Johann Wolfgang Goethe-Universität
Frankfurt am Main

Fachbereich Informatik

Credit Card Fraud Detection by Adaptive Neural Data Mining

R. Brause, T. Langsdorf, M. Hepp

INTERNER BERICHT 7/99

Fachbereich Informatik
Robert-Mayer-Straße 11-15
60054 Frankfurt am Main

Credit Card Fraud Detection by Adaptive Neural Data Mining

R. Brause,[1] T. Langsdorf[1], M. Hepp[2]
[1]*J.W.Goethe-University, Frankfurt a. M.,*
[2]*Gesellschaft f. Zahlungssysteme GZS, Frankfurt a. M., Germany*

Abstract

The prevention of credit card fraud is an important application for prediction techniques. One major obstacle for using neural network training techniques is the high necessary diagnostic quality: Since only one financial transaction of a thousand is invalid no prediction success less than 99.9% is acceptable.

Due to these credit card transaction proportions complete new concepts had to be developed and tested on real credit card data. This paper shows how advanced data mining techniques and neural network algorithm can be combined successfully to obtain a high fraud coverage combined with a low false alarm rate.

1 Introduction

The prediction of user behavior in financial systems can be used in many situations. Predicting client migration, marketing or public relations can save a lot of money and other resources. One of the most interesting fields of prediction is the fraud of credit lines, especially credit card payments. In our case, for the high data traffic of 400,000 transactions per day with the fraud of 10 million $ per year, a reduction of 10% of fraud triggers a saving of one million of dollars per year. The goal is clear: How can we save this money?

Certainly, all transactions which deal with accounts of known fraud are not authorized. Nevertheless, there are transactions which are formally valid, but experienced people can tell that these transactions are probably misused, caused by stolen cards or fake merchants. So, the task is to avoid a fraud by a credit card transaction *before* it is known as "illegal".

With an increasing number of transactions people can no longer control all of them. As remedy, one may catch the experience of the experts and put it into an expert system. This traditional approach has the disadvantage that the expert's knowledge, even when it can be extracted explicitly, changes rapidly with new kinds of organized attacks and patterns of credit card fraud. In order to keep track with this, no predefined fraud models as in [7] but automatic learning algorithms are needed.

This paper deals with the problems specific to this special data mining application and tries to solve them by a combined probabilistic and neuro-adaptive approach for a given data base of credit card transactions of the GZS.

1.1 The goal of fraud detection

The objective of the diagnosis can be formulated by the commonly used diagnostic scheme shown in Table 1.

Data\diagnosis	legal	fraud
legal	P(correct|legal)	P(false alarm|legal)
fraud	P(fraud not detected)	P(correct|fraud)

Table 1 *The outcome probability table*

A high correct diagnostic probability

$$P(\text{correct}) = P(\text{correct} \mid \text{fraud})\, P(\text{fraud}) + P(\text{correct} \mid \text{legal})\, P(\text{legal}) \quad (1.1)$$

can be obtained by minimizing the (generally weighted) sum

$$R = r_1\, P(\text{fraud not det.}) + r_2\, P(\text{false alarm}|\text{legal}) \quad (1.2)$$

Our objective function R to be minimized is determined by the costs r_1,r_2 which are implied by the wrong decisions. In practice, r_1 and r_2 are difficult to determine exactly. Therefore, we focus on minimizing the false alarm rate and the probability of not detected fraud at the same time. In principal, we are aiming for maximizing the number of fraud transactions correctly recognized and minimizing the number of false alarms in order to minimize the fraud costs.

For the *false alarm rate* FAR we know that

$$\text{FAR} = \frac{\#\text{false alarms}}{\#\text{all alarms}} \geq \frac{\#\text{false alarms}}{\#\text{all alarms}} \cdot \frac{\#\text{all alarms}}{\#\text{all legals}}$$

$$= \frac{\#\text{false alarms}}{\#\text{all legals}} = P(\text{false alarm}|\text{legal}) \quad (1.3)$$

because $1 \geq \frac{\#\text{all alarms}}{\#\text{all legals}}$ in most cases.

As we will see later, the false alarm rate is very sensitive for diagnostic changes whereas the probability of detected fraud is subject to smaller changes. Thus, we concentrate on minimizing the FAR = P(false alarm) while maintaining an acceptable high level of P(correct|fraud).

1.2 Modeling the data

The transaction data are characterized by some very special proportions:

- The probability of a fraud transaction is very low (0.2%) and has been lowered in a preprocessing step by a conventional fraud detecting system down to 0.1%.
- Most of the 38 data fields (about 26 fields) per transaction contain symbolic data as merchant code, account number, client name etc.
- A symbolic field can contain as low as two values (e.g. the kind of credit card) up to several hundred thousand values (as the merchant code).
- The confidence limit for a transaction abort is very subjective and subject to client policy. Transactions with a confidence for fraud of higher than 10% are accepted to be revised or aborted.

These data proportions have several implications. For the very low fraud occurrence of only 0.1% a constant, "stupid" diagnosis of "transaction is no fraud" for all transactions will have a success rate of P(correct) = 99.9%. All adaptive fraud diagnosis which has lower success than this 99.9% (e.g. [3] with 92.5% or [9] with 50%) is questionable. The outcome probability Table 1 becomes

Data\diagnosis	legal	fraud
legal	100%	0%
fraud	100%	0%

Table 2 *The outcome probability table of the constant diagnosis*

1.3 Preprocessing the data

One of the most tedious operations is the normalization of the data. For half a million transactions of a sample interval which we analyzed we used the following operations:

- All data can be (but must not be) produced by the chain *transaction authorization request – transaction authorization – transaction fraud claim.* In order to produce just one transaction record, all different transactions of one account concerning the same money transfer must be merged to one record. Transactions which reflect only status changes (as credit limit changes, etc.) are sorted out.

- Additionally, the resulting 5,850 fraud transactions and 542,858 legal transactions are ordered by their time stamps.

- Then, the ASCII data values of the symbolic raw data are converted into enumerated data entries. For non-available transaction features special symbols are used.

All these operations resulted in our normalized data base of fixed records of numbers and no text.

2 Mining the symbolic data

The task of recognizing the transactions with fraud is very demanding. One idea is learning the feature associations for credit fraud. In neural network literature, there are several models of learning association rules by associative memory. Let us consider the most popular one, the correlation matrix memory [5], and show its problems for our application. After this, we will deduce an alternative learning model.

2.1 Modeling associations by binary associative memory

Let us combine all symbolic features x_k of one transaction in one data tuple $\mathbf{x} = (x_1,...,x_n)$. Here, we encode every feature such that it consists only of binary variables. For example, take a four-valued feature $x_k \in \{a,b,c,d\}$. This feature may be encoded using a new set of four binary-valued features x_{ka}... x_{kd}, see Fig. 1.

