Journal, Advances in Mathematical & Computational Sciences
Vol. 8 No. 2 June 2020
www.mathematics-computationaljournal.info

Article Citation Format
Abdulsalami, B.A.. & James, O.O. (2020): Anomaly Detection Model in Online Transactions Using Supervised Learning Technique. Journal of Advances in Mathematical & Computational Sc. Vol.8, No. 2. Pp 47-66

Article Progression Time Stamps

Article Type: Research Article
Manuscript Received 27thApril, 2020
Final Acceptance: 12th June, 2020
Article DOI Prefix: dx.doi.org/10.22624

Anomaly Detection Model in Online Transactions Using Supervised Learning Technique

[1]Abdulsalami B.A. & [2]James O.O.
Department of Mathematical and Computer Sciences
Fountain University
Osogbo, Nigeria
E-mail: [1]basiratabdusalam@gmail.com, [2]seun.james@vdtcomms.com

ABSTRACT

Credit card fraud activities have rapidly increased all over the world and are still evolving, with different techniques been deployed by fraudsters to perpetrate the evil. Consequently, organizations and individual user suffers if the information of credit card could get leaked to these fraudsters, via loss of the cards or other means. The paper focus on investigating the extent to which Feed Forward Back Propagation Neural Network algorithm can be used to model anomalies detection in online credit card transactions. The model was simulated using MATLAB. The system considered only credit card online transaction among other online transactions. The credit card holder's transactions details (284,807 in number), which consist of demographic and transaction variables were acquired online at www.kaggle.com. Seventy percent (70%) of the transaction dataset was used in training while thirty percentages (30%) was used in validating the models via testing. It was discovered that Feed Forward Back Propagation Neural Network (BPNN) has classifier accuracy of 99.9%, AUPRC of 59.3% and Prediction Accuracy of 79.9%. Thus, this work has helped proven that Feed Forward BPNN based model can detect fraud in online transactions with 79.9% performance accuracy.

Keywords*:* Credit card fraud, Machine learning, Back propagation Neural Network, Online transactions, Security.

1. INTRODUCTION

In recent years, the internet has become the main medium for conducting electronic commerce. Many products, tangible and intangible, are browsed through and sold over the Internet. With the increasing popularity of e-commerce in our day-to-day business activities, credit card usage has dramatically increased, and has become the standard means of payments for e-commerce. As credit card had become a popular tool for online transactions in many countries lately, this has created opportunity for thieves to steal credit card details and subsequently commit fraud. (Samaneh et al., 2016; Abdulsalami *et al*., 2019). Fraud is an intentional deception with the purpose of obtaining financial gain or causing loss by implicit or explicit trick (Samaneh *et al*., 2016; Abdulsalami *et al*., 2019). Credit Card fraud is an evolving problem. It is increasing considerably with the development of modern technology and global super highways of communications which cost hundreds of millions of dollars annually (Akshata & Sheetal, 2015).

Credit card fraud affects the organization by financial losses and individual user is also affected if the credit card gets stolen (Abdulsalami et al., 2019). Different credit card fraud activities have rapidly increased all over the world and are still evolving, with different techniques implemented by fraudsters to perpetrate this menace. For many years, the credit card industry has studied computing models for automated detection systems. Recently, these models have been the subjects of academic research, especially with respect to e-commerce (Chan et al., 1999; Barnerjee *et al*., 2018; Abdulsalami *et al*., 2019). The growing number of credit card transactions provides more opportunity for thieves to steal credit card numbers and subsequently commit fraud. Despite significant efforts by merchants, card issuers and law enforcement to curb fraud, online fraud continues to plague electronic commerce web sites (Aderounmu et al., 2012; Abdulsalami et al., 2019).

Fraud detection is a continuously evolving discipline and ever-changing tactics to curb fraud. Thus, it needs special methods of intelligent data analysis for detection and prevention (Razak & Ahmed, 2014). Techniques based on Machine Learning(ML) such as Data mining, Fuzzy logic, Sequence Alignment, Clustering Algorithms, Genetic Programming, etc., has evolved in detecting various credit card fraudulent transactions. With this development, detection of anomalies has been one of the major focusing areas of researches in important topic in data mining and machine learning. Many real-world applications such as intrusion or fraud detection require an effective and efficient framework to identity deviated data instances. However, most anomaly detection methods are typically implemented in batch mode, and thus cannot be easily extended to large-scale problems without sacrificing computation and memory requirements (Lee, Yeh & Wang, 2013).

In this work, Feed forward Neural Network, a variety of ANN algorithm was modeled to detect anomalies in an online credit card transaction. It investigated the usefulness of applying this machine learning algorithm, which is a supervised learning technique in fraud detection in an online transaction, by evaluating the performance of the model. The rest of the paper is organized as follows: Section 2 discusses literature review, which elaborates on the subject matter and existing related works. Section 3 explains the methodology adopted outlining the phases of the methodology, followed by section 4, which discusses the implementation and the result presented in Section 5. Finally, the paper concludes with some final remarks as well as directions for future work.

2. LITERATURE REVIEW

2.1 Anomaly Detection

Anomaly detection is defined as the process of using models to identify behavior that is different from the normal behavior of a system. In other words, it refers to the problem of finding patterns in data that do not conform to expected behavior. These nonconforming patterns are often referred to as anomalies, outliers, discordant observations, exceptions, aberrations, surprises, peculiarities, or contaminants in different application domains. In 2011, Guardian Analytics stated that anomaly detection is the process of detecting something unusual relative to something expected. In the realm of online banking or online transactions, this can be termed as suspicious (unusual) behavior in order to identify account takeover and fraudulent transactions.

Anomaly detection finds extensive use in a wide variety of applications such as fraud detection for credit cards, insurance, or health care, intrusion detection for cyber-security, fault detection in safety critical systems, and military surveillance for enemy activities (Varun, Arindam & Vipin, 2009). Anomalies in credit card transaction data could indicate credit card or identity theft (Neda, Leila & Ebrahim, 2012). Thus, the on-time detection of anomalies in a reliable, efficient and robust manner had been seen to be highly imperative.

2.2 How Anomaly detection works

The most effective anomaly detection approach focuses on the individual account holder. Different users quite naturally have different online banking behavior. Each account holder has a unique online banking fingerprint. Anomaly detection takes advantage of this fact combined with knowledge of online banking fraud attacks and general online behavior to determine if a specific online session is legitimate or has high risk of being fraudulent (Guardian Analytics, 2011). Guardian Analytics (2011) presented a simple breakdown of the process anomaly detection solutions use to detect suspicious activity for each individual account holder:

i. Create and continually update a model of expected behaviour for each individual account holder.
ii. Monitor all online banking for each individual account holder.
iii. Analyze all individual account behaviour during an online banking session from login to logout – how they access their account, how they manage their accounts, the types of transactions they engage on, the frequency of activities, what kinds of activities take place during the same session, the type and amounts of payments, who the payees are, and much more.
iv. By comparing individual or groups of activities in this online session to demonstrated patterns of normal behavior, determine if the session is legitimate or unusual, unexpected, or suspicious.

2.3 Credit Card in Online Transaction

In today's world, credit cards are used for purchasing goods and services via online transaction or physical card for offline transaction, even street vendors now accepts cashless payments. This development could be nailed to globalization. The increased use of internet technology for online shopping has resulted in a considerable increase of credit card transactions throughout the world (Patel & Kumar, 2013), along with the rapid advances of e-commerce.
Credit card has become a convenience to all stakeholders in the financial economy. It is easy to carry and easy medium of payments while on the move and for online purchase (Dey & Sudha, 2018), and it might be physical or virtual. Despite all the benefits credit card serves, the rapid growth in the number of credit card transactions has led to a substantial rise in fraudulent activities. The rise in e-commerce has opened up new opportunities for criminals to steal credit card details and consequently commit fraud. According to Global Payments Report 2015, credit card is the highest used payment method globally in 2014 compared to other methods such as e-wallet and Bank Transfer. In the past couple of the years, credit card breaches have been trending alarmingly. Nilson Report also reported that the global credit card fraud losses reached $16.31 billion in 2014 and it was estimated that it will exceed $35 billion in 2020.

The development of efficient methods which can distinguish rare fraud activities from billions of legitimate transactions seems essential. Although, CCFD has gained attention and extensive study especially in recent years and there are lots of surveys about this kind of fraud (Samaneh Sorournejad *et al*, 2016).

2.4 Credit Card Fraud Detection Process

Credit card fraud being one of the major problems in the financial institutions such as banks, credit card industry etc., the goal of a detection system is to be able to detect fraud in the dataset in a real time manner, so as to reduce fraudulent transactions which cost hundreds of millions of dollars annually (Akshata & Sheetal, 2015). The main idea when detecting fraud is to firstly understand and identify the type of credit card fraud. There are various types of credit card fraud (both online and offline). Depending on the type of fraud faced by banks or credit card companies, various measures has been adopted and implemented curb these activities, but however, there is still a need for a more robust system in order to detect frauds more accurately.

A transaction is fraudulent if the transaction pattern and other properties (e.g. location, amount, etc.) do not conform or follow the regular pattern of that card dataset. A fraudulent transaction can be detected if the regular way in which the card owner uses his/her card isn't followed. The credit card dataset is used in training the ML algorithms. These algorithms are capable of learning and predicting without any human intervention. Consequently, the system will be able to recognize the patterns of every particular card holder and if any future transaction is fraudulent, the algorithm or model will be able to detect such fraudulent transactions.

