(IJAER) 2013, Vol. No. 6, Issue No. I, July **ISSN: 2231-5152**

DATA MINING MADE SIMPLE BY APRIORI ALGORITHM FOR MARKET ANALYSIS

*** Smt. Ch. Shobha Rani**
**** Smt. A. Aruna**
** Lecturer in Computer Science, ASM College for women, Warangal, Kakatiya University, Andhra Pradesh, India.*
*** Lecturer in Computer Science, ASM College for women, Warangal, Kakatiya University, Andhra Pradesh, India.*

ABSTRACT

Knowledge discovery is an emerging field which combines the techniques from mathematics, statistics, algorithms and Artificial Intelligence to extract the knowledge. Data mining is a main phase of Knowledge Discovery in Databases (KDD) for extracting the knowledge based on the patterns and their correlation by the application of appropriate association rules to the information's available from the data set. The outcome of the KDD is used to analyze or predict on the future aspects in any area of considerations. In this paper we propose an analysis and prediction of the market sales based on the historical information's from the database by considering the items information at different levels to generate the association rules. The widely used algorithm in data mining ie, apriori algorithm is specifically considered for the extraction of the knowledge.

INTRODUCTION

Knowledge discovery is an emerging field which combines the techniques from mathematics, statistics, algorithms and Artificial Intelligence to extract the knowledge. Data mining is a main phase of Knowledge Discovery in Databases (KDD) for extracting the knowledge based on the patterns and their correlation by the application of appropriate association rules to the information's available from the data set. The outcome of the KDD is used to analyze or predict on the future aspects in any area of considerations. In this paper we propose an analysis and prediction of the market sales based on the historical information's from the database by considering the items information at different levels to generate the association rules. The widely used algorithm in data mining ie, apriori algorithm is specifically considered for the extraction of the knowledge.

Apriori Algorithm made simple for data mining – Market basket analysis:
Apriori algorithm is a classic algorithm used in data mining for learning association rules. It is nowhere as complex as it sounds, on the contrary it is very simple; let us illustrate it. Suppose you have records of large number of transactions at a shopping center as follows:

Transactions	Items bought
T1	Item1, item2, item3
T2	Item1, item2
T3	Item2, item5
T4	Item1, item2, item5

Learning association rules basically means finding the items that are purchased together more frequently than others.
For example in the above table you can see Item1 and item2 are bought together frequently.

What is the use of learning association rules?

- Shopping centers use association rules to place the items next to each other so that users buy more items. If you are familiar with data mining you would know about the famous beer-diapers-Wal-Mart story. Basically Wal-Mart studied their data and found that on Friday afternoon young American males who buy diapers also tend to buy beer. So Wal-Mart placed beer next to diapers and the beer-sales went up. This is famous because no one would have predicted such a result and that's the power of data mining.

- Also if you are familiar with Amazon, they use association mining to recommend you the items based on the current item you are browsing/buying.

- Another application is the Google auto-complete, where after you type in a word it searches frequently associated words that user type after that particular word.

Apriori is the classic and probably the most basic algorithm to do it.

Let's start with a market basket example,

Transaction ID	Items Bought
T1	{Mango, Onion, Apple, Key-chain, Eggs, Banana}
T2	{Doll, Onion, Apple, Key-chain, Eggs, Banana}
T3	{Mango, Apple, Key-chain, Eggs}
T4	{Mango, Umbrella, Corn, Key-chain, Banana}
T5	{Corn, Onion, Onion, Key-chain, Ice-cream, Eggs}

Now, we follow a simple golden rule: we say an item/itemset is frequently bought if it is bought at least 60% of times. So for here it should be bought at least 3 times.

For simplicity
M = Mango
O = Onion
And so on……

So the table becomes

Original table:

Transaction ID	Items Bought
T1	{M, O, A, K, E, B }
T2	{D, O, A, K, E, B }
T3	{M, A, K, E}
T4	{M, U, C, K, B}
T5	{C, O, O, K, I, E}

Step 1: Count the number of transactions in which each item occurs, Note 'O=Onion' is bought 4 times in total, but, it occurs in just 3 transactions.

