Research Article GU J Sci 35(3): 854-873 (2022) DOI: 10.35378/gujs.854725

Gazi University

Journal of Science

http://dergipark.gov.tr/gujs

Money Laundering Detection with Node2Vec

Mehmet CAGLAYAN, Serif BAHTIYAR*

Department of Computer Engineering, Istanbul Technical University, Ayazaga Campus, 34469, Maslak, Istanbul, Turkey

Highlights

- This paper focuses on money laundering detections with machine learning.
- A graph-based representation for banking transactions is used.
- The proposed Node2Vec based solution is analyzed on a dataset.

Article Info

Received: 05 Jan 2021
Accepted: 26 Sep 2021

Keywords

Money laundering
Machine learning
Financial transaction
Security

Abstract

The widespread use of computing technology has been changing relationships among people in societies. Criminals are aware of the power of the technology so that many criminal activities involve more computing systems. Money laundering has been a significant criminal activity within financial computing systems for many decades. The dynamic nature of information systems has reduced the effectiveness of existing money laundering detection mechanisms that is an important challenge for societies. In this paper, we consider machine learning algorithms as complementary solutions to existing money laundering detection mechanisms. We have focused on graph-based representation of data with Node2Vec to have better classification results for money laundering detections with machine learning algorithms. Our experimental analyses show that Node2Vec enable us to select the most convenient machine learning algorithm for money laundering detections.

1. INTRODUCTION

Since the digitalization of the world has become faster than ever, the dependency on computing systems has increased dramatically that results in many social challenges. Cybercrime is one of the challenges that societies should overcome. For instance, some users alter, vandalize, and take advantage from computing systems for their own benefits, which contradicts with the benefit of societies. Actually, the word of cybercrime corresponds many areas. In this paper, we consider only one of the most influential cybercrimes for societies, which is called money laundering.

Recently, machine learning algorithms have been used to detect many cyber-attacks. However, there is no a common machine learning algorithm that is used to detect a particular cyber-attack with high accuracy. The main reason for that is the number of attacks in a dataset. Specifically, security related datasets have different properties than other datasets that are used with machine learning algorithms. Therefore, money laundering detections with machine learning algorithms have provided a significant opportunity for societies with huge research challenges.

We investigate machine learning algorithms to detect anomalies in money laundering attempts in banking transactions, which is still a huge research challenge. The state of the art contains many solutions related to detections of money laundering, which are applicable only to specific financial domains. We observe that more accurate money laundering detections are provided by using graph-based solutions. In this research, we have focused on graph-based money laundering detections with machine learning. Specifically, it has been observed that suspicious transactions may be detected by using patterns in the relational network with the help of graph-based anomaly detecting systems [1]. Therefore, we have selected Node2vec algorithm

*Corresponding author, e-mail: bahtiyars@itu.edu.tr for clustering purposes with machine learning algorithms to have better performance results. Actually, Node2Vec has been used for fraud detections, which is a similar problem with money laundering detections contains the skewness of data [2]. Particularly, Node2Vec is used for exposing hidden relationships within data. The results taken with Node2Vec is used for classification purposes. In this research, we have used many classification algorithms after Node2Vec algorithm to detect money laundering attempts, such as Naive Bayes (NB) and K-Nearest Neighborhood (K-NN) algorithms due to their plainness [3,4]. Our initial goal is to explore the performance of Node2Vec for money laundering classifications without over tuning any classification algorithm. Although, banking transactions and related accounts are used for money laundering detection in this paper like in [5], graph-based criminal relationship representations may be constructed with the analysis of the historical traffic.

The main contribution of this research is a new approach to determine the most convenient machine learning algorithms with Node2Vec for money laundering detections. To the best of our knowledge, this is the first research that use Node2Vec algorithm for money laundering detections. Our other contributions are as follows:

- Comparing machine learning algorithms in terms of accuracy in case of oversampling for money laundering detection;
- Determining the accuracy of classification algorithms for money laundering;
- Combining classifiers with Node2Vec algorithm for money laundering detections.

The rest of this paper is organized as follows. Section 2 is about the state of the art related to money laundering detections and machine learning algorithms. In section 3, we present our approach. Next section is devoted to performance evaluations. Section 5 concludes the paper.

2. MONEY LAUNDERING AND MECHINE LEARNING

2.1. Money Laundering and Society

According to International Criminal Police Organization (Interpol), money laundering is a transaction that results from illegal activities for profits, such as drug trafficking, robbery, or extortion to official institutions as clean and traceable through legal activities [6]. Money laundering is the root of many crime types that creates illegal money. This type of money is gathered within an organization that has a solid structure. According to United Nations Office on Drugs and Crime (UNODC), the volume of laundered money makes 2%-5% of the total GDP of all countries, which accounts from $800 billion to $2 trillion USD [7]. Figure 1 shows the distribution of transactional crime [8].

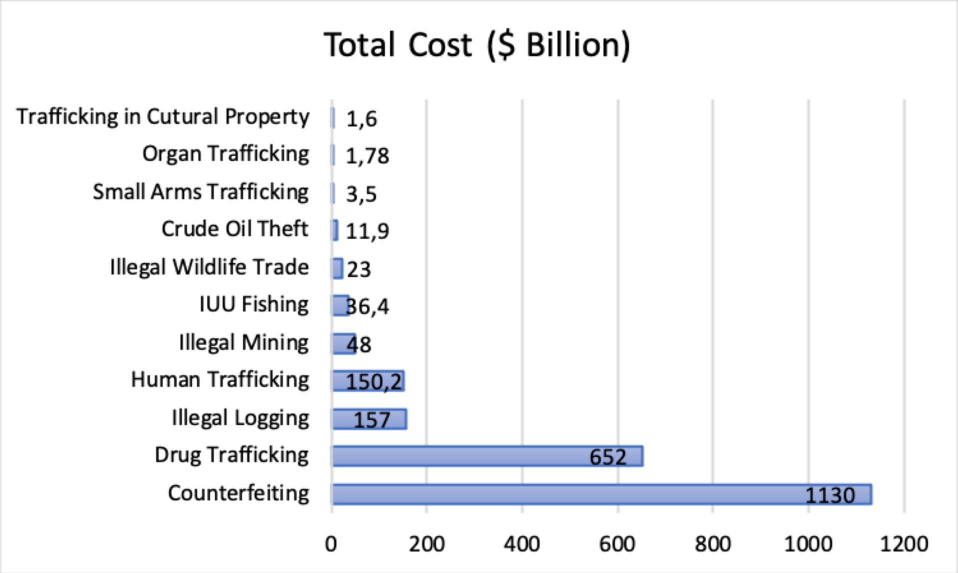


***Figure 1**. The distribution of transactional crime by type*

Money laundering activities may lead to corruption, bribery, and destabilized political institutions. When the amount of money is huge, the laundering process is happened with the involvement of corrupted politicians and government authorities. Most of the time, countries are badly affected with these types of relationships. Thus, the society and citizens are being unhappy due to the lack of adequate public services.

A great part of laundered money occurs within developing or non-developed countries. Initially, money that is laundered in these types of countries has small positive effect on economy, but then this money causes high inflation rates and volatile currency. The main reason behind the effect is that laundered money uses these countries only for cleaning routes before the final destination [9]. Specifically, money laundering affects not only government services of countries but also it negatively affects companies and financial institutions of these countries.

Money laundering is very critical for economies, therefore there are many legal actions against this kind of activities. If a money laundering scheme is detected or it is revealed to public, the scheme owners are penalized with a huge amount. According to OECD [10], US regulators convict ING bank for violating the sanctions, which is hold by US Office of Foreign Asset Control, against Iran and Cuba in 2012. The amount of fine was 619 billion USD. Another example is US Office of Foreign Asset Control (OFAC) that fined HSBC bank to 1.9 billion USD for laundering money which is originated form sanctioned countries such as Cuba, Iran, Libya, Myanmar, and Sudan in 2012. Moreover, the same money laundering activities of HSBC bank were punished by UK governments. Additionally, the sanctions were applied to HSBC and ING bank, JP Morgan with 88.3 million USD by the US Department of the Treasury. Actually, many financial institutions are increasingly aware of the significance of these harmful activities so that they conduct background screening of their clients carefully. Thus, money laundering has harmful effects on both institutions and societies. Particularly, the worst effect has occurred on financial sector since it depends highly on customer trust and the reputation of institutions.

