Data Mining and Data Warehousing 11

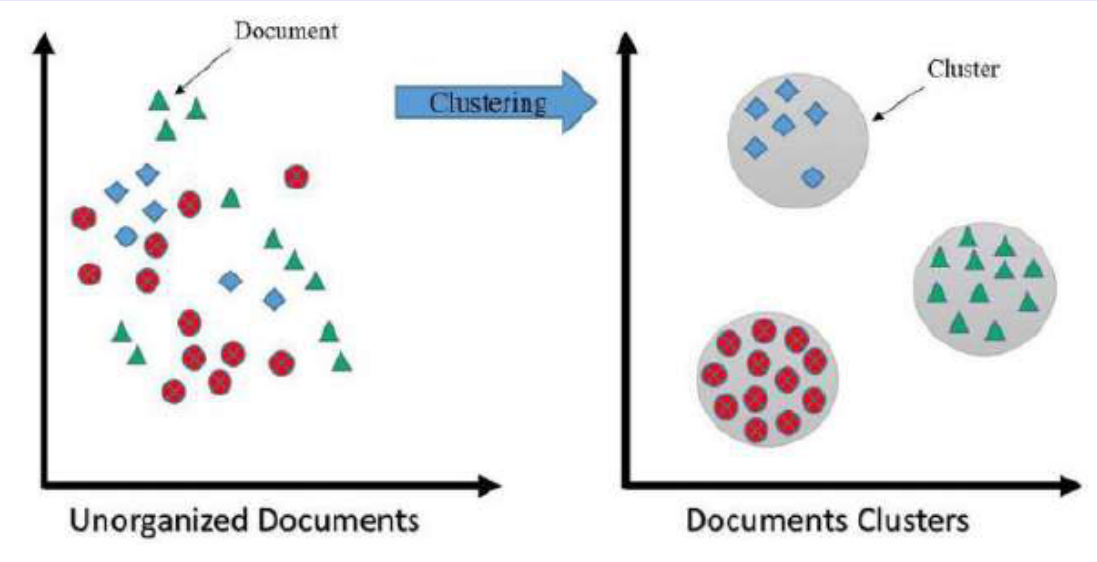
Clustering

Chittaranjan Pradhan
School of Computer Engineering,
KIIT University

Cluster Analysis

Cluster Analysis

- Classification discovers patterns in the data and these patterns are then utilized to predict the values of a class attribute of future data instances
- *Clustering is a process of finding groups of data points such that the data points in group will be similar to one another and different from data points in other groups*
- Cluster analysis is the process of finding similarities between data according to the characteristics found in the data and grouping similar data objects into clusters



Cluster Analysis...

Cluster Analysis...

- A good clustering will produce high quality clusters with high intra-cluster similarity and low inter-cluster similarity
- The quality of clustering depends upon the similarity measure used by the method as well as its ability to discover the hidden patterns
- Clustering is an example of unsupervised learning as it doesn't rely on predefined classes
- It is used as standalone tool to get insight into data distribution such as visualization of cluster preprocessing for algorithms
- **Applications**
  - Pattern recognition, Data analysis, Image processing
  - Outlier detection, Credit card fraud
  - Business intelligence

Cluster Analysis...

Types of Data in Cluster Analysis

- **Nominal or Categorical Attributes**: attribute that can take more than two categories that don't have a natural order. Ex: hair color
- **Binary Attributes**: is a nominal attribute with only two states. Ex: medical test
- **Ordinal Attributes**: has two or more values, but these values have a meaningful order. Ex: Size (like large, medium, small)
- **Interval-scaled Attributes**: there is order and the difference between the values are equally spaced. An interval attribute doesn't have a true zero-point. Ex: temperature in farenhit
- **Ratio-scaled Attributes**: has all the properties of an interval attribute, and also has an inherent zero-point. Ex: temprature in Kelvin
- **Attributes of Mixed Type**: combination of other types

Categories of Clustering Algorithm

Categories of Clustering Algorithm

- **Partitioning Method**: Given a database of n data tuples, a partitioning method constructs k partitions of the data, where each partition represents a cluster and $k < n$. Ex: K-means, K-medoids
- **Hierarchical Method**: creates a hierarchical decomposition of the given set of data objects
  - **agglomerative (or bottom-up) approach**: starts with each object forming a separate group. It successively merges the objects or groups that are close to one another, until all of the groups are merged into one
  - **divisive (or top-down) approach**: starts with all of the objects in the same cluster. In each successive iteration, a cluster is split up into smaller clusters until a termination condition holds
- **Density-based Method**: the basic idea is to continue growing the given cluster as long as the density in the neighborhood exceeds some threshold. Ex: DBSCAN
- **Model-based Method**: A model is hypothesized for each cluster to find the best fit of data for a given model

K-Means Algorithm

- K-means clustering is a type of unsupervised learning, which is used when we have unlabeled data. It aims to partition n observation into k clusters
- To perform K-means clustering, we must first specify the desired number of clusters K; then the K-means algorithm will assign each observation to exactly one of the K clusters
- Each cluster in the K-means clustering algorithm is represented by a centroid point (i.e. average of all the points in the set)
- The idea of the K-means algorithm is to find k-centroid points and every point in the dataset will belong either of K-sets having minimum Euclidean distance

K-Means Algorithm...

Algorithm

1. Specify number of clusters K
2. Randomly select 'K' points as the initial cluster center
3. Calculate the distance between each data point and cluster centers
4. Assign the data points to the cluster center whose distance from the cluster center is minimum of all the cluster centers
5. Recalculate the new centers by taking the average of the all data points that belong to each cluster
6. Recalculate the distance between each data point and new obtained cluster centers
7. Repeat steps 4, 5 and 6 until the same points are assigned to each cluster in consecutive rounds

K-Means Algorithm...

