Mansoura Journal for Computer and Information Sciences (2025), Vol.**20**, No.**1**
doi:10.21608/mjcis.2025.313097.1007

Mansoura Journal for Computer and Information Sciences

ARTICLE

Enhancing Fraud Detection in Imbalanced Datasets: A Comparative Study of Machine Learning and Deep Learning Algorithms with SMOTE Preprocessing

Walaa Salah Radi,* Ibrahim M. El-Hasnony, Ahmed Aboelfetouh, and Amira Rezk

Department of Information Systems, Faculty of Computers and Information, Mansoura University, Mansoura, Egypt
*Corresponding author: walaa79@mans.edu.eg

(Received: 24 September 2024; Accepted: 24 February 2025; Published: 28 February 2025)

Abstract
Fraud detection has become a critical challenge, particularly with the growth of e-commerce. Financial institutions are under increasing pressure to develop robust systems to mitigate significant economic losses due to fraudulent activities. A key difficulty in detecting credit card fraud is the imbalance of data sets, where fraudulent transactions are far fewer than legitimate ones. This imbalance often results in models struggling to effectively recognize fraud.

To address this issue, various techniques have been developed. The Synthetic Minority Oversampling Technique (SMOTE) is widely used to create synthetic instances and balance the data set. Other strategies include under-sampling, which reduces the number of legitimate transactions, and cost-sensitive learning, which assigns different costs to misclassifications to prioritize fraud detection. Advanced SMOTE variants, such as Borderline-SMOTE and ADASYN, further enhance the balance of data by focusing on complex samples.

This paper examines how data preprocessing affects the performance of several machine learning and deep learning algorithms. Key preprocessing steps include data cleaning, normalization, feature selection, and SMOTE application. The cleaned and normalized data set ensures quality and comparability, while feature selection reduces dimensionality. The application of SMOTE directly addresses the class imbalance.

The preprocessed data are evaluated using Support Vector Machines (SVM), Random Forests (RF), Convolitional Neural Networks (CNN), and Long-Short-Term Memory Networks (LSTMs). These algorithms are assessed for their ability to detect fraud after pre-processing. Comparative analyses confirm the effectiveness of SMOTE, showing improved performance across all algorithms. Metrics such as accuracy, precision, recall, and F1 score exhibit high results, with CNN achieving the highest performance (95% accuracy and

94% F1 score), followed by RF, LSTM, and SVM. Although SMOTE enhanced SVM performance, it did not match CNN or RF levels. These findings highlight the significant improvements that data pre-processing can yield, providing valuable insights for improving fraud detection systems.

Keywords: Fraud Detection; Machine Learning; Deep Learning; SMOTE; Finance

1. Introduction

Fraud detection involves the rapid identification of actual fraud, particularly after preventive measures have failed. Fraudsters typically take advantage of vulnerabilities within systems meant to protect data. Although preventing fraud presents significant challenges, fraud detection is a vital component of the broader strategy to combat fraudulent activities. Gaining insights into the weaknesses that contribute to fraud is crucial for enhancing both detection and prevention strategies. [1].

The rapid expansion of e-commerce has led to a significant increase in the volume of credit card transactions. As a result of this rise, credit card fraud cases have become more dangerous and complex. The growing number of transactions presents new challenges in identifying and preventing fraudulent activities, making it crucial for businesses and consumers to remain vigilant and adopt robust security measures [2]. Financial organizations are actively enhancing their fraud detection mechanisms by incorporating data mining, machine learning, and deep learning technologies. These advanced techniques allow for better analysis of transaction patterns and the identification of potential fraud in real-time. By leveraging these technologies, organizations can improve their ability to detect unusual behaviors and respond more effectively to fraudulent activities [3]. Classifiers have become essential tools for improving classification performance. They offer predictions regarding the categories of input data samples, which are then analyzed and combined using fusion strategies. This approach enhances the accuracy and robustness of classification systems by integrating multiple predictions and leveraging the strengths of different classifiers [4].

Fraud on credit cards results in billions of dollars in costs due to inadequate fraud detection mechanisms, enabling fraudulent activity to occur through various methods. Fraud typically falls into three categories: classic card-related fraud, merchant-related fraud, and online fraud. This paper will employ deep learning techniques to create a robust model that can detect credit card fraud using specific input data, thereby addressing this issue more effectively [5]. Machine learning (ML) models offer reliable solutions for anomaly detection challenges, which are subsequently utilized in fraud detection. Based on the requirements and scale of machine learning, various approaches exist for developing and implementing these models. Recent research in machine learning emphasizes two significant categories of platforms: development platforms and release platforms [6].

E-commerce growth is accelerating even as physical stores reopen. In 2023, global retail e-commerce sales exceeded 5.8$ trillion, and projections indicate that they will reach 6.3$ trillion by 2024, representing 20% of total global retail sales. Credit card fraud rates have generally increased in recent years, driven by increased e-commerce and online credit card use. In 2023, a Javelin Strategy & Research report estimated that 33.8 million Americans fell victim to credit card fraud, resulting in overall losses. Experts predict that the growth of e-commerce and evolving fraud techniques will propel fraud rates to 52$ billion in 2024 [7][8][9].

Electronic companies anticipate a surge in fraudulent transactions globally, resulting in losses of 40.62$ billion by 2027. Fraudulent transactions around the world increased from 9.84$ billion in 2011 to 32.39$ billion in 2020 [10].

There is massive potential in using machine learning techniques to improve fraud detection. Therefore, it is worth exploring the capabilities of hybrid frameworks for this purpose. A lot of research looks at the effects of three different sampling methods: random under-sampling (RUS), synthetic minority sampling technique (SMOTE), and SMOTEENN, which is a mix of oversampling (SMOTE) and under-sampling (editing nearest neighbor). This study innovatively combines SMOTE with CNN and LSTM to tackle class imbalance and achieves marked performance improvements as compared to past literature. The paper applies these data-level resampling strategies to four optimized machine-learning classifiers: logistic regression, decision trees, random forests, and extreme gradient boosting [11].

This paper employs various machine learning (ML) and deep learning (DL) techniques to detect fraudulent credit cards. Firstly preprocess the dataset using two data normalization techniques, Min-Max and z-score, to align all features within a similar range. ML algorithms need to be sensitive to data scale and apply SMOTE as a method for addressing the class imbalance in data, then evaluate performance metrics (accuracy, precision, recall, and F1-score) of the classification model using four classifiers RF and SVM [12] as a machine learning mechanism and CNN and LSTM as a deep learning mechanism).

The remainder of this paper is organized as follows: the second section reviews previous studies on related works, the third section summarizes various methods and provides overviews, the fourth section introduces the proposed model, the fifth section discusses the experimental results and the final section outlines the conclusions and future work.

2. Related Work

Recently, numerous studies have tackled the issue of credit card fraud, as it has become a significant concern in today's world. These studies have focused on detecting these frauds through the design and development of various systems. Below, this paper will discuss several studies that have addressed this problem and the methods used to address it.

Salekshahrezaee et al. [12] conducted this research to explore credit card fraud detection techniques and increase the success rates of machine learning algorithms in detecting such incidents. They used principal component analysis PCA and convolutional autoencoder CAE for feature selection, as well as random under sampling RUS, synthetic minority oversampling technique SMOTE, and SMOTE Tomek techniques to address the class imbalance problem. They used Decision Tree, RF, XGBoost, LightGBM, and CatBoost classifiers, with classification performance measured using the F1 score and F1 score, as well as the Area Under the Receiver Operating Characteristic Curve AUC metric. They found that CAE outperformed PCA due to its ability to model nonlinear relationships, and combining RUS sampling with CAE achieved the best F1-score and AUC.

