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

J. Nig. Soc. Phys. Sci. 7 (2025) 2066

Journal of the Nigerian Society of Physical Sciences

A stacked ensemble approach with resampling techniques for highly effective fraud detection in imbalanced datasets

Idongesit E. Eteng[a,*], Udeze L. Chinedu[b], Ayei E. Ibor[a]

[a]*Department of Computer Science, University of Calabar, Calabar, Nigeria*
[b]*Department of Computer Science and Creative Technologies, University of the West of England, Bristol, United Kingdom*

Abstract

In several earlier studies, machine learning (ML) has been widely explored for fraud detection. However, fraud detection is still a challenging problem. This is due to the imbalanced nature of fraud data, which leads to underperformance by most models in detecting a few fraud cases. Undetected fraud cases also account for the loss of several millions of dollars annually. Thus, we propose an ensemble approach that stacks five classifiers - Support Vector Machine, Decision Trees, Random Forests, Gaussian Naïve Bayes, and k-Nearest Neighbour, and uses the Logistic Regression meta-classifier to make predictions based on a stacking algorithm and novel pipeline. The effectiveness of the proposed model is examined on three datasets. The first two datasets were trained and tested initially without resampling and then compared with the results obtained using the Synthetic Minority Oversampling Technique (SMOTE) and RandomUnderSampler techniques. Only a balanced resampled dataset was trained on the third dataset that clearly showed an imbalance. From the results obtained, it is observed that the proposed model is highly competitive, with extant models producing ROC_AUC of 99% and scoring above 98% in all other metrics. The approach is recommended for detecting fraud cases in similar case studies.

DOI:10.46481/jnsps.2025.2066

Keywords: Imbalanced dataset, Ensemble approach, Fraud detection, Stacking algorithm, Synthetic Minority Oversampling Technique (SMOTE)

Article History :
Received: 08 April 2024
Received in revised form: 06 July 2024
Accepted for publication: 20 August 2024
Available online: 14 December 2025

Communicated by: Oluwatobi Akande

1. Introduction

The rise in financial crimes in recent times has tremendously impacted world economies with attendant consequences on economic prospects. Credit card fraud falls under financial crimes, and the ubiquity of Internet technology has resulted in the multiplicity of these crimes. Current financial transactions over wired and wireless networks are initiated and fulfilled through credit cards. These cards are usually encoded with customer confidential information and have become the target of varying degrees of fraud amounting to billions of dollars. For instance, Aitken *et al.* [1] asserted that fraudulent activities in the United States have resulted in a loss of more than $12 billion as of 2020. Credit card fraud usually occurs in two dimensions. As opined by Itoo *et al.* [2], these two dimensions include inner and external card fraud. Inner card fraud usually involves using a false identity to commit fraud due to the mutual agreement between the cardholders and their respective banks. Conversely,

*Corresponding Author Tel. No.: +234-803-872-2277.
Email address: ideteng@unical.edu.ng (Idongesit E. Eteng)

external card fraud involves receiving cash from unsuspecting victims using deceit. In the scenarios above, the effect of credit card fraud has kept financial institutions such as banks on a persistent search for a robust solution.

According to Huang *et al.* [3], the cumulative effect of credit card fraud can lead to excess currency liquidity and bad debts arising from credit card overdrafts that may create financial disruptions in nation-states. Furthermore, electronic banking paves the way for credit card fraud as more financial transaction gateways become vulnerable. With significantly vast amounts of monitored accounts across all financial transaction platforms and institutions, the big data generated from daily credit card transactions makes it difficult for human experts to detect the few instances of this fraud visibly. Most importantly, the imbalance in data distribution between legitimate and fraudulent transactions occasioned by using credit cards skews all observations towards non-fraudulent cases [4]. Consequently, most models cannot handle the class imbalance often found in datasets, thus underperforming in detecting the few fraudulent cases in the big data of credit card transactions. It's crucial to underscore that erroneous detections may trigger unwarranted credit card blocks, resulting in customer complaints and reputational damage to the institutions involved. Conversely, failure to block compromised cards can lead to substantial financial fraud. Generally, fraud detection is akin to binary classification challenges like spam filtering. Various methodologies, including decision trees, support vector machines (SVM), k-nearest neighbour (KNN), logistic regression, random forest, XGBoost, and neural networks, can address this classification. This paper proposes an ensemble model by stacking five models using Logistic Regression as a meta-classifier. These models include SVM, KNN, Random Forest (RF), Gaussian Naive Bayes (NB), and Decision Trees (DT). We use the stacking ensemble to address the problem of poor generalization of each model on the imbalanced dataset to improve detection accuracy. The choice of models was guided by a desire to leverage diverse algorithmic strengths. While Decision Trees and Random Forests are related, they contribute differently to the ensemble. Decision Trees provide a simple and interpretable model, whereas Random Forests offer robustness and reduce overfitting through aggregation of multiple trees. Including both allows the logistic regression model to weigh their predictions and potentially capture nuances that might be lost if only one type were used. The inclusion of SVM, Gaussian Naive Bayes, and the other models ensures a blend of linear, non-linear, probabilistic, and deterministic perspectives.

For the imbalanced dataset, we first used the baseline approach for the small and medium-sized datasets. We used oversampling and undersampling to resample all the datasets to a more balanced version to achieve an optimal ensemble model that is not skewed towards most non-fraudulent transactions. While SMOTE effectively addresses class imbalance by generating synthetic minority class samples, it can sometimes lead to over-representation of certain regions in the feature space. By combining SMOTE with undersampling, we were able to achieve a more balanced and representative dataset. This hybrid approach helps in mitigating any bias that purely synthetic oversampling might introduce, ensuring a more generalised model.

The principal contributions of this study include the following:

1. A comparative examination of outcomes across three datasets of varying sizes employing several resampling techniques was evaluated through five distinct metrics.

2. A stacking ensemble machine learning (ML) technique consisting of five base classifiers and the logistic regression meta-classifier is used to simulate fraud detection using practical datasets using Python libraries. This involves comparing the individual performance of the algorithms relative to the stacked model.

3. This is a novel pipeline for the ensemble approach, with visualizations that help appreciate the models' performance using Python libraries and the Jupyter Notebook.

4. This paper compares results obtained on imbalanced datasets with results obtained from the resampled copy of the same datasets to describe the effect of resampling on the model.

The remaining sections of the work are ordered as follows: Section 2 contains related work bearing other research works previously done in the subject area. Section 3 contains the materials and methods which houses the descriptions of the algorithms and datasets used. Section 4 contains result and discussion which houses the visualizations and tables obtained from the code. Section 5 houses the conclusion and future work.

2. Related works

Machine Learning (ML) algorithms make predictions by observing phenomena and constructing models, and are categorized into supervised, unsupervised, and reinforcement learning. Supervised learning uses predefined labels for training, typically involving regression and classification techniques, while unsupervised learning uses unlabeled data to find patterns and relationships, focusing on dimensionality reduction and clustering. Yousefi *et al.* [5] classified credit card fraud into types such as application, card imprint, mail non-receipt, lost or stolen, counterfeit, and card-not-present. They identified supervised ML algorithms (LR, ANN, SVM, DT, RF, NB, KNN) and unsupervised techniques (K-Means Clustering, SOM) for fraud detection.

Rushin *et al.* [6] compared Logistic Regression (LR) with deep learning and gradient boosting trees using a dataset of 80 million transactions with 69 attributes. LR performed the worst due to difficulty in detecting obscure patterns, while deep learning was more efficient. Artificial Neural Networks (ANN) are effective for complex data but require significant resources and can overfit. Wang *et al.* [7] developed a privacy-preserving Deep Neural Network (DNN) that outperformed non-private approaches. Support Vector Machines (SVM) excel in high-dimensional spaces with good generalization and low computational complexity. Decision Trees (DT) are easy to implement

but sensitive to skewed distributions, and their performance is affected by tree-splitting criteria. Bahnsen *et al.* [8] created a cost-sensitive DT model that reduces false prediction losses and results in a simpler tree. Random Forest (RF) minimizes overfitting and noise by generating independent trees and effectively addresses concept drift and imbalanced datasets, outperforming algorithms like SVM and LR. Naive Bayes (NB), a probabilistic classifier based on Bayes' Theorem, is effective with high-dimensional data but assumes conditional independence among features

Mohammed *et al.* [12] illustrated that the Naïve Bayes algorithm exhibits faster processing speed than Random Forest and balanced bagging ensemble in fraud detection. However, it may result in false alarms due to its lower precision. Mahmud *et al.* [13] and Mahmoudi and Duman [14] proved that DT algorithms and Fisher discriminant analysis give better accuracy and performance, respectively, in classification than the NB. In KNN, an instance is classified based on the nearest K neighbours It has a low error rate relative to other methods like NB, DT and LR. Seeja and Zareapoor [15] conducted studies with the UCSD data mining contest 2009 dataset and developed a pattern mining algorithm that outperforms the KNN. However, their results show that KNN has a better false detection rate than SVM and a more balanced classification rate than RF and SVM.

