UNIT-IV
DATA MINING ALGORITHMS

What Is Association Mining?

- **Association rule mining:**
  - ARM is based on Transactional database
  - It is also **Finding frequent patterns**, associations, correlations, or causal structures among sets of items or objects in transaction databases, relational databases, and other information repositories.
  - **With the help of ARM we can easily finds which items are frequently purchased together by customers**?
- Applications:
  - Basket data analysis, cross-marketing, catalog design,clustering, classification, etc.

Association rules

- **These are Techniques for data mining and knowledge discovery in databases**

Two important algorithms in the development of association rules are...

- **Apriori Algorithm**
- **FP growth (Frequent Pattern growth)**

Apriori algorithm

- Developed by Agrawal and Srikant 1994
- Innovative way to find association rules on large scale, allowing implication outcomes that consist of more than one item
- **Based on minimum support threshold**

Application of data mining

- Data mining can typically be used with transactional databases (for ex. in shopping cart analysis)
- Aim can be to build association rules about the shopping events
- Based on **item sets**, such as

  {milk, cocoa powder} 2-itemset

  {milk, corn flakes, bread} 3-itemset

Association rules

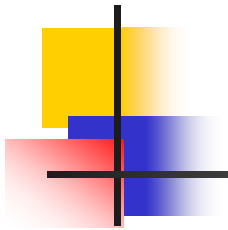
- Items that occur often together can be associated to each other
- These together occuring items form a **frequent itemset**
- Conclusions based on the frequent itemsets form **association rules**
- For ex. {milk, cocoa powder} can bring a rule *cocoa powder* ➔ *milk*

Sets of database

- Transactional database D
- All products an itemset $I = \{i_1, i_2, \ldots, i_m\}$
- Unique shopping event $T \subseteq I$
- T contains itemset X iff $X \subseteq T$
- Based on itemsets X and Y an association rule can be $X \rightarrow Y$
- It is required that $X \subset I$, $Y \subset I$ and $X \cap Y = \varnothing$

Subjective measures

- **Often based on earlier user experiences and believes.**
- Unexpectedness: rules are interesting if they are unknown or contradict the existing knowledge (or expectations).
- Actionability: rules are interesting if users can get advantage by using them
- Weak and strong believes.



Objective measures

- **Based on threshold values controlled by the user**
- **Some typical measures**
  - **support** (utility)
  - **confidence** (certainty)

Support and confidence

- If confidence gets a value of 100 % the rule is an **exact rule**
- Even if confidence reaches high values the rule is not useful unless the support value is high as well
- **Rules that have both high confidence and support are called <u>strong rules</u>**

Support

- **Support:** Support means the percentage of transactions in which A and B are purchased together.
- **Support of the rule S=P(AUB)**

  = Frequency of A & B purchased together *100

  Total number of Transactions

  Here A is known as “Antecedent” (forenunner) and B is known as “Consequence”(outcome)

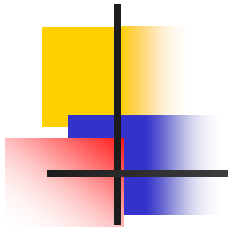
Confidence

- **Confidence:** Confidence means the percentage of transactions while purchasing A how many times B is also purchased.
- **Confidence of the rule C=P(A||B)**

  = Frequency of A & B purchased together *100

  Total number of Transactions containing A

Confidence of the rule identifies that **"How much the rule is correct or strong?".**



Cont...

- **Frequency**/Support/count item=**No of occurrence of item** in the **transaction database.**
- **Minimum Support means the occurrence of the item is equal to or greater than its frequent count .**
- Any rule is **strong** if it satisfies the **minimum(min sup) support value and minimum confidence(min conf) value.**

Confidence (certainty)

- Certainty of a rule can be measured with a threshold for confidence
- This parameter lets to measure how often an event's itemset that matches the left side of the implication in the association rule also matches for the right side
- Rules for events whose itemsets do not match sufficiently often the right side while mathching the left (defined by a threshold value) can be excluded

Generating association rules

- Usually consists of two subproblems
  1) Finding frequent itemsets whose occurences exceed a predefined minimum support threshold
  2) Deriving association rules from those frequent itemsets (with the constrains of minimum confidence threshold)
- These two subproblems are soleved iteratively until new rules no more emerge
- The second subproblem is quite straight- forward and most of the research focus is on the first subproblem

Use of Apriori algorithm

- Initial information: transactional database D and user-defined numeric minimun support threshold *min_sup*
- Algortihm uses knowledge from previous iteration phase to produce frequent itemsets
- This is reflected in the Latin origin of the name that means ”from what comes before”

Creating frequent sets

- Let's define:

  C_k as a candidate itemset of size k

  L_k as a frequent itemset of size k

- Main steps of iteration are:
  1) Find frequent set L_{k-1}
  2) Join step: C_k is generated by joining L_{k-1} with itself (cartesian product L_{k-1} x L_{k-1})
  3) Prune step (apriori property): Any (k – 1) size itemset that is not frequent cannot be a subset of a frequent k size itemset, hence should be removed
  4) Frequent set L_k has been achieved

Apriori algorithm in pseudocode

L_1 = {frequent items};
for (k= 2; L_{k-1} !=∅; k++) **do begin**
    C_k= candidates generated from L_{k-1} (that is: cartesian product L_{k-1} x L_{k-1} and eliminating any
    k-1 size itemset that is not frequent);
    for each transaction t in database **do**
        increment the count of all candidates in
        C_k that are contained in t
    L_k = candidates in C_k with *min_sup*
    end
return $\cup_k L_k$;

Apriori algorithm in pseudocode (2)

Apriori Pseudocode

Apriori (T, ε)

$L_1 \leftarrow \{$ *large 1-itemsets that appear in more than* ε *transactions* $\}$

$k \leftarrow 2$

while $L_{k-1} \neq \varnothing$

$C_k \leftarrow$ Generate(L_{k-1})

for transactions $t \in T$

$C_t \leftarrow$ Subset(C_k,t)

for candidates $c \in C_t$

$\text{count}[c] \leftarrow \text{count}[c] + 1$

$L_k \leftarrow \{c \in C_k | \text{ count}[c] \geq \varepsilon\}$

$k \leftarrow k + 1$

return $\bigcup L_k$

← Join step and prune step

Mining Association Rules-An Example

Transaction ID	Items Bought
2000	A,B,C
1000	A,C
4000	A,D
5000	B,E,F

Min. support 50%
Min. confidence 50%

Frequent Itemset	Support
{A}	75%
{B}	50%
{C}	50%
{A,C}	50%

For rule $A \Rightarrow C$:

support = support({*A* U*C* }) = 50%

confidence = support({*A* U*C* })/support({*A*}) = 66.6%

The Apriori principle:

Any subset of a frequent itemset must be frequent

Steps for Mining Frequent Item sets

- **Find the *frequent itemsets*: the sets of items that have minimum support**
  - A subset of a frequent itemset must also be a frequent itemset
    - i.e., if {*AB*} is a frequent itemset, both {*A*} and {*B*} should be a frequent itemset
  - Iteratively find frequent itemsets with cardinality from 1 to *k (k*-itemset*)*
- **Use the frequent itemsets to generate association rules.**

The Apriori Algorithm

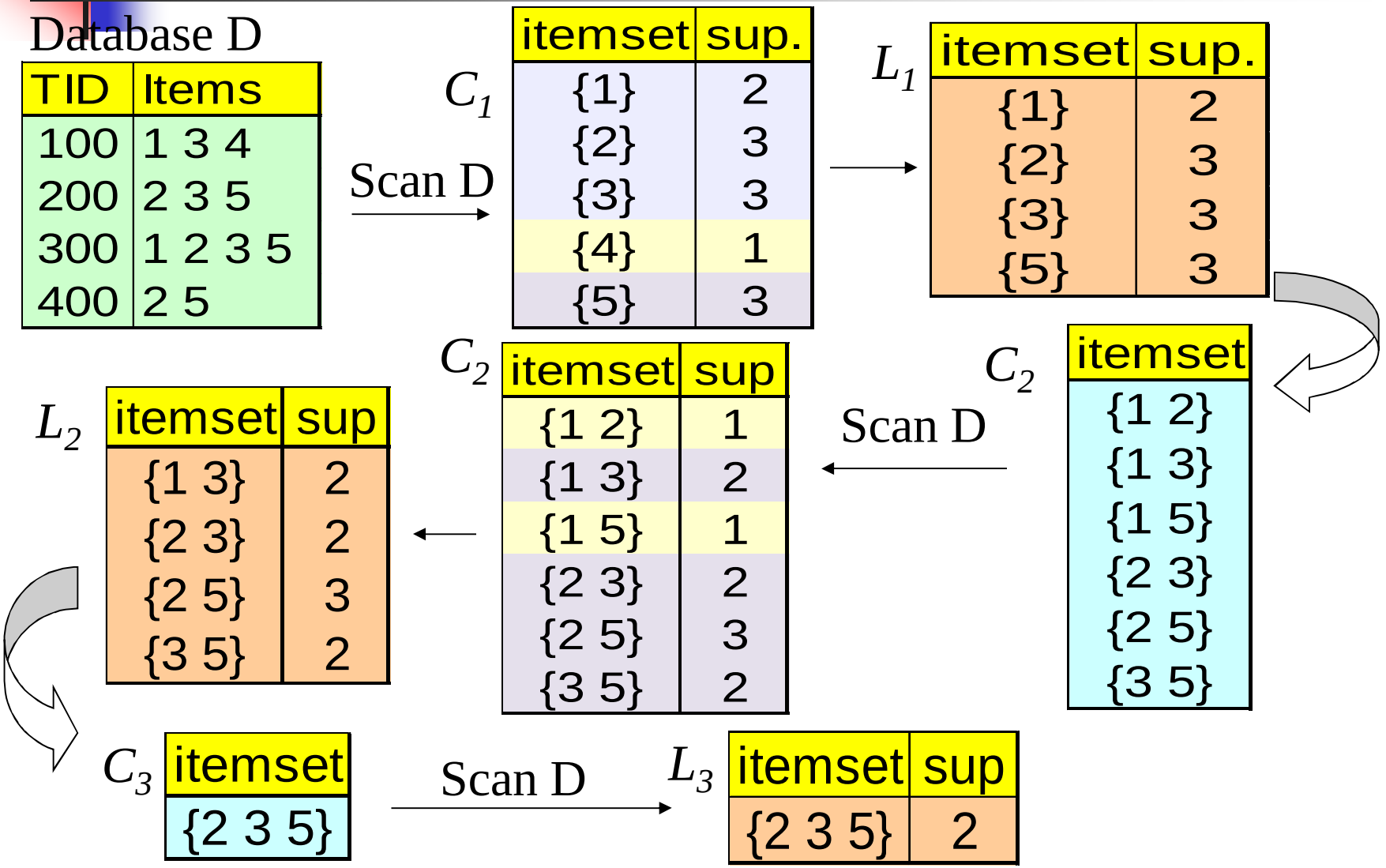
- C=Candidate Itemsets (all)
- L=Frequent Itemsets (selected)
- Join Step: C_k is generated by joining $L_{k-1}$with itself
- Prune Step: Any (k-1)-itemset that is not frequent cannot be a subset of a frequent k-itemset
- Pseudo-code:

C_k: Candidate itemset of size k
L_k : frequent itemset of size k

L_1 = {frequent items};
for (k = 1; L_k !=∅; k++) **do begin**
  C_{k+1} = candidates generated from L_k;
  for each transaction t in database do
    increment the count of all candidates in C_{k+1} that are contained in t
  L_{k+1} = candidates in C_{k+1} with min_support
  end
return $\cup_k L_k$;

The Apriori Algorithm — Example

min support=50% & confidence-50%



Steps to find Frequent item sets Step-I

- Take minimum support=2
- Compare item size 1,2 & 3 with main DB
- Item set of size 1={1,2,3,4,5}
- Item set of size 2
  ={(1,2),(1,3),(1,5),(2,3),(2,5),(3,5)}
- Item set of size 3
  ={(1,3,5),(1,2,3),(1,2,5),(2,3,5)}

Only item set having support value is equal to or greater than 2 are frequent patterns.