Q. Cluster the following eight points into three clusters using K-Means algorithm: (2, 10), (2,5), (8,4), (5,8), (7,5), (6,4), (1,2) and (4,9)

No. of clusters $K = 3$

Let initial cluster centers are: $C1 = (2, 10), C2 = (5, 8)$ and $C3 = (1, 2)$. Suppose considering the *Euclidean distance* as the distance measure: $D = \sqrt{(x_2 - x_1)^2 + (y_2 - y_1)^2}$

Iteration 1:

Calculating distance between cluster centers and each data points:

	Distance to			Point Belongs to Cluster
Point	$C1(2, 10)$	$C2(5, 8)$	$C3(1, 2)$	
(2, 10)	0	3.60	8.06	C1
(2, 5)	5	4.24	3.16	C3
(8, 4)	8.48	5	7.28	C2
(5, 8)	3.61	0	7.21	C2
(7, 5)	7.07	3.60	6.71	C2
(6, 4)	7.21	4.12	5.38	C2
(1, 2)	8.06	7.21	0	C3
(4, 9)	2.24	1.41	7.61	C2

Thus after first iteration:

$$Clusters\ 1 = \{(2, 10)\},$$

$$Cluster\ 2 = \{(8, 4), (5, 8), (7, 5), (6, 4), (4, 9)\},$$

$$Cluster\ 3 = \{(2, 5), (1, 2)\}$$

Now, new cluster centers are:

$$C1 = (2, 10)$$

$$C2 = ((8 + 5 + 7 + 6 + 4)/5, (4 + 8 + 5 + 4 + 9)/5) = (6, 6)$$

$$C3 = ((2 + 1)/2, (5 + 2)/2) = (1.5, 3.5)$$

K-Means Algorithm...

Iteration 2:

Calculating distance between new cluster centers and each data points:

Point	Distance to $C1(2, 10)$	$C2(6, 6)$	$C3(1.5, 3.5)$	Point Belongs to Cluster
(2, 10)	0	5.66	6.52	C1
(2, 5)	5	4.12	1.58	C3
(8, 4)	8.48	2.83	6.52	C2
(5, 8)	3.61	2.24	5.70	C2
(7, 5)	7.07	1.41	5.70	C2
(6, 4)	7.21	2	4.53	C2
(1, 2)	8.06	6.40	1.58	C3
(4, 9)	2.24	3.60	6.04	C1

Thus after second iteration:

$$Clusters\ 1 = \{(2, 10), (4, 9)\},$$
$$Cluster\ 2 = \{(8, 4), (5, 8), (7, 5), (6, 4)\},$$
$$Cluster\ 3 = \{(2, 5), (1, 2)\}$$

Now, new cluster centers are:

$$C1 = ((2 + 4)/2, (10 + 9)/2) = (3, 9.5)$$
$$C2 = ((8 + 5 + 7 + 6)/4, (4 + 8 + 5 + 4)/4) = (6.5, 5.25)$$
$$C3 = ((2 + 1)/2, (5 + 2)/2) = (1.5, 3.5)$$

K-Means Algorithm...

Iteration 3:

Calculating distance between new cluster centers and each data points:

Point	Distance to $C1(3, 9.5)$	Distance to $C2(6.5, 5.25)$	Distance to $C3(1.5, 3.5)$	Point Belongs to Cluster
(2, 10)	1.12	6.54	6.52	C1
(2, 5)	4.61	4.51	1.58	C3
(8, 4)	7.43	1.95	6.52	C2
(5, 8)	2.5	3.13	5.70	C1
(7, 5)	6.02	0.56	5.70	C2
(6, 4)	6.26	1.32	4.53	C2
(1, 2)	7.76	6.39	1.58	C3
(4, 9)	1.12	4.51	6.04	C1

Thus after third iteration:

$$Clusters\ 1 = \{(2, 10), (5, 8), (4, 9)\},$$
$$Cluster\ 2 = \{(8, 4), (7, 5), (6, 4)\},$$
$$Cluster\ 3 = \{(2, 5), (1, 2)\}$$

Now, new cluster centers are:

$$C1 = ((2 + 5 + 4)/3, (10 + 8 + 9)/3) = (3.67, 9)$$
$$C2 = ((8 + 7 + 6)/3, (4 + 5 + 4)/3) = (7, 4.3)$$
$$C3 = ((2 + 1)/2, (5 + 2)/2) = (1.5, 3.5)$$

K-Means Algorithm...

Iteration 4:

Calculating distance between new cluster centers and each data points:

	Distance to			Point Belongs to Cluster
Point	$C1(3.67, 9)$	$C2(7, 4.3)$	$C3(1.5, 3.5)$	
(2, 10)	1.95	7.58	6.52	C1
(2, 5)	4.33	5.05	1.58	C3
(8, 4)	6.61	1.04	6.52	C2
(5, 8)	1.66	4.20	5.70	C1
(7, 5)	5.20	0.7	5.70	C2
(6, 4)	5.52	1.04	4.53	C2
(1, 2)	7.49	6.42	1.58	C3
(4, 9)	0.33	5.57	6.04	C1

Thus after fourth iteration:

$$Clusters\ 1 = \{(2,10), (5,8), (4,9)\},$$
$$Cluster\ 2 = \{(8,4), (7,5), (6,4)\},$$
$$Cluster\ 3 = \{(2,5), (1,2)\}$$

The cluster of the data points obtained in fourth iteration is same as third iteration. It means that none of the data points has moved to other cluster. So this becomes the stopping condition for our algorithm.

K-Means Algorithm...

Advantages

- Fast, robust and easier to understand
- Gives best result when data set are dintinct or well separated from each other

Disadvantages

- Require number of clusters in advance
- Sensitive to noise and outlier data points
- Randomly choosing of the cluster centers can't lead us to fruitful result
- Applicable when mean is defined
- Algorithm fails for non-linear data

K-Medoids Algorithm

K-Medoids (Partitioning Around Medoid (PAM)) Algorithm

- This algorithm represents a cluster by medoid
- Medoid is the most centrally located object of the cluster, with minimum sum of distances to other points
- K-means algorithm selects the average of a cluster's points as its center whereas the K-medoid algorithm always picks the actual data points from the clusters as their centers
- K-medoids method is more robust tha K-means in the presence of noise and outliers as medoid is less influenced by outliers or noise
- Its processing is more costly than K-means method

K-Medoids Algorithm...