Item	*No of transactions*
M	3
O	3
A	2
K	5
E	4
B	3
D	1
A	1
U	1
C	2
I	1

Step 2: Now remember we said the item is said frequently bought if it is bought at least 3 times. So in this step we remove all the items that are bought less than 3 times from the above table and we are left with

Item	Number of transactions
M	3
O	3
K	5
E	4
B	3

This is the single items that are bought frequently. Now let's say we want to find a pair of items that are bought frequently. We continue from the above table (Table in step 2)

Step 3: We start making pairs from the first item, like MO,MK,ME,MB and then we start with the second item like OK,OE,OY. We did not do OM because we already did MO when we were making pairs with M and buying a Mango and Onion together is same as buying Onion and Mango together. After making all the pairs we get,

Item pairs
MO
MK
ME
MB
OK
OE
OB
KE
KB
EB

Step 4: Now we count how many times each pair is bought together. For example M and O is just bought together in {M,O,A,K,E,B}
While M and K is bought together 3 times in {M,O,A,K,E,B}, {M,A,K,E} AND {M,U,C, K, B}
After doing that for all the pairs we get

Item Pairs	Number of transactions
MO	1
MK	3
ME	2
MB	2
OK	3
OE	3
OB	2
KE	4
KB	3
EB	2

Step 5: Golden rule to the rescue. Remove all the item pairs with number of transactions less than three and we are left with

Item Pairs	Number of transactions
MK	3
OK	3
OE	3
KE	4
KB	3

These are the pairs of items frequently bought together.
Now let's say we want to find a set of three items that are brought together.
We use the above table (table in step 5) and make a set of 3 items.

Step 6: To make the set of three items we need one more rule (it's termed as self-join),
It simply means, from the Item pairs in the above table, we find two pairs with the same first Alphabet, so we get

- OK and OE, this gives OKE
- KE and KB, this gives KEB

Then we find how many times O,K,E are bought together in the original table and same for K,E,Y and we get the following table

Item Set	Number of transactions
OKE	3
KEB	2

While we are on this, suppose you have sets of 3 items say ABC, ABD, ACD, ACE, BCD and you want to generate item sets of 4 items you look for two sets having the same first two alphabets.

- ABC and ABD -> ABCD
- ACD and ACE -> ACDE

And so on … In general you have to look for sets having just the last alphabet/item different.

Step 7: So we again apply the golden rule, that is, the item set must be bought together at least 3 times which leaves us with just OKE, Since KEY are bought together just two times.

Thus the set of three items that are bought together most frequently are O, K, E Viz., Onion, Key-chain and Eggs in our example.

With the above algorithm predictions can be made simple and easy to generate great profits with historical datasets.

REFERENCES

- S. Chaudhuri and K. Shim, Including Groupby in Query Optimization, *Proc. Int',l Conf. Very Large Database Systems,* 1994.

- A. Gupta, V. Harinarayan and D. Quass, Generalized Projections: A Powerful Approach to Aggregation, *Proc. Int',l Conf. Very Large Database Systems,* 1995.

- W. Yan and P. Larson, Eager Aggregation and Lazy Aggregation, *Proc. 21st Int',l Conf. Very Large Databases (VLDB),* pp. 345-357, 1995.

- P. O',Neil and G. Graefe, Multi-Table Joins through Bitmapped Join Indices, *SIGMOD Record,* vol. 24, no. 3, pp. 8-11, 1995.

- J. Widom, Research Problems in Data Warehousing, *Proc. Fourth Int',l Conf. Information and Knowledge Management,* 1995.

 N. Roussopoulos, The Logical Access Path Schema of a Database, *IEEE Trans. Software Eng.*, vol. 8, pp. 563-573, Nov. 1982.

 T.K. Sellis, Multiple Query Optimization, *ACM Trans. Database Systems*, vol. 13, 1988.

 A. Cosar, E.-P. Lim and J. Srivastava, Multiple Query Optimization with Depth-First Branch-and-Bound and Dynamic Query Ordering, *Proc. Conf. Information and Knowledge Management (CIKM)*, 1993.

 V. Harinarayan, A. Rajaraman and J. Ullman, Implementing Data Cubes Efficiently, *Proc. ACM SIGMOD Int',l Conf. Mangement of Data*, 1996.

 H. Gupta, V. Harinarayan, A. Rajaraman and J. Ullman, Index Selection in OLAP, *Proc. Int',l Conf. Data Eng.*, 1997.

 U. Feige, A Threshold of ln n for Approximating Set Cover, *Proc. 28th Ann. ACM Symp. Theory of Computing (STOC)*, 1996.

 U.S. Chakravarthy and J. Minker, Processing Multiple Queries in Database Systems, *Database Eng.*, vol. 5, pp. 38-44, Sept. 1982.

 N. Nilsson, *Principles of Artificial Intelligence*. Morgan Kaufmann Publishers, Inc., 1980.

 H. Gupta and I. Mumick, Selection of Views to Materialize under a Maintenance Cost Constraint, *Proc. Int',l Conf. Database Theory*, 1999.

 H. Karloff and M. Mihail, On the Complexity of the View-Selection Problem, *Proc. Symp. Principles of Database Systems (PODS)*, 1999.

 R. Chirkova, A. Halevy and D. Suciu, A Formal Perspective on the View Selection Problem, *Proc. Int',l Conf. Very Large Database Systems*, 2001.