Therefore (2,3,5) are the candidate set of size of item set 3.

Step-II
To generate the Association Rule

- Item={2,3,5} with min support=2 i.e. 50%
  min confidence=50%

Support S is the subset of i

S={(2),(3),(5),(2,3),(2,5),(3,5)}

Rules are:

1. {2} → {3,5} see 3,5,with item {2}

$$\frac{\{2,3,5\}}{\text{Total no of Transactions}} = 2/4$$

Support=2/4=0.5=50%

Cont...

- **Confidence = {2,3,5}**

  Total no of transactions related to 2

  (how many times 2 occurs)

Confidence=2/3=66.6%

This shows strong Association Rule.

2. {3} → {2,5}

Support = 2/4=0.5=50%

Confidence=2/3=66.6=66.66%

Cont…

3. {5} → {2,3}

Support=2/4=0.5=50%

Confidence=2/3=.66=66%

4. {2,3} → {5}

Support=2/4=0.5=50%

Confidence=2/2=1=100%

5. {2,5} → {3}

Support=2/4=0.5=50%

Confidence=2/3=0.6=60%

Cont..

6. {3,5} → {2}

Support=2/4=0.5=50%

Confidence=2/2=1=100%

Question:2

Minimum support=3

Min. Conf.=80%

Find all frequent item set and set of association rule using Apriori Algorithm

Transactions (Tid)	Item
T1	2,3,5
T2	2,3,5,1
T3	1,2,3,5
T4	2,5

Step-II
To generate the Association Rule

- Item={2,3,5} with min support=3 i.e. 75%
  min confidence=80%

Support S is the subset of i

S={(2),(3),(5),(2,3),(2,5),(3,5)}

Rules are:

1. {2} → {3,5} see {2} with item {3,5}

$$\frac{\{2,3,5\}}{\text{Total no of Transactions}} = 3/4$$

Support=2/4=0.75=75%

Cont...

- **Confidence = $\frac{\{2,3,5\}}{\text{Total no of transactions related to 2}}$**

(how many times 2 occurs)

Confidence=3/4=75%

This shows strong Association Rule.

2. {3} → {2,5}

Support = 3/4=0.75=75%

Confidence=3/3=1=100%

Cont...

3. {5} → {2,3}

Support=3/4=0.75=75%

Confidence=3/3=1=100%

4. {2,3} → {5}

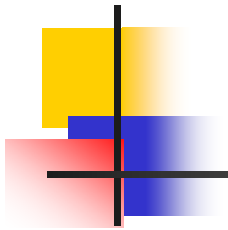
Support=3/4=0.75=75%

Confidence=3/3=1=100%

5. {2,5} → {3}

Support=3/4=0.75=75%

Confidence=3/3=1=100%



Cont..

6. {3,5} → {2}

Support=3/4=0.75=75%

Confidence=3/3=1=100%

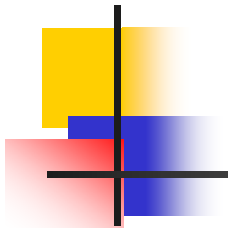
Question:3

Minimum support=60% (min support value=3)

Min. Conf.=80%

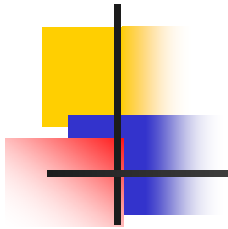
Find all frequent item set and set of association rule using Apriori Algorithm

Transactions(Tid)	Item
T1	M, O, N, K, E, Y
T2	D, O, N, K, E, Y
T3	M, A, K, E
T4	M, U, C, K, Y
T5	C, O, O, K, I, E



Disadvantages of Apriori :

- Huge candidate sets
- Again and again scanning of database
- Multiple scans of database
- Complexity is more
- Time taking



Limitations of Apriori algorithm

- Needs several iterations of the data
- Uses a **uniform** minimum support threshold
- Difficulties to find rarely occuring events
- Alternative methods (other than appriori) can address this by using a **non-uniform** minimum support thresold

(FP-tree)

Frequent-Pattern tree

Mining Frequent Patterns Without Candidate Generation

- Compress a large database into a compact, **Frequent-Pattern tree** (**FP-tree**) structure
  - highly condensed, but complete for frequent pattern mining
  - avoid costly database scans
- Develop an efficient, FP-tree-based frequent pattern mining method
  - A divide-and-conquer methodology: decompose mining tasks into smaller ones
  - Avoid candidate generation: sub-database test only!

Construct FP-tree from a Transaction DB

TID	Items bought	(ordered) frequent items
100	{f, a, c, d, g, i, m, p}	{f, c, a, m, p}
200	{a, b, c, f, l, m, o}	{f, c, a, b, m}
300	{b, f, h, j, o}	{f, b}
400	{b, c, k, s, p}	{c, b, p}
500	{a, f, c, e, l, p, m, n}	{f, c, a, m, p}

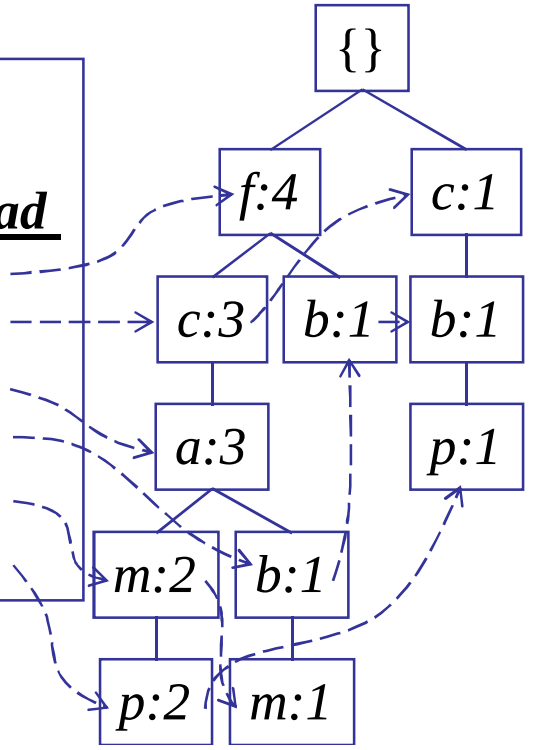
min_support = 0.5

Steps:

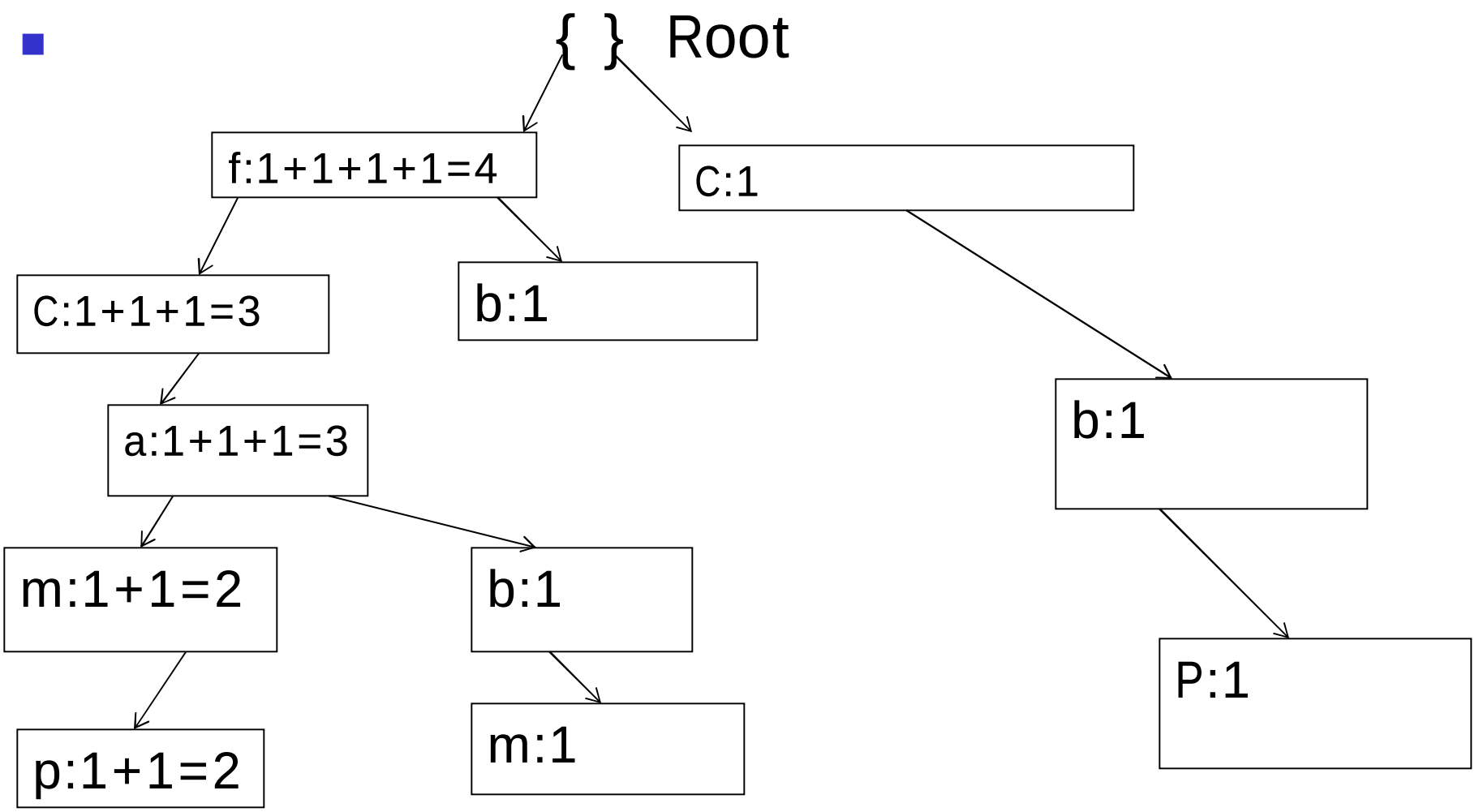
1. Scan DB once, find frequent 1-itemset (single item pattern)
2. Order frequent items in frequency descending order
3. Scan DB again, construct FP-tree