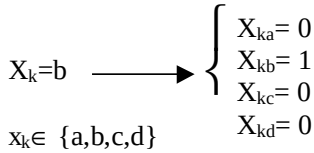

Fig. 1 *Encoding of a symbolic variable with 4 values*

In the figure, for the feature value x_k=b the new feature x_{kb} becomes 1 and all others of the set zero.
Now, each transaction can be associated to a result M_i, a fraud of type *i*, by storing the tuple ($\mathbf{x}$, M_i) in a binary associative memory. The complete associative device is shown in Fig. 2.
Without going into details (see [10],[5]), the learning rule updates the association weights (shown as thick dots in Fig. 2). On input of a transaction tuple an association between a fraud M_j and an input tuple x is triggered and the output M_j becomes 1.

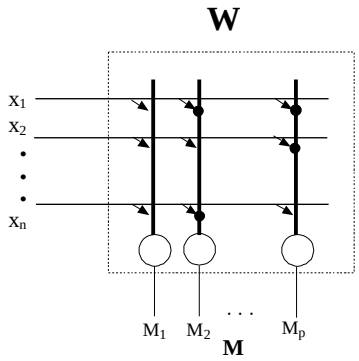

Fig. 2 *A binary associative memory structure yields the associations (x_1 x_n M_2) and (x_1 x_2 M_p)*

The regular binary encoding yields a very regular scheme, using a fixed threshold for the associative read-out operation.
This approach has several flaws:

- The input-output mapping, the association, is not weighted internally. This means that a fraud association can be triggered either by several different transactions with different occurrence probabilities or by just one. This situation is not adequately reflected by the device, not even by weighting the output.
- For features with very many possible values (e.g. several hundred thousand ones) the resulting binary inputs are encoded very sparsely by just one active input. This means that we have a very large association matrix **W** with a very small number of weights. This is inefficient to implement.
- The learning (and therefore the network activity) of this model is not based on accurate probabilities of the input, but on quite arbitrary learning rates.
- The "unlearning", i.e. the change of probabilities, is not reflected in the learning mechanism.
- There is no generalization mechanism defined in order to reduce the dependence of an association on unimportant input.

Let us consider all theses problems and try to modify our model according to the needs.

First, the inefficient implementation can be overcome by treating all transactions as association rules and store them as they are. This avoids the necessity of wasting huge amounts of memory for zeros. Special learning rules will reflect the necessity for change and adaptation to the probabilities of reality.

Nevertheless, the necessity for generalization, importance and probability weighting still remains.

2.2 Generalizing and weighting the association rules

The fraud transactions can not be used as fraud rules directly; they are too special and too many, they have to be generalized. In contrast to standard basket prediction association rules [1], [2], [6] our goal does not consist of generating long associating rules but of shortening our raw associations by generalizing them to the most common types of transactions. Additionally, we do not have binary features but features with multiple possible values. The excessive number of possible values of some features prohibits a mapping to new binary features as already mentioned for the binary associative memory. Although generalizations are common for symbolic AI, there are no standard algorithms in data mining to do this. For instance, all algorithms which compute all possible generalizations and then select only those rules according to some strategy [1], [2], [6] which fits the data sufficiently can not be used, because the set of all generalizations is too big in our case.

Now, how can such a generalization be done? We start with the data base of fraud transactions and compare each transaction with all others in order to find pairs of similar ones. Each pair is then merged into a generalized rule by replacing a non-identical feature by a 'don't-care'-symbol '*'. By doing so, a generalization process evolves, see Fig. 3. Here, the generalization of two transactions with the feature tuples $\mathbf{x}_1$ = (F,D,C,D,A) and $\mathbf{x}_2$ = (F,D,G,D,A) (dotted circle) to the rule (F,D,*,D,A) and further up to (F,*,*,D,A) and to (*,*,*,D,*) is shown.

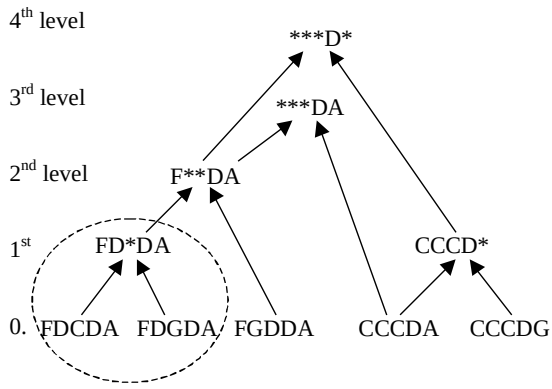

Fig. 3 *The generalization graph*

Thus, all raw transactions can be seen as association rules of level zero; each generalization provides at least one 'don't-care'-symbol for an unimportant feature, increases the generalization level by one and shortens the rule excluding one feature. All generalizations which have not been generalized themselves are the root of a subgraph, each forming a tree.

Rule	ACCT_NBR	TRN_TYP	CURR_CD	POS_ENT_CD	FAL_SCOR	CRD_TYP	ICA_CD	AID_CD	SIC_CD	ACT_CD	MSG_TYP	MER_ID	MER_CNTY_CD	CTY_1	POST_CD_1	CNTY_CD1	CR_LMT	ATV_IND	ACCT_STAT	CTY_2	POST_CD_2	ADDR_STAT	EMIT_NBR	INST_NBR	ISS_REAS	GEN_CD	CARD_TYP
1	*	EA	840	*	*	EM	2768	8403184	*	0	1100	*	0	*	*	*	*	*	I	*	*	*	0	*	*	N	*	*
2	*	EA	840	[1]	0	EM	*	*	563	0	1100	*	0	*	*	*	*	*	I	*	*	*	0	*	*	*	*	*
3	*	EA	840	*	0	EM	2768	8403184	*	*	1100	*	0	*	*	*	*	*	I	*	*	*	0	*	*	*	002	*
4	*	EA	840	*	995	EM	*	*	*	0	1100	*	0	*	0	*	*	I	FR	*	*	0	*	*	*	*	*	*

[1] ZZUTSZ1UZZZ1

Table 3 *Generalized transactions with 16 wildcards*

For the example of 5850 fraud data, there are 4 generalized rules in level 16 shown in Table 3. The feature names are labeled on the top of the columns. All rules differ from each other.

