Published by
NIGERIAN SOCIETY OF PHYSICAL SCIENCES
Available online @ https://journal.nsps.org.ng/index.php/jnsps

J. Nig. Soc. Phys. Sci. 4 (2022) 769

Journal of the Nigerian Society of Physical Sciences

Application of Machine Learning and Resampling Techniques to Credit Card Fraud Detection

Chinedu L. Udeze, Idongesit E. Eteng*, Ayei E. Ibor

Department of Computer Science, University of Calabar, Calabar, Cross River State, Nigeria

Abstract

The application of machine learning algorithms to the detection of fraudulent credit card transactions is a challenging problem domain due to the high imbalance in the datasets and confidentiality of financial data. This implies that legitimate transactions make up a high majority of the datasets such that a weak model with 99% accuracy and faulty predictions may still be assessed as high-performing. To build optimal models, four techniques were used in this research to sample the datasets including the baseline train_test_split method, the class weighted hyperparameter approach, and the undersampling and oversampling techniques. Three machine learning algorithms were implemented for the development of the models including the Random Forest, XGBoost and TensorFlow Deep Neural Network (DNN). Our observation is that the DNN is more efficient than the other 2 algorithms in modelling the under-sampled dataset while overall, the three algorithms had a better performance in the oversampling technique than in the undersampling technique. However, the Random Forest performed better than the other algorithms in the baseline approach. After comparing our results with some existing state-of-the-art works, we achieved an improved performance using real-world datasets.

DOI:10.46481/jnsps.2022.769

Keywords: Machine learning, Fraud detection, Random forest, Resampling techniques, XGBoost, TensorFlow, Deep neural network

Article History :
Received: 18 April 2022
Received in revised form: 25 July 2022
Accepted for publication: 01 August 2022
Published: 15 August 2022

Communicated by: T. Latunde

1. Introduction

In the USA, fraudulent activities amount to a loss of more than 12 billion dollars by 2020 [1]. Fraudulent records often make up about 0.2% of datasets making it a needle in the haystack task. Credit card fraud detection is often carried out using large datasets and historical data. Fraud often takes place when the card is lost, stolen or cloned by adversaries who perform third-party fraud by impersonating the identity of the original user.

Fraud could happen when a point of sales (POS) is compromised. False detection could lead to unjustified blocking of credit cards without fraudulent transactions which could lead to customer complaints and loss of reputation whereas non-blocking of violated cards can lead to huge financial fraud. In general, fraud detection can be categorized as a case of binary classification like other similar problems such as spam filtering. Such classification can be tackled using approaches like decision trees, support vector machines (SVM), k-nearest neighbour, logistic regression, random forest, XGBoost, neural networks and others.

The major contributions of this research are:

*Corresponding author tel. no: +2348038722277
Email address: ideteng@unical.edu.ng (Idongesit E. Eteng)

1. The comparative analysis of results from four dataset sampling techniques using some evaluation metrics.
2. Use of machine learning (ML) and deep learning techniques to model fraud detection in real-world datasets using Python libraries.

2. Related Works

Initially, rule-based management systems were used for creating fraud patterns, but this became too complex for manual analysis and Machine Learning (ML) techniques were adopted. ML techniques have also been applied to solve other complex problems [2-6]. Although neural networks emerged first, they have the limitation of using a black-box model. Consequently, Random forests, which could provide a description for the reason why a transaction was fraudulent and also avoid over-fitting became more applicable. ML needs online algorithms to learn from streams with a continuous flow of information.

Sarno et al. [7] devised a method of obtaining fraudulent records from transaction datasets using a process mining technique. The records comprise variables that are used together with professional expertise to build a collection of association rules. Association rule-based methods have the limitation of the inability to deal with the imbalance between negative and positive transactions. Statistical models such as logistic regression, multiple discriminant analysis, regression analysis and others are applicable in financial data mining.

Ivo et al. [8] used a dataset provided by fraud detection and mitigation experts from Feedzai comprising 5600 million transactions that have been anonymized as a security and privacy measure. The dataset contained only 0.05% fraudulent transactions. They successfully detected 80% of the data that had fraud with 70% accuracy by using the IBM Proactive Technology Online (PROTON) as their open-source baseline engine. They classified credit card fraud as offline and online fraud. The limitation of their research is that they were not able to correctly establish the precision and recall of their solution, partly because the dataset was historical and anonymized. They recommended the use of learning techniques to create rules after analysis of historical data.

Ishan et al. [9] observed that Random Forest is more accurate for identifying non-fraudulent cases while feed-forward Neural Network does better in the detection of fraudulent transactions. In their research, they combined both algorithms in an ensemble machine learning method to achieve better accuracy in detection. Ensemble learning involves the combination of different classifiers to improve the performance and predictive ability of the model. They identified the following challenges in the detection of credit card fraud: unavailability of datasets, dynamic behaviour of fraudsters, highly skewed datasets and parameters for evaluation.

