11/4/25

TId	Items
1	{a, b, d, e}
2	{b, c, d}
3	{a, b, d, e}
4	{a, c, d, e}
5	{b, c, d, e}
6	{b, d, e}
7	{c, d}
8	{a, b, c}
9	{a, d, e}
10	{b, d}

→ find all frequent itemsets using apriori algo. min-support = 2

→ find all the association rules corresponding to frequent itemsets identified in first part with given confidence % as 60%.

→ Draw a lattices structure to represent all the association rule corresponding to frequent itemsets in ① and use confidence to identify all the association rules not satisfying ~~condition~~ confidence % 60%.

1 itemset:

itemset	freq
{a}	5
{b}	6
{c}	5
{d}	8
{e}	6

2 itemset:

itemset	freq	itemset	freq
{a, b}	3	{a, d}	4
~~{a, c}~~	~~2~~	{a, e}	4

{b, c} 3 {c, d} 4

{b, d} 5 {e, e} 2

{b, e} 3 {d, e} 6

~~d, c~~

3 itemset:

{a, b, d} 2 {b, d, e} 3

{a, b, e} 2 {c, d, e} 2

{a, d, e} 4 ~~{a, c, d} 1~~

{b, c, d} 2 ~~{a, c, e} 1~~

~~{b, c, e} 1~~ ~~{a, b, c} 1~~

4 itemset:

~~a, b, c, d {a, b, c e}~~

~~{a, b, c, d}~~

~~{a, b, c, e}~~

{a, b, d, e} → 2 ⇒ frequent set is

~~{a, d, c, e}~~ abde

~~{b, c, d, e}~~

⇒ How to find association Rules now?

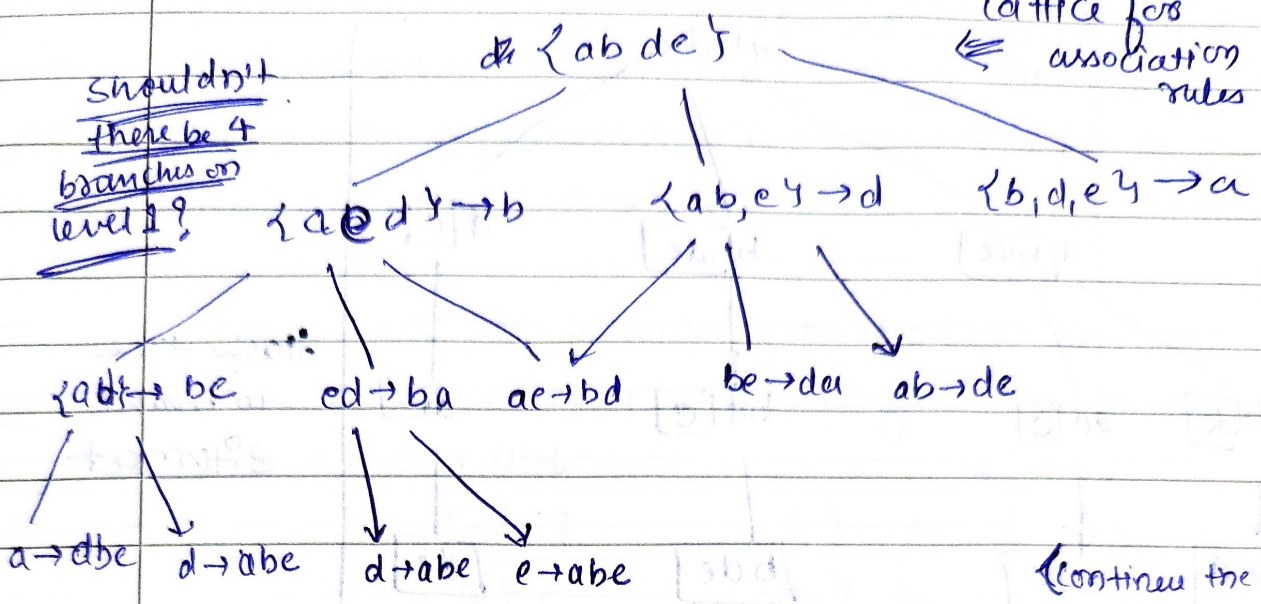
We can partition abde in anyway.

