After Mid-Sem From peeps.

Date : __/__/__
Page : _____

* Association Rule - Frequent Pattern

$A \rightarrow B$

Let $I = \{I_1, I_2, \ldots I_m\}$ be a set of items. In transition 'T' is said to contain 'A' iff

$A \subseteq T$

An association rule $A \rightarrow B$ where $A \subset I$, $B \subset I$ and $A \cap B = \phi$ holds in transaction T.

- Support $(A \rightarrow B) = P(A \cup B)$
- Confidence $(A \rightarrow B) = P(B/A)$

$$\text{confidence}(A \rightarrow B) = \frac{\text{Support}(A \cup B)}{\text{support}(A)}$$

$$= \frac{\text{support\_count}(A \cup B)}{\text{support\_count}(A)}$$

- Closed Frequent Itemset

⇒ A item set 'X' is closed in dataset 'S' if there exists no proper super item set 'Y' such that has the same support count as 'X' in 'S'.

- Maximal Frequent Itemset

⇒ An item set 'X' is maximal in dataset 'S' if there exist no super item set 'Y' such that $X \subset Y$ (X is proper subset of Y) and 'y' is frequent in 'S'.

Q. Suppose the database has only two transactions.

$\{(a_1, a_2, a_3, \ldots a_{100}), (a_1, a_2, a_3, \ldots a_{50})\}$

min^m support = 1. Find closed and maximal item set.

Closed set = {$(a_1, a_2, a_3, \ldots a_{100}), (a_1, a_2, \ldots a_{50})$}

↓ support count = 1 ↓ support count = 2

Maximal set = {$(a_1, a_2, a_3, \ldots a_{100})$}

↓ supp_count = 1.

*. Multilevel Association Rules:-

In this items brought are referred as different level of abstractions.

eg:- Single level Association Rule

buys('x', "Computer") → buys('x', "HP Printer")

Multilevel Association Rule

buys('x', "Laptop Computer") → buys('x', "HP Printer")

*. Multidimensional Association Rule:-

⇒ If the items or attributes in ~~an~~ association rule refer only one dimension then it is single dimension association rule.

eg:- buys('x', "Computer") → buys('x', "HP Printer")

buys('x', "Laptop Computer") → buys('x', "HP Printer")

⇒ If the rule refer 2 or more dimension such as age, income, buys, etc. then it is multidimension association rule.

eg:-

age('x', "30, 31, ---39") ∧ income('x', "42k, ---48k") ⟶ buys('x', "Smart TV")

* Apriori Algorithm:-

→ It uses level-wise search where k-item set are used to explore (k+1) item set.
→ It is used for mining boolean frequent item set.
→ It uses prior knowledge of frequent item set.
→ First, the set of frequent 1-itemset is found by scanning the database to accumulate the count for each item, and collecting those item that satisfy min^m. support. The resulting set is denoted by L_1.
→ Next, L_1 is used to find L_2 and so on.
→ Finding each L_k requires a full scanning of the database

• Apriori Property:-

→ All non-empty subset of frequent itemset must also be frequent.

→ If an item 'A' is added to the item set 'I', then the resulting item set I ∪ A cannot occur more frequent than I.

→ This property belongs to the special category of property called Anti-monotone in the sense that if a set can't pass a test, all of its superset will fail in the same test as well.

- In Apriori L_{k-1} is used to find L_k for $k \geq 2$.
- It is a two step process followed by Join and Prune step (action).

i) First is the Join step:-

To find L_k, a set of candidates of k-item set by joining L_{k-1} with itself. The candidates are developed as C_k.

ii) Second is prune step:-

C_k is a superset of L_k. A scan of data to determine the count of each candidate in C_k, would result in the determination of L_k.

C_k can be used to reduce the size of C_k apriori properties is used in (k-1) itemset that is not frequent can't be a subset of frequent itemset.

