Mining Financial Data with Scheduled Discovery of Exception Rules

Einoshin Suzuki

Electrical and Computer Engineering, Yokohama National University,
79-5 Tokiwadai, Hodogaya, Yokohama 240-8501, Japan
suzuki@dnj.ynu.ac.jp

Abstract. This paper shows preliminary results, on financial data, of an algorithm for discovering pairs of an exception rule and a common sense rule under a prespecified schedule. An exception rule, which represents a regularity of exceptions to a common sense rule, often exhibits interestingness. Discovery of pairs of an exception rule and a common sense rule under threshold scheduling has been successful in efficient discovery of interesting rules. In this paper, we apply it to financial data, which has been provided as a benchmark data set for data mining methods. Examples of discovered knowledge as well as a simple description of the approach are both provided in this paper.

Keywords: Rule discovery, Exception Rule, Threshold Scheduling, Financial Data, Data mining.

1 Introduction

An exception rule [2–6] represents a regularity of exceptions to a common sense rule which holds true, with high probability, for many examples. An exception rule is often useful since it represents a different fact from a common sense rule, which is often a basis of people's daily activities [5]. For instance, suppose a species of mushrooms most of which are poisonous but some of which are exceptionally edible. The exact description of the exceptions is highly beneficial since it enables exclusive possession of the edible mushrooms.

An undirected method obtains a set of pairs of an exception rule and a common sense rule [3–6]. Although this approach is relatively time-consuming, it often discovers unexpected rules since it is independent of user-supplied domain knowledge. Especially, an approach proposed by Suzuki [7] obtains a set of rule pairs by appropriately updating thresholds. This update is named as "threshold scheduling", and can potentially liberate users from worrying about specifications of thresholds in discovering interesting rules. In this paper, we apply this approach to financial data, which has been provided as a benchmark data set in Discovery Challenge 2000.

2 Undirected Discovery of Exception Rules

2.1 Problem Description

Let an atom represent an event which is either a single value assignment to a discrete attribute or a single range assignment to a continuous attribute. We define a conjunction rule as a production rule of which premise is represented by a conjunction of atoms and conclusion is a single atom.

Suzuki considered a problem of finding a set of rule pairs [3–7]. Here, a rule pair $r(x, x', Y_\mu, Z_\nu)$ is defined as a pair of two conjunction rules, which are a common sense rule $Y_\mu \rightarrow x$ and an exception rule $Y_\mu \wedge Z_\nu \rightarrow x'$.

$$r(x, x', Y_\mu, Z_\nu) \equiv \{Y_\mu \rightarrow x,\ Y_\mu \wedge Z_\nu \rightarrow x'\} \tag{1}$$

where x and x' are a single atom with the same attribute but different values. Each premise of rules represents a conjunction of atoms $Y_\mu \equiv y_1 \wedge y_2 \wedge \cdots \wedge y_\mu,\ Z_\nu \equiv z_1 \wedge z_2 \wedge \cdots \wedge z_\nu$.

The method employed in this paper outputs rule pairs which satisfy

$$\widehat{\Pr}(Y_\mu) \geq \theta_1^{\mathrm{S}} \tag{2}$$

$$\widehat{\Pr}(x|Y_\mu) \geq \mathrm{MAX}(\theta_1^{\mathrm{F}}, \widehat{\Pr}(x)) \tag{3}$$

$$\widehat{\Pr}(Y_\mu, Z_\nu) \geq \theta_2^{\mathrm{S}} \tag{4}$$

$$\widehat{\Pr}(x'|Y_\mu, Z_\nu) \geq \mathrm{MAX}(\theta_2^{\mathrm{F}}, \widehat{\Pr}(x')) \tag{5}$$

$$\widehat{\Pr}(x'|Z_\nu) \leq \mathrm{MIN}(\theta_2^{\mathrm{I}}, \widehat{\Pr}(x')) \tag{6}$$

where $\widehat{\Pr}(x)$ represents the ratio of an event x in the data set, and each of $\theta_1^{\mathrm{S}}, \theta_1^{\mathrm{F}}, \theta_2^{\mathrm{S}}, \theta_2^{\mathrm{F}}, \theta_2^{\mathrm{I}}$ is a threshold.