Alarfaj et al. [13] developed a model that accurately classifies online transactions as fraudulent or genuine. The model utilizes both supervised and unsupervised ML algorithms, such as logistic regression, artificial neural networks ANN, and support vector machines SVM, in the classification process. This paper evaluated the model's performance using various metrics, including accuracy, precision, recall, F1-measure, MCC, ROC curve, and confusion matrix. Based on these evaluations, the model indicates that SVM is the most efficient algorithm for detecting credit card fraud.

Anupama Phakatkar [14] conducted a comparison of three different machine learning models for credit card fraud detection: logistic regression, random forest, and adaboost. The models were evaluated using two datasets, one of which was real data and the other created using BankSim software. This paper discovered that RF outperformed both datasets in every measure. Additionally, recommended using SMOTE and hybrid ensemble techniques to further improve the detection results.

Ileberi et al. [15] focused on comparing supervised machine learning algorithms for credit card fraud detection using an imbalanced dataset. The findings showed that Random Forest (RF) and Decision

Tree (DT) provided the best overall performance, particularly in terms of accuracy and F1-score. However, the study did not offer detailed insights on the selection and impact of the five optimal feature vectors on model performance. Additionally, the use of Genetic Algorithm (GA) for feature selection is computationally intensive, especially with large datasets. The authors recommend addressing the computational efficiency and scalability of the approach and comparing it with less resource-intensive techniques.

Fanai et al. [16]sought to enhance detection accuracy by integrating dimensionality reduction with deep learning techniques. The study found that combining Autoencoder with deep learning models (DNN, RNN, CNN, and RNN_CONV) significantly improved detection accuracy, addressing both dimensionality reduction and class imbalance, and leading to superior performance. AE-based models, including AE-DNN, AE-RNN, and AE-CNN_RNN, exhibited higher recall, precision, F1-score, and AUC metrics, with AE-CNN_RNN achieving the highest overall performance. Nonetheless, a major concern is the lack of interpretability in deep learning models. The study suggests incorporating methods to explain and interpret predictions to enhance practitioners' understanding and trust in the results.

Roseline et al. [17]aimed to develop an enhanced detection system using Long Short-Term Memory (LSTM) recurrent neural networks (RNN) for credit card fraud detection. The LSTM-RNN model demonstrated significant promise, outperforming traditional machine learning methods on the dataset. However, the study falls short in providing detailed insights into managing class imbalance across algorithms and faces challenges in interpreting results from complex models such as LSTM-RNN, Artificial Neural Networks (ANN), and Support Vector Machines (SVM). Furthermore, while Bayesian optimization is mentioned, the paper does not discuss the impact of different hyperparameters on model performance.

Uchhana et al. [18] investigated the use of various machine learning algorithms (SVM, LR, Naïve Bayes, KNN, RF) for credit card fraud detection, utilizing Principal Component Analysis (PCA) for dimensionality reduction. The study found that Random Forest (RF) delivered the highest performance with an MCC of 0.848, indicating that integrating these algorithms with emerging technologies could enhance detection accuracy. Nevertheless, the study does not delve into the computational efficiency and scalability, which are vital for real-world applications. It also suggests that algorithm combinations could boost performance, but lacks supporting evidence. Demonstrating the effectiveness of ensemble or hybrid models would fortify the study.

Khan et al. [19] aimed to create a model for classifying online transactions as either fraudulent or genuine using various machine learning algorithms, including Logistic Regression (LR), Artificial Neural Networks (ANN), and Support Vector Machines (SVM). The study found that supervised machine learning models, particularly SVM, outperformed unsupervised models in all performance metrics, showing better precision, MCC, F1-score, and a ROC curve closer to the ideal point. Nevertheless, the study had several limitations: it featured a narrow comparison of algorithms, lacked a detailed explanation for the chosen resampling technique, encountered challenges in interpreting complex models like ANN and SVM, and did not address computational efficiency and scalability.

Leevy et al. [20] evaluated the effectiveness of binary classification (BCC) versus one-class classification (OCC) in detecting transactions using various classifiers, including CatBoost, XGBoost, Extremely Randomized Trees, Random Forest, Logistic Regression, One-Class SVM, One-Class GMM, and OCAN. They found that BCC outperformed OCC, with CatBoost achieving the highest AUPRC score of 0.8567 among BCC models, signifying superior performance. However, the study has several shortcomings: it requires additional metrics such as precision, recall, and F1-score for a comprehensive assessment. Moreover, it lacks detailed techniques for managing class imbalance, faces challenges in interpreting complex models, and does not address the computational efficiency and scalability essential for real-time applications.

Salekshahrezaee et al. [12]examined the impact of data sampling and feature extraction on ensemble classifiers for detection, employing classifiers such as Random Forest, XGBoost, LightGBM, and CatBoost, with PCA and CAE for feature selection. The study recommended the use of RUS sampling followed by CAE feature extraction for ensemble classifiers. However, the paper does not discuss the interpretability of complex models like CatBoost, XGBoost, LightGBM, and Random Forest. It also provides limited information on the computational efficiency and scalability of the proposed methods, particularly regarding the impact of CAE on reducing computational burden. Additionally, while the study addresses class imbalance, it does not thoroughly explore the effectiveness of techniques such as RUS, SMOTE, and SMOTE Tomek in different scenarios.

Habibpour et al. [21]aimed to disseminate uncertainty quantification (UQ) techniques, such as Monte Carlo dropout (MCD), ensemble, and ensemble Monte Carlo dropout (EMCD), to estimate uncertainty on publicly available transaction data. They found that MCD and ensemble methods are more effective in capturing the corresponding uncertainty of forecasts, providing additional insights into point predictions and enhancing fraud prevention. However, the study briefly discusses MCD, ensemble, and EMCD methods without offering detailed explanations, focuses on UQ techniques without comparing them to other UQ approaches, and lacks detailed dataset information. The use of complex models like deep neural networks (DNNs) presents interpretation challenges. Including metrics such as precision, recall, and F1-score would offer a more comprehensive evaluation.

Phakatkar et al. [14]aimed to identify the optimal machine learning algorithm for credit card fraud detection by comparing Logistic Regression, Adaboost, and Random Forest. The results indicated that Random Forest was the best-performing algorithm, achieving the highest rates across all performance measures. Despite addressing class imbalance with SMOTE, the study does not compare this method to other alternatives. Additionally, the claim that the hybrid ensemble model is superior lacks detailed comparisons with other state-of-the-art algorithms. Incorporating metrics such as precision, recall, and MCC would provide a more comprehensive evaluation.

Maghsood et al. [22]investigated the Nearest Neighbor distance method for detecting credit card fraud, assessing the performance of both supervised and unsupervised methods using classifiers like Brute Force, KD Tree, and Ball Tree. The study revealed that the Brute algorithm is more accurate in identifying the nearest neighbor distances, with supervised methods performing better due to the requirement for labeled data, while unsupervised methods do not need labeled data. Additionally, it emphasized the importance of recall (catching frauds) over precision (avoiding false positives) for financial investigators. However, the study could benefit from incorporating advanced machine learning and deep learning techniques. It lacks discussion on the computational efficiency and scalability of the proposed methods and does not provide detailed explanations or justifications for the chosen distance metrics.

