Scholars
www.setscholars.org
Knowledge is Power

May – 2012
Volume – 1, Issue – 2
Article #03

IJAR-CSIT
Research Paper
ISSN: 1839-8480

Link Analysis using Data Mining System

Mohammad Mobin Akhtar[1], Abu Sarwar Zamani[2], Ayman EL-SAYED[3]

[1]Computer Science & Information System Dept., Community College in quwayiyah, Shaqra University, KSA.

[2]Dept. of Computer Science College of Science & Humanity, Shaqra University, KSA.

[3]Computer Science & Eng. Dept., Faculty of Electronic Eng., Menoufiya University, 32952 Menouf, Egypt.

*Corresponding author's e-mail: jmi.mobin@gmail.com, sarwar_zamani@yahoo.com, ayman.elsayed@el-eng.menofia.edu.eg

Abstract

Center of attention of this paper on link analysis used by Data Mining systems to extract associations between individual data records or data sets involved in the same event. It demonstrates an implementation of the algorithm with custom modifications made to expand functionality and improve time and space complexity. The system makes use of the frequent itemsets to generate association rules, while also calculating support and confidence. The algorithms are integrated in a user-friendly system which can be used to generate frequent itemsets and extract association rules online in real time. Business intelligence mainly refers to computer-based techniques used in identifying, extracting, Gathering, storing, analyzing, and providing access to data to help enterprise users make better business decisions analyzing business data. Business Intelligence systems may be divided into reporting systems and data mining applications. Data mining is Knowledge Discovery in Data and the science of extracting useful knowledge from huge data repositories. Data mining applications often employ sophisticated mathematical and statistical techniques to perform data analysis, search for specific patterns or relationships, if they exist, and make future predictions.

Keywords: Business Intelligence, Data Mining, Database Query, Algorithm.

Citation: Akhtar MM *et al*. (2012), *Link Analysis using Data Mining System*. IJAR-CSIT 1(2): p. 38 – 49.

Received: 04-04-2012 **Accepted:** 09-05-2012

1. Introduction

Business Intelligence: Business Intelligence is Computer-based techniques used in spotting, digging-out, and analyzing 'hard' business data, such as sales revenue by products or departments or associated costs and incomes. Objectives of a Business Intelligence exercise include: (1) Understanding of a firm's internal and external strengths and weaknesses. (2) Understanding of the relationship between different data for better decision making. (3) Detection of opportunities for innovation.(4) Cost reduction and optimal deployment of resources. See also competitive intelligence [1]. Business data can be any type of data related to a business or organization but is usually sales data, transaction history, customer records, employee records or any kind of stored data which can be analyzed (usually statistically or mathematically) [2].

A. Responsibility of Business Intelligence in Organization.

Business Intelligence is particularly essential to organizations and their employees since they need access to timely information and analysis of that information. In order to pursue more strategic BI, organizations must babble to effectively manage all the data at their disposal [3].

The information that is retrieved must be of very high quality standards. If not, then this effectively means that organizations will be limited to a less strategic BI approach. To be successful in its data analysis a business must possess fast, relatively cheap data warehouses required to effectively support BI. A data warehouse refers to a database system that includes data, programs, and the necessary personnel who specialize in the preparation of data for BI processing.Fig.1 illustrates the components of a data warehouse [3]. Data are read from operational databases by the Extract, Transform and Load (ETL) System. This system will clean and prepare the data in order to be processed for BI. The extracted data will be stored in a data warehouse using a data warehouse Database Management System (DBMS). Additionally, metadata concerning the data's source, format assumptions, etc., is maintained. The data warehouse DBMS will then extract data and forward them to BI Tools with which BI users interact.

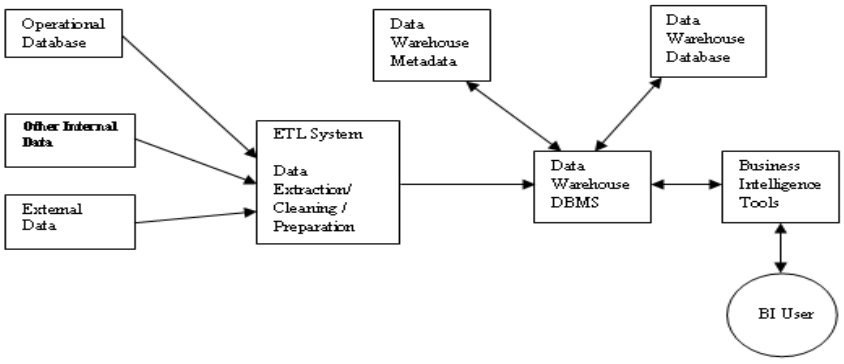

Fig.1: Data Warehouse Components.

