International Research Journal of Engineering and Technology (IRJET) e-ISSN: 2395-0056
Volume: 06 Issue: 04 | Apr 2019 www.irjet.net p-ISSN: 2395-0072

Minning Frequent Patterns, Associations and Correlations

Aravind Chowdary[1], Savya Chamarti[2], A Likith Reddy[3], Yavapuram Mahesh Babu[4] , K Radha[5]

[1,2,3,4]III-B.TECH-CSE,GITAM UNIVERSITY, Rudraram, Hyderabad, Telangana
[5] Asst Professor, CSE, GITAM UNIVERSITY, Rudraram, Hyderabad, Telangana, India

---

Abstract - In this paper, we will learn how to mine frequent patterns, association rules, and correlation rules when working with R programs. Then, we will evaluate all these methods with benchmark data to determine the interestingness of the frequent patterns and rules.

Key Words: Correlation Rule, Frequent Patterns, bench mark data.

1. INTRODUCTION

Frequent patterns: Frequent patterns are the ones that often occur in the source dataset. The dataset types for frequent pattern mining can be itemset, subsequence, or substructure.

These three frequent patterns

i. Frequent itemset

ii. Frequent subsequence

iii. Frequent substructures

Frequent patterns are patterns (such as itemsets, subsequences, or substructures) that appear in a data set frequently. For example, a set of items, such as milk and bread, that appear frequently together in a transaction data set is a frequent itemset. A subsequence, such as buying first a PC, then a digital camera, and then a memory card, if it occurs frequently in a shopping history database, is a (frequent) sequential pattern. A substructure can refer to different structural forms, such as subgraphs, subtrees, or sublattices, which may be combined with itemsets or subsequences.

Market basket analysis

Market basket analysis is the methodology used to mine a shopping cart of items bought or just those kept in the cart by customers. The concept is applicable to a variety of applications, especially for store operations. The source dataset is a massive data record. The aim of market basket analysis is to find the association rules between the items within the source dataset.

1.1. The market basket model

The market basket model is a model that illustrates the relation between a basket and its associated items. Many tasks from different areas of research have this relation in common. To summarize them all, the market basket model is suggested as the most typical example to be researched [1].

The basket is also known as the transaction set; this contains the itemsets that are sets of items belonging to same itemset.

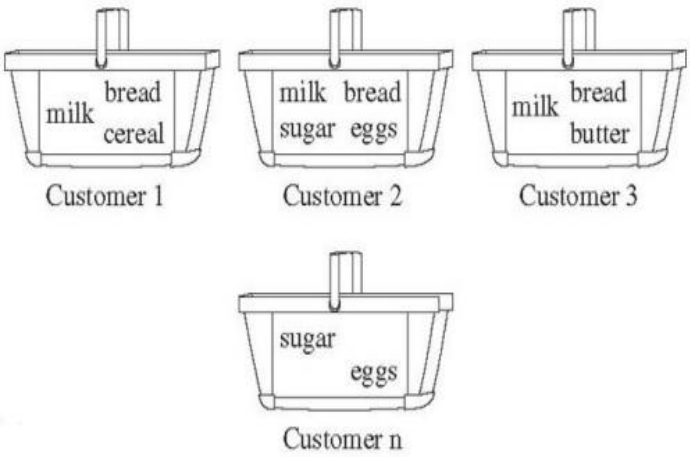

Fig.1: Market Basket Analysis

Confidence, Support, and Association Rules

If we think of the total set of items available in our set (sold at a physical store, at an online retailer, or something else altogether, such as transactions for fraud detection analysis), then each item can be represented by a Boolean variable, representing whether or not the item is present within a given "basket." Each basket is then simply a Boolean vector, possibly quite lengthy dependent on the number of available items. A dataset would then be the resulting matrix of all possible basket vectors.

This collection of Boolean basket vectors are then analyzed for associations, patterns, correlations, or whatever it is you would like to call these relationships. One of the most common ways to represent these patterns is via **association rules**, a single example of which is given below:

milk =>bread [support = 25%, confidence=60%]

How do we know how interesting or insightful a given rule may be? That's where support and confidence come in.

Support is a measure of *absolute frequency*. In the above example, the support of 25% indicates that, in our finite dataset, milk and bread are purchased together in 25% of all transactions.

Confidence is a measure of *correlative frequency*. In the above example, the confidence of 60% indicates that 60% of those who purchased milk also purchased bread.

1. In a given application, association rules are generally generated within the bounds of some predefined minimum threshold for both confidence and support, and rules are only considered interesting and insightful if they meet these minimum thresholds.

