International Journal on Smart & Sustainable Intelligent Computing
Vol. 01, Iss. 01, S. No. 002, pp. 19-33, July 2024
ISSN (E):3048-8508

A Comprehensive Ensemble Approach Using Blending and Stacking for Credit Card Fraud Detection*

Bharti Chugh, Preeti Garg*, Karnika Dwivedi
(Affiliation of authors): CSE, KIET Group of Institutions Delhi-NCR Ghaziabad
bharti.kathpalia@gmail.com, preeti.itgarg@gmail.com*, dwivedikarnika1995@gmail.com

How to cite this paper: Bharti Chugh, Preeti Garg, Karnika Dwivedi, "A Comprehensive Ensemble Approach Using Blending and Stacking for Credit Card Fraud Detection," *International Journal on Smart & Sustainable Intelligent Computing*, Vol. no. 01, Iss. No 01, S No. 002, pp. 19–33, July 2024.

Received: 27/05/2024
Revised: 20/06/2024
Accepted: 30/06/2024
Published: 31/07/2024

Abstract

Detection of credit card fraud transactions is a severe problem, which requires analyzing large volumes of transaction data to identify fraud patterns. It requires finding, which transactions are fraudulent out of millions of daily transactions. As the amount of data is increasing, it is now difficult for an individual to detect meaningful patterns from transaction data, often characterized by many samples, many dimensions, and online updates. As a result, there is a need for the best possible approach using machine learning that automates the process of identifying fraudulent patterns from large volumes of data. Therefore, this study proposed a comprehensive ensemble approach using Blending (Voting Classifier) and stacking for credit card fraud detection. As the dataset is imbalanced, the proposed method balanced the dataset resampling techniques. Working with selected features instead of all the features reduces the risk of over-fitting, improves accuracy, and decreases the training time. Afterwards, three base classifiers with chosen features, an ensemble voting classifier and a Stacking Classifier have been developed. The computational results indicate that the suggested stacking ensemble is the best, and its Random Forest (RF) classifier has also the best performance among other base classifiers. The ensemble stacking classifiers lead to 82.5% recall,85% F1-score and 82.5 % AUC which has superiority over other ensemble-based methods.

Keywords

fraud detection, ensemble learning, balancing, feature selection, voting, credit card

1. Introduction

Detecting fraud in card payment transactions is a difficult task. As the online transactions are becoming very common in the recent scenario, it increases the number of fraudulent transactions. Though the count of these type of illegal transactions are very less as compared to normal transactions, the influence of a single fraud transaction can be massive in terms of financial loss. Online fraud detection and prevention continues to be a growing landscape for financial institutions and merchants. It also remains a priority with an ability to drive value across several use cases from banking and digital commerce to reduce risk to applications and device intelligence. Therefore, the banks and the financial institutions need to identify fraudulent transactions more effectively. There is an outburst of demand for new payment methods. New payment methods and an extremely complex backend make fraud detection all the harder. There is a loss of 1.8 billion Euros on an average of fraudulent transactions spotted in Europe every year. Global illegal transactions have enlarged by three times from $9.84 billion to $32.39 billion in less than a decade (2011 to 2020). The implementation of machine learning methodologies is now widespread for classifying transactions as artifice or not.

In this study a comprehensive analysis has been done to explore the impact of machine learning methodologies in credit card scam detection. The presented work defines the capabilities of ensemble methods using stacking and blending approach for detecting the authentic and fake transactions to secure the credit card transaction system. These are the primary contributions of the study:

(a) An improved prediction performance model for imbalanced credit card transaction datasets by combining heterogeneous machine learning models is designed.

(b) To handle the issues of imbalanced dataset and overfitting, the proposed model is implemented using stratified cross validation.

(c) Model performance on various resampling techniques has been evaluated.

(d) To increase the performance of the model, Hyperparameter tuning has been performed and its effects are studied.

(e) An ensemble consisting of multiple base learner models was executed in order to enhance the performance.

The subsequent sections comprise the structure of the document. In Section 2, the proposed methodology is explicated. The examination of the experiment analysis is detailed in Section 3. The discussion and results are presented in Section 4. The proposed work is contrasted with other state-of-the-art schemes in Section 5. The work is concluded in Section 6.

2. Related Work

A lot of supervised learning algorithms ended up underperforming as the size of the dataset increased dramatically after balancing [1,2]. The study conducted by the authors in [3] revealed that Random Forest (RF) outperformed Support Vector Machine (SVM) and Logistic Regression (LR) in scenarios characterized by class imbalance. Classification models that utilize resampling methods for preprocessing the training subset demonstrated higher accuracy compared to models trained on the original imbalanced training subset [4, 5, 6]. SMOTE and feature selection methods [7] are the commonly applied techniques for making the classes balanced. Feature selection helps to reduce irrelevant and redundant features from the dataset, and it improves learning performance [8]. Feature engineering such as identifying the optimal number of features can lead to significant enhancements in detecting credit card fraud. [9,10].

Kalra et. al [11] have found that Random Forest (RF) demonstrates superior results in identifying credit card fraud compared to Logistic Regression (LR) and the decision tree (DT) model. The development of ensemble learning algorithms involves amalgamating the results of base classifiers. Weighted voting strategies, which seek the best decision from the classifiers in ensemble models, are commonly employed for result improvement [12].

Singh et al. [13] employs six models, including DT, RF, Bayesian Network (BN), Naïve Bayes (NB), Support Vector Machine (SVM), and K*, to create an ensemble model for fraud detection. The authors propose a voting strategy that enhances accuracy by reducing the false alarm rate. Another study introduces a novel ensemble method by combining bagging and boosting techniques to enhance accuracy in financial fraud detection [14].