The dataset they used was made up of credit card transactions by Europeans in September 2013 consisting of a total of 284, 807 transactions with 0.172% fraudulent records. To discover whether fraudulent records are outliers or clusters they applied unsupervised learning methods using k-means and other clustering techniques but found out that the fraudulent transactions were rather uniformly distributed in the clusters. The limitation of their work is that some of their classifiers did not attain optimal accuracy and the scope of their work was limited to a dataset with numerical values and not text.

Phuong et al. [10] used anomaly detection methods to perform credit card fraud detection using two data-driven approaches. The dataset consists of real data of European credit card users for e-commerce transactions. The dataset has some numerical values obtained from a principal component analysis (PCA) transformation. In the training phase, they used 284000 transactions while in the testing phase they used 200 non-fraudulent transactions and 200 fraudulent transactions. With this, they realized a high detection accuracy and low false negatives and false positives. They used a one-class support vector machine (OCSVM) using the optimal kernel parameter selection and T2 control chart.

Artikis et al. [11] used the machine learning model of SPEEDD which constructs the fraud pattern and contains a user interface for fraud analysts. The ML component of SPEEDD can be used for the online development of fraud patterns permitting it to adjust to the dynamic nature of fraud varieties. They were able to develop a good prototype through the assistance of the Feedzai company which specializes in ML fraud detection.

The algorithm tries to learn the patterns of fraudulent transactions using Inductive Logic Programming (ILP), a technique that uses the divide-and-conquer strategy. ILP makes use of logic programming to represent training sets and learnt rules in learning a logical theory called hypothesis which describes the pattern of the fraudulent records. They used the Online Learning of Event Definitions (OLED) to resolve problems of velocity and volume in training sets. OLED implements a heuristic for searching using statistical measures for learning with only a little portion of the records.

The OLED and ILP algorithms were developed with the Scala language, using the Clingo solver for reasoning. They used precision as the evaluation criteria for the rules. The limitation of their research is that the obtained precision, recall and runtime need further improvement. Kang et al. [12] used a convolutional neural network based on the principle of the animal visual cortex to perform classification to determine when transactions are fraudulent.

Abakarim et al. [13] designed a live credit card fraud detection system that implements a deep neural network (DNN) technology that uses an auto-encoder for the classification of operations as legitimate or illegitimate. The DNN is made up of 6 hidden layers comprising 3 encoders and 3 decoders. The DNN is modelled after the biological structure of human neurons made up of multilevel concealed layers of processing units that communicate with each other.

Their model performs classification on a live feed of credit card transactions making real-time decisions. In the first phase, they developed an ML model through periodic offline training of historical data. This stage is used for the transformation of the credit card operations into features and labels that would aid the classifier. After this, the dataset is broken into training and test sets used for training and testing the model.

The second stage involves predicting with the model on a live stream of transactions. To achieve this they used the symbolic math library of the open-source Python Tensorflow together with Keras a neural network library, and an Apache distributed streaming system Kafka that works with the Memsql. They compared their solution with the results of four other classifiers and recorded fair precision, accuracy and recall. The algorithms they used for comparison include linear SVM, logistic regression, non-linear auto ANN and NN-based classifier. The limitation of their solution is that although they achieved an improved recall and F1 score, the precision was relatively low. Another weakness is that they implemented simple DNN which could have been improved with more hyper-parameter tuning.

Lucas et al. [14] proposed a system that creates history-based features using Hidden Markov Models (HMM). The features are generated by computing the likelihood of a set of operations being legitimate. They measured the common traits between a transaction and previous fraudulent or normal operations recorded for the cardholders. Most current feature engineering methods used for this purpose generate descriptive features of card-holder historical data. This approach could be limited because it does not take into account the history of the account holder.

However HMM remediates these drawbacks as a generative probabilistic model for sequence modelling. Hence, these authors used a system with eight HMM-based features that were used to study the similarity between historical data and eight distributions previously learned with an unsupervised algorithm to obtain a fitting model. The HMM models used four training sets containing genuine and compromised credit cards and terminals. They developed the program using Python HMM learn. They used a dataset comprising 4.7 * 107 anonymized credit card transactions from 01/03/2015 to 31/05/2015 to measure the increase in the HMM detection. The dataset was split into a training set, validation set and testing set. They also trained a random forest algorithm to compare the accuracy of the system when HMM features are integrated.