Header Table

Item	frequency	head
f	4	
c	4	
a	3	
b	3	
m	3	
p	3	



STEPS TO GET FREQUENT PATTERNS USING FP TREE



Benefits of the FP-tree Structure

- Completeness:
  - never breaks a long pattern of any transaction
  - preserves complete information for frequent pattern mining
- Compactness
  - reduce irrelevant information—infrequent items are gone
  - frequency descending ordering: more frequent items are more likely to be shared
  - never be larger than the original database (if not count node-links and counts)

Mining Frequent Patterns Using FP-tree

- General idea (divide-and-conquer)
  - Recursively grow frequent pattern path using the FP-tree
- Method
  - For each item, construct its conditional pattern-base, and then its conditional FP-tree
  - Repeat the process on each newly created conditional FP-tree
  - Until the resulting FP-tree is empty, or it contains only one path (single path will generate all the combinations of its sub-paths, each of which is a frequent pattern)

Major Steps to Mine FP-tree

1) Construct conditional pattern base for each node in the FP-tree
2) Construct conditional FP-tree from each conditional pattern-base
3) Recursively mine conditional FP-trees and grow frequent patterns obtained so far
   - If the conditional FP-tree contains a single path, simply enumerate all the patterns

Principles of Frequent Pattern Growth

- Pattern growth property
  - Let α be a frequent itemset in DB, B be α's conditional pattern base, and β be an itemset in B. Then $\alpha \cup \beta$ is a frequent itemset in DB iff β is frequent in B.
- "*abcdef* " is a frequent pattern, if and only if
  - "*abcde* " is a frequent pattern, and
  - "*f* " is frequent in the set of transactions containing "*abcde* "

Why Is Frequent Pattern Growth Fast?

- Our performance study shows
  - FP-growth is an order of magnitude faster than Apriori.
- Reasoning
  - No candidate generation, no candidate test
  - Use compact data structure
  - Eliminate repeated database scan
  - Basic operation is counting and FP-tree building

Summary

- Association rule mining
  - probably the most significant contribution from the database community in KDD
- An interesting research direction
  - Association analysis in other types of data: spatial data, multimedia data, time series data, etc.

Exercises 1

A database has five transactions. Let $min\_sup = 60\%$ and $min\_conf = 80\%$.

TID	*items_bought*
T100	{M, O, N, K, E, Y}
T200	{D, O, N, K, E, Y }
T300	{M, A, K, E}
T400	{M, U, C, K, Y}
T500	{C, O, O, K, I ,E}

(a) Find all frequent itemsets using Apriori and FP-growth, respectively. Compare the efficiency of the two mining processes.

(b) List all of the *strong* association rules (with support s and confidence c) matching the following metarule, where X is a variable representing customers, and $item_i$ denotes variables representing items (e.g., "A", "B", etc.):

$$\forall x \in transaction,\ buys(X, item_1) \wedge buys(X, item_2) \Rightarrow buys(X, item_3) \quad [s, c]$$

Exercise 2

Tid	List of Items
T100	I1,I2,I5
T200	I2,I4
T300	I2,I3
T400	I1,I2,I4
T500	I1,I3
T600	I2,I3
T700	I1 ,I3
T800	I1,I2,I3,I5
T900	I1,I2, I3

- Find the frequent itemset using Apriori and FP-growth respectively. Compare the efficiency of the two mining processes.
- List all of the strong associatiojn rules (with support s and confidence c) matching the following metarule, where X is a variable representing customers, and $item_i$ denote variables represetngin items.

Classification

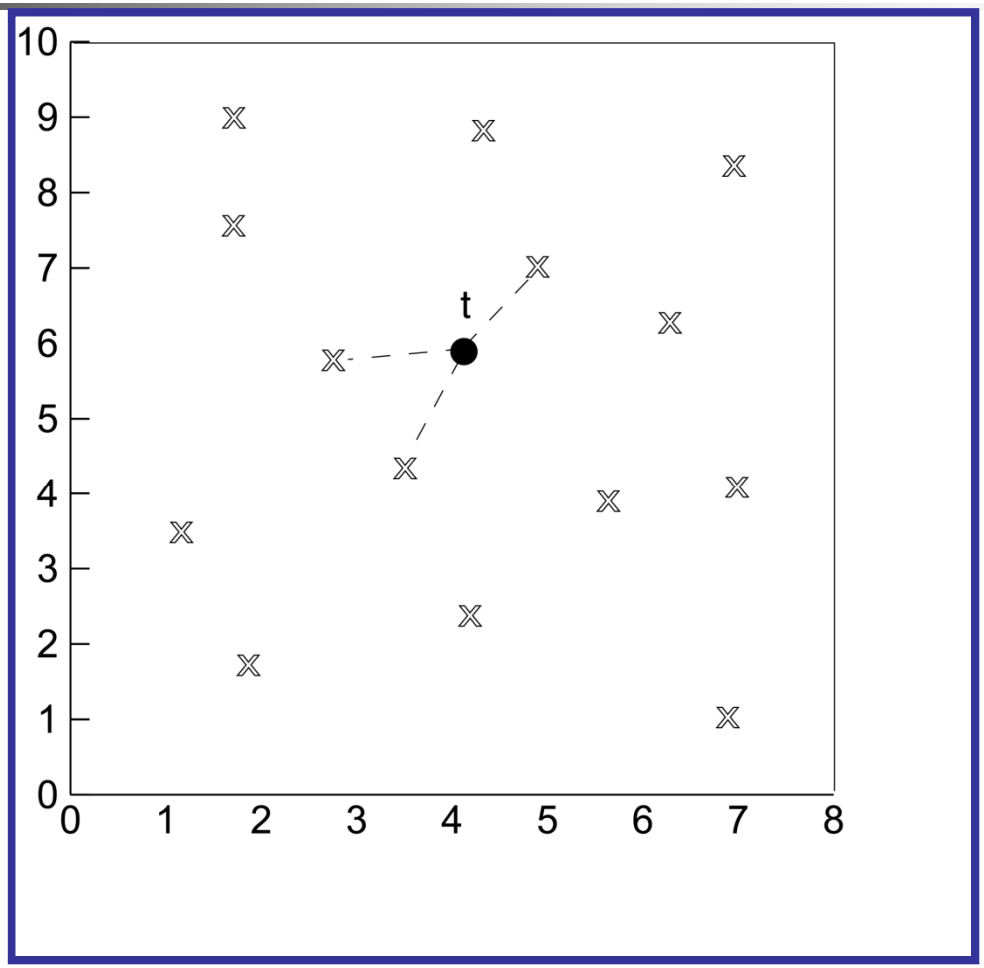
Classification Using Distance

- Place items in class to which they are "closest".
- Must determine distance between an item and a class.
- Classes represented by
  - ***Centroid:*** Central value.
  - ***Medoid:*** Representative point.
  - Individual points
- **Algorithm: KNN**

K Nearest Neighbor (KNN):

- Training set includes classes.
- Examine K items near item to be classified.
- New item placed in class with the most number of close items.
- O(q) for each tuple to be classified. (Here q is the size of the training set.)

KNN



KNN Algorithm

```
Input:
    D       //Training data
    K       //Number of neighbors
    t       //Input tuple to classify
Output:
    c       //Class to which t is assigned
KNN Algorithm:
            //Algorithm to classify tuple using KNN
    N = ∅;
                //Find set of neighbors, N, for t
    foreach d ∈ D do
        if | N |≤ K then
            N = N ∪ d;
        else
            if ∃ u ∈ N such that sim(t, u) ≥ sim(t, d) then
                begin
                    N = N − u;
                    N = N ∪ d;
                end
                //Find class for classification
    c = class to which the most u ∈ N are classified;
```

K-Nearest Neighbor Example- Classification

- Ex. 1 Given:{2,4,6,5,10,12,3,20,30,11,25}, k=2
- Randomly assign numbers: 3 and 12
- K_1={2,3,4,5,6} K_2={10,11,12,20,25,30}
- Ex.2 Given: {a,c,d,g,i,j,n,p,s,w,x,z}, k=2
- Randomly assign letters : g and s
- K_1={a,c,d,g,i,j} K_2={n,p,q,s,w,x,z}
- Ex.2 Given: {10%,30%,50,40%,60%,80%,90%}, k=2
- Randomly assign letters : 30% and 80%
- K_1={10%,30%,40%,50%} K_2={60%,80%, 100%}

Clustering

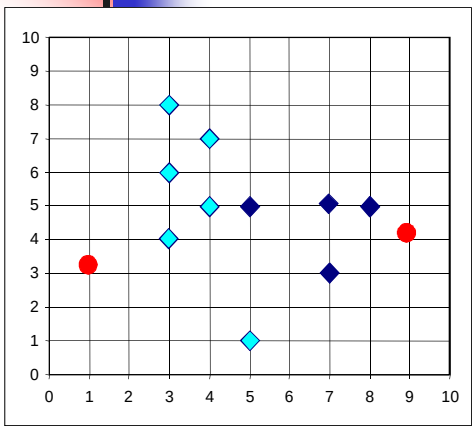
The *k-Means* Clustering Method

Given *k*, the *k-means* algorithm is implemented in four steps:

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

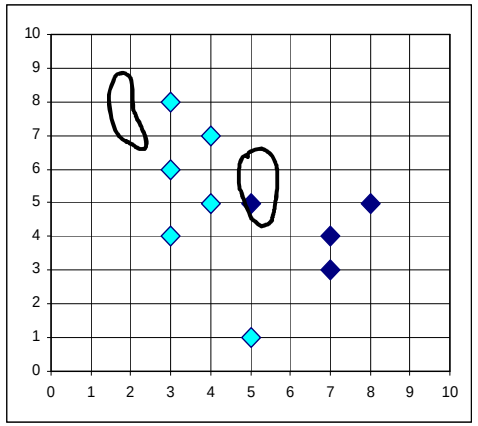
The *K-Means* Clustering Method

Example

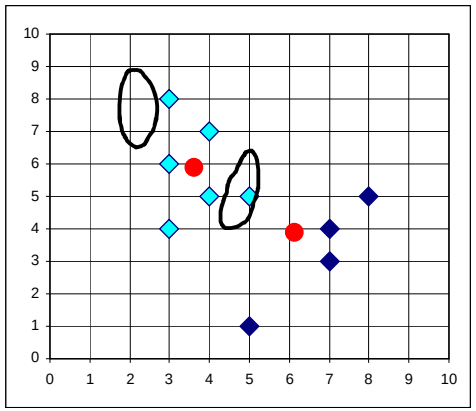


K=2

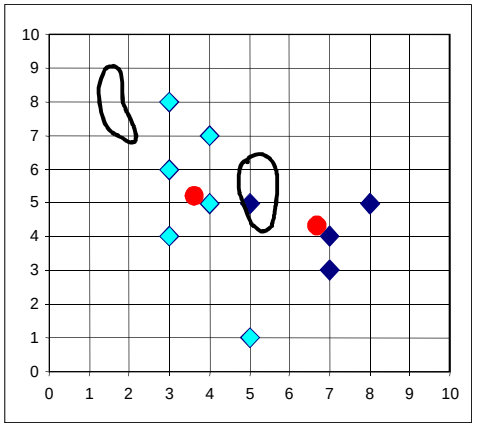
Arbitrarily choose K object as initial cluster center



