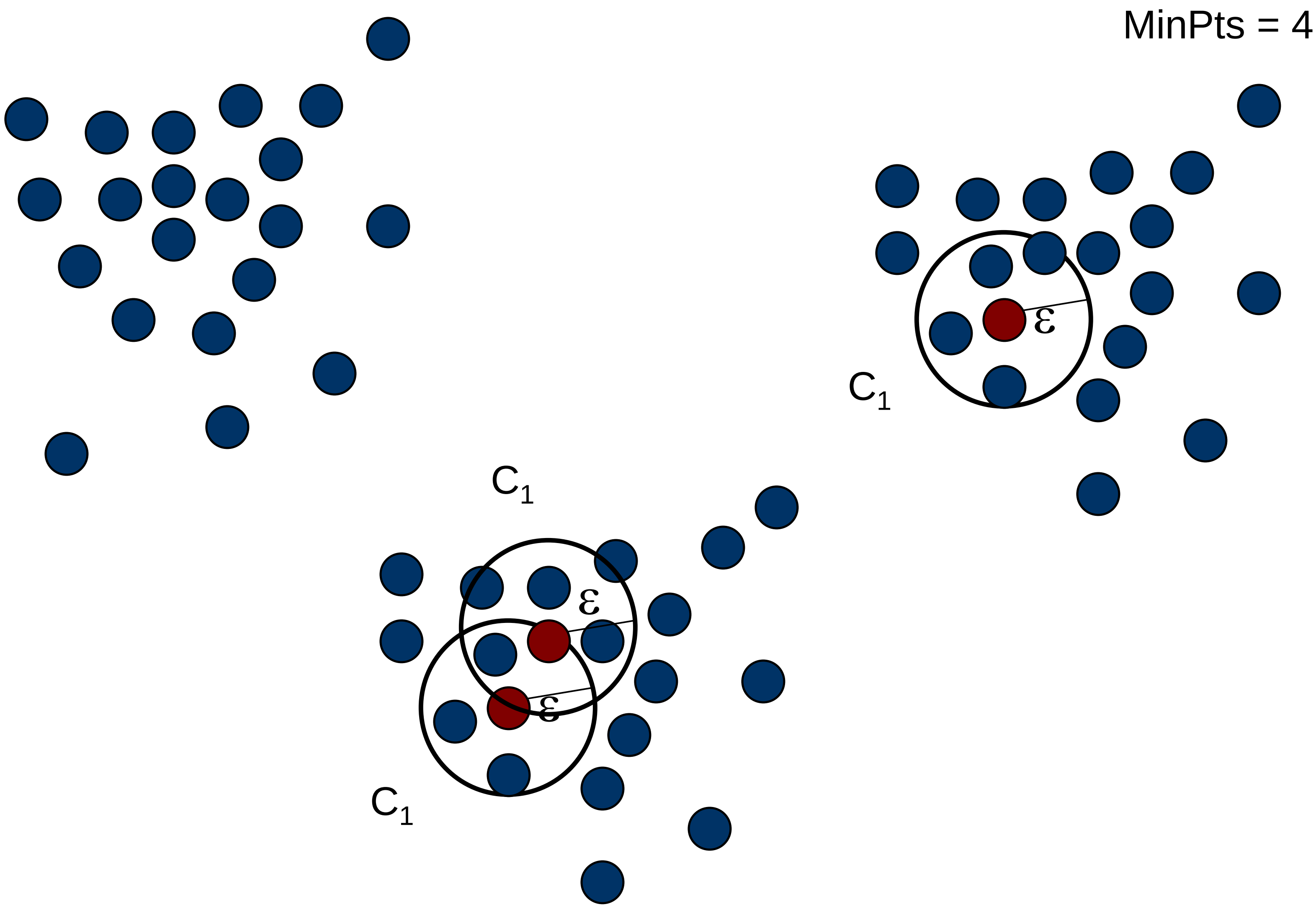


DBSCAN: The Algorithm

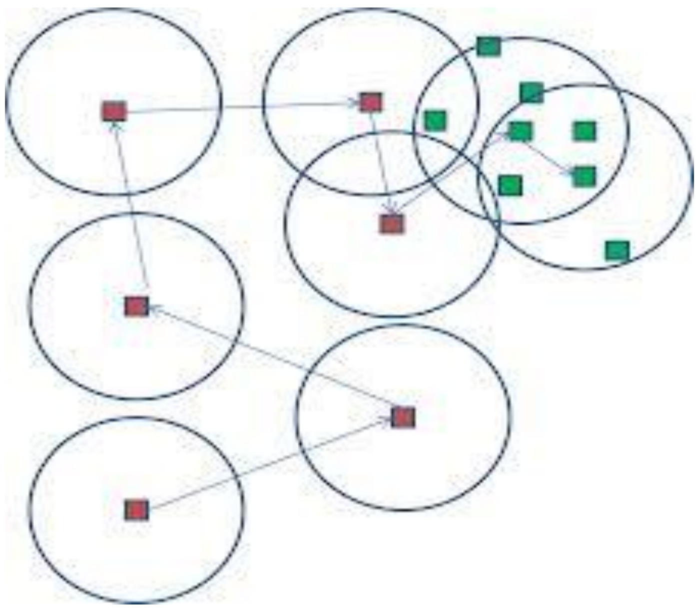
- Select a point ***p***
- Retrieve all points density-reachable from ***p*** wrt ε and ***MinPts***.
- If ***p*** is a core point, a cluster is formed.
- If ***p*** is a border point, no points are density-reachable from ***p*** and DBSCAN visits the next point of the database.
- Continue the process until all of the points have been processed.