Deshan et al. [15] analysed the dataset of European credit cardholders by applying sampling and resampling techniques to mitigate the effects of the unbalanced nature on the models. They applied some metrics in the evaluation of the performance of classification models on the unbalanced dataset. With five decision-tree-based algorithms, they were able to train the dataset, and determine the optimal model using cross-validation. With their research, they established a scheme for the analysis and prediction of credit card fraud.

Yuxin et al. [16] used a similar dataset to train machine learning models using Logistic Regression, Gradient Boosting Tree Model and 5-layer neural network with three hidden layers. The first hidden layer consists of 120 neurons, the second 60 neurons and the last 30 neurons. They realized that out of the three algorithms, Logistic Regression had the least performance. However, the tree model seems to involve overfitting and requires further regularization.

3. Materials and Methods

3.1. Dataset

The dataset used for this research comprises 30 features and was obtained from the Kaggle website [17]. It was derived from credit card transactions of European cardholders generated over 2 days in 2013. It has been pre-processed using Principal Component Analysis which transformed the columns V1 to V28 to numerical values, except 'Amount' and 'Time' to avoid exposure of sensitive data to privacy violations. The dataset contains 284, 807 records which comprise 284, 315 legitimate and 492 fraudulent transactions.

3.2. Methodology

The dataset only contains about 0.173% fraudulent transactions, hence, it is highly unbalanced. To train a more balanced sample of the dataset for optimal models that are not skewed towards the majority of the non-fraudulent transactions, the following four techniques were applied separately with different results in the confusion matrix:

1. **The baseline approach**: The dataset was first split into a training set and test set using a test size of 20% while the training set was further split into the training set and validation set using the same proportion with the train_test_split function.
2. **The class weighted approach**: In the Random Forest Classifier, a 'balanced' class weight was used to enhance the model. For the XGBoost Classifier, the *scale_pos_weight* was computed by taking the square root of the ratio of the frequency of the legitimate to the frequency of the fraudulent transactions. For the TensorFlow DNN, the reciprocal of the frequencies of the two classes were used in computing the class weight used for training the model.
3. **Oversampling method:** Using the ADASYN from the imbalanced-learn (imblearn) library, oversampling techniques were applied such that the original data contained 181961 legitimate and 315 fraudulent transactions but after the oversampling, it contained 181961 legitimate and 181956 fraudulent records.
4. **Undersampling method:** The RandomUnderSampler from the imblearn library was also applied to the dataset and it reduced the samples to 315 legitimate and 315 fraudulent data.

As shown in Figure 1, first, the dataset was split into training and testing sets like in the baseline approach. If a resampling technique is to be applied, then a balanced dataset is created before the training and validation of the model.

At this stage, the Random Forest, XGBoost (Extreme Gradient Boosting) and TensorFlow DNN (Deep Neural Network) algorithms were applied in training the model for each of the four ways of sampling the dataset. The confusion matrix for each of the models is obtained as the output of the Python programs which could be used to compute the evaluation metrics. The program also generates the numerical value for the AUPRC (Area Under the Precision-Recall Curve) for each model.