BI systems are also referred to as Decision Support Systems since they assist organizations with their decision making. They are used in analyzing present and past activities of the organization but also in accurately predicting future events. BI systems do not support operational activities such as processing or recording of orders. Fig.2 shows the role of BI systems in decision making. BI systems provide the tools for the transformation of data into information and knowledge which in turn assists the organization in its decision making. If the decisions undertaken by the organization are successful, it is expected that the organization will witness an important improvement in its competitiveness. BI systems should make it possible for the users to set precise objectives and to execute these objectives. Furthermore, they provide a basis for decision making and allow the optimization of future actions by acting upon various aspects of the company's performance. Ultimately, they help enterprises to realize their strategic objectives more effectively. To summarize, BI technologies provide historical, current and predictive views of business operations. Common functions of business intelligence technologies are reporting, online analytical processing, analytics, data mining, process mining, complex event processing, business performance management, benchmarking, text mining, and predictive analytics.

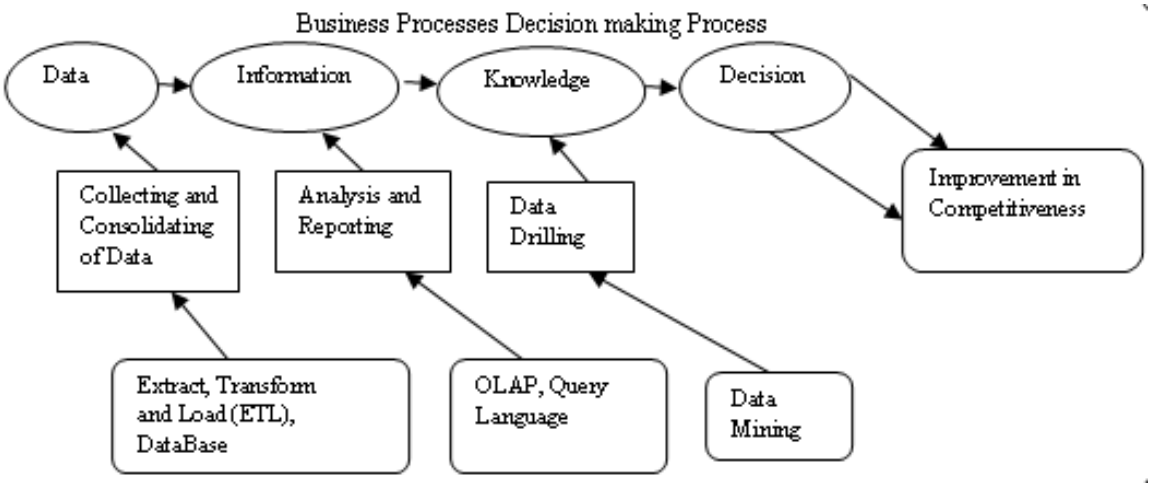

Fig. 2: The role of Business Intelligence Systems in decision making.

B. Data Mining

Data mining also known as Knowledge Discovery in Data is the science of extracting useful knowledge from huge data repositories. Data mining applications use mathematical and statistical techniques to perform what-if analysis, discover new patterns, make future predictions, and therefore assist in decision making [4].

Data Mining Operations and Techniques: Table I presents in briefly the description of the four main operations supported by Data Mining applications and the techniques associated with them.

This paper focuses on link analysis used by Data Mining systems to extract associations between individual data records or data sets involved in the same event. The system makes use of the frequent itemsets to generate association rules, while also calculating support and confidence.

The algorithms are integrated in a user-friendly system which can be used to generate frequent itemsets and extract association rules online in real time. The implementation aspect of this project aims to solve the problem of a computerized, fully working solution for a data mining system that produces association rules. The system allows a user to select attributes of a database along with the frequency, support and confidence of a rule. It then scans the database and generates frequent itemsets for the user's selections. Finally, it produces association rules for the aforementioned selections.