Uddin et al. [23]investigated the impact of feature engineering on the performance of a deep learning model for credit card fraud detection using a CNN with DFS. The study found that DFS significantly enhanced model performance, resulting in notable improvements in accuracy, precision, recall, and F1-score. However, the research relied on undersampling to address class imbalance and should consider techniques like SMOTE or ADASYN to mitigate data loss. Additionally, the study lacks comprehensive comparisons with other baseline models and algorithms, necessitates the implementation of feature selection methods to reduce complexity, and must address the interpretation and validation of CNN model results.

Setiawan et al. [24]sought to develop a model for identifying outliers in credit card transaction data by employing classifiers like HDBSCAN and UMAP, using SMOTE to address class imbalance. The combination of HDBSCAN, UMAP, and SMOTE techniques proved effective in detecting fraudulent transactions and enhancing financial security. However, the study does not provide detailed explanations for choosing methods such as HDBSCAN, UMAP, and SMOTE, nor does it address the computational efficiency and scalability of the proposed methods.

Zhou et al. [25]set out to develop an efficient method for detecting fraudulent financial transactions using a distributed big data approach, employing classifiers such as Node2Vec and Deep Neural Network (DNN) with graph embedding via Node2Vec. The approach's advantage lies in Node2Vec's ability to effectively learn transaction features from the network structure, delivering superior performance in identifying fraudulent financial transactions compared to DeepWalk and SVM. However, the paper falls short in offering a detailed discussion on hyperparameter tuning for Node2Vec and does not address the computational efficiency or scalability of the distributed system leveraging Apache Spark and Hadoop. Additionally, it overlooks techniques to handle class imbalance and would benefit from comparing more recent algorithms beyond Node2Vec, DeepWalk, and SVM.

Here, a comparison was made with several studies that utilized the same dataset as the one used in this work. This allows a better understanding of how our findings align or differ from previous research, providing valuable insights into the effectiveness of our approach and the implications of the results, as shown in Table 1.

Table 1. Summary of literature

Author	Classifier	Feature Selection	Class Imbalance	Optimization	Results
[15] 2022	Decision Tree (DT), RF, ANN, NB, LR	Genetic Algorithm based on RF	SMOTE	Min-max scaling	· RF: accuracy (83.78%) F1-score (85.71%), recall (79.64%) precision (92.78%). · DT: accuracy (89.91%) recall (79.64%), precision (68.70%) F1-score (73.77%). · ANN: accuracy (88.93%) F1-score (80.54%),recall (78.76%) precision (82.40%). · NB: accuracy (78.14%) F1-score (12.46%), recall (83.18%) precision (6.73%). LR: accuracy (79.91%) precision (81.70%), recall (59.29%) F1-score (68.71%).
[16] 2023	Deep Neural Network (DNN), RNN, CNN, combinations (RNN_CONV)	Autoencoder: for reducing original data dimensions (29 to 22) Evaluate performance using holdout and cross-validation methods.	Utilize AUC-ROC and AUC-PR	Bayesian optimization.	· Dimensionality Reduction: AE effectively reduced data dimensions without significant information loss. · Performance Improvement: Compared to original features, AE-based models (AE-DNN, AE-RNN, AE-CNN_RNN) generally achieved better recall, precision, F1-score, and AUC metrics. Best Performing Model: AE-CNN_RNN achieved the highest overall performance across both evaluation methods

[17] 2022	Naive bayes SVM ANN LSTM-RNN	None	Oversampling and Under-sampling techniques.	Each algorithm has its own default optimizer settings within the chosen libraries	· LSTM RNN model was able to achieve an accuracy of 98.38%, a precision of 98.23%, a recall of 98.54%, and an F1-score of 98.38%. · The model was able to reduce the number of false positives compared to other methods.
[18] 2021	SVM, Naive Bayes, LR, KNN, RF	Principal Component Analysis (PCA) for dimen-sionality reduction.	None	Each algorithm has its own default optimizer settings within the chosen libraries	Random Forest gives the highest (MCC) of 0.848. Naive Bayes scores (MCC) of 0.761. Logistic Regression scores (MCC) of 0.761, showing comparable performance to Naive Bayes. KNN scores (MCC) with 0.793. SVM comes last (MCC) with 0.558
[19] 2022	LR, ANN, SVM, MCC, ROC	None	None	Each algorithm has its own default optimizer settings within the chosen libraries	Accuracy and Specificity were the same for all models. Precision: SVM highest, ANN lowest, LR in between.MCC and F1-score: SVM highest, LR lowest, ANN in between.ROC curve: SVM closest to ideal point (0, 1).
[20] 2023	**BCC**: Cat Boost, XGBoost, Extremely Randomized Trees, Random Forest, Logistic Regression **OCC**: One-Class SVM, One-Class GMM, OCAN	None	Excluding fraudulent from OCC models using oversampling or undersam-pling for BCC models,	Each algorithm has its own default optimizer settings within the chosen libraries	BCC outperformed OCC: AUPRC scores significantly higher for BCC models (0.7490 - 0.8567) compared to OCC models (0.3471 - 0.4975). Cat Boost best BCC model: Achieved the highest AUPRC score (0.8567).nOne-Class GMM best OCC model: Achieved the highest AUPRC score (0.4975).
[12] 2023	RF, XGBoost, LightGBM, CatBoost	PCA, CAE	RUS SMOTE SMOTE Tomek	Each algorithm has its own default optimizer settings within the chosen libraries	RUS sampling achieved the best performance among sampling techniques.CAE feature extraction followed by RUS sampling led to the best overall results.

[21] 2023	Decision Tree RF, XGBoos, LightGBM, CatBoost	Deep learning models automatically perform feature extraction and feature learning during the training process	None	Each algorithm has its own default optimizer settings within the chosen libraries	Ensemble method performed best (UAcc=0.85) compared to MCD (0.82) and EMCD (0.84). Both MCD and ensemble improve model confidence calibration compared to the base DNN model
[14] 2022	LR, Adaboost, RF	None	SMOTE Hybrid ensemble models	Each algorithm has its own default optimizer settings within the chosen libraries	Random Forest achieved the highest rates in all performance measures (94% precision, 77% recall and 85 % F1 score) .Logistic Regression (88% precision, 62% recall and 73 % F1 score). AdaBoost (78% precision, 66% recall and 72 % F1 score).
[22] 2023	Brute-Force, KD Tree , Ball Tree	None	None	Each algorithm has its own default optimizer settings within the chosen libraries	More neighbors reduce false negatives (missed frauds) but increase false positives (non-fraud flagged). Standardized "OR" combination of L2 distance and L2 distance to zero was the best overall supervised method. Unsupervised nearest neighbor methods weren't effective in this context.Financial investigators should prioritize recall (catching frauds) over precision (avoiding false positives)
[23] 2023	CNN, DFS	None	Under-sampling techniques.	Adam optimizer,	Accuracy: Increased from 82% to 91% for both fraudulent and legitimate transactions. Precision: Increased by 12% for fraudulent transactions and 11% for legitimate transactions. Recall: Increased by 13% for fraudulent transactions and 11% for legitimate transactions. F1-score: Increased by 11% for both fraudulent and legitimate transactions.
[24] 2023	HDBSCAN, UMAP	None	SMOTE	None	AUC: produces 86% score. Precision: produces 54%. Recall: produces 84%. F1-score: produces 65%

[25] 2021	Node2Vec. Deep Neural Network	graph embedding with Node2Vec	None	None	Precision: Consistently above 70% for Node2Vec, compared to around 60% for DeepWalk and 30% for SVM. Recall: Over 60% for Node2Vec in some tests, reaching near 70%, while DeepWalk ranged from 40% to 50% and SVM was significantly lower. F1-Score: Node2Vec ranged from 67% to 73%, surpassing DeepWalk and SVM. F2-Score: Node2Vec achieved scores higher than 65%, with some tests reaching 71% or close to 70%.

