Multilayer Perceptron Neural Network Technique for Fraud Detection

Aji Mubalaike (Mubarek)
Istanbul Technical University
Faculty of Computer and Informatics
Istanbul, Turkey
mubalaike15@itu.edu.tr

Eşref Adalı
Istanbul Technical University
Faculty of Computer and Informatics
Istanbul, Turkey
adali@itu.edu.tr

***Abstract*—Fraud detection is an enduring topic that pose a threat to banking, insurance, financial sectors and information security systems such as intrusion detection systems (IDS), etc. Data mining and machine learning techniques help to anticipate and quickly detect fraud and take immediate action to minimize costs. This paper starts with the definition of intrusion detection system and its types, focuses on the implementation of a set of well-known machine learning classification algorithms (Decision Trees, Naive Bayes and Artificial Neural Networks), which can reduce the existing disadvantages of the intrusion detection systems. Experimental results on NSL-KDD dataset infer that our ANN-MLP method (Multilayer Perceptron) yields average better performance by calculating "confusion matrix" that in turn helps us to calculate performance measure such as, "Detection Rate Accuracy", "precision" and "recall".**

Keywords—fraud detection; IDS; artificial neural network (ANN); multilayer perceptron (MLP); confusion matrix; detection rate (DR) accuracy

***Özetçe*—Dolandırıcılık tespiti, bankacılık, sigortacılık, finans sektörü ve saldırı tespit sistemleri (STS) gibi bilgi sistemleri için her zaman tehdit oluşturan bir sorundur. Veri madenciliği ve makine öğrenmesi teknikleri dolandırıcılık tahminini hızlı bir şekilde tespit etmeyi ve en başarılı ve uygun eylemi ivedilikle gerçekleştirmeyi sağlar. Bu bildiride, saldırı tespit sistemi ve onun türlerini giriş kısmında özetleyerek, bilinen makine öğrenmesi sınıflandırma algoritmalarının (Karar Ağaçları, Naive Bayes ve Yapay Sinir Ağı) uygulanmasına ve karşılaştırılmasına odaklanılmakta; NSL-KDD veri kümesi üzerinde yapılan deney sonuçlarına göre, "Bulma" ve "Tutturma" gibi başarım ölçümlerini hesaplamamıza yardımcı olan "karışıklık matrisi"ni hesaplayarak, ANN-MLP (Yapay Sinir Ağı - Çok Katmanlı Algılayıcı) yönteminin daha iyi ortalama başarım sağladığı kanıtlanmaktadır.**

Anahtar Sözcükler—Dolandırıcılık tespiti; STS; Yapay Sinir Ağı Çok Katmanlı Algılayıcı; Tespit Hızının Doğruluğu

I. INTRODUCTION

Fraud, which is resulting in the loss of billions of dollars worldwide each year, is increasing expeditiously with the expansion of contemporary technology and the global superhighways of communication [1]. Fraudulent activity can also demonstrate the security devices such as intrusion detection system (IDS) that monitors network traffic or system activities in real-time and will send alerts to administrators or take some active actions once the attack is identified.

Intrusion detection systems (IDS) are commonly used to defend systems against fraudulent activities from authorized and unauthorized users, where they can be placed on a host or inside a network. Network-based IDS does not require modification of production servers or hosts. This is an advantage because production servers frequently have close operating tolerances for CPU, I/O, and disk capacity; installing additional software may exceed the system's capacities. Network intrusion detection systems tend to use signature analysis to meet performance requirements [2]. This will detect common programmed attacks from external sources, but it is inadequate for detecting more complex information threats. Host-based IDS tends to have lower false positive rates than do network-based systems. Since the range of commands executed on a specific host are much more focused than the types of traffic flowing across a network. This property can reduce the complexity of host based analysis engines. The disadvantages associated with host-based systems is that they tend to rely on the innate logging and monitoring capabilities of the server.

Rest of the paper is organized as follows. Section 2 discusses the previous work that is related to the detection of fraudulent activities using various kinds of machine learning algorithms. In Section 3, to be the experimental final results consistent and more accurate, we use NSL-KDD data set, then we demonstrate preprocessing work consists of feature engineering that reduces the dimensionality of the hypothesis search space and storage costs, enhance data mining performance, and simplify data mining experimental results. Our proposed approach is discussed in Section 4, we present our experimental results of various classification and intrusion detection techniques and find the classification model generates the most prominent and ideal detection accuracy. Finally, we conclude in Section 5.