In general, there are many rules in a level. What are the most important ones? Certainly, rules which are often used are more important than the others.
Thus, the occurrence probability or the relative number of transactions which are covered by that rule, the *support*, should be high.

$$\text{support} = \frac{\#\text{of transactions covered by the rule}}{\#\text{of transactions}} \qquad (2.4)$$

Especially, we define the *share* of a fraud rule as that relative part of fraud transactions which is covered by the rule

$$\text{share} = \frac{\#\text{of fraud transactions covered by the rule}}{\#\text{of fraud transactions}} \qquad (2.5)$$

Nevertheless, neither the support nor the share reflects the fact that there are also legal transactions which may fit a fraud rule and leading to a wrong diagnosis. The more transactions with a correct diagnosis we have the more confidence in the diagnostic process we get. We define therefore the *confidence* in a fraud diagnosis as

$$\text{confidence} = \frac{\#\text{of fraud transactions covered by the rule}}{\#\text{of transactions covered by the rule}} = \frac{\#\text{ correct alarms}}{\#\text{all alarms}} \qquad (2.6)$$

With

$$\frac{\#\text{ correct alarms}}{\#\text{all alarms}} = \frac{\#\text{all alarms} - \#\text{ false alarms}}{\#\text{all alarms}} = 1 - \frac{\#\text{ false alarms}}{\#\text{all alarms}} = 1 - P(\text{false alarm})$$

we know with eq. (1.3) that

$$\text{confidence} = 1 - P(\text{false alarm}) \le 1 - P(\text{false alarm}|\text{legal}).$$

Thus, when the confidence is maximized, the probability of a false alarm is minimized.

For all legal transactions, each one of the rules in Table 3 has a confidence bigger than 10% and a share bigger than 1%. All three measures, preceded by the absolute number of fraud transactions MTA and legal ones LTA for which they trigger an alarm, are evaluated for Table 3 and shown in Table 4.

Rule	MTA	LTA	Support	Confidence	Share
1	690	500	0.011	11.3%	12%
2	78	47	0.001	13.3%	1%
3	267	64	0.004	27.9%	5%
4	42	0	0.001	100%	1%

Table 4 *The three importance measures for the examples in Table 3*

Before using the definitions (2.1)-(2.3) the numbers of LTA have been adjusted to reflect the real proportion MTA:LTA of 1:1000.

How do the measures defined so far change by the generalization of the rules? We know that the share, the relative number of fraud transactions covered by a rule, will increase when we allow more possible values of a feature. Thus, the share only increases by generalization, see the proof in appendix A, theorem 1. Additionally, a generalization can not increase the confidence, but only decrease it or be constant. The proof for this is shown in appendix A, theorem 2.

2.3 Diagnostic implementation issues

The rule based diagnostic system described so far can be implemented in many different manners. For time critical applications the diagnostic rules can be stored in conventional hardware based content addressable memory (CAM), implemented for instance with low cost FPGAs and yielding a runtime speed gain of 50 relative to a software solution. Each time a transaction is fed to the CAM, one or several hits which eventually occur will indicate a fraud transaction.

Another possible implementation is the conversion of the parallel decision table (the rules) to a sequential decision procedure, i.e. to a decision tree, avoiding all not necessary comparisons [4]. In Fig. 4 a binary decision tree is shown which corresponds to the set of four fraud detection rules of level 16 shown in of Table 3.
The alarm is given when one of the rules are fulfilled, i.e. a exit “M” is reached. Otherwise, the program proceeds (exit “P”).

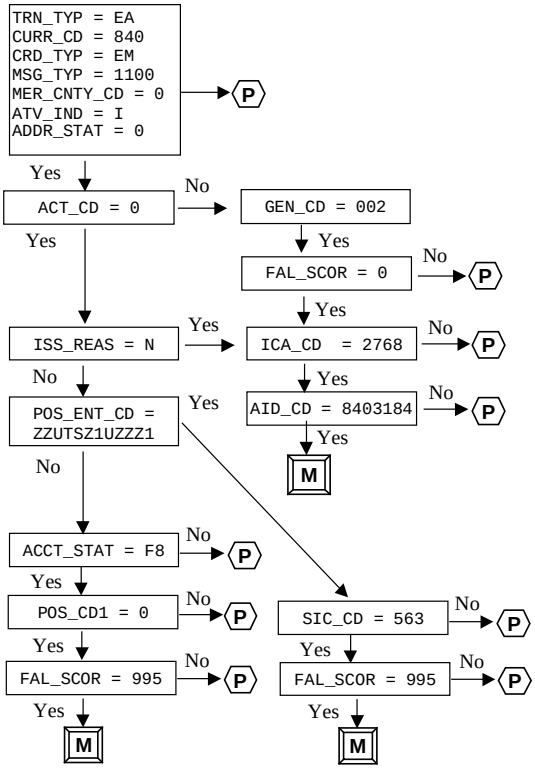

Fig. 4 *The binary decision tree for the rules of Table 3*

2.4 The mining algorithm

Now we want to present shortly the algorithm used.

- Perform the preprocessing on the data described in section 1.3.
- Perform a data base normalization: Encode all text data as binary numbers following the remarks in section 1.3.
- Now, generalize the transactions to association rules according to the following algorithm, noted in pseudo code:

```
ruleDist:=0; ROWLEN:=27; minShare:=0.02; min-
Conf:=0.1;
CurrList:=AllFraudData; NewList:= empty;
WHILE CurrList.length>0 AND ruleDist<ROWLEN DO
FOR j:=1 TO CurrList.length DO
  FOR k:=j+1 TO CurrList.length DO
      Generalize (rule[j],rule[k])
  ENDFOR
ENDFOR
ENDWHILE

IF NewList.length = 0
   THEN ruleDist := ruleDist+1
   ELSE ruleDist := 0
ENDIF

Delete marked rules in CurrList;
copy CurrList to the end of NewList;
CurrList := NewList; WriteOut(CurrList);
ENDWHILE
```

The algorithm scans all the existing rules and compares them with the other rules. When they have sufficient confidence and share they are stored in a list. If the second rule has no sufficient confidence but covers an important part of the misuse data, the subtree of the rule is searched in order to find a version of the rule which has sufficient confidence. Once found, the new rule is also stored.

All rules which have been generalized are marked and deleted for the next level. Marking and deleting does not destroy information because the generalized rule still covers all the misuse data with a sufficient level of confidence.

