*International Journal of Research in Advent Technology, Special Issue, March 2019*
*E-ISSN: 2321-9637*
*Available online at www.ijrat.org*

# Market Basket Analysis Based on Association Rule Mining by Using Apriori and fp-growth Algorithm

**Ankit Rathore[1], Gaurav Raj[2]**
[*]*Jaypee University Anoopshahr,*
*Bulandshahr, India*
*ankitrath99@gmail.com[1]*

**Abstract**-- The proposed paper focuses on the Analysis of buying habits of customers or market basket content. Market Basket Analysis(MBA) is a data mining technique that can be used for identify relationships between different items and is also known as association rule learning or affinity analysis.The goal of MBA is to determine the placement of goods, next best option for the customers, designing sales promotions for different segments of customers to improve customer satisfaction and hence the profit of the supermarket. Here, we apply Apriori algorithm and FP-Growth algorithm on the various data collected from different stores in order to trace the various association rules comprising of a basket.One discrete advantage of using FP-Growth algorithm instead of Apriori algorithm is that it avoids the generation of candidate sets, which is computationally exhaustive. The results and conclusions drawn can be used in optimizing the market.This will help in predicting future trends and behaviors, allowing businesses to make knowledge-driven decisions and also provides a new way to design store layout by placing in propinquity those items that are purchased together.

**Keywords**-Market basket analysis, Association rule, affinity analysis, Apriori algorithm, FP-Growth algorithm, candidate sets.

## 1. INTRODUCTION

Market basket analysis is based upon the identification of purchasing patterns of the customers. In this technique, the purchasing patterns of the customers are analyzed by identifying: which items are purchased together; which item is purchased next to the other item; which items are purchased in different seasons. This process tends to yield very useful information that if the customers are buying some set of items,what is the probability that they will buy some other set of items. For example:- in winter season, if customers are purchasing jackets, how probably will they also purchase sweaters? This information helps in increasing the sales of such item-sets and thus increases the total profit by re-designing the store layouts, i.e., by placing in propinquity those items that are purchased together. Suppose a manager of some shopping mall would like to identify the purchasing patterns of his customers. In a more specific manner, he tends to find "Which set of items will the customers purchase on a given visit to the store?" To do this, market basket analysis must be performed on the database containing different customer transactions at the store. These transactions can happen at different times of the day, week,month or year. After getting the results, the advertising or marketing strategies are planned and also the layout of the store is redesigned. For example: If there is a high probability that customers who buy jeans will also buy shirts, then it is better to place the shirts nearer to the jeans so as to increase the sales of both. But if we put these items at different places, then it may reduce the sales of both. Another strategy that can be applied is the recommendation of products: the less purchased products are recommended to customers when they are going to purchase some other products and also the discounts may be given on such products. This will also ultimately lead to the increase in sales and, therefore, results in the increase in overall profit. Although market basket analysis has many benefits, yet there are some problems associated with the market basket analysis that need to be dealt with. The first problem that needs to be dealt with is the varying needs of the customers with respect to seasons and time. Consequently, their purchasing patterns will change. Thus there arises the need of market basket analysis to be done again and again on the whole database containing different customer transactions. Now if the database is large in size, then it will take more time and will be more costly to do the market basket analysis on the whole dataset. That is, instead of increasing profit, this repeated need of market basket analysis may decrease profit.

## 2. DATA MINING

In general, the function of the process of data mining can be classified into two categories: descriptive and predictive. Predictive function provides global rules can be applied to the database. The descriptive function aims to provide a description of the available data sources. Descriptions are provided in summary form dense clusters that provide information, linkage, association or other forms.

### *A. Association Rules Mining*

Association rule mining is a procedure which is looking for a relationship among an item with other items. Association rule is usually used "if" and "then" ,this shows if A then B and C. To determine the Association's rules. it needs to be specified the support and confidence to restrict whether the rule is interesting or not. ARM is a two step process consisting of extraction of all frequent item sets followed by extraction of strong association rules from the obtained frequent item sets. Association rules are interesting or frequent if their support and confidence are greater than minimum support and confidence. The item sets that are expected to be frequent are known as candidate item sets.

**Support:** A measure that indicates how much the level of dominance of an item or itemset of the overall transaction.

**Confidence:** A measure that shows the relationship between items in a conditional. (e.g. how frequently purchased item B if the person buying the item A) .

**Example:** buy (x, "Eggs") buy (x, "bread") [support =50%, confidence = 80%]

Eggs and bread are purchased simultaneously by 50% of all transactions and 80% of all consumers who buy eggs also buy bread.