Authors in [15] utilizes ensemble techniques such as Bagging, Boosting, and Random Subspace, combining NB, Maximum Entropy (ME), DT, K-nearest Neighbor (KNN), and SVM. Kim et al. [16] proposed a champion-challenger framework for fraud detection based on a hybrid ensemble and deep learning. A bagging ensemble based on Decision Tree is constructed for predicting credit card fraud [17]. The authors explored the model's performance, revealing that, the bagging ensemble outperforms SVM, NB, and KNN algorithms. The combination of RF and rough set theory is suggested as an efficient model for fraud detection by authors in [18].

3. Proposed Methodology

In this work the German credit card dataset is taken from the UCI Machine learning repository [19] for implementing the proposed technique. The dataset has 1000 samples with 21 features out of which 17 are categorical and 4 are numeric. The ‘Class’ attribute is a target variable here. The feature class values of 1 and 0 correspond to transactions that are fake and authentic, correspondingly. The dataset is imbalanced as 300 transactions are fraudulent out of 1000 transactions. In the proposed work the concept of blending and stacking are used to improve the performance [20,21]. Blending, also known as model averaging, involves training multiple diverse models independently and then combining their predictions through a weighted average or a voting mechanism. Each base model contributes its prediction to the final output, and the blending process aims to mitigate the weaknesses of individual models. Stacking, also known as meta-classification or combiner, is the process of combining the predictions of a collection of disparate base models that have been trained [22,23]. The meta-model accumulates knowledge to generate the final prediction using the predictions from the base models as input features.

The German credit card dataset is preprocessed, as Fig. 1 illustrates, to improve its fit for machine learning analysis. The dataset is resampled after preprocessing to correct the class imbalance and provide a balanced dataset. The best resampling technique is determined after testing several approaches. There are three sets of the dataset: training, validation, and testing. It generates a set S of ML classifiers with different settings and methodologies. Stratified K-fold cross-validation, a technique that guarantees a balanced representation of classes in each fold, is used to validate the three best-performing models (M1, M2, and M3) that are chosen from S based on their F1-Score, a measure frequently used to evaluate classification models. The classifiers (M1, M2, and M3) that made the shortlist are subjected to hyperparameter optimization to optimize their parameters and enhance their overall performance. M1, M2, and M3 are then subjected to blending and stacking procedures, resulting in ensemble models that combine the advantages of individual models for improved predictive power.

Using the ensemble and optimized models, predictions are finally produced on the test data, offering an understanding of the generalization and performance capabilities of the machine-learning solutions that have been developed. Algorithm 1 explains the step by step procedure from preprocessing of the data set to the prediction.
Fig 2 shows the steps performed for data preprocessing which involves handling the missing values in the dataset, then removing the duplicate values. Step 3 is data normalization while step 4 involves transformation of data into required shape.