TABLE I: DATA MINING OPERATIONS AND TECHNIQUES

Operations	Data Mining Techniques
Predictive modeling: Analyzes data to determine essential characteristics (model).	Classification: Establishes a specific predetermined class for each data record. Value prediction: Estimates a continuous numeric value associated with each data record.
Database segmentation: Partitions a database into a number of homogeneous segments.	Demographic or Neural Clustering: Establish allowable data inputs to be used for analysis by calculating the distance between records.
Link analysis: Establishes links, called associations, between records or Sets of them in a database.	Association discovery: Finds items that imply the presence of other items in the same event. Sequential pattern discovery: Finds patterns between events. Similar time sequence discovery: Finds links between two sets of data that are time dependent.
Deviation detection: Identifies outliers.	Statistics: Identify outliers using for example linear regression. Visualization: Displays summaries and graphs to make deviations easy to detect.

The paper is organized as follow: section II describes association rules and algorithms, the case study of medical database is depicts in section III. The Modified association algorithms that are proposed, explained in section IV. The implementation and some sample results are shown in section V. Finally, the paper is concluded in section VI.

2. Association rules and algorithms

An association rule is a rule which implies certain association relationships among a set of objects, such as "occur together" or "one implies the other" [5,6,7]. Let $I = \{i_1, i_2, \ldots, i_m\}$ be a set of literals called items. The database consists of a set of sales transactions [2]. Each transaction T is a set of items such that $T \subseteq I$. A transaction T is said to contain the set of items X if and only if $X \subseteq T$. An association rule is an expression of the form $X \rightarrow Y$, where $X \subseteq I$ and $Y \subseteq I$ [8]. The intuitive meaning of such a rule is that the presence of the set of items X in a transaction set also indicates a possibility of the presence of the itemset Y. In plain English this means that the transactions of the database which contain X tend to contain Y as well. Two classical notions for establishing the strength of a rule are those of minimum support and minimum confidence [9]. The support of a rule $X \rightarrow Y$ is the fraction of transactions which contain both X and Y. The confidence of a rule $X \rightarrow Y$ is the fraction of transactions containing X which also contain Y. Thus, if we say that this rule has 90% confidence it means that 90% of the

tuples containing X also contain Y. The Apriori Algorithms an influential algorithm for mining frequent itemsets for Boolean association rules and the Key Concepts are defined as follow:

- Frequent Itemsets: The sets of item which has minimum support (denoted by Lifor i^{th}-Itemset).
- Apriori Property: Any subset of frequent itemset must be frequent.
- Join Operation: To find $L_{k,}$ a set of candidate k-itemsets is generated by joining L_{K-1} with itself.

Find all combinations of items that have transaction support above minimum support. Call these combinations frequent itemsets. Use the frequent itemsets to generate the desired rules. The general idea is that if, say, ABCD and AB are frequent itemsets, then we can determine if the rule ABCD holds by computing the ratio r = support(ABCD)/support(AB). The rule holds only if r >= minimum confidence. Note that the rule will have minimum support because ABCD is frequent. The Apriori algorithm in pseudo code appears in Fig. 3.

```
L1 = {large 1-itemsets};
     for ( k = 2; LK-1 ≠∅; k ++) do begin
CK = apriori-gen(LK-1); // New candidates
     forall transactions t ∈ D do begin
Ct = subset(Ck, t); // Candidates contained in t
     forall candidates c ∈ Ct do
   c.count++;
end
   LK = {c ∈ CK | c.count ≥ minsup};
end
  Answer = ∪KLk;
```

Fig. 3: The Apriori algorithm in pseudo code [10].

Apriori makes multiple passes over the database. In the first pass, the algorithm simply counts item occurrences to determine the frequent 1-itemsets (itemsets with 1 item). A subsequent pass, say pass k, consists of two phases. First, the frequent itemsets Lk-1 (the set of all frequent (k-1) itemsets) found in the $(k-1)^{th}$ pass are used to generate the candidate itemsets CK (that is candidates to become frequent itemsets) using the apriori-gen( ) function.

This function first joins L_{k-1} with L_{k-1}, the joining condition being that the lexicographically ordered first k-2 items are the same. Next, it deletes all those itemsets from the joined result that have some (k-1)-subset that is not in L_{k-1} yielding C_k. The algorithm then scans the database. For each transaction, it determines which of the candidates in C_k are contained in the transaction using a hash-tree data structure and increments the count of those candidates. At the end of the pass, C_k is examined to determine which candidates are frequent, yielding L_k. The algorithm terminates when L_k becomes empty.