one useful way to partition is using lattice structure. (as drawn on next page)

$^5C_3 = \frac{5 \times 4}{2}$

$^5C_4 = \frac{5!}{}$

$^5C_3 = \frac{5!}{3!2!} = \frac{5 \times 4}{2} = 10$



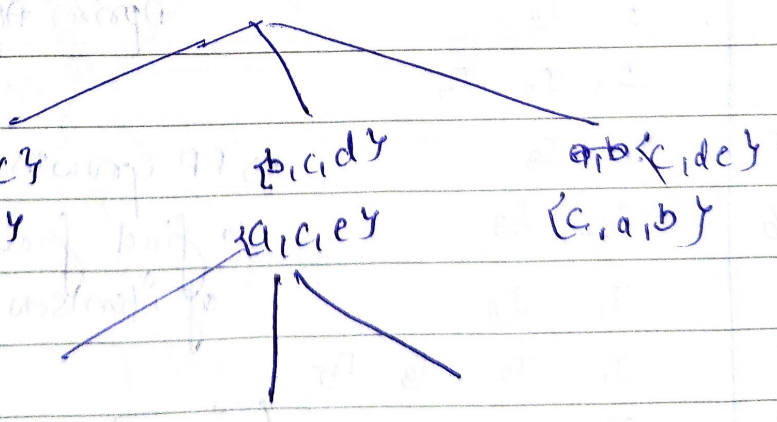
(X → Y - X)

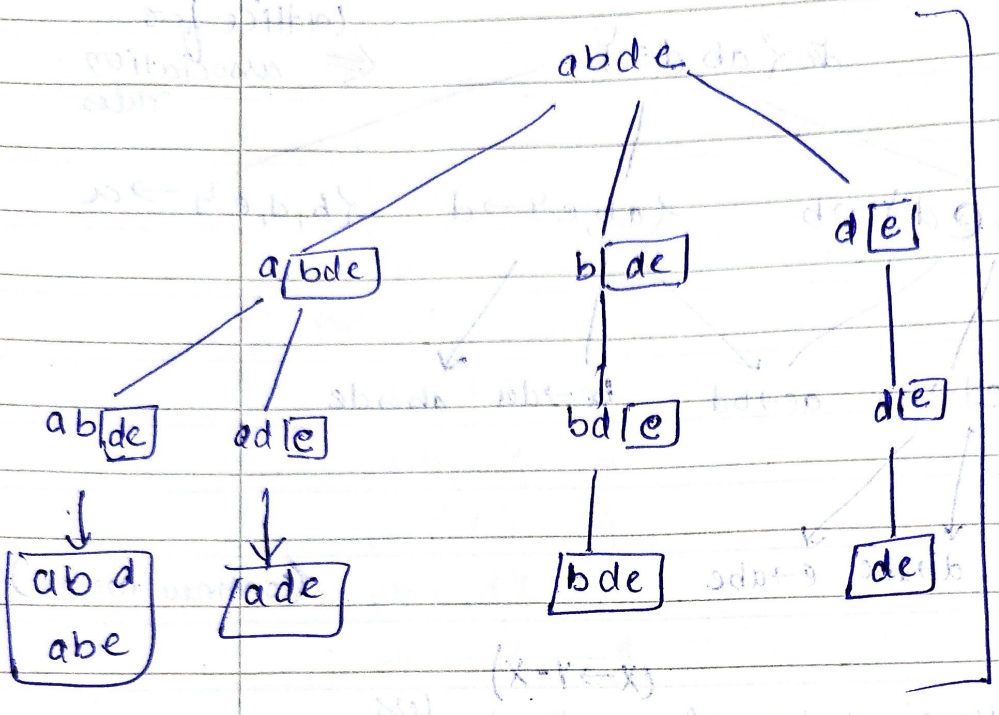
- Now if confidence value of a rule is ~~more~~ less than given threshold confidence value than any subset X' of X will also not satisfy the criteria

Hash Base Support counting method

- make hash tree for candidate itemset

ab
ac
ad
ae
bc
bd
be
cd
ce
de





Hash tree to make 3° itemset

FP Growth (Frequent Pattern Growth)

8 | 4 | 25

min support = 2

TID	items
T1	I_1 I_2 I_5
T2	I_2 I_4
T3	I_2 I_3
T4	I_1 I_2 I_4
T5	I_1 I_3
T6	I_2 I_3
T7	I_1 I_3
T8	I_1 I_2 I_3 I_5
T9	I_1 I_2 I_3

⇒ need to scan table for counting support of itemset → drawback of Apriori Algo.

