Association Analysis
(Pattern Mining)

2, 16, 3, 5, 10, 4, 2, 11, 3, 5, 10, 21, 14, 18, 9, 21, 19, 18, 23, 36, 21, 19, 18

like web mining

Another stream of data mining. If we have the above data we can find some patterns in data (i.e. values that are repeated)

- 3, 5 are repeated
- 21, 19, 18

finding pattern is imp. for diff app. If we want to design a mathematical func. to better represent the data like:

$$f(x) = x_1 + 2 \text{ or } x_1 - 2$$

These can be used to write algorithms better

So for our I/P we can rep. the patterns.

eg. In E-commerce website if we buy a shirt then we get recommended the pants by looking at the purchase history pattern of the users. Similarly in scientific analysis (eg. DNA analysis to prevent diseases), identify simila web doc for better classification, providing tags to the email for similarity etc.

If dataset is very large, pattern finding needs some good algo. not manually. Hence need to efficiently find pattern in data

Terminologies

Item: each thing bought by user in a particular transaction

Itemset: set of items → set

- 1-itemset → {Beer} has only item {nuts}, {milk} — each item is a 1-itemset
- 2-itemset → itemset with 2 items eg. {Beer, nuts}, {Beer, Diaper}
- k-itemset → itemset with k items

Support count: sc of an itemset is its freq. of occurrence / appearance in a dataset

eg. if we want to find sc of {Beer, Diaper} then it is 3

to judge whether an association rule is good -- we use support and confidence....

This means that itemset occurs more frequently as compared to other itemsets. i.e. customers are more likely to buy beer and diaper. So, it is freq pattern, freq patterns are rep. by association rules like Beer → Diaper

freq

So, many pattern and hence association rules exist eg. Diaper → Beer,

pattern finding ↓ find useful patterns

How to judge which association rule is good? i.e. which patterns are more useful/interesting?

There are 2 measures for it, we compute for the ass. rule:

support, confidence

If we have an ass. rule $X \rightarrow Y$

$$\text{support}(X \rightarrow Y) = \frac{\sigma(X \cup Y)}{N}$$

($\sigma(X \cup Y)$: support count; N: no. of trans.)

out of total transactions how many contain Beer and Diaper

$$= \frac{3}{5} = 60\%$$

$$\text{confidence}(X \rightarrow Y) = \frac{\sigma(X \cup Y)}{\sigma(X)}$$

where, X is present out of that Y is also present

$$= \frac{3}{3} = 100\%$$

clarity

Support means eg. 60% of students are agreed / liking something, 60% of customers are liking Beer with Diaper. i.e. the pattern is more likely, ↑ support more that pattern is liked by many customers.

reliability

Confidence means eg. out of 60% are all agreed on some constraint. If 2 var are there X and Y, $x = iy$, how much they are associated i.e. the numerical quantification of the (pattern) association

For any pattern then if both are good then it means it is always purchased & purchased together.

0	0.25	5	0.75	1	→	$0 \leq c \leq 1$
none	low	median	partial	high		

an itemset is said to be frequent if its support is greater than the equal to the min. support threshold...

An itemset is said to be freq. if its support is greater than or equal to the min. support threshold of the application

eg: if it is 3 and threshold is 2 then {Beer, Diaper} is frequent itemset

if we have an itemset $\{a_1, \ldots, a_{100}\}$

if in prev. we convert it to standard dataset

	Beer	Nut	Diaper	Coffee	Eggs	Milk
1	1	1	1	0	0	0
2	1	0	1	1	0	0
3	1	0	1	0	1	0
4	0	1	0	0	1	1
5	0	1	1	1	1	1

Here trans. dataset the items are represented as feature

In a general store the standard dataset will have a no. of features. Max length of itemset depends on the no of items in store.

In above max length is 6 ie $a_1, \ldots, a_6$.

