ADVANCED COMPUTING SESSION

ANOMALIES DETECTION IN THE APPLICATION LOGS USING KOHONEN SOM MACHINE LEARNING ALGORITHM

Vladimir S. Marković,
Angelina Njeguš*,
Marina Marjanović

Singidunum University,
Belgrade, Serbia

Abstract:
Internal fraud in the financial sector are difficult to detect since fraudulent transactions are indistinguishable from ordinary transactions, and standard checkpoints, in the form of transaction documentation and authorization, are skillfully avoided. Well-designed software has available and machine-readable application logs that can be analyzed to detect anomalies in application usage. This paper presents a data preparation technique using path analysis and Kohonen SOM clustering algorithm that can help better profile users of an application to reduce the number of cases that will be further investigated.

Keywords:
internal fraud, Self-Organizing Mapping, Kohonen, path analysis, detection anomalies.

1. INTRODUCTION

Traditional internal fraud detection techniques are complex, time-consuming, and require expertise. It incorporates knowledge from economy, finance, law and other relevant fields. Internal fraud detection techniques are usually based on prior experience, applying specific *if-then* rules within an expert system. Fraud is a "profitable" business, so fraudsters are constantly finding new ways to avoid detection systems and stay undetected. *If-then* rules are static, easily bypassed and quickly deprecated. In addition, after system setup, some rules can be expired. So, the rules show that all or at least the vast majority of employees are suspicious or no employees are suspicious. Therefore, the challenge, and research questions are: Whom to check first? How to fine-tune the system to reduce the number of *"false-negatives"* i.e. to reduce the time spent on deeper investigation?

The ability to describe business processes clearly and unambiguously through application logs enables us to understand and anticipate the behavior of users at the lowest level, observing the user through every aspect and part of the transaction, and then, grouping users by activity, profiles the user groups of interest for the analysis [1].

Correspondence:
Angelina Njeguš

e-mail:
anjegus@singidunum.ac.rs

DOI: 10.15308/Sinteza-2020-275-282

This type of analytics is called behavioral analytics. *Behavioral analytics* uses machine learning techniques to group users according to their activity, identify the most important user groups, and detect atypical patterns and anomalies.

These analyzes of application logs may be used to improve user experience, enhance the application, and detect malicious use of the application.

2. LITERATURE REVIEW

Types of anomalies

There are 3 basic types of anomaly [2]: contextual anomaly, point anomaly, and collective anomaly.

Contextual anomaly is a behavior atypical in the appropriate context, while in another context it represents a correct behavior. For example, air temperature of 18 degrees in January in Belgrade is a contextual anomaly relative to the average temperature, while 18 degrees in May is not an anomaly.

When a particular data instance deviates from the usual data instances, it can be considered a *point anomaly*. For example, 3 online purchases over 2 hours from a person who never or rarely purchases online is a point anomaly.

A *collective anomaly* is an atypical behavior of similar data instances relative to the entire dataset. For example, frequent insights into balance of certain accounts by some bank tellers can be a collective anomaly. It may happen that a single atypical instance of the data is not an anomaly itself, but together with other data instances it forms a collective anomaly. So, insight into the account balance is not an anomaly because it is part of the transaction, but together with some other transactions it may be a collective anomaly.

Main challanges in anomalies detection

The problem of detecting anomalies by analyzing deviations from normal behavior is not trivial. There are many challenges that modelers face to:

- There are no general techniques in anomaly detection. What is typical and what is atypical depends on the view of the data, business and problem. So, some speculative activities on the stock market can be seen as both fraud and common behavior.
- The data contains noise that looks as an anomaly, so it is difficult to separate the noise from the anomaly itself. The same techniques that detect anomaly in a business context are also used to detect noise in the data quality checking.
- The small number of detected fraudulent cases vs. the large number of transactions executed limits the choice of machine learning method. Therefore, the technique of unsupervised machine learning is used while the technique of semi-supervised or supervised machine learning is less common. Some techniques have also been described in earlier works [3,4].
- Normal behavior changes, therefore what was considered normal may no longer be considered and vice versa. For example, the reference values some biochemical analyzes change over time, and thus the context of their application changes.
- Malicious behavior aspires to become normal by imitating normal behavior.

