MODULE-5
CLUSTERING ANALYSIS

5.1 Introduction

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.

The greater the similarity within a group and the greater the difference between groups, the better or more distinct the clustering.

Cluster analysis divides data into groups (clusters) that are meaningful, useful, or both.

In the context of understanding data, clusters are potential classes and cluster analysis is the study of techniques for automatically finding classes.

Clustering: Applications

Biology: biologists have applied clustering to analyze the large amounts of genetic information that are now available.

For example, clustering has been used to find groups of genes that have similar functions. **Information Retrieval**:. The World Wide Web consists of billions of Web pages, and the results of a query to a search engine can return thousands of pages. Clustering can be used to group these search results into a small number of clusters, each of which captures a particular aspect of the query.

Climate: Understanding the Earth's climate requires finding patterns in the atmosphere and ocean. To that end, cluster analysis has been applied to find patterns in the atmospheric pressure of polar regions and are as of the ocean that have a significant impact on land climate.

Psychology and Medicine: An illness or condition frequently has a number of variations, and cluster analysis can be used to identify these different subcategories.

Business: Businesses collect large amounts of information on current and potential customers. Clustering can be used to segment customers into a small number of groups for additional analysis and marketing activities.

Types of Clustering's

A clustering is a set of clusters

Important distinction between hierarchical and partitional sets of clusters Partitional Clustering

– A division data objects into non-overlapping subsets (clusters) such that each data object is in exactly one subset

– Hierarchical clustering

A set of nested clusters organized as a hierarchical tree

Types of Clusters

- ➢ Well-separated clusters
- ➢ Center-based clusters
- ➢ Contiguous clusters
- ➢ Density-based clusters
- ➢ Property or Conceptual

Well-Separated Clusters:

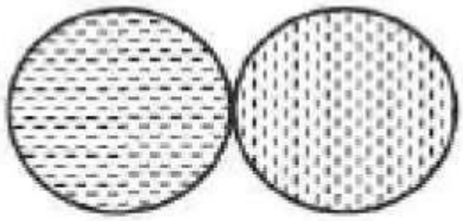
A cluster is a set of points such that any point in a cluster is closer (or more similar) to every other point in the cluster than to any point not in the cluster

(a) Well-separated clusters. Each point is closer to all of the points in its cluster than to any point in another cluster.

Center-based (proto type based)

– A cluster is a set of objects such that an object in a cluster is closer (more similar) to the “center” of a cluster, than to the center of any other cluster

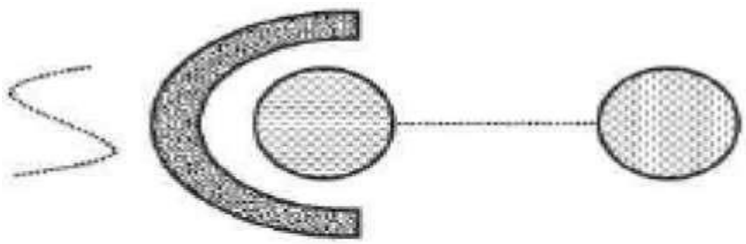
– The center of a cluster is often a centroid, the average of all the points in the cluster, or a medoid, the most “representative” point of a cluster



(b) Center-based clusters. Each point is closer to the center of its cluster than to the center of any other cluster.

Contiguous Cluster (Graph based)

– A cluster is a set of points such that a point in a cluster is closer (or more similar) to one or more other points in the cluster than to any point not in the cluster.

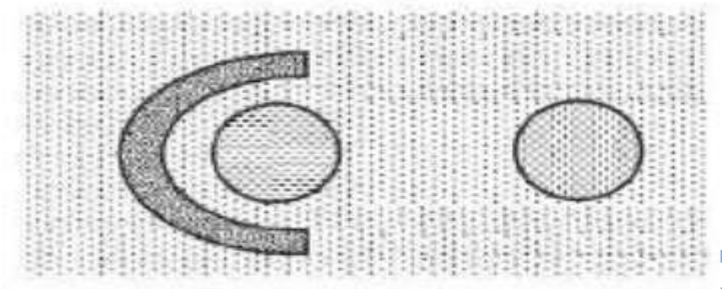


(c) Contiguity-based clusters. Each point is closer to at least one point in its cluster than to any point in another cluster.

Density-based

– A cluster is a dense region of points, which is separated by low-density regions, from other regions of high density.

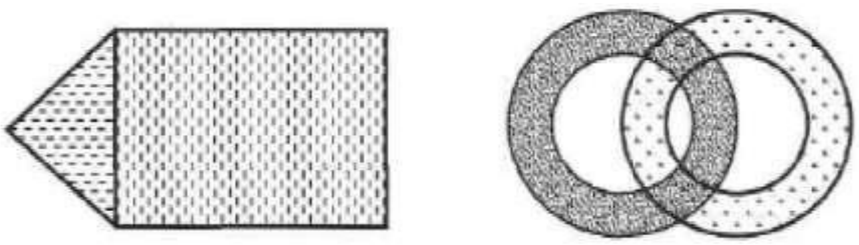
– Used when the clusters are irregular or intertwined, and when noise and outliers are present.



(d) Density-based clusters. Clusters are regions of high density separated by regions of low density.

Shared Property or Conceptual Clusters

– Finds clusters that share some common property or represent a particular concept.



(e) Conceptual clusters. Points in a cluster share some general property that derives from the entire set of points. (Points in the intersection of the circles belong to both.)

5.2 K-means Clustering

- Partitional clustering approach

- Each cluster is associated with a centroid (center point)
- Each point is assigned to the cluster with the closest centroid
- Number of clusters, K, must be specified

```
1: Select $K$ points as the initial centroids.
2: repeat
3:     Form $K$ clusters by assigning all points to the closest centroid.
4:     Recompute the centroid of each cluster.
5: until The centroids don't change
```

- Initial centroids are often chosen randomly.
- Clusters produced vary from one run to another.
- The centroid is (typically) the mean of the points in the cluster.
- „Closeness‟ is measured by Euclidean distance, cosine similarity, correlation, etc.
- K-means will converge for common similarity measures mentioned above.
- Most of the convergence happens in the first little iteration.
- Often the stopping condition is changed to „Until relatively few points change clusters‟
- Complexity is O(n * K * I * d)
- n = number of points, K = number of clusters, I = number of iterations, d = number of attributes

Evaluating K-means Clusters

Most common measure is Sum of Squared Error (SSE).
For each point, the error is the distance to the nearest cluster. To get SSE, we square these errors and sum them.

$$SSE = \sum_{i=1}^{K} \sum_{x \in C_i} dist^2(m_i, x)$$

x is a data point in cluster Ci and mi is the representative point for cluster Ci can show that mi corresponds to the center (mean) of the cluster.
Given two clusters, we can choose the one with the smallest error. One easy way to reduce SSE is to increase K, the number of clusters.

A good clustering with smaller K can have a lower SSE than a poor clustering with higher K. Problems with Selecting Initial Points

If there are K „real" clusters then the chance of selecting one centroid from each cluster is small.

Chance is relatively small when K is large If clusters are the same size, n, then

$$P = \frac{\text{number of ways to select one centroid from each cluster}}{\text{number of ways to select } K \text{ centroids}} = \frac{K!n^K}{(Kn)^K} = \frac{K!}{K^K}$$

K-means: Additional Issues Handling Empty Clusters

Basic K-means algorithm can yield empty clusters Several strategies to address this..

- ➢ Choose the point that contributes most to SSE
- ➢ Choose a point from the cluster with the highest SSE
- ➢ If there are several empty clusters, the above can be repeated several times

Updating Centers Incrementally

In the basic K-means algorithm, centroids are updated after all points are assigned to a centroid An alternative is to update the centroids after each assignment (incremental approach)

- – Each assignment updates zero or two centroids
- – More expensive
- – Introduces an order dependency
- – Never get an empty cluster
- – Can use "weights" to change the impact

Pre-processing and Post-processing

Pre-processing

- ➢ Normalize the data
- ➢ Eliminate outliers Post-processing
- ➢ Eliminate small clusters that may represent outliers

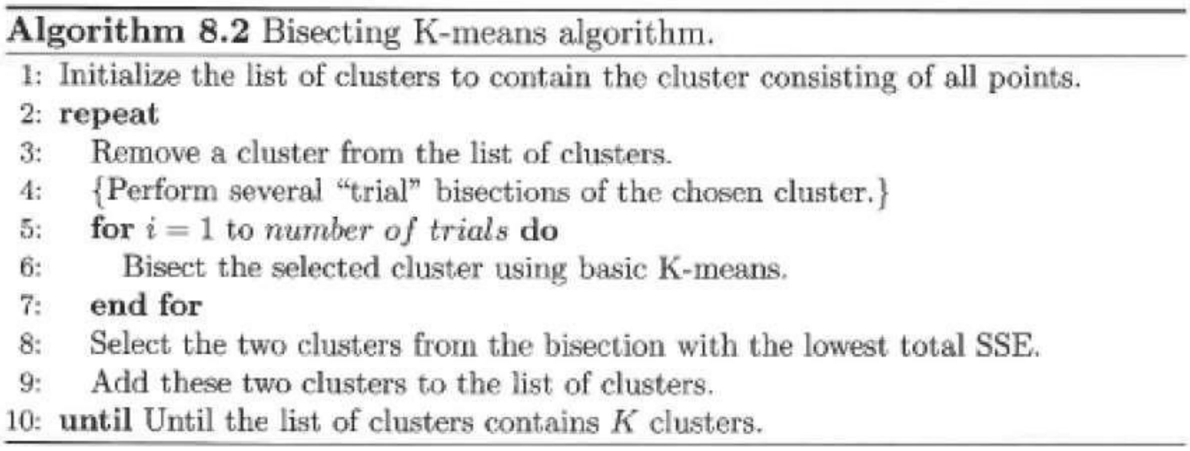
- Split „loose‟ clusters, i.e., clusters with relatively high SSE
- Merge clusters that are „close‟ and that have relatively low SSE
- Can use these steps during the clustering process

Strengths and Weaknesses of K-means (Limitations)

- K-means is simple and can be used for a wide variety of data types.
- It is also quite efficient, even though multiple runs are often performed.
- K-means is not suitable for all types of data.
- K-means has problems when clusters are of differing
  - Sizes
  - Densities
  - Non-globular shapes
- K-means has problems when the data contains outliers.