Algorithm

1. Randomly choose K data points as the initial medoids
2. Associate each remaining data points to the closest medoid by using any common distance measure
3. While the cost decreases:
   For each medoid m, for each non-medoid data point o:
   - Swap m and o, associate each data point to the closest medoid, recompute the cost
   - If the total cost is more than that in the previous step, undo the swap

For simplicity, we take Manhattan distance for cost measure:
$Cost(c) = \sum_{C_i} \sum_{P_i \in C_i} |P_i - C_i|$

K-Medoids Algorithm...

Q. Cluster the following data set into two clusters using K-Medoids algorithm: (5,6), (4,5), (4,6), (6,7) and (7,8)

No. of clusters K = 2

Let $C_1 = (4,6)$ and $C_2 = (6,7)$ are two initial mediods. Suppose considering the Manhattan distance metric as the distance measure: $d = |x_2 - x_1| + |y_2 - y_1|$

Iteration 1:

Calculating distance between mediods and each non-mediods points:

Points	Distance to		Cluster Assignment
	$C_1(4,6)$	$C_2(6,7)$	
(5,6)	1	2	C1
(4,5)	1	4	C1
(4,6)	-	-	C1
(6,7)	-	-	C2
(7,8)	5	2	C2

Thus after first iteration:

$Cluster\ 1 = \{(5,6), (4,5), (4,6)\}$

$Cluster\ 2 = \{(6,7), (7,8)\}$

$Total\ cost = (1+1) + (2) = 4$

K-Medoids Algorithm...

Iteration 2:

Now let us choose some other point to be a mediod instead of (4, 6).
Let us randomly choose (4, 5).
Now the new mediod set is: $C_1 = (4, 5)$ and $C_2 = (6, 7)$

Calculating distance between new mediods and each non-mediod points:

Points	Distance to		Cluster Assignment
	$C_1(4,5)$	$C_2(6,7)$	
(5,6)	2	2	C1
(4,5)	-	-	C1
(4,6)	1	3	C1
(6,7)	-	-	C2
(7,8)	6	2	C2

Thus after second iteration:

$$Cluster\ 1 = \{(5,6), (4,5), (4,6)\}$$
$$Cluster\ 2 = \{(6,7), (7,8)\}$$

$$Total\ cost = (2+1) + (2) = 5$$

The cost at this time is 5, which is larger than the previous cost $=4$. So undo the swap. Hence (4, 6) and (6, 7) are the final medoids. Therefore the clusters obtained finally are:

$$Cluster\ 1 = \{(5,6), (4,5), (4,6)\}$$
$$Cluster\ 2 = \{(6,7), (7,8)\}$$

K-Medoids Algorithm...

Q. Cluster the following data set into two clusters using K-Medoids algorithm: (1,3), (4,2), (6,2), (3,5) and (4,1)

No. of clusters K = 2

Let $C_1 = (1,3)$ and $C_2 = (6,2)$ are two initial mediods. Suppose considering the Manhattan distance metric as the distance measure:

$$d = |x_2 - x_1| + |y_2 - y_1|$$

Iteration 1:

Calculating distance between mediods and each non-mediod points:

Points	Distance to		Cluster Assignment
	$C_1(1,3)$	$C_2(6,2)$	
(1,3)	-	-	C1
(4,2)	4	2	C2
(6,2)	-	-	C2
(3,5)	4	6	C1
(4,1)	5	3	C2

Thus after first iteration:

$Cluster\ 1 = \{(1,3), (3,5)\}$

$Cluster\ 2 = \{(4,2), (6,2), (4,1)\}$

$Total\ cost = (4) + (2+3) = 9$

K-Medoids Algorithm...

Iteration 2:

Now let us choose some other point to be a mediod instead of (1, 3).
Let us randomly choose (3, 5).
Now the new mediod set is: $C_1 = (3,5)$ and $C_2 = (6,2)$

Calculating distance between new mediods and each non-mediod points:

Points	Distance to		Cluster Assignment
	$C_1(3,5)$	$C_2(6,2)$	
(1,3)	4	6	C1
(4,2)	4	2	C2
(6,2)	-	-	C2
(3,5)	-	-	C1
(4,1)	5	3	C2

Thus after second iteration:

$$Cluster\ 1 = \{(1,3),(3,5)\}$$
$$Cluster\ 2 = \{(4,2),(6,2),(4,1)\}$$

$$Total\ cost = (4) + (2+3) = 9$$

The cost is same as previous, thus no undo swap.

K-Medoids Algorithm...

Iteration 3:

Again let us choose some other point to be a mediod instead of (6, 2).
Let us randomly choose (4, 2).
Now the new mediod set is: $C_1 = (3, 5)$ and $C_2 = (4, 2)$

Calculating distance between new mediods and each non-mediod points:

Points	Distance to		Cluster Assignment
	$C_1(3,5)$	$C_2(4,2)$	
(1,3)	4	4	C1
(4,2)	-	-	C2
(6,2)	6	2	C2
(3,5)	-	-	C1
(4,1)	5	1	C2

Thus after third iteration:

$Cluster\ 1 = \{(1,3), (3,5)\}$
$Cluster\ 2 = \{(4,2), (6,2), (4,1)\}$

$Total\ cost = (4) + (2 + 1) = 7$

The cost at this time is 7, which is less than the previous cost = 9. So we will iterate again.

K-Medoids Algorithm...

Iteration 4:

Lets swap (4, 2) and (4, 1)
Now the new mediod set is: $C_1 = (3, 5)$ and $C_2 = (4, 1)$.

Calculating distance between new mediods and each non-mediod points:

Points	Distance to		Cluster Assignment
	$C_1(3,5)$	$C_2(4,1)$	
(1,3)	4	5	C1
(4,2)	4	1	C2
(6,2)	6	3	C2
(3,5)	-	-	C1
(4,1)	-	-	C2

Thus after fourth iteration:

$$Cluster\ 1 = \{(1,3), (3,5)\}$$
$$Cluster\ 2 = \{(4,2), (6,2), (4,1)\}$$

$$Total\ cost = (4) + (3 + 1) = 8$$

The cost at this time is 8, which is larger than the previous cost = 7. So undo the swap. Hence (3, 5) and (4, 2) are the final medoids. Therefore the clusters obtained finally are:

$$Cluster\ 1 = \{(1,3), (3,5)\}$$
$$Cluster\ 2 = \{(4,2), (6,2), (4,1)\}$$

K-Medoids Algorithm...