3. Case Study: Medical Database

The case uses a real medical database with historical patient data which includes blood analysis results, blood pressure measurements, clinical history data, and other. The patients described are people who have sustained or are in danger of sustaining a stroke. The data was prepared for use by completing the following activities:

- The data was cleaned by removing missing or incomplete values.
- Numerical values were changed to characters.
- The layout of the database was changed from a horizontal to a vertical layout.In the horizontal layout, all the attributes are in columns whereas, in the vertical layout all the attributes are seen as transactions, so there are only two columns; the first column is the Patient Id and the second is a valued attribute of the patient e.g. Cholesterol.
- The original database was in the form of a spreadsheet file which was then converted to a database.

TABLE II: DATA DICTIONARY OF THE MEDICAL DATABASE

Attribute	Attribute Description	Unit of measurement
ID	The unique identifying number of each subject	
IEVENT	Defines if subject suffered one or more strokes	1- suffered 2- didn't suffer
STENOSIS	Carotid stenos is Level	• small • medium • large • extra large • dangerous
AGE	The age of the Subject	• middle aged • middle senior • senior 1 • senior 2
SEX	The sex of the Subject	male / female
SMOKING	Defines if subject smokes or not	1 – smoker 0 – non-smoker
ARRHYTHM	Defines if subject suffers from Cardiac arrhythmia	1 – suffers 0 – does not suffer
HYPERTEN	Defines if subject suffers from hypertension	1 – suffers 0 – does not suffer
DIABETES	Defines if subject suffers from diabetes	1 – suffers 0 – does not suffer
CHOLESTE	Blood concentrations of total cholesterol	• normal • acceptable • high
TRIGLYCE	Blood concentrations of triglycerides	• low • normal • high
DEATH	Defines if subject Died	1 – died 0 – lives

A data dictionary presenting the contents of the database is shown in Table II.

4. Modified Association Algorithms

For the implementation described in this paper the Apriori algorithm was selected mainly because this can be implemented in pure SQL and uses only simple database primitives, namely sorting and merge scan join.

A. 1st modification: Generation of frequent itemsets with specific items

This will force the algorithm to run on a greatly reduced data set while at the same time reducing the chance of overfilling the RAM of the PC, and thus using the hard disk for saving the intermediate results. Subsequently, the algorithm will now execute in only a fraction of the time and will use minimal memory resources. The final results will still be correct as items will be included in the association rules. Fig. 5 shows the first modification made to the algorithm [11].

```
CREATE TABLE Only_Interesting_Items ( Id VARCHAR(30), Item VARCHAR(30));
      INSERT INTO Only_Interesting_Items
(SELECT * FROM database_table
     WHERE // enter specific items as SQL variables
Item=@Item1 OR
Item=@Item2 OR
       OR……
Item=@Itemk ); //First pass
    INSERT INTO C1
(SELECT Item AS Item1, COUNT(*) AS cnt
         FROM Only_Interesting_Items
GROUP BY Item HAVING COUNT(*) >= Frequency);
    INSERT INTO R1
(SELECT p.Id AS Id, p.Item AS Item1
        FROM Only_Interesting_Items p, C1 c
WHERE p.item = c.item1);
```

Fig. 5: First modification; only relevant itemsets are generated.

B. 2nd modification: Generation of association rule& calculation of support/confidence

The algorithm can be further expanded so that after it has generated all the related frequent itemsets, it can use them to calculate an association rule for them. This can be done as following:

```
SET @Support = (SELECT cnt FROM Ck WHERE
         (Item1 = @Dimension1 OR Item1 = @Dimension2
OR
     Item1 = @Dimensionk OR Item1=@Measure)
AND
    (Item2=@Dimension1 OR Item2 = @Dimension2 ...
OR
    item2 = Dimensionk OR Item2=@Measure)
AND
(
   Itemk=@Dimension1 OR Itemk = Dimension2
OR
   Itemk = Dimensionk OR Itemk=@Measure) / Total_Items_In_Database;
// Confidence
SET @Confidence = @Support / ( (Select cnt from Ck-1  WHERE
```

```
     (Item1 = @Dimension1 OR Item1 = @Dimension2 …
OR
      Item1 = @Dimensionk) AND
     (Item2 = @Dimension1 OR Item2 = @Dimension2
OR Item2 = @Dimensionk) AND
…
(
      Itemk-1 = @Dimension1 OR Itemk-1 = @Dimension2 …
       OR Itemk-1 = @Dimensionk)
   / Total_Items_In_Database
```

Fig. 6: Second modification; Calculating Support and Confidence.

