EXPT. NO. NAME

Page No.

Title : Case Study (Market Basket Analysis)

Problem Statement : A Mall has number of items for sale. Build a required Database to develop BA & I tool considering one aspect of growth to the business such as organization of products based on demand & patterns.

Input : Transaction Database & minimum Support

Output : Frequent item sets, Association Rules & graphical representation of rules as per confidence & lift

Pre-Lab : 1. knowledge of R programming language
2. Concept & theory of Apriori algorithm.

Theory :

By convention, the algorithm assumes that items within a transaction or itemset are sorted in lexicographic order. It employs an iterative approach known as a level-wise search, where (k-1) itemsets are used to explore k itemsets. First, the set of frequent 1-item sets is found by scanning the database to accumulate the count for each item, and collecting those items that satisfy minimum support. The resulting set is denoted as L_1.

Next L_1 is used to find L_2, the set of frequent 2-itemsets, which is the used to find L_3, and so on, until no more frequent k-itemsets can be found. The finding of each L_k requires one full scan of D.

To improve the efficiency of the level-wise generation of frequent itemsets, an important property called Apriori property is used to reduce the search space.

Apriori property: All nonempty subsets of a frequent itemset must also be frequent

This property is based on the following observation. If an itemset A does not satisfy the minimum support threshold, min_sup, then A is not frequent; i.e. $P(A) < min\_sup$. If an item B is added to the itemset A, then the resulting itemset AUB cannot occur more frequently than A. Therefore, AUB is not frequent either, that is, $P(AUB) < min\_sup$.

This property belongs to a special category of properties called antimonotone in the sense that if a set cannot pass a test, then all of its supersets will fail the same test as well.

EXPT NO | NAME | Page No

A two-step process is used to find L_k from L_{k-1}, for $k \geq 2$:

1. The join Step: To find L_k, a set of Candidate k-itemsets is generated in L_{k-1}. The notation $l_i[j]$ refers to the j^{th} item in l_i. Thus in l_i, the last item and the next to the last item are given respectively by $l_i[k-1]$ & $l_i[k-2]$. Any two itemsets l_1 & l_2 are joined if their first $(k-2)$ items are in common. That is, members '1' & '2' are joined if $(l_1[1] = l_2[1]) \wedge (l_1[2] = l_2[2]) \wedge \cdots \wedge (l_1[k-2] = l_2[k-2]) \wedge (l_1[k-1] < l_2[k-1])$. The condition $l_1[k-1] < l_2[k-1]$ simply ensures that no duplicates are generated. The resulting itemset formed by joining l_1 & l_2 is $\{l_1[1], l_1[2], \cdots l_1[k-2], l_1[k-1], l_2[k-1]\}$.

2. The prune step: Set C_k is superset of L_k, because although all the frequent k-itemsets are included in C_k, its members may or may not be frequent. One could scan the database to determine the count of each candidate in C_k & eliminate any itemset that does not meet the minimum support threshold. This would then give L_k. However,

C_k can be huge, and so this could be very time-consuming. To eliminate the infrequent itemsets, the Apriori property is used as follows. Any (k-1) itemset that is not frequent cannot be a subset of frequent k-itemset. This subset Testing can be done quickly by maintaining a hash tree of all frequent itemsets.

An example of the Use of the Apriori Algorithm:

We illustrate the use of the Apriori algorithm for finding frequent itemsets in our transaction database, D. In the first iteration of the algorithm, each item is a member of the set of candidates 1-itemsets, C1. The algorithm simply scans all the transactions in order to count the number of occurences of each item.

C1 itemset	Support count
{1}	6
{2}	7
{3}	6
{4}	2
{5}	2
{6}	1

Teacher's Signature:

The set of frequent 1-itemsets, L1, consists of the candidate itemsets satisfying the minimum support count of 2. Thus all the candidates in C1, except for {6}, are in L1.

L1 itemset	Support Count
{1}	6
{2}	7
{3}	6
{4}	2
{5}	2

To discover the set of frequent 2-itemsets, L2, the algorithm joins L1 with itself to generate a candidate set of 2-itemsets. C2. Note that no candidates are removed from C2 during the pruning step since each subset of the candidates is also frequent

C2 itemset
{1, 2}
{1, 3}
{1, 4}
{1, 5}
{2, 3}
{2, 4}
{2, 5}
{3, 4}
{3, 5}
{4, 5}

Next the transactions in D are scanned & the support count of each candidate itemset in C2 is accumulated

C2 itemset	Support count
{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

The set of frequent 2-itemsets, L2, is then determined, consisting of those candidate 2-itemsets in C2 having minimum support.

L2 itemset	Support count
{1, 2}	4
{1, 3}	4
{1, 5}	2
{2, 3}	4
{2, 4}	2
{2, 5}	2

Teacher's Signature:

Next C_3 is generated by joining L_2 with itself. The result is $C_3 = \{\{1,2,3\}, \{1,2,5\}, \{1,3,5\}, \{2,3,4\}, \{2,3,5\}, \{2,4,5\}\}$. C_3 is then pruned using the Apriori property: All nonempty subsets of frequent itemsets must also be frequent. From the way each candidate of C_3 is formed, it is clear that all we need to check is the subset obtaind from the last two members of the candidate set.
Since $\{2,3\}$ is a frequent itemset, we keep $\{1,2,3\}$ in C_3.
Since $\{2,5\}$ is a frequent itemset, we keep $\{1,2,5\}$ in C_3.
Since $\{3,5\}$ is not a frequent itemset, we remove $\{1,3,5\}$ from C_3.
Since $\{3,4\}$ is not a frequent itemset, we remove $\{2,3,4\}$ from C_3.
Since $\{3,5\}$ is not a frequent itemset we remove $\{2,3,5\}$ from C_3.
Since $\{4,5\}$ is not a frequent itemset we remove $\{2,4,5\}$ from C_3.
Therefore after pruning C_3 is given by:

C_3 itemset
$\{1,2,3\}$
$\{1,2,5\}$

The transactions in D are scanned to determine L_3, consisting of those candidates

3- itemsets in C3 having at least minimum support

C3 itemset	Support Count
{1, 2, 3}	2
{1, 2, 5}	2

Since both 3-itemsets in C3 have the least minimum support, L3 is therefore given by:

L3 itemset	Support count
{1, 2, 3}	2
{1, 2, 5}	2

Finally L3 is joined with itself to generate a candidate set of 4-itemsets, C4. This results in a single itemset {1, 2, 3, 5}. However this itemset is pruned since its subset {3, 5} is not frequent. Thus, C4 = ϕ & the algorithm terminates, having found all of the frequent itemsets.

Analysis: 1. Observe the graph for generated rules with different support confidence & lift.
2. Observe top rules & use this patterns for organization of products.

Teacher's Signature:

Page No.:

Conclusion : Thus the Groceries dataset is used to generate rules & applied rules for organization of products based on demand & patterns. Frequent itemsets are found using apriori algorithm based on association rule data mining technique. Observations are recorded in terms of graph.