Result is independent of the order of processing the points

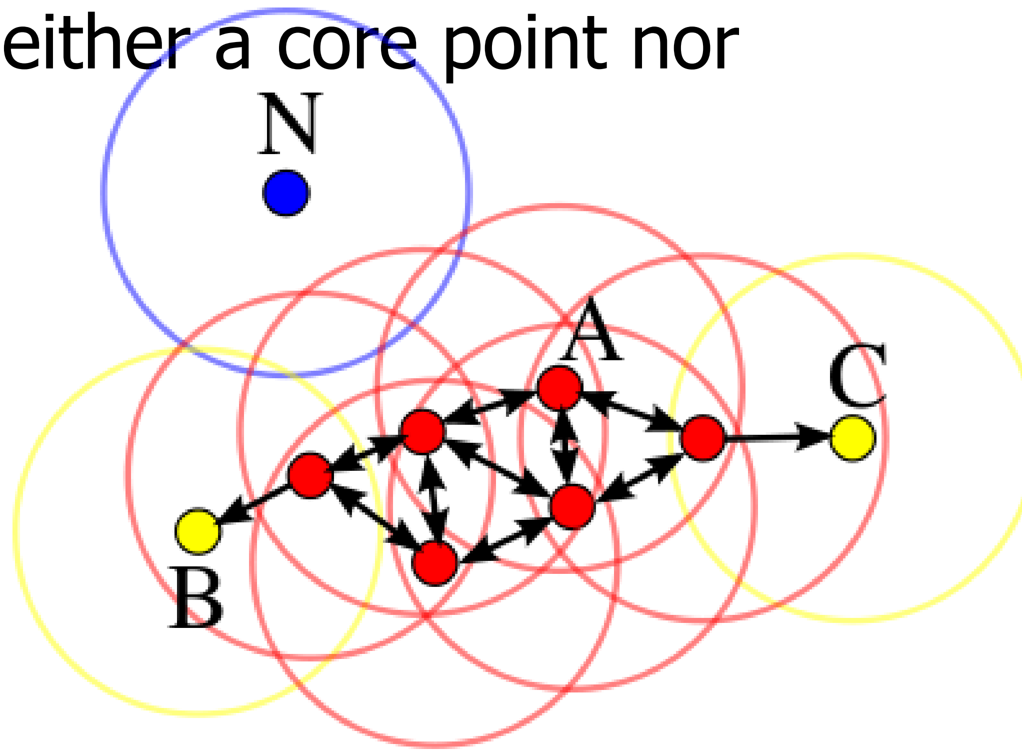
An Example



- Test in epsilon (Epsilon is the radius of the circles) if the number of point is 4. If yes start a cluster (green) otherwise mark as noise (red).
- The arrows show all the points you visited



- minPts = 3. A and the other red points are core points, because at least three points surround it in an ε radius. Because they are all reachable from one another, they form a single cluster. Points B and C are not core points, but are reachable from A (via other core points) and thus belong to the cluster as well. Point N is a noise point that is neither a core point nor density-reachable.



- **DBSCAN(D, eps, MinPts)**
  - C = 0
  - for each unvisited point P in dataset D
  - mark P as visited
  - N = getNeighbors (P, eps)
  - if sizeof(N) < MinPts mark P as NOISE
  - Else
  - C = next cluster
  - expandCluster(P, N, C, eps, MinPts)
- **expandCluster(P, N, C, eps, MinPts)**
  - add P to cluster C
  - for each point P' in N
  - if P' is not visited mark P' as visited
  - N' = getNeighbors(P', eps)
  - if sizeof(N') >= MinPts
  - N = N joined with N`
  - if P' is not yet member of any cluster add P' to cluster C