2. **Various patterns are proposed to improve the efficiency of mining on a dataset.**

[Reference: Y. Yuan, C. Yang, Y. Huang, and D. Mining, And the Optimization Technology nd Its Application. Beijing. Science Press, 2007]

1. Closed patterns
2. Maximal patterns
3. Approximate patterns
4. Condensed patterns
5. Discriminative frequent patterns

Finding such frequent patterns plays an essential role inmining associations, correlations, and many other interesting relationships among data.

Moreover, it helps in data classification, clustering, and other data mining tasks as well.Thus, frequent pattern mining has become an important data mining task and a focusedtheme in data mining research.

Association rules

In a later section, a method to show association analysis is illustrated; this is a useful method to discover interesting relationships within a huge dataset. The relations can be represented in the form of association rules or frequent itemsets [1].

Association rule mining is to find the result rule set on a given dataset (the transaction data set or other sequence-pattern-type dataset), a predefined minimum support count **s**, and a predefined confidence c, given any found rule, and is an association rule where ; X and Y are disjoint.

The interesting thing about this rule is that it is measured by its **support** and **confidence**. Support means the frequency in which this rule appears in the dataset, and confidence means the probability of the appearance of Y when X is present.

For association rules, the key measures of rule interestingness are rule support and confidence. Their relationship is given as follows:

support_count(X) is the count of itemset in the dataset, contained X.

As a convention, in support_count(X), in the confidence value and support count value are represented as a percentage between 0 and 100.

The association rule is strong once and . The predefinedminimum support threshold is **s**, and c is the predefined minimum confidence threshold.

The meaning of the found association rules should be explained with caution, especially when there is not enough to judge whether the rule implies causality. It only shows the co-occurrence of the prefix and postfix of the rule.

3. **The following are the different kinds of rules you can come across:**

- ✓ A rule is a Boolean association rule if it contains association of the presence of the item
- ✓ A rule is a single-dimensional association if there is, at the most, only one dimension referred to in the rules
- ✓ A rule is a multidimensional association rule if there are at least two dimensions referred to in the rules
- ✓ A rule is a correlation-association rule if the relations or rules are measured by statistical correlation, which, once passed, leads to a correlation rule
- ✓ A rule is a quantitative-association rule if at least one item or attribute contained in it is quantitative[2].

3.1. Correlation rules

In some situations, the support and confidence pairs are not sufficient to filter uninteresting association rules. In such a case, we will use support count, confidence, and correlations to filter association rules.

There are a lot of methods to calculate the correlation of an association rule, such as analyses, all-confidence analysis, and cosine. For a k-itemset , define the all-confidence value of X as:

$$Lift(X->Y)=confidence(X->Y) / P(Y)=P(XUY)/(P(X)P(Y))$$

4. Basic Concepts and a Road-Map

The basic concepts, techniques, and applications of frequent pattern mining using market basket analysis as an example. Many other kinds of data, user requests, and applications have led to the development of numerous, diverse methods for mining patterns, associations, and correlation relationships. Given the rich literature in this area, it is important to lay out a clear road map to help us get an organized picture of the field and to select the best methods for pattern mining applications [4].

Figure 1 outlines a general road map on pattern mining research. Most studies mainly address three pattern mining aspects: the kinds of patterns mined, mining methodologies, and applications. Some studies, however, integrate multiple aspects; for example, different applications may need to mine different patterns, which naturally leads to the development of new mining methodologies.

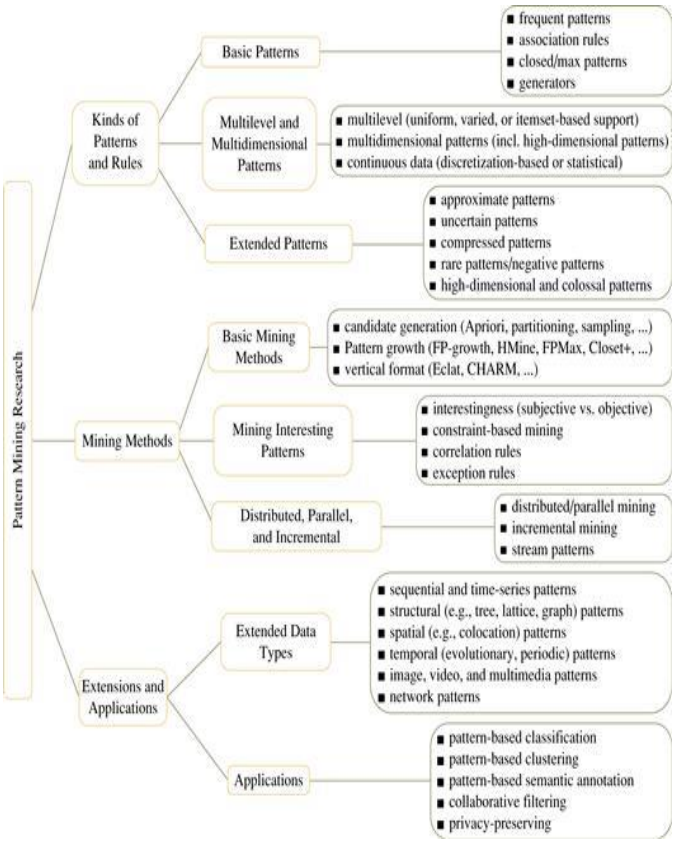

FIG. 2.A general road map on pattern mining research.