II. RELATED WORK

Each kinds of techniques used in previous work has its advantages and shortcomings. Performance of each model

978-1-5386-0930-9/17/$31.00 ©2017 IEEE

varies in terms of DR (Detection Rate), FAR (False Alarm Rate) and accuracy.

In 2008, P. Barapatre and N.Z. Taraporethave proposed "Training MLP Neural Network to Reduce False Alerts in IDS". Authors proposed a Multilayer Perceptron (MLP) with Back-Propagation algorithm. The individual attack detection rate is higher than overall attack detection rate so obtained average detection rate of 81.96% [3].

Z. Salek and R. Azmi proposed "Intrusion Detection using Neural Networks trained by Differential Evaluation algorithm" in 2013 [4]. Authors considered Differential Evolution algorithms like RBF, Probabilistic Neural network (PNN) and Multilayer Perceptron (MLP) on KDD99 Dataset by using PCA for reducing the dimensions of the dataset.

In 2015, "Intrusion Detection System Based on Multi-Layer Perceptron Neural Networks and Decision Tree" was proposed by J. Esmaily and R. Moradinezhad [5]. Authors investigated a method based on Artificial Neural Networks and Decision Trees to design an accurate Intrusion Detection System with high DR and low FAR. Also recommended the capability of such more hybrid method works.

F. Amato and F. Moscato proposed "Multilayer Perceptron: an Intelligent Model for Classification and Intrusion Detection" in 2017 [6]. Authors in their paper analyzed KDD99 data set to generalize power of neural networks to classify the attacks, recommended to try new types of attacks and compare the results with other machine learning models.

III. Dataset Description and Preprocessing

A. *Dataset Description*

The NSL-KDD data set is a refined version of its previous original dataset named KDD 99. In this paper, we will use NSL-KDD data set for analyzing the effectiveness of the various classification algorithms in detecting the anomalies in the network traffic patterns.

The NSL-KDD data set takes the stringent measures to solve some of the built-in problems of the KDD CUP'99 data set. But two important problems had been found that greatly affects the performance of evaluated systems, and results in a very poor evaluation of anomaly detection approaches. To solve these issues, a new data set, NSL-KDD, was generated, which consists of selected records of the complete KDD data set. There are three advantages of the NSL-KDD over the original KDD data set:

- It *does* not include redundant records in the train set, so the classifiers will not be biased to more frequent records.
- The number of selected records from each difficulty level group is not proportional to the percentage of records in the original KDD data set.
- The numbers of records in the train and test sets is ideal, which makes it suitable to run the experiments on the complete set without the need to randomly select a small portion.

The last feature named attack type includes five different values, DoS, U2R, R2L, Probe and Normal. The attack types are explained further below [7] [8].

1) Denial of Service Attack (DoS): is an attack in which the intruder makes some computing or memory resource too busy or too full to handle legitimate requests or denies legitimate users access to a machine.

2) User to Root Attack (U2R): is a class of exploit in which the attacker starts out with access to a normal user account on the system (perhaps gained by sniffing passwords, a dictionary attack, or social engineering) and can exploit some vulnerability to gain root access to the system.

3) Remote to Local Attack (R2L): occurs when an attacker who can send packets to a machine over a network but who does not have an account on that machine exploits some vulnerability to gain local access as a user of that machine.

4) Probing Attack: is an attempt to gather information about a network of computers for the apparent purpose of circumventing its security controls.

B. *Feature Analysis*

Removal of redundant or irrelevant attributes from data sets can facilitate practical applications in improving speed and relieving memory constraints. Data dimensionality reduction can reduce the computation burden [9]. The performance of network intrusion detection systems based on machine learning techniques in terms of accuracy and efficiency largely depends on the selected features.

Feature engineering is using our knowledge of the problem to choose features or create new features that allow machine learning algorithms to work more accurately [10][11]. We created a new feature named attack type shown in TABLE I. that describes the type of each given attack aiming to more clear observations and accurate detection to the result. This will be the Y value or the target value, we are trying to predict with our model. We categorized into five attack types, DoS, U2R, R2L, Probe and Normal from 23 various attack name. When training a machine learning model, we always need to do two things with our data set. First shuffle the data so it is in a random order, and second split the data into a training data set and a test data set. The test size in this paper equals 0.1 tells that we want to keep 90 percent of the data for training and pull out 10 percent of the data for testing.