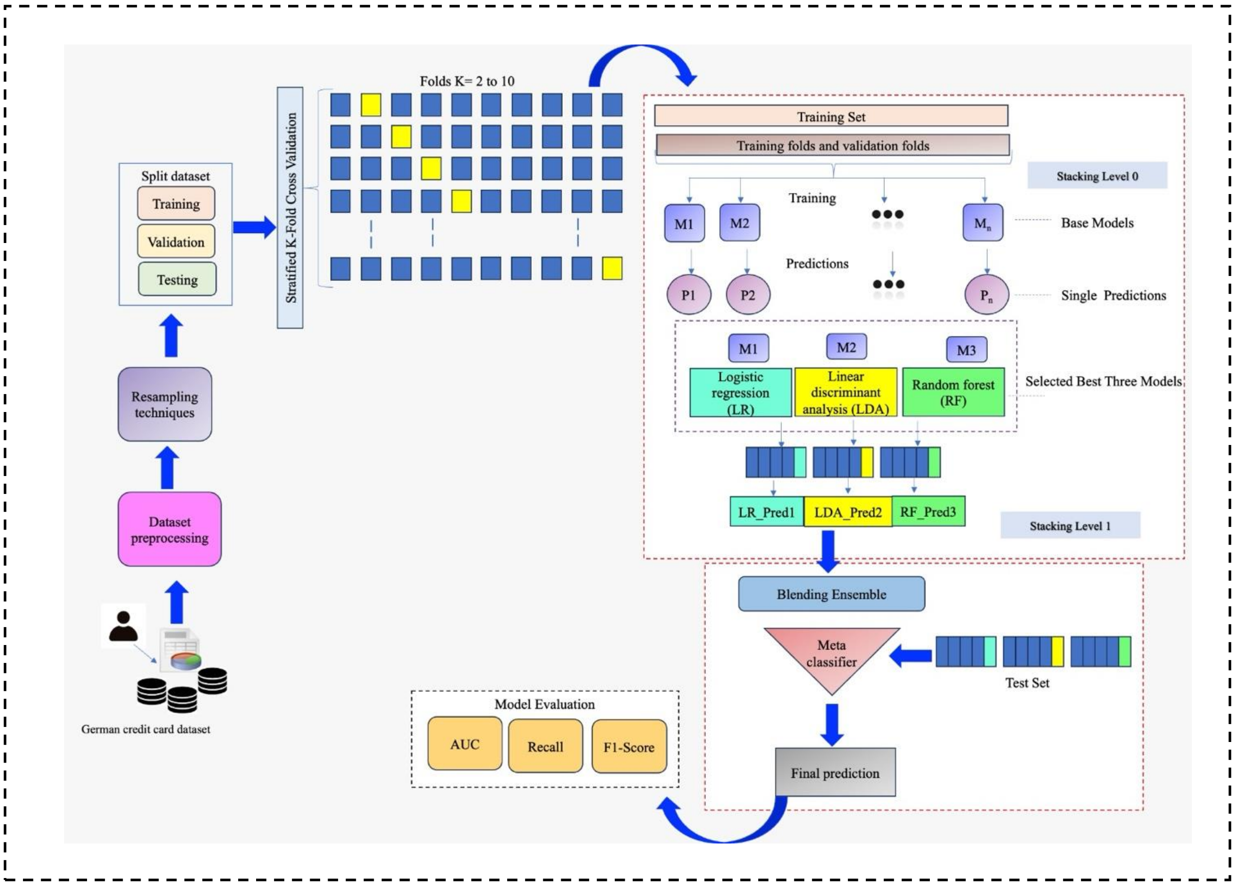

Fig 1 Block Diagram for the proposed methodology

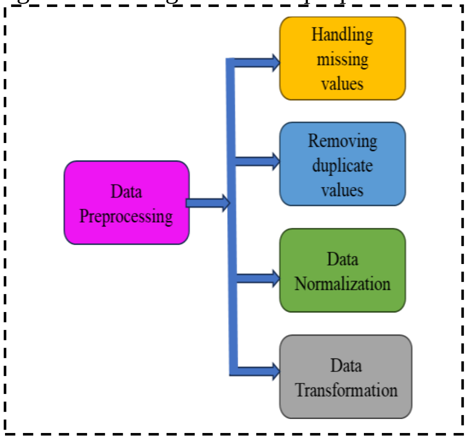

Fig 2: Block diagram of Data Pre-processing

Fig 3 shows the complete flow process of the proposed work. It shows that the processed data is passed into 14 different machine learning models, out of which three models are selected whose F1-score is higher than the Dummy Classifier. The hyperparameter tuning is done on these three models. Thereafter, the models are validated using stratified K fold CV method. After validation, ensemble model is created using Blending (voting Classifier is a blend of LR, LDA, RF) and Stacking (stacking classifier) techniques. The ensemble classifiers are finally evaluated on the different evaluation metrics.

ALGORITHM 1: The proposed Algorithm
Step I: Preprocessing of the German credit card dataset D.
Step II: Apply resampling techniques on D and identify the best among them. Generate balanced dataset $D_{balanced}$.
Step III: The set S of Machine Learning models are generated.
Step IV: The best three models (M1, M2 and M3) are selected from S based on the F1-Score.
Step V: M1, M2 and M3 are validated using stratified K-foldCV.
Step VI: Hyperparameter optimization is carried out on the shortlisted models.
Step VII: Perform blending and stacking on M1, M2 and M3 to create ensemble models.
Step VIII: Prediction on test data.

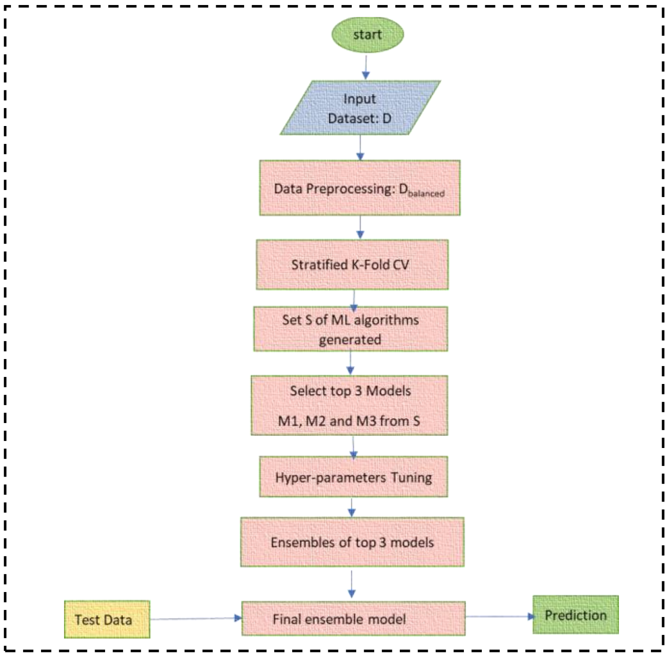
start
Input Dataset: D
Data Preprocessing: D_balanced
Stratified K-Fold CV
Set S of ML algorithms generated
Select top 3 Models M1, M2 and M3 from S
Hyper-parameters Tuning
Ensembles of top 3 models
Test Data
Final ensemble model
Prediction

Fig 3: Flowchart for the proposed approach

4 Figures and Tables

4.1 Data Preprocessing

An examination is conducted on the dataset to identify any missing, duplicate, or garbage values. The dataset is standardized using z-score as given in eq (1)

$$z = ((x - u))/s \qquad (1)$$

where u is the mean and s is the standard deviation. The standardization ensures data is centered around 0 and a unit standard deviation.

The transformation modifies the distribution's geometry to produce transformed data that adheres to a normal or nearly normal distribution. Yeo-Johnson transformation is applied on the numerical features to make it follow a normal distribution. This transformation is given in eq. (2)

$$\varphi(y,\lambda) = \begin{cases} \frac{(y+1)^{\lambda}-1}{\lambda} & y \geq 0 \ \ and \ \ \lambda \neq 0, \\ log(y+1) & y \geq 0 \ \ and \ \ \lambda = 0, \\ -\frac{(-y+1)^{2-\lambda}-1}{2-\lambda} & y < 0 \ \ and \ \ \lambda \neq 2, \\ -log(-y+1) & y < 0, \lambda = 2 \end{cases}$$

(2)

where λ is a transformation parameter.

Multi-collinearity is a state of high intercorrelations or inter-associations among the independent features in the dataset. It may reduce the coefficient value of the model and may result in unpredictable variance. The credit_amount and duration features in the dataset are highly correlated with each other so the features are grouped together to reduce multicollinearity.