2.2 Discovery Algorithm

In an undirected discovery method for exception rules [6], a discovery task is viewed as a search problem, in which a node of a search tree represents four rule pairs $r(x, x', Y_\mu, Z_\nu),\ r(x', x, Y_\mu, Z_\nu),\ r(x, x', Z_\nu, Y_\mu),\ r(x', x, Z_\nu, Y_\mu)$. Since these four rule pairs employ common atoms, this fourfold representation improves time efficiency by sharing calculations of probabilities of these atoms. Each node has four flags for representing rule pairs in consideration.

In the search tree, a node of depth one represents a single conclusion x or x'. Likewise, a node of depth two represents a pair of conclusions (x, x'). Let $\mu = 0$ and $\nu = 0$ represent the state in which the premise of a common sense rule and the premise of an exception rule contain no atoms respectively, then we see that $\mu = \nu = 0$ holds true in a node of depth two. As the depth increases by one, an atom is added to the premise of the common sense rule or of the exception rule. A node of depth three satisfies either $(\mu, \nu) = (0, 1)$ or $(1, 0)$, and a node of depth l (≥ 4), $\mu + \nu = l - 2$ $(\mu, \nu \geq 1)$.

A depth-first search method is employed to traverse this tree, and the maximum search depth M is given by the user. Note that $M - 2$ is equal to the maximum value for $\mu + \nu$. Now, we introduce a theorem [7] for efficient search.

Theorem 1. *Let* $\widehat{\Pr}(Y_{\mu'}) \geq \widehat{\Pr}(Y_{\mu})$, $\widehat{\Pr}(Z_{\nu'}) \geq \widehat{\Pr}(Z_{\nu})$ *and a rule pair* $r(x, x', Y_{\mu'}, Z_{\nu'})$ *satisfies at least an equation in (7) - (11), then a rule pair* $r(x, x', Y_{\mu}, Z_{\nu})$ *does not satisfy all of (2) - (6).*

$$\widehat{\Pr}(Y_{\mu'}) < \theta_1^{\mathrm{S}} \tag{7}$$

$$\widehat{\Pr}(x, Y_{\mu'}) < \theta_1^{\mathrm{S}} \mathrm{MAX}(\theta_1^{\mathrm{F}}, \widehat{\Pr}(x)) \tag{8}$$

$$\widehat{\Pr}(Y_{\mu'}, Z_{\nu'}) < \theta_2^{\mathrm{S}} \tag{9}$$

$$\widehat{\Pr}(x', Y_{\mu'}, Z_{\nu'}) < \theta_2^{\mathrm{S}} \mathrm{MAX}(\theta_2^{\mathrm{F}}, \widehat{\Pr}(x')) \tag{10}$$

$$\widehat{\Pr}(Z_{\nu'}) < \theta_2^{\mathrm{S}} \mathrm{MAX}(\theta_2^{\mathrm{F}}, \widehat{\Pr}(x'))/\mathrm{MIN}(\theta_2^{\mathrm{I}}, \widehat{\Pr}(x')) \tag{11}$$

The proof is straightforward, and is given in [7].

If a rule pair $r(x, x', Y_{\mu'}, Z_{\nu'})$ satisfies at least one of (7) - (11), the flag which corresponds to this rule pair is turned off. In expanding a search node, we define that information about flags is inherited to its children nodes. Thus, $r(x, x', Y_{\mu}, Z_{\nu})$ is ignored in the descendant nodes without changing the output. If the four flags are all turned off in a search node, the node is not expanded.

2.3 Threshold Scheduling

Specification for values of thresholds $\theta_1^{\mathrm{S}}, \theta_2^{\mathrm{S}}, \theta_1^{\mathrm{F}}, \theta_2^{\mathrm{F}}, \theta_2^{\mathrm{I}}$ is difficult in some cases. For example, if the data set contains few exception rules, strict values result in few or no discovered patterns. We can avoid this situation by settling the values loosely, but if the data set contains many exception rules, too many patterns would be discovered at the end of long computational time. Both cases are undesirable from the viewpoint of knowledge discovery.

