Minimizing Credit Card False Alerts Using Genetic Algorithm

M. Nandhini[1], P.Shanthi Bala[2]

ABSTRACT

This paper describes about the detection of credit card fraud using genetic algorithm. It helps to minimize the number of false alerts. When a card is copied or stolen or lost or captured by fraudsters, it is misused till the available limit is minimal. Hence, this approach focus on minimizing the total available limit on cards subject to fraud rather than the number of correctly classified transactions. It aims in minimizing the false alerts using genetic algorithm where a set of interval valued parameters are optimized.

Keywords: — Genetic algorithm, Credit card, Fraud Detection

1. INTRODUCTION

Fraud is considered to be an illegal activity taking place in an operational system and is concerned to be a major growing problem in the credit card industry. The credit card fraud is essentially of two types: application and behavioural fraud. We deal with behavioural fraud and its of four types: mail theft, stolen/lost card, counterfeit card and card holder not present fraud. The fraud occurred may be of any type but the fraudsters usually used them until its available limit is depleted. Several data mining algorithms [1] have been proposed like decision trees, computational intelligence and artificial neural networks, web-services based collaborative scheme, etc. The Traditional detection method mainly depends on database system and the education of customers, which usually are delayed, inaccurate and not in-time. After that methods based on decision trees and regression analysis are widely used to detect fraud by credit rate for cardholders and credit card transaction. In the existing system, the fraud is detected by capturing the IP address.

Over the years, along with the evolution of fraud detection methods, perpetrators of fraud have also been evolving their fraud practices to avoid detection. So the credit card fraud detection methods need constant innovation. In this study, we use the concept of genetic algorithm which produces an optimum solution as time progresses.

All the fake cards are not going to be fake there may be legitimate cards also and all the legitimate cards may have some fake cards. So it is necessary to minimize the false transactions. We use genetic algorithm not only to detect the fraudulent transactions but also to minimize the false alert. In this system, the fraud is detected by keeping an account of the customers' behaviour. The dataset consists of several fields which state the customer behaviour. The fields are credit card ID, pin number, overdraft, book balance, amount of transactions per day, etc., We also derive new formula for fitness function for producing best solution and in the initial population instead of using a single population. In addition, we use multi-population which helps in producing an optimized solution.

[1]*Dept. of Computer Science Pondicherry University Puducherry-14*
mnandhini2005@yahoo.com
[2]*Dept. of Computer Science Pondicherry University, Puducherry14*
pk_shanthi11@yahoo.co.in

II. Problem Defintion

During the credit card transaction, the fraud is detected and the number of false alert is being minimized by using genetic algorithm. Instead of maximizing the numbers of correctly classified transactions defined an objective function where the misclassification costs are variable.

The algorithm begins with multi-population of randomly generated chromosomes. These chromosomes undergo the operations of selection, crossover and mutation. Crossover combines the information from two parent chromosomes to produce new individuals, exploiting the best of the current generation, while mutation or randomly hanging some of the parameters allows exploration into other regions of the solution space. Natural selection via a problem specific cost function assures that only the best fit chromosomes remain in the population to mate and produce the next generation.

Upon iteration, the genetic algorithm converges to a global solution.

III. Literature Survey

Fraud detection has been usually seen as a data mining problem where the objective is to correctly classify the transactions as legitimate or fraudulent. For classification problems many performance measures are defined most of which are related with correct number of cases classified correctly[10].

Hamdi Ozcelik, Ekrem Duman, Mine Isik, Tugba Cevik[2] tried to improve the performance of an existing solution by defining a new objective function where the misclassification costs are variable and thus, correct classification of some transactions are more important than correctly classifying the others. Ekrem Duman, M. Hamdi Ozcelik [3] suggested a novel combination of the two well known meta-heuristic approaches, namely the genetic algorithms and the scatter search and improve the credit card fraud detection solution currently being used in a bank.

