Multi Agent Based Simulation (MABS) of Financial Transactions for Anti Money Laundering (AML)

Edgar Alonso Lopez-Rojas and Stefan Axelsson*

Blekinge Institute of Technology - School of Computing

Abstract. Mobile money is a service for performing financial transactions using a mobile phone. By law it has to have protection against money laundering and other types of fraud. Research into fraud detection methods is not as advanced as in other similar fields. However, getting access to real world data is difficult, due to the sensitive nature of financial transactions, and this makes research into detection methods difficult.
Thus, we propose an approach based on a Multi-Agent Based Simulation (MABS) for the generation of synthetic transaction data. We present the generation of synthetic data logs of transactions and the use of such a data set for the study of different detection scenarios using machine learning.

Keywords: Machine Learning, Anti-Money Laundering, Money Laundering, Synthetic Data, Data Simulation, Multi- Agent Based Simulation, Fraud detection

1 Introduction

Money Laundering affects the finances of nations and it may contribute to an increase in the funding of criminal activities [1].

Due to the large amount of transactions and the variety of money laundering techniques, it is difficult for the authorities to detect money laundering and prosecute the wrongdoers. Thus, it is not only the amount of transactions, but the ever changing characteristics of the methods used to launder money that are constantly being modified by the *fraudsters* which makes this problem interesting to study.

After analyzing the implications of using machine learning techniques for money laundering detection [2] (also known as Anti-Money Laundering, AML) in a synthetic data set, we propose an approach based on Multi-Agent Based Simulation (MABS).

The main goal and contribution of this paper is to study the generation and use of synthetic data as an approach for developing methods for money laundering detection. A case study containing different scenarios was used as a scientific methodological approach. This leads to identify measures of detection and control that could be applied in similar circumstances.

The case study is based on the company **AB** [1]. Company **AB** has developed a mobile money implementation that provides mobile phone users with the ability to transfer money between themselves using the phone as a sort of electronic wallet. The task at hand is to develop an approach that detects suspicious activities that are indicative of money laundering.

* EMails: {edgar.lopez,stefan.axelsson}@bth.se

[1] The identity of the Company AB unfortunately cannot be disclosed

The mobile money service is currently running in a demo phase. Hence, real data from this system is not available at this stage, and therefore the system does not produce representative data that can be used e.g. for the training of a machine learning detection algorithm. Thus, we have turned to the generation of synthetic data as an alternative.

Outline: The rest of this paper is organized as follows: Section 2 introduce the topic of money laundering and present related work. Sections 3 describes the problem, which is the generation of synthetic data for Anti-Money Laundering. Section 4 presents an implementation of a MABS for our domain. We present our results in section 5 and finish with a discussion and conclusions, including future work in section 6.

2 Background and Related work

Money Laundering exist somewhere in a complex chain that starts with *placement* of illegal funds into the legal financial systems, then a number of *layering* operations to hide the true origins and finally an *integration* stage that involves formal and legal economic activities [3].

Due to issues such as the large amount of transactions typically taking place in a financial service, it is a nontrivial task to find specific transactions that should be marked as suspicious. The reported suspicious activity needs to be supported by tangible evidence that allows relevant government agencies to investigate further.

In Sweden and other countries, most companies in the financial sector are required by law to implement money laundering detection. The cost of implementing such controls for AML is quite high, mainly because of the amount of manual labor required. In Sweden alone the cost is estimated to be around 400 million SEK annually [4]. The most recently notorious case of money laundering is the HSBC Bank case [5], where the lack of AML controls lead to large amounts of money being laundered and injected into the U.S. financial system from countries under strict control, such as Mexico and Iran.

The most common method today used for preventing illegal financial transactions consists on flagging different clients according to perceived risk and restricting their transactions using thresholds [6]. Transactions that exceed these thresholds require extra scrutiny whereby the client needs to declare the precedence of the funds. These thresholds are usually set by law without distinction made between different economic sectors or actors. This of course leads to fraudsters adapting their behavior in order to avoid this kind of control, by e.g. making many smaller transactions that fall just below the threshold. Hence, these and other similar methods have proven insufficient [4].

Several machine learning techniques have been used for the detection of fraud, and more specifically money laundering [7]. The application of machine learning to the problem is advantageous, due to the successful classification rate (high *True Positives* and low *False Positives*) that can be obtained in comparison to simple threshold methods [8, 9].