In order to cope with these problems, we have invented a novel data structure which efficiently manages discovered patterns with multiple indices. This data structure consists of several AVL trees [8] each of which has η nodes. In order to realize flexible scheduling, we assign a tree for each index. A node of a tree represents a pointer to a discovered pattern, and this enables fast transformation of the tree. We show, in figure 1, an example of this data structure which manages seven rule pairs $r1, r2, \cdots, r7$.

In updating values of thresholds, users either have a concrete plan or have no such plans. We show two scheduling policies for each situation. In the first policy, a user specifies threshold updating $(\theta_1, v_1), (\theta_2, v_2), \cdots, (\theta_\xi, v_\xi)$ where θ_i, v_i are a threshold and its value respectively. When each AVL tree has η nodes and a new rule pair is being discovered, a value of a threshold θ_i is updated to v_i according to current specification. Then this method deletes each stored rule pair which satisfies $\theta_i \leq v_i$ and the nodes in the trees each of which has a pointer to the rule pair. In the second policy, a user specifies a total order to a subset of indices $\widehat{\Pr}(Y_{\mu})$, $\widehat{\Pr}(Y_{\mu}, Z_{\nu})$, $\widehat{\Pr}(x|Y_{\mu})$, $\widehat{\Pr}(x'|Y_{\mu}, Z_{\nu})$, $\widehat{\Pr}(x'|Z_{\nu})$. This method iteratively deletes a rule pair based on an AVL tree which is selected according to this order. Suppose each AVL tree has η nodes and a new rule pair is being discovered. If the current index is $\widehat{\Pr}(x'|Z_{\nu})$, the maximum node of the AVL tree

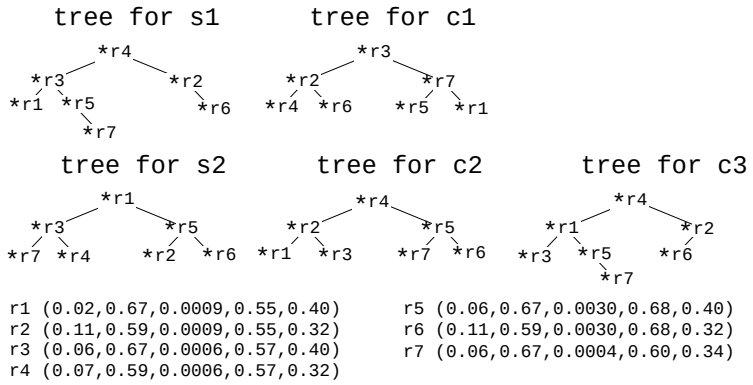

Fig. 1. Illustration based on proposed data structure, where an AVL tree is employed as a balanced search-tree, and keys are s1 ($\widehat{\Pr}(Y_\mu)$), c1 ($\widehat{\Pr}(x|Y_\mu)$), s2 ($\widehat{\Pr}(Y_\mu, Z_\nu)$), c2 ($\widehat{\Pr}(x'|Y_\mu, Z_\nu)$), c3 ($\widehat{\Pr}(x'|Z_\nu)$). Numbers in a pair of parentheses represent values of these indices of the corresponding rule pair. In a tree, $*r1, *r2, \cdots, *r7$ represent a pointer to the rule pairs r1, r2, $\cdots$, r7 respectively.

for $\widehat{\Pr}(x'|Z_\nu)$ is deleted. Otherwise, the minimum node of current AVL tree is deleted. The same deletion procedure occurs as in the first policy, and the value of the corresponding threshold is updated. Details of the algorithms are given in [7].

3 Experimental Evaluation

3.1 Conditions

Based on the previous Discovery Challenge [1], the financial data was preprocessed as follows.

1) We joined "account.asc", "disp.asc", "card.asc", "client.asc", and "loan.asc", where an example represents an account. In "disp.asc", we only recorded the existence of a second client (disponent), and ignored his client ID and account ID. The "birth number" of "client.asc" was transformed to two attributes: birth year and sex of a client.