A more appropriate measure is needed due to the inherent structure of credit card transactions. When a card is copied or stolen or lost and captured by fraudsters it is usually used until its available limit is depleted. Thus, rather than the number of correctly classified transactions, a solution which minimizes the total available limit on cards subject to fraud is more prominent [4].

Since the fraud detection problem has mostly been defined as a classification problem, in addition to some statistical approaches many data mining algorithms have been proposed to solve it. Among these, decision trees and artificial neural networks are the most popular ones. The study of Bolton and Hand provides a good summary of literature on fraud detection problems.

However, when the problem is approached as a classification problem with variable misclassification costs as discussed above, the classical data mining algorithms are not directly applicable; either some modifications should be made on them or new algorithms developed specifically for this purpose are needed. An alternative approach could be trying to make use of general purpose meta heuristic approaches like genetic algorithms.

IV. Evolution Process

Genetic algorithms are evolutionary algorithms which aim at obtaining better solutions as time progresses shown in Fig.1.Since their first introduction by Holland, they have been

successfully applied to many problem domains from astronomy to sports, from optimization to computer science, etc. They have also been used in data mining mainly for variable selection and are mostly coupled with other data mining algorithms. In this paper, we tried to solve our classification problem by using only a genetic algorithm solution.

In a genetic algorithm, a population of strings which encode candidate solutions to an optimization problem, evolves toward better solutions. Traditionally, solutions are represented in binary as strings of 0s and 1s, but other encodings are also possible. The evolution usually starts from a population of randomly generated individuals and happens in generations. In each generation, the fitness of every individual in the population is evaluated, multiple individuals are stochastically selected from the current population (based on their fitness), and modified (recombined and possibly randomly mutated) to form a new population. The new population is then used in the next iteration of the algorithm. Commonly, the algorithm terminates when either a maximum number of generations has been produced, or a satisfactory fitness level has been reached for the population. If the algorithm has terminated due to a maximum number of generations, a satisfactory solution may or may not have been reached.

Once the genetic representation of individuals and the fitness function defined, GA proceeds to initialize a population of solutions randomly, then improve it through repetitive application of mutation, crossover, inversion and selection operators.

A. Tournament Selection

Tournament selection has been used in this as it selects optimal individuals from diverse groups. It selects the individuals from the current population uniformly at random, forms a tournament and the best individual of a group wins the tournament and is put into the mating pool for recombination. This process is repeated the number of times necessary to achieve the desired size of intermediate population. The tournament size controls the selection strength. The larger the tournament size, the stronger is the selection process.

B. Elitist Selection

In order to make sure that the best individuals of the solution are passed to further generations, and should not be lost in random selection, this selection operator is used. So we used a few best chromosomes from each generation, based on the higher fitness value and are passed to the next generation of population.

C. Reproduction

To generate a second generation population of solutions from those selected through genetic operators: crossover (also called recombination), and/or mutation. For each new solution to be produced, a pair of "parent" solutions is selected for breeding from the pool selected previously. By producing a "child" solution using the above methods of crossover and mutation, a new solution is created which typically shares many of the characteristics of its "parents". New parents are selected for each new child, and the process continues until a new population of solutions of appropriate size is generated. Although reproduction methods that are based on the use of two parents are more "biology inspired", some research suggests more than two "parents" are better to be used to reproduce a good quality chromosome. These processes ultimately result in the next generation population of chromosomes that is

different from the initial generation. Generally the average fitness will have increased by this procedure for the population, since only the best organisms from the first generation are selected for breeding, along with a small proportion of less fit solutions, for reasons already mentioned above.

Although Crossover and Mutation are known as the main genetic operators, it is possible to use other operators such as regrouping, colonization-extinction, or migration in genetic algorithms.