If itemset is very long $\{a_1, \ldots, a_{100}\}$ then using it we can form different diff comb. ie. sub-itemset eg. $\{a_1\}$, $\{a_1, a_2\}$ and so on. In above, $2^6 - 1$

In above, $2^{100} - 1$

Out of all we calc. support & confidence for all and see which is good. This is not practical. We need an algo. to filter out the itemsets.

Closed itemset: itemset X is closed if X is frequent and there exists no (subset) superset Y of itemset with the same support as X.

eg. if {Beer, Diaper} is closed or not? → Frequent? if it is higher than min. support threshold 3 > 2. So, it is frequent

↓

Now, a lot of supersets can be formed but none has the same support value as {Beer, Diaper}

Itemset X is max-pattern if X is freq. and there exists no frequent super-pattern Y of X. ie. X is the only frequent itemset so that is possible

support count is maximum for X. eg. if we find a superset {Beer, Diaper, Nuts} then

closed itemset -- is frequent and no superset has same support count...
MAX PATTERN ....

its support count is only 1 which is not ≥ 2. Similarly, for other supersets we can check {B, D, {qq}}, {B, D, {coffee}}. So, {B, D} is both closed and max.

if we have 2 itemsets → $\langle a_1, \ldots, a_{100} \rangle$
→ $\langle a_1, \ldots, a_{50} \rangle$

Threshold = 1

$sc(a_1 \ldots a_{100}) = 1$

$sc(a_1 \ldots a_{50}) = 2$

Both are frequent. No superset of $\langle a_1, \ldots, a_{100} \rangle$ exists. So, it is closed

$\langle a_1, \ldots, a_{50} \rangle$ has a superset but neo superset's support is not ≥ 2. So, it is closed

Only, $\langle a_1, \ldots, a_{100} \rangle$ is max-pattern

For $\langle a_1, \ldots, a_{50} \rangle$ its superset is also frequent. So, it isn't max-pattern

set of all patterns → all possible subset

Problem → No of patterns

we need good algo. to id. the pattern & check if more good or not.

We have → Apriori
→ FPgrowth
→ ECLAT

Downward closure

"Any subset of a freq. itemset must be frequent". So, if we find a freq. itemset then no need to check its subset.

freq. → appears a lot → so obviously

eg. Beer or Diaper will be present in atleast 3 trans. So we don't need to calc the supp & confidence of more

Apriori Algorithm

freq-used in pattern mining

Upward property (Apriori principle)

If node is any itemset that is infrequent then its superset shouldn't be tested as they will also be infrequent. So, no need to check

no need to scan DB again → complexity reduced

Example

steps

Scan the whole DB to get frequent 1-itemset

(Generate) those generate list of 2-itemset, using 2 generate 3 (Using candidate itemset)

From candidate eliminate the infrequent

Repeat until ....

no need to scan DB again → complexity reduced

Example C1 → candidate for 1 itemset given, sup min = 2

{A}	2		
{B}	3		
{C}	3	→	
{D}	1	→ infrequent	
{E}	3		

1st scan

L1: {A}, {B}, {C}, {E} } frequent 1 itemset

From L1, prepare C2 i.e. (n+1) candidate itemset

{A, B}	1	✗	
{A, C}	2	→	L2: {A, C}
{A, E}	1	✗	{B, E}
{B, C}	2		{B, C}
{B, E}	3		{C, E}
{C, E}	2		

2nd scan

Now, 3-itemset C3

A, C, B ✗

A, B, E ✗ since {A, B} is in freq-its superset is also infrequent

A, C, E ✗

B, C, E

Then, length 4 no change. So, we stop

3/04/2025

Q. Solve the following.

Tid	Items
T1	I1, I2, I5
T2	I2, I4
T3	I2, I3
T4	1, 2, 4
T5	1, 3
T6	2, 3
T7	1, 3
T8	1, 2, 3, 5
T9	1, 2, 3

FGrowth (Frequent Pattern Growth)

Before this,

in above if we apply Apriori algorithm:

Candidate 1-itemset

itemset	support
{1}	6
{2}	7
{3}	6
{4}	2
{5}	2

support > min support (count = 2

all the candidate itemset are frequent. L_2 is same as above

$C_2 = L_1 \times L_1$

{1, 2}	4	
{1, 3}	4	
{1, 4}	1	✗
{1, 5}	2	
{2, 3}	4	
{2, 4}	2	
{2, 5}	2	
{3, 4}	0	✗
{3, 5}	1	✗
{4, 5}	0	✗

if while gen 3 itemset we get 4 itemset we ignore

freq 2 itemset: $\{1,2\}$ $\{2,3\}$
$\{1,3\}$ $\{2,4\}$
$\{1,5\}$ $\{2,5\}$

$C_3: L_2 \times L_2$ $\{1,2,3\}$ 2
$\{1,2,5\}$ 2
$\{1,2,4\}$ need not be tested as $\{1,4\}$ is in frequent
$\{1,3,5\}$ $\{3,5\}$

L_3 is same

$C_4: L_3 \times L_3$ $\{1,2,3,5\}$ 1 ✗
it is in frequent

Now we stop
- if L_4 is same as L_3, we stop
- if we get an empty list, we stop

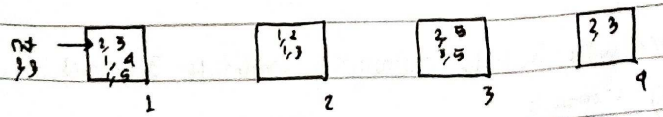
If Q. asks freq. for 3 itemset then stop after L_3

Problem:
This algo. is quite complex. e.g. while computing L_3 from L_2 we need to find support values for which we need to check the database i.e. across transactions (length) and within transactions (width). if a trans. has 3 items then $2^3 - 1$ itemset can be generated.

Attempts are made to reduce this complexity.

Hash-based (Support Counting) Support Count Mechanism

Hash Tree for Support Count. To reduce complexity, we use a hash func. & diff. hash buckets

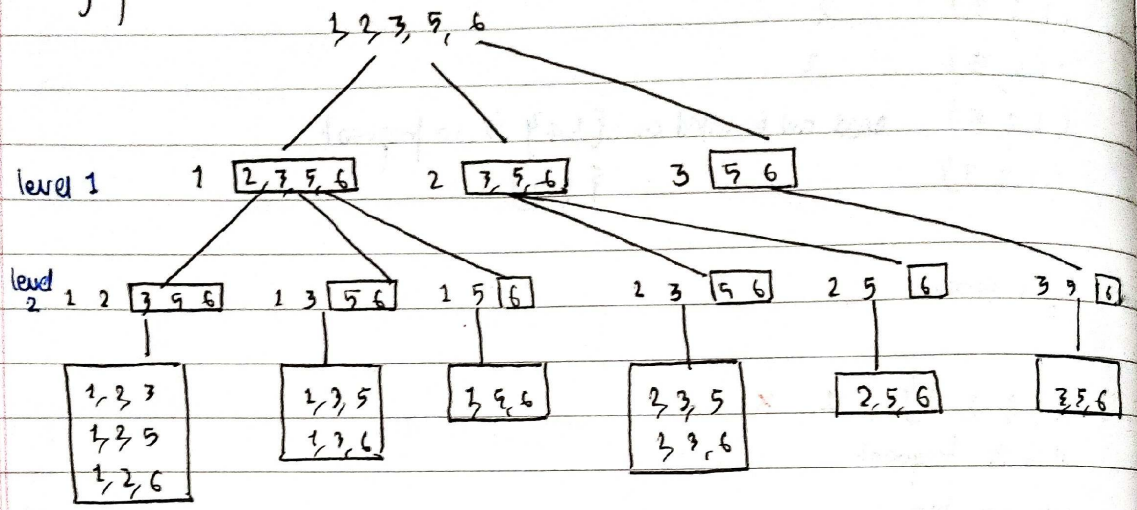


Candidate itemsets generated in each step are placed in buckets. Whatever 2-itemsets belonging to T_1 i.e. $\{1,2\}$, $\{2,5\}$, $\{3,5\}$. Previously, a particular transaction was compared against the width of transaction as well.