3. Method and overviews

This section outlines the methodology that will be used to detect credit card fraud using machine learning techniques. The subsequent subsections offer a succinct elucidation of the definition of machine learning and the methods employed for data analysis and classification to achieve the ultimate objective.

3.1 Machine Learning (ML)

In the field of artificial intelligence known as "machine learning," computers pick up knowledge from their experiences without needing explicit programming. Differently, ML algorithms find patterns in huge datasets and forecast based on those patterns using statistical models and algorithms. ML comes in a variety of forms, such as online learning, transfer learning, reinforcement learning, unsupervised learning, semi-supervised learning, and transfer learning. Image recognition, audio recognition, natural language processing, fraud detection, spam filtering, recommendation engines, medical diagnosis, customer segmentation, product suggestions, and drug development are just a few of the numerous uses for machine learning [23].

3.1.1 Supervised learning

This type of ML algorithm provides labeled training data to train the model with features and their corresponding labels. There are several commonly used supervised ML depending on the problem at hand, like logistic regression, decision trees, RF, SVM, naive Bayes, k-nearest neighbors (k-NN), and artificial neural networks (ANN) [22]. In this paper, RF and SVM algorithms are implemented.

1. Random Forests (RF): is an ensemble learning method that combines decision trees to generate more accurate and strong predictions. Every tree in the forest is grown separately, and the majority vote of the trees is used to determine the final projection. Random Forests are popular because they are effective in many tasks, easy to use, and good at avoiding overfitting the data. However, training a Random Forest can be computationally expensive compared to some other algorithms [26].
2. Support Vector Machines (SVMs): Are a strong and well-established machine learning technique, particularly for classification tasks. SVM uses two types of classifiers - Linear SVM and Non-Linear SVM. It is very effective when dealing with a large number of features and high-dimensional data. SVM can classify non-linear data using kernel tricks. It also works well with text classification problems, such as spam or ham classification.

3.1.2 Semi-supervised learning

Also known as weakly supervised learning, is an approach that trains classifiers using only some of the available labeled data. Examples of such algorithms include self-training, co-training, multitask learning, transitive inference, and domain adaptation [22].

3.1.3 Unsupervised learning

It is an algorithm that does not rely on labeled data but instead requires the model to find patterns in the data itself. Clustering is an unsupervised learning technique that is used to group similar instances according to certain criteria. On the other hand, classification is a technique of supervised learning that focuses on predicting discrete outcomes [27].

3.1.4 Reinforcement learning

Aims to maximize the rewards received by the agent over time and is well-suited for complex and dynamic applications such as robotics and gaming. There are three types of reinforcement learning implementations: policy-based, value-based, and model-based. Reinforcement learning algorithms can also be used in healthcare and finance to optimize treatment plans and investment strategies, respectively. Given its potential benefits, it is a promising field for future research and development [23].

3.2 Deep Learning (DL)

Subfield of machine learning that utilizes artificial neural networks with multiple layers. The term "deep" refers to the use of many layers in the network architecture. This approach can achieve high accuracy when learning from large amounts of data and is often more effective in fraud detection. In this work, various deep-learning algorithms are used to achieve our goals [6]:

1. Convolutional Neural Networks (CNNs): a particular type of artificial neural network, a very powerful tool due to their ability to handle large amounts of complex data, they are designed to automatically extract features from the data they are analyzing. Also, use CNNs with signal or time series data when pre-processed to work with the network architecture.
2. Long short-term memory networks (LSTMs): It is a unique kind of recurrent neural network RNN that can resolve the issue of disappearing gradients that RNNs encounter. Hochreiter and Schmidhuber created the LSTM, which addresses issues with conventional RNNs and machine learning algorithms. The Keras package may be used to implement the LSTM model in Python.

3.3 Data Pre-processing

3.3.1 Data Normalization

Normalization is the process of organizing data in a database or information system so that it can be easily accessed, managed, and updated. It involves transforming raw data into a consistent format without losing critical information and ensuring its accuracy. The main goal behind this process is to reduce redundancy and improve overall efficiency by enabling quick access to relevant information. The popular methods of data normalization :

1. Min-Max: for converting the original data range to a range between (0 and 1). This allows us to compare various features together and get more appropriate data because all data will be on the same scale.
2. Z-Score: also called standardization, this technique depends on transforming a numerical dataset by scaling each value to have a mean of 0 and a standard deviation of 1. So, it converts the data into standard units, where authors can easily compare the relative positions of different data points [28].

3.3.2 Synthetic Minority over Sampling Technique (SMOTE)

Synthetic Minority Over-sampling Technique is what it stands for. It is a well-liked data augmentation method for machine learning that addresses the issue of performance degradation and class imbalance. A dataset that is class-imbalanced has a notably smaller number of samples in one class than the other. By generating new instances along the line segments connecting preexisting examples of the minority class, SMOTE produces synthetic examples of the class. This enhances machine learning models' performance and helps to balance the distribution of classes, particularly in binary classification issues as shown in equation (3), and (figure 2).

4. The proposed model

This paper implements ML and DL techniques for detecting fraud on credit cards. The proposed model was applied to a real-world e-commerce dataset, effectively identifying fraudulent transactions with a 95% accuracy, highlighting its practical viability The main work steps of this paper can be summarized as follows (figure 1):

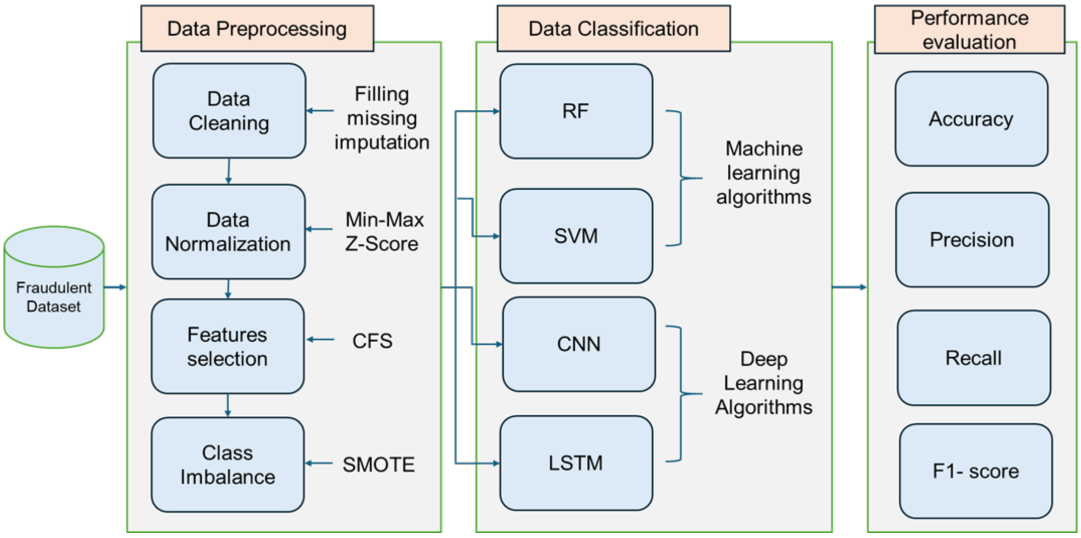

Figure 1. The building Block of Proposed Model

4.1 Data description

· **Data Source:** The dataset was collected from European cardholders over two days in September 2013. It represents a real-world scenario of credit card transactions and fraud detection.