Advantages

- Simple to understand and easy to implement
- Fast and converges in fixed number of steps
- Less sensitive to outliers
- Flexibility in distance metrics

Disadvantages

- Good for small data set, but doesn't scale well for large data sets
- Sensitive to initial medoids
- Non-suitable for clustering random shaped cluster

Application

- Image segmentation
- Test clustering
- Fraud detection

K-Medoids Algorithm...

K-Medoid vs. K-Means

- **Similarities**
  - Both algorithms are iterative
  - Both algorithms require the number of clusters to be specified in advance
  - Both algorithms can be used with Euclidean distance metrics
- **Disimilarities**
  - K-medoids represents clusters using medoids, while K-means represents clusters using centroids
  - K-medoids is less sensitive to outliers than K-means as the medoids are less influenced by individual data points than centroids
  - K-medoids is computationally more expensive than K-means
  - K-medoids can handle non-Euclidean distance metrics, while K-means is restricted to Euclidean distance metrics

K-Medians Algorithm

K-Medians Algorithm

K-median clustering is a partitioning method that aims to divide a dataset into k clusters, where each cluster is represented by its median

Algorithm

1. Randomly select k data points as initial medians
2. Assign each remaining data points to the cluster whose median it is closest to, using Manhattan distance
3. Recalculate the median for each cluster based on the newly assigned points. The median of a set of points is the point where the sum of absolute deviations to all other points in the set is minimized
4. Repeat steps 2 and 3 until the cluster assignments no longer change

K-median clustering is more robust to outliers than k-means clustering because the median is less affected by extreme values than the mean

K-Medians Algorithm...

Q. Cluster the following data set into two clusters using K-Median algorithm: (2, 5), (3, 8), (4, 4), (5, 8) and (6,3)

No. of clusters K = 2
Let initial medians are M1 = (3, 8) and M2 = (6,3)

Iteration 1

	M1 (3, 8)	M2 (6, 3)	Cluster
(2, 5)	4	6	C1
(3, 8)	0	8	C1
(4, 4)	5	3	C2
(5, 8)	2	6	C1
(6, 3)	8	0	C2

Cluster 1: (2, 5), (3, 8), (5, 8)
Cluster 2: (4, 4), (6, 3)

New Medians:
M1 = (3, 8)
M2 = (5, 3.5)

K-Medians Algorithm...

Iteration 2

	M1 (3, 8)	M2 (5, 3.5)	Cluster
(2, 5)	4	4.5	C1
(3, 8)	0	6.5	C1
(4, 4)	5	1.5	C2
(5, 8)	2	4.5	C1
(6, 3)	8	1.5	C2

Cluster 1: (2, 5), (3, 8), (5, 8)
Cluster 2: (4, 4), (6, 3)

The cluster of data points in second iteration is same as first iteration. So, this is the stopping condition for our algorithm

K-Median Algorithm...

Advantages

- Robustness to outliers
- Flexibility with distance metrics
- Handles discrete and ordinal data

Disadvantages

- Slower computation
- This algorithm is sometimes confused with k-medoids, which is another robust clustering method

K-Mode Algorithm

K-Mode Algorithm

K-Modes clustering is an unsupervised technique. It is a partition clustering algorithm used to group a dataset into K clusters. K-Modes clustering can be used to partition data having categorical variables

Algorithm

1. K-Modes clustering is an iterative algorithm that starts by selecting k initial data points as centroids of the cluster
2. Each data point in the dataset is assigned to a cluster based on its similarity with the centroids
3. After creating clusters for the first time, we select a new centroid in each cluster using the mode of each feature in the data
4. After selecting new clusters, we calculate their dissimilarity from each data point and regroup the clusters
5. This process continues until the process converges and there is no change to the clusters in two consecutive iterations

K-Mode Algorithm...

Q. Cluster the following eight data into three clusters using K-Mode algorithm: (1, A), (2, B), (3, B), (1, A), (1, C), (2, C), (3, C), (1, C)

No. of clusters K = 3
Let initial cluster centroids are : (1, A), (3, B), (3, C)

Iteration 1:

	C1 (1, A)	C2 (3, B)	C3 (3, C)	Cluster
(1, A)	0	2	2	C1
(2, B)	2	1	2	C2
(3, B)	2	0	1	C2
(1, A)	0	2	2	C1
(1, C)	1	2	1	C3
(2, C)	2	2	1	C3
(3, C)	2	1	0	C3
(1, C)	1	2	1	C3

Cluster1: (1, A), (1, A)
Cluster2: (2, B), (3, B)
Cluster3: (1, C), (2, C), (3, C), (1, C)

New Cluster centroids:
(1, A), (3, B), (1, C)

K-Mode Algorithm...

Iteration 2:

	C1 (1, A)	C2 (3, B)	C3 (1, C)	Cluster
(1, A)	0	2	1	C1
(2, B)	2	1	2	C2
(3, B)	2	0	2	C2
(1, A)	0	2	1	C1
(1, C)	1	2	0	C3
(2, C)	2	2	1	C3
(3, C)	2	1	1	C3
(1, C)	1	2	0	C3

Cluster 1: (1, A), (1, A)
Cluster2: (2, B), (3, B)
Cluster3: (1, C), (2, C), (3, C), (1, C)

The cluster of data points in second iteration is same as first iteration.
So, this is the stopping condition for our algorithm

K-Mode Algorithm...