2.2. Steps of Money Laundering

A typical money laundering process consists from three main parts which are placement, layering and integration [11]. Figure 2 shows a typical money laundering scheme [12]. Without proper precautions, the possibility of the success of the money laundering scheme is very high. In this paper, we explain existing precaution methods and we propose a new precaution method.

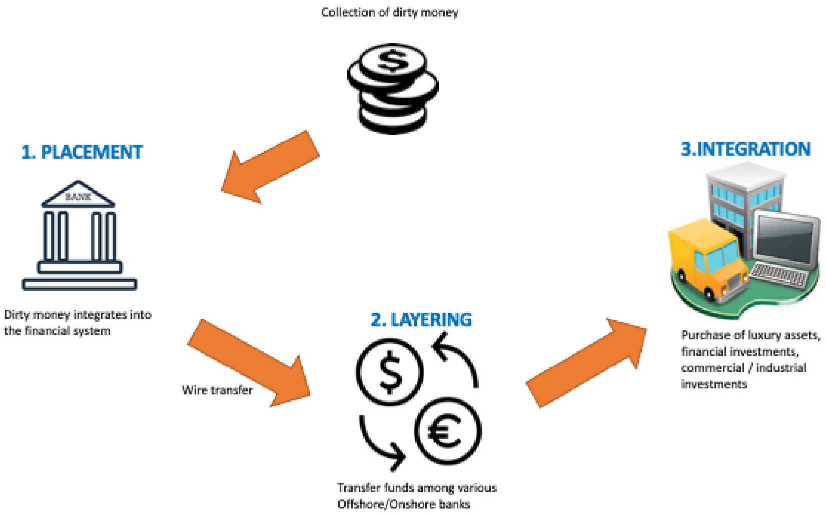

Collection of dirty money
1. PLACEMENT
BANK
Dirty money integrates into the financial system
Wire transfer
2. LAYERING
Transfer funds among various Offshore/Onshore banks
3.INTEGRATION
Purchase of luxury assets, financial investments, commercial / industrial investments

Figure 2. *A typical money laundering scheme*

Placement is the first step of a money laundering scheme. The main purpose of this step is putting illegally gathered money in financial systems without get caught on regulation limits. There are two methods for achieving these steps, namely primary deposit and secondary deposit. In primary deposit, the goal is to bypass regulation inspectors. It may be achieved by depositing certain money under the regulation border. For example, while this border in Australia is 15000 Euro, in United States of America it is 10000 USD. Another primary deposit method is taking control of some banks by buying or starting up a new bank in countries where regulations are not very strict, which are called off-shore countries.

In case of secondary deposit, money that will be laundered is divided and is handed out to legal people, who are aware or not aware of the money laundering scheme. They simply add money into banking systems. Other method used in secondary deposit is starting up a cash-intensive business, such as gastronomy, hotel sector, auctioneers, and galleries. Since services are obtained with money that is not tangible or the real value of services are unknown, the origin of money may be undetectable.

Layering is the second step of money laundering activities. Separating illegal activities and money is the main purpose of this step. In the layering, money is transferred through a complex network of transactions that are not directly related to money. Moreover, the transactions may contain buying and selling some assets. Total laundered money follows through many routes during transactions. Actually, dividing money into small portions makes its detection harder. Money laundering transactions use off-shore accounts that makes it harder to apply regulations because there is a lack of a common precaution and related bodies.

The last step of money laundering process is integration. The goal of this step is to clean the collected money from illegal activities. If the whole process of the last step is completed successfully, the laundered money may be used without being detected by financial agencies.

2.3. Machine Learning for Money Laundering

Machine learning algorithms have been used in the area of fraud detections for many years. Money laundering is mostly a part of fraud activities that has been changing continuously. Machine learning based fraud and money laundering detections mechanisms therefore need to be updated accordingly.

Existing fraud detection methods mostly rely on attribute-value data point-based machine learning techniques [13]. It is assumed that data points are independent and identically distributed in classical methods, therefore, they are far from being ideal for money laundering detections. Classical methods generally ignore relationships between data points. Moreover, these methods consider auto correlation in transaction data, which decreases the success rate of machine learning algorithms. Thus, graph representations of transactions appear to be more effective with machine learning algorithms. Specifically, graph-based machine learning approaches are used to reveal relationships between data points for money laundering detections. Sparse adjacent matrix representations are also used as the outlier matrix that reveals signs of fraud attempts. However, these methods may not explain the reason of fraud. D. Huang et al. propose a method to show the reason of fraud which uses a sparse similarity matrix and a feature matrix together [13].

There are other researches that use machine learning for money laundering detections. For instance, DeepWalk algorithm is used to detect money laundering with a transaction network graph that is converted into a lower dimensional latent presentation that is used with multiple machine learning algorithms, such as NB, Support Vector Machine (SVM) and Multilayer Perceptron (MLP) [14, 15]. Genetic algorithms are also used to detect money laundering [16] with many applications [17], which are inspired from nature by using random rules.

In another research, traditional asset laundering activities that contain laundering methods for coins are one of the new capitals of our age [18]. World Wide Web provides simpler and far more reachable methods than traditional laundering methods. For instance, the laundering method mentioned in [18] simply works as follows. The coins are circulated from wallet to wallet and the connection of coins with the source is eliminated to prevent traces. Although these methods perform complicated operations, graph-based solutions may be used to decode complex networks.

Money laundering activities may be detected with different kinds of solutions apart from graph-based machine learning methods. For example, the average, variance, kurtois, sparsity, discontinuity values, which are calculated with very known formulas, and transaction information obtained by examining bank accounts in the time and frequency domains may be used as features in machine learning algorithms to detect money laundering [19].

Rule-based methods are also popular in the industry. The principle of a rule-based method for money laundering detection is that there are some corner cases where not all money laundering activities may be detected by machine learning algorithms. It is emphasized that anti-money laundering systems should work in accordance with ethical values, have explainable results and be scalable. Moreover, there is no system that met these requirements. Therefore, a study that meets these criteria may be quite useful for financial intelligence units [20]. Although there are lots of machine learning approaches to detect money laundering in literature, dynamic behaviors of money laundering attempts reduce the success rate of the approaches with time so modified and new machine learning based approaches are needed for the high success ratio of money laundering detections.

3. MONEY LAUNDERING DETECTION WITH MACHINE LEARNING

We have proposed a machine learning approach on graphs to detect money laundering according to banking transaction data. To the best of our knowledge, graph-based approaches are more useful to evaluate transection data [14]. Our proposed solution uses Node2Vec algorithm during money laundering detections.

We use transaction data for money laundering detection as shown in Figure 3. Initially, we convert transaction data into a graph representation. Then, we apply Node2Vec algorithm on the graph representation to obtain a more meaningful dataset. In the next step, we split the dataset into test and training sets using K-Fold Cross Validation. Since the money laundering dataset contains a very small amount of money laundering instances, we reduce the imbalances in the transaction dataset. Then we classify our data using a classifier. Finally, we compare the results with and without using Node2Vec to detect money laundering.

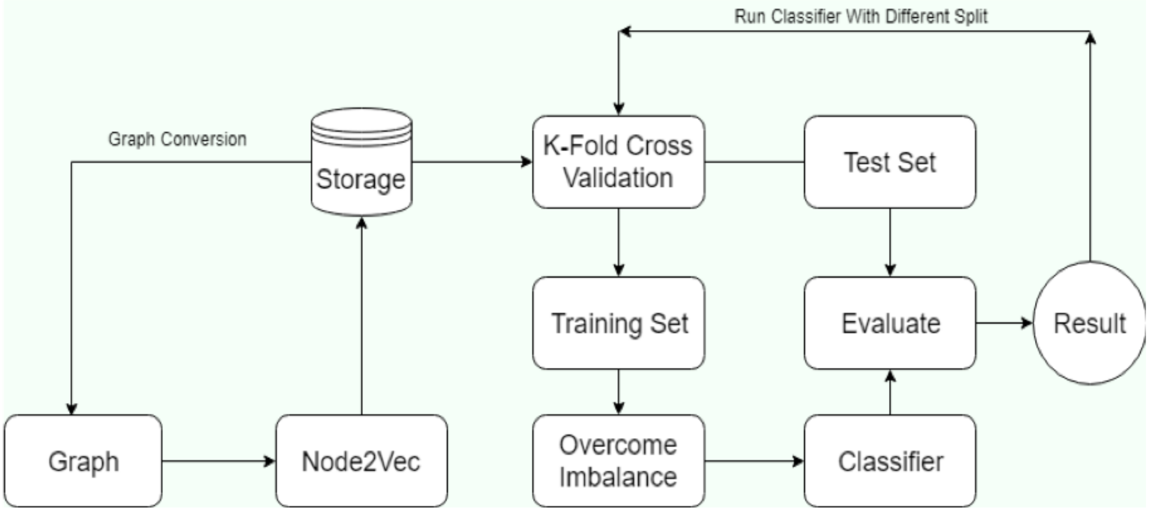


***Figure 3**. Machine learning approach for money laundering*

The first step of our machine learning approach for money laundering detections is to represent the dataset as a graph. We use *Algorithm 1* to convert the dataset into a graph. Then, we apply Node2Vec algorithm to the graph, which is given in [21]. Node2Vec is an algorithm, which converts graph data into lower dimensional space which maximizes likelihood of conserving neighbors of all nodes. The algorithm uses biased random walks.

