A TOOL FOR EFFECTIVE DETECTION OF FRAUD IN CREDIT CARD SYSTEM

SATVIK VATS, SURYA KANT DUBEY, NAVEEN KUMAR PANDEY

Institute of Technology and Management, AL-1, Sector-7 GIDA, Gorakhpur, Uttar Pradesh, INDIA
E-mail address- Satvik.vats@gmail.com

Abstract- Due to the rise and rapid growth of E-Commerce, use of credit cards for online purchases has dramatically increased and it caused an explosion in the credit card fraud. Fraud is one of the major ethical issues in the credit card industry. As credit card becomes the most popular mode of payment for both online as well as regular purchase, cases of fraud associated with it are also rising. In real life, fraudulent transactions are scattered with genuine transactions and simple pattern matching techniques are not often sufficient to detect those frauds accurately. Implementation of efficient fraud detection systems has thus become imperative for all credit card issuing banks to minimize their losses. Many modern techniques based on Artificial Intelligence, Data mining, Fuzzy logic, Machine learning, Sequence Alignment, Genetic Programming etc., has evolved in detecting various credit card fraudulent transactions.

Key Words: Electronic commerce, fraud, credit card, genetic algorithms, detection

1. Introduction

In recent years, the prevailing data mining concerns people with credit card fraud detection model based on data mining. Since our problem is approached as a classification

Problem, classical data mining algorithms are not directly applicable. So an alternative approach is made by using general purpose Meta heuristic approaches like genetic algorithms. The Credit Card Is A Small Plastic Card Issued To Users As A System Of Payment. It allows its cardholder to buy goods and services based on the cardholder's promise to pay for these goods and services. Credit card security relies on the physical security of the plastic card as well as the privacy of the credit card number [1]. The term fraud here refers to the abuse of a profit organisation's system without necessarily leading to direct legal consequences. In a competitive environment, fraud can become a business critical problem if it is very prevalent and if the prevention procedures are not fail-safe. Fraud detection, being part of the overall fraud control, automates and helps reduce the manual parts of a screening/checking process. This area has become one of the most established industry/government data mining applications [3]. It is impossible to be absolutely certain about the legitimacy of and intention behind an application or transaction. Given the reality, the best cost effective option is to tease out possible evidences of fraud from the available data using mathematical algorithms [3]. 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. It aims in minimizing the false alerts using genetic algorithm where a set of interval valued parameters are optimized. Fraud detection has been usually seen as a data mining problem where the objective is to correctly classify the transactions as legitimate or fraudulent [5]. For classification problems many performance measures are defined most of which are related with correct number of cases classified correctly [5].

2. Existing System

Ghosh and Reilly have proposed credit card fraud detection with a neural network. They have built a detection system, which is trained on a large sample of labelled credit card account transactions. These transactions contain example fraud cases due to lost cards, stolen cards, application fraud, counterfeit fraud, mail-order fraud, and nonreceived issue (NRI) fraud [4]. The sequence of operations in credit card transaction processing using a Hidden Markov Model (HMM) and show how it can be used for the detection of frauds. An HMM is initially trained with the normal behaviour of a cardholder. If an incoming credit card transaction is not accepted by the trained HMM with sufficiently high probability, it is considered to be fraudulent. At the same time,

ensure that genuine transactions are not rejected [12] .The Traditional detection method mainly depends on database system and the education of customers, which usually are delayed, inaccurate and not in-time [13]. After that method based on discriminate analysis and regression analyses are widely used [13], [10], which can detect fraud by credit rate for cardholders and credit card transaction. For a large amount of data it is not efficient.

3. Problem recognition

The high amount of losses due to fraud and the awareness of the relation between loss and the available limit have to be reduced. The fraud has to be deducted in real time and the number of false alert has to be minimized.

4. Proposed system

The proposed system overcomes the above mentioned issue in an efficient way. Using genetic algorithm the fraud is detected and the false alert is minimized and it produces an optimized result. The fraud is detected based on the customers' behavior. A new classification problem which has a variable misclassification cost is introduced. Here the genetic algorithms is made where a set of interval valued parameters are optimized.

5. System design

The process of design involves "conceiving and planning out in mind and making a drawing, pattern or a sketch". The system design transforms a logical representation of what a given system is required to do into the physical reality during development. Important design factors such as reliability, response time, throughput of the system, maintainability, expandability etc., should be taken into account. Design constraints like cost, hardware limitations, standard compliance etc should also be dealt with. The task of system design is to take the description and associate with it a specific set of facilities-men, machines (computing and other), accommodation, etc., to provide complete specifications of a workable system.

This new system must provide for all of the essential data processing and it may also do some of those tasks identified during the work of analysis as optional extras. It must work within the imposed constraints and show improvement over the existing system... At the outset of design a choice must be made between the main approaches. Talks of 'preliminary design' concerned with identification analysis and selections of the major design options are available for development and implementation of a system. These options are most readily distinguished in terms of the physical facilities to be used for the processing who or what does the work.

