УДК [338.24:330.131.7]:004.421.4

USING APRIORI ALGORITHM FOR INVESTIGATION OF ECONOMIC DATA

F. ACHEAMPONG
Master student, Department of Mathematical and Information Support of Economic Systems , Yanka Kupala State University of Grodno

N.V. MARKOVSKAYA, PhD
Associate professor, Department of Mathematical and Information Support of Economic Systems Yanka Kupala State University of Grodno

Yanka Kupala State Univer-sity of Grodno, Republic of Belarus
E-mail: freezle3079@yahoo.com, n.markovskaya@grsu.by

Abstract. Looking at the trend the world's economy is taking and the significant role the financial market is playing, a nation needs technical brains equipped with the necessary skills and knowledge to make good economics decisions, manage risk and most importantly, forecast what is expected in both the short and the long run. It is for this reason that the market needs a data analyst to forecast what is expected in the near future for profit maximization. This article is designed to explain and analyse the use of an algorithm Apriori using the associative rules in the available data sets.

Keywords: apriori algoritm, association rule.

INTRODUCTION: Apriori relates to or denotes reasoning or knowledge which proceeds from theoretical deduction rather than from observation or experience. Apriori Algorithm is one of the most popular algorithms in data mining for learning the concept of association rules. It is being used by so many people specifically for transaction operations and also it can be used in real time applications (for instance, grocery shop, general store, library etc.) by collecting the items bought by customers over the time so that frequent itemsets can be generated. Frequent itemsets (itemsets with frequency greater than or equal to a user specified minimum support) can be found very easily because of its combinatorial explosion. Once they are obtained, it is simply easy to generate association rules with confidence greater than or equal to a user specified minimum confidence. With the invent in technology of information and the need for extracting useful information of business people from dataset, data mining and its techniques is appeared to achieve and improve economic growth.

Definition: Following the original definition by Agrawal the problem of association rule mining is defined as: Let $\mathbf{I = \{i_1, i_2... i_n\}}$ be a set of *n* binary attributes called *items*. Let $\mathbf{D = \{t_1, t_2, \dots, t_n\}}$ be a set of transactions called the *database*. Each transaction in **D** has a unique transaction ID and contains a subset of the items in **I**. A *rule* is defined as an implication of the form **X - >Y** where $\mathbf{X, Y} \subseteq \mathbf{I}$ and $\mathbf{X} \cap Y = \emptyset$**.** The sets of items (for short *itemsets*) **X** and **Y** is called *antecedent* (left-hand-side or LHS) and *consequent* (right-hand-side or RHS) of the rule respectively. To illustrate the concepts, we use a small example from the supermarket domain. The set of items is ***I*= {milk, bread, butter, beer}** and a small database containing the items (1 codes presence and 0 absence of an item in a transaction) is shown in the table below. An example rule for the supermarket could be {milk, bread} => {butter} meaning that if milk and bread is bought, customers also buy butter. Note: this example is extremely small. In practical applications, a rule needs a support of several hundred transactions before it can be considered statistically significant, and datasets often contain thousands or millions of transactions.

Transaction ID	Milk	Bread	Butter	Beer
1	1	1	0	0
2	0	1	1	1
3	0	0	0	1
4	1	1	1	0
5	0	1	0	0
6	1	0	0	0
7	0	1	1	1
8	1	1	1	1
9	0	1	0	1
10	1	1	0	0
11	1	0	0	0
12	0	0	0	1
13	1	1	1	0
14	1	0	1	0
15	1	1	1	1

Useful Concepts.

To select interesting rules from the set of all possible rules, constraints on various measures of significance and interest can be used. The best-known constraints are minimum thresholds on support and confidence.

Support.

The support ***supp*(*X*)** of an itemset *X* is defined as the proportion of transactions in the data set which contain the itemset.

supp(X)= no. of transactions which contain the itemset X / total no. of transactions.

In the example database, the itemset {milk, bread, butter} has a support of 4 /15 = 0.26 since it occurs in 26% of all transactions. To be even more explicit we can point out that 4 is the number of transactions from the database which contain the itemset {milk, bread, butter} while 15 represents the total number of transactions.

Confidence.