Based on pattern diversity, pattern mining can be classified using the following criteria:

■ **Basic patterns:** A frequent pattern may have several alternative forms, including a simple frequent pattern, a closed pattern, or a max-pattern. To review, a **frequent pattern**is a pattern (or itemset) that satisfies a minimum support threshold. A pattern p is a **closed pattern** if there is no superpattern p' with the same support as p. Pattern p is a **max-pattern** if there exists no frequent superpattern of p. Frequent patterns can also be mapped into **association rules**, or other kinds of rules based on interestingness measures. Sometimes we may also be interested in **infrequent** or **rare patterns** (i.e., patterns that occur rarely but are of critical importance, or **negative patterns** (i.e., patterns that reveal a negative correlation between items).

■ **Based on the *abstraction* levels involved in a pattern:** Patterns or association rules may have items or concepts residing at high, low, or multiple abstraction levels. For example, suppose that a set of association rules mined includes the following rules where X is a variable representing a customer:

$$buys(X, \text{“}computer\text{”}) \Rightarrow buys(X, \text{“}printer\text{”}) \textbf{---(a)}$$

$$buys(X, \text{“}laptop\ computer\text{”}) \Rightarrow buys(X, \text{“}color\ laser\ printer\text{”}) \textbf{[1]}$$

In Rules (a) the items bought are referenced at different abstraction levels (e.g., “*computer*” is a higher-level abstraction of “*laptop computer*,” and “*color laser printer*” is a lower-level abstraction of “*printer*”). We refer to the rule set mined as consisting of **multilevel association rules**. If, instead, the rules within a given set do not reference items or attributes at different abstraction levels, then the set contains **single-level association rules**.

■ **Based on the *number of dimensions* involved in the rule or pattern:** If the items or attributes in an association rule or pattern reference only one dimension, it is a **single-dimensional association rule/pattern**. For example, in below Rules (a) and are single-dimensional association rules because they each refer to only one dimension, *buys*.[1] If a rule/pattern references two or more dimensions, such as *age, income*, and *buys*, then it is a **multidimensional association rule/pattern**. The following is an example of a multidimensional rule:

$$age(X, \text{“}20 \ldots 29\text{”}) \wedge income(X, \text{“}52K \ldots 58K\text{”}) \Rightarrow buys(X, \text{“}iPad\text{”}). \textbf{(b)}$$

■ **Based on the *types of values* handled in the rule or pattern:** If a rule involves associations between the presence or absence of items, it is a **Boolean association rule**. For example, Rules (a) and (b) are Boolean association rules obtained from market basket analysis. If a rule describes associations between quantitative items or attributes, then it is a **quantitative association rule**. In these rules, quantitative values for items or attributes are partitioned into intervals. Rule (b) can also be considered a quantitative association rule where the quantitative attributes *age*and *income* have been discretized.

■ **Based on the *constraints* or *criteria* used to mine *selective patterns*:** The patterns or rules to be discovered can be **constraint-based** (i.e., satisfying a set of user-definedconstraints), **approximate**, **compressed**, **near-match** (i.e., those that tally the support count of the near or almost matching itemsets), **top-*k*** (i.e., the *k* most frequent itemsets for a user-specified value, *k*), **redundancy-aware top-*k*** (i.e., the top-*k* patterns with similar or redundant patterns excluded), and so on.

Alternatively, pattern mining can be classified with respect to the kinds of data and applications involved, using the following criteria:

■ **Based on *kinds of data and features* to be mined:** Given relational and data warehouse data, most people are interested in itemsets. Thus, frequent pattern mining in this context is essentially **frequent itemset mining**, that is, to mine frequent *sets of items*. However, in many other applications, patterns may involve sequences and structures. For example, by studying the order in which items are frequently purchased, we may find that customers tend to first buy a PC, followed by a digital camera, and then a memory card. This leads to **sequential patterns**, that is, frequent *subsequences* (which are often separated by some other events) in a *sequence of ordered events*.