Choice of machine learning technique for anomaly detection

Cluster analysis is the grouping of a set of objects in such a way that objects in the same group (called cluster) are similar to each other (in a sense) than to those in other groups (clusters). The cluster analysis itself is not some specific algorithm, it is a general task that needs to be solved using the obtained clusters. This may be achieved by a variety of algorithms that differ significantly in understanding what constitutes clusters and how to find them effectively. Popular explanation for clusters involves groups with small distances between cluster members, dense data space, intervals or certain statistical distributions, various projections that keep distances between data instances, etc. Clustering is an iterative process of knowledge discovery or an interactive, more objective, optimization that involves trials and failures. It is often necessary to change the model parameters until the result reaches the desired effects.

Clustering is an unsupervised machine learning technique that does not require labeled data to find rules for grouping similar data instances. It is a basic technique for understanding unlabeled data and the most commonly used technique for anomaly detection. The aim of clustering is not only to determinate clusters, it includes the interpretation of clusters by analyst, who provides meaningful insight into data instances by analyzing each cluster separatelly, and depending on the results, appropriate actions may be taken.

The choice of a clustering algorithm is a very important part of the model development.

Different clustering algorithms have different measures of similarity/dissimilarity, using measures of distance from the cluster, which are based on the similarity of two data instances. The choosing depends on the assumption of the relationships of the data instance. Therefore if there are the following assumptions:

- Common data instances belong to data clusters, while anomalies do not belong to any of the clusters. The following algorithms are suitable: DBSCAN, ROCK i SNN.
- Common data instances are located close to the centroid within the cluster while anomalies are not so close to the centroid and located on the perimeters of the cluster. Suitable algorithms are: Self-Organizing Maps (SOM), K-means and Expectation Maximization (EM).
- Common data instances belong to large and dense clusters, while anomalies belong to small and sparse clusters. Suitable algorithm is Cluster-Based Local Outlier Factor (CBLOF).

3. RESEARCH METHODOLOGY

Dataset

The dataset contains application logs of banking core application. Logs are grouped into sessions whereby the session represents the work of an employee at the position over one day in the branch. Except for the session which incorporates date, employee, branch and position in the branch, the analysis uses time of the transaction and transaction type. In total, more than 30 million logs (basic transactions) were considered, accounting for approximately 3 million banking transactions.

In order to avoid additional complexity, the dataset is further limited to:

- logs in one month outside the holiday season,
- conduct of employees in the same position by doing the most basic banking transactions, assuming that everyone uses the application in the same manner, following clearly defined rules of usage.

Steps in analyzing logs

Step 1. Data preparation:

- Transformation of data into a structure suitable for path analysis.
- Recognition of behavioral patterns. Conducting path analysis and detect sequential usage patterns.
- Transformation of path analysis results into session vectors.
- Making fingerprint of each user by calculating statistics of session vectors.

Step 2. Fingerprints analysis and collective anomaly detection using Kohonen SOM learning.

Application logs paths analysis

Path analysis is an analytical technique used to identify sequential patterns in event history [5]. The sequential pattern found in this way is called *path or rule*. Each path contains one or more sequential events that relate to one application session. Path analysis may help in better understanding of application user behavior and business processes. This technique is most commonly used in analyzing visitor traffic through an e-commerce site, in determining the most commonly used paths, as well as the paths to purchase. The results of these analyzes are used to create for example association models used by Amazon, recognition of shopping preferences that are further used in CRM campaigns, personalization site content and direct marketing, and other.

Preparing data for path analysis is simple. It is necessary to transform the existing log table into the form as shown in Figure 1.

Figure 1. Table of application log

The table has to contain session identifier, transaction datetime, previous and current transaction type and transaction duration. Now, the rows contain all the information of both the current and the previous event, whereby the row represents a simple rule.

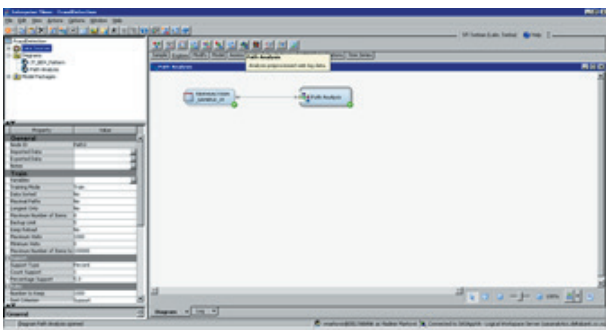


Figure 2. Path analysis using SAS Enterprise Miner

For each session, all subarrays of session events are found, with max length of the subarrays that is limited to predefined value. This restriction is introduced due the complexity of the algorithm.