⇒ FP growth → popular algo to find frequent pattern of itemsets.

Step 1 Scan the dataset & find support for all 1-itemset to find all frequent one itemset (Same as Apriori Algo)

from Step 1:

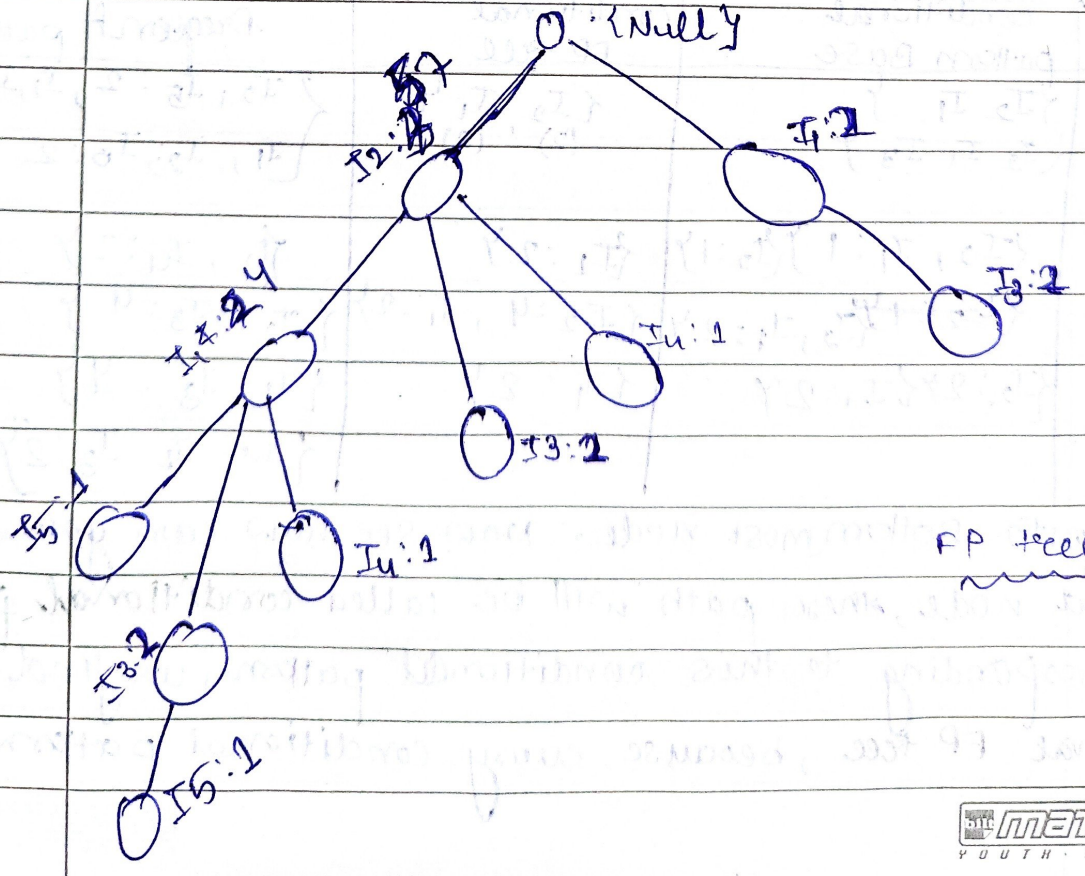
	Support count		Support count
{I1}	6	{I4}	2
{I2}	7	{I5}	2
{I3}	6		

Step 2 Sort list obtained from step 1 as per their support values (in Decreasing Order). This is called L order list.