Algorithm 1. Graph creation for Node2Vec

```
procedure CREATE-GRAPH(D):
    last-num ⟵ 1
    While end of D is not reached:
        If D.originAccount not in node file :
            var1 ⟵ last-num
            last-num ⟵ last-num + 1
        else:
            var1 ⟵ NODE-FILE(D.originAccount)
        If D.destAccount not in node file :
            var2 ⟵ last-num
            last-num ⟵ last-num + 1
        else:
            var2 ⟵ NODE-FILE(D.destAccount)
        WRITE-NODE(var1,var2)
end procedure
```

A simplified version of the output as a graph is given in Figure 4. In the figure, nodes represent bank accounts while edges are transactions between connected pairs.

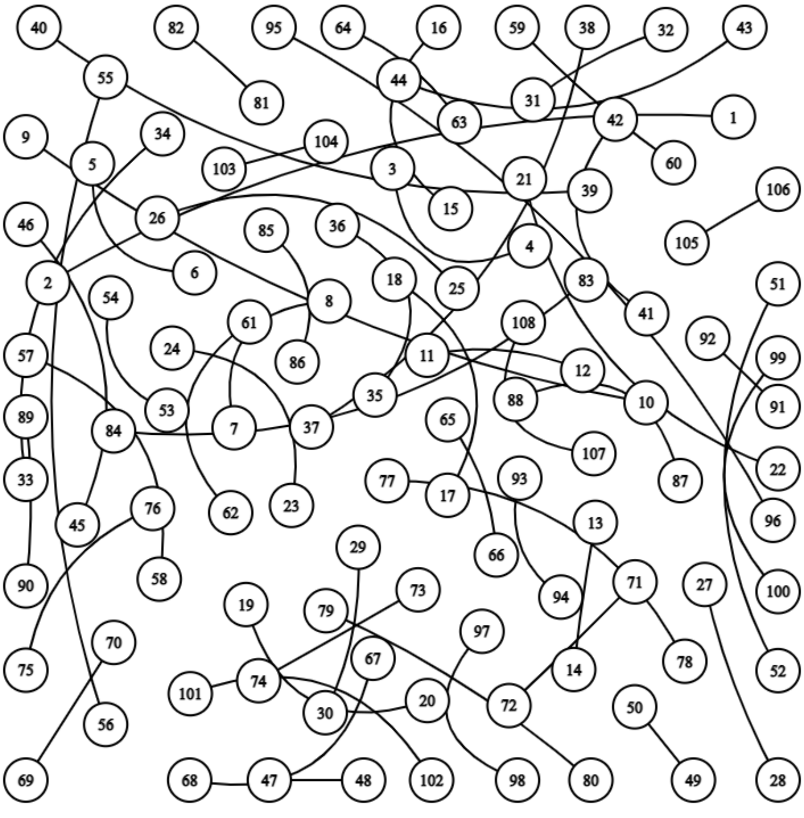


Figure 4*. A graph network that shows the relationships between bank accounts and transactions*

Random walk is a method which is used to discover the graph from a node or vertices. The algorithm starts with a given graph topology, G = (V, E). For instance, let *f* be the mapping function from a node to a feature that will give characteristic of data. Also, *f* is a matrix of size $|V| \times d$. For every source node $u \in V$, we define *Ns(u)* as a network neighborhood of node *u* generated through a neighborhood sampling strategy *S*. By using Skip-gram model, we preserve likelihood of neighbor nodes. We maximize the likelihood by using Equation (1) that is shown as follows.

$$max\left(\sum_{u\in V} logPr(Ns(u)|P(u))\right). \tag{1}$$

To make optimization problem traceable, we have two assumptions:

1. Conditional independence:

$$\Pr\big(Ns(u)\big|P(u)\big) = \prod_{n_i \in Ns(u)} \Pr\,(n_i|P(u)). \tag{2}$$

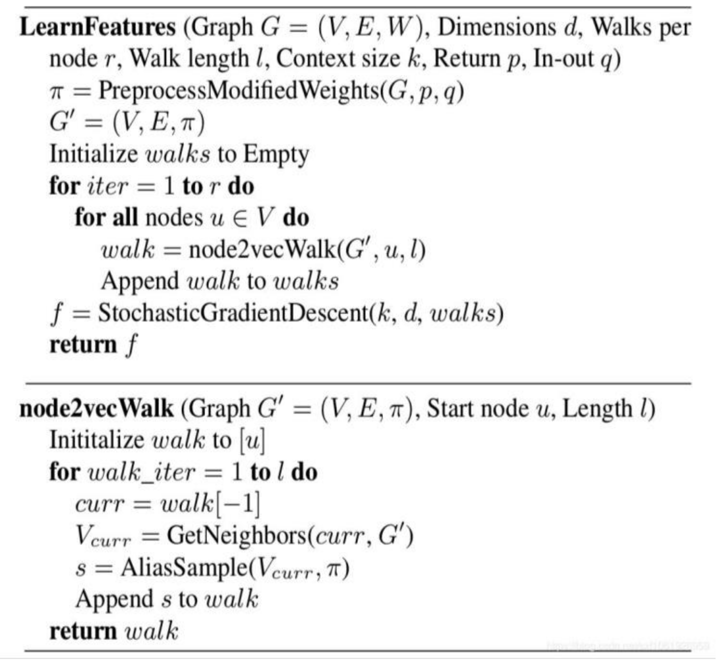
2. Symmetry in feature space. A source node and neighborhood node have a symmetric effect over each other in feature space.

$$\Pr\big(n_i\big|f(u)\big) = \frac{\exp\,(f(n_i).f(u))}{\sum_{v \in V} \exp(f_v)\,.f(u)}\,. \tag{3}$$

Node2Vec algorithm takes the following parameters:

- Number of random walks: Number of random walks from each node in the graph representation.
- Walk length: Length of every random walk.
- *P*: Return parameter which is equal to likelihood of revisiting *a* node in the walk.
- *Q*: In-out parameter which helps to formulize random walks. If *q<1*, random walks work like Depth First Sampling. If *q>1*, random walks work like Breath First Sampling.
- Skip-gram parameters (dimension, context size).

Algorithm 2. Node2Vec algorithm [21].



```
LearnFeatures (Graph $G = (V, E, W)$, Dimensions $d$, Walks per
  node $r$, Walk length $l$, Context size $k$, Return $p$, In-out $q$)
  $\pi$ = PreprocessModifiedWeights($G, p, q$)
  $G' = (V, E, \pi)$
  Initialize $walks$ to Empty
  for $iter = 1$ to $r$ do
    for all nodes $u \in V$ do
      $walk$ = node2vecWalk($G', u, l$)
      Append $walk$ to $walks$
  $f$ = StochasticGradientDescent($k$, $d$, $walks$)
  return $f$

node2vecWalk (Graph $G' = (V, E, \pi)$, Start node $u$, Length $l$)
  Inititalize $walk$ to $[u]$
  for $walk\_iter = 1$ to $l$ do
    $curr = walk[-1]$
    $V_{curr}$ = GetNeighbors($curr, G'$)
    $s$ = AliasSample($V_{curr}, \pi$)
    Append $s$ to $walk$
  return $walk$
```

We use Word2vec algorithm to transfer the collection of random walks. Specifically, Word2vec just calculates cosine distance between words. The bigger the result means bigger the correlation between two the words. Relationships between nodes in the graph are represented with Word2vec algorithm. Additionally, we use Unsupervised Network Representation Learning (UNRL) like in [14] for Node2Vec algorithm. The steps of Node2Vec algorithms given in *Algorithm 2*. Briefly, random walks are generated and are fed into Word2vec algorithm according to the walk-length parameter of the algorithm that is used for Node2Vec.