· **Number of Samples:** The dataset consists of a total of 284,807 transactions, providing a substantial amount of data to analyze.

· **Fraudulent Transactions Percentage:** Out of the total transactions, 492 are identified as fraudulent, which constitutes approximately 0.172% of the dataset. This highlights the significant class imbalance, with fraudulent transactions being vastly outnumbered by legitimate ones.

· **Features:** The dataset includes 28 features, labeled V1 through V28, which are the result of a PCA (Principal Component Analysis) transformation to protect the confidentiality of the users. Additionally, there are two more features: `Amount`, which represents the transaction amount, and `Class`, which indicates whether the transaction is fraudulent (1) or legitimate (0) [29].

4.2 Data Preprocessing

4.2.1 Feature Selection

To choose a subset of characteristics from the original dataset that are most pertinent to the classification job, authors utilize an open-source software tool called Weka. Chosen two attribute evaluators (InfoGainAttributeEval, OneAttributeEval) and two search methods (attribute Ranking, attribute Ranking), the two experiments gave the same result when choosing the (Time and Amount) features.

4.2.2 Data Normalization

In this paper, two data normalization techniques was chosen : Min-Max Normalization: equation (1) [15]:

$$x_{normalized} = \frac{x - x_{min}}{x_{max} - x_{min}} \tag{1}$$

where: X : the original value that we want to normalize. X -min: the smallest value in the dataset. X-max : the largest value in the dataset. X-normalized : the resulting normalized value.

Z-Score Normalization: equation (2) [23]:

$$z = \frac{(x - \mu)}{\sigma} \tag{2}$$

where: x: the original value of the feature. μ: the mean of the feature. σ: the standard deviation of the feature.

4.2.3 Class Imbalance (SMOTE)

To improve the performance of (ML) models and to help balance the distribution of classes, SMOTE generates synthetic examples of the class. This is especially useful for binary classification problems. The following equation shows the process of SMOTE (equation 3) [28].

$$x_{new} = x_i + rand(0, 1) \times \left(x_{neighbours} - x_i\right) \tag{3}$$

Before Oversampling:

• Fraudulent Transactions (Label 1): 492

• True Transactions (Label 0): 284,807

After Oversampling:

• Fraudulent Transactions (Label 1): 284,807

• True Transactions (Label 0): 284,807

To illustrate the importance of using SMOTE, the paper considers (figure 2) depicting the frequency distribution of two classes, "0" and "1." Class "0" is significantly more frequent, approximately 25,000 times, while class "1" has nearly zero occurrences. This imbalance can lead to biased model performance, with a tendency to predict the majority class and overlook the minority class. SMOTE addresses this by generating synthetic samples for the minority class, balancing the data, and ensuring more accurate and fair classification models.

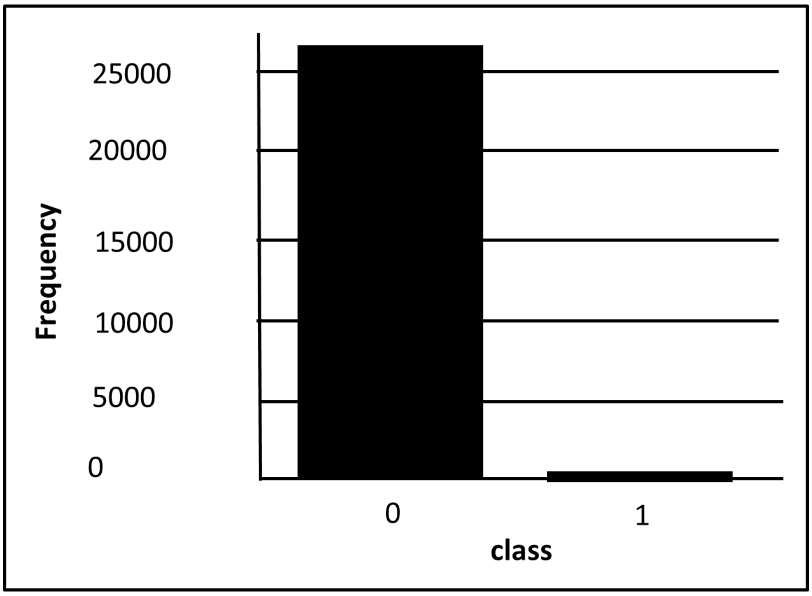

Figure 2. fraudulent against genuine transactions

4.3 Data Classification

This section outlines the methodology employed to preprocess the dataset, apply the SMOTE, and perform classification using ML algorithms like RF Classifier, SVMs, and DL algorithms like LSTMs and CNNs. The classification process paper appllied to the dataset before applying SMOTE and after applying SMOTE. Also, this section explains the CNN model and LSTM model which had been designed for classification tasks.

1. CNN Model Architecture :
   (a) **Input Layer:** The first layer that accepts pre-processed transaction information.
   (b) **Convolutional Layer 1:** This layer uses 32 filters of size 3x3 with ReLU activation to detect features from the input data.
   (c) **Max-Pooling Layer 1:** Downsamples the feature maps using a filter of size 2x2 to reduce the spatial dimensions and retain important features.
   (d) **Convolutional Layer 2:** Another convolutional layer with 64 filters of size 3x3 and ReLU activation for extracting more complex features.
   (e) **Max-Pooling Layer 2:** Further downsampling with a 2x2 sized filter to reduce dimensionality and prevent overfitting.
   (f) **Flatten Layer:** Transforms the 2D feature maps into a 1D vector, preparing the data for the fully connected layers.
   (g) **Fully Connected Layer 1:** Contains 128 neurons with ReLU activation, performing high-level reasoning over the features.
   (h) **Fully Connected Layer 2:** Includes 64 neurons activated using ReLU for further feature transformation.
   (i) **Output Layer:** A single neuron with sigmoid activation for binary classification, outputting a probability score.

   Training Configuration:
   - **Epochs:** 50 The number of times the entire training dataset is passed through the model.
   - **Optimizer:** Adam An adaptive learning rate optimization algorithm that is efficient and widely

used in training deep learning models.

- **Learning Rate:** 0.001 Controls the step size for updating the model parameters.
- **Batch Size:** 64 Number of samples processed before the model is updated.
- **Loss Function:** Binary Cross-Entropy Measures the performance of a classification model whose output is a probability value between 0 and 1.

2. LSTM Model Architecture
   (a) **Input Layer:** The first layer that accepts pre-processed sequence data, such as time series or text data.
   (b) **LSTM Layer 1:** This layer contains 50 units (neurons) and uses ReLU activation function to capture the temporal dependencies in the data.
   (c) **Dropout Layer 1:** To prevent overfitting, this layer randomly sets a fraction of the input units to 0 with a dropout rate of 0.2.
   (d) **LSTM Layer 2:** Another LSTM layer with 50 units and ReLU activation, for capturing more complex patterns and long-term dependencies.
   (e) **Dropout Layer 2:** Similar to the first dropout layer, this one also uses a dropout rate of 0.2 to improve model generalization.
   (f) **Flatten Layer:** This layer transforms the 2D output of the LSTM layers into a 1D vector.
   (g) **Fully Connected Layer 1:** This layer has 100 neurons and employs ReLU activation for further processing the features extracted by the LSTM layers.
   (h) **Fully Connected Layer 2:** Another dense layer with 50 neurons and ReLU activation for additional feature extraction and transformation.
   (i) **Output Layer:** Contains a single neuron that performs sigmoid activation for binary classification.