{I2} 7	{I4} 2
{I1} 6	{I5} 2
{I3} 6	

L list

Step 3 Prepare FP tree using L order list. ~~Consi~~ consider null as root node & list all transactions as a branch in FP tree as per the order of the DB ~~L order list~~.



Instead of seeing whole dataset as unit, divide it into subsets & find frequent itemset of each subset (main idea for FP tree)

⇒ To make the tree increment frequency of each item when a pattern can be traced on a branch else create a new branch.

⇒ following data structure is maintained:

items	support count	Links
I1	7	↓
I2	6	will include pointer
I3	6	to the respective item
I4	2	in tree (multiple pointers
I5	2	if item occurs in more than 1 node)

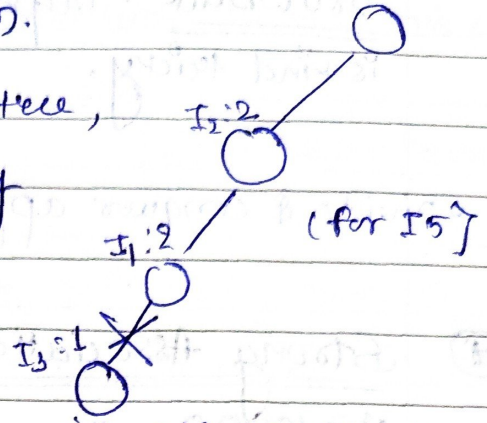
finding pattern using FP tree: start from bottom of the FP tree

items	conditional pattern Base	conditional FP tree	frequent pattern
{I5}	{I2 I1} {I2 I1 I3}	{I2 (2), I1 (2)}	{I2, I5 : 2, I1, I5 : 2,} {I1, I2, I5 : 2}
{I4}	{I2, I1 : 1}{I2 : 1}	{I2 : 2}	{I2, I4 : 2}
{I3}	{I2, I1 : 2} {I2 : 2}{I1 : 2}	{I2 : 4, I1 : 2} {I1 : 2}	{I2 I3 : 4} {I1 I3 : 4} {I2 I1 I3 : 2}

Start with Bottom most node. Now see how can you reach that node, these path will be called conditional pattern base. Corresponding to this conditional pattern, we find conditional FP tree, because every conditional pattern base

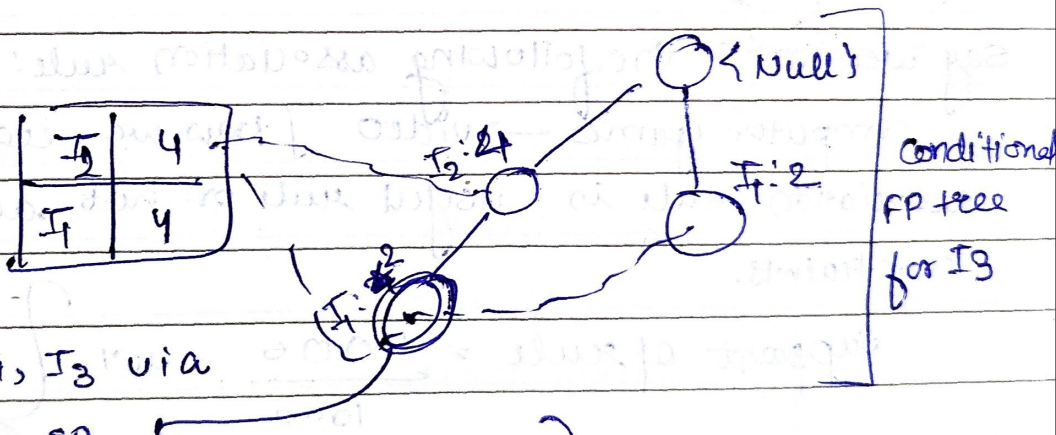
is considered as a partition.

Now from conditional FP tree, delete the nodes having support less than min. support count



concatenate the itemset with all combinations of conditional FP tree node.

Clearly not need to scan again for support values



we reach I_1, I_3 via this I_1, so it will be 4 (adding 2 from this)

table entry for I_1:

$\{I_1\}$	$\{I_2 : 4\}$	$\{I_2 : 4\}$ $\{I_2, I_1 : 4\}$

Similarly find entries for all itemsets.