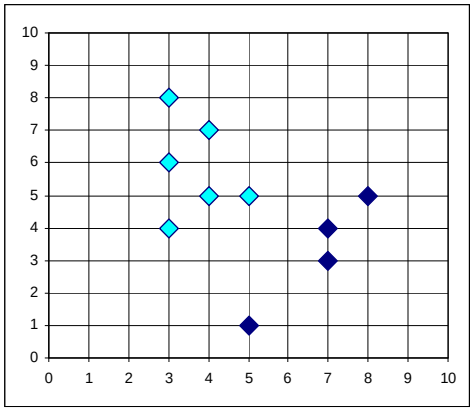
Assign each objects to most similar center



reassign



Update the cluster means



reassign

Update the cluster means

K-Means Clustering Example

- Given: {2,4,10,12,3,20,30,11,25}, k=2
- Randomly assign means: $m_1=3, m_2=4$
- $K_1=\{2,3\}$, $K_2=\{4,10,12,20,30,11,25\}$, $m_1=2.5, m_2=16$
- $K_1=\{2,3,4\}$, $K_2=\{10,12,20,30,11,25\}$, $m_1=3, m_2=18$
- $K_1=\{2,3,4,10\}$, $K_2=\{12,20,30,11,25\}$, $m_1=4.75, m_2=19.6$
- $K_1=\{2,3,4,10,11,12\}$, $K_2=\{20,30,25\}$, $m_1=7, m_2=25$
- Stop as the clusters with these means are the same.

k-Means Algorithm

Input:

$D = \{t_1, t_2, ..., t_n\}$ // Set of elements

A // Adjacency matrix showing distance between elements.

k // Number of desired clusters.

Output:

K // Set of clusters.

K-Means Algorithm:

assign initial values for means $m_1, m_2, ..., m_k$;

repeat

assign each item t_i to the cluster which has the closest mean ;

calculate new mean for each cluster;

until convergence criteria is met;

Comments on the *K-Means* Method

- Strength: Relatively efficient: O(tkn), where n is number of objects, k is number clusters, and t is number of iterations. Normally, k, t << n.
  - Comparing: PAM: $O(k(n-k)^2)$, CLARA: $O(ks^2 + k(n-k))$
- Comment: Often terminates at a local optimum. The global optimum may be found using techniques such as: deterministic annealing and genetic algorithms
- Weakness
  - Applicable only when mean is defined, then what about categorical data?
  - Need to specify k, the number of clusters, in advance
  - Unable to handle noisy data and outliers
  - Not suitable to discover clusters with non-convex shapes

Variations of the *K-Means* Method

- A few variants of the *k-means* which differ in
  - Selection of the initial *k* means
  - Dissimilarity calculations
  - Strategies to calculate cluster means
- Handling categorical data: *k-modes* (Huang'98)
  - Replacing means of clusters with modes
  - Using new dissimilarity measures to deal with categorical objects
  - Using a frequency-based method to update modes of clusters
  - A mixture of categorical and numerical data: *k-prototype* method

Nearest Neighbor Clustering

- Items are iteratively merged into the existing clusters that are closest.
- Incremental
- Threshold, t, used to determine if items are added to existing clusters or a new cluster is created.

Nearest Neighbor Algorithm

Input:

$D = \{t_1, t_2, ..., t_n\}$ // Set of elements

A // Adjacency matrix showing distance between elements.

Output:

K // Set of clusters.

Nearest Neighbor Algorithm:

$K_1 = \{t_1\}$;

$K = \{K_1\}$;

$k = 1$;

for $i = 1$ *to* n do

  find the t_m in some cluster K_m in K such that $dis(t_i, t_m)$ is the smallest;

  if $dis(t_i, t_m) \leq t$ then

    $K_m = K_m \cup t_i$

  else

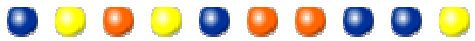
    $k = k + 1$;

    $K_k = \{t_i\}$;

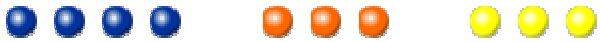
CLUSTERING TECHNIQUES

What is Clustering?

- Clustering is a kind of **unsupervised learning**.
- Clustering is a method of **grouping data** that share **similar trend and patterns**.
- Clustering of data is a method by which large sets of data is grouped into clusters of smaller sets of similar data.
  - Example:



After clustering:



Thus, we see clustering means grouping of data or dividing a large data set into smaller data sets of some similarity.

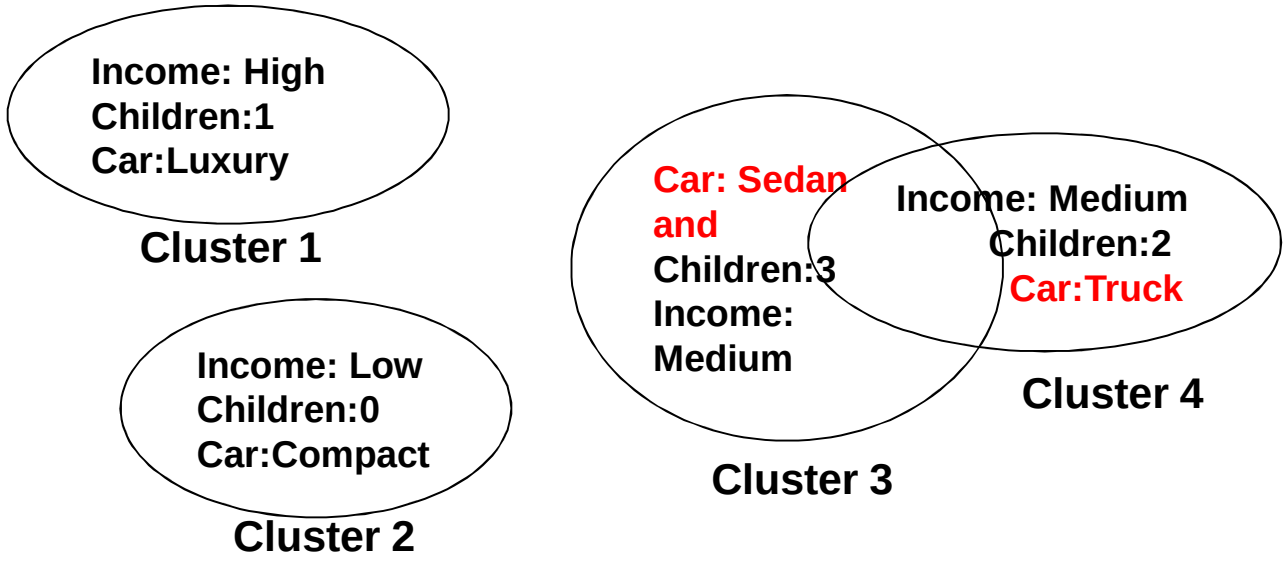
The usage of clustering

- Some engineering sciences such as pattern recognition, artificial intelligence have been using the concepts of cluster analysis.
- Typical examples to which clustering has been applied include handwritten characters, samples of speech, fingerprints, and pictures.
- In the life sciences (biology, botany, zoology, microbiology), the objects of analysis are life forms such as plants, animals, and insects.

Cont...

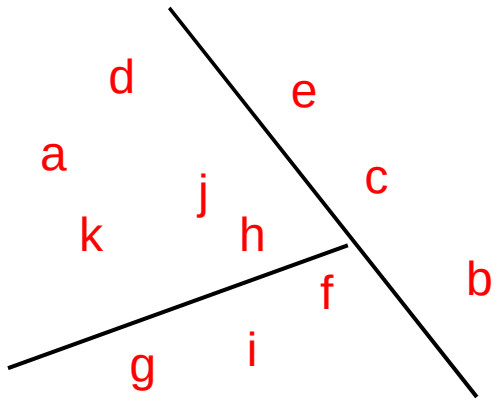
- The clustering analysis may range from developing complete taxonomies to classification of the species into subspecies. The subspecies can be further classified into subspecies.
- Clustering analysis is also widely used in information, policy and decision sciences.
- The various applications of clustering analysis to documents include votes on political issues, survey of markets, survey of products, survey of sales programs, and R & D.

A Clustering Example

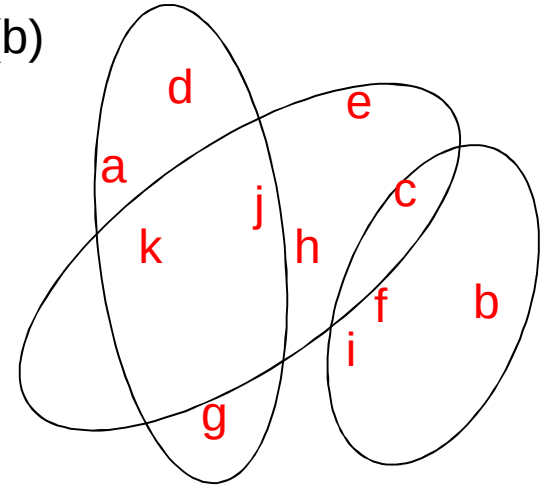


Different ways of representing clusters

(a)



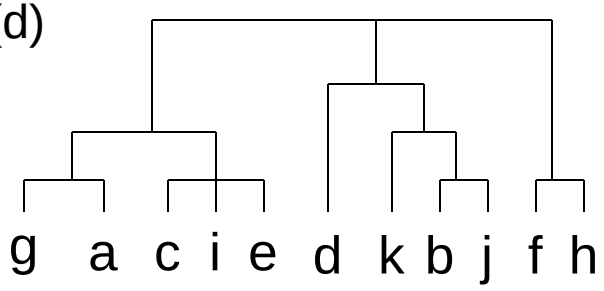
(b)



(c)

	1	2	3
a	0.4	0.1	0.5
b	0.1	0.8	0.1
c	0.3	0.3	0.4
...			

(d)



K Means Clustering
(Iterative distance-based clustering)

- K means clustering is an effective algorithm to extract a given number of clusters of patterns from a training set.
- Once done, the cluster locations can be used to classify patterns into distinct classes.