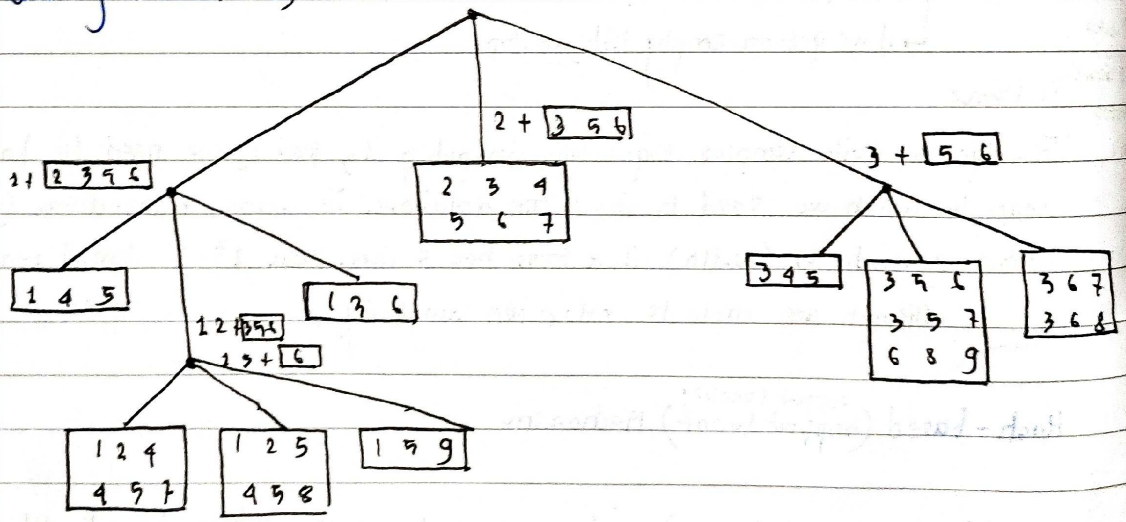
We hash $\{I_1, I_2\}$ to find its bucket and we need to only compare with itemsets

of that bucket only to find the support count.

Let us say we have in a transaction t with items 1, 3, 5, 6. We arrange items in lexicographical order. Now we need to check for candidate 3-itemset. We find all 3's belonging to the transaction