Q. $T_1 \rightarrow I_1, I_2, I_5$

$T_2 \rightarrow I_2, I_4$

$T_3 \rightarrow I_2, I_3$

$T_4 \rightarrow I_1, I_2, I_4$

$T_5 \rightarrow I_1, I_3$

$T_6 \rightarrow I_2, I_3$

$T_7 \rightarrow I_1, I_3$

$T_8 \rightarrow I_1, I_2, I_3, I_5$

$T_9 \rightarrow I_1, I_2, I_3$

min^m. support = 2

min^m. confidence = 70%

$I_1 \rightarrow 6$
$I_2 \rightarrow 7$
$I_3 \rightarrow 6$
$I_4 \rightarrow 2$
$I_5 \rightarrow 2$

C_1

$\longrightarrow$

$I_1 \rightarrow 6$
$I_2 \rightarrow 7$
$I_3 \rightarrow 6$
$I_4 \rightarrow 2$
$I_5 \rightarrow 2$

L_1

$\longrightarrow$

$I_1, I_2 = 4$
$I_1, I_3 = 4$
$I_1, I_4 = 1$ X
$I_1, I_5 = 2$
$I_2, I_3 = 4$
$I_2, I_4 = 2$
$I_2, I_5 = 2$
$I_3, I_4 = 0$ X
$I_3, I_5 = 1$ X
$I_4, I_5 = 0$ X

C_2

$\downarrow$

$I_1, I_2 = 4$
$I_1, I_3 = 4$
$I_4, I_5 = 2$
$I_2, I_3 = 4$
$I_2, I_4 = 2$
$I_2, I_5 = 2$

L_2

$\longleftarrow$

$I_1, I_2, I_3 = 2$
$I_1, I_2, I_5 = 2$
$I_1, I_2, I_4 = 1$ X
$I_1, I_3, I_5 = 1$ X
$I_2, I_3, I_4 = 0$ X
$I_2, I_4, I_5 = 0$ X

C_3

$\downarrow$

→ $I_1, I_2, I_3 = 2$

$I_1, I_2, I_5 = 2$

→ $I_1, I_2, I_3, I_5 = 1$ ✗

L_3

C_4

Frequent Itemset

Date : 21/03/2023
Page :

* Generating rules from frequent itemsets:-

$$\text{Confidence}(A \rightarrow B) = \frac{\text{support\_count}(A \cup B)}{\text{support\_count}(A)}$$

For each frequent itemset, l, generate all non-empty, subsets of l.
For every non-empty subset, s of l output the rule as $l - s$ where

$$\frac{\text{support\_count}(l)}{\text{support\_count}(s)} \geq \text{min\_confidence}$$

Suppose, the data contains, frequent item-set $l = \{I_1, I_2, I_5\}$, then subset of $\{I_1, I_2\}, \{I_1, I_5\}, \{I_2, I_5\}$
$\{I_1\}\ \{I_2\}\ \{I_5\}$

$I_1 \wedge I_2 \rightarrow I_5$ conf $= \frac{2}{4} = 50\%$

$I_1 \wedge I_5 \rightarrow I_2$ conf $= \frac{2}{2} = 100\%$

$I_2 \wedge I_5 \rightarrow I_1$ conf $= \frac{2}{2} = 100\%$

$I_1 \rightarrow I_2 \wedge I_5$ conf $= \frac{2}{6} = 33\%$

* FP Growth / FP-Tree :-

Apriori Algorithm suffers from 2 main problems:-
i) It may need to generate a huge no. of candidate sets.
ii) It may need to repeatedly scan the database.

→ In FP-Growth, the first scan of the database is same as Apriori, which derives the set of frequent items and their support count.
i) Let the minimum support count is 2.
ii) The set of frequent items are sorted in the descending order of support-count. The resulting set or list is

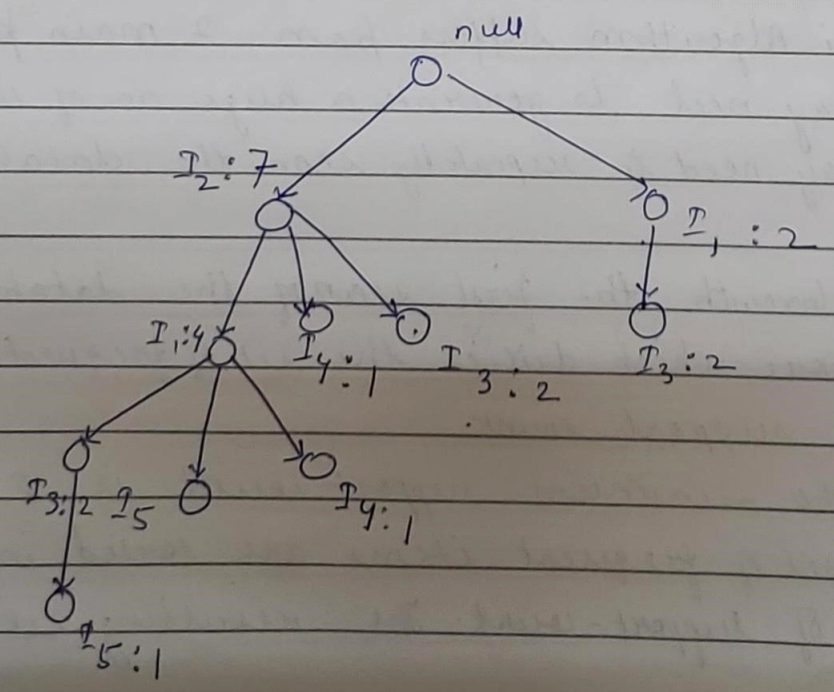
denoted as 'L'