Data mining based methods have also been used to detect fraud [10]. This leads to the observation that machine learning algorithms can identify novel methods of fraud by detecting those transactions that are different (anomalous) in comparison to the benign transactions. This problem in machine learning is known as novelty detection. Supervised learning algorithms have been used on synthetic data to prove the performance of outliers detection in a different domain [11].

One of the frameworks used for AML, presented by Gao(2007) [12], introduces the terms *legal transaction, usual transaction, unusual transaction, suspicious transaction and illegal transaction* for describing different possible categories of transactions.

Synthetic data has previously been used with similar reasons to the ones presented here [13]. The protection of the clients privacy is an advantage over using real data.

Multi-Agent Based Simulation is an approach that involves the use of autonomous and interactive agents and it is been used to model complex systems. The agents and their interaction with other agents, are described by simple rules, which generates complex emergent behavior usually found in different domains [14].

Previous work have shown the use of Multi-Agent based simulation to simulate social networks and analyzing social behavior [15]. This is similar to *Mobile Money* that resembles a social network of connected clients where the connections are represented by the transactions (money sent or received) and the nodes are represented by the clients.

There are several agent-based frameworks that incorporate toolkits to aid the development of such simulations. Some of them are freely available and are widely used in academic simulations (e.g. MASON[2], Repast [3], or Swarm[4]).

3 AML for Mobile Money

The specific domain covered here is the service *Mobile Money*. *Mobile Money* is a platform for transferring money between users by mobile phone. This is accomplished by the use of codes sent through text messages or the Internet. Mobile money brings several benefits for users, including the simplicity of transferring small sums of money between users. One user only needs to know the mobile phone number of the receiving user in order to send money.

3.1 Problem definition

The detection of money laundering in the mobile money service is non-trivial. Illegal transactions are intended to appear as normal and legal. In this paper we address this problem by learning from the experiences of past detected patterns of illegal behavior in order to hopefully gain knowledge about the possible rules or new patterns of fraud that could emerge in a mobile money system.

In the mobile money AML domain, we formulate the learning problem as [16]: **Task (T)**, Classification of transactions as normal or suspicious based on the known pattern of legal transactions. **Performance Measure (P)**, Percentage of transactions correctly classified as suspicious, also known as *True Positives* (TP), and the percentage of *False Positives* (FP). **Experience (E)**, Synthetic data generated with transactions labeled as legal (normal) and/or illegal (suspicious).

3.2 Data Preprocessing

Data preprocessing includes the selection of attributes, discretization, noise removal and in certain domains, data fusion.

The mobile money product stores all information about the users' interactions with the service in a database. For this study we need to select the database attributes that

[2] MASON http://cs.gmu.edu/ eclab/projects/mason/

[3] Repast http://repast.sourceforge.net/

[4] Swarm http://www.swarm.org/

contribute the most to the correct classification of suspicious transactions. Customers are associated with a specific profile at the opening of the account based on outside information about economic factors. High risk customers are limited e.g. in the amount and the frequency of the transactions that they can perform.

We selected the following attributes for our simulation: *Customer ID*, *Profile*, *Date of Transaction (steps)*, *Type of Transaction* (e.g. deposit, withdraw, transfer), *Amount Transferred*, *Location* (e.g. city) and *Customer Age* (e.g. 1=Young, 2=Adult or 3=Senior).

For each transaction of type *transfer* there is also a *deposit* transaction of the same value in a different customer account. These transfer transactions, describe a social network between customers. The rest of the fields are generated according to the given parameters of the simulation and random operations with range validation to guarantee consistent data that follows a realistic model.

Data labelled as *suspicious* were also added to the transaction database in order to train supervised learning algorithms. These anomalous records were created with the intention to replicate some of the common money laundering patterns used by fraudsters [17].

Although performance is also a topic that we are concerned with here, the learning algorithms selected are seriously affected by the amount of data provided for the training and the cross validation phase.

3.3 Machine Learning Training from Synthetic Data

We are interested in providing an accurate method to improve the detection rate (TP) and reduce the misclassification rate of the benign data (FP) counted on the data collected from the simulation.

Having the same data set, we studied possible algorithms for our detection research. The algorithms analyzed here are based on supervised learning with *Decision Tree* learning and *Decision Rules* techniques [18, 19]. The main advantage of using these algorithms (in comparison with other machine learning algorithms) for mobile money AML, is to enable an investigator to be able to determine common rules that classify suspicious behavior.

We cannot be completely certain of the illegal precedence of the funds in a transaction, that is why our detector only raises a *suspicion* flag that allows an investigator to perform further analysis of the evidence.