One of the major problems in real-world object detection and classification tasks is imbalanced datasets, which are represented with either majority of data or minority of data. These imbalances decrease the probability of successful classifications during learning processes of machine learning algorithms [22]. We select Node2Vec algorithm to create features. The goal of the feature creation is to decrease the processing time of a classification algorithm and is to increase the accuracy of classifications. Particularly, the probability of a successful classification in imbalanced datasets with general classification methods such as Naïve Bayes, Support Vector Machine (SVM) and Multilayer Perceptron (MLP), is very limited. Additionally, Node2Vec algorithm does not solve the imbalanced dataset problem, therefore, we use SMOTE and ADASYN algorithms to overcome imbalanced data in our work.

In our approach, we split the dataset into training and test sets during the preprocessing step. Then, we use K-Fold algorithm for cross validation. K-Fold Cross Validation is used for elimination of the skewed distribution during the splitting out dataset into training and test sets as in [23, 24]. Next, we use Gaussian Naïve Bayes method for classifications since it provides efficient results for datasets, which are similar to money laundering datasets.

We select confusion matrix and K-Fold Cross Validation to evaluate the proposed approach for money laundry detections. Binary classification confusion matrix consists of four classes:

1. **True Positives**: Number of true classifications for positive class.
2. **False Positives**: Number of false classifications for positive class.
3. **False Negatives**: Number of false classifications for negative class.
4. **True Negatives**: Number of true classifications for negative class.

There are two metrics that we use in confusion matrix to characterize results, namely sensitivity and specificity. Sensitivity is the ratio of number of true positive to number of real positives. Specificity is the ratio of number of true false to the number of real false. If we represent the graph as graph sensitivity and 1-specificity, Receiver Operating Characteristic (ROC) curve and the Area Under this Curve (AUC) scores are used to measure the classification quality. The bigger AUC score means better classification result we have.

4. PERFORMANCE EVALUATION

4.1. Dataset

Since financial transactions are confidential, we have used a freely available dataset, Kaggle dataset, to test our solution for money laundry detections. The dataset contains synthetic banking transactions generated by PAYSIM [25] synthetic banking data generator. Kaggle dataset is created for many financial purposes, such as fraud detection and money laundering. The dataset consists of approximately seven million transactions. We have used only 660000 of them, which represent money laundry transactions. We labeled 268 transactions to be suspicious for money laundering. We also used the following columns in the dataset in our experiments:

Step: Maps of unit time in real world.

Type: CASH-IN, CASH-OUT, DEBIT, PAY-MENT and TRANSFER.

Amount: Amount of transaction in local currency.

nameOrig: Customer number which starts transaction.

oldbalanceOrg: Initial balance before the transaction.

newbalanceOrig: Customer's balance after the transaction.

nameDest: Recipient of transaction's number.

oldbalanceDest: Initial recipient balance before the transaction.

newbalanceDest: Recipient's balance after the transaction.

isFraud: Fraud or not.

isFlaggedFraud: Money laundering or not according our heuristics.

We used Node2Vec to represent network relations in the dataset. In our representation, customers are nodes and transactions are edges. Node2Vec parameter dimension is set to be 80 and walk length set to be 65. Other parameters of algorithm have default values in our solution.

4.2. Preprocessing

During the preprocessing step, we consider two issues, randomness and imbalances. We split the dataset into training and test sets. These sets contain both positive and negative samples. Our dataset includes 123 negative samples. Actually, if there are lots of negative samples in the training set, classification results may be biased. Therefore, we used K-Fold Cross Validation to validate the dataset. We choose k to be 5 as in Table 1.

Table 1. *K-Fold Cross Validation for our data*

	Number of Suspicious Transactions	Number of Non-suspicious Transactions
Split 1		
Training Set	201	500929
Test Set	57	125216
Split 2		
Training Set	223	500907
Test Set	45	125238
Split 3		
Training Set	218	500912
Test Set	50	125233
Split 4		
Training Set	213	500918
Test Set	55	125227
Split 5		
Training Set	217	500914
Test Set	51	125231

Skews in the training set may overcame with under-sampling majority class or oversampling minority class. We use oversampling because the ratio of minority is very small. Combining the under sampled majority class with minority class represent very small portion of data. This circumstance gives quite skewed result, therefore, we selected oversampling methods, which are SMOTE and ADASYN.

Synthetic Minority Oversampling Technique (SMOTE) creates synthetic random data from minority class. The algorithm starts by visiting each node in minority class. While nodes are visited, k-nearest node that belongs current node is selected. Then some synthetic nodes are generated and added in the middle of the current node and the k-nearest node. Finally, nodes from different classes are merged to one class. Applying oversampling methods during K-Fold Cross Validation is expected to be more suitable rather than applying oversampling methods before K-Fold Cross Validation to avoid over optimization and overfitting [26]. In our experiments, we applied K-Fold Cross Validation for oversampling.

Adaptive Synthetic (ADASYN) sampling method creates synthetic data for minority class in training set. Like SMOTE, it uses K-NN. ADASYN locates new nodes randomly while creating synthetic nodes. In SMOTE, new nodes are distributed linearly. We also analyze the dataset with ADASYN to investigate oversampling ratio for minority class as in [27].

4.3. Classification Process

We used K-NN classifier, Naïve Bayes classifier, Random Forest classifier in our experiments. K-NN classifies unlabeled data according to k different neighbors by considering their distances [23]. The normal version of K-NN uses Euclidian distance that means it depends only on the number of neighbors. The k must be selected carefully due to handling labeled and unlabeled data appropriately. Best classification is provided with k that is equal to 5, 6, and 7 [28]. Therefore, we selected k to be 6 in our experiments. The algorithm of K-NN is shown below.

Algorithm 3. K-NN Classifier

```
1. Given a training set X = {(x1, y1), . . . ,(xN , yN )}, where xi ∈ X represents the i'th training sample, yi ∈ {ω1, ω2, . . . , ωc} represents the class label of the i'th training sample, N represents the total number of samples in the training set, and c is the total number of classes.
2. Choose the value of k.
3.  for all (Training samples (i = 1, 2, . . . , N)) do
4.    Calculate the distance between the testing sample (xtest) and the training samples (xi), as follows, $d_i = \sum_{i=1}^{N} 2(x_i - x_{test})^2$
5. end for
6. Select the nearest k training samples, i.e., minimum k distances.
7. Assign the class which has the most samples among the k nearest samples to the testing sample.
```