Unsupervised learning algorithms are best suited for anomaly detection and can perform better than supervised learning algorithms in detecting new fraud patterns. Kumari and Choubey [16] combine K-Means and Hidden Markov Model (HMM) techniques in fraud detection. They first used K-Means to cluster the historical data of customers using their spending habits, while HMM was used to predict the probability of fraud. Behera and Panigrahi [17] used a KM-based system to cluster transactions using the spending behaviours of cardholders by classifying a transaction as abnormal if the proximity to the centre of a cluster exceeds the threshold. In such a case, they further applied a feed-forward NN to classify the suspicious transaction, and they achieved an actual positive rate of up to 93.9% on a simulated dataset.

Jiang *et al.* [18] adopted a KM-based approach to group cardholders into three categories. They used a window-sliding technique to compile the transactions into groups by utilising the customers' behavioural trends. With this, they achieved results that were better than those of the RF and LR methods using simulated datasets. SOM is an unsupervised NN modelling algorithm that aids in visualising transaction patterns using a repetitive tuning of the neuron weights in the network. Olszewski [19] implemented the SOM in visualising a multi-dimensional dataset using threshold-based binary classification techniques on a real dataset to develop a 2-dimensional image that even laypeople can comprehend. Agaskar *et al.* [20] built a fraud detection model that generates clusters using SOM and then revalidates the clusters with association rules. They used the amount and location information of older transactions by the customers. Credit card datasets often comprise an exceptionally high number of validtransactions and a small proportion of suspicious transactions, which makes them highly unbalanced. Such imbalance could lead to a false classification of transactions in the minority learning class. Vaishnavi and Geetha [21] researched the use of ML algorithms to solve the problem of concept drift in a real-world credit card dataset. They applied a clustering algorithm to break the records into groups based on low, medium and high transactions. They also ran training for varying classifiers for each group using the obtained patterns, extracted the dataset's attributes, and conducted SMOTE oversampling. They also considered the Matthew correlation coefficient (MCC) and one-class classifiers in dealing with the imbalance. After using different techniques, the classifier with the highest rating score was chosen. Their results showed that logistic regression, decision trees, and random forests performed better. One of the limitations of their research is that the only metrics used were precision, accuracy, and MCC, and the results were not compared with existing results for evaluation of improvement.

Zhu *et al.* [22] implemented the Weighted Extreme Learning Machine (WELM) to address imbalanced datasets in credit card fraud detection using optimization techniques. They found that WELM, combined with a dandelion algorithm, outperformed conventional methods like genetic algorithms. The dandelion algorithm is inspired by dandelion seed dispersal behavior [23]. They used ELM to improve the training speed and generalization of neural networks, optimizing WELM parameters with linear, binomial, and exponential probability models. They compared ten algorithms across fourteen datasets by computing G-mean, AUC, and accuracy values using MATLAB R2018a. Despite extensive graphical representations, computations were limited to MATLAB functions. Differential evolution (DE) can enhance ELM performance [24], which is applicable in medical diagnosis and face recognition [25]. Kernel ELM (KELM) has been proposed for various practical problems [26–31]. Ensemble techniques like EasyEnsemble, UnderBagging, and SMOTEboost handle class imbalance effectively[32]. SMOTEboost combines SMOTE and boosting; EasyEnsemble uses undersampling with boosting, and UnderBagging employs bagging with undersampled datasets using multiple learners.

Izotova and Valiullin [33] used Poisson processes with HomoModel, LinearModel, and QuadraticModel to compute fraud prediction probabilities, combining ensemble techniques like LGBM, XGBoost, and CatBoost to improve performance. Their dataset had 95,662 transactions from 3,633 clients, with only 0.2% being fraudulent. However, all algorithms were gradient boosting variants with similar performance. Arora *et al.* [34] found that KNN, Ensemble Methods, and deep learning techniques yield optimal fraud detection results, achieving 0.93 accuracy with deep learning and three feature selection techniques. SVM had the highest accuracy, while Naïve Bayes had the least, but they only reported accuracy without other evaluation metrics.

Burnaev *et al.* [35] studied the impact of resampling techniques on binary classification accuracy for imbalanced datasets, using decision trees, KNN, logistic regression, and various datasets. They explored oversampling, undersampling, and SMOTE, concluding that performance depends on the clas-

sifier. Undersampling can reduce performance by eliminating too many majority class samples, while oversampling can cause overfitting. SMOTE introduces synthetic records but doesn't consider neighboring example labels, unlike ADASYN and Borderline-SMOTE..

The precision-recall curve and the receiver operating characteristic (ROC) curve are commonly utilised to evaluate the performance of various models by analysing the confusion matrices they generate. These evaluation metrics provide insights into the effectiveness of the models in distinguishing between classes and identifying true positive rates. The ROC curve plots the recall against the false positive rate, making it robust against class imbalances as its shape remains unaffected by skewed dataset distributions. Conversely, the precision-recall curve, depicting precision against recall, is more sensitive to class imbalances. This sensitivity can considerably reduce precision, affecting the area under the precision-recall curve (AUPRC) [36]. A high imbalance could widen the gap in the distribution between the training set and the test set, disproving the hypothesis that the model learned from the training set can be suitably applied to the test set without a false assumption. Such distribution discrepancies could further reduce the performance of the algorithms [37, 38]. Dataset shift and concept drift arise from the changing tactics of fraudsters and dynamic cardholder behaviours, which render some algorithms obsolete and inefficient in fraud detection [39]. One way to tackle dataset shifts is to frequently update the models using new data in adaptation towards the change. Gomes *et al.* [40] and Barddal and Enembreck [41] introduced novel techniques based on the random forest algorithm using a type of Hoeffding Adaptive Tree that detects drift to check the tree error patterns and progressively identify affected features and remodel the system. Feature engineering techniques are also used to make the features more significant for the algorithms or to generate more features to improve the dataset. They can pre-process categorical features into numerical equivalents to make them more suitable for ML algorithms.

It was identified that most of the previous research works in this area focused on a single dataset, especially the 2013 European credit card fraud dataset published by Kaggle [50]. This may introduce some bias in adapting these models to novel attack behaviours of fraudsters in other datasets. Hence, a model tested on multiple datasets could be indispensable in drawing valid conclusions since these datasets show an imbalance in the binary classes. Secondly, most authors do not reveal many evaluation metrics that could help better judge their model's performance. This is sensitive because 99% accuracy with an imbalanced dataset with only 0.172% fraudulent transactions may not be enough evidence to prove that a model performs well. This is because even if the model classifies all fraudulent cases as legitimate, it will still evaluate to 99.82% accuracy. Lastly, some authors obtained scores that needed further improvement, making it suitable to build ensemble models with a high chance of mitigating the weaknesses of individual algorithms and creating an innovative approach for classification. The summary of related works is given in Table 1.

3. Materials and method

3.1. Methods

The research individually assessed the performance of five base classification algorithms and then obtained the stacking ensemble model combining the five base classifiers using Logistic Regression as the meta-classifier. This was developed on a computer with 16GB RAM, an Intel Core-17 processor, a Windows 11 Operating System, and Python 3.10.0 in a Jupyter Notebook environment. Most of the classifiers were imported from the Sklearn library. In addition, other requisite libraries used include NumPy, pandas, matplotlib, Itertools, Seaborn, and operator.

3.1.1. Machine learning algorithms ensembled

a) K-Nearest Neighbour (KNN): This is a non-parametric supervised classification algorithm that computes the distance between the test object and each of the training samples to discover the K nearest ones and then classifies the new object to belong to the class of the majority neighbours. The distance measures can be determined using Euclidean, Minkowski or Manhattan distance methods. Minkowski's distance approach is more suitable for categorical classification, while the other two are better suited for continuous variables [45]. It is applicable in both classification and regression.