4 A Multi-Agent Based Simulation for Mobile Money

In order to illustrate our idea, we developed a simple MABS simulation for the Mobile Money domain. We used MASON for the simulation [20]. The main reason was that it has important extensions that facilitate the implementation of social networks.

4.1 Model

The implementation of a Multi-Agent Based Simulation was based on simulating the behavior of several clients interacting in a Mobile Money environment. Figure 1 shows the basic design we used. Our aim was to produce a log of transactions, represented by the class *Transaction*. This log was built to generate the attributes specified in sect. 3.2.

The simulation is managed by the class *Clients* which initialize the environment and creates the agents. The agents are represented by the class *Client*. This class has

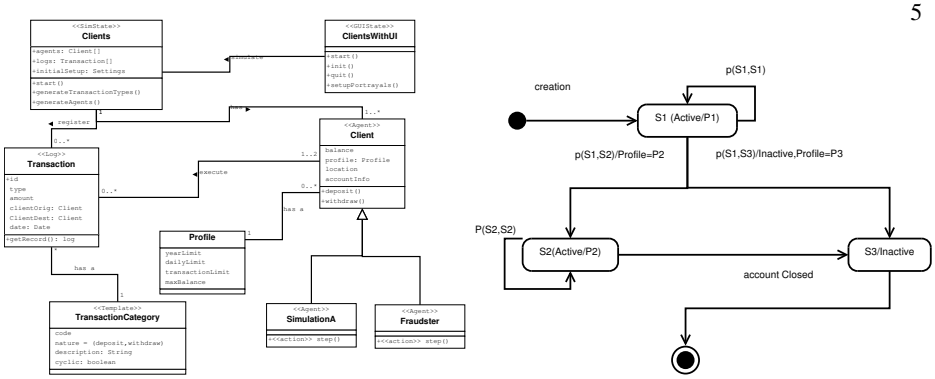

Fig. 1. Simplified Class Diagram and State Diagram for the Mobile Money Simulation

two child classes (*ClientSimA and Fraudster*) which inherits all the behavior added to the parent class *Client*. This allow us to create different types of agents and instantiate them together in the class *Clients*. The states of an agent are handled by a Markov transition matrix of probabilities. This tells the system when to change from Active to Inactive and from Profile P1 to Profile P2, which allows higher limits for transactions.

Each clients has four possible actions in each step of the simulation. They can either make a deposit, a withdrawal, a transfer or simply ”decide” not to do anything. The autonomy of the agent is implemented by a probabilistic transition function that computes the type of operation and the action that an agent will perform in each step. This transition function depends on the attributes of the client such as *Age* and the amount is calculated according to the balance and the limits of each client's profile.

For each simulation we can modify the parameters and the probabilities of occurrence for the transitions in order to improve the quality of the simulation. It is difficult to find the right probabilities that model a realistic scenario. Our implementation is based on pseudo random transitions. The given probabilities are based on 3 different configurations for the percentage of account balance in comparison with the maximum limit allowed by the client profile (Lower than 15%, higher than 80% and *medium balance* which is between *low* and *high*). The agent has a higher probability to make a deposit when the balance is low. When the balance is high the agent has a higher probability to make a withdrawal or a transfer, rather than a deposit.

4.2 Scenarios

Our chosen scenario is an hypothetical situation where 200 clients from 4 different cities perform several transactions with partners inside or outside their city. We decided to have around 10% of the clients behaving as malicious agents (fraudsters). In a real scenario it is more common to find a lower percentage of fraudsters. The idea behind a higher proportion of fraudsters is to prevent the class imbalance problem during the training of the detector. All of the fraudsters were connected in a network where the 3 roles of the money laundering chain are represented (injection, layering and integration).

The social network between the clients was built restricting the network to a maximum of five contacts per client inside the city, and two outside the city. The fraudsters can also interact with normal clients of the system.

All the transactions are stored in a log file. The simulation was run five times for a 1000 steps. Each step represents a time unit that we assume is the transaction rate of the

clients (1/3 per day). The files generated were merged and ultimately used as input for the machine learning algorithms presented in sect. 5.

To reflect a realistic scenario we conserved the thresholds imposed by the original money laundering system. Simplifying the model, all the values in the simulation are given in Swedish Kronor (SEK). For profile P1 there are limits of 2500 SEK (approx. 370 USD) for all transactions per day and a maximum balance of 16000 SEK. For profile P2, which are the validated users, both thresholds are increased to 35000 SEK.