2.5 Related Works

Detection of abnormalities in credit card online transactions has enjoyed quite number of attention and interest among researchers. A good amount of works in this domain are available in literatures. Navanshu *et al.* (2018), proposed a new collative comparison measure that reasonably represents the gains and losses due to fraud detection. They presented a cost sensitive method which is based on Bayes minimum risk using the proposed cost measure. The significance of their work was to find an algorithm to reduce the cost measure. An improvement up to 23% was obtained compared to other state of art algorithms. They used a real life transactional data by a large European company and the personal details in the data were kept confidential, the accuracy of the algorithm was gotten to be around 50%.

Fashoto *et al.* (2016) used a hybrid of K-means clustering with Multilayer Perceptron (MLP) and the Hidden Markov Model (HMM). They used K-means clustering to group together the suspected fraudulent transactions into a similar cluster. The output of this was used to train the HMM and the MLP, and later used to classify the incoming transactions. In their results, it was discovered that the detection accuracy of "MLP with K-means Clustering" is higher than the "HMM with K-means clustering" but the result is reversed for 10-fold cross-validation. In another work by Agrawal *et al.* (2015), they proposed testing credit card transaction for fraud using HMM, Behavior based technique and Genetic Algorithm (GA). HMM was used to maintain the record of previous transactions, Behavior based technique used for grouping the datasets while the GA was used for optimization, that is, calculating the threshold value.

Also, in the same year, Pooja *et al.* (2015) proposed simple K-means and Simple GA for fraud detection. They presented how K-means algorithm grouped the transactions based on the distinct attribute values, while GA was used for optimization because the increase in size of the input k-means algorithm produced outliers. Basically K-means produced clusters which were then optimized by the GA. In addition, Behera and Panigrahi (2015), proposed a hybrid approach to CCFD using Fuzzy Clustering and NN. Their work makes use of two phases. In phase one, they used a K-means clustering algorithm to generate a suspicious score of the transaction while in the next phase, a suspicious transaction was fed into NN to determine whether it was really fraudulent or not.

Esmaily *et al.* (2015) also proposed a hybrid of ANN and Decision Tree (DT). They used a two-phase approach. In the first phase, the classification results of DT and Multilayer perceptron were used to generate a new dataset which is the fed into the Multilayer perceptron, in the second phase, in order to finally classify the data. Their model promises reliability by giving very low false detection rate. Devaki *et al.* (2014) developed a CCFD using time series analysis. The parameters considered were transaction amount and transaction time. They used the periodic pattern in the spending behavior of a cardholder to detect the anomalies in the transaction with respect to the analyses of the past history of transactions belonging to an individual cardholder. Two levels of detection methods were used. The first level detects fraud by analyzing whether the new incoming transaction is fraud or not, using distance-based method, while in the second level, the next transaction was predicted by means of label-prediction methodology and compared with the actual transaction. A deviation implies a fraudulent transaction.

If the particular transaction is considered as a fraud, then the cardholder is asked to continue the transaction by asking a secret question. However, if the cardholder does not give correct answer, then the transaction is denied to continue further. The approach decreased the false positive (FP) situation and hence it is ensured that genuine transaction is not rejected. Patel and Gond (2014), developed SVM learning for CCFD. The SVM based method with multiple kernel involvement, includes several fields of user profile instead of only spending profile. The simulation result shows improvement in True positive (TP), True negative (TN) rate, and also decreases the False positive (FP) and False negative (FN) rate. They also looked into the effectiveness of combining two techniques using both the user profile and spending profile in detecting the anomalies in online transaction, with the goal of improving the false rejections and prediction accuracy, by comparing it with standalone units of the algorithms.

Avinash & Thool (2013), used HMM with clustering algorithm to implement CCFD. They were able to build a system that can detect fraud using spending profile. The system checks for the past transaction history of the customer and make decision from it. Its limitation is TP and FP issues. Falaki *et al.* (2012), developed a probabilistic CCFD system in online transactions. The developed model serves as a basis for mathematical derivation for adaptive threshold algorithm for detecting anomalous transactions. Experimental results show the performance and effectiveness of new approach system and demonstrate the usefulness of learning the spending profile of the cardholders. The optimization of parameters, posterior-viterbi cum new detection model performed better than viterbi cum old detection model. The results obtained from the evaluation showed the overall average of accuracy and precision are about 84% and 86% respectively.

Dhanapal & Gayathiri (2012) used DT and Hunts algorithm techniques to implement CCFDS. They find out the fraudulent customer/merchant by tracing fake mail and IP Address. Customer /merchant are suspicious if the mail is fake, they traced all information about the owner/sender through IP Address. Their work was termed “Tracing Email and IP fraud detection”. Pouramirarsalani *et al.* (2011), proposed a new method for fraud detection. They used a hybrid of feature selection and GA. They observed the salient features of the credit card transactions and detected any unusual feature by flagging it to be the fraud one. The GA was used in the optimization and search problems. In addition, Raj & Portia (2011), proposed a paper that presents a research about a case study involving CCFD, where data normalization was applied before Cluster Analysis. The results obtained shows that the clustering attributes of neuronal inputs can be minimized, and promising results can be obtained by using normalized data and data should be MLP trained. Their work was based on unsupervised learning. The significance of their paper was to find new methods for fraud detection and to increase the accuracy of results.

Patidar & Sharma. (2011), also used ANN to detect credit card fraud. They used GA to derive the optimal parameters of ANN. Like many other data mining techniques, ANNs make use of a number of parameters which need to be specified by software developers. Although the values of theses parameters can seriously affect the predicting accuracy of ANN models. A standard practice for specifying these parameters has never been established. The disadvantage of this was the time taken to train the algorithm with data and the optimization process. Arunabha *et al*. (2011) used Artificial Immune system (AIS) to detect fraud by matching binary strings, using detector and response. The system works like the human immune system by fighting fraud before it can occur. The limitation of their work is the complexity of the solution. However, a sound and clear understanding of all these approaches is needed, which will certainly lead to an efficient CCFDS. Thus, this work presented a clear understanding of how feed forward NN algorithm was model to detect fraud in credit card transaction, as compared to other existing works, showing the performance evaluation result of the algorithm.

Journal, Advances in Mathematical & Computational Sciences
Vol. 8 No. 2 June 2020
www.mathematics-computationaljournal.info

3. RESEARCH METHODOLOGY

CCFD in online transaction as developed in this work involved two phases; Data preparation phase and implementation phase, which is shown in Figure 1.

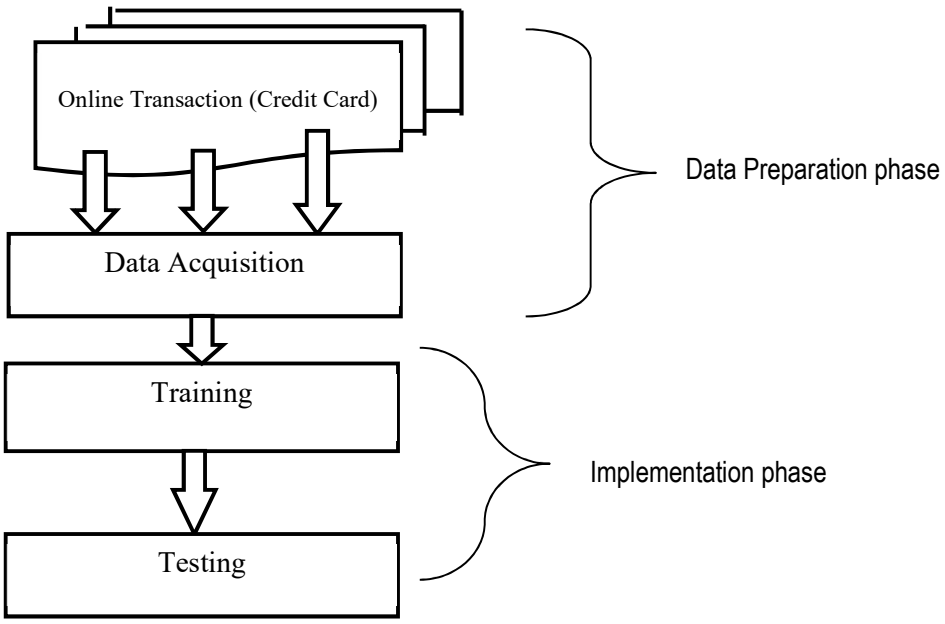


Figure 1: Architecture of the model

3.1 Data Preparation Phase

This phase involves the preparation of the accumulated data for training, which comprises majorly the card details. The card details consist of the demographic and transaction variables. The transaction variables provided information about the transaction details of the cardholder. According to Ghosh & Reilly, (1994), specific attributes in card transaction data are often not revealed but they should comprise of transaction value, transaction time and date, transaction category or payment channels (payment, refund, ATM, mobile top-up, etc.), ATM/POS indicator, Merchant code, card reader response codes, transacting address, account balance, card number, expiration date, etc.
It involves the following sub-phases:

a. **Data Acquisition**

The acquisition of dataset to be used for the model was a difficult task mainly because financial institutions do not generally agree to share their data with researchers for security reasons (Abdulsalami *et al.,* 2019). All efforts to obtain the data from these institutions prove abortive. As a result, a credit card transaction datasets of Europeans cardholders, which was provided on *Kaggle.com* was used to train the model and also test the model performance. A total number of Two Hundred and Eighty-Four Thousand, Eighty Hundred and Seven (284,807) real credit card transactions was used to train and validate the model to detect if the transaction was illegal or legal.