By Euclidean Distance, the distance between two points P1(x1,y1)and P2(x2,y2) can be expressed as:

$$d(P_1, P_2) = \sqrt{(x_2 - x_1)^2 + (y_2 - y_1)^2} \tag{1}$$

The functions give a concise representation of the distance:

$$Euclidean \quad \sqrt{\sum_{i=1}^{n} (x_i - y_i)^2} \tag{2}$$

$$Manhattan \quad \sum_{i=1}^{n} |x_i - y_i| \tag{3}$$

$$Minkowski \quad \left(\sum_{i=1}^{n} (|x_i - y_i|)^q\right)^{1/q} \tag{4}$$

The algorithm is given below:

b) Decision Tree: This tree-based supervised algorithm attempts to partition data, assigning records to nodes using some optimality criteria to create the most miniature possible tree. This works in a divide-and-conquer fashion, dividing the data into subsets until all nodes containing data belonging to a single class or termination criteria are achieved. The nodes are used to verify properties, while the edges branch is based on the values of

Figure 1. Pipeline of the stacking approach.

the chosen attribute, and the leaves are used to label the classes. The procedure encompasses tree-building and knowledge-inferencing. Information gain is calculated using criteria such as the Gini index and entropy, which decision tree algorithms utilise to determine the best split for a node

$$Gini = 1 - \sum_{i=1}^{n} p^2(c_i), \tag{6}$$

$$Entropy = \sum_{i=1}^{n} -pc_i log_2(p(c_i)), \tag{7}$$

where p(ci) is the probability/percentage of class ci in a node.

c) Naïve Bayes: This supervised classification algorithm operates based on the Bayes theorem and calculates the posterior probabilities of records belonging to each class and presumes that the class features are conditionally independent.

$$P(c \mid x) = \frac{P(x \mid c)\,P(c)}{P(x)} \tag{8}$$

where P(c|x) = Posterior Probability, P(x|c) = Likelihood, P(c) = Class Prior Probability, P(x) = Predictor Prior Probability

$$P(c\,jX) = P(x1\,jc) * P(x2\,jc) * \ldots * P(xn\,jc) * P(c) \tag{9}$$

Table 1. Summary of related works in ensemble ML techniques.

Article	Method and algorithm used	Details and Strength	Metrics used	Remarks and Weaknesses
[42]	Ensemble and mixed learning using clustering, KNN, NB, Logistic Regression, RF, Gradient-Boosted Trees and MLP.	Using mixed learning, they applied clustering for pre-processing before applying supervised classifiers and ensembles, obtaining results for two datasets. From their results, NN models performed better than most other models, while KNN achieved the best individual result among the supervised classifiers.	Accuracy, sensitivity, specificity and balanced classification rate.	The ensembles achieved better performance than the individual classifiers. However, they did not reveal some evaluation metrics, such as F1-score and ROC-AUC, which would have provided more evidence about their model's performance on imbalanced datasets.
[43]	Logistic Regression, SVM, NB, RF, DT and KNN	They applied resampling techniques, including under-sampling, SMOTE, and ADASYN. They conducted a comparative analysis and found that RF with oversampling achieved the best performance.	Precision accuracy, AUC, recall, and F1-score	The resampled dataset's classification was better than that of the imbalanced dataset. However, they used only one dataset and did not give extensive results on their model's performance when tested with other datasets.
[44]	NB, DT, KNN, SVM, and MLP, as well as ensemble techniques such as EasyEnsemble, are also used.	Seven resampling techniques were applied, including SMOTE, SMOTEENN, BorderlineSMOTE, ADASYN, and others. The meta-classifiers with resampling techniques achieved an improved performance after combining them with the base models.	Accuracy, F1-score, AUC	Their work did not reveal precision and recall, which are sensitive to imbalanced datasets. All their F1 scores were below 82%, and all the AUCs were not above 97%, which can still be improved.
[45]	Logistic Regression, KNN, RF, NB, MLP, AdaBoost, pipelining and ensemble techniques.	They applied the ADASYN resampling techniques to correct the imbalance in the dataset. The pipelining approach achieved the highest accuracy, followed by the ensemble method, while RF was the best-performing individual algorithm.	Accuracy, precision, recall, F1 score, Matthews Correlation Coefficient (MCC) and Balanced Classification rate	The precision, recall and F1-score they obtained for the fraudulent class was relatively low and needed improvement.
[46]	NB, DT, Logistic Regression, KNN, SVM, RF and ensemble algorithms such as AdaBoost and Bagging.	They performed under-sampling to generate an equal amount of fraudulent and everyday transactions. Logistic regression, RF, and ensemble techniques such as AdaBoost and Gradient Boosted Trees performed the best.	Precision, Recall, F1-score, accuracy and AUC.	The limitation of the research is that most of the methodology and evaluation was narrowed towards the features in a particular European dataset [50], such as Time and Amount
[47]	Deep reinforcement learning (DRL), AdaBoost, XGBoost, DNN, KNN, RF and Logistic Regression.	They used SMOTE and ADASYN oversampling techniques to achieve a balanced dataset. Of all the tests they conducted on the imbalanced dataset, RF and XGBoost had the best accuracies.	Accuracy, precision, recall, specificity, f1-score, MCC and ROC_AUC.	DNN and RF performed the best after oversampling, while LR performed the least. Their DRL achieved a very low accuracy of 34.8%, precision of 0.067% and F1-score of 13.7%, which still needed much improvement.
[33]	Three ensemble techniques: Light Gradient Boosting, XGBoost and CatBoost	Three models of the Poisson process model were used: HomoModel, LinearModel, and QuadraticModel. They obtained the best performance with XGBoost, followed by CatBoost, then LGBM.	ROC_AUC	The work omitted comprehensive results from multiple evaluation metrics and showed only the value for ROC-AUC.

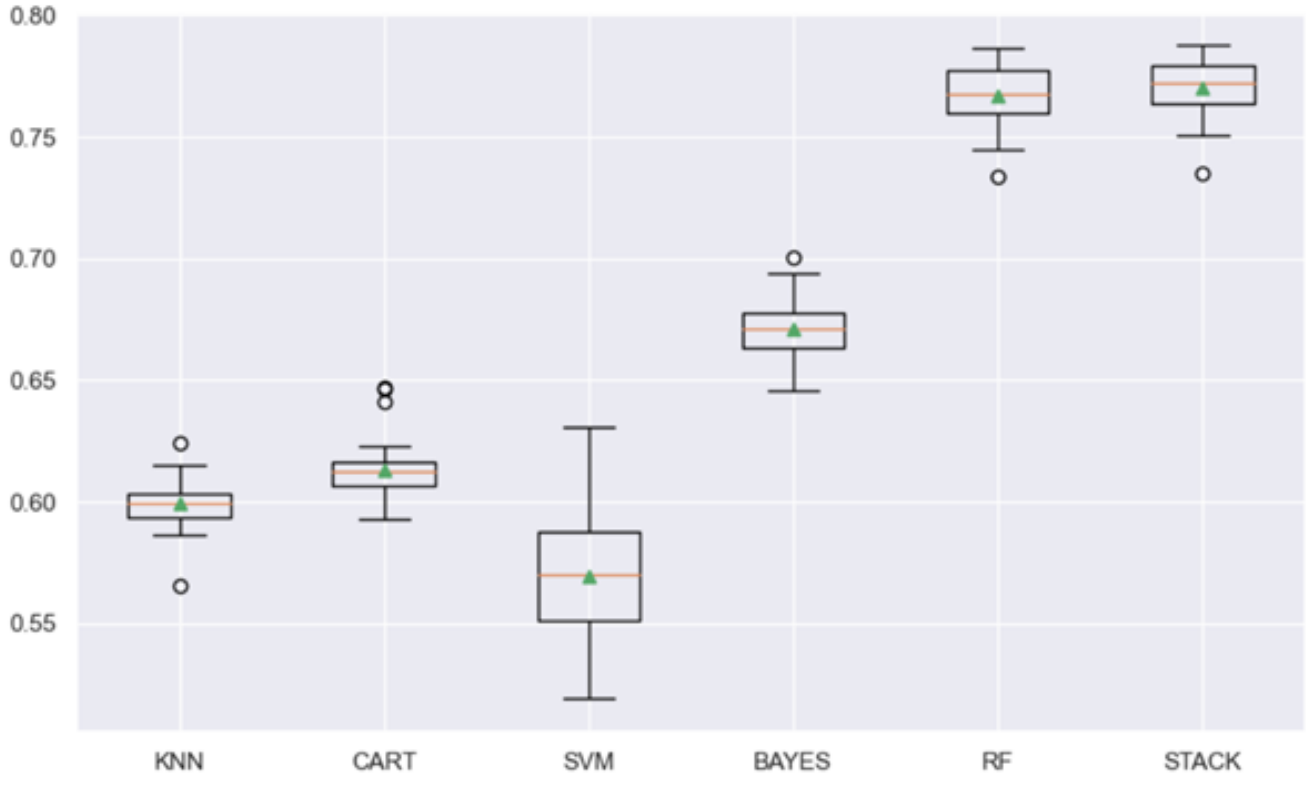

Figure 2. Box plots of the ROC_AUC of the models using the imbalanced UCI_Dataset.

