Exception Rules Mining Based on Negative Association Rules

Olena Daly and David Taniar

School of Business Systems, Monash University, Clayton, Victoria 3800, Australia
{Olena.Daly, David.Taniar}@infotech.monash.edu.au

Abstract. Exception rules have been previously defined as rules with low interest and high confidence. In this paper a new approach to mine exception rules will be proposed and evaluated. Interconnection between exception and negative association rules will be considered. Based on the knowledge about negative association rules in the database, the candidate exception rules will be generated. A novel *exceptionality* measure will be proposed to evaluate the candidate exception rules. The candidate exceptions with high exceptionality will form the final set of exception rules. Algorithms for mining exception rules will be developed and evaluated.

1 Introduction

Data Mining is a process of discovering new, unexpected, valuable patterns from existing databases [2, 5]. Though data mining is the evolution of a field with a long history, the term itself was only introduced relatively recently, in the 1990s. Data mining is best described as the union of historical and recent developments in statistics, artificial intelligence, and machine learning. These techniques are then used together to study data and find previously hidden trends or patterns within.

Data mining is finding increasing acceptance in science and business areas that need to analyze large amounts of data to discover trends, in which they could not otherwise find. Different applications may require different data mining techniques. The kinds of knowledge that could be discovered from a database are categorized into association rules mining, sequential patterns mining, classification and clustering [2].

Association rule is an implication of the form X=>Y, where X and Y are database itemsets. The example could be supermarket items purchased together frequently. Two measures have been developed to evaluate association rules, which are *support* and *confidence*. Association rules with high support and confidence are referred to as *strong rules* [1, 2, 3, 4].

Negative association rule is an implication of the form X=>~Y, ~X=>Y, ~X=>~Y, where X and Y are database itemsets, ~X, ~Y are negations of database items. Negative association rules consider both presence and absence of items in the database record and mine for negative implications between database items.

A. Laganà et al. (Eds.): ICCSA 2004, LNCS 3046, pp. 543–552, 2004.

In association rules mining only the rules with high support and high confidence are considered as interesting rules. The generated patterns represent the common trends in the databases and are valuable for the marketing campaigns.

The rules with low support are just as valuable as they may contain unusual, unexpected knowledge about databases. *Exception rules* have been defined as rules with low support and high confidence [6]. A traditional example of exception rules is the rule Champagne => Caviar. The rule may not have high support but it has high confidence. The items are expensive so they are not frequent in the database, but they are always brought together so the rule has high confidence. Exception rules provide valuable knowledge about database patterns.

In this paper a new approach to mine exception rules will be proposed and evaluated. An interconnection between exception and association rules will be considered. Based on the knowledge about negative association rules in the database, the candidate exception rules will be generated. A novel *exceptionality* measure will be proposed to evaluate the candidate exception rules. The candidate exceptions with high exceptionality will be listed in the search algorithm output as exception rules.

The proposed method for mining exceptions is highly valuable in fraud detection systems, security surveillance systems, fire prevention systems etc.

Example. From a healthcare database a strong rule has been discovered that patients that have been bulk-billed can never claim the bill from the health insurance company. If some day the patients do claim the bill that has been bulk-billed (double billing), that is exception (fraud).

Bulk-billed => No Claim – Strong rule
Bulk-billed => Claim – Exception (Fraud)

2 Preliminary

In this section problem statement and related work will be discussed. Essential definitions and examples will be given in the problem statement and current research in the exception rules area will be highlighted in the related work.

2.1 Problem Statement

The search for exception rules will be based on the knowledge about strong association rules in the database. An example: we discover a strong association rule in the database, for instance shares of companies X and Y most times go up together X=>Y. Then those cases when shares of the companies X and Y do not go up together, X=>~Y or ~X=>Y, we call *exceptions* when satisfying the *exceptionality* measure explained in the next section. An algorithm for mining exception rules based on the knowledge about association rules will be proposed in the following sections.

We explain a few terms that will be used along the paper. *Itemset* is a set of database items. *Association rule* is an implication of the form X=>Y, where X and Y are database itemsets. The rule X => Y has *support s*, if *s%* of all transactions contain

both X and Y. The rule X => Y has *confidence c,* if *c%* of transactions that contain X, also contain Y.