4.2 ML Models

The following supervised learning algorithms in which the class label of each training tuple is already known are used to build a binary classifier [24].

- Logistic regression (LR) utilizes a logistic function to model the potential outcomes of a solitary sample and calculate the corresponding probabilities [25]. The logistic sigmoid function in eq. (3) is used to map predictions to probabilities.

$$f(x) = 1/(1+e\hat{}((-x)\ )\ )$$

(3)

- In order to generate predictions, Linear Discriminant Analysis (LDA) computes the likelihood that a given set of inputs is a member of each class by calculating posterior probability $Pr(Y = k \mid X = x)$ as represented in eq. (4)

$$\Pr(Y = k|X = x) = \frac{pi_k f_k(x)}{\sum_{l=1}^{K} pi_k f_k(x)} \quad (4)$$

  Where Y is the response variable, k is the number of classes $⟦pi⟧\_k$ is prior, $f\_k \ (x)$ is a density function

- Ridge Regression adds L2 penalty as a regularization term to the loss function in multiple regression to figure out what class it goes to and is represented in eq (5)

$$E(w) = \frac{1}{2} \ \sum_{i=1}^{N} \{f(x_i, w) - y_i\}^2 + \frac{\lambda}{2} \parallel w \parallel^2 \quad (5)$$

  Where $\{(x\_i, w) \dashv - \vdash y\_i \ \}\hat{}2$ is a loss function, y_i is target value, $\lambda/2 \parallel w \parallel^2$ is regularization term, N denotes size of training data and λ is regularization parameter.

- K nearest neighbors are identified by the K-nearest neighbours (KNN). A fine-tuned value of K is required to achieve improved classification; in this process, the input x is allocated to the class that has the highest probability. $P(y = j \mid X = x)$ as presented in eq. (6)

$$P(y = j|X = x) = \frac{1}{K} \sum_{i \epsilon A} I(y^i = j) \quad (6)$$

  Where x is input.

- Each data item is represented as a point in n-dimensional space by the SVM algorithm, and classification is accomplished by locating the hyperplane that distinguishes the two classes as represented in eq. (7)

$$f(x) = \sum_{i}^{N} \alpha_i y_i k(x_i, x) + b \quad (7)$$

  Where $k(x\_i, x)$ is kernel, x_i is support vectors and $\alpha_(i \ )$is weight.

- A Naive Bayes (NB) Classifier gives the conditional probability $P(y\_i \mid x\_1, \ldots, x\_n)$ of each feature and multiply them together as presented in eq (8)

$$P(y_i|z_1, \ldots, z_n) = P(z_1, z_2, \ldots z_n|y_i).\frac{P(y_i)}{P(z_1, z_2 \ldots z_n)} \qquad (8)$$

  where $P(y\_i)$ is prior.
- A decision tree (DT) is a graphical representation resembling a flowchart, wherein every internal node corresponds to a feature test, every leaf node signifies a class label, and every branch represents a conjunction of features that results in the specified class labels. Classification rules depict the paths from root to leaf utilizing the most widely used selection measures. The term "entropy" E(S) is calculated using eq. (9)

$$E(S) = -p_{(+)} log p_{(+)} - p_{(-)} log_{(-)} \qquad (9)$$

  Here p_+ is the positive class's probability, p_- is the probability of the negative class and S is subset of training examples.
- Ensemble learning algorithms combine the predictions from multiple diverse base classifiers and are expected to perform better than any contributing base model. A random forest (RF) is an estimator that employs a consensus mechanism to determine the class label after fitting multiple decision tree classifiers to different subsamples and sub-features of the dataset. The importance for each feature on a decision tree is calculated as given in eq. (10)

$$fi_i = \frac{\sum j{:}node\ j\ splits\ on\ feature\ i\ ni_j}{\sum_{k \in all\ nodes} ni_k} \qquad (10)$$

  The values " f_i " and " n_i " represent the significance of feature i and node j, respectively.
- Extra Trees is faster algorithm than RF and use the whole original sample and choose arbitrarily the split point that reduces the variance.
- Gradient Boosting [26] minimizes a loss function in a iterative fashion to find the points towards the negative gradient. The loss function (L) is given by

$$L = \frac{1}{N}\sum_{i=0}^{N}(y_i - \gamma_i)^2 \qquad (11)$$

  Where y_i is the observed and gamma is the predicted value and N are number of data points.
- LightGBM extends the gradient boosting algorithm that grows vertically. The decision tree divides each node to the largest evidence gain. It can be represented by eq (12)

$$Y = Base_{tree(X)} - lr * Tree1(X) - lr * Tree2(X) - lr * Tree3(X) \qquad (12)$$

  Where Y is prediction, X is input space, lr* are the nodes in decision tree.
- AdaBoost reassigns the weights to each data instance and recalculates the error. The weight $w(x\_i\ y\_i)$ is calculated as in eq(13)

$$w(x_i y_i) = \frac{1}{N}, i = 1,2, \ldots..n \qquad (13)$$

  where n represents sample size.

- The Voting Classifier (blending) is designed to predict class labels by combining diverse machine learning classifiers and employing a majority vote (hard vote).
- Stacking involves the training of multiple models to forecast the outcome, followed by the construction of a meta-model that augments the original features with the predictions from those models. Then the results of the individual classifiers (LR, LDA and RF) are aggregated to form an ensemble voting classifier and ensemble stacking classifier.

4.3 Stratified K fold Cross-Validation

Across all K folds, this cross-validation preserves the identical class ratio that was present in the original dataset. By ensuring that no value is excessively or inadequately represented in both the training and test sets, an accurate estimation of the error or performance is obtained.