Advantages

- It is easy to understand and implement
- It is a scalable algorithm
- It is applicable to a dataset having categorical attributes

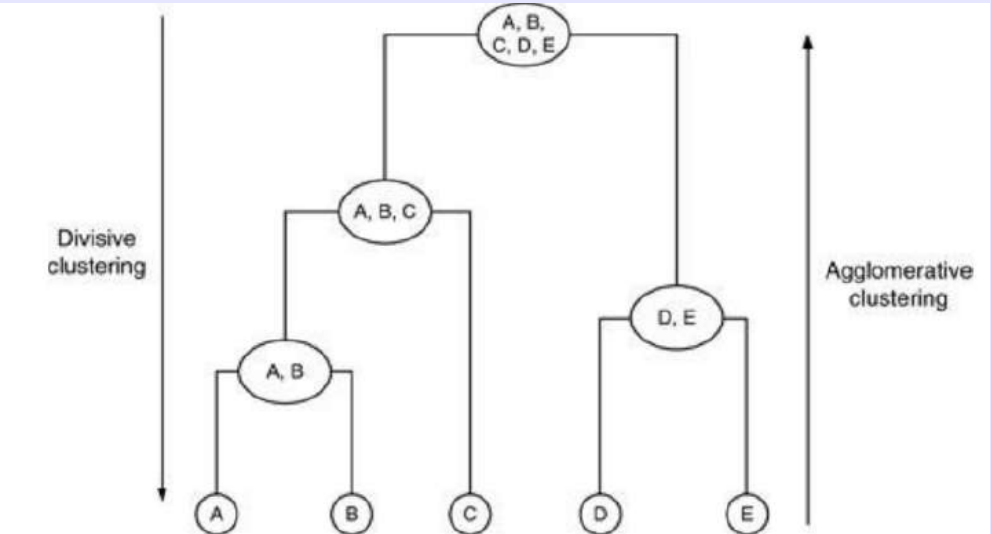
Disadvantages

- Number of clusters required to partition the data is unknown
- Computation required is hugely dependent on the choice of initial centroids
- The final clusters are heavily dependent on the initial centroids

Hierarchical Clustering

Hierarchical Clustering

- Hierarchical clustering is a method of clustering which seeks to build a hierarchy of clusters in a given dataset
- **Agglomerative Approach**: each object creates its own cluster. The single clusters are merged to make larger cluster and the process of merging continues until the single cluster or required number of clusters are formed
- **Divisive Approach**: all objects are arranged within a big single cluster and the large cluster is continuously divided into smaller clusters until each cluster has single object or required numbers of clusters are formed



Hierarchical Clustering...

Hierarchical Clustering...

The merging process can be done using three techniques:

- **Complete-linkage**: the distance between two clusters is defined as the longest distance between two points in each cluster
- **Single-linkage**: the distance between two clusters is defined as the shortest distance between two points in each cluster. This linkage may be used to detect high values in your dataset which may be outliers as they will be merged at the end
- **Average-linkage**: the distance between two clusters is defined as the average distance between each point in one cluster to every point in the other cluster

Hierarchical Clustering...

Dendrogram

- Dendrogram is 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 $i + 1$

Advantage

Any desired number of clusters can be obtained by cutting the tree at proper level

Disadvantage

Difficulty in selecting from where to merge or split a cluster or clusters

Hierarchical Clustering...

Agglomerative Clustering Algorithm

1. Compute the distance matrix between the input data points
2. Let each data points to be a cluster
3. **Repeat**
   - Merge the two clusters
   - Update the distance matrix
4. **Until** only K cluster remains

Divisive Clustering Algorithm

1. Start with all data points in single cluster
2. **Repeat**
   - Choice of the cluster to be split
   - Split of this cluster
3. **Until** only K cluster are created

Agglomerative Clustering

Q. Cluster the data points (1,1), (1.5,1.5), (5,5), (3,4), (4,4) and (3,3.5) into two clusters using single linkage Agglomerative clustering

Assume A = (1, 1), B = (1.5, 1.5), C = (5, 5), D = (3, 4), E = (4, 4), F = (3, 3.5)

The distance matrix is:

	A	B	C	D	E	F
A	0					
B	0.71	0				
C	5.66	4.95	0			
D	3.61	2.92	2.24	0		
E	4.24	3.54	1.41	1.00	0	
F	3.20	2.50	2.50	0.50	1.12	0

In above table, the minimum value is 0.50 which is the distance between D and F. Thus, we group cluster D and F into (D, F). Then we update the distance matrix as below:

	A	B	C	D, F	E
A	0				
B	0.71	0			
C	5.66	4.95	0		
D, F	3.20	2.50	2.24	0	
E	4.24	3.54	1.41	1.00	0

$$d_{(D,F)\mapsto A} = \min\left(d_{DA}, d_{FA}\right) = \min\left(3.61, 3.20\right) = 3.20$$

$$d_{(D,F)\mapsto B} = \min\left(d_{DB}, d_{FB}\right) = \min\left(2.92, 2.50\right) = 2.50$$

$$d_{(D,F)\mapsto C} = \min\left(d_{DC}, d_{FC}\right) = \min\left(2.24, 2.50\right) = 2.24$$

$$d_{E\mapsto(D,F)} = \min\left(d_{ED}, d_{EF}\right) = \min\left(1.00, 1.12\right) = 1.00$$

In updated distance matrix, the minimum value is 0.71 which is distance between cluster B and cluster A. Thus, we group cluster A and cluster B into a single cluster name (A, B). Now we update distance matrix as

Agglomerative Clustering...

Agglomerative Clustering...

	A, B	C	D, F	E
A, B	0			
C	4.95	0		
D, F	2.50	2.24	0	
E	3.54	1.41	1.00	0

$$d_{C\to(A,B)} = \min\left(d_{CA}, d_{CB}\right) = \min\left(5.66, 4.95\right) = 4.95$$

$$d_{(D,F)\to(A,B)} = \min\left(d_{DA}, d_{DB}, d_{FA}, d_{FB}\right) = \min\left(3.61, 2.92, 3.20, 2.50\right) = 2.50$$

$$d_{E\to(A,B)} = \min\left(d_{EA}, d_{EB}\right) = \min\left(4.24, 3.54\right) = 3.54$$

In updated distance matrix, the minimum value is 1.00 which is distance between cluster E and cluster (D, F). Thus, we group cluster E and cluster (D, F) into a single cluster ((D, F), E). The updated distance matrix is given below:

	A, B	C	(D, F), E
A, B	0		
C	4.95	0	
(D, F), E	2.50	1.41	0

$$d_{((D,F),E)\to(A,B)} = \min\left(d_{DA}, d_{DB}, d_{FA}, d_{FB}, d_{EA}, d_{EB}\right) = \min\left(3.61, 2.92, 3.20, 2.50, 4.24, 3.54\right) = 2.50$$

$$d_{((D,F),E)\to C} = \min\left(d_{DC}, d_{FC}, d_{EC}\right) = \min\left(2.24, 2.50, 1.41\right) = 1.41$$

Agglomerative Clustering...