BisectingK-means

The bisecting K-means algorithm is a straightforward extension of the basic K-means algorithm that is based on a simple idea: to obtain K clusters, split the set of all points into two clusters, select one of these clusters to split, and so on, until K clusters have been produced.



Algorithm 8.2 Bisecting K-means algorithm.

```
1: Initialize the list of clusters to contain the cluster consisting of all points.
2: repeat
3:    Remove a cluster from the list of clusters.
4:    {Perform several "trial" bisections of the chosen cluster.}
5:    for $i = 1$ to $number\ of\ trials$ do
6:       Bisect the selected cluster using basic K-means.
7:    end for
8:    Select the two clusters from the bisection with the lowest total SSE.
9:    Add these two clusters to the list of clusters.
10: until Until the list of clusters contains $K$ clusters.
```

There are a number of different ways to choose which cluster to split. We can choose the largest cluster at each step, choose the one with the largest SSE, or use a criterion based on both size and SSE. Different choices result in different clusters.

Exercise 1. K-means clustering

Use the k-means algorithm and Euclidean distance to cluster the following 8 examples into 3 clusters:
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

Suppose that the initial seeds (centers of each cluster) are A1, A4 and A7. Run the k-means algorithm for 1 epoch only. At the end of this epoch show:
a) The new clusters (i.e. the examples belonging to each cluster)
b) The centers of the new clusters
c) Draw a 10 by 10 space with all the 8 points and show the clusters after the first epoch and the new centroids.

Solution:
a)
d(a,b) denotes the Eucledian distance between a and b. It is obtained directly from the distance matrix or calculated as follows: $d(a,b)=\text{sqrt}((x_b-x_a)^2+(y_b-y_a)^2)$
seed1=A1=(2,10), seed2=A4=(5,8), seed3=A7=(1,2)

epoch1 – start:

A1:
d(A1, seed1)=0 as A1 is seed1
d(A1, seed2)= $\sqrt{13}$ >0
d(A1, seed3)= $\sqrt{65}$ >0
→A1 $\in$ cluster1

A2:
d(A2,seed1)= $\sqrt{25}$ = 5
d(A2, seed2)= $\sqrt{18}$ = 4.24
d(A2, seed3)= $\sqrt{10}$ = 3.16 ← smaller
→ A2 $\in$ cluster3

A3:
d(A3, seed1)= $\sqrt{36}$ = 6
d(A3, seed2)= $\sqrt{25}$ = 5 ← smaller
d(A3, seed3)= $\sqrt{53}$ = 7.28
→ A3 $\in$ cluster2

A4:
d(A4, seed1)= $\sqrt{13}$
d(A4, seed2)=0 as A4 is seed2
d(A4, seed3)= $\sqrt{52}$ >0
→ A4 $\in$ cluster2

A5:
d(A5, seed1)= $\sqrt{50}$ = 7.07

A6:
d(A6, seed1)= $\sqrt{52}$ = 7.21

d(A5, seed2)= $\sqrt{13}$ = 3.60 ← smaller
d(A5, seed3)= $\sqrt{45}$ = 6.70
→ A5 ∈ cluster2

d(A6, seed2)= $\sqrt{17}$ = 4.12 ← smaller
d(A6, seed3)= $\sqrt{29}$ = 5.38
→ A6 ∈ cluster2

A7:
d(A7, seed1)= $\sqrt{65}$ >0
d(A7, seed2)= $\sqrt{52}$ >0
d(A7, seed3)=0 as A7 is seed3
→ A7 ∈ cluster3

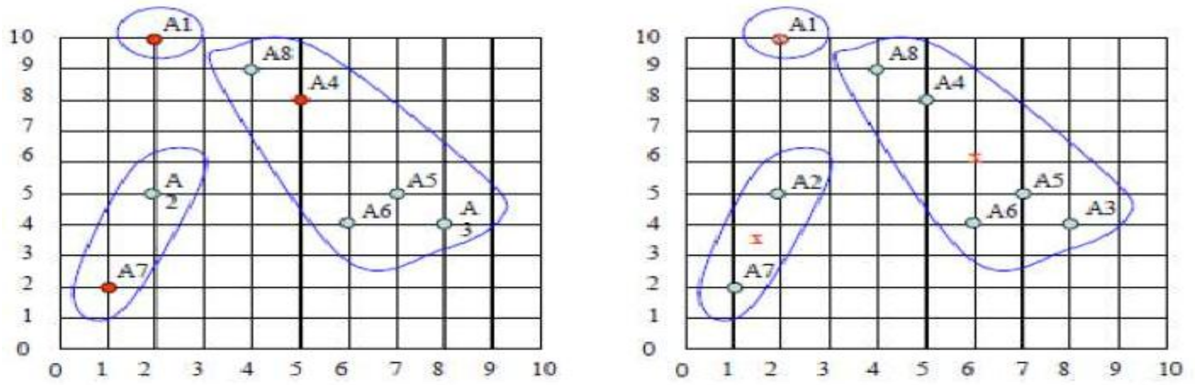
A8:
d(A8, seed1)= $\sqrt{5}$
d(A8, seed2)= $\sqrt{2}$ ← smaller
d(A8, seed3)= $\sqrt{58}$
→ A8 ∈ cluster2

end of epoch1

new clusters: 1: {A1}, 2: {A3, A4, A5, A6, A8}, 3: {A2, A7}

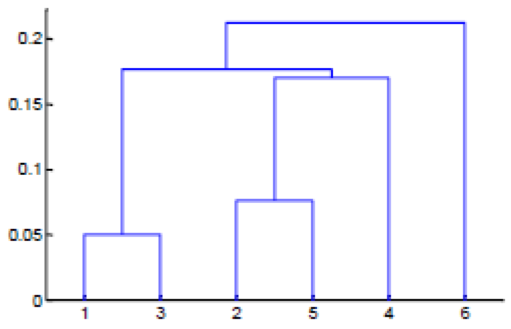
b) centers of the new clusters:
C1= (2, 10), C2= ((8+5+7+6+4)/5, (4+8+5+4+9)/5) = (6, 6), C3= ((2+1)/2, (5+2)/2) = (1.5, 3.5)

c)



5.3 Agglomerative Hierarchical Clustering

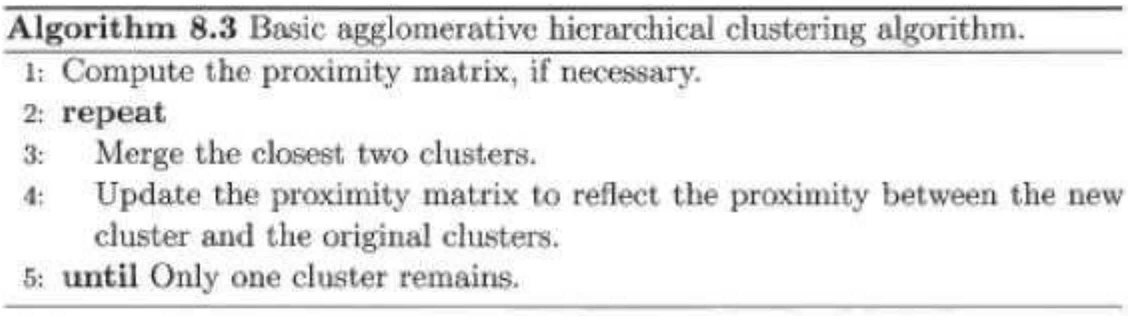
- More popular hierarchical clustering technique.
- Produces a set of nested clusters organized as a hierarchical tree.
- Can be visualized as a dendrogram.
- A tree like diagram that records the sequences of merges or splits.



Do not have to assume any particular number of clusters

– Any desired number of clusters can be obtained by „cutting" the dendogram at the proper level

Basic algorithm is

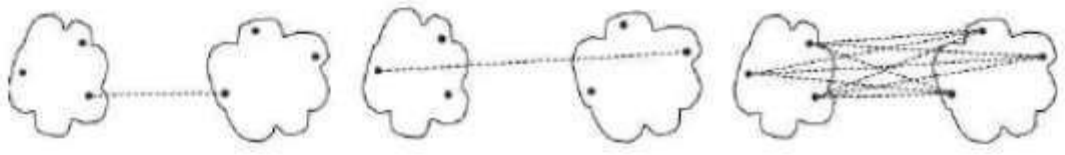


Algorithm 8.3 Basic agglomerative hierarchical clustering algorithm.

```
1: Compute the proximity matrix, if necessary.
2: repeat
3:    Merge the closest two clusters.
4:    Update the proximity matrix to reflect the proximity between the new
      cluster and the original clusters.
5: until Only one cluster remains.
```

How to Define Inter-Cluster Similarity (Proximity of two clusters)

MIN MAX
Group Average



(a) MIN (single link.) (b) MAX (complete link.) (c) Group average.

1) Single Link or MIN

For the single link or MIN version of hierarchical clustering, the proximity of two clusters is defined as the minimum of the distance (maximum of the similarity) between any two points in the two different clusters.

we shall use sample data that consists of 6 two-dimensional points, which are shown in Figure 8.15. The r and g coordinates of the points and the Euclidean distances between them are shown in Tables 8.3 and 8.4. respectively.