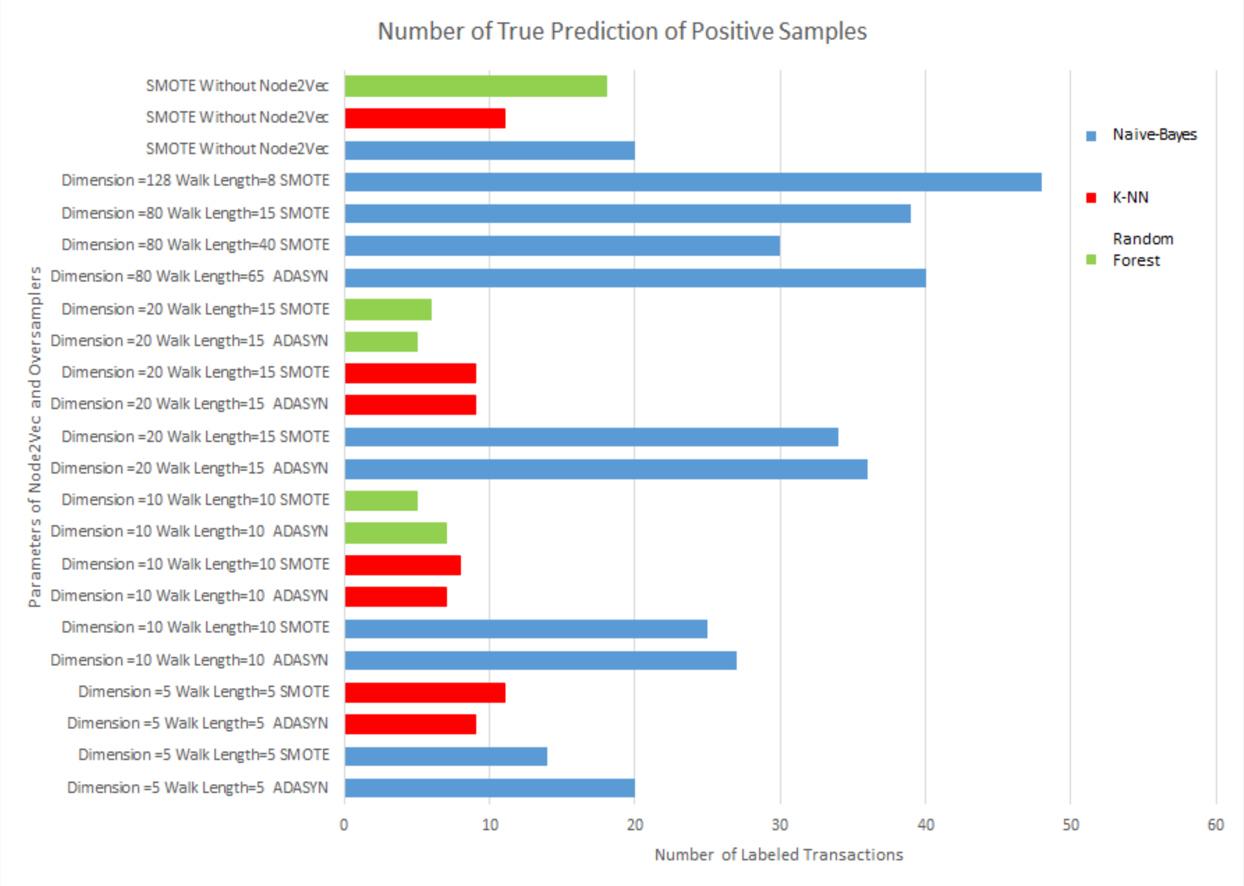


Figure 5*. True predictions of positive samples*

Naive Bayes algorithm is based on Bayes theorem. This theorem ended up with formula that describes probability of an event A when event B happened as follows. P(A) is probability of event A, P(B) is probability of event B and P(B|A) is probability of an event B when event A is happened

$$P(A|B) = \frac{P(B|A)P(A)}{P(B)}. \tag{4}$$

Naive-Bayes classifier calculates every likelihood for every features of different class label in the training set. At the last step, features of data are compared with likelihood values of the training set and a decision is made related to unlabeled data. Pseudocode of Naïve-Bayes algorithm is given bellow.

Algorithm 4. Naïve-Bayes Classifier

```
Read training set.
For each feature of each class label in training set Calculate P(Fi|Ci) where Fi is probability of given ith feature is observed and Ci is probability of ith class label is observed
end for
For each unlabeled data, detect the class which maximizes the likelihood of current data
end for
```

Random forest classifier uses *n* different decision trees for classification according to the importance of features. In this classifier, the number of features should be smaller than the number of random classifiers.

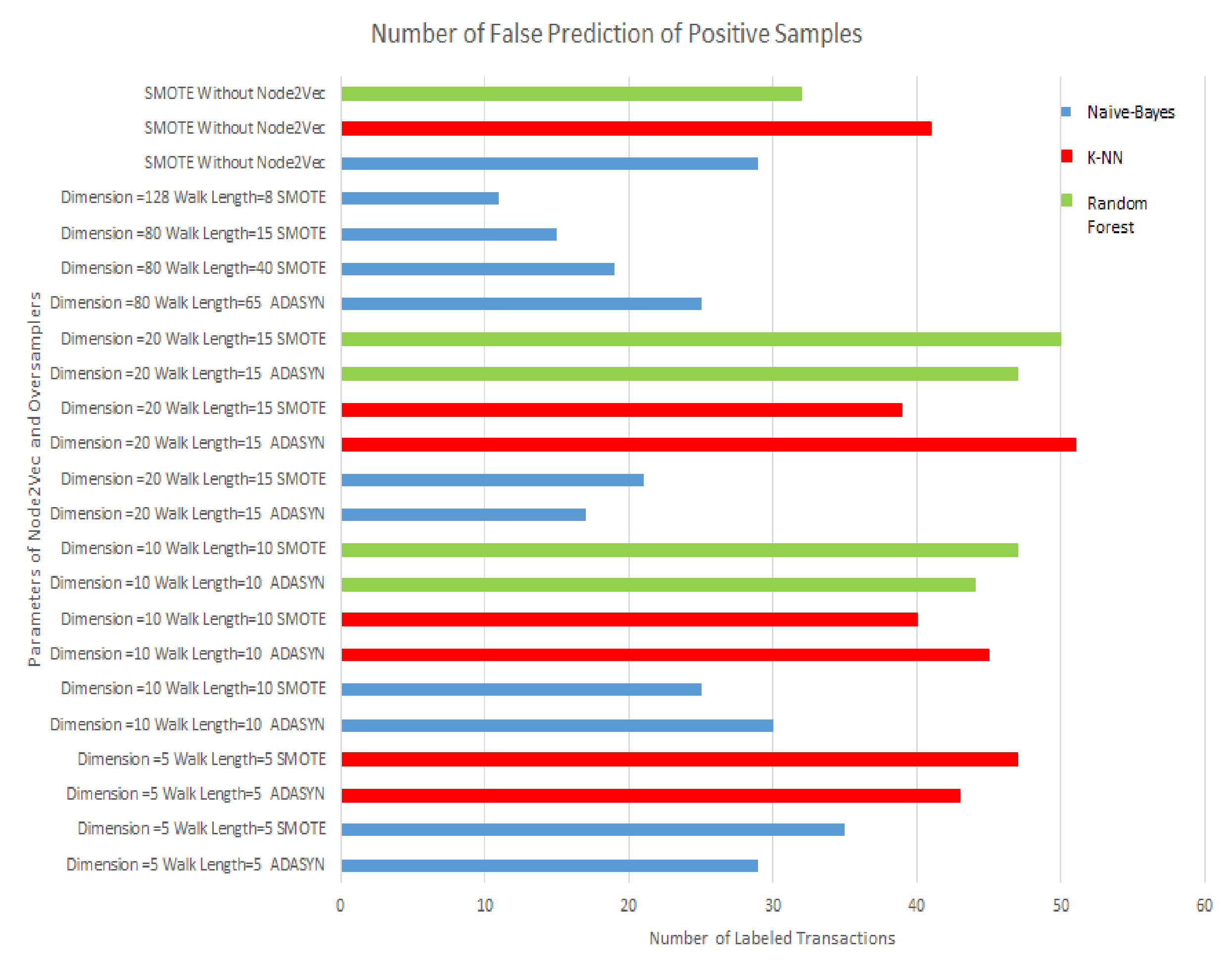


Figure 6. *False predictions of positive samples*

4.4. Analyses of Results

We evaluated the proposed approach according to four criteria. We specified parameters of each criterion, oversamples, and classification methods. Our criteria are listed below:

- Number of true predictions of positive samples
- Number of false predictions of positive samples
- Number of false predictions of negative samples
- Number of true predictions of negative samples

We show true prediction results of positive samples for our dataset in Figure 4. In these results, Gaussian Naïve-Bayes classifier provides the best classification accuracy for each parameter of Node2Vec and for each oversampling method used. Results also show that updating the parameters of Node2Vec such as dimension and random walk length may provide better classification results for Gaussian Naïve-Bayes classifier. On the other hand, other classifiers are not affected considerably with the parameter updates and methods for oversampling.

We also analyzed the dataset without using Node2Vec algorithm. Specifically, we directly implied classification methods without Node2Vec to our dataset. The analyses results show that true positives with K-NN and Random Forest Classifiers are better than applying Node2Vec algorithm. Moreover, Gaussian Naïve-Bayes classifier without Node2Vec provides worse accuracy in this case.

