

## Experiment 6 - Implementation of Association Rule Mining Algorithm

Name: Shaikh Shadab Rollno : 17DCO74  
Class : TE.CO Batch : B3

### #Theory

#### ➤ Association Rule

Association rule mining finds interesting associations and relationships among large sets of data items. This rule shows how frequently a itemset occurs in a transaction. A typical example is Market Based Analysis. The Association rule is very useful in analyzing datasets. The data is collected using bar-code scanners in supermarkets. Such databases consists of a large number of transaction records which list all items bought by a customer on a single purchase. So the manager could know if certain groups of items are consistently purchased together and use this data for adjusting store layouts, cross-selling, promotions based on statistics.

Market Based Analysis is one of the key techniques used by large relations to show associations between items. It allows retailers to identify relationships between the items that people buy together frequently.

Given a set of transactions, we can find rules that will predict the occurrence of an item based on the occurrences of other items in the transaction.

TID	ITEMS
1	Bread, Milk
2	Bread, Diaper, Beer, Eggs
3	Milk, Diaper, Beer, Coke
4	Bread, Milk, Diaper, Beer
5	Bread, Milk, Diaper, Coke

#### ➤ Basic definition

Support Count( $\sigma$ ) – Frequency of occurrence of a itemset.

Here  $\sigma$  ({Milk, Bread, Diaper})=2

Frequent Itemset – An itemset whose support is greater than or equal to minsup threshold.

Association Rule – An implication expression of the form  $X \rightarrow Y$ , where X and Y are any 2 itemsets. Example: {Milk, Diaper}  $\rightarrow$  {Beer}

➤ Rule Evaluation Metrics

Support(s) – The number of transactions that include items in the {X} and {Y} parts of the rule as a percentage of the total number of transaction. It is a measure of how frequently the collection of items occur together as a percentage of all transactions.

Support =  $\sigma(X+Y) \div \text{total}$  – It is interpreted as fraction of transactions that contain both X and Y

Confidence(c) – It is the ratio of the no of transactions that includes all items in {B} as well as the no of transactions that includes all items in {A} to the no of transactions that includes all items in {A}.

Conf(X=>Y) =  $\text{Supp}(X \cup Y) \div \text{Supp}(X)$  – It measures how often each item in Y appears in transactions that contains items in X also.

Lift(l) – The lift of the rule X=>Y is the confidence of the rule divided by the expected confidence, assuming that the itemsets X and Y are independent of each other. The expected confidence is the confidence divided by the frequency of {Y}.

Lift(X=>Y) =  $\text{Conf}(X=>Y) \div \text{Supp}(Y)$  – Lift value near 1 indicates X and Y almost often appear together as expected, greater than 1 means they appear together more than expected and less than 1 means they appear less than expected. Greater lift values indicate stronger association.

➤ Apriori Algorithm

Apriori algorithm is an influential algorithm for mining frequent item sets for Boolean association rules. It uses a bottom-up approach, where frequent subsets are extended one item a time (a step known as candidate generation, and groups of candidates are tested against the data). It is designed to operate on database containing transactions (for example, collections of items bought by customers, or details of a website frequentation) Apriori algorithm is given by R. Agrawal and R. Srikant in 1994 for. Name of algorithm is Apriori is because it uses prior knowledge of frequent itemset properties. We apply a iterative approach or level-wise search where k-frequent itemsets are used to find k+1 itemsets.

To improve the efficiency of level-wise generation of frequent itemsets an important property is used called Apriori property which helps by reducing the search space.

➤ 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 property). If a itemset is infrequent all its supersets will be infrequent.

## #Algorithm

Step 1 – Scan the transaction data base to get the support of S each 1 -itemset, compare S with min\_sup, and get a support of 1-item sets, L<sub>1</sub>

Step 2 – Use L<sub>k-1</sub> join L<sub>k-1</sub> to generate a set of candidate k-itemsets. And use Apriori property to prune the unfrequented k-itemsets from this set

Step 3 – Scan the transaction database to get the support S of each candidate k-itemset in the find set, compare S with min\_sup, and get a set of frequent k-itemsets L<sub>k</sub>

Step 4 – Check if the candidate set is null, if no then goto step 2

Step 5 – For each frequent itemset 1, generate all non empty subsets of 1

Step 6 – For every nonempty subsets of 1, output the rule  $\tilde{s} \Rightarrow [1-s]$  if confidence C of the rule  $\tilde{s} \Rightarrow [1-s]$  ( $= \text{support } s \text{ of } 1 / \text{support } S \text{ of } s$ )  $\geq \text{min\_conf}$