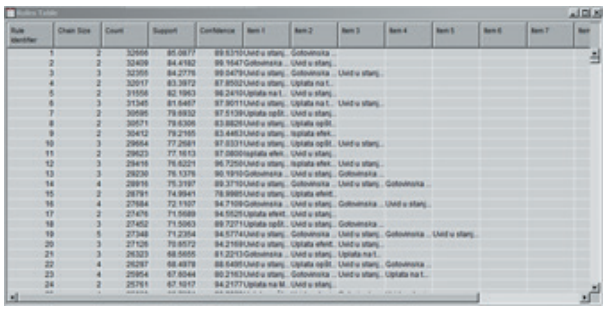


Figure 3. Result of path analysis – rules and their statistics

The following measures are calculated for each rule:

- *Count* – number of rules in the sample
- *Support* – percent of occurrence rule r in total number of rules in the sample R.
- *Confidence level.* Percent of occurrence rule B after A in A→B including all rules which contain A→B.

The result of the path analysis can be graphically represented as shown in the Figure 4. The size of nodes and colors determine the number of B after A in rules *A→B,* while the thickness and color of the edges determine the confidence level. The basic idea behind path analysis is to transform a transaction logs into a pattern suitable for further research. For each session, except basic variables, new variables of rules are introduced.

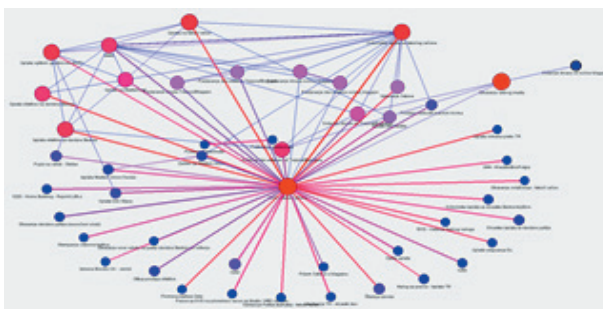
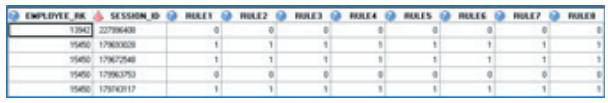


Figure 4. Result of path analysis – link graph visualization

Variables of rules are signed as <RULE><rule identifier>, whereby rule identifier taken from result of the analysis (Figure 3). To simplify, the variable is binary i.e. the value 1 means that the rule is appeared in the session, and otherwise the value is 0 (Figure 5).



EMPLOYEE_RK	SESSION_ID	RULE1	RULE2	RULE3	RULE4	RULE5	RULE6	RULE7	RULE8
13942	227996408	0	0	0	0	0	0	0	0
15450	179830828	1	1	1	1	1	1	1	1
15450	179672548	1	1	1	1	1	1	1	1
15450	179963753	0	0	0	0	0	0	0	0
15450	179740117	1	1	1	1	1	1	1	1

Figure 5. Analytical base table on granularity of the session

Instead of the binary identificatory, it is possible to use number of occurrences of rules in the session, or some other weighting of occurrences in order to favor some rules and change shape of data instances before clustering. The weighting is based on expert knowledge and previous fraud surveys. Also, sessions can be parts of the day, but not all day as in the sample.

It has been identified 508 different rules in the sample. Usually, the analysis takes the rules that satisfy that the Support or Confidence Level is higher than the predefine percentage (trash hold) in order to cut "less" important rules. Since, the aim is to detect anomalies in this paper, all rules were taken regardless of the number of occurrences. Rarely used rules represents the fine-tuning of using the application because the usage of the application is clearly defined trough business processes.

Making user fingerprints

The goal of the analysis is not to profile sessions or application logs, but to profile application users. Therefore, further aggregation of the session table into the user table is needed to create an analytical base table (ABT) at the user level, where an observation (row) describes how user uses the application for defined time period.