Agglomerative Clustering...

In updated distance matrix, the minimum value is 1.41 which is distance between cluster C and cluster ((D, F), E). Thus, we group them into a cluster (((D, F), E), C). The updated distance matrix is shown below:

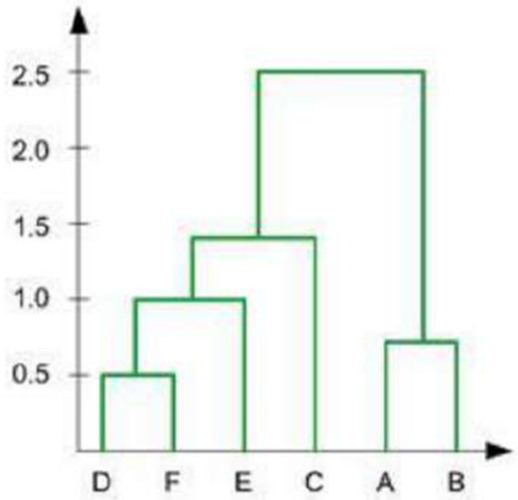
	A, B	((D, F), E), C
A, B	0	
((D, F), E), C	2.50	0

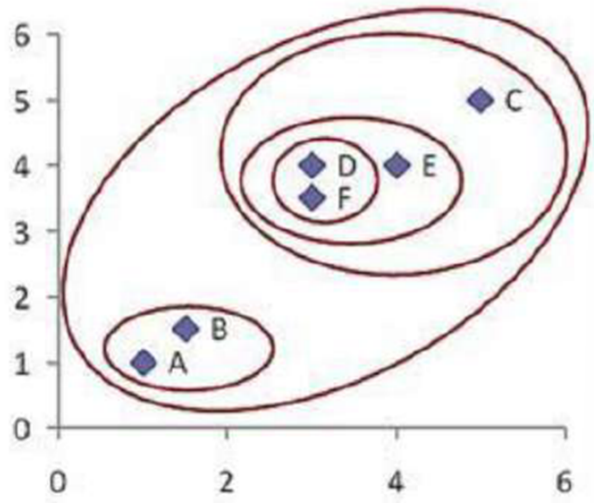
$$d_{(((D,F),E),C)\rightarrow(A,B)} = \min\left(3.61, 2.92, 3.20, 2.50, 4.24, 3.54, 5.66, 4.95\right) = 2.50$$

Required two clusters are: (A, B) and (((D, F), E), C).

If we merge the remaining two clusters, we will get only single cluster contain the whole 6 objects.

Using this information, we can now draw the final result of dendogram and clustering hierarchy into XY space:





Divisive Clustering

Divisive Analysis Clustering Algorithm (DIANA)

Q. Cluster the data points (1, 1), (1.5, 1.5), (5, 5), (3, 4), (4, 4) and (3, 3.5) using Divisive clustering

Assume A=(1, 1), B=(1.5, 1.5), C=(5, 5), D=(3, 4), E=(4, 4) and F=(3, 3.5)

The distance matrix is:

	A	B	C	D	E	F
A	0					
B	0.71	0				
C	5.66	4.95	0			
D	3.61	2.92	2.24	0		
E	4.24	3.54	1.41	1.00	0	
F	3.20	2.50	2.50	0.50	1.12	0

Average dissimilarity of A = (d(a,b) + d(a,c) + d(a,d) + d(a,e) + d(a,f)) / 5 = 3.484

Average dissimilarity of B = (d(b,a) + d(b,c) + d(b,d) + d(b,e) + d(b,f)) / 5 = 2.924

Average dissimilarity of C = (d(c, a) + d(c, b) + d(c,d) + d(c,e) + d(c,f)) / 5 = 3.352

Average dissimilarity of D = (d(d,a) + d(d,b) + d(d,c) + d(d,e) + d(d,f)) / 5 = 2.054

Average dissimilarity of E = (d(e,a) + d(e,b) + d(e,c) + d(e,d) + d(e,f)) / 5 = 2.262

Average dissimilarity of F = (d(f,a) + d(f,b) + d(f,c) + d(f,d) + d(f,e)) / 5 = 1.964

Highest average dissimilarity is 3.484. Cl1 = {B, C, D, E, F} and Cl2 = {A}

Divisive Clustering...

Cl1 = {B, C, D, E, F} and Cl2 = {A}

Calculate distance for each object in Cl1:

Db =average dissimilarity of b with objects in Cl1 – dissimilarity of b with Cl2
= (d(b,c) + d(b,d) + d(b,e) + d(b,f)) / 4 - d(b,a) = 3.4775 – 0.71 = 2.7675

Dc =average dissimilarity of c with objects in Cl1 – dissimilarity of c with Cl2
= (d(c, b) + d(c,d) + d(c,e) + d(c,f)) / 4 - d(c, a) = 2.775 – 5.66 = -2.885

Dd =average dissimilarity of d with objects in Cl1 – dissimilarity of d with Cl2
= (d(d,b) + d(d,c) + d(d,e) + d(d,f)) / 4 – d(d,a) = 1.665 – 3.61 = -1.945

De =average dissimilarity of e with objects in Cl1 – dissimilarity of e with Cl2
= (d(e,b) + d(e,c) + d(e,d) + d(e,f)) / 4 – d(e,a) = 1.7675 – 4.24 = -2.4725

Df =average dissimilarity of f with objects in Cl1 – dissimilarity of f with Cl2
= (d(f,b) + d(f,c) + d(f,d) + d(f,e)) / 4 - d(f,a) = 1.655 – 3.2 =-1.545

Since Db > 0, B is moving to Cl2

Cl1 = {C, D, E, F} and Cl2 = {A, B}

Divisive Clustering...