In association rules mining user-specified minimum confidence (minconf), minimum support(minsup) are given. Association rules with support>=minsup and confidence>=minconf are referred to as *strong rules.* Itemsets that have support at least equal to minsup are called *frequent itemsets*. *Negative* itemsets are itemsets that contain both items and their negations (for example XY~Z). ~Z means negation of the item Z (absence of the item Z in the database record).

Negative association rule is an implication of the form X=>~Y, ~X=>Y, ~X=>~Y, where X and Y are database items, ~X, ~Y are negations of database items. Examples of negative association rules could be Meat=>~Fish, which implies that when customers purchase meat at the supermarket they do not buy fish at the same time, or ~Sunny=>Windy, which means no sunshine implies wind, or ~OilStock=>~PetrolStock, which says if the price the oil shares is falling, petrol shares price will be falling too.

2.2 Related Work

There have been a few research papers considering exception rules in databases [6, 7]. Existing research has presented different methods for mining interesting exception rules. In [6] the deviation analysis has been employed to distinct interesting exceptions. In [7] the information theory measures have been adopted and the interest of the exception rules has been evaluated basing on the according common sense rule. We would like to search for exception rules based on the knowledge of the negative rules. This is a conceptually new approach.

3 Proposed Exceptionality Measure and Algorithm for Mining Exception Rules in Association Mining

In this section our exceptionality measure will be explained and an algorithm will be proposed to mine exception rules based on knowledge about association rules in the database.

3.1 Exceptionality Measure

We give a few proposed definitions first. For exception rules mining instead of minsup we employ *lower* and *upper bounds,* satisfying the conditions:
0<lower bound<upper bound<minsup;
Definition 1. ***Low*** support belongs to the range [lower bound; upper bound].
Definition 2. *Infrequent itemsets* have low support.

Note that the lower bound is always greater than 0, as we are not interested in rules with 0 support or close to 0. Upper bound is lower than minsup. The lower and upper bounds are chosen specifically for each data mining application.

Definition 3. *Exception Rules* are rules with *low* support and high *exceptionality* values.

In case of exception rules in association mining the confidence measure is not applicable to evaluate the exception rules. For example, we obtain a strong rule A=>B and would like to evaluate a potential exception rule A=>Not B. The strong rule A=>B has high confidence, implying that A=>Not B cannot have high confidence. Let us say the minimum confidence is 60%. The strong rule A=>B satisfies the minimum confidence constraint, so at least 60% of database records containing A also contain B. It means that maximum 40% records containing A do not contain B. The exception rule A=>Not B has maximum 40% confidence. As confidence is not applicable for evaluating exception rules, we propose a special measure *exceptionality* to evaluate the exceptions.

Definition 4. *Exceptionality* of a candidate exception rule given the corresponding association rule is defined by the following formula:

Exceptionality(CandExc/AssocRule)=FuzzySup(CandExc)+ +FuzzyFraction(CandExc/AssocRule)+Neglect(CandExc/AssocRule)

Definition 5. Infrequent itemsets with high exceptionality are called *exceptional* itemsets.

We now explain each of the components of the exceptionality measure.

1. *FuzzySup(CandExc)*

$$\text{FuzzySup(CandExc)} = \text{FuzzySup(support(CandExc))}$$

The support of the candidate exception has to be low (see definition 1). The support of the candidate exception is not allowed to be 0 or close to 0. We define the lower and upper bound of acceptable support values. The acceptable support range we divide into regions with corresponding fuzzy support values formed by a domain expert.

For example, low support belongs to the range is 1%-5%. Then
FuzzySup(0%-0.99%)=0; FuzzySup(1%-1.99%)=1; FuzzySup(2%-2.99%)=5; FuzzySup(3%-3.99%)=2; FuzzySup(4%-5%)=1;
It is clear from the above example that the candidate rules with support in the range [2%-3%] have a better chance to gain a high exceptionality value.

2. *FuzzyFraction(CandExc/AssosRule)*

$$\text{FuzzyFraction(CandExc/AssosRule)} = \text{FuzzyFraction}\left(\frac{\text{support(CandExc)}}{\text{support(AssosRule)}} * 100\%\right)$$

The ratio $\frac{\text{support(CandExc)}}{\text{support(AssosRule)}}$ has to be relatively low. The exception rule may only be a small fraction of the corresponding association rule. Similarly to the FuzzySup(CandExc) above, the acceptable support of FuzzyFraction (CandExc/AssosRule) is divided into fuzzy regions. For example,
FuzzyFraction(0%-9.99%)=6; FuzzyFraction(10%-19.99%)=5; FuzzyFraction(20%-39.99%)=4; FuzzyFraction(40%-59.99%)=2; FuzzyFraction(60%-100%)=0