*Working Of Association Rules Mining Algorithm*

Step-1 :- Assume k=1, and generate frequent item sets of length 1.

Step-2 :- Repeat till new frequent item sets are found.

- Obtain (k+1) candidate item sets from k-length frequent item sets.
- Prune infrequent candidate item sets containing subsets of length k.
- Calculate the support for each candidate item set by passing over the database.
- Item sets not having minimum support are discarded and leaving only frequent item sets called k-item sets.

1) $L_1$ = find_frequent_1-item-sets(D);
2) for ($k = 2$; $L_{k-1} \neq \emptyset$; k++) {
3) $C_k$ = apriori_gen($L_{k-1}$);
4) for each transaction $t \in D$ { //scan D for counts
5) $C_t$ = subset($C_k$ ,t); //get the candidate subsets of t
6) for each candidate $c \in C_t$
7) c.count++;
8) }
9) $L_k = \{c \in C_k \mid c.count \geq min\_sup\}$
10) }
11) return $L = \cup_k L_k$ ;

*Measures of Effectiveness of the Rule*

The measures of effectiveness of the rule by using an example dataset.

| TID | Itmes |
|---|---|
| 1 | Bread,Milk |
| 2 | Bread,Diaper,Beer,Eggs |
| 3 | Milk,Diaper,Beer,Coke |
| 4 | Bread,Milk,Diaper,Beer |
| 5 | Bread,Milk,Diaper,Coke |

1) for each item-set $l_1 \in L_{k-1}$
2) for each item-set $l_2 \in L_{k-1}$
3) if ($l_1[1] = l_2[1]$) ^ ($l_1[2] = l_2[2]$) ^...^ ($l_1[k-2] = l_2[k-2]$) ^ ($l_1[k-1] < l_2[k-1]$) then {
4) $c = l_1 \bowtie l_2$; // join step: candidates generation
5) if has_infrequent_subset(c, $L_{k-1}$) then
6) delete c; // remove unfruitful candidate
7) else add c to $C_k$;
8) }
9) return $C_k$;

Support can be calculated as the fraction of rows containing both A and B or joint probability of A and B. Among rows containing A, Confidence is the fraction of rows containing B or conditional probability of B given A.

**{Diaper,Beer}=>{Milk}**
**[support =2/5, confidence = 2/3]**
**{Milk}=>{Diaper,Beer}**
**[support =2/5, confidence = 2/4]**
**{Milk,Diaper}=>{Bread}**
**[support =2/5, confidence = 2/3]**

Lift is the ratio Confidence is to Support. If the lift is < 1 then A and B are negatively correlated else positively correlated and if it is equal to 1 it is not correlated.

***B. Apriori Algorithm***

Apriori algorithm is used for finding frequent item-sets for association rules among the item-sets in a database of customer transactions. Apriori algorithm uses a level-wise search, which is an iterative approach. In this approach, k item-sets are used to find (k+1)-item-sets. Firstly, the set of frequent 1-item-sets is found by scanning the entire database of customer transactions to find the count for each candidate item-set, and comparing the candidate support count with minimum support count, and then collecting those items which satisfy minimum support count. This set of frequent 1-itemsets is denoted by L1. Then this set of frequent 1-item-sets(L1) , is used for finding set of frequent 2-item-sets(L2), which is then used for finding set of frequent 3-item-sets(L3), and so on, until no more frequent k-item-sets can be found. In order to find each Lk, we need to scan the entire database of customer transactions. The main idea of Apriori algorithm is as under:

Input: D (Database containing customer transactions).
min_sup: (minimum support count).
Output: L (frequent item-sets).

The procedure apriori_gen(Lk-1) is used for candidates generation and for elimination of those candidates which are having infrequent subset. It is presented below: procedure apriori_gen(Lk-1:frequent (k-1)-item-sets)

The procedure used to test for infrequent subsets is given ahead:

procedure has_infrequent_subset(c: candidate k-item-set; Lk-1:frequent (k-1)-item-sets); // use prior knowledge

while doing market basket analysis using Apriori algorithm, we need to find candidate sets and frequent item sets repeatedly depending upon the input item-set of customer transactions. Thus in Apriori algorithm, most of the time is consumed in scanning the entire database of customer transactions again and again. Because of these problems, there is a need of an efficient method which reduces the time taken in repeated scanning of the database and increases the efficiency of the algorithm. To overcome from this problems. We will use single layer feedforward partially connected neural network.Feed-forward network is that neural network which applies supervised learning process to the training set, i.e., the outputs for the input training set are already known and as desired. Single layer feed-forward neural network is that neural network which has the single

1) *for each (k-1)-subset s of c*
2) *if* $s \notin L_{k-1}$ *then*
3) *return TRUE;*
4) *return FALSE;*

output layer of computation nodes, i.e., neurons and has no hidden layers.To train a feed-forward network, we need to adjust the network, i.e., we need to adjust the weights by doing back propagation so that we can get the desired output for each input training set. Thus by only adjusting the weights of a neural network we can eliminate the need of market basket analysis to be done again and again. This in turn will lead to the reduction in both cost and time applied in doing repeated market basket analysis over large database of customer transactions.