4.3 Synthetic Data generated

In total we simulated 486977 transactions after running 5 simulations, each one with 200 agents running 1000 steps. A total of 6006 transactions were generated by 107 malicious agents and labelled as *suspicious*. Each of the malicious agents was designed with a specific goal in mind chosen from the money laundering cycle that involves the three stages: placement (40), layering(33), and integration(34). The data generated by the simulation represent a realistic situation of the class imbalance problem, where one of the classes is very large in comparison to the other one. In this case only 1.23% of the data is suspicious. For the experiment we ran different supervised algorithms that were selected for the purpose of classifying the class labeled as suspicious transactions.

4.4 Evaluation of the model

We start the evaluation of our model with the verification and validation of the generated simulation data [21]. The verification ensures that the simulation correspond to the described model presented in the chosen scenarios. We can easily check the constraints in the generated data such as positive balance numbers, account age, consistency between the transfers, deposits and withdrawals with the changes in account balances. Validation of the model is a bit more complex, since we need to ascertain whether the model is an accurate representation of a real world situation. Since we do not have real world data at this time we need to rely on a description of the desired scenario and the opinion of experts in the field to validate that the basic statistics and the overall process of the simulation design correspond to a real world scenario. The complexity of the agents also matter here, the simpler the agents the easier is to validate the model.

5 Results and Analysis of performance for different classifiers

For the experiment we used the tool Weka [22]. The selected algorithms were based on *Decision Trees* and *Decision Rules*. From the decision tree category we selected *Random-Tree, Random-Forest and J48graft (C4.5)*. From the decision rules based classifiers we selected JRip, due to its capacity to describe the minority class, and Decision-Table. We added Naive-Bayes as a performance base-line to compare the other algorithms against.

The results can be seen in Table 1(a). We can see that *JRip* produces the best accuracy in TP (True Positive) rate and FP (False Positives) rate in comparison with the other algorithms. The MC (Misclassified) number of instances is a bit higher than for the other algorithms e.g J48graft or Random-Forest.

The tree generated by Random-Tree is relatively bigger than the one generated by J48graft which makes it easier to use by an inspector. However if we intend to add controls to detect money laundering in suspicious transactions we prefer to use Random-Forest or the JRip algorithm over others due to the higher detection rate.

Table 1. (a)Results for the class *money laundering* (suspicious). (b) Confusion Matrix

Algorithm	**TP**	**FP**	**MC**
Naive-Bayes	0.988	0.479	8543
Decision-Table	0.999	0.029	200
Jrip	0.999	0.012	115
Random-Forest	0.999	0.009	66
Random-Tree	0.999	0.015	173
J48graft	0.999	0.014	118

Algorithm	JRip		Random-Forest		J48graft	
class*	**a**	**b**	**a**	**b**	**a**	**b**
a	5934	72	5954	52	5922	84
b	43	480928	14	480957	34	480937

* a=Normal b=Suspicious

We prefer to use accuracy indicators such as (TP and FP) over ROC curve (Receiver Operating Characteristic) to compare the different algorithms, because we are more interested in providing a method to improve the detection rate (TP) and reduce the misclassification of the normal data (FP).

JRip pin points the behavior of our malicious agent with high accuracy. We notice that some of the rules are very strict regarding the balance, since malicious agents are more likely to have a balance that is approaching the threshold of the system. These rules are easily understandable by a human operator and can be straight forwardly incorporated into a money laundering detector.

In Table 1(b) we present the Confusion Matrix for the best three performing classifiers which are J48graft, Random-Forest and JRip. The intersection of class 'a' shows the number of TP, the intersection of class 'a' and 'b' shows the records misclassified. The worst classifier result was expected to be Naive-Bayes according to the TP rate and the high number of misclassified instances. We aim to find a classifier that output a high number of TPs for the class *suspicious* and reduces the number of FP for the class *normal*.

6 Conclusions

The problem of finding anomalies to detect instances of money laundering presents a challenge. Every time a new pattern of fraud is detected by the authorities, and the control mechanism changed, the fraudsters change their *modus operandi* and create a new method that is undetectable by the current threshold-based methods.

We have presented an example of the use of a synthetic data set representing an a simulated scenario in the mobile money domain, for experimentation with machine learning algorithms due to the lack of real data. By doing this we also avoid any possible issue related to privacy and identity protection of the customers of the service.

Through the use of our simulation we can discover flaws in the current system. This can also lead to the finding of new policies and legislation that could prevent the appearance of different patterns of money laundering in the future.

