INTELLIGENT SUPERMARKET USING APRIORI

Kasturi Medhekar[1], Arpita Mishra[2] , Needhi Kore[3] , Nilesh Dave[4]

[1,2,3,4]*Student, 3rd year Diploma, Computer Engineering Department, Thakur Polytechnic, Mumbai, Maharashtra, India.*

Abstract—*Modern organizations are geographically distributed. There is lot of online as well as offline purchase of data items taking place. Typically, each site locally stores its ever increasing amount of day-to-day data. Using centralized data mining to discover useful patterns in such organizations' data isn't always feasible because merging data sets from different sites into a centralized site incurs huge network communication costs. Data from these organizations are not only distributed over various locations but also vertically fragmented, making it difficult if not impossible to combine them in a central location. Distributed data mining has thus emerged as an active sub area of data mining research.*

Our proposed solution is going to apply the Apriori algorithm in distributed manner which is also called as Dipriori (Distributed Apriori) on real world large datasets with precision and with great speed. This approach can be used by both online shopping sites and offline marts.

Keywords: **Apriori, Dipriori, Distributed, Data mining, fragmentation, item sets.**

1. INTRODUCTION

Data mining is the buzzword in our recent technologies. Generally, data mining (sometimes called data or knowledge discovery) is the process of analysing data from different perspectives and summarizing it into useful information - information that can be used to increase revenue, cuts costs, or both. Apriori is the algorithm used to find frequent item sets which are further used to find meaningful patterns in datasets. It proceeds by identifying the frequent individual items in the database and extending them to larger and larger item sets as long as those item sets appear sufficiently often in the database. The frequent item sets determined by Apriori algorithm can be used to determine association rules which highlight general trends in the purchasing habit of customer.

There is a lot of data generated from shopping marts where there are several transactions carried out. Shopping marts like D-Mart and Big Bazaar have various branches. So we have proposed a system where these datasets will be mined and frequent item sets will be found out. But this will be carried out in a distributed manner parallel on more than one client. There would be average server which will calculate the average of that individual client's frequent item count. So for next set of transactions, this average count will be used. This will save the time of computation for all the transactions right from beginning.

Also this computation will be done on larger datasets and we will be using Distributed Apriori algorithm for this distributed computation.

2. REVIEW OF LITERATURE

A. STUDY OF EXISTING SYSTEM

Data mining is used for knowledge discovery in database i.e. data is mined from database and it generates information data which is used by humans effectively and efficiently. Data mining has area known as Association rule mining, which is based on pruning candidate keys. Hence in distributed Apriori algorithm, we first need to do data transformation i.e. data cleaning, integration and selection. We need to do frequent item sets mining where we get minimum support and confidence. First we join all the item set according to the algorithm and then we prune it as per our threshold value.Apriori algorithm is adopted to find the frequent item sets, then in order to get the global support and confidence without privacy disclosure, secure computation is used.

Many current data mining tasks can be accomplished successfully only in a distributed setting. The field of distributed data mining has therefore gained increasing importance in the last decade. This algorithm by Rakesh Agarwal is one of the best algorithms in terms of association rule mining It also serves as the base algorithm for most parallel algorithms. The enormity and high dimensionality of datasets typically available as input to problem of association rule discovery, makes it an ideal problem for solving on multiple processors in parallel.

Association rule mining [1] mining finds associations and/or correlation relationships among large set of data items. Association rules shows attribute value conditions that occur frequently together in a given dataset. A typical and widely used example of association rule mining is Market Basket Analysis. For example, data are collected using bar-code scanners in supermarkets .Databases of shopping markets have vast records. Each record lists all items bought by a customer on a single purchase transaction. Association rules provide information of this type in the form of "if-then" statements. These rules are computed from the data and association rules are probabilistic in nature unlike if-then. In addition to the antecedent (the "if" part) and the consequent (the "then" part), an association rule has two numbers that express the degree of uncertainty about the rule.

Support: In association analysis the antecedent and consequent called item sets do not have any items in common. The first number is called the support for the rule. The support just includes all the items that are included in the antecedent and consequent part. The support is sometimes expressed as a percentage of the total number of records in the database.