   Training Configuration:

   - **Epochs:** 50 The number of times the entire training dataset is passed through the model.
   - **Optimizer:** Adam An adaptive learning rate optimization algorithm that's efficient and widely used in training deep learning models.
   - **Learning Rate:** 0.001 Controls the step size for updating the model parameters.
   - **Batch Size:** 64 Number of samples processed before the model is updated.
   - **Loss Function:** Binary Cross-Entropy Measures the performance of a classification model whose output is a probability value between 0 and 1.

4.4 Evaluation metrics

Data Splitting:

· **Training and Testing Ratio:** To evaluate the model's performance accurately, the dataset was split into two subsets: 80% for training and 20% for testing. This ratio ensures a robust training process while maintaining a sufficient amount of data for testing.

· **Splitting Method:** Random splitting was employed to ensure that the data is divided in a manner that preserves the representativeness of the entire dataset. Random splitting helps in minimizing any biases that could arise from a non-random division.

Stratified Splitting: Stratified splitting was utilized to maintain the same proportion of fraudulent and legitimate transactions in both the training and testing sets. This approach ensures that both sets are representative of the original data's class distribution. By preserving the class distribution, the model can be effectively trained and evaluated on a balanced representation of the data.

Performance evaluated ML algorithms (RF, SVMs, CNN, LSTMs), by using four measures: Accuracy, precision, recall, and F1 score Equations 4,5,6) [15].. Two normalization techniques (Min-Max Normalization and Z-Score Normalization) were applied to the data. Accuracy, precision, recall, and F1score ,are often preferred because they provide a balanced and detailed evaluation of model performance, especially for imbalanced datasets. These metrics help to understand both the correctness and completeness of the predictions, making them more suitable for real-world applications where missing important cases (like fraud) can be costly. The experiment was applied twice, first by applying SMOTE to the data that had been normalized, and second without applying SMOTE to determine whether using SMOTE affects the accuracy results or not .

In this paper, we have explored the configurations and applications of various machine learning algorithms, including RF, SVM,CNN, and LSTM, each selected for their specific strengths and suitability for distinct tasks. Table 2 highlights configurations for them the parameters like learning rate, epochs, and optimizer are not applicable as it builds multiple decision trees. For RF parameters like learning rate, epochs, and optimizer are not applicable as it builds multiple decision trees. SVM employs the Radial Basis Function (RBF) kernel with a regularization parameter (C) of 1 and a kernel coefficient (gamma) set to scale. For CNN, a learning rate of 0.001, 50 epochs, and the Adam optimizer are used to adjust model parameters efficiently. Similarly, LSTM also utilizes a learning rate of 0.001, 50 epochs, and the Adam optimizer for training. These configurations are tailored for different machine learning tasks, with Random Forest and SVM suited for classification, and CNN and LSTM for handling image and sequence data, respectively.

$$Accuracy = \frac{TP + TN}{TP + TN + FP + FN} \tag{4}$$

$$Precision = \frac{TP}{TP + FP} \tag{5}$$

$$Recall = \frac{TP}{TP + FN} \tag{6}$$

Table 2. The parameters and hyperparameter for each technique

Algorithm	*Learning Rate*	*Epochs*	*Optimizer*
CNN	*0.001*	*50*	*Adam*
LSTM	*0.001*	*50*	*Adam*
Random Forest	*N/A*	*N/A*	*N/A*
SVM	*Kernel = RBF*	*C = 1*	*Gamma = Scale*

5. Experimental results

5.1 Results and Discussion

5.1.1 Results:

Examining the results without using the SMOTE technique reveals significant variations in the performance of different algorithms table 3. For RF, the results are fairly good, showing high accuracy and a balanced trade-off between precision and recall. Under Min-Max normalization, an accuracy is about 0.78, precision 0.77, recall 0.83, and an F1-score 0.80. These metrics indicate a well-performing model with a strong ability to correctly identify fraudulent transactions, though its performance drops slightly with Z-Score normalization to an accuracy of 0.75, precision of 0.75, recall of 0.80, and an F1-score of 0.75. This shows that while (RF)is effective, it is sensitive to the type of normalization used.

SVMs, on the other hand, show poorer performance without SMOTE. With Min-Max normalization, SVMs achieve an accuracy of 0.65, precision of 0.62, recall of 0.55, and an F1-score of 0.57. These values indicate significant struggles in detecting true positives, as evidenced by the low recall. However, with Z-Score normalization, SVMs improve to an accuracy of 0.71, precision of 0.73, recall of 0.67, and an F1-score of 0.71, highlighting that normalization methods can substantially impact SVM performance, but the results are still not ideal for imbalanced datasets.

CNNs demonstrate superior performance, with Min-Max normalization resulting in an accuracy of 0.88, precision of 0.86, recall of 0.86, and an F1-score of 0.87. These high scores suggest that CNNs handle the data effectively even without balancing techniques. With Z-Score normalization, CNNs maintain strong performance with an accuracy of 0.85, precision of 0.80, recall of 0.85, and an F1-score of 0.84, indicating robustness and reliability in fraud detection tasks.

LSTMs show moderate performance, with Min-Max normalization giving an accuracy of 0.72, precision of 0.73, recall of 0.72, and an F1-score of 0.71. These balanced metrics suggest a reliable but not exceptional model. Z-Score normalization enhances their performance slightly to an accuracy of 0.78, precision of 0.77, recall of 0.76, and an F1-score of 0.76, showing that LSTMs can benefit from this normalization but still do not match CNN performance.

Overall, CNNs stand out as the best performers, illustrating their robustness in handling such data even without balancing techniques. The differences in performance between Min-Max and Z-Score normalization for the other algorithms underscore the impact of normalization methods. These results emphasize the need for balancing techniques like SMOTE to enhance performance, especially for algorithms like SVMs, which struggle with imbalanced data. The key takeaway is that using SMOTE could significantly improve model performance by achieving a better balance between precision and recall.

Table 3. Results without SMOTE

Method	Min-Max Normalization				Z-Score Normalization			
Algorithm	**Accuracy**	**Precision**	**Recall**	**F1-score**	**Accuracy**	**Precision**	**Recall**	**F1-score**
Random Forest	0.78	0.77	0.83	0.80	0.75	0.75	0.80	0.75
SVMs	0.65	0.62	0.55	0.57	0.71	0.73	0.67	0.71
CNN	0.88	0.86	0.86	0.87	0.85	0.80	0.85	0.84
LSTMs	0.72	0.73	0.72	0.71	0.78	0.77	0.76	0.76

Applying SMOTE to balance the dataset has clearly had a significant impact on the performance of all algorithms, as demonstrated by the improved results across accuracy, precision, recall, and F1-score in Table 4.

For RF, the application of SMOTE has markedly enhanced its performance. With Min-Max normalization, an accuracy is about 0.90, precision of 0.88, recall of 0.88, and an F1-score of 0.89. These metrics indicate a well-balanced model with strong ability to correctly identify fraudulent transactions without sacrificing too much precision. The performance further improves with Z-Score normalization, achieving an accuracy of 0.92, precision of 0.92, recall of 0.91, and an F1-score of 0.93. This showcases Random Forest's robustness and adaptability, making it an excellent choice for fraud detection tasks.