4.4 Hyperparameter Optimization

The hyperparameter values when optimized can build a high-performance model, The authors enable the suggested models to experiment with various hyperparameter combinations throughout the training procedure and generate predictions using the optimal combination of hyperparameter values. Table 1 list the seven hyperparameter optimization algorithms that are used in proposed approach and Table 2 represents the hyperparameter values corresponding to the selected models.

Table 1: Hyperparameter Optimization Algorithms

ML Library	Optimization Algorithms
Sklearn	GridSearchCV
	RandomizedSearchCV
Optuna [27]	Tree-structured Parzen Estimator (TPE)
	CMA-ES
	Grid Search
	Random Search
scikit-optimize [28]	BayesSearchCV

The Tree-structured Parzen Estimator (TPE) is a sequential model-based optimization (SMBO) approach which sequentially construct models to approximate the performance of hyperparameters based on historical measurements and then subsequently choose new hyperparameters to test based on this model.
Covariance Matrix Adaption Evolutionary (CMA-ES) It is a method of evolution for continuous function optimization in which the search resolution is dynamically adjusted per hyperparameter, enabling searches at various scales to be conducted efficiently. By applying Bayesian optimization to the selection of the subsequent hyperparameter set for evaluation, it is possible to improve generalization performance on the test set and reduce the time required to arrive at the optimal set of parameters by leveraging information from the hyperparameter combinations encountered thus far.

4.5 Performance Evaluation Metrics

The proposed work has been evaluated using various parameters as described below.

Accuracy [29,30,31] is the proportion of correct predictions made by a model as shown in eq (14).

$$Accuracy = (True\ \ Positive + True\ \ Negative)/(True\ \ Positive + True\ \ Negative + False\ \ Negative + False\ \ Positive) \tag{14}$$

Recall attempts to determine what percentage of true positives were accurately identified. as shown in eq. (15)

$$Recall = (True\ \ Positive)/(True\ \ Positive + False\ \ Negative) \tag{15}$$

Recall and precision are two essential model evaluation metrics. Precision denotes the proportion of pertinent results, while recall signifies the proportion of the total relevant results that our classifier accurately classified. Recall is a better metric than the precision as far as classification of fraudulent transactions are concerned. It does so by decreasing the false negative and increasing the true positive, thereby decreasing the error rate. It is not possible to maximize precision & recall simultaneously for the same sample size and when both precision and recall are important, we take their harmonic mean which is termed as F1-Score as shown in eq. (16).

$$F1 = 2 * (precision * recall)/(precision + recall) \qquad (16)$$

5. Result and Discussion

The computer system used to perform the experiments was Mac OS with an Apple M1 chip processor having 8GB RAM. The development environment used for running all the experiments is given as follows: Jupitor Notebook, Python 3.7.9 Pandas, OpenCV, Numpy and Matplotlib. The proposed mechanism is implemented on German credit card dataset. The performance indicators are calculated and the comparison of these is shown in Table 3. Out of these classifiers, eight classifiers perform better than the dummy classifiers as shown in the Table. The performance of these 8 classifiers was marginally better on test data compared to the training data that proves there is no overfitting and model generalizes well. The machine learning model performance is compared with the baseline model 'DummyClassifier' generated by making use of Sklearn library. The dummy classifier makes predictions based on most frequent label in the data. Table 4 illustrates the model performance when six oversampling and ten under sampling techniques are applied to the imbalanced dataset, respectively. It can be decided that SMOTE and Tomek Links gives the best performance among all resampling techniques and the results are shown in Table 5. Here K ranges from 2 to 10 and average of the performance evaluation metric over 10 iterations is considered. Fig. 4 indicates the comparison of ML models those are giving best results with hyperparameter optimization. Table 6 indicates the comparison of ensemble ML models using voting classifier and stacking classifier.

Table 2: Hyperparameter Values for selected models

Model Name	Hyper-Parameter	Value
	bootstrap	TRUE
	class_weight	none
	criterion	gini
	max_features	auto
RF	max_depth	none
	max_samples	none
	n_estimators	100
	oob_score	FALSE
	max_leaf_nodes	none
	min_samples	2
	Alpha	1.492352
	class_weight	none
Ridge	fit_intercept	TRUE
	max_iter	none
	normalize	TRUE

	random_state	123
	Solver	auto
	Tol	0.001
	n_components	none
LDA	Priors	none
Analysis (LDA)	shrinkage	0.107681
	Solver	isqr
	store_covariance	FALSE
	Tol	0.0001

Table 3: Comparison of ML models Performance

Model	**AUC (%)**	**Recall (%)**	**F1-Score (%)**
RF	83.4	93.0	83.8
Ridge	70	88	83.1
LDA	80.3	86	82.3
LR	81.5	86.5	82.2
ET	81.2	89.5	82.19
LGBM	80.2	87	82.16
Ada Boost	79.4	83.5	81.9
GBC	80.9	87	81.7
KNN	76.9	89	81.5
Dummy Classifier	0.5	1	80
SVM	66	83	79.7
DT	61.8	75.6	75
QDA	54.5	81	74.7
NB	77.3	21.8	34.9

Table 4: Model performance on Resampling techniques

Resampling Techniques	**RF**			**Ridge**			**LDA**		
	AUC (%)	Recall (%)	F1-Score (%)	AUC (%)	Recall (%)	F1-Score (%)	AUC (%)	Recall (%)	F1-Score (%)
SMOTE	88.3	94.1	83.8	82.5	90.2	83.1	85	85.6	82.3
ADASYSN	89	85.5	83	79.8	83.2	77.1	79.3	79.8	77.8
Random Over Sampler	84.5	82.3	83.6	74.6	79.8	78.3	78.5	75.5	78.3
Borderline SMOTE	81.1	92	83.3	73.5	72.4	78.1	79	71.6	77.8
SVMSMOTE	82.3	90	83.2	74.8	78.6	81.4	80.9	77.6	82.3
CC	77	39.8	55.1	71.3	70.6	76.5	77.7	70.1	76.2
RUS	80.7	62.6	73.9	72.8	66.6	75.2	79.4	65.6	74.7
Near Miss	79.4	64.1	74.7	70.3	61.6	71.6	75.8	61.1	71.1
Tomek Links	83.5	90.5	84	70.7	85.5	82.4	80.2	84	81.8