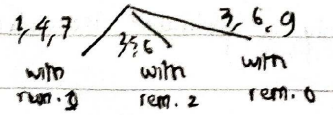
Creating a hash tree,



let us say we have a hash tree with buckets containing candidate 3-itemset

let's say hash func. is $H(p) = p \bmod 3$

$H(p) = p \bmod 3$ where p may be 1st item of itemset for 1st level

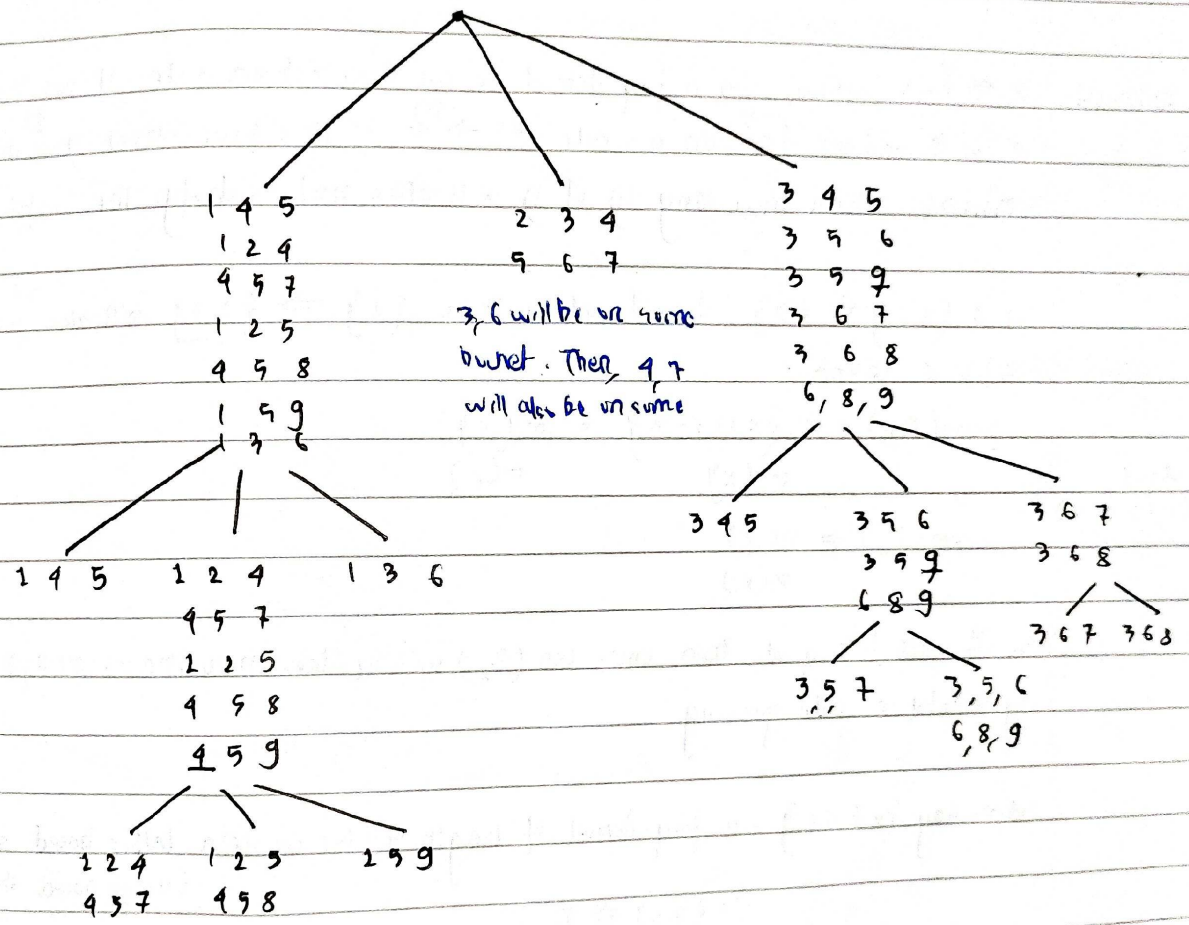


If we hash transaction 1, 2, 3, 5, 6 then those starting with 1 are on left, 2 middle, 3 on (left) right. This is for 1st level.

For 2nd level, 2nd item may be 2, 3, 5. If it is 2 then at middle, 5 then also middle, if 3 then on right. So, if our 2nd element is 2 or 5 in our trans. we need to check the middle. Similarly for others.

So, to find the support count only few of the itemsets gets compared not all. If {1, 4, 5}, {1, 2, 4}, ... is our candidate 3-itemset.

* 1st hash on 1st item, then 2nd item, then 3rd item



In this way, we form the Hash Tree

Confidence Based Pruning

If we have an itemset (freq.) Y of length 'k' then no of (associated) (valid) association rules (men) is $2^k - 2$.

e.g. if our itemset (freq.) is {1, 2, 3}. We can make the ass. rules:

{1} ⇒ {2, 3} {1, 2} ⇒ {3} {ϕ} ⇒ {1, 2, 3} {1, 2, 3} ⇒ {ϕ}

{2} ⇒ {1, 3}

not considered

confidence based pruning -- 2^k - 2

itemset is freq. if it satisfies support (only needs support)

We want to find the useful (itemsets / rules) that support the support & also confidence criteria. If an itemset is freq. then all the ass. rules will satisfy the support criteria i.e. $\frac{\sigma(X \cup Y)}{N}$. But, we also need it to support confidence. For each rule the confidence would be different i.e. $\frac{\sigma(X \cup Y)}{\sigma(X)}$ confidence based ass. is useful because while doing the apriori algo. we must have already found the necessary support values, so for conf. confidence we don't need to rescan the DB.