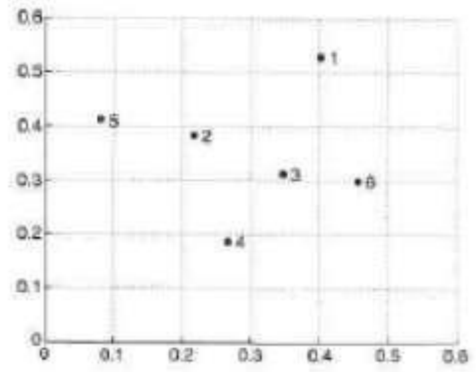


Figure 8.15. Set of 6 two-dimensional points.

Point	x Coordinate	y Coordinate
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

Table 8.3. xy coordinates of 6 points.

	p1	p2	p3	p4	p5	p6
p1	0.00	0.24	0.22	0.37	0.34	0.23
p2	0.24	0.00	0.15	0.20	0.14	0.25
p3	0.22	0.15	0.00	0.15	0.28	0.11
p4	0.37	0.20	0.15	0.00	0.29	0.22
p5	0.34	0.14	0.28	0.29	0.00	0.39
p6	0.23	0.25	0.11	0.22	0.39	0.00

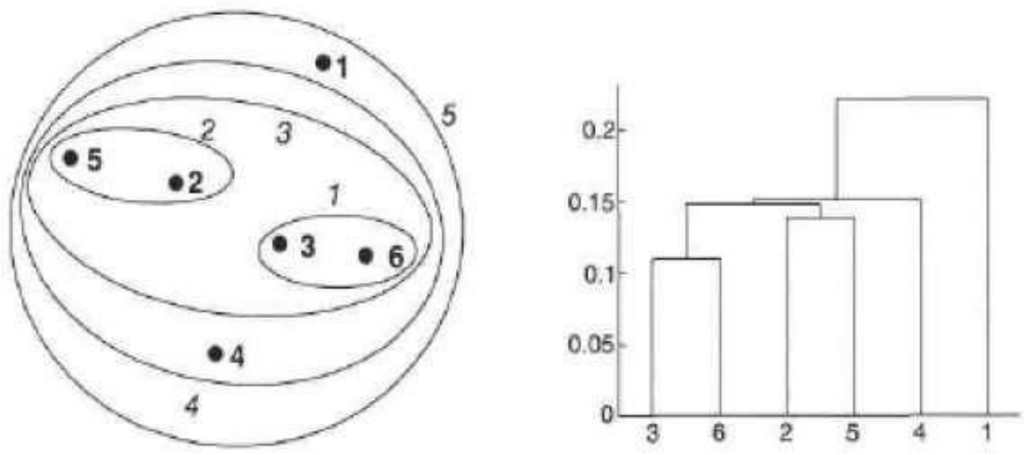
Table 8.4. Euclidean distance matrix for 6 points.

Figure 8.16 shows the result of applying the single link technique to our example data set of six points. Figure 8.16(a) shows the nested clusters as a sequence of nested ellipses, where the numbers associated with the ellipses indicate the order of the clustering. Figure S.16(b) shows the same information, but as a dendrogram.

The height at which two clusters are merged in the dendrogram reflects the distance of the two clusters.

For instance, from Table 8.4, we see that the distance between points 3 and 6 is 0.11, and that is the height at which they are joined into one cluster in the dendrogram. As another example, the distance between clusters {3,6} and {2,5} is given by

$$\begin{aligned} dist(\{3,6\},\{2,5\}) &= \min(dist(3,2), dist(6,2), dist(3,5), dist(6,5)) \\ &= \min(0.15, 0.25, 0.28, 0.39) \\ &= 0.15. \end{aligned}$$



(a) Single link clustering. (b) Single link dendrogram.

Figure 8.16. Single link clustering of the six points shown in Figure 8.15.

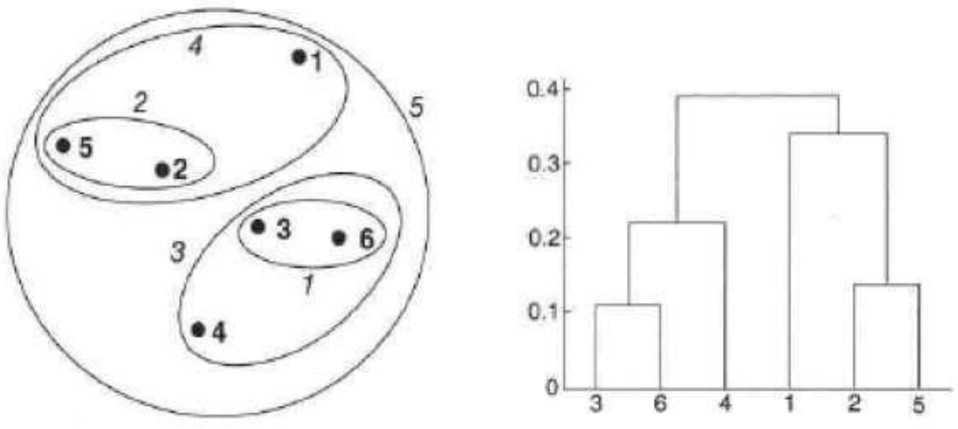
2) Complete Link or MAX or CLIQUE

For the complete link or MAX version of hierarchical clustering, the proximity of two clusters is defined as the maximum of the distance (minimum of the similarity) between any two points in the two different clusters. Using graph terminology, if you start with all points as singleton clusters and add links between points one at a time, shortest links first, then a group of points is not a cluster until all the points in it are completely linked, i.e., form a clique.

Example 8.5 (Complete Link). Figure 8.17 shows the results of applying MAX to the sample data set of six points. As with single link, points 3 and 6 are merged first. However, {3,6} is merged with {4}, instead of {2,5} or {1}.

are merged first. However, {3, 6} is merged with {4}, instead of {2, 5} or {1} because

$$
\begin{aligned}
dist(\{3,6\},\{4\}) &= \max(dist(3,4), dist(6,4)) \\
&= \max(0.15, 0.22) \\
&= 0.22. \\
dist(\{3,6\},\{2,5\}) &= \max(dist(3,2), dist(6,2), dist(3,5), dist(6,5)) \\
&= \max(0.15, 0.25, 0.28, 0.39) \\
&= 0.39. \\
dist(\{3,6\},\{1\}) &= \max(dist(3,1), dist(6,1)) \\
&= \max(0.22, 0.23) \\
&= 0.23.
\end{aligned}
$$



(a) Complete link clustering. (b) Complete link dendrogram.

Figure 8.17. Complete link clustering of the six points shown in Figure 8.15.

3) Group Average

For the group average version of hierarchical clustering, the proximity of two clusters is defined as the average pairwise proximity among all pairs of points in the different clusters.

This is an intermediate approach between the single and complete link approaches. Thus, for group average, the cluster proximity proximity(Ci,Cj) of clusters C,i and Cj', which are of size mi and mj, respectively, is expressed by the following equation.

$$proximity(C_i, C_j) = \frac{\sum_{\substack{x \in C_i \\ y \in C_j}} proximity(\mathbf{x}, \mathbf{y})}{m_i * m_j}.$$

Figure 8.18 shows the results of applying the group average approach to the sample data set of six points. To illustrate how group average works, we calculate the distance between some clusters.

$$\begin{aligned} dist(\{3,6,4\},\{1\}) &= (0.22 + 0.37 + 0.23)/(3 * 1) \\ &= 0.28 \\ dist(\{2,5\},\{1\}) &= (0.2357 + 0.3421)/(2 * 1) \\ &= 0.2889 \\ dist(\{3,6,4\},\{2,5\}) &= (0.15 + 0.28 + 0.25 + 0.39 + 0.20 + 0.29)/(6 * 2) \\ &= 0.26 \end{aligned}$$

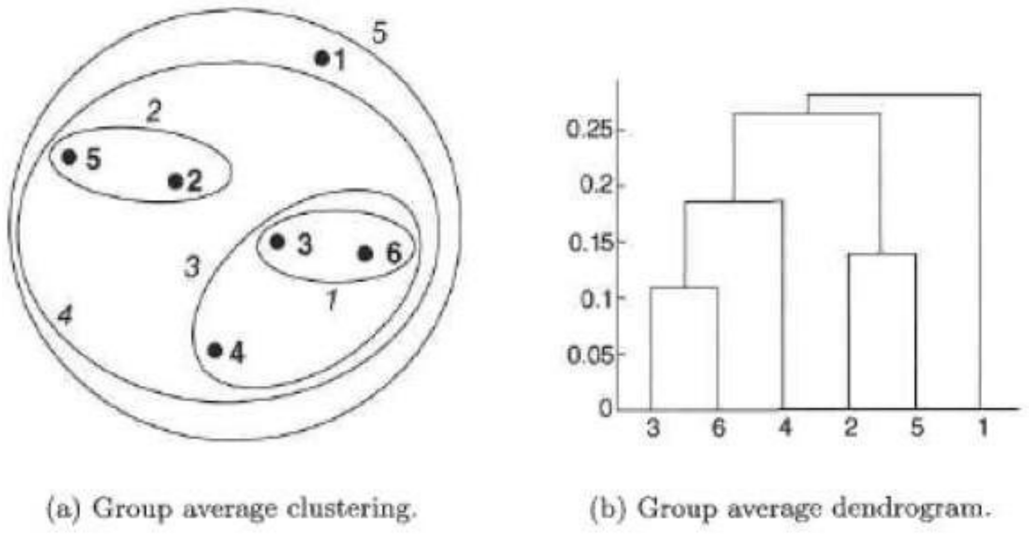


Figure 8.18. Group average clustering of the six points shown in Figure 8.15.