Algorithm 1

Input: A is the set of training instances, the test instance, n is a vector of the attributes, and B is the set of classes used as labels for the cases.

Output: gnϵ H, the class of n

For each instance, i ϵ A do

| Compute k(n, i), the distance between n and i;

End

Select P$\subseteq$ A, the set (neighbourhood) of m closest training instances for n;

$$c_z = argmax \sum_{y? N} I(v = class(c_y)), \quad (5)$$

where (.) is an indicator function that returns the value one if its argument is valid and 0 otherwise.

d) Random Forest: This algorithm builds an ensemble of separate DTs using a subset of the training records selected by sampling with replacement from the entire training set using Bootstrap. It decides the final classification by accepting the vote from the majority trees. Increasing the number of trees in the forest helps it to achieve improved outcomes and mitigate overfitting. It applies to both classification and regression. Random forests (RF) generate numerous individual decision trees during training. These trees collectively contribute to the final prediction by pooling their outputs, which typically involves selecting the mode of classes for classification tasks or averaging predictions for regression tasks. Due to their utilisation of multiple results to arrive at a final decision, they are classified as ensemble techniques

For each decision tree, Scikit-learn calculates the importance of a node using Gini Importance, assuming only two child nodes (binary tree):

$$ni_j = w_j C_j - - w_{left(j)} C_{left(j)} - - w_{right(j)} C_{right(j)}, \quad (10)$$

where n_i sub(j)= the importance of node j, w sub(j) = weighted number of samples reaching node j, C sub(j)= the impurity value of node j, left(j) = child node from left split on node j, right(j) = child node from right split on node j.

e) Support Vector Machines (SVM):This presents a supervised machine learning challenge, where the objective is to identify a hyperplane that effectively separates two classes. While Logistic Regression (LR) and Support Vector Machines (SVM) aim to locate the optimal hyperplane, they differ fundamentally in their approaches. LR operates on a probabilistic basis, whereas SVM relies on statistical methods. SVM employs margin optimisation and kernel representation to function within a high-dimensional feature space. Its objective is to determine a hyperplane that segregates the binary classes. To address the potential for an infinite number of hyperplanes accurately classifying the two classes, SVM resolves this by identifying the hyperplane with the maximum margin – representing the most significant distance between the two classes. This approach ensures robust classification by maximizing the margin between data points of different classes, leading to better generalization and improved performance in classification tasks. The simplified equation or soft margin formulation for SVM encapsulates this principle, providing a concise representation of the algorithm's objective

$$min\frac{1}{2}w^T w + C\sum_{i=1}^{m}\mathrm{E}_i \tag{11}$$

$$s.t. \qquad yi\left(x_i^T w + b\right) \geq 1 - \epsilon_i\epsilon_i \geq 0. \tag{12}$$

f) Logistic Regression: This was used as the meta-classifier in the stacking model. It calculates the probability of a record belonging to a particular class using explanatory variables. It is a multivariate statistical model that performs non-linear logistic transformation to determine the output variable by making interpretations as probability ratios using the maximum likelihood approach. It uses a set of feature values as an argument of the sigmoid function S(x). This mathematical expression resolves to a characteristic "S" curve whose output is a number that can be categorizedas 0 or 1. The mid-value 0.5 is often used as the threshold for rounding up numbers to the binary classes.

$$S\,(x) = \frac{1}{1+e^{-x}} = \frac{e^x}{1+e^x} = 1 - S\,(-x). \tag{13}$$

In the ensemble model developed in this research, the level-0 base classifiers were used for individual classifications, and the level-1 meta classifier (Logistic Regression) was used to combine the level-0 classifiers into the stacking classifier to attempt an improved performance that was better than most individual results.

3.2. The ensemble approach adopted

Our ensemble learning approach uses the stacking approach. This technique uses predictions from multiple models and finally passes the projections through a Meta classifier. Here, we ensemble predictions from Support Vector Machine (SVM), Random Trees (RT), Random Forests (RF), Naïve Bayes (NB) and K Nearest Neighbour (KNN).

Unlike bagging, where multiple instances of the same model are trained on different subsets of the data, stacking involves using diverse models for the ensemble - the five described above - and fit on the same dataset – we describe the three datasets used in the section below. Also, unlike boosting, in stacking, a single model is used to learn how to best combine the predictions from the contributing models, not to correct the predictions from the previous models.

The pipeline of our stacking model, depicted in Figure 1, comprises five base models, denoted as level 0 models, and a meta-model that amalgamates the predictions generated by these base models, termed level 1 models. The level 0 models are initially trained on the training data, and their predictions are computed. Subsequently, the level 1 model, the logistic regression model, learns the optimal approach to amalgamate the predictions from the five base models. The meta-model is subsequently trained using the projections generated by the five base models on out-of-sample data, which refers to data that was not used during the training of the base models. These predictions and the expected outputs form the input-output pairs utilizedfor training the meta-models. This process enables the meta-model to learn how to effectively combine the predictions from the base models to produce accurate and reliable results. We aim to combine these five different models: SVM, KNN, Decision Tree, Random Forests, and Naive Bayes by stacking them, using logistic regression as a meta-classifier. The primary purpose of the combination is to improve performance. We also test the individual models using performance metrics, including Accuracy, Precision, Recall, F1, and ROC-AUC, using the three datasets of varying sizes (small, medium and large) and test the derived new model. The algorithm for our approach is given in Algorithm 1.

3.3. Datasets used

Three different datasets were used for the experiment. The first dataset comprises 30,000 records of customer payments in Taiwan in 2005, including 6636 fraudulent and 23 364 genuine transactions obtained from the UCI Machine Learning Repository [48]. It contains 23 features with integer and actual values and a binary class label, as shown in Table 2.

The second dataset was obtained from the gksj7 GitHub repository [49]. It contains 3075 records comprising 448 fraudulent and 2627 genuine transactions, with 10 features and a binary class label. The features include MerchantId, Average amount/transaction/day, transaction_amount, is_declined, total number of transactions/day, isForeignTransaction, isHighRisk, daily_chargeBack_avg_amt, 6mth_chargeBack_avg_amt, and 6mth_chargeback_frq.

In the third dataset obtained from Kaggle[50], there was a high level of imbalance, comprising only 492 (0.172%) fraudulent transactions and 284315 (99.828%) genuine transactions. This dataset consists of actual transactions from credit cardholders in Europe in September 2013 within two days. This dataset comprises 30 features and the class label (1 for fraudulent and 0 for genuine). The numeric features V1 to V28 were

Table 2. Breakdown of the features in the first dataset.

Feature name	Content
X1	Amount (NT Dollars)
X2	Gender (male and female)
X3	Education (graduate school, university, high school, and others)
X4	Marital Status (single, married, others)
X5	Age (year)
X6 – X11	History of past payments (April to September 2005)
X12 – X17	Amount of Bill Statement (April to September 2005)
X18 – X23	Amount of previous Statement (April to September 2005)

Algorithm 2 Stacking Algorithm using k-fold validation

Input: $T = \{x_i, y_i, x_i \in R^n, y \in \{0,1\}\}$
$Dataset = \{Ds_1, Ds_2, Ds_3\}$ $Dataset \in Ds_i$, $i = 1..3$
$Base\ Classifiers\ m_k = \{KNN, RF, NB, SVM, DT\}$
Meta Classifier M
Output: *Ensemble Classifier E*
Using cross-validation, segment the dataset into k equal-sized subsets
Step 1: *Learn base-level classifiers – Level 0*
For k = 1 to T do
Learn m_k *based on* Ds_i
End For
Step 2: *Construct new dataset - Level 1*
For i = 1 to n do
$Ds_m = \{x_i, y_i\}_{where}\ x_i = \{m_1(x_i) \ldots m_T(x_i)\}$
Train M with k-fold cross-validation
End For
Step 3: *Learn a Meta Classifier*
Learn M based on Ds_m
Build E $\forall m_k \rightarrow M$
Evaluate M, E for accuracy
Return E

anonymised to preserve privacy and confidentiality using Principal Component Analysis(PCA), except for Time and Amount. Hence, to avert poor performance and long execution time from the ML models, the dataset was balanced to include an equal number of fraudulent and genuine transactions. This was accomplished by applying the Synthetic Minority Over-sampling Technique (SMOTE) to augment the samples in the minority class, followed by using RandomUnderSampler to remove instances of the majority class, ensuring an equal amount of data as the minority class. After the resampling, the dataset contained 11372 fraudulent and 11372 genuine transactions, totalling 22744 transactions. SMOTE synthesizes new transactions from the minority class by skewing the minority class samples and making the decision boundary between classes less specific. It has three major operations: randomizing instances of the minority class, computing the k nearest neighbour of minority class instances and generating synthetic instances for the minority group. This can be done by linearly interpolating minority instances chosen randomly and their neighbours. SMOTE is preferable because random oversampling by duplicating minority class samples does not improve classification performance [44]. Table 3 gives a summary of the datasets used in this work.

