MCS-043.

Block 3 - Unit 4.

Introduction to Data Mining.

Data mining: Data mining is the process of automatic extraction of interesting (non trivial, implicit, previously unknown and potentially useful) information or pattern from the data in large databases or in data warehouses or in flat files.

Use of data mining: Data is growing at a phenomenal rate today & the users expect more sophisticated information from this data. There is need for new techniques and tools that can automatically generate useful information and knowledge from large volumes of data. Data mining is one such technique of generating hidden information from the data.

Input for data mining is from data warehouses:

Reasons for the above is:

i) Data quality & consistency is essential for data mining: the data before loading the data, it is extracted, cleaned and transformed.

ii) Data in datawarehouses is from multiple sources: Data warehouses consists of integrated & subject oriented data.

iii) In data mining, the required data may be aggregated or summarized data. This is already there in datawarehouses.

iv) Data warehouses provide capabilities for analysing the data by OLAP operations.

Database processing Vs Data mining Processing.

Database processing	Data mining processing.
① query language for database processing is well defined	query language for datamining is poorly defined.

①	SQL is used for database processing.	there is no specific query language for data mining.
②	data used is operational data	data used is historical data.
③	output of query of database processing is precise & is the subset of the data.	output is fuzzy and it is not a subset of the data.
④	Eg: find all customers whose last name Singh, find all customers who purchased shirt.	find items that are purchased with shirts., find customers with similar buying habits.

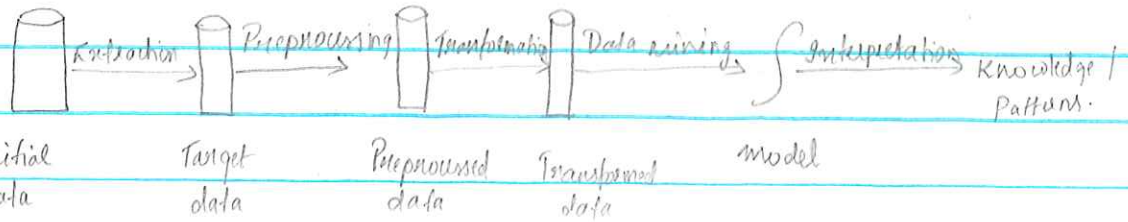
(X) Data mining Vs KDD (Knowledge Discovery in Databases)

Data mining is only one of the many steps of KDD.
KDD is the process of finding useful information, knowledge and patterns in data while data mining is the process of ~~extract~~ using algorithms to automatically extract the desired information and patterns, which are derived by the KDD process.

Steps in KDD:

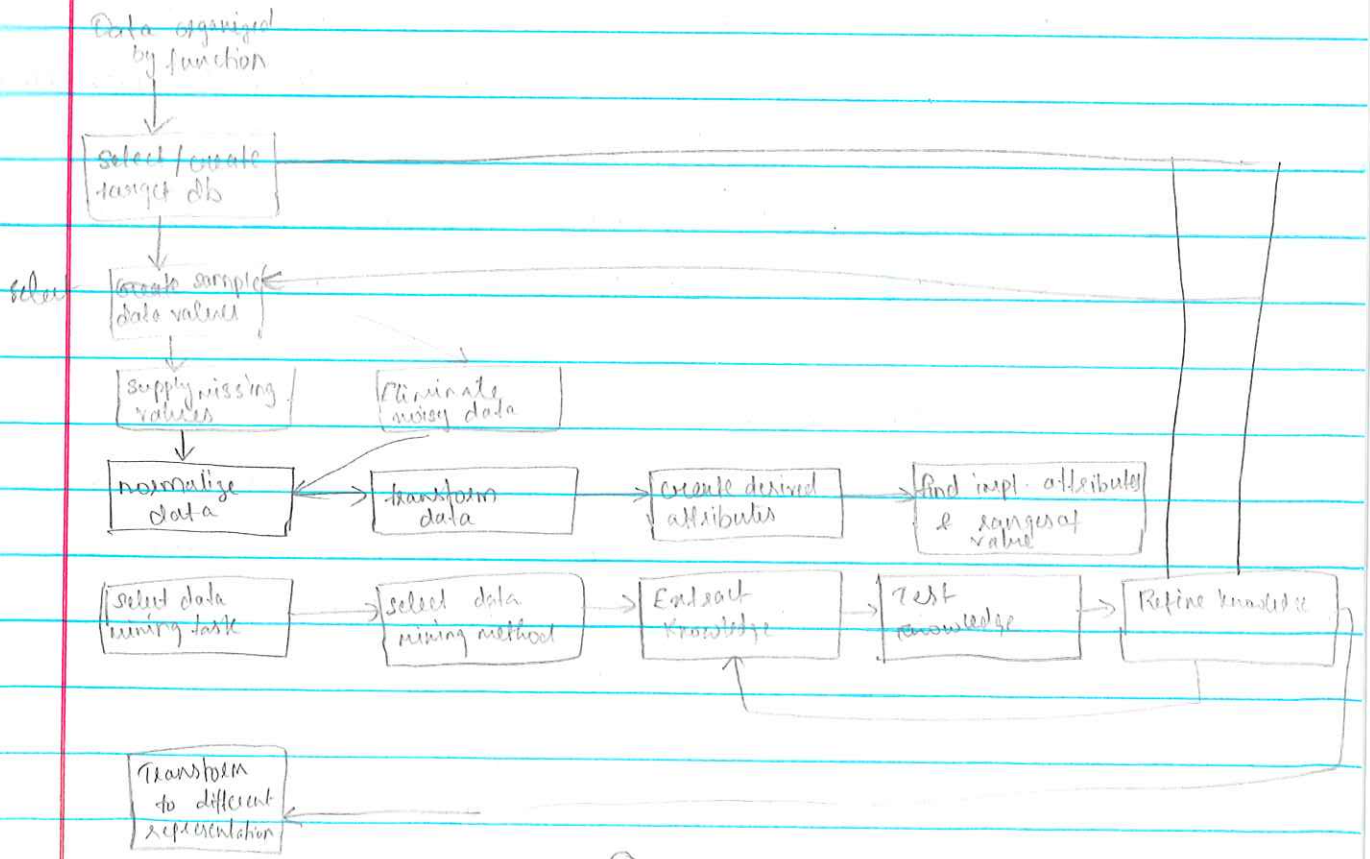
Extraction : Obtains data from various sources.
Preprocessing : It includes cleansing the data which has already been extracted by the above step.
Transformation: Data is converted into a common format, by applying some technique.
Data mining: Automatically extracts the information/pattern/knowledge.

Interpretation / Evaluation: Presents the results obtained through data mining to the users, in easily understandable & meaningful format.



Tasks in KDD:

1. Obtain information on application process:
2. Extract data set:
3. Data cleansing process
4. Data reduction & projection.
5. Select data mining task
6. Select data mining method:
7. Extracting the pattern
8. Interpretation & presentation of the pattern / model.



Approaches to data mining problems & tasks in data mining.

Approaches to data mining problems is based on type of ~~data~~ information/ knowledge to be mined.

3 approaches of data mining are:

↳ Classification ↳ clustering ↳ Association rules.

Association rule mining:

The task of association rules mining is to find certain association relationships among a set of items in the dataset/ database. In association rule mining, there are two measurements, support and confidence. Support(s) refers to the frequency of the pattern. Confidence(c) refers to the rules strength.

Definition: Let $I = \{i_1, i_2, \ldots i_m\}$ be a set of ~~items~~ literals called items. Let D be a set of transactions, where each transaction T is a set of items such that $T \subseteq I$. TID is the transaction identifier. An association rule is of the form: $X \rightarrow Y$, where $X \subset I$, $Y \subset I$ & $X \cap Y = \phi$. X is called the antecedent & Y is called the consequence of the rule.

Given a user specified minimum support & minimum confidence, the problem of association rule mining is to find all the association rules whose support and confidence is larger than the minimum support & minimum confidence.
One way to do this is Apriori Algorithm.

Apriori Algorithm.

It is based on the concept that if an itemset has minimum support, then all the subsets of the itemset will also have minimum support. An itemset with minimum support is called large itemset or frequent itemset.

It generates candidate itemsets to be counted in the pass, by using only the large itemsets in the previous pass.

It starts by finding all frequent 1-itemsets, then 2-itemsets from the 1-itemsets & so forth. During each iteration only candidates found to be frequent in the previous iteration are used to generate a new candidate set during the next iteration. The algorithm terminates when there are no frequent k-itemsets.