Tid		Items
T_1	→	$I_1, I_2, I_5 \rightarrow I_2, I_1, I_5$
T_2	→	$I_2, I_4 \rightarrow I_2, I_4$
T_3	→	$I_2, I_3 \rightarrow I_2, I_3$
T_4	→	$I_1, I_2, I_4 \rightarrow I_2, I_1, I_4$
T_5	→	$I_1, I_3 \rightarrow I_1, I_3$
T_6	→	$I_2, I_3 \rightarrow I_2, I_3$
T_7	→	$I_1, I_3 \rightarrow I_1, I_3$
T_8	→	$I_1, I_2, I_3, I_5 \rightarrow I_2, I_1, I_3, I_5$
T_9	→	$I_1, I_2, I_3 \rightarrow I_2, I_1, I_3$

An FP-Tree is constructed as follows:-

i) Firstly create, the root of tree labelled with null.

ii) Scan the database D, for the 2nd time. The items in each transaction are processed in the 'L' order i.e. sorted according to increasing support count and the branch is created for each transaction.



Date : / /
Page :

* An FP Tree is mined as follows -

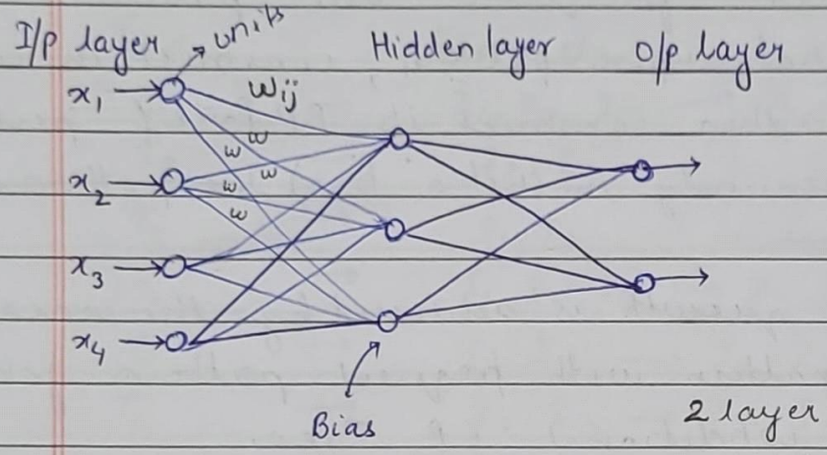
→ Start from each frequent one pattern (as an initial suffix pattern, construct its conditional pattern ways then construct its FP Tree & perform mining recursively in such a tree) ~~The pattern growth is achieve~~

The pattern growth is achieved by the concatenation of suffix pattern with frequent pattern generated from the conditional FP Tree

Item	Conditional Pattern Base	Conditional FP Tree	Frequent pattern generated
I_5	$\{I_2, I_1 : 1\}\{I_2, I_1, I_3 : 1\}$	$\{I_2 : 2, I_1 : 2\}$	$\{I_2, I_5 : 2\}\{I_1, I_5 : 2\}$ $\{I_2, I_1, I_5 : 2\}$
I_4	$\{I_2 : 1\}\{I_2, I_1 : 1\}$	$\{I_2 : 2\}$	$\{I_2, I_4 : 2\}$
I_3	$\{I_2, I_1 : 2\}\{I_2 : 2\}\{I_1 : 2\}$	$\{I_2 : 4, I_1 : 2\}\{I_1 : 2\}$	$\{I_2, I_3 : 4\}\{I_1, I_3 : 4\}$ $\{I_2, I_1, I_3 : 2\}$
I_1	$\{I_2 : 4\}$	$\{I_2 : 4\}$	$\{I_2, I_1 : 4\}$

Date 11/04/2023
Page :

Neural Networks :-



Neural Network is a set of connected input/output units in which each connection has a weight associated with it.
During the learning phase, the network learns by adjusting the weight associated with it so as to predict the correct class label of the input tuples.