3.4. Performance metrics

Accuracy is not usually an efficient measure for unbalanced datasets since the minority class is limited, hence, it does not determine a good classification performance. The AUC, which measures the whole threshold range of accuracy, is a more efficient metric for imbalanced classifiers. However, accuracy remains useful when rebalancing is achieved through some resampling techniques. Precision and recall, used to compute the F1-score, are more valuable in skewed distributions of unbalanced datasets. Equations (14)–(17) describe the performance metric used.

i Accuracy: This is the proportion of correct predictions in the entire prediction.

$$ACC = \frac{TP + TN}{TP + TN + FP + FN} \tag{14}$$

ii Precision: This is the proportion of optimistic predictions in the whole set of positive classes predicted.

$$Precision = \frac{TP}{TP + FP} \tag{15}$$

iii Recall: This is the proportion of positive predictions in the entire positive class in the test data.

$$Recall = \frac{TP}{TP + FN} \tag{16}$$

iv F1-score: This represents the harmonic mean of the recall and precision. High values can be interpreted as high classification performance.

$$F1 = \frac{2 * Precision * Recall}{Precision + Recall} \tag{17}$$

v The AUC (Area under the curve) of ROC (Receiver operating characteristic) is obtained by plotting the ratio of true positives against the false positives rates.

Table 3. Summary of datasets used.

Descriptions	UCI_Dataset	Github_Dataset	Kaggle_Dataset
Number of Features	23	10	30
Labels	Class (0 or 1)	Is_Fraudulent (0 or 1)	Class (0 or 1)
Number of Rows	30, 000	3, 075	284, 807
Percentage of Frauds vs. Non-fraud before resampling	22.12% vs. 77.88%	14.569% vs. 85.43%	0.172% vs. 99.828%
No of Frauds vs. Non-fraud before resampling	6,636 vs. 2,6364	448 vs. 2627	492 vs. 284,315
No of Frauds vs. Non-fraud after resampling	15,186 vs. 15,186	2,626 vs. 2,627	11,372 vs. 11,372
Resampling technique used	SMOTE and RandomUnderSampler	SMOTE only	SMOTE and RandomUnderSampler

4. Result and discussion

This section reports the performance of the base classifiers and the stacking algorithm using the three datasets, as depicted in Tables 4–8 and Figures 2– 5. In the experiment, UCI_Dataset and Github_Dataset were first trained and tested without resampling since their size is not too large for the stacking ensemble. Hence, the initial proportion of fraudulent and legitimate transactions in the imbalanced datasets was maintained. This had a rigorous effect on the SVM model, which had minimal precision, recall and f1-score, as shown in Table 4.

Using the medium-sized dataset with 30,000 rows (UCI_Dataset), Gaussian NB had the least accuracy but recorded the highest recall value. On this dataset, RF achieved the same accuracy as the Stacking model and attained a higher recall f1-score and recall than the Stacking model. However, the Stacking model had the highest precision and ROC_AUC as depicted in Table 5 and the box plots of Figure 2. The algorithms' performance was the lowest in unbalanced dataset 1, which had the highest number of legitimate transactions (23 364 out of 30 000 rows). The skewed distribution of this dataset affected all the models, as illustrated in Figure 3.

Using the small-sized imbalanced dataset with 3075 rows (Github_Dataset), the accuracies of all the models improved as tabulated in Tables 7.

After implementing the models on the imbalanced datasets, resampling techniques, including SMOTE and RandomUnderSampler, were applied to train the model again. The results obtained, as shown in Table 7, reveal a significant increase in performance. The decision tree-based models, such as DT and RF, had the best individual performance,which was very close to the stacking model.

The performance of the decision tree model, which is based on the Classification and Regression Trees (CART) algorithm, improved drastically in all the metrics, competing with RF and the stacked model. Even though the accuracy of the Gaussian NB increased, the precision, recall, and f1-score dropped drastically, almost approaching the minimal values similar to those of the SVM. The stacking model achieved the best performance concerning the accuracy, recall, precision, and f1-score even though RF had the highest ROC_AUC value for this dataset. The decision regions of the models concerning the imbalanced

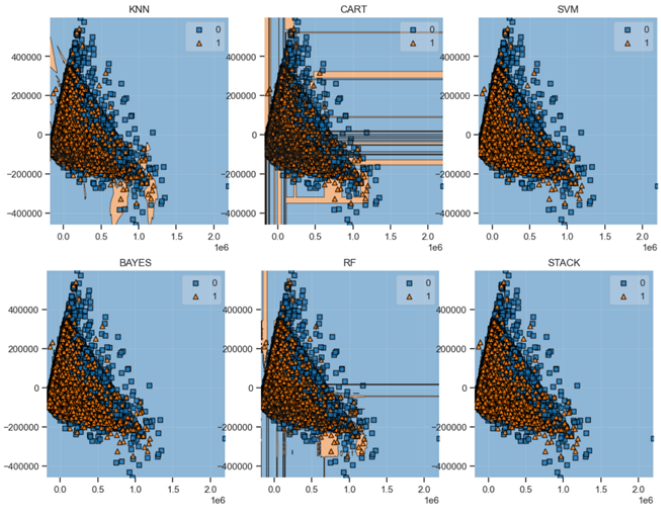

Figure 3. Plot of the decision regions using the precision of the models in the imbalanced UCI_Dataset.

Github_Dataset are represented in Figure 4, while the box plots of the models for the same dataset are given in Figure 5.

The third dataset from Kaggle solely applied the SMOTE and RandomUnderSampler to create a medium-sized dataset (22744 rows) with an equal number of fraudulent and legitimate transactions. With this, the SVM's precision, recall and f1-score increased immensely even though the accuracy dropped. This is shown in Table 8. The performance of KNN also improved with this dataset, producing precision, recall, f1-score and ROC_AUC that were competitive with the best-performing models. Overall, the performances of all the models were also very good, with this dataset recording ROC_AUC above 98% in at least four models and f1-score above 98% in at least three models, as shown in Table 8. Decision Tree and RF attained very competitive performance relative to the stacking model, attaining above 98% in all their metrics. RF, as an individual model, recorded the best performance over the stacking model in precision and attained the same perfect ROC_AUC of 1.0 with the stacked model. RF could achieve such performance because it is an ensemble of several decision trees. In general, using this balanced dataset, the stacking model performed bet-

Table 4. Results from UCI_Dataset before resampling.

Algorithm	Accuracy	Precision	Recall	F1-score	ROC_AUC
KNN	0.750	0.366	0.178	0.239	0.599
Decision Tree (DT)	0.725	0.384	0.410	0.402	0.613
SVM	0.779	0.000	0.000	0.000	0.659
Gaussian NB	0.379	0.247	0.885	0.387	0.671
RF	0.818	0.658	0.369	0.473	0.767
Stacking	0.818	0.666	0.353	0.462	0.770

Table 5. Results from UCI_Dataset after resampling.

Algorithm	Accuracy	Precision	Recall	F1-score	ROC_AUC
KNN	0.69600	0.65862	0.81397	0.72807	0.76565
Decision Tree (DT)	0.71300	0.70853	0.72817	0.71772	0.71424
SVM	0.62000	0.60320	0.70095	0.64837	0.68083
Gaussian NB	0.54500	0.52558	0.93196	0.67211	0.67604
RF	0.81100	0.83001	0.78177	0.80515	0.88772
Stacking	0.81900	0.81552	0.82847	0.81965	0.89699

Table 6. Results from Github_Dataset before resampling.

Algorithm	Accuracy	Precision	Recall	F1-score	ROC_AUC
KNN	0.857	0.541	0.151	0.233	0.577
Decision Tree (DT)	0.980	0.939	0.924	0.928	0.956
SVM	0.854	0.000	0.000	0.000	0.472
Gaussian NB	0.855	0.200	0.004	0.009	0.777
RF	0.983	0.963	0.922	0.941	0.996
Stacking	0.984	0.964	0.931	0.945	0.988

Table 7. Results from Github_Dataset after resampling.