Key Issues in Agglomerative Hierarchical Clustering (Strengths and Weaknesses)

- Once a decision is made to combine two clusters, it cannot be undone
- No objective function is directly minimized.
- Different schemes have problems with one or more of the following:

- Sensitivity to noise and outliers

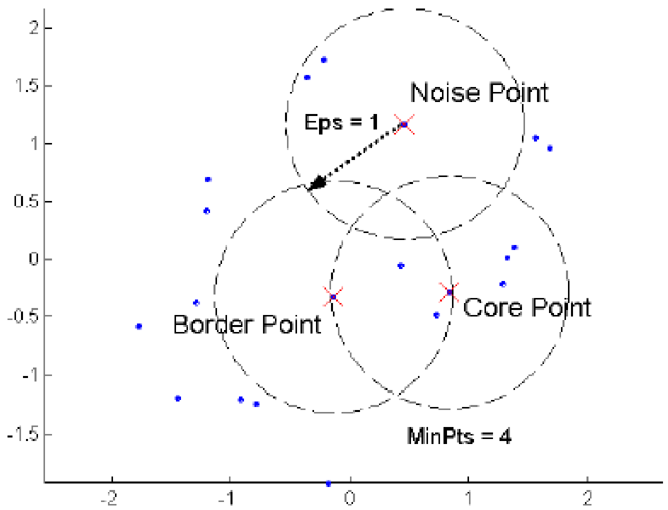
– Difficulty handling different sized clusters and convex shapes

– Breaking large clusters

5.5 The DBSCAN Algorithm

DBSCAN is a density-based algorithm.

- 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.



Eliminate noise points
Perform clustering on the remaining points

```
$current\_cluster\_label \leftarrow 1$
for all core points do
  if the core point has no cluster label then
    $current\_cluster\_label \leftarrow current\_cluster\_label + 1$
    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
```

Strengths and weaknesses of DBSCAN

- It is relatively Resistant to Noise.
- It can Can handle clusters of different shapes and sizes
- Does NOT Work Well when the clusters having Varying densities
- Does NOT Work Well With High-dimensional data.

Exercise 5: DBScan
If Epsilon is 2 and minpoint is 2, what are the clusters that DBScan would discover with the following 8 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 is the same as the one in Exercise 1. Draw the 10 by 10 space and illustrate the discovered clusters. What if Epsilon is increased to $\sqrt{10}$?

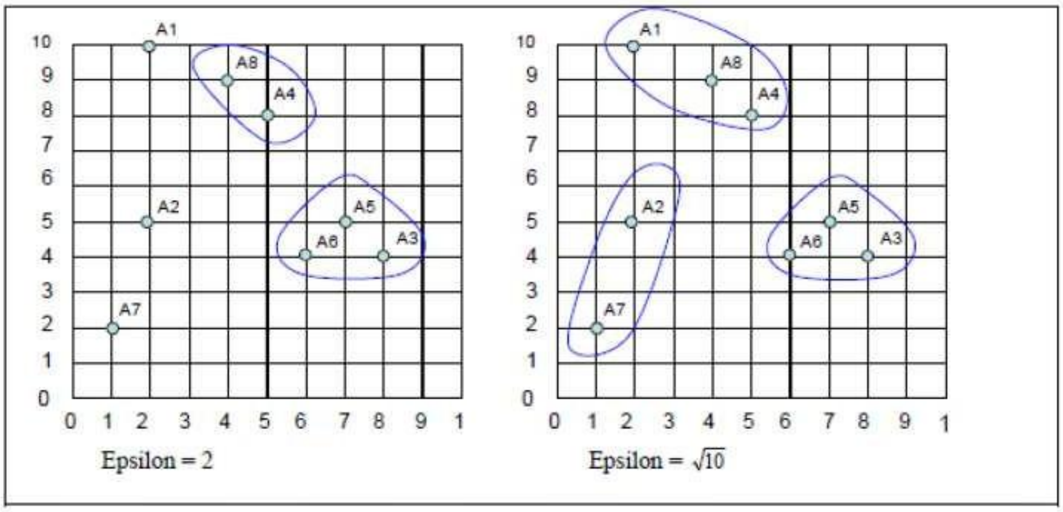
Solution:
What is the Epsilon neighborhood of each point?
$N_2(A1)=\{\}$; $N_2(A2)=\{\}$; $N_2(A3)=\{A5, A6\}$; $N_2(A4)=\{A8\}$; $N_2(A5)=\{A3, A6\}$;
$N_2(A6)=\{A3, A5\}$; $N_2(A7)=\{\}$; $N_2(A8)=\{A4\}$

So A1, A2, and A7 are outliers, while we have two clusters C1={A4, A8} and C2={A3, A5, A6}

If Epsilon is $\sqrt{10}$ then the neighborhood of some points will increase:
A1 would join the cluster C1 and A2 would joint with A7 to form cluster C3={A2, A7}.

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



5.6 Cluster Evaluation:

Why do we want to evaluate them?

- To avoid finding patterns in noise.
- To compare clustering algorithms
- To compare two sets of clusters.
- To compare two clusters.

Different Aspects of Cluster Validation:

1. Determining the clustering tendency of a set of data, i.e., distinguishing whether non-random structure actually exists in the data.

2. Comparing the results of a cluster analysis to externally known results, e.g., to externally given class labels.

3. Evaluating how well the results of a cluster analysis fit the data *without* reference to external information.
- Use only the data
1. Comparing the results of two different sets of cluster analyses to determine which is better.

2. Determining the „correct" number of clusters.

For 2, 3, and 4, we can further distinguish whether we want to evaluate the entire clustering or just individual clusters.

Supervised Cluster Evaluation Using Cohesion and Separation
Cluster cohesion (compactness, tightness), which determine how closely related the objects in a cluster are.
cluster separation (isolation), which determine how distinct or well separated a cluster is from other clusters.
Graph-Based View of Cohesion and Separation:
For graph-based clusters, the cohesion of a cluster can be defined as the sum of the weights of the links in the proximity graph that connect points within the cluster.
The separation between two clusters can be measured by the sum of the weights of the links from points in one cluster to points in the other cluster.
Mathematically, cohesion and separation for a graph-based cluster can be expressed using Equations 8.9 and 8.10, respectively.

$$cohesion(C_i) = \sum_{\substack{\mathbf{x}\in C_i \\ \mathbf{y}\in C_i}} proximity(\mathbf{x},\mathbf{y}) \quad (8.9)$$

$$separation(C_i, C_j) = \sum_{\substack{\mathbf{x}\in C_i \\ \mathbf{y}\in C_j}} proximity(\mathbf{x},\mathbf{y}) \quad (8.10)$$

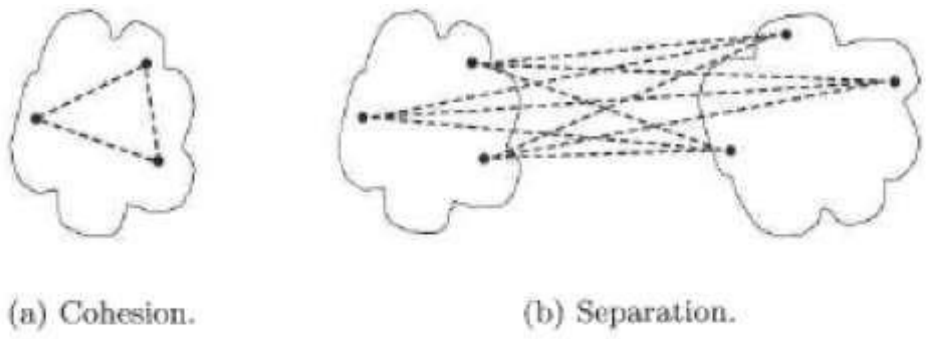


Figure 8.27. Graph-based view of cluster cohesion and separation.

Prototype-Based View of Cohesion and Separation

For prototype-based clusters, the cohesion of a cluster can be defined as the sum of the proximities with respect to the prototype (centroid or medoid) of the cluster. Similarly, the separation between two clusters can be measured by the proximity of the two cluster prototypes

$$cohesion(C_i) = \sum_{\mathbf{x} \in C_i} proximity(\mathbf{x}, \mathbf{c}_i) \qquad (8.11)$$

$$separation(C_i, C_j) = proximity(\mathbf{c}_i, \mathbf{c}_j) \qquad (8.12)$$

$$separation(C_i) = proximity(\mathbf{c}_i, \mathbf{c}) \qquad (8.13)$$

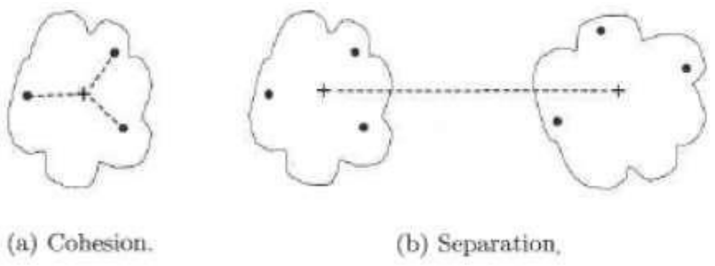


Figure 8.28. Prototype-based view of cluster cohesion and separation.

Measuring Cluster Validity Via Correlation

- Two matrices
- Similarity or Distance Matrix
- One row and one column for each data point
- An entry is the similarity or distance of the associated pair of points

- "Incidence" Matrix
- One row and one column for each data point
- An entry is 1 if the associated pair of points belong to the same cluster
- An entry is 0 if the associated pair of points belongs to different clusters

- Compute the correlation between the two matrices
- Since the matrices are symmetric, only the correlation between n(n-1) / 2 entries needs to be calculated.