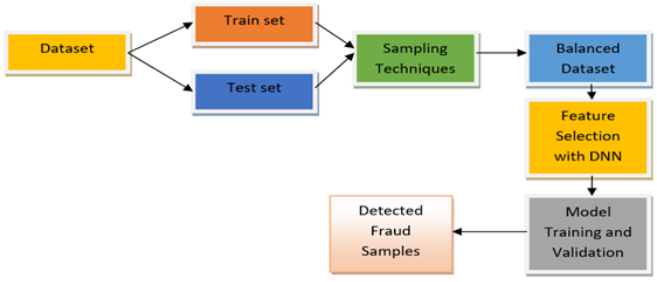

Figure 1. The architecture of the Proposed System

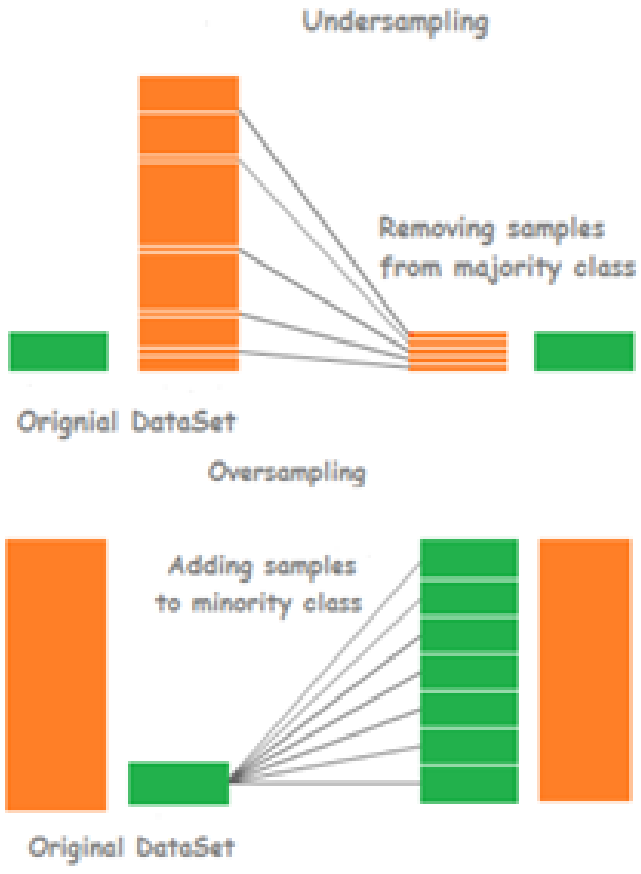

Figure 2. Undersampling and Oversampling [18]

3.2.1. *Random Forest*

Random Forest algorithms function like bagged decision trees with the major exception that every tree can only split on a subset of features m. In this case, the features m must be different for every classifier. In the Random Forest Classifier, a 'balanced' class weight was used to enhance the model. In running the code, the algorithm performed better than the other algorithms in the baseline approach. In support of [9], while running the experiments, Random Forest was more accurate for identifying non-fraudulent cases but not fraudulent cases. It was also noticed that the algorithm prevents overfitting of data and is fast to train with test data.

Table 1. Model of Confusion Matrix

		Predictions from the model	
		1	0
Actual Class	1	True Positive (TP)	False Negative (FN)
	0	False Positive (FP)	True Negative (TN)

Table 2. Confusion matrix after baseline splitting

		Fraudulent	**Legitimate**
		Random Forest	
	Fraudulent	77	21
	Legitimate	4	56860
		XGBoost	
Actual Class	Fraudulent	80	18
	Legitimate	6	56858
		TensorFlow DNN	
	Fraudulent	77	21
	Legitimate	14	56860

Table 3. Evaluation metrics after baseline splitting

	Accuracy	**Precision**	**Recall**	**F1-Score**
Random Forest	0.99956	0.95062	0.78571	0.86033
XGBoost	0.99958	0.93023	0.81632	0.86956
TensorFlow DNN	0.99944	0.84615	0.78571	0.81481

3.2.2. *XGBoost*

XGBoost is an implementation of gradient boosted decision trees. For the XGBoost Classifier used in our experiment, the weight used was computed by taking the square root of the ratio of the frequency of the legitimate transactions to the frequency of the fraudulent transactions. Overall, this algorithm performed better using the oversampling technique. This is supported by results reported in Tables 2 to 9.

3.2.3. *TensorFlow DNN*

The TensorFlow DNN was run with 200 epochs but implemented an Early Stopping that could prevent it from making all the iterations if a desirable result is achieved. It also plots the curves for the train, loss and validation loss together, and then it generates another plot of the training accuracy and validation accuracy using the number of epochs as the x-axis and also showing the AUPRC.

3.2.4. *Resampling techniques*

Due to heavy imbalance in the dataset, undersampling and oversampling methods were applied in some of the machine learning models used in the system. Undersampling involves reducing the number of majority class samples while oversampling is achieved by multiplying the number of minority class samples by repeating some records. The synthetic minority oversampling technique (SMOTE) is a popular example of an oversampling technique, however, in this research, we have implemented the ADASYN Python library for oversampling.