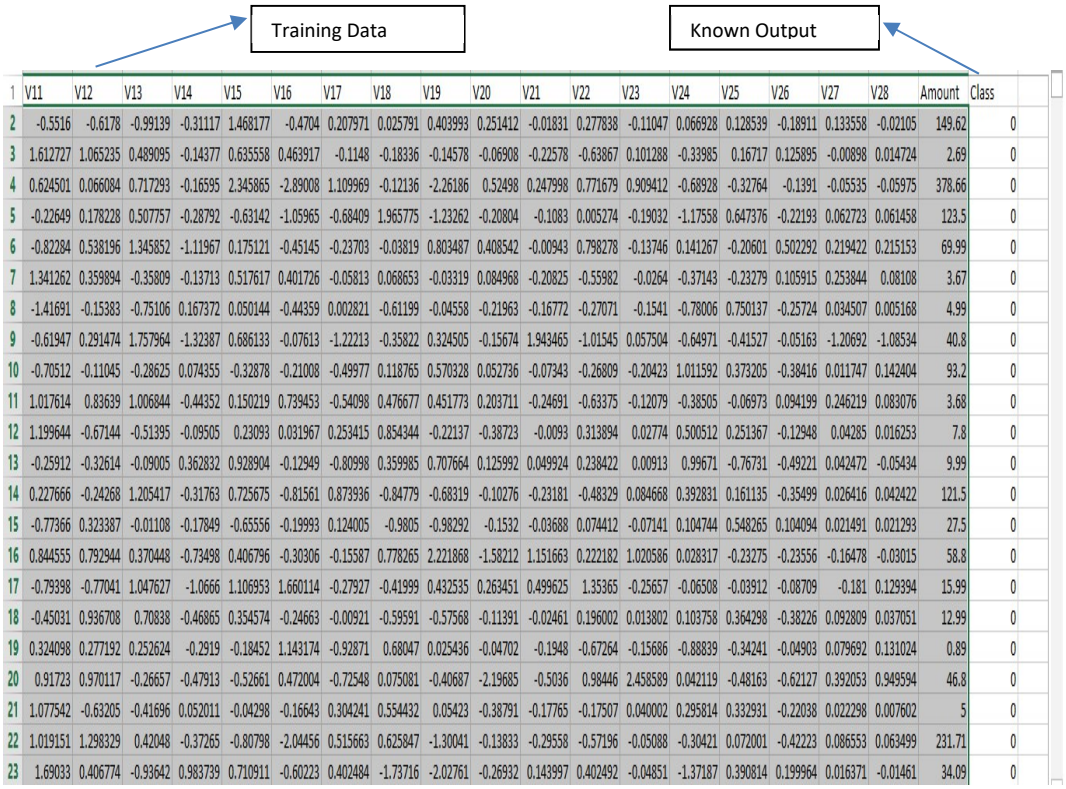
Journal, Advances in Mathematical & Computational Sciences
Vol. 8 No. 2 June 2020
www.mathematics-computationaljournal.info

b. **Data Preparation**

The accumulated data was prepared for training in a data-mapping style by matching the variables from the dataset with the parameters of the algorithm, therefore making the accumulated data present in a form acceptable by the developed model with respect to its parameters.

3.2 The Implementation Phase

This phase encompasses the training and testing phases. The training phase is the learning phase, which involves the processes that are required in preparing the algorithm model for testing purpose. The testing phase assigned fraudulent or non-fraudulent transaction values to each tested transaction using parameters learnt from the training phase. The technique used in this work is a supervised learning technique, which is like finding the correct solution to already known correct answer. Supervised learning is a learning rule that trains the algorithm based on already known correct output. NN stores information in terms of weights, which means, to train a NN with new information, we have to modify the weights. The weights are initialized with adequate values, and the input taken from training data. The input is formatted as input, correct output. The output from the NN algorithm is then compared with correct output and the error is calculated. Then, the weights are adjusted to reduce the error. And we keep re-calculating the error and adjusting the weight for all training data to reduce the error until desire output is achieved. From the dataset, the class column contains our already known correct output which is 0 for non-fraudulent and 1 for fraudulent activities, as shown in Figure 2.



	V11	V12	V13	V14	V15	V16	V17	V18	V19	V20	V21	V22	V23	V24	V25	V26	V27	V28	Amount	Class
2	-0.5516	-0.6178	-0.99139	-0.31117	1.468177	-0.4704	0.207971	0.025791	0.403993	0.251412	-0.01831	0.277838	-0.11047	0.066928	0.128539	-0.18911	0.133558	-0.02105	149.62	0
3	1.612727	1.065235	0.489095	-0.14377	0.635558	0.463917	-0.1148	-0.18336	-0.14578	-0.06908	-0.22578	-0.63867	0.101288	-0.33985	0.16717	0.125895	-0.00898	0.014724	2.69	0
4	0.624501	0.066084	0.717293	-0.16595	2.345865	-2.89008	1.109969	-0.12136	-2.26186	0.52498	0.247998	0.771679	0.909412	-0.68928	-0.32764	-0.1391	-0.05535	-0.05975	378.66	0
5	-0.22649	0.178228	0.507757	-0.28792	-0.63142	-1.05965	-0.68409	1.965775	-1.23262	-0.20804	-0.1083	0.005274	-0.19032	-1.17558	0.647376	-0.22193	0.062723	0.061458	123.5	0
6	-0.82284	0.538196	1.345852	-1.11967	0.175121	-0.45145	-0.23703	-0.03819	0.803487	0.408542	-0.00943	0.798278	-0.13746	0.141267	-0.20601	0.502292	0.219422	0.215153	69.99	0
7	1.341262	0.359894	-0.35809	-0.13713	0.517617	0.401726	-0.05813	0.068653	-0.03319	0.084968	-0.20825	-0.55982	-0.0264	-0.37143	-0.23279	0.105915	0.253844	0.08108	3.67	0
8	-1.41691	-0.15383	-0.75106	0.167372	0.050144	-0.44359	0.002821	-0.61199	-0.04558	-0.21963	-0.16772	-0.27071	-0.1541	-0.78006	0.750137	-0.25724	0.034507	0.005168	4.99	0
9	-0.61947	0.291474	1.757964	-1.32387	0.686133	-0.07613	-1.22213	-0.35822	0.324505	-0.15674	1.943465	-1.01545	0.057504	-0.64971	-0.41527	-0.05163	-1.20692	-1.08534	40.8	0
10	-0.70512	-0.11045	-0.28625	0.074355	-0.32878	-0.21008	-0.49977	0.118765	0.570328	0.052736	-0.07343	-0.26809	-0.20423	1.011592	0.373205	-0.38416	0.011747	0.142404	93.2	0
11	1.017614	0.83639	1.006844	-0.44352	0.150219	0.739453	-0.54098	0.476677	0.451773	0.203711	-0.24691	-0.63375	-0.12079	-0.38505	-0.06973	0.094199	0.246219	0.083076	3.68	0
12	1.199644	-0.67144	-0.51395	-0.09505	0.23093	0.031967	0.253415	0.854344	-0.22137	-0.38723	-0.0093	0.313894	0.02774	0.500512	0.251367	-0.12948	0.04285	0.016253	7.8	0
13	-0.25912	-0.32614	-0.09005	0.362832	0.928904	-0.12949	-0.80998	0.359985	0.707664	0.125992	0.049924	0.238422	0.00913	0.99671	-0.76731	-0.49221	0.042472	-0.05434	9.99	0
14	0.227666	-0.24268	1.205417	-0.31763	0.725675	-0.81561	0.873936	-0.84779	-0.68319	-0.10276	-0.23181	-0.48329	0.084668	0.392831	0.161135	-0.35499	0.026416	0.042422	121.5	0
15	-0.77366	0.323387	-0.01108	-0.17849	-0.65556	-0.19993	0.124005	-0.9805	-0.98292	-0.1532	-0.03688	0.074412	-0.07141	0.104744	0.548265	0.104094	0.021491	0.021293	27.5	0
16	0.844555	0.792944	0.370448	-0.73498	0.406796	-0.30306	-0.15587	0.778265	2.221868	-1.58212	1.151663	0.222182	1.020586	0.028317	-0.23275	-0.23556	-0.16478	-0.03015	58.8	0
17	-0.79398	-0.77041	1.047627	-1.0666	1.106953	1.660114	-0.27927	-0.41999	0.432535	0.263451	0.499625	1.35365	-0.25657	-0.06508	-0.03912	-0.08709	-0.181	0.129394	15.99	0
18	-0.45031	0.936708	0.70838	-0.46865	0.354574	-0.24663	-0.00921	-0.59591	-0.57568	-0.11391	-0.02461	0.196002	0.013802	0.103758	0.364298	-0.38226	0.092809	0.037051	12.99	0
19	0.324098	0.277192	0.252624	-0.2919	-0.18452	1.143174	-0.92871	0.68047	0.025436	-0.04702	-0.1948	-0.67264	-0.15686	-0.88839	-0.34241	-0.04903	0.079692	0.131024	0.89	0
20	0.91723	0.970117	-0.26657	-0.47913	-0.52661	0.472004	-0.72548	0.075081	-0.40687	-2.19685	-0.5036	0.98446	2.458589	0.042119	-0.48163	-0.62127	0.392053	0.949594	46.8	0
21	1.077542	-0.63205	-0.41696	0.052011	-0.04298	-0.16643	0.304241	0.554432	0.05423	-0.38791	-0.17765	-0.17507	0.040002	0.295814	0.332931	-0.22038	0.022298	0.007602	5	0
22	1.019151	1.298329	0.42048	-0.37265	-0.80798	-2.04456	0.515663	0.625847	-1.30041	-0.13833	-0.29558	-0.57196	-0.05088	-0.30421	0.072001	-0.42223	0.086553	0.063499	231.71	0
23	1.69033	0.406774	-0.93642	0.983739	0.710911	-0.60223	0.402484	-1.73716	-2.02761	-0.26932	0.143997	0.402492	-0.04851	-1.37187	0.390814	0.199964	0.016371	-0.01461	34.09	0