Two statistical measures are used to aggregate the sessions. These are the arithmetic mean and the coefficient of variation (CV).

EMPLOYEE_RK	CV_of_RULE1	MEAN_of_RULE1	CV_of_RULE2	MEAN_of_RULE2
13942	.	0	.	0
15450	45.614304805	0.829571426	45.614304805	0.829571426
22236	23.561727767	0.948051948	23.561727767	0.948051948
22480	27.989810491	0.929571426	41.194292044	0.957142857
27458	58.640485705	0.748666667	58.640485705	0.748666667

Figure 6. User fingerprints – vector space which describes using application

The coefficient of variation is a relative measure, usually in the interval [0,100], which shows the extent of variability in relation to mean of the population. It is used to compare the variability of two different samples with different arithmetic means.

As shown in the Figure 6, there is a significant difference in usage of RULE1 between employees *15450* and *22480*. Variability of RULE1 for employee *15450* is 1.6 times greater than variability of employee *22480*. The arithmetic mean represents the number of days the rule was used. So, employee *22236* uses the rule 95% of the day at least once.

An employee's ABT has been created with measures of using some rules (paths) for predefined time period. The ABT has 1016 variables (508 for CV and 508 for arithmetic mean). Now, the matrix (1800x1016) determines vector space of application usage where a column is a vector for a rule and a row is a vector of using application by a user called *fingerprint of the application usage*.

4. KOHONEN SOM CLASTERING ALGORITHM

Self-organizing mapping (SOM) is a competitive neural network that provide topological mappings from entry space to clusters [6]. SOM algorithm evolved from early neural network models, especially models of associative memory and adaptive learning [7], trying to explain the spatial organization of brain functions, which is especially observed in the cerebral cortex. Nevertheless, SOM was not pioneer in the area, there were also the spatially ordered von der Malsburg line detectors (1973) and the neural field model of Amari (1980). However, SOM is strong formal with defined convergence of the algorithm [8].

SOM was originally developed to visualize metric vector distributions, such as measurement values or statistical attributes, while later was shown that SOM mapping can be defined for any data instance, where distance can be defined.

Examples of non-vector data that can be adapted to this method are strings and arrays of segments in organic molecules. The first area of application of SOM was speech recognition. SOM has found widespread use in data analysis and data exploration [9-15].

The idea of the mapping is to group, visualize, and abstract a multidimensional space into usually two-dimensional, rarely one-dimensional, three-dimensional, or higher. SOM is an ordered mapping, a kind of projection of a multidimensional space into a two-dimensional, where an instance of the data will be mapped to a node whose model is the "most similar" data instance, i.e. it has the smallest distance from the data using specific metrics [16].

Using the SOM algorithm, it is tried to find clusters such that any two clusters that are close to each other originate from points of input space whose potential clusters are also close [17]. Vice versa it is not case. Points close to each other do not necessarily correspond to clusters that are close in the SOM network. Thus, the SOM is a discrete smooth mapping between regions of the input space into nodes (neurons) that determine the lattice in two-dimensional space [18].

SOM works as smoothing in a manner similar to the kernel estimation method, except that smoothing is performed in two-dimensional space rather than in the input space [19].

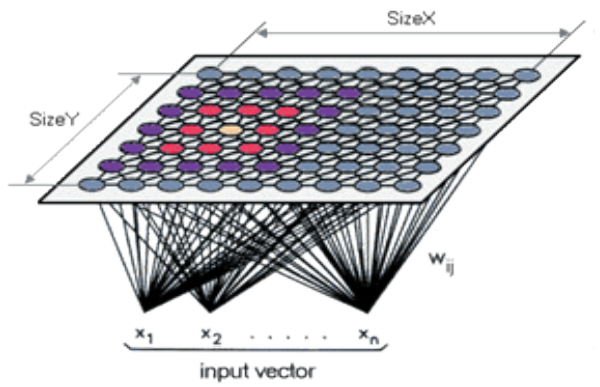


Figure 7. The projection of multidimensional space into two-dimensional [20]

Mathematical model definition

Let n-dimensional vectors in the Euclidean n-dimensional space.

$$x(t)=[x_1(t),x_2(t),\ldots,x_n(t)] \quad (1)$$

where t denotes the index of the data instance.