Algorithm	Accuracy	Precision	Recall	F1-score	ROC_AUC
KNN	0.67047	0.66424	0.68937	0.67638	0.73251
Decision Tree (DT)	1.00000	1.00000	1.00000	1.00000	1.00000
SVM	0.53620	0.57414	0.27774	0.37351	0.53169
Gaussian NB	0.70004	0.77957	0.55800	0.64994	0.79843
RF	1.00000	1.00000	1.00000	1.00000	1.00000
Stacking	1.00000	1.00000	1.00000	1.00000	1.00000

ter than all the individual models, reaching equal accuracy and an f1-score of 99.6%. Further discussions on how such models are used for predicting instances of fraud with all the transaction features can be found in [51]. This practice involves using the model's fit() and predict() Python functions to classify the transaction into the binary classes of legitimate or fraudulent. The decision regions and box plots of the models using the Kaggle_Dataset are shown in Figures 5 and 6, respectively.

It is noteworthy to mention that the decimal places were increased in the metrics obtained from the resampled small and medium-sized dataset to improve the level of precision of the results obtained. For research, the Python codes were developed, and the results were made available on a public repo

Table 8. Results from Kaggle_Dataset.

Algorithm	Accuracy	Precision	Recall	F1-score	ROC_AUC
KNN	0.841	0.825	0.865	0.845	0.915
Decision Tree (DT)	0.984	0.981	0.987	0.984	0.984
SVM	0.551	0.543	0.637	0.586	0.574
Gaussian NB	0.866	0.991	0.738	0.846	0.982
RF	0.995	0.998	0.993	0.995	1.000
Stacking	0.996	0.997	0.995	0.996	1.000

Table 9. Comparison of results with other authors.

Model	Highest Accuracy	Highest Precision	Highest Recall	Highest F1-score	ROC_AUC
Proposed approach on Kaggle_Dataset	0.996	0.998	0.995	0.996	1.000
Proposed approach on Github_Dataset	0.9999	1.000	1.000	0.9999	1.000
Proposed approach on UCI_Dataset	0.81900	0.83001	0.93196	0.81965	0.89699
[47]	0.999	0.999	1.000	0.999	1.000
[48]	0.939	0.950	0.940	0.940	0.940
[49]	0.999	1.000	1.000	1.000	-
[50]	0.999	0.999	1.000	0.999	0.999
[51]	0.999	-	-	0.8184	0.970
[52]	0.998	-	0.999	-	-

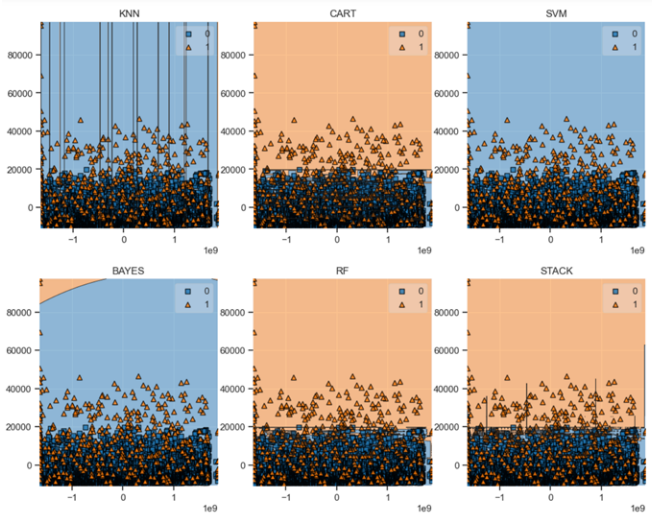

Figure 4. A plot of the decision regions using the accuracies of the models in the imbalanced Github_Dataset.

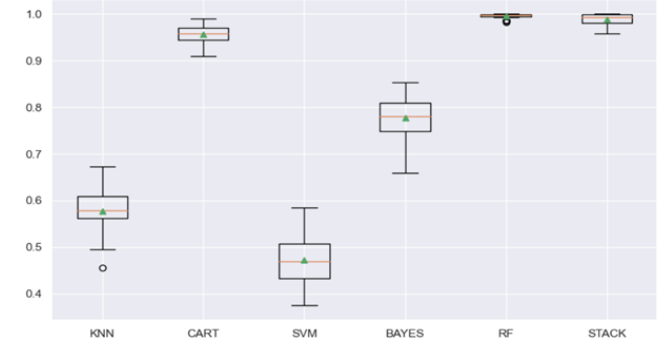

Figure 5. Box plot of F1-scores from the imbalanced Github_Dataset.

sitory in GitHub [52]. We acknowledge that the complexity of our approach is a significant consideration. Training multiple models and performing K-fold validation increases computational demands. However, we mitigated this by parallelising the training processes and using efficient data handling practices. In our study, we assessed the computational costs and deemed them acceptable given the predictive performance gains. Detailed complexity analysis and runtime performance data are beyond the scope of this work.

In our experiments, the Random Forest (RF) algorithm performed exceptionally well. However, the stacked logistic regression model still holds value. The stacking approach allows us to integrate and weight the predictions from all base models, potentially capturing complementary strengths that a single algorithm might miss. The marginal performance improvement in some cases may seem small, but it can be critical in high-stakes applications where even slight accuracy gains are valuable. Although the high performance metrics may suggest overfitting, we took several steps to validate our findings, including extensive cross-validation and testing on separate datasets as seen in Section 4. We also examined learning curves to ensure that performance gains were not merely artifacts of overfitting.

4.1. *Comparison of results*

The comparison of results with state-of-the-art models [42–47] is given in Table9.

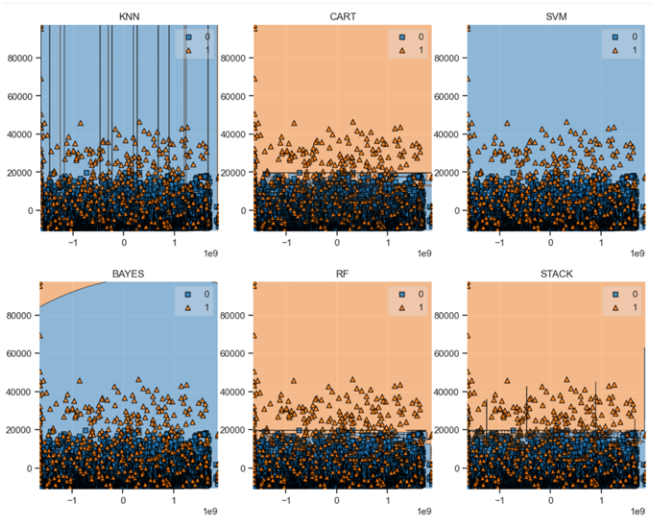

Figure 6. A plot of the decision regions using the F1-scores of the models in the balanced Kaggle_Dataset.

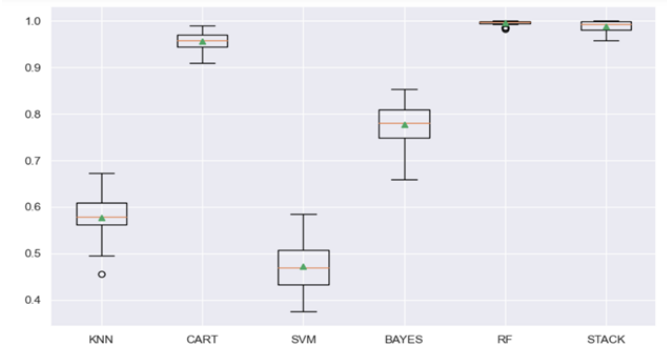

Figure 7. Box plot of ROC_AUC from the balanced Kaggle_Dataset.

This proposed stacking ensemble was benchmarked against the works of [42–47]. From the results obtained in this work, it is observed that the proposed model is highly competitive with extant models that used the Kaggle dataset [51], producing ROC_AUC of 100% and scoring above 99% in all other metrics. The work of Dang *et al.* [47] also attained an equivalent ROC_AUC with high scores in different metrics. They achieved this performance using SMOTE and ADASYN with several ML and deep learning approaches, arriving at their best performance with the RF model. The work inAhmed and Shamsuddin [43] also attained a notable performance after they used several ML algorithms with SMOTE and ADASYN. After applying some oversampling techniques to the dataset, they reached their best performance with the RF algorithm. InKerwin and Bastian [44], a high accuracy score was obtained after using the Multilayer perceptron, GradientBoostingMachine and AdaBoost using the SMOTE and SMOTEENN resampling techniques. Bagga *et al.* [45] achieved high precision, recall and f1-score by implementing pipelining and bagging ensemble techniques with seven ML algorithms using RF as the base classifier for the pipeline. Hence, it can be observed that the RF algorithm and ensemble techniques are very good at credit card fraud classification problems.