TABLE I. Generated Attribute After Feature Engineering

Attack Type	Attack Name
DoS	back, land, neptune, pod, smurf, teardrop
U2R	buffer overflow, load module, perl, rootki
R2L	ftp_write, guess_passwd, imap, multi_hop, phf, spy, warez_client, warez_master
Probe	ipsweep, nmap, portsweep, satan
Normal	normal

A 90/10 split is typical. Splitting the data into testing and training groups allows us to keep the test data hidden from the machine learning system until we are ready to verify its accuracy. If we verify its accuracy with training data, it had seen before it would not be much of a test. By using data, the model has not seen before, it proves that the model learned general rules for predicting attack types and it did not just memorize the answers for the specific attacks it had seen before. The attack distribution on training and test set is shown on Fig. 1. It is clearly observed that the instance number of Normal and DoS comprises the biggest part of the original dataset.

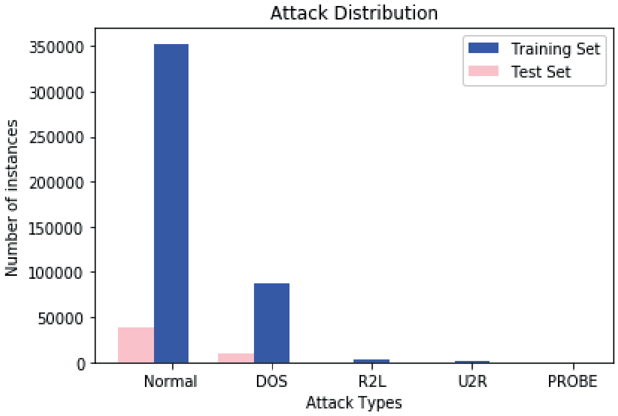


Fig. 1. Attack distribution on test and train set

Doing a good job of feature engineering will make a large difference in the quality of our model. To get the best result possible when training a machine learning algorithm, we want to make the problem as simple as possible for the algorithm to the model. That means we want to feed in features that correlate strongly with the output value. In fact, including useless features can harm the accuracy of our system. Deciding which features to include or exclude from our model is the simplest form of feature engineering. We can also combine multiple features into a single feature.

We cannot feed a feature name directly into our model because it is a string of text and not a number. Instead, we need a way to represent each protocol type as a number. The simplest solution would be to assign a number to each protocol type. But this does not work very well with some machine learning algorithms [12]. The problem is that the machine learning algorithm will think that the order of those numbers is significant. It will assume that bigger numbers are more important than smaller numbers. For instance, the protocal_type and service have the string values that order of which is also important. The order of those numbers is meaningless. This solution is to use a different representation called one-hot encoding. TABLE II. and TABLE III. demonstrate the before and after using of one-hot-encoding method. The method could not encode to the nominal outputs since the category of them could not take the ordinal values.

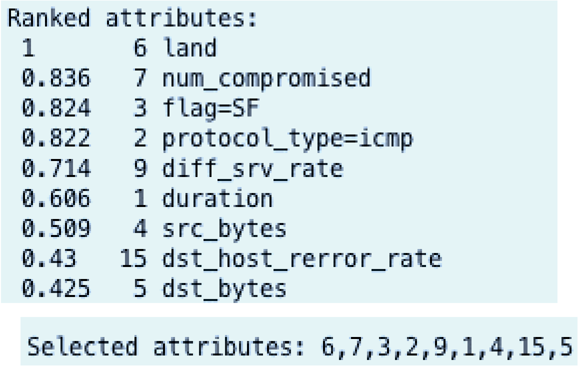
TABLE II. BEFORE USING ONE-HOT ENCODING METHOD

Service	Flag
Private	REJ
ftp_data	SF
ftp	SH
time	RSTR
http	RSTO
telnet	S0
login	S1
…	…
domain	S2

TABLE III. AFTER USING ONE-HOT ENCODING METHOD

Flag_SF	Flag_SH	Flag_RSTR	Flag_S1	…	Flag_S2
0	0	0	0	…	1
0	0	0	1	…	0
0	0	1	0	…	0
0	1	0	0	…	0
1	0	0	0	…	0