Cl1 = {C, D, E, F}

Average dissimilarity of C = (d(c,d) + d(c,e) + d(c,f)) / 3 = 2.05

Average dissimilarity of D = (d(d,c) + d(d,e) + d(d,f)) / 3 = 1.246

Average dissimilarity of E = (d(e,c) + d(e,d) + d(e,f)) / 3 = 1.176

Average dissimilarity of F = (d(f,c) + d(f,d) + d(f,e)) / 3 = 1.373

Highest average dissimilarity is 2.05. Cl3 = {D, E, F} and Cl4 = {C}

Calculate distance for each object in Cl3:
Dd =average dissimilarity of d with objects in Cl3 – dissimilarity of d with Cl4
= (d(d,e) + d(d,f)) / 2 - d(d,c) = 0.75 – 2.24 = -1.49

De =average dissimilarity of e with objects in Cl3 – dissimilarity of e with Cl4
= (d(e,d) + d(e,f)) / 2 - d(e,c) = 1.06 – 1.41 = -0.35

Df =average dissimilarity of f with objects in Cl3 – dissimilarity of f with Cl4
= (d(f,d) + d(f,e)) / 2 – d(f,c) = 0.81 – 2.5 = -1.69

Since none of Dx is greater than 0, Cl3 and Cl4 remain as it is.
Cl3 = {D, E, F} and Cl4 = {C}

Divisive Clustering...

Cl3= {D, E, F}

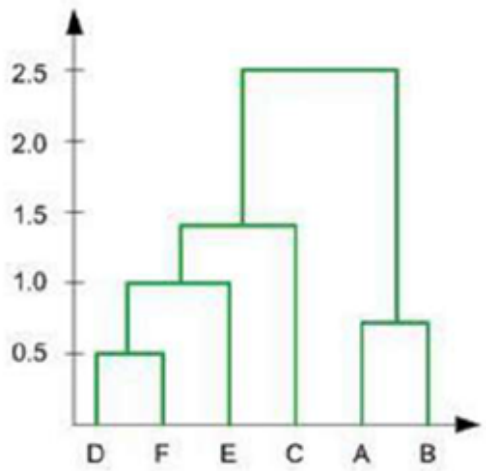
Average dissimilarity of D = (d(d,e) + d(d,f)) / 2 = 0.75

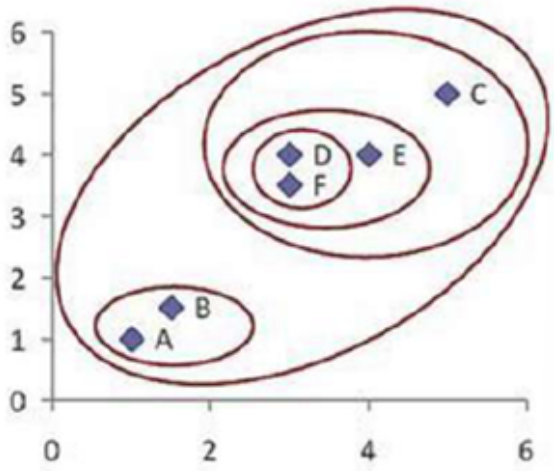
Average dissimilarity of E = (d(e,d) + d(e,f)) / 2 = 1.06

Average dissimilarity of F = (d(f,d) + d(f,e)) / 2 = 0.81

Highest average dissimilarity is 1.06.
Cl5 = {D, F} and Cl6 = {E}

So, final clusters: **Cl2 = {A, B}, Cl4 = {C}, Cl5 = {D, F} and Cl6 = {E}**





DBSCAN

DBSCAN(Density-Based Spatial Clustering of Applications with Noise)

- It can discover clusters of different shapes and sizes from a large amount of data, which is containing noise & outliers
- The main idea of DBSCAN is that a point belongs to a cluster if it is close to many points from that cluster
- DBSCAN algorithm uses two parameters:
  - **Epsilon(ϵ)**: The distance that specifies the neighborhoods. Two points are neighbors if the distance between them is less than or equal to ϵ
  - **minPts**: The minimum number of points (a threshold) clustered together for a region to be considered dense
- Based on these parameters, points are classified as:
  - **Core Point**: Data point that has at least *minPts* number of points within ϵ distance
  - **Border Point**: Data point that has at least one core point within ϵ distance and lower than *minPts* number of points within ϵ distance from it
  - **Noise or Outlier Point**: Data point that has no core points within ϵ distance

DBSCAN...

DBSCAN Algorithm

Step 1: Label Core point and Noise point

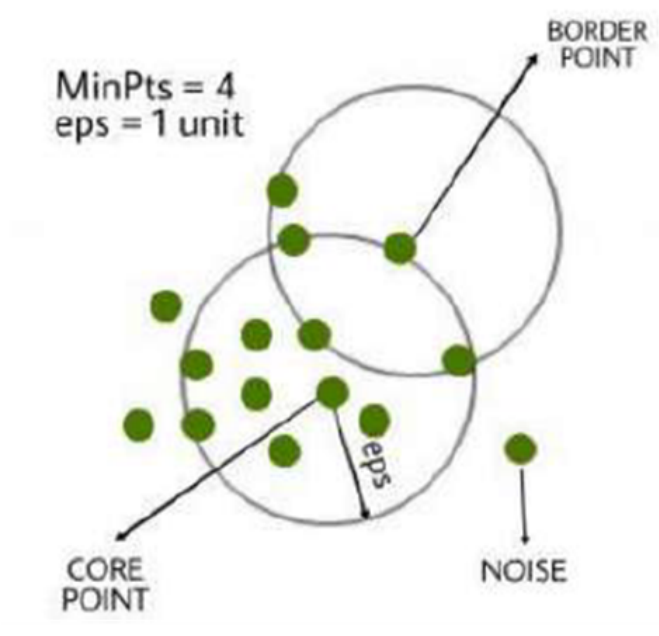
- Select a random starting point, say **O**
- Identify neighborhood of this point **O** using the radius ϵ
- Count the number of points, say **k**, in this neighborhood including point **O**
- If **k** $\geq$ **minPts** then mark **O** as a core point, Else it will be marked as noise point
- Select a new unvisited point and repeat the above steps

Step 2: Check if noise point can become boundary point

- If noise point is directly density reachable (i.e. within the boundary of radius ϵ from the core point), mark it as boundary points and it will form the part of the cluster
- A point which is neither core point nor boundary point is marked as noise

DBSCAN...

DBSCAN Algorithm...



DBSCAN Algorithm...