Figure 2: Snapshot of the dataset

3.3 How the Algorithm works

Presented below is the pseudocode describing how the algorithm works. Figure 3 shows the flowchart for the steps involved in the process of anomaly detection, and the flow diagram of the model is depicted in Figure 4

Pseudocode 1: Pseudocode of the model.

```
// The following describes the how the algorithm works.
assign all network inputs and output
initialize all weights with small random numbers
repeat
        for every pattern in the training set
                present the pattern to the network
//      propagated the input forward through the network:
        for each layer in the network
            for every node in the layer
                1. calculate the weight sum of the inputs to the node
                2. add the threshold to the sum
                3. calculate the activation for the node
end
end
//      propagate the errors backward through the network
                for every node in the output layer
                 calculate the error signal
                end
                for all hidden layers
                  for every node in the layer
                        1.          calculate the node's signal error
                        2.          ppdate each node's weight in the network
end
              end
  //    calculate the Error Function
end
while ((maximum number of iterations < = specified)
```

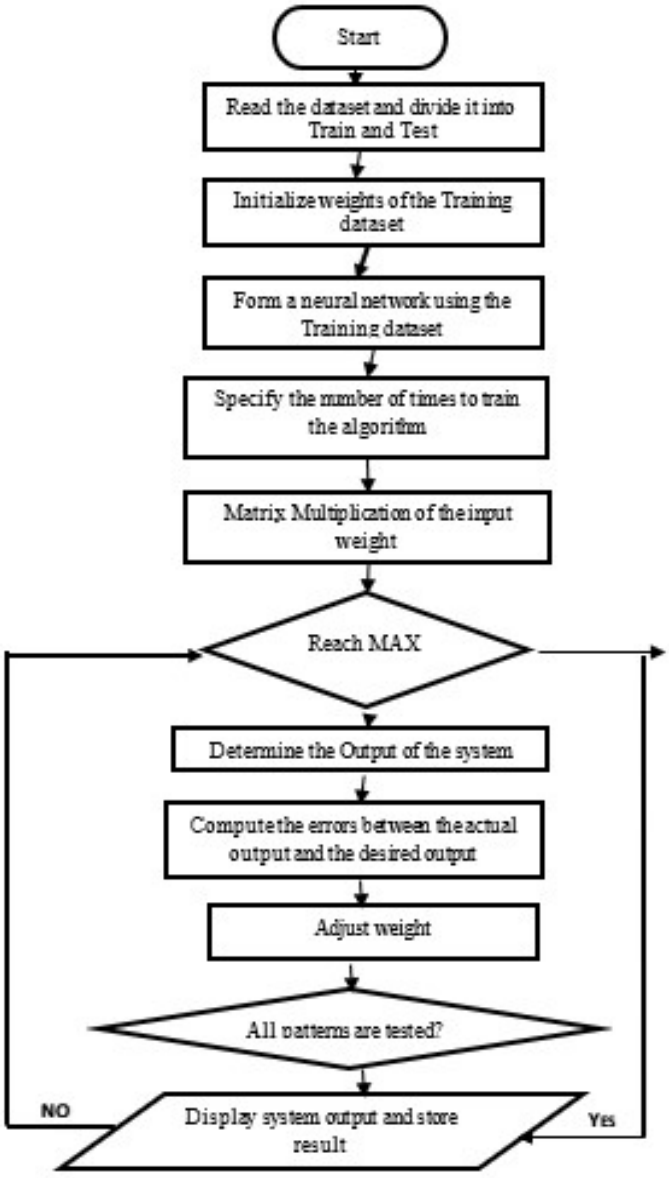


Figure 3: The System Flowchart

3.4 Evaluation Metrics

The performance evaluation metrics considered are System accuracy, Precision and Recall (Sensitivity), Error rate, False positive rate (Specificity), Prediction accuracy, Hit rate and Miss rate, which were calculated with indices True positive (TP), True negative (TN), False negative (FN) and False positive (FP) using equations 1.2, 1.3, 1.4, 1.5, and 1.6 and respectively.

$$Precision = \frac{TP}{(TP+FP)} \quad (1.2)$$

$$Recall = \frac{TP}{(TP+FN)} \quad (1.3)$$

$$Error\ Rate = \frac{FN+FP}{TN+FP+TP+F} \quad (1.4)$$

$$False\ Positive\ Rate\ (FPR) = \frac{TN}{FN+TN} \quad (1.5)$$

$$Predictive\ Accuracy = \frac{TP+TN}{TN+FP+TP+FN} \quad (1.6)$$

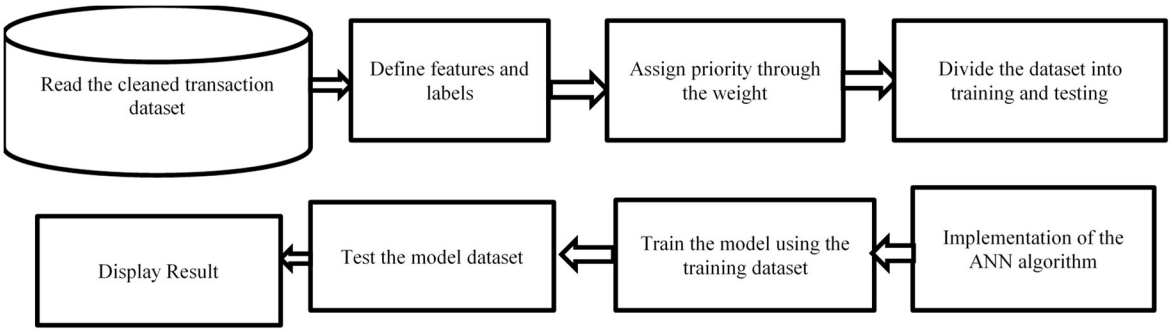


Figure 4: Flow diagram of the detection model

4. FRAMEWORK IMPLEMENTATION

MATLAB tool was the software framework adopted for the implementation of this work. MATLAB stand for Matrices Laboratory. It's a machine learning programming tool with interactive graphical user interface (GUI). It is a high-performance language for technical computing. The first step is to gather or collect the required transaction dataset and load into the simulator environment, as shown in Figure 5. Once the collected transaction datasets are imported into the workspace, input data and known output data are separated. What follow next is to create the training network/algorithm based on define type, structure and parameters, as depicted in Figure 6. This provides privilege to decide on; various network type MATLAB provided like feed forward, radial basis, activation function, internal structure of NN used like the number of nodes in the input layer, number of hidden layer and the number of nodes into the output layer, parameters values like weights, bias, delay.

As shown in Figure 7, Feed forward networks consist of a series of layers. The first layer has a connection from the network input. Each subsequent layer has a connection from the previous layer. The final layer produces the network's output. To ensure that the network is compatible with the problem we want to solve, we configure the network by arranging it. Usually the network is created with default values for its parameters, but one can change this either by reassigning. We used the default value of number of nodes, layers and hidden layer were used. 70% of dataset is used for training while 15% for validation and testing respectively. After the network has been configured, we initialized the weight and biases; these are adjustable network parameters which need to be tuned so that the network performance is optimized.