Let m_i, i^{th} model (prototype)

$$m_i(t)=[\mu_{i1}(t),\mu_{i2}(t),\ldots,\mu_{in}(t)] \quad (2)$$

where t denotes the sequence of data used to train the model.

Next value $m_i(t+1)$ is calculated using previous $m_i(t)$ and next data instance $x(t)$:

$$m_i(t+1)=m_i(t)+\alpha(t)h_{ic^t(x)}(t)[x(t)-m_i(t)] \quad (3)$$

$\alpha(t)$ - positive scalar called learning rate which is decreased for each next step

$c^t(x)$ – cluster which has the smallest Euclidean distance in relation to $x(t)$ called *best matching unit i.e.*

$$c^t(x)=agr\ min_{i\in[1,\dots,I]}\left\|x-m_i(t)\right\|^2 \quad (4)$$

$h_{ic^t(x)}(t)$ – smoothing function ("*neighborhood function*").

The function h is equal 1 when $i=c$, and h decreases as the distance between models m_i i m_c increase. Also, h decreases for each step t. The function h ensures convergence and must be carefully chosen [6]. Problem of initialization $m_i(1)$ is specifically considered at Kohonen [6, 12].

The result is a data space partition (Figure 8), called *Voronoï tessellation*, with a neighborhood structure between the clusters. The Kohonen map is the representation of the prototypes or of the cluster contents displayed according to the neighborhood structure [8].

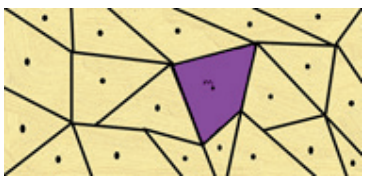


Figure 8. $c^t(x)=i \Leftrightarrow m_i$) is the *winning prototype of x*

Although the above iterative algorithm has been used successfully in many applications. It has also been proven in practice that the algorithm called *Batch Map* gives similar results but converges faster [5].

The basic idea is that for each node j determines mean of $x(t)$ denotes as $\bar{x}_j$, whereby m_j is the winning prototype for $x(t)$. The next formula can be used

$$m_i=\sum_j n_j h_{ji} \bar{x}_j / \sum_j n_j h_{ji} \quad (5)$$

where:

n_j - number of input instances mapped to a node j

j - represents all indexes near i.

m_i - for m_i update this schema has several iterations using the same group of input data instances to determine $\bar{x}$.

5. KOHONEN SOM ANALYSIS – EXPERIMENT

The analysis has been carried out on the dataset of 1800 employees for whom the application usage footprint has been calculated in one month (Figure 6) using SAS Enterprise Miner and Kohonen SOM node. Nadaraya-Watson smoothing method was used, where the convergence criterion is 0.0001 and the maximum number of learning iterations is 10.

Principal component analysis w(PCA) was used to initialize the two-dimensional plane, using split into 6x6 segments in which 1800 employees will be grouped - average of 50 employees per segment.

The reason for choosing this method lies in the desire to intersect the multidimensional plane with a two-dimensional plane defined by the principals. It represents the "best" projection, with the aim to maximize in keeping the distance between points in both directions - original space to two-dimensional space and vice versa.

In business terms, the desire is to get coherent clusters as soon as possible, i.e. that employees who do not use the application for the entire period be in separate clusters, and that we may look for potential fraudsters on the perimeter of the clusters that have the highest total distance points relative to the centroid.

At the beginning it was considered to use "Outlier" as method of plane initialization which is default in SAS Kohonen SOM node. That means using points "distant" from each other. This idea was abandoned because additional analysis is shown that outlier fingerprints belong to employees who were on sick leave or vacation. Based on experience, employees who do internal fraud usually do not go on vacation, come the first at the branch and leave the last.