⇒ The frequent itemsets obtained from the table are the same as ones found by Apriori Algorithm.

⇒ This method lists all freq. itemsets in the dataset.

Drawback : Preparing FP tree with large dataset is kind tricky.

- Divide & conquer approach is used.

(a) Strong Association Rule

N = 10000.

Computer Games = 6000

Video = 7500 · Computer Games & Videos = 4000

Support = 40% & confidence = 66%] Threshold values

Say we write the following association rule:

Computer games → video] Now we need to find if this association rule is useful rule or not according to given constraints.

Support of rule = $\frac{4000}{1000}$ = 40%.

confidence of Rule = $\frac{4000}{6000}$ = 66%. } Satisfies the constraints, hence it is a useful rule

If videos are purchased alone → their probability is 75% but if they are purchased with computer games the confidence is 66% ⇒ Negative impact of buying video with computer games. Hence, only support & confidence are not ~~not~~ useful. We need association value to find items are positively associated or negatively

associated with each other. One measure to find this out is lift measure.

$$\text{lift}(A, B) = \frac{P(A \cup B)}{P(A) \cdot P(B)} \quad (A, B \rightarrow \text{two Itemset})$$

If lift(A, B) = 1 ⇒ Itemsets are Independent
> 1 ⇒ Positive Association
< 1 ⇒ Negative association.

computing lift(computer games, video) = $\frac{4000}{\left(\frac{6000}{10000}\right)\left(\frac{7500}{10000}\right)}$

$$= \frac{40}{60 \times 75} = 0.88$$

This is less than 1, this means negative association b/w items.

Using χ^2 test we can find association here also.

contingency table:

	computer games	Computer games	
video	4000 (4500)	3500 (3000)	7500
videos	2000 (1500)	500 (1000)	2500
	6000	4000	10000

$$\chi^2 = \sum \frac{(\text{observed} - \text{Expected})^2}{\text{Expected}}$$

$$= \frac{(4000-4500)^2}{4500} + \frac{(3500-3000)^2}{3000} + \frac{(500)^2}{1500} + \frac{(500)^2}{1000}$$

$$= \frac{(500)^2}{1500}\left(\frac{1}{3} + \frac{1}{2} + 1\right) + \frac{500 \times 500}{1000}$$

$$= \frac{5 \times 500}{6}\left(\frac{2+3+6}{6}\right) + 50 = \frac{650}{3} = 216.667$$

$$= \frac{500 \times 11}{6} + 50 = 555.55$$

χ^2 test consider hypothesis that itemsets are independent

Eg 2 $N = 10000$

$n(CG) = 6000$ $\quad n(GG \& V) = 4000$

$n(V) = 7000$ $\quad n(V \& D) = 5500$

$n(D) = 8000$ $\quad n(CG \& D) = 5200$

min. support = 40%, min conf = 66%.

find DVD is likely to be bought with videos or computer games.

$$\text{Lift}(D, V) = \frac{5500/10000}{\frac{7000}{10000} \times \frac{8000}{10000}} = \frac{5500 \times 10000}{7000 \times 8000} = 0.982$$

$$\text{Lift}(D, CG) = \frac{4000 \times 10000}{8000 \times 6000} = 0.833 < \frac{5200 \times 1000}{8000 \times 6000} = 1$$

∴ DVD is more likely to be bought with videos.

Cluster Analysis

15/4/25

Clustering: Grouping of data objects
→ unsupervised learning algorithm.

⇒ Classification Methods (or Supervised learning Algos) are called as learning by examples (because we have labeled data)

⇒ Clustering methods are called learning by observations.

⇒ Data objects in same cluster must be very similar to each other & data objects in different clusters must be very dissimilar.

⇒ Clustering is also used as data preprocessing method. (Summarization of data, Data compression,

Vector Quantization → study

Good Quality cluster → similarity of data objects within cluster should be very high.

cohesive → high intra class similarity
Distinctive → low inter class similarity

What method to use to judge quality of a cluster?

4 types of clustering algo ⇒

① Partitioning Approach: Partitions are made into the dataset.

② Hierarchical Approach: Need to have some hierarchical relationship b/w clusters.