Journal, Advances in Mathematical & Computational Sciences
Vol. 8 No. 2 June 2020
www.mathematics-computationaljournal.info

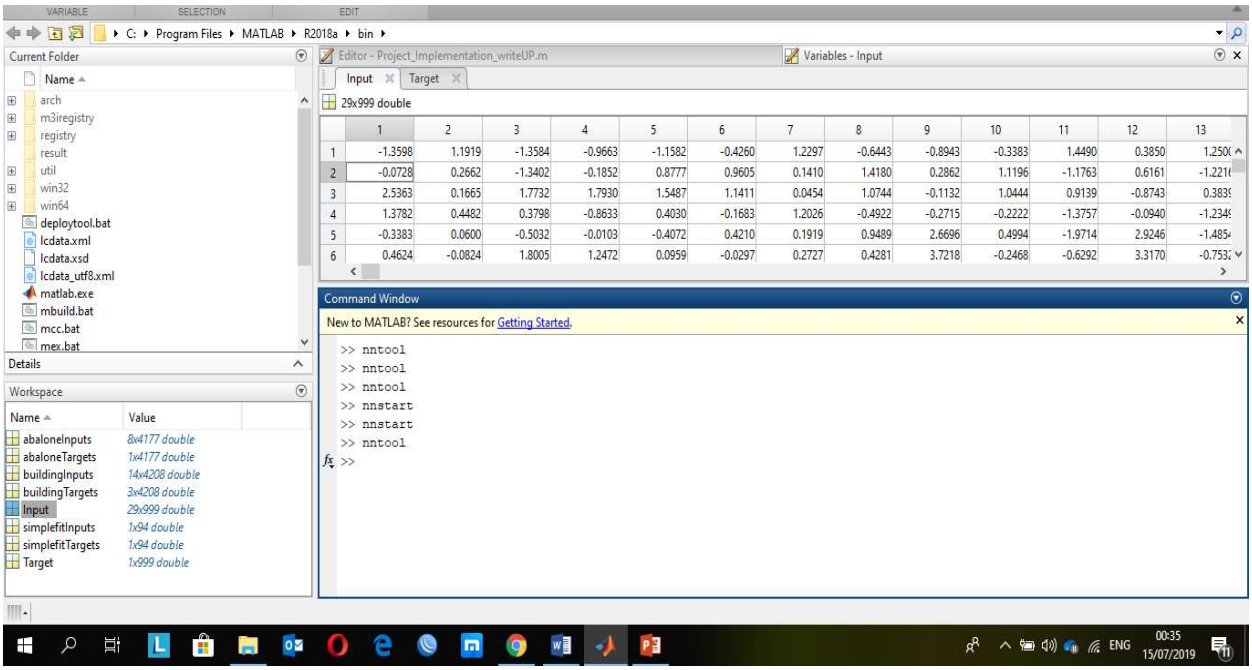


Figure 5: MATLAB environment

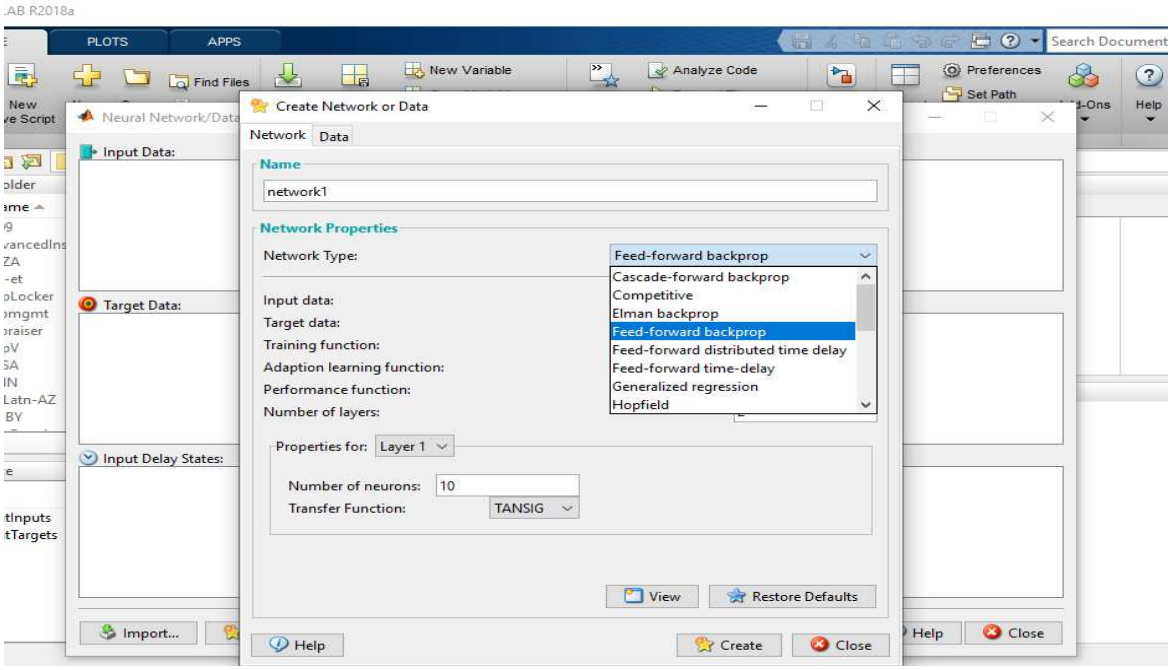


Figure 6: Creating the network

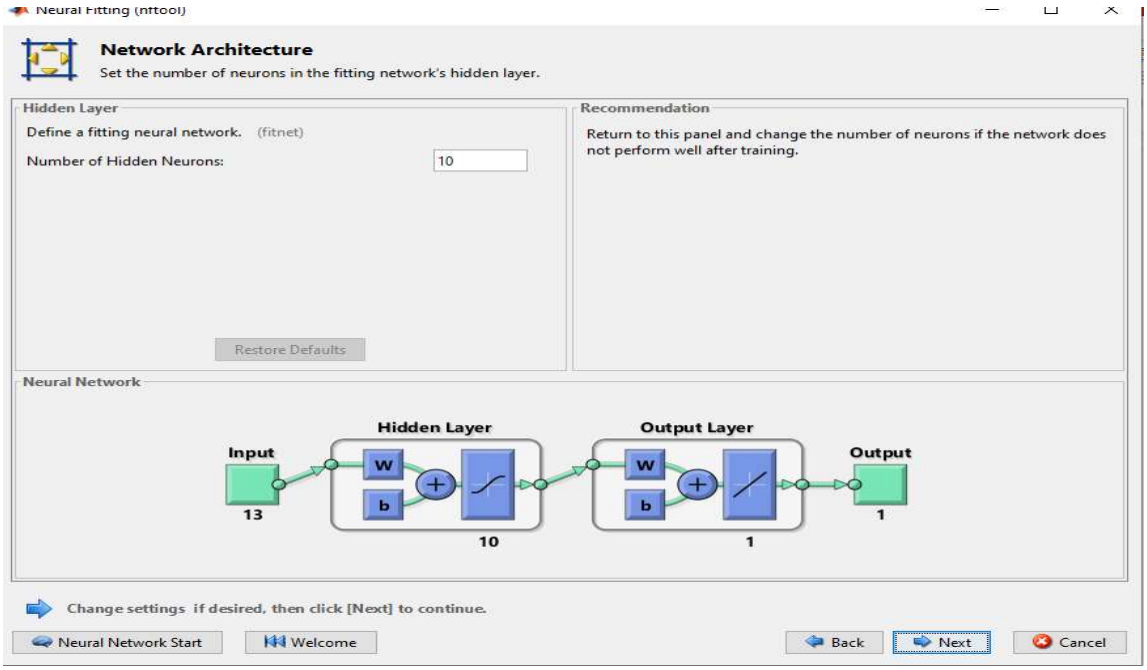


Figure 7: Showing the number of Hidden layers

Training the network, the pairs of input, output dataset is trained with the network created. Here, we created a two-layer feed-forward network, as shown in Figure 7 above. The network has ten (10) hidden layers with ten neurons.net = feedforwardnet(10); The network is trained for up to 1000 epochs to an error goal of 0.1 and then re-simulated.

net.trainParam.epochs = 1000;
net.trainParam.goal = 0.1;

To know when the training had converged, we set the parameter "**show**" before calling the training function

net.trainParam.show = 7;

In this case, the error value appeared on work space every "7" iterations like this:
TRAINB, Epoch 0/1000, MSE 0.5/0.1.
TRAINB, Epoch 7/1000, MSE 0.181122/0.1.
TRAINB, Epoch 14/1000, MSE 0.111233/0.1.
TRAINB, Epoch 21/1000, MSE 0.5189606/0.1.
TRAINB, Performance goal me

After training the network, we tested the performance on a test set. Figure 8 shows the training and testing section in the MATLAB environment.