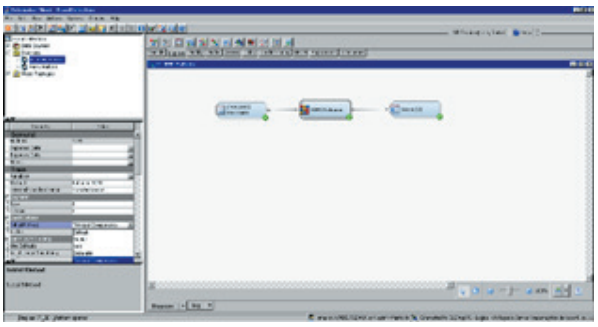


Figure 9. Kohonen SOM analysis using SAS Enterprise Miner

6. INTERPRETATION OF THE RESULT

The result of the analysis is to group each employee into one of 36 (6x6) segments and calculate the distance of each employee from the cluster center as well as the distance between the clusters. This result set provides a very efficient visual presentation of the results using a heat map.

The most commonly used measures for analysis:

- frequency of cluster - number of observations in each cluster
- root-mean-square error standard deviation
- maximum distance from cluster seed
- nearest cluster
- distance to nearest cluster

There are 4 heat maps which show 4 different measures (Figure 10). The first what is seen is the number of clusters (1,1) that is separated from other clusters and has employees far from the center of the cluster. The analysis of cluster (1,1) revealed that these were newcomers who had not worked for a full month and employees who had been absent for at least 7 days. Therefore, they are not subject to further analysis. The second is to analyze cluster (6,1) where frequency of cluster is the biggest. The challenge for the cluster is its coherence and all employees are so close to centroid. If there are some fraudsters it is hard to detect it because they have as same behavior as normal employees. So, we will skip this cluster and conduct some other type of analysis for it.

Assuming that fraudulent pretend to be ordinary employees and they don't want to be discovered, further research should be focused on clusters close to each other, with the average distance from the center being approximately the same in these clusters, i.e. we should observe a region surrounded by clusters (4,3) and (6,6) - a total of 12 clusters.

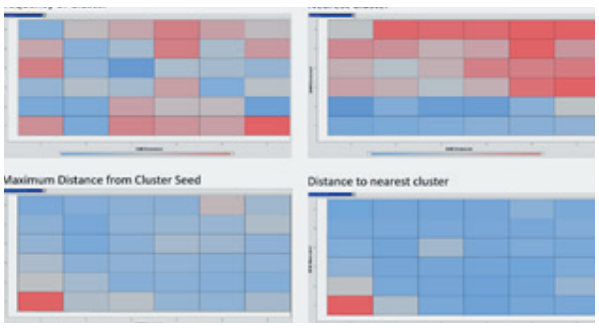


Figure 10. Visualization of Kohonen SOM clustering results

All of these clusters are close to (6,1) for which we assume that is normal behavior. We note that cluster (5,6) is the most numerous and clusters (4,5), (4,6) are the farthest in the group, so anomalies should be sought among them.

Based on analysis, a group of 10 employee may be extracted for which behaviors are anomalies (outliers in the clusters) and then fraud detection experts may conduct detailed analysis (investigation) using all available employee data as well as descriptive statistics methods. The results of the analysis are shown in Figure 11.

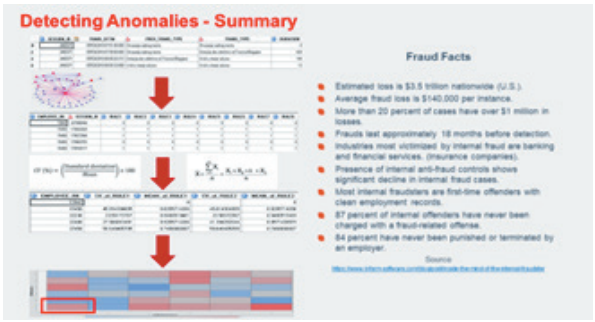


Figure 11. Data flow – from raw logs to the final result

7. CONCLUSION

Based on previous surveys, only 3% of frauds are detected using data analytics. Analyzes of application logs based on unsupervised machine learning to detect anomalies are usually the first step in fraud detection. Modern internal fraud detection systems are increasingly using the results of these analyzes to create general disclosure rules, as these analyzes may reduce the number of cases that will be the subject of further investigation. The paper shows the complex transformation of raw logs to employee's footprints i.e. defining session, finding prior transaction, analyzing paths, and finally aggregating session-level data into an employee's footprint. The cluster analysis was performed on employee's footprints using the Kohonen SOM learning, and the results of the analysis were interpreted in a business sense.