The heart of the algorithm is the generalization. Here, the procedure `merge` generates the new, abstracted rule by taking a copy of a rule and replacing all features of one rule which are different to the corresponding ones in the other rule by wildcard symbols *.

```
Generalize (rule1,rule2)::=
IF distance(rule1,rule2) ≠ ruleDist THEN RETURN
CurrRule := merge(rule1,rule2)
IF CurrRule NOT IN CurrList or NewList
  THEN
    CurrRule.share:= share(CurrRule)
    CurrRule.conf := conf(CurrRule)
    IF CurrRule.conf > minConf
     THEN insert CurrRule IN NewList
        mark(rule1); mark(rule2)
     ELSE
       IF CurrRule.share>minShare THEN
          REPEAT
               NewRule := nextInSubTree(rule2)
               Generalize(rule1,NewRule)
          UNTIL EndOfSubtree
             OR conf(CurrRule)> minConf;
          IF CurrRule.conf < minConf THEN
             REPEAT
               NewRule := nextInSubTree(rule1)
               Generalize(NewRule,rule2)
             UNTIL EndOfSubtree
                OR conf(CurrRule)> minConf;
          ENDIF
        ENDIF
    ENDIF
ENDIF
```

The run time complexity of the algorithm is determined by the data base of N legal transactions and K fraud ones. On the first generalization stage, the number of comparisons is quadratic in K Since we have at most $K_1=(K-1)^2/2$ new rules, the next generalization stage have to compare all rules in the set of new rules additionally with the set of old ones to produce a new rule set of level two. Thus, in the worst case the number of rules grow exponentially in the generalization level. In reality this is not the case; with increasing level the total number of rules drop sharply down to zero, see Fig. 5. The average run time complexity for rule comparison is therefore dominated by the $O(K^2)$ basic comparisons.

For the necessary computation of the share and confidence for each generated rule all fraud and legal transactions have to be scanned once. Therefore, we have about $O(K\cdot(K+N))$ computation operations.

In conclusion, the algorithm performs approximately

quadratic in K and linear in N.

2.5 Generalization and feature variance

The generalization of two rules into one rule by replacing one or several features by wildcards may introduce such an error that the resulting rule has a very low diagnostic power. This is caused by the fact that the generalization of the two rules, the wildcard, implies in most cases more feature values than two. A rule generalized like this will react not only on all transactions which were detected by the first and the second rule, but on other ones which might be erroneous.

What can we do to overcome such a problem? For the analog values we know that a high variance indicates many deviations of the mean, i.e. many possible values. For the symbolic feature values we might use the following strategy, based on the probabilities. Since we are interested in fraud rules which are different to legal transactions, we might compare the statistics of the features of the fraud data with those of the legal ones using as statistic measure the entropy of the feature. When there are no preferred feature values in fraud transactions, all symbolic values of that feature have the same occurrence probability and the entropy will be high, whereas when some values are preferred, the entropy becomes low. Thus, a high symbolic "variance" will be reflected by a high entropy; a high difference in entropy between fraud data and legal data will indicate an interesting feature. This can be stated in a table:

Legal transaction /fraud transaction	Small entropy	Big entropy
Small entropy	Preferred standard values	Preferred Fraud values
Big entropy	Preferred Legal values	No preferred values

Table 5 *The feature entropy decision table*

In the two cases where some values are preferred, i.e. the entropy difference is too high, the generalization process have to remember the feature values. Instead of a general wildcard which means the set of all possible values of this feature, only the set of the feature values of the generalized transactions is used as generalization at this place excluding all transactions with other feature values. Thus, the case where the presence of a special feature value indicates fraud (small fraud entropy) is also covered like the case where the absence of a feature value always present in legal transactions (small legal entropy) means fraud. Examples are the features MSG_TYP and TRN_TYP for small entropy in fraud transaction and the feature ICA_CD for small legal entropy.

2.6 Results

It should be noted that the mining algorithm still has a high runtime complexity. Therefore, we used only the 5,850 fraud data and 30,000 of the legal transactions. The resulting values for the confidence were compared to the whole set of transactions. Interestingly, large differences (of up to 1000%!) between the results for the sample data and for the whole data base were observed. For a second set of 30,000 differently chosen transactions we observed the same phenomena. Only the merge of the two sets showed similar values both for this training set and the test set of all transactions.

In the following Fig. 5 the performance of the rule diagnosis is shown as function of the generalization level.

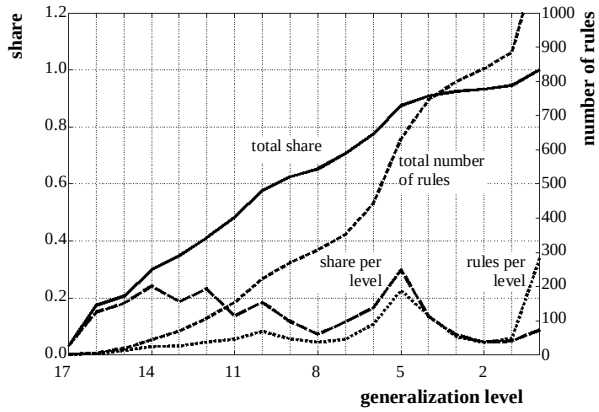

Fig. 5 *The share of the rule based diagnosis*

For each generalization level, i.e. for each number of wildcards, a set of active, non-generalized rules exists. They are denoted as "rules per level". Each set detects a certain part of the fraud, measured as "share per level". We can see that the main part of the share and the rules are obtained for level 5 and above.

While the share per generalized rule is higher than the share of the two rules itself, the number of rules drop sharply with increasing generalization level decreasing also the total share.

The performance of the fraud rule based diagnostic machine of level *n* can be obtained by taking all rule sets of level *n* and higher and measure how many fraud attempts they diagnose and how much share they have, see Fig. 5.

The confidence as a function of the generalization level is shown in Fig. 6 .

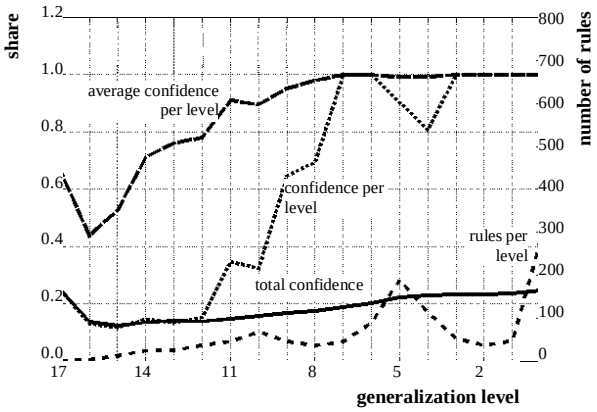

Fig. 6 *The confidence of the rule levels*

We know that a generalization can not increase the confidence, but only decrease it or be constant Nevertheless, Certainly, the more rules we take the better we perform, but, the less general the rules are the more the performance will depend on statistical variations of the fraud data. Considering this trade-off, we might plot the number of rules versus the percentage of fraud detection for the 60,000 legal transactions which is shown in Fig. 7 and try to take a compromise between high share, high confidence and low generality.