At first, the structure of the association rule needs to be defined. Then, the support and the confidence of the rule need to be calculated. Remember that for the rule A→B we have:

$$\text{Support } (A \rightarrow B) = \text{Probability } (A \cup B) = \text{Count } (A \cup B) / \text{Count (Total)}$$
$$\text{Confidence } (A \rightarrow B) = \text{Probability } (B|A) = \text{Count } (A \cup B) / \text{Count } (A)$$

The support and confidence for an association rule for any frequent k-itemset can therefore be calculated in SQL as shown in Fig. 6. The algorithm was expanded to store the results, along with the support and confidence, in a results table after checking if the produced rules are strong, i.e., it satisfies minimum support and confidence (see Fig. 7) [12].

```
CREATE TABLE ANSWER (Message VARCHAR (200));
IF @Support >= @Min_support/100.0 AND  @Confidence >= @Min_confidence/100.0
 BEGIN
   INSERT INTO ANSWER (SELECT 'The selected items generate this rule:')
   INSERT INTO ANSWER (SELECT @Dimension1+' AND '+@Dimensionk+ ' => ' +@Measure)
   INSERT INTO ANSWER
   (SELECT 'With Support= '+(CONVERT (VARCHAR(5), @Support)) +' and Confidence=
   '+(CONVERT (VARCHAR(5), @Confidence))
 END
ELSE
 BEGIN
     INSERT INTO ANSWER
     (SELECT 'Items selected do not satisfy min support or
     confidence so no rules are generated')
     SELECT * FROM ANSWER
     DROP TABLE ANSWER
 END
GO
```

Fig. 7: Second modification; Generation of rules.

C. 3rd modification: Generation of multiple association rules

The algorithm was expanded so that it generates multiple association rules for each pass. This was made possible by generating the confidence and support for each individual member in each itemset with its generation [13].

```
// Execute at the end of each pass of the algorithm
SELECT b.Item_x+' => '+b.Item_y+' With Support= '+ CONVERT(VARCHAR(500),
CAST ( (b.cnt*100.0) /@Subjects_No AS NUMERIC(5,2)) +'% AND Confidence= ' ++
CONVERT(VARCHAR(500)),
CAST ( (b.cnt*100.0) / a.cnt AS NUMERIC(5,2)) +'%'
        FROM table_x-1 a , table_x b
        WHERE a.Item_x=b.Item_x
        GROUP BY a.cnt, b.cnt, b.Item_x,b.Item_y
HAVING b.cnt >= 0
       AND b.cnt/(@Subjects_No/100.0) >= @Min_support
       AND (b.cnt*1.0) / (a.cnt/100.0) >= @Min_confidence;
```

Fig.8: Third modification: Support and confidence for multiple rules.

D. 4th modification: Making the order of the items in itemsets important

Originally, the algorithm was designed to generate all itemsets in a lexicographical manner. This means that it creates unique itemsets by taking unique combinations of items. It also means that the order of the items in an itemset is not considered important, i.e., the itemset {Stroke, Male} would be considered the same as the itemset {Male, Stroke}. In this case, only one of the aforementioned itemsets would be generated. Assuming though that both itemsets were available, the resulting association rules from these two itemsets would be:

Male=>Stroke Support: x%, Confidence: y%

Stroke=>Male Support: x%, Confidence: y%

These rules have totally different meanings: The first states the probability to have a stroke if you are male while the second shows from the total people who had a stroke, what percentage is male. It is therefore important to obtain all the rules. This can be easily done by a simple inequality test for both items at the time of their generation (see Fig. 9.) [14].

```
INSERT INTO RTMPk

SELECT p.Id AS Id, p.Item1 AS Item1, p.Item2 AS
        Item2, …, p.Itemk-1 AS Itemk-1, q.Itemk-1 AS Itemk
        FROM Rk-1 p, Rk-1 q,
         WHERE p.Id = q.Id    AND p.Item1 = q.Item1 ...
AND p.Itemk-2 = q.Itemk-2
// The following line has to be replaced by the line below
AND p.Itemk-1 < q.Itemk-1;
AND p.Itemk-1 <> q.Itemk-1;
```

Fig. 9: Fourth modification: Making the order of the items in an itemset important.

