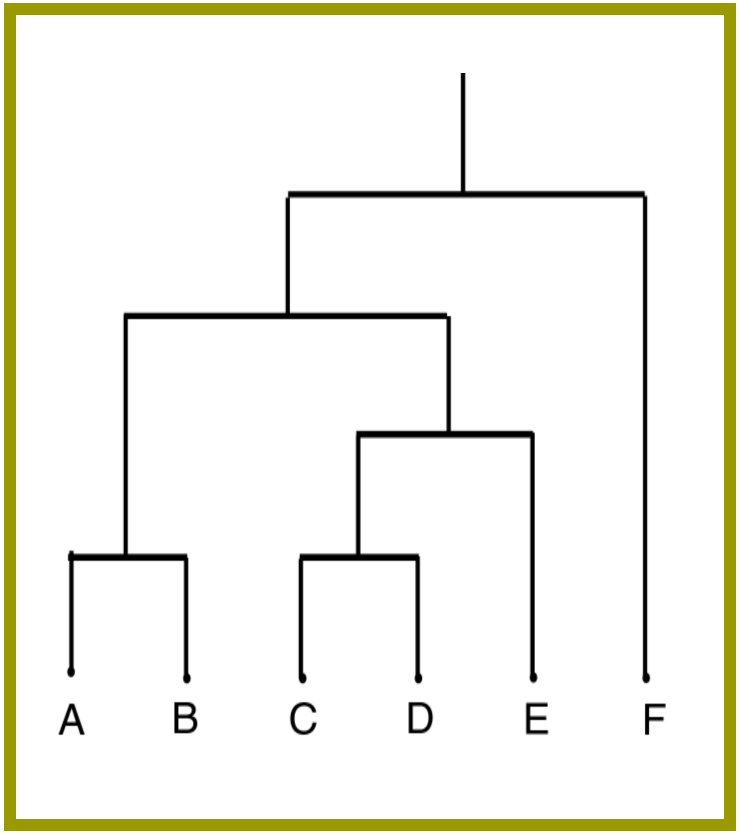
Hierarchical and Density Based Clustering

Hierarchical Clustering

- Clusters are created iteratively, using clusters created in previous step
- Construction of a hierarchy of clusters (*dendrogram*) merging clusters with minimum distance
- Use distance matrix as clustering criteria.
- The Hierarchical method works by grouping data objects(records) into a tree of clusters.
- Classified Further as
  - Agglomerative Hierarchical Clustering
  - Divisive Hierarchical Clustering

Dendrogram

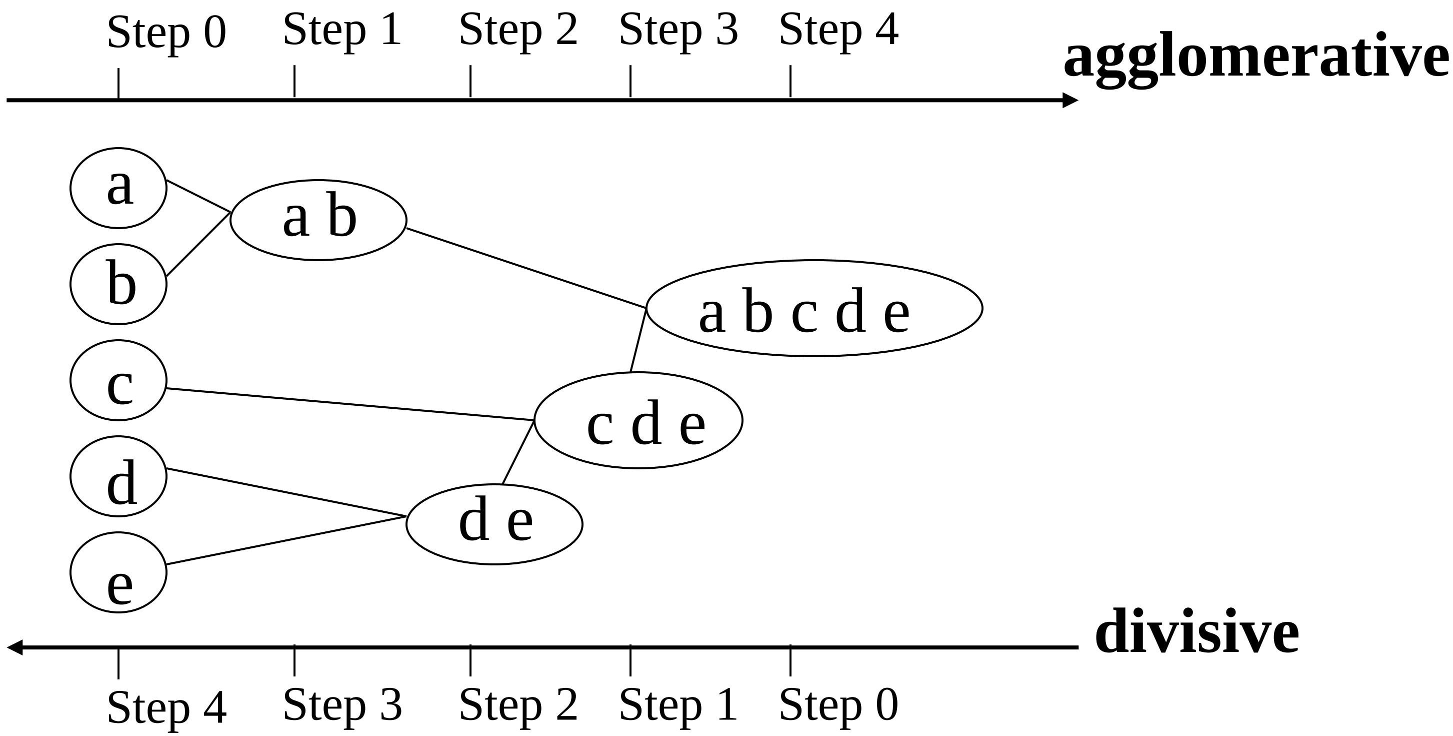
- ***Dendrogram:*** a tree data structure which illustrates hierarchical clustering techniques.
- Each level shows clusters for that level.
  - Leaf – individual clusters
  - Root – one cluster
- A cluster at level i is the union of its children clusters at level i+1.