- High correlation (positive for similarity, negative for distance) indicates that points that belong to the same cluster are close to each other.
- Not a good measure for some density or contiguity based clusters.

Supervised Measures of Cluster Validity (External Measures for Clustering Validity)

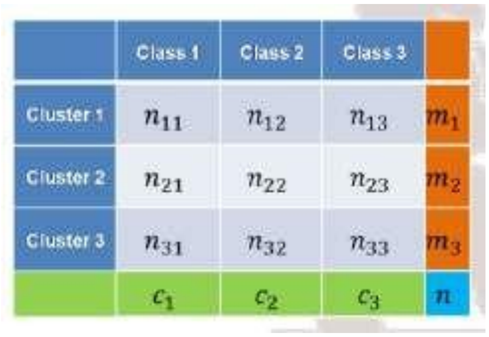
Two different kinds of approaches.

The first set of techniques use measures from classification, such as entropy, purity, and the F-measure. These measures evaluate the extent to which a cluster contains objects of a single class. The second group of methods is related to the similarity measures for binary data, such as the Jaccard measure . These approaches measure the extent to which two objects that are in the same class are in the same cluster and vice versa.

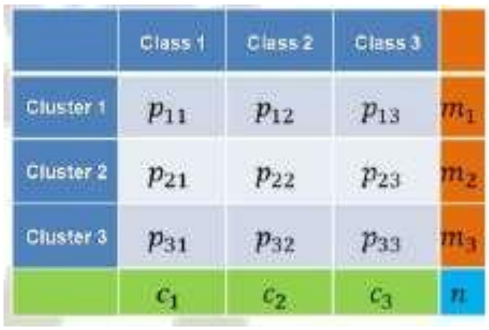
Classification-Oriented Measures of Cluster Validity

- Assume that the data is labeled with some class labels
- E.g., documents are classified into topics, people classified according to their income, politicians classified according to the political party.
- This is called the "ground truth"
- In this case we want the clusters to be homogeneous with respect to classes
- Each cluster should contain elements of mostly one class Each class should ideally be assigned to a single cluster

Confusion matrix

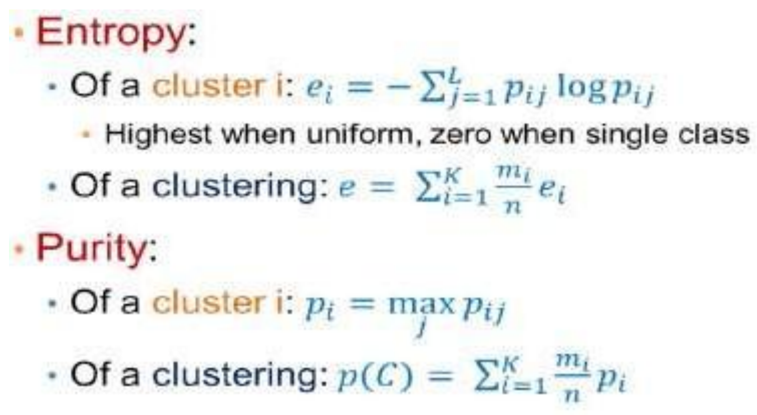


	Class 1	Class 2	Class 3	
Cluster 1	n_{11}	n_{12}	n_{13}	m_1
Cluster 2	n_{21}	n_{22}	n_{23}	m_2
Cluster 3	n_{31}	n_{32}	n_{33}	m_3
	c_1	c_2	c_3	n



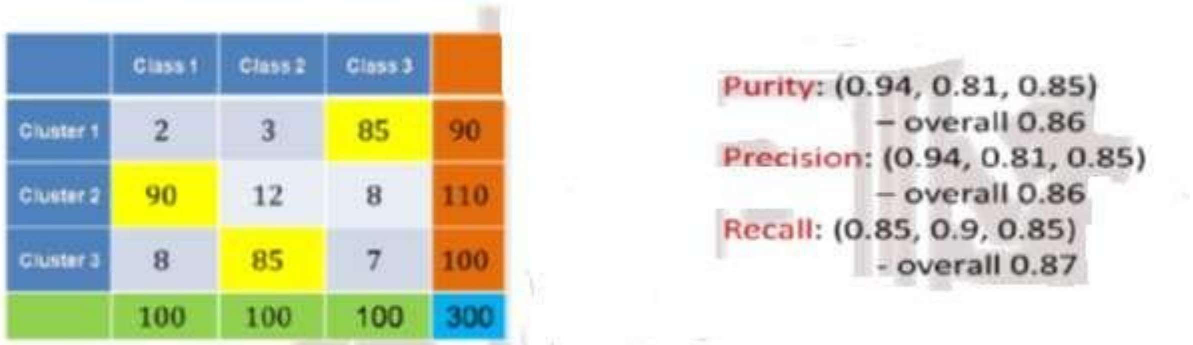
	Class 1	Class 2	Class 3	
Cluster 1	p_{11}	p_{12}	p_{13}	m_1
Cluster 2	p_{21}	p_{22}	p_{23}	m_2
Cluster 3	p_{31}	p_{32}	p_{33}	m_3
	c_1	c_2	c_3	n

Measures:



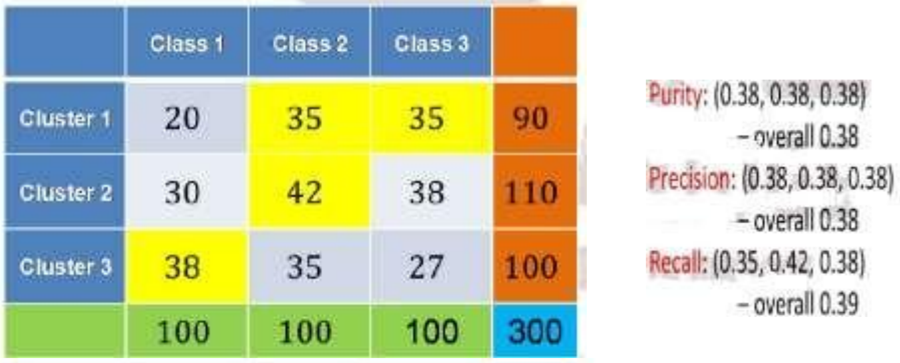
- Entropy:
  - Of a cluster i: $e_i = -\sum_{j=1}^{L} p_{ij} \log p_{ij}$
    - Highest when uniform, zero when single class
  - Of a clustering: $e = \sum_{i=1}^{K} \frac{m_i}{n} e_i$
- Purity:
  - Of a cluster i: $p_i = \max_j p_{ij}$
  - Of a clustering: $p(C) = \sum_{i=1}^{K} \frac{m_i}{n} p_i$

Example:



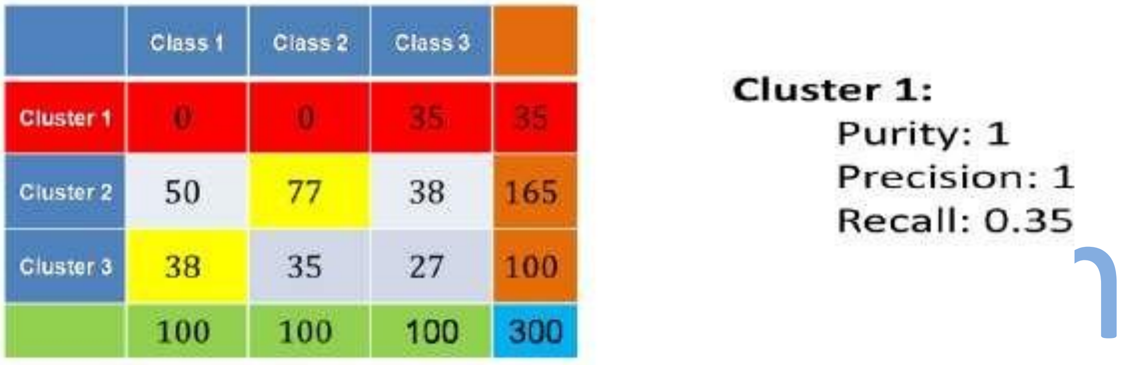
	Class 1	Class 2	Class 3	
Cluster 1	2	3	85	90
Cluster 2	90	12	8	110
Cluster 3	8	85	7	100
	100	100	100	300

Purity: (0.94, 0.81, 0.85)
– overall 0.86
Precision: (0.94, 0.81, 0.85)
– overall 0.86
Recall: (0.85, 0.9, 0.85)
- overall 0.87



	Class 1	Class 2	Class 3	
Cluster 1	20	35	35	90
Cluster 2	30	42	38	110
Cluster 3	38	35	27	100
	100	100	100	300

Purity: (0.38, 0.38, 0.38)
– overall 0.38
Precision: (0.38, 0.38, 0.38)
– overall 0.38
Recall: (0.35, 0.42, 0.38)
– overall 0.39



	Class 1	Class 2	Class 3	
Cluster 1	0	0	35	35
Cluster 2	50	77	38	165
Cluster 3	38	35	27	100
	100	100	100	300

Cluster 1:
Purity: 1
Precision: 1
Recall: 0.35

Similarity-Oriented Measures of Cluster Validity

The measures that we discuss in this section are all based on the premise that any two objects that are in the same cluster should be in the same class and vice versa.

We can view this approach to cluster validity as involving the comparison of two matrices: (1)The ideal cluster similarity matrix discussed previously, which has a 1 in the (i,j)th entry iftwo objects, i. and j, are in the same cluster and 0, otherwise.

(2) An ideal class similarity matrix defined with respect to class labels, which has a 1 in the (i,j) th entry if two objects, i and j, belong to the same class, and a 0 otherwise.