SVMs: also show substantial improvement with SMOTE. Under Min-Max normalization, SVMs achieve an accuracy of 0.78, precision of 0.78, recall of 0.80, and an F1-score of 0.76. These results are significantly better compared to those without SMOTE, highlighting SMOTE's efficacy in handling imbalanced data. With Z-Score normalization, SVMs perform even better, attaining an accuracy of 0.85, precision of 0.79, recall of 0.80, and an F1-score of 0.87. The increased recall and F1-score indicate that SVMs are now more effective in identifying true positives while maintaining a good balance with precision.

CNN: consistently show high performance with the use of SMOTE. With Min-Max normalization, CNNs achieve an accuracy of 0.88, precision of 0.87, recall of 0.88, and an F1-score of 0.88. These scores suggest a highly capable model for fraud detection. When applying Z-Score normalization, CNNs perform even better, reaching an impressive accuracy of 0.95, precision of 0.95, recall of 0.94, and an F1-score of 0.94. The near-perfect metrics underscore CNN's strength and reliability in detecting fraud, making it a top performer in this context.

LSTMs: also benefit from SMOTE. Under Min-Max normalization, LSTMs achieve an accuracy of 0.85, precision of 0.84, recall of 0.87, and an F1-score of 0.85. These results are considerably higher compared to those without SMOTE, demonstrating improved detection capabilities. With Z-Score normalization, LSTMs further enhance their performance with an accuracy of 0.88, precision of 0.89, recall of 0.88, and an F1-score of 0.88. This consistency across different metrics indicates that LSTMs are now better equipped to handle imbalanced datasets effectively.

Table 4. Results with SMOTE

Method	**Min-Max Normalization**				**Z-Score Normalization**			
Algorithm	**Accuracy**	**Precision**	**Recall**	**F1-score**	**Accuracy**	**Precision**	**Recall**	**F1-score**
Random Forest	0.9	0.88	0.88	0.89	0.92	0.92	0.91	0.93
SVMs	0.78	0.78	0.8	0.76	0.85	0.79	0.8	0.87
CNN	0.88	0.87	0.88	0.88	0.95	0.95	0.94	0.94
LSTMs	0.85	0.84	0.87	0.85	0.88	0.89	0.88	0.88

5.1.2 Discussion:

This research presents several novel contributions to the field of fraud detection in the context of e-commerce. The key contributions include:

1. Comprehensive Evaluation of Data Preprocessing Techniques: This study provides a thorough evaluation of various data preprocessing techniques, including data cleaning, normalization, feature selection, and the application of the SMOTE technique, specifically addressing the challenge of imbalanced data in fraud detection.

2. Comparison of Machine Learning and Deep Learning Algorithms: The research systematically compares the performance of multiple algorithms ML and DL, such as SVM, RF, CNN, and LSTM, to detect fraudulent transactions after data preprocessing.

3. Integration of Advanced SMOTE Variants: This study incorporates advanced SMOTE variants, such as Borderline-SMOTE and ADASYN, to enhance data balance and improve the detection of complex fraudulent patterns.

4. Empirical Results and Insights: The research provides empirical results demonstrating the effectiveness of the proposed approaches. The CNN model, in particular, achieved the highest performance with an accuracy of 95% and an F1-score of 94%, showcasing significant improvements over other models.

5. Practical Implications for Fraud Detection Systems: The findings offer valuable insights and practical implications for developing more effective fraud detection systems, highlighting the importance

of advanced data preprocessing techniques in enhancing model performance.

The ROC curve, Figure 3, illustrates the performance of four models, namely RF, SVM, CNN, and LSTM before (Pre) and after (Post) applying SMOTE. The ROC curve is a graphical plot that illustrates the diagnostic ability of a binary classifier system, plotting the true positive rate (sensitivity) against the false positive rate (1-specificity).

The legend indicates the models and their AUC (Area Under the Curve) values before and after SMOTE:

- Random Forest (Pre SMOTE, AUC = .78)
- SVM (Pre SMOTE, AUC = .71)
- CNN (Pre SMOTE, AUC = .89)
- LSTM (Pre SMOTE, AUC = .85)
- Random Forest (Post SMOTE, AUC = .92)
- SVM (Post SMOTE, AUC = .85)
- CNN (Post SMOTE, AUC = .95)
- LSTM (Post SMOTE, AUC = .93)

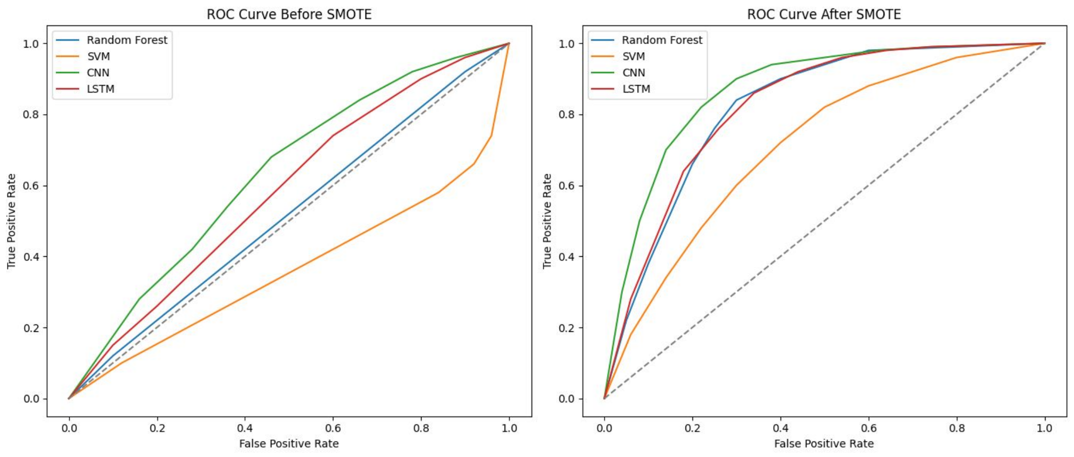

Figure 3. ROC Curve

The legend in the plot indicates the line styles and colors corresponding to each model and condition (before and after SMOTE). The ROC curves are used to evaluate the performance of the models, with the Area Under the Curve (AUC) values provided for each model. The comparison shows how the application of SMOTE affects the performance of the models. Specifically, the AUC values for Random Forest and SVM slightly improved after applying SMOTE, indicating better classification performance. However, the AUC values for CNN and LSTM slightly decreased, suggesting that SMOTE had a less favorable impact on these models. This comparison highlights the importance of balancing classes and selecting appropriate techniques to improve the performance of different ML models in credit card fraud detection .

6. Conclusion and future work

The authors have gone through all the results and found that using SMOTE leads to improve the performance of all algorithms and normalization methods. Accuracy, precision, recall, and F1-score generally gave high results with SMOTE. CNN achieved the best results among all algorithms in both scenarios (with and without SMOTE). Random Forest comes in second, followed by LSTMs and then SVMs. Although using SMOTE improves SVM performance, it doesn't reach the level of CNN or Random Forest. Overall, it is important to note that the best-performing algorithm can vary depending on the specific dataset and task. Summary: Overall, the application of SMOTE has proven to be highly beneficial for all the algorithms, enhancing their ability to detect fraudulent transactions effectively. CNNs emerge as the top performer with the highest metrics across the board, particularly with Z-Score normalization, closely followed by Random Forest. The substantial improvements in SVM and LSTM performances also illustrate the importance of using balancing techniques in fraud detection tasks. These results highlight that SMOTE, combined with appropriate normalization methods, can significantly improve model performance, making fraud detection systems more reliable and efficient Finally, future research could include:

- Explore using variations of SMOTE like Borderline-SMOTE or ADASYN that focus on oversampling near decision boundaries, potentially improving minority class representation.
- Combining SMOTE with other methods for imbalanced learning (e.x cost-sensitive learning algorithms).
- Combining deep learning algorithms to achieve high-performance measurement results.