Confidence: The other number is known as the confidence of the rule. Confidence is the ratio of the number of transactions that include all items in the consequent as well as the antecedent (namely, the support) to the number of transactions that include all items in the antecedent.

Count Distribution Algorithm (CDA)[2] first computes support counts of 1-itemsets from each site in the same manner as it does for the sequential Apriori. It broadcasts the item sets to other sites and generates global frequent one item set . Subsequently, each site generates candidate 2- item sets and computes their support counts. At the same time, this algorithm also eliminates all globally infrequent 1-itemsets from every transaction and inserts the new transaction (that is, a transaction without infrequent 1- item set) into memory. While inserting the new transaction, it checks whether that transaction is already in the memory. If it is, CDA increases that transaction's counter by one. After generating support counts of candidate 2-itemsets at each site, CDA generates the globally frequent 2-itemsets. It then iterates through the main memory (transactions without infrequent 1-itemsets) and generates the support counts of candidate item sets of respective length. Next, it generates the globally frequent item sets of that respective length by broadcasting the support counts of candidate item sets after every pass.

Fast Distribution Mining algorithm [3] is a modification of CDA. Here in each iteration, support counts of all item sets are completed. Considering the local minimum support count for that node, the infrequent items are pruned locally and candidate set is generated. The node, at which an item sets is frequent, broadcasts requests to all other nodes for collecting their support counts for that item sets. Once the node receives the counts, it computes the global count for that candidate. It compares this count with the global minimum support count to check if the candidate is globally large. At the end of each iteration globally large item sets found are broadcasted to all the nodes.

Features:

FDM uses both local and global pruning.

This algorithm requires O (n) for communication

Optimized Distributed Association Rule Mining (ODAM)[2][4]algorithm involves a central server which receives candidate sets from all the other nodes and generates global frequent item sets. It eliminates infrequent items, thereby reducing dataset size significantly; so that we can accumulate more transactions in the main memory.

B. DRAWBACKS OF EXISTING SYSTEM

- Cannot be used with dynamic database.
- The entire algorithm needs to be executed again when new transactions are added.
- Rules generated are not well presented for human analysis.
- Too many or too less rules may be generated based on threshold values.
- Takes a lot of time to execute the entire algorithm and generate results.

3. PROPOSED SOLUTION

Our proposed solution involves modifying the ODAM algorithm to make it suitable for batch processing in a time efficient manner.

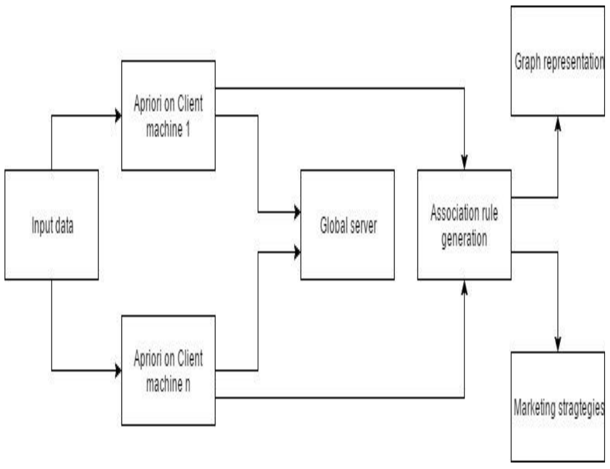

Graph representation
Apriori on Client machine 1
Input data
Global server
Association rule generation
Apriori on Client machine n
Marketing stragtegies

Figure -1: Block diagram of proposed solution.

Input data:

It contains the batch-wise data of transactions. The data is fragmented horizontally on the client machines. Then it is processed. Each batch is executed and control goes automatically to next batch of transactions. Minimum size of single batch is 1000. This dataset contains item labels at column headers and either 1 or 0 as column values. '1' indicates the presence of the item in the transaction and '0' indicates its absence.

Apriori on Client machines:

The Apriori algorithm is executed on the client machines and frequent item sets are generated locally. Then these item sets are sent over network using TCP/IP protocol to the Global server.
Also after data is received from the Global server, the client again iterate the process of Apriori for next set of transactions.

Global Server:

The local item counts are received from client machines. Then global server calculates the average of those item counts and the item sets greater than or equal to minimum support count are sent back to the client.