We may also mine **structural patterns**, that is, frequent *substructures*, in a *structured data set*. Note that *structure* is a general concept that covers many different kinds of structural forms such as directed graphs, undirected graphs, lattices, trees, sequences, sets, single items, or combinations of such structures. Single items are the simplest form of structure. Each element of a general pattern may contain a subsequence, a subtree, a subgraph, and so on, and such containment relationships can be defined recursively. Therefore, structural pattern mining can be considered as the most general form of frequent pattern mining.

■ **Based on *application domain-specific semantics*:** Both data and applications can be very diverse, and therefore the patterns to be mined can differ largely based on their domain-specific semantics. Various kinds of application data include spatial data, temporal data, spatiotemporal data, multimedia data (e.g., image, audio, and video data), text data, time-series data, DNA and biological sequences, software programs, chemical compound structures, web structures, sensor networks, social and information networks, biological networks, data streams, and so on. This diversity can lead to dramatically different pattern mining methodologies.

■ **Based on *data analysis usages*:** Frequent pattern mining often serves as an intermediate step for improved data understanding and more powerful data analysis. For example, it can be used as a feature extraction step for classification, which is often referred to as **pattern-based classification**. Similarly, **pattern-based clustering** has shown its strength at clustering high-dimensional data. For improved data understanding, patterns can be used for semantic annotation or contextual analysis. Pattern analysis can also be used in **recommender systems**, which recommend information items (e.g., books, movies, web pages) that are likely to be of interest to the user based on similar users' patterns. Different analysis tasks may require mining rather different kinds of patterns as well.[3]

The A-Priori algorithm is a level wise, itemset mining algorithm. The Eclat algorithm is a tidset intersection itemset mining algorithm based on tidset intersection in contrast to A-Priori. FP-growth is a frequent pattern tree algorithm. The tidset denotes a collection of zeros or IDs of transaction records.[3]

1) A-Priori algorithm:

The **Apriori Algorithm** is an influential **algorithm** for mining frequent itemsets for boolean association rules.

• **Apriori uses** a "bottom up" approach, where frequent subsets are extended one item at a time (a step known as candidate generation, and groups of candidates are tested against the data.

Efficiency:

An approach to improve the **efficiency of apriori algorithm**. Association rule mining has a great importance in data mining. **Apriori** is the key **algorithm** in association rule mining.

Apriori Property –

All nonempty subset of frequent itemset must be frequent.

The key concept of Apriori algorithm is its anti-monotonicity of support measure. Apriori assumes that

All subsets of a frequent itemset must be frequent(Apriori propertry).

If a itemset is infrequent all its supersets will be infrequent.

- **Consider the following dataset and we will find frequent itemsets and generate association rules on this.[7]**

TID	items
T1	I1, I2 , I5
T2	I2,I4
T3	I2,I3
T4	I1,I2,I4
T5	I1,I3
T6	I2,I3
T7	I1,I3
T8	I1,I2,I3,I5
T9	I1,I2,I3

minimum support count is 2
minimum confidence is 60%

Step-1: K=1

(1) Create a table containing support count of each item present in dataset – Called **C1(candidate set)**

Itemset	sup_count
I1	6
I2	7
I3	6
I4	2
I5	2

(2) compare candidate set item's support count with minimum support count(here min_support=2 if support_count of candidate set items is less than min_support then remove those items) this gives us itemset L1.

Itemset	sup_count
I1	6
I2	7
I3	6
I4	2
I5	2

Step-2: K=2

- Generate candidate set C2 using L1 (this is called join step). Condition of joining is L_{k-1} and L_{k-1} is that it should have (K-2) elements in common.
- Check all subsets of a itemset are frequent or not and if not frequent remove that itemset.(Example subset of{I1, I2} are {I1}, {I2} they are frequent.Check for each itemset)
- Now find support count of these itemsets by searching in dataset.

Itemset	sup_count
I1,I2	4
I1,I3	4
I1,I4	1
I1,I5	2
I2,I3	4
I2,I4	2
I2,I5	2
I3,I4	0
I3,I5	1
I4,I5	0

(2) compare candidate (C2) support count with minimum support count(here min_support=2 if support_count of candidate set item is less than min_support then remove those items) this gives us itemset L2.

Itemset	sup_count
I1,I2	4
I1,I3	4
I1,I5	2
I2,I3	4
I2,I4	2
I2,I5	2
I2,I5	2

Step-3:

1. Generate candidate set C3 using L2 (join step). Condition of joining L_{k-1} and L_{k-1} is it should have (K-2) elements in common. So here for L2 first element should match. So itemset generated by joining L2 is {I1, I2, I3}{I1, I2, I5}{I1, I3, i5}{I2, I3, I4}{I2, I4, I5}{I2, I3, I5}
2. Check all subsets of these itemsets are frequent or not and if not remove that itemset.(Here subset of {I1, I2, I3} are {I1, I2}{I2, I3}{I1, I3} which are frequent. For {I2, I3, I4} subset {I3, I4} is not frequent so remove this. Similarly check for every itemset)
3. find support count of these remaining itemset by searching in dataset.