Finally, feature selection technique named information gain can be used to find the most relevant attributes. The information gain, entropy value, is calculated for each attribute for the output variable. Entry value vary from zero meaning no information to one meaning maximum information. Those attributes that contribute more information will have a higher information gain value and can be selected, whereas those that do not add much information will have a lower score and can be removed. If we use an arbitrary cutoff of 0.4, nine important attributes will be generated. Fig. 2 shows the importance of each selected attribute after using information gain feature selection technique.



```
Ranked attributes:
 1        6 land
 0.836    7 num_compromised
 0.824    3 flag=SF
 0.822    2 protocol_type=icmp
 0.714    9 diff_srv_rate
 0.606    1 duration
 0.509    4 src_bytes
 0.43    15 dst_host_rerror_rate
 0.425    5 dst_bytes
```

```
Selected attributes: 6,7,3,2,9,1,4,15,5
```

Fig. 2. Importance of each selected feature

IV. DETECTION ANALYSIS

To evaluate the performance of our proposed algorithm, comparisons with a feature set composed of the selected nine features are carried out over the NSL-KDD data set using a multi-layer perceptron. Fig. 3 interprets the experimental methodology for the whole process, Firstly, we do the essential step of feature engineering to the original row data mentioned in previous part. After obtaining the dimensionality reduced dataset with the most important nine features, we should split the dataset into training and test set. Using training set, three prosperous classification algorithms that includes Naive Bayes, Decision Tree and Multilayer Perceptron, will be processed to generate the detection models. In the last, with the help of generated classification models, we can classify the new and unseen test date and choose the model with the highest accuracy.

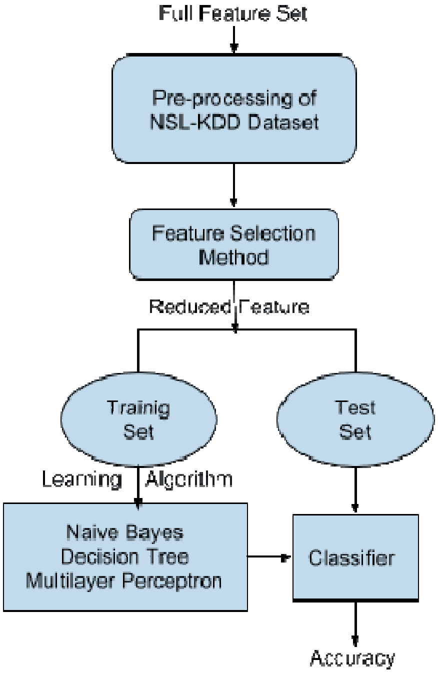


Fig. 3. Experimental methodology

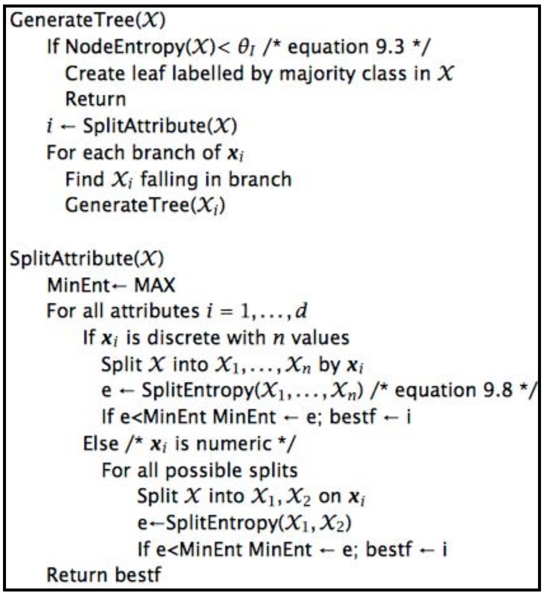
A. Naive Bayes Algorithm

Naive Bayes was originated from the Bayesian theorem. It is particularly suited when the dimensionality of the inputs is high. Parameter estimation for Naive Bayes models uses the method, as in (1), of maximum likelihood [13]. Naive Bayes' classifier ignores possible dependencies namely, correlations, among the inputs and reduces a multivariate problem to a group of univariate problems:

$$p(x|C) = \prod{}^{d}{}_{j=1}\, p(x_j|C) \tag{1}$$

B. Decision Tree Algorithm

Decision tree algorithm, which is shown on Fig. 4, is a hierarchical data structure implementing the divide-and-conquer strategy. It is an efficient nonparametric method, which can be used for classification to generate intrusion detection classifier. We discuss learning algorithms that build the tree from a given labeled training sample, as well as how the tree can be converted to a set of simple rules that are easy to understand. Another possibility is to learn a rule base directly [14].



```
GenerateTree($\mathcal{X}$)
    If NodeEntropy($\mathcal{X}$)< $\theta_I$ /* equation 9.3 */
        Create leaf labelled by majority class in $\mathcal{X}$
        Return
    $i$ ← SplitAttribute($\mathcal{X}$)
    For each branch of $x_i$
        Find $\mathcal{X}_i$ falling in branch
        GenerateTree($\mathcal{X}_i$)

SplitAttribute($\mathcal{X}$)
    MinEnt← MAX
    For all attributes $i = 1, \dots, d$
        If $x_i$ is discrete with $n$ values
            Split $\mathcal{X}$ into $\mathcal{X}_1, \dots, \mathcal{X}_n$ by $x_i$
            e ← SplitEntropy($\mathcal{X}_1, \dots, \mathcal{X}_n$) /* equation 9.8 */
            If e<MinEnt MinEnt ← e; bestf ← i
        Else /* $x_i$ is numeric */
            For all possible splits
                Split $\mathcal{X}$ into $\mathcal{X}_1, \mathcal{X}_2$ on $x_i$
                e←SplitEntropy($\mathcal{X}_1, \mathcal{X}_2$)
                If e<MinEnt MinEnt ← e; bestf ← i
    Return bestf
```

Fig. 4. Classification tree construction

C. Multilayer Perceptron Algorithm

The multilayer perceptron is an artificial neural network structure and is a nonparametric estimator that can be used for classifying and detecting intrusions. We will demonstrate the backpropagation algorithm to train a multilayer perceptron for the application of network intrusion detection.

Neural interconnections in the brain are abstracted and implemented on digital computers as neural network models. The resurgence of interest in neural networks has been fueled by the success in theory and applications. A typical multilayer perceptron (MLP) neural network and a hidden neuron in the hidden layer depicted in Fig. 5. A hidden layer is required for MLPs to classify linearly inseparable data sets [15]. The jth output of a feedforward MLP neural network is:

$$y_j = f\left(\sum_{i=1}^{K} \boldsymbol{W}_{ij}^{(2)} \emptyset_i(\boldsymbol{x}) + \boldsymbol{b}_j^{(2)}\right) \tag{2}$$

where $b_j^{(2)}$ is the bias of output neuron $j^{(x)}$ is the output of hidden neuron i, $\emptyset_i(x)$ is the input vector.

$$\emptyset_i(\boldsymbol{x}) = \boldsymbol{f}(\boldsymbol{W}_i^{(1)} \cdot \boldsymbol{x} + \boldsymbol{b}_i^{(1)}) \tag{3}$$

where $b_i^{(1)}$ is the bias of hidden neuron i shows a two layer MLP neural network with a hidden layer and an output layer. The input nodes do not carry out any processing.