Neural Network learning is also referred to as 'Connectionist Learning' due to the connection between units.
Neural Network involves long training time and are therefore more suitable for applications where this is feasible.

The most popular neural network algorithm is 'Back Propagation'.
The Back Propagation algorithm performs learning on a 'multilayer feedforward neural network'.

It iteratively learns a set of weights for prediction of class label of tuples.

A multilayer feedforward neural network consists of an input layer, one or more hidden layers and an output layer.
Each layer is made up of units.

i) The input to the network corresponds to attribute measured for each training tuple.
ii) The inputs are fed simultaneously into the units making up the input layer.

* BACK PROPAGATION:—

Back Propagation learns by iteratively processing a dataset of training tuples, comparing the network prediction of each tuple with the actual known target value.

For each training tuple, the weights are modified so as to minimize the mean-squared error between the network prediction and the actual known target value.
These modification are made in the backward direction i.e. from the output layer to each hidden layer upto the first hidden layer, hence it is called Back Propagation.

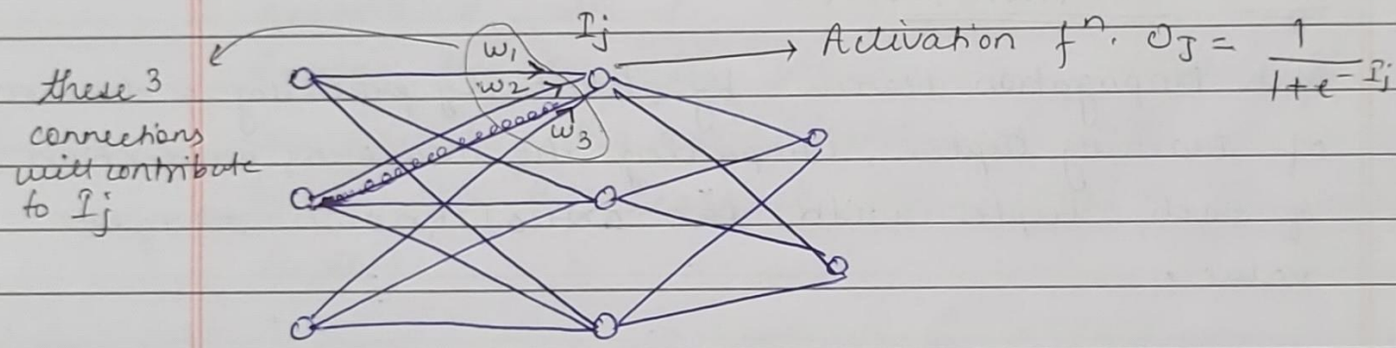
The net $$I_j = \sum w_{ij} O_i + \theta_j$$

In this w_{ij} is the weight of the connection from unit 'i' in the previous layer to unit 'j'. 'O_i' is the o/p of the unit i from the previous layer. θ_j is the bias of the unit.

It is used to vary the activity of the unit.

Each unit in the hidden layer and the output layer takes its next input and then applies an activation function to it. The sigmoid function is generally used for this purpose, given the net input I_j to unit j, then O_J, the output of unit j is computed as :—

$$O_J = \frac{1}{1 + e^{-I_J}}$$



This fn. is also referred to as 'Squashing fn.' because it maps large input domains into a smaller rng from 0 to 1.

Error is calculated as follows: —

$$Err_J = O_J (1 - O_J)(T_J - O_J)$$ Last layer

$$Err_J = O_J (1 - O_J) \sum Err_K \, W_{JK}$$ Hidden layer

W_{JK} = weight of the connection from unit J to unit K
T_J = known target value

Date : __/__/__
Page : _____

The weights and bias are updated to reflect the propagated error.