Itemset	sup_count
I1,I2,I3	2
I1,I2,I5	2

(2) Compare candidate (C3) support count with minimum support count(here min_support=2 if support_count of candidate set item is less than min_support then remove those items) this gives us itemset L3.

Itemset	sup_count
I1,I2,I3	2
I1,I2,I5	2

Step-4:

a. Generate candidate set C4 using L3 (join step). Condition of joining L_{k-1} and L_{k-1}(K=4)is these should have (K-2) elements in common. So here for L3 first 2 element(items) should match.

b. Check all subsets of these itemsets are frequent or not(Here itemset formed by joining L3 is {I1, I2, I3, I5} so its subset contain {I1, I3, I5} which is not frequent). so no itemset in C4

c. We stop here because no frequent itemset are found frequent further

Thus we discovered all frequent item-sets now generation of strong association rule comes into picture. For that we need to calculate confidence of each rule.

Confidence

A confidence of 60% means that 60% of the customers who purchased a milk and bread also bought the butter.

Confidence(A->B)=Support_count(A∪B)/Support_count(A)

So here By taking example of any frequent itemset we will show rule generation.
Itemset {I1, I2, I3} //from L3

SO rules can be

I. [I1^I2]=>[I3] //confidence = sup(I1^I2^I3)/sup(I1^I2) = 2/4*100=50%

II. [I1^I3]=>[I2] //confidence = sup(I1^I2^I3)/sup(I1^I3) = 2/4*100=50%

III. [I2^I3]=>[I1] //confidence = sup(I1^I2^I3)/sup(I2^I3) = 2/4*100=50%

IV. [I1]=>[I2^I3] //confidence = sup(I1^I2^I3)/sup(I1) = 2/6*100=33%

V. [I2]=>[I1^I3] //confidence = sup(I1^I2^I3)/sup(I2) = 2/7*100=28%

VI. [I3]=>[I1^I2] //confidence = sup(I1^I2^I3)/sup(I3) = 2/6*100=33%

So if minimum confidence is 50 % first 3 rules can be considered strong association rules.[6]

As a common strategy to design algorithms, the problem is divided into two subproblems:

The frequent itemset generation

Rule generation

The strategy dramatically decreases the search space for association mining algorithms.

4.1. Input data characteristics and data structure

As the input of the A-Priori algorithm, the original input itemset is binarized, that is, 1 represents the presence of a certain item in the itemset; otherwise, it is 0. As a default assumption, the average size of the itemset is small. The popular preprocessing method is to map each unique available item in the input dataset to a unique integer ID.

The itemsets are usually stored within databases or files and will go through several passes. To control the efficiency of the algorithm, we need to control the count of passes. During the process when itemsets pass through other itemsets, the representation format for each itemset you are interested in is required to count and store for further usage of the algorithm.

The A-Priori algorithm

Apriori

Apriori enjoys success as the most well-known example of a frequent pattern mining algorithm. Given the above treatment of market basket analysis and item representation, Apriori datasets tend to be large, sparse matrices, with items (attributes) along the horizontal axis, and transactions (instances) along the vertical axis.

From an initial dataset of *n* attributes, Apriori computes a list of candidate itemsets, generally ranging from size 2 to *n*-1, or some other specified bounds. The number of possible itemsets of size *n*-(*n*+1) to *n*-1 that can be constructed from a dataset of

$$C(n, n-(n+2)) + C(n, n-(n+3)) + \ldots + C(n, n-1).$$

size *n* can be determined as follows, using combinations:

The above can also be expressed using the binomial coefficient.

Very large itemsets held within extremely large and sparse matrices can prove very computationally expensive.

```
INPUT: S, support where S = dataset, min_support = real
OUTPUT: Set of Frequent Itemsets
Require: S ≠ ∅, 0 ≤ min_support ≤ 1
1: procedure GETFREQUENTITEMSETS
2:     freqSets[ ] ← null
3:     for all Itemsets i in S do
4:         if support ≥ min_support then
5:             freqSets[ ] ← i
6:         end if
7:     end for
8: end procedure
```