Statement: $X \Rightarrow Y - X$ (i), where Y is a freq. itemset, is an association rule. If we have an $X' \subset X$ and we find (wrote) an ass. rule $X' \Rightarrow Y - X'$ (ii), then if we know i) doesn't satisfies the confidence criteria then any ii) of i) will also not satisfy the confidence criteria.

eg- if $\{1,2\} \Rightarrow \{3\}$ doesn't satisfy then $\{1\} \Rightarrow \{2,3\}$ will also not satisfy the confidence criteria.

always $\sigma(X') \geq \sigma(X)$

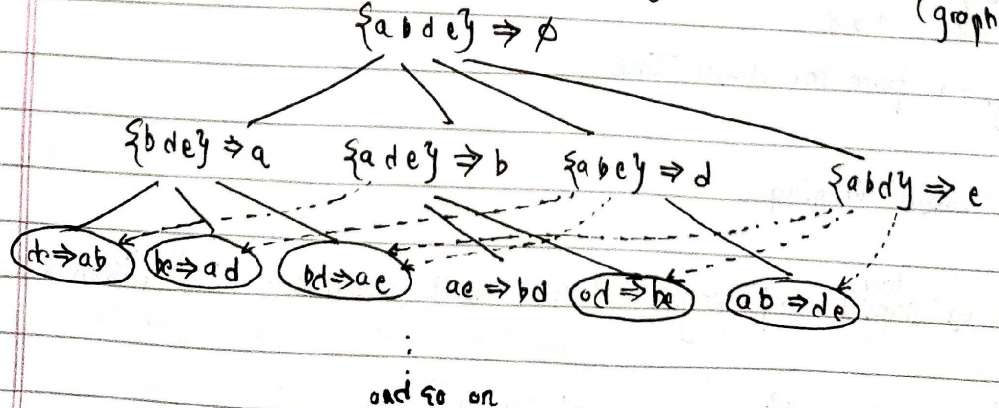
$$conf(R_2) = \frac{\sigma\{X \cup Y - X\}}{\sigma(X)} = \frac{\sigma(Y)}{\sigma(X)}$$

$$conf(R_1) = \frac{\sigma(Y)}{\sigma(X')}$$

eg, if $\sigma(X')$ is greater then obv. $conf(R_1)$ will be less than the $conf(R_2)$. This is very helpful in rule proving.

1st step:

let us say $\{a\,b\,d\,e\}$ is a freq. itemset of length 4. We create a lattice based structure (graph based structure)



2nd step:

once we get a frequent itemset. then we can use the lattice method to find out all the association rules... if the node does not satisfy the confidence then all the rules rooted on it. (in its subtree) will not satisfy so we eliminate them..

FP- growth is used to find the frequent itemset...improving apriori algorithm..

This is called lattice based structure for freq. itemset. If we find freq. 4 itemset using apriori, then our application may need the rules to support support, confidence, correlation acc. to its need.

This shows the possible ass. rules corresponding to the freq. itemset. With this, all operations of graph are applicable. Support we write any rule {b d e} ⇒ {a}. If this doesn't satisfy the confidence then all the rules rooted on it don't satisfy the confidence so we eliminate

We check 1st level, if all rules satisfy confidence then we go to next level. Then if any rule doesn't satisfy (them) then the sub-tree can be removed.

If we know support values of each we can also find closed & max itemset. If we say e ⇒ a b d then we need to check the immediate super sets (parent). If they don't have same support value then e ⇒ a b d isn't a closed. Similarly, for max pattern too.