### *C. **FP-Growth Algorithm***

FP-growth algorithm is used for finding frequent item set that does not have the candidate keys. It constructs the data construction (FP-tree) which is compressed highly, and it reduces the original database and convert into FP-tree. FP-growth (frequent pattern growth) uses an extended prefix-tree (FP-tree) structure to store the database in a compressed form. FP-growth adopts a divide-and-conquer approach to decompose both the mining tasks and the databases. It uses a pattern fragment growth method to avoid the costly process of candidate generation. FP-growth algorithm scans database altogether two times[8]. The first scanning database, we can obtain the frequent 1-itemset,and the second scanning database, we can get the database's non-frequent itemset by using of frequent 1-itemset, and FP-tree is generated simultaneously. Last, we can obtain the association rule by using of the FP-tree.

*Primitive Transactional Database*

Let the transactional database have 5 transactions. We TID Items bought (ordered)frequent items use Table I to illustrate the FP-Growth algorithm for finding frequent item-sets.

**Table I: Sample Database**

| TID | Items bought |
|---|---|
| 100 | {f,a,c,d,g,i,m,p} |
| 200 | {a,b,c,f,l,m,o} |
| 300 | {b,f,h,j,o} |
| 400 | {b,c,k,s,p} |
| 500 | {a,f,c,e,l,p,m,n} |

*Frequent-item-header Table*

Let the minimum support threshold be 3(i.e., min sup=3/5=60%). First, a scan of DB derives a list of frequent items, (f :4), (c:4), (a:3), (b:3), (m:3), (p:3) | (the number after ":" indicates the support), in which items are ordered in frequency descending order. This ordering is important since each path of a tree will follow this order.

**Table II: Header Table**

| Item | frequency |
|---|---|
| c | 4 |
| f | 4 |
| a | 3 |
| b | 3 |
| m | 3 |
| p | 3 |

*FP-Tree*

For convenience of later discussions, the frequent items in each transaction are listed in this ordering in the rightmost column of Table III. With the above observations, one may construct a frequent-pattern tree

**Table III:A Transaction Database**

| TID | Items bought | (ordered)frequent Items |
|---|---|---|
| 100 | {f,a,c,d,g,l,m,p} | {c,f,a,m,p} |
| 200 | {a,b,c,f,l,m,o,} | {c,f,a,b,m} |
| 300 | {b,f,h,j,o} | {f,b} |
| 400 | {b,c,k,s,p} | {c,b,p} |
| 500 | {a,f,c,e,l,p,m,n} | {c,f,a,m,p} |

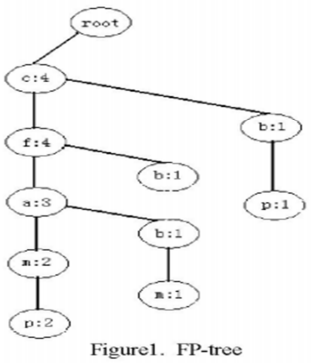


Figure1. FP-tree

*Creating Conditional Pattern-bases by FP-Tree*

From the beginning of the Header-Table, FP-Tree is scanned according to the link of every frequent item, and all the prefix paths which reach the frequent item are list. The conditional pattern-bases generated are summarized in table IV.

**Table IV: A Condition Pattern Base Table**

| Head Item | Condition pattern base |
|---|---|
| f | c:4 |
| a | cf:3 |
| b | cfa:1,cf:1,c:1 |
| M | cfa:2,cfab:1 |
| P | cfam:2,cb:1 |

*Frequent Item-set*

In the condition pattern base, from the beginning of the p-item, every item of the condition pattern base is scanned, and the common content is list, including a individual item and the combination. At the same time, the number of the corresponding item is counted and the condition FP-Tree is generated. And then FP-Tree is connected with the corresponding header-Table-item, and eventually frequent item-sets are generated. As shown in table V.