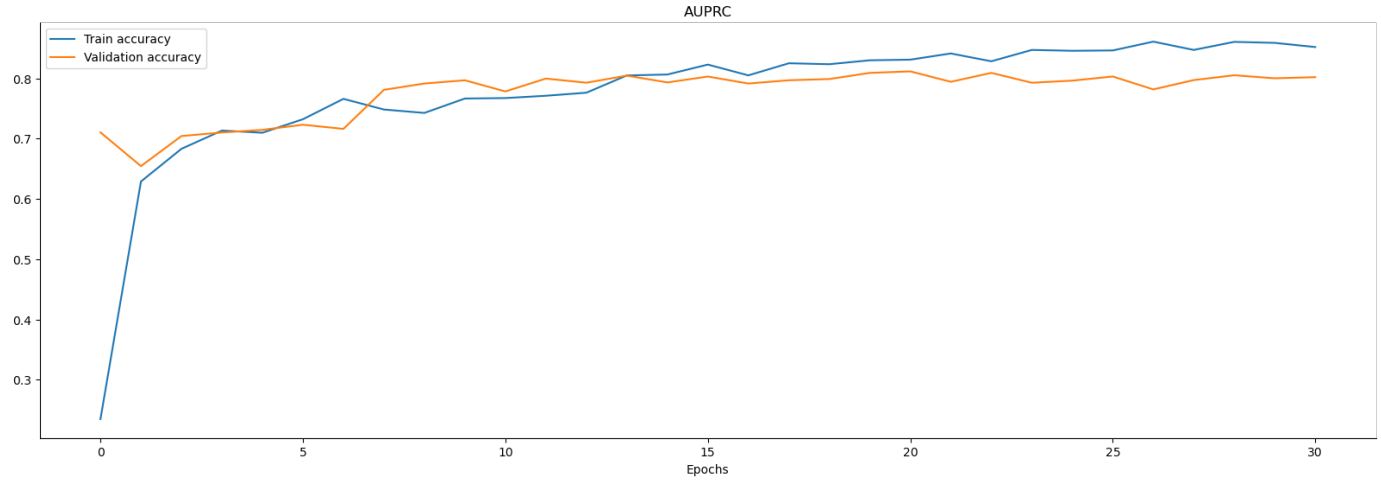

Figure 3. Train and validation accuracy curves using the baseline approach with the TensorFlow DNN

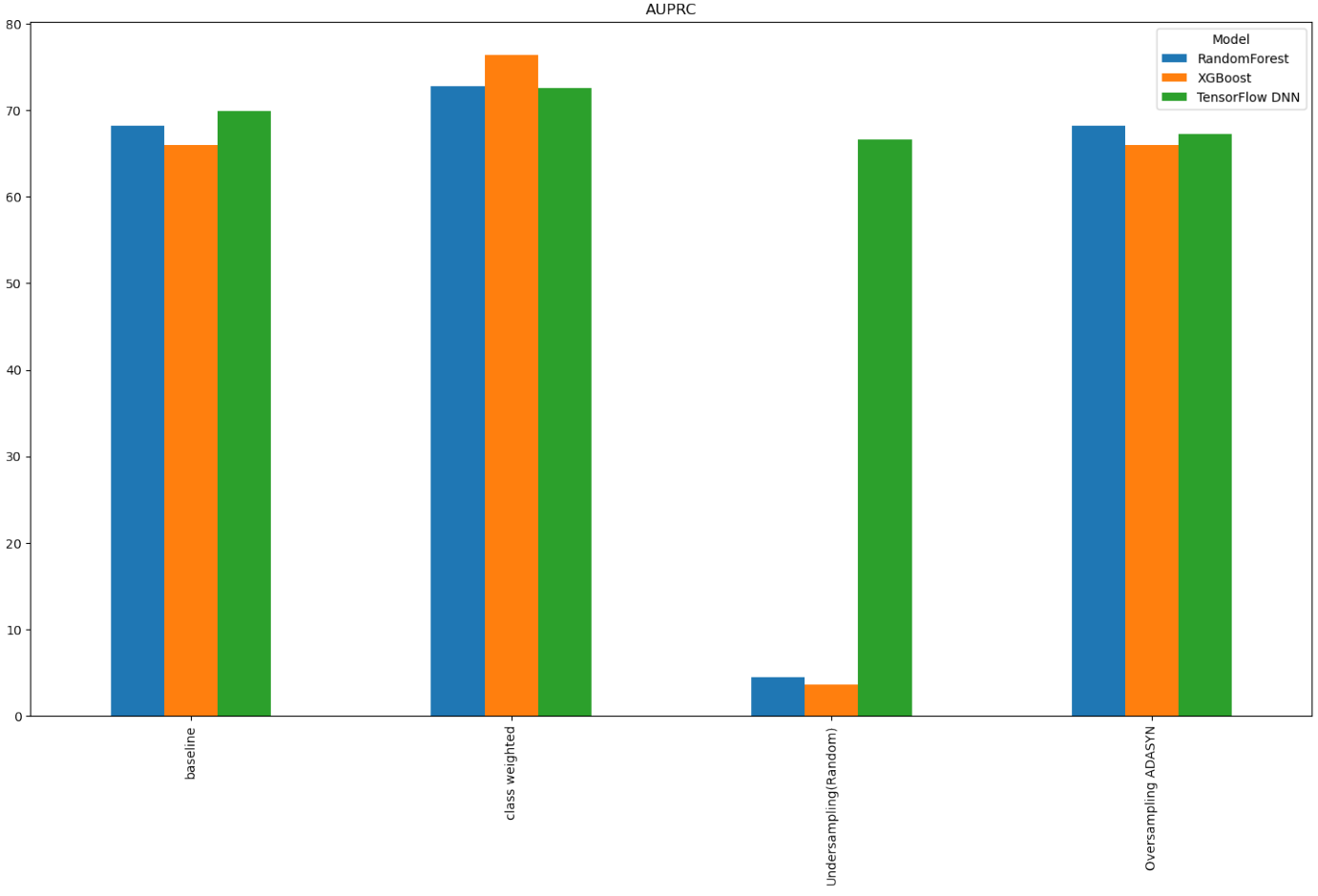

Figure 4. Bar chart showing Area Under Precision-Recall Curve for the 4 sampling methods