Notations in Apriori algorithm.

k-itemset	An itemset having k-elements
L_k	set of frequent k-itemset
C_k	set of candidate k-itemset

Apriori algorithm function takes L_{k-1} as argument and returns a superset of the set of all frequent k-itemsets.

It has 2 steps i) join step ii) Prune step.

Algorithm:

1. $k = 1$
2. find frequent set L_k from C_k of all candidate itemsets.
3. form C_{k+1} from L_k; $k = k+1$
4. Repeat 2-3 until C_k is empty.

Step 2: Scan the data set D & count each itemset in C_k, if its greater than minimum support, it is frequent.

Step 3:

- $k=1$, $C_k = 1$, frequent 1-itemset
- $k>1$, generate C_k from L_{k-1} as follows:

Join step:

C_k = k-2 way join of L_{k-1} with itself.

If both $\{a_1, \ldots a_{k-2}, a_{k-1}\}$ & $\{a_1, \ldots a_{k-2}, a_k\}$ are in L_{k-1} then add $\{a_1, \ldots a_{k-1}, a_k\}$ to C_k.

Prune step:

Remove $\{a_1, \ldots a_k\}$ if it contains a non-frequent $(k-1)$ subset.

Algorithm for association rule:

$R = \phi$

for each $I \in L$ do

for each $x \subset I$ such that $x <> \phi$ and $x <> I$ do

if support(I)/ support(x) $\geq c$ then

$R = R \cup \{x \rightarrow (I-x)\}$

Example of Apriori Algorithm.

Minimum support = 2 Minimum confidence = 70%.

ID	Itemsets
100	1, 2, 5
200	2, 4
300	2, 3
400	1, 2, 4
500	1, 3
600	1, 3
700	1, 3, 2, 5
800	1, 3
900	1, 2, 3

C_1

Itemset	Support
{1}	7
{2}	6
{3}	6
{4}	2
{5}	2

no. of itemsets in which each itemset occurs.

L_1 → take all itemset from C_1 whose support ≥ minimum support

Itemset	Support
{1}	7
{2}	6
{3}	6
{4}	2
{5}	2

C_2 : Itemset

Itemset	Support
{1, 2}	4
{1, 3}	5
{1, 4}	1
{1, 5}	2
{2, 3}	3
{2, 4}	2
{2, 5}	2
{3, 4}	0
{3, 5}	1
{4, 5}	0

L_2 :

Itemset	Support
{1, 2}	4
{1, 3}	5
{1, 5}	2
{2, 3}	3
{2, 4}	2
{2, 5}	2

C_3

Itemset	Support
{1,2,3}	2
{1,2,5}	2
{1,2,4}	1
{1,3,5}	1
{1,3,4}	0
{1,5,4}	0

L_3

Itemset	Support
{1,2,3}	2
{1,2,5}	2

C_4

Itemset	Support
{1,2,3,5}	1

Support is 1, so rejected.

Association rules

1 → 2^3	2	2/7 = 0.28	28%
2 → 1^3	2	2/6 = 0.33	33%
3 → 1^2	2	2/6 = 0.33	33%
1 → 2^5	2	2/7 = 0.28	28%
2 → 1^5	2	2/6 = 0.33	33%
5 → 1^2	2	2/2 = 1	100%

↓ Support from selected L_1 here it is L_3

↓ Support / Support in L_1

↘ convert into percentage

Clustering:

Clustering is grouping things with similar attribute values into the same group.

Given a database $D = \{t_1, t_2, \ldots t_n\}$ of tuples & an integer value k, then the clustering problem is to define a mapping where each tuple t_i is assigned to one cluster k_j, $1 \leq j \leq k$.
A cluster k_j contains those tuples mapped to it.

Clustering issues:

Outlier handling: how will the outlier be handled?
Dynamic handling: how will you handle dynamic data?
Interpreting results: how will the result be interpreted?
Evaluating results: how will the result be calculated?
Number of clusters: how many clusters will you consider for the given data?
Data to be used: whether we are dealing with quality data or noisy data? If data is noisy, how is it handled?
Scalability: whether the algorithm that is used is to be scaled for small as well as large dataset / database.

Algorithms for clustering:

Partitioning algorithm:

This algorithm classifies the n objects into k groups. Each group contains atleast one object & Each object must belong to exactly one group. A partitioning clustering algorithm normally requires users to input the desired number of clusters, k.