Hierarchical Clustering

- Clusters are created in levels actually creating sets of clusters at each level.
- *Agglomerative*
  - Initially each item in its own cluster
  - Iteratively clusters are merged together
  - Bottom Up
- *Divisive*
  - Initially all items in one cluster
  - Large clusters are successively divided
  - Top Down

Hierarchical Clustering



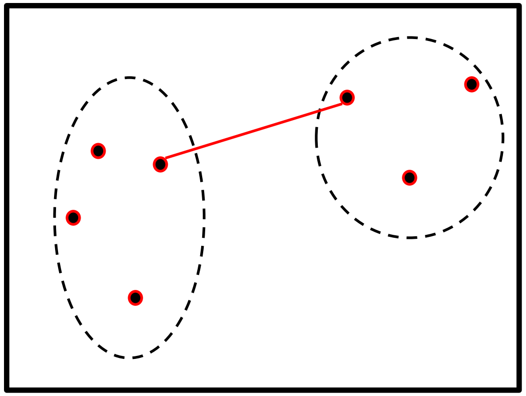
Distance Between Clusters

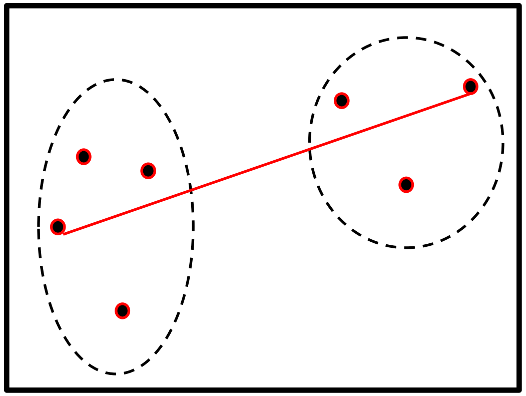
- *Single Link*: smallest distance between points