ENN	81.9	64.6	74.7	75	71.1	78.3	80.4	70.6	78
R-ENN	80.8	48.7	63.6	70.8	52.7	66.6	81.2	52.7	66.8
ALL-KNN	80.9	59.7	71.4	75	65.1	75.5	81.1	64.6	74.9
C-NN	79.8	70.6	76.9	72	71.1	77	79.5	70.6	76.5
One Sided Selection	81.2	89.5	82.7	70.7	85.5	82.4	73.3	73.6	82.1
NCR	82	75.1	79.6	80.2	84.5	78.7	79	74.1	79.2

Table 5: Comparison of top 3 models after resampling

Model+ Resampling Techniques	AUC (%)	Recall (%)	F1-Score (%)
RF+TomekLinks	83.5	90.5	84
Ridge+ SMOTE	82.5	90.2	83.1
LDA+ SMOTE	85	85.6	82.3

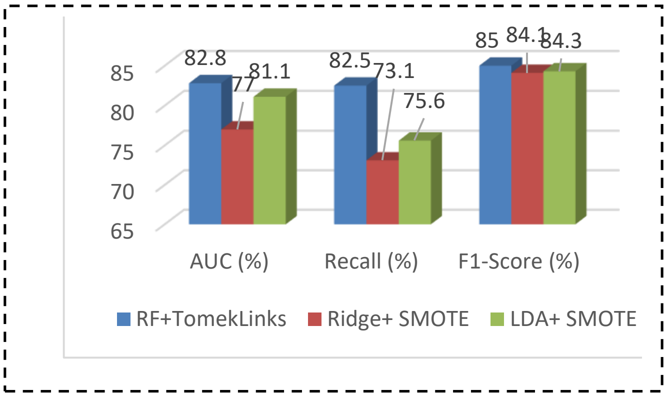

Fig 4: ML models performance after hyperparameter optimization

Table 7: Ensemble models with Voting and Stacking Classifier

	AUC (%)	**Recall (%)**	**F1-Score (%)**
Voting Classifier	75.3	73.6	80
Stacking Classifier	80.1	86	**83.3**

The confusion matrix for the ensemble voting classifier is presented in Table 8. It provides the following breakdown of the number of accurate and inaccurate predictions for each class: The count is as follows: 175 (TN), 52 (TP), and 48 (FP) for the Recall, F1-Score, Accuracy, and AUC-ROC curves, respectively.

Table 8: Confusion Matrix for Voting Classifier

		Actual Class	
		0	1
Predicted Class	0	59	41
	1	28	173

Fig. 5 shows the classification report for the ensemble voting classifier. It shows that the F1-score achieved by the proposed work is 83.4%.

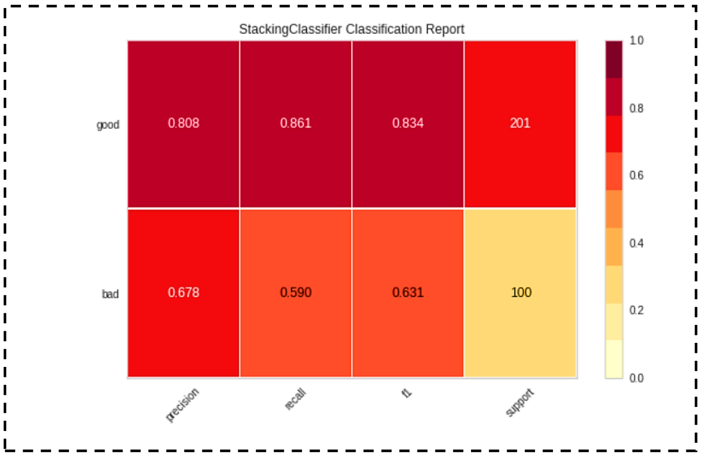

Fig 5: Classification Report for Voting Classifier

Fig. 6 illustrates the AU ROC curve, which represents the current distinction between all positive and negative points. An AUC value exceeding 0.5 indicates a significant probability that the classifier accurately anticipates the values. In the proposed work the authors have achieved AUC value 0.8 which proves its better performance. The precision-recall curve in Fig. 7 demonstrates the trade-off between precision and recall ensemble voting classifier. Red distribution shows the Average precision. Blue distribution depicts the precision-recall curve for fraud and non-fraud transactions