4/4/2025

FP Growth (Frequent Pattern)

Same Q as before

another method to find freq-itemset. Trying to improve the apriori algorithm

1st step: 1st we scan our dataset, find 1-itemset & their sc, same as apriori algorithm

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

2nd step: sort itemsets acc. to sc in decreasing order

{2}	7
{1}	6

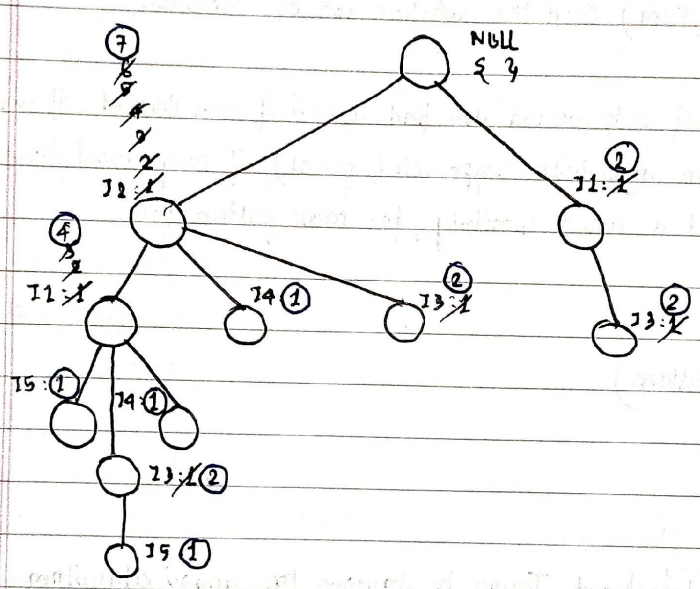
same then any order $1, 2, 3, 5 \rightarrow \underbrace{2, 1, 3, 5}_{\text{order}}$

{3} 6 called L-list
{2} 2
{5} 2

3rd step: Construct an FP tree

Root is considered as NULL scan the dataset & add branches to the tree as per the above order. 1st trans. is {1, 2, 5} so branch is I2, I1, I5 acc. to the L-list. Scan 1st trans., then 2nd and so on acc. to above order

For 1st time support is 1

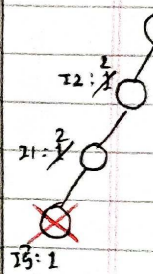


4th step: Create a table as follows:

Item Id	Support count	Node-link
I2	7	
I1	6	
I3	6	
I4	2	
I5	2	

5th step: Create the following:

Approach is to partition the dataset and find sup. corresponding to each partition. Compared to pattern we used the cond. FP-tree, Reduction of complexity (acc. to L-list order)



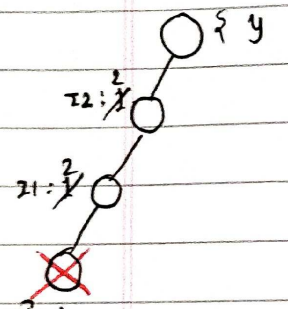
as it doesn't support the min support count of 2

ways to reach the item, ahead of partition of the dataset

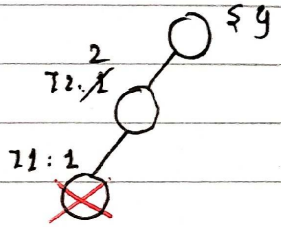
Item	Conditional Pattern Base	Conditional FP-tree	Freq. Pattern Generator
I5	{I2, I1 : 1} (1 of I5 on the branch) {I2, I1, I3 : 1}	{I2 : 2, I1 : 2} Using these we gen. freq. pattern by simply concat I5 with two items	{I2, I5 : 2} {I1, I5 : 2} i) {I2, I1, I5 : 2} → as in same branch
I4	{I2, I1, (I4) : 1} {I2 : 1}	{I2 : 2}	{I2, I4 : 2}
I3	{I2, I1 : 2} {I2 : 2} {I1 : 2}	{I2 : 4, I1 : 3, I1 : 2}	{I2, I3 : 4} {I1, I3 : 4} ii) {I2, I1, I3 : 2}
I1	{I2 : 4} {Null is not used as superscript}	{I2 : 4}	{I2, I1 : 4}