3. *Neglect(CandExc/AssosRule)*

Neglect(CandExc/AssosRule) is defined by a formula:

$$\frac{\sup(\text{only CandExc})}{\sup(\text{CandExc})} + \frac{\sup(\text{onlyAssosRule})}{\sup(\text{AssosRule})}$$

Support of *only* candidate exception is the fraction of database transactions that contain only items of candidate exception and no other items. Support of *only* association rule is defined similarly. Neglect defines what fraction of candidate exceptions and corresponding association rules occur in the database transaction on their own, while the rest of database items are absent in the transaction. The measure describes the bond between the elements of candidate exception/association rule when no other database items are present. The higher the neglect measure the stronger the bond between the items. In this case no other items could influence occurrence of the exception rule items with each other.

3.2 Classification of Exception Rules

We discuss our exception rules classification and explain the premises of mining exceptions based on the negative association rules in data bases. We suggest two general types of exception rules, which are exceptions in positive sense and exceptions in negative sense.

3.2.1 Exceptions in Negative Sense

After basic mining for positive and negative association rules in a database we obtain steady patterns of database items that occur together frequently. Let us say X and Y are database items and

$$\left.\begin{matrix} X \\ Y \end{matrix}\right\} \text{ are frequent } \quad XY \text{ is frequent} \qquad (1)$$

$X \Rightarrow Y$ high confidence

Also we obtain that

$$\left.\begin{matrix} X \sim Y \\ \text{or} \\ \sim XY \end{matrix}\right\} \text{ is infrequent} \qquad (2)$$

So we have a strong association rule (1), and we make sure that (2) are infrequent. (1) and (2) are our premises to check if one of the rules (3) has a high exceptionality, which would prove it is an exception in negative sense.

$$\left.\begin{matrix} X \Rightarrow \sim Y \\ \text{or} \\ \sim X \Rightarrow Y \end{matrix}\right\} \text{ if high exceptionality then Exception} \qquad (3)$$

Example. Consider two oil companies X and Y. Their stock normally goes up at the same time: $X \Rightarrow Y$ In the case when their shares do not go up at the same time $X \Rightarrow \sim Y$ we call the rule $X \Rightarrow \sim Y$ an exception if $X \sim Y$ is infrequent and has high exceptionality measure.

3.2.2 Exceptions in Positive Sense

After basic mining for positive and negative association in a database we obtain a steady pattern of database items. Let us say X and Y are database items and

X, Y } are frequent X~Y is frequent

X=>~Y or ~Y=>X } has high confidence (1)

Also we obtain that

XY is infrequent (2)

We have a strong negative association rule (1), and we make sure that (2) is infrequent. (1) and (2) are our premises to check if one of the rules (3) has a high exceptionality, which would prove it is an exception rule in positive sense.

X=>Y or Y=>X } if high exceptionality then Exception (3)

Example. Consider two oil companies X and Y. Their stock never goes up at the same time: X=>~Y In the case when their shares do go up at the same time X=>Y we call the rule X=>Y an exception if XY is infrequent and has high exceptionality measure.

3.3 Algorithm for Mining Exception Rules

Association rules are generated from frequent itemsets satisfying high confidence constraint. The confidence calculation is a straightforward procedure after all frequent itemsets have been generated. We do not consider the confidence calculation as it is easy and conceptually proven correct. The input of the exception rules mining algorithm are frequent 1-itemsets. The output of the algorithm is exceptional itemsets. Exceptional itemsets will become exception rules after the confidence of association rules has been checked.

We generate frequent itemsets and on each step k (k is the length of the itemset). We check the conditions (1), (2) from sections 3.2.1 and 3.2.2 and if they hold true, we check the exceptionality values for candidate exceptions. Figure 1 presents the Exceptional Itemsets Generation Algorithm.