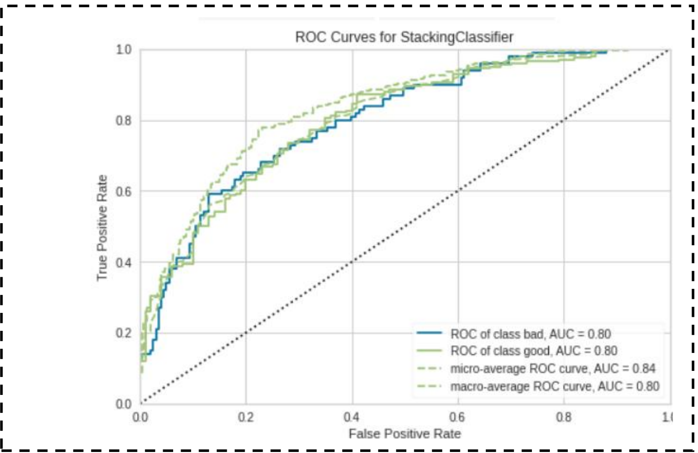

Fig 6: AU ROC curve for stacking classifier

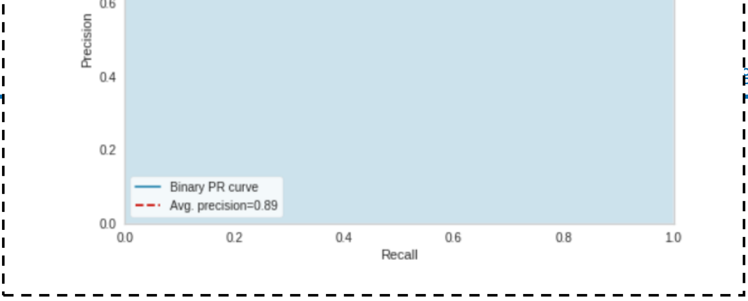

Fig 7: Precision-Recall curve for stacking classifier

6. Comparative Analysis

After a thorough examination of fourteen models and eighty-four evaluation metric values, the stacking classifier (an ensemble of RF, Ridge, and LDA) is the most effective model proposed. To illustrate this, the proposed model is contrasted with prior research employing a comparable methodology and dataset. The comparison with prior research is presented in Table 11. It is evident that the F1-score of the projected technique (83.3) surpasses that of other state-of-the-art algorithms.

Table 11 Comparative Analysis with previous work

Ref	Year	Model	F1-Score (%)
[32]	2022	Adaboost	68
[33]	2020	LightGBM	56.9
[27]	2020	RF	76
[21]	2019	AdaBoost	80.2
Pro-posed	2024	Proposed ensemble	83.3

7. Conclusion

The primary objective of this work is to enhance accuracy of the model. To achieve this goal, a balanced dataset is crucial, as an unbalanced dataset can introduce bias toward the class with a larger number of samples. Each classifier has its own advantages based on the type of data used. The objective is to combine the strengths of multiple classifiers; it was observed that the resulting output was superior to that of each individual classifier. The initial phase seeks to identify the eight classifiers whose F1 scores are higher than those of the dummy classifier. Each model is evaluated based on F1 score. In the first stage the eight classifiers are RF, DT, NB, Ridge, LR, ET, LGB, GB and KNN. In the second stage the three algorithms giving good results are found, which are RF, LDA and Ridge. Then these three techniques are ensembled using Voting Classifier and Stacking classifier. Finally, Random Forest among base classifier outperforms previous works with the same dataset in terms of F1 score (85%) and among ensemble stacking classifier give best result with F1 score (83.3%). The future work may include additional datasets to work upon.

Acronym

Table 12 shows the full form of all the Acronym used in the paper.

Acronym	Definition
AdaBoost	Adaptive Boosting
AUC	Area Under Curve
CC	Cluster Centroids
C-NN	Condensed Nearest Neighbour
DT	Decision Tree
ENN	Edited Nearest Neighbours
ET	Extra Tree Classifier
GB	Gradient Boosting
KNN	K-Nearest Neighbor
LDA	Linear Discriminant Analysis

LGB	Light Gradient Boosting
LR	Logistic Regression
ML	Machine Learning
NB	Naïve Bayes
NCR	Neighbourhood Cleaning Rule
R-ENN	Repeated Edited Nearest
RF	Random Forest
ROC	Receiver Operating Characteristic Curve
RUS	Random Under Sampler
SMOTE	Synthetic Minority Oversampling Tech-
SVM	Support Vector Machine

Funding: "This research received no external funding"

Conflicts of Interest: "The authors declare no conflict of interest."

References

[1] Mallidi, Manoj Kumar Reddy, and Yeshwanth Zagabathuni. "Analysis of Credit Card Fraud Detection using Machine Learning models on balanced and imbalanced datasets." *International Journal of Emerging Trends in Engineering Research* 9.7 (2021).

[2] Tiwari, Pooja, et al. "Credit Card Fraud Detection using Machine Learning: A Study." *arXiv preprint arXiv:2108.10005* (2021).

[3] Xie, Yalong, et al. "A Heterogeneous Ensemble Learning Model Based on Data Distribution for Credit Card Fraud Detection". *Wireless Communications and Mobile Computing* 2021 (2021).

[4] Dal Pozzolo, A., Caelen, O., Le Borgne, Y.-A., Waterschoot, S. and Bontempi, G. "Learned lessons in credit card fraud detection from a practitioner perspective". *Expert Syst. Appl.*, 41, 4915–4928 (2014).

[5] Díez-Pastor, J.F., Rodríguez, J.J., García-Osorio, C.I. and Kuncheva, L.I. "Diversity techniques improve the performance of the best imbalance learning ensembles". *Inf. Sci.*, 325, 98–117 (2015).

[6] Kültür, Y. and Çaglayan, M.U. "Hybrid approaches for detecting credit card fraud". *Expert. Syst.*, 34, 1–13 (2017).

[7] Fadaei Noghani, F., Moattar, M.H. "Ensemble classification and extended feature selection for credit card fraud detection". *J. AI Data Min.* 5(2), 235–243 (2017).

[8] Kuncheva, L.I. and Rodríguez, J.J. "A weighted voting framework for classifiers ensembles". *Knowl. Inf. Syst.*, 38, 259–275 (2014).

[9] Akila, S., Reddy, U.S. "Cost-sensitive risk induced bayesian inference bagging (ribib) for credit card fraud detection". *J. Comput. Sci*. 27, 247–254 (2018)

[10] Kim, E., Lee, J.; Shin, H., Yang, H., Cho, S.; Nam, S.k., Song, Y., Yoon, J.a., Kim, J.i., "Champion-challenger analysis for credit card fraud detection: Hybrid ensemble and deep learning". *Expert Syst. Appl*. 128, 214–224 (2019).