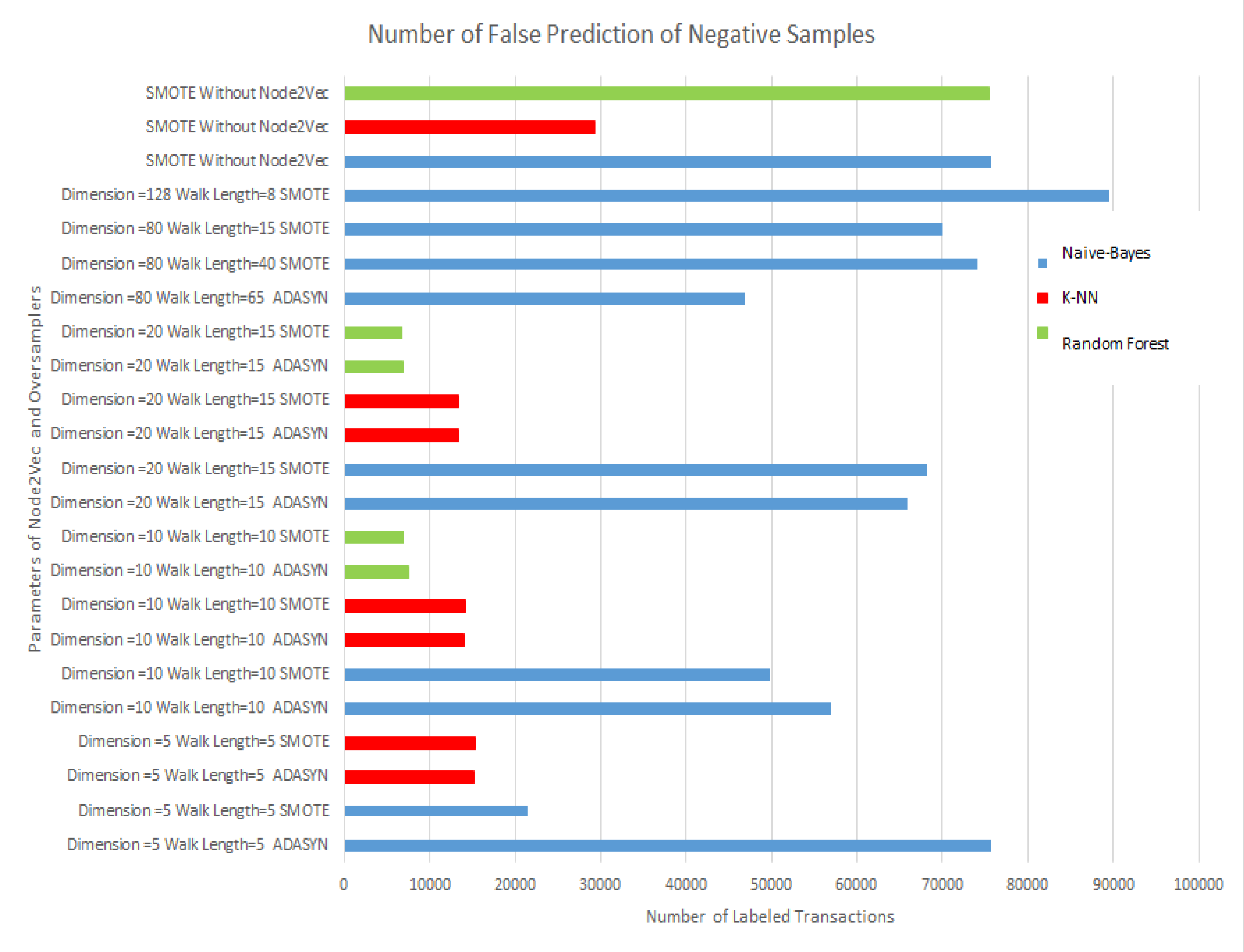


Figure 7. False predictions of negative samples

The number of true positive samples alone may not represent the success of the methods. Therefore, all the four options need to be analyzed. Figure 5 shows false predictions of positive samples of our analyses in the dataset. The results show that Gaussian Naïve-Bayes classifier provides minimum false predictions whereas K-NN and Random Forest classifiers higher prediction values.

Gaussian Naïve-Bayes classifier has worst results than K-NN and Random Forest classifiers for false predictions of negative samples as shown in Figure 6. Particularly, results of classification processes may be changed by updating parameters of Node2Vec which may provide different results for Gaussian Naïve-Bayes classifier while other classification methods remain the same for different parameters. We selected different values for dimension and walk length due to computational requirements of Node2Vec as in Figure 6. Analyses results show that Gaussian Naïve-Bayes classifier is more sensitive to Node2Vec parameters than K-NN and Random Forest classifiers. When we consider true predications of negative samples, experimental results in Figure 7 show that the results are opposite of false predictions of negative samples.

Classification results do not help us to decide effects of oversampling, which are significant for money laundering detections from a dataset. Specifically, experimental accuracy of ADASYN and SMOTE with same parameters are inconsistent. In some cases, ADASYN has better results while in some other cases SMOTE has better results. On the other hand, we observe that ADASYN provides better results than SMOTE in general. Moreover, applying classification methods without Node2Vec gives quite low accuracy except for K-NN classifier.

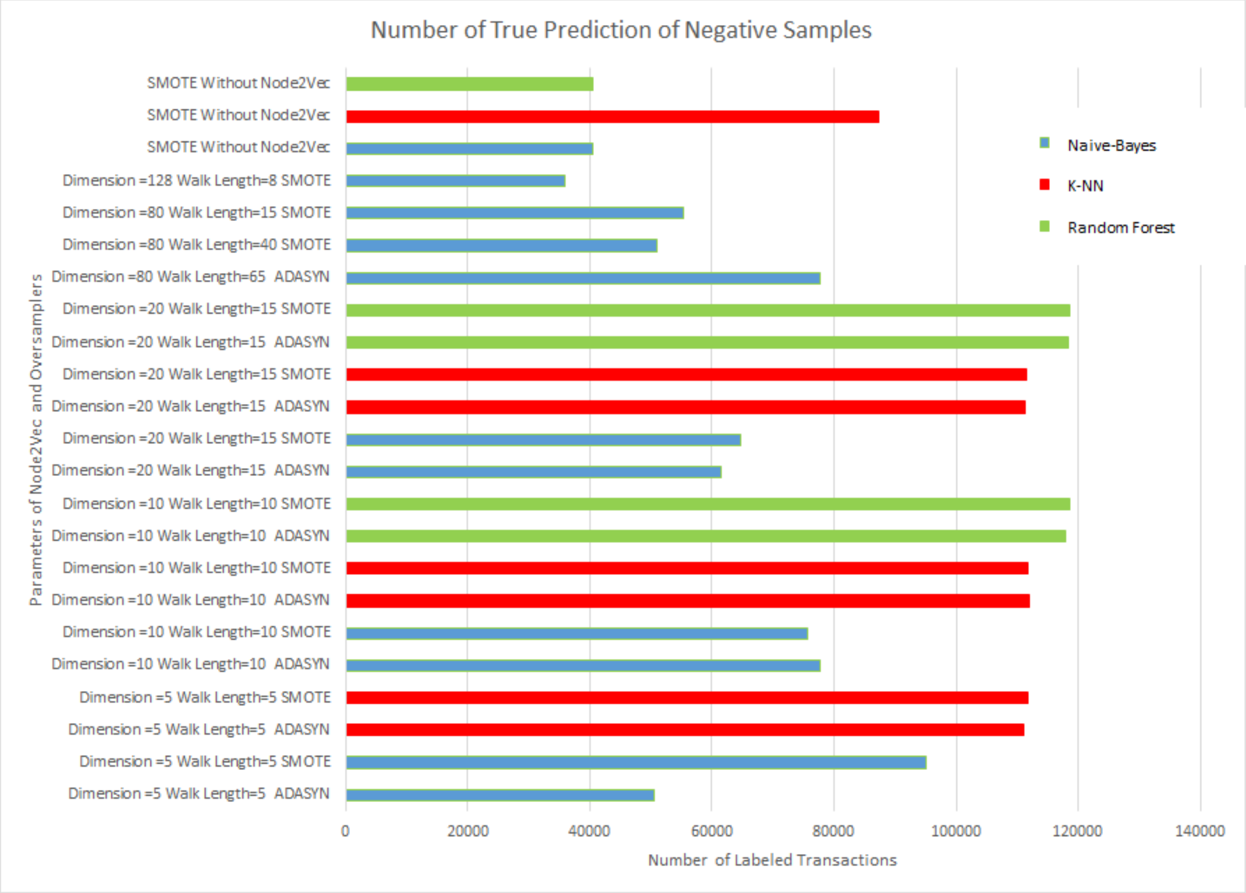


***Figure 8.** True predictions of negative samples*

When we look at overall results, we observe that directly applying classification methods may provide better results in high skewed data. If this type of conditions occur, it is observed that classifications of minority class samples are inaccurate. In money laundering like datasets minority samples are significant for detections, which samples represent fraudulent transactions. Specifically, minority samples are more important than majority samples for money laundering detections. Therefore, applying Node2Vec when minority samples are more important than majority samples provide better classification results.