③ Density Based Clustering: Based on density in particular area

④

K-Mean clustering Algo (Partitioning Algo)

Partition dataset into K no. of clusters such that sum of squared distances is minimized.

$$E = \sum_{k=1}^{K} \sum_{p \in C_i} (p - c_i)^2$$

(c_i = centroid of cluster)

22/4/25

Example for K Means Clustering Algo (K=2 given)

1) Select 2 random ~~points~~ data points.
2) Compute distance of rest of the points with previously choosen points. Cluster points acc. to distance from initial two points (keep it in cluster of closer point among the two choosen points). following this step for all points generates two clusters.
3) Now compute mean value for each cluster.
4) These means will be new centroids & corresponding to these points you need to recluster the data points again (as told in step 2).

Keep repeating steps 2 to 4 till
(1) No change in cluster points
OR (2) No change in mean value.

- for measuring distance any measure can be considered like Euclidean Distance or Manhattan Distance etc.

- Data objects gets clustered around the central data object. (This is the major intuition of K means clustering algo).

Drawbacks: (1) Applicable only to objects in continous n dimensional space (what about nominal attributes, this algo won't work there).

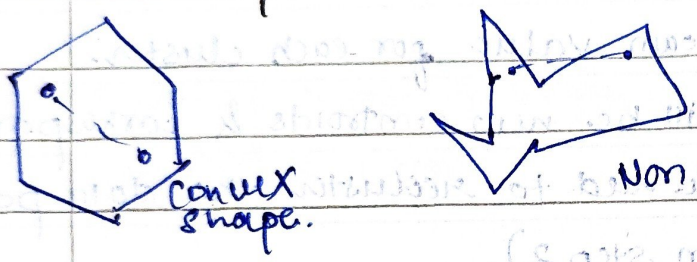
K-mode: Works for nominal or categorical attributes. Instead of calculating mean, calculate mode value,

rest of the algo is same as K means clustering)

2) Very sensitive towards outliers.

3) Not suitable to discover clusters with non convex shapes.

If we draw line to connect two points within a shape then that line must also lie within the shape ⇒ such shapes are called convex shapes. Otherwise called non convex shapes.



$$E = \sum_{i=1}^{k} \sum_{p \in C_i} (p - c_i)^2$$

Formula to compute quality of a cluster. (Within cluster Variance)

If E is low → good quality cluster.

K Medoid: Instead of mean of cluster, we find the best representative element of the cluster and that element is considered medoid element.

Distance b/w central element & all other elements is min.

PAM method: uses medoid method (Partition Around Medoid) : Starts from an initial set of medoids & iteratively replaces one of the mediods by one of the non mediods if it improves the total distance of resulting clustering

Example

	x_1	x_2
O_1	2	6
O_2	3	4 (A)
O_3	3	8
O_4	4	7
O_5	6	2
O_6	6	4
O_7	7	3
O_8	7	4 (B)
O_9	8	5
O_{10}	7	6

→ Use manhattan Distance as similarity measure

→ initial points

points	d(A)	d(B)	Cluster
O_1	3	7	
O_3	4	8	
O_4	4	6	
O_5	5	3	
O_6	3	1	
O_7	5	1	
O_9	6	2	
O_{10}	6	2	

$C_1 = \{O_2, O_1, O_3, O_4\}$ $C_2 = \{O_8, O_5, O_6, O_7, O_9, O_{10}\}$

Total Error $= E_1 + E_2$

Absolute Error : $E_1 = (O_1 - O_2) + (O_3 - O_2) + (O_4 - O_2)$

$= 3 + 4 + 4 = 11$

$E_2 = 3 + 1 + 1 + 2 + 2 = 9$

$E = E_1 + E_2 = 11 + 9 = \underline{20}$

Now acc. to PAM, lets replace O_8 by (say) O_7.

Now new cluster representative of C_2 is O_7.