Clustering Methods

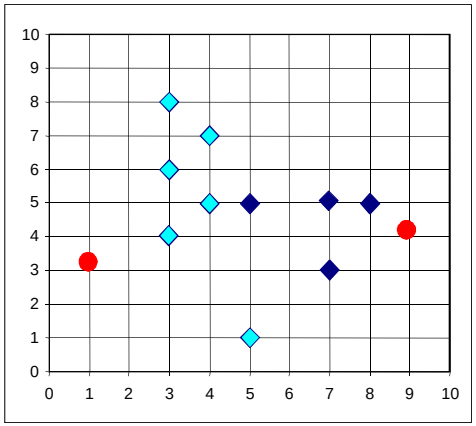
- **K-Mean Clustering**
- Minimum Spanning Tree
- Nearest Neighbor Algorithm

The *k-Means* Clustering Method

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

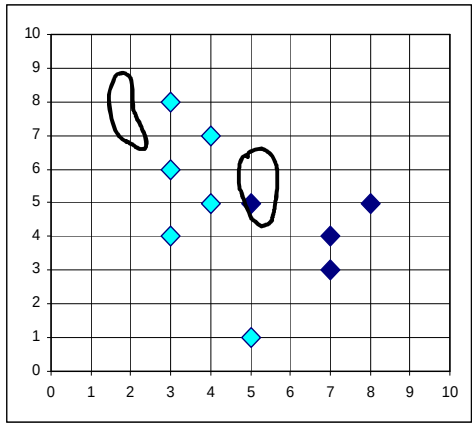
The *K-Means* Clustering Method

- Example

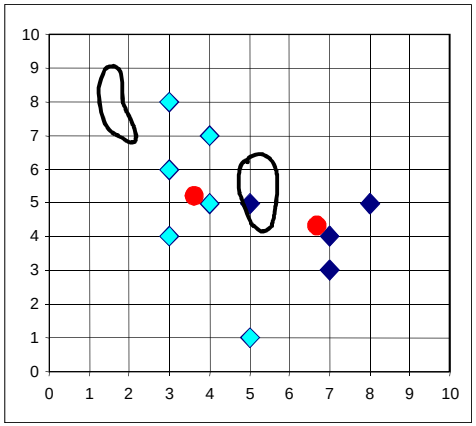


K=2

Arbitrarily choose K object as initial cluster center

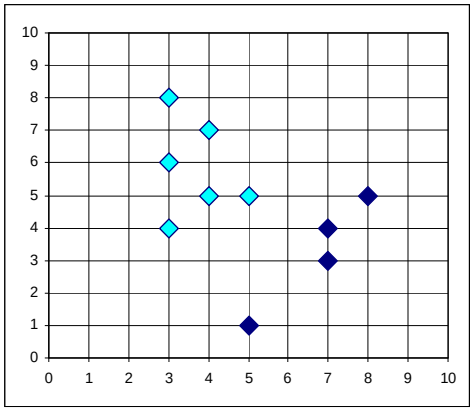


Assign each objects to most similar center

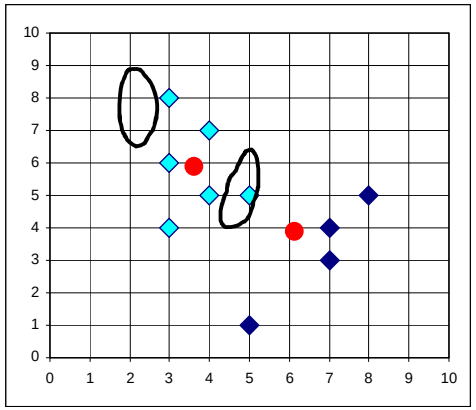


Update the cluster means

reassign



Update the cluster means



reassign

k-Means Algorithm

Input:

$D = \{t_1, t_2, ..., t_n\}$ // Set of elements

A // Adjacency matrix showing distance between elements.

k // Number of desired clusters.

Output:

K // Set of clusters.

K-Means Algorithm:

assign initial values for means $m_1, m_2, ..., m_k$;

repeat

assign each item t_i to the cluster which has the closest mean ;

calculate new mean for each cluster;

until convergence criteria is met;

K means clustering (Cont.)

Select the k cluster centers randomly.

Classify the entire training set. For each patern X_i in the training set, find the nearest cluster center C^* and classify X_i as a member of C^*.

For each cluster, recompute its center by finding the mean of the cluster :

$$M_k = \frac{1}{N_k} \cdot \sum_{j=1}^{N_k} X_{jk}$$

where M_k is the new mean, N_k is the number of training patterns in cluster k, and X_{jk} is the j - th pattern belonging to cluster k.

Loop until the change in cluster means is less the amount specified by the user.

Store the *k* cluster centers.

K-Means Clustering Example

- Given: {2,4,10,12,3,20,30,11,25}, **k=2**
- Randomly select means: **m_1=3,m_2=4**
- K_1={2,3}, K_2={4,10,12,20,30,11,25}, **m_1=2.5,m_2=16**
- K_1={2,3,4},K_2={10,12,20,30,11,25}, **m_1=3,m_2=18**
- K_1={2,3,4,10},K_2={12,20,30,11,25}, **m_1=4.75,m_2=19.6**
- K_1={2,3,4,10,11,12},K_2={20,30,25}, **m_1=7,m_2=25**
- K_1={2,3,4,10,11,12},K_2={20,30,25}, **m_1=7,m_2=25**
- Stop as the clusters with these means are the same.

K-Means Clustering Example-2

- Given: {1,4,10,5,12,2,21,3,8,20, 28,22,11,25}, k=2
- Randomly select means: m_1=4,m_2=5

Comments on the *K-Means* Method

- **Strength:** Relatively efficient: O(tkn), where n is number of objects, k is number clusters, and t is number of iterations.
- **Comment:** Often terminates at a local optimum. The global optimum may be found using techniques such as: genetic algorithms
- **Weakness**
  - Applicable only when mean is defined, then what about categorical data?
  - Need to specify k, the number of clusters, in advance
  - Unable to handle noisy data and outliers
  - Not suitable to discover clusters with non-convex shapes

Variations of the *K-Means* Method

- **A few variants of the *k-means* which differ in**
  - **Selection of the initial *k* means**
  - **Dissimilarity calculations**
  - **Strategies to calculate cluster means**
- **Handling categorical data: *k-modes***
  - **Replacing means of clusters with modes**
  - **Using new dissimilarity measures to deal with categorical objects**
  - **Using a frequency-based method to update modes of clusters**
  - **A mixture of categorical and numerical data: *k-prototype* method**

The drawbacks of K-means clustering

- The final clusters do not represent a global optimization result but only the local one, and complete different final clusters can arise from difference in the initial randomly chosen cluster centers.
- We have to know how many clusters we will have at the first.

classification

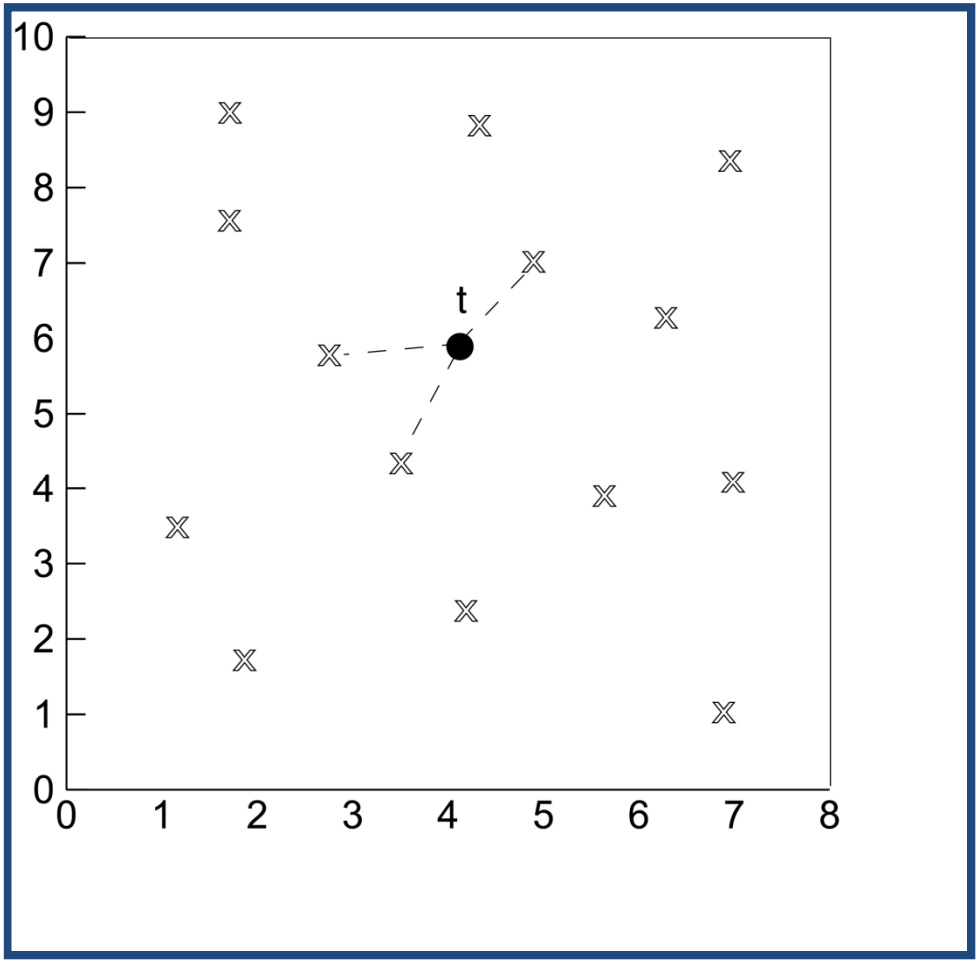
Classification Using Distance

- Place items in class to which they are “closest”.
- Must determine distance between an item and a class.
- Classes represented by
  - ***Centroid:*** Central value.
  - ***Medoid:*** Representative point.
  - Individual points
- **Algorithm: KNN**

K Nearest Neighbor (KNN):

- Training set includes classes.
- Examine K items near item to be classified.
- New item placed in class with the most number of close items.
- O(q) for each tuple to be classified. (Here q is the size of the training set.)

KNN



KNN Algorithm

```
Input:
   D      //Training data
   K      //Number of neighbors
   t      //Input tuple to classify
Output:
   c      //Class to which t is assigned
KNN Algorithm:
          //Algorithm to classify tuple using KNN
   N = ∅;
                //Find set of neighbors, N, for t
   foreach d ∈ D do
      if | N |≤ K then
         N = N ∪ d;
      else
         if ∃ u ∈ N such that sim(t, u) ≥ sim(t, d) then
            begin
               N = N − u;
               N = N ∪ d;
            end
            //Find class for classification
   c = class to which the most u ∈ N are classified;
```

K-Nearest Neighbor Example

- Given: {2,4,10,6,12,3,18,5, 21,11,16}, C=2
- Randomly assign values : m_1=4,m_2=10
- C_1={2,4,6,3,5}, C_2={10,12,18,21,11,16}
- There are two classes that is C1 And C2 according to the distance.