SYSTEM ARCHITECTURE

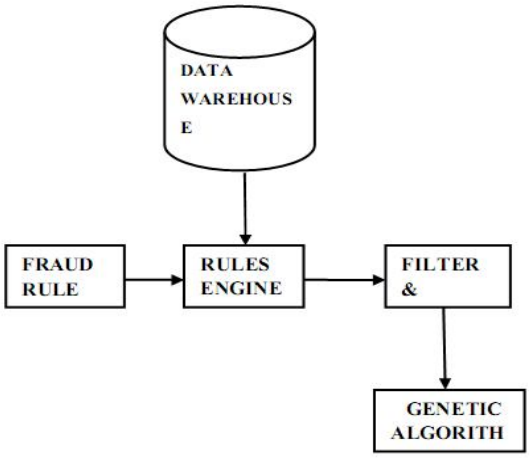

Figure 1: System architecture

5.1. Architectural design

Describing the overall features of the software is concerned with defining the requirements and establishing the high level of the system. During architectural design, the various web pages and their interconnections are identified and designed. The major software components are identified and decomposed into processing modules and conceptual data structures and the interconnections among the modules are identified. The following modules are identified in the proposed system.

6. Modules

6.1. User GUI

In this module, User Interface module is developed using Applet Viewer. This module is developed to user to identify the credit card fraud using genetic algorithm technique. So the user interface must be capable of providing the user to upload the dataset and make manipulations and finally must show the user whether fraud has been detected or not. Only final output will be in applet screen. All the generation details (crossover and mutation) will be in the console screen of eclipse.

6.2. Critical Value Identification

6.2.1. Based on CC usage Frequency

```
float ccfreq
=Float.valueOf(temp[3])/Float.valueOf(t
emp[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;
                     }
```

- 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 (i.e.) = 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 occurrence and critical value =ccfreq

6.2.2. Based on CC usage Location

```
                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;
                    }
```

- 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 (i.e.) = 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

6.2.3. Based on CC Overdraft

```
                float od
        =Float.valueOf(temp[5])/Float.valueOf(t
        emp[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;
                    }
```

- 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 (i.e) = 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 occurrence and critical value = Od w.r.t CU

6.2.4. Based on CC Book Balance

```
                float bb
        =Float.valueOf(temp[2])/Float.valueOf(t
        emp[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 (i.e) = If true, there may chance for fraud using this property and its

critical value is currBB * bb If false, no fraud occurrence and critical value = bb

6.3. Fraud Detection using Genetic Algorithm

Genetic algorithms are evolutionary algorithms which aim at obtaining better solutions as time progresses. Since their first introduction by Holland [6], they have been successfully applied to many problem domains from astronomy to sports [2], from optimization [8] to computer science [7], etc. They have also been used in data mining mainly for variable selection [9] and are mostly coupled with other data mining algorithms. In this study, we try to solve our classification problem by using only a genetic algorithm solution. In this module the system must detect whether any fraud has been occurred in the transaction or not. It must also display the user about the result.

In the following we make clear the concept of genetic algorithms by using an own example over boundary value testing. We implemented this algorithm in Java and can successfully generate inputs for the test. A genetic algorithm is a paradigm often used to search vast and poorly understood search spaces. With well defined functions the algorithm will converge into one area of the search space which holds the optimal solution.

A generic genetic algorithm [11]

```
SimpleGeneticAlgorithm ( )
 {
   initialise population ;
   evaluate population ;
while ( termination criteria not met )
   {
select solutions for nextgeneration ;
 perform crossover and mutation ;
     evaluate population ;
   } }
```

In the same way as a chromosome is the basic building block of nature, so it is of a genetic algorithm. The chromosome is an encoded statement of the data which one wishes to optimise. In our example the chromosome would represent a tuple of all of the input values, and it is encoded as a binary string. The reason for this choice will become clear when further genetic operators are considered. In our example, the inputs 8, 7, 3, 1 and 0 would be encoded as their binary equivalent, and concatenated: 1000 0111 0011 0001 0000. The second task is to write a function to compare the relative merit of chromosomes... The fallowing pseudo code shows, pseudo-code for fitness calculation over an encoding of five bytes, each representing an input integer. A set of chromosomes goes to make up the population of the algorithm. Our algorithm is started with a randomly generated population of chromosomes.

Evaluation of the fitness of a chromosome

```
Int fitness ( Chromosome input )
 {
    int fitness = 0 ;
    int [ ] ideal = new int [A : E ] ;// Array of ideal inputs A to E .
    int [ ] actual = input . to Array ( ) ; // Retrieved at a from chromosome .
     for ( int i = A : E ; i++)
      {
        fitness = absolute ( actual − ideal ) ;
      }
      return fitness ;
  }
```

Crossover is the operator used to reproduce chromosomes. This works by taking a pair of encoded chromosomes - the parents - and combining them to produce two different chromosomes - the progeny. When applied across two fit chromosomes this method aims to produce progeny that have inherited the best attributes of its parents, though this is not always the case. To illustrate the principal, let's consider two chromosomes and assume a central crossover: If the parents are 0011 and 1100 the two progeny will be 0000 and 1111 respectively - see Figure 4. This should make it clear that the bit string is simply crossed, as the name suggests. Depending on the encoding, crossing in the middle of the chromosome may not be likely to give rise to fit progeny, where this is the case other points may be chosen, or indeed more than one point. Another common solution is to select a random point up to the length of the chromosome, and cross there.

Crossover pseudo-code