D. One point Crossover

A crossover operator that randomly selects a crossover point within a chromosome then interchanges the two parent chromosomes at this point to produce two new offspring

E. Mutation

Mutation is a genetic operator that alters one ore more gene values in a chromosome from its initial state. This can result in entirely new gene values being added to the gene pool. With these new gene values, the genetic algorithm may be able to arrive at better solution than was previously possible. Mutation is an important part of the genetic search as help helps to prevent the population from stagnating at any local optima. Mutation occurs during evolution according to a user-definable mutation probability. This probability should usually be set fairly low (0.01 is a good first choice). If it is set to high, the search will turn into a primitive random search.

They are different types of mutation like flip-bit, boundary, Gaussian, uniform and non-uniform. Of which we are using uniform mutation.

F. Uniform mutation

A mutation operator that replaces the value of the chosen gene with a uniform random value selected between the user- specified upper and lower bounds for that gene. This mutation operator can only be used for integer and float genes.

G. Termination

This generational process is repeated until a termination condition has been reached. Common terminating conditions are:

- A solution is found that satisfies minimum criteria
- Fixed number of generations reached
- Allocated budget (computation time/money) reached
- The highest ranking solution's fitness is reaching or has reached a plateau such that successive iterations no longer produce better results.
- Manual inspection.
- Combinations of the above.

Where, n is the number of individuals in the population; χ is the fraction of the population to be replaced by crossover in each iteration; and μ is the mutation rate.

H. Fields used in detecting credit card fraud[5][6]:

The following fields are:

Credit card ID.
Pin number.
Current book balance.
Current usage.
Average book balance.
Overdraft.
Credit card age.
Card used today.
Location.
Amount of transaction.

```
GA(n, χ, μ)
// Initialise generation 0:
k := 0;
Pk := a population of n randomly-generated individuals;
// Evaluate Pk:
Compute fitness(i) for each i " Pk;
do
{ // Create generation k + 1: // 1. Copy:
Select (1 " χ) × n members of Pk and insert into Pk+1;
// 2. Crossover:
Select χ × n members of Pk; pair them up; produce offspring; insert
the offspring into Pk+1; // 3. Mutate:
Select μ × n members of Pk+1; invert a randomly-selected bit in each;
// Evaluate Pk+1:
Compute fitness(i) for each i " Pk; // Increment:
k := k + 1;
}
while fitness of fittest individual in Pk is not high enough;
return the fittest individual from Pk;
```

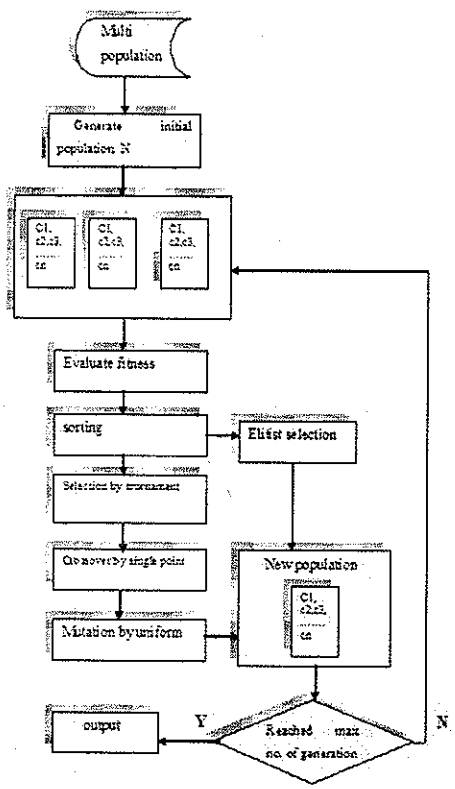

Fig. 1 Evolution Process

I. Objective Function:

Formula to calculate Critical Value

For each transaction in the dataset, the following five property critical values are calculated,

1) Based on CC usage Frequency

Ccfreq = Total number card used (CU) / CC age. If ccfreq is less than 0.2, it means this property is not applicable for fraud and critical value =ccfreq. Otherwise, it check for condition of fraud.

Fraud condition = number of time Card used Today (CUT) >(5 * ccfreq)

If true, there may chance for fraud using this property and its critical value is CUT*ccfreq.