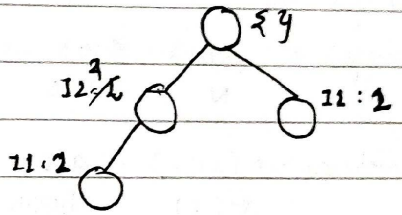
I5 is 1st as it is in the bottom of L-list

with cond. pattern base we construct a cond. FP-tree

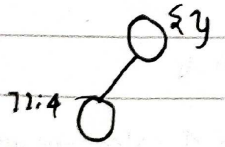


as it doesn't support the min support count of 2





Here I3b take 2 as we have {I2, I1 : 2}



i) and ii) are freq. 3-itemset. We also have freq. 2-itemsets

This doesn't use generate & test. Pattern use freq. grown based on the partition of data.

Strong Association Rule

N : 10,000

Computer Games : 6000

video : 7500

once we identify the frequent itemset, we can write all possible association rules ... by using lattice based structure and prune to identify the rules satisfying the both support and confidence criteria...

it might be the case that rule is satisfying both the criterias but it is still not strong association.....

Computer Games & Video = 4000

once we id the freq itemset, we can write all possible ass rules thereby using the lattice based structure & prune to id the rules satisfying the both support and confidence criteria.

If we have a dataset with 10,000 trans. 6000 customers buy Video, 4000 customers purchase both computer games and video.

If we write an ass. rule $X \rightarrow Y$ then,

Computer Games $\rightarrow$ Video

i.e. a customer buy C.G also buys Video,

Suppose, min. support = 40 % } for useful rules
min confidence = 66 %

For above if we find,

$$\text{support} = \frac{\sigma(X \cup Y)}{N} = \frac{4000}{10000} = 40\%.$$

$$\text{confidence} = \frac{\sigma(X \cup Y)}{\sigma(X)} = \frac{4000}{6000} = 66\%.$$

σ(Computer Games)

So, the above is a useful rule as it satisfies both criteria. But it isn't a strong ass rule As, we find probability of occurrence of video is $\frac{7500}{10000}$ = 75 %. But if video is bought along with (comp games) games then the confidence is 66 %. So, the prob. decreases. This is known as -ve correlation

↳ 2 itemsets are dependent & how much

Using this we can find better rules. Using the correlation we can ↑ sales. -ve correlation then the sales decreases.

So, to judge the usefulness of rule we have correlation as a criteria

Correlation

methods to compute:

i) Lift

$$lift(A,B) = \frac{P(A \cup B)}{P(A)P(B)}$$ intersection....

If 2 events are independent then, $P(A \cup B) = P(A) \cdot P(B)$

A, B are two itemsets.

lift may be $(= 0)\ (\geq 1)\ < 1$

- (0) lift when they are independent (no corr)
- $= 1$
- < 1 -ve correlation
- > 1
- (≥ 1) +ve correlation

In prev., $lift(G, V) = \frac{0.4}{0.6 \times 0.75} = 0.88$

ie- it is < 1, so we have a -ve correlation. This much we need to lift to get the good itss.

ii) Using Chi-Squared Test

χ^2: correlation

We prepare a contigency table

	Games (4500)	¬ Games / $\bar{G}$ ames	
Video	4000	(7500) 3500	7500
¬ Video	2000 (6000)	500	2500
	6000	4000	10000

These are observed values we can compute the expected value

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

$\frac{Games \times Video}{N} = \frac{6000 \times 7500}{10000} = 4500$. Similarly for all

Consider to be independent, now if hypothesis is rejected they are dependent. In above, the hypo. is rejected and they are corr. The value we get from above is the amount of correlation.