~~tt~~

weights are updated by the following eqn:-

$$\Delta w_{ij} = l \, Err_j \, O_i$$

$$w_{ij} = w_{ij} - \Delta w_{ij}$$

where l is the learning rate that typically varies in b/w 0.0 to 1

Bias are updated by the following eqn:-

$$\Delta \theta_j = (l) \, Err_j$$

$$\theta_j = \theta_j - \Delta \theta_j$$

* Chain Rule:-

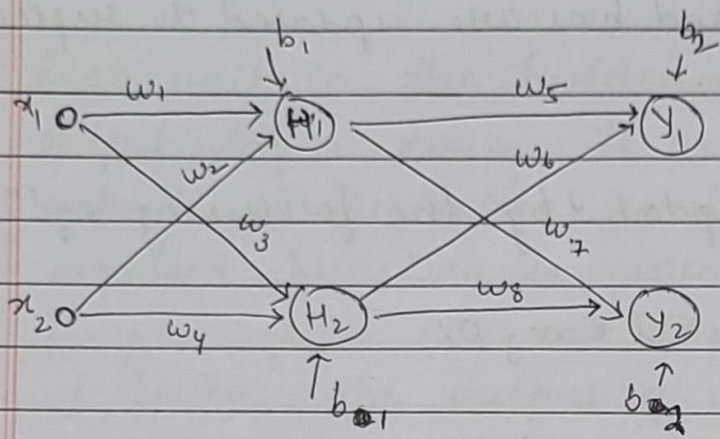
$$y = f(x) = \frac{1}{1+e^{-x}}$$

$$f'(x) = \frac{e^{-x}}{(1+e^{-x})^2}$$

$$f(x)(1-f(x)) = \frac{e^{-x}}{(1+e^{-x})^2}$$

$$\boxed{f'(x) = f(x)(1-f(x))}$$

* Partial Derivative :-



$x_1 = 0.05$	$b_1 = 0.35$	$T_1 = 0.01$
$x_2 = 0.10$	$b_2 = 0.6$	$T_2 = 0.99$
$w_1 = 0.15$	$w_5 = 0.4$	
$w_2 = 0.2$	$w_6 = 0.45$	
$w_3 = 0.25$	$w_7 = 0.5$	
$w_4 = 0.3$	$w_8 = 0.55$	

Sol^n.

$$H_{1(in)} = x_1 w_1 + x_2 w_2 + b_1 \simeq 0.377$$

$$H_{1(out)} = \frac{1}{1 + e^{-H_{1(in)}}} \simeq 0.5932$$

Similarly,

$$H_{2(out)} = 0.596$$

$$y_{1(in)} = H_1 w_5 + H_2 w_6 + b_2$$

(out) (out)

$$y_{1(out)} = \frac{1}{1 + e^{-y_{(in)}}}$$

$$E_{Total} = \frac{1}{2} \sum^{w} (Target - Actual\ o/p)^2$$

Assi 1 → Diff. b/w OLTP & OLAP systems. → Monday (17th Apr)

Date : / /
Page :

$$= \frac{1}{2}(T_1 - Y_{1out})^2 + \frac{1}{2}(T_2 - Y_{2out})^2$$

$$\Sigma \text{ Total} = E_1 + E_2$$

Backpropagating from here :—

$$\frac{\partial E_{Tot}}{\partial E_5} = \frac{\partial E_1}{\partial w_5} + \frac{\partial E_2}{\partial w_5}$$

$$\frac{\partial E_2}{\partial w_5} = \frac{\partial E_2}{\partial y_1} \times \frac{\partial y_1}{\partial w_5}$$

$$\frac{\partial E_1}{\partial w_5} = \frac{\partial E_1}{\partial y_{1(out)}} \times \frac{\partial y_{1(out)}}{\partial y_{1(in)}} \times \frac{\partial y_{1(in)}}{\partial w_5}$$

$$= (y_{1(out)} - T_1) \left[y_{1(out)} \times (1 - y_{1(out)}) \right] H_{1(out)}$$

$$\frac{\partial E_{total}}{\partial w_1} = \frac{\partial E_1}{\partial w_1} + \frac{\partial E_2}{\partial w_1}$$

$$\frac{\partial E_1}{\partial w_1} = \frac{\partial E_1}{\partial y_{1(out)}} \times \frac{\partial y_{1(out)}}{\partial y_{1(in)}} \times \frac{\partial y_{1(in)}}{\partial H_{1(out)}} \times \frac{\partial H_{1(out)}}{\partial H_{1(in)}} \times \frac{\partial H_{1(in)}}{\partial w_1}$$