$$d(i, j) = \min_{x \in C_i, y \in C_j} \{ d(x, y) \}$$

- *Complete Link:* largest distance between points

$$d(i, j) = \max_{x \in C_i, y \in C_j} \{ d(x, y) \}$$



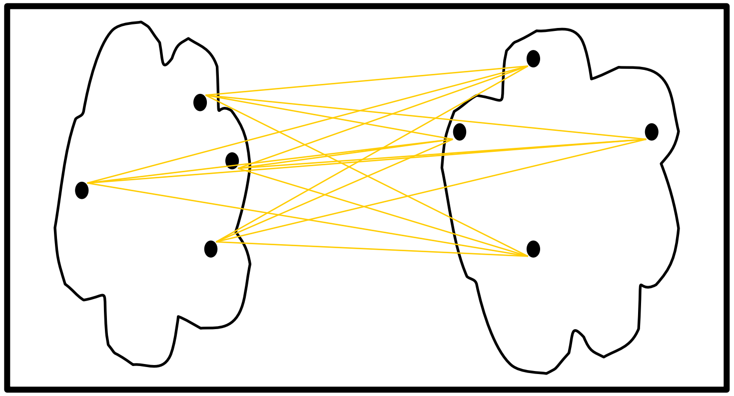


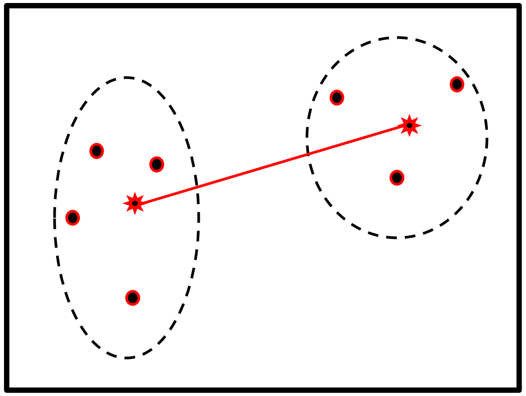
Distance Between Clusters

- *Average Link:* average distance between points

$$d(i, j) = avg_{x \in C_i, y \in C_j} \{ d(x, y) \}$$

- *Centroid:* distance between centroids





Agglomerative Algorithm

- The agglomerative method is basically a bottom-up approach which involves the following steps. An implementation however may include some variation of these steps
  1. Allocate each point to a cluster of its own. Thus we start with n clusters for n objects.
  2. Create a distance matrix by computing distances between all pairs of clusters either using, for example, the single-link metric or the complete-link metric. Some other metric may also be used. Sort these distances in ascending order.
  3. Find the two clusters that have the smallest distance between them
  4. Remove the pair of objects and merge them.
  5. If there is only one cluster left then stop.
  6. Compute all distances from the new cluster and update the distance matrix after the merger and go to Step 3.

Agglomerative Algorithm

- Allocate each point to a cluster and compute the the distance matrix
- Show the half portion of the matrix

Agglomerative Algorithm

- Consider the following data.

Student	Age	Marks1	Marks2	Marks 3
S_1	18	73	75	57
S_2	18	79	85	75
S_3	23	70	70	52
S_4	20	55	55	55
S_5	22	85	86	87
S_6	19	91	90	89
S_7	20	70	65	60
S_8	21	53	56	59
S_9	19	82	82	60
S_{10}	47	75	76	77

Agglomerative Example

	S_1	S_2	S_3	S_4	S_5	S_6	S_7	S_8	S_9	S_{10}
S_1	0									
S_2	34	0								
S_3	18	52	0							
S_4	42	76	36	0						
S_5	57	23	67	95	0					
S_6	66	32	82	106	15	0				
S_7	18	46	16	30	65	76	0			
S_8	44	74	40	8	91	104	28	0		
S_9	20	22	36	60	37	46	30	58	0	
S_{10}	52	44	60	90	55	70	60	86	58	0

Agglomerative Algorithm

- The smallest distance is 8 between S_4 and S_8.
- Combine them as cluster C1 and update the table by putting the C1 into the place where S_4 was.
- All distance except those with cluster C1 remain unchanged.

Agglomerative Algorithm

Student	Age	Marks1	Marks2	Marks3
S_1	18	73	75	57
S_2	18	79	85	75
S_3	23	70	70	52
C_1	20.5	54	55.5	57
S_5	22	85	86	87
S_6	19	91	90	89
S_7	20	70	65	60
S_9	19	82	82	60
S_{10}	47	75	76	77

Agglomerative Algorithm

	S_1	S_2	S_3	C_1	S_5	S_6	S_7	S_9	S_{10}
S_1	0								
S_2	34	0							
S_3	18	52	0						
C_1	41	75	38	0					
S_5	57	23	67	93	0				
S_6	66	32	82	105	15	0			
S_7	18	46	16	29	65	76	0		
S_9	20	22	36	59	37	46	30	0	
S_{10}	52	44	60	88	55	70	60	58	0

Agglomerative Example

- The smallest distance is now 15 between S_5 and S_6.
- Combine and update the table.