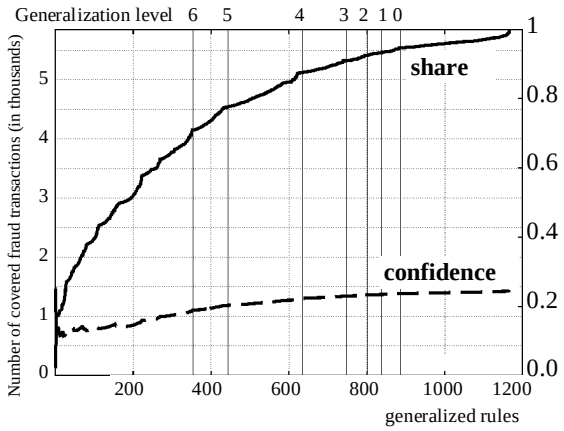

Fig. 7 *The confidence and the share as function of the rules*

We can see that a small set of rules, e.g. all rules with level 5 or higher, offer a good share of ca. 80% with the double confidence (20%) in fraud diagnosis as demanded.

If we take all the 747 rules from generalization level 4 up to level 17 we obtain a moderate confidence for the fraud detection on the set of all transactions, see Table 6.

#rules	% correct diagnosis			confidence
	legal	**fraud**	total	%
747	99.73	**90.91**	99.64 (99.72)	25.15 (**25.2**)
510	99.97	**83.08**	99.79 (99.96)	75.17 (**80.59**)
0	*99.9*	***0.0***	*99.9*	*0.0*

Table 6 *Fraud detection vs. confidence*

Here, the total correct diagnosis is computed by the basic proportion (1.1) and the confidence by

$$\text{conf (fraud diagn.)} = \left(1+\frac{N_l}{N_f}\frac{1-p_l}{p_f}\right)^{-1}$$

with the number of legal data N_l, the number of fraud data N_f, and the probabilities $p_l \equiv$ P(legal) and $p_f \equiv$ P(fraud).

However, when we select only those rules which also preserve their confidence sufficiently on the whole transaction set, we obtain 510 rules. Certainly, with less rules the fraud diagnosis probability decreases slightly, but, as we see in Table 6, the confidence in the diagnosis is dramatically increased up to 75 % due to the high proportion of legal data which are less misclassified. This is also true when we use the real proportion for legal vs. misuse transactions of 1000:1 which are shown in round brackets in Table 6. The total diagnosis performance is even better than the constant, "stupid" diagnosis mentioned before and noted in the last table row.

In conclusion, using the rule generalization mechanism described above we arrive by 25 % of all alarms to avoid the fraud up to 91% which means a saving of 9 million $ per year!

Can this success be increased? Certainly, we still have a too high degree of false alarms which should be decreased by additional means. One of them is to include the information of the analog data part of the transactions.

3 Mining the analog data

Each transaction is characterized by symbolic and analog data. So far we have only used the symbolic part of the transactions. Does the analog part containing transaction time, credit amount etc. provide any useful information? Will it be possible to enhance the fraud diagnosis? Let us first consider these questions for separate transactions and then for a sequence of transactions of one account.

3.1 Diagnosing the data of one transaction

The problem of fraud diagnosis can be seen as separating two kinds or classes of events: the good and the bad transactions. Our problem is indeed a classification problem. One major approach for dynamic classification with demand driven classification boundaries is the ap-

proach of *learning* the classification parameters, the classification boundaries, by an adaptive process. Learning is the domain of artificial neural networks, and we used a special model of it to perform the task.

3.1.1 The network

There are several possible network approaches for the task. For our model we used one expert neuron for each feature group (time, money, etc) and grouped the experts together to form a common vote. In Fig. 8 this architecture is shown.

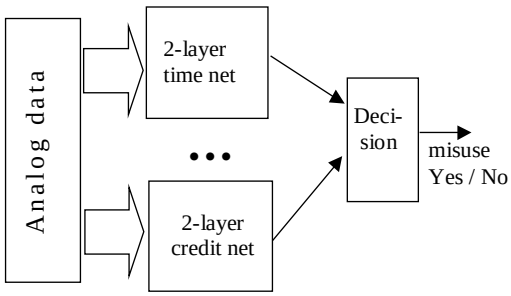

Fig. 8 *The neural network experts for analog data*

We used several networks of the Radial Basis Function (RBF) type [11], each one specialized on one topic.
One net consists of several RBF neurons which are placed such as to minimize the output error. The supervised training of each net was done sequentially. The number of units in the first (hidden) layer started with zero, adding a neuron each time the weight vector of the neuron next to the input vector has the wrong class label and the distance is bigger than the variance. Additionally, even if the next neuron has the desired class label, a new neuron is inserted if the distance is bigger than twice the variance. To avoid too many neurons, a time out mechanism deletes all neurons which are not activated within a certain "life time". The variance is updated during the training.
If the input vector is within the reach of the neuron of appropriate class, the neurons weights are updated to minimize the mean square error by a gradient descent. In Fig. 9 the pseudo code for the network construction and training is shown.

```
IF neuronVec.size()=0 THEN neuronVec.addNeuron
ENDIF
IF dist < minDist
  THEN
     IF desiredClass = Next.getNeuroClass()
       THEN
         IF 2*dist < minDist
           THEN  neuronVec.addNeuron()
           ELSE  Next.moveCenter(Input,Lrate)
                 Next.raiseWidth(Lrate)
          ENDIF
       ELSE
          neuronVec.addNeuron()
     ENDIF
  ELSE
     IF desiredClass = Next.getNeuroClass()
       THEN Next.raiseWidth(Lrate)
       ELSE Next.moveCenter(Input,-Lrate)
            Next.raiseWidth(-Lrate)
     ENDIF
ENDIF
```

Fig. 9 *The algorithm for network construction*

The second layer of each expert is a binary neuron, indicating fraud or not. It is trained by the Widrow-Hoff learning rule [11].
The RBF nets encounter a severe problem in comparison to the "sigma" net type, simple neurons of weighted sums: They can not learn the differences of the input data. For instance, if we have the transaction date and the card-creation date, the RBF neuron can be trained to be sensitive to the difference of the two (the time the card was already in use), but only to the absolute values of both. This problem made it necessary to perform preprocessing operations like difference or quotient of variables to get relative data which can be compared with data of other transactions.
By this, we finally got seven input groups and therefore seven nets with output {+1,–1} for {OK, FRAUD}.