If false,no fraud occurance and critical value = ccfreq.

```
float ccfreq
=Float.valueOf(temp[3])/Float.valueOf(temp[6]);
if(ccfreq>0.2)
{
if(Float.valueOf(temp[7])>(5*ccfreq))
                    {
res[0]=1;
res[1]=(Float.valueOf(temp[7])*ccfreq);
}
}
if(res[0]<1)
{
res[1]=(float)ccfreq;
}
```

2) Based on CC usage Location

Number of locations CC used so far (loc) obtained from dataset(loc).If loc is less than 5, it means this property is not applicable for fraud and critical value =0.01. Otherwise, it check for condition of fraud. Fraud condition = number of locations Card used Today (CUT) >(5 * loc) If true, there may chance for fraud using this property and its critical value is loc/CUT If false, no fraud occurrence and critical value =0.01

```
int loc=Integer.valueOf(temp[8]);
if((loc<= 5) && (Integer.valueOf(temp[9])>(
2 * loc)))
{
res[0]=1;
res[1]=(Float.valueOf(loc)/
Float.valueOf(temp[9]));
}
if(res[0]<1)
{
res[1]=(float)0.01;
}
```

3) Based on CC OverDraft

Number of times CC overdraft w.r.t CU occurred so far (od) can be found as, Od w.r.t CU = OD/CU.

If Od w.r.t CU is less than 0.02, it means this property is not applicable for fraud and critical value = Od w.r.t CU. Otherwise, it check for condition of fraud.

Fraud condition = check whether overdraft condition occurred today from (ODT dataset)

If true, there may chance for fraud using this property and its critical value is ODT * Od w.r.t CU. If false, no fraud occurance and critical value = Od w.r.t CU.

```
float od
=Float.valueOf(temp[5])/Float.valueOf(temp[3]
);
if(od<=0.2)
{
if(Float.valueOf(temp[10])>=1)
                {
res[0]=1;
res[1]=(Float.valueOf(temp[10])*od);
}
}
if(res[0]<1)
{
res[1]=(float)od;

}
```

4) Based on CC Book Balance

```
float bb
=Float.valueOf(temp[2])/Float.valueOf(temp[4]);
if(bb<=0.25)
{
res[0]=1;
res[1]=(Float.valueOf(2)*bb);
}
if(res[0]<1)
{
res[1]=(float)bb;
}
```

Standard Book balance can be found as, Bb = current BB / Avg. BB

If bb is less or equals than 0.25, it means this property is not applicable for fraud and critical value = bb. Otherwise, it check for condition of fraud.

If true, there may chance for fraud using this property and its critical value is currBB * bb. If false, no fraud occurance and critical value = bb.

5) Based on CC Average Daily Spending

```
float mon= Float.valueOf(temp[6])/30;
float bal= 100000 - Float.valueOf(temp[4]); float
tot = mon*bal;
float ds =tot/Float.valueOf(temp[6]);
if((10*ds)<Float.valueOf(temp[11]))
{
res[0]=1;
if(Float.valueOf(temp[11])>0)
res[1]=(Float.valueOf(temp[11])/ (10*ds));
else
res[1]=(float) 0.0;
}
if(res[0]<1)
{
res[1]=(float)0.01;
}
```

Age of CC in months can be calculated using CCage (from dataset) by, Age of cc by month = CCage/30.

Total money being spent from the available limit

(1 lakh _ 100000)

Bal = 100000 - avg BB.

So, total money spent can be found as, Tot = Age of cc by month * Bal

Total money spent on each month can be calculated as, Ds=tot* Age of cc by month

Otherwise check for condition of fraud.

Fraud condition = (10 * ds) is amount spent today (AmtT in dataset)

If true, there may chance for fraud using this property and its critical value is

AmtT/(10*ds).

If false, no fraud occurance and critical value 0.01.