5. Conclusion and future work

Ensemble ML models can learn from massive datasets and discover trends that individual ML algorithms may not observe when used separately. In this research, we evaluated the performance of a stacking ensemble of five classifiers, including K-NN, SVM, RF, GuassianNB and DT, using Logistic Regression as the meta-classifier. The objective was to develop a model that can detect credit card fraud given a dataset with several features. Employing K-Fold cross-validation with 10 splits and three repeats, the models were aggregated through stacking, revealing a consistent enhancement over individual models. This enhancement arises from the complementary strengths of different algorithms compensating for each other's weaknesses. In this problem domain in the banking sector, the RF proves to be better than most supervised ML classifiers since it operates as an ensemble of decision trees. Using SMOTE and RandomUnderSampler to reduce the effect of the imbalance on large datasets, the algorithms displayed a drastic increase in performance, reaching a peak accuracy and ROC-AUC that is competitive with other novel research. This shows that a balanced dataset without skewed distribution provides better training data for the algorithms to develop a better model.

Knowing that today, credit card transactions are among the fastest means of payment, such that VISA cards can perform about 2,000 transactions per second while the blockchain system can only perform about seven transactions per second with high energy bills and gas fees. It is expedient to validate accurate and precise models to help financial institutions keep watch over millions of transactions they process with credit cards. The COVID-19 pandemic has also encouraged worldwide use of credit card transactions, making it a new area for fraudsters using social engineering methods to conduct reconnaissance and defraud cardholders. The models developed in this study have been structured to allow the use of posterior methods established using historical datasets to predict future parameters. Through the use of fit and predict functions, these models enable the classification of new data based on previous predictive parameters, facilitating informed decision-making in practical contextsIn the future, we hope to explore novel techniques in deep neural networks (DNN) to develop models that train with sufficient neurons and layers. It is noteworthy that only a few public datasets are available in this domain due to data protection, privacy and confidentiality regulations. Hence, we hope to seek more current datasets in our subsequent studies to update our models. This is because there is concept drift and dataset shift in the behaviour of attackers, leading to a need to constantly update the models to meet the dynamic behaviour of cardholders and fraudsters. With such upgrades, more relevant models will be developed to make them easy to integrate into the applications in the banking sector.

We hope also to consider adversarial ML techniques that sometimes lead to data poisoning, which could cause models to yield false predictions. Lastly, our studies will further extend

to the application of such novel ensemble and resampling techniques to other critical areas, such as the security and privacy of healthcare data because attacks on some Internet of Medical Things (IoMT) devices such as heart pacemakers, deep brain implants, insulin pumps and defibrillators could lead to loss of life.

References

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

[2] F. Itoo & S. Singh "Comparison and analysis of logistic regression, Naïve Bayes and KNN machine learning algorithms for credit card fraud detection", International Journal of Information Technology **13** (2021) 1503. [Online] https://link.springer.com/article/10.1007/s41870-020-00430-y.

[3] D. Huang, Y. Lin, Z. Weng & J. Xiong, "Decision Analysis and Prediction Based on Credit Card Fraud Data", The 2nd European Symposium on Computer and Communications, New York, NY, USA, 20–26. https://doi.org/10.1145/3478301.3478305.

[4] L. Moumeni, M. Saber, I. Slimani, I. Elfarissi & Z. Bougroun, "Machine learning for credit card fraud detection", Lecture Notes in Electrical Engineering WITS 2020, Springer Singapore, 2021, pp. 211–221. http://dx.doi.org/10.1007/978-981-33-6893-4_20.

[5] N. Yousefi, M. Alaghband & I. Garibay, "A Comprehensive Survey on Machine Learning Techniques and User Authentication Approaches for Credit Card Fraud Detection", International Journal of Computer and Information Engineering **15** (2021) 599. https://publications.waset.org/pdf/10012319.

[6] G. Rushin, C. Stancil, M. Sun, S. Adams & P. Beling "Horse race analysis in credit card fraud detection using deep learning, logistic regression, and gradient boosted tree", IEEE Systems and Information Engineering Design Symposium (SIEDS), Charlottesville, Virginia, USA, 2017, pp. 117–121. https://doi.org/10.1109/SIEDS.2017.7937700.

[7] Y. Wang, S. Adams, P. Beling, S. Greenspan, S. Rajagopalan, M. Velez-Rojas, S. Mankovski, S. Boker & D. Brown, "Privacy-preserving distributed deep learning and its application in credit card fraud detection", IEEE International Conference On Trust, Security And Privacy In Computing And Communications/12th IEEE International Conference On Big Data Science And Engineering (TrustCom/BigDataSE), New York, NY, USA, 2018, 1070–8. https://ieeexplore.ieee.org/document/8456019.

[8] A. Bahnsen, A. Stojanovic, D. Aouada & B. Ottersten, "Cost-sensitive credit card fraud detection using Bayes minimum risk", International Conference on Machine Learning and Applications (ICMLA), Miami, Florida, USA, 2013, pp. 333–8. https://ieeexplore.ieee.org/document/6784638.

[9] A. Pozzolo, O. Caelen, Y. A Le Borgne, S. Waterschoot & G. Bontempi, "Learned lessons in credit card fraud detection from a practitioner perspective", Expert systems with applications **41** (2014) 4915. https://doi.org/10.1016/j.eswa.2014.02.026.

[10] A. Pozzolo, G. Boracchi, O. Caelen, C. Alippi & G. Bontempi, "Credit card fraud detection and concept-drift adaptation with delayed supervised information", International Joint Conference Neural Networks (IJCNN), Killarney, Ireland, 2015, PP. 1–8. https://ieeexplore.ieee.org/document/7280527.

[11] V. Vlasselaer, C. Bravo, O. Caelen, T. Eliassi-Rad, L. Akoglu, Snoeck & B. Baesens, "Apate: A novel approach for automated credit card transaction fraud detection using network-based extensions Decision Support Systems", Decis. Support Syst. **75** (2015) 38. http://dx.doi.org/10.1016/j.dss.2015.04.013.

[12] R. Mohammed, K. Wong, M. Shiratuddin & X. Wang, "Scalable machine learning techniques for highly imbalanced credit card fraud detection: A comparative study", Pacific Rim International Conference on Artificial Intelligence, Nanjing, China, 2018, pp. 237-246. https://doi.org/10.1007/978-3-319-97310-4_27.

[13] N. Mahmoudi & E. Duman "Detecting credit card fraud by modified fisher discriminant analysis", Expert Systems with Applications **42** (2015) 2510. https://doi.org/10.1016/j.eswa.2014.10.037.

[14] M. Mahmud, S. Meesad, "An evaluation of computational intelligence in credit card fraud detection", International Computer Science and Engineering Conference (ICSEC), Austin, Texas, USA, 2016 pp. 1–6. https://ieeexplore.ieee.org/document/7859947.

[15] K. R Seeja & M. Zareapoor, "Fraudminer: A novel credit card fraud detection model based on frequent itemset mining", The Scientific World Journal **2014** (2014) 1. https://doi.org/10.1155/2014/252797.

[16] S. Kumari & A. Choubey, "Credit card fraud detection using Hmm and k-means clustering algorithm", International Journal of Scientific Research Engineering and Technology (IJSRET) **6** (2017) 2278. [Online] https://www.semanticscholar.org/paper/Credit-Card-Fraud-Detection-Using-HMM-and-K-Means-Kumari-Bhilai/16146abaf34f53fa1380f4addb84527dd54e3fcf.

[17] T. Behera & S. Panigrahi, "redit card fraud detection: a hybrid approach using fuzzy clustering & neural network", International Conference of Advances in Computing and Communication Engineering (ICACCE), Dehradun, India, 2015, pp. 494-9. https://ieeexplore.ieee.org/document/7306735.

[18] C. Jiang, J. Song, G. Liu, L. Zheng & W. Luan, "Credit card fraud detection: A novel approach using aggregation strategy and feedback mechanism", IEEE Internet of Things Journal **5** (2018) 3637. https://doi.org/10.1109/JIOT.2018.2816007.

[19] D. Olszewski, "Fraud detection using self-organizing map visualizing the user profiles", Knowledge-Based Systems **70** (2014) 324. https://doi.org/10.1016/j.knosys.2014.07.008.

[20] V. Agaskar, M. Babariya, S. Chandran & N. Giri, "Unsupervised learning for credit card fraud detection", International Research Journal of Engineering and Technology **4** (2017) 2343. [Online] https://www.irjet.net/archives/V4/i3/IRJET-V4I3608.pdf.