5. Implementation and Sample Results

The modified algorithm was implemented and a user-friendly environment was created whereby the user begins by selecting a number of dimensions to run the association algorithm, by ticking checkboxes and selecting from the drop down menus (see Fig. 10). After the dimension selection the user can enter the desired frequency, support and confidence of the generated association rules. Assuming that all the entries are properly selected the user gets the results as in Fig. 11.

The algorithm has been tested on the database for a number of x-itemsets (where x represents an integer from 1 to 11 itemsets). For more clarity, a 3-itemset for example, will consist of sets of items with exactly 3 members in each set. After all the itemsets are found, a mathematical function is applied to find frequent itemsets, i.e., only those itemsets that satisfy a minimum support and confidence. An association rule is extracted by taking the generated frequent itemsets and the user-supplied support and confidence into account. The rule will depend upon the user attribute selection(s). For example, the user could select the attributes "cholesterol" and "hypertension" as dimensions and "stroke" as measure, to find an association rule (if one exists) for the relationship between cholesterol and hypertension as the cause of stroke.

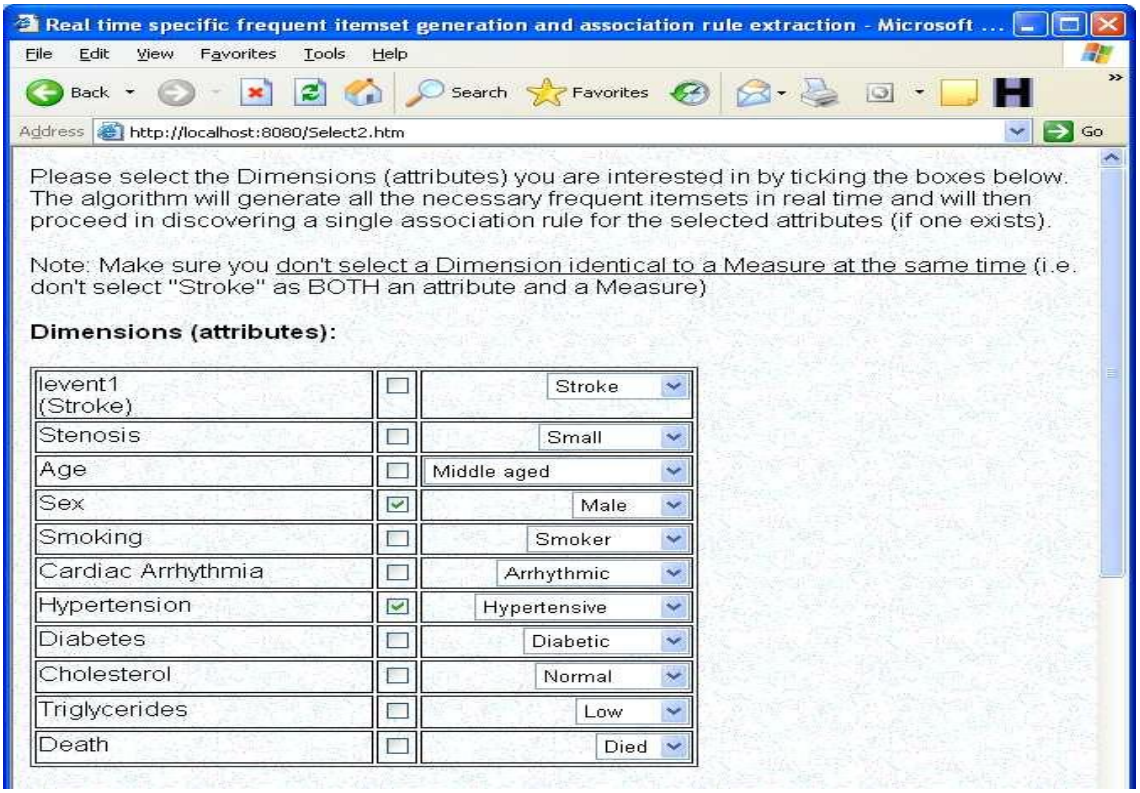

Fig.10: The rule generation page.