Agglomerative Algorithm

Student	Age	Marks1	Marks2	Marks3
S_1	18	73	75	57
S_2	18	79	85	75
S_3	23	70	70	52
C_1	20.5	54	55.5	57
C_2	20.5	88	88	88
S_7	20	70	65	60
S_9	19	82	82	60
S_{10}	47	75	76	77

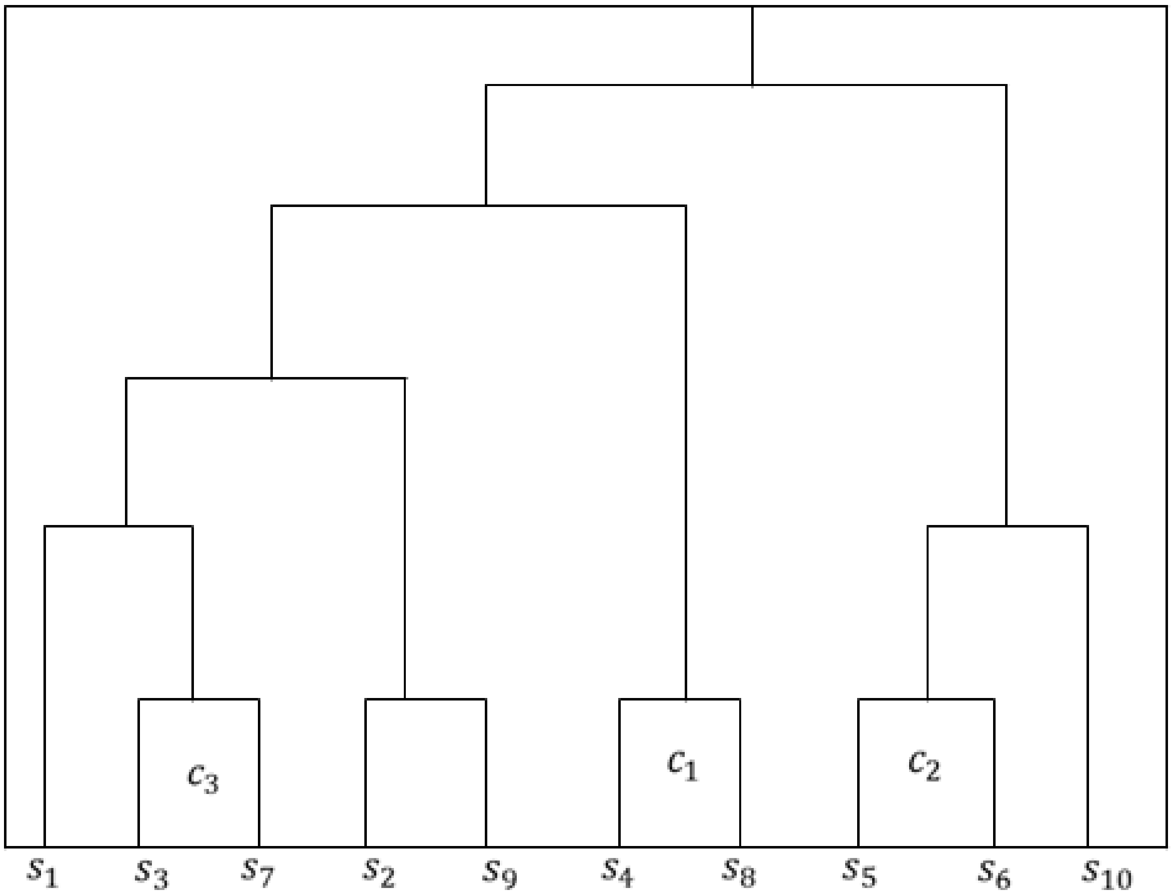
Agglomerative Algorithm

	S_1	S_2	S_3	C_1	C_2	S_7	S_9	S_{10}
S_1	0							
S_2	34	0						
S_3	18	52	0					
C_1	41	75	38	0				
C_2	61.5	27.5	74.5	97.5	0			
S_7	18	46	16	29	69.5	0		
S_9	20	22	36	59	41.5	30	0	
S_{10}	52	44	60	88	62.5	60	58	0

Agglomerative Example

- Merge S_3 and S_7 and put them as C_3.
- Continue the process.

Agglomerative Example



Agglomerative Example

- Let's now see a simple example: a hierarchical clustering of distances in kilometers between some Italian cities. The method used is single-linkage.
- *Single Link*: smallest distance between points
- **Input distance matrix** (L = 0 for all the clusters):

	BA	FI	MI	NA	RM	TO
BA	0	662	877	255	412	996
FI	662	0	295	468	268	400
MI	877	295	0	754	564	138
NA	255	468	754	0	219	869
RM	412	268	564	219	0	669
TO	996	400	138	869	669	0

Agglomerative Algorithm

- The nearest pair of cities is MI and TO, at distance 138.