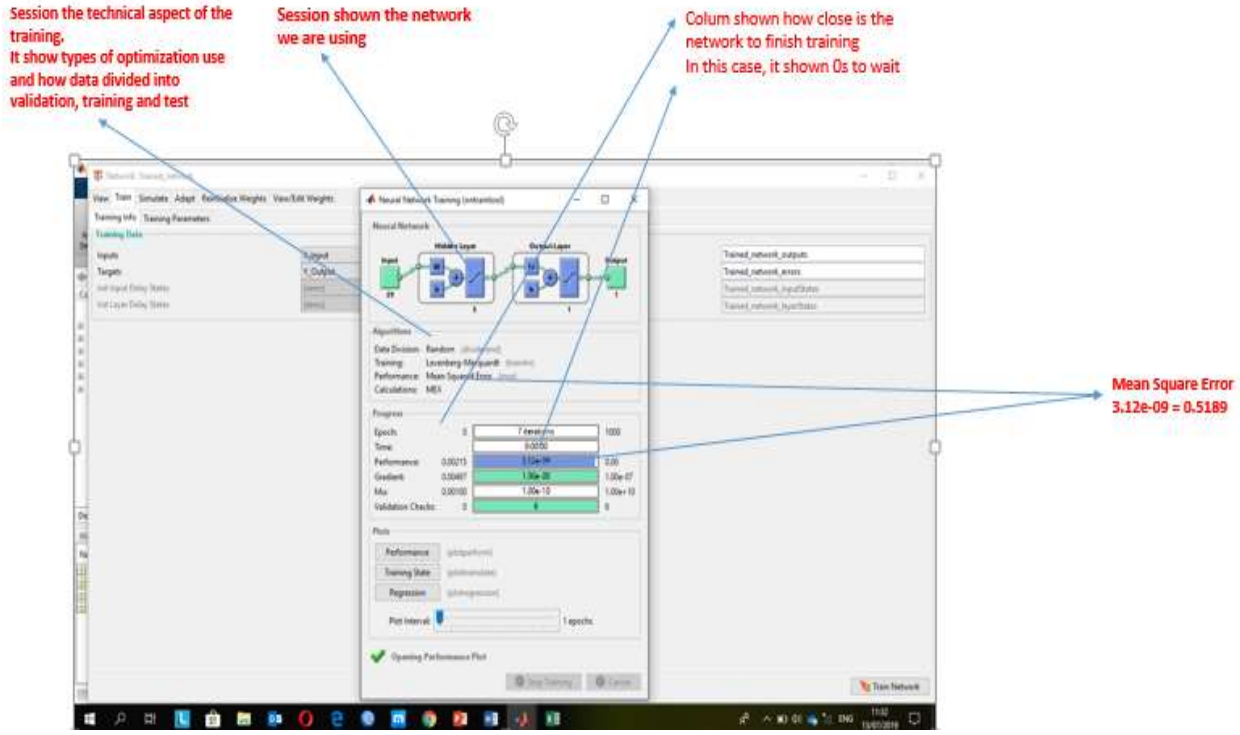


Figure 8: The Train and Test section

5. RESULTS AND DISCUSSION

The system was able to predict sample that are correctly classified and misclassified. The TP obtained is 92727and the TN is 103. Figure 9 shows the classification table where the diagonal element represents the testing samples that are correctly classified, while other diagonal element represents the testing samples that were misclassified. Since we had highly imbalance classes with less than 0.2% of fraudulent activities, the classification accuracy is extremely high, which is 0.9993 (99.93%) as depicted in Figure 10. The Area under precision recall curve (AUPRC) of this model, as shown in Figure 11 is 0.5937, which indicates the need for improvement because a model with higher AUPRC indicate better performance. In others words, if AUPRC is equal to 1, it means the classification is perfect with 100% TP rate and no FP or TN. The overall performance of system is 79% while the validation is 99.98% with training accuracy of 100% with target output, when the neural system hidden layer was adjusted to 10. The graphical result of the training and testing of the model is shown in Figure 12, revealing the training accuracy, validation and performance respectively. The average mean Square error (MSE) that is used as the loss function, i.e. the average squared difference between the estimated values and target is 0.378as depicted in Figure 13. The best performance path is achieved when the validation is at 0.0031179 at first epoch, as shown in Figure 14.

Journal, Advances in Mathematical & Computational Sciences
Vol. 8 No. 2 June 2020
www.mathematics-computationaljournal.info

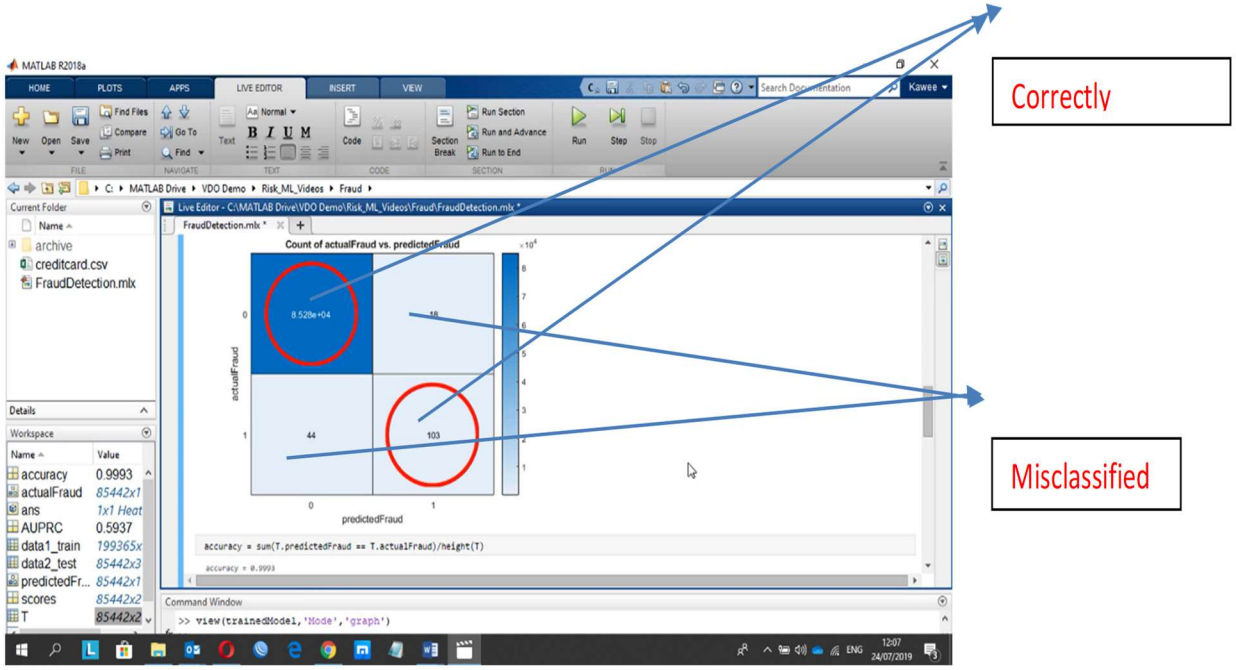


Figure 9: Classification table

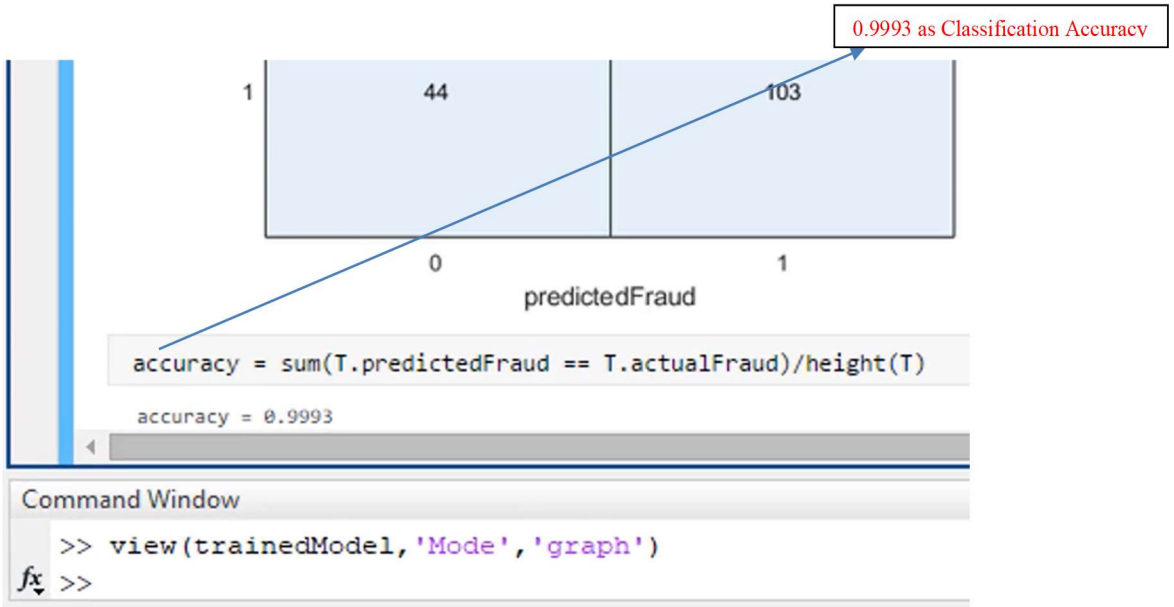


Figure 10: The model classification accuracy.