[21] N. Vaishnavi & S. Geetha, "Credit Card Fraud Detection using Machine Learning Algorithms", International Conference on Recent Trends in Advanced Computing, Chennai, India, 2019, pp. 631-641. https://doi.org/10.1016/j.procs.2020.01.057.

[22] H. Zhu, G. Liu, M. Zhou, Y. Xie, A. Abusorrah & Q. Kang, "Optimizing Weighted Extreme Learning Machines for imbalanced classification and application to credit card fraud detection", Neurocomputing **407** (2020) 50. https://doi.org/10.1016/j.neucom.2020.04.078.

[23] X. Li, S. Han, L. Zhao, C. Gong & X. Liu, "New dandelion algorithm optimizes extreme learning machine for biomedical classification problems", Comput. Intell. Neurosci. **2017** (2017) 1. https://doi.org/10.1155/2017/4523754.

[24] Y. Yu, S. Gao, Y. Wang & Y. Todo, "Global optimum-based search differential evolution", IEEE/CAA J. Autom. Sin. **6** (2019) 379. http://dx.doi.org/10.1109/JAS.2019.1911378.

[25] Z. Wang, G. Yu, Y. Kang, Y. Zhao & Q. Qu, "Breast tumor detection in digital mammography based on extreme learning machine", Neurocomputing **128** (2014) 17. https://doi.org/10.1016/j.neucom.2013.05.053.

[26] C. Chen, W. Li, H. Su & K. Liu, "Spectral-spatial classification of hyperspectral image based on kernel extreme learning machine", Remote Sens **6** (2014) 5795. https://doi.org/10.3390/rs6065795.

[27] T. Liu, L. Hu, C. Ma, Z. Wang & H. Chen, "A fast approach for detection of erythemato-squamous diseases based on extreme learning machine with maximum relevance minimum redundancy feature selection", Int. J. Syst. Sci. **46** (2015) 919. http://dx.doi.org/10.1080/00207721.2013.801096.

[28] Q. Li, H. Chen, H. Huang, X. Zhao, Z. Cai, C. Tong & X. Tian, "An enhanced grey wolf optimization based feature selection wrapped kernel extreme learning machine for medical diagnosis", Comput. Math. Methods Med. **2017** (2017) 1. https://doi.org/10.1155/2017/9512741

[29] D. Zhao, C. Huang, Y. Wei, F. Yu, M. Wang & H. Chen "An effective computational model for bankruptcy prediction using kernel extreme learning machine approach", Comput. Econ. **49** (2017) 325. https://doi.org/10.1007/s10614-016-9562-7.

[30] W. Deng, Q. Zheng & Z. Wang,"Cross-person activity recognition using reduced kernel extreme learning machine", Neural Netw. **53** (2014) 1. https://doi.org/10.1016/j.neunet.2014.01.008.

[31] Y. Lucas & Jurgovsky, "Credit card fraud detection using machine learning: A survey". [Online]. https://www.researchgate.net/publication/344639091_Credit_card_fraud_detection_using_machine_learning_A_survey.

[32] A. Dal Pozzolo, G. Boracchi, O. Caelen, C. Alippi, & G. Bontempi,

"Credit card fraud detection and concept-drift adaptation with delayed supervised information", International Joint Conference on Neural Networks (IJCNN), Killarney, Ireland, 2015, pp. 1-8.https://doi.org/10.1109/IJCNN.2015.7280767.

[33] A. Izotova & A. Valiullin, "Comparison of Poisson process and machine learning algorithms approach for credit card fraud detection", Procedia Computer Science **186** (2021) 721. https://doi.org/10.1016/j.procs.2021.04.214.

[34] S. Arora, S. Bindra, S. Singh & V. Nassa, "Prediction of credit card defaults through data analysis and machine learning techniques", Materials Today: Proceedings **51** (2021) 110. https://doi.org/10.1016/j.matpr.2021.04.588.

[35] E. Burnaev, P. Erofeev & A. Papanov, "Influence of Resampling on Accuracy of Imbalanced Classification", International Conference on Machine Vision, Lille, France, 2015, pp. 5–12. http://dx.doi.org/10.1117/12.2228523

[36] T. Saito & M. Rehmsmeier, "The Precision-Recall Plot Is More Informative than the ROC Plot When Evaluating Binary Classifiers on Imbalanced Datasets", PLOS ONE **10** (2015) e0118432. https://doi.org/10.1371/journal.pone.0118432 .

[37] A. Dal Pozzolo, O. Caelen & Y. Le Borgne, "Learned lessons in credit card fraud detection from a practitioner perspective", Expert systems with applications **41** (2014) 4915. https://doi.org/10.1016/j.eswa.2014.02.026.

[38] A. Abdallah, M. Maarof & A. Zainal "Fraud detection system: A survey", Journal of Network and Computer Applications **68** (2016) 90. https://doi.org/10.1016/j.jnca.2016.04.007.

[39] J. A. P. Karax, A. Malucelli & J. P Barddal, "Decision tree-based feature ranking in concept drifting data streams", Proceedings of the 34th ACM/SIGAPP Symposium on Applied Computing, Limassol, Cyprus, 2019, pp. 590–592. https://doi.org/10.1145/3297280.3297551.

[40] H. M.Gomes, A. Bifet, J. Read, J. P. Ba rddal, F. Enembreck, B. Pfharinger, G. Holmes & T. Abdessalem, "Adaptive random forest for evolving data stream classification", Machine Learning **106** (2017) 1. https://link.springer.com/article/10.1007/s10994-017-5642-8.

[41] J. P. Barddal & F. Enembreck, "Learning regularized hoeffding trees from data streams", Symposium on Applied Computing, Limassol, Cyprus, 2019, pp. 574–581 https://doi.org/10.1145/3297280.3297334.

[42] F. Carcillo, A. Dal Pozzolo, Y. Le Borgne, O. Caelen, Y. Mazzer & G. Bontempi, "Scarff: A scalable framework for imbalanced classification in stream learning", Information Sciences **557** (2021) 317. https://doi.org/10.1016/j.ins.2020.11.033.

[43] F. Ahmed & R. Shamsuddin, "A comparative study of credit card fraud detection using the combination of machine learning techniques with data imbalance solution", 2nd International Conference on Computing and Data Science, Stanford, CA, USA, 2021, pp. 112–118. https://doi.org/10.1109/CDS52072.2021.00026.

[44] K. Kerwin & N. D. Bastian, "Stacked generalizations in imbalanced fraud datasets using resampling methods", Journal of Defense Modeling and Simulation: Applications, Methodology, Technology (2021); **18** (2021) 175. https://doi.org/10.1177/1548512920962219.

[45] S. Bagga, A. Goyal, N. Gupta & A. Goyal, "Credit card fraud detection using pipelining and ensemble learning", International Conference on Smart Sustainable Intelligent Computing and Applications under ICITETM2020. Procedia Computer Science **173** (2020) 104. https://doi.org/10.1016/j.procs.2020.06.014.

[46] S. Rajora, D. L. Li, C. Jha, N. Bharill, O. P. Patel, S. Joshi, D Putal & M.prsad, "A Comparative Study of Machine Learning Techniques for Credit Card Fraud Detection Based on Time Variance", IEEE Symposium Series on Computational Intelligence (SSCI), Bangalore, India, 2018, pp. 1958–1963. https://doi.org/10.1109/SSCI.2018.8628930.

[47] T. K. Dang, T. C Tran, L. M. Tuan & M. V Tiep, "Machine Learning based on Resampling Approaches and Deep Reinforcement Learning for Credit Card Fraud Detection Systems", Applied Sciences **11** (2021) 10004. https://doi.org/10.3390/app112110004.

[48] UCI Machine Learning Repository. [Online] http://archive.ics.uci.edu/ml/datasets/default+of+credit+card+clients. [Accessed 25 January 2022].

[49] gksj7. GitHub. [Online] https://github.com/gksj7/creditcardcsvpresent/blob/main/creditcardcsvpresent.csv. [Accessed 24 January 2022].

[50] Kaggle, "Credit Card Fraud Detection", [Online] https://www.kaggle.com/mlg-ulb/creditcardfraud. [Accessed 23 January 2022].

[51] Machine Learning Mastery, "Stacking Ensemble Machine Learning with Python", [Online] https://machinelearningmastery.com/stacking-ensemble-machine-learning-with-python/. [Accessed 2 Febraury 2022].

[52] U. Leonard. GitHub. [Online] https://github.com/UdezeLeoportals/Machine-learning/blob/main/ensemble_credit_rerun1.ipynb.