The ***confidence*** of a rule is defined:

$$\mathbf{Conf\ (X \rightarrow Y) = Supp\ (X \cup Y) / Supp\ (X)} \tag{1}$$

For the rule {milk, bread} => {butter} we have the following confidence:

Supp ({milk, bread, butter}) / **supp**({milk, bread}) = 0.26 / 0.4 = 0.65.

This means that for 65% of the transactions containing milk and bread the rule is correct. Confidence can be interpreted as an estimate of the probability ***P*(*Y* | *X*)**, the probability of finding the RHS of the rule in transactions under the condition that these transactions also contain the LHS.

Lift

The ***lift*** of a rule is defined as:

$$\mathbf{Lift\ (X \rightarrow Y) = supp\ (X \cup Y) / supp\ (Y) * supp\ (X)} \tag{2}$$

The rule {milk, bread} => {butter} has the following lift:

Supp ({milk, bread, butter}) / supp({butter})*supp({milk, bread})= 0.26/0.46*0.4= 1.4.

Conviction

The ***conviction*** of a rule is defined as:

$$\boldsymbol{Conv(X \to Y) = \frac{1 - supp(Y)}{1 - conf(X \to Y)}} \tag{3}$$

The rule {milk, bread} => {butter} has the following conviction:

$$\frac{\mathbf{1-supp\{butter\}}}{\mathbf{1-conf\{milk,bread\}}} => \{\boldsymbol{butter}\} = \frac{\mathbf{1-0.46}}{\mathbf{1-0.65}} = \mathbf{1.54}.$$

The conviction of the rule **X=>Y** can be interpreted as the ratio of the expected frequency that **X** occurs without **Y** (that is to say, the frequency that the rule makes an incorrect prediction) if **X** and **Y** were independent divided by the observed frequency of incorrect predictions.

In this example, the conviction value of 1.54 shows that the rule {milk, bread}=>{butter} would be incorrect 54% more often (1.54 times as often) if the association between X and Y was purely random chance.

Apriori algorithm

General Process

Association rule generation is usually split up into two separate steps:

1. First, minimum support is applied to find all *frequent itemsets* in a database.
2. Second, these frequent itemsets and the minimum confidence constraint are used to form rules.

While the second step is straight forward, the first step needs more attention.

Finding all frequent itemsets in a database is difficult since it involves searching all possible itemsets (item combinations). The set of possible itemsets is the power set over *I* and has size $2n - 1$ (excluding the empty set which is not a valid itemset). Although the size of the power set grows exponentially in the number of items *n* in *I*, efficient search is possible using the ***downward-closure property*** of support (also called *anti-monotonicity*) which guarantees that for a frequent itemset, all its subsets are also frequent and thus for an infrequent itemset, all its supersets must also be infrequent. Exploiting this property, efficient algorithms (e.g. Apriori and Éclat) can find all frequent itemsets.

Apriori Algorithm Pseudo code

```
Procedure Apriori (T, minSupport) {//T is the database and minSupport is the minimum support
L1= {frequent items};
For (k= 2: $L_{k-1}$! =∅; k++)
 {Ck= candidates generated from Lk-1
//that is Cartesian product Lk-1 x Lk-1 and eliminating any k-1 size itemset that is not //frequent
For each transaction t in database do {#increment the count of all candidates in Ck that are contained in t
Lk = candidates in Ck with minSupport
}//end for each
}//end for return ;}
```

As is common in association rule mining, given a set of itemsets (for instance, sets of retail transactions, each listing individual items purchased), the algorithm attempts to find subsets which are common to at least a minimum number C of the itemsets. Apriori uses a "bottom up" approach, where frequent subsets are extended one item at a time (a step known as candidate generation), and groups of candidates are tested against the data. The algorithm terminates when no further successful extensions are found.

Apriori uses breadth-first search and a tree structure to count candidate item sets efficiently. It generates candidate item sets of length k from item sets of length k − 1. Then it prunes the candidates which have an infrequent sub pattern. According to the downward closure lemma, the candidate set contains all frequent k-length item sets. After that, it scans the transaction database to determine frequent item sets among the candidates.