Example 8.16 (Correlation between Cluster and Class Matrices). To demonstrate this idea more concretely, we give an example involving five data points, p_1, p_2, p_3, p_4, p_5, two clusters, $C_1 = \{p_1, p_2, p_3\}$ and $C_2 = \{p_4, p_5\}$, and two classes, $L_1 = \{p_1, p_2\}$ and $L2 = \{p_3, p_4, p_5\}$. The ideal cluster and class similarity matrices are given in Tables 8.10 and 8.11. The correlation between the entries of these two matrices is 0.359.

Table 8.10. Ideal cluster similarity matrix.

Point	p1	p2	p3	p4	p5
p1	1	1	1	0	0
p2	1	1	1	0	0
p3	1	1	1	0	0
p4	0	0	0	1	1
p5	0	0	0	1	1

Table 8.11. Ideal class similarity matrix.

Point	p1	p2	p3	p4	p5
p1	1	1	0	0	0
p2	1	1	0	0	0
p3	0	0	1	1	1
p4	0	0	1	1	1
p5	0	0	1	1	1

We also note that the four quantities, f_{00}, f_{01}, f_{10}, and f_{11}, define a *contingency* table as shown in Table 8.12.

Table 8.12. Two-way contingency table for determining whether pairs of objects are in the same class and same cluster.

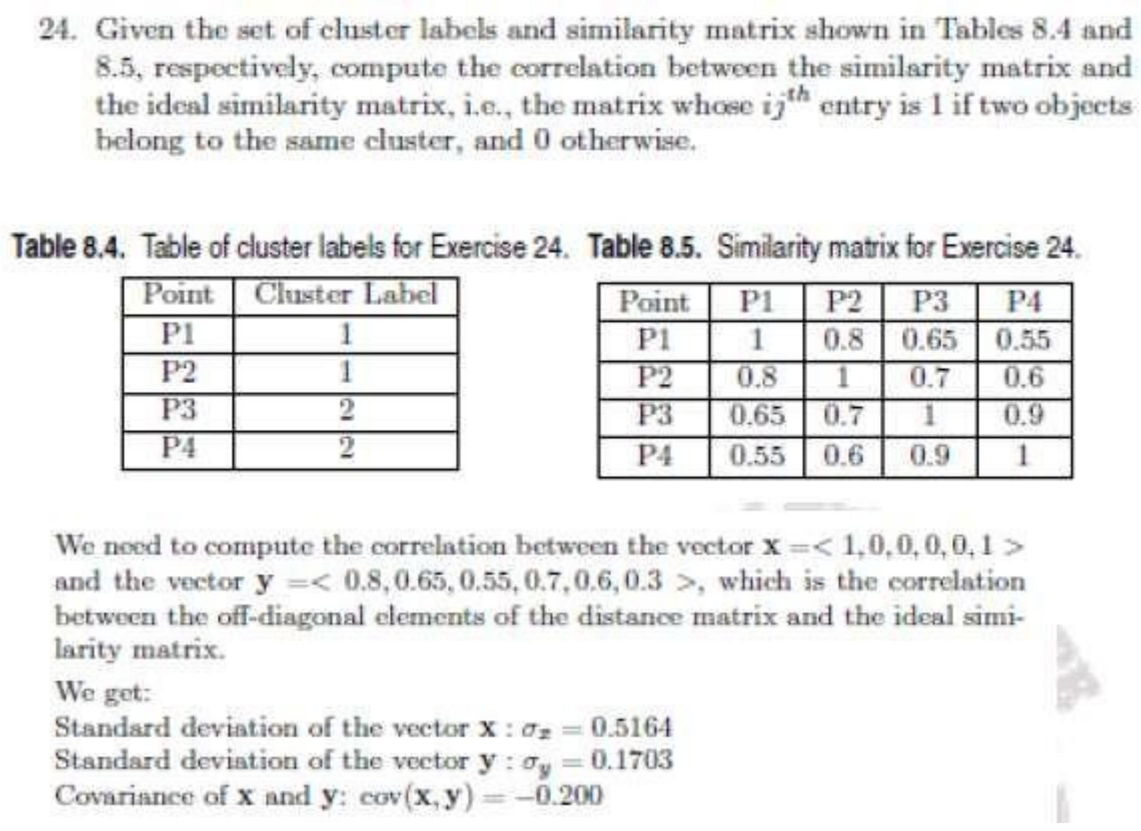
	Same Cluster	Different Cluster
Same Class	f_{11}	f_{10}
Different Class	f_{01}	f_{00}

f_{00} = number of pairs of objects having a different class and a different cluster
f_{01} = number of pairs of objects having a different class and the same cluster
f_{10} = number of pairs of objects having the same class and a different cluster
f_{11} = number of pairs of objects having the same class and the same cluster

In particular, the simple matching coefficient, which is known as the Rand statistic in this context, and the Jaccard coefficient are two of the most frequently used cluster validity measures.

$$\text{Rand statistic} = \frac{f_{00} + f_{11}}{f_{00} + f_{01} + f_{10} + f_{11}} \qquad (8.18)$$

$$\text{Jaccard coefficient} = \frac{f_{11}}{f_{01} + f_{10} + f_{11}} \qquad (8.19)$$



24. Given the set of cluster labels and similarity matrix shown in Tables 8.4 and 8.5, respectively, compute the correlation between the similarity matrix and the ideal similarity matrix, i.e., the matrix whose ij^{th} entry is 1 if two objects belong to the same cluster, and 0 otherwise.

Table 8.4. Table of cluster labels for Exercise 24.

Point	Cluster Label
P1	1
P2	1
P3	2
P4	2

Table 8.5. Similarity matrix for Exercise 24.

Point	P1	P2	P3	P4
P1	1	0.8	0.65	0.55
P2	0.8	1	0.7	0.6
P3	0.65	0.7	1	0.9
P4	0.55	0.6	0.9	1

We need to compute the correlation between the vector $\mathbf{x} = < 1, 0, 0, 0, 0, 1 >$ and the vector $\mathbf{y} = < 0.8, 0.65, 0.55, 0.7, 0.6, 0.3 >$, which is the correlation between the off-diagonal elements of the distance matrix and the ideal similarity matrix.

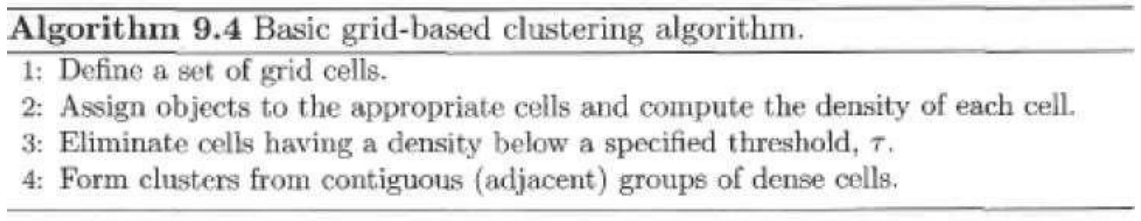
We get:
Standard deviation of the vector $\mathbf{x}$: $\sigma_x = 0.5164$
Standard deviation of the vector $\mathbf{y}$: $\sigma_y = 0.1703$
Covariance of $\mathbf{x}$ and $\mathbf{y}$: $\text{cov}(\mathbf{x}, \mathbf{y}) = -0.200$

5.7 Density-Based Clustering

- Grid-Based Clustering
- Subspace Clustering
- CLIQUE
- DENCLUE: A Kernel-Based Scheme for Density-Based
- Clustering

Grid-Based Clustering

The idea is to split the possible values of each attribute into a number of contiguous intervals, creating a set of grid cells.
Objects can be assigned to grid cells in one pass through the data, and information about each cell, such as the number of points in the cell, can also be gathered at the same time.



Algorithm 9.4 Basic grid-based clustering algorithm.

```
1: Define a set of grid cells.
2: Assign objects to the appropriate cells and compute the density of each cell.
3: Eliminate cells having a density below a specified threshold, $\tau$.
4: Form clusters from contiguous (adjacent) groups of dense cells.
```

Defining Grid Cells: This is a key step in the process, but also the least well defined, as there are many ways to split the possible values of each attribute into a number of contiguous intervals.

For continuous attributes, one common approach is to split the values into equal width intervals. If this approach is applied to each attribute, then the resulting grid cells all have the same olume, and the density of a cell is conveniently defined as the number of points in the cell.
The Density of Grid Cells: A natural way to define the density of a grid cell (or a more generally shaped region) is as the number of points divided by the volume of the region. In other words, density is the number of points per amount of space, regardless of the dimensionality of that space.

Example: Figure 9.10 shows two sets of two dimensional points divided into 49 cells using a 7- by-7 grid. The first set contains 200 points generated from a uniform distribution over a circle centered at (2, 3) of radius 2, while the second set has 100 points generated from a uniform distribution over a circle centered at (6, 3) of radius 1. The counts for the grid cells are shown in Table 9.2.

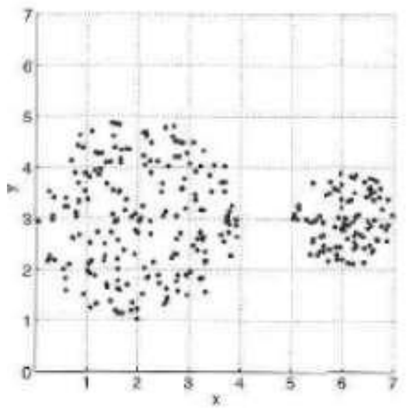


Figure 9.10. Grid-based density.

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 9.2. Point counts for grid cells.