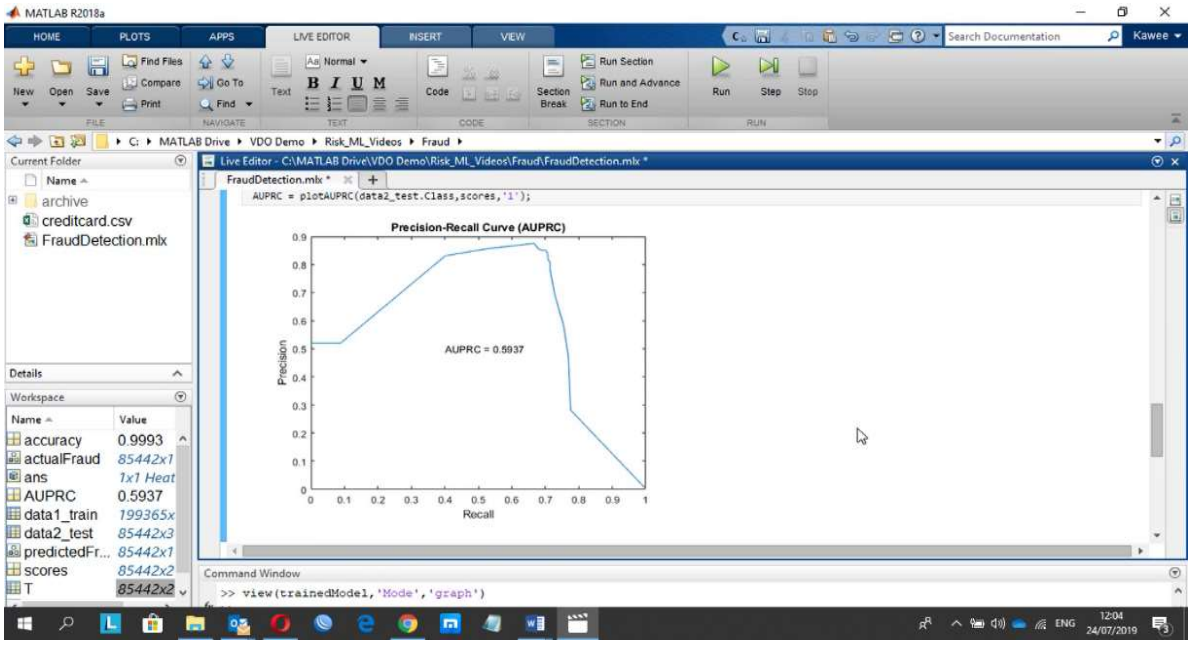


Figure 11: AUPRC Graph

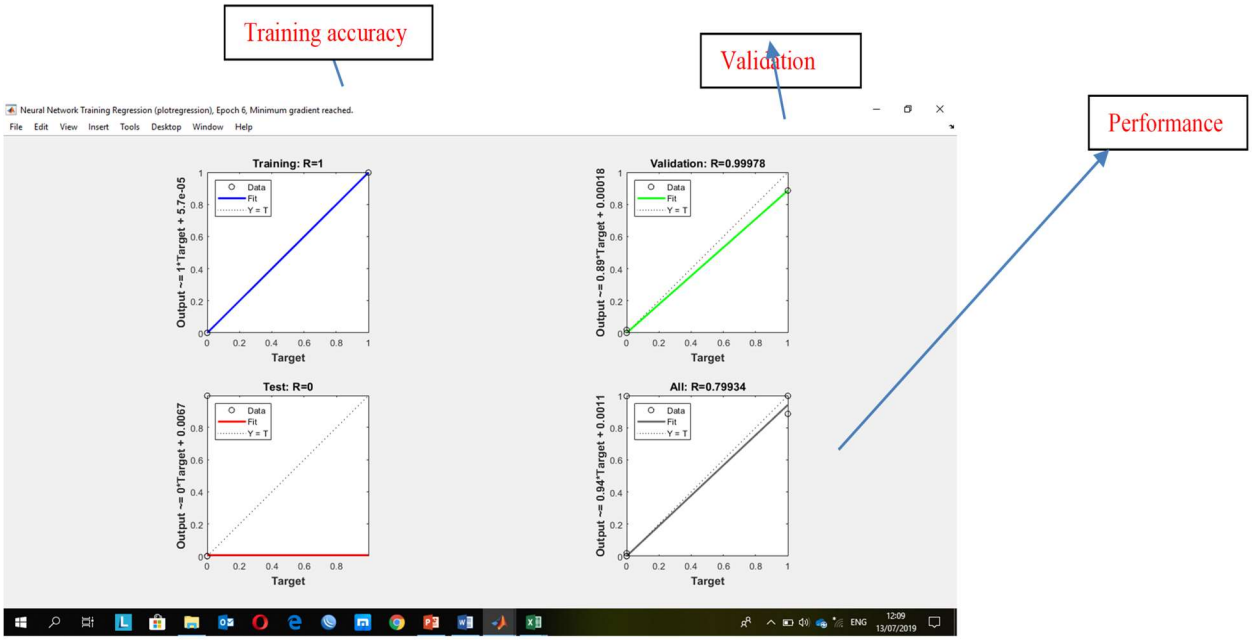


Figure 12: Graph Results of Train and Test

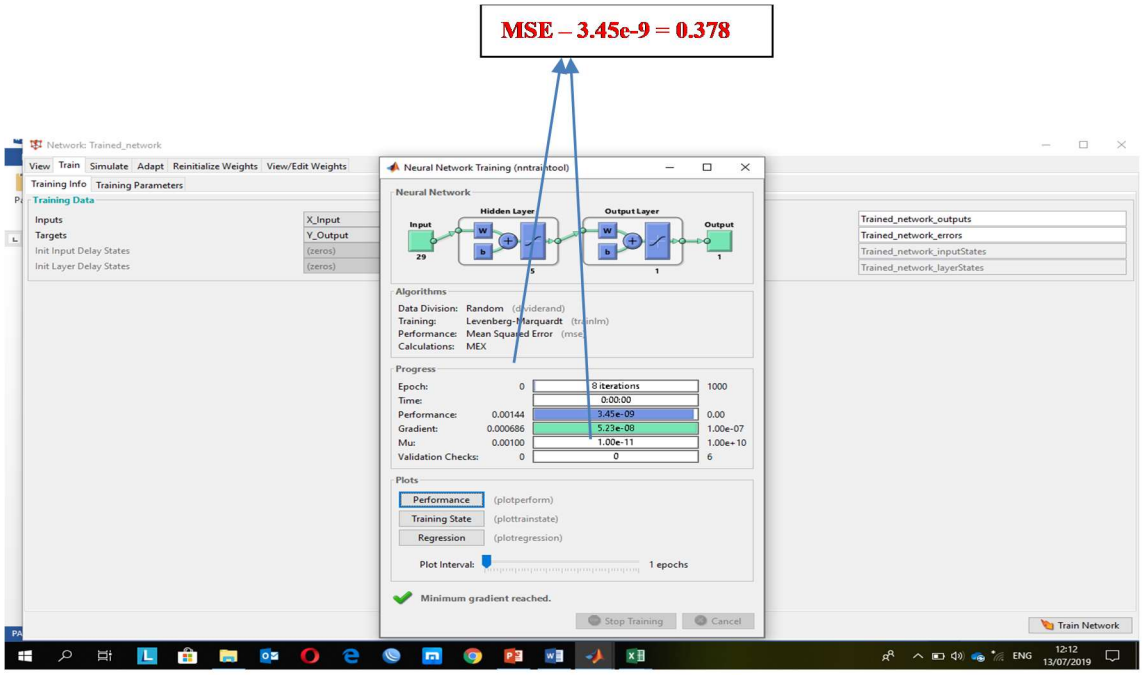


Figure 13: The Train and Test section

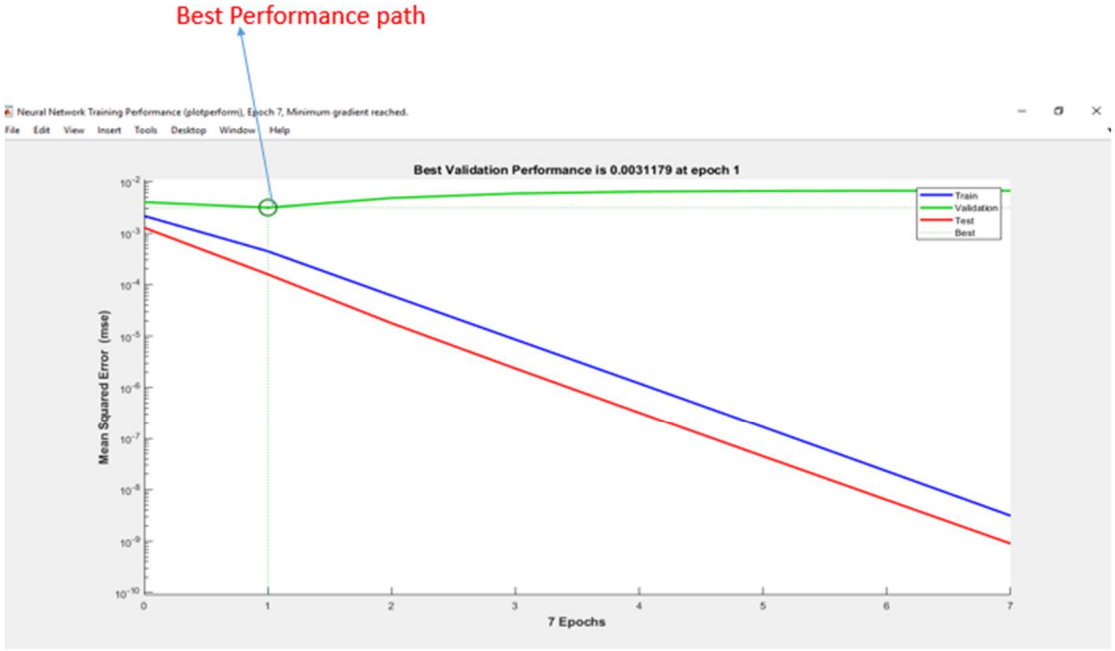


Figure 14: NN Training plot performance

The performance of the model as determined by the evaluation metrics used in this work, and their benchmark, with general comments are summarized below in Table 1

Table 1: Performance Summary of the model

Evaluation Metrics	VALUE	Benchmark	Comments
AUPRC	0.5937	0.4	There is need for improvement
Classifier Accuracy	0.9993	0.8	Best State because of less fraudulent activities
Detection Accuracy	0.7993	0.8	Expected value because of less fraudulent activities
Mean Square Error	0.378	0.5	There is need for improvement
Training Accuracy	1	0.7	Effective Training model
Validation	0.999	0.7	Effective Training model
Correctly |Classified	27 363		
Incorrectly Classified	62		
Detection Accuracy	79%	50%	Still Need more improvement

5.1 Statistical Summary

From Table 1 the percentage accuracy of the algorithm model is 79.9%, and the AUPRC is 0.59, with MSE of 0.378. The number of correctly classified transactions and incorrectly classified transactions are 27363 and 62 respectively.