Association Rule Generation:

After the pair of frequent 3 item sets is generated by client and global server, the association rules are generated. These rules show the relevance of frequent items and help to filter the items as per the confidence.

Graph representation:

Based on association rules, frequent item sets are displayed using graphs for better understanding of market trend. Sales of different products can also be observed.

Marketing strategies:

Based on association rules, marketing strategies will be generated for example, the discount to offer on products and the combos to be provided to the customers. Customers can be notified about the discounts and offers via email and message.

A. BENEFITS OF PROPOSED SOLUTION

- ✓ Demonstration of the algorithm in an efficient way thus making the algorithm realistic.
- ✓ Saves bandwidth cost as only counts of frequent item sets are sent to global site instead of entire transaction.
- ✓ Benefit for the local as well as global manager to understand the pattern and increase the sales on both levels in distributed manner.
- ✓ Comparing recent trends with old trends is easily achieved and can be used to develop timeline analysis of changing business scenario.
- ✓ Recent trends can be combined with old trends in a space and time efficient manner.
- ✓ UI design will make the entire process of generating association rule very flexible.

4. PROPOSED ARCHITECTURE

This is the flow of our proposed architecture. It consists of several modules .There are various blocks such as input data at location 1 and 2,client machines of the locations, the global server who calculates the average frequent item sets which leads to the generation of association rules and accordingly graphs are generated and appropriate marketing strategies are planned. The customers will be notified about the updates as well.

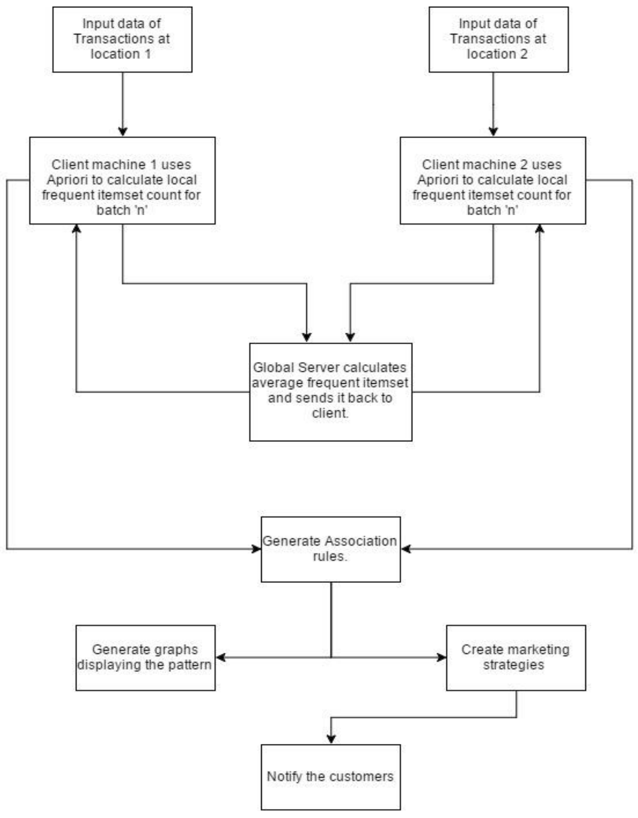

Input data of Transactions at location 1
Input data of Transactions at location 2
Client machine 1 uses Apriori to calculate local frequent itemset count for batch 'n'
Client machine 2 uses Apriori to calculate local frequent itemset count for batch 'n'
Global Server calculates average frequent itemset and sends it back to client.
Generate Association rules.
Generate graphs displaying the pattern
Create marketing strategies
Notify the customers

Figure -2: Flow graph of proposed architecture

We are making an application which will successfully demonstrate the flow and implementation of this algorithm in a distributed manner. Our proposed system will work in distributed manner on different client and server.

Following are the details of our proposed solution.

1. Entering transactions.
2. Calculating frequent set.
3. Showing local count and generate local pattern.
4. Then send local data and local count to the global server.
5. Organize data at global server.
6. Use Apriori algorithm at global server in distributed manner.