K-Nearest Neighbor Example- Classification

- Ex. 1 Given:{2,4,6,5,10,12,3,20,30,11,25}, k=2
- Randomly assign numbers: 3 and 12
- K_1={2,3,4,5,6} K_2={10,11,12,20,25,30}
- Ex.2 Given: {a,c,d,g,i,j,n,p,s,w,x,z}, k=2
- Randomly assign letters : g and s
- K_1={a,c,d,g,i,j} K_2={n,p,q,s,w,x,z}
- Ex.2 Given: {10%,30%,50,40%,60%,80%,90%}, k=2
- Randomly assign letters : 30% and 80%
- K_1={10%,30%,40%,50%} K_2={60%,80%, 100%}

K-Nearest Neighbor Example

- Given: {1,2,4,7,20,10,6,3,18,5, 21,8,11,17}, C=2
- Randomly assign values : m_1=3,m_2=11

K-NN

- In pattern recognition, the ***k*-nearest neighbors algorithm** (*k*-NN) is a method for classifying objects based on closest training examples in the feature space.
- *k*-NN is a type of instance-based learning, or lazy learning where the function is only approximated locally and all computation is deferred until classification.
- The *k*-nearest neighbor algorithm is amongst the simplest of all machine learning algorithms.

Cont…

- an object is classified by a majority vote of its neighbors, with the object being assigned to the class most common amongst its *k* nearest neighbors (*k* is a positive integer, typically small).
- If *k* = 1, then the object is simply assigned to the class of its nearest neighbor.

Cont...

- The same method can be used for regression, by simply assigning the property value for the object to be the average of the values of its *k* nearest neighbors.
- It can be useful to weight the contributions of the neighbors, so that the nearer neighbors contribute more to the average than the more distant ones. (A common weighting scheme is to give each neighbor a weight of 1/*d*, where *d* is the distance to the neighbor. This scheme is a generalization of linear interpolation.)

Cont...

- The neighbors are taken from a set of objects for which the correct classification (or, in the case of regression, the value of the property) is known.
- This can be thought of as the training set for the algorithm, though no explicit training step is required.
- The *k*-nearest neighbor algorithm is sensitive to the local structure of the data.

Cont…

- In the classification phase, *k* is a user-defined constant, and an unlabelled vector (a query or test point) is classified by assigning the label which is most frequent among the *k* training samples nearest to that query point.

Drawback

- A drawback to the basic "majority voting" classification is that the classes with the more frequent examples tend to dominate the prediction of the new vector, as they tend to come up in the *k* nearest neighbors when the neighbors are computed due to their large number.
- One way to overcome this problem is to weight the classification taking into account the distance from the test point to each of its *k* nearest neighbors.

Exercises 1

A database has five transactions. Let $min\_sup = 60\%$ and $min\_conf = 80\%$.

TID	*items_bought*
T100	{M, O, N, K, E, Y}
T200	{D, O, N, K, E, Y }
T300	{M, A, K, E}
T400	{M, U, C, K, Y}
T500	{C, O, O, K, I ,E}

(a) Find all frequent itemsets using Apriori and FP-growth, respectively. Compare the efficiency of the two mining processes.

(b) List all of the *strong* association rules (with support s and confidence c) matching the following metarule, where X is a variable representing customers, and $item_i$ denotes variables representing items (e.g., "A", "B", etc.):

$$\forall x \in transaction,\ buys(X, item_1) \wedge buys(X, item_2) \Rightarrow buys(X, item_3) \quad [s, c]$$

Exercise 2

Tid	List of Items
T100	I1,I2,I5
T200	I2,I4
T300	I2,I3
T400	I1,I2,I4
T500	I1,I3
T600	I2,I3
T700	I1 ,I3
T800	I1,I2,I3,I5
T900	I1,I2, I3

- Find the frequent itemset using Apriori and FP-growth respectively. Compare the efficiency of the two mining processes.
- List all of the strong associatiojn rules (with support s and confidence c) matching the following metarule, where X is a variable representing customers, and $item_i$ denote variables represetngin items.

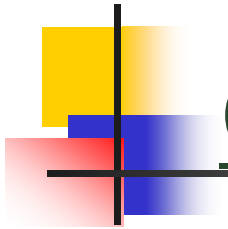
$\forall$ x $\in$ transactions, buys (X, $item_{1}$) $\wedge$ buys (X, $_{item2}$) ➔ buys(X, $item_3$) [s,c]

UNIT-IV

Data Mining

Contents

- Basic Data Mining tasks,
- Knowledge discovery in databases,
- Issues,
- OLTP systems,
- Information Retrieval,
- Dimensional Modeling,
- Web search engines,
- Data Mining Techniques



Overview of Data Mining

- Data mining definition: Data Mining is an **Extraction of interesting information from large data sources**
- The extracted information should be
  - **-Implicit (hidden)**
  - **-Non-trivial**
  - **-Previously unknown**
  - **and potentially useful**
- Query processing, simple statistics are not data mining
- **Databases + Statistics + Machine Learning = Data Mining**

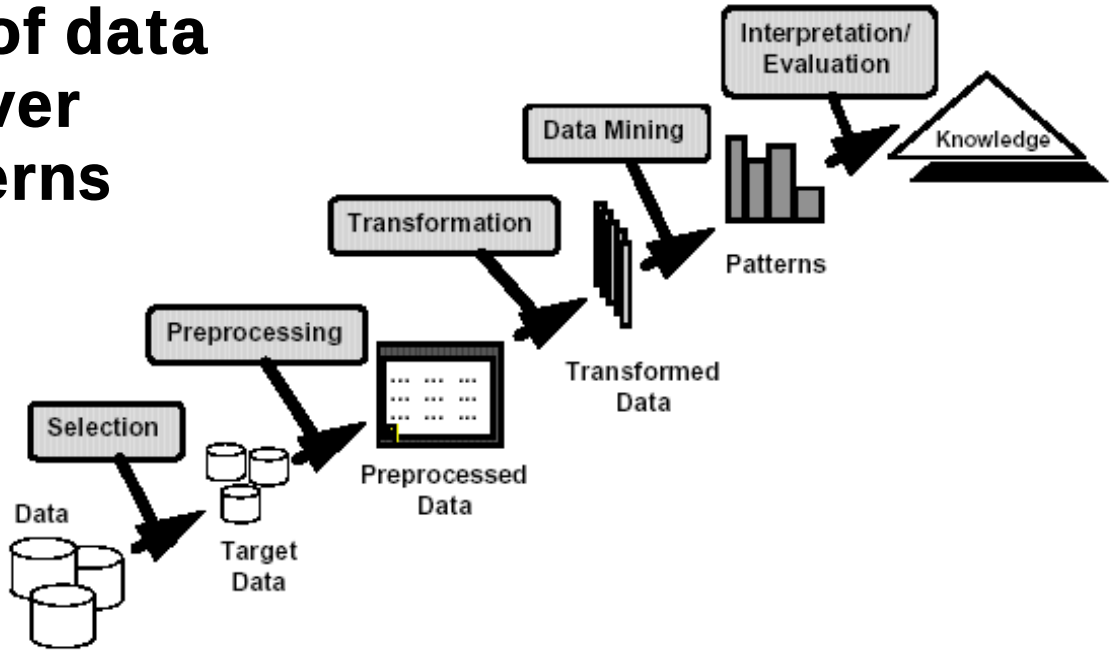
What Is Data Mining?

- Data mining (knowledge discovery in databases):
  - Extraction of interesting (non-trivial, implicit, previously unknown and potentially useful) information or patterns from data in large databases
- Alternative names and their "inside stories":
  - Knowledge discovery(mining) in databases (KDD), knowledge extraction, data/pattern analysis, data archeology, information harvesting, business intelligence, etc.
- What is not data mining?
  - (Deductive) query processing.
  - Expert systems or statistical programs

Cont...

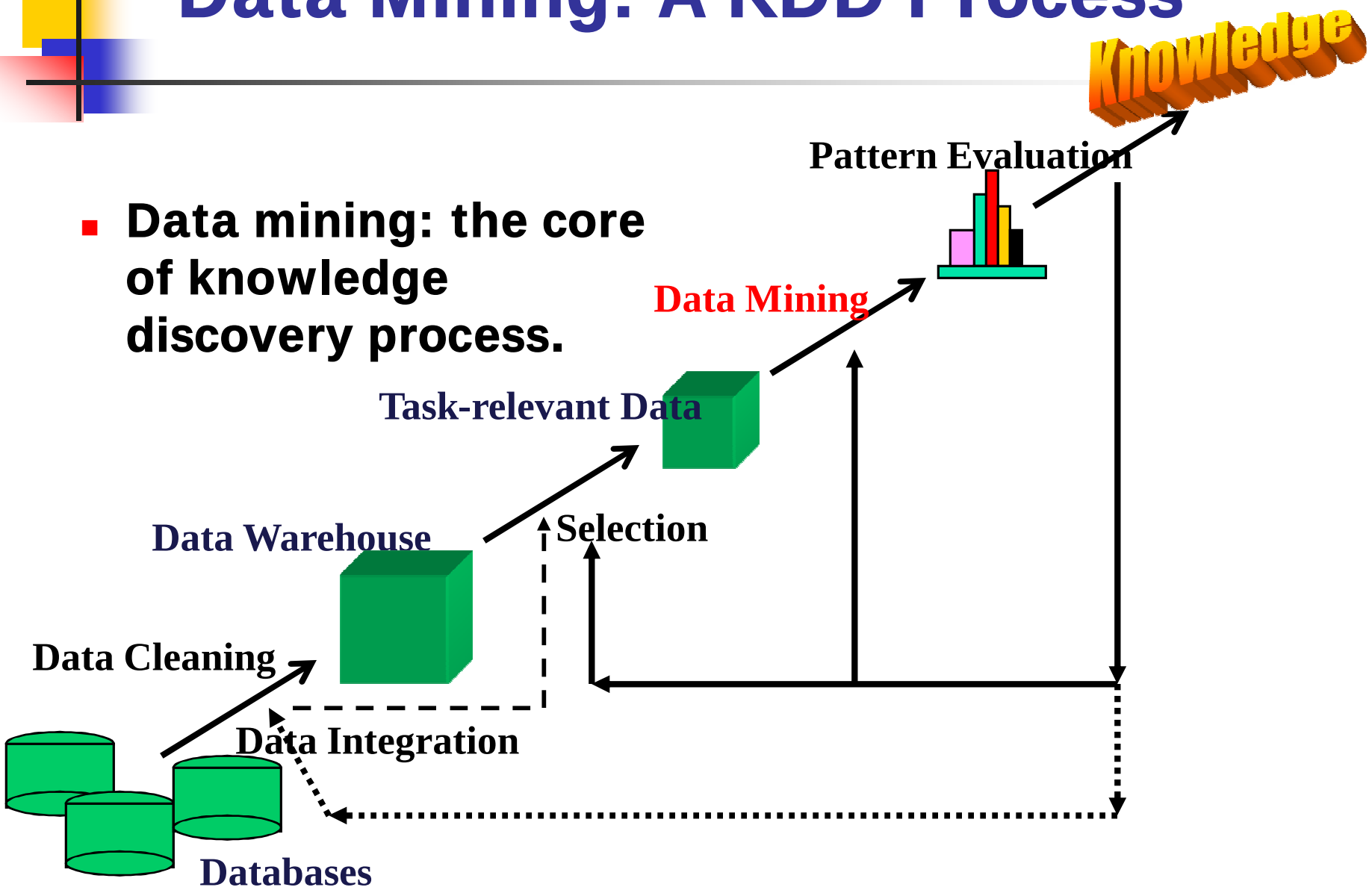
Many Definitions

- **Extraction of implicit, previously unknown and potentially useful information from data**
- **Exploration & analysis by automatic or semi-automatic means, of large quantities of data in order to discover meaningful patterns**