```
Chromosome crossover(Chromosome parentX , Chromosome parentY )
   { int c r o s s P o i n t = 8 ;
      String x First Half = parentX . substring ( 0 , cross Point ) ;
      String x Second Half = parentX . sub string ( cross Point , parentX . length ) ;
      String y First Half = parentY . sub string ( 0 , cross Point ) ;
      String y Second Half = parentY . sub string ( cross Point , parentY . length ) ;
      Chromosome crossed X = x First Half + y Second Half ;
```

Chromosome crossed Y = y First Half + x Second Half ; }

Mutation is essential to a true genetic algorithm. In popular culture mutation is often viewed in a negative light - simply consider how many horror films are based around some kind of mutant! In fact without mutation neither the world as we know it nor our algorithms would evolve efficiently. Mutation is defined as a minimal change to a chromosome, so when one is using a binary string representation often a single bit is flipped. These changes are usually applied at the end of each generation before the breeding pool and population are combined again, but only with a very small probability of each chromosome being affected. If this was not done then no new genetic information would be produced after the initial population - note that crossover doesn't create anything, rather just recombine existing chromosomes. Without new chromosomes the algorithm is likely to cease with a suboptimal population, or run infinitely never converging on a solution. If, on the other hand, mutation levels are set too high the stream of new chromosomes could be too large, disrupting any convergent progress. If mutation was set to affect every chromosome in each generation and crossover removed, then the search has become completely stochastic.

Mutation pseudo-code

```
Chromosome mutate (Chromosome)
{
    int randomValue = new Random( Chromosome . length ) ;
if ( Chromosome . valueAt ( randomValue ) == 0 )
{ Chromosome . valueAt ( randomValue ) == 1 ;
} else {
Chromosome . valueAt ( randomValue ) == 0 ;
    }
    return Chromosome ;
}
```

6.3.2 Flow of Genetic algorithm

- Initially the initial population is selected randomly from the sample space which has many populations.
- The fitness value is calculated for each chromosome in each population and is sorted out.
- In selection process two parent chromosomes are selected through tournament method.
- The Crossover forms new offspring (children) from the parent chromosomes using single point probability.
- Mutation mutates the new offspring using uniform probability measure.
- In elitism selection the best solution are passed to the further generation.
- The new population is generated and undergoes the same process it maximum number of generation is reached.

REFERENCES

[1] Benson Edwin Raj S., Annie Portia A.: Analysis on Credit Card Fraud Detection Methods. IEEE International Conference on Computer, Communication and Electrical Technology, 18th and 19th ,152-156, 2011.

[2] Charbonneau P., Genetic Algorithms in Astronomy and Astrophysics High Altitude Observatory.National Center for Athmospheric Research, pp. 309-334, 1995.

[3] Clifton phua , vincent lee1, kate smith & ross gayler.: A Comprehensive Survey of Data Mining-based Fraud Detection Research. 2005.

[4] Ghosh S., Reilly D.L., Credit Card Fraud Detection with a Neural-Network. Proc. 27th Hawaii Int'l Conf. System Sciences: Information Systems: Decision Support and Knowledge-Based Systems.vol. 3, pp. 621-630, 1994.

[5] Hamdi Ozcelik M., Ekrem Duman., Mine Isik., Tugba Cevik., Improving a credit card fraud detection system using genetic algorithm. International conference on Networking and information technology, 2010.

[6] Holland J., Adaptation in Natural and Artificial Systems. Ann Harbor, MI: University of Michigan Press. 1975.

[7] Kaya M., Autonomous Classifiers with Understanable Rule Using Multi-objective Genetic Algorithms. Expert Systems with Applications. Vol. 37, no. 4, pp.3489-3494, 2009.

[8] Levi M., Burrows J., Fleming M., Hopkins M., The Nature, Extent and Economic Impact of Fraud in the UK. Report for the Association of Chief Police Officers' Economic Crime Portfolio. 2007.

[9] Minaei-Bidgoli B., Kashy D., Kortemeyer G., Punch W., Predicting Student Performance: An Application of Data Mining Methods with the Educational Web-based System LON-CAPA. Proceedings of ASEEIIEEE Frontiers in Education Conference. 2003.

[10] Ren L., Liping Z., Yinqiang Z., A Study on Construction of Analysis Based CRM System. Computer Applications and Software. vol. 21, pp. 46-47, 2004.

[11] Srinivas M., Patnaik L., Genetic algorithms - a survey. IEEE Computer Society, 1994.

[12] Srivastava A., Kundu A , Sural S., Manjundar A.K .,Credit card fraud detection using hidden markov model. IEEE Transaction on Dependable and Secure Computing. vol. 5, no.1, pp. 37-48, 2008.

[13] Wen-Fang Y.U., Wang N., Research on Credit Card Fraud Detection Model Based on Distance Sum. IEEE International Joint Conference on Artificial Intelligence, 2009.