Q. Cluster the data points (3,7), (4,6), (5,5), (6,4), (7,3), (6,2), (7,2), (8,4), (3,3), (2,6), (3,5) and (2,4) using DBSCAN algorithm. Assume epsilon=1.9 and MinPts=4

To find the cluster by using DBSCAN we need to first calculate the distance among all pairs of given data point. Let us use *Euclidean distance* measure for distance calculation:

$$d = \sqrt{(x_2 - x_1)^2 + (y_2 - y_1)^2}$$

The distance matrix is:

P1	0.00											
P2	1.41	0.00										
P3	2.83	1.41	0.00									
P4	4.24	2.83	1.41	0.00								
P5	5.66	4.24	2.83	1.41	0.00							
P6	5.83	4.47	3.16	2.00	1.41	0.00						
P7	6.40	5.00	3.61	2.24	1.00	1.00	0.00					
P8	5.83	4.47	3.16	2.00	1.41	2.83	2.24	0.00				
P9	4.00	3.16	2.83	3.16	4.00	3.16	4.12	5.10	0.00			
P10	1.41	2.00	3.16	4.47	5.83	5.66	6.40	6.32	3.16	0.00		
P11	2.00	1.41	2.00	3.16	4.47	4.24	5.00	5.10	2.00	1.41	0.00	
P12	3.16	2.83	3.16	4.00	5.10	4.47	5.39	6.00	1.41	2.00	1.41	0.00
	P1	**P2**	**P3**	**P4**	**P5**	**P6**	**P7**	**P8**	**P9**	**P10**	**P11**	**P12**

DBSCAN...

DBSCAN Algorithm...

Point and their neighbours within the boundary of radius $\varepsilon = 1.9$ are given below:

P1: P2, P10	**P2**: P1, P3, P11	**P3**: P2, P4	**P4**: P3, P5
P5: P4, P6, P7, P8	**P6**: P5, P7	**P7**: P5, P6	**P8**: P5
P9: P12	**P10**: P1, P11	**P11**: P2, P10, P12	**P12**: P9, P11

Given, $minPts = 4$, Thus, Core points are: $P2, P5$ and $P11$

Outliers are: $P1, P3, P4, P6, P7, P8, P9, P10$ and $P12$

Check for direct density reachable condition for noise points (outliers). If density reachable condition is satisfied, convert noise to boundary point.

Points $P1, P3, P4, P6, P7, P8, P10$ and $P12$ become the boundary points.

Hence,

Core points are: $P2, P5$ and $P11$

Boundary points are: $P1, P3, P4, P6, P7, P8, P10$ and $P12$

Noise point: $P9$

Now constructing clusters:

$C_1 = \{P2, P1, P3, P11, P10, P12\} = \{P1, P2, P3, P10, P11, P12\}$

$C_2 = \{P5, P4, P6, P7, P8\} = \{P4, P5, P6, P7, P8\}$

DBSCAN Algorithm...

Q. Perform DBSCAN on the given problem with epsilon=2 and MinPts=2

	A	B	C	D	E	F	G	H
X	2	2	8	5	7	6	1	4
Y	10	5	4	8	5	4	2	9

To find the cluster by using DBSCAN we need to first calculate the distance among all pairs of given data point. Let us use *Euclidean distance* measure for distance calculation:

$$d = \sqrt{(x_2 - x_1)^2 + (y_2 - y_1)^2}$$

The distance matrix is:

A	0							
B	5	0						
C	8.49	6.08	0					
D	3.61	4.24	5	0				
E	7.07	5	1.41	3.61	0			
F	7.21	4.12	2	4.12	1.41	0		
G	8.06	3.16	7.28	7.21	6.71	5.39	0	
H	2.24	4.47	6.4	1.42	5	5.39	7.62	0
	A	***B***	***C***	***D***	***E***	***F***	***G***	***H***

Point and their neighbours within the boundary of radius $\varepsilon = 2$ are given below:

A: {}	***B***: {}	***C***: E, F	***D***: H
E: C, F	***F***: C, E	***G***: {}	***H***: D

Given, $minPt = 2$. Thus, Core points are: C, D, E, F, H

Outliers are: A, B, G

There are 2 clusters formed:

$C_1 = \{C, E, F\}$

$C_2 = \{D, H\}$

DBSCAN...

Advantages

- Doesn't require a prior specification of number of clusters
- Able to identify noise data while clustering
- DBSCAN algorithm is able to find arbitrarily size and arbitrarily shaped clusters
- Robust to outliers

Disadvantages

- Can't cluster well with huge differences in densities
- Doesn't work well in case of high dimensional data
- Can be confused when there's a border point that belongs to two clusters
- Choosing a meaningful ϵ value can be difficult if the dat isn't well understood

Grid-Based Clustering

- The space of instance is divided into a grid structure. Clustering techniques are then applied using the cells of the grid, instead of individual data points, as the base units
- Advantage of this method is to improve the processing time
- STING (Statistical Information Grid) is a grid-based clustering technique. It uses a multidimensional grid data structure that quantifies space into a finite number of cells. Instead of focusing on data points, it focuses on the value space surrounding the data points
- In STING, the spatial area is divided into rectangular cells and several levels of cells at different resolution levels. High level cells are divided into several low-level cells
- The statistical parameter of higher-level cells can easily be computed from the parameters of the lower-level cells

Classification vs. Clustering

Classification	*Clustering*
Process of classifying the input instances based on their corresponding class labels.	Grouping the instances based on their similarity without the help of class labels
Classification is a type of supervised learning method.	Clustering is a kind of unsupervised learning method.
The number of classes is known.	The number of classes is unknown.
It uses a training dataset.	It does not use a training dataset.
In classification, there are labels for training data.	In clustering, there are no labels for training data.
Its objective is to find which class a new object belongs to form the set of predefined classes.	Its objective is to group a set of objects to find whether there is any relationship between them.
More complex as compared to clustering.	Less complex as compared to classification.
Popular algorithms for classification include Naive Bayes Classifier, Decision Trees, Support Vector Machines etc.	Popular algorithms used for clustering include K-Means, DBSCAN etc.

Ref: J. Han, M. Kamber and J. Pei, "Data Mining: Concepts and Techniques", Morgan Kaufmann, 3rd edition