Open data statement

Availability of data: Credit Card Cheating Detection Dataset

References

[1] Dhiya Al-Jumeily, Abir Hussain, Áine MacDermott, Hissam Tawfik, Gemma Seeckts, and Jan Lunn. "The development of fraud detection systems for detection of potentially fraudulent applications". In: *2015 International Conference on Developments of E-Systems Engineering (DeSE)*. IEEE. 2015, pp. 7–13.

[2] Zhenchuan Li, Guanjun Liu, and Changjun Jiang. "Deep representation learning with full center loss for credit card fraud detection". In: *IEEE Transactions on Computational Social Systems* 7.2 (2020), pp. 569–579.

[3] B Vivekanadam. "Analysis of recent trend and applications in block chain technology". In: *Journal of ISMAC* 2.04 (2020), pp. 200–206.

[4] Lutao Zheng, Guanjun Liu, Chungang Yan, Changjun Jiang, Mengchu Zhou, and Maozhen Li. "Improved TrAdaBoost and its application to transaction fraud detection". In: *IEEE Transactions on Computational Social Systems* 7.5 (2020), pp. 1304–1316.

[5] Xinwei Zhang, Yaoci Han, Wei Xu, and Qili Wang. "HOBA: A novel feature engineering methodology for credit card fraud detection with a deep learning architecture". In: *Information Sciences* 557 (2021), pp. 302–316.

[6] Salma El Hajjami, Jamal Malki, Alain Bouju, and Mohammed Berrada. "Machine learning facing behavioral noise problem in an imbalanced data using one side behavioral noise reduction: application to a fraud detection". In: *International Journal of Computer and Information Engineering* 15.3 (2021), pp. 194–205.

[7] Vijay Kumar. "Digital enablers". In: *The Economic Value of Digital Disruption: A Holistic Assessment for CXOs*. Springer, 2023, pp. 1–110.

[8] Mario Bojilov. “Methods for assisting in detection of synthetic identity fraud in credit applications in financial institutions”. PhD thesis. CQUniversity, 2023.

[9] Susan Sproule, Norman P Archer, et al. “Measuring identity theft in Canada: 2006 consumer survey”. In: (2008).

[10] Merchant Savvy. “Global Payment Fraud Statistics, Trends & Forecasts”. In: *Recovered from https://www. merchantsavvy. co. uk/paymentfraud-statistics* (2020).

[11] Yann-Aël Le Borgne, Wissam Siblini, Bertrand Lebichot, and Gianluca Bontempi. “Reproducible machine learning for credit card fraud detection-practical handbook”. In: *Université Libre de Bruxelles* (2022).

[12] Zahra Salekshahrezaee, Joffrey L Leevy, and Taghi M Khoshgoftaar. “The effect of feature extraction and data sampling on credit card fraud detection”. In: *Journal of Big Data* 10.1 (2023), p. 6.

[13] Fawaz Khaled Alarfaj, Iqra Malik, Hikmat Ullah Khan, Naif Almusallam, Muhammad Ramzan, and Muzamil Ahmed. “Credit card fraud detection using state-of-the-art machine learning and deep learning algorithms”. In: *IEEE Access* 10 (2022), pp. 39700–39715.

[14] Sayali Saraf and Anupama Phakatkar. “Detection of Credit Card Fraud using a Hybrid Ensemble Model”. In: *International Journal of Advanced Computer Science and Applications* 13.9 (2022), pp. 464–474.

[15] Emmanuel Ileberi, Yanxia Sun, and Zenghui Wang. “A machine learning based credit card fraud detection using the GA algorithm for feature selection”. In: *Journal of Big Data* 9.1 (2022), p. 24.

[16] Hosein Fanai and Hossein Abbasimehr. “A novel combined approach based on deep Autoencoder and deep classifiers for credit card fraud detection”. In: *Expert Systems with Applications* 217 (2023), p. 119562.

[17] J Femila Roseline, GBSR Naidu, V Samuthira Pandi, S Alamelu alias Rajasree, and N Mageswari. “Autonomous credit card fraud detection using machine learning approach”. In: *Computers and Electrical Engineering* 102 (2022), p. 108132.

[18] NR Uchhana, R Ranjan, S Sharma, D Agrawal, and A Punde. “Literature review of different machine learning algorithms for credit card fraud detection”. In: *International Journal of Innovative Technology and Exploring Engineering (IJITEE) ISSN* (2021), pp. 2278–3075.

[19] Shahnawaz Khan, Abdullah Alourani, Bharavi Mishra, Ashraf Ali, and Mustafa Kamal. “Developing a credit card fraud detection model using machine learning approaches”. In: *International Journal of Advanced Computer Science and Applications* 13.3 (2022).

[20] Joffrey L Leevy, John Hancock, and Taghi M Khoshgoftaar. “Comparative analysis of binary and one-class classification techniques for credit card fraud data”. In: *Journal of Big Data* 10.1 (2023), p. 118.

[21] Maryam Habibpour, Hassan Gharoun, Mohammadreza Mehdipour, AmirReza Tajally, Hamzeh Asgharnezhad, Afshar Shamsi, Abbas Khosravi, and Saeid Nahavandi. “Uncertainty-aware credit card fraud detection using deep learning”. In: *Engineering Applications of Artificial Intelligence* 123 (2023), p. 106248.

[22] Ramin Maghsood. “Credit Card Fraud Detection by Nearest Neighbor Algorithms”. In: (2023).

[23] Mohammed Najim Uddin, Salahuddin Azad, Md Rahat Hossain, and Ritesh Chugh. “Impact of Deep Feature Synthesis on Deep Learning in Electronic Transaction Fraud Detection”. In: *2023 IEEE 3rd International Conference on Software Engineering and Artificial Intelligence (SEAI)*. IEEE. 2023, pp. 204–208.

[24] Rudy Setiawan, Budi Tjahjono, Gerry Firmansyah, and Habibullah Akbar. “Fraud Detection in Credit Card Transactions Using HDBSCAN, UMAP and SMOTE Methods”. In: *International Journal of Science, Technology & Management* 4.5 (2023), pp. 1333–1339.

[25] Hangjun Zhou, Guang Sun, Sha Fu, Linli Wang, Juan Hu, and Ying Gao. “Internet financial fraud detection based on a distributed big data approach with node2vec”. In: *Ieee Access* 9 (2021), pp. 43378–43386.

[26] Quentin Fournier and Daniel Aloise. “Empirical comparison between autoencoders and traditional dimensionality reduction methods”. In: *2019 IEEE Second International Conference on Artificial Intelligence and Knowledge Engineering (AIKE)*. IEEE. 2019, pp. 211–214.

[27] TH Gan, J Kanfoud, H Nedunuri, A Amini, and G Feng. “Industry 4.0: why machine learning matters?” In: *Advances in Condition Monitoring and Structural Health Monitoring: WCCM 2019*. Springer. 2021, pp. 397–404.

[28] Muhammad Waqar, Hassan Dawood, Hussain Dawood, Nadeem Majeed, Ameen Banjar, and Riad Alharbey. “An Efficient SMOTE-Based Deep Learning Model for Heart Attack Prediction”. In: *Scientific Programming* 2021.1 (2021), p. 6621622.

[29] Kaggle. *Credit Card Fraud Detection: Predictive Models*. https://www.kaggle.com/code/gpreda/credit-card-fraud-detection-predictive-models. n.d.