We also provide confusion matrix for our experimental results to clarify details. Specifically, Tables from 2 to 24 contain confusion matrix that show more detailed results of our experiments. In Gaussian Naïve Bayes classifier, we used SMOTE and ADASYN for oversampling. In Tables 2, 3, 4 and 5 parameters of Node2Vec were analyzed according to default values, where parameters are dimension and walk-length. We selected the values of dimension and walk length to be 5. Tables 4 and 5 contain classification results for K-Neighborhood with ADASYN and SMOTE, where both negative and positive samples are slightly lower than the classification results of Naïve Bayes with ADASYN and SMOTE.

Table 2*. Result of applying Gauissian Naïve Bayes with Node2Vec and ADASYN oversampler*

	FALSE PREDICTION	TRUE PREDICTION
POSITIVE SAMPLES	29	20
NEGATIVE SAMPLES	75782	50514

Table 3*. Result of applying Gauissian Naïve Bayes with Node2Vec and SMOTE oversampler*

	FALSE PREDICTION	TRUE PREDICTION
POSITIVE SAMPLES	35	14
NEGATIVE SAMPLES	21412	94973

Table 4*. Result of applying K-Neighborhood classifier with Node2Vec and ADASYN oversampler*

	FALSE PREDICTION	TRUE PREDICTION
POSITIVE SAMPLES	43	9
NEGATIVE SAMPLES	15201	111027

Table 5*. Result of applying K-Neighborhood classifier with Node2Vec and SMOTE oversampler*

	FALSE PREDICTION	TRUE PREDICTION
POSITIVE SAMPLES	47	11
NEGATIVE SAMPLES	15414	110867

To be able to show the performance of Node2Vec with different values of the parameters, we initially incremented the values of dimension and walk-length to be 10. Tables 6 and 7 show the results after these changes of parameters. Specifically, when the values of dimension and walk-length were increased from 5 to10, the number of positive samples increases while the number of negative samples decreases.

Table 6*. Result of applying Gauissian Naïve Bayes with Node2Vec and ADASYN oversampler*

	FALSE PREDICTION	TRUE PREDICTION
POSITIVE SAMPLES	30	27
NEGATIVE SAMPLES	56928	77825

***Table 7**. Result of applying Gauissian Naïve Bayes with Node2Vec and SMOTE oversampler*

	FALSE PREDICTION	TRUE PREDICTION
POSITIVE SAMPLES	25	25
NEGATIVE SAMPLES	49847	75611

When we compare K- Neighborhood classifier and Gaussian Naïve Bayes, we observe that Gaussian Naïve Bayes provides better results for positive samples while K- Neighborhood classifier provides is more convenient for classification of negative samples. Tables 6 and 7 show performance results for Gaussian Naïve Bayes and Tables 8 and 9 show the results for K- Neighborhood classifier.

***Table 8**. Result of applying K-Neighborhood Classifier with Node2Vec and ADASYN oversampler*

	FALSE PREDICTION	TRUE PREDICTION
POSITIVE SAMPLES	44	7
NEGATIVE SAMPLES	14051	111965

***Table 9**. Result of applying K-Neighborhood Classifier with Node2Vec and SMOTE oversampler*

	FALSE PREDICTION	TRUE PREDICTION
POSITIVE SAMPLES	40	8
NEGATIVE SAMPLES	14258	111673

We also analyzed Random Forest classifier for ADASYN and SMOTE with Node2Vec. Tables 10 and 11 presents the performance results of Random Forest classifier. The results show that Random Forest classifier provide better performance negative samples than both K-Neighborhood classifier and Gaussian Naïve Bayes. On the other hand, Random Forest classifier provides poor performance than the other classifiers.

***Table 10**. Result of applying Random Forest Classifier with Node2Vec and ADASYN oversampler*

	FALSE PREDICTION	TRUE PREDICTION
POSITIVE SAMPLES	44	7
NEGATIVE SAMPLES	14051	111965

***Table 11**. Result of applying Random Forest Classifier with Node2Vec and SMOTE oversampler*

	FALSE PREDICTION	TRUE PREDICTION
POSITIVE SAMPLES	47	5
NEGATIVE SAMPLES	6998	118705

We analyzed the behavior of the three classifiers by increasing values of the parameters. We set the value of dimension to be 20 and the value of walk-length to 15. Tables 12 - 17 contain experimental results for this setup. It was observed that the number of positive samples increases while the number of negative samples decreases for Gaussian Naïve Bayes classifier with Node2Vec for both ADASYN and SMOTE over-samplers. On the other hand, K-Neighborhood classifier with Node2Vec has very small effects on classifications for both ADASYN and SMOTE. It was also observed that Random Forest classifier is less affected from the update of the parameters.

***Table 12**. Result of applying Gaussian Naïve Bayes Classifier with Node2Vec and ADASYN oversampler*

	FALSE PREDICTION	TRUE PREDICTION
POSITIVE SAMPLES	17	36
NEGATIVE SAMPLES	65877	61427

***Table 13**. Result of applying Gaussian Naïve Bayes Classifier with Node2Vec and SMOTE oversampler*

	FALSE PREDICTION	TRUE PREDICTION
POSITIVE SAMPLES	21	34
NEGATIVE SAMPLES	68194	68262

***Table 14**. Result of applying K-Neighborhood Classifier with Node2Vec and ADASYN oversampler*

	FALSE PREDICTION	TRUE PREDICTION
POSITIVE SAMPLES	51	9
NEGATIVE SAMPLES	13443	111233

***Table 15**. Result of applying K-Neighborhood Classifier with Node2Vec and SMOTE oversampler*

	FALSE PREDICTION	TRUE PREDICTION
POSITIVE SAMPLES	39	9
NEGATIVE SAMPLES	13381	111590

***Table 16**. Result of applying Random Forest Classifier with Node2Vec and ADASYN oversampler*

	FALSE PREDICTION	TRUE PREDICTION
POSITIVE SAMPLES	47	5
NEGATIVE SAMPLES	6943	118431

***Table 17**. Result of applying Random Forest Classifier with Node2Vec and SMOTE oversampler*

	FALSE PREDICTION	TRUE PREDICTION
POSITIVE SAMPLES	50	6
NEGATIVE SAMPLES	6772	118705

Since Gaussian Naïve Bayes is the most sensitive classifier for parameter updates, we further updated the parameters and tested Gaussian Naïve Bayes classifier to observe the whole behavior of Node2Vec. We set the value of dimension to be 80 and the value of walk-length first to be 65 and then to be 15. Experimental results are given on next two tables. These experimental results show that Gaussian Naïve Bayes classifier has better results for lower walk-length values. Therefore, we further decrease the value of walk-length to be 8 and increase the value of dimension to be 128 as shown in Tables 18 - 23. Experimental results of this setup are provided in Tables 20 and 21. Overall analyses of Node2Vec with ADASYN and SMOTE show that higher value of dimension provides better predictions.

***Table 18**. Result of applying Gauissian Naïve Bayes with Node2Vec and ADASYN oversampler*

	FALSE PREDICTION	TRUE PREDICTION
POSITIVE SAMPLES	25	40
NEGATIVE SAMPLES	46928	77825

***Table 19**. Result of applying Gauissian Naïve Bayes with Node2Vec and SMOTE oversampler*

	FALSE PREDICTION	TRUE PREDICTION
POSITIVE SAMPLES	19	30
NEGATIVE SAMPLES	74056	51107

***Table 20**. Result of applying Gauissian Naïve Bayes with Node2Vec and ADASYN oversampler*

	FALSE PREDICTION	TRUE PREDICTION
POSITIVE SAMPLES	15	39
NEGATIVE SAMPLES	69981	55408

***Table 21**. Result of applying Gauissian Naïve Bayes with Node2Vec and ADASYN oversampler*

	FALSE PREDICTION	TRUE PREDICTION
POSITIVE SAMPLES	11	48
NEGATIVE SAMPLES	89558	35832

***Table 22**. Result of applying Gauissian Naïve Bayes without Node2Vec*

	FALSE PREDICTION	TRUE PREDICTION
POSITIVE SAMPLES	29	20
NEGATIVE SAMPLES	75782	40514

***Table 23**. Result of applyıng K-Neigborhood Classifier without Node2Vec*

	FALSE PREDICTION	TRUE PREDICTION
POSITIVE SAMPLES	41	11
NEGATIVE SAMPLES	29382	87369

Finally, we analyzed the classifiers without applying Node2Vec. The analyses results are given in Tables 22, 23, and 24. The results show that directly applying the classification methods may provide better results for highly skewed data. On the other hand, it is observed that classifications of minority classes like in money laundering detections, where fraudulent transactions occur, is significant. In other words, using machine learning algorithms with skewed data may provide misleading results, therefore, classifying minority classes is more significant for the detection of money laundry.