Some of the partitioning clustering algorithm are:

- Squared error
- k-Means.

Squared error:

It is the most commonly used partition clustering method.

Algorithm:

1. Select an initial partition with k clusters.
2. Assign each partition to its closest cluster center & compute the new cluster centers as the centroids of the clusters. Repeat this step until the cluster membership is stable.
3. Merge & split the clusters using some heuristic information, optionally repeating step 2.

Parameters used in this:

Centroid (C_m): $\sum_{i=1}^{N} t_{mi} / N$

Radius (R_m): $\sqrt{\sum_{i=1}^{N} (t_{mi} - C_m)^2 / N}$

Diameter (D_m) = $\sqrt{\sum_{i=1}^{N} \sum_{j=1}^{N} (t_{mi} - t_{mj})^2 / (N * (N-1))}$

K-means clustering:

Initially, a set of clusters is chosen in random. Then, the elements of the clusters are moved among the set of clusters until the right set is reached. Using this algorithm, high degree of similarity among elements in a cluster is obtained.

Given: a set of clusters $K_j = \{t_{i1}, t_{i2}, \ldots t_{im}\}$

cluster mean, $m_i = \frac{1}{m} (t_{i1} + t_{i2} + \ldots + t_{iM})$

Input :

D // set of elements

A // adjacency matrix showing distance between elements

k // number of desired clusters

Output :

K // set of clusters

Algorithm :

Assign initial values for means $m_1, m_2 \dots m_k$;

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.

Example :

$K = \{2, 3, 4, 10, 11, 12, 20, 25, 30\}$

$k = 2$.

$m_1 = 4, \quad m_2 = 12$

$k_1 = \{2, 3, 4\}$ $\quad k_2 = \{10, 11, 12, 20, 25, 30\}$

$m_1 = 3$ $\quad m_2 = 18$

$k_1 = \{2, 3, 4, 10\}$ $\quad k_2 = \{11, 12, 20, 25, 30\}$

$m_1 = 4.75 \ (5)$ $\quad m_2 = 19.6 \ (20)$

$k_1 = \{2, 3, 4, 10, 11, 12\}$ $\quad k_2 = \{20, 25, 30\}$

$m_1 = 7$ $\quad m_2 = 25$

$k_1 = \{2, 3, 4, 10, 11, 12\}$ $\quad k_2 = \{20, 25, 30\}$

$m_1 = 7$ $\quad m_2 = 25$

So, the desired cluster $k_1 = \{2, 3, 4, 10, 11, 12\}$

$k_2 = \{20, 25, 30\}$

Nearest neighbour clustering.

In this approach, the items are iteratively merged into existing clusters that are closest. It is an incremental method. The threshold, t, used to determine if items are added to existing clusters or a new cluster is created. This process continues until all patterns are labeled or no additional labeling occurs.

Nearest neighbour algorithm:

Input

$D = \{t_1, t_2 \cdots t_n\}$ // set of elements

A // Adjacency matrix showing distance between elements

k // number of desired clusters

Output

K // set of clusters

Algorithm

```
k1 = {t1};
K = {k1};
k = 1
for i = 2 to n do
    find the tm in some cluster km in K such that distance(ti, tm) is the smallest;
    if dis(ti, tm) <= t then
        Km = Km U ti;
    else
        k = k + 1;
        Kk = {ti};
```

Hierarchical clustering.

In this method, the clusters are created in levels and depending upon the threshold value at each level the clusters are again created.

Two types of hierarchical method:

Agglomerative approach:

→ it is a bottom up approach.

→ It begins with each tuple in a distinct cluster and successively merges clusters together until a stopping criterion is satisfied.

Divisive approach:

→ It is top down approach:

→ It begins with all tuples in a single cluster and performs splitting until a stopping criterion is met.

Hierarchical clustering algorithm is a variant of single-link, average-link & complete-link algorithms. Single-link & complete link algorithm are the most popular.

In single-link method, the distance between two clusters is minimum of the distances between all pairs of patterns drawn from the two clusters.

In complete-link method, the distance between two clusters is the maximum of pair-wise distances between patterns in the two clusters.

In either case, both two clusters are merged to form a larger cluster based on minimum distance criteria.