REFERENCES

[1] R.J. Bolton and D.J. Hand, "Unsupervised Profiling Methods for Fraud Detection", Technical Report (Department of Mathematics, Imperial College, London) 2002.

[2] V. Chandola, A. Banerjee, and V. Kumar, „Anomaly detection: A survey", *ACM Computing Surveys*, vol. 41, iss. 3, Article No. 15, July 2009.

[3] M. Ahmeda, A.N. Mahmood, M.R. Islam, "A survey of anomaly detection techniques in financial domain", *Future Generation Computer Systems*, vol. 55, pp. 278-288, February 2016.

[4] A-S. K. Pathan, T*he State of the Art in Intrusion Prevention and Detection*, 1st Edition, Auerbach Publications, MA, United States, 2014.

[5] SAS Institute, SAS/STAT 15.1 User's Guide. Cary, NC: SAS Institute, 2016. Available at: https://documentation.sas.com/?docsetId=statug&docsetTarget=titlepage.htm&docsetVersion=15.1&locale=en (23.07.2020)

[6] T. Kohonen, *Self-Organizing Maps*. Third Ed. Springer, Berlin, Heidelberg, New York, 2001.

[7] T. Kohonen, „Essentials of the self-organizing map", *Neural Networks*, vol. 37, pp. 52–65, 2013.

[8] H. Yin, „The Self-Organizing Maps: Background, Theories, Extensions and Applications", In: Fulcher J., Jain L.C. (eds) *Computational Intelligence: A Compendium. Studies in Computational Intelligence*, vol 115. Springer, Berlin, Heidelberg, 2008.

[9] Ł. Brocki, D. Koržinek, „Kohonen Self-Organizing Map for the Traveling Salesperson Problem". In: Jabłoński R., Turkowski M., Szewczyk R. (eds) *Recent Advances in Mechatronics*. Springer, Berlin, Heidelberg, pp. 116-119, 2007.

[10] J. Faigl, „An Application of Self-Organizing Map for Multirobot Multigoal Path Planning with Minmax Objective, *Computational Intelligence and Neuroscience*, vol. 2016, pp. 2720630:1-15, 2016.

[11] S. Kaski, J. Kangas, and T. Kohonen, „Bibliography of self-organizing map (SOM) papers: 1981-1997", Neural Computing Surveys, vol. 1, no. 3&4, pp. 1 – 176, 1998.

[12] M. Oja, S. Kaski, and T. Kohonen, „Bibliography of Self-Organizing Map (SOM) Papers: 1998-2001 Addendum". *Neural Computing Surveys*, vol. 3, pp. 1-156, 2003.

[13] M. Pöllä, T. Honkela, and T. Kohonen, „Bibliography of Self-Organizing Map (SOM) Papers: 2002-2005 Addendum", *Neural Computing Surveys*, 2007.

[14] I.I. Priezzhev, P.C.H. Veeken, S.V. Egorov, A.N. Nikiforov, U. Strecker, „Seismic waveform classification based on Kohonen 3D neural networks with RGB visualization", *First Break*, Vol. 37, Iss. 2, pp: 37-43, 2019.

[15] P. Stefanovič, O. Kurasova, „Outlier Detection inself-organizing maps and their quality estimation", *Neural Network World*, vol. 28, iss. 2, pp. 105-117, 2018.

[16] Q. Deng and G. Mei, "Combining self-organizing map and K-means clustering for detecting fraudulent financial statements," *IEEE International Conference on Granular Computing*, Nanchang, pp. 126-131, 2009.

[17] D. Brugger, M. Bogdan and W. Rosenstiel, "Automatic Cluster Detection in Kohonen's SOM", *IEEE Transactions on Neural Networks*, vol. 19, no. 3, pp. 442-459, March 2008.

[18] T. Kohonen & T. Honkela, „Kohonen network", Scholarpedia, vol. 2., no. 1, pp. 1568, 2007.

[19] R. Wehrens, R. Kruisselbrink, „Flexible Self-Organizing Maps in Kohonen 3.0", Journal of Statistical Software, vol. 87, pp. 1-18, 2018.

[20] J. Burguillo and B. Dorronsoro, „Using Complex Network Topologies and Self-Organizing Maps for Time Series Prediction". *Advances in Intelligent Systems and Computing*, vol. 210, 2013.