7. Calculate global count.
8. Show the global pattern.
9. Provide suggestions regarding marketing strategy.

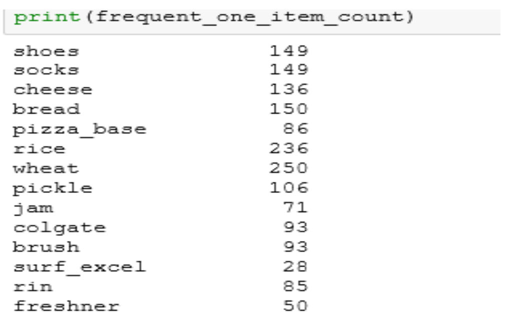
The result which we want to show through our application is the frequent item count resulting into meaningful patterns. The aspect which makes our system different is we are going to process this in a distributed manner. After the results are generated, we are going to suggest strategies which would help to improve the sales.

5. IMPLEMENTATION

We have implemented the DIPRIORI i.e. distributed apriori on our machines. We have considered 2 machines as clients and third machine as global server. The dataset which we provide as input has 7000 transactions of 40 data items. So totally we are working on 14000 transactions (7000 on each client).

The execution of our project happens in the following sequence.

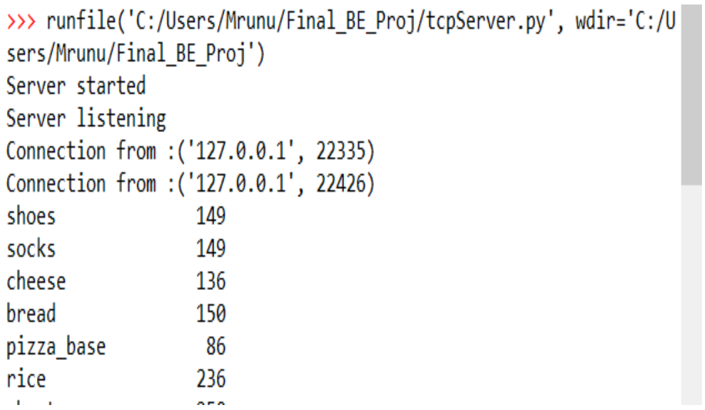
- The communication between clients and server is done through network switch and LAN cables. The clients calculate frequent item sets at the local machine.



```
print(frequent_one_item_count)

shoes          149
socks          149
cheese         136
bread          150
pizza_base      86
rice           236
wheat          250
pickle         106
jam             71
colgate         93
brush           93
surf_excel      28
rin             85
freshner        50
```

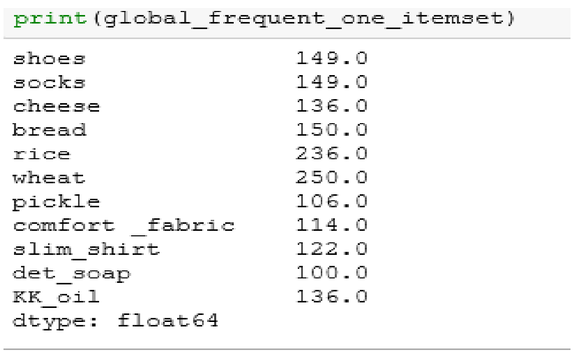
Figure -3: Item Count

- Then these item sets are sent to global server. Connection is established successfully and then server calculates the average at its end.



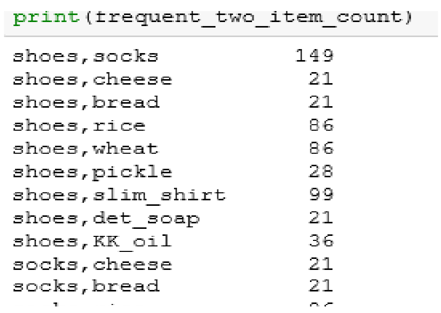
```
>>> runfile('C:/Users/Mrunu/Final_BE_Proj/tcpServer.py', wdir='C:/U
sers/Mrunu/Final_BE_Proj')
Server started
Server listening
Connection from :('127.0.0.1', 22335)
Connection from :('127.0.0.1', 22426)
shoes           149
socks           149
cheese          136
bread           150
pizza_base       86
rice            236
```