4 Performance Evaluation

The proposed algorithm, Exceptional Itemsets Generation Algorithm, was implemented in Java 2 SDK 1.3 and tested on a PC: Celeron 1.3GHz, 128MB RAM. The test database was downloaded from the UCI Repository of machine learning databases [8]. The database used in the performance evaluation was Intrusion Detection database, which is former KDD Cup 1999 data to distinct the attacks on the network among the database records. The database represents parameters of a network over a period of time. The original database includes 40 parameters and a

```
k=1
1-freq_itemsets      // generate frequent 1-itemsets
k=2
2_candidate_itemsets          // generate candidate 2-itemsets
forEach c in 2_candidate_itemsets
 if (c frequent)       // verify the condition (1)
       if (negative_sets infrequent)        // verify the condition (2)
            {generate_2_Exc_cand_negative
            check_Exceptionality: true:   //  verify condition (3):
              if high Exceptionality of candidate
                    ExceptionalItemsets.Add }// add to the exceptional itemsets
    else
        if (negative_sets frequent)          // verify the condition (1)
            {generate_2_Exc_cand_positive
             check_Exceptionality: true:   // verify condition (3):
               if high Exceptionality of candidate
                    ExceptionalItemsets.Add } // add to the exceptional itemsets
 k++
```

Fig. 1. Exceptional Itemsets Generation Algorithm

vast number of records. Most of the parameters are continuous so in this work the simplified model of 10 parameters and 10 thousand attributes was employed. The parameters are listed in Figure 2. The parameters are either continuous or discrete.

The database sample has been chosen randomly from the original database. The continuous parameters values have been divided into ranges according to min/max values.

Exception rules mining starts with 1-frequent itemsets mining. The 2-candidate are then derived from 1-frequent itemsets and the minsup of 2-candidate itemsets is evaluated. The negative subsets are generated from 2-candidate itemset. If minimum support of 2-candidate itemset is greater or equal minsup and support of one of negative subsets is less than minsup, the pair is the candidate exception (and vice versa). For instance, for a candidate X Y Z, negative subsets {~X Y Z, X~Y Z,X Y~Z, ~X~Y Z,~X Y~Z,X~Y~Z} are generated. To verify the support of negative subsets, a special formula has been developed and tested:

Sup(negativeSubset)=

$$\mathrm{Sup}(\mathrm{positiveItems}) + \sum_{i=1}^{\mathrm{negNumber}} (-1)^i * \mathrm{Sup}(\mathrm{genSubset}(\mathrm{negativeItems}, i) + (-1)^i * \mathrm{Sup}(\mathrm{allItems})$$

In the negative subset V~X Y~Z positive items are V Y, negative items are X Z, all items are V X Y Z. genSubset generates all possible 1, 2, i, … combinations of negative items and concatenates them with positive items. Negative number is number of items negations in the itemset.

Sup(V~X Y~Z)=Sup(VZ)-Sup(VXY)-Sup(VYZ)+Sup(VXYZ);

negNumber(V~X Y~Z)=2;

There is no need in additional database scans to calculate the support of negative subsets. The support is calculated based on items positive support updated while searching frequent itemsets.

1. duration	length (number of seconds) of the connection	cont
2. protocol type	type of the protocol, e.g. tcp, udp, etc	discrete
3. flag	normal or error status of the connection	discrete
4. src_bytes	number of data bytes from source to destination	cont
5. dst_bytes	number of data bytes from destination to source	cont
6.urgent	number of urgent packets	cont
7.hot	number of ``hot'' indicators	cont
8.logged_in	1 if successfully logged in; 0 otherwise	discrete
9.Num_compromised	number of ``compromised'' conditions	cont
10.Num_file_creation	number of file creation operations	cont

Fig. 2. Network parameters

Figure 3 presents a graph of dependency between minimum support value and number of generated exception rules. Positive exceptions mean exceptions in positive sense, negative exceptions mean exceptions in negative sense (see 3.2.1, 3.2.2). Exception rules pictured in Figure 3 are the rules with the highest exceptionality measure among the candidate exception rules. The exceptions rules number is the average value for all minsup values.

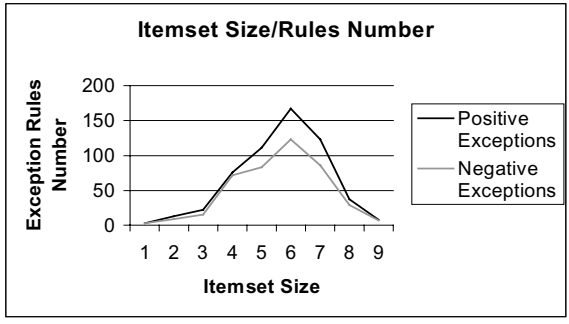

Fig. 3. Itemset Size/Rules Number Graph

Figure 4 presents minimum support value /number of generated exception rules dependency. The graph changes direction of falling/rising at different minsup values.