Roweida *et al.* [18] took studies with two resampling techniques and some classifiers on a Kaggle dataset used to predict customers that will make specific transactions in the future, using Scikit-learn, NumPy and Pandas. They implemented a non-heuristic algorithm for oversampling to balance class distribution by a random iteration of minority class records. One of the limitations of such an approach is overfitting since it replicates similar records of the minority class. However, oversampling could be beneficial when the dataset size is small, or when a particular class is small making the dataset to be skewed to-

Table 4. Class weighted model confusion matrix

		Fraudulent	Legitimate
		Random Forest	
	Fraudulent	74	24
	Legitimate	3	56861
		XGBoost	
Actual Class	Fraudulent	82	16
	Legitimate	8	56856
		TensorFlow DNN	
	Fraudulent	88	10
	Legitimate	576	56288

Table 5. Class weighted model evaluation metrics

	Accuracy	Precision	Recall	F1-Score
Random Forest	0.99953	0.96104	0.75510	0.84571
XGBoost	0.99958	0.91111	0.83673	0.87234
TensorFlow DNN	0.98971	0.13253	0.89796	0.23097

Table 6. Random undersampling confusion matrix

		Fraudulent	Legitimate
		Random Forest	
	Fraudulent	90	8
	Legitimate	1780	54707
		XGBoost	
Actual Class	Fraudulent	89	9
	Legitimate	2157	56856
		TensorFlow DNN	
	Fraudulent	87	11
	Legitimate	320	56544

Table 7. Random undersampling evaluation metrics

	Accuracy	Precision	Recall	F1-Score
Random Forest	0.96840	0.04813	0.91837	0.0884
XGBoost	0.96335	0.03963	0.90816	0.07595
TensorFlow DNN	0.99419	0.21376	0.88776	0.34456

Table 8. Adasyn Oversampling confusion matrix

		Fraudulent	Legitimate
		Random Forest	
	Fraudulent	77	21
	Legitimate	12	56852
		XGBoost	
Actual Class	Fraudulent	85	13
	Legitimate	27	56837
		TensorFlow DNN	
	Fraudulent	82	16
	Legitimate	70	56794

Table 9. Adasyn oversampling evaluation metrics

	Accuracy	Precision	Recall	F1-Score
Random Forest	0.99942	0.86517	0.78571	0.82353
XGBoost	0.99930	0.75893	0.86735	0.80953
TensorFlow DNN	0.99849	0.53947	0.83673	0.65600

wards the majority class instances.

They also implemented a random undersampling algorithm, a non-heuristic method of balancing class spreading by reducing the number of records in the majority class. The drawback is that it can result in the loss of information through the elimination of valuable data. Nevertheless, when the dataset size is large, the effect can be negligible. Their results show that oversampling techniques give better performance with the classifier models when compared to the undersampling techniques. A depiction of undersampling and oversampling techniques is given in Figure 2.

3.3. Evaluation Metrics

The following evaluation metrics are used to assess the performance of our approach as well as other comparative approaches. These are standard evaluation metrics, which have been used in several other works [8, 19].

1. **Confusion matrix:** This is a table that shows the performance of a machine learning model. The rows contain the results from an actual class while the column represents the predictions from an algorithm.
   In Table 1, 1 stands for fraudulent transaction while 0 stands for a legitimate transaction, such that TN is for legitimate transactions correctly detected, FP is for legitimate transactions classified as fraudulent, FN is for fraudulent transactions classified as legitimate and TP is for a correctly detected fraudulent transaction.
2. **Accuracy:** This is the proportion of the right predictions (TP + TN) amidst the entire number of transactions examined.
$$ACC = \frac{TP + TN}{TP + TN + FP + FN}$$
3. **Precision:** This is the measure of the ability of the model to correctly predict a positive value. It is given as the proportion of true positives from the actual total number of transactions in the positive class. A precision of 80% implies that out of every 100 detected fraud, at least 80 of them are correctly inferred.
$$Precision = \frac{TP}{TP + FP}$$
4. **Recall:** This is also known as the sensitivity of the model and it is given as the proportion of true positives in the entire collection of transactions that were in the original positive class. A recall of 90% implies that of every 100 cases of fraud that traverses through the system, at least 90 are detected.
$$Recall = \frac{TP}{TP + FN}$$