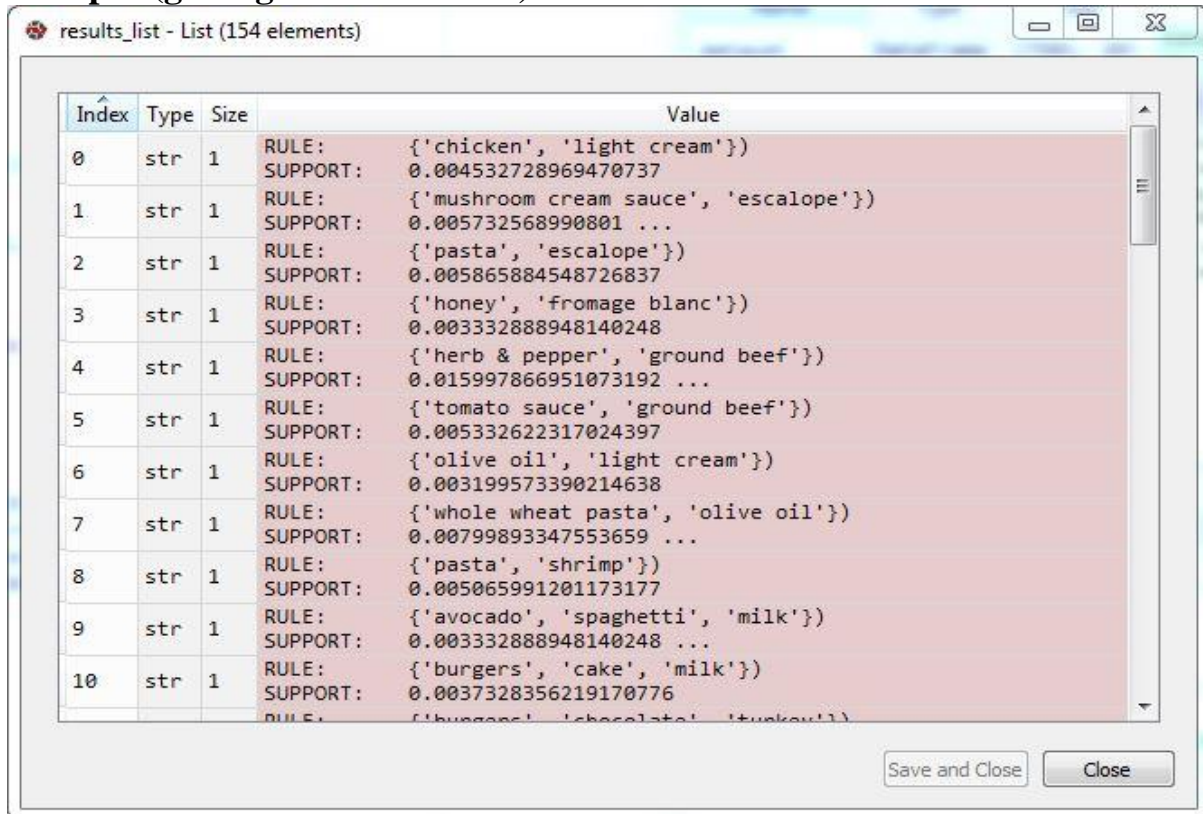
## #Source Code

```
#importing the libraries
import numpy as np
import matplotlib.pyplot as plt
import pandas as pd

#importing the dataset
dataset=pd.read_csv('Market_Basket_Optimisation.csv', header = None)
transactions=[]
for i in range (0,7501):
    transactions.append([str(dataset.values[i,j]) for j in range(0,20)])
#Getting data in the form of list column in row comprising of 20


#training Apriori on the dataset
from apyori import apriori
rules = apriori(transactions,min_support=0.003, min_confidence=0.2, min_lift=3,
min_length=2)
#visualizing results
results=list(rules)
results_list=[]
for i in range(0,len(results)):
    results_list.append('RULE:\t' + str(results[i][0]).replace('frozenset(',')' +
        '\nSUPPORT:\t' + str(results[i][1]) +
        '\nCONF:\t' + str(results[i][2][0][2]) +
        '\nLIFT:\t' + str(results[i][2][0][3]))
    #using replace for frozen set string
```

## #Output (getting the list of rules)



Index	Type	Size	Value
0	str	1	RULE: {'chicken', 'light cream'}) SUPPORT: 0.004532728969470737
1	str	1	RULE: {'mushroom cream sauce', 'escalope'}) SUPPORT: 0.005732568990801 ...
2	str	1	RULE: {'pasta', 'escalope'}) SUPPORT: 0.005865884548726837
3	str	1	RULE: {'honey', 'fromage blanc'}) SUPPORT: 0.003332888948140248
4	str	1	RULE: {'herb & pepper', 'ground beef'}) SUPPORT: 0.015997866951073192 ...
5	str	1	RULE: {'tomato sauce', 'ground beef'}) SUPPORT: 0.005332622317024397
6	str	1	RULE: {'olive oil', 'light cream'}) SUPPORT: 0.003199573390214638
7	str	1	RULE: {'whole wheat pasta', 'olive oil'}) SUPPORT: 0.00799893347553659 ...
8	str	1	RULE: {'pasta', 'shrimp'}) SUPPORT: 0.005065991201173177
9	str	1	RULE: {'avocado', 'spaghetti', 'milk'}) SUPPORT: 0.003332888948140248 ...
10	str	1	RULE: {'burgers', 'cake', 'milk'}) SUPPORT: 0.0037328356219170776

## #Output (Selecting one rule)



RULE:	{ 'pasta', 'shrimp' }
SUPPORT:	0.005065991201173177
CONF:	0.3220338983050847
LIFT:	4.506672147735896

## #Conclusion

In this experiment we have seen association mining algorithm, and have seen various definition related to this such as support, confidence, lift and have learned how apriori algorithm is used to find the rules to determine the frequency of occurrences of data.