	BA	FI	MI	NA	RM	TO
BA	0					
FI	662	0				
MI	877	295	0			
NA	255	468	754	0		
RM	412	268	564	219	0	
TO	996	400	138	869	669	0

Agglomerative Algorithm

- The level of the new cluster is L(MI/TO) = 138 and the new sequence number is m = 1.
- The distance from the compound object to another object is equal to the shortest distance from any member of the cluster to the outside object.

Agglomerative Algorithm

- Dist(MI/TO, BA)=min{Dist(MI,BA), Dist(TO,BA)}
- Min{877, 996}
- 877

877

	BA	FI	MI	NA	RM	TO
BA	0					
FI	662	0				
MI	877	295	0			
NA	255	468	754	0		
RM	412	268	564	219	0	
TO	996	400	138	869	669	0

Agglomerative Algorithm

- Dist(MI/TO, FI)=min{Dist(MI,FI), Dist(TO,FI)}
- Min{295, 400}
- 295

295

	BA	FI	MI	NA	RM	TO
BA	0					
FI	662	0				
MI	877	295	0			
NA	255	468	754	0		
RM	412	268	564	219	0	
TO	996	400	138	869	669	0

Agglomerative Algorithm

- Dist(MI/TO, NA)=min{Dist(MI,NA), Dist(TO,NA)}
- Min{754, 869}
- 754

754

	BA	FI	MI	NA	RM	TO
BA	0					
FI	662	0				
MI	877	295	0			
NA	255	468	754	0		
RM	412	268	564	219	0	
TO	996	400	138	869	669	0

Agglomerative Algorithm

- Dist(MI/TO, RM)=min{Dist(MI,RM), Dist(TO,RM)}
- Min{564, 669}
- 564

564

	BA	FI	MI	NA	RM	TO
BA	0					
FI	662	0				
MI	877	295	0			
NA	255	468	754	0		
RM	412	268	564	219	0	
TO	996	400	138	869	669	0

Agglomerative Example

- After merging MI with TO we obtain the following matrix:

	BA	FI	MI/TO	NA	RM
BA	0				
FI	662	0			
MI/TO	877	295	0		
NA	255	468	754	0	
RM	412	268	564	219	0

Agglomerative Example

- min d(i,j) = d(NA,RM) = 219
- merge NA and RM into a new cluster called NA/RM, L(NA/RM) = 219
- m = 2

	BA	FI	MI/TO	NA	RM
BA	0				
FI	662	0			
MI/TO	877	295	0		
NA	255	468	754	0	
RM	412	268	564	219	0

Agglomerative Example

- After merging NA with RM we obtain the following matrix:

	BA	FI	MI/TO	NA/RM
BA	0			
FI	662	0		
MI/TO	877	295	0	
NA/RM	255	268	564	0

Agglomerative Example

- min d(i,j) = d(BA,NA/RM) = 255
- merge BA and NA/RM into a new cluster called BA/NA/RM
- L(BA/NA/RM) = 255
- m = 3

	BA	FI	MI/TO	NA/RM
BA	0			
FI	662	0		
MI/TO	877	295	0	
NA/RM	255	268	564	0

Agglomerative Example

- After merging BA with NA/RM we obtain the following matrix:

	BA/NA/RM	FI	MI/TO
BA/NA/RM	0		
FI	268	0	
MI/TO	564	295	0

Agglomerative Example

- min d(i,j) = d(BA/NA/RM,FI) = 268
- merge BA/NA/RM and FI into a new cluster called BA/FI/NA/RM
- L(BA/FI/NA/RM) = 268
- m = 4

	BA/NA/RM	FI	MI/TO
BA/NA/RM	0		
FI	268	0	
MI/TO	564	295	0

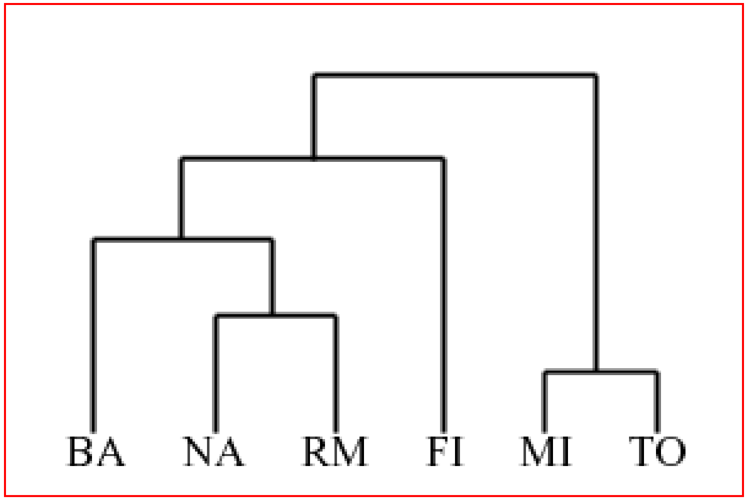
Agglomerative Example

- After merging FI with BA/NA/RM we obtain the following matrix:

	BA/FI/NA/RM	MI/TO
BA/FI/NA/RM	0	295
MI/TO	295	0

Agglomerative Example

- Finally, we merge the last two clusters at level 295.
- The process is summarized by the following hierarchical tree:



Agglomerative Example

- Advantages:
  - Is simple and outputs a hierarchy, a structure that is more informative
  - It does not require us to pre-specify the number of clusters
- Disadvantages:
  - Selection of merge or split points is critical as once a group of objects is merged or split, it will operate on the newly generated clusters and will not undo what was done previously.
  - Thus merge or split decisions if not well chosen may lead to low-quality clusters

Divisive Hierarchical Clustering

- A typical polythetic divisive method works like the following
  1. Decide on a method of measuring the distance between two objects. Also decide a threshold distance.
  2. Create a distance matrix by computing distances between all pairs of object within the cluster. Sort these distances in ascending order.
  3. Find the two objects that have the largest distance between them. They are the most dissimilar objects.
  4. If the distance between the two objects is smaller than the pre-specified threshold and there is no other cluster that needs to be divided then stop, otherwise continue.
  5. Use the pair of objects as seeds of a K-means method to create two new clusters.
  6. If there is only one object in each cluster then stop otherwise continue with step 2.

Divisive Hierarchical Clustering

- Consider the following data

Student	Age	Marks1	Marks2	Marks 3
S_1	18	73	75	57
S_2	18	79	85	75
S_3	23	70	70	52
S_4	20	55	55	55
S_5	22	85	86	87
S_6	19	91	90	89
S_7	20	70	65	60
S_8	21	53	56	59
S_9	19	82	82	60
S_{10}	47	75	76	77

Divisive Hierarchical Clustering

	S_1	S_2	S_3	S_4	S_5	S_6	S_7	S_8	S_9	S_{10}
S_1	0									
S_2	34	0								
S_3	18	52	0							
S_4	42	76	36	0						
S_5	57	23	67	95	0					
S_6	66	32	82	106	15	0				
S_7	18	46	16	30	65	76	0			
S_8	44	74	40	8	91	104	28	0		
S_9	20	22	36	60	37	46	30	58	0	
S_{10}	52	44	60	90	55	70	60	86	58	0

Divisive Hierarchical Clustering

- The largest distance is between S_4 and S_6
- Use the as new two seed
- Use k-mean method two find new clusters

	S_1	S_2	S_3	S_4	S_5	S_6	S_7	S_8	S_9	S_{10}
S_4	42	76	36	0	95	106	30	8	60	90
S_6	66	32	82	106	15	0	76	104	46	70

Cluster membership

Cluster-1 (S_4):

Cluster-2 (S_6):

Divisive Hierarchical Clustering

- Use k-mean method two find new clusters
- Dist(S_4,S_1)=42 and Dist(S_6,S_1)=66
- Minimum=42
- S_1 belongs to Cluster 2.

	S_1	S_2	S_3	S_4	S_5	S_6	S_7	S_8	S_9	S_{10}
S_4	42	76	36	0	95	106	30	8	60	90
S_6	66	32	82	106	15	0	76	104	46	70

Cluster membership

Cluster-1 (S_4): S_1

Cluster-2 (S_6):

Divisive Hierarchical Clustering

- Use k-mean method two find new clusters
- Dist(S_4,S_2)=76 and Dist(S_6,S_2)=32
- Minimum=32
- S_2 belongs to Cluster 2.

	S_1	S_2	S_3	S_4	S_5	S_6	S_7	S_8	S_9	S_{10}
S_4	42	76	36	0	95	106	30	8	60	90
S_6	66	32	82	106	15	0	76	104	46	70

Cluster membership

Cluster-1 (S_4): S_1

Cluster-2 (S_6): S_2

Divisive Hierarchical Clustering

- Use k-mean method two find new clusters
- Dist(S_4,S_3)=36 and Dist(S_6,S_3)=82
- Minimum=36
- S_3 belongs to Cluster 1.