Table 10. AUPRC Figures for the algorithms and sampling techniques

	Baseline approach	Class weighted	Undersampling	Oversampling
Random Forest	0.74889	0.72789	0.04467	0.68173
XGBoost	0.76111	0.76392	0.03648	0.65952
TensorFlow DNN	0.84296	0.72615	0.66570	0.67274

5. **F1-score:** This is the harmonic mean of the recall and precision

$$F1 = \frac{2 * Precision * Recall}{Precision + Recall}$$

4. Results and Discussion

4.1. Experimental Testbed

The computation was carried out on a Windows 11 64-bit Operating system with 8GB RAM and 500GB SSD. Python 3.9.7 was used on an Anaconda Navigator and Jupyter environment. The major libraries used include TensorFlow, Sklearn, Matplotlib, Pandas, Imblearn, Xgboost, Numpy, Seaborn and Math. The code samples from [20] were resourceful in optimizing our models.

4.2. Results

The results of experimentation are tabulated in Tables 2 to 9 and discussed in this section.

4.3. Discussion of results

In the baseline splitting of datasets, the other two decision-tree-based algorithms achieved better performance than the deep learning model as shown in Tables 2 to 3. For both the baseline dataset splitting and the class weighted approach (Tables 2 to 5), the XGBoost achieved a better performance than the other two algorithms while the DNN resulted in relatively low precision and F1 score. After the random undersampling, the performance of the three algorithms were reduced with a very poor precision by the random forest and XGBoost models as tabulated in Tables 6 and 7, however, although the DNN performance was reduced, it produced a better accuracy, precision and F1-score than the other two models. After the ADASYN oversampling, the performance of the three algorithms was better than the undersampling results and the random forest model produced a better accuracy, precision and F1 score than the other two algorithms. This is illustrated in Tables 8-9. A summary of these results is given in Table 10. Furthermore, the train and validation accuracy curves of the TensorFlow DNN are given in Figure 3. Similarly, in Figure 4, the comparative performance of the four sampling methods is depicted.

4.4. Comparison of results

Our work was compared with the some existing work [9], [13], [15] and [16]. This comparison, which was based on the accuracy and FI scores of each model s tabulated in Table 11.

From Table 11, it can be observed that our approach demonstrated promising performance in terms of accuracy and F1 in comparison to other approaches using the same dataset. Having used three algorithms and four sampling techniques, we arrived at the best performance using the XGBoost with the class-weighted approach.

Table 11. Comparison of Approaches

Approach	Accuracy	F1
Yuxin et al. [16]	0.9993	0.777
Deshan et al. [15]	-	0.850
Abakarim et al. [13]	0.9861	0.294
Ishan et al. [9]	0.9995	-
Our Approach	0.99958	0.87234

Furthermore, the Random Forest model also achieved good performance with the oversampling technique while the TensorFlow DNN also resulted in a fair accuracy using the undersampling technique. This also affirms that neural networks could outperform other models when the dataset size is small. Overall, observation from the AUPRC figures of the three algorithms used shows that the TensorFlow DNN had more stable values and obtained the highest AUPRC so far using the baseline approach.

From this, we have shown that the use of machine learning algorithms and deep neural work can be applied to the problem of fraud detection with significant results. This is very useful since deep neural networks can learn from data by understanding the representations in the data through input-output mapping [20].

5. Conclusion

In this research, we conducted a comparative analysis of the performance of three machine learning algorithms in the detection of credit card fraud. Since the fraud cases constituted a minority group in the dataset, ADASYN oversampling and random undersampling techniques were used for resampling to create uniformly distributed classes. Asides from these two approaches, results were also obtained with the baseline train_test_split method and the class weighted method. Evaluation metrics were defined for the machine learning and deep learning classifiers including Random Forest, XGBoost and TensorFlow DNN. Each of these algorithms has special cases where they are best fitted from the results obtained.

In the future, it will be expedient to create an ensemble of machine learning and deep learning algorithms to take advantage of the strengths of each of the models. It is also important to try some resampling techniques like SMOTE, more machine learning algorithms and another dataset that is not entirely made up of numeric values.

Acknowledgments

We thank the referees for the positive enlightening comments and suggestions, which have greatly helped us in making improvements to this paper.

References

[1] R. Aitken, "U.S. card fraud losses could exceed 12B USD by 2020", Forbes, (2016), http://www.forbes.com/sites/rogeraitken/2016/10/26/us-card-fraud-losses-could-exceed-12bn-by-2020/

[2] V. Umarani, A. Julian & J. Deepa, "Sentiment analysis using various machine learning and deep learning Techniques", Journal of the Nigerian Society of Physical Sciences (2021) 385.