13/1/24 Distributed System Lab

→ pipes (IPC)

⇒ Now recluster the data points with O_7 & O_2 as cluster points. We'll still get the same cluster structure

points	$d(O_2)$	$d(O_7)$	
O_1	3	8	
O_3	4	9	
O_4	4	7	
O_5	5	2	$C_1 = \{O_2, O_1, O_3, O_4\}$
O_6	3	2	$C_2 = \{O_7, O_5, O_6, O_8$
O_8	4	1	$O_9, O_{10}\}$
O_9	6	3	
O_{10}	6	3	

⇒ Now recalculate error

$E_1 = 3+4+4 = 11$

$E_2 = 2+2+1+3+3 = 11$

$E = 22$ → more than old error

If new error > old error ⇒ Don't accept else the cluster representative is selected.

Stopping condition:

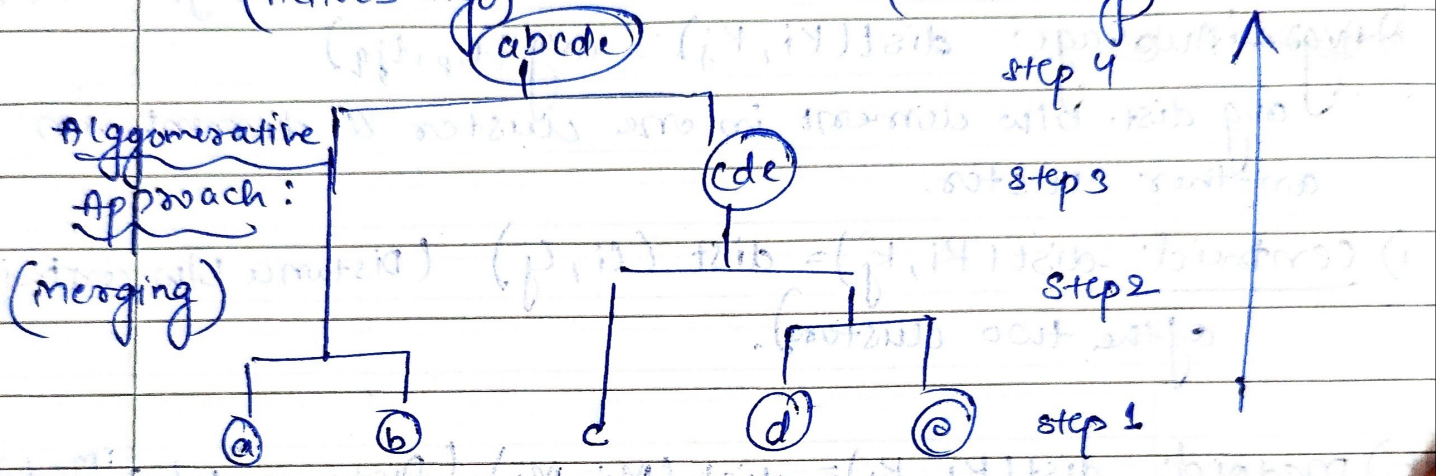
⇒ PAM works well for small dataset but not for large one.

Hierarchical Clustering

Two types

(1) Agglomerative (Bottom to up approch) (AGNES Algo)

(2) Divisive (Splitting) (Top to bottom approach). (DIANA Algo)



Datapoints like a, b, c, d, e can either be single data points or cluster representative.

~~Now question is should a be~~

→ finding best merge point is very important but very challenging also.

AGNES (Agglomerative Nesting)

Merge one cluster with other if they have least dissimilarity (or max similarity). (various methods to measure similarity)

Dendrogram: Representation of how data objects in hierarchical clustering are associated.

Methods to find distance b/w clusters:

1) Single link: Min. dist. b/w an element in one cluster & an element in another cluster. i.e. $dist(K_i, K_j) = min(t_{ip}, t_{jq})$

2) Complete link: largest dist. b/w an element in one cluster & an element in another cluster i.e. $dist(K_i, K_j) = max(t_{ip}, t_{jq})$

3) ~~Avg dist~~ Average: $dist(K_i, K_j) = avg(t_{ip}, t_{jq})$
   - avg. dist. b/w element in one cluster & element in another cluster.

4) Centroid: $dist(K_i, K_j) = dist(C_i, C_j)$ (Distance b/w centroids of the two clusters).

5) Medoid: $dist(K_i, K_j) = dist(M_i, M_j)$ (Distance b/w medoids i.e. best representatives of the two clusters).