	S_1	S_2	S_3	S_4	S_5	S_6	S_7	S_8	S_9	S_{10}
S_4	42	76	36	0	95	106	30	8	60	90
S_6	66	32	82	106	15	0	76	104	46	70

Cluster membership

Cluster-1 (S_4): $S_{1,}$ S_3

Cluster-2 (S_6): S_2

Divisive Hierarchical Clustering

- Finally we get the following:

	S_1	S_2	S_3	S_4	S_5	S_6	S_7	S_8	S_9	S_{10}
S_4	42	76	36	0	95	106	30	8	60	90
S_6	66	32	82	106	15	0	76	104	46	70

Cluster membership

Cluster-1 (S_4): S_1, S_3, S_4 , S_7, S_8

Cluster-2 (S_6): S_2, S_5, S_6 , S_9, S_{10}

Divisive Hierarchical Clustering

- None of the stopping criteria have been met
- Split table by two seed naming $S_{1,}$ and S_8
- Take the larger cluster and continue the process.

	S_1	S_3	S_4	S_7	S_8
S_1	0				
S_3	18	0			
S_4	42	36	0		
S_7	18	16	30	0	
S_8	44	40	8	28	0

Divisive Hierarchical Clustering

- Table below shows the member of cluster 2.
- Take the larger cluster and continue the process.

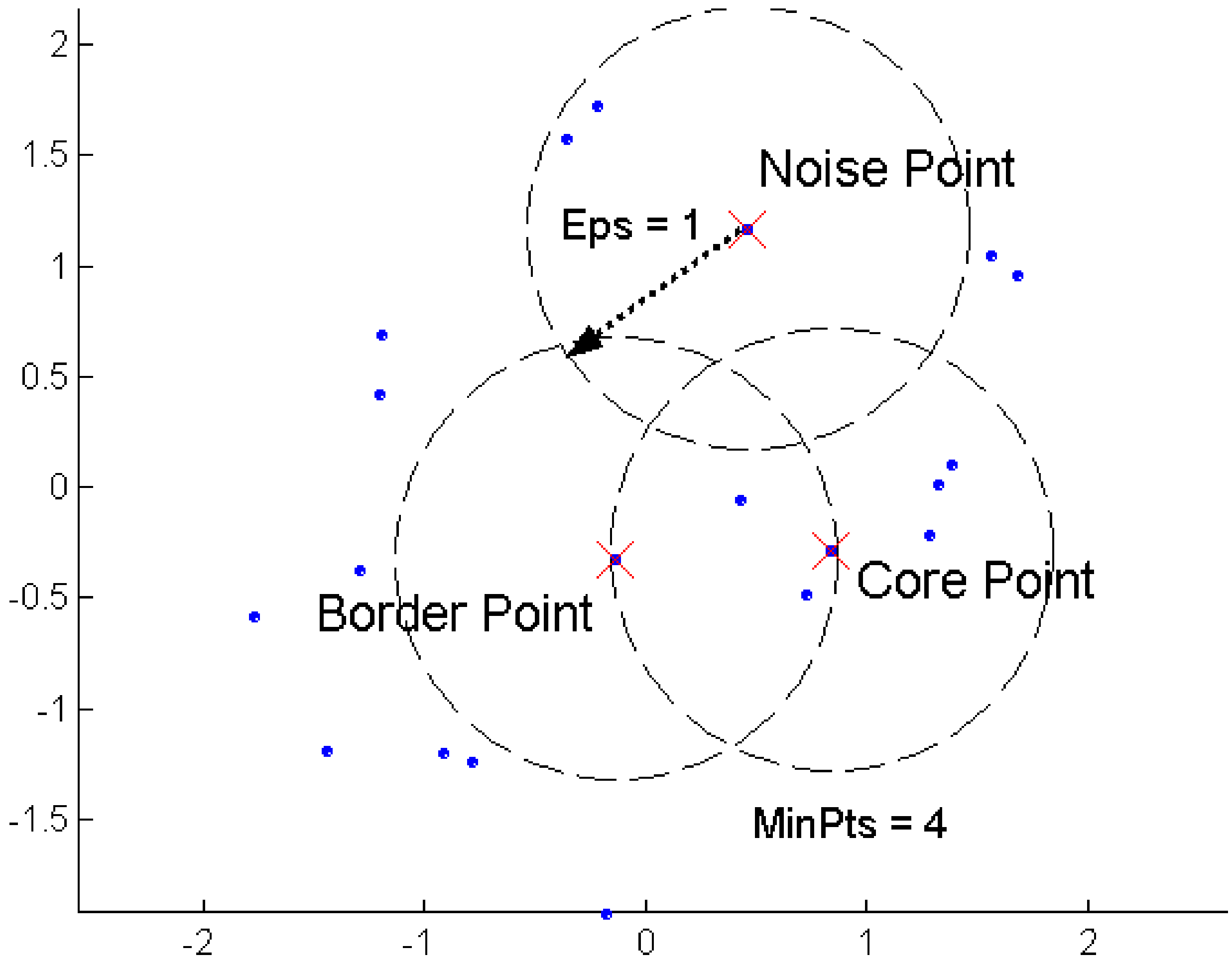
	S_2	S_5	S_6	S_9	S_{10}
S_2	0				
S_5	23	0			
S_6	32	15	0		
S_9	22	37	46	0	
S_{10}	44	55	70	58	0