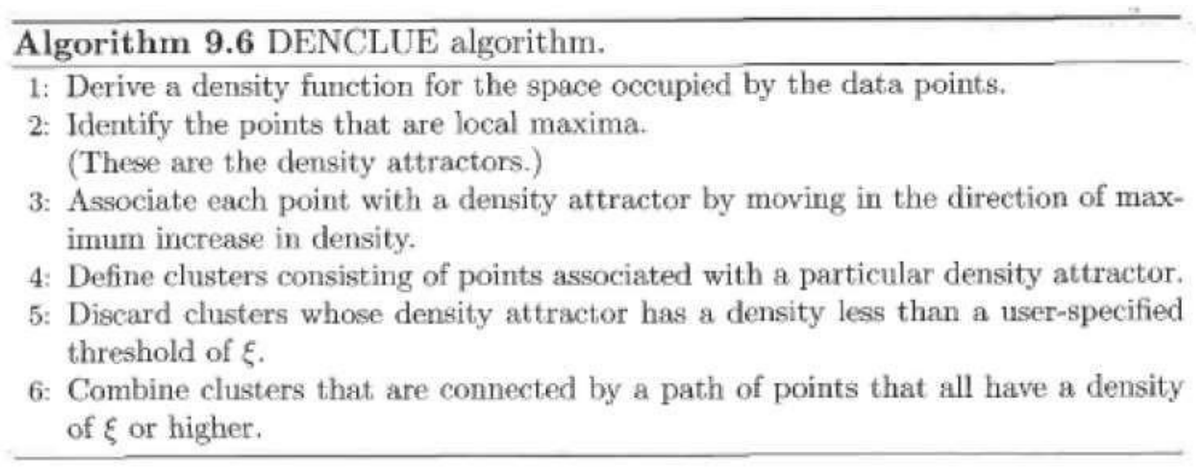
CLIQUE

CLIQUE (Clustering In QUEst) is a grid-based clustering algorithm that methodically finds subspace clusters. It is impractical to check each subspace for clusters since the number of such subspaces is exponential in the number of dimensions. Instead, CLIQUE relies on the following property;
Monotonicity property of density-based clusters If a set of points forms a density-based cluster in k dimensions (attributes), then the same set of points is also part of a density-based cluster in all possible subsets of those dimensions.

Algorithm 9.5 CLIQUE.

```
1: Find all the dense areas in the one-dimensional spaces corresponding to each
   attribute. This is the set of dense one-dimensional cells.
2: $k \leftarrow 2$
3: repeat
4:    Generate all candidate dense $k$-dimensional cells from dense $(k-1)$-dimensional
      cells.
5:    Eliminate cells that have fewer than $\xi$ points.
6:    $k \leftarrow k+1$
7: until There are no candidate dense $k$-dimensional cells.
8: Find clusters by taking the union of all adjacent, high-density cells.
9: Summarize each cluster using a small set of inequalities that describe the attribute
   ranges of the cells in the cluster.
```

DENCLUE: A Kernel-Based Scheme for Density-Based Clustering DENCLUE (DENsity ClUstEring) is a density-based clustering approach that models the overall density of a set of points as the sum of influence functions associated with each point. The resulting overall density function will have local peaks, i.e., local density maxima, and these local peaks can be used to define clusters in a natural way. Specifically, for each data point, a hill climbing procedure finds the nearest peak associated with that point, and the set of all data points associated with a particular peak (called a local density attractor) becomes a cluster.



Algorithm 9.6 DENCLUE algorithm.

```
1: Derive a density function for the space occupied by the data points.
2: Identify the points that are local maxima.
   (These are the density attractors.)
3: Associate each point with a density attractor by moving in the direction of max-
   imum increase in density.
4: Define clusters consisting of points associated with a particular density attractor.
5: Discard clusters whose density attractor has a density less than a user-specified
   threshold of ξ.
6: Combine clusters that are connected by a path of points that all have a density
   of ξ or higher.
```

5.8 Graph-Based Clustering

Graph-Based clustering uses the proximity graph

- Start with the proximity matrix
- Consider each point as a node in a graph
- Each edge between two nodes has a weight which is the proximity between the two points
- Initially the proximity graph is fully connected
- MIN (single-link) and MAX (complete-link) can be viewed as starting with this graph In the simplest case, clusters are connected components in the graph.

Sparsification

The amount of data that needs to be processed is drastically reduced.

- Sparsification can eliminate more than 99% of the entries in a proximity matrix
- The amount of time required to cluster the data is drastically reduced
- The size of the problems that can be handled is increased.

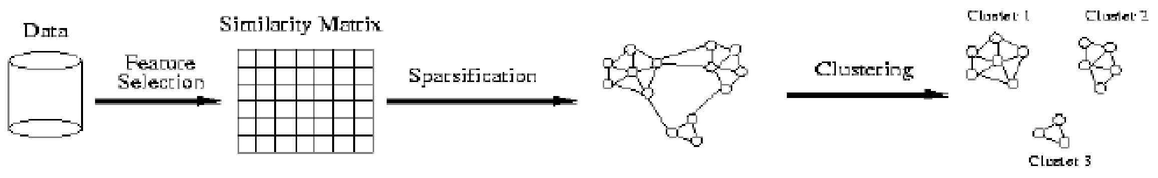
Clustering may work better

Sparsification techniques keep the connections to the most similar (nearest) neighbors of a point while breaking the connections to less similar points.

- The nearest neighbors of a point tend to belong to the same class as the point itself.
- This reduces the impact of noise and outliers and sharpens the distinction between clusters.

Sparsification facilitates the use of graph partitioning algorithms (or algorithms based on graph partitioning algorithms.

Chameleon and Hypergraph-based Clustering



Limitations of Current Merging Schemes

Existing merging schemes in hierarchical clustering algorithms are static in nature

– MIN or CURE:

merge two clusters based on their *closeness* (or minimum distance)

– GROUP-AVERAGE:

merge two clusters based on their average *connectivity*

Chameleon: Clustering Using Dynamic Modeling.

Adapt to the characteristics of the data set to find the natural clusters Use a dynamic model to measure the similarity between clusters

- Main property is the relative closeness and relative inter-connectivity of the cluster
- Two clusters are combined if the resulting cluster shares certain *properties* with the constituent clusters
- The merging scheme preserves *self-similarity*

Steps

Preprocessing Step: Represent the Data by a Graph

➢ Given a set of points, construct the k-nearest-neighbor (k-NN) graph to capture the relationship between a point and its k nearest neighbors

➢ Concept of neighborhood is captured dynamically (even if region is sparse)

Phase 1: Use a multilevel graph partitioning algorithm on the graph to find a large number of clusters of well-connected vertices.

➢ Each cluster should contain mostly points from one “true” cluster, i.e., is a sub-cluster of a “real” cluster.

Phase 2: Use Hierarchical Agglomerative Clustering to merge sub-clusters

➢ Two clusters are combined if the *resulting cluster shares certain properties with the constituent clusters*

➢ Two key properties used to model cluster similarity:

Relative Interconnectivity: Absolute interconnectivity of two clusters normalized by the internal connectivity of the clusters

Relative Closeness: Absolute closeness of two clusters normalized by the internal closeness of the clusters

SNN Clustering Algorithm:

1) Compute the similarity matrix

This corresponds to a similarity graph with data points for nodes and edges whose weights are the similarities between data points

2) Sparsify the similarity matrix by keeping only the *k* most similar neighbors This corresponds to only keeping the *k* strongest links of the similarity graph

3) Construct the shared nearest neighbor graph from the sparsified similarity matrix.

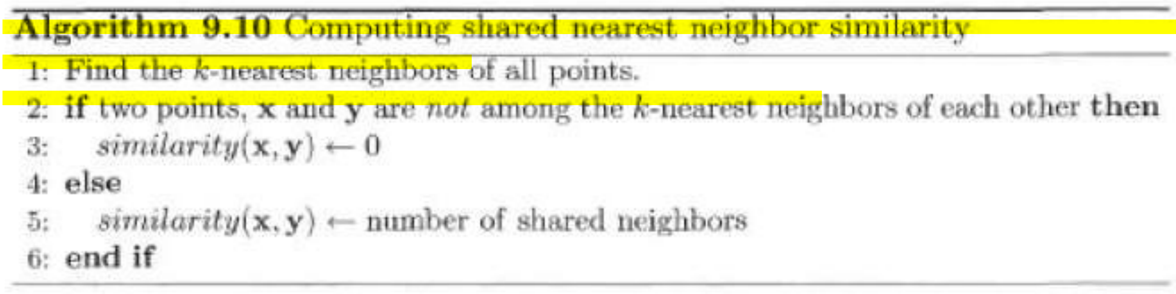
At this point, we could apply a similarity threshold and find the connected components to obtain the clusters (Jarvis-Patrick algorithm)

4) Find the SNN density of each Point.

Using a user specified parameters, *Eps*, find the number points that have an SNN imilarity of *Eps* or greater to each point. This is the SNN density of the point. 5)Find the core points

Using a user specified parameter, *MinPts*, find the core points, i.e., all points that have an SNN density greater than *MinPts.*

6)Form clusters from the core points.
If two core points are within a radius, *Eps*, of each other they are place in the same cluster. 7)Discard all noise points.
All non-core points that are not within a radius of *Eps* of a core point are discarded . 8)Assign all non-noise, non-core points to clusters.
This can be done by assigning such points to the nearest core point



Algorithm 9.10 Computing shared nearest neighbor similarity

```
1: Find the $k$-nearest neighbors of all points.
2: if two points, $\mathbf{x}$ and $\mathbf{y}$ are not among the $k$-nearest neighbors of each other then
3:     $similarity(\mathbf{x}, \mathbf{y}) \leftarrow 0$
4: else
5:     $similarity(\mathbf{x}, \mathbf{y}) \leftarrow$ number of shared neighbors
6: end if
```

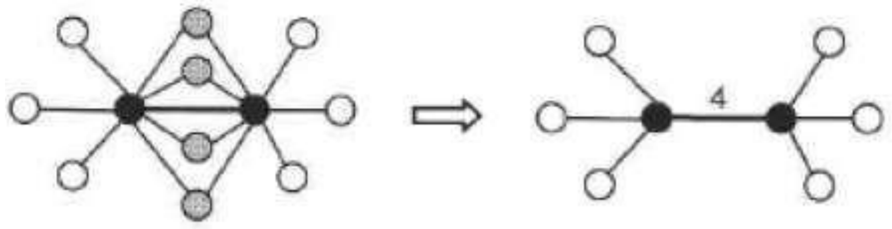


Figure 9.23. Computation of SNN similarity between two points.