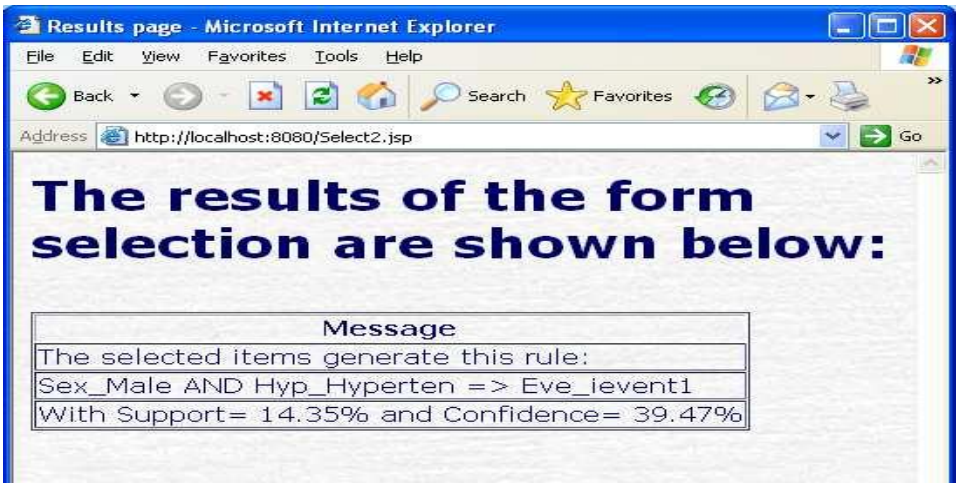

Fig.11: Association rule for dimension selection.

6. Conclusions

This paper presented an implementation of the Apriori algorithm in a medical data mining application. The algorithm has undergone some customization to expand its functionality and improve time and space complexity. It was then tested on the database for a number of x-itemsets. Association rules were extracted successfully, support and confidence were calculated. The implementation also demonstrated how tight coupling between a DBMS and a Data Mining system can be fully achieved.

REFERENCES

[1] C. M. Olszak, E. Ziemba, "Approach to building and implementing Business Intelligence Systems", Interdisciplinary Journal of Information, Knowledge, and Management, Vol. 2, pp. 135-148, 2007.

[2] C. M. Olszak, E. Ziemba, "Approach to building and implementing Business Intelligence Systems", Interdisciplinary Journal of Information, Knowledge, and Management, Vol. 2, pp. 135-148, 2007.

[3] M. Smith., "BI is serious business", http://businessintelligence.com/research/302, 2011.

[4] T. M. Connonly, C. E. Begg, Database Systems: a Practical Approach to Design, Implementation, and Management, 5thEd, Addison-Wesley, 2010.

[5] S. Brin, R. Motwani, C. Silverstein, "Beyond market baskets: generalizing association rules to correlations", Proc. of ACM SIGMOD Conf., pp. 265-276, 1997.

[6] A. Savasere, E. Omiecinski, S. Navathe, "Mining for strong negative associations in a large database of customer transactions", Proc. of ICDE, pp. 494-502, 1998.

[7] L. Rossetti, "What is business intelligence (BI)?," http://searchdatamanagement.techtarget.com/definition/business-intelligence,2006.

[8] Karuna Pande Joshi, "Analysis of Data Mining algorithms", TechReport, UMBC, March,1997.

[9] L. Wei,A. Mozes, “Computing frequent itemsets inside Oracle 10G”, VLDB, Toronto, Canada, August 2004, pp. 1253-1256.

[10] M. A. W. Houtsma, A.N. Swami, “Set-oriented mining for association rules in relational databases”, ICDE 95 Proc. of the 11th International Conference on Data Engineering, IEEE Computer Society, pp. 25-33, 1995.

[11] J. Hipp, U. Guntzer, G. Nakhaeizadeh, “Algorithms for association rule mining – a general survey and comparison”, SIGKDD Explorations, Vol. 2(1), pp. 58- 64, 2000.

[12] R. Agrawal, T.Imielinski, A.Swami, “Mining association rules between sets of items in large databases”, Proc. ACM SIGMOD Conf. Management of Data, pp. 207-216, May1993.

[13] R. Rantzau, “Algorithms and applications for universal quantification in relational databases”, Information Systems,Vol. 28(1-2), March 2003.

[14] J. Han, J. Pei, and Y. Yin, “Mining frequent patterns without candidate generation”, Proc. of the 2000 ACM SIGMOD Intern. Conf. on Management of Data, USA, 2000.