In Figure 5 there are a few samples of generated exception rules featuring high exceptionality value. Our algorithm generates exceptional itemsets that become exception rules after computationally simple confidence value verification. Frequent itemsets represent strong rules in the database. When an exception based on strong rule has been generated, it indicates something unusual like invasion detection in the network.

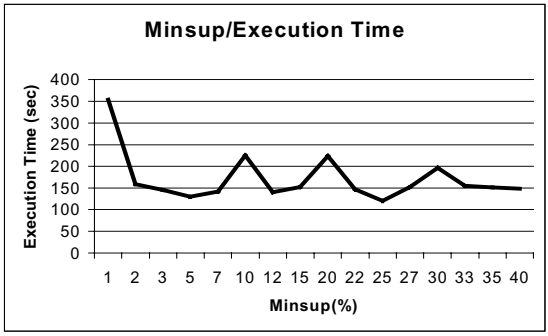

Fig. 4. Minsup/Execution Time Graph

Positive Exceptions	**Frequent Itemset**: protocol_type=tcp flag=SF urgent=0 dst_bytes C[0;500 000b] logged_in=1 **Exceptional Itemset:** protocol type=tcp flag=Not SF urgent=0 dst_bytes C[0;500 000b] logged_in=0
	Frequent Itemset: urgent=0 dst_bytes C[0;500 000b] hot C [0,9] logged_in=1 num_compromised=0 num_file_creations=0 **Exceptional Itemset:** urgent=0 dst_bytes C[0;500 000b] hot C [0,9] logged_in=1 num_compromised>0 num_file_creations>0
Negative Exceptions	**Frequent Itemset:** flag=SF source_bytes C [0;10000b] num_file_creations=0 **Exceptional Itemset:** flag=SF source_bytes C [0;10000b] num_file_creations>0
	Frequent Itemset: flag=REJ logged_in=0; **Exceptional Itemset:** flag=REJ logged_in=1;

Fig. 5. Samples of generated exceptions

5 Conclusion and Future Work

The project considers the interconnection between negative association rules and exceptions rules. The exceptions rules mining algorithm employs the knowledge about the negative association rules in the database and generates candidate exceptional itemsets. The candidate itemsets exceptionality measure is then verified. If the exceptionality satisfies the minimum exceptionality constraint, the candidate exceptional itemsets will be listed in the output of the algorithm as exceptional itemsets.

In the future work we are going to consider temporal exceptions, which are the temporal patterns in the database related with negative association rules and changing over time. Additional measures will be considered to distinct the temporal exceptions in the database.

References

[1] Agrawal, R., Imielinski, T. and Swami, A. "Mining association rules between sets of items in large databases". *Proceedings ACM-SIGMOD Int. Conf. Management of Data*, pp. 207-216, Washington, D.C., May 1993

[2] Chen, M., Han, J. and Yu, P. "Data Mining: An Overview from a Database Perspective" *IEEE Transactions on Knowledge and Data Engineering*, vol. 8, no. 6, pp. 866-883 1996

[3] Agrawal, R., Mannila, H., Srikant, R., Toivonen, H. and Verkamo, A. "Fast discovery of association rules" In *Fayyad et al, Advances in Knowledge Discovery and Data Mining, AAAI Press* pp. 307-328, 1996

[4] Agrawal, R. and Srikant, R. "Fast algorithms for mining association rules in large databases" *Proceedings of the 20th International Conference on Very Large Data Bases*, pp. 487-499, 1994

[5] Savasere, A., Omiecinski, E. and Navathe, S. "An efficient algorithm for mining association rules in large databases" *Proceedings of the International Conference on Very Large Data Bases,* pp. 432-444, 1995

[6] Liu, H., Lu, H., Feng, L. and Hussain, F. "Efficient Search of Reliable Exceptions" *Proceedings of the 3rd Pacific-Asia Conference on Knowledge Discovery and Data Mining* pp. 194-203, 1999

[7] Hussain, F., Liu, H., Suzuki, E. and Lu, H. "Exception Rule Mining with a Relative Interestingness Measure" *Proceedings of the 4th Pacific-Asia Conference on Knowledge Discovery and Data Mining* pp. 86-97, 2000

[8] Blake, C.L. & Merz, C.J. UCI Repository of machine learning databases [http://www.ics.uci.edu/~mlearn/MLRepository.html]. Irvine, CA: University of California, Department of Information and Computer Science, 1998.