Fig.2: Apriori Candidate Itemset Generation Algorithm

A support value is provided to the algorithm. First, the algorithm generates a list of candidate itemsets, which includes all of the itemsets appearing within the dataset. Of the candidate itemsets generated, an itemset can be determined to be frequent if the number of transactions that it appears in is greater than the support value.

```
INPUT: S where S = dataset
OUTPUT: Set of Candidate Itemsets
Require: S ≠ ∅
1: procedure GENERATECANDIDATES
2:     i ← 2
3:     num ← NumAttributes(S)
4:     candidates[ ] ← null
5:     while i < num do
6:         candidates[ ] ← all sets of size i, support
7:         i ← i + 1
8:     end while
9: end procedure
```

Fig.3: Apriori Frequency Itemset Selection Algorithm

[**Reference:** J. Han and M. Kamber, Conception and Technology of Data Mining, Beijing: China Machine Press, 2007.][3]

Explicit association rules can then trivially be generated by traversing the frequent itemsets, and computing associated confidence levels. Confidence is the proportion of the transactions containing item A which also contains item B, and is calculated as

$$\text{Confidence}(A \Rightarrow B) = \frac{\text{Support}(A \cup B)}{\text{Support}(A)}$$

ID	Rule	Support	Confidence
r1	{a, b, c}⇒{e}	0.5	1.0
r2	{a}→{c, e, f}	0.5	0.66
r3	{a, b} →{e, f}	0.5	1.0
r4	{b} →{e, f}	0.75	0.75
r5	{a} →{e, f}	0.75	1.0
r6	{c} →{f}	0.5	1.0
r7	{a}→{b}	0.5	0.66
…	…	…	…

Fig.4: Sample Association Rules with Support and Confidence

(Source: An introduction to frequent pattern mining, by Philippe Fournier-Viger.

The manner in which Apriori works is quite simple; it computes all of the rules that meet minimum support and confidence values. The number of possible potential rules increases exponentially with the number of items in the itemset. Since the computation of new rules does not rely on previously computed rules, the Apriori algorithm provides an opportunity for parallelism to offset computation time.[5]

FP tree Algorithm

FP tree algorithm, which use to identify frequent patterns in the area of Data Mining. I'm sure! after this tutorial you can draw a FP tree and to identify frequent patterns from that tree you have to read my next post, How to identify frequent patterns from FP tree.

Suppose you got a question as follows:

Question : Find all frequent itemsets or frequent patterns in the following database using FP-growth algorithm. Take minimum support as 30%.

TID	Items
1	E, A, D, B
2	D, A, C, E, B
3	C, A, B. E
4	B, A, D
5	D
6	D,B
7	A,D,E
8	B,C

Table 1 - Snapshot of the Database

Step 1 - Calculate Minimum support

First should calculate the minimum support count. Question says minimum support should be 30%. It calculate as follows:

Minimum support count(30/100 * 8) = **2.4**

As a result, 2.4 appears but to empower the easy calculation it can be rounded to to the ceiling value. Now,

Minimum support count is **ceiling**(30/100 * 8) = **3**

Step 2 - Find frequency of occurrence

Now time to find the frequency of occurrence of each item in the Database table. For example, item A occurs in row 1,row 2,row 3,row 4 and row 7. Totally 5 times occurs in the Database table. You can see the counted frequency of occurrence of each item in Table 2.

Item	Frequency
A	5
B	6
C	3
D	6
E	4

Table2 -Frequency of Occurrence

Step 3 - Prioritize the items

In Table 2 you can see the numbers written in Redpen. Those are the priority of each item according to it's frequency of occurrence. Item B got the highest priority (**1**) due to it's highest number of occurrences. At the same time you have opportunity to drop the items which not fulfill the minimum support requirement.For instance, if Database contain **F** which has frequency 1, then you can drop it.

*Some people display the frequent items using list instead of table. The frequent item list for the above table will be **B:6, D:6, A: 5, E:4, C: 3**.

Step 4 -Order the items according to priority

As you see in the Table 3 new column added to the Table 1. In the Ordered Items column all the items are queued according to it's priority, which mentioned in the Red ink in Table 2. For example, in the case of ordering row 1, the highest priority item is B and after that D, A and E respectively.

TID	Items	Ordered Items
1	E, A, D, B	B,D,A,E
2	D, A, C, E, B	B,D,A,E,C
3	C, A, B. E	B,A,E,C
4	B, A, D	B,D,A
5	D	D
6	D,B	B,D
7	A,D,E	D,A,E
8	B,C	B,C

Table 3 - New version of the Table 1

Step5-Order the items according to priority

As a result of previous steps we got a ordered items table (Table 3). Now it's time to draw then FP-tree. I'll mention it row by row.