5.2 Findings

From the above Statistical analysis, we can conclude that feed forward BPNN is effective in detection of anomalies in online credit card transactions. However, the value of the AUPRC indicates the need for improvement because a model with higher AUPRC indicates better performance.

6. CONCLUSION AND FUTURE WORK

This work has contributed to the body of knowledge by successfully demonstrating comprehensively the effectiveness of Feed forward NN as a ML technique for anomalies detection, while generating few false alarms, in online credit card transactions, through implementation with MATLAB. It can be concluded that the model developed can detect fraudulent transaction from any datasets it is subjected to. The model is of greater accuracy and has least tolerant for raising false alarms when compared to some existing work on other models. However, future work can be carried out using real datasets, and comparing the effect of other ANN algorithms with another optimization algorithm.

REFERENCES

1. Abdulsalami, B. A., Kolawole, A. A., Ogunrinde, M. A., Lawal, M., Azeez, R. A., & Afolabi, A. Z. (2019). Comparative Analysis of Back-propagation Neural Network and K-Means Clustering Algorithm in Fraud Detection in Online Credit Card Transactions. *Fountain Journal of Natural and Applied Sciences*, *8*(1). Retrieved from http://fountainjournals.com/index.php/ FUJNAS/article/view/284.
2. Aderounmu, G.A., Adewale, O.S., Alese, B.K., Ismaila, W.O. & Omidiora, E.O. (2012). Investigating the effects of Threshold in Credit Card Fraud Detection System. *International Journal of Engineering and Technology*. 2(7), 328-1332.
3. Agrawal, A., Kumar, S. & Mishra, A.K. (2015). Credit Card Fraud Detection: A case study. 2nd *International Conference on Computing for Sustainable Global Development (INDIACom), New Delhi* 5-7.
4. Akshata, H. & Sheetal, Y. (2015). Online Credit Card Fraud Detection. *International Journal for Research in Engineering Application and Management* 1(2), 1-3.
5. Antara Dey & R. Kavitha Sudha (2018). Credit Card Fraud Detection Based on the Transaction by using Hidden Markov Model and PHP Software. *International Journal of Emerging Technology in Computer Science & Electronics (IJETCSE) ISSN: 0976-1353 Volume 25 Issue 5.*
6. Arunabha Mukhopadhyay, Sayali Mukherjee, Ambuj Mahanti (2011), "Artificial Immune System for detecting online credit card frauds, Research Front, www.csi-india.org, CSI Communications.
7. Avinash, I. & Thool, R. C. (2013). Credit Card Fraud Detection Using Hidden Markov Model and Its Performance. *International Journal of Advanced Research in Computer Science and Software Engineering,* 3(6), 24-31.
8. Banerjee, R., Bourla, G.,Chen, S., Kashyap, M., Purohit S., & Battipaglia, J. (2018). Comparative Analysis of Machine Learning Algorithms through Credit Card Fraud Detection. *New Jersey's Governor's School of Engineering and Technology.* Retrieved from https://www.soe.rutgers.edu on 6th February, 2019.
9. Behera, T.K. & Panigrahi, S. (2015). Credit Card Fraud Detection: A Hybrid Approach Using Fuzzy Clustering & Neural Network. *In Proceedings of the 2015 Second International Conference on Advances in Computing and Communication Engineering (ICACCE)*, Dehradun, India, 494–499.
10. Chan, P.K., Fan, W., Prodromidis, A. L., & Stolfo, S. J. (1999). Distributed data mining in credit card fraud detection, *Intelligent Systems and their Applications, IEEE* 14(60), 67-74.
11. Demla, N. & Agrawal, A.N. (2016). Credit card fraud detection using SVM and Reduction of false alarms, *International Journal of Innovations in Engineering and Technology* 7(2). 176-182.
12. Devaki, R., Kathiresan, V. & Gunasekaran, S. (2014). Credit Card Fraud Detection using Time Series Analysis. *International Journal of Computer Applications Proceedings on International Conference on Simulations in Computing Nexus ICSCN*(3), 8-10.
13. Dhanapal, R. & Gayathiri, P. (2012). Credit Card Fraud Detection Using Decision Tree for Tracing Email and IP, *International Journal of Computer Science* 9(5), 406-412.
14. Esmaily, J., Moradinezhad, R. & Ghasemi, J. (2015). Intrusion detection system based on Multi-Layer Perceptron Neural Networks and Decision Tree. *7th Conference on Information and Knowledge Technology (IKT)*, *Urmia* 1-5, doi: 10.1109/IKT.2015.7288736.
15. Falaki, S.O, Alese, B.K., Adewale, O.S., Ayeni, J., Aderounmu, G.A. & Ismaila, W.O. (2012). Probabilistic Credit Card Fraud Detection System in Online Transactions. *International Journal Software Engneering Application* 6(4), 69–78.

16. Fashoto, S., Adeleye, O. & Wandera, J. (2016). Hybrid Methods for Credit Card Fraud Detection Using K-means Clustering with Hidden Markov Model and Multilayer Perceptron Algorithm. *British Journal of Applied Science & Technology* 13(5), 1-11.
17. Ghosh, S., Reilly, D.L. (1994). Credit card fraud detection with a neural-network. In Nunamaker, J. F., Sprague, R. H. (Eds.) Proceedings of the 27th annual Hawaii international conference on system science, Vol 3: Information systems: DSS/knowledge-based systems (pp. 621–630).
18. Khan, M.Z., Pathan, J.D., & Ahmed, A. H. (2014), Credit Card Fraud Detection System Using Hidden Markov Model and K-Clustering, *International Journal of Advanced Research in Computer and Communication Engineering* 3(2), 2319-5940.
19. Navanshu, K. & Saad, Y. S. (2018), Credit Card Fraud Detection using Machine Learning models and collating Machine Learning models. *International Journal of Pure and Applied Mathematics* 118(20), 825-838.
20. Neda N., Leila B. & Ebrahim N. (2012). Surveying Different Aspects of Anomaly Detection and Its Applications. The Journal of Mathematics and Computer Science, 4 (2):129-138.
21. Patidar, R., & Sharma, L. (2011). Credit card fraud detection using neural network. International Journal of Soft Computing and Engineering (IJSCE), 1, 32-38
22. Patel, S. & Gond, S. (2014). Supervised Machine (SVM) Learning for Credit Card Fraud Detection. *International Journal of Engineering Trends and Technology* 8(3), 137-139.
23. Pooja, C., Thakare, A.D., Prajakta, K., Madhura, G. & Priyanka, N. (2015). Genetic K-means Algorithm for Credit Card Fraud Detection. *International Journal of Computer Science and Informaion Technologies* 6(2), 1724-1727.
24. Pouramirarsalani, A., Khalilian, M., & Nikravabshalmani, A. (2017). Fraud Detection in E-banking by Using the Hybrid Feature Selection and Evolutionary Algorithms. *International Journal of Computer Science and Network Security* 17(8), 271-279.
25. Raj, S. B. & Portia, A. A. (2011). Analysis on Credit Card Fraud Detection Methods. *International Conference on Conference on Computer Communication and Electrical Technology* https://doi.org/10.1109/ICCCET.2011.5762457.
26. Razak T. A. & Ahmed G. N. (2014). A Comparative Analysis on Credit Card Fraud Techniques Using Data Mining. International Journal of Data Mining Techniques and Applications, Integrated Intelligent Research (IIR), 3 (2): 398-400.
27. Rinky D. Patel and Dheeraj Kumar Singh (2013). Credit Card Fraud Detection & Prevention of Fraud Using Genetic Algorithm. *International Journal of Soft Computing and Engineering (IJSCE) ISSN: 2231-2307, Volume-2, Issue-6.*
28. Samaneh, S., Zahra, Z., Ebrahimi, A., & Amir, H. M. (2016). A Survey of Credit Card Fraud Detection Techniques: Data and Technique Oriented Perspective. *arXiv preprint arXiv:1611.06439*.
29. Y. Lee, Y. Yeh & Y. F. Wang *(2013), "Anomaly Detection via Online Oversampling Principal* Component Analysis*," in IEEE Transactions on Knowledge and Data Engineering, vol. 25, no. 7, pp. 1460-1470, , doi: 10.1109/TKDE.2012.99.*
30. Varun C., Arindam B. & Vipin K. (2009). Anomaly Detection: A Survey. ACM Computing Surveys, 41 (3):1-58.