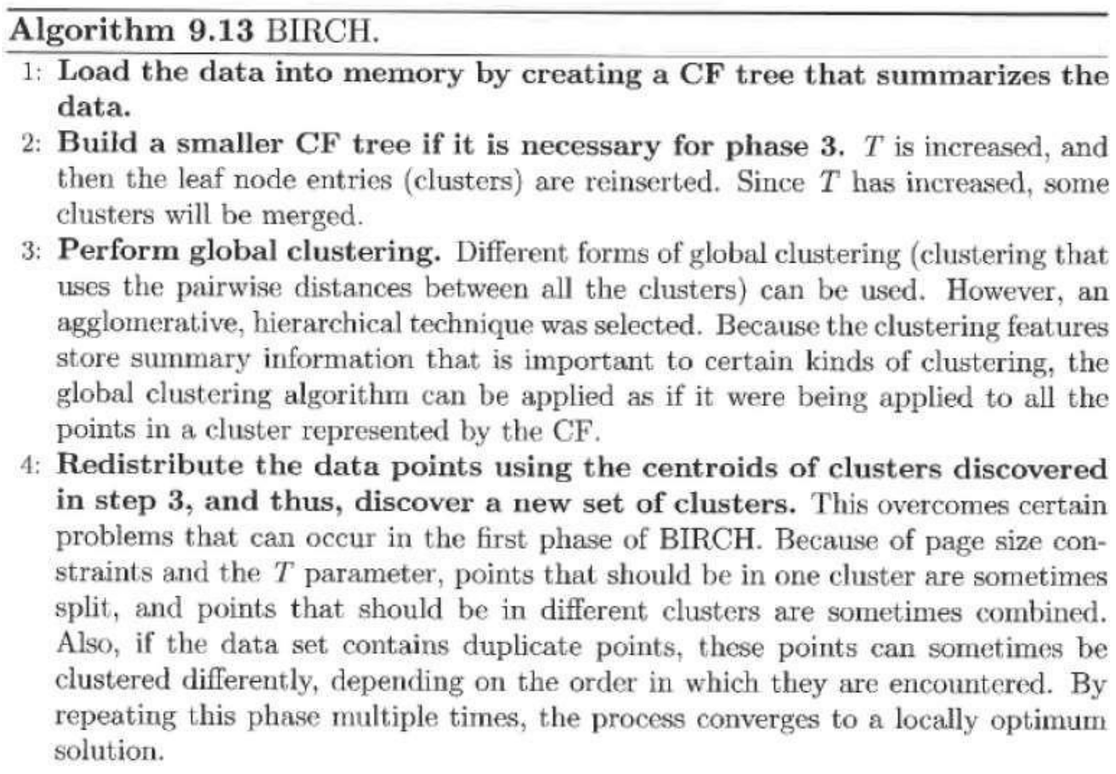
Limitations of SNN Clustering Complexity of SNN Clustering is high

– O(n * time to find numbers of neighbor within Eps)

– In worst case, this is O(n2)

5.9 Scalable Clustering Algorithms BIRCH

BIRCH (Balanced Iterative Reducing and Clustering using Hierarchies) is a highly efficient clustering technique for data in Euclidean vector spaces, i'e., data for which averages make sense. BIRCH can efficiently cluster such data with one pass and can improve that clustering with additional passes. BIRCH can also deal effectively with outliers.



Algorithm 9.13 BIRCH.

1: **Load the data into memory by creating a CF tree that summarizes the data.**

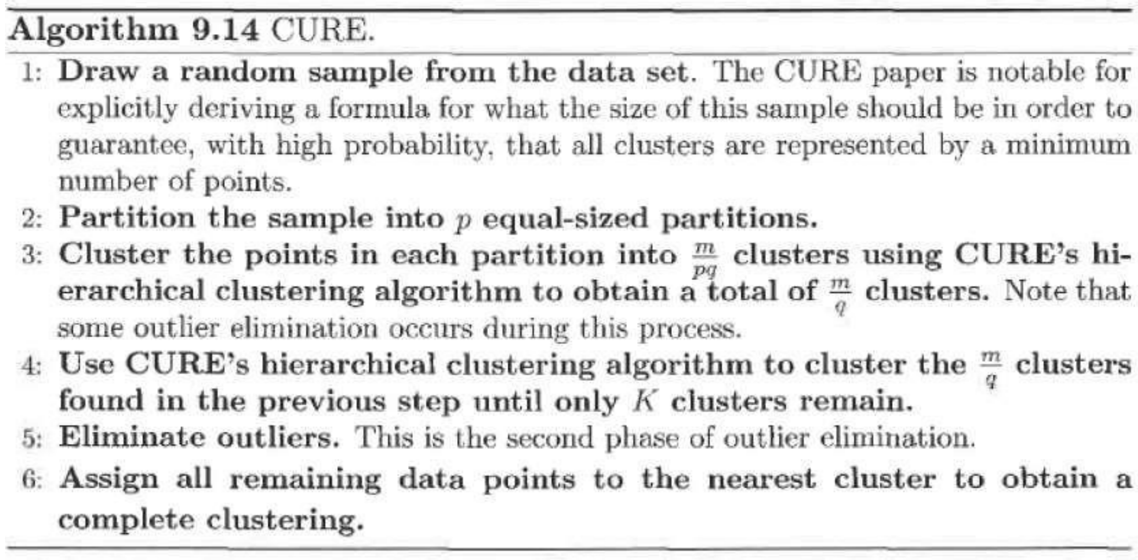
2: **Build a smaller CF tree if it is necessary for phase 3.** T is increased, and then the leaf node entries (clusters) are reinserted. Since T has increased, some clusters will be merged.

3: **Perform global clustering.** Different forms of global clustering (clustering that uses the pairwise distances between all the clusters) can be used. However, an agglomerative, hierarchical technique was selected. Because the clustering features store summary information that is important to certain kinds of clustering, the global clustering algorithm can be applied as if it were being applied to all the points in a cluster represented by the CF.

4: **Redistribute the data points using the centroids of clusters discovered in step 3, and thus, discover a new set of clusters.** This overcomes certain problems that can occur in the first phase of BIRCH. Because of page size constraints and the T parameter, points that should be in one cluster are sometimes split, and points that should be in different clusters are sometimes combined. Also, if the data set contains duplicate points, these points can sometimes be clustered differently, depending on the order in which they are encountered. By repeating this phase multiple times, the process converges to a locally optimum solution.

CURE

CURE (Clustering Using REpresentatives) is a clustering algorithm that uses a variety of different techniques to create an approach that can handle large data sets, outliers, and clusters with non-spherical shapes and non-uniform sizes. CURE represents a cluster by using multiple representative points from the cluster. These points will, in theory, capture the geometry and shape of the cluster. The first representative point is chosen to be the point farthest from
the center of the cluster, while the remaining points are chosen so that they are farthest from all the previously chosen points. In this way, the representative points are naturally relatively well distributed.



Algorithm 9.14 CURE.

1: **Draw a random sample from the data set.** The CURE paper is notable for explicitly deriving a formula for what the size of this sample should be in order to guarantee, with high probability, that all clusters are represented by a minimum number of points.

2: **Partition the sample into p equal-sized partitions.**

3: **Cluster the points in each partition into $\frac{m}{pq}$ clusters using CURE's hierarchical clustering algorithm to obtain a total of $\frac{m}{q}$ clusters.** Note that some outlier elimination occurs during this process.

4: **Use CURE's hierarchical clustering algorithm to cluster the $\frac{m}{q}$ clusters found in the previous step until only K clusters remain.**

5: **Eliminate outliers.** This is the second phase of outlier elimination.

6: **Assign all remaining data points to the nearest cluster to obtain a complete clustering.**

5.10 Question Bank: Clustering Analysis

1. Explain desired features of cluster analysis.
2. Explain how distance between a pair of points can be computed.
3. Write a short note on density-based methods.
4. Write and explain basic K-Means algorithm.
5. Explain DBSCAN clustering algorithm.
6. What are the limitations of K Means algorithm.
7. Explain cluster analysis methods briefly.
8. Explain agglomerative hierarchical clustering.
9. Explain bisecting K Means algorithm.
10. Distinguish between various types of clustering.
11. What are unsupervised, supervised and relative evaluation measures that are applied to judge various aspects of cluster validity.
12. Explain different types of defining proximity between clusters.
13. Differentiate between exclusive and overlapping clustering.
14. What are the various issues considered for cluster validation? Explain different evaluation measures used for cluster validity.
15. Explain unsupervised cluster evaluation using cohesion and separation.
16. Explain unsupervised cluster evaluation using proximity matrix.
17. List and explain classification-oriented measures of cluster validity.
18. Explain similarity – oriented measures of cluster validity.
19. Explain grid-based clustering algorithm.
20. Explain subspace clustering.
21. Write and explain CLIQUE algorithm.
22. Write and explain DENCLUE algorithm.
23. Explain different graph-based clustering