Density Based (DBSCAN)

DBSCAN Algorithm

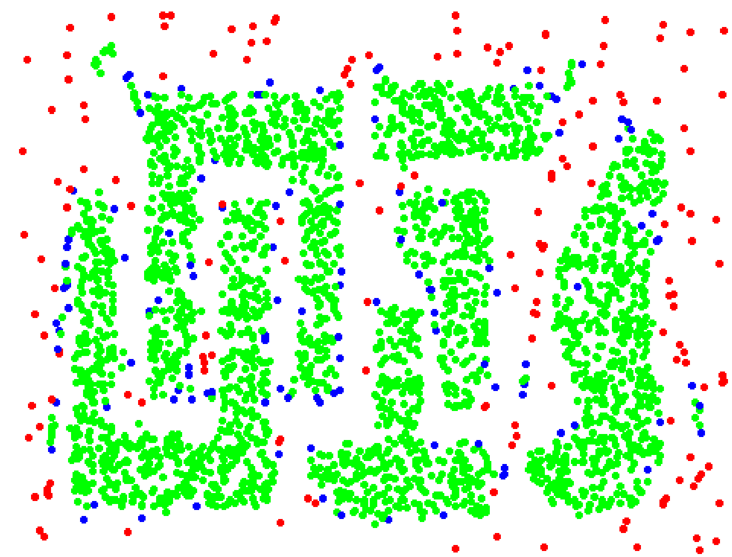
- Density: number points within specified radius.
  - The parameter epsilon (Eps) defines the radius of neighborhood around a point x
  - The parameter MinPts is the minimum number of neighbors within "Eps" radius.
- Three types of points:
  - Core point: has more than *MinPts* points within *Eps*
    - These points are in interior of cluster
  - Border point: has fewer than *MinPts* within Eps but is within *Eps* of a core point
    - These are on the boundary of the cluster
  - Noise point: neither a core or border point
    - Not within any cluster

Core, Border, and Noise Points



Core, Border, and Noise Points

Original Points

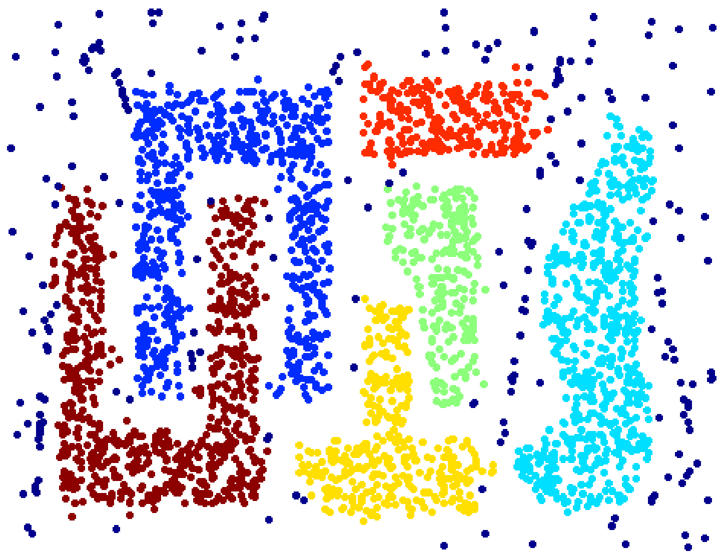


Core (green), border (blue) and noise (red)

Eps = 10, MinPts = 4

When DBSCAN Works Well

Original Points



Clusters

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

References

Gupta, G. K. Introduction to data mining with case studies. PHI Learning Pvt. Ltd., 2006.

Dunham, Margaret H. *Data mining: Introductory and advanced topics*. Pearson Education India, 2006.

Han, Jiawei, Micheline Kamber, and Jian Pei. *Data mining: concepts and techniques*. Morgan kaufmann, 2006.

Thank you