Apriori, while historically significant, suffers from a number of inefficiencies or trade-offs, which have spawned other algorithms. Candidate generation generates large numbers of subsets (the algorithm attempts to load up the candidate set with as many as possible before each scan). Bottom-up subset exploration

(Essentially a breadth-first traversal of the subset lattice) finds any maximal subset S only after all 2 | S | − 1 of its proper subsets.

Sample usage of Apriori algorithm

A large supermarket tracks sales data by Stock-keeping unit (SKU) for each item, and thus is able to know what items are typically purchased together. Apriori is a moderately efficient way to build a list of frequent purchased item pairs from this data. Let the database of transactions consist of the sets {A,B,E}, {B,D}, {B,C}, {A,B,D}, {A,C}, {B,C}, {A,C}, {A,B,C,E}, {A,B,C}. Each number corresponds to a product such as "butter" or "water". The first step of Apriori is to count up the frequencies, called the supports, of each member item separately:

Transaction ID	**List of Items**
1	A,B,E
2	B,D
3	B,C
4	A,B,D
5	A,C
6	B,C
7	A,C
8	A,B,C,E
9	A,B,C

Let Minimum Support=2. For Candidate 1:

Itemsets	**Support Count**
A	6
B	7
C	6
D	2
E	2

We can define a minimum support level to qualify as "frequent," which depends on the context. For this case, let min support = 2. Therefore, all are frequent. The next step is to generate a list of all 2-pairsof the frequent items. Had any of the above items not been frequent, they wouldn't have been included as a possible member of possible 2-item pairs. In this way, Apriori *prunes* the tree of all possible sets. In next step we again select only these items (now 2-pairs are items) which are frequent (the pairs which are **bold**):

Itemset	**Support Count**
AB	4
AC	4
AD	1
AE	2
BC	4
BD	2
BE	2
CD	0
CE	1
DE	0

We generate the list of all 3-triples of the frequent items (by connecting frequent pair with frequent single item).
Step 2.

Item	**Support**
A,B,C	2
A,B,E	2
A,B,D	1
A,C,E	1
A,C,D	0
B,C,D	0
B,C,E	1
B,D,E	0

Again the item set which does not meet the required minimum support {Items **not** bold} is pruned. The next step is to find the 4-pairs item set {A.B.C.E}. The algorithm will end here because the pair {A, B, C, E} =1, generated at the next step does not have the desired support. The algorithm terminates when no further successful extensions are found.

CONCLUSION: We have come to the realization that due to the ever increasingly chunk of economic data and technology, every aspect of the economic sector needs an algorithm Apriori used by data analyst to scan database transactions to critically assist and deal with such variable data sets.

References

[1]. Lin M., Lee P. & Hsueh S. (2012). Apriori-based Frequent Itemset Mining Algorithms on MapReduce. Proc. of the 16th International Conference on Ubiquitous Information Management and Communication (ICUIMC ’12). New York, NY, USA, ACM: Article No.76.

[2]. “Data Mining - concepts and techniques” by Jiawei Han and Micheline Kamber

[3]. Paul S. & Saravanan V. (2008). Hash Partitioned Apriori in Parallel and Distributed Data Mining Environment with Dynamic Data Allocation Approach. Proc. of the International Conference on Computer Science and Information Technology (ICCSIT ’08). Singapore, IEEE: 481 – 485.

[4]. D. Burdick, M. Calimlim, and J. Gehrke, “Mafia: A maximal frequent itemset algorithm for transactional databases,” in Proceedings of the 17th International Conference on Data Engineering, 2001. IEEE, 2001, pp.443–452

[5]. R.Agrawal, T. Imieli´nski, and A. Swami, “Mining association rules between sets of items in large databases,” in ACM SIGMOD Record, vol. 22, no. 2. ACM, 1993, pp. 207–216.

[6]. R. Agrawal, R. Srikant et al., “Fast algorithms for mining association rules,” in Proceedings of the 20th international conference of very large data bases, VLDB, vol. 1215, 1994, pp. 487–499.

[7]. M. J. Zaki, S. Parthasarathy, M. Ogihara, W. Li et al., “New algorithms for fast discovery of association rules,” in KDD, vol. 97, 1997, pp. 283–286.