DBSCAN

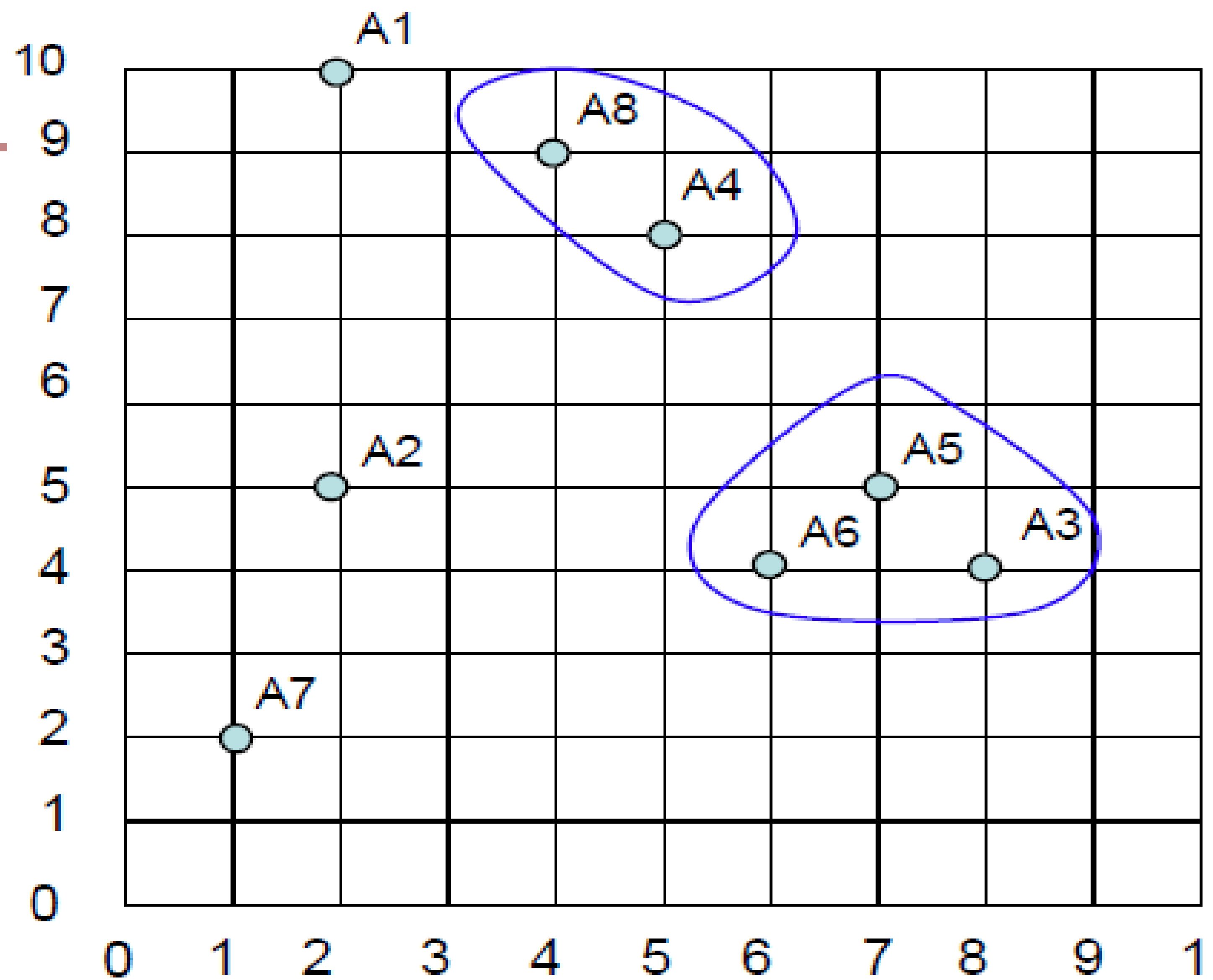
- If Epsilon is 2 and minpoint is 2, what are the clusters that DBScan would discover with the following 8 data objects.
- examples: A1=(2,10), A2=(2,5), A3=(8,4), A4=(5,8), A5=(7,5), A6=(6,4), A7=(1,2), A8=(4,9).

- The distance matrix based on the Euclidean distance is given below:

	A1	A2	A3	A4	A5	A6	A7	A8
A1	0	$\sqrt{25}$	$\sqrt{36}$	$\sqrt{13}$	$\sqrt{50}$	$\sqrt{52}$	$\sqrt{65}$	$\sqrt{5}$
A2		0	$\sqrt{37}$	$\sqrt{18}$	$\sqrt{25}$	$\sqrt{17}$	$\sqrt{10}$	$\sqrt{20}$
A3			0	$\sqrt{25}$	$\sqrt{2}$	$\sqrt{2}$	$\sqrt{53}$	$\sqrt{41}$
A4				0	$\sqrt{13}$	$\sqrt{17}$	$\sqrt{52}$	$\sqrt{2}$
A5					0	$\sqrt{2}$	$\sqrt{45}$	$\sqrt{25}$
A6						0	$\sqrt{29}$	$\sqrt{29}$
A7							0	$\sqrt{58}$
A8								0

- ***Solution:***

- What is the Epsilon neighborhood of each point?
- (A1)={};
- (A2)={};
- (A3)={A5, A6};
- (A4)={A8};
- (A5)={A3, A6};
- (A6)={A3, A5};
- (A7)={};
- (A8)={A4}
- So A1, A2, and A7 are outliers, while we have two clusters C1={A4, A8} and C2={A3, A5, A6}



Epsilon = 2

DBSCAN

- If Epsilon is 2 and minpoint is 2, what are the clusters that DBScan would discover with the following 8 data objects.
- examples: A1=(2,10), A2=(2,5), A3=(8,4), A4=(5,8), A5=(7,5), A6=(6,4), A7=(1,2), A8=(4,9).
- What if Epsilon is increased to sqrt(10)?

OPTICS

- Self Study