Our analysis employs some of the machine learning algorithms from the categories *Decision Trees* and *Decision Rules*. We think that these algorithms produce an output more understandable by human operators than other machine learning algorithms.

When working with synthetic data, there is always the risk of generating a data set that does not realistically represent the real world. This can lead to results that are biased by the way the data was generated. However, a synthetic data set can also simulate different scenarios (Sect. 4.2) that are not available for experimentation and analysis as they are unusual, catastrophic etc.

Further work will focus on building an improved model with increased fidelity to the real world, for the simulation of mobile money transactions and other examples of fraud . We expect to implement real-world geographical locations with the extension for MASON called GEOMASON.

The generation of a realistic synthetic data set for this domain, that can be validated and verified, is another planned task. We also aim to test the performance of several machine learning algorithms such as Support Vector Machine (SVM), neural networks, Link Analysis and Bayesian networks. These algorithms have been used successfully in previous studies and it is of interest to evaluate them in this domain as well.

References

1. Bartlett, B.: The negative effects of money laundering on economic development. Asian Development Bank Regional Technical Assistance Project No (5967) (2002)
2. Lopez-Rojas, E.A., Axelsson, S.: Money Laundering Detection using Synthetic Data. In: 27th SAIS workshop, Örebro, Linköping University Press (2012)
3. Buchanan, B.: Money launderinga global obstacle. Research in International Business and Finance **18**(1) (April 2004) 115–127
4. Magnusson, D.: The costs of implementing the anti-money laundering regulations in Sweden. Journal of Money Laundering Control **12**(2) (2009) 101–112
5. Levin, C., Bean, E.J., Martin-browne, K.: U.S. Vulnerabilities to Money Laundering , Drugs , and Terrorist Financing : HSBC Case History. Technical report (2012)
6. Bolton, R., Hand, D.: Statistical fraud detection: A review. Statistical Science **17**(3) (2002)
7. Sudjianto, A., Nair, S., Yuan, M., Zhang, A., Kern, D., Cela-Díaz, F.: Statistical Methods for Fighting Financial Crimes. Technometrics **52**(1) (February 2010) 5–19
8. Zhang, Z., Salerno, J.: Applying data mining in investigating money laundering crimes. discovery and data mining (Mlc) (2003) 747
9. Yue, D., Wu, X., Wang, Y.: A Review of Data Mining-Based Financial Fraud Detection Research. In: 2007 Wireless Comm., Networking and Mobile Computing, Ieee (2007)
10. Phua, C., Lee, V., Smith, K., Gayler, R.: A comprehensive survey of data mining-based fraud detection research. Arxiv preprint arXiv:1009.6119 (2010)
11. Abe, N., Zadrozny, B., Langford, J.: Outlier detection by active learning. Proceedings of the 12th ACM Int. conference on KDD '06 (2006)
12. Gao, Z., Ye, M.: A framework for data mining-based anti-money laundering research. Journal of Money Laundering Control **10**(2) (2007) 170–179
13. Barse, E., Kvarnstrom, H., Johnson, E.: Synthesizing test data for fraud detection systems. 19th Annual Computer Security Applications Conference, 2003. Proceedings. (2003)
14. Salamon, T.: Design of Agent-Based Models. Tomas Bruckner, Zivonin (2011)
15. Pavon, J., Arroyo, M., Hassan, S., Sansores, C.: Agent-based modelling and simulation for the analysis of social patterns. Pattern Recognition Letters **29**(8) (June 2008) 1039–1048
16. Mitchell, T.M.: Machine learning. McGraw-Hill series in artificial intelligence, 99-0177686-4. McGraw-Hill, New York (1997)
17. Irwin, A.S.M., Choo, K.K.R., Liu, L.: Modelling of money laundering and terrorism financing typologies. Journal of Money Laundering Control **15**(3) (2012) 316–335
18. Quinlan, J.: Simplifying decision trees. Int. journal of man-machine studies **27**(3) (1987)
19. Breiman, L.: Random forests. Machine learning (2001) 5–32
20. Luke, S.: MASON: A Multiagent Simulation Environment. Simulation **81**(7) (2005)
21. Ormerod, P., Rosewell, B.: Validation and Verification of Agent-Based Models in the Social Sciences. In Squazzoni, F., ed.: LNCS. Springer Berlin / Heidelberg (2009) 130–140
22. Garner, S.: Weka: The waikato environment for knowledge analysis. Proceedings of the New Zealand computer science (1995)