V. System Architecture

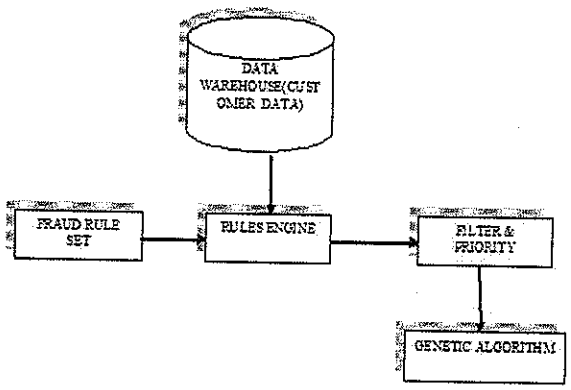

Fig.2 Overall System Design

The above architecture in Fig.2 describes the work structure of the system and use case diagram show the flow of data as in Fig.3. The customer data in the data warehouse is subjected to the rules engine which consists of the fraud rule set [3],[7]&[8].

The filter and priority module sets the priority for the data and then sends it to the genetic algorithm which performs its functions and generates the output.

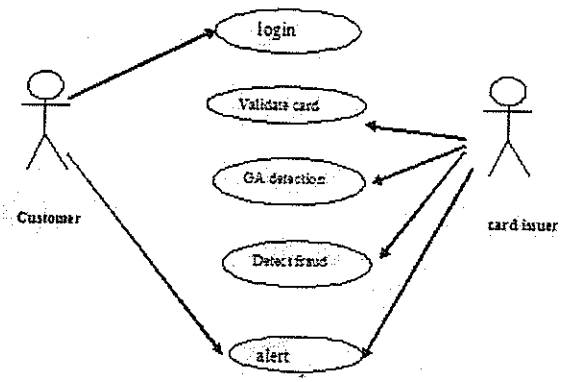

Fig.3 Use case diagram

VI. Implementation

The application is completely written Java. This enables the credit card issuers to use this application across wide variety of devices independent of the vendor of the devices. Oracle used as a back end for storing database.

The stored data and the updated information about the card owner will be taken into account when he/she is transacting and it will be verified whether its fake or legitimate. The fields used for storing the data of the card owner are the main factors. Using those fields five new formulas has been derived which plays an important role in detecting the fraud transactions.

Several fields which states the customers behaviour are used in the dataset but the initial population consists of only important fields which play a key role in identifying the fraudulent transactions are used. These fields are mentioned earlier.

The newly derived formulas are the fitness function which helps in detecting fraud transactions. After each process the newly generated population will give us the fraudulent transacted cards as an optimized value[9],[10].

Initially equal number of fake and legitimate cards was taken and the process was done but the output

was not an optimal one. Then 80% of legitimate cards were taken but the output was not an optimal one. Later only 20% of legitimate cards were taken and it produced the best result an optimal solution. The final result will be shown as the critical fraud transactions detected and ordinary fraud transactions detected.

SAMPLE OUTPUT

Critical Values Found after Limited number of Generations (sorted order)

0.8895925
1.245
1.27
1.27
1.3809904
1.3809904
1.4014348
1.5033333
1.5033333
1.5985714
1.8985714
51.8985714
2.17
2.17
3.669799
3.8700001
4.0200005
4.2309904
4.4197993
4.4700003

Critical Values of each transaction of given Data set

Account No.	Fraud Occurance	Critical Value
11111.0	0.0	0.0
11112.0	0.0	0.0
11113.0	1.0	0.14285715
11114.0	0.0	0.0
11115.0	3.0	2.1289055
11116.0	0.0	0.0
11117.0	0.0	0.0
11118.0	0.0	0.0
11119.0	0.0	0.0
11120.0	4.0	5.171741
11121.0	0.0	0.0
11122.0	1.0	0.18181819
11123.0	0.0	0.0
11124.0	1.0	0.63265306
11125.0	4.0	4.289769
11126.0	1.0	0.16666667
11127.0	1.0	0.1764706
11128.0	0.0	0.0
11129.0	0.0	0.0
11130.0	3.0	4.8449016
Value of Critic,	Monitor and	Ordinary Frauds
3.669799	1.5985714	1.3809904