***Table 24**. Result of applying Random Forest Classifier without Node2Vec*

	FALSE PREDICTION	TRUE PREDICTION
POSITIVE SAMPLES	32	18
NEGATIVE SAMPLES	75544	40514

Experimental evaluations show that applying Node2Vec when the minority of data is more important than the majority of data provides better results for classifications. Additionally, parameter tunning may increase the correct predictions. Thus, applying Node2Vec will help to detect money laundering cases in a more precise manner.

5. CONCLUSION AND FUTURE WORK

Money laundering is a significant topic for societies, governments, and economies. Therefore, automated money laundering detections of financial transactions have been a crucial research topic, where different methods have been applied. In this research, we considered banking transactions from a dataset to detect money laundering transactions. Fraudulent transactions are a small part of genuine transactions so that it is, most of the time, impossible to detect money laundering transactions with conventional fraud detection methods.

In this research, we used unsupervised network representations with Node2Vec for classification purposes of money laundering transactions. Node2Vec uses random walks through a graph, which is useful to apply with machine learning algorithms. We also used SMOTE and ADASYN oversampling techniques for classifications of banking transactions. Specifically, we analyzed true positives, false positives, false negatives, and true negatives over a banking transactions dataset that contains money laundering transactions. The analyses results show that applying Node2Vec algorithm provides better results for classifications of money laundering transactions. To the best of our knowledge, this is the first research that applies Node2Vec algorithm to represent unsupervised networks for money laundry detections on banking transactions.

We have been working on different datasets to test our approach for detecting money laundering as future works. Moreover, we plan to represent financial transactions as a complex network to be able to investigate money laundering transactions with complex networks analyses methods. Furthermore, we plan to combine this work with the complex networks analyses results.

CONFLICTS OF INTEREST

No conflict of interest was declared by the authors.

REFERENCES

[1] Pourhabibi, T., Ong, K-L., Kam, H. B., and Boo, L. Y., “Fraud detection: A systematic literature review of graph-based anomaly detection approaches”, Decision Support Systems, 133: 113303, (2020).

[2] Zhou, H., Sun, G., Fu, S., Wang, L., Hu, J., and Gao, Y., “Internet Financial Fraud Detection Based on a Distributed Big Data Approach With Node2vec”, IEEE Access, 9: 43378-43386, (2021).

[3] Jabbar, M. A., Deekshatulu, B. L., and Chandra, P., “Classification of Heart Disease Using K- Nearest Neighbor and Genetic Algorithm”, Procedia Technology, 10: 85-94, (2013).

[4] Taheri, S., and Mammadov, M., “Learning the naive Bayes classifier with optimization models”, International Journal of Applied Mathematics and Computer Science, 23(4): 787–795, (2013).

[5] Drezewski, R., Sepielak, J., and Filipkowski, W., “The application of social network analysis algorithms in a system supporting money laundering detection”, Information Sciences, 295: 18-32, (2015).

[6] Bridle, J. S., “Probabilistic Interpretation of Feedforward Classifi-cation Network Outputs, with Relationships to Statistical Pattern Recognition”, Neurocomputing—Algorithms, Architectures and Applications, F. Fogelman-Soulie and J. Herault, eds., NATO ASI Series F68, Berlin, Springer-Verlag, 227-236, (1989).

[7] https://www.unodc.org/unodc/en/money-laundering/globalization.html. Access date: 12.11.2019

[8] https://blog.revolut.com/money-laundering-what-is-it-and-why-should-we-care/. Access date: 20.11.2019

[9] https://www.fatf-gafi.org/faq/moneylaundering/. Access date: 18.11.2019

[10] Internet: OECD (2014), “Combating money laundering”, in Illicit Financial Flows from Developing Countries: Measuring OECD Responses, OECD Publishing, Paris, (2014). DOI: https://doi.org/10.1787/9789264203501-5-en. Access date: 20.11.2019

[11] Schneider, F., and Windischbauer, U., “Money Laundering: Some Facts”, European Journal of Law and Economics, 26, 387-404, (2008).

[12] http://www.antimoneylaundering.gov.ie/en/AMLCU/Pages/. Access date: 2.12.2019

[13] Huang, D., Mu, D., Yang, L., and Cai, X., "CoDetect: Financial Fraud Detection with Anomaly Feature Detection", IEEE Access, 6, 19161-19174, (2018).

[14] Wagner, D., “Latent representations of transaction network graphs in continuous vector spaces as features for money laundering detection”, Becker, M. (Hrsg.), SKILL 2019 - Studierendenkonferenz Informatik, Gesellschaft für In-formatik e.V., Bonn, 143-154, (2019).

[15] Internet: Khosla, M., Anand, A., and Setty, V., “A Comprehensive Comparison of Unsupervised Network Representation Learning Methods”, CoRR abs/1903.07902/, (2019). URL: http://arxiv.org/abs/1903.07902.

[16] Sadgali, I., Sael, N., and Benabbou, F., “Performance of machine learning techniques in the detection of financial frauds”, Procedia Computer Science, 148, 45 – 54, (2019).

[17] Dionysios, S. D., “Fighting money laundering with technology: A case study of Bank X in the UK”, Decision Support Systems, 105, 96-107, (2018).

[18] Wu, J., Liu, J., Chen, W., Huang, H., Zheng, Z., and Zhang, Y., "Detecting Mixing Services via Mining Bitcoin Transaction Network with Hybrid Motifs", IEEE Transactions on Systems, Man, and Cybernetics: Systems, 1-13, (2021).

[19] Ketenci, U. G., Kurt, T., Önal, S., Erbil, C., Aktürkoğlu, S., and İlhan, H. Ş., "A Time-Frequency Based Suspicious Activity Detection for Anti-Money Laundering", IEEE Access, 9: 59957-59967, (2021).

[20] Bellomarini, L., Laurenza, E., and Sallinger, E., “Rule-based anti-money laundering in financial intelligence units: experience and vision”, In International Joint Conference on Rules and Reasoning, (2020).

[21] Grover, A., and Leskovec, J., “Node2vec: Scalable feature learning for networks”, In ACM SIGKDD, 855–864, (2016).

[22] Khan, S. H., Hayat, M., Bennamoun, M., Sohel, F. A., and Togneri, R.,” Cost-Sensitive Learning of Deep Feature Representations from Imbalanced Data”, IEEE Transactions on Neural Networks and Learning Systems, 29(8): 3573-3587, (2018).

[23] Ghorbanzadeh, O., Rostamzadeh, H., Blaschke, T., Gholaminia, K., Aryal, J., “A new GIS-based data mining technique using an adaptive neuro-fuzzy inference system (ANFIS) and k-fold cross-validation approach for land subsidence susceptibility mapping”, Natural Hazards 94, 497–517, (2018). DOI: https://doi.org/10.1007/s11069-018-3449-y

[24] Gholinejad, S., Naeini, A. A., and Amiri-Simkooei, A., “Robust Particle Swarm Optimization of RFMs for High-Resolution Satellite Images Based on K-Fold Cross-Validation”, IEEE Journal of Selected Topics in Applied Earth Observations and Remote Sensing, 12, 2594-2599, (2019).

[25] Lopez-Rojas, E., Elmir, A., and Axelsson, S., “Paysim: A financial mobile money simulator for fraud detection”, In 28th European Modeling and Simulation Symposium, EMSS 2016, Dime University of Genoa, 249–255, (2016).

[26] Santos, M. S., Soares, J. P., Abreu, P. H., Araujo, H., and Santos, J., “Cross-Validation for Imbalanced Datasets: Avoiding Overoptimistic and Overfitting Approaches [Research Frontier]”, IEEE Computational Intelligence Magazine, 13, 59-76, (2018).

[27] Seo, J-H., and Kim, Y-H., “Machine-Learning Approach to Optimize SMOTE Ratio in Class Imbalance Dataset for Intrusion Detection”, Computational Intelligence and Neuroscience, (2018).

[28] Tharwat, A., Mahdi, H., Elhoseny, M., and Hassanien, A. E., “Recognizing human activity in mobile crowdsensing environment using optimized k-NN algorithm”, Expert Systems with Application, 107, 32-44, (2018).