2) We joined "district.asc" to this table which has two district IDs. These IDs concern the account and its client, and sometimes differ. In the join process, we treated these IDs individually.

3) We joined "order.asc" to this table by creating two attributes for each value "?", "LEASING", "POJISTN", "SIPO", and "UVER" of the attribute "k_symbol". These two attributes represent the total number and amount of orders. In the join process, we ignored attributes which concern the recipient.

4) We joined "trans.asc" to this table by summarizing the transactions which were executed before a loan is granted. In the join process, we considered only accounts with a loan. We created three attributes which represent the number

of transactions (TransNum), the number of days from creation of an account to the grant of a loan (TransDay), and the average balance per day (TransIbal). We also created two attributes which represent the total number and amount of each type of transactions. Basically, these types were created based on the attribute "k_symbol" which includes "POJISTNE", "SLUZBY", "UROK", "SANKC. UROK", "SIPO", "DUCHOD", and "UVER". When the value of "k_symbol" is missing, we judged the type of a transaction based on the attribute "operation" which includes "VYBER KARTOU", "VKLAD", "PREVOD Z UCTU", "VYBER", and "PREVOD NA UCET".

5) Finally, we settled the task to discovering exception rules each of which concerns good loans and bad loans. We transformed the value "a", "b", "c", and "d" of the attribute "loan status" to "G", "B", "G", and "B" respectively. We deleted several attributes which were judged irrelevant such as IDs. We also deleted attributes which concern unemployment rates and committed crimes because several loans were granted before these statistics were available.

The final data set includes 69 attributes about 682 accounts. In the data set, 55 attributes are continuous, and the other attributes are nominal with 2 to 77 values. Table 1 shows these attributes. We later ignored several attributes which had a unique value for all accounts and two attributes which concern 77 district names due to time efficiency.

Table 1. Attributes of the final data set. In district, "ad" and "cd" represent account district and customer district respectively.

data set	attributes
loan	loanStatus
account	frequency, accountDate
disp	disponent
card	cardType, cardDate
client	birthYear, sex
loan	loanDate, loanAmount, loanDuration, loanPayments
district	accountDistrict, accountRegion, adInhabitants, adMuni499, adMuni1999, adMuni9999, adMuni10000, adCities, adrUrbanInhabitants, adAverageSalary, adEntrepreneurs, customerDistrict, customerRegion, cdInhabitants, cdMuni499, cdMuni1999, cdMuni9999, cdMuni10000, cdCities, cdrUrbanInhabitants, cdAverageSalary, cdEntrepreneurs
order	? Num, ? Amount, LeasingNum, LeasingAmount, PojistnNum, PojistnAmount, SipoNum, SipoAmount, UverNum, UverAmount
trans	TransNum, TransDay, TransIbal, TransPojistneNum, TransPojistneAmount, TransSluzbyNum, TransSluzbyAmount, TransUrokNum, TransUrokAmount, TransSankcurokNum, TransSankcurokAmount, TransSipoNum, TransSipoAmount, TransDuchodNum, TransDuchodAmount, TransKartouNum, TransKartouAmount, TransVkladNum, TransVkladAmount, TransUctuNum, TransUctuAmount, TransVyberNum, TransVyberAmount, TransUcetNum, TransUcetAmount

3.2 Good and Bad loans

All experimental results presented in this paper are computed on several personal computers with Linux operating systems. Two indices $s2(= \widehat{\Pr}(Y_\mu, Z_\nu))$ and $c2(= \widehat{\Pr}(x'|Y_\mu, Z_\nu))$ are specified in this order with the second scheduling policy presented in the previous section. In the experiments, we settled the maximum number of discovered rule pairs to $\eta = 50$ and the maximum search depth to $M = 5$. The attributes were discretized using the method described in [7] with the number of bins chosen as 4, 5 and 6.