Data Mining: A KDD Process

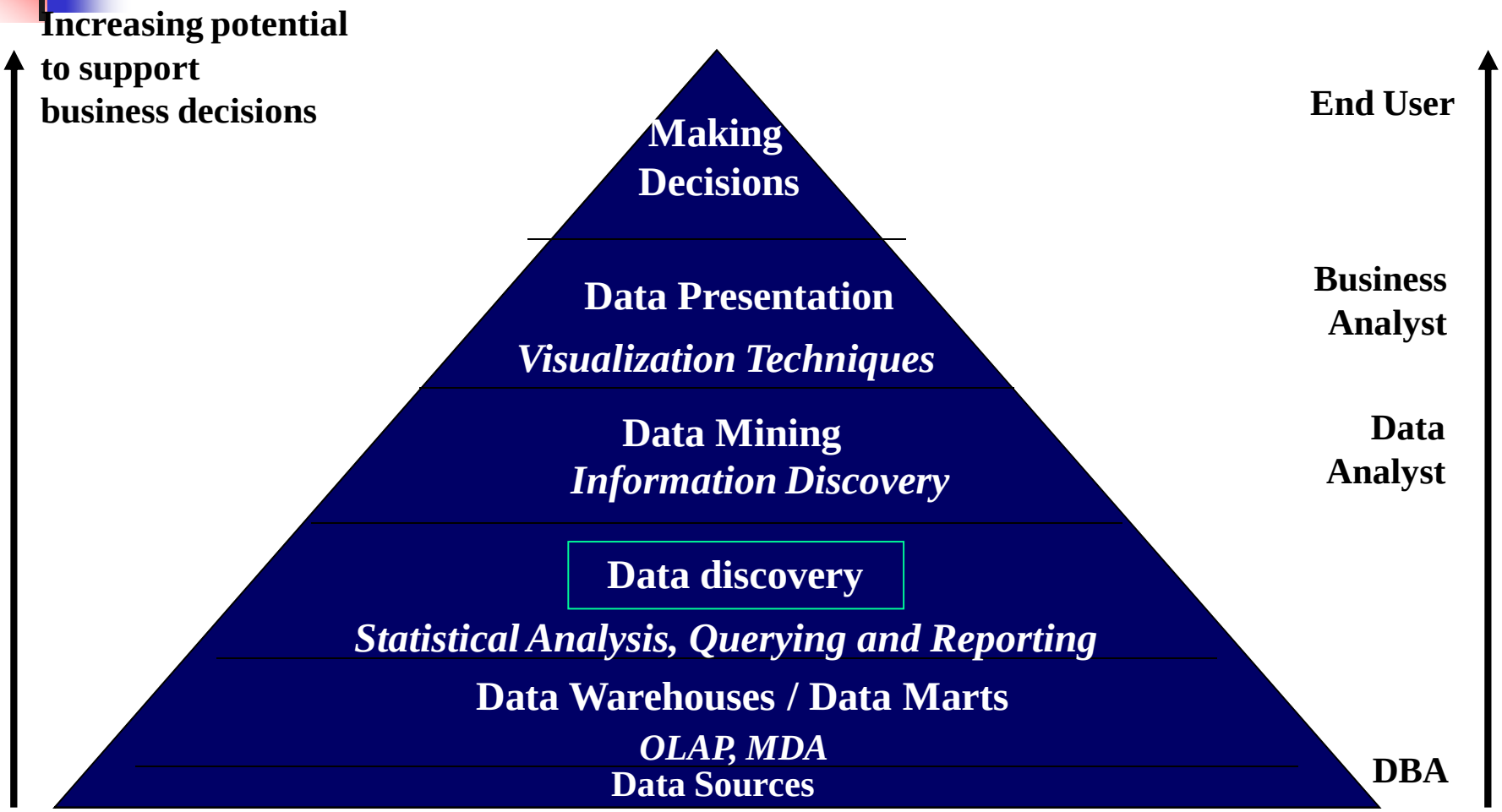
- **Data mining: the core of knowledge discovery process.**



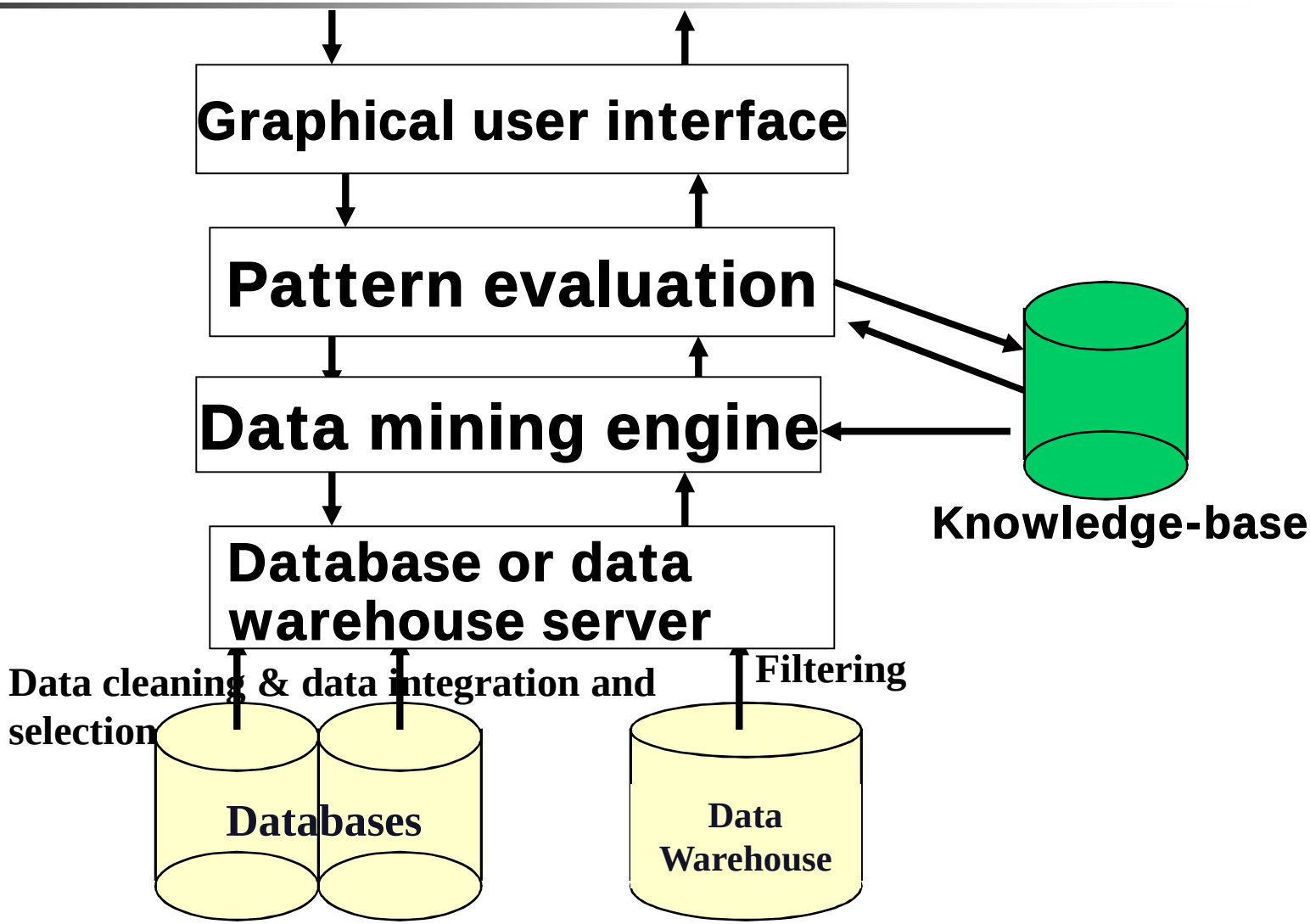
Steps of a KDD Process

- **Learning the application domain (area):**
  - relevant prior knowledge and goals of application
- Creating a target data set: data selection
- **Data cleaning** and preprocessing
- **Data reduction and transformation**:
  - Find useful features, dimensionality/variable reduction, invariant representation.
- Choosing functions of data mining
  - summarization, classification, regression, association, clustering.
- Choosing the mining algorithm(s)
- **Data mining**: search for patterns of interest
- **Pattern evaluation and knowledge presentation**
  - visualization, transformation, removing redundant patterns, etc.
- Use of discovered knowledge

Data Mining and Business Intelligence



Major Components of Data Mining System Architecture



Explanation

The major components of Data Mining system architecture is as follows.

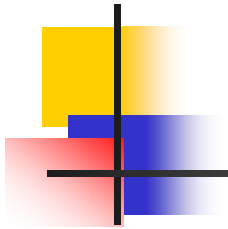
- **Database, Data Warehouse or other information Repository**
- This is a set of databases, data warehouses, spreadsheets or other kinds of information repositories
- Data cleaning and data integration techniques performed on the data.

Cont…

- **Database or data warehouse server:**
- It is responsible for fetching the relevant data based on the users data mining request.
- **Data Mining Engine:**
- This is essential to the data mining system
- It consists of **set of modules** for tasks such as
- **classification, Association and correlation analysis, Classification, prediction, cluster analysis, outlier analysis and evolution analysis**

Cont…

- **Knowledge Base:**
- This is the **domain(area) knowledge**
- It is used to guide the search of resulting patterns
- Such knowledge include concept of hierarchies used to organized attributes or attribute values in to different levels of abstraction
- Eg. Metadata



Cont....