3.1.2 The results

Because we have a very low fraud occurrence of only 0.1% the simple constant diagnosis "transactions is no fraud" will have a success rate of 99.9%. To compete with this trivial diagnosis, the task of really diagnosing a transaction is not easy to do. If we use only the analog data, all transactions patterns characterized by n symbolic and m analog features are projected from the $n+m$-dimensional space into the m-dimensional space. Generally, this results in overlapping classes and therefore in diagnostic success far worse than 99.9%. Thus, even using adaptive neural networks, we have no chance: the diagnosis of analog data can only serve as an additional information source, not as the main diagnostic criterion.

This principal idea may be underlined by an example. We trained a neural network as shown in Fig. 8 with equally distributed fraud and legal data by 100 training cycles. The test for 100 test data showed that from the 500 fraud data 464 (92.8%) were classified as "fraud" whereas this is the case for 340 (68%) legal ones. Obviously, the good fraud diagnosis property is dominated by

a high false alarm rate which occurs 1000 times more often. This situation leads to a confidence of only Conf = (5,626·0.928) / (5,626·0.928+5,626,000·0.68) = 0.14% which is unacceptable low; at least 10% is demanded. In Fig. 10 the typical situation is shown for the separation of two classes by one analog variable.

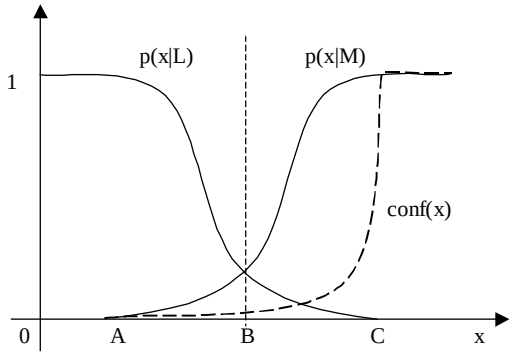

Fig. 10 *Diagnosis for overlapping classes*

Here, the two probability density functions p(x|M) for the fraud data and p(x|L) for the legal data are shown. For the best separation probability of the two clusters, the class boundary is located at point B in Fig. 10 where both densities are equal. For our two goals of high fraud detection success and high confidence in the detection we encounter a trade-off: If we choose the boundary at point A we get a high fraud discover probability and a low confidence (high false alarm rate) whereas for a high confidence we have to choose the class decision boundary at point C with a smaller fraud discovery success. Note that due to the high proportion of legal data in the data set the confidence drops sharply when, changing the class boundary, legal transactions are diagnosed as fraud.
All training procedures which settle a classification boundary have to reflect this basic property.

Now, let us diagnose one transaction by the means of the neural network. For that purpose, we used the neural expert system shown in Fig. 8 and trained it with our fraud data. We used 300 transactions for training and analyzed the state of the whole network afterwards by presenting 250 legal and 250 fraud data. The proportion of legal to fraud data for training was changed, causing different diagnosing behavior. The results are shown in Table 7.

proportion	**correct diagnosis %**			**faulty diagnosis %**		**confidence %**
	total	legal	fraud	legal	fraud	
2:1	78.8	95.2	62.4	4.8	37.6	1.3
3:1	78.2	98.4	58.4	1.6	41.6	3.5
4:1	58.2	99.6	16.8	0.4	83.2	4.0
5:1	52.5	99.2	6.0	0.8	94.0	0.7
10:1	50.0	100	0	0	100	100

Table 7 *Shifting the class boundary*

As we can see, by augmenting the number of legal transactions in the training the class boundary shifts towards point C in Fig. 10. Here, the confidence is high, but the fraud discovery becomes zero.

3.2 Diagnosing sequences of data

One of the most interesting topics is the question whether the sequence of transactions of one account can be used to detect fraud transactions. Here, two ideas evolve. First, there can be typical fraud sequences, for instance the behavior of a thief after copying or picking the credit card. Second, there can be a "typical" behavior of the user (a "user profile") which it does not correspond to the actual transaction sequence may indicate a credit card misuse. Can we detect one of these cases by appropriate means?

3.2.1 Symbolic user profile

To answer this question, we ordered our data in time for each account. This revealed that most of the accounts had less than 30 transactions as a sequence which is far too small for good statistics.
Additionally, the analysis of the symbolic part of the transaction data is demotivated by the fact that the interesting features such as the merchant ID can take many different values. Finding probable temporal sequences of the symbolic states means learning the transaction probability between many states, e.g. 100,000. This needs not only a vast amount of storage for the state transition matrix of all possible states for each account, but also much more transaction data to fill the matrix which we do not have. Thus, a markov model for the state transitions is out of reach for our task. Instead, we implemented a "preference counter": For a time window of several transactions of one account the number of equal values of a symbolic feature is counted. Strong preferences of symbolic values ("habitudes") are reflected by this variable. For a set of 1000 sequences of length 3 (triples) composed by 50 fraud associated triples and 50 legal transaction triples the probability for a fraud (or legal) transaction to be recognized when a triple of equal symbolic values are shown in the columns of Table 8 for each feature.

feature	Legal TA		Fraud TA	
	probability	#	probability	#
TRN-NBR	0.48	24	0.98	49
CURR-CD	0.58	29	1.00	50
POS-ENT-CD	0.38	19	0.60	30
ICA-CD	0.72	36	0.52	26
AID-CD	0.42	21	0.52	26
SIC-CD	0.16	8	0.50	25
ACT-CD	0.96	48	0.92	46
MSG-TYP	0.56	28	1.00	50
MER-ID	0.02	1	0.32	16

Table 8 *The occurrence of symbolic values in a sequence of length 3*

Since all fraud and legal transactions are rarely recognized by exclusive triples of equal feature values, the confidence of fraud detection based on these probabilities is very low, shown in the last two columns. Here again, the dominance of legal transactions impedes a proper recognition. Nevertheless, they constitute an additional source of diagnostic information.

3.2.2 Analog user profile

For the sequences of analog values we used a neural network, similar to the one of section 3.1.1. As input, we considered *n* inputs for each analog feature, corresponding to the *n* values of a window of *n* time steps. We consider the time distance between the transactions by including the time difference as additional analog input variable. By this approach, we hope to discover fraud behavior patterns like many high transaction amounts in a short time interval. For processing, the analog features are divided by the mean values. Additionally, the amounts are divided by the time difference to get the cash flow of the account as input.