In these experiments, the loan status is the only attribute allowed in the conclusions. We settled initial values of the thresholds loosely to $\theta_1^{\mathrm{S}} = 0.008$, $\theta_2^{\mathrm{S}} = 10/682$, $\theta_1^{\mathrm{F}} = 0.4$, $\theta_2^{\mathrm{F}} = 0.4$, $\theta_2^{\mathrm{I}} = 0.7$, and applied the proposed method. For each trial with different numbers of bins, 50 rule pairs were discovered, and the final values of the thresholds were $\theta_2^{\mathrm{S}} = 0.014,\ 0.017,\ 0.024$, $\theta_2^{\mathrm{F}} = 0.4,\ 0.466,\ 0.444$ respectively. In the applications, $4.47{*}10^7$, $1.21{*}10^8$, $3.42{*}10^8$ nodes were searched, and the number of rule pairs stored in the data structure (including those which were deleted) were 282, 316, and 276 respectively. With the use of the scheduling capability, it took only 2h40, 6h45, and 17h to obtain the results respectively. Note that these experiments were done on PCs.

Below, we show an example of rule pairs discovered in these experiments, where a "+" represents the premise of the corresponding common sense rule. We also substitute $ns1\ (= n\widehat{\Pr}(Y_\mu))$, $ns2\ (= n\widehat{\Pr}(Y_\mu, Z_\nu))$ and $ns2c2\ (= n\widehat{\Pr}(x', Y_\mu, Z_\nu))$ for $s1$, $s2$ and $c2$ since they are easier to be understood, where n is the number of examples in the data set, i.e. 682.

$214 \leq$ TransDay $\leq 393 \rightarrow$ loanStatus = G
$+\ 950125 \leq$ accountDate ≤ 970111, $13 \leq$ TransUrokNum $\leq 35 \rightarrow$ loanStatus = B
$ns1 = 225,\ c1 = 0.893,\ ns2 = 18,\ ns2c2 = 10,\ c3 = 0.107$

According to this rule pair, 89.3 % of the 225 accounts whose transaction days were between 214 and 393 had good loan status. However, 10 among 18 of them (which was created between 950125 and 970111, and whose number of transactions for "urok" was between 13 and 35 in addition to the above condition) had bad loan status. This exception rule is unexpected and interesting since only 10.7 % of the account which was created between 950125 and 970111, and whose number of transactions for "urok" was between 13 and 35 had bad loan status.

Below are other examples of discovered rule pairs.

$100 \leq$ customer district Entrepreneurs ≤ 167, $16 \leq$ TransUrokNum $\leq 57 \rightarrow$ loanStatus=G
$+\ 6 \leq$ TransSipoNum $\leq 9 \rightarrow$ loanStatus = B
$ns1 = 206,\ c1 = 0.902,\ ns2 = 17,\ ns2c2 = 8,\ c3 = 0.094$

$1901 \leq$ UverAmount ≤ 9910, $1 \leq$ TransSipoNum $\leq 9 \rightarrow$ loanStatus = G
$+ 16 \leq$ TransUrokNum $\leq 35 \rightarrow$ loanStatus = B
$ns1 = 189,\ c1 = 0.888,\ ns2 = 20,\ ns2c2 = 9,\ c3 = 0.104$

$16 \leq$ TransUrokNum $\leq 35 \rightarrow$ loanStatus = G
$+ 1901 \leq$ loanPayments ≤ 9910, $1 \leq$ TransSipoNum $\leq 9 \rightarrow$ loanStatus = B
$ns1 = 230,\ c1 = 0.895,\ ns2 = 20,\ ns2c2 = 9,\ c3 = 0.111$

$5.82 \leq$ TransUrokAmount $\leq 15.93 \rightarrow$ loanStatus = G
$+ 529.41 \leq$ TransIbal ≤ 41725.33, $364.31 \leq$ TransUctuAmount $\leq 1965.37 \rightarrow$ loanStatus = B
$ns1 = 274,\ c1 = 0.901,\ ns2 = 12,\ ns2c2 = 6,\ c3 = 0.108$

$1948 \leq$ birthYear $\leq 1958 \rightarrow$ loanStatus = G
$+ 42821 \leq$ account district Inhabitants ≤ 88884, $1 \leq$? Num $\leq 2 \rightarrow$ loanStatus = B
$ns1 = 177,\ c1 = 0.903,\ ns2 = 11,\ ns2c2 = 5,\ c3 = 0.096$