- **Pattern Evaluation module:**
- This component gives interesting measure and interact with data mining modules to focus the search towards the interesting patterns.
- Pattern evolution module may integrated with the mining module, depending on the implementation data mining method used.

Cont…

- **Graphical User Interface:**
- This module communicate between **user** and **data mining system**
- Allowing the user to interact with the system by using a data mining query
- Providing information
- Allows the user to browse database and data warehouse schemas, evaluate mined patterns and visualize the patterns in different forms.

Data Mining Applications

- **Database analysis and decision support**
  - **Market analysis and management**
    - **target marketing, customer relation management, market basket analysis, cross selling, market segmentation**
  - **Risk analysis and management**
    - **Forecasting, customer retention, quality control, competitive analysis**
  - **Fraud detection and management**
- **Other Applications**
  - **Text mining (news group, email, documents) and Web analysis.**
  - **Intelligent query answering**
- **Sports**
  - **To analyzed game statistics (shots blocked, assists, and fouls) to gain competitive advantages.**

Market Analysis and Management

- **Data sources for analysis**
  - **Credit card transactions, loyalty cards, discount coupons, customer complaint calls, plus (public) lifestyle studies**
- **Target marketing**
  - **Find clusters of "model" customers who share the same characteristics: interest, income level, spending habits, etc.**
- **Cross-market analysis**
  - **Associations/co-relations between product sales**
  - **Prediction based on the association information**

Cont…

- **Customer profiling**
  - data mining can tell you what types of customers buy what products (clustering or classification)
- **Identifying customer requirements**
  - identifying the best products for different customers
  - use prediction to find what factors will attract new customers
- **Provides summary information**
  - various multidimensional summary reports
  - statistical summary information (data central tendency and variation)

Corporate Analysis and Risk Management

- **Finance planning and asset evaluation**
  - cash flow analysis and prediction
  - party claim analysis to evaluate assets
  - cross-sectional and time series analysis (financial-ratio, trend analysis, etc.)
- **Resource planning:**
  - summarize and compare the resources and spending
- **Competition:**
  - monitor competitors and market directions
  - group customers into classes and a class-based pricing procedure
  - set pricing strategy in a highly competitive market

Fraud Detection and Management

- **Applications**
  - **widely used in health care, retail, credit card services, telecommunications (phone card fraud), etc.**
- **Approach**
  - **use historical data to build models of false behavior and use data mining to help identify similar instances**
- **Examples**
  - **auto insurance: detect a group of people who stage accidents to collect on insurance**
  - **money laundering: detect suspicious money transactions**
  - **medical insurance: detect professional patients and ring of doctors and ring of references**

Cont...

- **Detecting inappropriate medical treatment.**

- **Detecting telephone fraud**
  - **Telephone call model: destination of the call, duration, time of day or week. Analyze patterns that deviate from an expected norm.**
  - **British Telecom identified discrete groups of callers with frequent intra-group calls, especially mobile phones, and broke a multimillion dollar fraud.**
- **Retail**
  - **Analysts estimate that 38% of retail shrink is due to dishonest employees.**

What Kinds of Data?

- Relational database
- Transactional databases
- Advanced DB and information repositories
  - Object-oriented and object-relational databases
  - Spatial databases (eg. geographic (map),computer aided design,medical and satellite image databases)
  - Time-series data(eg. Stock exchange, enventory control of perticular pattern) and temporal data (time and date related attributes)
  - Text databases and multimedia databases
  - Heterogeneous and legacy(long history) databases
  - WWW

Data Mining Techniques

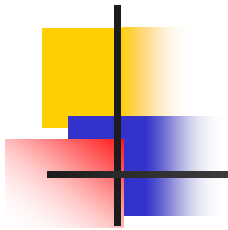
Various data mining Techniques are as below:

- **Characterization and discrimination (favoritism)**
  - Generalize, summarize, and contrast data characteristics, e.g., dry vs. wet regions
- **Association and (correlation)**
  - Multi-dimensional vs. single-dimensional association
  - Market Basket Analysis

Cont…

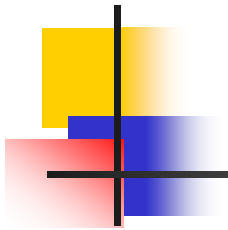
Classification and Prediction

- Classification is one of the most popular **data mining strategy.**
- **This method classifies data into a desired number of classes.**
- **Classification method** comes under **supervised learning**(we have to train data sets)
- Finding models (functions) that describe and distinguish classes or concepts for future prediction.



Cont…

- E.g., classify countries based on climate, or classify cars based on gas mileage
- Classification using decision-tree, classification rule, neural network
- **Prediction:** Predict some unknown or missing numerical values
- Prediction refers to both numeric prediction and class label prediction



Cont..

- **Cluster analysis**
  - Clustering is another strategy of data mining

    Which labels a new sample as a member of a group of similar samples and forms a cluster
  - In Clustering objects are **Clustered** or **grouped**.

Cont….

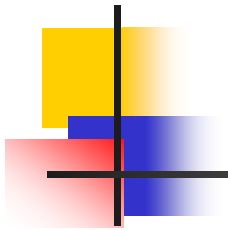
- This method is an **unsupervised way of learning**(no need of train data sets, self trained)
- Common applications of clustering include image segmentation, disease outbreaks, market segmentation etc.
- Eg. Clustering could be useful for a company to identify a group of customers based on their income, age and purchase behavior

Cont…

- **Outlier analysis**
  - **Outlier:** a data object that does not fulfill the general behavior of the data (**out of scale**)
  - It can be considered as **noise** or exception but is quite **useful in fraud detection**, rare events analysis
- **Trend and Evolution analysis**
  - Data evolution analysis describes and models regularities or **trend for objects whose behavior changes over time**

Cont...

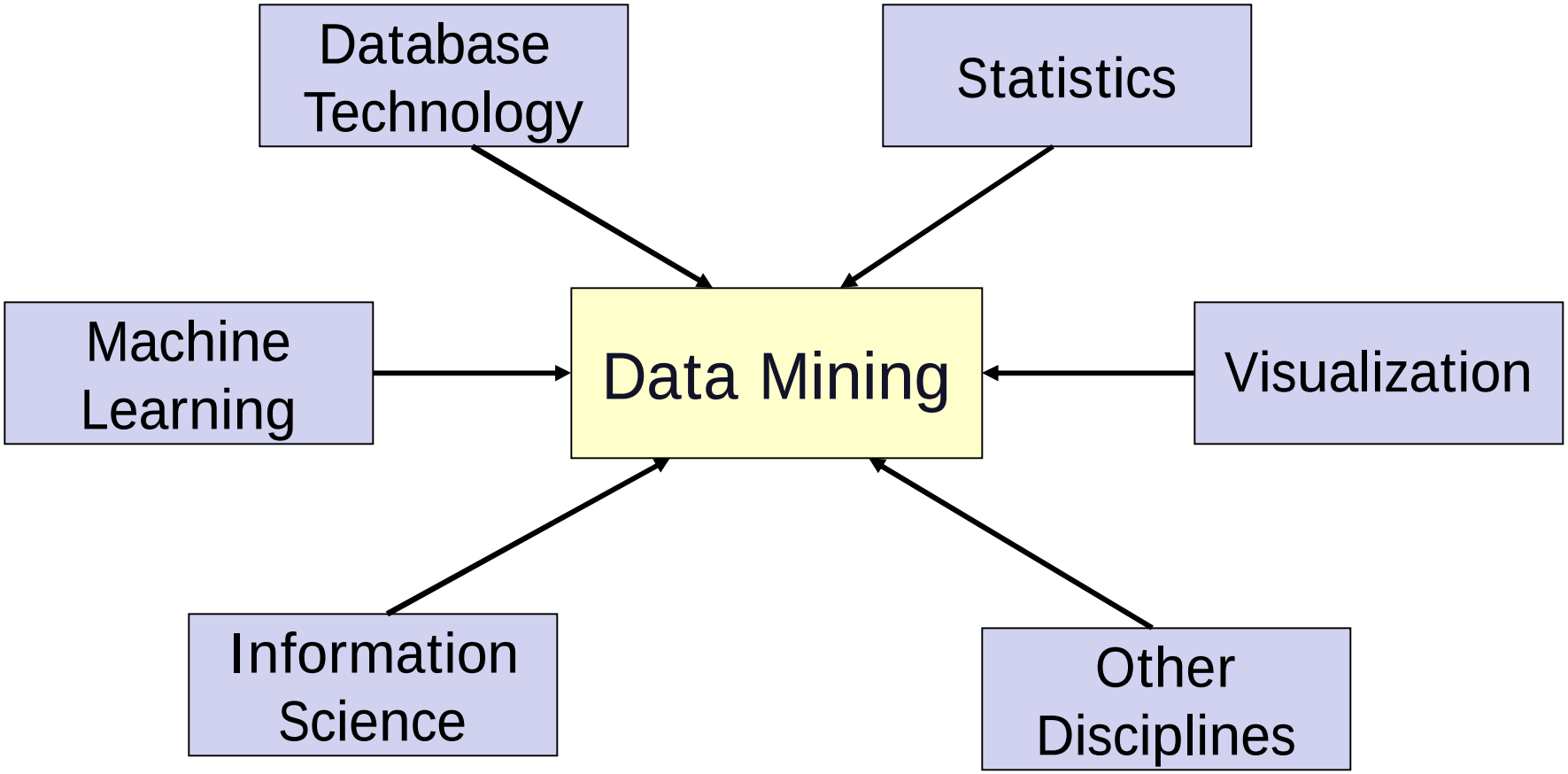
- It includes characterization, association, classification, prediction or clustering of time related data and pattern analysis.
- Trend and deviation: regression analysis
- Sequential pattern mining, periodicity analysis
- Similarity-based analysis
- Other pattern-directed or statistical analysis



Cont..

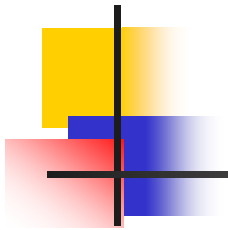
- General functionality
  - Descriptive data mining
  - Predictive data mining
- Different views, different classifications
  - Kinds of databases to be mined
  - Kinds of knowledge to be discovered
  - Kinds of techniques utilized
  - Kinds of applications adapted

Classification of Data Mining System



Explanation

- Data mining is an interdisciplinary field
- It includes database systems, statistics, machine learning, visualization and information science
- Due to this variety of disciplines, it is necessary to provide a clear classification of data mining system
- **Classification help users to distinguish between such system and identify the best match as per their need.**



Summary

- Data mining: discovering interesting patterns from large amounts of data
- A natural evolution of database technology, in great demand, with wide applications
- A KDD process includes data cleaning, data integration, data selection, transformation, data mining, pattern evaluation, and knowledge presentation
- Mining can be performed in a variety of information repositories
- Data mining functionalities: characterization, discrimination, association, classification, clustering, outlier and trend analysis, etc.
- Classification of data mining systems
- Major issues in data mining