Table V: Frequent Item-set

| *Head Item* | *condition pattern base* | *condition* **FP-Tree** | **Frequency Item** |
|---|---|---|---|
| *p* | *cfam:2, cb:1* | *< c:3 >* | *cp:3* |
| *m* | *cfa:2, cfab:1* | *<c:3,f:3,a:3,cf:3,ca:3,fa:3, fa:3>* | *cm:3,fm:3,am:3,cfm:3,cam:3 ,fam:3,cfam:3* |
| *b* | *cfa:1, cf:1, c:1* | *< c:3 >* | *cb:3* |
| *a* | *cf:3* | *< cf:3 >* | *ca:3,fa:3,cfa:3* |
| *f* | *c:4* | *< c:4 >* | *cf:4* |
| *c* | *Φ* | *Φ* | *Φ* |

## 3. RESULTS

I used Jupyter NoteBook as a software to perform our work. I written the Code in Python language and created a program. which finds the candidate sets of items. Figure 2 given below shows the results of market basket analysis:

```
Out[12]: [RelationRecord(items=frozenset({'chicken', 'light cream'}), support=0.00659681800543267 3, ordered_statistics=[OrderedStatist
ic(items_base=frozenset({'light cream'}), items_add=frozenset({'chicken'}), confidence=0.2931034482758620 4, lift=3.4888110217
40861)]),
 RelationRecord(items=frozenset({'escalope', 'mushroom cream sauce'}), support=0.0069848661233993 01, ordered_statistics=[Orde
redStatistic(items_base=frozenset({'mushroom cream sauce'}), items_add=frozenset({'escalope'}), confidence=0.288, lift=3.4519
813953488367)]),
 RelationRecord(items=frozenset({'pasta', 'escalope'}), support=0.0050446255335661 62, ordered_statistics=[OrderedStatistic(it
ems_base=frozenset({'pasta'}), items_add=frozenset({'escalope'}), confidence=0.2888888888888889, lift=3.4626356589147287)]),
 RelationRecord(items=frozenset({'fromage blanc', 'honey'}), support=0.0048506014745828 48, ordered_statistics=[OrderedStatist
ic(items_base=frozenset({'fromage blanc'}), items_add=frozenset({'honey'}), confidence=0.2475247524752475, lift=3.8081867888 2
81358)]),
 RelationRecord(items=frozenset({'olive oil', 'whole wheat pasta'}), support=0.0110593713620488 94, ordered_statistics=[Ordere
dStatistic(items_base=frozenset({'whole wheat pasta'}), items_add=frozenset({'olive oil'}), confidence=0.2714285714285714 6, l
ift=3.081371932032725)]),
 RelationRecord(items=frozenset({'pasta', 'shrimp'}), support=0.0071788901823826 15, ordered_statistics=[OrderedStatistic(item
s_base=frozenset({'pasta'}), items_add=frozenset({'shrimp'}), confidence=0.4111111111111115, lift=4.138411458333334)]),
 RelationRecord(items=frozenset({'burgers', 'eggs', 'almonds'}), support=0.0032984090027163367, ordered_statistics=[OrderedSt
atistic(items_base=frozenset({'eggs', 'almonds'}), items_add=frozenset({'burgers'}), confidence=0.3469387755102041, lift=3.03
58615432590694)]),
```

Figure 2. Output of market basket analysis

## 4. CONCLUSION

Market basket analysis is a technique for the identification and analysis of purchasing patterns of the customers. There are some problems associated with market basket analysis. The first one is that the needs of the customers keep on changing with respect to seasons and time. So we need to perform it repeatedly. Another problem is related to Apriori algorithm which requires a repeated scan of the entire database of customer transactions to find candidate sets and frequent itemsets. To overcome this problem. We use Fp-growth algorithm, which takes only two scan of the database and it reduces the time taken in repeated scanning and also increases the efficiency of the algorithm.

1) FP-Growth beats Apriori by far. It has less memory usage and less runtime. The differences are huge.
2) FP-Growth is more scalable because of its linear running time.

**REFERENCES**

[1] Yongmei Liu and Yong Guan "FP-Growth Algorithm for Application in Research of Market Basket Analysis" , 2009 WRI World Congress on Computer Science and Information Engineering
[2] Anshul Bhargav, Robin Prakash Mathur, Munish Bhargav "Market basket ananlysis using artificial neural network",International Conference for Convergence for Technology-2014
[3] JIAWEI HAN, JIAN PEI, YIWEN YIN, RUNYING MAO Mining Frequent Patterns without Candidate Generation: A Frequent-Pattern Tree Approach. Data Mining and Knowledge Discovery, 8, 53-87, 2004.