$529.41 \leq$ TransIbal ≤ 38594.03, $364.31 \leq$ TransUctuAmount $\leq 1965.37 \rightarrow$ loanStatus=G
$+5.33 \leq$ TransUrokAmount $\leq 15.93 \rightarrow$ loanStatus = B
$ns1 = 85,\ c1 = 0.894,\ ns2 = 12,\ ns2c2 = 5,\ c3 = 0.099$

$2 \leq$ PojistnAmount $\leq 9115 \rightarrow$ loanStatus = G
$+1948 \leq$ birthYear ≤ 1958, $42821 \leq$ cdInhabitants $\leq 122603 \rightarrow$ loanStatus = B
$ns1 = 114,\ c1 = 0.912,\ ns2 = 12,\ ns2c2 = 5,\ c3 = 0.104$

accountRegion = centralBohemia $\rightarrow$ loanStatus = G
$+ 106 \leq$ customer district Entrepreneurs ≤ 114, $4.15 \leq$ TransUrokAmount $\leq 6.83 \rightarrow$ loanStatus = B
$ns1 = 90,\ c1 = 0.888,\ ns2 = 12,\ ns2c2 = 5,\ c3 = 0.102$

$529.41 \leq$ TransIbal ≤ 38594.03, $3 \leq$ TransUctuNum $\leq 24 \rightarrow$ loanStatus=G
$+$ loanDuration $= 12 \rightarrow$ loanStatus = B
$ns1 = 85,\ c1 = 0.894,\ ns2 = 12,\ ns2c2 = 5,\ c3 = 0.083$

4 Conclusions

In this paper, we applied, to the financial data, simultaneous discovery of an exception rule and a common sense rule with a scheduling capability [7]. The method updates values of multiple thresholds appropriately according to the discovery process, and discovers appropriate number of rule pairs in reasonable time. Several examples of rule pairs are shown as preliminary results.

Acknowledgement

This work was partially supported by the grant-in-aid for scientific research on priority area "Discovery Science" from the Japanese Ministry of Education, Science, Sports and Culture.

References

1. P. Berka: *Workshop Notes on Discovery Challenge*, Univ. of Economics, Prague, Czech Republic (1999).
2. B. Padmanabhan and A. Tuzhilin: "A Belief-Driven Method for Discovering Unexpected Patterns", *Proc. Fourth Int'l Conf. Knowledge Discovery and Data Mining (KDD)*, AAAI Press, Menlo Park, Calif., pp. 94–100 (1998).
3. E. Suzuki and M. Shimura : Exceptional Knowledge Discovery in Databases Based on Information Theory, *Proc. Second Int'l Conf. Knowledge Discovery and Data Mining (KDD)*, AAAI Press, Menlo Park, Calif., pp. 275–278 (1996).
4. E. Suzuki: "Discovering Unexpected Exceptions: A Stochastic Approach", *Proc. Fourth Int'l Workshop Rough Sets, Fuzzy Sets, and Machine Discovery (RSFD)*, Japanese Research Group on Rough Sets, Tokyo, pp. 225–232 (1996).
5. E. Suzuki: "Autonomous Discovery of Reliable Exception Rules", *Proc. Third Int'l Conf. Knowledge Discovery and Data Mining (KDD)*, AAAI Press, Menlo Park, Calif., pp. 259–262 (1997).
6. E. Suzuki and Y. Kodratoff: "Discovery of Surprising Exception Rules based on Intensity of Implication", *Principles of Data Mining and Knowledge Discovery (PKDD), LNAI 1510*, Springer-Verlag, Berlin, pp. 10–18 (1998).
7. E. Suzuki: "Scheduled Discovery of Exception Rules", *Discovery Science (DS), LNAI 1721*, Springer-Verlag, Berlin, pp. 184–195 (1999).
8. N. Wirth: *Algorithms + Data Structures = Programs*, Prentice-Hall, Englewood Cliffs, N.J. (1976).