3.2.3 Results of the combined approach

The user profile diagnostic network was designed as a combination of the symbolic and the analog subnets by a threshold neuron. The network was trained by a mixture of 200 fraud transactions and 200 legal ones. The output was activated when the sum of the input superceded the threshold. In turn, when a sequence occurred each input line was activated and weighted by their fraud probability. Thus, the activity as the sum of the marginal probabilities reflected the probability conditions only roughly. The probability of a correct diagnosis evolved to 0.7 in the training process. The validation on a test set of 125 fraud transactions and 125 legal ones resulted in the slightly smaller probability of 0.65. Certainly, these results depend on the sequence length *n*. In Table 9, this is shown for different time window length.

n	correct diagnosis			confidence
	legal	fraud	total	
3	0.94	0.37	0.65	0.57 %
4	0.73	0.43	0.58	0.16 %
5	0.98	0.26	0.62	1.58 %
6	0.61	0.58	0.59	0.15 %

Table 9 *Classification success of profile data*

Here, the diagnosis heavily depends on the different influences. For short time windows, the diagnostic influence of the tuples of symbolic data is bigger than the influence of the analog data. Increasing the window length lowers the number of equal features and therefore its diagnostic influence until it reaches zero.
It should be noted that, due to the small proportion 1:1000 of fraud data the resulting confidence is determined again by the diagnostic success of the legal data.

4 Combining symbolic and analog information

In the previous sections we encountered the fact that the analog data of neither one nor several transactions of an account can serve as a satisfying criterion for fraud diagnosis. Therefore, we have to combine the diagnostic information of the rule-based association system of section 2 with the expert information of section 3.

4.1 A hybrid expert architecture

There are several possible architectures for an hybrid diagnostic system. In Fig. 11 and in Fig. 12 two versions are shown.

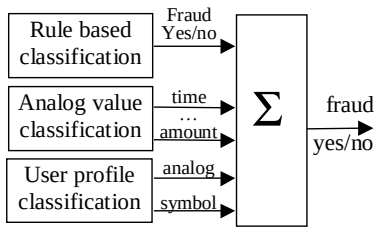

Fig. 11 *A parallel diagnosis*

In Fig. 11, the diagnostic results are used in parallel. The diagnostic influence of all the experts are initially the same and converge by training in the limit to their appropriate value. In all situations, decisions based on the analog data can override the rule based expert.

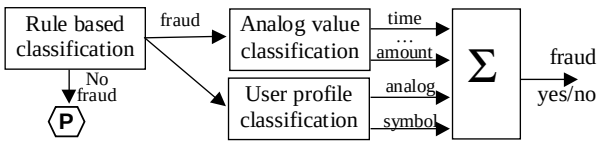

Fig. 12 *A sequential diagnosis*

The second architecture in Fig. 12 tries to avoid this situation. From the beginning, the rule based system dominates. Its diagnosis can be only corrected in the case of a false fraud assignment. The sequential architecture avoids wrong fraud assignments by a kind of logical AND decision. Obviously, this will optimize the confidence in fraud decisions, not the probability of fraud detection.

4.2 Results

For a special test subset of 1000 transactions which are selected from the set of all transactions with multiple transactions per account we computed the rule-based diagnosis, the analog diagnosis probabilities and the user profile diagnosis probabilities on different test sets. In Table 10, the results are compared to the results of Table 6.

Correct diagnosis	Legal TA	Fraud TA	total	confidence %
Rule based	1.00	0.802	0.901	100.00
	0.9998	**0.8308**	0.9153	80.60
1:1000			0.99963	**80.59**
Analog	0.952	0.754	0.853	1.550
	0.924	**0.71**	0.817	0.931
1:1000			0.92379	**0.926**
Profile	0.934	**0.436**	0.685	**0.66**

Table 10 *Correct diagnosis of the rule based, the analog data based and the profile based system for 1000 transactions and 11700 transactions using 1:1 proportions. In brackets the computed values for a 1:1000 proportion are given.*

The rules are selected according to a share of 80% giving a set of 510 rules, approximately 10% of the number of fraud transactions. In Table 10, only the separate diagnostic results on 1000 transactions are shown. For the profile data only a small amount of account history data were available for training and analysis. Thus, all training for the profiles was restricted to 300 samples, i.e. 150 legal and 150 fraud data.
When we combine all diagnostic modules into a parallel network of experts (see Fig. 11), we can increase the fraud diagnosis benefits. This is shown in Table 11.

Diagnostic method	P(correct fraud diagn.)		Confidence %	
Data set size	1000	11,700	1000	11,700
Rules + analog	.856	.879	100	1.048
Rules + profile	.802	-	100	-
Analog + profile	.752	-	3.04	-
All, Training 1:1	**.848**	**-**	**12.38**	**-**
All, Training 1:2	**.812**	**-**	**100**	**-**
All, Training 1:3	**.796**	**-**	**100**	**-**

Table 11 *Comparison of different parallel diagnostic expert systems on two sets of data*

Due to the small sample size of profiling data there was no profiling diagnosis available for the sample size of 11,700.
Certainly, the training 1:1 with equal proportional fraud and legal data does not reflect the real proportions well. Changing the training sample properties to 1:2 or 1:3 (instead of 1:1000) for the sample size of 1000 transactions, we get different diagnostic probabilities actions, see Table 11. As we already discussed before in section 3, the classification probabilities decrease slightly when changing the training proportions from 1:1 to 1:2 and finally to 1:3, but the confidence increases dramatically from 12% up to 100% as shown in Fig. 10.

In the table above we notice that the rule based system dominates by its unmatched diagnostic power. Using the additional diagnostic modules results in a smaller fraud diagnosis probability and less confidence. So, instead of augmenting the diagnostic abilities of the rule based system the analog and profile information spoil the diagnostic process. How can we overcome this? If we use the rule based system first and let the other experts diagnose its output, the result should be better.

Therefore, the sequential model of Fig. 12 promises a better fraud detection and additional confidence. Certainly, this kind of system does not decrease the probability for the first stage to classify fraud data as "legal", but it should increase the probability for the diagnosis "fraud" to be correct and therefore increase the confidence and decrease the number of false alarms.

Is this true? Let us regard the results for the sequential combination of rules R, analog experts A and profile expert P, listed in Table 12.

diagnostic method	P(correct fraud diagn.)		confidence %	
data set size	1000	11,700	1000	11,700
510 R+A	0.69	0.75	100.0	81.5
747 R+A	0.80	0.82	28.6	49.0
837 R+A	0.82	0.84	29.0	62.1
510 R+A+P	0.85	–	100.0	–
747 R+A+P	0.87	–	100.0	–
837 R+A+P	**0.95**	–	**100.0**	–

Table 12 *Comparing the performance of different sequential diagnostic expert systems on two sets of data*

For a training of 1:1 we measured the performance of the sequential scheme. We can observe that the combined power of rule and analog expert does not increase the amount of detected fraud, but detect it more securely with 100% confidence just as we expected. Nevertheless, the probability of fraud detection is too low compared with the rule based system only. Therefore, we tested the strategy of adding additional rules even with lower confidence. As we can see in Table 12, more rules give more alarms which, filtered by the analog experts, increase the probability of fraud detection. The confidence values should be taken not literally but as a hint for the performance of the scheme: The drop of 100% confidence to 81.5% in the first row of the table is caused by just one erroneously classified legal transaction.