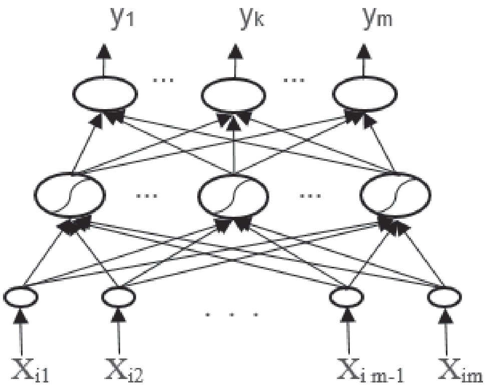


Fig. 5. A two layer MLP neural network with a hidden layer and an output layer

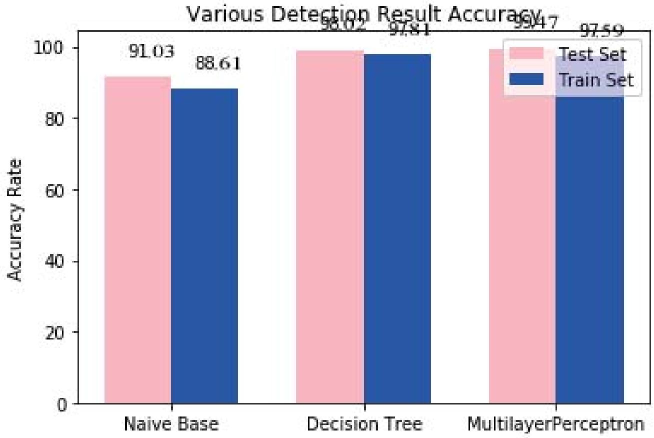


Fig. 6. Detection result using various detection algorithms

TABLE IV. DETECTION RATE ACCURACY RESULTS THROUGH VARIOUS METHOD

Dataset	**DR Accuracy (%)**		
	Naive Bayes	***Decision Tree***	***ANN-MLP***
Training Set	88.61	97.81	97.59
Test Set	91.03	99.02	99.47

V. CONCLUSION

Intrusion Detection Systems (IDSs) are the security tools used to detect anomalous or fraudulent activities from inside and outside intruders. A set of machine learning algorithms can be used to get meaningful insights into the data that are helpful to making effective decisions for the fraudulent activities. In this paper, we have generated intrusion detection and classification models using multilayer perceptron algorithm compared it with Naive Bayes and Decision Tree algorithm methods.

The experiment result on Fig. 6 presents the accuracy value of each methods on the row test data and the Multilayer perceptron algorithm has the highest accuracy with 99.47% by using nine selected features. The accuracy of this method is higher than the accuracy of full data and is also as highly as accuracy of other methods. Future work can include a comparison of the other more up-to-date algorithms obtaining high accuracy as well as less complexity.

REFERENCES

[1] R. Bolton and D. Hand, "Statistical Fraud Detection: A Review," Statistical Science, August, 2002.

[2] BindView and Clicknet, "Intrusion Detection Systems, Buyers' Guide," ICSA Labs.

[3] B. Prachi and N.Z.Tarapore, "Training MLP neural network to reduce false alerts in IDS," International Conferences on Computing, 2008.

[4] Z. Salek and R. Azmi, "Intrusion Detection using Neural Networks trained by Differential Evaluation algorithm," 10th International ISC Conference on Information Security, 2013.

[5] J. Esmaily and R. Moradinizhad, "Intrusion Detection System Based on Multi-Layer Perceptron Neural Networks and Decision Tree," 7th international Conference on Information and Knowledge Technology, 2015

[6] A. Flora, M. Nicola, and V. Emilio, "Multilayer Perceptron: An Intelligent Model for Classification and Intrusion Detection," 31st International Conference on Advanced Information Networking and Applications Workshops, 2017.

[7] NSL-KDD Dataset Description, improvement to the KDD' Dataset, [Online] available http://www.unb.ca/cic/research/datasets/nsl.html

[8] H. Chae, B. Jo, and S. Choi, "Feature Selection for Intrusion Detection using NSL-KDD," Recent Advances in Computer Science, Korea, 2014

[9] Y. Lei and L Huan, "Feature selection for high-dimensional data: A fast correlation-based filter solution," In ICML, volume 3, 2003, pp. 56–863.

[10] G. Gokhan, C. Zehra, and Y. Lei. "Stable and accurate feature selection. In Machine learning and knowledge discovery in databases," Springer 2009, pp. 455–468.

[11] Y. Chen, A. Abraham, and B. Yang, "Feature selection and classification flexible neural tree," Neurocomputing, vol. 70, no. 1, 2006, pp. 305– 313.

[12] C. Zehra, E. Umit, and C. Tanju, "Online Feature Selected Semi-Supervised Decision Trees for Network Intrusion Detection," Istanbul Technical University.

[13] G. Fatma, C. Okan, and E. Zeki, "Online Naive Bayes Classification for Network Intrusion Detection," IEEE/ACM International Conference on Advances in Social Networks Analysis and Mining, 2014.

[14] E. Alpaydın, Introduction to Machine Learning. 3rd Edition Cambridge, MA: The MIT Press, 2014.

[15] W. Lipo and F. Xiuju, "Data Mining with Computational Intelligence," Germany, Springer, 2005.