Row 1:

Note that all FP trees have 'null' node as the root node. So draw the root node first and attach the items of the row 1 one by one respectively. (See the Figure 1) And write their occurrences in front of it. (write using a pencil dear,because next time we have to erase it. :D)

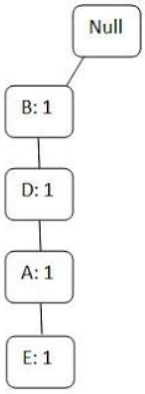

Figure 1- FP tree for Row 1

Row 2:

Then update the above tree (Figure 1) by entering the items of row 2. The items of row 2 are B,D,A,E,C. Then without creating another branch you can go through the previous branch up to E and then you have to create new node after that for C. This case same as a scenario of traveling through a road to visit the towns of the country. You should go through the same road to achieve another town near to the particular town.

When you going through the branch second time you should erase one and write two for indicating the two times you visit to that node.If you visit through three times then write three after erase two. Figure 2 shows the FP tree after adding row 1 and row 2. Note that the red underlines which indicate the traverse times through the each node.

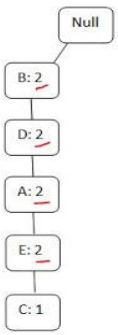

Figure 2- FP tree for Row 1,2

Row 3:

In row 3 you have to visit B,A,E and C respectively. So you may think you can follow the same branch again by replacing the values of B,A,E and C . But you can't do that you have opportunity to come through the B. But can't connect B to existing A overtaking D. As a result you should draw another A and connect it to B and then connect new E to that A and new C to new E. See Figure 3.

Row 4:

Then row 4 contain B,D,A. Now we can just rename the frequency of occurrences in the existing branch. As B:4,D,A:3.

Row 5:

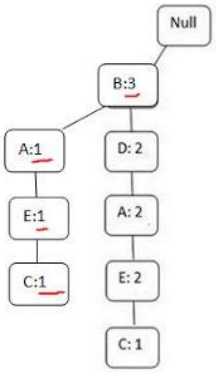

Figure 3 - After adding third row

In fifth raw have only item D. Now we have opportunity draw new branch from 'null' node. See Figure 4.

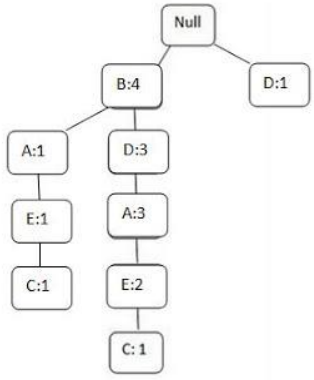

Figure 4- Connect D to null node

Row 6:

B and D appears in row 6. So just change the B:4 to B:5 and D:3 to D:4.

Row 7:

Attach two new nodes A and E to the D node which hanging on the null node. Then mark D,A,E as D:2,A:1 and E:1.

Row 8 :(Ohh.. last row)

Attach new node C to B. Change the traverse times.(B:6,C:1)

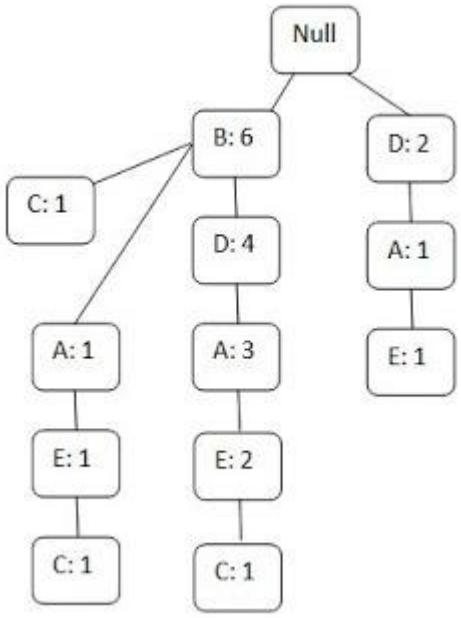

Figure 5 - Final FP tree

Step6-Validation

After the five steps the final FP tree as follows: Figure 5.

How we know is this correct?