Fraud Detected used Genetic Algorithm:

Critical Fraud Detected:

Credit Card with ID 11120.0 is detected as fraud with 4.0 occurance and its critical value is 5.171741

Credit Card with ID 11125.0 is detected as fraud with 4.0 occurance and its critical value is 4.289769

Credit Card with ID 11130.0 is detected as fraud with 3.0 occurance and its critical value is 4.8449016

Monitorable Fraud Detected:

Credit Card with ID 11115.0 is detected as fraud with 3.0 occurance and its critical value is 2.1289055.

VII. Conclusion

This method proves accurate in deducting fraudulent transaction and minimizing the number of false alert. Genetic algorithm is a novel one in this literature in terms of application domain. If this algorithm is applied into bank credit card fraud detection system, the probability of fraud transactions can be predicted soon after credit card

transactions. And a series of anti-fraud strategies can be adopted to prevent banks from great losses and reduce risks. The objective of the study was taken differently than the typical classification problems in that we had a variable misclassification cost. As the standard data mining algorithms does not fit well with this situation, decided to use multi population genetic algorithm to obtain an optimized parameter.

VIII. Future Enhancements

The findings obtained here may not be generalized to the global fraud detection problem. In future, some effective algorithm which can perform well for the classification problem with variable misclassification costs could be developed.

IX. References

[1] J .Han ,M. Kamber, "Data Mining :Concepts and Techniques", MorganKaufmaan Series,2000.

2] M. Hamdi Ozcelik, Ekrem Duman, Mine Isik, Tugba Cevik, "Improving a credit card fraud detection system using genetic algorithm", International Journal of Soft Computing and Engineering (IJSCE) , Vol.1, pp.436-440, 2011.

[3] Ekrem Duman, M. Hamdi Ozcelik," Detecting credit card fraud by genetic algorithm and scatter search" An International Journal Expert Systems with Applications,38(10),2011, pp.13057-13063.

[4] Clifton Phua, Vincent Lee1, Kate Smith & Ross Gayler,"A Comprehensive Survey of Data Mining-based Fraud Detection Research" Shiguo Wang in International Conference on Intelligent Computation Technology and Automation, ICICTA 2010.

[5] Wen-Fang YU, Na Wang, "Research on Credit Card Fraud Detection Model Based on Distance Sum", IEEE International Joint Conference on Artificial Intelligence,2009, pp.353-356.

[6] Abinav Srivastava, Amlan Kundu, Shamik Sural, Arun K.Majumdar "Credit card fraud detection using hidden markov model " , IEEE Trans. Dependable Sec. Computer., 2008, pp.37-48.

[7] P.H.Chan, Wei Fan, Andreas, Salvotore, " Distributed Data Mining in Credit Card Fraud Detection", IEEE Intelligent Systems, Nov 2009.

[8] Srivastava, A., Kundu, A., Sural, S., Majumdar, AK, "Credit Card Fraud Detection Using Hidden Markov Model ", IEEE Transactions on Dependable and Secure Computing, Vol. 5, Issue 1,(2008) 37-48

[9] B. John , B. Jabber ,V. Sudharshan, "A study on Credit Card fraud Detection Methods using Genetic Algorithm," IJECCE, Vol. 3 , No.3, 2012.

10. Linda Delamaire, Hussein Abdou , John Pointon , "Credit card fraud and detection techniques: a review", Banks and Bank Systems, Volume 4, Issue 2, 2009.

Author's Biography

M. Nandhini, received her B.Sc and M.C.A degree from Bharathidasan University, Trichirappalli. M.Phil in Computer Science form Alagappa University, Karaikudi. She is the faculty of department of Computer Science, Pondicherry University, Puducherry.

P.Shanthibala, received her B.Tech and M.Tech degree in Computer Science discipline from Pondicherry University, Puducherry. She is the faculty of department of Computer Science, Pondicherry University, Puducherry.