In summary, by an automatically generated rule system we managed to increase the correct diagnosis of 99.9% to 99.95 %. Including also the analog and profiling information we increased this to 99.995%.
As most important topic, the detection of fraud is increased from 0% to 80% by using the generalized rule system. Adding the analog information of the transactions by training additional analog and profile expert modules we succeeded to drive the fraud detection probability up to 95% with the confidence of nearly 100%. Thus, our system promises to save 95% of the fraud, i.e. 9.5 million dollars per year.
.

5 Discussion

In this contribution we developed concepts for the statistic-based credit card fraud diagnosis. We showed that this task has to be based on the very special diagnostic situation imposed by the very small proportion of fraud data of 1:1000.

We showed that a naive association memory approach for the symbolic features of the transaction data has severe implementation problems which can be overcome by treating all transactions as generalized rules of level 0. By algorithmically generalizing these rules we obtain higher levels of diagnostic rules. The high intrinsic run-time complexity of this process can not be applied to the whole data base. Instead, representative sample sizes had been chosen and the partial results have been validated on the whole data set.
Additionally, the analog transaction data can be analyzed by specially designed neural networks. However, the good results produce too many false alarms giving bad diagnostic confidence. Also the user habits (user profile) can produce valid fraud information, but the associated confidence is not sufficient.
Finally, we discussed the concept of combining all the diagnostic information into one adaptive multi-expert system. This concept can improve both the confidence and the diagnostic probability.

In summary, combining rule-based information and adaptive classification methods yields good results, even in the case of the very difficult analysis of credit card fraud detection. Additional work is necessary to design an online learning and diagnostic system based on the results.

References

[1] R. Agrawal, H.Mannila, R. Srikant, H. Toivonen, A.I. Verkamo: *Fast Discovery of Association Rules*. In: U. Fayyad, G. Piatesky-Shapiro, P. Smyth, R. Uthurusamy (eds.): Advances in Knowledge Discovery and Data Mining. Menlo Park, AAAI/MIT Press 1996

[2] R. Agrawal, R. Srikant: *Fast Algorithms for mining association rules*. Proceedings of the VLDB Conference, Santiago, Chile, 1994

[3] P. Barson, S. Field, N. Davey, G. McAskie, R. Frank*: The Detection of Fraud in Mobile Phone Networks;* Neural Network World 6, 4, pp. 477-484 (1996)

[4] L. Braimann, J.H. Friedman, R.A. Olshen, C.J. Stone: *Classification and Regression Trees*; Wadsworth Int. Group, Belmont, CA (1984)

[5] T. Kohonen: *Correlation Matrix Memories*; IEEE Transactions on Computers, Vol C21, pp.353-359, (1972)

[6] H. Mannila, H. Toivonen, I. Verkamo: Efficient Algorithms for Discovering Association Rules; AAAI Workshop on Knowledge Discovery in Databases, pp.181-192, Seattle, Washington 1994

[7] S. Ghosh, D.L. Reilly: *Credit Card Fraud Detection with a Neural-Network*; Proc. 27th Annual Hawaii Int. Conf. on System Science, IEEE Comp. Soc. Press, Vol.3, pp.621-630 (1994)

[8] R.J. Hildermann, C.L. Carter, H.J. Hamilton, N. Cercone: *Mining Association Rules from Market Basket Data using Share Measures and Characterized Itemsets;* Int. J. of AI tools vol.7, No.2, pp.189-220, 1998

[9] Y. Moreau, H. Verrelst, J. Vandwalle: *Detection of Mobile Phone Fraud using Supervised Neural Networks: A First Prototype;* Proc. ICANN ´97, Lecture notes on computer science LNCS 1327, Springer Verlag 1997

[10] G. Palm: *On Associative Memory*; Biolog. Cybernetics, Vol. 36, pp. 19-31 (1980)

[11] S. Haykin: *Neural networks* - a comprehensive foundation, MacMillan, New York 1994

[12] R. Srikant, R. Agrawal : *Mining generalized association rules.* Proc. VLDB Conference, Zurich, Switzerland, 1995

Appendix A

Share, Confidence and Generalization

Theorem 1
For a generalization of *n* rules the share S_{res} of the resulting rule has as least the maximum of the share of all the rules

$$S_{res} \geq \max \{S_1, ..., S_n\}$$

Proof
Let us consider the case of two rules. Their share is by eq. (2.5)

$$S_i = a_i/b$$

with a_i = #fraud transactions covered by rule i
and b = #fraud transactions

The new rule covers all the transactions of the base rules

$$a_{res} \geq \max \{a_1, ..., a_n\}$$

such that we get

$$S_{res} = a_{res}/b \geq \max \{a_1/b, ..., a_n/b\} = \max \{S_1, ..., S_n\}$$

Remarks

In the best case, the new rule covers only non-intersecting transaction sets

$$S_{res} = \sum_{i=1}^{n} S_i$$

When several, but not all values of a feature are replaced by one wildcard, the share can become even bigger .

$$S_{res} \geq \sum_{i=1}^{n} S_i$$

Theorem 2
For a generalization of *n* rules the confidence C_{res} of the resulting rule has as at most the maximum of the confidences all the rules

$$C_{res} \leq \max \{C_1, ..., C_n\}$$

Proof
Let us consider the case of two rules. Their confidences are after (2.6)

$$C_1 = \frac{a_1}{b_1} \quad \text{and} \quad C_2 = \frac{a_2}{b_2} = \frac{\alpha a_1}{\beta b_1}$$

with $\alpha,\beta > 0$ and $C_{res} = \frac{a_1 + a_2}{b_1 + b_2}$.

Let us assume $C_1 \geq C_2$. Then, we have $\alpha \leq \beta$ and therefore

$$\alpha \leq \beta \Leftrightarrow 1+\alpha \leq 1+\beta \Leftrightarrow \frac{1+\alpha}{1+\beta} \leq 1$$

$$\Leftrightarrow \frac{(1+\alpha)a}{(1+\beta)b} \leq C_1 \Leftrightarrow C_{res} = \frac{a_1 + a_2}{b_1 + b_2} \leq C_1$$

This can be easily generalized to three rules by generalizing the resulting rule with the third rule, the result with the forth rule and so on. Therefore, this is also valid for *n* rules and the resulting confidence is lower or equal than the maximum of all the rules.

For the share of a generalized rule, we have a complementary result.