Now count the frequency of occurrence of each item of the FP tree and compare it with Table 2. If both counts equal, then it is positive point to indicate your tree is correct.[3]

[**Reference:** Y. Yuan, C. Yang, Y. Huang, and D. Mining, And the Optimization Technology nd Its Application. Beijing. Science Press, 2007]

5. Constraint-Based Association Mining

A data mining process may uncover thousands of rules from a given set of data, most of which end up being unrelated or uninteresting to the users. Often, users have a good sense of which—direction of mining may lead to interesting patterns and the —form‖ of the patterns or rules they would like to find. Thus, a good heuristic is to have the users specify such intuition or expectations as *constraints*to confine the search space. This strategy is known as

constraint-based mining. The constraints can include the following:

- **Knowledge type constraints:** These specify the type of knowledge to be mined, such as association or correlation.
- **Data constraints:** These specify the set of task-relevant data.
- **Dimension/level constraints:** These specify the desired dimensions (or attributes) of the data, or levels of the concept hierarchies, to be used in mining.
- **Interestingness constraints:** These specify thresholds on statistical measures of rule interestingness, such as support, confidence, and correlation.
- **Rule constraints:** These specify the form of rules to be mined. Such constraints may be expressed as metarules (rule templates), as the maximum or minimum number of predicates that can occur in the rule antecedent or consequent, or as relationships among attributes, attribute values, and/or aggregates.

Fig.5 constraint-based mining

5.1. Metarule-Guided Mining of Association Rules

"How are metarules useful?" Metarules allow users to specify the syntactic form of rules that they are interested in mining. The rule forms can be used as constraints to help improve the efficiency of the mining process. Metarules may be based on the analyst's experience, expectations, or intuition regarding the data or may be automatically generated based on the database schema

5.2. Metarule-guided mining:-

Suppose that as a market analyst for *AllElectronics*, you have access to the data describing customers (such as customer age, address, and credit rating) as well as the list of customer transactions. You are interested in finding associations between customer traits and the items that customers buy. However, rather than finding *all* of the association rules reflecting these relationships, you are particularly interested only in determining which pairs of customer traits SCE Department of Information Technology promote the sale of office software.A metarule can be used to specify this information describing the form of rules you are interested in finding. An example of such a metarule is

$$P_1(X,Y) \wedge P_2(X,W) \Rightarrow buys(X, \text{"office software"}),$$

where $P1$ and $P2$ are predicate variables that are instantiated to attributes from the given database during the mining process, X is a variable representing a customer, and Y and W take on values of the attributes assigned to $P1$ and $P2$, respectively. Typically, a user will specify a list of attributes to be considered for instantiation with $P1$ and $P2$. Otherwise, a default set may be used.

5.3. Constraint Pushing: Mining Guided by Rule Constraints

Rule constraints specify expected set/subset relationships of the variables in the mined rules, constant initiation of variables, and aggregate functions. Users typically employ their knowledge of the application or data to specify rule constraints for the mining task. These rule constraints may be used together with, or as an alternative to, metarule-guided mining. In this section, we examine rule constraints as to how they can be used to make the mining process more efficient. Let's study an example where rule constraints are used to mine hybrid-dimensional association rules.

Our association mining query is to *"Find the sales of which cheap items (where the sum of the prices is less than $100) may promote the sales of which expensive items (where the minimum price is $500) of the same group for Chicago customers in 2004."* This can be expressed in the DMQL data mining query language as follows,[3]

(1) mine associations as
(2) $lives\_in(C, \_, \text{"Chicago"}) \wedge sales^{+}(C, ?\{I\}, \{S\}) \Rightarrow sales^{+}(C, ?\{J\}, \{T\})$
(3) from sales
(4) where S.year = 2004 and T.year = 2004 and I.group = J.group
(5) group by C, I.group
(6) having sum(I.price) < 100 and min(J.price) ≥ 500
(7) with support threshold = 1%
(8) with confidence threshold = 50%

Fig:6

Summary

In this chapter, we looked at the following topics:

Market basket analysis

- As the first step of association rule mining, the frequent itemset is the key factor. Along with the algorithm design, closed itemsets and maximum frequent itemsets are defined too.
- As the target of association rule mining, association rules are mined with the measure of support count and confidence. Correlation rules mining are mined with the correlation formulae, in addition to the support count.
- Monotonicity of frequent itemset; if an itemset is frequent, then all its subsets are frequent.

- The A-Priori algorithm, which is the first efficient mining algorithm to mine frequent patterns; many variants originated from it.

- Frequent patterns in sequence.

REFERENCES

[1] J. Han and M. Kamber, Conception and Technology of Data Mining, Beijing: China Machine Press, 2007.

[2] J. N. Wong, translated, Tutorials of Data Mining. Beijing. Tsinghua University Press, 2003.

[3] Y. Yuan, C. Yang, Y. Huang, and D. Mining, And the Optimization Technology nd Its Application. Beijing. Science Press, 2007.

[4] Y. S. Kon and N. Rounteren, "Rare association rule mining and knowledge discovery: technologies for frequent and critical event detection. H ERSHEY," PA: Information Science Reference, 2010

[5]. http://www.ijcce.org/papers/128-B047.pdf

[6] https://arxiv.org/pdf/1403.3948

[7].https://www.brainkart.com/article/Constraint-Based-Association-Mining_8319/