- After the data is received at client end, the clients calculate 2 items



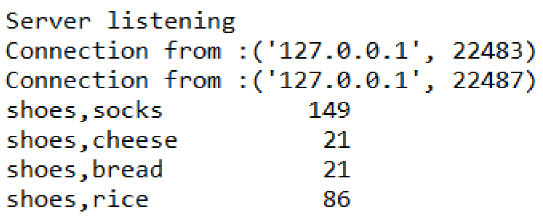
```
print(global_frequent_one_itemset)

shoes                 149.0
socks                 149.0
cheese                136.0
bread                 150.0
rice                  236.0
wheat                 250.0
pickle                106.0
comfort _fabric       114.0
slim_shirt            122.0
det_soap              100.0
KK_oil                136.0
dtype: float64
```

- Then two item sets are sent to server again



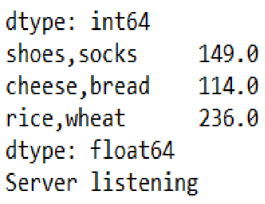
```
print(frequent_two_item_count)

shoes,socks            149
shoes,cheese            21
shoes,bread             21
shoes,rice              86
shoes,wheat             86
shoes,pickle            28
shoes,slim_shirt        99
shoes,det_soap          21
shoes,KK_oil            36
socks,cheese            21
socks,bread             21
```

- Then server calculates average of 2 item sets



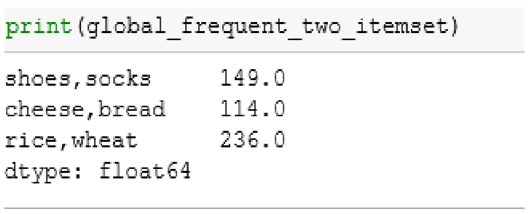
```
Server listening
Connection from :('127.0.0.1', 22483)
Connection from :('127.0.0.1', 22487)
shoes,socks          149
shoes,cheese          21
shoes,bread           21
shoes,rice            86
```

- Average is calculated:



```
dtype: int64
shoes,socks     149.0
cheese,bread    114.0
rice,wheat      236.0
dtype: float64
Server listening
```

- The clients receive these item sets and process further on those.



```
print(global_frequent_two_itemset)

shoes,socks      149.0
cheese,bread     114.0
rice,wheat       236.0
dtype: float64
```

And so on the flow goes till 3 pair of item sets are found.

6. CONCLUSION

Our main aim behind developing this project was to implement apriori algorithm in a time efficient and distributed manner. After doing thorough research and working on it for more than 9 Months, we have implemented Apriori algorithm on batch-wise data. We developed a code for Apriori algorithm which leveraged the power of the existing algorithm and added the functionality of implementing it on dynamic data. The proposed solution gives great results and is bound to improve on the time and space efficiency in processing batch-wise data.

The proposed solution helps in improving the algorithm by speeding up the computation of datasets available on different machines in a distributed manner.

The modules developed are of great importance:

The ‘Graph Generation’ module displays the association rules related to each item which helps the users to view the association rules of the item he is interested in, making analysis simpler.

The ‘Marketing strategies’ module helps the salesperson to devise various strategies and discount offers on items and then notify the customers about the discount which helps in tremendous increase in the sales of the super-mart.

7. FUTURE SCOPE

The following are the points which can be added to our project:

- Implementing it using Hadoop environment can further enhance the speed.
- The algorithm can be further modified to allow dynamic thresholding so that user does not have to worry about deciding the ideal threshold to generate appropriate association rules.
- Automating the batch process iterations after certain time period; eg. - Hourly basis, daily basis, etc.
- Modify the algorithm further for dynamic streaming of data from web so it can be used with e-commerce websites.
- Modify the algorithm to work with vertically fragmented database and hybrid-fragmented databases.

8. REFERENCES

[1] Qiankun Zhao and Sourav S. Bhowmick, “Association Rule Mining: A Survey”, Communication of the Association for Information Systems, 2003.

[2] Mafruz Zaman Ashrafi, David Taniar, Kate Smith, “ODAM: An Optimized Distributed Association Rule Mining Algorithm”, IEEE DISTRIBUTED SYSTEMS ONLINE 1541-4922 © 2004 Published by the IEEE Computer Society Vol. 5, No. 3; March 2004.

[3] Vinaya Sawant, Ketan Shah, “A Survey of Distributed Association Rule Mining Algorithms”, Journal of Emerging Trends in Computing and Information Sciences Vol. 5, No. 5, May 2014.