[3] D. O. Oyewola, E. G. Dada, J. N. Ndunagu, T. A. Umar & S. A. Akinwunmi, "COVID-19 risk factors, economic factors, and epidemiological factors nexus on economic impact: machine learning and structural equation modelling approaches", Journal of the Nigerian Society of Physical Sciences **3** (2021) 395.

[4] A. B. Yusuf, R. M. Dima & S. K. Aina, "Optimized breast cancer classification using feature selection and outliers detection", Journal of the Nigerian Society of Physical Sciences **3** (2021) 298.

[5] O. E. Ojo, A. Gelbukh, H. Calvo & O. O. Adebanji, "Performance study of N-grams in the analysis of sentiments", Journal of the Nigerian Society of Physical Sciences **3** (2021) 477.

[6] O. Olubi, E. Oniya, & T. Owolabi, "Development of predictive model for radon-222 estimation in the atmosphere using stepwise regression and grid search based-random forest regression", Journal of the Nigerian Society of Physical Sciences **2** (2021) 132-139.

[7] R. Sarno, R. D. Dewandono, T. Ahmad, M. F. Naufal & F. Sinaga, "Hybrid association rule learning and process mining for fraud detection", IAENG International Journal of Computer Science **42** (2015) 59.

[8] C. Ivo, F. Fabiana & S. Inna, "Industry paper: The uncertain case of credit card fraud detection," Proceedings of the 9th ACM International Conference on Distributed Event-based Systems, (2015), https://dl.acm.org/doi/10.1145/2675743.2771877

[9] S. Ishan, P. Rameshwar & N. Ullas, "Ensemble learning for credit card fraud detection", The ACM India Joint International Conference on Data Science and Management of Data, (2018), https://dl.acm.org/doi/10.1145/3152494.3156815

[10] H. T. Phuong, P. T. Kim, T. H. Truong, H. Cedric, H. T. Phuong & H. L. Thi, "Real time data-driven approaches for credit card fraud detection", Proceedings of the 2018 International Conference on E-business and Applications, (2018), https://dl.acm.org/doi/10.1145/3194188.3194196

[11] A. Artikis, N. Katzouris, I. Correia, C. Baber, N. Morar, I. Skarbovsky, F. Fournier & G. Paliouras,"A prototype for credit card fraud management: industry paper", The Proceedings of the 11th ACM International Conference on Distributed and Event-Based Systems (2017), https://dl.acm.org/doi/10.1145/3093742.3093912

[12] F. Kang, C. Dawei, T. Yi & Z. Liqing, "Credit card fraud detection using convolutional neural networks," International Conference on Neural Information Processing. Springer (2016) 483, https://www.springerprofessional.de/en/credit-card-fraud-detection-using-convolutional-neural-networks/10799390

[13] Y. Abakarim, M. Lahby & A. Attioui, "An efficient real time model for credit card fraud detection based on deep learning," The Proceedings of the 12th International Conference on Intelligent Systems: Theories and Applications **30** (2018) 1, https://dl.acm.org/doi/10.1145/3289402.3289530

[14] Y. Lucas, P.-E. Portier, L. Laporte, S. Calabretto, O. Caelen, L. He-Guelton & M. Granitzer, "Multiple perspectives HMM-based feature engineering for credit card fraud detection", The Proceedings of the 34th ACM/SIGAPP Symposium on Applied Computing (2019) 1359, https://dl.acm.org/doi/10.1145/3297280.3297586

[15] H. Deshan, L. Yu, W. Zhaoxing & X. Jiajie, "Decision analysis and prediction based on credit card fraud data", The 2nd European Symposium on Computer and Communications (ESCC '21), Belgrade, Serbia. ACM, New York, NY, USA (2021), https://doi.org/10.1145/3478301.3478305

[16] G. Yuxin, Z. Shuoming & L. Jiapeng, "Machine learning for credit card fraud detection", Proceedings of the 2021 International Conference on Control and Intelligent Robotics (2021), https://dl.acm.org/doi/abs/10.1145/3473714.3473749

[17] Kaggle, *Credit Card Fraud Detection*, (2022), https://www.kaggle.com/mlg-ulb/creditcardfraud

[18] M. Roweida, R. Jumanah & A. Malak, "Machine learning with oversampling and undersampling techniques: overview study and experimental results", 11th International Conference on Information and Communication Systems (2020).

[19] A. E. Ibor, O. B. Okunoye, F. A. Oladeji, andK. A. Abdulsalam, "Novel hybrid model for intrusion prediction on cyber-physical systems' Communication Networks based on Bio-inspired Deep Neural Network Structure", Journal of Information Security and Applications **65** (2022).

[20] G. Zoto, "Credit card fraud detection using ML and deep learning", YouTube, (2020), https://www.youtube.com/watch?v=yX1_iDV0E50