[11] Kalra M. and Patni J.. "Playing Doom with Deep Reinforcement Learning". *International Journal of Computer Applications*, vol. 1, pp.14-20, (2019).

[12] Singh, Ajeet, and Anurag Jain. "Adaptive credit card fraud detection techniques based on feature selection method." *Advances in computer communication and computational sciences*. Springer, Singapore, 167-178 (2019).

[13] Bian, Y., Cheng, M., Yang, C., Yuan, Y., Li, Q., Zhao, J.L. and Liang, L. "Financial Fraud Detection: A New Ensemble Learning Approach for Imbalanced Data". *In Proc. PACIS 2016. Chiayi, Taiwan, June 27, PACIS, Taiwan* (2016).

[14] Wang, G., Sun, J., Ma, J., Xu, K. and Gu, J. "Sentiment classification: The contribution of ensemble learning". *Decis. Support. Syst.*, 57, 77–93 (2014).

[15] Krawczyk, B. and Wozniak, M. "Untrained weighted classifier combination with embedded ensemble pruning". *Neurocomputing*, 196, 14–22 (2016).

[16] M. Zareapoor, P. Shamsolmoali, and others, "Application of credit card fraud detection: Based on bagging ensemble classifier". *Procedia Comput. Sci*., vol. 48, pp. 679– 685, (2015). https://doi.org/10.1016/j.procs.2015.04.201.

[17] F. H. Chen, D.-J. Chi, and J.-Y. Zhu, “Application of Random Forest, Rough Set Theory, Decision Tree and Neural Network to Detect Financial Statement Fraud--Taking Corporate Governance into Consideration,” in *International Conference on Intelligent Computing*, pp. 221–234 (2014). https://doi.org/10.1007/978-3-319-09333-8_24

[18] S. Patil, V. Nemade, and P. K. Soni, “Predictive Modelling For Credit Card Fraud Detection Using Data Analytics”. *Procedia Comput. Sci.*, vol. 132, pp. 385–395, (2018). https://doi.org/10.1016/j.procs.2018.05.199.

[19] UCI Machine Learning Repository: Statlog (German Credit Data) Data Set Available at: https://archive.ics.uci.edu/dataset/144/statlog+german+credit+data

[20] Correa Bahnsen, A., Aouada, D., Stojanovic, A. and Ottersten, B. “Feature engineering strategies for credit card fraud detection”. *Expert Syst. Appl*., 51, 134–142, (2016).

[21] Singh, Ajeet, and Anurag Jain. "Adaptive credit card fraud detection techniques based on feature selection method." *Advances in computer communication and computational sciences*. Springer, Singapore, 167-178 (2019).

[22] Sohony, Ishan, Rameshwar Pratap, and Ullas Nambiar. "Ensemble learning for credit card fraud detection." *Proceedings of the ACM India Joint International Conference on Data Science and Management of Data*. 2018.

[23] Patil, Suraj, Varsha Nemade, and Piyush Kumar Soni. "Predictive modelling for credit card fraud detection using data analytics." *Procedia computer science,* 132 385-395, (2018).

[24] Alfaiz, Noor Saleh, and Suliman Mohamed Fati. "Enhanced Credit Card Fraud Detection Model Using Machine Learning." *Electronics* 11.4 :662, (2022).

[25] Chugh, Bharti, and Nitin Malik. "Machine Learning Classifiers for Detecting Credit Card Fraudulent Transactions." *Information and Communication Technology for Competitive Strategies (ICTCS 2021) ICT: Applications and Social Interfaces*. Singapore: Springer Nature Singapore. 223-231, (2022).

[26] Zou, Yao, and Changchun Gao. "Extreme Learning Machine Enhanced Gradient Boosting for Credit Scoring." *Algorithms* 15.5: 149, (2022).

[27] Aung, Maung Hein, et al. "Random Forest Classifier for Detecting Credit Card Fraud based on Performance Metrics." *2020 IEEE Asia-Pacific Conference on Computer Science and Data Engineering (CSDE)*. IEEE, 2020.

[28] He, H., Garcia, E.A. “Learning from imbalanced data”. *IEEE Trans. Knowl. Data Eng*, 21, 1263–1284, (2009).

[29] Scikit-Learn-Contrib. Imbalanced-Learn. Available online: https://github.com/scikit-learn-contrib/imbalanced-learn (accessed on 22 January 2022).

[30] Saheed, Y.K., Baba, U.A. and Raji, M.A. (2022), "Big Data Analytics for Credit Card Fraud Detection Using Supervised Machine Learning Models", Sood, K., Balusamy, B., Grima, S. and Marano, P. (Ed.) Big Data Analytics in the Insurance Market (Emerald Studies in Finance, Insurance, and Risk Management), Emerald Publishing Limited, Leeds, pp. 31-56. https://doi.org/10.1108/978-1-80262-637-720221003.

[31] Estelami, H. and Liu, K. (2024), "Content analysis of American consumers’ credit card fraud complaints filed with the Consumer Financial Protection Bureau", *Journal of Financial Crime*, Vol. 31 No. 3, pp. 618-628. https://doi.org/10.1108/JFC-03-2023-0070

[32] Chaudhari, Anagha. "Credit Card Fraud Detection Using Machine Learning and Predictive Models: A Comparative Study." *Hybrid Intelligent Systems: 21st International Conference on Hybrid Intelligent Systems (HIS 2021), December 14-16, 2021*. Vol. 420. Springer Nature, 2022.

[33] Taha, Altyeb Altaher, and Sharaf Jameel Malebary. "An intelligent approach to credit card fraud detection using an optimized light gradient boosting machine." *IEEE Access* 8: 25579-25587, (2020).